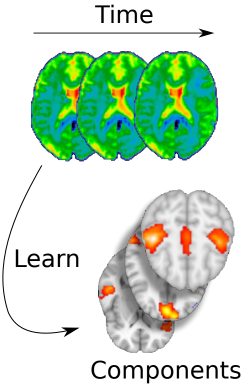
Massive Matrix Factorization for Resting-State fMRI Decomposition

Arthur Mensch, Julien Mairal,
Bertrand Thirion, Gaël Varoquaux

Inria Parietal/Thoth, CEA, Neurospin, Université Paris-Saclay

July 24, 2017

Introduction



Resting-state functional MRI data:

- **Undirect measure of brain activity**
- Subject at rest for 15 min
- 200 000 voxels (8 mm^3), every 2 s
- $\sim$ 1 000 brain maps per records

Unsupervised pattern analysis:

- Learn characteristic **spatial components**

= Functional networks

- Validated by prediction on *unseen* subjects

How to extract functional networks ?

Popular methods:

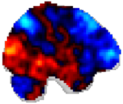
- Principal Components Anaysis [5]
- Independant Component Analysis [2]

Caveats

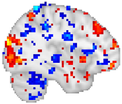
- Noise/signal components
- Small dataset size (ortho. constraints)
- Thresholding to obtain sparse networks

Alternative: sparse matrix factorization [6]

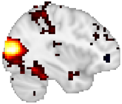
- Natural sparsity — within the objective
- Can be adapted for large datasets



PCA



ICA (+ thresholding)



Matrix factorization

Challenge and contribution

Community effort to provide larger datasets:

- From 15 subjects in 1997 to 100000 in 2020
- Costly studies on large cohort: HCP, UKBiobank
- From **2 GB** to **5 TB** datasets

Decomposition algorithm should have a reasonable cost:

- Often followed by ROI extraction, connectivity analysis
- We pick sparse matrix factorization

New algorithms for massive matrix factorization.

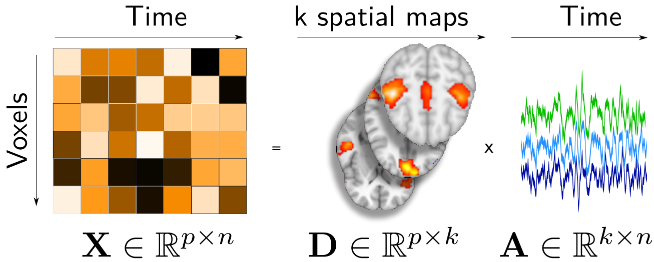
A faster cousin of stochastic gradient descent.

Matrix Factorization for Functional MRI

Resting-state data analysis:

- **Input:** concatenated time-series for many subjects
- $\mathbf{X} \in \mathbb{R}^{p \times n}$, $n = 5 \cdot 10^6$, $p = 2 \cdot 10^5$
- **Goal:** Extract representative sparse components **D**

$=$ Functional networks

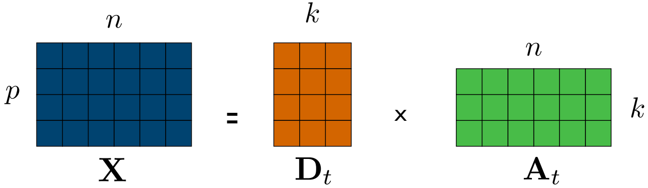


Formalism

Non-convex matrix factorization:

$$\min_{\mathbf{D}\in\mathcal{C},\mathbf{A}\in\mathbf{R}^{k\times n}} \|\mathbf{X} - \mathbf{D}\mathbf{A}\|_F^2 + \lambda\Omega(\mathbf{A})$$

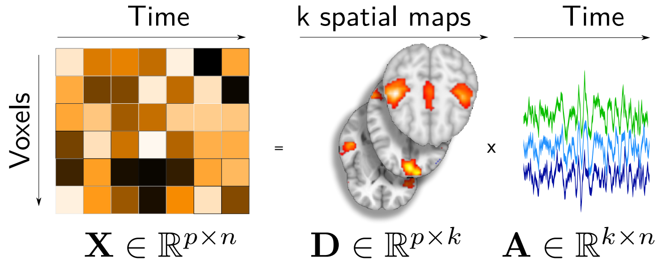
- Constraints on the dictionary $\mathbf{D}$: each column $\mathbf{d}^{(j)}$ in $\mathcal{B}_2$ or $\mathcal{B}_1$
- Penalty on the code $\mathbf{A}$: ℓ_1, ℓ_2 (+ non-negativity)



Algorithm design

Naive resolution:

