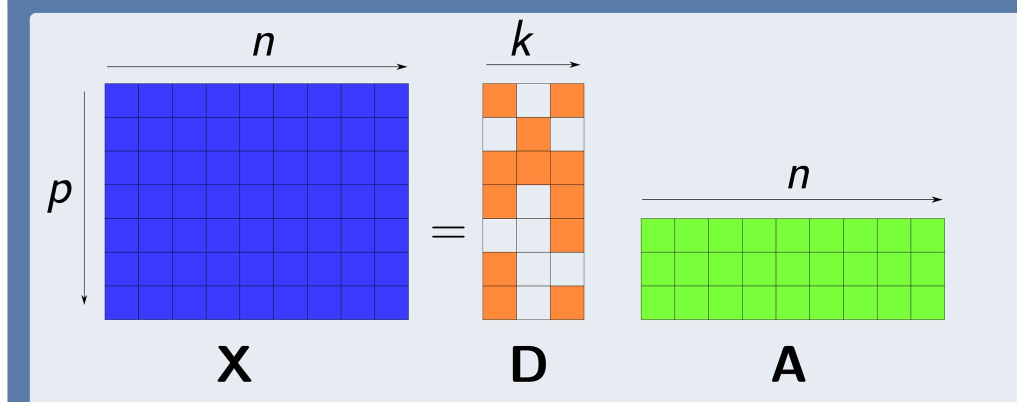
# Dictionary Learning for Massive Matrix Factorization


Arthur Mensch[1] Julien Mairal[2] Gaël Varoquaux[1] Bertrand Thirion[1]

[1]Parietal team, Inria, CEA, Neurospin, Paris-Saclay University. Gif-sur-Yvette, France
[2]Thoth team, Inria. Grenoble, France


## Scaling-up matrix factorization




$$\min_{\mathbf{D}\in\mathcal{C},\mathbf{A}\in\mathbb{R}^{k\times n}} \|\mathbf{X}-\mathbf{D}\mathbf{A}\|_2^2 + \lambda\Omega(\mathbf{A})$$

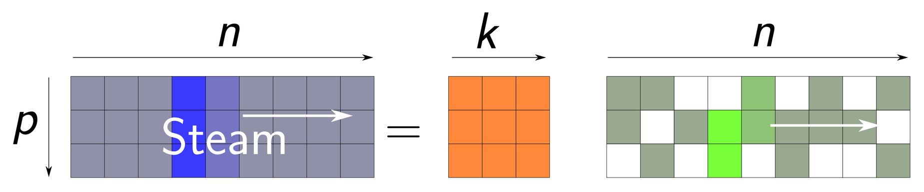
- $\mathbf{X} \approx \mathbf{D}\mathbf{A} \in \mathbb{R}^{p\times k} \times \mathbb{R}^{k\times n}$
- Flexible tool for unsupervised data analysis
- Dataset has lower underlying complexity than appearing size

How to scale MF to datasets large in both directions ? (fMRI, **2TB**)

## Scaling in $n$

Online algorithm for matrix factorization [3]

- Stream $(\mathbf{x}_t)$, update $\mathbf{D}$ at each $t$
- Single iteration in $\mathcal{O}(p)$, a few epochs



