Matrices efficientes pour le traitement du signal et l'apprentissage automatique

Soutenance de thèse
présentée par **Luc Le Magoarou**
sous la direction de **Rémi Gribonval**

Inria
Centre Inria Rennes - Bretagne Atlantique

24 novembre 2016

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

erc European Research Council Established by the European Commission

PLEASE projection, learning and sparsity for efficient data processing

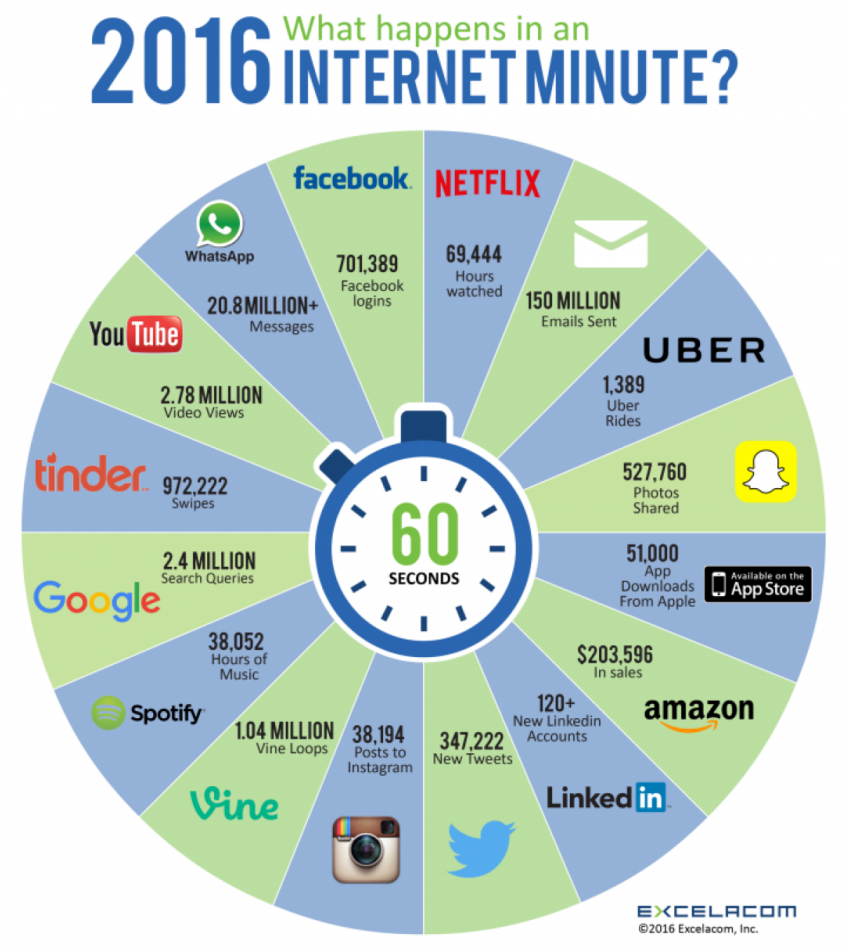
Inria INVENTEURS DU MONDE NUMÉRIQUE

Motivation

Des données sont produites en quantité énorme.

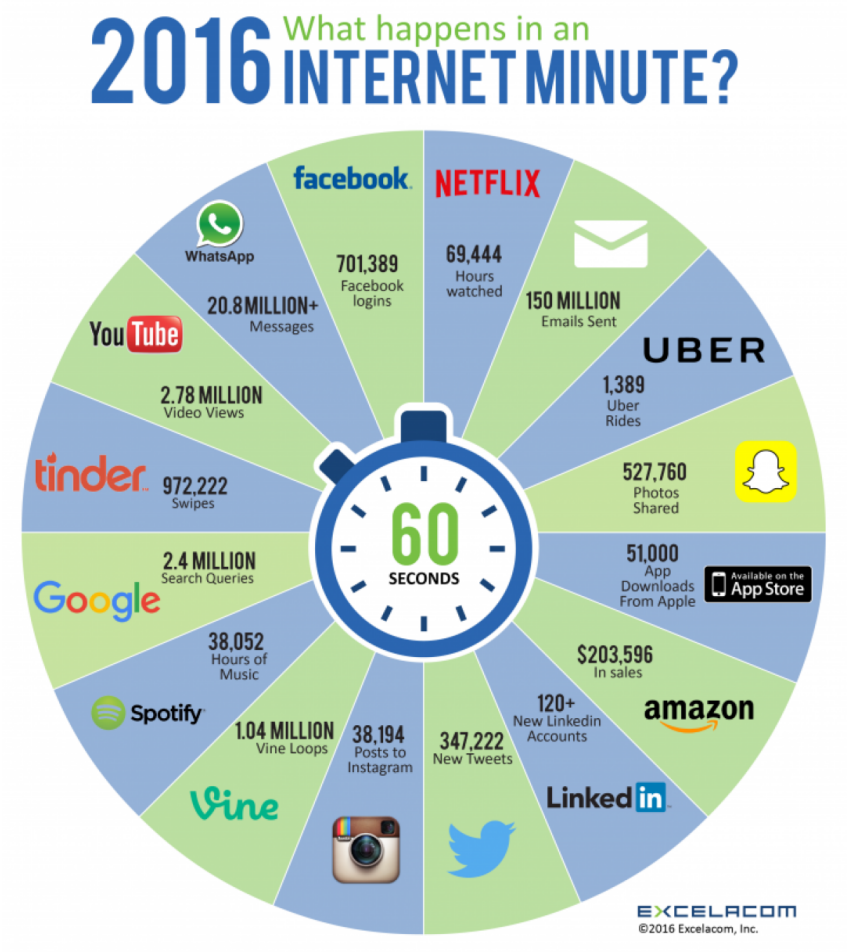
Motivation

Des données sont produites en quantité énorme.



Motivation

Des données sont produites en quantité énorme.



Comment traiter ces données ?

Motivation

Une tâche de traitement de données peut s'analyser à partir :

- de l'**objectif** fixé,
- des **moyens** mis en oeuvre,
- du **résultat** atteint.

Motivation

Une tâche de traitement de données peut s'analyser à partir :

- de l'**objectif** fixé,
- des **moyens** mis en oeuvre,
- du **résultat** atteint.

On considère deux mesures de performance :

- **Efficacité** (résultat/objectif)
- **Efficience** (résultat/moyens)

Motivation

Une tâche de traitement de données peut s'analyser à partir :

- de l'**objectif** fixé,
- des **moyens** mis en oeuvre,
- du **résultat** atteint.

On considère deux mesures de performance :

- **Efficacité** (résultat/objectif)
- **Efficience** (résultat/moyens)

Des données massives requièrent des traitements efficients.

Motivation

Les applications linéaires sont partout en traitement de données.

$$\mathbf{y} = \mathbf{A}\mathbf{x}$$

Motivation

Les applications linéaires sont partout en traitement de données.

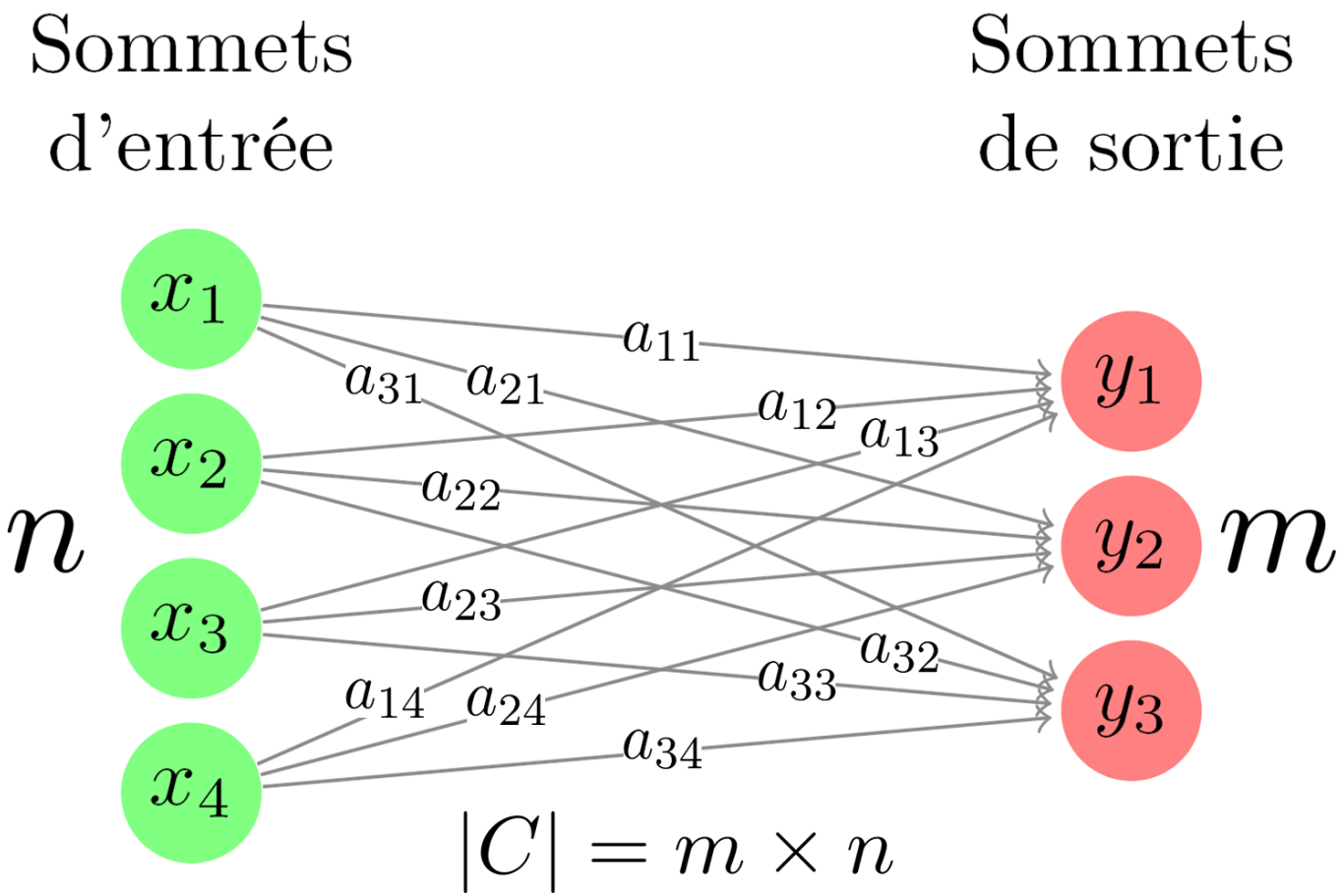
$$\mathbf{y} = \mathbf{A}\mathbf{x}$$

Peut-on effectuer des applications linéaires de manière efficiente ?

Sommaire

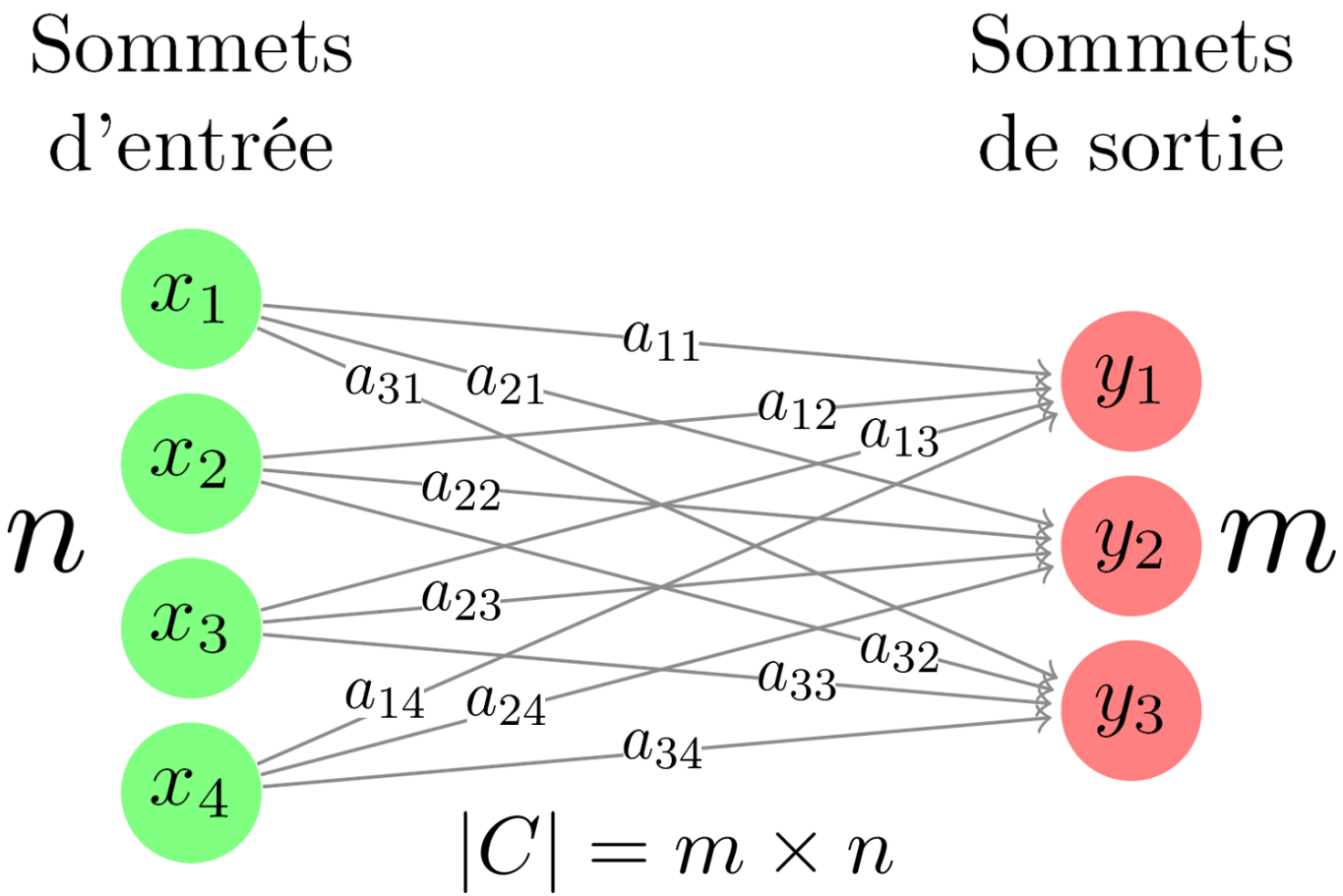
Matrices efficientes

On associe un circuit linéaire C à toute matrice $\mathbf{A} \in \mathbb{R}^{m \times n}$:



Matrices efficientes

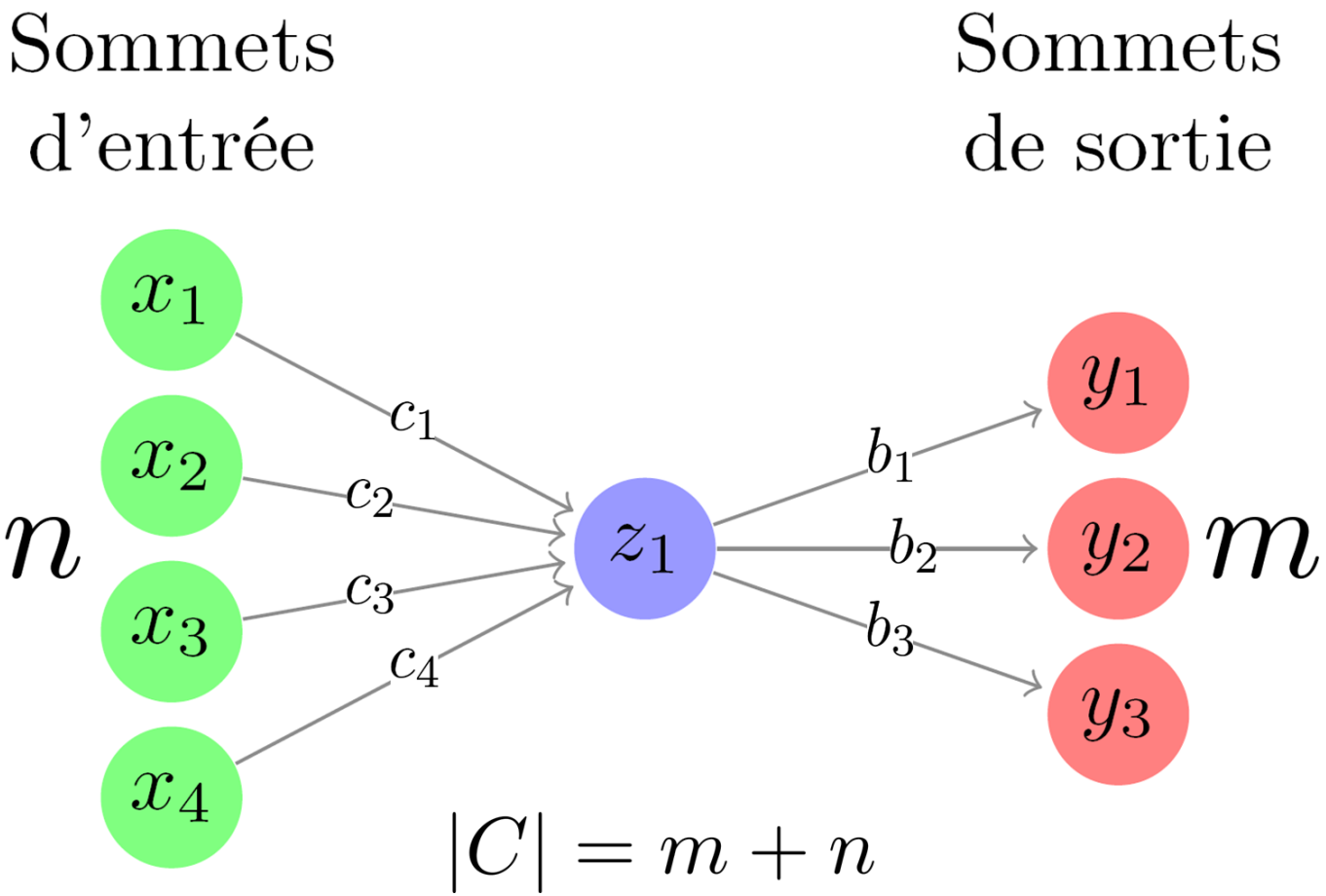
On associe un circuit linéaire C à toute matrice $\mathbf{A} \in \mathbb{R}^{m \times n}$:



Existe-t-il des circuits de plus petite taille ?

