Calcul formel et numérique
Licence de Physique, 2-ième année, Université de Provence

Transformation de Fourier I

15 mars 2006

La transformée de Fourier finie d'un vecteur $\underline{u} = \{u_0, \dots u_{N-1}\} \in \mathbb{C}^N$ est le vecteur $\hat{\underline{u}} = \mathcal{F}\underline{u} = \{\hat{u}_0, \dots \hat{u}_{N-1}\} \in \mathbb{C}^N$ défini par

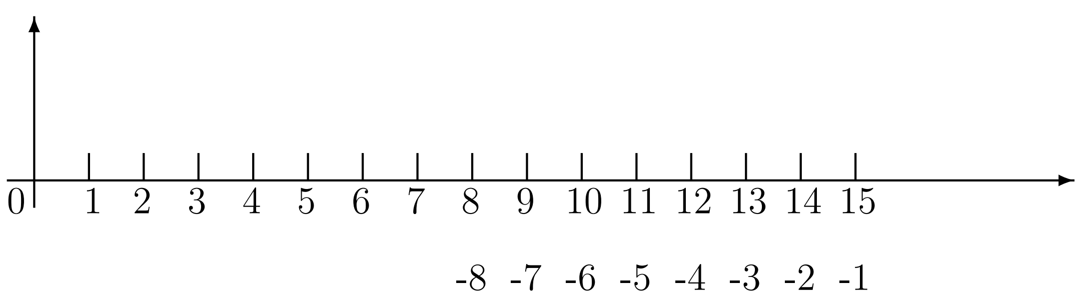
$$\hat{u}_k = \sum_{n=0}^{N-1} u_n e^{-2i\pi kn/N} \ , \quad k = 0, \dots N-1 \ . \tag{$\flat$}$$

La transformation de Fourier finie est inversible, via

$$u_n = \frac{1}{N} \sum_{k=0}^{N-1} \hat{u}_k e^{2i\pi kn/N} \ , \quad n = 0, \dots N-1 \ . \tag{$\sharp$}$$

La transformation de Fourier finie est implémentée sous OCTAVEvia les instructions `fft` (transformation de Fourier finie) et `ifft` (transformation de Fourier finie inverse). Ces instructions implémentent en fait l'algorithme de **Transformation de Fourier Rapide** (TFR).

> **Conventions d'indice :** *Il est important d'être attentif au point suivant : dans l'environnement numérique* OCTAVE, *les indices de tableau commencent à 1, et pas 0. Par contre, les routines* `fft` *et* `ifft` *considèrent le premier élément du tableau comme la coordonnée d'indice 0. Par exemple, l'élement* `u(1)` *du vecteur* $\underline{u}$ *contient en fait la coordonnée* u_0*, l'élément* `u(2)` *contient la coordonnée* u_1*, et ainsi de suite.*
>
> **Remarque :** *Par convention, la TFF* $\hat{u}_k$ *est définie pour* $k \in \{0, 1, \dots N-1\}$*. Ceci dit, la formule* ($\flat$) *peut aussi être utilisée pour n'importe quel* k *entier, mais ceci n'apporte rien de plus car* $\hat{u}_{k+N} = \hat{u}_k = \hat{u}_{k-N}$ *quel que soit* k*. Par convention, on appelle "basses fréquences" les valeurs de* k *proches de 0, et "hautes fréquences" les valeurs de* k *proches de* $N/2$ *(voir ci-dessous).*



1 Préliminaires

On peut générer une sinusoïde de fréquence f donnée par l'instruction

```
sinusoide = sin(2*pi*f*(0 :(N-1))/N)
```

N étant la longueur du vecteur (ligne) ainsi généré, et $f \in [0, N-1]$ sa fréquence. Le vérifier, et afficher graphiquement des sinusoïdes pour diverses valeurs de la fréquence, et étudier l'influence de celle-ci.

