Table of Contents


```
% 4 - Echantillonage

clear all;
close all;
```

Exercice 4.1 echantillonnage ideal d'un signal

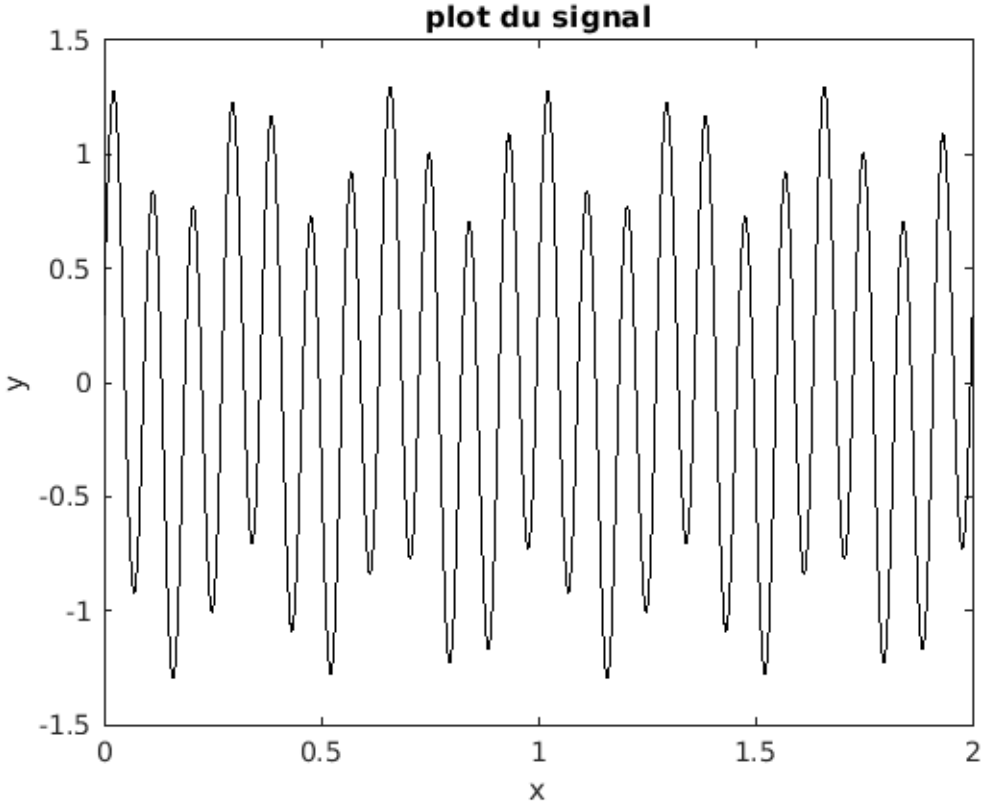
```
T = 2;  % Duree du signal
time_step = T/5000;
t_interv=0:time_step:T;
nu_1=11; nu_2=3;
fun = @(t_interv) sin(2*pi*nu_1*t_interv)+0.3*cos(2*pi*nu_2*t_interv);
signal = fun(t_interv);
% d'apres le theoreme de shannon, pour une reconstruction parfaite, il
 faut
% chosir un pas d'echantillonnage T<1/2*B soit T<1/4 et T>1/4 pour ne
 pas
% reconstruire le signal.
```

