Efficient and Robust Nonlocal Means Denoising of MR Data Based on Salient Features Matching

Antonio Tristán-Vega[1] Verónica García-Pérez[2]
Santiago Aja-Fernández[2] Carl-Fredrik Westin[1]

[1] *Laboratory of Mathematics in Imaging (Harvard Medical School, Boston, MA)*

[2] *Laboratory of Image Processing. University of Valladolid (Spain)*

Abstract

The Nonlocal Means (NLM) filter has become a popular approach for denoising medical images due to its excellent performance. However, its heavy computational load has been an important shortcoming preventing its use. NLM works by averaging pixels in nonlocal vicinities, weighting them depending on their similarity with the pixel of interest. This similarity is assessed based on the squared difference between corresponding pixels inside local patches centered at the locations compared. Our proposal is to reduce the computational load of this comparison by checking only a subset of salient features associated to the pixels, which suffice to estimate the actual difference as computed in the original NLM approach. The speedup achieved with respect to the original implementation is over one order of magnitude, and, when compared to more recent NLM improvements for MRI denoising, our method is nearly twice as fast. At the same time, we evidence from both synthetic and *in vivo* experiments that computing of appropriate salient features make the estimation of NLM weights more robust to noise. Consequently, we are able to improve the outcomes achieved with recent state of the art techniques for a wide range of realistic Signal-to-Noise Ratio scenarios like diffusion MRI. Finally, the statistical characterization of the features computed allows to get rid of some of the heuristics commonly used for parameter tuning.

Key words: Nonlocal means, Image denoising, Magnetic Resonance Imaging

1 Introduction

Denoising of medical images is an important and rather challenging task, due to the peculiarities of the noise acquired by imaging sensors in UltraSounds (US), Computer Tomography (CT), or, of course, Magnetic Resonance Images (MRI) [1]. A number of filtering techniques have appeared in the literature including anisotropic diffusion [2], wavelets [3], and many others [1]. Among these solutions, the Nonlocal Means (NLM) first described in [4] is lately gaining an increasing popularity due to its excellent performance. NLM is a nonlinear filter based on a Weighted Average (WA) of pixels inside a search window which is relatively large compared to traditional neighborhood techniques, hence the term *nonlocal*. To preserve the structures of the image, the pixels are weighted according to their similarity with the pixel of interest, being the agreement measured as the Mean Squared Difference (MSD) between patches surrounding the pixels under comparison.

NLM has been proven optimal for Gaussian additive and multiplicative noise in [4]. Although the nature of disturbances may differ from Gaussian in some imaging modalities inducing a certain bias [5], NLM has been successfully adapted to US [6], MRI [7,8], and diffusion MRI [9–11]. In these cases, some modified schemes have to be introduced to cope with the particular statistics of noise [7,12] or the particular structure of data [10,13].

Some other approaches have been intended to generalize NLM to higher order models [14], or to perform subband denoising in wavelet decompositions [8,15,16], while other works aim to find optimal values for the parameters of NLM [7,17].

In all these cases, the main drawback of NLM is the need for very intensive computations due to the reckoning of the squared distance between the comparison patches. For this reason, some alternative techniques have been proposed in the literature to speedup the computation of nonlocal averages, both for textured [18] or non-textured [19] images. They are based on diverse methodologies related to the one here presented to some degree, and hence they are further discussed in the next section.

In this paper we propose a method to heavily accelerate the calculation of patch distances, and hence of NLM, by considering only the difference between salient features associated to the pixels to be weighted. In comparison to other related proposals, our technique shows a number of key advantages that are tested over typical MRI data sets: **first**, our calculation preserves the statistical characterization of patches in the original NLM, and therefore its optimality properties. **Second**, this characterization is local, so that we obtain an adaptive behavior. **Third**, salient features are computed for all pixels, so the acceleration is achieved for all patches: we are able to accomplish a net and predictable speedup regardless on the input Signal-to-Noise Ratio (SNR). **Finally**, these features are robustly computed translating in a more accurate estimate of weights, and thus in a notably improved filtering performance in most of cases. The relevance of the contributions previously discussed will become clear from the comparison with the state of the art techniques analyzed hereafter.

2 Background

2.1 Nonlocal Means-based denoising

In the standard formulation, $u(x_i)$ (or u_i) is the gray level of the pixel at position x_i, and the filtered output is computed as [4]:

$$\widehat{u}(x_i) = \sum_{x_j \in \Omega_i} w(x_i, x_j) u(x_j), \tag{1}$$

where Ω_i is a *large* search window centered at pixel x_i (in the original description of the algorithm, Ω_i is indeed the entire image) and $w(x_i, x_j)$ is the weight assigned to pixel x_j with respect to pixel x_i, according to the similarity between two patches $\mathcal{N}_i$ and $\mathcal{N}_j$ centered at x_i and x_j, respectively:

$$w(x_i, x_j) = \frac{1}{Z_i} \exp\left(-\frac{d(x_i, x_j)}{h^2}\right); \; d(x_i, x_j) = \frac{1}{N} \|\mathbf{u}(\mathcal{N}_i) - \mathbf{u}(\mathcal{N}_j)\|_2^2, \tag{2}$$

where Z_i is a normalizing constant so that $\sum_{x_j} w(x_i, x_j) = 1$, and $\mathbf{u}(\mathcal{N}_i)$ denotes an $N \times 1$ vector with all the values $u(x_j)$ at the pixels $x_j \in \mathcal{N}_i$. The parameter h has a clear statistical meaning: it has to be proportional to the expected value of the distance between patches, $E\{d(x_i, x_j)\}$, and hence it is related to the noise power of the image, σ^2. Typically, it is set to $h^2 = \beta^2\sigma^2$, for $\beta \in [0.8, 1.2]$ [7]. It is therefore necessary to correctly estimate $E\{d(x_i, x_j)\}$: if it is overestimated, NLM produces over-smoothing of the structures of interest. If it is underestimated, NLM is not able to properly remove the noise in the image.

From eqs. (1) and (2), it is easy to understand the enormous computational load of NLM: assume that Ω_i (resp. $\mathcal{N}_i$) is an n-dimensional square window of radius M (resp. B). The WA includes all pixels in the search window Ω_i, and hence $(2M + 1)^n$ weights have to be computed. The calculation of each of them requires the evaluation of eq. (2), so that the norm of a vector of length $(2B + 1)^n$ has to be reckoned. Consequently, to process each pixel $(2M + 1)^n(2B + 1)^n$ squared differences have to be computed. Our aim is to heavily reduce this load by efficiently estimating $d(x_i, x_j)$.

2.2 Speedup methods for Nonlocal Means

Different efforts have been reported in the literature to reduce the complexity of NLM. A relevant work in this sense is [19], which propose to combine several acceleration techniques, mainly:

- Voxel preselection. Instead of computing the weights in eq. (2) for all $x_j \in \Omega_i$, those pixels too dissimilar to x_i are assigned a weight 0 *a priori*. The local mean and variance at each x_j are precomputed, and comparing them with those at x_i a decision may be taken on wether to discard x_j for the WA or not.
- Block-wise implementation: the image is divided into overlapping blocks which are NLM-like denoised, and then the pixels are *cleaned* depending on the blocks they belong to. This technique is compatible with voxel preselection and also with our own approach, so we can consider it as a further improvement (although it results in a worsening of the filtering accuracy).

The idea of voxel preselection is not unique in [19], and has been thoroughly explored by some other authors in different ways. In [20] all vectors $\mathbf{u}(\mathcal{N}_i)$

across the image are arranged into one single matrix, and Singular Value Decomposition (SVD) is used to find an optimal base to represent vectors $\mathbf{u}(\mathcal{N}_i)$. By keeping only those coefficients corresponding to the largest Singular Values (SV), it is possible to obtain representations of the patches with increasing accuracy, and to progressively discard dissimilar pixels much like in [19].

A very similar technique is proposed in [21], taking into account not only the local mean and variance but also a set of features related to directional derivatives of the image and other features. Like in the previous cases, the preselection criterion is not obviously related to the actual distance $d(x_i, x_j)$ between the pixels. To overcome this limitation, it is suggested in [18] to build a cluster tree to hierarchically find similar patches based on the distance $d(x_i, x_j)$, thus keeping the statistical meaning of NLM. Unlike the present paper, this work deals with textured images.

To this point, we have only reviewed proposals exclusively based on preselection. An important limitation of this methodology is that the acceleration is only achieved for those voxels which are excluded from the WA, but the heavy computation of $d(x_i, x_j)$ is still required for the remaining ones. This way, the acceleration strongly depends on the peculiarities of each image and the input SNR, yielding unpredictable speedups which might be only marginal.

