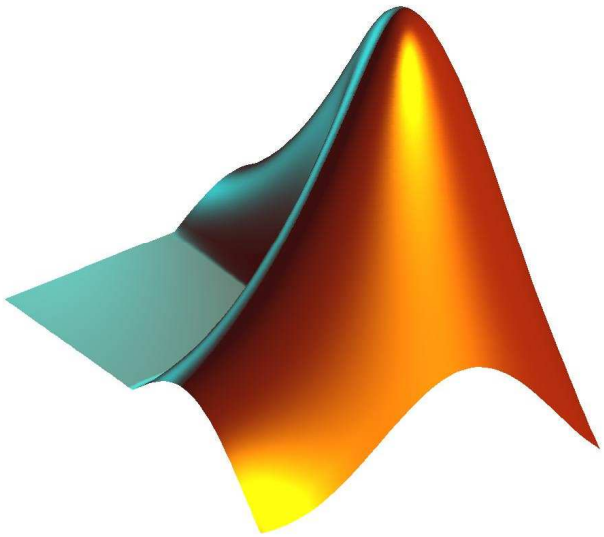
TP Traitement du signal sous Matlab



TP1 : Qu'est-ce que Matlab ?

MATLAB est l'abréviation de MATrix LABoratory. C'est un logiciel pour le calcul numérique et la visualisation optimisé pour le calcul matriciel. MATLAB peut être considéré comme un langage de programmation, il dispose d'une syntaxe spécifique mais elle est simple et intuitive. MATLAB est un interpréteur : les instructions sont interprétées et exécutées ligne par ligne. MATLAB propose deux types de fonctionnement :

- un fonctionnement en ligne de commande où MATLAB exécute les instructions au fur et à mesure qu'elles sont données par l'utilisateur.
- un fonctionnement via des scripts, dans ce cas MATLAB exécute ligne par ligne un fichier texte que l'on désigne par programme.

MATLAB propose également une aide très complète illustrée d'exemple, étant donné le très grand nombre d'instructions utilisables, il est important de bien savoir l'utiliser.

I_ Apprendre les fonctionnalités.

I.1_ L'environnement.

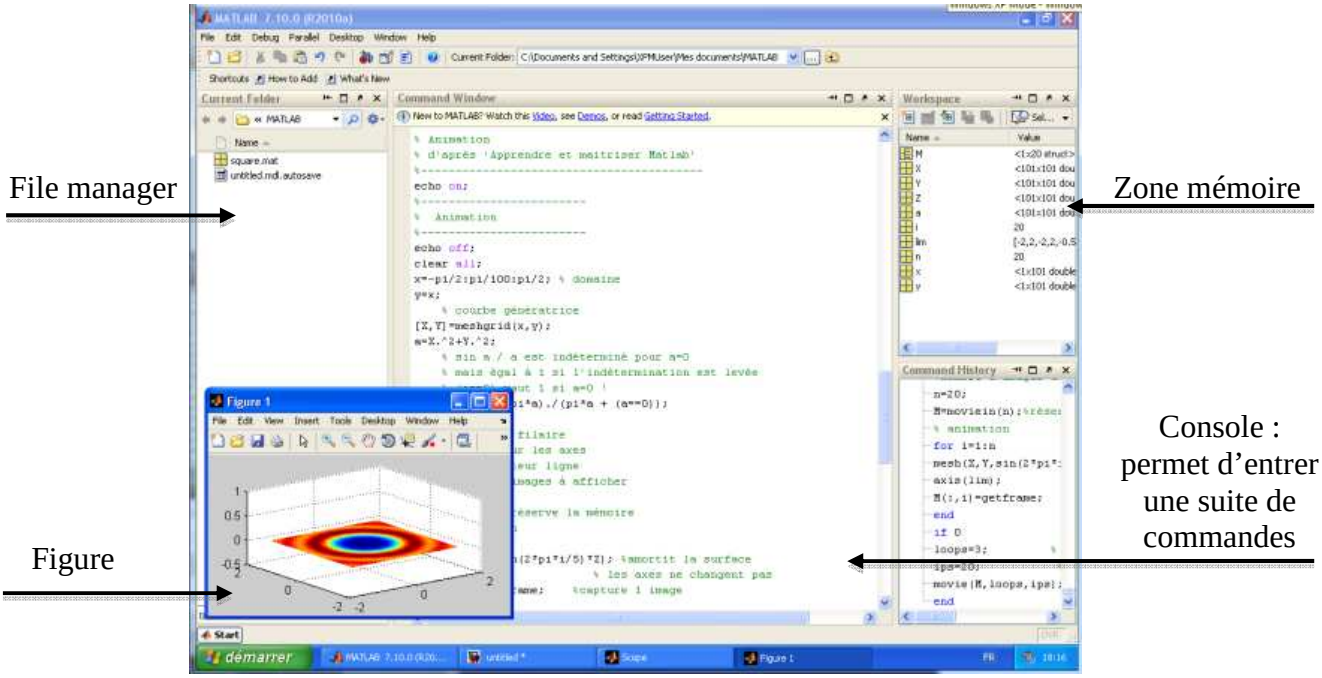


Fig. 1 : Environnement.

I.2_ Variable & Calcul.

Les variables manipulées sont en priorité des matrices à éléments réels ou complexes. Un scalaire est une matrice 1x1. Un exemple de matrice :

```
a=[1,2,3 ;4,5,6]
```

1	2	3
4	5	6

Lorsque les composantes d'un vecteur forment une suite régulièrement espacée, on peut utiliser une boucle dite « implicite » de la forme :

`a=(0 :2 :10)` *ce qui est équivalent à* `a=[0,2,4,6,8,10]`

0	2	4	6	8	10

On accède au premier élément d'une matrice par a(1,1) et à la première ligne de la matrice par a(1, :).

***Remarque** : Il faut respecter les minuscules majuscules dans l'appellation des variables.*

I.2.1_ Résolution de systèmes linéaires.

Soit un four à cuisson lente de 10 kWh stocké à 27° pouvant monter linéairement à une température 1000°C en 24 h. Combien de temps sera nécessaire pour atteindre 371°C ?

```
clear ;
T=[27:1:1000];
Tmax = T(974) ;
tmaxh=24 ;
k=(Tmax)/tmaxh;
Ta=27 ;
Tc=371 ;
deltaT=Tc-Ta ;
t=deltaT;
figure(1); plot (T);
```

Que renvoie la size (T) ?

Les variables ont chacunes un format parmi les suivants :

- unsigned char (1 octet)
- unsigned short (2 octets)
- unsigned char (couleurs)
- signed short (2 octets)
- int (long) (4 octets)
- float (4 octets)
- double (8 octets)
- complex signed byte (2x 1 octet)
- complex signed short (2x 2 octets)
- complex float (2x 4 octets)
- complex double (2x 8 octets)

Aurait-il été possible de stocker T dans un « UnsignedChar » ?

I.3 _ Fonctions Vecteurs et signaux.

Voici Les fonctions mathématiques de base :

- abs, sqrt, real, imag, conj, round, fix, floor, ceil, sign, rem , exp, log.

Les fonctions trigonométriques :

- sin, cos, tan, asin, acos, atan, sinh, cosh, tanh, asinh, acosh, atanh.

Autres fonctions utiles :

- La fonction eig permet d'obtenir les valeurs propres et vecteurs propres d'une matrice.
- La fonction poly donne le polynôme caractéristique associé à la matrice.
- La fonction roots fournit les racines d'un polynôme.

