Toward Fast Transform Learning

Olivier Chabiron[1], François Malgouyres[2], Jean-Yves Tourneret[1], Nicolas Dobigeon[1]

[1] Institut de Recherche en Informatique de Toulouse (IRIT)
[2] Institut de Mathématiques de Toulouse (IMT)

This work is supported by the CIMI Excellence Laboratory

May 11, 2014

Introduction to sparse representation

Notations

Objects u live in $\mathbb{R}^{\mathcal{P}}$ where $\mathcal{P}$ is a set of pixels (such as $\{1, \ldots, N\}^2$).

In image processing, many problems are underdetermined. For example, in dictionary learning, we want to solve

$$\min \|\alpha\|_* \quad \text{subject to} \quad \|D\alpha - u\|_2 \leq \tau$$

Introduction to sparse representation

Notations

Objects u live in $\mathbb{R}^{\mathcal{P}}$ where $\mathcal{P}$ is a set of pixels (such as $\{1,\dots,N\}^2$).

In image processing, many problems are underdetermined. For example, in dictionary learning, we want to solve

$$\min \|\alpha\|_* \quad \text{subject to} \quad \|D\alpha - u\|_2 \leq \tau$$

Principle of sparse representation/approximation

For many applications, $\|.\|_*$ *should be* $\|.\|_0$.

$$\|\alpha\|_0 = \#\{j \,;\, \alpha_j \neq 0\}$$

Introduction to sparse representation

Notations

Objects u live in $\mathbb{R}^{\mathcal{P}}$ where $\mathcal{P}$ is a set of pixels (such as $\{1, \dots, N\}^2$).

In image processing, many problems are underdetermined. For example, in dictionary learning, we want to solve

$$\min \|\alpha\|_* \quad \text{subject to} \quad \|D\alpha - u\|_2 \leq \tau$$

Principle of sparse representation/approximation

For many applications, $\|.\|_*$ *should be* $\|.\|_0$.

$$\|\alpha\|_0 = \#\{j \,;\, \alpha_j \neq 0\}$$

Issue

The sparse representation problem is (in general) NP-hard. However, successful algorithms exist when the columns of D are almost orthogonal.

Dictionary learning

Choosing a dictionary (Fourier, wavelets,...)

- + fast transform
- - limited sparsity

Learning the dictionary on the data

- - no fast transform
- + better sparsity

The DL problem

Learn an efficient representation frame for an image class, solving

$$\text{argmin}_{D,\alpha} \sum_u \left(\mu \|D\alpha - u\|_2^2 + \|\alpha\|_*\right)$$

DL problems are often resolved in two steps
$\text{argmin}_\alpha \longrightarrow$ Sparse coding stage,
$\text{argmin}_D \longrightarrow$ Dictionary update stage.

Motivations (1)

$$\#\mathcal{P} \text{ image size } \left\{ \underbrace{\begin{pmatrix} \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & D & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots \end{pmatrix}}_{\#D \text{ number of atoms}} \right. \left. \begin{pmatrix} \\ \alpha \\ \\ \end{pmatrix} \right\} \#D = \left. \begin{pmatrix} \\ u \\ \\ \end{pmatrix} \right\} \#\mathcal{P}$$

Usually, $\#D \gg \#\mathcal{P}$.

Computing $D\alpha$ costs $O(\#D\#\mathcal{P}) > O(\#\mathcal{P}^2)$ operations.

Computing sparse codes is very expensive.

Storing D is very expensive.

Motivations (2)

Our objectives:

- Define a fast transform to compute $D\alpha$.

Motivations (2)

Our objectives:

- Define a fast transform to compute $D\alpha$.
- Ensure a fast update so that larger atoms can be learned.

Model

Model for a dictionary update with a single atom $H \in \mathbb{R}^{\mathcal{P}}$.
How to include every possible translation of H ?

$$\sum_{p' \in \mathcal{P}} \alpha_{p'} H_{p-p'} = (\alpha * H)_p$$

Model

Image is a sum of weighted translations of one atom

$$u = \alpha * H + b, \tag{1}$$

where $u \in \mathbb{R}^{\mathcal{P}}$ is the image data, $\alpha \in \mathbb{R}^{\mathcal{P}}$ is the code, $H \in \mathbb{R}^{\mathcal{P}}$ the target and b is noise.

