Patch-based methods - Sparse coding and Dictionary Learning

Julie Digne

LIRIS - Équipe GeoMod - CNRS
Patch-based processing

- Do not process an image only by looking at a pixel but by looking at patches
Similarity Analysis: Non Local Means [Buadès et al. 2005]

- Idea: denoise a point by comparing it to similar neighborhoods
- Compute local patch $P(p)$ around each point $p$
- Similarity measure between two points: $w(p, q) = \exp - \frac{\text{dist}(P(p), P(q))^2}{\sigma}$
- Update of the image:

$$I_{new}(p) = \frac{\sum_{q \in I} w(p, q)I(q)}{\sum_{q \in I} w(p, q)}$$
Non local means result

noisy

non-local means
Noisy image
Gaussian filter result
Gaussian filter result
Gaussian filter result
Median result
Median result
NLmeans result
Comparison
Outline

1 Visual Summary

2 Efficient Similar Patch Search

3 Sparse Coding of signals

4 Dictionary Learning

5 Optimal transportation meets nonlocal means

6 Half-toning
Goal: Produce a smaller image that summarizes the content of the larger image.
Comparing two images

Bidirectional Distance (BDS)

A source image $S$, a target image $T$, then:

$$d_{BDS}(S, T) = \frac{1}{N_S} \sum_{s \subset S} \min_{t \subset T} D(s, t) + \frac{1}{N_T} \sum_{t \subset T} \min_{s \subset S} D(t, s)$$

where $s$ and $t$ are patches of fixed size of $S$ and $T$. $D$ is the sum of squared difference between patches. $N_S$, $N_T$ number of patches in $S$ and $T$. 

Comparing two images

Bidirectional Distance (BDS)

A source image $S$, a target image $T$, then:

$$d_{BDS}(S, T) = \frac{1}{N_S} \sum_{s \subset S} \min_{t \subset T} D(s, t) + \frac{1}{N_T} \sum_{t \subset T} \min_{s \subset S} D(t, s)$$

where $s$ and $t$ are patches of fixed size of $S$ and $T$. $D$ is the sum of squared difference between patches. $N_S$, $N_T$ number of patches in $S$ and $T$.

- Completeness term
Comparing two images

Bidirectional Distance (BDS)

A source image $S$, a target image $T$, then:

$$d_{BDS}(S, T) = \frac{1}{N_S} \sum_{s \subseteq S} \min_{t \subseteq T} D(s, t) + \frac{1}{N_T} \sum_{t \subseteq T} \min_{s \subseteq S} D(t, s)$$

where $s$ and $t$ are patches of fixed size of $S$ and $T$. $D$ is the sum of squared difference between patches. $N_S$, $N_T$ number of patches in $S$ and $T$.

- Completeness term
- Coherence term
Comparing two images

Bidirectional Distance (BDS)

A source image $S$, a target image $T$, then:

$$d_{BDS}(S, T) = \frac{1}{N_S} \sum_{s \subset S} \min_{t \subset T} D(s, t) + \frac{1}{N_T} \sum_{t \subset T} \min_{s \subset S} D(t, s)$$

where $s$ and $t$ are patches of fixed size of $S$ and $T$. $D$ is the sum of squared difference between patches. $N_S$, $N_T$ number of patches in $S$ and $T$.

- Completeness term
- Coherence term

Reconstruction

Starting from an initial guess $T_0$ for $T$, build an image iteratively as the minimizer $T$ of $d_{BDS}(S, T)$
Similarity distance

Source

(a) complete and coherent

(b) complete coherent

(c) complete coherent
Similarity distance

\[ d = 667 \quad d = 857 \quad d = 597 \]
A three steps algorithm

1. For each patch of $S$ find its closest patch on $T$
A three steps algorithm

1. For each patch of $S$ find its closest patch on $T$
2. For each patch of $T$ find its closest patch on $S$
A three steps algorithm

1. For each patch of $S$ find its closest patch on $T$
2. For each patch of $T$ find its closest patch on $S$
3. Aggregate color of nearest patches an update colors of $T$
Aggregation Step: contribution of a pixel to the coherence measure

Let $q$ be a pixel of $T$, $q$ lies inside $m$ neighboring patches $Q_1, Q_2, \ldots, Q_m$. 

\[ \text{Contribution}_1 = \sum_{i=1}^{N_T} \| S(p_i) - T(q) \|_2 \]
Aggregation Step: contribution of a pixel to the coherence measure

- Let $q$ be a pixel of $T$, $q$ lies inside $m$ neighboring patches $Q_1, Q_2, \cdots, Q_m$
- These patches are matched to $P_1, P_2, \cdots, P_m$
Aggregation Step: contribution of a pixel to the coherence measure

- Let \( q \) be a pixel of \( T \), \( q \) lies inside \( m \) neighboring patches \( Q_1, Q_2, \cdots, Q_m \)
- These patches are matched to \( P_1, P_2, \cdots, P_m \)
- the positions corresponding to \( q \) in \( P_1, P_2, \cdots, P_m \) are \( p_1, \cdots, p_m \)
Let $q$ be a pixel of $T$, $q$ lies inside $m$ neighboring patches $Q_1, Q_2, \cdots, Q_m$

These patches are matched to $P_1, P_2, \cdots, P_m$

the positions corresponding to $q$ in $P_1, P_2, \cdots, P_m$ are $p_1, \cdots, p_m$

**Contribution**

\[
\frac{1}{N_T} \sum_{i=1}^{m} \| S(p_i) - T(q) \|^2
\]
Aggregation Step: contribution of a pixel to the completeness measure

Let $q$ be a pixel of $T$, \[ n \sum_{i=1}^{n} \| S(\hat{p}_i) - T(q) \|^2 \]
Aggregation Step: contribution of a pixel to the completeness measure

- Let $q$ be a pixel of $T$,
- $q$ lies inside $n$ neighboring patches $\hat{Q}_1, \hat{Q}_2, \ldots, \hat{Q}_n$ that are the nearest patch to some patches of $S \hat{P}_1, \hat{P}_2, \ldots \hat{P}_n$
Aggregation Step: contribution of a pixel to the completeness measure

- Let $q$ be a pixel of $T$,
- $q$ lies inside $n$ neighboring patches $\hat{Q}_1, \hat{Q}_2, \ldots, \hat{Q}_n$ that are the nearest patch to some patches of $S \hat{P}_1, \hat{P}_2, \ldots \hat{P}_n$.
- The positions corresponding to $q$ in $\hat{P}_1, \hat{P}_2, \ldots, \hat{P}_m$ are $\hat{p}_1, \ldots, \hat{p}_m$. 