- Alternated minimization: use full **X** at each iteration
- **Slow:** single iteration cost in $\mathcal{O}(np)$



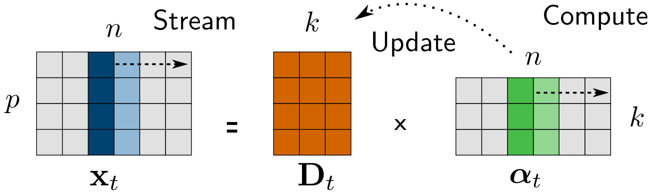
X is large (**5TB**) in both number of samples n and sample dimension p

New stochastic algorithms that scale in **both** directions

Online matrix factorization [1]

Scaling with n:

- Stream $(\mathbf{x}_t)$ and update $(\mathbf{D}_t)$ at each t
- Single iteration cost in $\mathcal{O}(p)$
- Convergence in a few epochs $\rightarrow$ large speed-up



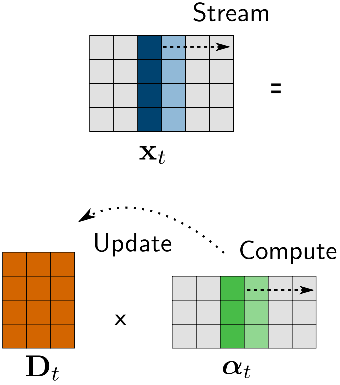
Previous use case:

- Large n, regular p, *e.g.*, image patches — sparse **A**

$$p = 256 \qquad n \approx 10^6 \qquad \mathbf{1GB}$$

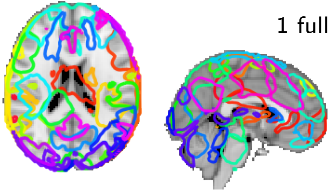
Scaling-up for massive matrices

Out-of-the-box online algorithm ?



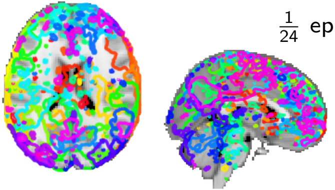
Limited time budget ?

Need to accomodate large p



1 full epoch

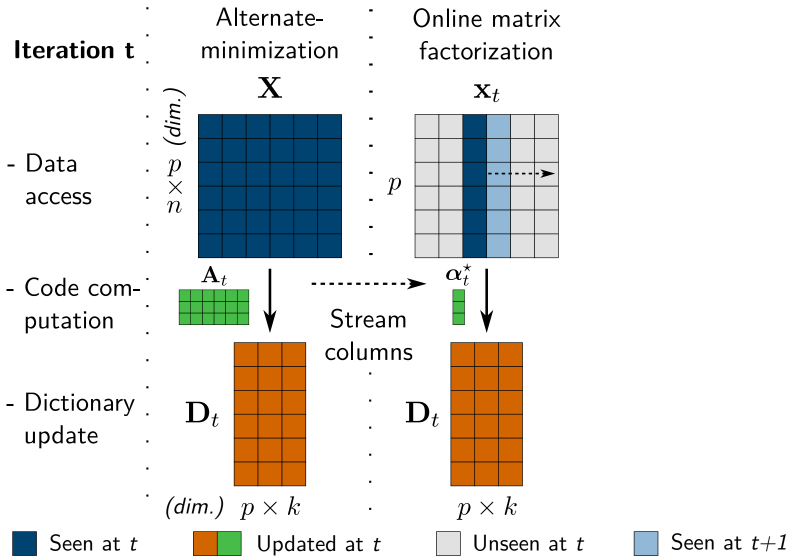
48 h run time



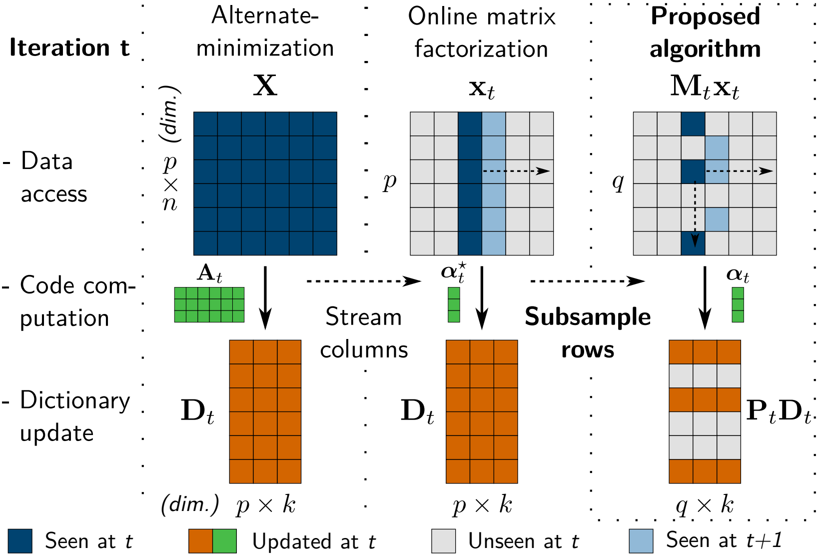
$\frac{1}{24}$ epoch

2 h run time

Scaling-up in both directions



Scaling-up in both directions



Algorithm design

Online dictionary learning [1]