Fast Transform

How ?

Atoms computed with a composition of K convolutions

$$H \approx h^1 * h^2 * \cdots * h^K$$

Kernels $(h^k)_{1 \leq k \leq K}$ have constrained supports defined by a mapping S^k:

$$\forall k \in \{1, \ldots, K\}, \text{supp}\left(h^k\right) \subset \text{rg}\left(S^k\right)$$

where $\text{rg}\left(S^k\right) = \{S^k(1), \ldots, S^k(S)\}$ contains all the possible locations of the non-zero elements of h^k.

Notation : $\mathbf{h} = (h^k)_{1 \leq k \leq K} \in (\mathbb{R}^{\mathcal{P}})^K$.

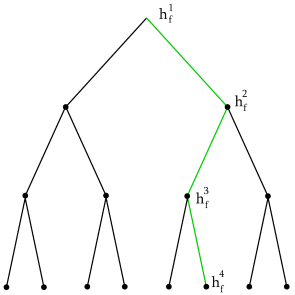


Figure: Tree structure for a dictionary.

Example of support mapping

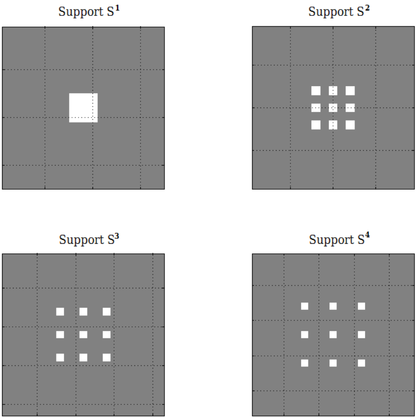


Figure: Supports $(S^k)_{1 \leq k \leq 4}$ of size $S = 3 \times 3$ upsampled by a factor k.

(P_0)

First formulation

$$(P_0): \quad \underset{(h^k)_{1\leq k\leq K}\in(\mathbb{R}^{\mathcal{P}})^K}{\operatorname{argmin}} \|\alpha * h^1 * \cdots * h^K - u\|_2^2 \quad \text{s.t. } \operatorname{supp}\left(h^k\right) \subset \operatorname{rg}\left(S^k\right)$$

(P_0)

First formulation

$$(P_0): \quad \underset{(h^k)_{1\leq k\leq K}\in(\mathbb{R}^{\mathcal{P}})^K}{\operatorname{argmin}} \|\alpha * h^1 * \cdots * h^K - u\|_2^2 \quad \text{s.t. } \operatorname{supp}\left(h^k\right) \subset \operatorname{rg}\left(S^k\right)$$

Energy gradient

$$\frac{\partial E_0(\mathbf{h})}{\partial h^k} = 2\tilde{H}^k * (\alpha * h^1 * \cdots * h^K - u), \tag{2}$$

where

$$H^k = \alpha * h^1 * \cdots * h^{k-1} * h^{k+1} * \cdots * h^K, \tag{3}$$

and where the $\tilde{.}$ operator is defined for any $h \in \mathbb{R}^{\mathcal{P}}$ as

$$\tilde{h}_p = h_{-p}, \quad \forall p \in \mathcal{P}. \tag{4}$$

(P_0)

Shortcoming

If $h^1 = h^2 = 0$,
$\nabla E_0(\mathbf{h}) = 0$ but not a global minimum.

(P_0)

Shortcoming

If $h^1 = h^2 = 0$,
$\nabla E_0(\mathbf{h}) = 0$ but not a global minimum.

Another view

$\forall (\mu_k)_{1\le k\le K} \in \mathbb{R}^K$ such that $\prod_{k=1}^K \mu_k = 1$, we have

$$E_0\left[(\mu_k h^k)_{1\le k\le K}\right] = E_0(\mathbf{h}),$$

for any $k \in \{1,\dots,K\}$,

$$\frac{\partial E_0}{\partial h^k}\left[(\mu_k h^k)_{1\le k\le K}\right] = \frac{1}{\mu_k}\frac{\partial E_0}{\partial h^k}(\mathbf{h}).$$

The gradient depends on quantities which are irrelevant regarding the value of the objective function.

