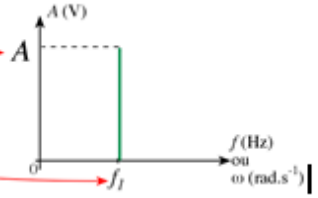
3 Représentation des signaux.

1. Ecrire un fichier qui crée un signal sinusoïdal de 440 Hz, d'une amplitude de 0,5 et d'une durée de 4s avec une fréquence d'échantillonnage de 22kHz.
   Ecouter et visualiser ce signal.
   Pourquoi n'est-il pas possible de visualiser ce signal ?
   Modifier ce signal pour que la visualisation soit possible. (par exemple en se limitant à 10 périodes)
   Ecouter le nouveau signal.
   Pourquoi n'entendez-vous rien ?
   (remarque : un signal peut être écouté directement à partir d'une carte son en utilisant l'instruction "sound")

Un peu de théorie sur les signaux :

Un signal sinusoïdal *u(t)* peut être représenté par un trait de hauteur (ordonnée) son amplitude et d'abscisse sa fréquence *f* ou sa pulsation ω.

$$u(t) = A.\sin(\omega.t + \varphi)$$



Un signal *s(t)* peut, après avoir été décomposé en une série de Fourier, être représenté de cette façon.
C'est ce que l'on appelle la <u>représentation en fréquence</u> ou la <u>représentation spectrale</u> ou le **<u>spectre</u>** de *s(t)*.

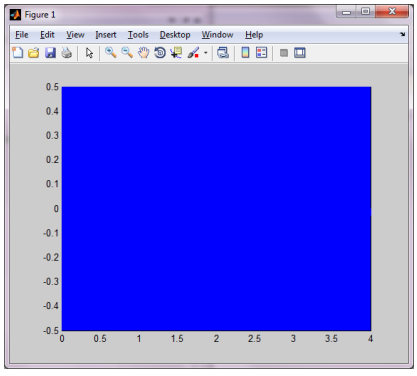
1. Ecrire un fichier qui crée un signal sinusoïdal de 440 Hz, d'une amplitude de 0,5 et d'une durée de 4s avec une fréquence d'échantillonnage de 22kHz.

Ecouter et visualiser ce signal.

```
clear all;
close all;

f = 440;
Fe = 22000;          %fréquence d'échantillonage
Te=1/Fe;             %période d'échantillonage (taux d'échntillonnage)
t = 0 : Te : 4;      %intervalle de 0 à 4s, avec une période
                     %d'échantillonnage de 1/Fe (taux d'échantillonnage)
A = 0.5;             %amplitude du signal

w = 2*pi*f;          % calcule de la pulsation (rad/s) d'un signal ==> sa
                     %vitesse angulaire

s = 0.5*sin(w*t);    %y signal périodique

plot(t,s);           % pour visualiser

output = 'ex3_1.wav';
WAVWRITE(s,Fe,output);

sound(s,Fe);         %pour l'ecouter
```

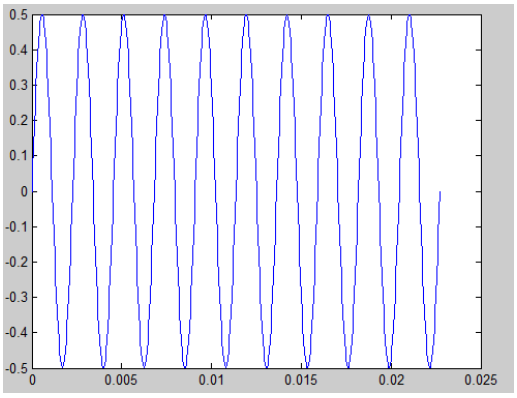
Pourquoi n'est-il pas possible de visualiser ce signal ?

La fréquence du signal étant trop élevé par rapport à sa Fe, les sinus sont énormément rapprochés au point qu'on ne voit qu'un fond bleu.



Modifier ce signal pour que la visualisation soit possible. (par exemple en se limitant à 10 périodes)

```
clear all;
close all;

f = 440;
Fe = 22000;          %fréquence d'échantillonage
Te=1/Fe;             %période d'échantillonage (taux d'échntillonnage)
t = 0 : Te : 10/f;   %intervalle de 0 à 4s, avec 10 périodes et un période
                     %d'échantillonnage de 1/Fe (taux d'échantillonnage)

A = 0.5;             %amplitude du signal

w = 2*pi*f;          % calcule de la pulsation (rad/s) d'un signal ==> sa
                     %vitesse angulaire

s = A*sin(w*t);    %y signal périodique

plot(t,s);           % pour visualiser

output = 'ex3_1.wav';
WAVWRITE(s,Fe,output);

sound(s,Fe);         %pour l'ecouter
```



Quatre secondes étant trop long pour ce signal, on nous demande d'utiliser 10 périodes. Une période = 1/440 donc 10 périodes = 10/440.