The aim in this paper is precisely to obtain a net speedup for all voxels inside Ω_i. Although still quite different, the work in [12] is related to ours in this sense: distances $d(x_i, x_j)$ can be estimated instead of explicitly computed, recognizing that the distance $d(x_i, x_j)$ in eq. (2) may be seen as a sample estimate of the expected value of the quadratic difference between the pixels:

$$E\{(u_i - u_j)^2\} = (E\{u_i\} - E\{u_j\})^2 + \mathrm{Var}\{u_i\} + \mathrm{Var}\{u_j\} + \mathrm{Cov}\{u_i, u_j\}, \quad (3)$$

where each term can be replaced by its local sample mean. The problem with this solution is that it does not account for structural similarity between patches, since the simple computation of local statistics does not suffice to describe, for example, image contours. Hence, it highly differs from the original conception of NLM. The same problem is found in SVD approaches [20,22]: since singular vectors are computed globally, the base used cannot properly describe the similarity between local structures.

A more recent approach aims to estimate the distances $d(x_i, x_j)$ based on Principal Component Analysis (PCA) of $\mathcal{N}_i$ and $\mathcal{N}_j$ [23]: only a small subset of principal components describing the patches are compared. However, PCA is carried out over the entire image, so the same weakness when describing local structures arises.

Finally, all the works previously introduced, except for [18], share a common limitation: the distance $d(x_i, x_j)$ is replaced with a difference between features (local statistics or features, SVD or PCA vectors) which is not trivially related to $E\{d(x_i, x_j)\}$. This pitfall compels to heuristically determine h^2, thus compromising the optimality of NLM. Regarding [18], it is still based on preselection, with its inherent limitations. Yet, this work is intended for textured images, which are of less interest for the medical imaging community.

3 Methods

We aim to avoid the state of the art limitations described in the previous paragraphs using the following methodology:

(1) The features used describe the local structure of non-textured patches,

maintaining the original NLM formulation in [4].

(2) We reduce the computation of $d(x_i, x_j)$ to a small subset of features for all pixels $x_j \in \Omega_i$, always obtaining a net (and predictable) speedup.

(3) The statistics of the distance in the features space are easily related to those of $d(x_i, x_j)$, so that the statistical characterization of patches is also conserved allowing to fix the noise parameter h^2 straightforward.

Additionally, our proposal is also compatible with voxel preselection and blockwise implementations (though the latter are not discussed in the paper). Each of these issues is respectively addressed in the next subsections.

3.1 Polynomial representation of comparison patches

To describe the image with a small number of features, we model it locally (inside the patch $\mathcal{N}_i$) as an (hyper)surface of the form (for 2-D images):

$$u(s_j, t_j) \simeq c_0 + c_s s_j + c_t t_j + \frac{1}{2} c_{ss} s_j^2 + \frac{1}{2} c_{tt} t_j^2 + c_{st} s_j t_j + \dots, \tag{4}$$

where (s_j, t_j) is the offset of pixel $x_j \in \mathcal{N}_i$ with respect to x_i. In eq. (4) c_0 is related to the local mean value of $u(x_i, y_i)$; c_s and c_t to the local variance; c_{ss}, c_{tt}, and c_{st} to the third order moment, and so on. Instead of the global truncated SVD in [20] or the global PCA in [23], we use a truncated **local** Taylor series expansion. The features c describe the local structural properties of the image, such as its mean gray level (c_0) or its gradient (c_s, c_t), related to image contours. To robustly extract these features, we use Least Squares (LS). In case the series in eq. (4) is truncated to degree 2, the problem statement is:

$$\begin{bmatrix} 1 & s_1 & t_1 & \frac{1}{2}s_1^2 & \frac{1}{2}t_1^2 & s_1t_1 \\ 1 & s_2 & t_2 & \frac{1}{2}s_2^2 & \frac{1}{2}t_2^2 & s_2t_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & s_N & t_N & \frac{1}{2}s_N^2 & \frac{1}{2}t_N^2 & s_Nt_N \end{bmatrix} \begin{bmatrix} c_0 \\ c_s \\ c_t \\ c_{ss} \\ c_{tt} \\ c_{st} \end{bmatrix} \simeq \begin{bmatrix} u(s_1,t_1) \\ u(s_2,t_2) \\ \vdots \\ u(s_N,t_N) \end{bmatrix} \Leftrightarrow \mathbf{X}\cdot\mathbf{c} \simeq \mathbf{u}, \tag{5}$$

where N is the number of pixels inside the patch, arranged in the vector $\mathbf{u} = [u(s_1,t_1),\ldots,u(s_N,t_N)]^T$. The LS matrix $\mathbf{X}$ contains only the relative positions (offsets) of the pixels in the neighborhood, and hence it is the same for all patches. The vector $\mathbf{c} = [c_0,\ldots,c_{st}]^T$ can be computed in closed form:

$$\mathbf{c} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{u}. \tag{6}$$

Eq. (6) can be explicitly evaluated for orders 0, 1, and 2 for the standard case of square, symmetric patches, yielding, for order 0:

$$c_0 = \overline{u}, \tag{7}$$

where $\overline{u}$ is the sample mean of u: $\overline{u} = \sum_{i=1}^{N} u(s_i,t_i)/N$. For order 1:

$$c_0 = \overline{u}; \qquad c_s = \frac{\overline{s\cdot u}}{\overline{s^2}}; \qquad c_t = \frac{\overline{t\cdot u}}{\overline{t^2}} \tag{8}$$

Finally, for order 2:

$$\begin{aligned} c_0 &= \frac{(\overline{s^2t^2}^2 - \overline{s^4}\,\overline{t^4})\overline{u} + (\overline{s^2}\,\overline{t^4} - \overline{s^2t^2}\,\overline{t^2})\overline{s^2u} + (\overline{s^4}\,\overline{t^2} - \overline{s^2t^2}\,\overline{s^2})\overline{t^2u}}{(\overline{s^4} - \overline{s^2t^2})(2\overline{s^2}^2 - \overline{s^2t^2} - \overline{s^4})}; \\ c_s &= \frac{\overline{s\cdot u}}{\overline{s^2}}; \qquad c_t = \frac{\overline{t\cdot u}}{\overline{t^2}}; \qquad c_{st} = \frac{\overline{s\cdot t\cdot u}}{\overline{s^2t^2}}; \\ c_{ss} &= \frac{(\overline{s^2}\,\overline{t^4} - \overline{s^2t^2}\,\overline{t^2})\overline{u} + (\overline{t^2}^2 - \overline{t^4})\overline{s^2u} + (\overline{s^2t^2} - \overline{s^2}\,\overline{t^2})\overline{t^2u}}{(\overline{s^4} - \overline{s^2t^2})(2\overline{s^2}^2 - \overline{s^2t^2} - \overline{s^4})}; \\ c_{tt} &= \frac{(\overline{t^2}\,\overline{s^4} - \overline{s^2t^2}\,\overline{s^2})\overline{u} + (\overline{s^2}^2 - \overline{s^4})\overline{t^2u} + (\overline{s^2t^2} - \overline{s^2}\,\overline{t^2})\overline{s^2u}}{(\overline{s^4} - \overline{s^2t^2})(2\overline{s^2}^2 - \overline{s^2t^2} - \overline{s^4})}; \end{aligned} \tag{9}$$

Note that all terms in the previous equations can be precomputed except for $\overline{s^p t^q u}$, with $p, q = 0, 1, 2$. These terms are computed in an efficient way as separable convolutions, so the overload due to their calculation is negligible. For truncation order 1 (resp. 2), it is only necessary to compute 3 (resp. 6) separable convolutions. For 3-D, this number grows to 4 (resp. 10).

3.2 Approximation of patch distances

Our aim is to estimate patch distances $d(x_i, x_j)$ as distances in the features space, $\widetilde{d}(x_i, x_j)$. To that end, we compute the differences between the LS-fitted surfaces, instead of the original pixels themselves. The interpolated patch surrounding x_i, $\widetilde{\mathbf{u}}_i$, can be written in terms of the coefficients $\mathbf{c}_i$ obtained from eq. (6) for $\mathbf{u}(\mathcal{N}_i)$:

$$\widetilde{\mathbf{u}}_i = \mathbf{X} \cdot \mathbf{c}_i, \tag{10}$$

and the MSD between the interpolated surfaces reads:

$$\widetilde{d}(x_i, x_j) = \frac{1}{N}(\widetilde{\mathbf{u}}_i - \widetilde{\mathbf{u}}_j)^T(\widetilde{\mathbf{u}}_i - \widetilde{\mathbf{u}}_j) = \frac{1}{N}(\mathbf{c}_i - \mathbf{c}_j)^T \mathbf{X}^T \mathbf{X} (\mathbf{c}_i - \mathbf{c}_j). \tag{11}$$

For order 1, $\mathbf{X}^T\mathbf{X}$ reduces to a very simple diagonal matrix (note that $\overline{s} = \overline{t} = \overline{st} = 0$), and hence:

$$\widetilde{d}(x_i, x_j) = (c_{0i} - c_{0j})^2 + (c_{si} - c_{sj})^2 \overline{s^2} + (c_{ti} - c_{tj})^2 \overline{t^2}, \tag{12}$$

so the computation of the norm of a $(2b+1)^2 \times 1$ vector in eq. (2) is reduced to the computation of the norm of a 3×1 vector. Since this computation is the slowest part of NLM, the overall speedup is nearly $(2b+1)^2/3$. For order 2, eq. (12) becomes more complicated, but the advantage is still notable.

