Nonnegative Matrix Factorization

Recap: identifiability issues in BSS

Generative model: $\mathbf{X} = \mathbf{A}^*\mathbf{S}^*$

But

$$\Rightarrow \mathbf{X} = \mathbf{A}^*\mathbf{P}\mathbf{P}^{-1}\mathbf{S}^* \quad \text{with } \mathbf{P} \text{ any invertible matrix}$$

$$\Rightarrow \mathbf{X} = \mathbf{A}\mathbf{S} \quad \text{with } \mathbf{A} = \mathbf{A}^*\mathbf{P} \text{ and } \mathbf{S} = \mathbf{P}^{-1}\mathbf{S}^*$$

$\mathbf{A}, \mathbf{S}$ can also explain the mixture

Thus, there is a infinite number of possible solutions which do not correspond to the true generating $\mathbf{A}^*, \mathbf{S}^*$ factors.

Recap 2 : Introducing additional priors

Said differently, BSS is an ill-posed problem.

=> need to introduce additional information, or also additional *priors*, on the sought after factors **A***, **S***.

You have seen two different types of priors in BSS:

- ➢ Assume the independence of **S** (ICA)
- ➢ Assume the sparsity of **S** (SBSS)

In addition, we have also discussed the optimization framework, and how to leverage the information contained within a training set, which might induce additional (implicit priors)

We will now have a look at a last explicit prior:

➢ **Use non-negativity (NMF) of A*and S***

=> each family has its strengths and weaknesses

NMF constraint

- Non-negative matrix factorization model, in short:

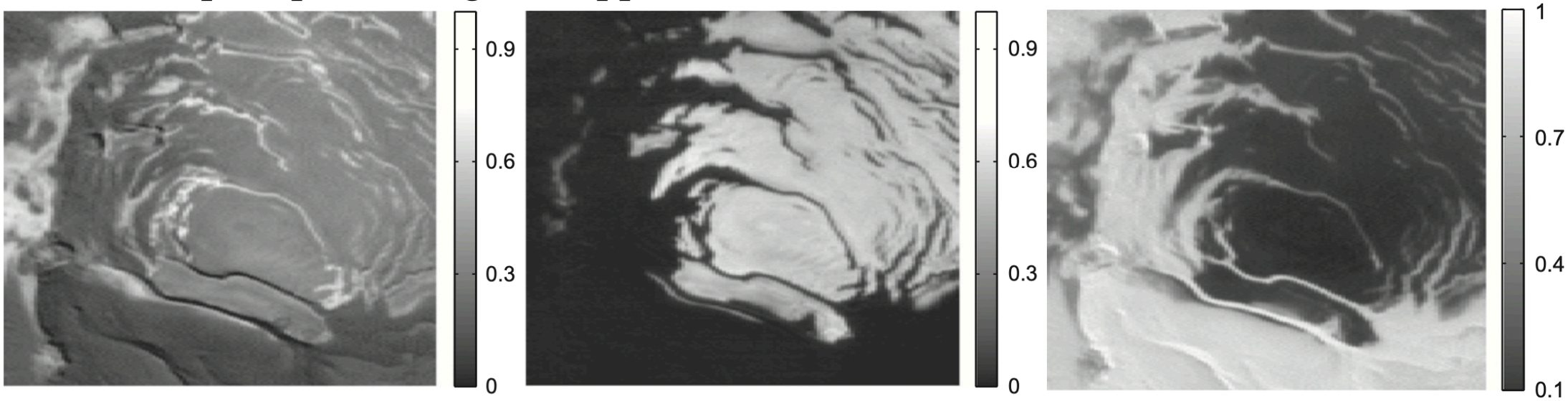
$$\mathbf{X} = \mathbf{A}^*\mathbf{S}^* + \mathbf{N} \quad \text{with} \quad \mathbf{A}^* \geq 0 \quad \text{and} \quad \mathbf{S}^* \geq 0$$

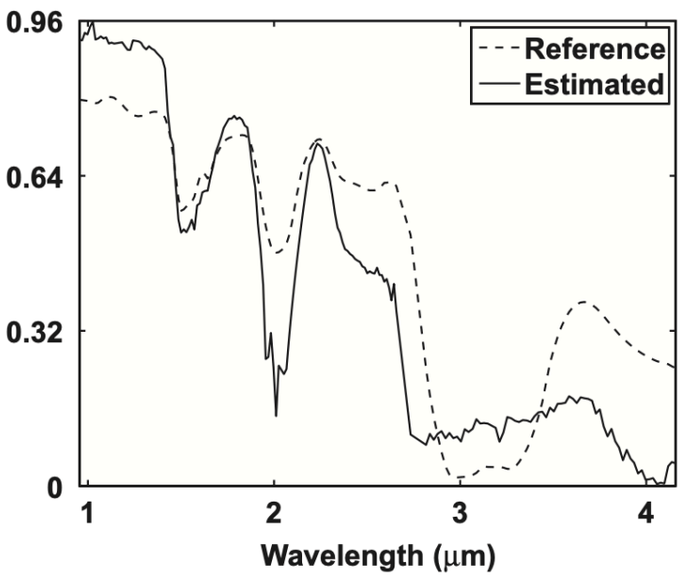
- There are applications in which it naturally makes sense, such as:
  - **multi/hyperspectral unmixing** (abundances $\mathbf{S}^*$ are concentrations, endmember signatures $\mathbf{A}^*$ are spectra)
  - **text mining** (corresponding to words counts)
  - **audio**, with works on the modulus of the Fourier transform…

- Beyond the fact that it is a naturally filled-constraint in many real-world application, why NMF?
  - In contrast to ICA, NMF can cope with **noise**.
  - In contrast to sparsity, more theoretical results have been obtained

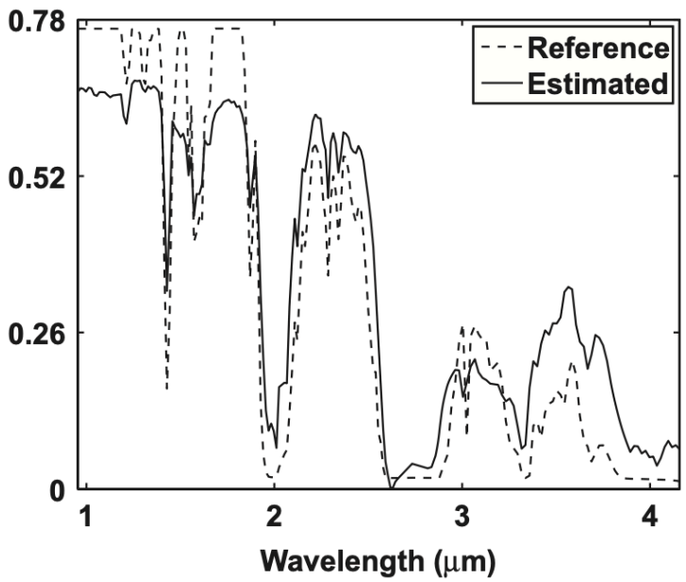
NMF : illustration

- Example: Mars express dataset
  - ICA was not fully satisfying, because neither spatial and spectral independences hypotheses was not fulfilled.
  - An NMF post-processing was applied to refine the results.

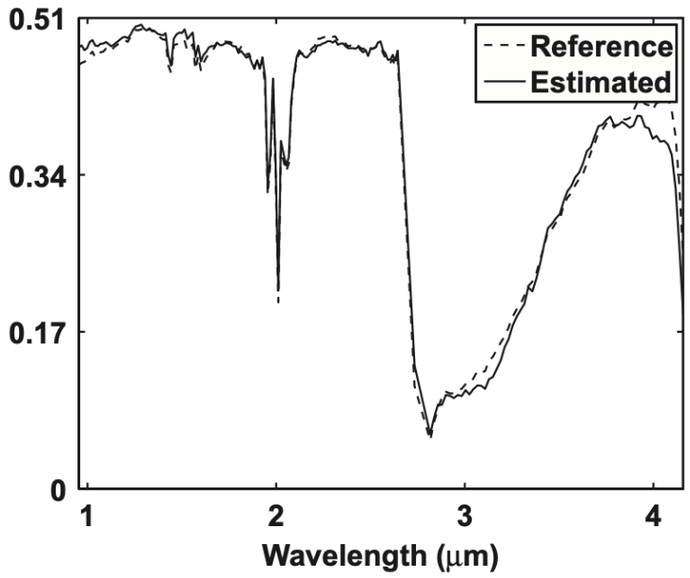




H_2O ice



CO_2 ice



CO_2 ice

Outline

- **Plain NMF**
    - Problem statement
    - Optimization framework
        - PALM
        - Multiplicative updates
- **Near-separable NMF**
    - Definition
    - Algorithms: brute force and greedy
    - Recovery guarantees
- **Minimum volume NMF**
- **Extensions to Deep Learning**

Plain Nonnegative Matrix Factorization (NMF)

- There are several sub-families of NMF:
  - Each of them might introduce (or not) further priors, in addition to nonnegativity
  - Each have dedicated algorithms and recovery guanrantees

The first we will consider is

- **<u>Plain NMF:</u>**

  Given a nonnegative matrix $\mathbf{X} \in \mathbb{R}_{+}^{m \times t}$, a number of sources n (= factorization rank), and a norm distance measure $D(.\,,.)$ between matrices, compute two nonnegative matrices $\hat{\mathbf{A}} \in \mathbb{R}^{m \times n}$ and $\hat{\mathbf{S}} \in \mathbb{R}^{n \times t}$ such that

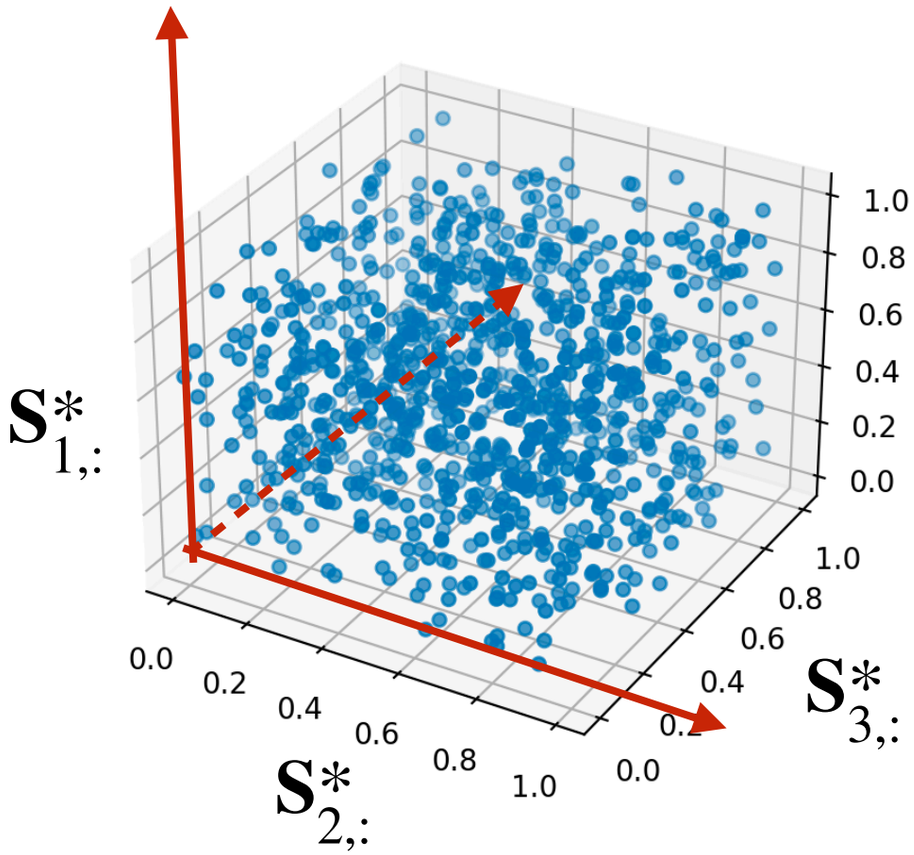
$$\hat{\mathbf{A}}, \hat{\mathbf{S}} = \arg \min_{\mathbf{A} \geq 0, \mathbf{S} \geq 0} D(\mathbf{X}, \mathbf{AS})$$

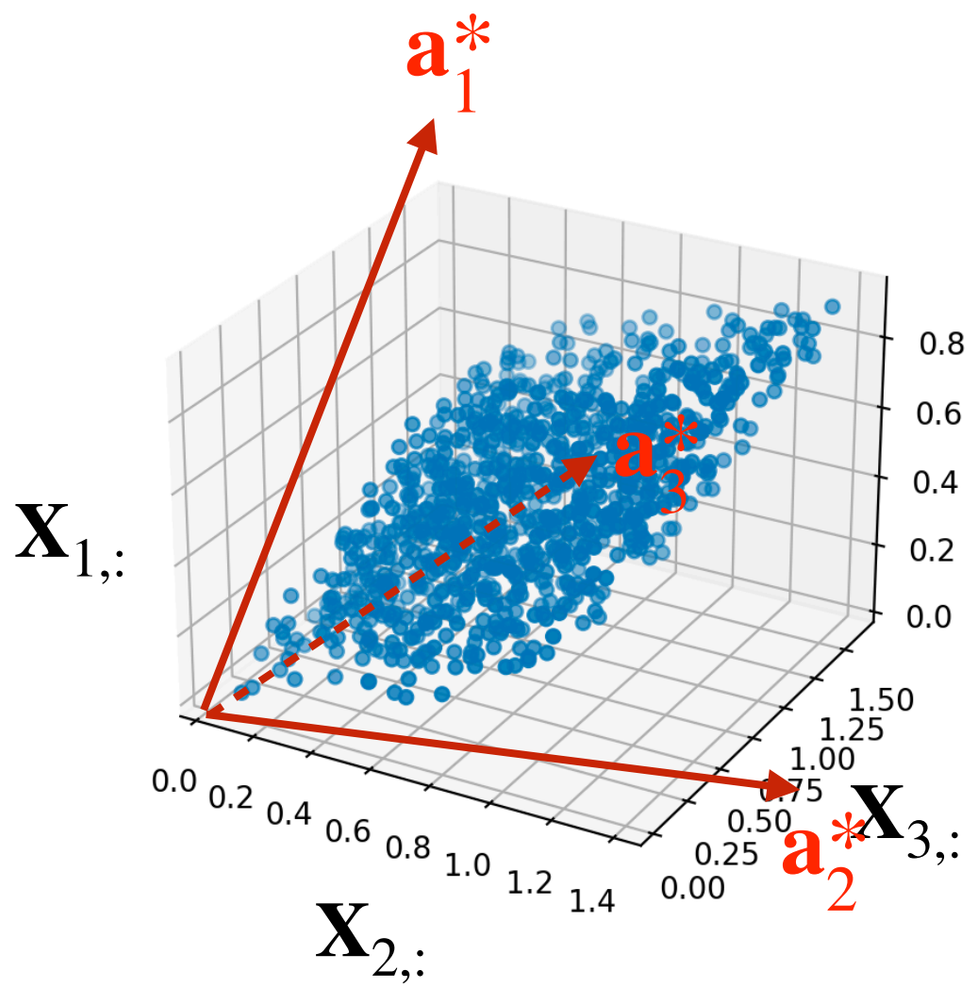
- A simple case: use Euclidean distance (for Gaussian noise)

$$\hat{\mathbf{A}}, \hat{\mathbf{S}} = \arg \min_{\mathbf{A} \geq 0, \mathbf{S} \geq 0} \frac{1}{2} \|\mathbf{X} - \mathbf{AS}\|_F^2$$

