Tensor Factorizations, Data Fusion & Applications

Evrim Acar

Simula Metropolitan Center for Digital Engineering

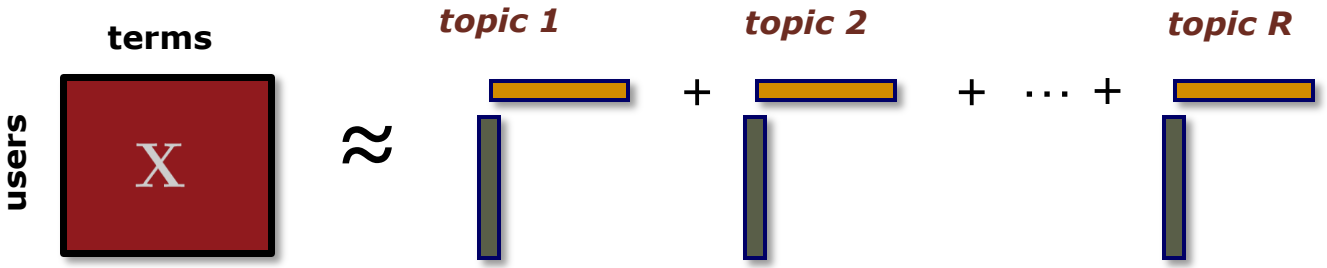
Oslo, Norway

July 2, 2018, LVA/ICA

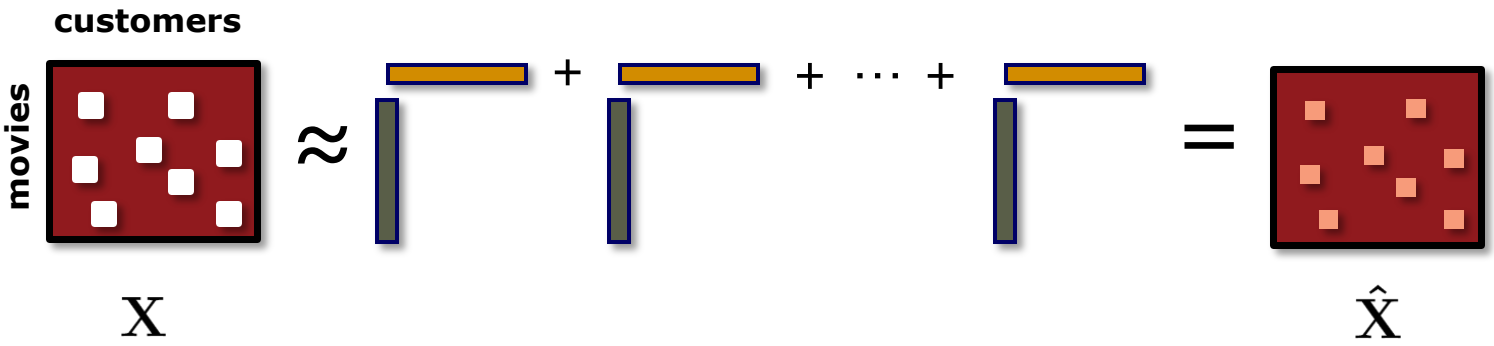
simula

Matrix Factorizations in Data Mining

Capturing underlying hidden factors/patterns



Incomplete Matrix Factorization/ Matrix Completion



simula

Data is often squeezed to be two-way!

2-way

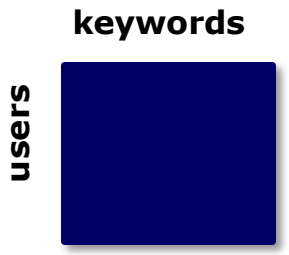
Social networks : **<users, keywords>**
<employee, employee>
<users, users>

Text mining: **< documents, terms>**

Recommender Systems: **<customers, items>**

Computer vision: **<people, pixels>**

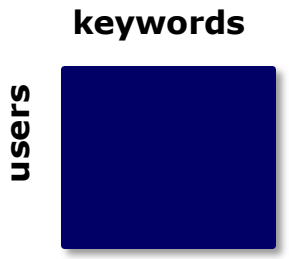
Neuroscience: **<electrodes, time>**

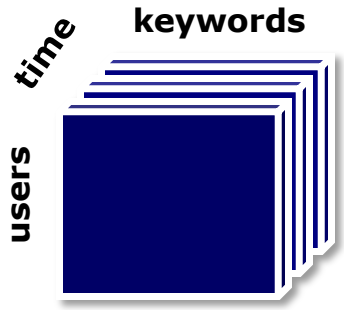


simula

Data is often squeezed to be two-way!

	2-way	Multi-way
Social networks :	**<users, keywords>**	**< users, keywords, time>**
	<employee, employee>	**< employee, employee, time>**
	<users, users>	**< users, users, communication type>**
Text mining:	**< documents, terms>**	**< documents, terms, terms>**
Recommender Systems:	**<customers, items>**	**<customers, items, tags>**
Computer vision:	**<people, pixels>**	**<people, pixels, viewpoints>**
Neuroscience:	**<electrodes, time>**	**<electrodes, time, frequency>**
		<electrodes, time, subjects>

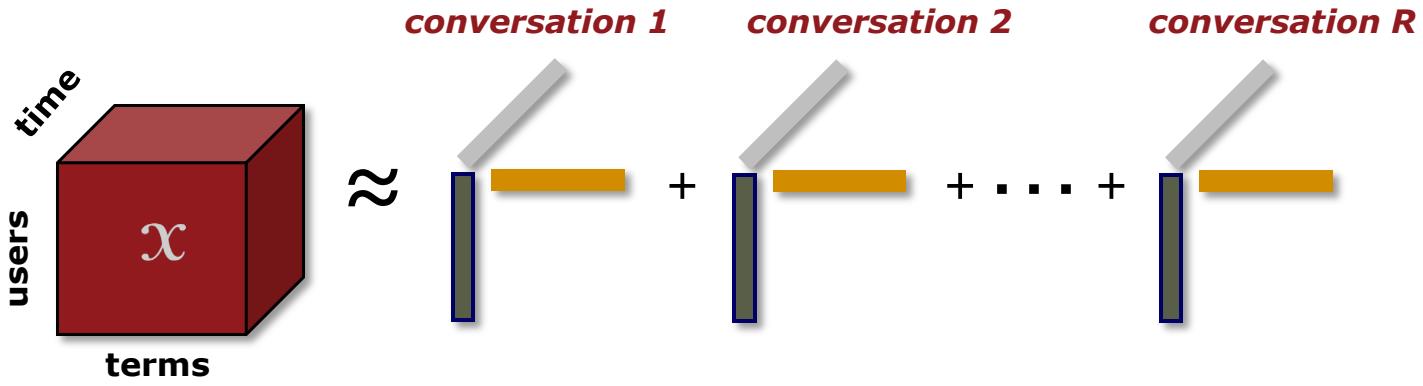




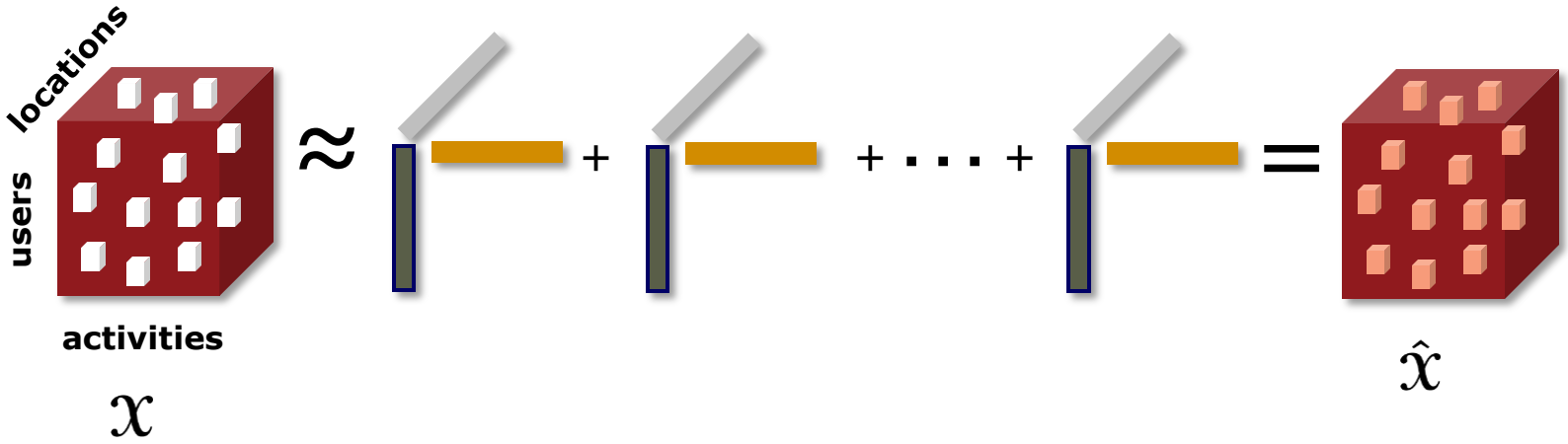
simula

Data is often multi-way!

Capturing underlying hidden factors/patterns



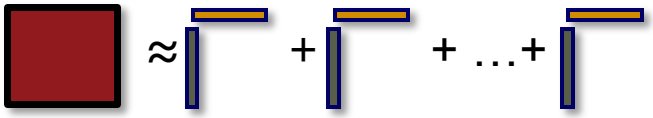
Incomplete Tensor Factorization/ Tensor Completion



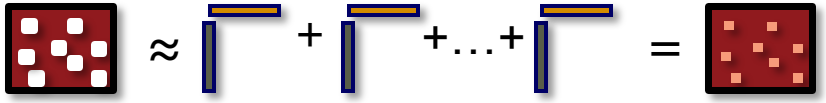
simula

Matrix Factorizations in Data Mining

Matrix Factorizations

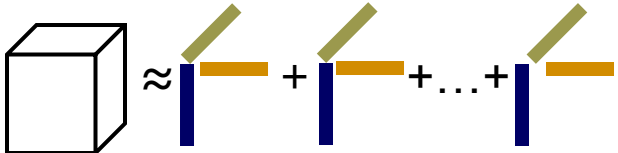


Matrix Completion

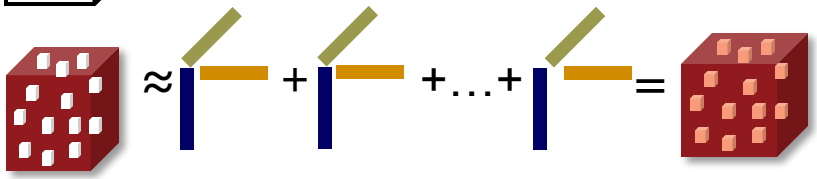


Data sets are often multi-way: Tensor Factorizations

Tensor Factorizations



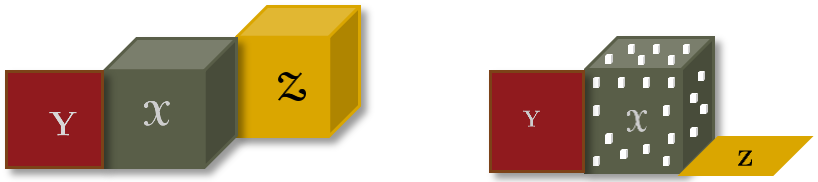
Tensor Completion



Data sets often come from multiple sources: Data Fusion

Coupled Matrix Factorizations

Coupled Tensor Factorizations

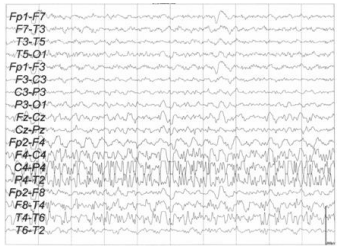


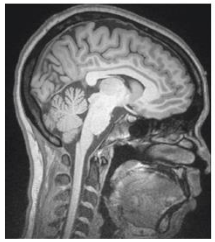
Motivation(1): Joint analysis of measurements from multiple platforms has the potential to enhance biomarker discovery

Neuroscience: The complexity of the human brain often necessitates the use of **multiple neuroimaging techniques** to better **understand neural activities**. Neuroimaging techniques provide **complementary** information at different scales.

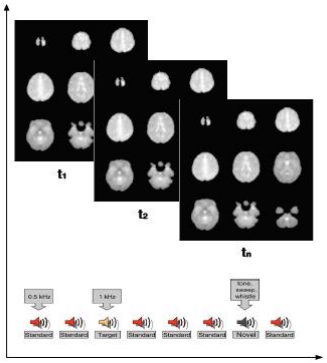
Functional Magnetic Resonance Imaging (fMRI)

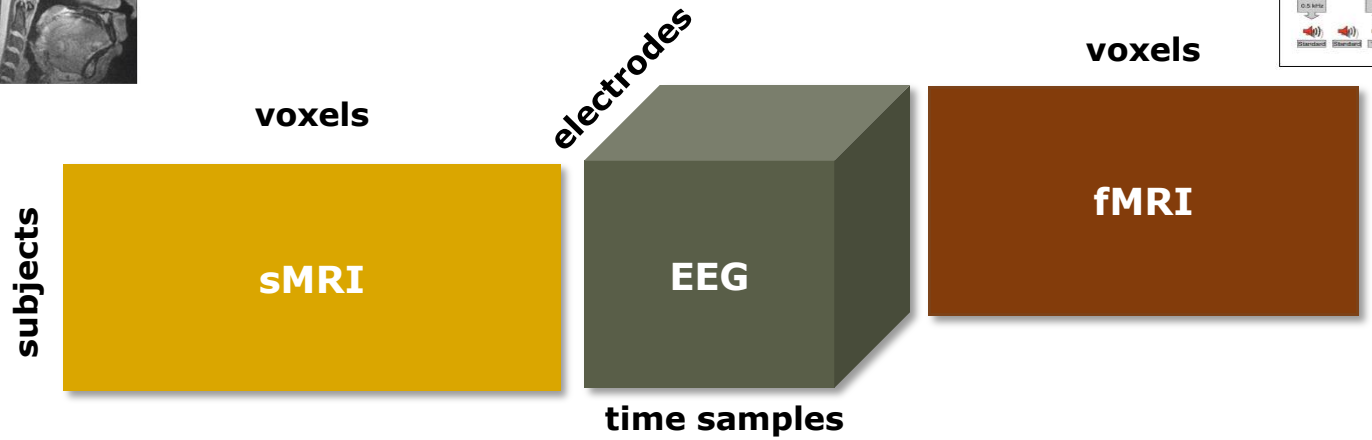
EEG (electroencephalography)





Structural MRI (sMRI)



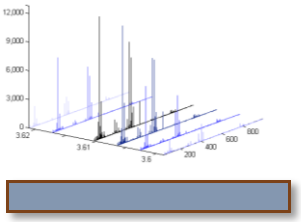


Can we jointly analyze these data sets and capture functional/structural patterns that differ between healthy controls and patients suffering from a disorder?

Motivation(1): Joint analysis of measurements from multiple platforms has the potential to enhance biomarker discovery

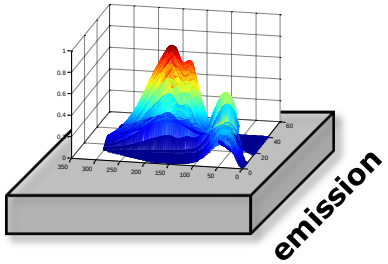
Metabolomics: The goal is to **detect** a wide range of **metabolites** in biological samples, e.g., blood or urine, and to **discover** the significant metabolic **biomarkers** related to certain conditions such as food intake or various diseases.

LC-MS (Liquid Chromatography-Mass Spectrometry)



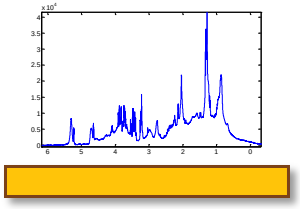
peaks

EEM (Fluorescence Spectroscopy)



excitation

NMR (Nuclear Magnetic Resonance)

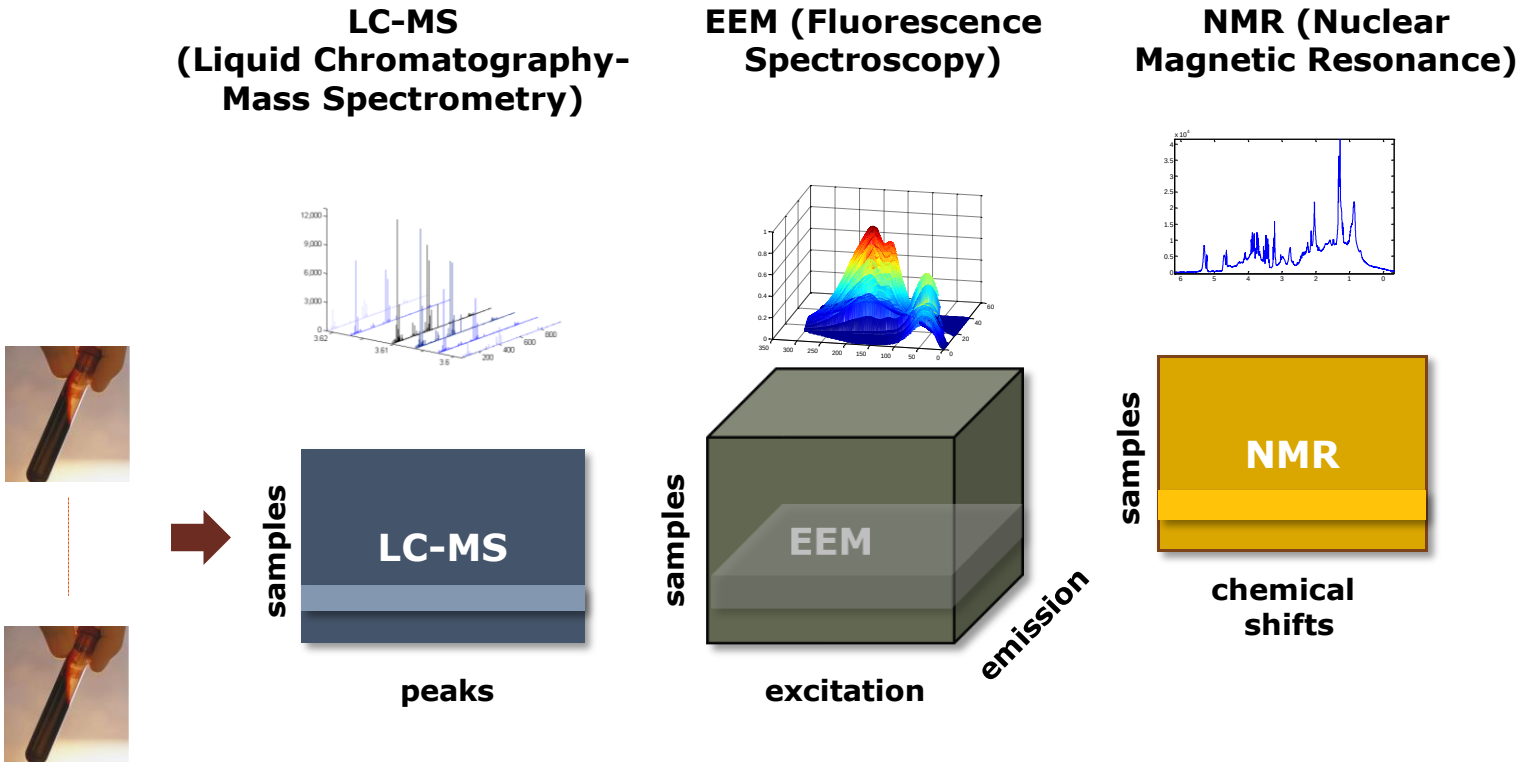


chemical shifts

simula

Motivation(1): Joint analysis of measurements from multiple platforms has the potential to enhance biomarker discovery

Metabolomics: The goal is to **detect** a wide range of **metabolites** in biological samples, e.g., blood or urine, and to **discover** the significant metabolic **biomarkers** related to certain conditions such as food intake or various diseases.

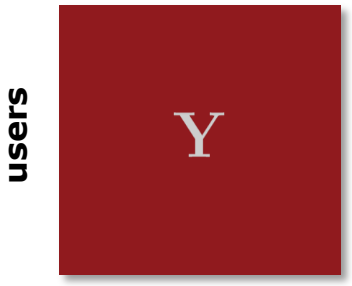


How can we jointly analyze these coupled data sets and capture the underlying patterns related to various conditions?

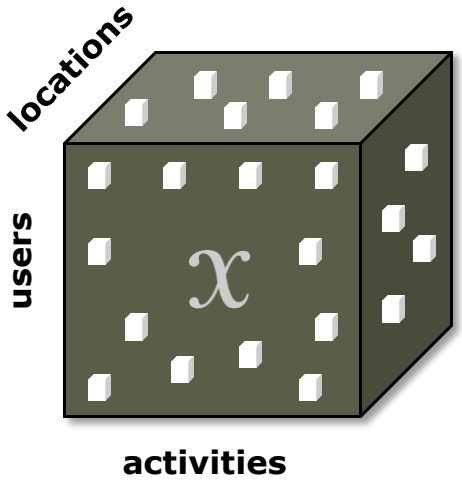
simula

Motivation(2): Data fusion can improve missing data estimation performance

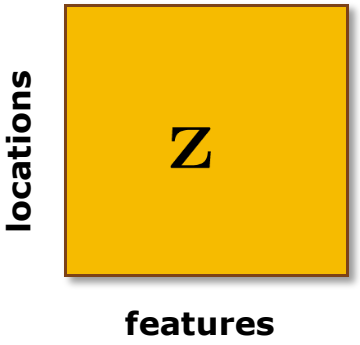
Recommender Systems: The goal is to recommend users activities that they may be interested in doing at various locations.



where they usually hang out based on GPS data



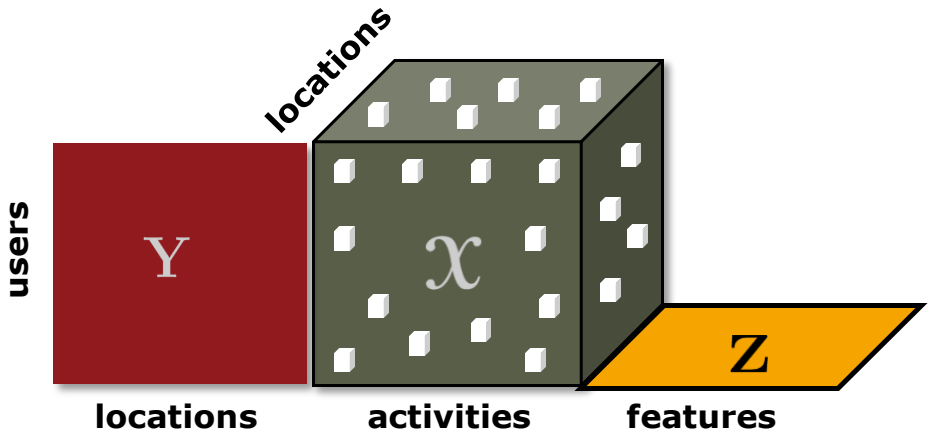
Users rarely tag their locations/activities

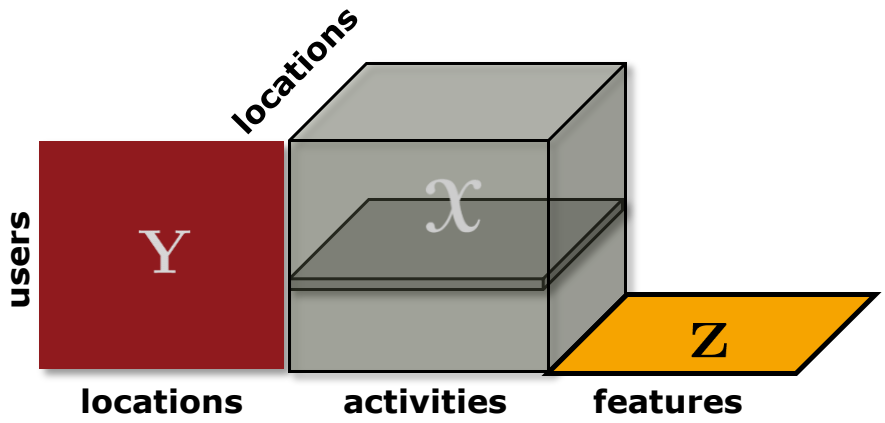


number of points of interest at different locations

Motivation(2): Data fusion can improve missing data estimation performance

Recommender Systems: The goal is to recommend users activities that they may be interested in doing at various locations.



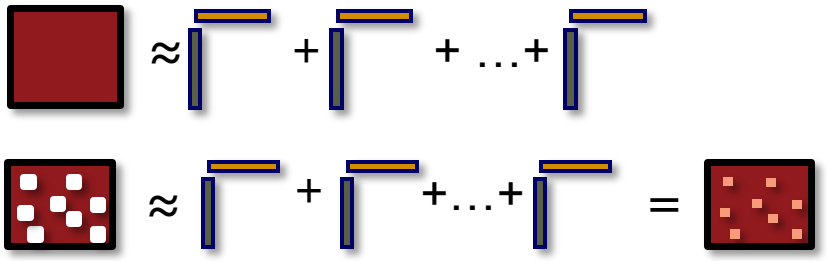


How can we fill in missing entries of the tensor using additional sources of information?

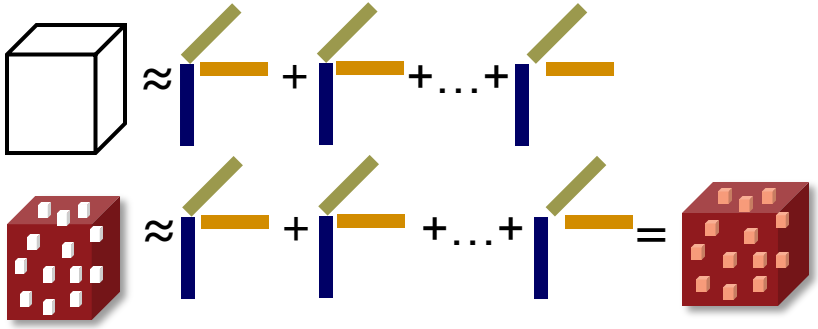
simula

Tutorial Outline

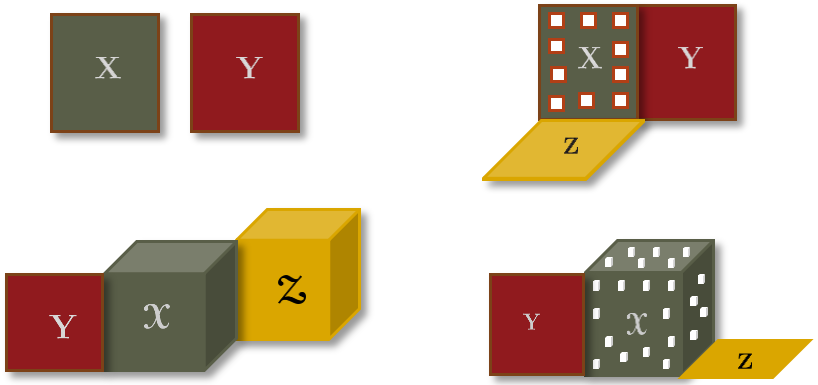
Matrix Factorizations



Tensor Factorizations

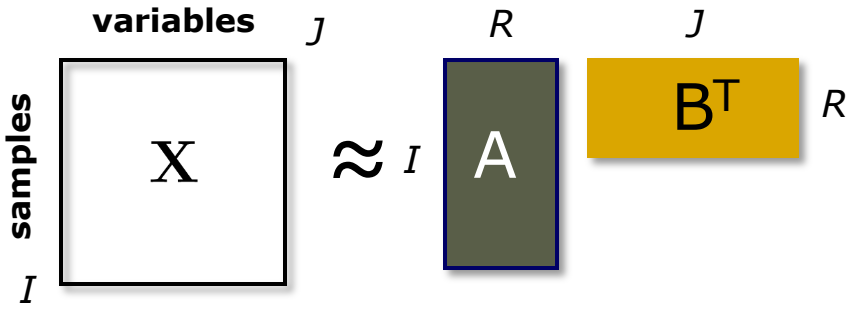


Data Fusion based on Coupled Matrix and Tensor Factorizations



Matrix Factorizations

Matrix Factorizations, e.g., SVD (Singular Value Decomposition), NMF (Nonnegative Matrix Factorization), and Sparse Matrix Factorizations, are commonly used in data mining.



$$\mathbf{X} \approx \sum_{r=1}^{R} \mathbf{a}_r \mathbf{b}_r^\mathsf{T} \approx \mathbf{A}\mathbf{B}^\mathsf{T}$$

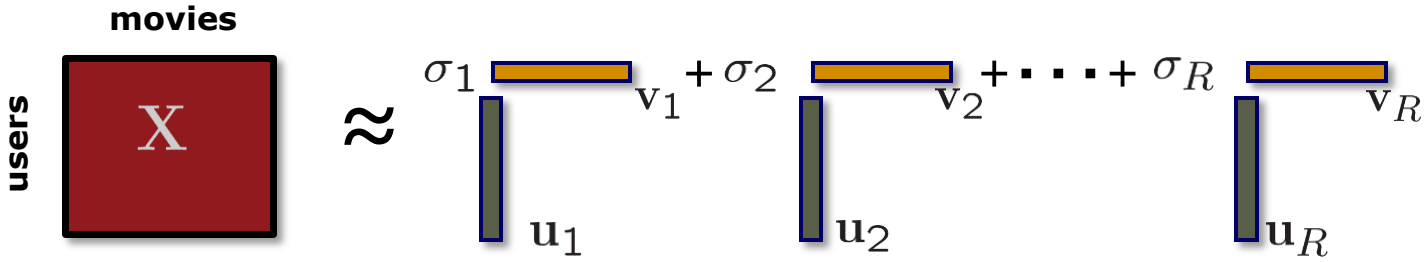
$$\mathbf{A} \in \mathbb{R}^{I \times R} = \begin{bmatrix} \mathbf{a}_1 & \cdots & \mathbf{a}_R \end{bmatrix}$$

$$\mathbf{B} \in \mathbb{R}^{J \times R} = \begin{bmatrix} \mathbf{b}_1 & \cdots & \mathbf{b}_R \end{bmatrix}$$

simula

Matrix Factorizations: Finding Underlying Structures

We can use SVD to find the underlying structures in a data set. For instance, using SVD of a *users* by *movies* matrix, we can capture the movie types and user groups.



$$\mathbf{X} \approx \sum_{r=1}^{R} \sigma_r \mathbf{u}_r \mathbf{v}_r^{\mathsf{T}}$$
$$\approx \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^{\mathsf{T}}$$

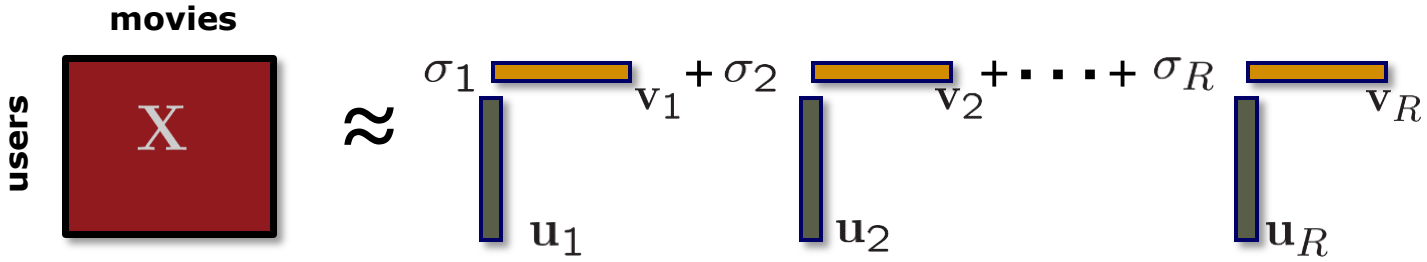
$$\min_{\mathbf{U},\mathbf{V},\sigma} \left\| \mathbf{X} - \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^{\mathsf{T}} \right\|^2$$

$$\text{s.t.} \quad \mathbf{U}^{\mathsf{T}}\mathbf{U} = \mathbf{I}, \mathbf{V}^{\mathsf{T}}\mathbf{V} = \mathbf{I}$$
$$\sigma_r \geq 0 \text{ for } r \in \{1 : R\}.$$

simula

Matrix Factorizations: Finding Underlying Structures

We can use SVD to find the underlying structures in a data set. For instance, using SVD of a *users* by *movies* matrix, we can capture the movie types and user groups.



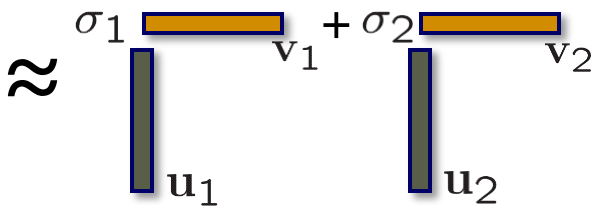
	Aloha	Star Wars	American Pie	Hunger Games	Silver Linings	Maze Runner
User 1	5	2	4	1	3	2
User 2	1	4	1	4	1	4
User 3	3	1	5	2	5	2
User 4	1	4	1	5	3	4
User 5	3	1	4	2	3	2
User 6	2	4	2	4	2	4

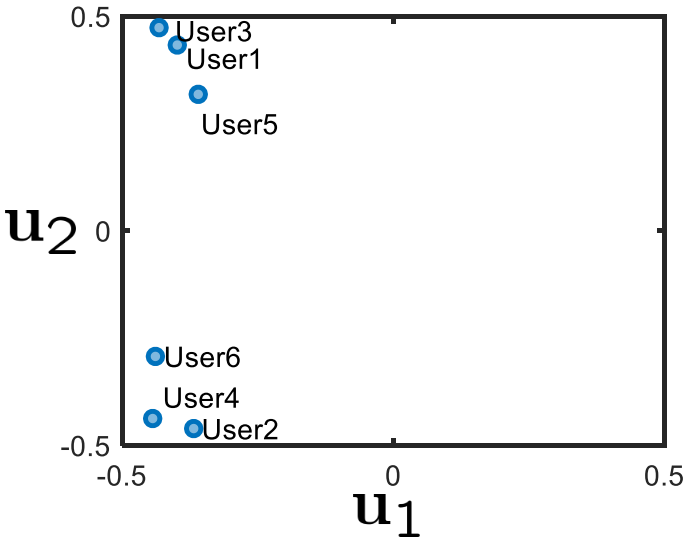
Who likes what type of movies?

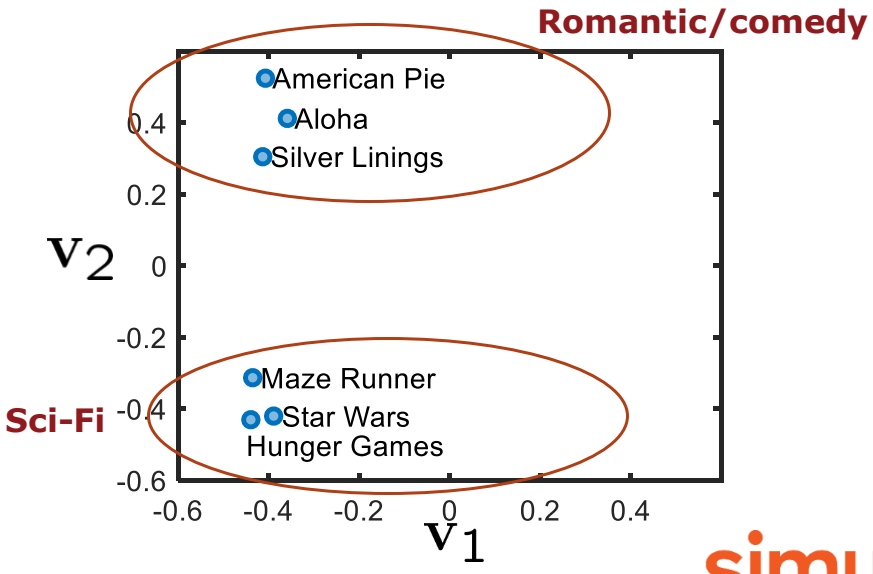
simula

Matrix Factorizations: Finding Underlying Structures

	Aloha	Star Wars	American Pie	Hunger Games	Silver Linings	Maze Runner
User 1	5	2	4	1	3	2
User 2	1	4	1	4	1	4
User 3	3	1	5	2	5	2
User 4	1	4	1	5	3	4
User 5	3	1	4	2	3	2
User 6	2	4	2	4	2	4



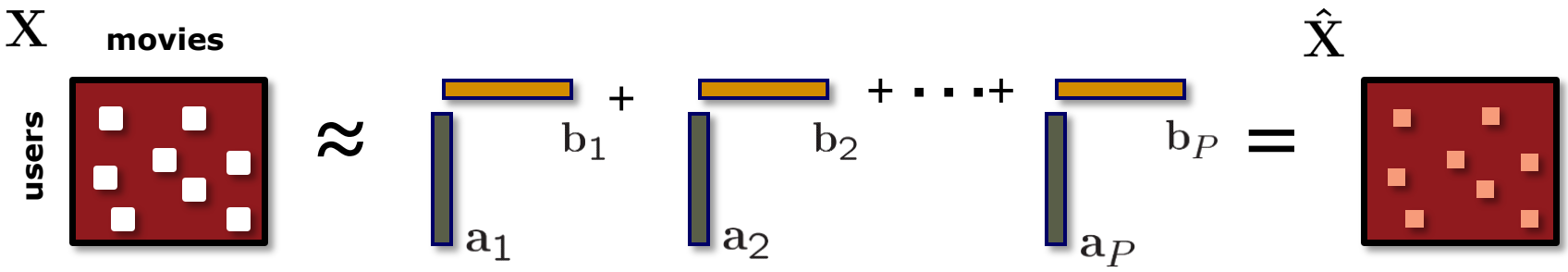




simula

Matrix Factorizations: Missing Data Estimation (a.k.a. matrix completion)

In the case of incomplete data, matrix factorizations can be used to fill in the missing entries:



Finding low-rank approximation:

$$\min_{\mathbf{A},\mathbf{B}} \left\| \mathbf{W} * (\mathbf{X} - \mathbf{A}\mathbf{B}^{\mathsf{T}}) \right\|^2$$

$$w_{ij} = \begin{cases} 1 & \text{if } x_{ij} \text{ is known,} \\ 0 & \text{if } x_{ij} \text{ is missing.} \end{cases}$$

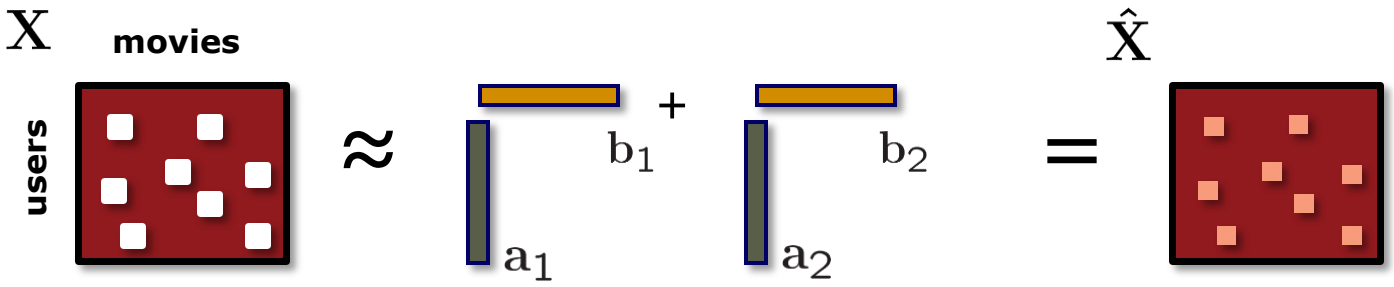
Data reconstruction:

$$\hat{\mathbf{X}} = \mathbf{A}\mathbf{B}^{\mathsf{T}}$$

It is possible to recover an unknown low-rank matrix from a nearly minimal set of entries. For more on matrix completion [Candes and Plan, 2010]

Matrix Factorizations: Missing Data Estimation

For instance, we can still factorize the incomplete **X** and then use those factors to reconstruct the matrix and estimate the missing entries.



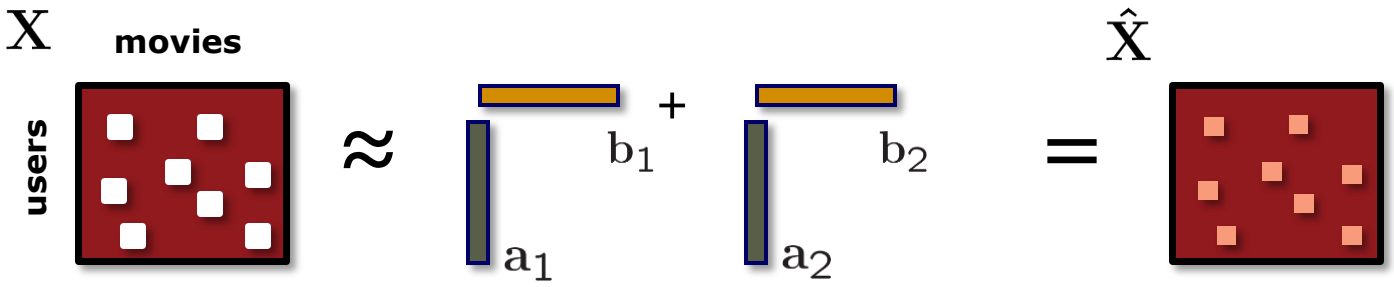
$$\hat{X} = AB^{T}$$

$$X = \begin{array}{c} \\ \text{User 1} \\ \text{User 2} \\ \text{User 3} \\ \text{User 4} \\ \text{User 5} \\ \text{User 6} \end{array} \begin{array}{c} \begin{array}{cccccc} \text{Aloha} & \text{Star Wars} & \text{American Pie} & \text{Hunger Games} & \text{Silver Linings} & \text{Maze Runner} \end{array} \\ \begin{bmatrix} 5 & 2 & 4 & 1 & 3 & ? \\ 1 & 4 & 1 & 4 & 1 & 4 \\ 3 & 1 & 5 & 2 & 5 & 2 \\ 1 & 4 & 1 & ? & 3 & 4 \\ 3 & 1 & 4 & 2 & 3 & 2 \\ 2 & ? & 2 & 4 & 2 & 4 \end{bmatrix} \end{array} \quad \begin{bmatrix} 3.72 & 1.27 & 4.41 & 1.60 & 3.78 & 1.91 \\ 0.82 & 3.91 & 0.74 & 4.05 & 1.53 & 3.91 \\ 4.06 & 1.37 & 4.82 & 1.72 & 4.12 & 2.07 \\ 1.37 & 4.06 & 1.39 & 4.24 & 2.08 & 4.15 \\ 3.15 & 1.38 & 3.71 & 1.65 & 3.25 & 1.90 \\ 1.77 & 3.76 & 1.89 & 3.97 & 2.40 & 3.94 \end{bmatrix}$$

simula

Matrix Factorizations: Missing Data Estimation

For instance, we can still factorize the incomplete **X** and then use those factors to reconstruct the matrix and estimate the missing entries.

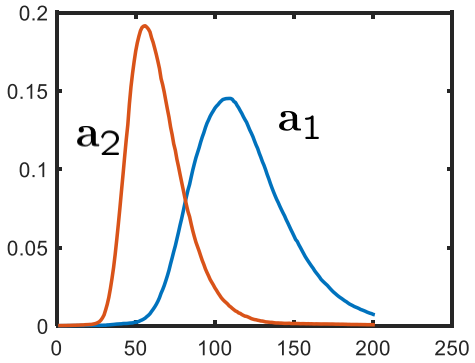


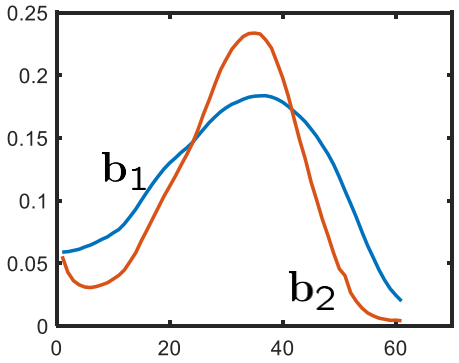
$$\hat{X} = AB^{T}$$

$$X = \begin{array}{c} \\ \text{User 1} \\ \text{User 2} \\ \text{User 3} \\ \text{User 4} \\ \text{User 5} \\ \text{User 6} \end{array} \begin{array}{c} \begin{array}{cccccc} \text{Aloha} & \text{Star Wars} & \text{American Pie} & \text{Hunger Games} & \text{Silver Linings} & \text{Maze Runner} \end{array} \\ \begin{bmatrix} 5 & 2 & 4 & 1 & 3 & 2 \\ 1 & 4 & 1 & 4 & 1 & 4 \\ 3 & 1 & 5 & 2 & 5 & 2 \\ 1 & 4 & 1 & 5 & 3 & 4 \\ 3 & 1 & 4 & 2 & 3 & 2 \\ 2 & 4 & 2 & 4 & 2 & 4 \end{bmatrix} \end{array} \quad \begin{bmatrix} 3.72 & 1.27 & 4.41 & 1.60 & 3.78 & 1.91 \\ 0.82 & 3.91 & 0.74 & 4.05 & 1.53 & 3.91 \\ 4.06 & 1.37 & 4.82 & 1.72 & 4.12 & 2.07 \\ 1.37 & 4.06 & 1.39 & 4.24 & 2.08 & 4.15 \\ 3.15 & 1.38 & 3.71 & 1.65 & 3.25 & 1.90 \\ 1.77 & 3.76 & 1.89 & 3.97 & 2.40 & 3.94 \end{bmatrix}$$

simula

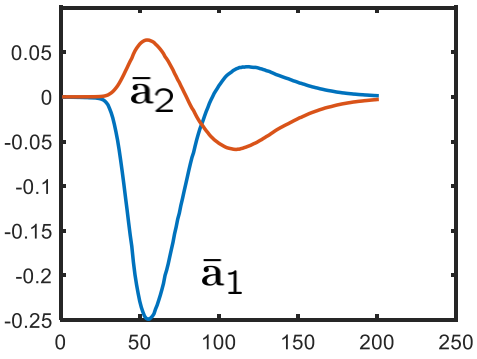
Matrix factorizations are not unique!

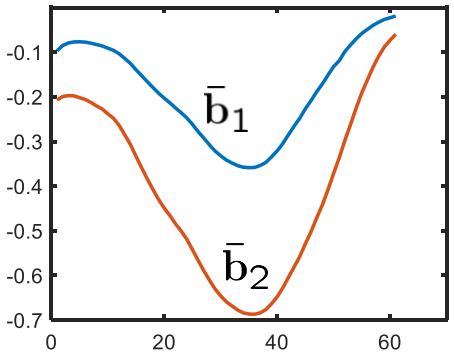




$$\mathbf{A} \in \mathbb{R}^{201\times 2} = \begin{bmatrix} \mathbf{a}_1 & \mathbf{a}_2 \end{bmatrix} \qquad \mathbf{B} \in \mathbb{R}^{61\times 2} = \begin{bmatrix} \mathbf{b}_1 & \mathbf{b}_2 \end{bmatrix}$$

$$\begin{aligned} \mathbf{X} &= \mathbf{A}\mathbf{B}^\mathsf{T} \\ &= \mathbf{A}\mathbf{M}\mathbf{M}^{-1}\mathbf{B}^\mathsf{T} \\ &= \bar{\mathbf{A}}\bar{\mathbf{B}}^\mathsf{T} \end{aligned}$$





simula

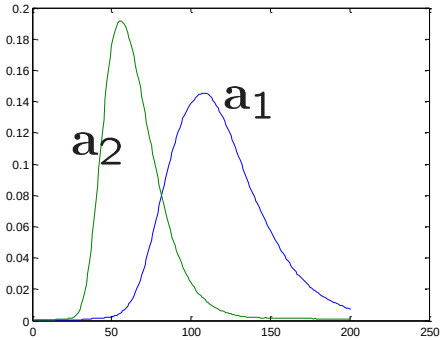
Uniqueness is an issue

$$X = AB^T = AMM^{-1}B^T = \bar{A}\bar{B}^T$$

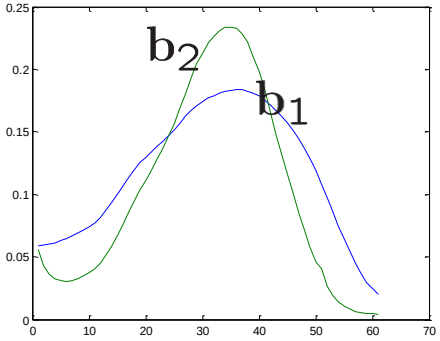
Constraints are used to deal with the uniqueness problem, e.g., SVD. However, factorizations with constraints may not be meaningful in terms of the application.

