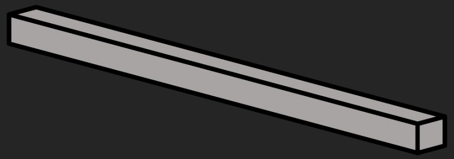
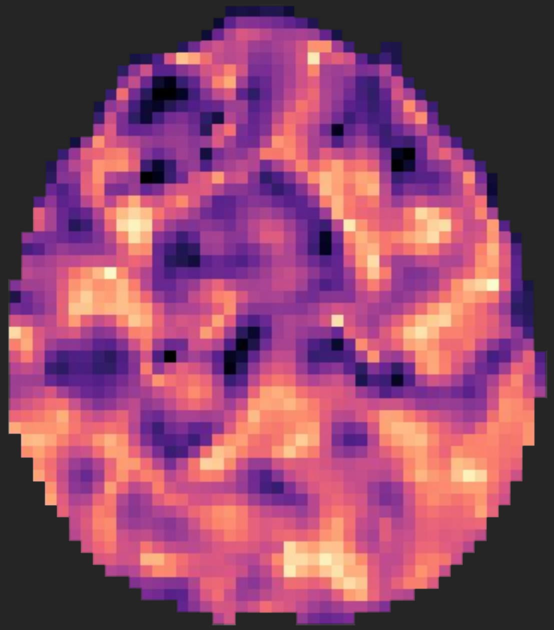
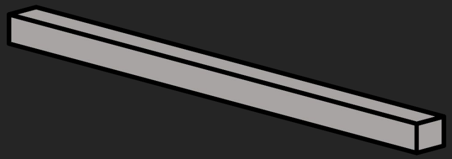
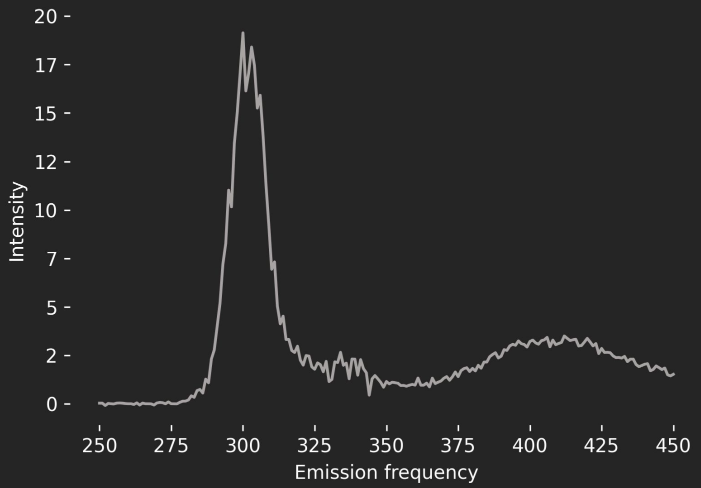
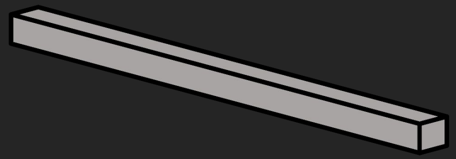


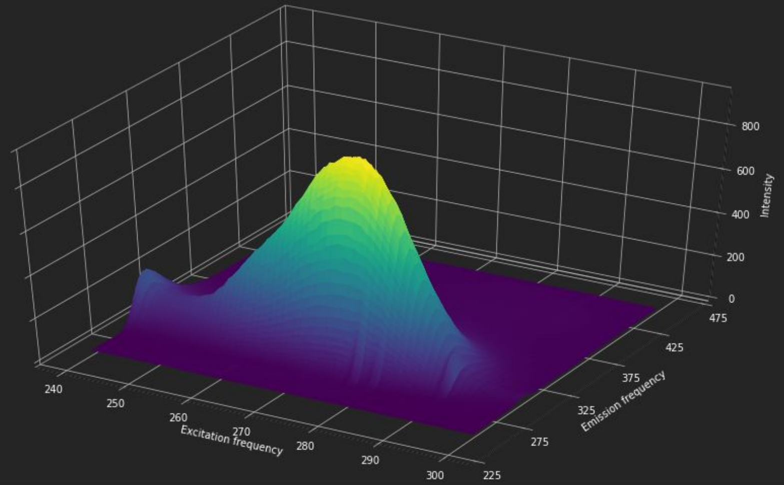


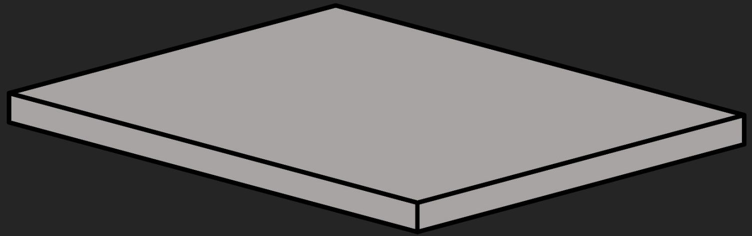


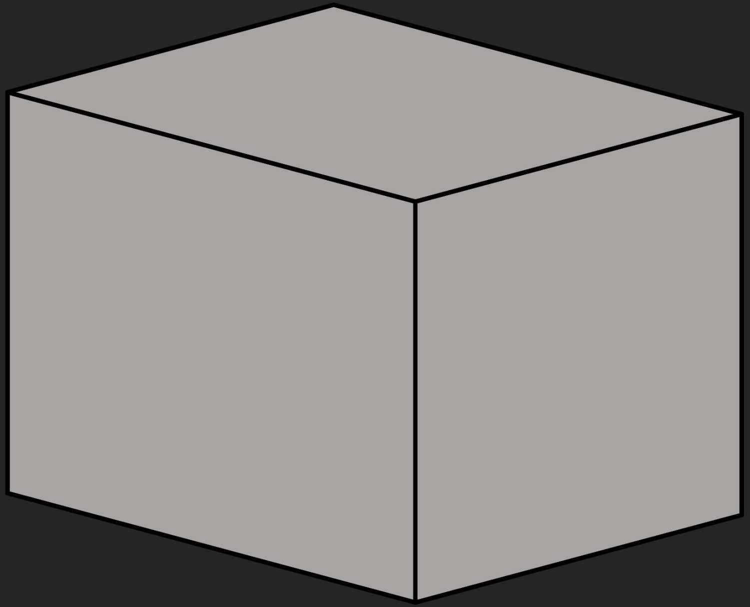


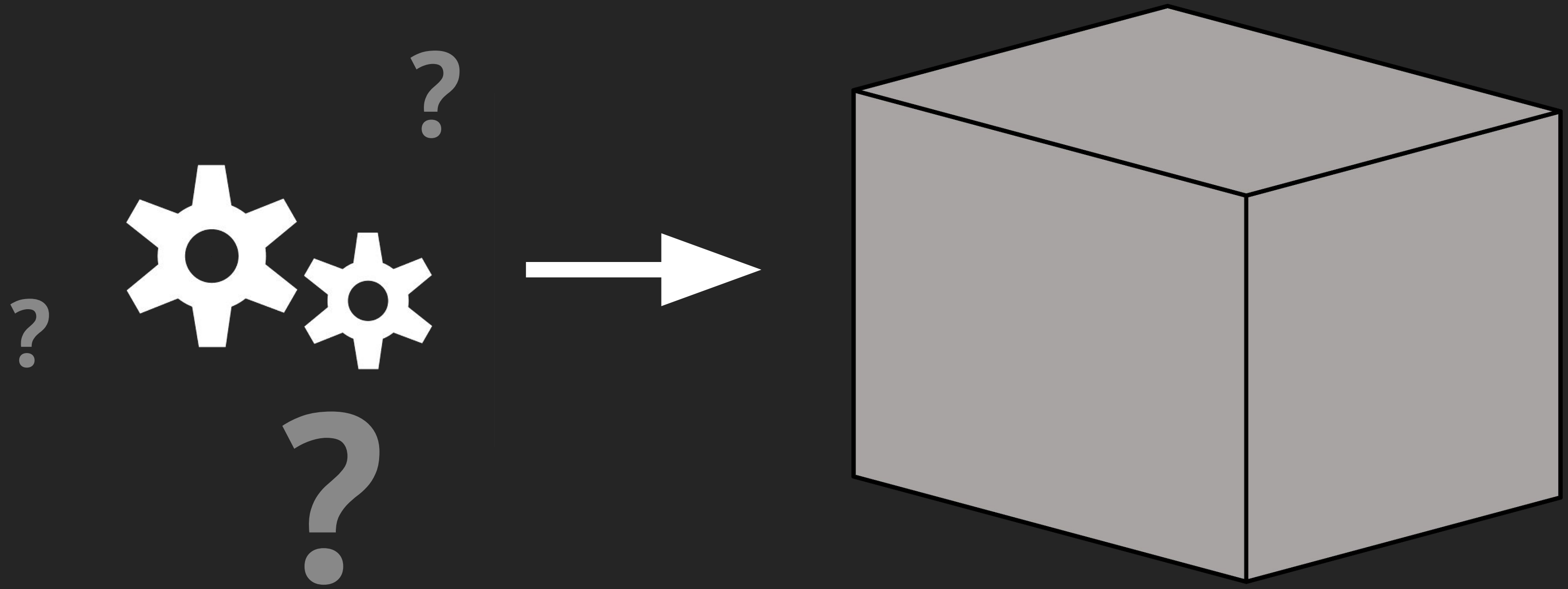
Intensity
Emission frequency
20
17
15
12
10
7
5
2
0
250
275
300
325
350
375
400
425
450

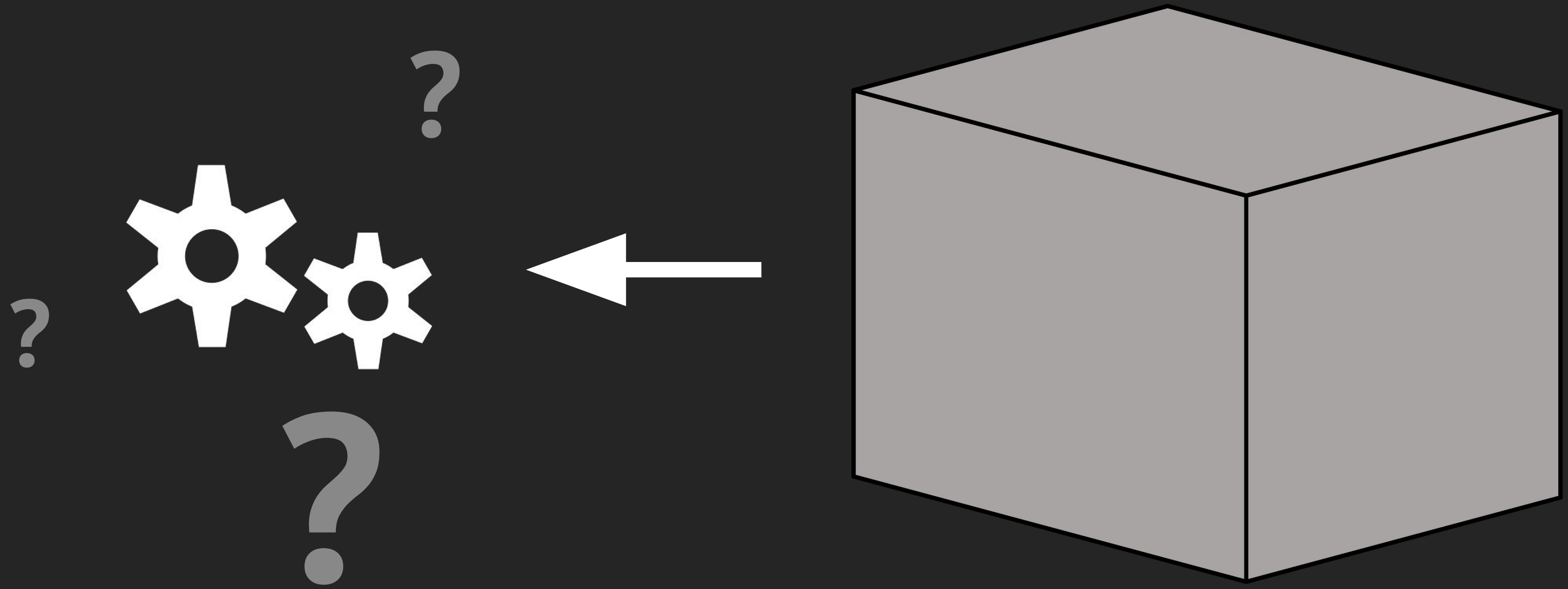



800
600
400
200
0
Intensity
475
425
375
325
275
225
Emission frequency
240
250
260
270
280
290
300
Excitation frequency

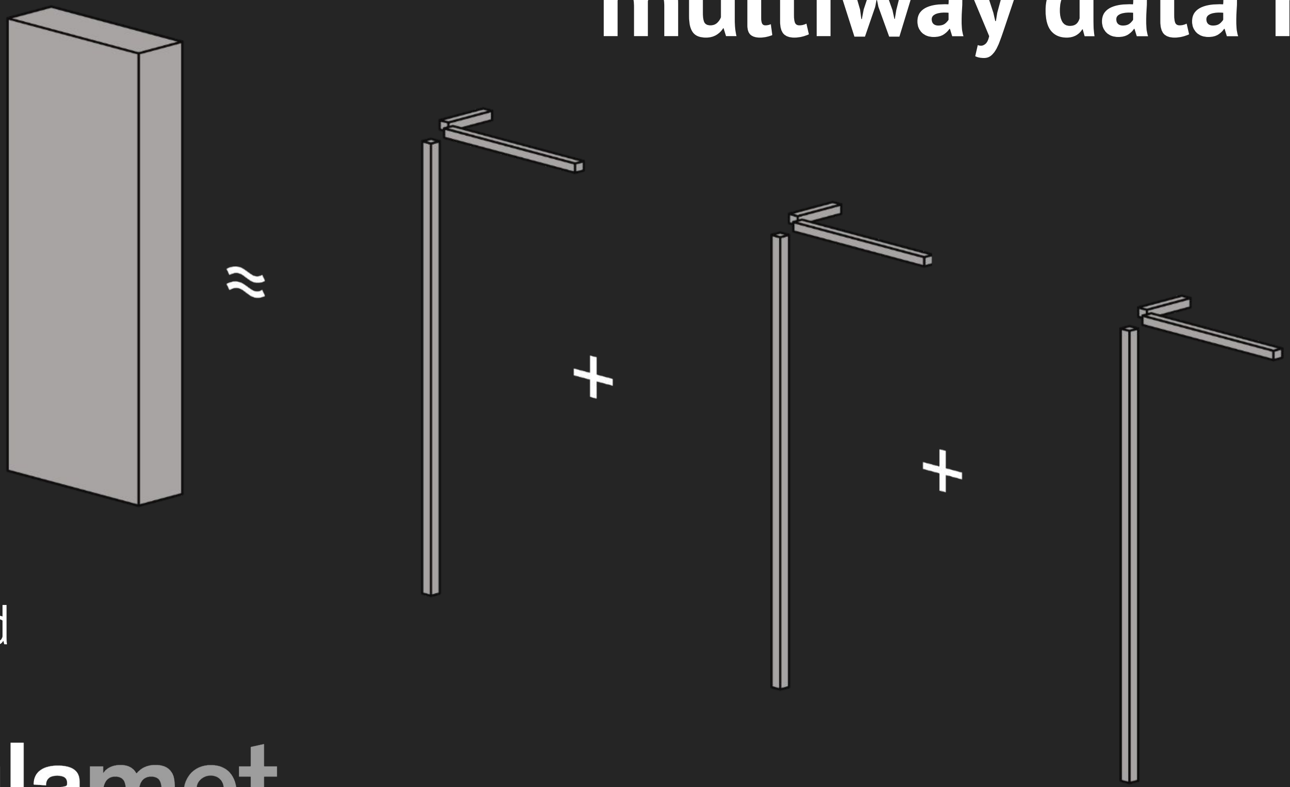






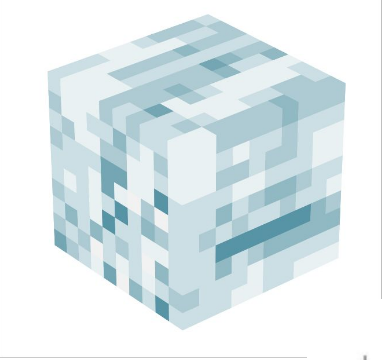


Tensor decomposition for multiway data mining

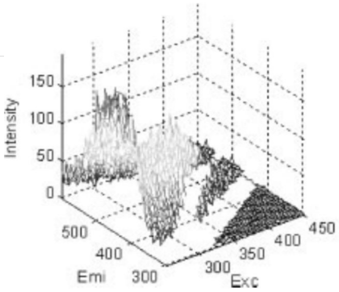


Marie Roald
20.10.21

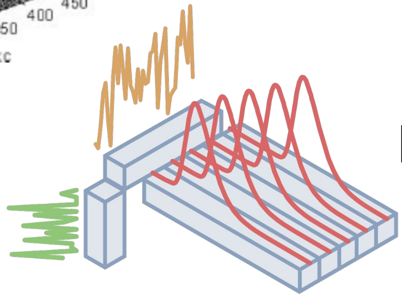
simulamet



Matrix and tensor decomposition

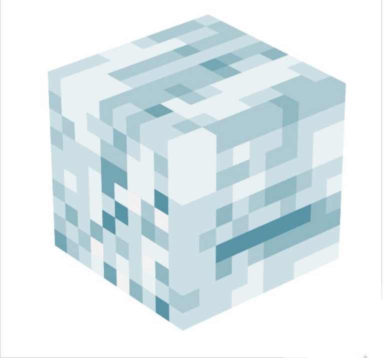


Applications of PARAFAC

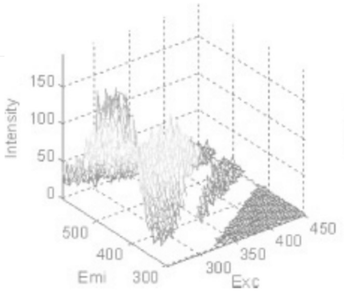


My research and PARAFAC2

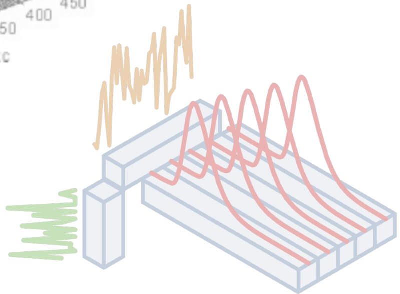
Code demonstration



Matrix and tensor decomposition



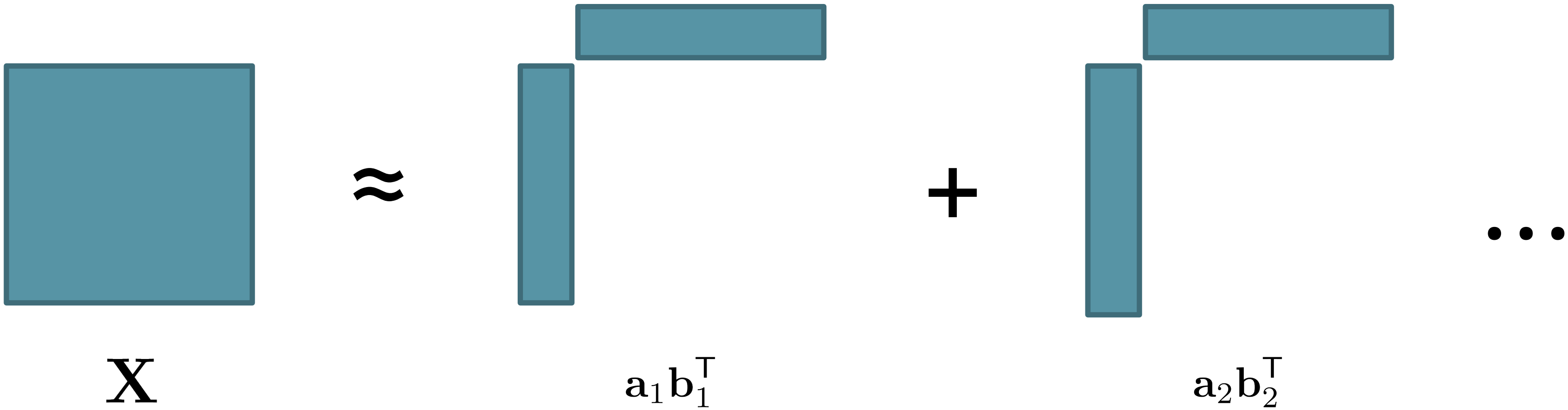
Applications of PARAFAC



My research and PARAFAC2

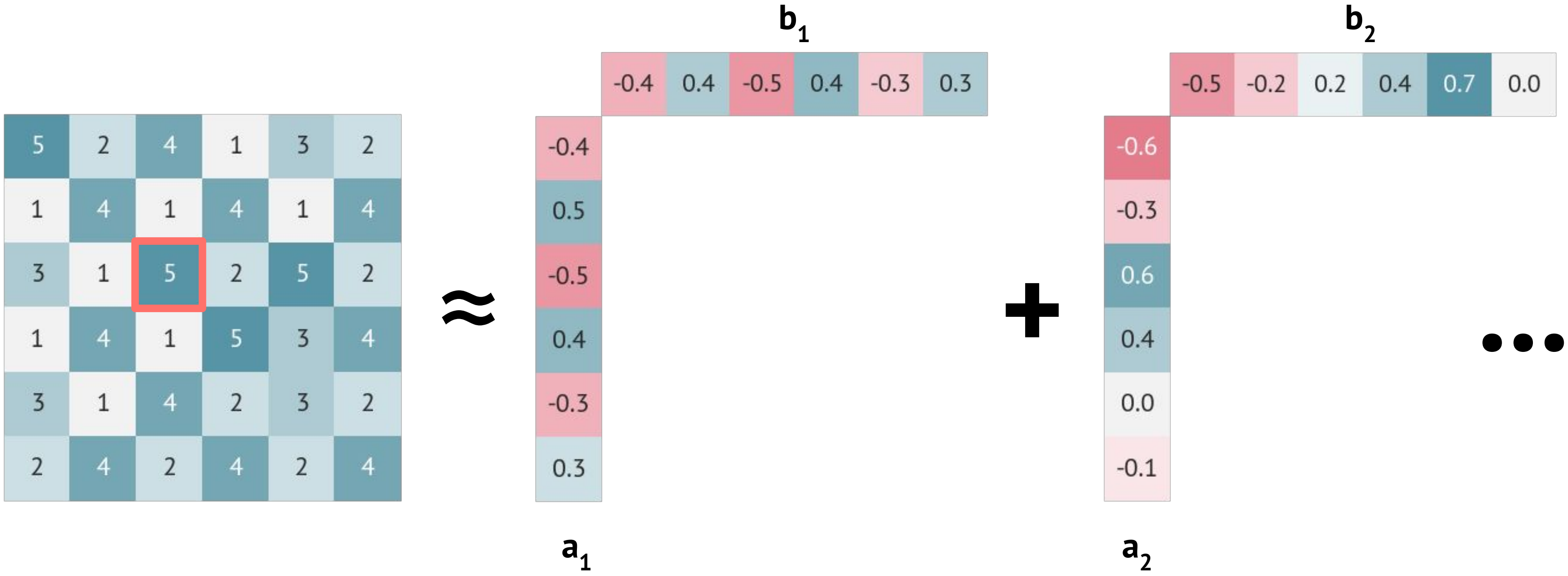
Code demonstration

Low rank matrix factorisation methods decompose a matrix into a sum of low rank components

