



ÉCOLE CENTRALE LYON

UE ELC
RAPPORT

Image processing - Frequency domain

Élèves :

Paul HOAREAU
Alexandra MICHON
Cécile SOUDÉ

Enseignant :

Mohsen ARDABILIAN

Table des matières

1	Introduction	2
2	Filtrage d'image	2
2.1	Transformée de Fourier discrète en 2D	2
3	Importance de la phase de la transformée de Fourier discrète (DFT)	5
3.1	Calcul de la magnitude et de la phase	6
3.2	Croisement de la magnitude et de la phase	7
3.3	Analyse des résultats	8
3.4	Rôle de la phase dans la transformée de Fourier	9
4	Filtrage linéaire d'images	9
5	Compression d'image	15
5.1	Fonctionnement de la Compression d'Image	15
5.2	Analyse des Résultats	16
6	Conclusion	18

1 Introduction

Le traitement d'images est aujourd'hui utilisé en permanence dans tous les outils technologiques. Maîtriser les opérations classiques permet de mieux comprendre le fonctionnement de ceux-ci, ainsi que d'être en mesure d'analyser certaines propriétés des images.

En particulier, toutes les propriétés des images ne sont pas mises en avant dans le domaine du réel. La conversion dans le domaine fréquentiel par la transformation de Fourier permet d'avoir un meilleur accès à d'autres propriétés, notamment structurelles, mais aussi d'obtenir des informations sur la composition des images.

Ce rapport présente donc les résultats du BE 2 réalisé en cours, et se décompose en plusieurs parties : application de la transformée de Fourier à des images ; mise en relief de l'importance de la phase dans la transformée de Fourier ; filtrage numérique d'images, et enfin exploration des processus de compression d'image.

Le lien entre ces approches réside dans le fait que pour compresser efficacement une image, on peut passer dans le domaine fréquentiel, afin de mieux percevoir quelles fréquences jouent un rôle secondaire, et peuvent donc être supprimées de l'image afin de réduire le stockage de celle-ci ou son temps d'exportation par exemple, sans impacter la perception de l'image par l'humain.

2 Filtrage d'image

2.1 Transformée de Fourier discrète en 2D

Cet exercice a pour objectif de mettre en évidence les propriétés d'une image mises en avant par la transformée de Fourier.

Pour ce faire, on s'intéresse à trois images différentes, qu'on convertit en nuances de gris (image originelle en haut à gauche de chaque quadrant).

On applique alors à chacune la transformée de Fourier (image en haut à droite des quadrants), puis on écrête le résultat et on le met à l'échelle afin de mieux observer la magnitude de la transformée de Fourier (résultat situé en bas à gauche des cellules). Enfin, on calcule le logarithme de cette transformée de Fourier, pour voir la structure de la magnitude transformée (en bas à droite des figures).

Les résultats du code exposé plus bas sont disponibles ci-dessous :

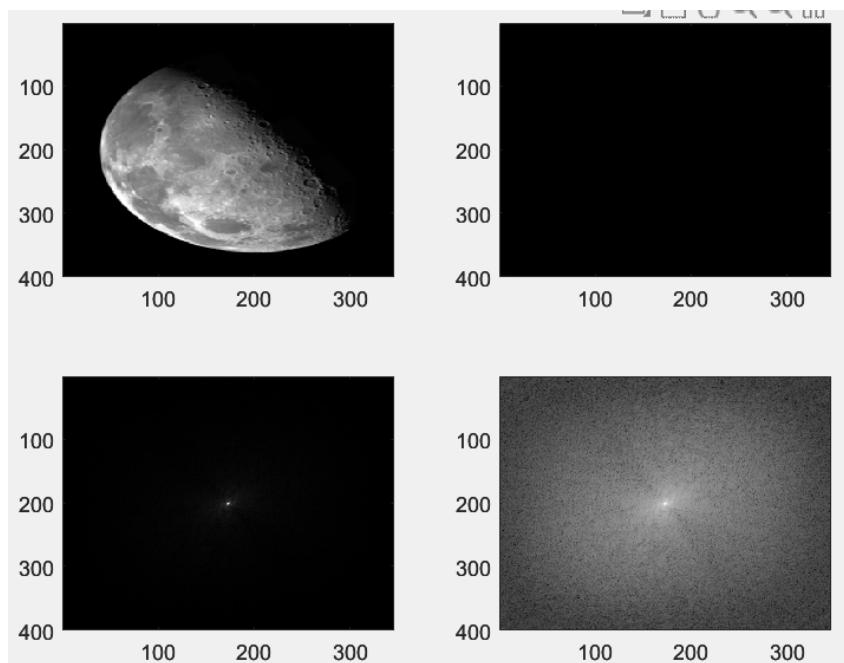


FIGURE 1 – Visualisation de l'image de Lune, lorsque les différentes étapes lui sont appliquées

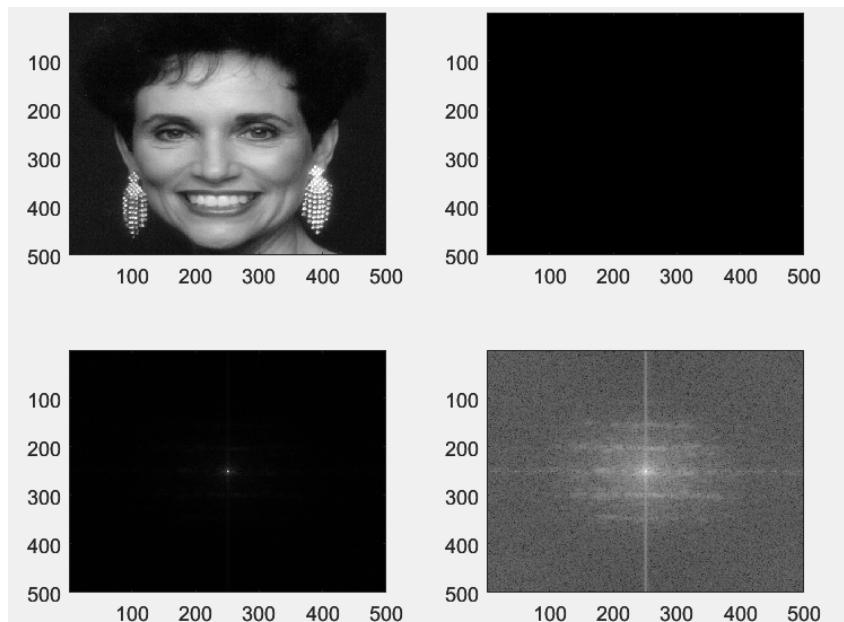


FIGURE 2 – Visage de femme, initialement et après chaque opération précédemment citée

