ON THE PARAMETER CHOICE FOR THE NON-LOCAL MEANS *

VINCENT DUVAL†, JEAN-FRANÇOIS AUJOL‡, AND YANN GOUSSEAU†

Abstract. This paper deals with the parameter choice for the NL-Means algorithm. Starting with basic computations on toy models, we study the bias-variance trade-off of this filter using a simple notion of regularity in the patch space. We show that this regularity is necessarily local and so should be the parameters of the filter. Relying on Stein's Unbiased Risk Estimate, we then propose a way to locally set these parameters, and we compare this method with the Non-Local Means with optimal global parameter.

1. Introduction. In recent years, patch-based methods have drawn a lot of attention in the image processing community. Inspired by the work of Efros and Leung [10] for the texture synthesis problem, Buades et al. have proposed the Non-Local Means (or NLmeans) denoising algorithm in their seminal paper [6]. Their idea (also suggested independently in [1]), is to take advantage of self-similarities in images by comparing local neighborhoods (the "patches") across the whole image. Each pixel value is estimated as a weighted average of all others, and the pixels whose neighborhoods are the most similar to the neighborhood of the one to be denoised are given the largest weights. This novel point of view has stimulated many authors and it is at the core of most of the state-of-the-art image denoising algorithms.

The aim of this paper is to discuss the choice of parameters of the standard NLmeans filter. Our study reveals that the parameters should be set locally within the image. Numerically, it raises the problem of automatic parameter tuning. Resorting to Stein's Unbiased Risk Estimate (SURE) [29], we introduce an algorithm which automatically computes the best local parameters in the image, so that we can propose a spatially adaptive version of the NLmeans filter.

To discuss the tuning of parameters of the NLmeans algorithm, we interpret this choice as a bias-variance dilemma. Contrary to the approach of [14], which also relies on a bias variance analysis, we mainly focus on the choice of the smoothing parameter rather than on the search window. This choice should be made depending on the regularity of the image, a notion that is to be defined. To begin with, we make use of the condition presented in [25] as a necessary assumption on the signal for NLmeans to work:

Assumption (Patch Regularity) Similar patches have similar central pixels.

We shall refer to this property as *patch regularity*. This concept will be made precise in the rest of the paper. Notice that a similar, information theoretic formulation

* This work was supported by the grant FREEDOM (ANR07-JCJC-0048-01), "Movies, restoration and missing data."

†Institut Telecom, Telecom ParisTech, CNRS UMR 5141, F-70513 Paris, France (vincent.duval@telecom-paristech.fr, yann.gousseau@telecom-paristech.fr). The first author's research was supported by the French "Agence Nationale de la Recherche" (ANR) under grant NATIMAGES (ANR-08-EMER-009), "Adaptivity for natural images and texture representations."

‡CMLA, ENS Cachan, CNRS, UniverSud, F-94230 Cachan, France (Jean-Francois.Aujol@cmla.ens-cachan.fr), and LATP, CMI, Université de Provence, UMR CNRS 6632, 13453 Marseille cedex 13, France (aujol@cmi.univ-mrs.fr). This author's research was supported by the French "Agence Nationale de la Recherche" (ANR) under grant NATIMAGES (ANR-08-EMER-009), "Adaptivity for natural images and texture representations."

can be found in [1], stating that conditionally to the rest of the patch, the entropy of the law of the central pixel is very low.

The different contributions in the literature range from a better understanding of the algorithm to the proposition of brand new algorithms, but one may classify them in three categories.

Several authors have proposed a variational framework for non-local image processing [15, 11, 12], bridging the gap between total variation based methods and non-local algorithms. Let us also mention the work of Brox et al. [4] and Azzabou et al. [3] based on similar variational considerations. Generally, these variational problems are solved using iterative methods. Since non local filters require a lot of calculations, reducing their computation time is a crucial matter, not only for iterative denoising methods but also for real time denoising, and several authors have proposed fast implementations of the Non-Local Means, among which [8, 5].

A different point of view consists in addressing the performance of the NLmeans algorithm by studying images in the patch space: Peyré shows in [20] that for several models of signals (e.g. smooth or cartoon), images lie on a manifold in the patch space, and he deduces restoration algorithms in that framework. In [28], the NLmeans algorithm is interpreted as one step of a heat equation in the patch space and the connection with classical diffusion-based denoising algorithms is established. The analysis of this non-local heat equation is carried further in [19, 23], where a whole point of view on spectral analysis on manifolds is developed.

Eventually, one may examine non local methods from a more statistical point of view: in [13], the NLmeans are extended to deal with coloured noise and the choice of the weight function is discussed. In [2], the authors propose an adaptive denoising algorithm based on Marginal Posterior Modes estimation, whereas Deledalle et al. [9] introduce an iterative algorithm based on maximum likelihood estimation which allows one to deal with other noises than Gaussian, e.g. multiplicative speckle noise. Moreover, a new light was shed on the classical NLmeans by Salmon and Le Pennec [22] who interpret it as the aggregation of estimators in a PAC-Bayesian approach. Also, it was recently proposed in [29] to globally set the parameters of the NLmeans by the SURE method.

In terms of performances, some of the best denoising results are obtained, to our knowledge, by two different approaches that rely on non-locality and patch-denoising: the BM3D algorithm, which is built on classical signal processing tools, proposed by [7] and another one which is based on learning dictionaries of patches in [17].

Notations. Let us consider a d-dimensional signal u, with $d \in \{1, 2\}$, defined on a domain $\Omega \subset \mathbb{Z}^d$. Given an odd number s and a pixel $x \in \Omega$, we define the patch $U(x)$ of width s centered at x as the s^d-dimensional vector whose coordinates are gray values of the pixels in a square neighborhood of x with side s :

$$U(x) = (u(x + j))_{|j|_\infty \leq \frac{s-1}{2}}. \tag{1.1}$$

The NLmeans filter compares patches in the image u to restore the value at pixel x according to the following formula:

$$NLu(x) = \frac{\sum_{y\in\Omega} e^{-\frac{\|U(x)-U(y)\|^2}{2h^2}} u(y)}{\sum_{y\in\Omega} e^{-\frac{\|U(x)-U(y)\|^2}{2h^2}}} \tag{1.2}$$

The l^2 norm we use to compare the patches is *normalized*:

$$\|U(x) - U(y)\|^2 = \frac{1}{s^d} \sum_{|j| \leq \frac{s-1}{2}} (u(x+j) - u(y+j))^2,$$

so that h is homogeneous to a gray level.

Since the common habit is to restrict the above search for patches to a search window of side-length W around x, the sums in (1.2) may be replaced by sums over all $y \in \Omega$ such that $|x-y|_\infty \leq \frac{W-1}{2}$.

Therefore, when denoising an image using NLmeans, the user has to set the following parameters:

- The patch size s, which is often set to 7×7 or 9×9, but should a priori be related to the scale of objects in the image.
- The smoothing parameter h, which stands for the typical distance between similar patches, and which depends on the noise level. Some authors have proposed to choose it proportional to the noise standard deviation.
- The size of the search window W, which has a dramatic impact on the computation time, but which also has an influence on the visual quality of the results.

In fact, these parameters are far from being independent, and making the right choice is no easy task. Our aim is to give some insight on how these parameters interact with one another. The guiding principle of our study is the bias variance trade-off: Section 2 is devoted to highlighting some qualitative properties of the NLmeans filter on one and two-dimensional toy examples, in particular the bias of the NLmeans in these cases. In general, explicit computations of the solutions are out of reach, so that in Section 3 we use a representation in the patch space to predict whether the estimation of a pixel is much biased or not. This is of course related to the *patch regularity*, and a few examples should convince the reader that this regularity is necessarily local. In the last Section, we handle this trade-off locally, first by using an oracle which gives the main guidelines of an efficient denoising, then by relying on *Stein's Unbiased Risk Estimate*, first proposed in [29] in the framework of NLmeans. Incidentally, we propose a fast method to find the optimal global parameter of the NLmeans.

Let us emphasize the main contributions of the paper :

- understanding the *patch regularity* in term of bias variance trade-off, and in particular showing the need to tune the parameters of the NLmeans algorithm locally.
- introducing an automatic spatially adaptive version of the NLmeans, and demonstrating the effectiveness of the approach on various numerical examples.

2. Understanding NLmeans with basic examples. The aim of this section is to illustrate some qualitative properties of the NLmeans filter using elementary computations. We deal with deterministic signals and we assume that there is no noise. Informally, one may think of this as a study of the "bias" of the filter.

Most examples are one-dimensional but the conclusions extend straightforwardly to images that are invariant along one axis.

2.1. Periodic crenel. The reader who is unfamiliar with the NLmeans filter might think that it is able to restore arbitrarily well any periodic signal. However, it can be shown that the only functions that are invariant by NLmeans are constant

functions (a direct proof is given in [11]), so that even periodic signals must be altered one way or another. The following example gives a concrete illustration of this fact.

Assume we want to denoise a quickly oscillating periodic texture which can be modelled by a crenel. We work on a one-dimensional signal, but the result extends to images of black and white stripes which are parallel to the axes of the image.

Let us precise the framework. For simplicity, we shall assume that:

- The periodic signal is a crenel, with period T and intensity α.
- The period is $T = 2p$ where p is an odd number.
- The patch size is $s = \left(k + \frac{1}{2}\right) T$, for $k \in \mathbb{N}$ (i.e. the period of the signal is small compared to the patch size). Notice that the following result still holds when $s = kT$, but it is unusual to set an even patch size.
- The size of the search window is infinite, or it is a multiple of T.

An illustration of the following discussion is given in Figure 2.1.

Patch distances. Fixing one pixel x, let us compute the patch distances. For $-\frac{T}{2} + 1 \leq j \leq \frac{T}{2}$, a translation by j pixels leads to a difference of $(2k+1)|j|$ pixels, and the squared distance between two patches is :

$$\|U(x) - U(x_j)\|^2 = \frac{(2k+1)|j|\alpha^2}{s} = \frac{2}{T}|j|\alpha^2$$

where $U(x)$ and $U(x_j)$ denote the patches centered respectively at x and x_j, and $x_j = x + j$.

By periodicity, we get the other distances and we see that the distance function is a periodic triangle (see Figure 2.1).

Output of the filter. Assume that the pixel x we want to denoise belongs to the crenel, i.e. there exist $j_1, j_2 \geq 0$ such that $u(x_j) = \alpha$ for $-j_2 \leq j \leq j_1$, with $j_1 + j_2 + 1 = \frac{T}{2}$ and $u(x_j) = 0$ otherwise.

By periodicity, we have :

$$NLu(x) = \frac{\sum_{-\frac{T}{2} < j \leq \frac{T}{2}} e^{-\frac{\|U(x) - U(x_j)\|^2}{2h^2}} u(j)}{\sum_{-\frac{T}{2} < j \leq \frac{T}{2}} e^{-\frac{\|U(x) - U(x_j)\|^2}{2h^2}}}$$

Let us write $r = \frac{1}{T}\frac{\alpha^2}{h^2}$. Then $NLu(x) = \alpha \frac{A}{B}$ with on the one hand:

$$B = 2\sum_{j=0}^{\frac{T}{2}-1} e^{-rj} - 1 + e^{-r\frac{T}{2}} = \frac{1}{e^{-r} - 1}(e^{-r\frac{T}{2}} - 1)(1 + e^{-r})$$

and on the other hand:

$$A = \sum_{j=0}^{j_1} e^{-rj} - 1 + \sum_{j=0}^{j_2} e^{-rj} = \frac{1}{e^{-r} - 1}\left(2e^{-(\frac{j_1+j_2}{2}+1)r}\cosh(\frac{j_1 - j_2}{2}r) - (e^{-r} - 1)\right).$$

Recalling that $j_1 = \frac{T}{2} - 1 - j_2$, we set the origin at the center of the crenel (i.e. $x = 0$ is the central pixel of the crenel) and we get:

$$NLu(x) = \frac{\alpha}{(1 - e^{-r\frac{T}{2}})(1 + e^{-r})}\left(1 - e^{-r} - 2e^{-\frac{1}{2}(\frac{T}{2}+1)r}\cosh rx\right). \qquad (2.1)$$

The expression for the other parts of the signal are obtained by replacing $NLu(x)$ with $\alpha - NLu(x)$ and translating the signal by $\frac{T}{2}$.

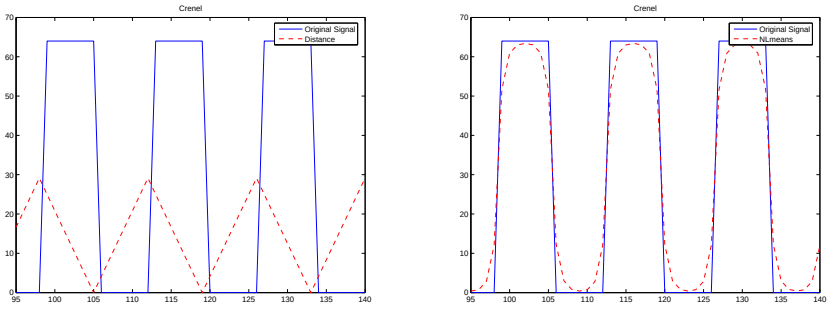


FIG. 2.1. *Effect of NLmeans on a periodic crenel. Left, an extract of the original signal (solid blue line) and the patch distance to the patch centered at pixel 118 (dashed red line, rescaled). Right, the same extract of the original signal (solid blue line) and the result provided by the NLmeans filter (red dashed line): it is a piecewise catenary. The analytic expression of this result is given by (2.1). The period of the crenel is* $T = 14$*, its intensity* $\alpha = 64$*, the patch size is* $s = 21$*,* $h = 20$*.*

Comments. In particular, at the boundary of the crenel, the intensity of the pixel is:

$$NLu(x) = \alpha \frac{1}{1 + e^{-r}}. \tag{2.2}$$

As a consequence, the filter smoothes the discontinuity.

Clearly, *the NLmeans algorithm does not preserve this periodic signal.* In this particular case we can draw the following conclusions :

- In each region of the domain, the solution given by NLmeans is a *catenary.*
- Averaging property : for fixed α, T, and $h \to +\infty$ (infinite tolerance) we get: $NLu(x) \sim \frac{\alpha}{2}$ (grey image). For $\alpha \gg h$ (i.e. $r \to +\infty$), $NLu(x) \sim \alpha$.
- The less contrasted the details, the more they are degraded by the filter. The non-linearity of this effect is illustrated in (2.2) since r depends on α.
- When there is no noise, the patch size has no influence on the final result provided that it is larger than the characteristic size of the periodic pattern. Considering larger or smaller patches does not allow to discriminate the pixels of the crenel better. When dealing with noisy images, it is of course better to consider larger patches, since for large s, the (normalized) squared distance $\|\tilde{U}(x) - \tilde{U}(x_j)\|^2 = \frac{1}{s}\sum_{i=1}^{s}(\tilde{u}(x+i) - \tilde{u}(x_j+i))^2$ is likely to be close to its expectancy $\mathbb{E}\|\tilde{U}(x) - \tilde{U}(x_j)\|^2 = \|U(x) - U(x_j)\|^2 + 2\sigma^2$ by the law of large numbers, which, as noticed in [6], preserves the order of similarity between patches. Yet, textures are not exactly periodic: they may be repeated slightly differently, their period or pattern may vary slowly (or quickly) which prevents from using very large patches (see Fig. 3.8). We shall come back to this dilemma in Subsection 3.4.

2.2. Isolated details and choice of the search window. Assume now that we want to denoise an isolated pattern, modeled by a single crenel of size $\frac{T}{2}$ and intensity α, and, for simplicity, that the patch size is also equal to $\frac{T}{2}$. This time we assume that the signal has length $N \gg T$.