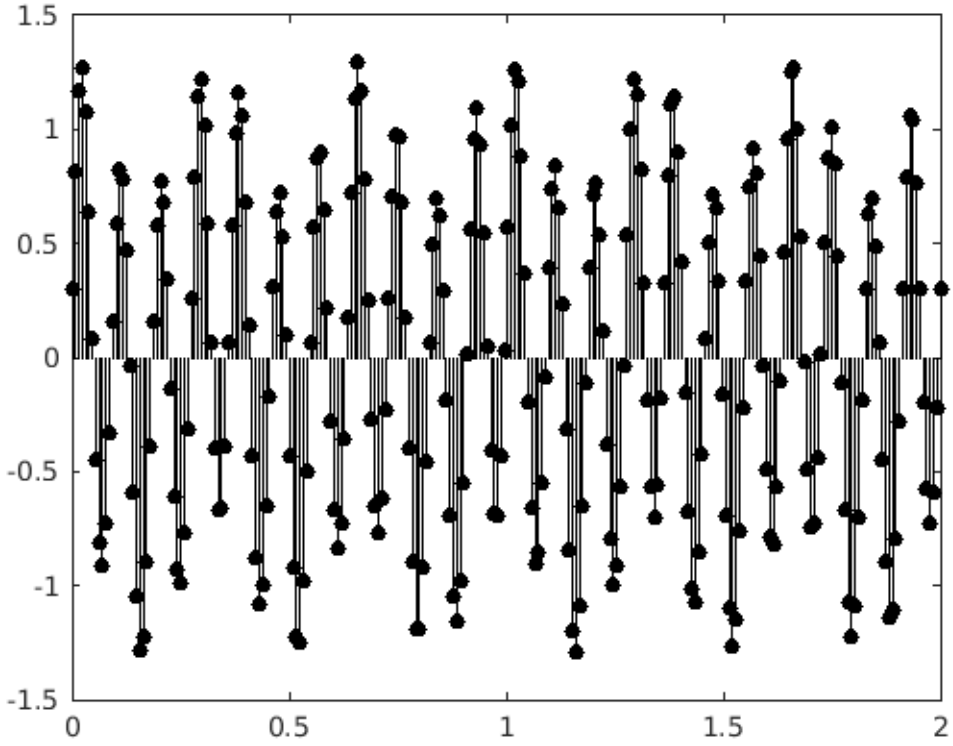
Exercices 4.1.1 et 4.1.2 Signal bien echantillonne et sa reconstruction / Signal sous-echantillonne et sa reconstruction

Question 1

```
figure(1)
```

```
% plot du signal

periode_ech = 0.01;
x1= t_interv;
y1=signal;
plot(x1,y1,'k');
xlabel('x');
ylabel('y');
title('plot du signal');

% parametres pour l'echantillonnage (a ne pas modifier)
no_ech = T/periode_ech+1;                  % nombre d'echantillons
no_ech = 2^nextpow2(no_ech);               % nombre d'echantillons
 arrondi a la premiere puissance de 2 (plus simple pour calculer sa
 transformee)
periode_ech = T/(no_ech-1);                % periode d'echantillonage
 ajoute
echantillons = 0:periode_ech:T;            % instants d'echantillonage
signal_echantillone = fun(echantillons);   % signal d'echantillon

% Visualisation signal echantillonns
figure(2)
x1=echantillons;
y1=zeros(1,length(x1));
x2=echantillons;
y2=signal_echantillone;
plot([x1;x2],[y1;y2],'k');
hold on;
scatter(x2,y2,'k','filled');
```

