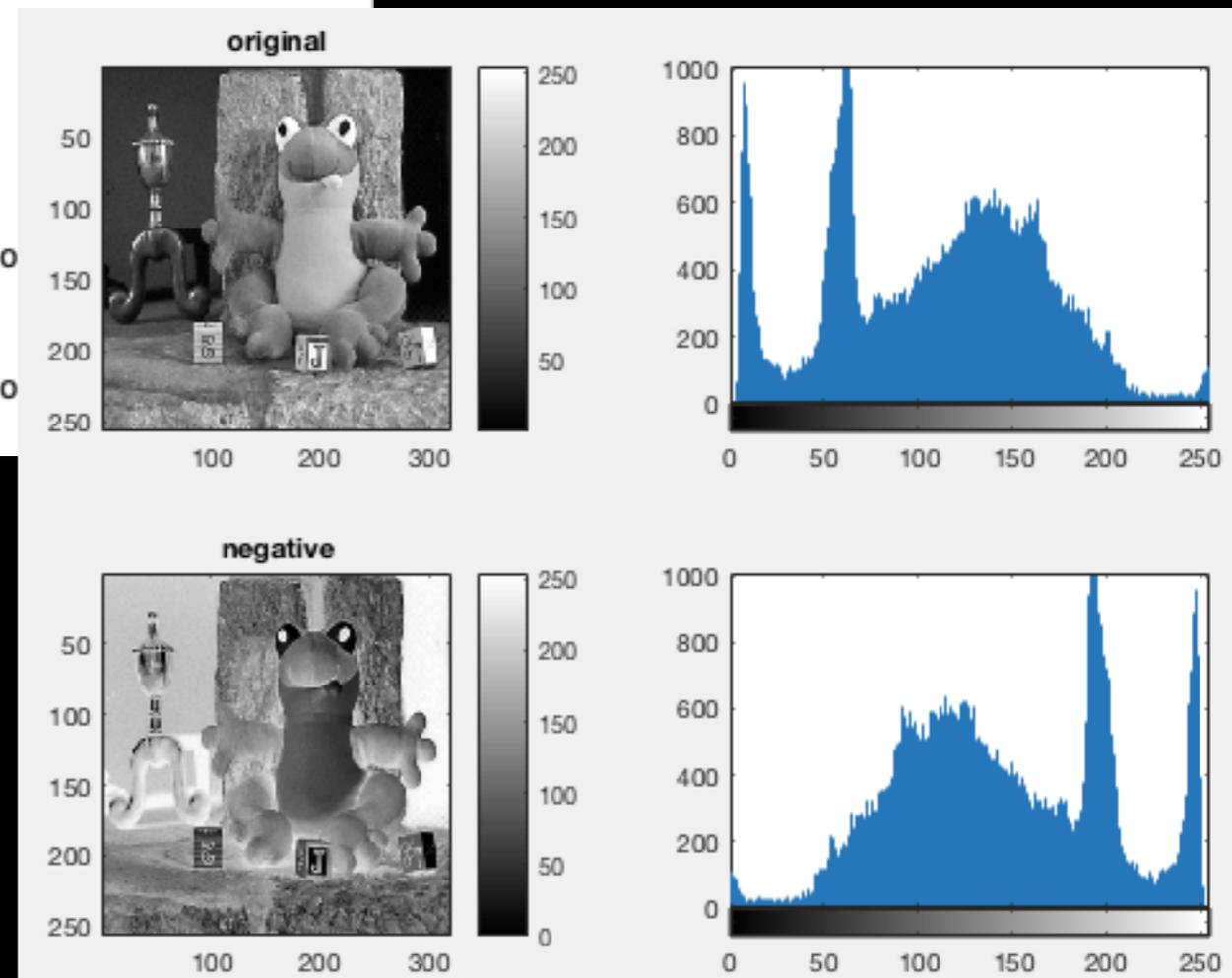


# II. Analyse élémentaire d'images

- **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)
```



# II. Analyse élémentaire d'images

## 2- ÉGALISATION D'HISTOGRAMME

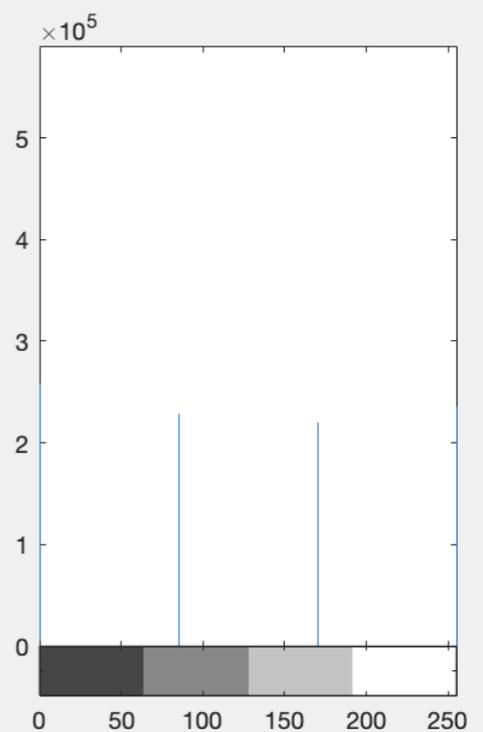
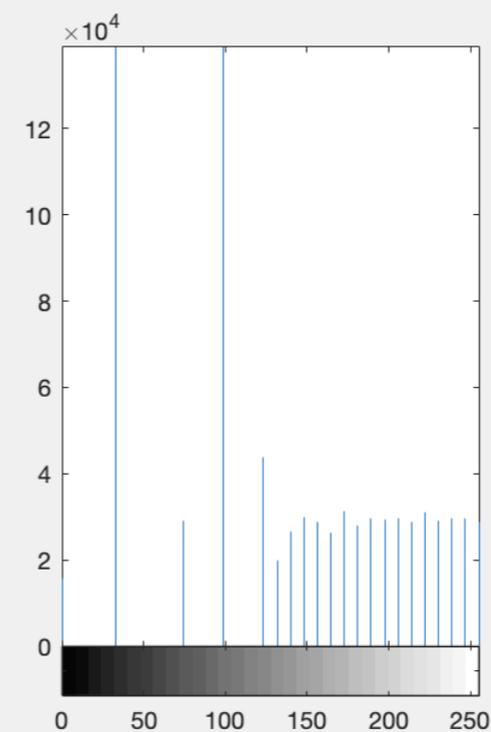
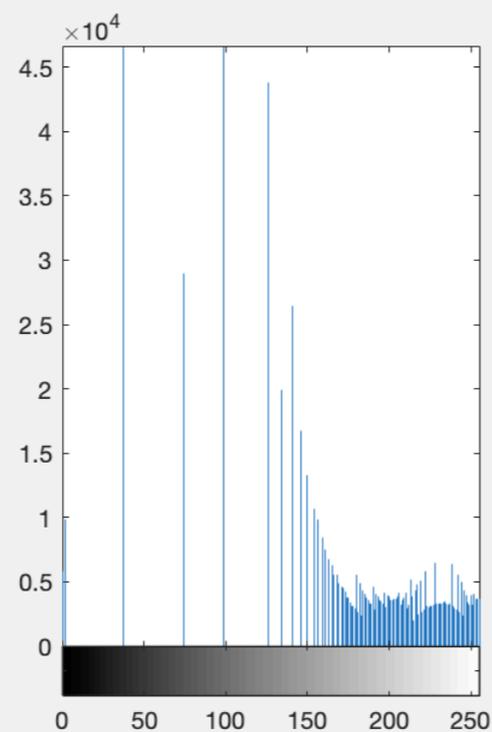
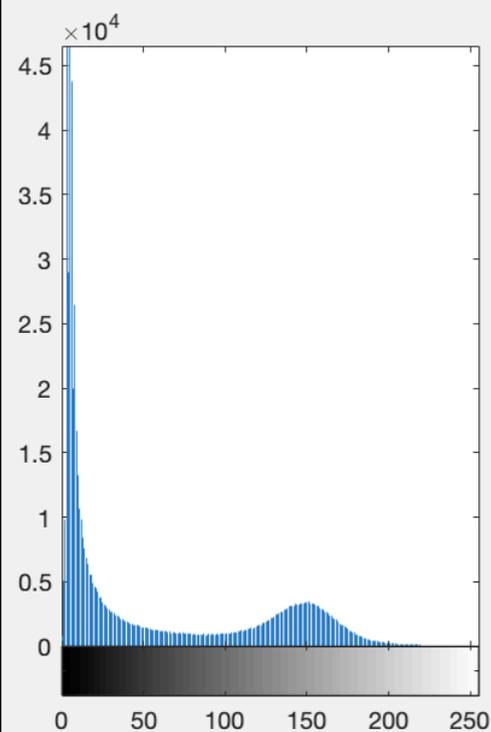
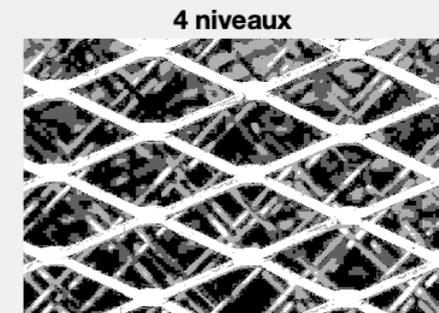
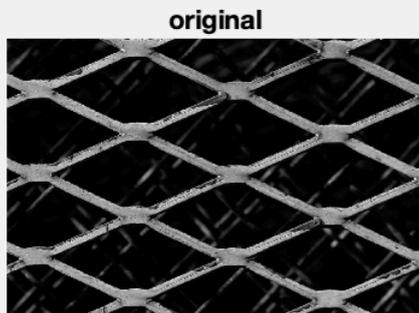
- Peut améliorer des images de mauvaise qualité (contraste : trop sombre/ trop claire, mauvaise répartition des niveaux d'intensité).
- But : rendre l'histogramme d'une image aussi plat que possible afin que chaque niveau d'intensité soit équitablement représenté (et ainsi profiter de toute la dynamique possible).
- Moyen : une transformation  $f$  des niveaux d'intensité qui pour l'intensité  $i$  d'un pixel donné dans l'image originale va rendre  $f(i)$  dans l'image que l'on va appeler « égalisée ».
- En général, on choisit une fonction  $f$  en escalier, en déterminant largeur et hauteur de chaque marche de manière à aplanir au mieux l'histogramme.
- Sous Matlab/Octave : instruction HISTEQ  
`>> frog_eq = histeq(frog_int, n);`  
avec :  $n$  = nombre de niveaux d'intensité dans l'image égalisée

# II. Analyse élémentaire d'images

- **EXERCICE 2 : Égaliser l'histogramme d'une image de votre choix (p.ex. le grillage rouillé en niveaux de gris). Comparer avant et après (images et histogrammes). Égaliser avec moins de niveaux, que se passe-t-il ?...**