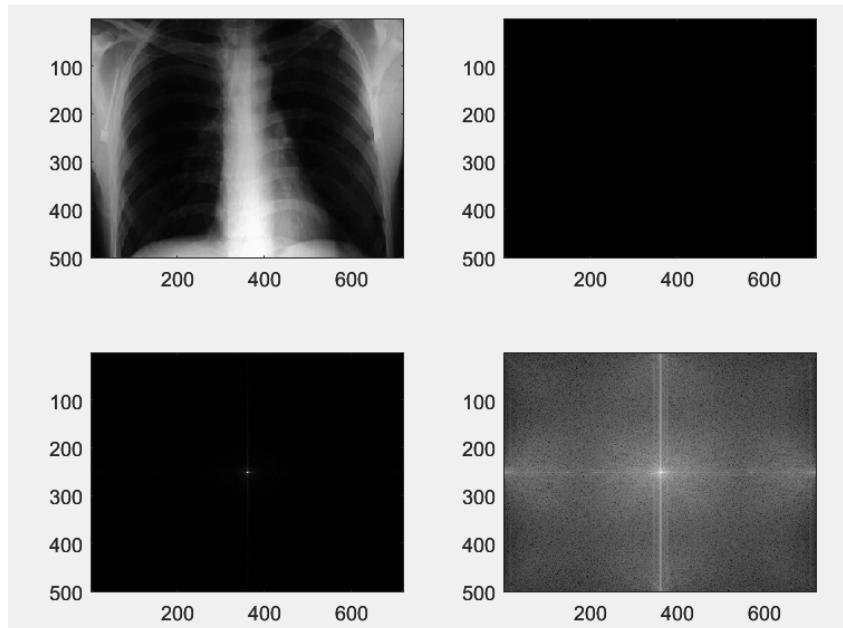


FIGURE 3 – Présentation des résultats de chaque opération appliquée à la première image (poumons)

On peut constater la nécessité d'effectuer un recadrage des transformées de Fourier des images, puisque toutes les cellules en haut à droite (FFT des images originales) sont noires. Cependant, la version écrêtée de celles-ci n'apporte que peu d'informations supplémentaires : la quasi-entièreté des images en bas à gauche sont noires, à l'exception d'un point blanc central, d'amplitude et de contours de plus en plus marqués entre les images de Lune, de femme et de poumons.

La troisième opération (application du logarithme) est celle qui apporte le plus de données sur les images initialement choisies :

La transformée de Fourier donne ainsi accès aux différentes fréquences présentes dans une image. Des fréquences élevées signalent la présence de fortes variations de couleurs, caractéristiques des bords d'un objet par exemple. A l'inverse, des zones uniformes sont représentées par des fréquences basses.

L'application de filtres passe-haut ou passe-bas permet de supprimer des informations selon l'objectif du traitement d'image : ôter les fréquences élevées revient à lisser, ou brouiller l'image, en supprimant les détails et les zones de forte variation de couleur, ce qui peut être utile pour compresser l'image en préservant l'essentiel de son contenu.

Par ailleurs, enlever les fréquences basses d'une image laisse seulement les détails et bordures de l'objet examiné.

Le code qui a servi pour modifier les images présentées au-dessus est le suivant :

```
%1ere image
lune=im2gray(imread('Fig_Moon.tif'));

moon=fft2(lune);
CS_moon=fftshift(moon);
```

```

CS_log_moon=log(1+abs(CS_moon));

figure(1);
subplot(2,2,1); imagesc(lune); colormap gray;
subplot(2,2,2); imagesc(abs(moon)); colormap gray;
subplot(2,2,3); imagesc(abs(CS_moon).^0.5); colormap gray;
subplot(2,2,4); imagesc(abs(CS_log_moon)); colormap gray;

%2eme image
femme=imread('Fig_Woman.tif');

woman=fft2(femme);
CS_woman=fftshift(woman);
CS_log_woman=log(1+abs(CS_woman));

figure(2);
subplot(2,2,1); imagesc(femme); colormap gray;
subplot(2,2,2); imagesc(abs(woman)); colormap gray;
subplot(2,2,3); imagesc(abs(CS_woman).^0.5); colormap gray;
subplot(2,2,4); imagesc(abs(CS_log_woman)); colormap gray;

%3eme image
rayon=im2gray(imread('Fig_Xray.tif'));
%im2gray permet de convertir de RGB en niveaux de gris directement

ray=fft2(rayon);
CS_ray=fftshift(ray);
CS_log_ray=log(1+abs(CS_ray));

figure(3);
subplot(2,2,1); imagesc(rayon); colormap gray;
subplot(2,2,2); imagesc(abs(ray)); colormap gray;
subplot(2,2,3); imagesc(abs(CS_ray).^0.5); colormap gray;
subplot(2,2,4); imagesc(abs(CS_log_ray)); colormap gray;

```

La fonction `im2gray` permet de convertir une palette de couleurs aléatoire en nuances de gris, et donc d'appliquer les opérations classiques par la suite à l'image étudiée.

La fonction `fftshift` utilisée dans le code a pour rôle de translater le point de magnitude du coin supérieur gauche au centre de l'Imagen, au centre de la plage de magnititude.

3 Importance de la phase de la transformée de Fourier discrète (DFT)

Dans cet exercice, nous explorons l'importance de la phase de la transformée de Fourier discrète (DFT) dans la préservation des propriétés d'une image. Pour ce faire, nous

réalisons une série d'expériences simples en utilisant deux images en niveaux de gris, Fig_woman.tif et Fig_Barre500.tif.

3.1 Calcul de la magnitude et de la phase

Tout d'abord, nous définissons la magnitude uniquement et la phase uniquement des images en utilisant les définitions suivantes :

$$\begin{aligned} f_{\text{mag}}[m, n] &\longleftrightarrow |F[i, k]| \\ f_{\text{phase}}[m, n] &\longleftrightarrow e^{j \cdot \arg\{F[i, k]\}} \end{aligned}$$

où $F[i, k]$ est la transformée de Fourier 2D de l'image $f[m, n]$, et $\arg\{\cdot\}$ représente l'angle de phase. Nous avons développé un programme MATLAB pour calculer ces deux versions des images et les afficher dans une grille 2×2 , comme illustré ci-dessous :

