

Compte Rendu TP1

Initiation Au Traitement D'Image Sous MATLAB IGE43 – Groupe 1

Présenté par :

- ❖ HALLA Senia
- ❖ KAID Nassima

Mardi 18 Mai 2021

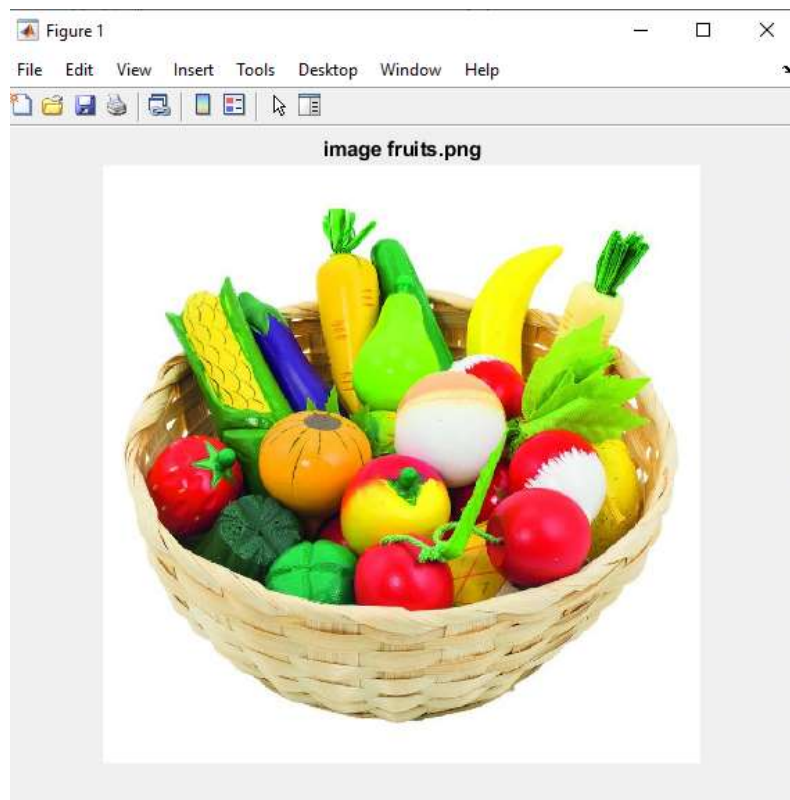
Partie 1 : Lecture Et Affichage d'une image

1. Initialisation de l'environnement MATLAB :
2. Création de la variable 'chemin' jusqu'à l'image « fruits.jpg » :
3. Création de la variable 'img' contenant le nom de l'image :

```
chemin = 'E:\cours\SEMESTRE 2\TI\TP\TP1\image_TP1\';  
img = 'fruits.jpg';
```

4. La fonction imread () :

```
ima=imread(cat(2,chemin,img),'jpg');
```
5. Lire l'image :

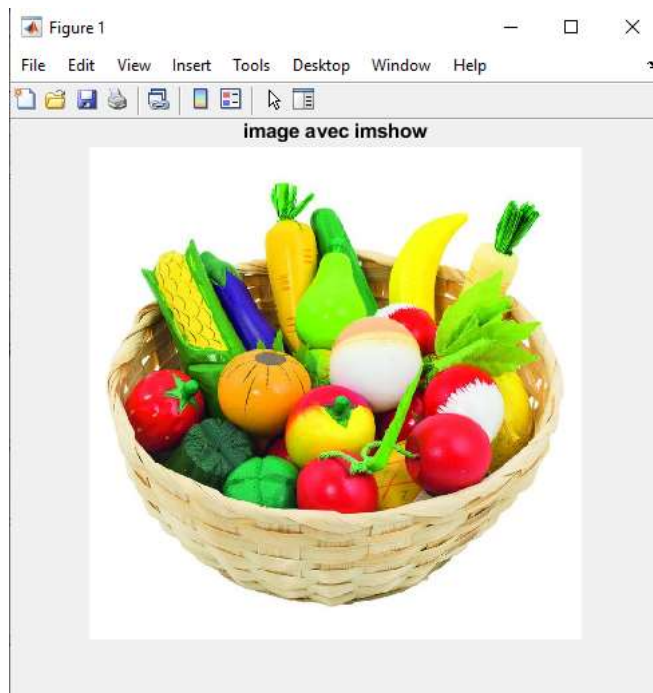


6. Le format de l'image lue dans le gestionnaire de variable 'whos'
Elle nous donne des informations sur les variables stockées dans la mémoire dans notre cas ce sont les variables chemin, ima et img

| Name | Size | Bytes | Class | Attributes |
|--------|-----------|---------|-------|------------|
| chemin | 1x53 | 106 | char | |
| ima | 800x800x3 | 1920000 | uint8 | |
| img | 1x10 | 20 | char | |

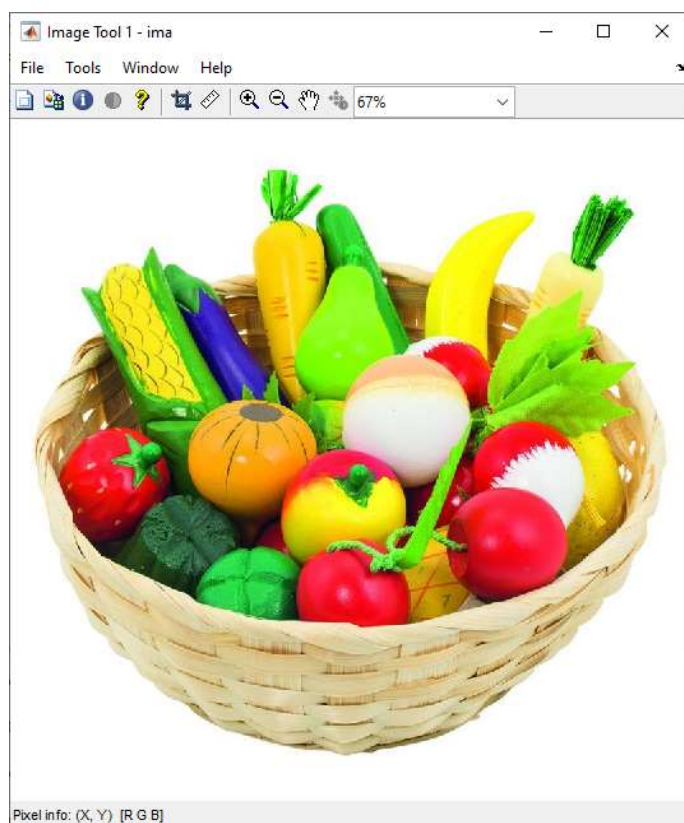
7. Affichage de l'image avec :

➤ La fonction imshow ()



