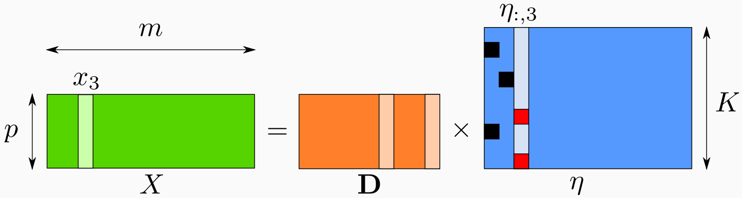
ECE 285

Image and video restoration

Chapter VII – Patch models and dictionary learning

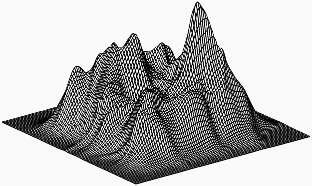
Charles Deledalle

May 31, 2019



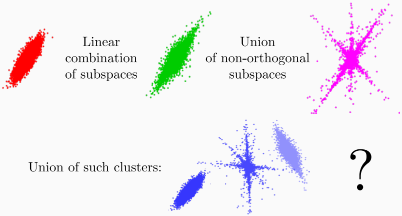
Motivations

- Modeling the distribution of images is difficult.
- Images lie in a complex and large dimensional space/manifold.
- Their distribution may be spread out on different clusters.



Underlying prior $x \mapsto p(x)$

$\approx$



Divide and conquer approach:

Break down images into small patches and model their distribution.

Motivations

Patches capture local context: geometry and texture.

Theoretical and experimental works on the primary visual cortex have shed new light on the importance of patch-level image coding.

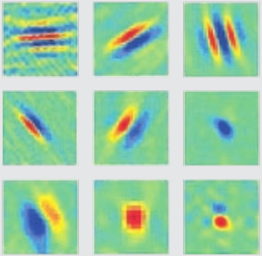
Biological observations (1/2)

[Olshausen et al., 1996]:

The receptive fields of cells in mammalian primary visual cortex are

1. spatially localized,
2. oriented,
3. bandpass.

receptive fields estimated by reverse correlation:



[Ringach, 2002]

Theoretical and experimental works on the primary visual cortex have shed new light on the importance of patch-level image coding.

Biological observations (2/2)

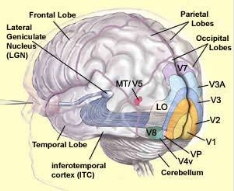
[Olshausen and Field, 2004]:

Neural responses in the primary cortex are:

- sparse,
- sparser thanks to interactions with other areas.

This sparse coding confers several advantages

- eases read out at subsequent levels,
- increases storage capacity in associative memories,
- saves energy.



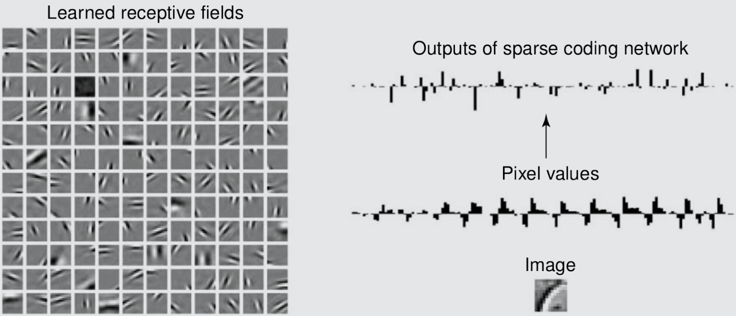
http://thebrain.mcgill.ca

Theoretical and experimental works on the primary visual cortex have shed new light on the importance of patch-level image coding.

Computational models of biological vision

[Olshausen et al. 1996, Olshausen and Field, 2004, Vinje and Gallant, 2000]:

Sparse coding of patches proposed to model the primary visual cortex:



Learning sparse representations

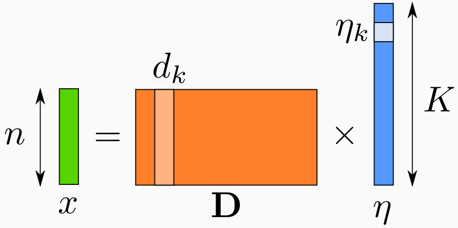
Dictionary learning problem

Reminder about sparse decomposition

Given a dictionary $\boldsymbol{D} = (d_1, d_2, \ldots, d_K) \in \mathbb{R}^{n\times K}$, with $K > n$ for redundancy, represent an image x as a sparse linear combination of the atoms

$$\eta^\star \in \operatorname*{argmin}_{\eta\in\mathbb{R}^K} \frac{1}{2} \underbrace{\|x - \sum_{k=1}^{K} \eta_k d_k\|_2^2}_{\text{data fit}} + \tau \underbrace{\sum_{k=1}^{K} |\eta_k|^p}_{\text{sparsity}}, \quad \tau \geqslant 0$$

$$= \operatorname*{argmin}_{\eta\in\mathbb{R}^K} \frac{1}{2} \underbrace{\|x - \boldsymbol{D}\eta\|_2^2}_{\text{data fit}} + \tau \underbrace{\|\eta\|_\rho^\rho}_{\text{sparsity}}, \quad \rho \geqslant 0$$



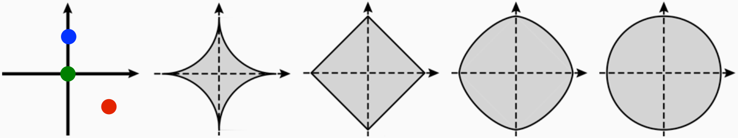
Dictionary learning problem

Reminder about sparse priors

- ℓ_ρ prior — $\|\eta\|_\rho^\rho = \sum_k |\eta_k|^\rho$
- convexity — $\rho \geqslant 1$
- sparsity — $\rho \leqslant 1$
- ℓ_0 prior — $\|\eta\|_0$ =number of non-zero elements
- ℓ_1 prior — $\|\eta\|_1 = \sum_k |\eta_k|$

2-dim vector:

$\|\boldsymbol{x}\|_0 = 0$	null image ●
$\|\boldsymbol{x}\|_0 = 1$	sparse image ●
$\|\boldsymbol{x}\|_0 = 2$	dense image ●



$\rho = 0$ $\quad$ $\rho = 1/2$ $\quad$ $\rho = 1$ $\quad$ $\rho = 3/2$ $\quad$ $\rho = 2$

(Source: G. Peyré)

Dictionary learning problem

Instead of choosing the dictionary: wavelet basis, derivative filters, ...
Can we learn it from a data-set $x_1, \dots, x_m$?

Sparsifying dictionary learning for images

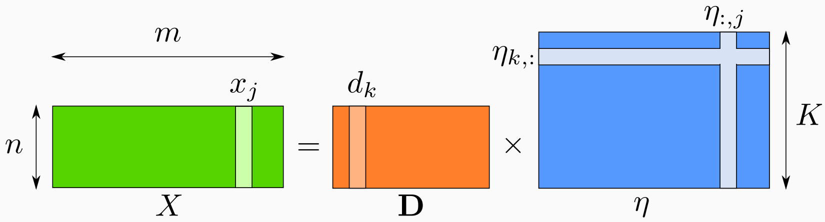
Find a dictionary $\boldsymbol{D} = (d_1, d_2, \dots, d_K) \in \mathbb{R}^{n \times K}$, with $K > n$ for redundancy, such that the data-set $X = (x_1, x_2, \dots, x_m) \in \mathbb{R}^{n \times m}$ can be represented by sparse linear combinations of the atoms

$$\boldsymbol{D}^\star \in \underset{\boldsymbol{D} \in \mathbb{R}^{n \times K}}{\operatorname{argmin}} \ \min_{\eta \in \mathbb{R}^{K \times m}} \ \frac{1}{2} \sum_{j=1}^{m} \underbrace{\| x_j - \sum_{k=1}^{K} \eta_{k,j} d_k \|_2^2}_{\text{data fit}} + \tau \underbrace{\sum_{j=1}^{m} \sum_{k=1}^{K} | \eta_{k,j} |^\rho}_{\text{sparsity}}$$

$$= \underset{\boldsymbol{D} \in \mathbb{R}^{n \times K}}{\operatorname{argmin}} \ \min_{\eta \in \mathbb{R}^{K \times m}} \ \frac{1}{2} \underbrace{\| X - \boldsymbol{D} \eta \|_F^2}_{\text{data fit}} + \tau \underbrace{\| \eta \|_\rho^\rho}_{\text{sparsity}} .$$

Idea: find a dictionary that sparsifies the data-set.

