# Image denoising and the structure of images

Jean-Michel Morel,
CMLA, Ecole Normale Supérieure de Cachan, France

with contributions of
Miguel Colom, Axel Davy, Thibaud Ehret,
Gabriele Facciolo, Marc Lebrun, Nicola Pierazzo, Martin Rais, Yiqing Wang, Pablo Arias, Samuel Hurault

Collège de France Séminaire de Stéphane Mallat 14/02/2018

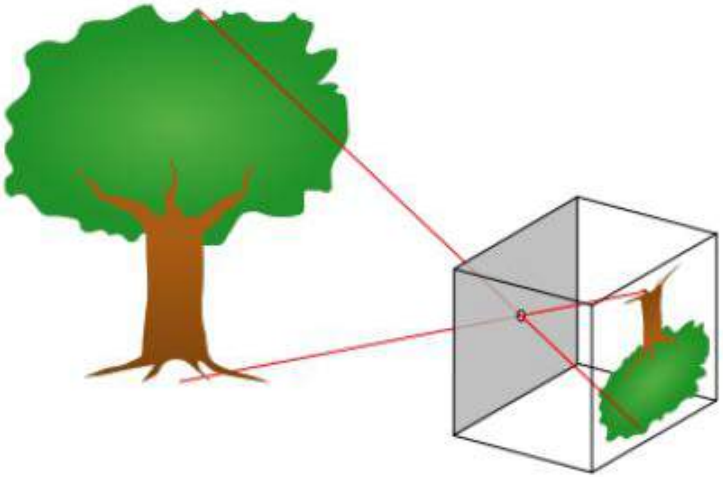
*Image denoising is a complex mathematical operation performed routinely in billions of cameras. Every digital image and every video is systematically processed numerically.*

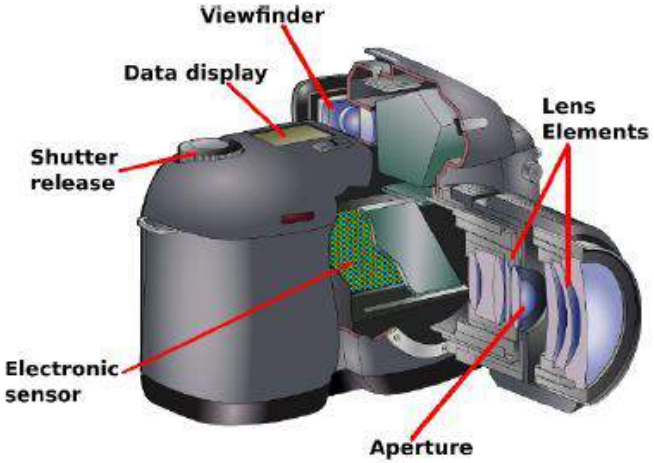
*Simple integral formulas have been invented in the past ten years and account for the steady improvement of image quality.*

**Science and technology require verification: Most algorithms that I'll show can be tested on any image in the electronic journal**

**Image Processing on Line (IPOL)**

**http://www.ipol.im/**



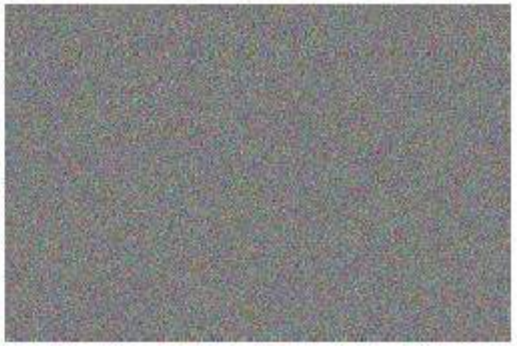



A « camera oscura » (this goes back at least to the 16th century) and a modern camera are in fact quite similar. The photons emitted by objects pass through the virtual pinhole (the lens focus) or through a real one, and hit a photosensitive surface. In modern cameras , this photosensitive surface is an electronic sensor matrix, so the image is directly sampled on a rectangular grid. It follows that digital image is nothing but a matrix of observed numbers.

Each pixel (for picture element) $\mathbf{x}$ receives a number of photons $u(\mathbf{x})$. This number is a random variable due to quantum effects of light emission and other random perturbation. Its expectation is the "ideal image" $\mathbb{E}u(\mathbf{x})$, and the difference $u(\mathbf{x}) - \mathbb{E}u(\mathbf{x})$ between observed and ideal is the noise. Our goal is to remove the noise, namely to guess $\mathbb{E}u(\mathbf{x})$.

$u(\mathbf{x})$

$\mathbb{E}u(\mathbf{x})$ ideal image



$u(\mathbf{x}) - \mathbb{E}u(\mathbf{x})$ "image noise"

- Discrete image domain $\Omega = [[0, m]] \times [[0, n]]$

For a color image, $u(\mathbf{x})$ is actually a three component vector (red, green, blue), $u(\mathbf{x}) = (r(\mathbf{x}), g(\mathbf{x}), b(\mathbf{x})) \in \mathbb{R}^3$. Each component $r(\mathbf{x})$, $g(\mathbf{x})$, $b(\mathbf{x})$ is itself a Poisson random variable and denoising therefore means estimating the three true values (expectations) of these components at each pixel $\mathbf{x}$.

Without loss of generality, we assume that the noise is white : Gaussian, independent at each pixel and for each color channel, with uniform variance $\sigma$.

Each image pixel is a random vector, and we dispose of a single sample of each. Yet, we want to estimate the expectation of this random vector, from this single sample! This problem seems completely ill-posed but will be solved by grouping pixels into patches.

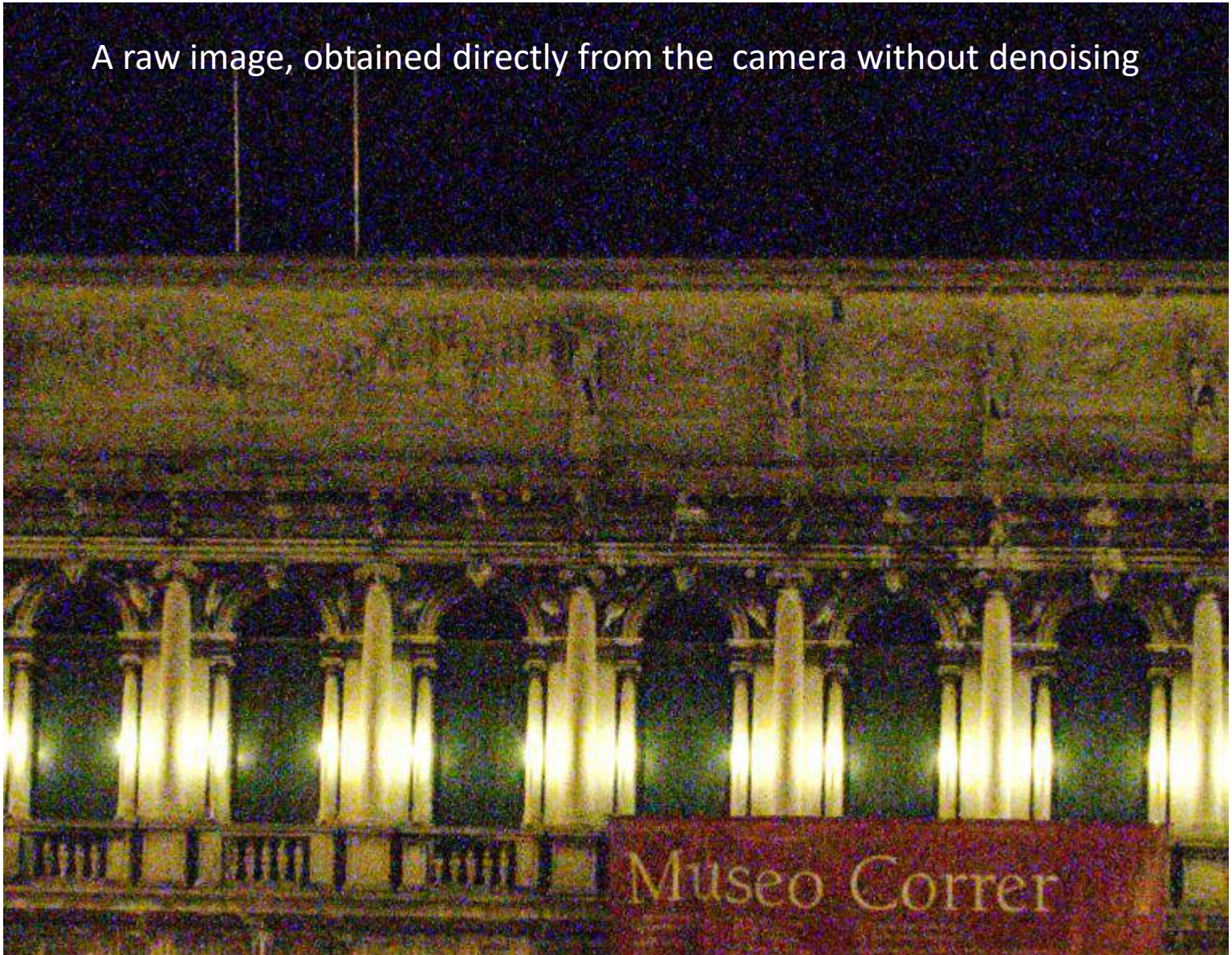
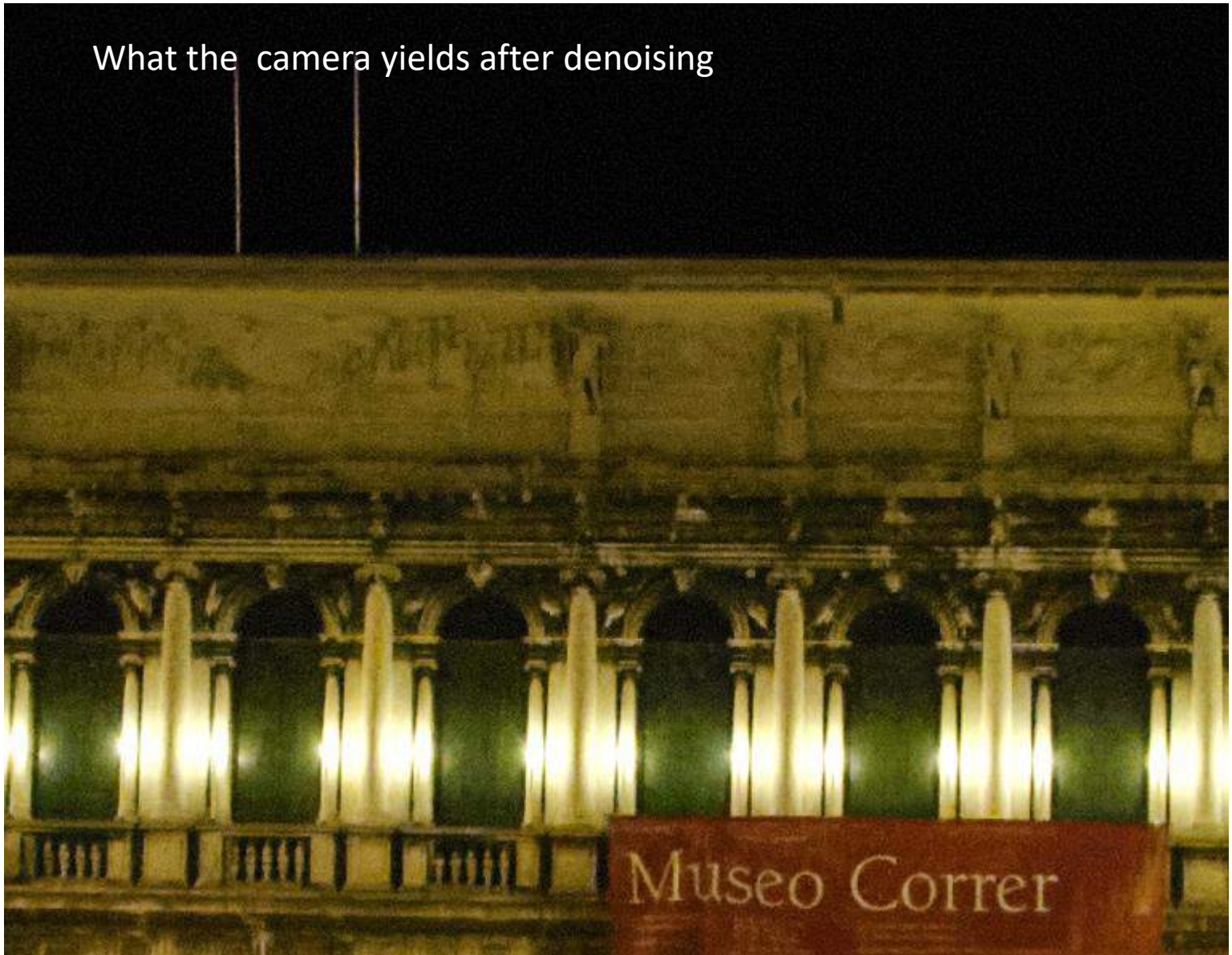



Image credit: DxO Labs

Experiment by DxO-Labs, a company specialized on image processing. One can compare the resulting image with-and-without denoising. By night the problem is more difficult because there are less photons and therefore more noise.

Image credit: DxO Labs



# Denoising in five formulas from local to global

**1-Transform thresholding: the example of DCT denoising**

**2-Neighborhood filters: an old and fantastic trick**

**3-A slight extension: nonlocal means**

**4-The Bayesian denoising paradigm from « non-local » to « global »**

**5-Machine learning algorithms**

**6-Back to our starting point: dual denoising and transform thresholding**

***Where to test all algorithms:*** ***Image Processing on Line (IPOL)***
***http://www.ipol.im/***

# Linear frequency transform thresholding: the example of discrete cosine transform (DCT) denoising

Transform thresholding can be based on the Fourier transform ( Shannon, Wiener), on wavelets (Meyer, Daubechies, Mallat, Coifman, Donoho & Johnstone, Starck,..), or on the discrete cosine transform (Yaroslavski).

Transform thresholding assumes that **the image is sparse on one of these bases** (Candès, Romberg, Tao). The noise frequency coefficients have instead all the same variance, and are therefore uniformly small.

Thus, by cancelling the small frequency coefficients of the noisy image (the threshold is typically lower than $3\sigma$), the noise is reduced and the signal is mostly preserved.

In DCT denoising this operation is performed on each image 8x8 pixels « patch ».

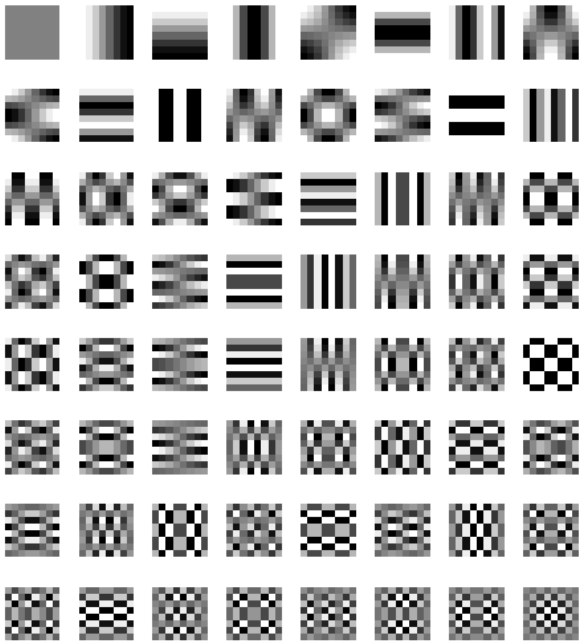
**Wiener, Norbert (1949). *Extrapolation, Interpolation, and Smoothing of Stationary Time Series*. New York: Wiley.**

# Linear frequency transform thresholding: the example of discrete cosine transform (DCT) denoising

"patch" $P_x$: restriction of the image $u$ to a small block around a pixel x

$\sigma$: standard deviation of the noise

Each image patch is decomposed on the DCT basis and all of its small frequency coefficients are canceled. The DCT basis is a local variant of the Fourier transform on 8x8 patches.



**DCT basis for 8x8 patches:**
*cos(nx)cos(my), m,n =0,…,7*

Input: noisy image, noise standard deviation $\sigma$:
For each 8x8 image patch **P_x** :

-Calculate 2D-DCT transform of the patch;

-Cancel all DCT coefficients with absolute value below $3\sigma$.

-Calculate inverse 2D-DCT transform of the patch.

- Obtain the denoised color **u(x)** as the average of all those obtained for all 64 patches containing **x.**

**DCT denoising algorithm**

Crop of denoised images by sliding DCT thresholding filter and incrementally adding:

-use of a YoUoVo colour system,

-uniform aggregation,

-variance based aggregation and

-iteration with the “oracle” given by the first step.

The corresponding PSNR are 26,85; 27,33; 30,65; 30,73; 31,25.

$$\mathrm{RMSE} = \sqrt{\frac{||y-x||_2^2}{|\Omega|}}.$$

$$\mathrm{PSNR} = 20\log_{10}\left(\frac{255}{RMSE}\right).$$

**σ=15 noisy**

DCT denoising is quite successful with moderate noise. **This amounts to cancel the smaller DCT coefficients of each patch**, as they mainly contain noise.

DCT transform threshold denoising: σ=15: G. Yu, G. Sapiro http://www.ipol.im/

σ=15 denoised

DCT denoising is quite successful with moderate noise. **This amounts to cancel the smaller DCT coefficients of each patch**, as they mainly contain noise.

DCT transform threshold denoising: σ=15: G. Yu, G. Sapiro http://www.ipol.im/

σ=15
original

DCT denoising is quite successful with moderate noise. **This amounts to cancel the smaller DCT coefficients of each patch**, as they mainly contain noise.

DCT transform threshold denoising: σ=15: G. Yu, G. Sapiro http://www.ipol.im/

DCT denoising and in general (wavelet) transform thresholding with a too large threshold creates **Gibbs effects** for large noise. This Gibbs effect or « ringing » appears everywhere

**2-DCT transform threshold denoising: σ=40: G. Yu, G. Sapiro** http://www.ipol.im/

DCT denoising and in general (wavelet) transform thresholding with a too large threshold creates **Gibbs effects** for large noise. This Gibbs effect or « ringing » appears everywhere

**2-DCT transform threshold denoising: σ=40: G. Yu, G. Sapiro** http://www.ipol.im/

DCT denoising and in general (wavelet) transform thresholding with a too large threshold creates **Gibbs effects** for large noise. This Gibbs effect or « ringing » appears everywhere

**Multiscale DCT denoising: σ=40:  G. Facciolo, N. Pierazzo** http://www.ipol.im/

DCT denoising and in general (wavelet) transform thresholding with a too large threshold creates **Gibbs effects** for large noise. This Gibbs effect or « ringing » appears everywhere

**Multiscale DCT denoising: σ=40:  G. Facciolo, N. Pierazzo** http://www.ipol.im/

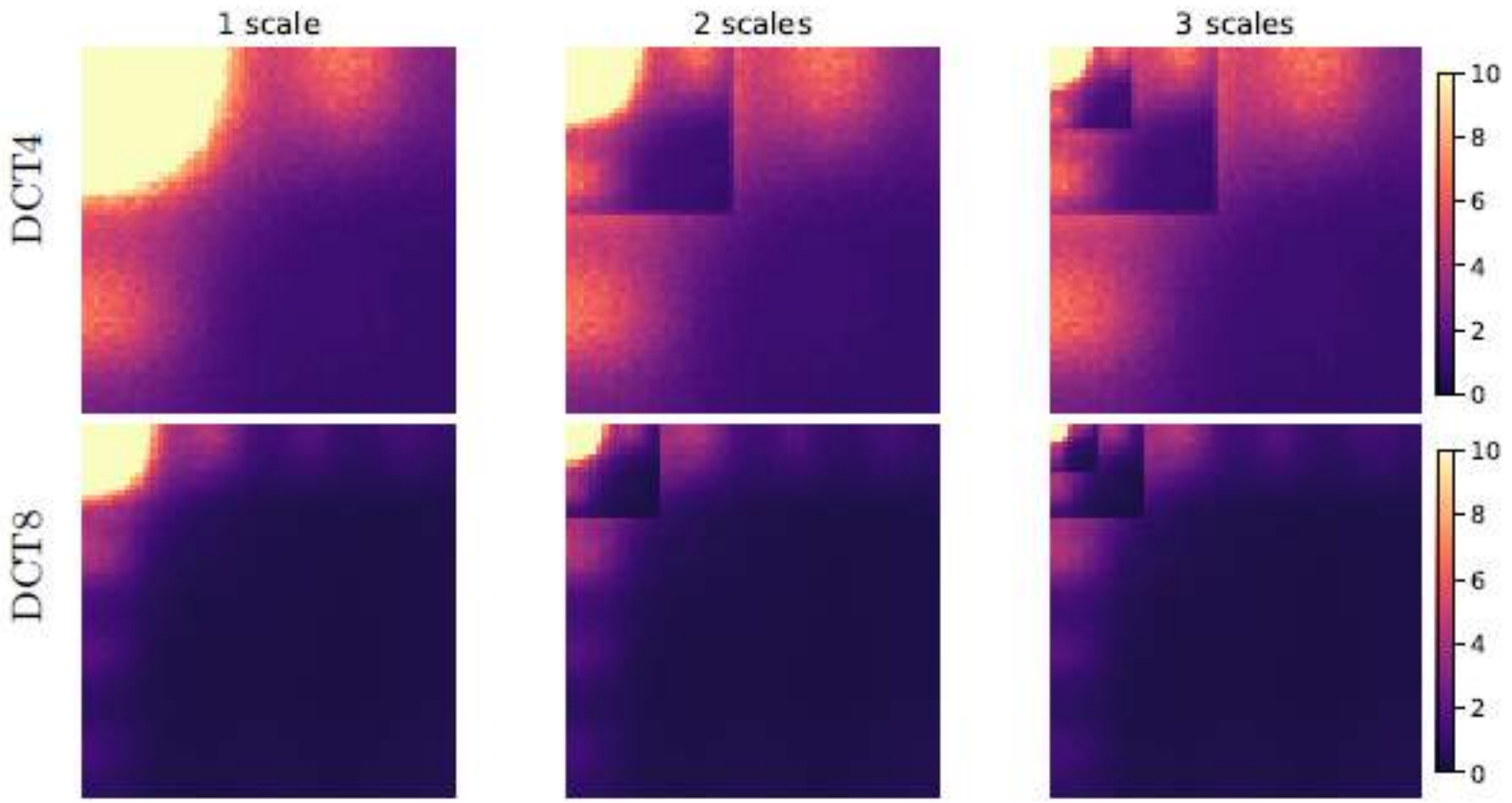



FIGURE 6.1: DCT transform amplitude of results of the multiscale DCT denoising algorithm applied to an image of pure white noise. The columns correspond to different numbers of scales. Notice the remaining low frequency noise in the upper left corner of the single scale results (first column). As expected, in the multi-scale results the residual noise is much lower. The results in the first row are computed with DCT denoising using $4 \times 4$ patches and using a recomposition factor $f_{rec} = 0.9$, while for the second row $8 \times 8$ patches are used with $f_{rec} = 0.5$.

# Denoising in five formulas from local to global

**1-Transform thresholding: the example of DCT denoising**

**2-Neighborhood filters: an old and fantastic trick**

**3-A slight extension: nonlocal means**

**4-The Bayesian denoising paradigm from « non-local » to « global »**

**5-Machine learning algorithms**

**6-Back to our starting point: dual denoising and transform thresholding**

***Where to test all algorithms:*** ***Image Processing on Line (IPOL)***
***http://www.ipol.im/***

**The trivial idea behind neighborhood filters: find the right samples in the image and average them. This is still a local smoothing but it is local in higher dimension (image + values)**

## A meaningful simple formula

Assume that $u(\mathbf{x}_1)$, $u(\mathbf{x}_2)$, ..., $u(\mathbf{x}_n)$ are observed noisy samples of the same underlying color $u(\mathbf{x})$. Then the mean $\frac{1}{n}\sum_{i=1}^{n} u(\mathbf{x}_i)$ is a much better estimate of $u(\mathbf{x})$ than each $u(\mathbf{x}_i)$. Indeed,

$$\mathrm{Var}\left(\frac{1}{n}\sum_{i=1}^{n} u(\mathbf{x}_i)\right) = \frac{1}{n}\mathrm{Var}(u(\mathbf{x}_1)).$$

Thus, it is enough to find, say, 16 samples for $u(\mathbf{x})$ to divide the noise by 4. Neighborhood filters propose a new way to select these samples.

Yaroslavsky, Leonid P. "Digital picture processing: an introduction." Applied Optics 25 (1986)

Tomasi, Carlo, and Roberto Manduchi. "Bilateral filtering for gray and color images." Computer Vision, 1998. Sixth International Conference on. IEEE, (1998)

# Gaussian convolution as a denoiser

A first simple way to select the samples that will permit to estimate $u(\mathbf{x})$ is to assume that all pixels in a spatial (Gaussian weighted) neighborhood have the same underlying colour. This amounts to convolve the image with a Gaussian, in other terms to replace $u(\mathbf{x})$ by a weighted average of the values $u(\mathbf{y})$ in a neighborhood of **x**.

$$G_\sigma^2 * u(\mathbf{x}) = \frac{\int_\Omega G_\sigma^2(\mathbf{x}-\mathbf{y})u(\mathbf{y})d\mathbf{y}}{\int_\Omega G_\sigma^2(\mathbf{x}-\mathbf{y})d\mathbf{y}}$$

← weighted average

← normalization term

But a still much better way was invented with neighborhood filters: $u(\mathbf{y})$ will contribute to the estimate of u(**x**) if and only **x** is close to **y** **but also** $u(\mathbf{y})$ is close to $u(\mathbf{x})$. The next slide gives the formula.