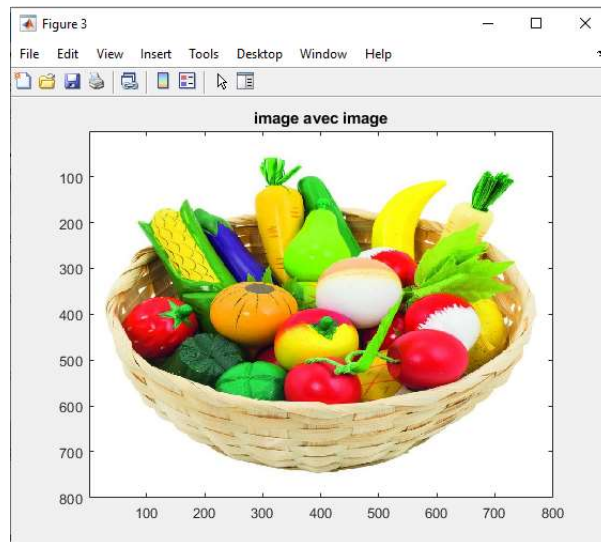
Avec imshow on affiche juste l'image

- La fonction imshow ()

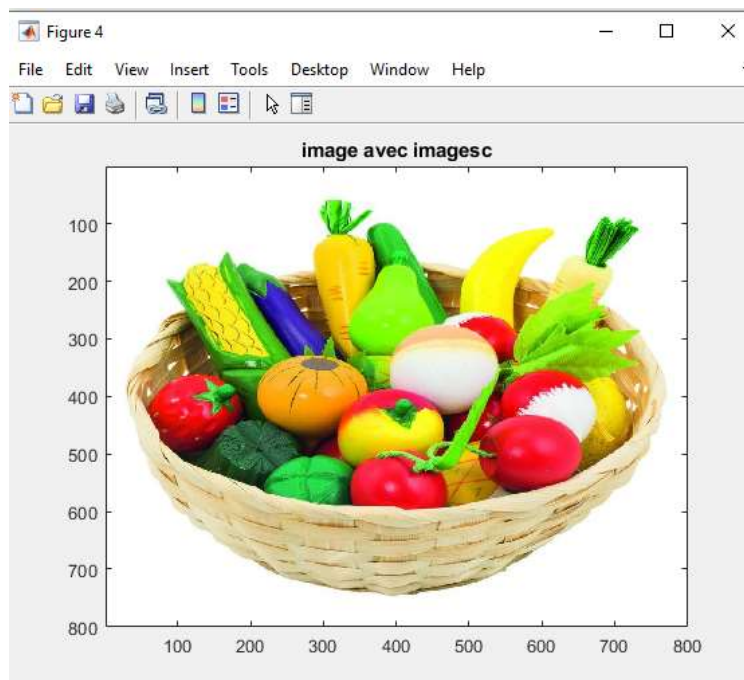


Avec imtool on a la possibilité de faire des modifications sur les valeurs des pixels

- La fonction image ()

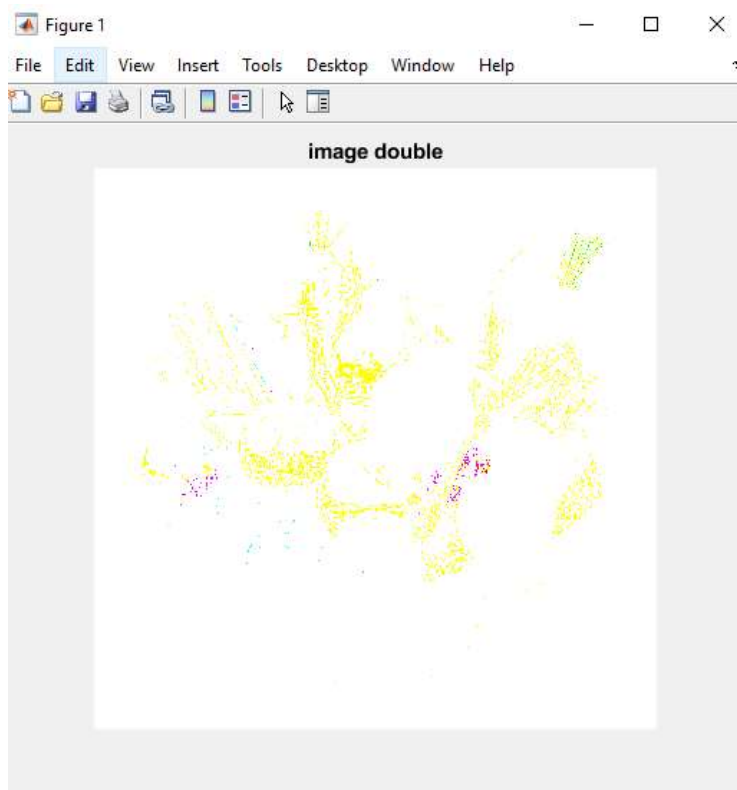


- La fonction `imagesec()`
Affiche l'image avec une échelle (scaling data)