True factors

$$A \in \mathbb{R}^{201\times 2} = \begin{bmatrix} a_1 & a_2 \end{bmatrix}$$

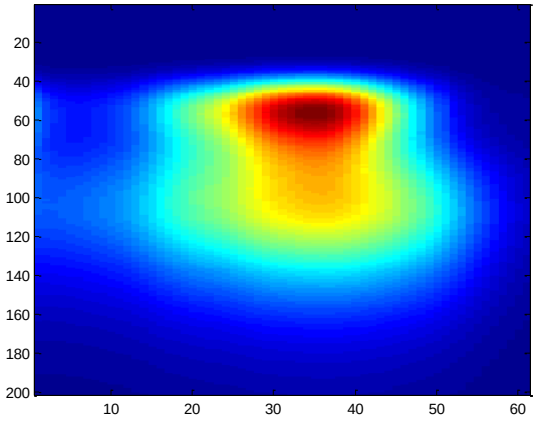


$$B \in \mathbb{R}^{61\times 2} = \begin{bmatrix} b_1 & b_2 \end{bmatrix}$$



Data matrix

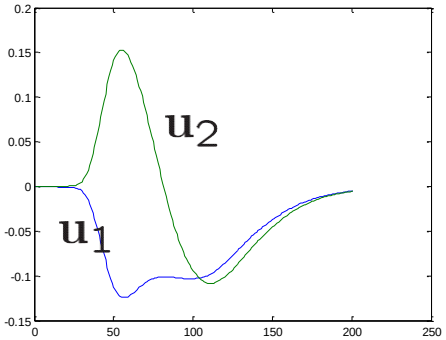
$$X = AB^T$$

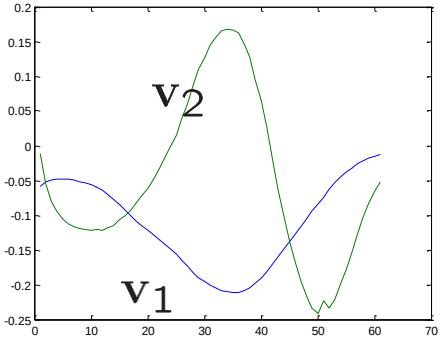


Given X, can we recover the true factors?

SVD captures…

$$X \approx \hat{X} = U\Sigma V^T$$





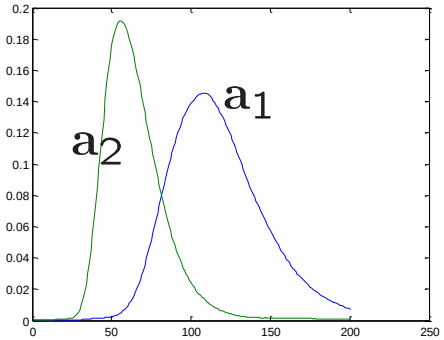
Uniqueness is an issue

$$\mathbf{X} = \mathbf{A}\mathbf{B}^{\mathsf{T}} = \mathbf{A}\mathbf{M}\mathbf{M}^{-1}\mathbf{B}^{\mathsf{T}} = \bar{\mathbf{A}}\bar{\mathbf{B}}^{\mathsf{T}}$$

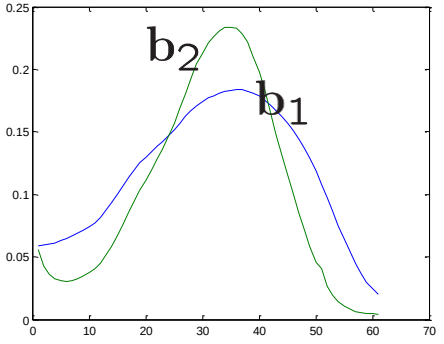
Constraints are used to deal with the uniqueness problem, e.g., SVD. However, factorizations with constraints may not be meaningful in terms of the application.

True factors

$$\mathbf{A} \in \mathbb{R}^{201\times 2} = \begin{bmatrix}\mathbf{a}_1 & \mathbf{a}_2\end{bmatrix}$$

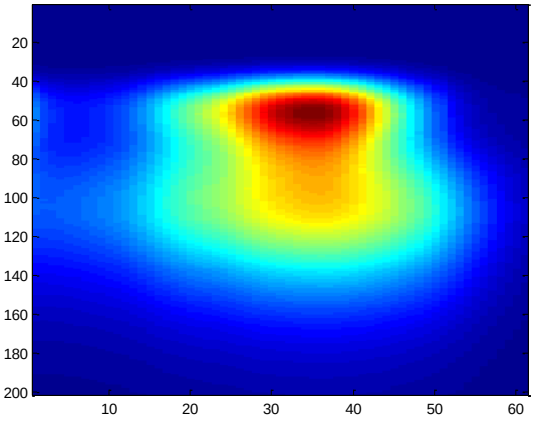


$$\mathbf{B} \in \mathbb{R}^{61\times 2} = \begin{bmatrix}\mathbf{b}_1 & \mathbf{b}_2\end{bmatrix}$$



Data matrix

$$\mathbf{X} = \mathbf{A}\mathbf{B}^{\mathsf{T}}$$

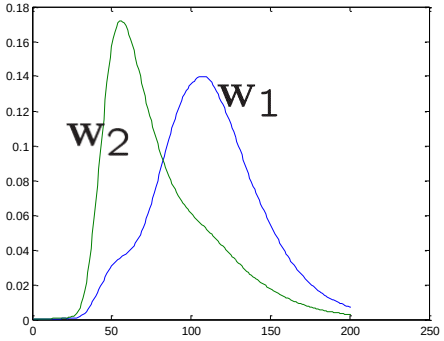


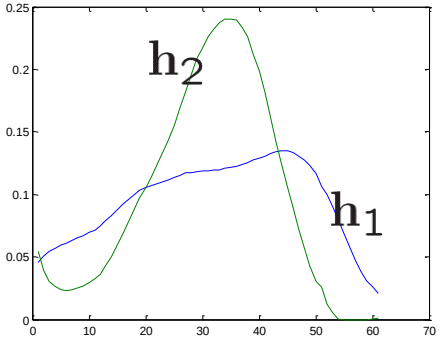
Given X, can we recover the true factors?

NMF captures…

$$\mathbf{X} \approx \hat{\mathbf{X}} = \mathbf{W}\mathbf{H}^{\mathsf{T}}$$

$$w_{ir}, h_{jr} \geq 0$$





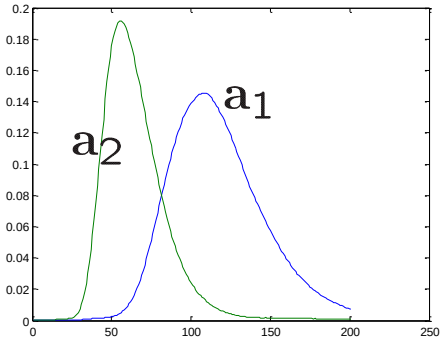
Uniqueness is an issue

$$\mathbf{X} = \mathbf{A}\mathbf{B}^\top = \mathbf{A}\mathbf{M}\mathbf{M}^{-1}\mathbf{B}^\top = \bar{\mathbf{A}}\bar{\mathbf{B}}^\top$$

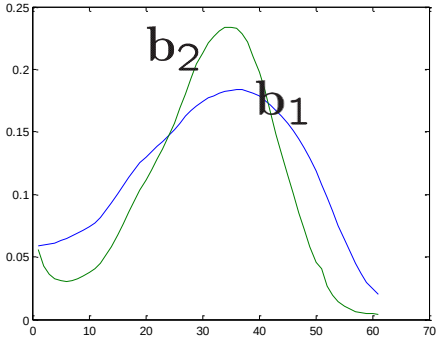
Constraints are used to deal with the uniqueness problem, e.g., SVD. However, factorizations with constraints may not be meaningful in terms of the application.

True factors

$$\mathbf{A} \in \mathbb{R}^{201\times 2} = \begin{bmatrix}\mathbf{a}_1 & \mathbf{a}_2\end{bmatrix}$$

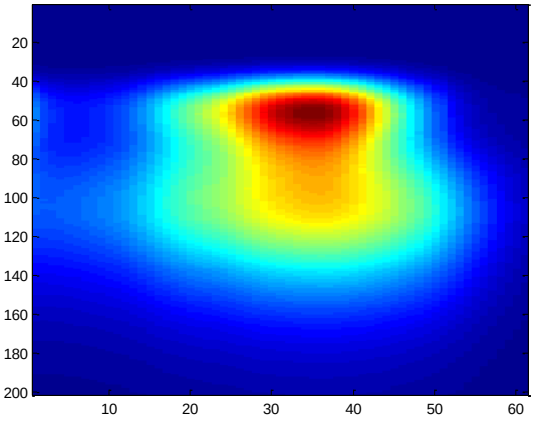


$$\mathbf{B} \in \mathbb{R}^{61\times 2} = \begin{bmatrix}\mathbf{b}_1 & \mathbf{b}_2\end{bmatrix}$$



Data matrix

$$\mathbf{X} = \mathbf{A}\mathbf{B}^\top$$

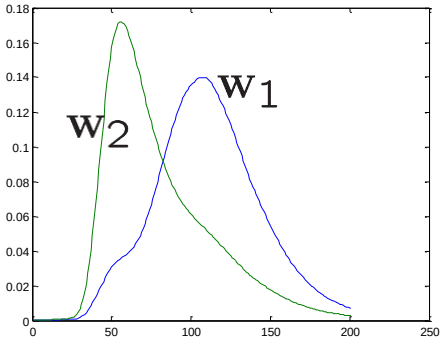


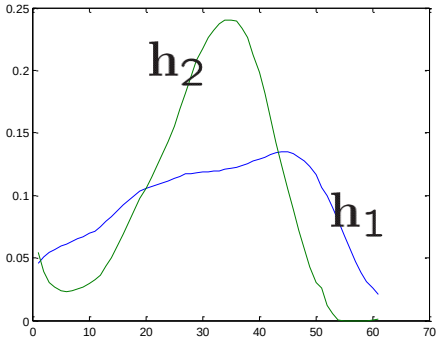
Given X, can we recover the true factors?

NMF captures…

$$\mathbf{X} \approx \hat{\mathbf{X}} = \underbrace{\mathbf{W}\mathbf{M}}_{\bar{\mathbf{W}}}\underbrace{\mathbf{M}^{-1}\mathbf{H}^\top}_{\bar{\mathbf{H}}^\top}$$

$$\bar{w}_{ir}, \bar{h}_{jr} \geq 0$$

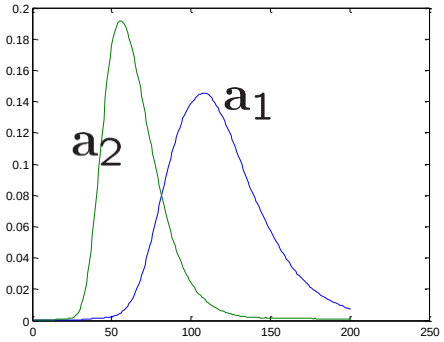




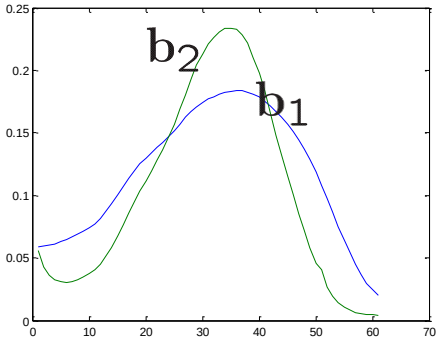
What if we have multiple matrices with the same underlying factors but in different proportions…

True factors

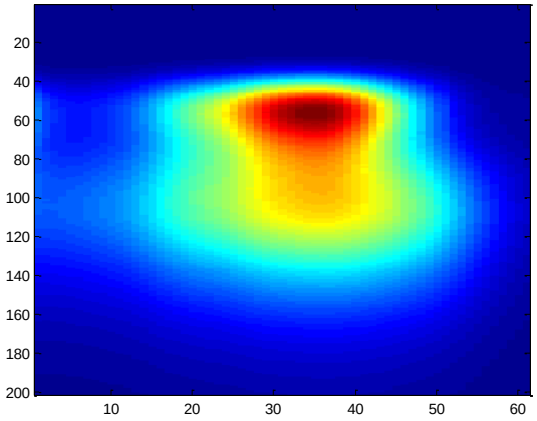
$$\mathbf{A} \in \mathbb{R}^{201\times 2} = \begin{bmatrix} \mathbf{a}_1 & \mathbf{a}_2 \end{bmatrix}$$



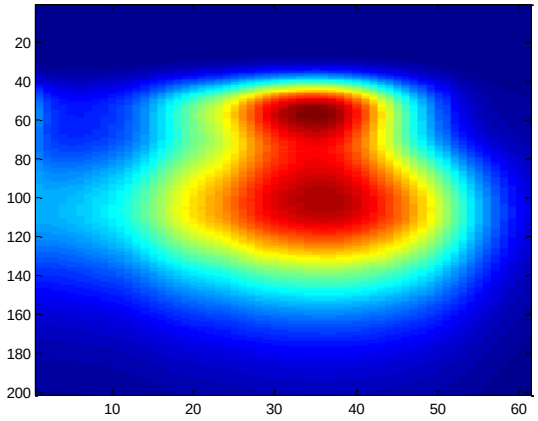
$$\mathbf{B} \in \mathbb{R}^{61\times 2} = \begin{bmatrix} \mathbf{b}_1 & \mathbf{b}_2 \end{bmatrix}$$



$$\mathbf{X}_1 = \mathbf{A}\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\mathbf{B}^\mathsf{T}$$



$$\mathbf{X}_2 = \mathbf{A}\begin{bmatrix} 3 & 0 \\ 0 & 2 \end{bmatrix}\mathbf{B}^\mathsf{T}$$

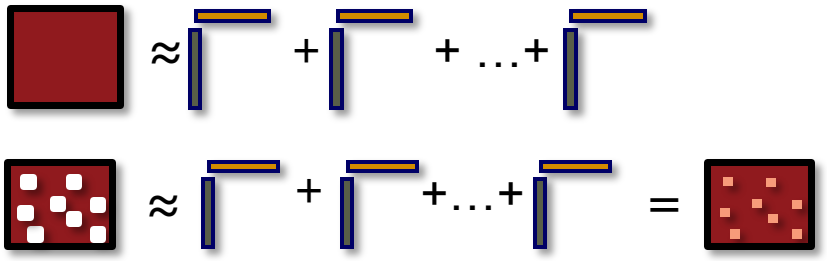


We can recover the true factors uniquely up to trivial indeterminacies, i.e., scaling and permutation.

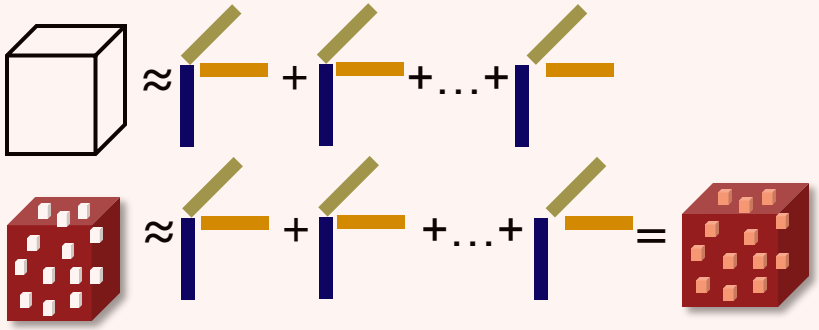
simula

Tutorial Outline

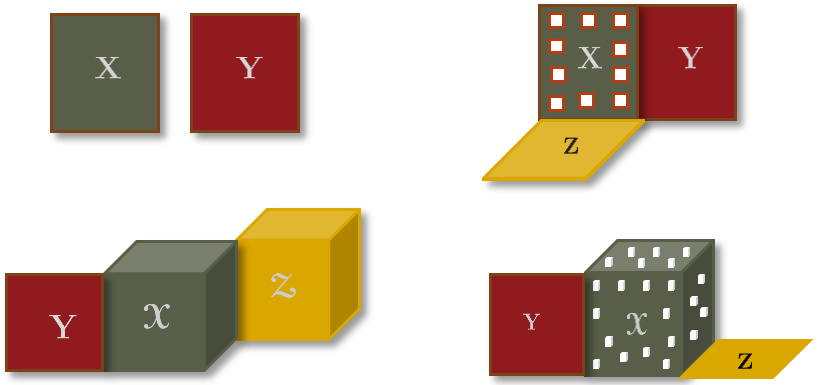
Matrix Factorizations



Tensor Factorizations



Data Fusion based on Coupled Matrix and Tensor Factorizations

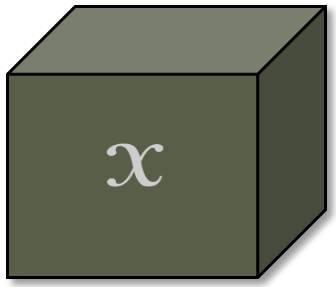


Tensors: Terminology

Order: Number of Modes or Ways

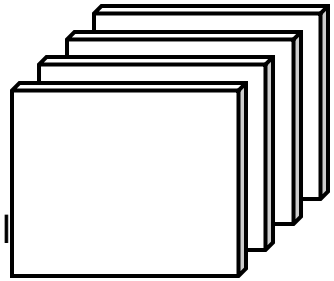
first-order tensor (vector)

second-order tensor (matrix)

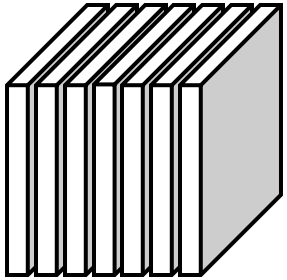


third-order tensor (three-way array)

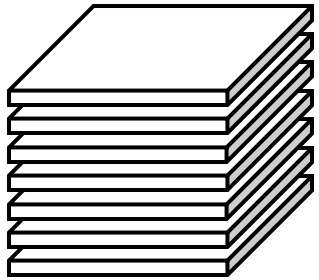
higher-order tensors (multi-way arrays)



Frontal slices

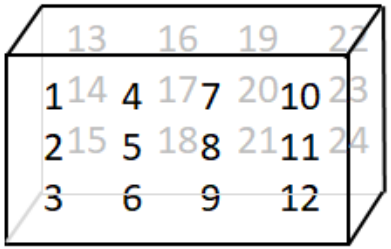


Vertical slices



Horizontal slices

$$\mathcal{X} =$$



Order: 3

Size: $3 \times 4 \times 2$

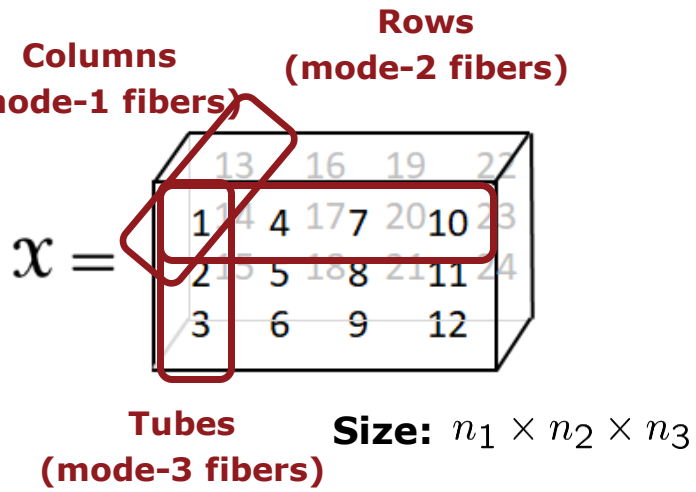
$$\mathbf{X}(:,:,1) = \begin{bmatrix} 1 & 4 & 7 & 10 \\ 2 & 5 & 8 & 11 \\ 3 & 6 & 9 & 12 \end{bmatrix}$$

Frontal slices

$$\mathbf{X}(:,:,2) = \begin{bmatrix} 13 & 16 & 19 & 22 \\ 14 & 17 & 20 & 23 \\ 15 & 18 & 21 & 24 \end{bmatrix}$$

Tensors: Basic Operations

Matricization (a.k.a. Unfolding or Flattening)



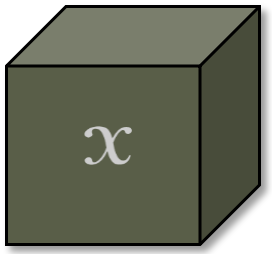
Size: $n_1 \times n_2 \times n_3$

$$\underset{n_1 \times n_2 n_3}{\mathbf{X}_{(1)}} = \begin{bmatrix} 1 & 4 & 7 & 10 & 13 & \dots & 22 \\ 2 & 5 & 8 & 11 & 14 & \dots & 23 \\ 3 & 6 & 9 & 12 & 15 & \dots & 24 \end{bmatrix}$$

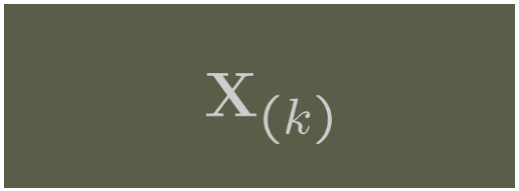
$$\underset{n_2 \times n_1 n_3}{\mathbf{X}_{(2)}} = \begin{bmatrix} 1 & 2 & 3 & 13 & \dots & 15 \\ 4 & 5 & 6 & 16 & \dots & 18 \\ 7 & 8 & 9 & 19 & \dots & 21 \\ 10 & 11 & 12 & 22 & \dots & 24 \end{bmatrix}$$

$$\underset{n_3 \times n_1 n_2}{\mathbf{X}_{(3)}} = \begin{bmatrix} 1 & 2 & 3 & 4 & \dots & 12 \\ 13 & 14 & 15 & 16 & \dots & 24 \end{bmatrix}$$

Unfolding a *d*-way array (mode-*k* unfolding)



Size: $n_1 \times n_2 \times \dots \times n_d$



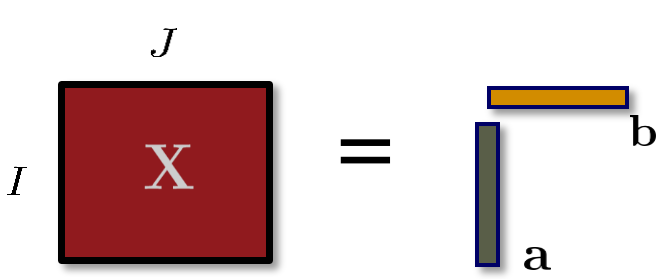
$$n_k \times \prod_{i, i \neq k}^{d} n_i$$

simula

Tensors: Basic Operations (cont.)

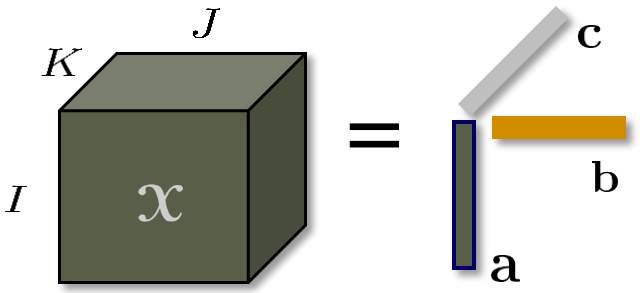
Vector outer product $\mathbf{a} \in \mathbb{R}^I, \mathbf{b} \in \mathbb{R}^J, \mathbf{c} \in \mathbb{R}^K$

$$\mathbf{X} = \mathbf{a}\mathbf{b}^\mathsf{T} = \mathbf{a} \circ \mathbf{b}$$



$$\mathbf{X} \in \mathbb{R}^{I \times J}, x_{ij} = a_i b_j$$

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



$$\mathcal{X} \in \mathbb{R}^{I \times J \times K}, x_{ijk} = a_i b_j c_k$$

Kronecker product

$\mathbf{a} \in \mathbb{R}^I, \mathbf{b} \in \mathbb{R}^J, \mathbf{A} \in \mathbb{R}^{I \times M}, \mathbf{B} \in \mathbb{R}^{J \times N}$

$$\mathbf{a} \otimes \mathbf{b} = \begin{bmatrix} a_1\mathbf{b} \\ a_2\mathbf{b} \\ \vdots \\ a_I\mathbf{b} \end{bmatrix} \in \mathbb{R}^{IJ}$$

$$\mathbf{A} \otimes \mathbf{B} = \begin{bmatrix} a_{11}\mathbf{B} & \dots & a_{1M}\mathbf{B} \\ a_{21}\mathbf{B} & \dots & a_{2M}\mathbf{B} \\ \vdots & & \vdots \\ a_{I1}\mathbf{B} & \dots & a_{IM}\mathbf{B} \end{bmatrix} \in \mathbb{R}^{IJ \times MN}$$

Khatri-Rao product (Columnwise Kronecker Product)

$\mathbf{A} \in \mathbb{R}^{I \times R}, \mathbf{B} \in \mathbb{R}^{J \times R}$

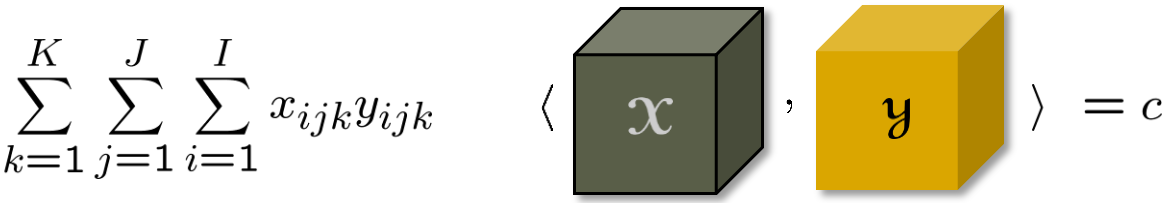
$$\mathbf{A} \odot \mathbf{B} = \begin{bmatrix} \mathbf{a}_1 \otimes \mathbf{b}_1 & \mathbf{a}_2 \otimes \mathbf{b}_2 & \dots & \mathbf{a}_R \otimes \mathbf{b}_R \end{bmatrix} \in \mathbb{R}^{IJ \times R}$$

simula

Tensors: Basic Operations (cont.)

Tensor Inner Product $\mathcal{X} \in \mathbb{R}^{I \times J \times K}, \mathcal{Y} \in \mathbb{R}^{I \times J \times K}$

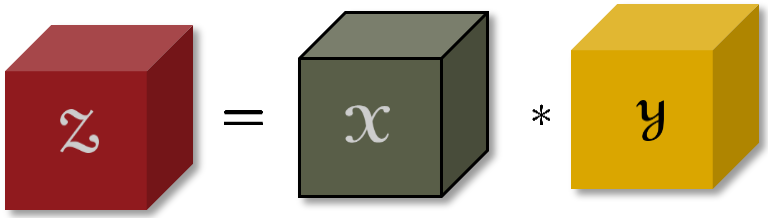
$$\langle \mathcal{X}, \mathcal{Y} \rangle = \sum_{k=1}^{K} \sum_{j=1}^{J} \sum_{i=1}^{I} x_{ijk} y_{ijk}$$



Frobenius Norm

$$\| \mathcal{X} \|^2 = \langle \mathcal{X}, \mathcal{X} \rangle = \sum_{k=1}^{K} \sum_{j=1}^{J} \sum_{i=1}^{I} x_{ijk}^2$$

Hadamard Product $\mathcal{X}, \mathcal{Y}, \mathcal{Z} \in \mathbb{R}^{I \times J \times K}$



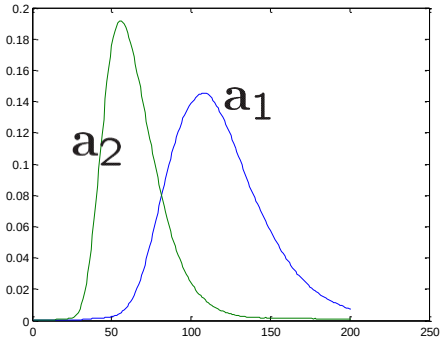
$$z_{ijk} = x_{ijk} y_{ijk}$$

simula

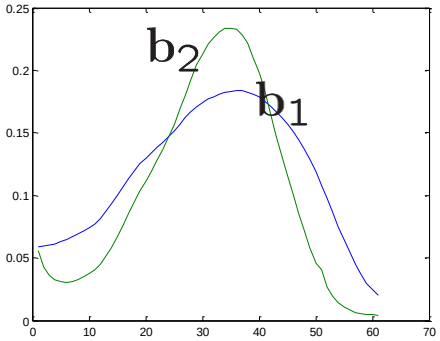
What if we have multiple matrices with the same underlying factors but in different proportions…

True factors

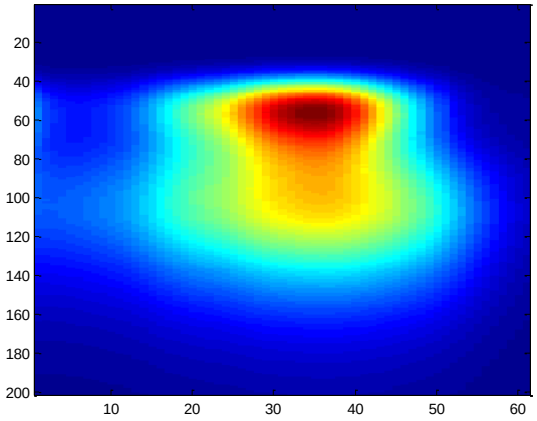
$$\mathbf{A} \in \mathbb{R}^{201\times 2} = \begin{bmatrix} \mathbf{a}_1 & \mathbf{a}_2 \end{bmatrix}$$



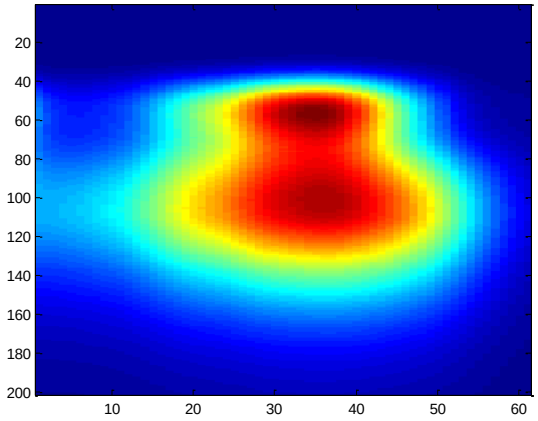
$$\mathbf{B} \in \mathbb{R}^{61\times 2} = \begin{bmatrix} \mathbf{b}_1 & \mathbf{b}_2 \end{bmatrix}$$



$$\mathbf{X}_1 = \mathbf{A}\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\mathbf{B}^\mathsf{T}$$



$$\mathbf{X}_2 = \mathbf{A}\begin{bmatrix} 3 & 0 \\ 0 & 2 \end{bmatrix}\mathbf{B}^\mathsf{T}$$



We can recover the true factors uniquely up to trivial indeterminacies, i.e., scaling and permutation.

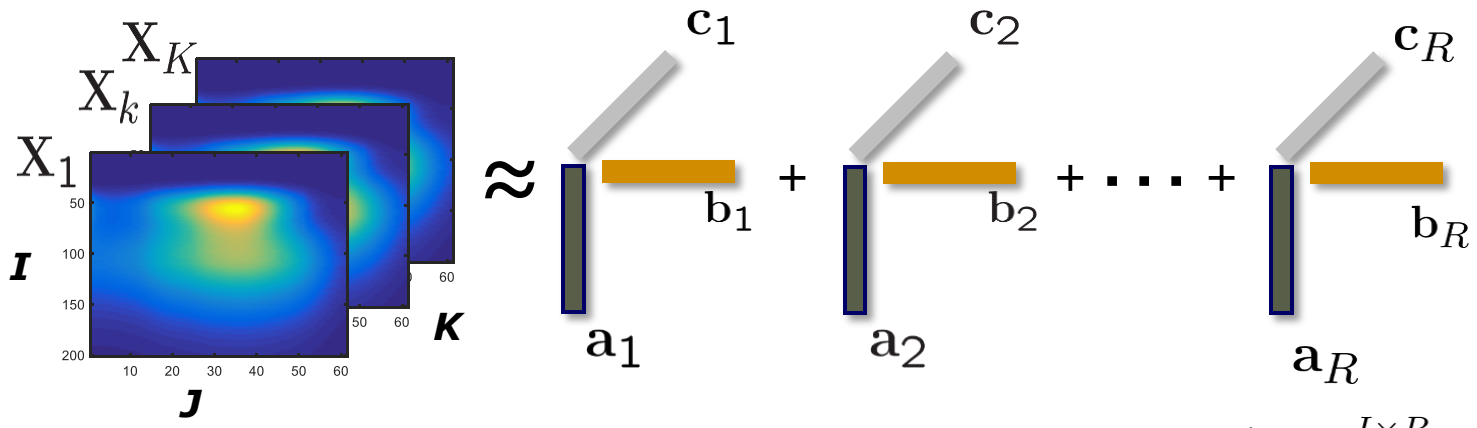
simula

A popular tensor factorization model: CP

Canonical Polyadic (CP)
CANDECOMP/PARAFAC (CP)

Hitchcock, 1927: Polyadic form of a tensor
Harshman, 1970: Parallel Factor Analysis (PARAFAC)
Carroll & Chang, 1970: Canonical Decomposition (CANDECOMP)

As an extension of matrix factorizations to higher-order tensors (multi-way arrays), tensor factorizations are used to extract the underlying factors in higher-order data sets. The CP model represents a tensor as a sum of rank-one tensors:



$$\mathbf{X}_k \approx \mathbf{A} \begin{bmatrix} c_{k1} & & \\ & \cdots & \\ & & c_{kR} \end{bmatrix} \mathbf{B}^\top$$

$$\mathcal{X} \approx \sum_{r=1}^{R} \mathbf{a}_r \circ \mathbf{b}_r \circ \mathbf{c}_r \approx [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$$

$$\mathbf{A} \in \mathbb{R}^{I \times R} = \begin{bmatrix} \mathbf{a}_1 & \cdots & \mathbf{a}_R \end{bmatrix}$$

$$\mathbf{B} \in \mathbb{R}^{J \times R} = \begin{bmatrix} \mathbf{b}_1 & \cdots & \mathbf{b}_R \end{bmatrix}$$

$$\mathbf{C} \in \mathbb{R}^{K \times R} = \begin{bmatrix} \mathbf{c}_1 & \cdots & \mathbf{c}_R \end{bmatrix}$$

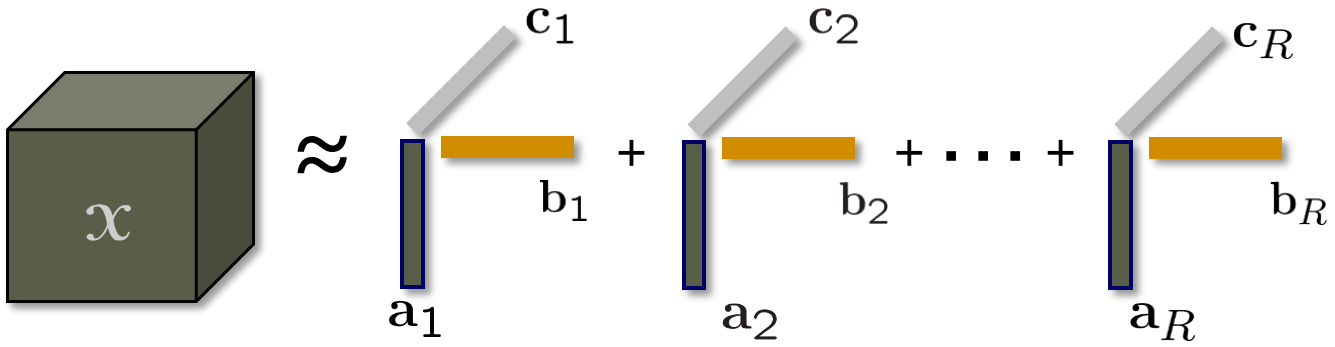
CP is unique up to scaling and permutation under the condition that

$$\text{krank}(\mathbf{A}) + \text{krank}(\mathbf{B}) + \text{krank}(\mathbf{C}) \geq 2R + 2$$

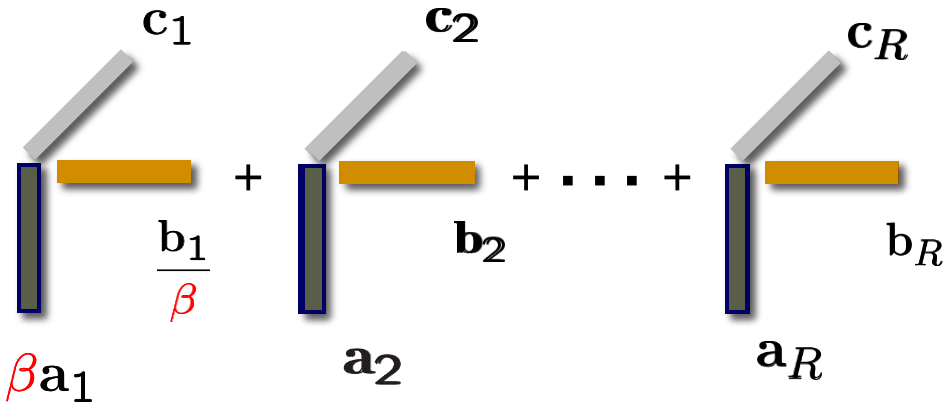
[Kruskal, 1977; Sidiropoulos and Bro, 2000]

where krank (A) = max. value of *k* such that any *k* columns of A is linearly independent.

CP is unique up to trivial indeterminacies



scaling



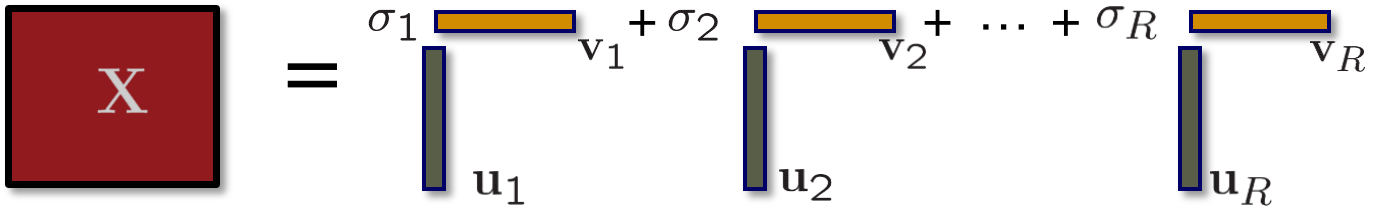
permutation

+ + · · · +

simula

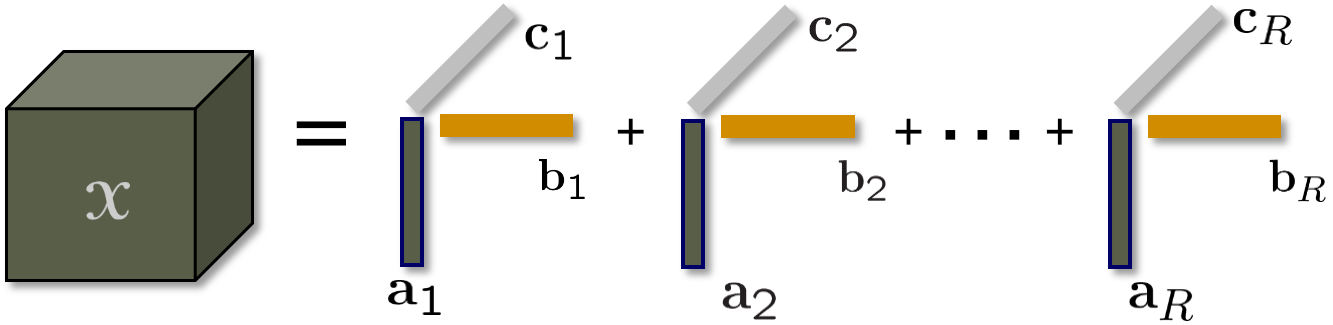
SVD vs. CP

SVD expresses a matrix as the sum of rank-one matrices.



$$\mathbf{X} = \sum_{r=1}^{R} \sigma_r \, \mathbf{u}_r \circ \mathbf{v}_r$$

CP expresses a higher-order tensor as the sum of rank-one tensors.



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

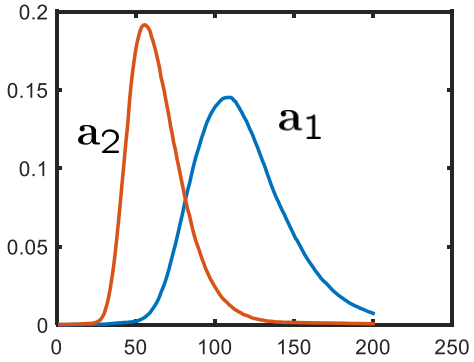
CP is an extension of SVD to higher-order tensors but it is different in many ways.

Does it really work?

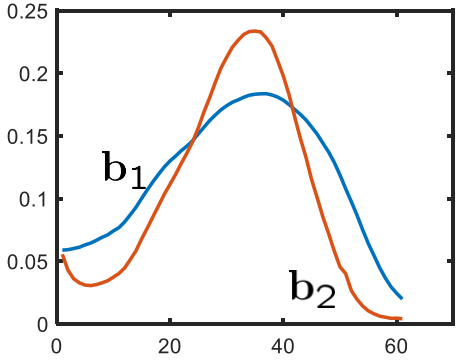
EXAMPLE:

True factors

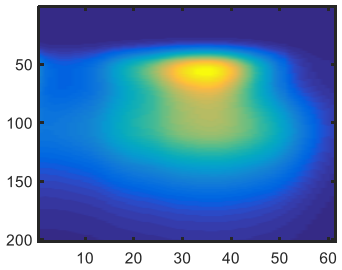
$$\mathbf{A} \in \mathbb{R}^{201\times 2} = \begin{bmatrix} \mathbf{a}_1 & \mathbf{a}_2 \end{bmatrix}$$

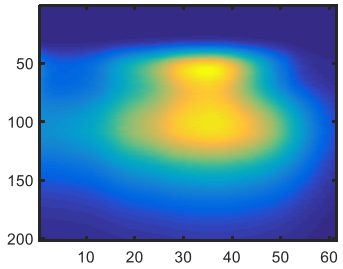


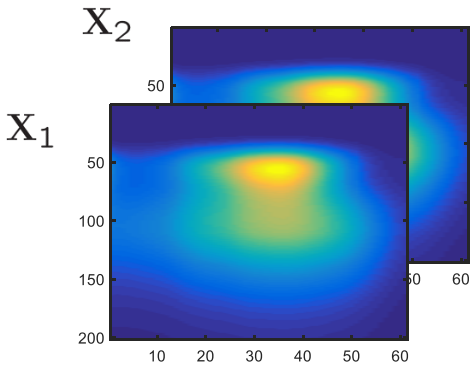
$$\mathbf{B} \in \mathbb{R}^{61\times 2} = \begin{bmatrix} \mathbf{b}_1 & \mathbf{b}_2 \end{bmatrix}$$



$$\mathbf{X}_1 = \mathbf{A}\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\mathbf{B}^\mathsf{T} \qquad \mathbf{X}_2 = \mathbf{A}\begin{bmatrix} 3 & 0 \\ 0 & 2 \end{bmatrix}\mathbf{B}^\mathsf{T}$$



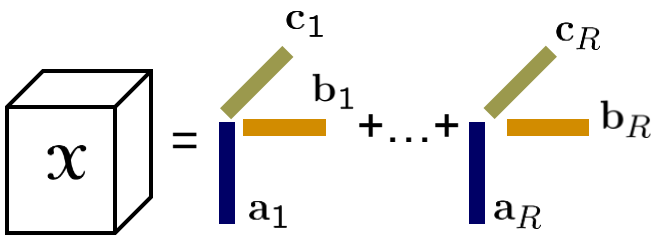




$$\mathcal{X} \approx [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$$

```
load toydata.mat
XX(:,:,1) =A*[1 0; 0 1]*B';
XX(:,:,2) =A*[3 0; 0 2]*B';
%set optimization parameters
options              = ncg('defaults');
options.DisplayIters = 50;
options.RelFuncTol   = 1e-8;
options.StopTol      = 1e-8;
% call cp_opt to fit the CP model
[Fac,FacInit,out]  = cp_opt(tensor(XX), 2,'init','nvecs','alg_options',options);
subplot(2,1,1);plot(Fac.U{1});
subplot(2,1,2);plot(Fac.U{2});
```

Algorithms for fitting the CP model



The goal is to find matrices **A**, **B**, **C** that solve the following optimization problem:

$$\min_{\mathbf{A},\mathbf{B},\mathbf{C}} \| \mathcal{X} - [\![\mathbf{A},\mathbf{B},\mathbf{C}]\!] \|^2$$

Traditional Approach:
Alternating Least Squares (ALS)
[Harshman, 1970; Carroll & Chang, 1970]

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

Matricized Tensor Times Khatri-Rao Product (MTTKRP)

while "not converged" **do**

Solve for **A** (with fixed **B**, **C**)

$$\min_{\mathbf{A}} \| \mathcal{X} - [\![\mathbf{A},\mathbf{B},\mathbf{C}]\!] \|^2$$

Solve for **B** (with fixed **A** and **C**)

$$\min_{\mathbf{B}} \| \mathcal{X} - [\![\mathbf{A},\mathbf{B},\mathbf{C}]\!] \|^2$$

Solve for **C** (with fixed **A** and **B**)

$$\min_{\mathbf{C}} \| \mathcal{X} - [\![\mathbf{A},\mathbf{B},\mathbf{C}]\!] \|^2$$

end while

However, better convergence properties and accuracy have been achieved using all-at-once optimization approaches:

First-order methods: [Paatero, 1999], CP-OPT [Acar, Dunlavy, Kolda, 2011]

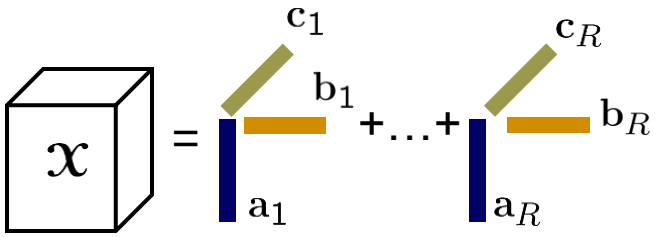
Second-order methods: [Paatero, 1997; Tomasi and Bro, 2006; Phan et al., 2013; Sorber et al., 2013]

CP-OPT: Gradient-based All-at-once Approach

[Acar, Dunlavy, and Kolda, 2011]

CP-OPT is a general gradient–based optimization approach for fitting a CP model.

$$\min_{\mathbf{A},\mathbf{B},\mathbf{C}} \| \mathcal{X} - [\![\mathbf{A},\mathbf{B},\mathbf{C}]\!] \|^2$$



Step1: Define the objective function

$$f(\mathbf{A},\mathbf{B},\mathbf{C}) = \frac{1}{2} \| \mathcal{X} - [\![\mathbf{A},\mathbf{B},\mathbf{C}]\!] \|^2$$

Step2: Compute the gradient ***Remember the ALS steps!***

$$\frac{\partial f}{\partial \mathbf{A}} = -\mathbf{X}_{(1)}(\mathbf{C} \odot \mathbf{B}) + \mathbf{A}(\mathbf{C}^\mathsf{T}\mathbf{C} * \mathbf{B}^\mathsf{T}\mathbf{B})$$

$$\frac{\partial f}{\partial \mathbf{B}} = -\mathbf{X}_{(2)}(\mathbf{C} \odot \mathbf{A}) + \mathbf{B}(\mathbf{C}^\mathsf{T}\mathbf{C} * \mathbf{A}^\mathsf{T}\mathbf{A})$$

$$\frac{\partial f}{\partial \mathbf{C}} = -\mathbf{X}_{(3)}(\mathbf{B} \odot \mathbf{A}) + \mathbf{C}(\mathbf{B}^\mathsf{T}\mathbf{B} * \mathbf{A}^\mathsf{T}\mathbf{A})$$

Vectorize and concatenate the partials

$$\nabla f = \begin{bmatrix} \frac{\partial f}{\partial \mathbf{a}_1} \\ \vdots \\ \frac{\partial f}{\partial \mathbf{a}_R} \\ \frac{\partial f}{\partial \mathbf{b}_1} \\ \vdots \\ \frac{\partial f}{\partial \mathbf{b}_R} \\ \frac{\partial f}{\partial \mathbf{c}_1} \\ \vdots \\ \frac{\partial f}{\partial \mathbf{c}_R} \end{bmatrix}$$

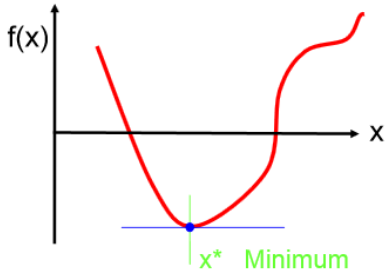
Step3: Pick a first-order optimization method

e.g., Nonlinear Conjugate Gradient (NCG) from the **Poblano Toolbox** [Dunlavy, Kolda and Acar, 2010]

Few words on optimization

Optimization problem

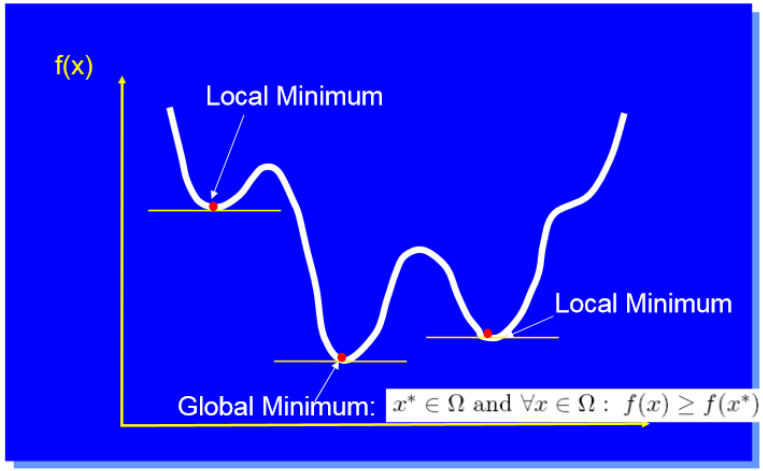
$$\min_x f(x)$$



Figures from Moritz Diehl's slides

Aim of most optimization algorithms

Find a local minimizer x*



Gradient

$$\nabla f(\mathbf{x}) = \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \vdots \\ \frac{\partial f}{\partial x_n} \end{bmatrix}$$

Hessian

$$\nabla^2 f = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1 \partial x_1} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2 \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_2 \partial x_n} \\ \vdots & \ddots & \vdots & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \frac{\partial^2 f}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_n \partial x_n} \end{bmatrix}$$

First-order Necessary Conditions

If x* is a local minimizer and *f* is continuously differentiable in an open neighborhood of x*, then $\nabla f(\mathbf{x}^*) = 0$.