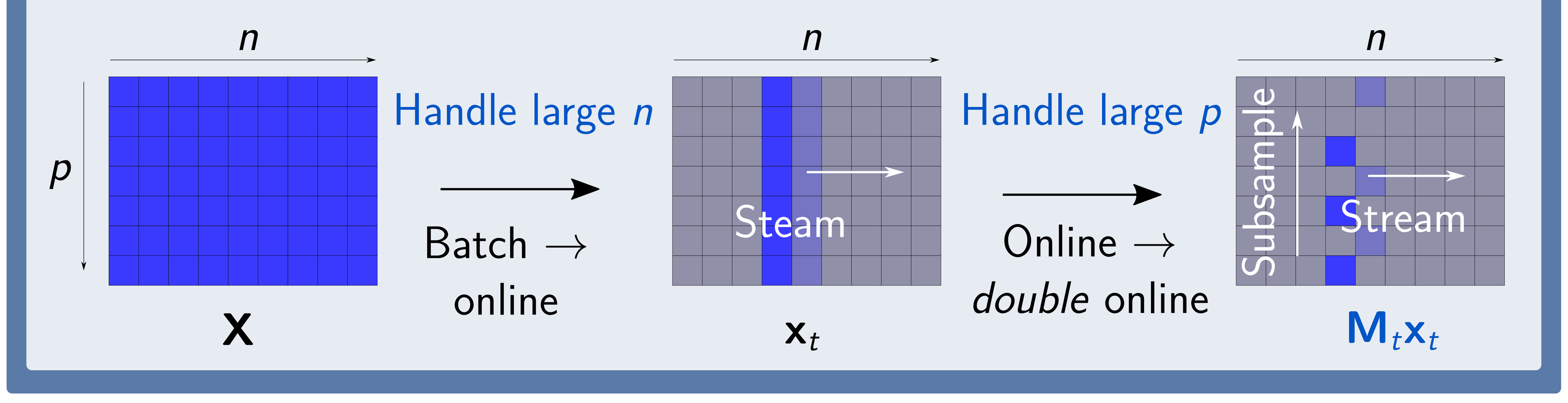

## Scaling in $p$

Use random fractions of features

- Random projection
- Linear algebra (*eg* SVD)
- *Sketching*

← Originally designed for large $n$, small $p$ (vision)

## Scaling in both direction: random subsampling




## Online matrix factorization

$$\min_{\mathbf{D}\in\mathcal{C}} \mathbb{E}_{\mathbf{x}}[\min_{\boldsymbol{\alpha}\in\mathbb{R}^k} \|\mathbf{x}_t - \mathbf{D}\boldsymbol{\alpha}\|_2^2 + \lambda\Omega(\boldsymbol{\alpha})]$$

- Compute code – $\mathcal{O}(p)$

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

- Update surrogate – $\mathcal{O}(p)$

$$g_t = \frac{1}{t}\sum_{i=1}^{t} \|\mathbf{x}_i - \mathbf{D}\boldsymbol{\alpha}_i\|_2^2 = \mathrm{Tr}\,(\mathbf{D}^\top\mathbf{D}\mathbf{A}_t - \mathbf{D}^\top\mathbf{B}_t)$$

$$\mathbf{A}_t = \frac{1}{t}\sum_{i=1}^{t}\boldsymbol{\alpha}_i\boldsymbol{\alpha}_i^\top \quad \mathbf{B}_t = \frac{1}{t}\sum_{i=1}^{t}\mathbf{x}_i\boldsymbol{\alpha}_i^\top$$

- Minimize surrogate – $\mathcal{O}(p)$

$$\mathbf{D}_t = \operatorname*{argmin}_{\mathbf{D}\in\mathcal{C}} g_t(\mathbf{D})$$

## Random subsampling

- Vanilla algorithm: $\mathcal{O}(p)$ iteration

**Reduce it !**

- Random diagonal matrix

$$\mathbf{M}_t \in \mathrm{Diag}([0,1]^p)$$
$$\mathrm{rk}\,\mathbf{M}_t = s_t < p$$

- Restrict algorithm data access:

$$\mathbf{x}_t \to \mathbf{M}_t\mathbf{X}_t$$
$$p \to s_t \sim [\frac{p}{2}, \frac{p}{20}]$$

**Design constraint: complexity $\mathcal{O}(s)$ per single iteration**

## Code computation

- Linear regression with random sampling:

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

- Approximative $\mathcal{O}(s)$ solution of

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

- Validity for large $p$ and incoherent features:

$$\mathbf{D}^\top\mathbf{M}_t\mathbf{D} \approx \frac{s}{p}\mathbf{D}^\top\mathbf{D} \quad \mathbf{D}^\top\mathbf{M}_t\mathbf{x}_t \approx \frac{s}{p}\mathbf{D}^\top\mathbf{x}_t$$

## Surrogate aggregation

- Approximate $\mathbf{A}_t$ and $\mathbf{B}_t$ from $(\mathbf{M}_t\mathbf{x}_t)_t$
- $\mathbf{A}_t$ computed from approximate code
- *Partial* update of $\mathbf{B}_t$ in $\mathcal{O}(s)$

$$\mathbf{B}_t = \frac{1}{\sum_{i=1}^t \mathbf{M}_i}\sum_{i=1}^{t}\mathbf{M}_i\mathbf{x}_i\boldsymbol{\alpha}_i^\top$$
$$= \mathbf{B}_{t-1} + \frac{1}{\sum_{i=1}^t \mathbf{M}_i}(\mathbf{M}_t\mathbf{x}_t\boldsymbol{\alpha}_t^\top - \mathbf{M}_t\mathbf{B}_{t-1})$$