3.3 Statistical characterization of approximated patch distances

As stated before, we need to compute $E\{\widetilde{d}(x_i,x_j)\}$ related to $E\{d(x_i,x_j)\}$ to properly fix h^2 in eq. (2). Intuitively, the LS fitting in eq. (5) eliminates degrees of freedom in the computation of patch distances. It seems logical to think these distances will be smaller, and so is the value of h^2. This issue is not properly addressed in [19,20], where the preselection threshold is heuristically selected, or in [12,23] either. Thus, we aim to compute:

$$E\{\widetilde{d}(x_i,x_j)\} = \frac{1}{N} E\left\{(\mathbf{c}_i-\mathbf{c}_j)^T\mathbf{X}^T\mathbf{X}(\mathbf{c}_i-\mathbf{c}_j)\right\} = \frac{1}{N}\mathrm{tr}\left(E\{\mathbf{X}\mathbf{d}\mathbf{d}^T\mathbf{X}^T\}\right), \tag{13}$$

where $\mathbf{d}=\mathbf{c}_i-\mathbf{c}_j$ and $\mathrm{tr}(\mathbf{A})$ is the trace of $\mathbf{A}$. From eq. (6), it follows:

$$\begin{aligned}
E\{\widetilde{d}(x_i,x_j)\} &= N^{-1}\mathrm{tr}\left(E\{\mathbf{X}\left((\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{w}\right)\left((\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{w}\right)^T\mathbf{X}^T\}\right)\\
&= N^{-1}\mathrm{tr}\left(\mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T E\{\mathbf{w}\mathbf{w}^T\}\mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\right)\\
&= N^{-1}\mathrm{tr}\left(\mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\left(E\{d(x_i,x_j)\}\mathbf{I}_M\right)\mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\right)\\
&= N^{-1}\mathrm{tr}\left(\mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\right)\cdot E\{d(x_i,x_j)\} = \frac{K}{N}\cdot E\{d(x_i,x_j)\},
\end{aligned} \tag{14}$$

where $\mathbf{w}=\mathbf{u}_i-\mathbf{u}_j$ and we have assumed that all pixels are uncorrelated. From eq. (5), it is easy to check that $K=\mathrm{tr}\left(\mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\right)$ exactly matches the number η of coefficients c describing the surface in eq. (4). Hence, the meaning of eq. (14) is that the effective value of h^2 has to be reduced to:

$$h^2_{\text{eff}} = \frac{\eta}{N}h^2. \tag{15}$$

As a final remark, it has been observed in [7] that the computation of $\widetilde{d}(x_i,x_i)$ will always yield 0, overweighting the central pixel $x_i\in\mathcal{N}_i$. Instead, we fix: $\widetilde{d}(x_i,x_i)\triangleq E\{\widetilde{d}(x_i,x_j)\}$ to avoid such bias.

3.4 Weighted distance functions

It is a common practice to compute weighted patch distances $d(x_i, x_j)$, as already suggested in [4]. The quadratic differences between each pair of corresponding pixels are pondered depending on its physical distance to the center of the patch $\mathcal{N}_i$:

$$d(x_i, x_j) = (\mathbf{u}_i - \mathbf{u}_j)^T \mathbf{R} (\mathbf{u}_i - \mathbf{u}_j), \tag{16}$$

where $\mathbf{R}$ is a diagonal matrix whose entries correspond to the n-dimensional kernel used to ponder the distances. Interestingly, this strategy nicely fits in our formalism: it is trivial to show that the expressions given in eqs. (7), (8), and (9) for the coefficients c remain exactly the same. The only difference relies on the way local averages are computed, taking into account the weighting kernel; for example:

$$\overline{u} = \sum_{x_j \in \mathcal{N}_i} \rho_j u(x_j) = \mathbf{1}^T \mathbf{R} \mathbf{u}; \quad \sum_{x_j \in \mathcal{N}_i} \rho_j = 1, \tag{17}$$

where $\mathbf{1}$ is an $N \times 1$ vector of all ones and ρ_j is the value of the multivariate kernel at each location $x_j \in \mathcal{N}_i$. If the kernel is separable, all local averages can be computed as separable convolutions. Of course, particularizing the kernel to the case $\rho_j = 1/N, \forall j$ yields the unweighted patch distance.

Obviously, this calculation affects the statistical characterization of distances. A similar development to that in eq. (14) proves:

$$E\{\widetilde{d}(x_i, x_j)\} = \mathrm{tr}\left(\mathbf{R}\mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\right) \cdot E\{d(x_i, x_j)\}. \tag{18}$$

Therefore, the effective value of h has to be:

$$h_{\text{eff}}^2 = \mathrm{tr}\left(\mathbf{R}\mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\right) \cdot h^2. \tag{19}$$

Although this expression is not so simple as eq. (15), its value can be precomputed, and our model does not suffer any substantial modification.

3.5 Hierarchical preselection based on patch distances

Another advantage of our method is that it is compatible with a hierarchical preselection strategy similar to [18]; indeed, it is also based on true patch distances, with the advantage this implies. Suppose we establish a threshold μ the distance $d(x_i, x_j)$ has to lay within for x_j to be considered in the WA. We proceed as follows:

(1) If $\widetilde{d_0}(x_i, x_j) = (c_{0i} - c_{0j})^2 > \mu \cdot h^2_{\text{eff},0}$, the voxel is discarded: the selection is based on the estimated distance $\widetilde{d_0}$ and the effective parameter $h^2_{\text{eff},0}$ for the Taylor series of order 0. In case the test is not passed, it has been necessary to compute only one difference. If the voxel is not discarded:

(2) The test is repeated for order 1, and the voxel is discarded if $\widetilde{d_1}(x_i, x_j) > \mu \cdot h^2_{\text{eff},1}$. Besides, $\widetilde{d_1}$ can be easily computed from $\widetilde{d_0}$ using eq. (12).

(3) If higher orders are considered, the test can be repeated with a threshold $\mu \cdot h^2_{\text{eff},l}$. At each step, we have actual estimates $\widetilde{d_l}(x_i, x_j)$ of $d(x_i, x_j)$; in case the voxel is not discarded in the final level of the hierarchy, the estimated distance to calculate the weight has been computed “for free”.

3.6 About the computation of exponential weights

The calculation of exponentials is a time consuming task even with modern hardware. We have found that a rational approximation to the negative exponential in eq. (2) can achieve a non-negligible speedup (a constant factor nearly 1.3) with virtually identical results. In practice, we use:

$$\exp\left(-t^2\right) \simeq \begin{cases} \dfrac{1}{1+t^2}\dfrac{2-t^2}{2} + \dfrac{1}{\left(1+t^2\right)^2}\dfrac{t^2}{2}, & t^2 < 1+\sqrt{3}; \\ \\ 0, & \text{otherwise.} \end{cases} \tag{20}$$

3.7 Summary

Our methodology can be outlined as follows:

(1) The local features c in eqs. (7), (8), or (9) are precomputed calculating local averages as separable convolutions like eq. (17). These features are stored in contiguous memory locations for each pixel.

(2) For each $x_j \in \Omega_i$, the differences in the features space are sequentially compared with the preselection threshold for each truncation order.

(3) In case the pixel passes all preselection tests, the distance in the features space is normalized using the effective h^2 value of eq. (19).

(4) Accordingly, the WA coefficient is computed using eq. (20) and used to update the sum of eq. (1).

4 Experimental results and discussion

4.1 Setting-up of the experiments

Like many other previous works, we use the realistic MRI phantom described in [24] as a ground truth [1]. It is a $181 \times 217 \times 181$ 3-D data-set [2] with 1mm^3 resolution simulating a noise free T1-MRI volume. To simulate a realistic MRI, this phantom is contaminated with Rician noise of desired SNR; from the noise-free image $v(x_i)$, the noisy image is obtained as:

$$u(x_i) = \left|(v(x_i) + \eta_c(x_i)) + j\eta_s(x_i)\right|, \tag{21}$$

where $\eta_{c,s}(x_i)$ are uncorrelated Gaussian processes with variance σ^2. To remove the bias induced by Rician noise, we use the approach suggested in [11,12]. The squared value of $u(x_i)$ is estimated using eq. (1), so that NLM becomes:

$$\widehat{u}(x_i) = \left(\max\left\{\sum_{x_j \in \Omega_i} w(x_i, x_j) u^2(x_j) - 2\sigma^2,\ 0\right\}\right)^{1/2}. \tag{22}$$

This methodology is well accepted and has been tested in a number of recent works [8–10]. Like in [7], we compute weighted patch distances by introducing a kernel $\mathbf{R}$ corresponding to a separable Gaussian with isotropic variance 1.