Second-order Necessary Conditions

If x* is a local minimizer of *f* and $\nabla^2 f$ exists and is continuous in an open neighborhood of x*, then $\nabla f(\mathbf{x}^*) = 0$ and $\nabla^2 f(x^*)$ is positive semidefinite.

Few words on optimization (cont.)

The general structure of the optimization algorithms in the Poblano Toolbox:

choose a starting value x_0
while not converged **do**
  find the search direction p_k
  determine the step length α
  new iterate $x_k = x_{k-1} + \alpha p_k$
end while

Different choices of search direction and step length lead to different optimization algorithms.

Taylor's approximation:

Suppose that *f* is twice continuously differentiable, then,

$$f(x_k + p) \approx f_k + p^{\mathsf{T}} \nabla f_k + \frac{1}{2} p^{\mathsf{T}} \nabla^2 f_k p$$

Steepest descent: $p_k = -\nabla f_k$

Newton's direction: $p_k = -(\nabla^2 f_k)^{-1} \nabla f_k$

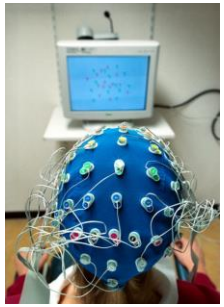
Quasi-Newton Methods (e.g., Limited-Memory BFGS): $p_k = -B_k^{-1} \nabla f_k$

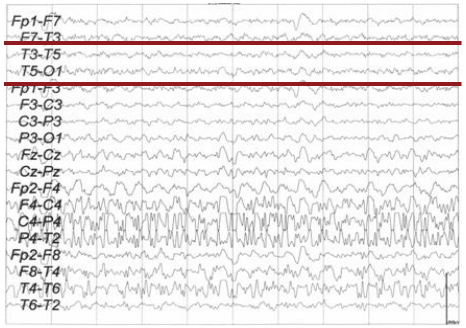
Nonlinear Conjugate Gradient Methods: $p_k = -\nabla f_k + \beta_k p_{k-1}$

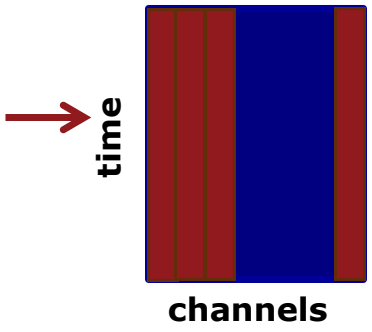
The ones using only the first-order info are implemented in the Poblano Toolbox.

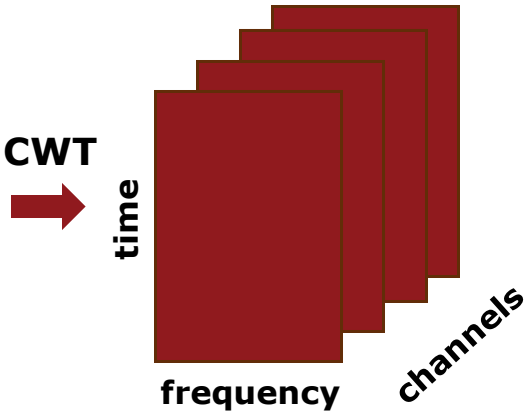
Application$_1$ (Neuroscience): CP has proved useful in epileptic seizure localization

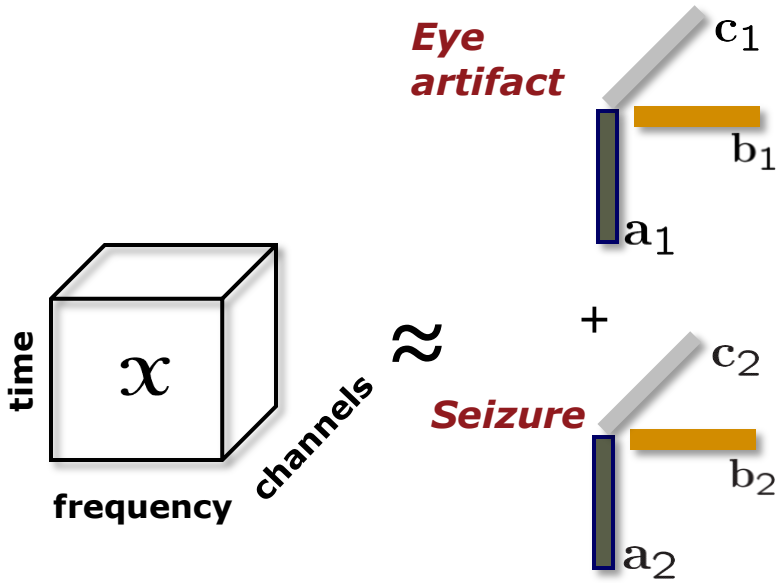
[Acar et al., 2007; De Vos et al., 2007]

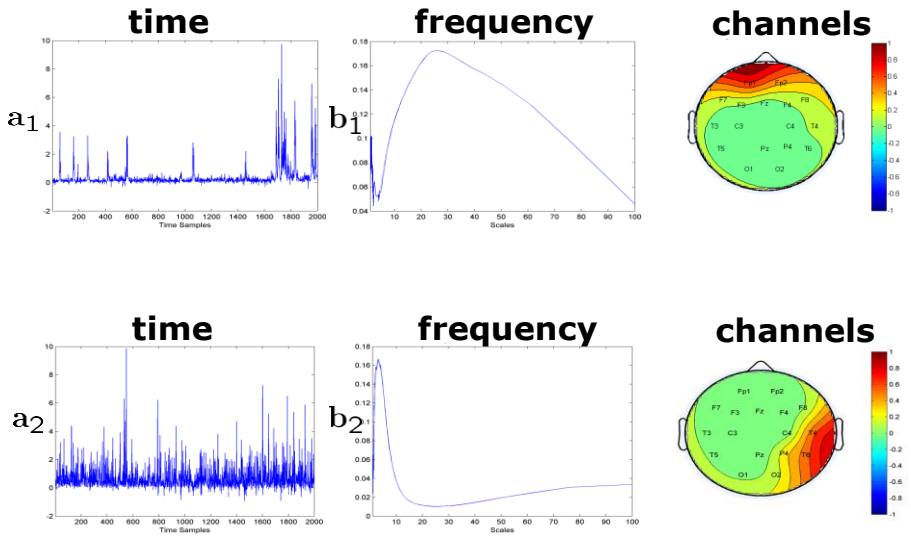










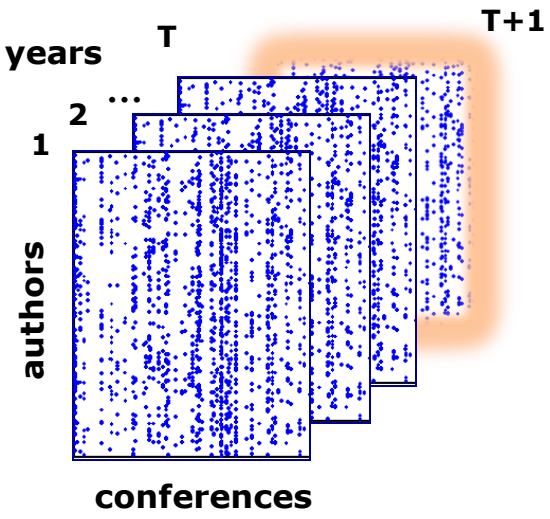


Application$_2$ (Recommender Systems): CP can capture temporal patterns useful for link prediction

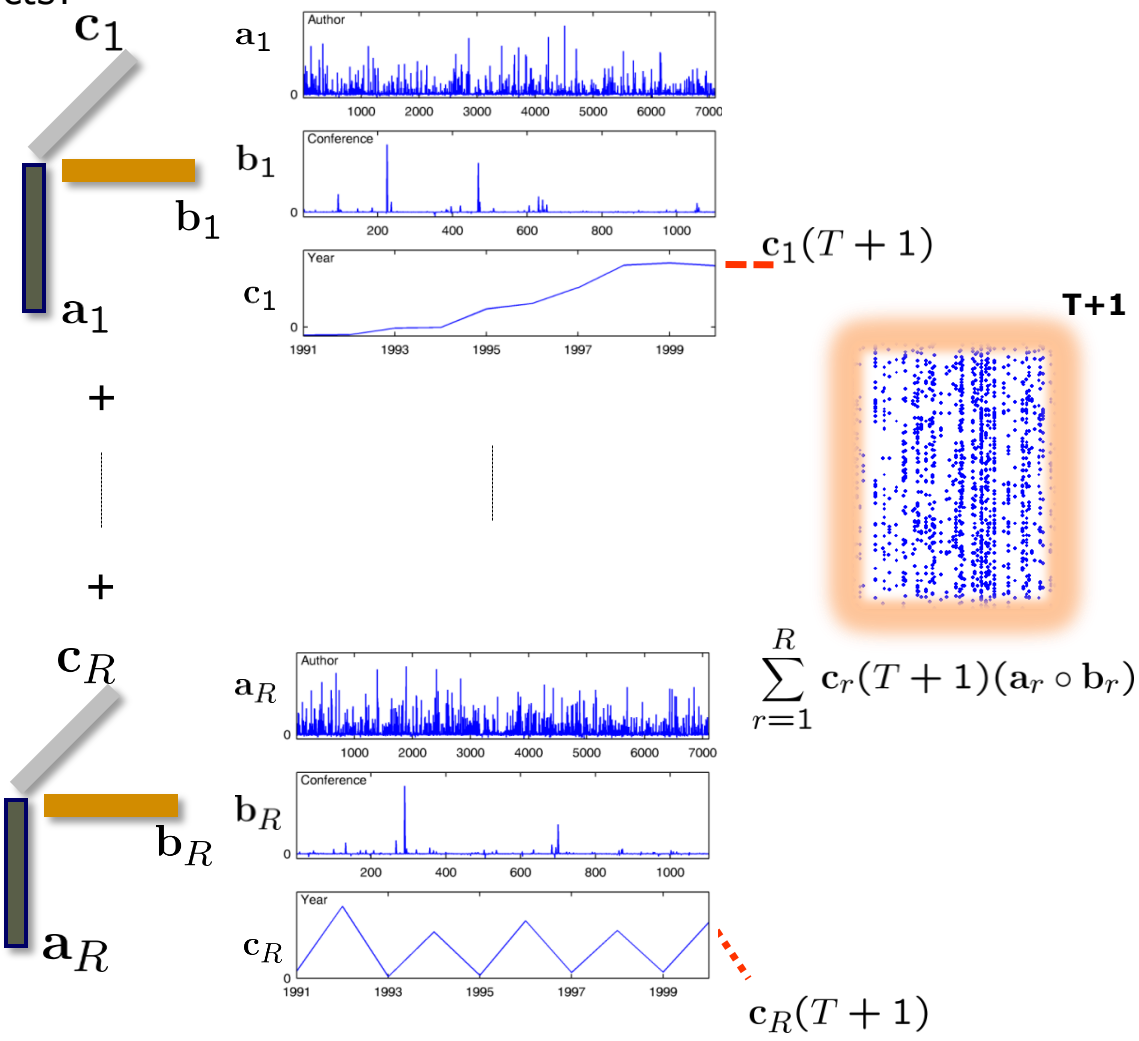
[Dunlavy, Kolda, Acar, 2011]

Temporal Link Prediction

Which customers will buy which products?
Which webpages will users visit?



$\approx$

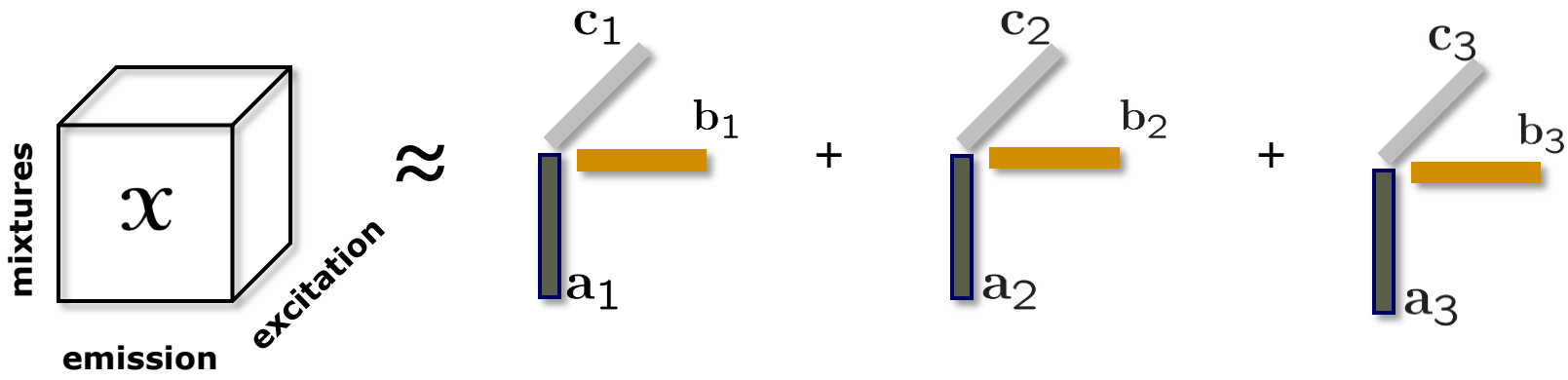


simula

Application$_3$ (Chemometrics): CP can separate the chemicals in mixtures

[Andersen and Bro, 2003]

A popular application of the CP model is the separation of individual chemicals from mixtures of chemicals measured using fluorescence spectroscopy.



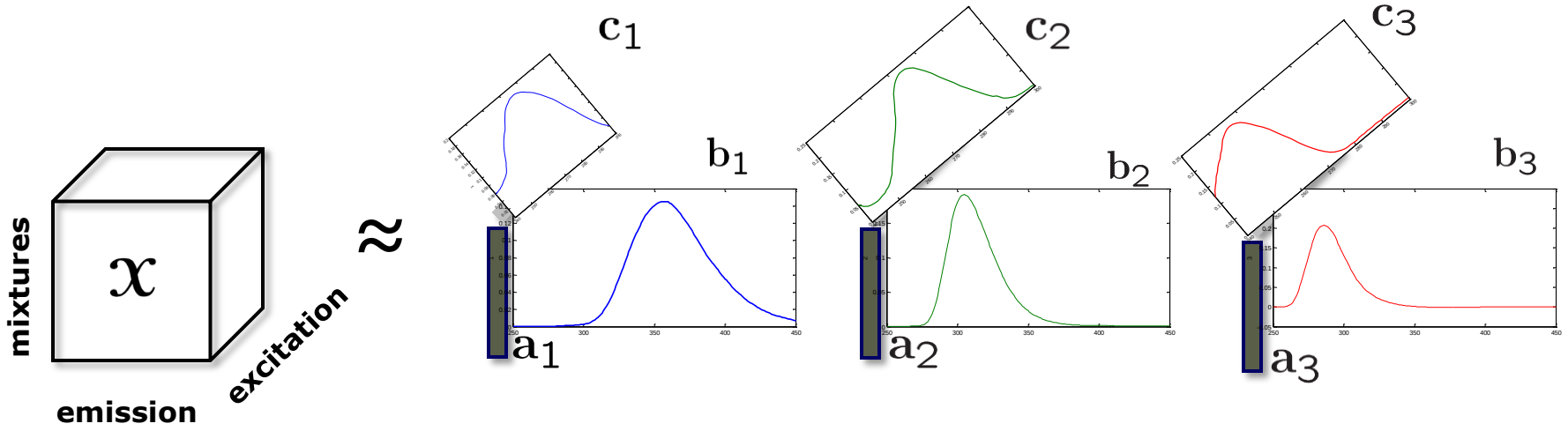
Amino Acid Data
http://www.models.life.ku.dk/

simula

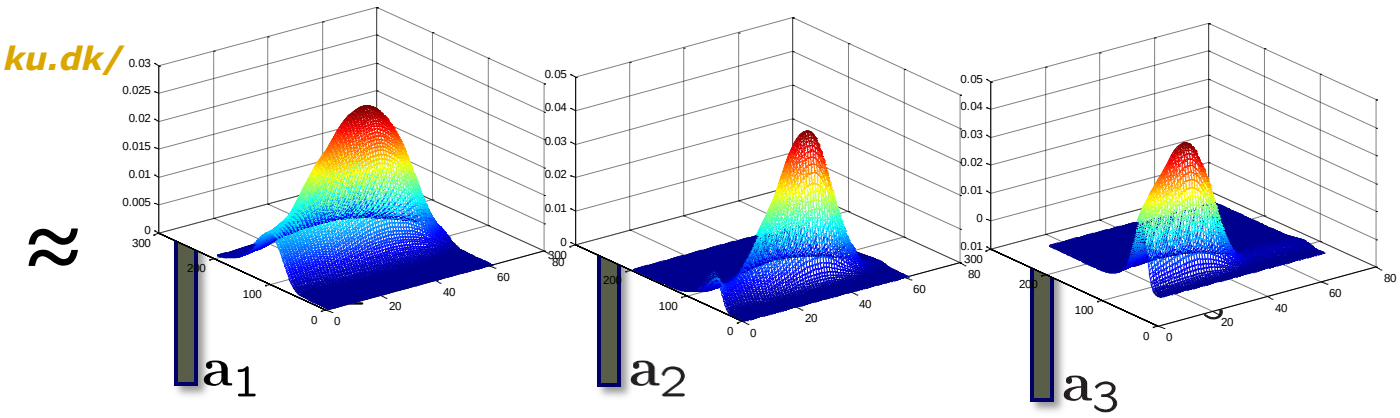
Application$_3$ (Chemometrics): CP can separate the chemicals in mixtures

[Andersen and Bro, 2003]

A popular application of the CP model is the separation of individual chemicals from mixtures of chemicals measured using fluorescence spectroscopy.



Amino Acid Data
http://www.models.life.ku.dk/



simula

Modeling the aminoacids data

```
% Load data
load aminoacids.mat
X = tensor(X.data);
X = X/norm(X);
% Set optimization parameters and fit a CP model using CP-OPT
options                = ncg('defaults');
options.DisplayIters   = 50;
options.RelFuncTol     = 1e-10;
options.StopTol        = 1e-10;
[Fac,FacInit,out]      = cp_opt(X, 3,'init','nvecs','alg','ncg','alg_options',options);
for i=1:3
    subplot(3,1,i)
    plot(Fac.U{i})
end
```

Model: CANDECOMP/PARAFAC (CP)

Algorithm: We can pick a gradient-based optimization method

- Nonlinear Conjugate Gradient (ncg)
- Limited–memory BFGS (lbfgs)
- ….

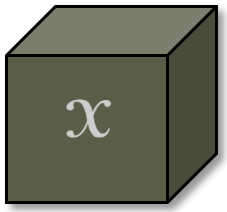
Initialization of factor matrices:

- svd-based
- entries randomly sampled from a standard normal distribution
- pre-defined cell array containing the factor matrices

simula

SVD-based Initialization

Left singular vectors (i.e., $\mathbf{U_1}$, $\mathbf{U_2}$, $\mathbf{U_3}$) of matricized tensors in each mode can be used to initialize the factor matrices.



$$\min_{\mathbf{A},\mathbf{B},\mathbf{C}} \| \mathcal{X} - [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!] \|^2$$

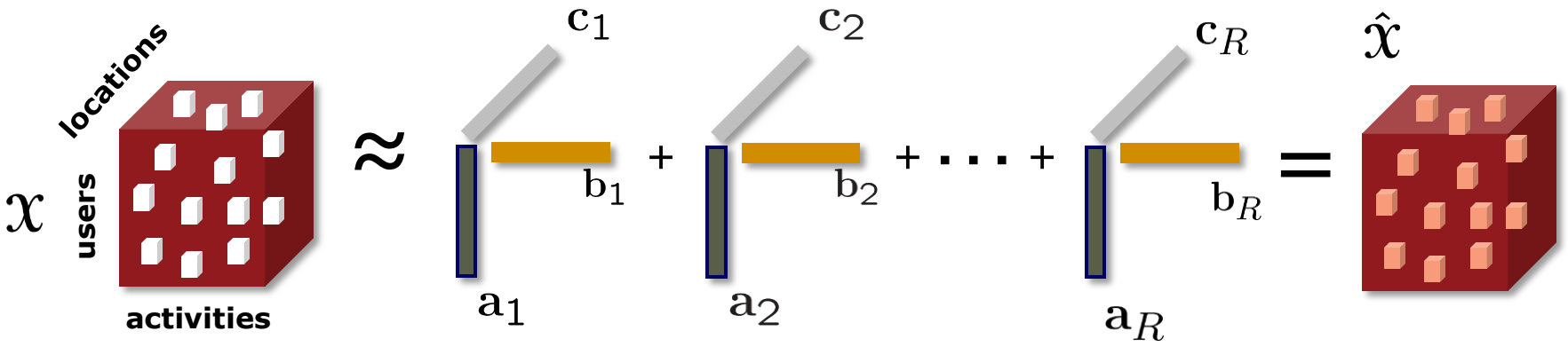
$$\mathbf{X}_{(1)} \approx \mathbf{U}_1 \, \boldsymbol{\Sigma}_1 \, \mathbf{V}_1^{\mathsf{T}}$$

$$\mathbf{X}_{(2)} \approx \mathbf{U}_2 \, \boldsymbol{\Sigma}_2 \, \mathbf{V}_2^{\mathsf{T}}$$

$$\mathbf{X}_{(3)} \approx \mathbf{U}_3 \, \boldsymbol{\Sigma}_3 \, \mathbf{V}_3^{\mathsf{T}}$$

Tensor Factorizations: Missing Data Estimation (a.k.a. Tensor Completion)

Similar to the matrix case, in the case of missing data we can use low-rank tensor approximations to fill in missing entries [Tomasi and Bro, 2005]:



Finding low-rank approximation:

$$\min_{\mathbf{A},\mathbf{B},\mathbf{C}} \| \mathcal{W} * (\mathcal{X} - [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]) \|^2$$

$$w_{ijk} = \begin{cases} 1 & \text{if } x_{ijk} \text{ is known,} \\ 0 & \text{if } x_{ijk} \text{ is missing.} \end{cases}$$

Data reconstruction:

$$\hat{\mathcal{X}} = [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$$

simula

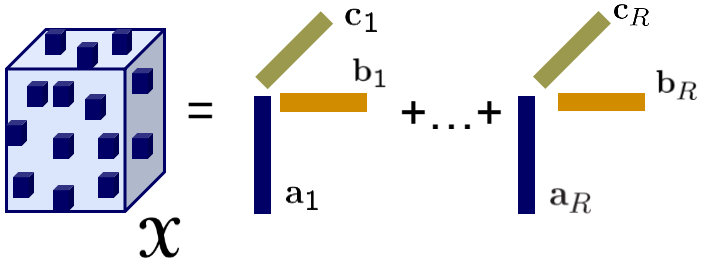
In the presence of missing data: CP-WOPT

[Acar, Dunlavy, Kolda, Mørup, 2011]

CP-WOPT (CP Weighted OPTimization) solves the following weighted optimization problem to fit the CP model to the known data entries in the tensor by ignoring the missing entries:

$$\min_{\mathbf{A},\mathbf{B},\mathbf{C}} \| \mathcal{W} * (\mathcal{X} - [\![\mathbf{A},\mathbf{B},\mathbf{C}]\!]) \|^2$$

$$w_{ijk} = \begin{cases} 1 & \text{if } x_{ijk} \text{ is known,} \\ 0 & \text{if } x_{ijk} \text{ is missing.} \end{cases}$$



Step1: Define the objective function

$$f_{\mathcal{W}}(\mathbf{A},\mathbf{B},\mathbf{C}) = \| \mathcal{W} * (\mathcal{X} - [\![\mathbf{A},\mathbf{B},\mathbf{C}]\!]) \|^2 = \| \mathcal{Y} - \mathcal{Z} \|^2$$

$$\mathcal{Y} = \mathcal{W} * \mathcal{X}$$
$$\mathcal{Z} = \mathcal{W} * [\![\mathbf{A},\mathbf{B},\mathbf{C}]\!]$$

Step2: Compute the gradient

$$\frac{\partial f_{\mathcal{W}}}{\partial \mathbf{A}} = 2(\mathbf{Z}_{(1)} - \mathbf{Y}_{(1)})(\mathbf{C} \odot \mathbf{B})$$

$$\frac{\partial f_{\mathcal{W}}}{\partial \mathbf{B}} = 2(\mathbf{Z}_{(2)} - \mathbf{Y}_{(2)})(\mathbf{C} \odot \mathbf{A})$$

$$\frac{\partial f_{\mathcal{W}}}{\partial \mathbf{C}} = 2(\mathbf{Z}_{(3)} - \mathbf{Y}_{(3)})(\mathbf{B} \odot \mathbf{A})$$

Vectorize and concatenate the partials

$$\nabla f_{\mathcal{W}} = \begin{bmatrix} \frac{\partial f_{\mathcal{W}}}{\partial \mathbf{a}_1} \\ \vdots \\ \frac{\partial f_{\mathcal{W}}}{\partial \mathbf{a}_R} \\ \frac{\partial f_{\mathcal{W}}}{\partial \mathbf{b}_1} \\ \vdots \\ \frac{\partial f_{\mathcal{W}}}{\partial \mathbf{b}_R} \\ \frac{\partial f_{\mathcal{W}}}{\partial \mathbf{c}_1} \\ \vdots \\ \frac{\partial f_{\mathcal{W}}}{\partial \mathbf{c}_R} \end{bmatrix}$$

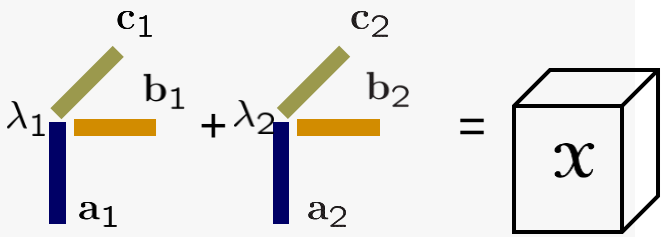
Step3: Pick a first-order optimization method

simula

Tensor Completion Example

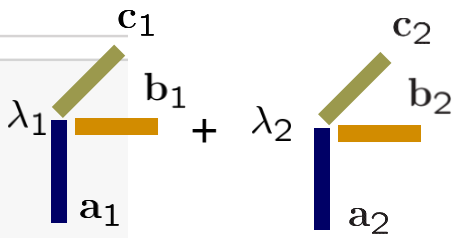
Construct a tensor with an underlying CP model

```
sz = [10 5 3];
R  = 2;
for i=1:length(sz)
    A{i}= randn(sz(i),R);
    for r=1:R
        A{i}(:,r)=A{i}(:,r)/norm(A{i}(:,r));
    end
end
lambda = [2 3]';
X = full(ktensor(lambda, A));
Xorig = X.data;
```



Compare the following with ktensor

```
Y=zeros(sz);
for r=1:R
    for k=1:size(Y,3)
        Y(:,:,k)= Y(:,:,k)+ lambda(r)*A{1}(:,r)*A{2}(:,r)'*A{3}(k,r);
    end
end
```

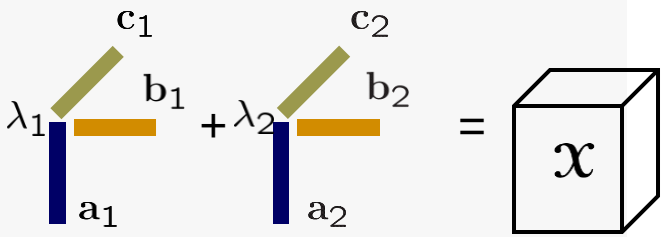


simula

Tensor Completion Example (cont.)

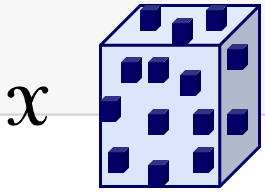
Construct a tensor with an underlying CP model

```
sz = [10 5 3];
R  = 2;
for i=1:length(sz)
    A{i}  = randn(sz(i),R);
    for r = 1:R
      A{i}(:,r)= A{i}(:,r)/norm(A{i}(:,r));
    end
end
lambda = [2 3]';

X     = full(ktensor(lambda,A));
Xorig = X;
```



Pick some random entries and set them to missing

```
M = round(rand(sz));
X(find(M==0))=NaN;
```

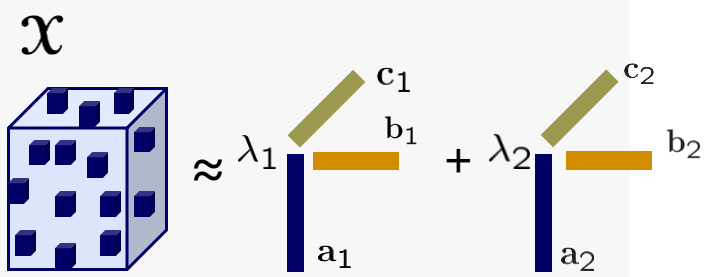


simula

Tensor Completion Example (cont.)

Suppose you are given the incomplete tensor X and you want to find the underlying CP components

```
% Record the places of the missing entries
W = tensor(~isnan(X.data));

% Set those missing entries in the tensor to 0
X(find(W==0))=0;

% parameters for the gradient-based optimization algorithm
options = ncg('defaults');
options.MaxFuncEvals = 10000;
options.MaxIters     = 10000; % change this line to change the max number of iterations
options.StopTol      = 1e-6;
options.RelFuncTol   = 1e-6;
options.DisplayIters = 50;
[Fac, FacInit, output] = cp_wopt(X,W,R,'alg', 'ncg','alg_options',options,'init','nvecs');

% Compare Fac(factors extracted by the model) with A (true factors)
for i=1:3
    corr(Fac.U{i},A{i})
end
```



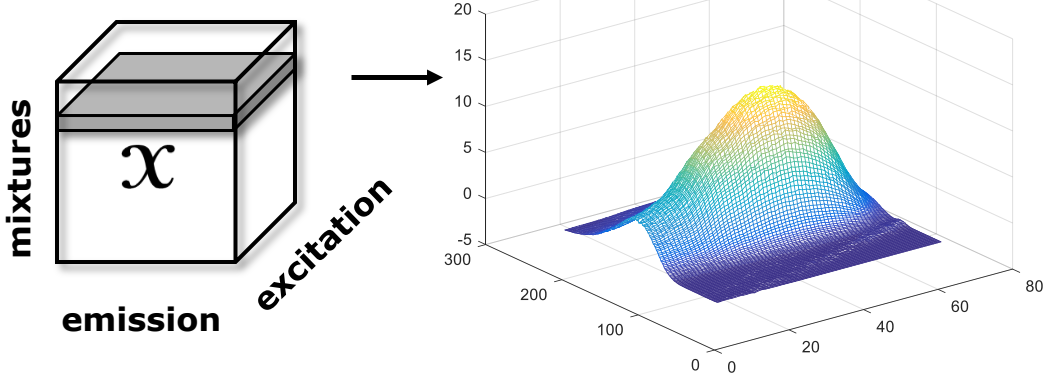
Reconstruct your data from the factor matrices

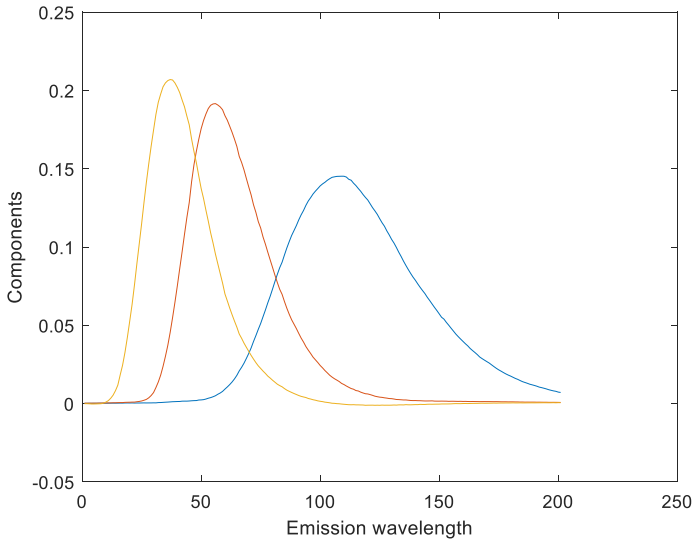
```
Xhat = full(Fac);
% Plot original vs. estimated values for the missing data
plot(Xorig(find(W==0)), Xhat(find(W==0)),'*');
xlabel('Original'); ylabel('Estimated')
```

$$\hat{\mathcal{X}} = [\![\mathrm{A}, \mathrm{B}, \mathrm{C}]\!]$$

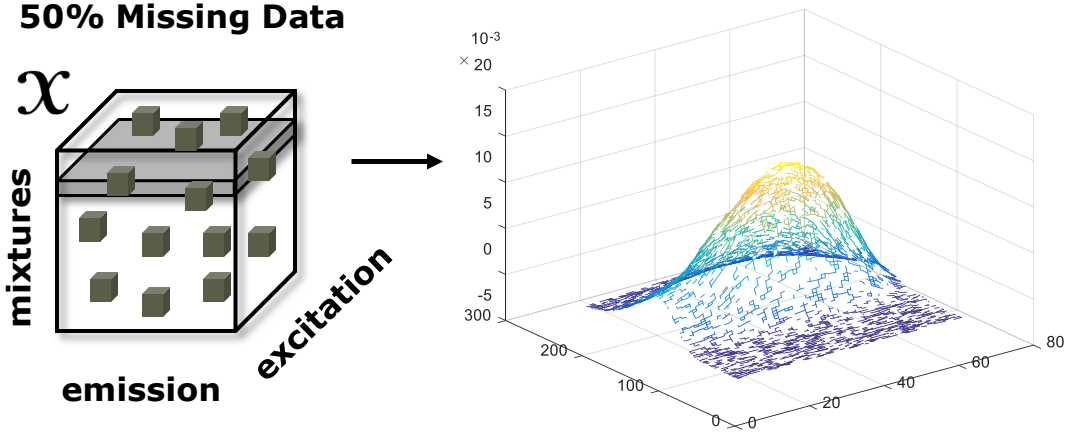
simula

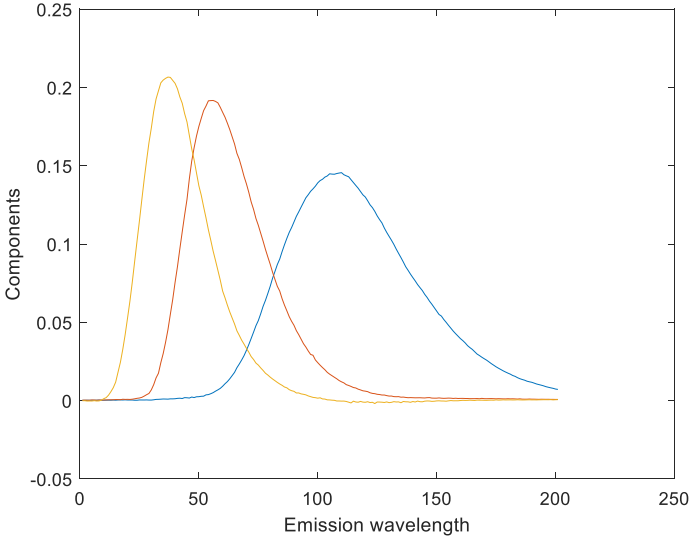
CP-WOPT can accurately capture the factors even with high amounts of missing data!





50% Missing Data

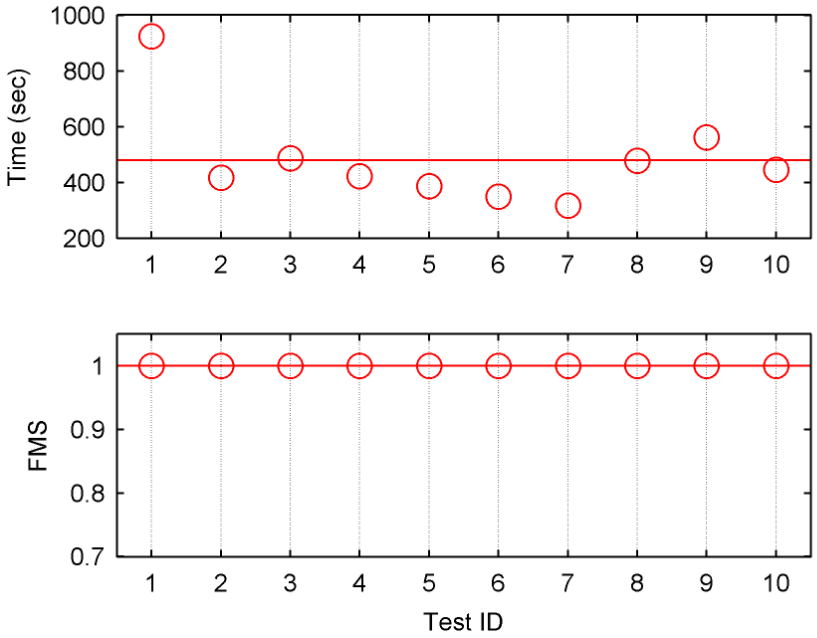




CP-WOPT scales to large problems!

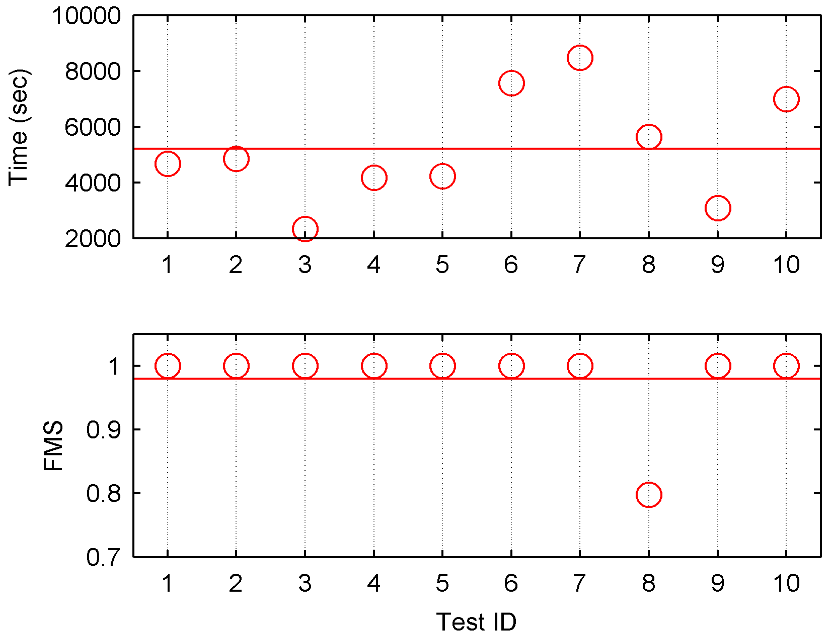
500 x 500 x 500 with 99% missing entries (i.e., 1.25 million known entries)

Dense storage = 1GB
Sparse storage = 40MB



1000 x 1000 x 1000 with 99.5% missing entries (i.e., 5 million known entries)

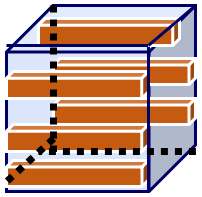
Dense storage = 8GB
Sparse storage = 160MB



Factor Match Score

$$\text{FMS} = \max_{\sigma \in \Pi(R,\bar{R})} \frac{1}{\bar{R}} \sum_{r=1}^{R} \left(1 - \frac{\left| \lambda_r - \bar{\lambda}_{\sigma(r)} \right|}{\max\left\{ \lambda_r, \bar{\lambda}_{\sigma(r)} \right\}} \right) \prod_{n=1}^{N} \left| a_r^{(n)\mathrm{T}} \bar{a}_{\sigma(r)}^{(n)} \right|$$

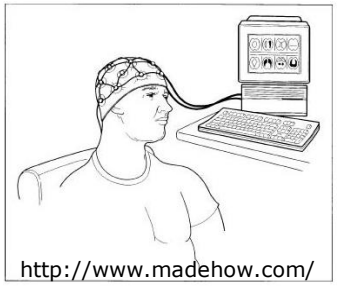
On a Linux workstation with 2 Quad-Core Intel Xeon 3.0 GHz processors and 32 GB RAM

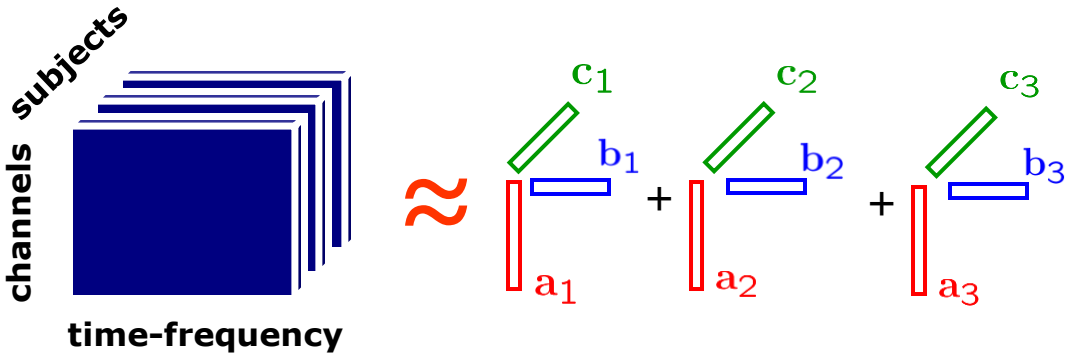


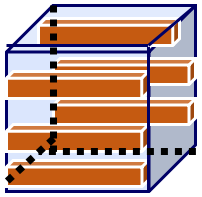
Application$_4$ (Neuroscience): We can capture underlying factors accurately from EEG data with missing channels

[Acar, Dunlavy, Kolda, Mørup, 2011]

Goal: To differentiate between left and right hand stimulation



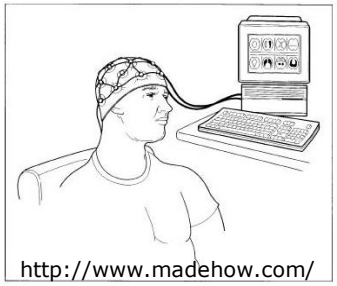


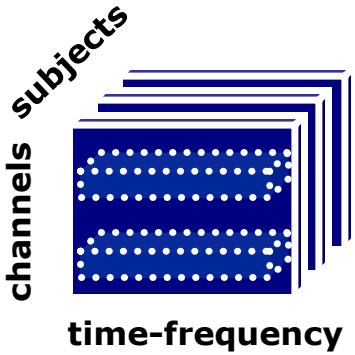


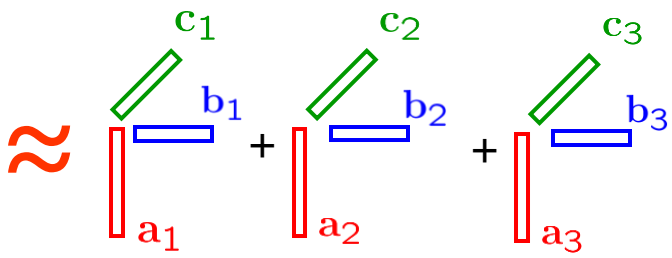
Application$_4$ (Neuroscience): We can capture underlying factors accurately from EEG data with missing channels

[Acar, Dunlavy, Kolda, Mørup, 2011]

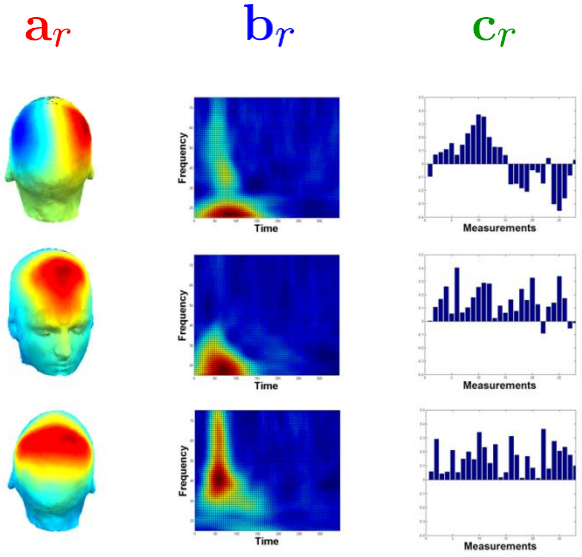
Goal: To differentiate between left and right hand stimulation



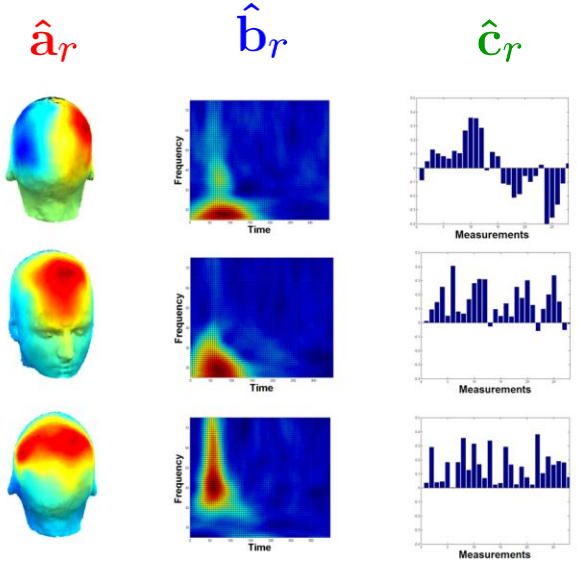


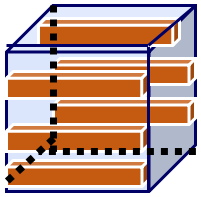


No Missing Data



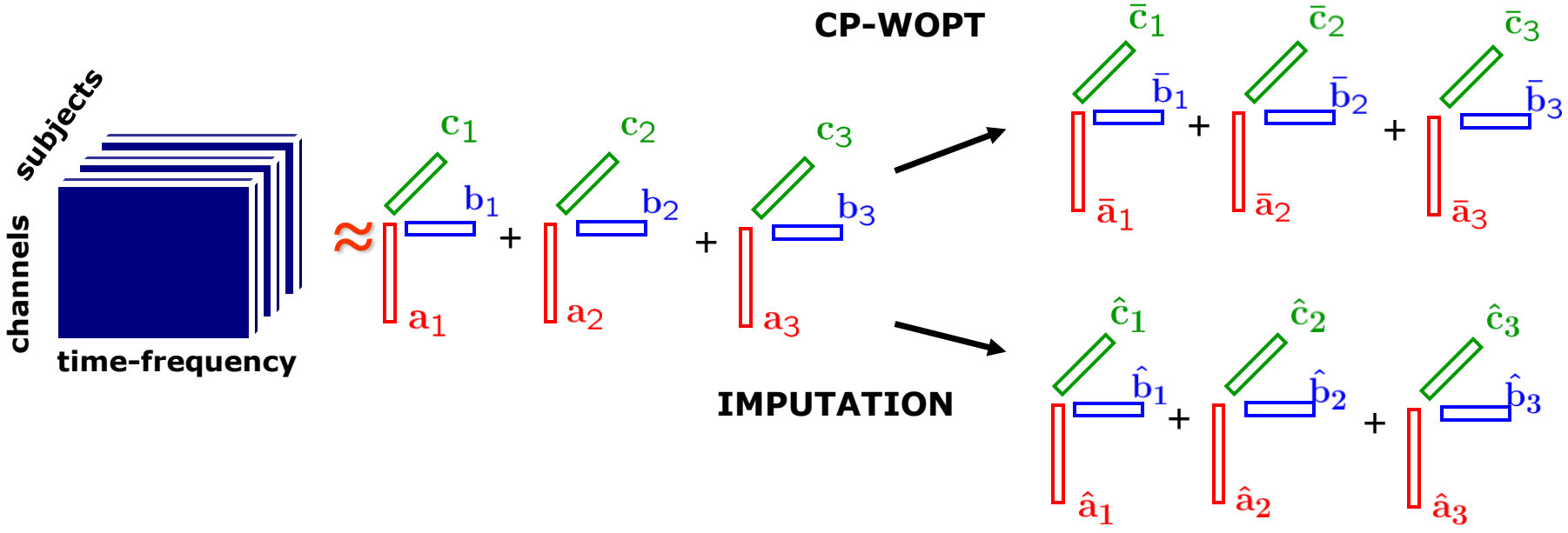
30 Channels/slice Missing

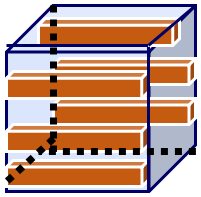




Ignoring missing entries vs. Imputation

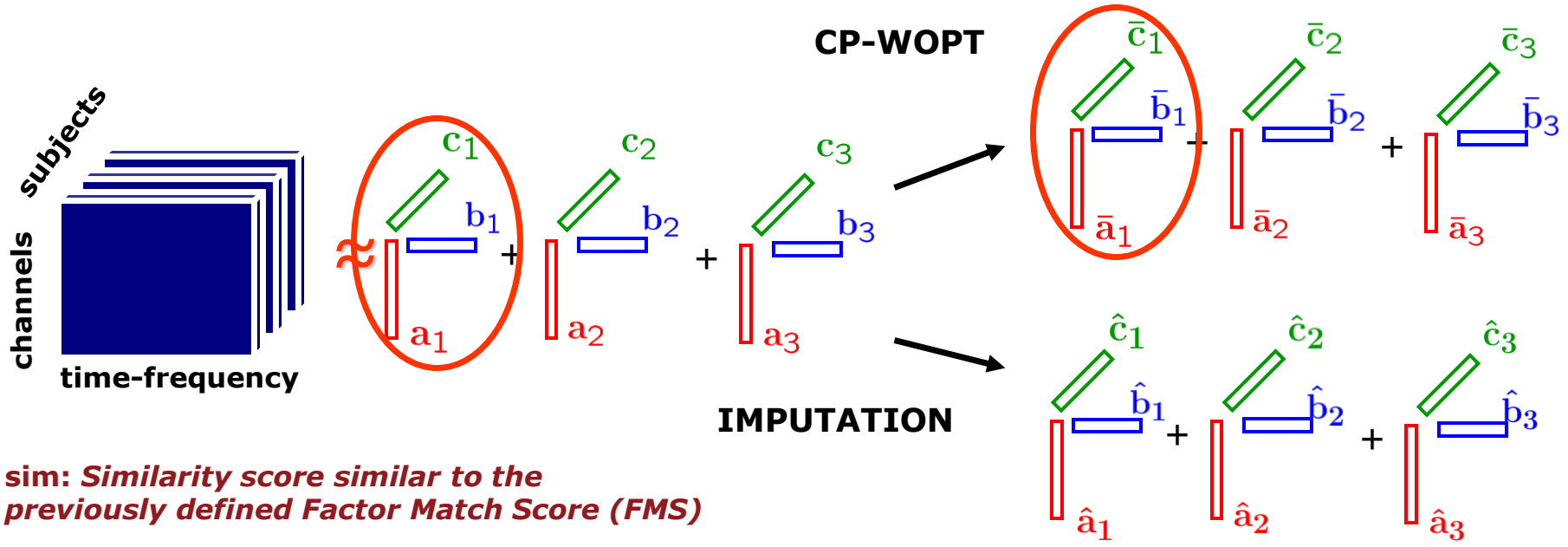
Ignoring missing entries, in particular, when there is large amount of missing data is better than alternatives based on imputation, e.g., imputing missing entries with mean.





