

## TP 2 : Transformée de Fourier et restauration d'image

Dans ce TP 2, nous allons nous intéresser à la problématique de la restauration d'image. Lorsque l'on fait une acquisition, l'image de la scène observée ne correspond pas au reflet parfait de la scène : l'image est dégradée par la réponse impulsionnelle de l'instrument utilisé pour l'acquisition (on parle de PSF pour *point spread function*) et par la superposition de bruit dû à l'environnement. On a vu en cours différentes techniques de restauration d'image, que ce soit des opérations de déconvolution ou de débruitage. Dans ce TP, on s'intéressera uniquement aux opérations de déconvolution. Que ce soit la technique de déconvolution naïve ou le filtrage de Wiener, ces opérations font intervenir les transformées de Fourier des images et de la PSF du système d'acquisition, c'est pourquoi la première partie de ce TP sera consacrée à l'analyse des propriétés de la transformée de Fourier bidimensionnelle.

### Quelques rappels de fonctions utiles sous Matlab

```
>> I = imread('image.png'); # Chargement d'une image

>> Igris = rgb2gray(I); # Transformation d'une image RGB en niveaux de
gris.

>> Igris = double(Igris); # Transformation de l'image codée en uint8 en
double.

>> [M,N] = size(Igris); # Calcul de la taille de l'image et stockage dans
les variables M et N.

>> I(i,:); # Sélection de la ligne d'indice i de l'image I.

>> I(:,j); # Sélection de la colonne d'indice j de l'image
I.

>> I(i1:i2, j1:j2); # Sélection de la partie de l'image comprise entre
les lignes i1 et i2 et les colonne j1 et j2.

>> imshow(I, [ ]); # Affichage de l'image I, ne pas oublier l'option
[ ] dans le cas où I est codée en double.

>> TF = fftshift(fft2(Igris)); # Calcul de la transformée de Fourier de Igris
avec la fréquence spatiale (0,0) au centre du module et de la phase de la TF.

>> real(A); # Partie réelle du nombre complexe A.
```

# 1 Rappels théoriques sur la déconvolution

## 1.1 Filtrage dans le domaine de l'image

Le filtrage d'une image  $\mathbf{I}$  de taille  $M \times N$  par une image  $\mathbf{H}$  de taille  $P \times Q$  s'exprime à l'aide d'une convolution :

$$(\mathbf{I} * \mathbf{H})(x, y) = \sum_{i=0}^{P-1} \sum_{j=0}^{Q-1} \mathbf{H}(i, j) \mathbf{I}(x - i, y - j)$$

On a vu que la convolution de deux images de tailles  $M \times N$  et  $P \times Q$  donne une image finale de taille  $(M + P - 1) \times (N + Q - 1)$  en tenant compte des problèmes aux bords. Lorsque l'on utilise la fonction `conv2` de Matlab, il est possible d'utiliser trois options différentes pour tronquer l'image convolué à la taille souhaitée :

- 'full' : l'image retournée par `conv2` est de taille  $(M + P - 1) \times (N + Q - 1)$ ,
- 'same' : l'image retournée par `conv2` est de taille  $M \times N$ ,
- 'valid' : l'image retournée par `conv2` est de taille  $(M - (P - 1)) \times (N - (Q - 1))$ .

Par défaut la convolution est implémentée au sens de l'option 'full' qui permet de rendre compte de "l'étalement" de l'image  $\mathbf{I}$  par le filtre  $\mathbf{H}$ .

## 1.2 Filtrage dans le domaine de Fourier

Soient  $f$  et  $g$  deux signaux<sup>1</sup> continus à support infini. La transformée de Fourier (TF) de  $f * g$  est égale au produit des transformées de Fourier :

$$TF(f * g) = TF(f) \times TF(g). \quad (1)$$

Dans le cas d'un signal discret à support borné (c'est le cas des images numériques), la transformée de Fourier discrète (TFD) est elle-même un signal discret et le spectre de l'image se retrouve périodisé (TF d'un signal échantillonné = périodisation en fréquence). De même le calcul de la  $TDF^{-1}$  se retrouve périodisé puisque l'on utilise une période (de la même taille que l'image originale) de la TFD pour la calculer. Il faut donc garder en tête ce problème de périodisation dans le domaine de Fourier et dans le domaine de l'image lorsque l'on effectue le filtrage dans le domaine fréquentiel.