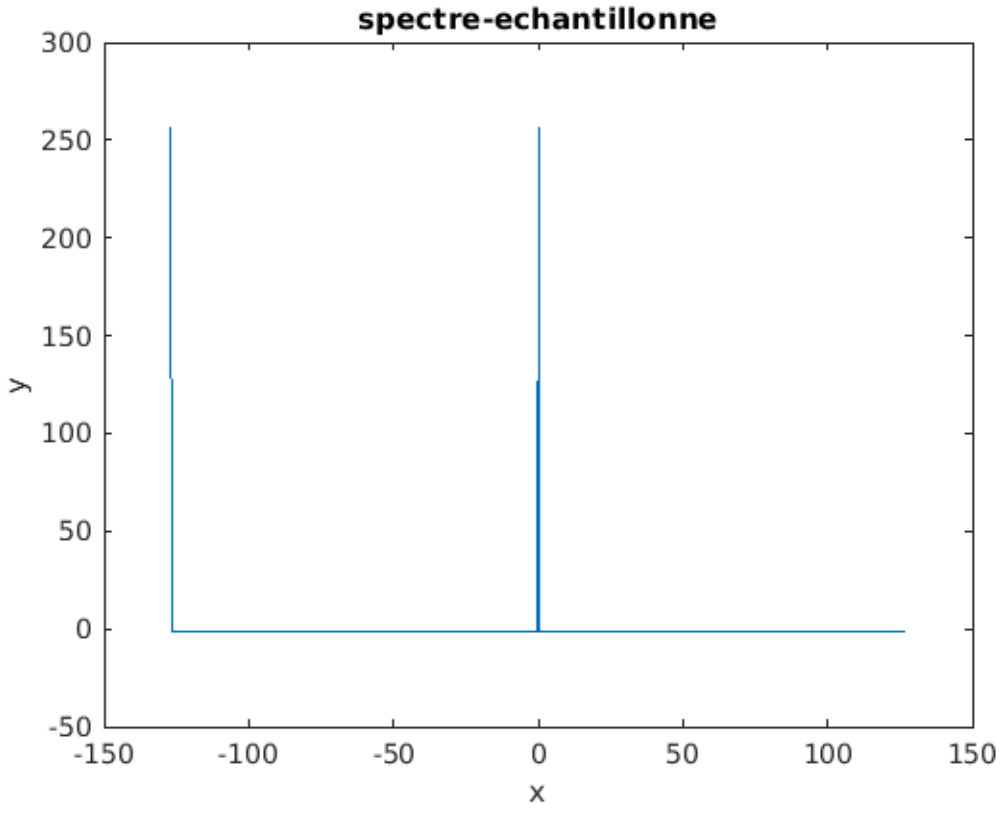

plot du signal
1.5
1
0.5
0
-0.5
-1
-1.5
y
0
0.5
1
1.5
2
x


1.5
1
0.5
0
-0.5
-1
-1.5
0
0.5
1
1.5
2

Question 2

```
spectre = fft(echantillons);
% Definition de l'axe des frequences (ne pas modifier)
freq_step=1/periode_ech/no_ech;
freq = -1/periode_ech+freq_step:freq_step:1/periode_ech-freq_step;
spectre_echantillonne(1:no_ech) = spectre;            % Definition du
 spectre pour frequences negatives (pas fait par Matlab)
spectre_echantillonne(no_ech:1:2*no_ech-1) = spectre; % Definition du
 spectre pour frequences negatives (pas fait par Matlab)
figure(3);
% plot du module du "spectre_echantillonne"

plot(freq,spectre_echantillonne);
xlabel('x');
ylabel('y');
title('spectre-echantillonne');
```

Warning: Imaginary parts of complex X and/or Y arguments ignored.

