Published in Image Processing On Line on 2015–01–28.
Submitted on 2014–09–17, accepted on 2014–11–25.
ISSN 2105–1232 © 2015 IPOL & the authors CC–BY–NC–SA
This article is available online with supplementary materials, software, datasets and online demo at
https://doi.org/10.5201/ipol.2015.125

The Noise Clinic: a Blind Image Denoising Algorithm

Marc Lebrun, Miguel Colom, Jean-Michel Morel

CMLA, ENS Cachan, France ({marc.lebrun, colom, morel}@cmla.ens-cachan.fr)

Communicated by Jacques Froment *Demo edited by* Miguel Colom

Abstract

This paper describes the complete implementation of a blind image denoising algorithm, that takes any digital image as input. In a first step the algorithm estimates a Signal and Frequency Dependent (SFD) noise model. In a second step, the image is denoised by a multiscale adaptation of the Non-local Bayes denoising method. We focus here on a careful analysis of the denoising step and present a detailed discussion of the influence of its parameters. Extensive commented tests of the blind denoising algorithm are presented, on real JPEG images and on scans of old photographs.

Source Code

The source code (ANSI C), its documentation, and the online demo are accessible at the IPOL web page of this article[1].

Keywords: denoising; noise estimation; signal dependency; frequency dependency; multiscale approach; patch based; blind denoising

1 Introduction

Blind denoising is the conjunction of a noise estimation method and of a denoising method. To be useful to all image users, who generally have only access to the end result of a complex processing chain, blind denoising must be able to cope with both raw and preprocessed images of all sorts. Yet most denoising algorithms published in the literature and even their IPOL implementations only deal with white noise. This is the case for DCT denoising [27], TV denoising [9], K-SVD [15], NL-means [3], BM3D [10] and NL-Bayes [12]. Blind denoising has been considered by Portilla [21], [20], Rabie [22] and Liu, Freeman, Szeliski and Kang [16]. Portilla's method is an adaptation of the BLS-GSM algorithm modeling wavelet patches by a Gaussian scale mixture (GSM), combined with a Bayesian least square (BLS) estimation.

Liu, Freeman, Szeliski and Kang [16] have also proposed a denoising framework for JPEG images which performs an automatic estimation of a "noise level function" which is an upper estimate of

[1]https://doi.org/10.5201/ipol.2015.125

the noise variance, followed by a removal of colored noise. Contrarily to Portilla's method, no code is available for this complex method.

The method proposed by Rabie [22] works only for Gaussian uniform colored noise, which is estimated on flat image regions. Contrarily to the Noise Clinic, this method only deals with a signal-independent Gaussian noise. An entropy-based noise level estimator is proposed in [8], giving a "noise level" but not a complete noise model. For the Noise Clinic we shall rely on a method proposed by Colom et al. [5]. This method is an adaptation of the Ponomarenko et al. method [19] which builds a frequency dependent noise model.

JPEG images constitute the overwhelming majority of digital images for which the image formation history has been lost. Thus we pay a special attention to them and to the particular noise structure left by JPEG compression. Nevertheless, the proposed method is adapted to any signal dependent colored noise. For example a JPEG image obtained by scanning an old photograph is a valid patient for the Noise Clinic.

The Noise Clinic was introduced in [14] and its theory and a sketch description is given in [13]. Here, our goal is to present a thorough description of its implementation to make it fully reproducible. For several steps, we nevertheless rely on the IPOL description of the noise estimation algorithm [6] and on the IPOL paper [12].

Plan of the paper. Section 2 describes the adaptation of the NL-Bayes denoising method to a signal and frequency dependent noise model. Section 3 describes the noise model and gives the algorithm computing the signal dependent noise patch covariance matrix. Section 4 is dedicated to the multiscale denoising procedure. Section 4.2 gives a thorough parameter analysis. Section 5 describes experiments on real noisy images with unknown history. The Noise Clinic is compared with Portilla's blind denoising [21] and with the results of the commercial software *Neat Image.*

2 A Generalized Nonlocal Bayesian Algorithm

In order to help the reader with the huge amount of notations introduced in this section, a brief description is presented in tables 1 and 2, including notations and parameters.

NL-Bayes has been originally developed to deal with Gaussian signal-independent noise. Thus, small modifications of the original NL-Bayes are required to adapt it to signal-dependent colored noise. This also leads to several improvements of this algorithm discussed at the end of this section in order to better deal with noise estimations. Let us start with the classic set up where the noise is white Gaussian. This assumption is not so restrictive as it can be extended to raw image thanks to the Anscombe transform [1] transforming Poisson noise into approximately Gaussian white noise.

The Gaussian white noise assumption permits to compare method results easily by their PSNR or RMSE on noise-free images u to which a known white Gaussian noise n of standard deviation σ has been added. We shall denote the noisy image $\tilde{u}$. Thus, in a simple experimental setup, denoising algorithms are applied on $\tilde{u}$ thus constructed, knowing σ. Once the denoised image $\hat{u}$ is obtained, PSNR and RMSE are directly obtained from $\hat{u}$ and u.

Our selected method for blind denoising is NL-Bayes. This algorithm denoises all noisy square patches extracted from the noisy image $\tilde{u}$. The final denoised image $\hat{u}$ is obtained by the classic "aggregation", namely replacing each pixel value by an average of the denoised values obtained for all patches containing this pixel. $\tilde{P}$ denotes a reference patch extracted from the image, and $\mathcal{P}(\tilde{P})$ a set of patches $\tilde{Q}$ similar to $\tilde{P}$. An easy Bayesian argument shows that if $\tilde{Q}$ follows a Gaussian model, a first estimation of the denoised patch P, which we shall call "basic estimation", is described in

u	original noise-free image
n	noise
σ	standard deviation of the noise
$\tilde{u}$	noisy input image
$\hat{u}$	denoised output image
P	original noise-free reference patch
$\tilde{P}$	reference noisy patch
$\mathcal{P}(\tilde{P})$	3D group composed of similar patches to $\tilde{P}$
$\tilde{Q}$	generic noisy patch
$P^{\mathbf{basic}}$	basic estimation of $\tilde{P}$ obtained after the first step of NL-Bayes
$P^{\mathbf{final}}$	final estimation of $\tilde{P}$ obtained after the second step of NL-Bayes
$\overline{\tilde{P}}$	barycenter of $\mathcal{P}(\tilde{P})$, i.e. average of all similar patches to $\tilde{P}$
$\overline{\tilde{P}}^{\mathbf{basic}}$	barycenter of similar patches to $\tilde{P}^{\mathbf{basic}}$
$\mathbf{C}_{\tilde{P}}$	empirical covariance matrix of the patches similar to $\tilde{P}$
$\mathbf{C}_{\tilde{P}}^{\mathbf{basic}}$	empirical covariance matrix of the patches similar to $\tilde{P}^{\mathbf{basic}}$
$\mathbf{C}_N$	covariance matrix of the noise
c	the index of the current channel
N_c	number of channels of the image
$\mathbf{i}$	intensity
$\mathbf{I}$	identity matrix

Table 1: Notations introduced in Section 2.

$\kappa \times \kappa$	size of patches
α_c	weight for each channel c during the basic estimation
N_{min}	minimal number of similar patches at least contained in 3D groups
N_{max}	maximal number of similar patches that a 3D group is allowed to contain
w	width of the image
h	height of the image
$\eta \times \eta$	size of the search window
τ	similarity threshold for similar patches
γ	thresholding parameter to determine if a 3D group belongs or not to an homogeneous area
k_L	low threshold for homogeneous area criteria
k_H	high threshold for homogeneous area criteria

Table 2: Parameter notations introduced in Section 2.

Algorithm 1 and can be obtained [11] by

$$P^{\mathbf{basic}} = \overline{\tilde{P}} + \left[\mathbf{C}_{\tilde{P}} - \mathbf{C}_n\right] \mathbf{C}_{\tilde{P}}^{-1} \left(\tilde{P} - \overline{\tilde{P}}\right) \tag{1}$$

where

- $\overline{\tilde{P}}$ is the empirical average of the patches similar to $\tilde{P}$

$$\overline{\tilde{P}} \simeq \frac{1}{\#\mathcal{P}(\tilde{P})} \sum_{\tilde{Q} \in \mathcal{P}(\tilde{P})} \tilde{Q}, \tag{2}$$

- $\mathbf{C}_n$ is the covariance matrix of the noise,
- $\mathbf{C}_{\tilde{P}}$ is the empirical covariance matrix of the patches similar to $\tilde{P}$, which may be obtained by

$$\mathbf{C}_{\tilde{P}} \simeq \frac{1}{\#\mathcal{P}(\tilde{P}) - 1} \sum_{\tilde{Q} \in \mathcal{P}(\tilde{P})} \left(\tilde{Q} - \overline{\tilde{P}}\right) \left(\tilde{Q} - \overline{\tilde{P}}\right)^t. \tag{3}$$

Algorithm 1 Description of the basic estimation of similar patches.

```
**Input :** noisy 3D groups $\left(\mathcal{P}_c(\tilde{P})\right)_c$, one for each channel,
**Input :** For each intensity $\mathbf{i}$, the corresponding covariance matrices of the noise $\left(\mathbf{C}_{n_{\mathbf{i},c}}\right)_c$, one for each channel.
**Output :** Basic 3D groups $\left(\mathcal{P}_c(P^{\mathbf{basic}})\right)_c$ for each channel.
for each $c = 1$ to $N_c$ do
  Compute the average intensity:
    $$\mathbf{i} = \frac{1}{\#\mathcal{P}_c(\tilde{P})\kappa^2} \sum_{\tilde{Q}_c \in \mathcal{P}_c(\tilde{P})} \sum_{p=1}^{\kappa} \sum_{q=1}^{\kappa} \tilde{Q}_c(p,q)$$
  Get the corresponding noise covariance matrix: $\mathbf{C}_{n_{\mathbf{i},c}}$
  Define a confidence interval for $\mathcal{P}_c(\tilde{P})$:
  $m_c = \min\left(\mathcal{P}_c(\tilde{P})\right) - \frac{\mathrm{Tr}(\mathbf{C}_{n_{\mathbf{i},c}})}{\#\mathcal{P}_c(\tilde{P})}$
  $M_c = \max(\mathcal{P}_c(\tilde{P})) + \frac{\mathrm{Tr}(\mathbf{C}_{n_{\mathbf{i},c}})}{\#\mathcal{P}_c(\tilde{P})}$
  Compute the barycenter $\overline{\tilde{P}}$                                   ▷ Equation (2)
  Compute the covariance matrix of the noisy patches $\mathbf{C}_{\tilde{P}}$     ▷ Equation (3)
  Correct $\mathbf{C}_{\tilde{P}}$                                                ▷ Algorithm 6
  for each $\tilde{Q}_c \in \mathcal{P}_c(\tilde{P})$ do
    Compute the estimation $Q_c^{\mathbf{basic}}$                                 ▷ Equation (1)
  end for
  Force the estimated values into reasonable bounds:
  for $\tilde{Q}_c \in \mathcal{P}_c(\tilde{P})$ do
    for $(p,q) \in [\![1,\kappa]\!]^2$ do
      $Q_c^{\mathbf{basic}}(p,q) = \min(\max(Q_c^{\mathbf{basic}}(p,q), m_c)M_c)$
    end for
  end for
end for
return $\left(\mathcal{P}_c(P^{\mathbf{basic}})\right)_c$.
```

In the following, we denote by barycenter the average among the third dimension of all similar patches $\overline{\tilde{P}}$ as defined in Equation (2), and by mean of the set of similar patches $\mathcal{P}(\tilde{P})$ the average value of all pixels of all similar patches

$$\frac{1}{\#\mathcal{P}_c(\tilde{P})\kappa^2} \sum_{\tilde{Q}_c \in \mathcal{P}_c(\tilde{P})} \sum_{p=1}^{\kappa} \sum_{q=1}^{\kappa} \tilde{Q}_c(p,q).$$

In the Gaussian white noise case with variance σ^2 one has $\mathbf{C}_n = \sigma^2\mathbf{I}$. The above "basic" estimate would be the optimal Bayesian estimate, if $\mathbf{C}_{\tilde{P}}$ and $\overline{\tilde{P}}$ were the true covariance matrix and expectation of the patches similar to $\tilde{P}$. In a second step of NL-Bayes, all the denoised patches obtained after the previous first step estimation can be reused to obtain a better unbiased estimation $\mathbf{C}_{\tilde{P}}^{\mathbf{basic}}$ for the covariance of the 3D group containing P. This is done by applying again (3), but to the denoised patches $\hat{Q}$ similar to the denoised patch $\hat{P}$, all of these patches being obtained after the basic denoising step. Similarly, a new estimation $\overline{\tilde{P}}^{\mathbf{basic}}$ of the average of patches similar to P can be obtained. This leads to a second Wiener-Bayes estimate described in Algorithm 2

$$P^{\mathbf{final}} = \overline{\tilde{P}}^{\mathbf{basic}} + \mathbf{C}_{\tilde{P}}^{\mathbf{basic}} \left[\mathbf{C}_{\tilde{P}}^{\mathbf{basic}} - \mathbf{C}_n\right]^{-1} \left(\tilde{P} - \overline{\tilde{P}}^{\mathbf{basic}}\right). \tag{4}$$

The final algorithm is described in Algorithms 3 and 4.

Algorithm 2 Description of the final estimation of similar patches.

```
Input : noisy 3D group $\mathcal{P}(\tilde{P})$,
Input : basic 3D group $\mathcal{P}^{\mathbf{basic}}(\tilde{P})$,
Input : For each intensity $\mathbf{i}$, the corresponding covariance matrix of the noise $\mathbf{C}_{n_\mathbf{i}}$.
Output : Final 3D group $\mathcal{P}(P^{\mathbf{final}})$.
for each $c = 1$ to $N_c$ do
    Compute the average intensity:
        $$\mathbf{i}_c = \frac{1}{\#\mathcal{P}_c(\tilde{P})\kappa^2} \sum_{\tilde{Q}_c \in \mathcal{P}_c(\tilde{P})} \sum_{p=1}^{\kappa} \sum_{q=1}^{\kappa} \tilde{Q}_c(p,q)$$
    Get the corresponding noise covariance matrix: $\mathbf{C}_{n_{\mathbf{i},c}}$
    Define a confidence interval for $\mathcal{P}_c(\tilde{P})$:
    $m_c = \min\left(\mathcal{P}_c(\tilde{P})\right) - \frac{\mathrm{Tr}(\mathbf{C}_{n_{\mathbf{i},c}})}{\#\mathcal{P}_c(\tilde{P})}$
    $M_c = \max(\mathcal{P}_c(\tilde{P})) + \frac{\mathrm{Tr}(\mathbf{C}_{n_{\mathbf{i},c}})}{\#\mathcal{P}_c(\tilde{P})}$
end for
Compute the barycenter $\bar{\tilde{P}}^{\mathbf{basic}}$                                  ▷ Equation (2)
Compute the covariance matrix of the basic patches $\mathbf{C}^{\mathbf{basic}}_{\tilde{P}}$:
    $$\mathbf{C}^{\mathbf{basic}}_{\tilde{P}} = \frac{1}{\#\mathcal{P}^{\mathbf{basic}}(\tilde{P}) - 1} \sum_{Q^{\mathbf{basic}} \in \mathcal{P}^{\mathbf{basic}}(\tilde{P})} \left(Q^{\mathbf{basic}} - \bar{\tilde{P}}^{\mathbf{basic}}\right)\left(Q^{\mathbf{basic}} - \bar{\tilde{P}}^{\mathbf{basic}}\right)^t$$
for each $Q^{\mathbf{basic}} \in \mathcal{P}^{\mathbf{basic}}(\tilde{P})$ do
    Compute the estimation $Q^{\mathbf{final}}$                                          ▷ Equation (4)
end for
Make sure that the estimated values are not absurd:
for $c = 1$ to $N_c$ do
    for $\tilde{Q}_c \in \mathcal{P}_c(\tilde{P})$ do
        for $(p,q) \in [\![1,\kappa]\!]^2$ do
            $Q^{\mathbf{final}}_c(p,q) = \min(\max(Q^{\mathbf{final}}_c(p,q), m_c)M_c)$
        end for
    end for
end for
return $\mathcal{P}(P^{\mathbf{final}})$.
```

Algorithm 3 Description of the first step of the modified NL-Bayes algorithm used in the Noise Clinic.

```
Input : Noisy image $\tilde{u}$,
Input : For each intensity $\mathbf{i}$, the corresponding covariance matrices of the noise $\left(\mathbf{C}_{n_{\mathbf{i},c}}\right)_c$, one for each channel,
Input : $w$ width and $h$ height of the image,
Output : Basic image $u^{\mathbf{basic}}$.
Initializations
Parameters initialization                                                    ▷ Section 4.2
Initialization of the mask of unused patches: $\forall (i,j) \in [\![1,h]\!] \times [\![1,w]\!], \mathrm{mask}(i,j) = \mathrm{true}$
Initialization of the aggregated estimated values: $\forall c \in [\![1,N_c]\!], \forall (i,j) \in [\![1,h]\!] \times [\![1,w]\!], v_c(i,j) = 0$
Initialization of the weight: $\forall c \in [\![1,N_c]\!], \forall (i,j) \in [\![1,h]\!] \times [\![1,w]\!], w_c(i,j) = 0$
Scan all pixels of the image
for each $(i_r,j_r) \in [\![1,h]\!] \times [\![1,w]\!]$ do
  Check that the current patch has not been processed yet
  if $\mathrm{mask}(i_r,j_r)$ then
    Compute the 3D group of similar patches $\mathcal{P}(\tilde{P})$
    for $(i,j) \in [\![i_r - \eta/2, i_r + \eta/2]\!] \times [\![j_r - \eta/2]\!] \times [\![j_r + \eta/2]\!]$ do
      $d(i\eta + j)\{1\} = (i,j)$
      Compute the distance: $d(i\eta + j)\{2\} = d(\tilde{Q}, \tilde{P})$                ▷ Equation (5)
    end for
    Select the similar patches randomly according to $d$                  ▷ Algorithm 5
    Compute the Bayes estimation                                           ▷ Algorithm 1
    Apply the homogeneous area trick                                       ▷ Algorithm 7
    Compute the aggregation of all estimates of this current 3D group
    for $c = 1$ to $N_c$ do
      for $\tilde{Q}_c \in \mathcal{P}_c(\tilde{P})$ do
        Let $(i_Q, j_Q)$ be the position of the top left pixel of the patch $Q$
        for $(p,q) \in [\![1,\kappa]\!]^2$ do
          $w_c(i_Q + p, j_Q + q) = w_c(i_Q + p, j_Q + q) + 1$
          $v_c(i_Q + p, j_Q + q) = v_c(i_Q + p, j_Q + q) + Q_c^{\mathbf{basic}}(p,q)$
        end for
      end for
    end for
    Update the mask
    for $\tilde{Q} \in \mathcal{P}(\tilde{P})$ do
      $\mathrm{mask}(i_Q, j_Q) = false$
    end for
  end if
end for
Compute the basic aggregation
for $c = 1$ to $N_c$ do
  for $(i,j) \in [\![1,h]\!] \times [\![1,w]\!]$ do
    $u_c^{\mathbf{basic}}(i,j) = \dfrac{v_c(i,j)}{w_c(i,j)}$
  end for
end for
return $u^{\mathbf{basic}}$
```

Algorithm 4 Description of the second step of the modified NL-Bayes algorithm used in the Noise Clinic.

```
Input : Noisy image $\tilde{u}$,
Input : Basic image $u^{\textbf{basic}}$,
Input : For each intensity $\mathbf{i}$, the corresponding multi-channel covariance matrix of the noise $\mathbf{C}_{n_{\mathbf{i}}}$,
Input : $w$ width and $h$ height of the image,
Output : Final denoised image $u^{\textbf{final}}$.
Initializations
Parameters initialization                                                    ▷ Section 4.2
Initialization of the mask of unused patches: $\forall (i,j) \in [\![1,h]\!] \times [\![1,w]\!], \text{mask}(i,j) = \text{true}$
Initialization of the aggregated estimated values: $\forall c \in [\![1,N_c]\!], \forall (i,j) \in [\![1,h]\!] \times [\![1,w]\!], v_c(i,j) = 0$
Initialization of the weight: $\forall c \in [\![1,N_c]\!], \forall (i,j) \in [\![1,h]\!] \times [\![1,w]\!], w_c(i,j) = 0$
Scan all pixels of the image
for each $(i_r, j_r) \in [\![1,h]\!] \times [\![1,w]\!]$ do
  Check that the current patch has not been processed yet
  if mask$(i_r, j_r)$ then
    Compute the 3D groups of similar multi-channel patches $\mathcal{P}(\tilde{P})$ and $\mathcal{P}^{\textbf{basic}}(\tilde{P})$
    for $(i,j) \in [\![i_r - \eta/2, i_r + \eta/2]\!] \times [\![j_r - \eta/2]\!] \times [\![j_r + \eta/2]\!]$ do
      $d(i\eta + j)\{1\} = (i,j)$
      Compute the distance: $d(i\eta + j)\{2\} = \frac{1}{N_c} \sum_{c=1}^{N_c} d(Q_c^{\textbf{basic}}, P_c^{\textbf{basic}})$
    end for
    Select the similar patches randomly according to $d$                     ▷ Algorithm 5
    Compute the Bayes estimation                                              ▷ Algorithm 2
    Compute the aggregation of all estimates of this current 3D group
    for $c = 1$ to $N_c$ do
      for $\tilde{Q} \in \mathcal{P}(\tilde{P})_c$ do
        Let $(i_Q, j_Q)$ be the position of the top left pixel of the patch $Q$
        for $(p,q) \in [\![1,\kappa]\!]^2$ do
          $w_c(i_Q + p, j_Q + q) = w_c(i_Q + p, j_Q + q) + 1$
          $v_c\,(i_Q + p, j_Q + q) = v_c\,(i_Q + p, j_Q + q) + Q^{\textbf{final}}(p,q)$
        end for
      end for
    end for
    Update the mask
    for $\tilde{Q} \in \mathcal{P}(\tilde{P})$ do
      mask$(i_Q, j_Q) = false$
    end for
  end if
end for
Compute the final aggregation
for $c = 1$ to $N_c$ do
  for $(i,j) \in [\![1,h]\!] \times [\![1,w]\!]$ do
    $u_c^{\textbf{final}}(i,j) = \frac{v_c(i,j)}{w_c(i,j)}$
  end for
end for
return $u^{\textbf{final}}$
```

2.1 Adaptation to Signal-Dependent Noise

Formulas (1) and (4) can be applied with a patch noise model $\mathbf{C}_n$ depending on each patch $\tilde{P}$. This application only requires an estimate of the covariance matrix of the noise associated with each group of similar patches. The algorithm computing this matrix is given in Section 3. The noise model being signal dependent, for each intensity $\mathbf{i}$ in the range intensity $[0, 255]$ of the image a noise covariance matrix $\mathbf{C}_{n\mathbf{i}}$ will be available. The noise model for each group of patches similar to $\tilde{P}$ will depend on $\tilde{P}$ through their mean $\mathbf{i}$. The reference intensity for the current 3D group $\mathcal{P}(\tilde{P})$ must therefore be estimated to apply formulas (1) and (4) with the appropriate noise covariance matrix. This intensity is simply estimated as the average of all pixels contained in $\mathcal{P}(\tilde{P})$.

2.2 Computation of the Distance

The original NL-Bayes algorithm computes the distances for the first step only on the luminance (Y) channel. The problem is that if two areas of different chrominance but similar luminance are close, then this distance will see both areas as unique. To avoid this bias, the distance during the first step will be computed over all channels Y, U, V. But as chrominance channels have more noise than the luminance channel, they are less precise. This justifies the use of a weighted distance as follows, based on the classic l^2 norm between patches denoted by d_2

$$d(\tilde{Q}, \tilde{P}) = \frac{1}{\Sigma_\alpha} \sum_{c=1}^{N_c} \alpha_c d_2(\tilde{Q}_c, \tilde{P}_c) \tag{5}$$

where

$$\Sigma_\alpha = \sum_{c=1}^{N_c} \alpha_c \text{ with } \alpha_Y = \sqrt{\frac{1}{2}} \qquad \alpha_U = \sqrt{\frac{1}{4}} \qquad \alpha_V = \sqrt{\frac{1}{4}}.$$

2.3 Selection of Similar Patches

The original NL-Bayes Algorithm [11] performs a selection of similar patches as follows:

- **Step 1:** The N_{min} most similar patches are kept, depending on their distance to the reference patch;
- **Step 2:** N_{min} most similar patches are chosen at least according to their distance to the reference patch. If the distance of the N_{min}^{th} patch is below a certain threshold, then $\mathcal{P}(\tilde{P})$ is composed of all patches of the neighbourhood for which the distance is below this threshold.

The most similar patches according to the l^2 distance may be influenced by the local structure of the noise, especially when working with small size patches. Moreover, if the image is processed line by line, in perfect homogeneous areas the similar patches will be all chosen on one line. As the algorithm does not take an already estimated patch as a new reference one (for speed-up purposes), all similar patches of the line will be discarded and then will get at the end only one estimation, which may create line artifacts.

To avoid this problem, a random patch selection will be used, as described in Algorithm 5. The selection of similar patches is done as follows:

1. A threshold τ is fixed depending on the trace of the covariance matrix of the noise $\text{Tr}(\mathbf{C}_n)$ for the first step ($\tau = 3 \sum_{c=1}^{N_c} \text{Tr}(\mathbf{C}_n)_c^2 \alpha_c^2$), or independently of the noise for the second step ($\tau = 3$);

2. The distance to the reference patch is recorded for the all patches of the search window;
3. Let N be the number of all the patches whose distance is below τ;
4.
    - If $N < N_{min}$, the N_{min} most similar patches are kept;
    - Else if $N \leq N_{max}$, the N most similar patches are kept;
    - Else, N_{max} patches are randomly selected among the N best.

