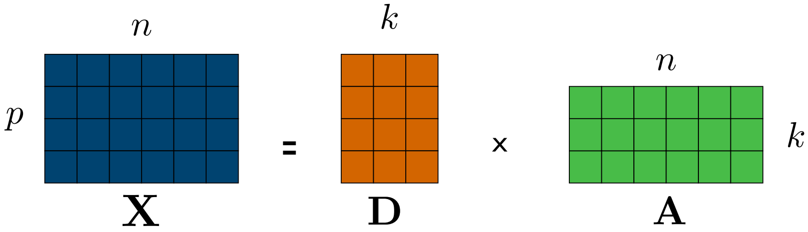
Stochastic Subsampling for Massive Matrix Factorization

Arthur Mensch, Julien Mairal,
Gaël Varoquaux, Bertrand Thirion

Inria Parietal, Université Paris-Saclay

April 13, 2018

Matrix factorization



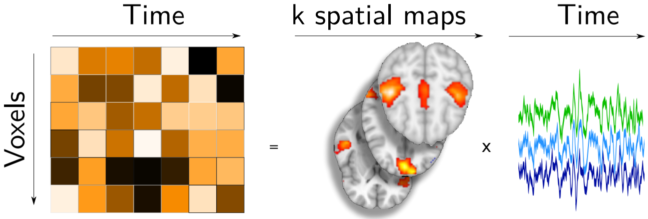
- $\mathbf{X} \in \mathbb{R}^{p \times n} = \mathbf{DA} \in \mathbb{R}^{p \times k} \times \mathbb{R}^{k \times n}$
- Flexible tool for unsupervised data analysis
- Dataset has lower underlying complexity than appearing size

How to scale it to very large datasets ?
(Brain imaging, **4TB**, hyperspectral imaging, **100 GB**)

Example: resting-state fMRI

Resting-state data analysis:

- **Input:** 3D brain images across time for 900 subjects
- $\mathbf{X} \in \mathbb{R}^{p \times n}$, $n = 5 \cdot 10^6$, $p = 2 \cdot 10^5$
- **Goal:** Extract representative sparse brain components $\mathbf{D}$
- **Functional networks:** correlated brain activity in localized areas (*e.g.*, auditory, visual, motor cortex)



$\mathbf{X} \in \mathbb{R}^{p \times n}$ $\quad$ $\mathbf{D} \in \mathbb{R}^{p \times k}$ $\quad$ $\mathbf{A} \in \mathbb{R}^{k \times n}$

Other examples

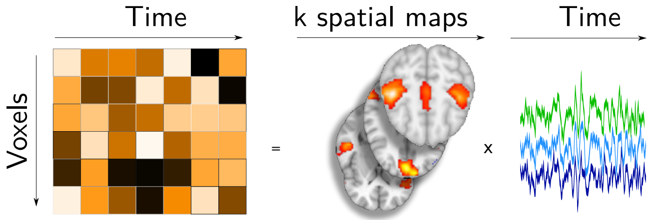
Computer vision:

- Patches of an image
- Decompose onto a **dictionary**
- Sparse loadings

Collaborative filtering:

- Incomplete user-item rating matrix
- Decompose into a low-rank factorisation

Designing new efficient algorithms



$\mathbf{X} \in \mathbb{R}^{p \times n}$ $\mathbf{D} \in \mathbb{R}^{p \times k}$ $\mathbf{A} \in \mathbb{R}^{k \times n}$

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

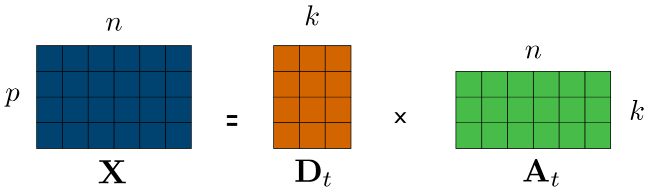
New stochastic algorithms that scale in **both** directions

Formalism and methods

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)



