CentraleSupélec

2019 - 2020

Traitement d'images numériques

T.D. 2

Récupérer les images en format binaire Matlab (.mat) sur edunao.

I. Filtrage linéaire : lissage

I.1. Convoluer l'image 'cameraman.tif' par le masque normalisé suivant : $\frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}$

La fonction `conv2` de Matlab permet de produire une image soit de même taille ou de plus grande ou de plus petite taille que l'image de départ. Expliquer pourquoi et prédéterminer le nombre de pixels de ces différentes images. Le vérifier.

`C = conv2(...,shape)` returns a subsection of the two-dimensional convolution, as specified by the `shape`parameter:

`'full'`	Returns the full two-dimensional convolution (default).
`'same'`	Returns the central part of the convolution of the same size as `A`.
`'valid'`	Returns only those parts of the convolution that are computed without the zero-padded edges. Using this option, `size(C) = max([ma-max(0,mb-1),na-max(0,nb-1)],0)`.

Le support d'une conv 2D: [N x N]*[P x P] est [N+P-1 x N+P-1].

L'image filtrée correspond à la partie centrale du produit de convolution de même taille que l'image initiale:[N x N] pixels.

Elle est impactée par des effets de bords sur la demi-longueur de la réponse impulsionnelle ((P-1)/2 pixels) selon chaque dimension. Pour éviter ces effets

de bord on peut limiter le support du produit de convolution à [N-((P-1)/2) x N -((P-1)/2)].

I.2. La moyenne des intensités de l'image est-elle modifiée par cette opération de convolution. Justifier. Quelle est l'influence de la taille du masque (5x5, 7x7) sur l'image filtrée ?

Le masque correspond à la réponse impulsionnelle d'un filtre moyenneur => La moyenne de l'image est inchangée.

$$s[k_1,k_2]=h*e[k_1,k_2]=\frac{1}{(2p+1)^2}\sum_{n_1=-p}^{p}\sum_{n_2=-p}^{p}e[k_1-n_1,k_2-n_2]$$

$$h[k_1,k_2]=\frac{1}{(2p+1)^2},-p\le k_1,k_2\le p$$

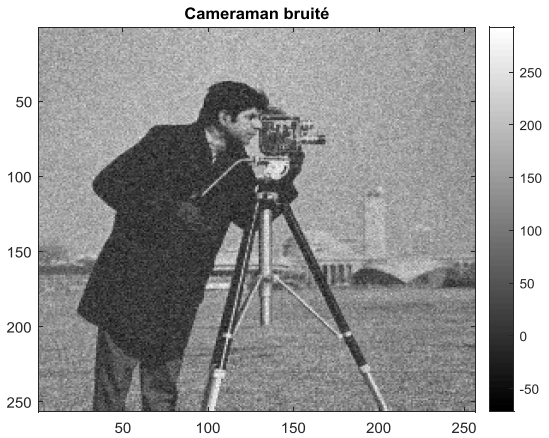
le gain du filtre est : $\sum_{k_1,k_2}h[k_1,k_2]=1$

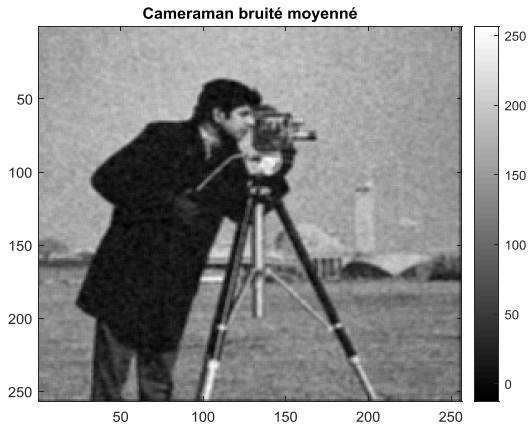
La netteté décroit avec la taille du masque (car la bande passante du filtre moyenneur décroit avec la taille de sa réponse impulsionnelle).

