Rapport du TP2

1. Objectifs:

 Comprendre comment manipuler un signal audio avec Matlab, en effectuant certaines opérations classiques sur un fichier audio d'une phrase enregistrée via un smartphone.

 Comprendre la notion des sons purs à travers la synthèse et l'analyse spectrale d'une gamme de musique.

Commentaires :

Il est à remarquer que ce TP traite en principe des signaux continus.Or, l'utilisation de Matlab suppose l'échantillonnage du signal. Il faudra donc être vigilant par rapport aux différences de traitement entre le temps continu et le temps discret.

Tracé des figures :

toutes les figures devront être tracées avec les axes et les légendes des axes appropriés.

Travail demandé :

un script Matlab commenté contenant le travail réalisé et des commentaires sur ce que vous avez compris et pas compris, ou sur ce qui vous a semblé intéressant ou pas, bref tout commentaire pertinent sur le TP.

2. Jeux de mots:

« phrase.wave » est un fichier audio enregistré à l'aide d'un smartphone, en prononçant lentement la phrase :

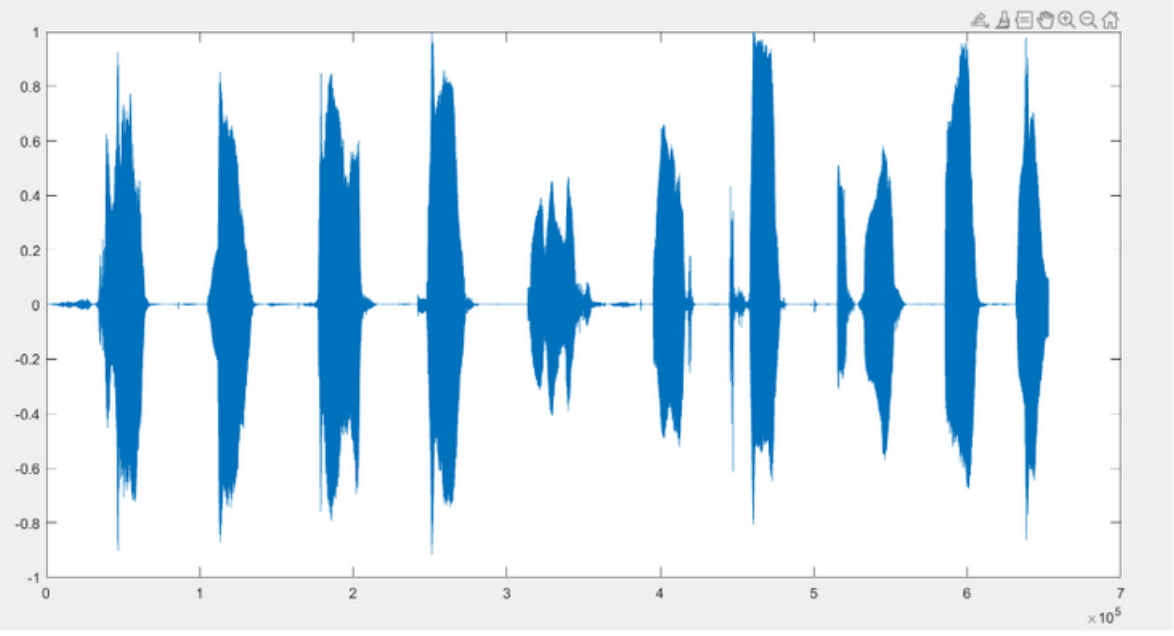
- « Rien ne sert de courir, il faut partir à point »

1- Sauvegardez ce fichier sur votre répertoire de travail, puis charger-le dans MATLAB à l'aide de la commande « audioread ».

```
[a,f] = audioread("phrase.wave");
```

2- Tracez le signal enregistré en fonction du temps, puis écoutez-le en utilisant la commande « sound ».

```
N=length(a)
ts=1/f
t=(0:N-1)*ts;
plot(t,a)
```