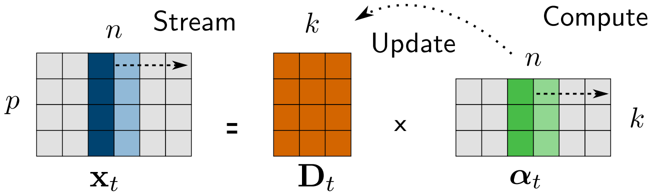
Naive resolution:

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

Online matrix factorization [Mairal et al., 2010]

Scaling in 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 → large speed-up



Use case:

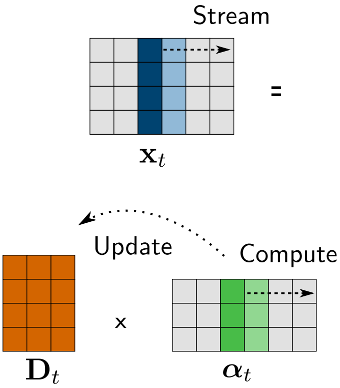
- Large n, regular p, *e.g.*, image patches:

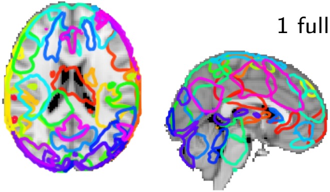
$$p = 256 \qquad n \approx 10^6 \qquad \textbf{1GB}$$

- Low-rank factorization / sparse coding

Scaling-up for massive matrices

Out-of-the-box online algorithm ?



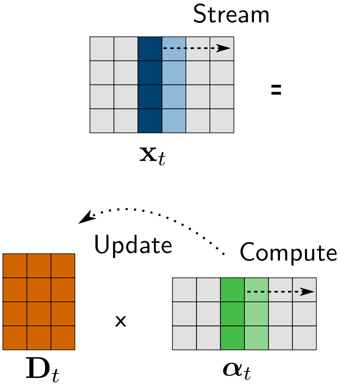


1 full epoch

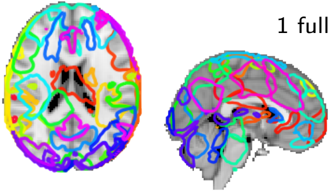
235 h run time

Scaling-up for massive matrices

Out-of-the-box online algorithm ?

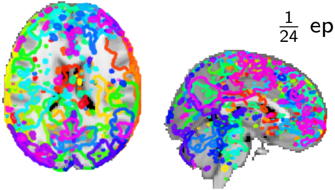


Limited time budget ?



1 full epoch

235 h run time

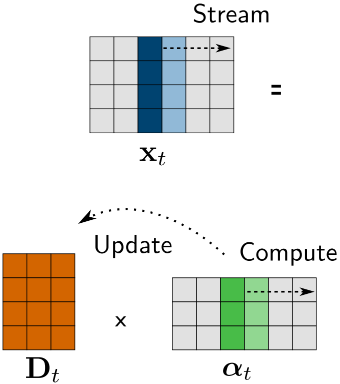


$\frac{1}{24}$ epoch

10 h run time

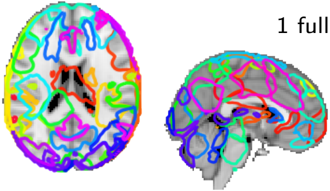
Scaling-up for massive matrices

Out-of-the-box online algorithm ?



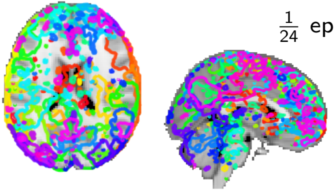
Limited time budget ?