```
1 clear
2 close all
3 |
4 I=imread('/Users/marcel/Documents/MATLAB/GBM/0-images/grillage.jpeg');
5 Iint=rgb2gray(I);
6 figure, subplot(2,4,1), imshow(Iint), title('original')
7 subplot(2,4,5), imhist(Iint, 256)
8
9 n = 256;
10 Ieq = histeq(Iint, n);
11 subplot(2,4,2), imshow(Ieq), title('256 niveaux')
12 subplot(2,4,6), imhist(Ieq, n)
13
14 n = 32;
15 Ieq = histeq(Iint, n);
16 subplot(2,4,3), imshow(Ieq), title('32 niveaux')
17 subplot(2,4,7), imhist(Ieq, n)
18
19 n = 4;
20 Ieq = histeq(Iint, n);
21 subplot(2,4,4), imshow(Ieq), title('4 niveaux')
22 subplot(2,4,8), imhist(Ieq, n)
```

# II. Analyse élémentaire d'images

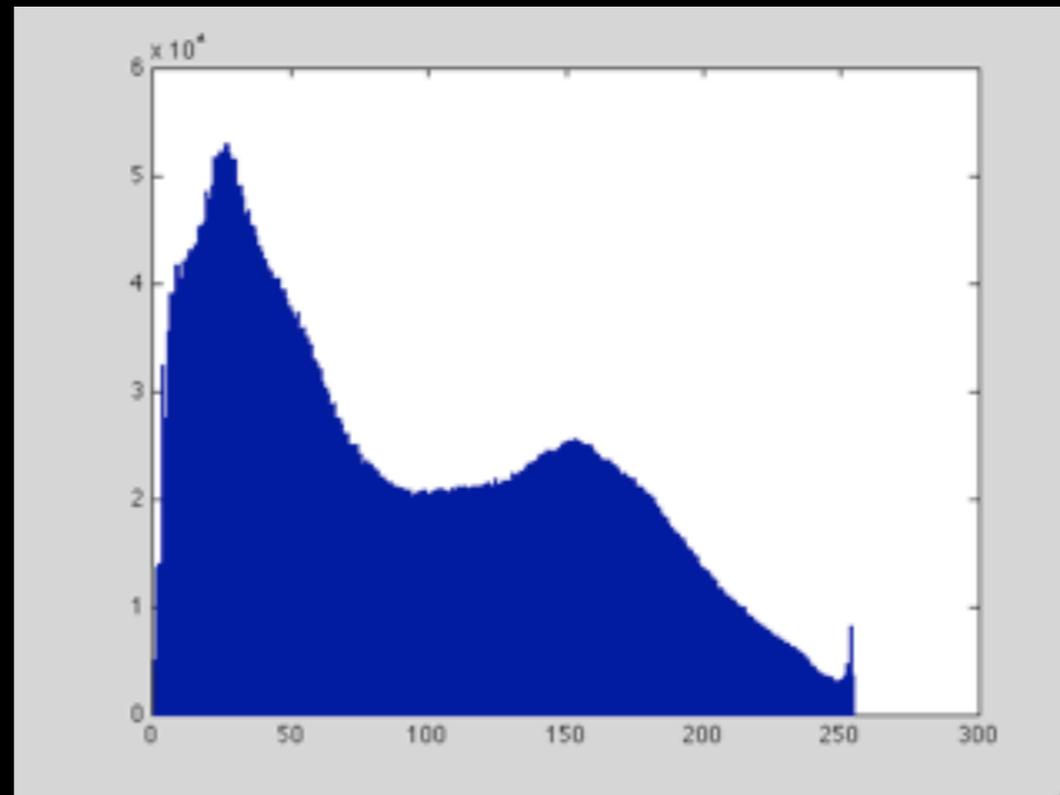


# II. Analyse élémentaire d'images

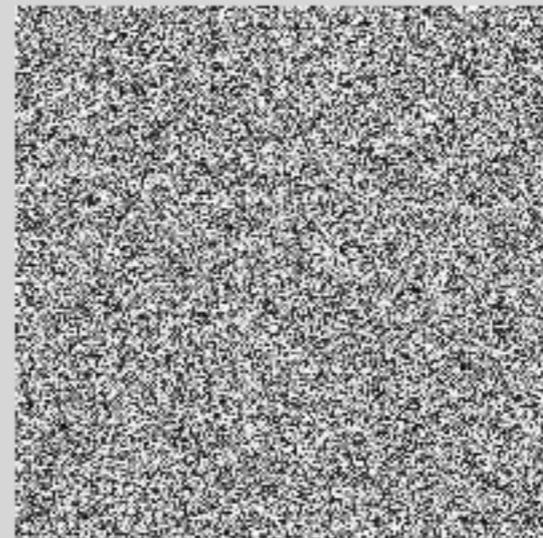
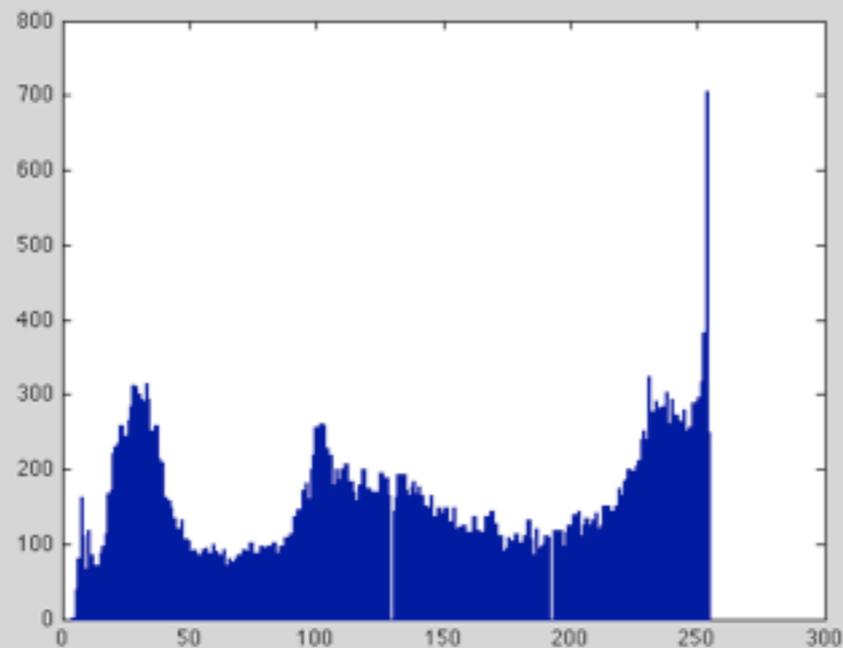
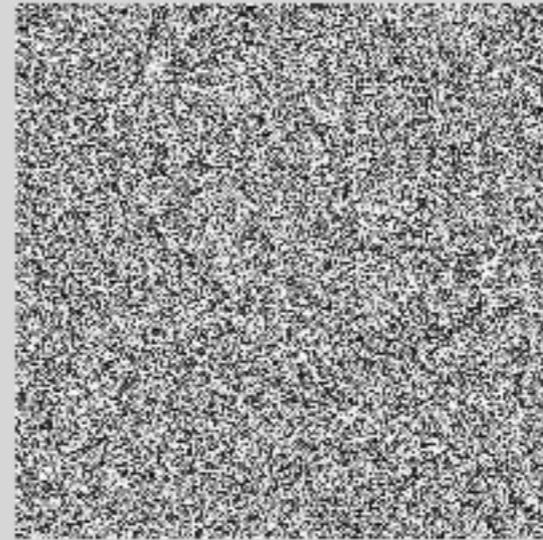
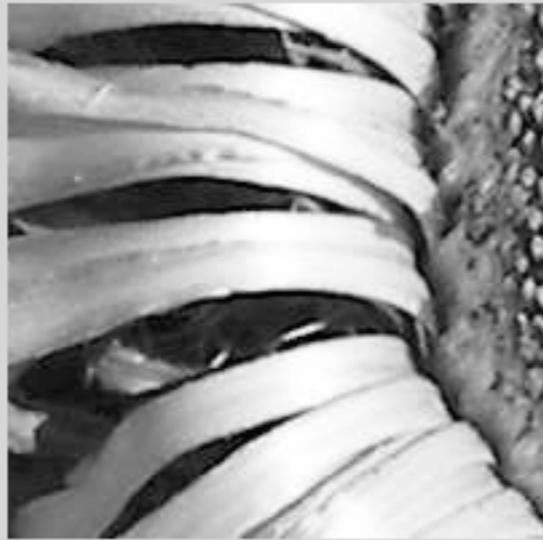
## 3- REMARQUE

- **L'histogramme représente une analyse en un seul point !**