Dictionary learning problem



$D \in \mathbb{R}^{n\times K}$ has too many degrees of freedom!

It cannot be estimated properly, and even then, it does not fit in memory.

It would be feasible if the images were 8×8, but they are not.

Dictionary learning problem

Remarks: for wavelets and gradients, the atoms

❶ have limited support ❷ are stationary ❸ are overlapping

$$
\boldsymbol{D} = \begin{pmatrix}
\begin{matrix} | & | & & | \\ d_1 & d_2 & \dots & d_K \\ | & | & & | \end{matrix} & & & & & \\
& \begin{matrix} | & | & & | \\ d_1 & d_2 & \dots & d_K \\ | & | & & | \end{matrix} & & & & \\
& & \begin{matrix} | & | & & | \\ d_1 & d_2 & \dots & d_K \\ | & | & & | \end{matrix} & & & \\
& & & \ddots & & \\
& & & & \begin{matrix} | & | & & | \\ d_1 & d_2 & \dots & d_K \\ | & | & & | \end{matrix} & \\
& & & & & \begin{matrix} | & | & & | \\ d_1 & d_2 & \dots & d_K \\ | & | & & | \end{matrix}
\end{pmatrix}
$$

If we cannot learn D, can we learn the small atoms $d_1, \dots, \mathrm{d}_K$?

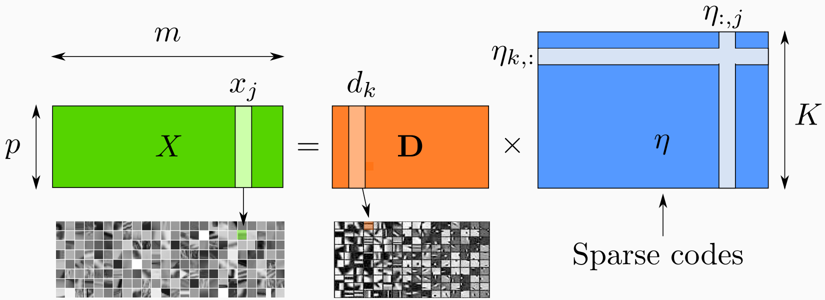
Ex: if the support is $p = 8 \times 8$ and $K = 512$, the learning problem is tractable even with a reasonable number K of training images x_k.

Dictionary learning problem

Sparsifying dictionary learning for patches

Find a dictionary **of patches** $\boldsymbol{D} = (d_1, d_2, \dots, d_K) \in \mathbb{R}^{p \times K}$, with $K > p$ for redundancy, such that the data-set **of patches** $X = (x_1, x_2, \dots, x_m) \in \mathbb{R}^{p \times m}$ can be represented by sparse linear combinations of the atoms

$$\boldsymbol{D}^\star \in \underset{\boldsymbol{D} \in \mathbb{R}^{p \times K}}{\operatorname{argmin}} \; \min_{\eta \in \mathbb{R}^{K \times m}} \; \frac{1}{2} \underbrace{\|X - \boldsymbol{D}\eta\|_F^2}_{\text{data fit}} + \tau \underbrace{\|\eta\|_{\rho,1}}_{\text{sparsity}} .$$



Dictionary learning problem

$$\boldsymbol{D}^\star \in \underset{\boldsymbol{D}\in\mathbb{R}^{p\times K}}{\operatorname{argmin}} \ \min_{\eta\in\mathbb{R}^{K\times m}} \frac{1}{2}\underbrace{\|X-\boldsymbol{D}\eta\|_F^2}_{\text{data fit}} + \tau \underbrace{\|\eta\|_\rho^\rho}_{\text{sparsity}} .$$

Optimization problem

- Add the constraint: $\|d_k\|_2 \leqslant 1$,

  Otherwise: $\boldsymbol{D} \to \infty$ and $\eta \to 0$.

- For $\rho \geqslant 1$:
  - Convex with respect to $\boldsymbol{D}$,
  - Convex with respect to η,
  - **Non-convex** with respect to $(\boldsymbol{D}, \eta)$.

- For $\rho = 0$:
  - Convex with respect to $\boldsymbol{D}$,
  - Non-convex with respect to η,
  - **Non-convex** with respect to $(\boldsymbol{D}, \eta)$.

Dictionary learning problem with k-SVD

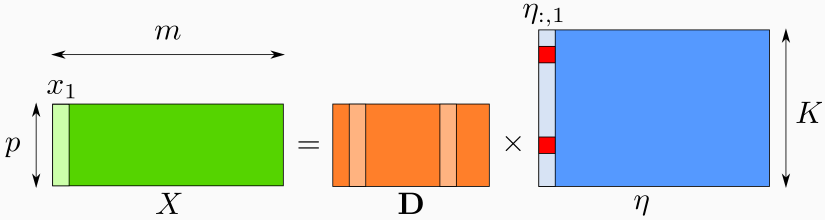
$$\boldsymbol{D}^\star \in \underset{\boldsymbol{D}\in\mathbb{R}^{p\times K}}{\operatorname{argmin}} \min_{\eta\in\mathbb{R}^{K\times m}} \frac{1}{2}\underbrace{\|X - \boldsymbol{D}\eta\|_F^2}_{\text{data fit}} + \tau \underbrace{\|\eta\|_0}_{\text{sparsity}} \quad \text{subject to} \quad \|d_k\|_2 \leqslant 1$$

k-SVD: Greedy algorithm for $\rho = 0$ (1/2) **[Aharon *et al.*, 2006]**

1. Initialize $\boldsymbol{D}$ with normalized columns $\|d_k\|_2 = 1$.
2. **Sparse-coding stage:** fix $\boldsymbol{D}$ and solve for each $1 \leqslant j \leqslant m$

$$\eta^\star_{:,j} \in \min_{\eta\in\mathbb{R}^K} \frac{1}{2}\underbrace{\|x_j - \boldsymbol{D}\eta\|_F^2}_{\text{data fit}} + \tau \underbrace{\|\eta\|_0}_{\text{sparsity}}$$

with matching pursuit or orthogonal matching pursuit (see previous class).



Dictionary learning problem with k-SVD

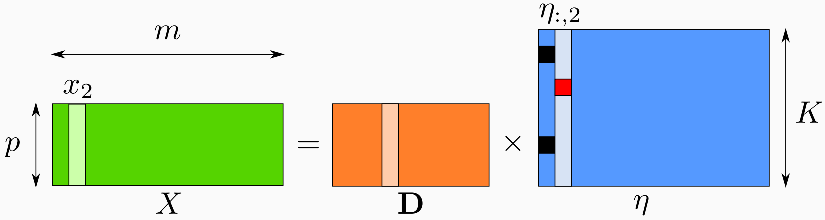
$$\boldsymbol{D}^\star \in \underset{\boldsymbol{D}\in\mathbb{R}^{p\times K}}{\operatorname{argmin}} \min_{\eta\in\mathbb{R}^{K\times m}} \frac{1}{2}\underbrace{\|X - \boldsymbol{D}\eta\|_F^2}_{\text{data fit}} + \tau \underbrace{\|\eta\|_0}_{\text{sparsity}} \quad \text{subject to} \quad \|d_k\|_2 \leqslant 1$$

k-SVD: Greedy algorithm for $\rho = 0$ (1/2) **[Aharon *et al.*, 2006]**

1. Initialize $\boldsymbol{D}$ with normalized columns $\|d_k\|_2 = 1$.
2. **Sparse-coding stage:** fix $\boldsymbol{D}$ and solve for each $1 \leqslant j \leqslant m$

$$\eta^\star_{:,j} \in \min_{\eta\in\mathbb{R}^K} \frac{1}{2}\underbrace{\|x_j - \boldsymbol{D}\eta\|_F^2}_{\text{data fit}} + \tau \underbrace{\|\eta\|_0}_{\text{sparsity}}$$

with matching pursuit or orthogonal matching pursuit (see previous class).