Need to accomodate large p



1 full epoch

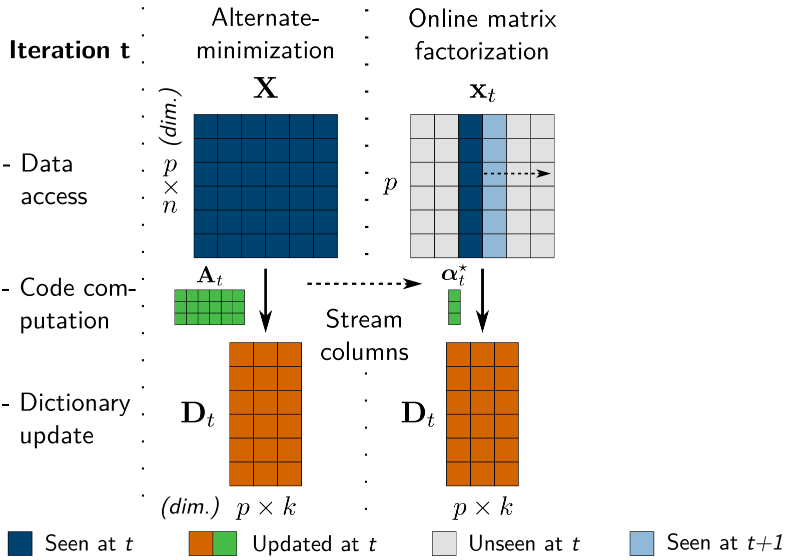
235 h run time



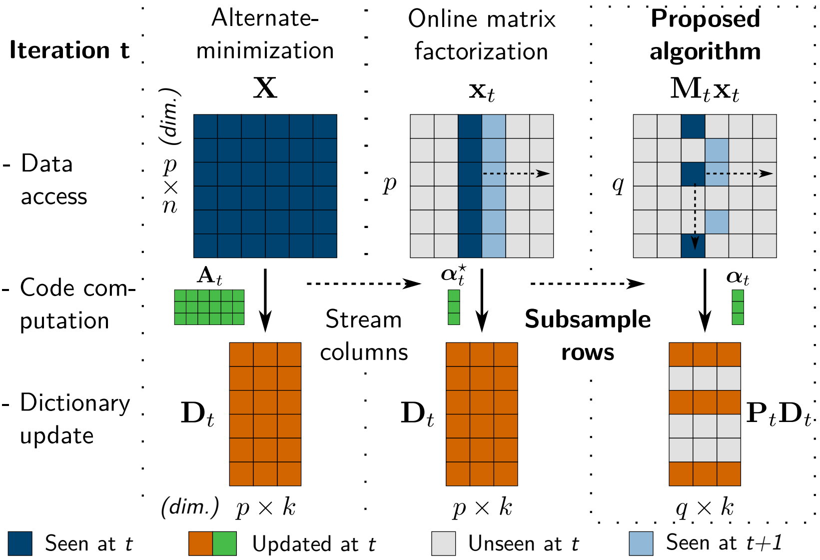
$\frac{1}{24}$ epoch

10 h run time

Scaling-up in both directions



Scaling-up in both directions



Online dictionary learning: details

We learn the **left side factor**: $\mathbf{D}^\star$ solution of

$$\min_{\mathbf{D}\in\mathcal{C}} \frac{1}{n}\sum_{i=1}^{n} \|\mathbf{x}^{(i)} - \mathbf{D}\boldsymbol{\alpha}^{(i)}(\mathbf{D})\|_2^2$$

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

Online dictionary learning: details

We learn the **left side factor**: $\mathbf{D}^\star$ solution of

$$\min_{\mathbf{D}\in\mathcal{C}} \frac{1}{n}\sum_{i=1}^{n} \|\mathbf{x}^{(i)} - \mathbf{D}\boldsymbol{\alpha}^{(i)}(\mathbf{D})\|_2^2$$

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

Expected risk minimization problem: $(i_t)_t$ sampled uniformly

$$\min_{\mathbf{D}\in\mathcal{C}} \mathbb{E}[f_t(\mathbf{D})] \qquad f_t(\mathbf{D}) \triangleq \|\mathbf{x}^{(i_t)} - \mathbf{D}\boldsymbol{\alpha}^{(i_t)}(\mathbf{D})\|_2^2$$

