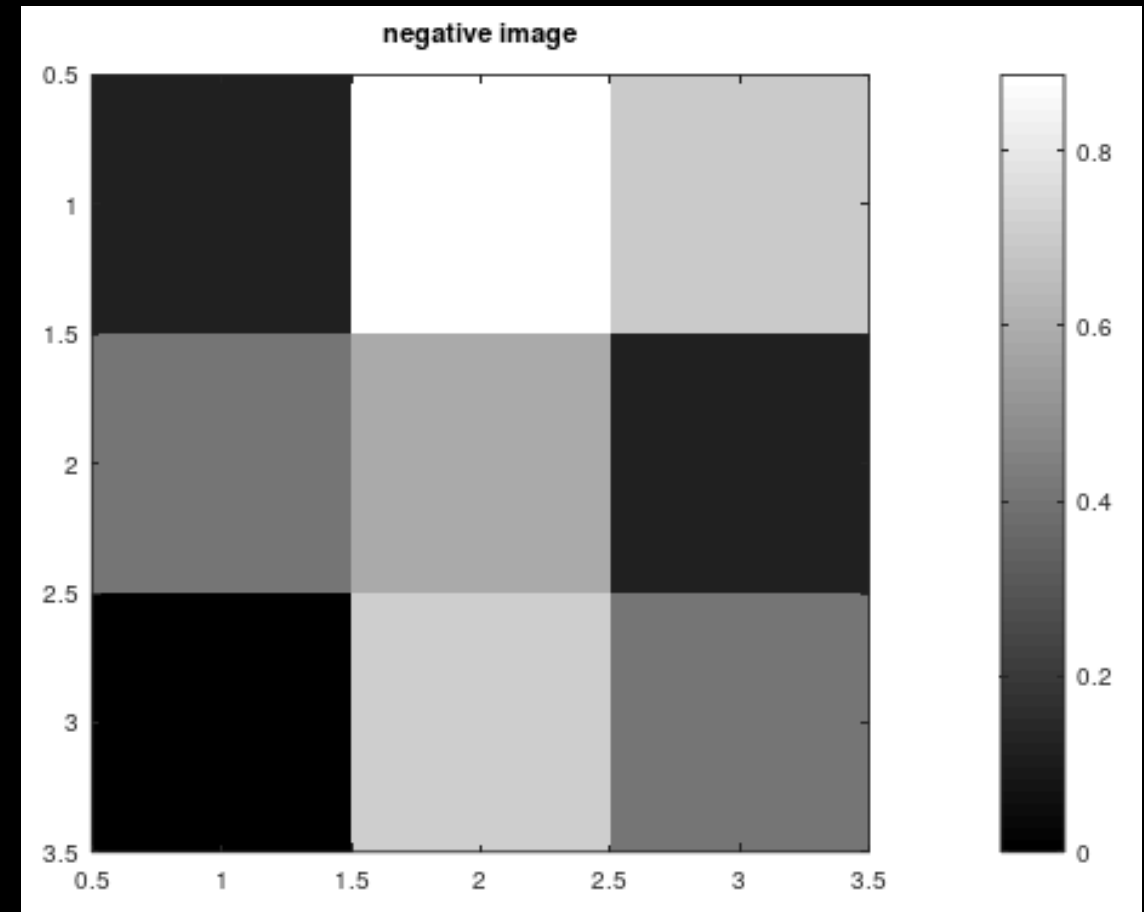
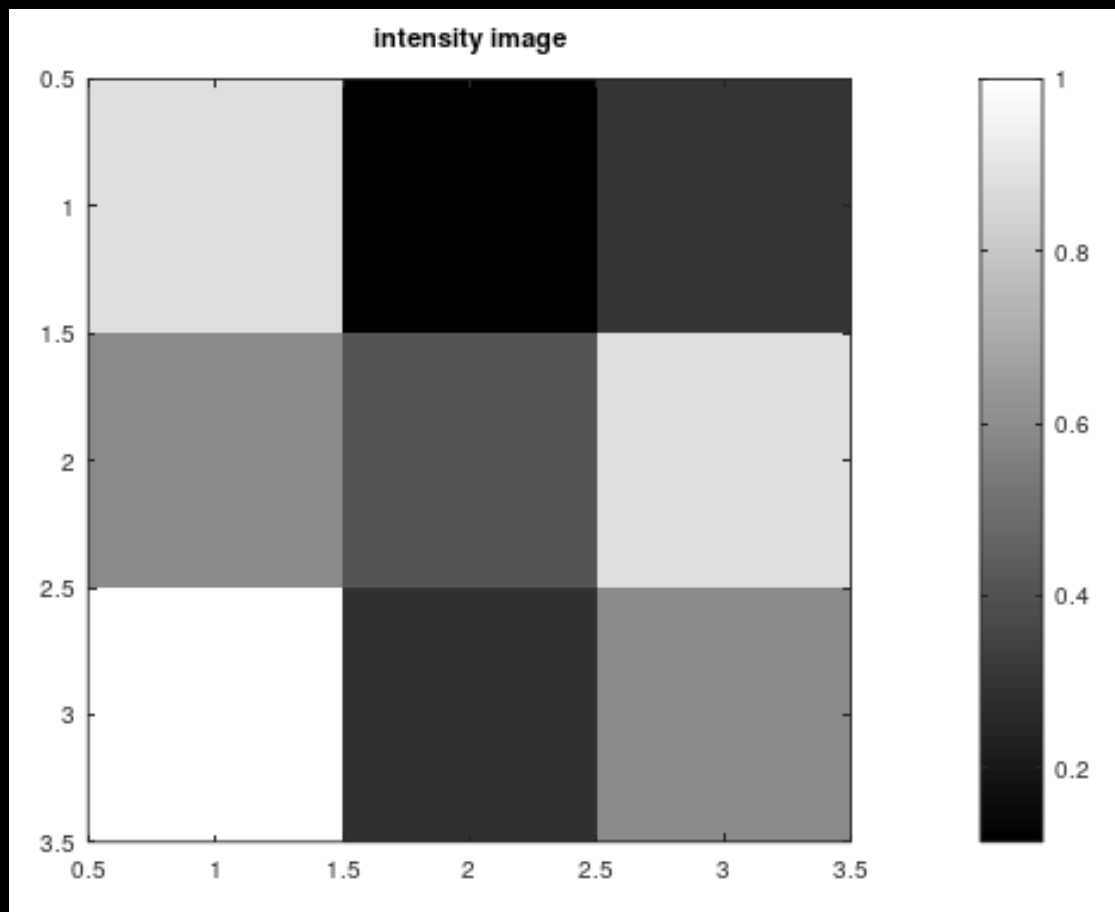


