TP DE TRAITEMENT NUMÉRIQUE DU SIGNAL: IMPLÉMENTATION TEMPS-RÉEL D'UN EQUALIZER TROIS BANDES

Table des matières

1. Objectifs

— Comprendre le cours !
— Savoir implémenter les filtres idéaux
— Savoir implémenter les synthèses de filtres FIR par fenêtrage
— Créer un equalizer 3 bandes (ou plus) temps-réel sous matlab

2. Filtres idéaux

2.1. Implémentation "batch" (non temps réel).

(1) Écrire une fonction

```
function [s_filtre] = passe_bas_ideal(s,fe,fc)
```

qui implémente le filtrage passe-bas idéal, telle qu'on ai
— Entrée
  — `s` : signal d'entré.
  — `fe` : frequence d'échantillonnage
  — `fc` : frequence de coupure du signal
— Sortie :
  — `ts_filtre` : signal filtré

(2) La tester avec le script suivant :

```
% Lecture du fichier audio
[signal, fe] = audioread('Met.wav');

% Conversion stereo to mono
signal = 0.5*signal(:,1) + 0.5*signal(:,2);

% Ecoute du signal
sound(signal,fe);

% Filtrage ideal du signal

fc = 500; % Choix de la frequence de coupure en Hz
signal_PB = passe_bas_ideal(s,fe,fc);

% Ecoute du signal filtre
sound(signal_PB,fe);
```

(3) Écrire et tester les fonctions
- `[s_filtre] = passe_haut_ideal(s,fe,fc)`
- `[s_filtre] = passe_bande_ideal(s,fe,fc1,fc2)`

2.2. **Implémentation temps réel.** On rappelle la structure d'une boucle temps-réel pour un signal audio

```
% Lecture d'un fichier audio:
AFR = dsp.AudioFileReader('fichier.wav','SamplesPerFrame',frameLength);

% Ecriture d'un fichier audio:
ADW = dsp.AudioPlayer(AFR.SampleRate);

while ~isDone(AFR)
  % Lecture d'un block de signal
  audio = step(AFR);

  % Traitement de ce block
  audio_processed = signal_processing(audio);

  % Ecriture du block de signal
  step(ADW, audio_processed)
end

% Fermeture des ressources
close(AFR);
close(ADW);
```

(1) Implémenter un filtrage passe-bas par bloc en temps-réel.

(2) Que remarquez-vous ?

3. Filtrage FIR en temps-réel

(1) Implémenter un filtre FIR passe-bas, par méthode de fenêtrage. Pour cela :

(a) Écrire une fonction

`[FIRcoeff] = coeff_passe_bas_fir(fe,fc,ordre,fenetre,gainDB,gainMax)`

qui calcule les coefficients du filtres intervenant dans l'équation aux différences, avec
— Entrée :

    - `s` : signal d'entré.
    - `fe` : frequence d'échantillonnage
    - `fc` : frequence de coupure du filtre
    - `ordre` : l'ordre du filtre
    - `fenetre` : forme de la fenetre
    - `gainDB` : gain en dB
    - `gainMax` : le gain Max en DB
  - Sortie :
    - `FIRcoeff` : les coefficients du filtre

(b) Écrire une fonction

```
[ sig_filtered ] = realTime_filtering(sig_pad,frameLength,FIRcoeff)
```

qui implémente le filtrage (la convolution) en temps réel. Avec
- Entrée :
  - `sig_pad` : un vecteur contenant le bloc du signal a traitée, ainsi que le bloc de signal précédent
  - `frameLength` : la taille d'un bloc de signal
  - `FIRcoeff` : les coefficients du filtre
- Sortie :
  - `sig_filtered` : le signal filtré

4. Equalizer trois bandes

A partir des fichiers `EQ3bandes_TP.m` et `EQ3bandes_TP.fig`, implémenter un equalizer trois bandes en temps-réel. On rappelle les bandes de fréquences utiles :
- sub-basses : 30 a 63 Hz
- basses : 63 a 250 Hz
- bas-mediums : 250 a 500 Hz
- mediums : 500 a 2000 Hz
- haut-mediums : 2 a 4 kHz
- aigus : 4 a 20 kHz