Algorithm 5 Algorithm of random selection of similar patches depending on a distance table.

```
Input : List of distances and positions $d$,
Input : Similarity threshold $\tau$,
Input : Minimal number of similar patches $N_{min}$,
Input : Maximal number of similar patches $N_{max}$,
Input : Size of the search window $\eta$.
Output : Index of similar patches $index$,
Output : Number of similar patches kept $N$.
The list $d$ is sorted according to the distances, its first element is
$d_s(1) = \text{sort}(d, 1)$
Count the number of patches susceptible of being similar
if $d_s(N_{min}) > \tau$ then
    $N = N_{min}$
    for $n = 1$ to $N$ do
        $index(n) = d_s(n)\{2\}$
    end for
else
    Look for the maximal number of patches which distances are below $\tau$
    $N = 0$
    for $n \leq \eta^2$ do
        if $d(n)\{1\} \leq \tau$ then
            $N = N + 1$
        end if
    end for
    The similar patches are chosen randomly
    for $n = 1$ to $\min(N, N_m)$, randomly chosen in $[\![1, N]\!]$ do
        $index(n) = d_s(n)\{2\}$
    end for
end if
return $index$ and $N$.
```

2.4 Local Correction of the Covariance Matrix

If the matrices $\{\mathbf{C}_{n\mathbf{i}}\}_{\mathbf{i}\in[0,255]}$ are not accurate enough, denoising can cause artifacts from the first step. A conservative estimation strategy must be applied on the first Bayesian step to avoid noise overestimation artifacts generally due to texture. This strategy ensures that the noise variances are always smaller than the noisy patch variances. This sanity check based on the diagonal values of both $\mathbf{C}_{\tilde{P}}$ and $\mathbf{C}_n$ covariance matrices leads to the following more conservative estimate of the diagonal elements of the patch covariance matrix used in (1)

$$\forall p \in [\![0, \kappa^2 - 1]\!], \mathbf{C}_{\tilde{P}}(p,p) = \max\left(\mathbf{C}_{\tilde{P}}(p,p), \mathbf{C}_n(p,p)\right). \tag{6}$$

Indeed, by the independence of the patch and of the noise, the true covariance of the noisy patch is the sum of the covariance of the patch and of the covariance of the noise. Thus, their variances on the diagonal add, and therefore we should always have $\mathbf{C}_{\tilde{P}}(p,p) \geq \mathbf{C}_n(p,p)$.

This local correction is summarized in Algorithm 6.

Algorithm 6 Local correction of the covariance matrix.

```
Input : 3D group of similar patches $\mathcal{P}(\tilde{P})$;
Input : Patch size $\kappa$;
Input : Covariance matrix of the noise $\mathbf{C}_n$.
Output : $\mathbf{C}_{\tilde{P}}$, corrected empirical covariance matrix of patches similar to $\tilde{P}$.
Get $\mathbf{C}_{\tilde{P}}$.
for each $p = 0$ to $\kappa^2 - 1$ do
   for each $q = 0$ to $\kappa^2 - 1$ do
      $$\mathbf{C}_{\tilde{P}}(p,q) = \frac{1}{\#\mathcal{P}(\tilde{P}) - 1} \sum_{\tilde{Q} \in \mathcal{P}(\tilde{P})} \left(\tilde{Q}(p) - \overline{\tilde{P}}(p)\right)\left(\tilde{Q}(q) - \overline{\tilde{P}}(q)\right)$$
   end for
end for
Locally correct the covariance matrix
for each $p = 0$ to $\kappa^2 - 1$ do
   $\mathbf{C}_{\tilde{P}}(p,p) = \max\left(\mathbf{C}_{\tilde{P}}(p,p), \mathbf{C}_n(p,p)\right)$
end for
return $\mathbf{C}_{\tilde{P}}$.
```

2.5 Homogeneous Area Detection

The original NL-Bayes Algorithm [11] has a statistical test to determine if a 3D group belongs to an homogeneous area, and in this case the estimation of all the patches is replaced by the global mean over all the pixels contained in the 3D group. This criterion is merely based on the comparison of the empirical standard deviation of all pixels of $\mathcal{P}(\tilde{P})$ with σ^2.

In our generalization of this algorithm, σ doesn't exist since $\mathbf{C}_n \neq \sigma^2 \mathbf{I}$. So this criterion must be adapted to better take into account $\mathbf{C}_n$ in the following way:

- First, compute the traces of both covariance matrices for each channel c, $\mathrm{Tr}(\mathbf{C}_{\tilde{P}})$ and $\mathrm{Tr}(\mathbf{C}_n)$;
- To ensure that the 3D group belongs to an homogeneous area and not to an edge, the barycenter $\overline{\tilde{P}}$ of the 3D group is computed and compared to the mean of all the pixels of the 3D group $\overline{\overline{\tilde{P}}}$

$$D = \frac{1}{\kappa^2} \sum_{p,q}^{[\![0,\kappa-1]\!]^2} \left(\overline{\tilde{P}}(p,q) - \overline{\overline{\tilde{P}}}\right)^2.$$

Then, if $D \leq \gamma \mathrm{Tr}(\mathbf{C}_n)$, the 3D group is detected as belonging to an homogeneous area. Where γ is a thresholding parameters set to 1 in our implementation, used to determine whether a 3D group belongs to an homogeneous are or not.

- Denote by $\hat{\tilde{Q}}$ a first estimation of $\tilde{Q}$ obtained by (1). Then the basic estimate is $\forall \tilde{Q} \in \mathcal{P}(\tilde{P})$,

$$Q^{\textbf{basic}} = \begin{cases} \overline{\overline{\tilde{P}}} & \text{if } \mathrm{Tr}(\mathbf{C}_{\tilde{P}}) < k_L \mathrm{Tr}(\mathbf{C}_n) \\ \hat{\tilde{Q}} & \text{if } \mathrm{Tr}(\mathbf{C}_{\tilde{P}}) > k_H \mathrm{Tr}(\mathbf{C}_n) \\ \alpha \hat{\tilde{Q}} + (1-\alpha)\overline{\overline{\tilde{P}}} & \text{otherwise,} \end{cases} \tag{7}$$

  where

$$\alpha = \frac{\mathrm{Tr}(\mathbf{C}_{\tilde{P}}) - k_L \mathrm{Tr}(\mathbf{C}_n)}{k_H \mathrm{Tr}(\mathbf{C}_n) - k_L \mathrm{Tr}(\mathbf{C}_n)},$$

  and

$$\overline{\overline{\tilde{P}}} = \frac{1}{\#\mathcal{P}(\tilde{P})\kappa^2} \sum_{\tilde{Q}\in\mathcal{P}(\tilde{P})} \sum_{p=1}^{\kappa} \sum_{q=1}^{\kappa} \tilde{Q}(p,q).$$

  The thresholds (k_L, k_H) are chosen equal to $(4, 8)$ for the finest scale, and $(2, 4)$ for lower scales.

This homogeneous area detection is summarized in Algorithm 7.

Algorithm 7 Detection and estimation of homogeneous area.

```
Input : Bayesian estimation of the 3D group: $\{\hat{Q}, \tilde{Q} \in \mathcal{P}(\tilde{P})\}$,
Input : Mean of the noisy patches: $\overline{\overline{\tilde{P}}} = \frac{1}{\#\mathcal{P}(\tilde{P})\kappa^2} \sum_{\tilde{Q}\in\mathcal{P}(\tilde{P})} \sum_{p,q} \tilde{Q}(p,q)$,
Input : Barycenter of the noisy patches: $\overline{\tilde{P}}$,
Input : Trace of the covariance matrix of the noise: $\mathrm{Tr}(\mathbf{C}_n)$,
Input : Trace of the covariance matrix of the similar noisy patches: $\mathrm{Tr}(\mathbf{C}_{\tilde{P}})$,
Input : Thresholding parameters: $k_L$, $k_H$, $\gamma$.
Output : Final estimation of the 3D group before aggregation.
Computation of the difference to the barycenter
    $$D = \frac{1}{\kappa^2} \sum_{p,q}^{[\![0,\kappa-1]\!]^2} \left(\overline{\tilde{P}}(p,q) - \overline{\overline{\tilde{P}}}\right)^2$$
Detection if truly homogeneous area
if $D \le \gamma \mathrm{Tr}(\mathbf{C}_n)$ and $\mathrm{Tr}(\mathbf{C}_{\tilde{P}}) \le k_H \mathrm{Tr}(\mathbf{C}_n)$ then
  if $\mathrm{Tr}(\mathbf{C}_{\tilde{P}}) \le k_L \mathrm{Tr}(\mathbf{C}_n)$ then
      $$\forall \tilde{Q} \in \mathcal{P}(\tilde{P}), \forall (p,q) \in [\![0,\kappa-1]\!]^2, \hat{Q}(p,q) = \overline{\overline{\tilde{P}}}$$
  else
      $$\alpha = \frac{\mathrm{Tr}(\mathbf{C}_{\tilde{P}}) - k_L \mathrm{Tr}(\mathbf{C}_n)}{k_H \mathrm{Tr}(\mathbf{C}_n) - k_L \mathrm{Tr}(\mathbf{C}_n)}$$
      $$\forall \tilde{Q} \in \mathcal{P}(\tilde{P}), \forall (p,q) \in [\![0,\kappa-1]\!]^2, \hat{Q}(p,q) = \alpha \hat{Q}(p,q) + (1-\alpha)\overline{\overline{\tilde{P}}}$$
  end if
end if
return $\{\hat{Q}\}_{\tilde{Q}\in\mathcal{P}(\tilde{P})}$
```

3 Obtaining the Covariance Matrix of Noise Patches

In order to help the reader with the new notations introduced in this section, a summary is presented in tables 3 and 4, including new notations and parameters.

DCT	Discret Cosine Transform
$\mathcal{D}$	DCT matrix
$N_{\mathbf{i}}$	$\kappa \times \kappa$ stochastic noise patch model at intensity $\mathbf{i}$
$\mathbf{M}_{\mathbf{i}}$	estimated variance matrix in DCT space at intensity $\mathbf{i}$

Table 3: Notations introduced in Section 3.

N_b	number of bins used during the noise estimation (depends on the size of the image)
N_f	number of iterations for the matrix filtering algorithm
S_f	size of the filtering window

Table 4: Parameter notations introduced in Section 3.

Colom et al., [4] and [5], proposed an adaptation of the Ponomarenko et al. [19] method estimating a frequency dependent noise to estimate noise in JPEG images. Given a patch size $\kappa \times \kappa$, the method extracts from the image a set with fixed cardinality of sample blocks with very similar patches in DCT space, which are therefore likely to contain only noise. These noise blocks are transformed by a DCT, and an empirical standard deviation of their DCT coefficients is computed. This gives a noise model that is proved in [4] and [5] to be accurately consistent with the noise observed in JPEG images. This algorithm computes for every intensity $\mathbf{i}$ a multi-frequency noise estimate given by a $\kappa^2 \times \kappa^2$ matrix

$$\mathbf{M}_{\mathbf{i}} := \mathbb{E}\left(\mathcal{D}N_{\mathbf{i}}\left(\mathcal{D}N_{\mathbf{i}}\right)^t\right) \tag{8}$$

where:

- $\mathcal{D}$ is the $\kappa^2 \times \kappa^2$ matrix of the discrete cosine transform (DCT) ;
- $N_{\mathbf{i}}$ denotes the $\kappa \times \kappa$ stochastic noise patch model at intensity $\mathbf{i}$.

For a given intensity $\mathbf{i}$, the covariance matrix of the noise is by definition

$$\mathrm{Cov}(N_{\mathbf{i}}) = \mathbb{E}\left(N_{\mathbf{i}}N_{\mathbf{i}}^t\right)$$

which leads to

$$\begin{aligned} \mathcal{D}\mathrm{Cov}(N_{\mathbf{i}})\mathcal{D}^t &= \mathcal{D}\mathbb{E}\left(N_{\mathbf{i}}N_{\mathbf{i}}^t\right)\mathcal{D}^t \\ &= \mathbb{E}\left(\mathcal{D}N_{\mathbf{i}}N_{\mathbf{i}}^t\mathcal{D}^t\right) \\ &= \mathbb{E}\left(\mathcal{D}N_{\mathbf{i}}\left(\mathcal{D}N_{\mathbf{i}}\right)^t\right) \\ &= \mathbf{M}_{\mathbf{i}} \end{aligned} \tag{9}$$

thanks to Equation (8). Since $\mathcal{D}^{-1} = \mathcal{D}^t$, then from Equation (9) we get

$$\mathrm{Cov}(N_{\mathbf{i}}) = \mathcal{D}^t\mathbf{M}_{\mathbf{i}}\mathcal{D}. \tag{10}$$

This obtention is summarized in Algorithm 8. Since the estimation of the covariance matrix of the noise is channel independent, the pseudo-code only describes the computation for one channel.

Algorithm 8 Obtention of $\mathbf{C}_n$ from the estimation of the variances of the noise in DCT space.

```
Input : Set of diagonal matrices of the estimation of the variances of the noise in DCT space
$(\mathbf{M_i})_\mathbf{i}$,                                    ▷ Equation (8)
Input : DCT matrix $\mathcal{D}$,
Input : Patch size $\kappa$.
Output : Set of covariance matrices of the noise $(\mathbf{C}_{n_\mathbf{i}})_\mathbf{i}$.
for each i do
  for each $p = 0$ to $\kappa^2 - 1$ do
    for each $q = 0$ to $\kappa^2 - 1$ do
        $\mathbf{C}_{n_\mathbf{i}}(p,q) = \sum_{k=0}^{\kappa^2-1} \mathcal{D}(k,p)\, \mathbf{M_i}(k,k)\, \mathcal{D}(k,q)$
    end for
  end for
end for
return $(\mathbf{C}_{n_\mathbf{i}})_\mathbf{i}$.
```

3.1 Are Noise Covariances Negligible in the Block DCT Space?

The method summarized in the preceding section only estimates the variances of the DCT coefficients of noise blocks and not their covariances. The covariance matrices are therefore assumed to be diagonal, which amounts to assume that the DCT decorrelates the noise. A formal argument can be given in favor of this assumption. Assume that the initial image noise was white Gaussian, and that the image has undergone a symmetric, real, periodic linear filter H. Then this filter corresponds to applying a diagonal operator to the image in the DCT frequency domain. Thus the noise covariance of the filtered noise remains diagonal in the DCT domain. Yet, this argument is only valid for a *global* image DCT. Here, because we need a signal dependent noise model, we are estimating it on *local* DCTs applied to each block. It is therefore no more true that the blocks have undergone a periodic convolution filter. Thus, it cannot be *exactly* true that after the application of a global linear filter, the noise block DCTs have a diagonal covariance. To check nonetheless the quantitative validity of this assumption, we tested three different filters applied to a white noise:

- $\mathbf{H}_1$ with coefficients $\frac{1}{4}\begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}$ supported by the pixels $(-\frac{1}{2},-\frac{1}{2})$, $(\frac{1}{2},\frac{1}{2})$, $(\frac{1}{2},-\frac{1}{2})$, $(-\frac{1}{2},\frac{1}{2})$;
- $\mathbf{H}_2$ the centered filter with coefficients $\frac{1}{16}\begin{pmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{pmatrix}$;
- $\mathbf{H}_3$ the centered filter with coefficients $\frac{1}{88}\begin{pmatrix} 1 & 2 & 4 & 8 & 4 & 2 & 1 \\ 2 & 4 & 8 & 16 & 8 & 4 & 2 \\ 1 & 2 & 4 & 8 & 4 & 2 & 1 \end{pmatrix}$.

The noise image $\tilde{u}$ was a 256×256 Gaussian white noise with mean 128, and standard deviation $\sigma = 20$. After convolution, we extracted N distinct $\kappa \times \kappa$ patches $\{P_n\}_{n \in N}$ from the image and a 2D normalized DCT was applied on them. Finally, their empirical $\kappa^2 \times \kappa^2$ covariance matrix $\mathbf{C}$ was computed as

$$\begin{aligned}
&\forall (p,q),(i,j) \in [\![0,\kappa-1]\!]^2, \\
&\mathbf{C}(p,q,i,j) = \frac{1}{N}\sum_{n=1}^{N}\hat{P}(p,q)\hat{P}(i,j) \\
&-\frac{1}{N^2}\left(\sum_{n=1}^{N}\hat{P}(p,q)\right)\left(\sum_{n=1}^{N}\hat{P}(i,j)\right).
\end{aligned} \tag{11}$$

These covariances matrices can be visualized by the absolute value of their coefficients $|C_{i,j}|$, normalized in $[0,1]$ so that the largest coefficient is set equal to 1, and the smallest equal to 0. The following colour code is used in the visualization: a coefficient appears in blue if it is near 0; in green if it is near 0.5 and in red if it is near 1. The results for various patch sizes are shown in Figure 1, 2 and 3. These illustrations and the quantitative tables 5, 6 and 7 confirm that the block DCT noise covariance matrices are nearly diagonal.

κ	4	6	8	16
mean $\{\lvert C_{i,j}\rvert\}_{i\neq j}$	0.83	0.48	0.31	0.10
mean $\{\lvert C_{i,j}\rvert\}_{i=j}$	24.89	25.31	24.95	24.73
median $\{\lvert C_{i,j}\rvert\}_{i\neq j}$	0.04	0.03	0.02	0.01
median $\{\lvert C_{i,j}\rvert\}_{i=j}$	19.42	17.14	16.28	14.41

Table 5: Statistics of the estimated DCT covariance matrix of noise filtered by $\mathbf{H}_1$.

κ	4	6	8	16
mean $\{\lvert C_{i,j}\rvert\}_{i\neq j}$	0.48	0.28	0.19	0.06
mean $\{\lvert C_{i,j}\rvert\}_{i=j}$	14.59	13.95	14.45	14.23
median $\{\lvert C_{i,j}\rvert\}_{i\neq j}$	0.010	0.008	0.005	0.002
median $\{\lvert C_{i,j}\rvert\}_{i=j}$	6.75	4.50	3.77	2.35

Table 6: Statistics of the estimated DCT covariance matrix of noise filtered by $\mathbf{H}_2$.

κ	4	6	8	16
mean $\{\lvert C_{i,j}\rvert\}_{i\neq j}$	0.22	0.15	0.10	0.04
mean $\{\lvert C_{i,j}\rvert\}_{i=j}$	9.28	8.94	9.04	8.51
median $\{\lvert C_{i,j}\rvert\}_{i\neq j}$	0.020	0.016	0.010	0.003
median $\{\lvert C_{i,j}\rvert\}_{i=j}$	3.32	2.86	2.44	1.73

Table 7: Statistics of the estimated DCT covariance matrix of noise filtered by $\mathbf{H}_3$.

As one can see, strict covariance coefficients are really small before variance coefficients, which strengthen the DCT decorrelation assumption. So from now on, only variance coefficients will be considered in DCT space.

3.2 Covariance Matrix Filtering

Since the noise covariance matrices can only be estimated for sparse bins in the intensity range, an interpolation must be applied to obtain a covariance matrix of the noise for each intensity. The

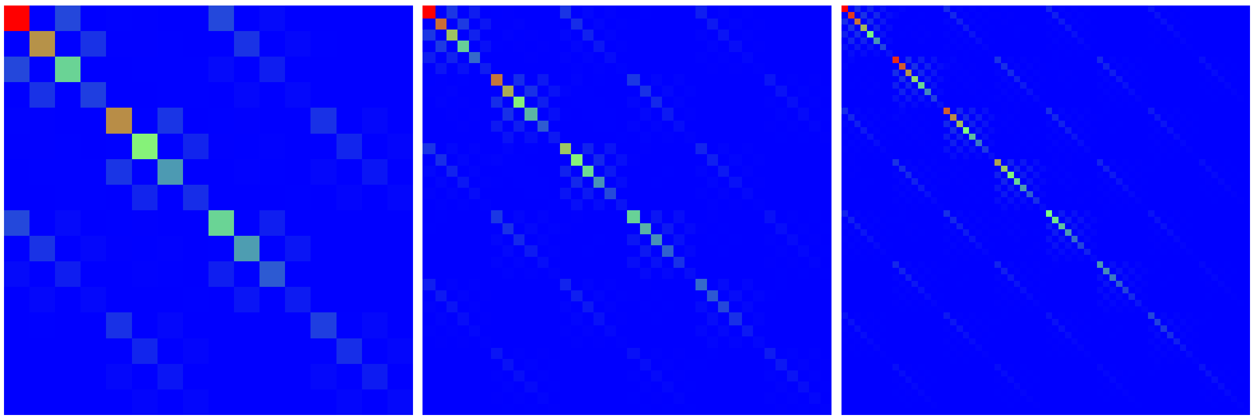


Figure 1: Visualization of the noise covariance matrices in DCT space after applying filter $\mathbf{H}_1$ to illustrate that it is almost diagonal. From left to right, patch size $\kappa = 4, 6, 8$.

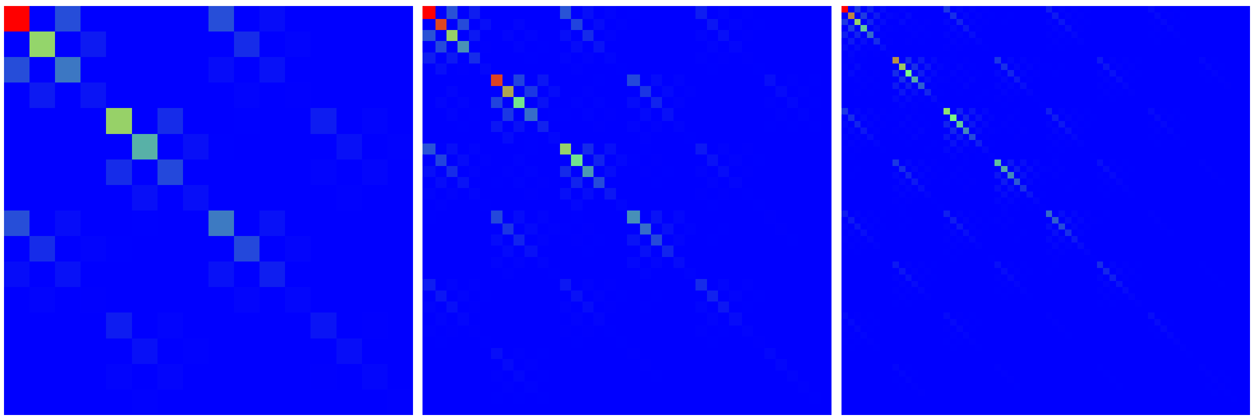


Figure 2: Visualization of the noise covariance matrices in DCT space after applying filter $\mathbf{H}_2$ to illustrate that it is almost diagonal. From left to right, patch size $\kappa = 4, 6, 8$.

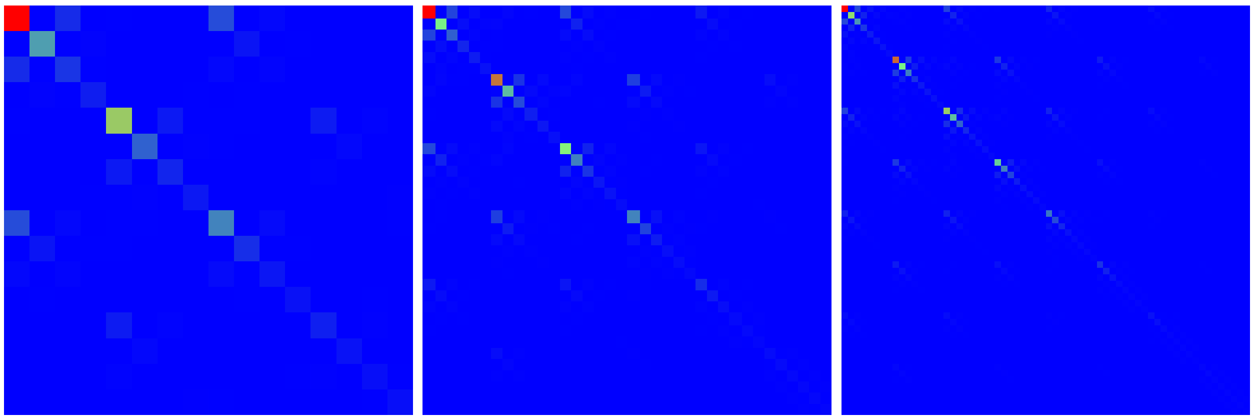


Figure 3: Visualization of the noise covariance matrices in DCT space after applying filter $\mathbf{H}_3$ to illustrate that it is almost diagonal. From left to right, patch size $\kappa = 4, 6, 8$.

covariance matrices must be smoothed before such an interpolation. This can be obtained by a regularization of the covariance matrices in DCT space before applying the inverse DCT to get back a covariance matrix in the image domain. We found that a robust regularization could be performed as follows:

1. For each frequency independently, perform a linear interpolation between the bin values to obtain a noise curve for this frequency, giving the variance as a function of the signal $\mathbf{i}$. Smooth this curve by applying a sliding average;

2. For every bin, replace each matrix coefficient by the median of its four neighbours and itself.

This filtering is summarized in Algorithm 9. Since the filtering is channel independent, the pseudo-code only describes the filtering for one channel.