Période = temps écoulé après un tour complet du cercle trigonométrique donc temps que met une sinusoidale.

Ecouter le nouveau signal.

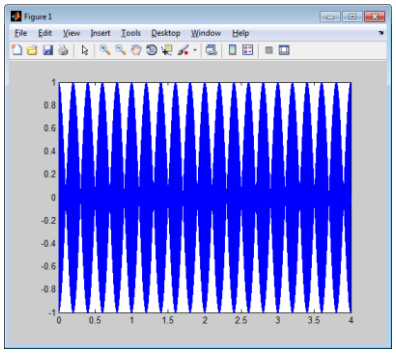
Pourquoi n'entendez-vous rien ? (remarque : un signal peut être écouté directement à partir d'une carte son en utilisant l'instruction "sound")

- **Signal trop court.**

2. Ecrire un fichier qui crée un signal qui est la somme d'un signal sinusoïdal de 440 Hz et d'un signal de 445 Hz d'une amplitude de 0,5 et d'une durée de 4s avec une fréquence d'échantillonnage de 22kHz. Visualiser et écouter le battement.

```
clear all;
close all;

f = 440;
f2 = 445;
Fe = 22000; %fréquence d'échantillonage
Te=1/Fe; %période d'échantillonage

t = 0 :Te:4;%intervalle de 0 à 4s, avec une période d'échantillonnage de
1/Fe

A = 0.5;        %amplitude

%1er signal
w = 2*pi*f;
y = A*sin(w*t); %y signal périodique

%2ème signal
w2 = 2*pi*f2;
y2 = A*sin(w2*t);

x = (y+y2);

plot(t,x); % pour visualiser

sound(x,Fe); %pour l'ecouter
```

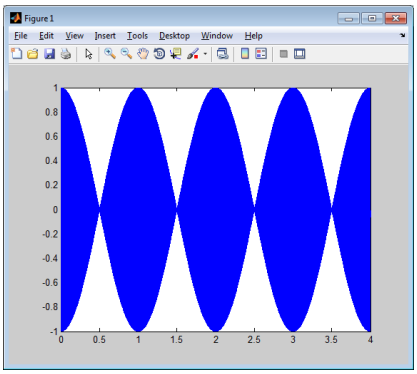
- En additionant les signaux on distingue un battement (au niveau du son mais aussi du graphe : zones blanches sur l'image).



3. Reprendre l'exemple précédent avec 440 et 441 Hz

```
clear all;
close all;

Fe = 22000; %fréquence d'échantillonage
Te=1/Fe; %période d'échantillonage

t = 0 :Te:4;%intervalle de 0 à 4s, avec une période d'échantillonnage de
1/Fe

A = 0.5;        %amplitude
w = 2*pi*440;
y = A*sin(w*t); %y signal périodique

w2 = 2*pi*441
y2 = A*sin(w2*t);

x = (y+y2);

plot(t,x); % pour visualiser

sound(x,Fe); %pour l'ecouter
```

- Avec 441, on distingue un battement plus lent.



4. Sachant que le "la" de la 3ème octave a une fréquence de 440Hz et que l'on obtient la note suivante d'une gamme en multipliant la note précédente par 2^(1/6), créer une succession de 18 notes au moyen d'une boucle for. (il faut remarquer que notre gamme n'est pas complète puisqu'elle ne tient pas compte des demi-tons entre le "mi" et le "fa" et entre de "sol" et le "do".
Voir par exemple le site "http ://fr.wikipedia.org/wiki/La fréquence d'échantillonnage sera de 11025Hz.
Ecouter le résultat obtenu.

```
fs = 11025;                     % échantillonage a 11Khz, FS =
                                    sampling frequency
  t = 0: 1/fs : 0.1;            % de 0 à 0.1 secondes (assez rapide)
  f = 440;                      % fréquence de base
  y = [];

  for i=1 : 18
        w = 2 * pi * f;         % omega = 2*pi*f
        x = 0.5 * sin(w*t);     % calcul du sinus = A*sin(wt)
        y = [y,x];              % concatenation en 1 ligne
        f =  f*2^(1/6);         % passer a la note suivante
  end ;

  sound(y,fs) ;
```