2 Prise en main

La transformation de Fourier permet de mettre en évidence les fréquences de sinusoïdes. Par exemple, la transformée de Fourier d'une sinusiïde de fréquence donnée exhibera un "pic" à cette fréquence, la transformée de Fourier de la somme de deux sinusoïdes exhibera deux "pics" aux fréquences correspondantes,... C'est ce que l'on se propose de vérifier dans un premier temps.

2.1 Mise en évidence de la fréquence d'une sinusoïde

Réaliser un *script* effectuant les opérations suivantes. Pour ce qui est de l'affichage graphique, on pourra utiliser les instructions `subplot (3,1,1)` pour tracer le résultat dans la partie supérieure du graphique, `subplot (3,1,2)` pour la partie médiane, et `subplot (3,1,3)` pour la partie basse.

1. Générer un vecteur (colonne) `vec` de longueur N fixée (ordre de grandeur pour N : quelques centaines), et le représenter graphiquement. On pourra par exemple considérer notamment une sinusoïde de fréquence variable (de la forme $n \to \cos(2\pi f n/N)$, f étant la fréquence, comprise entre 0 et 1), ou des sommes de telles sinusoïdes.
2. Calculer la TFF `vecchapeau` du vecteur ainsi généré. en utilisant `fft`, et afficher cette dernière graphiquement (*attention :* `vecchapeau` *est en général complexe, il est préférable de tracer sa partie réelle, ou mieux son module, en utilisant la fonction* `abs`).
3. En utilisant l'instruction `ifft`, calculer la transformée de Fourier inverse `vecrec` de `vecchapeau` , et la représenter graphiquement (*attention : si* `vec` *est réel,* `vecrec` *doit l'être aussi, on étudiera donc sa partie imaginaire*).

2.2 Mise en évidence de la fréquence d'une sinusoïde "bruitée"

Renouveler ces opérations dans le cas où l'on remplace la sinusoïde par la somme d'une sinusoïde et d'un "bruit", c'est à dire un vecteur de même taille, contenant des nombres aléatoires. On peut générer de tels nombres en utilisant la fonction `randn`, par exemple en calculant

```
vec 2 = vec + C*randn(size(vec))
```

(se documenter avec le `help` en ligne sur ces fonctions) où `C` est une constante (prendre `C=1` par défaut ; on pourra aussi effectuer des expériences en prenant des valeurs de `C` de plus en plus grandes, ou de plus en plus petites si nécessaire).

3 Séries de Fourier tronquées et phénomène de Gibbs

On se propose ici d'étudier la convergence des séries de Fourier tronquées. Par exemple, étant donnée une fonction $f : [0,1] \to \mathbb{C}$, sa série de Fourier tronquée est donnée par

$$f_N(t) = \sum_{k=-N}^{N} c_k(f) e^{2i\pi kt} \ ,$$

où les nombres $c_k(f)$ sont les coefficients de Fourier usuels de f :

$$c_k(f) = \int_0^1 f(t) \exp(-2i\pi kt)\, dt \ .$$

On construira de telles reconstructions partielles pour différentes valeurs de N, et on les affichera graphiquement.

1. Construire un vecteur comportant des variations brusques, par exemple un vecteur $u \in \mathbb{R}^N$ de longueur N, dont les composantes u_n valent ± 1, et sont constantes sur des intervalles de longueur fixée $L \ll N$. On pourra par exemple générer un vecteur uniformément égal à 1 (`vec = ones(N,1)`), puis remplacer certaines valeurs par des -1 (par exemple `vec(1 :floor(N/2))=-1`). Représenter graphiquement ce vecteur.
2. Calculer sa transformée de Fourier finie `vecchapeau`, et en représenter graphiquement le module.
3. Calculer une transformée modifiée, en remplaçant par des valeurs nulles les $\hat{u}_k$ correspondant aux plus hautes fréquences, c'est à dire aux valeurs de k les plus proches de $N/2$.
4. Calculer la transformée de Fourier inverse `vecrec`, et la représenter graphiquement.

On pourra effectuer ce calcul en faisant varier le nombre de coefficients de Fourier annulés, et analyser la reconstruction obtenue dans ces différents cas.