Finally, we keep the truncation order of eq. (4) equal to 1, since using order 0 produces an excessive over-blurring (see Fig. 1). On the other hand, order 2 approximations do not carry on a systematic improvement of the results, and

[1] Available online: `http://mouldy.bic.mni.mcgill.ca/brainweb/`.

[2] Although the above derivations have been shown for the 2-D case to avoid an excessively verbose typesetting, note the extension to 3-D is done in the obvious way just by adding the corresponding terms in the 'z' coordinate.

they unfruitfully increase computation times.

[Fig. 1 about here.]

4.2 Algorithms compared

At the first stage, we extensively compare three variants of NLM in terms of their filtering performance and parameter sensitivity [3]:

(1) Our own implementation of the original NLM as described in [4]. To achieve a fair comparison, we have adapted it to Rician noise (see above) and used eq. (20) to compute the weights. This is our standard for comparison in the first set of experiments, and will be referred to as **NLM**.
(2) The method proposed in this paper without hierarchical preselection. It will be named **PFNLM** after "Polynomial-Fit" NLM.
(3) The method proposed in this paper with preselection threshold $\mu = 1$ (this value has been empirically fixed), namely **PFNLM-1.0**.

4.3 Quality measures

The outcome produced by the filter in each case is compared to the noise-free ground-truth using three different similarity measures:

(1) The Root Mean Squared Error (RMSE) between the images.
(2) The Structural Similarity Index (SSIM) described in [25]. As opposed to the RMSE, this index accounts for the similarity between image structures and not between grey levels. It is bounded between 0 (worst quality)

[3] Source code available at `http://www.lpi.tel.uva.es/~atriveg/nlm.tar.gz`.

and 1 (identical to ground-truth).

(3) The Quality Index based on Local Variance (QILV) [26], which is more sensitive to the blurring of image edges. It is bounded between 0 (worst quality) and 1 (identical to ground-truth).

While RMSE and SSIM describe how well noise is removed, QILV describes how well structures are preserved, providing complimentary information.

4.4 Choosing the optimal parameters

[Fig. 2 about here.]

The first parameter to set is h (note that once h is fixed for NLM the effective h_{eff} for PFNLM is immediately obtained via eq. (19)). Values ranging from 1.0σ to 1.2σ are proposed in [7], and for T1 images $h = 1.2\sigma$ is suggested. However, we have empirically tested (see Fig. 2) that $h = \sigma$ is more appropriate. We conjecture this difference is due to the alternative way Rician bias correction is accomplished in [7].

As expected, the original NLM is not able to properly remove the noise for too small values of h^2. This translates in Fig. 2 in worse values of both RMSE and SSIM in the left column (remember these indices measure how well noise is removed). If we increase h^2, NLM shows a better behavior and both RMSE and SSIM are improved. However, this parameter cannot be arbitrarily increased, as illustrated in the last row of Fig. 2 for the QILV index (which accounts for structures/edges preservation): if h^2 is increased to $1.2\sigma^2$, the QILV index is clearly worsened for NLM. As a summary, we may conclude that $h^2 = \sigma^2$ (i.e. $\beta = 1$) is an adequate trade-off between noise removal and edge preservation, at least for the kind of images under consideration. If we fix our attention in the optimal case $\beta = 1$, we can argue the three approaches produce a similar blurring of structures (a similar QILV index) but PFNLM produces *cleaner* images in terms of a smaller RMSE or higher SSIM. Note that PFNLM-1.0 does not seem to carry on any particular advantage, although the corresponding QILV curve is slightly above those of NLM and PFNLM indicating that it could be preferable for edge preservation. Comparing again the behavior for different h^2, PFNLM and PFNLM-1.0 seem to be more robust to the election of h^2, which is an important advantage in practice: small deviations in the estimation of σ^2 (note that this parameter is not known in general) will not drive to an important deterioration of the output of our filter.

[Fig. 3 about here.]

The remaining parameters to set are the radius M of Ω_i and the radius B of $\mathcal{N}_i$, Fig. 3 showing representative results to this respect. The conclusions drawn from the three quality indices are similar: increasing M improves denoising until $M = 5$ is reached and only a marginal advantage is expected. Note, however, that the QILV measure is decreased (i.e. the structural information is partially blurred) in high-SNR scenarios for PFNLM with $M \geq 3$, although RMSE and SSIM are still amended. With regard to the radius of the comparison patch, using $B > 1$ notably worsens the output quality (except for NLM with input RMSE of 25); since the computational complexity increases as $(2B + 1)^n$, it seems reasonable to choose $M = 5$ and $B = 1$ as optimal parameters (as has been done in all our experiments, including those of Fig. 2). It is worth noting this conclusion is in complete agreement with the results previously reported independently in [7] and [19].

4.5 Filtering performance for different powers of noise

Figs. 2 and 3 already suggest that PFNLM without preselection compares favorably to the other algorithms. For the optimal parameters, the meaning of Fig. 2 (for QILV) is that the three algorithms produce a similar blurring (at the sight of Fig. 1, the smoothing seems negligible). With regard to noise removal, Fig. 3 suggests that PFNLM outperforms the original NLM unless a very high SNR is considered, in which case a certain over-blurring may arise. On the other hand, the hierarchical preselection slightly worsens the results with respect to PFNLM, although it seems to palliate to some extent the blurring for very high SNR. Fixing our attention in Fig. 2 (center), it seems clear that PFNLM is more effective for noise removal than the original NLM (with the same blurring) for practically all the SNR range. In the next sections we provide some representative examples in this sense.

4.6 More on the filtering performance with different NLM approaches

In this section we aim giving some additional insights into the numerical results presented above. For the sake of completeness, we compare in what follows three additional techniques to sum up to those in section 4.2, which, following the discussion in section 2.2, are the most closely related to our own:

- The fast NLM technique described in [19] for MRI denoising [4], with the parameters suggested by the authors (note they agree with those in section 4.4, so the comparison is fair). We have not used the block-wise implementation, since it is known to worsen the filtering accuracy [19].

[4] The method can be tested online: `https://www.irisa.fr/visages/benchmarks`.

- The wavelet sub-band denoising NLM in [15]. This scheme [5] is mostly based on the previous one, but wavelet analysis is used to improve the performance. Though an adaptive implementation to cope with parallel acquisition techniques is described in [8], we have used the non-adaptive version since such kind of images are out of the scope of the present paper.
- The NLM implementation for textured images in [18] [6]. In this case we cannot attain a fair comparison in terms of performance for two main reasons:

(1) The software is not designed for Rician noise, and we cannot adapt it using eq. (22) since the source code is not available.

(2) The binaries provided are only for a 2-D case, which will clearly bias the results in favor of the remaining approaches compared.

For all these methods, illustrative examples are shown in Fig. 4.

[Fig. 4 about here.]

Comparing the original NLM and our novel PFNLM, it is clear that the numerical differences reported translate into a very significant melioration of the visual quality (see zoomed regions). As expected, the method proposed in [18] yields the worst results, since it is not intended to work with 3-D MRI images. With regard to the method in [19], it seems to provide a better performance than the original NLM (as already stated by the authors therein), but our approach still outperforms it: for RMSE=20 the difference is quite subtle, but it is still visible that the edges are better preserved with our PFNLM (please, see the electronic version of this manuscript). For input RMSE=35, the meliora-

[5] Code available: `http://personales.upv.es/jmanjon/denoising/arnlm.html`.

[6] `http://www.cs.berkeley.edu/~brox/resources/nlmeans_brox_tip08Linux64.zip`

tion with PFNLM is more evident: the structures of interest become enhanced with our PFNLM, while the background noise is barely noticeable; with the approach in [19], on the contrary, the structure is less clearly visible and mixed up with a granulated background. Finally, the wavelet method in [8] provides qualitative results very similar to [19]. For input RMSE=35 their visual quality is virtually identical, while for RMSE=20 the wavelet method seems to preserve the edges slightly better than [19]. Nonetheless, it seems to artificially enlarge the dark regions when compared to [19] and our PFNLM (see the rightmost part of the zoomed region).

To conclude this analysis, Table 1 shows the corresponding quality indices for the algorithms compared in Fig. 4; according to our comments above, the QILV index for our PFNLM is higher than for [19], meaning the former produces less blurring. The wavelet method [8] is able to improve the RMSE of the output, even outperforming PFNLM for input RMSE=20. However, and according to the observation in the previous paragraph, the QILV index is worse than for PFNLM: while wavelet denoising is able to better preserve the grey levels of the image, the advantage of our method when it comes to the subject of structure preservation is still clear, both qualitatively (Fig. 4) and quantitatively (Table 1). For low SNR, the novel PFNLM outperforms it for all indices. As a final remark, we want to stress that the outcomes obtained for low SNR are virtually identical (except for a certain difference in the output RMSE) for both [19] and [8].

