



# ANALYSE D'IMAGE

Rapport de TD4

Guénon Marie et Favreau Jean-Dominique  
VIM / Master SSTIM

# Table des matières

---

Diffusion isotrope .....	2
Définition des formules .....	2
Mise en pratique .....	2
Diffusion anisotrope .....	5
Théorie .....	5
1. Démonstration .....	5
2. Dédution .....	5
3. $c(s)$ .....	6
Pratique .....	7
Annexes .....	8
diffusion_isotropique.m .....	8
diffusion_anisotropique.m .....	9

# Diffusion isotrope

## Définition des formules

Nous allons implémenter l'équation de la chaleur et donc résoudre le problème suivant :

$$\begin{cases} \frac{du}{dt}(x, y, t) = \Delta u(x, y, t) \\ u(x, y, t) = g(x, y) \end{cases}$$

En discrétisant  $u(x, y, t)$  avec les pas  $dt$  et  $dx = dy = 1$  et en posant  $u_k = u(x, y, k * dt)$  pour  $k \in \mathbb{N}$ , on obtient :

$$\begin{cases} \frac{u_{k+1} - u_k}{dt} = \Delta u_k = \text{div}(\nabla u_k) \\ u_0 = g \\ 0 \leq dt \leq \frac{1}{4} \end{cases} \quad (1) \text{ c'est-à-dire } u_{k+1} = u_k + dt * \text{div}(\nabla u_k)$$

Avec

$$\nabla_x u(i, j) = \begin{cases} u(i+1, j) - u(i, j) & \text{si } i < N \\ 0 & \text{si } i = N \end{cases} \quad (2)$$

$$\nabla_y u(i, j) = \begin{cases} u(i, j+1) - u(i, j) & \text{si } j < N \\ 0 & \text{si } j = N \end{cases} \quad (3)$$

$$\text{div}(v) = \begin{cases} v_x(i, j) - v_x(i-1, j) & \text{si } 1 < i < N \\ v_x(i, j) & \text{si } i = 1 \\ -v_x(i-1, j) & \text{si } i = N \end{cases} + \begin{cases} v_y(i, j) - v_y(i, j-1) & \text{si } 1 < j < N \\ v_y(i, j) & \text{si } j = 1 \\ -v_y(i, j-1) & \text{si } j = N \end{cases} \quad (4)$$

## Mise en pratique

Les différentes formules vues précédemment sont implémentées de la manière suivante :

```
for k = 1:K

    grad_x = zeros(N,N);
    grad_x(:,1:N-1) = u(:,2:N) - u(:,1:N-1);

    grad_y = zeros(N,N);
    grad_y(1:N-1,:) = u(2:N,:) - u(1:N-1,:);

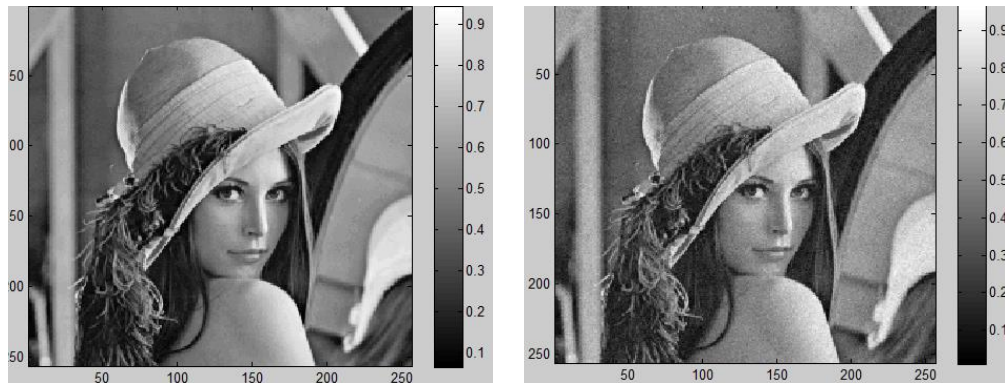
    div_v = zeros(N,N);

    div_v(:,1) = grad_x(:,1);
    div_v(:,2:N-1) = grad_x(:,2:N-1) - grad_x(:,1:N-2);
    div_v(:,N) = - grad_x(:,N-1);

    div_v(1,:) = div_v(1,:) + grad_y(1,:);
    div_v(2:N-1,:) = div_v(2:N-1,:) + grad_y(2:N-1,:) - grad_y(1:N-2,:);
    div_v(N,:) = div_v(N,:) - grad_y(N-1,:);

    u = u + dt * div_v;
end
```



