

# IMA201 - 2023

## TP3

### 1 Détection de contours

#### 1.1 Filtre de gradient local par masque

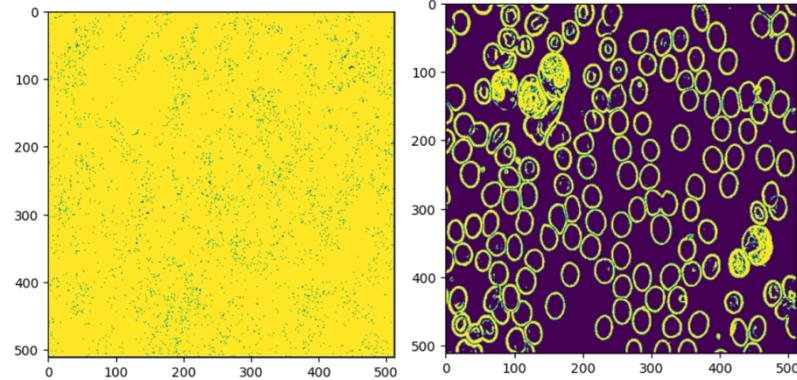
- Rappelez l'intérêt du filtre de Sobel, par rapport au filtre différence, qui calcule une dérivée par la simple différence entre deux pixels voisins

Une image peut être considérée comme une fonction discrète  $f(x,y) = z$  où  $(x,y)$  sont les pixels de l'image et  $z$  la couleur du pixel. Le filtre de différence peut être considéré comme une dérivée ou une différence finie (dans le cas discret) sur l'axe des  $x$  de l'image puisque nous calculons la différence des pixels adjacents sur cet axe. Cependant, l'intérêt du filtre de Sobel est dû au fait que Sobel calcule la dérivée ou la différence sur les deux axes et qu'en outre, il revient à calculer la moyenne des pixels adjacents avant de calculer la dérivée.

$$\mathbf{G}_x = \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} * ([+1 \ 0 \ -1] * \mathbf{A}) \quad \text{and} \quad \mathbf{G}_y = \begin{bmatrix} +1 \\ 0 \\ -1 \end{bmatrix} * ([1 \ 2 \ 1] * \mathbf{A})$$

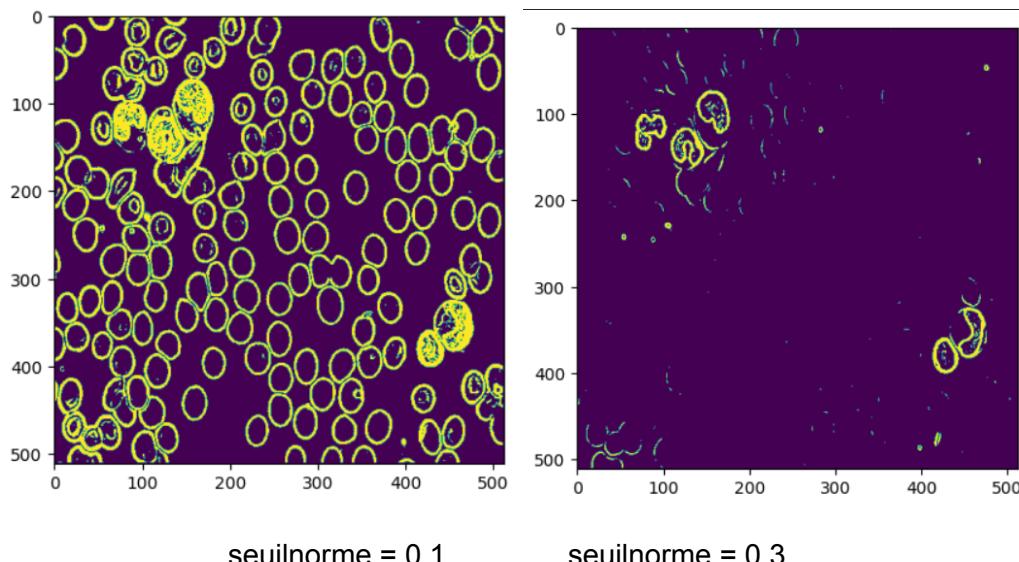
- Est-il nécessaire de faire un filtre passe-bas de l'image avant d'utiliser le filtre de Sobel ?

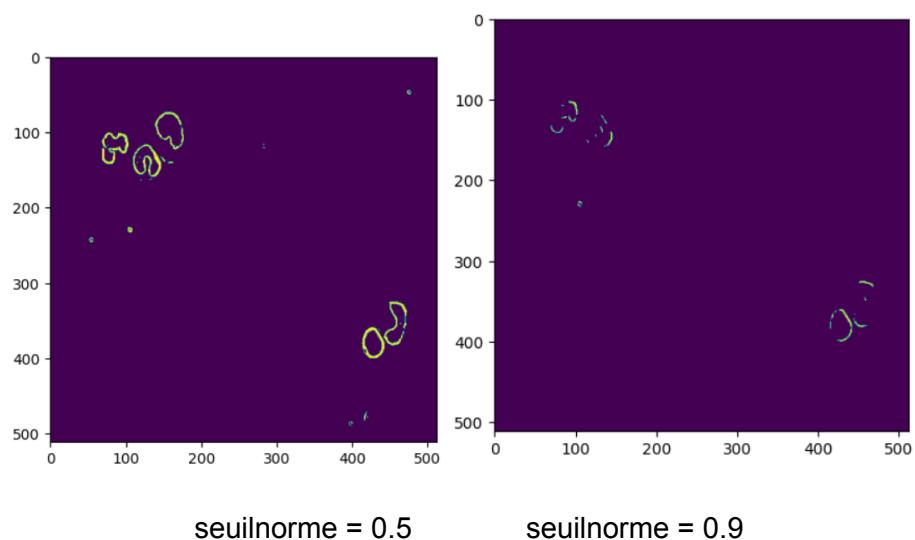
Oui, c'est nécessaire dans la mesure où nous cherchons à obtenir une meilleure qualité de détection des contours de l'image car le filtre de Sobel est très sensible au bruit car un fort contraste dans une image signifie une haute fréquence, un bruit fort est donc une haute fréquence, donc lorsque nous passons l'image à travers un filtre passe-bas, nous perdons une partie de la qualité et du contraste originaux de l'image mais nous éliminons également les bruits les plus forts et les plus marqués de l'image.