[Table 1 about here.]

4.7 *Speedup achieved by PFNLM*

[Fig. 5 about here.]

In this section we study the computation times for each method. The three algorithms in section 4.2 have been identically coded in C++ with multi-thread based on the ITK libraries [7] [27], except for the specific parts of each one (eq. (20) is used in all cases). To avoid any influence of hardware limitations, we conducted our experiments in a 32 GB RAM, 16 Intel© CPU machine running a CentOS Enterprise-class linux distribution. All additional processes other than those associated to common tasks of the system were suspended to perform the tests. The execution times we provide were measured with the `time` command, reporting:

1. (1) The *user* time, i.e. the amount of CPU time (for all CPU) consumed by the user segment of the process (without considering system calls).
2. (2) The *real* time, i.e. the actual duration of the process; if the computation is done in parallel by several CPU, it can be drastically reduced.

The speedup in each case is computed as the time measured for the original NLM over the time measured for the algorithm being compared. The results are shown in Fig. 5. The speedup (*user* time) increases linearly with the radius of the search window, and for the optimum $M = 5$ it grows over one order of magnitude. As expected, voxel preselection achieves an additional acceleration, although it is not as important as in [19] since the actual computation of patch distances is less time consuming. If we now consider the *real* time, the speedup is even more important, reaching a factor 20. This behavior is easy to explain:

[7] `http://www.itk.org`.

when multiple threads run in parallel, their execution may (and it does) take quite different times; when the last thread is running alone, most CPU are not used. Since NLM is slower, additional CPU remain unused longer, so it makes less use of multi-threading. The advantage of PFNLM-1.0 in this case is only noticeable for $M = 4, 5$, but these are the useful scenarios.

The speedup achieved for $M = 5$ is over 10 even in the worst case. We estimate the norm of a $(2B + 1)^3 = 27$ components vector as the norm of a 4 components vector, so the predicted speedup would be only $27/4 = 6.75$. This additional acceleration can also be explained: as mentioned above, image features describing patches are always stored in contiguous locations, so accessing the memory is more efficient. With NLM, the pixels in the comparison window will not be contiguous in memory, producing cache failures. This is an intrinsic problem of NLM and does not rely on our implementation. With less powerful computers (with smaller caches) this problem will be even accentuated, so that PFNLM should be especially advantageous.

With respect to [19], we cannot attain a fair comparison since no binaries are provided for testing. As a guidance, the reported computation time for the whole volume is 43'12", i.e. 10.5 times slower than our approach (4'6"). Taking into account that the acceleration with the block-wise implementation is 6.2 in the best case [19, Table IV], our approach would be at least **1.7 times faster**; yet, it should be noted that block-wise NLM yields less accurate filtering outcomes, so the performance improvement of our method in that case would be even more important. When compared to the wavelet based approach [8] (source code is available), our implementation becomes **1.8 times faster** without preselection (over 2 with preselection), similar to the value estimated for [19].

Finally, for [18], we have used a 2-D search window of 35×35 pixels, which implies roughly the same number of averages as our 3-D search window of $11 \times 11 \times 11$ voxels. The computation time for the whole volume is 14'30", so in the worst case our method is still **3.5 times faster**. Moreover, this comparison is rather conservative: when extending the 2-D algorithm to 3-D, the computation should take longer than the sum of the times for each 2-D slice; for example, we have neglected for [18] the overload due the computation of 3-D patch distances, so the speedup of our method compared to [18] will be actually more than 3.5 times.

4.8 In vivo experiments

The major strengths of our proposal show up in low SNR scenarios. An application of paramount importance in this sense is diffusion MRI: in this modality, the preferential directions of water diffusion are probed by means of strong sensitizing gradients in the magnetic field of the scanner. Such gradients translate in severe attenuations of the received T2 echoes for the same noise power, dramatically worsening the final SNR. To illustrate this situation, we have gathered a real diffusion data set scanned in a 1.5 Tesla GE Echospeed system, comprising six independent gradient directions with $b = 700s/\mathrm{mm}^2$. The diffusion tensor at each voxel is estimated as in [28]. Although nowadays protocols use a larger number of sensitizing gradients and fit the diffusion tensor in a more robust manner via LS [29], we deliberately chose this reduced set: otherwise, the extra regularization introduced by LS would hinder the differences in the actual performances of the filters compared.

Fig. 6 shows an axial slice of the data-set, where it may be checked that the power of noise is far larger than that in T1 or T2 volumes. This noise power has to be estimated in this case, for which we have used the method proposed in [30, eq. (12)] as a good trade-off between simplicity and performance. Remarkably, we obtain a very similar result , $\sigma = 65$, as that provided by the online tool that implements the method in [19]. As it was predictable from the above experiments, our PFNLM outperforms the other methods in terms of the visual quality achieved. Especially, the structures marked with arrows in the figure are better denoised, and their contours clearly enhanced compared to (b) and (d). Also, given the low SNR of this data-set, the results for [19] and [8] are visually identical and hence they are not duplicated.

[Fig. 6 about here.]

However, diffusion MRI are used directly not very often, and the most interesting information is provided after the diffusion tensor is estimated: its principal eigenvector can then be tracked to recover entire fiber bundles connecting regions of interest in the brain. We have conducted a final experiment in this sense using 3-D Slicer[8]. The seeding points from which tracking is started have been manually placed in the cerebellar peduncle to obtain the results in Fig. 7.

[Fig. 7 about here.]

Apart from the evident noisy behavior of the fibers, due to the very reduced number of gradient directions, our PFNLM method yields the smoothest pathways. What is more important, the corresponding fiber bundles are also anatomically more meaningful: the curvature of the pyramidal tract is more

[8] `http://http://www.slicer.org/`

correctly recovered after PFNLM denoising (1), while the integrity of the fiber bundles is also better preserved, so that most of them connect with the cortex region. Finally, the integrity and orientation of the middle cerebellar peduncle (2) is adequately conserved with PFNLM filtering (compare the number of fibers traced in each case).

5 Conclusions

Our NLM implementation is based on estimating the similarity between the pixels in a features space. Since these features are computed in a robust manner with LS, we are even able to outperform the conventional NLM for realistic SNR. The idea of using polynomial fitting has been already explored in [14], but in that work it is the search window Ω_i which is modeled as a polynomial surface to generalize the WA of NLM to a higher order model (and hence the computational complexity is even increased). Therefore, the idea of measuring patch distances in the space of the Taylor series coefficients is a novel contribution in our paper.

The speedup achieved by our method with standard parameters is at least one order of magnitude, and can be even higher if hierarchical preselection is implemented (although PFNLM-1.0 slightly worsens the filtering accuracy with respect to PFNLM, it is yet preferable to NLM for low SNR, see Fig. 3). In case the filter is run in a multiple CPU machine, the acceleration can reach a factor near 20 even without preselection. Voxel preselection techniques like that in [19] can achieve a speedup of at most 5 to 7 in the same conditions (i.e. with multi-threading), so the advantage of our technique remains clear. Moreover, combining our proposal with the block-wise implementation in [19] (at the expense of worsening its accuracy), an acceleration rate over two orders of magnitude could be attained.

With respect to the improvement in the filtering outcomes, our approach compares favorably to the original NLM for the specific case of MRI images. Although an extensive comparison with all the recently introduced methodologies is not feasible due to the lack of available code, we have shown enlightening results for the advanced NLM techniques closest to our own work, i.e. [8], [18], and [19], that clearly suggest our proposal is certainly advantageous. Besides, we have issued open-sourced code of our methods, so further comparisons should be easy to perform with future research results.

Finally, the potential of our method depends on the ability to describe image patches with its mean value and gradient. For textured images (such as photographs), this first order approximation might not suffice, being necessary to consider order 2 or higher. Although a certain speedup is achieved in this case, it might be only marginal. In these scenarios the preselection technique proposed in [18] for textured images should be preferable. Note, however, that our approach is mainly oriented to MRI, which are inherently non-textured. The ability of PFNLM to deal with very noisy data is specially interesting for diffusion MRI, which typically exhibit a poor SNR; nevertheless, and as pointed out in [10], the proper denoising of diffusion data cannot be carried out as a simple channel by channel operation, but instead has to take into account the peculiarities of this kind of data. The inclusion of the diffusion model within our framework will be for sure an important future research line.

Acknowledgments.