**Reminder**: $$G_\sigma^N(\mathbf{x}) = \frac{1}{(2\pi\sigma)^{\frac{N}{2}}} e^{-\frac{||\mathbf{x}||^2}{2\sigma^2}}$$

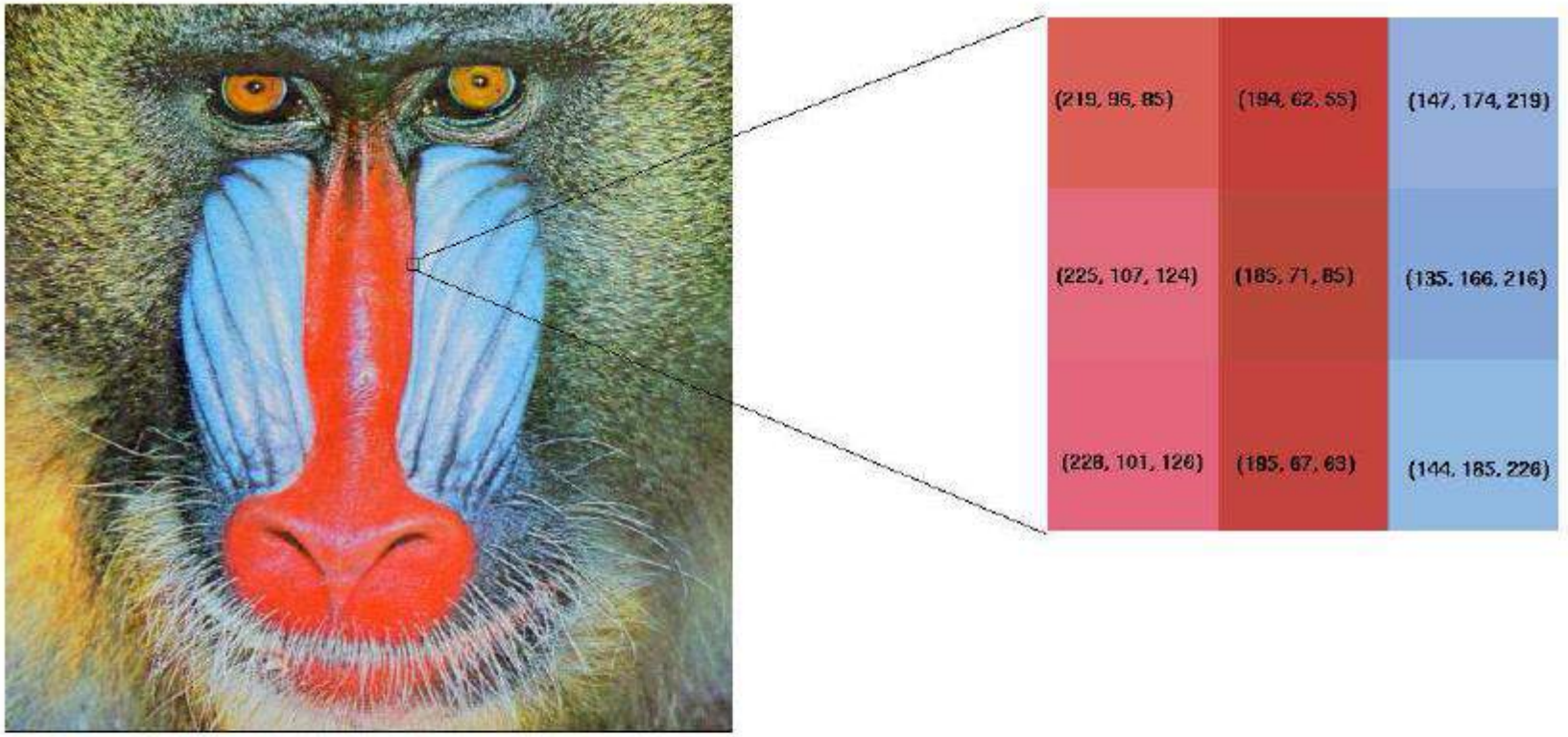



Figure 1: A. Buades, B. Coll, and J.M M "Neighborhood filters and PDE's", Numerische Mathematik, 105 (1), 2006.

**3-Neighborhood filters: an old and fantastic trick**

# Neighborhood filter, Sigma filter, SUSAN, Bilateral filter

[Yaroslavski 80, Lee 83, Smith-Brady 97, Tomasi-Manduchi 98]

Now the Gaussian weights guarantee a proximity in space and color:

$$\mathbf{x}, \mathbf{y} : \text{pixels}; u(\mathbf{x}) : \text{image color at } \mathbf{x} \qquad G_\sigma^N(\mathbf{x}) = \frac{1}{(2\pi\sigma)^{\frac{N}{2}}} e^{-\frac{||\mathbf{x}||^2}{2\sigma^2}}$$

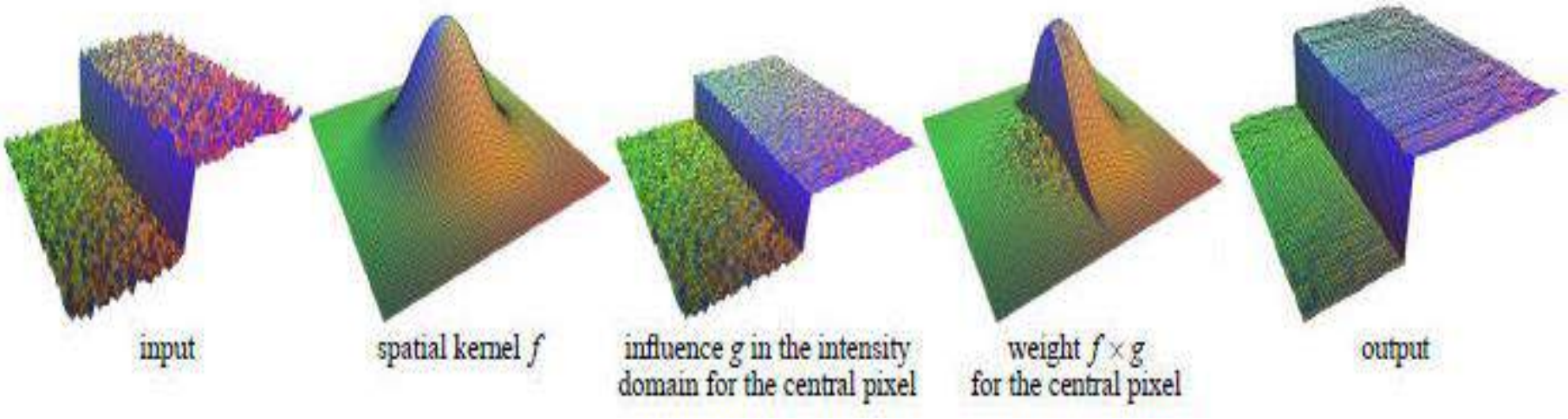
Spatial Filter f

Range Filter g

$$NFu(\mathbf{x}) = \frac{\int_\Omega G^2_{\sigma_s}(\mathbf{x}-\mathbf{y}) G^3_{\sigma_r}(u(\mathbf{x})-u(\mathbf{y})) u(\mathbf{y}) d\mathbf{y}}{\int_\Omega G^2_{\sigma_s}(\mathbf{x}-\mathbf{y}) G^3_{\sigma_r}(u(\mathbf{x})-u(\mathbf{y})) d\mathbf{y}}$$

← weighted average

← normalization term




**2-Neighborhood filters: an old and fantastic trick**

# Denoising in five formulas from local to global

1-Transform thresholding: the example of DCT denoising

2-Neighborhood filters: an old and fantastic trick

3-A slight extension: nonlocal means

4-The Bayesian denoising paradigm from « non-local » to « global »

5-Machine learning algorithms

6-Back to our starting point: dual denoising and transform thresholding

*Where to test all algorithms: Image Processing on Line (IPOL)*
*http://www.ipol.im/*

# Regularity in the patch space?

## Nonlocal-means denoising (2004)

$$NL(P_{\mathbf{x}}) = \frac{\int_{\Omega} G^N_{\sigma_p}(P_{\mathbf{x}} - P_{\mathbf{y}}) P_{\mathbf{y}} d\mathbf{y}}{\int_{\Omega} G^N_{\sigma_p}(P_{\mathbf{x}} - P_{\mathbf{y}}) d\mathbf{y}}$$

$$G^N_{\sigma}(\mathbf{x}) = \frac{1}{(2\pi\sigma)^{\frac{N}{2}}} e^{-\frac{||\mathbf{x}||^2}{2\sigma^2}}$$

$N$ : patch dimension (typically $8 \times 8$)

Here, the idea is to retain as valid samples for $u(\mathbf{x})$ only samples $u(\mathbf{y})$ whose surrounding patch $P_{\mathbf{y}}$ is similar to the patch $P_{\mathbf{x}}$ surrounding $\mathbf{x}$.

**This idea goes back to Shannon (1948) who used it to simulate text.**

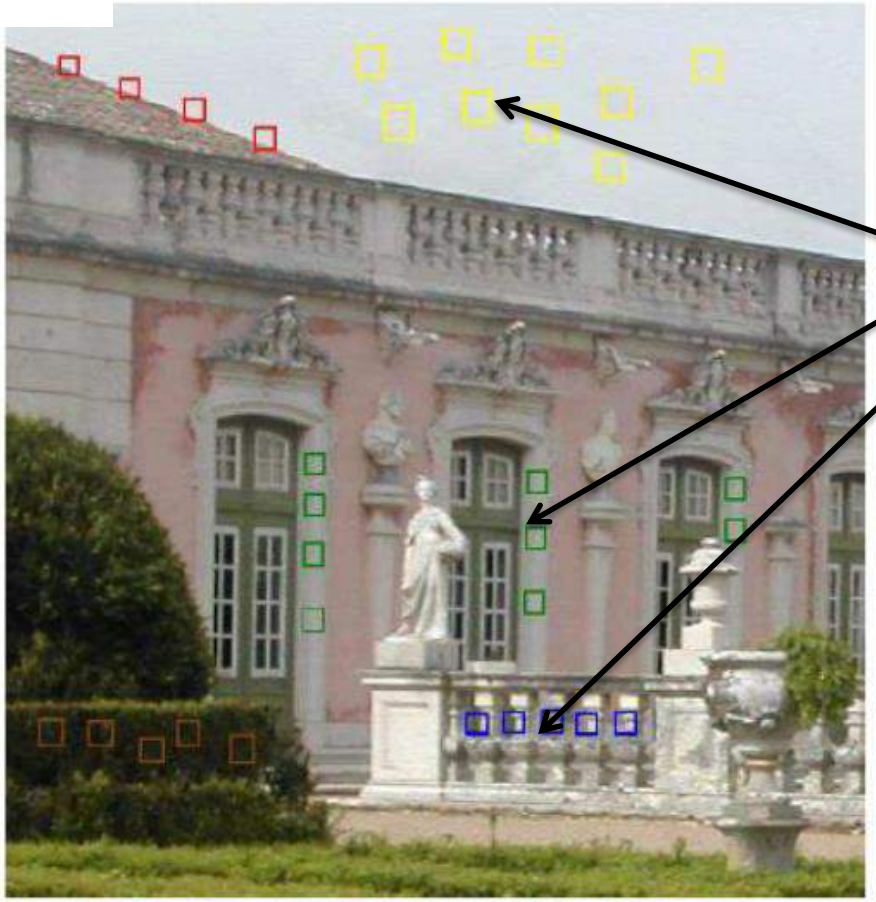
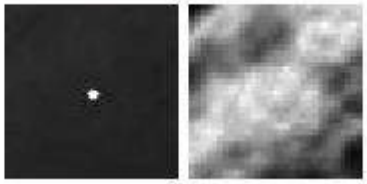


Image and its groups of « self-similar » patches. Nonlocal means computes an average of these groups to denoise them jointly.

**Shannon's ideas extended by Efros and Leung (1999) to expand any texture from a small sample.**

**3-Nonlocal Means**

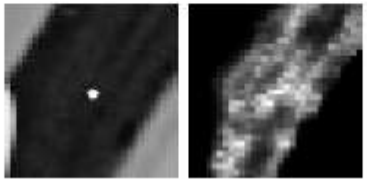
- Flat region. The large coefficients are spread out like a convolution.



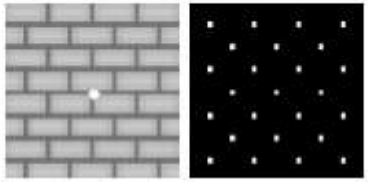
**Visualization of the patch « heat kernel » back-projected onto the image**

- Curved edge. The weights favor pixels belonging to the same contour.

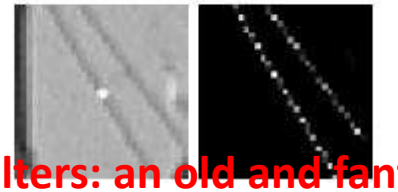
- Flat neighborhood. The average is made in the grey level neighborhood as the neighborhood filter.



- Periodic case. The large coefficients are distributed across the texture (non local).



- Repetitive structures. The weights favor similar configurations even they are far away (non local).



**3-Neighborhood filters: an old and fantastic trick**

**All these denoising algorithms boil down to three Gaussian convolutions with growing specificity: the first one is the classic Wiener spatial smoothing, the second is space+value nonlinear convolution, the third is a convolution with a Gaussian in the « patch space »**

Gaussian convolution

$$G_\sigma^2 * u(\mathbf{x}) = \frac{\int_\Omega G_\sigma^2(\mathbf{x}-\mathbf{y})u(\mathbf{y})d\mathbf{y}}{\int_\Omega G_\sigma^2(\mathbf{x}-\mathbf{y})d\mathbf{y}}$$

Neighborhood filter (Yaroslavski, Lee 1980)

$$NFu(\mathbf{x}) = \frac{\int_\Omega G_{\sigma_s}^2(\mathbf{x}-\mathbf{y})G_{\sigma_r}^3(u(\mathbf{x})-u(\mathbf{y}))u(\mathbf{y})d\mathbf{y}}{\int_\Omega G_{\sigma_s}^2(\mathbf{x}-\mathbf{y})G_{\sigma_r}^3(u(\mathbf{x})-u(\mathbf{y}))d\mathbf{y}}$$

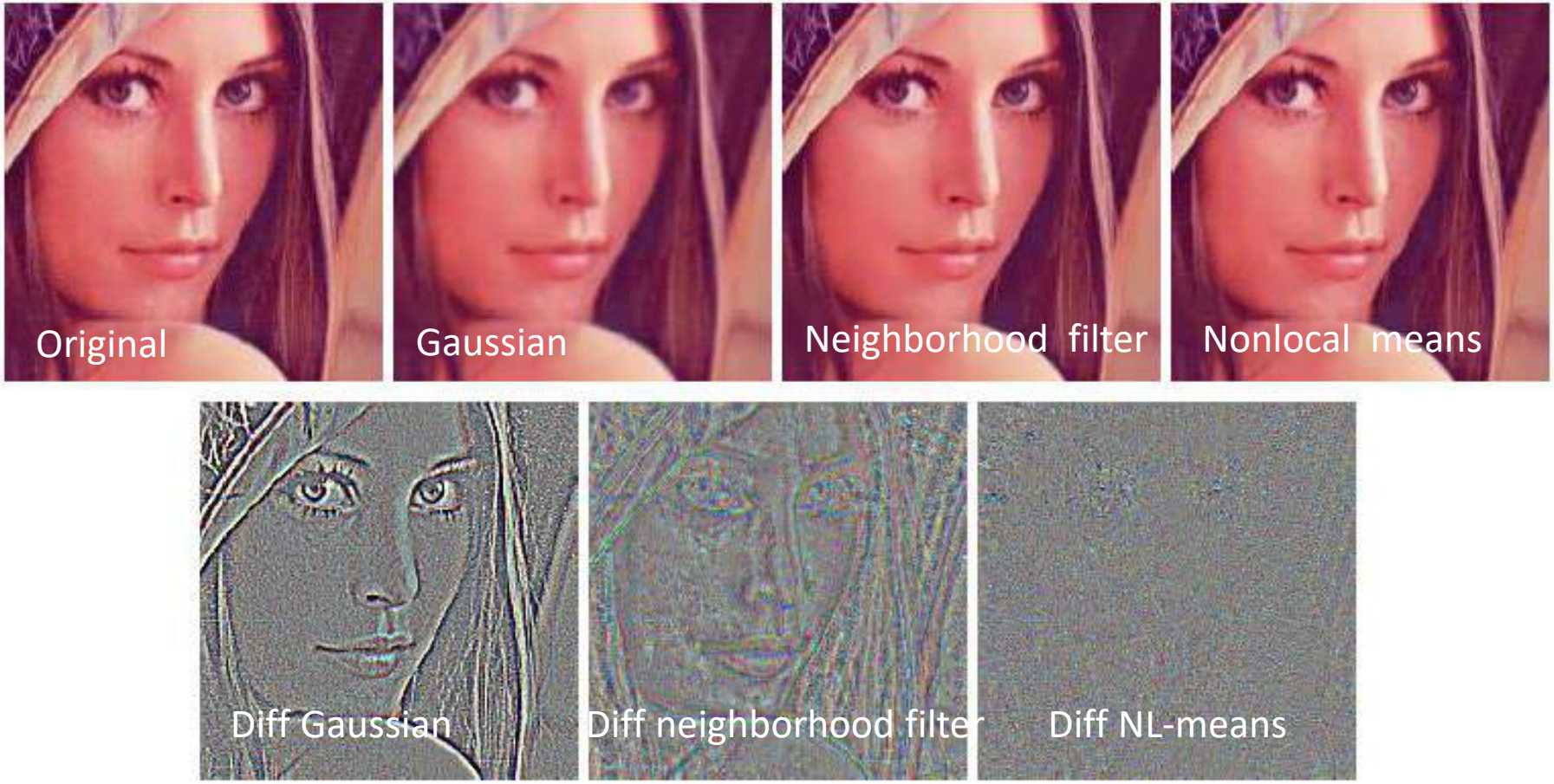
Nonlocal-means denoising (Buades & Coll & M. 2004)

$$NL(P_\mathbf{x}) = \frac{\int_\Omega G_{\sigma_p}^N(P_\mathbf{x}-P_\mathbf{y})P_\mathbf{y}d\mathbf{y}}{\int_\Omega G_{\sigma_p}^N(P_\mathbf{x}-P_\mathbf{y})d\mathbf{y}}$$

**3-Nonlocal Means**

## Visual Comparison

Comparison of **Gaussian convolution**, **neighborhood filter**, **nonlocal means** on a real scanned image (Lena). The difference between the image and its filtered version should look like noise




Difference between the image and its filtered version. It should contain no visible structure. Hence NL-means is better than NF, which is better than the Gaussian convolution.

# Denoising in five formulas from local to global

**1-Transform thresholding: the example of DCT denoising**

**2-Neighborhood filters: an old and fantastic trick**

**3-A slight extension: nonlocal means**

**4-The Bayesian denoising paradigm from « non-local » to « global »**

**5-Machine learning algorithms**

**6-Back to our starting point: dual denoising and transform thresholding**

***Where to test all algorithms:*** ***Image Processing on Line (IPOL)***
***http://www.ipol.im/***


30

# Global Bayesian Denoising (Nadler & Levin 2011)

$\mathcal{P}$ is the set of all patches in the world
The probability of observing the noisy patch $P$ given a perfect patch $Q$ is

$$\mathbb{P}(P \mid Q) = \frac{1}{(2\pi\sigma^2)^{\frac{N}{2}}} e^{-\frac{\|P-Q\|^2}{2\sigma^2}} = G_\sigma^N(P-Q)$$

Thus, given a noisy patch $P$ its optimal estimator for the mean square error is by Bayes' formula

$$GBD(P) = \mathbb{E}[Q \mid P] = \int \mathbb{P}(Q \mid P) Q dQ = \int \frac{\mathbb{P}(P \mid Q)}{\mathbb{P}(P)} \mathbb{P}(Q) Q dQ$$

Hence the Global Bayes Denoiser

$$GBD(P) = \frac{\int_{\mathcal{P}} G_\sigma^N(P-Q) Q dQ}{\int_{\mathcal{P}} G_\sigma^N(P-Q) dQ}$$

**Gives the best estimate of the noisy patch $P$ knowing all the patches $Q \in \mathcal{P}$ in the world! Tested on $2^{10}$ patches.**

# Global Bayesian Denoising (Nadler & Levin 2011)

$\mathcal{P}$ is the set of all patches in the world

The probability of observing the noisy patch $P$ given a perfect patch $Q$ is

$$\mathbb{P}(P \mid Q) = \frac{1}{(2\pi\sigma^2)^{\frac{N}{2}}} e^{-\frac{\|P-Q\|^2}{2\sigma^2}} = G_\sigma^N(P-Q)$$

Thus, given a noisy patch $P$ its optimal estimator for the mean square error is by Bayes' formula

$$GBD(P) = \mathbb{E}[Q \mid P] = \int \mathbb{P}(Q \mid P) Q dQ = \int \frac{\mathbb{P}(P \mid Q)}{\mathbb{P}(P)} \mathbb{P}(Q) Q dQ$$

Hence the Global Bayes Denoiser

$$GBD(P) = \frac{\int_{\mathcal{P}} G_\sigma^N(P-Q) Q dQ}{\int_{\mathcal{P}} G_\sigma^N(P-Q) dQ}$$

Gives the best estimate of the noisy patch $P$ knowing all the patches $Q \in \mathcal{P}$ in the world! Tested on $2^{10}$ patches.

N.B: the last integral is obtained by a change of variables. The preceding integral is w.r. to the Lebesgue measure. **The last integral is made on the « space of patches ».**

## The Levin and Nadler optimal « global denoising algorithm » uses « all patches of the world »

- **Input**: Noisy image $\tilde{u}$, its patches $\tilde{P}$
- **Input**: Very large set of $M = 2^{10}$ patches $P_i$ extracted from a large set of noiseless natural images (20000)
- **Output**: Denoised image $\hat{u}$.
- **for all patches $\tilde{P}$ extracted from $\tilde{u}$:** Compute the MMSE denoised estimate of $\tilde{P}$

$$\hat{P} \simeq \frac{\sum_{i=1}^{M} \mathbb{P}(\tilde{P} \mid P_i) P_i}{\sum_{i=1}^{M} \mathbb{P}(\tilde{P} \mid P_i)} \qquad G_\sigma^N(\mathbf{x}) = \frac{1}{(2\pi\sigma)^{\frac{N}{2}}} e^{-\frac{\|\mathbf{x}\|^2}{2\sigma^2}}$$

  where $\mathbb{P}(\tilde{P} \mid P_i)$ is known from the Gaussian noise distribution.
- (Aggregation) : for each pixel $\mathbf{j}$ of $u$, compute the denoised version $\hat{u}_\mathbf{j}$ as the average of all values $\hat{P}(\mathbf{j})$ for all patches

A. Levin, B. Nadler. *CVPR* 2011. Natural image denoising: Optimality and inherent bounds
Zoran, D., & Weiss, Y. *ICCV* 2011. From learning models of natural image patches to whole image restoration.

# NL-Bayes : the only diference w.r. to global denoising is that a Gaussian model is locally estimated in the image patch space

- patch noise model $\mathbb{P}(\tilde{P}|P) = c \cdot e^{-\frac{\|\tilde{P}-P\|^2}{2\sigma^2}}$
- Bayes' rule $\mathbb{P}(P|\tilde{P}) = \frac{\mathbb{P}(\tilde{P}|P)\mathbb{P}(P)}{\mathbb{P}(\tilde{P})}$
- assume we got a patch Gaussian model $\mathbb{P}(Q) = c.e^{-\frac{(Q-\overline{P})^t \mathbf{C}_P^{-1}(Q-\overline{P})}{2}}$
- hence the variational problem

$$\begin{aligned}
\max_P \mathbb{P}(P|\tilde{P}) &\Leftrightarrow \max_P \mathbb{P}(\tilde{P}|P)\mathbb{P}(P) \\
&\Leftrightarrow \max_P e^{-\frac{\|P-\tilde{P}\|^2}{2\sigma^2}} e^{-\frac{(P-\overline{P})^t \mathbf{C}_P^{-1}(P-\overline{P})}{2}} \\
&\Leftrightarrow \min_P \frac{\|P-\tilde{P}\|^2}{\sigma^2} + (P-\overline{P})^t \mathbf{C}_P^{-1}(P-\overline{P}).
\end{aligned}$$

- An empirical covariance matrix $C_{\tilde{P}}$ can be obtained for the patches $\tilde{Q}$ similar to $\tilde{P}$. $P$ and the noise $n$ being independent, $\mathbf{C}_{\tilde{P}} = \mathbf{C}_P + \sigma^2\mathbf{I}$; $\quad E\tilde{Q} = \overline{P}$

A. Buades, M. Lebrun, J.M.M. : A Non-local Bayesian image denoising algorithm, SIIMS 2013

*BLS-GSM*: J. Portilla, V. Strela, M.J. Wainwright, E.P. Simoncelli, TIP 2003

# Bayesian denoising : NL-Bayes

$\max_P \mathbb{P}(P|\tilde{P}) \Leftrightarrow \min_P \frac{\|P-\tilde{P}\|^2}{\sigma^2} + (P-\overline{\tilde{P}})^t(\mathbf{C}_{\tilde{P}} - \sigma^2\mathbf{I})^{-1}(P-\overline{\tilde{P}})$

one step estimation $\hat{P}_1 = \overline{\tilde{P}} + \left[\mathbf{C}_{\tilde{P}} - \sigma^2\mathbf{I}\right]\mathbf{C}_{\tilde{P}}^{-1}(\tilde{P}-\overline{\tilde{P}})$, where empirically:

$$\mathbf{C}_{\tilde{P}} \simeq \frac{1}{\#\mathcal{P}(\tilde{P})-1}\sum_{\tilde{Q}\in\mathcal{P}(\tilde{P})}\left(\tilde{Q}-\overline{\tilde{P}}\right)\left(\tilde{Q}-\overline{\tilde{P}}\right)^t, \quad \overline{\tilde{P}} \simeq \frac{1}{\#\mathcal{P}(\tilde{P})}\sum_{\tilde{Q}\in\mathcal{P}(\tilde{P})}\tilde{Q}.$$

Iteration ("oracle estimation"): $\hat{P}_2 = \overline{\tilde{P}}^1 + \mathbf{C}_{\hat{P}_1}\left[\mathbf{C}_{\hat{P}_1} + \sigma^2\mathbf{I}\right]^{-1}(\tilde{P}-\overline{\tilde{P}}^1)$

where

$$\mathbf{C}_{\hat{P}_1} \simeq \frac{1}{\#\mathcal{P}(\hat{P}_1)-1}\sum_{\hat{Q}_1\in\mathcal{P}(\hat{P}_1)}\left(\hat{Q}_1-\overline{\tilde{P}}^1\right)\left(\hat{Q}_1-\overline{\tilde{P}}^1\right)^t, \quad \overline{\tilde{P}}^1 \simeq \frac{1}{\#\mathcal{P}(\hat{P}_1)}\sum_{\hat{Q}_1\in\mathcal{P}(\hat{P}_1)}\tilde{Q}.$$