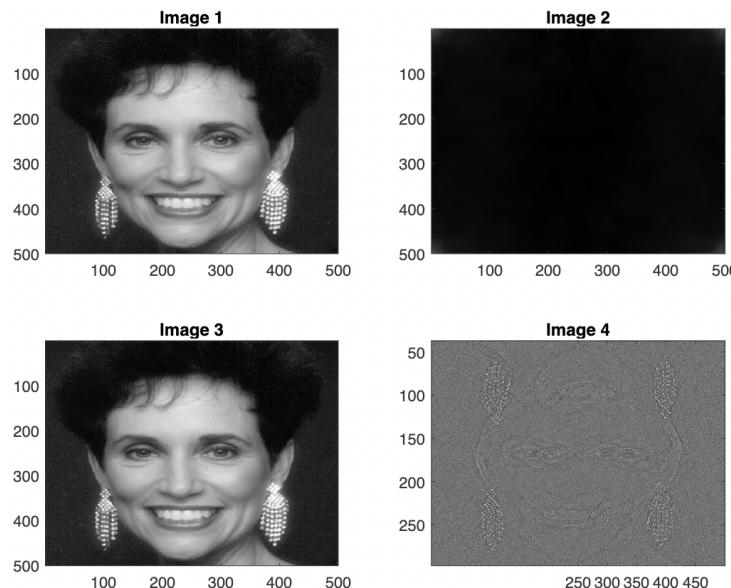


FIGURE 4 – Images générées montrant la magnitude et la phase pour l'image de la femme

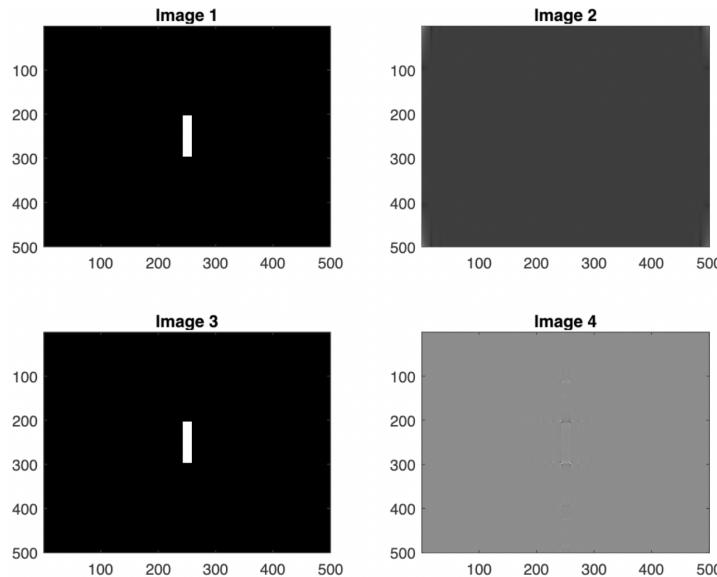


FIGURE 5 – Images générées montrant la magnitude et la phase pour l'image de la barre

NB : Même lorsqu'on normalise, rien ne semble apparaître dans l'image représentant la magnitude. Cependant, dans l'image de la phase on peut facilement distinguer les formes qu'on retrouve dans nos images principales.

Le code MATLAB utilisé pour cette tâche est le suivant :

```
function [im_mag, im_phase] = make_ph_mag(chemin) %Création des images de
magnitude et de phase
    %Acquisition et fft2 de l'image
    im = imread(chemin);
    im_fft2 = fft2(im);

    %Génération de l'image magnitude
    im_fft2_mag = abs(im_fft2);
    im_mag = ifft2(im_fft2_mag);

    %Génération de l'image phase
    im_fft2_phase = real(exp(1i*angle(im_fft2)));
    im_phase = ifft2(im_fft2_phase);
end
```

3.2 Croisement de la magnitude et de la phase

Dans cette partie, nous échangeons les magnitudes et les phases des transformées de Fourier 2D des deux images précédentes. Ensuite, nous reconstruisons les images en utilisant les nouvelles magnitudes croisées avec les phases d'origine. Les résultats sont affichés dans une grille 2×2 , comme suit :

Le code MATLAB utilisé pour cette tâche est le même que celui utilisé précédemment, avec une légère modification pour effectuer le croisement des magnitudes et des phases.

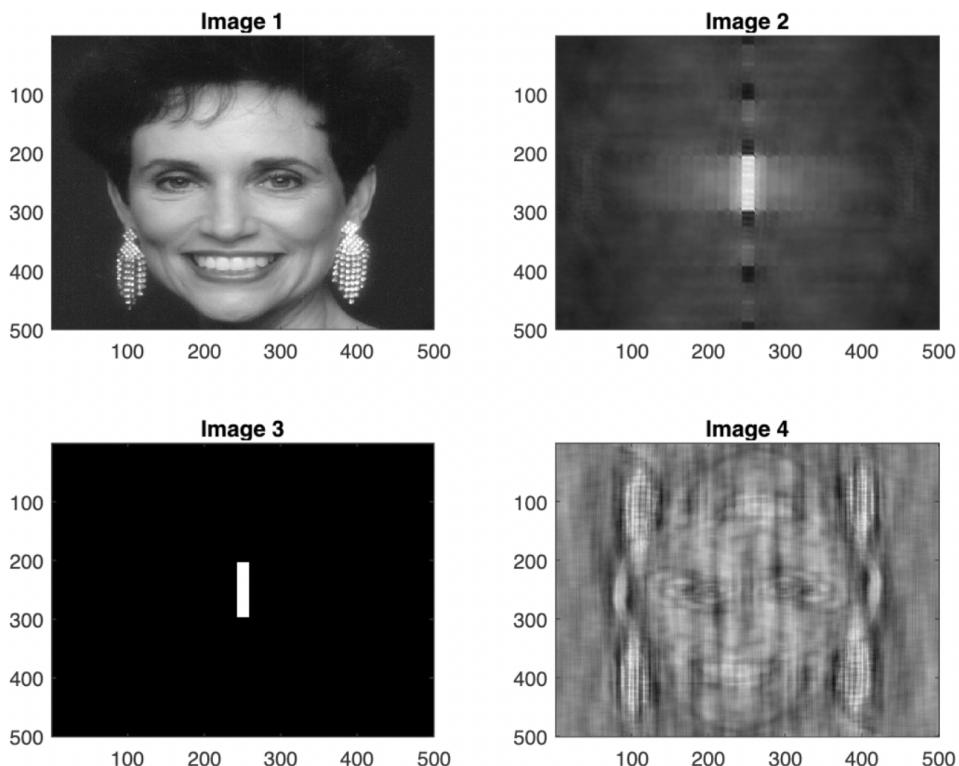


FIGURE 6 – Images générées montrant les magnitudes croisées avec les phases d'origine

```

function [image_cross] = cross_ph_mag(im1, im2)
    im1_fft2_phase = angle(fft2(im1));
    im1_phase = real(exp(1i*im1_fft2_phase));

    im2_mag = abs(fft2(im2));
    image_cross = ifft2(im1_phase.*im2_mag);

end

```

3.3 Analyse des résultats

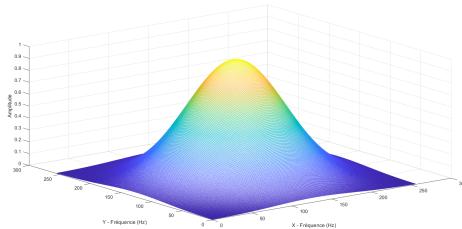
À partir des résultats de nos expériences, ainsi que des conclusions tirées de l'exercice précédent, nous pouvons observer que la phase de la transformée de Fourier discrète joue un rôle crucial dans la préservation des propriétés de l'image. Alors que la magnitude contribue à la représentation de la distribution d'amplitude des fréquences, la phase détient des informations cruciales sur la structure spatiale de l'image. Cela suggère que la phase est souvent plus importante que la magnitude pour préserver les propriétés originales de l'image.

