UNIVERSITY OF BOLOGNA
SCUOLA di Architettura ed Ingegneria

Two years master in

Biomedical Engineering

Marcello Sicbaldi
927417

Project for Numerical Analysis PART B
A.A 2020/21

Project PDE N. 29

Nonlinear Perona Malik diffusion for image denoising

CONTENTS

PROJECT INTRODUCTION – IMAGE DENOISING

Due to inherent physical limitations of various recording devices, images become prone to manifestation of some random noise during image acquisition. Noise can be understood as a basic signal distortion which affects the process of image analysis and information extraction.
Image denoising aims at removing noise from a noisy image in order to restore the true image: overall, recovering meaningful information from noisy images to obtain high quality images is an important and challenging task.
Mathematically, the problem of image denoising can be modeled as follows (inverse ill-posed problem):
$y = x + n$
where y is the observed noisy image, x is the unknown clean image and n represent additive Gaussian noise. The goal of noise removal is to decrease noise in natural images while minimizing the loss of true features. However, since noise, edges, and texture are high frequency components, it is difficult to distinguish them in the process of denoising and the denoised images could inevitably lose some details. Therefore, the main challenges for image restoration are:

- Flat areas should be smooth
- Edges should be preserved and not blurred
- New artifacts should not be generated.

One of the most widespread techniques for image denoising are spatial filters, which can be further classified into two types: linear filters and non-linear filters. Straightforward extension of 1D signal theory dictates the application of linear filters to consist in a 2D convolution between the input image and the impulse response function (point spread function or kernel) of the filter. The simplest and fastest linear filter is the mean filter, which simply replaces each pixel intensity with the average over a given neighborhood. Gaussian filter (GF), on the other hand, gives more weight to the pixels closer to the given one, therefore being more effective, since closer pixels are more likely to be part of the same surface. Its only parameter is the standard deviation σ (higher σ means more smoothing).

Well, what PDEs have to do with all this?
Let's consider the diffusion equation: $\partial_t u = div\,(D \cdot \nabla u)$.
Diffusion can be thought as a physical process that equilibrates concentration differences without creating or destroying mass; in image processing we may identify the concentration with the grey value at a certain location. If the diffusion tensor D is a constant, the diffusion equation reduces to $\partial_t u = \Delta u$, which is the linear diffusion equation or heat equation. By analyzing the solution of this equation, we can state that linear diffusion is equivalent to Gaussian filtering with $\sigma = \sqrt{2t}$.
Nevertheless, this type of filtering smoothes indiscriminately the image a time passes, blurring edges: instead, we would like to diffuse in flat regions and stop the diffusion at the boundaries. Keeping that in mind, Perona and Malik modified the linear diffusion equation by choosing D as a function of the magnitude of the gradient of u, as we will analyze in the next section.

DESCRIPTION OF THE MATHEMATICAL MODEL

Perona and Malik proposed a nonlinear diffusion method for avoiding the blurring and localization problems of linear diffusion filtering. They apply an inhomogeneous process that reduces the diffusivity at those locations which have a larger likelihood to be edges. This likelihood is measured by $|\nabla u|^2$. The Perona-Malik filter is based on the equation

$$\partial_t u \; = \; div \; (g(|\nabla u|^2) \; \nabla u)$$

and it uses the following diffusivity function

$$g(s^2) = \frac{1}{1+\frac{s^2}{K}} \quad , \quad s = \; |\nabla u|.$$

The value of *g* is inversely proportional to the magnitude of the gradient. Since the magnitude of gradient is weak within uniform regions the diffusion coefficient is almost 1, the PM model acts as linear diffusion, smoothing the inner region and removing noise. Near the boundaries the magnitude of the gradient is low, thereby the diffusion coefficient is almost zero, so the diffusion is stopped across boundaries and it preserves edges.