Algorithm 9 Covariance matrix filtering.

```
Input : Number of bins $N_b$;
Input : Patch size $\kappa$;
Input : Set of covariance matrices $\{\mathbf{C}_b\}_{b \in N_b}$ of size $\kappa^2 \times \kappa^2$
Input : Iteration number $N_f$;
Input : Filtering window size $S_f$;
Input : Mean values for each bin $\{m_b\}_{b \in N_b}$.
Output : Set of filtered covariance matrices $\{\tilde{\mathbf{C}}_b\}_{b \in N_b}$.
Iteration of the filtering
for each scale $n = 1$ to $N_f$ do
   Regularize frequency among matrices
   for $(p,q) \in [1,\kappa^2] \times [1,\kappa^2]$ do
      Interpolate standard deviation for each intensity value $\mathbf{i} \in [0,255]$ into $\{V_\mathbf{i}\}_{\mathbf{i} \in [0,255]}$
      for each $b = 1$ to $N_b$ do
         $\alpha = \dfrac{\mathbf{C}_b(p,q)^2 - \mathbf{C}_{b-1}(p,q)^2}{m_b - m_{b-1}}$
         $\beta = \dfrac{\mathbf{C}_{b-1}(p,q)^2 m_b - \mathbf{C}_b(p,q)^2 m_{b-1}}{m_b - m_{b-1}}$
         for each $\mathbf{i} = m_{b-1} + 1$ to $m_b$ do
            $V_\mathbf{i} = \alpha \mathbf{i} + \beta$.
         end for
      end for
      Smooth bin values
      for each $b = 1$ to $N_b$ do
         $\mathbf{C}_b(p,q) = \left( \dfrac{1}{2S_f + 1} \sum_{\mathbf{i}=-S_f}^{S_f} V_\mathbf{i} \right)^{\frac{1}{2}}$
      end for
   end for
   Regularize matrices
   for each $b = 1$ to $N_b$ do
      For each frequency, replace it by the median of its neighbours
      for $(p,q) \in [1,\kappa^2] \times [1,\kappa^2]$ do
         $\tilde{\mathbf{C}}_b(p,q) = \sqrt{median\left(\mathbf{C}_b(p \pm 1, q \pm 1)^2, \mathbf{C}_b(p,q)^2\right)}$
      end for
   end for
end for
return $\{\tilde{\mathbf{C}}_b\}_{b \in N_b}$.
```

3.3 Computing the DCT Matrix

A $\kappa \times \kappa$ patch P is represented in 2D by a matrix

$$P = \begin{pmatrix} p_{1,1} & \cdots & p_{1,\kappa} \\ \vdots & & \vdots \\ p_{\kappa,1} & \cdots & p_{\kappa,\kappa} \end{pmatrix}, \tag{12}$$

or in 1D by a vector

$$P = \left(p_{1,1}, \cdots, p_{1,\kappa}, p_{2,1}, \cdots, p_{2,\kappa}, \cdots, p_{\kappa,1}, \cdots, p_{\kappa,\kappa}\right)^t. \tag{13}$$

This section defines a $\kappa^2 \times \kappa^2$ matrix $\mathcal{D}$ such that

$$\mathcal{D}P = DCT(P),$$

where DCT is the two-dimensional orthonormal forward discrete cosine transform (DCT) defined by

$$\begin{aligned} \forall (i,j) \quad \in [\![1,\kappa]\!]^2, \quad & DCT(x_{i,j}) = X_{i,j} \\ = \qquad & \alpha_{i,j} \sum_{p=0}^{\kappa-1} \sum_{q=0}^{\kappa-1} x_{p,q} \cos\left[\frac{\pi}{\kappa}\left(p + \frac{1}{2}\right) i\right] \\ & \cos\left[\frac{\pi}{\kappa}\left(q + \frac{1}{2}\right) j\right], \end{aligned} \tag{14}$$

and $\alpha_{i,j}$ are the normalization coefficients

$$\alpha_{i,j} = \begin{cases} \frac{1}{\kappa} & \text{if } (i,j) = (0,0) \\ \frac{\sqrt{2}}{\kappa} & \text{if } i = 0 \text{ and } j > 0 \text{ or } j = 0 \text{ and } i > 0 \\ \frac{2}{\kappa} & \text{otherwise.} \end{cases} \tag{15}$$

According to the indexation of Equation (13), one can set the indices of $\mathcal{D}$ coefficients as $(i,j) \in [\![1,\kappa^2]\!] \times [\![1,\kappa^2]\!]$, $i = a\kappa + b$, $(a,b) \in [\![1,\kappa]\!] \times [\![1,\kappa]\!]$ and $j = m\kappa + n$, $(m,n) \in [\![1,\kappa]\!] \times [\![1,\kappa]\!]$. Thus we have

$$\begin{aligned} \mathcal{D}(i,j) &= \mathcal{D}(a\kappa + b, m\kappa + n) \\ &= \alpha_{a-1,b-1} \cos\left[\frac{\pi}{\kappa}\left(m - 1 + \frac{1}{2}\right)(a-1)\right] \\ & \quad \cos\left[\frac{\pi}{\kappa}\left(n - 1 + \frac{1}{2}\right)(b-1)\right]. \end{aligned}$$

4 The Multiscale Algorithm

4.1 The Mean Sub- and Up-Sampling Method

Classic denoising algorithms such as BM3D (Dabov et al. [7]), NL-means (Buades et al. [2]), K-SVD (Mairal et al. [17], [18]), Wiener filters applied on DCT (Yaroslavsky et al. [26], [25]) or on wavelet transform (Donoho et al. [24]) and the total variation minimization (Rudin et al. [23]) achieve good results for moderate noise ($\sigma \leq 20$). Yet for larger noise artifacts inherent to each method (and different for each method) start appearing. In particular all keep an often disturbing low frequency noise. A natural idea to deal with low frequency noise is to involve a coarse to fine multiscale procedure denoising the image progressively at all scales. We refer to [13] for a discussion of the advantages of a multiscale procedure. The proposed multiscale algorithm subsamples recursively the image into four channels that are partly redundant. The four channels are obtained by moving the sub-sampling grid by respectively $(0,0)$, $(1,0)$, $(0,1)$, $(1,1)$. In that way there is enough information for up-sampling after denoising the denoised images at the lower scale.

We shall denote by s the current dyadic scale of the multiscale algorithm. For the particular case of white noise, the aim of the sub-sampling is to obtain from $\tilde{u}_s$ an image $\tilde{u}_{s+1}$ where the standard deviation of the noise has been divided by two compared to the noise contained in $\tilde{u}_s$. To get this result, one can use a filter $f(i,j)$ satisfying

$$\sum_{i,j} f(i,j) = 1 \quad \text{and} \quad \sum_{i,j} f(i,j)^2 = \frac{1}{4}.$$

The simplest filter coping with these conditions is the average filter $\mathbf{F}$, defined by

$$\mathbf{F}(i,j) = \begin{cases} \frac{1}{4} & \text{if } (i,j) \in [(0,0),(0,1),(1,0),(1,1)]\,, \\ 0 & \text{otherwise.} \end{cases}$$

which averages each group of four adjacent neighboring pixels. There are four different filter+sub-sample results, as shown in Figure 4. Moreover if the image $\tilde{u}_s$ is well-sampled, so is $\tilde{u}_s * \mathbf{F}$. Thus, the difference image is not aliased. Since all sub-sampled images are available, the noise estimation can

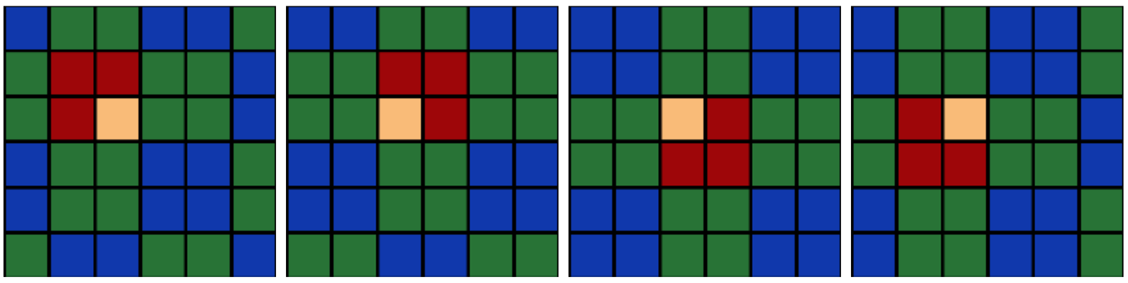


Figure 4: Four different ways to average red neighbors of the yellow reference pixel.

work with the same amount of samples at every scale. All sub-sampled images must also be denoised. They will be processed jointly in a single image, in order to avoid creating artificial borders. The four sub-sampled images are regrouped in one mosaic image, as shown in Figure 5 and described in Algorithm 11. The boundaries of the sub-images are in that way better denoised, because they are included in a smooth larger image. The sub-sampling method is summarized in Algorithm 10. As it is channel independent, the pseudo-code only describes the sub-sampling for one channel.

The aim of the up-sampling is to go back to the upper scale, after denoising the four sub-images obtained by sub-sampling as seen in Section 4.1. The four sub-images $\tilde{u}_1$, $\tilde{u}_2$, $\tilde{u}_3$ and $\tilde{u}_4$ have their pixel center (resp. in red, purple, green and blue in Figure 6) located at the center of four pixels of $\tilde{u}$ (in black in Figure 6). Thus they are shifted by $\pm\frac{1}{2}$ in both coordinate directions. The reconstruction of the pixels of $\tilde{u}$ (see the example of the pixel in yellow in Figure 6) will be done by averaging their four neighbors, each one belonging to each sub-image.

The up-sampling method is summarized in Algorithm 12. As it is channel independent, the pseudo-code only describes the up-sampling for one channel.

Contrarily to raw images, in JPEG images the noise increases at lower scales, as illustrated in Figure 7, which are the noise curves of the image shown in Figure 21. This figure displays average noise curves for high and low frequencies respectively, in the three scales noise estimation from a JPEG image. The low-frequency noise is not altered by JPEG and becomes a high-frequency noise after three[2] subsampling operations.

[2]Since JPEG transform is based on the 8×8 DCT transform, after three subsamplings the 8×8 pixels patches

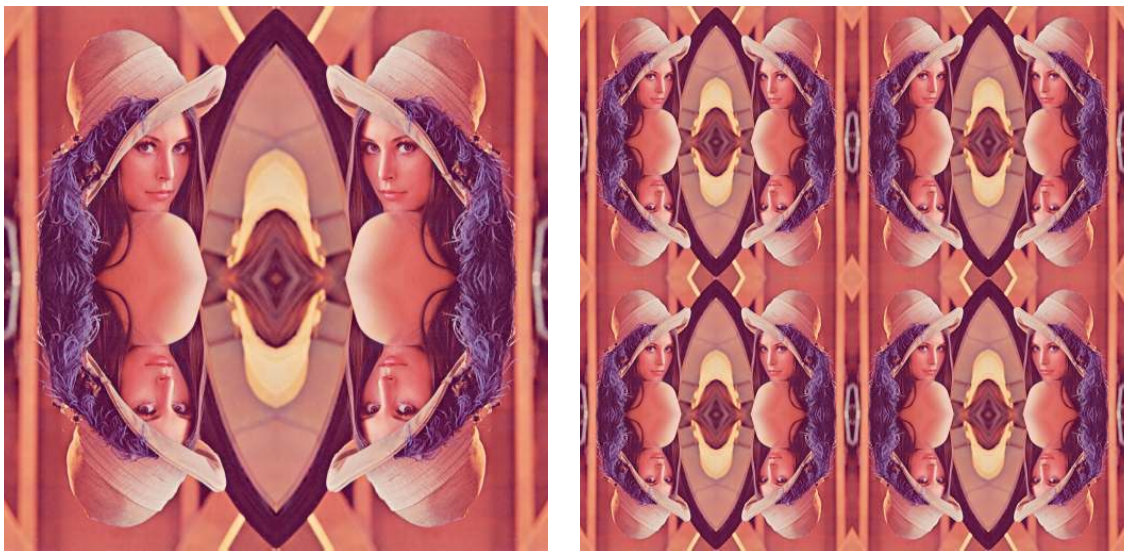


Figure 5: Left: mosaic of the scale 1 sub-images. Right: mosaic of the scale 2 sub-images, The input image has scale 0.

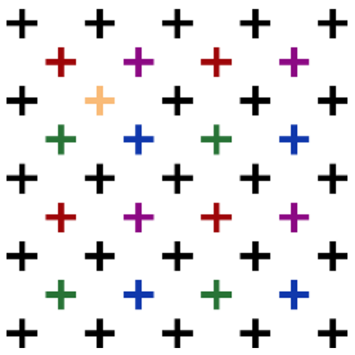


Figure 6: Position of the center of pixels in the original image $\tilde{u}$ in black, in the four sub-images $\tilde{u}_1$ in red, $\tilde{u}_2$ in purple, $\tilde{u}_3$ in green and $\tilde{u}_4$ in blue. The yellow pixel will be reconstructed by averaging the top left red pixel, the top right purple pixel, the bottom left green pixel and the bottom right blue pixel of its four pixel neighborhood.

The whole coarse to fine multiscale procedure of the Noise Clinic is summarized in Algorithm 13. During the sub-sampling the four sub-images are kept and assembled in a mosaic to be denoised together. It follows that for each scale, the mosaic keeps the original image size. Thus the complexity for the algorithm is approximately equal to N times the complexity of the one scale algorithm.

4.2 Parameters

Even if the final algorithm is fully automatic, and the only parameter tunable by the user is the number of scales, other parameters listed below are part of the algorithm:

- Patch size κ;
- Number of similar patches* N_{min};

become a single pixel. Thus, at the third scale the noise is only high-frequency and uncorrelated.

Algorithm 10 Sub-sampling of one image into four sub-images.

```
Input : Image $u$ of size $w \times h$.
Output : Four sub-images $(u_k)_{k \in [\![1,4]\!]}$ of size $\frac{w}{2} \times \frac{h}{2}$.
Check the size of $u$
if $w \equiv 1 \pmod 2$ then
  Add one column to $u$ by symmetry.
end if
if $h \equiv 1 \pmod 2$ then
  Add one line to $u$ by symmetry.
end if
Average the pixels
for each $i = 0$ to $\frac{h}{2} - 1$ do
  for each $j = 0$ to $\frac{w}{2} - 1$ do
    Get the first sub-image
    $u_1(i,j) = \frac{1}{4}\left(u(2i,2j) + u(2i+1,2j) + u(2i,2j+1) + u(2i+1,2j+1)\right)$
    Get the second sub-image
    if $j = \frac{w}{2} - 1$ then
      $u_2(i,j) = \frac{1}{2}\left(u(2i,2j+1) + u(2i+1,2j+1)\right)$
    else
      $u_2(i,j) = \frac{1}{4}\left(u(2i,2j+1) + u(2i+1,2j+1) + u(2i,2j+2) + u(2i+1,2j+2)\right)$
    end if
    Get the third sub-image
    if $i = \frac{h}{2} - 1$ then
      $u_3(i,j) = \frac{1}{2}\left(u(2i+1,2j) + u(2i+1,2j+1)\right)$
    else
      $u_3(i,j) = \frac{1}{4}\left(u(2i+1,2j) + u(2i+2,2j) + u(2i+1,2j+1) + u(2i+2,2j+1)\right)$
    end if
    Get the fourth sub-image
    if $i = \frac{h}{2} - 1$ and $j = \frac{w}{2} - 1$ then
      $u_4(i,j) = \left(u(2i+1,2j+1)\right)$
    else if $i = \frac{h}{2} - 1$ then
      $u_4(i,j) = \frac{1}{2}\left(u(2i+1,2j+1) + u(2i+1,2j+2)\right)$
    else if $j = \frac{w}{2} - 1$ then
      $u_4(i,j) = \frac{1}{2}\left(u(2i+1,2j+1) + u(2i+2,2j+1)\right)$
    else
      $u_4(i,j) = \frac{1}{4}\left(u(2i+1,2j+1) + u(2i+2,2j+1) + u(2i+1,2j+2) + u(2i+2,2j+2)\right)$
    end if
  end for
end for
return $(u_k)_{k \in [\![1,4]\!]}$.
```

- Maximal number of similar patches N_{max};
- Size of the search window* η;
- Threshold to choose more similar patches*;
- Thresholds for homogeneous area k_L and k_H;
- Number of scales N;

Algorithm 11 Construct a mosaic image from a set of sub-images.

```
Input : Set of sub-images $\{u_s^k\}_{k\in[\![1,4^s]\!]}$ at the current scale $s$,
Input : Size $(w_s, h_s)$ of sub-images.
Output : Size $(w_m, h_m)$ of output mosaic image,
Output : Mosaic image $u_m$.
Obtention of the size of the mosaic
$N = \sqrt{4^s}$
$w_m = N w_s$
$h_m = N h_s$
Loop over sub-images
for $p = 0$ to $N-1$ do
  for $q = 0$ to $N-1$ do
    Loop over pixels
    for $i = 0$ to $h_s - 1$ do
      for $j = 0$ to $w_s - 1$ do
        Position initialization
        $n = pN + q$
        if $p \equiv 0 \pmod 2$ then
          $d_i = i$
        else
          $d_i = h_s - 1 - i$
        end if
        if $q \equiv 0 \pmod 2$ then
          $d_j = j$
        else
          $d_j = w_s - 1 - j$
        end if
        Copy pixel values into the mosaic
        $u_m(d_i + ph_s + i, d_j + qw_s + j) = u_s^n(i, j)$
      end for
    end for
  end for
end for
return $(w_m, h_m)$ and $u_m$.
```

- Multiplicative factor γ_f.

Parameters followed by * are directly dependent on the size of the patches, as determined in [12]. The other parameters will be discussed in this section.

4.2.1 Influence of the Patch Size κ

As shown in [12], the patch size directly depends on the noise strength. Here, the noise is signal dependent, and it will be very difficult to adapt the patch size to the intensity (and noise level) of the current 3D group. Furthermore, the size of the patches used during the noise estimation is the same as the one used during the denoising. To avoid useless complexity, a fixed size was chosen for all intensity values. It must be as small as possible, to avoid loosing detail by inaccurate local patch models in the Bayesian estimate. Furthermore, the algorithm being multiscale, the real size in ratio to the original image size doubles at each scale, which justifies again taking the smallest possible

Algorithm 12 Up-sampling of four sub-images into one.

```
Input : Four sub-images $(u_k)_{k\in[\![1,4]\!]}$ of size $\frac{w}{2} \times \frac{h}{2}$.
Output : Image $u$ of size $w \times h$.
for each $i = 0$ to $h-1$ do
  for each $j = 0$ to $w-1$ do
    Get the indexes of sub-images
    $p_1 = \lfloor \frac{i}{2} \rfloor$;   $p_2 = \lfloor \frac{i-1}{2} \rfloor$;   $q_1 = \lfloor \frac{j}{2} \rfloor$;   $q_2 = \lfloor \frac{j-1}{2} \rfloor$.
    Top left pixel
    if $i = 0$ and $j = 0$ then
      $u(i,j) = u_1(p_1,q_1)$
      First row
    else if $i = 0$ then
      $u(i,j) = \frac{1}{2}(u_1(p_1,q_1) + u_2(p_1,q_2))$
      First column
    else if $j = 0$ then
      $u(i,j) = \frac{1}{2}(u_1(p_1,q_1) + u_3(p_2,q_1))$
      All the rest of the image
    else
      $u(i,j) = \frac{1}{4}(u_1(p_1,q_1) + u_2(p_1,q_2) + u_3(p_2,q_1) + u_4(p_2,q_2))$
    end if
  end for
end for
return $u$.
```

size. For these combined reasons, the size of patches for all steps and all scales in the Noise Clinic was fixed to $\kappa = 4$.

4.2.2 Maximal Number of Similar Patches N_{max}

The aim of this parameter is to allow the possibility to chose the similar patches randomly. Since in homogeneous areas the color estimation is obtained as an average of the colors of the whole 3D group, we need the largest possible number of similar patches to get the smallest noise influence on it.

Yet taking $N_{max} = \eta^2$ is not optimal. Indeed, the algorithm is sped up by never taking as reference patch a patch that has previously been taken in group of similar patches. Thus in homogeneous areas all patches of the search window will not be re-used as reference patches. This may cause staircase artifacts in homogeneous areas. In order to avoid them, one solution is to chose randomly the similar patches, which leads to chose N_{max} smaller than η^2. According to these arguments, $N_{max} = \dfrac{\eta^2}{2}$ is a good compromise. The interest of the random selection among the eligible patches is illustrated in Figure 8.

4.2.3 Influence of the Threshold of Similarity τ

The number of eligible patches for the random selection depends directly on τ, being defined by $\{\tilde{Q}$ such that $d(\tilde{Q},\tilde{P}) \leq \tau^{\mathbf{basic}}\}$ for the first step, and by $\{\tilde{Q}$ such that $d(\tilde{Q},\tilde{P}) \leq \tau^{\mathbf{final}}\}$ for the second step.

The value of $\tau^{\mathbf{final}}$ is the same as the one determined in [12], i.e. $\tau^{\mathbf{final}} = 3$. The value of $\tau^{\mathbf{basic}}$ is

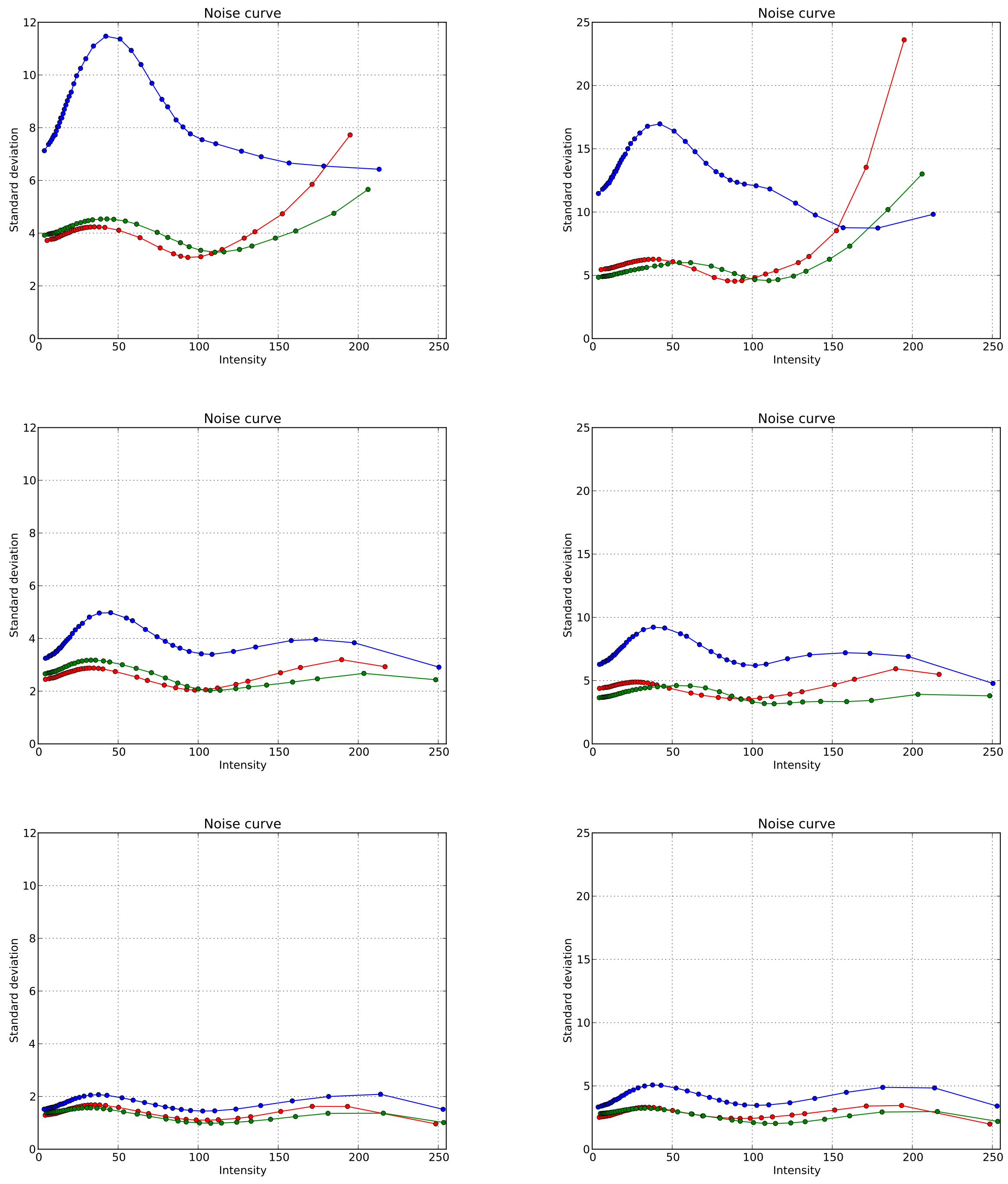


Figure 7: Average noise curves for a typical JPEG-encoded image (shown in Figure 21). From left to right: low frequencies, high frequencies. From top to bottom: scale 2, scale 1, scale 0. Instead of being divided by two at each scale (as it should happen with white noise), the noise grows in lower scales, where JPEG has not removed it.

set to $3\sum_{c=1}^{N_c}\alpha_c^2\mathrm{Tr}(\mathbf{C}_n)_c^2$, since typically the distance between similar patches corrupted by Gaussian noise of standard deviation σ should be below $3\sigma^2$.