Cette observation peut être attribuée à la nature même du processus de transformation de Fourier, où la phase contient des informations sur la localisation spatiale des motifs de l'image, tandis que la magnitude ne fournit que des informations sur leur amplitude. Ainsi, même si la magnitude peut être utilisée pour reconstruire une image, elle ne garantit pas la préservation de ses caractéristiques spatiales originales.

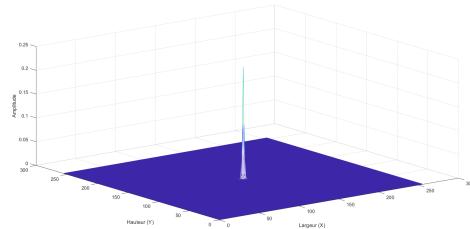
3.4 Rôle de la phase dans la transformée de Fourier

4 Filtrage linéaire d'images

Le filtre-passe bas construit à partir de la fonction *lpfilter* admet les réponses impulsionales et fréquentielles suivantes [7], [8], et [9], selon le type de filtre appliqué ("Gaussian", "Ideal", "btw") et la fréquence d'arrêt :

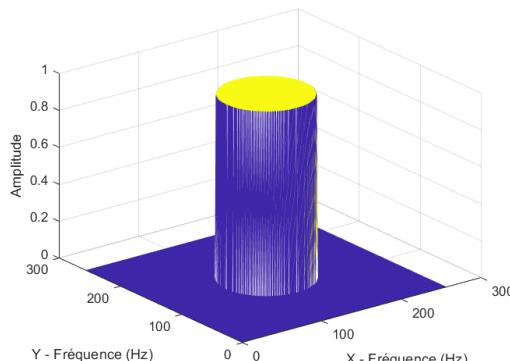


[H] (a) Réponse fréquentielle

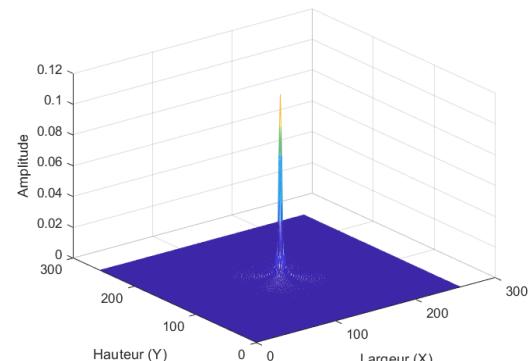


(b) Réponse impulsionale

FIGURE 7 – Passe-bas, type de filtre : 'gaussian', D0 = 50

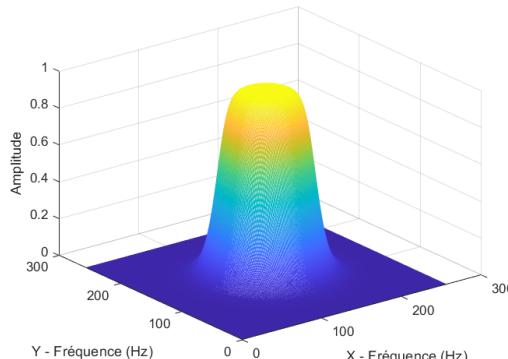


(a) Réponse fréquentielle

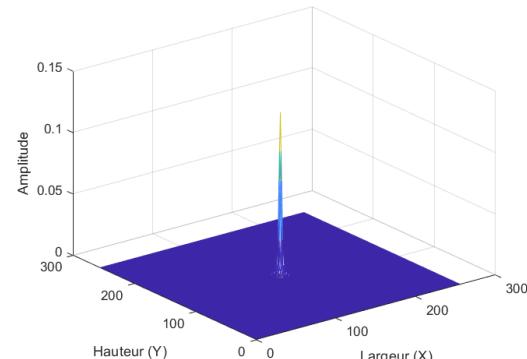


(b) Réponse impulsionale

FIGURE 8 – Passe-bas, type de filtre : 'ideal', D0 = 50



(a) Réponse fréquentielle

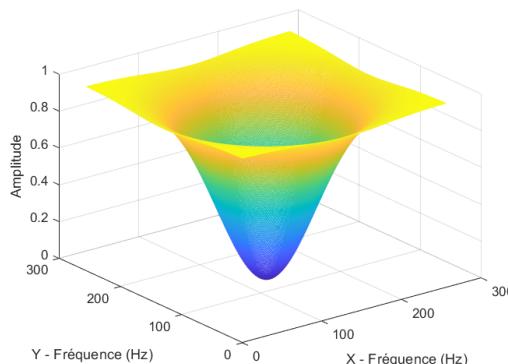


(b) Réponse impulsionnelle

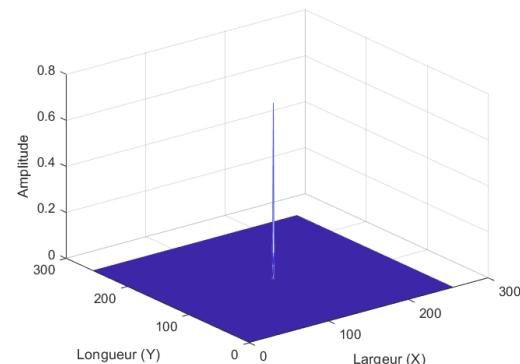
FIGURE 9 – Passe-bas, type de filtre : 'btw', D0 = 50, n=4

Par sa forme en cloche [10a], le **filtre gaussien** applique un filtrage progressif des fréquences qui atténuent les hautes fréquences mais pas totalement. Les valeurs maximales sont concentrées au centre de la grille, indiquant les basses fréquences, tandis que les valeurs diminuent progressivement en s'éloignant du centre, représentant les hautes fréquences atténuées par le filtre. Contrairement à ce filtre, le **filtre idéal** supprime totalement les fréquences non comprises dans le cercle de centre correspondant au milieu de l'image et de rayon égal à la fréquence d'arrêt (D0) (voir [8a]). Enfin, le **filtre de Butterworth**[9a] offre un compromis entre le filtre gaussien et le filtre idéal, avec une transition progressive entre les fréquences coupées et non coupées.