Sans filtre passe-bas      Avec filtre passe-bas

- Le seuillage de la norme du gradient permet d'obtenir des contours. Commentez la qualité des contours obtenus (robustesse au bruit, continuité, épaisseur, position...) quand l'on fait varier ce seuil.





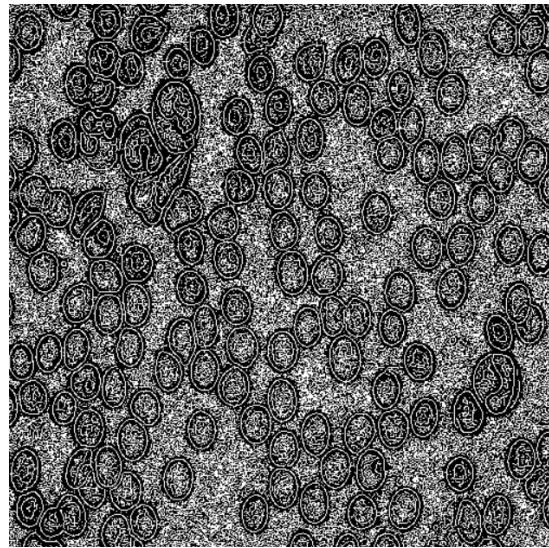
Plus le seuil est élevé, moins il y a de bruit, mais moins les contours sont visibles, car seuls ceux qui présentent le contraste le plus élevé sont affichés, ce qui n'est pas forcément favorable, car les contours moins forts qui sont encore des contours sont ignorés. En outre, plus le seuil est élevé, plus la continuité est perdue car, bien que deux fragments puissent faire partie du même contour, ces deux parties ne sont pas nécessairement situées à proximité immédiate et leurs pixels voisins sont très différents.

## 1.2 Maximum du gradient filtré dans la direction du gradient

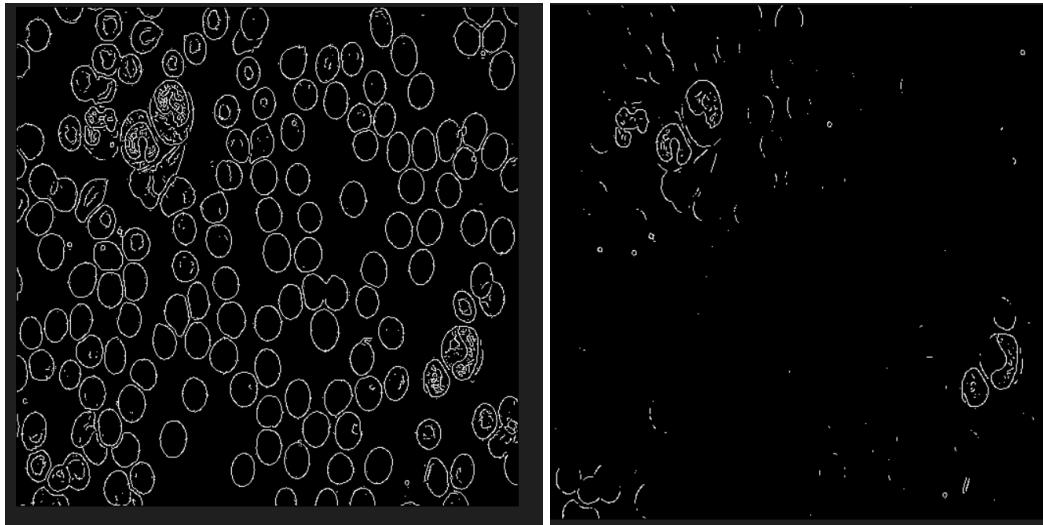
**La fonction maximaDirectionGradient permet de déterminer les pixels de l'image qui sont des maxima du gradient dans la direction du gradient.**

- **Quel critère de qualité est optimisé par ce procédé ?**

Permet de réduire la largeur des bords détectés et d'obtenir des bords plus précis



- Il est possible d'éliminer les contours dont la norme est inférieure à un seuil donné. Commentez les résultats obtenus en terme de position et de continuité des contours, et de robustesse au bruit en faisant varier ce seuil.