Algorithm 13 Noise Clinic

```
Input : Noisy image $\tilde{u}_0$,
Input : Number of scales $N$.
Output : Denoised image $\hat{u}_0$.
Part 1 : Builds the image scale pyramid and records the difference images
for each scale $s = 1$ to $N-1$ do
    Let $\{\tilde{u}_{s-1}^k\}_{k\in[\![1,4^{s-1}]\!]}$ be the set of noisy subsampled images obtained at the previous scale. (For scale $s = 1$, it is $\tilde{u}_0$);
    for $k = 1$ to $4^{s-1}$ do
        Downsample $\tilde{u}_{s-1}^k$ into 4 sub-images : $\{\tilde{u}_s^{4(k-1)+i}\}_{i\in[\![1,4]\!]}$    ▷ Section 4.1, Algorithm 10
        Save difference images for this scale :
            $$\tilde{d}_{s-1}^k = \tilde{u}_{s-1}^k - \mathcal{U}\left(\{\tilde{u}_s^{4(k-1)+i}\}_{i\in[\![1,4]\!]}\right)^k.$$
    end for
    if $s = N-1$ then
        Set $\{\tilde{v}_{N-1}^k\}_k = \{\tilde{u}_{N-1}^k\}_k$
        Build the noisy mosaic image $\tilde{m}_{N-1}$ from $\{\tilde{v}_{N-1}^k\}_{k\in[\![1,4^{N-1}]\!]}$.    ▷ Algorithm 11
    end if
end for
Part 2 : Estimates noise and denoises bottom-up in the pyramid
for $s = N-1$ to $0$ do
    Estimate the noise covariance matrices $\{\mathcal{D}^t\mathbf{M_i}\mathcal{D}\}_\mathbf{i}$ on $\tilde{m}_s$.    ▷ Section 3, Algorithm 8
    Denoise $\tilde{m}_s$ with the modified NL-Bayes by using $\{\mathcal{D}^t\mathbf{M_i}\mathcal{D}\}_\mathbf{i}$ to obtain $\hat{m}_s$.    ▷ Section 2,
                                                                        ▷ Algorithms 3 and 4
    if $s > 0$ then
        for $k = 1$ to $4^{s-1}$ do
            Up-sample $\{\hat{u}_{s,4(k-1)+i}\}_{i\in[\![1,4]\!]}$    ▷ Algorithm 12
            Add the saved details $\tilde{d}_{s-1}^k$ to get $\tilde{v}_{s-1}^k$.
        end for
        Build the mosaic image of the next scale $\tilde{m}_{s-1}$ from $\{\tilde{v}_{s-1}^k\}$.    ▷ Algorithm 11
    else
        $\hat{u}_0 = \hat{u}_0^1$
    end if
end for
return $\hat{u}_0$.
```

4.2.4 Influence of the Thresholds for the Homogeneous Areas, k_L and k_H

Those parameters are quite crucial. If they are taken too small, the homogeneous area algorithm will be inactive, and much noise will be left on homogeneous areas. If they are chosen too large, a lot of detail will be removed. The homogeneity criterion must be applied at all scales, but for lower scales this criterion must become far more conservative. Indeed, despite the fact that we got the same amount of pixels at each scale, the noise estimator is not as accurate as at the finest scale because sub-images are too small, and large details may be seen as noise which is removed with the homogeneity criterion.

Determining optimal values for these parameters is hard, because they depend on the accuracy of the noise estimation, and directly on the image. Our choice will be to set conservative values to these parameters, simply securing that the criterion is visually helpful for a large proportion of

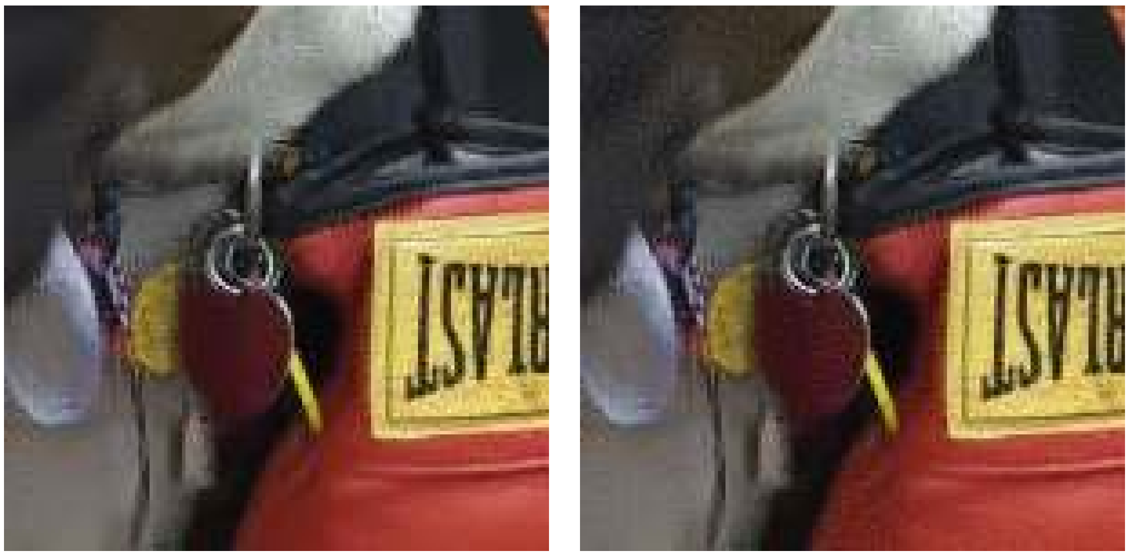


Figure 8: From left to right: Blind denoising with and without the random selection of similar patches on a detail of “Dog” image.

images, but that it never degrades images by causing excessive blur. Therefore, the retained values are

	Lower scales	Finest scale
k_L	2.0	4.0
k_H	4.0	8.0

The importance of setting conservative values and the crucial importance of these two parameters are illustrated in Figure 9. One can observe that the sky is well denoised with $(k_L = 4, k_H = 8)$ for the finest scale, and that the details of the trees and the houses are preserved, whereas if (k_L, k_H) are taken too large, details start disappearing, without getting any improvement on the sky.

To improve the detection of homogeneous areas, the criterion based on the difference between the mean and the barycenter of the 3D group helps slightly (see Algorithm 7), especially on borders, as shown in Figure 10. One can see that this improvement does not affect the sky, but helps preserve tiny details on the trees and the houses.

4.2.5 Influence of the Number of Scales N

Theoretically any number of scales can be chosen. At a very coarse scale the noise should be null and estimated as such, so that no denoising would eventually occur. However, if too many scales are chosen, one may observe some detail loss and color attenuation with the Noise Clinic. At the first scale this sort of detail is small and hardly distinguishable from noise; it will be hopelessly removed by the denoising. Yet, for lower scales, some huge details may disappear that are clearly not noise. There are two reasons at least for this phenomenon: As more scales are used, even if all sub-images are used to estimate the noise, the noise estimation accuracy gets harder for the lowest scales. Indeed when sub-images are too small, there are no more homogeneous areas left in them to estimate the noise. Furthermore, in such small images some signal may be confused with noise, thus causing a noise overestimation.

Thus, it is almost always better to use a minimal number of scales, in most cases not more than three: as JPEG compression works on 8×8 patches, after three scales the influence of the JPEG

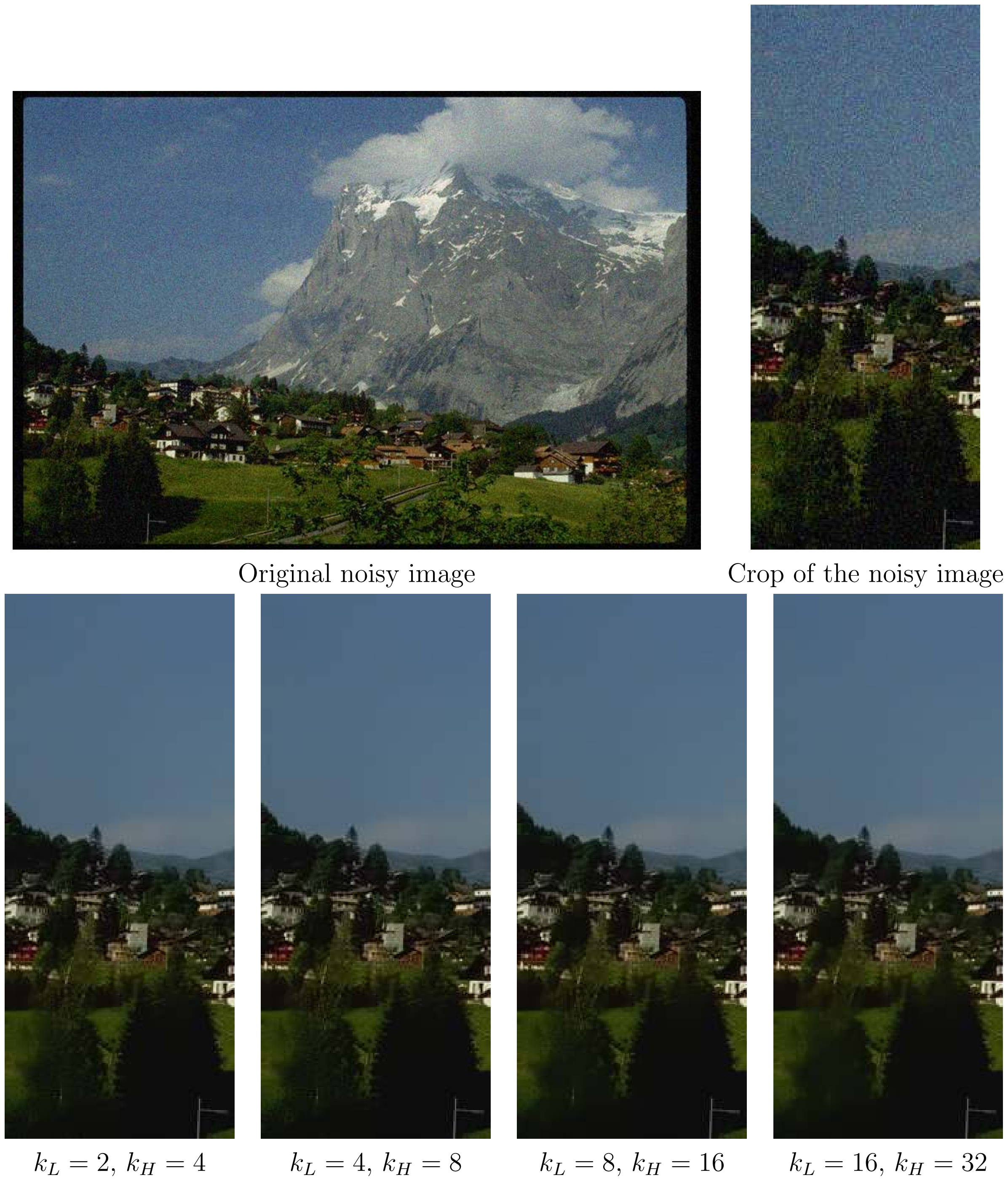


Figure 9: Blind denoising with increasing k_L, k_H parameters for the finest scale on "Postcard" image, with $(k_L = 2, k_H = 4)$ for lower scales.

compression is no more visible. However, for some images that have undergone a zoom in, threshold transforms, or for scans of old photographs, this rule is no more valid. We found that for some images with very low frequency noise it was sometimes better to use up to five scales. From that point of view our "blind denoising" is not fully blind and requires an user's evaluation on the number of scales involved. For all these reasons, a conservative default value is set to $N = 2$ on the web-demo, but the possibility to change this parameter is let to the user. We found that the value $N = 2$ works for a strong majority of images of unknown origin.

Illustrations of the use of the "right" number of scales are presented in Figures 11, 12 and 13.

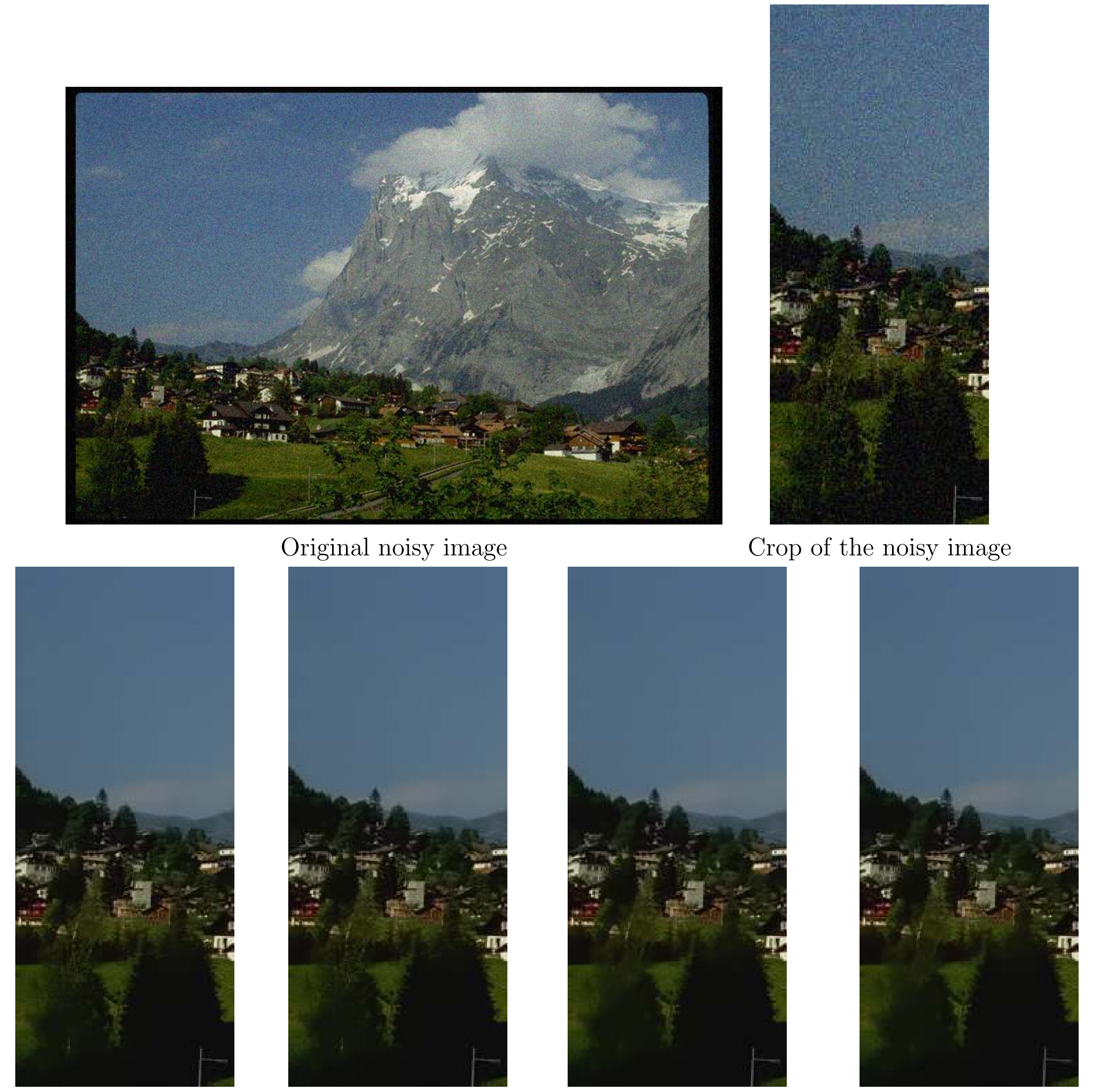

Original noisy image
Crop of the noisy image
$k_L = 2,\ k_H = 4,\ \gamma = 1$
$k_L = 4,\ k_H = 8,\ \gamma = \infty$
$k_L = 8,\ k_H = 16,\ \gamma = 1$
$k_L = 16,\ k_H = 32,\ \gamma = \infty$

Figure 10: Blind denoising with increasing k_L, k_H parameters for the finest scale and with $(\gamma = 1)$ and without $(\gamma = \infty)$ the improved homogeneous criteria based on the difference between the mean and the barycenter of the 3D group on "Postcard" image, with $(k_L = 2, k_H = 4)$ for lower scales.

For the "Bears" image in Figure 11 using only two scales gives a very nice result by eliminating the noise whereas tiny details such as the fur of the bear or wavelet on water remains. The corresponding difference image shows that only noise has been removed since one cannot distinguish any structure on it. By using four scales these tiny details are blurred out, since they may appear as noise at lower scales. These details therefore pop up in the corresponding difference image.

For the "Girls" image in Figure 12, which presents a strong low-frequency noise, using two scales

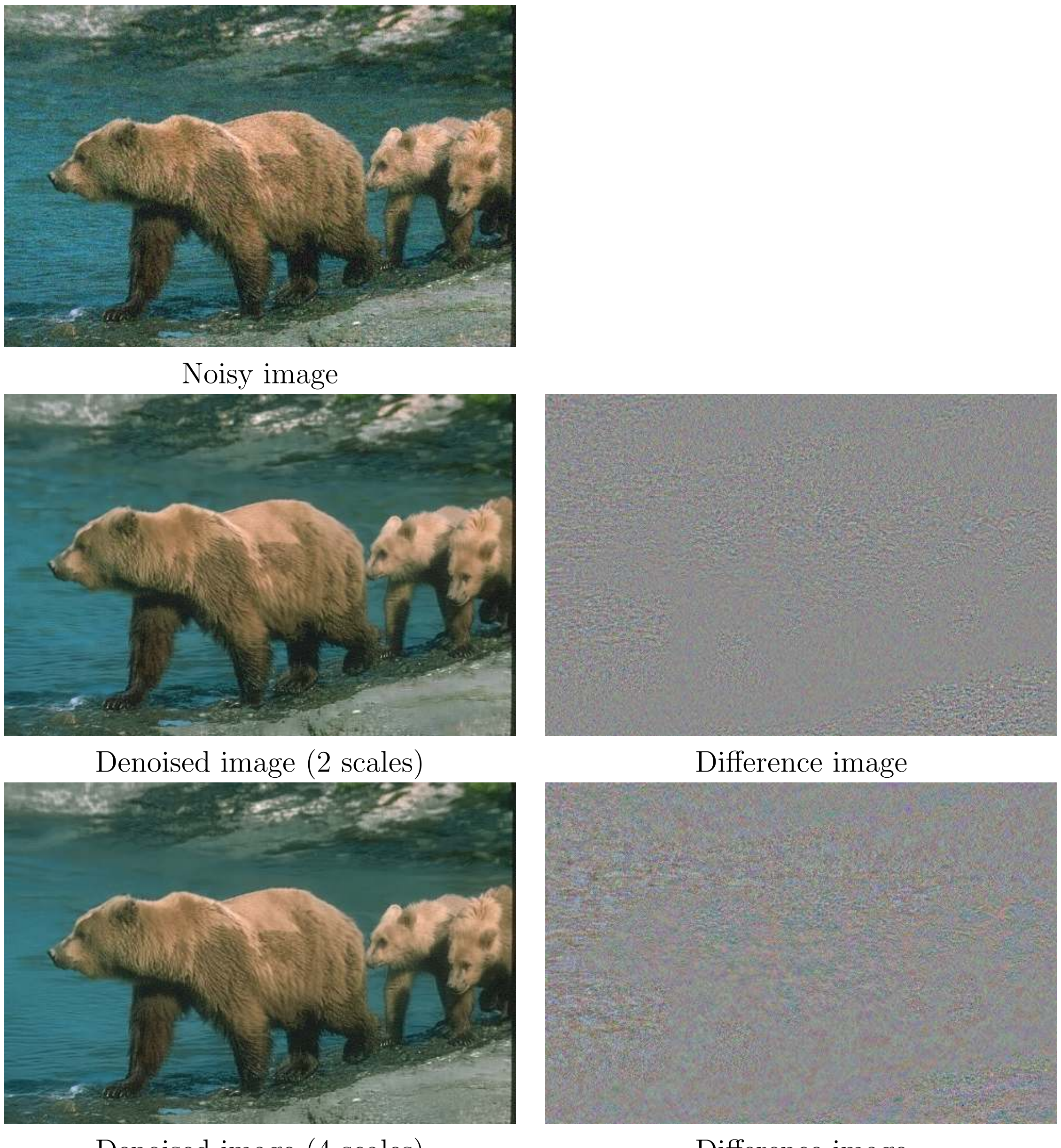


Figure 11: Blind denoising when varying the number of scales on the “Bears” image.

is not enough. Going up to three scales provides a better result removing correctly low-frequency noise without destroying tiny details such as hairs. Notice the low frequency of the noise observable in the difference image for the three scales result.

For the “Postcard” image in Figure 13, one can observe how tiny details remain when the number of scales increases, while some large details such as the clouds are removed if too many scales are being used. These details are seen as noise by the algorithm, whereas they really are structure. If only two scales are used, a slight low-frequency noise still remains. All results seen independently are equally agreeable, so the final choice must be left to the user.

4.2.6 Influence of the Multiplicative Factor γ_f

Some underestimation of the noise can be observed on old photographs. For such images, the web-demo allows the user to increase (or decrease) γ_f, which directly multiplies the noise curve at each

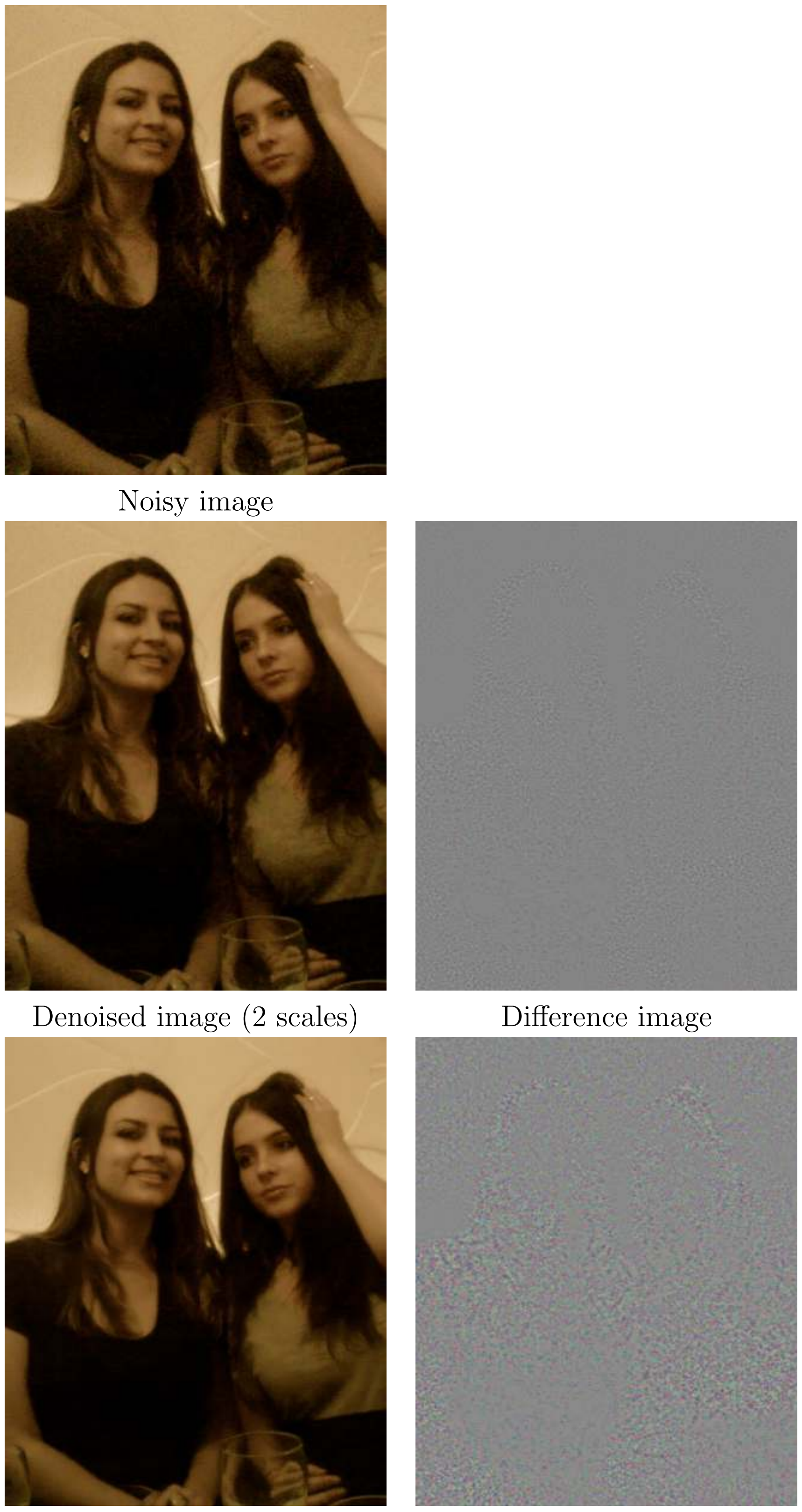


Figure 12: Blind denoising when varying the number of scales on the "Girls" image.

step and each scale by this coefficient. One can tune the denoising result with this parameter. Yet for most images, the default value of $\gamma_f = 1$ is valid. Figure 14 shows an example of the influence of

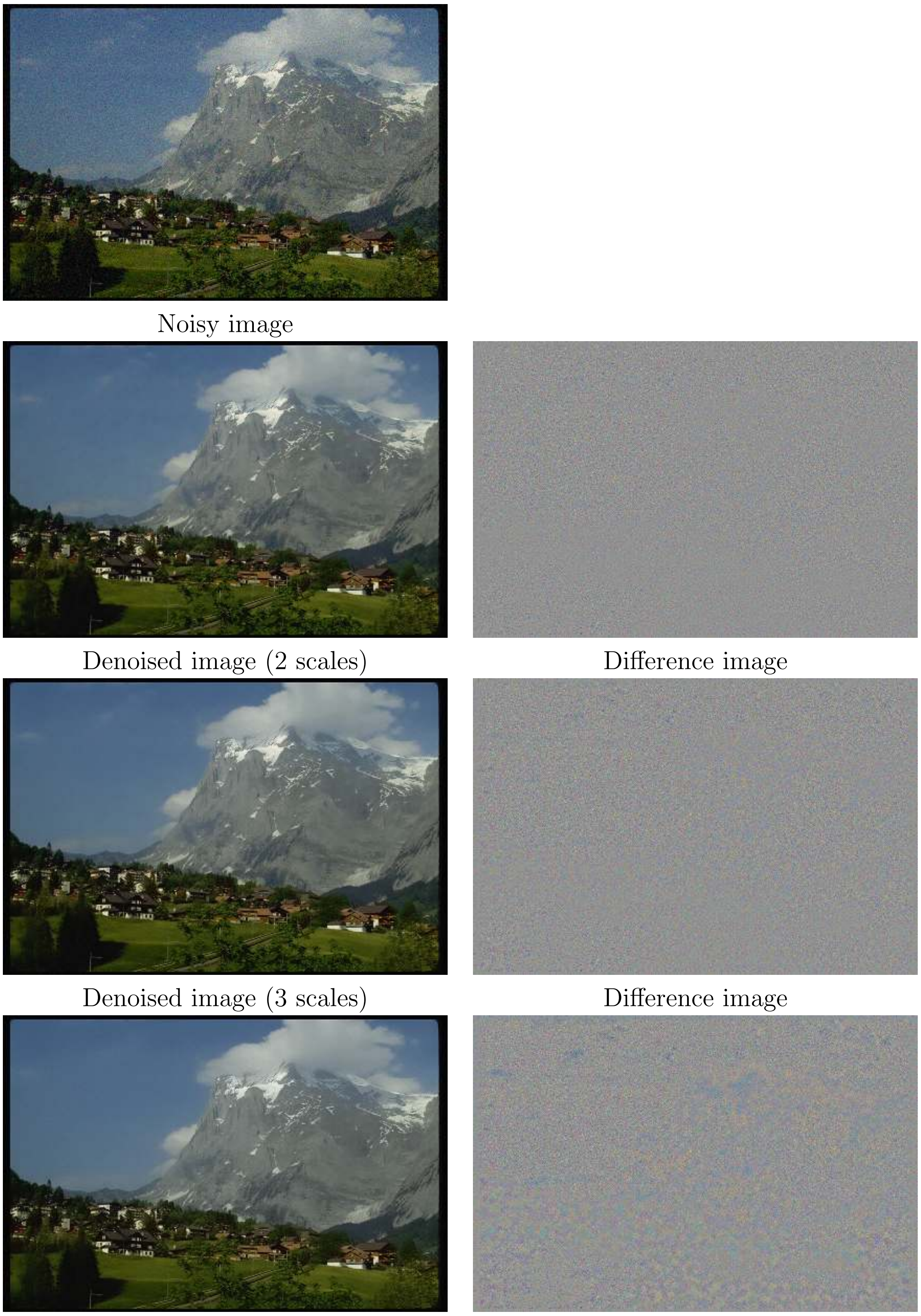
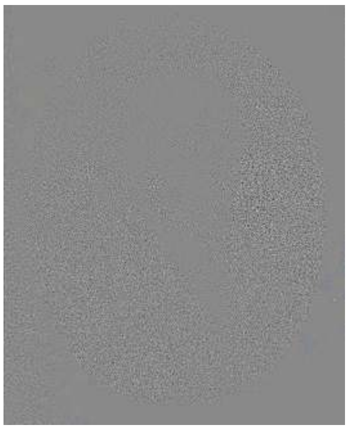

Noisy image
Denoised image (2 scales)
Difference image
Denoised image (3 scales)
Difference image
Denoised image (4 scales)
Difference image

Figure 13: Blind denoising when varying the number of scales on the "Postcard" image.

the multiplicative factor.