I.3. Bruiter l'image 'cameraman.tif' en utilisant un modèle de bruit blanc additif gaussien d'écart type σ. On utilisera pour cela la fonction `randn` qui génère une matrice aléatoire de loi gaussienne. Prédéterminer l'impact du filtrage par le masque normalisé (3x3) sur l'image bruitée. Le vérifier.

Par moyennage sur 9 pixels la variance du bruit diminue théoriquement (pour une ROI de taille suffisamment grande) d'un facteur 9 (écart type d'un facteur 3).

```
   % Bruitage
sigma=20;
imbbag=im +sigma*randn(size(im));
figure;imagesc(im);title('Cameraman bruité');colormap(gray);colorbar
Roibbag=imbbag(1:100,200:end); % ROI = fond d'image
std(Roibbag(:))=20.9 % écart type

% Débruitage
hm=ones(3,3)/9; % masque de moyennage 3x3 normalisé
imm=filter2(hm,imbbag);
figure;imagesc(imm);title('Cameraman bruité moyenné');colormap(gray);colorbar
Roim=imm(1:100,200:end); % ROI = fond d'image
std(Roim(:))= 12.59    % écart type
```

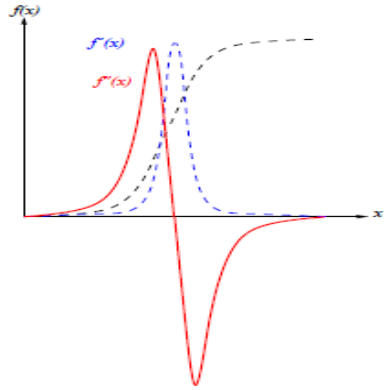




II. Filtrage linéaire : contours et contraste

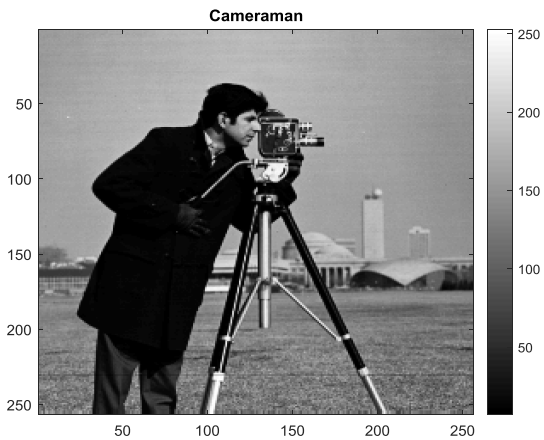
II.1. On souhaite améliorer le contraste de l'image 'cameraman.tif' filtrée par le masque normalisé (3x3) de la question I (image floue). Soit le masque de convolution 2D : $h = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$. Prédéterminer l'effet de l'opération de convolution par h sur les caractéristiques de l'image : moyenne, … Vérifier en traçant l'histogramme.

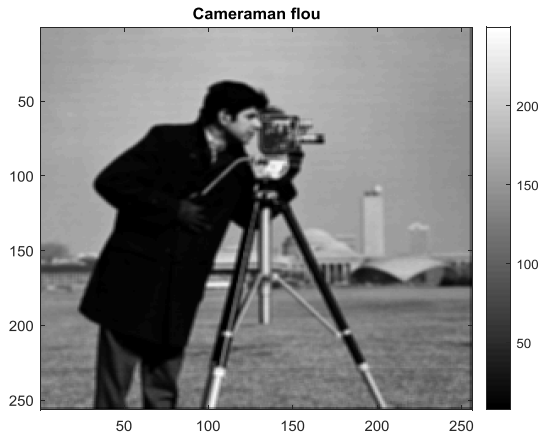
Le masque *h* correspond à la réponse impulsionnelle d'un filtre Laplacien