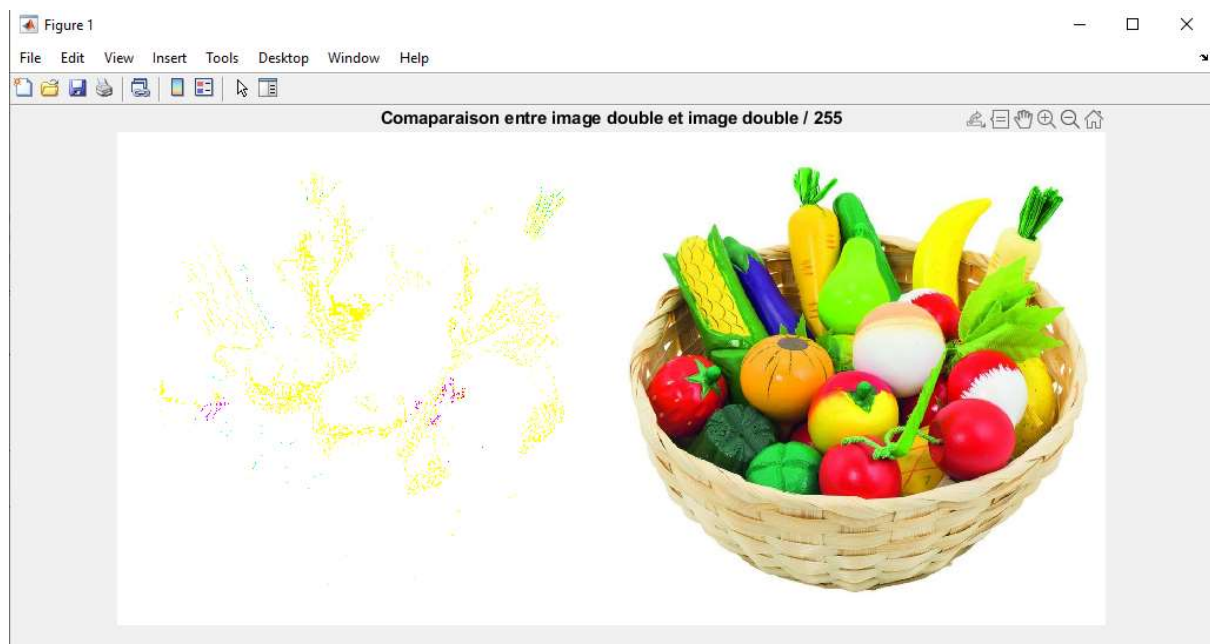


- Création de le variable 'img_double' : `img_d = double(ima;`

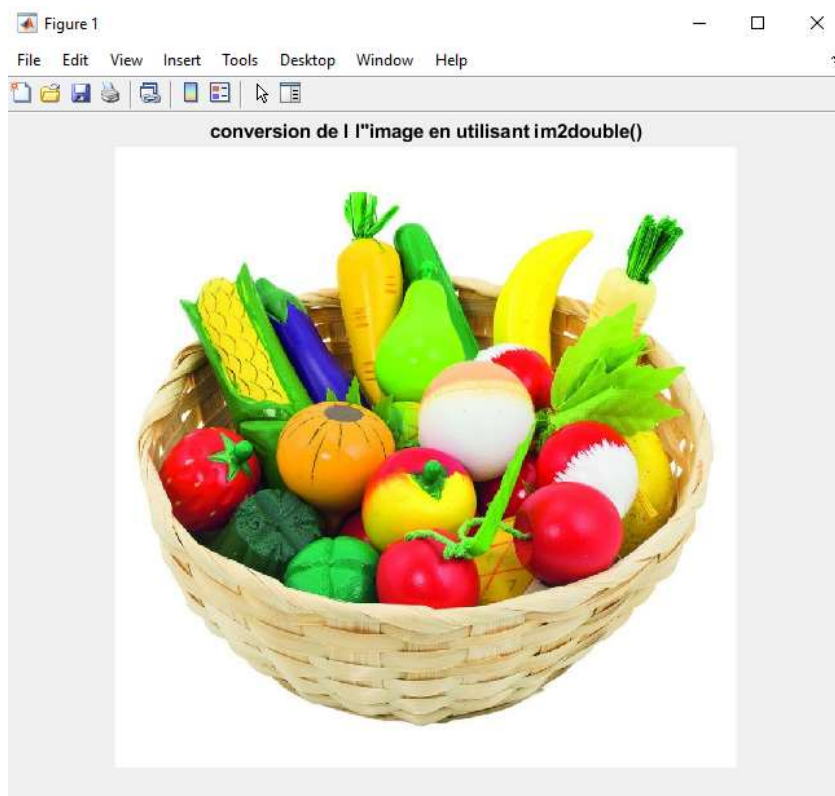
En utilisant la fonction `double()` c'est comme si on multiplie la matrice de notre image par 255



9. Diviser de l'image 'img_double' par la valeur 255 : $\text{img_d1} = \text{img_d}/255;$
On retrouve l'image originale



10. Création de l'image 'img_2_double' : $\text{img_2_d} = \text{im2double}(\text{ima});$



10. Comparaison des deux images selon le les valeurs des pixels :

```

size(img_2_d)
size(img_d)

>> size(img_2_d)

ans =

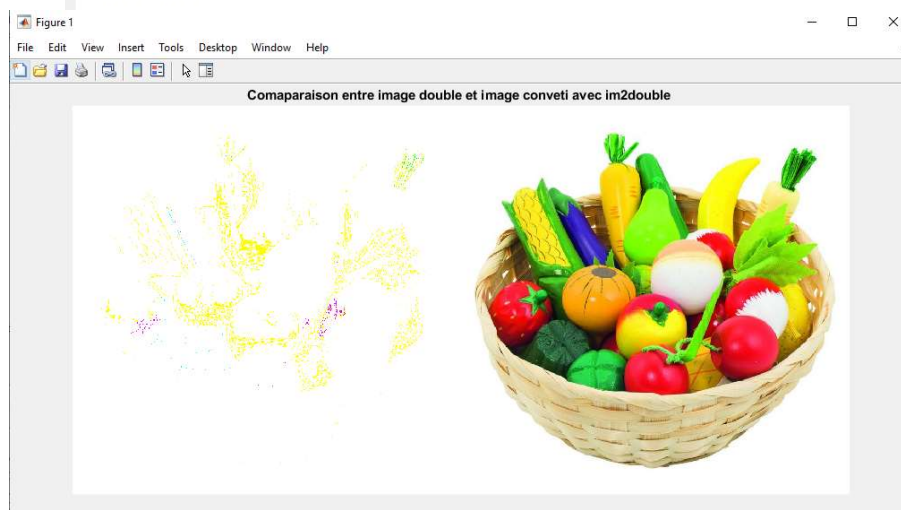
    800    800     3

>> size(img_d)

ans =

    800    800     3

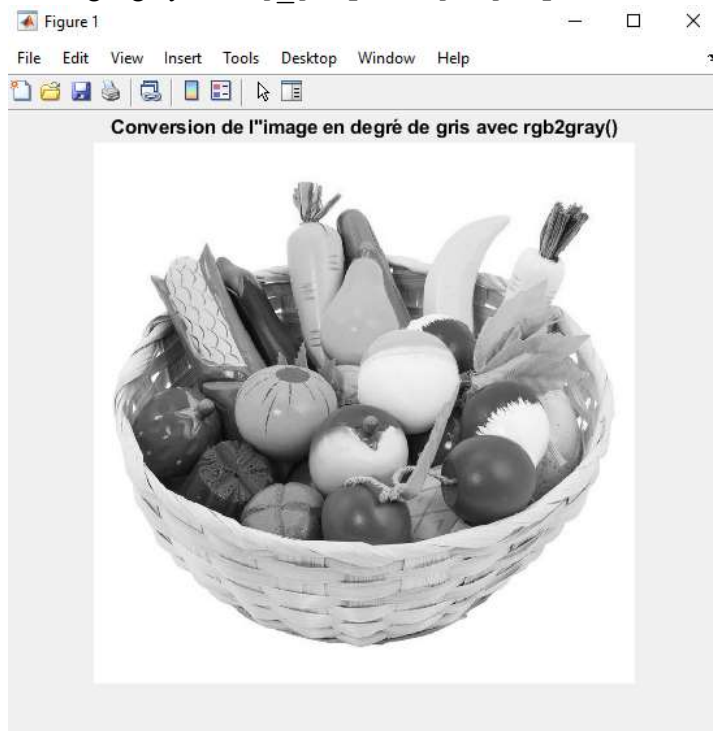
```



