Documentation of the NLM denoising plug-in for ImageJ

Julien Pontabry

June 3, 2015

Abstract

Noise is one the first artefact caused by acquisition system. The non-local means (NLM) denoising plug-in for ImageJ is designed to remove the noise component from images and this document aims to be a guide for users.

Contents

1 Introduction

With any measurement provided by any acquisition system comes a noise component. By restoring original signal, i.e. detecting and removing noise component (denoising), further processing and analysis tasks on that signal might be simplified and/or more relevant. The same applies to images.

There are many ways to denoise images. The easiest one is to acquire multiple images of the same object at the same time and average them. The idea is to take advantage of the statistical fluctuation of the noise to attenuate it. Nevertheless, it assumes the object under study is not moving, at least for an instant. If only one image is available, one can then use denoising algorithms. The literature about such algorithms is huge but here is a quick overview of methods: 1) local smoothing, such as the Gaussian smoothing model (Lindenbaum et al. 1994); 2) frequency domain filtering, such as the Wiener filter (Yaroslavsky 1985) (this filter can also be used for deconvolution) and 3) statistical neighbourhood approaches such as the DUDE algorithm (Ordentlich et al. 2003). For a more detailed review of denoising methods, please refer to Buades et al. (2005).

During the last decade, new methods have increased the denoising efficiency. For instance, the non-local means (NLM) denoising method is one the most powerful and the most used technique to denoise images (e.g. see Buades et al. (2005) for general images, Coupé et al. (2008) for medical images and Boulanger et al. (2010) for biological images). This document is meant to describe the use the NLM denoising plug-in for ImageJ. Despite it can be applied to any images, the main purpose of this implementation was about confocal microscopy image denoising.

The rest of this document is subdivided into five sections. First, the origin of noise and its modelling in images are defined.

Then, the mathematics principles of the NLM denoising technique are briefly explained. Later, the graphical user interface (GUI) of the plug-in is described. Finally, four use cases are demonstrated.

2 Noise model

Noise is inherent to the image acquisition process. The main source of noise is caused by the quantum nature of light and is called shot noise. When the number of particles that carry energy (photons or electrons) is small enough, statistical fluctuations become detectable in a measurement. In addition, the sensor causes two sources of noise: lecture noise (susceptibility of the sensor; negligible in comparison with other noises) and thermal noise (due to the increase of sensor's temperature with the exposure-time). Other sources of noise could be due to dust on sample, air fluctuation, biological markers, etc. Here we are interested in only denoising the acquisition noise, i.e. the shot and thermal noises. However, if the other noises can be modelled by similar statistics as acquisition noises, this plug-in can still be useful.

A standard noise model of the acquisition noise is the Poisson/Gaussian model. It is a mixture of a Poisson process and a Gaussian process describing the shot and the thermal noises:

$$u = g \times Z + \epsilon \ , \tag{1}$$

where u is the measured intensity, g is the gain, $Z \sim \mathcal{P}(I)$ is the Poisson component with I the intensity without noise and $\epsilon \sim \mathcal{N}(\mu, \sigma^2)$ is the Gaussian component.

According to central limit theorem, when the number of photon reaching the sensor is high, the Poisson/Gaussian model can be well approximated by a Gaussian model. Indeed, for large numbers, the Poisson distribution approaches a normal distribution. Then, the noise equation model becomes:

$$u = g \times I + \varepsilon \ , \tag{2}$$

where u is the measured intensity, g is the gain, I is the intensity without noise and $\varepsilon \sim \mathcal{N}(\mu, \sigma^2)$ is the noise component.

3 Denoising images: seeking for redundancy

Since the acquisition noise can be described by a stochastic process, acquiring multiple times an image is efficient for denoising. Indeed, only the noise component is different in each image (random process) and therefore the noised intensities can be averaged in order to estimate the denoised intensities. This method is working because advantage is taken from redundant information in the images, i.e. regions that are similar except for the noise component.

Sometimes, the accumulation of multiple images is not feasible (e.g. acquisition of a living sample time-lapse). Instead, the NLM denoising principle is to seek for redundant information in only one image and use it to denoise this image (Buades et al. 2005). Indeed, there is a high degree of redundancy in natural images (here natural means real and not simulated by a random process): for every small patch in a natural image, it is possible to find many similar patches in the same image (stationary assumption).

