Numérisation et sur-échantillonnage

1. Introduction

La numérisation d'un signal doit se faire en respectant la condition de Nyquist-Shannon. Le spectre du signal ne doit pas s'étendre au delà de la moitié de la fréquence d'échantillonnage. Pour obtenir cette condition, il y a deux méthodes :

- ▷ Utilisation d'un filtre analogique anti-repliement.
- ▷ Méthode du sur-échantillonnage.

La méthode du sur-échantillonnage consiste à échantillonner le signal à une fréquence beaucoup plus grande que la largeur de bande utile, de manière à échantillonner correctement les hautes fréquences présentes dans le signal. On fait subir au signal numérique un filtrage passe-bas avant de réduire sa fréquence d'échantillonnage pour le stockage ou la transmission.

La méthode du sur-échantillonnage est aujourd'hui privilégiée, car les filtres anti-repliement analogiques sont difficiles à réaliser et d'une efficacité limitée.

Un exemple est la numérisation du son pour l'enregistrement haute-fidélité sur CD. Les sons audibles sont dans la bande de fréquence allant de 50 Hz à 20 kHz, mais certaines harmoniques peuvent se trouver au delà de 20 kHz. Pour éviter le repliement de spectre, le son est échantillonné à 196 kHz, avant de subir un filtrage puis une réduction à 44 kHz, fréquence à laquelle il est stocké.

Ces TP ont pour objectif de mettre en place un sur-échantillonnage et le traitement du signal associé (filtrage et réduction). On testera le système sur un signal périodique dont le spectre est très étendu, un signal créneau, avant de le mettre en application sur un signal audio.

2. Matériel et outils numériques

Matériel utilisé :

- ▷ Générateur de signaux (GBF).
- ▷ Oscilloscope.
- ▷ Carte d'acquisition SysamSP5.
- ▷ Ordinateur et environnement Python.
- ▷ Logiciel PureData.
- ▷ Câble de liaison pour la sortie son.

Outils numériques :

- ▷ analogiqueNumerique.py : numérisation d'un signal avec la carte Sysam SP5.
- ▷ traitementSignal.py : fonctions de traitement du signal (analyse spectrale, calcul de filtres, filtrage, réduction).
- ▷ lectureWAV.pd : fichier pureData pour la lecture d'un fichier audio WAV.
- ▷ acquisitionSansFinFiltrage.py : acquisition et filtrage en temps réel avec tracé du signal.

Les deux premiers scripts contiennent des fonctions à utiliser. Il faut les enregistrer dans le dossier personnel de travail. Pour utiliser les fonctions dans un script python, faire les importations suivantes :

```
from analogiqueNumerique import *
from traitementSignal import *
```

3. Réalisation du système de numérisation et de traitement

Voici les conditions que l'on impose au système :

- ▷ Fréquence d'échantillonnage de la numérisation fixée à $f_e = 20\ kHz$.
- ▷ Fréquence d'échantillonnage finale (après filtrage et réduction) égale à $4\ kHz$.

Toutes les observations et les choix effectués devront être reportés dans le compte-rendu écrit. À la fin de chaque partie, un compte rendu oral devra être fait à l'examinateur.

3.a. Étude du signal de test

Pour tester le système, on utilise un signal créneau, dont le spectre est très riche en harmonique. On choisira la fréquence maximale de ce signal pour que le repliement de spectre soit négligeable.

Faire l'acquisition et l'étude spectrale pour différentes fréquences. Observer le repliement du spectre. Choisir une fréquence pour la suite de l'étude.

3.b. Conception du filtre passe-bas

Expliquer le rôle de ce filtre.
Choisir une fréquence de coupure pour ce filtre, en justifiant.
Concevoir un filtre RIF et étudier sa réponse fréquentielle. On rappelle que la sélectivité du filtre est d'autant plus grande que l'indice de troncature est grand.

3.c. Filtrage et réduction

Effectuer le filtrage du signal créneau numérisé.
Effectuer la réduction pour arriver à la fréquence d'échantillonnage finale.

3.d. Test de qualité

Ce test peut être fait avec un signal en créneau, mais aussi avec un signal sinusoïdal.

Vérifier que le système final n'est pas sujet au repliement de spectre, ou tout au moins que le repliement est très faible. Augmenter si nécessaire la sélectivité du filtre.

4. Numérisation d'un son

La bande son à traiter est disponible sous forme d'un fichier WAV, accessible dans l'espace d'échange. Le son est émis par le convertisseur N/A audio de l'ordinateur (sortie casque). Le patch PureData lectureWAV.pd permet de faire la lecture du fichier. La sortie audio du PC sera reliée à la voie EA0 de la carte Sysam SP5.