Geometric interpretation of plain NMF

- Key to understand the problem and design new algorithms
  - The sources are contained into the nonnegative orthant
  - The dataset is contained within a cone, which is (**at most**) limited by the columns of **A***





Plain NMF: minimization of the cost function

How to solve the optimization problem?

$$\hat{\mathbf{A}}, \hat{\mathbf{S}} = \arg \min_{\mathbf{A} \geq 0, \mathbf{S} \geq 0} \frac{1}{2} \|\mathbf{X} - \mathbf{AS}\|_F^2$$

$$= \arg \min_{\mathbf{A},\mathbf{S}} \frac{1}{2} \|\mathbf{X} - \mathbf{AS}\|_F^2 + \iota_{\cdot \geq 0}(\mathbf{A}) + \iota_{\cdot \geq 0}(\mathbf{S})$$

- We can hope to find a local minimum of the cost function using PALM

- It is still a non-convex problem **(NMF is a NP-hard problem)**, entailing spurious critical points

Plain NMF: minimization of the cost function

$$\arg\min_{\mathbf{A},\mathbf{S}} \frac{1}{2}\|\mathbf{X}-\mathbf{AS}\|_F^2 + \iota_{\cdot\geq 0}(\mathbf{A}) + \iota_{\cdot\geq 0}(\mathbf{S})$$

- **PALM**

Initialize $\mathbf{S}^{(0)}, \mathbf{A}^{(0)}$

$k = 0$

while **not** converged do:

$\mathbf{S}^{(k+1)} = \Pi_{\cdot\geq 0}(\mathbf{S}^{(k)} - \gamma \mathbf{A}^{(k)T}(\mathbf{A}^{(k)}\mathbf{S}^{(k)} - \mathbf{X}))$

$\mathbf{A}^{(k+1)} = \Pi_{\cdot\geq 0}(\mathbf{A}^{(k)} - \eta(\mathbf{A}^{(k)}\mathbf{S}^{(k+1)} - \mathbf{X})\mathbf{S}^{(k+1)})$

$k \leftarrow k+1$

end

return $\mathbf{S}^{(k)}, \mathbf{A}^{(k)}$

Multiplicative update (MU) algorithm: motivation

$$\arg\min_{\mathbf{A},\mathbf{S}} \frac{1}{2}\|\mathbf{X}-\mathbf{AS}\|_F^2 + \iota_{\cdot\geq 0}(\mathbf{A}) + \iota_{\cdot\geq 0}(\mathbf{S})$$

Other method : the **multiplicative update**

The MU algorithm is a frequently-found in the NMF literature algorithm:

- Historically, it was the first algorithm for NMF (before PALM)
- It is **easily implemented**
- It **does not require hyperparameter tuning**
- It **scales well** (scales linearly with the input $\mathbf{X}$ dimension and the number of sources n)
- Although performing worse than SOA when the Frobenius norm is used as data-fidelity term, it **works well with KL distances** (or more generally, with β divergences with $\beta < 2$).

MU algorithm

- **MU algorithm for Frobenius norm**

Initialize $\mathbf{S}^{(0)}, \mathbf{A}^{(0)}$ with **positive** (non-zeros) entries

$k = 0$

while **not** converged do:

$$\mathbf{S}^{(k+1)} = \mathbf{S}^{(k)} \odot \frac{\mathbf{A}^{(k)T}\mathbf{X}}{\mathbf{A}^{(k)T}\mathbf{A}^{(k)}\mathbf{S}^{(k)}}$$

$$\mathbf{A}^{(k+1)} = \mathbf{A}^{(k)} \odot \frac{\mathbf{X}\mathbf{S}^{(k+1)T}}{\mathbf{A}^{(k)}\mathbf{S}^{(k+1)}\mathbf{S}^{(k+1)T}}$$

$k \leftarrow k + 1$

end

return $\mathbf{S}^{(k)}, \mathbf{A}^{(k)}$

where the $\odot$ and $\frac{}{}$ operations have to be understood elementwise (*i.e.* $\odot$ is the Hadamar product)

- The above algorithm can been proved to converge and to monotonically decrease the cost function.

MU as a fixed point iteration of the KKT conditions

Coming back to the original problem:

$$\arg \min_{\mathbf{A}\in\mathbb{R}_+^{m\times n},\, \mathbf{S}\in\mathbb{R}_+^{n\times t}} \frac{1}{2}\|\mathbf{X}-\mathbf{AS}\|_F^2$$

For instance, for $\mathbf{S}$, it yields a Lagrangian:

$$\mathscr{L}(\mathbf{S},\Theta) = \frac{1}{2}\underbrace{\|\mathbf{X}-\mathbf{AS}\|_F^2}_{=f(\mathbf{S})} - \Theta \odot \mathbf{S}$$

(minus because the constraint is not negativity but nonnegativity)

We can then solve the following KKT conditions:

$$(1) \quad \nabla_{\mathbf{S}}\mathscr{L} = \nabla_{\mathbf{S}} f - \Theta = \mathbf{A}^T(\mathbf{AS}-\mathbf{X}) - \Theta = 0$$ (the Lagrangian gradient is 0)

$$(2) \quad \Theta \geq 0$$ (the Lagrangian parameter are nonnegative)

$$(3) \quad \mathbf{S} \geq 0$$ (the constraint is filled)

$$(4) \quad \Theta \odot \mathbf{S} = 0$$ (slackness condition)

(1) and (2) imply that $\nabla_{\mathbf{S}} f \geq 0$

(1) and (4) imply that $\forall i \in [1,n], j \in [1,t],\ [\nabla_{\mathbf{S}} f]_{i,j} = 0$ or $\mathbf{S}_{ij} = 0$

MU as a fixed point iteration of the KKT conditions

(1) and (4) imply that $\forall i \in [1,n], j \in [1,t]$, $[\nabla_{\mathbf{S}} f]_{i,j} = 0$ or $\mathbf{S}_{ij} = 0$ (slackness condition)

The MU update is based on 2 arguments:

- when $\mathbf{S}_{ij} \geq 0$, the above condition implies that we must have for any optimal point

$$[\mathbf{A}^T\mathbf{X}]_{ij} = [\mathbf{A}^T\mathbf{A}\mathbf{S}]_{ij} \quad \text{(fixed point condition)}$$

- if $[\nabla_{\mathbf{S}} f]_{ij} > 0$, a sufficiently small decrease of $\mathbf{S}_{ij}$ decreases the objective function. Therefore, it makes sense to decrease $\mathbf{S}_{ij}$ if $[\mathbf{A}^T\mathbf{A}\mathbf{S}]_{ij} > [\mathbf{A}^T\mathbf{X}]_{ij}$. If $[\nabla_{\mathbf{A}} f]_{ij} < 0$, the converse if true.

Altogether, this makes that multiplying the current $\mathbf{S}$ by the ratio of the positive and the negative parts of the gradient

$$\mathbf{S}^{(k+1)} = \mathbf{S}^{(k)} \odot \frac{\mathbf{A}^{(k)T}\mathbf{X}}{\mathbf{A}^{(k)T}\mathbf{A}^{(k)}\mathbf{S}^{(k)}}$$

might lead to a relevant update.

MU as a rescaled gradient descent

- Recall PALM update of a single source $\mathbf{s}$: $\mathbf{s}^{(k+1)} = \Pi_{\cdot \geq 0}(\mathbf{s}^{(k)} - \gamma \nabla h(\mathbf{A}^{(k)}, \mathbf{s}^{(k)}))$
- To try to accelerate the convergence speed, we can incorporate second-order statistics:

$$\mathbf{s}^{(k+1)} = \Pi_{\cdot \geq 0}(\mathbf{s}^{(k)} - \mathbf{H}^{-1} \nabla h(\mathbf{A}^{(k)}, \mathbf{s}^{(k)})) \quad \text{(projected Newton step)}$$

where $\mathbf{H}$ is the Hessian matrix (containing all the order-two derivatives of the data-fidelity term).

- However, computing the $\mathbf{H}$ matrix is usually very costly. Therefore, it is often approximated with another (usually positive-definite) matrix $\mathbf{B}$:

$$\mathbf{s}^{(k+1)} = \Pi_{\cdot \geq 0}(\mathbf{s}^{(k)} - \mathbf{B} \nabla h(\mathbf{A}^{(k)}, \mathbf{s}^{(k)}))$$

- In particular, MU corresponds to a projected rescaled gradient method, where $\mathbf{B}$ is a diagonal matrix. Considering

$$\mathbf{B} = \texttt{diag}\left(\frac{\mathbf{s}^{(k)}}{\mathbf{A}^T\mathbf{A}\mathbf{s}^{(k)}}\right),$$

we obtain the MU update

$$\mathbf{s}^{(k+1)} = \frac{\mathbf{s}^{(k)} \odot \mathbf{A}^T\mathbf{x}}{\mathbf{A}^T\mathbf{A}\mathbf{s}}$$

Zero-locking phenomenon

- In a coefficient becomes 0 in the MU updates $\mathbf{S}^{(k+1)} = \mathbf{S}^{(k)} \odot \frac{\mathbf{A}^{(k)T}\mathbf{X}}{\mathbf{A}^{(k)T}\mathbf{A}^{(k)}\mathbf{S}^{(k)}}$ and $\mathbf{A}^{(k+1)} = \mathbf{A}^{(k)} \odot \frac{\mathbf{X}\mathbf{S}^{(k+1)T}}{\mathbf{A}^{(k)}\mathbf{S}^{(k+1)T}\mathbf{S}^{(k+1)}}$, it stays zero forever.
- This is known as the zero-locking phenomenon and impedes MU results.

- **Modified MU**

Initialize $\mathbf{S}^{(0)}, \mathbf{A}^{(0)}$ with **positive** (non-zeros) entries

$k = 0$

while **not** converged do:

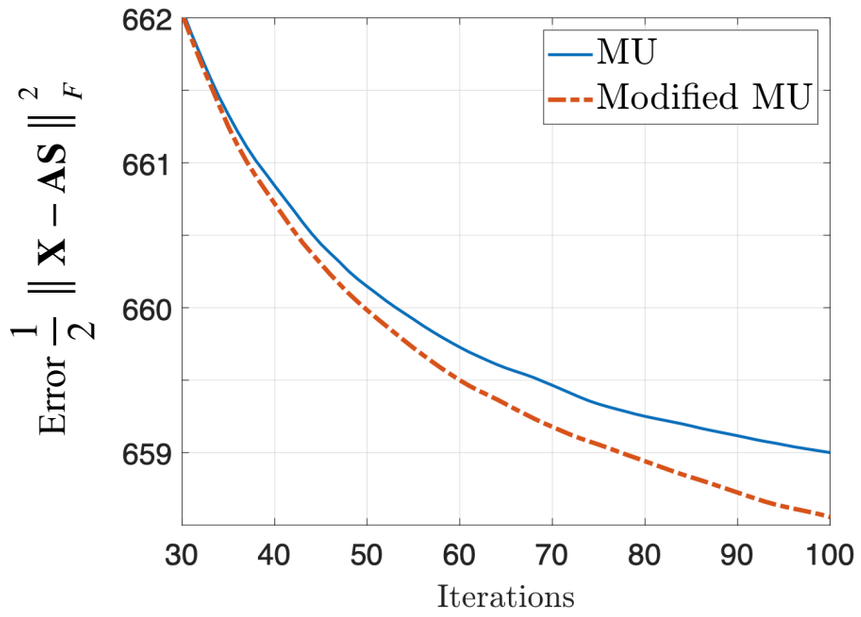
$$\mathbf{S}^{(k+1)} = \max\left(\epsilon, \mathbf{S}^{(k)} \odot \frac{\mathbf{A}^{(k)T}\mathbf{X}}{\mathbf{A}^{(k)T}\mathbf{A}^{(k)}\mathbf{S}^{(k)}}\right)$$

$$\mathbf{A}^{(k+1)} = \max\left(\epsilon, \mathbf{A}^{(k)} \odot \frac{\mathbf{X}\mathbf{S}^{(k+1)T}}{\mathbf{A}^{(k)}\mathbf{S}^{(k+1)T}\mathbf{S}^{(k+1)}}\right)$$

