Traitement Numérique du Signal
Session 2

Première année Département Sciences du Numérique

2020 – 2021

1 Introduction

1.1 Objectifs

Les objectifs de cette session de rattrapage sont de vérifier que vous êtes capables :

1. De générer un signal numérique simple et de le tracer avec une échelle temporelle en secondes.
2. D'estimer la représentation fréquentielle d'un signal en numérique et de la tracer avec une échelle fréquentielle en Hz.
3. De calculer la réponse impulsionnelle de filtres simples de type RIF (filtres à Réponse Impulsionnelle Finie).
4. De synthétiser des filtres RIF simples : choisir leurs paramètres pour satisfaire à un gabarit donné.
5. De filtrer un signal et d'expliquer les résultats obtenus.

1.2 Travail à rendre

Le travail devra être rendu en fin de séance :

1. Une copie contenant les calculs et explications demandés.
2. Les codes réalisés, permettant de visualiser les tracés demandés, déposés sur Moodle dans l'espace prévu à cet effet.

1.3 Consignes à respecter

1. Tous vos tracés doivent comporter des labels sur les axes (utiliser *xlabel.m* et *ylabel.m* sous matlab) et un titre (utiliser *title.m* sous matlab).
2. Si plusieurs courbes sont tracées sur la même figure, celle-ci devra comporter une légende (utiliser *legend.m* sous matlab).
3. Vos codes doivent être commentés de manière suffisante et claire. Un nouvel utilisateur doit pouvoir comprendre ce que vous avez souhaité implanter.

Remarque : il vous est demandé de réaliser plusieurs tracés. Ces tracés correspondent à ce qu'il est important de visualiser et de savoir analyser. Ces tracés, ainsi que les explications associées, seront évalués pour donner la note, mais, bien entendu, rien ne vous empêche de visualiser d'autres signaux/spectres si cela vous aide dans vos analyses.

2 Etude théorique

Dans le cas où plusieurs utilisateurs se partagent un même canal de propagation, une méthode de partage de la ressource doit être utilisée. Une possibilité est de laisser les utilisateurs transmettre en même temps mais pas dans les mêmes bandes de fréquence (on parle de multiplexage en fréquence ou FDMA en anglais pour Frequency Division Multiple Access). Dans ce cas, chaque utilisateur doit transporter son message dans la bande qui lui est allouée autour d'une certaine fréquence. Pour cela, il doit utiliser une technique de transposition de fréquence. Nous allons étudier la modulation d'amplitude : utilisation d'un cosinus porteur, de fréquence égale à la fréquence centrale de la bande allouée, qui portera le message à transmettre dans son amplitude. Le message à transmettre $m(t)$ sera considéré comme un signal déterministe à énergie finie occupant une bande de fréquence $[-b, b]$ autour de 0.

1. Modulation :
   Calculer la transformée de Fourier, $X(f)$, du signal modulé sur porteuse $x(t) = m(t)\cos(2\pi f_0 t)$ en fonction de la transformée de Fourier de $m(t)$, notée $M(f)$.
2. Démodulation :
   (a) Afin de retrouver le message $m(t)$ à partir du signal modulé $x(t)$, on va, dans un premier temps, multiplier $x(t)$ par le même cosinus que celui ayant servi à moduler, pour donner le signal suivant : $y(t) = x(t)\cos(2\pi f_0 t)$.
   Déterminer la transformée de Fourier, $Y(f)$, du signal $y(t)$ en fonction de $M(f)$.
   (b) La dernière opération à réaliser afin de retrouver le message émis $m(t)$ à partir de $y(t)$ est une opération de filtrage :
      i. Expliquer pourquoi on doit positionner un filtre sur le signal $y(t)$ afin de retrouver le message $m(t)$.
      ii. Quel type de filtre (passe-bas, passe-haut, passe-bande, réjecteur) doit-on utiliser ?
      iii. Ce filtre devra être implanté en numérique comme un filtre à réponse impulsonnelle finie (RIF). Calculer les élements $h_I(k)$ de la réponse impulsionnelle idéale échantillonnée du filtre à implanter. Vous pouvez vous aider pour cela de l'annexe qui rappelle la méthode de synthèse d'un filtre de type RIF. Donner le/les paramètre(s) à utiliser de manière à retrouver le message $m(t)$ en sortie.