Sans rentrer dans les détails de calcul<sup>2</sup>, afin d'avoir une équivalence entre le filtrage dans le domaine de l'image et le filtrage dans le domaine de Fourier, il faut compléter les images par des zéros (*zero-padding*) avant de passer dans le domaine de Fourier comme illustré sur la figure 1. On prendra soin d'effectuer les opérations de zero-padding des images et des filtres ainsi que de troncature du résultat final dans le TP.

## 1.3 Déconvolution naïve par filtrage inverse

Lors de l'opération de déconvolution, on cherche à inverser la dégradation modélisée par le filtre  $\mathbf{H}$ . Le filtrage inverse se fait dans le domaine de Fourier :

$$\hat{\mathcal{I}} = \frac{\mathcal{J}}{\mathcal{H}}$$

où

- $\mathcal{J}$  est la transformée de Fourier de l'image dégradée  $\mathbf{J}$ .
- $\mathcal{H}$  est la transformée de Fourier de la dégradation  $\mathbf{H}$ .
- $\hat{\mathcal{I}}$  est la transformée de Fourier de l'image restaurée  $\hat{\mathbf{I}}$ .

---

1. Signaux de dimensions quelconques : 1D, 2D, ...

2. Se reporter à la section 4.6.3 de *Digital Image Processing*, 2nd edition, R. Gonzalez et R. Woods pour plus de détails.

