INSA
Département Télécommunications, Services & Usages
3 TC

TD5 - SIGNAUX & SYSTÈMES

Synthétiseur de musique

Exercice n° 1

Le principe de base est le suivant : chaque note jouée sur un synthétiseur est émise pendant une durée D ; elle correspond à l'émission d'une sinusoïde à une fréquence Fi dont l'enveloppe est modulée par une fonction pour lui donner un certain timbre que l'on peut associer à tel ou tel instrument.

Ci-dessous les fréquences correspondantes aux différentes notes de deux octaves.

Note	Do3	Ré3	Mi3	Fa3	Sol3	La3	Si3
	C3	D3	E3	F3	G3	A3	B3
Frequence	261.6Hz	293.7Hz	329.6Hz	349.6Hz	391.9Hz	440.0Hz	493.9Hz
Note	Do4	Ré4	Mi4	Fa4	Sol4	La4	Si4
	C4	D4	E4	F4	G4	A4	B4
Frequence	523.2Hz	587.4Hz	659.2Hz	699.2Hz	783.8Hz	880.0Hz	987.8Hz

Sur la base de ces principes, nous allons sous Matlab générer un morceau de musique.

Nous travaillerons à la fréquence d'échantillonnage Fs=8 kHz. Nous émettrons toutes les notes pendant un temps fixe D=0,5 s.

1) Combien d'échantillons y a-t-il par note ?

Nous allons commencer par émettre une note « pure » comme étant l'émission d'une sinusoïde à la fréquence Fi pendant une durée D.

2) Si nous étions en analogique, donnez l'expression de *Note(t)* en fonction de *Rect(t)*. En déduire l'expression de la transformée de Fourier *Note(f)* et représentez son module.
3) Nous allons passer sur un signal échantillonné à la fréquence *Fs=1/Te* de durée finie égale à D et commençant à 0. Donnez l'expression de *Note(kTe)*. Représentez le module du spectre de cette note. Donnez l'expression du signal discret *Note[n]* que nous manipulerons dans Matlab.

Retrouvez ces résultats avec quelques commandes sous Matlab

```
clear;
D=0.5;          % Durée d'émission d'une note
Fs=8000;        % Fréquence d'échantillonnage
Te = 1/Fs;      % Période d'échantillonnage
N = round(D/Te);% Nombre d'échantillons dans une note
n = Te*(0:N-1); % Vecteur temps pour une note
octave  = [ 261.6, 293.7, 329.6, 349.6, 391.9, 440.0, 493.9 ]; % octave
freqs = [octave 2*octave]'; % Vecteur de fréquences pour 2 octaves

% Jouons les notes des deux octaves
gamme = [];
for k = 1 : length(freqs)
    note(k,:) = sin(2*pi*freqs(k)*n);
    gamme = [gamme note(k,:)];
end
soundsc(gamme,Fs);

s3=note(3,:); % signal Mi (329,6 Hz)
f4=[0:3999]./4000*Fs; % Création d'un vecteur de fréquence de 4000 points allant de 0 à Fs
f256=[0:255999]./256000*Fs; % Création d'un vecteur de fréquence de 256000 points allant de 0
à Fs

tiledlayout(2,1)
nexttile; plot(f4,abs(fft(s3,4000))); % Affichage du module du spectre du Mi
nexttile; plot(f256,abs(fft(s3,256000)));;
```

4) Expliquer pourquoi les deux spectres affichés sont différents. Lequel est le plus « juste » ?

Pour avoir une note plus riche, nous allons modifier l'enveloppe de la note par une fonction enveloppe (Env) définie de 0 à T.

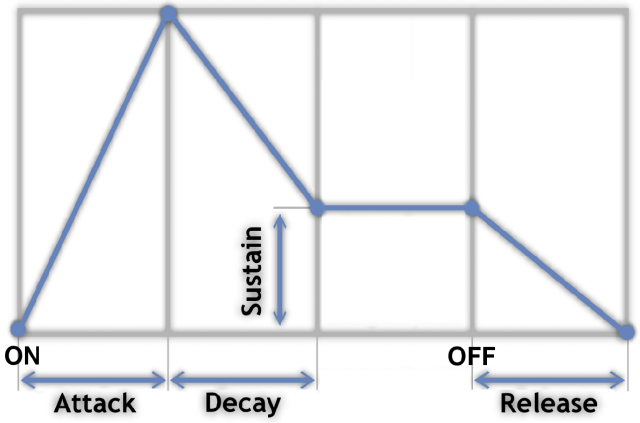
5) Donnez l'expression de la note modifiée par cette fonction enveloppe *Note_m[n]*. Quel sera la forme du spectre de la note modifié ?

Un premier exemple sous Matlab avec une enveloppe exponentielle

```
Env = exp(-n/(D/4));
Note_m = s3.*Env;
soundsc(Note_m,Fs);
figure;
tiledlayout(4,1)
nexttile; plot(n,Env);
nexttile; plot(n,Note_m);
nexttile; plot(f4,abs(fft(Note_m)));
f4c=([0:3999]-2000)./4000*Fs;
nexttile; plot(f4c,fftshift(abs(fft(Note_m))));
```

On analysera entre autres les deux dernières lignes.

En pratique les fonctions enveloppe utilisées sont plutôt du type ADSR telle que représentée ci-dessous.



Forme de l'enveloppe ADSR avec les contraintes suivantes :
Tattack < Tdecay < Trelease <1 et 0 < HeightSustain <1

Vous pouvez récupérer le fichier `SIS-TD5-Synthetiseur.mlx` sous Moodle. Les fichiers avec l'extension mlx sont des fichiers dit « live script » pour Matlab que vous pouvez exécuter par section tout en observant les variables, figures, équations …
Dans ce fichier, nous avons défini des fonctions pour créer une enveloppe ADSR que l'on applique sur les notes.

6) A vous de prendre en main ce fichier et de lui ajouter des affichages de spectres et de différents signaux.