Pour convertir le filtre passe-haut en fonction du filtre passe-bas, il suffit de retirer à 1, la réponse fréquentielle du filtre passe-bas. En effet, les hautes et basses fréquences couvrent l'ensemble du spectre fréquentielle. Concernant le filtre passe-bas, les fréquences d'amplitude égale à 1 passe contrairement à celles valent 0. Ainsi, les fréquences ayant une amplitude nulle pour le filtre passe-bas, passent entièrement pour le filtre passe-haut. La figure [10], montre la réponse fréquentielle et impulsionnelle obtenue pour un filtre passe-haut gaussien avec une valeur d'arrêt D0 = 50. On remarque bien que contrairement à la figure [7], représentant le filtre passe-bas, les fréquences de sortie du filtre sont "l'inverse" de celles du filtre passe-bas.



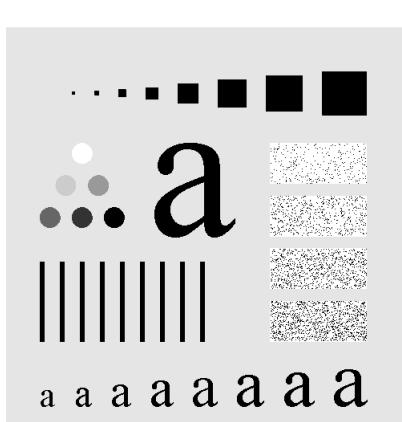
(a) Réponse fréquentielle



(b) Réponse impulsionnelle

FIGURE 10 – Passe-haut, 'gaussian', D0 = 50

A.3.2. L'image test est de taille 500x500. Le filtre choisi est un filtre de Butterworth d'ordre 4 avec une fréquence d'arrêt de 15 Hz. Le filtre passe-bas atténue les hautes fréquences associées aux bords nets, et changement rapide de couleurs. Ainsi, après application du filtre, l'image devient beaucoup plus floue avec des bords moins nets. C'est bien ce que l'on observe sur les figures [11] et [12].

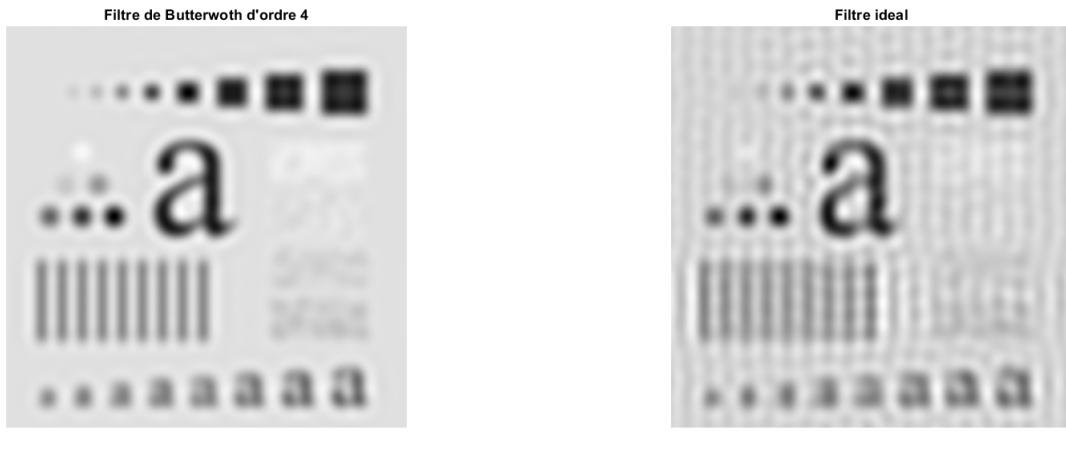


(a) Image originelle



(b) Filtrage Gaussien

FIGURE 11 – Comparaison des différentes méthodes de filtrage



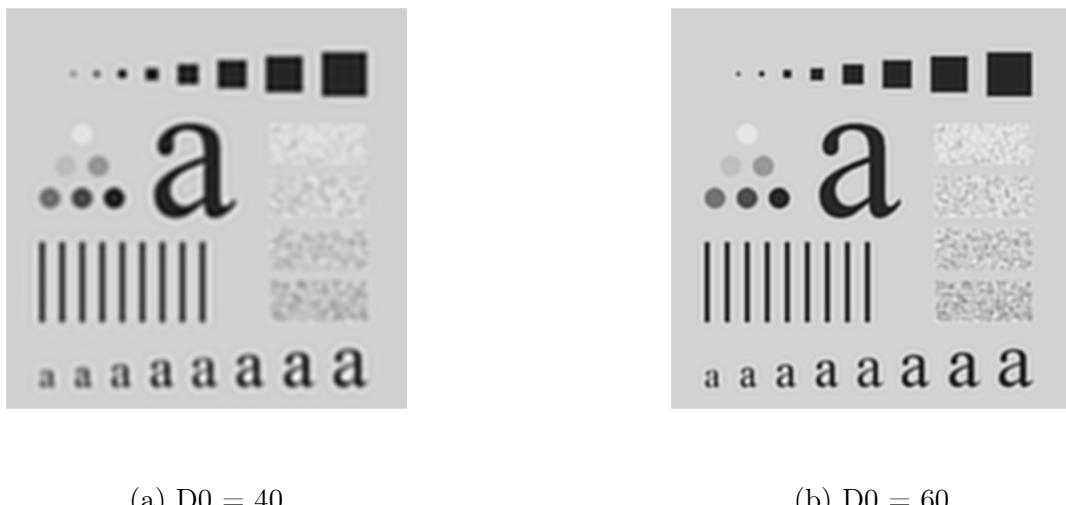
(a) Filtrage de Butterworth d'ordre 4

(b) Filtrage Ideal

FIGURE 12 – Comparaison des différentes méthodes de filtrage

Les différentes méthodes de filtrage appliquent chacune un filtrage des fréquences hautes différentes. Les remarques sont les mêmes que lors de la première question. Le filtrage ideal [12b] atténue nettement les hautes fréquences tandis que les filtrages de Gauss et Butterworth ([11b] et [11]) atténuent de manière plus progressive les hautes fréquences, rendant ainsi l'image floue sur son ensemble. L'atténuation du filtre de Gauss reste tout de même beaucoup moins prononcée que celui du filtre de Butterworth.