Les deux images ont la même taille mais pas les mêmes valeurs des pixels

11. Conversion de l'image 'img' en une image en niveau de gris :

⇒ Avec `rgb2gray` : `img_gray = rgb2gray(ima);`



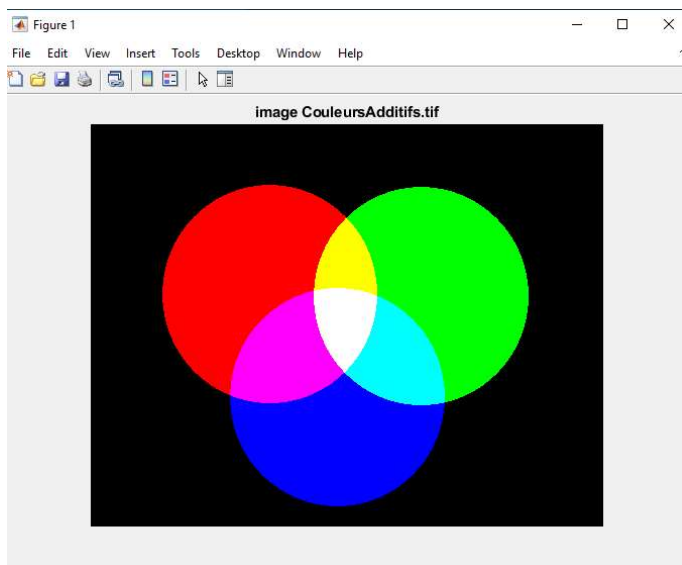
⇒ Avec les coefficients :

```
img_gray_1 = ima(:,:,1)*0.1 + ima(:,:,2)*0.1 + ima(:,:,3)*0.8;  
img_gray_2 = ima(:,:,1)*0.1 + ima(:,:,2)*0.8 + ima(:,:,3)*0.1;  
img_gray_3 = ima(:,:,1)*0.8 + ima(:,:,2)*0.1 + ima(:,:,3)*0.1;
```

Partie 2 : Types D'Image Dans MATLAB

1. Images niveaux de gris :

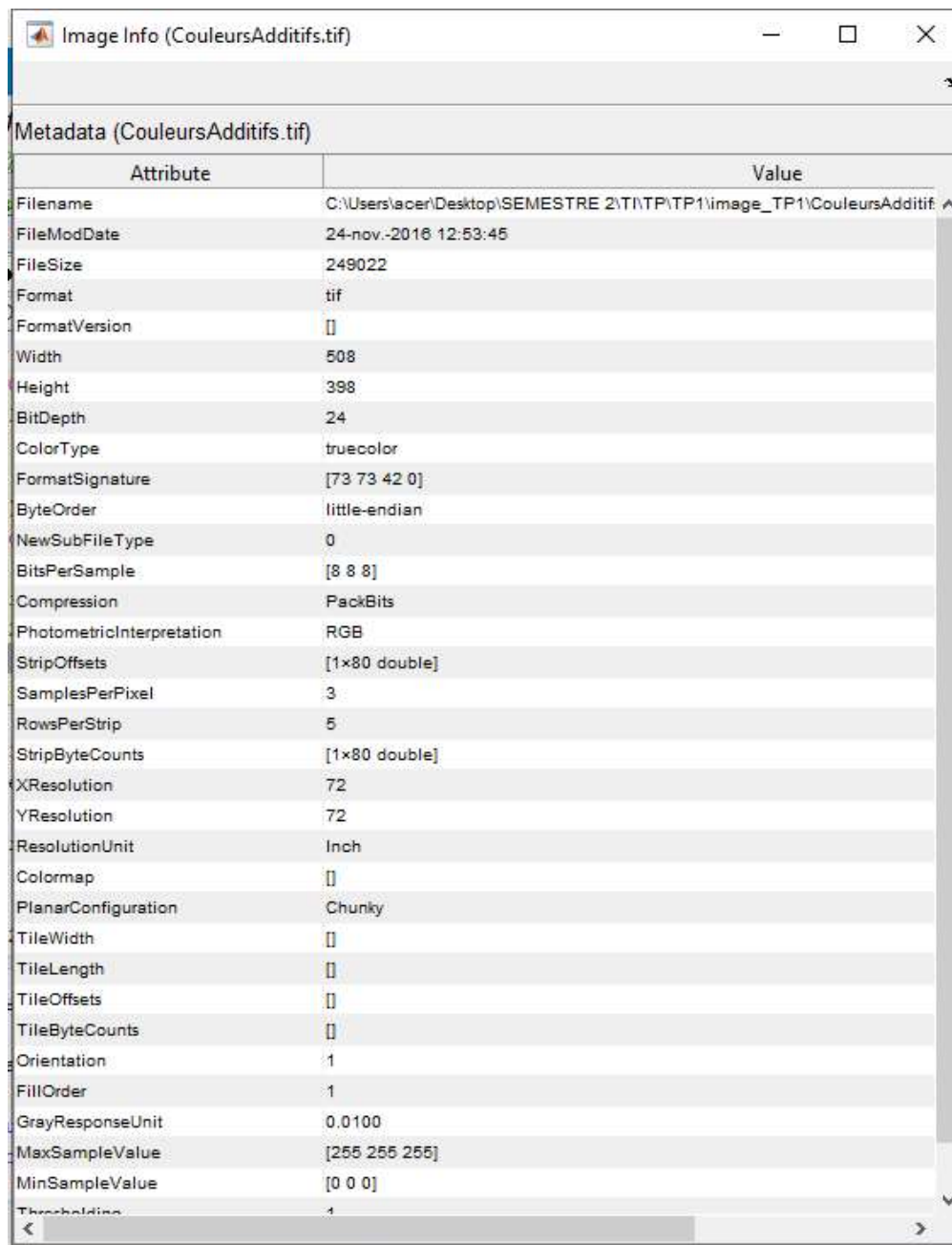
➤ Chargement de l'image 'CouleursAdditifs.tif'



- Description des caractéristiques

```
imageinfo('CouleursAdditifs.tif');
```

C'est une image truecolor (RGB) de taille [398 ;508], les valeurs des plans sont en uint8