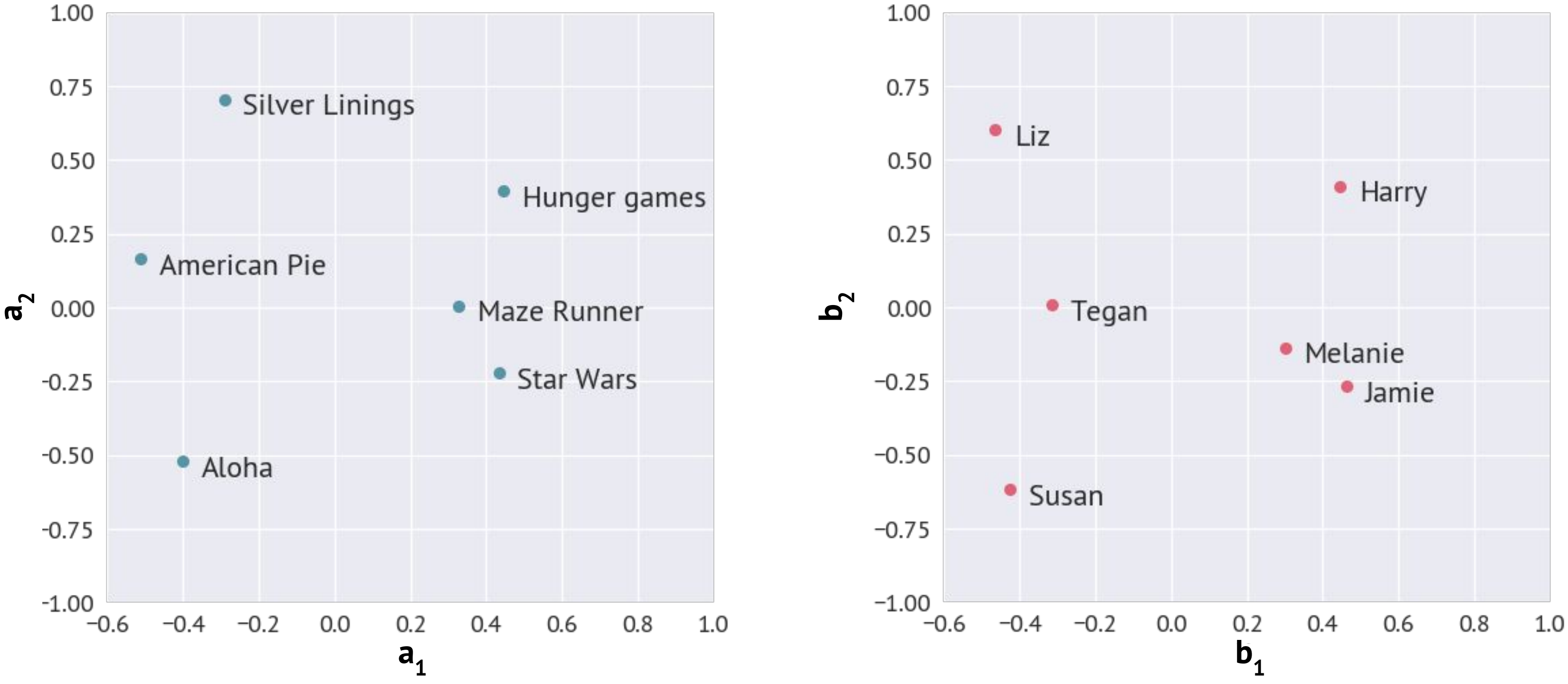


$$\mathbf{X} \approx \mathbf{A}\mathbf{B}^\top$$

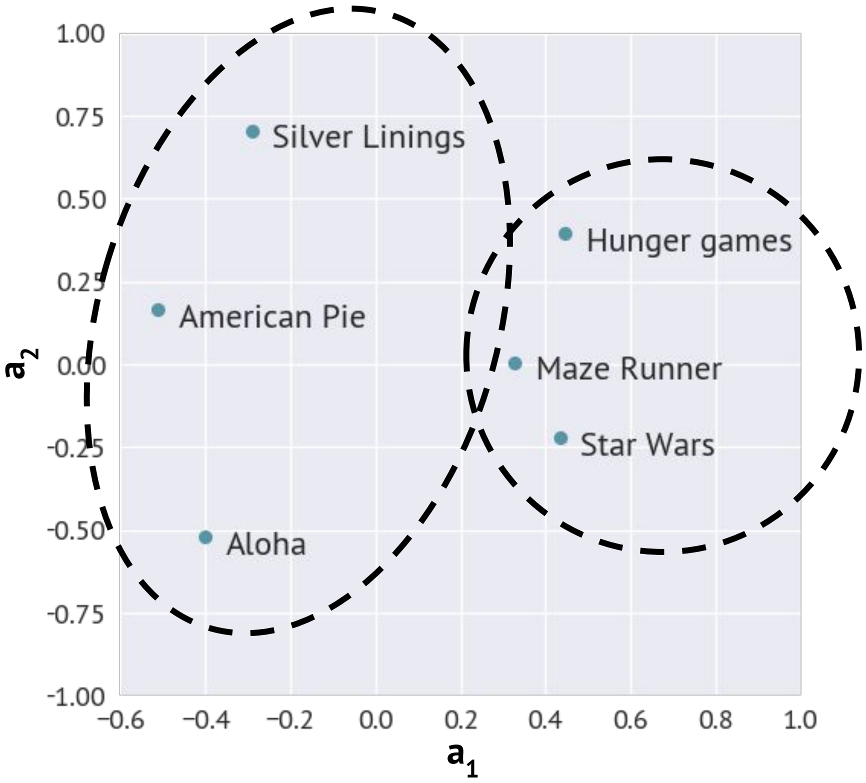
Low rank matrix factorisation methods decompose a matrix into a sum of low rank components

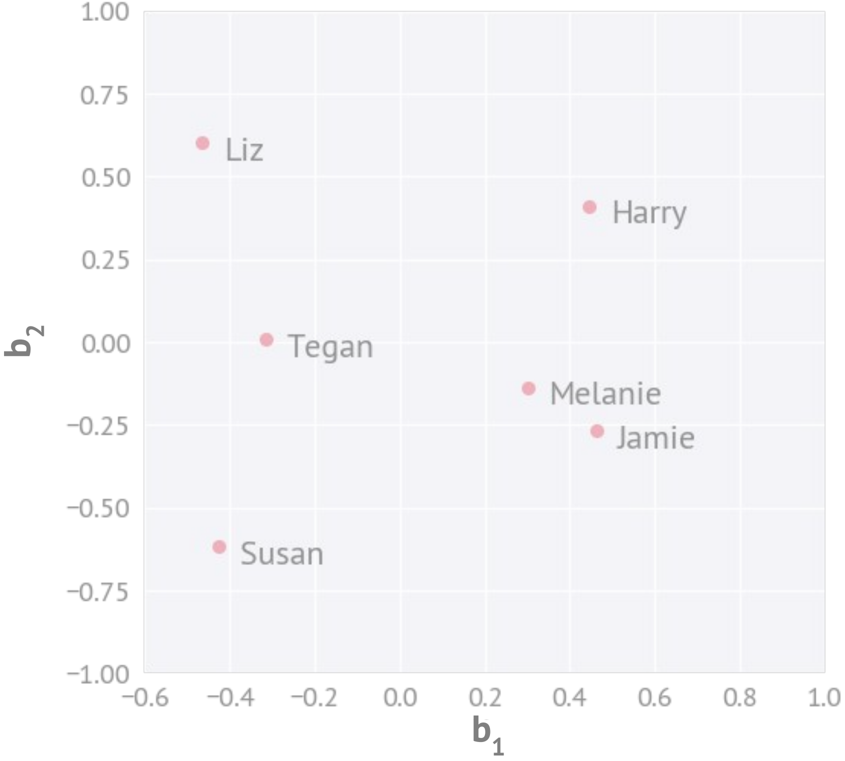


The components can give meaningful information about the underlying structure of the data

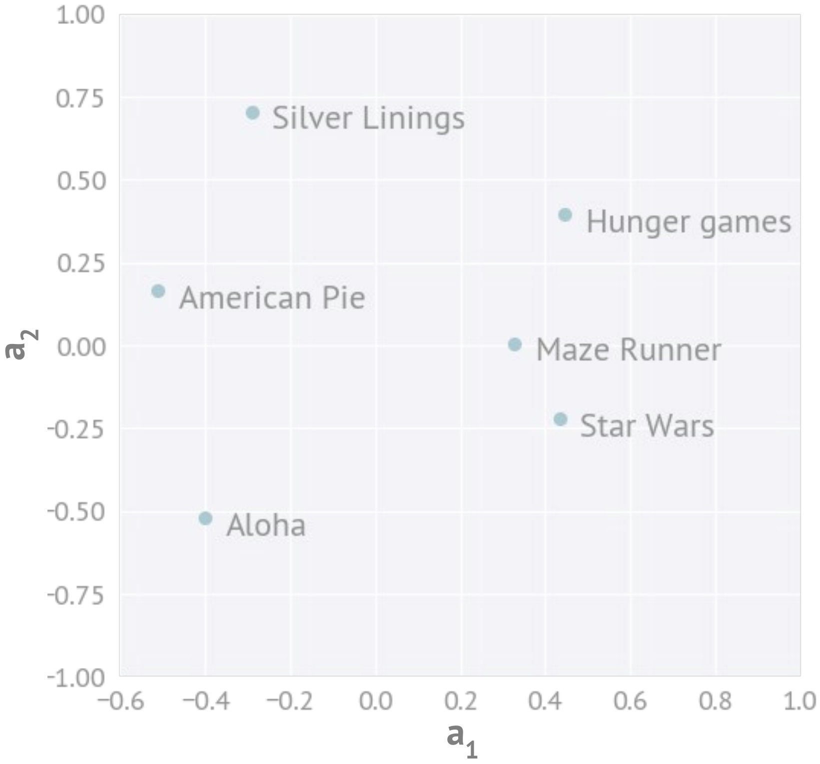


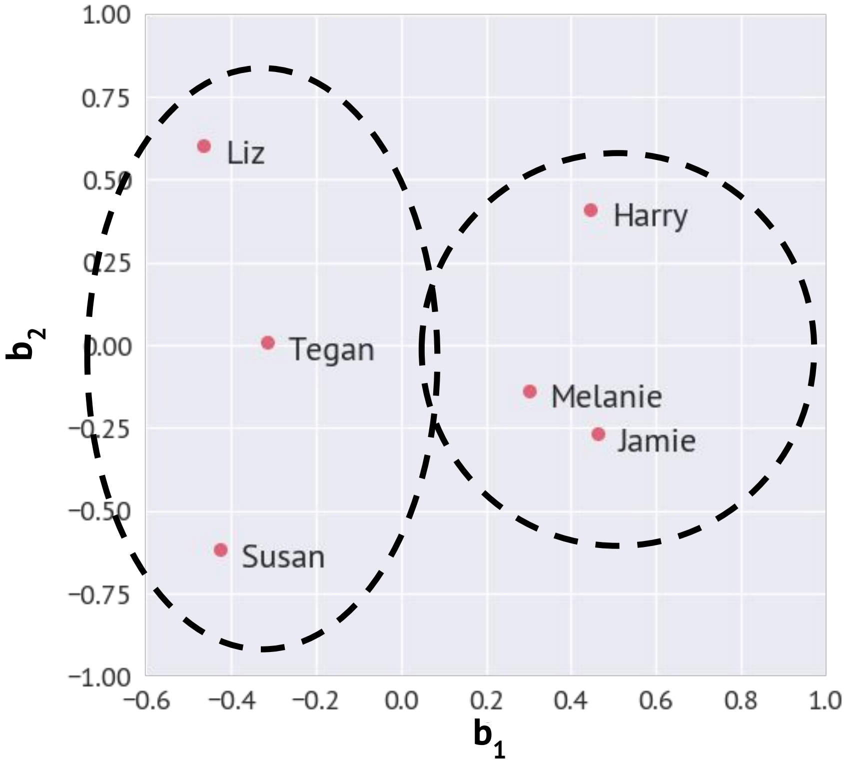
Components for the movie-mode can reveal movie genres





Components for the user-mode can identify networks of users with similar taste in movies





Matrix factorisation is not unique

One potential matrix factorisation

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

Matrix factorisation is not unique

One potential matrix factorisation $$\mathbf{X} = \mathbf{A}\mathbf{B}^\intercal$$

Multiply with identity matrix $$\mathbf{X} = \mathbf{A}(\mathbf{M}\mathbf{M}^{-1})\mathbf{B}^\intercal$$

Matrix factorisation is not unique

One potential matrix factorisation $\mathbf{X} = \mathbf{A}\mathbf{B}^{\mathsf{T}}$

Multiply with identity matrix $\mathbf{X} = \mathbf{A}(\mathbf{M}\mathbf{M}^{-1})\mathbf{B}^{\mathsf{T}}$

Reorder using transposition rules $\mathbf{X} = (\mathbf{A}\mathbf{M})(\mathbf{B}\mathbf{M}^{-\mathsf{T}})^{\mathsf{T}}$

Matrix factorisation is not unique

One potential matrix factorisation $\mathbf{X} = \mathbf{A}\mathbf{B}^\intercal$

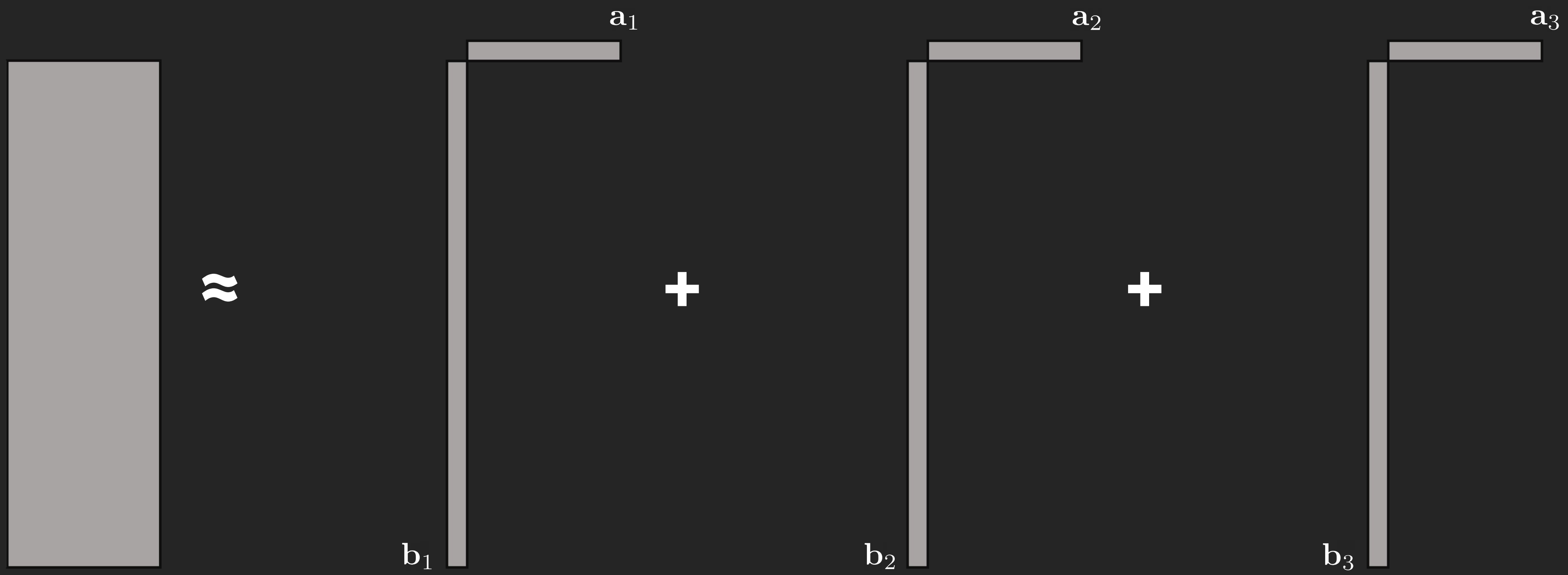
Multiply with identity matrix $\mathbf{X} = \mathbf{A}(\mathbf{M}\mathbf{M}^{-1})\mathbf{B}^\intercal$

Reorder using transposition rules $\mathbf{X} = (\mathbf{A}\mathbf{M})(\mathbf{B}\mathbf{M}^{-\intercal})^\intercal$

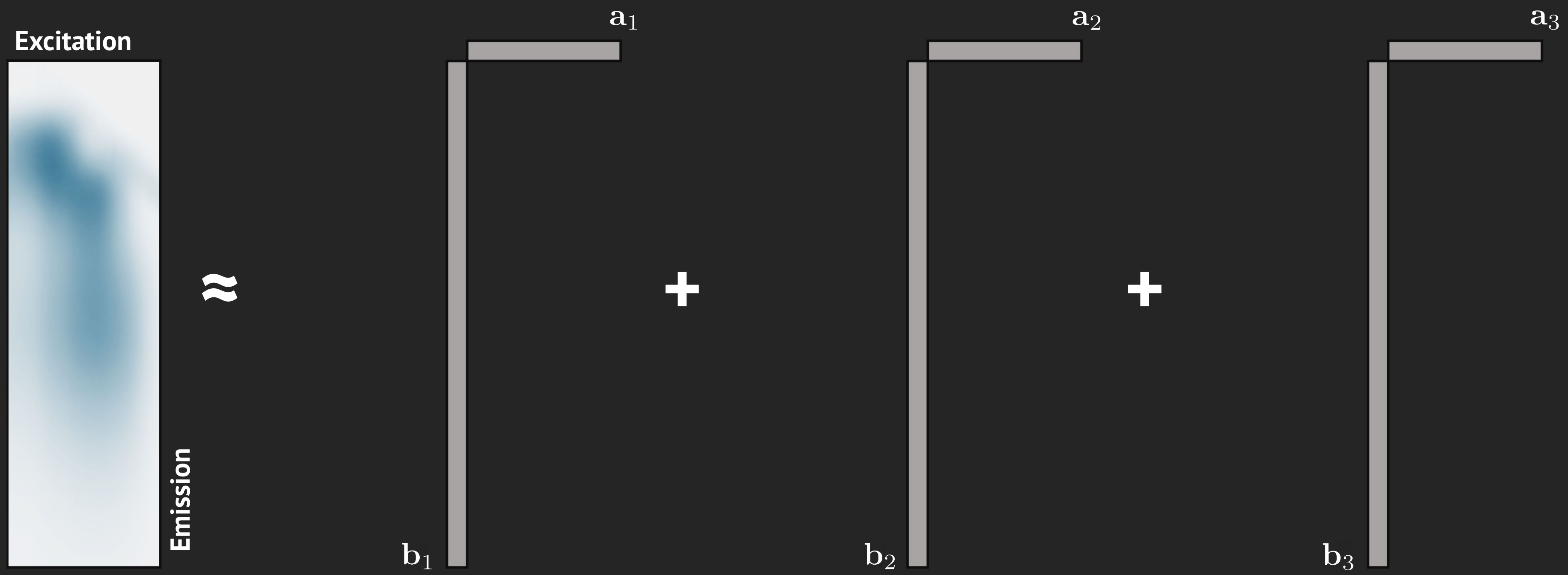
We obtain transformed components $\mathbf{X} = \tilde{\mathbf{A}}\tilde{\mathbf{B}}^\intercal$