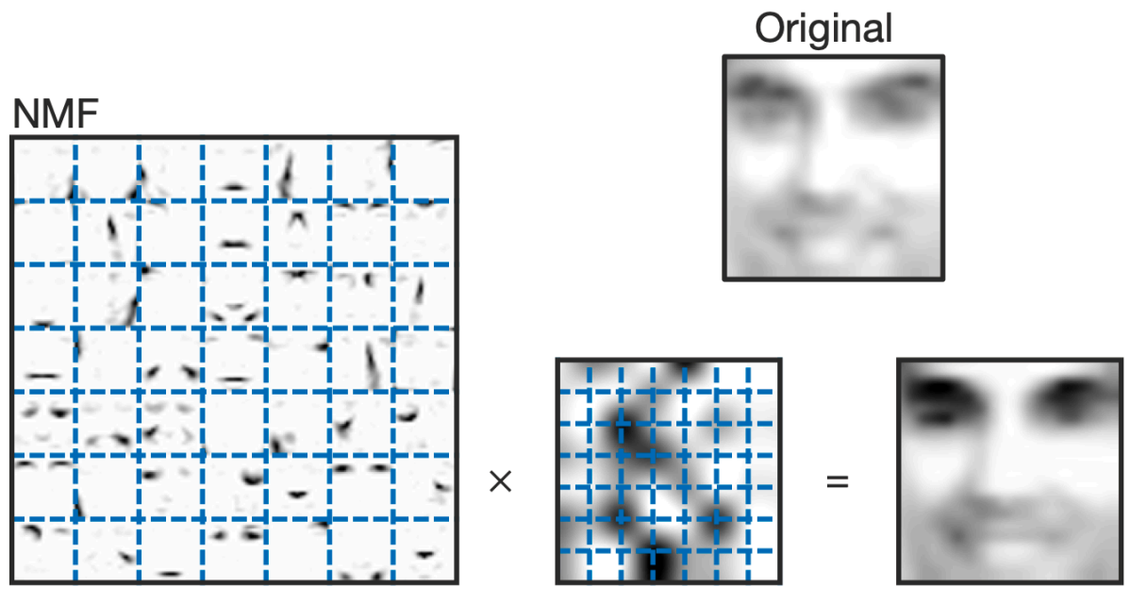
$k \leftarrow k + 1$

end

return $\mathbf{S}^{(k)}, \mathbf{A}^{(k)}$



Original illustration of the MU: decomposing images in patches



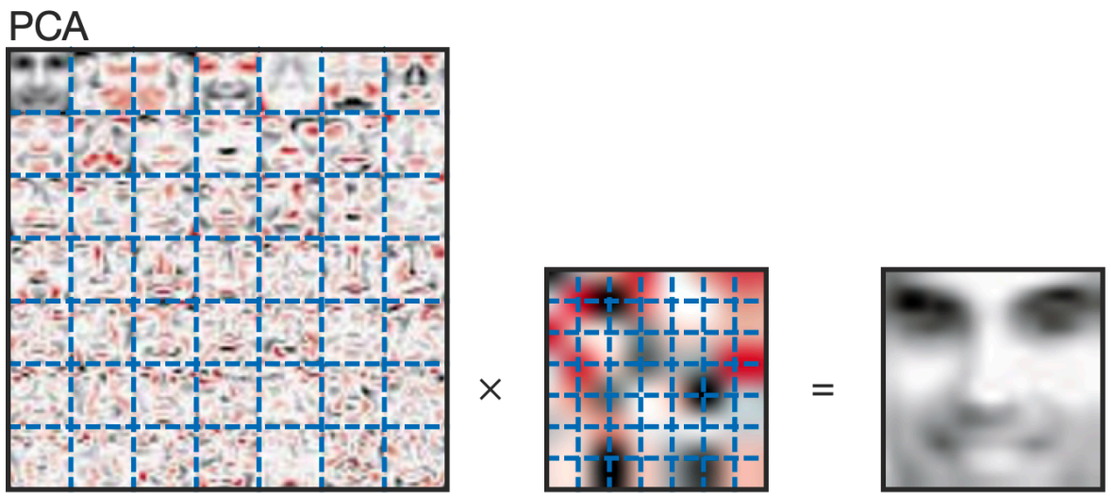


Figure 1 Non-negative matrix factorization (NMF) learns a parts-based representation of faces, whereas vector quantization (VQ) and principal components analysis (PCA) learn holistic representations. The three learning methods were applied to a database of $m = 2{,}429$ facial images, each consisting of $n = 19 \times 19$ pixels, and constituting an $n \times m$ matrix V. All three find approximate factorizations of the form $V \approx WH$, but with three different types of constraints on W and H, as described more fully in the main text and methods. As shown in the 7×7 montages, each method has learned a set of $r = 49$ basis images. Positive values are illustrated with black pixels and negative values with red pixels. A particular instance of a face, shown at top right, is approximately represented by a linear superposition of basis images. The coefficients of the linear superposition are shown next to each montage, in a 7×7 grid, and the resulting superpositions are shown on the other side of the equality sign. Unlike VQ and PCA, NMF learns to represent faces with a set of basis images resembling parts of faces.

Outline

- **Plain NMF**

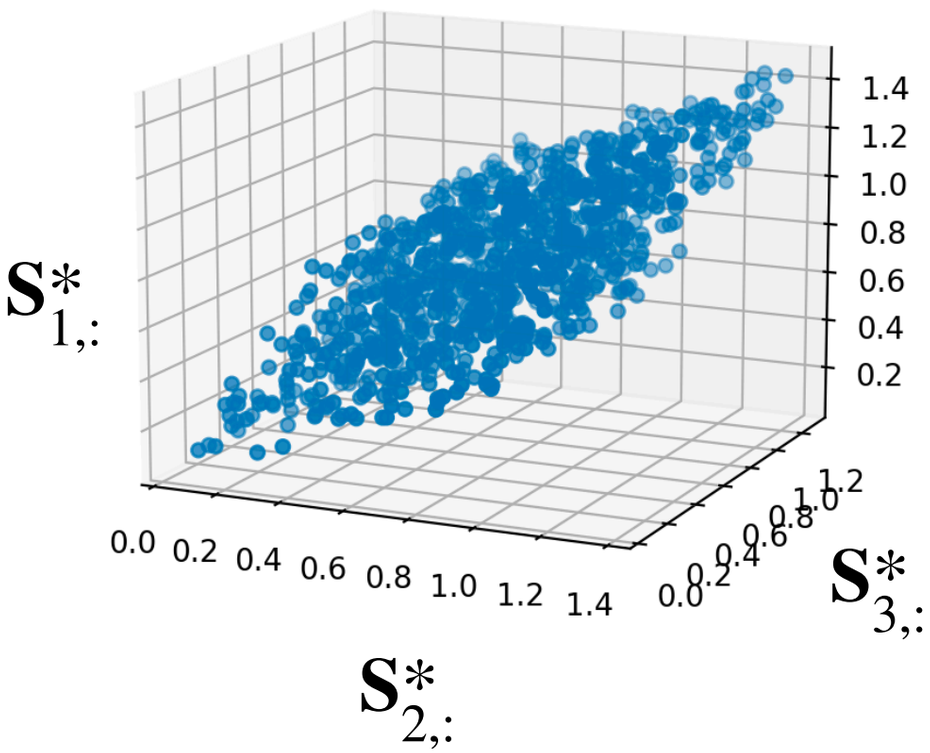
- **Near-separable NMF**
  - Definition
  - Algorithms
  - Recovery guarantees

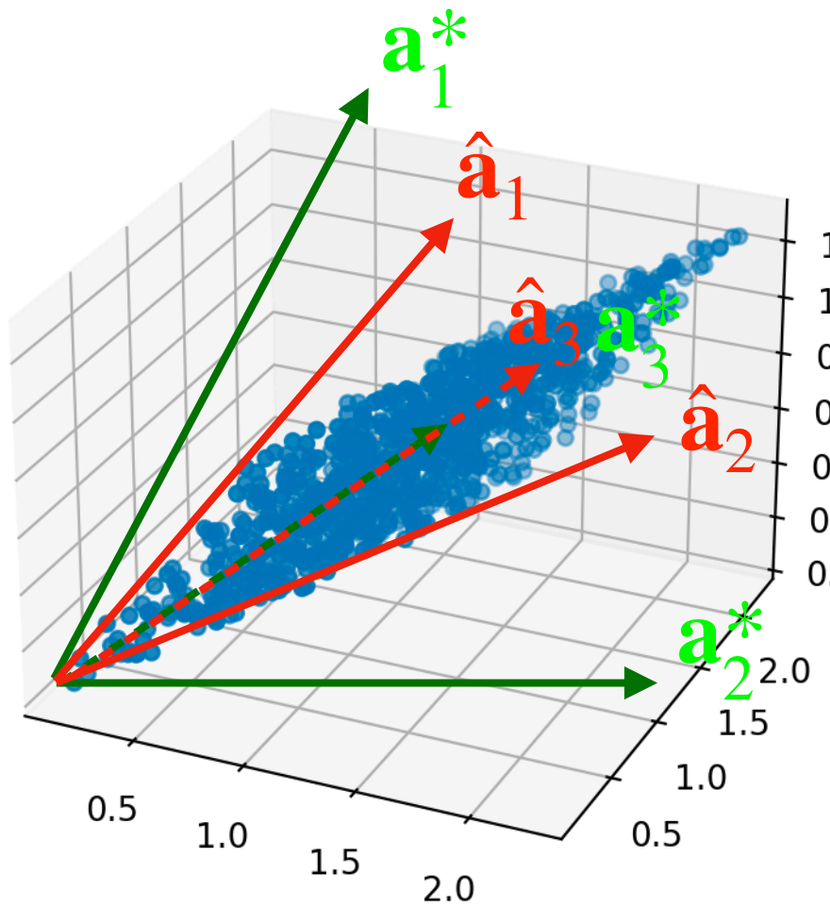
- **Minimum volume NMF**

- **Extensions to Deep Learning**

Identifiability of plain NMF

- Question : are we guaranteed to recover the sources at the origin of the dataset **X**, up to a permutation and a scaling factor?





- Answer: plain NMF does not guarantee the unicity of the solution
- Said differently: plain NMF is still an ill-posed problem (nonnegativity is not a strong enough prior)

=> Several sub-families of NMF have emerged, to make the problem well-posed.

Outline

- Question : are we guaranteed to recover the sources at the origin of the dataset **X**, up to a permutation and a scaling factor?

=> We will see two additional constraints on **A***, **S*** to make NMF well-posed (and enabling theoretical recovery guarantees !!):

- Near-separability
- Minimum-volume NMF

During the TP, you will play with the three families of algorithms !

Note that there exist other sub-families of NMF (sparse NMF, orthogonal NMF…).

Separable NMF (1/3)

- We have seen that plain NMF suffers from to flaws :
  - NP-hardness (the problem is **computationally intractable**)
  - Identifiability (**X does not admit a unique factorization**)

Separable NMF overcomes this two flaws.

=> Definition: near separable NMF assumes that $\mathbf{X} = \mathbf{A}^*\mathbf{S}^*$ with $\mathbf{S}^*$ a separable matrix.

Separable matrix

A matrix $\mathbf{S}^* \in \mathbb{R}^{n \times t}$ is said to be separable if $\texttt{cone}(\mathbf{S}^*) = \mathbb{R}^n_+$.

$$\text{with } \texttt{cone}(\mathbf{S}) = \{\mathbf{y} \in \mathbb{R}^n \,|\, \mathbf{y} = \mathbf{S}\mathbf{x}, \mathbf{x} \geq 0\}$$

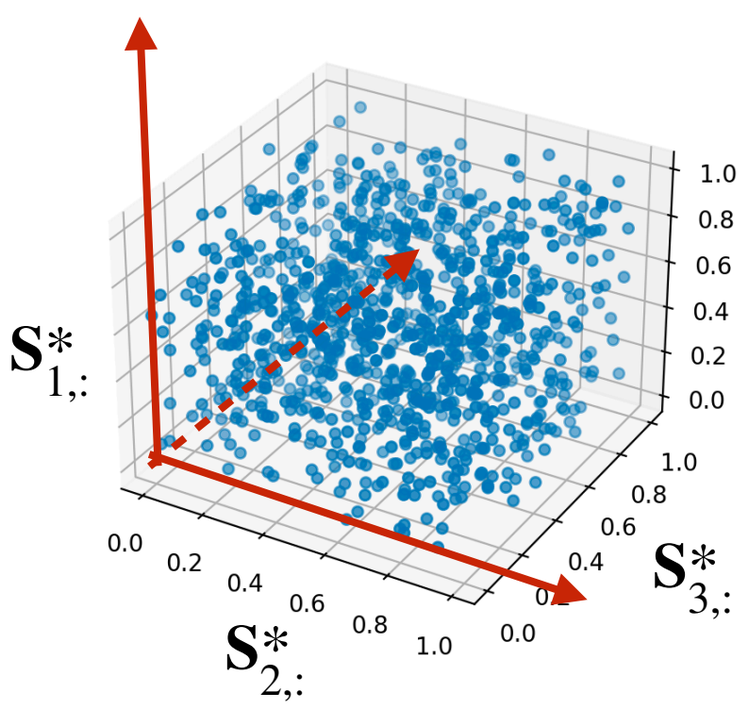
Separable NMF (2/3)