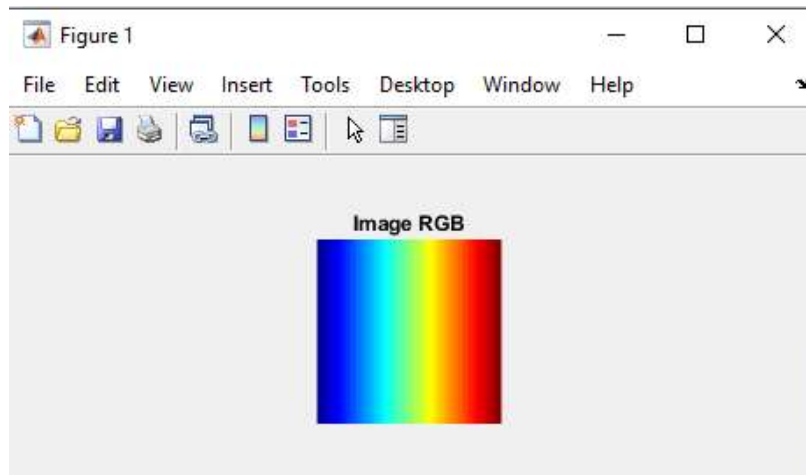


| Metadata (CouleursAdditifs.tif) | |
|---------------------------------|--|
| Attribute | Value |
| Filename | C:\Users\acer\Desktop\SEMESTRE 2\TI\TP\TP1\image_TP1\CouleursAdditif |
| FileModDate | 24-nov.-2016 12:53:45 |
| FileSize | 249022 |
| Format | tif |
| FormatVersion | [] |
| Width | 508 |
| Height | 398 |
| BitDepth | 24 |
| ColorType | truecolor |
| FormatSignature | [73 73 42 0] |
| ByteOrder | little-endian |
| NewSubFileType | 0 |
| BitsPerSample | [8 8 8] |
| Compression | PackBits |
| PhotometricInterpretation | RGB |
| StripOffsets | [1x80 double] |
| SamplesPerPixel | 3 |
| RowsPerStrip | 5 |
| StripByteCounts | [1x80 double] |
| XResolution | 72 |
| YResolution | 72 |
| ResolutionUnit | Inch |
| Colormap | [] |
| PlanarConfiguration | Chunky |
| TileWidth | [] |
| TileLength | [] |
| TileOffsets | [] |
| TileByteCounts | [] |
| Orientation | 1 |
| FillOrder | 1 |
| GrayResponseUnit | 0.0100 |
| MaxSampleValue | [255 255 255] |
| MinSampleValue | [0 0 0] |
| Thresholding | 1 |

2. Images en couleur :

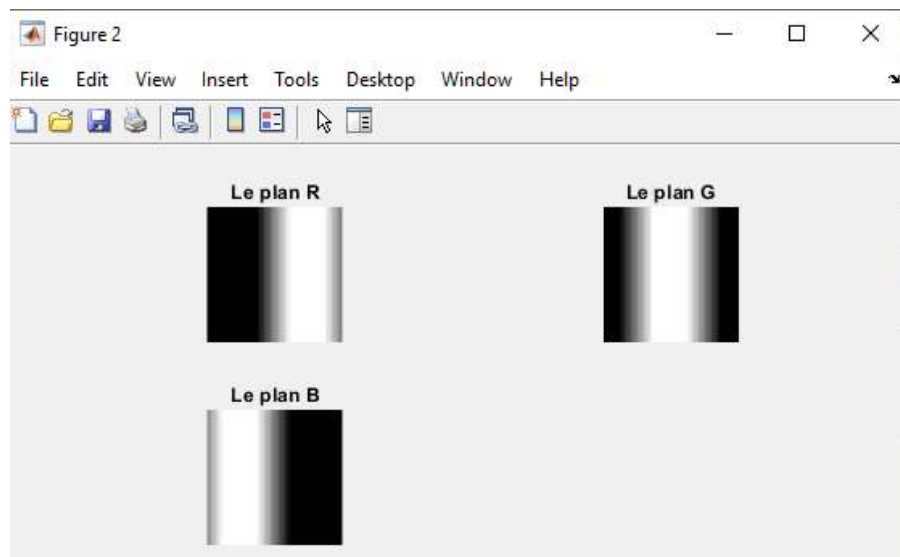
- Création d'une image trueColor contenant les 3 plans RGB

```
RGB=reshape(ones(64,1)*reshape(jet(64),1,192),  
[64,64,3]);
```

- Affichage des trois plans séparément :

```
subplot(3,2,1); imshow(RGB(:,:,1))
subplot(3,2,2); imshow(RGB(:,:,2))
subplot(3,2,3); imshow(RGB(:,:,3))
```



- Commentaires sur les commandes utilisées :

Jet (64) retourne une matrice 64x3 contenant le code des couleurs de la palette de couleur utilisé dans l'image présente

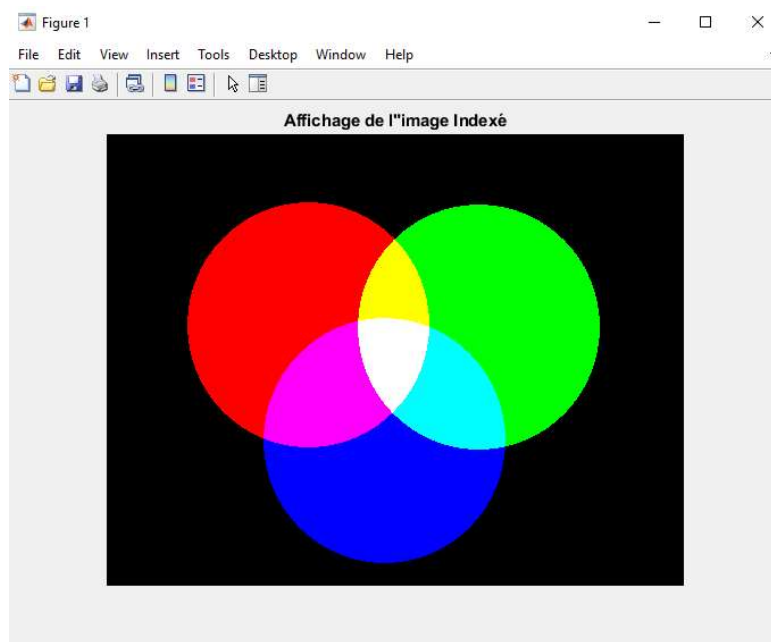
Reshape (vecteur (x0: xn),L,C) : créer une matrice de L ligne et C colonne la première Case de cette matrice est x0 et la dernière xn

3. Images indexées :

- Charger l'image couleurAdditifs.tif sous forme d'image indexée

```
[X, map]=imread('CouleursAdditifs.tif') ;
imwrite([X,map], 'CouleursAdditifIndex.tif');
```

- Afficher cette image et ces caractéristiques :



| Metadata (CouleursAdditifIndex.tif) | |
|-------------------------------------|--|
| Attribute | Value |
| Filename | C:\Users\acer\Desktop\SEMESTRE 2\TI\TP1\image_TP1\CouleursAdditifIndex.tif |
| FileModDate | 19-mai-2021 12:43:37 |
| FileSize | 248918 |
| Format | tif |
| FormatVersion | [] |
| Width | 508 |
| Height | 398 |
| BitDepth | 24 |
| ColorType | truecolor |
| FormatSignature | [73 73 42 0] |
| ByteOrder | little-endian |
| NewSubFileType | 0 |
| BitsPerSample | [8 8 8] |
| Compression | PackBits |
| PhotometricInterpretation | RGB |
| StripOffsets | [1x67 double] |
| SamplesPerPixel | 3 |
| RowsPerStrip | 6 |
| StripByteCounts | [1x67 double] |
| XResolution | 72 |
| YResolution | 72 |
| ResolutionUnit | Inch |
| Colormap | [] |
| PlanarConfiguration | Chunky |
| TileWidth | [] |
| TileLength | [] |
| TileOffsets | [] |
| TileByteCounts | [] |
| Orientation | 1 |
| FillOrder | 1 |
| GrayResponseUnit | 0.0100 |
| MaxSampleValue | [255 255 255] |
| MinSampleValue | [0 0 0] |
| Thresholds | 1 |

