IMA TP5 : Transformée de Fourier 2D

pierre.maurel@irisa.fr

http://www.normalesup.org/~pmaurel/IMA/

1 Transformée de Fourier 1D

La transformée de Fourier discrète 1D est calculée en matlab par la fonction `fft`, le résultat est complexe et on choisit généralement d'en afficher le module.

La fonction `fftshift` réorganise le résultat de `fft` pour afficher la fréquence nulle au centre (voir `doc fftshift`).

– Calculez et affichez le module de la transformée de Fourier d'une fonction sinusoïdale pour différentes fréquences :

```
x = 0:0.01:1;
freq_max = floor(numel(x)/2);
for w=0:freq_max
   f = sin(x*2*pi*w);
   F = fft(f);
   subplot(2,1,1)
   plot(x,f);
   subplot(2,1,2)
   plot(-freq_max:freq_max,abs(fftshift(F)));
   drawnow;
end
```

– Que se passe-t-il pour `w=freq_max` ? Que se passe-t-il si `w>freq_max` ?

– Testez également sur une combinaison linéaire de sinusoïdes de différentes fréquences.

2 Transformée de Fourier 2D

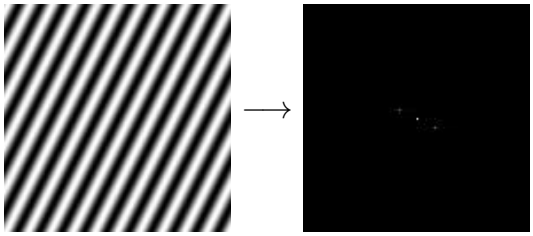
La transformée de Fourier discrète 2D est calculée en matlab par la fonction `fft2`. La fonction `fftshift` réorganise le résultat de `fft` pour afficher la fréquence nulle au centre.

Quelques remarques :

1. De la même manière que le module de la transformée de Fourier d'un signal 1D réel est symétrique par rapport à son centre (après application de `fftshift`), le module de la transformée de Fourier d'une image 2D est symétrique par rapport à son centre $(0, 0)$, après application de `fftshift`.
2. Pour (k, l) donné, la fréquence correspondant au coefficient $S(k, l)$ est d'autant plus grande que le point (k, l) est éloigné du centre

3. la direction donnée par les 2 points `(k,l)` et `(end/2+1,end/2+1)` donne la direction des oscillations correspondantes

– Calculez et affichez le module de la transformée de Fourier de l'image `test1.jpg`.



Le point central d'une transformée de Fourier correspond à la fréquence nulle. Plus précisément on peut facilement vérifier qu'il vaut la somme (ou la moyenne en fonction des définitions choisies) de l'image initiale. Il est donc généralement beaucoup plus élevé que les autres coefficients et peut nuire à la visualisation de la transformée de Fourier. Une possibilité est de visualiser en échelle logarithmique, c'est à dire d'afficher l'image `log(abs(F)+e)` au lieu de l'image `abs(F)`.

– Pour chacune des images `test*.jpg` : chargez l'image et retirez lui sa moyenne (`I = I-mean(I(:))`). Calculez et affichez les transformées de Fourier correspondantes.

– Chargez l'image `lena.png`. Calculez sa transformée de Fourier. Annulez le coefficient correspondant à la fréquence nulle (coefficient `(1,1)` avant `ifftshift` ou `(end/2+1,end/2+1)` après). Effectuez la transformée de Fourier inverse (`ifft2`). Comparez la moyenne des deux images.

```
L = double(imread('lena.png'));
FL = fft2(L);
FL(1,1)=0;
FFL = ifft2(FL);
mean(FFL(:))
```

– Annulez un coefficient de plus en plus éloigné de la fréquence nulle (ainsi que son symétrique pour s'assurer que la transformée de Fourier inverse restera à valeurs réelles). Mesurez, à l'aide de la fonction `norm` par exemple, la différence entre l'image initiale et l'image ainsi reconstruite, et observez l'évolution de cette mesure en fonction de la fréquence du coefficient modifié.

```
a=1;
L = double(imread('barb.png'));
FL = fftshift(fft2(L));
FL(end/2+1+a,end/2+1)=0;
FL(end/2+1-a,end/2+1)=0;
FFL = ifft2(ifftshift(FL));
```

3 Application à la compression d'images

La représentation fréquentielle des images est utilisée dans les standards de compression d'image (comme JPEG par exemple). L'objectif des standards de compression est de

réduire la taille des images en les détériorant le moins possible. Pour cela, ils s'appuient sur une propriété du système visuel humain : l'oeil est beaucoup moins sensible aux hautes fréquences qu'aux basses fréquences (une modification dans une image sera beaucoup plus visible sur une zone uniforme que sur une zone avec de fortes variations).

