TP ENSC

Analyse spectrale et filtrage

L'ANALYSE SPECTRALE est un outil qui permet de mettre en évidence des périodicités ou des pseudo-périodicités dans un phénomène étudié, par l'intermédiaire de signaux mesurés. Plus généralement c'est l'outil d'étude du contenu fréquentiel d'un signal et il est fondé sur la transformée de Fourier. L'objectif de ce travail est de mettre en pratique une partie des outils présentés dans différents cours, en plus de vous familiariser avec un outil logiciel très puissant : *matlab / octave*.

→ mini-doc de JF → aider à installer matlab
cheat sheet → license d'essais 30j au besoin.

1 Simulation de signaux

On considère une sinusoïde pure à la fréquence $f_0 = 440\ Hz$

$$x(t) = \alpha \sin\left[2\pi(f_0 t + \varphi)\right], \quad \text{pour } t \in \mathbb{R},$$

où $\alpha \in \mathbb{R}$ est l'amplitude et $\varphi \in [0, 1]$ est la phase réduite. Il s'agit du « la » du diapason et c'est aussi la fréquence que vous entendez lorsque vous décrochez votre téléphone.

On échantillonne ce signal à la cadence $F_e = 1/T_e = 10\ kHz$ ce qui donne un signal à temps discret

$$x_n = \alpha \sin\left[2\pi(\nu_0 n + \varphi)\right], \quad \text{pour } n \in \mathbb{Z},$$ $\nu_0 = \frac{f_0}{F_e}$

où ν_0 est la fréquence réduite. Que vaut-elle en fonction de f_0 et F_e ?

Sous *matlab*, on commence par fabriquer un vecteur contenant les échantillons de ce signal, pour $n = 0, 1, \ldots, N-1$. Pour cela, créez un fichier de commande contenant la séquence suivante.

```
% Nettoyage
    clear all
    close all

% Définition des constantes
    f0 = 440;
    Fe = 1e4;
    Nu0 = f0/Fe;
    Phi = 0.27;
    Alpha = 1.2;
    N = 1024;
```

F5 workspace

Exécutez alors le programme et assurez-vous que tout s'est passé comme attendu. La commande `whos` donne la liste des variables en mémoire. Pour obtenir la valeur d'une variable : tapez simplement son nom (puis la touche `enter`).

ou plutôt "disp" dans le code

Remarque 1 — *Pour toutes les commandes matlab vous avez accès à une aide en ligne en tapant* `help` *: par exemple* `help plot` *donne le manuel d'utilisation de la commande de tracé* `plot`*. La fin de chaque manuel renvoie à d'autres commandes : la fin du manuel de la commande* `plot` *renvoie* *par exemple aux commandes `title`, `xlabel`, `ylabel` qui permettent d'inclure des titres et légendes aux figures.*

Par ailleurs, la commande `lookfor` permet de faire une recherche dans les manuels : `lookfor fft` renvoie la liste des comandes matlab dont le manuel contient `fft`.

→ chatGPT
" je cherche une commande matlab pour faire une fft. Donne un MWE. "

La création du signal lui même est simple :

```
% Création du signal
    TpsD = (0:N-1)';
    Sig = Alpha * sin(2*pi*(Nu0*TpsD+Phi));
```

La première ligne fabrique un vecteur contenant le temps discret $[0, 1, \ldots, N-1]$. La seconde est *vectorielle* : chaque élément du vecteur `TpsD` est multiplié par la valeur `Nu0`, la valeur de `Phi` est ajoutée à chaque composante obtenue, puis chaque composante est multipliée par 2π et la fonction `sin` donne le sinus de chacune des composantes. Il s'agit là d'une caractéristique essentielle du langage *matlab* : toutes les opérations sont *matricielles* ou *vectorielles* et l'ensemble du TP se réalise sans boucle « for ». Exécutez à nouveau le programme et assurez vous que tout s'est passé comme prévu. Les vecteurs `TpsD` et `Sig` sont-ils des vecteurs lignes ou des vecteurs colonnes ?

Le tracé est également très simple.

```
% Tracé du signal
    figure(1),clf
    plot(TpsD,Sig)
    grid
    title('Signal')
    xlabel('Temps')
```

L'axe horizontal est alors gradué en temps discret. Modifier le programme pour faire apparaître le temps « physique » : créez pour cela une variable `TpsP`. Prenez le temps d'apprendre le rôle de chacune des commandes rencontrées. Voyez les possibilités des commandes `axis`, `grid`, `title`, `xlabel`, *etc...* Essayez également la séquence suivante.

```
% Tracé du signal
    figure(1),clf
    subplot(211),plot(TpsP,Sig)
    subplot(212),plot(TpsP(10:100),Sig(10:100))
```

Observez l'effet obtenu si vous multipliez la fréquence d'échantillonnage par 10 ou par 100. Si vous la divisez par 10 ou par 100.