  - => négligence de toute corrélation entre les points (voisins ou pas)**
  - => pas de notion de forme, de détails, de fréquence spatiale**
  - => on peut redistribuer tous les pixels d'une image, la rendant méconnaissable, complètement différente ou même sans forme notable, sans texture aucune, elle aura toujours le même histogramme.**
- **Pour avoir une idée de la texture : analyse en 2 points nécessaire**
  - => densité de probabilité bi-dimensionnelle, i.e. co-occurrence de valeurs en deux points d'une image...**

# II. Analyse élémentaire d'images



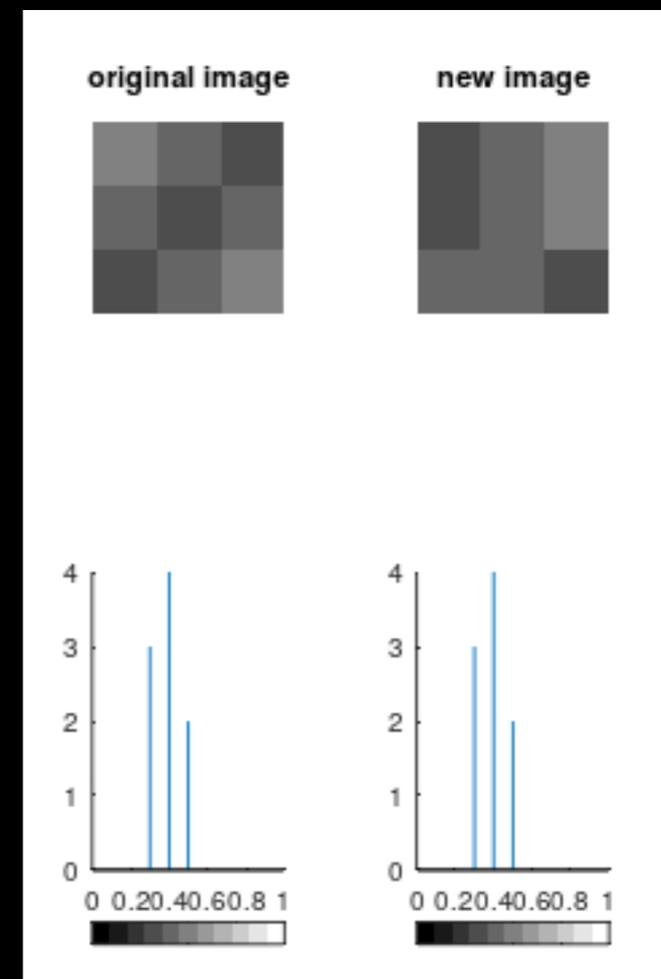
# II. Analyse élémentaire d'images



# II. Analyse élémentaire d'images

- **EXERCICE 3** : Reprendre l'image  $I$  du début du chapitre, calculer son histogramme, puis redistribuer les valeurs des pixels dans une nouvelle image (ou même plusieurs). Calculer le nouvel histogramme. Comparer images et histogrammes.