Une autre étude du filtrage consiste à visualiser l'impact d'un changement de fréquence d'arrêt. Pour ce faire, plusieurs fréquences ont été choisies entre 20 et 200 Hz, donnant les résultats suivant (filtre de Butterworth d'ordre 4 appliqué) :


 (a) $D_0 = 40$

 (b) $D_0 = 60$

FIGURE 13 – Influence de la fréquence d'arrêt

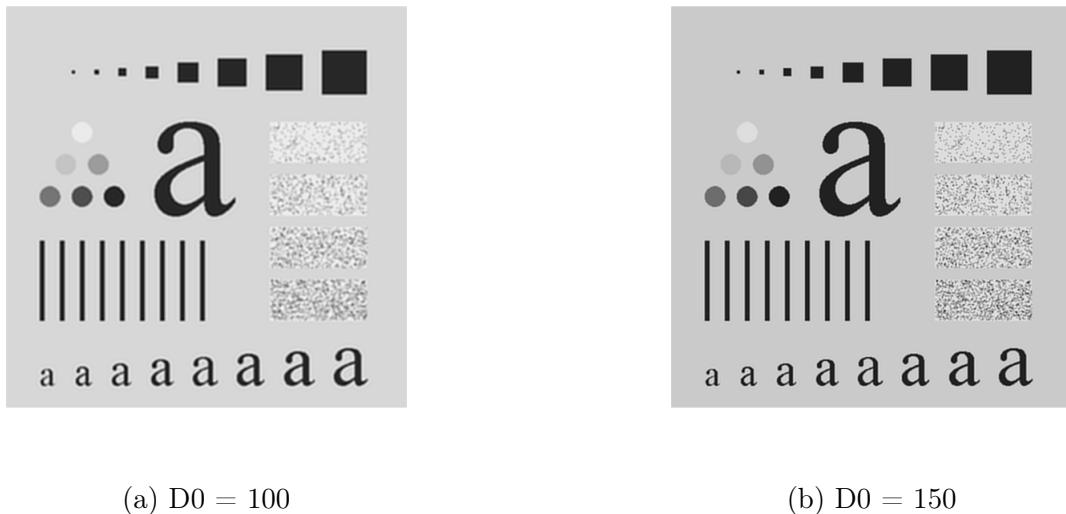


FIGURE 14 – Influence de la fréquence d’arrêt

Les figures [13] et [16] montrent que l’augmentation de la fréquence d’arrêt améliore la netteté. En effet, les hautes fréquences sont beaucoup moins atténuées par le filtre puisque la fréquence de coupure augmente.

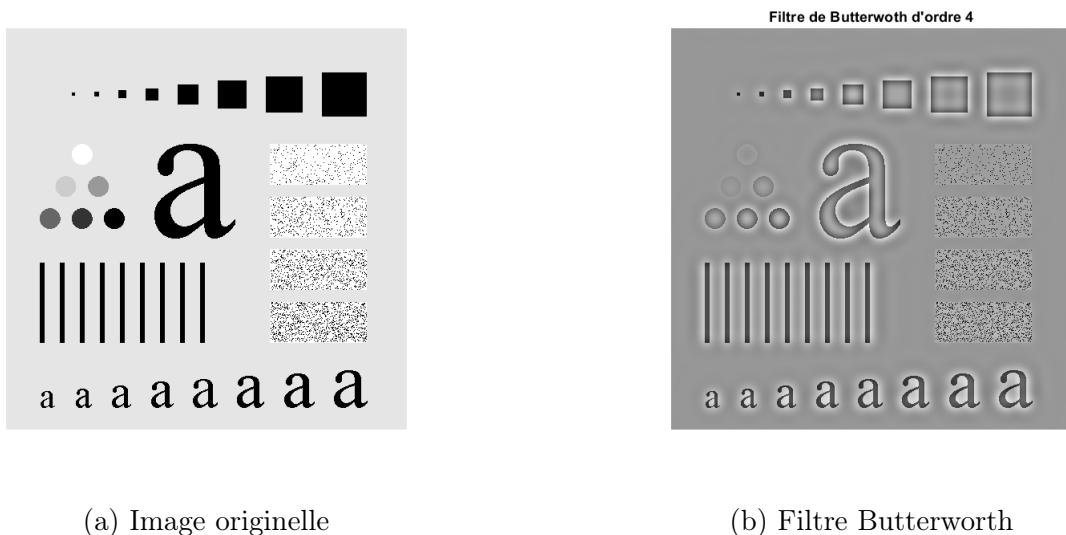


FIGURE 15 – Comparaison des différentes méthodes de filtrage - Passe-haut

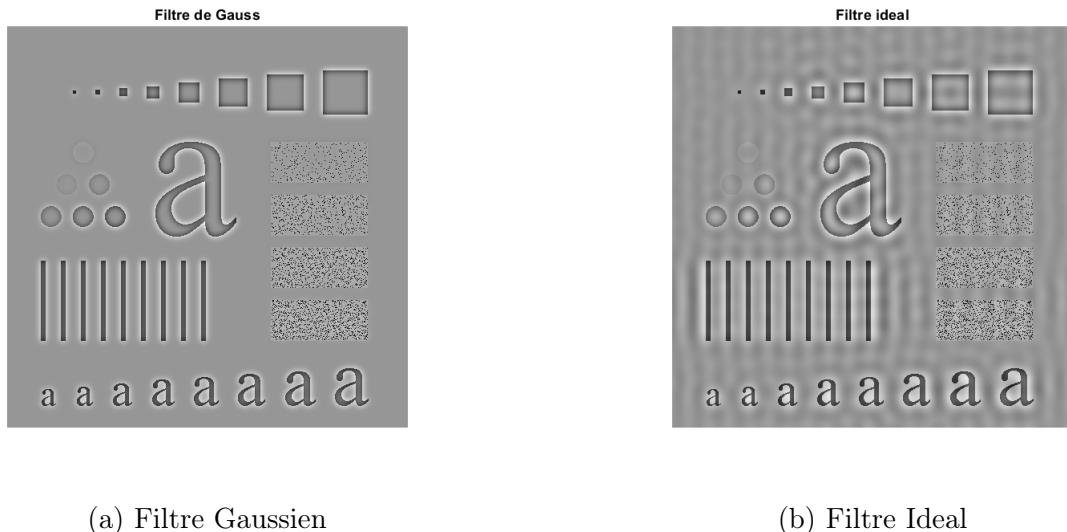
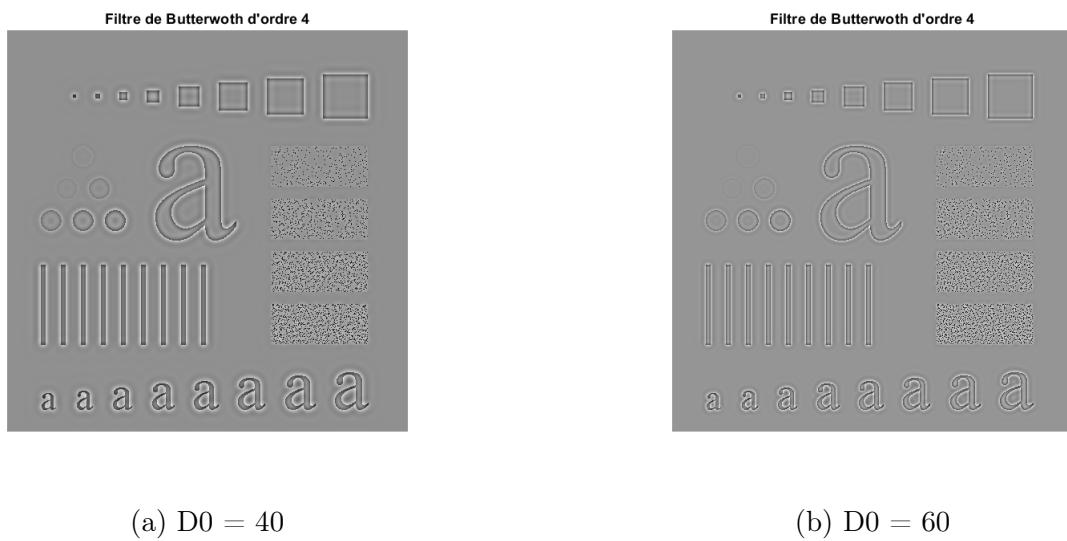
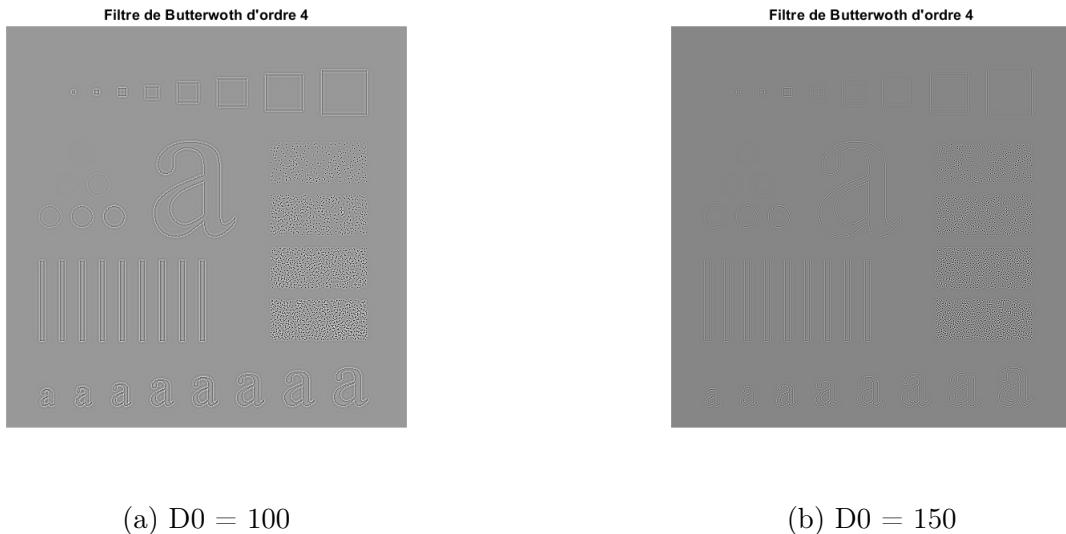