I.3.1_ A l'aide des fonctions ci-dessus, tracer un cercle de diamètre 4.

```
clear
X=[-pi:0.1:+pi];
Y=sin(X);
figure(2);plot (X,Y);
Z=cos(X);
figure(3);plot (X,Z);
figure(4);plot (Y,Z);
```

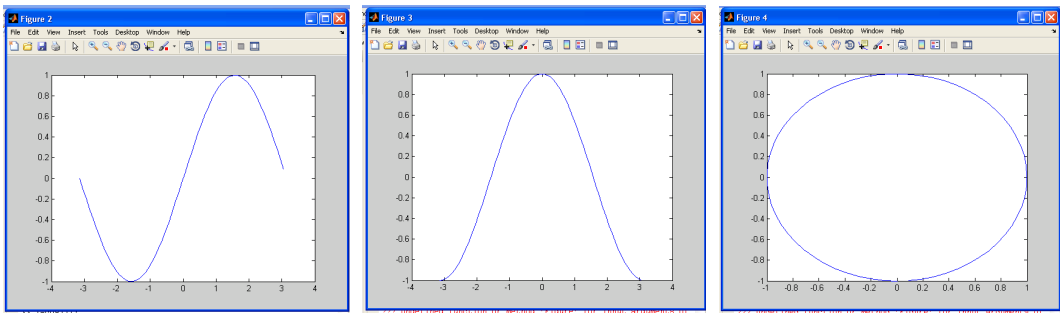


Fig. 2 : Fonctions trigonométriques.

Que ce passe t-il quand la commande suivante est entrée ? Tracez un sinus en retournant ses valeurs négatives autour de Zéro.

```
figure(3);plot (X,Z, 'o');grid
```

I.3.2_ A l'aide des fonctions ci-dessous, tracer une spirale.

```
n = 5;                          % spirale 2D à n tours
a = 0; b = n.*2.*pi;            % bornes de l'intervalle

% intervalle de t : [a;b] découpé en 2000 parties, donc 2001
points

% (x(t);y(t))
t = a:(b-a)./2000:b;

%calcul de la trajectoire (cercle à rayon r variant avec t)

r = t ./ (2*pi);
x = r .* cos(t);
y = r.* sin(t);
plot(x,y,'r','LineWidth',3)   % trajectoire rouge de largeur 3
axis equal                    % axes égaux
set(gcf,'Color','w')          % fond blanc
```

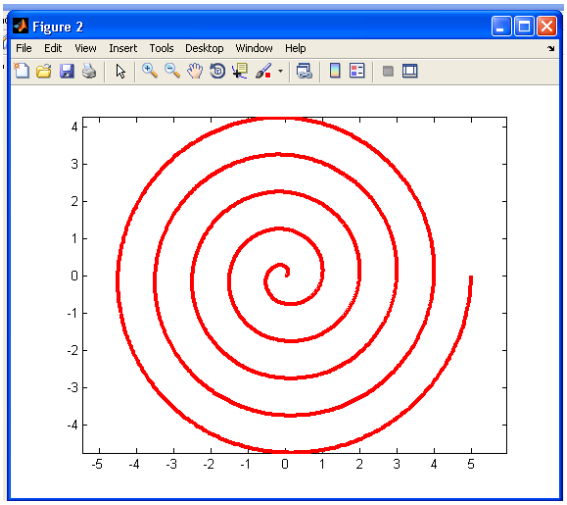


Fig. 3 : Spirale.

Est-il possible de modifier le sens de rotation de la figure 3 ?

I.3.3_Echantillonnage et sous-échantillonnage.

A l'aide du code ci-dessous, tracer un sinus de fréquence croissante dépassant la fréquence d'échantillonnage et montrer la limite acceptable de Nyquiste.

```
%===== SINUS80.M
f0=80;                          % Sinus Freq.
obsdur=0.06;                    % Observation Time
Fs1=200; Fs2=1500;              % Sampling Freq.
n1=round(obsdur*Fs1); n2=round(obsdur*Fs2);
tps1=[0:n1-1]/Fs1; tps2=[0:n2-1]/Fs2;
s1=3*sin(2*pi*f0*tps1); s2=3*sin(2*pi*f0*tps2);
subplot(211); plot(tps1,s1,'x'); grid
subplot(212); plot(tps2,s2,'x'); grid
```

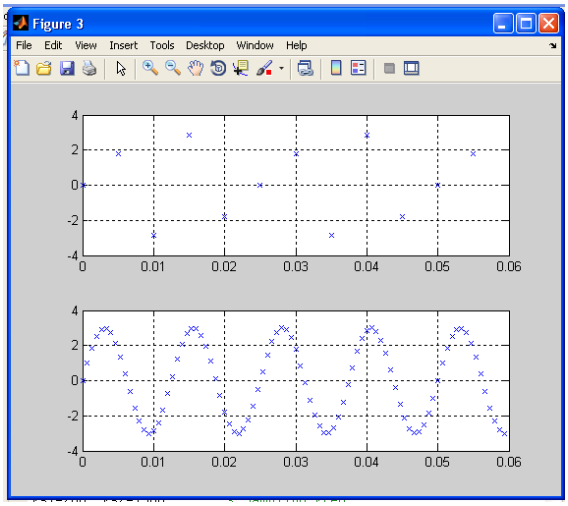


Fig. 4 : Sous-échantillonnage

TP2 : Signal et FFT.

I_ Les signal.

I.1_ Impulsionnel et indiciel.

L'utilisation d'un signal impulsionnel est très courante et à été théorisé par Paul Dirac. Il correspond à une impulsion d'aire unitaire ou la durée est inversement proportionnelle à son niveau. En revanche, un signal indiciel correspond à un simple changement d'état. (Toujours de manière unitaire).

Tracer ces courbes.

I.2_ harmonique.

Puisque Fourier montre que tous signal est composé d'une suite d'harmoniques, retrouver à quel signal ces suites correspondent.

- $$x_{\text{carré}}(t) = \frac{4}{\pi}\sum_{k=1}^{\infty}\frac{\sin\left((2k-1)2\pi ft\right)}{(2k-1)} = \frac{4}{\pi}\left(\sin(2\pi ft)+\frac{1}{3}\sin(6\pi ft)+\frac{1}{5}\sin(10\pi ft)+\cdots\right).$$

- Y=1/2(sin (x)-1/2sin(2x)+1/3sin(3x)-1/5sin(5x))

I.3_ Discrétisation du signal.

Le signal étant échantillonné, il ne comporte qu'une partie de l'information initiale. Le tracé de la suite d'harmoniques est par conséquent discrétisé. A l'aide du code ci-dessous ; tracer les différentes fonctions précédentes en faisant varier le nombre de points.

Traçage d'une période :

```
pts=100 ;          %nombre de point d'une période
fond=1000 ;              % fréquence fondamentale
duree1=1/fond ;          % première durée de période

t  = [0 : duree1/ pts: duree1];

x=2*pi*fond*t ;
f0=sin(x);       h2=sin(2*x);  h3=sin(3*x);
h4=sin(4*x);  h5=sin(5*x);  h6=sin(6*x);
h7=sin(7*x);  h9=sin(9*x);

sinus=     f0 ;                      figure(1); plot(sinus) ;
carre= f0+h3/3+h5/5+h7/7+h9/9;       figure(2); plot(carre);
scie  = 2*(f0-h2/2+h3/3-h4/4+h5/5-h6/6+h7/7); figure(3);
plot(scie);
```