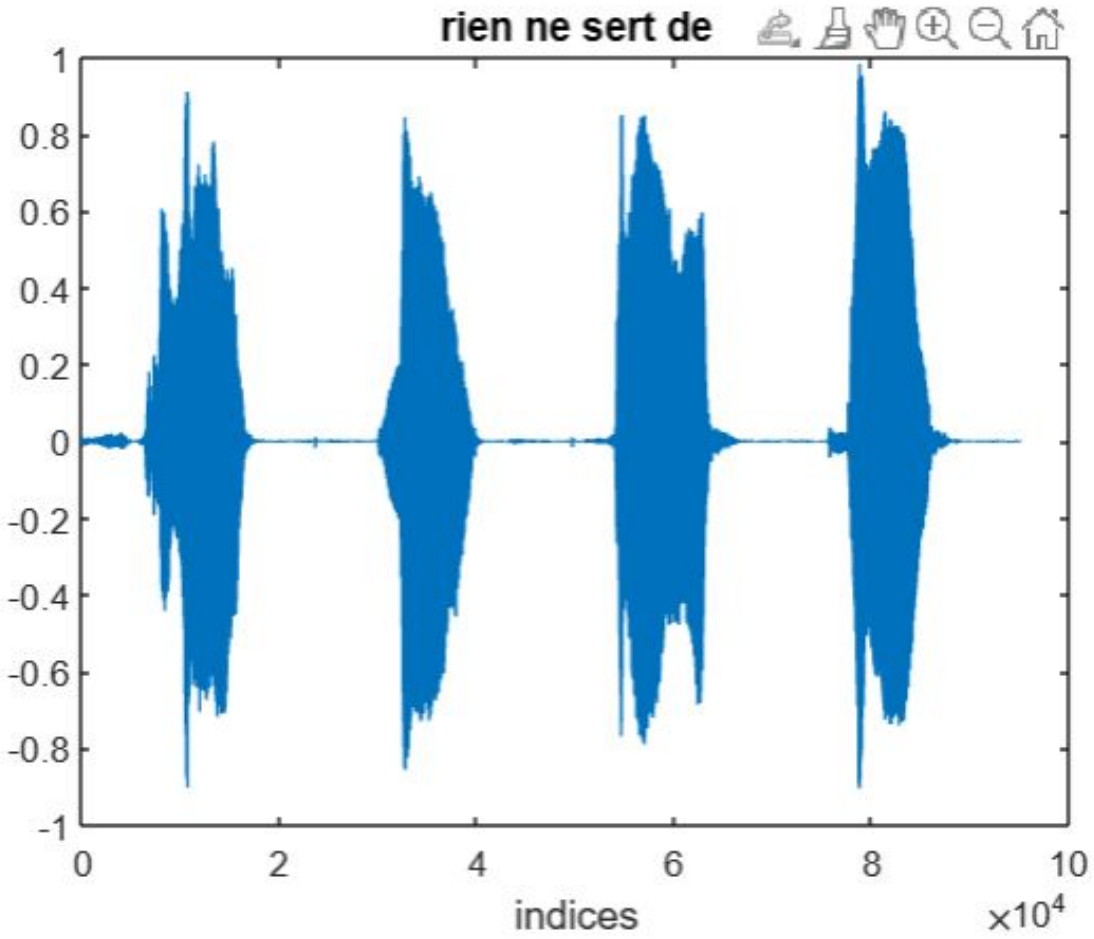


3- Cette commande permet d'écouter la phrase à sa fréquence d'échantillonnage d'enregistrement. Ecoutez la phrase en modifiant la fréquence d'échantillonnage à double ou deux fois plus petite pour vous faire parler comme « Terminator » ou « Donald Duck ». En effet, modifier la fréquence d'échantillonnage revient à appliquer un changement d'échelle y(t) = x(at) en fonction de la valeur du facteur d'échelle, cela revient à opérer une compression ou une dilatation du spectre initial d'où la version plus grave ou plus aigüe du signal écouté.

```
sound(a,f*2) % Donald Duck (audio compressé)

sound(a,f*0.5) %Terminator (audio dilaté)
```