Matrices efficientes

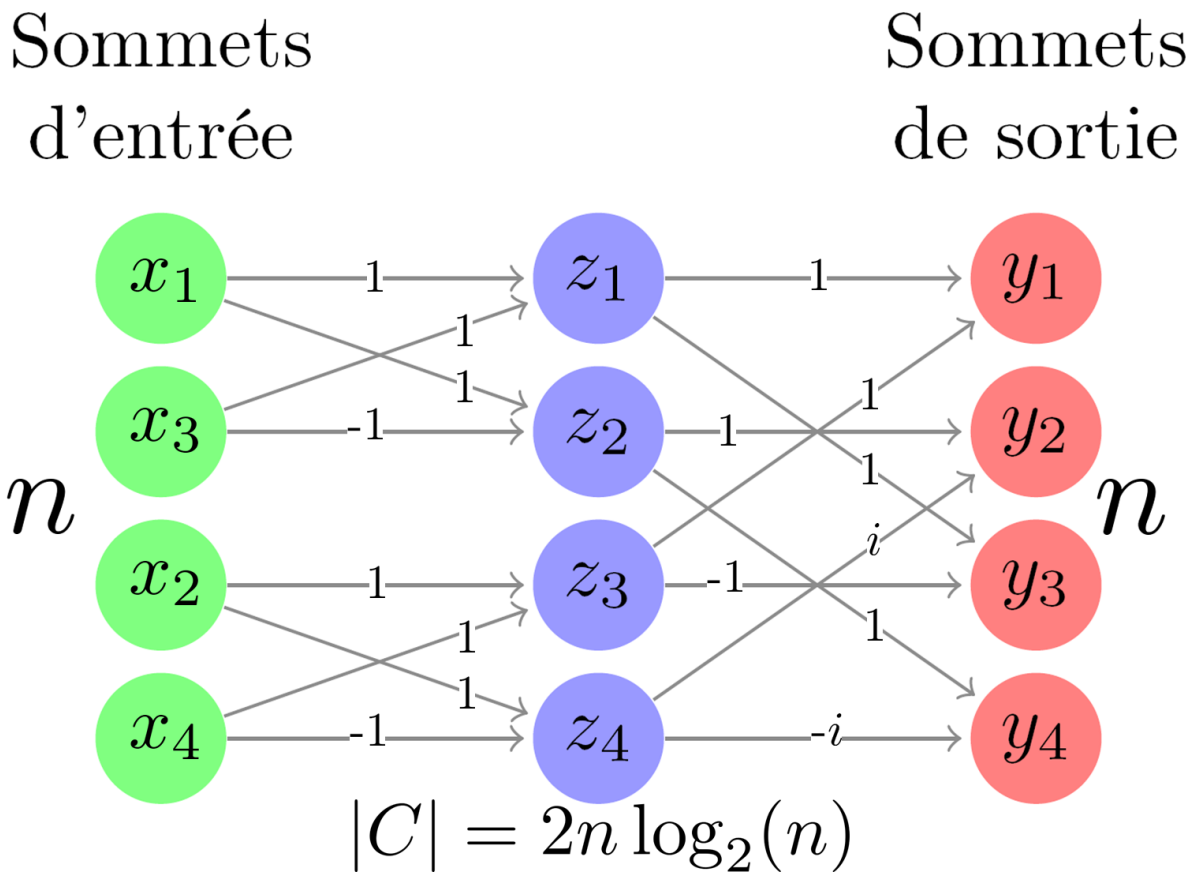
Si $\mathbf{A} \in \mathbb{R}^{m \times n}$ est de rang faible ($\mathbf{A} = \mathbf{B}\mathbf{C}^T$) :



$$|C| = m + n$$

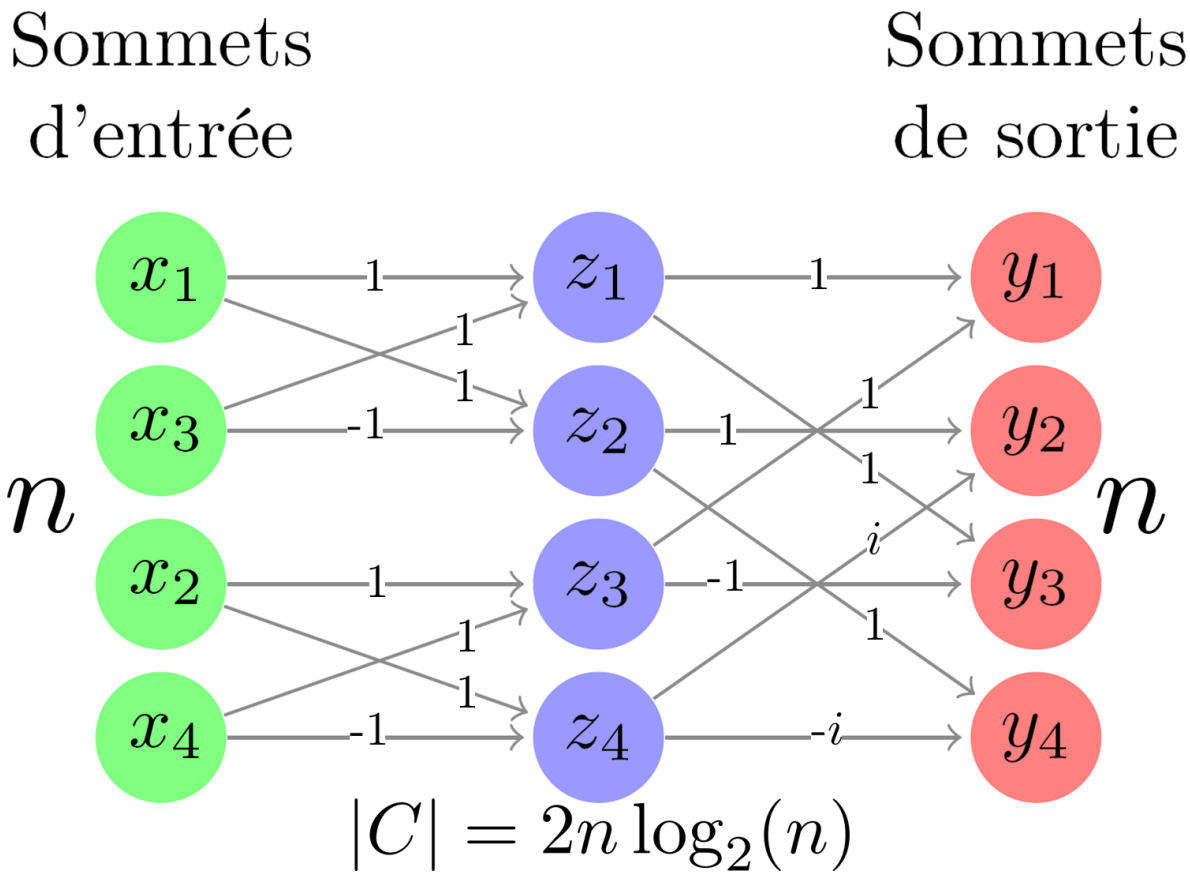
Matrices efficientes

Si $\mathbf{A} \in \mathbb{R}^{m \times n}$ correspond à une application pour laquelle il existe un algorithme rapide :



Matrices efficientes

Si $\mathbf{A} \in \mathbb{R}^{m \times n}$ correspond à une application pour laquelle il existe un algorithme rapide :



Dans ce cas : $\mathbf{A} = \mathbf{S}_J \dots \mathbf{S}_1$.

Matrices efficientes

Définition

L'efficience d'un circuit linéaire C ayant n entrées et m sorties est égale au rapport suivant :

$$\mathscr{E}(C) \triangleq \frac{mn}{|C|}.$$

Définition

L'efficience d'une matrice $\mathbf{A}$, notée $\mathscr{E}(\mathbf{A})$, est égale à l'efficience du circuit associé à $\mathbf{A}$ le plus efficient :

$$\mathscr{E}(\mathbf{A}) \triangleq \max_{C \in \mathcal{C}_{\mathbf{A}}} \mathscr{E}(C).$$

Matrices efficientes

Question : *Peut-on calculer ou borner l'efficience d'une matrice ?*

Matrices efficientes

Question : *Peut-on calculer ou borner l'efficience d'une matrice ?*

La question reste ouverte. [1] [2] [3] [4]

[1] J. Morgenstern, **The Linear Complexity of Computation.** *J. ACM*, 1975

[2] L. Valiant, **Graph-theoretic arguments in low-level complexity.** *Math. Found. of Computer Science*, 1977

[3] P. Pudlak, **A note on the use of determinant for proving lower bounds on the size of linear circuits.** *Inf. Process. Lett.*, 2000

[4] S. Lokam, **Complexity lower bounds using linear algebra.** *F. and T. in theoretical computer science*, 2009

Objectif général

Approcher ou estimer des matrices $\mathbf{A} \in \mathbb{R}^{m \times n}$, sous la contrainte

$$\hat{\mathbf{A}} = \mathbf{S}_J \dots \mathbf{S}_1,$$

où les $\mathbf{S}_j$, $j \in \{1, \dots, J\}$ sont des matrices creuses telles que $\sum_{j=1}^{J} \|\mathbf{S}_j\|_0 \leq mn$.

Objectif général

Approcher ou estimer des matrices $\mathbf{A} \in \mathbb{R}^{m \times n}$, sous la contrainte

$$\hat{\mathbf{A}} = \mathbf{S}_J \dots \mathbf{S}_1,$$

où les $\mathbf{S}_j$, $j \in \{1, \dots, J\}$ sont des matrices creuses telles que $\sum_{j=1}^{J} \|\mathbf{S}_j\|_0 \leq mn$.

- La matrice $\hat{\mathbf{A}}$ est efficiente par construction

Objectif général

Approcher ou estimer des matrices $\mathbf{A} \in \mathbb{R}^{m \times n}$, sous la contrainte

$$\hat{\mathbf{A}} = \mathbf{S}_J \dots \mathbf{S}_1,$$

où les $\mathbf{S}_j$, $j \in \{1, \dots, J\}$ sont des matrices creuses telles que $\sum_{j=1}^{J} \|\mathbf{S}_j\|_0 \leq mn$.

- La matrice $\hat{\mathbf{A}}$ est efficiente par construction
- $\mathscr{E}(\hat{\mathbf{A}}) \geq \frac{mn}{\sum_{j=1}^{J} \|\mathbf{S}_j\|_0} \triangleq$ RCG

Objectif général

Approcher ou estimer des matrices $\mathbf{A} \in \mathbb{R}^{m \times n}$, sous la contrainte

$$\hat{\mathbf{A}} = \mathbf{S}_J \dots \mathbf{S}_1,$$

où les $\mathbf{S}_j$, $j \in \{1, \dots, J\}$ sont des matrices creuses telles que $\sum_{j=1}^{J} \|\mathbf{S}_j\|_0 \leq mn$.

- La matrice $\hat{\mathbf{A}}$ est efficiente par construction
- $\mathscr{E}(\hat{\mathbf{A}}) \geq \frac{mn}{\sum_{j=1}^{J} \|\mathbf{S}_j\|_0} \triangleq$ RCG
- $\mathbf{S}_J \dots \mathbf{S}_1 \triangleq \prod_{j=1}^{J} \mathbf{S}_j$: FAμST (Flexible Approximate MUlti-layer Sparse Transform)

Sommaire

Problème d'optimisation

- **Entrée :** matrice $\mathbf{A} \in \mathbb{R}^{m \times n}$

Problème d'optimisation

- **Entrée :** matrice $\mathbf{A} \in \mathbb{R}^{m \times n}$
- **But :** trouver J matrices creuses $\mathbf{S}_j$ telles que $\mathbf{A} \approx \mathbf{S}_J \dots \mathbf{S}_1$

Problème d'optimisation

- **Entrée :** matrice $\mathbf{A} \in \mathbb{R}^{m \times n}$
- **But :** trouver J matrices creuses $\mathbf{S}_j$ telles que $\mathbf{A} \approx \mathbf{S}_J \dots \mathbf{S}_1$
- **Méthode :**

$$\underset{\lambda, \mathbf{S}_1, \dots, \mathbf{S}_J}{\text{Minimiser}} \quad \underbrace{\frac{1}{2} \left\| \mathbf{A} - \lambda \prod_{j=1}^{J} \mathbf{S}_j \right\|_F^2}_{\text{Attache aux données}}$$

Problème d'optimisation

- **Entrée :** matrice $\mathbf{A} \in \mathbb{R}^{m \times n}$
- **But :** trouver J matrices creuses $\mathbf{S}_j$ telles que $\mathbf{A} \approx \mathbf{S}_J \dots \mathbf{S}_1$
- **Méthode :**

$$\underset{\lambda, \mathbf{S}_1, \dots, \mathbf{S}_J}{\text{Minimiser}} \quad \underbrace{\frac{1}{2} \left\| \mathbf{A} - \lambda \prod_{j=1}^{J} \mathbf{S}_j \right\|_F^2}_{\text{Attache aux données}} + \underbrace{\sum_{j=1}^{J} \delta_{\mathcal{S}_j}(\mathbf{S}_j)}_{\text{Parcimonie}}$$

Terme de parcimonie : fonctions indicatrices d'ensembles, par exemple $\mathcal{S}_j = \{\mathbf{S} \in \mathbb{R}^{a_j \times a_{j+1}} : \|\mathbf{S}\|_0 \leq p_j, \|\mathbf{S}\|_F = 1\}$.

Problème d'optimisation

- **Entrée :** matrice $\mathbf{A} \in \mathbb{R}^{m \times n}$
- **But :** trouver J matrices creuses $\mathbf{S}_j$ telles que $\mathbf{A} \approx \mathbf{S}_J \dots \mathbf{S}_1$
- **Méthode :**

$$\underset{\lambda, \mathbf{S}_1, \dots, \mathbf{S}_J}{\text{Minimiser}} \quad \underbrace{\frac{1}{2} \left\| \mathbf{A} - \lambda \prod_{j=1}^{J} \mathbf{S}_j \right\|_F^2}_{\text{Attache aux données}} + \underbrace{\sum_{j=1}^{J} \delta_{\mathcal{S}_j}(\mathbf{S}_j)}_{\text{Parcimonie}}$$

Terme de parcimonie : fonctions indicatrices d'ensembles, par exemple $\mathcal{S}_j = \{\mathbf{S} \in \mathbb{R}^{a_j \times a_{j+1}} : \|\mathbf{S}\|_0 \leq p_j, \|\mathbf{S}\|_F = 1\}$.

Ce problème d'optimisation est non-convexe et non-lisse.

PALM pour l'approximation efficiente

L'algorithme PALM : *Proximal Alternating Linearized Minimization*[5] peut être utilisé avec :

$$H(\lambda, \mathbf{S}_1, \ldots, \mathbf{S}_J) \triangleq \tfrac{1}{2}\left\|\mathbf{A} - \lambda \prod_{j=1}^{J} \mathbf{S}_j\right\|_F^2,$$

et

$$\mathcal{S}_j \triangleq \{\mathbf{S} \in \mathbb{R}^{a_j \times a_{j+1}} : \|\mathbf{S}\|_0 \leq p_j, \|\mathbf{S}\|_F = 1\}.$$

Algorithme : Itération de `palm4MSA`

1: **for** $j = 1$ to J **do**

2: $\quad \mathbf{S}_j^{i+1} \leftarrow P_{\mathcal{S}_j}\left(\mathbf{S}_j^i - \frac{1}{c_j^i}\nabla_{\mathbf{S}_j} H\left(\lambda^i, \mathbf{S}_1^{i+1}, \ldots, \mathbf{S}_j^i, \ldots, \mathbf{S}_J^i\right)\right)$

3: **end for**

[5] J. Bolte et al., **Proximal alternating linearized minimization for nonconvex and nonsmooth problems.** *Math. Program.*, 2013.