Traçage des spectres pour 100 périodes de signal :

```
fond=100 ;                          % frequence fondamentale
N=44100 ;                           % nb pts sequence
fe=44100 ;                          % frequence d'echantillonnage
t = (1:N)/fe;                       % Axe des temps
x=2*pi*fond*t ;                     % synthese des harmoniques
f0=sin(x); h2=sin(2*x); h3=sin(3*x); h4=sin(4*x); h5=sin(5*x);
h6=sin(6*x); h7=sin(7*x); h9=sin(9*x);

% synthese des fonctions+graph (0<x<441,-1<y<1)
sinus=f0; subplot(2,3,1); plot(sinus); grid; axis([0 441 -1 1]);
title('sinus'); legend('sinus pur') ; xlabel('temps (sec)');
ylabel('amp') ;

carre=f0+h3/3+h5/5+h7/7+h9/9; subplot(2,3,2); plot(carre); grid;
axis([0 441 -1 1]);title('carre'); legend('caré');
xlabel('temps (sec)'); ylabel('amp') ;

scie=1/2*(f0-h2/2+h3/3-h4/4+h5/5-h6/6+h7/7); subplot(2,3,3);
plot(scie); grid; axis([0 441 -1 1]); title('scie'); legend('scie');
xlabel('temps (sec)'); ylabel('amp');

ssi=fft(sinus)/N ; subplot(2,3,4); loglog(abs(ssi));% fft+graph
axis([0 1000 0.0001 10]); title('Spectre du sinus');
legend('sinus pur'); xlabel('freq (Hz)'); ylabel('dB');

sca=fft(carre)/N ; subplot(2,3,5) ; loglog(abs(sca));
axis([0 1000 0.0001 10]); title('Spectre du carre');
legend('carré '); xlabel('freq (Hz)'); ylabel('dB');

ssc=fft(scie)/N; subplot(2,3,6); loglog(abs(ssc));
axis([0 10000 0.0001 10]); title('Spectre du scie');
legend('scie') xlabel('freq (Hz)'); ylabel('dB');
```

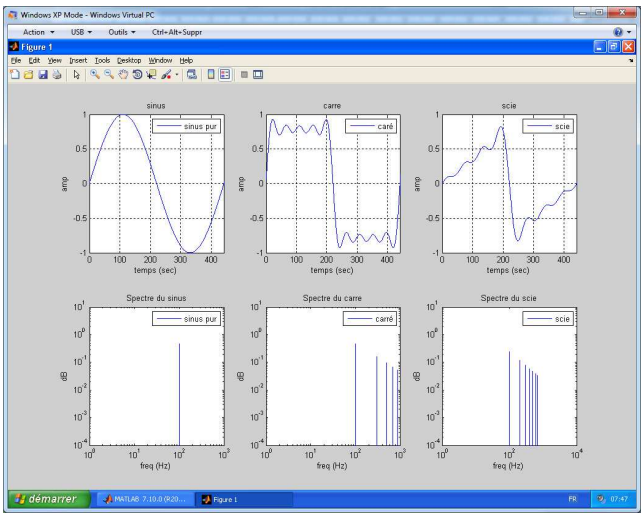


Fig. 5 : Suites harmoniques de nos trois signaux.

Expliquer ce qui ce produit quand on modifie l'affichage de la suite du signal carré. Qui sont les harmoniques situés à droite ? Quel rapport leurs fréquences entretiennent-elle avec la fréquence d'échantillonnages.

```
figure(6); semilogy(abs(sca)); title('Spectre du carre');
legend('carré');xlabel('freq (Hz)'); ylabel('dB');
```

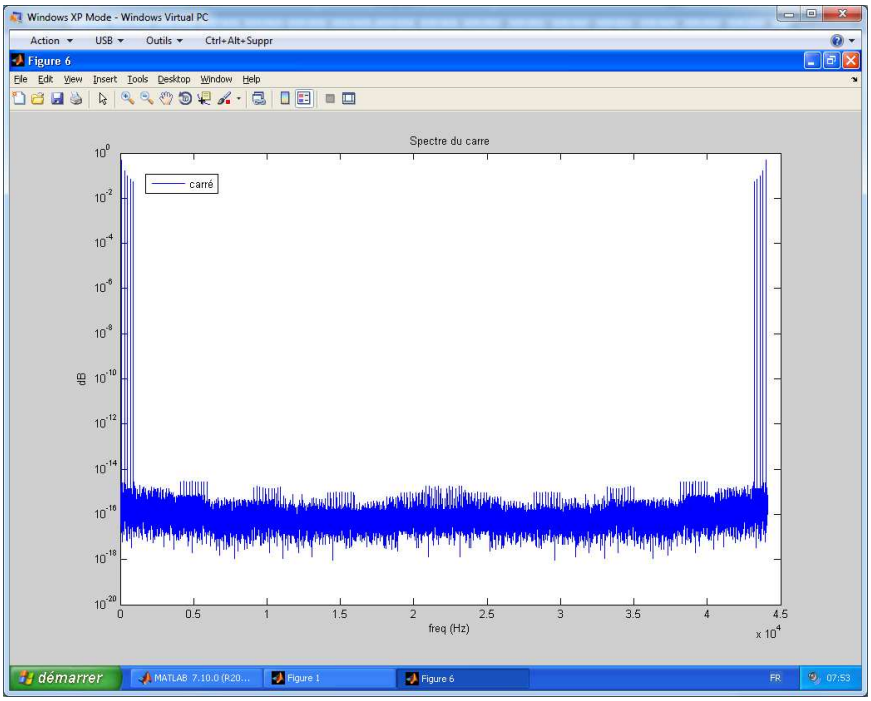


Fig. 6 : Repliement autour de Fe/2.

I.3.2_ Spectrogrammes.

A l'aide du code ci-dessous tracer le spectrogramme des signaux précédant

```
S=spectrogram(x)
S=spectrogram(x,window)
S=spectrogram(x,window,noverlap)
S=spectrogram(x,window,noverlap,nfft)
S=spectrogram(x,window,noverlap,nfft,fs)
spectrogram(Carre)….
```

Quelle influence ont les réglages du paramètre de la largeur de blocs (windows) et du recouvrement (overlap) ? Est il utile ici ?

```
figure(1);subplot(3,3,1);spectrogram(scie,1000,1);axis([0 1/2 75 1000])
figure(1);subplot(3,3,1);spectrogram(scie,1000,1);axis([0 1/2 75 1000])
figure(1);subplot(3,3,2);spectrogram(scie,1000,50);axis([0 1/2 75 1000])
figure(1);subplot(3,3,3);spectrogram(scie,1000,100);axis([0 1/2 75 1000])
figure(1);subplot(3,3,4);spectrogram(scie,100,1);axis([0 1/2 75 1000])
figure(1);subplot(3,3,5);spectrogram(scie,100,50);axis([0 1/2 75 1000])
figure(1);subplot(3,3,6);spectrogram(scie,100,100);axis([0 1/2 75 1000])
figure(1);subplot(3,3,7);spectrogram(scie,10,1);axis([0 1/2 75 1000])
figure(1);subplot(3,3,8);spectrogram(scie,10,50);axis([0 1/2 75 1000])
figure(1);subplot(3,3,9);spectrogram(scie,10,100);axis([0 1/2 75 1000])
```