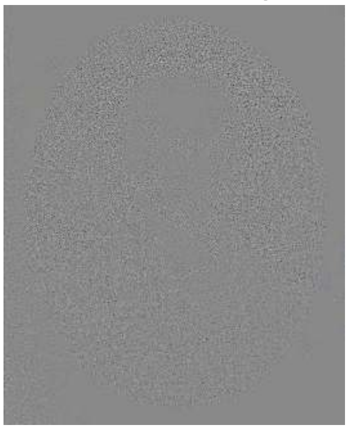
Noisy image

Denoised image (3 scales, $\gamma_f = 1.0$)

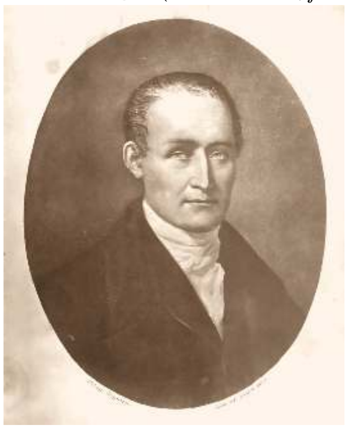


Difference image

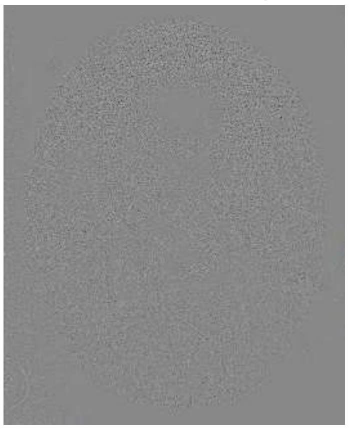
Denoised image (3 scales, $\gamma_f = 1.5$)



Difference image



Denoised image (3 scales, $\gamma_f = 2.0$)



Difference image

Figure 14: Blind denoising when varying the multiplicative factor on the “Dujardin” image.

The default parameter values are summarized in Table 8.

Parameter	Description	Default value
κ	Size of patches	4
η	Size of the search window	$\kappa^2 - 1$
N	Number of scales	2
k_L	Lower threshold for homogeneous area	2 (lower scale) - 4 (finest scale)
k_H	Higher threshold for homogeneous area	4 (lower scale) - 8(finest scale)
γ	Threshold to separate homogeneous area from edges	1.0
N_{min}	Minimal number of similar patches	$2\kappa^2$
N_{max}	Maximal number of similar patches	$\frac{\eta^2}{2}$
γ_f	Multiplicative coefficient of the estimated noise	1.0
τ	Similarity threshold	3.0
α_Y	Coefficient for weighting distance on the first step	$\sqrt{\frac{1}{2}}$
α_U	Coefficient for weighting distance on the first step	$\sqrt{\frac{1}{4}}$
α_V	Coefficient for weighting distance on the first step	$\sqrt{\frac{1}{4}}$
N_f	Number of iterations for the noise estimation filtering	5
S_f	Size of the noise estimation filtering	10

Table 8: Default parameter values of the Noise Clinic algorithm.

5 Results

5.1 Estimated Noise Curves and Illustrative Denoising Results

The Noise Clinic was applied to real noisy images for which no noise model was available. For each experiment we shall present the noisy input image and for each scale:

- the noisy image where noise has already been removed at coarser scales;
- the denoised image at this scale;
- the difference image = noisy - denoised at this scale;
- the average noise curve over high frequencies;
- the average noise curve over low frequencies.

For each scale larger than 1, the subsampled images are up-sampled to keep the original image size. Similarly, the noisy image shown at each scale is the sum of the upsampled version of the denoised sub-images of the previous scale and of the still noisy difference image kept in reserve. In other terms this image contains the remaining noise at the current scale; the noise at coarser scales has in principle already been removed. Visual results are shown in figures 15, 17, and 19. The corresponding noise curves are presented in Figures 16, 18, and 20. These experiments confirm that noise in JPEG images is not white, being signal and frequency dependent. Indeed, the noise curves are not flat, and they are not divided by two from a scale to the next, as they should if the noise were white.

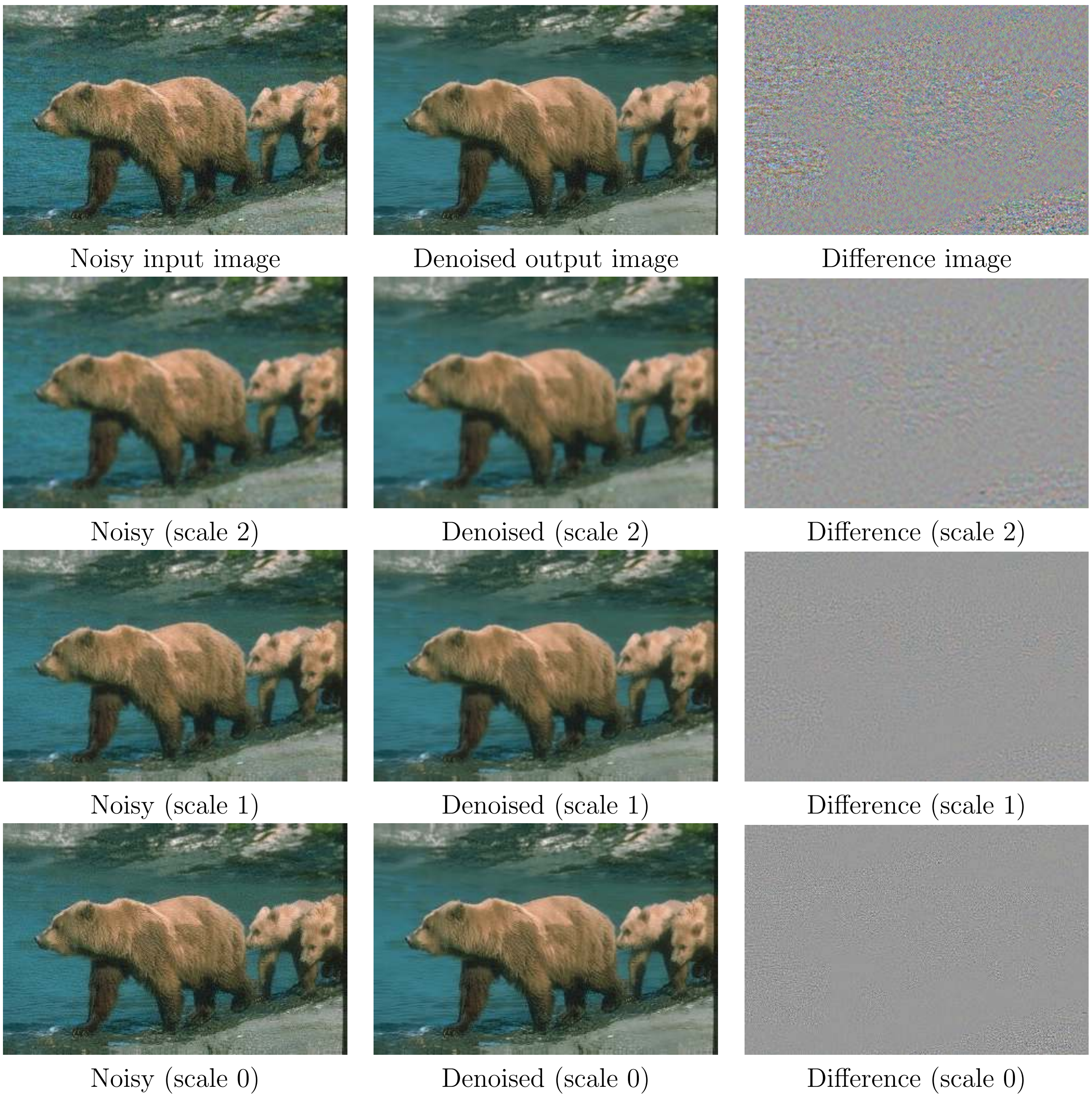
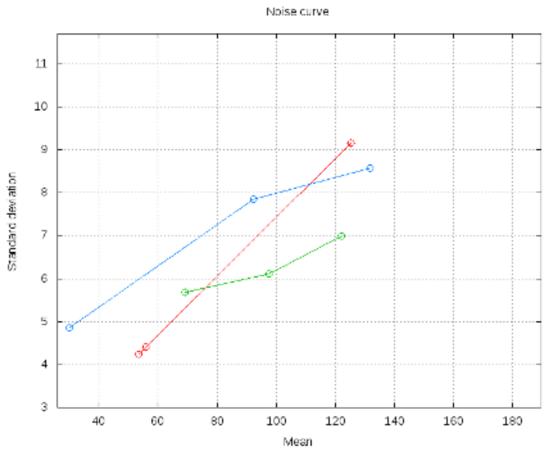
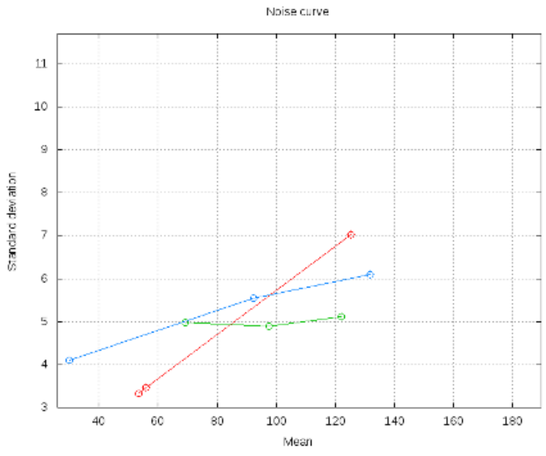


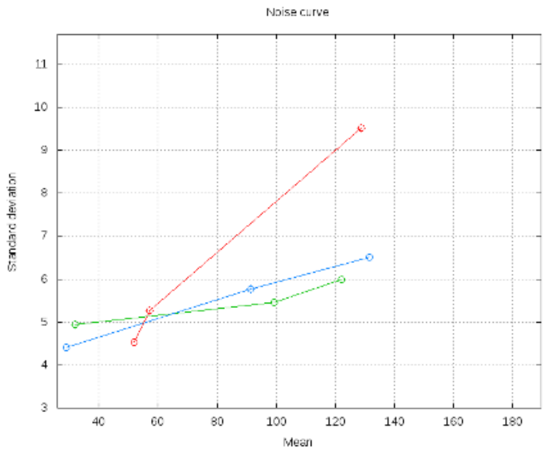
Figure 15: Results of the Noise Clinic on the “Bears” image.



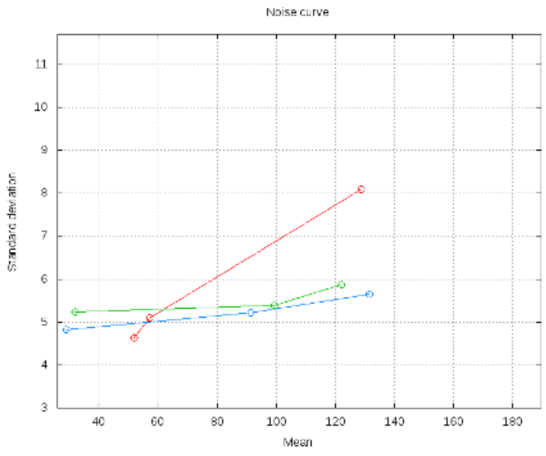
Low freq. av. curve (s. 2)



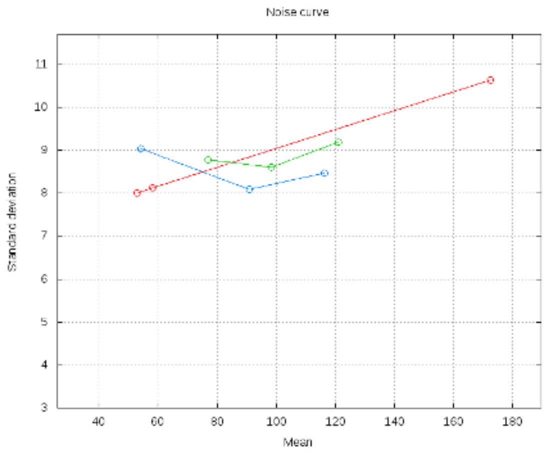
High freq. av. curve (s. 2)



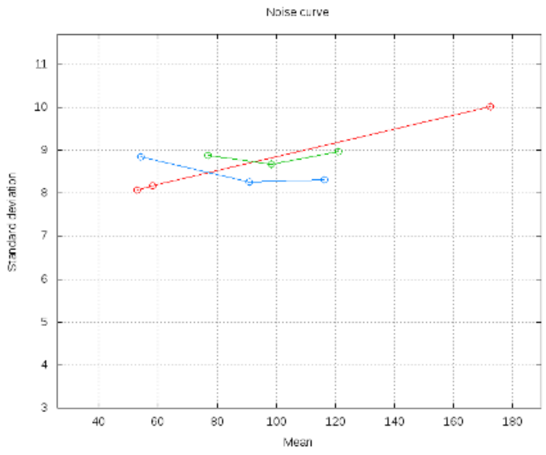
Low freq. av. curve (s. 1)



High freq. av. curve (s. 1)



Low freq. av. curve (s. 0)



High freq. av. curve (s. 0)

Figure 16: Results of the noise estimation on the "Bears" image. The noise in this image is clearly colored: it increases with descending octaves instead of being divided by two, as it should if it were white.

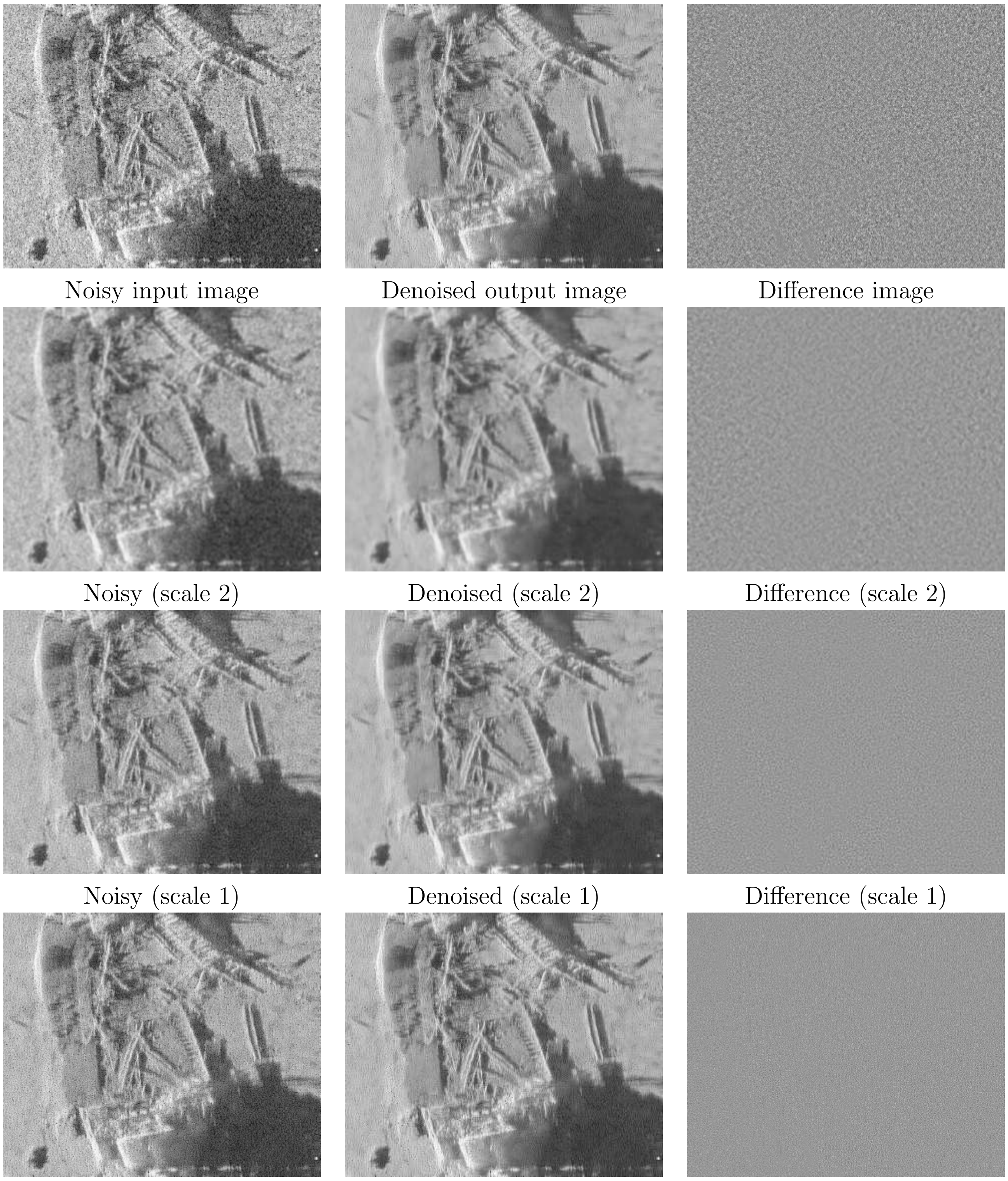
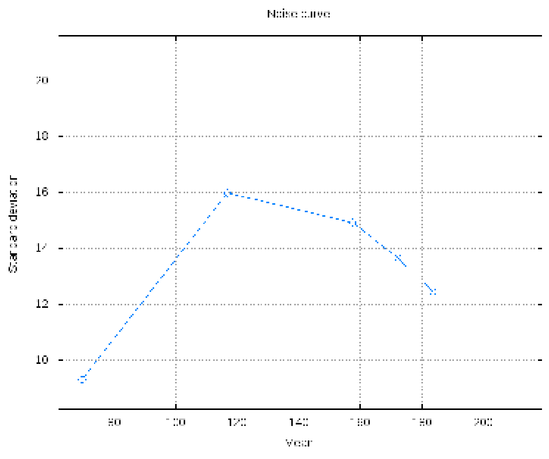
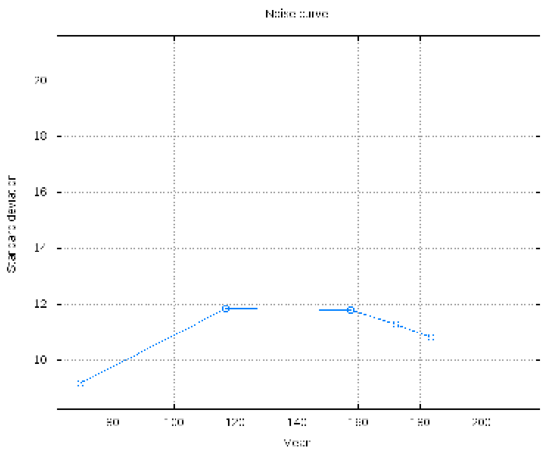


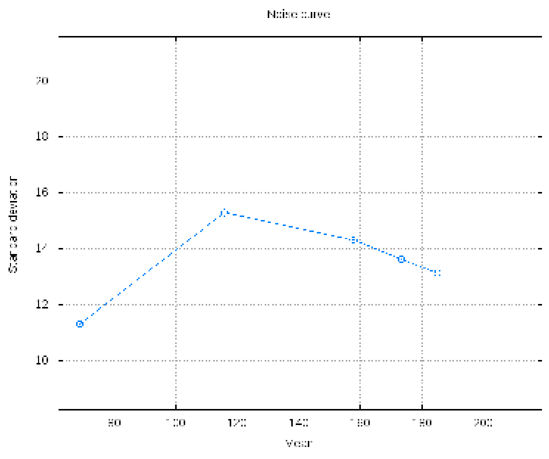
Figure 17: Results of the Noise Clinic on the “Ship” image.



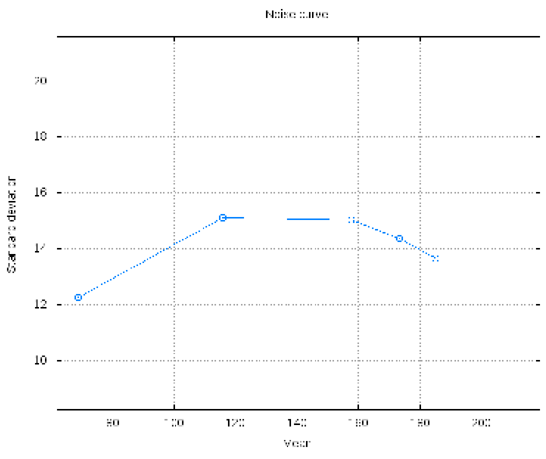
Low freq. av. curve (s. 2)



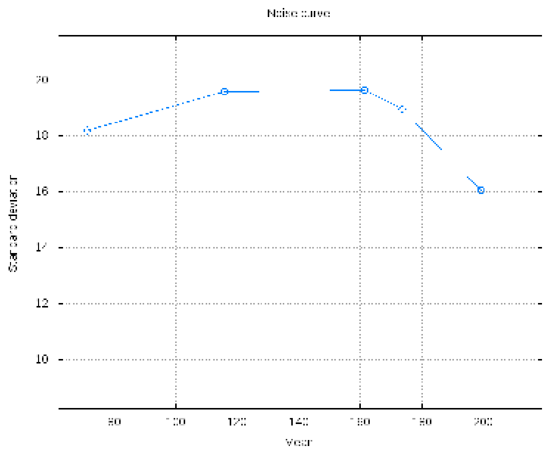
High freq. av. curve (s. 2)



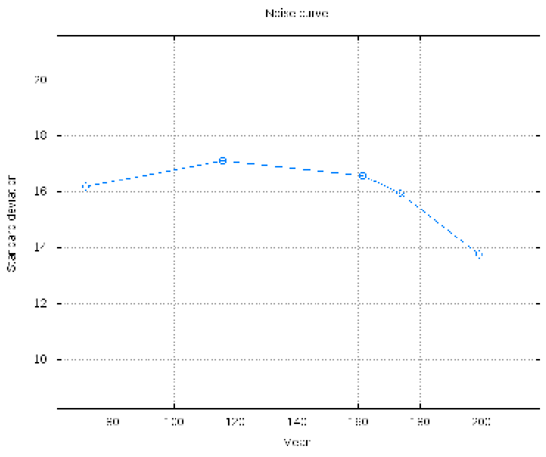
Low freq. av. curve (s. 1)



High freq. av. curve (s. 1)

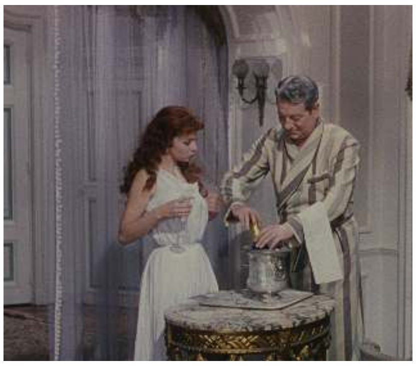


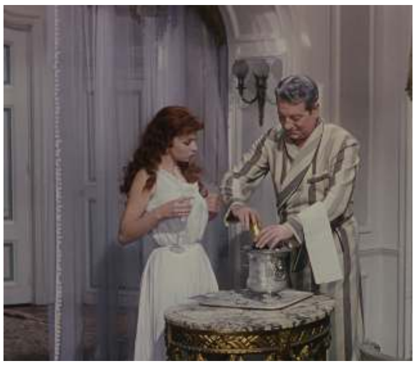
Low freq. av. curve (s. 0)

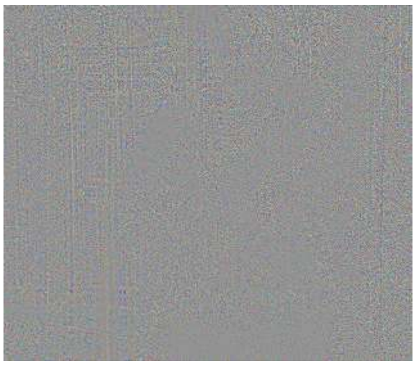


High freq. av. curve (s. 0)

Figure 18: Results of the noise estimation on the “Ship” image. The noise in this image is clearly not white: it has almost the same value whatever the scale, instead of being divided by two at each scale.

