Rapport de projet du système à la fonction

Synthèse d'un banc de filtres numériques

Suite à un problème de périphérique, et à un manque de temps, nous avons perdus nos documents / images de la dernière séance et également les images de la première séance (soucis matériel).

Il a été temporellement impossible d'aller régénérer les images perdues, ou les notes. Malheureusement, certaines informations sont à regret manquantes.

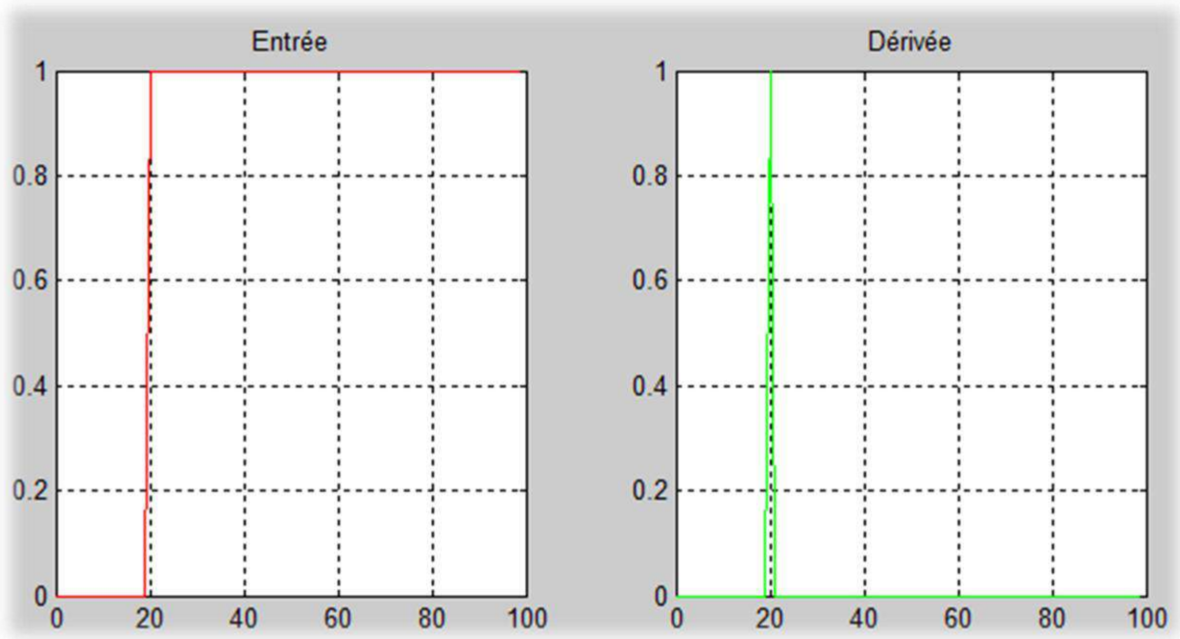
Table des matières

Programmation sous Matlab

1) Modélisation et calculs sous Matlab

Après avoir ouvert le logiciel de modélisation mathématique (Matlab), on implante directement le fichier algorithmique nécessaire à la réalisation du projet : **L1derivativ.m**

En effet, le but de cet exercice est d'arriver à modéliser le calcul d'une dérivée, à l'aide du programme fourni. Ainsi, en exécutant le code par le logiciel, on obtient donc cette courbe :



Comme on peut le remarquer, la courbe rouge (représentant la fonction à laquelle on s'intéresse) vaut globalement 1 : elle représente l'entrée. La courbe verte cependant représente la sortie du calcul, et celle-ci vaut 0. Donc la transformation passe la fonction de valeur 1 à 0 : le programme de Matlab a bien dérivé la fonction. La raison du fait que la courbe atteint sa valeur finale à partir de 20 est due à l'instruction temporelle **t=20**, en début de programme.

Ensuite, nous allons nous intéresser à la manière de programmer cette fonction. L'énoncé nous demande de programmer cette même fonction avec la directive suivante :

e = [zeros(1,t0) ones(1,N-t0)]

On peut décortiquer cette instruction :

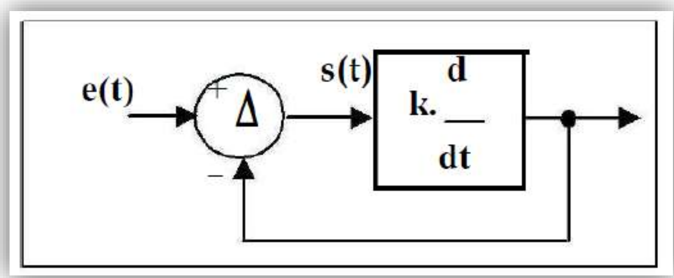
- **Zeros (1 , t0) :** Le rôle de cette partie est de mettre des zéros jusqu'en t0 (défini à 20).
- **Ones (1, N-t0) :** Similairement, l'instruction remplie de 1 la partie énoncée.