We can get uniqueness by imposing orthogonality as in PCA

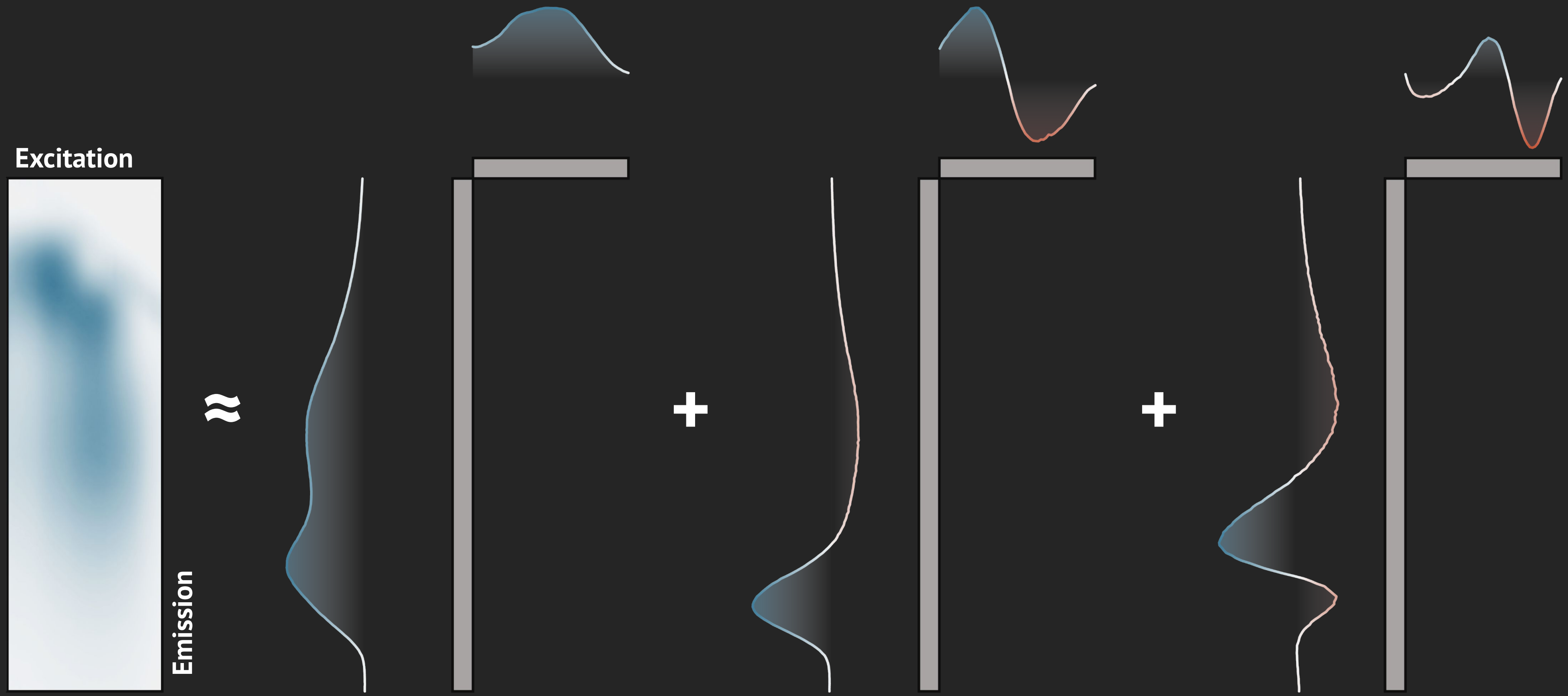
$$\mathbf{A}^\top \mathbf{A} = \mathbf{I} \qquad \mathbf{B}^\top \mathbf{B} = \mathbf{I}$$



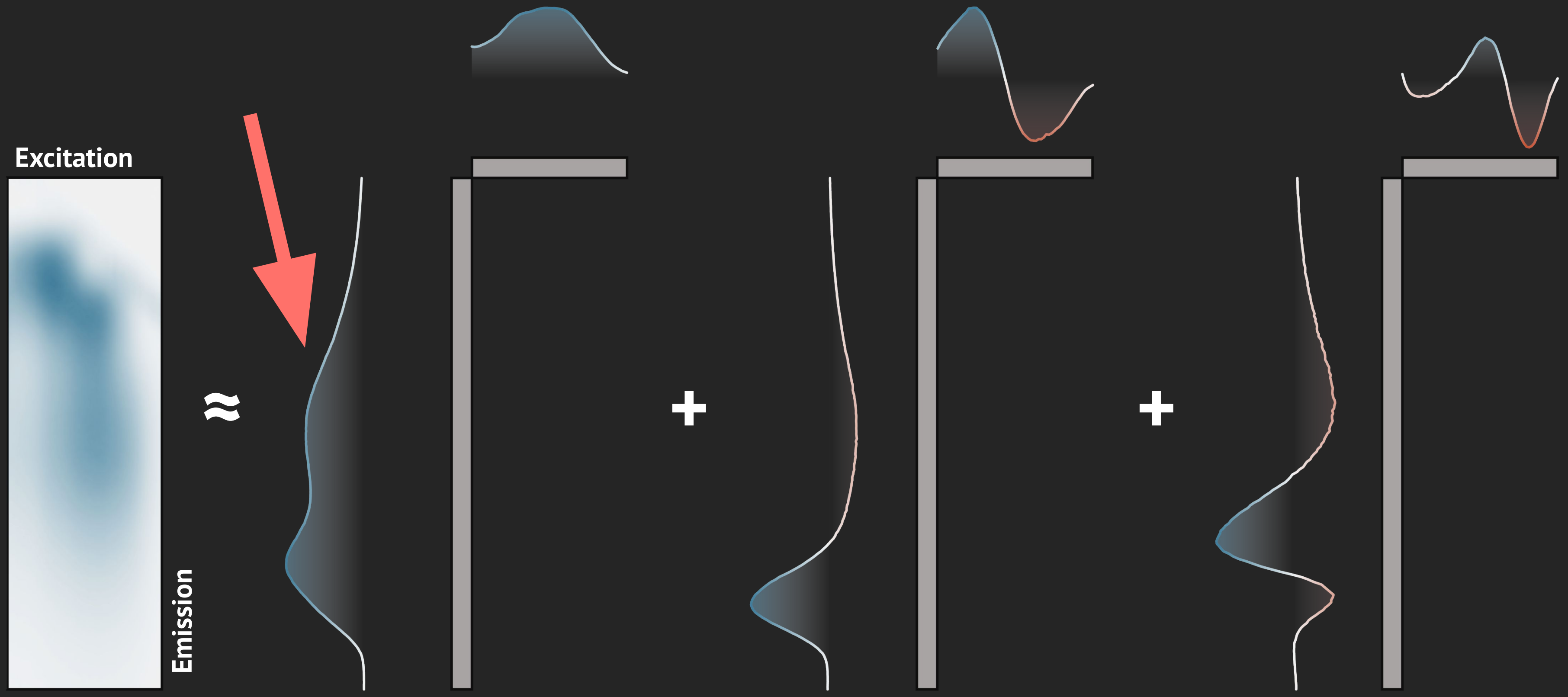
We can get uniqueness by imposing orthogonality as in PCA



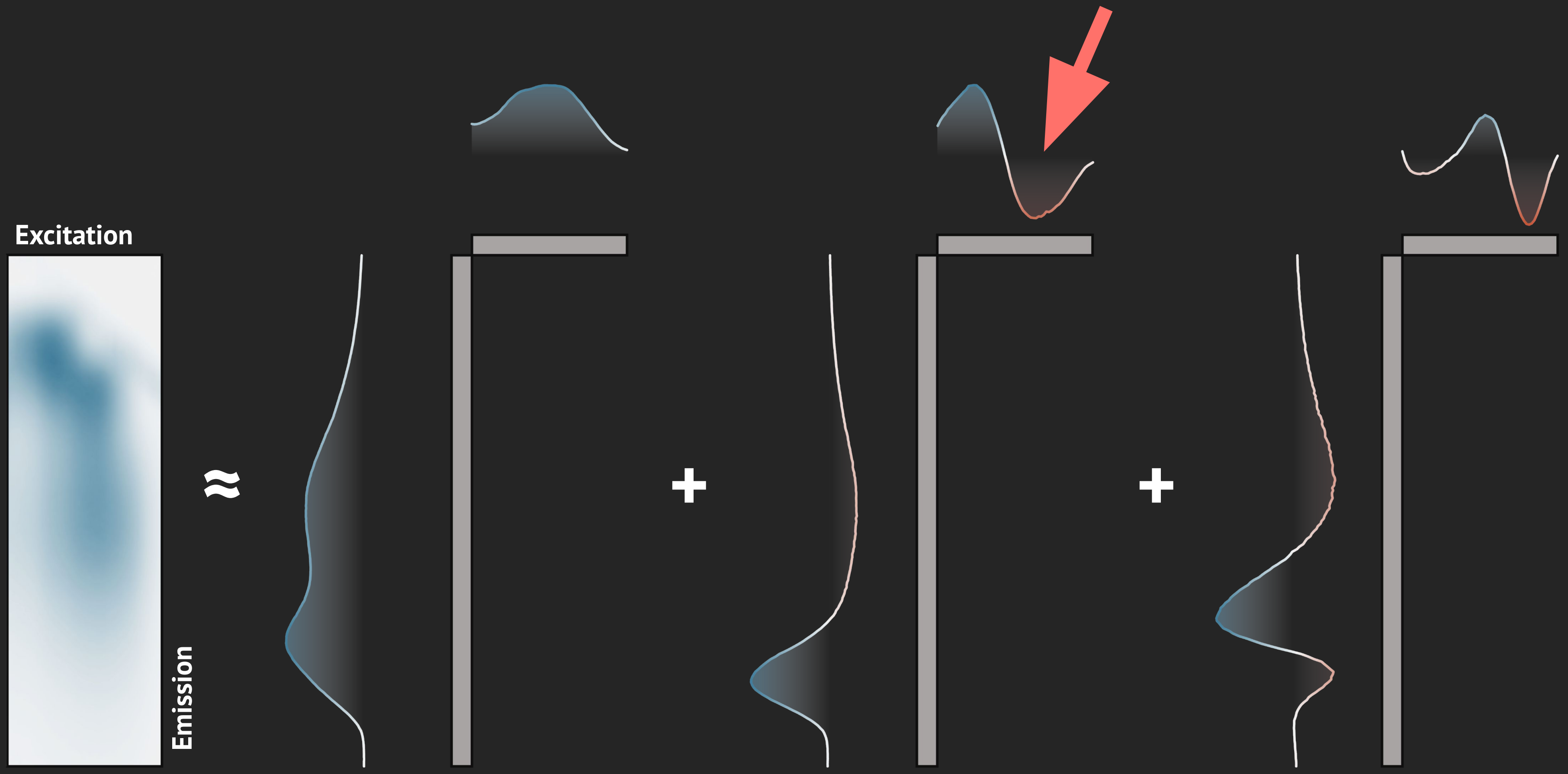
However these components are not necessarily meaningful



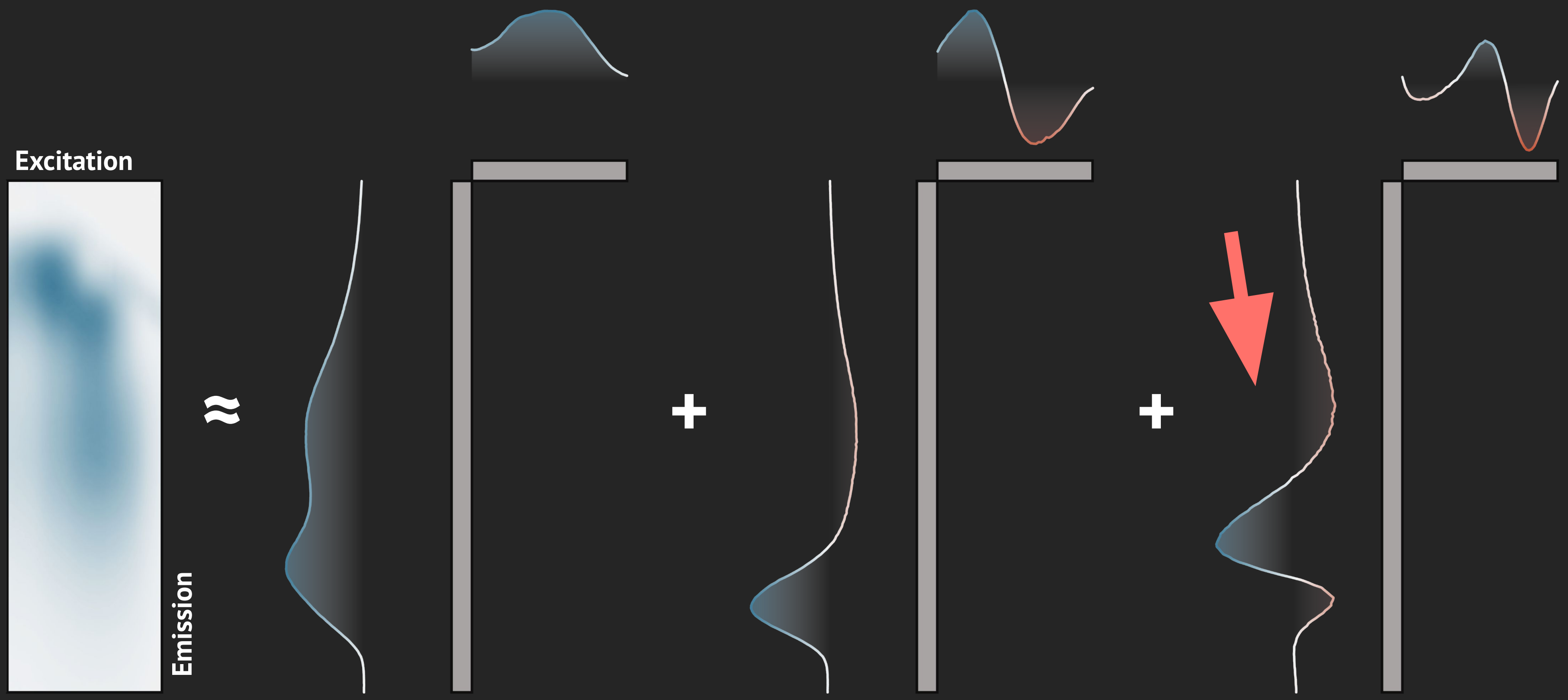
However these components are not necessarily meaningful



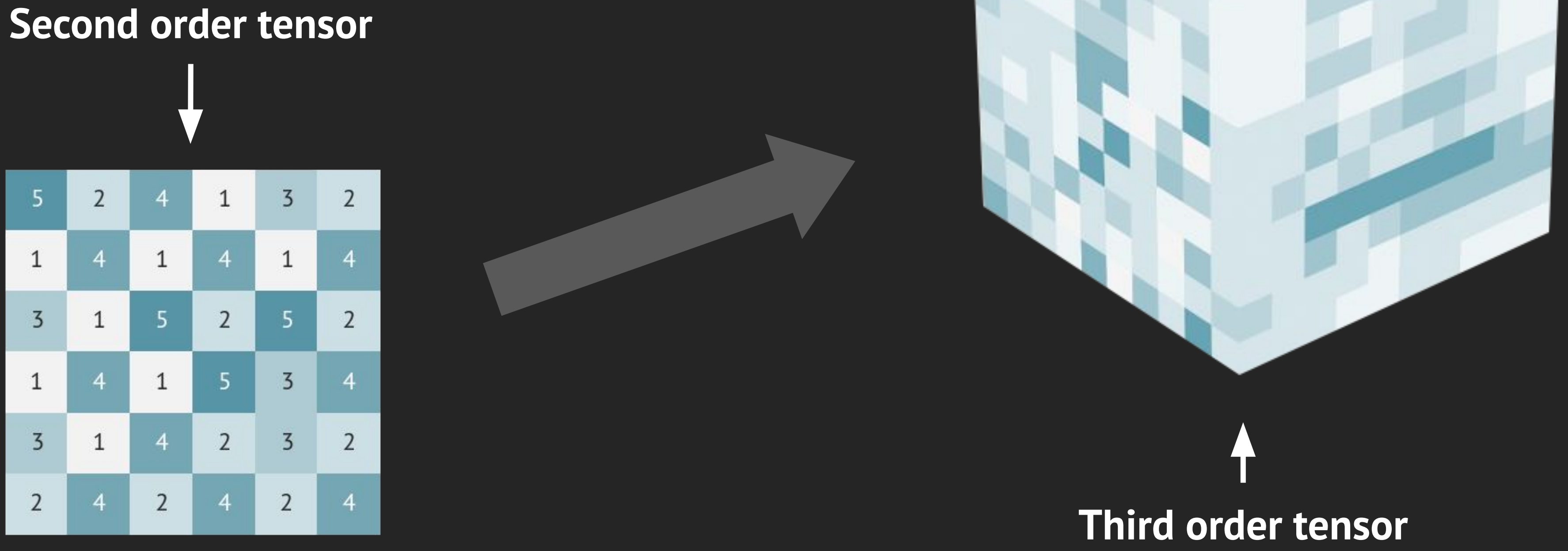
However these components are not necessarily meaningful



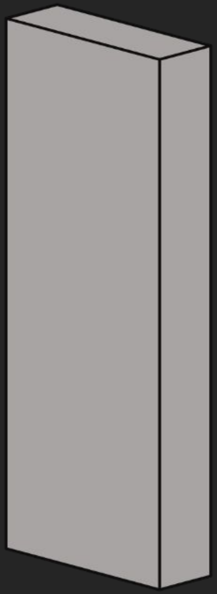
However these components are not necessarily meaningful



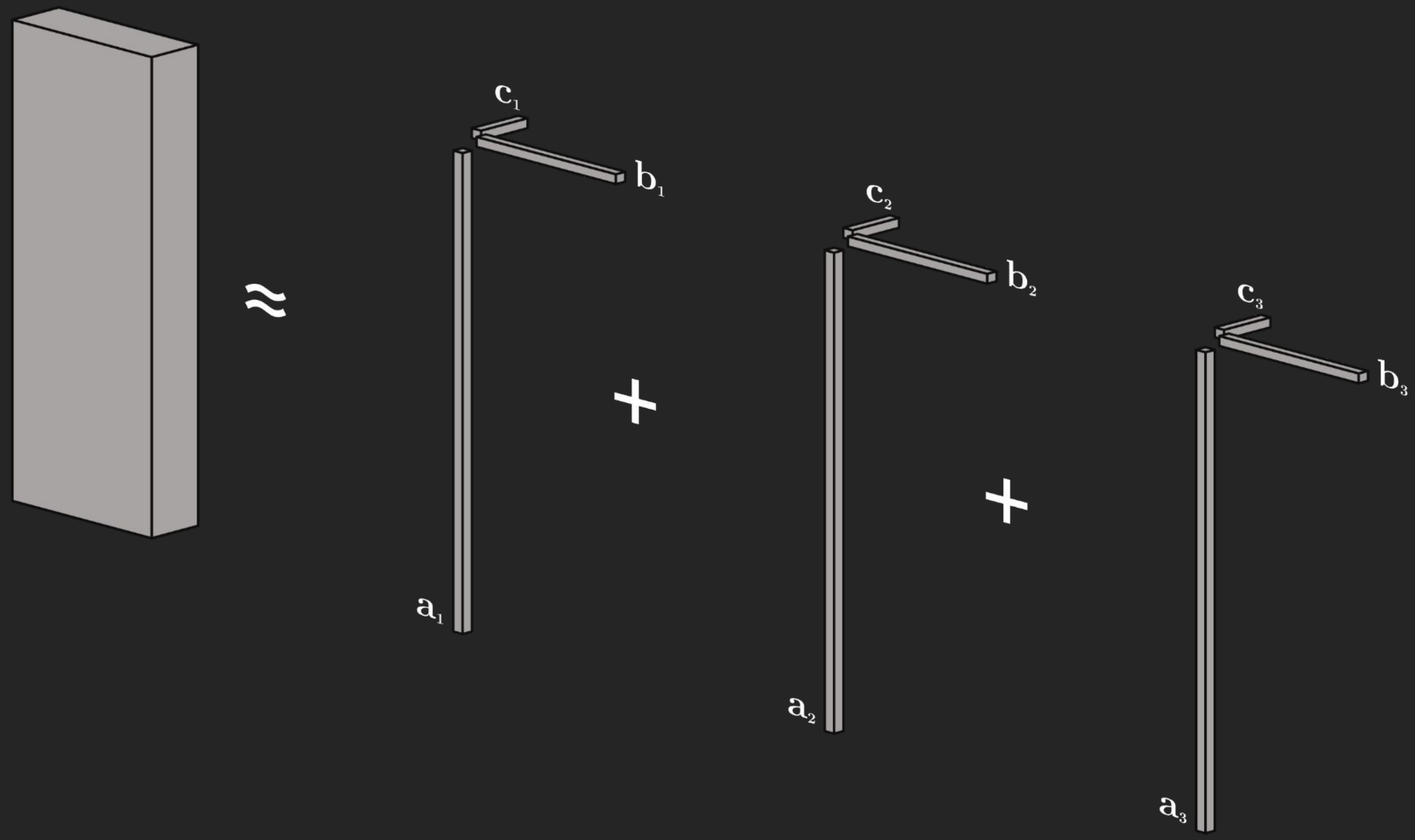
A tensor is a *higher-order* generalisation of a matrix