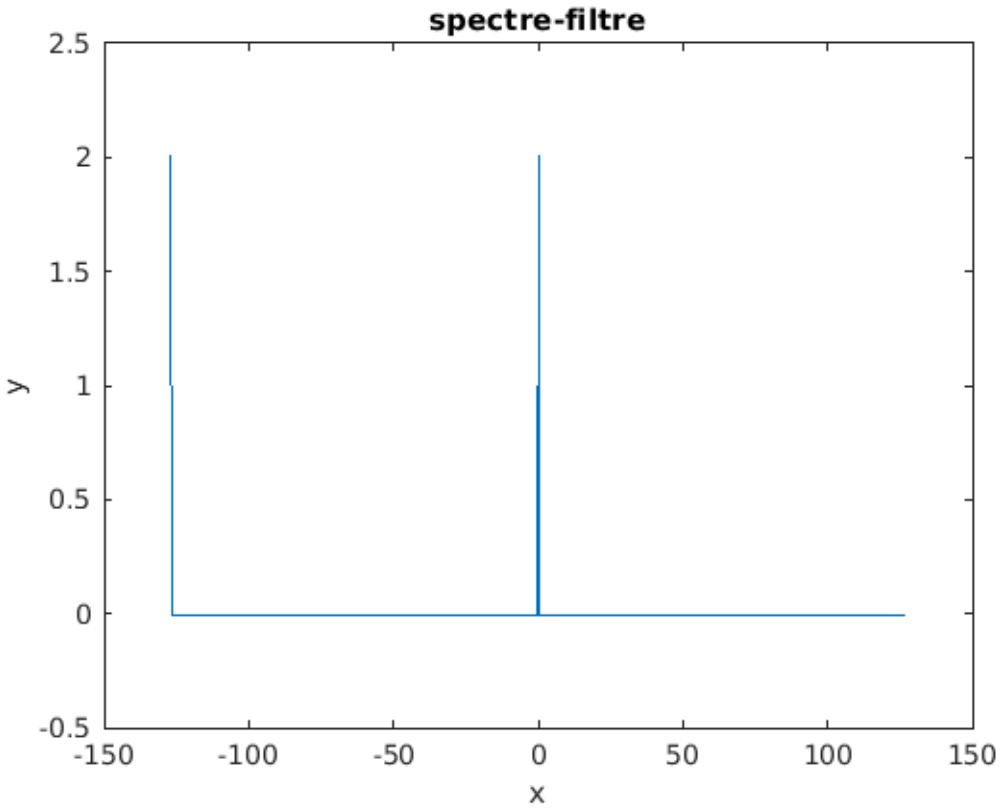


Question 3

```
% Filtrage de la transformee de Fourier du signal echantillons
 (necessaire avant de reconstruire le signal)
sigma=1/periode_ech;
```

```
fun1 = @(t_interv)  porte(t_interv,-sigma/2,sigma/2);
f1=fun1(periode_ech);
spectre_filtre = spectre_echantillonne.*f1;
figure(4);
%  on supprime les repliques avec un #ltre passe-bas ideale de largeur
 1 T et amplitude T.
% plot du module du "spectre_filtre"
plot(freq,spectre_filtre);
xlabel('x');
ylabel('y');
title('spectre-filtre');
```

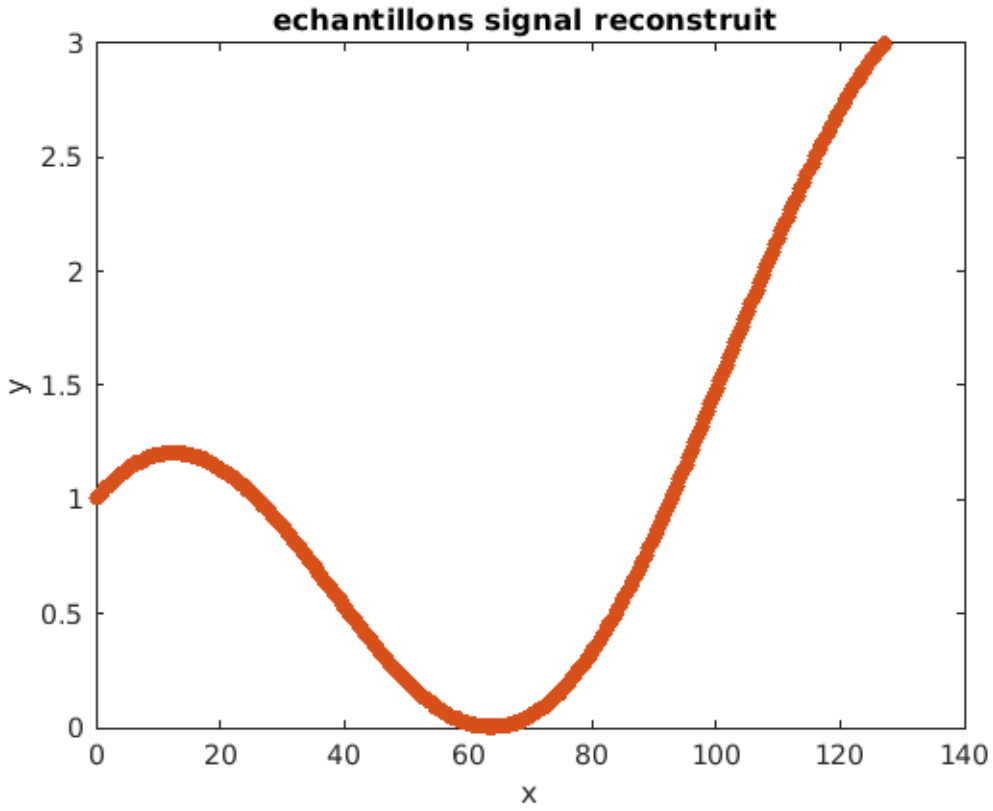
Warning: Imaginary parts of complex X and/or Y arguments ignored.