La suppression du point-virgule en fin de ligne est inattendue : en informatique, ce caractère est souvent synonyme de fin d'instruction. Cependant, en l'enlevant on observe de nouvelles écritures dans la consoles : les instructions et leurs résultantes ne sont plus cachées.
Le « ; » sert donc à masquer les instructions dont le résultat n'est pas nécessairement utile ou pertinent.

Programmation de filtres

Filtrage passe-bas du premier ordre

Nous allons désormais nous intéresser à la programmation d'un filtre passe-bas du premier ordre, sous Matlab. Pour cela, nous utiliserons une fonction mathématique émulant un filtre passe-bas. On peut trouver la représentation de cette fonction sur l'exemple de l'énoncé.



Représentation schématique de la fonction du filtre passe-bas

On cherche à montrer qu'un peut programmer un filtre passe bas de premier ordre au moyen d'une récurrence de la forme suivante : $\boldsymbol{s(t)} = \boldsymbol{\alpha}.\boldsymbol{e(t)} + \boldsymbol{\beta}.\boldsymbol{s(t-1)}$

D'après le montage, en mettant sous forme d'équation le cheminement et en arrangeant le tout, on peut arriver à une sortie s(t) de la forme :

$$s(t) = \frac{e(t)}{1+k} + \frac{k * s(t-1)}{1+k}$$

Ainsi, en isolant le e(t) et le s(t-1) de la première et seconde fraction, on obtient.

$$s(t) = \frac{1}{1+k} e(t) + \frac{k}{1+k} s(t-1)$$

Ainsi : $\boldsymbol{\alpha} = \frac{1}{1+k}$, $et\ \boldsymbol{\beta} = \frac{k}{1+k}$

Nous avons programmé la réponse indicielle pour 3 valeurs distingues de **k**

1. Images manquantes (voir page 2)
2. Images manquantes (voir page 2)
3. Images manquantes (voir page 2)

La relation entre **k** et la coupure du filtre se voit sur la réponse indicielle du filtre : **si le gain augmente, la pente sera plus faible. Une pente plus faible signifie une lenteur du filtre passe-bas**.

$$k = \frac{1}{2\pi f0}$$

Filtrage passe-haut du premier ordre

Nous allons désormais nous intéresser à la programmation d'un filtre passe-haut du premier ordre, sous Matlab de manière similaire à l'étude du filtre passe-bas de premier ordre.

Grâce aux calculs fait en séance, et à l'aide de nos professeurs, nous avons déterminés que le filtre passe-haut d'ordre 1 peut être modélisé de la manière suivante :

$$z(t) = e(t) - \left(\left(\frac{1}{1+k}\right) * e(t) + \left(\frac{k}{1+k}\right) * s(t-1)\right)$$

Ainsi, nous allons tester sa réponse indicielle, avec différentes valeurs de K :

1. *Images manquantes (voir page 2)*

Celui-ci semble adopter le même comportement qu'un filtre passe-bas de premier ordre, par rapport à **K**.

Filtre passe-bas du second ordre

Désormais, nous allons reprendre le filtre passe-bas, mais cette fois nous allons nous intéresser à un filtre passe-bas de second ordre. Un filtre passe-bas de premier ordre pourrait s'apparenter à une dérivée simple, dans le cas d'un filtre de second ordre on se sert du filtre de premier ordre pour faire une dérivée seconde.

On a vu dans la démarche précédente qu'un filtre passe-bas de premier ordre suit la formule suivante :

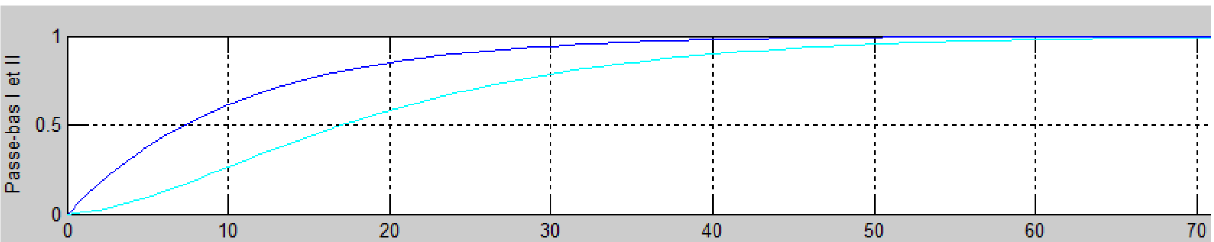
$$\boldsymbol{s(t) = \left(\frac{1}{1+k}\right) * e(t) + \left(\frac{k}{1+k}\right) * s(t-1)}$$

On appliquera donc à cette équation un système de « cascade » afin d'utiliser les entrées et les sorties dans le filtre passe-bas de second ordre. L'équation du filtre passe-bas du second ordre, obtenu via l'équation du filtre passe-bas de premier ordre, que l'on rentrera dans Matlab, est la suivante :

$$\boldsymbol{ss(t) = \left(\frac{1}{1+k}\right) * s(t) + \left(\frac{k}{1+k}\right) * ss(t-1)}$$

ss(t) symbolisant donc la sortie de l'équation du filtre passe-bas de second ordre

On analyse donc la réponse indicielle des deux filtres : passe bas de premier ordre et passe bas de second ordre, pour un gain k = 10.