New formulation: Problem (P_1)

Second formulation

$$(P_1): \quad \operatorname{argmin}_{\lambda\geq 0, \mathbf{h}\in\mathcal{D}} \|\lambda\alpha * h^1 * \cdots * h^K - u\|_2^2,$$

with

$$\mathcal{D} = \left\{\mathbf{h} \in (\mathbb{R}^{\mathcal{P}})^K \,|\, \forall k \in \{1,\ldots,K\}, \|h^k\|_2 = 1 \text{ and } \operatorname{supp}\left(h^k\right) \subset \operatorname{rg}\left(S^k\right)\right\}$$

Reminder : $\mathbf{h} = (h^k)_{1\leq k\leq K} \in (\mathbb{R}^{\mathcal{P}})^K$.

See *On the best rank-1 and rank-(R 1, R 2,..., Rn) approximation of higher-order tensors*, L. De Lathauwer, B. De Moor, J. Vandewalle, SIAM Journal on Matrix Analysis and Applications 21 (4), 1324-1342, 2000.

Existence of a solution of (P_1)

Proposition. [Existence of a solution]

For any $(u, \alpha, (S^k)_{1\leq k\leq K}) \in \left(\mathbb{R}^{\mathcal{P}} \times \mathbb{R}^{\mathcal{P}} \times (\mathcal{P}^S)^K\right)$, if

$$\forall \mathbf{h} \in \mathcal{D}, \qquad \alpha * h^1 * \ldots * h^K \neq 0, \tag{5}$$

then the problem (P_1) has a minimizer.

Existence of a solution of (P_1)

Proposition. [Existence of a solution]

For any $(u, \alpha, (S^k)_{1\leq k\leq K}) \in \left(\mathbb{R}^{\mathcal{P}} \times \mathbb{R}^{\mathcal{P}} \times (\mathcal{P}^S)^K\right)$, if

$$\forall \mathbf{h} \in \mathcal{D}, \qquad \alpha * h^1 * \ldots * h^K \neq 0, \tag{5}$$

then the problem (P_1) has a minimizer.

Proof.

Idea : use compacity of $\mathcal{D}$ and λ-coercivity of the objective function. □

Link between (P_0) and (P_1)

Proposition. [(P_1) is equivalent to (P_0)]

Let $(u, \alpha, (S^k)_{1\leq k\leq K}) \in \left(\mathbb{R}^{\mathcal{P}} \times \mathbb{R}^{\mathcal{P}} \times (\mathcal{P}^S)^K\right)$ be such that (5) holds. For any $(\lambda, \mathbf{h}) \in \mathbb{R} \times (\mathbb{R}^{\mathcal{P}})^K$, we consider the kernels $\mathbf{g} = (g^k)_{1\leq k\leq K} \in (\mathbb{R}^{\mathcal{P}})^K$ defined by

$$g^1 = \lambda\, h^1 \text{ and } g^k = h^k, \quad \forall k \in \{2, \ldots, K\}. \tag{6}$$

The following statements hold:

1. if $(\lambda, \mathbf{h}) \in \mathbb{R} \times (\mathbb{R}^{\mathcal{P}})^K$ is a stationary point of (P_1) and $\lambda > 0$ then $\mathbf{g}$ is a stationary point of (P_0).
2. if $(\lambda, \mathbf{h}) \in \mathbb{R} \times (\mathbb{R}^{\mathcal{P}})^K$ is a global minimizer of (P_1) then $\mathbf{g}$ is a global minimizer of (P_0).

Block formulation of (P_1)

Problem (P_k)

$$(P_k): \begin{cases} \mathrm{argmin}_{\lambda \geq 0, h \in \mathbb{R}^{\mathcal{P}}} \|\lambda\alpha * h^1 * \cdots * h^{k-1} * h * h^{k+1} * \ldots * h^K - u\|_2^2, \\ \text{s.t.} \quad \mathrm{supp}(h) \subset \mathrm{rg}\left(S^k\right) \text{ and } \|h\|_2 = 1 \end{cases}$$

where the kernels $(h_p^{k'})_{p \in \mathcal{P}}$ are fixed $\forall k' \neq k$.