PALM pour l'approximation efficiente

L'algorithme PALM : *Proximal Alternating Linearized Minimization*[5] peut être utilisé avec :

$$H(\lambda, \mathbf{S}_1, \ldots, \mathbf{S}_J) \triangleq \tfrac{1}{2}\left\|\mathbf{A} - \lambda \prod_{j=1}^{J} \mathbf{S}_j\right\|_F^2,$$

et

$$\mathcal{S}_j \triangleq \{\mathbf{S} \in \mathbb{R}^{a_j \times a_{j+1}} : \|\mathbf{S}\|_0 \leq p_j, \|\mathbf{S}\|_F = 1\}.$$

Algorithme : Itération de `palm4MSA`

1: **for** $j = 1$ to J **do**

2: $\quad \mathbf{S}_j^{i+1} \leftarrow P_{\mathcal{S}_j}\left(\mathbf{S}_j^i - \frac{1}{c_j^i}\nabla_{\mathbf{S}_j} H\left(\lambda^i, \mathbf{S}_1^{i+1}, \ldots, \mathbf{S}_j^i, \ldots, \mathbf{S}_J^i\right)\right)$

3: **end for**

Convergence : Toute séquence bornée générée par `palm4MSA` converge vers un point stationnaire de l'objectif.

[5] J. Bolte et al., **Proximal alternating linearized minimization for nonconvex and nonsmooth problems.** *Math. Program.*, 2013.

Stratégie hiérarchique

Afin d'initialiser les facteurs dans une bonne région, on adopte une stratégie de factorisation hiérarchique, rappelant l'entraînement couche par couche des réseaux de neurones [6] :

[6] G. Hinton and R. Salakhutdinov, **Reducing the dimensionality of data with neural networks**, *Science*, vol. 313, no. 5786, 2006.

Stratégie hiérarchique

Afin d'initialiser les facteurs dans une bonne région, on adopte une stratégie de factorisation hiérarchique, rappelant l'entraînement couche par couche des réseaux de neurones [6] :

$$\mathbf{A} \approx \mathbf{S}_1 \mathbf{R}_1$$

[6] G. Hinton and R. Salakhutdinov, **Reducing the dimensionality of data with neural networks**, *Science*, vol. 313, no. 5786, 2006.

Stratégie hiérarchique

Afin d'initialiser les facteurs dans une bonne région, on adopte une stratégie de factorisation hiérarchique, rappelant l'entraînement couche par couche des réseaux de neurones [6] :

$$\mathbf{A} \approx \mathbf{S}_1 \mathbf{R}_1 \approx \mathbf{S}_2 \mathbf{R}_2$$

[6] G. Hinton and R. Salakhutdinov, **Reducing the dimensionality of data with neural networks**, *Science*, vol. 313, no. 5786, 2006.

Stratégie hiérarchique

Afin d'initialiser les facteurs dans une bonne région, on adopte une stratégie de factorisation hiérarchique, rappelant l'entraînement couche par couche des réseaux de neurones [6] :

$$\mathbf{A} \approx \mathbf{S}_1 \mathbf{R}_1 \approx \mathbf{S}_2 \mathbf{R}_2 \approx \cdots \approx \mathbf{S}_{J-1} \mathbf{S}_J$$

[6] G. Hinton and R. Salakhutdinov, **Reducing the dimensionality of data with neural networks**, *Science*, vol. 313, no. 5786, 2006.

Stratégie hiérarchique

Afin d'initialiser les facteurs dans une bonne région, on adopte une stratégie de factorisation hiérarchique, rappelant l'entraînement couche par couche des réseaux de neurones [6] :

$$\mathbf{A} \approx \mathbf{S}_1 \mathbf{R}_1 \approx \mathbf{S}_2 \mathbf{R}_2 \approx \cdots \approx \mathbf{S}_{J-1} \mathbf{S}_J$$

Cette factorisation hiérarchique est très efficace, les minima locaux atteints sont bien meilleurs.

[6] G. Hinton and R. Salakhutdinov, **Reducing the dimensionality of data with neural networks**, *Science*, vol. 313, no. 5786, 2006.

Problèmes inverses

Données $\mathbf{y}$ et paramètres $\boldsymbol{\gamma}$ sont liés par l'opérateur $\mathbf{M}$:

$$\mathbf{y} \approx \mathbf{M}\boldsymbol{\gamma}$$

Problèmes inverses

Données $\mathbf{y}$ et paramètres $\boldsymbol{\gamma}$ sont liés par l'opérateur $\mathbf{M}$:

$$\mathbf{y} \approx \mathbf{M}\boldsymbol{\gamma}$$

Les méthodes de résolution sont généralement des algorithmes itératifs reposant sur des multiplications par $\mathbf{M}$ et sa transposée, ce qui peut être coûteux en grande dimension.

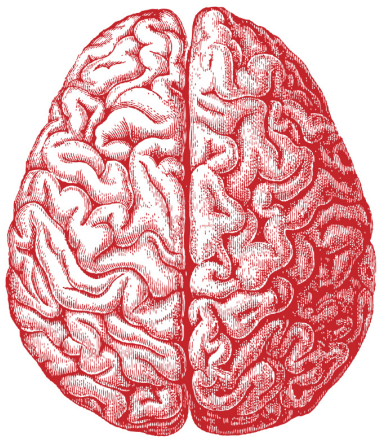
Problèmes inverses

Données $\mathbf{y}$ et paramètres $\boldsymbol{\gamma}$ sont liés par l'opérateur $\mathbf{M}$:

$$\mathbf{y} \approx \underbrace{\mathbf{M}\boldsymbol{\gamma}}_{J} \prod_{j=1} \mathbf{S}_j$$

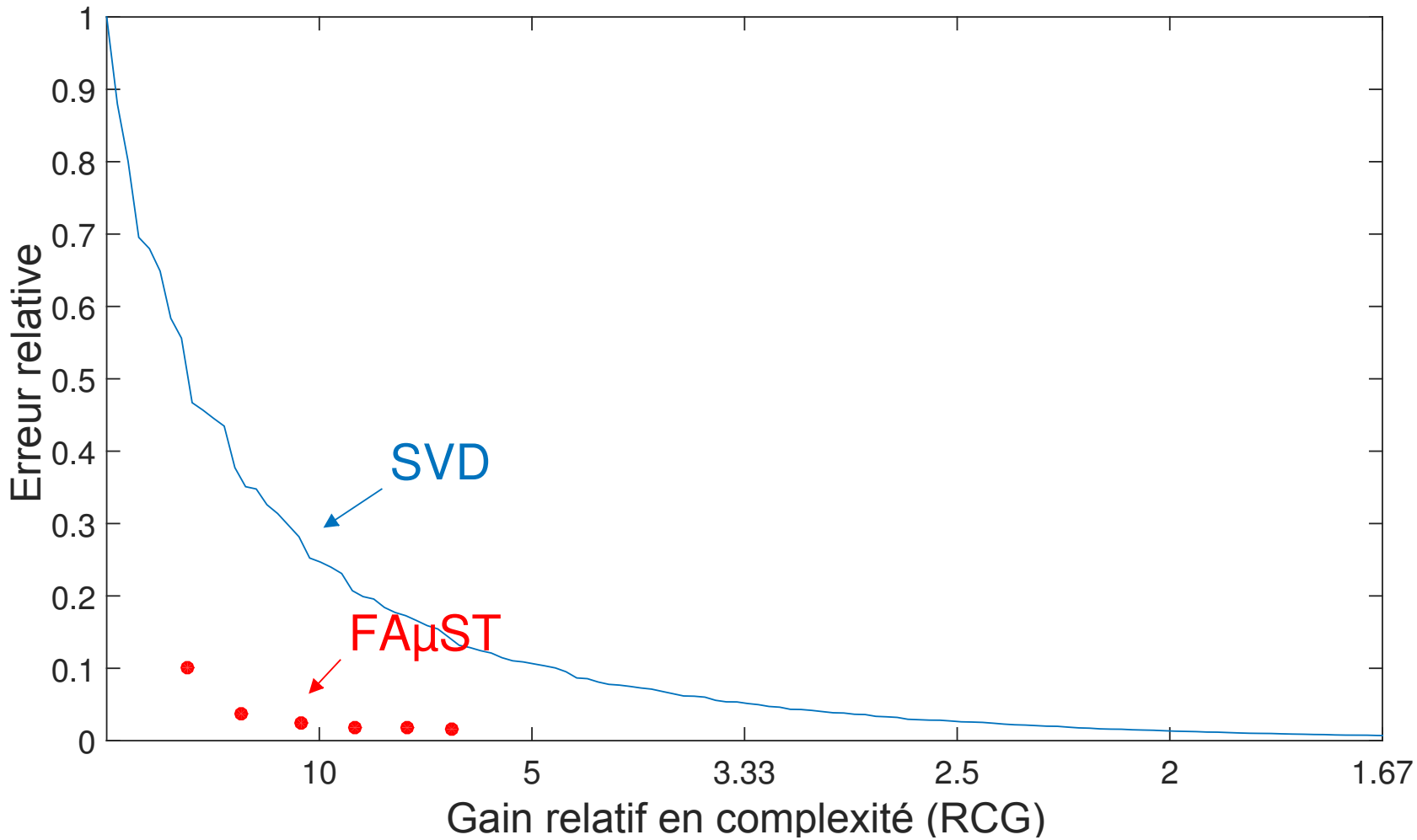
Les méthodes de résolution sont généralement des algorithmes itératifs reposant sur des multiplications par $\mathbf{M}$ et sa transposée, ce qui peut être coûteux en grande dimension.

Problèmes inverses : Imagerie MEG

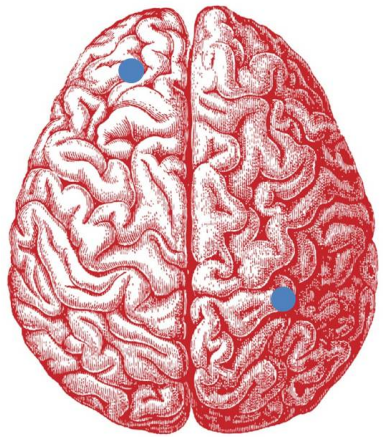


- $\boldsymbol{\gamma} \in \mathbb{R}^{8193}$ représente des sources électriques à différentes positions.
- $\mathbf{y} \in \mathbb{R}^{204}$ est l'intensité du signal mesurée par des électrodes.
- $\mathbf{M} \in \mathbb{R}^{204 \times 8193}$ modélise la physique de la propagation (équations de Maxwell).

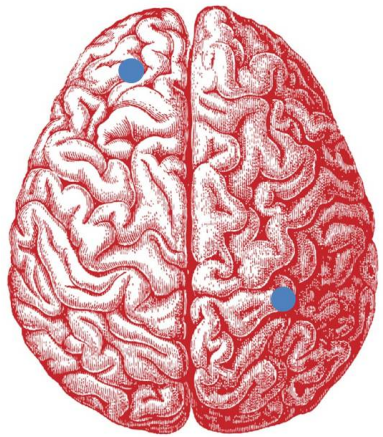
Problèmes inverses : Factorisation de $\mathbf{M}$



Problèmes inverses : Localisation de sources

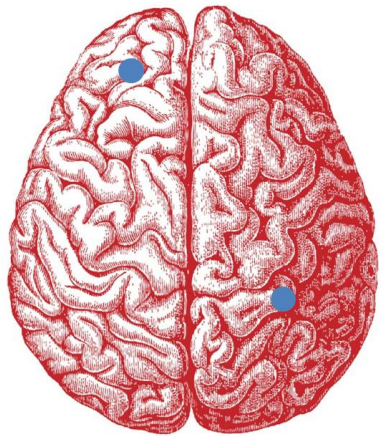


Problèmes inverses : Localisation de sources



$$\|\boldsymbol{\gamma}\|_0 = 2$$

Problèmes inverses : Localisation de sources

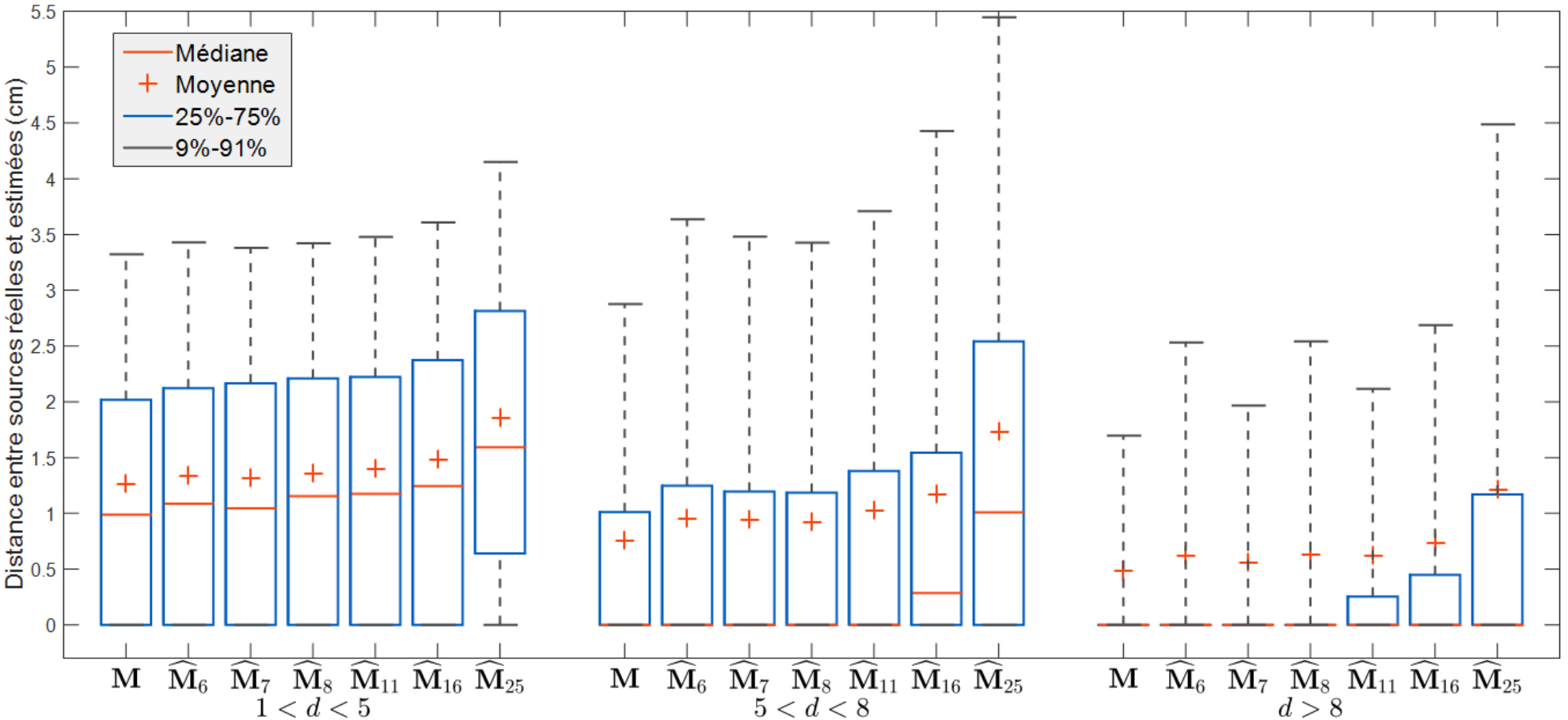


$$\|\boldsymbol{\gamma}\|_0 = 2$$

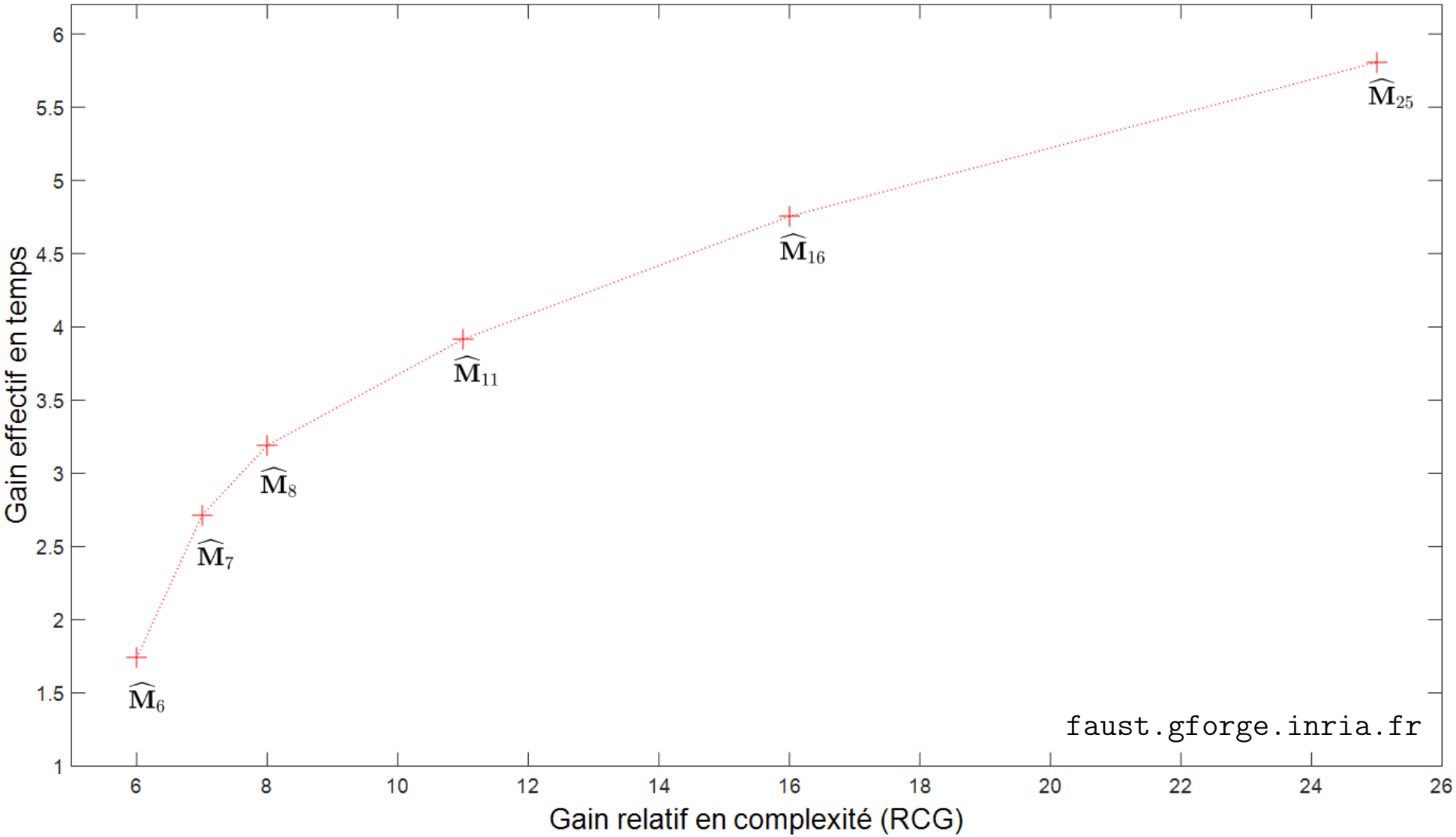
Matrices utilisées :