Online dictionary learning: details

We learn the **left side factor**: $\mathbf{D}^\star$ solution of

$$\min_{\mathbf{D}\in\mathcal{C}} \frac{1}{n}\sum_{i=1}^{n} \|\mathbf{x}^{(i)} - \mathbf{D}\boldsymbol{\alpha}^{(i)}(\mathbf{D})\|_2^2$$

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

Expected risk minimization problem: $(i_t)_t$ sampled uniformly

$$\min_{\mathbf{D}\in\mathcal{C}} \mathbb{E}[f_t(\mathbf{D})] \qquad f_t(\mathbf{D}) \triangleq \|\mathbf{x}^{(i_t)} - \mathbf{D}\boldsymbol{\alpha}^{(i_t)}(\mathbf{D})\|_2^2$$

How to minimize it ?

A majorization-minimization algorithm

Expected risk minimization: $x_t \triangleq x^{(i_t)}$

$$\min_{\mathbf{D} \in \mathcal{C}} \bar{f}(\mathbf{D}) = \mathbb{E}[f_t(\mathbf{D})] \qquad f_t(\mathbf{D}) \triangleq \|\mathbf{x}_t - \mathbf{D}\boldsymbol{\alpha}_t(\mathbf{D})\|_2^2$$

A majorization-minimization algorithm

Expected risk minimization: $x_t \triangleq x^{(i_t)}$

$$\min_{\mathbf{D}\in\mathcal{C}} \bar{f}(\mathbf{D}) = \mathbb{E}[f_t(\mathbf{D})] \qquad f_t(\mathbf{D}) \triangleq \|\mathbf{x}_t - \mathbf{D}\boldsymbol{\alpha}_t(\mathbf{D})\|_2^2$$

At iteration t: we build a **pointwise majorizing surrogate**

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

$$g_t(\mathbf{D}) = \|\mathbf{x}_t - \mathbf{D}\boldsymbol{\alpha}_t\|_2^2 \geq f_t(\mathbf{D}) \qquad g_t(\mathbf{D}_{t-1}) = f_t(\mathbf{D}_{t-1})$$

A majorization-minimization algorithm

Expected risk minimization: $x_t \triangleq x^{(i_t)}$

$$\min_{\mathbf{D}\in\mathcal{C}} \bar{f}(\mathbf{D}) = \mathbb{E}[f_t(\mathbf{D})] \qquad f_t(\mathbf{D}) \triangleq \|\mathbf{x}_t - \mathbf{D}\boldsymbol{\alpha}_t(\mathbf{D})\|_2^2$$

At iteration t: we build a **pointwise majorizing surrogate**

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

$$g_t(\mathbf{D}) = \|\mathbf{x}_t - \mathbf{D}\boldsymbol{\alpha}_t\|_2^2 \geq f_t(\mathbf{D}) \qquad g_t(\mathbf{D}_{t-1}) = f_t(\mathbf{D}_{t-1})$$

We minimize the aggregated surrogate

$$\bar{g}_t(\mathbf{D}) \triangleq \frac{1}{t}\sum_{s=1}^{t} g_s(\mathbf{D}) \geq \frac{1}{t}\sum_{s=1}^{t} f_s(\mathbf{D}) \triangleq \bar{f}_t(\mathbf{D})$$

and obtain $\mathbf{D}_t$

A majorization-minimization algorithm

The surrogate is a simple quadratic

$$\bar{g}_t(\mathbf{D}) = \mathrm{Tr}\left(\mathbf{D}^\top \mathbf{D} \bar{\mathbf{C}}_t - \mathbf{D}^\top \bar{\mathbf{B}}_t\right)$$

with parameters that can be updated **online**

$$\bar{\mathbf{C}}_t = \frac{1}{t} \sum_{s=1}^{t} \boldsymbol{\alpha}_s \boldsymbol{\alpha}_s^\top \qquad \bar{\mathbf{B}}_t = \frac{1}{t} \sum_{s=1}^{t} \mathbf{x}_s \boldsymbol{\alpha}_s^\top$$