- $\mathbf{B}_t$ behaves like $\mathbb{E}_{\mathbf{x}}[\mathbf{x}\boldsymbol{\alpha}]$ for large $t$

## Software – Ongoing work

- Documented Python package `http://github.com/arthurmensch/modl`
- Heuristic at contribution time – no convergence guarantees
- A follow-up algorithm has convergence guarantees in *finite* sample setting

→ Extending stochastic majorization-minimization [2]

## Surrogate minimization

- Avoid $\mathcal{O}(p)$ block coordinate descent on $\mathbf{D}$
- Leave $\mathbf{D}_t$ rows unchanged for unseen features

$$\min_{\mathbf{D}\in\mathcal{C},\,(\mathbf{I}-\mathbf{M}_t)\mathbf{D}=(\mathbf{I}-\mathbf{M}_t)\mathbf{D}_{t-1}} g_t(\mathbf{D})$$

- $\ell_1, \ell_2$ constraints: $\mathcal{O}(s)$ BCD, projection on

$$\mathcal{C}_j^r = \{\mathbf{d}\in\mathbb{R}^k / \|\mathbf{M}_t\mathbf{d}\|_i \leq \|\mathbf{M}_t(\mathbf{d}_j)_{t-1}\|_i\}$$

**Optional full projection**

- Partial gradient step
- Full lazy projection
- $\mathcal{O}(s)$ for $\ell_2$ constraint
- $\mathcal{O}(s\log p)$ for $\ell_1$ constraint [1]

## Our algorithm

1. Code computation

$$\boldsymbol{\alpha}_t = \operatorname*{argmin}_{\boldsymbol{\alpha}\in\mathbb{R}^k} \|\mathbf{M}_t(\mathbf{x}_t - \mathbf{D}_{t-1}\boldsymbol{\alpha})\|_2^2 + \lambda\frac{\mathrm{rk}\,\mathbf{M}_t}{p}\Omega(\boldsymbol{\alpha}_t)$$

2. Surrogate aggregation

$$\mathbf{A}_t = \mathbf{A}_{t-1} + \frac{1}{t}(\boldsymbol{\alpha}_i\boldsymbol{\alpha}_i^\top - \mathbf{A}_{t-1})$$
$$\mathbf{B}_t = \mathbf{B}_{t-1} + \frac{1}{\sum_{i=1}^t \mathbf{M}_i}(\mathbf{M}_t\mathbf{x}_t\boldsymbol{\alpha}_t^\top - \mathbf{M}_t\mathbf{B}_{t-1})$$

3. Surrogate minimization

$$\mathbf{M}_t\mathbf{D}_j \leftarrow p^{\perp}_{\mathcal{C}_j^r}(\mathbf{M}_t\mathbf{D}_j - \frac{1}{(\mathbf{A}_t)_{j,j}}\mathbf{M}_t(\mathbf{D}(\mathbf{A}_t)_j - (\mathbf{B}_t)_j))$$