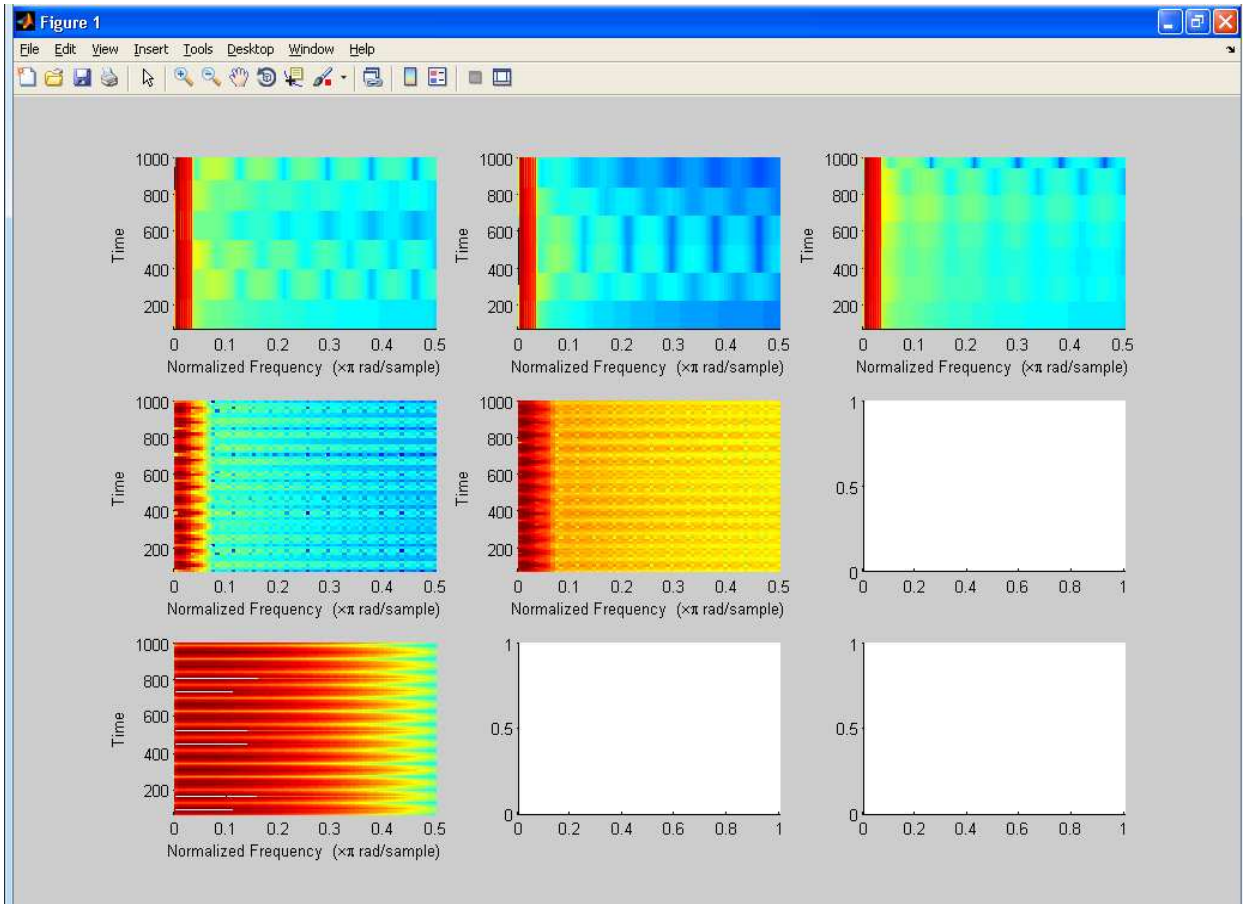


Fig. 7 : Fenêtrage vs overlap@ npf=(defaut).

```
figure(2);subplot(3,3,1);spectrogram(scie,1000,1,1000);axis([0 1/2 75 1000])
figure(2);subplot(3,3,2);spectrogram(scie,1000,50,1000);axis([0 1/2 75 1000])
figure(2);subplot(3,3,3);spectrogram(scie,1000,100,1000);axis([0 1/2 75 1000])
figure(2);subplot(3,3,4);spectrogram(scie,100,1,1000);axis([0 1/2 75 1000])
figure(2);subplot(3,3,5);spectrogram(scie,100,50,1000);axis([0 1/2 75 1000])
figure(2);subplot(3,3,6);spectrogram(scie,100,100,1000);axis([0 1/2 75 1000])
figure(2);subplot(3,3,7);spectrogram(scie,10,1,1000);axis([0 1/2 75 1000])
figure(2);subplot(3,3,8);spectrogram(scie,10,50,1000);axis([0 1/2 75 1000])
figure(2);subplot(3,3,9);spectrogram(scie,10,100,1000);axis([0 1/2 75 1000])
```

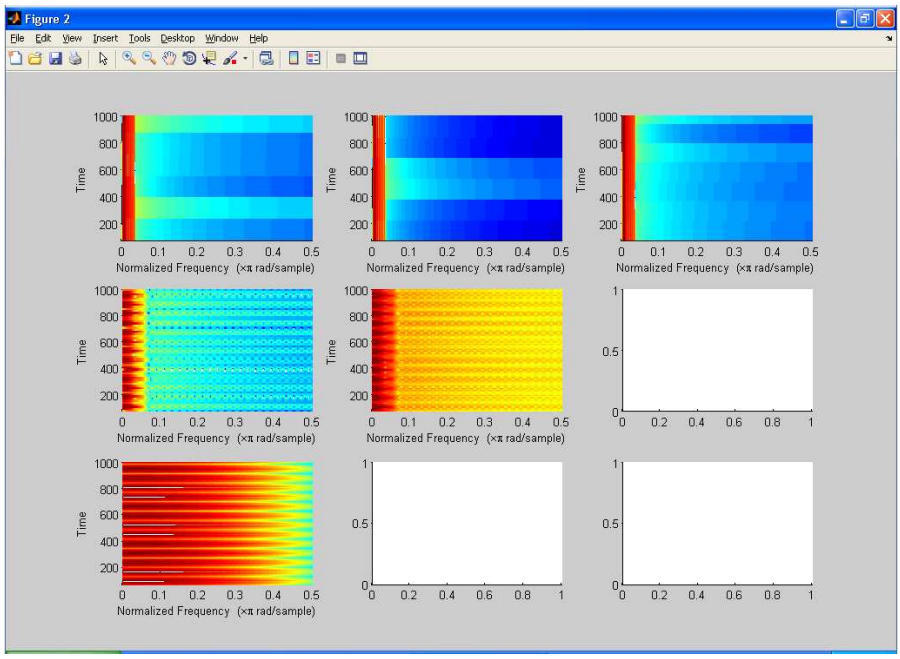


Fig. 8 : fenêtrage vs overlap @ npf=1k pts freq.

```
figure(3); spectrogram(carre,1000,1,1000);axis([0 1/2 75 1000])
figure(4); spectrogram(carre,1000,50,1000);axis([0 1/2 75 1000])
figure(5); spectrogram(carre,1000,100,1000);axis([0 1/2 75 1000])
figure(6); spectrogram(carre,100,1,1000);axis([0 1/2 75 1000])
figure(7); spectrogram(carre,100,50,1000);axis([0 1/2 75 1000])
figure(8); spectrogram(carre,10,1,1000);axis([0 1/2 75 1000])
```

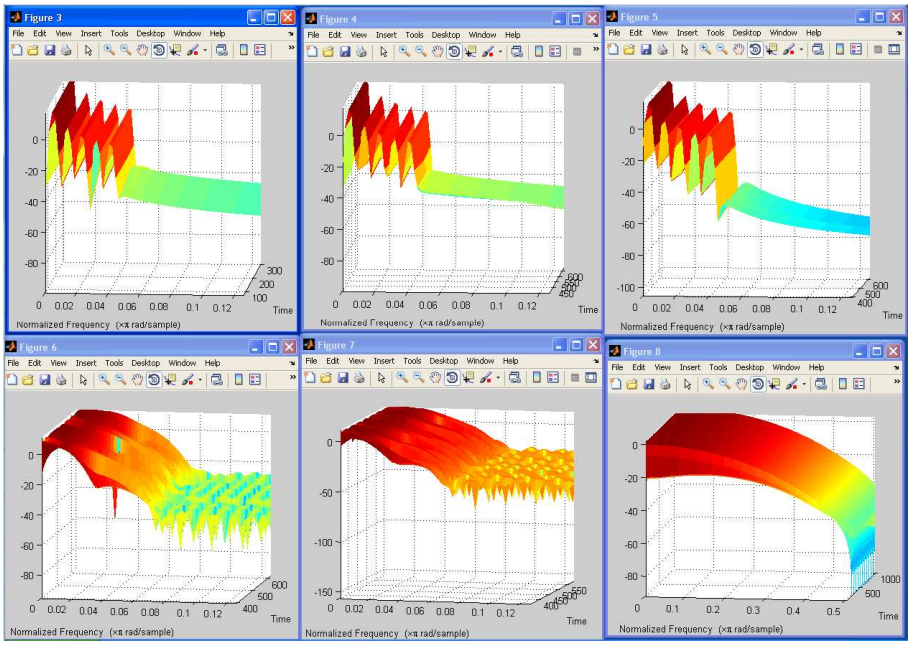


Fig. 9 : Influence de la taille de blocs.