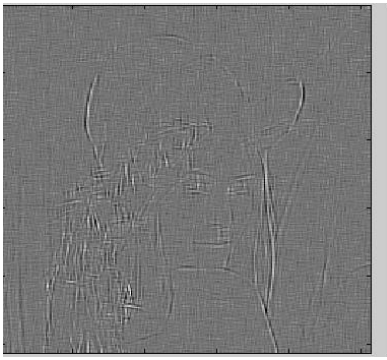



Nous commençons par bruiteur l'image de Lena, et obtenons l'image suivante (originale à gauche) :

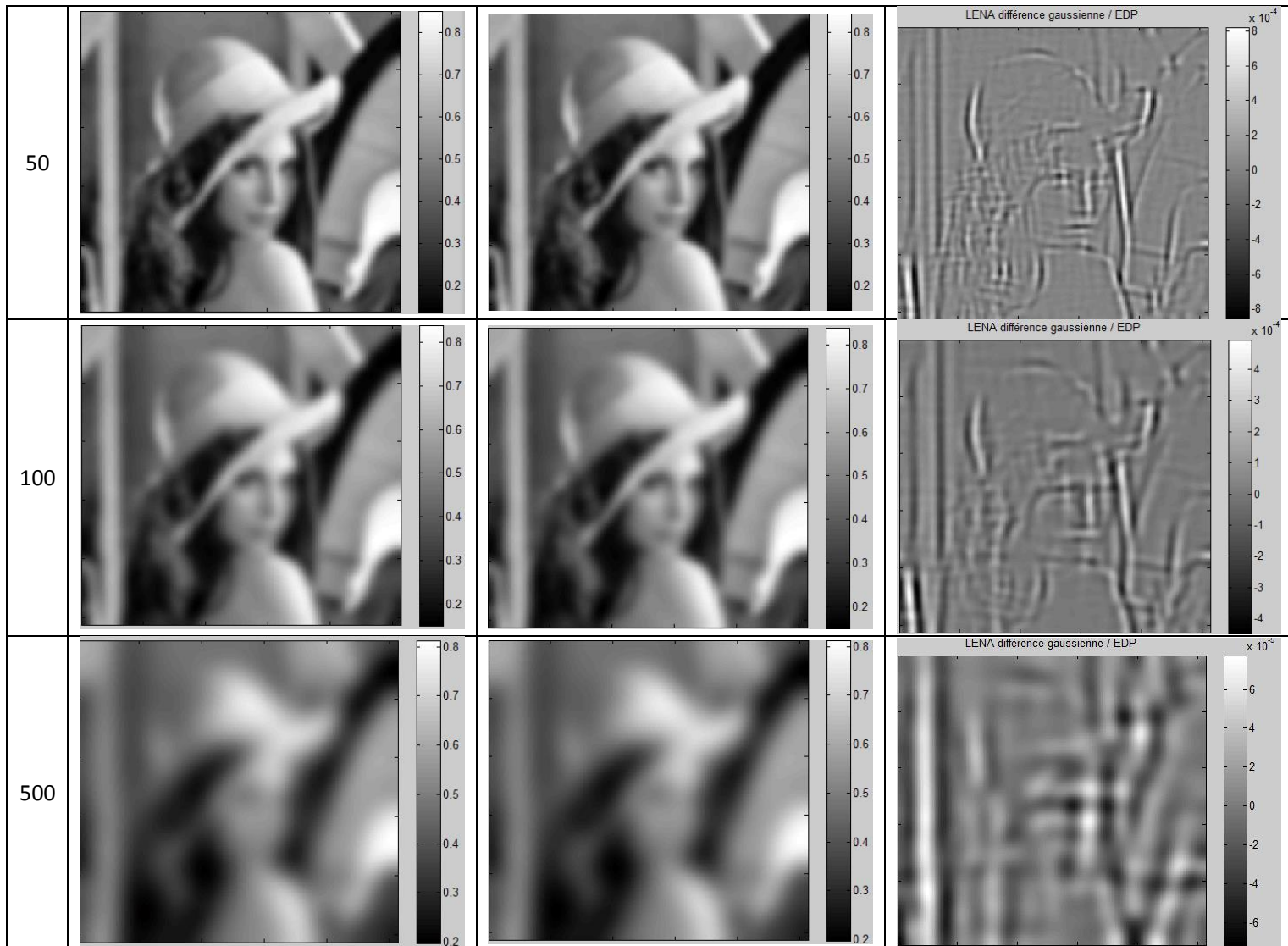


Nous appliquons maintenant les algorithmes vus précédemment sur l'image floutée avec  $dt = 0.1$  et nous comparons les résultats obtenus après un certain nombre d'itérations  $K$  avec la gaussienne  $G_\sigma$  qui est la solution théorique de l'équation de la chaleur et qui suit la formule suivante :

$$G_\sigma = \frac{1}{4\pi t} e^{-\frac{(x^2+y^2)}{4t}}$$

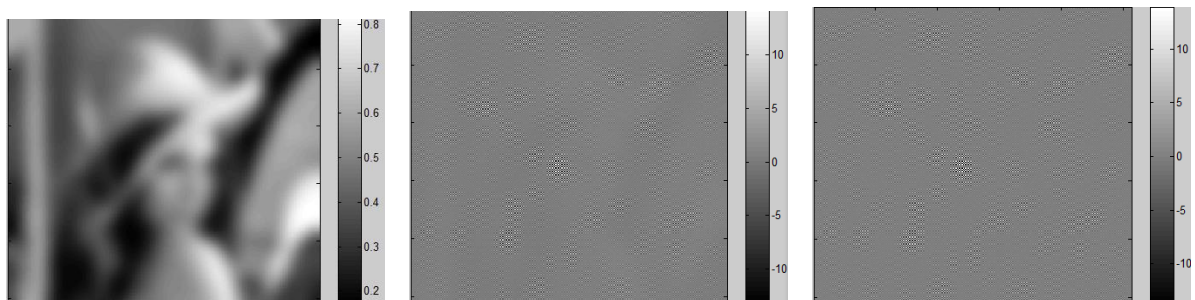
Nous obtenons donc les résultats suivants que nous comparons entre eux :

K	$G_\sigma$	EDP	Norme de la différence
5			
10			



Nous pouvons constater que quand on augmente  $K$ , la distance entre les deux images obtenues diminue nettement. En effet, en regardant seulement l'ordre de grandeur de la norme de la différence, en multipliant par 100 le nombre d'itérations, nous avons multiplié par 1000 la précision de notre image.

Attention cependant au  $\Delta t$  choisit, car il garantit la stabilité la solution de l'EDP. En effet, comme il a été dit plus haut dans les équations, le  $\Delta t$  doit être inférieur à  $\frac{1}{4}$  pour que la solution converge. Si l'on prend un  $\Delta t$  supérieur, notre solution divergera, en effet par exemple pour  $\Delta t=0,26$  et  $K=100$  nous obtenons les résultats suivant (gaussienne à gauche, EDP au milieu et norme de la différence à droite):



Nous pouvons voir tout de suite la divergence de la solution dans l'EDP et donc une distance entre les deux résultats qui est bien plus grande que les distances obtenues jusqu'à présent.

# Diffusion anisotrope

## Théorie

### 1. Démonstration

Notons déjà :  $u_{xx} + u_{yy} = \Delta u = u_{\varepsilon\varepsilon} + u_{\eta\eta}$

Démontrons que  $\operatorname{div}(c(|\nabla u|^2)\nabla u) = u_{\varepsilon\varepsilon}c(|\nabla u|^2) + u_{\eta\eta}[c(|\nabla u|^2) + 2|\nabla u|^2c'(|\nabla u|^2)]$  :