A. Buades, M. Lebrun, J.M.M.: A Non-local Bayesian image denoising algorithm, SIIMS 2013

## Gaussian convolution

**All denoising integral formulas in one slide**

$$G_\sigma^2 * u(\mathbf{x}) = \frac{\int_\Omega G_\sigma^2(\mathbf{x}-\mathbf{y})u(\mathbf{y})d\mathbf{y}}{\int_\Omega G_\sigma^2(\mathbf{x}-\mathbf{y})d\mathbf{y}}$$

## Neighborhood filter (Yaroslavski, Lee 1980)

$$NFu(\mathbf{x}) = \frac{\int_\Omega G_{\sigma_s}^2(\mathbf{x}-\mathbf{y})G_{\sigma_r}^3(u(\mathbf{x})-u(\mathbf{y}))u(\mathbf{y})d\mathbf{y}}{\int_\Omega G_{\sigma_s}^2(\mathbf{x}-\mathbf{y})G_{\sigma_r}^3(u(\mathbf{x})-u(\mathbf{y}))d\mathbf{y}}$$

## Nonlocal-means denoising (Buades & Coll & M. 2004)

$$NL(P_\mathbf{x}) = \frac{\int_\Omega G_{\sigma_p}^N(P_\mathbf{x}-P_\mathbf{y})P_\mathbf{y}d\mathbf{y}}{\int_\Omega G_{\sigma_p}^N(P_\mathbf{x}-P_\mathbf{y})d\mathbf{y}}$$

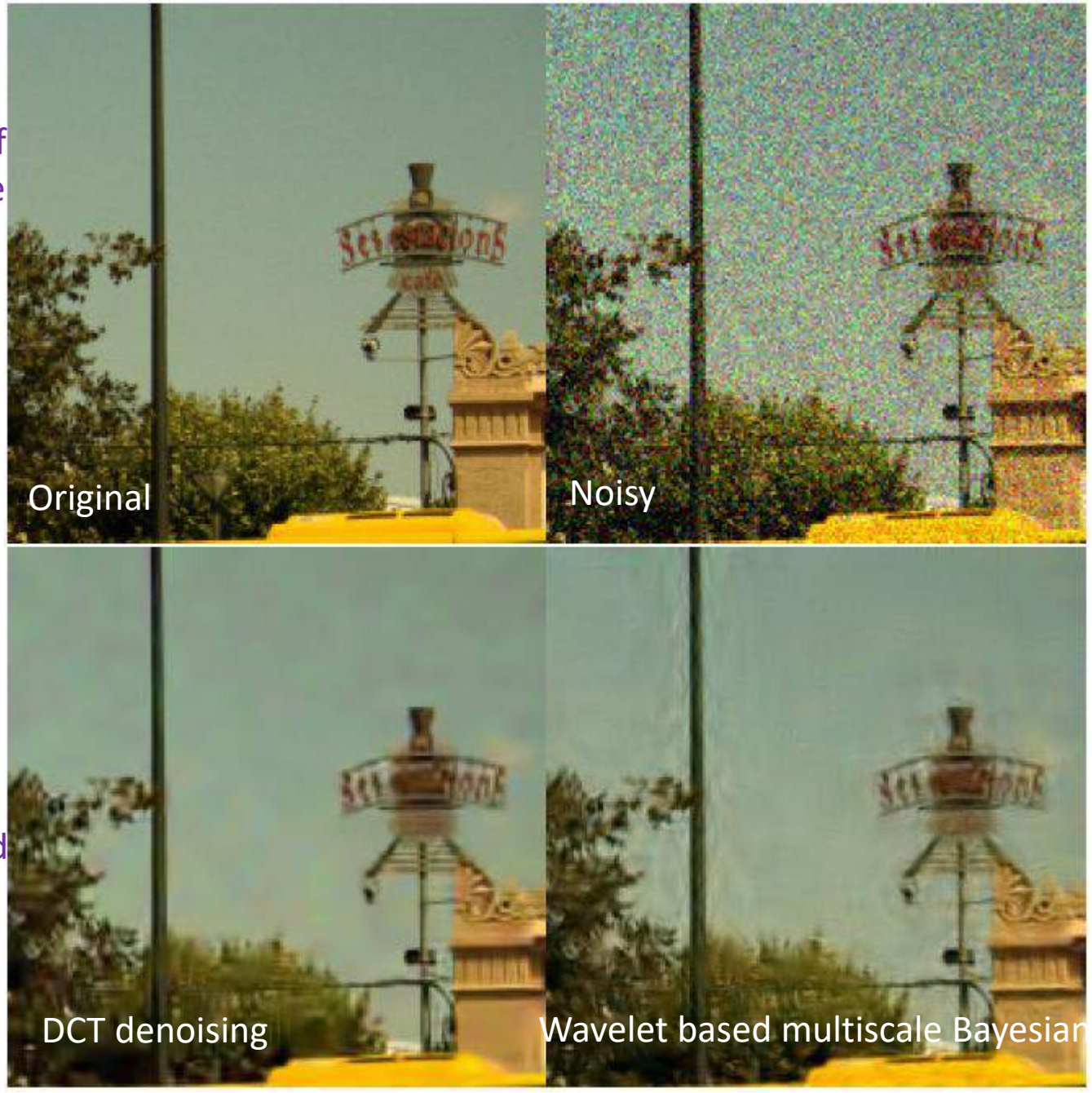
## Global Bayesian Denoising (Nadler & Levin 2011)

$$GBD(P) = \frac{\int_\mathcal{P} G_\sigma^N(P-Q)QdQ}{\int_\mathcal{P} G_\sigma^N(P-Q)dQ}$$

$\mathcal{P}$ is the set of all patches in the world

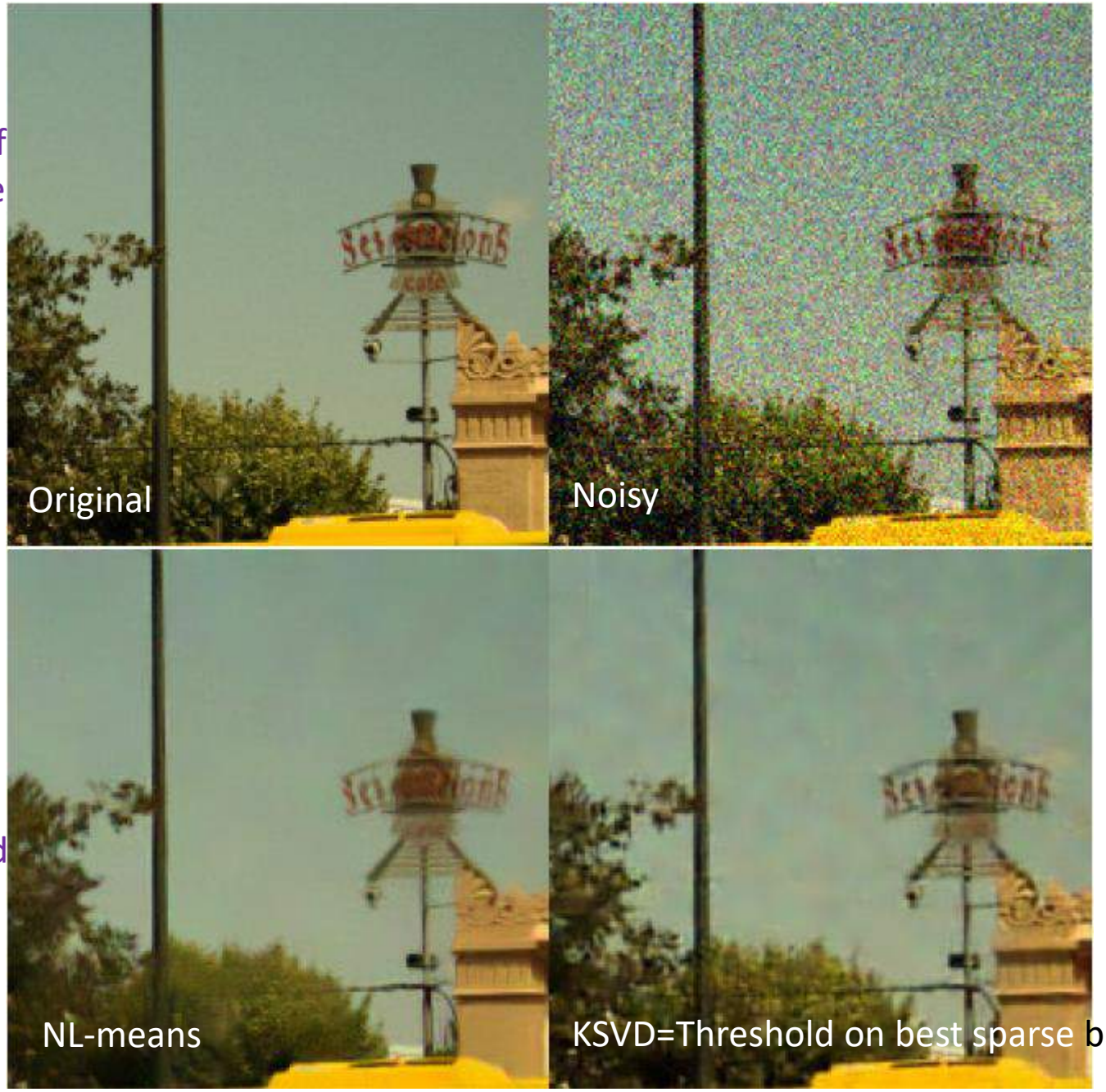
Szlam, A. D., Maggioni, M., & Coifman, R. R. *JMLR* 2008. Regularization on graphs with function-adapted diffusion processes.

Comparison of six state of the art methods, which combine the above mentioned principles: transform thresholding, nonlocal means and the Bayesian global estimator. Some are used in real cameras.




Original, noisy, DCT sliding window, BLS-GSM. Experiments: IPOL

Comparison of six state of the art methods, which combine the above mentioned principles: transform thresholding, nonlocal means and the Bayesian global estimator. Some are used in real cameras.




Original, noisy, NL-means, K-SVD Experiments: IPOL

Comparison of six state of the art methods, which combine the above mentioned principles: transform thresholding, nonlocal means and the Bayesian global estimator. Some are used in real cameras.


Original, noisy, BM3D and Non-local Bayes (selected by DxO) Experiments: IPOL

# Denoising in five formulas from local to global

**1-Transform thresholding: the example of DCT denoising**

**2-Neighborhood filters: an old and fantastic trick**

**3-A slight extension: nonlocal means**

**4-The Bayesian denoising paradigm from « non-local » to « global »**

**5-Machine learning algorithms: EPLL**

**6-Back to our starting point: dual denoising and transform thresholding**

***Where to test all algorithms:*** ***Image Processing on Line (IPOL)***
***http://www.ipol.im/***

# The patch space as a Gaussian mixture

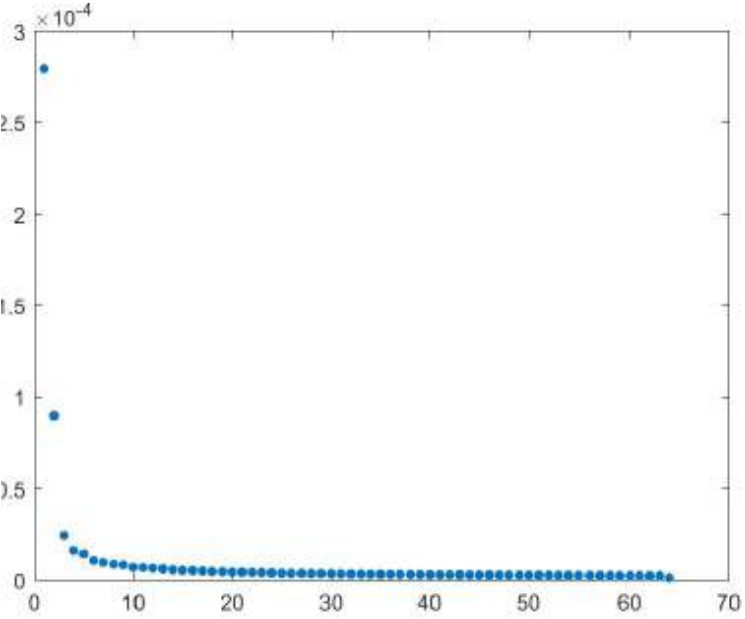
We used the database of images [70] composed of TIF images of size $768 \times 576$. In order to consider them noiseless, we zoomed out these images by a factor of 2. To do so, the images were filtered by a Gaussian kernel of standard deviation $0.8\sqrt{3}$ and one pixel out of 4 was selected. Among the 419 images of size $384 \times 288$ , we selected uniformly around 70% images to perform learning, the remaining 30% becoming a test dataset. We transformed them to grayscale images by extracting the $Y$ channel from the color images converted into the $YUV$ color space.

$$Y = 0.30R + 0.59G + 0.11B$$

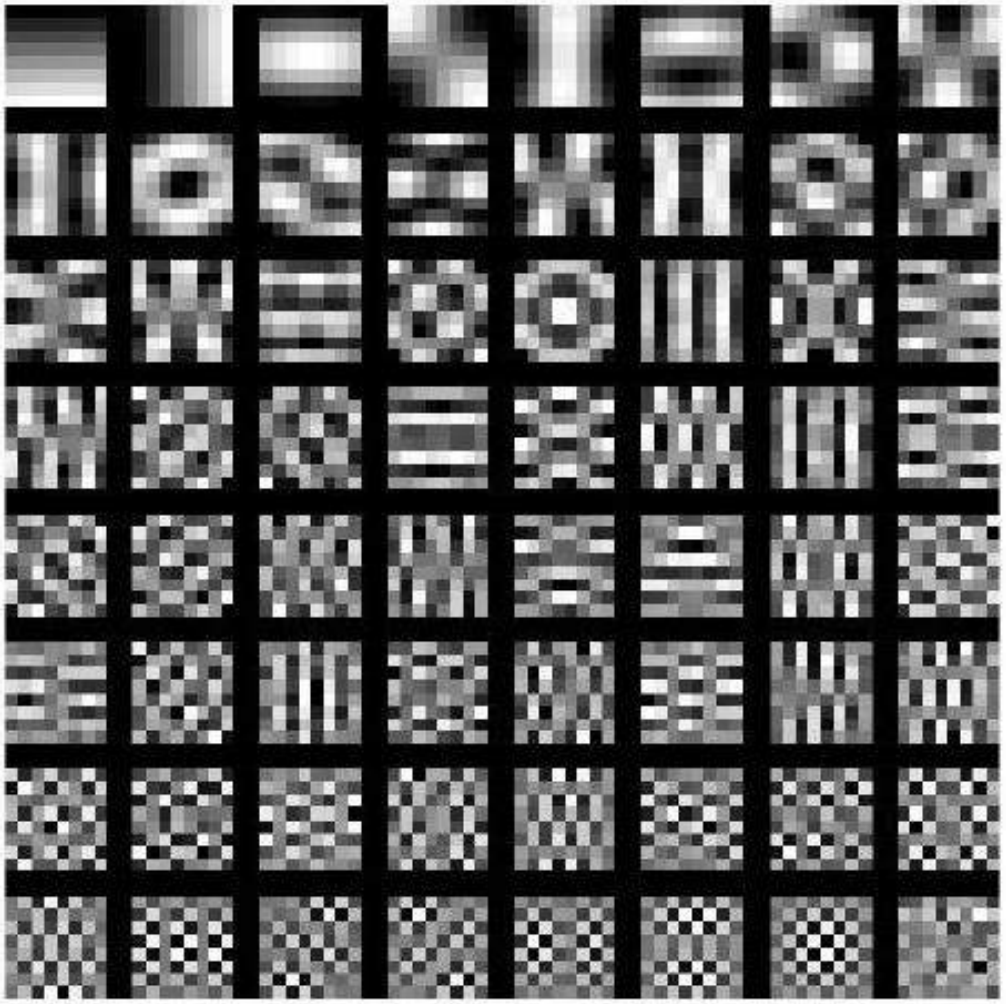
where R,G,B are the channels of the color image. The learning step was realized independently from the denoising step. The GMM was learned once, and then used for the different denoising tasks.

A patch $x$ is represented by a vector $x \in \mathbb{R}^d$. From this database, we extracted around $30 \times 10^6$ patches and removed their DC component, then we randomly selected $2 \times 10^6$ patches to perform EM.

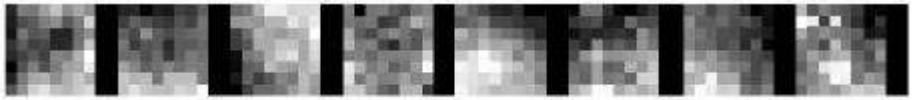
Zoran, Daniel, and Yair Weiss. "From learning models of natural image patches to whole image restoration." Computer Vision (ICCV), 2011 IEEE International Conference on. IEEE, 2011.




(a) eigenvalues sorted decreasingly

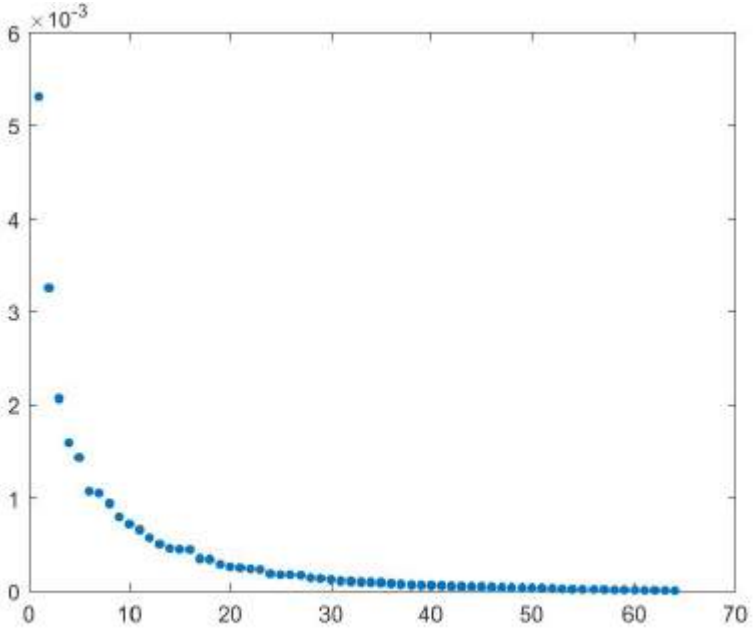


(b) corresponding eigenvectors

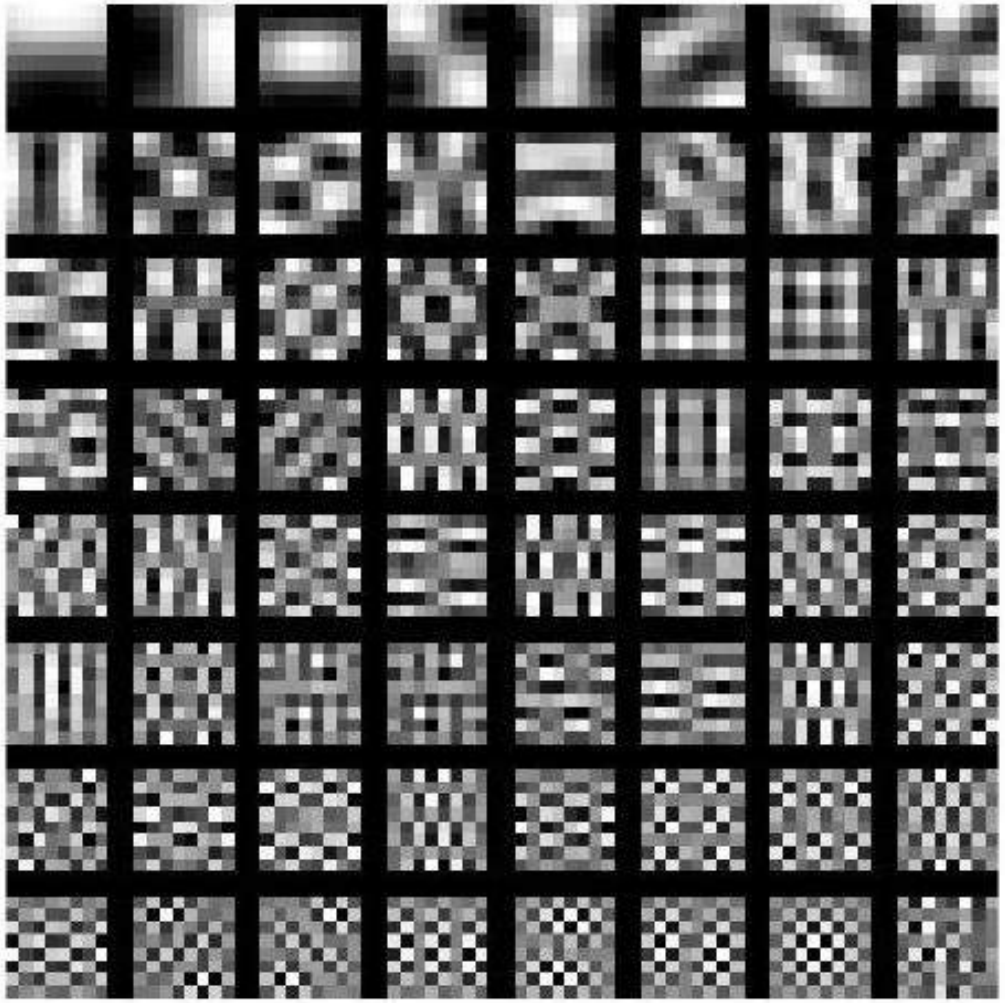


(c) 8 simulated patches

FIGURE 7.2: Component 1 of the Gaussian Mixture Model. The very low eigenvalues indicates that it represents flat patches.

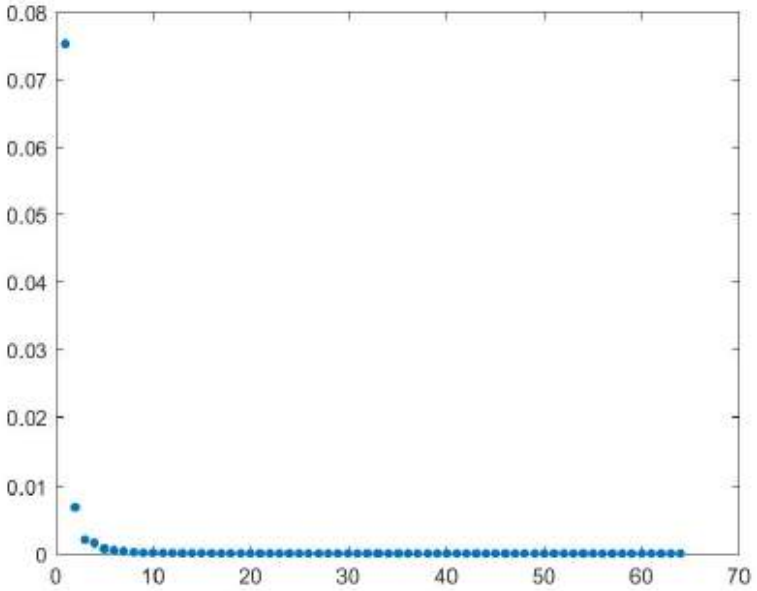



(a) eighenvalues sorted decreasingly

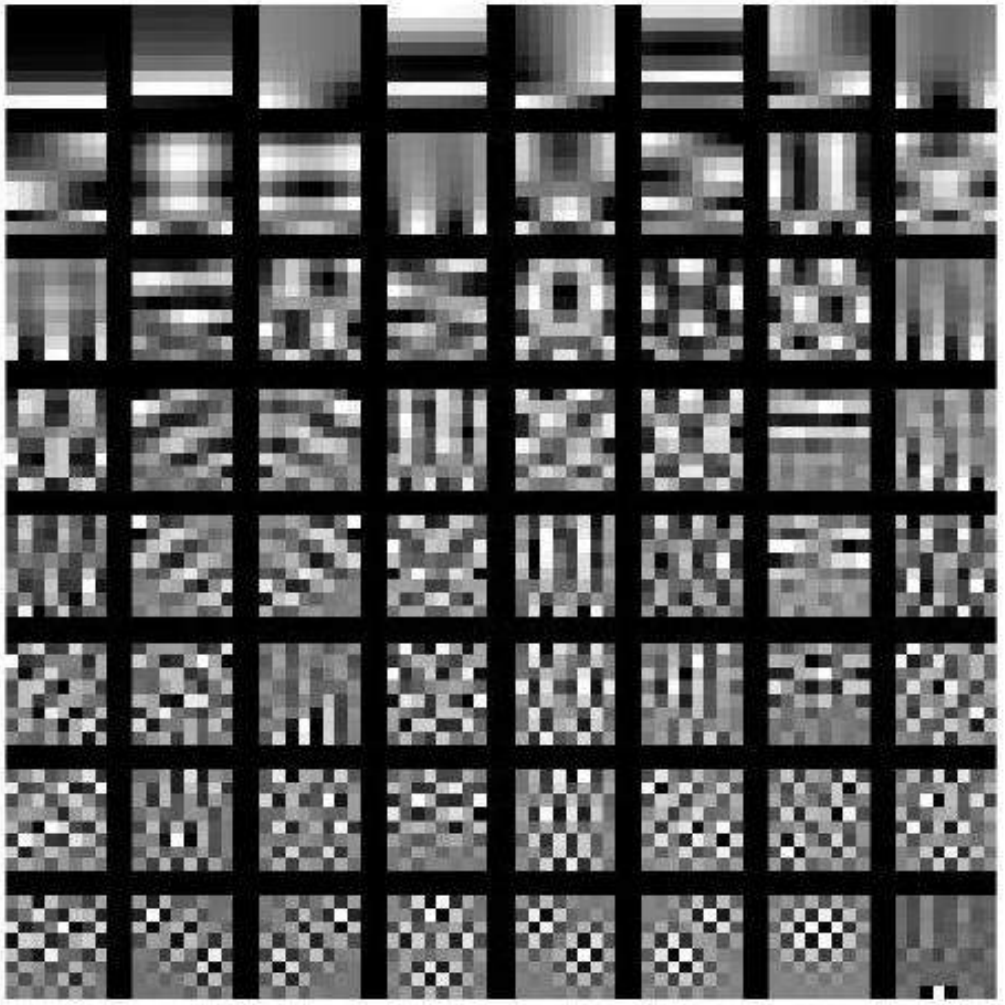


(b) corresponding eigenvectors

FIGURE 7.4: Component 13 of the Gaussian Mixture Model. The decay of the eigenvalues is slower and gives importance to many eigenvectors in the simulated patches : it therefore simulates a texture patch.