I.4_ Ouverture de fichier .wav

Ouvrir les fichiers contenus dans le dossier /Sons et les afficher sur trois graphiques différents.

Graph1 : Temps magnitude
Graph2 : Fréquences magnitude
Graph3 : Temps fréquence magnitude

```
S=spectrogram(x)
S=spectrogram(x,window)
S=spectrogram(x,window,noverlap)
S=spectrogram(x,window,noverlap,nfft)
S=spectrogram(x,window,noverlap,nfft,fs)
```

Décrire les fichiers et trouver le bon overlap pour afficher leur spectrogramme. Quel rapport l'overlap entretient-il avec les signaux tonal et impulsifs ?

```
subplot(321);spectrogram(data,1000,1,1000); title('mauvais37');
subplot(322);spectrogram(data2,1000,50,1000); title('mauvais29');
subplot(323);plot(data); xlabel('Temps'); ylabel('Mag');
subplot(324);plot(data2); xlabel('Temps'); ylabel('Mag');
subplot(325);plot(abs(data)); xlabel('freq (Hz)'); ylabel('dB');
subplot(326);plot(abs(data2)); xlabel('freq (Hz)'); ylabel('dB');
```

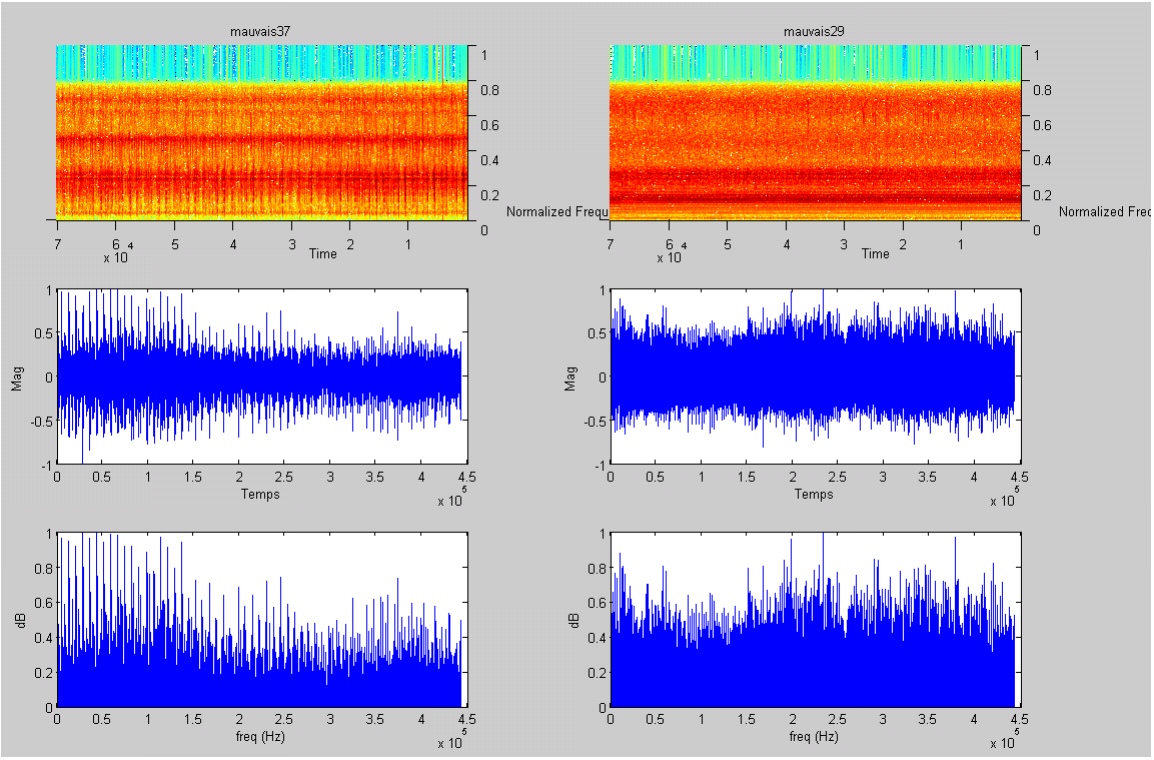


Fig. 10 : Signaux de roulements défectueux.

II_ Les Filtres

Soit une fonction de filtrage y=1+exp(-2*pi*j*f) .Tracer le gain et la phase de la fonction en fonction de la fréquence ainsi que sa réponse impulsionnelle.

Gain et phase :

```
f=[0:0.01:1];
gaincplxe=1+exp(-2*pi*j*f);
gain=abs(gaincplxe);
phase=angle(gaincplxe)*180/pi; %phase en degres
subplot(211);plot(f,gain);
subplot(212);plot(f,phase);grid
```

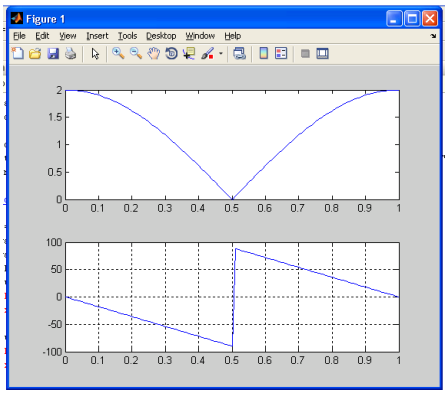


Fig. 11 : Fonction de transfert.

Réponse impulsionnelle :

```
N=60;
tps=[0:N-1];
theta=pi/12;
rho=.96;
rep=rho.^tps.*sin((tps+1)*theta)/sin(theta);
stem(tps,rep);grid
```

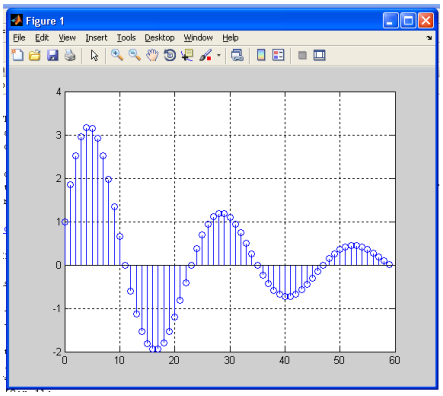


Fig. 12 : Réponse impulsionnelle.

Après avoir divisé par mille la fréquence de coupure du gain complexe, appliquer ce filtrage aux signaux précédant.

```
vin=carre;
f=[0:100:20000];
gaincplxe=1+exp(-0.002*pi*j*f);
vout=filter(gaincplxe,1,vin);
figure(3);subplot(2,2,1);plot(t,vin);axis([0 0.01 -1 1])
figure(3);subplot(2,2,2);plot(t,vout);axis([0 0.01 -500 500])
figure(3);subplot(2,2,3);spectrogram(vin,1000,1,1000);
axis([0 1/3 75 1000])
figure(3);subplot(2,2,4);spectrogram(vout,1000,1,1000);
axis([0 1/3 75 1000])
```