Work partially funded by grants R01 MH074794, R01 R01MH092862, and P41 RR013218 from the NIH; SAN126/VA33/09 and VA0339A10-2 (Junta de Castilla y León), GRS 292/A/08 (Consejería de Sanidad de Castilla y León), and TEC2007-67073/TCM (Ministerio de Educación y Cultura, Spain – European Regional Development Fund). The corresponding author is supported by grant number FU2010-17982 from the Fulbright Program/Spanish Ministry of Education.

References

[1] I. Bankman (Ed.), Handbook of Medical Imaging: Processing and Analysis Management, Academic Press; Elsevier Science & Technology Books, 2000.

[2] G. Gerig, O. Kübler, R. Kikinis, F. Jolesz, Nonlinear anisotropic filtering of MRI data, IEEE Transactions on Medical Imaging 11 (2) (1992) 221–232.

[3] R. Nowak, Wavelet-based Rician noise removal for Magnetic Resonance Imaging, IEEE Transactions on Image Processing 8 (10) (1999) 108–1419.

[4] A. Buades, B. Coll, J. Morel, A review of image denoising algorithms, with a new one, Multiscale Modelling and Simulation 4 (2) (2005) 490–530.

[5] H. Xu, J. Xu, F. Wu, On the biased estimation of Nonlocal Means filter, in: Proceedings of IEEE International Conference on Multimedia and Expo, 2008, pp. 1149–1152.

[6] P. Coupé, P. Hellier, C. Kervrann, C. Barillot, Nonlocal Means-based speckle filtering for Ultrasound images, IEEE Transactions on Image Processing 18 (10) (2009) 2221–2229.

[7] J. Manjón, J. Carbonell, J. Lull, G. García-Martí, L. Martí-Bonmatí, M. Robles, MRI denoising using Non-Local Means, Medical Image Analysis 12 (2008) 514–523.

[8] J. V. Manjón, P. Coupé, L. Martí-Bonmatí, D. L. Collins, M. Robles, Adaptive Non-Local Means denoising of MR images with spatially varying noise levels, Journal of Magnetic Resonance Imaging 31 (2010) 192–203.

[9] M. Descotaux, N. Wiest-Daesslé, S. Prima, C. Barillot, R. Deriche, Impact of Rician adapted Non–Local Means filtering on HARDI, in: Medical Image Computing and Computer Assisted Intervention, Vol. 5242 of Lecure Notes in Computer Science, Springer–Verlag, 2008, pp. 122–130.

[10] A. Tristán-Vega, S. Aja-Fernández, DWI filtering using joint information for DTI and HARDI, Medical Image Analysis 14 (2) (2010) 205–218.

[11] N. Wiest-Daesslé, S. Prima, P. Coupé, S. Morrissey, C. Barillot, Rician noise removal by Non–Local Means filtering for low signal–to–noise ratio MRI: Applications to DT–MRI, in: Medical Image Computing and Computer Assisted Intervention, Vol. 5242 of Lecture Notes in Computer Science, Springer–Verlag, 2008, pp. 171–179.

[12] S. Aja-Fernández, K. Krissian, An unbiased Non-Local Means scheme for DWI filtering, in: Proceedings of the Medical Image Computing and Computer Assisted Intervention: Workshop on Computational Diffusion MRI, 2008, pp. 277–284.

[13] N. Wiest-Daesslé, S. Prima, P. Coupé, S. Morrissey, C. Barillot, Non-Local Means variants for denoising of diffusion-weighted and diffusion tensor MRI, in: Medical Image Computing and Computer Assisted Intervention, Vol. 4792 of Lecture Notes in Computer Science, Springer–Verlag, 2007, pp. 344–351.

[14] P. Chatterjee, P. Milanfar, A generalization on Non-Local Means via kernel

regression, in: Proceedings of the International Society of Optics and Photonics (SPIE) Conference on Computational Imaging VI, Vol. 6814, San Jose, USA, 2007, pp. 1–8.

[15] P. Coupé, P. Hellier, S. Prima, C. Kervrann, C. Barillot, 3D wavelet subbands mixing for image denoising, International Journal of Biomedical Imaging (2008) 1–8 DOI: 10.1155/2008/590183.

[16] L.Li, L. Kong, Image denoising based on non-local means with Wiener filtering in wavelet domain, in: Proceedings of Intelligent Information Hiding and Multimedia Signal Processing, 2009, pp. 471–474.

[17] D. Van De Ville, M. Kocher, SURE-based Non-Local Means, IEEE Signal Processing Letters 16 (11) (2009) 973–976.

[18] T. Brox, O. Kleinschmidt, D. Cremers, Efficient Nonlocal Means for denoising of textural patterns, IEEE Transactions on Image Processing 17 (7) (2008) 1083–1092.

[19] P. Coupé, P. Yger, S. Prima, P. Hellier, C. Kervrann, C. Barillot, An optimized blockwise Non Local Means denoising filter for 3D magnetic resonance images, IEEE Transactions on Medical Imaging 27 (4) (2008) 425–441.

[20] J. Orchad, M. Ebrahimi, A. Wong, Efficient Nonlocal-Means denoising using the SVD, in: IEEE International Conference on Image Processing (ICIP), 2008, pp. 1732–1735.

[21] M. Mahmoudi, G. Sapiro, Fast image and video denoising via nonlocal means of similar windows, IEEE Signal Processing Letters 12 (12) (2005) 839–842.

[22] C.-C. J. Thapanich, Tanaphol; Kuo, An adaptive nonlocal means scheme for medical image denoising, in: Proceedings of the International Society of Optics and Photonics (SPIE) Conference on Medical Imaging, Vol. 7623, San Diego, USA, 2010, pp. 76230M–76230M–12.

[23] T. Tasdizen, Principal components for Non-Local Means image denoising, in: IEEE International Conference on Image Processing (ICIP), 2008, pp. 1728–1731.

[24] D. Collins, A. Zijdenbos, V. Kollokian, J. Sled, N. Kabani, C. Holmes, A. Evans, Design and construction of a realistic digital brain phantom, IEEE Transactions on Medical Imaging 17 (3) (1998) 463–468.

[25] Z. Wang, A.-C. Bovik, H.-R. Sheikh, E.-P. Simoncelli, Image quality assessment: form error visibility to structural similarity, IEEE Transactions on Image Processing 13 (4) (2004) 600–612.

[26] S. Aja-Fernández, R. San-José-Estépar, C. Alberola-López, C.-F. Westin, Image quality assessment based on local variance, in: Proceedings of the IEEE Engineering in Medicine and Biology Society (EMBS) Conference, New York, USA, 2005, pp. 4815–4818.

[27] L. Ibañez, W. Schroeder, L. Ng, J. Cates, The ITK Software Guide, Kitware, Inc. ISBN 1-930934-15-7, http://www.itk.org/ItkSoftwareGuide.pdf, 2nd Edition (2005).

[28] C.-F. Westin, S. Maier, H. Mamata, A. Nabavi, F. Jolesz, R. Kikinis, Processing and visualization for diffusion tensor MRI, Medical Image Analysis 6 (2001) 93–108.

[29] R. Salvador, A. Peña, D.-K. Menon, T.-A. Carpenter, J.-D. Pickard, E.-T. Bullmore, Formal characterization and extension of the linearized diffusion tensor model, Human Brain Mapping 24 (2005) 144–155.

[30] S. Aja-Fernández, A. Tristán-Vega, C. Alberola-López, Noise estimation in single and multiple coil MR data based on statistical models, Magentic Resonance Imaging 27 (2009) 1397–1409.

List of Figures

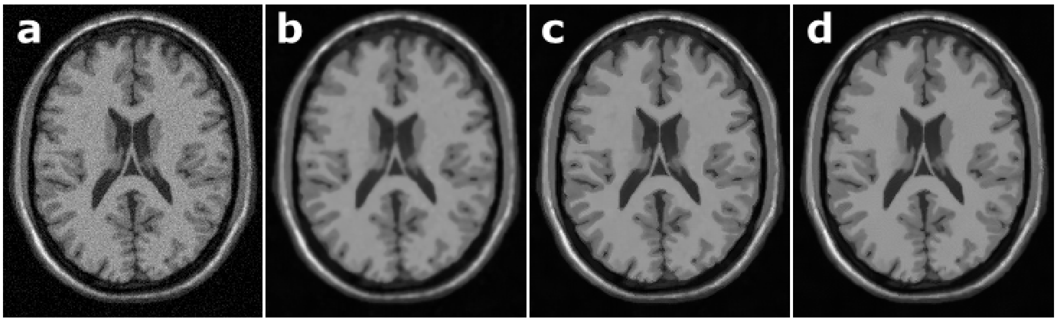

Fig. 1. Performance of different truncation orders: original noisy image (a), our proposal with order 0 (b), with order 1 (c) -no improvement is achieved with order 2-, and original NLM (d). Order 0 truncation produces over-blurring and hence it is not adequate (b), but the result for order 1 (c) is undistinguishable from the traditional NLM (d).