Dictionary learning problem with k-SVD

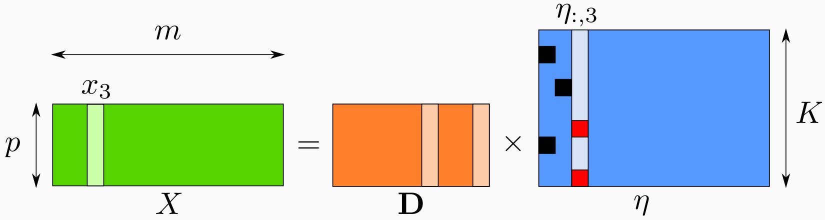
$$\boldsymbol{D}^\star \in \underset{\boldsymbol{D}\in\mathbb{R}^{p\times K}}{\operatorname{argmin}} \min_{\eta\in\mathbb{R}^{K\times m}} \frac{1}{2} \underbrace{\|X - \boldsymbol{D}\eta\|_F^2}_{\text{data fit}} + \tau \underbrace{\|\eta\|_0}_{\text{sparsity}} \quad \text{subject to} \quad \|d_k\|_2 \leqslant 1$$

k-SVD: Greedy algorithm for $\rho = 0$ (1/2) **[Aharon *et al.*, 2006]**

1. Initialize $\boldsymbol{D}$ with normalized columns $\|d_k\|_2 = 1$.
2. **Sparse-coding stage:** fix $\boldsymbol{D}$ and solve for each $1 \leqslant j \leqslant m$

$$\eta^\star_{:,j} \in \min_{\eta\in\mathbb{R}^K} \frac{1}{2} \underbrace{\|x_j - \boldsymbol{D}\eta\|_F^2}_{\text{data fit}} + \tau \underbrace{\|\eta\|_0}_{\text{sparsity}}$$

with matching pursuit or orthogonal matching pursuit (see previous class).



Dictionary learning problem with k-SVD

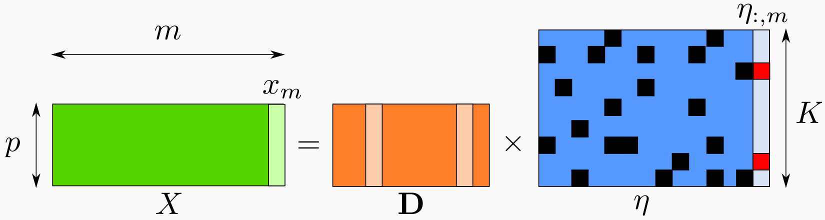
$$\boldsymbol{D}^\star \in \underset{\boldsymbol{D}\in\mathbb{R}^{p\times K}}{\operatorname{argmin}} \min_{\eta\in\mathbb{R}^{K\times m}} \frac{1}{2}\underbrace{\|X - \boldsymbol{D}\eta\|_F^2}_{\text{data fit}} + \tau \underbrace{\|\eta\|_0}_{\text{sparsity}} \quad \text{subject to} \quad \|d_k\|_2 \leqslant 1$$

k-SVD: Greedy algorithm for $\rho = 0$ (1/2) **[Aharon *et al.*, 2006]**

1. Initialize $\boldsymbol{D}$ with normalized columns $\|d_k\|_2 = 1$.
2. **Sparse-coding stage:** fix $\boldsymbol{D}$ and solve for each $1 \leqslant j \leqslant m$

$$\eta^\star_{:,j} \in \min_{\eta\in\mathbb{R}^{K}} \frac{1}{2}\underbrace{\|x_j - \boldsymbol{D}\eta\|_F^2}_{\text{data fit}} + \tau \underbrace{\|\eta\|_0}_{\text{sparsity}}$$

with matching pursuit or orthogonal matching pursuit (see previous class).



Dictionary learning with k-SVD

k-SVD: Greedy algorithm for $\rho = 0$ (2/2) **[Aharon *et al.*, 2006]**

❸ **Dictionary update:** for all columns $1 \leqslant k \leqslant K$

- Compute the residual without using the current atom d_k:

$$\boldsymbol{E}_k = X - \sum_{l \neq k} d_l \eta_{l,:} = X - (\boldsymbol{D}\eta - d_k \eta_{k,:})$$

- $\boldsymbol{E}_k^R$: pick only the columns j of $\boldsymbol{E}_k$ for patches x_j using atom d_k,
- Update d_k and $\eta_{k,:}$ by finding the best rank 1 approximation

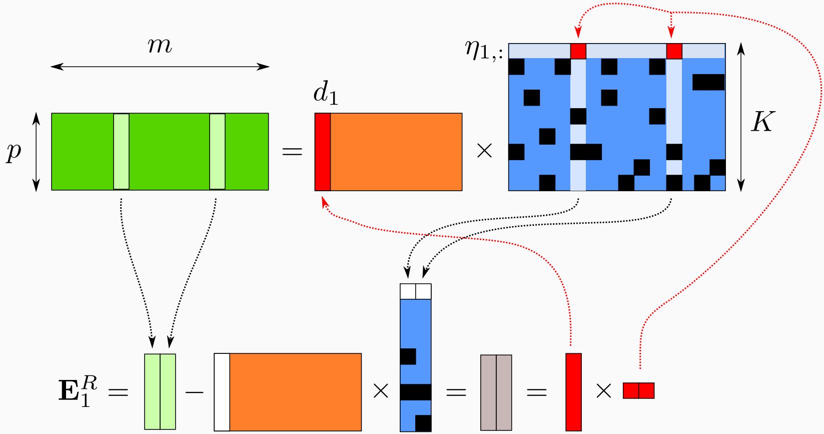
$$\boldsymbol{E}_k^R \approx d_k \eta_{k,:} \quad \text{subject to} \quad \|d_k\|_2 = 1$$

Use reduces SVD for rank 1 matrices:

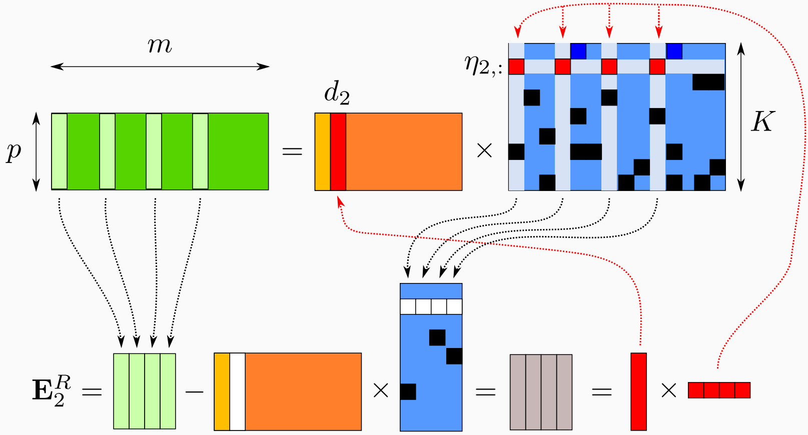
$$\boldsymbol{E}_k^R = USV^T \quad \Rightarrow \quad d_k = U_{:,1} \quad \text{and} \quad \eta_{k,:} = S_{1,1} V_{:,1}$$

- Return to step 2 until convergence.

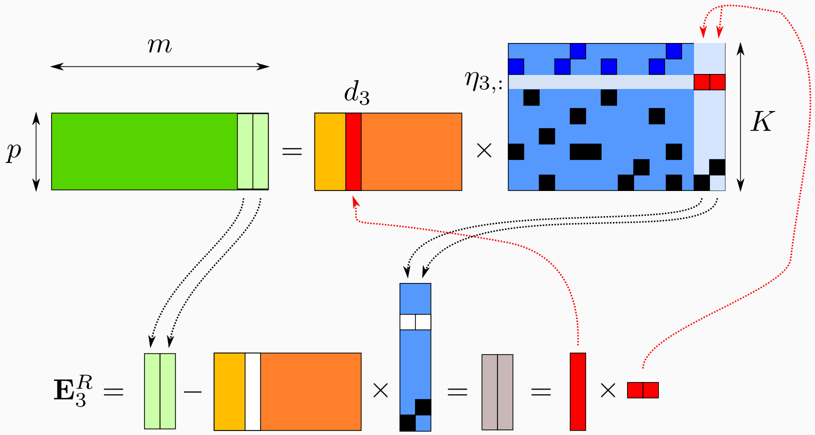
$\longrightarrow$ $k \times$ **SVD are performed at each iteration.**