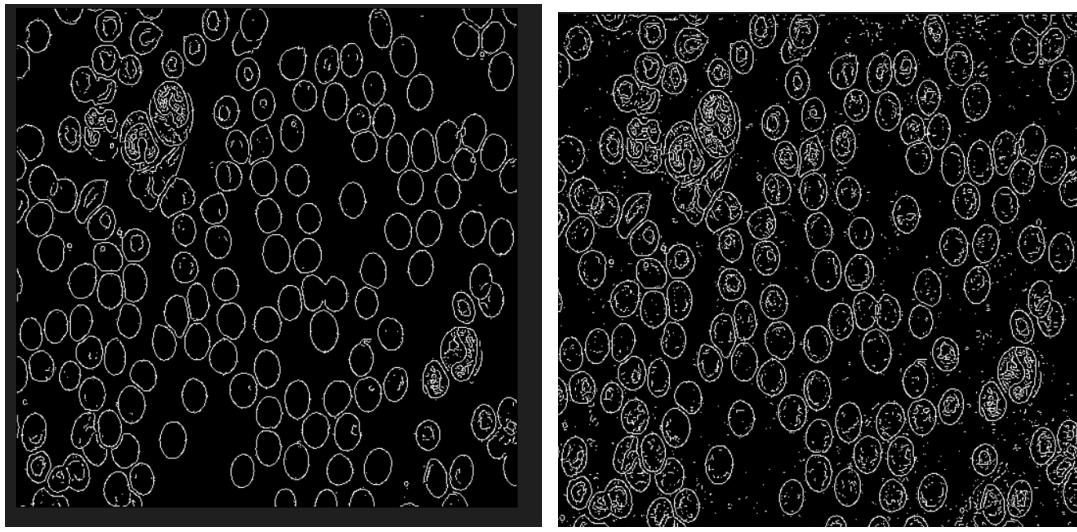


seuilnorme = 0.1      seuilnorme = 0.3

Bien que le bruit diminue à mesure que la valeur seuil augmente, les contours perdent leur continuité car toutes les parties d'un même contour n'ont pas la même norme.

- Cherchez à fixer le seuil sur la norme de `facon` à obtenir un compromis entre robustesse au bruit et continuité des contour

0,1 est une très bonne valeur pour le seuil car nous avons déjà observé que lorsque cette valeur est augmentée, les contours perdent beaucoup de continuité et que si nous la diminuons, nous augmentons le bruit dans l'image.



seuilnorme = 0.1

seuilnorme = 0.05

### 1.3 Filtre récursif de Deriche

Les lignes de code corrigées sont les suivantes

```
for j in range(2,nc):
    b1[j] = l[j-1] + 2*ae*b1[j-1] - (ae**2)*b1[j-2]      # LIGNE
A MODIFIER
    b1[0]=b1[2]
    b1[1]=b1[2]

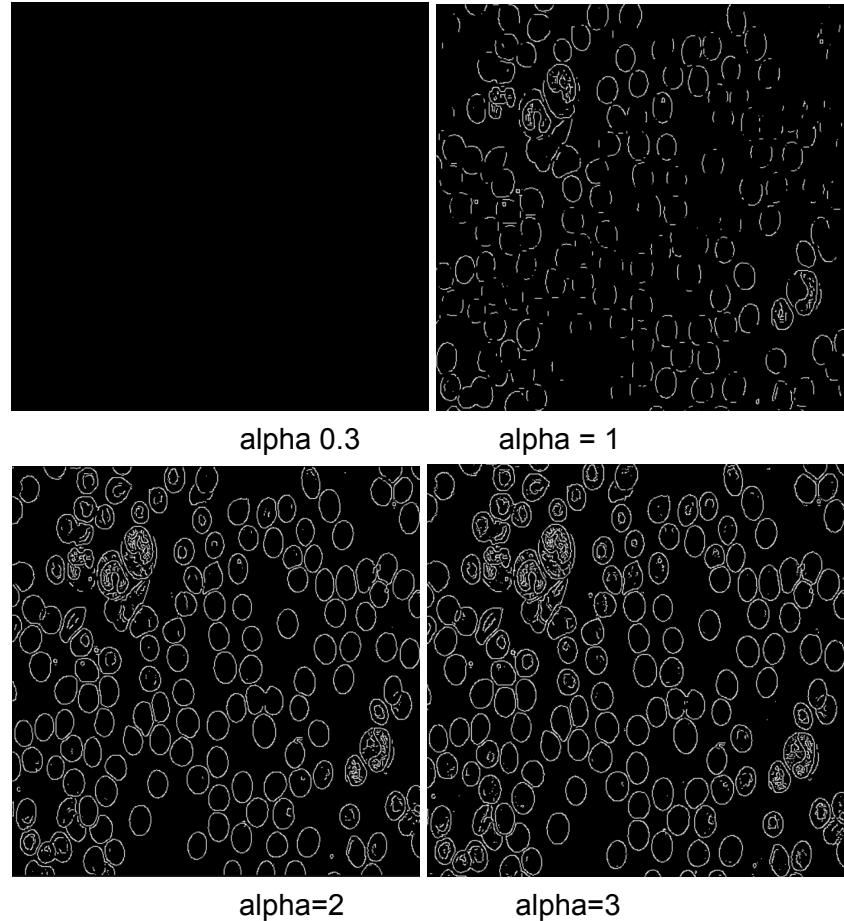
    for j in range(nc-3,-1,-1):
        b2[j] = l[j+1] + 2*ae*b2[j+1]- (ae**2)*b2[j+2]      # LIGNE A
MODIFIER
    b2[nc-2]=b2[nc-3]
    b2[nc-1]=b2[nc-3]
```

représentant cette partie de l'algorithme qui n'étaient pas complètes

$$y_1(n) = x(n-1) + 2e^{-\alpha}y_1(n-1) - e^{-2\alpha}y_1(n-2) \text{ pour } n = 3, \dots, N$$

$$y_2(n) = x(n+1) + 2e^{-\alpha}y_2(n+1) - e^{-2\alpha}y_2(n+2) \text{ pour } n = N-2, \dots, 1$$

- Testez la détection de contours avec ce filtre sur plusieurs images. Décrivez l'effet du paramètre  $\alpha$  sur les résultats de la segmentation (faites varier ce paramètre sur l'intervalle 0, 3...3, 0).