Algorithm overview

Algorithm 1: Overview of the ALS algorithm

Input:
u: target measurements;
α: known coefficients;
$(S^k)_{1\leq k\leq K}$: supports of the kernels $(h^k)_{1\leq k\leq K}$.
Output:
λ and kernels $(h^k)_{1\leq k\leq K}$ such that $\lambda h^1 * \ldots * h^K \approx H$.

begin
  Initialize the kernels $(h^k)_{1\leq k\leq K}$;
  while *not converged* **do**
    for $k = 1, \ldots, K$ **do**
      Update λ and h^k with a minimizer of (P_k).

Matrix formulation of (P_k)

(P_k)

$$(P_k): \quad \mathrm{argmin}_{\lambda \geq 0, h \in \mathbb{R}^S} \|\lambda C_k h - u\|_2^2 \qquad \text{s.t. } \|h\|_2 = 1$$

Matrix formulation of (P_k)

(P_k)

$$(P_k): \quad \mathrm{argmin}_{\lambda \geq 0, h \in \mathbb{R}^S} \|\lambda C_k h - u\|_2^2 \qquad \text{s.t. } \|h\|_2 = 1$$

Alternative: (P'_k)

$$(P'_k): \quad \mathrm{argmin}_{h \in \mathbb{R}^S} \|C_k h - u\|_2^2.$$

(P'_k) has a minimizer $h^* \in \mathbb{R}^S$.
Computation of a stationary point yields

$$h^* = (C_k^T C_k)^{-1} C_k^T u, \tag{7}$$

Update rule

- Find h^* solution of (P'_k)

Update rule

- Find h^* solution of (P'_k)
- Update

$$\lambda = \|h^*\|_2 \qquad \text{and} \qquad h^k = \begin{cases} \frac{h^*}{\|h^*\|_2} & \text{, if } \|h^*\|_2 \neq 0, \\ \frac{1}{\sqrt{S}} \mathbb{1}_{\{1,\ldots,S\}} & \text{, otherwise,} \end{cases} \tag{8}$$

Matrix C_k

$$C_k h = \#\mathcal{P} \left\{ \underbrace{\begin{pmatrix} H^k_{p-S^k(s)} \end{pmatrix}}_{S} \begin{pmatrix} h_s \end{pmatrix} \right\} S$$

$$\underbrace{C_k^T u}_{S\times 1} = S \left\{ \underbrace{\begin{pmatrix} H^k_{p-S^k(s)} \end{pmatrix}}_{\#\mathcal{P}} \begin{pmatrix} \vdots \\ u_p \\ \vdots \end{pmatrix} \right\} \#\mathcal{P} = S \langle \ldots, \ldots \rangle_{\mathbb{R}^{\mathcal{P}}} \text{ complexity } O(S\#\mathcal{P})$$

$$\underbrace{C_k^T C_k}_{S\times S} = \begin{pmatrix} H^k_{p-S^k(s)} \end{pmatrix} \begin{pmatrix} H^k_{p-S^k(s)} \end{pmatrix} = S^2 \langle \ldots, \ldots \rangle_{\mathbb{R}^{\mathcal{P}}} \text{ complexity } O(S^2\#\mathcal{P})$$

ALS algorithm

Algorithm 2: Detailed ALS algorithm

Input:
u: target measurements;
α: known coefficients;
$(S^k)_{1\leq k\leq K}$: supports of the kernels $(h^k)_{1\leq k\leq K}$.
Output:
$(h^k)_{1\leq k\leq K}$: convolution kernels such that $h^1 * \ldots * h^K \approx H$.

begin
  Initialize the kernels $((h_p^k)_{p\in\mathcal{P}})_{1\leq k\leq K}$;
  while *not converged* **do**
    for *k = 1 ,..., K* **do**
      Compute H^k according to (3) — $O((K-1)S\#\mathcal{P})$
      Compute $C_k^T C_k$ and $C_k^T u$ — $O((S+1)S\#\mathcal{P})$
      Compute h^* according to (7); — $O(S^3)$
      Update h^k and λ according to (8); — $O(S)$
    $O(KS(K+S)\#\mathcal{P})$ per iteration of the while loop