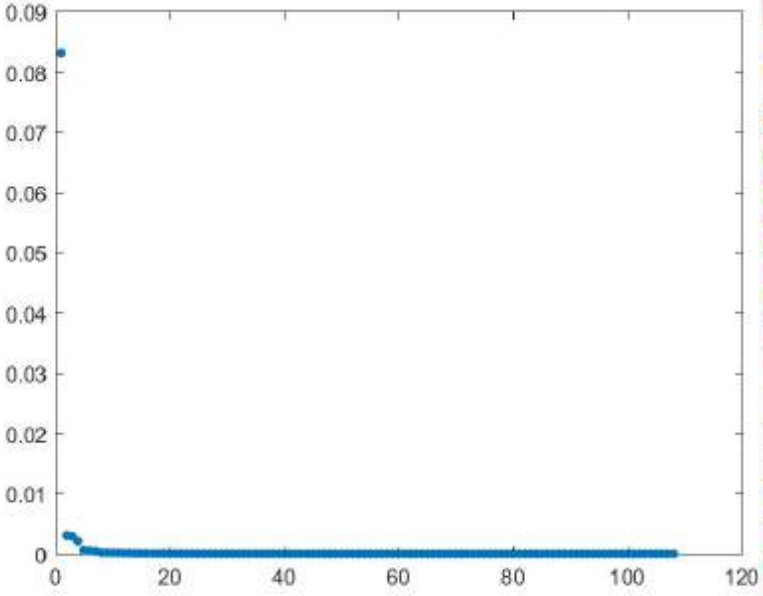

(a) eighenvalues sorted decreasingly



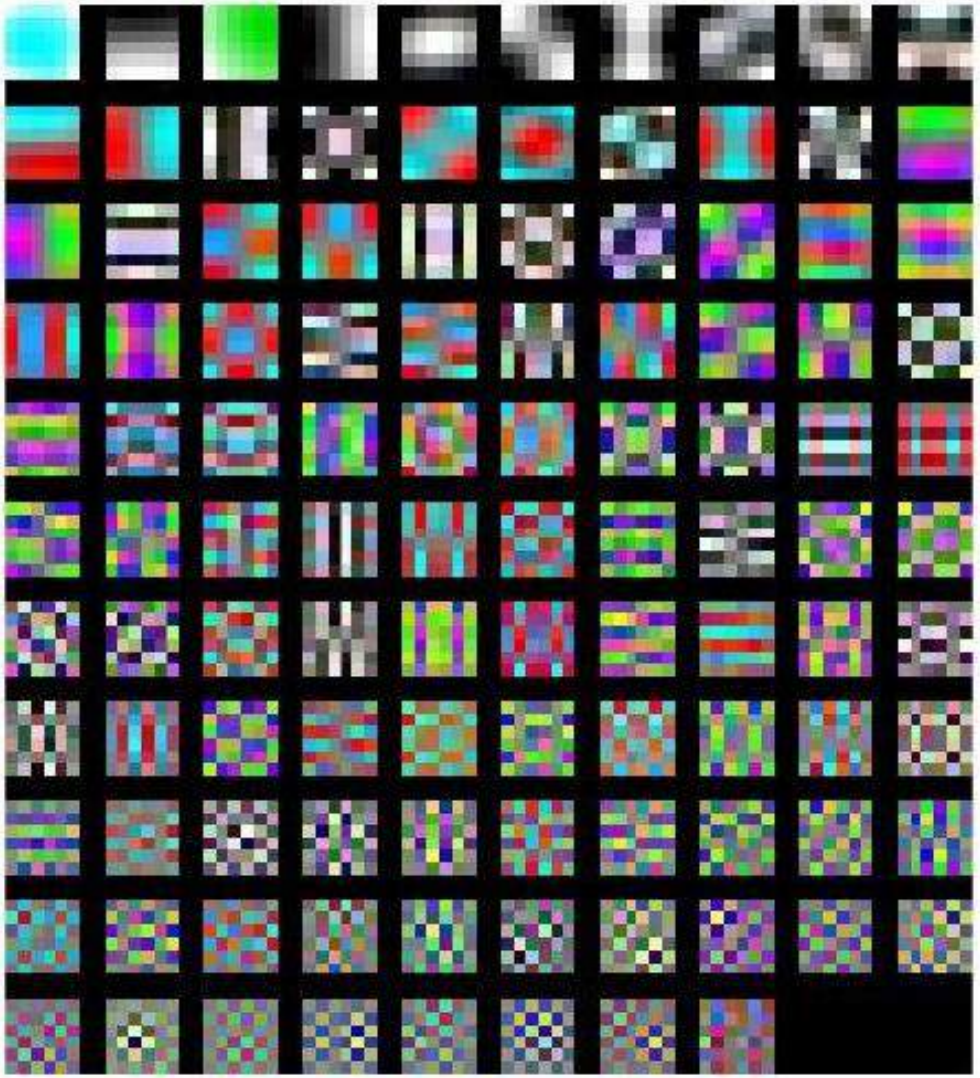
(b) corresponding eigenvectors

(c) 8 simulated patches

FIGURE 7.5: Component 14 of the Gaussian Mixture Model. The sharp decay of the eigenvalues gives importance only to the top eigenvectors in the simulated patches : it simulates patches with energy present mainly on the patch bottom.

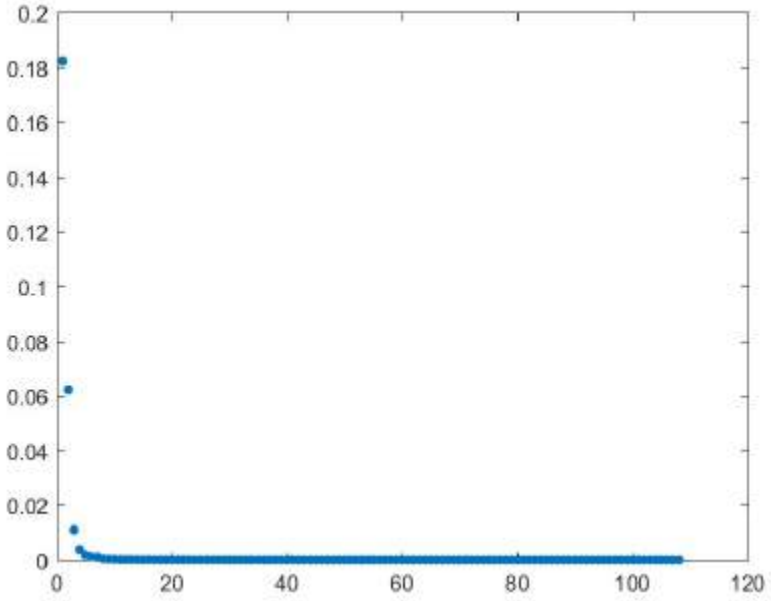



(a) eigenvalues sorted decreasingly

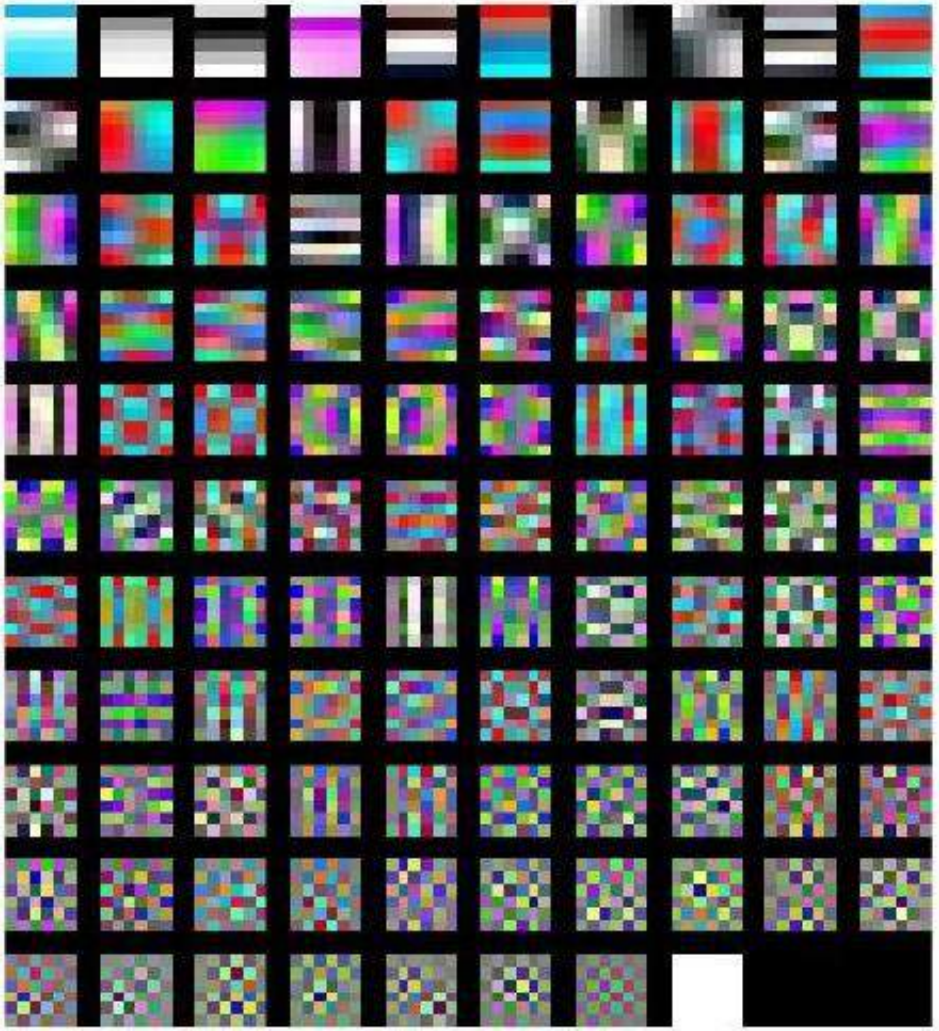


(c) 10 simulated patches

FIGURE 7.6: Component 2 of the Gaussian Mixture Model. The very low eigenvalues indicates that it represents flat patches with blue as a dominant color.



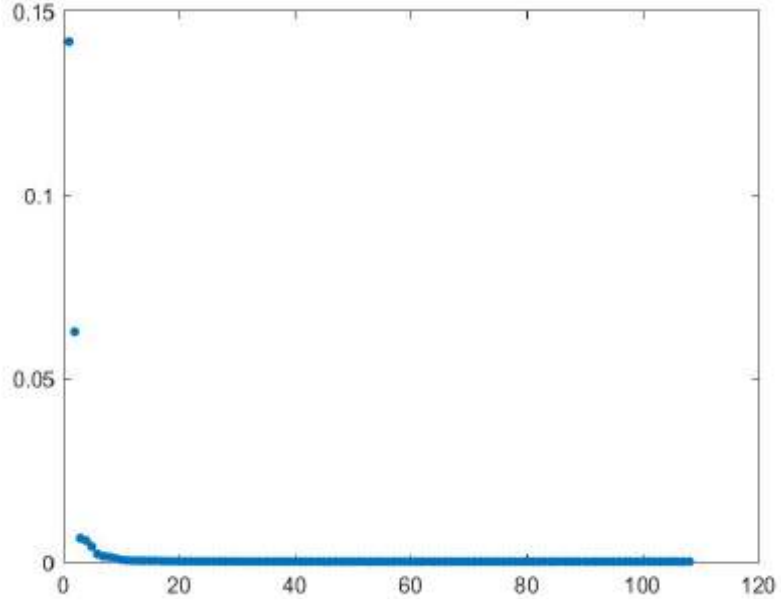

(a) eigenvalues sorted decreasingly



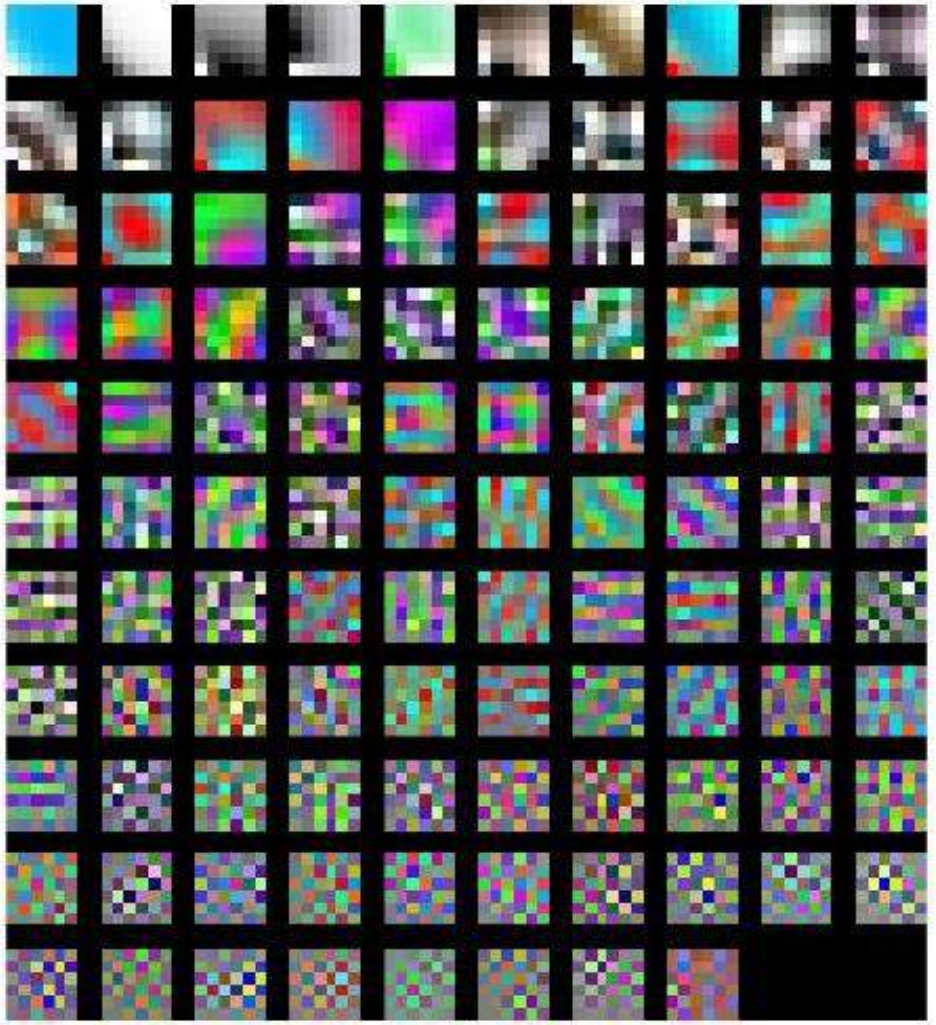
(b) corresponding eigenvectors

(c) 10 simulated patches

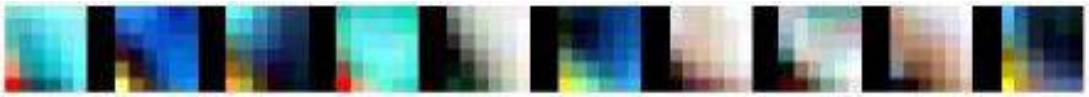
FIGURE 7.7: Component 6 of the Gaussian Mixture Model. The sharp decay of the eigenvalues gives importance only to the top eigenvectors in the simulated patches. As illustrated by the simulated patches (c) this Gaussian models a patch with a horizontal edge near its top.




(a) eigenvalues sorted decreasingly

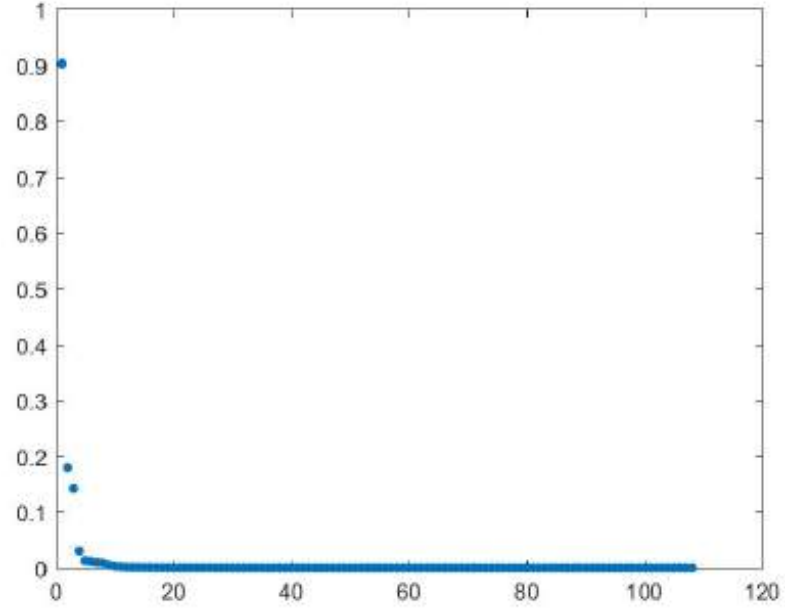


(b) corresponding eigenvectors

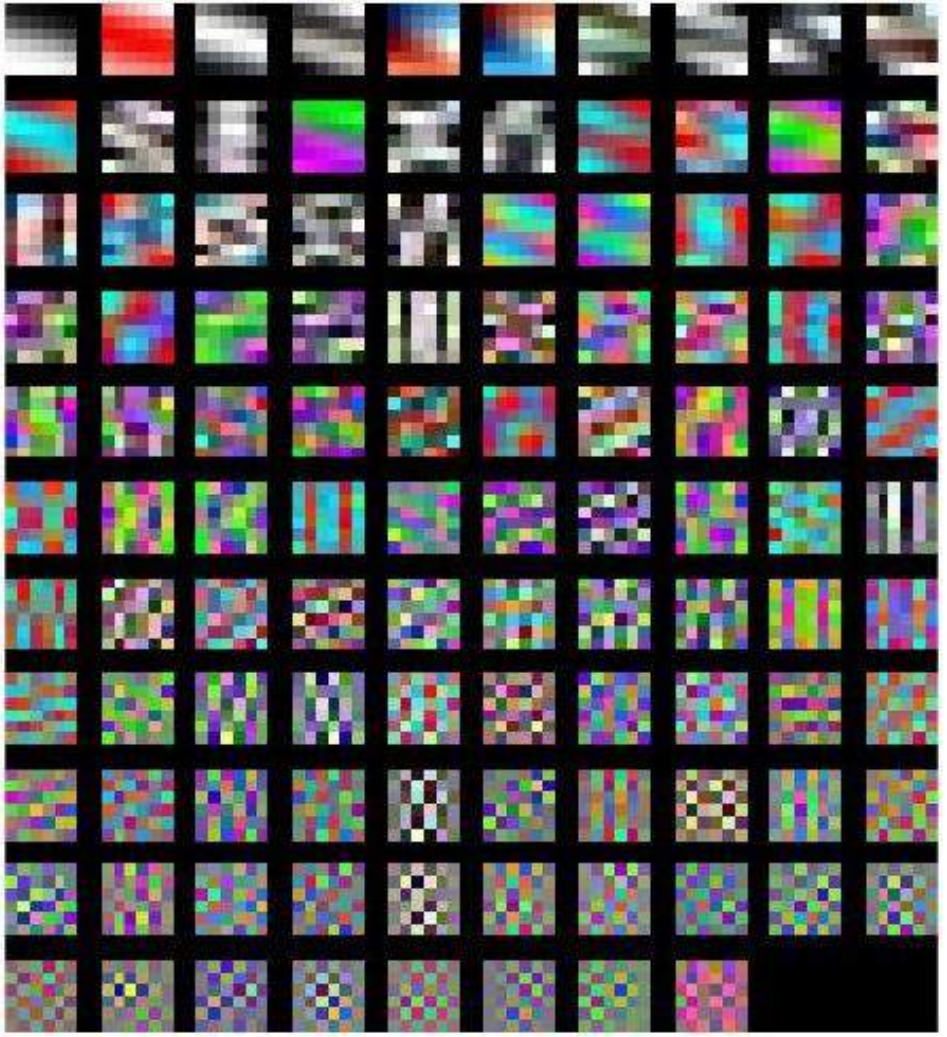


(c) 10 simulated patches

FIGURE 7.8: Component 34 of the Gaussian Mixture Model. The sharp decay of the eigenvalues gives importance only to the top eigenvectors in the simulated patches. As illustrated by the simulated patches (c) this Gaussian models a patch with a highlight in the bottom left corner.




(a) eigenvalues sorted decreasingly



(b) corresponding eigenvectors

(c) 10 simulated patches

FIGURE 7.9: Component 45 of the Gaussian Mixture Model. The sharp decay of the eigenvalues gives importance only to the top eigenvectors in the simulated patches : as shown by the simulated patches (c), this Gaussian models diagonal edges.

# The patch space as a Gaussian mixture and the EPLL algorithm

The patch prior we shall use is a finite Gaussian mixture model (GMM) learned as described in section 7.4. The probability of a patch $P$, represented by a vector of $\mathbb{R}^d$, is assumed to satisfy

$$\mathbb{P}(P) = \sum_{k=1}^{K} \pi_k \mathcal{N}(P|\mu_k, \Sigma_k).$$

Here, $\mathcal{N}(P|\mu_k, \Sigma_k)$ is a $d$-dimensional Gaussian density (a patch in the image is $\sqrt{d} \times \sqrt{d}$) of mean $\mu_k$ $(d \times 1)$ and covariance matrix $\Sigma_k$ $(d \times d)$. The coefficients $\pi_k$ represent the mixing weights (or probabilities) for each mixture component and therefore satisfy $\sum_k \pi_k = 1$.

Learning of the Gaussian mixture was performed using the Expectation Maximization (EM) algorithm computed over $N$ natural patches $P_n$ with their DC component [1] removed. Removing the DC component makes learning easier as it removes the dominant dimension from the patch space.

1. The DC component corresponds to the mean intensity value of the patch

### 9.1.2 Denoising of grayscale images

#### The energy to minimize

Solving a maximum a posteriori (MAP) problem to build a clean patch $P$ given a noisy patch $\tilde{P}$ and a local prior $\mathbb{P}$ would lead to the following energy to minimize :

$$E(P|\tilde{P}) = \frac{||P - \tilde{P}||^2}{2\sigma^2} - \log(\mathbb{P}(P)).$$

A patch position in $\Omega$ can be identified with the projection operator $P$ extracting the corresponding pixels from any image $U$ defined on $\Omega$. We call $\mathcal{P}$ the set of such patch projectors. The patches of $U$ therefore are the family $(PU)_{P\in\mathcal{P}}$. We then define the expected log likelihood of the image

$$\text{EPLL}_{\mathbb{P}}(U) = \sum_{P\in\mathcal{P}} \log \mathbb{P}(PU). \tag{9.1}$$

Given the noisy image $\tilde{U}$, the energy we propose to minimize in order to find the denoised image using the patch prior $\mathbb{P}$ is

$$E(U|\tilde{U}) = \sum_{P\in\mathcal{P}} \frac{||PU - P\tilde{U}||^2}{2\sigma^2} - \text{EPLL}_{\mathbb{P}}(U) \tag{9.2}$$

$$= \sum_{\mathbf{i}\in\Omega} N_{\mathbf{i}} \frac{||U_{\mathbf{i}} - \tilde{U}_{\mathbf{i}}||^2}{2\sigma^2} - \text{EPLL}_{\mathbb{P}}(U) \tag{9.3}$$

$$= \sum_{\mathbf{i}\in\Omega} N_{\mathbf{i}} \frac{||U_{\mathbf{i}} - \tilde{U}_{\mathbf{i}}||^2}{2\sigma^2} - \sum_{P\in\mathcal{P}} \log \mathbb{P}(PU). \tag{9.4}$$

where $N_{\mathbf{i}}$ represents the number of overlapping patches in which the pixel $\mathbf{i}$ appears.

| $\sigma$ | K=200 | K=100 | K=50 |
|---|---|---|---|
| 5 | 40.443 | 40.379 | 40.298 |
| 10 | 36.575 | 36.523 | 36.447 |
| 20 | 32.915 | 32.893 | 32.840 |
| 30 | 30.814 | 30.765 | 30.743 |
| 40 | 29.385 | 29.185 | 29.178 |
| 50 | 28.307 | 27.854 | 27.861 |

TABLE 9.1 – Influence of $K$ on the PSNR of the result

| $5 \leq \sigma < 30$ | $30 \leq \sigma$ |
|---|---|
| $K = 100$ | $K = 200$ |

**Size of the dataset of patches $N$**

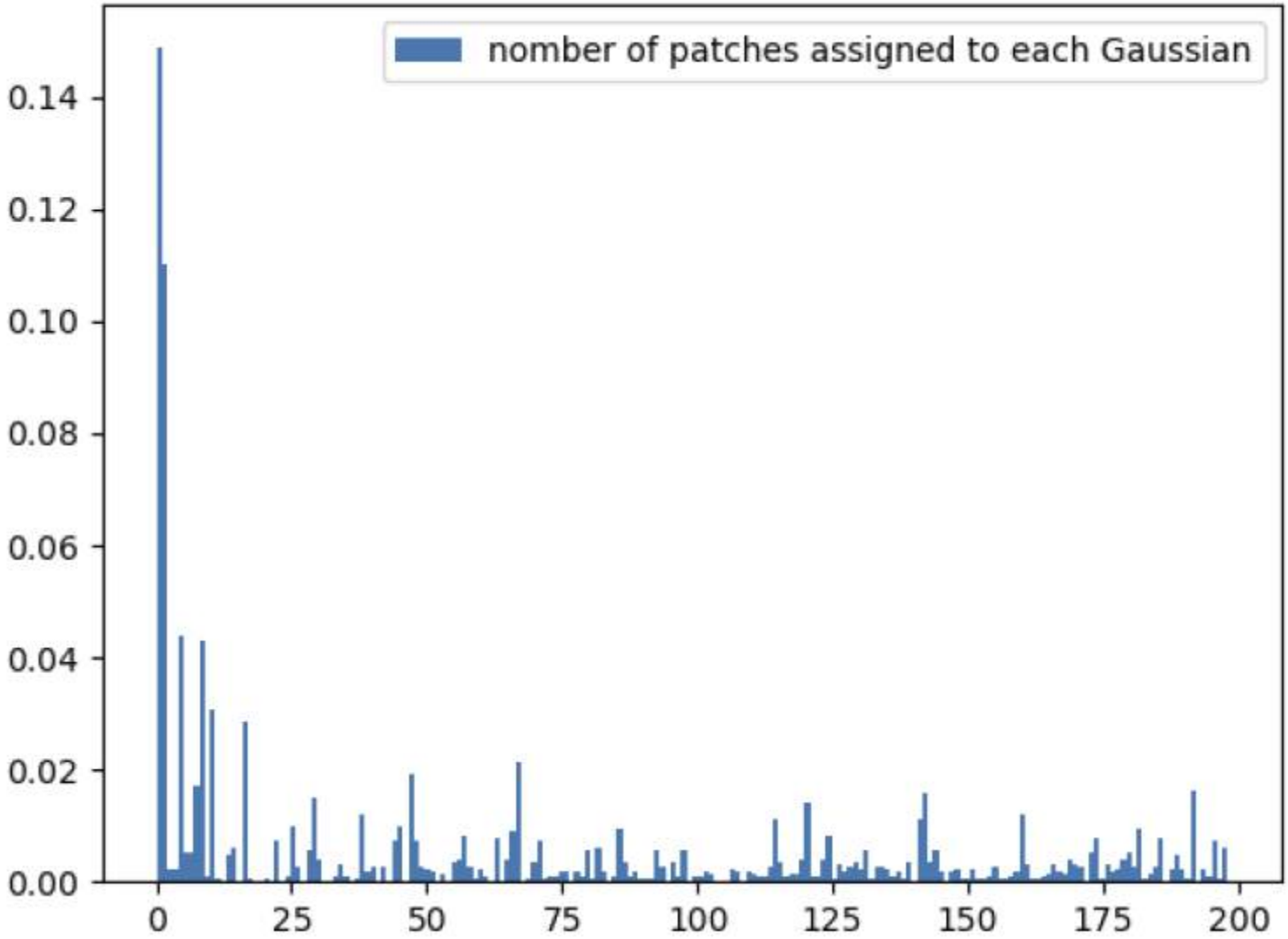
In continuation, we fixed $K = 200$ and modified $N$ to obtain table 9.2.

| $\sigma$ | $N = 5.10^5$ | $N = 1.10^6$ | $N = 2.10^6$ |
|---|---|---|---|
| 5 | 40.277 | 40.281 | 40.286 |
| 10 | 36.303 | 36.307 | 36.306 |
| 20 | 32.480 | 32.482 | 32.476 |
| 30 | 30.237 | 30.238 | 30.321 |
| 40 | 28.570 | 28.567 | 28.843 |
| 50 | 27.073 | 27.066 | 27.692 |

TABLE 9.2 – Influence on the PSNR of $N$, the number of patches used for learning the GMM (with $K$ fixed to 200). It follows that $N = 2.10^6$ yields the best performance in presence of high noise.