The surrogate is minimized using **block coordinate descent**

Convergence guarantees (informal)

$(\mathbf{D}_t)_t$ converges a.s. towards a critical point of the expected risk

$$\min_{\mathbf{D}\in\mathcal{C}} \bar{f}(\mathbf{D}) = \mathbb{E}[f_t(\mathbf{D})]$$

Convergence guarantees (informal)

$(\mathbf{D}_t)_t$ converges a.s. towards a critical point of the expected risk

$$\min_{\mathbf{D} \in \mathcal{C}} \bar{f}(\mathbf{D}) = \mathbb{E}[f_t(\mathbf{D})]$$

Major lemma: $\bar{g}_t$ is a tighter and tighter surrogate:

$$\bar{g}_t(\mathbf{D}_{\mathbf{D}_{t-1}}) - \bar{f}_t(\bar{g}_t(\mathbf{D}_{t-1})) \to 0$$

and the algorithm is asymptotically a majorization-minimization algorithm.

Algorithm design: summary

Online dictionary learning [Mairal et al., 2010]

1. **Compute code**

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

2. **Update surrogate**

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

3. **Minimize surrogate**

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

Algorithm design: summary

Online dictionary learning [Mairal et al., 2010]

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

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

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

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

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

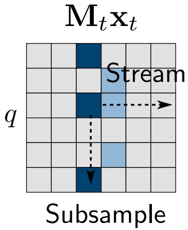
$$\mathbf{D}_t = \operatorname*{argmin}_{\mathbf{D} \in \mathcal{C}} \bar{g}_t(\mathbf{D}) = \operatorname*{argmin}_{\mathbf{D} \in \mathcal{C}} \mathrm{Tr}\left(\mathbf{D}^\top \mathbf{D}\mathbf{C}_t - \mathbf{D}^\top \mathbf{B}_t\right)$$

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

Stochastic subsampling

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

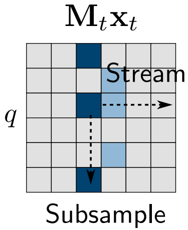
- Sample masking matrix $\mathbf{M}_t$
- Diagonal matrix with rescaled Bernouilli coefficients, $\mathbb{E}[\mathrm{rank}\, \mathbf{M}_t] = q$



Stochastic subsampling

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

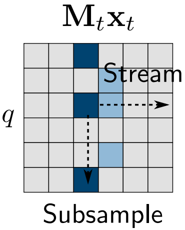
- Sample masking matrix $\mathbf{M}_t$
- Diagonal matrix with rescaled Bernouilli coefficients, $\mathbb{E}[\mathrm{rank}\ \mathbf{M}_t] = q$
- $\mathbf{x}_t \to \mathbf{M}_t \mathbf{x}_t$, $\mathbb{E}[\mathbf{M}_t \mathbf{x}_t] = \mathbf{x}_t$



Stochastic 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

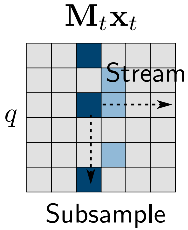


Stochastic subsampling

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

Noisy updates **but single iteration in $\mathcal{O}(q)$**

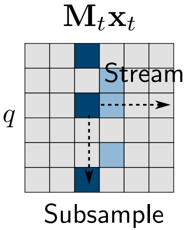


Stochastic subsampling

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

Noisy updates **but single iteration in $\mathcal{O}(q)$**



Subsampled Online matrix Factorization (SOMF)