- La véritable matrice $\mathbf{M}$.
- Les approximations efficientes $\widehat{\mathbf{M}}_{25}$, $\widehat{\mathbf{M}}_{16}$, $\widehat{\mathbf{M}}_{11}$, $\widehat{\mathbf{M}}_{8}$, $\widehat{\mathbf{M}}_{7}$, $\widehat{\mathbf{M}}_{6}$ (où l'indice correspond au RCG arrondi à l'entier le plus proche).

Problèmes inverses : Résultats de localisation



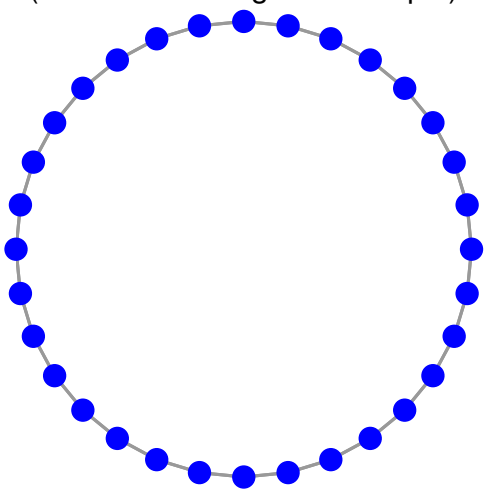
Problèmes inverses : Gain en temps de résolution



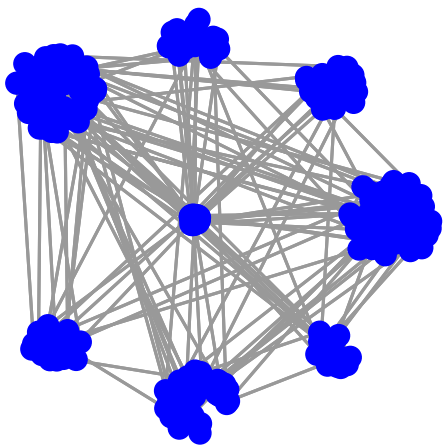
Traitement du signal sur graphe

Objectif : généraliser les outils classiques du traitement du signal aux signaux définis sur les sommets d'un graphe quelconque.

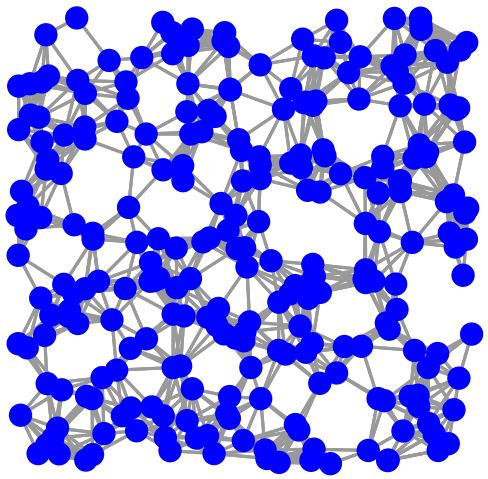
Graphe boucle
(traitement du signal classique)



Graphe de communauté



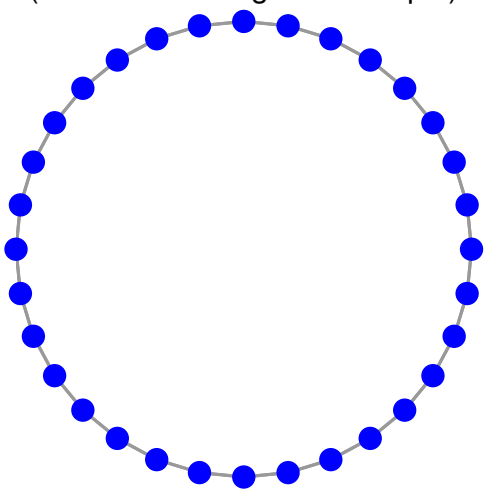
Graphe de réseau de capteurs



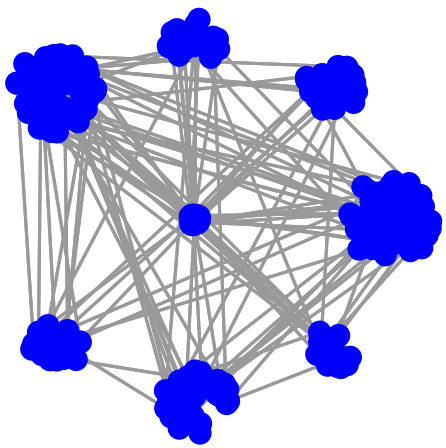
Traitement du signal sur graphe

Objectif : généraliser les outils classiques du traitement du signal aux signaux définis sur les sommets d'un graphe quelconque.

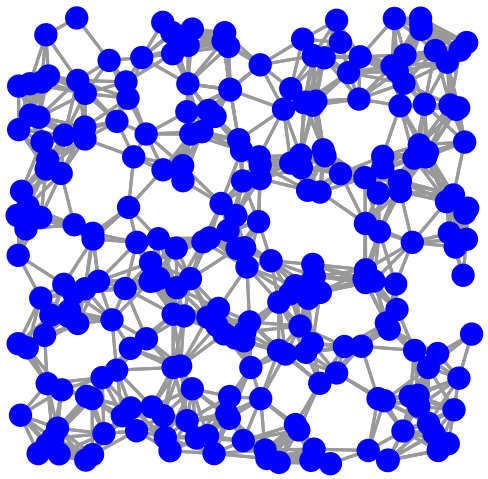
Graphe boucle
(traitement du signal classique)



Graphe de communauté



Graphe de réseau de capteurs



La matrice Laplacienne $\mathbf{L}$ encode la topologie d'un graphe.

FFT sur graphe

Le Laplacien $\mathbf{L}$ est diagonalisable par une matrice orthogonale $\mathbf{U}$:

$$\mathbf{L} = \mathbf{U}\mathbf{D}\mathbf{U}^T$$

FFT sur graphe

Le Laplacien $\mathbf{L}$ est diagonalisable par une matrice orthogonale $\mathbf{U}$:

$$\mathbf{L} = \mathbf{U}\mathbf{D}\mathbf{U}^T$$

La transformée de Fourier sur graphe peut se définir [7] comme le changement de base suivant :

$$\mathbf{y} = \mathbf{U}^T\mathbf{x} \quad \mathcal{O}(n^2)$$

[7] D. Shuman et al., **The emerging field of signal processing on graphs : Extending high-dimensional data analysis to networks and other irregular domains.**, *IEEE SP Mag.*, 2013.

FFT sur graphe

Le Laplacien $\mathbf{L}$ est diagonalisable par une matrice orthogonale $\mathbf{U}$:

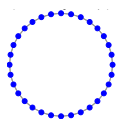
$$\mathbf{L} = \mathbf{U}\mathbf{D}\mathbf{U}^T$$

La transformée de Fourier sur graphe peut se définir comme le changement de base suivant :

$$\mathbf{y} = \mathbf{U}^T\mathbf{x} \quad \mathcal{O}(n^2)$$

Dans le cas classique (graphe boucle), on a :

$$\mathbf{U} = \mathbf{S}_J \dots \mathbf{S}_1 \quad \mathcal{O}(n \log n)$$



FFT sur graphe

Le Laplacien $\mathbf{L}$ est diagonalisable par une matrice orthogonale $\mathbf{U}$:

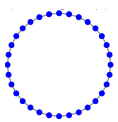
$$\mathbf{L} = \mathbf{U}\mathbf{D}\mathbf{U}^T$$

La transformée de Fourier sur graphe peut se définir comme le changement de base suivant :

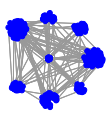
$$\mathbf{y} = \mathbf{U}^T\mathbf{x} \quad \mathcal{O}(n^2)$$

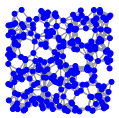
Dans le cas classique (graphe boucle), on a :

$$\mathbf{U} = \mathbf{S}_J \dots \mathbf{S}_1 \quad \mathcal{O}(n \log n)$$



Peut-on généraliser cela à des graphes quelconques ?





FFT sur graphe : Approches

1. **Factorisation :** $\mathbf{U} \approx \hat{\mathbf{U}}_{\mathsf{f}} = \mathbf{S}_J \dots \mathbf{S}_1$.

FFT sur graphe : Approches

1. **Factorisation :** $\mathbf{U} \approx \hat{\mathbf{U}}_{\mathsf{f}} = \mathbf{S}_J \dots \mathbf{S}_1$.

FFT sur graphe : Approches

1. **Factorisation :** $\mathbf{U} \approx \hat{\mathbf{U}}_{\mathsf{f}} = \mathbf{S}_J \dots \mathbf{S}_1$.

- Nécessite de connaître $\mathbf{U}$ (coût $\mathcal{O}(n^3)$)

FFT sur graphe : Approches

1. **Factorisation :** $\mathbf{U} \approx \hat{\mathbf{U}}_{\mathsf{f}} = \mathbf{S}_J \dots \mathbf{S}_1$.

- Nécessite de connaître $\mathbf{U}$ (coût $\mathcal{O}(n^3)$)
- Les FFT approchées obtenues ne sont pas orthogonales

FFT sur graphe : Approches

1. Factorisation : $\mathbf{U} \approx \hat{\mathbf{U}}_{\mathsf{f}} = \mathbf{S}_J \dots \mathbf{S}_1$.

- Nécessite de connaître $\mathbf{U}$ (coût $\mathcal{O}(n^3)$)
- Les FFT approchées obtenues ne sont pas orthogonales

2. Diagonalisation : $\mathbf{L} \approx \underbrace{\mathbf{S}_1 \dots \mathbf{S}_J}_{\hat{\mathbf{U}}_{\mathsf{d}}} \hat{\mathbf{D}} \underbrace{\mathbf{S}_J^T \dots \mathbf{S}_1^T}_{\hat{\mathbf{U}}_{\mathsf{d}}^T}$.

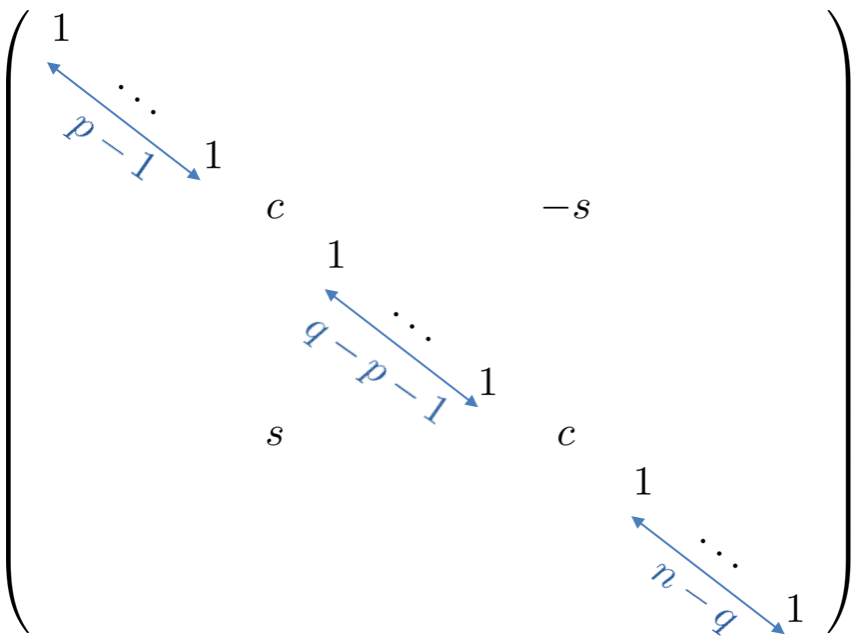
FFT sur graphe : Diagonalisation gloutonne

$$\mathbf{L} \approx \underbrace{\mathbf{S}_1 \dots \mathbf{S}_J}_{\hat{\mathbf{U}}_\mathsf{d}} \hat{\mathbf{D}} \underbrace{\mathbf{S}_J^T \dots \mathbf{S}_1^T}_{\hat{\mathbf{U}}_\mathsf{d}^T}$$

FFT sur graphe : Diagonalisation gloutonne

$$\mathbf{L} \approx \underbrace{\mathbf{S}_1 \dots \mathbf{S}_J}_{\hat{\mathbf{U}}_\mathsf{d}} \hat{\mathbf{D}} \underbrace{\mathbf{S}_J^T \dots \mathbf{S}_1^T}_{\hat{\mathbf{U}}_\mathsf{d}^T}$$

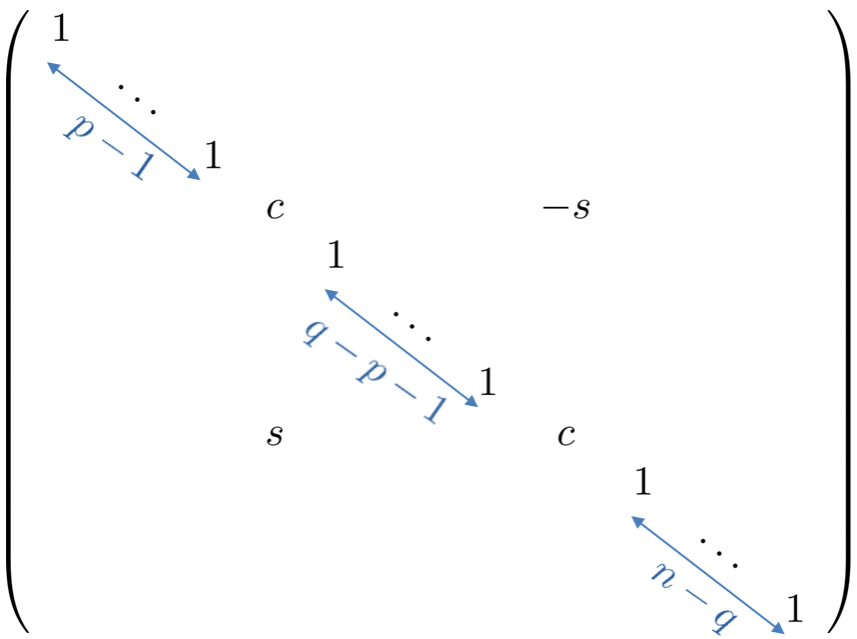
Les facteurs creux $\mathbf{S}_j$ sont des rotations de Givens :



FFT sur graphe : Diagonalisation gloutonne

$$\mathbf{L} \approx \underbrace{\mathbf{S}_1 \dots \mathbf{S}_J}_{\hat{\mathbf{U}}_\mathrm{d}} \hat{\mathbf{D}} \underbrace{\mathbf{S}_J^T \dots \mathbf{S}_1^T}_{\hat{\mathbf{U}}_\mathrm{d}^T}$$

Les facteurs creux $\mathbf{S}_j$ sont des rotations de Givens :



Coût d'application : 4 multiplications et 2 additions.