## 9.4 Denoising of low frequencies

In the denoising process, a large part of the images have a majority of homogeneous patches, the "flat" Gaussians will therefore be often chosen to realize the MAP estimation : it will have the highest conditional mixing weight $\pi'_k$ for numerous parches. In figure 9.1, the histograms represent, for the various steps of the denoising of the image Lena with $\sigma = 20$, the rate of patches assigned to each Gaussian.


nomber of patches assigned to each Gaussian
0.14
0.12
0.10
0.08
0.06
0.04
0.02
0.00
0
25
50
75
100
125
150
175
200

| $\sigma$ | EPLL | MS EPLL |
|---|---|---|
| 5 | **43.389** | 43.325 |
| 10 | **39.513** | 39.476 |
| 20 | 35.592 | **35.652** |
| 30 | 33.308 | **33.471** |
| 40 | 31.662 | **31.909** |

TABLE 9.7 – Influence of multi-scale denoising.

(a) Original image

(b) noisy image $\sigma = 30$

(c) EPLL PSNR=33.256

(d) **MS EPLL PSNR=33.451**

FIGURE 9.2 – Visual comparison between single and multi-scale denoising on an image from the test set.

## 9.6 Taking the clean image as oracle

To have an idea of the optimal performance that the algorithm might reach, we run it using the clean image as oracle. At each iteration (each value of $\beta$), we use the conditional mixing weight obtained from the clean image instead of taking it from the current image estimate. Therefore, each patch has always the same Gaussian assigned to calculate the MAP estimate. Table 9.8 gives the results on the image *Lena*.

| $\sigma$ | Normal | Oracle |
|---|---|---|
| 5 | 37.96 | 38.41 |
| 10 | 33.96 | 34.63 |
| 20 | 30.58 | 31.44 |
| 30 | 28.59 | 29.41 |
| 40 | 27.13 | 27.80 |

TABLE 9.8 – Denoising taking the clean image as oracle

This experiments show that :

— The learned GMM has a very good denoising potential.

— One of the main difficulties is to determine the right Gaussian to calculate the MAP estimate.

## Denoising channel by channel

$$A_{YUV} = \begin{pmatrix} 0.30 & 0.59 & 0.11 \\ -0.15 & -0.29 & 0.44 \\ 0.61 & -0.51 & -0.10 \end{pmatrix}$$

$$A_{OPP} = \begin{pmatrix} \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \\ \frac{1}{2} & 0 & -\frac{1}{2} \\ \frac{1}{4} & -\frac{1}{2} & \frac{1}{4} \end{pmatrix}$$

| $\sigma$ | RGB | YUV | OPP | GMM color |
|---|---|---|---|---|
| 5 | 41.300 | 41.954 | 42.239 | **42.508** |
| 10 | 37.367 | 38.066 | 38.457 | **38.716** |
| 20 | 33.544 | 34.316 | 34.745 | **34.944** |
| 30 | 31.321 | 32.150 | 32.609 | **32.690** |
| 40 | 29.757 | 29.909 | **31.106** | 31.069 |
| 50 | 28.560 | 29.509 | **29.910** | 29.766 |
| 60 | 28.560 | 29.509 | **29.910** | 29.766 |

TABLE 9.9 – Comparison between the different color denoising methods

(a) Original image

(b) noisy image $\sigma = 40$

(c) EPLL PSNR=31,124

(d) **MS EPLL PSNR=31.255**

FIGURE 9.3 – Single and multi-scale color denoising

(a) Original image

(b) noisy image $\sigma = 30$

(c) BM3D PSNR=29.89

(d) NL-Bayes PSNR=29.719

(c) BM3D PSNR=29.89

(d) NL-Bayes PSNR=29.719

(e) DCT PSNR=29.175

(g) EPLL PSNR=29.886

(a) Original image

(b) noisy image $\sigma = 30$

(c) K-SVD PSNR=32.847

(d) BM3D PSNR=33.950

(e) NLBayes PSNR=32.833

(f) DCT PSNR=32.941

(a) Original image

(b) noisy image $\sigma = 30$

(c) K-SVD PSNR=32.847

(d) BM3D PSNR=33.950

(g) GMM-color PSNR=33.466

(h) OPP PSNR=33.477

(a) Original image

(b) noisy image $\sigma = 60$

(d) BM3D PSNR=29.52

(e) NLBayes PSNR=29.59

(e) NLBayes PSNR=29.59

(f) DCT PSNR= 28.42

(g) GMM-color PSNR=29.516

(h) OPP PSNR=29.610

(a) Original image

(b) noisy image $\sigma = 50$

(d) BM3D PSNR=26.44

(e) NLBayes PSNR=26.83

(e) NLBayes PSNR=26.83

(f) DCT PSNR=26.00

(g) GMM-color PSNR=25.78

(h) OPP PSNR= 26.14

# Denoising in five formulas from local to global

**1-Transform thresholding: the paradigmatic example of DCT denoising**

**2-Neighborhood filters: an old and fantastic trick**

**3-A slight extension: nonlocal means**

**4-The Bayesian denoising paradigm from « non-local » to « global »**

**5-Machine learning algorithms: Multilayer perceptron**

**6-Back to our starting point: dual denoising and transform thresholding**

***Where to test all algorithms:*** ***Image Processing on Line (IPOL)***
***http://www.ipol.im/***

Image denoising can be described as the problem of mapping from a noisy image to a noise-free image.

The best currently available denoising methods approximate this mapping with cleverly engineered algorithms.

This mapping can be learnt directly with a plain multi layer perceptron (MLP) applied to image patches!

*Multilayer perceptron with four hidden layers of size 2047 and a patch size of 17 x 17 on 362 million training samples, requiring approximately one month of computation time on a GPU*.

The training uses clean and noisy patches and is done separately for each noise level.

**Harold C. Burger, Christian J. Schuler, and Stefan Harmeling, CVPR 2012**
**Image denoising: Can plain Neural Networks compete with BM3D?**

clean (name: 008934)

noisy ($\sigma = 25$)PSNR:20.16dB

BM3D: PSNR:29.65dB

ours: PSNR:30.03dB

clean (name: barbara)

noisy ($\sigma = 25$)PSNR:20.19dB

BM3D: PSNR:30.67dB

ours: PSNR:29.21dB

# A dramatic turn of events in 2012-2013: learning denoising directly

| image | GSM [18] | KSVD [1] | BM3D [3] | us |
|---|---|---|---|---|
| Barbara | 27.83dB | *29.49dB* | **30.67dB** | 29.21dB |
| Boat | 29.29dB | 29.24dB | *29.86dB* | **29.89dB** |
| C.man | 28.64dB | 28.64dB | **29.40dB** | *29.32dB* |
| Couple | 28.94dB | 28.87dB | *29.68dB* | **29.70dB** |
| F.print | 27.13dB | 27.24dB | **27.72dB** | *27.50dB* |
| Hill | 29.26dB | 29.20dB | *29.81dB* | **29.82dB** |
| House | 31.60dB | 32.08dB | **32.92dB** | *32.50dB* |
| Lena | 31.25dB | 31.30dB | *32.04dB* | **32.12dB** |
| Man | 29.16dB | 29.08dB | *29.58dB* | **29.81dB** |
| Montage | 30.73dB | 30.91dB | **32.24dB** | *31.85dB* |
| Peppers | 29.49dB | 29.69dB | *30.18dB* | **30.25dB** |

Table 1. PSNRs (in dB) on standard test images, $\sigma = 25$.

This table shows that the four layer (with 2047 neurones) MLP trained on 362 millions 17x17 patches reaches the « state of the art »

**Harold C. Burger, Christian J. Schuler, and Stefan Harmeling, CVPR 2012**
**Image denoising: Can plain Neural Networks compete with BM3D?**


**The oldest heliographic engraving known in the world**, a reproduction of a 17th century Flemish engraving. Nicéphore Niépce in 1825, ( Bibliothèque nationale de France).

# Denoising in five formulas from local to global

**1-Transform thresholding: the paradigmatic example of DCT denoising**

**2-Neighborhood filters: an old and fantastic trick**

**3-A slight extension: nonlocal means**

**4-The Bayesian denoising paradigm from « non-local » to « global »**

**5-Machine learning algorithms**

**6-Back to our starting point: dual denoising and transform thresholding**

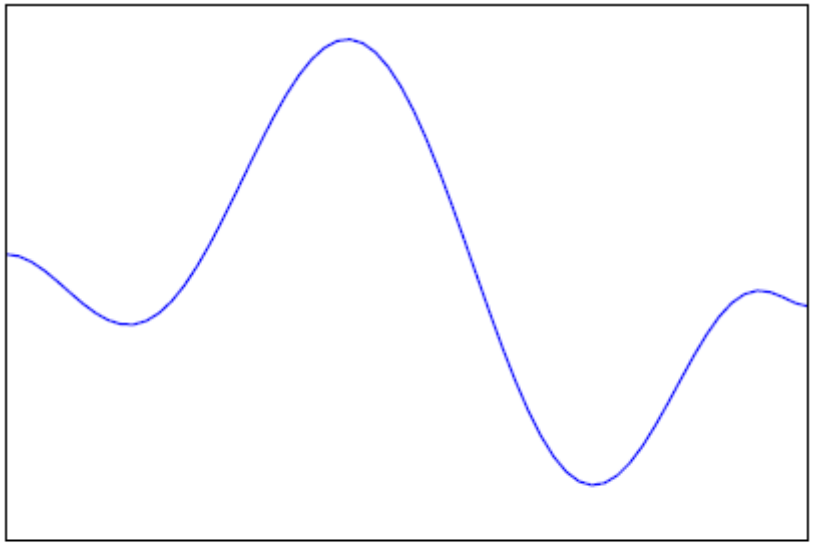
***Where to test all algorithms:*** ***Image Processing on Line (IPOL)***
***http://www.ipol.im/***


73

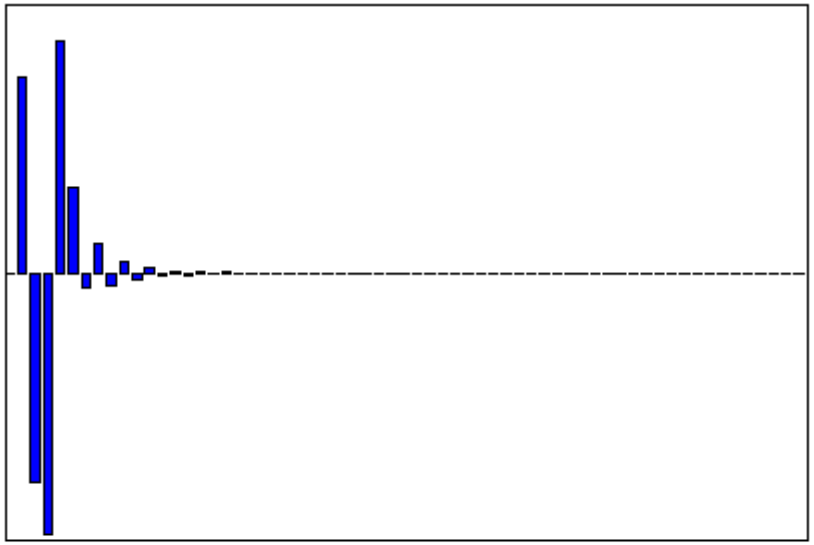
```
function x = DDID(y, sigma2)
    x = step(y, y, sigma2, 15, 7, 100, 4.0);
    x = step(x, y, sigma2, 15, 7, 8.7, 0.4);
    x = step(x, y, sigma2, 15, 7, 0.7, 0.8);
end

function xt = step(x, y, sigma2, r, sigma_s, gamma_r, gamma_f)

    [dx dy] = meshgrid(-r:r);
    h = exp(- (dx.^2 + dy.^2) / (2 * sigma_s^2));
    xp = padarray(x, [r r], 'symmetric');
    yp = padarray(y, [r r], 'symmetric');
    xt = zeros(size(x));

    parfor p = 1:numel(x), [i j] = ind2sub(size(x), p);

        % Spatial Domain: Bilateral Filter
        g  = xp(i:i+2*r, j:j+2*r);
        y  = yp(i:i+2*r, j:j+2*r);
        d  = g - g(1+r, 1+r);
        k  = exp(- d.^2 ./ (gamma_r * sigma2)) .* h;     % Eq. 4
        gt = sum(sum(g .* k)) / sum(k(:));               % Eq. 2
        st = sum(sum(y .* k)) / sum(k(:));               % Eq. 3

        % Fourier Domain: Wavelet Shrinkage
        V  = sigma2 .* sum(k(:).^2);                     % Eq. 5
        G  = fft2(ifftshift((g - gt) .* k));             % Eq. 6
        S  = fft2(ifftshift((y - st) .* k));             % Eq. 7
        K  = exp(- gamma_f * V ./ (G .* conj(G)));       % Eq. 9
        St = sum(sum(S .* K)) / numel(K);                % Eq. 8

        xt(p) = st + real(St);                           % Eq. 1
    end
end
```

**Algorithm 1**: *MATLAB code of Dual-Domain Image Denoising. This code reproduces all grayscale images in this paper.*

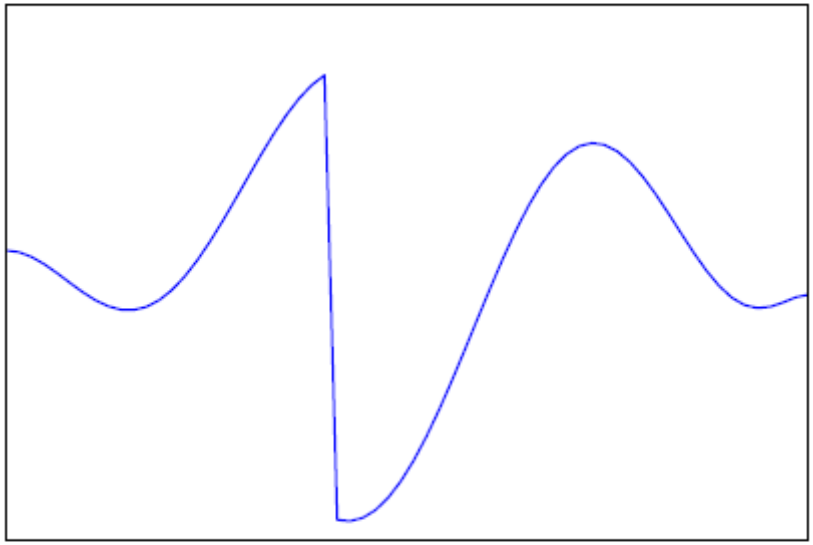
**Dual domain denoising Claude Knaus, Matthias Zwicker, ICIP 2013**



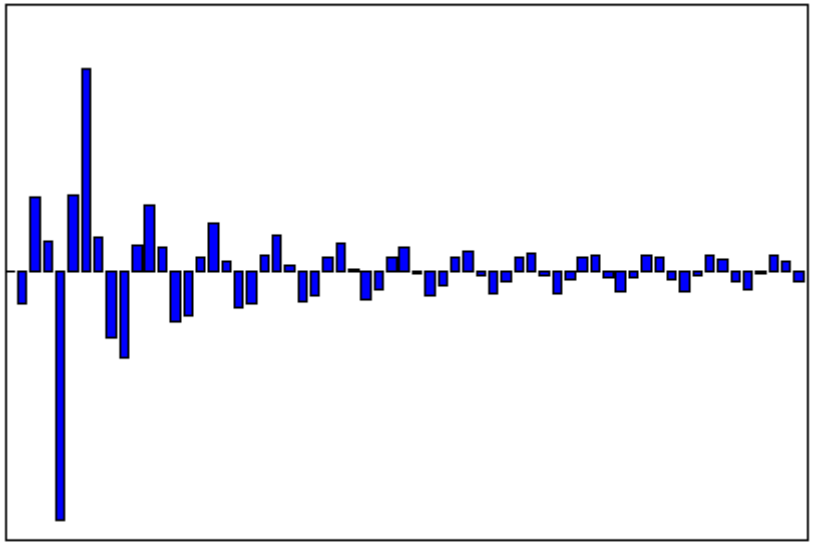
(a) Smooth signal



(b) DCT of smooth signal



(c) Signal with discontinuity (edge)



(d) DCT of signal with discontinuity

Figure 2.6 – Behaviour of DCT coefficients. Signals containing discontinuities have their energy less concentrated in the DCT domain. This makes the DCT basis less effective for denoising purposes in presence of edges.

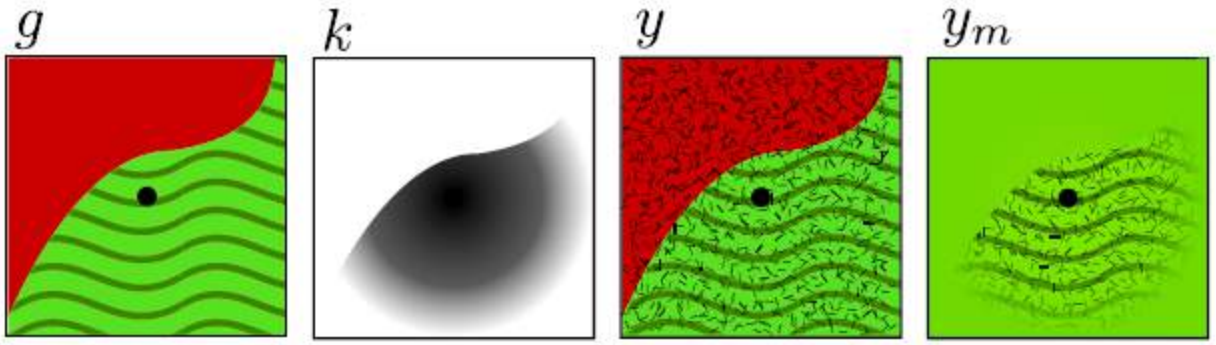
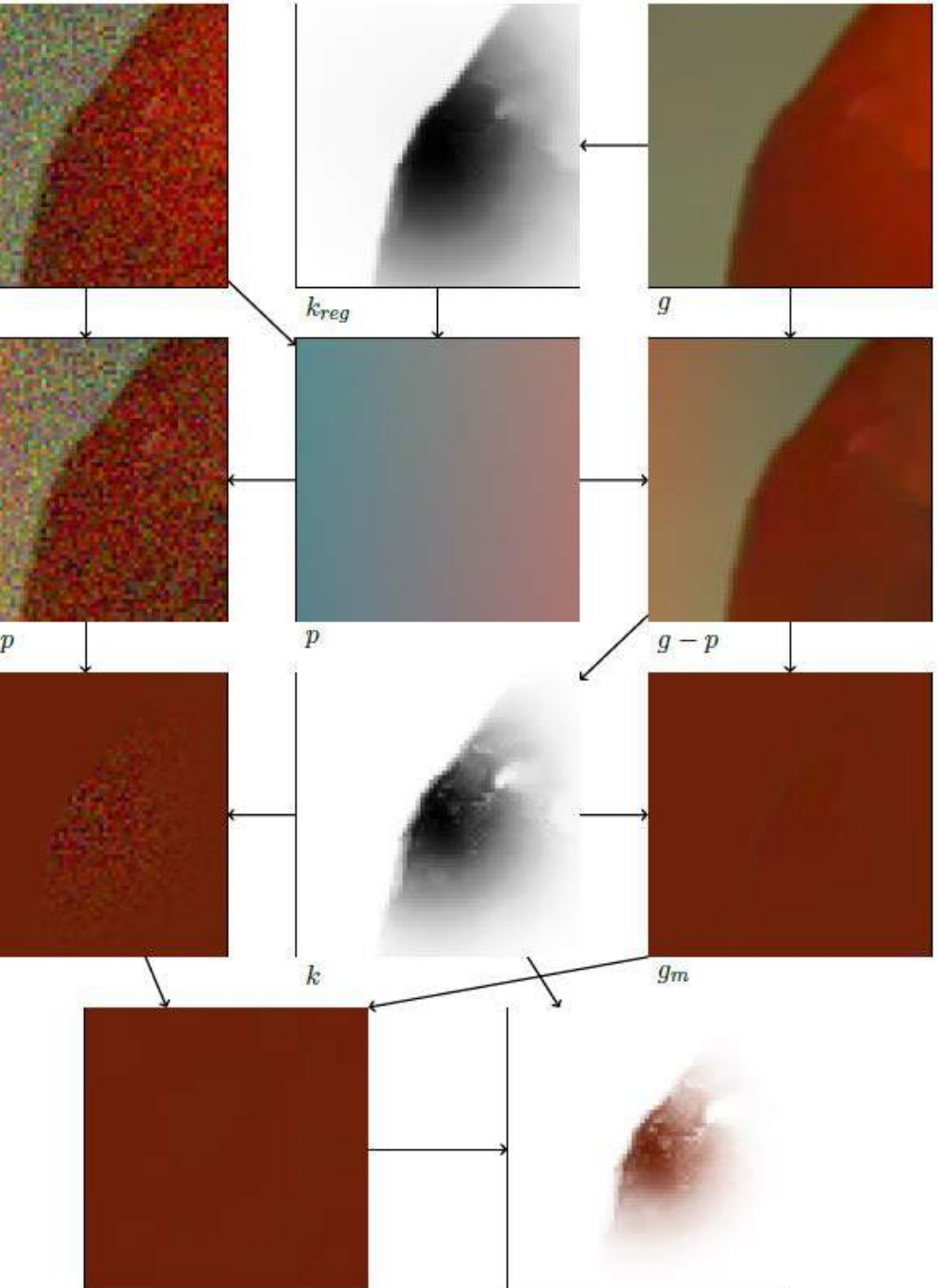



**Fig. 2.** Illustration of DDID's preprocessing of a patch. The kernel $k$ is computed using the guide $g$. In the modified patch $y_m$ all object discontinuities have been removed, leaving only the texture information corresponding to the object selected by the kernel $k$. The removed pixels are replaced by $\tilde{s}$: the average of the *meaningful* portion of the patch.