As before, considering a point x, we can compute the distance between patches:

$$\|U(x) - U(x_j)\|^2 = \begin{cases} \frac{2|j|\alpha^2}{s} & \text{for } |j| \leq j_1 + j_2 + 1 = \frac{T}{2} \\ \frac{2T\alpha^2}{s} & \text{otherwise.} \end{cases} \tag{2.3}$$

Let us write $r = 2\frac{\alpha^2}{sh^2}$.

Let A (resp. B) be the numerator (resp. denominator). We have :

$$A = \sum_{j=0}^{j_1} e^{-rj} - 1 + \sum_{j=0}^{j_2} e^{-rj} = \frac{1}{1-e^{-r}} \left(1 - e^{-r} - 2e^{-(\frac{j_1+j_2}{2}+1)r} \cosh(\frac{j_1-j_2}{2} r)\right),$$

$$B = 2\sum_{j=0}^{\frac{T}{2}} e^{-rj} - 1 + (N-T-1)e^{-r\frac{T}{2}}.$$

Therefore, the result of NL-means inside the crenel is (see Figure 2.2):

$$NLu(x) = \alpha \frac{1 - e^{-r} - 2e^{-\frac{1}{2}(\frac{T}{2}+1)r} \cosh rx}{(1-e^{-r})\left(2\sum_{j=0}^{\frac{T}{2}} e^{-rj} - 1 + (N-T-1)e^{-r\frac{T}{2}}\right)}. \tag{2.4}$$

In the case of an infinite signal (or image) ($N \to +\infty$) we see that $NLu(x) = 0$ for all x ! In practice, the image is finite, so we do not get a black picture. However, it is worth noticing that the result when denoising an object depends on the size of the image one embeds it in. This behavior is not absurd since in large images a small structure is more likely to appear because of the noise. However, this is quite different from what happens when denoising images with total variation based methods for instance. An interpretation of this phenomenon is that NLmeans has the following a-priori : *good images are images with repetitive patterns.* The larger the background, the less the isolated detail is "repeated" compared to the background.

In dimension 2, this phenomenon also happens, not only on a tiny set of pixels, but also on lines, no matter how long they are. Indeed, let us consider a white line Δ over a dark background. In an image of N pixels, such a line has at most, say, $\sqrt{N}$ pixels. Then the result for a point x of the line is approximately :

$$\begin{aligned} NLu(x) &= \frac{\alpha \sum_{x_j \in \Delta} e^{-\frac{\|U(x)-U(x_j)\|^2}{2h^2}}}{\sum_{x_j \in \Delta} e^{-\frac{\|U(x)-U(x_j)\|^2}{2h^2}} + \sum_{x_j \notin \Delta} e^{-\frac{\|U(x)-U(x_j)\|^2}{2h^2}}} \\ &\approx \frac{\alpha\sqrt{N}}{\sqrt{N} + (N-\sqrt{N})e^{-\frac{\delta^2}{2h^2}}} \\ &\approx \alpha \frac{e^{\frac{\delta^2}{2h^2}}}{\sqrt{N}} \end{aligned} \tag{2.5}$$

where we used the approximation $e^{-\frac{\|U(x)-U(x_j)\|^2}{2h^2}} \approx 1$ for all point $x_j \in \Delta$, and $e^{-\frac{\|U(x)-U(x_j)\|^2}{2h^2}} \approx e^{-\frac{\delta^2}{2h^2}}$ otherwise. Here δ^2 stands for the squared distance from $U(x)$ to a dark patch.

As $N \to +\infty$, the line fades (see Figure 2.3). The problem is that, when denoising a small detail or a line, pixels are averaged with *any other*. Because of the exponential function, the weights assigned to the wrong patches are small, but they are nonzero. If these patches are very numerous they will have a strong influence.

Notice that this behavior also appears with the Yaroslavsky filter [30], but with NLmeans it is strengthened by the use of large patches. Indeed, in that case we see

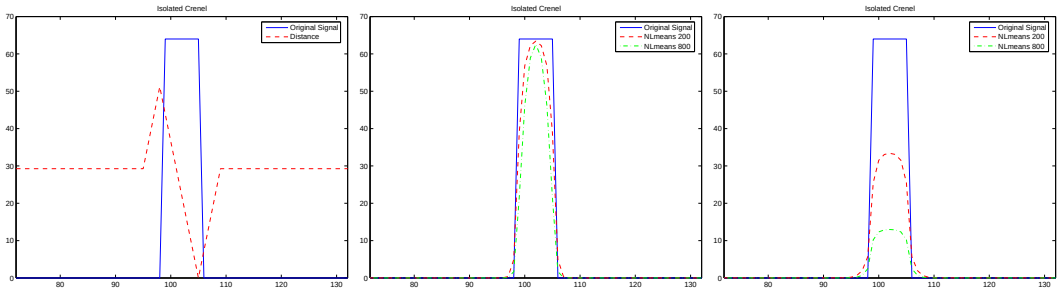


FIG. 2.2. *Bias and loss of isolated details. Left: an extract of the synthetic input signal modelling an isolated detail (solid blue line) and the patch distance to the patch centered at pixel 105 (dashed red line, rescaled). Middle: the same input (solid blue line) and the result provided by the NLmeans filter (red dashed line: when the total signal size is $N = 200$, green dotted line when $N = 800$ - zeros where added): it is a catenary inside the crenel. The analytic expression of this result and therefore the bias is given by (2.4). The size of the crenel is 7, its intensity $\alpha = 64$, the patch size is $s = 7$, $h = 20$. The increase of the total size of the signal has little effect on the bias since the pattern has a similar scale as the patches ($e^{-r\frac{T}{2}} \ll 1$). Right: same experiment, using a patch size of $s = 15$. Since the patch size is larger than the pattern, the size of the image has a large impact on the bias ($e^{-r\frac{T}{2}} \approx 1$).*

FIG. 2.3. *Loss of lines and isolated details. Left: Boat image with little noise ($\sigma = 5$). Middle: Result of the NLmeans filter ($h = 6$, $s^2 = 7 \times 7$, search window 11×11). Right: same experiment, using a search window 61×61. Notice that several ropes vanish when the size of the search window increases (this should be seen on a computer screen).*

that in the above examples (Eq. 2.4 and 2.5), the weights assigned to the background pixels (namely $e^{-r\frac{T}{2}}$ and $e^{-\frac{\delta^2}{2h^2}}$) are large, which implies that the details fade even more (see Figure 2.2, right). In the literature, the dependence on the size of the image is transferred to another issue because one usually restricts the set of patches to a search window around x. Originally ([6, 1]) this was intended to speed up the algorithm, but several authors (e.g. [11]) reported that it also lead to an improvement of the visual result. We shall discuss this point in Section 3.3. For now, let us notice that this raises two issues. First, using a search window turns the problem of the size of the image into another one: one has to choose the size of the window. Second of all, on the theoretical ground, it is very different to restrict the search of patches to a window for computational reasons than to do this because we know that it will produce better results. The second approach contradicts the "non local philosophy": it amounts to saying that beyond a certain (spatial) distance objects are useless for the estimation of a pixel. However, if a similar object appears at the other side of the image (and this happens in natural images) there is no reason why we should refrain from using it to denoise x.

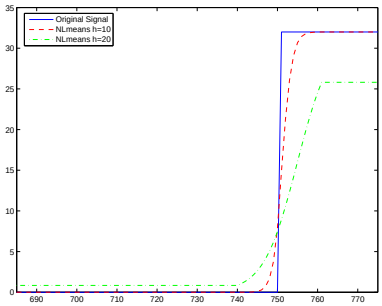


FIG. 2.4. *Effect of the NLmeans on a step signal. The size of the signal is* $N = 1000$*, the first* 750 *coefficients are zero, the other 250 have intensity* $\alpha = 32$*. The parameters are* $s = 21$*,* $h = 10$ *(dashed line) and* $h = 20$ *(dash-dot line). The length of the transition area is equal to the patch size. The bias at each pixel is all the more important as* h *is large and the pixel value is rare (see (2.6)).*

2.3. Step edges and the patch size. In this last toy example, we wish to focus on the effect of the patch size on the smoothing of edges. Consider for instance an image with N pixels, divided in two parts, one being black and the other being white (with intensity value α). We assume that there are θN black pixels and $(1-\theta)N$ white pixels, with $\theta \in (0,1)$. Arguing as in Paragraph 2.1, we may compute the analytic expression of the output of the NLmeans filter, and we see that there are two phases, black and white (very slightly moved towards grey), and a transition between them which lies along the edge and with width proportional to the patch size s. The analytic expression is a bit more complicated than before, since the position of the pixel appears both at the numerator and at the denominator, but taking the limit as $N \to +\infty$, one can show that :

$$NLu(x) = \frac{\alpha(1-\theta)}{(1-\theta) + \theta e^{(2b(x)-s)r}}, \tag{2.6}$$

where $b(x)$ is the number of black pixels in the patch centered at x (e.g. $b(x) = s$ if the patch of center x lies completely inside the black region), and $r = \frac{\alpha^2}{2h^2 s}$. An example for $\theta = \frac{3}{4}$ is shown in Figure 2.4.

In practice, the smoothing effects described in this section are not always observed because they may be negligible compared to the noise. The models studied here mainly describe the vanishing of details and textures with low contrast, as the numerical experiments of Section 4 illustrate.

To sum up, we have seen three toy examples showing that:

- Even periodic signals are biased.
- Small details vanish as the search window increases.
- Low contrasted edges are blurred.

Of course, the point is not to reproach the filter with smoothing images: we simply try to give some intuition on the "bias part" of the error. In the next section, we carry on the discussion taking the noise into account.

3. Bias, risk and regularity in the patch space. In general, and contrary to the above toy examples, one may not be able to derive analytic expressions of the output of the filter. In this section, we regard the choice of the parameter as a bias-variance trade-off. To this end, we propose in this section a simple notion of regularity to predict if a pixel will suffer from a strong bias or not.

3.1. Non-Local regularity and the patch space. In this subsection, we try to make precise the Patch Regularity assumption exposed in the introduction, which

determines the bias of the Non-Local Means estimator. While several authors ([19, 20, 27, 28, 23]) interpret the behavior of the algorithm as a diffusion on a manifold in the patch space (and, indeed, there is strong evidence that the patches of an image lie on a manifold), the point of view we adopt is more elementary and does not take into account the smoothness of the set of patches of the image.

Let us precise the notations. In this paragraph, we may assume that images/signals are defined on a continuous domain $\Omega \subset \mathbb{R}^d$.

Consider the patch application:

$$U : \begin{array}{ccc} \Omega & \longrightarrow & \mathcal{P} \subset \mathbb{R}^{s^d} \\ x & \mapsto & (u(y), |y-x|_\infty \leq \frac{s-1}{2}) \end{array} \tag{3.1}$$

which maps every pixel x of the domain to the patch of center x in the patch space $\mathcal{P}$. Provided u is smooth enough, this gives a parametrization of a curve/ surface in the patch space (however, we do not need any manifold assumption in the following). Consider for instance a periodic image u represented in the patch space: the manifold is a closed curve/surface. However this representation does take into account the repetition of patches, which clearly matters for NLmeans.

Therefore, if we want to study the patch regularity in the patch space, we should not focus on the geometry but on the mass of the patch set: we need to *count* the number of patches with the same values that actually belong to the image. To this end, let us define a measure on the patch space by pushing forward the Lebesgue measure of the signal domain:

$$\forall A \in \mathcal{B}(\mathbb{R}^{s^d}), m(A) = \mathcal{L}^d(U^{-1}(A))$$

Notice that, if the search window is equal to the whole image, the weights of NLmeans $w(x,y) = \frac{1}{C(x)} e^{-\frac{\|U(x)-U(y)\|^2}{2h^2}}$ (where $C(x)$ is the normalizing factor) actually only depend on the patch value $U(x)$ and not on the pixel position x itself. Therefore we can denote them by $W(U(x),U(y)) = \frac{1}{C(U(x))} e^{-\frac{\|U(x)-U(y)\|^2}{h^2}}$ and use the transfer theorem:

$$NLu(x) = \int_\Omega u(y)w(x,y)dy = \int_{\mathcal{P}} c(P)W(U(x),P)dm(P) \tag{3.2}$$

where c is the application that maps a patch to the value of its central pixel, that is, up to a change of coordinates, the projection to the first axis in the patch space (equivalently, $P^1 = c(P)$).

This formula states that the solution of NLmeans can be computed directly in the patch space provided that given each patch $P \in \mathcal{P}$, one knows exactly how many times $dm(P)$ it appears in the image. Then the value $NLu(x)$ is the first moment along the first axis (Ox_1) of the measure $W(U(x),P)dm(P)$. An image whose density m is locally widely spread along the Ox_1 axis will be considerably modified by the NLmeans algorithm, whereas an image whose density m is concentrated along Ox_1 will be preserved.

REMARK 3.1. *Because of the normalization of the weights, the measure $W(U(x),P)dm(P)$ has total mass 1. The "bias" of the filter can therefore be expressed as:*

$$NLu(x) - u(x) = \int_{\mathcal{P}} \left(c(P) - c(U(x))\right) W(U(x),P)dm(P).$$

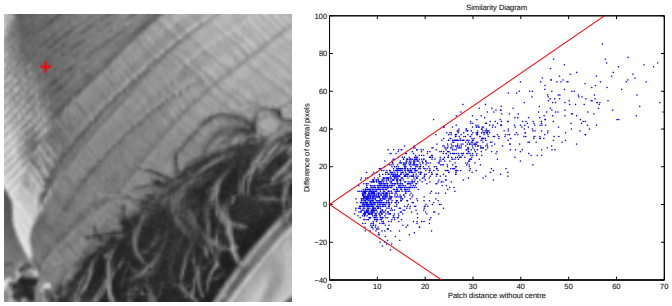


FIG. 3.1. *Example of a similarity diagram. Left: extract of the Lena image (without noise). Right: its similarity diagram at the point x indicated by a red cross: patches are represented depending on their distance to the reference patch (abscissa) and the difference between their central pixel (ordinate). For clarity, the search was limited to a 51×51 window. The assumption "Similar patches have similar central pixels" seems relevant in this case. The bias of the NLmeans filter at x is equal to the weighted-moment of this distribution along the ordinate axis.*

One may reformulate the patch regularity assumption by imposing that the measure m makes the above integral small.

Two sufficient conditions for this are given below:

- (Uniform bound) Assuming that the support of the measure $W(U(x), P)dm(P)$ is contained in the set $\{P \in \mathbb{R}^{s^d}, |P^1 - u(x)| \leq \epsilon\}$, we trivially have $|NLu(x) - u(x)| \leq \epsilon$.
- (Lipschitz bound) A way to allow patches P that are very different from $U(x)$ to have a different central pixel is to merely impose that the support of $W(U(x), P)dm(P)$ is contained in the set $\{P \in \mathbb{R}^{s^d}, |P^1 - c(U(x))| \leq k\|\dot{P} - \dot{U}(x)\|\}$, where $\dot{P} := (P^2, \ldots, P^{s^d})$ denotes the patch excluding its central pixel and $k > 0$ is a constant. This provides the upper bound:

$$|NLu(x) - u(x)| \leq \alpha(k) \int_{\mathcal{P}} \|P - U(x)\| W(U(x), P) dm(P),$$

  for some constant $\alpha(k) > 0$. However, the above integral can be arbitrarily large since the gaussian weights $W(Q, P) = \frac{1}{C(Q)} e^{-\frac{\|Q-P\|^2}{h^2}}$ of the NLmeans never cancel. On the contrary, if we replace them with $W(Q, P) = \frac{1}{C(Q)} \varphi\left(\frac{\|Q-P\|^2}{h^2}\right)$ where φ has compact support ($\varphi(x) = 0$ for $x \geq r > 0$) the above upper-bound becomes $\alpha(k)r$: it is small when k is small.