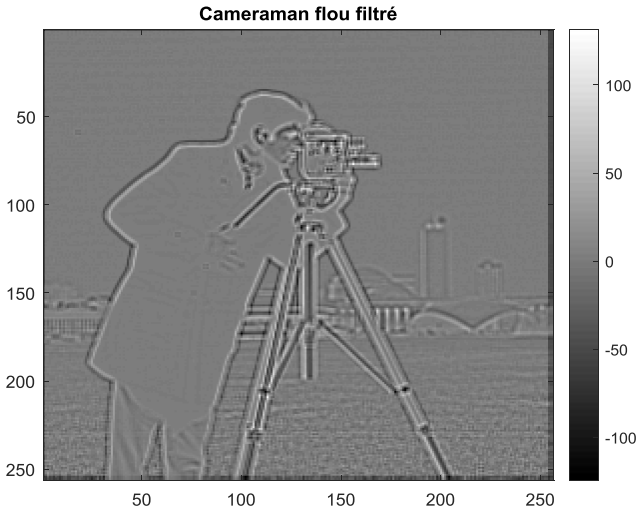


⇨ Le filtre Laplacien est un filtre passe-haut qui élimine la composante fréquentielle spatiale *f*=0. L'image obtenue est de moyenne nulle.
Les régions d'intensité constante dans l'image initiale ont une intensité nulle après filtrage et les régions d'intensité croissante ou décroissante ont une intensité moyenne nulle après filtrage.

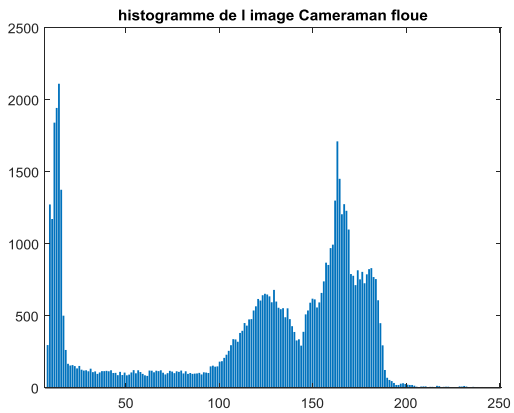
```
% Image floue
clear; close all
im=imread('cameraman.tif'); % image initiale
figure;imagesc(im);title('Cameraman');colormap(gray);colorbar
h=[0 1 0;1 -4 1;0 1 0];
hc=ones(3,3)/9;
im=filter2(hc,im); % Image floue
imf=filter2(h,im); % Image floue filtrée
figure;imagesc(im);title('Cameraman flou');colormap(gray);colorbar
figure;imagesc(imf);title('Cameraman flou filtré');colormap(gray);colorbar
```

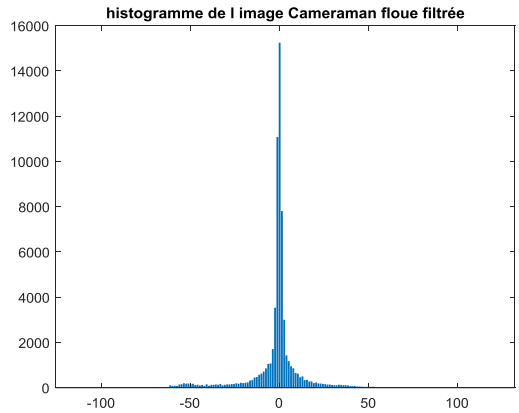






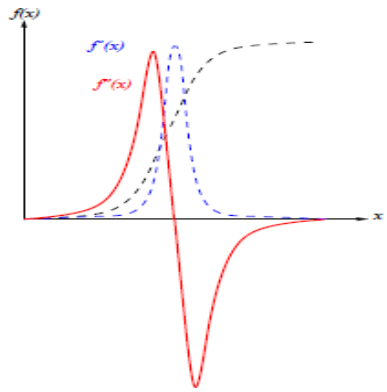

```
%% histogrammes
% image floue
im1D=double(im(:));% Image vectorisee
mean(im1D) % moyenne
[N,x]=hist(im1D,200);
figure
bar(x,N)
title('histogramme de l image Cameraman floue')

% image  floue filtrée
im1D=double(imf(:));
mean(im1D)
[N,x]=hist(im1D,200);
figure
bar(x,N)
title('histogramme de l image Cameraman floue filtrée')
```



