

Feuille de TP numéro 1

1 Consignes pour le rapport de TP

Cette feuille de TP comporte des questions auxquelles il est demandé de répondre dans un rapport de TP individuel. **Chaque jour de retard dans la remise du rapport entraînera une pénalisation de 2 points par jour.**

1.1 Format du rapport

Les rapports de TP sont à rendre sous forme de fichier .pdf rédigés avec \LaTeX . \LaTeX est le langage standard **incontournable** pour rédiger des documents en mathématiques (et dans bien d'autres matières scientifiques). Pour vous aidez dans la rédaction de ce rapport un modèle de fichier \LaTeX (extension .tex) vous pouvez le télécharger de http://dev.ipol.im/~facciolo/hilbert/modele_latex.zip. Ce fichier sera accompagné des conseils concernant les logiciels à utiliser ainsi que des liens internet vers des documentations.

1.2 Remise du rapport

Le rapport de ce TP doit être rendu pour le **lundi 19 novembre 2012**. Votre rapport, au format .pdf, sont à envoyer par email à l'adresse **gabriele.facciolo@cmla.ens-cachan.fr**. Le sujet de l'email doit être **Rapport du TP1 de Nom Prénom** et le fichier .pdf doit être intitulé **nom_prenom_tp1.pdf**.

2 Mise en route

Tout d'abord ouvrez un terminal (fenêtre permettant d'entrer des lignes de commandes, accessible depuis le menu : Menu KDE -> Applications -> Systèmes -> Terminal). Dans ce terminal tapez les commandes suivantes pour créer un dossier `tp_fourier` et se placer dans ce dossier :

```
mkdir tp_fourier
cd tp_fourier
```

Dans le même terminal tapez maintenant les commandes suivantes pour télécharger les fichiers nécessaires dans votre dossier `tp_fourier`.

```
wget dev.ipol.im/~facciolo/hilbert/tp1.zip
unzip tp1.zip
cd tp1
```

Toujours dans le même terminal lancez Matlab grâce à la commande `matlab`. Vous écrirez les différentes instructions dans des fichiers du type `mon_script.m` et vous lancerez les différentes instructions en surlignant avec la souris les passages à évaluer et en appuyant sur la touche F9 (ou alors clique droit et `Evaluate selection`).

3 Notions de base sur les images numériques

Nous travaillerons exclusivement avec des images en niveaux de gris.

3.1 Niveaux de gris

Une image numérique standard est un tableau de pixels (pouvant être vu comme une matrice), dont les coefficients sont des entiers compris entre 0 et 255. Il n'y a donc que 256 valeurs possibles pour les niveaux de gris. Il est très facile de créer des images simples avec Matlab. Par exemple le script suivant crée une image de taille 500×500 représentant un carré noir sur fond blanc :

```
u = 255*ones(500,500);
u(201:400,101:300) = zeros(200,200);
```

Pour visualiser une image `u` (ou n'importe quelle tableau) nous utiliserons les instructions suivantes (le problème de la visualisation d'une image est abordée en détail dans la partie suivante) :

```
imagesc(u, [0,255]);
colormap gray;
axis image;
```

- **Question 1 :** Créez une image de taille 256×256 qui soit constante sur chaque colonne et dont le niveau de gris des colonnes varie progressivement de 0 à 255.

Aide Matlab : Si `a` et `b` sont deux entiers alors `(a:b)` est le vecteur ligne `[a, a+1, ..., b-1, b]`.

- **Question 2 :** Quoi représente l'image suivante ?

0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	120	120	120	120	120	0	0
0	0	120	255	255	255	120	0	0
0	0	120	255	255	255	120	0	0
0	0	120	255	255	255	120	0	0
0	0	120	120	120	120	120	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0

3.2 Visualisation des images

3.2.1 Plage de valeurs

La manipulation mathématique des images amène à considérer des images dont les niveaux de gris sont des nombres réels (ou du moins la représentation informatique des nombres réels,

c'est-à-dire des nombres flottants). En effet, toutes les images ne peuvent pas être limités à des valeurs entières entre 0 et 255. Néanmoins il faut garder à l'esprit que toutes les images affichées à l'écran sont des images standards qui respectent ces contraintes.

Il y a alors deux méthodes pour visualiser une image u à valeur réelle. Soit l'image u est proche d'une image standard et alors on arrondit chaque valeur $u(i, j)$ à l'entier le plus proche entre 0 et 255 (ce que réalise la fonction Matlab `uint8`). Soit on effectue un changement de variable affine qui ramène l'intervalle $[\min(u), \max(u)]$ sur l'intervalle $[0, 255]$, et on effectue ensuite une discrétisation des valeurs avec `uint8`. Cette deuxième méthode est particulièrement adaptée pour visualiser des images dont la plage de valeurs n'est pas standard (comme la somme de deux images par exemple).