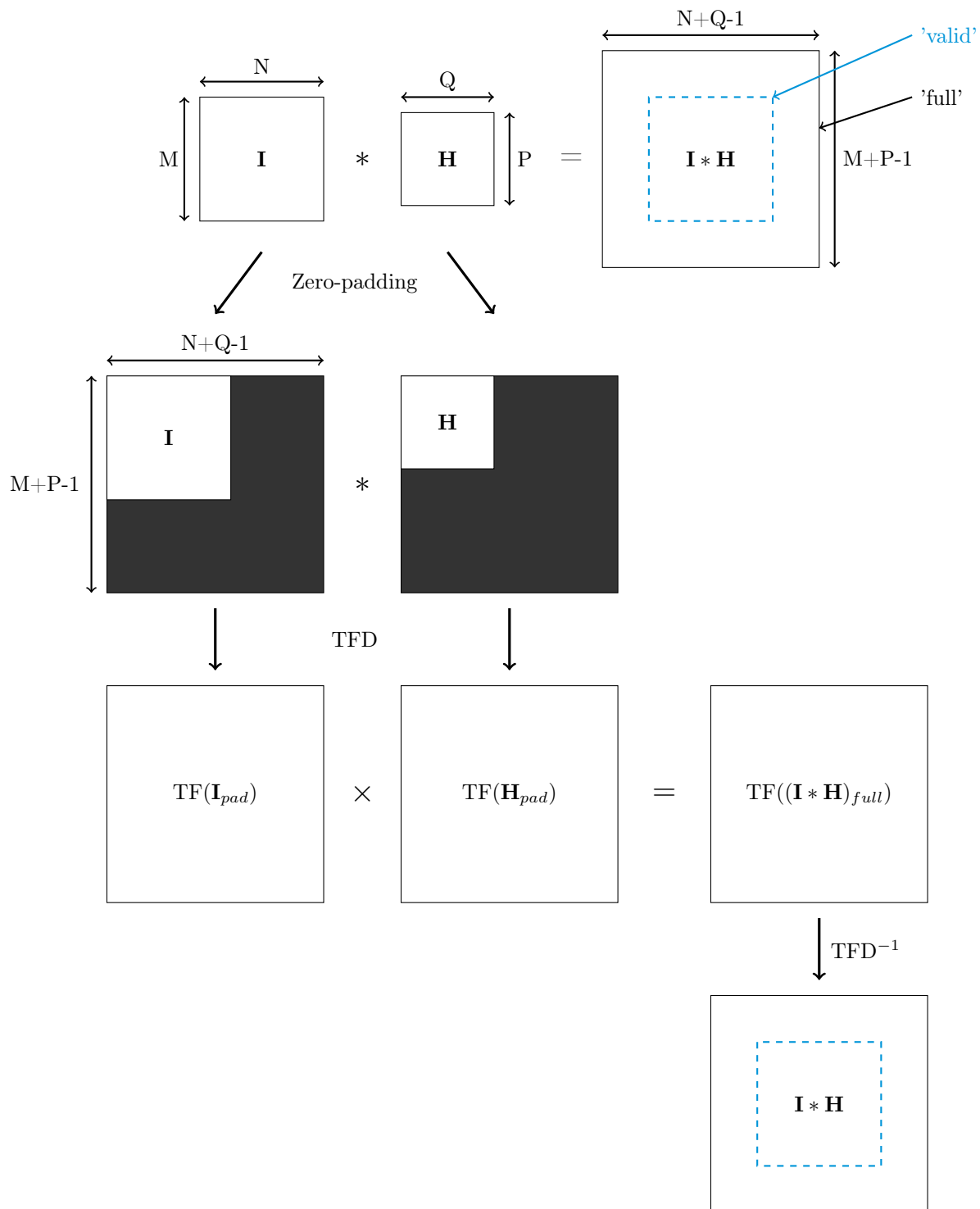


FIGURE 1 – Illustration du zero-padding nécessaire au calcul du filtrage dans le domaine de Fourier. Les parties grisées des images correspondent à des opérations de zéro-padding.

En général, on a seulement la partie valide ('valid' en pointillés bleus sur la figure 1) de  $\mathbf{J} = \mathbf{I} * \mathbf{H}$  et le filtre  $\mathbf{H}$ . Afin d'inverser le filtrage, il manque donc les informations situées autour de la partie valide de  $\mathbf{J}$ . Afin de gérer ces problèmes de bords, il faut donc appliquer l'une des quatre solutions proposées dans le cours (zéro-padding, miroir, duplication des bords, ou périodisation) pour former une image de taille  $(M + P - 1) \times (N + Q - 1)$  avant de calculer sa TFD et de la diviser par la TFD du filtre  $\mathbf{H}$ . Les opérations nécessaires au filtrage inverse sont illustrées sur la figure 2.