Une idée de base de la compression d'images est donc la suivante : une fois que l'on a obtenu une représentation fréquentielle de notre image, on décide de ne garder qu'un certain nombre de coefficients de celle-ci. Ce seront ces coefficients qui vont maintenant représenter notre image compressée : en effet puisqu'on a gardé moins de coefficient, la taille des données est donc plus faible.

Le choix des coefficients à garder peut se faire selon plusieurs critères.

3.1 Sélection des coefficients "basse fréquence"

Puisqu'on a vu que les coefficients "basse fréquence" avaient une influence plus grande sur la reconstruction que les coefficients "haute fréquence", une première idée naturelle consiste en ne conserver que les coefficients contenus dans un carré centré sur le point central correspondant à la fréquence nulle (après `fftshift`).

- Pour simuler cette opération, calculez la transformée de Fourier de l'image `lena.png`. Annulez tous les coefficients en dehors d'un carré de côté 32 puis appliquez la fonction `ifftshift` puis la transformée de Fourier inverse. Faites de même pour des carrés de côté 65 et 129.
- Calculez, pour chaque carré, le pourcentage de coefficients conservés. Cette valeur nous donne la compression correspondante.
- Afin de bien comprendre les modifications engendrées par cette opération, affichez, pour un carré de côté 64, une "coupe" de l'image, avant et après compression :

```
figure;
subplot(2,1,1);
plot(Avant(:,end/2));
axis tight; title('Avant');
subplot(2,1,2);
plot(Apres(:,end/2));
axis tight; title('Après');
```

3.2 Sélection des coefficients par seuillage

La sélection précédente supposait que les coefficients les plus "significatifs" étaient ceux le plus près du centre (basse fréquence). Cela peut dépendre des images. Une autre méthode consiste à sélectionner les coefficients ayant une valeur suffisamment grande et à annuler les autres.

- Gardez uniquement les coefficients de Fourier dont le module est au dessus d'un certain seuil puis effectuez la transformée de Fourier inverse.
- Choisissez le seuil pour garder le même nombre de coefficient que dans la partie précédente et comparez les résultats obtenues.
- Comparez de même avec l'image `barb.jpg`.

4 Synthèse de texture avec la transformée de Fourier

Une texture peut souvent être caractérisé par sa transformée de Fourier : deux textures du même type auront des transformées de Fourier très proche. On va voir des cas très simples de synthèse de textures générales. Le principe est de créer une transformée de Fourier vérifiant certaines propriétés, puis d'effectuer la transformée de Fourier inverse pour produire une texture correspondante aux contraintes que l'on s'impose.

4.1 Textures "nuage"

Exercice 1 Construisez une image `FI` de complexes dont le module et la phase sont tirés aléatoirement, calculez la transformée de Fourier inverse de `FI` et affichez la texture ainsi produite. La phase devra être comprise entre 0 et $2\,\pi$. De plus si celle-ci n'est pas antisymétrique, le résultat de la transformée de Fourier inverse sera complexe. Dans ce cas, n'affichez que la partie réelle du résultat.

Exercice 2 Synthétisez une texture régulière uniforme en construisant une image `FI` dont l'amplitude diminue en $(1+r)^{-\alpha}$ où r est le rayon des fréquences, $r_{k,l} = \sqrt{(k-n/2)^2+(l-n/2)^2}$, α un paramètre positif, et dont la phase est aléatoire. Testez différentes valeurs de α (e.g. $0.5, 1, 2, \ldots$).

4.2 Synthèse de texture par "randomisation" de phase

La "randomisation" de phase, qui consiste à modifier aléatoirement la phase des coefficients de fourier, garde fixé l'amplitude de la texture. Elle peut permettre de générer une texture "similaire" à une texture donnée. Cette méthode est décrite dans "B. Galerne, Y. Gousseau and J.-M. Morel, Random Phase Textures : Theory and Synthesis, preprint CMLA N2009-24, 2009".

Exercice 3 Chargez un exemple de texture. Effectuez les étapes suivantes :

- générer une phase `P` aléatoire et antisymétrique à valeur dans $[0, 2*\pi]$. La phase de la transformée de Fourier d'un bruit blanc peut être utilisée ;
- modifier la phase de la transformée de Fourier de la texture original en utilisant cette phase aléatoire. On pourra par exemple multiplier la transformée de Fourier originale par `exp(i*P)`.
- effectuer la transformée de Fourier inverse pour obtenir une nouvelle texture.

Vous pourrez tester sur les images `texture*.png`.

5 Bonus

Téléchargez et testez `InverseFFT2D_DEMO.zip` : Démo Matlab [1] permettant d'étudier l'effet des différentes composantes fréquentielles 2D. Utilisation : lancez IFFTDemo.m. Cliquez dans la partie FFT pour intégrer les composantes fréquentielles 2D.

1. `http://visiome.neuroinf.jp/modules/xoonips/detail.php?item_id=6448`