Update for atom $k = 1$

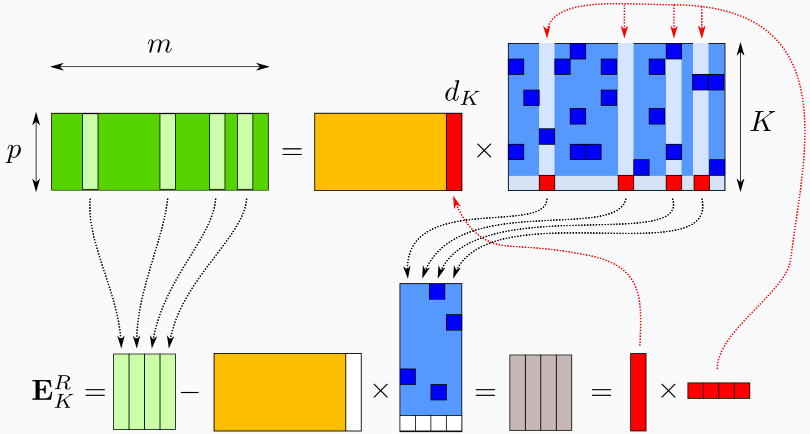


Update for atom $k = 2$



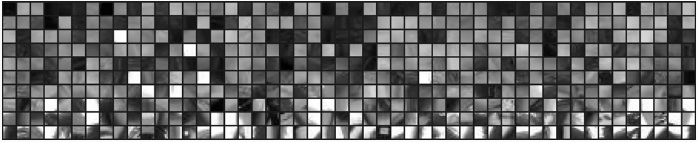
Update for atom $k = 3$

Dictionary learning with k-SVD

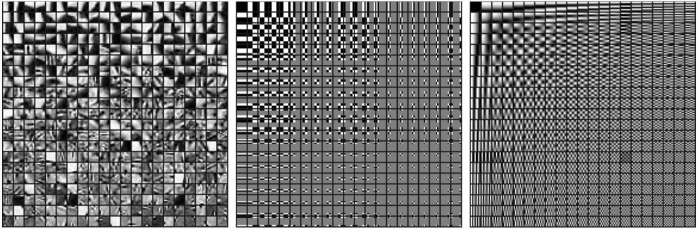


Update for atom $k = K$

Dictionary learning with k-SVD



A collection of 500 random patches (8x8) that were used for training, sorted by their variance.



(a) (b) (c)

(a) The learned dictionary. Its elements are sorted in an ascending order of their variance and stretched to maximal range for display purposes. (b) The overcomplete separable Haar dictionary and (c) the over complete DCT dictionary are used for comparison.

k-SVD based restoration (1/3) **[Elad *et al.*, 2006]**

$$\min_{\substack{x \in \mathbb{R}^n \\ \eta_1, \ldots, \eta_n \in \mathbb{R}^K}} \quad \frac{1}{2\sigma^2} \underbrace{\|\boldsymbol{H}x - y\|_2^2}_{\text{data fit}} + \sum_{i=1}^{n} \left[\frac{\beta}{2} \underbrace{\|\mathcal{P}_i x - \boldsymbol{D}\eta_i\|_2^2}_{\text{patch approximation}} + \underbrace{\tau \|\eta_i\|_0}_{\text{sparsity}} \right]$$

(Half-Quadratic Splitting)

- $x \in \mathbb{R}^n$: unknown image,
- $y = \boldsymbol{H}x + w \in \mathbb{R}^q$: observed image with $w \sim \mathcal{N}(0, \sigma^2 \mathrm{Id}_q)$,
- $\boldsymbol{H} \in \mathbb{R}^{q \times n}$: blur, super-resolution, Radon transform...
- $\mathcal{P}_i \in \mathbb{R}^{p \times n}$: extract a patch of size p around pixel with index i,
- $\boldsymbol{D} \in \mathbb{R}^{p \times K}$: learned patch dictionary,
- $\eta_i \in \mathbb{R}^K$: sparse code for patch with index i,
- $\beta > 0, \tau > 0$: hyper-parameters.

Look for an image such that all its patches are well explained by sparse linear combinations of learned atoms.

Dictionary learning with k-SVD

$$\min_{\substack{x \in \mathbb{R}^n \\ \eta_1, \ldots, \eta_n \in \mathbb{R}^K}} \quad \frac{1}{2\sigma^2} \underbrace{\|\boldsymbol{H}x - y\|_2^2}_{\text{data fit}} + \sum_{i=1}^{n} \left[\frac{\beta}{2} \underbrace{\|\mathcal{P}_i x - \boldsymbol{D}\eta_i\|_2^2}_{\text{patch approximation}} + \underbrace{\tau \|\eta_i\|_0}_{\text{sparsity}} \right]$$

k-SVD based restoration (2/3) **[Elad *et al.*, 2006]**

Alternate minimization:

1. Initialize x, and repeat steps 2 and 3 until convergence,
2. **Sparse coding:** fix x and solve for all index $1 \leqslant i \leqslant n$

$$\underset{\eta_i \in \mathbb{R}^K}{\operatorname{argmin}} \quad \frac{\beta}{2} \underbrace{\|\mathcal{P}_i x - \boldsymbol{D}\eta_i\|_2^2}_{\text{patch approximation}} + \underbrace{\tau \|\eta_i\|_0}_{\text{sparsity}}$$

with matching pursuit or orthogonal matching pursuit (see previous class).

$$\min_{\substack{x \in \mathbb{R}^n \\ \eta_1, \ldots, \eta_n \in \mathbb{R}^K}} \quad \frac{1}{2\sigma^2} \underbrace{\|\boldsymbol{H}x - y\|_2^2}_{\text{data fit}} + \sum_{i=1}^n \left[\frac{\beta}{2} \underbrace{\|\mathcal{P}_i x - \boldsymbol{D}\eta_i\|_2^2}_{\text{patch approximation}} + \underbrace{\tau \|\eta_i\|_0}_{\text{sparsity}} \right]$$

k-SVD based restoration (3/3) [Elad *et al.*, 2006]

Alternate minimization:

❸ **Patch reprojection:** for all η_i and solve for x

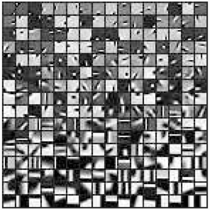
$$x^\star \in \operatorname*{argmin}_{x \in \mathbb{R}^n} \quad \frac{1}{2\sigma^2} \underbrace{\|\boldsymbol{H}x - y\|_2^2}_{\text{data fit}} + \sum_{i=1}^n \left[\frac{\beta}{2} \underbrace{\|\mathcal{P}_i x - \boldsymbol{D}\eta_i\|_2^2}_{\text{patch approximation}} \right]$$

$$= \left(\boldsymbol{H}^* \boldsymbol{H} + \sigma^2 \beta \sum_{i=1}^n \mathcal{P}_i^* \mathcal{P}_i \right)^{-1} \left(\boldsymbol{H}^* y + \sigma^2 \beta \sum_{i=1}^n \mathcal{P}_i^* \boldsymbol{D}\eta_i \right)$$

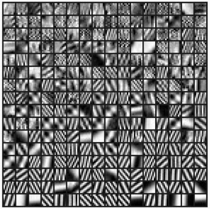
- If $\boldsymbol{H} = \mathrm{Id}_n$: average overlapping patches $\boldsymbol{D}\eta_i$ with the noisy image y
- Otherwise, solved by conjugate gradient, or efficiently depending on $\boldsymbol{H}$.

Dictionary learning with k-SVD

Globally trained dictionary.

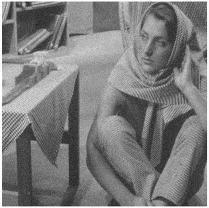


Adaptively trained dictionary.