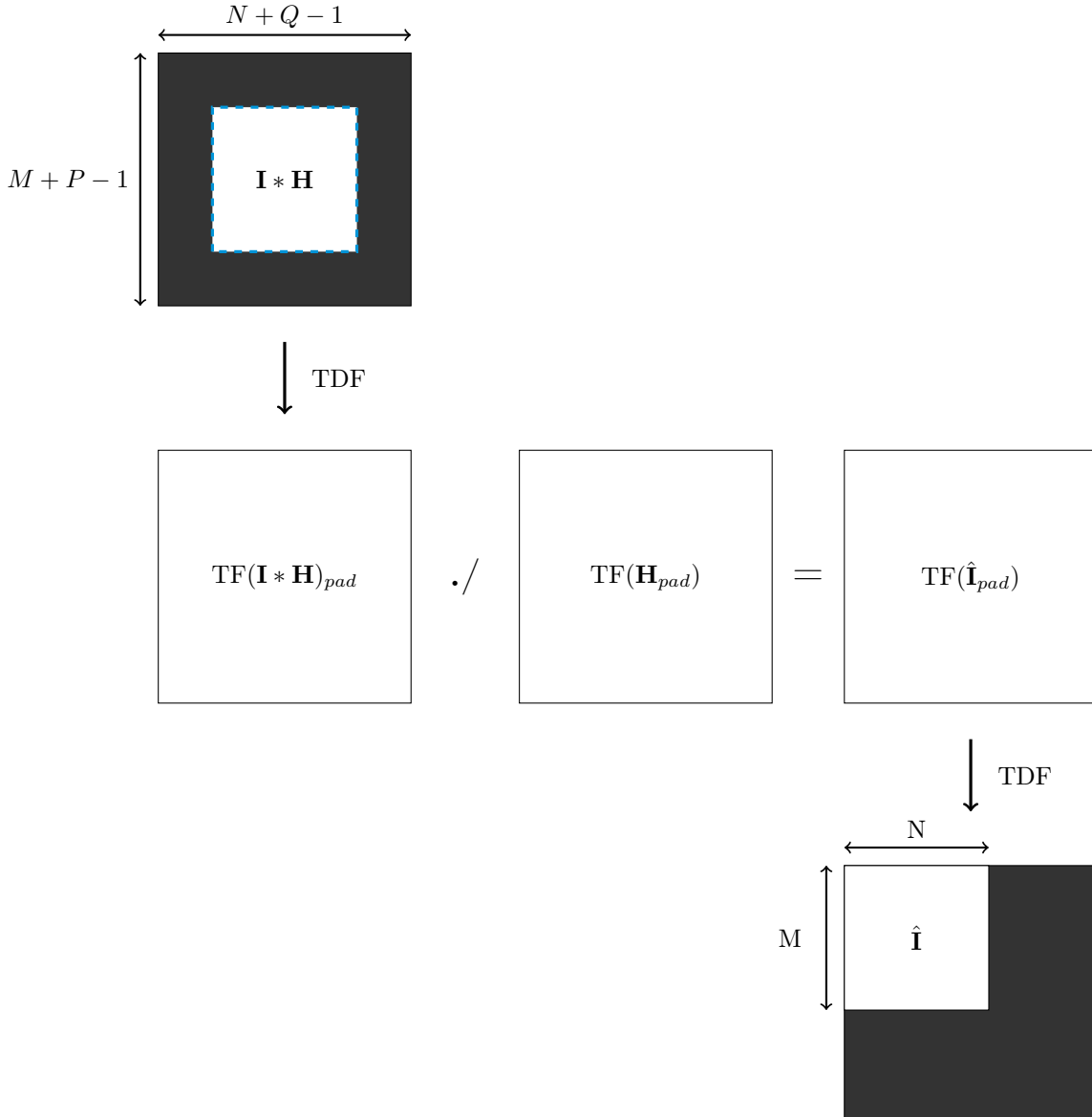


FIGURE 2 – Illustration des opérations de filtrage inverse dans le domaine de Fourier. Les parties grisées des images correspondent à des opérations de zéro-padding.

## 1.4 Filtrage de Wiener

On rappelle l'expression vue en cours du filtre de Wiener  $\mathcal{G}$  dans le domaine de Fourier :

$$\mathcal{G} = \frac{\mathcal{H}^* S_{\mathbf{I}}}{|\mathcal{H}|^2 S_{\mathbf{I}} + S_{\mathbf{B}}} = \frac{\mathcal{H}^*}{|\mathcal{H}|^2 + \frac{S_{\mathbf{B}}}{S_{\mathbf{I}}}} \quad (2)$$

où :

- $\mathcal{H}$  est la transformée de Fourier de la dégradation  $\mathbf{H}$
- $\mathcal{I}$  est la transformée de Fourier de l'image  $\mathbf{I}$
- $\mathcal{H}^*$  est le conjugué de  $\mathcal{H}$
- $|\mathcal{H}|^2 = \mathcal{H}^* \mathcal{H}$  est le carré du module de  $\mathcal{H}$
- $S_{\mathbf{I}} = |\mathcal{I}|^2$  est la densité spectrale de puissance de l'image
- $S_{\mathbf{B}} = |\mathcal{B}|^2$  est la densité spectrale de puissance du bruit

On rappelle que toutes les opérations (addition, division, etc) se font pixel à pixel.