Plus la valeur alpha est élevée, plus la norme globale des pixels est élevée, ce qui permet à un plus grand nombre de pixels de dépasser le seuil et d'apparaître en blanc dans l'image.

- **Le temps de calcul dépend-il de la valeur de  $\alpha$  ? Expliquez pourquoi.**

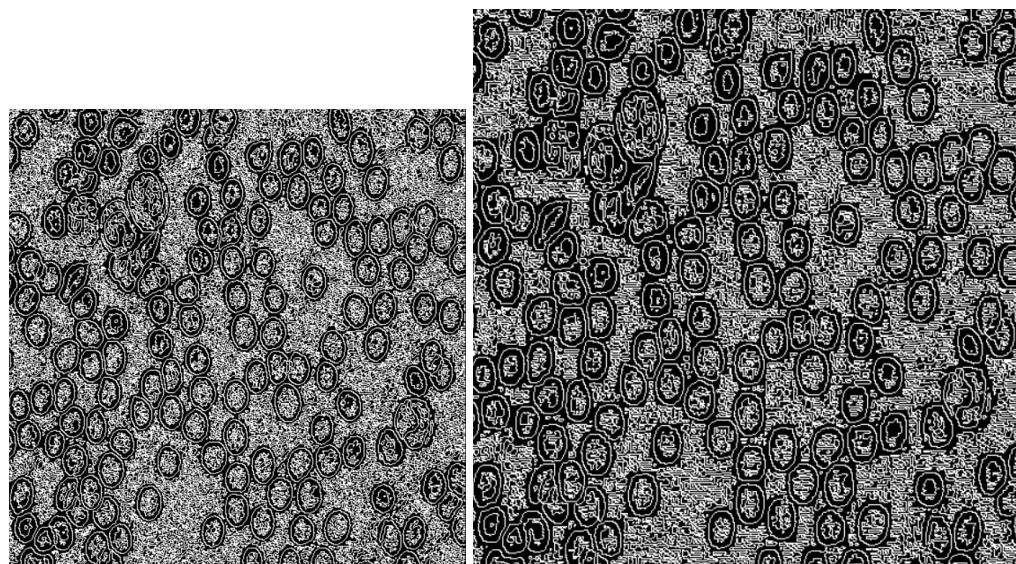
Non, car la valeur de alpha ne modifie pas le nombre d'itérations que l'algorithme doit effectuer, elle ne modifie que N

- **Comment et dans quel but les fonctions `dericheSmoothX` et `dericheSmoothY` sont-elles utilisées (cf. le filtre de Sobel).**

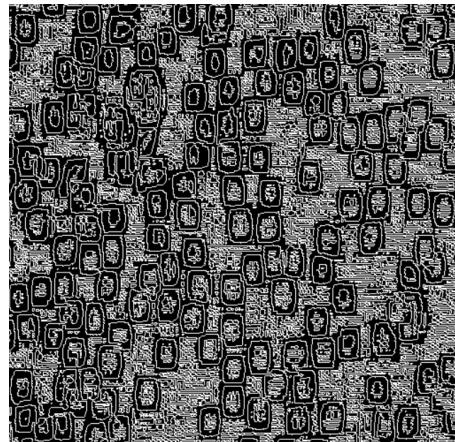
Reproduit l'application d'un filtre passe-bas afin de supprimer le bruit de l'image.

#### 1.4 Passage par zéro du laplacien

- **Quel est l'effet du paramètre  $\alpha$  sur les résultats ?**



alpha = 3    alpha=1



alpha=0.5

Plus la valeur d'alpha est élevée, plus les résultats sont continus. Cependant, après la valeur de 1, le changement n'est plus aussi important, en raison de l'utilisation intensive de la fonction exponentielle.

- Sur l'image cell.tif, quelles sont les principales différences par rapport aux résultats fournis par les opérateurs vus précédemment (contours, Deriche) ?

Que l'opérateur de Laplace nous donne toujours des contours fermés par rapport aux opérateurs vus ci-dessus.

- Sur l'image pyramide.tif, comment est-il possible de supprimer les faux contours créés par cette approche ?