FFT sur graphe : Comparaison des approches

	Communauté	Capteurs	Erdős-Rényi	Ligne
$\frac{\left\|\mathbf{U}-\hat{\mathbf{U}}\right\|_F}{\|\mathbf{U}\|_F}$	**0.07**|0.82	**0.12**|0.76	**0.31**|1.21	**0.37** |1.00
$\frac{\left\|\hat{\mathbf{U}}^T\mathbf{L}\hat{\mathbf{U}}\right\|_{\text{offdiag}}}{\|\mathbf{L}\|_F}$	0.09|**0.03**	0.16|**0.04**	0.37|**0.09**	0.40|**0.06**

Tableau: Résultats d'approximation pour les deux approches (avec la même complexité, RCG ≈ 3), deux mesures de qualité et divers graphes de taille $n = 128$ (la moyenne sur 10 réalisations est donnée ici).

FFT sur graphe : Filtrage

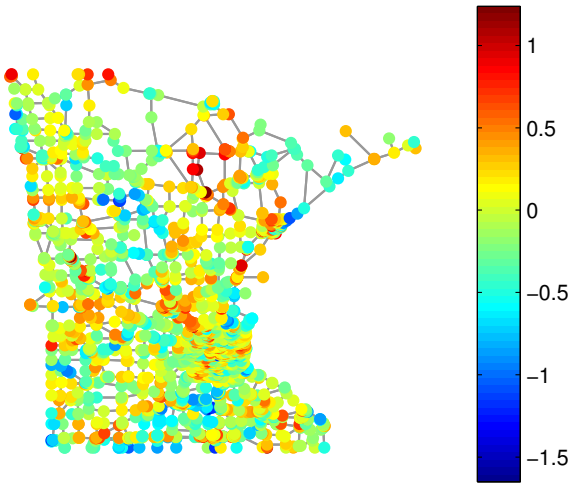
Des signaux bruités sur le graphe "Minnesota" ($n = 2642$) sont filtrés passe-bas avec un filtre exponentiel ($\mathbf{y} = \hat{\mathbf{U}}\mathbf{H}\hat{\mathbf{U}}^T\mathbf{x}$).

	RCG	$\sigma = 0.3$	$\sigma = 0.4$	$\sigma = 0.5$
Bruité	✗	1.82	-0.68	-2.65
Filtré avec $\mathbf{U}$	1	5.17	4.55	3.95
Filtré avec $\hat{\mathbf{U}}_\mathsf{f}$	8	4.70	4.23	3.59
Filtré avec $\hat{\mathbf{U}}_\mathsf{d}$	35	4.97	4.41	3.87

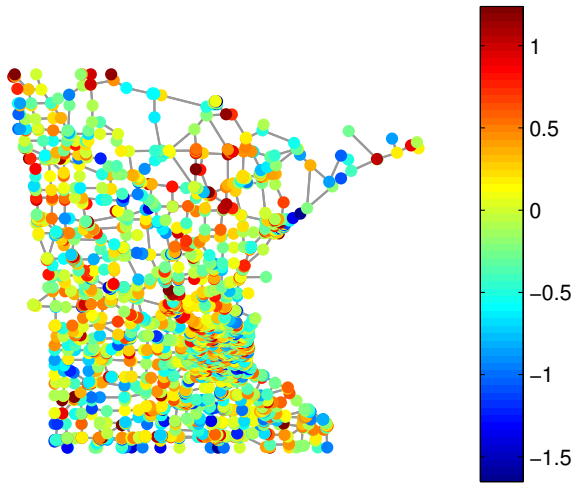
Tableau: Résultats de filtrage, le SNR en dB et en moyenne sur 100 signaux aléatoires pour chaque niveau de bruit est donné.

FFT sur graphe : Filtrage

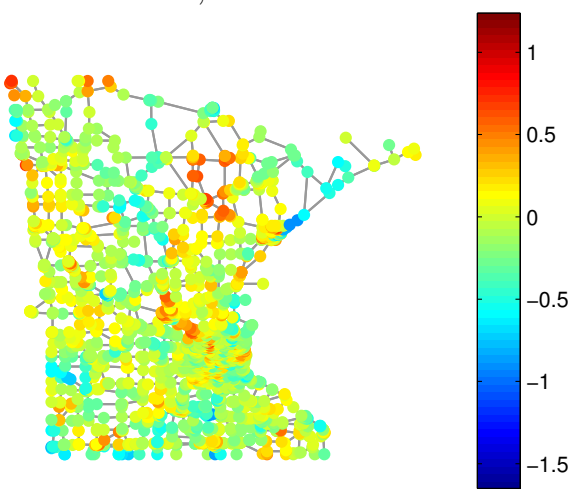
Signal propre



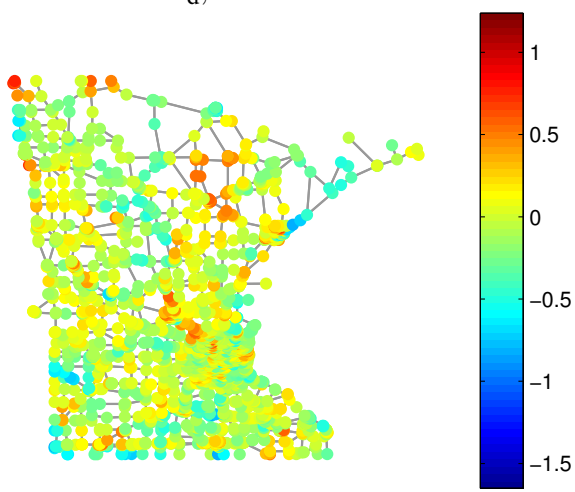
Signal bruité, SNR=-0.84dB



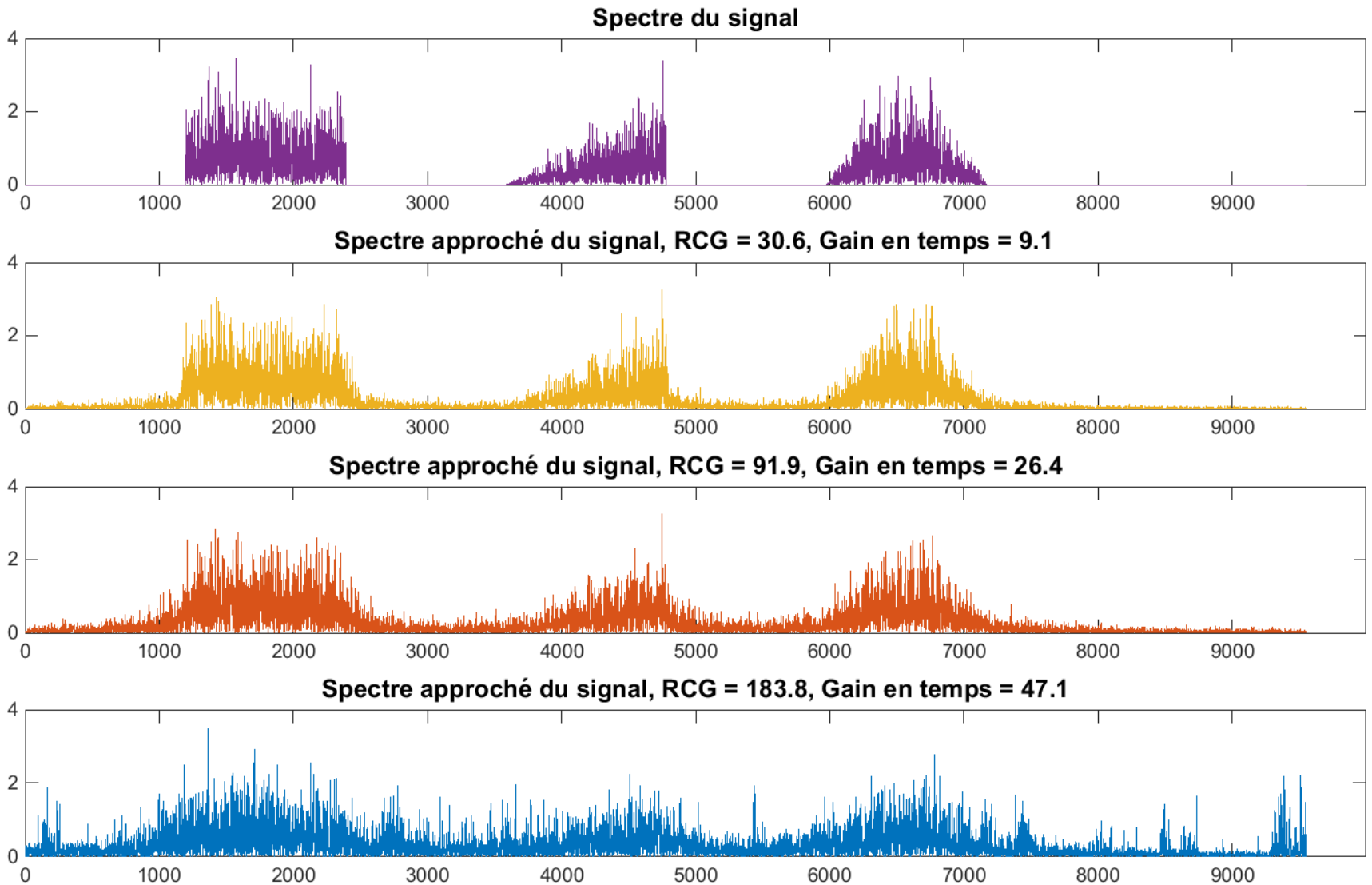
Signal filtré avec $\mathbf{U}$, SNR=4.57dB



Signal filtré avec $\hat{\mathbf{U}}_d$, SNR=4.39dB



FFT sur graphe : Calcul de la transformée de Fourier



Identifiabilité

Question : *Sous quelles conditions peut-on retrouver des facteurs creux* $\mathbf{S}_1, \ldots, \mathbf{S}_J$ *à partir de l'observation de* $\mathbf{A} = \mathbf{S}_J \ldots \mathbf{S}_1$ *?*

Identifiabilité

Question : *Sous quelles conditions peut-on retrouver des facteurs creux* $\mathbf{S}_1, \ldots, \mathbf{S}_J$ *à partir de l'observation de* $\mathbf{A} = \mathbf{S}_J \ldots \mathbf{S}_1$*, leurs supports étant connus*[7] *?*

[7] F. Malgouyres and J. Landsberg, **On the identifiability and stable recovery of deep/multi-layer structured matrix factorization**, *ITW*, 2016.

Identifiabilité

Question : *Sous quelles conditions peut-on retrouver deux facteurs creux* $\mathbf{X}, \mathbf{Y}$ *à partir de l'observation de* $\mathbf{Z} = \mathbf{XY}$*, leurs supports étant connus ?*

Identifiabilité

Question : *Sous quelles conditions peut-on retrouver deux facteurs creux* $\mathbf{X}, \mathbf{Y}$ *à partir de l'observation de* $\mathbf{Z} = \mathbf{XY}$*, leurs supports étant connus ?*

- $\mathbf{Z} = \mathbf{XY} = \sum_{i=1}^{r} \underbrace{\mathbf{x}_i . (\mathbf{y}^i)^T}_{\mathbf{C}_i}$

Identifiabilité

Question : *Sous quelles conditions peut-on retrouver deux facteurs creux* $\mathbf{X}, \mathbf{Y}$ *à partir de l'observation de* $\mathbf{Z} = \mathbf{XY}$*, leurs supports étant connus ?*

- $\mathbf{Z} = \mathbf{XY} = \sum_{i=1}^{r} \underbrace{\mathbf{x}_i . (\mathbf{y}^i)^T}_{\mathbf{C}_i}$

$$\begin{pmatrix} \times & \times & \times & 0 \\ \times & \times & \times & \times \\ \times & \times & \times & \times \\ 0 & \times & \times & \times \end{pmatrix}$$

Identifiabilité

Question : *Sous quelles conditions peut-on retrouver deux facteurs creux* $\mathbf{X}, \mathbf{Y}$ *à partir de l'observation de* $\mathbf{Z} = \mathbf{XY}$*, leurs supports étant connus ?*

- $\mathbf{Z} = \mathbf{XY} = \sum_{i=1}^{r} \underbrace{\mathbf{x}_i . (\mathbf{y}^i)^T}_{\mathbf{C}_i}$

$$\begin{pmatrix} \times & \times & \times & 0 \\ \times & \times & \times & \times \\ \times & \times & \times & \times \\ 0 & \times & \times & \times \end{pmatrix} \qquad \begin{pmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \end{pmatrix}$$

Identifiabilité

Théorème

Soient $\mathbf{F}_1, \ldots, \mathbf{F}_J$ les facteurs correspondant aux étapes de la FFT classique. La résolution exacte du problème d'optimisation

$$\underset{\mathbf{X},\mathbf{Y}}{\text{minimiser}} \quad \tfrac{1}{2} \left\| \mathbf{F}_J \ldots \mathbf{F}_\ell - \mathbf{X}\mathbf{Y} \right\|_F^2 + \delta_{\mathcal{X}}(\mathbf{X}) + \delta_{\mathcal{Y}}(\mathbf{Y}),$$

pour $\ell = 1, \ldots, J-1$ avec les contraintes adéquates permet de retrouver ces facteurs.

Identifiabilité

Théorème

Soient $\mathbf{F}_1, \ldots, \mathbf{F}_J$ les facteurs correspondant aux étapes de la FFT classique. La résolution exacte du problème d'optimisation

$$\underset{\mathbf{X},\mathbf{Y}}{\text{minimiser}} \quad \tfrac{1}{2} \left\| \mathbf{F}_J \ldots \mathbf{F}_\ell - \mathbf{X}\mathbf{Y} \right\|_F^2 + \delta_{\mathcal{X}}(\mathbf{X}) + \delta_{\mathcal{Y}}(\mathbf{Y}),$$

pour $\ell = 1, \ldots, J-1$ avec les contraintes adéquates permet de retrouver ces facteurs.

Le minimum global de la stratégie hiérarchique appelée sur la transformée de Fourier correspond aux facteurs de la FFT.

Sommaire

Problème d'optimisation

- **Entrée :** données i.i.d. $\mathbf{x}_i \sim p$ avec $i \in \{1, \ldots, N\}$

Problème d'optimisation

- **Entrée :** données i.i.d. $\mathbf{x}_i \sim p$ avec $i \in \{1, \ldots, N\}$
- **But (apprentissage) :** trouver J matrices creuses $\mathbf{S}_j$ telles que en moyenne pour $\mathbf{x} \sim p$, $f(\mathbf{x}, \prod_{j=1}^{J} \mathbf{S}_j)$ est faible.

Problème d'optimisation

- **Entrée :** données i.i.d. $\mathbf{x}_i \sim p$ avec $i \in \{1, \dots, N\}$
- **But (apprentissage) :** trouver J matrices creuses $\mathbf{S}_j$ telles que en moyenne pour $\mathbf{x} \sim p$, $f(\mathbf{x}, \prod_{j=1}^{J} \mathbf{S}_j)$ est faible.
- **Approche :**

$$\underset{\mathbf{S}_1,\dots,\mathbf{S}_J}{\text{Minimiser}} \quad \underbrace{\mathbb{E}_{\mathbf{x}}\left[f(\mathbf{x}, \prod_{j=1}^{J} \mathbf{S}_j)\right]}_{\text{Attache aux données}} + \underbrace{\sum_{j=1}^{J} \delta_{\mathcal{S}_j}(\mathbf{S}_j)}_{\text{Parcimonie}}$$

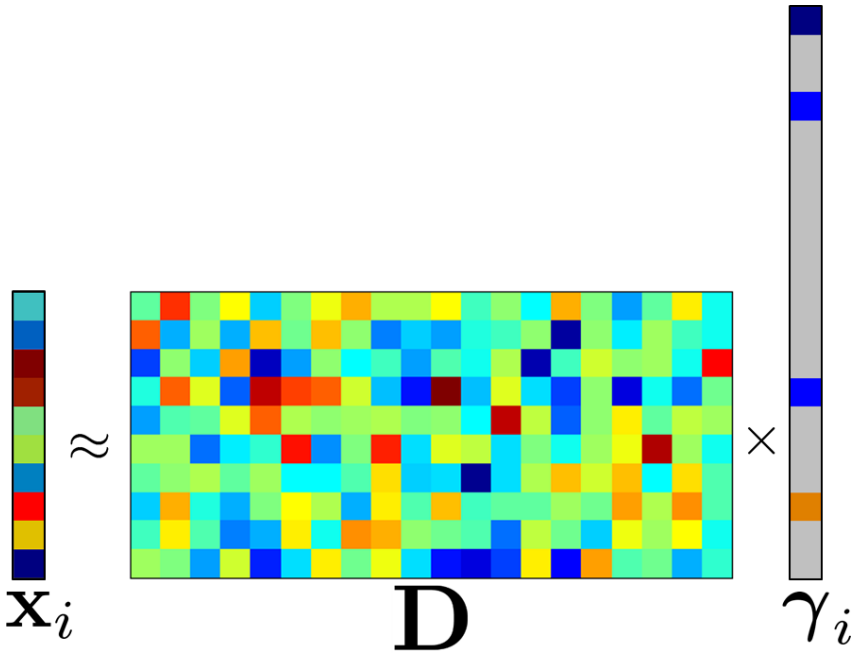
Problème d'optimisation

- **Entrée :** données i.i.d. $\mathbf{x}_i \sim p$ avec $i \in \{1, \dots, N\}$
- **But (apprentissage) :** trouver J matrices creuses $\mathbf{S}_j$ telles que en moyenne pour $\mathbf{x} \sim p$, $f(\mathbf{x}, \prod_{j=1}^{J} \mathbf{S}_j)$ est faible.
- **Approche :**

$$\underset{\mathbf{S}_1,\dots,\mathbf{S}_J}{\text{Minimiser}} \quad \underbrace{\frac{1}{N}\sum_{i=1}^{N} f(\mathbf{x}_i, \prod_{j=1}^{J} \mathbf{S}_j)}_{\text{Attache aux données}} + \underbrace{\sum_{j=1}^{J} \delta_{\mathcal{S}_j}(\mathbf{S}_j)}_{\text{Parcimonie}}$$