La constante  $\frac{S_{\mathbf{B}}}{S_{\mathbf{I}}}$  est souvent inconnue, on la remplacera par une constante  $K$  qu'il faudra régler au mieux pour déconvoluer les images.

## 2 Propriétés de la transformée de Fourier (2h)

On rappelle que la transformée de Fourier discrète d'une image de taille  $M \times N$  s'écrit :

$$\mathcal{I}(u, v) = \frac{1}{MN} \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} I(i, j) e^{-j 2\pi \left( \frac{ui}{M} + \frac{vj}{N} \right)}$$

### 2.1 Base de décomposition de Fourier

✦ On peut montrer que la transformée de Fourier est un produit scalaire entre l'image et une base d'images que l'on explicitera.

#### Manipulation 1

---

➡ Créer les images de la base qui servent à calculer la transformée de Fourier. On testera quelques valeurs de  $u$  et de  $v$  en particulier en fixant  $u = 0$  et en faisant varier  $v$  dans son domaine d'existence et inversement en fixant  $v = 0$  et en faisant varier  $u$  dans son domaine d'existence. On travaillera avec des images de taille  $M = N = 64$ .

➡ Pour chaque couple de valeurs  $(u, v)$  afficher la partie réelle, la partie imaginaire, le module et la phase.

---

### 2.2 Transformations d'une image et conséquences sur sa transformée de Fourier

On va maintenant mettre en évidence différentes propriétés de la transformée de Fourier lors de transformations simples d'une image synthétique (translation, homothétie).

---

**Manipulation 2**

---

- ➔ Créer une image synthétique simple de taille  $64 \times 64$  contenant des zéros partout et un rectangle centré de taille  $32 \times 16$  contenant la valeur 1.
  - ➔ Calculer la transformée de Fourier de cette image et afficher son module et sa phase.
  - ➔ Créer une image où l'on translate le rectangle verticalement de 8 pixels. Calculer sa transformée de Fourier et afficher le module et la phase de la nouvelle image.
  - ➔ Créer une image où l'on translate le rectangle horizontalement de 8 pixels. Calculer sa transformée de Fourier et afficher le module et la phase de la nouvelle image.
  - ➔ Créer une image où le rectangle a subi une homothétie horizontalement d'un facteur  $1/2$ . Calculer sa transformée de Fourier et afficher le module et la phase de la nouvelle image.
- 

⚡ Analyser les différentes propriétés de la transformée de Fourier dans le cas des différentes modifications de l'image.

### 2.3 Cas d'une image séparable selon les lignes et selon les colonnes

L'image  $I(x, y)$  créée précédemment est séparable en une image  $I_x(x, \cdot)$  ne variant que selon  $x$  et une image  $I_y(\cdot, y)$  ne variant que selon  $y$ , l'image originale étant un produit terme à terme de ces deux images :

$$I(x, y) = I_x(x, \cdot) \times I_y(\cdot, y)$$

- ⚡ Trouver les deux images ne variant respectivement que selon  $x$  et que selon  $y$ .
- ⚡ Montrer que la transformée de Fourier de l'image originale est séparable en une transformée de Fourier uniquement selon les lignes et une transformée de Fourier uniquement selon les colonnes.

---

**Manipulation 3**

---

- ➔ Créer les deux images formant l'image originale de la partie précédente (image synthétique simple de taille  $64 \times 64$  contenant des zéros partout et un rectangle centré de taille  $32 \times 16$  contenant la valeur 1).
  - ➔ Calculer la transformée de Fourier de ces deux images (selon les lignes et selon les colonnes, pour une fois, la boucle `for` est autorisée).
  - ➔ Comparer à la transformée bidimensionnelle de l'image originale.
- 

⚡ Conclure.