Aggregation Step: contribution of a pixel to the completeness measure

- Let $q$ be a pixel of $T$,
- $q$ lies inside $n$ neighboring patches $\hat{Q}_1, \hat{Q}_2, \cdots, \hat{Q}_n$ that are the nearest patch to some patches of $S\ \hat{P}_1, \hat{P}_2, \cdots, \hat{P}_n$
- The positions corresponding to $q$ in $\hat{P}_1, \hat{P}_2, \cdots, \hat{P}_m$ are $\hat{p}_1, \cdots, \hat{p}_m$

Contribution

\[
\frac{1}{N_S} \sum_{i=1}^{n} \| S(\hat{p}_i) - T(q) \|^2
\]
Color Update

The best $T(q)$ should minimize:

\[
\frac{1}{N_S} \sum_{i=1}^{n} \| S(\hat{p}_i) - T(q) \|^2 + \frac{1}{N_T} \sum_{i=1}^{m} \| S(p_i) - T(q) \|^2
\]
Color Update

The best $T(q)$ should minimize:

$$
\frac{1}{N_s} \sum_{i=1}^{n} \| S(\hat{p}_i) - T(q) \|^2 + \frac{1}{N_T} \sum_{i=1}^{m} \| S(p_i) - T(q) \|^2
$$

Color Update

$$
T(q) = \frac{1}{N_s} \sum_{i=1}^{n} S(\hat{p}_i) + \frac{1}{N_T} \sum_{i=1}^{m} S(p_i)
$$
Visual Summary

Source = $T_0$

$d(S, T_0) = 0$

$d(S, T_1) = 175$

$d(S, T_2) = 345$

$d = 504$

$d = 597$

$d = 640$

$d = 980$

$1408$

$2150$
Gradual resizing

- When the target has a very different size from the source: what is a good initial guess?

Video
Gradual resizing

- When the target has a very different size from the source: what is a good initial guess?
- Iterative process: downsample the image and apply the reconstruction

```
Video
```
Key ingredient for all these methods

**Requirement**

A fast method to find similar patches

- Naive way: traverse the whole image at each query
Key ingredient for all these methods

Requirement
A fast method to find similar patches

- Naive way: traverse the whole image at each query
- Better: put all patches in a search structure
Key ingredient for all these methods

**Requirement**

A fast method to find similar patches

- Naive way: traverse the whole image at each query
- **Better:** put all patches in a search structure
- **Even better:** the patch match algorithm
Outline

1 Visual Summary

2 Efficient Similar Patch Search

3 Sparse Coding of signals

4 Dictionary Learning

5 Optimal transportation meets nonlocal means

6 Half-toning
Patch Match
Similar patches

The similarity distance between two patches $p_A$, $p_B$ of size $n \times n$ is computed as $
abla_{1 \leq i, j \leq n} \|p_A(i, j) - p_B(i, j)\|_2^2$. 

Similarity

Two patches are considered as similar if their similarity distance is small.
Patch Match

Goal

Given an image $A$ and an image $B$ find *efficiently* for all patches of image $A$ the corresponding nearest patch of image $B$. 
Patch Match

Goal
Given an image $A$ and an image $B$ find efficiently for all patches of image $A$ the corresponding nearest patch of image $B$.

Patch Match Principle
Assume we have found a patch $p_B$ of $B$ corresponding to a given patch $p_A$ of $A$, assume we have a patch $p'_A$ located close to $p_A$ in image $A$, then its corresponding patch $p'_B$ has a high probability to lie close to $p_B$

- Look for $p'_B$ close to $p_B$!
What if?

- If we have an initial corresponding pairs \((p_A, p_B)\) then the search is made easier.
What if?

- If we have an initial corresponding pairs \((p_A, p_B)\) then the search is made easier.
- **However:** How can we find an initial pair?
What if?

- If we have an initial corresponding pairs \((p_A, p_B)\) then the search is made easier.
- **However:** How can we find an initial pair?
- **However:** Should we start with a single initial pair? Why not many?
What if?

- If we have an initial corresponding pairs \((p_A, p_B)\) then the search is made easier.
- **However:** How can we find an initial pair?
- **However:** Should we start with a single initial pair? Why not many?

**Notation**

Let \(p_A\) be a patch centered at \(a\) in image \(A\) and \(p_B\) a patch centered at \(b\) in image \(B\). We define **an offset vector** \(f(a)\) as \(f(a) = b - a\). The set of all offset vectors is called the **Nearest Neighbor Field (NNF)**.
Algorithm

1. Initialize the NNF with random vectors
2. **Propagation:** for $i = 1 \cdots M$, for $j = 1 \cdots M$
   1. Evaluate the offset $f(i-1,j)$, $f(i-1,j-1)$, $f(i-1,j+1)$ and $f(i,j-1)$
   2. If one of them is better than $f(i,j)$ replace $f(i,j)$ with it.
3. **Randomization:** For all $(i,j)$, draw a random offset $w$, if $w$ is better than $f(i,j)$ set $f(i,j) = w$
Algorithm

(a) Initialization

(b) Propagation

(c) Search
Algorithm
Probabilistic analysis

Exercise
Assume we have two images of size $M$ and assign randomly patches of image $A$ to patches of image $B$ (+ unicity of the correspondence)
Exercise
Assume we have two images of size $M$ and assign randomly patches of image $A$ to patches of image $B$ (+ unicity of the correspondence)

What is the probability that at least one patch is indeed paired to its corresponding patch?
Probabilistic analysis

Exercise
Assume we have two images of size $M$ and assign randomly patches of image $A$ to patches of image $B$ (+ unicity of the correspondence)

What is the probability that at least one patch is indeed paired to its corresponding patch?

Simplification
Assume that a pair is correct if a patch is assigned to a patch that is spatially close (in a neighborhood of size $C$) to its true correspondence
Probabilistic analysis

Exercise

Assume we have two images of size $M$ and assign randomly patches of image $A$ to patches of image $B$ (+ unicity of the correspondence)

What is the probability that at least one patch is indeed paired to its corresponding patch?

Simplification

Assume that a pair is correct if a patch is assigned to a patch that is spatially close (in a neighborhood of size $C$) to its true correspondence

What is the probability that at least one patch is paired to an approximate corresponding patch?
Reshuffling Application
Deformation Application

(a) building marked by user

(b) scaled up, preserving texture

(c) bush marked by user

(d) scaled up, preserving texture.
Outline

1. Visual Summary
2. Efficient Similar Patch Search
3. Sparse Coding of signals
4. Dictionary Learning
5. Optimal transportation meets nonlocal means
6. Half-toning
Sparse processing of signals

Signal Processing aims to decompose complex signals using elementary functions which are then easier to manipulate.

\[ x(t) = \sum_{i=-\infty}^{\infty} \alpha_i \varphi_i(t) \]

*Image: Pier Luigi Dragotti*
Sparse processing of signals

Image: Pier Luigi Dragotti
Sparse processing of signals

- Between two representations of a signal pick the ones with the higher number of zero coefficients.
Patch-based approaches for images and surfaces

- Texture synthesis [Efros 99], Non local means [Buades et al. 2005].

**Compressive sensing theory [Candes et al. 2006]**

There exists spaces, in which the signals would be sparsely represented, that are especially well suited for processing the signals.