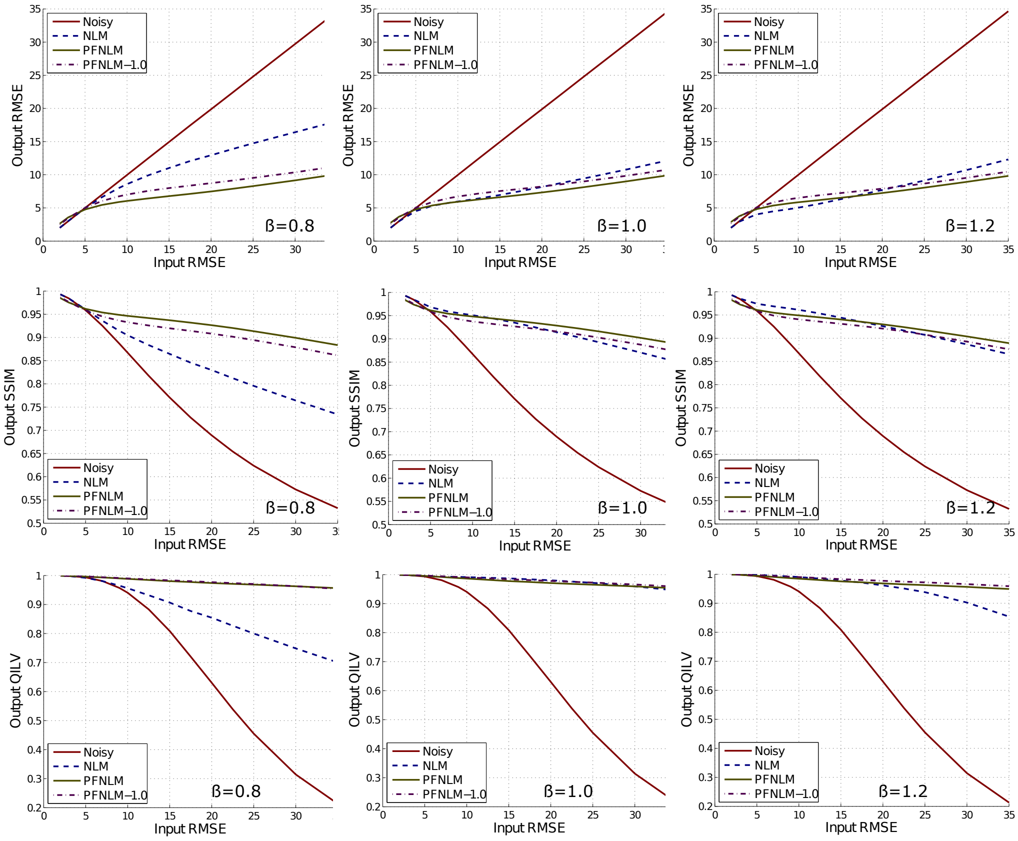

Fig. 2. Quality indices versus input RMSE for the three algorithms compared and for: (left) $h = 0.8\sigma$; (center) $h = 1.0\sigma$; (right) $h = 1.2\sigma$. (Top) RMSE; (middle) SSIM; (bottom) QILV. The indices corresponding to the noisy images are represented as well for the sake of comparison.

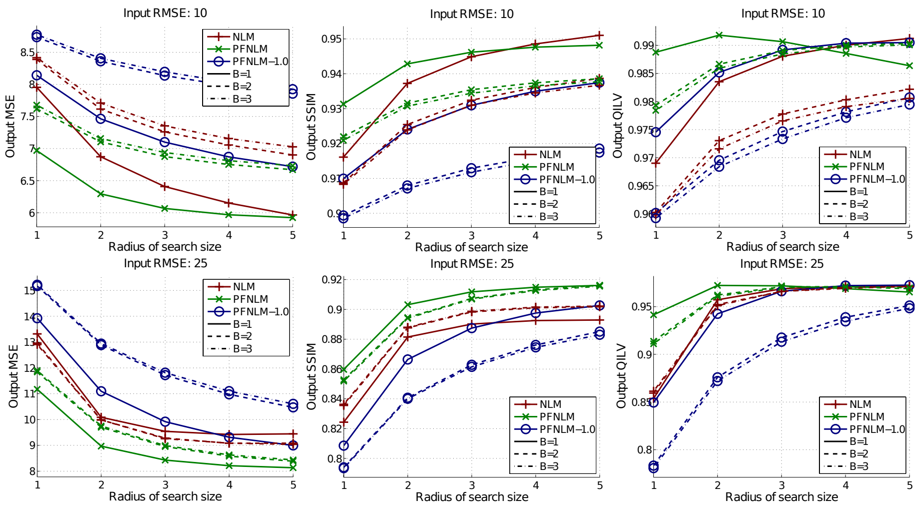

Fig. 3. Output quality indices as a function of the radius of the search window (M) for different radii of the comparison patch (B). Results are shown for a high SNR scenario (top) and for low SNR (bottom).

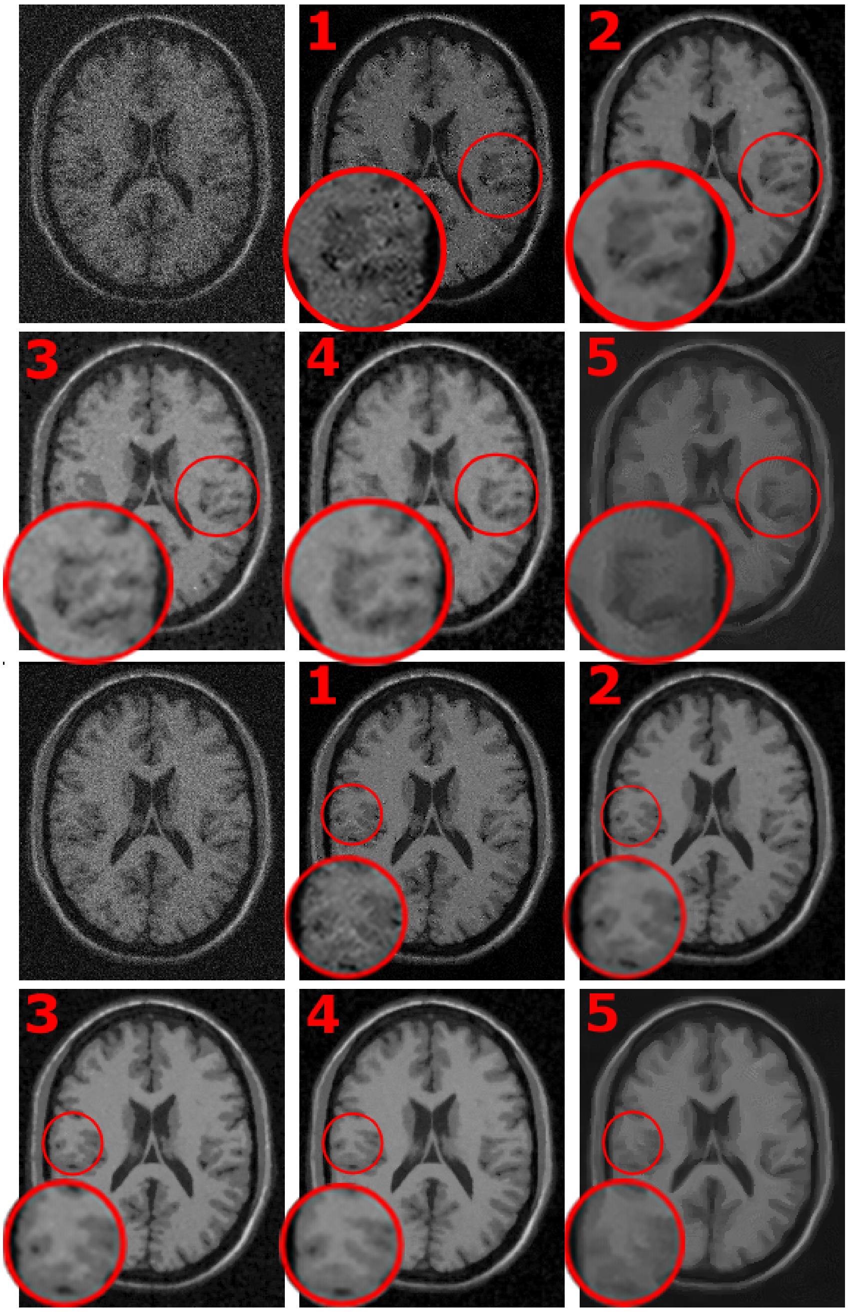

Fig. 4. Central axial slice of the T1 phantom used in the synthetic experiments, contaminated with Rician noise with input RMSE of 35 (top) or 20 (botton). For guidance, the original noisy image is shown together with the image filtered with: the original NLM with Rician bias correction (**1**); our novel PFNLM method (**2**). For the sake of comparison, we show also: the fast NLM method for MRI described in [19] (**3**), the wavelet sub-band MRI denoising method in [8] (**4**), and the fast NLM approach for textured images denoising in [18] (**5**).