## 3 Filtrage dans le domaine de Fourier (1h30min)

Cette partie du TP permet de mettre en pratique le filtrage dans le domaine de Fourier et de bien comprendre les opérations illustrées dans la figure 1.

Afin d'effectuer les opérations de zéro-padding nécessaires pour réaliser le filtrage dans le domaine de Fourier, il faut créer une matrice  $M$  remplie de zéros ( $M = \text{zeros}(A, B)$ ) et remplacer les valeurs de  $M$  par celles de l'image que l'on souhaite agrandir ( $M(i1 : i2, j1 : j2) = I$ ).

---

#### Manipulation 4

- ➔ Charger l'image 'moon.tif' qui est disponible par défaut dans la bibliothèque Matlab.
  - ➔ Créer un filtre moyenneur  $\mathbf{H}$  de taille  $P = 31$  et  $Q = 51$ .
  - ➔ Effectuer la convolution dans le domaine de l'image avec la fonction `conv2` et l'option 'full'.
  - ➔ Calculer la transformée de Fourier discrète de l'image 'moon.tif' et du filtre  $\mathbf{H}$ . On prendra soin d'effectuer les opérations de zero-padding avant d'utiliser la fonction `fft2`.
  - ➔ Effectuer le filtrage dans le domaine de Fourier et calculer la transformée de Fourier discrète inverse du résultat obtenu.
  - ➔ On comparera les résultats de la convolution dans le domaine de l'image et dans le domaine de Fourier en affichant les résultats et en affichant leur différence.
- 

⚡ Commenter les résultats obtenus : peut-on considérer que les deux approches donnent des résultats similaires ?

## 4 Déconvolution naïve par filtrage inverse (1h30)

Nous allons mettre en oeuvre dans cette partie la déconvolution par filtrage inverse sur un cas de dégradation d'image quasi réel. Au lancement du télescope spatial Hubble, l'optique du télescope comportait une anomalie qui n'avait pas été détectée sur Terre. Lors de l'acquisition des premières images dans l'espace, les astronomes ont découvert avec stupéfaction la présence de cette anomalie : les images étaient déformées et floues. L'anomalie a depuis été corrigée et le télescope Hubble produit des images bien résolues. La PSF du télescope correspondant à cette anomalie a été estimée, elle est stockée dans le fichier 'Hubble\_psf.jpg'. On utilise une image de galaxie observée par Hubble après la correction de l'anomalie comme image "avant dégradation". On a dégradé l'image par convolution avec la PSF de l'anomalie et on obtient l'image stockée dans le fichier 'blurred\_image.mat'.

---

#### Manipulation 5

- ➔ Charger l'image originale (qui servira de vérité terrain pour la comparaison) contenue dans le fichier 'ground\_truth.mat' avec la commande :  

```
>> load('ground_truth.mat','truth')
```
  - ➔ Charger l'image dégradée contenue dans le fichier 'blurred\_image.mat' avec la commande :  

```
>> load('blurred_image.mat','blurred_image')
```
  - L'image dégradée est stockée dans la variable `blurred_image`.
  - ➔ Afficher cette image avec `imshow`.
  - ➔ Charger l'image de la PSF avec la commande :  

```
>> load('Hubble_psf.mat','psf')
```
  - ➔ Effectuer les différents zéro-padding sur la PSF et sur l'image dégradée conformément aux rappels théoriques donnés dans les paragraphes 1.2 et 1.3.
  - ➔ Déconvoluer l'image à l'aide du filtre inverse dans le domaine de Fourier.
  - ➔ Calculer la transformée de Fourier inverse et tronquer convenablement le résultat. Afficher l'image déconvoluée.
  - ➔ Ajouter du bruit gaussien à l'image `blurred_image` et recommencer l'opération de déconvolution par filtrage inverse sur l'image dégradée et bruitée. On choisira une valeur de RSB de 60dB.
  - ➔ Evaluer la qualité de la restauration avec les différents critères vus en cours.
- 

Une image de bruit gaussien à RSB fixé se crée à partir des commandes suivantes :

```
>> RSB = 60; % en dB
>> sigma2 = mean((blurred_image(:).^2))/10^(RSB/10); % Calcul de la puissance du
bruit (variance)
>> noise = randn(M, N)*sqrt(sigma2); % création d'une matrice de taille M x N
contenant du bruit.
```

⚡ Commenter les résultats obtenus sur l'image dégradée avec ou sans bruit.