Convergence of the algorithm

Convergence of Algorithm 2

For any $(u, \alpha, (S^k)_{1\leq k\leq K}) \in \left(\mathbb{R}^{\mathcal{P}} \times \mathbb{R}^{\mathcal{P}} \times (\mathcal{P}^S)^K\right)$, if

$$\alpha * h^1 * \ldots * h^K \neq 0, \qquad \forall \mathbf{h} \in \mathcal{D}, \tag{9}$$

then the following statements hold:

1. The sequence generated by Algorithm 2 is bounded and its limit points are in $\mathbb{R} \times \mathcal{D}$. The value of the objective function is the same for all these limit points.
2. For any limit point $(\lambda^*, \mathbf{h}^*) \in \mathbb{R} \times \mathcal{D}$, if for all $k \in \{1, \ldots, K\}$, the matrix C_k generated using $T_k(\mathbf{h}^*)$ is full column rank and $C_k^T u \neq 0$, then $(\lambda^*, \mathbf{h}^*) = T(\mathbf{h}^*)$ and $(\lambda^*, \mathbf{h}^*)$ is a stationary point of the problem (P_1).

Convergence proof

Proof.

1. The sequence of kernels generated by the algorithm belongs to $\mathcal{D}$ and $\mathcal{D}$ is compact. The objective function of (P_1) is coercive with respect to λ when (9) holds. The objective function decreases during the iterative process and is continuous.

□

Convergence proof

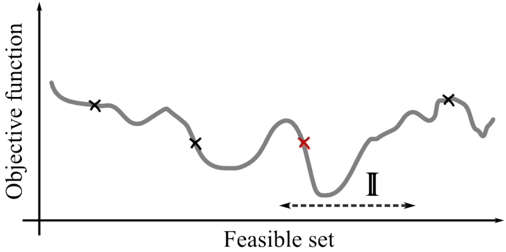
Proof.

1.
2. Consider a subsequence converging to a limit point $(\lambda^*, \mathbf{h}^*)$. The objective function is continuous. When applying the loop to the subsequence, the objective function value converges to $F(\lambda^*, \mathbf{h}^*)$. The "for" loop T is a continuous mapping in a neighborhood of $(\lambda^*, \mathbf{h}^*)$, so
$$F\left(T(\mathbf{h}^*)\right) = F(\lambda^*, \mathbf{h}^*).$$
For all k, the objective function value is the minimal value of (P_k) (unique if C_k is full column rank). So $(\lambda^*, \mathbf{h}^*)$ is also a stationary point of (P_k), and thus of (P_1).

□

Initialization and restarts

Kernel coefficients are initialized uniformly on
$\mathcal{D} = \left\{ \mathbf{h} \in (\mathbb{R}^{\mathcal{P}})^K | \forall k \in \{1, \dots, K\}, \|h^k\|_2 = 1 \text{ and } \operatorname{supp}\left(h^k\right) \subset \operatorname{rg}\left(S^k\right) \right\}$.



Drawing R initializations and returning the result for which the objective function is the smallest will yield a global minimum with probability

$$\mathbb{P}\left(\text{global}\right) = 1 - \left[\mathbb{P}\left(\mathbf{h} \notin \mathbb{I}\right)\right]^R$$

Approximation experiments setting

- Build H (a wavelet, a curvelet, a cosine ...)
- Build α (Dirac delta function, Bernoulli-Gaussian ...)
- Build $u = \alpha * H + b$
- Estimate $\lambda, (h^k)_{1 \leq k \leq K}$ from α and u, with ALS.

Approximation experiments setting

- Build H (a wavelet, a curvelet, a cosine ...)
- Build α (Dirac delta function, Bernoulli-Gaussian ...)
- Build $u = \alpha * H + b$
- Estimate $\lambda, (h^k)_{1 \leq k \leq K}$ from α and u, with ALS.

PSNR_H

$$\mathrm{PSNR}_H = 10.\log_{10}\left(r^2/\mathrm{MSE}_H\right).$$

where $r = \max_{p \in \mathcal{P}}(H_p) - \min_{p \in \mathcal{P}}(H_p)$. and

$$\mathrm{MSE}_H = \frac{\|\lambda h^1 * \cdots * h^K - H\|_2^2}{\# \mathrm{supp}\,(H)}.$$

Approximation experiments setting

- Build H (a wavelet, a curvelet, a cosine ...)
- Build α (Dirac delta function, Bernoulli-Gaussian ...)
- Build $u = \alpha * H + b$
- Estimate $\lambda, (h^k)_{1 \leq k \leq K}$ from α and u, with ALS.