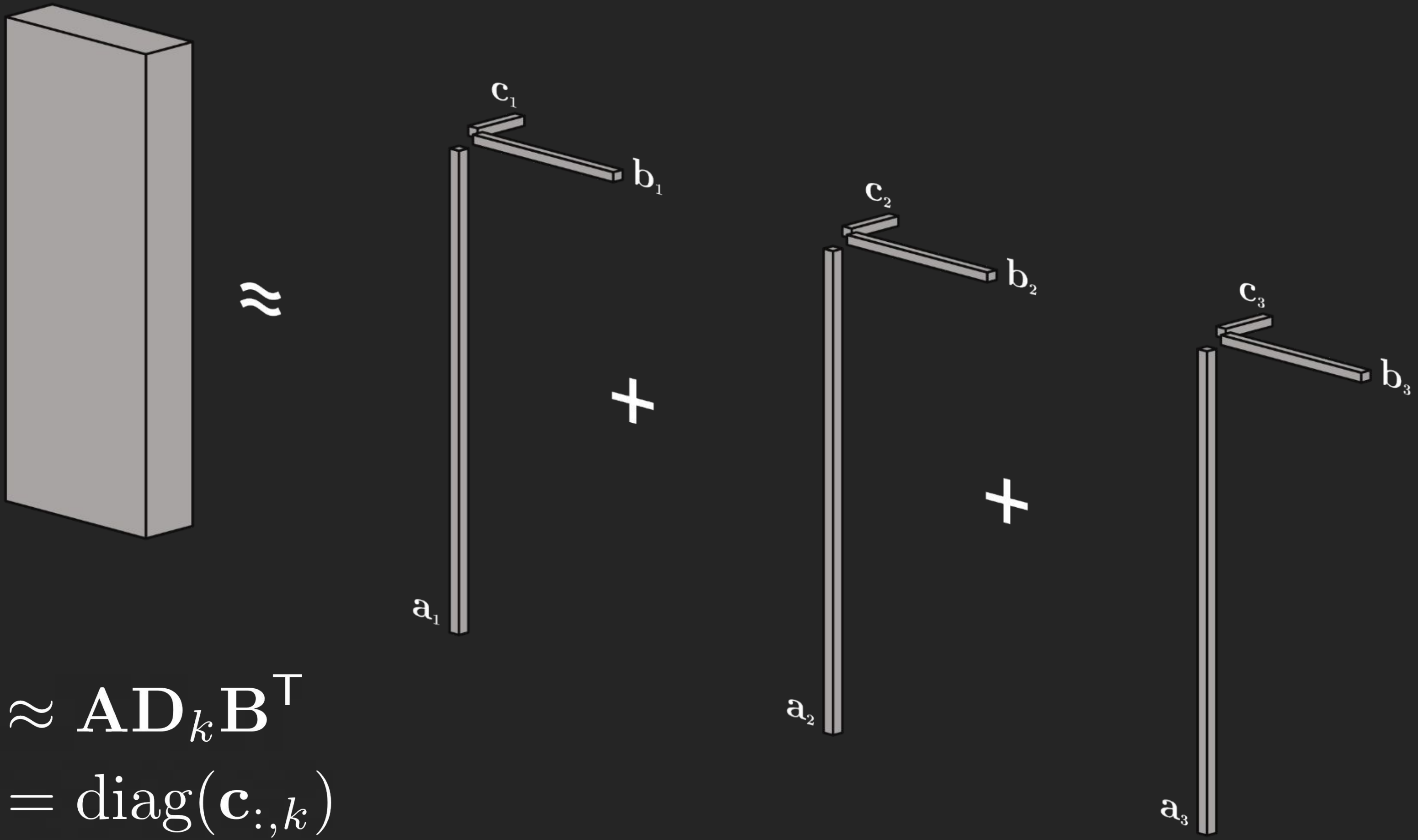
PARAFAC extends matrix decompositions to tensor-data



PARAFAC extends matrix decompositions to tensor-data

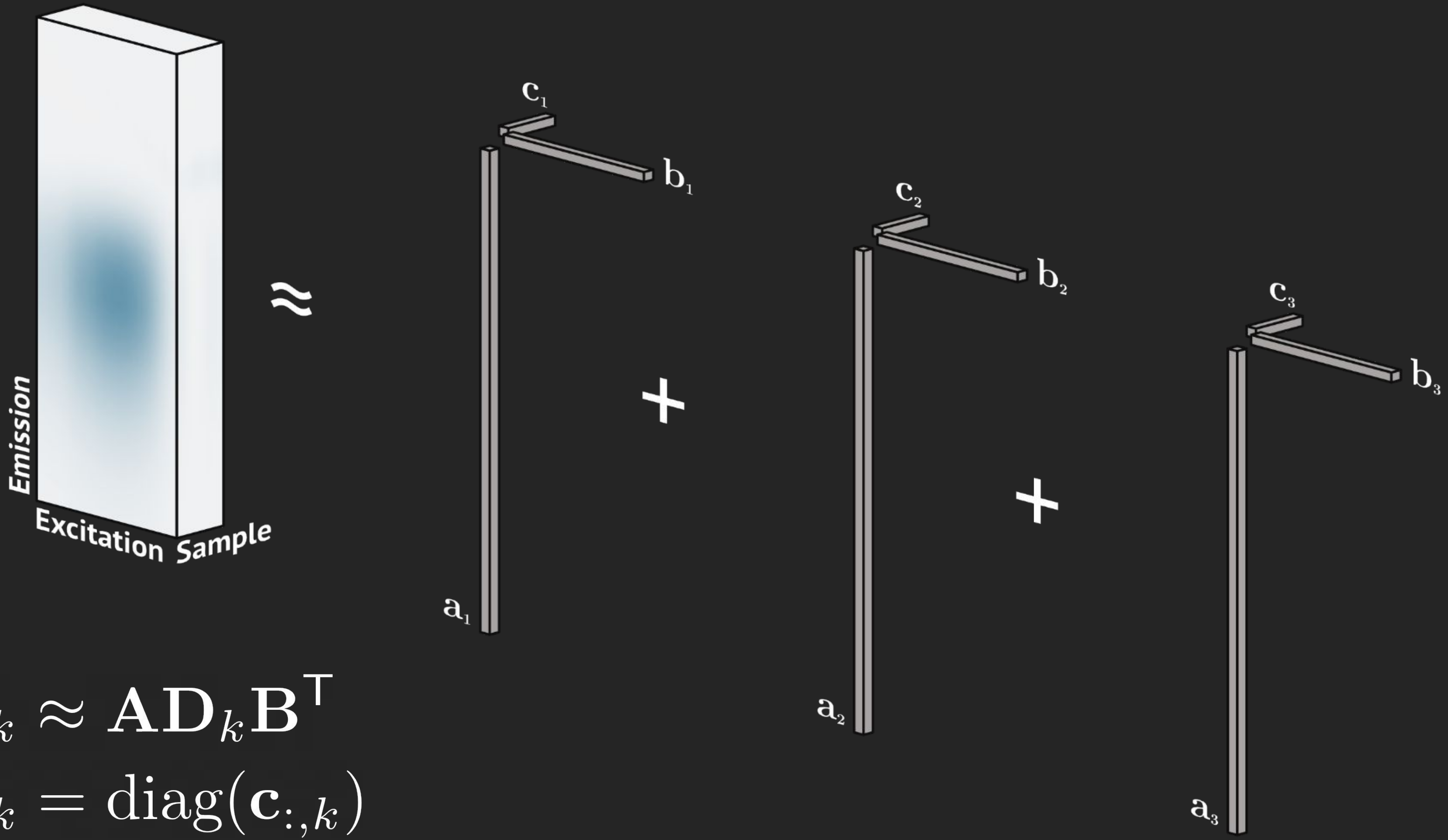


PARAFAC extends matrix decompositions to tensor-data



$$\mathbf{X}_k \approx \mathbf{A}\mathbf{D}_k\mathbf{B}^\top$$
$$\mathbf{D}_k = \mathrm{diag}(\mathbf{c}_{:,k})$$

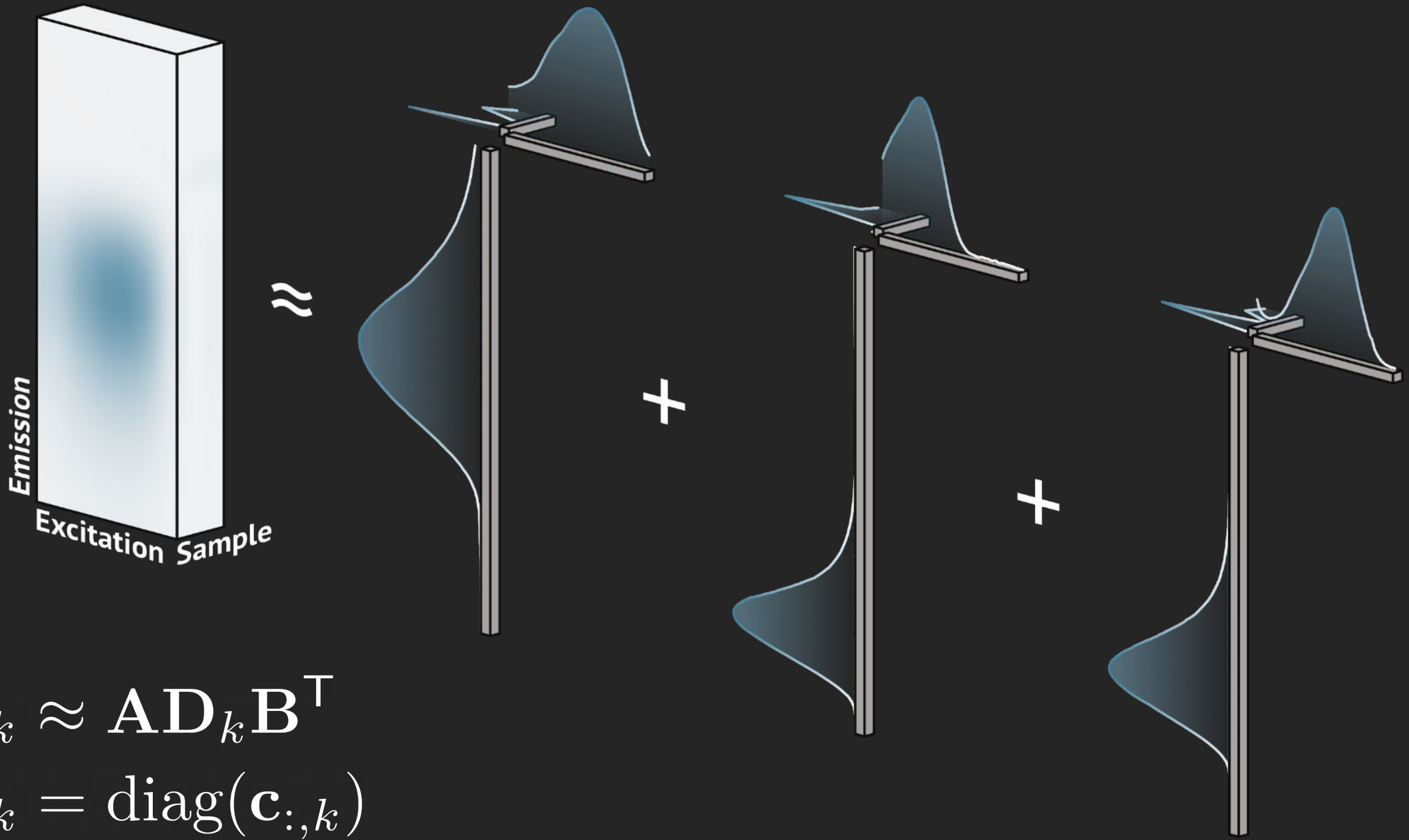
With PARAFAC, we find the EEM-spectra underlying the data



$$\mathbf{X}_k \approx \mathbf{A}\mathbf{D}_k\mathbf{B}^\top$$

$$\mathbf{D}_k = \mathrm{diag}(\mathbf{c}_{:,k})$$

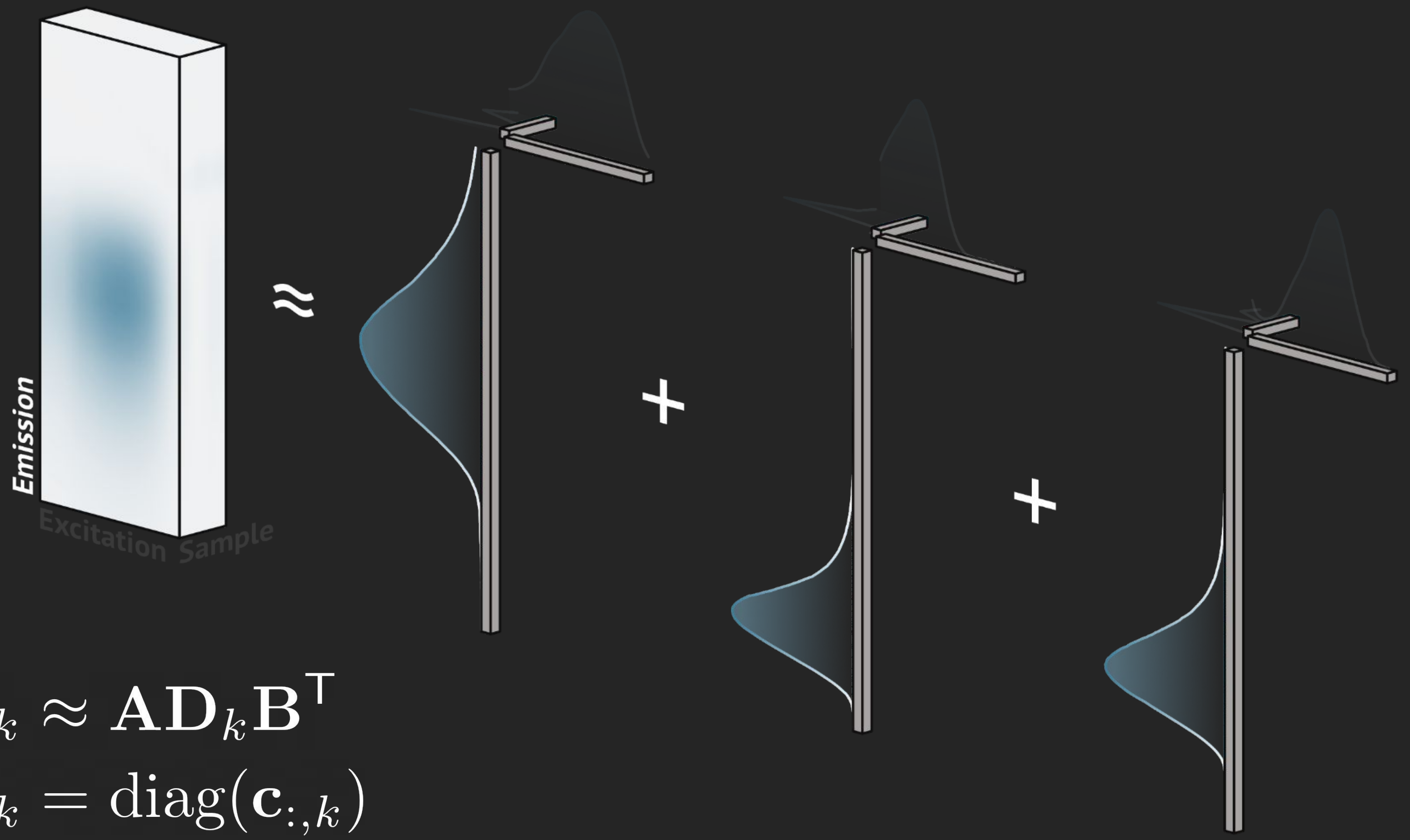
With PARAFAC, we find the EEM-spectra underlying the data



$$\mathbf{X}_k \approx \mathbf{A}\mathbf{D}_k\mathbf{B}^\top$$

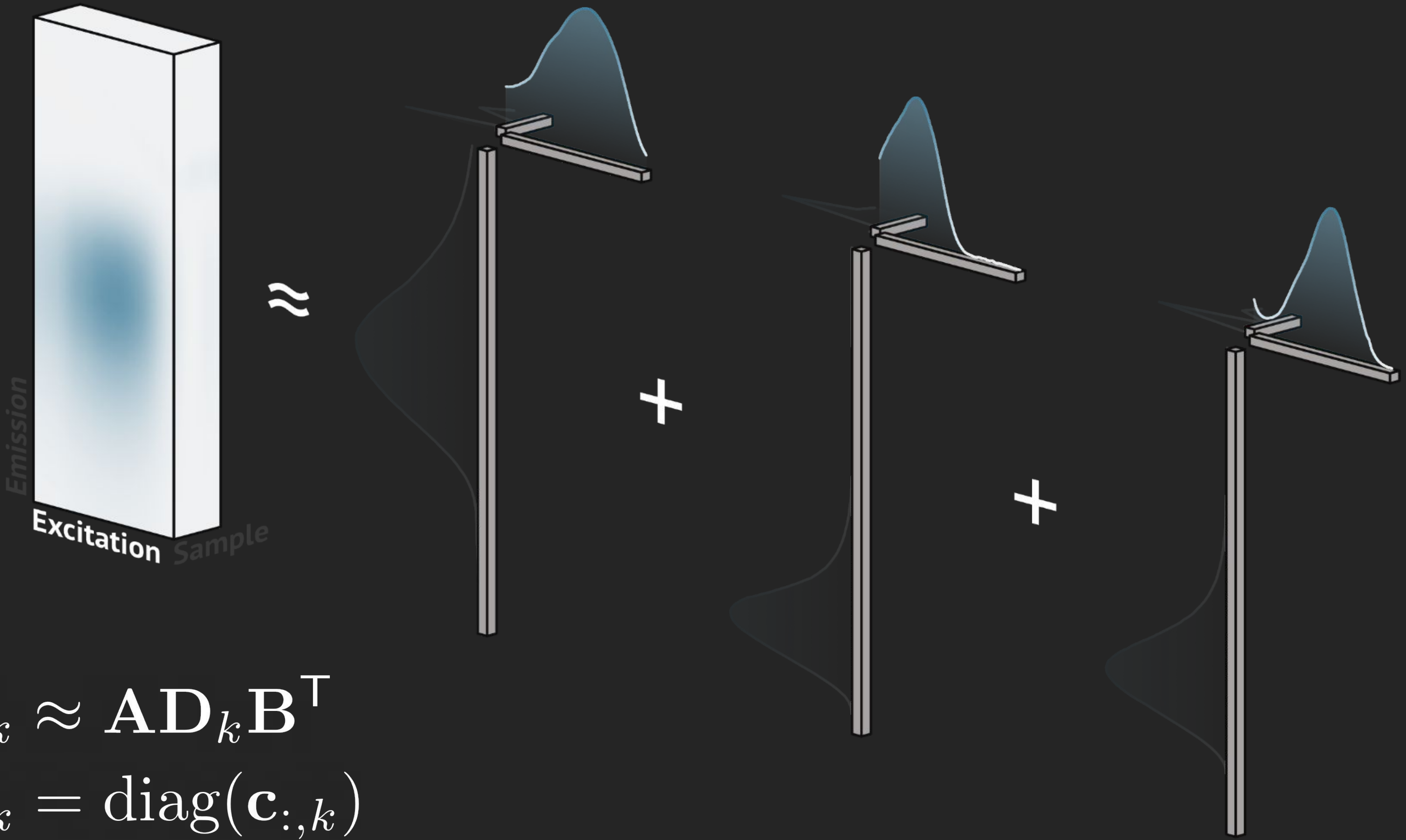
$$\mathbf{D}_k = \mathrm{diag}(\mathbf{c}_{:,k})$$

With PARAFAC, we find the EEM-spectra underlying the data



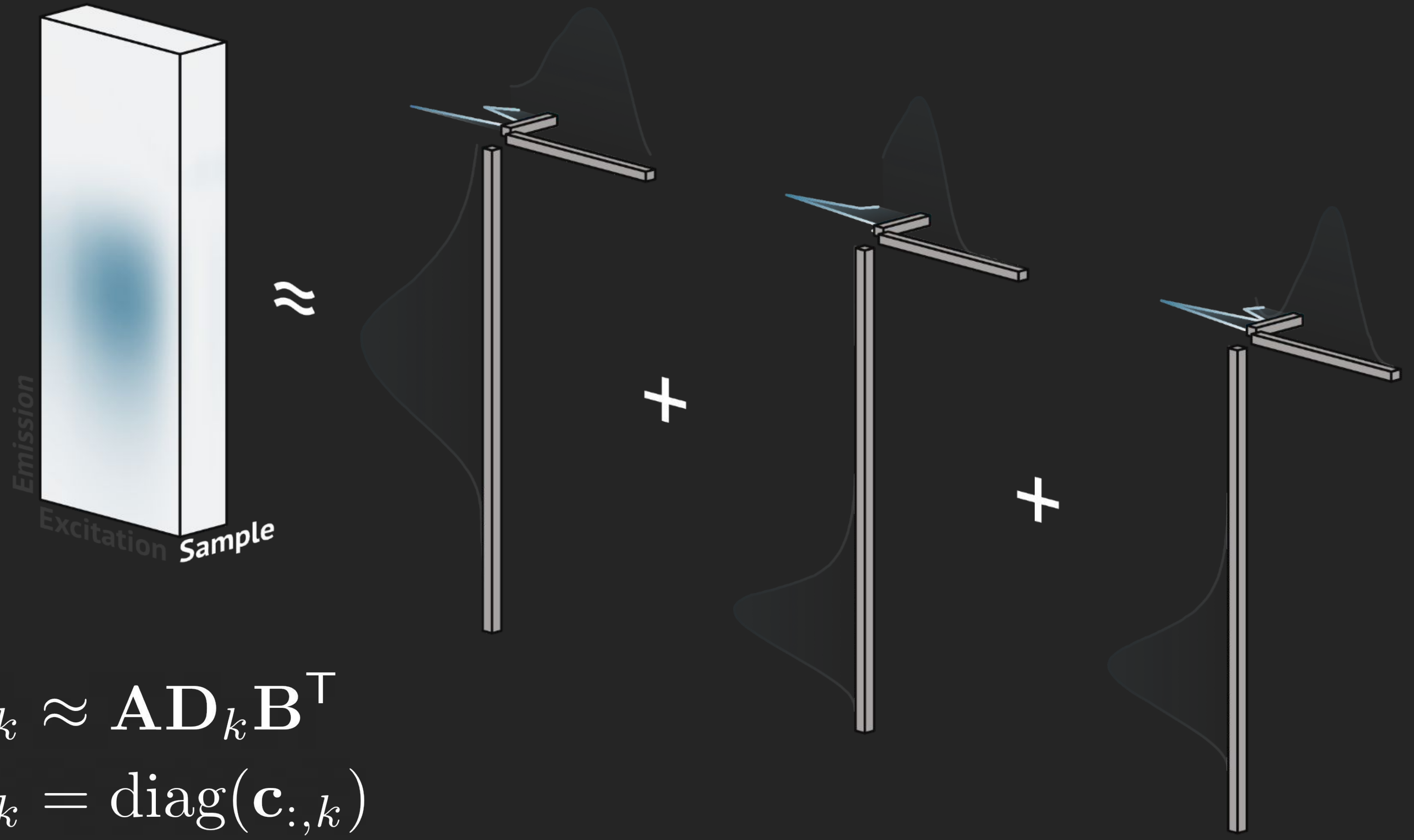
$$\mathbf{X}_k \approx \mathbf{A}\mathbf{D}_k\mathbf{B}^\top$$
$$\mathbf{D}_k = \mathrm{diag}(\mathbf{c}_{:,k})$$