$$\begin{aligned}\operatorname{div}(c(|\nabla u|^2)\nabla u) &= [c(|\nabla u|^2)u_x]_x + [c(|\nabla u|^2)u_y]_y \\&= [c(|\nabla u|^2)]_x u_x + c(|\nabla u|^2)u_{xx} + [c(|\nabla u|^2)]_y u_y + c(|\nabla u|^2)u_{yy} \\&= |\nabla u|_x^2 c'(|\nabla u|^2)u_x + c(|\nabla u|^2)u_{xx} + |\nabla u|_y^2 c'(|\nabla u|^2)u_y + c(|\nabla u|^2)u_{yy} \\&= [u_x^2 + u_y^2]_x c'(|\nabla u|^2)u_x + c(|\nabla u|^2)u_{xx} + [u_x^2 + u_y^2]_y c'(|\nabla u|^2)u_y + c(|\nabla u|^2)u_{yy} \\&= (2u_{xx}u_x + 2u_{xy}u_y)u_x c'(|\nabla u|^2) + c(|\nabla u|^2)u_{xx} \\&\quad + (2u_{xy}u_x + 2u_{yy}u_y)u_y c'(|\nabla u|^2) + c(|\nabla u|^2)u_{yy} \\&= 2[u_{xx}u_x^2 + 2u_{xy}u_y^2 + u_{xy}u_x u_y]c'(|\nabla u|^2) + c(|\nabla u|^2)[u_{xx} + u_{yy}] \\&= 2[u_{xx}u_x^2 + 2u_{xy}u_y^2 + u_{xy}u_x u_y]c'(|\nabla u|^2) + c(|\nabla u|^2)[u_{\varepsilon\varepsilon} + u_{\eta\eta}] \\&= 2u_{\eta\eta}|\nabla u|^2 c'(|\nabla u|^2) + c(|\nabla u|^2)[u_{\varepsilon\varepsilon} + u_{\eta\eta}] \\&= u_{\varepsilon\varepsilon}c(|\nabla u|^2) + u_{\eta\eta}[c(|\nabla u|^2) + 2|\nabla u|^2 c'(|\nabla u|^2)]\end{aligned}$$

### 2. Dédution

Lorsque  $s \rightarrow \infty$  on souhaite diffuser uniquement dans la direction tangentielle  $\varepsilon$  et avoir aucune diffusion dans la direction normale  $\eta$  afin de préserver les contours. C'est-à-dire :

$$c(|\nabla u|^2) + 2|\nabla u|^2 c'(|\nabla u|^2) \xrightarrow{|\nabla u|^2 \rightarrow \infty} 0$$

On pose  $|\nabla u|^2 = s$  :

$$c(s) + 2s * c'(s) \xrightarrow{s \rightarrow \infty} 0 \Leftrightarrow c'(s) + \frac{1}{2s} c(s) \xrightarrow{s \rightarrow \infty} 0$$

Réolvons cette équation différentielle :

$$c'(s) + \frac{1}{2s} c(s) = 0$$

$$c(s) = a * e^{-\frac{1}{2}\ln(s)} = a * \frac{1}{e^{\ln(\sqrt{s})}} = \frac{a}{\sqrt{s}}$$

D'où  $c(s) \approx \frac{a}{\sqrt{s}}$  lorsque  $s \rightarrow \infty$

### 3. $c(s)$

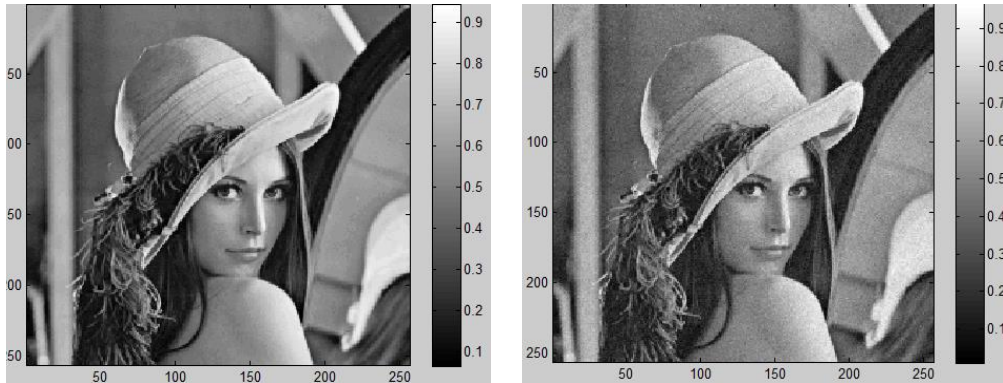
On pose  $c(s) = \frac{1}{\sqrt{1+s}}$

- De manière évidente,  $c$  est décroissante (composée d'une fonction décroissante et de fonctions croissantes sur  $\mathbb{R}^+$ ).
- De manière aisée,  $c(0) = \frac{1}{\sqrt{1+0}} = \frac{1}{\sqrt{1}} = 1$
- De manière évidente et ultra-connue, 1 est négligeable devant  $s$  quand  $s \rightarrow \infty$  alors  $c(s) \approx \frac{1}{\sqrt{s}}$
- $c'(s) = -\frac{1}{2(\sqrt{1+s})^3}$  d'où :  $(\forall s > 0) \ c(s) + 2s * c'(s) = \frac{1}{\sqrt{1+s}} - \frac{s}{(\sqrt{1+s})^3} = \frac{1}{(\sqrt{1+s})^3} > 0$