- Calcul et comparaison de la taille d'une image RGB avec la même image convertie en image indexé

```
size(X)
size(ima)
```

ils ont la même taille

```
>> size(X)

ans =

    398    508     3

>> size(ima)

ans =

    398    508     3
```

4. La profondeur :

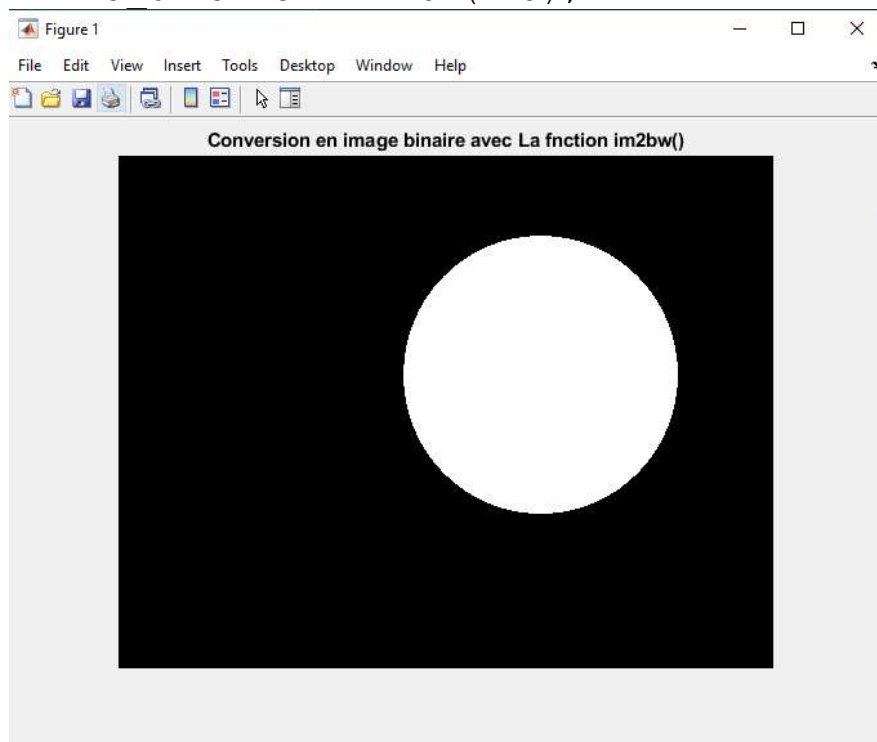
- L'importance de la profondeur pour l'affichage :

La profondeur définit la quantité d'informations chromatiques disponibles pour chaque pixel dans une image. Le nombre de couleurs disponibles et la précision de la représentation des couleurs dans une image sont proportionnels au nombre de bits d'informations par pixel.

Ainsi, dans une image d'une profondeur de 1 bit par pixel, les pixels peuvent prendre deux valeurs possibles : noir et blanc. Une image avec une profondeur de 8 bits par pixel compte 256 valeurs possibles

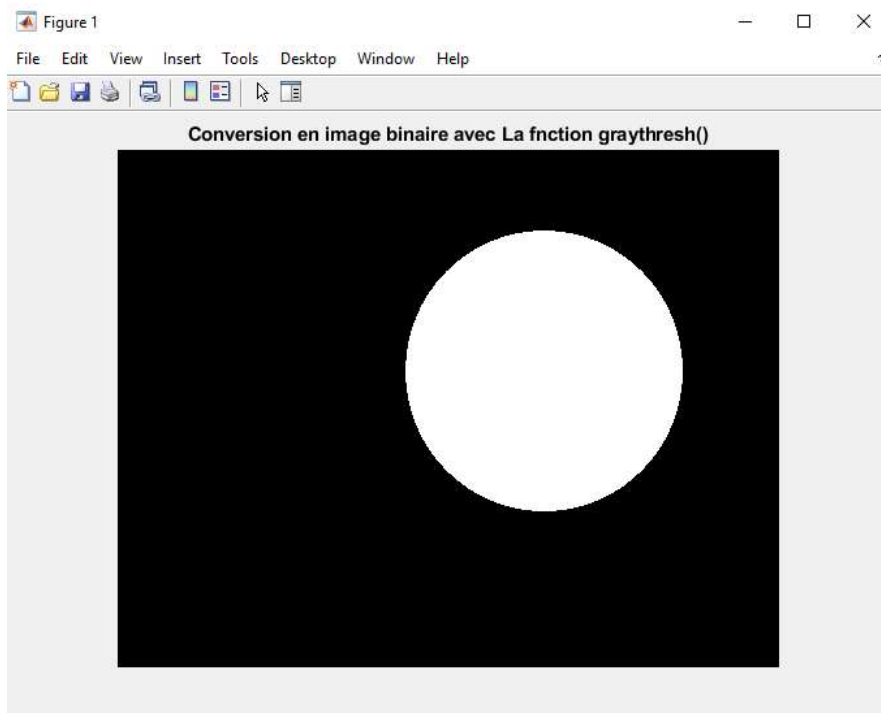
Images Binaire :

- Conversion de l'image binaire en une image indexée
`ima_binaire = im2bw(ima);`

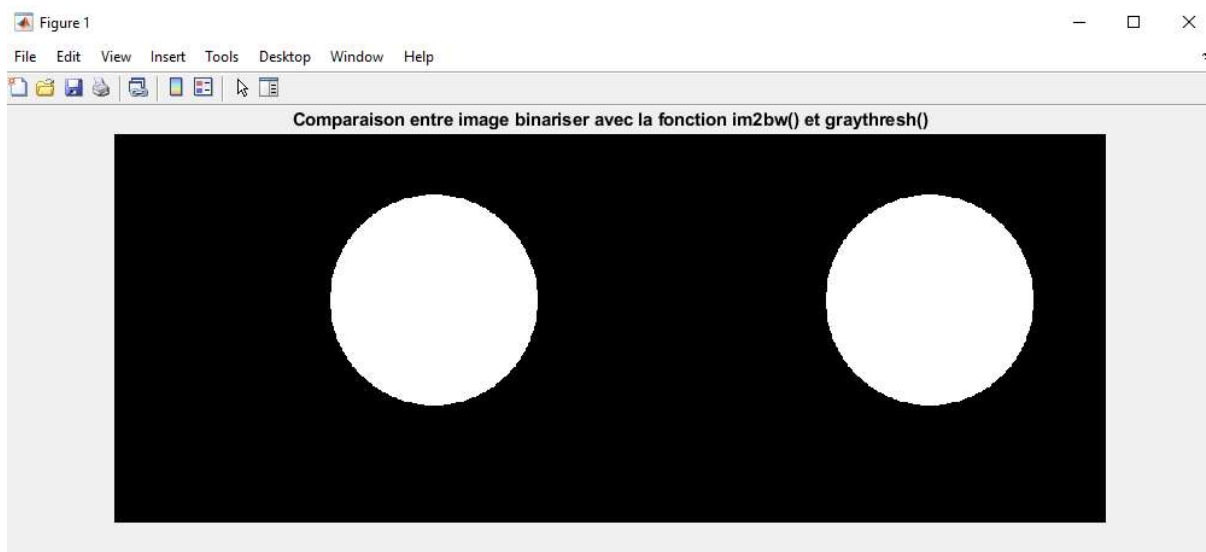


- Binariser l'image avec la commande 'graythresh'