I. Introduction

- **EXERCICE 4** : Représenter le *négatif* de l'image en niveaux de gris de l'exemple précédent (*lint*, le résultat de *ind2gray(ll,map)*).

I. Introduction

```
1 clear
2 close all
3
4 map=[1 1 0; 0 0 1; 1 0 0; 0 1 0; 1 0 1; 1 1 1; 0 .5 0; 1 .5 0];
5 II=[1 2 3; 4 5 1; 6 7 8];
6
7 Iint=ind2gray(II, map);
8 figure(1), imagesc(Iint), axis('square'), title('int. img'), colorbar
9 colormap(gray)
10
11 Ineg=1-Iint;
12 figure(2), imagesc(Ineg), axis('square'), title('neg. img'), colorbar
13 colormap(gray)
```



I. Introduction

4- FORMATS D'IMAGE SUR LE DISQUE

- Avant toute chose :

>> <i>pwd</i>	(pour voir où l'on se trouve)
>> <i>ls</i>	(pour voir ce qu'il y a là)
>> <i>mkdir MATLAB</i>	(pour créer le répertoire <i>MATLAB</i>)
>> <i>cd MATLAB</i>	(pour y aller)
>> <i>mkdir 0-images</i>	(pour créer le répertoire <i>0-images</i>)

puis y mettre dedans les images que vous trouverez sur
<https://lagrange.oca.eu/carbillet/enseignement/M2-GBM/images>