With PARAFAC, we find the EEM-spectra underlying the data



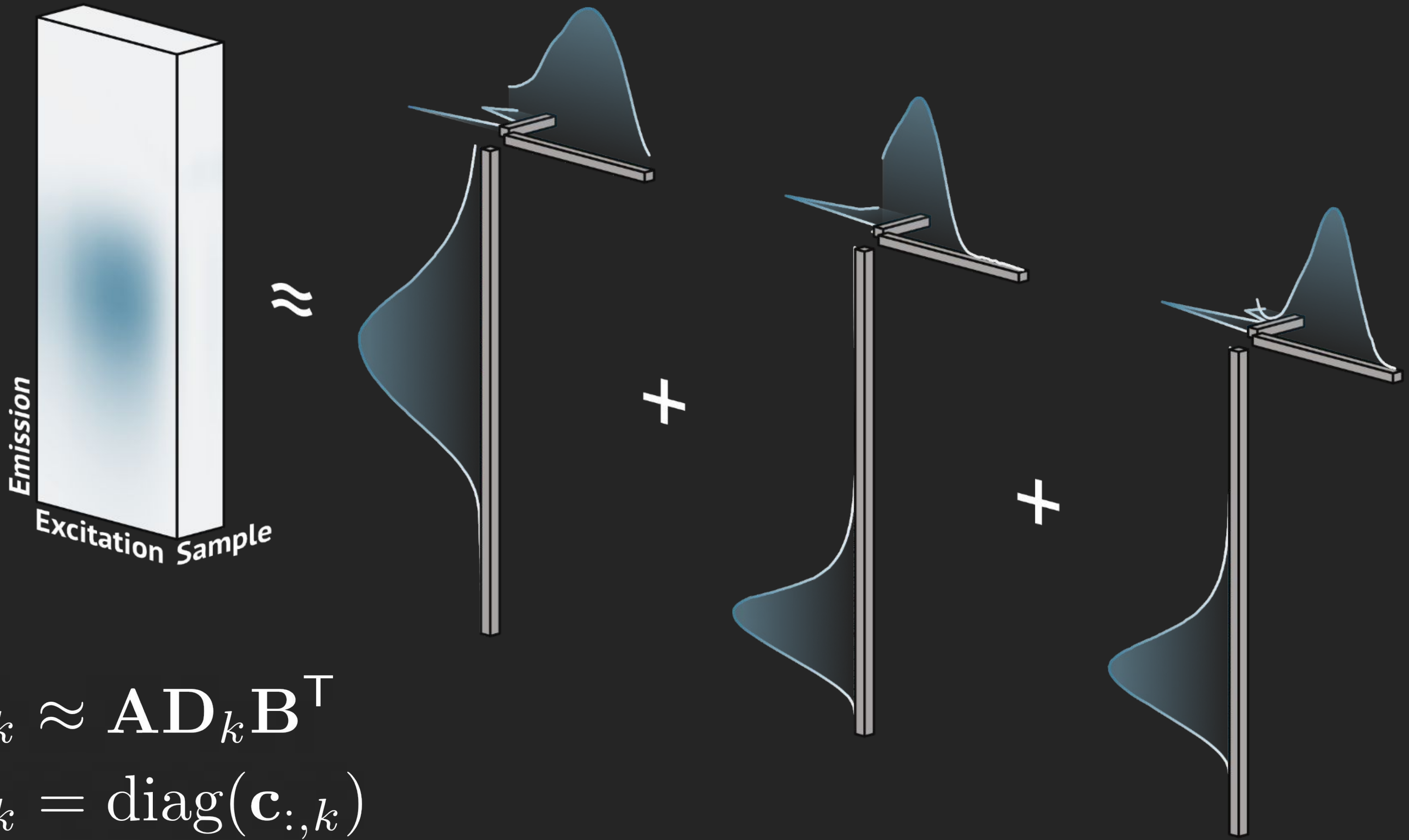
$$\mathbf{X}_k \approx \mathbf{A}\mathbf{D}_k\mathbf{B}^\top$$
$$\mathbf{D}_k = \mathrm{diag}(\mathbf{c}_{:,k})$$

With PARAFAC, we find the EEM-spectra underlying the data



$$\mathbf{X}_k \approx \mathbf{A}\mathbf{D}_k\mathbf{B}^\top$$
$$\mathbf{D}_k = \mathrm{diag}(\mathbf{c}_{:,k})$$

With PARAFAC, we find the EEM-spectra underlying the data



$$\mathbf{X}_k \approx \mathbf{A}\mathbf{D}_k\mathbf{B}^\top$$
$$\mathbf{D}_k = \mathrm{diag}(\mathbf{c}_{:,k})$$

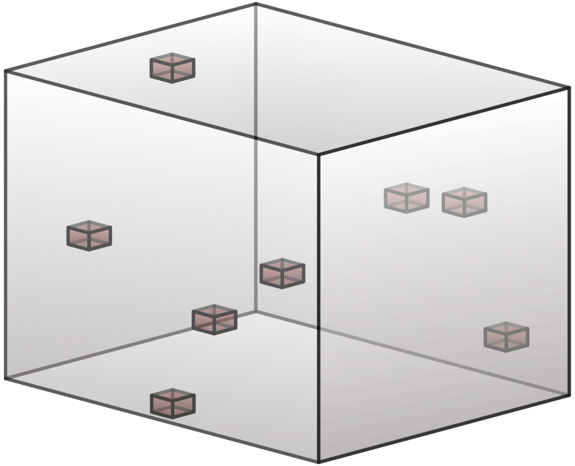
To find the PARAFAC components, we solve a nonlinear least squares problem

$$\min_{\mathbf{A},\mathbf{B},\mathbf{C}} \sum_{ijk} \left(x_{ijk} - \sum_{r} a_{ir} b_{jr} c_{kr} \right)^2$$

This formulation makes it possible to constrain the model to obtain non-negative components

$$\min_{\mathbf{A},\mathbf{B},\mathbf{C} \geq 0} \sum_{ijk} \left(x_{ijk} - \sum_{r} a_{ir} b_{jr} c_{kr} \right)^2$$

We can also handle missing data

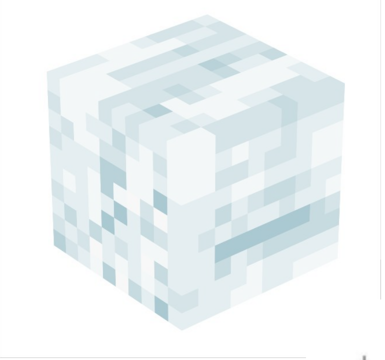


$$\min_{\mathbf{A},\mathbf{B},\mathbf{C}} \sum_{ijk} \left(x_{ijk} - \sum_r a_{ir} b_{jr} c_{kr} \right)^2$$

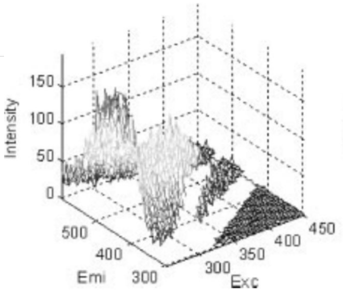
$$\downarrow$$

$$\min_{\mathbf{A},\mathbf{B},\mathbf{C}} \sum_{ijk} w_{ijk} \left(x_{ijk} - \sum_r a_{ir} b_{jr} c_{kr} \right)^2$$

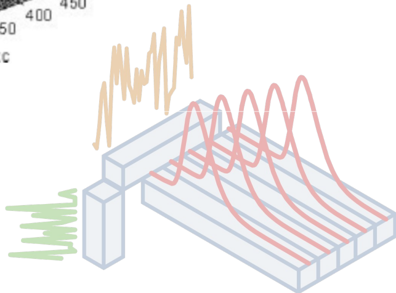
$$w_{ijk} = \begin{cases} 0 & \text{if } x_{ijk} \text{ is missing} \\ 1 & \text{otherwise} \end{cases}$$



Matrix and tensor decomposition



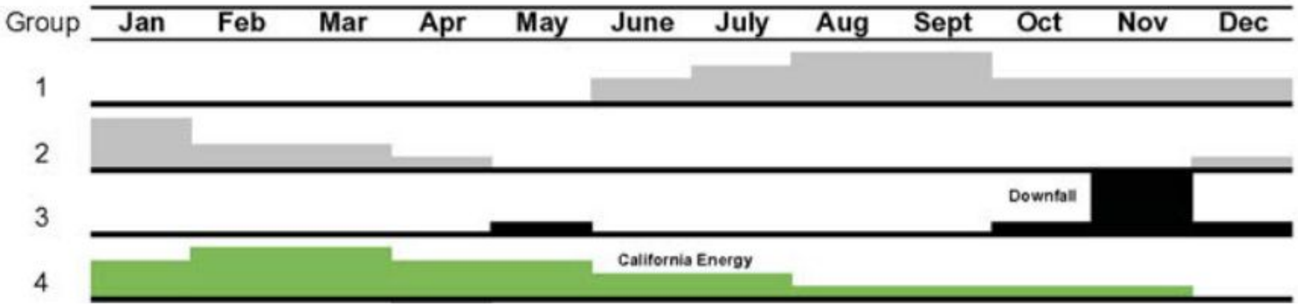
Applications of PARAFAC

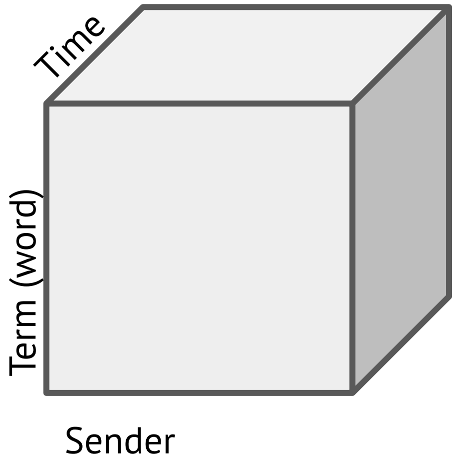


My research and PARAFAC2

Code demonstration

PARAFAC can discover e-mail topics and their popularity

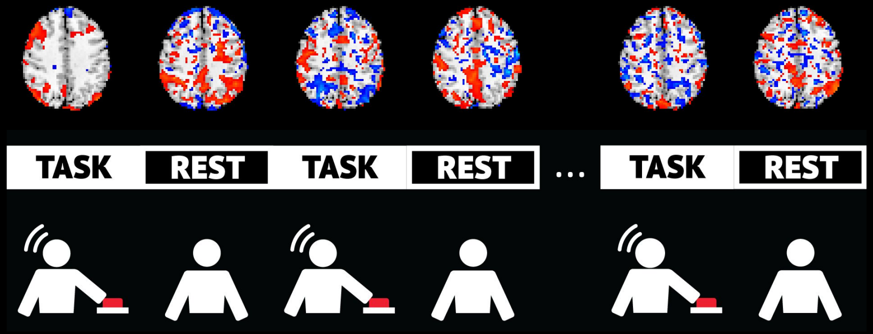


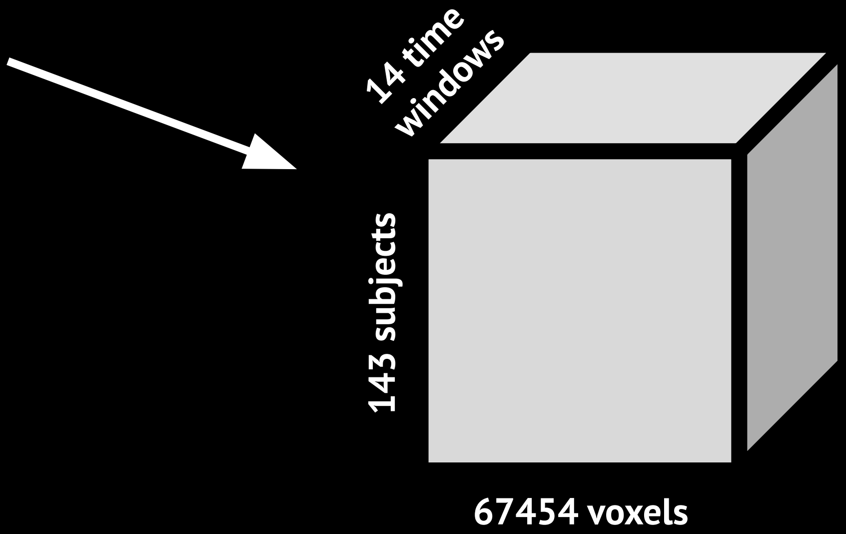


Popularity of the first four components
as a function of time

[Bader et al. (2008)]

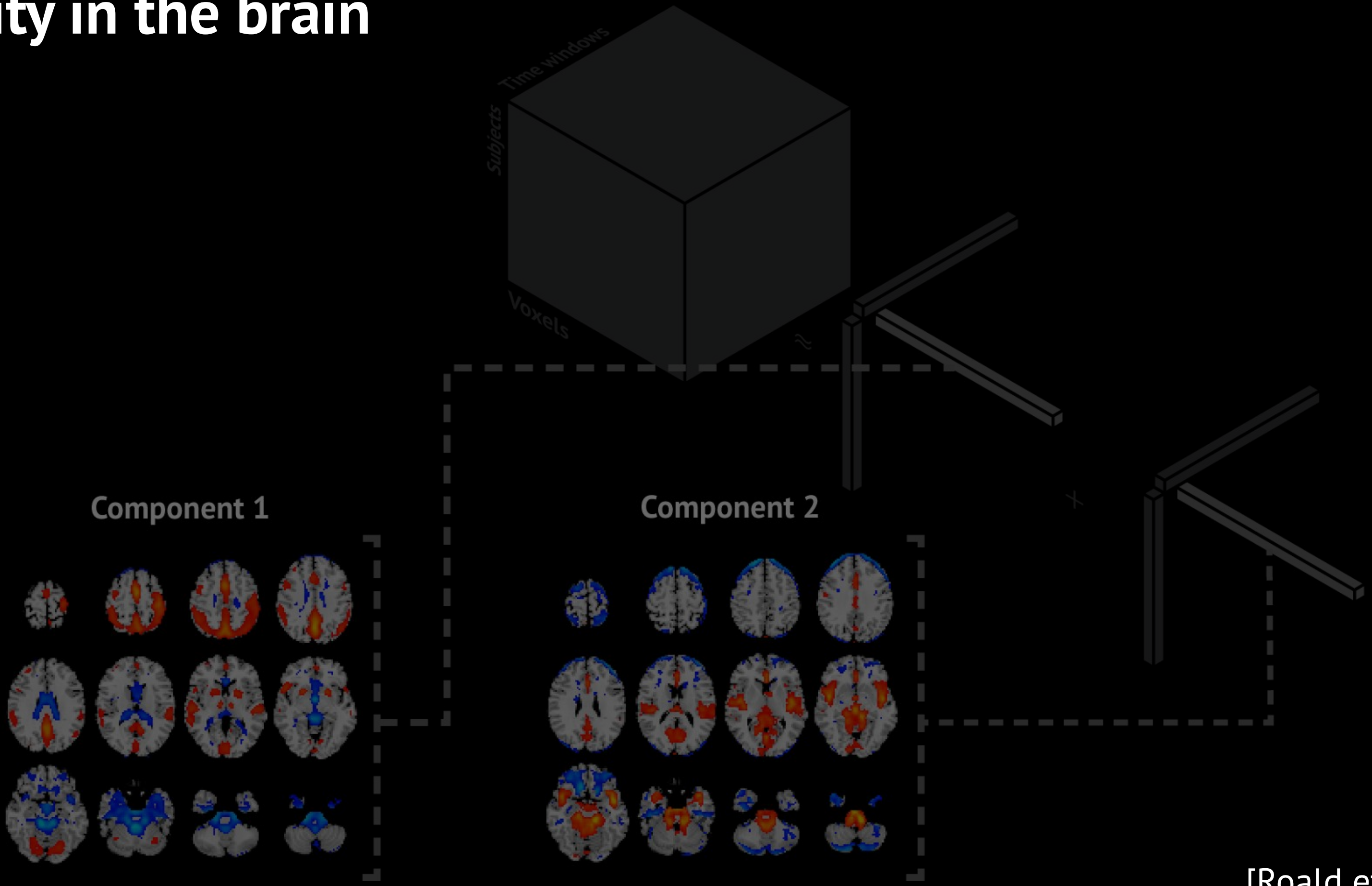
PARAFAC has also been used to discover networks of neural connectivity





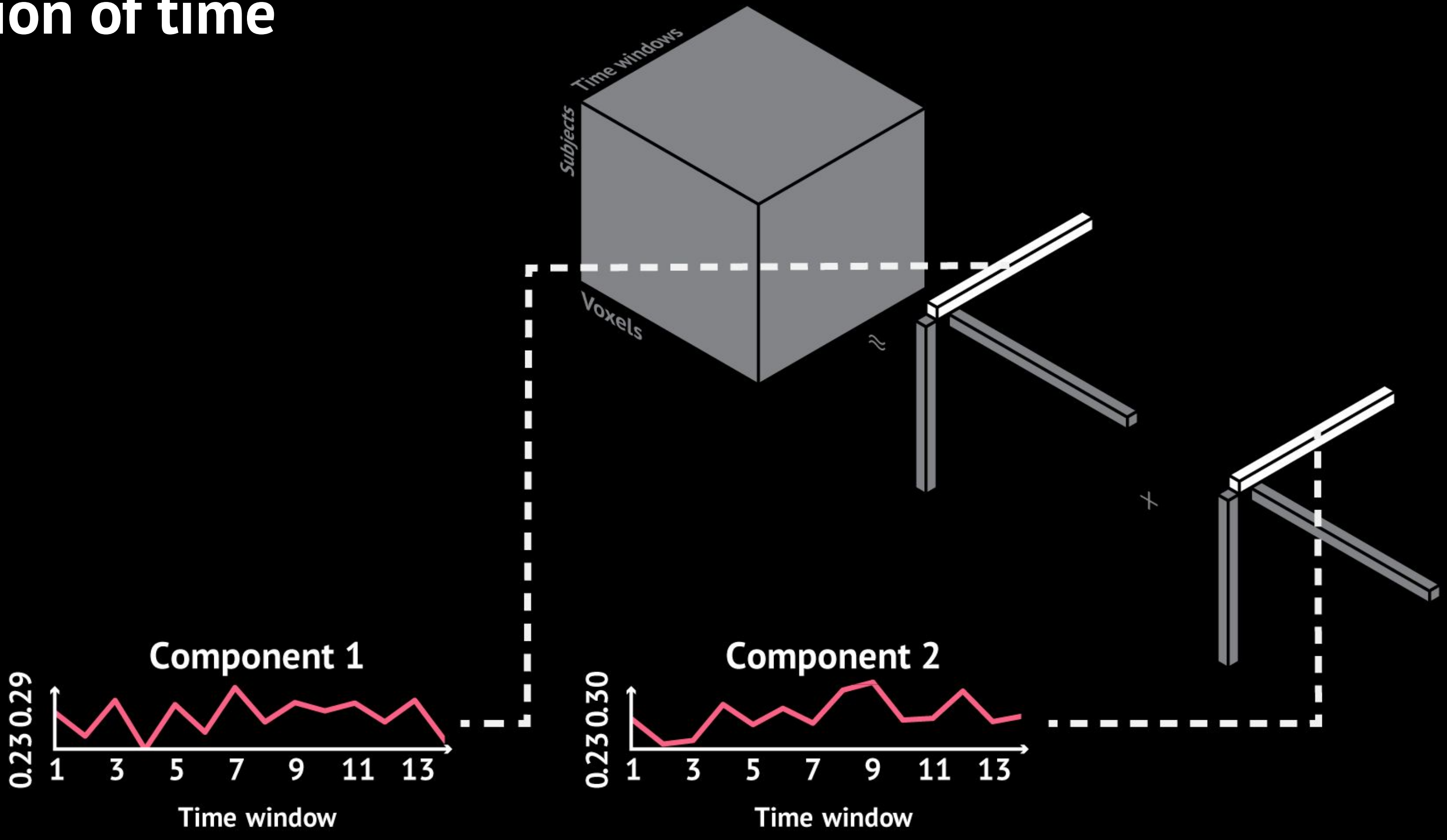
Subject distribution:
90 Healthy controls
53 Patients

PARAFAC can also be used to discover networks of neural connectivity in the brain