- **Bleu foncé** : Courbe du filtre **passe-bas de 1er ordre**.
- **Bleu clair** : Courbe du filtre **passe-bas de 2nd ordre**.

On observe donc que la pente (qui symbole la rapidité du filtre) est plus rapide pour le passe-bas de 1er ordre que pour le passe-bas de 2nd ordre.

Filtre passe-bas du second ordre

On s'intéresse à un filtre passe-haut du 2nd ordre de la même manière que pour le passe-bas du 2nd ordre. Le filtre passe-haut du 1er ordre étant égal à

$$z(t) = e(t) - \left(\left(\frac{1}{1+k}\right) * e(t) + \left(\frac{k}{1+k}\right) * s(t-1)\right)$$

Par le même procédé de cascade, on obtient donc l'équation du filtre passe-haut de 2nd ordre en s'intéressant aux entrées/sorties à inclure :

$$zz(t) = z(t) - \left(\left(\frac{1}{1+k}\right) * z(t) + \left(\frac{k}{1+k}\right) * s(t-1)\right)$$

zz(t) symbolisant la sortie finale du filtre passe-haut de second ordre.

1. Images manquantes (voir page 2)

En conclusion : avec un gain de k=10, on se rend compte que son comportement est l'inverse des filtres passe-bas de 1er et 2nd ordre.

Filtre passe-bande

Nous cherchons désormais à créer un filtre passe-bande, qui est à la jointure du filtre passe-bas et du filtre passe-haut car sa combinaison nous permet de ne laisser passer qu'une certaine fréquence (ce qui est très pratique, dans le cas pratique, en télécommunication ou en réseau).

Pour cela, nous allons reproduire de manière presque similaire le phénomène de cascade vu précédemment, mais cette fois celle-ci sera entre un filtre passe-haut et un filtre passe bas.

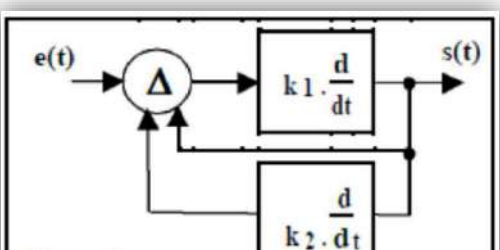


Figure 6 :
Filtrage passe-bande

Voici l'organisation d'un filtre passe bande : un enchévetrement entre un filtre passe-haut et un filtre passe bas.

D'après l'adaptation faite en séance, nous en avons déduis que son équation est la suivante :

$$\textbf{Y(t)} = \left(\frac{1}{1+k1+k1*k2}\right)*e(t)+\left(\frac{(k1+(k1*k2))}{1+k1+k1*k2}*y(t-1)-\left(\frac{k1*k2}{1+k1+k1*k2}\right)*y(t-2)\right)$$

Nous allons désormais nous intéresser à la réponse d'un tel filtre pour une réponse indicielle avec k1 = 0.5 etk2 = 20 :

- Images manquantes (voir page 2)
- Images manquantes (voir page 2)

On peut observer en effectuant différents tests que le **Q** influe sur la réponse indicielle : en augmentant Q (**le facteur qualité**), on diminue la **réponse indicielle**.

Réponse impulsionnelle et réponse harmonique

Observation des signaux et calculs des spectres

Nous allons maintenant nous intéresser à l'étude des signaux et des spectres sonnores : plus précisements des signaux audio. Pour commencer, nous allons ouvrir le programme fournit pour l'étude de cette séance.

En ouvrant le programme, on peut repérer quelques variables nécessaires pour les instructions, c'est les conditions initiales.

Par exemple, on peut repérer la fréquence qui correspond du coup à la fréquence de notre génération. Dans cet exemple, elle est égale à **44,1 khZ**, soit 44100 Hz pour une seconde.
Ce qui équivaut donc à **44100 échantillons** (soit **N = 44100**).

La condition de Shannon explique que l'on peut compresser sans réelle perte d'information si la fréquence d'échantillonnage est environ suppérieur à 2 fois les fréquences du spectre que l'on analyse.

Ainsi, la fréquence maximale fMAX des signaux que nous pouvons produire ou traiter est égale à la moitié du taux d'échantillonage.

$$fMAX = \frac{fe}{2} = 22\ 050\ Hz$$

(Partie manquante dû à une perte matérielle de dernière minute, impossible de régénérer l'ensemble des images et notes en séances de TD…)