Question 4

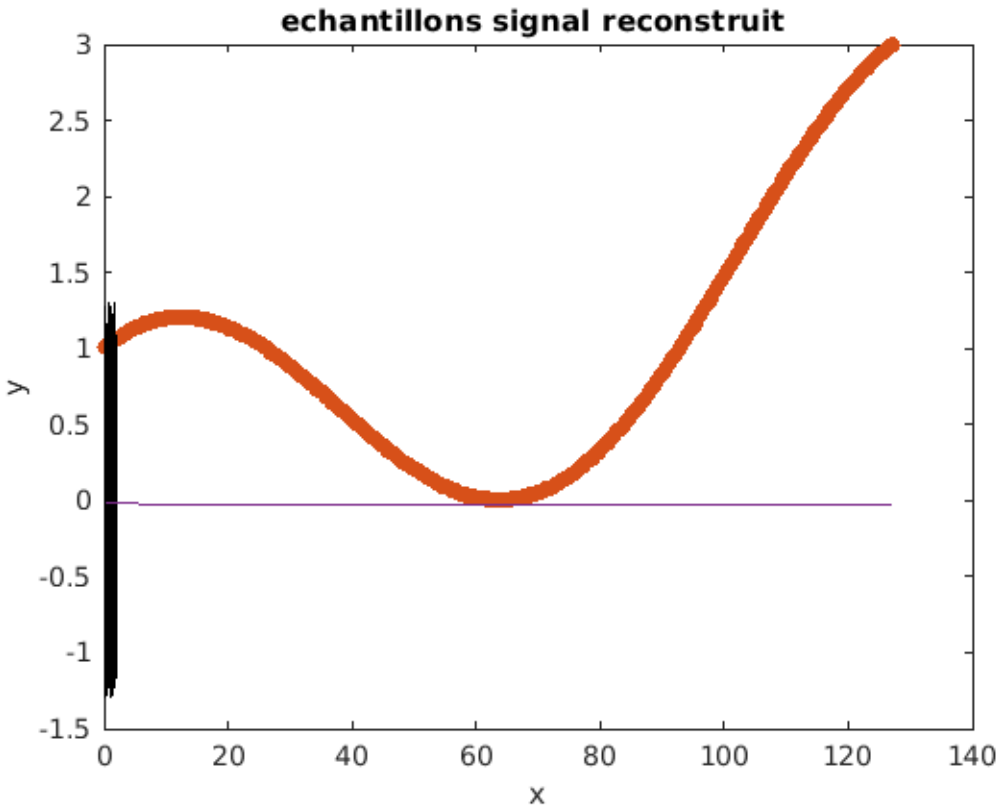
```
% Inversion de la transformee de Fourier filtree (ne pas modifier)
spectre_pour_inversion(1:0.5*no_ech)        =
 spectre_filtre(no_ech:no_ech+0.5*no_ech-1);
spectre_pour_inversion(0.5*no_ech+1:no_ech) =
 spectre_filtre(no_ech-0.5*no_ech+1:no_ech);
freq_pour_inversion=0:freq_step:1/periode_ech-freq_step;
signal_samples = (1/periode_ech).*ifft(spectre_pour_inversion);

% Visualiser les echantillons du signal reconstruit (prendre leur
 partie reelle "real(signal_samples)" )
```

```
figure(5)
plot(freq_pour_inversion,real(signal_samples));
xlabel('x');
ylabel('y');
title('echantillons signal reconstruit ');
% utiliser la fonction "scatter" pour visualiser les echantillons
 real(signal_samples)
hold on
scatter(freq_pour_inversion,real(signal_samples),'filled');
```