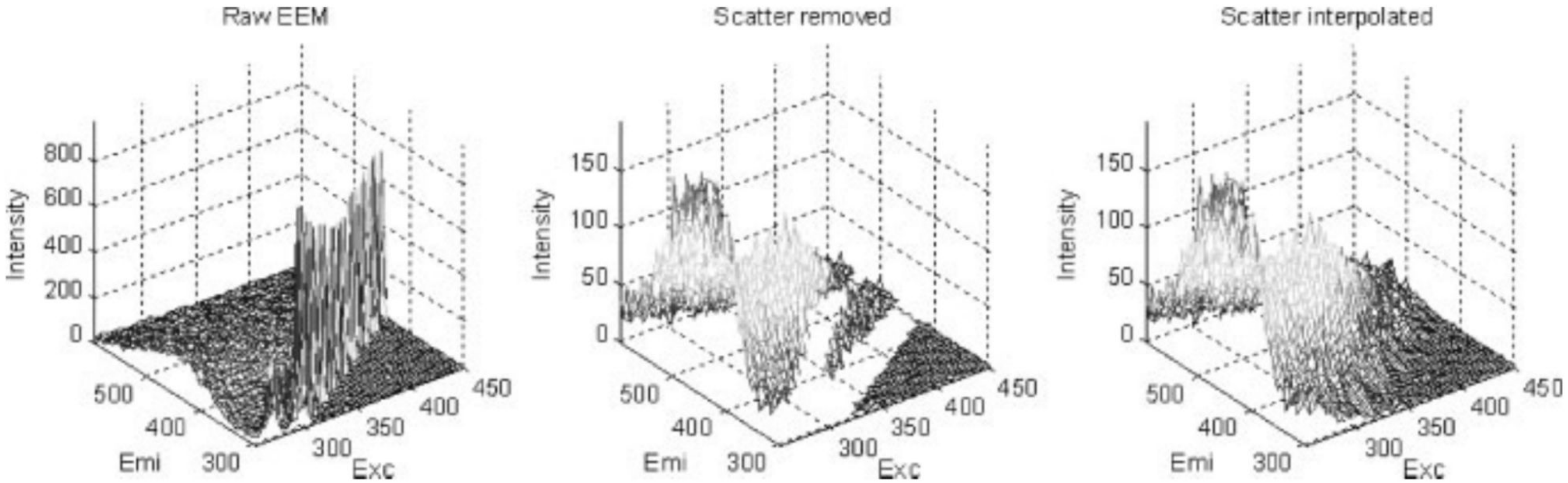
[Roald et al. (2020)]

The time-mode component, shows the networks' activation profile as a function of time



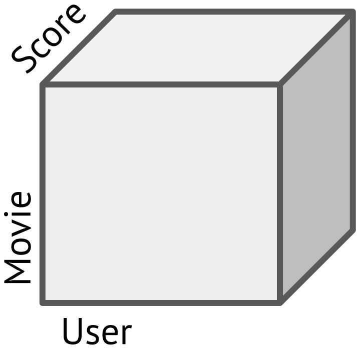
[Roald et al. (2020)]

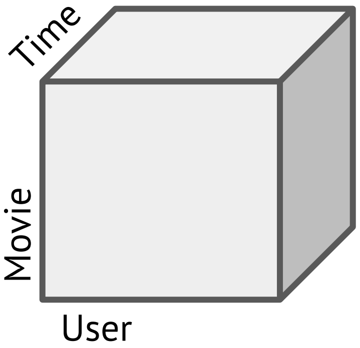
Which makes PARAFAC a good tool to analyse EEM-data with scattering artefacts



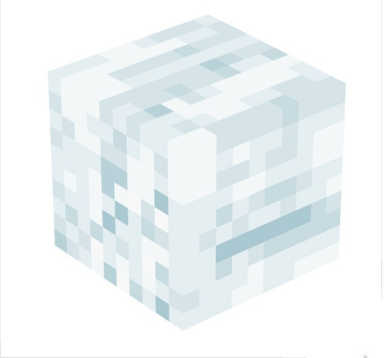
[Lawaetz et al. (2011)]

Weighted PARAFAC has also been used for recommendation engines

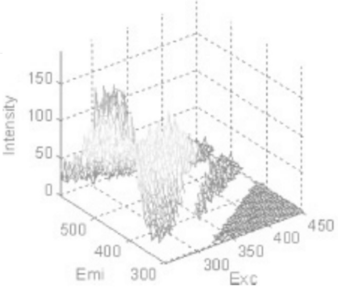




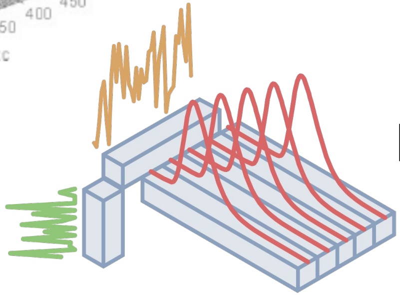
[Du et al. (2018)]



Matrix and tensor decomposition



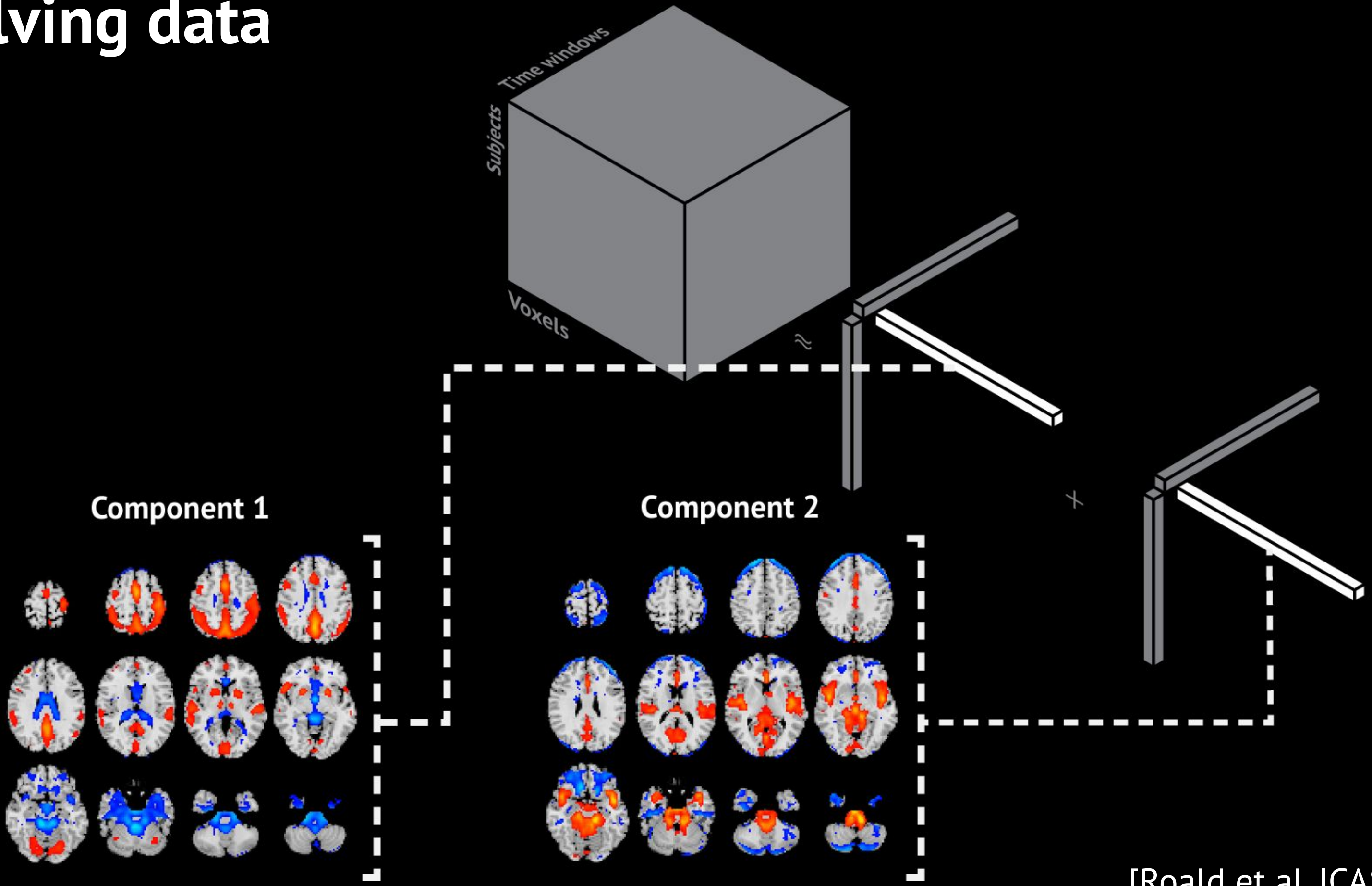
Applications of PARAFAC



My research and PARAFAC2

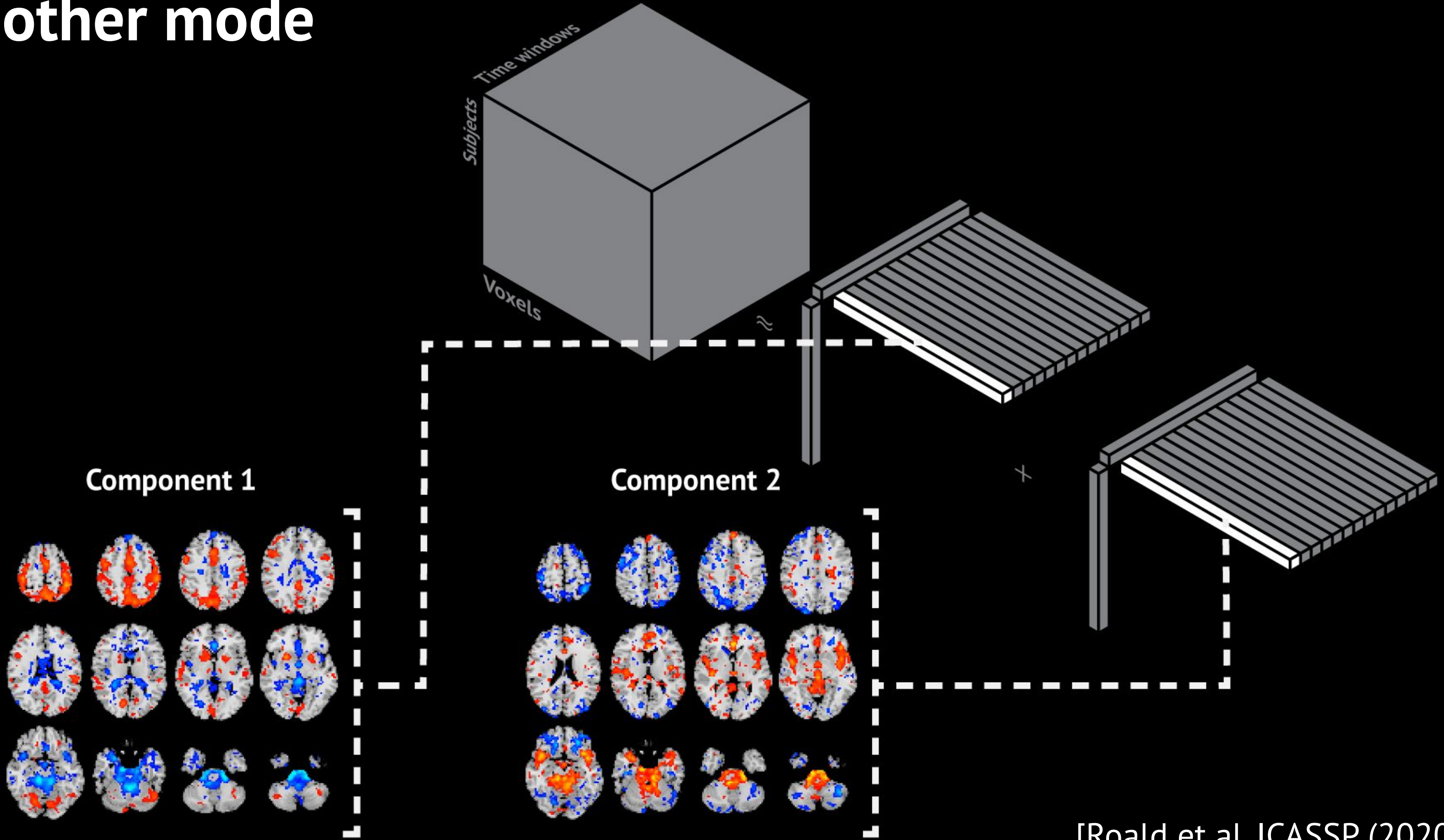
Code demonstration

The multilinearity of PARAFAC may be too restrictive for time-evolving data



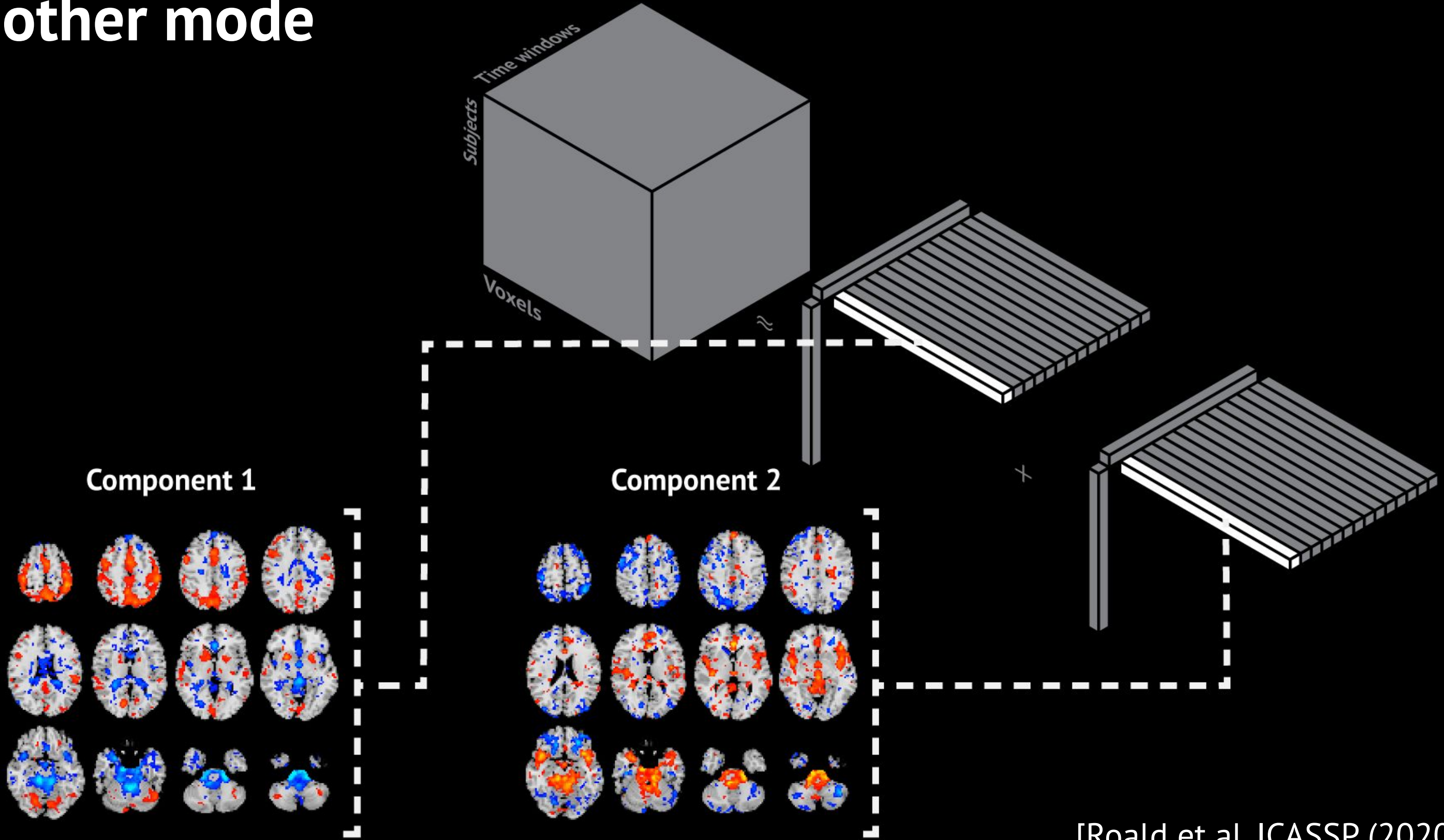
[Roald et al. ICASSP (2020)]

PARAFAC2 allows the components in one mode to evolve across another mode



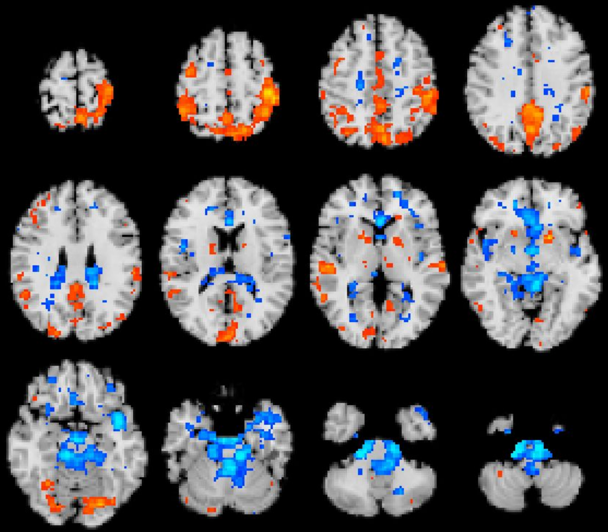
[Roald et al. ICASSP (2020)]

PARAFAC2 allows the components in one mode to evolve across another mode



[Roald et al. ICASSP (2020)]

However, the components obtained with PARAFAC2 were noisier and less stable than those obtained with PARAFAC



However, PARAFAC2 models are constrained in a way that makes it difficult to add additional regularisation

$$\mathbf{X}_k \approx \mathbf{A}\mathbf{D}_k\mathbf{B}_k^\top$$

$$\mathbf{B}_{k_1}^\top\mathbf{B}_{k_1} = \mathbf{B}_{k_2}^\top\mathbf{B}_{k_2}$$

[Harshman (1972)]

However, PARAFAC2 models are constrained in a way that makes it difficult to add additional regularisation

$$\mathbf{X}_k \approx \mathbf{A}\mathbf{D}_k\mathbf{B}_k^\top$$

$$\mathbf{B}_{k_1}^\top \mathbf{B}_{k_1} = \mathbf{B}_{k_2}^\top \mathbf{B}_{k_2}$$

[Harshman (1972)]

However, PARAFAC2 models are constrained in a way that makes it difficult to add additional regularisation

$$\mathbf{X}_k \approx \mathbf{A}\mathbf{D}_k\mathbf{B}_k^\top$$

$$\mathbf{B}_{k_1}^\top \mathbf{B}_{k_1} = \mathbf{B}_{k_2}^\top \mathbf{B}_{k_2}$$

[Harshman (1972)]

However, PARAFAC2 models are constrained in a way that makes it difficult to add additional regularisation

$$\mathbf{X}_k \approx \mathbf{A}\mathbf{D}_k\mathbf{B}_k^\top$$

$$\mathbf{B}_{k_1}^\top\mathbf{B}_{k_1} = \mathbf{B}_{k_2}^\top\mathbf{B}_{k_2}$$

$$\min_{\substack{\mathbf{A},\mathbf{B}_1,\ldots,\mathbf{B}_K,\mathbf{C} \\ \mathbf{B}_{k_1}^\top\mathbf{B}_{k_1} = \mathbf{B}_{k_2}^\top\mathbf{B}_{k_2}}} \left(x_{ijk} - \sum_r a_{ir} b_{kjr} c_{kr} \right)^2$$

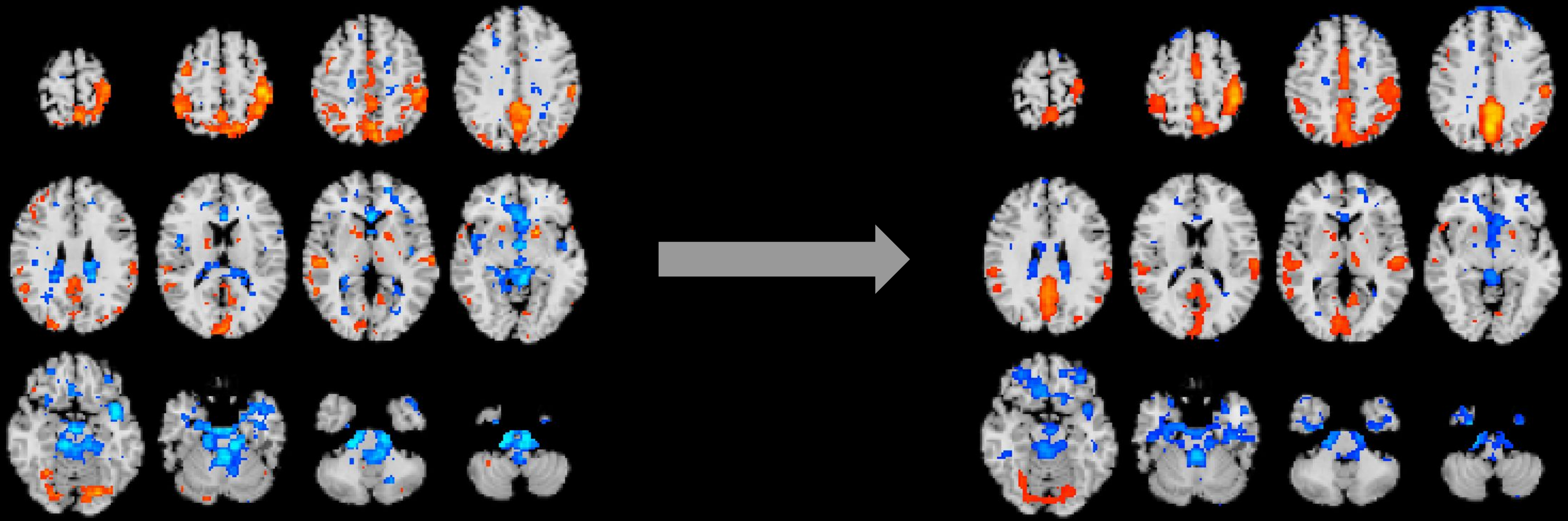
[Harshman (1972)]