Formally, the NLM denoising method is defined as the following. Let $\Omega \subset \mathbb{N}^N$ be an image domain and $I : \Omega \to \mathbb{R}$ the image to denoise. Then, let $\omega_i(p)$ be a similarity function between patches centred on pixels $i, p \in \Omega$ and defined as

$$\omega_i(p) = \frac{K_h(\mathcal{V}_i - \mathcal{V}_p)}{\sum_{j \in \Omega} K_h(\mathcal{V}_j - \mathcal{V}_p)} \ , \tag{3}$$

where K_h is a kernel with a bandwidth h and $\mathcal{V}_i$ is the patch (or neighbourhood) centred on pixel i. Assuming a Gaussian model of the acquisition noise, the NLM denoising method is to look for similar patches in an image and weight them depending on their similarity, and then to estimate the intensity value of the central pixel as the weighted sum of other pixels:

$$NLMD(p) = \sum_{i \in \Omega} \omega_i(p) u(i) \ . \tag{4}$$

In the Gaussian model, the variance ε^2 is an unknown parameter and represent the noise level. It can be set by the user or it can be estimated automatically by using the pseudo-residuals method (Gasser et al. 1986).

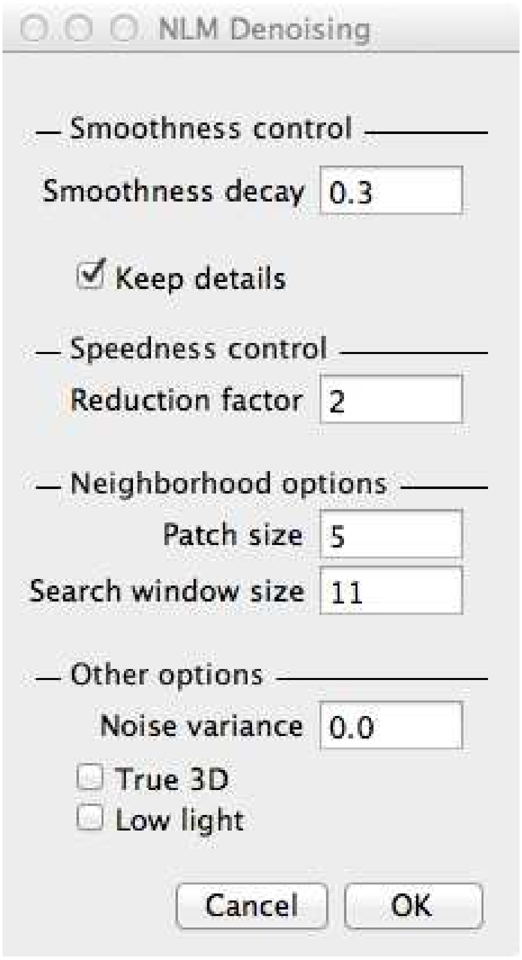
The stationary assumption[1] is not true everywhere, as each image may contain exceptional and non repeated structures. These structures may be smoothed by the NLM algorithm. To prevent this behaviour, the local noise variance can be estimated and compared to the global noise variance. If the local variance is larger than the global one, it means that the current part of the image reflects a structure and should not be smoothed (Buades et al. 2005).

When the image is acquired in low light condition, the Gaussian assumption does not hold any more. Therefore, a pre-step should executed to stabilise the variance, i.e. to change the statistics of the image from Poisson/Gaussian to Gaussian model. Despite it is not straightforward, it can be done using the Anscombe transform (Murtagh et al. 1995).

4 Graphical user interface

Once installed, the plug-in can be found in menu Plugins > Restoration > NLM Denoising. As shown in figure 1, the graphical user interface (GUI) of the plug-in is subdivided into four parts to control: the smoothness, the speediness, the neighbourhood parameters and the other options of the algorithm.

[1] i.e. as the size of image grows, we are able to find in image many similar patches for all of the details of the image


NLM Denoising

Smoothness control

Smoothness decay: 0.3

[✓] Keep details

Speedness control

Reduction factor: 2

Neighborhood options

Patch size: 5

Search window size: 11

Other options

Noise variance: 0.0

[ ] True 3D

[ ] Low light

Figure 1: Graphical user interface (GUI) of the plug-in.

Smoothness This part is used to control the smoothness of the denoising algorithm and contains two parameters:

- the *decay* parameter allows the user to have a control on the smoothness of the denoising (default is 0.3 and range is $[0, 1]$);
- the "*keep details*" option is used to force the algorithm to preserve small details that are smoothed otherwise (default is activated).

Speediness The *reduction factor* allows the user to control the speediness: it increases the speed of the algorithm as the reduction factor increases (default is 2 and range is $[1, \infty[$).

Neighbourhood This part is used to control the neighbourhood options of the algorithm and contains two parameters:

- the *patch size* parameter controls the size of patches used to match similar regions in image (default is 5 pixels);
- the *window size* parameter controls the size of the search window used to find similar regions in image (default is 11 pixels).