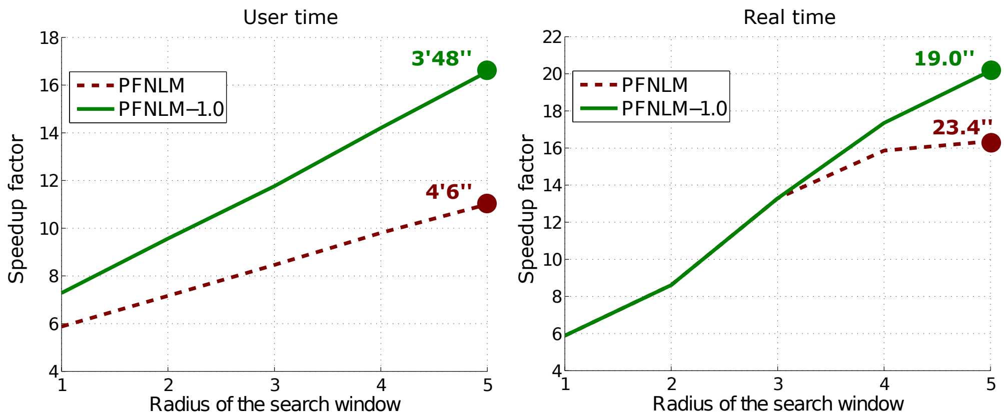

Fig. 5. Speedup, with respect to the original NLM, achieved by PFNLM and PFNLM-1.0, for: (left) One single CPU; (right) A 16 core machine. The absolute execution times are given for the optimal value $M = 5$ as a guidance.

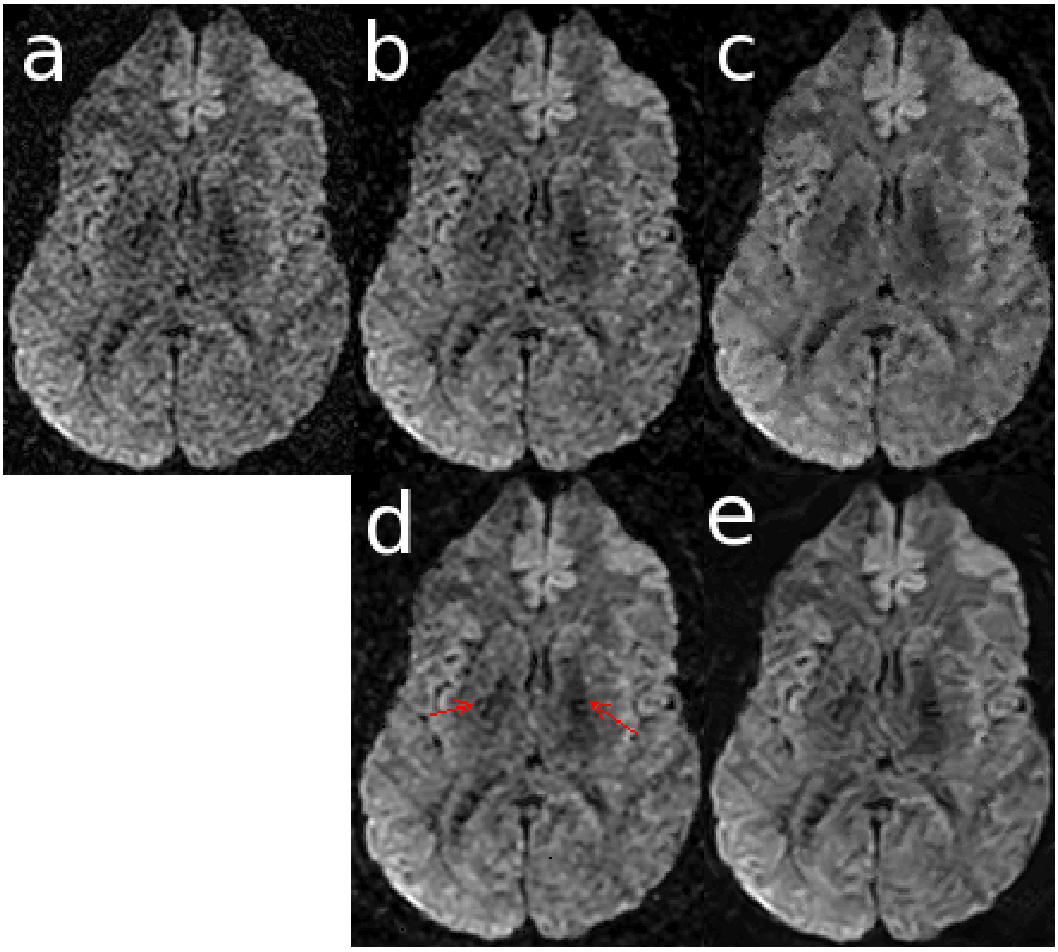

Fig. 6. An axial slice of the diffusion MRI volume acquired (an arbitrary gradient direction is shown in (a)), together with the NLM (b) and PFNLM (c) filtered versions of this same volume. The algorithms in [19] (that in [8] yields virtually identical results) and [18] are respectively shown in (d) and (e) for the sake of comparison.

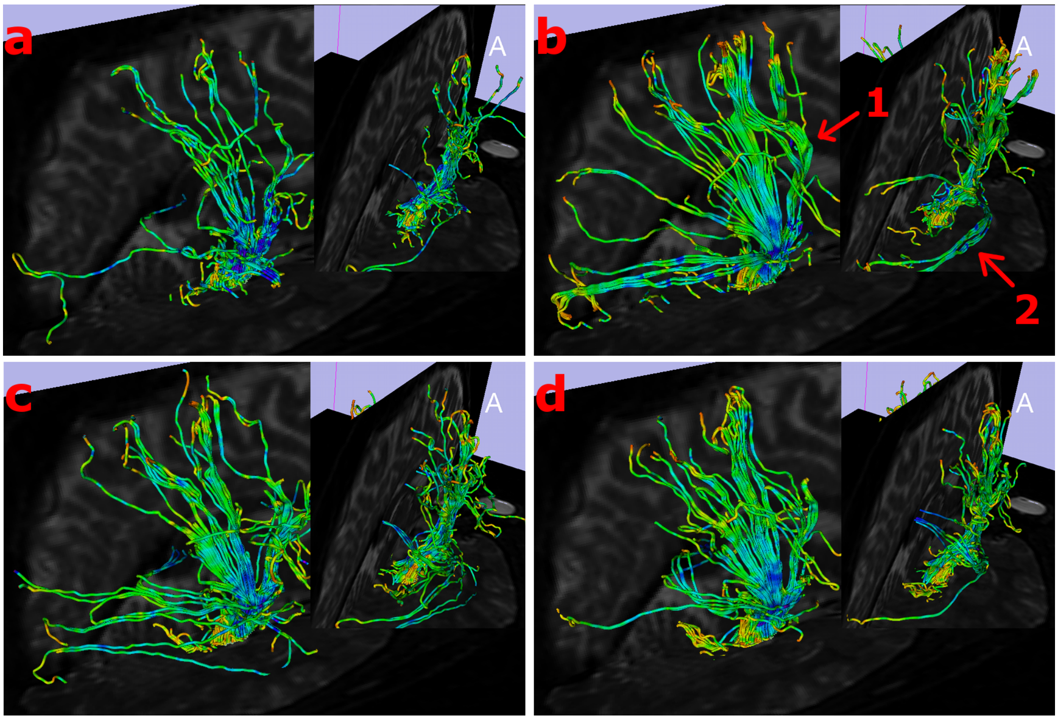

Fig. 7. Fiber tracking obtained after estimating the diffusion tensor from data filtered with (a) NLM, (b) PFNLM, (c) the approach in [19] (that in [8] yields virtually identical results), and (d) the approach in [18]. Seeding points have been placed in the cerebellar peduncle. The fiber bundles have been colored according to the fractional anisotropy (normalized variance of the eigenvalues of the diffusion tensor) at each location.

List of Tables

Table 1
Performance indices obtained for the experiments shown in Fig. 4. Regarding the values in italics for [18], the comparison cannot be considered fair for the reasons discussed in the text.

input	RMSE=35			RMSE=20		
output	RMSE	SSIM	QILV	RMSE	SSIM	QILV
NLM	12.5	0.85	0.95	8.5	0.92	0.98
PFNLM	**10.0**	**0.90**	**0.95**	7.4	**0.93**	**0.98**
[19]	12.3	0.87	0.93	7.4	0.93	0.96
[8]	10.3	0.88	0.93	**7.2**	0.93	0.96
[18]	*30.1*	*0.77*	0.25	*20.0*	*0.85*	*0.58*