Original Image

Noisy Image (22.1307 dB, σ=20)



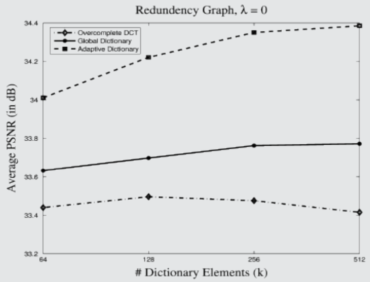
Denoised Image Using
Global Trained Dictionary (28.8528 dB)

Denoised Image Using
Adaptive Dictionary (30.8295 dB)

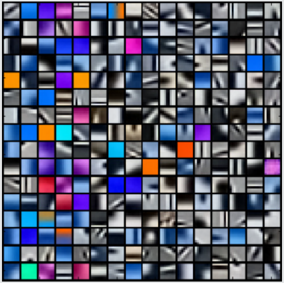
k-SVD for denoising

- **Quality improvement:** learned the dictionary on the noisy image itself,
- **Speed-up:** only once: sparse coding + reprojection (do not iterate).

k-SVD: Importance of redundancy



Color k-SVD [Mairal *et al.*, 2008]



$8 \times 8 \times 3$ patches

Related works:

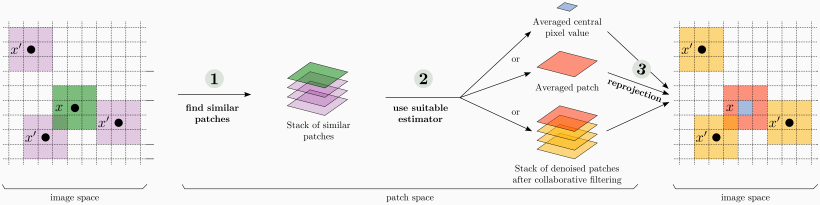
- Non-negative k-SVD [Aharon *et al.*, 2005].
- Color k-SVD [Mairal *et al.*, 2008].
- Analysis k-SVD [Rubinstein *et al.*, 2013].

Sparsity with collaborative filtering

Sparsity with collaborative filtering

Motivations

- k-SVD:
  - Patches are denoised independently,
  - Use non-linear shrinkages to create sparsity,
  - Use redundant learned dictionaries.
- Non-local Bayes:
  - Denoise similar patches together,
  - Use linear shrinkages (LMMSE),
  - Use (non-local) orthogonal PCA basis.

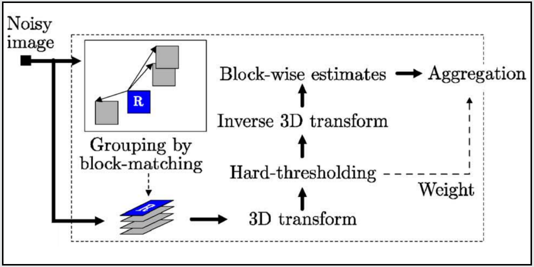


Idea: use sparsity on stacks of similar patches.

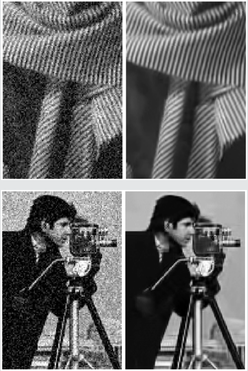
Sparsity with collaborative filtering – BM3D

BM3D: Block-matching and 3D filtering [Dabov et al., 2007]

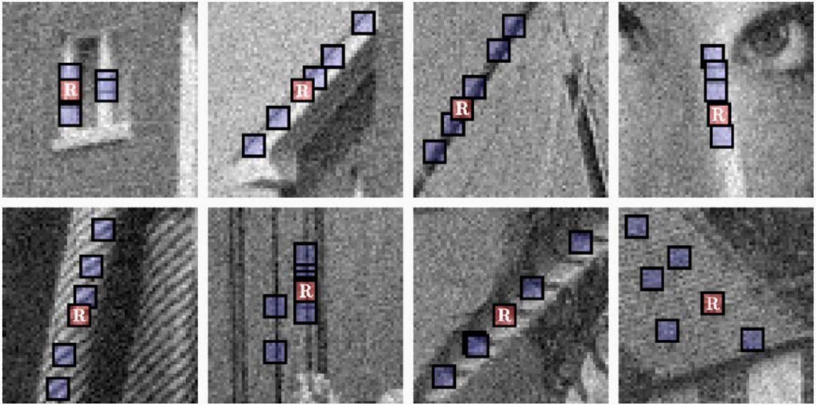
- Build groups of similar patches,
- Apply sparsifying 3D transform,
- Denoise each group (thresholding or LMMSE),
- Reproject/Aggregate overlapping patches.



Some results:

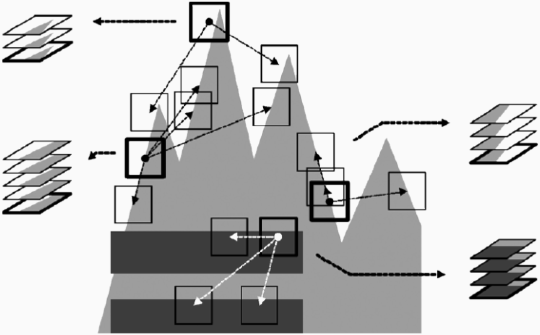


Grouping by matching



R is a targeting patch, other patches are grouped with this patch by similarity (Euclidean distance).

Grouping by matching



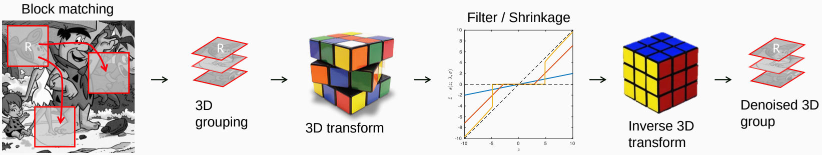
Non-locality: patches that are far apart can be stacked together.

Sparsity with collaborative filtering – BM3D

Grouping for collaborative filtering

- As groups contain similar patches:
  - intrapatch correlation: peculiarity of natural images,
  - interpatch correlation: results of grouping by similarity,

  $\Rightarrow$ **highly sparse representation**.



Collaborative filtering: $\left\{ \right.$

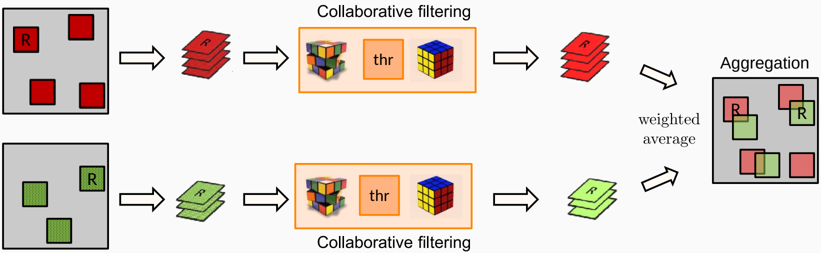
- reveals finest details shared by similar patches,
- preserves unique features of each patch.

Aggregation

- Each pixel gets multiple estimates from different groups
- Naive approach: average all estimates

  ... not all estimates are as good.
- Give higher weights to more reliable estimates

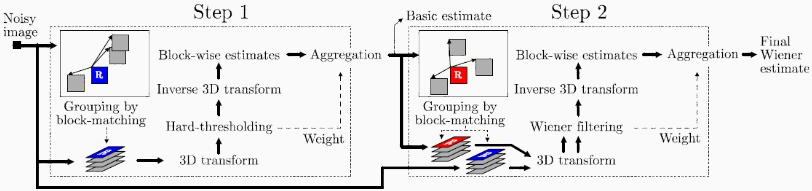
  ... measured according to their sparsity.



Sparsity with collaborative filtering – BM3D

Two steps filtering

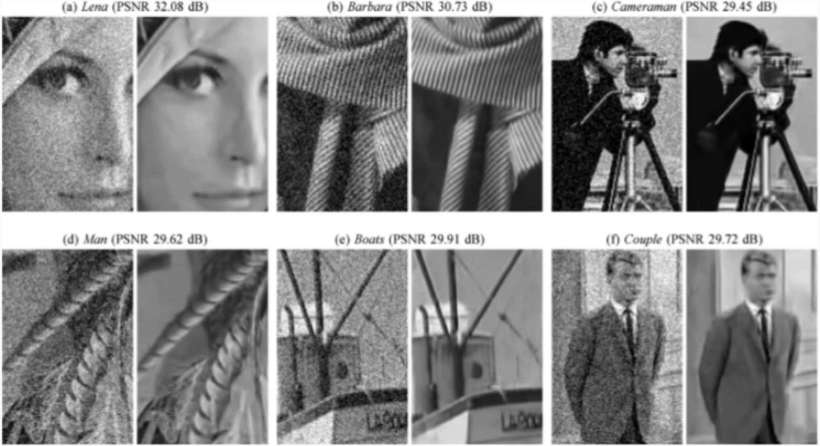
- Noise may result in poor matching ⇒ degrades denoising performance.
- As for NL-Bayes, use two stages. At the second stage:
  - Build stacks based on the similarity of pre-denoised patches,
  - Use pre-denoised stacks to refine the shrinkage.



