Transformée de Fourier discrète.

1 Préliminaire : Téléchargement de Wavelab

Wavelab est une *toolbox* matlab, c'est à dire un ensemble de programmes matlab élaborés par une équipe de l'université de Stanford. Cette toolbox est gratuite et se télécharge simplement en tapant *Wavelab802* ou *Wavelab850* dans *Google*. Une fois que la toolbox est téléchargée, il faut utiliser la commande *Set Path* dans le menu *file* de matlab pour s'assurer que Matlab ira chercher les fonctions de Wavelab si on les utilise dans matlab.
Pour vérifier que Wavelab est installé et accessible, lancer la commande :

```
>>S1=MakeSignal('Piece-Regular',1024);
>>S2=MakeSignal('Blocks',2048);
>>plot(S1);figure;plot(S2);
```

Cette toolbox a été créée pour la transformée en ondelettes mais elle permet aussi de créer des signaux 1D et des images assez simplement.

2 Transformée de Fourier discrète de fft

La Transformée de Fourier Discrète (TFD) est une transformée qui peut être définie en plusieurs dimensions. En 1D la TFD est une transformation qui à un vecteur de nombres complexes de taille N renvoie un vecteur de nombres complexes de taille N. En dimension supérieure cette transformée envoie également un rectangle discret sur un rectangle discret de même taille. Nous allons nous placer dans un premier temps en 1D et nous verrons ensuite le cas de la 2D.
Si on note $(f[n])_{1\leqslant n\leqslant N}$ un vecteur un vecteur de taille N, sa transformée de Fourier discrète sera le vecteur $(\hat{f}[k])_{1\leqslant k\leqslant N}$ défini par :

$$\hat{f}[k]=\sum_{n=1}^{N} f[n]e^{-\frac{2i\pi(k-1)(n-1)}{N}}=\sum_{n=1}^{N} f[n]\omega_N^{(k-1)(n-1)} \tag{1}$$

où $\omega_N=e^{\frac{-2i\pi}{N}}$. Il faut faire très attention à la définition que l'on prend. Certaines définitions font varier k entre 0 et $N-1$. En matlab comme le premier indice des vecteurs est 1, on a cette définition.
Le calcul d'un tel vecteur peut être fait de manière rapide quand N est une puissance de 2 en utilisant l'algorithme de transformée de Fourier rapide (Fast Fourier Transform, fft). La commande matlab est *fft*. Il existe aussi une commande d'inversion de la transformée discrète *ifft* basée sur la formule de reconstruction suivante :

$$f[n]=\frac{1}{N}\sum_{k=1}^{N} \hat{f}[k]e^{\frac{2i\pi(k-1)(n-1)}{N}}=\sum_{n=1}^{N} f[n]\omega_N^{-(k-1)(n-1)} \tag{2}$$

La différence entre la formule de Transformée de Fourier Discrète (TFD) et son inverse réside dans le signe de l'exponentielle et dans la renormalisation en $\frac{1}{N}$ dans la transformée inverse. Si vous êtes amené à utiliser des TFD dans d'autres langages, vérifier ces deux points. Ils peuvent varier. On peut avoir une renormalisation en $\frac{1}{N}$ dans le transformée directe et dans ce cas pas dans la trasnformée inverse. On peut avoir dans les deux transformées un facteur $\frac{1}{\sqrt{N}}$ et,ou plus rarement un signe plus dans l'exponentielle dans la transformée directe et dans ce cas un signe moins dans la transformée inverse.
Dans la suite on utilisera par abus de langage le terme *FFT* qui est un algorithme pour *DFT* qui est une transformation.

3 Visualisation de la fft

La FFT d'un vecteur est un vecteur complexe. Même si pour des raisons théoriques le la FFT d'un vecteur doit être réelle, numériquement il y aura toujours une partie imaginaire due aux imprécisions machine. C'est pourquoi on couplera souvent les commandes *ifft* et *real*.