- **Le son devient plus aigue car la fréquence augmente.**

Dans l'exemple précédent on commencait par le La de la 3ème octave. Si l'on veut commencer au Do de la 3ème octave et faire Do Ré Mi Fa Sol La Si Do :

```
clear all;
close all;

  fs = 11025;                   % échantillonage a 11Khz, FS =
                                % sampling frequency
  t = 0: 1/fs : 0.5;            % de 0 à 0.1 secondes (assez rapide)
  f = 261.6;                      % fréquence de base
  y = [];

  for i=1 : 8
        w = 2 * pi * f;         % omega = 2*pi*f
        x = 0.01 * sin(w*t);     % calcul du sinus = A*sin(wt)
        y = [y,x];              % concatenation en 1 ligne
        f =  f*2^(1/6);         % passer a la note suivante
  end ;

  sound(y,fs) ;
```

Do de la 3ème octave = 261.6

5. Créer un signal de 440Hz d'une durée de 4s bruité. Pour produire ce résultat, utiliser la fonction "rand". Ecouter votre résultat.

```
FS = 22000;                    %Fréquence d'échantillonnage
t = 0 : 1/FS : 4;

f = 440 ;                      %fréquence
w = 2*pi*f;                    %vitesse angulaire
A = 0.1 ;                      %petite amplitude pour pas s'exloser les timpans
y = sin(w*t);                  %calcul du sinus = A*sin(wt)
                               %l'amplitude sera ajouté au signal bruité

r = rand(size(y));             %Génère une matrice avec des valeurs
                               %aléatoires (bruit)
y = (y + r)*A;
sound(y,FS);
```

Donc on créé un son avec des valeurs aléatoires (bruit). Vu qu'on veut l'additionner ce bruit au signal de base, il est nécessaire que les deux signaux aient la même taille sinon il est impossible de les additionner. C'est pourquoi on utilise size(y) dans le rand : pour que le signal généré par les valeurs aléatoires ait le même nombre de lignes (une ici vu qu'un son est un vecteur) et de colonnes. Size indique c'est 2 propriétés.

6. Créer une succession d'oscillations amorties. La fréquence des oscillations sera de 440 Hz. Un cycle d'oscillations sera de 0.5s et il faudra choisir un facteur d'amortissement adéquat. Visualisez et écouter le résultat.
Suggestion : utiliser l'opérateur de concaténation [x,y]

Un peu de théorie sur les oscillations amorties :

III. Oscillations amorties

- **Généralités**

Une oscillation amortie est une oscillation dont l'amplitude diminue progressivement au cours du temps. Cette diminution résulte d'une perte d'énergie causé par le déplacement dans un milieu visqueux.

L'enveloppe de la courbe est une exponentielle.

La formule d'un signal amorti est :

$$s(t) = A_0 . e^{-\delta t} \cos(\omega t + \varphi)$$

$A_0 = amplitude\ du\ signal\ au\ temps\ 0$
$e^{-\delta t} = exponentiel\ négative\ provoquant\ l'amortissement$
$\delta = facteur\ d'amortissement$
$\omega = 2\pi f$
$\varphi = déphasage$

```
clear all;close all;clc;

f=440;                  %la fréquence des signaux amorties
fs = 10000;             %fréquence d'échantillonnage
t=0:1/fs:0.5;           %le cylcle d'oscillation = 0.5s
delta=10;               %le facteur d'amortissement est de 10

s=exp(-delta*t).*sin(2*pi*f*t); %Signal amorti

s2=[];                  %tableau qui va regrouper toutes les signaux amortis
for i=1:10              %succession d'oscillations amorties
    s2=[s2,s];
end
plot(s2);
sound(s2,fs);
```

On voit bien que chaque signal amorti a une durée de 0.5s lorsque l'on met une fréquence d'échantillonnage de 10000.

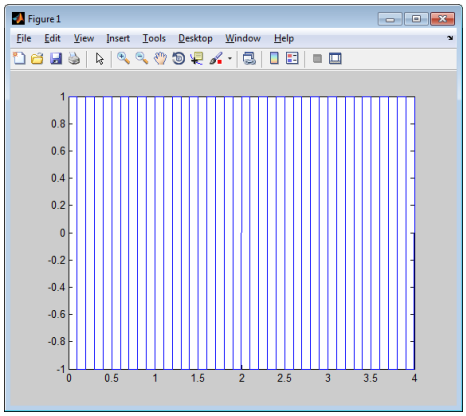
Nous avons utilisé la concaténation pour rassembler tous les 10 signaux amortis.

On ne spécifie pas l'amplitude, elle revient donc à 1. Il n'y a pas de déphasage dans la formule de s car il n'y a qu'un signal par amortissement.