Similarity diagram. In [23], the authors study the denoising of a step signal using NLmeans with patch size 2 (i.e. the patch is $(u(x), u(x+1))$) from a diffusion point of view. To this end, they represent the set of patches as the couples $(u(y), u(y+1))$ for all y. In view of the above remarks, we use a slightly different representation (see Figure 3.1), taking $U(x)$ for a given x as the origin, and focusing on the Ox_1 axis. When the patch size is more than 2, we simply plot the patch set as the couples: $(\|\dot{P} - \dot{U}(x)\|, c(P) - u(x))$, where again $\dot{P} \in \mathbb{R}^{s^d - 1}$ denotes the patch excluding its central pixel. This representation contains exactly the information NLmeans needs to compute $NLu(x)$, and it allows to see if the estimation of each pixel will be much biased. We shall refer to this representation in terms of distance between patches and difference of central pixels as a *similarity diagram*.

3.2. The bias-variance dilemma. In this subsection, we assume that the variance σ^2 of the noise is small, and that the patch size is large[1]. Thus, if $\tilde{U}(z)$ denotes the patches of the noisy image $\tilde{u} = u + n$, with n i.i.d. gaussian noise, we have[2]: $\|\tilde{U}(x) - \tilde{U}(y)\|^2 \approx \mathbb{E}\|\tilde{U}(x) - \tilde{U}(y)\|^2 = \|U(x) - U(y)\|^2 + 2\sigma^2$. Therefore, we may consider the weights *deterministic*.

Then, the risk of denoising the pixel x is given by:

$$\begin{aligned}\mathbb{E}|NL\tilde{u}(x) - u(x)|^2 = \mathbb{E}|NL\tilde{u}(x) - NLu(x)|^2 + \mathbb{E}|NLu(x) - u(x)|^2 \\ + 2\mathbb{E}\left((NL\tilde{u} - NLu(x))(NLu(x) - u(x))\right).\end{aligned}$$

The last term vanishes, since $\mathbb{E}(NL\tilde{u}(x) - NLu(x)) = \sum_y \mathbb{E}(n(y))w(x,y) = 0$. The first one depends on the noise variance, since :

$$\begin{aligned}\mathbb{E}|NL\tilde{u}(x) - NLu(x)|^2 &= \mathbb{E}\left(\sum_y n(y)w(x,y)\right)^2 \\ &= \sigma^2 \sum_y \left(w(x,y)\right)^2 \\ &= \sigma^2 \int_{\mathcal{P}} \left(W(U(x),P)\right)^2 dm(P) := R_1, \qquad (3.3)\end{aligned}$$

where m is the measure defined in Section 3.1. The second one, $R_2 := |NLu(x) - u(x)|^2$, can be interpreted as a bias term, and it depends on the "patch regularity" of the image. The Cauchy-Schwarz inequality gives an upper bound of it[3]:

$$\begin{aligned}R_2 &:= |NLu(x) - u(x)|^2 \\ &= \left(\int_{\mathcal{P}} (c(P) - c(U(x)))W(U(x),P)dm(P)\right)^2 \qquad (3.4) \\ &\leq \left(\int_{\mathcal{P}} (c(P) - c(U(x)))^2 W(U(x),P)dm(P)\right)\left(\int_{\mathcal{P}} W(U(x),P)dm(P)\right) \\ &\leq \int_{\mathcal{P}} (c(P) - c(U(x)))^2 W(U(x),P)dm(P) \qquad (3.5)\end{aligned}$$

As a consequence, the problem of finding the value h of the parameter that minimizes the risk can be reformulated as finding the value of h that minimizes :

$$R_1 + R_2 = \sigma^2 \int_{\mathcal{P}} \left(W(U(x),P)\right)^2 dm(P) + \left(\int_{\mathcal{P}} (c(P) - c(U(x)))W(U(x),P)dm(P)\right)^2. \qquad (3.6)$$

To interpret this quantity, let us momentarily assume that the weights are not gaussian but given by an indicator function: $w(x,y) = \frac{1}{C(x)}\mathbf{1}_{\|U(x)-U(y)\|^2 \leq h^2}$, where $C(x)$ is an appropriate normalization factor. In the patch space, it can be written as:

---

[1] Roughly, we assume that $\frac{\sigma^2}{s^{d/2}}$ is small compared to the typical square distance $\|U(x)-U(y)\|^2$, which is typically 10^2 as illustrated in Figure 3.7.

[2] This approximation is not valid for $x = y$. However, the same qualitative conclusions can be drawn by slightly adapting the following discussion. We skip this for the sake of clarity.

[3] Notice that the integral of the upper bound (3.5) over the image gives the non-local H^1 norm used in ([12, 16]) in a variational framework.

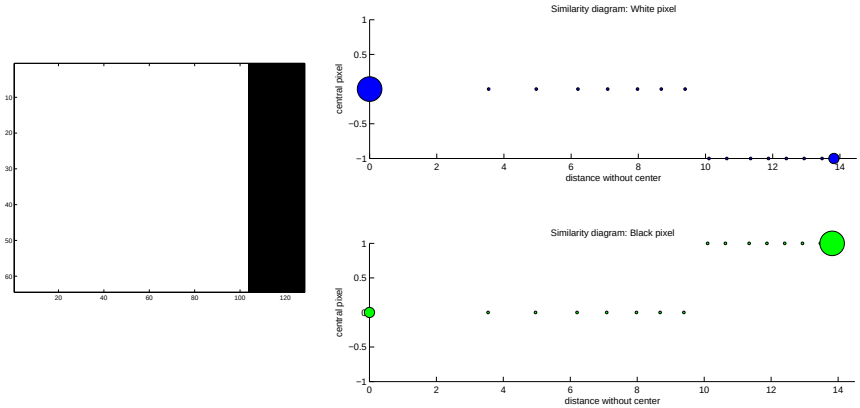


FIG. 3.2. *Left: a synthetic image, with different proportions of black and white. Right: the similarity diagram of this noise-free image for a white pixel (top) and a black pixel (bottom). The size of the disks is proportional to the number of patches at a given distance and difference of central pixels. Once noise is added, it is clear that when denoising white pixels, one may choose a large smoothing parameter h, since the bias is very low. On the contrary, when denoising a black pixel, one cannot reduce the variance of the noise by a large amount since the large bias compels one to use a small smoothing parameter.*

$W(P,Q) = \frac{\mathbf{1}_{B_h(P)}(Q)}{m(B_h(P))}$, where $B_h(P)$ denotes a ball with radius h and center P, and $m(B_h(P))$ denotes its m-measure, i.e. the number of patches within distance h of P. Then, similar computations show that:

$$R_1 + R_2 = \frac{\sigma^2}{m(B_{h'}(U(x)))} + \left(\frac{1}{m(B_{h'}(U(x)))} \int_{B_{h'}(U(x))} (c(P) - c(U(x)))dm(P) \right)^2, \tag{3.7}$$

with $h' = \sqrt{h^2 - 2\sigma^2}$.

The first term is a non-increasing function of h, whereas the second one is not necessarily monotone. However, if the image satisfies the Patch Regularity assumption, it is reasonable to expect that the second term (or at least the upper bound (3.5)) is roughly non-decreasing, since we allow patches that are less similar to have a less similar central pixel (see Figure 3.1). This formula shows that the best choice for h is a trade-off between reducing the variance by taking a large number of pixels in the average, and not averaging pixels that belong to very different patches.

This trade-off is all the easier as the image is patch regular. In our experiments we have noticed that although the Lipschitz condition is often verified, the corresponding constant k varies with the pixel x. The following two examples should convince the reader that the patch regularity is in fact necessarily local.

Example 1. Figure 3.2 revisits the example of Section 2.3 (a step signal/image where the proportions of black and white pixels are different) using similarity diagrams. For a given value of h, the minority pixels suffer more from bias than the majority pixels. Therefore the optimal value to denoise the black pixels is different from the optimal one for the white pixels, as the plotting of the bound on the risk (Eq. 3.6) would show.

Example 2. We may interpret the smoothing of periodic crenels described in Section 2.1. Figure 3.3 shows the similarity diagram of a pixel which lies at the

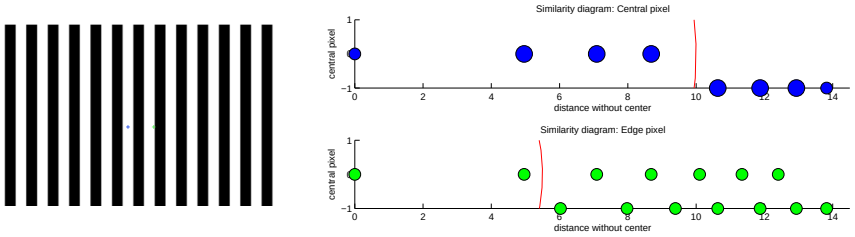


FIG. 3.3. *Similarity diagrams for a crenel image/signal. Left: synthetic crenel image. Right: similarity diagram for a pixel at the middle of a white crenel (top) and at the edge of a crenel (bottom). The size of the blue disks is proportional to the frequency each patch is actually encountered in the image. The image is patch regular at the first pixel, but not at the second one. The red line indicates the boundary of a ball centered at $U(x)$, i.e. a level-line of the weight function $P \mapsto W(U(x), P)$.*

middle of the crenel and another one that lies near the edge. A possible thresholding value of the distance is suggested. It is clear that the first one is easier to denoise than the second one, since in the first case, one may use a large value of h in order to decrease the variance, whereas in the second case any such value would introduce a large bias . In fact, even though the considered signal is periodic, *it is not regular in the sense of NLmeans at the points near the edges*. In dimension 2, it is classical when working with NLmeans to show that the weights adapt to the local geometry of edges and therefore preserve them. However this is not true anymore when working with less contrasted edges or isolated details. In fact, Figure 3.3 advocates for using a different smoothing parameter in regions that suffer from a large bias than in regions where the image is patch regular.

REMARK 3.2. *The expression in (3.6) depends on the measure m related to the noise free image. However one may wonder how this measure is modified by the noise. When the noise variance is small, we expect that the "noisy" measure $\tilde{m}$ is not very different from m. Indeed, by Campbell's formula (or simply Fubini's theorem), the "expected measure" on $\mathcal{P}$ has a density with respect to the Lebesgue measure given by the convolution of the probability density function of the noise and the measure m. In other words, for all continuous bounded function f on $\mathbb{R}^{s^d}$:*

$$\mathbb{E}\left(\int_{\mathcal{P}} f(P)d\tilde{m}(P)\right) = \int_{\mathbb{R}^{s^d}} f(Y)\left(\int_{\mathcal{P}} g(Y-Z)dm(Z)\right)dY$$

where g is, in our case, a gaussian density with covariance matrix $\sigma^2 I_{s^d}$.

3.3. Bias and the size of the search window. We discussed in Section 2.2 the fact that the size of the search window has an impact on the final result. Using a small search window is the common use and the reason invoked is that, besides the speed-up, images are not really non-local. In this subsection we relate an experiment that precises this idea.

We run the NLmeans filter on several images degraded by a noise with $\sigma = 10$, and look at the evolution of the PSNR when the size of the search window increases. However, we truncate the weights in the following way:

$$w(x,y) = \begin{cases} \frac{1}{C(x)} e^{-\frac{\|\tilde{U}(x)-\tilde{U}(y)\|^2}{2h^2}} \text{ if } \|\tilde{U}(x)-\tilde{U}(y)\| \leq T, \\ 0 \text{ otherwise.} \end{cases} \tag{3.8}$$

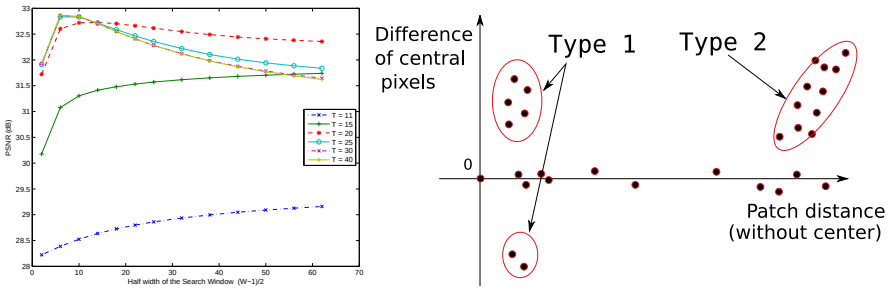


FIG. 3.4. *Left: Evolution of the PSNR as the size of the search window increases, for different values of the threshold T, on the boat image. Using a good threshold (e.g. $T = 20$) for the weights makes the algorithm more robust to changes of the search window. The parameter are: $h = 10$, $s = 7$. The noise level is $\sigma = 10$. Right: the different kinds of irrelevant pixels in a similarity diagram.*

where $C(x)$ is, as usual, a normalizing factor. Since the patch distance is normalized, for a threshold value $T = 255$, the algorithm is equivalent to the usual NLmeans algorithm. The parameter h is set to 10, which gives good visual results when the size of the search window is 25×25.

The typical curves for several values of T are shown in Figure 3.4. The PSNR first increases as the size W of the search window increases, since on very small neighborhoods all the pixels are relevant to denoise a pixel x, so the more pixels the better. When W gets large enough, many pixels in the search window are irrelevant to denoise x and the PSNR drops.

There are *two different kinds of irrelevant pixels* when applying the NLmeans filter at a pixel x (see Figure 3.4). *First*, pixels that belong to patches that are similar but have very different central values ($\|U(x) - U(y)\|$ is small but $|u(x) - u(y)|$ is large): these patches are cause of non patch regularity. We have seen that this kind of pixels arise for instance at edges. *Second*, pixels that belong to very different patches ($\|U(x) - U(y)\|$ is large). These pixels have a small weight but, as in Section 2.2, it is non-zero. Contrary to the pixels of the first type, we can get rid of them by truncating the weights as in (3.8).

The interesting point in this experiment is that when imposing a small threshold value T, the filter is almost insensitive to the increase of the search window. This shows that the pixels of the first kind are not prominent when the search window increases, and therefore the loss of PSNR without thresholding is due to the bias induced by pixels of the second kind. As a consequence, there is a gain in using a small search window, not because images are not patch regular, but because the proportion of irrelevant patches of the second kind becomes overwhelming when the search window gets larger.

As a consequence, this advocates for the use of truncated weights rather than gaussian weights. In [13], it is proposed to use weight functions with compact support and the authors show that this allows to preserve textures better (which can also be understood in light of Section 2.2). Let us stress the fact that it also makes the algorithm more robust to the choice of the search window. Therefore, when using weight functions with compact support, the interest of using a reduced search window should mainly be computational rather than for the quality of the result.

Of course, the question is how to choose the thresholding parameter T, or the smoothing parameter h when using a function with compact support. This point is discussed in Section 4.