Pour lancer la lecture du son dans PureData, activer le DSP, cliquer sur `open fichier.wav` puis sur `start`.

Dans un premier temps, on fera une analyse du son en faisant des acquisitions de durée 1 seconde.

> Faire plusieurs acquisitions et analyses spectrales. Quelle est la bande spectrale approximative occupée par le son de cet enregistrement ?

On utilise le système de numérisation et de réduction mis au point précédemment pour numériser et stocker cette bande son sans trop altérer sa qualité.

> Le choix de la fréquence d'échantillonnage finale ($4\ kHz$) est-il convenable ? Tester le traitement pour des paquets de 1 seconde.

Pour finir, on pourra tester le traitement en flux continu avec le script acquisitionSansFinFiltrage.py, qui permet de visualiser en temps réel le résultat du filtrage et de la réduction. La représentation temporellle du signal est tracée, mais on pourra modifier le script pour qu'il affiche le spectre calculé sur la fenêtre d'analyse.

5. Annexe : filtrage numérique

Soit un signal numérique constitué par la suite de nombres x_n, échantillonnés à la fréquence f_e. D'une manière générale, un filtre linéaire est défini par la relation de récurrence suivante :

$$y_n = \sum_{k=0}^{N-1} b_k x_{n-k} - \sum_{k=1}^{M-1} a_k y_{n-k} \tag{1}$$

La suite y_n est le signal de sortie du filtre. Il est donc défini par M coefficients a_k et N coefficients b_k. Par convention, on a toujours $a_0 = 1$. Lorsque tous les autres a_k sont nuls, le filtre a une réponse impulsionnelle finie (filtre RIF) ; les coefficients b_k définissent alors la réponse impulsionnelle. Lorsque certains coefficients a_k sont non nuls (pour $k > 0$), le filtre est dit *récursif* car la sortie à l'instant n dépend de manière récursive de la sortie aux instants antérieurs. Un tel filtre a en général une réponse impulsionnelle infinie (filtre RII).

Selon l'application visée et la sélectivité souhaitée, on utilisera un filtre RIF ou un filtre RII. Les filtres RIF peuvent être très sélectifs avec un grand nombre de coefficients. Les filtres RII sont plus efficaces lorsque le nombre de coefficients doit être petit (inférieur à 10), mais ils peuvent être instables.

La réponse fréquentielle d'un filtre (gain et déphasage) est obtenue avec sa fonction de transfert en Z :

$$H(z) = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2} + \cdots}{1 + a_1 z^{-1} + a_2 z^{-2} + \cdots} \tag{2}$$

en posant :

$$z = e^{j2\pi \frac{f}{f_e}} \tag{3}$$

Le rapport f/f_e doit être inférieur à $1/2$ (condition de Nyquist-Shannon).

Le fichier traitementSignal.py contient deux fonctions permettant de calculer le gain et le déphasage :

- ▷ `(f,g,phi)=reponse_freq_filtre(a,b)`
- ▷ `(f,log_f,g_db,phi)=reponse_freq_filtre_bode(a,b,fmin)`

Ces deux fonctions prennent en argument les listes des coefficients sous forme de tableaux numpy. La fréquence calculée est en fait le rapport de la fréquence réelle sur la fréquence d'échantillonnage. La première fonction répartit les fréquences de manière linéaire sur l'intervalle $[0, 1/2]$. La seconde les répartit de manière logarithmique, c'est pourquoi il faut lui fournir la fréquence minimale `fmin`.

Le calcul de la sortie par la relation (1) nécessite de fixer les valeurs initiales des $M-1$ premières valeurs de y_n. On choisit généralement des valeurs initiales nulles.

Le calcul directe par la relation (1) est appelé *réalisation directe de type 1*. Une autre manière de calculer la sortie est d'utiliser un signal intermédiaire w_n défini par :

$$w_n = x_n - a_1 w_{n-1} - a_2 w_{n-2} - \cdots - a_{M-1} w_{M-1} \tag{4}$$

La sortie est alors :

$$y_n = b_0 w_n + b_1 w_1 + b_2 w_2 + \cdots + b_{N-1} w_{N-1} \tag{5}$$

On montre que ces deux relations sont équivalentes à la relation (1) (en régime stationnaire). La condition initiale consiste à fixer les $M-1$ premières valeurs de w_k, en général à zéro. Ces deux relations définissent la *réalisation directe de type 2*. La fonction `filtrage` effectue le calcul par cette méthode.