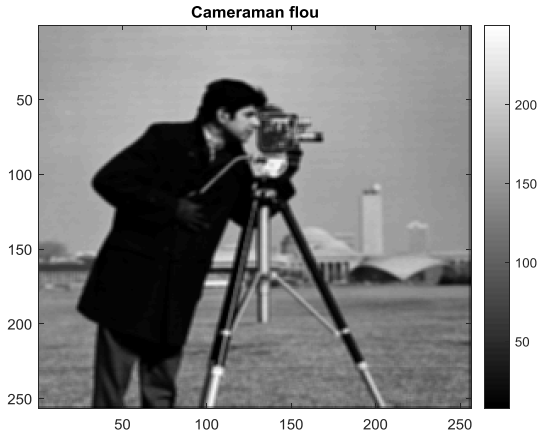


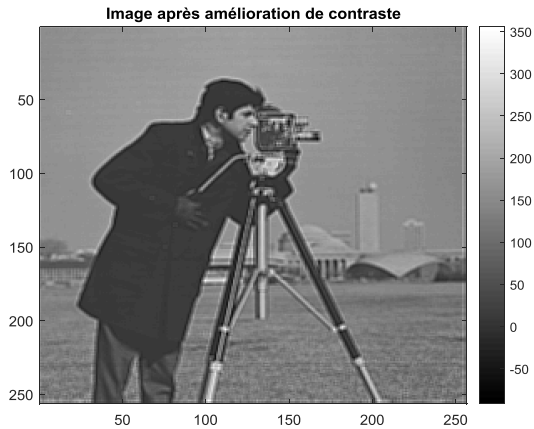
II.2. Comment resserrer les zones de transition de l'image 'cameraman.tif' filtrée par le masque normalisé (3x3) de la question I (image floue) à l'aide du masque *h* ? Le vérifier.

On resserre les transitions en soustrayant à l'image floue une proportion de son laplacien obtenu par filtrage par le filtre de réponse impulsionnelle h.




```
% Amélioration de contraste
imc=double(im)-1*imf; % image floue -laplacien(image floue)
figure;imagesc(imc);title('Image après amélioration de
contraste');colormap(gray);colorbar
```



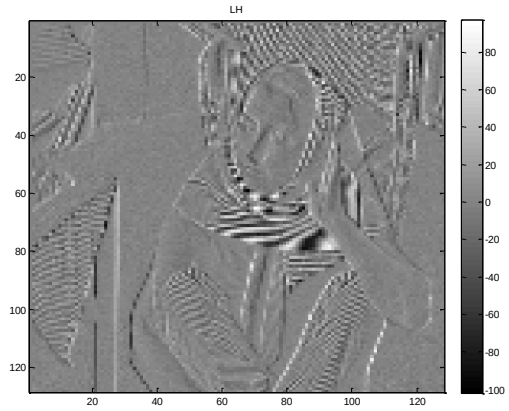


II3.1.Analyser l'image 'barbara.mat' en utilisant les filtres 1D $l=\left[0 \quad \frac{1}{\sqrt{2}} \quad \frac{1}{\sqrt{2}}\right]$ et $h=\left[0 \quad \frac{1}{\sqrt{2}} \quad \frac{-1}{\sqrt{2}}\right]$ suivis d'un sous-échantillonnage d'un facteur 2 (par prélèvement d'un pixel sur deux : y_2=y(1 :2 :end)). On utilisera la fonction `filter` pour mettre en œuvre des filtrages unidimensionnels successifs selon les deux dimensions pour toutes les combinaisons possibles des filtres l et h. Quelles caractéristiques de l'image met-on en évidence par ces filtrages ?