1. Utiliser le programme suivant pour visualiser les tranformées de Fourier des signaux $S1$ et $S2$ calculés plus haut.

```
function Visufft(S)
FS=fft(S);
subplot(3,1,1)
plot(real(FS));
subplot(3,1,2);
plot(imag(FS));
subplot(3,1,3);
plot(abs(FS));
```

Les coefficients de la fft peuvent être vu comme les coefficients de la décomposition d'un vecteur dans la base formée par les exponentielles complexes discrètes. En effet pour tout entier k entre 1 et N on peut construire le vecteur e_k de longueur N défini par : $e_k(n) = e^{\frac{2i\pi(k-1)(n-1)}{N}}$. La famille $(e_k)_{1\leqslant k\leqslant N}$ forme une base orthogonale pour le produit scalaire défini par

$$\langle f, g\rangle = \sum_{k=1}^{N} f[k]\overline{g[k]}. \tag{3}$$

Cette base n'est pas orthonormée dans la mesure où les vecteurs ont tous une norme $\sqrt{N}$. Si on avait un facteur $\frac{1}{\sqrt{N}}$ dans la transformée directe on aurait une Base Orthonormée (BO). Comme tous les vecteurs ont la même norme, on peut faire l'abus de langage consistant à dire que les exponentielles complexes discrètes forment une BO pour le produit scalaire usuelle. Une autre ruse consiste à modifier le produit scalaire par un facteur $\frac{1}{N}$ pour e,n faire une BO.

2. Construire pour $k = 0, 1, 2, 5, 10$, le vecteur e_k et visualiser sa transformée de Fourier à l'aide du programme précédent.
3. Afficher la fft du vecteur suivant :

```
>>k=7;
>>N=1024;t=[0:1/N:1-1/N];S3=sin(2*pi*t);
```

et interpréter le résultats.

4. Tester d'autres valeurs de k, remplacer le *sinus* par un *cosinus*, introduisez une phase ϕ et essayer d'anticiper le résultat. Tester en particulier les valeurs $k = 1020, 1021, 1022$. Qu'en pensez vous ?
5. Justifier que si $(f[k])_k$ est un vecteur dont les composantes sont **réelles**, le module de la fft admet une symétrie. Plus précisément montrer que pour tout $k > 0$, $\hat{f}[k+1] = \overline{\hat{f}[N-k+1]}$.
6. Est ce encore vrai si le vecteur f n'est pas réel ?
   L'espace vectoriel engendré par $e_{k+1} = e^{\frac{2i\pi k\cdot}{N}}$ et $e_{N-k+1} = e^{-\frac{2i\pi k\cdot}{N}}$ est le même que celui engendré par $\sin(\frac{2i\pi k\cdot}{N})$ et $\cos(\frac{2i\pi k\cdot}{N})$. Ansi pour tout $k > 1$, les composantes d'indice $k+1$ et $N-k+1$ de la *fft* peuvent être interprétées comme la composante d'un vecteur sur les sinusoides de fréquence $\frac{2\pi k}{N}$.
   On appelle basse fréquences, les valeurs de $\hat{f}[k]$ pour les petites valeurs de k. Elles correspondent aux coeffcients de décomposition d'un vecteur f sur des exponentielles discrète complexes qui oscillent peu. On appelle *hautes fréquences*, les coeffcients associés aux exponentielles qui oscille beaucoup. Les exemples précédents indiquent que la fft de matlab dispose les *hautes fréquence* au centre et les *basses fréquences* à gauche et à droite.
   Ces concepts de *basses fréquences* et de *hautes fréquences* discrètes sont discutables dans la mesure où chaque coefficients correspond à une infinité de fréquences différentes.
7. A quoi correspond la composante $\hat{f}[1]$?
8. Expliquer ce que renvoie le programme suivant :

```
function Exo1(S,k)
plot(S);hold on;
FC=0*S;
FS=fft(S);
FC(k+1)=FS(k+1);
FC(N-k+1)=FS(k+1);
C=real(ifft(FC));
plot(C,'r');hold off;
```

9. Après l'avoir testé sur différents vecteurs, expliquer ce que renvoie le programme suivant :

```
function Exo2(S,k)
close all;
FS=fft(S);
for j=1:k
plot(S);hold on;
FApp=FS;
FApp(j+1:N-j+1)=0;
App=real(ifft(Fapp));
plot(App,'r');hold off;
pause(0.1);
end
```

10. Que se passe t-il si on utilise cette fonction si S est un vecteur e_k ou une sinusoide de la forme $S[n] = \sin(\frac{2i\pi(k-1)n}{N} + \phi)$?
11. La commande *fftshift* est une commande qui a pour but de dsiposer les coefficients de la FFT différamment en mettant au centre les coefficients dits basse fréquence au centre et haute fréquence aux extrémités. Attention cette command n'effectue pas de FFT elle échange juste des coefficients. Tester la commande *ffshift* sur plusieurs exemples.