>> <i>cd 0-images</i>	(pour y aller)
>> <i>ls</i>	(pour voir ce qu'il y a là)
>> <i>cd ..</i>	(pour remonter d'un cran)
>> <i>mkdir 1-intro</i>	(pour créer le répertoire <i>1-intro</i>)
>> <i>movefile('exo.m', '1-intro/exo.m')</i>	(pour y mettre <i>exo.m</i>), ou bien :
>> <i>movefile exo.m 1-intro/exo.m</i>	(de manière équivalente)

Si vous avez appelé l'exercice précédent « *exo.m* » et l'avez déposé dans le répertoire *MATLAB*

>> <i>cd 1-intro</i>	(pour y aller)
>> <i>pwd</i>	(pour vérifier où l'on se trouve)
>> <i>ls</i>	(pour voir ce qu'il y a là)

I. Introduction

- Format Matlab/Octave : *.mat

Exemple sur l'image en niveau de gris définie précédemment *lint* (aller préalablement dans *1-intro*)

>> <i>pwd</i>	→	pour voir où l'on est sur le disque
>> <i>ls</i>	→	pour voir ce qu'il y a sur le disque
>> <i>save Image.mat lint</i>		
>> <i>ls</i>	→	« <i>Image.mat</i> » apparaît [disque]
>> <i>clear lint</i>		
>> <i>whos</i>	→	<i>lint</i> n'existe plus ! [matlab]
>> <i>load Image.mat</i>		
>> <i>whos</i>	→	<i>lint</i> est réapparue... [matlab]
>> <i>imshow(lint, 'InitialMagnification', 'fit')</i>		(ou <i>imagesc(lint), colormap(gray)</i>)

Fonctionne aussi pour les images indexées (en sauvant les 2 tableaux *II* et *map* dans le même fichier : *save Image_ind.mat II map*) ou les images RGB.

I. Introduction

- Formats graphiques usuels : jpeg, tiff, png, gif, etc.

Exemple sur *lint* en .tif

```
>> imwrite(lint, 'Image.tif', 'tif') → « Image.tif » apparaît  
>> clear lint  
>> whos → lint n'existe plus !  
>> lint = imread('Image.tif', 'tif');  
>> whos → lint est réapparue...  
>> imshow(lint, 'InitialMagnification', 'fit')
```

'jpg' or 'jpeg'	JPEG — Joint Photographic Experts Group	8-bit, 12-bit, and 16-bit Baseline JPEG images
i Note imwrite converts indexed images to RGB before writing data to JPEG files, because the JPEG format does not support indexed images.		

I. Introduction

- Remarques finales :

- Du *help* de *imwrite* :

```
If the input array is of class double, and the image is a grayscale or RGB color image, imwrite assumes the dynamic range is [0,1] and automatically scales the data by 255 before writing it to the file as 8-bit values.
```

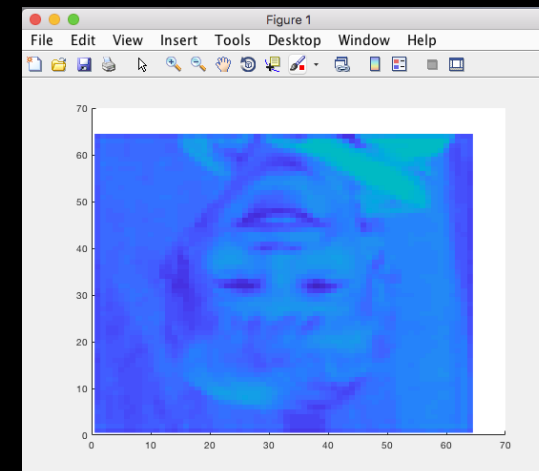
=> quand on charge l'image avec **IMREAD**, elle est donc de type *uint8* (à 256 niveaux de gris de 0 à 255)...