3.4. The patch size dilemma. A last issue is the choice of the patch size. Defining a patch size amounts to choosing a patch space, which determines the similarities, and therefore the optimal smoothing parameter h. It is remarkable that, in the literature ([26, 29, 17]), the best results with strong noise are obtained with large patches. As noted in [23], using a large patch allows a more robust discrimination between areas that are not actually similar, which is interesting in the presence of noise. Let us borrow and reformulate their argument in an experiment illustrated in Figure 3.5. We consider an image with two regions (with intensities α and 0) and a noise with standard deviation σ such that the two noisy regions are hard to distinguish. In a similarity diagram, the black and grey regions cannot be discriminated with a 3×3 patch, but with size 15×15 one clearly sees two different clouds. Indeed, let P be a perfect grey patch and $\tilde{U}(x)$ be a noisy patch completely included in the grey region. By the Bienaymé-Tchebychev inequality:

$$\forall \epsilon > 0,\ \mathbb{P}\left(\left|\|\tilde{U}(x) - P\|^2 - \sigma^2\right| > \epsilon\right) \leq \frac{2\sigma^4}{s^d \epsilon^2},$$

so that for a large patch size s^d, most patches of the grey regions lie near the sphere of radius σ and center P. Similarly the patches of the black regions are centered around the perfect black patch Q, and the proportion of patches that mislead NLmeans is approximately the proportion of points on each sphere that are closer to the center of the other sphere. Now, that proportion vanishes as $s \to +\infty$. Indeed, it is given by $\frac{\int_{\sigma-\epsilon}^{\sigma+\epsilon} r^{s^d-1} \int_0^{\theta_0(r)} (\sin\theta)^{s^d-2} d\theta dr A_{s^d-2}}{\int_{\sigma-\epsilon}^{\sigma+\epsilon} r^{s^d-1} \int_0^{\pi} (\sin\theta)^{s^d-2} d\theta dr A_{s^d-2}}$, where $\cos\theta_0(r) = \frac{\alpha}{2r}$ (see Figure 3.7) and A_{s^d-2} is the area of the unit sphere in $\mathbb{R}^{s^d-1}$. Taking $\epsilon = s^{-d/4}$, and noticing that $\theta_0(r) \leq \theta_0(\sigma + \epsilon) < \frac{\pi}{2}$, one sees that it vanishes as $s \to +\infty$: for large s, most patches are closer to the center of their own sphere. Thus, the NLmeans algorithm is better able to distinguish the two regions for a large patch size

This example should encourage us to use large patches for their robustness to noise. An example is shown in Figure 3.7 where a small patch size induces a mottling effect. However, two objections should temper this conclusion. First, as explained in Section 2.2 the other side of the coin is that using a large patch size reduces the importance of little contrasted small details, so that they are more blurred. Second, if the image has textures with highly contrasted transitions or which are not exactly repeated, using a too large patch will prevent the algorithm from finding redundancies, as Figures 3.8 and 3.9 show.

A way to overcome both issues is to select the smoothing parameter h accordingly: when choosing the useful patches, be more selective in the first case and more tolerant in the second one. These two contradictory behaviors advocate for a local choice of h again.

In a nutshell, the patch size should ideally be chosen depending on the local scale of the signal/image, but a locally defined h should make this choice less critical. In the next section we propose to set both the smoothing parameter and the patch size automatically.

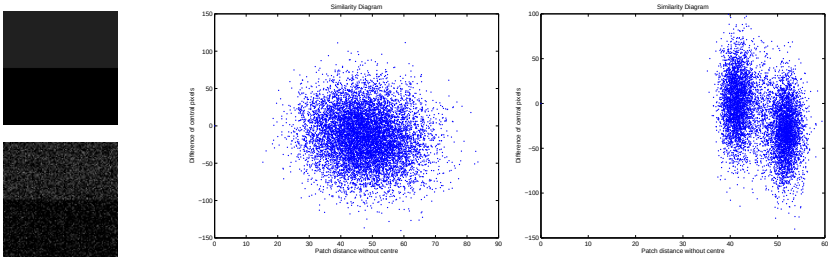


FIG. 3.5. *Left: original (intensity* 0 *and* $\alpha = 32$*) and noisy image* ($\sigma = 30$*). Middle and right: similarity diagram for a pixel at the center of the grey region (the patch size is respectively* 3×3 *and* 15×15*). With a large patch size one sees two clusters. Question: Does the closest cluster correspond to patches which have the same original intensity as* $U(x)$*? When* s *is large, the answer is yes with high probability: in strong noise, a large patch size discriminates better the two regions than a small one.*

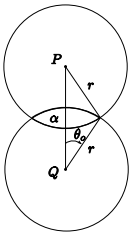


FIG. 3.6. *Two spheres in the patch space centered around the grey patch* P *and the black patch* Q*. The patches on each sphere that lie closer to the center of the other one are represented in bold: for them, the closest cluster is the one that comes from a different original intensity than theirs. The maximum angle* $\theta_0(r)$ *satisfies* $\cos\theta_0 = \frac{\alpha}{2r}$*. The proportion of bold patches vanishes as* $s \to +\infty$*.*

FIG. 3.7. *Left and right: extract of the result of the NLmeans filter with* $s^2 = 3 \times 3$ *and* $s^2 = 5 \times 5$ *on a noisy image* ($\sigma = 10$*,* h *optimized for PSNR). Using a too small patch size makes the algorithm less robust to noise and Lena's skin looks mottled. It looks smoother with patch size* 5×5 *but visual artifacts appear in the eye.*

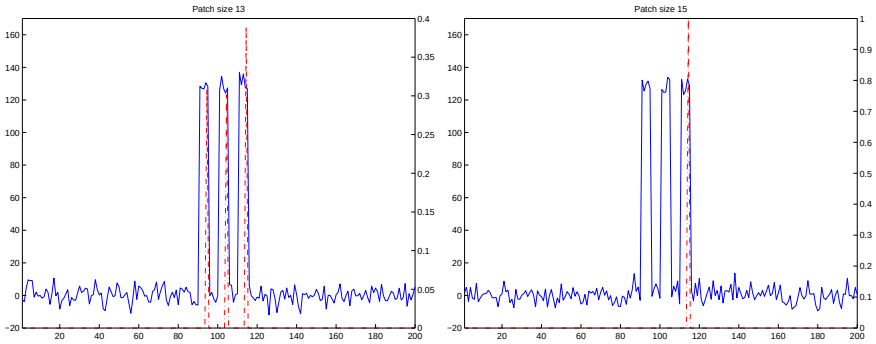
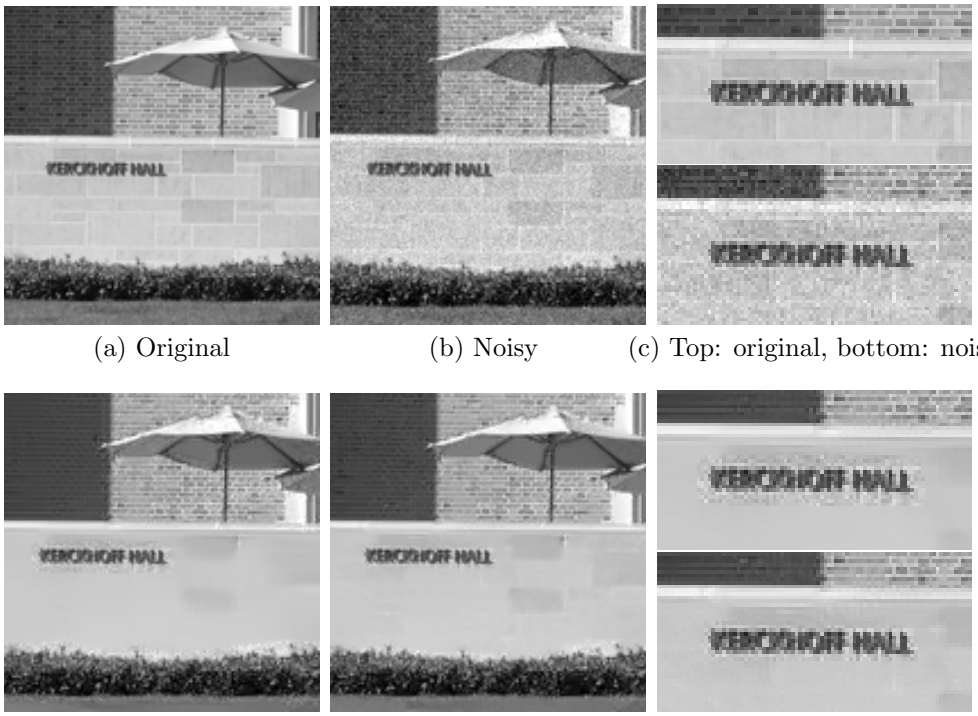


FIG. 3.8. *Choice of the patch size: a one-dimensional example. In solid blue line, a noisy signal given as an input to the NLmeans filter; in dashed red line, the weights related to the pixel indicated by the largest peak. On the left, a small patch size (13) allows to find the pixels that are useful to denoise. Right: the patch size is too large (15), no similar patch can be found because of the transition of texture.*



(a) Original (b) Noisy (c) Top: original, bottom: noisy

(d) Patch size 9×9 (e) Patch size 5×5 (f) Top: 9×9, bottom: 5×5

FIG. 3.9. *Choice of the patch size (bis): original image (a), noisy image (b) ($\sigma = 10$), result of the NLmeans filter with patch size $s^2 = 9 \times 9$ (d), and $s^2 = 5 \times 5$ (e). Zoomed versions are shown in (c) and (f). The search window is $W = 29 \times 29$, in each case the smoothing parameter h was chosen to maximize the PSNR. Notice that around the letters it is very difficult to find similar patches, so that a noisy halo appears. Using a smaller patch size reduces the spread of the halo since it allows to find similar patches for the furthest pixels. Another solution would have been to force a high smoothing parameter.*

4. Making the Non-Local Means spatially adaptive. At this point of the discussion, let us sum up the main conclusions drawn above :

1. The bias of the Non-local means filter should not be neglected, even on periodic signals.
2. It is determined by the patch regularity, a property that is mostly true in natural images, but not everywhere (e.g. near edges).
3. Although images generally satisfy the patch regularity assumption, increasing the size of the search window leads to a loss of PSNR because the non relevant patches become overwhelming, and they have a positive weight.
4. The impact of the non relevant patches on details is all the more important as the details are small compared to the patch size.
5. This loss can be reduced with the use of truncated weights.
6. Large patch sizes make the algorithm more robust to noise.
7. The choice of the patch size determines the similarity diagram at each pixel, which itself determines the bias and therefore the optimal value of h. There is no clear correlation between the patch size and the optimal smoothing parameter h.

It is therefore natural to choose the parameters (at least h, and probably s) locally, and to use weights with compact support. Let us stress the fact that the interest of using truncated weights to reduce the bias was also noticed in [13].

This last section is divided in two parts, both dealing with the bias-variance dilemma described in Section 3. First, we use an oracle to minimize the risk locally: this gives the main guidelines on what an adaptive method should do and what it can provide. Then we approximate the oracle using a SURE estimation of the risk: we propose an algorithm which can locally set h by minimizing the risk and which inherits the properties of the oracle.

In order to use weights with compact support (on top of using gaussian weights) we write the weights as $w(x,y) = \frac{\varphi(\frac{\|U(x)-U(y)\|^2}{2h^2})}{C(x)}$, where $C(x)$ is the normalizing factor. We have tried several functions φ, among which we have not noticed much difference provided that they have compact support. In the following, we will essentially refer to the following functions:

$$\varphi_0(x) = \mathbf{1}_{[0,1]}(x) = \begin{cases} 1 & \text{if } 0 \leq x \leq 1 \\ 0 & \text{otherwise.} \end{cases},$$
$$\varphi_1(x) = \exp(-x) \text{ (classical Non-local means weights)},$$
$$\varphi_2(x) = \mathbf{1}_{[0,1]}(x)(1-x)^2 \text{(bisquare weights)},$$
$$\varphi_3(x) = \mathbf{1}_{[0,1]}(x)\left(1 - (10x^6 - 24x^5 + 15x^4)\right) \text{(spline weights)}.$$

The first function φ_0 is an indicator function used in the next experiment, while φ_1 corresponds to the classical NLmeans. Function φ_2 was proposed in [13]. We introduce the function φ_3 as a C^2 approximation of an indicator function, which is necessary for the gradient descent exposed in Appendix A if one wants to quickly select a global parameter.

4.1. Oracle estimation. As an application of the bias-variance considerations of Section 3, we design an oracle which minimizes the risk (3.7). Of course in practice we do not have access to the noise free image to estimate the risk, but this experiment

illustrates the potential benefits of locally adapting the parameters: it sheds light on what can be expected in terms of visual quality and what parameters should be used in which region.

To this end, we use the indicator weights associated with φ_0 instead of the gaussian weights for the following reasons. First, we would like to have weights with compact support, as explained earlier. Second, the risk (3.7) can be computed faster than with other weights using an induction. Last but not least, these weights do not suffer from the overestimation of the "self-weights" $w(x,x)$ pointed out in [21]. Let us recall that for $y \neq x$, the distance between noisy patches $\|\tilde{U}(x)-\tilde{U}(y)\|^2 \approx \|U(x)-U(y)\|^2+2\sigma^2$ is increased by the noise level, whereas the "self-distance" $\|\tilde{U}(x)-\tilde{U}(x)\|^2$ is always zero. Therefore, with gaussian weights, the weight $w(x,x) = \frac{1}{C(x)}$ is proportionally $e^{\frac{2\sigma^2}{2h^2}}$ times larger in presence of noise than it would be without noise. As a consequence, some authors set $w(x,x)$ to $\frac{1}{C(x)}e^{-\frac{\delta^2}{2h^2}}$, where $\delta^2 = \min_{y\neq x}\|\tilde{U}(x)-\tilde{U}(y)\|^2$, or replace it with $\frac{1}{C(x)}e^{-\frac{\sigma^2}{h^2}}$ (see [21]). On the contrary, the indicator weights do not behave this way, and we need not give a special value to $w(x,x)$.

The filter has therefore the following form:

$$NLIu(x) = \frac{\sum_{y\in\Omega} \mathbf{1}_{\frac{\|U(x)-U(y)\|^2}{2}\leq h^2} u(y)}{\sum_{y\in\Omega} \mathbf{1}_{\frac{\|U(x)-U(y)\|^2}{2}\leq h^2}} \tag{4.1}$$

The expression (3.7), which gives (an approximation of) the risk, actually only depends on $m(B_{h'}(U(x)))$, the number of pixels that are taken into account in the estimation of x, rather than on h' itself. Thus, minimizing this bound over the radius h' amounts to finding the optimal number of pixels $n_x := m(B_{h'}(U(x)))$ when denoising x. The computation of this bound for each integer can be performed very fast using an induction.

To build our oracle estimate, we compute the risk on the *noise free image* u for each integer, and we keep the minimizing value n_x. Then we can estimate $u(x)$ from $\tilde{u}$ by averaging the centers of the n_x patches $\tilde{U}(y)$ that are the nearest to $\tilde{U}(x)$ in euclidean distance. In a nutshell, we use the oracle to define a map n_x and then compute the non-local filter on the noisy image $\tilde{u}$, keeping only the best n_x patches. Therefore, contrary to [4], we denoise each pixel using a locally varying number of pixels n over the image.

The obtained visual results will be discussed in the next Section and compared with a local SURE estimation. For now, let us examine some properties of the oracle.

Figure 4.1 shows the number of pixels n_x recommended by the oracle and the associated smoothing parameter $\sqrt{2}h_x$ (that is, we display the norm $\|U(x)-U(y)\|$ where y is the last pixel taken into account). We limited the maximum number of pixels in the average to $n_x^{max} = 3600$. As one would have expected, in very smooth regions the oracle advises us to use as many pixels as possible whereas in regions where the image is not patch regular (i.e. near edges) the oracle recommends to use very few pixels. The case of textures is in-between. A bit more surprising is the map of the corresponding h_x: the values prescribed near the edges are much higher than in smooth regions or texture regions. In fact, even though the oracle prescribes very few pixels to reduce the variance term, one has to go very far in the patch space to gather enough pixels. This is illustrated in Figure 4.2 (notice that here these similarity diagrams are computed on the noisy image). Therefore, *one should use*

much higher values of h near edges. Let us stress again the fact that this problem is *not* related with the overestimation of the self-weight mentioned above, since the indicator functions do not suffer from this drawback.