4- Tracez le signal en fonction des indices du vecteur x, puis essayez de repérer les indices de début et de fin de la phrase « Rien ne sert de ».

```
rien_ne_sert_de = a(5055:100000);
plot(rien_ne_sert_de);
title('Rien ne sert de');
```



5- Pour segmenter le premier mot, il faut par exemple créer un vecteur « riennesertde » contenant les n première valeurs du signal enregistré x qui correspondent à ce morceau. Créez ce vecteur, puis écoutez le mot segmenté.

```
riennesertde = a(5055:100000);
sound(riennesertde,f)
```

6- Segmentez cette fois-ci toute la phrase en créant les variables suivantes :riennesertde, courir, ilfaut, partirapoint.

```
rien_ne_sert_de_indice_debut = 5055;
rien_ne_sert_de_indice_fin = 100000;
rien_ne_sert_de=a(rien_ne_sert_de_indice_debut:rien_ne_sert_de_indice_fin);
sound(rien_ne_sert_de,f)


% Trouver les indices du vecteur "courir" dans le signal
courir_indice_debut = 100000;
courir_indice_fin = 130000;
courir=a(courir_indice_debut:courir_indice_fin);
 sound(courir,f)

% Trouver les indices du vecteur "il faut" dans le signal
il_faut_indice_debut = 130000;
il_faut_indice_fin = 170000;
 il_faut=a(il_faut_indice_debut:il_faut_indice_fin);
sound(il_faut,f)

% Trouver les indices du vecteur "partir a point" dans le signal
partir_a_point_indice_debut = 176000;
partir_a_point_indice_fin = 219000;
partir_a_point=a(partir_a_point_indice_debut:partir_a_point_indice_fin);
sound(partir_a_point,f)
```

7- Notez que le signal initial de parole est un vecteur colonne contenant un certain nombre de valeurs (length(x)). Réarrangez ce vecteur pour écouter la phrase synthétisée « Rien ne sert de partir à point, il faut courir ».

```
vecteur = [rien_ne_sert_de ; partir_a_point ; il_faut ; courir];
sound(vecteur,f)
```

(Revenir au sommaire)

3. Synthèse et analyse spectrale d'une gamme de musique.:

- Synthèse d'une gamme de musique

Les notes de musique produites par un piano peuvent être synthétisées approximativement numériquement. En effet, chaque note peut être considérée comme étant un son pur produit par un signal sinusoïdal. La fréquence de la note « La » est par exemple de 440 Hz.