1. **Step 1:** Shrinkage: Hard thresholding with fixed threshold
   3D trans.: Bi-orthogonal wavelets in space + Haar in 3rd dim.

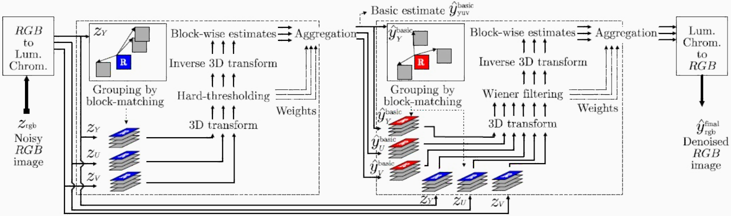
2. **Step 2:** Shrinkage: LMMSE with signal variances deduced form step 1.
   3D trans.: DCT in space + Haar in 3rd dim.

Sparsity with collaborative filtering – BM3D

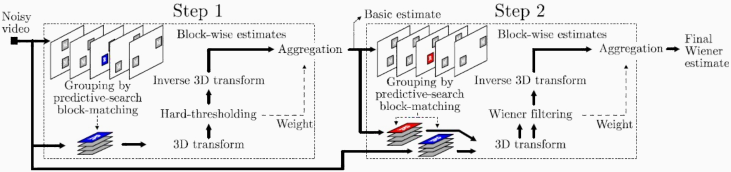


BM3D provides impressive results.
Since 2007, denoising results have not <u>significantly</u> improved.

Adaptations to color and videos.



Color image denoising with BM3D [Dabov, 2007]



Video denoising with BM3D [Dabov, 2007]

Plug-and-play ADMM with BM3D

$$\min_{x\in\mathbb{R}^n}\left\{E(x) = \underbrace{\frac{1}{2\sigma^2}\|y - \boldsymbol{H}x\|_2^2}_{F(x)} + R(x)\right\}$$

Adaptation to inverse-problems with Plug-and-play ADMM (1/2)

- Reminder: **ADMM algorithm** reads, for $\gamma > 0$, as

$$\begin{aligned} x^{k+1} &= \mathrm{Prox}_{\gamma F}(\tilde{x}^k + d^k) && (1)\\ \tilde{x}^{k+1} &= \mathrm{Prox}_{\gamma R}(x^{k+1} - d^k) && (2)\\ d^{k+1} &= d^k - x^{k+1} + \tilde{x}^{k+1} \end{aligned}$$

$$(1) \Rightarrow x^{k+1} = (\sigma^2 \mathrm{Id}_n + \gamma \boldsymbol{H}^*\boldsymbol{H})^{-1}(\sigma^2(\tilde{x}^k + d^k) + \gamma \boldsymbol{H}^* y) \quad \textbf{(inversion)}$$

$$(2) \Rightarrow \tilde{x}^{k+1} = \underset{z}{\mathrm{argmin}}\ \frac{1}{2}\|z - (x^{k+1} - d^k)\|_2^2 + \gamma R(z) \quad \textbf{(denoising)}$$

- Convergence when R is convex.
- Convergence when R is non-convex in some cases [Hong *et al.* 2016].

Plug-and-play ADMM with BM3D

$$\min_{x \in \mathbb{R}^n} \left\{ E(x) = \underbrace{\frac{1}{2\sigma^2} \|y - \boldsymbol{H} x\|_2^2}_{F(x)} + R(x) \right\}$$

Adaptation to inverse-problems with Plug-and-play ADMM (2/2)

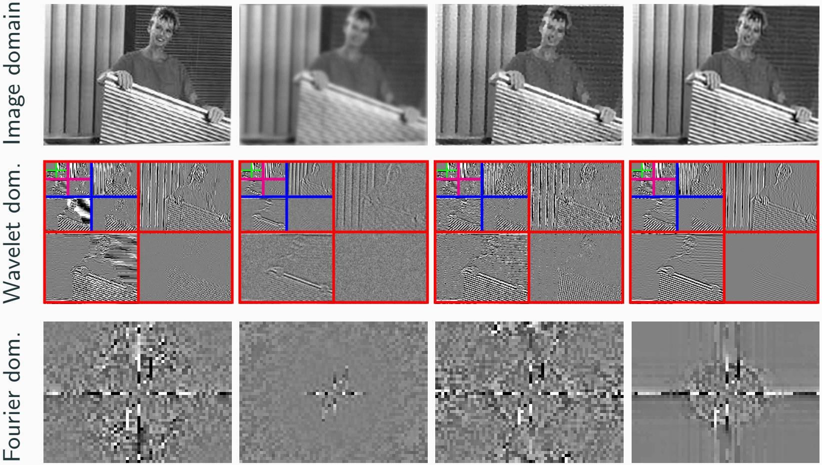
- **Plug-and-play ADMM** [Venkatakrishnan *et al.* 2013]
- Use any Gaussian denoiser for the denoising step

$$\text{ex:} \quad (2) \Rightarrow \tilde{x}^{k+1} = \text{BM3D}(x^{k+1} - d^k, \gamma)$$

- The regularization R is implicit.
- Convergence in some cases [Chan *et al.* 2016].
- Non-Gaussian noise: adapt F in (1) but (2) remains a Gaussian denoiser. [Rond *et al.* 2015, Deledalle *et al.* 2017].

Simple solution allowing to use any of the many and very efficient Gaussian denoisers to solve different kinds of image restoration problems.

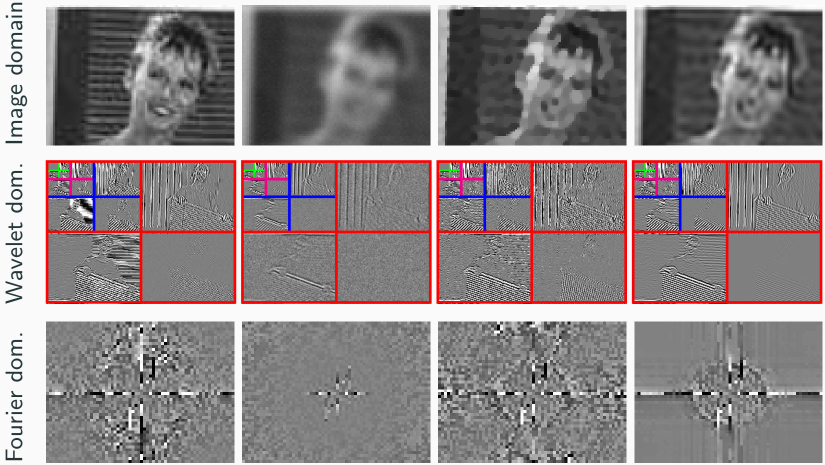
Plug-and-play ADMM with BM3D



(a) Original image (b) Blurry image (c) Total-Variation (d) BM3D

Lost frequencies are recovered. Spatial contents and scales as well.

Plug-and-play ADMM with BM3D



(a) Original image (b) Blurry image (c) Total-Variation (d) BM3D

Lost frequencies are recovered. Spatial contents and scales as well.

Sparsity and patch similarity – Denoising with NLSM

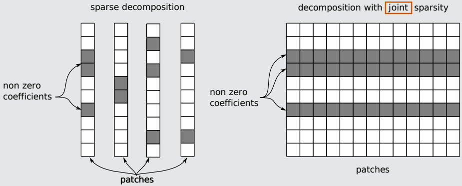
BM3D uses fix dictionaries. Can we learn them *à la* k-SVD?

Non-local sparse model (NLSM) **[Mairal, 2009]**

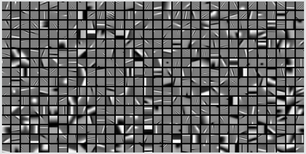
Learn a dictionary of patches:

- Use group sparsity for similar patches,
- Force similar patches to use the same atoms (joint sparsity).

Then denoise each patch by sparse coding.