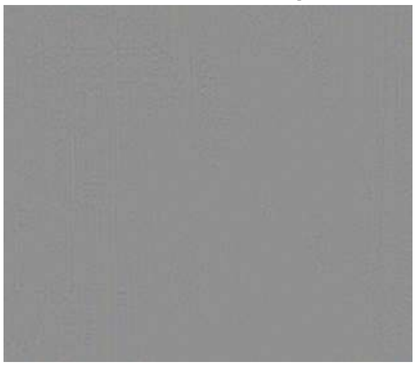

Noisy input image


Denoised output image


Difference image

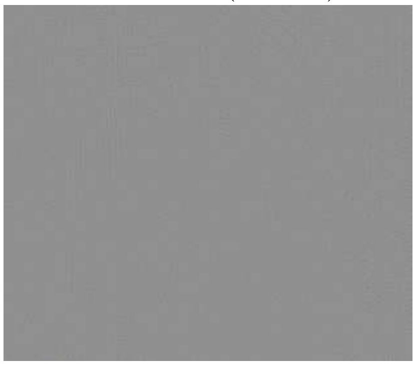
Noisy (scale 2)

Denoised (scale 2)

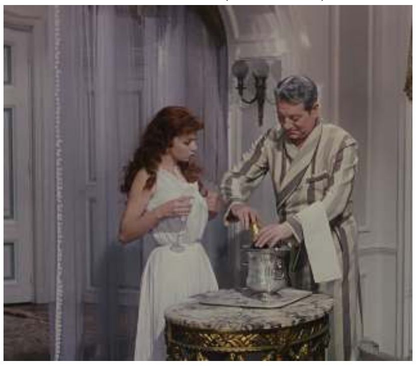

Difference (scale 2)

Noisy (scale 1)

Denoised (scale 1)


Difference (scale 1)

Noisy (scale 0)


Denoised (scale 0)

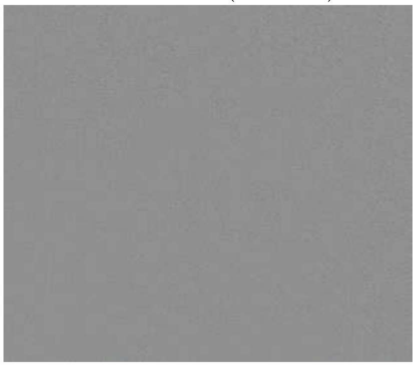
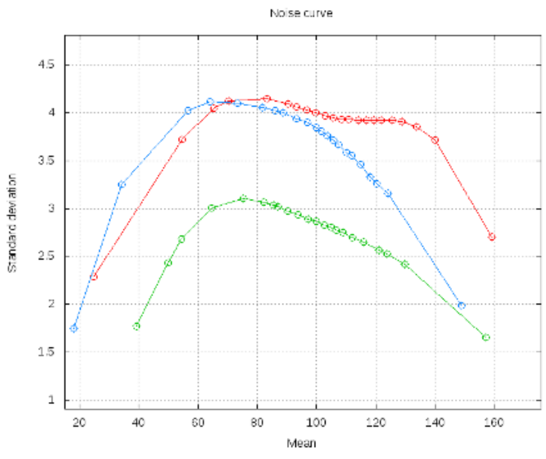
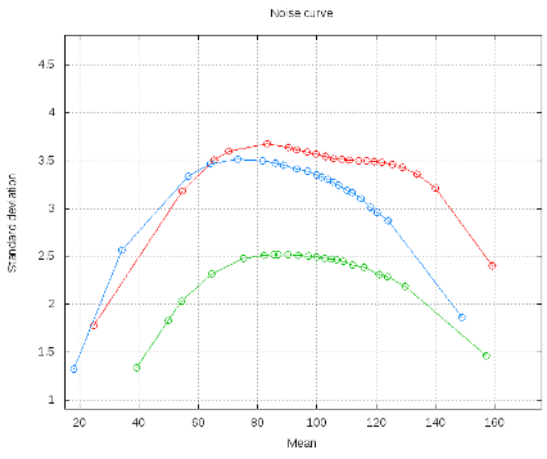

Difference (scale 0)

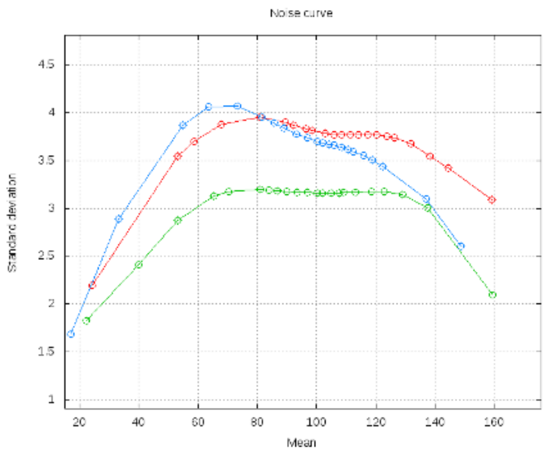
Figure 19: Blind denoising of the “Movie2” image.



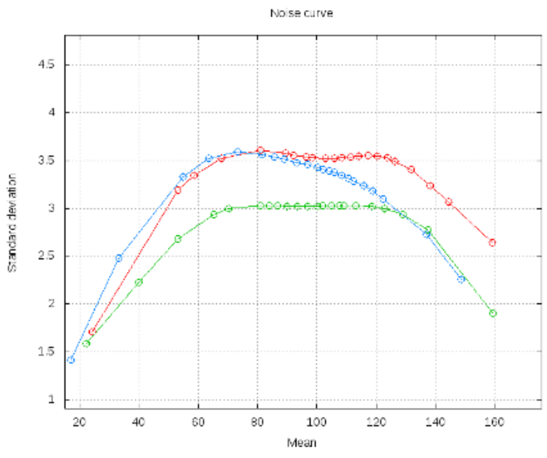
Low freq. av. curve (s. 2)



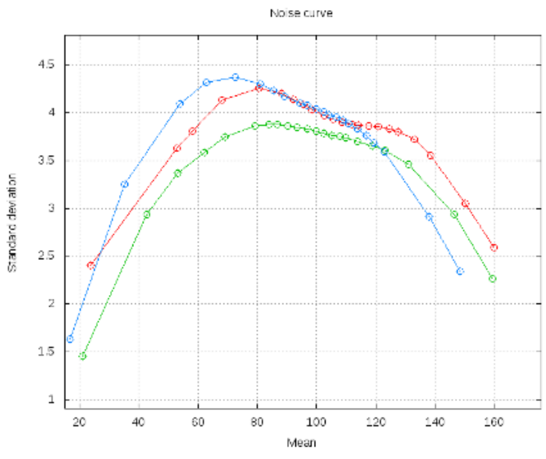
High freq. av. curve (s. 2)



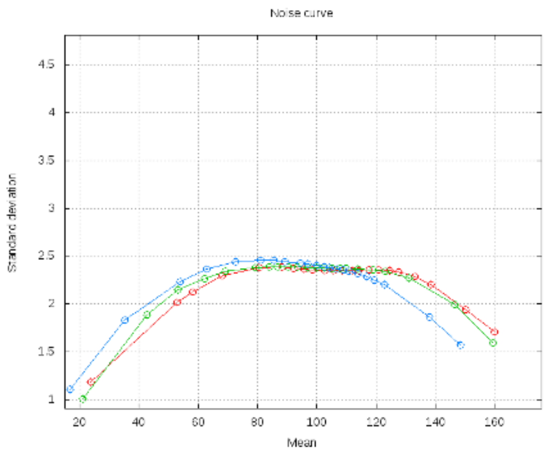
Low freq. av. curve (s. 1)



High freq. av. curve (s. 1)



Low freq. av. curve (s. 0)



High freq. av. curve (s. 0)

Figure 20: Noise estimation on the “Movie2” image, with typical concave shapes for the noise curves probably caused by the gamma-correction applied previous to compression.

5.2 Result on a Typical Low-light JPEG Image Taken From a Cell Phone

The images shown in Figure 21 were taken in a bar with low light conditions, so the initial SNR was low. One can observe big colored spots caused by the demosaicking that are attenuated by the Noise Clinic. Yet, some structured JPEG artifacts are left behing, being self-similar.

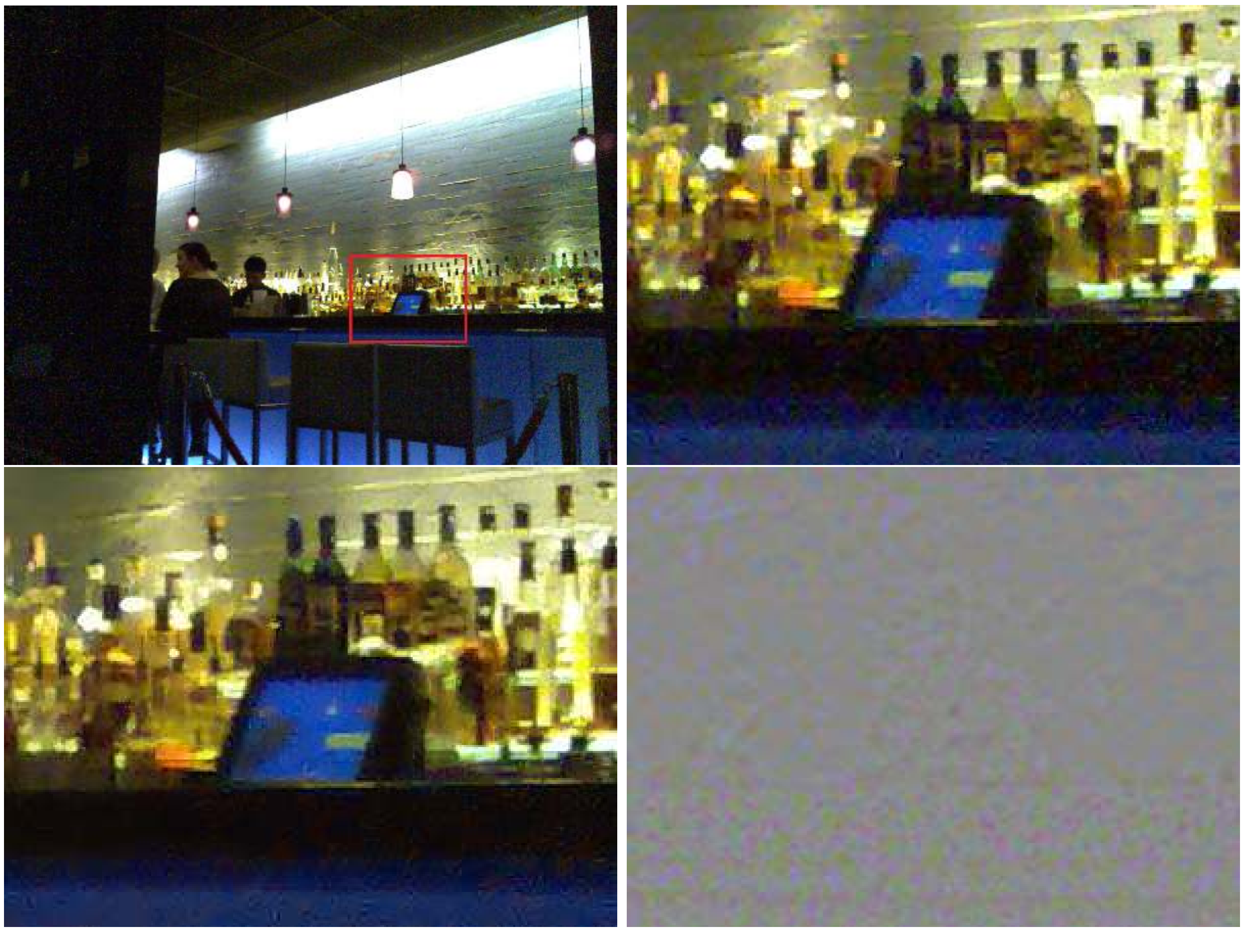


Figure 21: Blind denoising on “Bar”, using three scales. From left to right, top to bottom : input noisy image, crop of the noisy image, crop of the output denoised image, crop of the difference image.

5.3 Results on Old Photographs

Figures 22, 23, and 24 show results obtained by the Noise Clinic on scans of old noisy photographs, the *1969 Apollo*[3] image and the 1998 *Kleiner*[4] image (it is advised to zoom in strongly on details).

[3]This file is in the public domain because it was solely created by NASA. NASA copyright policy states that “NASA material is not protected by copyright unless noted”. http://dayton.hq.nasa.gov/IMAGES/LARGE/GPN-2000-001849.jpg

[4]Image licensed under the Creative Commons Attribution-Share Alike 3.0 Unported license, taken by Wikimedia Commons user Olaf1541, http://commons.wikimedia.org/wiki/File:Kleiner_hecht.png

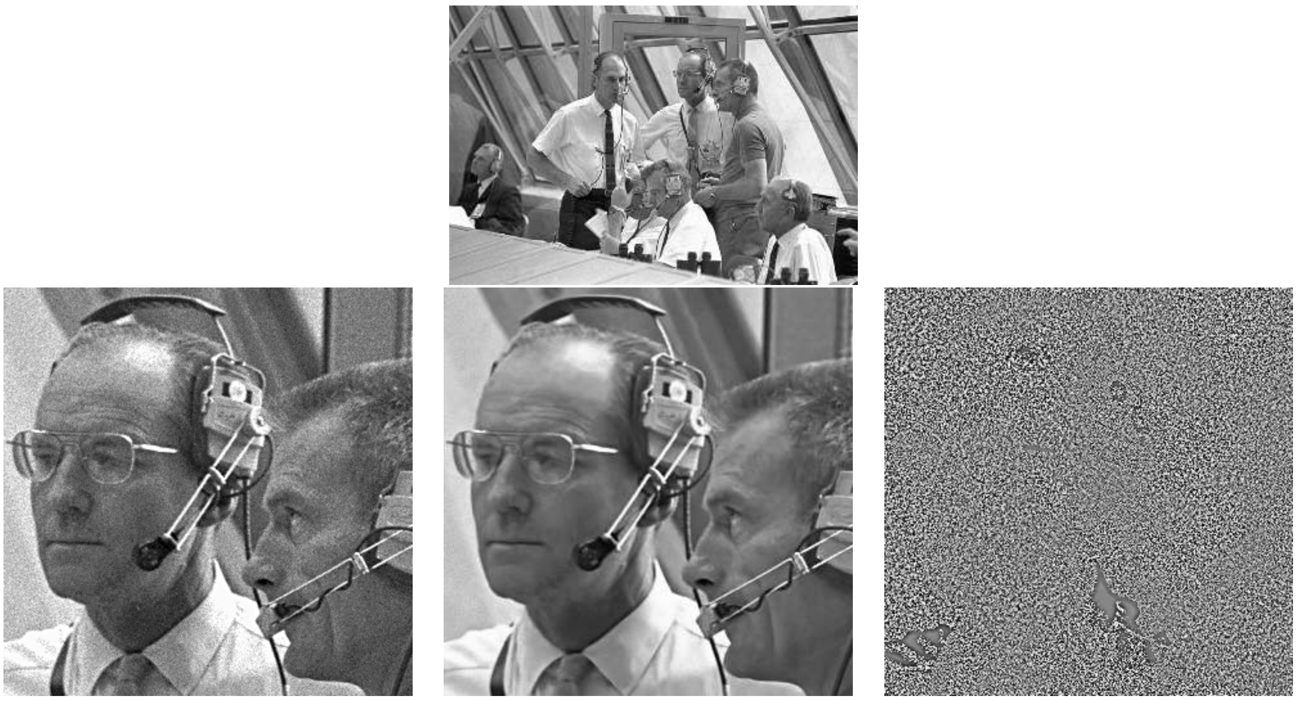


Figure 22: Top: original noisy image “Apollo”. Bottom: detailed result of the Noise Clinic by using 2 scales.

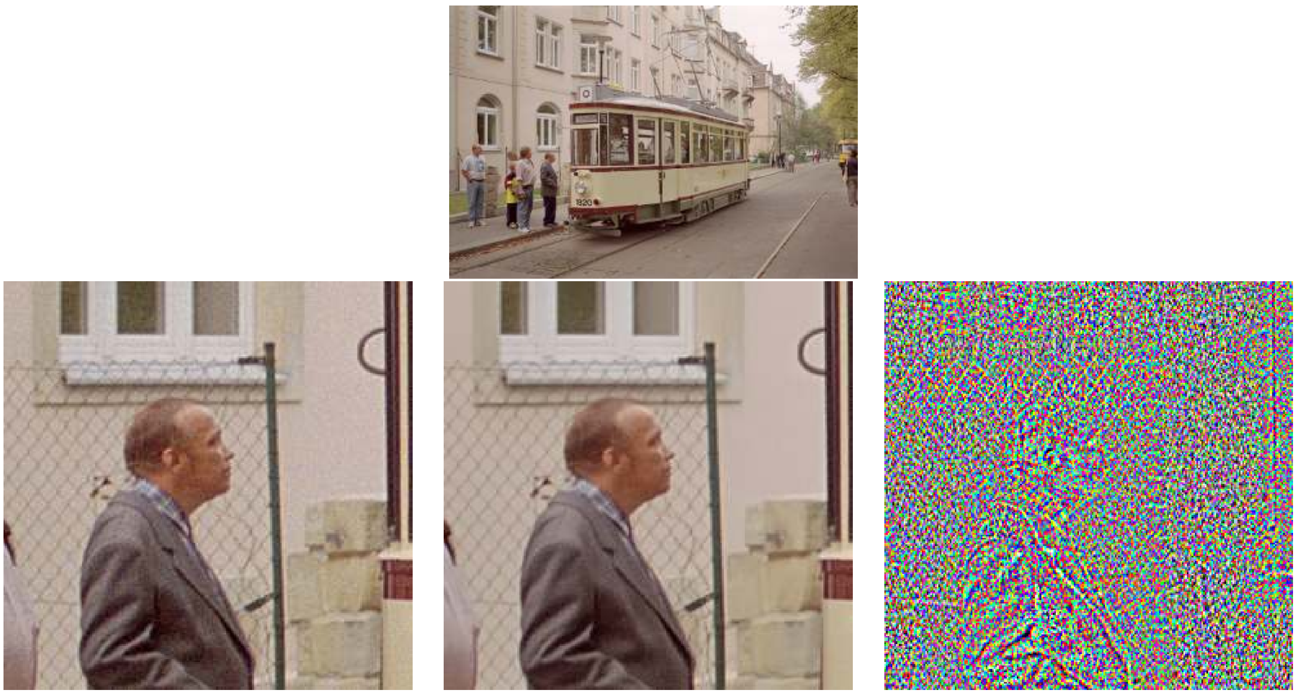
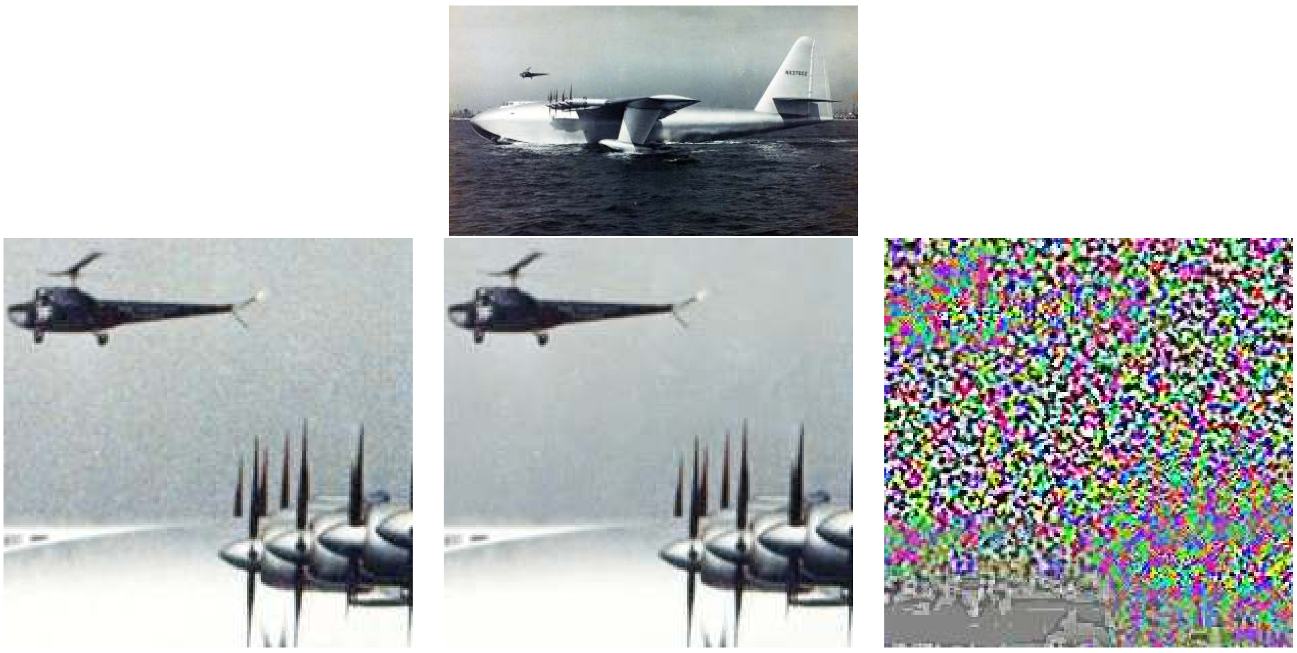


Figure 23: Top: original noisy image “Kleiner”. Bottom: detailed result of the Noise Clinic by using 2 scales.



Noisy image Denoised image Difference image

Figure 24: Top: original noisy image “Plane”. Bottom: detailed result of the Noise Clinic by using 2 scales.

5.4 Comparison to Portilla’s Blind BLS-GSM

Blind BLS-GSM introduced in [20] and [21] is a state-of-the-art blind denoising algorithm. It is multiscale and models wavelet coefficient patches at each scale, making a global Bayesian estimation of them as a Gaussian mixture.

One can observe in Figure 25 how remarkably BLS-GSM deals with strongly structured noise, namely noise that is sparse in the Fourier domain. The Noise Clinic instead keeps it and even re-enforces it.

In the images of Figure 26 with a less singular noise, the Noise Clinic performs instead better. This is due to the fact that the Noise Clinic is inherently local and therefore probably makes a better local distinction between noise and signal.

5.5 Comparison to One of the Best Commercial Software for JPEG Denoising: Neat Image

We end this experimental section with a comparison to the celebrated commercial software *Neat Image*[5]. Of course, as it is a commercial software, we cannot characterize the method underlying this software, so the comparison must remain purely qualitative. We used the free demo of this software with both “luminance” and “chrominance” denoising cursor put to 100%. For some images, Neat Image failed to estimate correctly the noise, so we used the manual tools for homogeneous area selection in order to produce the best visual result. It is important to notice here that this software is not fully automatic, contrarily to the Noise Clinic.

For each produced images, the difference between the denoised result and the noisy input is given, to help the reader detect the denoising artifacts. The difference should contain only noise, and no image detail. These results are presented in Figures 27, 28, 29, 30, 31, 32.