Cristian Alejandro Chávez Becerra

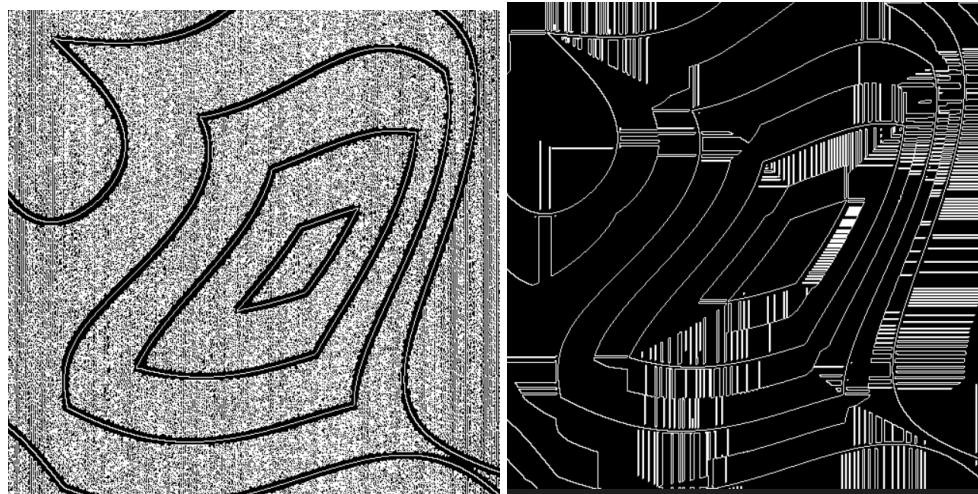
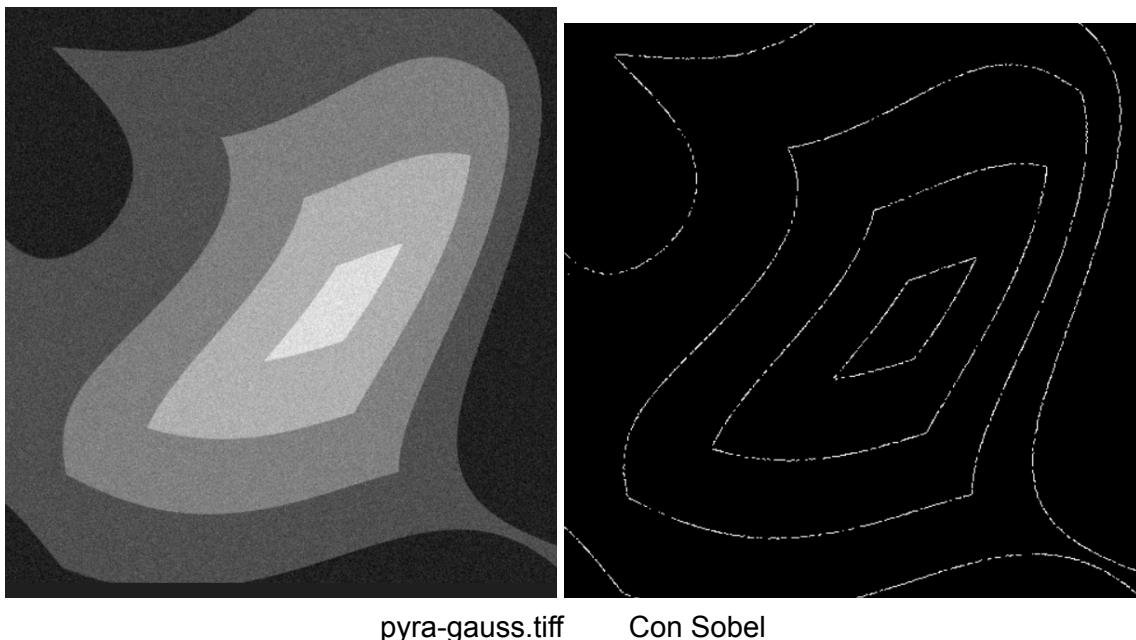


Image avec raisins secs      Image sans raisins secs

L'ajout d'un filtre de moyennage permet d'obtenir un meilleur résultat, car le laplacien est très sensible au bruit, et ce qui aggrave la situation, c'est qu'il cherche à envoyer un contour fermé même dans les cas où il détecte du bruit.

### 1.5 Changez d'image

- Quel opérateur choisiriez-vous pour segmenter l'image pyra-gauss.tif ?



Comme les contours ont des valeurs très similaires malgré le bruit gaussien, la tâche est très compliquée. J'utiliserais donc Sobel, mais aussi Deriche, car il ne serait pas difficile de trouver les contours.

- Quels seraient les pré-traitements et les post-traitements à effectuer ?

Dans Sobel, j'augmenterais la valeur du seuil à 0,9 même si le bruit n'est pas si important par rapport à la force des contrastes des contours, et dans Deriche, j'appliquerais un filtre de moyenne afin de réduire le bruit, ce que Sobel fait par nature.