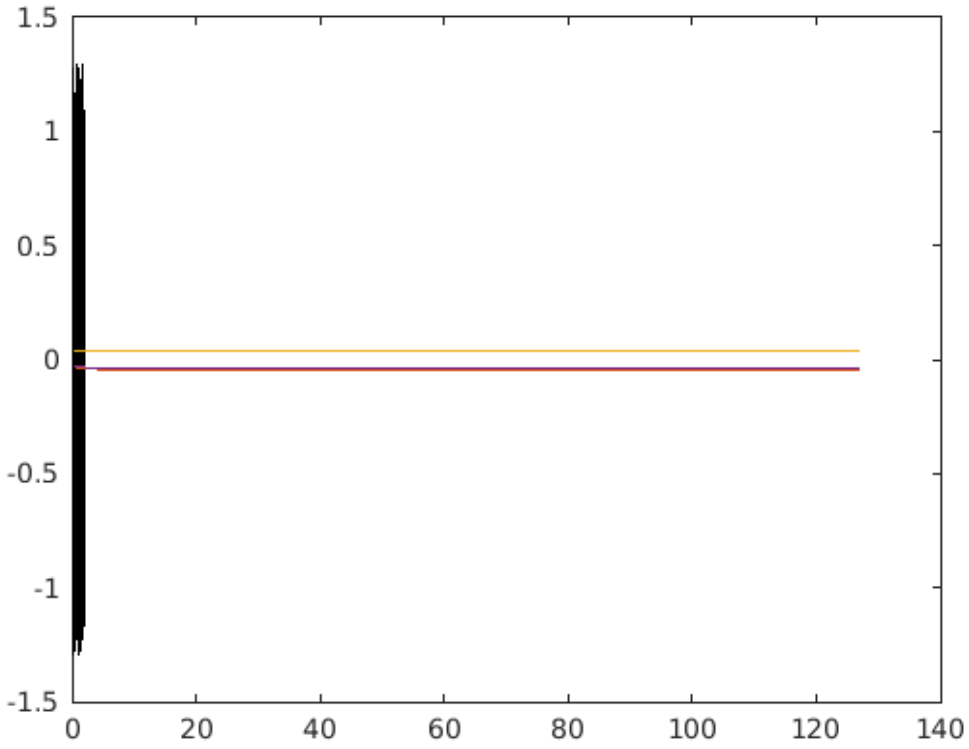
Question 5

```
% plot du signal originaire
plot(t_interv,signal,'k');
% calcul du signal interpole avec un developpement de sinc
signal2=0.*freq_pour_inversion;
for n=1:length(signal_samples)
    signal2=signal2 + real(signal_samples(n)).*sinc(pi*periode_ech./
freq_pour_inversion-n*pi);
end
% plot du signal interpole
hold on;
plot(freq_pour_inversion,signal2);
```



Question 6

```
figure(6)
% Plot du signal originaire
plot(t_interv,signal,'k');
% Plot des premiers trois termes du developpement en sinc
hold on;
x1=real(signal_samples(1)).*sinc(pi.*periode_ech./freq_pour_inversion-
pi);
plot(freq_pour_inversion,x1);
x2=real(signal_samples(2)).*sinc(pi.*periode_ech./freq_pour_inversion-
pi*2);
plot(freq_pour_inversion,x2);
x3=real(signal_samples(3)).*sinc(pi.*periode_ech./freq_pour_inversion-
pi*3);
plot(freq_pour_inversion,x3);

% on refais l'exercice avec une periode_ech>1/4 qui ne respecte pas le
% theoreme de shannon


% 4.1.2
```



Exercice 4.1 echantillonnage ideal d'un signal

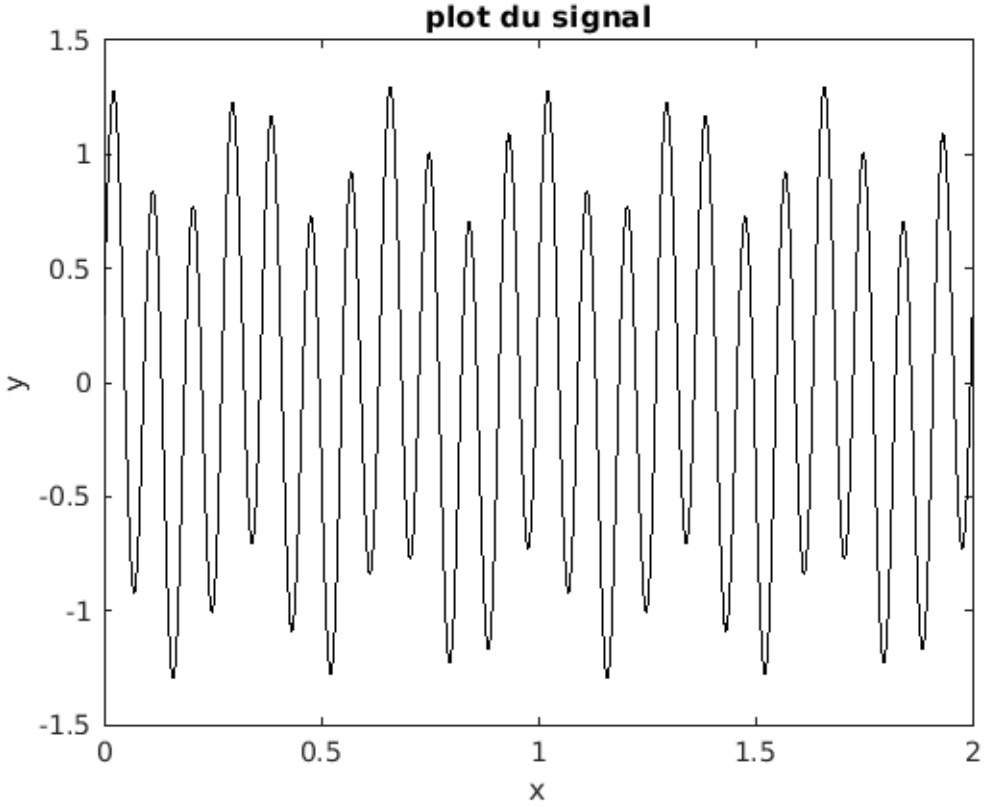
```
T = 2;  % Duree du signal
time_step = T/5000;
t_interv=0:time_step:T;
nu_1=11; nu_2=3;
fun = @(t_interv) sin(2*pi*nu_1*t_interv)+0.3*cos(2*pi*nu_2*t_interv);
signal = fun(t_interv);
% d'apres le theoreme de shannon, pour une reconstruction parfaite, il
 faut
% chosir un pas d'echantillonnage T<1/2*B soit T<1/4 et T>1/4 pour ne
 pas
% reconstruire le signal.
```