Ignoring missing entries vs. Imputation

Ignoring missing entries, in particular, when there is large amount of missing data is better than alternatives based on imputation, e.g., imputing missing entries with mean.



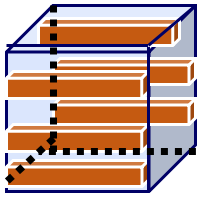
sim: ***Similarity score similar to the previously defined Factor Match Score (FMS)***

CP-WOPT

Missing Channels	sim($r=1$)	sim($r=2$)	sim($r=3$)
1	0.9989	0.9995	0.9991
10	0.9869	0.9936	0.9894
20	0.9560	0.9826	0.9697
30	0.9046	0.9604	0.9312
40	0.6192	0.8673	0.7546

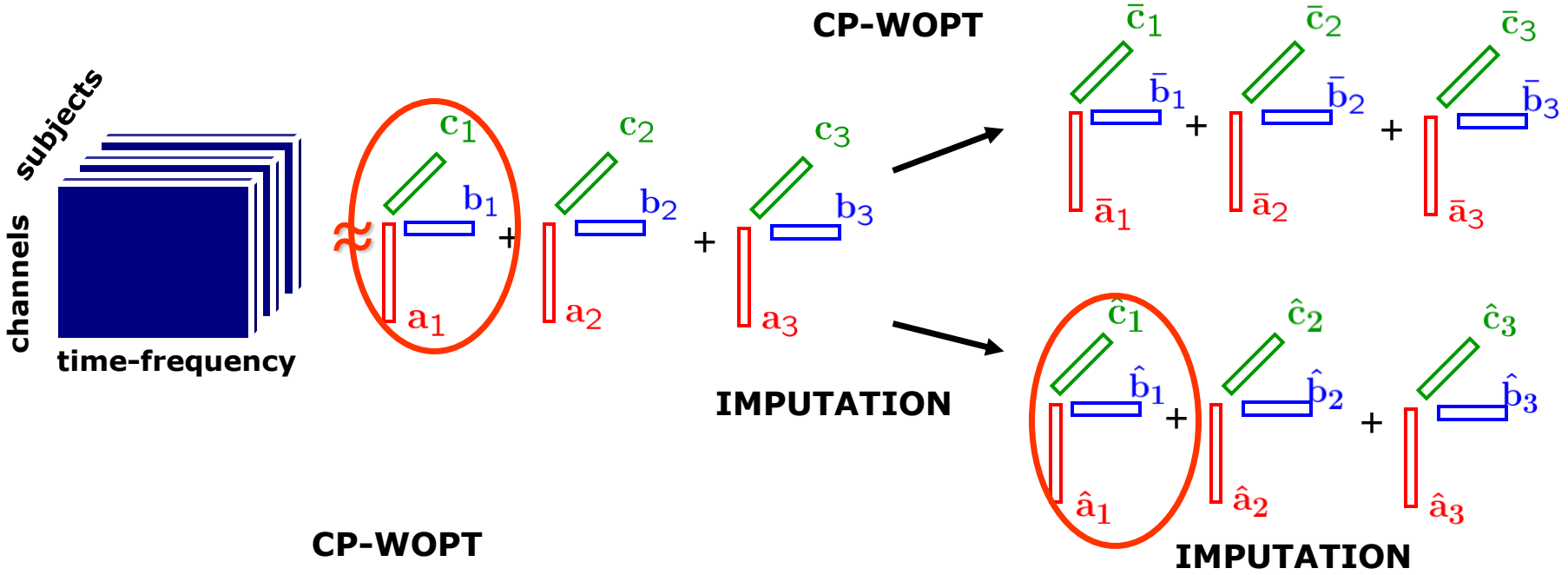
IMPUTATION

Missing Channels	sim($r=1$)	sim($r=2$)	sim($r=3$)
1	0.9970	0.9984	0.9982
10	0.9481	0.9729	0.9752
20	0.9002	0.9475	0.9371
30	0.6435	0.8719	0.8100
40	0.3045	0.7126	0.5699



Ignoring missing entries vs. Imputation

Ignoring missing entries, in particular, when there is large amount of missing data is better than alternatives based on imputation, e.g., imputing missing entries with mean.

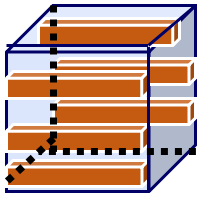


CP-WOPT

Missing Channels	sim($r=1$)	sim($r=2$)	sim($r=3$)
1	0.9989	0.9995	0.9991
10	0.9869	0.9936	0.9894
20	0.9560	0.9826	0.9697
30	0.9046	0.9604	0.9312
40	0.6192	0.8673	0.7546

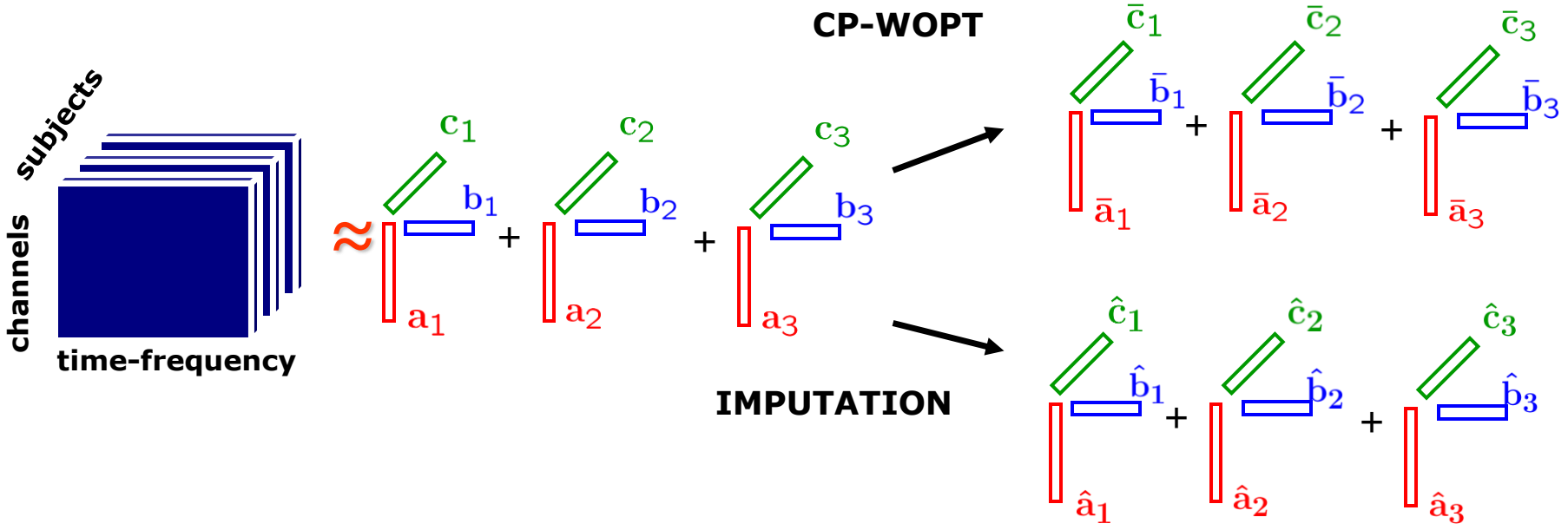
IMPUTATION

Missing Channels	sim($r=1$)	sim($r=2$)	sim($r=3$)
1	0.9970	0.9984	0.9982
10	0.9481	0.9729	0.9752
20	0.9002	0.9475	0.9371
30	0.6435	0.8719	0.8100
40	0.3045	0.7126	0.5699



Ignoring missing entries vs. Imputation

Ignoring missing entries, in particular, when there is large amount of missing data is better than alternatives based on imputation, e.g., imputing missing entries with mean.



CP-WOPT

Missing Channels	sim($r=1$)	sim($r=2$)	sim($r=3$)
1	0.9989	0.9995	0.9991
10	0.9869	0.9936	0.9894
20	0.9560	0.9826	0.9697
30	0.9046	0.9604	0.9312
40	0.6192	0.8673	0.7546

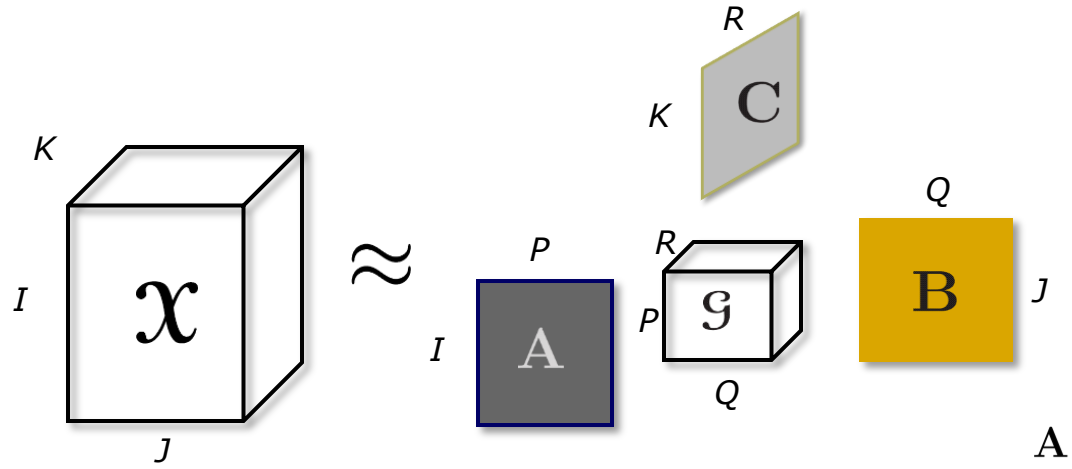
IMPUTATION

Missing Channels	sim($r=1$)	sim($r=2$)	sim($r=3$)
1	0.9970	0.9984	0.9982
10	0.9481	0.9729	0.9752
20	0.9002	0.9475	0.9371
30	0.6435	0.8719	0.8100
40	0.3045	0.7126	0.5699

Another popular tensor factorization model: Tucker

[Tucker, 1963 & 1964 & 1966]

A more flexible tensor factorization model compared to the CP model is the Tucker3 model, which models a third-order tensor using three factor matrices corresponding to each mode as well as a core array.



$$\mathcal{X} \approx \sum_{p=1}^{P} \sum_{q=1}^{Q} \sum_{r=1}^{R} g_{pqr} \mathbf{a}_p \circ \mathbf{b}_q \circ \mathbf{c}_r$$

$$\mathcal{X} \approx [\![\mathcal{G}; \mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$$

$$\mathbf{A} \in \mathbb{R}^{I \times P} = \begin{bmatrix} \mathbf{a}_1 & \cdots & \mathbf{a}_P \end{bmatrix}$$

$$\mathbf{B} \in \mathbb{R}^{J \times Q} = \begin{bmatrix} \mathbf{b}_1 & \cdots & \mathbf{b}_Q \end{bmatrix}$$

$$\mathbf{C} \in \mathbb{R}^{K \times R} = \begin{bmatrix} \mathbf{c}_1 & \cdots & \mathbf{c}_R \end{bmatrix}$$

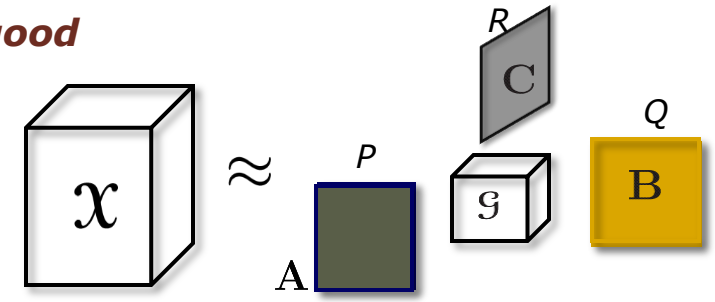
$\mathcal{G} \in \mathbb{R}^{P \times Q \times R}$: core tensor

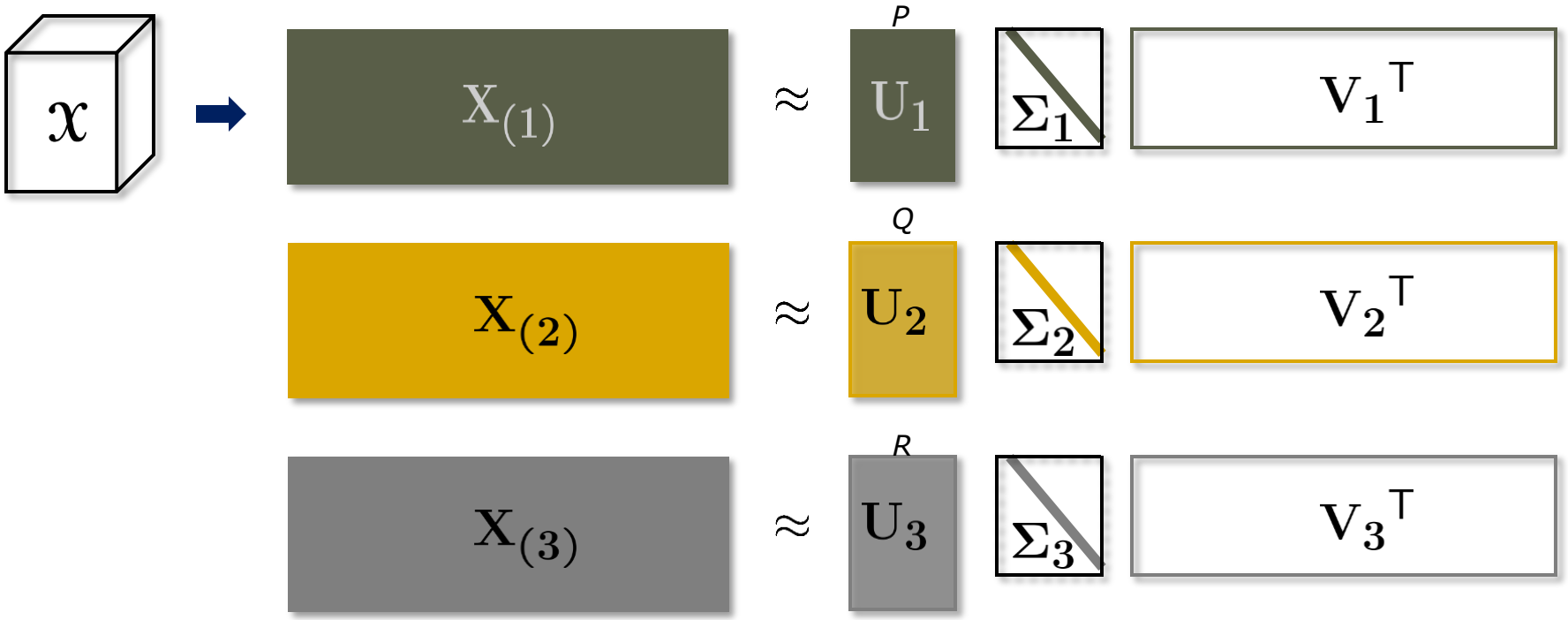
Tucker with orthogonality constraints on A, B, and C corresponds to the best rank-(P,Q,R) approximation of the tensor.

Tucker is considered to be another generalization of SVD to higher-order tensors: Higher-order SVD (HOSVD)

[Tucker, 1966; De Lathauwer, De Moor, Vandewalle, 2000]

Truncation of HOSVD gives a good rank-(P,Q,R) approximation

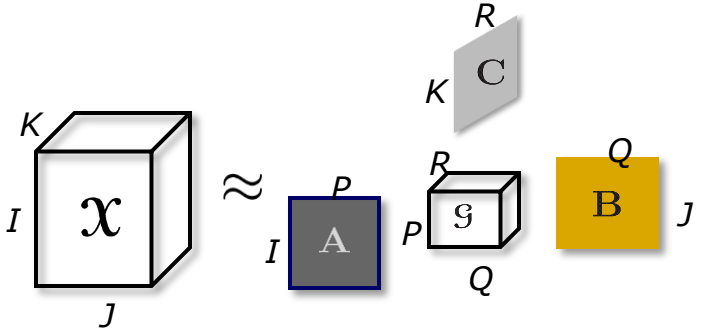




$$\mathcal{G} = \mathcal{X} \times_1 \mathbf{U}_1^{\mathsf{T}} \times_2 \mathbf{U}_2^{\mathsf{T}} \times_3 \mathbf{U}_3^{\mathsf{T}}$$

Tucker vs. CP

Tucker



$$\mathcal{X} \approx \sum_{p=1}^{P} \sum_{q=1}^{Q} \sum_{r=1}^{R} g_{pqr} \mathbf{a}_p \circ \mathbf{b}_q \circ \mathbf{c}_r$$

$$\mathcal{X} \approx [\![\mathcal{G}; \mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$$

More flexible than CP

Different number of components in each mode

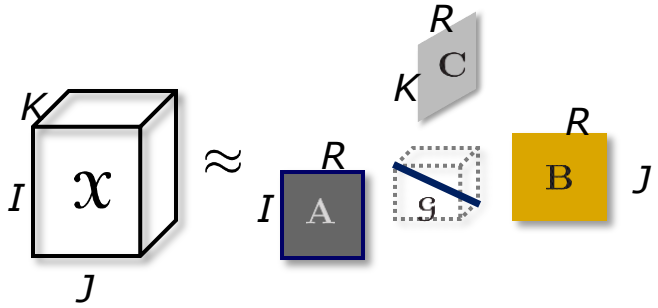
The core tensor models the interaction between factors in different modes

Not-unique

$$[\![\mathcal{G}; \mathbf{A}, \mathbf{B}, \mathbf{C}]\!] = [\![\mathcal{G} \times_1 \mathbf{U}; \mathbf{A}\mathbf{U}^{-1}, \mathbf{B}, \mathbf{C}]\!]$$

EXPLORATORY DATA ANALYSIS & COMPRESSION!

CP



$$\mathcal{X} \approx \sum_{r=1}^{R} g_r \mathbf{a}_r \circ \mathbf{b}_r \circ \mathbf{c}_r$$

Restricted model with a specific structure

Same number of components in each mode

Super-diagonal core tensor

Nice uniqueness properties

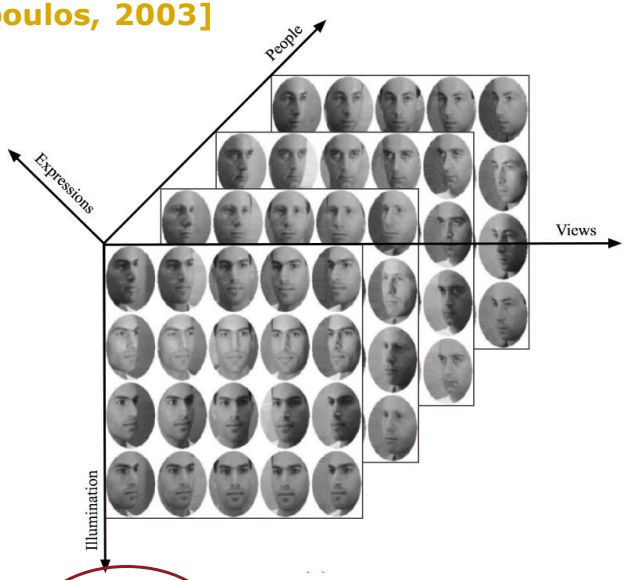
INTERPRETABILITY!!!

Application$_5$ (Face Recognition): TensorFaces

[Vasilescu and Terzopoulos, 2003]

Facial image data tensor with modes (people x viewpoints x illuminations x expressions x pixels) is constructed by arranging the images from 28 people in 5 viewpoints, 4 illuminations and 3 expressions.

$$\mathcal{D} : 28 \times 5 \times 4 \times 3 \times 7943$$



HOSVD gives the following:

$$\mathcal{D} = \mathcal{Z} \times_1 \mathbf{U}_{\text{people}} \times_2 \mathbf{U}_{\text{views}} \times_3 \mathbf{U}_{\text{illums}} \times_4 \mathbf{U}_{\text{expres}} \times_5 \mathbf{U}_{\text{pixels}}$$

eigenfaces

$$\mathcal{Z} \times_5 \mathbf{U}_{\text{pixels}}$$

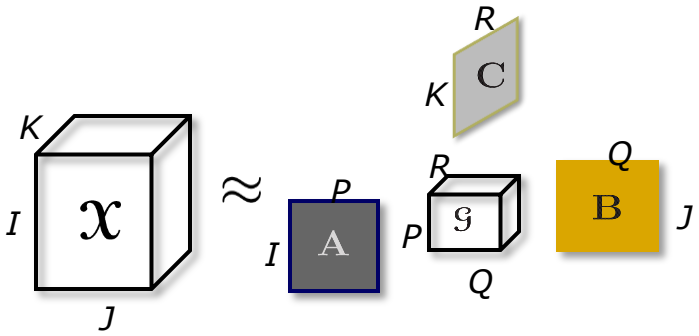
tensorfaces

Better face recognition performance using tensorfaces compared to eigenfaces

Truncation in the illumination mode:

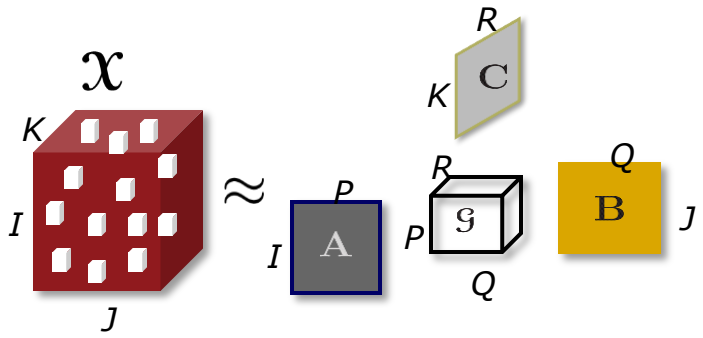
simula

Similar to CP, Tucker model can be fit to incomplete data using weighted optimization!



$$\mathcal{X} \approx [\![\mathcal{G}; \mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$$

$$\min_{\mathcal{G},\mathbf{A},\mathbf{B},\mathbf{C}} \| \mathcal{X} - [\![\mathcal{G}; \mathbf{A}, \mathbf{B}, \mathbf{C}]\!] \|^2$$



$$\min_{\mathcal{G},\mathbf{A},\mathbf{B},\mathbf{C}} \| \mathcal{W} * (\mathcal{X} - [\![\mathcal{G}; \mathbf{A}, \mathbf{B}, \mathbf{C}]\!]) \|^2$$

$$w_{ijk} = \begin{cases} 1 & \text{if } x_{ijk} \text{ is known,} \\ 0 & \text{if } x_{ijk} \text{ is missing.} \end{cases}$$

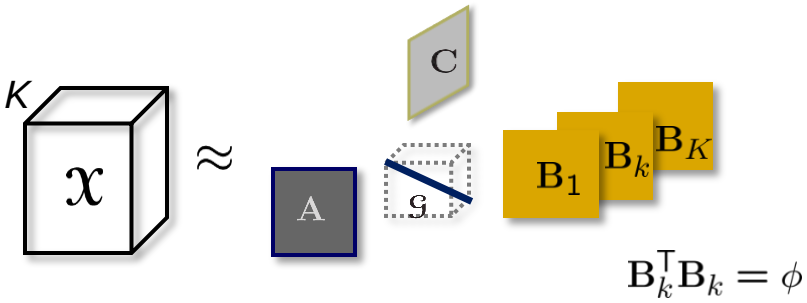
$$\hat{\mathcal{X}} = [\![\mathcal{G}; \mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$$

simula

There are other tensor factorization models

Mainly for applications with uniqueness & interpretability concerns

PARAFAC2 [Harshman, 1972]

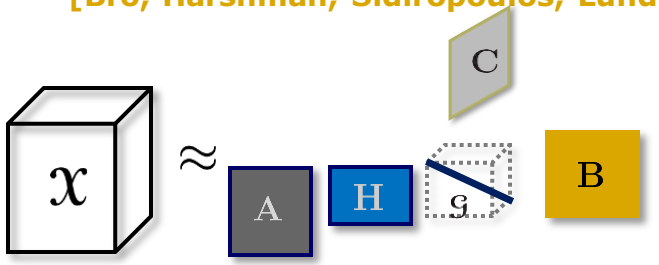


$$\mathbf{B}_k^\top \mathbf{B}_k = \phi$$

Handles differences between factors across different slices. Similar in that sense, there are also Shifted PARAFAC and Convolutive PARAFAC.

PARALIND (Parallel profiles with Linear Dependences)

[Bro, Harshman, Sidiropoulos, Lundy, 2009]



DEDICOM (Decomposition into directional components [Harshman, 1978]

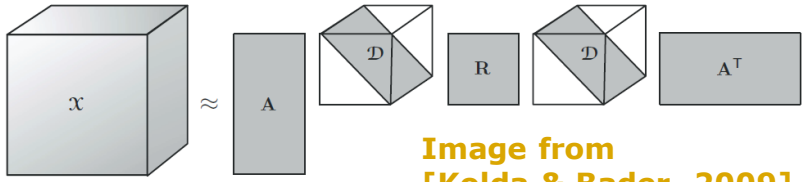


Image from [Kolda & Bader, 2009]

Mainly for compression & reconstruction applications and higher-order tensors with many modes, e.g., 10th-order tensor

Tensor Train (TT) ***And hierarchical tucker (H-Tucker)***

[Oseledets, 2011]

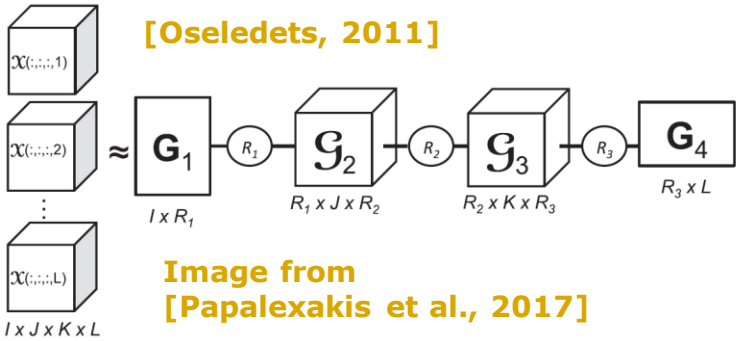


Image from [Papalexakis et al., 2017]

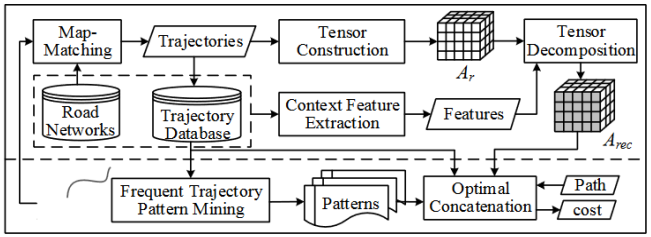
Sequences of SVDs on various unfoldings:

$$\text{SVD of reshape}(\mathcal{X}, [I, JKL]) : \mathbf{G}_1, \mathbf{Z}_1$$
$$\text{SVD of reshape}(\mathbf{Z}_1, [R_1 J, KL]) : \mathcal{G}_2, \mathbf{Z}_2$$
$$\text{SVD of reshape}(\mathbf{Z}_2, [R_2 K, L]) : \mathcal{G}_3, \mathbf{G}_4$$

There are many other applications of tensor factorizations

Urban Computing

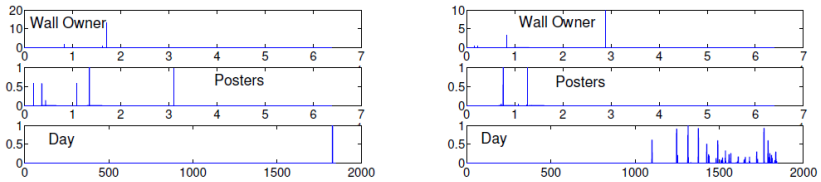
Travel Time Estimation [Wang et al., *KDD,* 2014]



road segments x drivers x time slots

Social Network Analysis

[Papalexakis et al., *PKDD,* 2012]



(a) FACEBOOK anomaly (Wall owner's birthday)

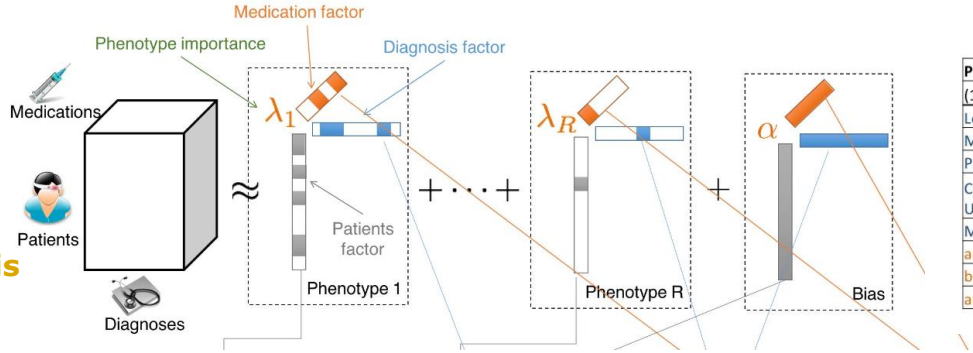
(b) FACEBOOK normal activity

wall owner x poster x day

Electronic Healthcare Records

EHR-based Phenotyping [Henderson et al., *ICHI,* 2017]

patients x medications x diagnosis



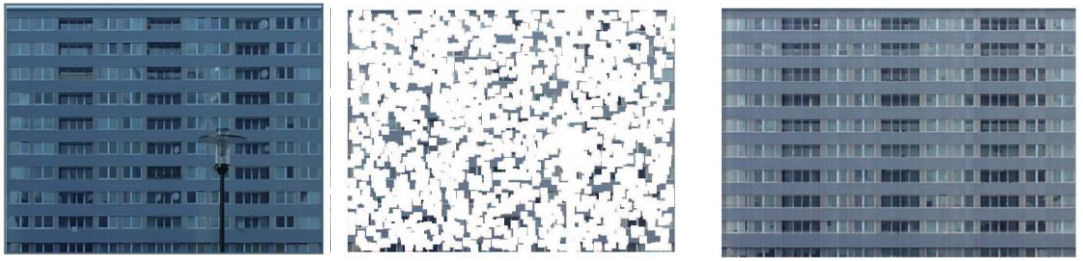
Phenotype 1
(15.43% of Patients)
Legally Blind
Major Symptoms, Abnormalities (1,2)
Polyneuropathy
Cerebrovascular Disease Late Effects, Unspecified
Multiple Sclerosis
anticonvulsants
bronchodilators
anxiolytics, sedatives, and hypnotics

Computer Vision

Image Reconstruction: [Liu et al., *PAMI,* 2013]

Tensor: original color image with color channels

original ***observed*** ***reconstruction***



Active Research Areas

Manifesto from Dagstuhl Perspectives Workshop 16152

Tensor Computing for Internet of Things

SCHLOSS DAGSTUHL
Leibniz-Zentrum für Informatik

[Acar, Anandkumar, Mullin, Rusitschka, Tresp, 2018]

Scalability

Parallel and distributed computations [Smith and Karypis, 2016], [Kaya and Ucar, 2015 & 2016]

Portability to emerging architectures [Phipps et al., 2017]

Sampling [Phan and Cichocki, 2011; Papalexakis et al., 2012; Vervliet and De Lathauwer, 2016; Battaglino et al., 2018]

Different Loss Functions

[Yilmaz, Cemgil, Simsekli, 2011; Ermis, Acar, Cemgil, 2015] [Kolda et al., *Presented at TRICAP*, 2018] [Vandecappelle et al., *Presented at TRICAP*, 2018]

Deep Learning and Tensor Factorizations

Tensor Train Factorization of a Feed Forward Layer [Novikov, Podoprikhin, Osokin, Vetrov, 2015]

TT-Recurrent Neural Networks [Yang, Krompass, Tresp, 2017]

Software

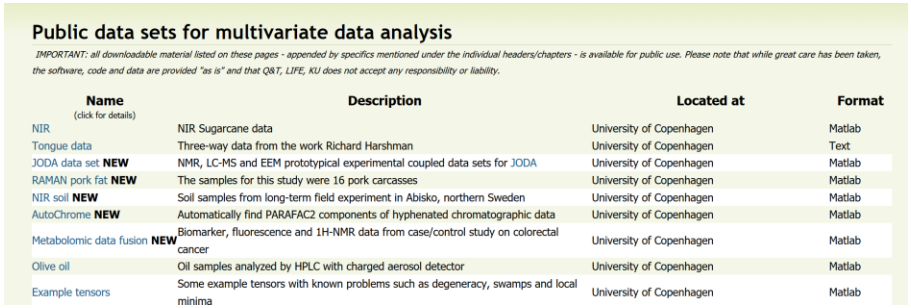
MATLAB: Nway Toolbox, Tensor Toolbox, TT Toolbox, TensorLab

R: multiway

Sparse: *http://frostt.io/*

Data Sets

Dense: *www.models.life.ku.dk*



Public data sets for multivariate data analysis

IMPORTANT: all downloadable material listed on these pages - appended by specifics mentioned under the individual headers/chapters - is available for public use. Please note that while great care has been taken, the software, code and data are provided "as is" and that Q&T, LIFE, KU does not accept any responsibility or liability.

Name (click for details)	Description	Located at	Format
NIR	NIR Sugarcane data	University of Copenhagen	Matlab
Tongue data	Three-way data from the work Richard Harshman	University of Copenhagen	Text
JODA data set **NEW**	NMR, LC-MS and EEM prototypical experimental coupled data sets for JODA	University of Copenhagen	Matlab
RAMAN pork fat **NEW**	The samples for this study were 16 pork carcasses	University of Copenhagen	Matlab
NIR soil **NEW**	Soil samples from long-term field experiment in Abisko, northern Sweden	University of Copenhagen	Matlab
AutoChrome **NEW**	Automatically find PARAFAC2 components of hyphenated chromatographic data	University of Copenhagen	Matlab
Metabolomic data fusion **NEW**	Biomarker, fluorescence and 1H-NMR data from case/control study on colorectal cancer	University of Copenhagen	Matlab
Olive oil	Oil samples analyzed by HPLC with charged aerosol detector	University of Copenhagen	Matlab
Example tensors	Some example tensors with known problems such as degeneracy, swamps and local minima	University of Copenhagen	Matlab

All tensors:

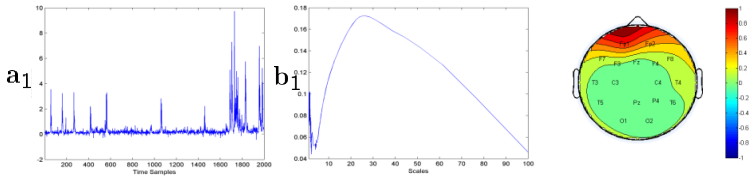
Name	Non-zeros	Order
Amazon Reviews	1,741,809,018	3
Chicago Crime	5,330,673	4
Delicious	140,126,181	4
Enron Emails	54,202,099	4
Flickr	112,890,310	4
LBNL-Network	1,698,825	5
Matrix Multiplication	M*K*N	3
NELL-1	143,599,552	3
NELL-2	76,879,419	3
NIPS Publications	3,101,609	4
Patents	3,596,640,708	3
Reddit-2015	4,687,474,081	3
Uber Pickups	3,309,490	4
VAST 2015 Mini-Challenge 1	26,021,945	5

Summary

Matrix Factorizations: How to use matrix factorizations to (i) find the underlying factors, (ii) predict missing entries

Matrix factorizations without constraints are not unique.

Constraints need to make sense in terms of the application; otherwise, the model will not reveal what we are looking for.



Tensor Factorizations

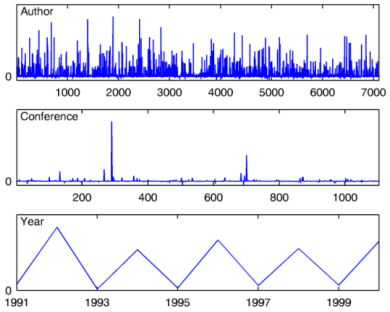
Tensor factorization methods such as CP have better uniqueness properties compared to matrix factorizations.

How to use tensor factorizations, in particular, CP, to find the underlying factors, e.g.,

Chemometrics

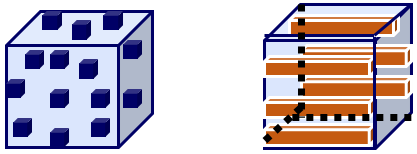
Neuroscience

Temporal Link Prediction



How to fit a CP model to a higher-order tensor with missing entries

Algorithms: ALS, Gradient-based optimization approaches



MATLAB functions

cp-opt: Fitting a CP model to a tensor **(from Tensor Toolbox)**

```
[Fac, FacInit, output] = cp_opt(X, R, 'init', 'random', 'alg','ncg', 'alg_options', options);
```

cp-wopt: Fitting a CP model to a tensor with missing entries **(from Tensor Toolbox)**

```
[Fac, FacInit, output] = cp_wopt(X, W, R, 'init', 'random', 'alg','ncg', 'alg_options', options);
```

Model: Optimization problem

$$\min_{\mathbf{A},\mathbf{B},\mathbf{C}} \| \mathcal{X} - [\![\mathbf{A},\mathbf{B},\mathbf{C}]\!] \|^2 \qquad \min_{\mathbf{A},\mathbf{B},\mathbf{C}} \| \mathcal{W} * (\mathcal{X} - [\![\mathbf{A},\mathbf{B},\mathbf{C}]\!]) \|^2$$

Tensor Toolbox
Computation of the function value and the gradient

$$f(\mathbf{A},\mathbf{B},\mathbf{C}) = \frac{1}{2} \| \mathcal{X} - [\![\mathbf{A},\mathbf{B},\mathbf{C}]\!] \|^2$$

$$\nabla f = \begin{bmatrix} \frac{\partial f}{\partial \mathbf{a}_1} \\ \vdots \\ \frac{\partial f}{\partial \mathbf{a}_R} \\ \frac{\partial f}{\partial \mathbf{b}_1} \\ \vdots \\ \frac{\partial f}{\partial \mathbf{b}_R} \\ \frac{\partial f}{\partial \mathbf{c}_1} \\ \vdots \\ \frac{\partial f}{\partial \mathbf{c}_R} \end{bmatrix} \quad \begin{aligned} \frac{\partial f}{\partial \mathbf{A}} &= -\mathbf{X}_{(1)}(\mathbf{C} \odot \mathbf{B}) + \mathbf{A}(\mathbf{C}^\mathsf{T}\mathbf{C} * \mathbf{B}^\mathsf{T}\mathbf{B}) \\ \frac{\partial f}{\partial \mathbf{B}} &= -\mathbf{X}_{(2)}(\mathbf{C} \odot \mathbf{A}) + \mathbf{B}(\mathbf{C}^\mathsf{T}\mathbf{C} * \mathbf{A}^\mathsf{T}\mathbf{A}) \\ \frac{\partial f}{\partial \mathbf{C}} &= -\mathbf{X}_{(3)}(\mathbf{B} \odot \mathbf{A}) + \mathbf{C}(\mathbf{B}^\mathsf{T}\mathbf{B} * \mathbf{A}^\mathsf{T}\mathbf{A}) \end{aligned}$$

cp_opt

$$f_{\mathcal{W}}(\mathbf{A},\mathbf{B},\mathbf{C}) = \| \mathcal{W} * (\mathcal{X} - [\![\mathbf{A},\mathbf{B},\mathbf{C}]\!]) \|^2 = \| \mathcal{Y} - \mathcal{Z} \|^2$$

$$\nabla f_{\mathcal{W}} = \begin{bmatrix} \frac{\partial f_{\mathcal{W}}}{\partial \mathbf{a}_1} \\ \vdots \\ \frac{\partial f_{\mathcal{W}}}{\partial \mathbf{a}_R} \\ \frac{\partial f_{\mathcal{W}}}{\partial \mathbf{b}_1} \\ \vdots \\ \frac{\partial f_{\mathcal{W}}}{\partial \mathbf{b}_R} \\ \frac{\partial f_{\mathcal{W}}}{\partial \mathbf{c}_1} \\ \vdots \\ \frac{\partial f_{\mathcal{W}}}{\partial \mathbf{c}_R} \end{bmatrix} \quad \begin{aligned} \frac{\partial f_{\mathcal{W}}}{\partial \mathbf{A}} &= 2(\mathbf{Z}_{(1)} - \mathbf{Y}_{(1)})(\mathbf{C} \odot \mathbf{B}) \\ \frac{\partial f_{\mathcal{W}}}{\partial \mathbf{B}} &= 2(\mathbf{Z}_{(2)} - \mathbf{Y}_{(2)})(\mathbf{C} \odot \mathbf{A}) \\ \frac{\partial f_{\mathcal{W}}}{\partial \mathbf{C}} &= 2(\mathbf{Z}_{(3)} - \mathbf{Y}_{(3)})(\mathbf{B} \odot \mathbf{A}) \end{aligned}$$

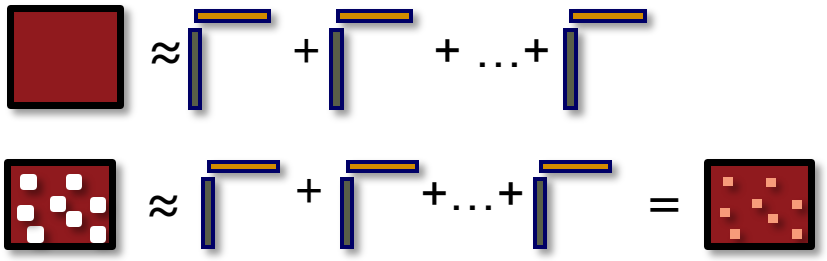
cp_wopt

Poblano Toolbox
Gradient-based optimization algorithms

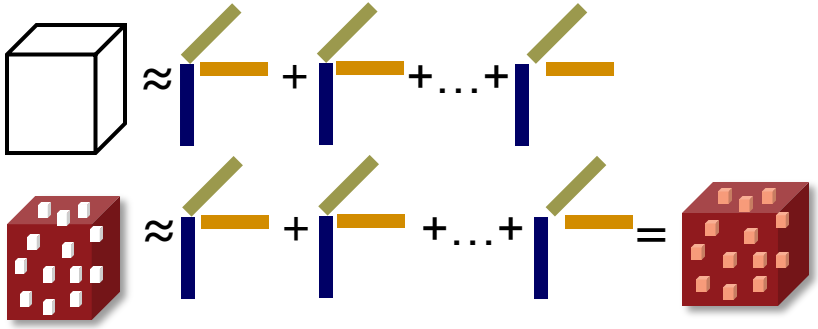
ncg, lbfgs, …

Tutorial Outline

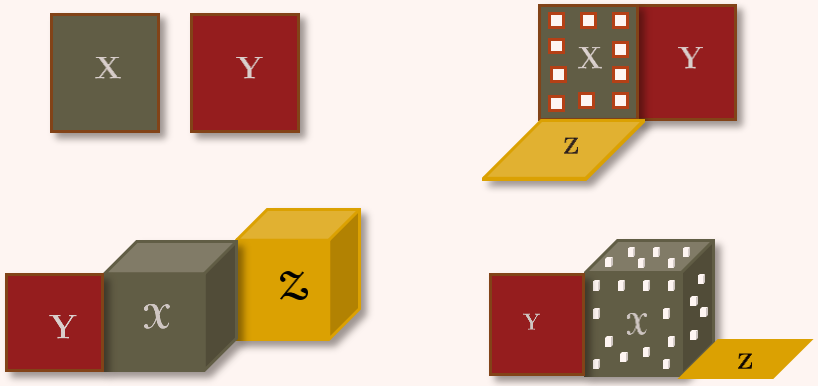
Matrix Factorizations



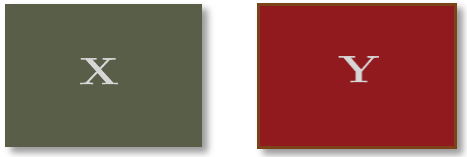
Tensor Factorizations



Data Fusion based on Coupled Matrix and Tensor Factorizations



Data Fusion based on Coupled Factorizations



$$\min_{\mathbf{U},\mathbf{V},\mathbf{W}} \left\| \mathbf{X} - \mathbf{U}\mathbf{V}^{\mathsf{T}} \right\|^2 + \left\| \mathbf{Y} - \mathbf{U}\mathbf{W}^{\mathsf{T}} \right\|^2$$

Psychometrics: Simultaneous factorization of Gramian matrices **[Levin, 1966]**; Simultaneous Component Analysis **[Kiers and Ten Berge, 1994]**

Chemometrics: Principal Component Analysis of multiple matrices **[Westerhuis et al., 1998]**; Augmented Multivariate Curve Resolution **[De Juan and Tauler, 2000]**