- Plein de TIFF (et autres formats) dans :

*/Applications/MATLAB_R***.app/toolbox/images/indemos/*

- Image « cachée » :

>> image (qui est en fait une erreur !)



II. Analyse élémentaire d'images

1- HISTOGRAMME

- Densité de probabilité des niveaux de gris d'une image = histogramme NORMALISÉ
- En abscisses : le nombre de niveaux de quantification de l'image
- En ordonnées : le nombre de pixels de l'image correspondant à chaque niveau de quantification
- Sous Matlab/Octave : instruction IMHIST
(faire « **>> help imhist** » pour lire les caractéristiques de la fonction (ou aussi « **>> doc imhist** » sous Matlab))
(sous Octave : il faut au préalable avoir chargé le package 'image' à l'aide de « **>> pkg load image** »)

II. Analyse élémentaire d'images

```
-- Function File: imhist (I)
-- Function File: imhist (I, N)
-- Function File: imhist (X, CMAP)
-- Function File: [COUNTS, X] = imhist (...)
    Produce histogram counts of image I.
```

The second argument can either be `N`, a scalar that specifies the number of bins; or `CMAP`, a colormap in which case `X` is expected to be an indexed image. If not specified, `N` defaults to 2 for binary images, and 256 for grayscale images.

If output is requested, `COUNTS` is the number of counts for each bin and `X` is a range for the bins so that `'stem (X, COUNTS)'` will show the histogram.

Note: specially high peaks that may prevent an overview of the histogram may not be displayed. To avoid this, use `'axis "auto y"'` after the call to `'imhist'`.

See also: `hist`, `histc`, `histeq`.

Additional help for built-in functions and operators is available in the online version of the manual. Use the command `'doc <topic>'` to search the manual index.

Help and information about Octave is also available on the WWW at <https://www.octave.org> and via the help@octave.org mailing list.

II. Analyse élémentaire d'images

- Exemple :

```
>> I = [0.5 0.4 0.3 ; 0.4 0.3 0.4 ; 0.3 0.4 0.5];
```

```
>> [n, x] = imhist(I, 11)
```

rend :

```
n = [0  0  0  3  4  2  0  0  0  0  0 ]
```

```
x = [0.0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1.0]
```

```
>> figure, imhist(I, 11)
```

rend : la représentation de cet histogramme

- Représenter à la fois l'image et son histogramme :

```
>> figure, subplot(1, 2, 1), colormap(gray)
```