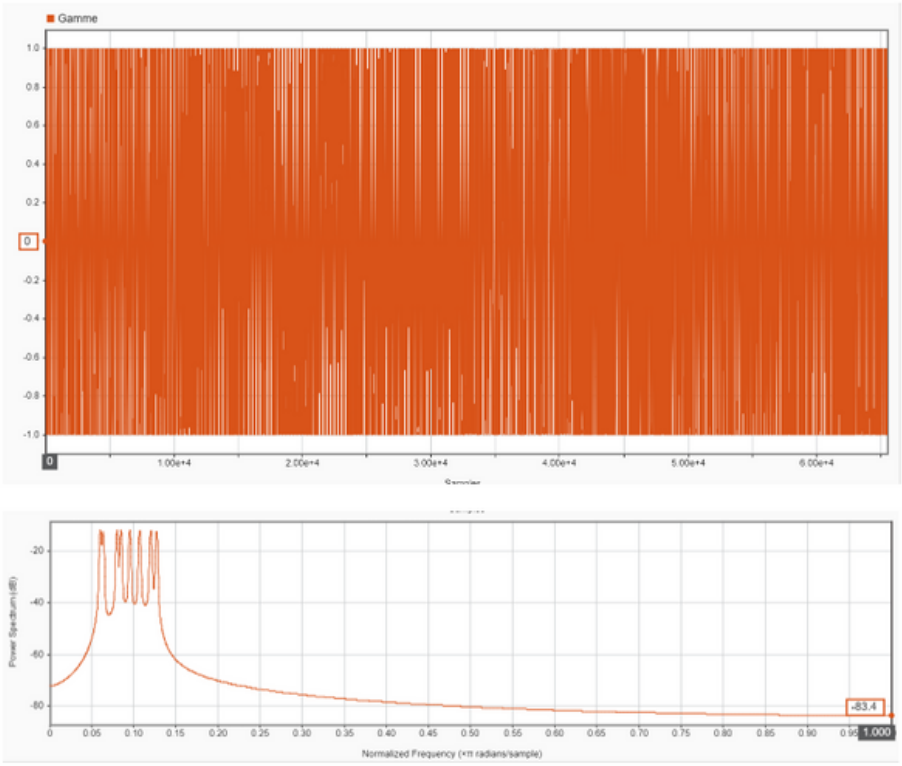
1- Créez un programme qui permet de jouer une gamme de musique. La fréquence de chaque note est précisée dans le tableau ci-dessous. Chaque note aura une durée de 1s. La durée de la gamme sera donc de 8s. La fréquence d'échantillonnage fe sera fixée à 8192 Hz.

```
fe = 8192;
 te = 1/fe
 t = 0:te:1;
do1 = sin(2*pi*262*t);
 re = sin(2*pi*249*t);
 mi = sin(2*pi*330*t);
 fa = sin(2*pi*349*t);
sol = sin(2*pi*392*t);
 la = sin(2*pi*440*t);
 si = sin(2*pi*494*t);
do2 = sin(2*pi*523*t);
 % 1 note
note1 = input('Donnez la premiére note : \n 1:DO \n 2:RE \n 3:MI \n 4:FA \n 5:SOL \n 6:LA \n 7:SI \n 8:DO(2) \n');
 if note1 == 1
  A = do1
elseif note1 == 2
  A = re
elseif note1 == 3
  A = mi
elseif note1 == 4
  A = fa
elseif note1 == 5
  A = sol
elseif note1 == 6
  A = la
elseif note1 == 7
  A = si
elseif note1 == 8
  A = do2
 end


 % 2 note
note2 = input('Donnez la deuxième note : \n 1:DO \n 2:RE \n 3:MI \n 4:FA \n 5:SOL \n 6:LA \n 7:SI \n 8:DO(2) \n');
 if note2 == 1
  B = do1
elseif note2 == 2
  B = re
elseif note2 == 3
  B = mi
elseif note2 == 4
  B = fa
elseif note2 == 5
  B = sol
elseif note2 == 6
  B = la
elseif note2 == 7
  B = si
elseif note2 == 8
  B = do2
 end

 % 3 note
note3 = input('Donnez la troisième note : \n 1:DO \n 2:RE \n 3:MI \n 4:FA \n 5:SOL \n 6:LA \n 7:SI \n 8:DO(2) \n');
 if note3 == 1
  C = do1
elseif note3 == 2
  C = re
elseif note3 == 3
  C = mi
elseif note3 == 4
  C = fa
elseif note3 == 5
  C = sol
elseif note3 == 6
  C = la
elseif note3 == 7
  C = si
elseif note3 == 8
  C = do2
 end


 % 4 note
note4 = input('Donnez la quatrième note : \n 1:DO \n 2:RE \n 3:MI \n 4:FA \n 5:SOL \n 6:LA \n 7:SI \n 8:DO(2) \n');
 if note4 == 1
  D = do1
elseif note4 == 2
  D = re
elseif note4 == 3
  D = mi
elseif note4 == 4
  D = fa
elseif note4 == 5
  D = sol
elseif note4 == 6
  D = la
elseif note4 == 7
  D = si
elseif note4 == 8
  D = do2
 end
```