Bioinformatics: Comparing genome-scale expression data from multiple organisms **[Alter et al., 2003; Ponnapalli et al., 2011]**; Clustering microarray data **[Badea, 2007]**; Simultaneous Component Analysis with rotation to common and distinct components **[Van Deun et al., 2012]**; Decomposing multiple matrices into terms explaining joint and individual variation **[Lock et al., 2013]**

Signal Processing: Joint diagonalization of multiple matrices **[Yeredor, 2002; Ziehe et al., 2004]**; Audio source separation **[Yoo et al., 2010]**; Joint Independent Component Analysis **[Calhoun et al., 2006]**; Soft nonnegative matrix co-factorization **[Seichepine et al., 2013]**

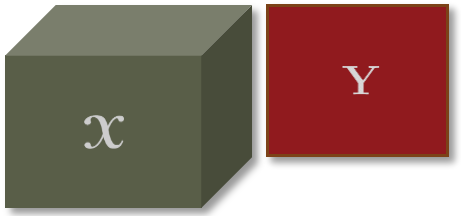
Data Mining: Collective matrix factorization **[Singh and Gordon, 2008]**; Clustering multi-type relational data **[Long et al., 2006]**; Social recommendation **[Ma et al., 2008]**; Coupled matrix factorization with sparse factors **[Van Deun et al., 2011; Acar et al., 2012]**; Nonnegative shared subspace learning **[Gupta et al., 2010]**; Bayesian interbattery factor analysis **[Klami et al., 2013]**

Data Fusion based on Coupled Factorizations

$$\min_{\mathbf{U},\mathbf{V},\mathbf{W}} \left\| \mathbf{X} - \mathbf{U}\mathbf{V}^{\mathsf{T}} \right\|^2 + \left\| \mathbf{Y} - \mathbf{U}\mathbf{W}^{\mathsf{T}} \right\|^2$$

psychometrics, chemometrics, bioinformatics, signal processing, data mining, …

Cannot handle joint analysis of matrices and higher-order tensors!



$$\min_{\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D}} \| \mathcal{X} - [\![\mathbf{A},\mathbf{B},\mathbf{C}]\!] \|^2 + \| \mathbf{Y} - \mathbf{A}\mathbf{D}^{\mathsf{T}} \|^2$$

Psychometrics: Linked-mode PARAFAC **[Harshman and Lundy, 1984]**

Chemometrics: Multi-way Multi-block component models **[Smilde et al., 2000]**

Bioinformatics: Coupled analysis of in vitro and histology tissue samples **[Acar et al., 2012]**

Signal Processing: Joint analysis of a covariance matrix and a cumulant tensor **[De Lathauwer and Vandewalle, 2004; Comon, 2004]**; Generalized Coupled Tensor Factorizations **[Yilmaz et al., 2011]**; Structured Data Fusion **[Sorber et al., 2015]**

Data Mining: Multi-way Clustering **[Banerjee et al., 2007]**; Community detection **[Lin et al., 2009]**; Missing value estimation **[Zheng et al., 2010]**; Link prediction **[Ermis et al., 2012]**; Scalable CMTF approaches (sampling–based **[Papalexakis et al., 2014]**, distributed stochastic gradient running on MAPREDUCE **[Beutel et al., 2014]**, distributed ALS running on MAPREDUCE **[Jeon et al., 2016]**)

Coupled Matrix Factorizations (CMF)

$$\mathbf{X} \approx \mathbf{A}\mathbf{B}^\mathsf{T} \qquad \mathbf{Y} \approx \mathbf{A}\mathbf{C}^\mathsf{T}$$

$$\min_{\mathbf{A},\mathbf{B},\mathbf{C}} \left\| \mathbf{X} - \mathbf{A}\mathbf{B}^\mathsf{T} \right\|^2 + \left\| \mathbf{Y} - \mathbf{A}\mathbf{C}^\mathsf{T} \right\|^2$$

Objective function:

$$f(\mathbf{A},\mathbf{B},\mathbf{C}) = \underbrace{\left\| \mathbf{X} - \mathbf{A}\mathbf{B}^\mathsf{T} \right\|^2}_{f_1} + \underbrace{\left\| \mathbf{Y} - \mathbf{A}\mathbf{C}^\mathsf{T} \right\|^2}_{f_2}$$

Compute the gradient:

$$\frac{\partial f}{\partial \mathbf{A}} = \frac{\partial f_1}{\partial \mathbf{A}} + \frac{\partial f_2}{\partial \mathbf{A}}$$

$$\frac{\partial f}{\partial \mathbf{B}} = \frac{\partial f_1}{\partial \mathbf{B}}$$

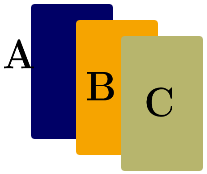
$$\frac{\partial f}{\partial \mathbf{C}} = \frac{\partial f_2}{\partial \mathbf{C}}$$

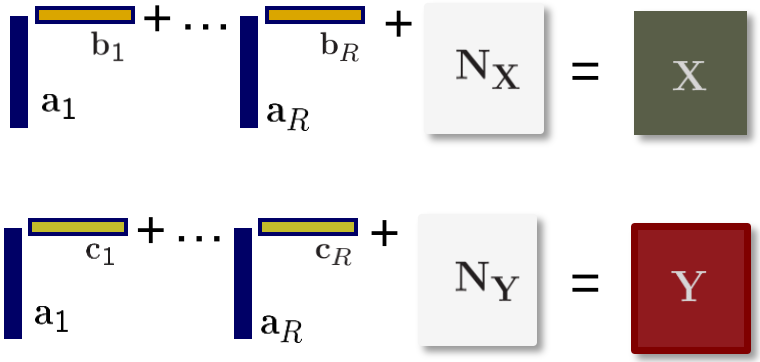
Use a first-order optimization method to fit the CMF model

simula

Coupled Matrix Factorizations: Example

Generate factor matrices





$$\mathbf{X} = \mathbf{A}\mathbf{B}^{\mathsf{T}} + \eta\mathbf{N}_{\mathbf{X}}$$
$$\mathbf{Y} = \mathbf{A}\mathbf{C}^{\mathsf{T}} + \eta\mathbf{N}_{\mathbf{Y}}$$

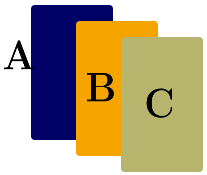
Construct coupled matrices

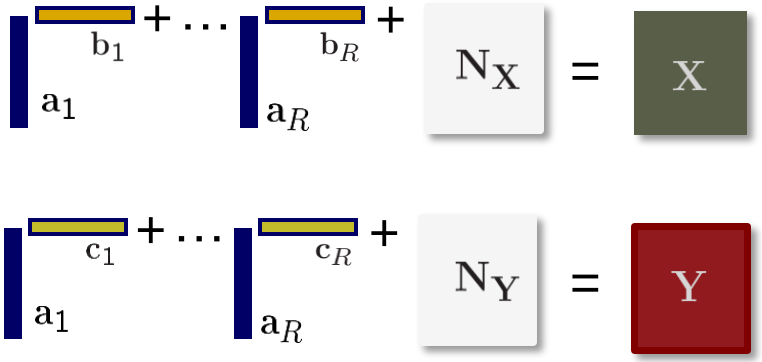
```
sz    = [30 20 10];
modes = {[1 2],[1 3]};
R  = 2;
for i=1:length(sz)
    A{i}  = randn(sz(i),R);
    for r = 1:R
      A{i}(:,r)= A{i}(:,r)/norm(A{i}(:,r));
    end
end
X{1} = A{1}*A{2}';
X{2} = A{1}*A{3}';
% add noise
% for i=1:2
%     N{i} = randn(size(X{i}));
%     X{i} = X{i} + 0.1* N{i}/norm(N{i},'fro')*norm(X{i},'fro');
% end

P = length(X);
for p=1:P
    Z.object{p} = tensor(X{p});
    Z.object{p} = Z.object{p}/norm(Z.object{p});
end
Z.modes = modes;
Z.size  = sz;
```

simula

Coupled Matrix Factorizations: Example (cont.)

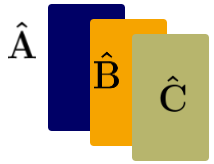
Generate factor matrices





Solve

$$\min_{A,B,C} \left\| X - AB^T \right\|^2 + \left\| Y - AC^T \right\|^2$$



$$X = AB^T + \eta N_X$$
$$Y = AC^T + \eta N_Y$$

Suppose you are given coupled matrices and want to jointly factorize them

```
options = ncg('defaults');
options.MaxFuncEvals = 10000;
options.MaxIters     = 10000;
options.StopTol      = 1e-6;
options.RelFuncTol   = 1e-6;
%options.DisplayIters = 50;
[Fac, FacInit, out] = cmtf_opt(Z,R,'alg', 'ncg','alg_options',options,'init','random');
```

Compare the reconstructed matrices with the originals

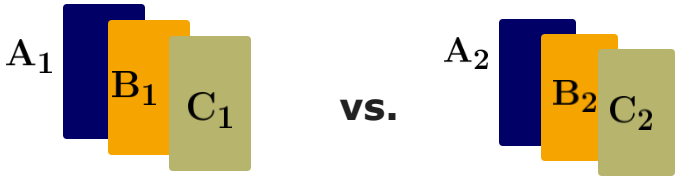
```
Xhat{1} = Fac.U{1}*Fac.U{2}';
Xhat{2} = Fac.U{1}*Fac.U{3}';
norm(Z.object{1}.data-Xhat{1},'fro')
norm(Z.object{2}.data-Xhat{2},'fro')
```

simula

CMF is not unique!

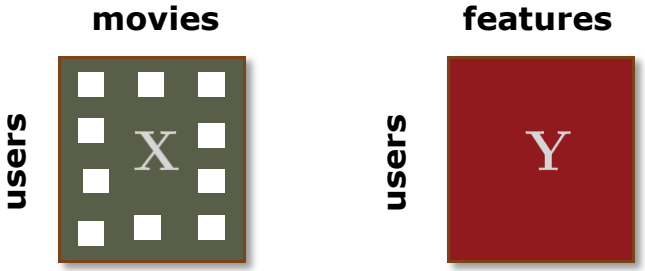
Compare the factor matrices corresponding to the same function value

```
options.Display      ='final';
[Fac1, FacInit1, out1] = cmtf_opt(Z,R,'alg', 'ncg','alg_options',options,'init','random');
[Fac2, FacInit2, out2] = cmtf_opt(Z,R,'alg', 'ncg','alg_options',options,'init','random');
for i=1:3
    corr(Fac1.U{i},Fac2.U{i})
    pause
end
```



simula

CMF extends to joint analysis of incomplete matrices!



$$\mathbf{X} \approx \mathbf{A}\mathbf{B}^\mathsf{T} \qquad \mathbf{Y} \approx \mathbf{A}\mathbf{C}^\mathsf{T}$$

$$\min_{\mathbf{A},\mathbf{B},\mathbf{C}} \left\| \mathbf{W} * (\mathbf{X} - \mathbf{A}\mathbf{B}^\mathsf{T}) \right\|^2 + \left\| \mathbf{Y} - \mathbf{A}\mathbf{C}^\mathsf{T} \right\|^2$$

Objective function:

$$f_{\mathbf{W}}(\mathbf{A},\mathbf{B},\mathbf{C}) = \underbrace{\left\| \mathbf{W} * (\mathbf{X} - \mathbf{A}\mathbf{B}^\mathsf{T}) \right\|^2}_{f_{W_1}} + \underbrace{\left\| \mathbf{Y} - \mathbf{A}\mathbf{C}^\mathsf{T} \right\|^2}_{f_{W_2}}$$

Compute the gradient:

$$\frac{\partial f_{\mathbf{W}}}{\partial \mathbf{A}} = \frac{\partial f_{\mathbf{W}_1}}{\partial \mathbf{A}} + \frac{\partial f_{\mathbf{W}_2}}{\partial \mathbf{A}}$$

$$\frac{\partial f_{\mathbf{W}}}{\partial \mathbf{B}} = \frac{\partial f_{\mathbf{W}_1}}{\partial \mathbf{B}}$$

$$\frac{\partial f_{\mathbf{W}}}{\partial \mathbf{C}} = \frac{\partial f_{\mathbf{W}_2}}{\partial \mathbf{C}}$$

Use a first-order optimization method to fit the CMF model

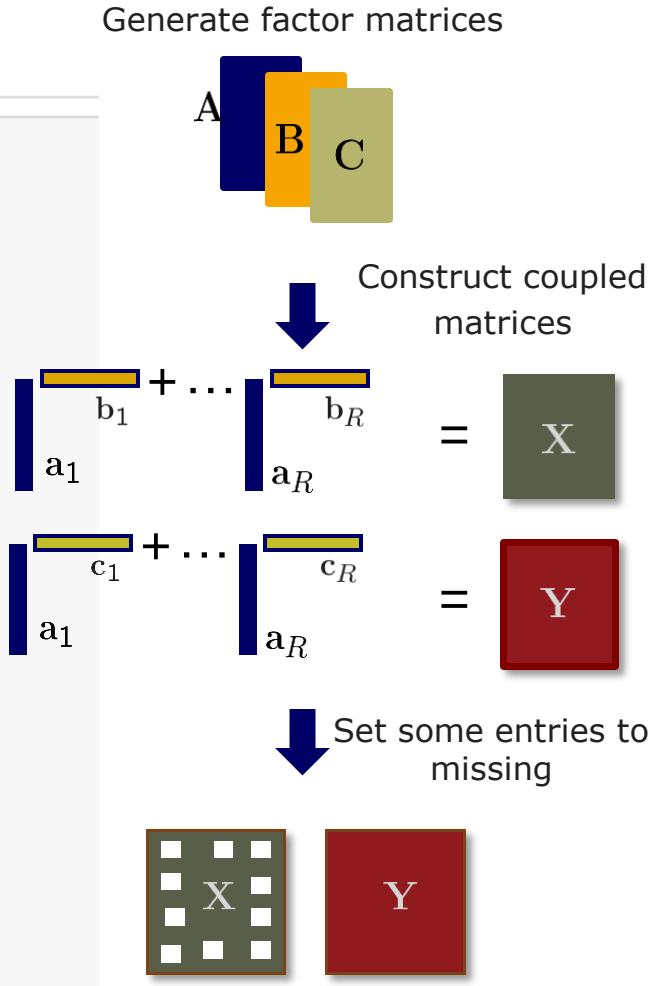
simula

Coupled Matrix Factorizations: Missing Data Estimation

Construct coupled matrices and set some entries to missing in one of them

```
sz    = [10 5 4];
modes = {[1 2],[1 3]};
R  = 2;
for i=1:length(sz)
    A{i}  = randn(sz(i),R);
    for r = 1:R
      A{i}(:,r)= A{i}(:,r)/norm(A{i}(:,r));
    end
end
X{1}  = A{1}*A{2}';
X{2}  = A{1}*A{3}';
Xorig = X;
W     = round(rand(size(X{1})));
X{1}(W==0)= NaN;

X{1}(W==0)= 0;
P = length(X);
for p=1:P
    Z.object{p}  = tensor(X{p});
    obj_norm(p)  = norm(Z.object{p});
    Z.object{p}  = Z.object{p}/obj_norm(p);
end
Z.miss{1} = tensor(W);
Z.miss{2} =[];
Z.modes   = modes;
Z.size    = sz;
```

$$\mathbf{X} = \mathbf{A}\mathbf{B}^\mathsf{T}$$
$$\mathbf{Y} = \mathbf{A}\mathbf{C}^\mathsf{T}$$

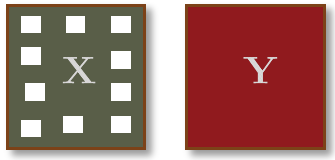


simula

Coupled Matrix Factorizations: Missing Data Estimation (cont.)

Suppose you are given coupled matrices and want to jointly factorize them

```
options = ncg('defaults');
options.MaxFuncEvals = 10000;
options.MaxIters     = 10000;
options.StopTol      = 1e-8;
options.RelFuncTol   = 1e-8;
options.DisplayIters = 50;
[Fac, FacInit, out] = cmtf_opt(Z,R,'alg', 'ncg','alg_options',options,'init','random');
```



Solve

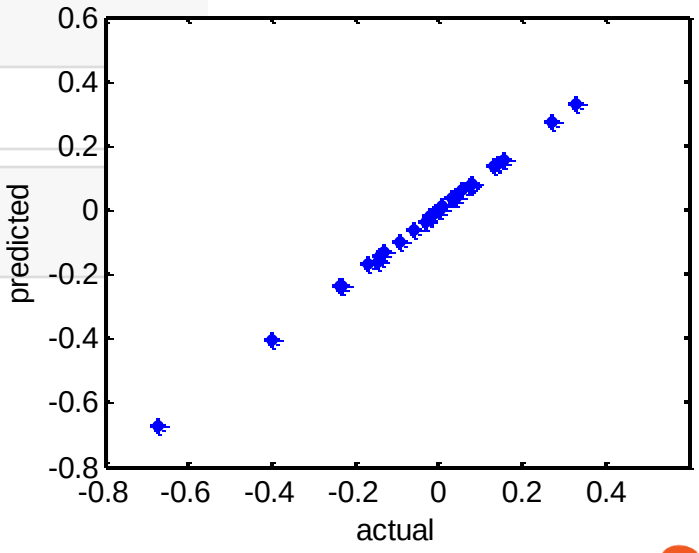
$$\min_{\mathbf{A},\mathbf{B},\mathbf{C}} \left\| \mathbf{W} * (\mathbf{X} - \mathbf{A}\mathbf{B}^{\mathsf{T}}) \right\|^2 + \left\| \mathbf{Y} - \mathbf{A}\mathbf{C}^{\mathsf{T}} \right\|^2$$

Reconstruct matrices and multiple by the norms we scaled the data sets with

```
Xhat{1} = Fac.U{1}*Fac.U{2}';
Xhat{2} = Fac.U{1}*Fac.U{3}';
for i=1:2
    Xhat{i}=Xhat{i}*obj_norm(i);
end
```

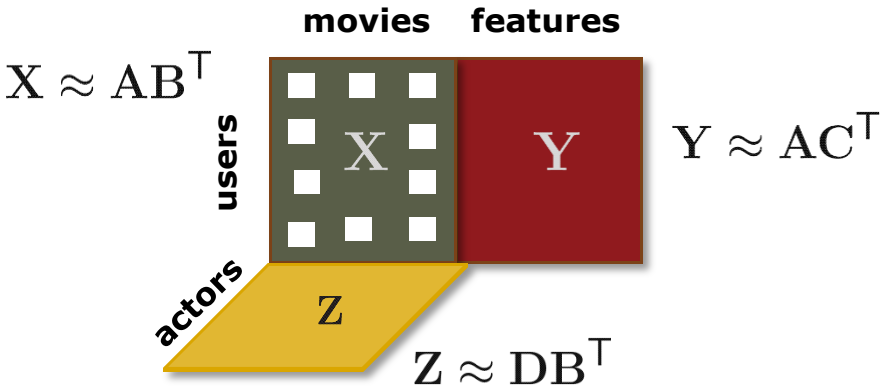
Estimating missing data

```
plot(Xorig{1}(W==0),Xhat{1}(W==0),'*')
ylabel('predicted');xlabel('actual')
```



simula

CMF can jointly analyze multiple matrices coupled in different modes!



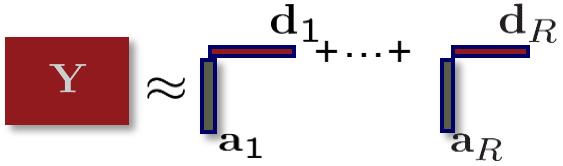
$$\min_{\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D}} \left\| \mathbf{W} * (\mathbf{X} - \mathbf{A}\mathbf{B}^\mathsf{T}) \right\|^2 + \left\| \mathbf{Y} - \mathbf{A}\mathbf{C}^\mathsf{T} \right\|^2 + \left\| \mathbf{Z} - \mathbf{D}\mathbf{B}^\mathsf{T} \right\|^2$$

simula

Data Fusion based on Coupled Matrix and Tensor Factorizations

Joint analysis of heterogeneous data from multiple sources can be formulated as a coupled matrix and tensor factorization (CMTF) problem. In CMTF, higher-order tensors and matrices are simultaneously factorized by fitting a CP model to higher-order tensors and factorizing matrices in a coupled manner.

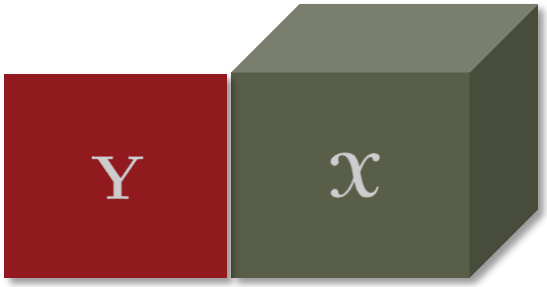
Matrix Factorization:



$$\mathbf{Y} \approx \sum_{r=1}^{R} \mathbf{a}_r \mathbf{d}_r^{\mathsf{T}} \approx \mathbf{A}\mathbf{D}^{\mathsf{T}}$$

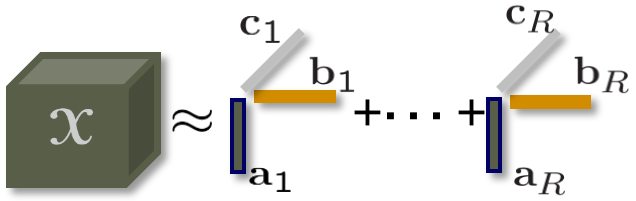
$$\mathbf{A} \in \mathbb{R}^{I \times R} = \begin{bmatrix} \mathbf{a}_1 & \cdots & \mathbf{a}_R \end{bmatrix}$$

$$\mathbf{D} \in \mathbb{R}^{M \times R} = \begin{bmatrix} \mathbf{d}_1 & \cdots & \mathbf{d}_R \end{bmatrix}$$



$$\mathbf{Y} \approx \mathbf{A}\mathbf{D}^{\mathsf{T}} \quad \mathcal{X} \approx [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$$

Tensor Factorization: CP



$$\mathcal{X} \approx \sum_{r=1}^{R} \mathbf{a}_r \circ \mathbf{b}_r \circ \mathbf{c}_r \approx [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$$

$$\mathbf{A} \in \mathbb{R}^{I \times R} = \begin{bmatrix} \mathbf{a}_1 & \cdots & \mathbf{a}_R \end{bmatrix}$$

$$\mathbf{B} \in \mathbb{R}^{J \times R} = \begin{bmatrix} \mathbf{b}_1 & \cdots & \mathbf{b}_R \end{bmatrix}$$

$$\mathbf{C} \in \mathbb{R}^{K \times R} = \begin{bmatrix} \mathbf{c}_1 & \cdots & \mathbf{c}_R \end{bmatrix}$$

The problem can be formulated as:

$$\min_{\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D}} \| \mathcal{X} - [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!] \|^2 + \| \mathbf{Y} - \mathbf{A}\mathbf{D}^{\mathsf{T}} \|^2$$

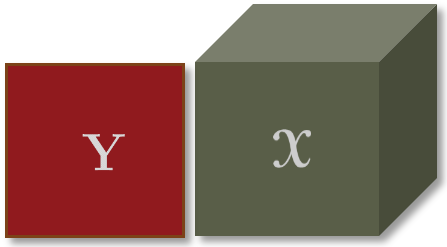
simula

All-at-once Optimization for CMTF

[Acar, Dunlavy, Kolda, 2011]

CMTF-OPT is a gradient–based optimization approach for joint factorization of coupled matrices and higher-order tensors.

$$\min_{\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D}} \| \mathcal{X} - [\![\mathbf{A},\mathbf{B},\mathbf{C}]\!] \|^2 + \| \mathbf{Y} - \mathbf{A}\mathbf{D}^\mathsf{T} \|^2$$



$$\mathbf{Y} \approx \mathbf{A}\mathbf{D}^\mathsf{T} \quad \mathcal{X} \approx [\![\mathbf{A},\mathbf{B},\mathbf{C}]\!]$$

Step 1: Define the objective function

$$f(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D}) = \frac{1}{2} \| \mathcal{X} - [\![\mathbf{A},\mathbf{B},\mathbf{C}]\!] \|^2 + \frac{1}{2} \| \mathbf{Y} - \mathbf{A}\mathbf{D}^\mathsf{T} \|^2$$

Step 2: Compute the gradient

$$\frac{\partial f}{\partial \mathbf{A}} = -\mathbf{X}_{(1)}(\mathbf{C} \odot \mathbf{B}) + \mathbf{A}(\mathbf{C}^\mathsf{T}\mathbf{C} * \mathbf{B}^\mathsf{T}\mathbf{B}) - \mathbf{Y}\mathbf{D} + \mathbf{A}\mathbf{D}^\mathsf{T}\mathbf{D}$$

$$\frac{\partial f}{\partial \mathbf{B}} = -\mathbf{X}_{(2)}(\mathbf{C} \odot \mathbf{A}) + \mathbf{B}(\mathbf{C}^\mathsf{T}\mathbf{C} * \mathbf{A}^\mathsf{T}\mathbf{A})$$

$$\frac{\partial f}{\partial \mathbf{C}} = -\mathbf{X}_{(3)}(\mathbf{B} \odot \mathbf{A}) + \mathbf{C}(\mathbf{B}^\mathsf{T}\mathbf{B} * \mathbf{A}^\mathsf{T}\mathbf{A})$$

$$\frac{\partial f}{\partial \mathbf{D}} = -\mathbf{Y}^\mathsf{T}\mathbf{A} + \mathbf{D}\mathbf{A}^\mathsf{T}\mathbf{A}$$

Vectorize and concatenate the partials

$$\nabla f = \begin{bmatrix} \frac{\partial f}{\partial \mathbf{a}_1} \\ \vdots \\ \frac{\partial f}{\partial \mathbf{a}_R} \\ \frac{\partial f}{\partial \mathbf{b}_1} \\ \vdots \\ \frac{\partial f}{\partial \mathbf{b}_R} \\ \frac{\partial f}{\partial \mathbf{c}_1} \\ \vdots \\ \frac{\partial f}{\partial \mathbf{c}_R} \\ \frac{\partial f}{\partial \mathbf{d}_1} \\ \vdots \\ \frac{\partial f}{\partial \mathbf{d}_R} \end{bmatrix}$$

Step 3: Pick a first-order optimization method

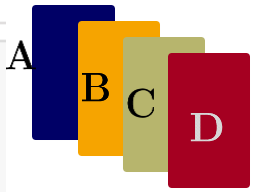
simula

Coupled Matrix and Tensor Factorizations: Example

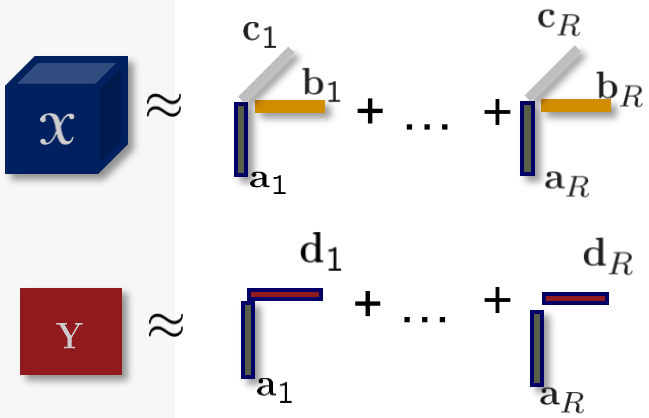
Construct coupled matrices

```
sz    = [30 20 10 15];
modes = {[1 2 3],[1 4]};
R  = 2;
for i=1:length(sz)
    A{i}  = randn(sz(i),R);
    for r = 1:R
      A{i}(:,r)= A{i}(:,r)/norm(A{i}(:,r));
    end
end
X{1} = full(ktensor(A(1:3)));
X{2} = tensor(A{1}*A{4}');
% add noise
for i=1:2
    N{i} = tensor(randn(size(X{i})));
    X{i} = X{i} + 0.3* N{i}/norm(N{i})*norm(X{i});
end

P = length(X);
for p=1:P
    Z.object{p} = X{p};
    Z.object{p} = Z.object{p}/norm(Z.object{p});
end
Z.modes = modes;
Z.size  = sz;
```

Generate factor matrices



Construct coupled data sets + add noise

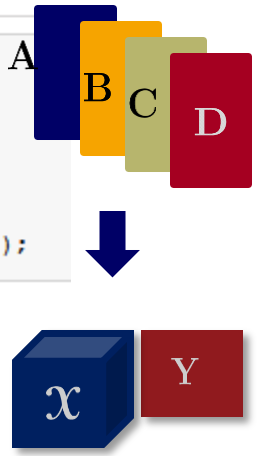


simula

Coupled Matrix and Tensor Factorizations: Example (cont.)

Suppose you are given coupled matrices/tensors and want to jointly factorize them

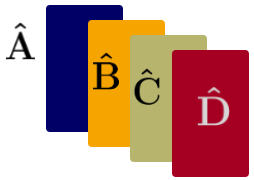
```
options = ncg('defaults');
options.MaxFuncEvals = 10000;
options.MaxIters     = 10000;
options.StopTol      = 1e-8;
options.RelFuncTol   = 1e-8;
[Fac, FacInit, out] = cmtf_opt(Z,R,'alg', 'ncg','alg_options',options,'init','nvecs');
```



Original factor matrices

Minimize *f* using CMTF-OPT

$$f(\hat{A},\hat{B},\hat{C},\hat{D}) = \left\| \mathcal{X} - [\![\hat{A},\hat{B},\hat{C}]\!] \right\|^2 + \left\| Y - \hat{A}\hat{D}^T \right\|^2$$



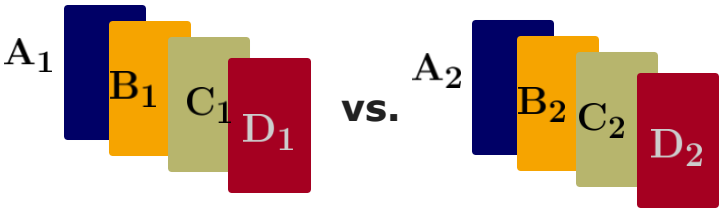
Compare the extracted factor matrices with the original ones

```
Fac = normalize(Fac);
for i=1:4
    corr(Fac.U{i},A{i})
end
```

simula

CMTF inherits uniqueness from CP!

[Sorensen and De Lathauwer, 2015]



Compare the factor matrices corresponding to the same function value

```
options.Display ='final';
[Fac1, FacInit1, out1] = cmtf_opt(Z,R,'alg', 'ncg','alg_options',options,'init','random');
[Fac2, FacInit2, out2] = cmtf_opt(Z,R,'alg', 'ncg','alg_options',options,'init','random');
for i=1:4
    corr(Fac1.U{i},Fac2.U{i})
end
```

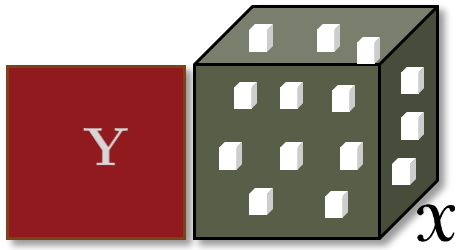
simula

CMTF easily extends to incomplete data sets

We fit the model only to the known data entries and ignore the missing entries (in higher-order tensors and/or matrices).

$$\min_{\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D}} \| \mathcal{W} * (\mathcal{X} - [\![\mathbf{A},\mathbf{B},\mathbf{C}]\!]) \|^2 + \left\| \mathbf{Y} - \mathbf{A}\mathbf{D}^\mathsf{T} \right\|^2$$

$$w_{ijk} = \begin{cases} 1 & \text{if } x_{ijk} \text{ is known,} \\ 0 & \text{if } x_{ijk} \text{ is missing.} \end{cases}$$



$$\mathbf{Y} \approx \mathbf{A}\mathbf{D}^\mathsf{T} \quad \mathcal{X} \approx [\![\mathbf{A},\mathbf{B},\mathbf{C}]\!]$$

Our objective:

$$f_{\mathcal{W}}(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D}) = \frac{1}{2} \| \mathcal{W} * (\mathcal{X} - [\![\mathbf{A},\mathbf{B},\mathbf{C}]\!]) \|^2 + \frac{1}{2} \left\| \mathbf{Y} - \mathbf{A}\mathbf{D}^\mathsf{T} \right\|^2$$

Gradient: Let $\mathcal{Z} = [\![\mathbf{A},\mathbf{B},\mathbf{C}]\!]$

$$\frac{\partial f_{\mathcal{W}}}{\partial \mathbf{A}} = (\mathbf{W}_{(1)} * \mathbf{Z}_{(1)} - \mathbf{W}_{(1)} * \mathbf{X}_{(1)})(\mathbf{C} \odot \mathbf{B}) - \mathbf{Y}\mathbf{D} + \mathbf{A}\mathbf{D}^\mathsf{T}\mathbf{D}$$

$$\frac{\partial f_{\mathcal{W}}}{\partial \mathbf{B}} = (\mathbf{W}_{(2)} * \mathbf{Z}_{(2)} - \mathbf{W}_{(2)} * \mathbf{X}_{(2)})(\mathbf{C} \odot \mathbf{A})$$

$$\frac{\partial f_{\mathcal{W}}}{\partial \mathbf{C}} = (\mathbf{W}_{(3)} * \mathbf{Z}_{(3)} - \mathbf{W}_{(3)} * \mathbf{X}_{(3)})(\mathbf{B} \odot \mathbf{A})$$

$$\frac{\partial f_{\mathcal{W}}}{\partial \mathbf{D}} = -\mathbf{Y}^\mathsf{T}\mathbf{A} + \mathbf{D}\mathbf{A}^\mathsf{T}\mathbf{A}$$

Vectorize and concatenate the partials

$$\nabla f_{\mathcal{W}} = \begin{bmatrix} \frac{\partial f_{\mathcal{W}}}{\partial \mathbf{a}_1} \\ \vdots \\ \frac{\partial f_{\mathcal{W}}}{\partial \mathbf{a}_R} \\ \frac{\partial f_{\mathcal{W}}}{\partial \mathbf{b}_1} \\ \vdots \\ \frac{\partial f_{\mathcal{W}}}{\partial \mathbf{b}_R} \\ \frac{\partial f_{\mathcal{W}}}{\partial \mathbf{c}_1} \\ \vdots \\ \frac{\partial f_{\mathcal{W}}}{\partial \mathbf{c}_R} \\ \frac{\partial f_{\mathcal{W}}}{\partial \mathbf{d}_1} \\ \vdots \\ \frac{\partial f_{\mathcal{W}}}{\partial \mathbf{d}_R} \end{bmatrix} \quad \cdots$$

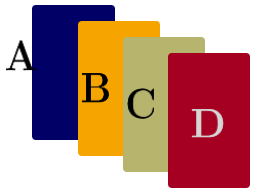
simula

CMTF: Missing Data Estimation

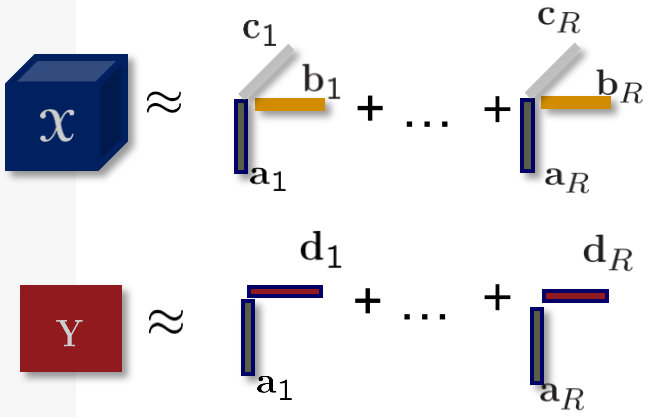
Construct coupled data sets and set some entries to missing in one of them

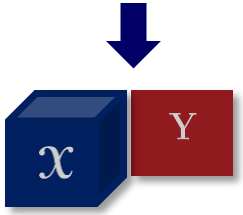
```
sz    = [30 20 10 15];
modes = {[1 2 3],[1 4]};
R  = 2;
for i=1:length(sz)
    A{i}  = randn(sz(i),R);
    for r = 1:R
      A{i}(:,r)= A{i}(:,r)/norm(A{i}(:,r));
    end
end
X{1} = full(ktensor(A(1:3)));
X{2} = tensor(A{1}*A{4}');
% add noise
for i=1:2
    N{i} = tensor(randn(size(X{i})));
    X{i} = X{i} + 0.1* N{i}/norm(N{i})*norm(X{i});
end
Xorig = X;
W     = round(rand(size(X{1})));
X{1}(find(W==0))= NaN;

X{1}(find(W==0))= 0;
P = length(X);
for p=1:P
    Z.object{p}  = tensor(X{p});
    obj_norms(p) =  norm(Z.object{p});
    Z.object{p}  = Z.object{p}/obj_norms(p);
end
Z.miss{1} = tensor(W);
Z.miss{2} =[];
Z.modes   = modes;
Z.size    = sz;
```

Generate factor matrices

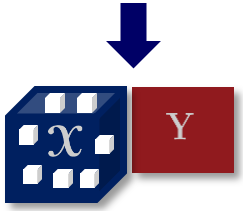


Construct coupled data sets + add noise





Set some entries to missing

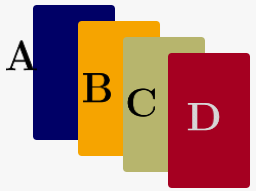


simula

CMTF: Missing Data Estimation (cont.)

Suppose you are given the incomplete tensor coupled with the matrix and want to jointly factorize them

```
options = ncg('defaults');
options.MaxFuncEvals = 10000;
options.MaxIters     = 10000;
options.StopTol      = 1e-8;
options.RelFuncTol   = 1e-8;
[Fac, FacInit, out] = cmtf_opt(Z,R,'alg', 'ncg','alg_options',options,'init','random');
```



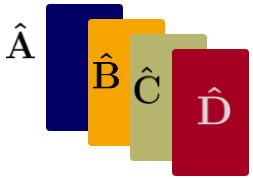
Original factor matrices

Minimize *f* using CMTF-OPT

$$f(\hat{A},\hat{B},\hat{C},\hat{D}) = \left\| \mathcal{W} * (\mathcal{X} - [\![\hat{A},\hat{B},\hat{C}]\!]) \right\|^2 + \left\| Y - \hat{A}\hat{D}^T \right\|^2$$

Compare the estimated missing entries with the original values

```
Xhat{1} = obj_norms(1)* full(ktensor(Fac.U(1:3)));
plot(Xorig{1}(find(W==0)),Xhat{1}(find(W==0)),'*')
ylabel('predicted');xlabel('actual')
```



Compare the extracted factor matrices with the original ones

```
Fac = normalize(Fac);
for i=1:4
    corr(Fac.U{i},A{i})
end
```

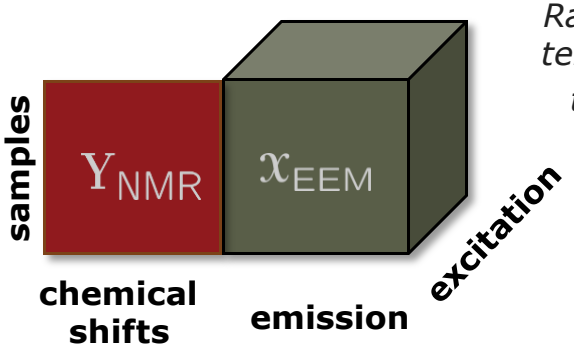
Reconstruct the tensor

$$\hat{\mathcal{X}} = [\![\hat{A},\hat{B},\hat{C}]\!]$$

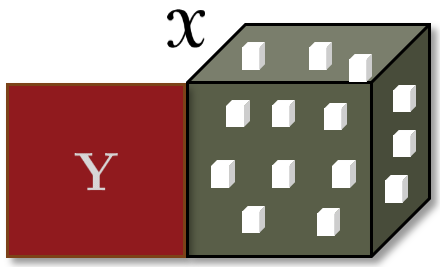
Application$_6$ (Metabolomics): Data fusion can improve missing data estimation performance

[Acar, Rasmussen, Savorani, Næs, Bro, 2013]

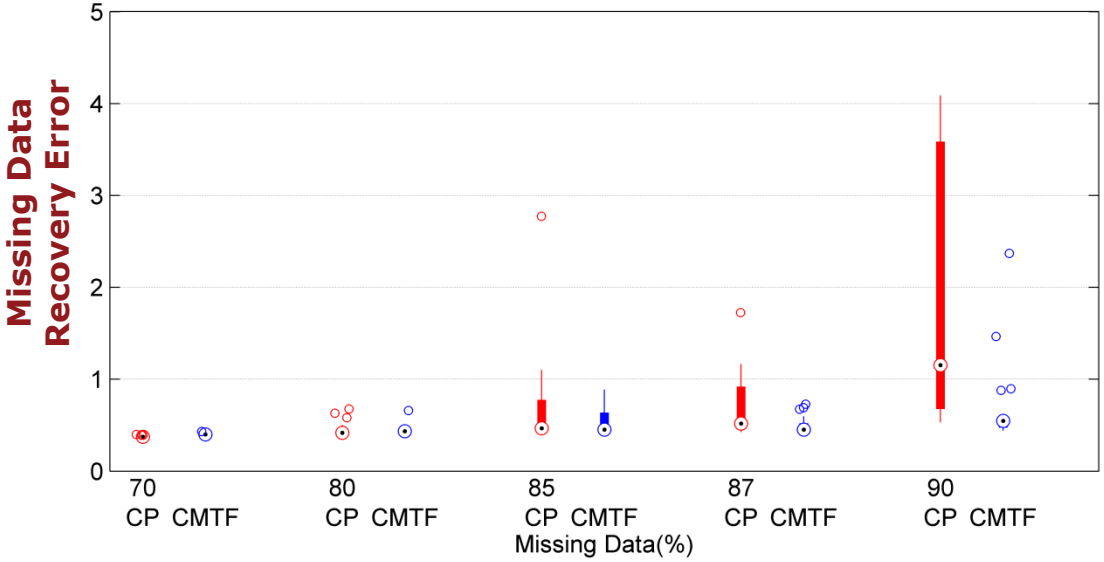
We have plasma samples measured using different analytical techniques, i.e., NMR and Fluorescence Spectroscopy.



Randomly set tensor entries to missing



$$\| \mathcal{W} * (\mathcal{X} - [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]) \|^2 + \left\| \mathbf{Y} - \mathbf{A}\mathbf{D}^{\mathsf{T}} \right\|^2$$

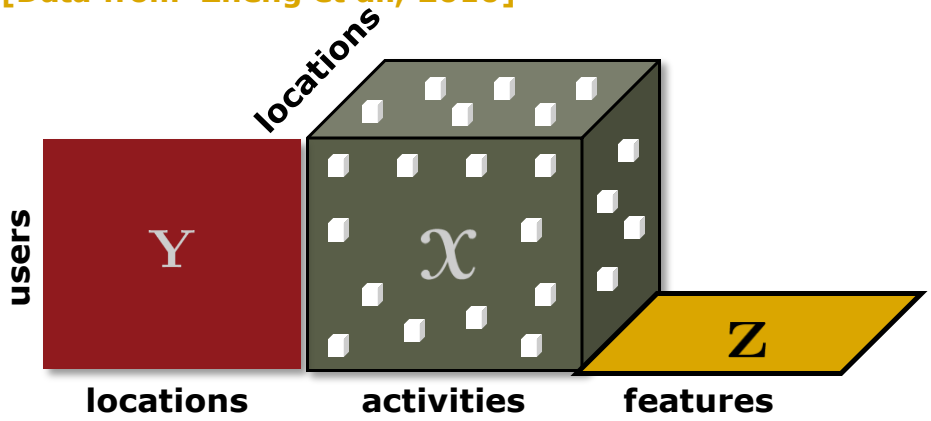


$$\text{Missing Data Recovery Error} = \frac{\left\| (1 - \mathcal{W}) * (\mathcal{X} - \hat{\mathcal{X}}) \right\|}{\| (1 - \mathcal{W}) * \mathcal{X} \|}$$

Missing data recovery error is lower using the coupled approach at high amounts of missing data.

Application$_7$ (Activity Recommendation): Data fusion can improve missing data estimation performance

[Data from Zheng et al., 2010]

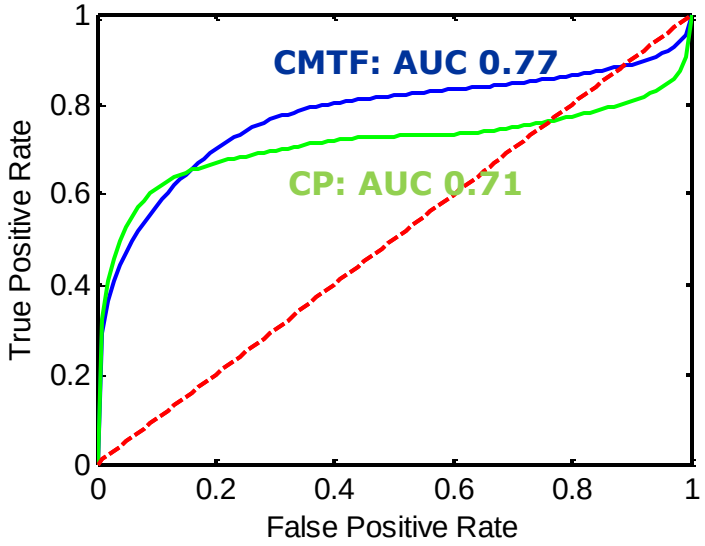


$$x_{ijk} = \begin{cases} 1 & \text{if user i performs activity j at location k,} \\ 0 & \text{otherwise.} \end{cases}$$

For the missing entries:

Original values x_{ijk} Estimated values $\hat{x}_{ijk}$

Average ROC curve for 80% missing data



CP :

$$\min_{\mathbf{A},\mathbf{B},\mathbf{C}} \| \mathcal{W} * (\mathcal{X} - [\![\mathbf{A},\mathbf{B},\mathbf{C}]\!]) \|^2$$

$$\hat{\mathcal{X}} = [\![\mathbf{A},\mathbf{B},\mathbf{C}]\!]$$

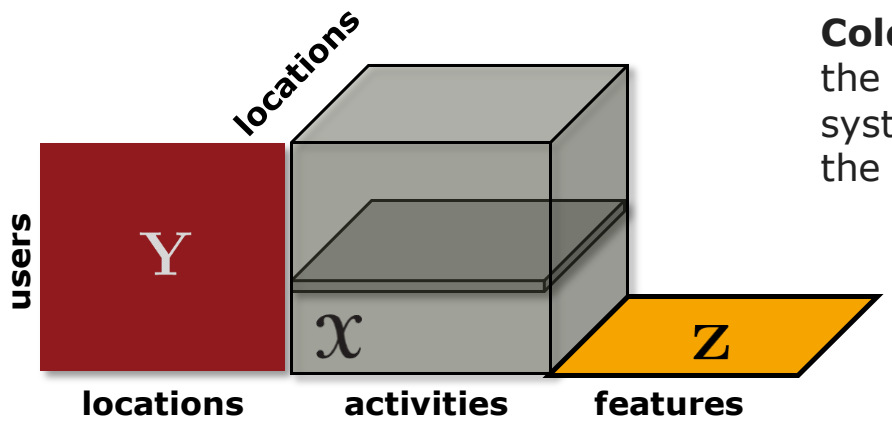
CMTF:

$$\min_{\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D},\mathbf{E}} \| \mathcal{W} * (\mathcal{X} - [\![\mathbf{A},\mathbf{B},\mathbf{C}]\!]) \|^2 + \left\| \mathbf{Y} - \mathbf{A}\mathbf{D}^{\mathsf{T}} \right\|^2 + \left\| \mathbf{Z} - \mathbf{C}\mathbf{E}^{\mathsf{T}} \right\|^2$$

$$\hat{\mathcal{X}} = [\![\mathbf{A},\mathbf{B},\mathbf{C}]\!]$$

simula

Data fusion can also handle structured missing data effectively



Cold-start problem: When a new user starts using the application, i.e., location-activity recommender system, this will correspond to a missing slice for the new user.

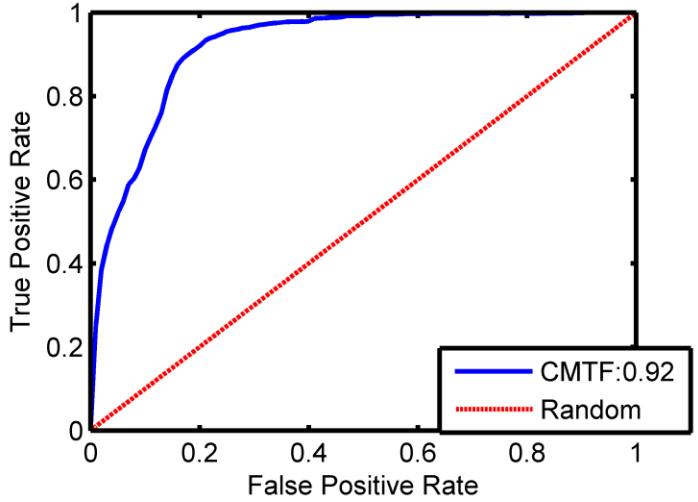
For the missing slice i (for i=1,2,…,*I*):

Original values	Estimated values using CMTF
$\text{vec}(\mathbf{X}_i)$	$\text{vec}(\hat{\mathbf{X}}_i)$

$$x_{ijk} = \begin{cases} 1 & \text{if user i performs activity j at location k,} \\ 0 & \text{otherwise.} \end{cases}$$

We cannot use low-rank approximation of a tensor to fill in the missing slice. However, we can make use of additional sources of information through the coupled model.