These parameters are optimized for best results and should not be changed, unless you know what you are doing.

Other options The last part contains three miscellaneous options or parameters:

- the *noise variance* that can be set by the user or automatically estimated when 0 (default is 0);
- the "*true 3D*" option can be used to consider image as real 3D image and not a 2D image or a stack of slices; non-recommended for 3D images with non-isotropic voxels (default is deactivated);

- the Poisson/Gaussian model can be used when activating the *low light condition* option; a variance stabilisation step is then performed before the denoising (default is deactivated).

5 Implementation details

The main algorithm is implemented using a multi-scale block-wise scheme (see Buades et al. (2005) for more details). The underlying idea is to consider overlapping blocks of pixels instead of pixels alone. The size of these blocks is controlled by the speediness parameter (see section 4). Within this optimisation, the speed of the algorithm is decreased by approximately s^2, where s is the speediness parameter set by user. Setting this parameter to 1 will run the naïve implementation (pixel-by-pixel).

The noise variance is automatically estimated using the pseudo-residuals method (Gasser et al. 1986). For the low light conditions option, the variance is stabilised in the input image using the general Anscombe transform and the parameters of this transform are estimated using a quad-tree algorithm (see Boulanger et al. (2010) for more details).

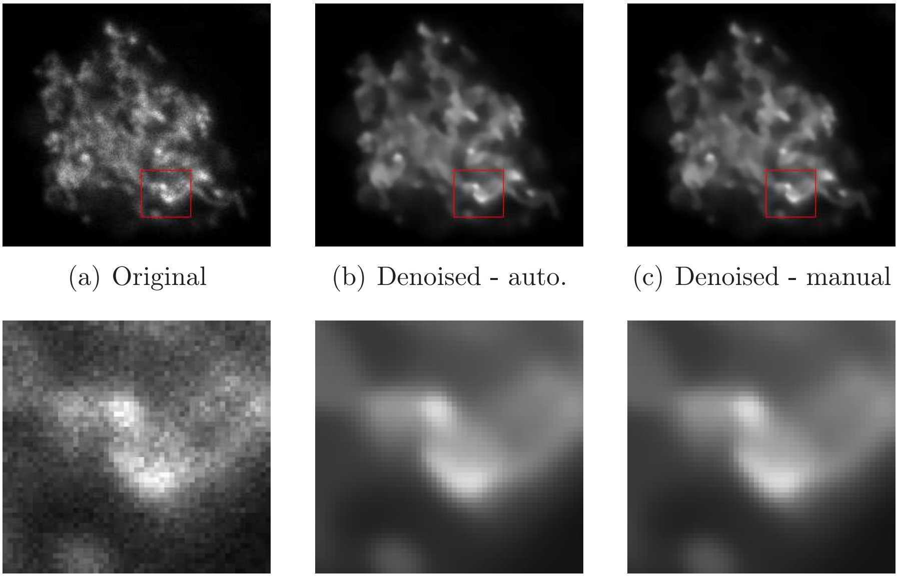
The plug-in is implemented in Java with the plug-in interface of ImageJ and its source code is available on demand. The plug-in is working fine with the version 1.49r of ImageJ and Fiji.

6 Use cases

6.1 Example 1: classical denoising

Within this first example, the denoising process and the use of the noise variance option are demonstrated (see the results in figure 2). By leaving the noise variance to 0, it will be estimated automatically by the plug-in.

If the results of the automatic estimation are not satisfactory, you can set manually this option by a simple procedure. First,



(a) Original (b) Denoised - auto. (c) Denoised - manual

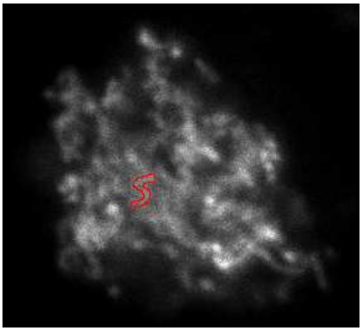
(d) Zoom original (e) Zoom denoised - auto. (f) Zoom denoised - manual

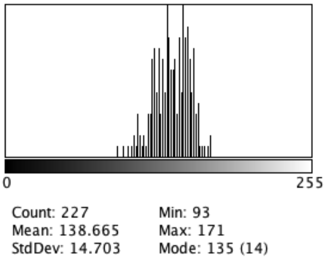
Figure 2: Classical example of image denoising with both automatic and manual setting for the noise variance.

find a homogeneous area (i.e. without any structure variation; see in figure 3(a)). Then, compute the histogram of this area (use the histogram command of ImageJ; see figure 3(b)). If the chosen area is appropriate, the histogram should look like a Gaussian distribution (assuming a Gaussian model for the noise component). The noise variance is then indicated by the statistics of this histogram (the variance is the square of the standard deviation).