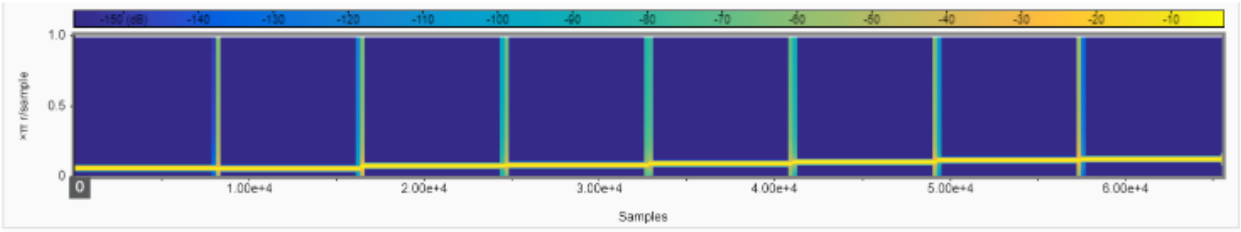
```
% 5 note
note5 = input('Donnez la cinquième note : \n 1:DO \n 2:RE \n 3:MI \n 4:FA \n 5:SOL \n 6:LA \n 7:SI \n 8:DO(2) \n');
if note5 == 1
    E = do1
elseif note5 == 2
    E = re
elseif note5 == 3
    E = mi
elseif note5 == 4
    E = fa
elseif note5 == 5
    E = sol
elseif note5 == 6
    E = la
elseif note5 == 7
    E = si
elseif note5 == 8
    E = do2
end

% 6 note
note6 = input('Donnez la sixième note : \n 1:DO \n 2:RE \n 3:MI \n 4:FA \n 5:SOL \n 6:LA \n 7:SI \n 8:DO(2) \n');
if note6 == 1
    F = do1
elseif note6 == 2
    F = re
elseif note6 == 3
    F = mi
elseif note6 == 4
    F = fa
elseif note6 == 5
    F = sol
elseif note6 == 6
    F = la
elseif note6 == 7
    F = si
elseif note6 == 8
    F = do2
end

% 7 note
note7 = input('Donnez la septième note : \n 1:DO \n 2:RE \n 3:MI \n 4:FA \n 5:SOL \n 6:LA \n 7:SI \n 8:DO(2) \n');
if note7 == 1
    G = do1
elseif note7 == 2
    G = re
elseif note7 == 3
    G = mi
elseif note7 == 4
    G = fa
elseif note7 == 5
    G = sol
elseif note7 == 6
    G = la
elseif note7 == 7
    G = si
elseif note7 == 8
    G = do2
end

% 8 note
note8 = input('Donnez la huitième note : \n 1:DO \n 2:RE \n 3:MI \n 4:FA \n 5:SOL \n 6:LA \n 7:SI \n 8:DO(2) \n');
if note8 == 1
    H = do1
elseif note8 == 2
    H = re
elseif note8 == 3
    H = mi
elseif note8 == 4
    H = fa
elseif note8 == 5
    H = sol
elseif note8 == 6
    H = la
elseif note8 == 7
    H = si
elseif note8 == 8
    H = do2
end
%la gamme
Gamme = [ A B C D E F G H ];
sound(Gamme,fe)
```

Spectre de la gamme de musique

2- Utilisez l'outil graphique d'analyse de signaux signalAnalyzer pour visualiser le spectre de votre gamme. Observez les 8 fréquences contenues dans la gamme et vérifiez leur valeur numérique à l'aide des curseurs.



3- Tracez le spectrogramme qui permet de visualiser le contenu fréquentiel du signal au cours du temps (comme le fait une partition de musique) mais la précision sur l'axe des fréquences n'est pas suffisante pour relever précisément les 8 fréquences.