FIGURE 16 – Comparaison des différentes méthodes de filtrage - Passe-haut

Le filtre passe-haut semble accentuer les contours et les détails tout en supprimant les basses fréquences de l'image. Ainsi, les zones uniformes et les dégradés lents, sont atténuées davantage, conduisant à une diminution de la luminosité globale de l'image et à un contraste accru entre les différentes zones de l'image. Comme nous pouvons le voir sur l'image [16], le filtre passe-haut semble rendre plus visible le bruit de l'image. Comme les remarques précédentes, le filtre gaussien atténue de manière plus faible les fréquences basses tandis que le filtre ideal est beaucoup plus "brute" et ne fait pas de compromis. Le filtre de Butterworth est un juste milieu entre ces deux filtres.


 FIGURE 17 – Influence de la fréquence de coupure D_0


 FIGURE 18 – Influence de la fréquence de coupure D_0

D’après les observations des figures [17] et [18], pour une augmentation de la fréquence de coupure, le filtre semble laisser passer davantage de détails fins et de variations rapides dans l’image. Cela conduit alors à une augmentation de la netteté des contours et des textures.

5 Compression d’image

La compression d’image est une technique cruciale dans le domaine du traitement d’images, permettant de réduire la taille des fichiers tout en préservant une qualité visuelle acceptable. Dans cette section, nous abordons la mise en œuvre de la compression et de la décompression d’images à l’aide de la transformation en cosinus discrète (DCT), ainsi que l’analyse des résultats obtenus.

5.1 Fonctionnement de la Compression d’Image

La compression d’image repose sur plusieurs étapes clés, dont la transformation en cosinus discrète (DCT) et la quantification. La DCT est utilisée pour convertir les données spatiales de l’image en données fréquentielles, ce qui permet de concentrer l’énergie de l’image dans un nombre réduit de coefficients. Ensuite, la quantification est appliquée pour réduire la précision de ces coefficients, ce qui permet une compression plus efficace en éliminant les informations moins perceptibles pour l’œil humain. Le processus de décompression implique simplement l’inverse de ces étapes pour reconstruire l’image à partir des coefficients compressés.

Dans le processus de compression d’image mis en œuvre par le code fourni, plusieurs étapes sont entreprises pour réduire la taille du fichier tout en préservant les informations visuelles importantes. Tout d’abord, l’image d’entrée est lue à partir d’un fichier bitmap et convertie en double précision pour permettre des calculs précis. Ensuite, chaque bloc de l’image est divisé en segments de 8x8 pixels, sur lesquels la transformation en cosinus discrète (DCT) est appliquée. Cette transformation permet de convertir les pixels de

l'image en coefficients de fréquence spatiale, révélant les caractéristiques de haute et basse fréquence de l'image.

Une fois la transformation DCT effectuée, les coefficients résultants sont quantifiés en fonction d'un facteur de qualité prédéfini. La quantification réduit la précision des coefficients en les divisant par des valeurs spécifiques déterminées par une matrice de quantification. Cette étape permet de supprimer les détails moins perceptibles de l'image, facilitant ainsi une compression plus efficace. Enfin, les coefficients DCT quantifiés sont écrits dans un fichier compressé, généralement sous la forme d'un pseudo-JPG, prêt à être stocké ou transmis. Ce processus de compression garantit une réduction significative de la taille du fichier tout en maintenant une qualité d'image acceptable pour la décompression ultérieure.

Pour illustrer ce processus, nous avons mis en œuvre des fonctions MATLAB pour compresser et décompresser une image. Le code ci-dessous montre comment nous avons calculé l'erreur entre l'image originale et l'image décompressée pour différents facteurs de quantification.

```
% Charger l'image
original_image = imread('Nature.bmp'); % Charger votre image ici

% Initialiser des vecteurs pour stocker les résultats
difference_error = zeros(1,25);
standard_error = zeros(1,25);

% Parcourir les différents facteurs de quantification
for quantization_factor = 1:25
    % Compression de l'image
    main_compression(quantization_factor, 'Nature.bmp', 'Nature_compressed.jpg');
    % Décompression de l'image
    main_decompression('Nature_compressed.jpg', 'Nature_decompressed.bmp');
    decompressed_image = imread('Nature_decompressed.bmp');
    % Calcul de la différence entre l'image originale et celle décompressée
    difference = double(original_image) - double(decompressed_image);
    difference_error(quantization_factor) = mean(abs(difference(:))); % Erreur moyenne
    standard_error(quantization_factor) = std(abs(difference(:))); % Erreur standard
end
```

5.2 Analyse des Résultats

Une fois les images compressées et décompressées pour différents facteurs de quantification, nous avons évalué les performances de compression en calculant l'erreur moyenne et l'erreur standard entre l'image originale et l'image décompressée. Les graphiques ci-dessous montrent l'évolution de ces métriques en fonction du facteur de quantification.