Some results:



Expected patch log-likelihood (EPLL)

Expected patch log-likelihood (EPLL)

Expected patch log-likelihood (1/2) **[Zoran & Weiss, 2011]**

- Use MAP with prior expressed on patches

$$\min_{x \in \mathbb{R}^n} \quad \frac{1}{2\sigma^2} \underbrace{\|\boldsymbol{H}x - y\|_2^2}_{\text{data fit}} + \sum_{i=1}^{n} - \underbrace{\log p(\mathcal{P}_i x)}_{\text{patch prior}}$$

- $x \in \mathbb{R}^n$: unknown image,
- $y = \boldsymbol{H}x + w \in \mathbb{R}^q$: observed image with $w \sim \mathcal{N}(0, \sigma^2 \mathrm{Id}_q)$,
- $\boldsymbol{H} \in \mathbb{R}^{q \times n}$: blur, super-resolution, Radon transform...
- $\mathcal{P}_i \in \mathbb{R}^{p \times n}$: extract a patch of size p around pixel with index i,

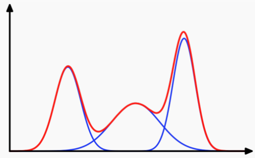
Look for an image such that all its patches are well explained by the patch prior.

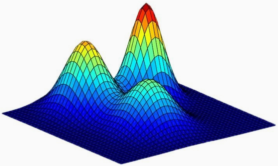
Expected patch log-likelihood (2/2) **[Zoran & Weiss, 2011]**

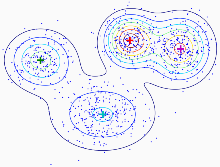
- Prior for a path $z_i = \mathcal{P}_i x \in \mathbb{R}^p$, a Gaussian Mixture Model (GMM):

$$p(z_i) = \sum_{k=1}^{K} w_k \mathcal{N}(z_i; \mu_k, \Sigma_k),$$

- $w_k > 0$: weights of Gaussian component k ($\sum_k w_k = 1$),
- $\mu_k \in \mathbb{R}^p$: mean of Gaussian component k,
- $\Sigma_k \in \mathbb{R}^{p \times p}$: covariance matrix of Gaussian component k.







Represent the patch distribution by a superposition of ellipsoids.

Expected patch log-likelihood (EPLL)

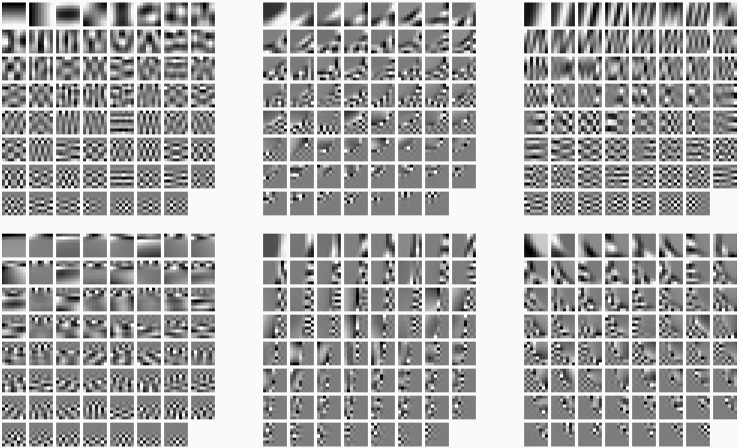
Learning step

- **Fit the distribution** on a large dataset of clean patches:

$$\begin{aligned} \text{input:} \quad & x_1, x_2, \ldots, x_m \quad \text{clean patches} \\ \text{output:} \quad & w_k, \mu_k, \Sigma_k \quad \text{for all} \quad 1 \leqslant k \leqslant K \end{aligned}$$

- **Standard choice**:
  - Dataset of $m =$2,000,000 patches,
  - Patch size $p = 8 \times 8$,
  - Number of clusters $K = 200$.
- Use **Expectation-Maximization algorithm** [Dempster, 1977]
  - Iterative algorithm similar to K-means,
  - Greedy (maximizes the likelihood at each iteration),
  - Converges to a local optimum (depending on the initialization).

Expected patch log-likelihood (EPLL)



Eigenvectors of 6 (among 200) covariance matrices of the learned GMM.

Some look like Fourier atoms while others model textures, edges or other structures at different scales and orientations.

Expected patch log-likelihood (EPLL)

$$\min_{x \in \mathbb{R}^n} \quad \frac{1}{2\sigma^2} \|\boldsymbol{H}x - y\|_2^2 + \sum_{i=1}^{n} - \log p(\mathcal{P}_i x)$$

Optimization by Half-Quadratic Splitting

- Use **Half-Quadratic Splitting** (as done in k-SVD)

$$\min_{\substack{x \in \mathbb{R}^n \\ z_1, \ldots, z_n \in \mathbb{R}^p}} \quad \frac{1}{2\sigma^2} \underbrace{\|\boldsymbol{H}x - y\|_2^2}_{\text{data fit}} + \sum_{i=1}^{n} \left[\frac{\beta}{2} \underbrace{\|\mathcal{P}_i x - z_i\|_2^2}_{\text{patch approximation}} - \underbrace{\log p(z_i)}_{\text{patch prior}} \right]$$

  with $\beta > 0$ an hyper-parameter.

- Alternate the minimization for all z_i and x.

- Increase β after each iteration.

Expected patch log-likelihood (EPLL)

Greedy alternate minimization

- Repeat steps 1 and 2 (usually 5 iterations are enough):

1. **Fix x and optimize for all patch z_i:**

$$\min_{z_i \in \mathbb{R}^p} \quad \frac{\beta}{2}\|\mathcal{P}_i x - z_i\|_2^2 + \sum_{i=1}^{n} -\log\left(\sum_{k=1}^{K} w_k \mathcal{N}(z_i; \mu_k, \Sigma_k)\right)$$

   - Prior is multi-modal: non-convex optimization problem.
   - Look for the most likely Gaussian component $k_i^\star$ given z_i.
   - Performs LMMSE with this Gaussian prior $\mathcal{N}(\mu_{k_i^\star}, \Sigma_{k_i^\star})$.

2. **Fix z_i and optimize for the image x:**

$$\min_{x \in \mathbb{R}^n} \quad \frac{1}{2\sigma^2}\|\boldsymbol{H}x - y\|_2^2 + \frac{\beta}{2}\sum_{i=1}^{n}\|\mathcal{P}_i x - z_i\|_2^2$$

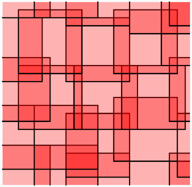
   - Linear solution: same patch reprojection as for k-SVD (see slide 20).

Expected patch log-likelihood (EPLL)

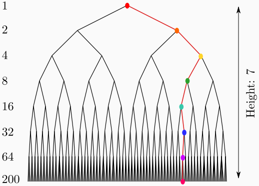
Fast EPLL **[Parameswaran *et al.*, 2017]**

1. Process only 3% of the patches at each iteration (chosen randomly),
2. Use a binary search tree to match for the best Gaussian component,
3. Approximate smallest eigenvalues of the covariance matrices.

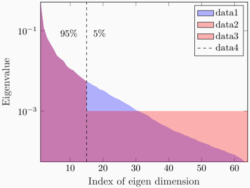
$\Rightarrow$ **$180\times$ speed-up**



(a) Random selection

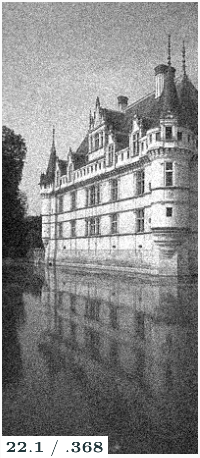


(b) Binary tree



(c) Cov. approx.