3.2.2 Lecture et écriture des fichiers d'images

La fonction Matlab `imread` permet de lire des images enregistrées sous forme de fichier sur l'ordinateur :

```
u = imread('tile.png');
```

Ainsi obtenue u à des valeurs de type `uint8` (*i.e.* des entiers compris entre 0 et 255). Pour pouvoir faire des calculs sur cette image, il faut la convertir en nombre flottant grâce à la fonction `double` :

```
u = double(u);  
% ou directement  
u = double( imread('tile.png') );
```

Pour enregistrer une image u dans un fichier d'image il faut faire la conversion inverse et utiliser la fonction `imwrite` :

```
imwrite(uint8(u), 'nouvelle_image.png');
```

ATTENTION : Matlab écrase les fichiers existants sans prévenir !

Alternativement, pour enregistrer le contenu d'une fenêtre, vous pouvez aussi utiliser l'option Enregistrer sous.

3.2.3 Visualisation des images avec Matlab

imagesc : La fonction qu'on utilisera pour visualiser des images avec Matlab est `imagesc(im, [rang])`. Le paramètre `im` est le nom d'une variable contenant un tableau à valeurs réelles ou entières. Le paramètre optionnel `rang` indique la plage de valeurs à montrer ; Pour exemple si `rang = [-10, 10]`, la valeur -10 sera représenté avec la couleur noire, la valeur 10 avec blanche et les valeurs dans l'intervalle avec des tonnes de gris (si `colormap` est "gray"), toutes les valeurs plus petites que -10 ou plus grandes que 10 sont saturés donc montrées comme noire ou blanche respectivement. Si le paramètre `range` n'est pas spécifié, la gamme sera établi a $[\min(im(:)), \max(im(:))]$. Pour plus d'information `help imagesc`.

colormap : La gamme de couleurs utilisées pour l’affichage de l’image s’appelle `colormap`, le colormap par défaut pour chaque nouvelle fenêtre est “jet” (couleurs d’arc en ciel), le commande `colormap gray`; nous permet de montrer les images avec une gamme de niveau de gris. Pour plus d’information `help colormap`.

- **Question 3** : Chargez l’image `water.png` dans la variable `im`, et testez les commandes suivantes :

```
imagesc(im); colormap gray; axis image
imagesc(im,[0,255]); colormap gray; axis image
imagesc(im,[0, 100]); colormap gray; axis image
imagesc(im,[0,255]); colormap jet; axis image
```

Comment expliquez-vous les différences observées entre les visualisations ?

3.2.4 Visualisation conservant son aspect originale

axis : Le commande `axis` contrôle l’aspect des axes, concrètement : `axis image` rend les pixels carrés. Pour plus d’information `help axis`.

Essayez avec l’exemple ci-dessous :

```
u = double(imread('water.png'));
imagesc(u);
colormap gray;
axis image;
axis off;
```

4 Transformée de Fourier discrète (TFD)

4.1 En dimension 1

- **Question 4** : Lire l’aide de la commande `fft` (`help fft` où `doc fft`). Quelles sont les différences par rapport à la définition du cours ?

Toutes ces différences sont des problèmes de conventions qui ne changent pas les propriétés quantitatives de la TFD. Par soucis de simplicité on utilisera donc la définition de Matlab pour ces TP.

Créez un signal discret u de longueur $N = 512$ représentant un créneau :

```
% Longueur du signal :
N=256;
ind = 1:N;
% Creation du signal :
u = zeros(1,N);
u( (N/4+1) : (3*N/4) ) = ones(1,N/2);
% Affichage du signal :
figure(1);
plot(ind, u, 'b*-'); axis([0, N, -0.1, 1.1]);
```

- **Question 5** : Calculez le module de Fourier modu du signal `u` (utilisez `abs`) et visualisez-le avec la commande `plot(ind, modu); axis tight;`. Le graphique obtenu est-il conforme à vos attentes ? Recommencez en utilisant la fonction `fftshift`.

4.2 En dimension 2

4.2.1 TFD et TFD inverse

La TFD en dimension 2 est donnée par la fonction `fft2`, la transformation inverse est donnée par `ifft2`. Lisez la section “Algorithm” de l’aide en ligne de cette fonction. Remarquez que la propriété de séparabilité de la transformation est mise en œuvre.