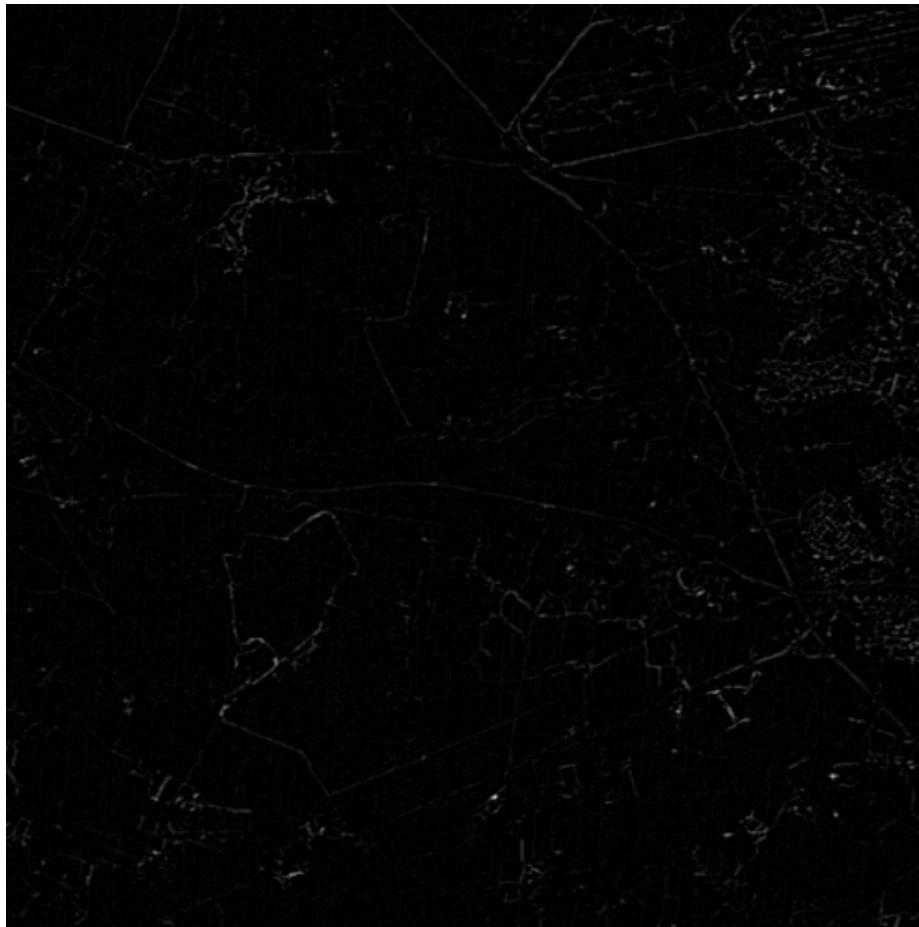
## 2 Seuillage avec hyst'er'esis

### 2.1 Application `a la d'etection de lignes

- Appliquez le filtre du Chapeau haut de forme (tophat) `a une image SPOT pour effectuer une détection de lignes :

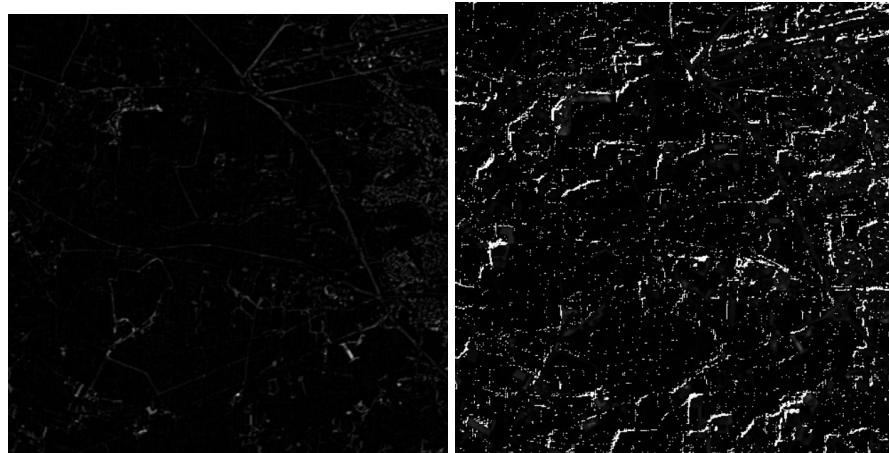


original image



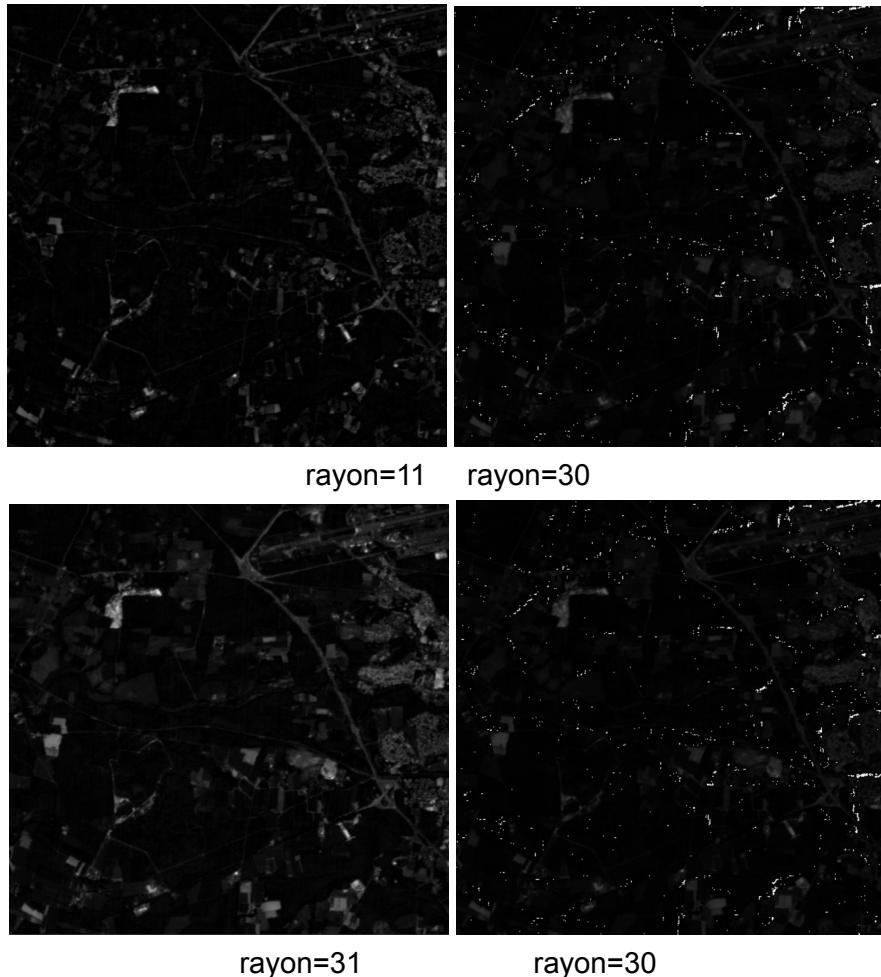
Tophat filter

- Modifiez le rayon de l'élément structurant utilisé pour calculer le filtre tophat, et indiquez comment évoluent les lignes détectées.