## 5 Filtrage de Wiener (1h30)

On va maintenant s'intéresser à la déconvolution par filtrage de Wiener sur la même image dégradée que précédemment.

### Manipulation 6

---

➔ Charger l'image originale (qui servira de vérité terrain pour la comparaison) contenue dans le fichier 'ground\_truth.mat' avec la commande :

```
>> load('ground_truth.mat','truth')
```

➔ Charger l'image dégradée contenue dans le fichier 'blurred\_image.mat' avec la commande :

```
>> load('blurred_image.mat','blurred_image')
```

L'image dégradée est stockée dans la variable `blurred_image`.

➔ Charger l'image de la PSF avec la commande :

```
>> load('Hubble_psf.mat','psf').
```

➔ Effectuer les différents zéro-padding sur la PSF et sur l'image dégradée conformément aux rappels théoriques donnés dans les paragraphes 1.2 et 1.3.

➔ Implémenter le filtre de Wiener dans le domaine de Fourier (équation (2)).

➔ Ajouter du bruit gaussien à l'image `blurred_image`. On choisira une valeur de RSB de 60dB.

➔ Appliquer le filtre de Wiener sur l'image dégradée et bruitée et évaluer la qualité de la restauration avec les différents critères vus en cours. Le filtre de Wiener dépend d'un paramètre  $K$ , il faut donc tester plusieurs valeurs de  $K$  pour obtenir le meilleur résultat possible. On testera  $K = 10, 1000, 5000, 7000, 10000$ .

---

**A noter :** Une grande partie des opérations (zéro-padding, ajout du bruit, calcul des TF) sont identiques pour les deux filtres (filtre inverse et filtre de Wiener). Il est donc possible de réutiliser une partie du code dédié au filtrage inverse.

⚡ Commenter les résultats obtenus.

⚡ Quelle valeur de  $K$  faut-il utiliser pour un bruit de RSB = 60dB ? Justifier ce choix.

### Manipulation 7

---

➔ On veut tester le filtrage de Wiener pour différentes valeurs de RSB (50 dB, 60 dB, 70 dB, 80dB, 90 dB, 100 dB).

➔ Pour chaque niveau de bruit, il faut déterminer la valeur de  $K$  la mieux adaptée. On testera des valeurs de  $K$  comprises entre 100 et 30000.

---

⚡ Pour chaque niveau de bruit, on relèvera la valeur de  $K$  la plus adaptée.

## 6 Comparaison des deux méthodes de déconvolution (30min)

On veut tester l'efficacité des méthodes de déconvolution en présence de bruit plus ou moins fort.

### Manipulation 8

---

➔ Ajouter du bruit gaussien à l'image `blurred_image` et appliquer les opérations de déconvolution par filtrage inverse et par filtrage de Wiener sur l'image dégradée et bruitée. On testera différentes valeurs de RSB (50 dB, 60 dB, 70 dB, 80dB, 90 dB, 100 dB).

➔ Attention pour le filtrage de Wiener, pour chaque niveau de bruit, il faut utiliser la meilleure valeur de  $K$  déterminée dans le paragraphe précédent.

➔ Évaluer la qualité de la restauration avec les différents critères vus en cours pour les différents niveaux de bruit et pour les deux méthodes de déconvolution.

---

⚡ Représenter sur un graphe les valeurs prises par les différents critères (RSB de reconstruction, EQM) en fonction du RSB pour le filtrage inverse et pour le filtrage de Wiener.

⚡ Commenter les résultats obtenus.