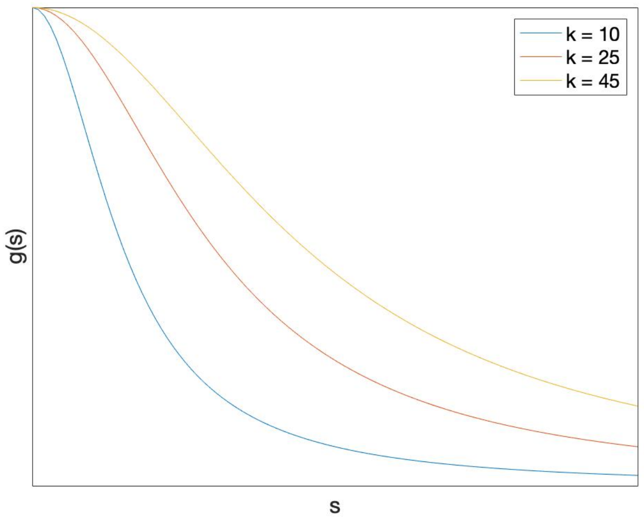

Figure 1: diffusion function varying K

The constant *K* controls the change in curvature of *g,* and it can be seen as a threshold parameter that decides the amount of diffusion to take place: a lower K allows our diffusion function to detect softer edges, as *g* goes faster to 0. the "optimal" value for the contrast parameter λ has to depend on the problem. Several proposals have been made to facilitate such a choice in practice, for instance adapting it to image geometry or to the estimated noise.

Although often referred as *anisotropic*, it should be noted that the only difference with respect to linear diffusion is the employment of a scalar valued function instead of a constant diffusivity: since

it doesn't make use of the concept of gradient direction, it is an *isotropic* model with simply inhibited diffusion at the edges.
For the sake of this project only isotropic nonlinear diffusion is considered, analyzing a different numerical scheme for its implementation with respect to the one discussed by Perona and Malik in their original paper. In particular, let's consider the divergence of the product between a scalar valued function *f* and a vector field *v*.
We can write:

$$\begin{aligned} \nabla \cdot (fv) &= \sum(\partial_i f v_i) \\ &= \sum(f\partial_i v_i + v_i \partial_i f) \\ &= f\sum\partial_i v_i + \sum v_i \partial_i f \\ &= f\nabla \cdot v + v \cdot \nabla f \end{aligned}$$

We can apply this sort of product rule of the divergence to the original Perona-Malik equation, which becomes:

$$u_t = g\Delta u + \nabla g \cdot \nabla u$$

One the right-hand side of the equation we have two terms that contributes to the evolution. The first term is a parabolic contribution that models diffusion, while the second term is a hyperbolic contribution, which corresponds to advection along an underlying velocity field ∇*g* whose direction and strength depend on edge position.

The following section is dedicated to the numerical methods used to discretize this equivalent formulation of the Perona Malik diffusion model.

REALIZATION OF THE PROJECT IN MATLAB

Since we are in 2 dimensions, we can write:

$$u_t = g(u_{xx} + u_{yy}) + g_x u_x + g_y u_y$$

The initial condition is the initial image $u(x, y, 0) = I(x, y)$. We use Neumann homogeneous boundary conditions.

The diffusive term is discretized applying the explicit methods of lines (MOL):

1. Discretization of the space domain using the 5-point Laplacian, by keeping in mind that g is an evolving function depending on the image structure and not a constant:

$$g_{i,j}\Delta_5 u_{i,j} = g_{i,j}\left[\frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{h^2} + \frac{u_{i,j+1} - 2u_{i,j} + u_{i,j+1}}{h^2}\right]$$

where *g* is discretized with backward differences.

2. Discretization of the system of ODE-IVPs for the unknowns $u_{i,j}(t)$ with Explicit Euler: $U^{n+1} = U^n + dt * LU^n$, where *dt* is the temporal step (the spatial step equal to 1) and L is the discretization matrix of the Laplacian operator.

The advection term is approximated using an upwind scheme, which is implemented by checking the sign of each component of ∇g (g_x and g_y) and constructing one-sided difference approximation of ∇u in the appropriate (upwind) direction. Being U^n_{x+} and U^n_{x+} the backward and forward differences of U at time step n, we can write:

$$\mathrm{a} = \max(g_x, 0)\, U^n_{x-} + \min(g_x, 0)\, U^n_{x+} + \max\left(g_y, 0\right) U^n_{y-} + \min\left(g_y, 0\right) U^n_{y+}$$

Combining the two discretization, our equation is overall approximated by the following explicit scheme:

$$U^{n+1} = U^n + dt * (LU^n + a)$$

Where U^{n+1} is the image at time step n+1 and U^n is the image at time step n.