3 Implantation

Pour l'implantation on considèrera une fréquence d'échantillonnage $F_e = 1000$ Hz, une fréquence de $f_0 = 110$ Hz et une amplitude de 1 (V) pour le cosinus porteur.

3.1 Génération et étude du message $m(t)$

1. Générer une information binaire (suite aléatoire de 0 et 1) à transmettre, en utilisant, par exemple, la fonction *randi.m* de Matlab : bits=randi([0, 1],1,Nb) ; où Nb représente le nombre d'éléments binaires à générer. Vous pouvez prendre, par exemple, $Nb = 1000$.
2. Créer le message à transmettre, $m(t)$, à partir de cette information binaire en codant les 0 et les 1 par des niveaux bas ou haut de durées $T_s = N_s T_e$, en prenant $N_s = 20$ échantillons par niveau (voir figure 1). Pour cela, à partir de la suite de bits générée précédemment, on pourra, par exemple, utiliser la fonction *kron.m* de Matlab de la manière suivante : m =kron($2 * bits - 1$,ones(1,Ns)) ;
3. En utilisant la fonction *plot.m* de Matlab, tracer le message généré avec une échelle temporelle en secondes (figure 1 de votre programme).

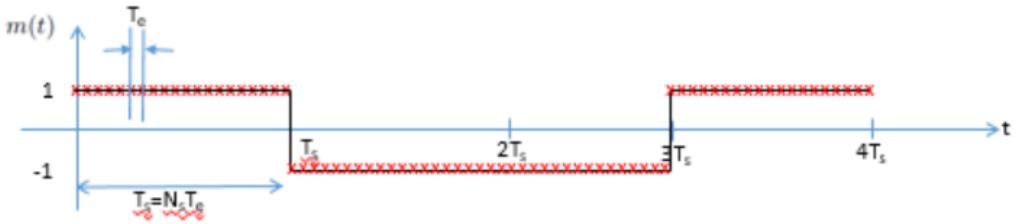

FIGURE 1 – Message à transmettre, generé avec $N_s = 20$ échantillons par niveau.

4. Estimer la transformée de Fourier du message grâce à la fonction *fft.m* de Matlab et tracer son module (fonction *abs.m*) avec une échelle de fréquence en Hz (figure 2 de votre programme).

3.2 Implantation de la transposition sur fréquence porteuse

1. Générer un nombre d'échantillon du cosinus porteur (fonction *cos.m* sous matlab) identique au nombre d'échantillons du message $m(t)$.
2. La transposition de fréquence sera réalisée ici en multipliant le message par le cosinus porteur (pour multiplier deux vecteurs entre eux sous Matlab on doit utiliser .*)
3. Tracer le module de la transformée de Fourier du signal modulé avec une échelle de fréquence en Hz (figure 3 de votre programme). Le tracé observé est-il conforme à l'expression théorique obtenue ? Expliquer votre réponse.

3.3 Implantation du retour à basse fréquence

1. Multiplier le signal modulé par le même cosinus (même phase, même fréquence) que celui ayant servi à réaliser la transposition sur fréquence porteuse.
2. Estimer la transformée de Fourier du signal obtenu après multiplication par le cosinus et tracer son module avec une échelle de fréquence en Hz (figure 4 de votre programme). Le tracé observé est-il conforme à l'expression théorique obtenue ? Expliquer votre réponse.
3. La dernière opération à réaliser afin de retrouver le message émis est une opération de filtrage.
   (a) Calculer sous Matlab, puis tracer (figure 5 de votre programme), "un morceau" ($k = -NT_e, ..., NT_e$) de la réponse impulsionnelle idéale déterminée dans la partie théorique en choisissant, dans un premier temps, N de manière arbitraire. La fonction sinus cardinal existe sous matlab (*sinc.m*), attention cependant $sinc(x)$ sous Matlab calcule $\frac{\sin(\pi x)}{\pi x}$.
   (b) Calculer la réponse en fréquence correspondant à la réponse impulsionnelle implantée, en utilisant la fonction *fft.m* de Matlab, et tracer son module avec une échelle fréquentielle en Hz (figure 6 de votre programme). Le filtre réalisée est-il bien de type passe-bas ?
   (c) Vérifier que le filtre synthétisé réalise bien la fonction souhaitée en superposant sur une même figure (figure 7 de votre programme) le module de la transformée de Fourier du signal en sortie du cosinus de réception et le module de la réponse en fréquence du filtre. Si ce n'est pas le cas modifier l'ordre et/ou la fenêtre de troncature du filtre.
   Remarque 1 : Afin de permettre une meilleure visualisation il peut être nécessaire de normaliser le module de la transformée de Fourier du signal en sortie du cosinus avant affichage : Ynormalise=(1/max(abs(Y)))*abs(Y) ; Remarque 2 : Pensez à utiliser du Zero Padding pour mieux visualiser les représentations fréquentielles, mais également pour obtenir le même nombre de points calculés en vue d'un tracé superposé.