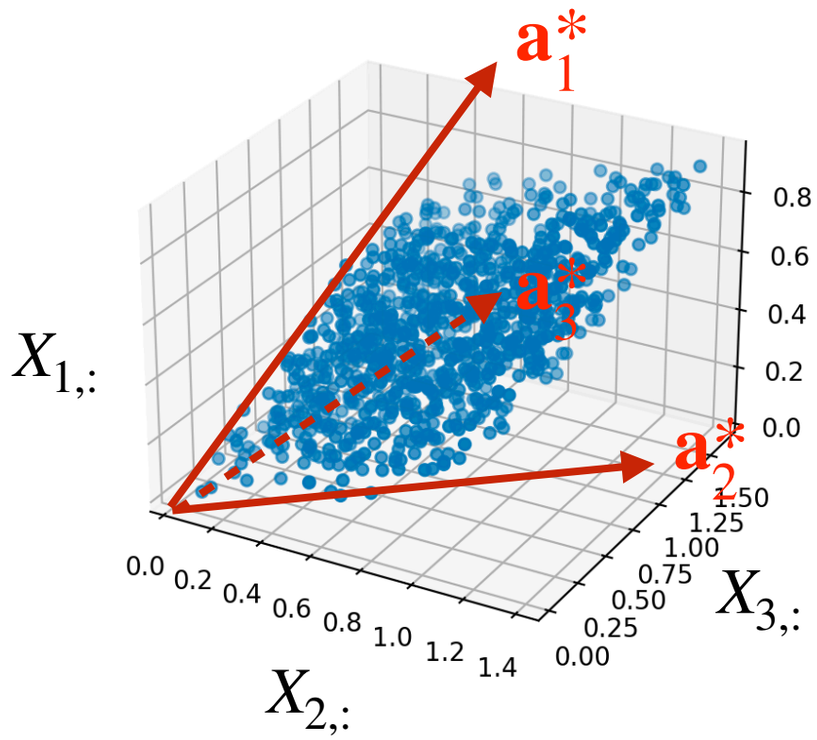
Separable matrix

A matrix $\mathbf{S}^* \in \mathbb{R}^{n \times t}$ is said to be separable if $\texttt{cone}(\mathbf{S}^*) = \mathbb{R}^n_+$.

$$\text{with } \texttt{cone}(\mathbf{S}) = \{\mathbf{y} \in \mathbb{R}^n \,|\, \mathbf{y} = \mathbf{S}\mathbf{x}, \mathbf{x} \geq 0\}$$

=> separability of $\mathbf{S}^*$ requires all the unit columns to « hide » (up to a scaling) among the columns of $\mathbf{S}^*$





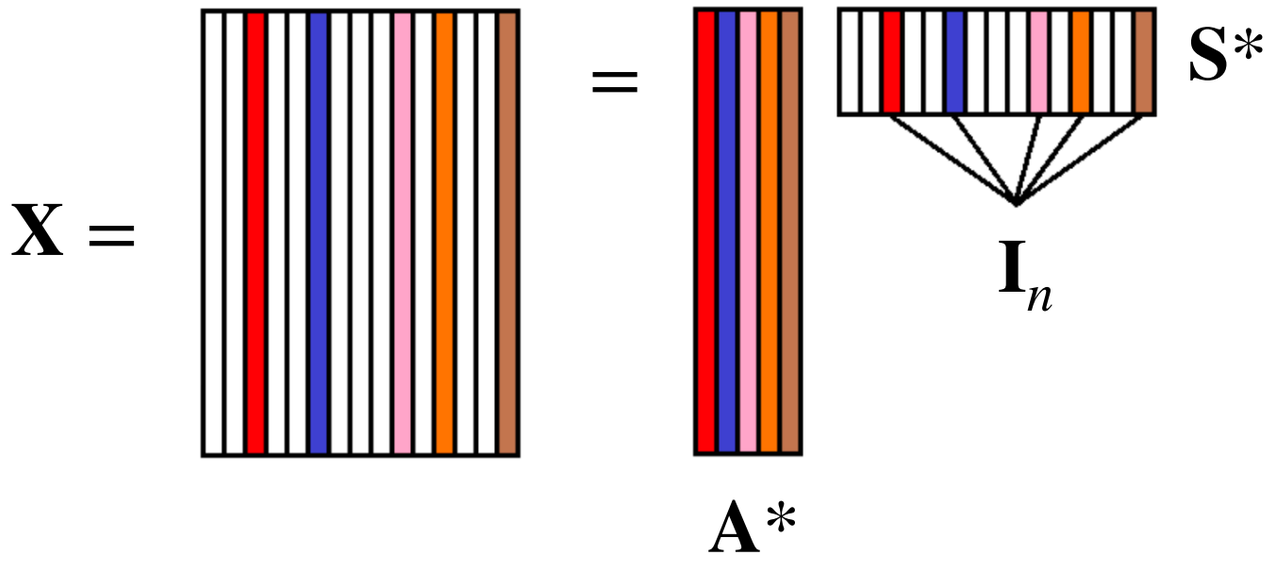
Separable NMF (3/3)

Separable NMF

Because of the scaling degree of freedom in the decomposition $\mathbf{X} = \mathbf{A}^*\mathbf{S}^*$, separability of $\mathbf{S}^*$ amounts to assume that there exists an index set $\mathcal{K} \subset [1,t]$ such that $\mathbf{A}^* = \mathbf{X}(:, \mathcal{K})$.

The mixing thus becomes:

$$\mathbf{X} = \mathbf{X}(:, \mathcal{K})\mathbf{S}^*$$



=> Under the separable model, the problem becomes identifiable

Near-separable factorization

In practice, most dataset $\mathbf{X}$ do not admit an exact separable NMF decomposition, due to **noise** and **model misfit**. Let us thus rather assume the following more realistic assumptions:

Near-Separable NMF

The matrix $\mathbf{X} \in \mathbb{R}^{m \times t}$, admits a near-separable factorization of rank n if it has the form:

$$\mathbf{X} = \underbrace{\mathbf{A}^*\mathbf{S}^*(:,\mathscr{K})}_{=\mathbf{X}_{\text{noiseless}}(:,\mathscr{K})}\mathbf{S}^* + \mathbf{N} \quad \text{with } \|\mathbf{n}_j\| \leq \epsilon \text{ for all } j$$

with the additional assumptions

(i) $\forall k \in [1,t], \|\mathbf{n}_k\|_p \leq \epsilon$ (bounded noise assumption)

(ii) $\max \|\mathbf{a}_k\|_p = 1$ (unitary mixing matrices) with $p = 1$ or 2

(iii) $\forall k \in [1,t], \|\mathbf{s}_k\|_p \leq 1$ (unitary sources columns)

(iv) $\alpha = \min_{1 \leq i \leq n, \mathbf{x} \in \Delta} \|\mathbf{a}_i - \mathbf{A}_{\setminus i}\mathbf{x}\| > 0$ (A is α-robust simplicial)

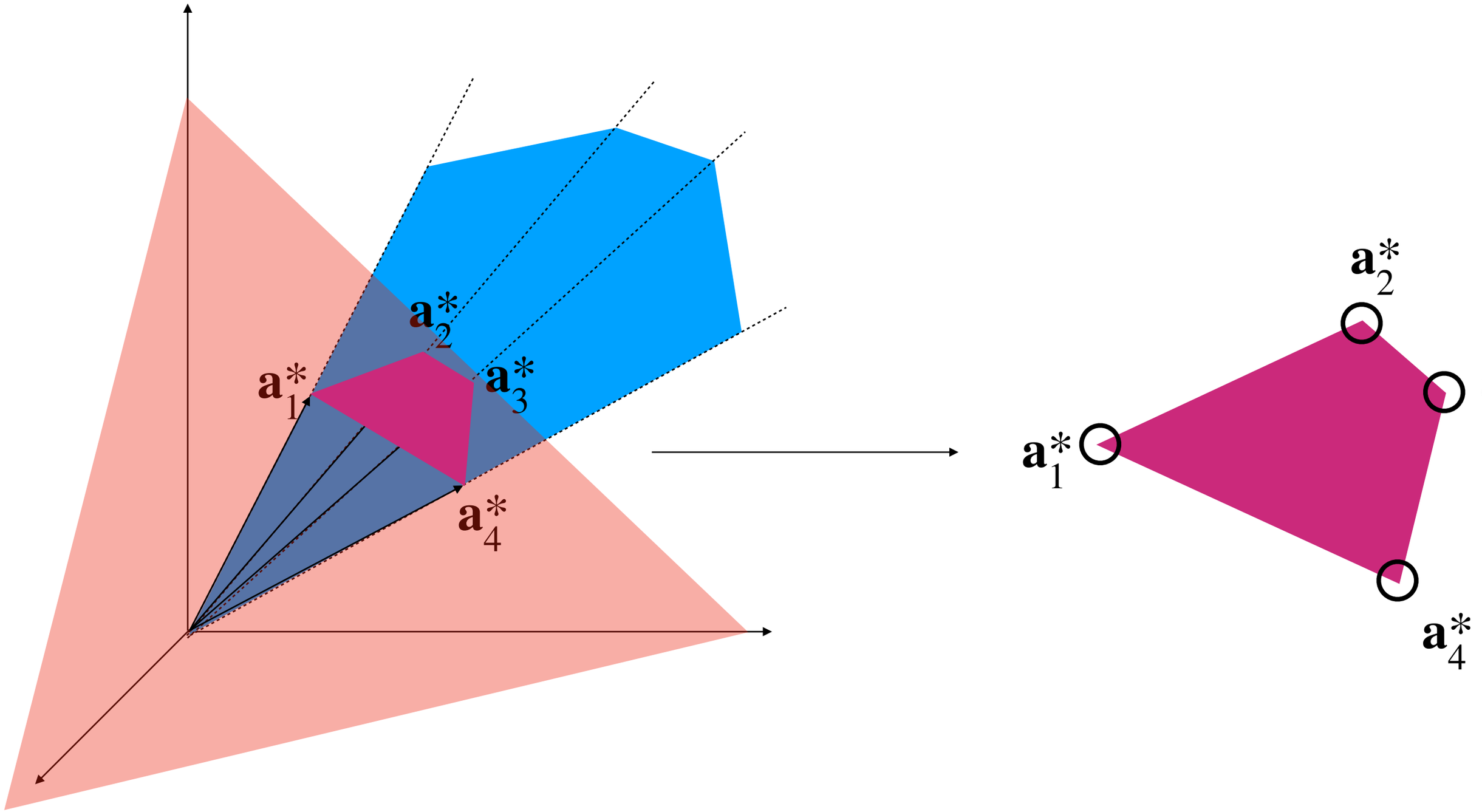
Under these assumptions, the problem is **much easier**, as finding $\mathbf{A}$ only amounts to find the indices $\mathscr{K}$.

Additional remarks for near-separable NMF

- Any matrix $\mathbf{X}$ is near-separable, when choosing ϵ big enough. We will focus on case in which ϵ is small enough.
- Similarly, we assume n to be small with respect to t.
- The column unit-norm of $\mathbf{A}$ is natural in many applications (in particular, HSU for astrophysics)
- The unit norm of $\mathbf{S}$ assumption (iii) $\forall k \in [1,t], \|\mathbf{s}_k\|_p \leq 1$ is not necessary (but convenient). In addition, it can be done without loss of generality by normalizing the columns of $\mathbf{X}$:

$$
\begin{aligned}
1 = \sum_{i=1}^{m} x_{ij} &= \sum_{i=1}^{m} \sum_{k=1}^{n} a_{ik}^{*} s_{kj}^{*} \\
&= \sum_{k=1}^{n} s_{kj}^{*} \sum_{i=1}^{m} a_{ik}^{*} \\
&= \sum_{k=1}^{n} s_{kj}^{*}
\end{aligned}
$$

Geometric interpretation



- The columns of $\mathbf{A}^*$ are the vertices of a polytope containing all the normalized data columns

Assumptions we will use for near-separable NMF

$$\mathbf{X} = \mathbf{A}^*\mathbf{S}^* + \mathbf{N}$$

- **Nonnegativity** of **A** and **S**

- **Sum-to-one** columns of **S***

$$\sum_{k=1}^{n} s_{ki} = 1 \text{ for all } k \in [1,n]$$

We will write $\mathbf{S}^* \in \Delta$ if $\mathbf{S}^*$ satisfies the two above conditions

- **A** is α-**robust simplicial**:

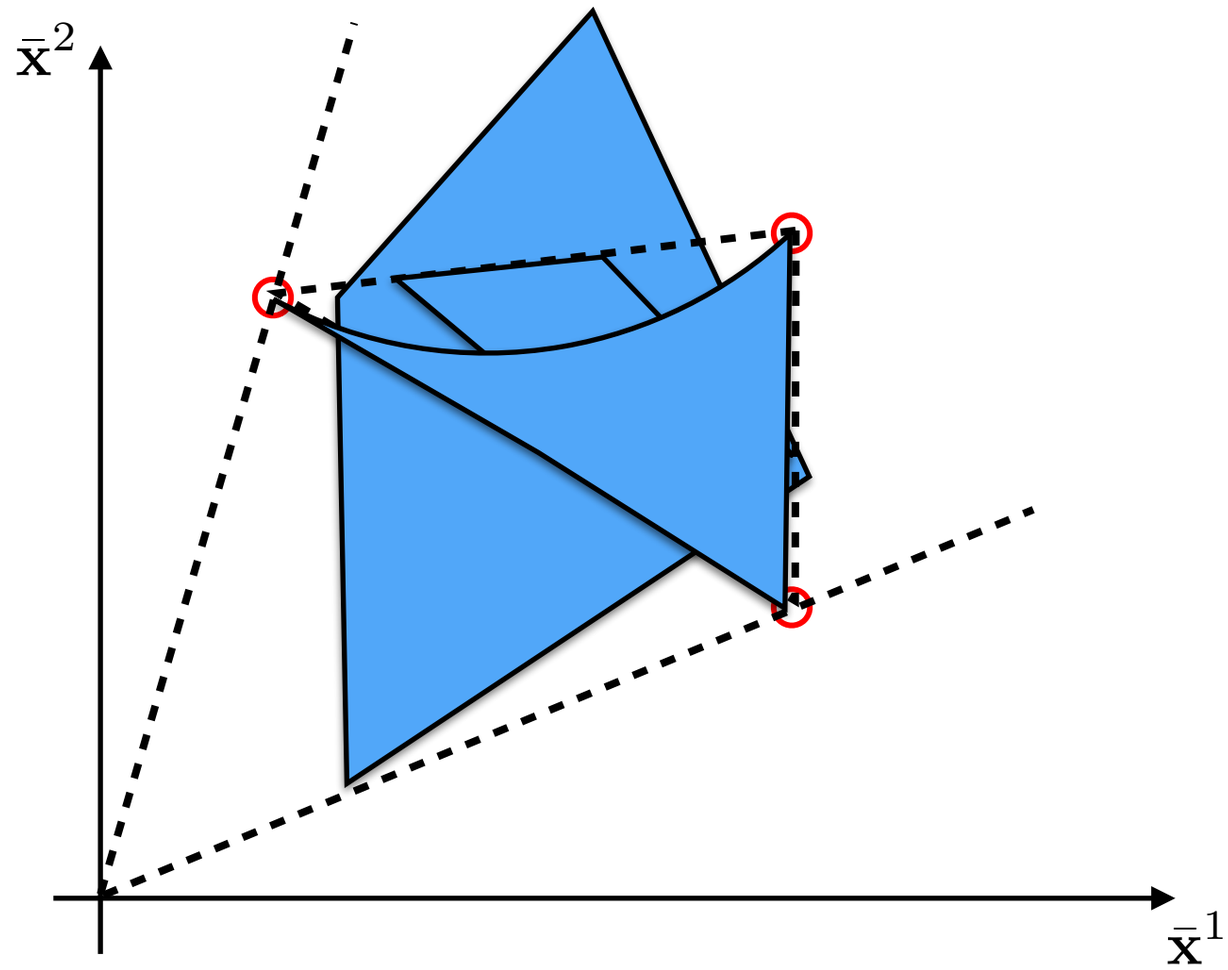
$$\alpha = \min_{1 \le i \le n, \mathbf{x} \in \Delta} \|\mathbf{a}_i - \mathbf{A}_{\setminus i}\mathbf{x}\| > 0$$