**This explanation of Dual denoising comes from:**

**Non-local dual image denoising**

N. Pierazzo, M. Lebrun, M. Rais, and G. Facciolo, ICIP 2014



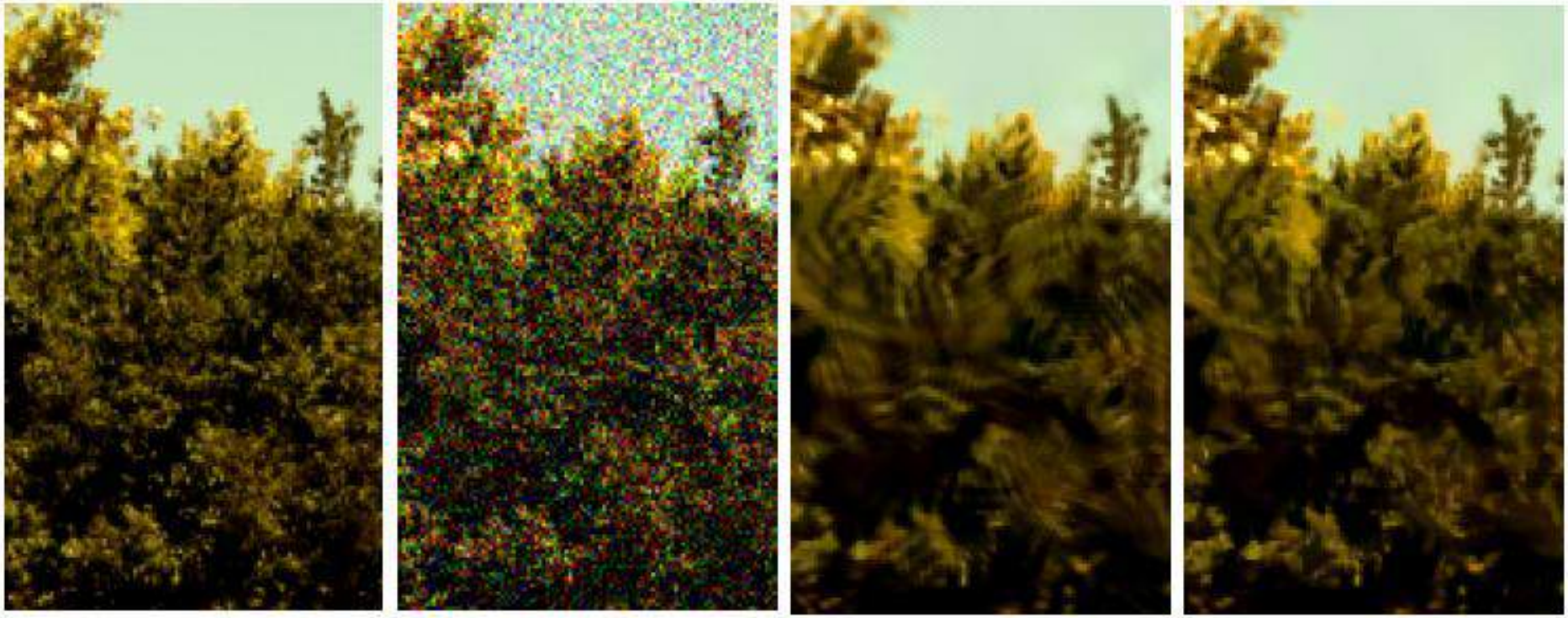

**Steps DA3D**
**(Data adaptive dual domain denoising)**

This figure shows what happens to a noisy patch taken in a natural image, containing an edge. The arrows indicate the elements needed to compute every step of the algorithm. Notice that, thanks to the weight function, the useful part of the patch is kept, while the discontinuities are completely removed.

Taken from

**N. Pierazzo, PhD thesis, 2016**

**Fig. 3**. Artifacts in DDID. From left to right: the noisy image (with $\sigma = 30$), the result of the first, second, and last iteration of the algorithm.



(a) Original (b) Noisy (c) DDID (d) NLDD

**Fig. 1**. A detail of the artifacts produced by DDID and the corresponding result of NLDD. In this example $\sigma = 30$.

**Pierazzo, N., Lebrun, M., Rais, M. E. & Facciolo, G. (2014, October). Non-local dual image denoising. ICIP 2014**

**N. Pierazzo, PhD thesis, 2016**

**On line demo: http://dev.ipol.im/~pierazzo/ipol_demo/ddmd/**

**Joseph-Nicéphore Niépce (1765-1833): first indoor photograph,**
Denoised by the Noise Clinic, IPOL (Image Processing on Line www.ipol.im)

**Joseph-Nicéphore Niépce (1765-1833): first indoor photograph,**
Denoised by the Noise Clinic, IPOL (Image Processing on Line www.ipol.im)

It has been long admitted that the structure of 2D functions is described by local characteristics, for example a local Fourier or wavelet expansion, or more trivially a Taylor expansion of some order. The regularity of the function would be encoded in the decay of a local expansion or in the boundedness of some norm measuring regularity (Sobolev, Besov, BV,…).

Image processing has strayed away from this model inherited from harmonic analysis and geometric measure theory. It looks directly at the "patch space".

Patches are 8 * 8 or 10 * 10 square images cropped from any image. Image characteristics seem to be better described in the patch space (of dimension 64, 100,…). This is a dimension reduction (from the space of images that would have a dimension of several millions), or on the contrary a dimension lifting from two dimensions (the image) to many more.

Can we explore the patch space and find some evidence about its regularity? This is an experimental question, because we can now analyze patches by billions. Still, this is a far too small number to sample a space in such a high dimension, unless it shows some regularity.

Even the sparse information that we have gathered on the patch space changes our view of image perception.

I'll illustrate it on two classic image processing problems: image denoising and anomaly detection.

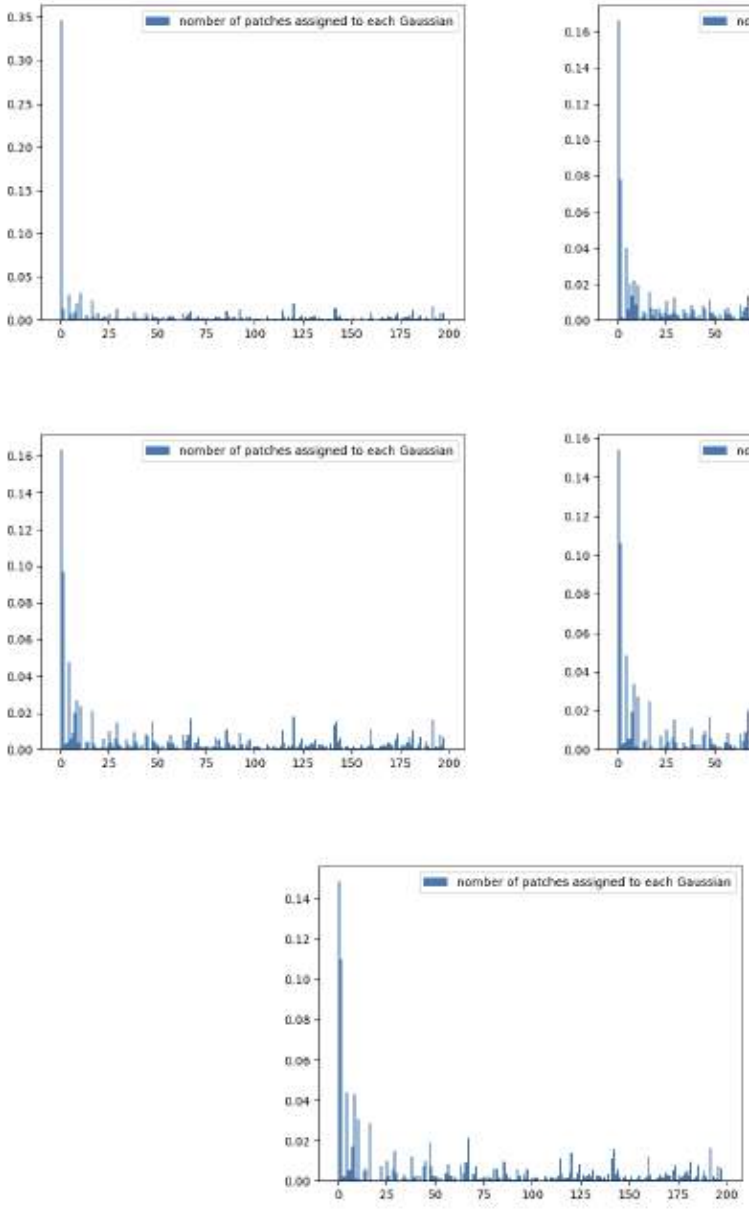



FIGURE 9.1 – At each iteration $\beta = (1, 4, 8, 16, 32)$, number of Gaussians applied to each component of the GMM

Results and PSNR of DCT denoising without oracle (1step), with oracle (2step), and with multiscale using patches of sizes 4 4, 8 8, and 16 16.

DCT4 1step (26.10 dB)

DCT4 2step (25.84 dB)

MS DCT4 2step (**27.75** dB

DCT8 1step (27.32 dB)

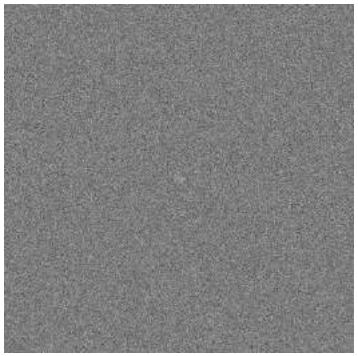
DCT8 2step (27.63 dB)

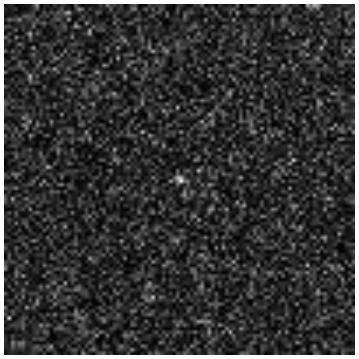
MS DCT8 2step (**27.99** dB

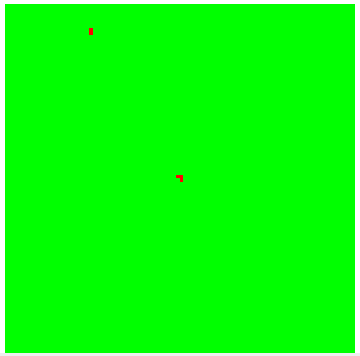
DCT16 1step (27.41 dB)

DCT16 2step (**27.77** dB)

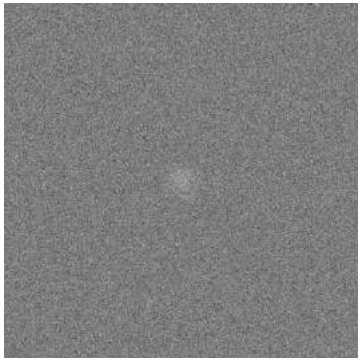
MS DCT16 2step (27.73 dB

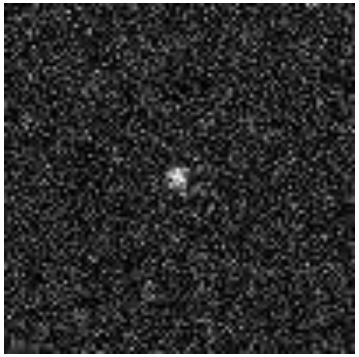


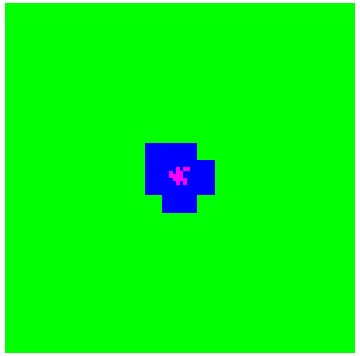




Minimum log NFA = -10.7

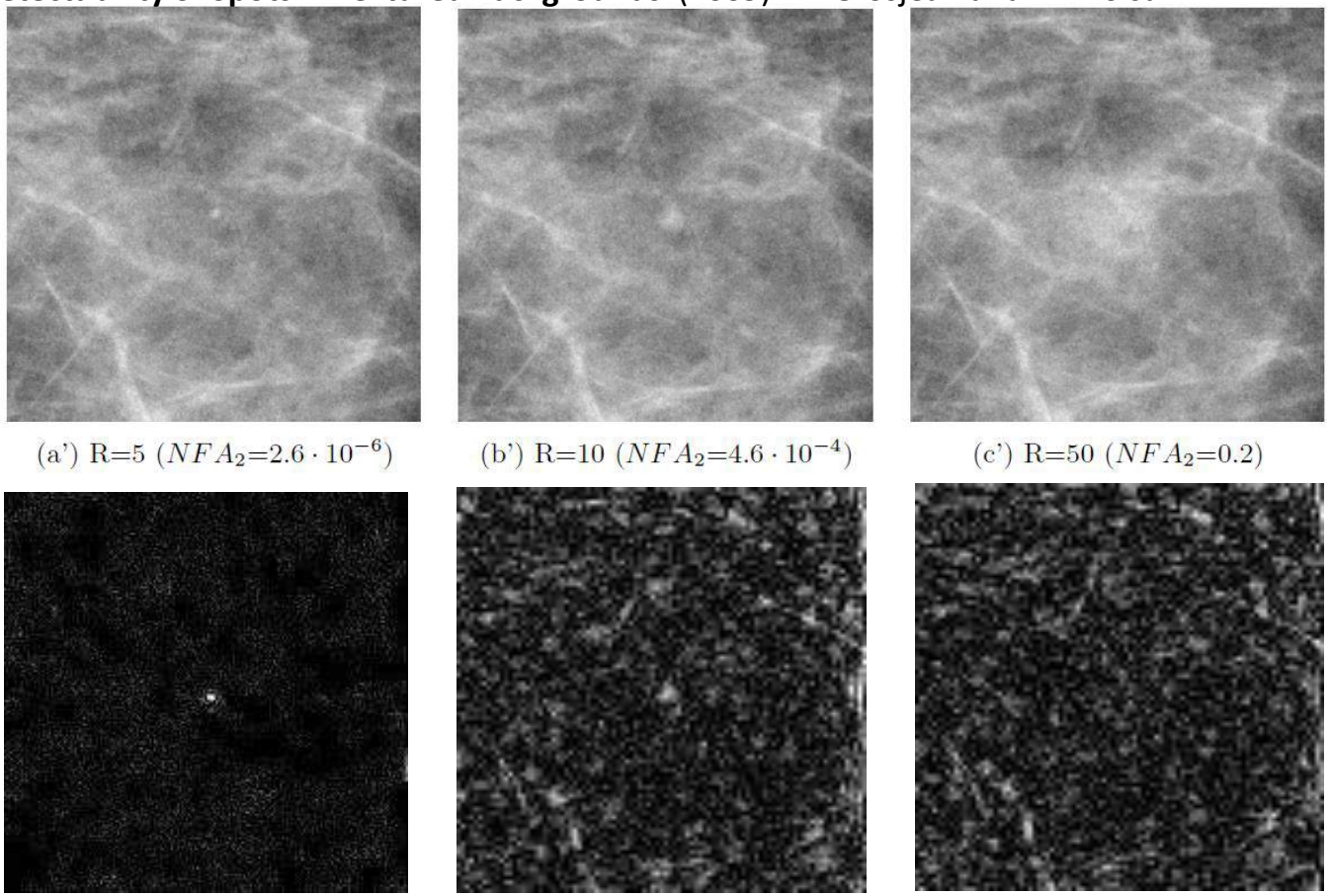






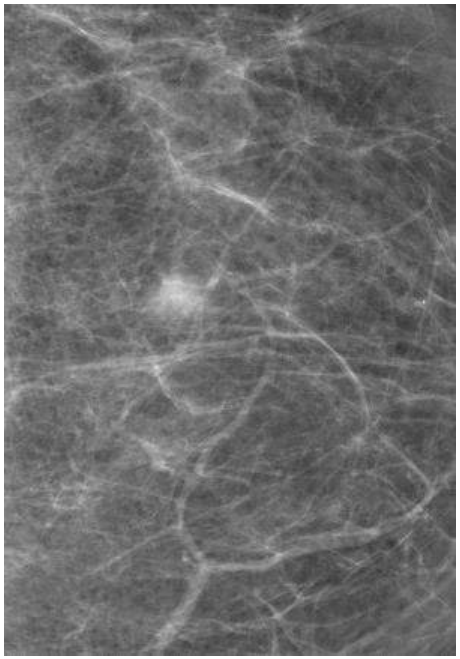
log NFA = -45.3,

**A-contrario Detectability of Spots in Textured Backgrounds** (**2009**) B. Grosjean and L. Moisan

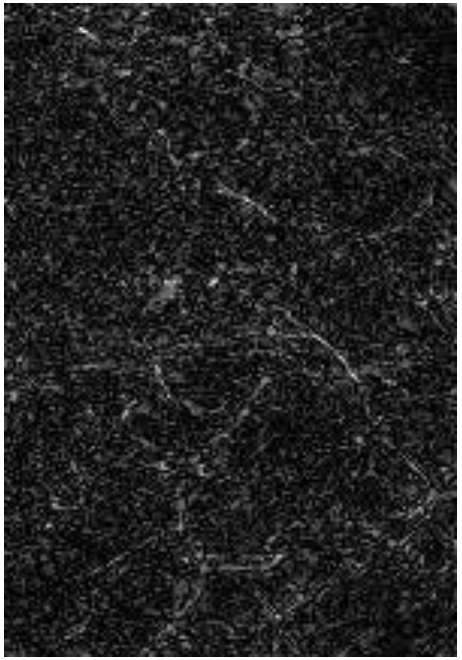



log NFA = -63. log NFA = -18. no detection

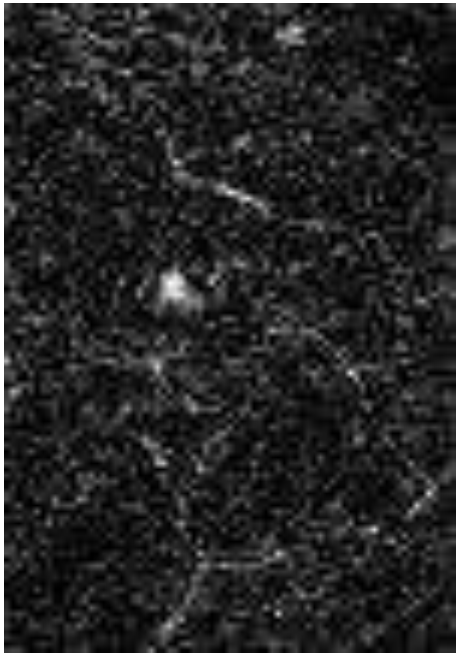
**A-contrario Detectability of Spots in Textured Backgrounds** (**2009**) B. Grosjean and L. Moisan



Minimum logNFA = -27.4337
Bayesian



Minimum logNFA = -27.05



Minimum logNFA = -39.2857

original noiseless

noisy ($\sigma = 50$, 15.0 dB)

NL-Bayes (**28.11** dB)

DCT8 1step (27.32 dB)

DCT8 2step (27.63 dB)

MS DCT8 2step (**27.99** dB)

DCT8 1step (27.32 dB)

DCT8 2step (27.63 dB)

MS DCT8 2step (**27.99** dB)

# NL-Bayes : the only diference w.r. to global denoising is that a Gaussian model is locally estimated in the image patch space

- patch noise model $\mathbb{P}(\tilde{P}|P) = c \cdot e^{-\frac{\|\tilde{P}-P\|^2}{2\sigma^2}}$
- Bayes' rule $\mathbb{P}(P|\tilde{P}) = \frac{\mathbb{P}(\tilde{P}|P)\mathbb{P}(P)}{\mathbb{P}(\tilde{P})}$
- assume we got a patch Gaussian model $\mathbb{P}(Q) = c.e^{-\frac{(Q-\overline{P})^t \mathbf{C}_P^{-1}(Q-\overline{P})}{2}}$
- hence the variational problem

$$\begin{aligned}
\max_P \mathbb{P}(P|\tilde{P}) &\Leftrightarrow \max_P \mathbb{P}(\tilde{P}|P)\mathbb{P}(P) \\
&\Leftrightarrow \max_P e^{-\frac{\|P-\tilde{P}\|^2}{2\sigma^2}} e^{-\frac{(P-\overline{P})^t \mathbf{C}_P^{-1}(P-\overline{P})}{2}} \\
&\Leftrightarrow \min_P \frac{\|P-\tilde{P}\|^2}{\sigma^2} + (P-\overline{P})^t \mathbf{C}_P^{-1}(P-\overline{P}).
\end{aligned}$$

- An empirical covariance matrix $C_{\tilde{P}}$ can be obtained for the patches $\tilde{Q}$ similar to $\tilde{P}$. $P$ and the noise $n$ being independent, $\mathbf{C}_{\tilde{P}} = \mathbf{C}_P + \sigma^2\mathbf{I}$; $E\tilde{Q} = \overline{P}$

A. Buades, M. Lebrun, J.M.M. : A Non-local Bayesian image denoising algorithm, SIIMS 2013

*BLS-GSM*: J. Portilla, V. Strela, M.J. Wainwright, E.P. Simoncelli, TIP 2003

# Bayesian denoising : NL-Bayes

$\max_P \mathbb{P}(P|\tilde{P}) \Leftrightarrow \min_P \frac{\|P-\tilde{P}\|^2}{\sigma^2} + (P-\overline{\tilde{P}})^t(\mathbf{C}_{\tilde{P}} - \sigma^2\mathbf{I})^{-1}(P-\overline{\tilde{P}})$

one step estimation $\hat{P}_1 = \overline{\tilde{P}} + \left[\mathbf{C}_{\tilde{P}} - \sigma^2\mathbf{I}\right]\mathbf{C}_{\tilde{P}}^{-1}(\tilde{P}-\overline{\tilde{P}})$, where empirically:

$$\mathbf{C}_{\tilde{P}} \simeq \frac{1}{\#\mathcal{P}(\tilde{P})-1}\sum_{\tilde{Q}\in\mathcal{P}(\tilde{P})}\left(\tilde{Q}-\overline{\tilde{P}}\right)\left(\tilde{Q}-\overline{\tilde{P}}\right)^t, \quad \overline{\tilde{P}} \simeq \frac{1}{\#\mathcal{P}(\tilde{P})}\sum_{\tilde{Q}\in\mathcal{P}(\tilde{P})}\tilde{Q}.$$

Iteration ("oracle estimation"): $\hat{P}_2 = \overline{\tilde{P}}^1 + \mathbf{C}_{\hat{P}_1}\left[\mathbf{C}_{\hat{P}_1} + \sigma^2\mathbf{I}\right]^{-1}(\tilde{P}-\overline{\tilde{P}}^1)$

where

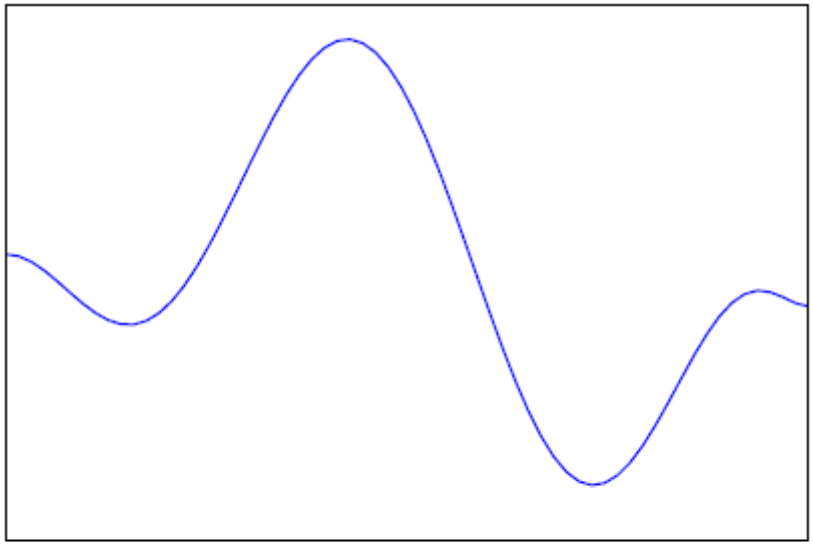
$$\mathbf{C}_{\hat{P}_1} \simeq \frac{1}{\#\mathcal{P}(\hat{P}_1)-1}\sum_{\hat{Q}_1\in\mathcal{P}(\hat{P}_1)}\left(\hat{Q}_1-\overline{\tilde{P}}^1\right)\left(\hat{Q}_1-\overline{\tilde{P}}^1\right)^t, \quad \overline{\tilde{P}}^1 \simeq \frac{1}{\#\mathcal{P}(\hat{P}_1)}\sum_{\hat{Q}_1\in\mathcal{P}(\hat{P}_1)}\tilde{Q}.$$

A. Buades, M. Lebrun, J.M.M.: A Non-local Bayesian image denoising algorithm, SIIMS 2013

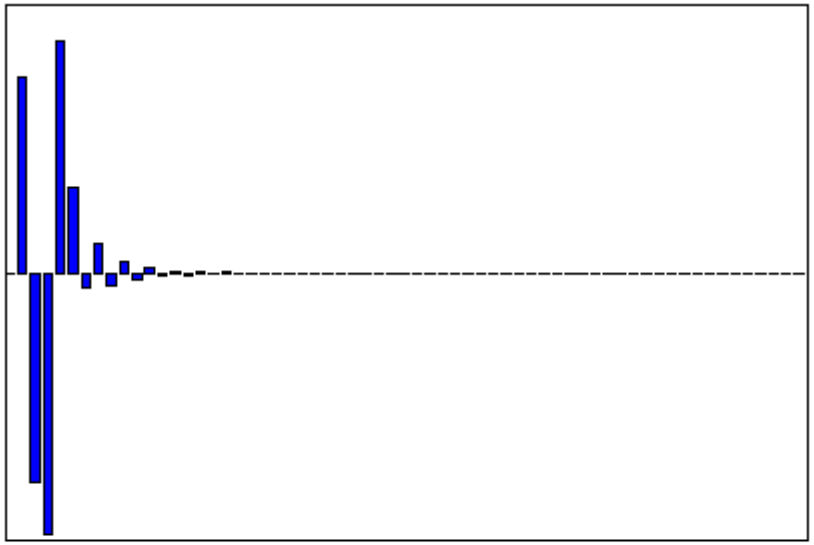
```
function x = DDID(y, sigma2)
    x = step(y, y, sigma2, 15, 7, 100, 4.0);
    x = step(x, y, sigma2, 15, 7, 8.7, 0.4);
    x = step(x, y, sigma2, 15, 7, 0.7, 0.8);
end

function xt = step(x, y, sigma2, r, sigma_s, gamma_r, gamma_f)

    [dx dy] = meshgrid(-r:r);
    h = exp(- (dx.^2 + dy.^2) / (2 * sigma_s^2));
    xp = padarray(x, [r r], 'symmetric');
    yp = padarray(y, [r r], 'symmetric');
    xt = zeros(size(x));

    parfor p = 1:numel(x), [i j] = ind2sub(size(x), p);

        % Spatial Domain: Bilateral Filter
        g  = xp(i:i+2*r, j:j+2*r);
        y  = yp(i:i+2*r, j:j+2*r);
        d  = g - g(1+r, 1+r);
        k  = exp(- d.^2 ./ (gamma_r * sigma2)) .* h;     % Eq. 4
        gt = sum(sum(g .* k)) / sum(k(:));               % Eq. 2
        st = sum(sum(y .* k)) / sum(k(:));               % Eq. 3

        % Fourier Domain: Wavelet Shrinkage
        V  = sigma2 .* sum(k(:).^2);                     % Eq. 5
        G  = fft2(ifftshift((g - gt) .* k));             % Eq. 6
        S  = fft2(ifftshift((y - st) .* k));             % Eq. 7
        K  = exp(- gamma_f * V ./ (G .* conj(G)));       % Eq. 9
        St = sum(sum(S .* K)) / numel(K);                % Eq. 8

        xt(p) = st + real(St);                           % Eq. 1
    end
end
```

**Algorithm 1**: *MATLAB code of Dual-Domain Image Denoising. This code reproduces all grayscale images in this paper.*

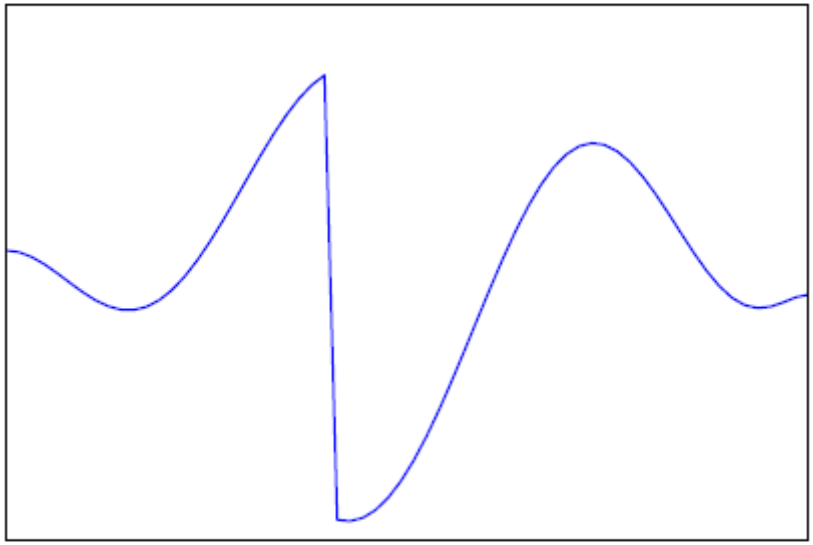
**Dual domain denoising Claude Knaus, Matthias Zwicker, ICIP 2013**



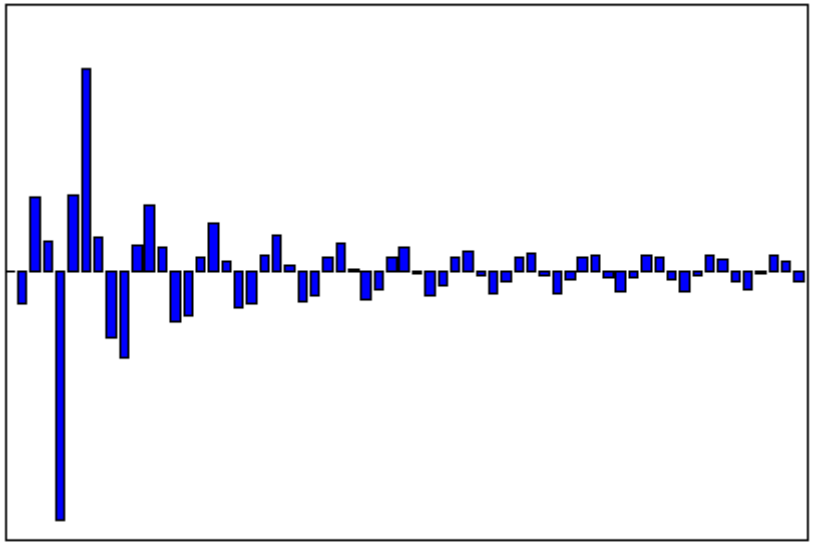
(a) Smooth signal



(b) DCT of smooth signal



(c) Signal with discontinuity (edge)



(d) DCT of signal with discontinuity

Figure 2.6 – Behaviour of DCT coefficients. Signals containing discontinuities have their energy less concentrated in the DCT domain. This makes the DCT basis less effective for denoising purposes in presence of edges.