Figure 4.4 shows the PSNR as a function of the size of search window (to be compared with Figure 3.4). This time, the global trend of the PSNR is non decreasing with the size of the search window. This suggests that there is no harm in computing *fully non-local* means, provided that we choose well the pixels in the means[4]. At worst, the PSNR tends to stabilize for a side-length W greater than 25, in which case the extra-computations due to a fully non local filter are not justified. In images which contain large and smooth regions, the PSNR is, strictly speaking, increasing, because there is always interest in finding more pixels to reduce the variance. In the general case, one may observe slight oscillations of the PSNR: when increasing the search window, one adds to the mean several relevant pixels, but also a few pixels of the first kind which perturb the estimation of $u(x)$. However they are very few, so that on the overall the PSNR is stabilized. With a local smoothing parameter h, (or n_x), the filter is therefore more robust to the choice of W than it is with a global one, so that in the following, we fix W in advance and then we choose locally h. This approach is dual to the one proposed by Kervrann et al. [14] who fix h and then control the bias and variance of the filter by choosing the size of the search window.

To sum up, adaptivity allows to deal with pixels that have very few similar patches (like edges) by increasing the smoothing parameter and makes the filter more robust to the choice of the search window.

REMARK 4.1. *There is apparently a contradiction between the idea of adapting the smoothing parameter h (whether globally or locally) to the content of the image and the observation of several authors (e.g. [6, 26, 29]) that h should be set proportional to the noise standard deviation σ. Figure 4.3 shows that the relation between the optimal global parameter h and σ is indeed approximately linear and it is remarkable that the slope does not vary much between different images.*

On the one hand, this linear relation is so flagrant, that there is probably a deep reason for it. On the other, the maps displayed on Figures 4.1 and 4.2 clearly illustrate that the optimal value of h depends on the local content of the image. Moreover, the uncertainty on the optimal h predicted by the linear relation is not negligible: for a fixed noise level, swapping the optimal values for two different images may induce a noticeable visual difference. As a consequence, this empirical rule only gives a rough idea of the optimal value (that can be refined by the SURE estimation, see [29] or below).

How can we reconcile the two points of view? A way to derive a linear relation between h and σ is to use a χ^2 test (see for instance [14]) to decide if two patches are replicas: for instance, one chooses the smallest h such that 99% of the replicas contaminated by the noise are accepted. We presume that this framework suits well the smooth dominant regions which have a strong influence on the behavior of the PSNR. Indeed, there are some replicas (or almost replicas) of each patch. However, along winding edges or in highly oscillating regions (see Figure 4.1), one can never find exact replicas, and we are bound to average patches that are not too different with the hope that the central pixels are close to one another. This is the meaning of the bias-variance approach of this paper.

[4]Put it this way, this statement looks as a tautology, but one should remember that the expression (4.1) imposes a structure on the choice of the pixels, and it would not be true if pixels of the first kind (see Section 3.3) were overwhelming in natural images.

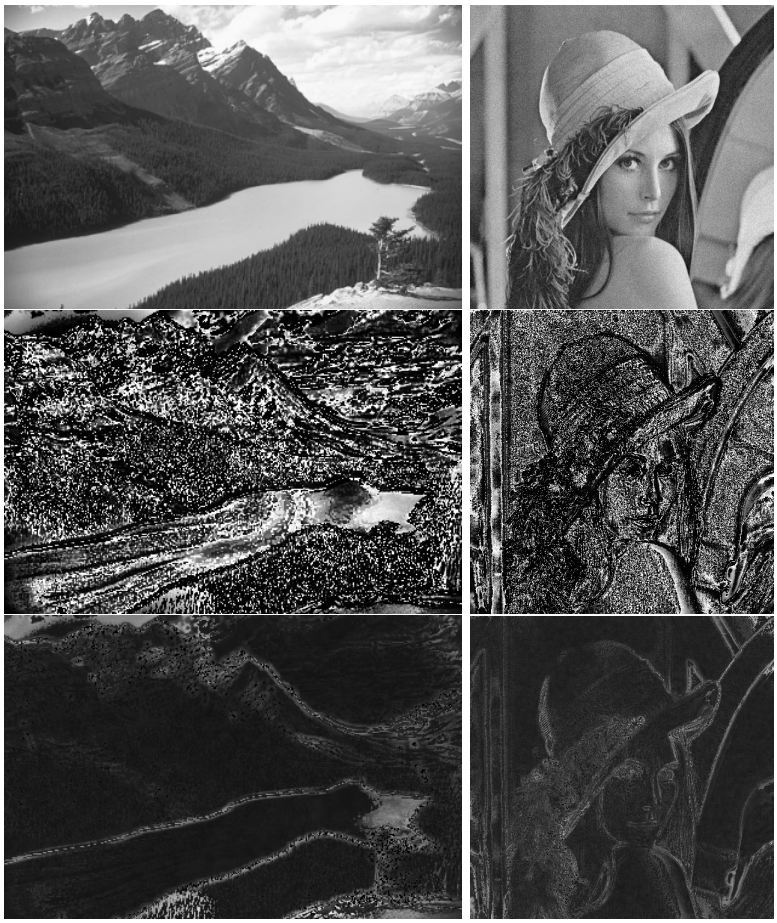
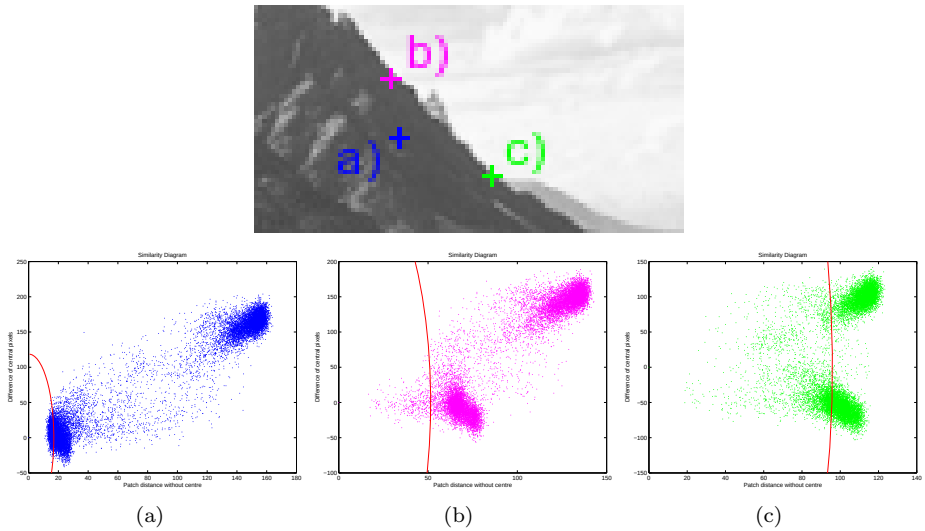


FIG. 4.1. *Map of parameters prescribed by the oracle. The original images were degraded with* $\sigma = 10$*, the patch size is* $s = 7$*. From top to bottom: original images (without noise), map of the number of pixels in the mean, map of the corresponding h parameter. In the second row, the white regions represent a number of pixels* $n_x \approx 3000$*. In the third row, first image, the parameter h is approximately equal to 14 on the lake, while it ranges from 15 to 20 in the forest, and from 40 to 75 along the edges of the mountains and the lake. On the rocks it is around 30. Although the denoising of edges should be performed with very few pixels, the corresponding parameter h should be very large.*

4.2. SURE. We have not managed to approximate the oracle for indicator weights with a practical estimator. Possible explanations are the fact that the estimation using indicator weights is in general not robust, or the fact that, in Equation (3.6), all the computations have been drawn assuming that the weights are deterministic, which is arguable. A way to handle the bias-variance dilemma without making this assumption is to rely on the SURE estimation, which is well-known in the wavelet community. After we derived the results of Section 4.2.1, we discovered the recent work of [29] where the SURE estimation of the risk is derived for the classical NLmeans algorithm.

4.2.1. Estimation of the risk. Let us first recall the result by Stein (see [24]):



(a) (b) (c)

FIG. 4.2. *Similarity diagram for three different pixels. The red line indicates the ball of radius h_x prescribed by the oracle (the pixels at the right of this line are not taken into account into the mean.). In the interior of homogeneous regions (a), a small radius ($h_x \approx 17$) is sufficient to reduce the variance of the noise. Near edges (b and c), one has to look very far in the patch space to find enough pixels to reduce the variance ($h_x \approx 50$ and 95 respectively). The threshold is especially large in the third case since the compensation of the darker and brighter pixels yields a very small bias.*

FIG. 4.3. *Left: Evolution of the optimal global smoothing parameter h when the noise level σ varies, for six different images. The linear behavior is clear. Middle and right: visual difference between the optimal smoothing parameter for this specific image ($h = 14.5$, $PSNR = 30,84dB$, only an extract is shown) and the optimal one for another image ($h = 16$, $PSNR = 30,63dB$): some of the texture was lost.*

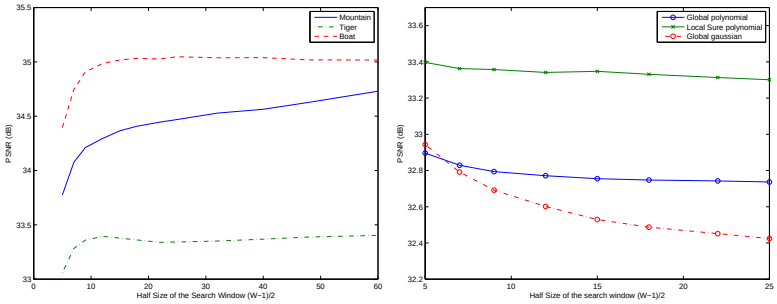


FIG. 4.4. *Evolution of the PSNR as the size of the search window W increases. Left: for the oracle with indicator weights on different images. The patch size is $s = 7$, the noise level is $\sigma = 10$. With a locally defined smoothing parameter h, there is no or very little loss when using a large window, so that the choice of W is not a real issue, contrary to Figure 3.4. Right: for the Local SURE filter on the image "mountain". The local SURE filter (solid green line), the NLmeans with polynomial weight (solid blue line) and the classical NLmeans (dashed red line) are displayed (for the last two filters, the parameter h was optimized for PSNR for each size of the search window). The local SURE filter is robust to the size of the search window, mainly because of the compact support of the weights.*

PROPOSITION 4.2 (Stein). *Let $x \in \mathbb{R}$, $Z \sim \mathcal{N}(0, \sigma^2)$, and $Y = x + Z$. If $\gamma : \mathbb{R} \to \mathbb{R}$ is absolutely continuous, and*

i) $\lim_{|z|\to\infty} \gamma(x+z)e^{-\frac{z^2}{2\sigma^2}} = 0$,

ii) $\mathbb{E}(\gamma(x+Z))^2 < +\infty$ *and* $\mathbb{E}|\gamma'(x+Z)| < +\infty$,

then the risk of the estimator $Y + \gamma(Y)$ of x is given by:

$$\mathbb{E}|Y + \gamma(Y) - x|^2 = \mathbb{E}\left(\sigma^2 + 2\sigma^2\gamma'(Y) + \gamma(Y)^2\right) \tag{4.2}$$

The proof relies on an integration by parts. Let us denote by $\tilde{u}$ the noisy image, $NL\tilde{u}$ the result of NLmeans applied to the noisy image using the noisy weights, and z the noise at pixel x (i.e. $\tilde{u}(x) = u(x) + z$). Therefore, if we set $\gamma(\tilde{u}(x)) = NL\tilde{u}(x) - \tilde{u}(x)$ we see that:

$$J(x) := -\sigma^2 + 2\sigma^2\left(\frac{d}{dz}NL\tilde{u}(x)\right) + (NL\tilde{u}(x) - \tilde{u}(x))^2 \tag{4.3}$$

is an unbiased estimator of the risk at pixel x, and in [29], an analytic expression of J is given in the case of the gaussian weights. The authors show that this estimator yields a very robust estimation of the global mean square error, once we have computed the solution of NLmeans.

In this paper, since we want to use weights defined by functions φ_i with compact support instead, we compute the general expression of the middle term. To use a more compact notation, the pixel x we want to denoise being fixed, we write:

$$\alpha := \frac{1}{h^2},\ \varphi_y := \varphi\left(\frac{\|\tilde{U}(x) - \tilde{U}(y)\|^2}{2h^2}\right) = \varphi\left(\alpha\frac{\|\tilde{U}(x) - \tilde{U}(y)\|^2}{2}\right),\ \text{and}\ C := C(x) = \sum_y \varphi_y.$$

Then, we have :

$$\frac{d}{dz}\left(\sum_y \tilde{u}(y)\frac{\varphi_y}{C}\right) = \frac{\varphi(0)}{C} + \frac{1}{C}\sum_y \tilde{u}(y)\frac{\partial\varphi_y}{\partial z} - \frac{1}{C^2}\left(\sum_y \tilde{u}(y)\varphi_y\right)\left(\sum_w \frac{\partial\varphi_w}{\partial z}\right). \tag{4.4}$$

The first term comes from $\tilde{u}(x) = u(x) + z$, whereas $\frac{\partial}{\partial z}(\tilde{u}(y)) = 0$ for $y \neq x$. The derivative of φ_y is given by:

$$\frac{\partial\varphi_y}{\partial z} = \frac{\alpha}{s^d}(\tilde{u}(x) - \tilde{u}(y) - \underbrace{(\tilde{u}(x+i_0) - \tilde{u}(x))}_{\text{if } \exists i_0, \in P,\ x = y + i_0})\varphi'_y$$

where the last two terms only appear when y belongs to the patch centered at x (i.e. $|x-y|_\infty \leq \frac{s-1}{2}$). Plugging this expression in (4.3), we get an estimation of the risk for the Non-Local Means with weights defined by φ. As with the gaussian weights, this is a very reliable estimation of the (global) mean square error when it is summed over all pixel x in the image. Notice that this expression can be computed simultaneously with the NLmeans filter without changing its complexity. In Appendix A, we propose a fast way to set the global parameter h using a gradient descent on the risk estimate.

4.2.2. Local Approach. Let us recall that we are interested in computing local parameters, so that, as in Section 3.2, we want to minimize the *local* risk $R_1(x)+R_2(x)$ depending on the local content of the image (textured areas, smooth regions, ...). The SURE estimation allows us to do this *without assuming that the weights are deterministic*. The algorithm described here should not be regarded as a whole new filter but simply as a way to set the parameters of the NLmeans.

Obviously, the pointwise estimation of the risk is not robust, so that we need to locally average the estimations. Since images are not stationary, we should find the right balance between having enough samples to estimate the risk, and keeping a local estimation. Informally, it is not absurd to average the risk even in textured areas since the risk should be very high in the whole region, but we should avoid to mix the risk of areas of different kinds (smooth/textured).

In a local framework, if we use a gradient descent as in Appendix A, the values of h at each pixel evolve independently, so that locally averaging the risk makes here little sense. Therefore we consider a set of values $\{h_1, h_2, \dots h_n\}$ for the smoothing parameter, and for each value we compute the output of the NLmeans filter $(NL_{h_i}\tilde{u})_{i=1..n}$ and the associated risk map $(J_{h_i})_{i=1..n}$. We convolve each risk map J_{h_i} with a disk indicator or a gaussian with small radius to have a more robust estimation of the local risk. Then we choose for each pixel x the value $h_i(x)$ that minimizes the convolved risk at pixel x, $J_{h_i}(x)$, and we retain the corresponding estimation $NL_{h_i(x)}\tilde{u}(x)$.