- **Near-separable** NMF

$$\mathbf{X} \simeq \mathbf{A}[\mathbf{I}_r, \mathbf{S}']\mathbf{P}$$

- **Bounded noise** **N**

$$\max_i \|\mathbf{n}_i\|_1 \le \epsilon$$



A column of **A***

Limits imposed on the data set

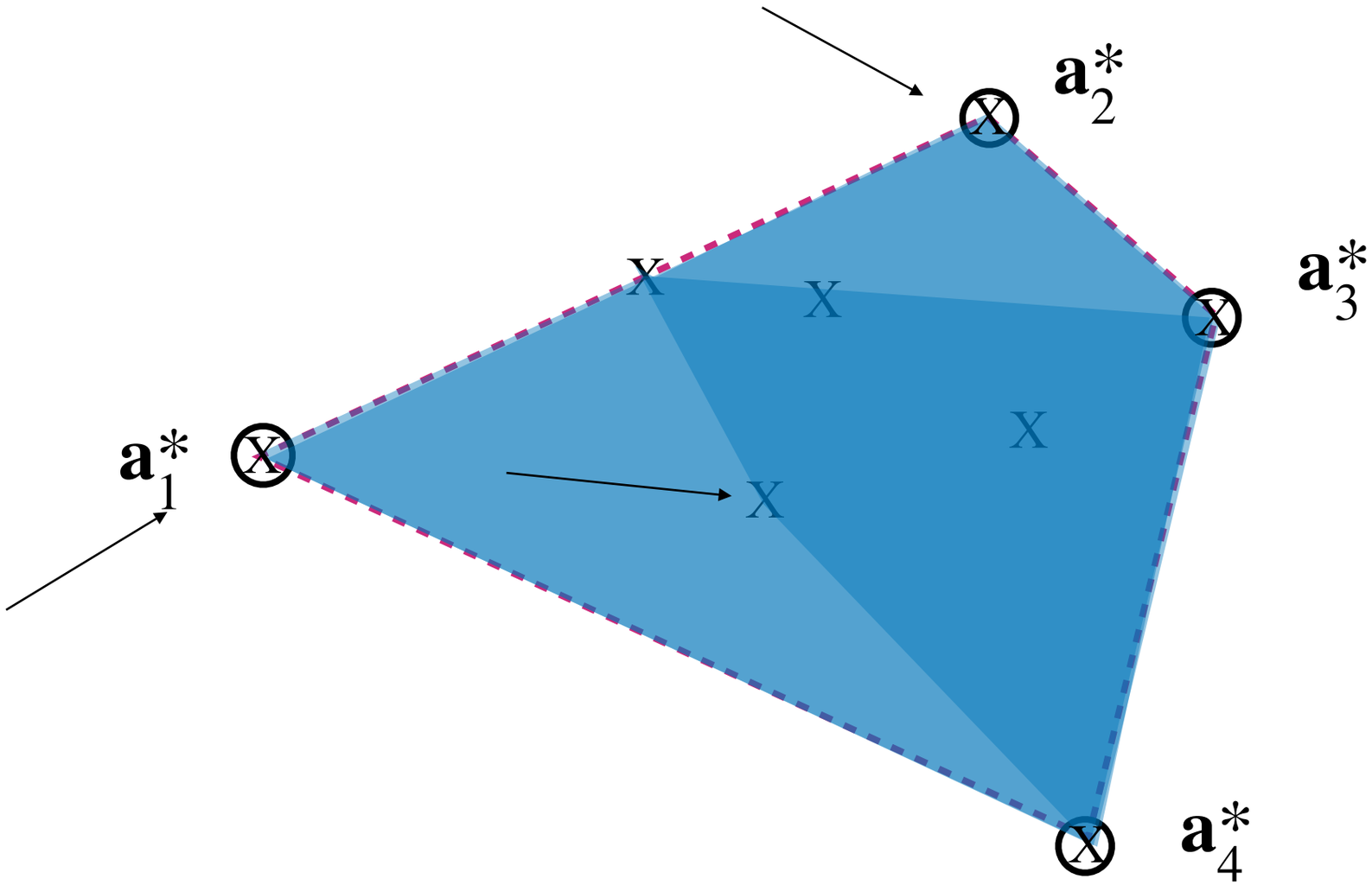
An example of data set

Outline

- **Plain NMF**
    - Problem statement
    - Optimization framework
        - PALM
        - Multiplicative updates
- **Near-separable NMF**
    - Definition
    - Algorithms: brute force, greedy and self-dictionary algorithms
    - Recovery guarantees
- **Minimum volume NMF**

A first provable algorithm: Arora

- The first provably robust algorithm for near-separable NMF has been proposed by Arora
- It is a brute-force algorithm, based on the geometric interpretation of near-separable NMF.
- Let us first assume that there is no noise: $\mathbf{X} = \mathbf{AS}$. w.l.o.g., let us further assume that there is no duplicated column.
- Arora's algorithm proposes to loop over all the data columns and check for the ones that are not included in the convex hull of the other ones.



Arora brute force algorithm

Brute Force algorithm

Preprocessing: remove all the duplicated columns of $\mathbf{X}$

Let $\mathscr{K} = \{\}$

for $k \in [1,t]$

if $\min_{\mathbf{s}\in\Delta} \left\| \mathbf{x}_k - \mathbf{X}_{\setminus k}\mathbf{s} \right\|_1 > 0$ where Δ is the unit simplex

$\mathscr{K} = \mathscr{K} \cup \{k\}$

end if

end for

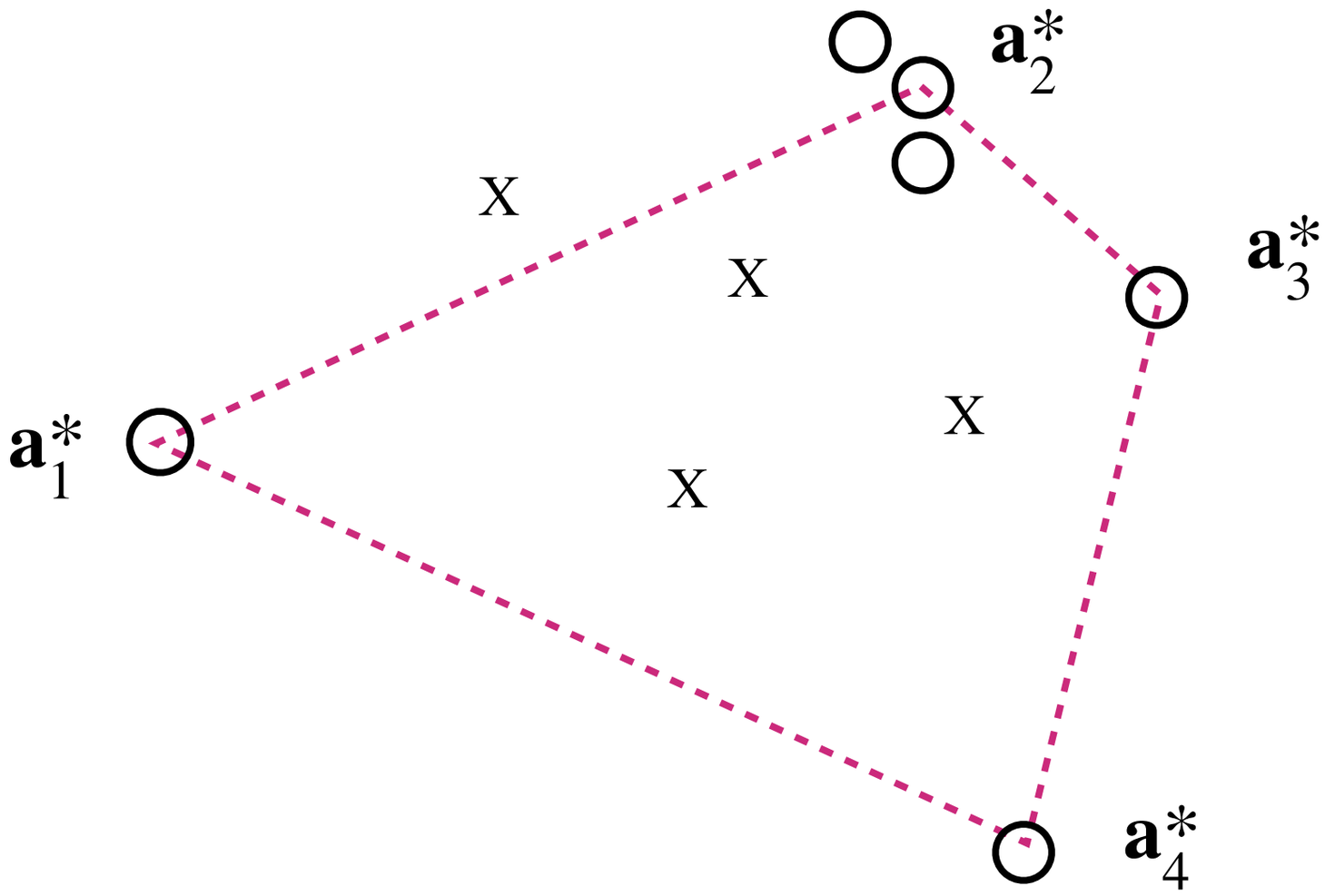
return $\mathcal{K}$ then, $\mathbf{W} = \mathbf{X}_{\mathscr{K}}$

Remarks:

- Here, $\mathbf{X}$ is assumed to be noiseless
- Other functions that the l_1-norm can be used

Arora's algorithm in the presence of noise

- In practice, things are however slightly more complicated due to the noise
- Two issues:
  - Some mixed point can move outside of the convex hull
  - Some columns of $\mathbf{A}$ can be duplicated up to the noise



Arora brute force algorithm

Brute Force algorithm

Let $\mathscr{K} = \{\}$

for $k \in [1,t]$

$$\text{if } \min_{\mathbf{x}\in\Delta} \left\| \mathbf{x}_j - \mathbf{X}_{\setminus \{i \mid \|\mathbf{x}_i - \mathbf{x}_j\|_1 > 5\epsilon/\alpha + 2\epsilon\}} \mathbf{s} \right\|_2^2 > 2\epsilon$$

$$\mathscr{K} = \mathscr{K} \cup \{k\}$$

end if

end for

Post-processing: clustering of the similar columns in $\mathscr{K}$

return $\mathcal{K}$ then, $\mathbf{W} = \mathbf{X}_{\mathscr{K}}$

where Δ is the unit simplex

$$\max_i \|\mathbf{n}_i\|_1 \leq \epsilon$$

$$\alpha = \min_{1\leq i\leq n, \mathbf{x}\in\Delta} \|\mathbf{a}_i - \mathbf{A}_{\setminus i}\mathbf{x}\| > 0$$

Remarks:

- In practice, α and ϵ might be unknown
- However, the number of sources is not required

Algorithms for near-separable NMF

Unfortunately, Arora algorithm is a brute-force algorithm.