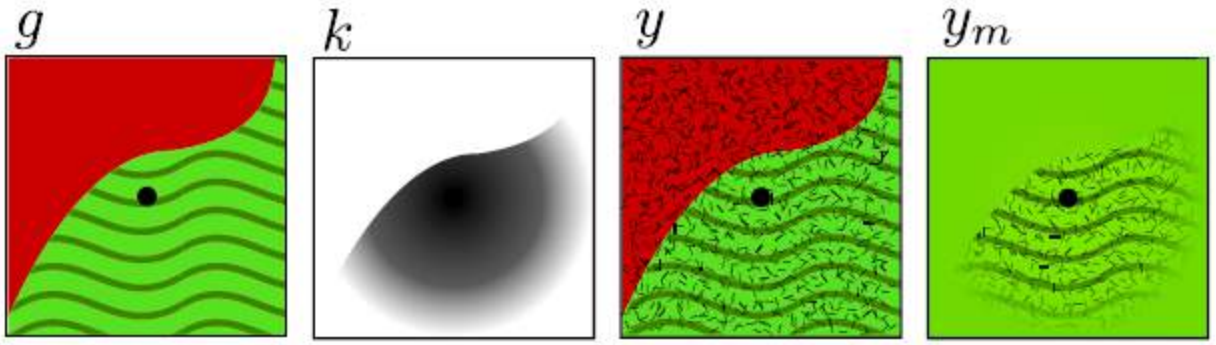
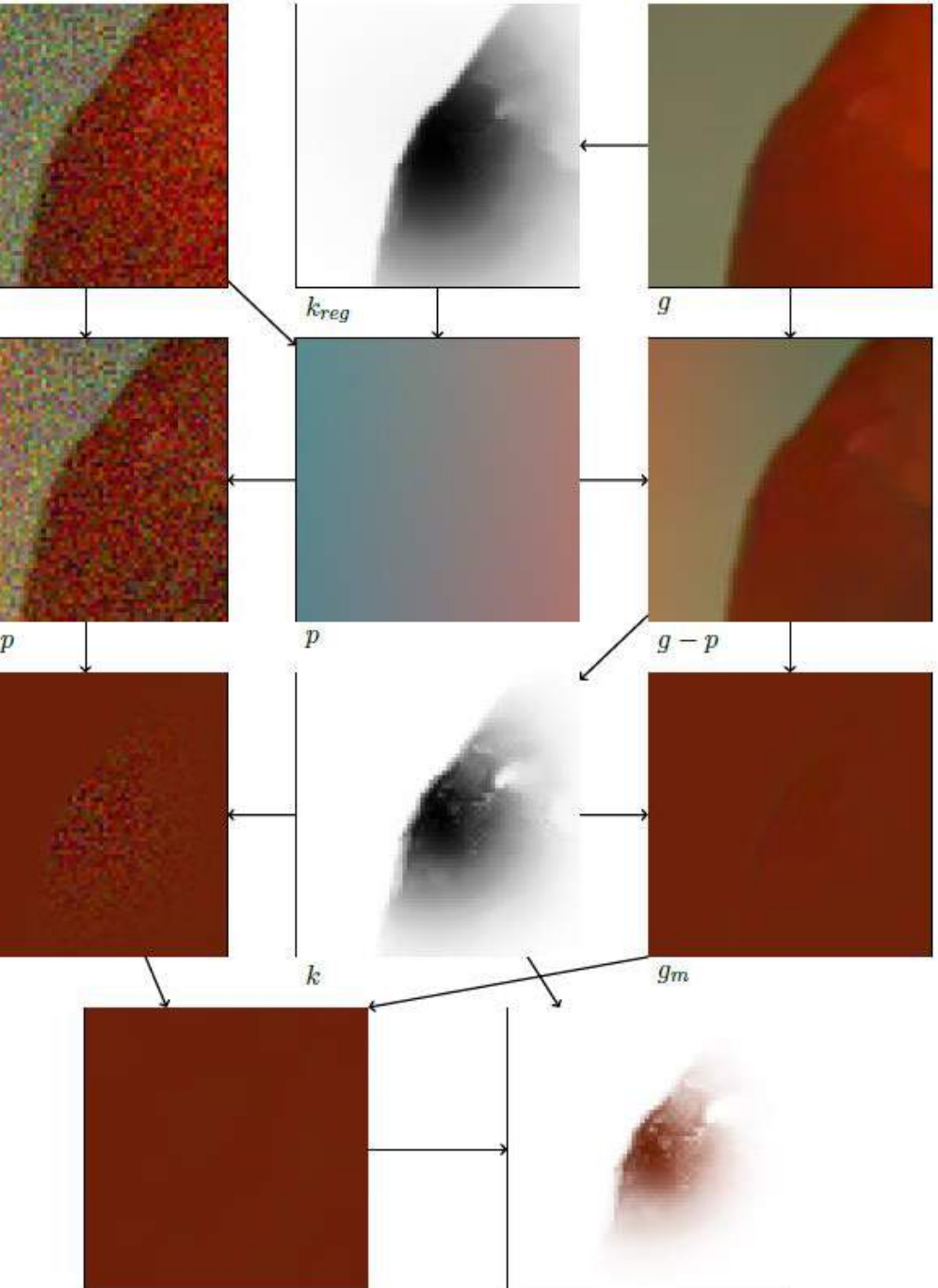



**Fig. 2.** Illustration of DDID's preprocessing of a patch. The kernel $k$ is computed using the guide $g$. In the modified patch $y_m$ all object discontinuities have been removed, leaving only the texture information corresponding to the object selected by the kernel $k$. The removed pixels are replaced by $\tilde{s}$: the average of the *meaningful* portion of the patch.

**This explanation of Dual denoising comes from:**

**Non-local dual image denoising**

N. Pierazzo, M. Lebrun, M. Rais, and G. Facciolo, ICIP 2014



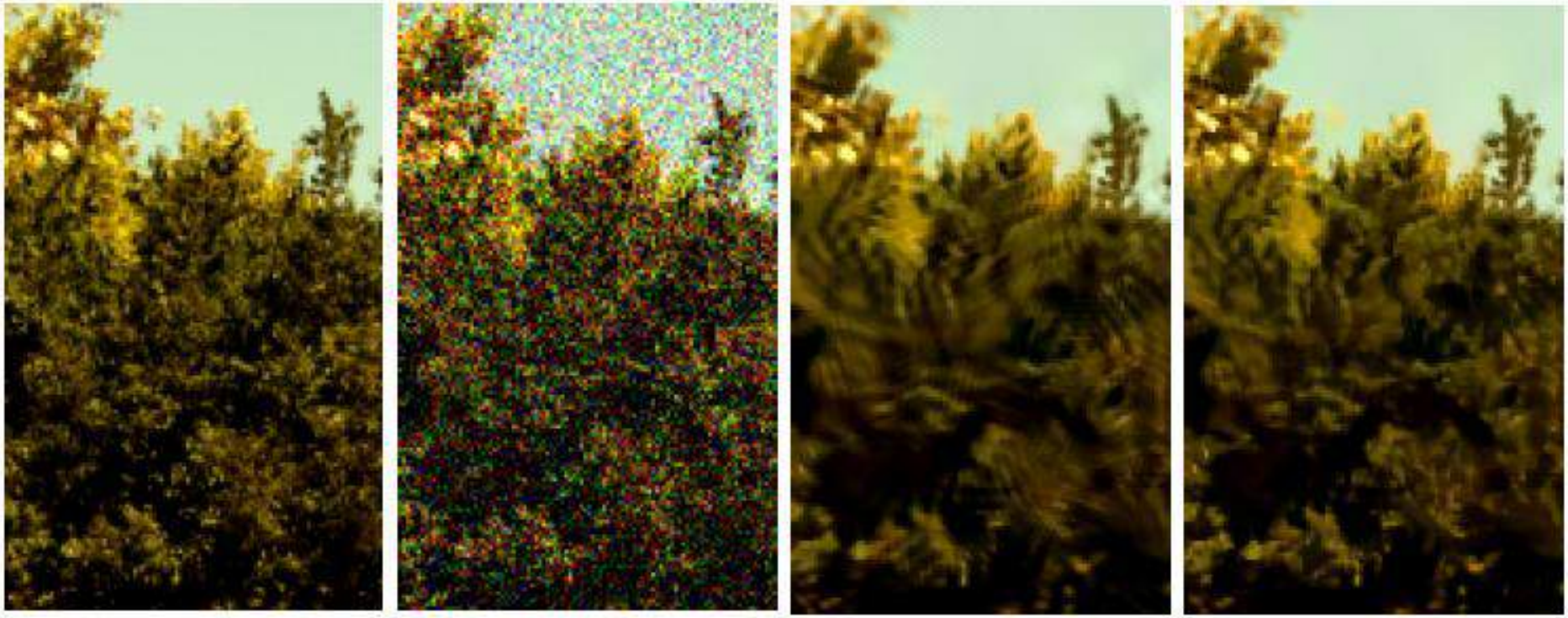

**Steps DA3D**
**(Data adaptive dual domain denoising)**

This figure shows what happens to a noisy patch taken in a natural image, containing an edge. The arrows indicate the elements needed to compute every step of the algorithm. Notice that, thanks to the weight function, the useful part of the patch is kept, while the discontinuities are completely removed.

Taken from

**N. Pierazzo, PhD thesis, 2016**

**Fig. 3**. Artifacts in DDID. From left to right: the noisy image (with $\sigma = 30$), the result of the first, second, and last iteration of the algorithm.



(a) Original (b) Noisy (c) DDID (d) NLDD

**Fig. 1**. A detail of the artifacts produced by DDID and the corresponding result of NLDD. In this example $\sigma = 30$.

**Pierazzo, N., Lebrun, M., Rais, M. E. & Facciolo, G. (2014, October). Non-local dual image denoising. ICIP 2014**

**N. Pierazzo, PhD thesis, 2016**

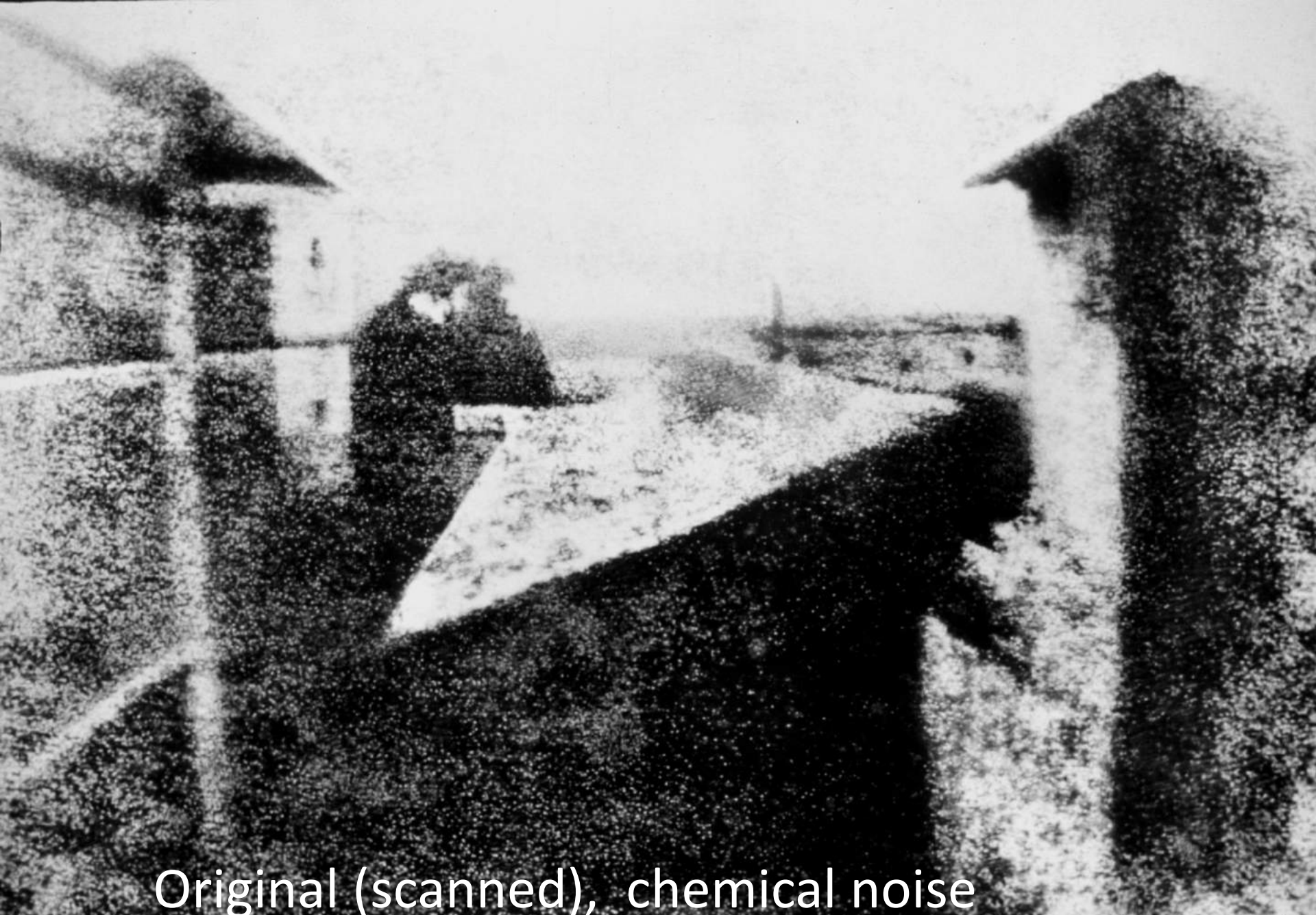
**On line demo: http://dev.ipol.im/~pierazzo/ipol_demo/ddmd/**

# 6-The noise clinic

The noise clinic at IPOL: estimating and denoising « any » image. This requires  to estimate the noise before denoising. For image that have been manipulated, noise can be :
-signal dependent
-frequency dependent
-scale dependent
Thus « noise curves » are established for each color level, each dyadic scale and each DCT frequency
Based on this a Bayesian algorithm can be applied (NL-Bayes)

**Where to test all algorithms:** **Image Processing on Line (IPOL)** **http://www.ipol.im/**

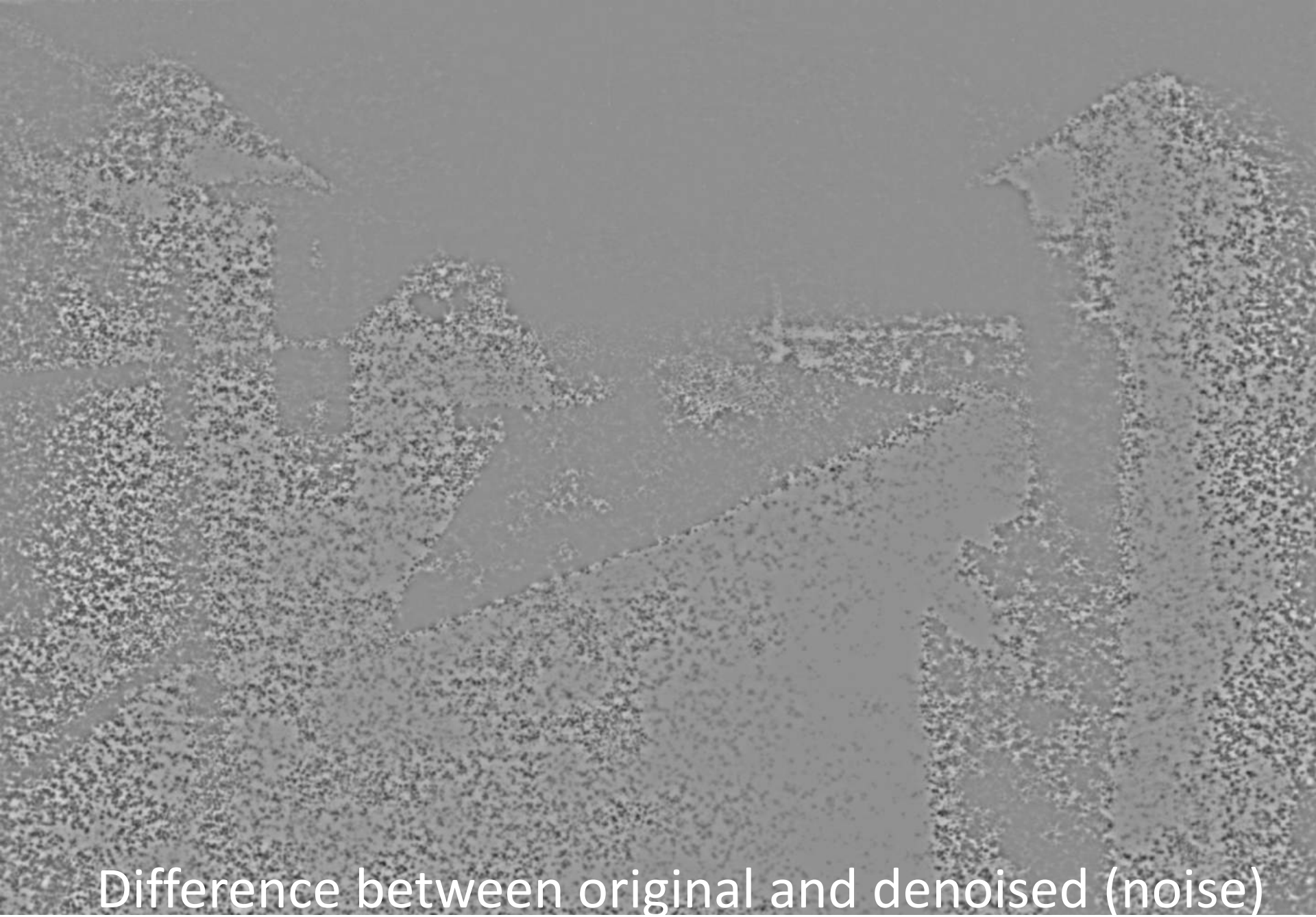
**Lebrun, Marc, Miguel Colom, and JMM. "The Noise Clinic: a blind image denoising algorithm." *Image Processing On Line* 5 (2015): 1-54.**




View from the Window at Le Gras (1826), Joseph Nicéphore Niépce




Denoised by the Noise Clinic, IPOL (Image Processing on Line www.ipol.im)



# Difference between original and denoised (noise)

View from the Window at Le Gras (1826), Joseph Nicéphore Niépce

**The oldest heliographic engraving known in the world**, a reproduction of a 17th century Flemish engraving. Nicéphore Niépce in 1825, ( Bibliothèque nationale de France).

**Joseph-Nicéphore Niépce (1765-1833): first indoor photograph,**
Denoised by the Noise Clinic, IPOL (Image Processing on Line www.ipol.im)

**Joseph-Nicéphore Niépce (1765-1833): first indoor photograph,**
Denoised by the Noise Clinic, IPOL (Image Processing on Line www.ipol.im)

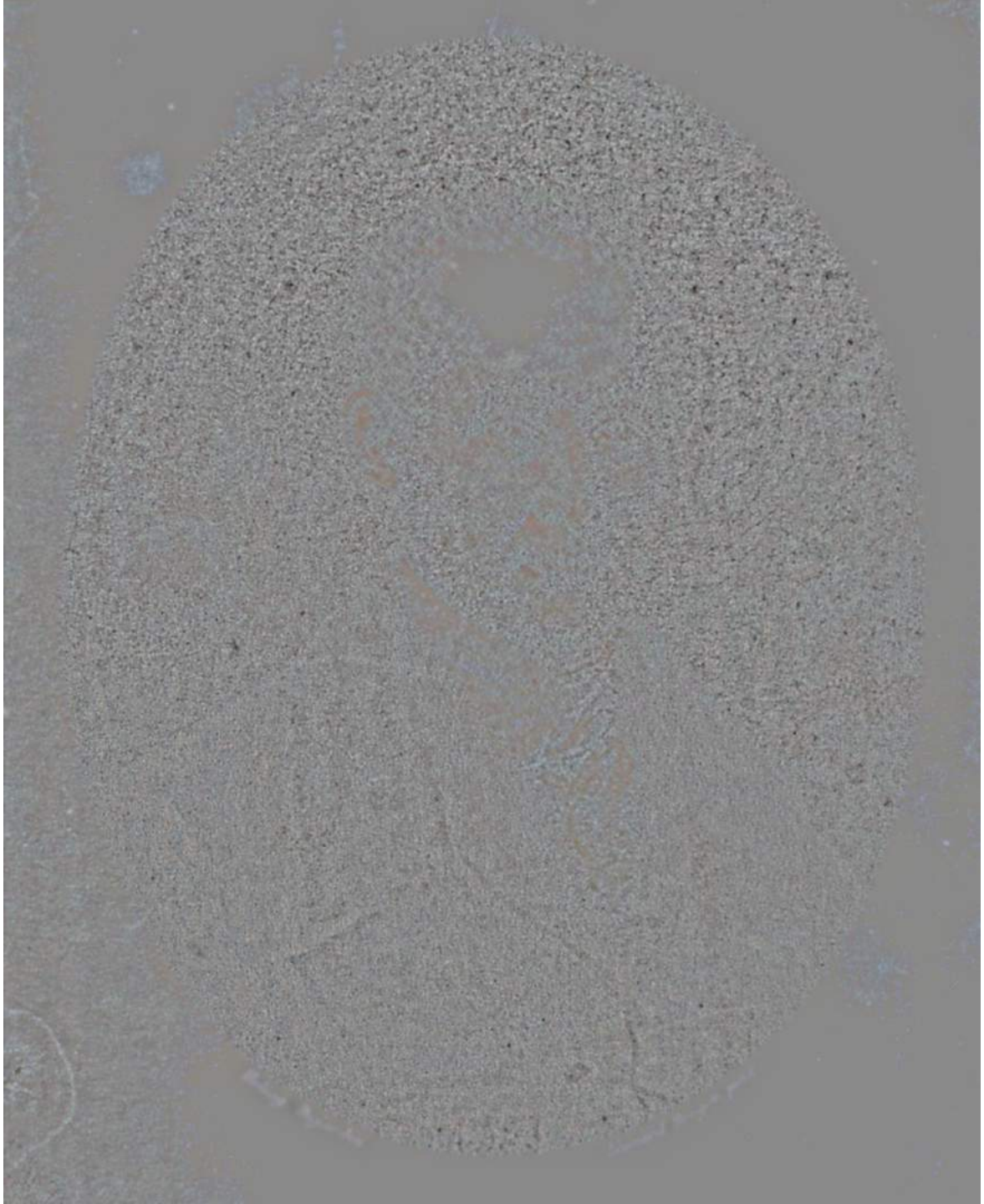
**Joseph-Nicéphore Niépce (1765-1833)**

*Photograph by Dujardin of a portrait of N. Niépce by L.F. Berger. Repr. by Günter Josef Radig, Wikipedia*

*We can as well denoise the scanned version of a 19th century photograph of this portrait…*

**Joseph-Nicéphore Niépce (1765-1833)**

Denoised by the Noise Clinic,
IPOL (Image Processing on Line www.ipol.im)



**Joseph-Nicéphore Niépce (1765-1833)**

Difference between portrait and its denoised version by the Noise Clinic, IPOL (Image Processing on Line www.ipol.im)

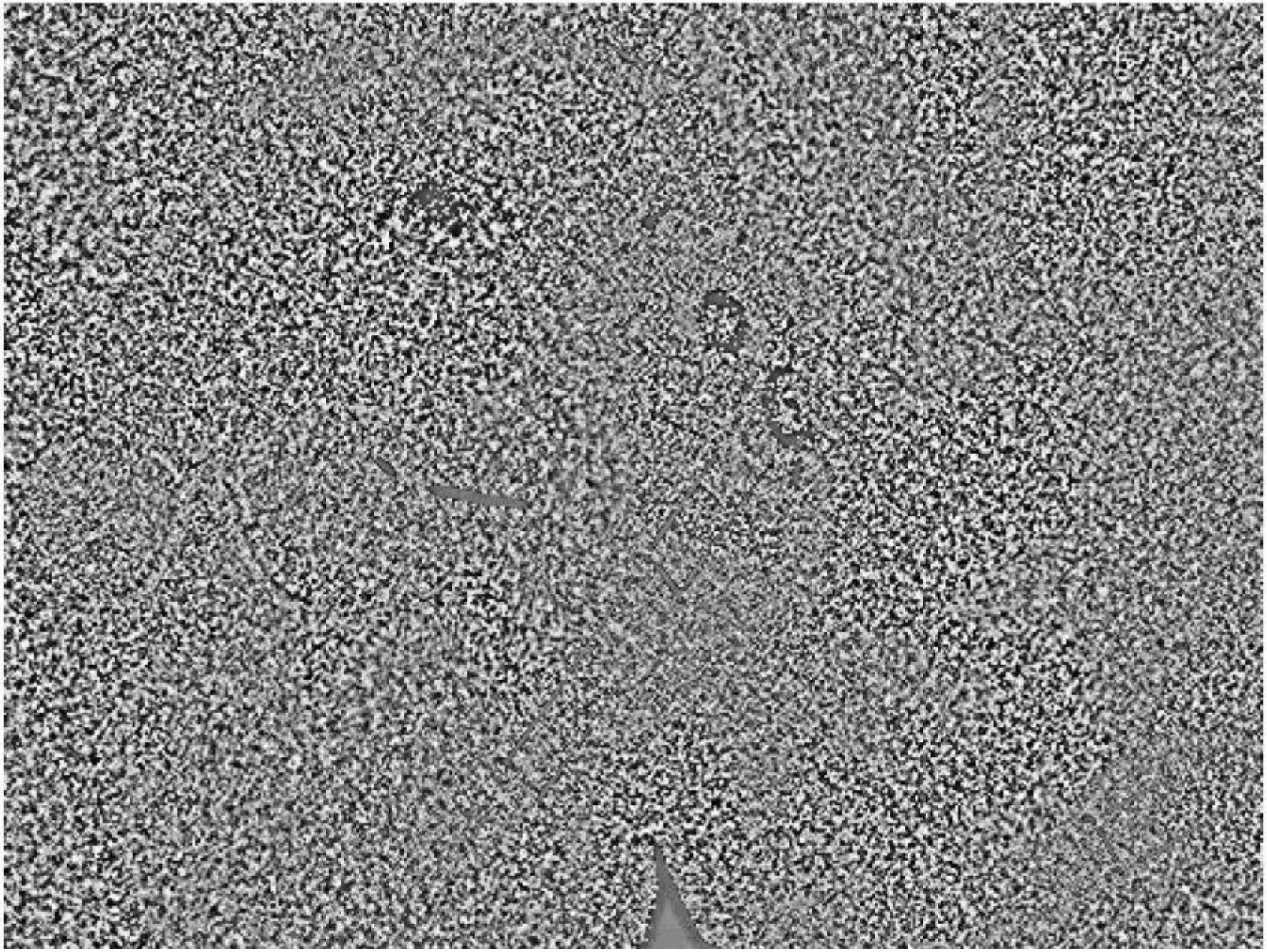
Making the difference between original and denoised permits to check if some detail has been removed at the same time as the noise. It is the case here.