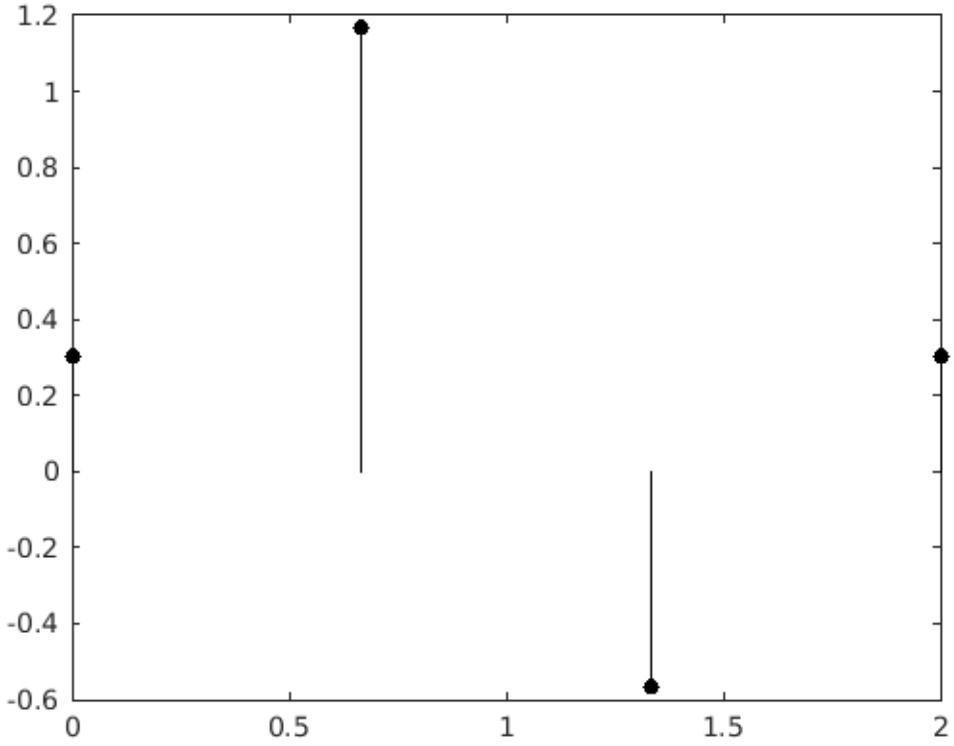
Exercices 4.1.1 et 4.1.2 Signal bien echantillonne et sa reconstruction / Signal sous-echantillonne et sa reconstruction