Implementation. In our experiment, we have used about $n = 60$ values of h, the set $\{h_1, h_2, \dots h_n\}$ depending on the kind of weights we use and the noise level. For instance, using spline weights, we take $h = 3 : 0.5 : 35$ for $\sigma = 10$. This is of course very robust to the change of images, and a good heuristic is to increase h proportionally when the noise level increases (e.g. $h = 9 : 1.5 : 105$ for $\sigma = 30$). On first thought, it looks like we need to compute 60 NLmeans filters. Even if it were the case, several iterative methods proposed in the literature also have to perform similar numbers of NLmeans filters, which proves that this is achievable. However, our method is *not* iterative and a few simplifications appear: the expensive computations of the patch distances needs be performed only once (since all the filters work with the same input image), moreover the other computations are independent, they can therefore be parallelized. To sum up, we propose the procedure described as Algorithm 1 below.

The *for* loops at lines 4 and 11 can easily be vectorized, so that, well implemented, this algorithm should not be much slower to run than the regular NLmeans. To execute lines from 1 to 15 (the rest is negligible) on a 256×384 image using search window W of 23×23 and 64 values of the h parameter, our code takes 26 s with the spline weights. And there is room for acceleration. The computer we use is equipped with an Intel Core2 Duo 2.5GHz and 4Gb RAM. Our C code uses SSE instructions to perform the same operations on four floats at a time but runs only on one of the two cores. Moreover our implementation does not take advantage of the fact that for each pixel, if a weight is zero for some value h_1, it is necessarily zero for all $h_2 \leq h_1$. More generally, several approaches proposed in the literature (like the integrated sums of squares to compute the patch distances in [8], or the cluster tree in [5] to accelerate the NLmeans) could be adapted.

The last point is to decide how many samples we should use to estimate the local risk (this corresponds to line 17 in the algorithm: convolution of the risk with a gaussian or a disk indicator). We propose to set $r = 14 \times \frac{\sigma}{10}$. This value is justified heuristically in Appendix B.

REMARK 4.3. *Instead of convolving the risk for each value of h, another way*

Algorithm 1 NLmeans with Local Sure

for each pixel x **do**
  for each translation $k \in \mathbb{Z}^d$, $|k|_\infty \leq \frac{W-1}{2}$ **do**
    $\text{dist} \leftarrow \frac{\|U(x)-U(x+k)\|^2}{2}$
    for i=1 to n **do**
      $(\sum u\varphi)_i \leftarrow (\sum u\varphi)_i + u(x+k)\varphi(\frac{\text{dist}}{h_i^2})$
      $(\sum \varphi)_i \leftarrow (\sum \varphi)_i + \varphi(\frac{\text{dist}}{h_i^2})$
      $(\sum u\varphi')_i \leftarrow (\sum u\varphi')_i + u(x+k)\varphi'(\frac{\text{dist}}{h_i^2})$
      $(\sum \varphi')_i \leftarrow (\sum \varphi')_i + \varphi'(\frac{\text{dist}}{h_i^2})$
    end for
  end for
  for i=1 to n **do**
    $NL_i u(x) \leftarrow (\sum u\varphi)_i \,/\, (\sum \varphi)_i$
    $J_i(x) \leftarrow \ldots$ (Equation 4.3)
  end for
end for
for i=1 to n **do**
  $J_i \leftarrow J_i * G_r$
end for
for each pixel x **do**
  $i(x) \leftarrow \arg\min_i J_i(x)$
  $NLSure(x) \leftarrow NL_{i(x)}u(x)$
end for

to average it would be to reuse the weights computed by the NLmeans filter with the corresponding value of h and perform a non-local filtering of the risk. The idea is that the estimation of the risk given in (4.3) depends on the patch (and its similarity with the others) rather than the spatial position of the pixel. This method provides sharper maps of h than with the convolution method. However, there is hardly any visual difference in the corresponding denoised images. Since it doubles the computation time, we have not retained this approach.

4.3. Experimental Results. In this section, we illustrate the differences between the NLmeans filter with optimal global parameter and the NLmeans with local parameter using SURE. Notice that by optimal global parameter, we mean that we have retained the value of h that minimizes the true MSE rather than the SURE estimator for the whole image (which would yield very little difference however). The indicator oracle (Section 4.1) is also shown. Notice that, to be coherent with the whole paper, the NLmeans filter we use here is rigorously the one described by (1.2) when replacing the exponential with a function φ described above[5]. Of course, in the following, the local and global versions share the same values for the parameters that

[5]Precisely, we do not replace the self weight $w(x,x)$ by the maximum of the weights $w(x,y)$. This would slightly reduce the rare patch effect described here but it would not solve it, because this effect is due to the configuration of the patch cloud (see Figure 4.2) and even the indicator weights suffer from it. This trick would also favor the loss of details. Moreover, its relevance is questioned in [21]. However, let us mention that the estimation of the risk (4.3) could easily be modified to take this trick or the one proposed in [21] into account.

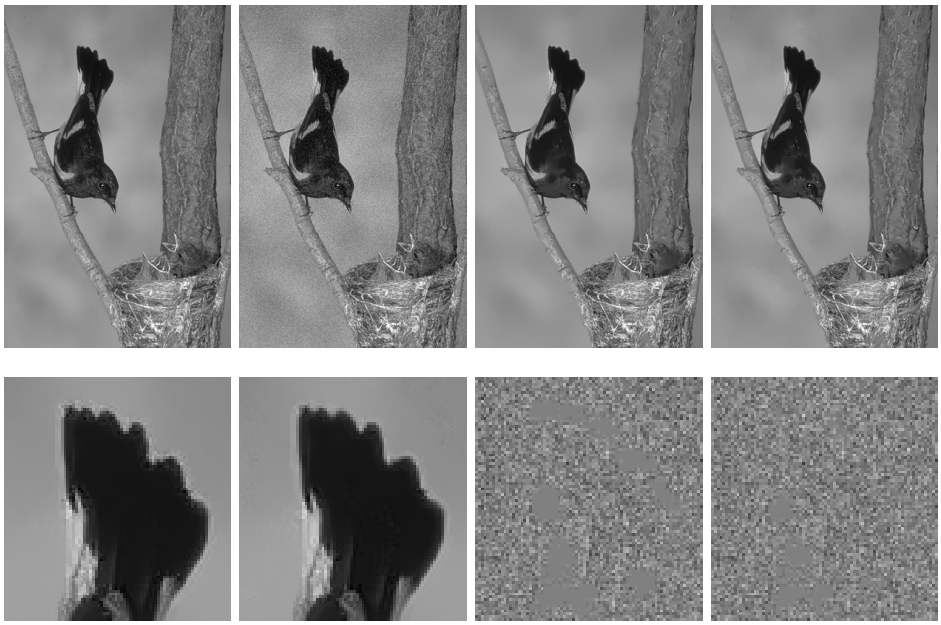


FIG. 4.5. *Top, from left to right: Original image, noisy image ($\sigma = 10$), NLmeans with global smoothing parameter h optimized for PSNR (32.65 dB), NLmeans with local h prescribed by SURE (32.96 dB). Bottom: zoom of the NLmeans with global h, local h , and their respective method noise ($\tilde{u} - NL\tilde{u}$). Along contrasted edges, the global NLmeans leaves a noisy halo. This "rare patch effect" is all the stronger as the edge is winding. The local choice of the smoothing parameter corrects this shortcoming.*

are not locally estimated. Unless otherwise stated, the size of the search window is set to 29×29, the patch size is 7×7, and the weights are spline-based ($\varphi = \varphi_3$). Notice that the use of compactly supported weights already gives a better result than the original gaussian ones, as explained in [13].

Figure 4.5 shows a comparison between the NLmeans with the best global parameter and the NLmeans filter with local SURE parameter h. One may observe the rare patch effect already mentioned in Figure 3.9 and in Section 4.1: for some pixels (especially near contrasted edges) the most similar patches are very far in the patch space. Using a fixed value h that is globally optimal, the algorithm does not select enough patches to significantly reduce the variance. This phenomenon may also appear on contrasted textures, as illustrated in Figures 4.7 and 4.8: the NLmeans with global parameter leaves the noise on the fur of the tiger. On the contrary, the NLmeans with local SURE uses a very high value of h in these difficult regions in order to gather enough pixels. A map of the parameter h is shown in Figure 4.6: regions with rare patches are clearly highlighted.

With a smaller patch size ($s^2 = 5 \times 5$) the "rare patch effect" is less noticeable, but it is still there, as can be seen on Figure 4.9. Using an even smaller patch size (e.g. 3×3) for that noise level would leave too much noise (mottled aspect) as explained in Section 3.4 and Figure 3.7. One may notice that the local SURE version removes the rare patch effect and that it is quite robust to the patch size (the visual impression is even slightly better with patch size $s^2 = 7 \times 7$).

Figure 4.10 provides an illustration of the better preservation of textures with the local parameter h. Notice also the preservation of the eye of the baby penguin. This figure also shows the result provided by the indicator oracle. It is far above the other

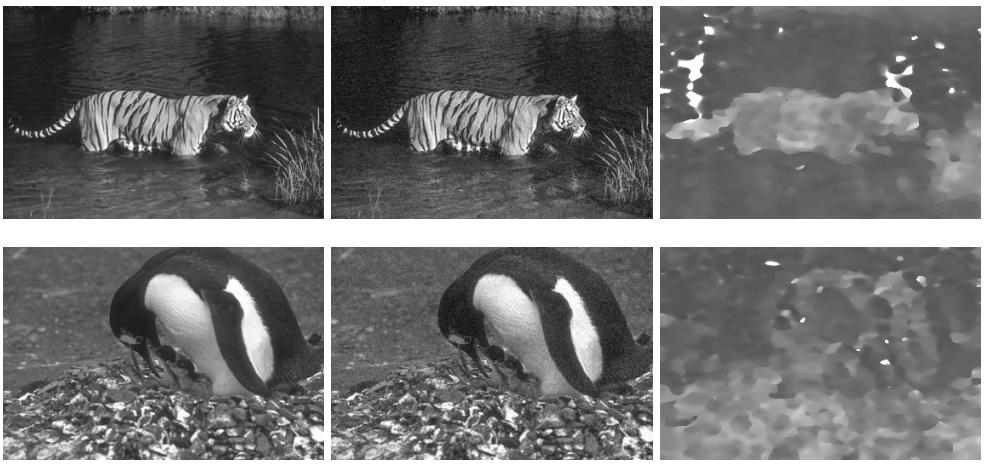


FIG. 4.6. *From left to right: original images, noisy images ($\sigma = 10$, PSNR 28.12 dB on the top, 28.13 dB on the bottom), map of the values of smoothing parameter h used by the local-SURE. Notice that, as with the oracle, the values along contrasted edges are high. Similarly, it is very high on the fur of the tiger where it is difficult to find similar patches.*

FIG. 4.7. *From left to right : NLmeans with global parameter $h = 14$ chosen to maximize the PSNR (31.71dB), result of the local SURE denoising (PSNR 32.33dB).*

FIG. 4.8. *Zoom of Figure 4.7. With the local parameter prescribed by SURE, less noise is left in regions where the most similar patches lie at a large distance in the patch space.*

two methods but it is not available in practice since it relies on the noise free image.

With a higher noise level ($\sigma = 20$), Figure 4.11 reveals that the value of h should be chosen in function of the local contrast if one wants both to reduce the noise and keep sharp edges. Using an optimal global value makes the less contrasted edge blurry, similarly to the toy examples of Section 2. Yet, assigning a smaller global value that makes the less contrasted edge sharp gives a noisy halo around the more contrasted edge. Both the NLmeans with local SURE and the oracle deal locally with the contrast.

The ability to preserve small details is also illustrated in Figure 4.12. We compare the classical NLmeans (with gaussian weights), the NLmeans with spline weights and global smoothing parameter h, and the NLmeans with spline weights and local h, on a noisy image ($\sigma = 10$). The smoothing parameter of the first two filters is set to maximize the PSNR. We observe that the main gain in detail preservation is provided by adaptivity and not by the weight function.

One issue with adaptivity, however, is that the visual result might not look very natural when the decision on the smoothing parameter varies quickly because of the randomness of the SURE estimator. In Figure 4.12, the filter tries to preserve a rope in region 4 but it leaves a noisy spot instead. Figure 4.13 shows this phenomenon on the ground (notice however that the texture of the roof is better preserved with the local SURE filter). A solution is to enlarge the size of the neighborhood when averaging the risk, but it reduces the adaptivity of the filter.

Choice of the other parameters. In Section 4.1, we have shown that the oracle could benefit from having a large search window. Figure 4.4 displays the evolution of the PSNR as the search window increases for the NLmeans with local SURE. The filter is very robust to the change of the search window (as a consequence we leave this parameter unchanged in the experiments), but contrary to the oracle, the PSNR tends to decrease when the search window gets large. Presumably one needs an accuracy in the choice of h that only the oracle is able to provide in order to benefit from large search windows. Comparisons with the global parameter and gaussian weights show that this robustness mainly stems from the compact support of the weights, rather than from the adaptivity.

Eventually, we can locally adapt the patch size as well, using the local SURE. Figures 4.15 and 4.14 recap the trade-off one has to face when choosing the global patch size: a too small patch size yields a mottling effect but it allows to preserve details, whereas a large patch sizes produces smoother images except in regions where it brings the rare patch effect. Choosing locally h already makes this choice easier by allowing to preserve details with large patch sizes.

Therefore, the gain in letting s vary locally is visually small, although in our experiments it always provides a slightly higher PSNR. An example of the fully local method is shown in Figure 4.16. Because of the small textures, the best PSNR (32.88 dB) with global parameter h and s is provided by the patch size $s^2 = 3 \times 3$. However the image looks mottled, especially on the water and on the clouds. The local choice of h and s^2 provides a slightly higher PSNR (33.18 dB) and it removes the mottling effect while preserving textures. However, using a fixed patch size (e.g. 7×7) and a local h would yield a very similar result (not represented here): the extra-computation time might not be worth it. The map of s shows that the algorithm chooses a small patch size in the textured regions and a large patch size in smooth regions.