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

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

1. Code computation

Linear regression with random sampling

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

approximative (sketched) solution of

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

$(\mathbf{M}_t)_t$ introduces errors in $(\boldsymbol{\alpha}_t)_t$ computations

- The error can be controlled and reduced

3. Surrogate minimization

Original OMF: block coordinate descent with projection on $\mathcal{C}$

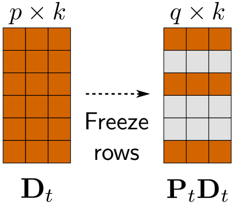
$$\min_{\mathbf{D}\in\mathcal{C}} \bar{g}_t(\mathbf{D}) \qquad\qquad \mathbf{d}^{(j)} \leftarrow p^{\perp}_{\mathcal{C}[j,j]}(\mathbf{d}^{(j)} - \frac{1}{\bar{\mathbf{C}}_{j,j}}(\mathbf{D}\bar{\mathbf{c}}_t^{(j)} - \bar{\mathbf{b}}_t^{(j)})$$

Original OMF: block coordinate descent with projection on $\mathcal{C}$

$$\min_{\mathbf{D} \in \mathcal{C}} \bar{g}_t(\mathbf{D}) \qquad \mathbf{d}^{(j)} \leftarrow p^{\perp}_{\mathcal{C}[j,j]}(\mathbf{d}^{(j)} - \frac{1}{\bar{\mathbf{C}}_{j,j}}(\mathbf{D}\bar{\mathbf{c}}_t^{(j)} - \bar{\mathbf{b}}_t^{(j)})$$

SOMF: Freeze the rows not selected by $\mathbf{M}_t$

$$\min_{\substack{\mathbf{D} \in \mathcal{C} \\ \mathbf{P}_t^{\perp}\mathbf{D} = \mathbf{P}_t^{\perp}\mathbf{D}_{t-1}}} \bar{g}_t(\mathbf{D})$$



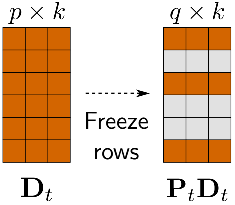
3. Surrogate minimization

Original OMF: block coordinate descent with projection on $\mathcal{C}$

$$\min_{\mathbf{D}\in\mathcal{C}} \bar{g}_t(\mathbf{D}) \qquad \mathbf{d}^{(j)} \leftarrow p^{\perp}_{\mathcal{C}[j,j]}\Big(\mathbf{d}^{(j)} - \frac{1}{\bar{\mathbf{C}}_{j,j}}(\mathbf{D}\bar{\mathbf{c}}^{(j)}_t - \bar{\mathbf{b}}^{(j)}_t)\Big)$$

SOMF: Freeze the rows not selected by $\mathbf{M}_t$

$$\min_{\substack{\mathbf{D}\in\mathcal{C}\\ \mathbf{P}_t^{\perp}\mathbf{D}=\mathbf{P}_t^{\perp}\mathbf{D}_{t-1}}} \bar{g}_t(\mathbf{D})$$



Reduces to a block coordinate descent in $\mathbb{R}^{q\times k}$!

Theoretical analysis

Convergence theorem (informal)

$\bar{f}(\mathbf{D}_t)$ 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 $\mathbf{D} \in \mathcal{C}$

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

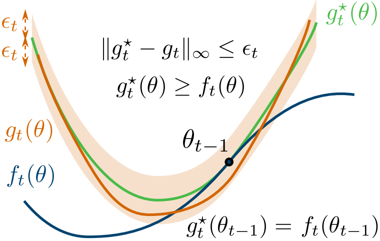
Theoretical analysis

Convergence theorem (informal)

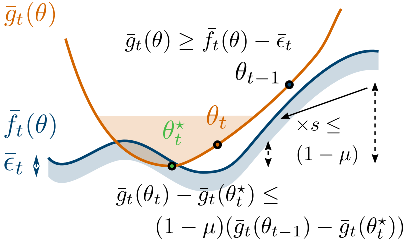
$\bar{f}(\mathbf{D}_t)$ converges with probability one and every limit point $\mathbf{D}_\infty$ of $(\mathbf{D}_t)_t$ is a stationary point of $\bar{f}$: for all $\mathbf{D} \in \mathcal{C}$