PSNR_H

$$\mathrm{PSNR}_H = 10.\log_{10}\left(r^2/\mathrm{MSE}_H\right).$$

where $r = \max_{p\in\mathcal{P}}(H_p) - \min_{p\in\mathcal{P}}(H_p)$. and

$$\mathrm{MSE}_H = \frac{\|\lambda h^1 * \cdots * h^K - H\|_2^2}{\#\operatorname{supp}(H)}.$$

NRE

$$\mathrm{NRE} = \frac{\|\lambda\alpha * h^1 * \cdots * h^K - u\|_2^2}{\|u\|_2^2}. \tag{10}$$

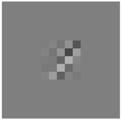
Curvelet

H obtained by inverse curvelet transform of a Dirac function in a 128×128 image. $K = 7, S = 5 \times 5$. α is a Dirac function, $\frac{\#\text{supp}(H)}{KS} \sim 43$.

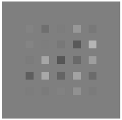
Approximation $\lambda h^1 * \cdots * h^K$
$\mathrm{PSNR}_H = 44.30$

True curvelet atom H

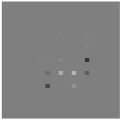
Curvelet



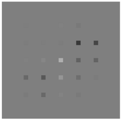
kernel h_1



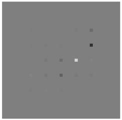
kernel h_2



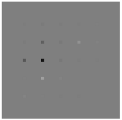
kernel h_3



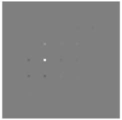
kernel h_4



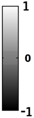
kernel h_5



kernel h_6

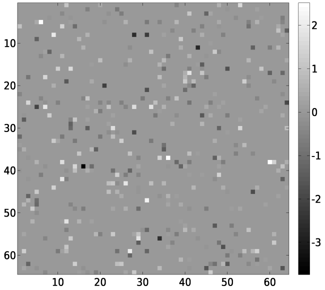


kernel h_7



Cosine function

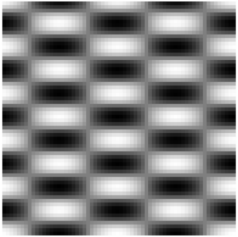
Target atom is a 2D 64×64 cosine.
The code α is Bernoulli-Gaussian distributed.



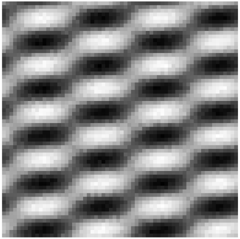
Code α

Cosine function

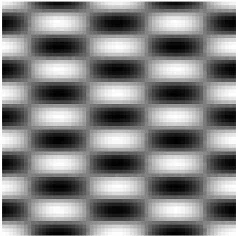
Reconstruction of H, $K = 7$, $S = 5 \times 5$ $\sigma^2 = 0.5$, $\frac{\#\mathrm{supp}(H)}{KS} \sim 23$.



Atom H



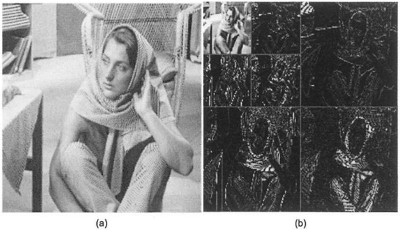
$u = \alpha * H + b$



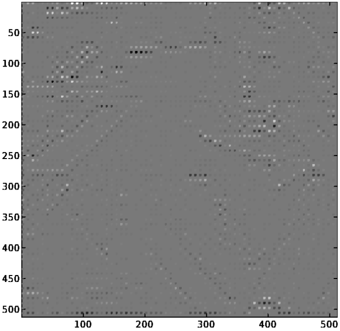
$\lambda * h^1 * \cdots * h^K$
$\mathrm{PSNR}_H = 41.44$

Wavelet

H chosen as 3-level horizontal detail wavelet.
Code α obtained by 2^3 upsampling the IWT of 3-level horizontal coefficients.