For the sake of comparing this implementation with the original one, the parameter K was assigned to a fixed value, 10.

STABILITY

The stability of our explicit algorithm depends only on the choice of the temporal step *dt*: this is because we are dealing with a grid of pixels, so the steps in both spatial coordinates are the same and equal to 1. To analyze stability, we can distinguish between two situations:

- Near the edges of the image $g^{i,j} \approx 0$, so we have only the advection term (figure 2). To satisfy the CLF conditions we would need $\left|\nabla g^{i,j}\right| * dt \leq 1$, so $dt \leq \frac{1}{\left|\nabla g^{i,j}\right|}$. The worst situation would be when $\nabla g^{i,j}$ is maximum, but being $0 \leq g_x^{i,j}, g_y^{i,j} \leq 1$, we simply obtain dt ≤ 1.

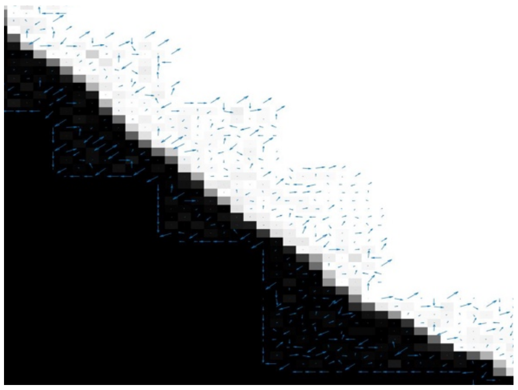

Figure 2. The contribute of ∇g, which is present only near the edges.

- In flat regions we have $g^{i,j} \approx 1$ and $\nabla g^{i,j} \approx 0$. Then our equation reduces to the bi-dimensional heat equation. Its stability is guaranteed for $dt \leq \frac{\min(h_x, h_y)^2}{4}$. In our case this means $dt \leq 0.25$.

Taking the above observations into account, a temporal step $dt = 0.2$ is chosen.

NUMERICAL EXPERIMENTATION

In this section, the different discretization of the Perona Malik's model will be compared with other routines that concern image manipulation according to different types of filters, such as linear diffusion, Perona Malik's original algorithm and Total Variation filter. The effectiveness will be evaluated in terms of error compared to the original image without noise. The evaluation metric employed is the relative error in terms of Frobenius norm.

The images are perturbed by applying additive white Gaussian noise (AWGN) with 0 mean and standard deviation equal to 0.3.

In the comparison with PM original algorithm, we expect that the two different implementations will reach the same result at some point in time, given that the model is the same. Nevertheless, since the algorithm is different, the time at which the two schemes will reach the same result is different (i.e. different number of iterations to reach the same level of filtering).
In particular, we observe that the implementation based on the diffusive + advective term (PM new) is slower than the original one (PM original), so it requires a greater number of iterations to reach the same error with respect to the true image (figure 1).

The different filtering techniques are tested of two images: the famous standard test image Lena (512x512, figure 4) and the CT-scan of a hand (450x450, figure 6).

For Lena, the original Perona-Malik scheme reaches the minimum error of 0.774 at 80 iterations, while our algorithm reaches the minimum error of 0.767 at 250 iterations. For the hand CT-scan, the error shows a similar trend, with a minimum error of 0.0570 at 55 iterations for PM original and 0.0566 at 155 iterations for PM new. These numbers of iterations are chosen when showing the results in the following pages.