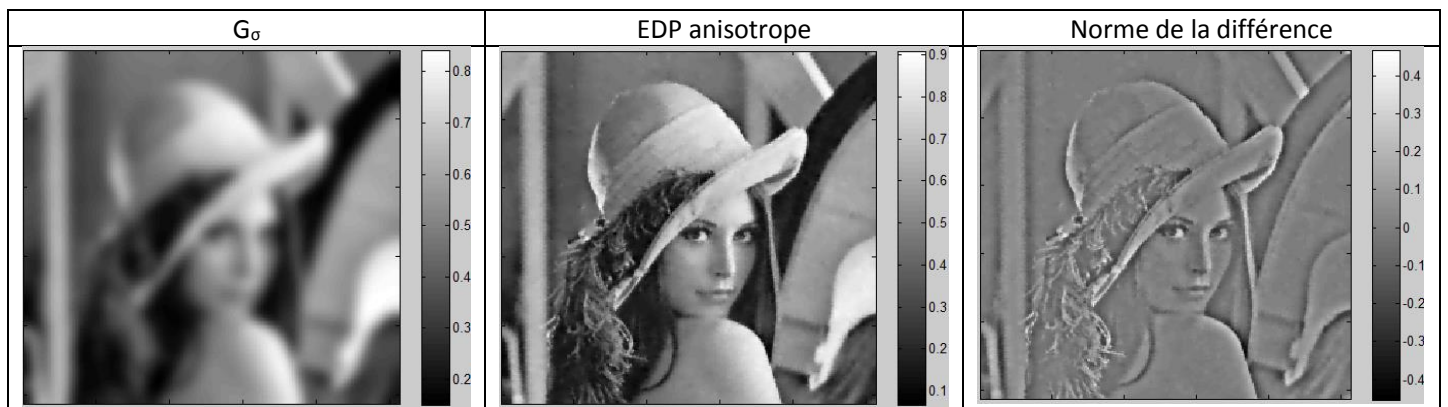


## Pratique

Passons maintenant à la mise en pratique. Comme précédemment, nous commençons par bruiteur l'image de Lena, et obtenons l'image suivante (originale à gauche) :



Puis nous lui appliquons notre algorithme avec  $K = 100$  et  $dt = 0,1$  :



On peut voir ici que le résultat obtenu est nettement plus net que précédemment et nous obtenons une image restaurée de Lena très proche de l'image d'origine, même si l'on peut voir quelques différences au niveau des contours. Ces différences sont mises en valeur sur la Norme de la différence par les valeurs les plus claires et les plus foncées, qui se situent bien au niveau des contours, entre autre au niveau du chapeau et de l'épaule.