- Sparse regularization for image analysis, inpainting... [Elad et al. 2006], [Mairal 2009] The K-SVD algorithm
A brief reminder on norms

Norm definition

Let $E$ be a vector space over a subfield $K$, a norm on $E$ is an application with nonnegative values $\|\| : E \rightarrow R$ such that for all $\alpha \in K$ and $u, v \in E$:

- $\| \alpha v \| = |\alpha| \| v \|$ (positive homogeneity)
- $\| u + v \| \leq \| u \| + \| v \|$ (subadditivity)
- $\| u \| = 0_K \iff u = 0_E$ (separation)
A brief reminder on norms

Norm definition

Let $E$ be a vector space over a subfield $K$, a norm on $E$ is an application with nonnegative values $||| : E \rightarrow R$ such that for all $\alpha \in K$ and $u, v \in E$:

- $||\alpha v|| = |\alpha||v||$ (positive homogeneity)
- $||u + v|| \leq ||u|| + ||v||$ (subadditivity)
- $||u|| = 0_K \iff u = 0_E$ (separation)

- The $\ell^2$ norm is also called the euclidean norm. Let $x$ be a vector in $\mathbb{R}^n$ with coordinates $(x_1, \cdots, x_n)$ in the canonical basis, the $\ell^2$ norm writes:

$$||x||_2 = \sqrt{x \cdot x^T} = \left(\sum_{i=1}^{n} x_i^2\right)^{\frac{1}{2}}$$
Norm Examples on vectors of $\mathbb{R}^n$

- $\ell^1$ Norm (Manhattan)

$$\|x\|_1 = \left( \sum_{i=1}^{n} |x_i| \right)$$
Norm Examples on vectors of $\mathbb{R}^n$

- $\ell^1$ Norm (Manhattan)

$$\|x\|_1 = \left( \sum_{i=1}^{n} |x_i| \right)$$

- $\ell^3$

$$\|x\|_3 = \left( \sum_{i=1}^{n} |x_i|^3 \right)^{\frac{1}{3}}$$
Norm Examples on vectors of $\mathbb{R}^n$

- $\ell^1$ Norm (Manhattan)
  $$\|x\|_1 = \left( \sum_{i=1}^{n} |x_i| \right)$$

- $\ell^3$
  $$\|x\|_3 = \left( \sum_{i=1}^{n} |x_i|^3 \right)^{\frac{1}{3}}$$

- $\ell^{2.1}$
  $$\|x\|_{2.1} = \left( \sum_{i=1}^{n} x_i^{2.1} \right)^{\frac{1}{2.1}}$$

Exercice: Prove that $\ell^\infty$ is indeed a norm?
Norm Examples on vectors of $\mathbb{R}^n$

- $\ell^1$ Norm (Manhattan)
  \[ \|x\|_1 = \left( \sum_{i=1}^{n} |x_i| \right) \]

- $\ell^3$
  \[ \|x\|_3 = \left( \sum_{i=1}^{n} |x_i|^3 \right)^{\frac{1}{3}} \]

- $\ell^{2.1}$
  \[ \|x\|_{2.1} = \left( \sum_{i=1}^{n} x_i^{2.1} \right)^{\frac{1}{2.1}} \]

- $\ell^p$ pour $p \geq 1$
  \[ \|x\|_p = \left( \sum_{i=1}^{n} |x_i|^p \right)^{\frac{1}{p}} \]
Norm Examples on vectors of $\mathbb{R}^n$

- $\ell^1$ Norm (Manhattan)
  \[ \|x\|_1 = \left( \sum_{i=1}^{n} |x_i| \right) \]

- $\ell^3$
  \[ \|x\|_3 = \left( \sum_{i=1}^{n} |x_i|^3 \right)^{\frac{1}{3}} \]

- $\ell^{2.1}$:
  \[ \|x\|_{2.1} = \left( \sum_{i=1}^{n} x_i^{2.1} \right)^{\frac{1}{2.1}} \]

- $\ell^p$ pour $p \geq 1$
  \[ \|x\|_p = \left( \sum_{i=1}^{n} |x_i|^p \right)^{\frac{1}{p}} \]

- $\ell^\infty$
  \[ \|x\|_\infty = \max_{i=1\ldots n} |x_i| \]

**Exercice:** Prove that $\ell^\infty$ is indeed a norm?
The ball of radius 1 for norms $\ell^p$ with $p \geq 2$
The ball of radius 1 for norms $\ell^p$ with $p \geq 2$
The ball of radius 1 for norms $\ell^p$ with $p \geq 2$
The ball of radius 1 for norms $\ell^p$ with $p \geq 2$
The ball of radius 1 for norms $\ell^p$ with $p \leq 2$
The ball of radius 1 for norms $\ell^p$ with $p \leq 2$
The ball of radius 1 for norms $\ell^p$ with $p \leq 2$
The ball of radius 1 for norms $\ell^p$ with $p \leq 2$
The ball of radius 1 with norms and quasi-norms $\ell^p$
Norm and sparsity

**Sparsity definition**

A vector $x \in \mathbb{R}^N$ is said to be $s$-sparse if at most $s$ of its entries are non-zero, i.e.

$$\text{card } \text{support}(x) \leq s$$

where $\text{support}(x) = \{i | x_i \neq 0\}$.

We note $\|x\|_0 = \text{card } \text{support}(x)$ and call it $\ell^0$. 
Norm and sparsity

**Sparsity definition**

A vector $x \in \mathbb{R}^N$ is said to be $s$-sparse if at most $s$ of its entries are non zero, i.e.

$$\text{card } \text{support}(x) \leq s$$

where $\text{support}(x) = \{i | x_i \neq 0\}$.

We note $\|x\|_0 = \text{card } \text{support}(x)$ and call it $\ell^0$.

- Is $\ell^0$ a norm?
Sparse Coding with the $\ell^0$ norm

Problem statement

Let $A \in \mathbb{R}^{m \times n}$ and $x$ a $s$-sparse vector in $\mathbb{R}^n$ Let $y \in \mathbb{R}^m$ such that $y = Ax$. Assume only $y$ and $A$ are known and we want to recover $x$. If $m < n$, the system is underdetermined.
Sparse Coding with the $\ell^0$ norm

Problem statement

Let $A \in \mathbb{R}^{m \times n}$ and $x$ a $s$-sparse vector in $\mathbb{R}^n$ Let $y \in \mathbb{R}^m$ such that $y = Ax$. Assume only $y$ and $A$ are known and we want to recover $x$. If $m < n$, the system is underdetermined.