4 FFT et note de musique

La transformée de Fourier est un des principaux outils en traitement du son. Elle permet d'analyser et de mesurer les différentes fréquences présentes dans un son. Si le son est musical on peut associer à chaque fréquence une note.
Télécharger sur ma page web un son de saxo, de violon et un mélange de différents instruments. Ces fichiers sont au format *.wav*. Pour transformer ces fichiers en vecteur matlab utiliser la commande suivante :

```
>>[Y,Fs,B]=wavread('Mix11.wav');
```

La variable Y est un vecteur (parfois une matrice avec deux lignes si le son est enregistré en stéréo), contenant la position de la membrane d'un haut parleur par rapport à sa position de repos à chaque échantillon de temps. Y prend des valeurs réelles entre -1 et 1.
La variable Fs contient la fréquence d'échantillonnage du son exprimé en Hertz. C'est cette fréquence qui permet de passer d'un son à un vecteur et d'un vecteur à un son. Si $Fs = 44000$ cela signifie que le son est échantilloné à 44000 Hz, autrement dit deux valeurs consécutives de Y représentent les positions de la membrane d'un haut parleur à deux instants séparés de $\frac{1}{44000}$ secondes.
La variable B ne sera pas utilisé et contient le nombre de bits sur lesquels sont codés les composantes de Y.

12. Visualiser le vecteur Y dans sa totalité pour chacun des 3 sons.
    Calculer la DFT d'un vecteur représentant un son musical sur un temps long n'a pas d'intérêt majeur. En effet, la transformée de Fourier représente un vecteur comme une somme de sinusoides discrètes. Quand plusieurs notes sont jouées successivement il est rarement intéressant de représenter le son comme une somme de sinusoides. Ainsi dans la pratique, on découpe de le son en sous-vecteurs, appelés *fenêtres* sur lesquels on calcule des DFT.
13. Dans les différents vecteurs issus de son musicaux, extraire des sous vecteurs de taille 1024 ou 2048 et les visualiser. A quelle durée de son ces extraits correspondent-ils ?
14. Visualiser les FFT de ces extraits.
    Le premier coefficient de la FFT correspond à la somme des coefficients de f, il mesure donc la moyenne du vecteur f. Le dernier coefficient correspond à la fréquence d'échantillonnage : par exemple 44000 Hz. Les autres coefficients correspondent à des fréquences régulièrement espacées entre 0 et FS. Si on calcule une FFT sur 44001 points, chaque coefficient correspond à *un intervalle fréquenciel de 1 Hz*. Ainsi le 101 ième coefficient correspond à une fréquence de 100 Hz.
    Si on fait une FFT sur 441 points chaque coefficient correspond à un intervalle de 10 Hz. Ainsi plus la fenêtre est grande plus on a de précision fréquentielle mais moins on a de précision spatiale.
15. Dans les notes jouées par le saxophone et le violon on remarque plusieurs coefficients de Fourier importants. A quoi correspondent ils ? Déterminer les notes jouées.
16. A l'aide la commande *wavwrite* qui s'utilise de manière suivante :

    ```
    >>wavwrite(Y,FS,'toto.wav');
    ```

    générer un petit fichier wav d'une durée de 2 secondes avec une note pure, par exemple le *la* 440. C'est à dire à 440 Hz.

5 Filtrage et convolution

Si f et g sont deux vecteurs de longueur N, la convolution circulaire de f et g est le vecteur $h = f \star g$ de longuer N défini par :

$$h[n] \sum_{k=1}^{N} f[k]g[n-k] = \sum_{k=1}^{N} f[n-k]g[k] \tag{4}$$

où f et g sont considérés comme des vecteurs périodiques de période N, c'est à dire tels que $f[l] = f[l+N]$ et $g[l] = g[l+N]$, $\forall l \in \mathbb{Z}$. Un calcul élémentaire montre que $\forall k \in \mathbb{Z}$, $\hat{h}[k] = \hat{f}[k]\hat{g}[k]$.