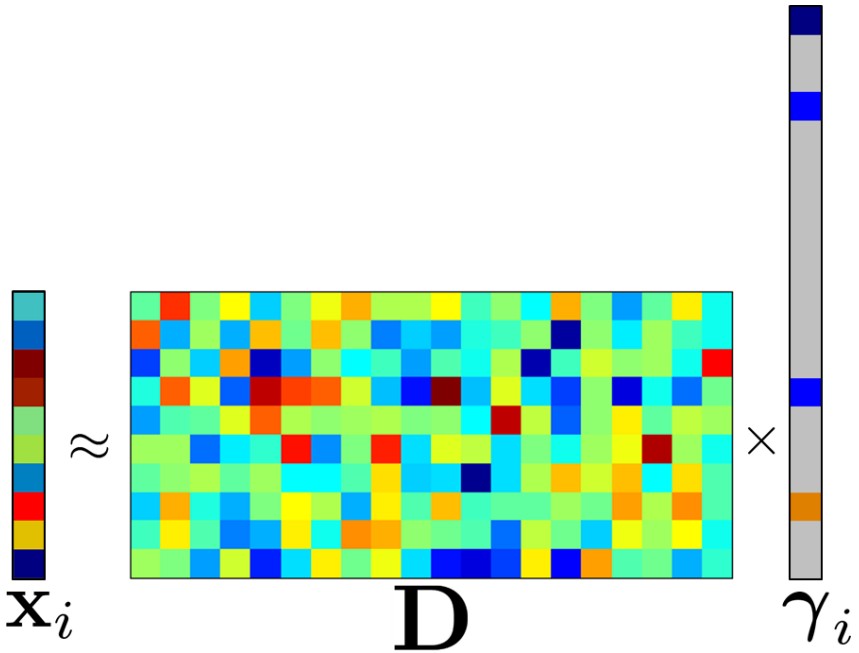
Ce problème est compatible avec PALM et la stratégie hiérarchique.

Apprentissage de dictionnaire



$$\mathbf{X} \approx \mathbf{D}\Gamma$$

Apprentissage de dictionnaire



$$\mathbf{X} \approx \underbrace{\mathbf{D}\boldsymbol{\Gamma}}_{J} \prod_{j=1} \mathbf{S}_j$$

Apprentissage de dictionnaire : Expérience

Débruitage d'image :

- $\mathbf{X} \in \mathbb{R}^{64 \times 10000}$: morceaux d'une image bruitée.
- Le dictionnaire appris est utilisé pour débruiter l'image entière.

Apprentissage de dictionnaire : Expérience

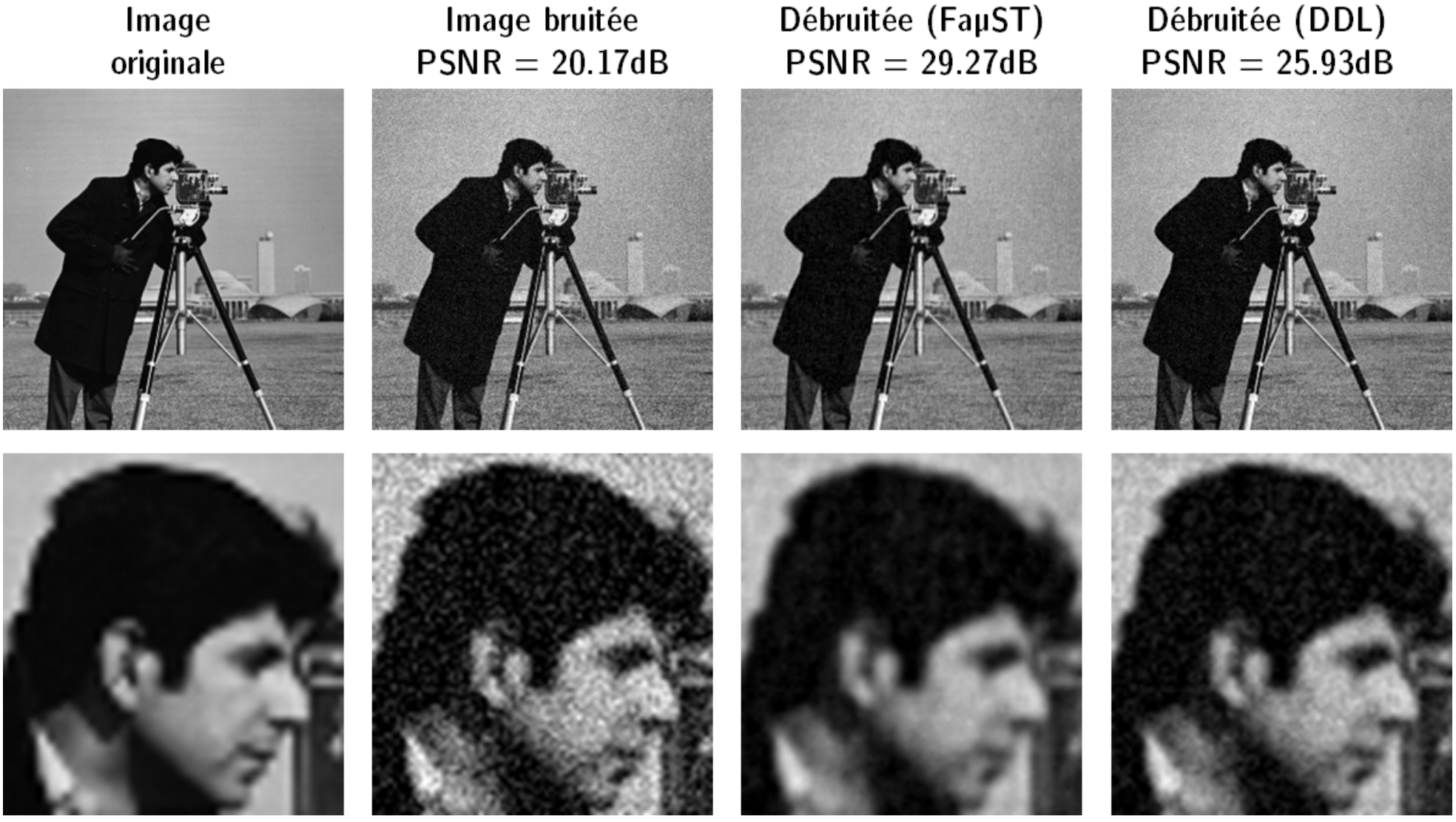
Débruitage d'image :

- $\mathbf{X} \in \mathbb{R}^{64 \times 10000}$: morceaux d'une image bruitée.
- Le dictionnaire appris est utilisé pour débruiter l'image entière.

Classe de dictionnaire $\mathcal{D}$:

- Dictionnaire dense (DDL)
- Dictionnaire efficient (FAμST : $\mathbf{D} = \prod_{j=1}^{J} \mathbf{S}_j$)

Apprentissage de dictionnaire : Exemple



Propriétés de généralisation

Question : *Comment se comporte le dictionnaire estimé sur de nouvelles données ?*

Propriétés de généralisation

Question : *Comment se comporte le dictionnaire estimé sur de nouvelles données ?*

On répond en bornant $\sup_{\mathbf{D}\in\mathcal{D}} \left| \frac{1}{N} \sum_{i=1}^{N} f(\mathbf{x}_i, \mathbf{D}) - \mathbb{E}_{\mathbf{x}\sim p} f(\mathbf{x}, \mathbf{D}) \right|$.

Propriétés de généralisation

Question : *Comment se comporte le dictionnaire estimé sur de nouvelles données ?*

Théorème (Gribonval et al.)

Dans le cas de l'apprentissage de dictionnaire[7]*,*

$$\sup_{\mathbf{D}\in\mathcal{D}} \left| \frac{1}{N} \sum_{i=1}^{N} f(\mathbf{x}_i, \mathbf{D}) - \mathbb{E}_{\mathbf{x}\sim p} f(\mathbf{x}, \mathbf{D}) \right| \leq \eta(N, \mathcal{D}),$$

avec $\eta = \mathcal{O}\left(\sqrt{d(\mathcal{D})\frac{\log N}{N}}\right)$.

[7] R. Gribonval et al., **Sample Complexity of Dictionary Learning and other Matrix Factorizations.** *IEEE Trans. Inf. Theory.* 2015.

Propriétés de généralisation

Lemme (Gribonval et al.)

Pour des dictionnaires denses :

$$d(\mathcal{D}) = mn.$$

$$\Rightarrow \eta_{DDL} = \mathcal{O}\left(\sqrt{mn\frac{\log N}{N}}\right)$$

Propriétés de généralisation

Lemme (Gribonval et al.)

Pour des dictionnaires denses :

$$d(\mathcal{D}) = mn.$$

$$\Rightarrow \eta_{DDL} = \mathcal{O}\left(\sqrt{mn\frac{\log N}{N}}\right)$$

Lemme

Pour des dictionnaires efficients $\mathbf{D} = \prod_{j=1}^{J} \mathbf{S}_j$ *:*

$$d(\mathcal{D}) = \sum_{j=1}^{J} \|\mathbf{S}_j\|_0 .$$

$$\Rightarrow \eta_{FA\mu ST} = \mathcal{O}\left(\sqrt{\sum_{j=1}^{J} \|\mathbf{S}_j\|_0 \frac{\log N}{N}}\right)$$

Propriétés de généralisation

Lemme (Gribonval et al.)

Pour des dictionnaires denses :

$$d(\mathcal{D}) = mn.$$

$$\Rightarrow \eta_{DDL} = \mathcal{O}\left(\sqrt{mn\frac{\log N}{N}}\right)$$

Lemme

Pour des dictionnaires efficients $\mathbf{D} = \prod_{j=1}^{J} \mathbf{S}_j$ *:*

$$d(\mathcal{D}) = \sum_{j=1}^{J} \|\mathbf{S}_j\|_0 .$$

$$\Rightarrow \eta_{FA\mu ST} = \mathcal{O}\left(\sqrt{\sum_{j=1}^{J} \|\mathbf{S}_j\|_0 \frac{\log N}{N}}\right)$$

$$\eta_{\mathsf{FA}\mu\mathsf{ST}} = \frac{1}{\sqrt{\mathsf{RCG}}}\eta_{\mathsf{DDL}}$$

Application aux nouveaux modèles parcimonieux

De nouveaux modèles parcimonieux **flexibles** [8] à **encodeur explicite** [9] [10] ont été proposés.

[8] J. Mairal et al., **Task-driven dictionary learning.** *IEEE TPAMI.* 2012.

[9] P. Sprechmann et al., **Learning efficient sparse and low rank models.** *IEEE TPAMI.* 2015.

[10] A. Fawzi et al., **Dictionary learning for fast classification based on soft-thresholding.** *IJCV.* 2014.

Application aux nouveaux modèles parcimonieux

De nouveaux modèles parcimonieux **flexibles** [8] à **encodeur explicite** [9] [10] ont été proposés.

Théorème

Pour ces modèles, avec une classe d'hypothèse $\mathcal{H}$ compacte et un encodeur Lipschitz de module borné par L_e, on a

$$\sup_{\mathbf{H}\in\mathcal{H}} \left| \frac{1}{N} \sum_{i=1}^{N} f_{\mathbf{H}}(\mathbf{z}_i) - \mathbb{E}_{\mathbf{z}\sim\mu} f_{\mathbf{H}}(\mathbf{z}) \right| \leq \eta(N, \mathcal{H}),$$

avec $\eta = \mathcal{O}\left(\sqrt{d(\mathcal{H}) \log(L_e) \frac{\log N}{N}} \right)$.

[8] J. Mairal et al., **Task-driven dictionary learning.** *IEEE TPAMI.* 2012.

[9] P. Sprechmann et al., **Learning efficient sparse and low rank models.** *IEEE TPAMI.* 2015.

[10] A. Fawzi et al., **Dictionary learning for fast classification based on soft-thresholding.** *IJCV.* 2014.

Sommaire

Résumé des contributions

Les travaux effectués au cours de cette thèse ont mené à :

Résumé des contributions

Les travaux effectués au cours de cette thèse ont mené à :

- L'introduction du **concept** de matrice efficiente

Résumé des contributions

Les travaux effectués au cours de cette thèse ont mené à :

- L'introduction du **concept** de matrice efficiente
- Des **algorithmes** généraux pour approcher et estimer avec des matrices efficientes

Résumé des contributions

Les travaux effectués au cours de cette thèse ont mené à :

- L'introduction du **concept** de matrice efficiente
- Des **algorithmes** généraux pour approcher et estimer avec des matrices efficientes
- Des résultats prometteurs dans plusieurs **applications**
  - Problèmes inverses
  - FFT sur graphe
  - Apprentissage de dictionnaire

Résumé des contributions

Les travaux effectués au cours de cette thèse ont mené à :

- L'introduction du **concept** de matrice efficiente
- Des **algorithmes** généraux pour approcher et estimer avec des matrices efficientes
- Des résultats prometteurs dans plusieurs **applications**
  - Problèmes inverses
  - FFT sur graphe
  - Apprentissage de dictionnaire
- Des éclairages **théoriques** sur les matrices efficientes
  - Identifiabilité
  - Généralisation

Perspectives

- Réduction du coût de factorisation
  - factorisation à l'aide de données d'entraînement
  - stratégies gloutonnes

Perspectives

- Réduction du coût de factorisation
  - factorisation à l'aide de données d'entraînement
  - stratégies gloutonnes
- Réduction de l'écart entre RCG et gain pratique
  - facteurs structurés
  - implémentation optimisée

Perspectives

- Réduction du coût de factorisation
  - factorisation à l'aide de données d'entraînement
  - stratégies gloutonnes
- Réduction de l'écart entre RCG et gain pratique
  - facteurs structurés
  - implémentation optimisée
- Traitement du signal sur graphe (approfondissement)

Perspectives

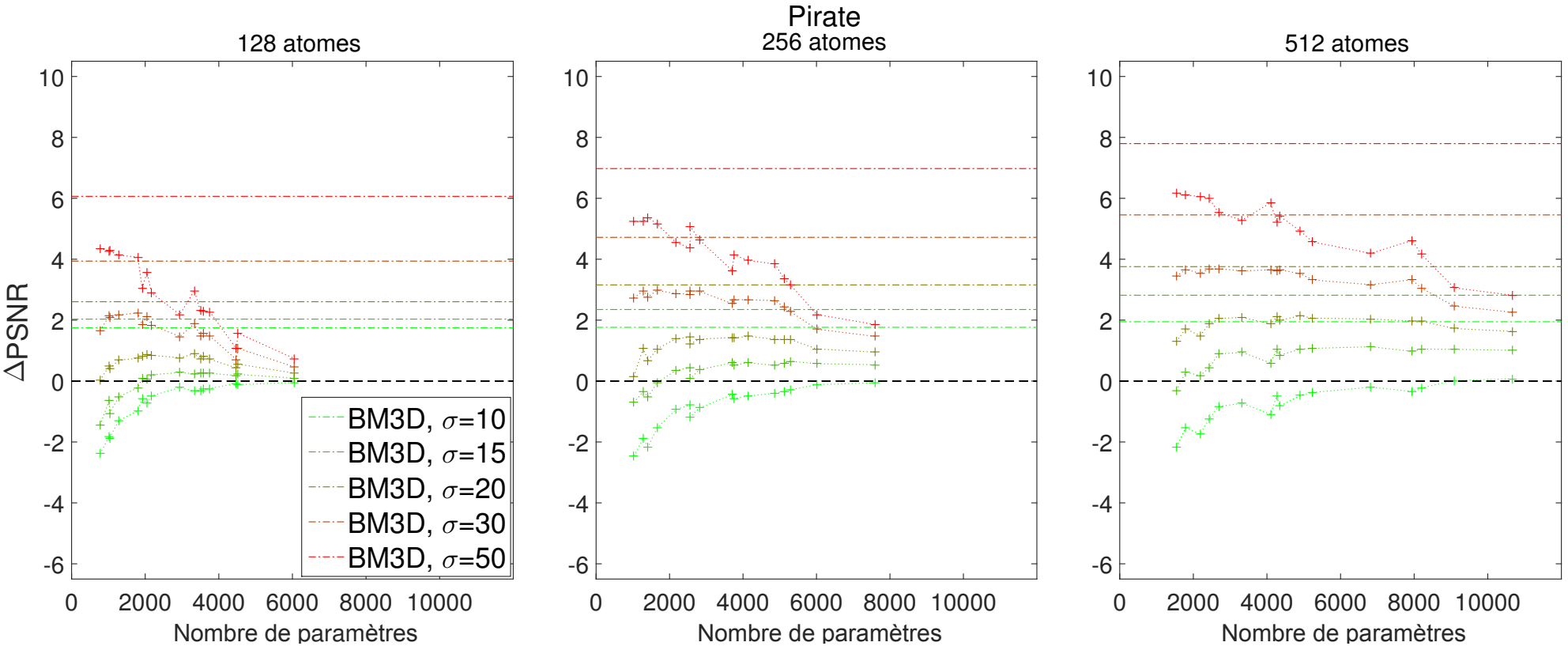
- Réduction du coût de factorisation
  - factorisation à l'aide de données d'entraînement
  - stratégies gloutonnes
- Réduction de l'écart entre RCG et gain pratique
  - facteurs structurés
  - implémentation optimisée
- Traitement du signal sur graphe (approfondissement)
- Nouveaux modèles parcimonieux et apprentissage profond