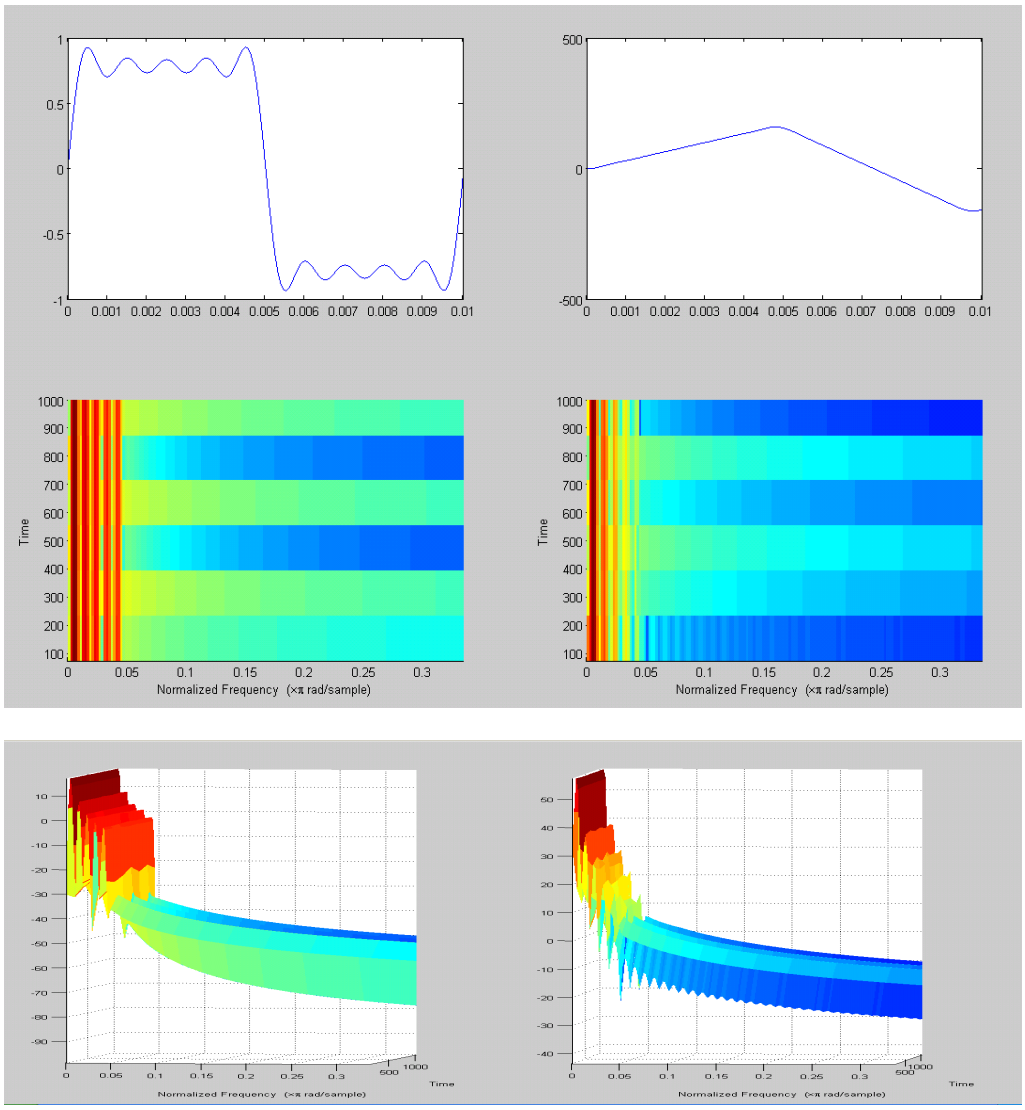


Fig. 13 : Signal carré à gauche et signal carré filtré à droite.

TP 3: initiation au traitement d'image sous Matlab

Image Processing Toolbox :

Les fonctions de la librairie _ traitement d'image _de MATLAB :
_ Lecture, écriture et affichage d'une couleur ou niveau de gris,
_ Transformations spatiales et transformations fréquentielles,
_ Filtrage linéaire et non linéaire,
_ Binarisation et morphologie mathématique,
_ Analyse, et restauration d'image,
_ Changement d'espace couleur,
_ . . .

Codage d'une image, représentation spatiale :

Une image est considérée comme un ensemble de points ou pixels (picture element), associé au quadrillage rectangulaire de l'image d'origine. La représentation d'une image se fait donc par l'intermédiaire d'une matrice d'entiers codés entre 0 et 255. Les images en niveau de gris sont représentées par des matrices 2D, les images couleurs représentés par 3 composantes (Rouge, Vert, Bleu) sont représentées par des matrices 3D. On accède à un pixel grâce a son indice de ligne et son indice de colonne. Le premier pixel d'une image est le pixel en haut à gauche. Cette représentation est appelé représentation spatiale de l'image.

I.2_ Remplissage.

Créer une matrice carrée de 100 de coté contenant des zéros et remplir le centre par une matrice carrée contenant des 1 de 30 de coté.

```
%Premiere image
b=zeros(100,100);
b(35:65,35:65)=255;
figure;imshow(b);
figure;surf(b);
```

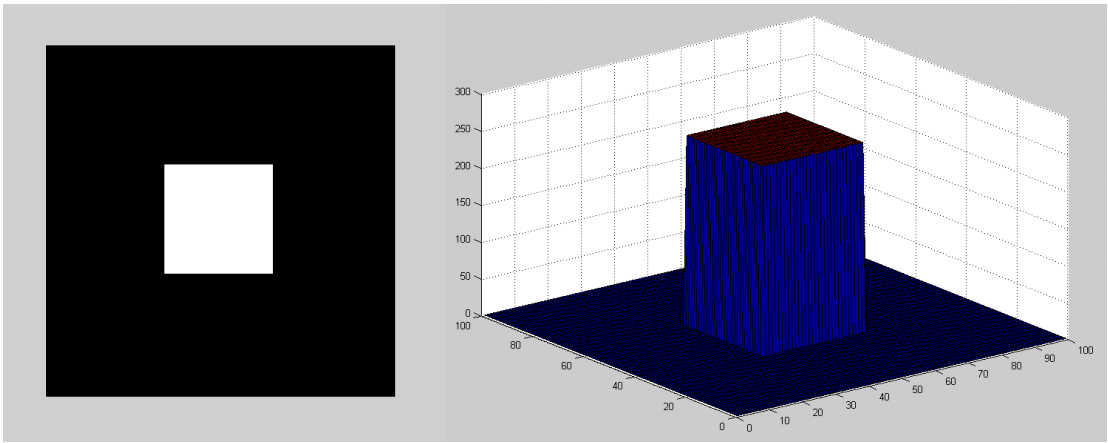


Fig. 14 et 15 : Affichage d'une matrice image à gauche et l'amplitude des niveaux à droite.

Quelles sont les différences entre les deux modes d'affichage ?

I.2_ Ouverture de fichier.

L'ouverture de fichier est réalisée par la fonction « `imread` » et charge dans une variable un fichier préalablement importé

```
p = imread('cameraman.tif'); % lire une image
```

I.3_ Histogramme.

A l'aide de la fonction « `imhist()` » réaliser l'histogramme de l'image précédente. En plaçant un seuil sur le niveau de gris créer une image binaire comparer les histogrammes et tracer à main levée la fonction de transfert qui lie les deux images.

```
seuil = 100
b = (p<seuil); % seuillage et créer une image binaire
% affichage, compare 2 images
subplot(1,2,1);
imshow(p,[0 512]);
subplot(1,2,2);
imshow(b);
impixelinfo;
```

Comparaison.

```
figure(3);subplot(2,2,1);imshow(p)
figure(3);subplot(2,2,2);imhist(p)
figure(3);subplot(2,2,3);imshow(b)
figure(3);subplot(2,2,4);imhist(b)
```

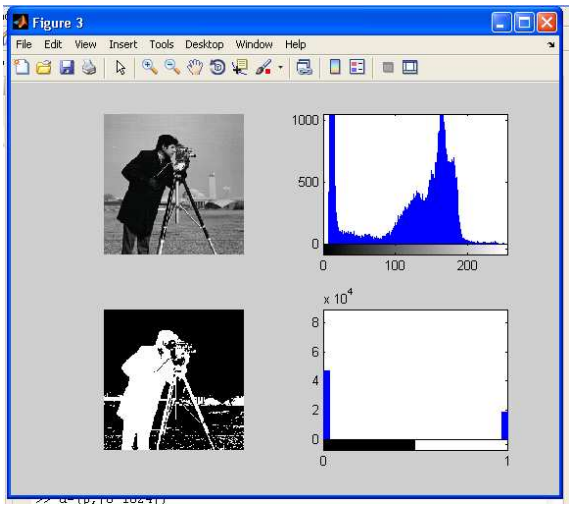


Fig. 16 : comparaison des histogrammes d'une image modifiée.