```
1 clear
2 close all
3
4 I=[.5 .4 .3 ; .4 .3 .4 ; .3 .4 .5];
5 figure
6 subplot(2,4,1), imshow(I), title('original image')
7 subplot(2,4,5), imhist(I,11)
8
9 I2=[.3 .4 .5 ; .3 .4 .5 ; .4 .4 .3];
10 subplot(2,4,2), imshow(I2), title('new image')
11 subplot(2,4,6), imhist(I2,11)
```



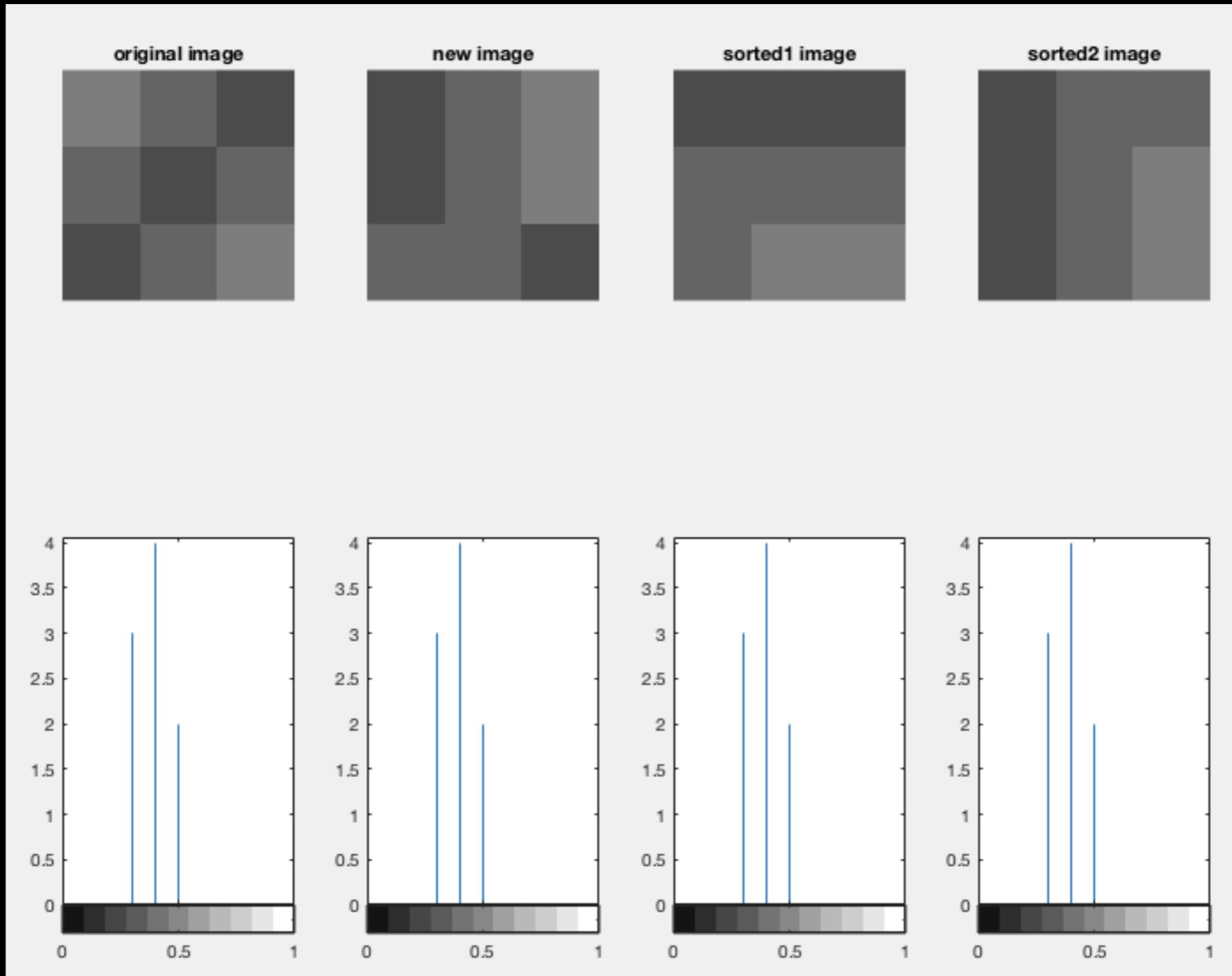
# II. Analyse élémentaire d'images

- **EXERCICE 3 (suite) : Trouver une fonction pouvant transcoder l'image et vérifier que l'on a toujours le même histogramme quelque soit la distribution spatiale des pixels.**

plusieurs solutions possibles : `sort(sort())`, `fliplr/flipud`, `rot90`, `shuffle (randperm)`, etc.