Conclusion. The local SURE method chooses automatically the smoothing parameter h of the NLmeans filter, and it is robust to the choice of the search window

(a) Original (b) Global h, $s^2 = 5 \times 5$ (c) Global h, $s^2 = 7 \times 7$

(d) Noisy $\sigma = 10$ (e) Local h, $s^2 = 5 \times 5$ (f) Local h, $s^2 = 7 \times 7$

FIG. 4.9. *Extract of an experiment on the "couple" image. First row: extract of the original image (a), results of NLmeans with global parameter h: the patch size is* $s^2 = 5 \times 5$ *in (b) (PSNR* $32.46dB$*), and* $s^2 = 7 \times 7$ *in (c) (PSNR* $32.14dB$*). In both cases, h was chosen to maximize the PSNR. Second row: noisy image (c) (*$\sigma = 10$*), results of NLmeans with local-SURE parameter h: the patch size is* $s^2 = 5 \times 5$ *in (e) (PSNR* $32.77dB$*), and* $s^2 = 7 \times 7$ *in (f) (PSNR* $32.70dB$*). Notice how the face, the tie and the shoulder are smoother with the local SURE (the reader should zoom on this picture) with both patch sizes. Yet the contrast of the wall is not lost.*

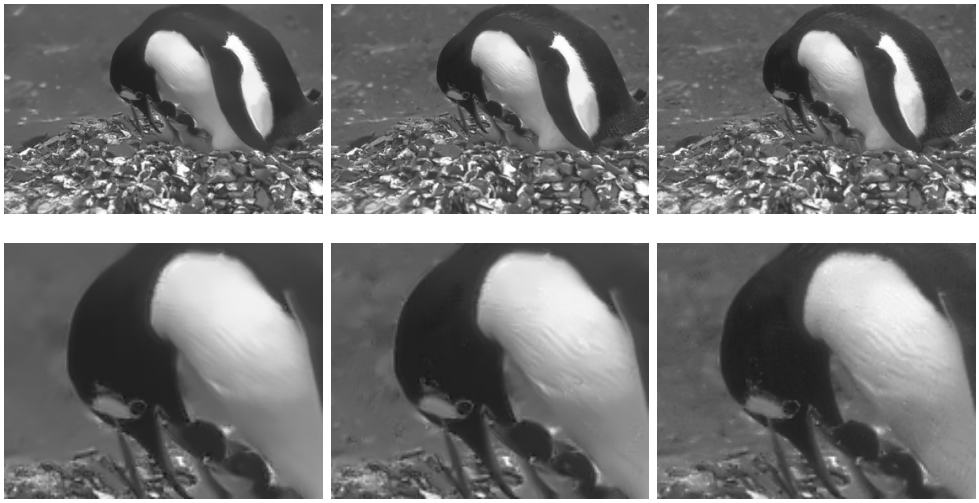


FIG. 4.10. *From left to right: result of the NLmeans filter with global h optimized for PSNR (*31.17 *dB), result of the filter with local h prescribed by SURE (PSNR* 31.96 *dB), result of the indicator oracle filter (PSNR* $33,23$ *dB). Let us recall that the last result is computed using the noise free image and it is therefore not computable in practice.*

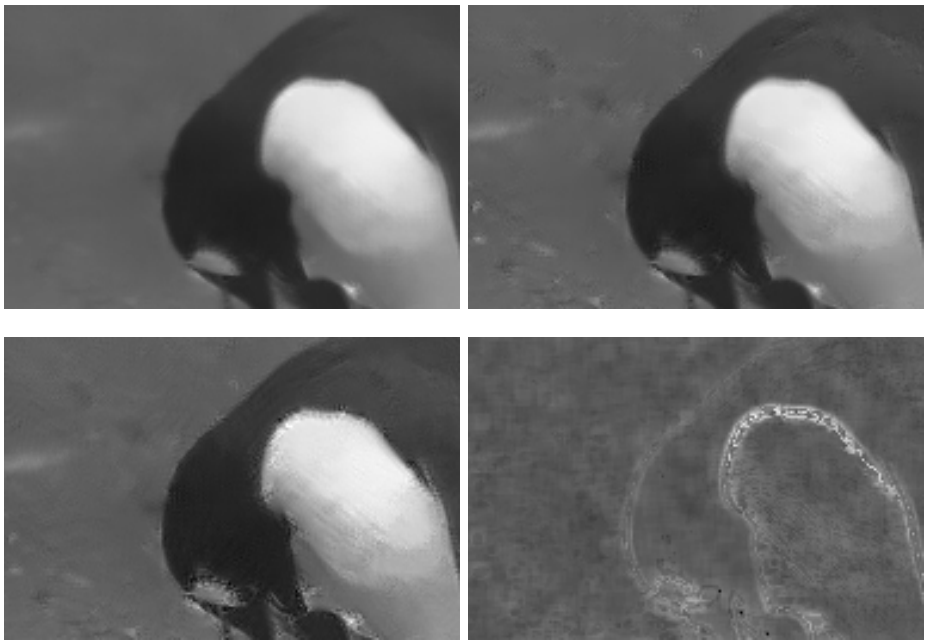


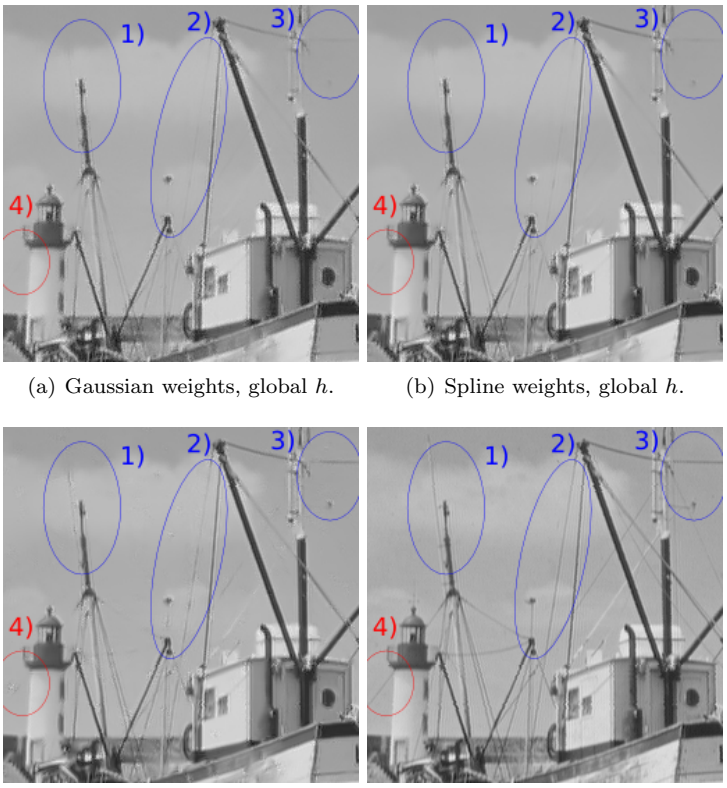
FIG. 4.11. *Experiment with a noise level $\sigma = 20$. Top left: result of the NLmeans filter (spline weights, $h = 30$ optimized for the PSNR). The global optimal parameter is too high for the least contrasted edges, so that, as in (2.6), they are blurred. Top right: result of the local SURE filter. Along the least contrasted edge, the chosen value of h is about 24. If we set the global parameter to 24 (bottom left), these edges become sharp but the more contrasted edges become noisy. Bottom right: map of the values of h prescribed by the indicator oracle: the more contrasted the edge, the higher h should be.*

W and the patch size s^2. The patch size s^2 can be chosen locally but it provides hardly any visual improvement over a global (well chosen) patch size: we suggest to select a global patch size large enough to avoid the mottling effect (the prescription of global SURE works fine) and a local parameter h.

The main improvement brought by the locality is to remove the "rare patch effect". This is definitely a visual improvement, but since the concerned regions, along edges, or highly contrasted textures, are not prominent in images, the gain on the global PSNR is moderate. A second, less striking, improvement is the better preservation of the contrast of details and textures. However, a limitation of the method is that the decision of the local SURE might yield small visual artifacts in some regions. We plan to study further the behavior of SURE for NLmeans estimator in order to reduce this effect.

Acknowledgements. The first author would like to thank Joseph Salmon, Charles Deledalle and Julien Rabin for fruitful discussions about this work.

Appendix A: A Gradient descent to choose an optimal global parameter. Here we propose a fast way to select an optimal global parameter h without computing NLmeans for many different values of h as in the local approach. To this end, we compute the derivative of the SURE estimator, which allows for gradient descents or any descent method that relies on the derivative of the SURE estimator, as in [31] in the wavelet framework. For instance:



(a) Gaussian weights, global h. (b) Spline weights, global h.

(c) Spline weights, local h. (d) Original image (without noise).

FIG. 4.12. *Comparison of the NLmeans filter on a noisy image ($\sigma = 10$) : (a) with gaussian weights, global h, (b) spline weights, global h, (c) spline weights, h is locally set by the local SURE. The original image (without noise) is shown in (d). In (a) and (b), the parameter h is optimized for PSNR. The difference between them is barely visible. In regions 1),2) and 3), the adaptivity (c) allows to reconstruct fine structures such as ropes and antennas. However, the filter leaves a noisy spot in region 4) when trying to preserve a fine rope.*

Line-search approach Compute the derivative of the SURE estimator: $\frac{\partial J}{\partial h}$ (see below) and perform a line search :

1. Choose h_L (small), h_R (large) such that $\frac{\partial J}{\partial h}(h_L) < 0$ and $\frac{\partial J}{\partial h}(h_R) > 0$. Set $h := \frac{h_L + h_R}{2}$.
2. Compute $\frac{\partial J}{\partial h}(h)$:
   - If $\frac{\partial J}{\partial h}(h) = 0$ Stop.
   - If $\frac{\partial J}{\partial h}(h) < 0$, set $h_L := h$. Go to step 3.
   - If $\frac{\partial J}{\partial h}(h) > 0$, set $h_R := h$. Go to step 3.
3. Update $h := \frac{h_L + h_R}{2}$. Go to step 2.

To find the optimal value h, we would (ideally) perform this line search or a gradient descent over the Mean Square Error. Provided that one can swap the expectancy and the derivation symbols, we see that $\frac{\partial J}{\partial h}$ is an unbiased estimator of $\frac{\partial MSE}{\partial h}$, and we can perform the descent with J. The computation of this derivative is a bit long, but one may find the following expression:

FIG. 4.13. *Top left : Original image. Top right: noisy image ($\sigma = 10$). Bottom left: result of the NLmeans filter ($h = 6.5$, optimized for the PSNR : 30.85dB). Bottom right: result of the local SURE filter (PSNR 31.45dB). The texture of the roof is better preserved, on the ground however the preservation of the texture in small areas looks a bit chaotic.*

$$\frac{\partial J}{\partial \alpha} = 2\sigma^2 \underbrace{\frac{\partial^2}{\partial \alpha \partial z}\left(NL\tilde{u}(x)\right)}_{:=A} + 2(NL\tilde{u}(x) - \tilde{u}(x)) \underbrace{\frac{\partial}{\partial \alpha}\left(NL\tilde{u}(x)\right)}_{:=B} \tag{4.5}$$

with:

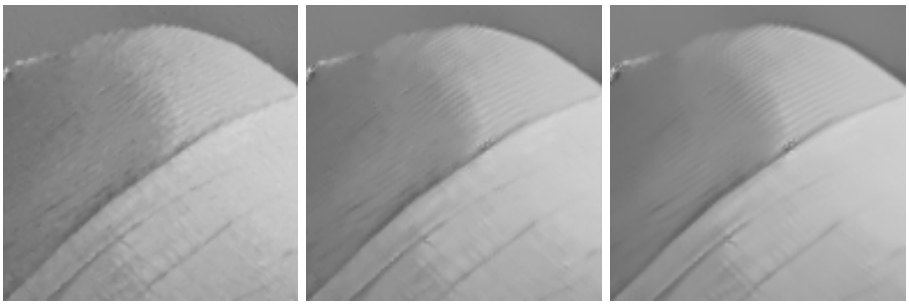
$$A = \frac{\partial^2}{\partial \alpha \partial z}\left(\sum_y \frac{\varphi_y}{C}\right) = -\frac{\varphi(0)}{C^2}\sum_w \frac{\partial \varphi_w}{\partial \alpha} + \sum_y \tilde{u}(y)\frac{\partial^2}{\partial \alpha \partial z}\left(\frac{\varphi_y}{C}\right),$$

$$B = \frac{\partial}{\partial \alpha}\left(\sum_y \tilde{u}(y)\frac{\varphi_y}{C}\right) = \frac{1}{C}\sum_y \tilde{u}(y)\frac{\partial \varphi_y}{\partial \alpha} - \frac{1}{C^2}\left(\sum_y \tilde{u}(y)\varphi_y\right)\left(\sum_w \frac{\partial \varphi_w}{\partial \alpha}\right),$$

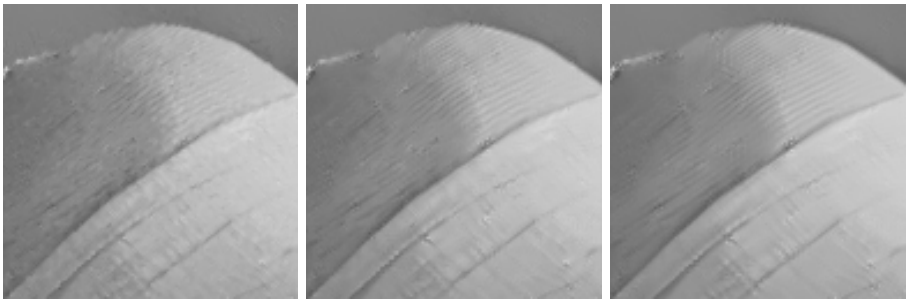
and :

$$\begin{aligned}\sum_y \tilde{u}(y)\frac{\partial^2}{\partial \alpha \partial z}\left(\frac{\varphi_y}{C}\right) = {} & \frac{1}{C}\sum_y \tilde{u}(y)\frac{\partial^2 \varphi_y}{\partial \alpha \partial z} - \frac{1}{C^2}\left(\sum_y \tilde{u}(y)\frac{\partial \varphi_y}{\partial \alpha}\right)\left(\sum_w \frac{\partial \varphi_w}{\partial z}\right) \\ & - \frac{1}{C^2}\left(\sum_y \tilde{u}(y)\frac{\partial \varphi_y}{\partial z}\right)\left(\sum_w \frac{\partial \varphi_w}{\partial \alpha}\right) - \frac{1}{C^2}\left(\sum_y \tilde{u}(y)\varphi_y\right)\left(\sum_w \frac{\partial^2 \varphi_w}{\partial \alpha \partial z}\right) \\ & + \frac{2}{C^3}\left(\sum_y \tilde{u}(y)\varphi_y\right)\left(\sum_w \frac{\partial \varphi_w}{\partial z}\right)\left(\sum_r \frac{\partial \varphi_r}{\partial \alpha}\right). \end{aligned} \tag{4.6}$$

Eventually, we need to express the derivatives of φ_y, and to remember that $\frac{\partial J}{\partial h} = -\frac{2}{h^3}\frac{\partial J}{\partial \alpha}$:



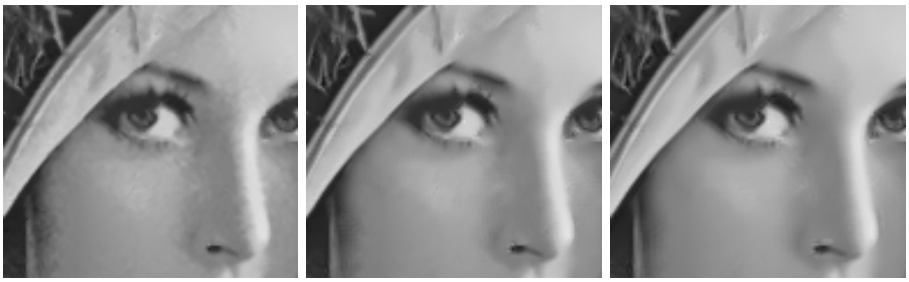
(a) Global h, $s^2 = 3 \times 3$ (b) Global h, $s^2 = 5 \times 5$ (c) Global h, $s^2 = 7 \times 7$



(d) Local h, $s^2 = 3 \times 3$ (e) Local h, $s^2 = 5 \times 5$ (f) Local h, $s^2 = 7 \times 7$

FIG. 4.14. *Patch size and textures. Top row: NLmeans with global smoothing parameter h (optimized for PSNR) and different values of the patch size s^2. Bottom row: Local parameter h. The least contrasted textures are better preserved with a small patch size. However, what looks like texture with patch size 3×3 might as well be the mottling artifact (see below). With the local parameter h, the textures are preserved even with a large patch size.*

(a) Global h, $s^2 = 3 \times 3$ (b) Global h, $s^2 = 5 \times 5$ (c) Global h, $s^2 = 7 \times 7$

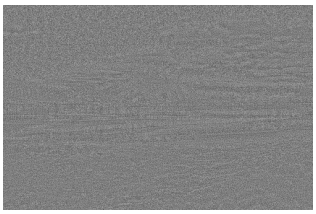


(d) Local h, $s^2 = 3 \times 3$ (e) Local h, $s^2 = 5 \times 5$ (f) Local h, $s^2 = 7 \times 7$