[5]Free demo available at `http://www.neatimage.com/`

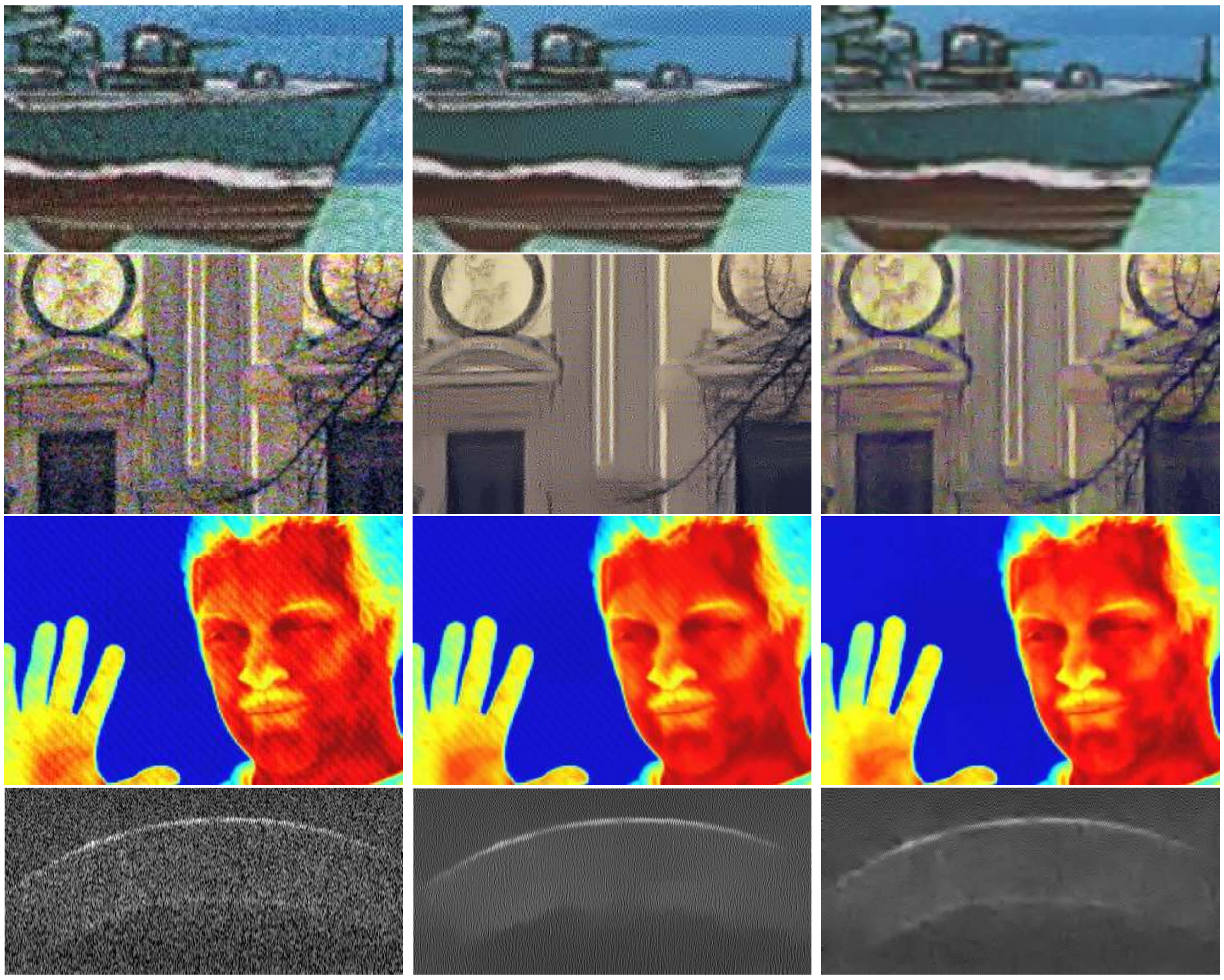


Figure 25: Results of our blind denoising and of Blind BLS-GSM on several images from [20]. From left to right: Noisy image, result of the Noise Clinic, result of the Blind BLS-GSM algorithm. It is advised to zoom in by a 300% factor the digital document to examine details.

One can observe that *Neat Image* produces generally a smoother version than the Noise Clinic, with well flattened homogeneous areas, but to the cost of (sometimes) huge loss of image detail. On the contrary as one can easily observe on the difference images, the Noise Clinic rarely removes details, and it gives more “natural” looking results. It also appears that for small images, *Neat Image* fails to estimate correctly the noise, and generally produces an overestimation. Of course, the software gives a cursor for both luminance and chrominance denoising, to allow the user to manually adjust the denoising power. In the same spirit, the user may adjust the window for the noise estimation, to center it precisely on an homogeneous area. For a fair comparison, we used the automatic tools and cursor put to 100%. The Noise Clinic itself may be tuned too, by adjusting the thresholds of homogeneous area (k_L, k_H) and the multiplicative factor γ_f.

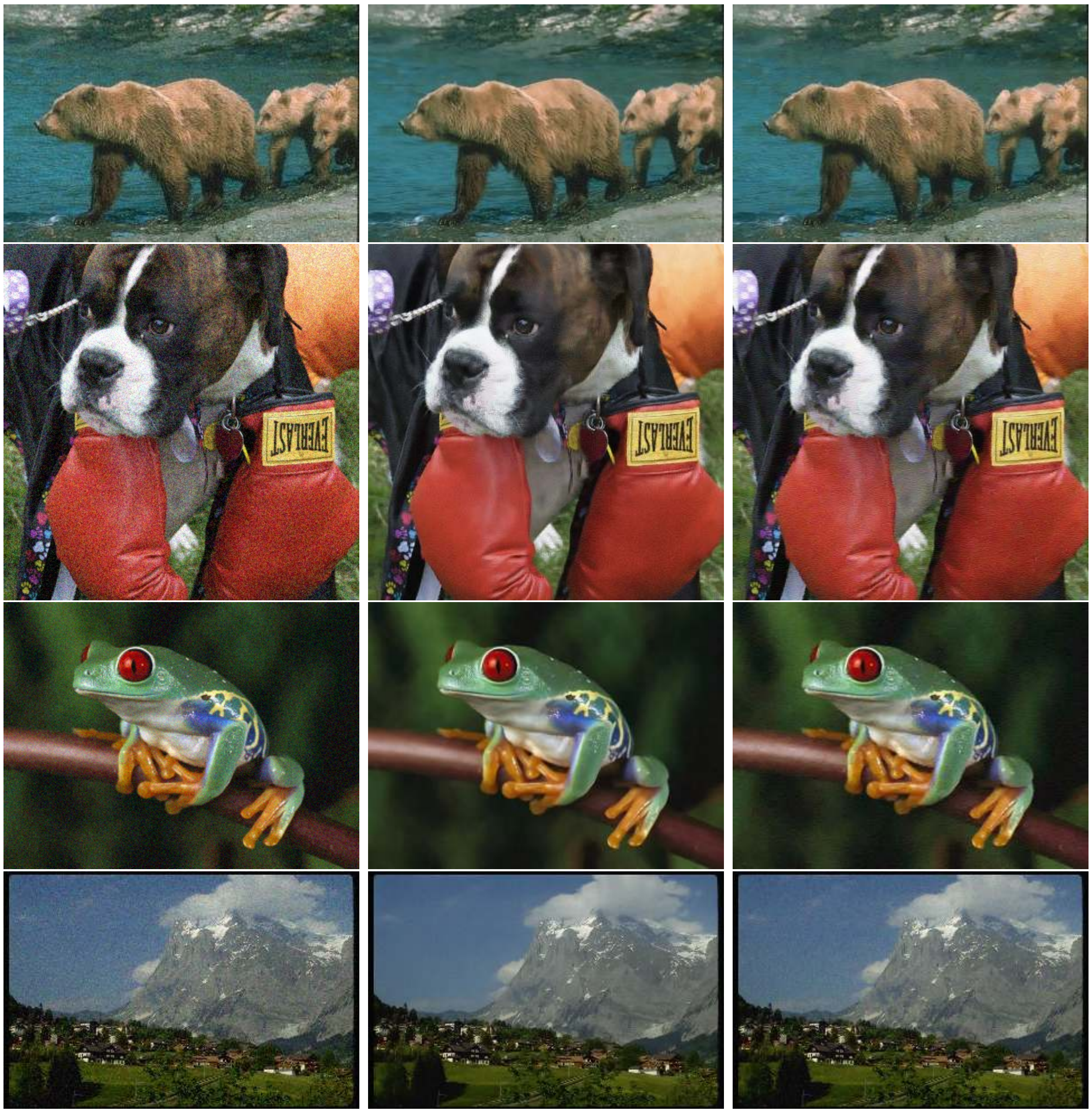
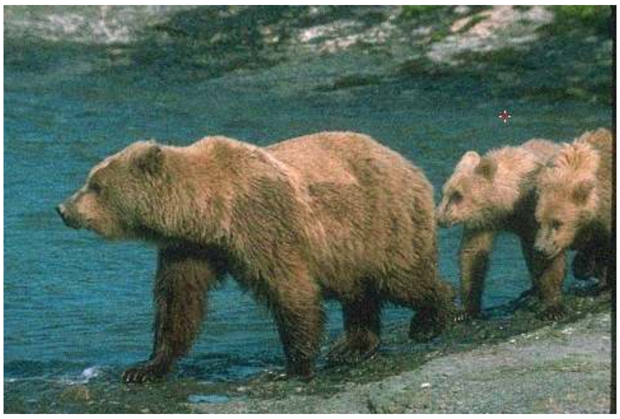
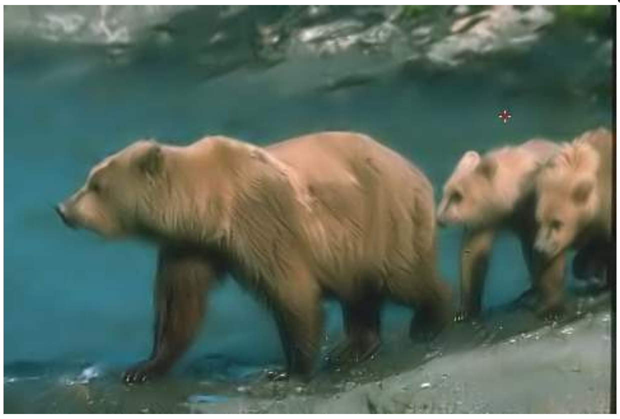
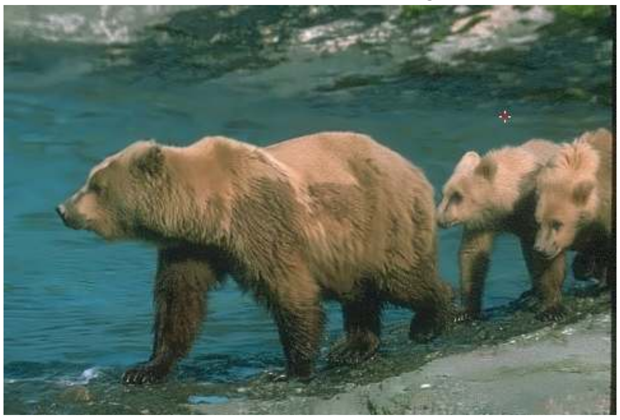


Figure 26: Comparing our blind denoising with Blind BLS-GSM on several images. It is advised to zoom in by a 400% factor the digital document to examine details. From left to right: Noisy image, result of the Noise Clinic, result of the Blind BLS-GSM algorithm.


Noisy input


Result of *Neat Image*

Difference with the noisy input


Result of the Noise Clinic using two scales

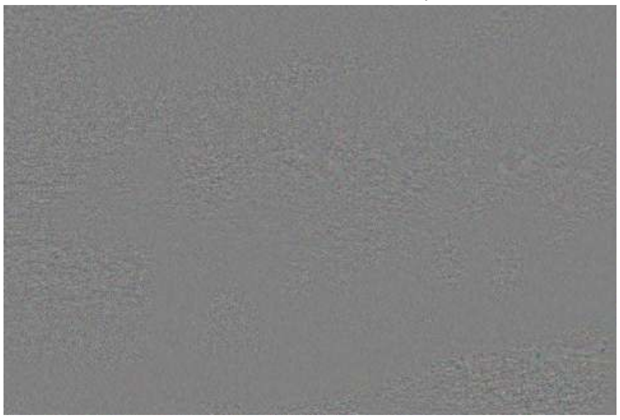
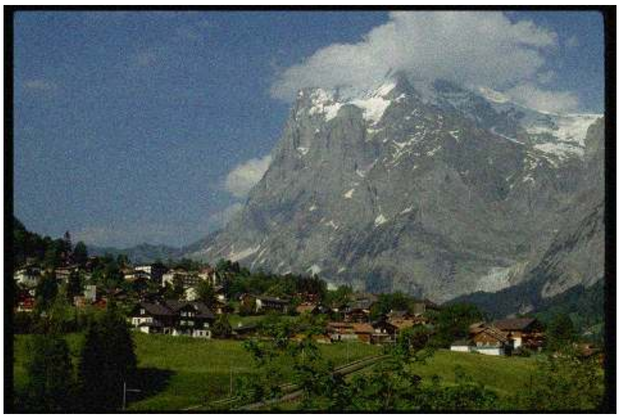
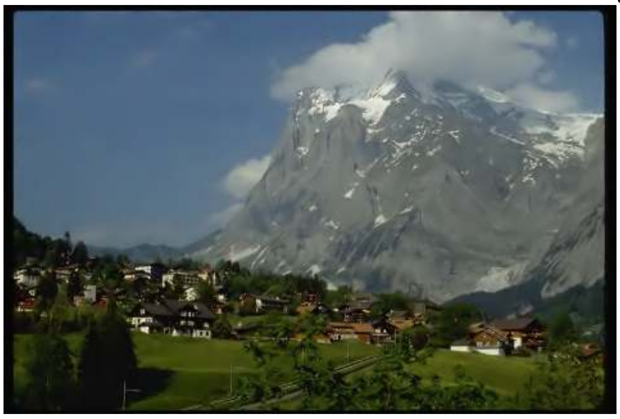

Difference with the noisy input

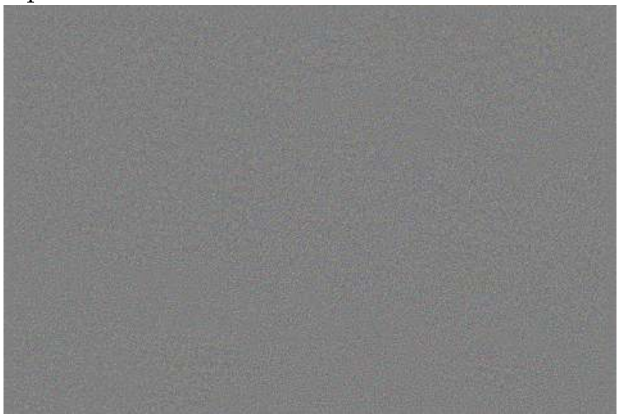
Figure 27: Comparison between the commercial software *Neat Image* and the Noise Clinic on the "Bears" image.



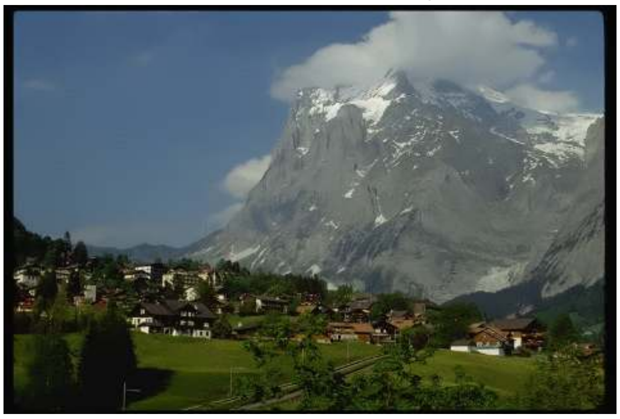
Noisy input



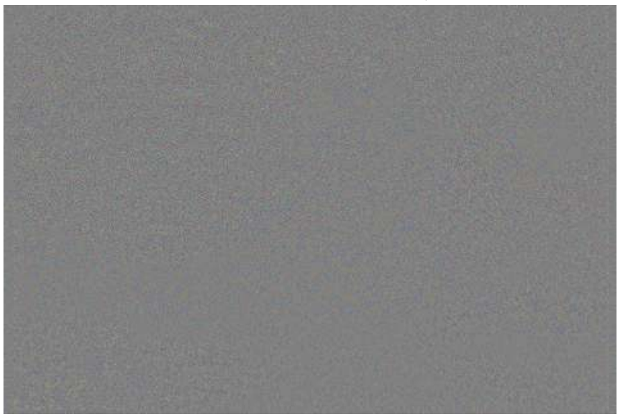
Result of *Neat Image*



Difference with the noisy input

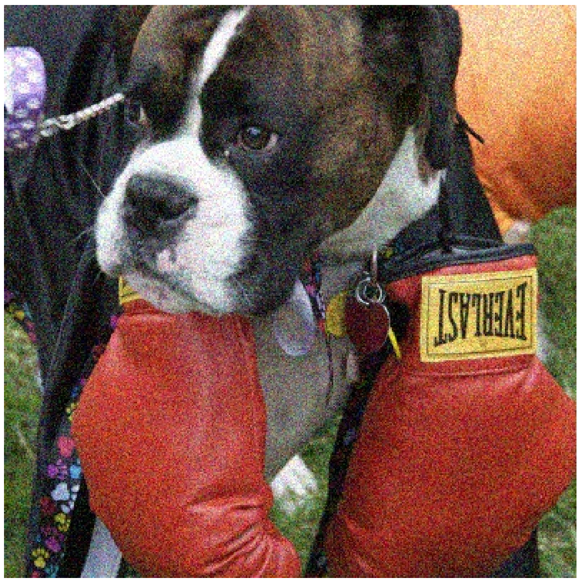


Result of the Noise Clinic using three scales

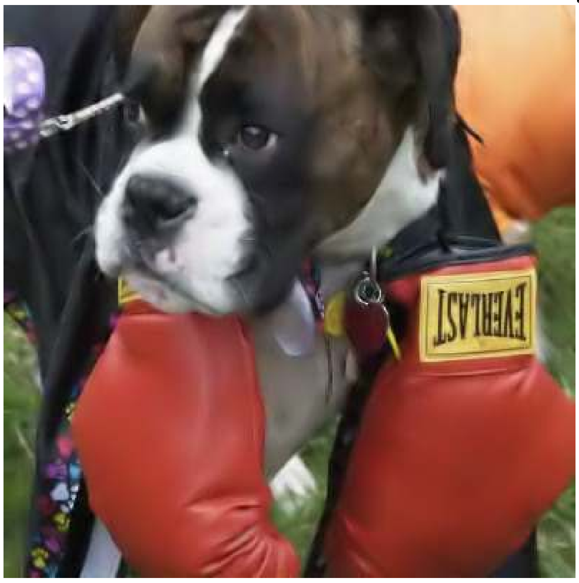


Difference with the noisy input

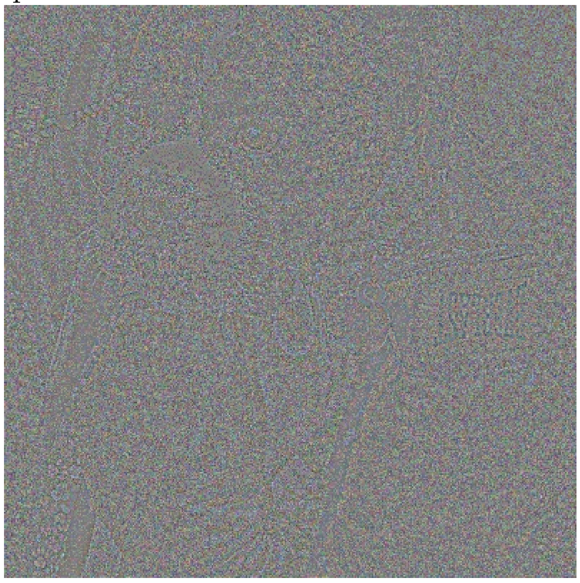
Figure 28: Comparison between the commercial software *Neat Image* and the Noise Clinic on the “Postcard” image.



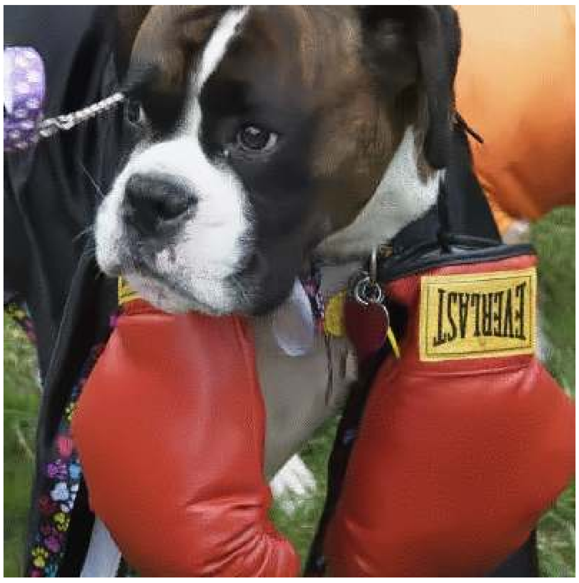
Noisy input



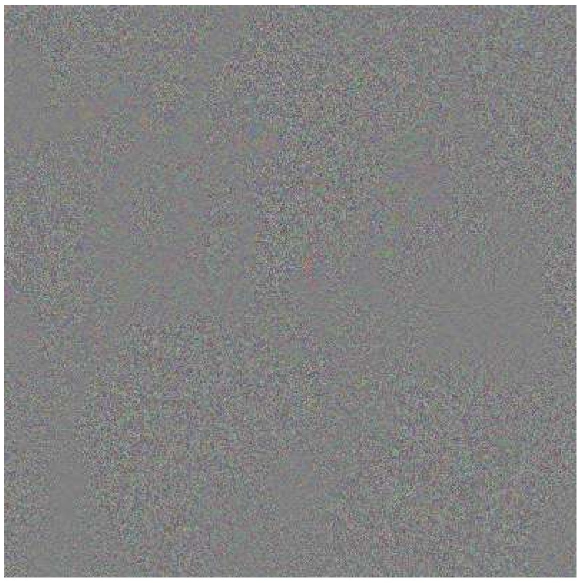
Result of *Neat Image*



Difference with the noisy input

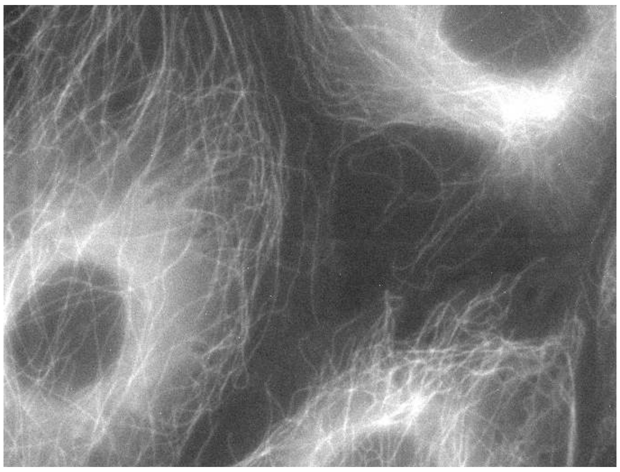


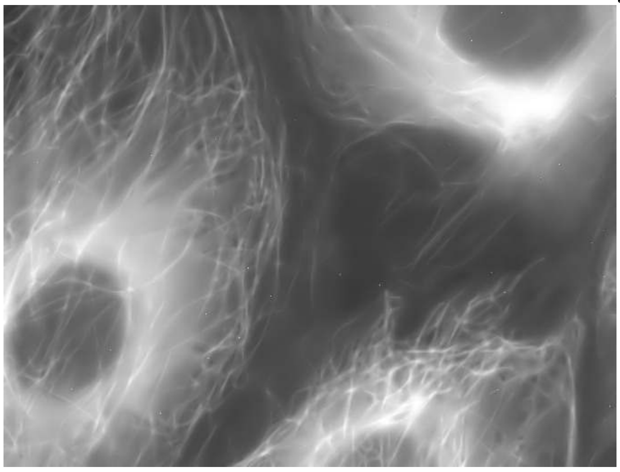
Result of the Noise Clinic using three scales

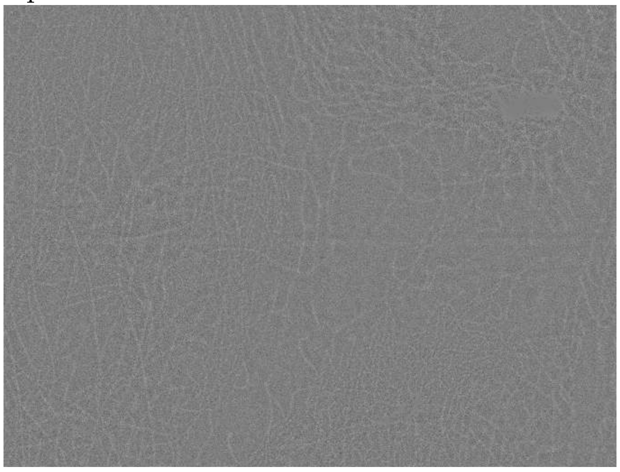


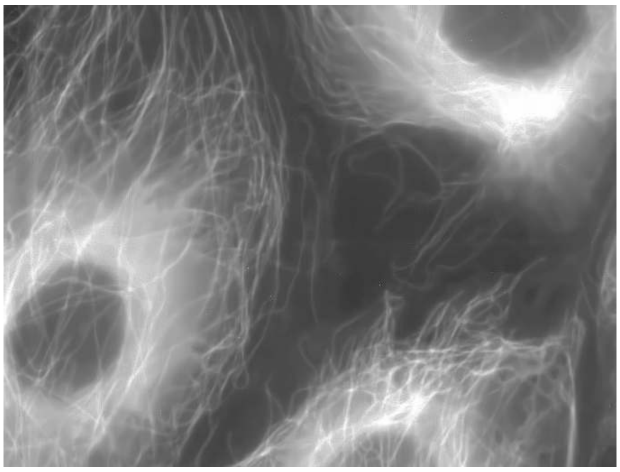
Difference with the noisy input

Figure 29: Comparison between the commercial software *Neat Image* and the Noise Clinic on the "Dog" image.