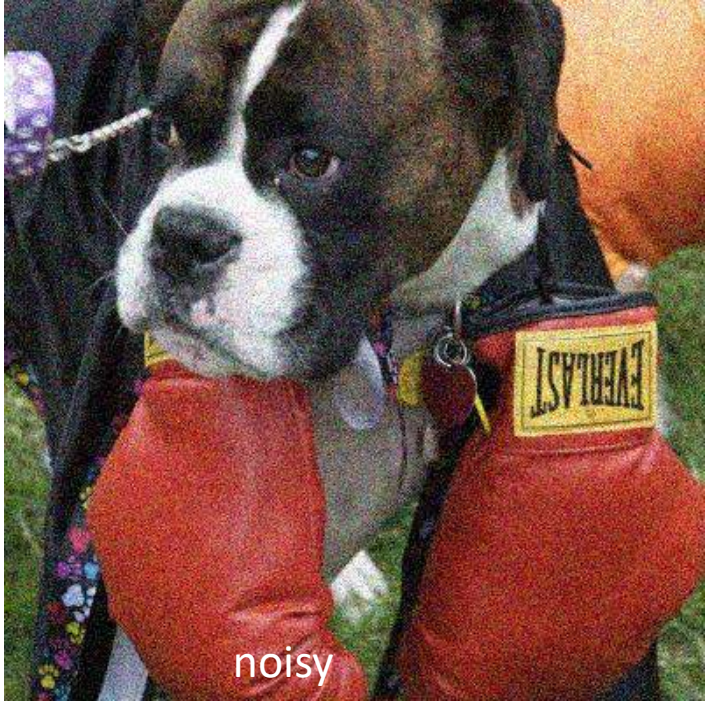
**Joseph-Nicéphore Niépce (1765-1833)**
He made in 1826 the first outdoor successful photograph — an image of his courtyard, seen from his house — by putting a pewter plate coated with bitumen (a light-sensitive material) in the back of a camera obscura, a black box with a pinhole.
He also made the first known indoor heliographic engraving in 1825.

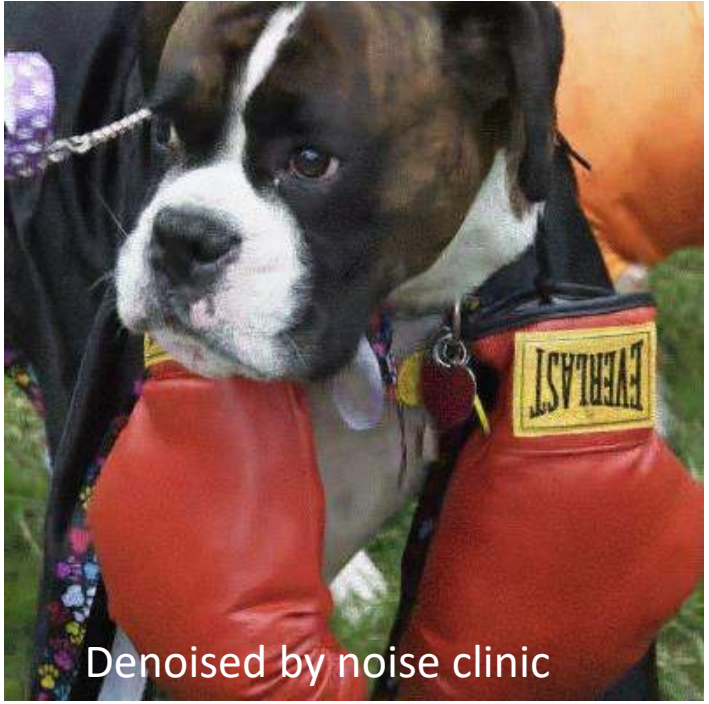
# The Noise Clinic

# The Noise Clinic

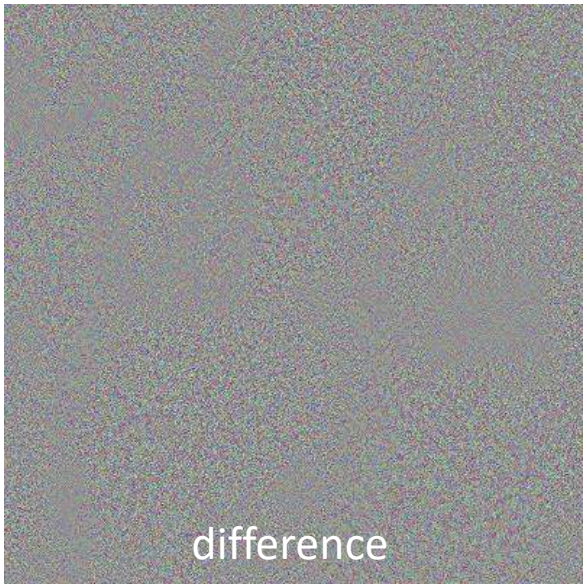
# The Noise Clinic






# Than you:

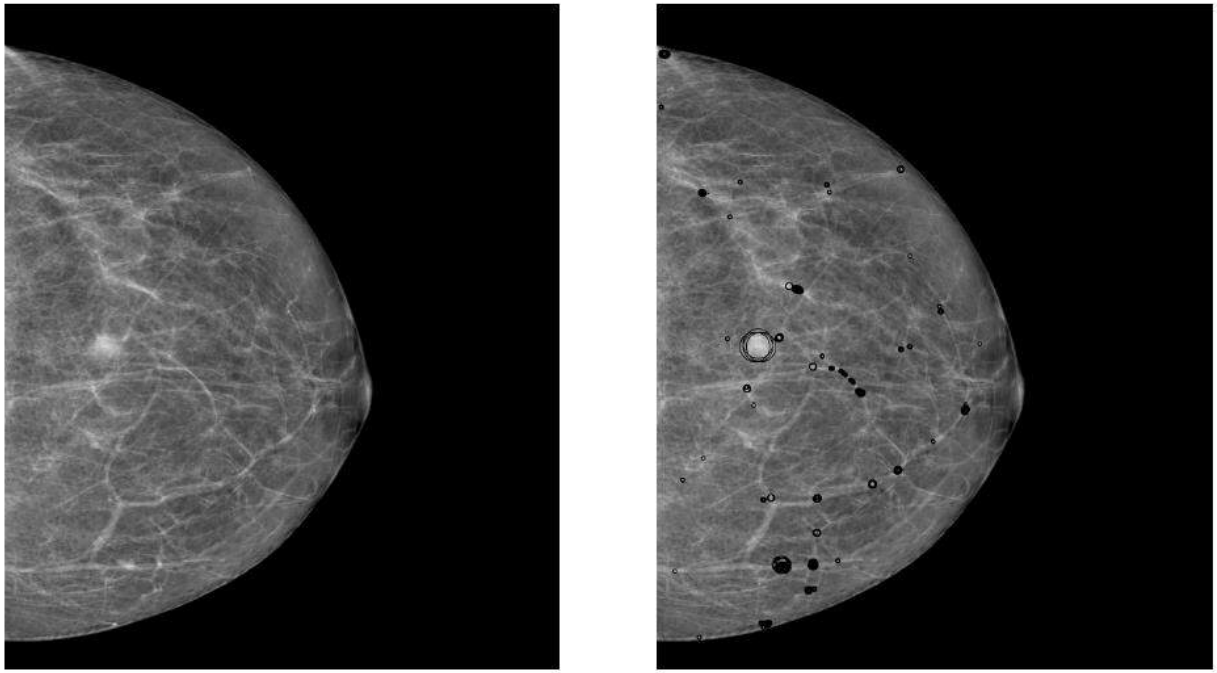



# questions?

**Where to test all algorithms:** Image Processing on Line (IPOL) http://www.ipol.im/

**A-contrario Detectability of Spots in Textured Backgrounds** (2009)
B. Grosjean and L. Moisan
***Tumor detection in mammography (data: General Electric)***

Problem: background too complex (and samples too sparse) for a simple stochastic model!



(a) Original image (b) $\varepsilon = 1$

"Figure 9: A-contrario detection of spots in a mammography image, for two values of the threshold " applied to the detection metric $NFA_2$. The large opacity is well detected (its $NFA_2$ is equal to 0.15). Some clinically wrong detections also occur (small spots), mainly because the curvilinear breast structures are not taken into account by the texture model (these are false alarms clinically speaking, but are not with respect to the naive model used for the breast texture)."

**A-contrario Detectability of Spots in Textured Backgrounds** (**2009**) B. Grosjean and L. Moisan

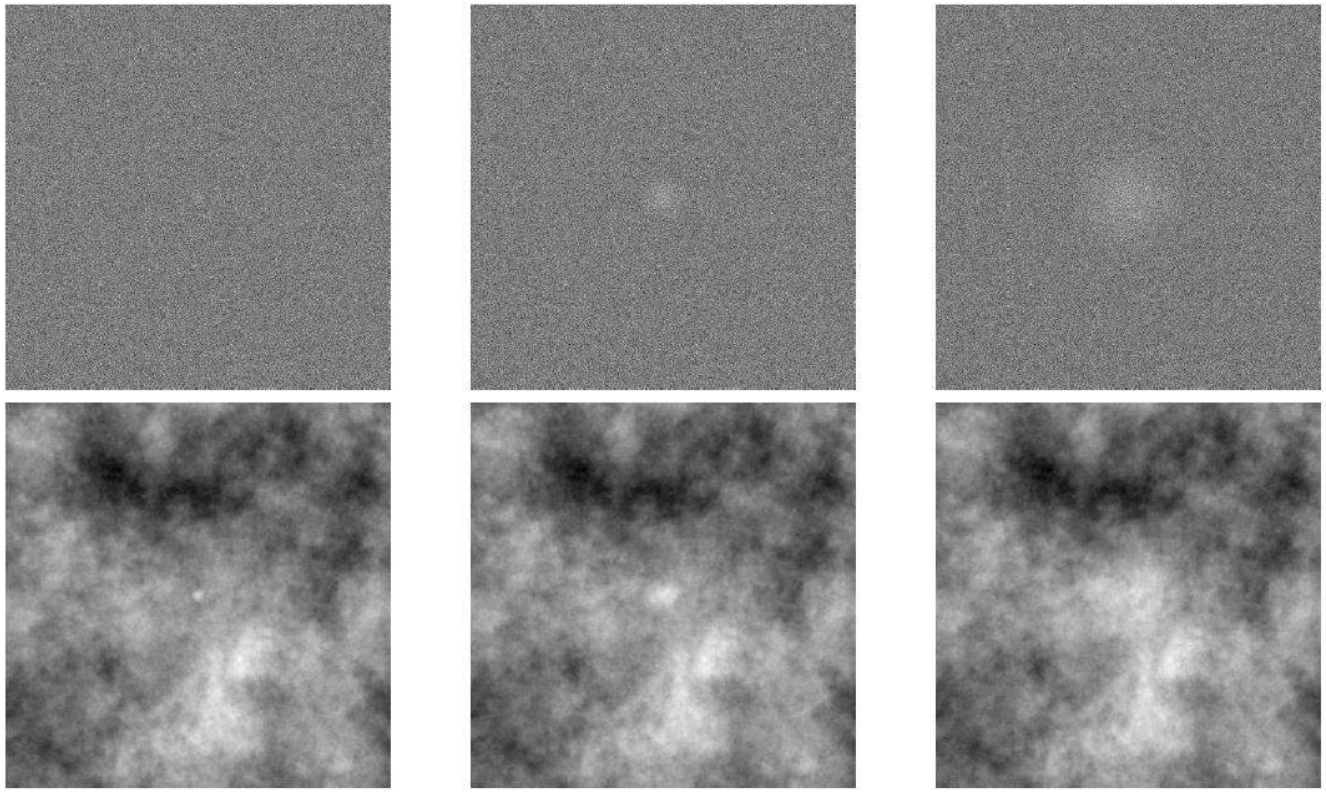
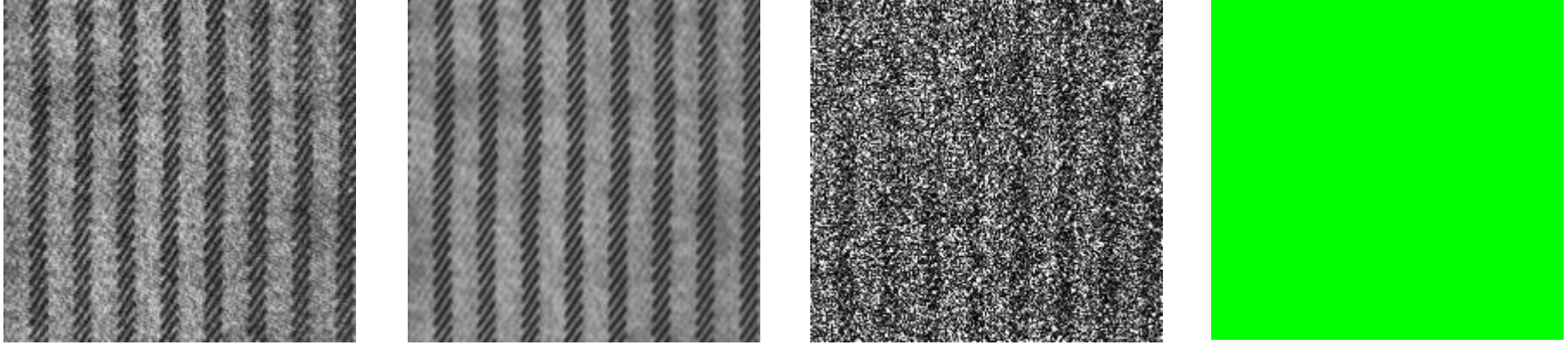



Figure 2: Examples of simulated spots with various sizes (R = 10, 30 and 80) but similar contrast, in a white noise texture (top row) and in a colored noise texture (bottom row). In the white noise texture, the saliency of the spot increases with its size. On the contrary, in the colored noise background, the unexpected reverse phenomenon occurs: the larger the spot, the less visible it is.

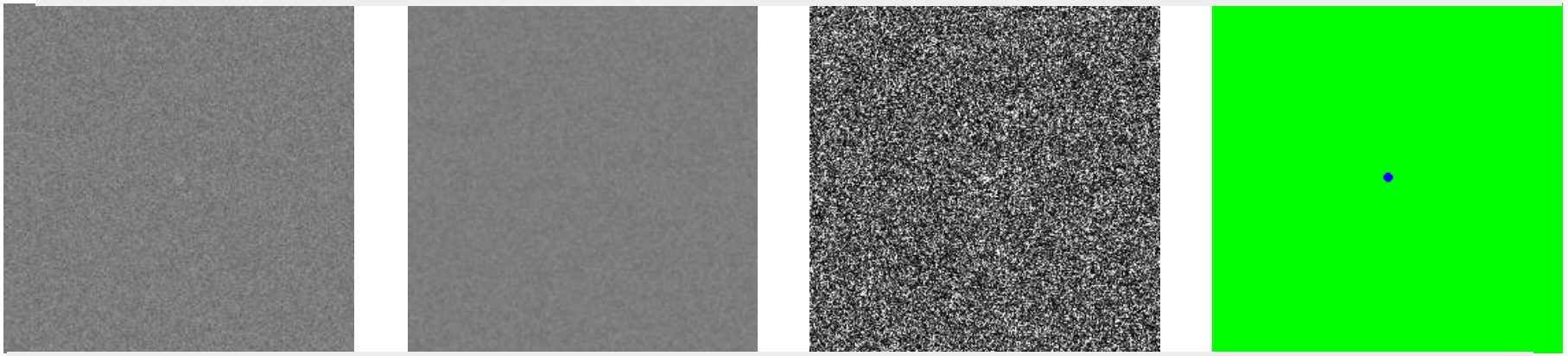
## Type 2 Anomaly detection

- Take the difference image $N = \tilde{u} - u$ where $\tilde{u}$ is the estimated image model. $N$ should be white noise.
- Compute the standard deviation $\sigma$ of $N$
- Detect all **exceptional pixels** $x$, such that $\mathbb{P}(N(x) > s\sigma)$, (s=4)
- For each square window $W$ with size $n$ (e.g. $n = 16^2$); count the number $k$ of exceptional pixels in $W$
- Compute the **Number of false alarms** of the **exceptional square window**, $NFA(k, s) := n' \begin{pmatrix} k \\ n \end{pmatrix} \mathbb{P}(N(x) > s\sigma)^k$. ($n'$ is the number of tested regions)

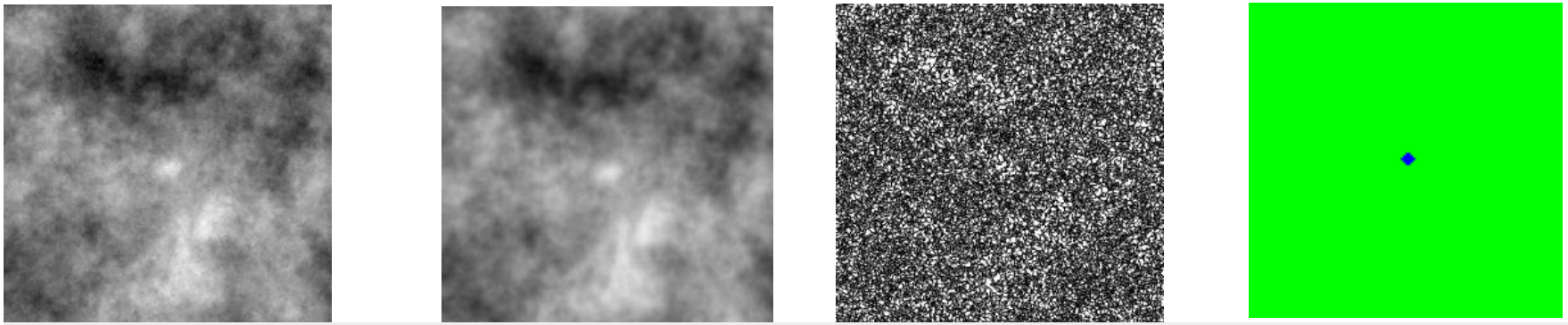
Desolneux, Agnes, Lionel Moisan, and JMM. *From gestalt theory to image analysis: a probabilistic approach*. Vol. 34. Springer Science & Business Media, 2007.



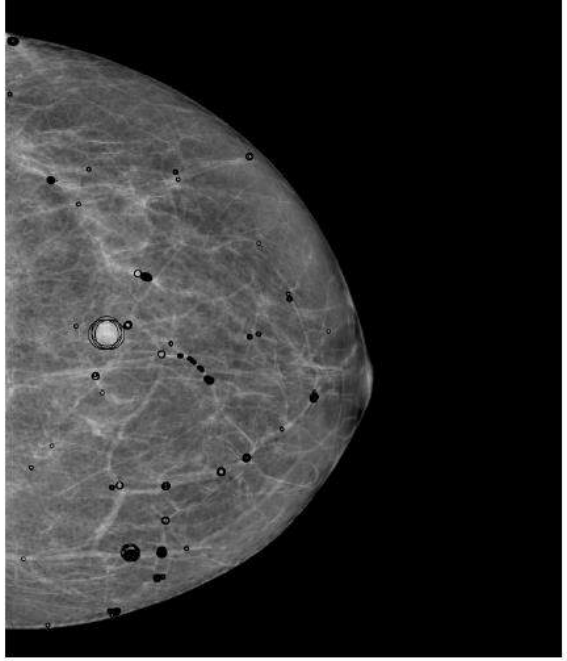
Original denoised noise no detection



Original denoised noise detection: log NFA = -10.7



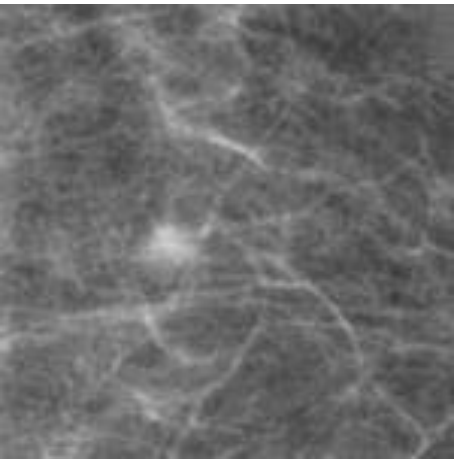
Original denoised noise detection: log NFA = -45.3

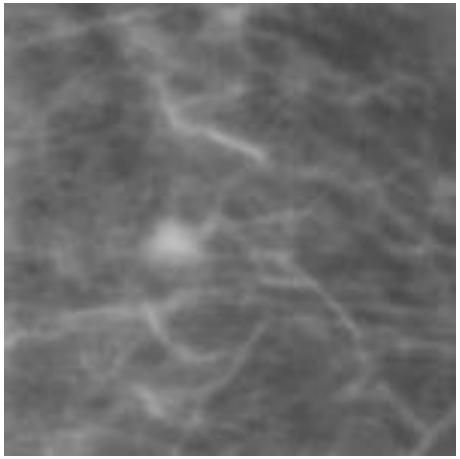


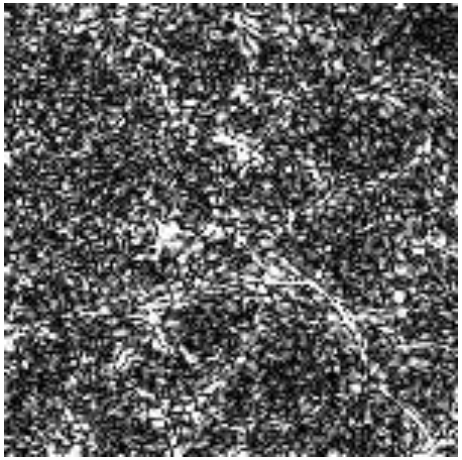
**A-contrario Detectability of Spots in Textured Backgrounds (2009)**
B. Grosjean and L. Moisan

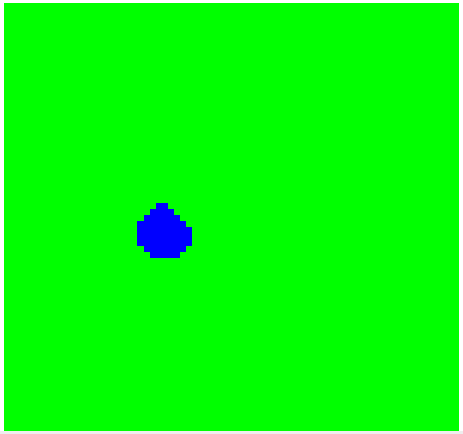
(a) Original image

NFA = 0.15 for the tumor, many false detections









log NFA = -39.2857
By the very same method applied to the noise