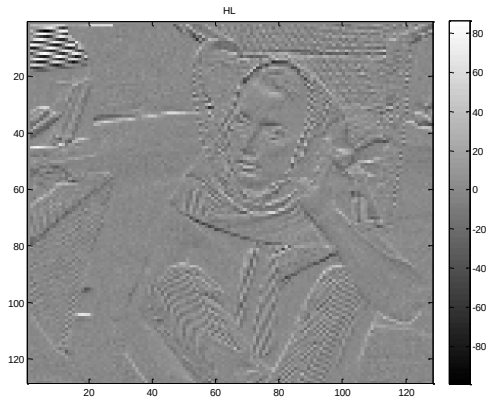
$l = \left[0 \quad \frac{1}{\sqrt{2}} \quad \frac{1}{\sqrt{2}}\right]$ étant la réponse impulsionnelle d'un filtre passe-bas 1D et $h = \left[0 \quad \frac{1}{\sqrt{2}} \quad \frac{-1}{\sqrt{2}}\right]$ celle d'un filtre passe-haut 1D, ces filtrages mettent en évidence les caractéristiques BF et HF selon les directions (ligne et/ou colonne) du filtrage 1D.

```
Clear;close all
load barbara
image=double(X);
taille=size(image);
figure(1) ;imagesc(image) ;title('image') ;colormap(gray) % niveaux  de gris
colorbar

% filtres
a=sqrt(2);
l=[0 1/a 1/a]; %passe-bas
h=[0 1/a -1/a];%passe-haut

% filtrages 1D
imfiltl=filter(l,1,image,[],1); % filtrage passe-bas selon la dim 1
imfilth=filter(h,1,image,[],1); % filtrage passe-haut selon la dim 1
imfiltll=filter(l,1,imfiltl,[],2); % filtrage passe-bas selon la dim 2
imll=imfiltll(1:2:end,1:2:end); % sous échantillonnage d'un facteur 2
imfiltlh=filter(h,1,imfiltl,[],2);
imlh=imfiltlh(1:2:end,1:2:end);
imfilthl=filter(l,1,imfilth,[],2);
imhl=imfilthl(1:2:end,1:2:end);
imfilthh=filter(h,1,imfilth,[],2);
imhh=imfilthh(1:2:end,1:2:end);
figure;imagesc(imll);title('LL');colormap(gray);colorbar
```

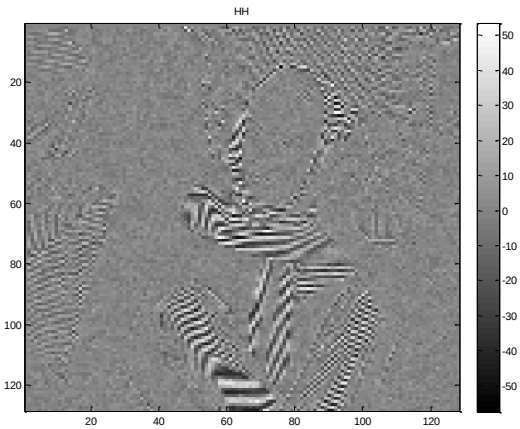

```
Figure;imagesc(imlh);title('LH');colormap(gray);colorbar
```




```
Figure;imagesc(imhl);title('HL');colormap(gray);colorbar
```




```
Figure;imagesc(imhh);title('HH');colormap(gray);colorbar
```

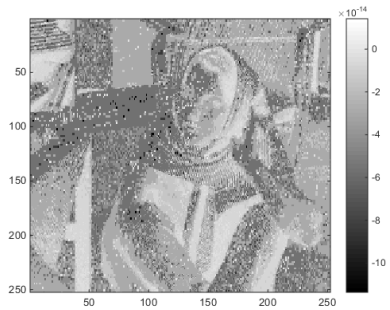


Les coefficients LL, HL, LH, HH correspondent aux coefficients de la décomposition en ondelettes de l'image par l'ondelette de Haar. On remarque que la plupart des coefficients correspondants aux HF : HL, LH, HH sont nuls. Cette propriété est exploitée en compression (cf. standard JPEG 2000).