Ajoutez une seconde raie assez proche de la première, par exemple à $f_1 = 460\ Hz$ et d'amplitude voisine. Ce signal permettra, dans la suite, d'évaluer le pouvoir de résolution de l'analyse de Fourier c'est-à-dire la capacité à séparer, à distinguer deux fréquences proches. Ajoutez également une composante à $2,5\ kHz$ d'amplitude moitié.

2 Transformée de Fourier

plusieurs définition suivant là où l'on met le 2π, $\sqrt{2\pi}$ ...

2.1 Introduction et rappel

Quelle est la transformée de Fourier à temps continu du signal $[x(t)]_{t\in\mathbb{R}}$? Quelle est la transformée de Fourier à temps discret du signal $[x_n]_{n\in\mathbb{Z}}$? Quel lien y-a-t-il entre elles ? Prenez les temps de répondre proprement à ces questions.

Remarque 2 — *Une expression du type*

$$F(\nu) = \sum_{n=-\infty}^{+\infty} f(n) e^{-2j\pi\nu n},$$

est une transformée de Fourier à temps discret. Il s'agit bien sûr d'une fonction 1-périodique de la variable ν continue. Pour la connaître entièrement il suffit donc de la connaître sur $[0,1]$ ou sur $[-0.5, 0.5]$.

En général, on ne peut pas calculer une telle expression, pour deux raisons. Premièrement la somme s'étend sur une infinité de termes et deuxièmement on ne peut pas réaliser ce calcul pour une infinité de valeurs de ν entre 0 et 1 (ou entre -0.5 et 0.5). En revanche le calcul devient possible

1. *si seulement un nombre fini de termes $f(n)$ est non nul et*
2. *si on se contente d'un nombre fini de valeurs de ν.*

Par exemple, si $f(n)$ est nul pour $n < 0$ et pour $n \geqslant N$, on peut calculer exactement les valeurs de $F(\nu)$ sur une grille fréquentielle régulière de M valeurs de ν entre 0 et 1 : $\nu_0, \nu_1, \ldots, \nu_{M-1}$, avec $\nu_m = m/M$. En pratique on choisira M assez grand pour avoir « l'impression » du continu (par exemple $M = 16N$ ou $M = 32N$). Ces valeurs s'obtiennent en utilisant un algorithme de transformée de Fourier rapide, si M est une puissance de 2. La syntaxe matlab à utiliser est `fft(x,M)`.

convergence de la série.

2.2 Manipulation de base

On se place dans la situation précédemment construite avec `N=1024` et `Fe=1e4`. En utilisant la fonction `fft` de *matlab*, calculez la FFT du vecteur `Sig` : `fft(Sig,M)`. Tracez le résultat en partie réelle, partie imaginaire, module et phase (la fonction `real` donne la partie réelle d'un nombre complexe et le manuel de cette fonction renvoie aux fonctions permettant de déterminer la partie imaginaire, le module et la phase). Tracez également le $\log_{10}$ du module. Graduez correctement l'axe des fréquences, à la fois en fréquence réduite et en fréquence réelle. Pour cela, vous pouvez créer un vecteur contenant l'axe des fréquences en utilisant la commande `linspace`. Pour représenter les transformées de Fourier entre -0.5 et 0.5, utilisez la fonction `fftshift`. Consultez le manuel : `help fftshift`. Cette fonction ne calcule pas de FFT, elle modifie seulement l'ordre des échantillons d'un vecteur : essayez `x=1:6` suivi de `fftshift(x)`. → dans le command window

Vous en arrivez à une partie essentielle et on se concentre sur l'analyse du module de la FFT.

- Vous devez retrouver des « pics » aux fréquences 440 Hz, 460 Hz et 2,5 kHz. Sont-ils positionnés précisément ? Évaluez cette précision en fonction de M. Expliquez la présence des trois autres pics.
- Vous devez constater également l'apparition de « *rebonds* » au pieds des pics. On parle de « *ringing* » dans la litérature anglosaxone. Expliquez leur origine. discrétise → convolue ⊓ → Sinus cardinal en Fourier

- On se concentre maintenant sur les composantes à 440 et 460 Hz. Observez-les lorsque $N = 1024$, $N = 512$ et $N = 256$. Commentez.
- Que se passe-t-il si on change F_e en $F_e = 6\ kHz$, en $F_e = 4\ kHz$, en $F_e = 0,5\ kHz$? Commentez.