Average ROC curve for *I*=146 users



$$\min_{\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D},\mathbf{E}} \| \mathcal{W} * (\mathcal{X} - [\![\mathbf{A},\mathbf{B},\mathbf{C}]\!]) \|^2 + \left\| \mathbf{Y} - \mathbf{A}\mathbf{D}^{\mathsf{T}} \right\|^2 + \left\| \mathbf{Z} - \mathbf{C}\mathbf{E}^{\mathsf{T}} \right\|^2$$

For coupled analysis using different tensor models and loss functions, see **[Ermis, Acar, Cemgil, 2015]**

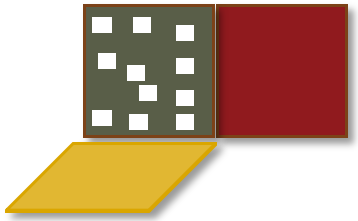
Summary

Coupled Matrix Factorizations

How to jointly factorize matrices using coupled matrix factorizations

How to handle and estimate missing entries using CMF

Algorithms: Gradient-based optimization algorithms



Coupled Matrix and Tensor Factorizations

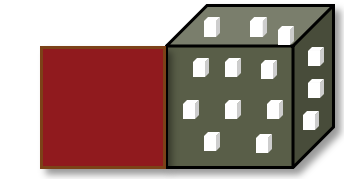
How to jointly factorize matrices and higher-order tensors

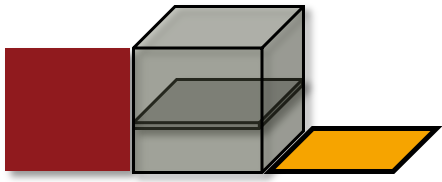
How to jointly analyze data sets with missing entries and estimate missing data, with applications from:

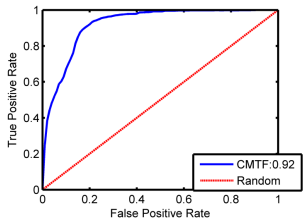
- **Metabolomics**
- **Recommender systems**

CMTF enables us to handle and estimate structured missing data

Algorithms: Gradient-based optimization algorithms







MATLAB functions

cmtf-opt: Fitting CMF and CMTF to coupled data sets **(from CMTF Toolbox)**

[Fac, FacInit, output] = cmtf_opt(X, R, ‘init’, ‘random’, ‘alg’,’ncg’, ‘alg_options’, options);

Model: Optimization problem

$$\min_{\mathbf{A},\mathbf{B},\mathbf{C}} \left\| \mathbf{X} - \mathbf{A}\mathbf{B}^{\mathsf{T}} \right\|^2 + \left\| \mathbf{Y} - \mathbf{A}\mathbf{C}^{\mathsf{T}} \right\|^2 \qquad \min_{\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D}} \| \mathcal{X} - [\![\mathbf{A},\mathbf{B},\mathbf{C}]\!] \|^2 + \| \mathbf{Y} - \mathbf{A}\mathbf{D}^{\mathsf{T}} \|^2$$

$$\min_{\mathbf{A},\mathbf{B},\mathbf{C}} \left\| \mathbf{W} * (\mathbf{X} - \mathbf{A}\mathbf{B}^{\mathsf{T}}) \right\|^2 + \left\| \mathbf{Y} - \mathbf{A}\mathbf{C}^{\mathsf{T}} \right\|^2 \qquad \min_{\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D}} \| \mathcal{W} * (\mathcal{X} - [\![\mathbf{A},\mathbf{B},\mathbf{C}]\!]) \|^2 + \left\| \mathbf{Y} - \mathbf{A}\mathbf{D}^{\mathsf{T}} \right\|^2$$

Tensor & CMTF Toolbox
Computation of the function value and the gradient

$$f(\mathbf{A},\mathbf{B},\mathbf{C}) = \frac{1}{2}\left\| \mathbf{X} - \mathbf{A}\mathbf{B}^{\mathsf{T}} \right\|^2 + \frac{1}{2}\left\| \mathbf{Y} - \mathbf{A}\mathbf{C}^{\mathsf{T}} \right\|^2 \qquad \nabla f = \begin{bmatrix} \frac{\partial f}{\partial \mathbf{a}_1} \\ \vdots \\ \frac{\partial f}{\partial \mathbf{a}_R} \\ \frac{\partial f}{\partial \mathbf{b}_1} \\ \vdots \\ \frac{\partial f}{\partial \mathbf{b}_R} \\ \frac{\partial f}{\partial \mathbf{c}_1} \\ \vdots \\ \frac{\partial f}{\partial \mathbf{c}_R} \end{bmatrix}$$

$$f_{\mathbf{W}}(\mathbf{A},\mathbf{B},\mathbf{C}) = \frac{1}{2}\left\| \mathbf{W} * (\mathbf{X} - \mathbf{A}\mathbf{B}^{\mathsf{T}}) \right\|^2 + \frac{1}{2}\left\| \mathbf{Y} - \mathbf{A}\mathbf{C}^{\mathsf{T}} \right\|^2$$

$$f(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D}) = \frac{1}{2}\| \mathcal{X} - [\![\mathbf{A},\mathbf{B},\mathbf{C}]\!] \|^2 + \frac{1}{2}\| \mathbf{Y} - \mathbf{A}\mathbf{D}^{\mathsf{T}} \|^2 \qquad \nabla f = \begin{bmatrix} \frac{\partial f}{\partial \mathbf{a}_1} \\ \vdots \\ \frac{\partial f}{\partial \mathbf{a}_R} \\ \frac{\partial f}{\partial \mathbf{b}_1} \\ \vdots \\ \frac{\partial f}{\partial \mathbf{b}_R} \\ \frac{\partial f}{\partial \mathbf{c}_1} \\ \vdots \\ \frac{\partial f}{\partial \mathbf{c}_R} \\ \frac{\partial f}{\partial \mathbf{d}_1} \\ \vdots \\ \frac{\partial f}{\partial \mathbf{d}_R} \end{bmatrix}$$

$$f_{\mathcal{W}}(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D}) = \frac{1}{2}\| \mathcal{W} * (\mathcal{X} - [\![\mathbf{A},\mathbf{B},\mathbf{C}]\!]) \|^2 + \frac{1}{2}\left\| \mathbf{Y} - \mathbf{A}\mathbf{D}^{\mathsf{T}} \right\|^2$$

cmtf_opt

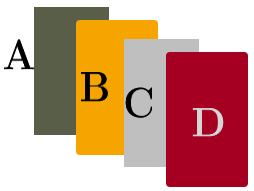
Poblano Toolbox
Gradient-based optimization algorithms

ncg, lbfgs, …

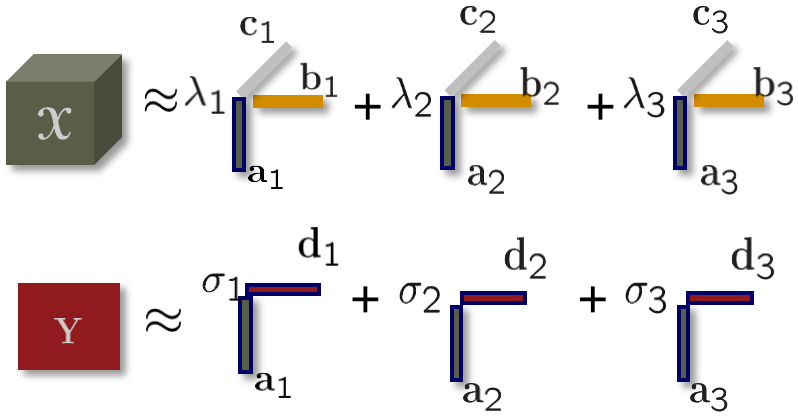
Limitation: CMTF assumes that all factors are shared!

In real applications, coupled data sets often have both **shared** and **unshared** factors. However, the CMTF formulation focuses on modeling only shared factors and fails in the presence of both shared/unshared factors.

Generate factor matrices

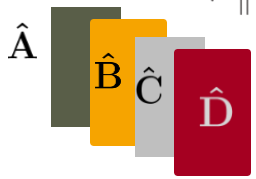


Construct coupled data sets



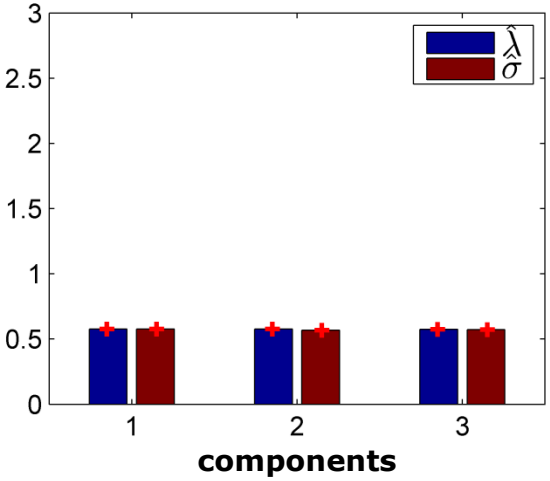
Minimize *f* using CMTF-OPT

$$f(\hat{\mathbf{A}}, \hat{\mathbf{B}}, \hat{\mathbf{C}}, \hat{\mathbf{D}}) = \left\| \mathcal{X} - [\![\hat{\mathbf{A}}, \hat{\mathbf{B}}, \hat{\mathbf{C}}]\!] \right\|^2 + \left\| \mathbf{Y} - \hat{\mathbf{A}} \hat{\mathbf{D}}^{\mathsf{T}} \right\|^2$$



Example 1: All components are shared $\lambda_1 = \lambda_2 = \lambda_3 > 0, \sigma_1 = \sigma_2 = \sigma_3 > 0$

$\hat{\lambda}_i, \hat{\sigma}_i$:



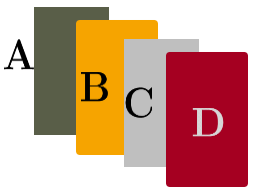
True structure captured!

simula

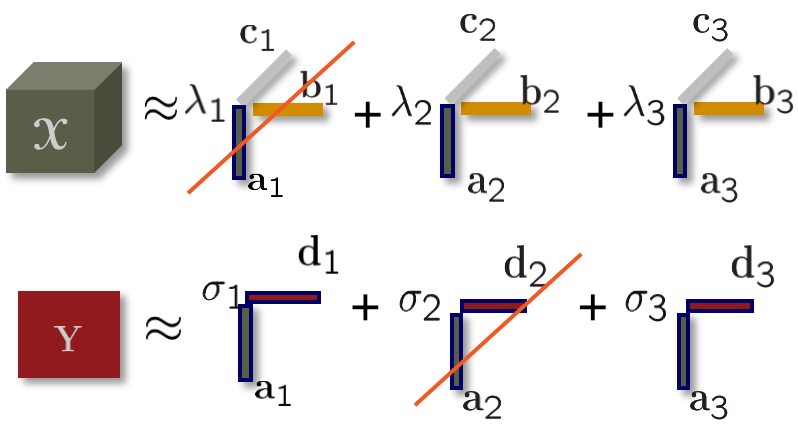
Limitation: CMTF fails to identify shared/unshared factors!

In real applications, coupled data sets often have both **shared** and **unshared** factors. However, the CMTF formulation focuses on modeling only shared factors and fails in the presence of both shared/unshared factors.

Generate factor matrices

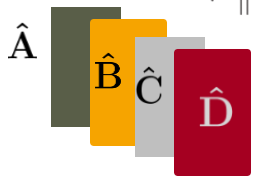


Construct coupled data sets



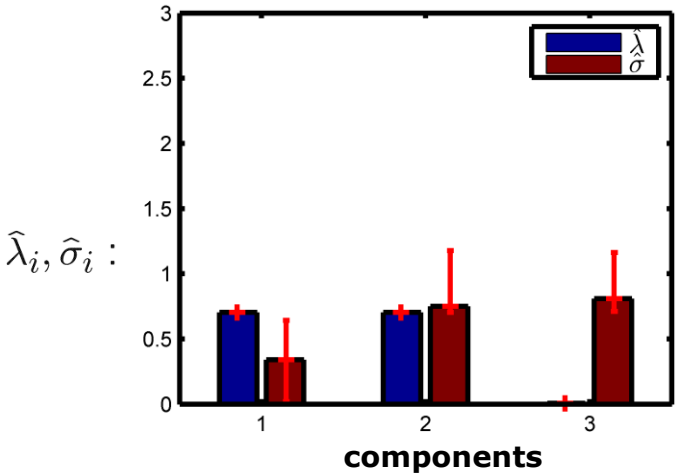
Minimize *f* using CMTF-OPT

$$f(\hat{\mathbf{A}}, \hat{\mathbf{B}}, \hat{\mathbf{C}}, \hat{\mathbf{D}}) = \left\| \mathcal{X} - [\![\hat{\mathbf{A}}, \hat{\mathbf{B}}, \hat{\mathbf{C}}]\!] \right\|^2 + \left\| \mathbf{Y} - \hat{\mathbf{A}}\hat{\mathbf{D}}^{\mathsf{T}} \right\|^2$$



Example 2: One shared and one unshared component in each data set

$$[\lambda_1\ \lambda_2\ \lambda_3] = [0\ 1\ 1]$$
$$[\sigma_1\ \sigma_2\ \sigma_3] = [1\ 0\ 1]$$



Fails to identify shared and unshared components!

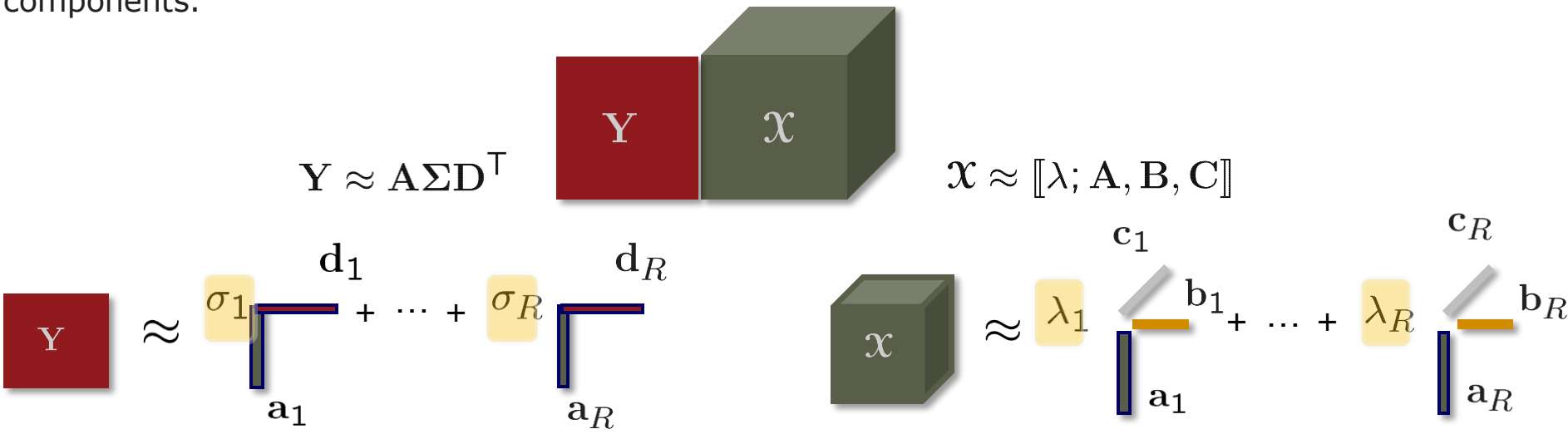
simula

ACMTF (Advanced CMTF): Structure-Revealing CMTF

[Acar et al., *BMC Bioinformatics*, 2014]

We reformulate the coupled matrix and tensor factorization problem by having factor matrices with unit norm columns and explicitly representing the weights of rank-one components in the formulation. Through modeling constraints/penalties, we let the model identify shared/unshared components.

$$\mathbf{Y} \approx \mathbf{A}\boldsymbol{\Sigma}\mathbf{D}^{\mathsf{T}} \qquad \mathcal{Y} \approx [\![\lambda; \mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$$



Structure-revealing model:

$$\min_{\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D},\boldsymbol{\Sigma},\lambda} \| \mathcal{X} - [\![\lambda; \mathbf{A}, \mathbf{B}, \mathbf{C}]\!] \|^2 + \| \mathbf{Y} - \mathbf{A}\boldsymbol{\Sigma}\mathbf{D}^{\mathsf{T}} \|^2 + \beta \| \lambda \|_1 + \beta \| \sigma \|_1$$

$$\text{s.t.} \; \| \mathbf{a}_r \|_2 = \| \mathbf{b}_r \|_2 = \| \mathbf{c}_r \|_2 = \| \mathbf{d}_r \|_2 = 1, \text{for } r = 1, \ldots, R.$$

l1-norm

$$\| \mathbf{x} \|_1 = \sum |x_i|$$

Original CMTF

$$\min_{\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D}} \| \mathcal{X} - [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!] \|^2 + \| \mathbf{Y} - \mathbf{A}\mathbf{D}^{\mathsf{T}} \|^2$$

simula

ACMTF: Unconstrained Optimization

Optimization Problem:

$$\min_{\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D},\Sigma,\lambda} \| \mathcal{X} - [\![\lambda; \mathbf{A}, \mathbf{B}, \mathbf{C}]\!] \|^2 + \| \mathbf{Y} - \mathbf{A}\Sigma\mathbf{D}^\mathsf{T} \|^2 + \beta \| \lambda \|_1 + \beta \| \sigma \|_1$$

$$\texttt{s.t.} \; \| \mathbf{a}_r \|_2 = \| \mathbf{b}_r \|_2 = \| \mathbf{c}_r \|_2 = \| \mathbf{d}_r \|_2 = 1, \text{ for } r = 1, \ldots, R.$$

Define the objective function:

Add as quadratic penalty terms

$$f(\lambda, \Sigma, \mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}) = \| \mathcal{X} - [\![\lambda; \mathbf{A}, \mathbf{B}, \mathbf{C}]\!] \|^2 + \| \mathbf{Y} - \mathbf{A}\Sigma\mathbf{D}^\mathsf{T} \|^2 + \beta \| \lambda \|_1 + \beta \| \sigma \|_1 + \ldots$$

Replace sparsity penalties with differentiable approximations

Smooth Approximation:

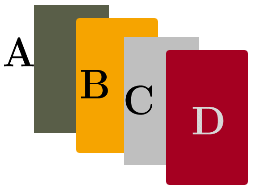
$$f(\lambda, \Sigma, \mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}) = \| \mathcal{X} - [\![\lambda; \mathbf{A}, \mathbf{B}, \mathbf{C}]\!] \|^2 + \| \mathbf{Y} - \mathbf{A}\Sigma\mathbf{D}^\mathsf{T} \|^2 + \beta \sum_{r=1}^{R} \sqrt{\lambda_r^2 + \epsilon} + \beta \sum_{r=1}^{R} \sqrt{\sigma_r^2 + \epsilon} + \ldots$$

Compute the gradient and pick a first order optimization method

simula

Sparsity penalties enable us to capture the true structure

Generate factor matrices



Construct coupled data sets

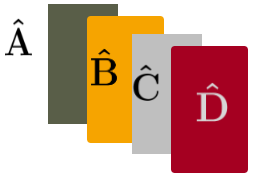
$$\mathcal{X} \approx \lambda_1\, a_1 \circ b_1 \circ c_1 + \lambda_2\, a_2 \circ b_2 \circ c_2 + \lambda_3\, a_3 \circ b_3 \circ c_3$$

$$Y \approx \sigma_1\, a_1 \circ d_1 + \sigma_2\, a_2 \circ d_2 + \sigma_3\, a_3 \circ d_3$$

Minimize

CMTF: $f(\hat{A}, \hat{B}, \hat{C}, \hat{D})$

ACMTF: $f(\hat{\lambda}, \hat{\Sigma}, \hat{A}, \hat{B}, \hat{C}, \hat{D})$

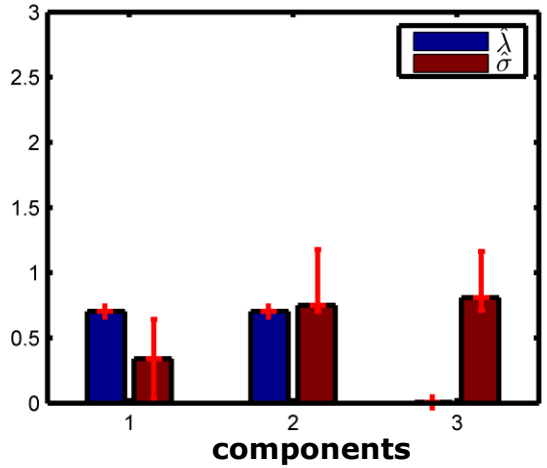


One shared and one unshared component in each data set:

$$[\lambda_1\ \lambda_2\ \lambda_3] = [0\ 1\ 1]$$
$$[\sigma_1\ \sigma_2\ \sigma_3] = [1\ 0\ 1]$$

CMTF

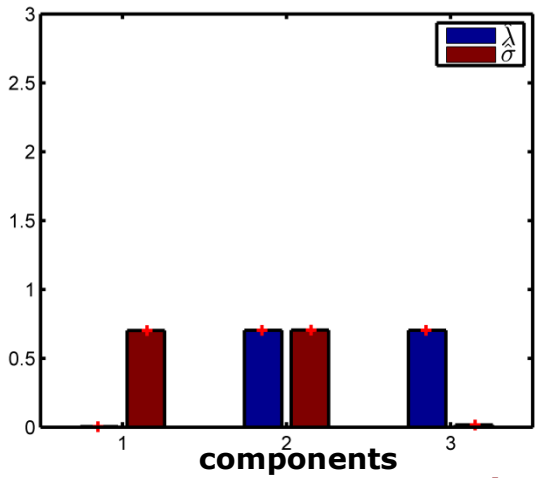
$\hat{\lambda}_i, \hat{\sigma}_i$:



Fails to identify shared/unshared components!

ACMTF

$\hat{\lambda}_i, \hat{\sigma}_i$:



True structure captured!

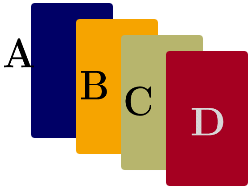
simula

ACMTF: Example

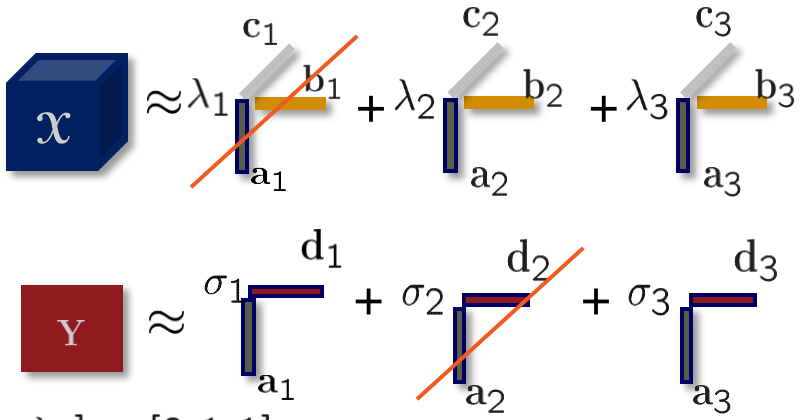
Construct coupled data sets with shared/unshared factors

```
sz    = [30 20 10 15];
modes = {[1 2 3],[1 4]};
R  = 3;
for i=1:length(sz)
    A{i}  = randn(sz(i),R);
    for r = 1:R
      A{i}(:,r)= A{i}(:,r)/norm(A{i}(:,r));
    end
end
lambda(1,:) = [1 0 1];
lambda(2,:) = [1 1 0];
X{1} = full(ktensor(lambda(1,:)',A(1:3)));
X{2} = tensor(A{1}*diag(lambda(2,:))*A{4}');

% add noise
for i=1:2
     N{i} = tensor(randn(size(X{i})));
     X{i} = X{i} + 0.1* N{i}/norm(N{i})*norm(X{i});
 end

P = length(X);
for p=1:P
    Z.object{p}  = tensor(X{p});
    obj_norms(p) = norm(Z.object{p});
    Z.object{p}  = Z.object{p}/obj_norms(p);
end
Z.modes = modes;
Z.size  = sz;
```

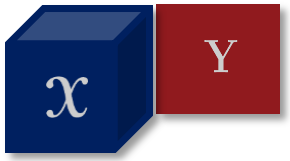
Generate factor matrices



Construct coupled data sets



$$[\lambda_1\ \lambda_2\ \lambda_3] = [0\ 1\ 1]$$
$$[\sigma_1\ \sigma_2\ \sigma_3] = [1\ 0\ 1]$$



simula

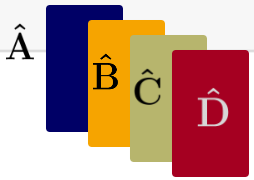
ACMTF: Example (cont.)

Fit ACMTF using one of the first-order optimization algorithms

```
options = ncg('defaults');
options.MaxFuncEvals = 10000;
options.MaxIters     = 10000;
options.StopTol      = 1e-8;
options.RelFuncTol   = 1e-8;
options.DisplayIters = 50;

% fit ACMTF-OPT
[Fac, FacInit, out] = acmtf_opt(Z,R,'alg', 'ncg','alg_options',options,'init','random', 'beta_cp',1e-3, 'beta_pca', 1e-3);

% check weights for shared/unshared factors
lrec = zeros(P,R);
for p =1:P
    temp        = normalize(Fac{p});
    lrec(p,:)   = temp.lambda;
end
%  lrec(1,:)*obj_norms(1) will adjust the weights and correspond to
%  original weights. Similarly, compute lrec(2,:)*obj_norms(2) to get
%  weights comparable with lambda(2,:)

% compare the factors with the original factor matrices
for i=1:3
    corr(Fac{1}.U{i},A{i})
end
corr(Fac{2}.U{2},A{4})
```

ACMTF:

$$f(\hat{\lambda}, \hat{\Sigma}, \hat{\mathrm{A}}, \hat{\mathrm{B}}, \hat{\mathrm{C}}, \hat{\mathrm{D}})$$



simula

CMF models that focus on identification of shared/unshared factors

GSVD (Generalized Singular Value Decomposition) [Van Loan, 1976; Paige & Saunders, 1981] **and Adapted GSVD** [Van Deun et al., 2012]

Given $\mathbf{X} \in \mathbb{R}^{M \times N}$ $(M \geq N)$, $\mathbf{Y} \in \mathbb{R}^{P \times N}$ $(P \geq N)$, let $\mathbf{Z} = [\mathbf{X}^\mathsf{T}\ \mathbf{Y}^\mathsf{T}]^\mathsf{T}$, $R = \text{rank}(\mathbf{Z})$

diagonal

$$\mathbf{X} = \mathbf{U}\mathbf{C}\mathbf{W}$$
$$\mathbf{Y} = \mathbf{V}\mathbf{S}\mathbf{W}$$

orthogonal

$$\mathbf{C} = \text{diag}(c_1, \ldots c_N) \in \mathbb{R}^{M \times N}$$
$$\mathbf{S} = \text{diag}(s_1, \ldots s_N) \in \mathbb{R}^{P \times N}$$

$$c_i^2 + s_i^2 = 1$$

the relation used to identify shared/unshared components in genome-scale expression data sets [Alter et al., 2003]

JIVE (Joint and Individual Variation Explained) [Lock et al., 2013]

Given $\mathbf{X}^{(1)}, \mathbf{X}^{(2)}, \ldots, \mathbf{X}^{(M)} \in \mathbb{R}^{I_m \times N}$, for *m=1, 2, …, M.*

joint individual

$$\mathbf{X}^{(1)} = \mathbf{U}^{(1)}\mathbf{S} + \mathbf{W}^{(1)}\mathbf{S}^{(1)} + \mathbf{R}^{(1)}$$
$$\vdots$$
$$\mathbf{X}^{(M)} = \mathbf{U}^{(M)}\mathbf{S} + \mathbf{W}^{(M)}\mathbf{S}^{(M)} + \mathbf{R}^{(M)},$$

Assumes that joint and individual parts are orthogonal!

BIBFA (Bayesian Interbattery Factor Analysis) [Klami et al., 2013]

Given $\mathbf{X}^{(m)}$, for *m=1, 2.* BIBFA relies on the following probabilistic model:

$$\begin{aligned} \mathbf{z} &\sim \text{N}(0, \mathbf{I}), \\ \mathbf{z}^{(m)} &\sim \text{N}(0, \mathbf{I}), \\ \mathbf{x}^{(m)} &\sim \text{N}(\mathbf{A}^{(m)}\mathbf{z} + \mathbf{B}^{(m)}\mathbf{z}^{(m)}, \boldsymbol{\Sigma}^{(m)}) \end{aligned}$$

shared unshared

$$\begin{aligned} \mathbf{y} &\sim \text{N}(0, \mathbf{I}), \\ \mathbf{x} &\sim \text{N}(\mathbf{W}\mathbf{y}, \boldsymbol{\Sigma}), \end{aligned}$$

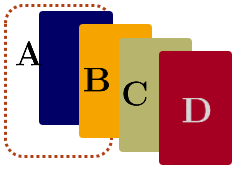
$$\mathbf{W} = \begin{bmatrix} \mathbf{A}^{(1)} & \mathbf{B}^{(1)} & 0 \\ \mathbf{A}^{(2)} & 0 & \mathbf{B}^{(2)} \end{bmatrix}, \mathbf{y} = \begin{bmatrix} \mathbf{z} \\ \mathbf{z}^{(1)} \\ \mathbf{z}^{(2)} \end{bmatrix}, \boldsymbol{\Sigma} = \begin{bmatrix} \boldsymbol{\Sigma}^{(1)} & 0 \\ 0 & \boldsymbol{\Sigma}^{(2)} \end{bmatrix}.$$

Uniqueness is achieved by assuming maximally orthogonal components!

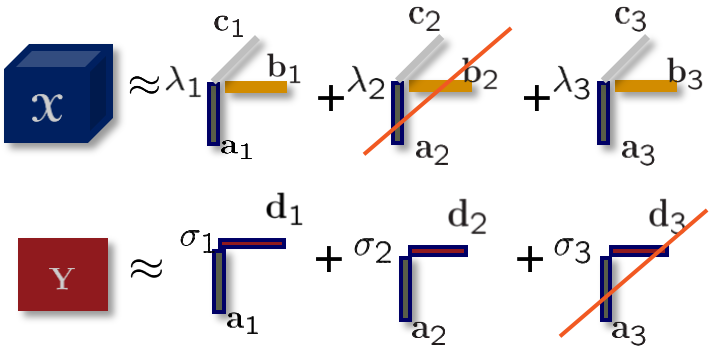
Case 1 (Easy): All methods can capture the underlying factors well!

[Acar, Bro, and Smilde, 2015]

Generate factor matrices

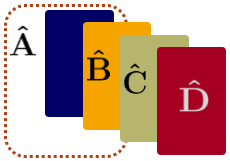


Construct coupled data sets



Compare

GSVD, Adapted GSVD, JIVE, BIBFA, ACMTF

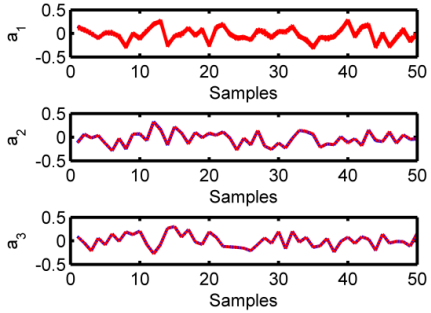


A vs. Â

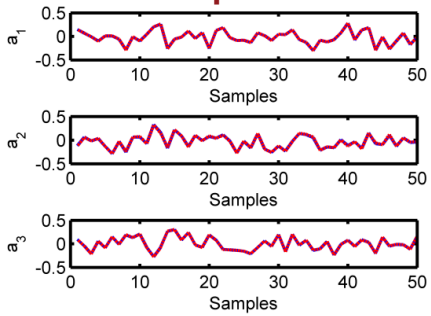
One shared and one unshared component in each data set:

$$[\lambda_1\ \lambda_2\ \lambda_3] = [1\ 0\ 1]$$
$$[\sigma_1\ \sigma_2\ \sigma_3] = [1\ 1\ 0]$$

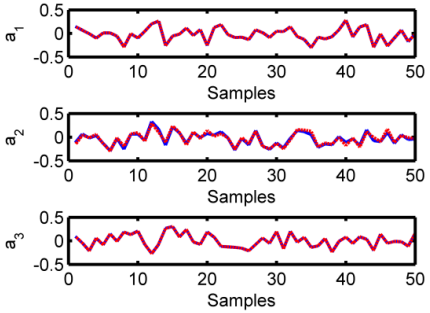
GSVD



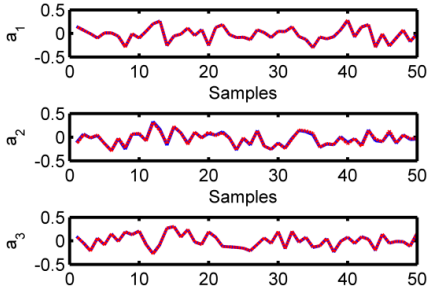
Adapted GSVD



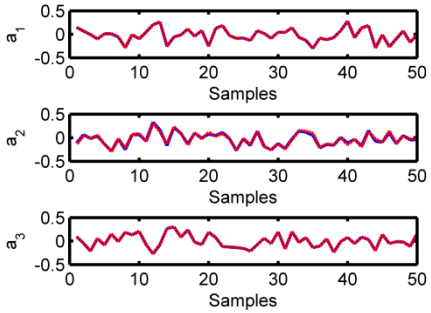
JIVE

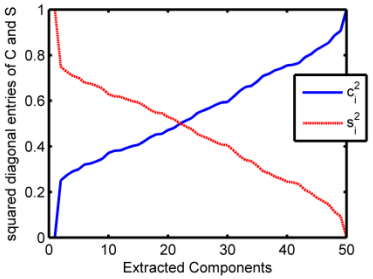


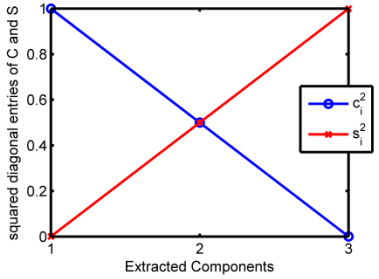
BIBFA



ACMTF



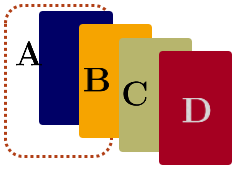




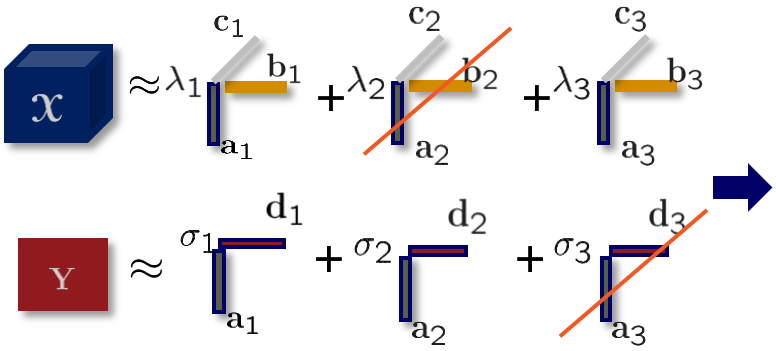
many components similar to the shared component

Case 1 (Easy): All methods can capture the underlying factors well!

Generate factor matrices

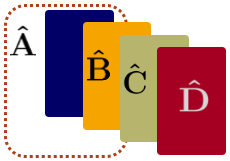


Construct coupled data sets



Compare

GSVD, Adapted GSVD, JIVE, BIBFA, ACMTF

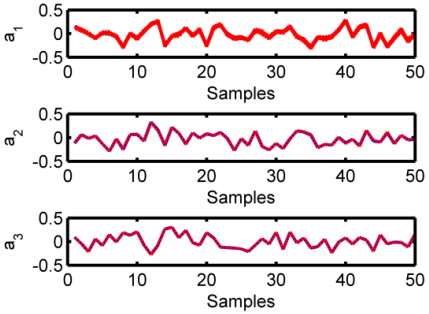


A vs. Â

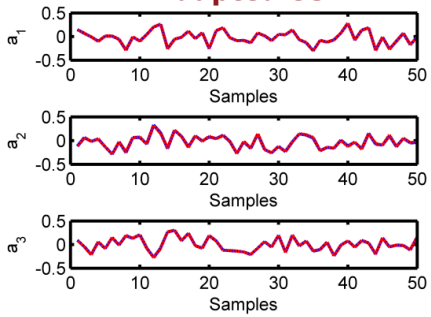
One shared and one unshared component in each data set:

$$[\lambda_1\ \lambda_2\ \lambda_3] = [1\ 0\ 1]$$
$$[\sigma_1\ \sigma_2\ \sigma_3] = [1\ 1\ 0]$$

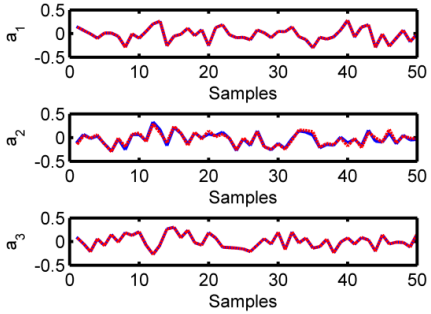
GSVD



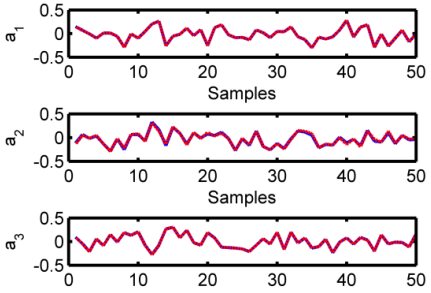
Adapted GSVD



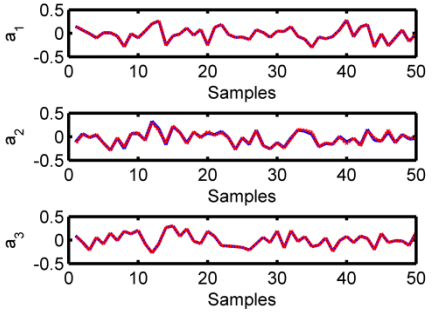
JIVE



BIBFA



ACMTF

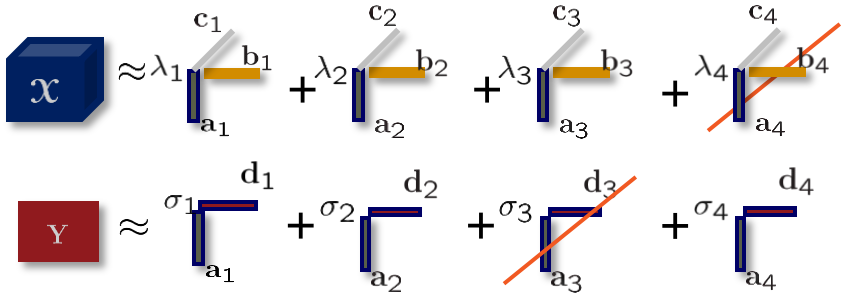


$\frac{\hat{a}_r^T a_r}{\|\hat{a}_r\| \|a_r\|}$	Case1		
	1	2	3
GSVD	1.00*	1.00	1.00
Adapted GSVD	1.00	1.00	1.00
JIVE	1.00	0.97	1.00
BIBFA	1.00	0.99	1.00
ACMTF	1.00	0.99	1.00

Case 2 (Moderate): BIBFA and ACMTF are competitive!

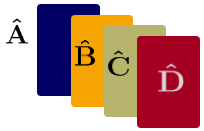
Generate factor matrices

Correlations between the factors are higher!



Compare

GSVD, Adapted GSVD, JIVE, BIBFA, ACMTF

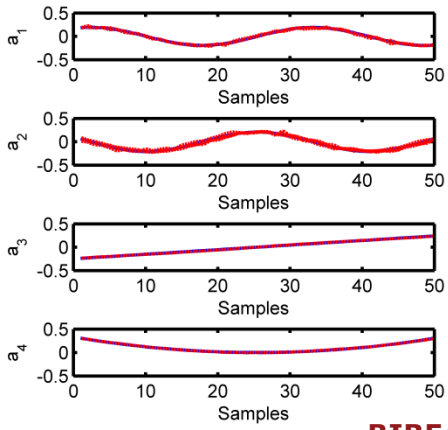


$[\lambda_1\ \lambda_2\ \lambda_3\ \lambda_4] = [1\ 1\ 1\ 0]$

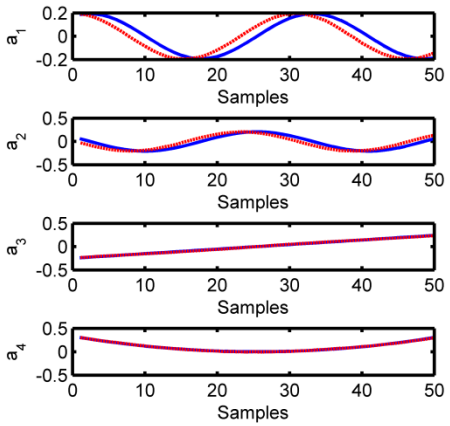
$[\sigma_1\ \sigma_2\ \sigma_3\ \sigma_4] = [1\ 1\ 0\ 1]$

A vs. Â

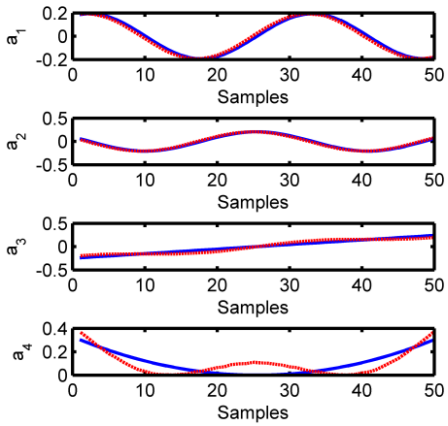
GSVD



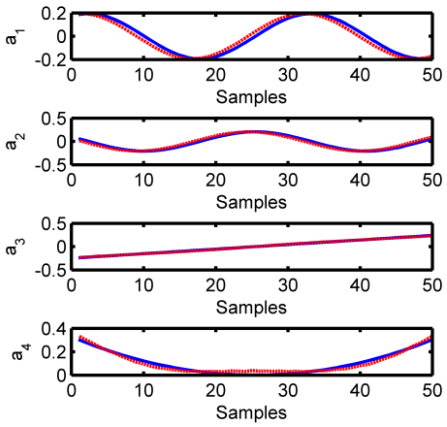
Adapted GSVD



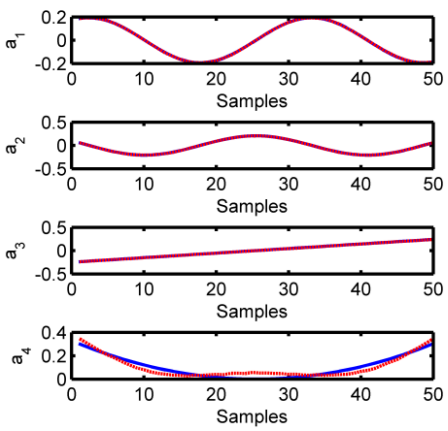
JIVE



BIBFA

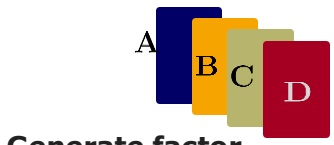


ACMTF

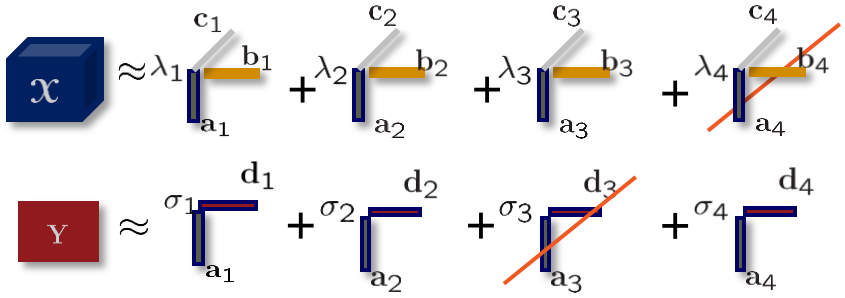


$\frac{\hat{a}_r^T a_r}{\|\hat{a}_r\| \|a_r\|}$	Case2			
	1	2	3	4
GSVD	1.00*	0.99*	1.00	1.00
Adapted GSVD	0.90	0.90	1.00	1.00
JIVE	0.99	0.99	0.97	0.89
BIBFA	0.98	0.98	1.00	0.99
ACMTF	1.00	1.00	1.00	0.97

Case 3 (Difficult): Increase noise level and matrix factorization-based approaches all fail!

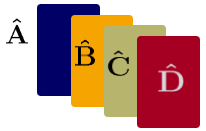


Generate factor matrices



Compare

GSVD, Adapted GSVD, JIVE, BIBFA, ACMTF

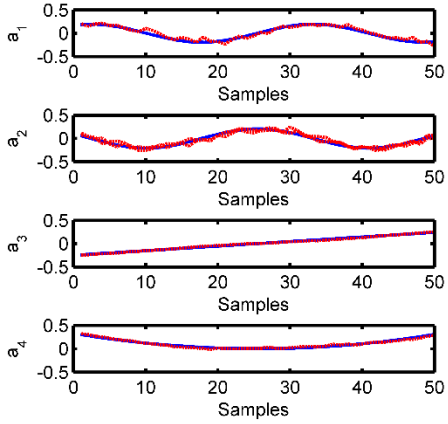


$[\lambda_1\ \lambda_2\ \lambda_3\ \lambda_4] = [1\ 1\ 1\ 0]$

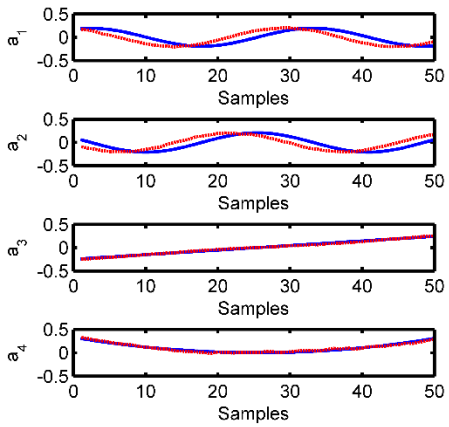
$[\sigma_1\ \sigma_2\ \sigma_3\ \sigma_4] = [1\ 1\ 0\ 1]$

A vs. Â

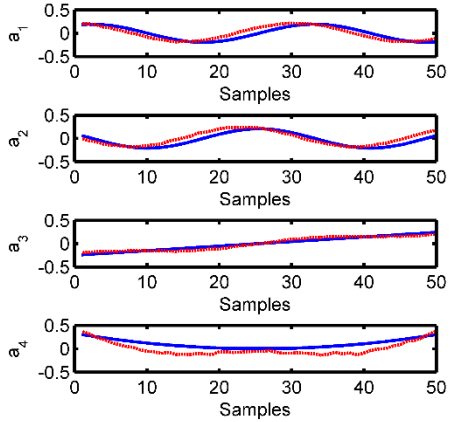
GSVD



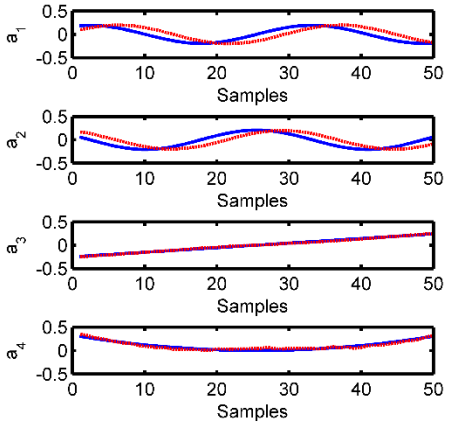
Adapted GSVD



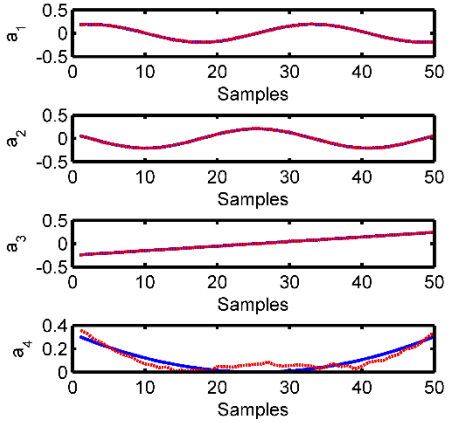
JIVE



BIBFA



ACMTF



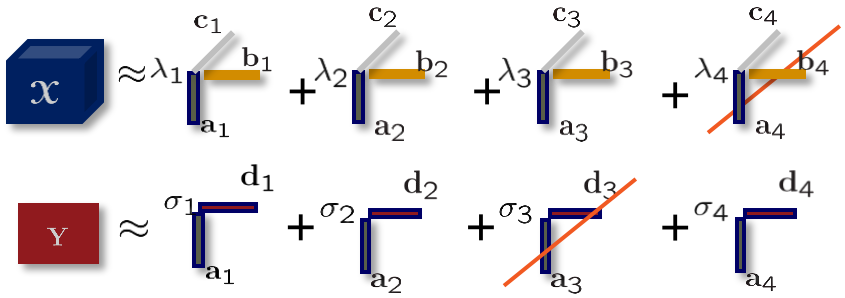
$\frac{\hat{a}_r^T a_r}{\|\hat{a}_r\| \|a_r\|}$	Case3			
	1	2	3	4
GSVD	0.95	0.95	1.00	0.99
Adapted GSVD	0.72	0.72	1.00	0.99
JIVE	0.87	0.84	0.96	0.61
BIBFA	0.72	0.72	1.00	0.98
ACMTF	1.00	1.00	1.00	0.96

Case 4 (Difficult): Make factors highly correlated and matrix factorization-based approaches all fail!