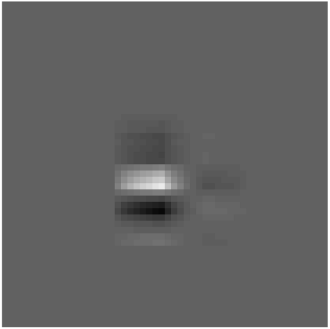


Wavelet decomposition

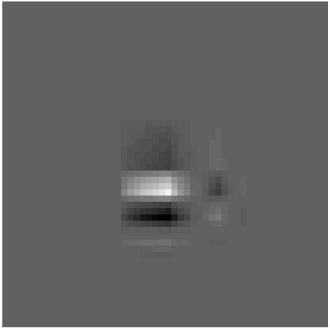


Code α

Approximation of H with u obtained as IWT of horizontal detail coefficients
Noise power $\sigma^2 = 5$, $K = 6, S = 3 \times 3$. The reachable support is a size 42×42 window.



$\lambda * h^1 * \cdots * h^K$
$\mathrm{PSNR}_H = 36.61$



Atom H

Sinc function

Zoom $\times 3$ of a $N_0 = 128$ signal with a $N = 384$ sinc generated with a length 128 step function in the Fourier domain. $K = 9, S = 9, \sharp\mathrm{supp}(H) = 384$, $\frac{\#\mathrm{supp}(H)}{KS} \sim 4.7$.

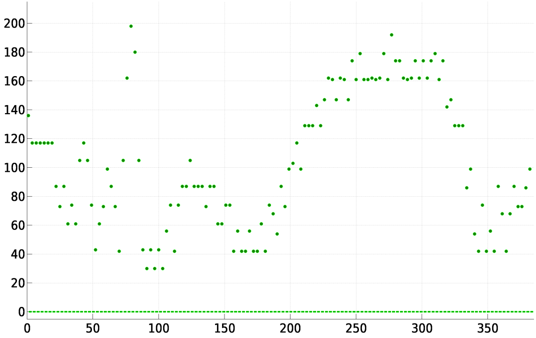
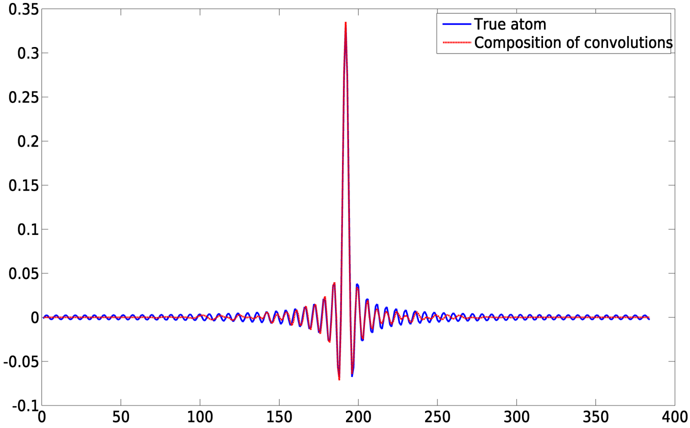


Figure: Code α

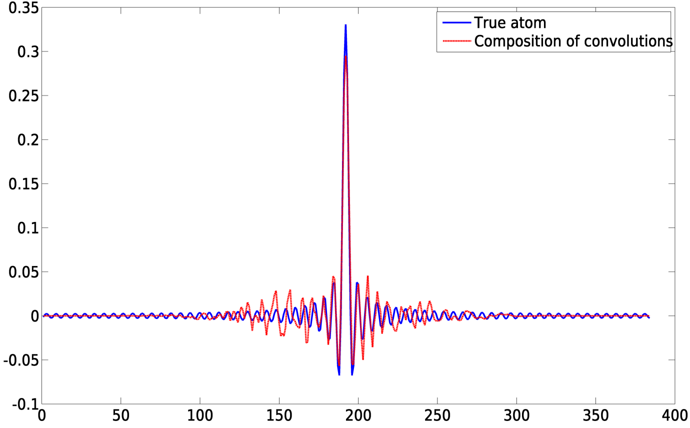
Sinc function

Reconstruction of H, $\sigma^2 = 0$:



Sinc function

Reconstruction of H, $\sigma^2 = 5$ ($\mathrm{PSNR}_H = 44.5dB$) :



Observations

PSNR_H(dB)	$K = 3$	$K = 5$	$K = 7$	$K = 9$	$K = 11$
$S = 3 \times 3$	11.79	12.27	13.81	25.15	30.09
$S = 5 \times 5$	11.94	15.97	41.44	38.94	39.82

Table: 2D Cosine: PSNR_H.

