TD/TP3

Filtres linéaires

Le but de ce court [1] TD/TP est de construire quelques filtres linéaires avec des fréquences de coupure données, et de les tester sur des signaux de synthèse.

Pour générer des signaux, on utilisera le code suivant [2] :

Listing 1 – Fichier `gensig.m`

```
function [x,f,s]=gensig(N,Fe,F,A)
Te=1/Fe;
t=((1:N)/Fe)';
x=sin(2*pi*t*F(:)')*A(:);
f=(0:N/2-1)*Fe/N;
y=fft(x,N)/N;
s=2*abs(y(1:(N/2)));
```

Ce code génère un échantillon de N valeurs correspondant à une somme de sinusoïdes échantillonnées à une fréquence F_e. Chaque sinusoïde est caractérisée par sa fréquence F et son amplitude A :

$$x(t) = \sum_{k=1}^{n} A_k sin(2\pi F_k t) = \sum_{k=1}^{n} A_k sin(\omega_k t)$$

en notant $\omega_k = 2\pi F_k$ et donc le signal échantillonné vaut

$$X(n) = x(kT_e) = \sum_{k=1}^{n} A_k sin(\omega_k k T_e)$$

où $T_e = \frac{1}{F_e}$.

Par exemple, si on entre les commandes suivantes [3]

```
>> N=1024;
>> Fe=16000;
>> [x,f,s]=gensig(N,Fe,[1000,2000],[3;5]);
```

alors

```
>> figure(1),plot(x)
```

permet de tracer le signal échantillonné

$$x(t) = 3\sin(2\pi\, 1000t) + 5\sin(2\pi\, 2000t)$$

1. Court pour vous permettre de terminer le précédent et de préparer l'examen !
2. Code disponible sur http://monge.u-bourgogne.fr/ebusvelle/teaching.php
3. Ne pas taper les chevrons ">>" qui représentent l'invite Matlab

Le code calcule aussi la transformée de Fourier du signal généré (ce sont les trois dernières lignes de la fonction) et le spectre [4]. On trace ce dernier par

```
>> figure(2),plot(f,s,'-')
```

Question 1. Générer une somme de 6 sinusoïdes de même amplitude et de fréquences 2000 Hz, 2001 Hz, 2010 Hz, 2100 Hz, 3000 Hz et 5000 Hz échantillonnées à une fréquence $F_e = 16000$ Hz. Calculer la TF de ce signal avec $N = 1024$ points. Que voyez vous ? Augmentez N (par puissances de 2) jusqu'à distinguer les six fréquences du signal. Combien vaut N alors ? Est-ce cohérent avec la théorie ?

On veut concevoir un filtre numérique de type passe-bas qui permette de supprimer la fréquence 5000 Hz en préservant les autres.

Avec les fonctions Matlab, on génère les coefficients du filtre par

```
>> Fc=4000 ;
>> Rp=0.5 ;
>> [b,a]=cheby1(9,Rp,Fc/(Fe/2)) ;
```

que l'on applique au système avec

```
>> xf=filter(b,a,x) ;
```

Question 2. Calculer la TF de x_f et vérifier que le but est atteint.

Question 3. Que représentent les commandes

```
>> [h,ff]=freqz(b,a,N/2,Fe) ;
>> plot(ff,abs(h)),grid,
```

Question 4. Montrer que le filtre obtenu est stable (mots clés : `roots` et `abs`).

Question 5. Dans ce cas comme dans les deux suivants (IIR et FIR "maison"), essayer de changer la fréquence de coupure pour éliminer une à une les hautes fréquences.

La méthode utilisée par Matlab est issue de la théorie des filtres linéaires (filtres de Chebyshev, de Butterworth,...). On peut se débrouiller sans, avec ce qu'on a vu en cours et en TD, et qui constitue la base de la synthèse des filtres.

Question 6. Proposer la construction d'un filtre à réponse impulsionnelle infinie, à partir du filtre passe-bas continu de fréquence de coupure adéquate. Pour cela, on peut, comme en cours et en TD, partir du filtre continu, de transformée de Laplace

$$H(s) = \frac{\omega_0}{\omega_0 + s}$$

et calculer le filtre passe-bas numérique correspondant, soit par la transformation linéaire $s = \frac{1}{T_e}(z-1)$, soit à l'aide de la transformation bilinéaire.

Tester, tracer la réponse fréquentielle, la stabilité.

Question 7. Proposer la construction d'un filtre à réponse impulsionnelle finie à partir de spécifications de type "gabarit" [5]. Mêmes questions.

4. Le facteur 2 dans $s_j = 2|y_j|$ est un artifice pour retrouver la bonne valeur des amplitudes, du fait que l'on travail sur des signaux réels.

5. La méthode est dans le cours, comme d'habitude