7. Ecrire un programme pour tester les signaux suivants ainsi que leurs différents paramètres. Ecouter le résultat.

(a) square

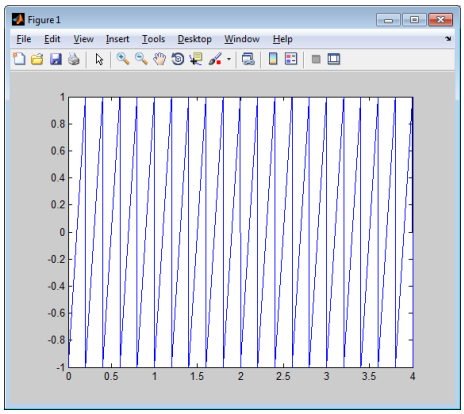
```
%carré sinusoidale

FS = 8000;
t = 0 : 1/FS : 4;
y = square(2*pi*5*t);

plot(t,y);
sound(y,FS);
```



On crée un signal carré de 5hz à partir d'une pulsation sa fréquence (5hz) et le temps.

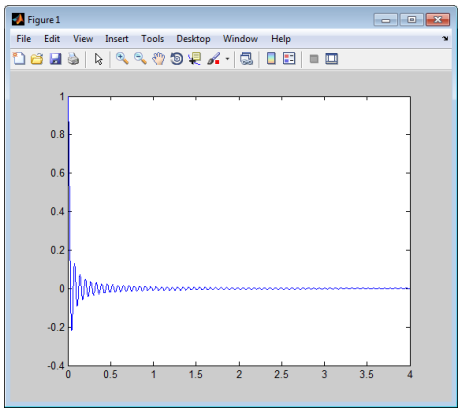
(b) sawtooth

```
%PIC sinusoidale

FS = 8000;
t = 0 : 1/FS : 4;
y = sawtooth(2*pi*5*t);

plot(t,y);
sound(y,FS);
```



(c) sinc

```
FS = 8000;
t = 0 : 1/FS : 4;
y = sinc(2*pi*5*t);

plot(t,y);
sound(y,FS);
```

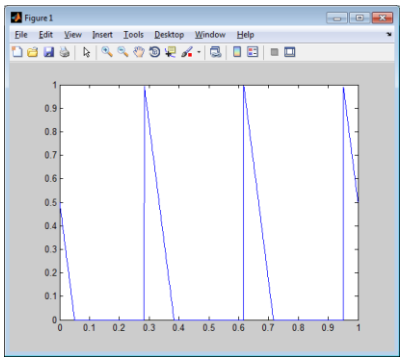


Le sinus cardinal est un sinus amorti.

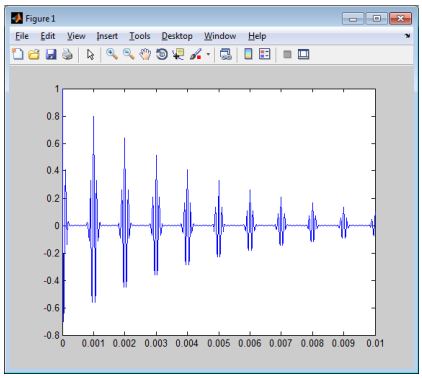
SINC : $\frac{\sin \pi x}{\pi x}$

(d) pulstran avec les options : rectpuls, tripuls, gauspuls

```
RECTPULS Sampled aperiodic rectangle generator.
  RECTPULS(T) generates samples of a continuous, aperiodic,
  unity-height rectangle at the points specified in array T, centered
  about T=0.  By default, the rectangle has width 1.  Note that the
  interval of non-zero amplitude is defined to be open on the right,
  i.e., RECTPULS(-0.5)=1 while RECTPULS(0.5)=0.

  RECTPULS(T,W) generates a rectangle of width W.


% Example 1: Generate an asymmetric sawtooth waveform with a
% repetition frequency of 3 Hz and a sawtooth width of 0.1 sec.
% The signal is to be 1 second long with a sample rate of 1kHz.

T = 0 : 1/1E3 : 1;  % 1 kHz sample freq for 1 sec
D = 0 : 1/3 : 1;    % 3 Hz repetition freq
Y = pulstran(T,D,'tripuls',0.1,-1); plot(T,Y)
```




```
% Example 2: Generate a periodic Gaussian pulse signal at 10 kHz,
% with 50% bandwidth.  The pulse repetition frequency is 1 kHz,
% sample rate is 50 kHz, and pulse train length is 10msec.  The
% repetition amplitude should attenuate by 0.8 each time.  Uses
% a function handle to refer to the generator function.

T = 0 : 1/50E3 : 10E-3;
D = [0 : 1/1E3 : 10E-3 ; 0.8.^(0:10)]';
Y = pulstran(T,D,@gauspuls,10E3,.5); plot(T,Y)
```



(e) chirp

```
FS = 8000;
t = 0 : 1/FS : 4;
y = chirp(2*pi*5*t);

plot(t,y);
sound(y,FS);
```

?????sert à quoi ?????????