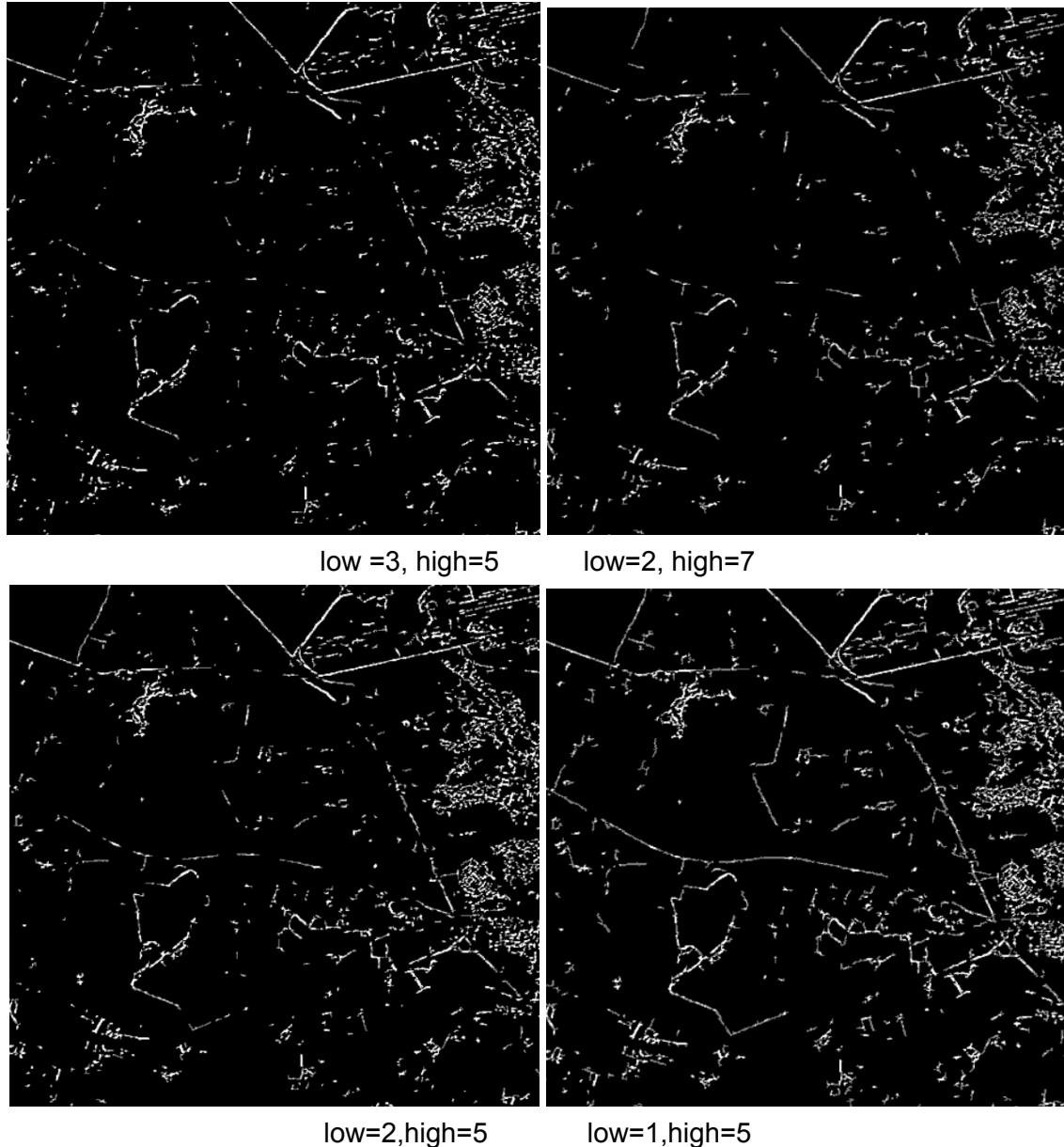
rayon 5      rayon = 10

Cristian Alejandro Chávez Becerra



On obtient de meilleurs résultats avec des valeurs impaires qu'avec des valeurs paires, et plus la valeur du rayon est grande, plus les contrastes sont marqués.

- **Modifiez les valeurs des deux seuils, et examinez comment les lignes sont supprimées ou préservées. Quels sont les seuils qui donnent, à votre avis, le meilleur résultat ?**



Si le seuil bas est maintenu constant, plus la valeur du seuil le plus grand est élevée, moins nous pourrons observer de contours car nous obtiendrons moins de pixels forts (ceux qui dépassent le seuil le plus élevé) auxquels sont connectés des pixels faibles (ceux qui dépassent le seuil le plus bas mais pas le seuil le plus élevé).