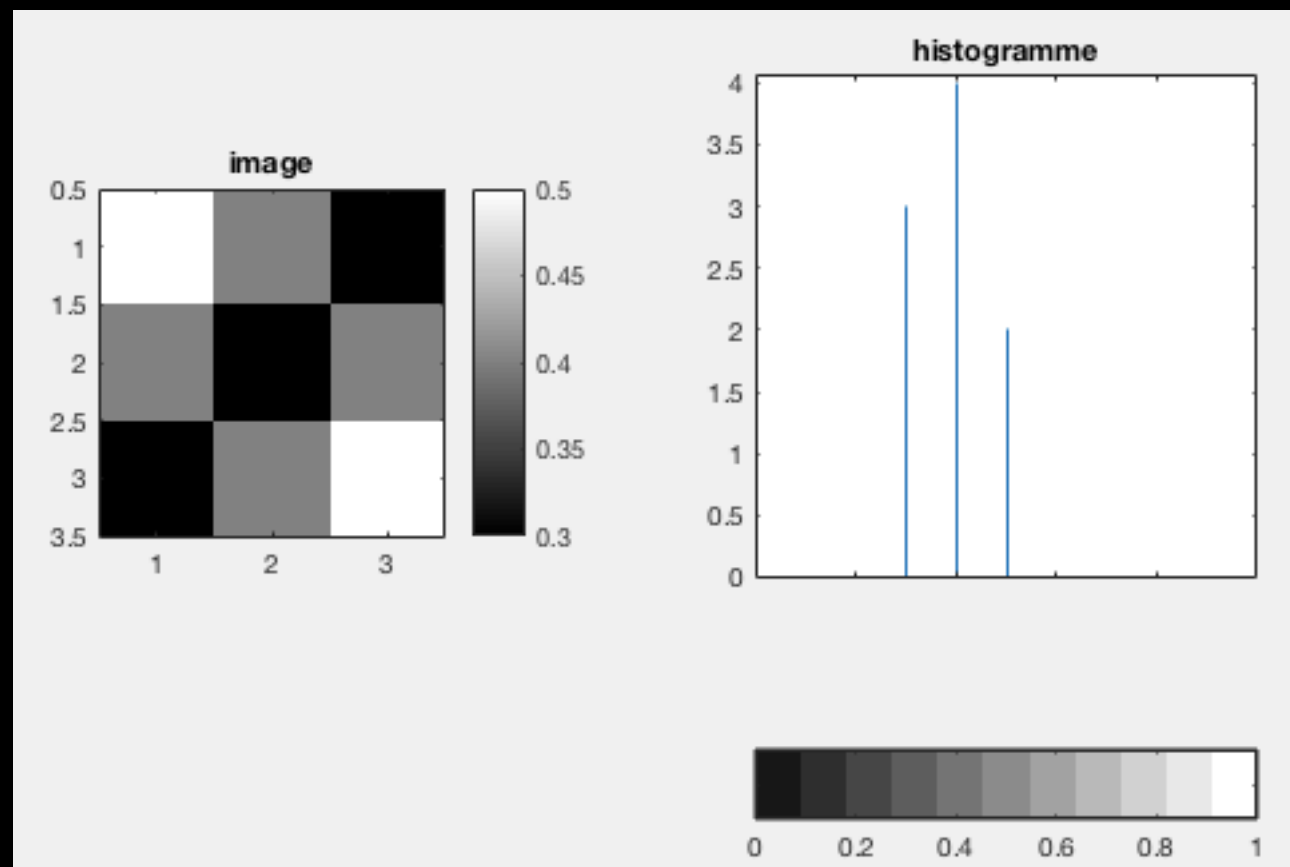
```
>> imagesc(I), colorbar, axis('square')
```

(ou : >> imshow(I, 'InitialMagnification','fit'), colorbar)

```
>> subplot(1, 2, 2), imhist(I, 11)
```

II. Analyse élémentaire d'images

```
1 clear
2 close all
3 |
4 I=[.5 .4 .3; .4 .3 .4; .3 .4 .5]
5 [n, x] = imhist(I, 11);
6 'niveaux de quantification dans l'image = ', x
7 'nb de px dans chq niveaux = ', n
8
9 figure
10 subplot(1,2,1)
11 imagesc(I), colormap(gray), colorbar, axis('square'), title('image')
12 subplot(1,2,2)
13 imhist(I, 11), axis('square'), title('histogramme')
```



II. Analyse élémentaire d'images

- **EXERCICE 1** : Lire une image plus compliquée (p.ex. frog.jpg) en couleur. La convertir en image d'intensité puis tracer son histogramme ainsi que celui de son négatif, sur 256 niveaux. Comparer.

II. Analyse élémentaire d'images

```
1 clear
2 close all
3
4 frog=imread('/Users/marcel/Documents/MATLAB/GBM/0-images/frog.jpg');
5 'type(frog) : ', whos frog
6
7 frog_int=rgb2gray(frog);
8 'type(frog_int) : ', whos frog_int
9 'min(frog_int) : ', min(min(frog_int))
10 'max(frog_int) : ', max(max(frog_int))
11
12 frog_neg=255-frog_int;
13 'min(frog_neg) : ', min(min(frog_neg))
14 'max(frog_neg) : ', max(max(frog_neg))
15
16 figure
17 subplot(2,2,1)
18 imagesc(frog_int), colormap(gray), title('original'), colorbar
19 subplot(2,2,2), imhist(frog_int, 256)
20 subplot(2,2,3)
21 imagesc(frog_neg), colormap(gray), title('negative'), colorbar
22 subplot(2,2,4), imhist(frog_neg, 256)
```