Pour aller plus loin : reconstruire l'image en sur-échantillonnant d'un facteur 2 et utilisant les filtres $\tilde{l} = \left[\frac{1}{\sqrt{2}} \quad \frac{1}{\sqrt{2}}\, 0\right]$ *et* $\tilde{h} = \left[\frac{-1}{\sqrt{2}} \quad \frac{1}{\sqrt{2}}\, 0\right]$. *Conclure sur l'erreur de reconstruction.*

```
% reconstruction de l'image
```

```
imllrec=zeros(256,256);
imllrec(1:2:end,1:2:end)=imll;
imfiltrecl=filter(fliplr(l),1,imllrec,[],1);
imfiltrecll=filter(fliplr(l),1,imfiltrecl,[],2);
imhhrec=zeros(256,256);
imhhrec(1:2:end,1:2:end)=imhh;
imfiltrech=filter(fliplr(h),1,imhhrec,[],1);
imfiltrechh=filter(fliplr(h),1,imfiltrech,[],2);
imlhrec=zeros(256,256);
imlhrec(1:2:end,1:2:end)=imlh;
imfiltrecl=filter(fliplr(l),1,imlhrec,[],1);
imfiltreclh=filter(fliplr(h),1,imfiltrecl,[],2);
imhlrec=zeros(256,256);
imhlrec(1:2:end,1:2:end)=imhl;
imfiltrech=filter(fliplr(h),1,imhlrec,[],1);
imfiltrechl=filter(fliplr(l),1,imfiltrech,[],2);
imrec=imfiltrecll+imfiltreclh+imfiltrechl+imfiltrechh;
figure
imagesc(imrec);colorbar;colormap(gray)
figure
imagesc(imrec(4:end-1,4:end-1)-image(2:end-3,2:end-3)); title('Erreur de reconstruction')
colorbar;colormap(gray)
Erreur de reconstruction =10^-14
```