We reformulated the loss function to allow for regularisation of all components

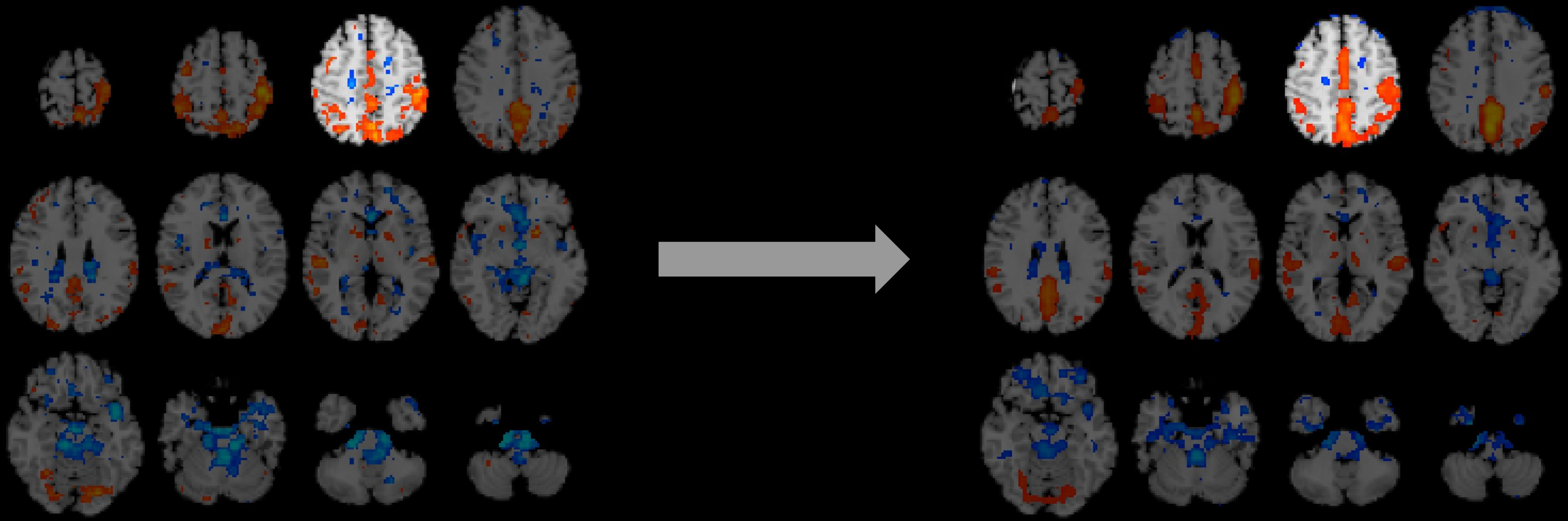
$$\underset{\{\mathbf{B}_k, \mathbf{Z}_{\mathbf{B}_k}, \mathbf{Y}_{\mathbf{B}_k}\}_{k \le K}}{\text{minimize}} \quad \sum_{k=1}^{K} \left\| \mathbf{A}\mathbf{D}_k \mathbf{B}_k^\top - \mathbf{X}_k \right\|_F^2 + g_{\mathbf{B}}\left(\mathbf{Z}_{\mathbf{B}_k}\right)$$

$$\begin{aligned} \text{s.t.} \quad \mathbf{B}_k &= \mathbf{Z}_{\mathbf{B}_k}, && \forall k \\ \mathbf{B}_k &= \mathbf{Y}_{\mathbf{B}_k}, && \forall k \\ \mathbf{Y}_{\mathbf{B}_k}^\top \mathbf{Y}_{\mathbf{B}_k} &= \Phi, && \forall k \end{aligned}$$

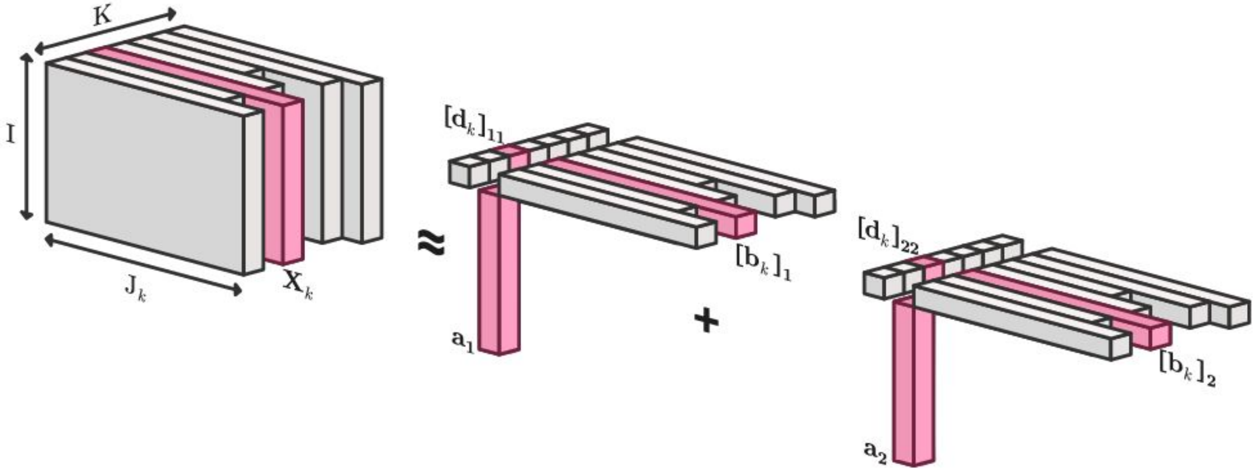
Smoothness regularisation leads to less noisy brain-activation maps when applied to fMRI data



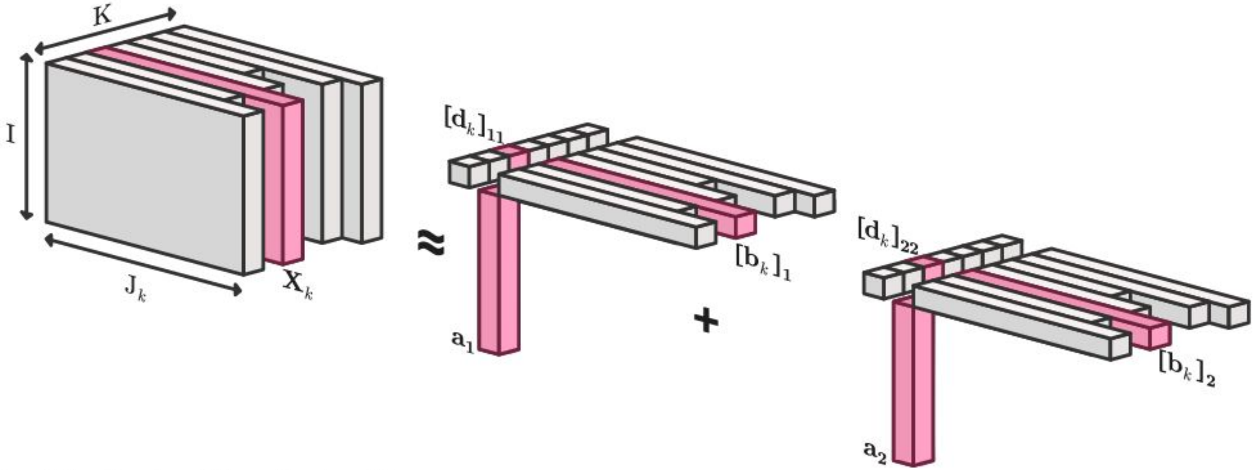
Smoothness regularisation leads to less noisy brain-activation maps when applied to fMRI data

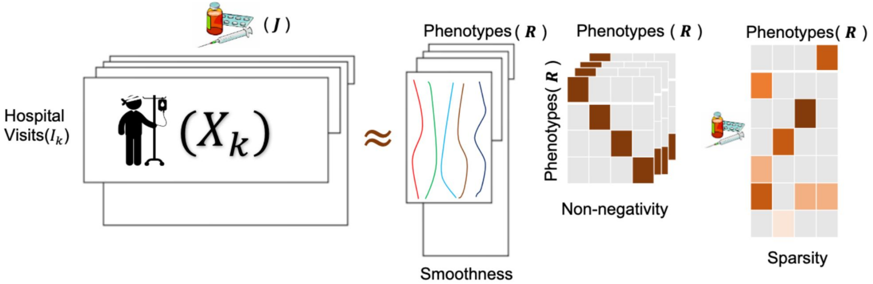


PARAFAC2 is also useful for a variety of applications where one mode varies across another



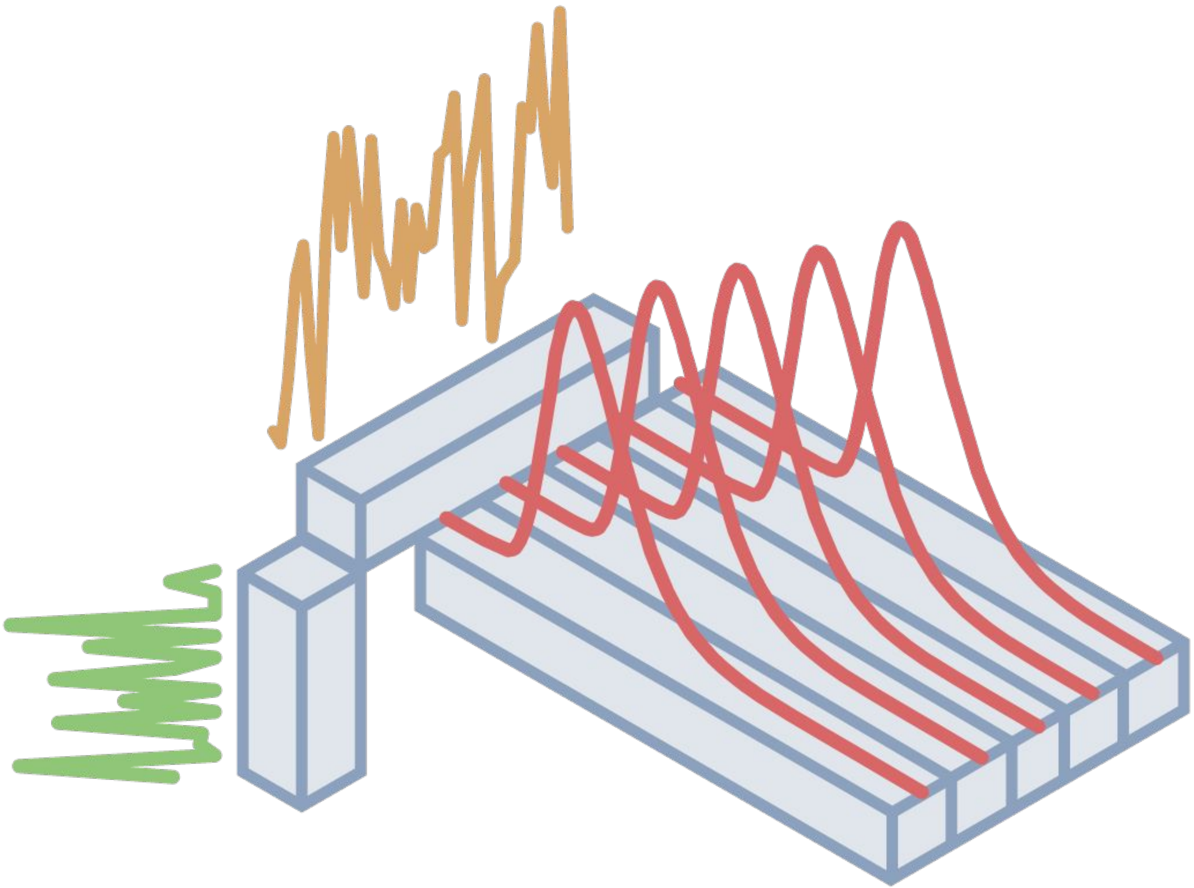
PARAFAC2 is also useful for analysing electronic health records, where the patients have different number of visits



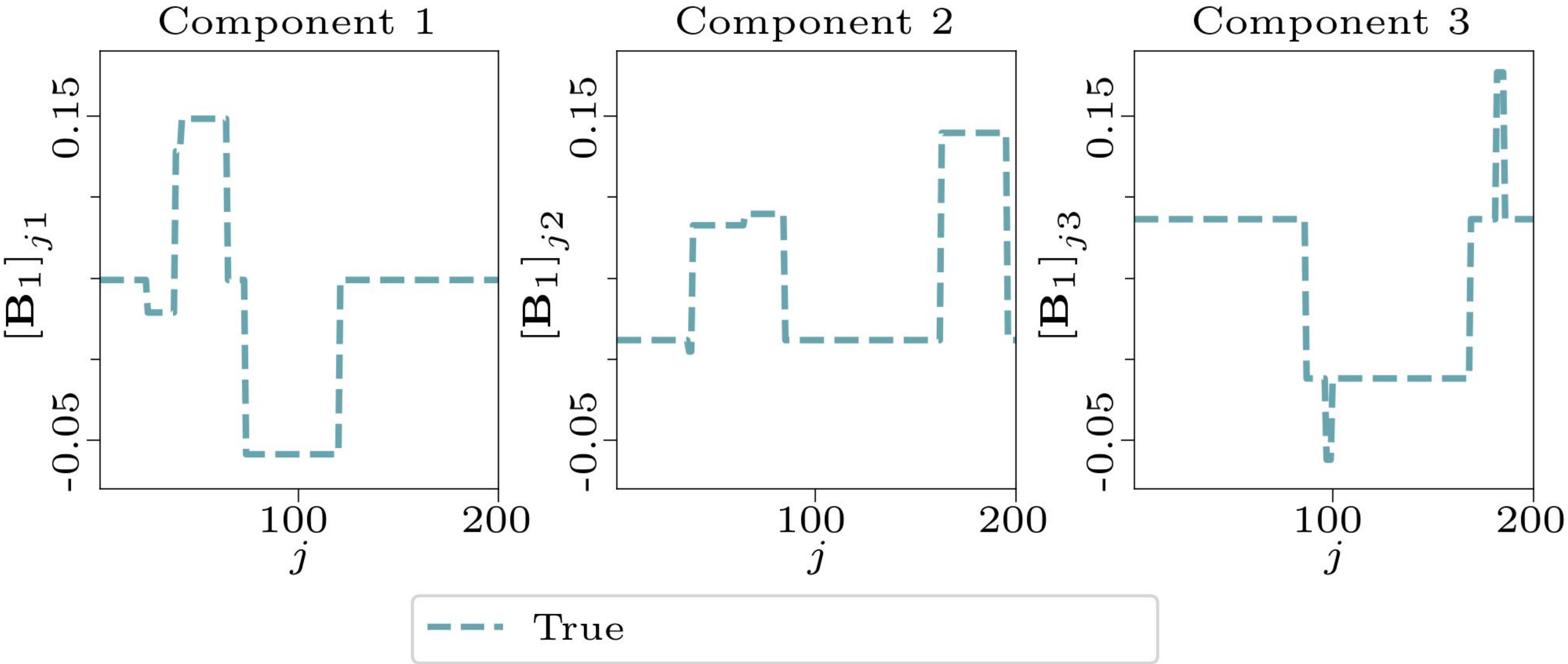


[Afshar et al. (2018)]

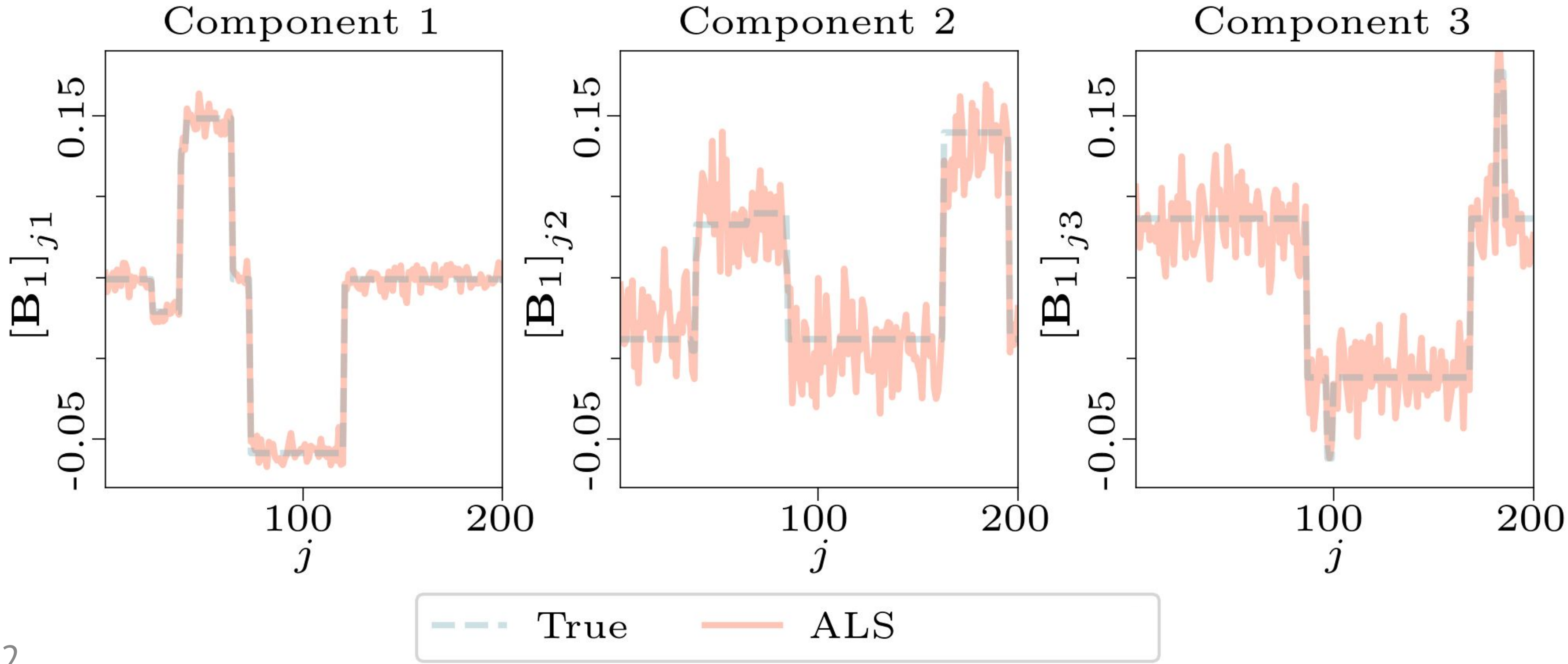
We tested the framework on a variety of real and simulated datasets



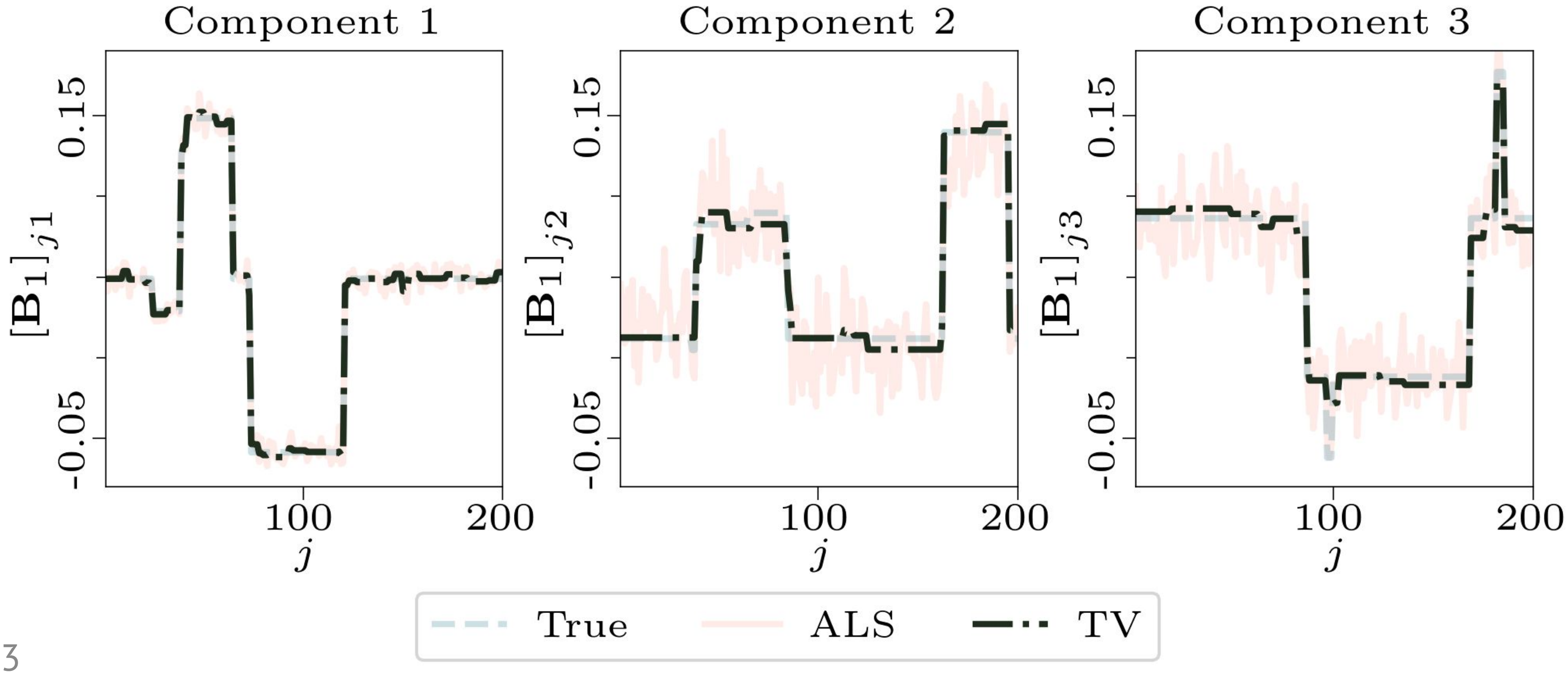
One of the setups used shifting piecewise-constant components