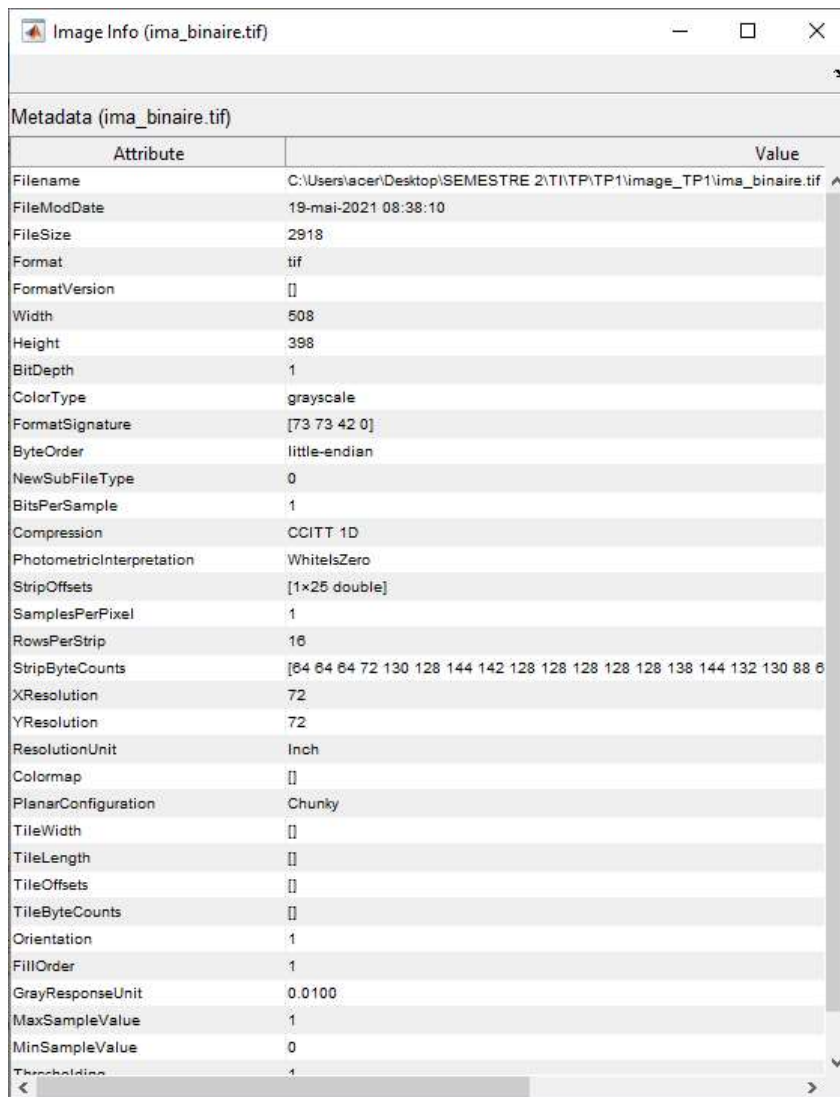
```
I = imread('CouleursAdditifs.tif');  
level = graythresh(I);  
ima_binarise = im2bw(I,level);
```
- Affichage de l'image



- Comparaison avec le résultat précédent
C le même résultat mais il est recommandé d'utiliser IMBINARIZE dans graythresh



- Les caractéristiques de l'image :



Partie 3 : Propriétés D'Une Image

1. Affichage des pixels de la ligne 11 jusqu'à ligne 14 et de la colonne 44 jusqu'à 50 :

```
ima_gray = imread('img_gray_1.jpg');
ima_gray(11:14, 44:50)
```

```
>> ima_gray(11:14, 44:50)
```

```
ans =
```

```
4x7 uint8 matrix
```

```

255    255    255    255    255    255    255
255    255    255    255    255    255    255
255    255    255    255    255    255    255
255    255    255    255    255    255    255
```

Partie de l'image :lignes 11 à 14 , colonne 44 à 50



2. Affichage de la dimension de l'image –On stock le nombre de ligne dans 'nl' et le nombre de colonne dans 'nc'-

```
[nl,nc]=size(ima_gray)
```

```
nl =  
  
    800
```

```
nc =  
  
    800
```

3. Mettre la valeur maximum dans un vecteur maxlig

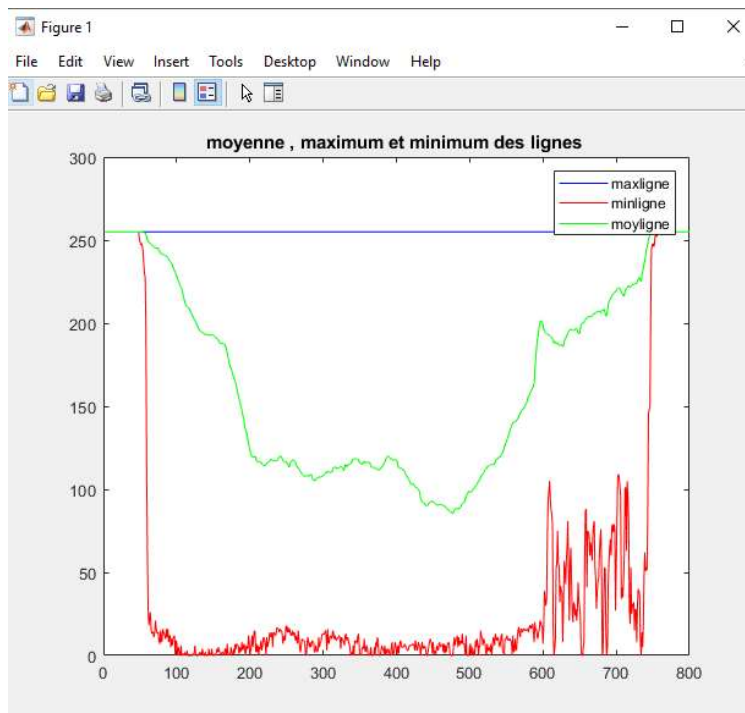
```
maxligne = max(ima_gray ,[], 2) ;
```