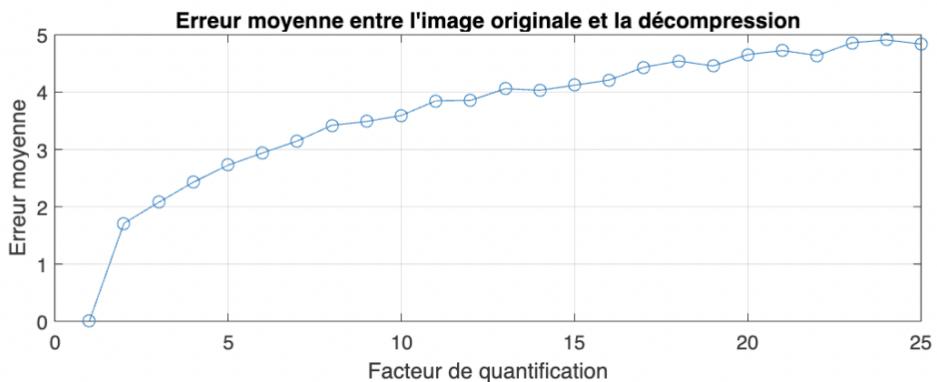


FIGURE 19 – Erreur moyenne entre l'image originale et la décompression en fonction du facteur de quantification.

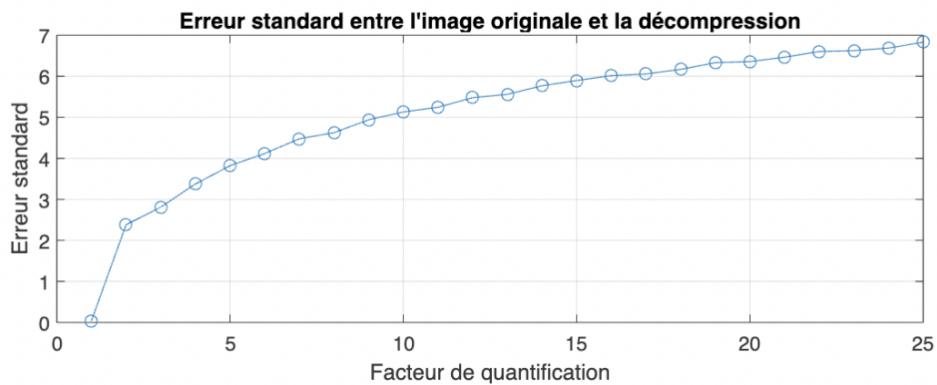


FIGURE 20 – Erreur standard entre l'image originale et la décompression en fonction du facteur de quantification.

Nous observons que plus le facteur de quantification est élevé, plus l'erreur moyenne et l'erreur standard augmentent, ce qui est attendu car une plus grande quantification conduit à une perte d'information plus importante. Ces résultats mettent en évidence le compromis entre la qualité de l'image et la taille du fichier compressé.

De plus, nous avons calculé le taux de compression en mesurant le nombre de coefficients DCT non nuls pour chaque facteur de quantification. Le graphique ci-dessous présente le taux de compression en pourcentage en fonction du facteur de quantification.

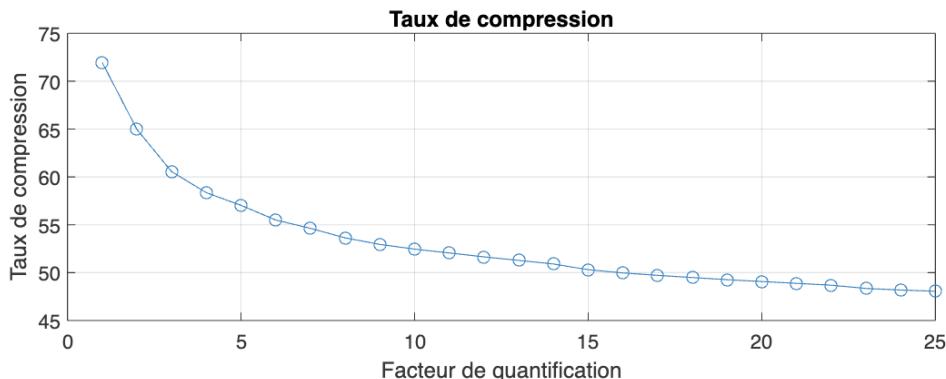


FIGURE 21 – Taux de compression en pourcentage en fonction du facteur de quantification

Nous constatons que le taux de compression augmente de manière significative avec le facteur de quantification, ce qui confirme l'efficacité de la compression basée sur la DCT. Cependant, il est important de noter que des valeurs de quantification plus élevées entraînent une perte de qualité d'image plus importante, comme le montrent les résultats précédents sur l'erreur de compression.

En conclusion, la compression d'image basée sur la DCT offre un moyen efficace de réduire la taille des fichiers tout en préservant une qualité visuelle acceptable, mais nécessite un compromis entre la taille du fichier et la qualité de l'image.

6 Conclusion

En conclusion, ce rapport met en évidence l'importance du traitement d'images et de l'analyse fréquentielle pour comprendre et manipuler efficacement les données visuelles. À travers l'application de la transformation de Fourier, nous avons pu explorer les propriétés structurelles des images et mettre en évidence l'influence cruciale de la phase dans la préservation des informations. Le filtrage numérique a également été abordé, démontrant comment les filtres peuvent être conçus et appliqués dans le domaine fréquentiel pour améliorer ou altérer des caractéristiques spécifiques des images.

Par ailleurs, l'étude sur la compression d'images a souligné l'importance de la représentation fréquentielle pour réduire la taille des données tout en préservant la qualité visuelle. En analysant les courbes d'erreur et les taux de compression en fonction des facteurs de quantification, nous avons pu observer comment les coefficients DCT influent sur la qualité de l'image compressée.

Enfin, ce rapport illustre l'interconnexion entre les différentes techniques de traitement d'images, montrant comment la compréhension des concepts fondamentaux tels que la transformation de Fourier peut conduire à des avancées significatives dans des domaines variés tels que la compression d'images. Ainsi, cette étude souligne l'importance de maîtriser ces outils et techniques pour manipuler efficacement les images dans divers contextes technologiques et scientifiques.