(d) Enfin, réaliser le filtrage du signal en sortie du cosinus de réception, soit en utilisant la fonction *conv.m* de Matlab qui permet de réaliser le produit de convolution entre deux signaux, soit en utilisant la fonction *filter* de Matlab. Afin de ne pas avoir à gérer le retard introduit par le filtrage, on pourra utiliser le paramètre 'same' dans la fonction conv (voir help de Matlab) : conv(y,h,'same') ; si y représente le signal à filtrer et h la réponse impulsionnelle de votre filtre.

(e) Tracer sur une même figure (figure 8 de votre programme) le signal en sortie du filtre et le message transmis. Expliquer les différences constatées. Quels impacts auraient un changement d'ordre et un changement de fenêtre de troncature sur le signal retrouvé en sortie du filtre ?

4 Annexe : synthèse d'un filtre de type RIF

La synthèse de filtres RIF est assez intuitive et facile à mettre en oeuvre. Elle se compose des étapes suivantes :

1. On se donne une réponse en fréquences idéale, $H_I(f)$, du filtre à réaliser et un gabarit à respecter (limites autour de $H_I(f)$ dans lesquelles doit rester la réponse en fréquence du filtre réel, $H(f)$, qui sera non idéal). On travaille en numérique, on doit donc considérer que cette réponse en fréquence est périodique de période F_e (TF de la réponse impulsionnelle $h_I(t)$ qui doit être échantillonnée à T_e et tronquée). Elle est donc décomposable en série de Fourier :

$$H_I(f) = \sum_{k=-\infty}^{+\infty} h_I(k)\, e^{-j2\pi k \frac{f}{F_e}} \tag{1}$$

2. Les éléments de la réponse impulsionnelle idéale associée sont donnés par les coefficients de la série de Fourier :

$$h_I(k) = \frac{1}{F_e} \int_{-\frac{F_e}{2}}^{\frac{F_e}{2}} H_I(f) e^{+j2\pi k \frac{f}{F_e}} df \tag{2}$$

   $h_I(k)$ représente ici le k$^{\text{ième}}$ point de la reponse impulsionnelle $h_I(t)$ du filtre qui est échantillonnée avec une période d'échantillonnage $T_e = \frac{1}{F_e}$ (il s'agit en réalité de $h_I(kT_e)$).

3. La réponse impulsionnelle réelle sera, en numérique, représentée par un tableau de valeurs correspondant à une troncature de la réponse impulsionnelle idéale échantillonnée :
   — $[h_I(-N)...h_I(N)]$, en supposant que l'on conserve $2N+1$ éléments de $h_I(k)$ et que l'on utilise une fenêtre rectangulaire (fenêtre naturelle) de troncature.
   — $[h_I(-N)w(-N)...h_I(N)w(N)]$, en supposant que l'on conserve $2N+1$ éléments de $h_I(k)$ et que l'on utilise une fenêtre de troncature, w, de $2N+1$ échantillons : $[w(-N)...w(N)]$.

   Le nombre de points, $2N+1$, conservé sur la réponse impulsionnelle idéale pour former le tableau représentant la réponse impulsionnelle réelle est appelé *ORDRE* du filtre. Les éléments du tableau représentant la réponse impulsionnelle réelle sont appelés *COEFFICIENTS* du filtre. La synthèse va alors consister à déterminer l'ordre du filtre, ainsi que la fenêtre de troncature à utiliser, afin que celui-ci satisfasse au gabarit souhaité.