I.3.1_Exercice : Suivi de température par détection de niveau max()

Soit une camera thermique ayant une sensibilité de 4°C/niveau de gris ayant une gamme de température admissible comprise entre 0 et 1000°C.

A combien de degrés correspondent le niveau de gris 0x0F ?

A laide de la fonction max(), capturer la valeur maximum des pixels de l'image d'entrée et arrêter le four à la température 128°C

```
ma=1 ;            % marcher-arret = on : four en marche
pm=max('cameraman.tif')

if  pm > 32

      ma=0 ;

else

      ma=1 ;

end
```

Les filtres linéaires par convolution.

Le produit de convolution a*b est une méthode de transformation de signal facilement compréhensible grâce moyenne mobile. En effet pour un signal mono-vecteur S(n), la moyenne mobile MS(n) est la moyenne des m points adjacents tout au long du signal.

MS(1)=(Σ S(1) à S(m+1))/m

MS(n)=(Σ S(n) à S(m+n))/m

Ici la largeur de fenêtre est m et lors de la somme, les échantillons juxtaposés ne subissent pas de pondération. C'est une fenêtre carrée. D'autre fonctions sont couramment utilisées comme la fenêtre triangulaire.

Chaque fonction correspond à une fonction de filtrage bien précise ; voici quelques exemples de filtres linéaires mono vecteur :

filtre moyenneur et gaussien,	[1/9 1/9 1/9 1/9 1/9 1/9 1/9 1/9 1/9]
filtre passe haut, passe bas, passe bande,	[0 -1 0 -1 5 -1 0 -1 0]
	[1 1 1 1 4 1 1 1 1]
filtre dérivateur (Prewitt, Sobel),	
filtre laplacien,	[-1 -1 -1 -1 8 -1 -1 -1 -1]

Tracer les différentes fonctions sur une même feuille en y ajoutant les noms des axes et des graphiques.

I.4_ Filtrage d'une image via une fonction de test.

Le filtrage par convolution est réalisé par la fonction conv2(x,y) où x est le signal image et y le signal filtre. Créer une fonction qui affiche l'image de départ et les images filtrées.

```
nf=4 ;            % Comptreur de filtres
ftr1=[1/9 1/9 1/9 1/9 1/9 1/9 1/9 1/9 1/9]; %filtre 1 à 4
ftr2=[0 -1 0 -1 5 -1 0 -1 0];
ftr3=[1 1 1 1 4 1 1 1 1];
ftr4=[-1 -1 -1 -1 8 -1 -1 -1 -1];

mxftr=[ftr1;ftr2 ;ftr3 ;ftr4] ; %creation d'une banque de filtres

imin=imread('cameraman.tif');
```

```
function [ imout ] = cmp_ftr( nf,mxftr,imin )
%================cmp_ftr.m====================================
%   fonction de comparaison de filtres.
%
% affiche les filtres et leurs incidences sur une image de départ
%
% entrée :  nf          nombre de filtres
%           mxftr       banque de  filtres colonne 1 à n
%           imin        image de depart
%sortie :   imout       image d'arrivee
%
%=============================================================

fn=1 ;      %creation du pointeur de filtres courants
figure(1);subplot(2,4,1);imshow(imin);
imin=double(imin)/255;
figure(1);subplot(2,4,2);imshow(imin);

for fn = 1 : nf
    ftrn=mxftr(fn:fn,1:9);%capture du filtre dans la banque
    imout=conv2(imin,ftrn);

    figure(1);subplot(2,4,fn+4);imshow(imout);

end
%=================fin de la fonction============================

end
```

```
cmp_ftr(nf, mxftr, imin, )% appel de fonction
```

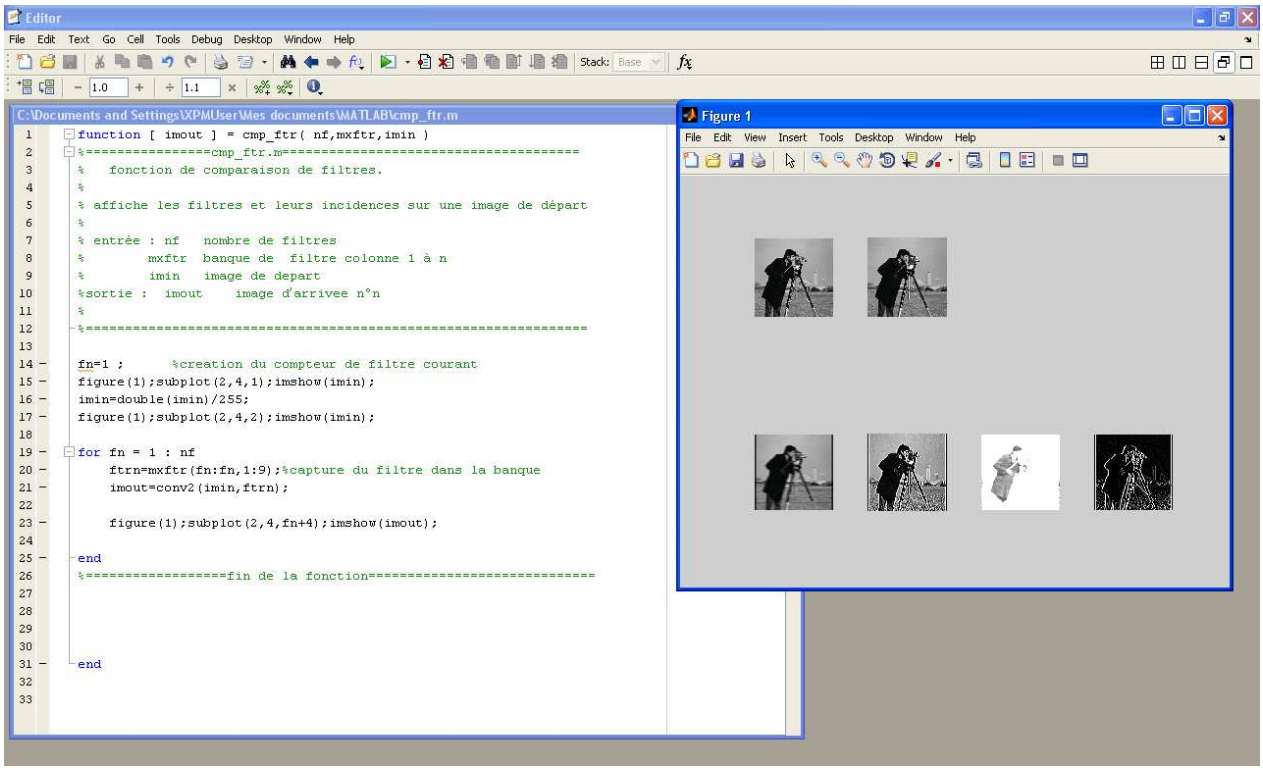


Fig. 17 : Comparaison des images filtrées.