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

Surrogate approximation

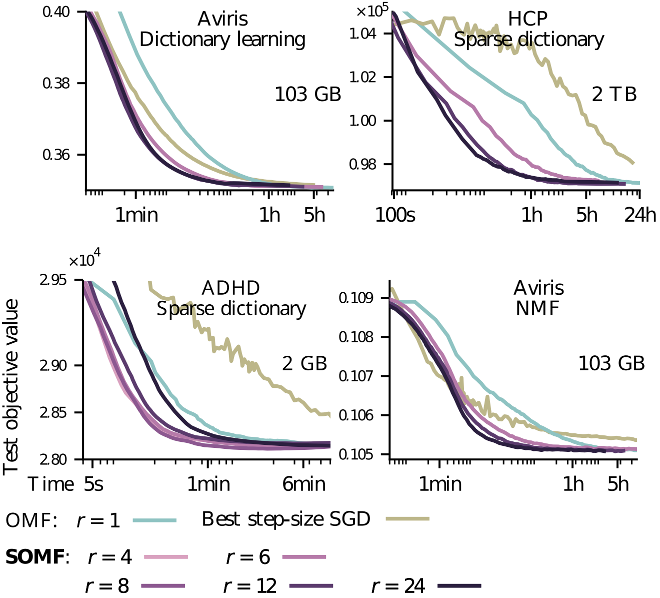


Partial minimization



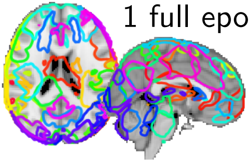
Proof: Control perturbation (red) from the online matrix factorization algorithm (green)

Results: up to 12x speed-up



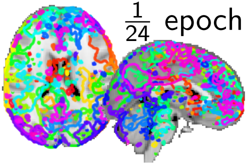
Resting-state fMRI

Online dictionary learning



1 full epoch

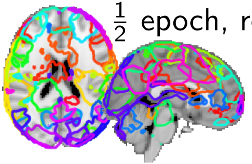
235 h run time



$\frac{1}{24}$ epoch

10 h run time

Proposed method

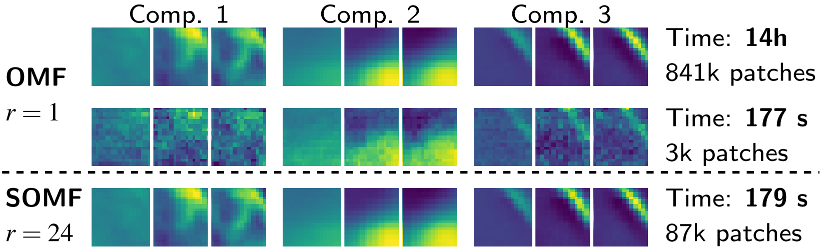


$\frac{1}{2}$ epoch, reduction $r=12$

10 h run time

Qualitatively, usable maps are obtained **10× faster**

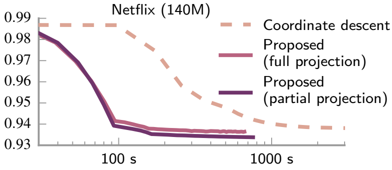
Hyperspectral imaging



SOMF atoms are more focal and less noisy given a certain time budget

Collaborative filtering

- $\mathbf{M}_t \mathbf{x}_t$ movie ratings from user t
- vs. coordinate descent for MMMF loss (no hyperparameters)



Dataset	Test RMSE		Speed
	CD	**MODL**	-up
ML 1M	0.872	**0.866**	$\times$0.75
ML 10M	0.802	**0.799**	$\times$**3.7**
NF (140M)	0.938	**0.934**	$\times$**6.8**

- Outperform coordinate descent beyond 10M ratings
- Same prediction performance
- Speed-up 6.8$\times$ on Netflix

Conclusion

New efficient algorithm with many potential use-case.

- Subsampling mini-batches at each iteration.

Python package (`github.com/arthurmensch/modl`)

Perspectives:

- Efficient heuristics and adaptative subsampling ratio
- Is this kind of approach transposable to SGD setting ?

Publications

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

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

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