Sparsity hypothesis

Identifying the solution $x$ under the $s$-sparsity hypothesis is easier.
Sparse Coding with the $\ell^0$ norm

**Optimization problem**

Given a measurement matrix $A \in \mathbb{R}^{m \times n}$ and $y$ a vector in $\mathbb{R}^n$, under the $s$-sparse assumption, the vector $x$ can be reconstructed as the solution of:

\[
\begin{align*}
\text{Minimize} &\quad \|x\|_0 \\
\text{s.t.} &\quad y = Ax
\end{align*}
\]  

($P_0$)
Sparse Coding with the $\ell^0$ norm

**Optimization problem**

Given a measurement matrix $A \in \mathbb{R}^{m \times n}$ and $y$ a vector in $\mathbb{R}^n$, under the $s$-sparse assumption, the vector $x$ can be reconstructed as the solution of:

$$\begin{align*}
\text{Minimize} & \quad \|x\|_0 \\
\text{s.t.} & \quad y = Ax
\end{align*} \quad (P_0)$$

- This Optimization is a nonconvex optimization problem
Problem \((P_0)\) is a NP-hard problem

- Reformulate the problem as
  \[
  \min_{x \in \mathbb{R}^n} \|x\|_0
  \]
  \(\text{s.t. } \|y - Ax\|_2 \leq \eta\) 

Theorem

Problem \((P_{0,\eta})\) is a NP-hard problem

- NP-hardness: all problems for which a solving algorithm could be turned in polynomial time into a solving algorithm for any NP-problem.
- Proof: demonstrate that using Problem \((P_{0,\eta})\) one can solve for the exact cover 3-set problem.
- Reminder: Given a collection \(S\) of 3-subsets of a set \(X\), an exact cover of \(X\) is a subcollection \(S_{sub}\) of \(S\) such that the intersection of two distinct elements of \(S_{sub}\) is empty and the union of all elements of \(S_{sub}\) cover \(X\).
Sparse decomposition algorithm

**Sparse Decomposition**

Given a dictionary $D \in \mathbb{R}^{m \times n}$ whose columns have norm 1 and a signal $y \in \mathbb{R}^n$ find a vector $x$ whose sparsity is $s$ minimizing $Dx = y$

- Efficient greedy algorithms have been proposed to find an approximate solution.
Matching Pursuit

Matching Pursuit Algorithm [Mallat & Zhang 1993]

- Set \( k = 0, \alpha = 0_{\mathbb{R}^n} \)
- While \( k < s \) and \( \|x - D\alpha\| > 0 \) do:
  - Select index \( j \) maximizing \( |D_j^T \cdot (x - D\alpha)| \)
  - Update coefficients \( \alpha(j) = \alpha(j) + D_j^T \cdot (x - D\alpha) \)
Matching Pursuit

Matching Pursuit Algorithm [Mallat & Zhang 1993]

- Set \( k = 0, \alpha = 0_{\mathbb{R}^n} \)
- While \( k < s \) and \( \| x - D\alpha \| > 0 \) do:
  - Select index \( j \) maximizing \( |D_j^T \cdot (x - D\alpha)| \)
  - Update coefficients \( \alpha(j) = \alpha(j) + D_j^T \cdot (x - D\alpha) \)

At each step the algorithm finds the atom that best represents the residual \( r = x - D\alpha \)
### Matching Pursuit Algorithm [Mallat & Zhang 1993]

- Set $k = 0$, $\alpha = 0_{\mathbb{R}^n}$
- While $k < s$ and $\|x - D\alpha\| > 0$ do:
  - Select index $j$ maximizing $|D_j^T \cdot (x - D\alpha)|$
  - Update coefficients $\alpha(j) = \alpha(j) + D_j^T \cdot (x - D\alpha)$

- At each step the algorithm finds the atom that best represents the residual $r = x - D\alpha$
- The residual monotonically decreases until the residual is orthogonal to all $D_j$
Matching Pursuit

Matching Pursuit Algorithm [Mallat & Zhang 1993]

- Set $k = 0$, $\alpha = 0_{\mathbb{R}^n}$
- While $k < s$ and $\|x - D\alpha\| > 0$ do:
  - Select index $j$ maximizing $|D_j^T \cdot (x - D\alpha)|$
  - Update coefficients $\alpha(j) = \alpha(j) + D_j^T \cdot (x - D\alpha)$

- At each step the algorithm finds the atom that best represents the residual $r = x - D\alpha$
- The residual monotonically decreases until the residual is orthogonal to all $D_j$

How does the number of nonzeros behave?
Matching Pursuit

Matching Pursuit Algorithm [Mallat & Zhang 1993]

- Set $k = 0$, $\alpha = 0_{\mathbb{R}^n}$
- While $k < s$ and $\|x - D\alpha\| > 0$ do:
  - Select index $j$ maximizing $|D_j^T \cdot (x - D\alpha)|$
  - Update coefficients $\alpha(j) = \alpha(j) + D_j^T \cdot (x - D\alpha)$

- At each step the algorithm finds the atom that best represents the residual $r = x - D\alpha$
- The residual monotonically decreases until the residual is orthogonal to all $D_j$

- How does the number of nonzeros behave? nondecreasing
Orthogonal Matching Pursuit

- **Goal**: The number of nonzeros should increase at each step.
Orthogonal Matching Pursuit

- Goal: The number of nonzeros should increase at each step.

- How?

Orthogonal Matching Pursuit (OMP)

\[
\begin{align*}
&\text{Set } k = 0, \alpha = 0, R_n, \Gamma = \emptyset, \\
&\text{While } k < s \text{ and } \|x - D\Gamma \alpha\| > 0 \text{ do:} \\
&\quad \text{Select index } j \text{ maximizing } |D^T_j \cdot (x - D\Gamma \alpha)| \\
&\quad \text{Update the active set } \Gamma = \Gamma \cup \{j\} \\
&\quad \text{Recompute } \alpha \text{ minimizing } x - D\Gamma \alpha \\
&\quad \text{Set } \alpha_{\overline{\Gamma}} = 0
\end{align*}
\]

Remark: \(D\Gamma, \alpha\Gamma\): matrix (resp. vector) composed of the columns (resp. elements) of \(D\) (resp. \(\alpha\)) whose indices are in \(\Gamma\).
Orthogonal Matching Pursuit

- **Goal**: The number of nonzeros should increase at each step.
- **How?** Render the residual orthogonal to all selected atoms.
Orthogonal Matching Pursuit

- **Goal**: The number of nonzeros should increase at each step.

- **How?** Render the residual orthogonal to all selected atoms.
Orthogonal Matching Pursuit

- **Goal:** The number of nonzeros should increase at each step.

- **How?** Render the residual orthogonal to all selected atoms.

---

Orthogonal Matching Pursuit (OMP)

- Set $k = 0$, $\alpha = 0_{\mathbb{R}^n}$, $\Gamma = \emptyset$
- While $k < s$ and $\|x - D_{\Gamma} \alpha_{\Gamma}\| > 0$ do:
  - Select index $j$ maximizing $|D_j^T \cdot (x - D_{\Gamma} \alpha_{\Gamma})|$
  - Update the active set $\Gamma = \Gamma \cup \{j\}$
  - Recompute $\alpha_{\Gamma}$ minimizing $x - D_{\Gamma} \alpha_{\Gamma}$
  - Set $\alpha_{\bar{\Gamma}} = 0$

- Remark: $D_{\Gamma}$, $\alpha_{\Gamma}$: matrix (resp. vector) composed of the columns (resp. elements) of $D$ (resp. $\alpha$) whose indices are in $\Gamma$. 
What can we prove about OMP?

- The index selection is guided by finding the one that makes the error decrease most.
What can we prove about OMP?

- The index selection is guided by finding the one that makes the error decrease most.
- What about the case where a vector is a linear combination of 3 vectors.
What can we prove about OMP?