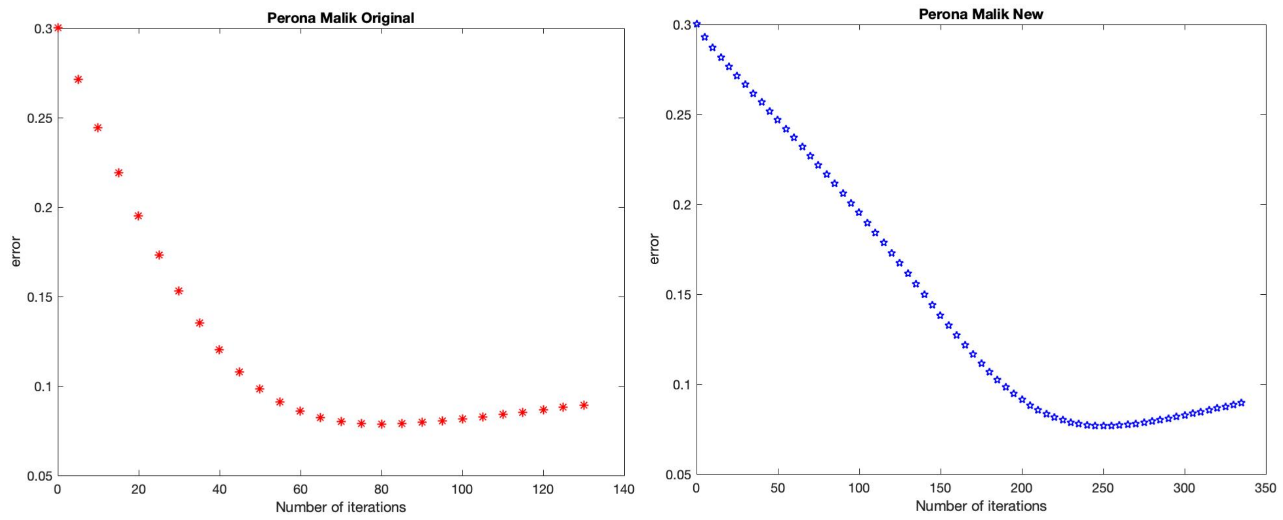

Figure 3: Comparison between the trends of the error with respect to the number of iterations (LENA).

LENA				
	Temporal step	N iterations	K	Error
HEAT EQUATION	0.2	80		0.1239
PM ORIGINAL	0.2	80	10	0.0774
PM NEW	0.2	250	10	0.0767
TOTAL VARIATION	0.03	2500		0.0710

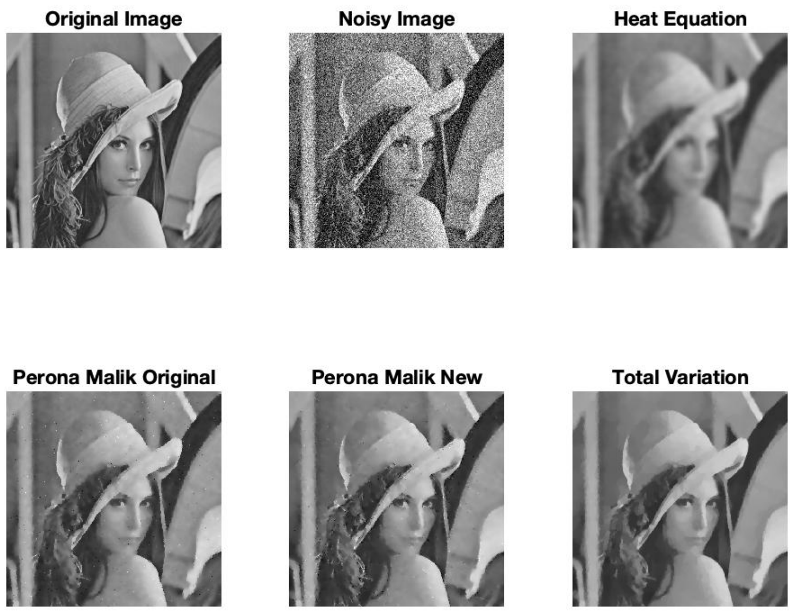

Figure 4: LENA. Comparison between the different filters.

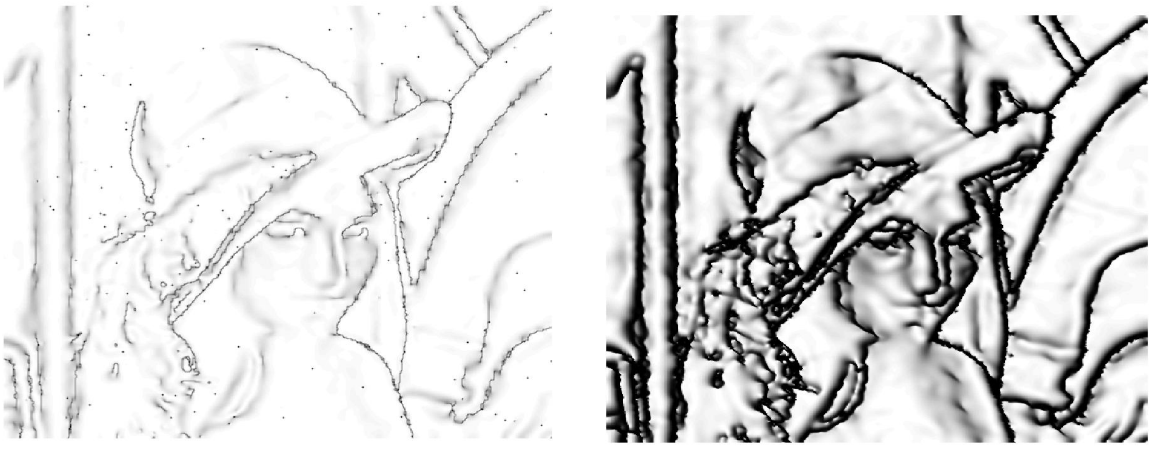
Figure 5. Comparison between the edge maps at the end of the diffusion process. Left: PM original; Right: PM new