17. Ecrire une fonction matlab

    ```
    function h=ConvFourier(f,g)
    ```

    qui calcule la convolution de deux vecteurs en utilisant la FFT.
18. Faites plusieurs tests et afficher les résultats.
19. Quel est le résultat de la convolution entre deux vecteurs e_k et e_l ?
20. Déterminer l'élément neutre pour la convolution discrète circulaire.
21. Quel est le résultat de la convolution entre un vecteur S de taille N et un vecteur de la forme $\sin(2k\pi/N)$?
    On construit une gaussienne discrète à l'aide la fonction suivante :

```
function G=Gaussienne(N,s)
t=[0:1/(N-1):1]*2-1;
G=exp(-t.^2/s);
G=G/sum(G);
```

22. Afficher plusieurs gaussiennes en testant différentes valeurs de s.
23. Pour différentes valeurs de s afficher également en parallèle le module de la FFT de la gaussienne en centrant les basses fréquences (en utilisant fftshif.)
24. A quoi ressemble le module de la FFT d'une gaussienne discrète ?
    Dans la plupart des convolution, on réalise la convolution entre un objet d'intérêt (Signal ou Image) avec un vecteur lié à un effet (Flou, dérivation, moyennage local). Ce vecteur lié à la transformation est appelé filtre.
25. Appliquer la fonction suivante à différents vecteurs

```
function Chaleur(S)
N=length(S);
for k=1:50
G=Gaussienne(N,k/5);
C=ConvFourier(S,G);
plot(S);hold on;plot(C,'r');hold off;pause(0.1);
end
```

26. A quelle opération physique correspond la convolution par une gaussienne ?
27. A quelle opération correspond la convolution par le vecteur *dec* suivant :

```
>>dec=0*S;
>>dec(30)=1;
```

28. A quelle opération correspond la convolution par le vecteur d suivant :

```
>>d=0*S;
>>d(1)=1;d(2)=-1;
```

6 Transformée de Fourier 2D

Pour Matlab, une image de taille (m,n) est une matrice I de taille (m,n), et la valeur de $I(i,j)$ correspond à la valeur du niveau de gris de l'image au pixel (i,j).
Matlab est capable de lire à peu près tous les formats standards d'images.

6.1 Exemples : visualisation d'une image

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Chargement d'une image en Matlab:
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
load gatlin2;
% -> L'image est chargee dans la variable X
%Autres images:
%load clown; load gatlin; load mandrill;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Visualisation:
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
imagesc(X);
colormap gray;
%pour voir l'image en niveaux de gris
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Pour ouvrir une deuxieme figure:
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
figure(2);
colormap gray;
XX=imread('cameraman.tif');
imshow(XX);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

On peut aussi utiliser la commande *imwrite*, puis visualiser l'image sauvegardée avec son éditeur d'image préféré (souvent plus pratique que ceux de Matlab).
Sources d'erreur classique : Il faut faire attention que les commandes *imwrite* ou *imread* suppose que les images sont codées comme des entiers non signés (*uint8* permet de faire la conversion), alors

que *imagesc* lit les images au format *double*, et surtout que vous ferez tous vos calculs au format *double*. Si vous ne faîtes pas attention au format de vos données, vous obtiendrez de mangifiques images noires en sortie de vos codes ...

Choix d'une image : Vous pouvez continuer le TP avec votre image favorite (à condition qu'elle soit de taille *raisonnable*). Sachant que le TP va être basé sur la transformée de Fourier, et qu'avec matlab vous disposez de la FFT, donner une condition nécessaire pour que la taille soit *raisonnable* ?

Si vous choisissez une image couleur, n'oubliez pas de la convertir en niveau de gris (e.g. *a=mean(a,3)*).