Three families of near-separable NMF algorithms with recovery guarantees:

- **Brute-force** algorithms: Arora (**very costly**)
- **Greedy** algorithms**:** SPA, FAW, VCA, SNPA (**robust**, **fast**)
- **Convex-optimization** algorithms: MLP, self-dictionary (**good robustness** but **costly**)

- **Heuristic** algorithms: PPI, N-FINDR… (**no robustness** guarantees)

SNPA

SNPA algorithm

Input : $\mathbf{X}$

Output : set of indices $\mathcal{K}$ such that $\mathbf{X}_{\mathcal{K}} \simeq \mathbf{A}^*$ (up to a permutation)

Let $\mathbf{R} = \mathbf{X}$, $\mathcal{K} = \{\}$, $k = 1$

while $k \leq n$ **do:**

$p = \arg\max_j \|\mathbf{r}_j\|_2$ % Selection step: recall that the columns of $\mathbf{S}$ sum to 1

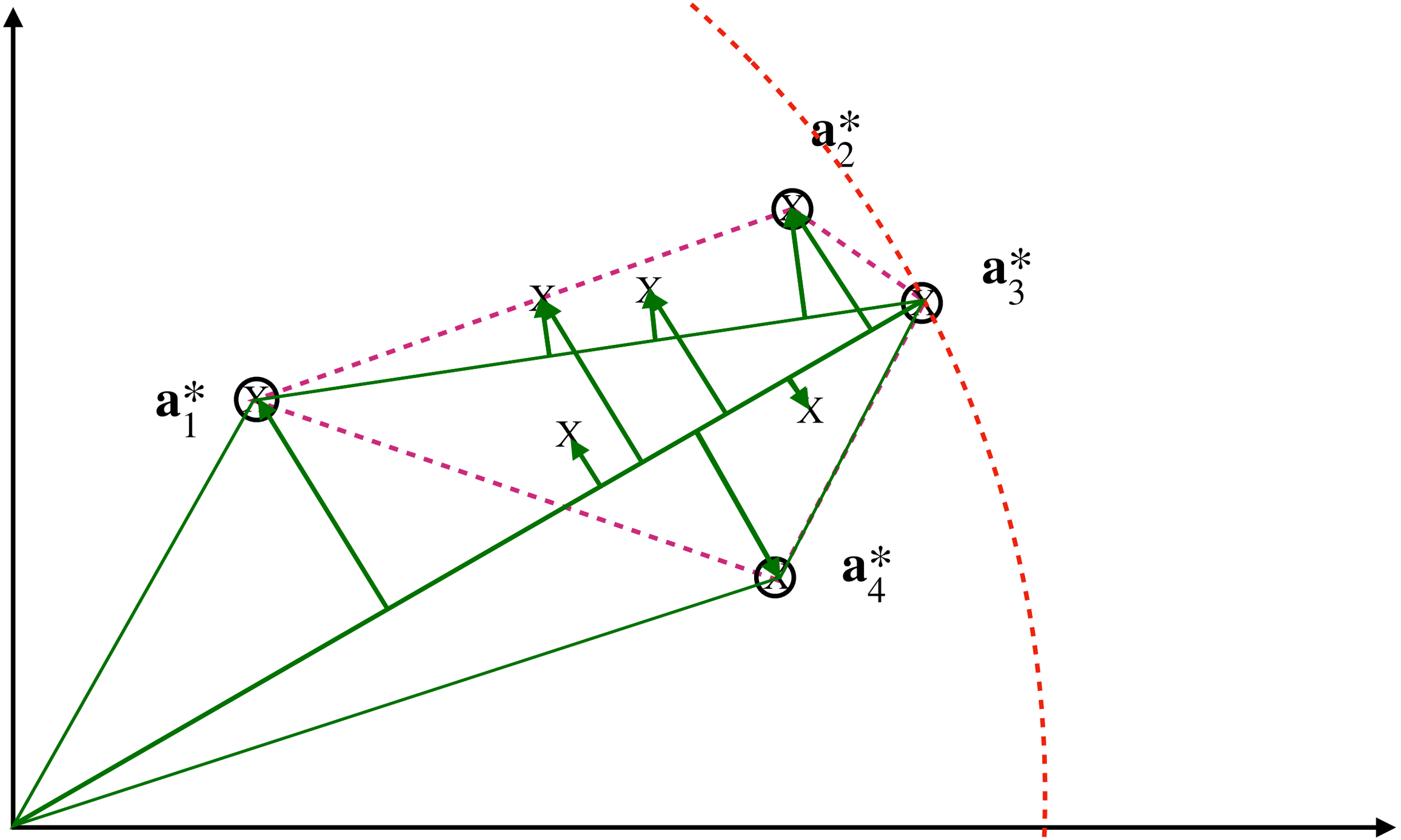
$\mathcal{K} = \mathcal{K} \cup \{p\}$

$\mathbf{R} = \mathbf{X} - \mathbf{X}(:, \mathcal{K})\mathbf{H}^*$ with $\mathbf{H}^* = \arg\min_{\mathbf{H} \in \Delta} \|\mathbf{X} - \mathbf{X}(:, \mathcal{K})\mathbf{H}\|_F^2$ % Projection step

$k \leftarrow k + 1$

end

SNPA graphical example



A more spread algorithm, which will be used for the TP: SPA

SPA algorithm

Input : $\mathbf{X}$

Output : set of indices $\mathcal{K}$ such that $\mathbf{X}_{\mathcal{K}} \simeq \mathbf{A}^*$ (up to a permutation)

Let $\mathbf{R} = \mathbf{X}$, $\mathcal{K} = \{\}$, $k = 1$

while $k \leq n$ **do:**

$p = \arg\max_j \|\mathbf{r}_j\|_2$ % Selection step: recall that the columns of **S** sum to 1

$\mathcal{K} = \mathcal{K} \cup \{p\}$

$\mathbf{R} = \left(\mathbf{I} - \frac{\mathbf{r}_p \mathbf{r}_p^T}{\|\mathbf{r}_p\|_2^2} \right) \mathbf{R}$ % Projection step

$k \leftarrow k + 1$

end

N.B. : here the nonnegativity is not taken into account in the projection step

Self-dictionary algorithms

- An issue with the above exact algorithms is that they consider each data column one by one
- By contrast, self-dictionary algorithms consider all the data columns simultaneously

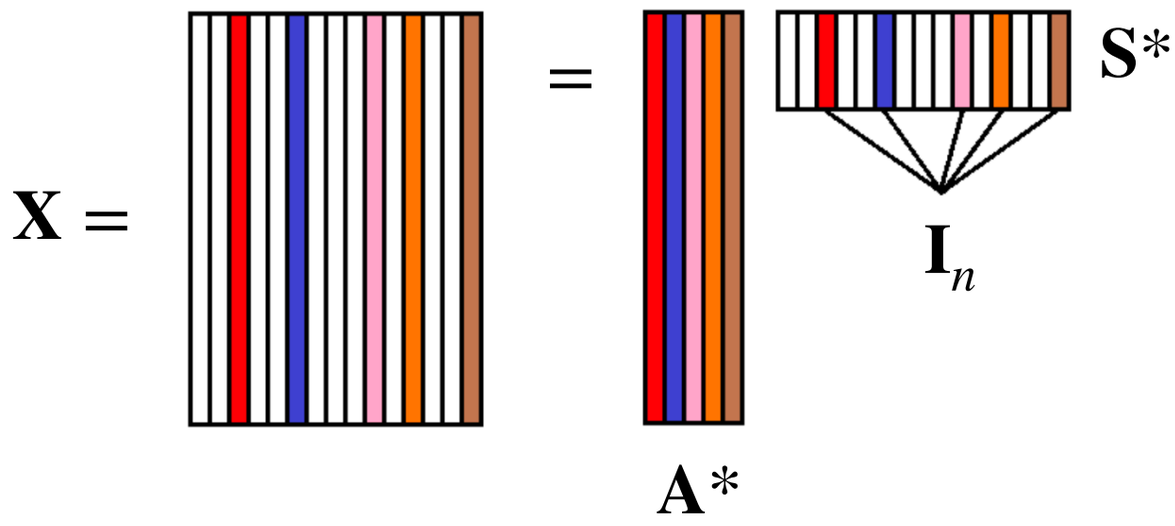
=> in practice, it makes them arguably more robust to noise

Self-dictionary algorithms

- Explicitly use the fact that $\mathbf{A}^* \simeq \mathbf{X}(:, \mathscr{K})$ for some subset $\mathscr{K}$, by leveraging this constraint inside of the minimization of a cost function
- Let us assume that there is no duplicated columns of $\mathbf{A}^*$ in $\mathbf{X}$.

- Let us introduce the set $\mathscr{Y} = \{\mathbf{Y} \in \mathbb{R}_+^{t \times t} \,|\, \mathbf{X} = \mathbf{XY}\}$

  => contains all nonnegative matrices that can be used to reconstruct $\mathbf{X}$ from itself.



Self-dictionary algorithms: noiseless case

- Let us introduce the set $\mathcal{Y} = \{\mathbf{Y} \in \mathbb{R}_+^{t \times t} \,|\, \mathbf{X} = \mathbf{XY}\}$
- $I_n \in \mathcal{Y}$, which is not very useful

- Considering the specific case in which the first columns of $\mathbf{X}$ corresponds to $\mathbf{A}^*$, we would like to have:

$$\mathbf{X} = \begin{pmatrix} \mathbf{a}_1^* \, \mathbf{a}_2^* \, .. \, \mathbf{a}_n^* & \tilde{\mathbf{X}} \end{pmatrix} = \begin{pmatrix} \mathbf{a}_1^* \, \mathbf{a}_2^* \, .. \, \mathbf{a}_n^* & \tilde{\mathbf{X}} \end{pmatrix} \begin{pmatrix} \mathbf{I}_d & \tilde{\mathbf{S}} \\ \hline (\mathbf{0}) & (\mathbf{0}) \end{pmatrix}$$

Should be 0 since we want to explain the columns of $\mathbf{X}$ with nonnegative combinations of the columns of $\mathbf{A}^*$ *only* (and not using the mixture points to explain the other mixture points)

=> $\mathbf{Y} \in \mathcal{Y}$ should have as few nonzeros rows as possible (a zero row of $\mathbf{Y}$ means that the corresponding column of $\mathbf{X}$ is not used in the decomposition)

Self-dictionary algorithms: noiseless case

- Let us introduce the set $\mathcal{Y} = \{\mathbf{Y} \in \mathbb{R}_+^{t \times t} \,|\, \mathbf{X} = \mathbf{XY}\}$

  => $\mathbf{Y} \in \mathcal{Y}$ should have as few nonzeros rows as possible (a zero row of $\mathbf{Y}$ means that the corresponding column of $\mathbf{X}$ is not used in the decomposition)

- First self-dictionary algorithm (with $\|.\|_{row,0}$ the number of non-zero rows):

$$\mathbf{Y}^* = \arg\min_{\mathbf{Y} \in \mathcal{Y}} \|\mathbf{Y}\|_{row,0}$$

- This requires to solve a difficult non-convex **combinatorial** problem

- Provided that $\mathbf{X}$ is **noiseless** near-separable (and follows the other hypotheses we evoked) mixtures, this algorithm can be shown to **provide a near-separable factorization**.

$$\mathbf{X} = \sum_{\{j \in [1,n] \;|\; \|\mathbf{y}^j\|_0 \neq 0\}} \mathbf{x}_j \, \mathbf{y}^j$$

- N.B. : we don't need to specify n: $\|\mathbf{Y}\|_{row,0} = n$ is naturally fulfilled

Self-dictionary algorithms: noisy case, no duplicated columns

- The constraint $\mathcal{Y} = \{\mathbf{Y} \in \mathbb{R}_+^{t \times t} \,|\, \mathbf{X} = \mathbf{XY}\}$ might not be realistic in presence of **noise**
- $\|\mathbf{Y}\|_{row,0}$ encompasses a **combinatorial** problem

  => **relax** it

- Solve, for a given $\lambda > 0$,

$$\arg \min_{\mathbf{Y} \in \mathbb{R}^{t \times t}} \|\mathbf{Y}\|_{1,\infty} + \lambda \|\mathbf{X} - \mathbf{XY}\|_F^2$$

- It has been proved that this principle enables to recover the ground truth:

$$\mathbf{A}^* \simeq \mathbf{X}_{\mathcal{K}_y} \text{ where } \mathcal{K}_y = \{k \in [1,t] \mid \|\mathbf{y}^k\|_\infty \neq 0\}$$

N.B.: **this result requires the absence of (near-)duplicated columns** of $\mathbf{A}^*$ in $\mathbf{X}$, which is still a big flaw.

Self-dictionary algorithms: noisy case, duplicated columns

Hottopixx

$$\mathbf{Y}^* = \arg\min_{\mathbf{Y}\in\mathbb{R}^{t\times t}} \mathbf{v}^T\mathbf{diag}(\mathbf{Y})$$

$$\text{such that} \quad \max_j \|\mathbf{x}_j - \mathbf{X}\mathbf{y}_j\|_1 \leq \delta,$$

$$tr(\mathbf{Y}) = r,$$

$$0 \leq \mathbf{Y}(i,j) \leq \mathbf{Y}(i,i) \leq 1 \text{ for all } i,j$$

- $\mathbf{v}$ is a random vector, enabling to break the symmetry of the near-separable NMF problem hence allowing to deal with duplicated columns

- Can be solved as a **linear program**, at the price that n **must be specified**.

- At the end, recovering $\mathbf{A}^*$ is done by taking the n largest diagonal entries of $\mathbf{Y}^*$.

- In the **noiseless case**, can cope with **duplicated columns** of $\mathbf{A}^*$ within $\mathbf{X}$.

- Can also cope with noise, by adding a post-processing of the diagonal entries of $\mathbf{Y}^*$: the diagonal entires of $\mathbf{Y}^*$ are clustered together depending on the distances between the corresponding columns of $\mathbf{X}$ (using a k-means with weights).

Self-dictionary algorithms: drawback

- Might **rely heavily on near-separability**: if this is violated, greedy algorithms might provide better results.