- The index selection is guided by finding the one that makes the error decrease most.
- What about the case where a vector is a linear combination of 3 vectors.
- Tropp and Gilbert (2007): OMP is able to reliably recover a sparse vector from random measurements.
What can we prove about OMP?

- The index selection is guided by finding the one that makes the error decrease most.
- What about the case where a vector is a linear combination of 3 vectors.
- Tropp and Gilbert (2007): OMP is able to reliably recover a sparse vector from random measurements.
- OMP is slower than MP
How can we make OMP faster?

Which step is computationally intensive?

- Computing the best index means computing $D_f^T(x - D_\Gamma \alpha_{\Gamma})$ and taking the index of the smallest coefficient.
How can we make OMP faster?

Which step is computationally intensive?

- Computing the best index means computing $D_f^T(x - D_f \alpha_f)$ and taking the index of the smallest coefficient
- Then we compute $\alpha_f'$ as $\alpha$ minimizing $\|x - D_f' \alpha\|^2$
How can we make OMP faster?

Which step is computationally intensive?

- Computing the best index means computing $D_f^T(x - D_f\alpha_f)$ and taking the index of the smallest coefficient
- Then we compute $\alpha_{f'}$ as $\alpha$ minimizing $\|x - D_{f'}\alpha\|^2$
- Closed form solution:
  
  $$\alpha_{f'} = (D_{f'}^TD_{f'})^{-1}D_{f'}x$$
How can we make OMP faster?

Which step is computationally intensive?

- Computing the best index means computing $D_{\Gamma}^T (x - D_{\Gamma} \alpha_{\Gamma})$ and taking the index of the smallest coefficient.
- Then we compute $\alpha_{\Gamma'}$ as $\alpha$ minimizing $\|x - D_{\Gamma'} \alpha\|^2$.
- Closed form solution:
  \[
  \alpha_{\Gamma'} = (D_{\Gamma'}^T D_{\Gamma'})^{-1} D_{\Gamma'} x
  \]

Making OMP faster

Invert quickly $D_{\Gamma'}^T D_{\Gamma'}$, knowing the inverse of $D_{\Gamma}^T D_{\Gamma}$.
Update of the inverse of $D^T D$ when appending a column $d$

- $u_1 \leftarrow D^T d$
- $u_2 \leftarrow (D^T D)^{-1} u_1$
- $u_3 \leftarrow d u_2$
- $A \leftarrow (D^T D)^{-1} + d u_2^T u_2$
- $s \leftarrow \frac{1}{d^T d - u_1^T u_2}$

**Updated inverse:**

$$
\begin{pmatrix}
A & -u_3 \\
-u_3^T & s
\end{pmatrix}
$$
An application of OMP: synthesizing terrains based on examples [Guérin et al. 2016]

A terrain is seen as a set of blended patches

Sparse Coding of signals
An application of OMP: synthesizing terrains based on examples [Guérin et al. 2016]

- Build a dictionary by decomposing a real-world elevation map into patches
- Decompose patches to synthesize on it
Various possible applications in the terrain setting

Sparse Coding of signals
Terrain Amplification

Creation of a multi-resolution dictionary

Remark
This does not work for natural images, but works for terrains (a simpler case than natural images).
Terrain Amplification

Creation of a multi-resolution dictionary

Remark
This does not work for natural images, but works for terrains (a simpler case than natural images).
Terrain Super-resolution

Result

Exemplar

Sketch

ε = 1 km

ε = 125 m

ε = 4 m
Another viewpoint on sparsity: the $\ell^1$ norm

- Using the $\ell^0$ or $\ell^p$, $p < 1$ yields nonconvex problems
- What about the $\ell^1$ norm?

$\ell^1$ norm (blue), $\ell^2$ norm (red)
Sparse Coding with the $\ell^1$ norm

Problem

Find $\alpha$ minimizing $\frac{1}{2} \| x - D\alpha \|_2^2 + \lambda \| \alpha \|_1$ this is called the LASSO problem
Sparse Coding with the $\ell^1$ norm

Problem
Find $\alpha$ minimizing $\frac{1}{2} \| x - D\alpha \|^2_2 + \lambda \| \alpha \|_1$ this is called the LASSO problem

Coordinate descent algorithm
- Select a coordinate index $j$ randomly
Sparse Coding with the $\ell^1$ norm

**Problem**

Find $\alpha$ minimizing $\frac{1}{2} \| x - D\alpha \|_2^2 + \lambda \| \alpha \|_1$ this is called the LASSO problem

**Coordinate descent algorithm**

- Select a coordinate index $j$ randomly
- update $\alpha[j]$ as the minimizer of $\| x - \sum_{l \neq j} \alpha[l] d_l - \alpha d_j \|_2^2 + \lambda |\alpha|$
Sparse Coding with the $\ell^1$ norm

Problem

Find $\alpha$ minimizing $\frac{1}{2} \| x - D\alpha \|_2^2 + \lambda \| \alpha \|_1$ this is called the LASSO problem

Coordinate descent algorithm

- Select a coordinate index $j$ randomly
- update $\alpha[j]$ as the minimizer of $\| x - \sum_{l \neq j} \alpha[l] d_l - \alpha d_j \|_2^2 + \lambda |\alpha|$
- $\alpha[j] \leftarrow S_\lambda(\alpha[j] + \frac{d_j^T(x - D\alpha)}{\| d_j \|_2^2})$
Sparse Coding with the $\ell^1$ norm

**Problem**

Find $\alpha$ minimizing $\frac{1}{2} \| x - D\alpha \|_2^2 + \lambda \| \alpha \|_1$ this is called the LASSO problem

**Coordinate descent algorithm**

- Select a coordinate index $j$ randomly
- update $\alpha[j]$ as the minimizer of $\| x - \sum_{l \neq j} \alpha[l]d_l - \alpha d_j \|_2^2 + \lambda \| \alpha \|
- $\alpha[j] \leftarrow S_\lambda(\alpha[j] + \frac{d_j^T(x - D\alpha)}{\|d_j\|_2^2})$

- $S_\lambda$ is the soft-thresholding operator $S_\lambda(u) = \text{sign}(u) \max(|u| - \lambda, 0)$
Outline

1 Visual Summary

2 Efficient Similar Patch Search

3 Sparse Coding of signals

4 Dictionary Learning

5 Optimal transportation meets nonlocal means

6 Half-toning
What is the dictionary $D$?

**Problem**

In the context of Image Processing and Synthesis, we only have access to a set of signals for which we want to build a dictionary.
What is the dictionary $D$?

**Problem**

In the context of Image Processing and Synthesis, we only have access to a set of signals for which we want to build a dictionary.