6.2 Example 2: keeping image details

A useful functionality of the plug-in is the "keeping details" option. By using it, you can denoise an image while preserving parts of the image that can be statistically considered as details of the acquired structure and not as noise.


(a) Original



```
Count: 227        Min: 93
Mean: 138.665     Max: 171
StdDev: 14.703    Mode: 135 (14)
```

(b) Histogram

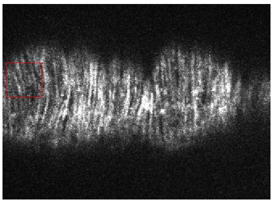
Figure 3: Histogram of the homogeneous area delimited by the red ROI. This histogram is used to estimate the noise variance (indicated on the bottom right).

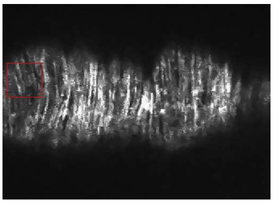
In figure 4, the denoising is shown with and without using this option. While using it, you can clearly see a gain on this image where small structures (e.g. actin filaments) should not be considered as a normal statistical fluctuation of the area (noise).

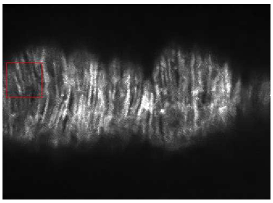
6.3 Example 3: low light conditions

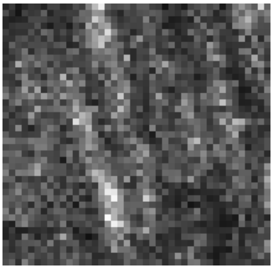
Confocal images are often acquired in low light conditions, i.e. the amount of photons emitted from the sample is small (e.g. if you need to put the microscope in the photon counting mode). In these particular conditions, the Gaussian approximation of the noise (see equation (2)) is not valid any more. Therefore, before applying the denoising algorithm described in section 3, a variance stabilisation is required first. This is achieved using a dedicated algorithm (see section 5). Then, the statistics of the noise are "converted" from Poisson/Gaussian to Gaussian.

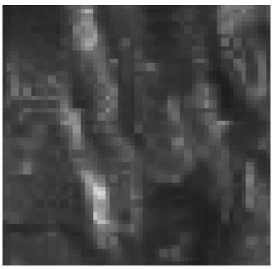
By using the low light conditions option, a variance stabilisation is applied prior to image denoising. In figure 5, the denoising is shown with and without using this option. You can see a clear gain on 1) overall denoising and 2) denoising accuracy at different intensity level (the Poisson/Gaussian model is a signal-dependent


(a) Original


(b) Denoised


(c) Keeping details


(d) Zoom original


(e) Zoom denoised

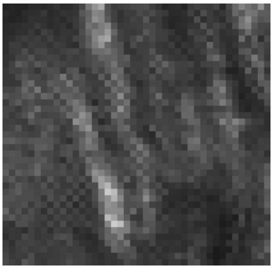
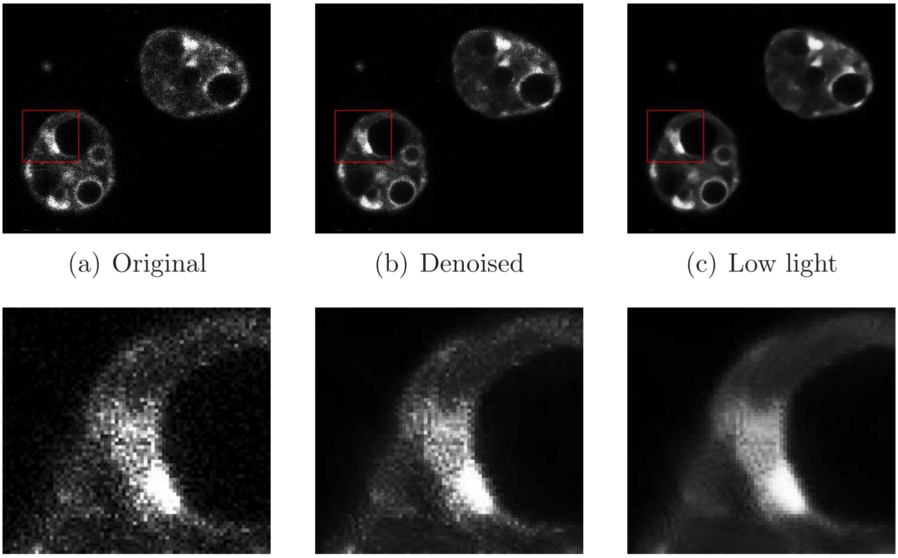

(f) Zoom keeping details

Figure 4: Effect example of the "keeping details" option. By activating this option, the small structures are better preserved on this example.