I.5_ Utilisation des filtre pour dé-bruiter et renforcer une image

Le fichier « bruitflou.m » dégrade une image avec un filtre gaussien et applique différents filtre de restauration et de détection de contours. L'exécuter et modifier les filtres.

```
%Convolution Laplacien
im=imread('cameraman.tif');
im=double(im)/255;
im=imnoise(im,'gaussian',0,0.1);% bruitage de l'image
l=[-1 -1 -1;-1 8 -1;-1 -1 -1];
figure;imshow(im);
imf=conv2(im,l);              % renforcement des contours
figure;imshow(imf);
g=ones(5,5)/25;% elimination du bruits et conservation des détails
%g=[1 1 1;1 1 1;1 1 1]/9;
%g=[1 2 1;2 4 2;1 2 1]/16;
im=conv2(im,g);
figure;imshow(im);
%imf=conv2(im,l);
%figure;imshow(imf);
sx=[-1 -2 -1;0 0 0;1 2 1]; ;% filtre gardian directionnel x
sy=[-1 0 1;-2 0 2;-1 0 1]; ;% filtre gardian directionnel y
ga=conv2(im,sx);
gb=conv2(im,sy);
figure;imshow(ga);
figure;imshow(gb);
```

```
figure;imshow(ga+gb);

norme=sqrt(ga.^2+gb.^2);

coutour=zeros(258,258);
m=mean(mean(norme));
for ii=1:258
for jj=1:258
if(norme(ii,jj)>m)contour(ii,jj)=1;end
end
end
figure;imshow(contour);
```

II_ Création de fonctions.

II.1_ Création de fonction de convolution.

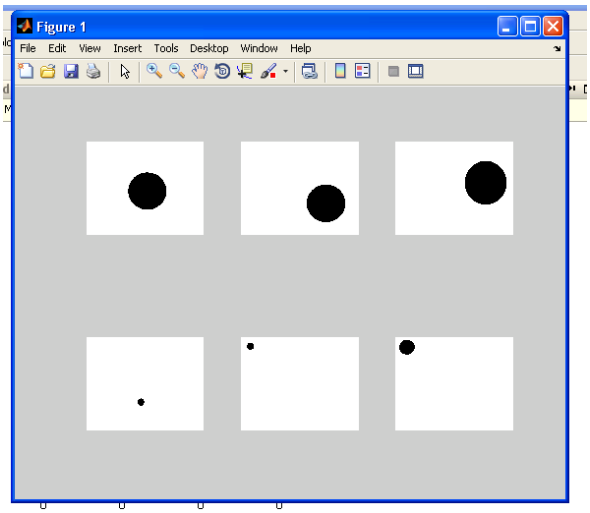
A l'aide de fonction « for », réaliser une fonction qui pour chaque pixel de sortie calcule la moyenne des pixels d'entrés pondérés par le filtre :

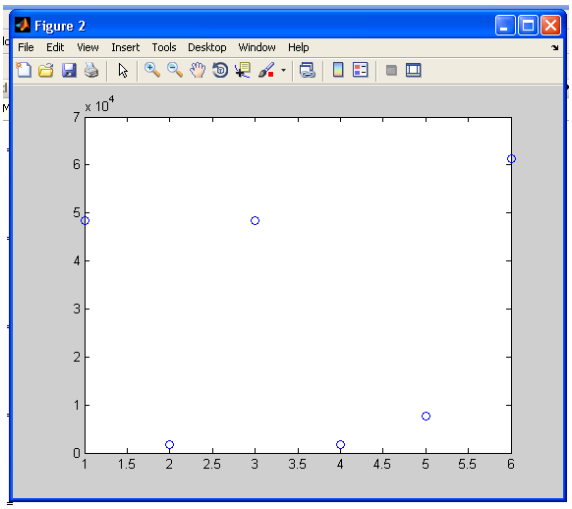
Pxo(1,1)=(pxi(1,1)*filtre(1,1)+….+pixi(3,3)*filtre(3,3))/9

```
imin=imread('cameraman.tif');
ftr=[-1 -1 -1;-1 8 -1;-1 -1 -1];    convo(imin,ftr)

function [ imout ] = convo( imin ,ftr)
%================convo.m=======================================
%   fonction de convolution.
%
% pondere le pixel de sortie en fonction des 9 pixels d'entrée
%
% entrée :  imin  image de depart
%           ftr   filtre3x3
% sortie :  imout    image d'arrivee
%
%================================================================
r=1
imout=zeros(260,260);
imin=double(imin);
for i=1:240
    for j=1:240
        for a=0:2
            for b=0:2
                 r=r+(imin(i+a,j+b)*ftr(a+1,b+1))/200
            end
        end
        r=r/9 ;
        imout(i,j)=r;

    end
end

figure(4);imshow(imout)
end
```

TP5 KNN (k nearest neighbours)

```
g1=cdata;
 imshow(g1)
 g2=cdata;
 g3=cdata;
 p1=cdata;
 p2=cdata;
 p3=cdata;
 figure(1); subplot(1,3,1);imshow(g1)
 figure(1); subplot(2,3,1);imshow(g1)
 figure(1); subplot(2,3,2);imshow(g2)
 figure(1); subplot(2,3,3);imshow(g3)
 figure(1); subplot(2,3,4);imshow(p1)
 figure(1); subplot(2,3,5);imshow(p2)
 figure(1); subplot(2,3,6);imshow(p3)
```

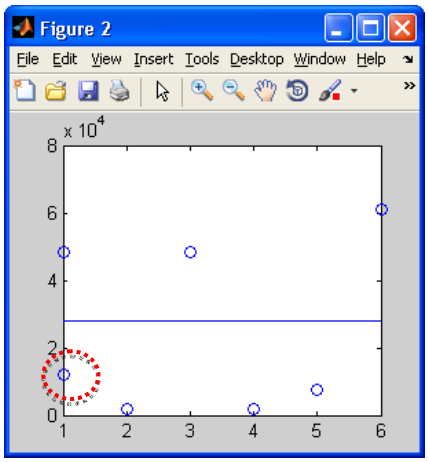




Détection des vecteurs taille

```
function [ npxo ] = pixzerocount( imin )
%calcule le nombre de pixel à zero d'une image
%
npxo=0
for i=1:614
    for j=1:768
        if imin(i,j)<0.5
            npxo=npxo+1
        end
    end

end
```

```
nbg1=pixzerocount(g1);
function [ taille ] = knn( imin,lim )
%descrimine un rond par nb de pixel pa raport a une limite
%   Detailed explanation goes here
nbnouv=pixzerocount(imin);
if nbnouv<lim
    taille='petit'
else
    taille='gros'
end

nbg2=pixzerocount(g2);
nbg3=pixzerocount(g3);
nbp1=pixzerocount(p1);
nbp2=pixzerocount(p2);
nbp3=pixzerocount(p3);

ecart=nbg1-nbp3
lim=nbg1-ecart/2
nouv=cdata
taille=knn(nouv,lim)

npxo =       12263
taille =petit
```



Détection du nouveau fichier comme « petit »

Partiel n°1

Traitement du signal appliqué au son et à l'image.

I_ Variable. 1 pts

Quel type de multiplication faut-il utiliser pour obtenir la troisième matrice à partir des deux premières. Présentez le code)

1	2	3	4	5

0	0	1	1	0

0	0	3	4	0

II_ Vecteur & son. 7 pts

Création 1 pts
Affichage 2 pts

Soit un signal sinusoïdal x(t) d'amplitude 2, de fréquence1kHz et d'une durée de 10.5ms échantillonné toutes les 50µs.

Créer et afficher x(t) avec le nom des axes en utilisant la syntaxe suivante :

```
xlabel('___');ylabel('___');
```

Statistique et manipulation 4 pts

Ajouter au signal précédent une sinusoïde d'amplitude 1, de fréquence double et déterminer la valeur maximum à l'aide de la fonction m=max(x) .

Peut-on réaliser cette fonction avec des boucles if et for ?

III_ Matrice image. 12pts

Création et ouverture : 2 pts
Affichage : 4 pts

Créer une matrice de 25 par 25 et la remplir avec un damier. Afficher celle-ci dans une figure pouvant contenir quatre graphiques. Ouvrir l'image « cameraman.tif » et l'afficher également dans cette figure en position 2.

Manipulation : Création de la convolution : 6 pts

Créer une fonction de convolution qui appliquera un filtre « moyenne mobile » f=[1 1] sur le damier précédant avec un recouvrement de 1. Afficher le damier filtré en position 3 ainsi que l'image filtrée sur ces 24* 24 premiers pixels en position 4.