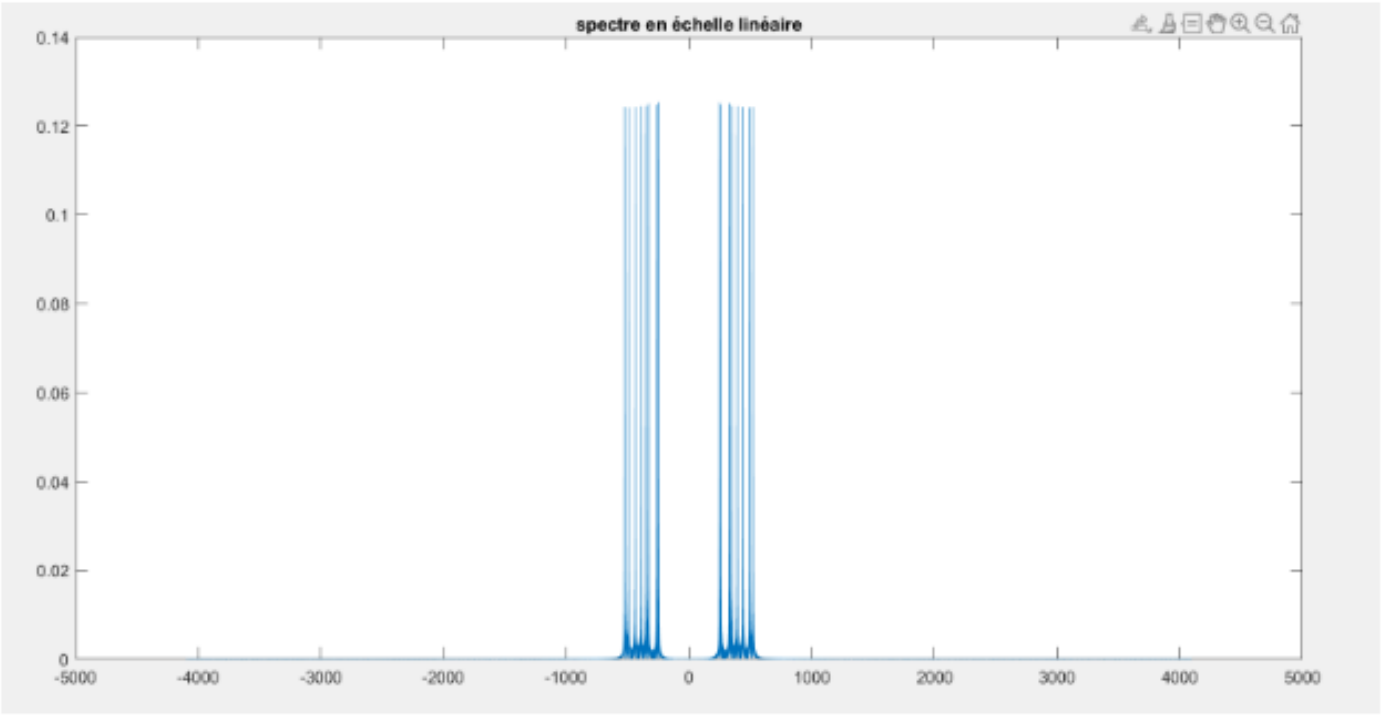


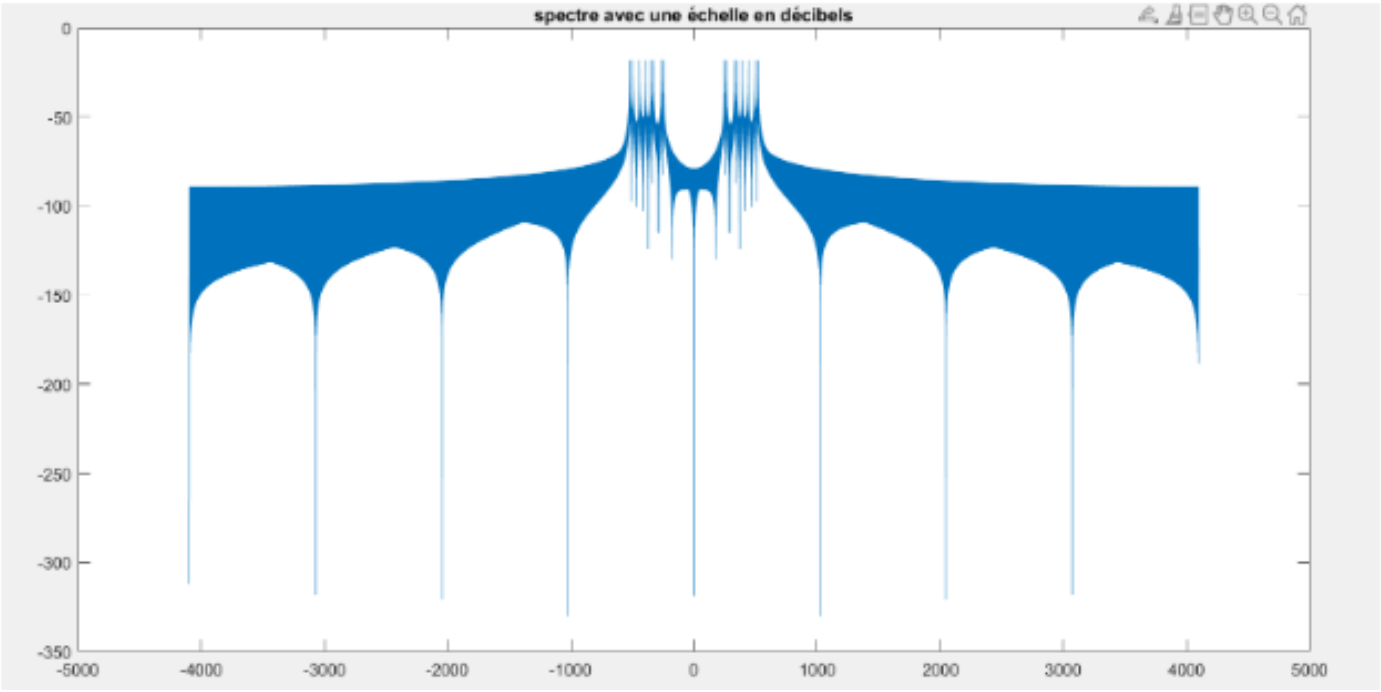
Approximation du spectre d'un signal sinusoïdal à temps continu par FFT

4- Le spectre d'un signal à temps continu peut être approché par transformée de Fourier discrète (TFD) ou sa version rapide (Fast Fourier Transform (FFT). Afficher le spectre de fréquence de la gamme musicale crée en échelle linéaire, puis avec une échelle en décibels.

$$S(dB) = 20 \times \log_{10}|S(f)|$$

```
N = length(Gamme);
fshift = (-N/2:N/2-1)*(fe/N);
%spectre en échelle linéaire
plot(fshift,fftshift(2*abs(fft(Gamme))/N));
title('spectre en échelle linéaire');

%spectre avec une échelle en décibels

plot(fshift,20*log10(fftshift(2*abs(fft(Gamme))/N)));
title('spectre avec une échelle en décibels');
```


spectre en échelle linéaire
0.14
0.12
0.1
0.08
0.06
0.04
0.02
0
-5000
-4000
-3000
-2000
-1000
0
1000
2000
3000
4000
5000


spectre avec une échelle en décibels
0
-50
-100
-150
-200
-250
-300
-350
-5000
-4000
-3000
-2000
-1000
0
1000
2000
3000
4000
5000