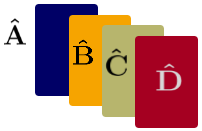
$$[\lambda_1\ \lambda_2\ \lambda_3\ \lambda_4] = [1\ 1\ 1\ 0]$$
$$[\sigma_1\ \sigma_2\ \sigma_3\ \sigma_4] = [1\ 1\ 0\ 1]$$

Generate factor matrices

Highly correlated factors!

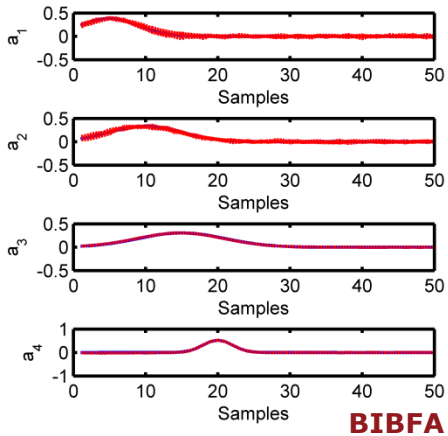


Compare
GSVD, Adapted GSVD, JIVE, BIBFA, ACMTF

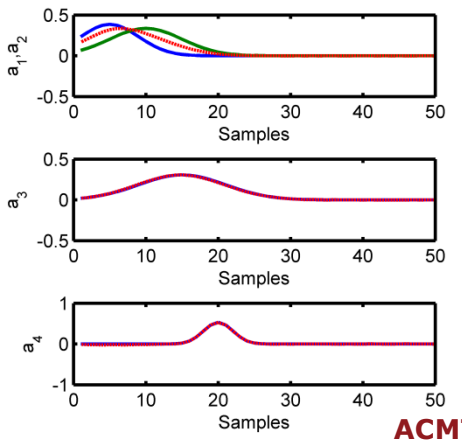


A vs. Â

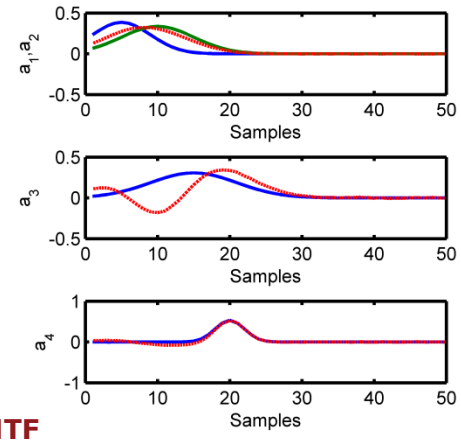
GSVD



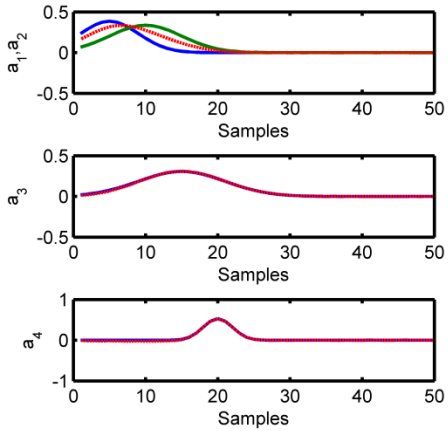
Adapted GSVD



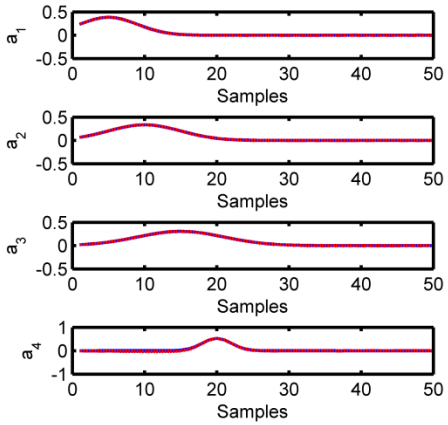
JIVE



BIBFA



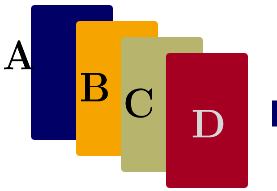
ACMTF



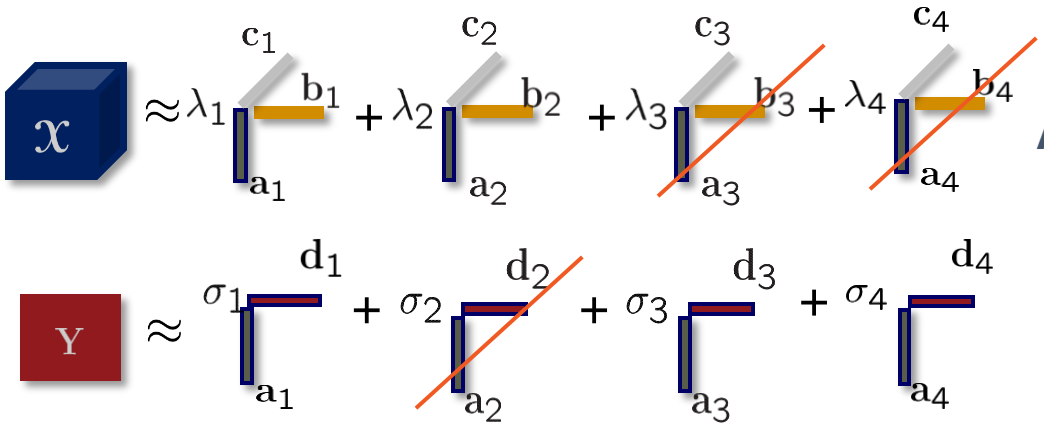
$\frac{\hat{a}_r^T a_r}{\|\hat{a}_r\| \|a_r\|}$	Case4			
	1	2	3	4
GSVD	0.99*	1.00*	1.00	1.00
Adapted GSVD	0.94	0.93	1.00	1.00
JIVE	0.87	0.98	0.49	0.98
BIBFA	0.93	0.93	1.00	1.00
ACMTF	1.00	1.00	1.00	1.00

Uniqueness: There will be cases where ACMTF is not unique!

Generate factor matrices



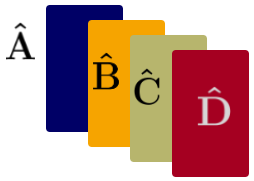
Construct coupled data sets



ACMTF

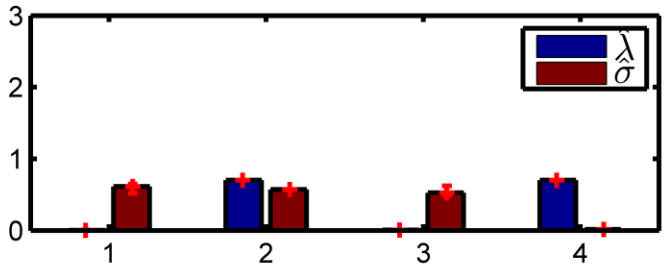
Minimize

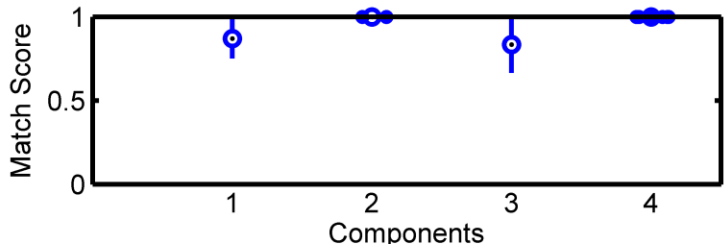
$f(\hat{\lambda}, \hat{\Sigma}, \hat{A}, \hat{B}, \hat{C}, \hat{D})$



Two unshared components in the matrix:

$$[\lambda_1 \; \lambda_2 \; \lambda_3 \; \lambda_4] = [1 \; 1 \; 0 \; 0]$$
$$[\sigma_1 \; \sigma_2 \; \sigma_3 \; \sigma_4] = [1 \; 0 \; 1 \; 1]$$





$$\frac{\hat{a}_r^{\top} a_r}{\| \hat{a}_r \| \, \| a_r \|}$$

Two unshared components in the matrix span the same subspace in different runs returning the same function value.

Since they cannot be recovered uniquely, though, they cannot be compared using the match score anymore.

simula

Application$_8$ (Chemometrics): Joint analysis of measurements from multiple platforms has the potential to enhance biomarker discovery

[Acar et al., *BMC Bioinformatics*, 2014]

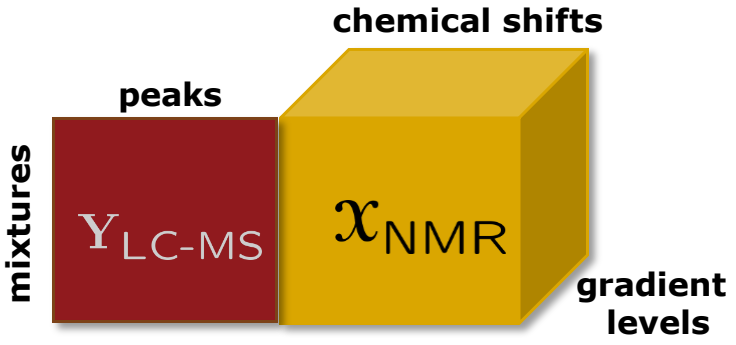
Goal: To identify shared/unshared factors in each data set

Data: 29 mixtures measured using DOSY (diffusion-ordered) NMR spectroscopy and LC-MS. Mixtures are prepared using the following five chemicals:

- Val-Try-Val
- Trp – Gly
- Phe
- Maltoheptaose
- Propanol

Four chemicals are expected to show up in both types of measurements!

→ **Can only be captured by NMR**

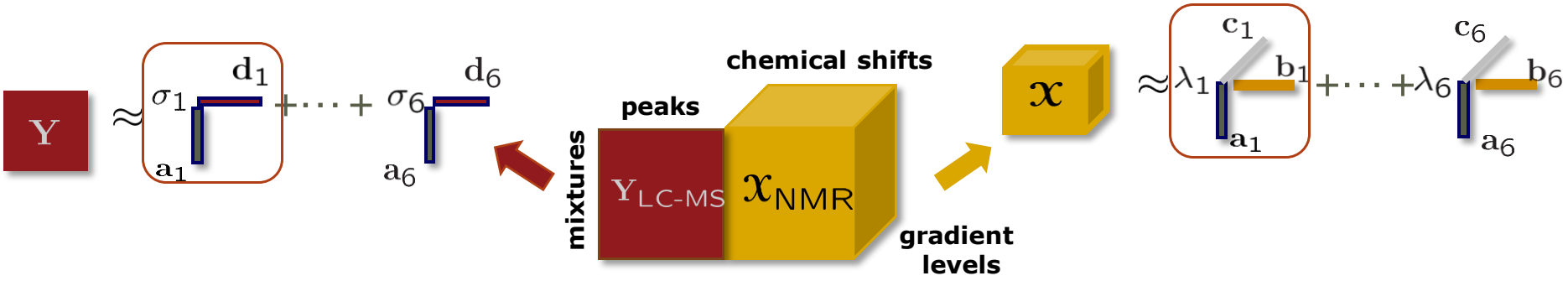


simula

ACMTF can capture the chemicals

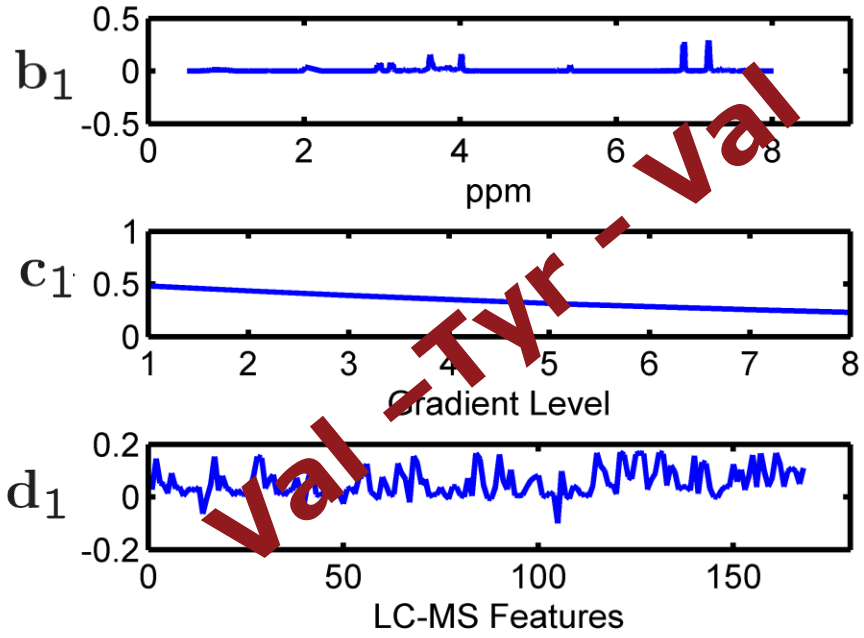
$$\min_{A,B,C,D,\Sigma,\lambda} \| \mathcal{X} - [\![\lambda; A, B, C]\!] \|^2 + \left\| Y - A\Sigma D^T \right\|^2 + \beta\|\lambda\|_1 + \beta\|\sigma\|_1$$

$$s.t. \|a_r\|_2 = \|b_r\|_2 = \|c_r\|_2 = \|d_r\|_2 = 1, \text{for } r = 1, \ldots, R$$



LC-MS	NMR
σ_r	λ_r
0.20	0.28
0.15	0.19
0.17	0.24
0.16	0.77
0.06	0.18
0.65	0.07

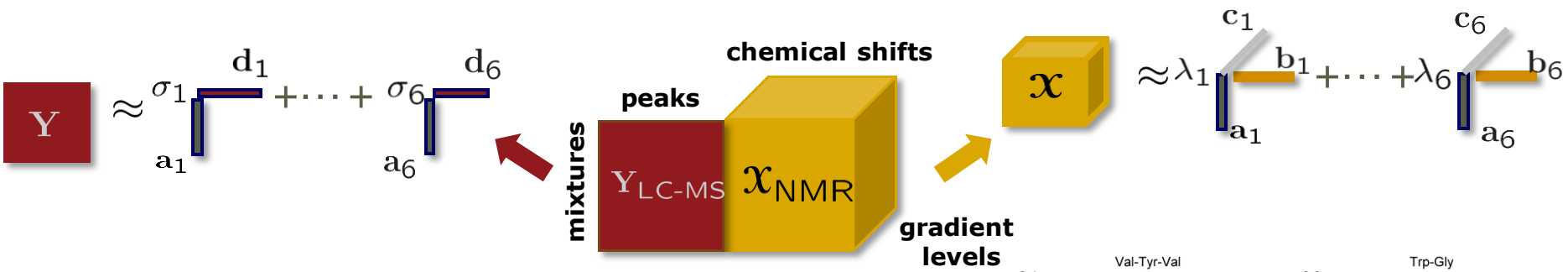
Val –Tyr - Val



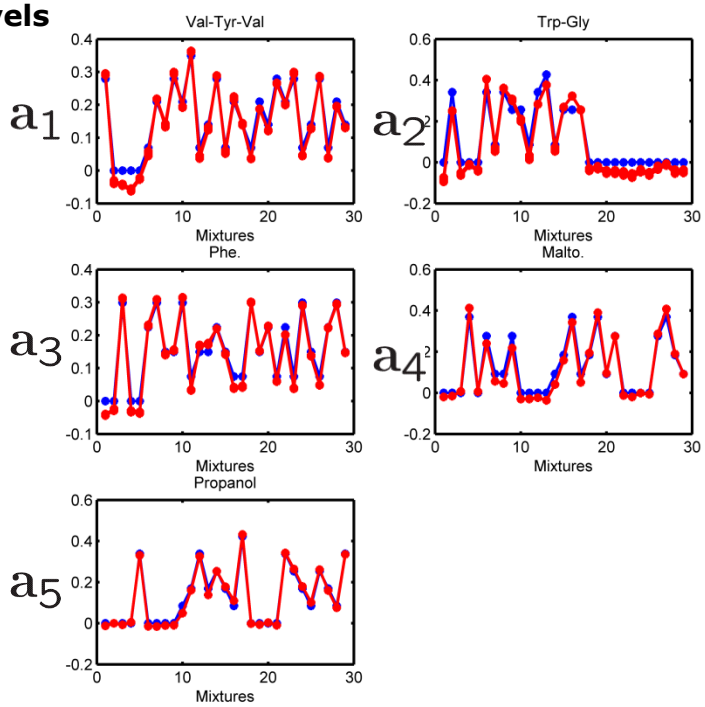
ACMTF can capture the chemicals and the design

$$\min_{A,B,C,D,\Sigma,\lambda} \| \mathcal{X} - [\![\lambda; A, B, C]\!] \|^2 + \left\| Y - A\Sigma D^T \right\|^2 + \beta\|\lambda\|_1 + \beta\|\sigma\|_1$$

$$s.t. \|a_r\|_2 = \|b_r\|_2 = \|c_r\|_2 = \|d_r\|_2 = 1, \text{for } r = 1, \ldots, R$$

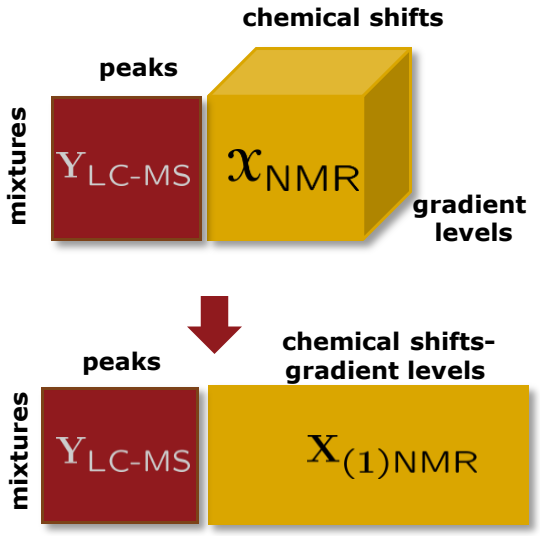


LC-MS	NMR	
σ_r	λ_r	
0.20	0.28	Val –Tyr - Val
0.15	0.19	Trp - Gly
0.17	0.24	Phe
0.16	0.77	Maltoheptaose
0.06	0.18	Propanol
0.65	0.07	Noise

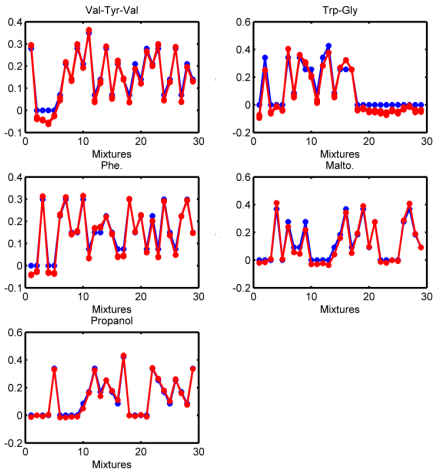


Matrix factorization-based data fusion models fail!

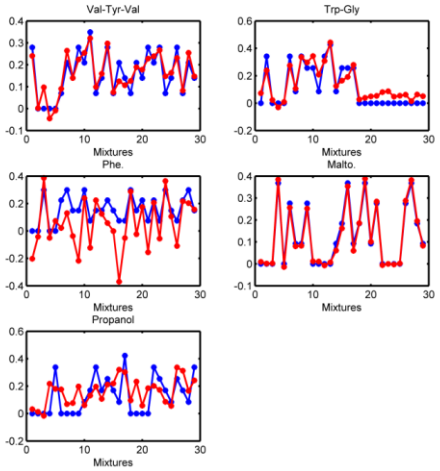
[Acar, Bro, Smilde, 2015]



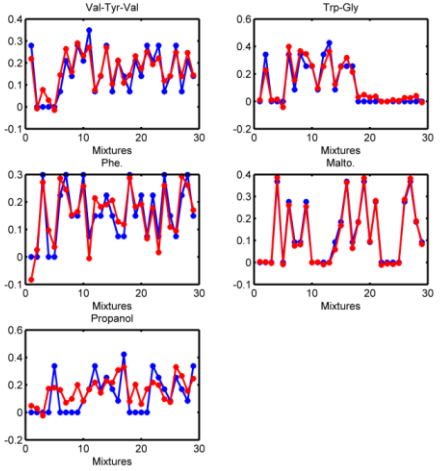
ACMTF



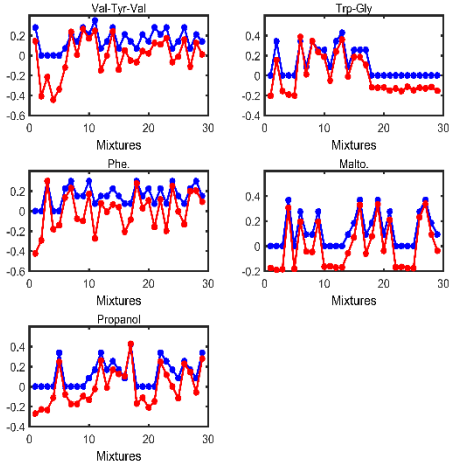
GSVD



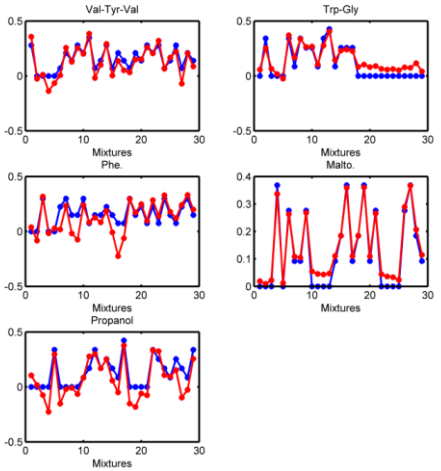
Adapted GSVD



JIVE

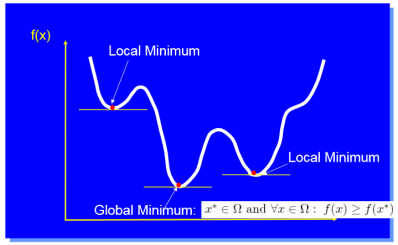


BIBFA



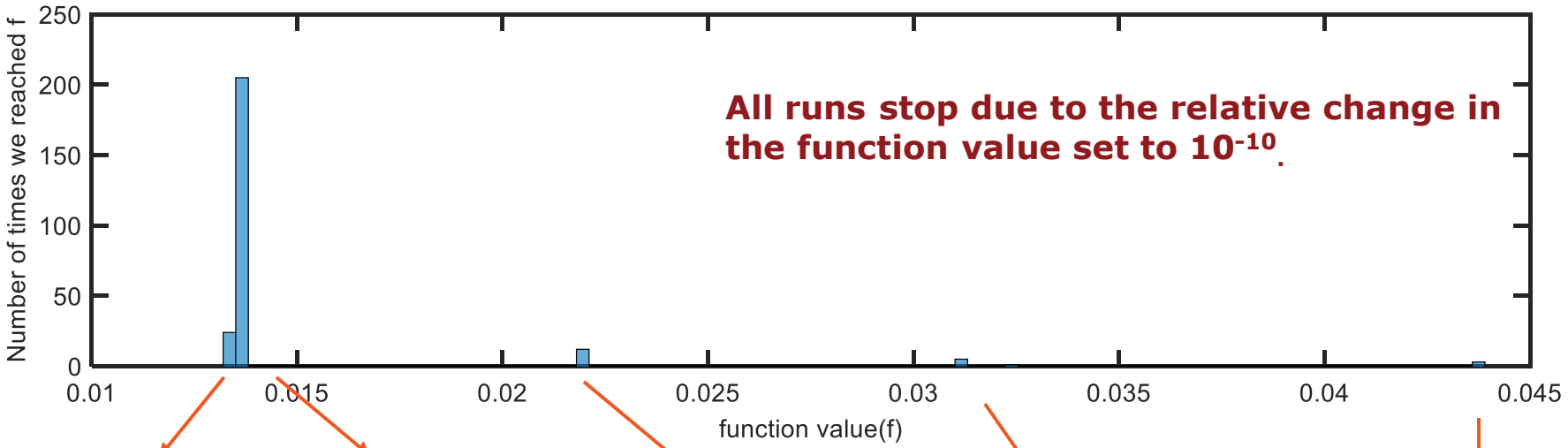
Recently more studies on structure-revealing CMTF models, i.e., Khan et al., Machine Learning, 2016; Farias et al., IEEE Transactions on Signal Processing, 2016.

Different initializations may lead to different solutions!



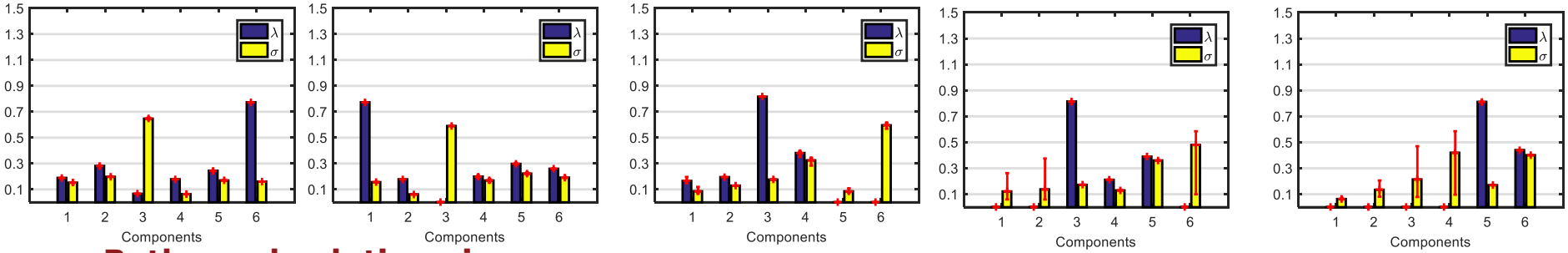
The minimum function value: $f(\hat{\lambda}, \hat{\Sigma}, \hat{\mathbf{A}}, \hat{\mathbf{B}}, \hat{\mathbf{C}}, \hat{\mathbf{D}}) = 0.0134$

Out of 250 runs with random initializations + one svd-based initialization, we get the minimum function value 24 times.



All runs stop due to the relative change in the function value set to 10^{-10}.

$f = 0.0134$ $f = 0.0136$ $f = 0.0220$ $f = 0.0312$ $f = 0.0438$



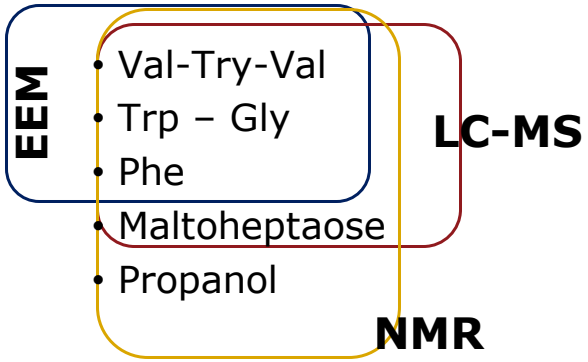
Both good solutions in terms of the application!

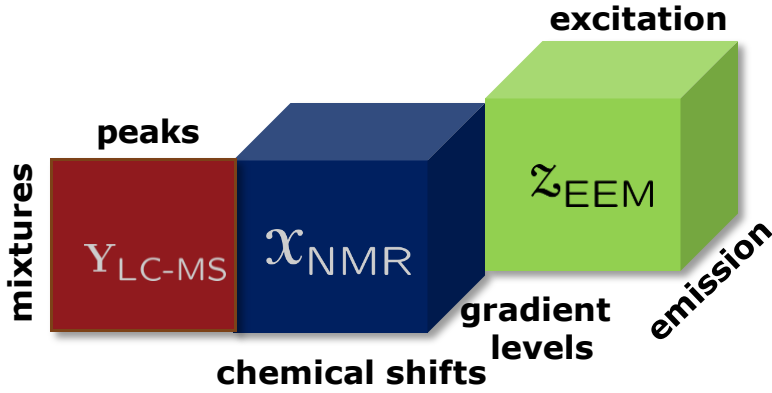
Completely wrong results!

ACMTF reveals chemicals visible in different platforms!

Goal: To identify shared/unshared factors in each data set

Data: Mixtures measured using DOSY (diffusion-ordered) NMR spectroscopy and LC-MS. Mixtures are prepared using the following five chemicals:



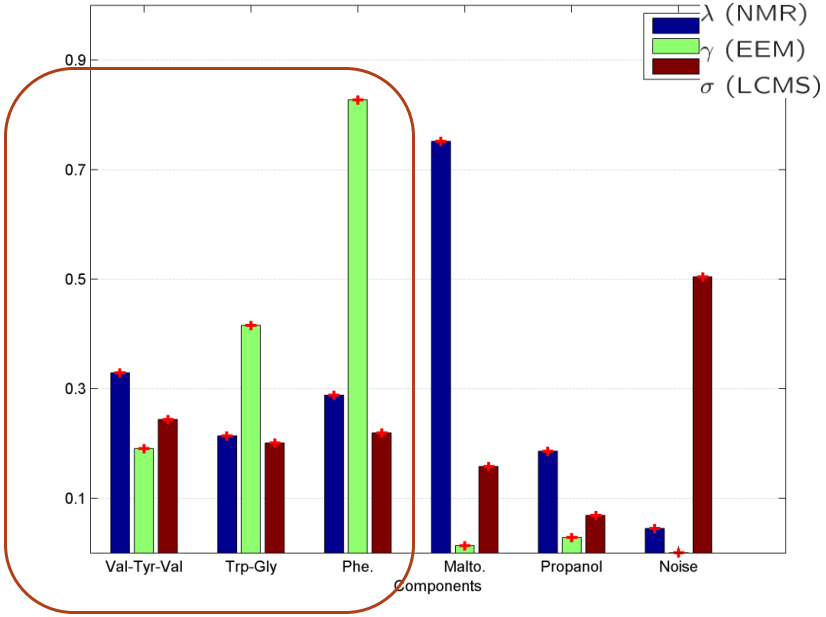


$$\mathcal{X} \approx [\![\lambda; \mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$$

$$\mathbf{Y} \approx \mathbf{A}\Sigma\mathbf{D}^{\mathsf{T}}$$

$$\mathcal{Z} \approx [\![\gamma; \mathbf{A}, \mathbf{E}, \mathbf{F}]\!]$$

Shared among all

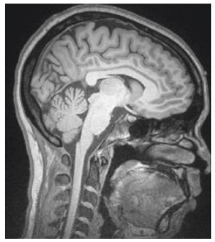


Application$_9$ (Neuroscience): Joint analysis of signals from multiple neuroimaging modalities promises to provide a better understanding of the brain function

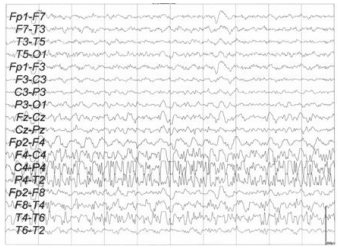
The complexity of the human brain often necessitates the use of **multiple neuroimaging techniques** to better **understand neural activities**. Neuroimaging techniques provide **complementary** information at different scales.

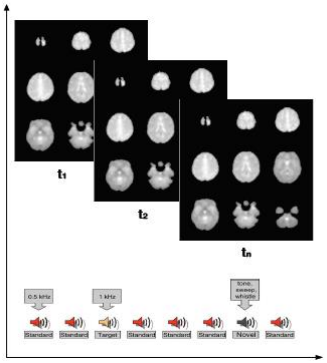
Functional Magnetic Resonance Imaging (fMRI)

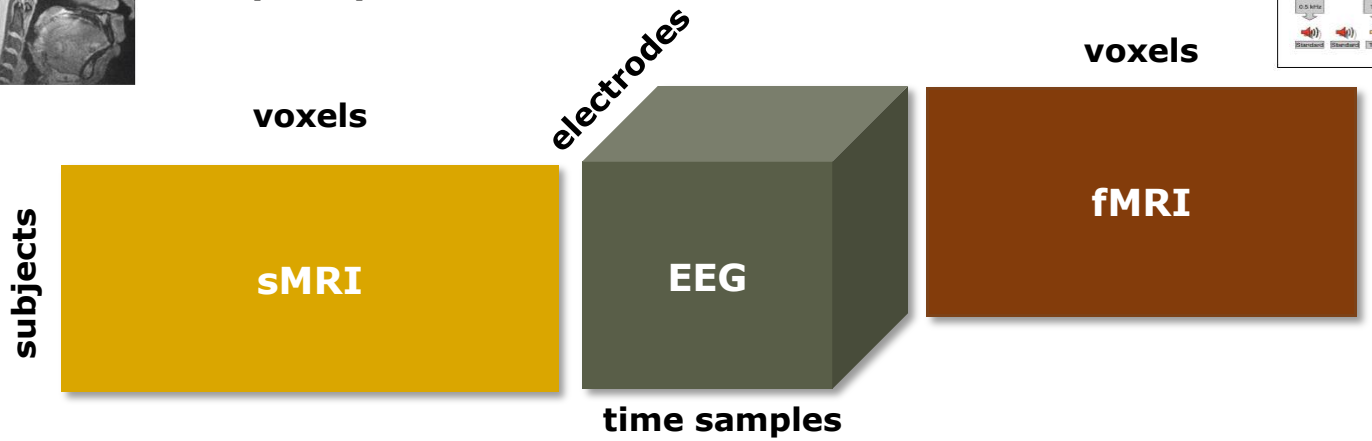
EEG (electroencephalography)



Structural MRI (sMRI)



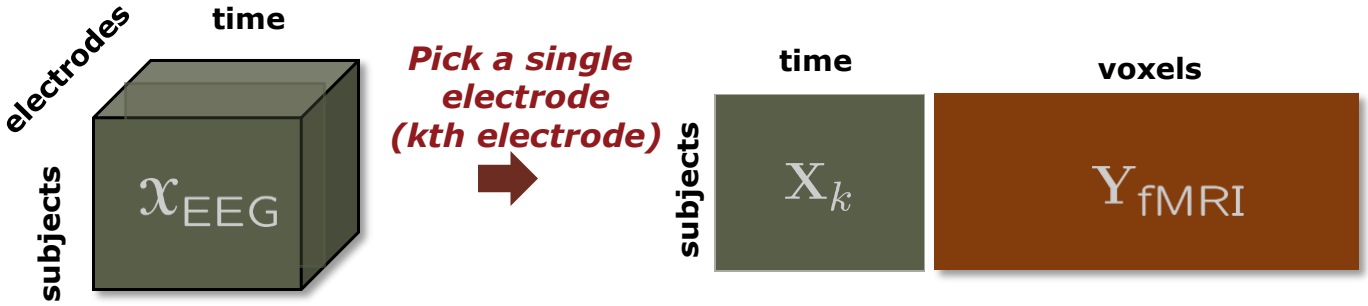




Can we jointly analyze these data sets and capture functional/structural patterns that differ between healthy controls and patients suffering from a disorder?

Joint Independent Component Analysis (jICA) has been a popular approach to multi-modal neuroimaging data fusion

Single Electrode:



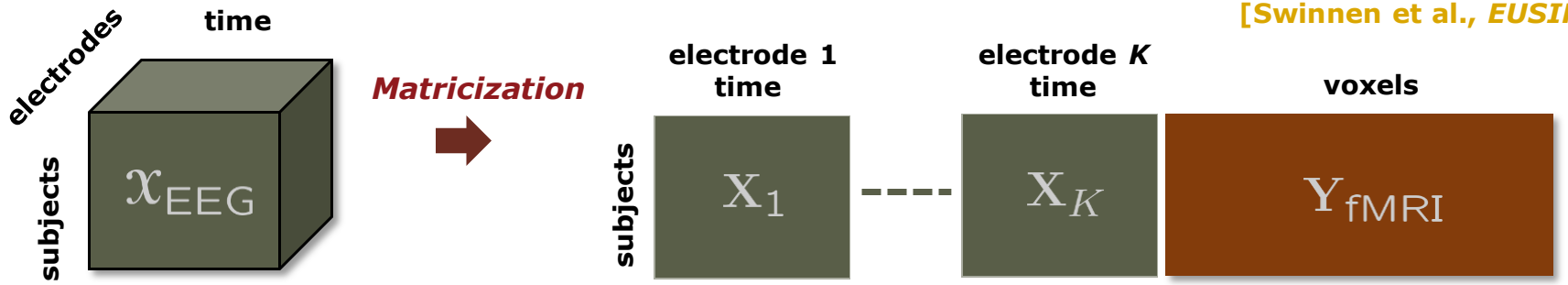
[Calhoun et al. *NeuroImage*, 2006]

$$[\mathbf{X}_k \quad \mathbf{Y}] = \mathbf{A}\mathbf{S}$$

Multiple Electrodes:

$$[\mathbf{X}_1 \quad \mathbf{X}_2 \quad \ldots \quad \mathbf{X}_K \quad \mathbf{Y}] = \mathbf{A}\mathbf{S}$$

[Swinnen et al., *EUSIPCO*, 2014]

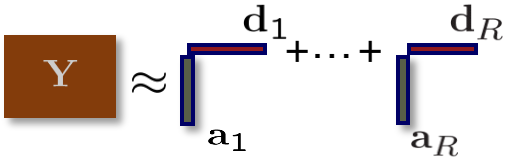


However, these approaches do not take into account the potential multi-linear structure of the EEG data.

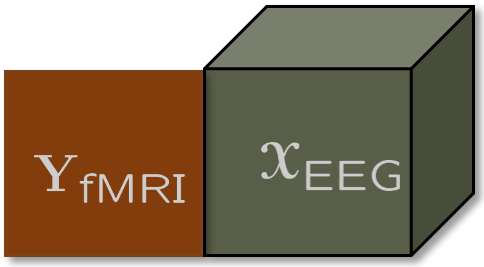
simula

Multi-way structure can be preserved using Coupled Matrix and Tensor Factorizations

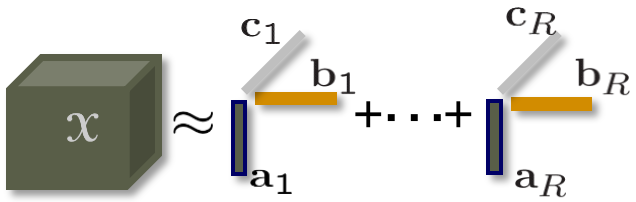
Matrix Factorization



$$\mathbf{Y} \approx \sum_{r=1}^{R} \mathbf{a}_r \mathbf{d}_r^{\mathsf{T}} \approx \mathbf{A}\mathbf{D}^{\mathsf{T}}$$



Tensor Factorization based on CP



$$\mathcal{X} \approx \sum_{r=1}^{R} \mathbf{a}_r \circ \mathbf{b}_r \circ \mathbf{c}_r \approx [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$$

$$\mathbf{Y} \approx \mathbf{A}\mathbf{D}^{\mathsf{T}} \quad \mathcal{X} \approx [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$$

$$\min_{\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D}} \| \mathcal{X} - [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!] \|^2 + \| \mathbf{Y} - \mathbf{A}\mathbf{D}^{\mathsf{T}} \|^2$$

Coupled tensor factorizations in neuroscience:

EEG & magnetoencephalography (MEG) **[Becker et al., *EUSIPCO*, 2012; Naskovska et al., *EUSIPCO*, 2017]**

EEG & Gaze data **[Rivet et al., *IEEE EMBC*, 2015]**

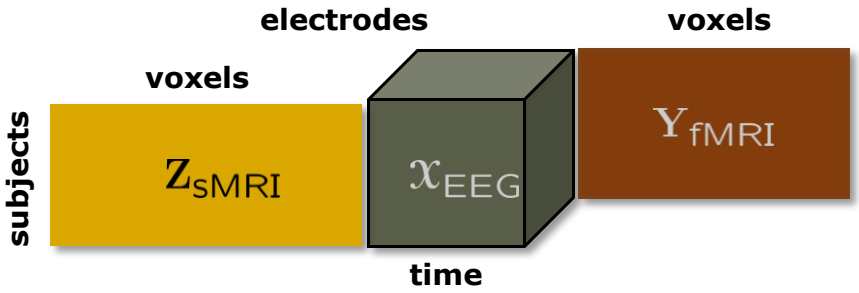
EEG & fMRI:

- Single-subject data coupled in spatial mode **[Karahan et al., *Proceedings of the IEEE*, 2015]**
- Data from multiple subjects coupled in the subjects mode modeled using CMTF **[Hunyadi et al., *EUSIPCO*, 2016]**

Our Approach to Multi-modal Neuroimaging Data Fusion

[Acar, Levin-Schwartz, Calhoun, Adali, 2017]

Data representation: Multi-subject EEG, fMRI and sMRI data coupled in the *subjects* mode:

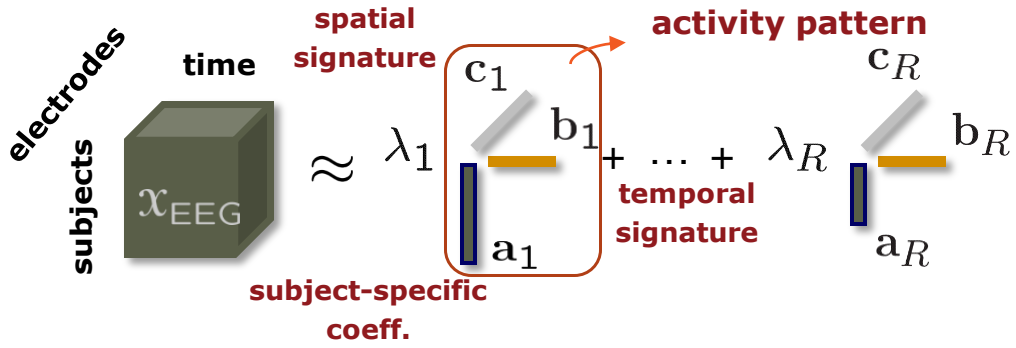


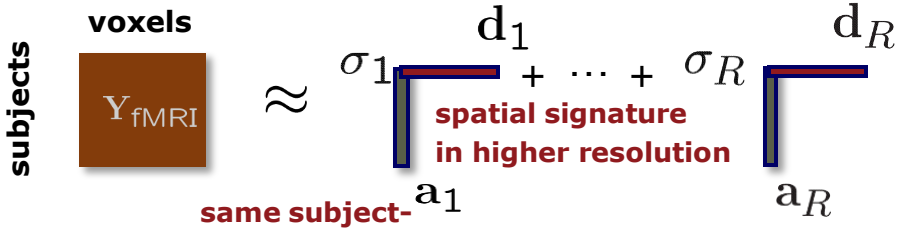
Model: Structure-revealing CMTF

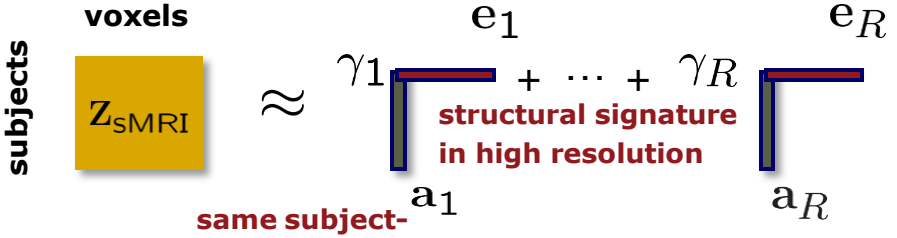
$$\mathcal{X} \approx [\![\lambda; \mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$$

$$\mathbf{Y} \approx \mathbf{A}\boldsymbol{\Sigma}\mathbf{D}^{\mathsf{T}}$$

$$\mathbf{Z} \approx \mathbf{A}\boldsymbol{\Gamma}\mathbf{E}^{\mathsf{T}}$$



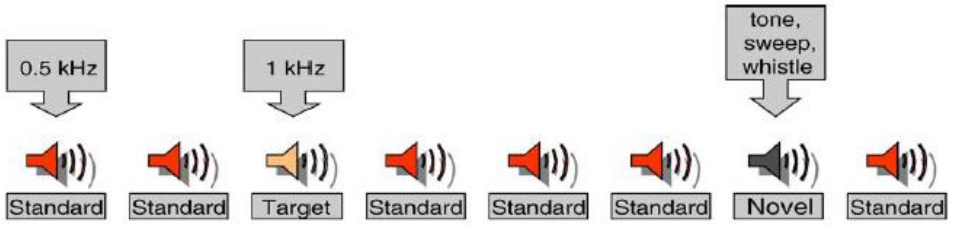


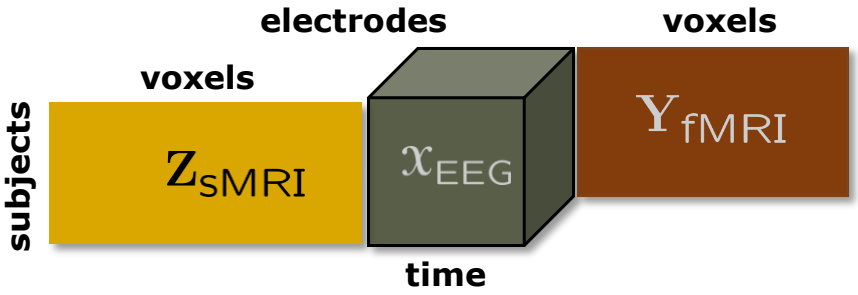


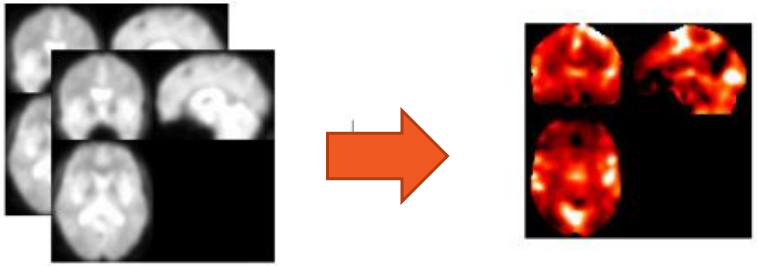
Modeling Assumption:
Subject coefficients are the same in all modalities!