## Original online MF

1. Code computation

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

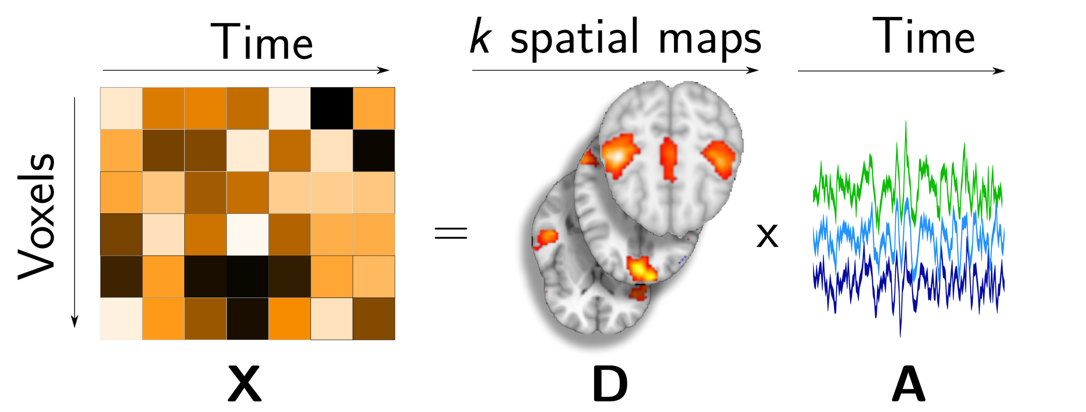
2. Surrogate aggregation

$$\mathbf{A}_t = \mathbf{A}_{t-1} + \frac{1}{t}(\boldsymbol{\alpha}_i\boldsymbol{\alpha}_i^\top - \mathbf{A}_{t-1})$$
$$\mathbf{B}_t = \mathbf{B}_{t-1} + \frac{1}{t}(\mathbf{x}_t\boldsymbol{\alpha}_t^\top - \mathbf{B}_{t-1})$$

3. Surrogate minimization

$$\mathbf{D}_j \leftarrow p^{\perp}_{\mathcal{C}_j}(\mathbf{D}_j - \frac{1}{(\mathbf{A}_t)_{j,j}}(\mathbf{D}(\mathbf{A}_t)_j - (\mathbf{B}_t)_j))$$

## Large fMRI dataset

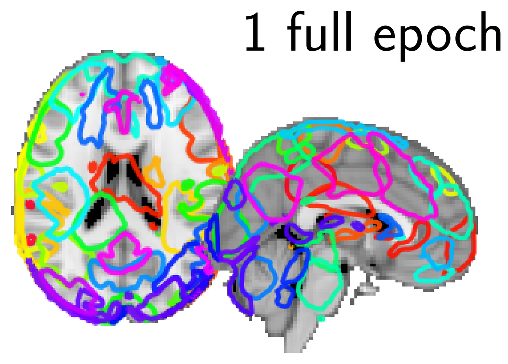



- **2 TB** dataset (HCP)
- Large in both directions
- $p = 2\cdot 10^5$, $n = 2\cdot 10^6$
- Extract 70 sparse spatial maps