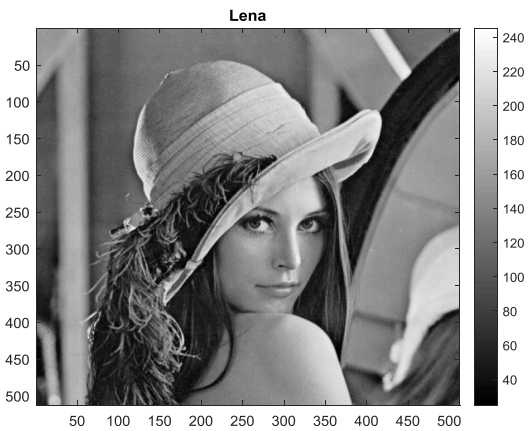


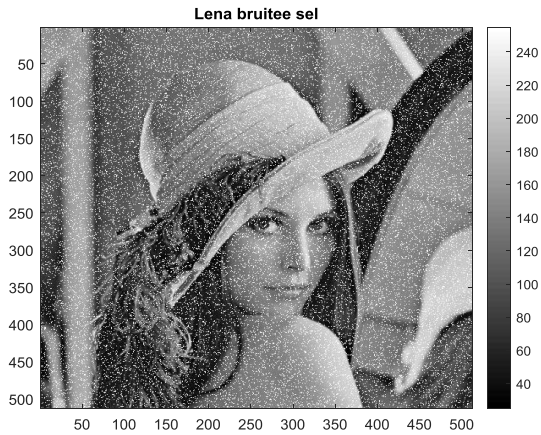
III. Filtrage non-linéaire

III.1. Bruiter l'image 'lena512.mat' en utilisant un modèle de bruit « poivre et sel » caractérisé par la probabilité *p* de remplacement des pixels de l'image. On pourra utiliser la fonction `rand` de Matlab pour générer une variable aléatoire de loi uniforme (ou utiliser la fonction `imnoise` pour bruiter l'image).

```
load lena512.mat
X=lena512;
P=0.1;
figure;imagesc(X); title('Lena');colormap(gray);colorbar
Xps=bruitps(X,p); % bruitage sel
```

```
figure;imagesc(Xps); title('Lena bruitee sel');colormap(gray);colorbar

% définition de la fonction de bruitage sel
function s=bruitps(e,d)
taille=size(e);
s=e;
for i=1:taille(1),
    for j=1:taille(2),
        p(i,j)=rand;
        if p(i,j) < d,
           s(i,j)=255;
        end;
    end;
end;
% for i=1:taille(1),
%    for j=1:taille(2),
%        p(i,j)=rand;
%        if p(i,j) < d,
%           s(i,j)=0;
%        end;
%     end;
% end;
end
```



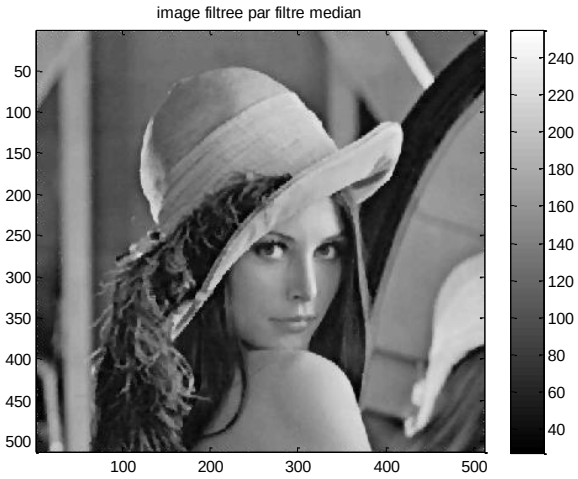


III.2. Proposer et mettre en oeuvre différents filtrages permettant de débruiter l'image.

Le filtrage le plus approprié pour un bruit poivre et sel est un filtrage médian

```
% Filtrage médian sur voisinage [voisin, voisin]
ordre=3;
imfiltmed=med(imbruitps,ordre);

% définition de la fonction med (filtrage médian)
```

```
function s=med(e,voisin)
taille=size(e);
s=e;
Nblim=floor(voisin/2);
for i=1+Nblim:taille(1)-Nblim,
    for j=1+Nblim:taille(2)-Nblim,
       s(i,j)= median(median(e(i-Nblim:i+Nblim,j-Nblim:j+Nblim)));
    end;
end;


figure;imagesc(imfiltmed)%image filtrée
title('image filtree par filtre median');colormap(gray);colorbar
```



III.3. Evaluer à l'aide du *PSNR* les performances des méthodes proposées pour les probabilités $p=10^{-1},10^{-2},10^{-3}$.

Le *PSNR* est défini par le rapport entre le carré de la valeur crête à crête de l'image et la puissance du bruit. Il est utilisé classiquement pour quantifier la « qualité » (en terme de niveau de bruit) d'une image.

$$PSNR = 10\log_{10}\left(\frac{(v_{\max}-v_{\min})^2}{EQM}\right) \quad EQM = \frac{1}{N}\sum_{i,j\in I_i}(I_i(i,j)-I_c(i,j))^2$$

```
   % PSNR
   vcc2=(max(max(X))-min(min(X)))^2 ; % carré de la valeur crete a crete
   EQM=sum(sum((image-imbruitps).^2))/(taille(1)*taille(2)); %puissance du bruit
disp('bruit poivre et sel')
PSNR=10*log10(vcc2/EQM) % PSNR image bruitee p=10^-1
   >  PSNR =13.96 dB % image bruitee


EQM=sum(sum((image-imfiltmed).^2))/(taille(1)*taille(2));
disp('filtrage median')
   PSNR=10*log10(vcc2/EQM) % PSNR image debruitee
   ➔ PSNR =27.51dB % image debruitee
```

EL, CS, 01/2020