Construction of Coupled Data sets: Feature Extraction

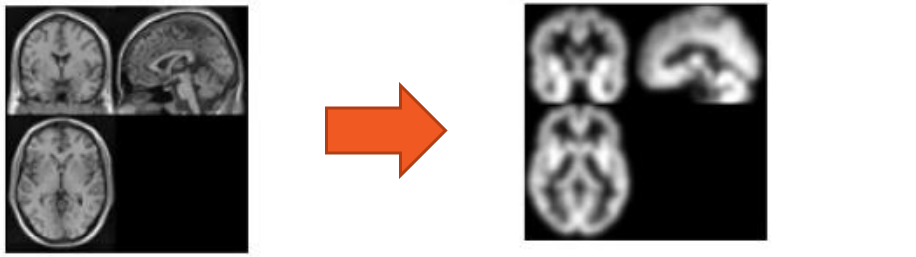
Healthy controls and **Patients** with schizophrenia
From an auditory oddball task (AOD)



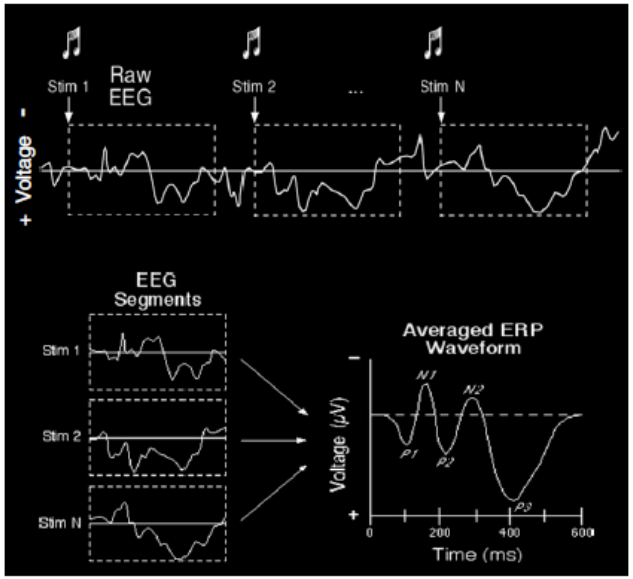




fMRI time series images *Task-related contrast images*



sMRI images *Segmented gray matter images*



Event related potential (ERP)

Experimental Setting

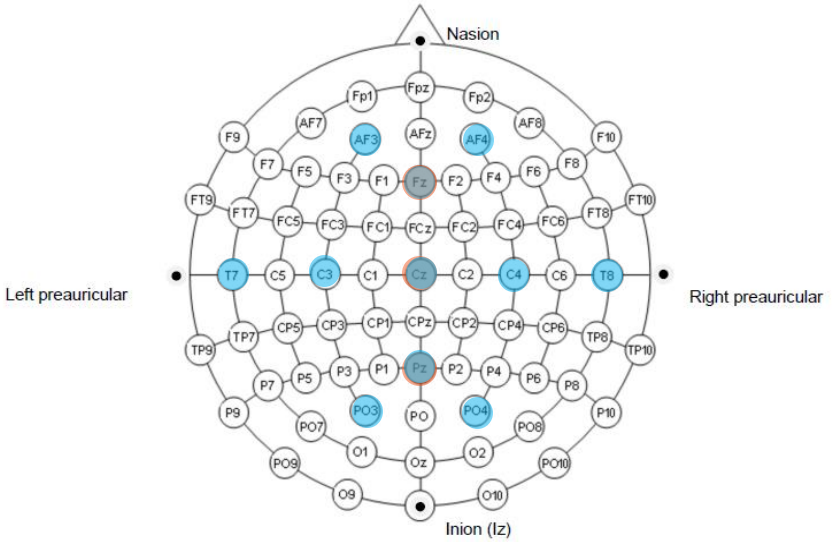
Models:

CP Model to analyze multi-channel EEG

3 electrodes : Cz, Pz, Fz

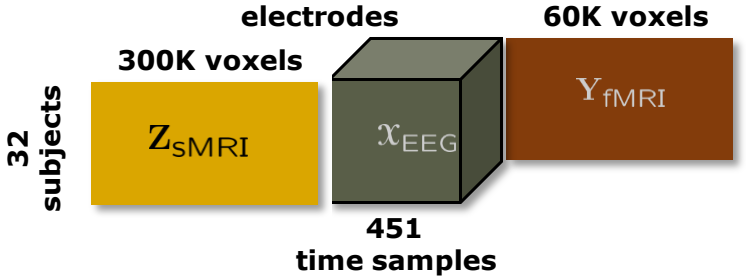
11 electrodes : AF3, AF4, Fz, T7, C3, Cz, C4, T8, Pz, PO3, PO4

62 electrodes excluding VEOG/HEOG



Structure-revealing CMTF to jointly analyze EEG-fMRI

Structure-revealing CMTF vs. Joint ICA to jointly analyze EEG-fMRI-sMRI



Algorithms: Multiple random starts are used to deal with the local minima problem

CP: CP-OPT (Tensor Toolbox)

Structure-revealing CMTF: ACMTF-OPT (CMTF Toolbox)

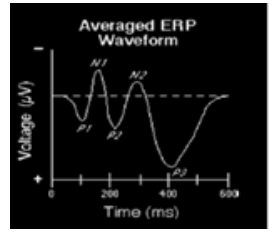
Joint ICA: ICA (Entropy Bound Minimization [Li & Adali, *IEEE Trans Signal Process*, 2010])

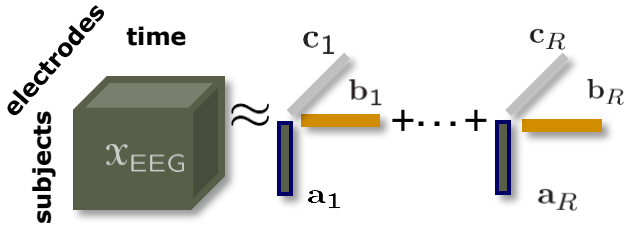
We study

- ***which electrodes to choose***
- ***added value of joint analysis***
- ***structure-revealing CMTF vs. jICA***

simula

EEG only: CP captures statistically significant factors





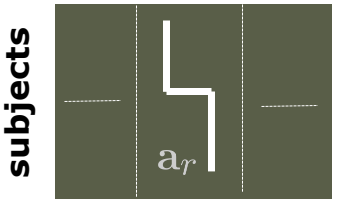
$$\mathcal{X} \approx \sum_{r=1}^{R} \mathbf{a}_r \circ \mathbf{b}_r \circ \mathbf{c}_r \approx [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$$

Number of components: *R* = 3

Significant components identified using a two-sample *t*-test on the columns of A

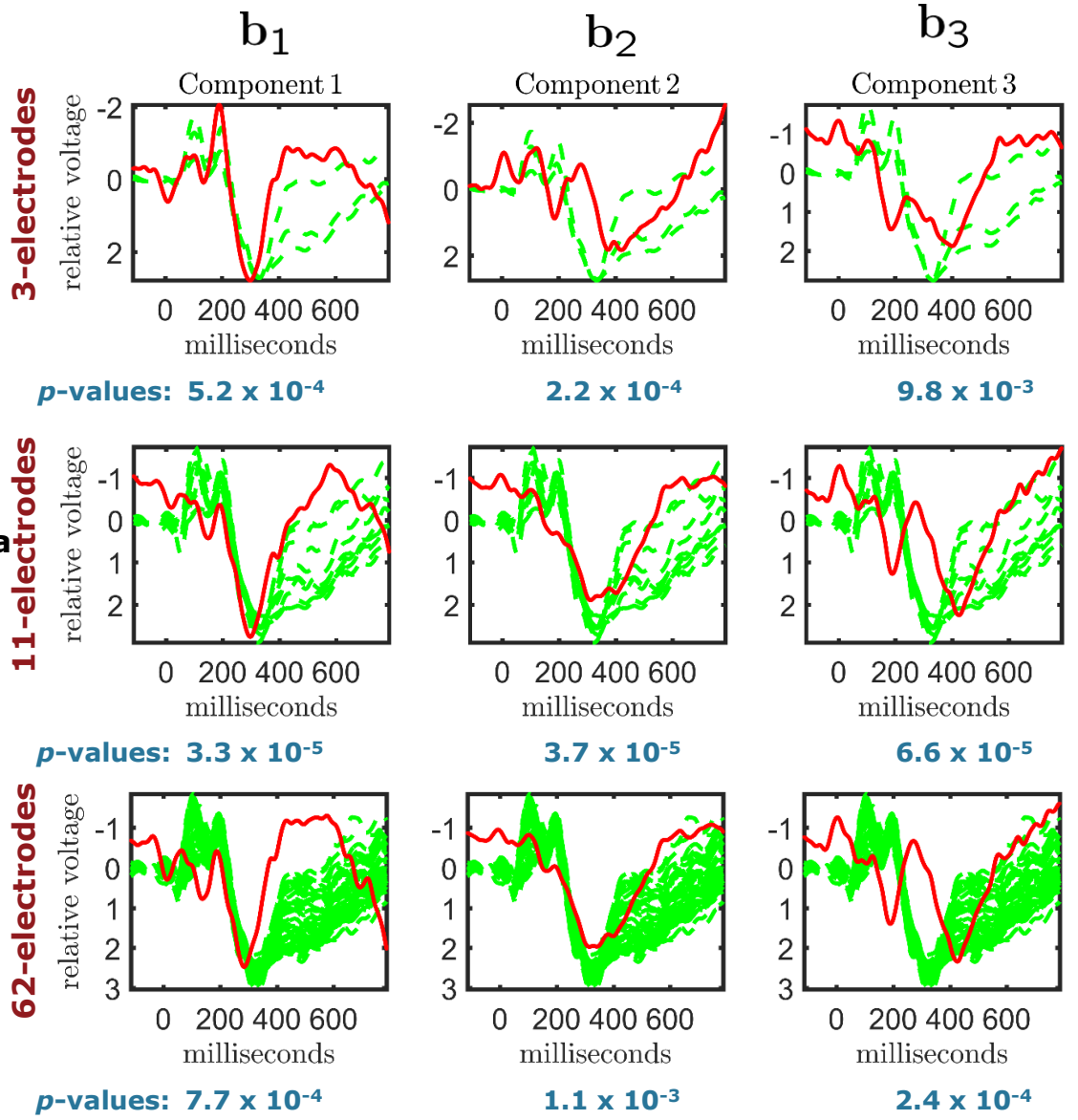
22 healthy

16 patients

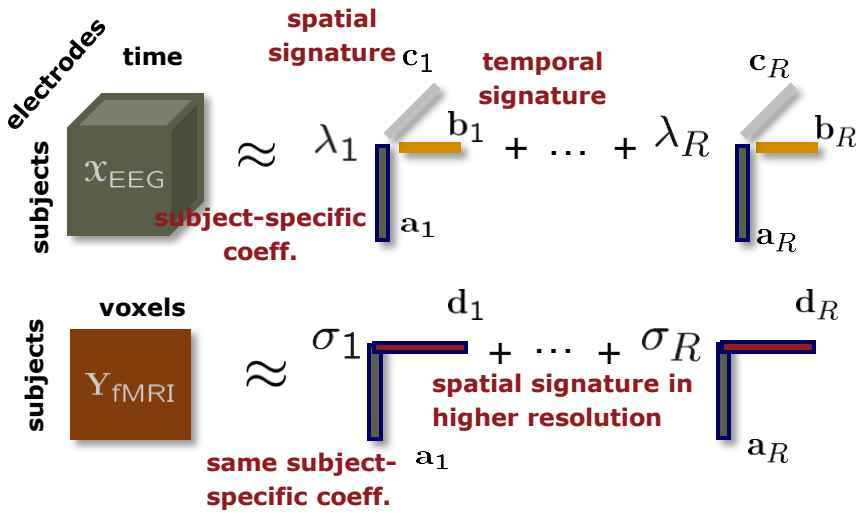


Meaningful components, in particular,* Comp1: *N2-P3 transition

Highest statistical significance with 11 electrodes



EEG coupled with fMRI: Structure-revealing CMTF captures meaningful temporal & spatial patterns with high resolution



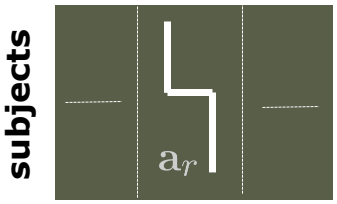
$$\mathcal{X} \approx [\![\lambda; \mathbf{A}, \mathbf{B}, \mathbf{C}]\!] \qquad \mathbf{Y} \approx \mathbf{A}\Sigma\mathbf{D}^{\mathsf{T}}$$

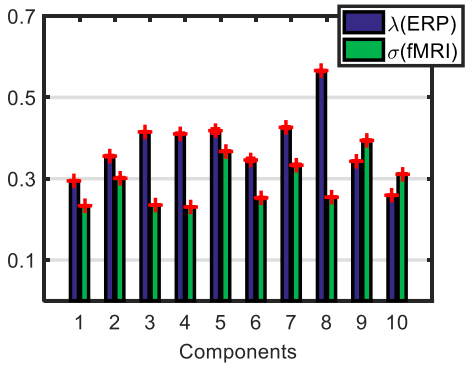
Number of components: *R* = 10

Significant components identified using a two-sample *t*-test on the columns of A

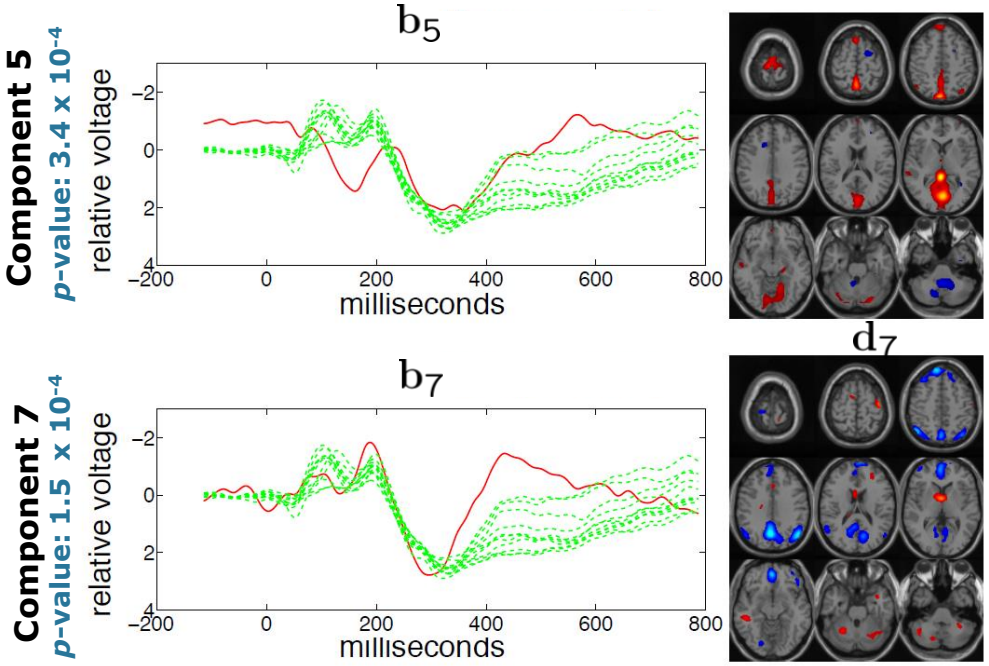
22 healthy

16 patients





11-electrodes

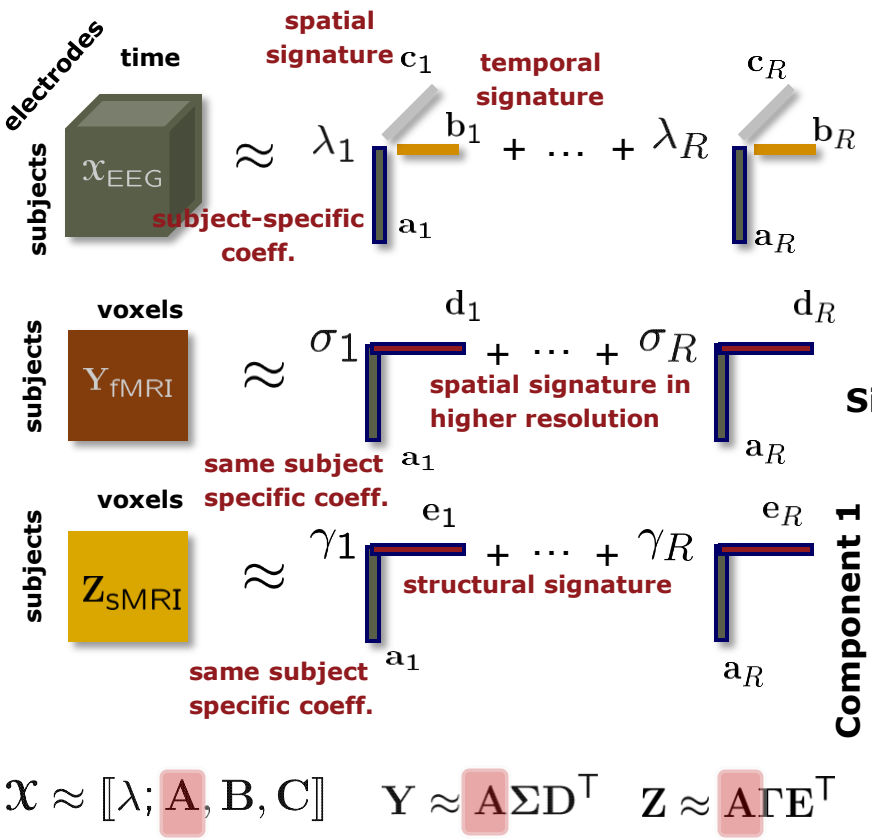


Meaningful components

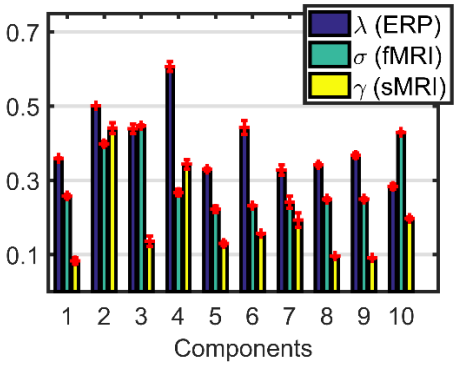
Comp5: ***P2/P3 peaks in EEG, fMRI activation in the superior parietal and visual cortex***

Comp7: ***N2 -P3 transition in EEG, DMN (Default Mode Network) activation in fMRI***

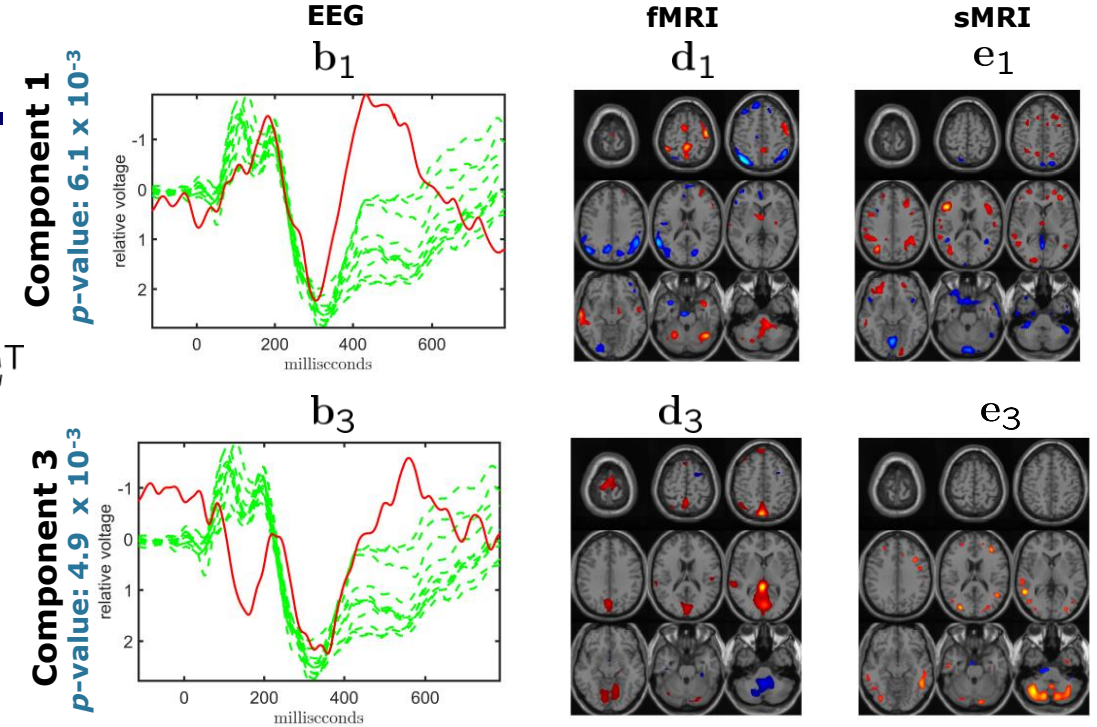
EEG coupled with fMRI & sMRI: Structure-revealing CMTF captures meaningful temporal & spatial patterns + structure information!



$$\mathcal{X} \approx [\![\lambda; \mathbf{A}, \mathbf{B}, \mathbf{C}]\!] \quad \mathbf{Y} \approx \mathbf{A}\Sigma\mathbf{D}^\top \quad \mathbf{Z} \approx \mathbf{A}\Gamma\mathbf{E}^\top$$



Significant components—based on two-sample *t*-test on $\mathbf{a}_r$

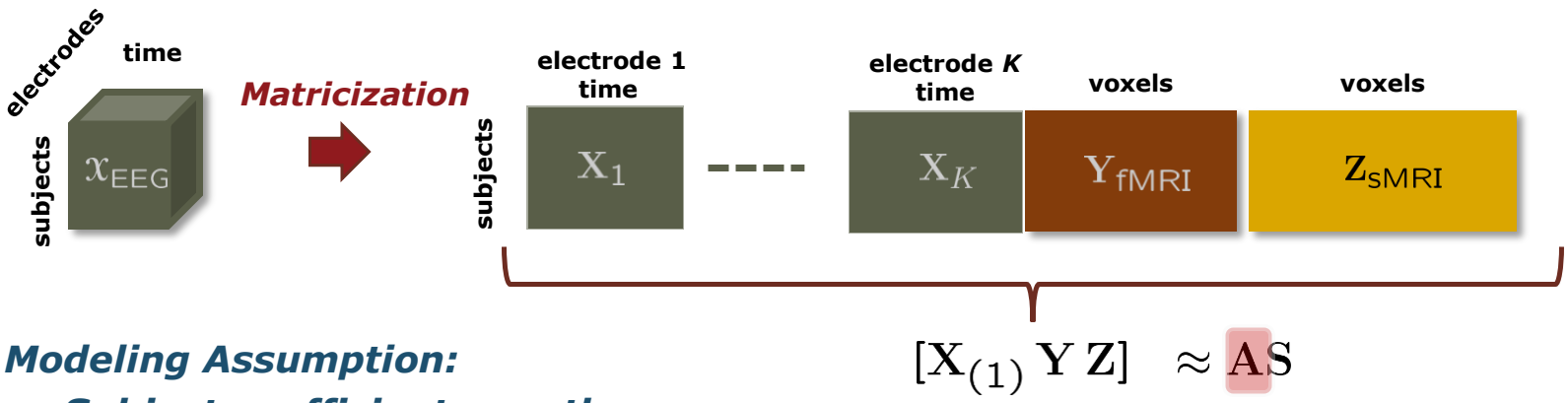


Number of components: *R* = 10

Comp1: ***N2-P3 transition in EEG, DMN activation in fMRI, gray matter concentration in the parietal lobe***

Comp3: ***P2/P3 peaks in EEG, fMRI activation in the superior parietal and visual cortex, gray matter concentration in parietal lobe and cerebellum***

EEG coupled with fMRI & sMRI: Joint ICA also captures components differentiating between patients and controls!



Modeling Assumption:

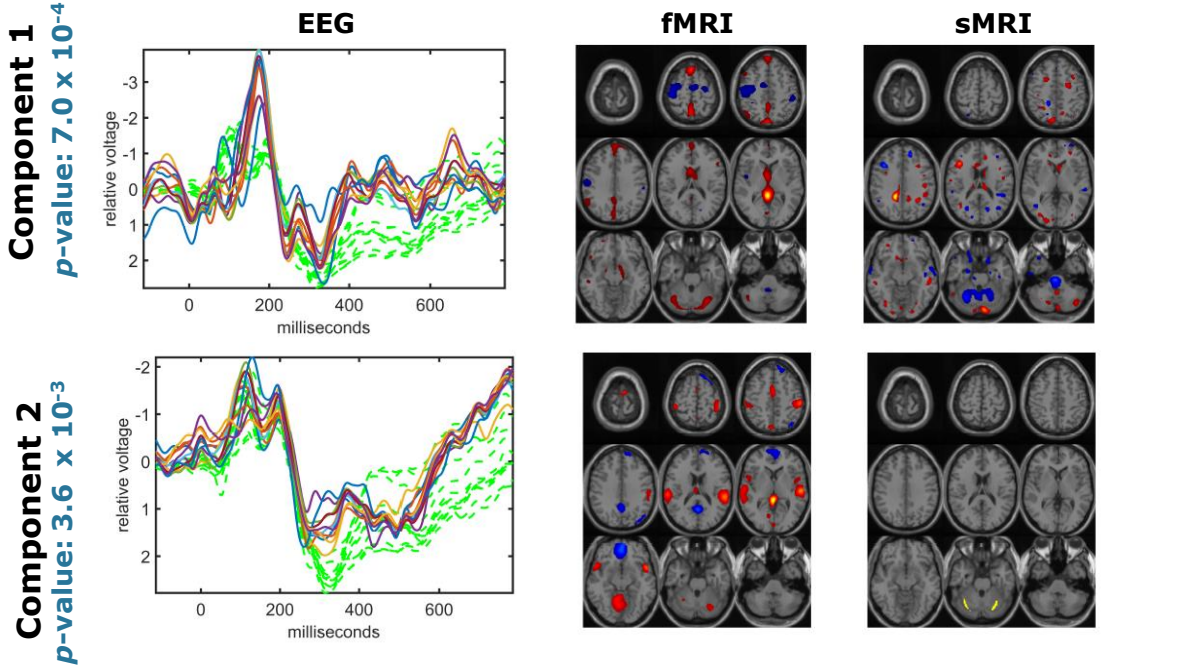
- ***Subject coefficients are the same in all modalities!***
- ***Rows of S are assumed to be statistically independent***

Number of components: *R* = 10

Comp1: ***N2 peak in EEG, fMRI activation in the superior parietal cortex, gray matter concentration in the cerebellum***

Comp2: ***N1, P2, N2 peaks and N2-P3 transition in EEG, fMRI activation in the motor cortex and temporal lobe, gray matter concentration in the cerebellum***

Significant components—based on two-sample *t*-test on a_r



Clustering measures demonstrate the added value of fMRI and clearly the performance difference between ACMTF and joint ICA

$$\mathcal{X} \approx [\![\lambda; \mathbf{A}_{cp}, \mathbf{B}, \mathbf{C}]\!] \qquad \mathcal{X} \approx [\![\lambda; \mathbf{A}_{acmtf}, \mathbf{B}, \mathbf{C}]\!] \qquad [\mathbf{X}_{(1)}\, \mathbf{Y}\, \mathbf{Z}] \approx \mathbf{A}_{jica}\mathbf{S}$$

$$\mathbf{Y} \approx \mathbf{A}_{acmtf}\boldsymbol{\Sigma}\mathbf{D}^{\mathsf{T}}$$

$$\mathbf{Z} \approx \mathbf{A}_{acmtf}\boldsymbol{\Gamma}\mathbf{E}^{\mathsf{T}}$$

Data Sets	Model	# of electrodes	Performance of k-means *			
Added value of fMRI			Accuracy (%)	F-score	Purity	Normalized Mutual Information
EEG (38 subjects)	CP	3	79	0.78	0.79	0.28
	CP	11	79	0.80	0.79	0.42
	CP	62	79	0.80	0.79	0.42
EEG -fMRI (38 subjects)	ACMTF	3	82	0.76	0.82	0.31
	ACMTF	11	87	0.84	0.87	0.43
	ACMTF	62	87	0.86	0.87	0.47
EEG- fMRI - sMRI (32 subjects)	ACMTF	11	88	0.81	0.88	0.43
	Joint ICA	11	81	0.77	0.81	0.34

**best performance out of all possible combinations of components*

ACMTF can be formulated as a constrained optimization problem

[Acar, Nilsson, Saunders, 2014]

In order to have a flexible modeling framework, we use a general-purpose optimization solver SNOPT (Sparse Nonlinear OPTimizer) [Gill et al., *SIAM Review*, 2005]

SNOPT is designed for large constrained optimization problems with smooth nonlinear functions in the objective and constraints.

SNOPT uses a sequential quadratic programming (SQP) algorithm to minimize an augmented Lagrangian.

SNOPT

$$\min_{x \in \mathbb{R}^n} \quad \phi(x)$$

$$\texttt{s.t.} \quad l \leq \begin{pmatrix} x \\ Ax \\ c(x) \end{pmatrix} \leq u$$

where *c(x)* indicates nonlinear functions, and A is a sparse matrix.

Structure-revealing CMTF model:

$$\min_{\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D},\mathbf{\Sigma},\lambda} \| \mathcal{X} - [\![\lambda; \mathbf{A}, \mathbf{B}, \mathbf{C}]\!] \|^2 + \| \mathbf{Y} - \mathbf{A}\mathbf{\Sigma}\mathbf{D}^\mathsf{T} \|^2$$

$$\texttt{s.t.} \quad \| \mathbf{a}_r \|_2 = \| \mathbf{b}_r \|_2 = \| \mathbf{c}_r \|_2 = \| \mathbf{d}_r \|_2 = 1$$

$$\sum_{r=1}^{R} \lambda_r \leq \beta, \sum_{r=1}^{R} \sigma_r \leq \beta$$

$$\sigma_r, \lambda_r \geq 0, \text{for } r = 1, ..., R.$$

simula

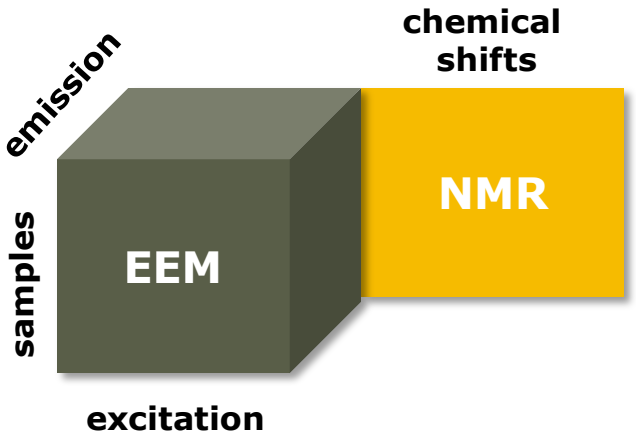
Additional constraints are easily incorporated

In many data fusion problems, we may need the following constraints to capture the underlying structures accurately and improve the interpretation.

Nonnegativity Constraints:

$$\min_{\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D},\Sigma,\lambda} \| \mathcal{X} - [\![\lambda; \mathbf{A}, \mathbf{B}, \mathbf{C}]\!] \|^2 + \| \mathbf{Y} - \mathbf{A}\Sigma\mathbf{D}^{\mathsf{T}} \|^2$$

$$\begin{aligned} \texttt{s.t.} \quad & \| \mathbf{a}_r \|_2 = \| \mathbf{b}_r \|_2 = \| \mathbf{c}_r \|_2 = \| \mathbf{d}_r \|_2 = 1 \\ & \sum_{r=1}^{R} \lambda_r \leq \beta, \sum_{r=1}^{R} \sigma_r \leq \beta \\ & \sigma_r, \lambda_r \geq 0, b_{jr}, c_{kr}, d_{mr} \geq 0 \\ & \text{for } r = 1:R, j = 1:J, k = 1:K, m = 1:M. \end{aligned}$$



Angular Constraints: When coupled data sets are overfactored, one shared factor may be represented by two closely-correlated factors. In that case, the structure–revealing model will fail to identify shared factors accurately.

$$\min_{\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D},\Sigma,\lambda} \| \mathcal{X} - [\![\lambda; \mathbf{A}, \mathbf{B}, \mathbf{C}]\!] \|^2 + \| \mathbf{Y} - \mathbf{A}\Sigma\mathbf{D}^{\mathsf{T}} \|^2$$

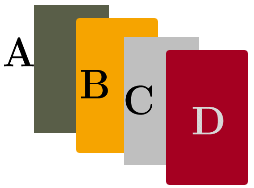
$$\begin{aligned} \texttt{s.t.} \quad & \| \mathbf{a}_r \|_2 = \| \mathbf{b}_r \|_2 = \| \mathbf{c}_r \|_2 = \| \mathbf{d}_r \|_2 = 1 \\ & |\mathbf{a}_r^{\mathsf{T}}\mathbf{a}_p| \leq \theta, |\mathbf{b}_r^{\mathsf{T}}\mathbf{b}_p| \leq \theta, |\mathbf{c}_r^{\mathsf{T}}\mathbf{c}_p| \leq \theta, |\mathbf{d}_r^{\mathsf{T}}\mathbf{d}_p| \leq \theta \\ & \sum_{r=1}^{R} \lambda_r \leq \beta, \sum_{r=1}^{R} \sigma_r \leq \beta \\ & \sigma_r, \lambda_r \geq 0 \text{ for } r, p \in \{1:R\}, r \neq p. \end{aligned}$$

simula

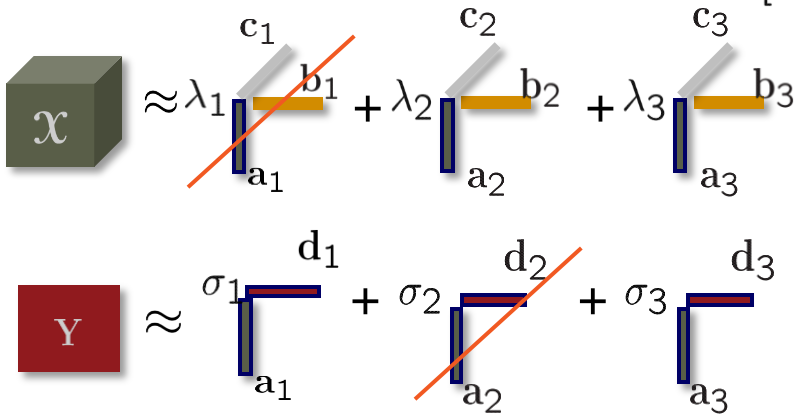
Angular constraints have a promising performance in the case of overfactoring!

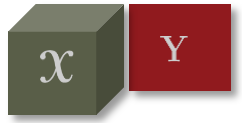
$$[\lambda_1\ \lambda_2\ \lambda_3] = [0\ 1\ 1]$$
$$[\sigma_1\ \sigma_2\ \sigma_3] = [1\ 0\ 1]$$

Generate factor matrices



Construct coupled data sets





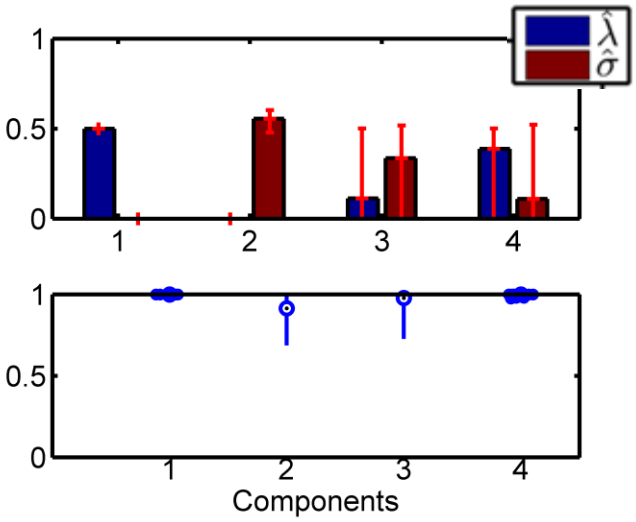
Solve for A, B, C, D, λ, and Σ using constrained optimization

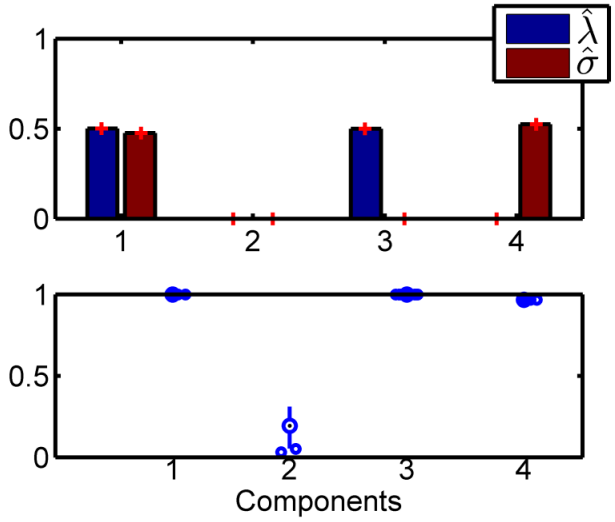
Overfactoring (R=4):

Angular Constraints Inactive

Angular Constraints Active

$\theta = 0.25$

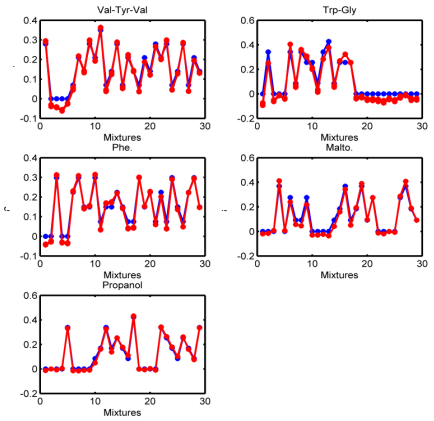




Match score:

$$\frac{\hat{a}_r^{\top} a_r}{\| \hat{a}_r \| \| a_r \|}$$

Summary



Structure-revealing data fusion models

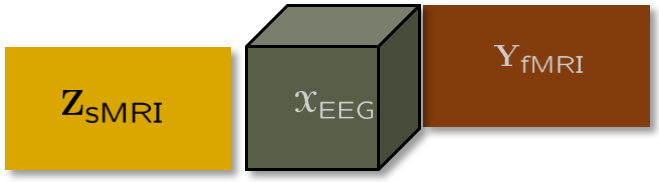
Structure-revealing CMTF (ACMTF)

- Reformulation of CMTF to identify shared/unshared factors
- Can handle missing data just like CMTF

Comparison of ACMTF with matrix factorization-based structure-revealing fusion models

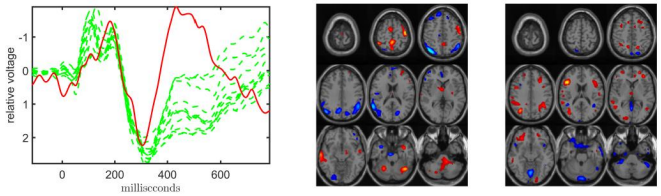
Algorithms:

- **Unconstrained optimization:** Gradient-based optimization algorithms
- **Constrained optimization:** Using SNOPT, we have a more flexible modeling framework



Applications:

- **Chemometrics/metabolomics**
- **Neuroscience**



MATLAB functions

acmtf-opt: Fitting an ACMTF model to coupled data sets **(from CMTF Toolbox)**

[Fac, FacInit, output] = acmtf_opt(X, R, 'init', ..., 'alg',..., 'alg_options', ..., **'beta_cp'**, ..., **'beta_pca'**, ...);

Model: Optimization problem

$$\min_{\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D},\Sigma,\lambda} \| \mathcal{X} - [\![\lambda; \mathbf{A}, \mathbf{B}, \mathbf{C}]\!] \|^2 + \| \mathbf{Y} - \mathbf{A}\Sigma\mathbf{D}^\mathsf{T} \|^2 + \beta \| \lambda \|_1 + \beta \| \sigma \|_1$$
$$\texttt{s.t.} \; \| \mathbf{a}_r \|_2 = \| \mathbf{b}_r \|_2 = \| \mathbf{c}_r \|_2 = \| \mathbf{d}_r \|_2 = 1, \text{for } r = 1, \ldots, R$$

Tensor & CMTF Toolbox
Computation of the function value and the gradient

$$\begin{aligned} f(\lambda, \Sigma, \mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}) &= \| \mathcal{X} - [\![\lambda; \mathbf{A}, \mathbf{B}, \mathbf{C}]\!] \|^2 + \| \mathbf{Y} - \mathbf{A}\Sigma\mathbf{D}^\mathsf{T} \|^2 \\ &+ \beta \sum_{r=1}^{R} \sqrt{\lambda_r^2 + \epsilon} + \beta \sum_{r=1}^{R} \sqrt{\sigma_r^2 + \epsilon} \\ &+ \alpha \sum_{r=1}^{R} (\| \mathbf{a}_r \| - 1)^2 + \alpha \sum_{r=1}^{R} (\| \mathbf{b}_r \| - 1)^2 \\ &+ \alpha \sum_{r=1}^{R} (\| \mathbf{c}_r \| - 1)^2 + \alpha \sum_{r=1}^{R} (\| \mathbf{d}_r \| - 1)^2, \end{aligned}$$

$$\nabla f = \begin{bmatrix} \frac{\partial f}{\partial \mathbf{a}_1} \\ \vdots \\ \frac{\partial f}{\partial \mathbf{a}_R} \\ \frac{\partial f}{\partial \mathbf{b}_1} \\ \vdots \\ \frac{\partial f}{\partial \mathbf{b}_R} \\ \frac{\partial f}{\partial \mathbf{c}_1} \\ \vdots \\ \frac{\partial f}{\partial \mathbf{c}_R} \\ \frac{\partial f}{\partial \mathbf{d}_1} \\ \vdots \\ \frac{\partial f}{\partial \mathbf{d}_R} \\ \frac{\partial f}{\partial \lambda} \\ \frac{\partial f}{\partial \sigma} \end{bmatrix}$$

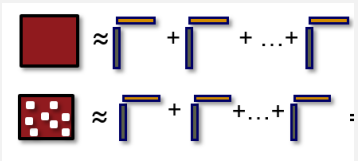
acmtf_opt

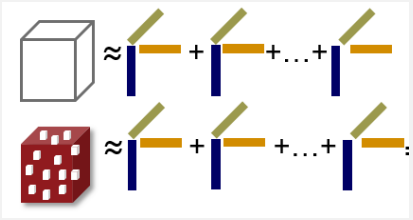
Poblano Toolbox
Gradient-based optimization algorithms

ncg, lbfgs, …

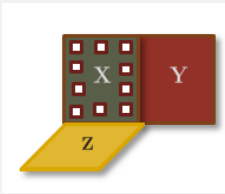
Models & Algorithms

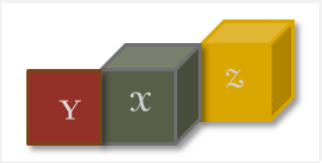
Matrix/Tensor factorizations





Data fusion





Applications

- **Recommender Systems**
- **Neuroscience**
- **Chemometrics/ Metabolomics**

For what we have not discussed (rank, uniqueness, more on algorithms and constraints, other tensor models, scalability, many more applications…)

A. Smilde, R. Bro, P. Geladi. Multi-way Analysis: Applications in the Chemical Sciences, Wiley, England, 2004

T. G. Kolda, B. W. Bader. Tensor Decompositions and Applications. *SIAM Review*, 51(3): 455-500, 2009

E. Acar, B. Yener. Unsupervised Multiway Data Analysis: A Literature Survey. *IEEE Transactions on Knowledge and Data Engineering*, 21(1):6-20, 2009

L. Grasedyck, D. Kressner, C. Tobler. A literature survey of low rank tensor approximation techniques. *GAMM-Mitteilungen*, 36: 53-78, 2013

P. Comon. Tensors: A Brief Introduction. *IEEE Signal Processing Magazine*, 31(3): 44-53, 2014

E. E. Papalexakis, C. Faloutsos, N. D. Sidiropoulos. Tensors for Data Mining and Data Fusion: Models, Applications, and Scalable Algorithms. *ACM Transactions on Intelligent Systems and Technology*, 8(2), Article 16, 2016

N. Sidiropoulos, L. De Lathauwer, X. Fu, K. Huang, E. E. Papalexakis, C. Faloutsos. Tensor Decomposition for Signal Processing and Machine Learning. *IEEE Transactions on Signal Processing*, 65(13): 3551-3582, 2017

Acknowledgments

CP-OPT
CP-WOPT
CMTF-OPT

Tamara G. Kolda, Sandia National Labs, CA

Daniel M. Dunlavy, Sandia National Labs, NM

Structure-revealing CMTF and chemometrics/metabolomics applications

Rasmus Bro, University of Copenhagen, Denmark

Age K. Smilde, University of Amsterdam, The Netherlands

Morten A. Rasmussen, University of Copenhagen, Denmark

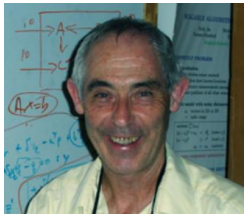
Using structure-revealing CMTF for fusion of multi-modal neuroimaging data

Tülay Adalı, University of Maryland Baltimore County, MD

Link prediction using CMTF with different loss functions and tensor models

Taylan Cemgil, Bogazici University, Turkey

Beyza Ermis, Bogazici University, Turkey



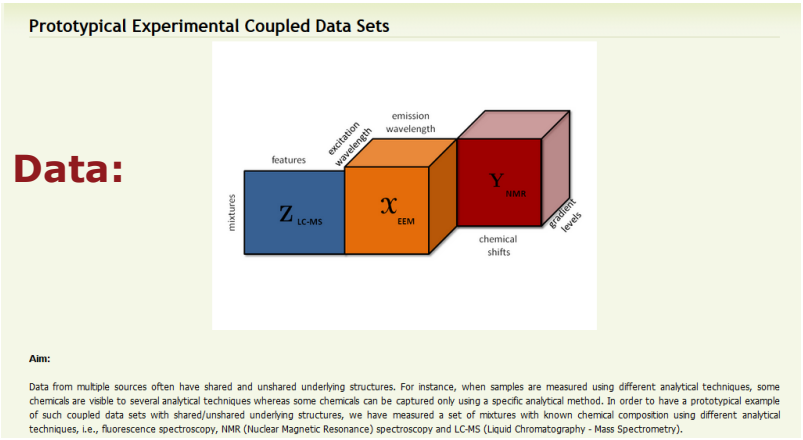
Structure-revealing CMTF using SNOPT

Michael Saunders, Stanford University, CA

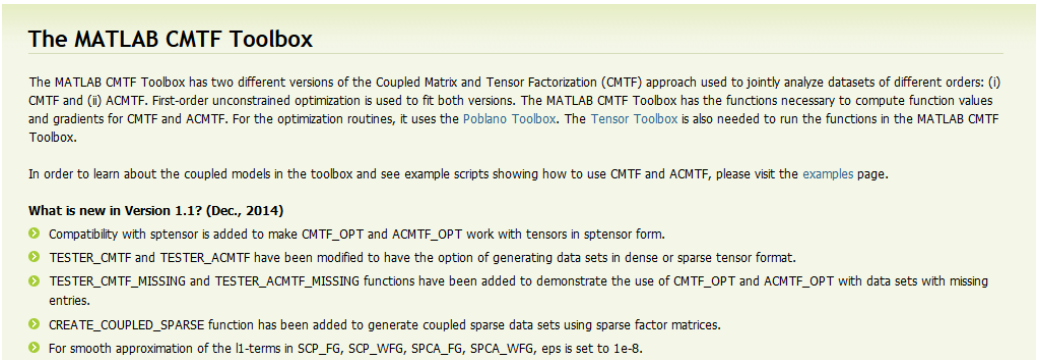
JODA: Joint Data Analysis for Enhanced Knowledge Discovery

www.models.life.ku.dk/joda

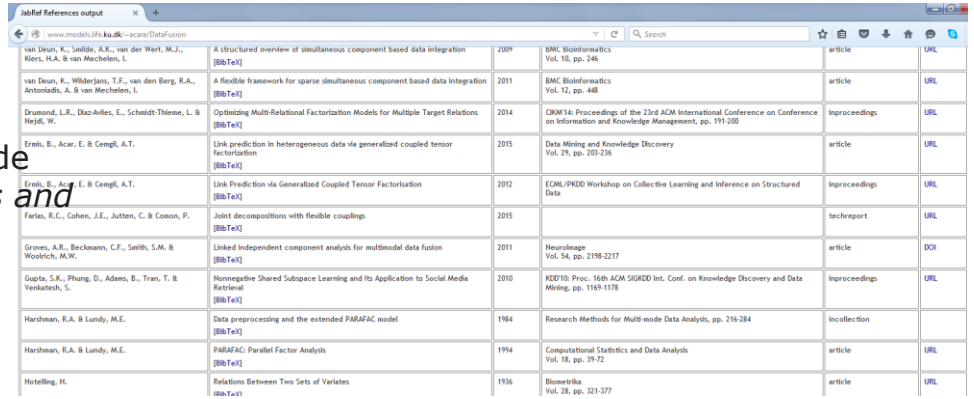
Data:



MATLAB CMTF Toolbox:



Literature on Coupled factorizations:
Over 120 papers as of June 2018



I. Van Mechelen, A. K. Smilde. A generic linked-mode decomposition model for data fusion. *Chemometrics and Intelligent Laboratory Systems*, 104: 83-94, 2010

E. Acar, R. Bro, A. K. Smilde. Data Fusion in Metabolomics based on Coupled Matrix and Tensor Factorizations. *Proceedings of the IEEE*, 103: 1602-1620, 2015

D. Lahat, T. Adalı, C. Jutten. Multimodal Data Fusion: An Overview of Methods, Challenges, and Prospects. *Proceedings of the IEEE*, 103: 1449-1477, 2015

Evrim Acar
evrim@simula.no
www.models.life.ku.dk/~acare

simula