1. **Compute code** – $\mathcal{O}(p)$

$$\boldsymbol{\alpha}_t = \underset{\boldsymbol{\alpha} \in \mathbb{R}^k}{\operatorname{argmin}} \|\mathbf{x}_t - \mathbf{D}_{t-1}\boldsymbol{\alpha}\|_2^2 + \lambda\Omega(\boldsymbol{\alpha}_t)$$

2. **Update surrogate** – $\mathcal{O}(p)$

$$\bar{g}_t(\mathbf{D}) = \frac{1}{t}\sum_{s=1}^{t} \|\mathbf{x}_s - \mathbf{D}\alpha_s\|_2^2 = \mathrm{Tr}\left(\mathbf{D}^\top \mathbf{D}\bar{\mathbf{C}}_t - \mathbf{D}^\top \bar{\mathbf{B}}_t\right)$$

3. **Minimize surrogate** – $\mathcal{O}(p)$

$$\mathbf{D}_t = \underset{\mathbf{D} \in \mathcal{C}}{\operatorname{argmin}}\, \bar{g}_t(\mathbf{D}) = \underset{\mathbf{D} \in \mathcal{C}}{\operatorname{argmin}}\, \mathrm{Tr}\left(\mathbf{D}^\top \mathbf{D}\mathbf{C}_t - \mathbf{D}^\top \mathbf{B}_t\right)$$

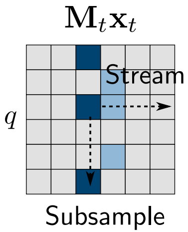
Access to $\mathbf{x}_t$ $\rightarrow$ Algorithm in $\mathcal{O}(p)$ (complexity dependency in p)

Introducing subsampling

How to reduce single iteration cost $\mathcal{O}(p)$?

- Sample masking matrix $\mathbf{M}_t$
- Diagonal matrix with rescaled Bernouilli coefficients, $\mathbb{E}[\mathrm{rank}\, \mathbf{M}_t] = q$
- $\mathbf{x}_t \to \mathbf{M}_t\mathbf{x}_t$, $\mathbb{E}[\mathbf{M}_t\mathbf{x}_t] = \mathbf{x}_t$
- Use only $\mathbf{M}_t\mathbf{x}_t$ in algorithm computations

$\Rightarrow$ **Complexity in $\mathcal{O}(q)$**



Algorithmic contribution

Adapt the 3 parts of the algorith to obtain $\mathcal{O}(q)$ complexity

1. **Code computation**
2. **Surrogate update**
3. **Surrogate minimization**

Subsampled Online matrix Factorization (SOMF)

Theoretical guarantees

Objective function for the dictionary:

$$\bar{f} \triangleq \min_{\mathbf{A}} \|\mathbf{X} - \mathbf{D}\mathbf{A}\|_F^2 + \lambda\Omega(\mathbf{A}).$$

Asymptotic convergence towards a critical point.

Proposition: $\bar{f}(\mathbf{D}_t)$ converges with probability one and every limit point $\mathbf{D}_\infty$ of $(\mathbf{D}_t)_t$ is a stationary point of $\bar{f}$: for all $\mathbf{D} \in \mathcal{C}$

$$\nabla \bar{f}(\mathbf{D}_\infty, \mathbf{D} - \mathbf{D}_\infty) \geq 0$$

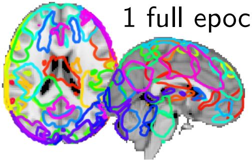
Low hypotheses: For all $t > 0$, $\bar{\mathbf{C}}_t, \mathbf{D}_t^\top \mathbf{D}_t \geq \rho \mathbf{I}$

- No assumption on $r = \frac{p}{q}$ (!)
- No rates (as the original **OMF**)