Noisy input


Result of *Neat Image*


Difference with the noisy input


Result of the Noise Clinic using two scales

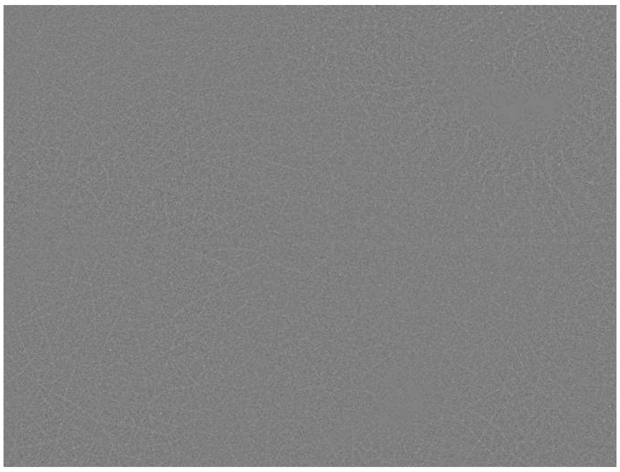
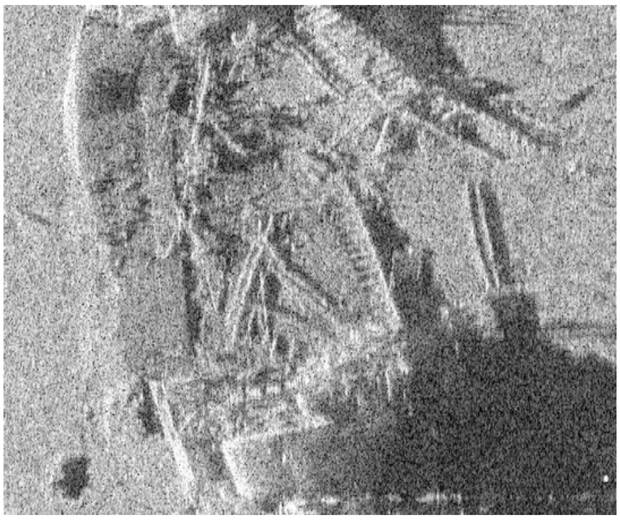
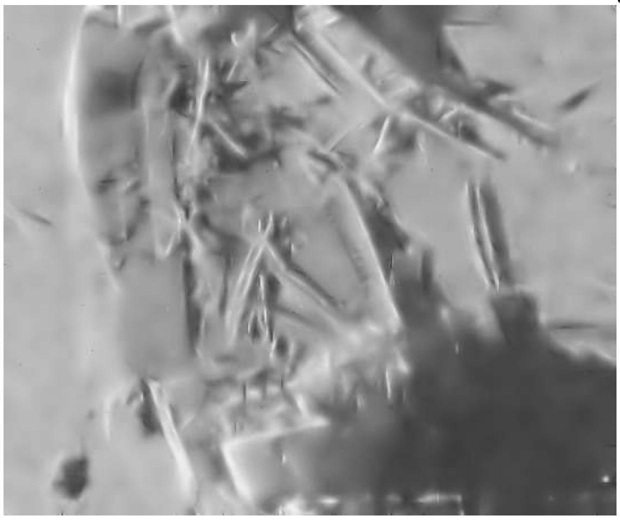

Difference with the noisy input

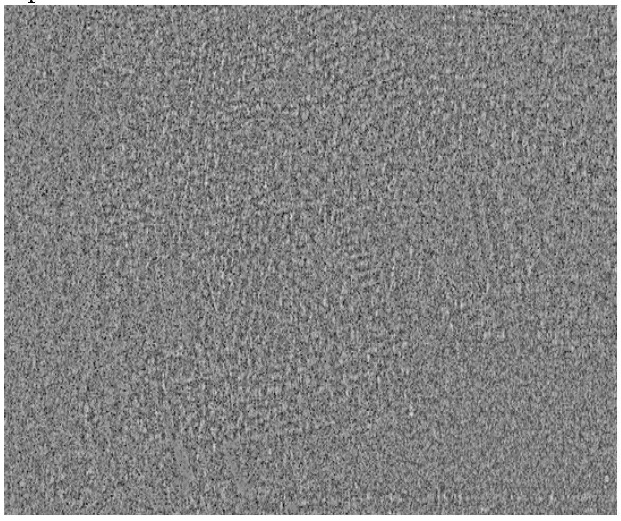
Figure 30: Comparison between the commercial software *Neat Image* and the Noise Clinic on the "Fibers" image.



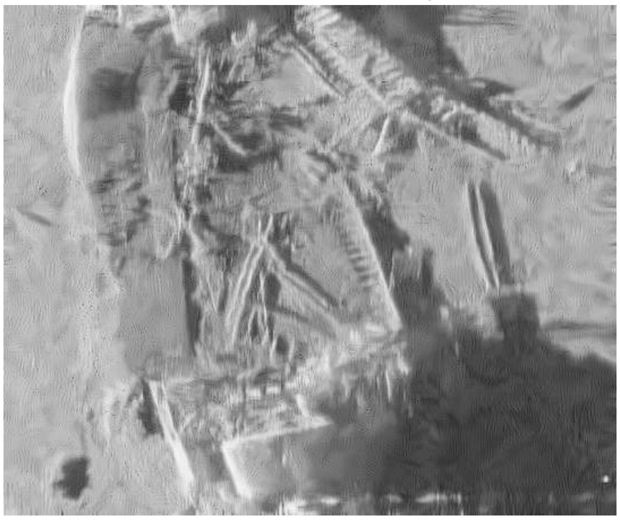
Noisy input



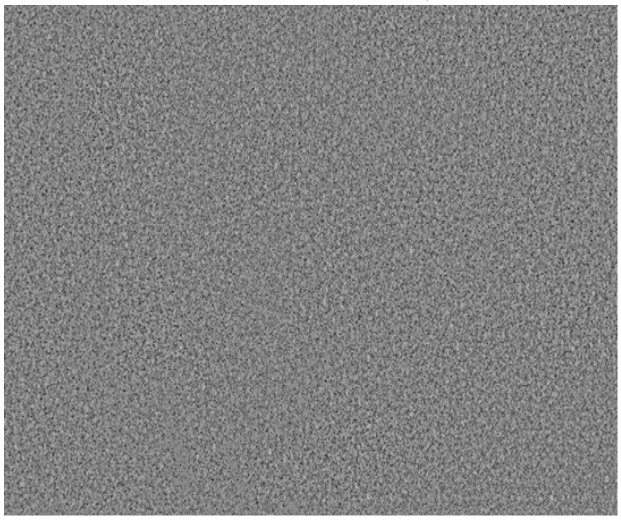
Result of *Neat Image*



Difference with the noisy input

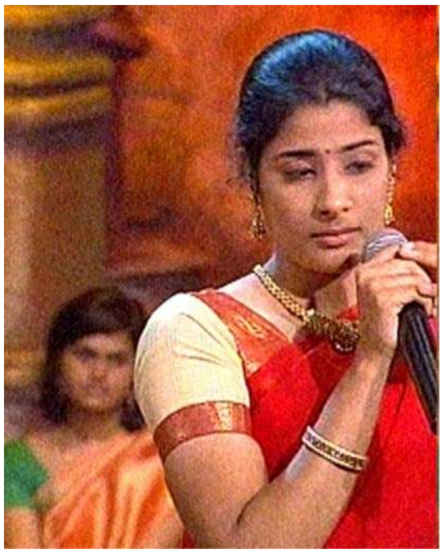


Result of the Noise Clinic using two scales

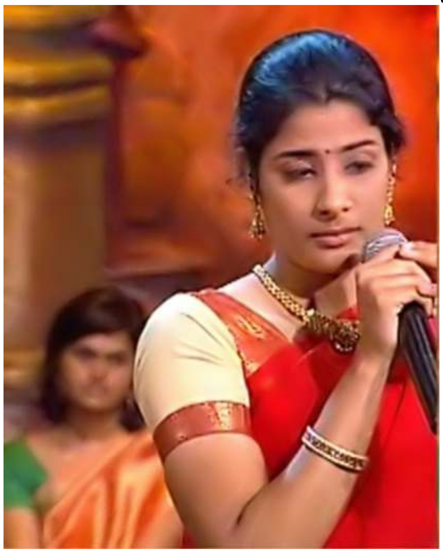


Difference with the noisy input

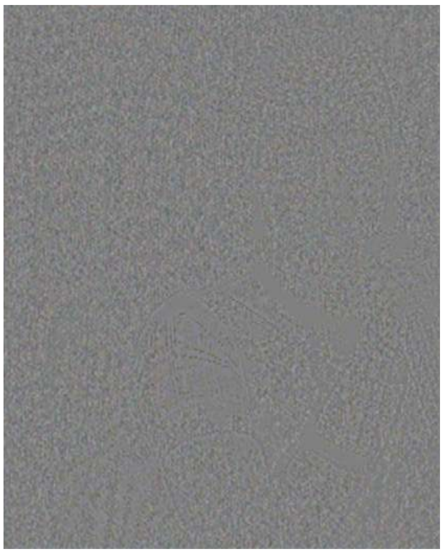
Figure 31: Comparison between the commercial software *Neat Image* and the Noise Clinic on the "Ship" image.



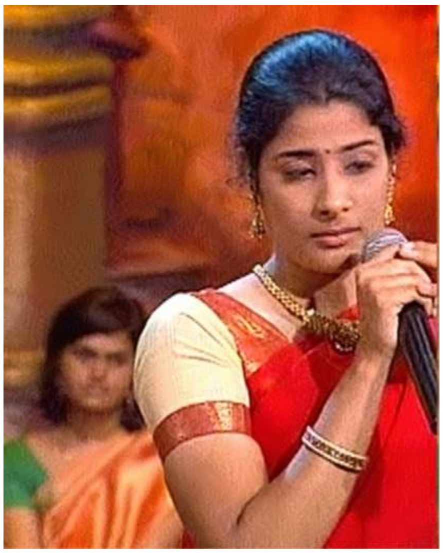
Noisy input



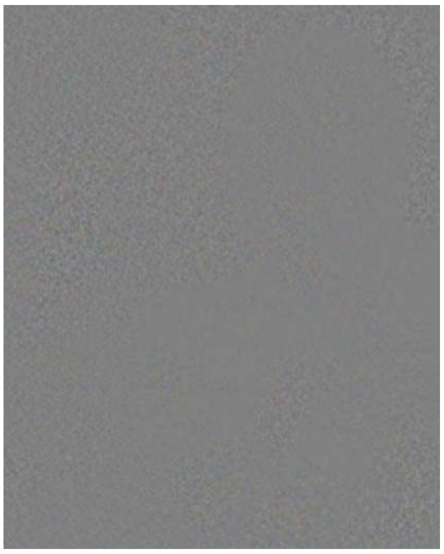
Result of *Neat Image*



Difference with the noisy input

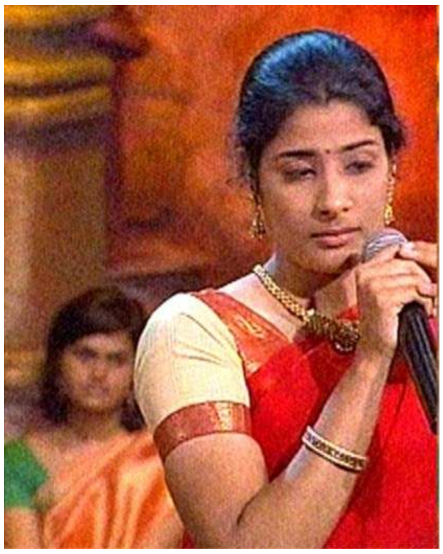


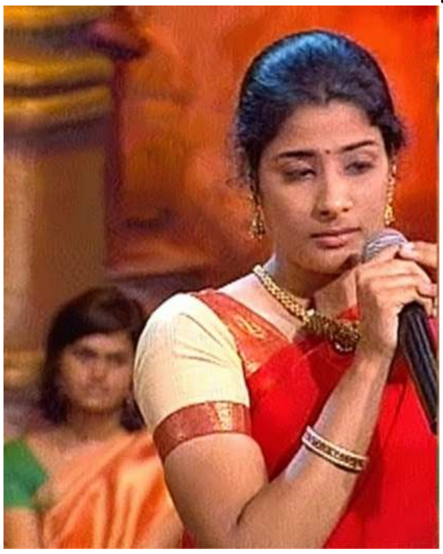
Result of the Noise Clinic using three scales

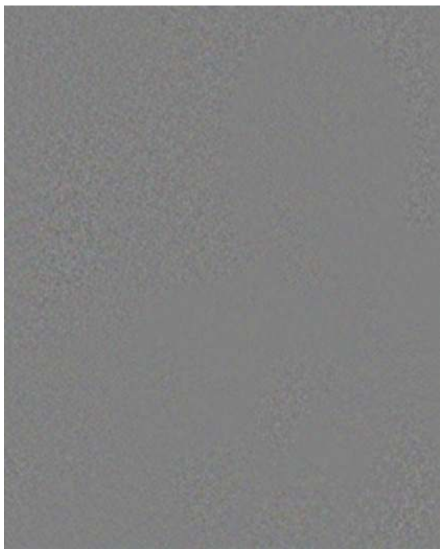


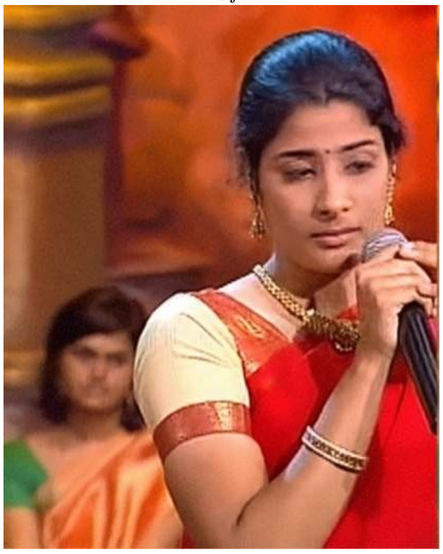
Difference with the noisy input

Figure 32: Comparison between the commercial software *Neat Image* and the Noise Clinic on the "Singer" image. Remark, by increasing the multiplicative factor γ_f up to 1.5, the Noise Clinic presents similar results than neat image (see Figure 33).


Noisy input


Result of the Noise Clinic using three scales, $\gamma_f = 1.0$


Difference with the noisy input


Result of the Noise Clinic using three scales, $\gamma_f = 1.5$

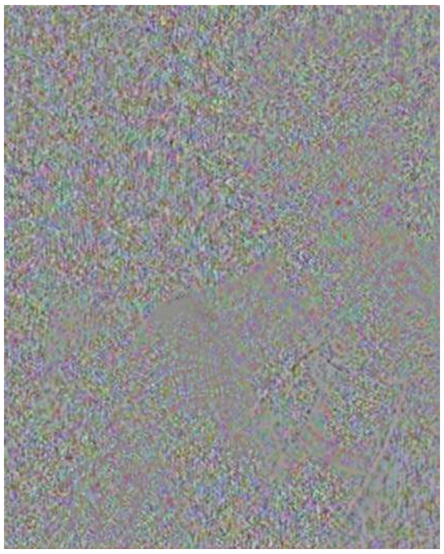

Difference with the noisy input

Figure 33: Illustration of the usefulness of the multiplicative factor γ_f to achieve visually better denoising result for difficult images.

5.6 Validation

Whereas the purpose of the Noise Clinic is to denoise real-life images for which a "noise-free" ground truth is not available, we shall apply a "sanity check", by comparing the Noise Clinic to two non-blind state-of-the-art denoising algorithms, BM3D and NL-Bayes. We have done a series of experiments on an ecclectic set of noiseless images to which white noise of a known variance was added. The contest was biased in the sense that BM3D and NL-Bayes assumed the noise to be Gaussian and were given the exact value. The Noise Clinic instead was applied blindly. One should therefore expect an inferior result for the Noise Clinic. The question is how much is lost by ignoring these two key pieces of information: a) that the noise is white Gaussian and b) the exact value of its standard deviation. A Gaussian noise with six different standard deviations was added to eight different noise-free images. Then the Noise Clinic was applied on those noisy images, with different number of scales. The average results over the eight different images are presented in Table 9.

One can observe an average loss of approximately 0.8dB between NL-Bayes and the Noise Clinic (1 scale), and approximately 0.6dB between BM3D and the Noise Clinic (1 scale). The gap increases when using more scales. Nevertheless a clear gain in visual quality can be observed with 2 or 3 scales with respect to one scale. This (moderate) performance gap may be easily explained:

- The NL-Bayes algorithm used in the Noise Clinic is slightly different from the classic non-blind one, especially the detection of homogeneous area. This detection is more subtle in the Noise Clinic to avoid to remove tiny details by overestimating the noise, whereas in NL-Bayes this detection takes into account the exact knowledge of the noise variance, which is not fair. The detection of flat regions is therefore sharper, and this accounts for most of the gap;
- The noise estimation is of course noisy, depending on relatively few samples. It never gives a completely flat noise curve as can be expected. Then there is over- and under-denoising for some intensity ranges.

σ	BM3D	NL-Bayes	Noise Clinic 1s.	Noise Clinic 2s.	Noise Clinic 3s.
2	45.76	**46.16**	45.07	44.30	43.09
5	40.61	**40.96**	40.31	39.64	39.13
10	36.84	**37.06**	36.26	35.95	35.47
20	33.23	**33.43**	31.55	31.47	30.58
30	31.18	**31.22**	30.08	29.22	28.51
40	**29.71**	29.63	28.59	27.94	27.93

Table 9: Average results in PSNR over eight noise-free images, corrupted by a white Gaussian noise of standard deviation σ. In **bold** the best PSNR for a given σ.

6 Disclaimers, How to Use the Sliders in the Online Demo

This method seems satisfactory for most JPEG images with unknown origin and with scanned old photographs. It compares favourably to commercial software, which tends to lose more detail. Nevertheless we observed that it was sometimes useful to boost the noise estimation by a factor that goes up to two, to improve the visual result. The reader is invited to check this fact for example with the "Singer" image, (see Figure 33). Also a decision must sometimes be made to increase the number of scales, up to three, to deal with thick grain. Thus, we left two sliders in the online demo. The method does not apply to impulse or multiplicative noise, that should be estimated and removed by a separate procedure. Nevertheless, multiplicative noise can be handled by the Noise Clinic after it has been transformed to additive noise by taking the logarithm of the image. The local noise estimation procedure does not cope with highly structured noise, which is sparse in the frequency domain, like in the third infrared image of Figure 25. Strongly compressed images where blocking effects dominate over noise cannot be improved by the present method.

Acknowledgments

Work partially supported by DxO-Labs, the Centre National d'Etudes Spatiales (CNES, MISS Project), the European Research Council (Advanced Grant Twelve Labours), and the Office of Naval Research (Grant N00014-97-1-0839).

Image Credits

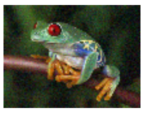
by NASA, not protected by copyright.

, taken from [20] (C) J. Portilla.

, taken from the article "Gaussian KD-Trees for Fast High-Dimensional Filtering" by A. Adams, N. Gelfand, J. Dolson and M. Levoy.

by Wikimedia Commons (user Olaf1541), Creative Commons Attribution-Share Alike 3.0 Unported license.

, Wikimedia Commons, public domain.

 , NL-means archive.

References

[1] F. J. Anscombe, *The transformation of Poisson, binomial and negative-binomial data*, Biometrika, 35 (1948), pp. 246–254.

[2] A. Buades, B. Coll, and J.M. Morel, *A non local algorithm for image denoising*, IEEE Computer Vision and Pattern Recognition, 2 (2005), pp. 60–65. DOI: http://dx.doi.org/10.1109/CVPR.2005.38.

[3] ——, *Non-Local Means Denoising*, Image Processing On Line, 2011 (2011). http://dx.doi.org/10.5201/ipol.2011.bcm_nlm.

[4] M. Colom and A. Buades, *Analysis and Extension of the Percentile Method, Estimating a Noise Curve from a Single Image*, Image Processing On Line, 3 (2013), pp. 332–359. http://dx.doi.org/10.5201/ipol.2013.90.

[5] M. Colom, M. Lebrun, A. Buades, and J.M. Morel, *A non-parametric approach for the estimation of intensity-frequency dependent noise*, in Proceedings of IEEE International Conference on Image Processing, 2014.

[6] ——, *Obtaining a Complete Intensity-Frequency Noise Model from a Single Image*, Image Processing On Line, (2014). Submitted.

[7] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, *Image denoising by sparse 3D transform-domain collaborative filtering*, IEEE Transactions on image processing, 16 (2007), pp. 3736–3745. DOI: http://dx.doi.org/10.1109/TIP.2007.901238.

[8] S. Gabarda and G. Cristóbal, *The generalized Rényi image entropy as a noise indicator*, Noise and Fluctuations in Photonics, Quantum Optics, and Communications, 6603 (2007). http://dx.doi.org/10.1117/12.725086.

[9] P. Getreuer, *Rudin-Osher-Fatemi Total Variation Denoising using Split Bregman*, Image Processing On Line, 2012 (2012). http://dx.doi.org/10.5201/ipol.2012.g-tvd.

[10] M. Lebrun, *An Analysis and Implementation of the BM3D Image Denoising Method*, Image Processing On Line, 2012 (2012). http://dx.doi.org/10.5201/ipol.2012.l-bm3d.

[11] M. Lebrun, A. Buades, and J.M. Morel, *A Nonlocal Bayesian Image Denoising Algorithm*, SIAM Journal Image Science, 6 (2013), pp. 1665–1688. http://dx.doi.org/10.1137/120874989.

[12] ——, *Implementation of the "non-local Bayes" (NL-Bayes) image denoising algorithm*, Image Processing On Line, 2013 (2013), pp. 1–42. http://dx.doi.org/10.5201/ipol.2013.16.

[13] M. Lebrun, M. Colom, and J.M. Morel, *Multiscale Image Blind Denoising*, IEEE Transactions on Image Processing, (2014). Submitted.

[14] ——, *The Noise Clinic, a universal blind denoising algorithm*, in Proceedings of IEEE International Conference on Image Processing, 2014.

[15] M. Lebrun and A. Leclaire, *An Implementation and Detailed Analysis of the K-SVD Image Denoising Algorithm*, Image Processing On Line, 2012 (2012). http://dx.doi.org/10.5201/ipol.2012.llm-ksvd.

[16] Liu, W. Freeman, R. Szeliski, and S. Kang, *Automatic estimation and removal of noise from a single image*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 30 (2008), pp. 299–314. DOI: http://dx.doi.org/10.1109/TPAMI.2007.1176.

[17] J. Mairal, M. Elad, and G. Sapiro, *Sparse representation for color image restoration*, IEEE Transactions on Image Processing, 17 (2008), pp. 53–69. DOI: http://dx.doi.org/10.1109/TIP.2007.911828.

[18] J. Mairal, G. Sapiro, and M. Elad, *Learning multiscale sparse representations for image and video restoration*, SIAM Multiscale Modeling and Simulation, 7 (2008), pp. 214–241. DOI: http://dx.doi.org/10.1137/070697653.

[19] N. Ponomarenko, V. Lukin, K. Egiazarian, and J. Astola, *A method for blind estimation of spatially correlated noise characteristics*, in IS&T/SPIE Electronic Imaging, International Society for Optics and Photonics, 2010, pp. 753208–753208. http://dx.doi.org/10.1117/12.847986.

[20] J. Portilla, *Blind non-white noise removal in images using Gaussian scale mixtures in the wavelet domain*, Benelux Signal Processing Symposium, (2004).

[21] ——, *Full blind denoising through noise covariance estimation using Gaussian scale mixtures in the wavelet domain*, in IEEE International Conference on Image Processing, vol. 2, 2004, pp. 1217–1220. DOI: http://dx.doi.org/10.1109/ICIP.2004.1419524.

[22] T. Rabie, *Robust estimation approach for blind denoising*, IEEE Transactions on Image Processing, 14 (2005), pp. 1755–1765. DOI: http://dx.doi.org/10.1109/TIP.2005.857276.

[23] L. I. Rudin, S. Osher, and E. Fatemi, *Nonlinear total variation based noise removal algorithms*, Physica D: Nonlinear Phenomena, 60 (1992), pp. 259–268. DOI: http://dx.doi.org/10.1016/0167-2789(92)90242-F.

[24] J.L. Starck, E.J. Candès, and D.L. Donoho, *The curvelet transform for image denoising*, IEEE Transactions on image processing, 11 (2002), pp. 670–684. DOI: http://dx.doi.org/10.1109/TIP.2002.1014998.

[25] L.P. Yaroslavsky, *Local adaptive image restoration and enhancement with the use of DFT and DCT in a running window*, in Proceedings of Society of Photo-Optical Instrumentation Engineers (SPIE), vol. 2825, 1996, pp. 2–13. DOI: http://dx.doi.org/10.1117/12.255218.

[26] L.P. Yaroslavsky, K.O. Egiazarian, and J.T. Astola, *Transform domain image restoration methods: review, comparison, and interpretation*, in Proceedings of Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series, vol. 4304, May 2001, pp. 155–169. DOI: http://dx.doi.org/10.1117/12.424970.

[27] G. Yu and G. Sapiro, *DCT image denoising: a simple and effective image denoising algorithm*, Image Processing On Line, 2011 (2011). http://dx.doi.org/10.5201/ipol.2011.ys-dct.