6.2 Un premier exemple de fonction Matlab : bruitage d'une image

```
function out = bruitage_gaussien(I,s)
%bruitage gaussien (d'ecart-type s) d'une image I
% bruitage_gaussien(I,s)

[m,n]=size(I);
J=zeros(m,n);

%creation du bruit gaussien
J=s*randn(m,n);

%bruitage
out=I+J;
```

7 Transformée de Fourier d'une image

En dimension 2, La TFD de l'image $\{f[k,l]\}$ $(k,l = 0,\ldots,N)$ est définie par

$$\hat{f}[k,l] = \sum_{n=1}^{N}\sum_{n=1}^{N} f_[m,n]\omega_N^{-(m-1)(k-1)}\omega_N^{-(l-1)(n-1)}$$

La transformée inverse est donnée par

$$\hat{f}[k,l] = \frac{1}{N^2}\sum_{n=1}^{N}\sum_{n=1}^{N} \hat{f}[m,n]\omega_N^{(m-1)(k-1)}\omega_N^{(n-1)(l-1)}$$

7.1 Visualisation du spectre d'une image synthétique

On va commencer par observer le spectre d'images synthétiques.

Sinusoïde

```
x=[0:255]';
y=2*pi*(8/256)*ones(1,256);
a=sin(x*y);
```

29. Visualiser a dans une première figure. Pour visualiser son spectre, on peut faire :

```
A=fft2(a);
A=fftshift(A);
figure
imagesc(1+log(abs(A)));
colomap gray;
```

30. Expliquer le spectre observé (nombre de points lumineux, distance entre ces points,...).

Carré

```
b=zeros(256,256);
b(63:191,63:191)=1;
```

31. Visaliser b et son spectre comme précédemment. Commentaires ?

Sinusoïde fenêtrée Avec les notations précédentes :

```
c=b.*sin(x*y);
```

32. Visualiser c et son spectre comme précédemment. Commentaires ?

7.2 Visualisation du spectre d'une image réelle

33. Charger l'image clown. La mettre dans la variable a.
    Calculer la TFD de l'image a (fonction $fft2$). La stocker dans la variable A
    Visualiser $log(1+abs(A))$.
34. **A quoi correspondent les structures horizontales, verticales, obliques, observées dans la TFD ?**
35. Refaire l'opération avec d'autres images (gatlin, mandrill, ...). Commentaires ?

8 Phase et module de la transformée de Fourier

36. Nous allons illustrer l'importance respective de la phase et du module des coefficients de la Transformée de Fourier. Nous commençons par échanger les phases des TFD de deux images.
    – Choisisser une image b différente de a. Calculer les TFD A et B respectivement de a et b. Remplacer les coefficients de B par leurs modules, puis multiplier les par la phase des coefficients correspondants de a (on pourra utiliser les fonctions *abs* et *angle*). Utiliser la fonction $ifft2$, et afficher le résultat.
    – Commentaires ?

9 Filtrage passe-haut et images

Les contours dans une image constituent une information essentielle utilisée par notre système visuel pour comprendre le contenu d'une image. L'extraction des contours est un point clé du traitement d'images. Comme ces contours correspondent à des discontinuités, une méthode simple pour les mettre en évidence consiste à appliquer les filtres suivants, passe-haut : Si I est une image,

```
>>f=0*I;f(1,1)=1;f(1,2)=-1;
>>g=f';
```

Pour appliquer les filtres on effectue une convolution en passant par la transformée de Fourier 2D :

```
function C=ConvFourier2D(I,h)
C=real(ifft2(fft2(I).*fft2(h)));
```

Dans le cas présent le passage par Fourier n'est pas nécessaire car les filtres que nous utilisons sont de tout petit support mais c'est toujours un moyen possible.

37. Construire et visualiser

    ```
    >>g1=ConvFourier2D(I,f);
    >>g2=ConvFourier2D(I,g);
    ```

    puis construire une image par sommation des carrés des images $g1$ et $g2$:

    ```
    >>q=sqrt{g1^2+g2^2};
    ```

    Il s'agit d'une approximation discrète du module du gradient des images. Appliquer cette procédure à différentes images (éventuellement brutiée).

    L'inconvénient d'une telle méthode est sa forte snesibilité au bruit et aux textures des images. On procède donc en général à uen régularisation préalable de l'image. Effectuez l'opérération précédente après avoir convolué l'image par un filtre gaussien 2D. Un tel filtre se construit comme le produit de deux gaussiennes 1D.