```
1 clear
2 close all
3
4 I=[.5 .4 .3 ; .4 .3 .4 ; .3 .4 .5];
5 figure
6 subplot(2,4,1), imshow(I), title('original image')
7 subplot(2,4,5), imhist(I,11)
8
9 I2=[.3 .4 .5 ; .3 .4 .5 ; .4 .4 .3];
10 subplot(2,4,2), imshow(I2), title('new image')
11 subplot(2,4,6), imhist(I2,11)
12
13 I3=sort(sort(I),2)
14 subplot(2,4,3), imshow(I3), title('sorted1 image')
15 subplot(2,4,7), imhist(I3,11)
16
17 I4=sort(sort(I,2))
18 subplot(2,4,4), imshow(I4), title('sorted2 image')
19 subplot(2,4,8), imhist(I4,11)
```

# II. Analyse élémentaire d'images



# II. Analyse élémentaire d'images

- **EXERCICE 3bis : Appliquer à une image plus complexes (p.ex. la grenouille en niveaux de gris)...**

pour rappel, plusieurs solutions possibles : `sort(sort())`, `fliplr/flipud`, `rot90`, `shuffle (randperm)`, etc.

# II. Analyse élémentaire d'images

```
clear
close all

frog=imread('/Users/marcel/Documents/MATLAB/0-images/frog.jpg');
frog=rgb2gray(frog);

whos frog

figure(1)
subplot(2,3,1), imshow(frog), title('original image')
subplot(2,3,4), imhist(frog,256)

frog1=sort(sort(frog),2);
subplot(2,3,2), imshow(frog1), title('sorted1 image')
subplot(2,3,5), imhist(frog1,256)

frog2=sort(sort(frog,2));
subplot(2,3,3), imshow(frog2), title('sorted2 image')
subplot(2,3,6), imhist(frog2,256)

figure(2)
frog3=rot90(frog);
subplot(2,3,1), imshow(frog3), title('rot90(frog)')
subplot(2,3,4), imhist(frog3,256)

frog4=fliplr(frog);
subplot(2,3,2), imshow(frog4), title('fliplr(frog)')
subplot(2,3,5), imhist(frog4,256)