Question 1

```
figure(7)
% plot du signal
```

```
periode_ech = 1;
x1= t_interv;
y1=signal;
plot(x1,y1,'k');
xlabel('x');
ylabel('y');
title('plot du signal');

% parametres pour l'echantillonnage (a ne pas modifier)
no_ech = T/periode_ech+1;                  % nombre d'echantillons
no_ech = 2^nextpow2(no_ech);               % nombre d'echantillons
 arrondi a la premiere puissance de 2 (plus simple pour calculer sa
 transformee)
periode_ech = T/(no_ech-1);                % periode d'echantillonage
 ajoute
echantillons = 0:periode_ech:T;            % instants d'echantillonage
signal_echantillone = fun(echantillons);   % signal d'echantillon

% Visualisation signal echantillonns
figure(8)
x1=echantillons;
y1=zeros(1,length(x1));
x2=echantillons;
y2=signal_echantillone;
plot([x1;x2],[y1;y2],'k');
hold on;
scatter(x2,y2,'k','filled');
```


plot du signal
1.5
1
0.5
0
-0.5
-1
-1.5
y
0
0.5
1
1.5
2
x


1.2
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
0
0.5
1
1.5
2

Question 2

```
spectre = fft(echantillons);
% Definition de l'axe des frequences (ne pas modifier)
freq_step=1/periode_ech/no_ech;
freq = -1/periode_ech+freq_step:freq_step:1/periode_ech-freq_step;
spectre_echantillonne(1:no_ech) = spectre;            % Definition du
 spectre pour frequences negatives (pas fait par Matlab)
spectre_echantillonne(no_ech:1:2*no_ech-1) = spectre; % Definition du
 spectre pour frequences negatives (pas fait par Matlab)
figure(9);
% plot du module du "spectre_echantillonne"

plot(freq,spectre_echantillonne);
xlabel('x');
ylabel('y');
title('spectre-echantillonne');

Error using plot
Vectors must be the same length.

Error in TP4 (line 195)
plot(freq,spectre_echantillonne);
```

Question 3

```
% Filtrage de la transformee de Fourier du signal echantillons
 (necessaire avant de reconstruire le signal)
sigma=1/periode_ech;
fun1 = @(t_interv)  porte(t_interv,-sigma/2,sigma/2);
f1=fun1(periode_ech);
spectre_filtre = spectre_echantillonne.*f1;
figure(10);
%  on supprime les repliques avec un #ltre passe-bas ideale de largeur
 1 T et amplitude T.
% plot du module du "spectre_filtre"
plot(freq,spectre_filtre);
xlabel('x');
ylabel('y');
title('spectre-filtre');
```

Question 4

```
% Inversion de la transformee de Fourier filtree (ne pas modifier)
spectre_pour_inversion(1:0.5*no_ech)        =
 spectre_filtre(no_ech:no_ech+0.5*no_ech-1);
spectre_pour_inversion(0.5*no_ech+1:no_ech) =
 spectre_filtre(no_ech-0.5*no_ech+1:no_ech);
freq_pour_inversion=0:freq_step:1/periode_ech-freq_step;
signal_samples = (1/periode_ech).*ifft(spectre_pour_inversion);
```

```
% Visualiser les echantillons du signal reconstruit (prendre leur
 partie reelle "real(signal_samples)" )
figure(11)
plot(freq_pour_inversion,real(signal_samples));
xlabel('x');
ylabel('y');
title('echantillons signal reconstruit ');
% utiliser la fonction "scatter" pour visualiser les echantillons
 real(signal_samples)
hold on
scatter(freq_pour_inversion,real(signal_samples),'filled');
```

Question 5

```
% plot du signal originaire
plot(t_interv,signal,'k');
% calcul du signal interpole avec un developpement de sinc
signal2=0.*freq_pour_inversion;
for n=1:length(signal_samples)
    signal2=signal2 + real(signal_samples(n)).*sinc(pi*periode_ech./
freq_pour_inversion-n*pi);
end
% plot du signal interpole
hold on;
plot(freq_pour_inversion,signal2);
```

Question 6

```
figure(12)
% Plot du signal originaire
plot(t_interv,signal,'k');
% Plot des premiers trois termes du developpement en sinc
hold on;
x1=real(signal_samples(1)).*sinc(pi.*periode_ech./freq_pour_inversion-
pi);
plot(freq_pour_inversion,x1);
x2=real(signal_samples(2)).*sinc(pi.*periode_ech./freq_pour_inversion-
pi*2);
plot(freq_pour_inversion,x2);
x3=real(signal_samples(3)).*sinc(pi.*periode_ech./freq_pour_inversion-
pi*3);
plot(freq_pour_inversion,x3);

% on refais l'exercice avec une periode_ech>1/4 qui ne respecte pas le
% theoreme de shannon
```

Published with MATLAB® R2020a