- They still require parameter tuning: λ, δ, r, depending on the algorithms

- Require to solve a convex optimization problem in $\mathcal{O}(t^2)$ variables => **computationally expansive**

  => they are often used after a pre-processing in which greedy algorithms select a few sample candidates (which also enables to limit the problem of (near-)duplicated columns)

Near-separable NMF

Pros

- Easily geometrically interpretable
- Robustness guarantees **in the presence of noise**
- Can be used as a pre-processing of other NMF algorithms

Cons

- Near separability can be a very strong assumption in HS
- It is **not** a necessary condition
- In practice, near-separable algorithms might be too simplistic

Outline

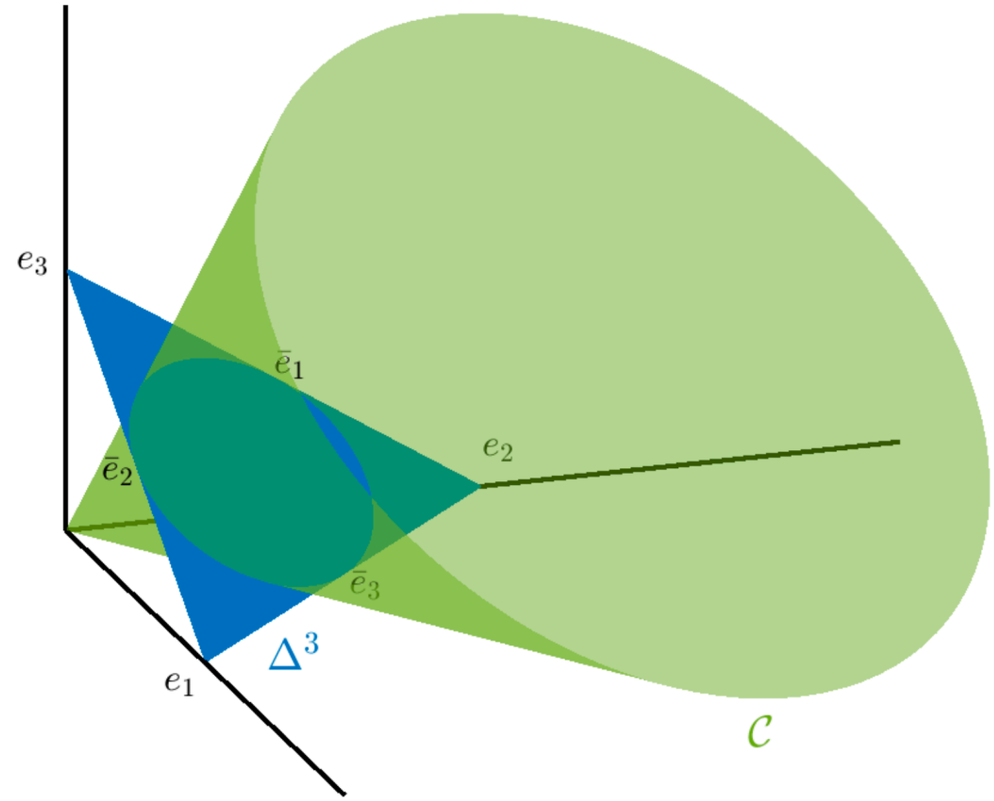
- **Plain NMF**
- **Near-separable NMF**
- **Minimum volume NMF**

Relaxation of near-separability: Sufficiently Scattered Condition

- As said, near-separability is not a necessary condition

  => the **sufficiently scattered condition (SSC)** is more general and also leads to identifiability results in the absence of noise

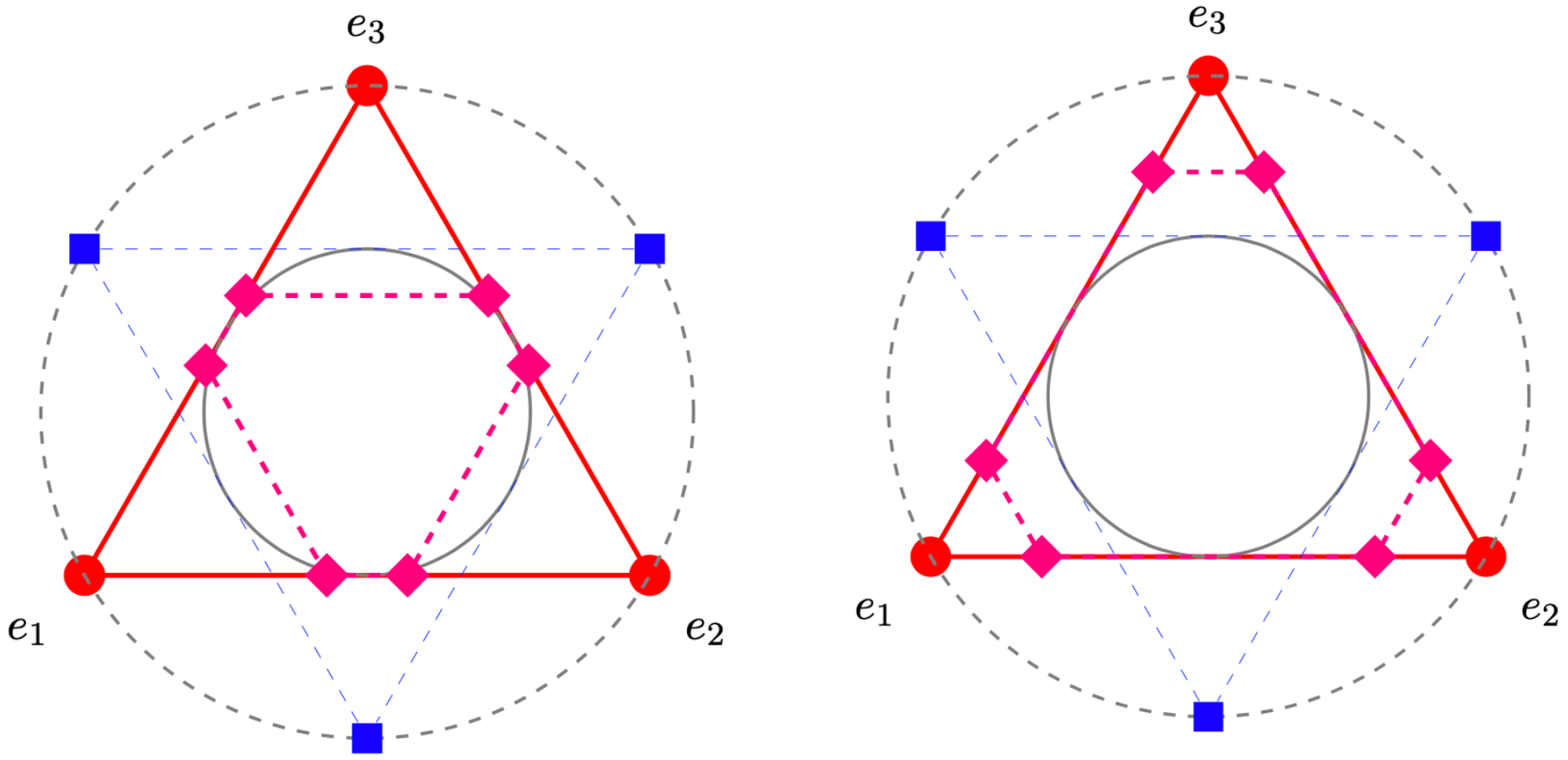
- Intuitively, the SSC condition requires the column of $\mathbf{S}^*$ to span a « sufficiently » big area in the nonnegative orthant



- It further means that $\mathbf{S}^*$ has some degree of sparsity

Sufficiently Scattered Condition: mathematical formulation

- **Sufficiently Scattered Conditions**: $\mathbf{S}^*$ is said to be sufficiently scattered if:
  - $\mathscr{C} = \{\mathbf{x} \in \mathbb{R}^n_+ \,|\, \mathbf{e}^T\mathbf{x} \geq \sqrt{n-1}\|\mathbf{x}\|_2\} \subseteq \texttt{cone}(\mathbf{S}^*)$ with $\mathbf{e}^T = (1,1,...,1)$ (the ice-cream cone is contained within the cone generated by $\mathbf{S}^*$)
  - There does not exist any orthogonal matrix $\mathbf{Q}$ such that $\texttt{cone}(\mathbf{S}^*) \subseteq \texttt{cone}(\mathbf{Q})$, except permutation matrices. Remember that $\mathbf{Q}$ is a orthogonal if $\mathbf{Q}^T\mathbf{Q} = \mathbf{Q}\mathbf{Q}^T = \mathbf{I}$



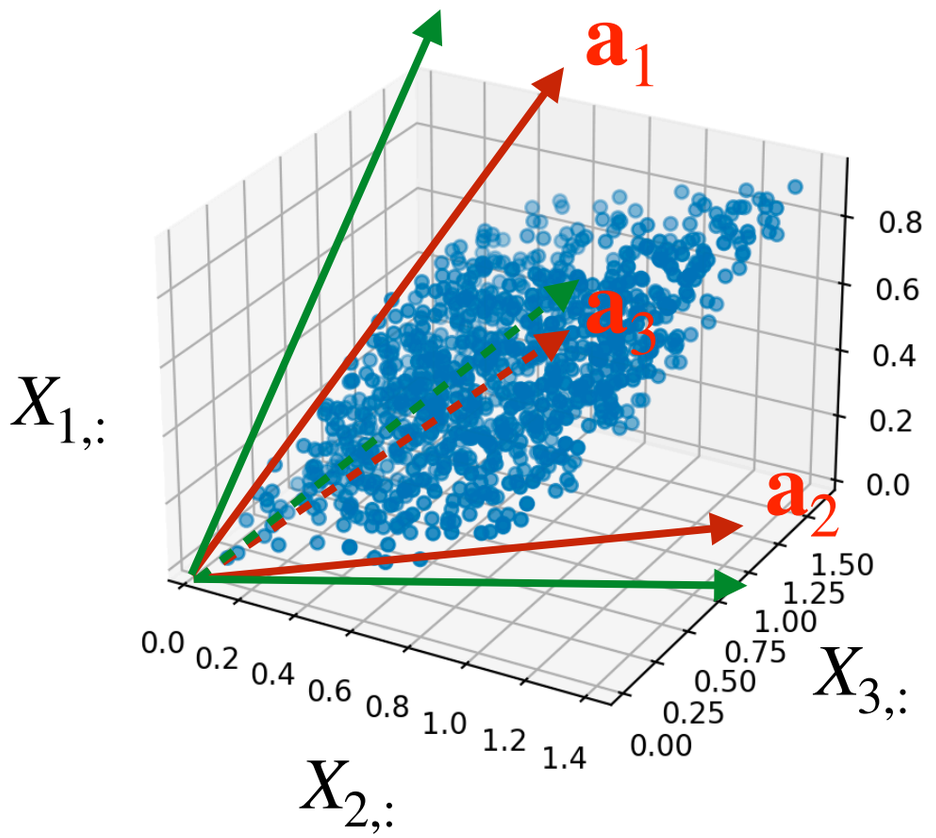
Rem: if $\mathbf{S}^*$ is near-separable, then it is sufficiently scattered

Minimum volume NMF

- If we consider the noiseless mixture $\mathbf{X} = \mathbf{A}^*\mathbf{S}^*$, then it makes sense to look among all the possible $\mathbf{A}$ such that $\texttt{conv}(\mathbf{X}) \subseteq \texttt{conv}(\mathbf{A})$ for the ones such that the volume of $\texttt{conv}(\mathbf{A})$ is minimal.

$$\text{where } \texttt{conv}(\mathbf{A}) = \{\mathbf{x} \mid \mathbf{x} = \mathbf{A}\mathbf{y}, \mathbf{y} \in \mathbb{R}^n, \mathbf{y} \geq 0, \mathbf{e}^T\mathbf{y} = 1\}$$

- Intuitively, minimum-volume NMF looks for a matrix $\hat{\mathbf{A}}$ as close as possible to the dataset $\mathbf{X}$ columns:



- **If $\mathbf{S}^*$ is sufficiently scattered, then such a $\hat{\mathbf{A}}$ corresponds to $\mathbf{A}^*$!**

Minimum volume NMF: a few precisions

How to measure the volume of `conv(A)`?

- We can use the determinant:

$$\frac{1}{n!}\sqrt{\det(\mathbf{A}^T\mathbf{A})}$$

- Using the logarithm of $\det(\mathbf{A}^T\mathbf{A})$ yields better practical results, (less sensitive to very small and very large singular values of $\mathbf{A}$).
- Adding a small positive value δ prevents the logarithm to go to $-\infty$ in the rank deficient case

$$\log\det(\mathbf{A}^T\mathbf{A} + \delta\mathbf{I}_n)$$

Different models for min-vol NMF:

- There exists different min-vol NMF algorithm, with subtleties, in particular in the normalization used:

1. $\mathbf{S}^T\mathbf{e} = \mathbf{e}$
2. $\mathbf{S}\mathbf{e} = \mathbf{e}$
3. $\mathbf{A}^T\mathbf{e} = \mathbf{e}$

with $\mathbf{e}^T = (1,1,1,...,1)$

Minimum volume NMF: deriving practical algorithms

Cost function

- The problem is usually solved as the minimization of a cost function
- In the noisy case, we must balance the data fidelity term and the minimum volume one:

$$\hat{\mathbf{A}}, \hat{\mathbf{S}} = \arg\min_{\mathbf{A},\mathbf{S}\geq 0} \frac{1}{2}\|\mathbf{X} - \mathbf{AS}\|_F + \lambda \log\det(\mathbf{A}^T\mathbf{A} + \delta\mathbf{I})$$

under the constraints on $\mathbf{A}$ and $\mathbf{S}$ evoked above, with $\delta > 0$ a small parameter preventing logdet $\rightarrow -\infty$ when rank(A) < n

Minimization algorithm

- The log function is concave…
- … a way to apply to proximal algorithms we have seen before is to find a convex majorizer of the cost function (Maximization Minimization algorithms)
- Then, we can use proximal algorithms to minimize the majorizer.
- Note however that, depending on the chosen normalization of the factors, the proximal operators might not be explicit.

  => this is slightly tricky, so most of the code will be already implemented for the TP.