**Dictionary Learning Problem**

Given a set of signals $x_i$ for $i = 1 \cdots N$ in $\mathbb{R}^n$ we want to build a matrix $D \in \mathbb{R}^{n \times m}$ and coefficients $\alpha_i \in \mathbb{R}^m$ for $i = 1 \cdots n$ solving:

\[
\begin{align*}
\text{Minimize} & \quad \sum_{i=1}^{N} \| x_i - D \cdot \alpha_i \|_2^2 \\
\text{subject to} & \quad D \in \mathbb{R}^{n \times m}, \\
& \quad \| D_i \|_2 \leq 1, \\
& \quad \forall i = 1 \cdots N, \alpha_i \in \mathbb{R}^m, \\
& \quad \forall i = 1 \cdots N, \| \alpha_i \|_0 = s
\end{align*}
\]

$$(P_{D,\alpha,0})$$
Dictionary learning problems

- Still a nonconvex problem
- Common approach: alternate minimization
  - Fix the dictionary $D$ and compute the sparse decomposition $\alpha$
  - Fix the sparse decomposition $\alpha$ and compute $D$
Method of Optimal Directions (MOD) for $\ell^0$ constraints

- First introduced by Engan et al. [1999]
Method of Optimal Directions (MOD) for $\ell^0$ constraints

- First introduced by Engan et al. [1999]
- Step 1: Compute the sparse codes?
Method of Optimal Directions (MOD) for $\ell^0$ constraints

- First introduced by Engan et al. [1999]
- Step 1: Compute the sparse codes? MP, OMP, Iterative Hard Thresholding
Method of Optimal Directions (MOD) for $\ell^0$ constraints

- First introduced by Engan et al. [1999]
- Step 1: Compute the sparse codes? MP, OMP, Iterative Hard Thresholding
- Step 2: Update the dictionary
Assume all coefficients $\alpha_i$ are fixed, Problem $\left( P_{D,\alpha,0} \right)$ becomes

$$\text{Minimize} \sum_{i=1}^{N} \left\| x_i - D \cdot \alpha_i \right\|^2_2$$

subject to $D \in \mathbb{R}^{n \times m}$ and $\| D_i \|_2 \leq 1$.
MOD: Dictionary Update

- Assume all coefficients $\alpha_i$ are fixed, Problem $(P_{D,\alpha,0})$ becomes

$$\begin{align*}
\text{Minimize} & \quad \sum_{i=1}^{N} \left\| x_i - D \cdot \alpha_i \right\|_2^2 \\
\text{subject to} & \quad \|D_i\|_2 \leq 1
\end{align*}$$

- Convex objective function with convex constraints.
MOD: Dictionary Update

- Assume all coefficients $\alpha_i$ are fixed, Problem $(P_{D,\alpha,0})$ becomes

$$\text{Minimize} \sum_{i=1}^{N} \left\| x_i - D \cdot \alpha_i \right\|_2^2$$

- Convex objective function with convex constraints.
- Discarding the sparse constraint yields a least squares objective
Assume all coefficients $\alpha_i$ are fixed, Problem $(P_{D,\alpha,0})$ becomes

$$\text{Minimize} \sum_{i=1}^{N} \|x_i - D \cdot \alpha_i\|_2^2$$

with $D \in \mathbb{R}^{n \times m}$ and $\|D_i\|_2 \leq 1$.

- Convex objective function with convex constraints.
- Discarding the sparse constraint yields a least squares objective.
- Idea: Solve the least squares problem and project the solution onto the convex set of admissible solutions.
MOD: Dictionary Update

- Assume all coefficients $\alpha_i$ are fixed, Problem $(P_{D,\alpha,0})$ becomes

$$\text{Minimize} \sum_{i=1}^{N} \|x_i - D \cdot \alpha_i\|_2^2$$

- Convex objective function with convex constraints.
- Discarding the sparse constraint yields a least squares objective
- Idea: Solve the least squares problem and project the solution onto the convex set of admissible solutions

Bertsekas 1999

In general, solving the general problem and projecting the solution on the convex constraints set yields a poor solution
MOD: Dictionary Update

- Assume all coefficients $\alpha_i$ are fixed, Problem $(P_{D,\alpha,0})$ becomes

$$\text{Minimize } \sum_{i=1}^{N} \left\| x_i - D \cdot \alpha_i \right\|^2_2 \quad \text{subject to} \quad \|D_i\|_2 \leq 1$$

- Convex objective function with convex constraints.
- Discarding the sparse constraint yields a least squares objective.
- Idea: Solve the least squares problem and project the solution onto the convex set of admissible solutions.

Bertsekas 1999

Since the $\ell^0$ norm remains constant when a vector undergoes a nonzero rescaling, the projection is valid.
Dictionary Update

Least Squares Problem

Solve for $D$ in:

$$\text{Minimize}_{D \in \mathbb{R}^{m \times n}} \sum_{i=1}^{N} \| x_i - D \alpha_i \|_2^2$$

Setting $A = (\alpha_1 | \alpha_2 | \cdots | \alpha_N)$, $X = (x_1 | x_2 | \cdots | x_N)$, one has:

$$D = X A^T (A A^T)^{-1}$$

Projection on the constraint set: normalizing each column of $D$ if its norm is above 1.
Least Squares Problem

Solve for $D$ in:

$$\text{Minimize}_{D \in \mathbb{R}^{m \times n}} \sum_{i=1}^{N} \| x_i - D \alpha_i \|_2^2$$

- Compute the gradient and set to 0:
  $$\sum_{i=1}^{N} (x_i - D \alpha_i) \alpha_i^T = 0$$
Least Squares Problem

Solve for $D$ in:

\[
\text{Minimize } \min_{D \in \mathbb{R}^{m \times n}} \sum_{i=1}^{N} \| x_i - D \alpha_i \|^2_2
\]

- Compute the gradient+set to 0: \( \sum_{i=1}^{N} (x_i - D \alpha_i) \alpha_i^T = 0 \)
- \( D = (\sum_{i=1}^{N} x_i \alpha_i^T)(\sum_{i=1}^{N} \alpha_i \alpha_i^T)^{-1} \)
**Least Squares Problem**

Solve for $D$ in:

$$\text{Minimize}_{D \in \mathbb{R}^{m \times n}} \sum_{i=1}^{N} \| x_i - D \alpha_i \|^2_2$$

- Compute the gradient + set to 0: \( \sum_{i=1}^{N} (x_i - D \alpha_i) \alpha_i^T = 0 \)
- \( D = (\sum_{i=1}^{N} x_i \alpha_i^T)(\sum_{i=1}^{N} \alpha_i \alpha_i^T)^{-1} \)
- Setting \( A = (\alpha_1 \mid \alpha_2 \mid \cdots \mid \alpha_N) \), \( X = (x_1 \mid x_2 \mid \cdots \mid x_N) \)
  one has: \( D = X A^T (A A^T)^{-1} \)
Least Squares Problem

Solve for $D$ in:

$$\text{Minimize}_{D \in \mathbb{R}^{m \times n}} \sum_{i=1}^{N} \left\| x_i - D \alpha_i \right\|_2^2$$

- Compute the gradient and set to 0:
  $$\sum_{i=1}^{N} (x_i - D \alpha_i) \alpha_i^T = 0$$
- $$D = (\sum_{i=1}^{N} x_i \alpha_i^T) (\sum_{i=1}^{N} \alpha_i \alpha_i^T)^{-1}$$
- Setting $A = \begin{pmatrix} \alpha_1 & \alpha_2 & \cdots & \alpha_N \end{pmatrix}$, $X = \begin{pmatrix} x_1 & x_2 & \cdots & x_N \end{pmatrix}$
  one has: $D = X A^T (A A^T)^{-1}$
- Projection on the constraint set: normalizing each column of $D$ if its norm is above 1.
MOD algorithm for the $\ell^1$ norm