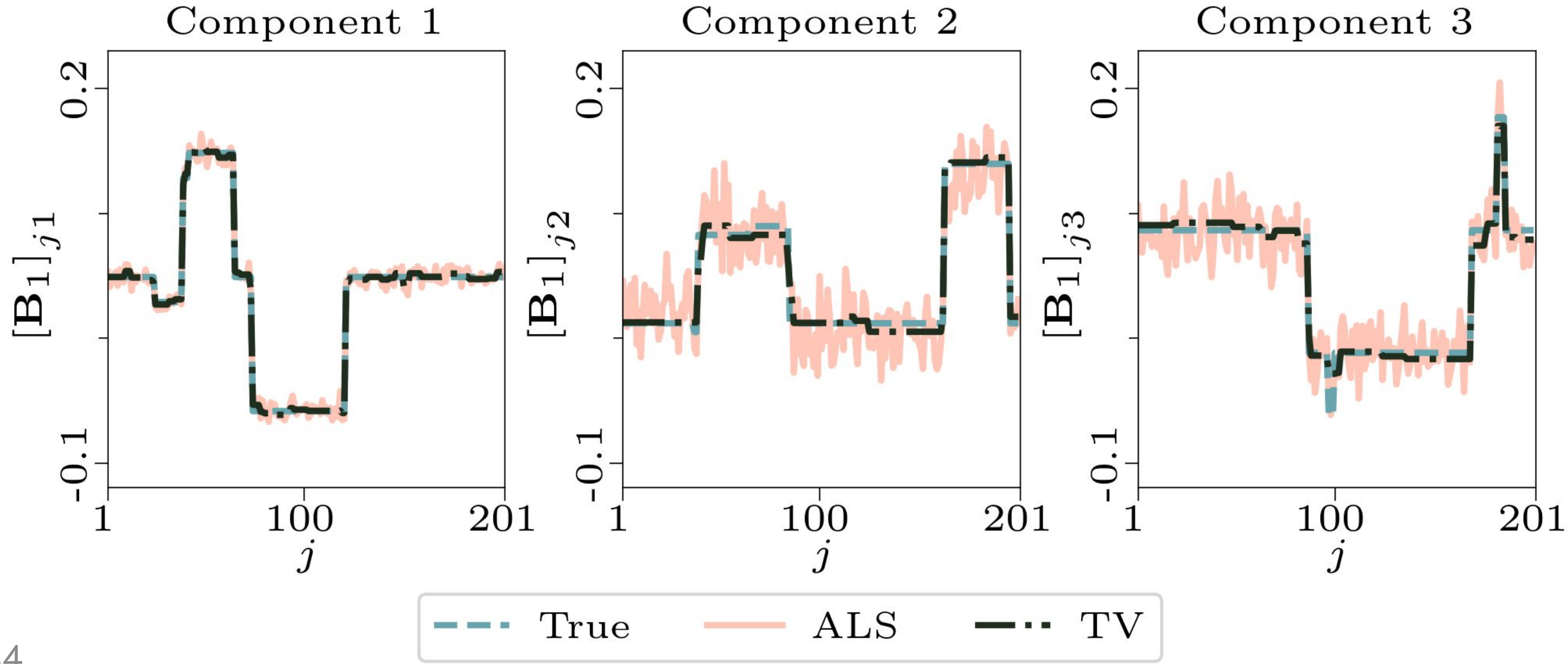
The standard PARAFAC2 algorithm yielded noisy components



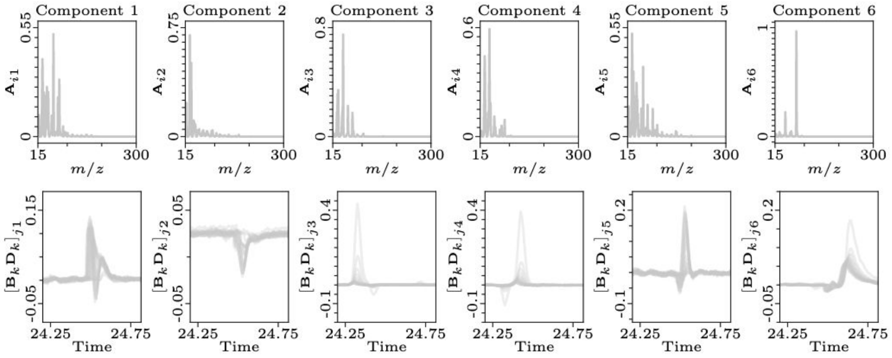
While the regularised PARAFAC2 model captured the components much better



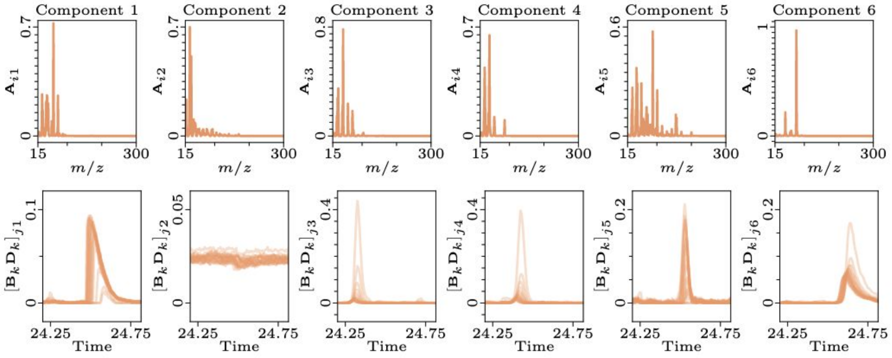
While the regularised PARAFAC2 model captured the components much better



Constrained PARAFAC2 is useful in a variety of applications

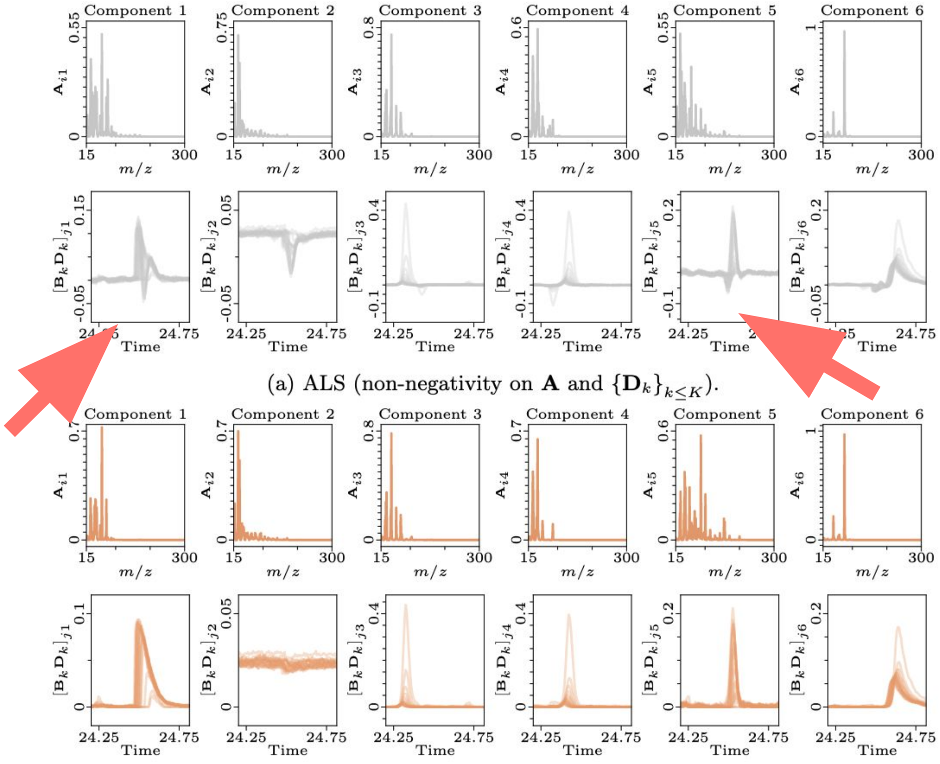


(a) ALS (non-negativity on $\mathbf{A}$ and $\{\mathbf{D}_k\}_{k \leq K}$).



(b) AO-ADMM (non-negativity on all modes).

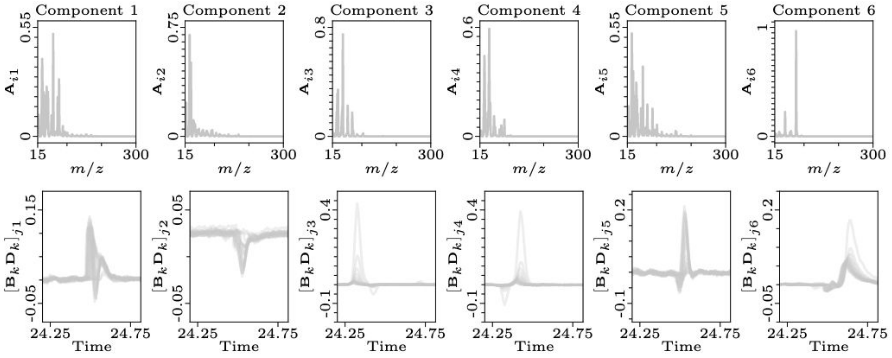
Constrained PARAFAC2 is useful in a variety of applications



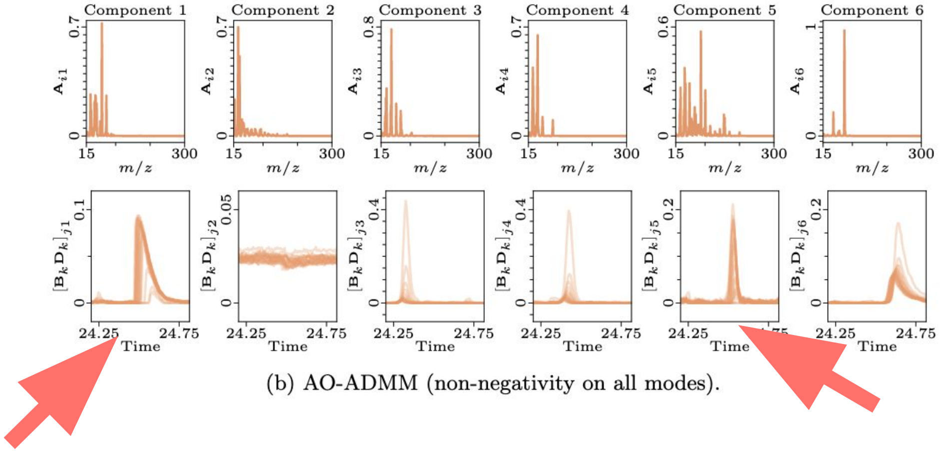
(a) ALS (non-negativity on $\mathbf{A}$ and $\{\mathbf{D}_k\}_{k \leq K}$).

(b) AO-ADMM (non-negativity on all modes).

Constrained PARAFAC2 is useful in a variety of applications

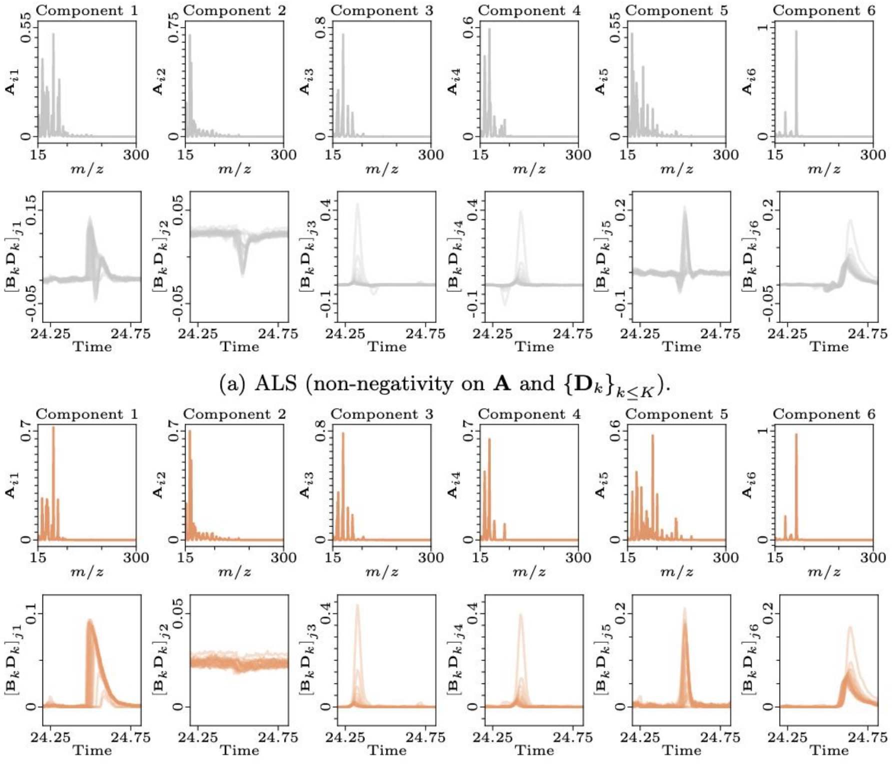


(a) ALS (non-negativity on $\mathbf{A}$ and $\{\mathbf{D}_k\}_{k \leq K}$).



(b) AO-ADMM (non-negativity on all modes).

Constrained PARAFAC2 is useful in a variety of applications



(a) ALS (non-negativity on $\mathbf{A}$ and $\{\mathbf{D}_k\}_{k \leq K}$).

(b) AO-ADMM (non-negativity on all modes).

An AO-ADMM approach to constraining PARAFAC2 on all modes*

Marie Roald† Carla Schenker† Rasmus Bro‡
Jeremy E. Cohen§ Evrim Acar¶

Abstract

Analyzing multi-way measurements with variations across one mode of the dataset is a challenge in various fields including data mining, neuroscience and chemometrics. For example, measurements may evolve over time or have unaligned time profiles. The PARAFAC2 model has been successfully used to analyze such data by allowing the underlying factor matrices in one mode (i.e., the evolving mode) to change across slices. The traditional approach to fit a PARAFAC2 model is to use an alternating least squares-based algorithm, which handles the constant cross-product constraint of the PARAFAC2 model by implicitly estimating the evolving factor matrices. This approach makes imposing regularization on these factor matrices challenging. There is currently no algorithm to flexibly impose such regularization with general penalty functions and hard constraints. In order to address this challenge and to avoid the implicit estimation, in this paper, we propose an algorithm for fitting PARAFAC2 based on alternating optimization with the alternating direction method of multipliers (AO-ADMM). With numerical experiments on simulated data, we show that the proposed PARAFAC2 AO-ADMM approach allows for flexible constraints, recovers the underlying patterns accurately, and is computationally efficient compared to the state-of-the-art. We also apply our model to a real-world chromatography dataset, and show that constraining the evolving mode improves the interpretability of the extracted patterns.

1 Introduction

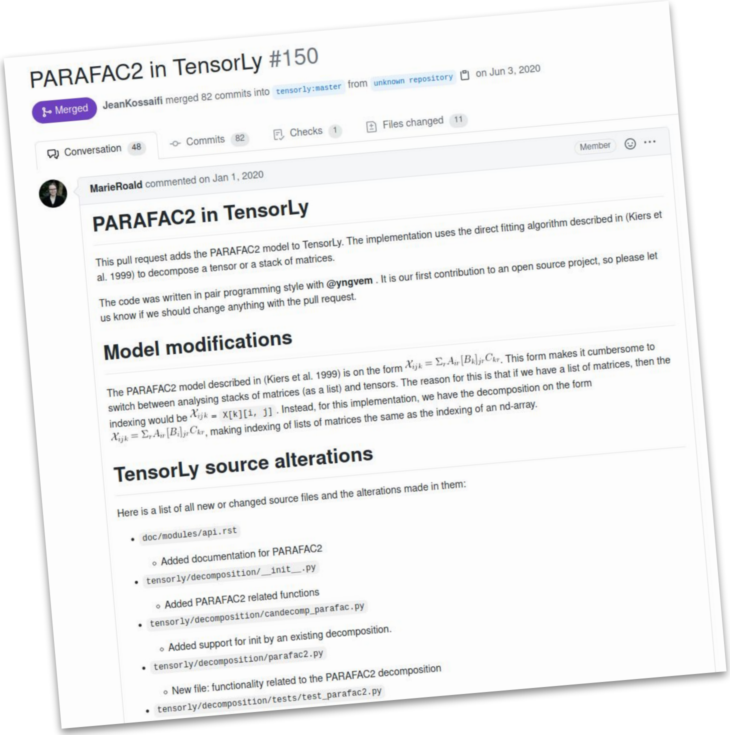
For many applications in different domains, measurements are obtained in the form of sequences of matrices, which can be arranged as a third-order tensor. The CANDE-

*Funding: This work was supported in part by the Research Council of Norway through project 300489 (IKTPLUSS) and ANR JCJC project LoRAiA ANR-20-CE23-0010.

†Department of Machine Intelligence, Simula Metropolitan Center for Digital Engineering, Oslo, Norway & Faculty of Technology, Art and Design, Oslo Metropolitan University, Oslo, Norway (mariero@simula.no, carla@simula.no).
‡Department of Food Science, University of Copenhagen, Copenhagen, Denmark (rb@food.ku.dk).
§CNRS, University of Rennes 1, IRISA, Rennes, France (jeremy.cohen@irisa.fr).
¶Department of Machine Intelligence, Simula Metropolitan Center for Digital Engineering, Oslo, Norway (evrim@simula.no).

arXiv: 2110.01278

I am currently working on implementing my framework as a Python package that I plan to publish as a software paper



PARAFAC2 in TensorLy #150

Merged JeanKossaifi merged 82 commits into `tensorly:master` from `unknown repository` on Jun 3, 2020

Conversation 48 · Commits 82 · Checks 1 · Files changed 11

MarieRoald commented on Jan 1, 2020 — Member

PARAFAC2 in TensorLy

This pull request adds the PARAFAC2 model to TensorLy. The implementation uses the direct fitting algorithm described in (Kiers et al. 1999) to decompose a tensor or a stack of matrices.

The code was written in pair programming style with **@yngvem** . It is our first contribution to an open source project, so please let us know if we should change anything with the pull request.

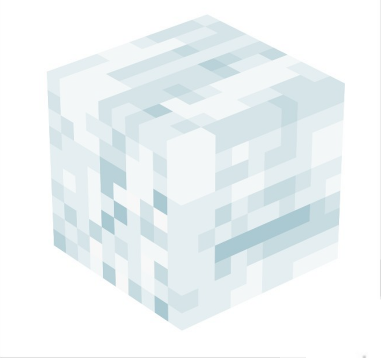
Model modifications

The PARAFAC2 model described in (Kiers et al. 1999) is on the form $\mathcal{X}_{ijk} = \Sigma_r A_{ir} [B_k]_{jr} C_{kr}$. This form makes it cumbersome to switch between analysing stacks of matrices (as a list) and tensors. The reason for this is that if we have a list of matrices, then the indexing would be $\mathcal{X}_{ijk}$ = `X[k][i, j]` . Instead, for this implementation, we have the decomposition on the form $\mathcal{X}_{ijk} = \Sigma_r A_{ir} [B_i]_{jr} C_{kr}$, making indexing of lists of matrices the same as the indexing of an nd-array.

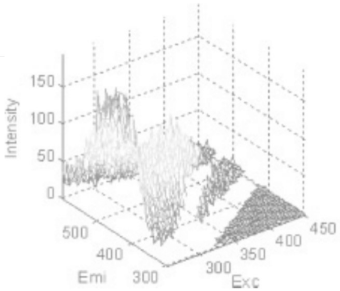
TensorLy source alterations

Here is a list of all new or changed source files and the alterations made in them:

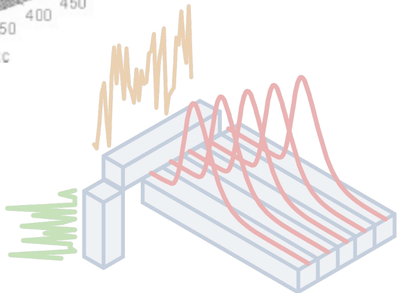
- `doc/modules/api.rst`
  - Added documentation for PARAFAC2
- `tensorly/decomposition/__init__.py`
  - Added PARAFAC2 related functions
- `tensorly/decomposition/candecomp_parafac.py`
  - Added support for init by an existing decomposition.
- `tensorly/decomposition/parafac2.py`
  - New file: functionality related to the PARAFAC2 decomposition
- `tensorly/decomposition/tests/test_parafac2.py`



Matrix and tensor decomposition



Applications of PARAFAC



My research and PARAFAC2

Code demonstration

The Jupyter notebooks are available on GitHub and can be run locally or online with Binder

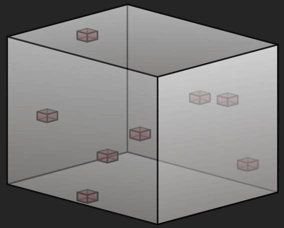
https://github.com/MarieRoald/nmbu-tensor-seminar-2021

https://mybinder.org/v2/gh/MarieRoald/nmbu-tensor-seminar-2021/HEAD

We have seen that tensor decomposition methods:

utilise the multi-way structure of the data

provide interpretable components



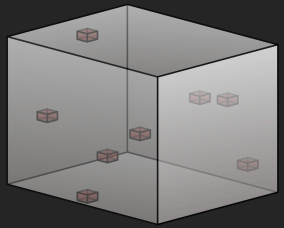
can handle missing data naturally

simulamet

We have seen that tensor decomposition methods:

utilise the multi-way structure of the data

provide interpretable components



can handle missing data naturally

Questions?