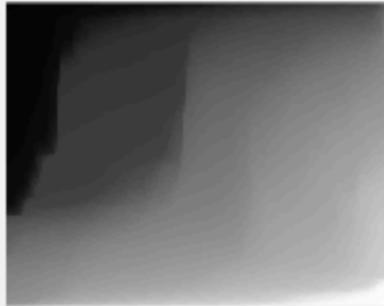
frog5=flipud(frog);
subplot(2,3,3), imshow(frog5), title('flipud(frog)')
subplot(2,3,6), imhist(frog3,256)
```

# II. Analyse élémentaire d'images

original image



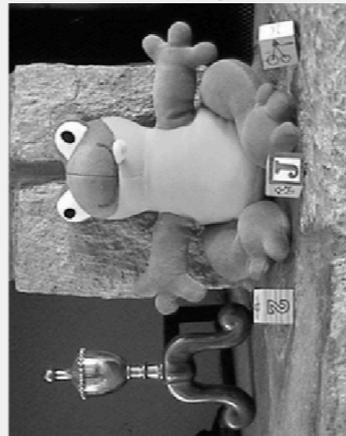
sorted1 image



sorted2 image



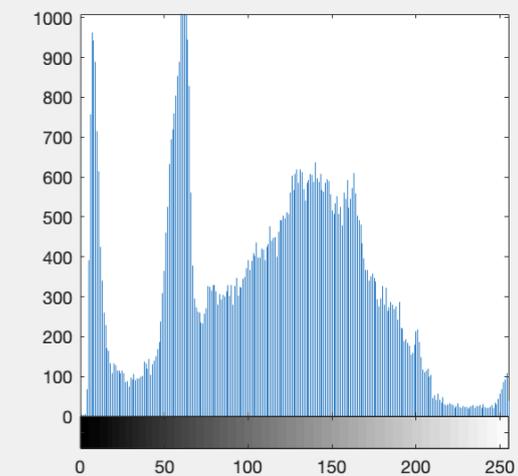
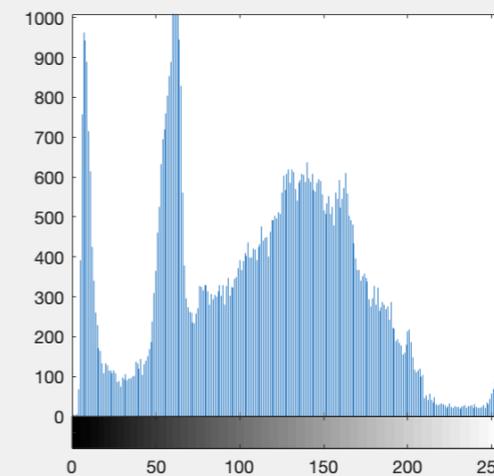
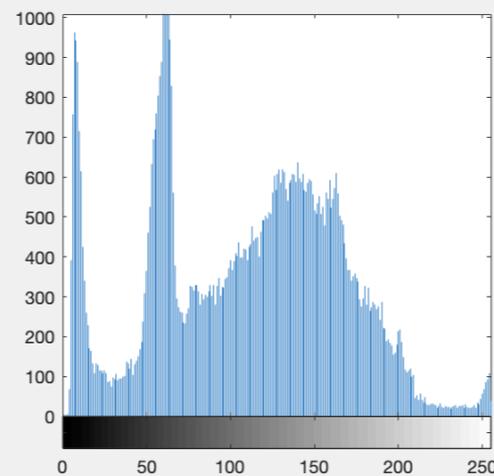
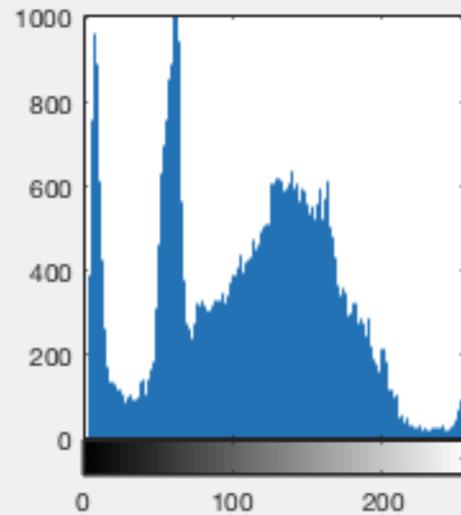
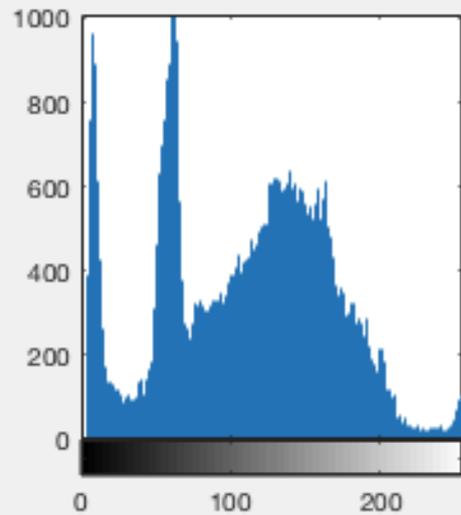
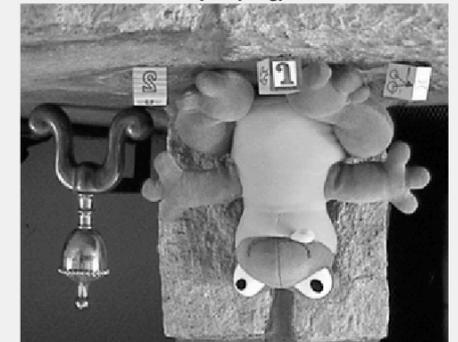
rot90(frog)



fliplr(frog)



flipud(frog)



# II. Analyse élémentaire d'images

## 4- COUPE D'UNE IMAGE

- Afficher une image, lancer *IMPROFILE*, un click pour le 1er point, deux clicks pour le 2me. (Pas encore implémenté sous Octave...)
- En alternative : *PLOT* ( p.ex. : `>> plot(frog_int(:,20))` → plot de la col. n. 20)

# II. Analyse élémentaire d'images

## 5- AMÉLIORATION DU CONTRASTE

- ***IMADJUST*** : sature 1% des basses et hautes intensités et fait en sorte d'utiliser toute la dynamique disponible (en redistribuant les intensités pour utiliser toute la gamme de représentation permise (par uint8 p.ex.)).  
**>> *I\_adj = imadjust(I)***
- On peut aussi choisir la gamme de valeurs d'intensité sur laquelle agir :  
**>> *I\_adj2 = imadjust(I, [0, 0.5], [])***  
(« *[0, 0.5]* » correspondant à [0, moitié des niveaux possibles] - p.ex. !)

# II. Analyse élémentaire d'images

- **EXERCICE 4** : Reprendre l'image du grillages et comparer le résultat de l'égalisation d'histogramme (sur 256 niveaux) à celui d'IMADJUST. Calculer et afficher la valeur moyenne de chaque image et comparer image, histogramme et valeur moyenne de l'image.
- **EXERCICE 4bis** : Toujours en utilisant IMADJUST, adapter la gamme à la première bosse de l'histogramme (celle correspondant aux valeurs de l'intensité entre 0 et 20/255). Que se passe-t-il ?

# II. Analyse élémentaire d'images