- The same principle as before can be used: alternate directions
MOD algorithm for the $\ell^1$ norm

- The same principle as before can be used: alternate directions
- Compute the sparse codes using coordinate gradient descent (or homotopy)
MOD algorithm for the $\ell^1$ norm

- The same principle as before can be used: alternate directions
- Compute the sparse codes using coordinate gradient descent (or homotopy)
- Update the dictionary by minimizing $\frac{1}{2} \sum_i \|x_i - D\alpha_i\|_2^2$
Results on point cloud denoising

<table>
<thead>
<tr>
<th>Original</th>
<th>LPF</th>
<th>RIMLS</th>
</tr>
</thead>
</table>

[Images of original point cloud and denoised versions]
Dictionary update for MOD-L1

Updating column $d_j$

Find $d$ s.t. $\|d\|_2 \leq 1$ minimizing $\sum_{i=1}^{n} \frac{1}{2} \|x_i - \sum_{l \neq j} \alpha_i(l)d_l - \alpha_i(j)d\|_2^2$

- Derive the final update for the dictionary.
K-SVD algorithm

- Still an alternating direction minimization method
K-SVD algorithm

- Still an alternating direction minimization method
- Goal: Incorporate the sparsity constraint also in the dictionary update step
Goal

- A set of training signals \( \{y_i\}_{i=1}^N \in \mathbb{R}^n \)
- Design a dictionary \( D \in \mathbb{R}^{n \times K} \) such that there exists \( x \in \mathbb{R}^k \) such that either \( y = Dx \) or \( y \approx Dx \) s.t. \( \|y - Dx\|_p \leq \varepsilon \)
- If \( n < K \) and \( D \) is full-ranged the solution must be constrained
  - \( \min_x \|x\|_0 \) s.t. \( y = Dx \)
  - \( \min_x \|x\|_0 \) s.t. \( \|y - Dx\|_2 \leq \varepsilon \)
- Design \( D \) in order to best fit the sparsity model imposed
An extension of K-means

- K-means search for the best possible representative enforcing that each representation uses a single atom with coefficient 1.
- K-SVD solves $\min_{D,X} \|Y - DX\|^2_F$ s.t. $\forall i \|x_i\|_0 \leq T_0$
- An iterative approach that alternates between two steps
  - **Sparse coding** of the examples based on the current dictionary
  - **Update of the dictionary** so as to better fit the data
- $Y \in \mathbb{R}^{n \times N}$: training samples, $X \in \mathbb{R}^{K \times N}$ matrix of coefficients
Sparse Coding stage

- $\mathbf{D}$ is fixed, compute the best representation $\mathbf{x}_i$ of sample $\mathbf{y}_i$
- Find $\mathbf{x}_i$ minimizing $\|\mathbf{y}_i - \mathbf{D}\mathbf{x}_i\|_2^2$ s.t. $\|\mathbf{x}_i\|_0 \leq T_0$
- Can be done using a pursuit algorithm (e.g. Orthogonal Matching Pursuit)
Dictionary Update stage

- The update will be done atom by atom.
- $\|Y - DX\|_F^2 = \|Y - \sum_{j=1}^{N} d_j x_T^j\|_F^2 = \|Y - \sum_{j=1,j\neq k}^{N} d_j x_T^j - d_k x_T^k\|_F^2$
- $E_k = Y - \sum_{j=1,j\neq k}^{N} d_j x_T^j$ error obtained by omitting atom $d_k$ in the decomposition
- Finally solve for:
  \[ \|E_k - d_k x_T^k\|_F \text{ w.r.t. } d_k, x_T^k \]
- Solve using SVD? if so sparsity not enforced.
Trick to enforce the sparsity

\( \omega_k = \{i|1 \leq i \leq K, x^k_T(i) \neq 0\} \)

- Restrict \( E_k \) and \( x^k_T \) to \( E^R_k \) and \( x^R_k \) by selecting only the columns of indices included in the support of \( x^k_T \)
- Use SVD to decompose \( E^R_k = U\Delta V^T \)
- Set \( d_k \) to be the first column of \( U \)
- Set \( x^R_k \) to be the first column of \( V \) multiplied by \( \Delta(1,1) \)
- the columns of \( D \) remain normalized and the support of the representations can not increase
Abstract

Application to the denoising of images

- noisy input image \( y \)
- Build \( \hat{D} \) and \((\hat{x}_i)\); the dictionary and representations of all patches of image \( y \)
- \((P_i(x))\); the set of all image \( x \) patches.
- \( \hat{D}\hat{\alpha}_i \) is the representation of patch \( P_i(y) \)
- Find \( x \) minimizing \( \lambda \|x - y\|^2 + \sum_i \|\hat{D}\hat{\alpha}_i - P_i(x)\|^2 \)
  - fidelity term
  - proximity of the reconstruction to the denoised patch
Can be tested on IPOL http://www.ipol.im/pub/algo/llm_ksvd
Learned dictionary

Dictionary learned from face patches
Train time 9.0s on 94500 patches
Learned Color dictionary

Dictionnaire initial

Dictionary Learning
Denoising via dictionary learning

noisy image
Denoising via dictionary learning

Denoising each channel separately
Denoising via dictionary learning

Denoising each channel separately
Denoising via dictionary learning

Denoising each channel separately (left) vs globally (right)
Comparison to NL-means

Original
Comparison to NL-means

Dictionary learning
Comparison to NL-means
Application: Point Cloud Compression


Self-similarity for compression

[Hubo et al. 2008]
- Cluster surface patches by similarity
- Replace each patch by a word of the codebook

Compression for rendering and not precision!

Patch-based self-similarity
Local patches capture local variations, comparing them underlines the self-similarity
Two samplings of the same shape
Pipeline

Original → Seeds and patches → Parameterization

Patch descriptions

1 2 n-1 n

Coefficients

(0.0; 1.5; 0.0)

(1.0; 0.0; 0.0)

(0.0; 0.5; -1.0)

(0.0; 1.0; -1.0)

Dictionary Learning
Working assumptions

- **Topological condition:** Surface covered by a set of topological disks centered around seeds.

- **Sampling condition:**
  $R$-neighborhood of a seed containing enough points.

- **Noise level:** Noise magnitude strictly below radius $R$.