Remarque : Il est possible que la TFD inverse de la TFD d’une image réelle ait une “petite partie” imaginaire (à cause des erreurs d’arrondi). Dans ce cas il faut extraire la partie réelle avec la commande `real`.

4.2.2 Visualisation du module

Pour visualiser le module d’une image nous utiliserons la fonction `vis_mod_img` donnée dans l’archive. Pour voir l’aide de la commande : `help vis_mod_img`.

- **Question 6 :** Lisez le fichier `vis_mod_img.m`. En utilisant la fonction `subplot`, visualisez sur une même figure le module de l’image `room.png` avec les trois jeux de paramètres $(d, opt) = (0, 0), (1, 0), (1, 1)$. Expliquez l’intérêt du seuillage. Quelle propriété de symétrie du module peut-on observer ?

On ne visualisera pas la phase d’une image car son interprétation est assez difficile. Néanmoins, comme le montre les expériences qui suivent, la phase d’une image est au moins aussi importante que son module du point de vue de l’information qu’elle contient.

4.2.3 Echange du module et de la phase de deux images

Calculez et visualisez les deux images obtenues en échangeant le module et la phase des images `building.png` et `tile.png`. Recommencez cette expérience avec les couples d’images $(\text{sand.png}, \text{water.png})$ et $(\text{tile.png}, \text{sand.png})$.

- **Question 7 :** Quelles conclusions générales peut-on tirer de ces expériences sur le contenu visuel du correspondant respectivement au module et à la phase d’une image ? (On distinguera deux types de “contenu visuel” : les contours géométriques et les textures.)

5 Effet de Gibbs et zooms avant

5.1 Polynôme trigonométrique associé à un signal discret

Soit $u_k, k = 0, \dots, N - 1$ un signal discret et $\tilde{u}_n, n = -\frac{N}{2}, \dots, \frac{N}{2} - 1$ sa TFD. On suppose que N est un entier pair. On rappelle que le polynôme trigonométrique associé à u est le polynôme P_u défini par

$$P_u(x) = \sum_{n=-\frac{N}{2}}^{\frac{N}{2}-1} \tilde{u}_n e^{\frac{2i\pi nx}{N}}.$$

Le polynôme P_u vérifie $P_u(k) = u_k$ pour tout $k = 0, \dots, N - 1$. Alors P_u est un polynôme trigonométrique qui interpole les valeurs du signal u . En revanche, P_u n’est pas nécessairement

à valeur réelle ce qui n'est pas satisfaisant en pratique.

- **Question 8 :** Montrer que P_u est à valeur réelle si et seulement si $\tilde{u}_{-\frac{N}{2}} = 0$. Noter également que pour tout signal u , la partie réelle de P_u est un polynôme trigonométrique qui interpole les valeurs de u .

En pratique, on utilise la partie réelle de P_u pour interpoler le signal u . Il en est de même en dimension 2.

5.2 Effet de Gibbs sur les images

L'effet de Gibbs est le phénomène suivant : les projections basses fréquences d'un signal présentant des discontinuités ont des comportements oscillants autour des discontinuités. Nous allons mettre en évidence ce phénomène à partir d'une image discrète (voir le polycopié pour le cas des signaux unidimensionnels).

Créez une image `rectangle` de taille 512×512 représentant un rectangle gris clair (niveau de gris 200) sur un fond gris foncé (niveau de gris 50), voire la section 3.1. Calculez la TFD de l'image `rectangle`. Annulez toutes les fréquences qui ne sont pas dans le carré central de taille $n \times n$ avec $n = 64$ et calculez l'image `rectbf` correspondant à ce spectre ne comportant que des basses fréquences. On pourra visualiser cette image en niveau de gris mais aussi en tant que surface grâce aux commandes suivantes :

```
surf(rectbf, 'EdgeColor', 'none');  
colormap(jet); colorbar; axis tight;
```

- **Question 9 :** Commentez les résultats observés.

5.3 Zooms avant

Le fichier d'instructions `zooms_avant.m` vous permet de comparer trois méthodes pour zoomer une image. La première est la méthode "pixel le plus proche". La deuxième repose sur l'interpolation bilinéaire. La troisième est le zoom par zéro-padding (extension du spectre avec des hautes fréquences nulles).

- **Question 10 :** Testez les zooms sur plusieurs images. Décrivez brièvement les défauts de chacune des méthodes. Expliquez le lien entre les défauts du zoom par zéro-padding et l'effet de Gibbs.

