TP n°7 Filtrage numérique

PSI 2022/2023

On s'intéresse dans ce document au filtrage numérique de l'enregistrement de deux notes jouées successivement au piano (Ré de l'octave six (2349 Hz)1 puis Do de l'octave deux (131 Hz)). Nous étudierons plus particulièrement l'influence de la fréquence d'échantillonnage.

Le signal sonore, d'une durée de 3 s, est, dans un premier temps, échantillonné à la fréquence $f_e = 44,1$ kHz. Le filtre numérique utilisé est synthétisé à partir d'un filtre analogique passe bas du premier ordre via la méthode d'Euler.

L'objectif est d'atténuer la note la plus aigüe (Ré) sans modifier significativement la note la plus grave (Do) ainsi que de déterminer si le filtre se comporte comme attendu pour des fréquences d'échantillonnages plus faibles.

A Introduction au filtrage numérique sur un exemple simple

Avant de s'intéresser au filtrage numérique de l'enregistrement audio, nous nous proposons d'étudier le filtrage d'un signal élémentaire, de durée $\Delta t = 0,2$ s, se décomposant en un signal sinusoïdal de fréquence $f = 30$ Hz et d'amplitude $A = 1$ V, superposé à un "bruit" de plus haute fréquence f_b ($f_b > f$) et d'amplitude $A' = 0,6$V :

$$e(t) = A\sin(2\pi f t) + A_b \sin(2\pi f_b t)$$

Le signal $e(t)$ est alors échantillonné à la fréquence $f_e = 2000$ Hz et les valeurs des N_e échantillons sont données par la relation :

$$\forall k \in [0, N_e - 1] e[k] = A\sin(2\pi f k T_e) + A_b \sin(2\pi f_b k T_e)$$

avec $T_e = \frac{1}{f_e}$.

1. Quel est le nombre N_e d'échantillons nécessaires ?
2. A l'aide de la fonction `linspace` du module `numpy`, on peut créer un tel signal :

```
import numpy as np
import matplotlib.pyplot as plt

A = 1
Ab = 0.6
f = 30
fb = 270
f_e = 2000
```

```
N_e = ......................
t = np.linspace(...........)
e = A*np.sin(2*pi*f*t)+Ab*np.sin(2*pi*fb*t)
plt.plot(t,e,'b.')
plt.xlabel('temps(s)')
plt.ylabel('e(t)')
```

3. On considère la fonction `spectre(F,Deltat)` qui affiche le spectre d'une fonction échantillonnée `F` sur une durée totale `Delta t` :

```
def spectre(F,Deltat):
  Ne = len(F)
  coeffComplex = np.fft.fft(F)
  coeffComplex2 = np.concatenate((coeffComplex[0]/Ne,coeffComplex[1:]/(2*
   Ne),axis=None)
  coeffReel = np.absolute(coeffComplex2)

  freq = np.array([i/Deltat for i in range(Ne)])

  plt.plot(freq,coeffReel,'b')
  plt.xlabel('frequence (Hz)')
  plt.ylabel('spectre du signal')
```

4. Tracer le spectre du signal d'entrée. Commenter.

   Afin de synthétiser le filtre numérique, considérons l'équation différentielle correspondant à un filtre analogique passe bas du premier ordre :

$$e(t) = s(t) + \frac{1}{2\pi f_c}\frac{\mathrm{d}s}{\mathrm{d}t}(t)$$

   $s(t)$ et $e(t)$ étant respectivement les signaux en entrée et sortie du filtre et f_c sa fréquence de coupure.

5. L'équation précédente peut s'écrire :

$$e(t) = s(t) + \frac{1}{2\pi f_c}\frac{s(t+\mathrm{d}t) - s(t)}{\mathrm{d}t}(t)$$

   En utilisant la méthode d'Euler explicite, en déduire la relation de récurrence vérifée par le signal de sortie échantillonné `s[k]`. *Pour les plus rapides, faire des recherches sur le schéma d'Euler implicite et le coder.*

6. Ecrire une fonction `filtrePB(e,fc)` qui prend en argument une fréquence de coupure `fc` et un signal échantillonné `e` et qui renvoie le signal de sortie échantillonné.
7. Dans la suite, on choisit $f_c = 40$ Hz. Tracer le signal en sortie du filtre numérique ainsi que son spectre. Commenter.
8. Reprendre les questions précédentes avec $f_b = 1990$ Hz. Commenter.

B Filtrage numérique d'un enregistrement sonore

On s'intéresse désormais au filtrage de l'enregistrement3 de deux notes jouées successivement au piano (Ré de l'octave six (2349 Hz) puis Do de l'octave deux (131 Hz)) et échantillonné à la fréquence $f_e = 44,1$ kHz (Le critère de Shannon est ici largement respecté).

1. Télécharger le fichier "piano.wav" sur le site cahier de prépas. L'écouter. A l'aide des lignes ci-dessous, tracer l'allure du signal ainsi que son spectre. Reconnaître les harmoniques remarquables.

```
import scipy.io.wavfile as wav

# Attention a bien specifier le chemin d'acces au fichier notes_piano.wav
#dans la ligne suivante.
 #fe_origine est la frequence d'echantillonnage (ici 44100 Hz)
fe_origine, signal_entree_origine = wav.read('notes_piano.wav')
```

2. Filtrer le signal avec le filtre numérique passe-bas en prenant une fréquence de coupure à 720 Hz. Tracer l'allure du signal de sortie ainsi que son spectre. L'enregistrer sous format .wav et l'écouter. Commenter. On pourra utiliser :

```
wav.write('notes_piano_filtre.wav',fe,signal_sortie)
```

3. Rééchantillonner maintenant le signal d'entrée à la fréquence $f_e = 2450$ Hz. Tracer l'allure du signal obtenu. Enregistrer la version rééchantillonnée sous format .wav et l'écouter. Filtrer ce signal avec le filtre précédent *on préfèrera utiliser la méthode d'Euler implicite dans ce cas pour plus de précision.* Analyser le signal de sortie. Commenter.