$\Rightarrow$ Empirical validation of speed-ups

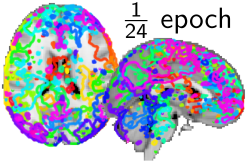
Results

Online dictionary learning



1 full epoch

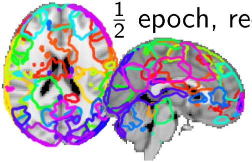
48 h run time



$\frac{1}{24}$ epoch

2 h run time

Proposed method

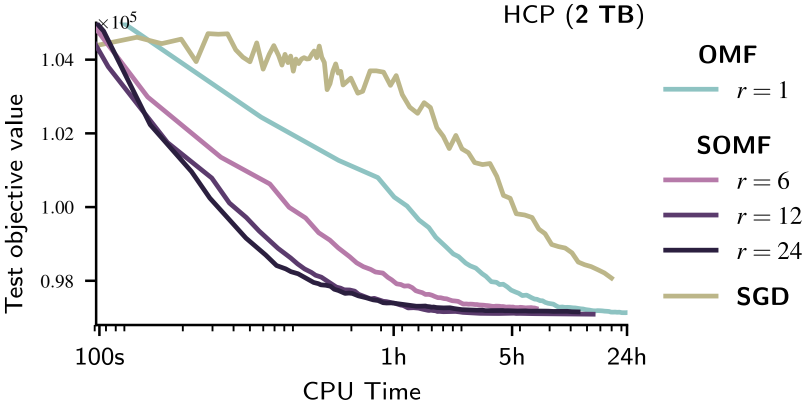


$\frac{1}{2}$ epoch, reduction $r = 12$

2 h run time

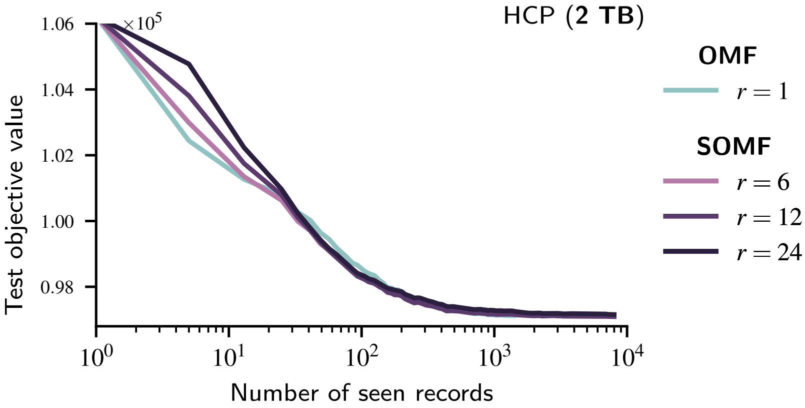
Qualitatively, usable maps are obtained **12× faster**

Quantitative results



Speed-up close to reduction factor $\frac{p}{q}$

Quantitative results



Stochastic subsampling introduce little noise

Information is acquired faster

Conclusion

New efficient algorithm with many potential use-cases:

- Hyperspectral images [4]
- Collaborative filtering [3]
- Genomics

Perspectives:

- Efficient heuristics and adaptative subsampling ratio
- To integrate in our Python package (`github.com/arthurmensch/modl`)
- Is this kind of approach transposable to SGD setting ?

We may now easily use these dictionaries in a supervised setting.

Bibliography I

[1] J. Mairal, F. Bach, J. Ponce, and G. Sapiro.
Online learning for matrix factorization and sparse coding.
The Journal of Machine Learning Research, 11:19–60, 2010.

[2] M. J. McKeown, S. Makeig, G. G. Brown, T. P. Jung, S. S. Kindermann, A. J. Bell, and T. J. Sejnowski.
Analysis of fMRI Data by Blind Separation into Independent Spatial Components.
Human Brain Mapping, 6(3):160–188, 1998.

[3] A. Mensch, J. Mairal, B. Thirion, and G. Varoquaux.
Dictionary learning for massive matrix factorization.
In *33rd International Conference on Machine Learning (ICML)*, June 2016.

[4] A. Mensch, J. Mairal, B. Thirion, and G. Varoquaux.
Stochastic Subsampling for Factorizing Huge Matrices.
arXiv:1701.05363 [cs, math, q-bio, stat], Jan. 2017.

Bibliography II

[5] J. Moeller and S. Strother.
A regional covariance approach to the analysis of functional patterns in positron emission tomographic data.
Journal of Cerebral Blood Flow & Metabolism, 11(1_suppl):A121–A135, 1991.

[6] G. Varoquaux, A. Gramfort, F. Pedregosa, V. Michel, and B. Thirion.
Multi-subject dictionary learning to segment an atlas of brain spontaneous activity.
In *Proceedings of the Information Processing in Medical Imaging Conference*, volume 22, pages 562–573. Springer, 2011.