Perspectives

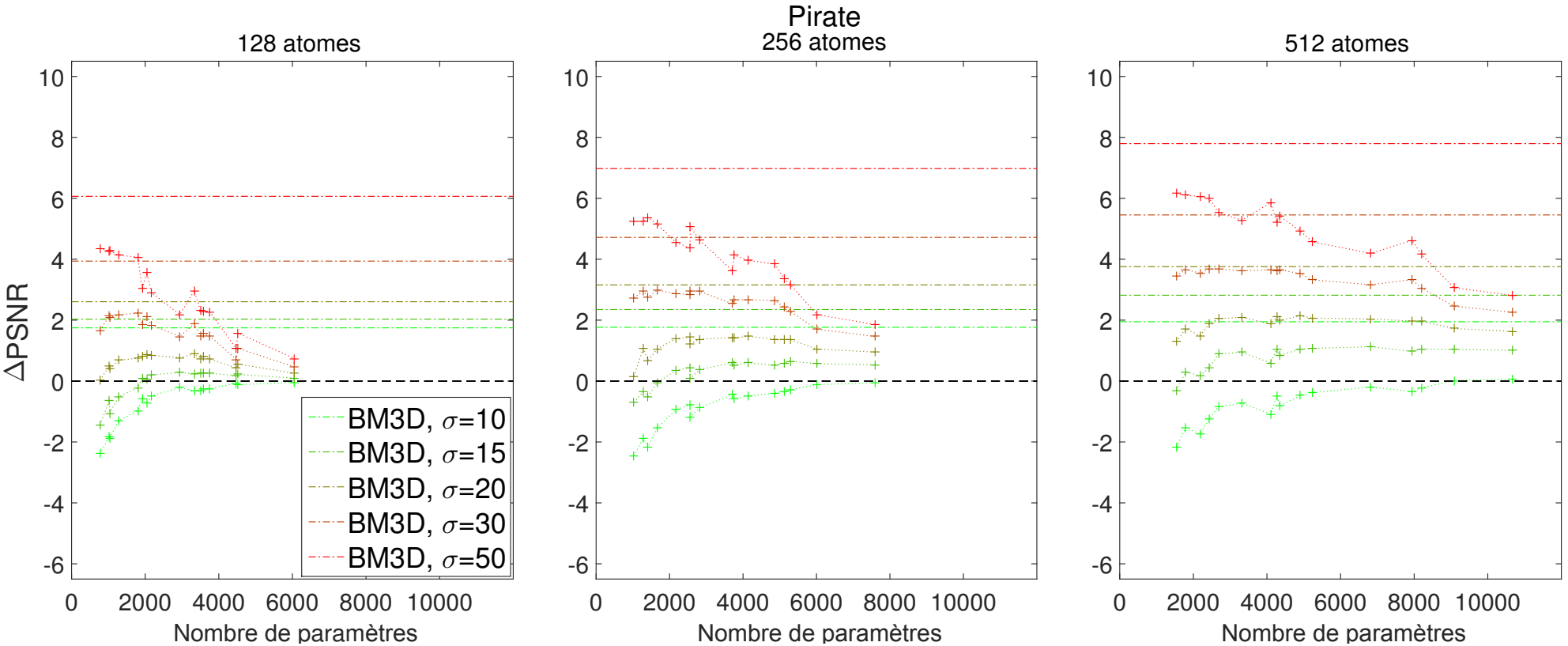
- Réduction du coût de factorisation
  - factorisation à l'aide de données d'entraînement
  - stratégies gloutonnes
- Réduction de l'écart entre RCG et gain pratique
  - facteurs structurés
  - implémentation optimisée
- Traitement du signal sur graphe (approfondissement)
- Nouveaux modèles parcimonieux et apprentissage profond
- Échantillonnage compressé

Questions ?

Apprentissage de dictionnaire : Résultats de débruitage

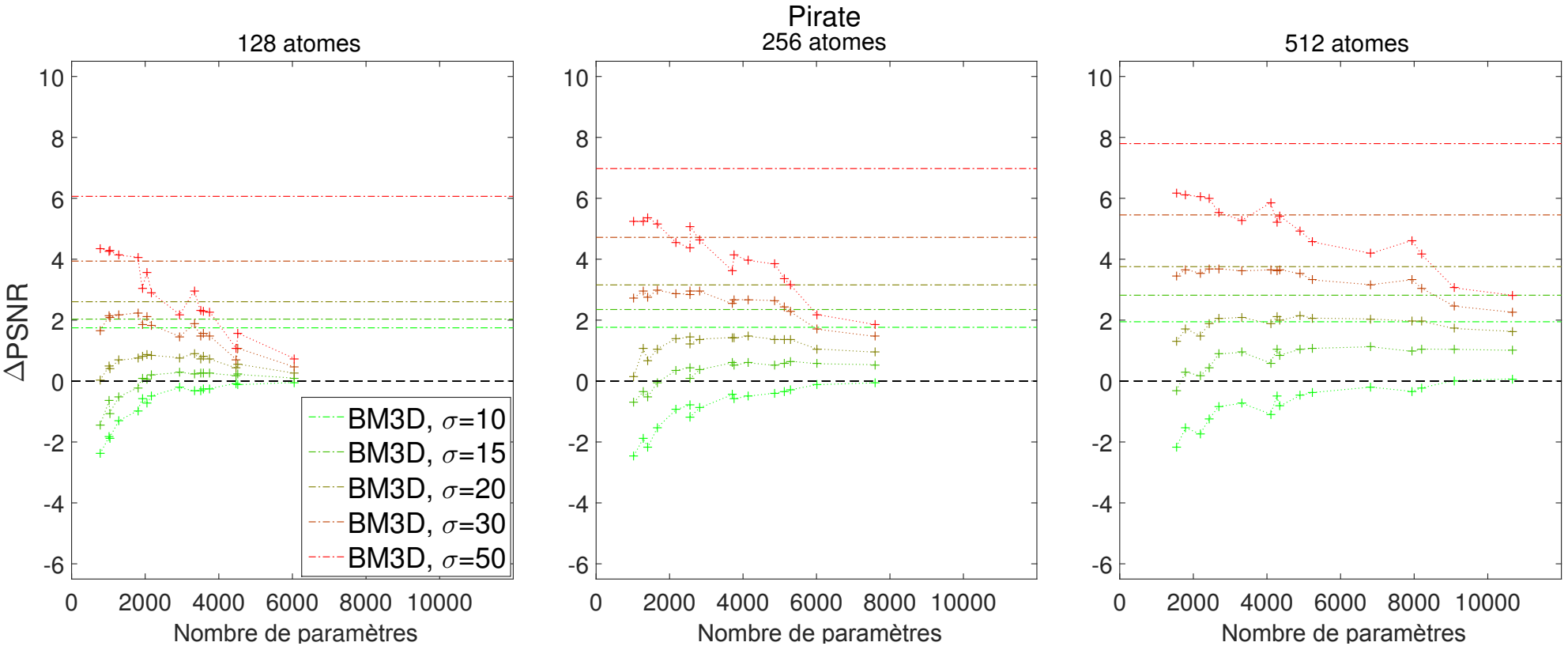


Apprentissage de dictionnaire : Résultats de débruitage



- FAμST meilleur que DDL à niveau de bruit élevé.

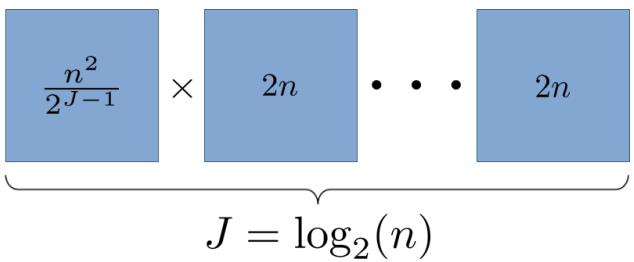
Apprentissage de dictionnaire : Résultats de débruitage



- FAμST meilleur que DDL à niveau de bruit élevé.
- Pour un niveau de bruit élevé, les FAμSTs les plus creuses sont les meilleures.

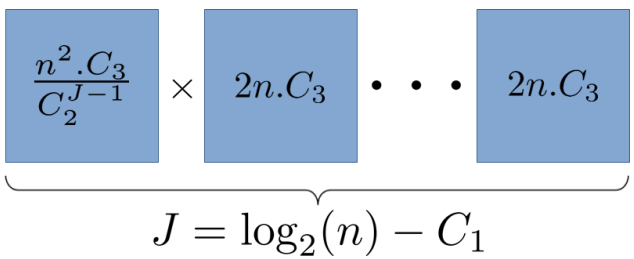
FFT sur graphe : Factorisation

Configuration de factorisation :



FFT sur graphe : Factorisation

Configuration de factorisation :



PALM convergence conditions

The following conditions are sufficient (not necessary) to ensure that each bounded sequence generated by PALM converges to a stationary point of its objective :

1. H is smooth.
2. The $\mathcal{E}_j$s are semi-algebraic sets.
3. $\nabla_{\mathbf{x}_j} H$ is globally Lipschitz for all j, with Lipschitz moduli $L_j(\mathbf{x}_1 \ldots \mathbf{x}_{j-1}, \mathbf{x}_{j+1} \ldots \mathbf{x}_N)$.
4. $\forall i,\ c_j^i > L_j(\mathbf{x}_1^{i+1} \ldots \mathbf{x}_{j-1}^{i+1}, \mathbf{x}_{j+1}^i \ldots \mathbf{x}_N^i)$ (the inequality need not be strict for convex f_j).

The palm4MSA algorithm

Algorithme : PALM for Multi-layer Sparse Approximations

Input: Matrix $\mathbf{X}$, desired number of factors J, constraint sets $\mathcal{E}_j$, $j \in \{1 \dots J\}$ and a stopping criterion.

1: **for** $i = 0$ to $N_{iter} - 1$ **do**
2: **for** $j = 1$ to J **do**
3: Set $c_j^i > (\lambda^i)^2 \|\mathbf{R}\|_2^2 . \|\mathbf{L}\|_2^2$
4: $\mathbf{S}_j^{i+1} \leftarrow P_{\mathcal{E}_j}\left(\mathbf{S}_j^i - \frac{1}{c_j^i}\lambda \mathbf{L}^T(\lambda \mathbf{L}\mathbf{S}_j^i\mathbf{R} - \mathbf{X})\mathbf{R}^T\right)$
5: **end for**
6: $\lambda^{i+1} \leftarrow \frac{\mathrm{Tr}(\mathbf{X}^T\hat{\mathbf{X}})}{\mathrm{Tr}(\hat{\mathbf{X}}^T\hat{\mathbf{X}})}$
7: **end for**

Output: $\lambda^{N_{iter}}, \{\mathbf{S}_k^{N_{iter}}\}_{k=1}^J = \texttt{palm4MSA}(\mathbf{X}, J, \{\mathcal{E}_j\}_{j=1}^J)$

Proposition. Each bounded sequence generated by palm4MSA converges to a stationary point of the objective.

Hierarchical factorization algorithm

Algorithme : Hierarchical factorization

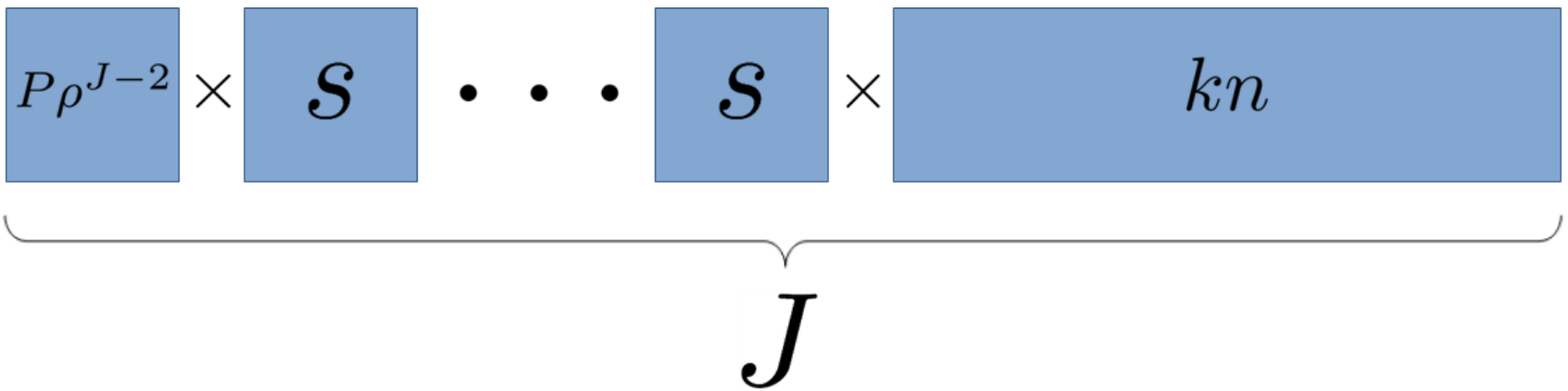
Input: Matrix $\mathbf{X}$, desired number of factors J and the constraint sets $\mathcal{E}_k,\ k \in \{1 \dots J-1\}$ and $\tilde{\mathcal{E}}_k,\ k \in \{1 \dots J-1\}$.

1: $\mathbf{R} \leftarrow \mathbf{X}$
2: **for** $k = 1$ to $J-1$ **do**
3: $\quad \lambda', \{\mathbf{T}_1, \mathbf{T}_2\} = \texttt{palm4MSA}(\mathbf{R},\ 2,\ \{\mathcal{E}_k, \tilde{\mathcal{E}}_k\})$
4: $\quad \mathbf{S}_k \leftarrow \lambda' \mathbf{T}_1$ and $\mathbf{R} \leftarrow \mathbf{T}_2$
5: $\quad \lambda, \left\{\{\mathbf{S}_j\}_{j=1}^{k}, \mathbf{R}\right\} = \texttt{palm4MSA}(\mathbf{X},\ k+1,\ \left\{\{\mathcal{E}_j\}_{j=1}^{k}, \tilde{\mathcal{E}}_k\right\})$
6: **end for**
7: $\mathbf{S}_J \leftarrow \mathbf{R}$

Output: $\lambda, \{\mathbf{S}_k\}_{k=1}^{J}$.

Inverse problems : Factorization of $\mathbf{M}$

Objective : Factorize $\mathbf{M}$ in order to make complexity savings.



What complexity/accuracy trade-offs are achievable ?

Dictionary learning : Algorithm

Algorithme : Hierarchical factorization for dictionary learning

Input: Data matrix $\mathbf{Y}$; Dictionary $\mathbf{D}$; Coefficients $\mathbf{\Gamma}$; desired number of factors J ; constraint sets $\mathcal{E}_k$ and $\tilde{\mathcal{E}}_k$, $k \in \{1 \dots J-1\}$.

1: $\mathbf{T}_0 \leftarrow \mathbf{D}$
2: **for** $k = 1$ to $J-1$ **do**
3: Factorize the residual $\mathbf{T}_{k-1}$ into 2 factors : $\lambda', \{\mathbf{F}_2, \mathbf{F}_1\} = \texttt{palm4MSA}(\mathbf{T}_{k-1}, 2, \{\tilde{\mathcal{E}}_k, \mathcal{E}_k\}, \dots)$
4: $\mathbf{T}_k \leftarrow \lambda' \mathbf{F}_2$ and $\mathbf{S}_k \leftarrow \mathbf{F}_1$
5: Global optimization using `palm4MSA`
6: Coefficients update : $\mathbf{\Gamma} = \texttt{sparseCoding}(\mathbf{Y}, \mathbf{T}_k \prod_{j=1}^{k} \mathbf{S}_j)$
7: **end for**
8: $\mathbf{S}_J \leftarrow \mathbf{T}_{J-1}$

Output: The estimated factorization : $\lambda, \{\mathbf{S}_j\}_{j=1}^{J}$, $\mathbf{\Gamma}$.

Dictionary learning : Algorithm

Algorithme : Hierarchical factorization for dictionary learning

Input: Data matrix $\mathbf{Y}$; Dictionary $\mathbf{D}$; Coefficients $\mathbf{\Gamma}$; desired number of factors J ; constraint sets $\mathcal{E}_k$ and $\tilde{\mathcal{E}}_k$, $k \in \{1 \dots J-1\}$.

