Travaux pratiques

Ces travaux pratiques ont pour objectif d'analyser les différent types de filtres et de réaliser le filtrage d'un signal audio. Le but est de savoir manipuler les fonctions disponibles sous Matlab pour synthétiser un filtre satisfaisant un gabarit imposé, l'appliquer au filtrage d'un signal et enfin vérifier son bon fonctionnement.
Le compte-rendu de TP est constitué des réponses aux questions posées, vos commentaires sur les résultats obtenus ainsi que du script Matlab qui sera placé en annexe. Ce compte-rendu doit être transmis sous format pdf à `said.moussaoui@ec-nantes.fr`*, au plus tard deux jours après la dernière séance du TP.*

1 Filtrage d'une gamme audio

Les notes musicales d'un piano peuvent être synthétisées numériquement. Dans une version simplifiée, chaque note est produite par un signal sinusoïdal dont la fréquence est précisée dans le tableau ci-dessous. Une fonction `gamme.m` permettant de générer une gamme de musique est fournie sur le serveur pédagogique.

Note	Do	Ré	Mi	Fa	Sol	La	Si	Do
Fréquence (Hz)	262	294	330	349	392	440	494	523

Question préliminaire. Générer puis écouter une gamme d'une durée de $1s$ par note avec une fréquence d'échantillonnage $f_e = 8192$ Hz.

1.1 Filtrage passe-bas analogique

On souhaite utiliser des filtres analogiques pour éliminer les trois dernières notes de la gamme. On retiendra les spécifications suivantes ($A_p = 3$ dB, $A_a = 40$ dB, $f_c = 420$ Hz et $\Delta f = 100$ Hz). Afin de comparer les quatre types de filtres (Butterworth, Chabyshev de type 1 ou 2 et Cauer).

1. Calculer à l'aide de Matlab l'ordre de chaque filtre. Quel filtre vous semble avantageux ?
2. Tracer sur le même graphique (se servir de `legend`) les réponses fréquentielles en amplitude (exprimée en décibels) des quatre filtres. Commenter les différences entre ces réponses.
3. Appliquer les filtres au signal audio synthétique. Ecouter le signal après filtrage. A t-on éliminé toutes les notes souhaitées ?
4. Tracer sur la même figure (se servir de `subplot`) le signal filtré et son spectre d'amplitude. On se limitera à l'intervalle de fréquences de 250 Hz à 550 Hz. Commenter les résultats.
5. Refaire la même analyse et commenter les résultats pour les spécifications suivantes :
   a) réduction de la largeur de la bande de transition : ($A_p = 3$ dB et $\Delta f = 20$ Hz).
   b) réduction des ondulations en bande atténuée : ($A_p = 1$ dB et $\Delta f = 20$ Hz).

1.2 Elimination d'une note par filtrage coupe-bande

On souhaite éliminer une seule note de la gamme musicale et on utilisera des filtres numériques RII.

1. Synthétiser un filtre numérique coupe-bande de type Chebyshev 2 qui élimine la note **FA** de la gamme musicale. On retiendra les spécifications suivantes ($f_c^b = 340$ Hz, $f_c^h = 360$ Hz, $\Delta f_b = \Delta f_h = 10$ Hz, $A_p = 1$ dB, $A_a = 40$ dB)
2. Tracer sur la même figure l'allure temporelle et le spectre d'amplitude de la gamme filtrée. On se limitera à la bande spectrale comprise entre 250 et 550 Hz,

3. Réaliser les mêmes opérations avec un filtre RIF de longueur minimale (fenêtre de Kaiser) en utilisant la fonction `fir1`.
4. Tracer sur le même graphique les réponses fréquentielles des filtres RII et RIF. Commentaires.

2 Quelques indications pour la programmation sous Matlab

Ecrire un seul script pour traiter toutes les questions liées à chaque section du TP. Ci-dessous quelques indications sur les fonctions Matlab utiles. Il vous est recommandé de faire appel à l'aide détaillée de Matlab sur la syntaxe d'utilisation de chaque fonction

```
>> help nomdelafonction
```

Génération et analyse du signal

- Génération du signal de la gamme musicale

```
>> duree = 1; fe=8192;
>> [sig,t] = gamme(duree, fe);
>> soundsc(sig, fe)
```

- Analyse fréquentielle d'un signal : `fft, plot, axis.`

```
>> f = [0:length(sig)-1]*(fe/length(sig));
>> S = fft(sig)/fe; plot(f, abs(S)); axis([fmin fmax 0 ymax])
```

Filtrage analogique ou numérique RII

- Ordre nécessaire d'un filtre : `buttord, cheb1ord, cheb2ord, ellipord`.

```
>> [nc wnc]= buttord(2*pi*fp,2*pi*fa,Ap,Aa,'s'); % en analogique
>> [nd wnd]= buttord(2*fp/fe,2*fa/fe,Ap,Aa); % en discret
>> % fp et fa auront deux composantes pour un passe/coupe-bande
```

- Calcul des coefficients de la fonction de transfert : `butter, cheby1, cheby2` ou `ellip`.

```
>> [numc, dend] = butter(nc,wnc,'low','s');% en analogique
>> [numd, dend] = butter(nd,wnd,'low'); % en discret
```

- Calcul de la réponse fréquentielle d'un filtre : `freqs, freqz`

```
>> f = linspace(fmin,fmax,1000);
>> Hcont = freqs(numc,denc,2*pi*f); plot(f,20*log10(abs(Hc))
>> Hdisc = freqz(numd,dend,f,fe); plot(f,20*log10(abs(Hd))
```

- Filtrage d'un signal, `sig` évalué aux instants définis dans un vecteur `t` :

```
>> sigf = lsim(tf(numc,denc),sig,t);
>> sigf = filter(numd,dend,sig);
```

Filtrage numérique RIF

- Calcul de l'ordre de la fenêtre de pondération

```
>> [n,wn,beta] = kaiserord(fb,amp,dev,fe)
```

  - `fb` : fréquences caractéristiques des bandes passantes et atténuées par ordre croissant.
  - `amp` : gain du filtre dans chaque bande (échelle linéaire).
  - `dev` : amplitude des ondulations (δ_a, ϵ_p) dans chaque bande. Pour un fIltre coupe-bande

```
>> [n,wc,beta] = kaiserord([fpb fab fah fph],[1 0 1],[ep da
ep],fe);
```

- Calcul des coefficients d'un filtre RIF par troncature de la réponse impulsionnelle du filtre idéal.

```
>> h = fir1(n,wc,ftype,fenetre)
```

  - `ftype` : chaîne de caractères précisant le type de filtre (`low`, `high`, `bandpass`, `stop`).
  - `fenetre` : vecteur contenant les $n+1$ coefficients de la fenêtre de pondération `bartlett, blackman, rectwin, hamming, hann, kaiser`.