```
clear
close all

% original image
I=imread('/Users/marcel/Documents/MATLAB/0-images/grillage.jpeg');
I_int=rgb2gray(I);
mean_int = mean(mean(I_int));

figure(1)
subplot(2,4,1), imagesc(I_int), colormap(gray), colorbar, axis('image')
title({'image originale' ; ['mean=',int2str(mean_int)]})
subplot(2,4,2), imhist(I_int, 256), axis('square')

% equalized image (256)
I_eq = histeq(I_int, 256);
mean_eq = mean(mean(I_eq));

subplot(2,4,3), imagesc(I_eq), colormap(gray), colorbar, axis('image')
title({'image "égalisée" (256 niv.)' ; ['mean=',num2str(mean_eq)]})
subplot(2,4,4), imhist(I_eq, 256), axis('square')

% "adjusted" image
I_adj = imadjust(I_int);
mean_adj = mean(mean(I_adj));

subplot(2,4,5), imagesc(I_adj), colormap(gray), colorbar, axis('image')
title({'image "ajustée" [0,255/255]' ; ['mean=',int2str(mean_adj)]})
subplot(2,4,6), imhist(I_adj, 256), axis('square')

% "adjusted" image with selected input range to adjust
I_adj2 = imadjust(I_int, [0., 20./255], []);
mean_adj2 = mean(mean(I_adj2));

subplot(2,4,7), imagesc(I_adj2), colormap(gray), colorbar, axis('image')
title({'image "ajustée" [0, 20/255]' ; ['mean=',int2str(mean_adj2)]})

subplot(2,4,8), imhist(I_adj2, 256), axis('square')
```

# II. Analyse élémentaire d'images