HAND CT-scan

	Temporal step	N iterations	K	Error
HEAT EQUATION	0.2	55		0.0838
PM ORIGINAL	0.2	55	10	0.0570
PM NEW	0.2	155	10	0.0566
TOTAL VARIATION	0.03	2500		0.0528

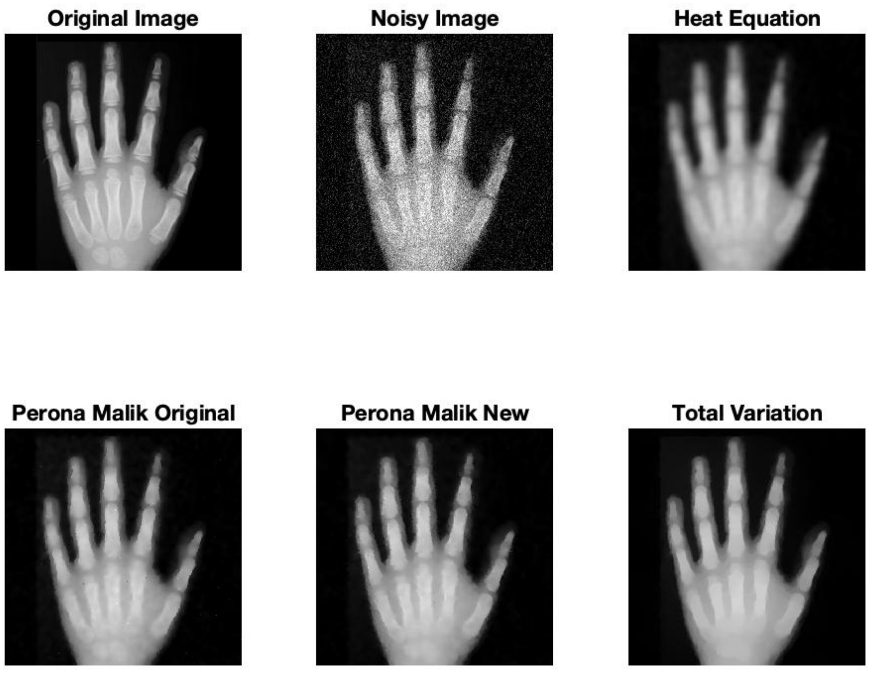

Figure 6: Hand CT-scan. Comparison between the different filters

CONCLUSIVE CONSIDERATIONS

The implementation of the Perona-Malik's model based on the diffusive and advective term is proven to reach the same result of the original implementation, showing the effectiveness of the model in denoising images without blurring edges. The model is clearly superior to linear diffusion filtering, while the total variation model allows to reach a slightly higher level of restoration.
Nevertheless, as mentioned in the second section, the Perona Malik's model is still isotropic: it does not actually differentiate between direction of diffusion and because of this, noise at edges cannot be eliminated successfully by this process. Ideally, we would want to diffuse *along* edges and not *across* (perpendicular to) them. One could then improve the Perona Malik's model by replacing the scalar diffusivity with a diffusion tensor D (2x2 matrix), which depends on the structure tensor of the image, allowing for a more sophisticated description of local image structure. In this way, one could control the direction of diffusion by choosing accordingly the eigenvalues of D: this is done by more sophisticated anisotropic diffusion models such as *edge enhancing diffusion* and *coherence enhancing diffusion*, which would lead to superior results.