# Annexes

---

## diffusion\_isotropique.m

```
function diffusion_isotropique()
    close all;
    img=imread('LENA.BMP');
    N=size(img,1);
    img=double(img(:,:,1));
    sigma_b = 7;
    b=sigma_b * randn(size(img));
    u0 = img + b;
    u=u0;
    K=100;
    dt=.1;

    for k = 1:K

        grad_x = zeros(N,N);
        grad_x(:,1:N-1) = u(:,2:N) - u(:,1:N-1);

        grad_y = zeros(N,N);
        grad_y(1:N-1,:) = u(2:N,:) - u(1:N-1,:);

        div_v = zeros(N,N);

        div_v(:,1) = grad_x(:,1);
        div_v(:,2:N-1) = grad_x(:,2:N-1) - grad_x(:,1:N-2);
        div_v(:,N) = - grad_x(:,N-1);

        div_v(1,:) = div_v(1,:) + grad_y(1,:);
        div_v(2:N-1,:) = div_v(2:N-1,:) + grad_y(2:N-1,:) - grad_y(1:N-
2,:);
        div_v(N,:) = div_v(N,:) - grad_y(N-1,:);

        u = u + dt * div_v;
    end

    std = sqrt(2*K*dt);
    Gsigma = fspecial('gaussian',N-1,std);%/(4*pi*K*dt);
    sol2 = imfilter(u0,Gsigma,'symmetric');
    disp(norm(u-sol2));
    figure;imagesc(img/255);
    title(['LENA origine']);
    colorbar;
    colormap(gray);
    figure;imagesc(u0/255);
    title(['LENA bruitée']);
    colorbar;
    colormap(gray);
    figure;imagesc(u/255);
    title(['LENA solution EDP, k=',num2str(k),', dt=',num2str(dt)]);
    colorbar;
    colormap(gray);
```

```

figure;imagesc(sol2/255);
title(['LENA gaussienne']);
colorbar;
colormap(gray);
figure;imagesc((u-sol2)/255);
title(['LENA différence gaussienne / EDP']);
colorbar;
colormap(gray);
end

```

## diffusion\_anisotropique.m

```

function diffusion_anisotropique()
function cs=c(s)
    %cs=1;
    cs=1./sqrt(1+s);
end
function cs=cprim(s)
    %cs=0;
    cs=-1./(2*(sqrt(1+s)).^3);
end
close all;
img=imread('LENA.BMP');
N=size(img,1);
img=double(img(:,:,1));
sigma_b = 7;
b=sigma_b * randn(size(img));
u0 = img + b;
u=u0;
K=100;
dt=.1;

for k = 1:K

    ux = zeros(N,N);
    ux(:,1:N-1) = u(:,2:N) - u(:,1:N-1);

    uy = zeros(N,N);
    uy(1:N-1,:) = u(2:N,:) - u(1:N-1,:);

    norm_grad2 = ux.*ux + uy.*uy;
    cgrad = c(norm_grad2);
    cprim_grad = cprim(norm_grad2);
    uxx = zeros(N,N);
    uxx(:,1) = ux(:,1);
    uxx(:,2:N-1) = ux(:,2:N-1) - ux(:,1:N-2);
    uxx(:,N) = - ux(:,N-1);
    uyy = zeros(N,N);
    uyy(1,:) = uy(1,:);
    uyy(2:N-1,:) = uy(2:N-1,:) - uy(1:N-2,:);
    uyy(N,:) = - uy(N-1,:);
    uxy = zeros(N,N);
    uxy(:,1) = uy(:,1);
    uxy(:,2:N-1) = uy(:,2:N-1) - uy(:,1:N-2);
    uxy(:,N) = - uy(:,N-1);

```

```

uee=((ux.*ux).*uxx + (uy.*uy).*uyy + 2.*ux.*uy.*uxy);%./norm_grad2;
unn=((ux.*ux).*uyy + (uy.*uy).*uxx - 2.*ux.*uy.*uxy);%./norm_grad2;
for row = 1 : size(uee,1)
    for col = 1 : size(uee,2)
        aa = norm_grad2(row,col);
        if(aa~=0)
            uee(row,col) = uee(row,col)/aa;
            unn(row,col) = unn(row,col)/aa;
        end
    end
end

% disp(norm(uee+unn-uxx-uyy));
div_v = cgrad.*uee+(cgrad+2*norm_grad2.*cprim_grad).*unn;
%div_v = uxx+uyy;
%div_v = uee+unn;
u = u + dt * div_v;
end

std = sqrt(2*K*dt);
Gsigma = fspecial('gaussian',N-1,std);%/(4*pi*K*dt);
sol2 = imfilter(u0,Gsigma,'symmetric');
disp(norm(u-sol2));
figure;imagesc(img/255);
title(['LENA origine']);
colorbar;
colormap(gray);
figure;imagesc(u0/255);
title(['LENA bruitée']);
colorbar;
colormap(gray);
figure;imagesc(u/255);
title(['LENA solution EDP, k=',num2str(k),', dt=',num2str(dt)]);
colorbar;
colormap(gray);
figure;imagesc(sol2/255);
title(['LENA gaussienne']);
colorbar;
colormap(gray);
figure;imagesc((u-sol2)/255);
title(['LENA différence gaussienne / EDP']);
colorbar;
colormap(gray);
end

```