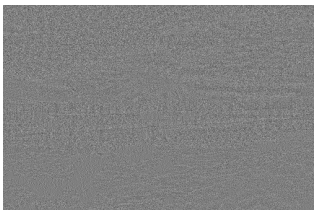
FIG. 4.15. *Patch size and robustness to noise (extract of the same experiment as in Figure 4.14). With a too small patch size, the algorithm leaves too much noise: Lena's skin looks mottled. As the patch size increases, this effect reduces but the rare patch artifact appears. With a local h, the rare patch effect is reduced, which allows to use large patches.*

(a) Global h, $s^2 = 3 \times 3$

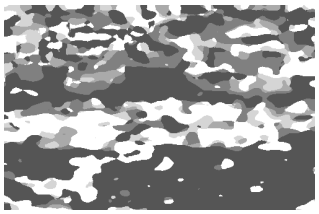
(b) Local h and s^2



(c) Global h, $s^2 = 3 \times 3$



(d) Local h and s^2



(e) Map of s

FIG. 4.16. *NLmeans with global parameters (a) on a noisy image ($\sigma = 10$): because of the small textures, the best PSNR is achieved with patch size is 3×3 (PSNR= 32.88 dB - the reader should zoom on this image). In (b), NLmeans with local patch size $s^2 \in \{3 \times 3, 5 \times 5, 7 \times 7, 9 \times 9\}$ and local h: the result is smoother (PSNR 33.13 dB). The respective method noises ($\tilde{u} - NL\tilde{u}$) are shown in (c) and (d). The map of patch size is shown in (e). The algorithm chooses large patch sizes (white) in smooth regions and small patch size (dark) where the local scale is small.*

$$\frac{\partial \varphi_y}{\partial \alpha} = \frac{\|\tilde{U}(x) - \tilde{U}(y)\|^2}{2} \varphi_y',$$

$$\frac{\partial^2 \varphi_y}{\partial \alpha \partial z} = \frac{1}{s^d}(\tilde{u}(x) - \tilde{u}(y) - \underbrace{(\tilde{u}(x + i_0) - \tilde{u}(x))}_{\text{if } \exists i_0, \in P,\ x = y + i_0}) \left[\varphi_y' + \alpha \frac{\|\tilde{U}(x) - \tilde{U}(y)\|^2}{2} \varphi_y'' \right].$$

Figure 4.17 shows the evolution of the Mean Square Error (MSE) as a function of h, and its estimation by SURE. It is typically a quasi-convex function, so that the line search described above is appropriate to find a minimizer. The cumulative sums of the derivative $\frac{\partial J}{\partial h}$ are also displayed: it is a robust estimation of the derivative of the MSE. It also shows the convergence of the MSE when performing the line search iterations. Starting with $h_L = 0.7$, $h_R = 45$ (which is not particularly adequate, one should rather start with the linear prediction on h), the algorithm converges in 5 or 6 iterations.

Appendix B: Heuristic for the convolution radius r. Let us try to bound the number N of pixels necessary to perform a good estimation of the optimal denoised value. This rough bound aims only at showing that the local estimation method can be performed in practice by giving an order of magnitude of N. Let $h_0 := \arg\min_h \left(NL_h\tilde{u}(x) - \tilde{u}(x)\right)^2$ be the optimal smoothing parameter for NLmeans at pixel x, and $h_1 := \arg\min_h J_h(x)$ the value of h given by minimizing the SURE estimator over h. Let $V := \text{Var}\left(J_h(x)) - \left(NL_h\tilde{u}(x) - \tilde{u}(x)\right)^2\right)$ be the variance of the error when estimating the risk, and assume momentarily that it does not depend

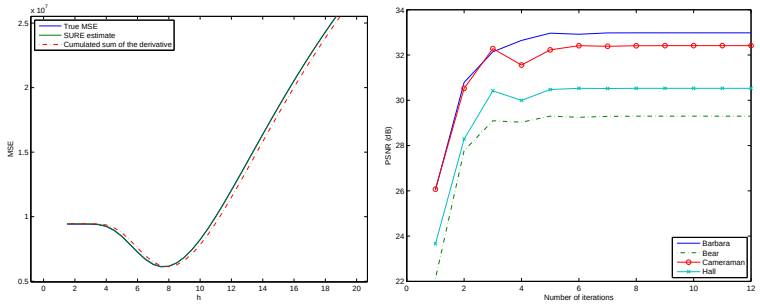


FIG. 4.17. *Left: Typical curve of the Mean Square Error as a function of h. The curve of the SURE estimation (solid green line) and the true MSE (solid blue line) perfectly fit. The dashed red line shows the cumulative sums (rectangle integration) of the estimated derivative (Eq. 4.5). The slight shift can be attributed to the integration method. Right: Evolution of the PSNR during a line search (see Appendix A) based on the estimate of the gradient: 5 or 6 iterations are sufficient for convergence.*

on h. Then, we approximate $\Big(J_h(x)) - (NL_h\tilde{u}(x) - u(x))^2\Big)^2 \approx V$, so that :

$$\begin{aligned}\Big||NL_h\tilde{u}(x) - u(x)| - \sqrt{J_h(x)}\Big| &\approx \frac{\sqrt{V}}{|NL_h\tilde{u}(x) - u(x)| + \sqrt{J_h(x)}} \\ &\approx \frac{\sqrt{V}}{2|NL_h\tilde{u}(x) - u(x)|} \text{ as a first order approximation.}\end{aligned}$$

The right member being uniformly upper-bounded by $\frac{\sqrt{V}}{2|NL_{h_0}\tilde{u}(x)-u(x)|}$, we see that the difference between the minimum values is bounded by :

$$\Big||NL_{h_0}\tilde{u}(x) - u(x)| - \sqrt{J_{h_1}(x)}\Big| \leq \frac{\sqrt{V}}{2|NL_{h_0}\tilde{u}(x) - u(x)|}.$$

Now, we can show that the error using the SURE estimate is close to the minimal error:

$$\begin{aligned}|NL_{h_1}\tilde{u}(x) - u(x)| &\leq \Big||NL_{h_1}\tilde{u}(x) - u(x)| - \sqrt{J_{h_1}}\Big| + \sqrt{J_{h_1}} \\ &\leq \frac{\sqrt{V}}{2|NL_{h_1}\tilde{u}(x) - u(x)|} + |NL_{h_0}\tilde{u}(x) - u(x)| + \frac{\sqrt{V}}{2|NL_{h_0}\tilde{u}(x) - u(x)|} \\ &\leq |NL_{h_0}\tilde{u}(x) - u(x)| + \frac{\sqrt{V}}{|NL_{h_0}\tilde{u}(x) - u(x)|}\end{aligned}$$

If we assume[6] that we can average the risk in a local neighborhood of x which contains N pixels, the above calculations hold with V being replaced by $\frac{V}{N}$.

Obtaining an analytic expression of the variance V as done in [18] in the context of frame denoising seems very difficult here, and it may not even provide useful information on the number of pixels N we are looking for. Instead, we estimate V empirically. We consider, for a fixed h and different noise levels (σ ranging from 5 to

[6] In particular, we assume that the properties of the image in this neighborhood are stationary.

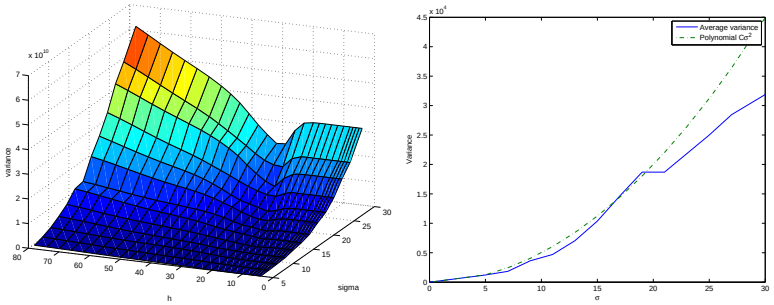


FIG. 4.18. *Left: Average variance of $J - MSE$ as a function of σ and h: for each value of σ, the variance depends on h. The value of h prescribed by the SURE estimator is usually close to the minimum of variance over h. Right : Variance of the $J - MSE$ as a function of σ, for $h = h(\sigma)$ the value prescribed by SURE (solid line). In dashed green line, the function $\sigma \mapsto C\sigma$ with C such that the curves match at $\sigma = 5$.*

30), say, 500 realizations of the noise on an image u and we estimate the variance of $J(x) - (NL\tilde{u}(x) - u(x))^2$, i.e. the variance of the error when estimating the risk. We perform this estimation inside different regions (textured/ smooth areas).

For instance, we find that for $\sigma = 10$ and $h = 10$ (using gaussian weights), $V \approx 10^4$. Moreover, the order of magnitude of $|NL_{h_0}\tilde{u}(x) - u(x)|$ is given by the square root of the Mean Square Error (MSE). For these noise levels, the typical optimal PSNR is about 32 dB, which means MSE ≈ 40.

If the local neighborhood is a disk of radius r around x, the error when estimating the optimal value is

$$|NL_{h_1}\tilde{u}(x) - u(x)| \lesssim |NL_{h_0}\tilde{u}(x) - u(x)| + \frac{10^2}{\sqrt{40\pi r}} \approx |NL_{h_0}\tilde{u}(x) - u(x)| + \frac{10}{r}.$$

In other words, *if we average the risk on a disk of radius* 10*, the difference between the denoising with optimal parameter and the one prescribed by SURE is less than one gray level, which is negligible.*

This computation gives an order of magnitude for a suitable radius and it does not aim at prescribing the exact radius one should use, but it is very close to the value we have empirically found suitable (in practice we use a radius of 13 or 14).

When σ varies and the parameter h is chosen to minimize J, we find that the variance V is less than $C\sigma^2$ for some constant C (see Figure 4.18). Therefore, if we choose the convolution radius proportional to σ, we can keep the difference between the optimal error and the one of the estimator below 1.

REFERENCES

[1] S. P. Awate and R. T. Whitaker. Unsupervised, information-theoretic, adaptive image filtering for image restoration. *IEEE Trans. Pattern Anal. Mach. Intell.*, 28(3):364–376, 2006.

[2] N. Azzabou, N. Paragios, and F. Guichard. Uniform and textured regions separation in natural images towards MPM adaptive denoising. In *SSVM*, pages 418–429, 2007.

[3] N. Azzabou, N. Paragios, F. Guichard, and F. Cao. Variable bandwidth image denoising using image-based noise. In *CVPR*, 2007.

[4] T. Brox and D. Cremers. Iterated nonlocal means for texture restoration. In F. Sgallari, A. Murli, and N. Paragios, editors, *Proc. International Conference on Scale Space and Variational Methods in Computer Vision*, volume 4485 of *LNCS*, pages 13–24, Ischia, Italy, May 2007. Springer.

[5] T. Brox, O. Kleinschmidt, and D. Cremers. Efficient nonlocal means for denoising of textural patterns. *IEEE Transactions on Image Processing*, 17(7):1083–1092, July 2008.

[6] A. Buades, B. Coll, and J.M Morel. A review of image denoising algorithms, with a new one. *Multiscale Modeling and Simulation (SIAM interdisciplinary journal)*, 4:490–530, 2005.

[7] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazaran. Image denoising by sparse 3d-transform-domain collaborative filtering. In *IEEE Transactions on Image Processing*, volume 16, 2007.

[8] J. Darbon, A. Cunha, T.-F. Chan, S. Osher, and G.J. Jensen. Fast nonlocal filtering applied to electron cryomicroscopy. In *IEEE International Symposium on Biomedical Imaging (ISBI): From Nano to Macro*, pages 1331–1334, 2008.

[9] C.-A. Deledalle, L. Denis, and F. Tupin. Iterative weighted maximum likelihood denoising with probabilistic patch-based weights. *IEEE Trans. on Image Processing*, 18, 2009.

[10] A. A. Efros and T. K. Leung. Texture synthesis by non-parametric sampling. In *IEEE International Conference on Computer Vision*, pages 1033–1038, Corfu, Greece, September 1999.

[11] G. Gilboa and S. Osher. Nonlocal linear image regularization and supervised segmentation. *Multiscale Modeling and Simulation (SIAM interdisciplinary journal)*, 6:595–630, 2007.

[12] G. Gilboa and S. Osher. Nonlocal operators with applications to image processing. *Multiscale Modeling and Simulation (SIAM interdisciplinary journal)*, 7:1005–1028, 2008.

[13] B. Goossens, Q. Luong, A. Pizurica, and W. Philips. An improved non-local denoising algorithm. In *International Workshop on Local and Non-Local Approximation in Image Processing*, 2008.

[14] C. Kervrann and J. Boulanger. Local adaptivity to variable smoothness for exemplar-based image denoising and representation. *Int. J. Computer Vision*, 79:45–69, 2008.

[15] S. Kindermann, S. Osher, and P. Jones. Deblurring and denoising of images by nonlocal functionals. *Multiscale Modeling and Simulation (SIAM interdisciplinary journal)*, 4:1091–1115, 2005.

[16] Y. Lou, X. Zhang, S. Osher, and A. Bertozzi. Image recovery via nonlocal operators. *Journal of Scientific Computing*, 2009. published online, DOI 10.1007/s10915-009-9320-2, 2009.

[17] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman. Non-local sparse models for image restoration. In *International Conference on Computer Vision (ICCV)*, 2009.

[18] J.-C. Pesquet, A. Benazza-Benyahia, and C. Chaux. A sure approach for digital signal/image deconvolution problems. *IEEE Transactions on Signal Processing*, 57:4616–4632, 2009.

[19] G. Peyré. Image processing with non-local spectral bases. *Multiscale Modeling and Simulation (SIAM interdisciplinary journal)*, 7:703–730, 2008.

[20] G. Peyré. Manifold models for signals and images. *Computer Vision and Image Understanding*, 113:249–260, 2009.

[21] J. Salmon. On two parameters for denoising with non-local means. *IEEE Signal Process. Lett.*, 2010.

[22] J. Salmon and E. Le Pennec. An aggregator point of view on NL-Means. In *Wavelets XIII*, volume 7446, page 74461E. SPIE, 2009.

[23] A. Singer, Y. Shkolnisky, and B. Nadler. Diffusion interpretation of non-local neighborhood filters for signal denoising. *SIAM Journal on Imaging Sciences*, 2:118–139, 2009.

[24] C. Stein. Estimation of the mean of a multivariate normal distribution. *Ann. Statist.*, 9:1135–1151, 1981.

[25] A. Szlam. Non-local means for audio denoising. Technical Report 08-56, UCLA CAM report, 2008.

[26] T. Tasdizen. Principal neighborhood dictionaries for nonlocal means image denoising. *IEEE Transactions on Signal Processing*, 18:2649–2660, 2009.

[27] D. Tschumperlé and L. Brun. Image denoising and registration by PDE's on the space of patches. In *International Workshop on Local and Non-Local Approximation in Image Processing (LNLA'08), Lausanne/Switzerland*, 2008.

[28] D. Tschumperlé and L. Brun. Non-local image smoothing by applying anisotropic diffusion PDE's in the space of patches. In *IEEE International Conference on Image Processing (ICIP'09), Cairo/Egypt*, 2009.

[29] D. Van De Ville and M. Kocher. SURE based non-local means. *IEEE Signal Processing Letters*, 16:973–976, 2009.

[30] L. P. Yaroslavsky. *Digital Picture Processing - An Introduction*. Springer Verlag, 1985.

[31] Xiao-Ping Zhang and Zhi-Quan Luo. A new time-scale adaptive denoising method based on wavelet shrinkage. In *ICASSP '99: Proceedings of the Acoustics, Speech, and Signal Processing, 1999. on 1999 IEEE International Conference*, pages 1629–1632, Washington, DC, USA, 1999. IEEE Computer Society.