6 Translation non entière et rotation

6.1 Translation non entière

Soit $(u_{k,l})$ une image carrée de taille $N \times N$ et $(\tilde{u}_{m,n})$ sa TFD. On pose

$$P_u(x, y) = \sum_{m=-\frac{N}{2}}^{\frac{N}{2}-1} \sum_{n=-\frac{N}{2}}^{\frac{N}{2}-1} \tilde{u}_{m,n} e^{\frac{2i\pi mx}{N}} e^{\frac{2i\pi ny}{N}}$$

le polynôme trigonométrique représenté par u .

- **Question 11 :** Soit a et b deux réels. Exprimez la TFD de $(k, l) \mapsto P_u(k-a, l-b)$ en fonction des $\tilde{u}_{m,n}$. Codez dans la fonction $v = \text{translation_fourier}(u, a, b)$ un algorithme qui calcule la translation de l'image u par le vecteur (a, b) (**Aide Matlab :** l'opération multiplication coordonnées par coordonnées de deux matrices A et B s'écrit $A .* B$). Cette fonction retournera la partie réelle de la translation de P_u , en accord avec les remarques de la section 5.1. **Testez votre fonction en faisant une translation sous-pixelienne de (3.3 ; 50.7) pixels d'une image.**

6.2 Rotation d'une image (*devoir à la maison*)

L'algorithme de rotation d'une image est basé sur la décomposition suivante :

$$R_{-\theta} := \begin{pmatrix} \cos \theta & \sin \theta \\ -\sin \theta & \cos \theta \end{pmatrix}$$

$$\begin{pmatrix} \cos \theta & \sin \theta \\ -\sin \theta & \cos \theta \end{pmatrix} = \underbrace{\begin{pmatrix} 1 & \tan \frac{\theta}{2} \\ 0 & 1 \end{pmatrix}}_{:=T(\theta)} \underbrace{\begin{pmatrix} 1 & 0 \\ -\sin \theta & 1 \end{pmatrix}}_{:=S(\theta)} \begin{pmatrix} 1 & \tan \frac{\theta}{2} \\ 0 & 1 \end{pmatrix}$$

$$R_{-\theta} := T_{\theta} S_{\theta} T_{\theta}$$

Remarquez que le changement de variable donné par $T(\theta)$ revient à traduire chaque colonne : $u(k, l)$ devient $u(k + \tan(\frac{\theta}{2})l, l)$. De même, le changement de variable donné par $S(\theta)$ revient à traduire chaque ligne : $u(k, l)$ devient $u(k, l - \sin(\theta)k)$.

Les consignes qui suivent ont pour but de coder l'algorithme de rotation.

- **Question 12 :** Codez une fonction $v = \text{translation_colonnes}(u, t)$ qui translate chaque colonne k de u par le vecteur $t(k)$ en utilisant la TFD unidimensionnelle (**Aide Matlab :** si A est une matrice, alors $\text{fft}(A)$ renvoie la matrice des TFD des colonnes de A). Testez cette fonction avec des vecteurs de translation non entiers.
- **Question 13 :** Codez une fonction $v = \text{translation_lignes}(u, t)$ qui translate chaque ligne k de u par le vecteur $t(k)$. On va utiliser la fonction $\text{translation_colonnes}$ et les transpositions de matrice (**Aide Matlab :** la transposée de A est A').
- **Question 14 :** Coder la fonction $v = \text{rotation_fourier}(u, \theta)$ en effectuant trois appels successifs aux fonctions $\text{translation_colonnes}$ et $\text{translation_lignes}$. Commencez par copier l'image u au centre d'une image nulle deux fois plus grande pour éviter les problèmes de bords. Imposez à chacune des translations de ne pas déplacer la colonne (ou la ligne) du centre de l'image. Illustrez la méthode en visualisant chacune des images intermédiaires. **Testez votre fonction en faisant une rotation de 13° d'une image.**

7 Zoom arrière et aliasing (*devoir à la maison*)

Le fichier d'instructions `aliasing.m` illustre le phénomène d'aliasing ou de repliement du spectre. Suivre les instructions dans le fichier et répondre aux questions suivantes.

- ▶ **Question 15 :** Dans la partie 1). Qu'observe-t-on ? Comment cela se traduit-il au niveau du spectre ?
- ▶ **Question 16 :** Dans la partie 2). Expliquez précisément le comportement des deux ondes sous-échantillonnées. Quelle est la fréquence de l'onde de l'image sous-échantillonnée par 4 ? Qu'observe-t-on ? Comment cela se traduit-il au niveau du spectre ?

Pour éviter le phénomène de repliement du spectre on met à zéro les hautes fréquences avant de sous-échantillonner.