Capacités numériques : calcul du spectre d'un signal par transformée de Fourier discrète rapide

I Introduction

Il est impossible avec un calculateur de calculer numériquement la transformée de Fourier d'un signal (il faudrait pouvoir calculer la valeur de cette fonction en un infinité de points, chaque calcul utilisant de plus lui-même un calcul d'intégrale impropore !)

Comme toujours en numérique, on va passer par de la discrétisation.

II Définition de la Transformée de Fourier Disctrète (TFD)

La TFD d'un signal numérique à N échantillons $s_E = [s_0, s_1, \cdots, s_{N-1}]$ est une liste de N nombres complexes $[\underline{c}_0, \underline{c}_1, \cdots, \underline{c}_{N-1}]$ définie par

$$\forall k \in [|0, N-1|], \underline{c}_k = \sum_{j=0}^{N-1} s_j \exp\left(-i\frac{2\pi kj}{N}\right)$$

On vérifie facilement que $\underline{c}_0$ est réel et que $\forall k \in [|1, N-1|], \underline{c}_{N-k} = \underline{c}_k^*$.

Si l'on implémente le calcul en suivant la définition on obtient une complexité quadratique $\mathcal{O}(N^2)$. Cela n'est pas acceptable surtout si on veut obtenir quasiment en temps réel la TFD.

III Alogrithme de Cooley et Tuckey

L'algorithme de Cooley-Tuckey, algorithme récursif basé sur la méthode "diviser pour régner", permet d'obtenir une meilleure complexité temporelle. .

L'idée est de séparer le calcul des termes correspondant à j pairs et ceux à j impairs.

$$\underline{c}_k = \sum_{j'=0}^{N/2-1} s_{2j'} \exp\left(-i\frac{2\pi k(2j')}{N}\right) + \sum_{j'=0}^{N/2-1} s_{2j'+1} \exp\left(-i\frac{2\pi k(2j'+1)}{N}\right)$$

$$= \sum_{j'=0}^{N/2-1} s_{2j'} \exp\left(-i\frac{2\pi kj'}{N/2}\right) + \exp\left(-i\frac{2\pi k}{N}\right) \sum_{j'=0}^{N/2-1} s_{2j'+1} \exp\left(-i\frac{2\pi kj'}{N/2}\right)$$

Dans cette expression on reconnaît deux autres TFD !

- Si on pose $\underline{P}_k = \sum_{j'=0}^{N/2-1} s_{2j'} \exp\left(-i\frac{2\pi kj'}{N/2}\right)$, alors $[\underline{P}_0, \underline{P}_1, \cdots, \underline{P}_{N/2-1}]$ est la TFD du signal à $N/2$ échantillons $[s_0, s_2, \cdots, s_{N-4}, s_{N-2}]$;
- Si on pose $\underline{I}_k = \sum_{j'=0}^{N/2-1} s_{2j'+1} \exp\left(-i\frac{2\pi kj'}{N/2}\right)$, alors $[\underline{I}_0, \underline{I}_1, \cdots, \underline{I}_{N/2-1}]$ est la TFD du signal à $N/2$ échantillons $[s_1, s_3, \cdots, s_{N-3}, s_{N-1}]$;

Ainsi en notant $w = \exp\left(-i\frac{2\pi}{N}\right)$, on peut écrire pour $0 \leqslant k \leqslant N/2$:

$$\underline{c}_k = \underline{P}_k + w^k \underline{I}_k \tag{1}$$

D'autre part pour $0 \leqslant k \leqslant N/2$

- $\underline{c}_{N/2+k} = \underline{c}^*_{N/2-k}$;
- $\underline{c}^*_{N/2-k} = \underline{P}^*_{N/2-k} + \exp\left(i\frac{2\pi\,(N/2-k)}{N}\right)\underline{I}^*_{N/2-k}$
- $\underline{P}^*_{N/2-k} = \underline{P}_k$ et $\underline{I}^*_{N/2-k} = \underline{I}_k$.

Finalement pour $0 \leqslant k \leqslant N/2$:

$$\underline{c}_{N/2+k} = \underline{P}_k - w^k \underline{I}_k \tag{2}$$

Ainsi on ramène le calcul d'une TFD d'un signal à N échantillons à deux calculs de TFD de signaux à $N/2$ échantillons.

IV Algorithme et travail demandé

On a ainsi l'algorithme pour le calcul de la TFD d'un signal à $N = 2^p$ échantillons :

- Cas de base : si $N = 1$ (i.e. $p = 0$), le programme renvoie la liste $[s_0]$;
- Récursion : si $N > 1$ (i.e. $p > 0$), on fait deux appels récursifs pour calculer les TFD des signaux à $N/2 = 2^{p-1}$ échantillons constitués par les échantillons de rang pair et les échantillons de rang impair du signal original, puis on calcule la TFD du signal original par les relations (1) et (2).

Écrire en python une fonction TFD qui prend en argument une liste de valeurs d'échantillons d'un signal numérique, dont la taille est une puissance de 2. On pourra utiliser la librairie cmath qui permet de calculer directement avec des nombres complexes. La racine de l'unité i s'écrit alors en python 1j. Cette librairie offre une fonction exp permettant le calcul d'exponentielles complexes.

Quelle est la complexité de cet algorithme récursif ?

V Lien entre la TFD et le spectre du signal

Désormais on suppose que le signal de N échantillons $s_E = [s_0, s_1, \cdots, s_{N-1}]$ est l'échantillonnage d'un signal $s(t)$ avec la fréquence d'échantillonnage f_e . Alors la TFD donne l'évaluation suivante du spectre de $s(t)$:

- la composante continue de $s(t)$ est : $\frac{c_0}{N}$ (on se rappelle que c_0 est réel) ;
- pour $0 < k \leqslant N/2$, l'amplitude de la fréquence $f_k = k\frac{f_e}{N}$ dans le signal $s(t)$ est : $\frac{2}{N}\,|c_k|$;
- le déphasage de la composante de fréquence f_k dans le signal $s(t)$ est : $\arg(c_k)$.

Ce spectre calculé se rapproche du spectre réel du signal $s(t)$ dans la mesure où d'une part le critère de Nyquist-Shannon, pour éviter le repliement est respecté et que d'autre part l'échantillonnage est celui d'un signal périodique sur un nombre entier de périodes.

Écrire une une fonction spectre qui reçoit une liste d'instants d'échantillonnage (régulièrement espacés) et la liste des valeurs d'un signal à ces instants, et retourne une liste de fréquences, la liste des amplitudes et la liste des déphasages pour ces fréquences.

On utilisera dans un premier temps des listes d'échantillonnage dont la taille est une puissance de 2.

On pourra ensuite écrire une fonction supplémentaire qui lorsque N n'est pas une puissance de 2 calcule par interpolation un nouveau signal échantillonné avec un nombre de points N' échantillonnés qui soit une puissance de 2, $N' = 2^p$ où p est choisi de telle manière que tel N et N' sont les plus proches possible.

Tester votre fonction sur des exemples classiques du cours (signal sinsoïdal, rectangulaire, triangulaire).