- **Seeds selection:** anchors to define local patches
Self-similarity compression

- **Seeds** selection
- Local patches represented in a comparable way

- Patches decomposed upon a dictionary found by the K-SVD algorithm
- Final data: a set of seeds with local frames, a small dictionary and the (sparse) coefficients for each patch.
Further compression

- **Seeds**: kd-tree compression [Gandoin and Devillers, 2002].
- **Local parameterization** (3 Euler angles): predictive coding
- **Dictionary**: lossless compression.
- **Coefficients**: scalar quantization (increases sparsity) followed by entropy coding.
Controlling the error

Two types of errors are introduced:

- **Resampling error**
  ⇒ *Increasing the accuracy of the resampling pattern*

- **Compression error**
  ⇒ *Increasing the number of atoms in the dictionary*
Decompression

1. Decompress
   - seed positions
   - euler angles
   - dictionary $D$
   - coefficients $X$

2. Reconstruct the patches:
   $P_{rec} = D \ast X$

3. Consolidate the reconstructed point cloud in overlapping areas.
Anubis (9,9M pts) compressed to 0.96bpp; error = 0.01mm (0.003%)
St Matthew (93.5M pts) compressed to 0.83bpp; error = 0.05cm (0.002%)
Mire (16, 1M pts) compressed to 0.73bpp; error = 0.03mm (0.011%). Screened Poisson Reconstruction [Kazdhan, 2013]
Comparison with kd-tree coding. 4.83bpp against 0.6bpp in our method.
Lovers (15, 8M pts) compressed to 0.59 \(bpp\); error = 0.01mm (0.006%)
Breaking the working assumptions

Church (69, 9M pts) compressed to 0.76bpp; error = 1.48cm (0.005%)
rate/distortion performance compared to previous works
Outline

1 Visual Summary

2 Efficient Similar Patch Search

3 Sparse Coding of signals

4 Dictionary Learning

5 Optimal transportation meets nonlocal means

6 Half-toning
Generating half-toning images: *Dithering*

- For example: using only black ink for printing a grayscale image
- Color case: 4 inks instead of $255^3$ possible values.

Many of these slides are due to Raphaëlle Chaine.
Generating half-toning images: *Dithering*

For example: using only black ink for printing a grayscale image

Color case: 4 inks instead of $255^3$ possible values.

We will study only grayscale images.

Many of these slides are due to Raphaëlle Chaine.
Color Example
Half-toning in a nutshell

Half-toning Noir et Blanc

Create a binary approximation of a grayscale image which appears to be continuous.
Example

Original Image
(Wikipedia, user: Gerbrant)
Example

Quantification: *No continuity impression.*
(Image Wikipedia, user: Gerbrant)
Principe du Half-toning

Print black dots to give the illusion of gray values.
Principe du Half-toning

Print black dots to give the illusion of gray values.

- The human eye integrates the dots and perceives a flat colour.
Principe du Half-toning

Print black dots to give the illusion of gray values.

- The human eye integrates the dots and perceives a flat colour.

Problem

Choose a layout of points that minimize visual artefacts.
Dithering patterns

- The printed image is paved by dithering patterns
- Each dithering pattern contains a distribution of black and white dots.
- Each pattern gives a gray level corresponding to the ratio of black/white pixels.
Pattern example

Motifs 2x2
Pattern example

Motifs 3x3
First method

- Each pixel corresponds to a square pattern
- The pixel value is encoded by the corresponding pattern
Remark

Printing

- On a professionnal printing device 1200 dpi, 4*4 binary dots per pixel.
- On a 300dpi printer, only 1 binary dot pixel.
Choosing the layout of the dots

Goals

Layout algorithms aim at obtaining good gray values while minimizing the artefacts.

- Several algorithms exist (regular layout, irregular layout, dots centered or not centered in the patterns...).
Classical Halftoning

Dot areas are proportional to the image intensity.

$I(x,y)$

$P(x,y)$
Example

Newspaper Image

From New York Times, 9/21/99

Image T.A. Funkhouser
Dithering

- Random dithering
- Ordered dithering
- Error-diffusion dithering
Random Dithering

- Instead of using a fixed threshold, use a random one
Random Dithering

- Instead of using a fixed threshold, use a random one per pixel
Random Dithering

- Instead of using a fixed threshold, use a random one
Ordered Dithering

- The random thresholds are replaced by local schemes stored in matrices

For dithering patterns of size $2 \times 2$:

$$D_2 = \begin{pmatrix} 3 & 1 \\ 0 & 2 \end{pmatrix}$$
**Algorithm 1: Ordered Dithering**

**Input:** Grayscale image $I$, matrix $D_n (\mathbb{R}^{n \times n})$

**Output:** A binary image $J$

1. **for all pixels** $x, y$ **do**
2. $i = x \text{ modulo } n$;
3. $j = y \text{ modulo } n$;
4. **if** $I(x, y) > D(i, j)$ **then**
5. $J(x, y) = 1$;
6. **else**
7. $J(x, y) = 0$;
Ordered Dithering

- Bayer matrices for dithering

\[
D_n = \begin{bmatrix}
4D_{n/2} + D_2(1,1)U_{n/2} & 4D_{n/2} + D_2(1,2)U_{n/2} \\
4D_{n/2} + D_2(2,1)U_{n/2} & 4D_{n/2} + D_2(2,2)U_{n/2}
\end{bmatrix}
\]

\[
D_2 = \begin{bmatrix}
3 & 1 \\
0 & 2
\end{bmatrix}
\]

\[
D_4 = \begin{bmatrix}
15 & 7 & 13 & 5 \\
3 & 11 & 1 & 9 \\
12 & 4 & 14 & 6 \\
0 & 8 & 2 & 10
\end{bmatrix}
\]
Ordered Dithering

Often used for journal printing.
Dithering with error diffusion: Floyd-Steinberg

**Principle**

Distribute the error on neighboring pixels.

\[ \text{Error: } e = I(x, y) - \text{threshold}(I(x, y)) \]

Error distribution:

\[ \begin{align*}
&I(x, y+1) = I(x, y+1) + \alpha e \\
&I(x+1, y-1) = I(x+1, y-1) + \beta e \\
&I(x+1, y) = I(x+1, y) + \gamma e \\
&I(x+1, y+1) = I(x+1, y+1) + \delta e
\end{align*} \]

with \( \alpha + \beta + \gamma + \delta = 1 \)
Dithering with error diffusion: Floyd-Steinberg

**Principle**

Distribute the error on neighboring pixels.

- Threshold intensity value of $\text{threshold}(I(x, y))$
Dithering with error diffusion: Floyd-Steinberg

Principle

Distribute the error on neighboring pixels.

- Threshold intensity value of $\text{threshold}(I(x, y))$
- Error: $e = I(x, y) - \text{threshold}(I(x, y))$. 
Dithering with error diffusion: Floyd-Steinberg

**Principle**

Distribute the error on neighboring pixels.

- Threshold intensity value of $\text{threshold}(I(x, y))$
- Error: $e = I(x, y) - \text{threshold}(I(x, y))$.
- Error distribution:
  - $I(x, y + 1) = I(x, y + 1) + \alpha e$
  - $I(x + 1, y - 1) = I(x + 1, y - 1) + \beta e$
  - $I(x + 1, y) = I(x + 1, y) + \gamma e$
  - $I(x + 1, y + 1) = I(x + 1, y + 1) + \delta e$
  - with $\alpha + \beta + \gamma + \delta = 1$
Dithering with error diffusion: Floyd-Steinberg
Some methods use optimal transportation to generate density samplings [DeGoes et al. 2012]
Recommended Readings