Conclusion on Minimum volume NMF

Pros

- It is more general than near-separability
- The sufficiently scattered condition is practically rather mild
- It leads to better results in practice

Cons

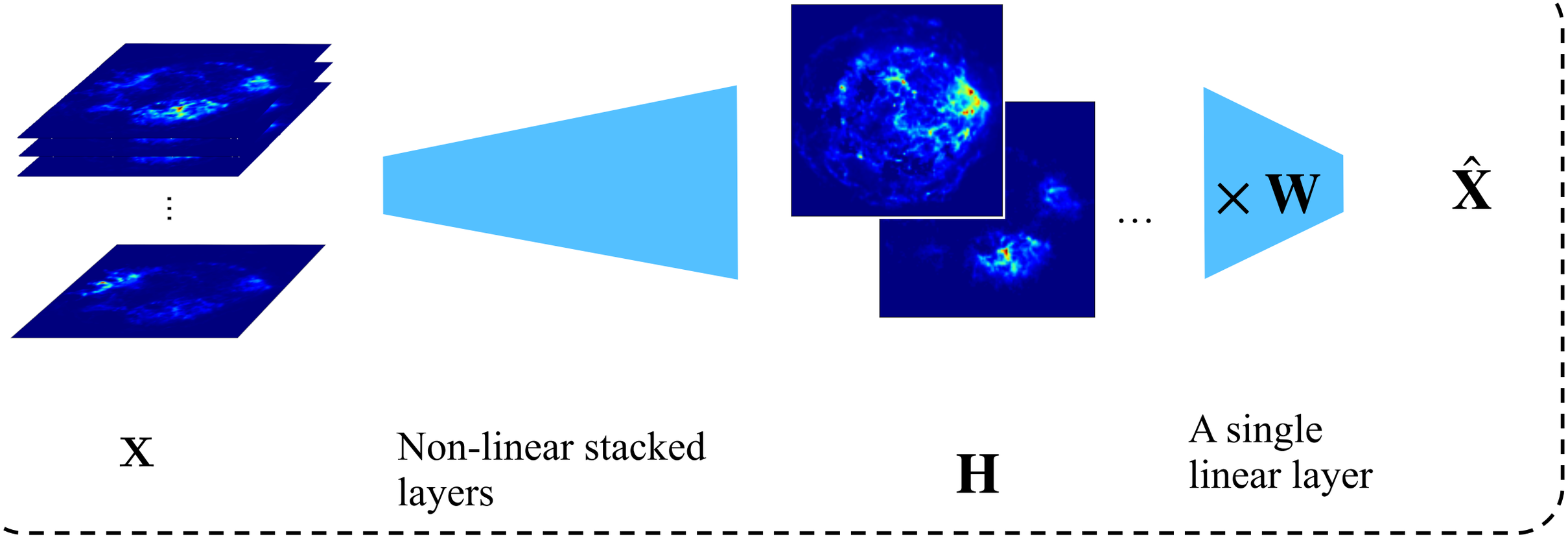
- Solving exactly min-vol NMF is NP-hard (alleviating the near-separability thus comes at a price)
- An open question is thus when is it possible to solve it efficiently? And how to do it in practice?
- The behaviour of min-vol NMF is not well understood in the presence of noise

Outline

- **Plain NMF**
    - Problem statement
    - Optimization framework
        - PALM
        - Multiplicative updates
- **Near-separable NMF**
    - Definition
    - Algorithms: brute force and greedy
    - Recovery guarantees
- **Minimum volume NMF**
- **Extensions to Deep Learning**

Extension to deep learning: insight

- Most of the current extensions of NMF (at least for imaging) are based on auto-encoders for doing hyperspectral unmixing



- The most basic training function is $\arg \min_{weights} \frac{1}{2}\|\mathbf{X} - \hat{\mathbf{X}}\|_F^2$.

- The use of a single linear layer makes the auto-encoder decoder architecture quite interpretable: the latent space is expected to correspond to $\mathbf{S}^*$ ($\mathbf{H} \simeq \mathbf{S}^*$) since the output is $\hat{\mathbf{X}} = \mathbf{WH}$ and $\mathbf{X} \simeq \hat{\mathbf{X}}$. The $\mathbf{A}^*$ matrix is approximated by the weights $\mathbf{W}$.

Need for regularization

- Of course, $\hat{\mathbf{X}} = \mathbf{X} \Leftrightarrow \mathbf{WH} = \mathbf{A}^*\mathbf{S}^*$ does not imply that $\mathbf{W} = \mathbf{A}^*$ and $\mathbf{H} = \mathbf{S}^*$.

  => Need for regularization!

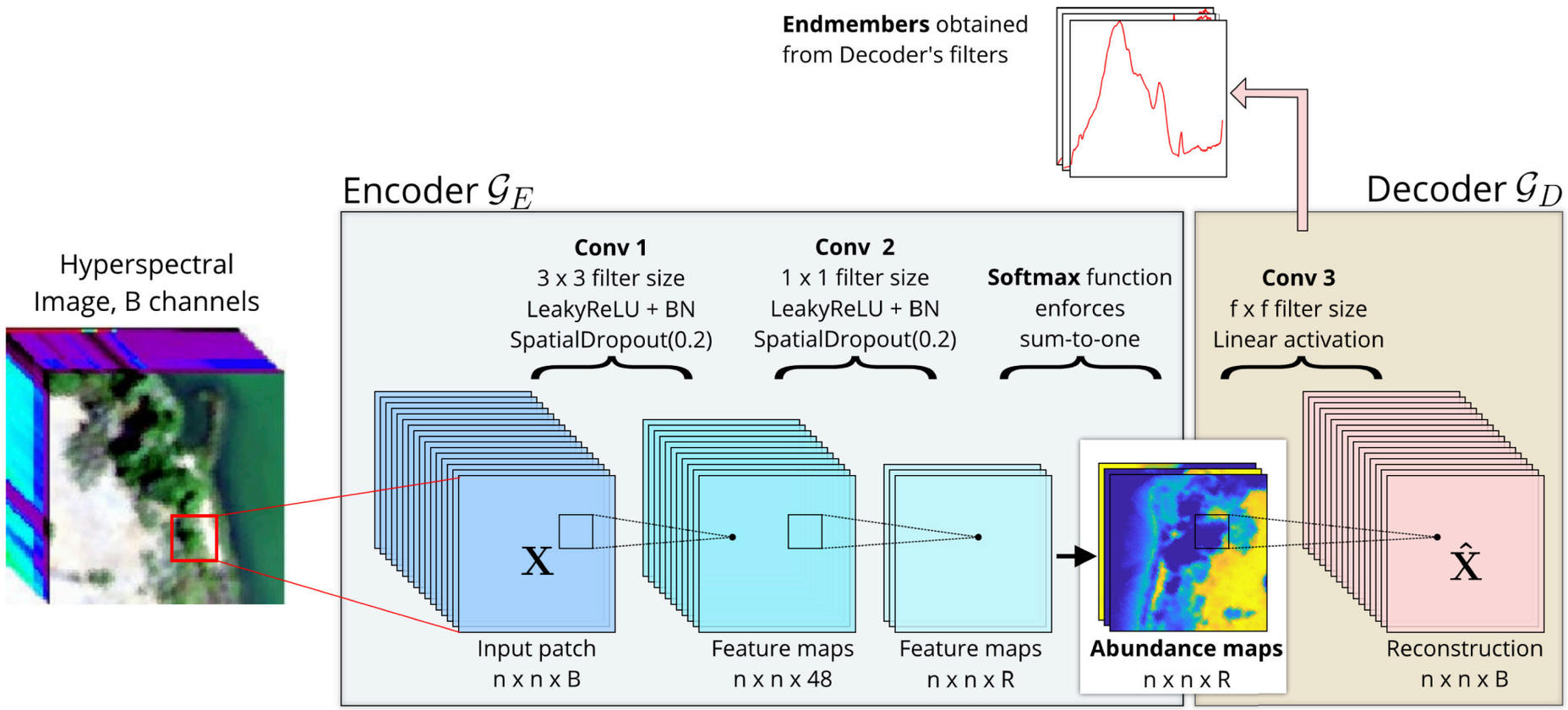
- Among the constraints which are often implemented:
  - Nonnegativity of the $\mathbf{H}$ coefficients
  - Sum-to-one of the coefficients in each pixel of $\mathbf{H}$ (due to the fact that we are looking for concentrations of elements => this is not always true in astrophysics, why?)

=> both constraints are often implemented using a softmax nonlinearity at the end of the encoder

$$\text{if } \mathbf{h} \in \mathbb{R}^n,\ S(\mathbf{h})_i = \frac{e^{\mathbf{h}_i}}{\sum_{k=1}^{n} e^{\mathbf{h}_k}}$$

Need for regularization

- Even using the NN and sum-to-one constraints, the problem is still ill-posed, requiring several optimization trick to avoid bad solutions.
- Among many neural networks, [Palsson21] has obtain a large success in hyperspectral unmixing

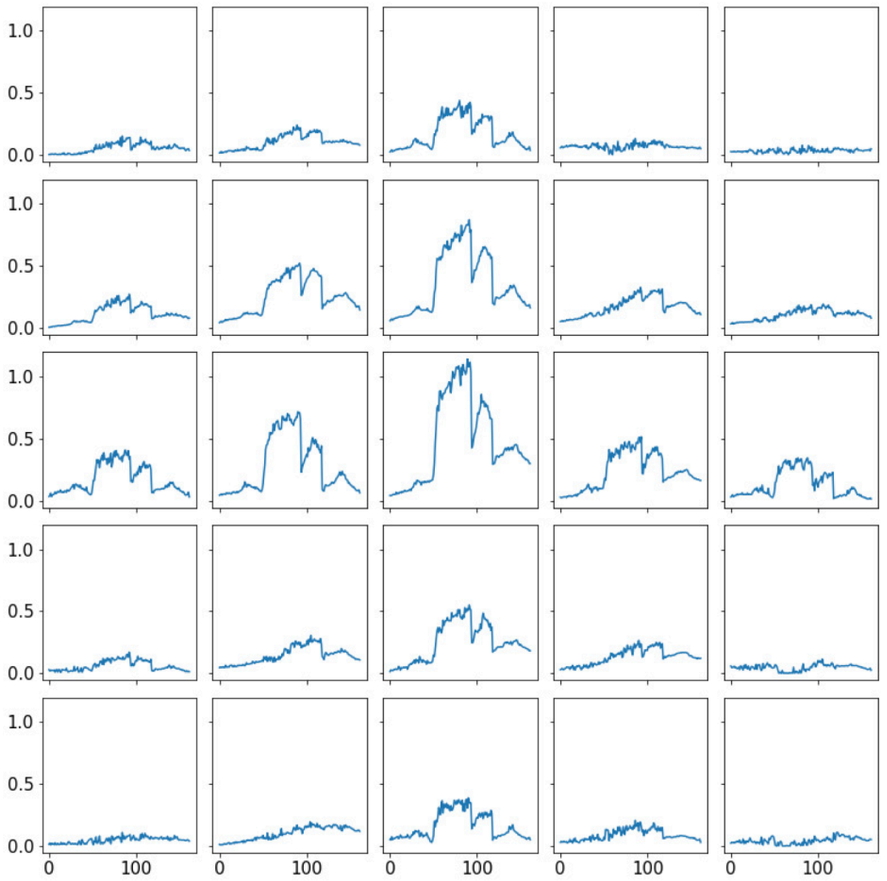


- They use LeakyRelu, batch-normalization, spatial dropout, convolutive encoder… But the main originality lie in using a convolutive *decoder*

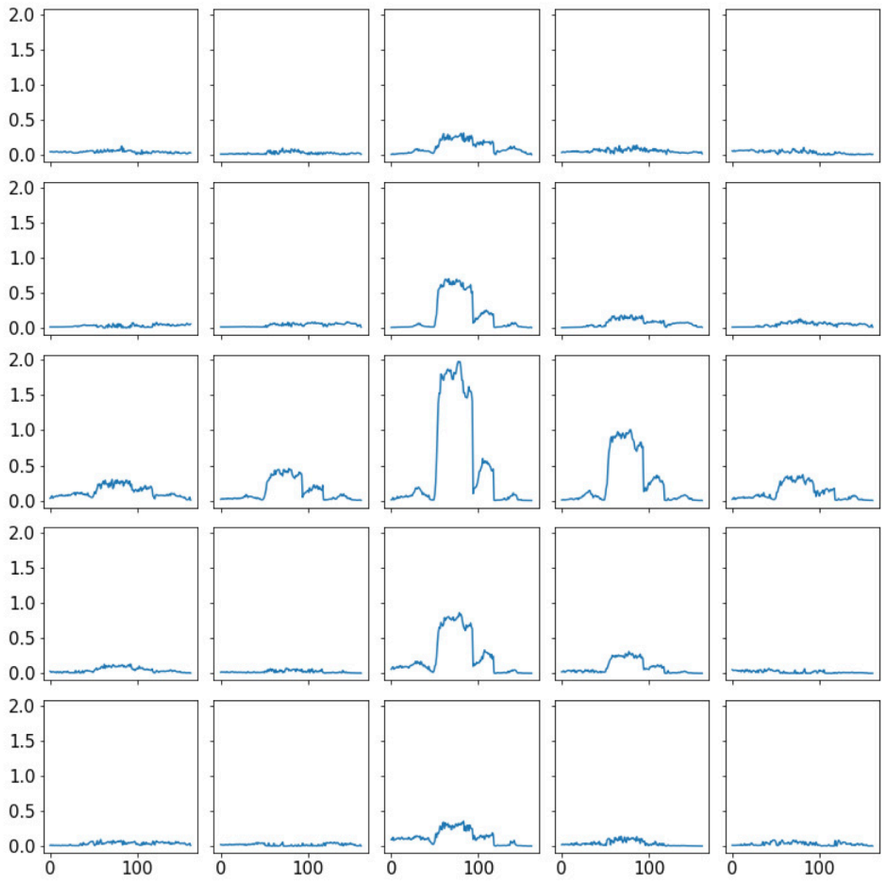
Need for regularization

- Using convolutive decoders enable to take into account spatial correlations between the spectral

- Exemple for two materials present in the scene:



(a) Grass



(b) Tree

NMF: conclusion

- NMF is a relatively recent paradigm
- It corresponds to **naturally fulfilled conditions**
- In contrast to ICA, it **can cope with noise**
- In contrast to sparse BSS, more theoretical works have been done on the **identifiability**
- Which model to use?

  => It is actually very problem-dependent

  => it is also possible to hybrid methods (e.g.: sparse NMF)
- There are extensions to neural networks

- There is also quite a lot of **works about different type of noises**
- Recently, deep NMF models have been proposed, to try to mimic neural networks
- There is also several other specific NMF type, for instance orthogonal NMF