1: $\mathbf{T}_0 \leftarrow \mathbf{D}$
2: **for** $k = 1$ to $J-1$ **do**
3: Factorize the residual $\mathbf{T}_{k-1}$ into 2 factors : $\lambda', \{\mathbf{F}_2, \mathbf{F}_1\} = \texttt{palm4MSA}(\mathbf{T}_{k-1}, 2, \{\tilde{\mathcal{E}}_k, \mathcal{E}_k\}, \dots)$
4: $\mathbf{T}_k \leftarrow \lambda' \mathbf{F}_2$ and $\mathbf{S}_k \leftarrow \mathbf{F}_1$
5: Global optimization using `palm4MSA`
6: Coefficients update : $\mathbf{\Gamma} = \texttt{sparseCoding}(\mathbf{Y}, \mathbf{T}_k \prod_{j=1}^{k} \mathbf{S}_j)$
7: **end for**
8: $\mathbf{S}_J \leftarrow \mathbf{T}_{J-1}$

Output: The estimated factorization : $\lambda, \{\mathbf{S}_j\}_{j=1}^{J}$, $\mathbf{\Gamma}$.

Dictionary learning : Algorithm

Algorithme : Hierarchical factorization for dictionary learning

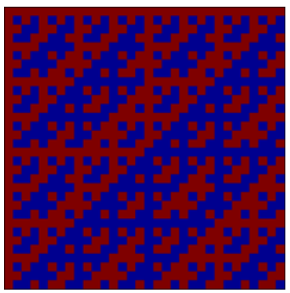
Input: Data matrix $\mathbf{Y}$; Dictionary $\mathbf{D}$; Coefficients $\mathbf{\Gamma}$; desired number of factors J ; constraint sets $\mathcal{E}_k$ and $\tilde{\mathcal{E}}_k$, $k \in \{1 \dots J-1\}$.

1: $\mathbf{T}_0 \leftarrow \mathbf{D}$
2: **for** $k = 1$ to $J-1$ **do**
3: Factorize the residual $\mathbf{T}_{k-1}$ into 2 factors : $\lambda', \{\mathbf{F}_2, \mathbf{F}_1\} = \texttt{palm4MSA}(\mathbf{T}_{k-1}, 2, \{\tilde{\mathcal{E}}_k, \mathcal{E}_k\}, \dots)$
4: $\mathbf{T}_k \leftarrow \lambda' \mathbf{F}_2$ and $\mathbf{S}_k \leftarrow \mathbf{F}_1$
5: Global optimization using `palm4MSA`
6: Coefficients update : $\mathbf{\Gamma} = \texttt{sparseCoding}(\mathbf{Y}, \mathbf{T}_k \prod_{j=1}^{k} \mathbf{S}_j)$
7: **end for**
8: $\mathbf{S}_J \leftarrow \mathbf{T}_{J-1}$

Output: The estimated factorization : $\lambda, \{\mathbf{S}_j\}_{j=1}^{J}$, $\mathbf{\Gamma}$.

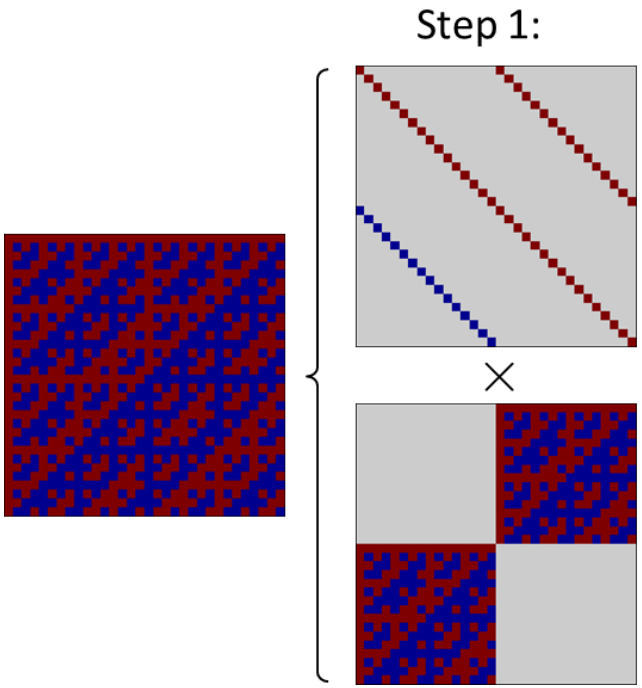
Fast transform retrieval example : Hadamard transform

The hierarchical factorization allows to retrieve the fast implementation of the Hadamard transform of size n, in running time $\mathcal{O}(n^2)$:



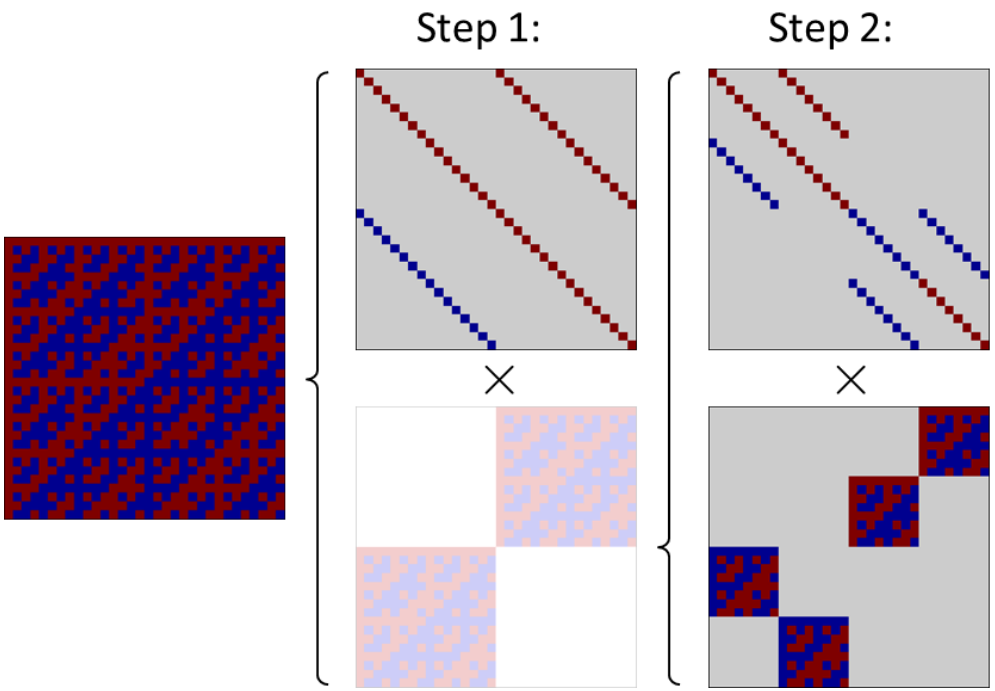
Fast transform retrieval example : Hadamard transform

The hierarchical factorization allows to retrieve the fast implementation of the Hadamard transform of size n, in running time $\mathcal{O}(n^2)$:



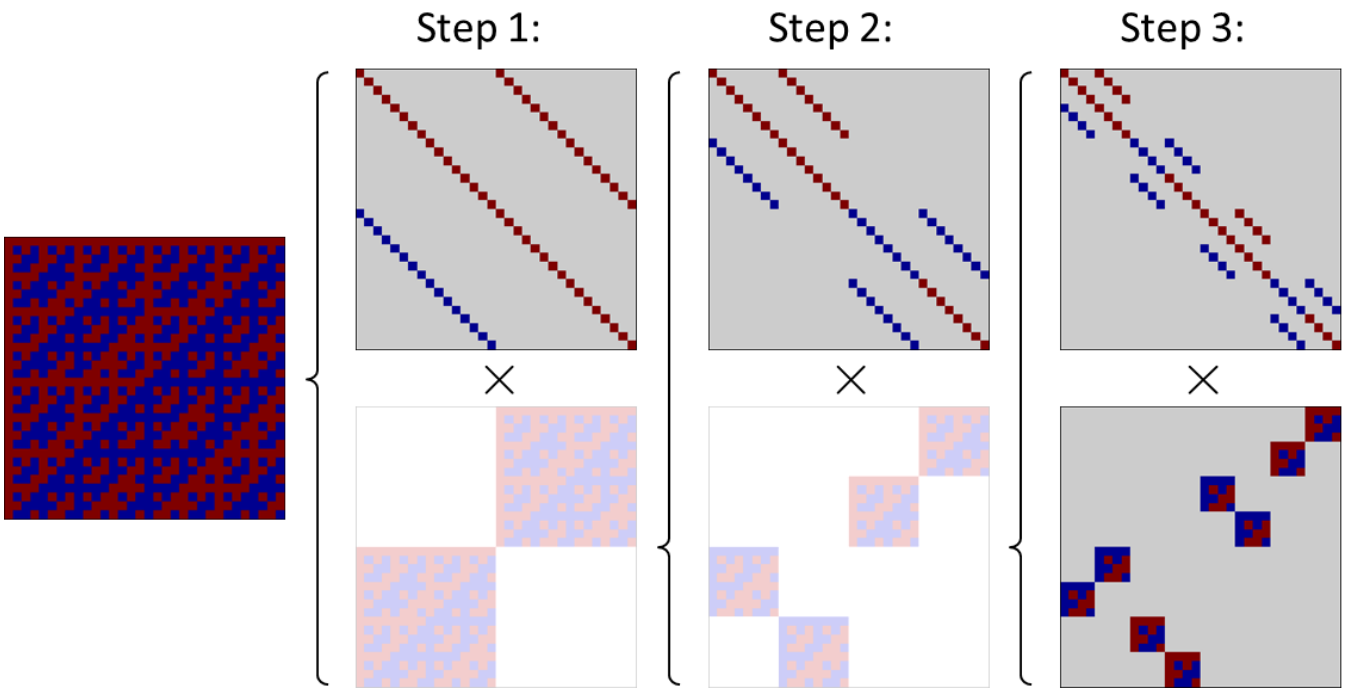
Fast transform retrieval example : Hadamard transform

The hierarchical factorization allows to retrieve the fast implementation of the Hadamard transform of size n, in running time $\mathcal{O}(n^2)$:



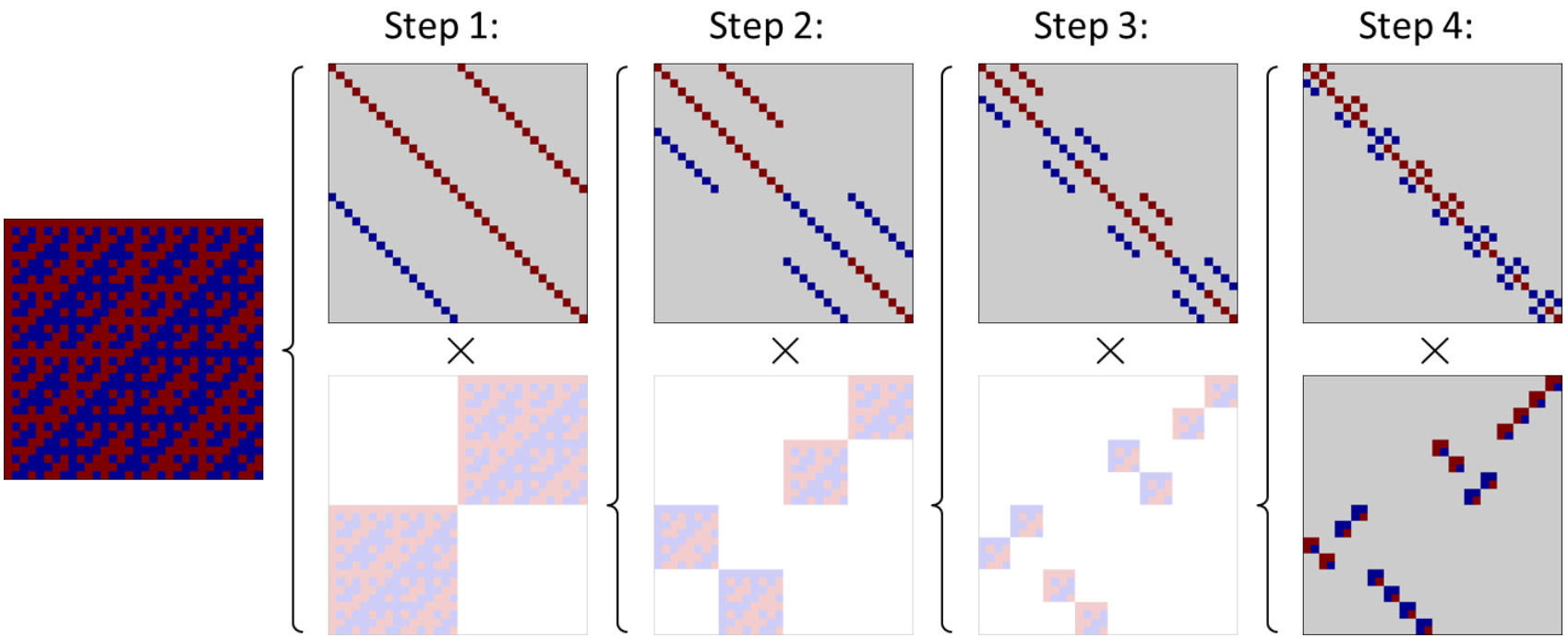
Fast transform retrieval example : Hadamard transform

The hierarchical factorization allows to retrieve the fast implementation of the Hadamard transform of size n, in running time $\mathcal{O}(n^2)$:



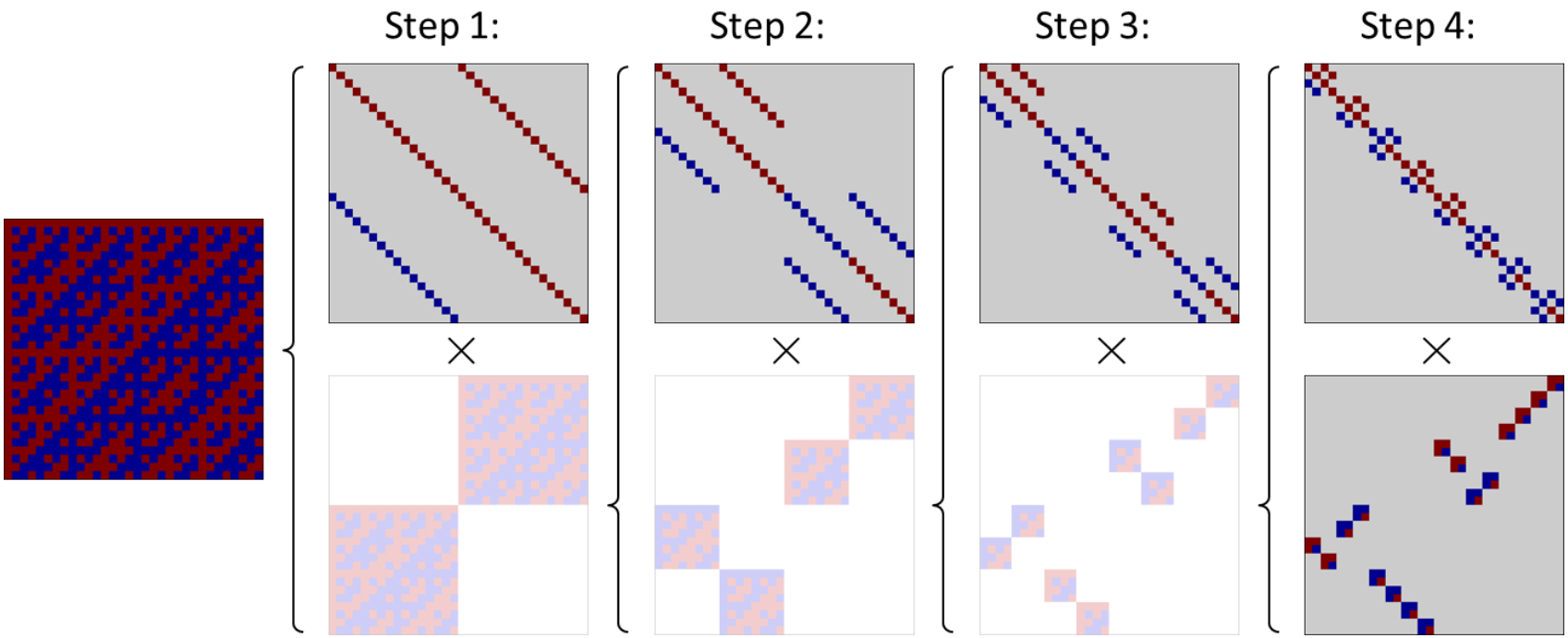
Fast transform retrieval example : Hadamard transform

The hierarchical factorization allows to retrieve the fast implementation of the Hadamard transform of size n, in running time $\mathcal{O}(n^2)$:



Fast transform retrieval example : Hadamard transform

The hierarchical factorization allows to retrieve the fast implementation of the Hadamard transform of size n, in running time $\mathcal{O}(n^2)$:



This factorization is as good as the reference.