(a) Original (b) Denoised (c) Low light

(d) Zoom original (e) Zoom denoised (f) Zoom low light

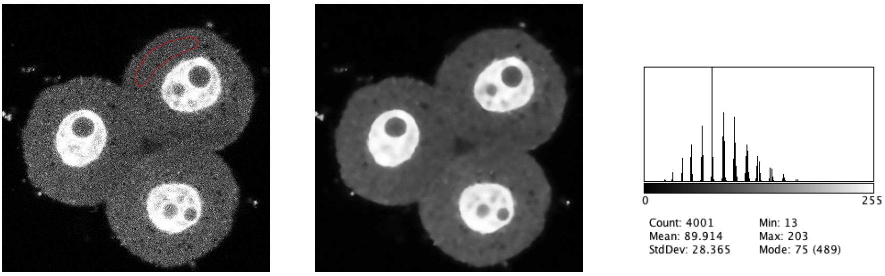
Figure 5: Effect of the low light conditions option. By activating this option, the noise component is correctly modeled and as a consequence, the denoising is more efficient.

model, i.e. the noise variance depends on the signal level).

6.4 Example 4: other source of noise

This plug-in can be used not only to denoise acquisition noise, but also to denoise any noise having Poisson/Gaussian (see equation (1)) or Gaussian (see equation (2)) statistics.

In figure 6, you can see an example of such denoising. The noise in the cytoplasm is due to the experiment itself and not to the acquisition process. Anyway, depending on the noise properties, the plug-in may still be applicable. The first step is to check if the noise component has one of the two statistics managed by the plug-in. This is achieved by drawing a ROI in the cytoplasm (red circle in figure 6(a)), and then computing the histogram (see figure 6(c)). Since the histogram curve shows a Gaussian bell curve



```
Count: 4001      Min: 13
Mean: 89.914     Max: 203
StdDev: 28.365   Mode: 75 (489)
```

(a) Original (b) Denoised (c) Histogram

Figure 6: Denoising example of noise not due to acquisition. Here the noise is due to the experiment itself. The histogram shows the statistics for the part of the image surrounded by a red circle and that should be homogeneous. With this histogram, we are able to see that the statistics look like a Gaussian with a standard deviation of 28.4. Therefore, we can try to remove this noise with the plug-in.

with standard deviation of 28.4, we can use the plug-in to denoise the image (eventually by specifying 28.4^2 as noise variance in the GUI).

7 References

Boulanger, J., C. Kervrann, P. Bouthemy, P. Elbau, J.-B. Sibarita, and J. Salamero (2010). “Patch-Based Nonlocal Functional for Denoising Fluorescence Microscopy Image Sequences”. In: *Medical Imaging, IEEE Transactions on* 29.2, pp. 442–454. ISSN: 0278-0062.

Buades, A., B. Coll, and J. Morel (2005). “A Review of Image Denoising Algorithms, with a New One”. In: *Multiscale Modeling &amp; Simulation* 4.2, pp. 490–530.

Coupé, P., P. Yger, S. Prima, P. Hellier, C. Kervrann, and C. Barillot (2008). “An Optimized Blockwise Nonlocal Means Denoising Filter for (3-D) Magnetic Resonance Images”. In: *(IEEE) Transactions on Medical Imaging* 27.4, pp. 425–441.

Gasser, T., L. Sroka, and C. Steinmetz (1986). “Residual variance and residual pattern in nonlinear regression”. In: *Biometrika* 73.3, pp. 625–633.

Lindenbaum, M., M. Fischer, and A.M. Bruckstein (1994). “On gabor contribution to image enhancement”. In: *Pattern Recognition* 27, pp. 1–8.

Murtagh, F., J. Starck, and A. Bijaoui (1995). “Image restoration with noise suppression using a multiresolution support”. In: *Astron. Astrophys.* 112, pp. 197–189.

Ordentlich, E., G. Seroussi, S. Weinberger M. Verdu, and T. Weissman (2003). “A discrete universal denoiser and its application to binary images”. In: *IEEE International Conference on Image Processing (ICIP)*. Vol. 1, pp. 117–120.

Yaroslavsky, L.P. (1985). *Digital Picture Processing - An Introduction.* Ed. by L.P. Yaroslavsky. Vol. 9. Springer Series in Information Sciences. Springer Verlag.