(a) Reference



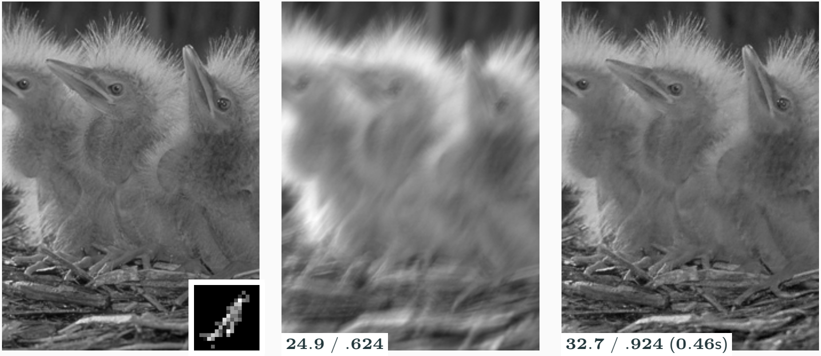
(b) Noisy image

(c) BM3D result

(d) FEPLL result

Results of denoising ($\sigma = 20$)

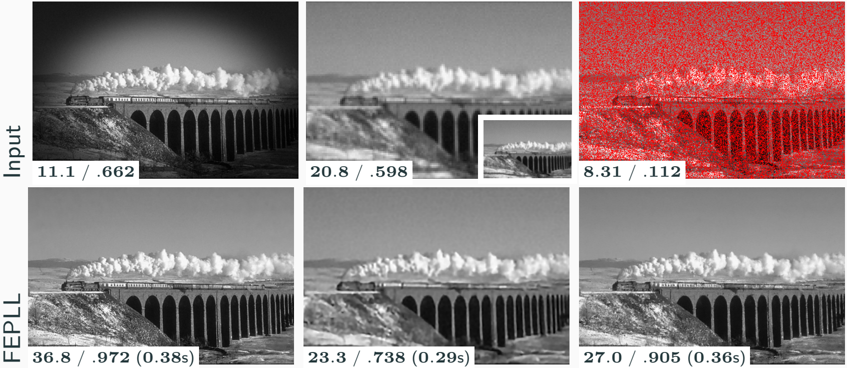
Expected patch log-likelihood (EPLL)



(a) Reference / Blur kernel (b) Blurry image (c) FEPLL result

Results of removing motion blur (subject to noise $\sigma = 0.5$)

Expected patch log-likelihood (EPLL)



(a) devignetting (b) $\times 3$ super-resolution (c) 50% random inpainting

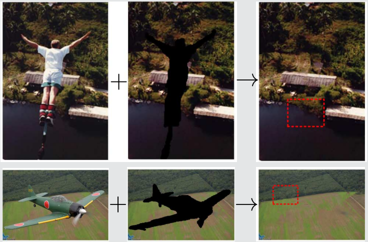
Various inverse problems (subject to noise $\sigma = 2$)

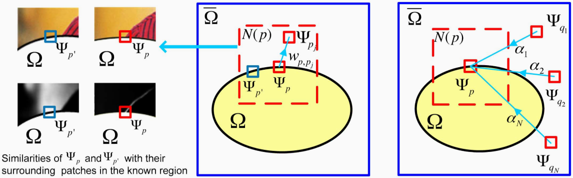
Other patch based restoration models

Inpainting

Patch propagation [Xu and Sun, 2010]

- Inpaint progressively from the edges of the missing region.
- Start with the pixels whose patches are "rare" (i.e., sparse similarity maps).



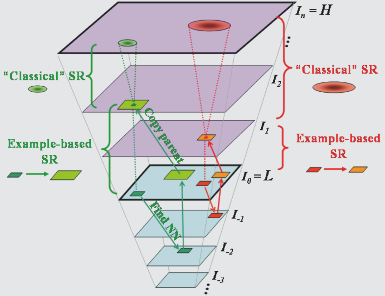


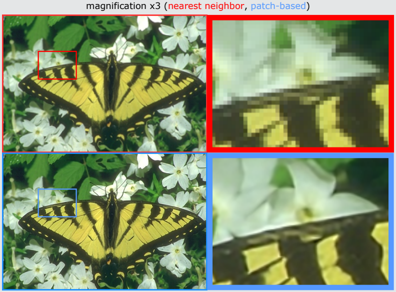
(a) Patch selection (b) Patch inpainting

Super-resolution

Super-resolution from a single image [Glasner et al., 2009]

- Simulate multi-frames: use similar patches and their sub-pixel registration.
- Match patches from low-res **and** hi-res pairs.

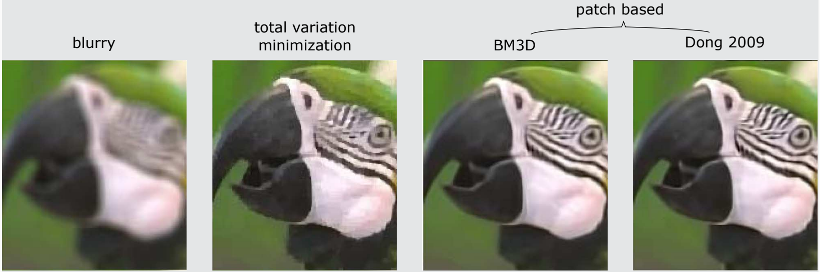




Deblurring

Adaptive sparse domain selection and regularization [Dong et al., 2011]

- Locally select dictionaries (sub-spaces),
- Perform sparse coding with the selected dictionary,
- Enforce stability under non-local filtering.



What’s next?

What's next?

Next open problems to deal with

- **Blind denoising:** statistics of the noise are unknown.
- **Blind deconvolution:** convolution kernel is unknown.
- **Non-stationary blur:** ex: moving objects, Bokeh...
- **Non-linear degradations:** ex: saturation, atmospheric turbulence...

(a) Motion blur

(b) Bokeh (Mulholand drive, 2001)

(c) Turbulence (OTIS dataset)

What's next?

Next generations of restoration techniques

- Instead of learning statistics of images or patches, such as:
    - Mean power spectral density (for Wiener filtering),
    - PCA (for LMMSE),
    - Non-local PCA (for NL Bayes),
    - Sparsifying dictionaries (k-SVD),
    - Gaussian mixture models (EPLL).

$\Rightarrow$ **Learn directly the algorithm**.

What do all these algorithms have in common?

What's next?

Non-Local means

```
for k in range(-s1, s1 + 1):
    for l in range(-s2, s2 + 1):
        yshift = shift(y, k, l)                  # Global linear
        dist2 = (yshift - y)**2                  # Pointwise non-linear
        dist2 = convolve(dist2, nu)              # Global linear
        w = phi(dist2, sig, h, P * c)            # Pointwise non-linear
        x += w * yshift                          # Pointwise non-linear
```

Regularized anisotropic diffusion

```
for k in range(m-1):
    gconv = grad(convolve(x, nu))                # Global linear
    alpha = g(norm2(gconv))                      # Pointwise non-linear
    g = grad(z)                                  # Global linear
    v = alpha * g                                # Pointwise non-linear
    x = x + gamma * div(v)                       # Global linear
```

ISTA+LASSO+UDWT+Deconvolution: BaB $= W H^* H W^+$

```
while condition:
    z = z - gamma * (BaB(z) - Bay)               # Global linear
    z = SoftT(z, gamma * tau / lambda)           # Pointwise non-linear
```

What's next?

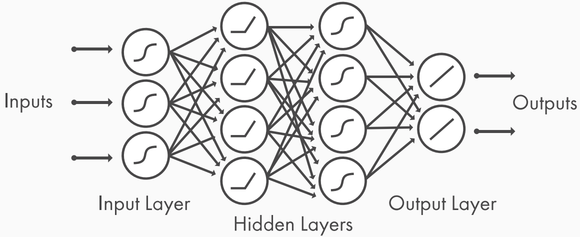
All restoration methods perform successions of:

- **global linear operations** (mixing everything):

  ex: convolutions, shifts, patch extractions, aggregations, decimations...

- **pointwise non-linear operations** (taking decisions):

  ex: thresholdings, exponentials, squares, element-wise products...

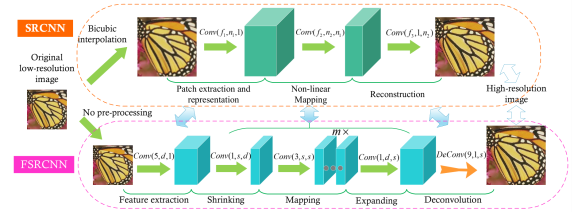


These are artificial (deep convolutional) Neural Networks (NNs).

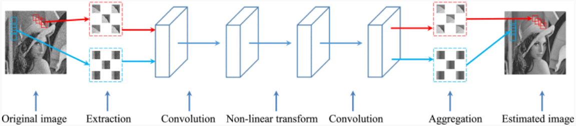
Instead of designing all steps yourself, let the machine learn them.

What's next?

Fast Super-Resolution Convolutional NN [Dong *et al.*, 2016]



BM3D-Net [Yang & Sun, 2017]



Want to learn more?

Fall quarter 2019: Machine Learning for Image Processing

Questions?

That's all folks!

Sources, images courtesy and acknowledgment

L. Denis

J. Gilles

A. Horodniceanu

G. Peyré

W. Sharba

F. Tupin

Wikipedia