2.3 Un peu de son...

Dans cette partie on s'intéresse aux propriétés fréquentielles d'un son musical. Cette étude permet de mettre en lien l'idée physique de fréquence, sa représentation mathématique et, éventuellement, sa perception.

Récupérez le fichier `Joli.wav` contenant un joli signal sonore et chargez-le en mémoire en utilisant la commande `wavread`. Sur certains systèmes, vous avez la possibilité de l'écouter. On pourra utiliser la séquence de commande suivante.

```
% Chargement
    [Sig Freq] wavread('Joli');
    [Sig, Freq] = audioread("Joli.wav")
% Ecoute
    sound(Sig,Freq);
```

Tracez-le en fonction du temps en faisant par exemple

```
% Tracé
    TpsP = ...
    figure(5),clf
    plot(TpsP,Sig)
    grid;title('Signal');xlabel('Temps')
```

et reconnaissez certaines caractéristiques (attaques, tenues, oscillations...).

Comme vous l'avez fait dans la section précédente, calculez et tracez le module de la FFT de ce signal. Commentez le résultat observé.

Naturellement, l'étude de la FFT (globale) de ce signal ne permet pas d'analyser les évolutions temporelles. Pour cela, on peut procéder à une analyse dite à fenêtre glissante : on découpe le signal temporel en une succession de tronçons, éventuellement recouvrant, et on réalise une analyse par FFT sur chaque tronçon. Cet outil sort du strict cadre de ce cours mais on vous fournit un fichier de commande qui permet de la réaliser : `PeriodoGlissant`. Récupérer ce fichier et utlisez-le de la manière suivante.

periodogramme
welch

```
% Chargement
    PeriodoGlissant(Sig,Freq);
```

et commentez ~~Hugement~~ ce que vous observez. Mettez un soin particulier à cette analyse. Reconnaissez certaines caractéristiques que vous avez observées temporellement (attaques, tenues, oscillations...).

2.4 En prime : un peu de bruit, mais pas trop

Dans cette partie on revient au signal que vous avez généré dans la section 1. On étudie l'influence d'un « *bruit* » : un exemple simple est constitué d'une suite de variables aléatoires indépendantes et identiquement distribuées sous une gaussienne centrée de variance $r_{\mathrm{b}} = \sigma_{\mathrm{b}}^2$.

```
% Ajout d'un bruit
    Bruit = SigmaBruit * randn(size(Sig));
    Sig = Sig + Bruit;
```

On définit un *rapport signal sur bruit* empirique en dB de la manière suivante :

$$RSB = 10 \log_{10} \left[\frac{\sum_1^N x_n^2}{\sum_1^N b_n^2} \right] \tag{1}$$

où les x_n et les b_n sont respectivement les échantillons du signal et du bruit.

Ajoutez différents niveaux de bruit et observez le signal temporel et le module de sa FFT. Commentez. Jusqu'à quel niveau de bruit les trois raies restent visibles ?

Proposez une opération de séparation du signal et du bruit.

Méthode des sous espaces vectoriels / ESPRIT / MUSIC

2.5 Filtrage

On veut réaliser maintenant une opération de filtrage ~~passe-pas~~ passe-Bas du signal, pour éliminer la composante à $2,5\ kHz$ et ne conserver que les composantes à $440\ Hz$ et $460\ Hz$. On éliminera pour cela les fréquences au delà de $2\ kHz$. Réalisez très simplement cette opération dans le domaine de Fourier : vous avez ainsi réalisé un filtrage passe-bas « idéal ».

Cette opération de filtrage idéal dans le domaine de Fourier ne peut pas être utilisée pour réaliser un filtrage en ligne. Expliquez pourquoi. Dans le cas où on est contraint de réaliser ce filtrage en ligne, une possibilité repose sur le filtrage convolutif. On pourra utiliser, par exemple, le filtre de réponse impulsionnelle h définie par

$$h(n) = \begin{cases} 1/P & \text{si } n \in [0; P-1] \\ 0 & \text{sinon} \end{cases}$$

S'agit-il bien d'un filtre passe-bas ? Quel est son transfert en fréquence ? Utilisez la fonction `filter` pour filtrer le signal d'origine et éliminer les composantes au delà de $2\ kHz$. Réglez empiriquement la valeur de P. Comment prévoir à l'avance une « bonne valeur » pour P. Comparez les résultats obtenus ici à ceux obtenus par filtrage idéal.

$$\sum_m h(m) = 1$$