… Good basis for unseen subjects

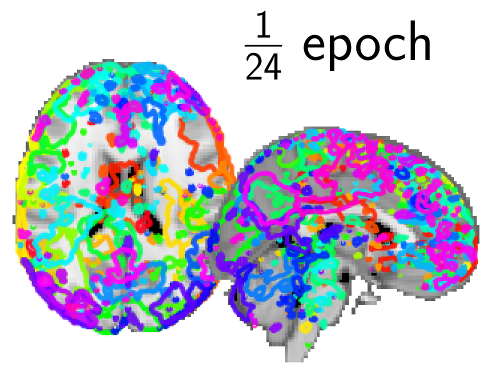
## Baseline online algorithm

1 full epoch



**235 h** run time

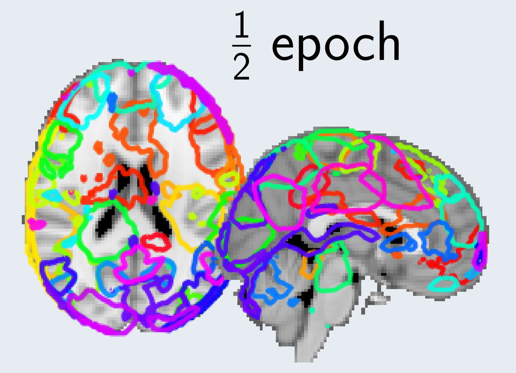
$\frac{1}{24}$ epoch



**10 h** run time

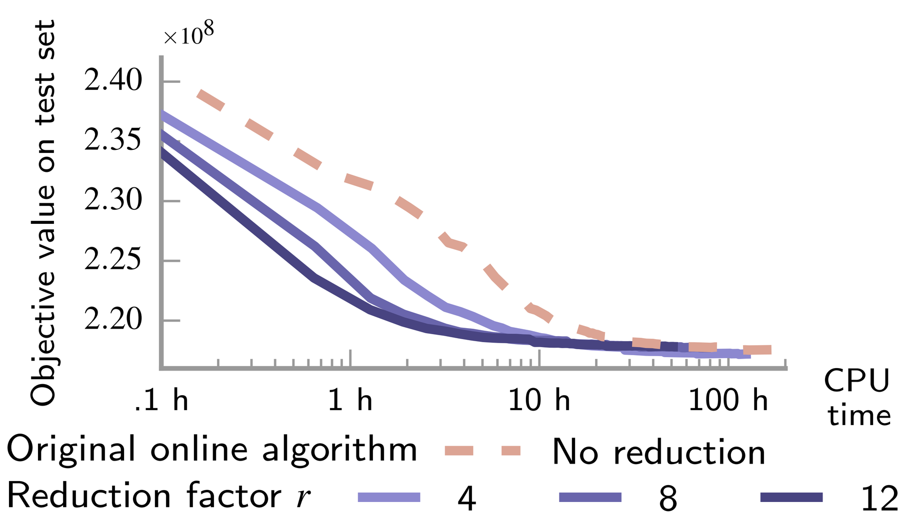
## Reduction $r = 12$

$\frac{1}{2}$ epoch



**10 h** run time

**Well defined sparse maps (noiseless contours) are obtained $10\times$ faster**

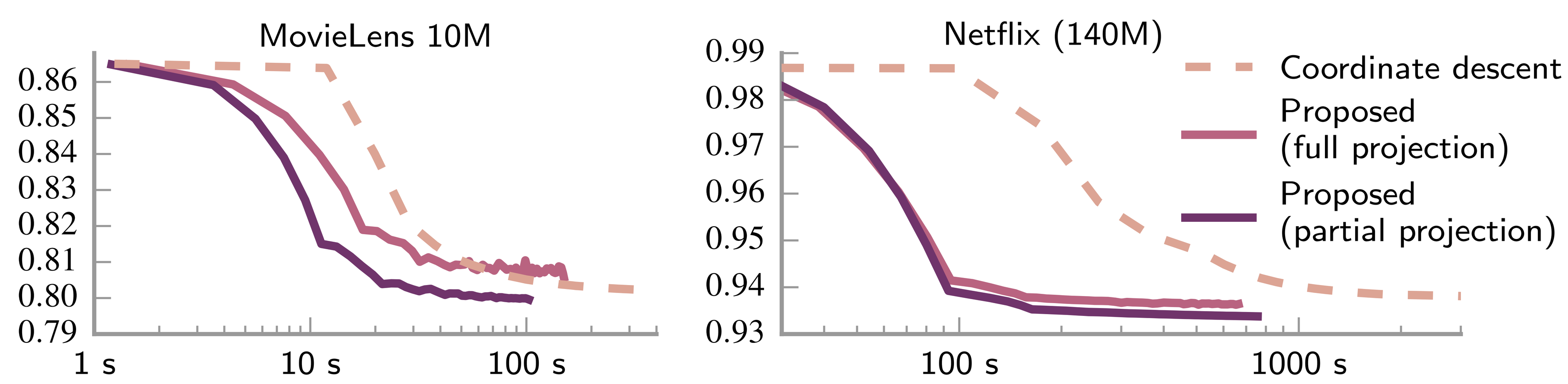



- Speed-up close to reduction factor
- Information is retrieved faster
- Final results have comparable performance

→ Similar explained variance / sparsity

△ Slighty increased sparsity for high regularization

## Collaborative filtering




- Natural masks: $\mathbf{M}_t\mathbf{x}_t \leftarrow$ movie ratings from user $t$ (setting of [5])
- Compared to coordinate descent for MMMF loss (no hyperparmaters)

- Outperform CD beyond 10M ratings
- Same prediction performance

… On a simple linear interaction model

- **Speed-up 6.8× on Netflix**
- Algorithm not sensitive to hyperparameters


[1] J. Duchi, S. Shalev-Shwartz, Y. Singer, and T. Chandra. Efficient projections onto the l 1-ball for learning in high dimensions. In *ICML*, pages 272–279, 2008.

[2] J. Mairal. Stochastic majorization-minimization algorithms for large-scale optimization. In *Advances in Neural Information Processing Systems*, pages 2283–2291, 2013.

[3] 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.

[4] A. Mensch, J. Mairal, B. Thirion, and G. Varoquaux. Dictionary Learning for Massive Matrix Factorization. *Proceedings of The 33rd ICML*, pages 1737–1746, 2016.

[5] Z. Szabó, B. Póczos, and A. Lorincz. Online group-structured dictionary learning. In *Proceedings of CVPR*, pages 2865–2872. IEEE, 2011.