NRE	$K = 3$	$K = 5$	$K = 7$	$K = 9$	$K = 11$
$S = 3 \times 3$	1.02	0.89	0.41	0.04	0.02
$S = 5 \times 5$	0.96	0.24	0.01	0.01	0.01

Table: 2D Cosine: NRE.

2D Cosine approximation: PSNR_H and NRE for several values of K and S.

Observations

Noiseless experiments

- + PSNR_H improves with K and S.
- + NRE improves with K and S.

Noisy experiments

- - Improvement of PSNR_H not stable because of the lack of regularization.
- + Several occurences of the atom (through the code) improves noise robustness.

  Keep an eye on the conditioning of the convolution with the code.

Conclusions

- **Composition of sparse convolution can be optimized:**
  - Algorithm complexity linear with respect to the image size.
  - Small search space for large atoms in large images.
- **A composition of convolutions accurately approximate atom-like signals and images.**

Thank you for your attention !

Find the paper and a few experiments: google: Malgouyres Toulouse

Principle

This section evaluates $\mathbb{P}(\mathbf{h} \in \mathbb{I})$
for 1D signals of length $\#\mathcal{P} = 128$ and $(K, S) \in \{2, \ldots, 7\} \times \{2, \ldots, 10\}$.

$\forall k \in \{1, \ldots, K\}$,
Random support mappings: $\mathrm{rg}\left(S^k\right) \sim \mathcal{U}_{\{1,\ldots,10\}}$
Independent random kernels:

$$h_p^k \begin{cases} \sim \mathcal{N}(0,1) & \text{, if } p \in \mathrm{rg}\left(S^k\right) \\ = 0 & \text{, otherwise.} \end{cases}$$

The image u is obtained by convolving the kernels

$$u = h^1 * \cdots * h^K + b$$

where $b \sim \mathcal{N}(0, \sigma^2)$.

Performance measure

We consider that Algorithm 2 has converged to a global minimum if

$$\|\alpha * \overline{h}^1 * \ldots * \overline{h}^K - u\|_2^2 \leq \sigma^2 (\#\mathcal{S}) + 10^{-4} \|u\|_2^2. \tag{11}$$

Performance measure

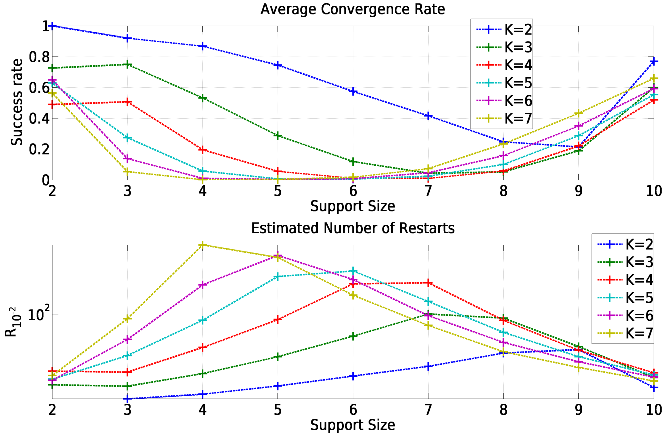
For any fixed $(K,S) \in \{2,\ldots,7\} \times \{2,\ldots,10\}$,
Generate $L = 50K^2$ experiments. For each experiment, draw $R = 25$ random initializations according to a uniform distribution in $\mathcal{D}$.
Estimation of the probability of reaching a global minimum of (P_1):

$$\mathbb{P}(\text{global minimizer}) \simeq \frac{1}{LR} \sum_{l=1}^{L} \sum_{r=1}^{R} \mathbb{1}(l,r).$$

with

$$\mathbb{1}(l,r) = \begin{cases} 1, & \text{if (11) holds for the } r\text{th result obtained from the } l\text{th input,} \\ 0, & \text{otherwise.} \end{cases}$$

Results (noise-free case)



Results (noisy case)