4. Faire du même avec le minimum et la moyenne

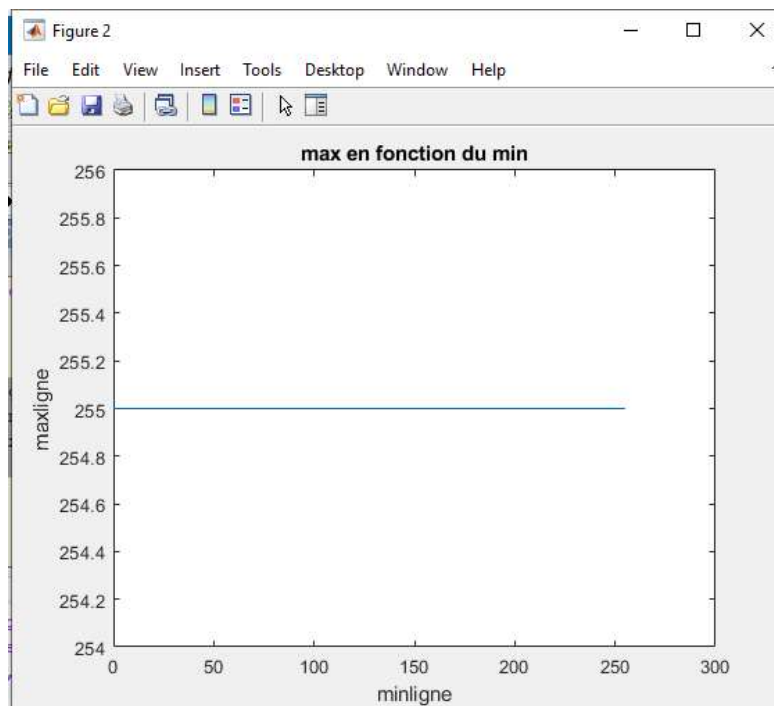
```
minligne = min(ima_gray , [],2) ;  
moyligne = mean(ima_gray,2) ;
```

| | | | |
|----------|-------|------|--------|
| maxligne | 800x1 | 800 | uint8 |
| minligne | 800x1 | 800 | uint8 |
| moyligne | 800x1 | 6400 | double |

5. Le tracé des vecteurs sur le même graphique



6. Le tracé du maximum en fonction du minimum



7. Calculer la moyenne, le max, la min et la somme de toute l'image, puis la partie centrale :
 ⇒ Pour toute l'image :

```
maxx =
    uint8
    255

minn =
    uint8
    0
```

```

moyy =
    166.9731

summ =
    106862812

```

⇒ la partie centrale :

```

max2 =
    uint8
    255

min2 =
    uint8
    15

max2 =
    uint8
    255

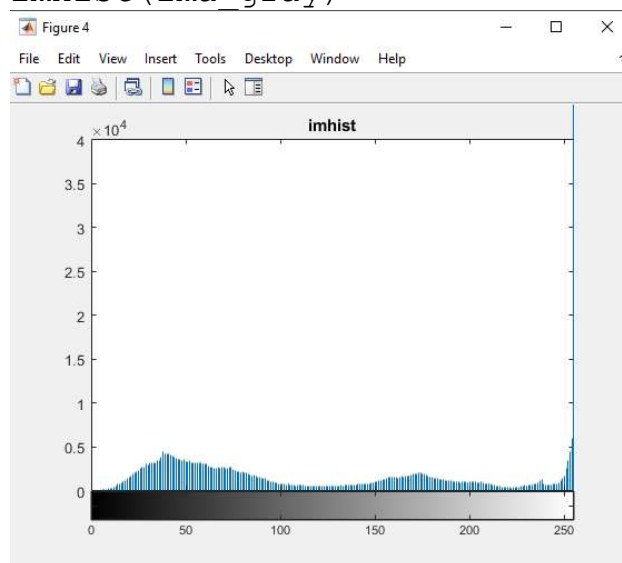
min2 =
    uint8
    15

```

8. Le tracé de l'histogramme de l'image :

➤ Avec imhist

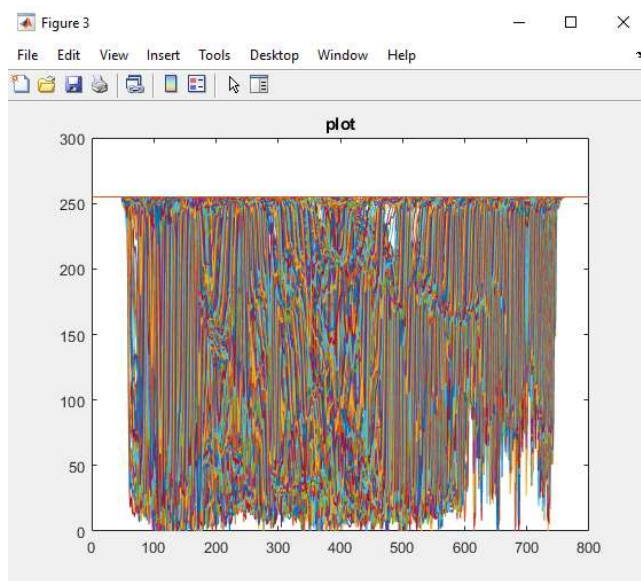
```
imhist(ima_gray)
```



Ce graphe montre la distribution des intensités de l'image niveau de gris. On Remarque un maximum au environ de 40 qui correspond a la parte sombre dominante de l'image

➤ Avec plot

```
plot(ima_gray)
```



CONCLUSION :

Dans ce premier TP, Nous avons abordé une initiation au traitement d'image avec l'outil mathématique MATLAB

Les principaux points abordés dans ce TP étaient :

1. La manipulation des image
2. La conversion entre les différents types d'image
3. Propriétés D'Une Image

Nous avons commencé par l'introduction des fonctions de bases tel que :

La fonction `imread()` pour lire une image en la convertissant en une matrice de pixel

Les fonction `imshow()` `imtool()` `image()` `imagesc()` pour l'affichage de la figure de l'image

La fonction `imwrite()` pour de sauvegarder les illustration

Les types d'image que nous avons traité

1. L'image binaire
2. L'image niveau de gris
3. L'image couleur RGB
4. L'image indexé

Puis on s'est intéressé aux différents conversions qu'on peut effectuer sur les images tel que :

La fonction `double()` : transformation de la résolution spectrale de l'image en double

La fonction `im2double()` : doubler la précision

La fonction `rgb2gray` : conversion d'une image couleur en une image niveau de gris

La fonction `[X,map]` pour la conversion en image indexé

...

Nous avons conclue par un aperçu sur les propriétés des images (en utilisant le max des pixels , le min , la moyenne et la somme ainsi que l'histogramme)