En revanche, si le seuil haut est maintenu constant, plus la valeur du seuil le plus bas est faible, plus nous pourrons observer de contours car nous obtiendrons le même nombre de pixels forts mais plus de pixels faibles qui peuvent être connectés aux pixels forts.

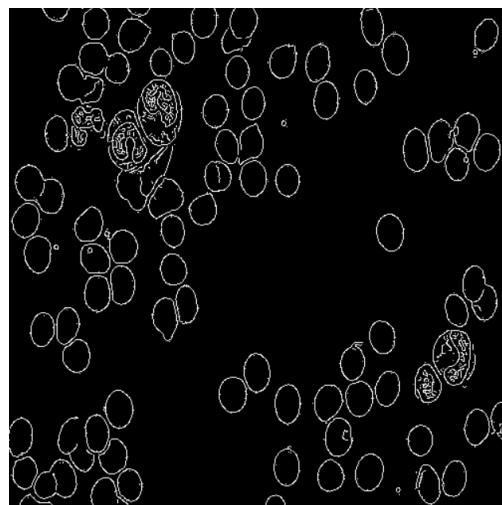
Les valeurs qui m'intéressent le plus sont celles où  $\text{low}=1, \text{high}=5$

Cristian Alejandro Chávez Becerra

- Appliquez le seuillage par hystérésis pour améliorer la détection de contours obtenue avec un des opérateurs vus précédemment sur une image de votre choix. Précisez la mise en oeuvre que vous proposez et commentez les résultats.



Sobel Seuilnorme = 0.3



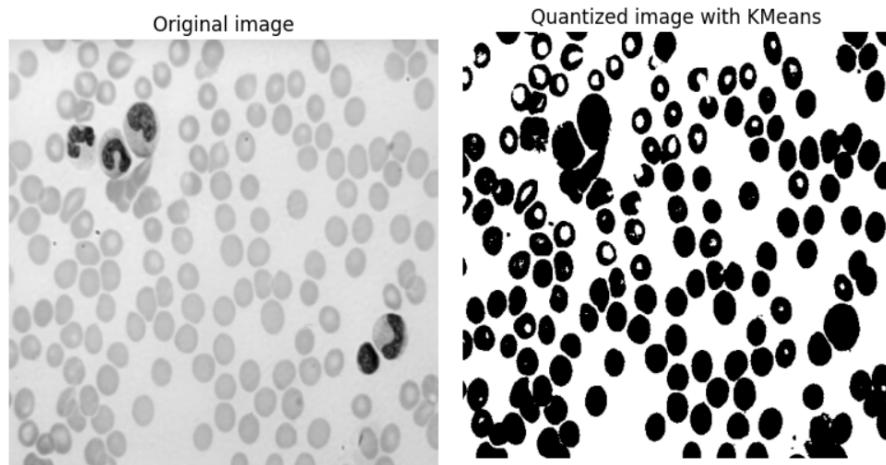
Sobel sobelNormeLow=0.1, sobelNormeHigh=0.3

Le seuil par hystérésis a été utilisé avec le filtre de Sobel et une amélioration de la continuité peut être observée si le seuil utilisé à l'origine est élevé.

### 3 Segmentation par classification : K-moyennes

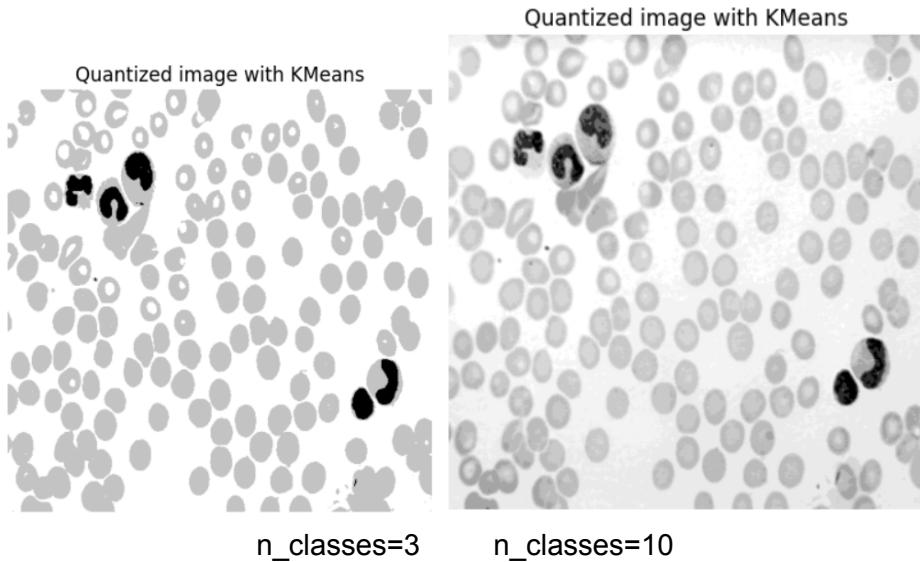
#### 3.1 Image à niveaux de gris

- Testez l'algorithme des k-moyennes sur l'image cell.tif pour une classification en 2 classes. Cette classification segmente-t-elle correctement les différents types de cellules ? Si non, que proposez-vous ?



Il ne s'agit pas d'une bonne classification car nous avons en fait 3 types différents de nuances fortes, puisqu'il y en a 2 pour les 2 cellules différentes plus une pour le fond, et comme la classification dans ce cas a été faite pour 2 couleurs, nous perdons de l'information.

- **Testez les différentes possibilités pour initialiser les classes. D'écrivez si possible ces différentes méthodes.**



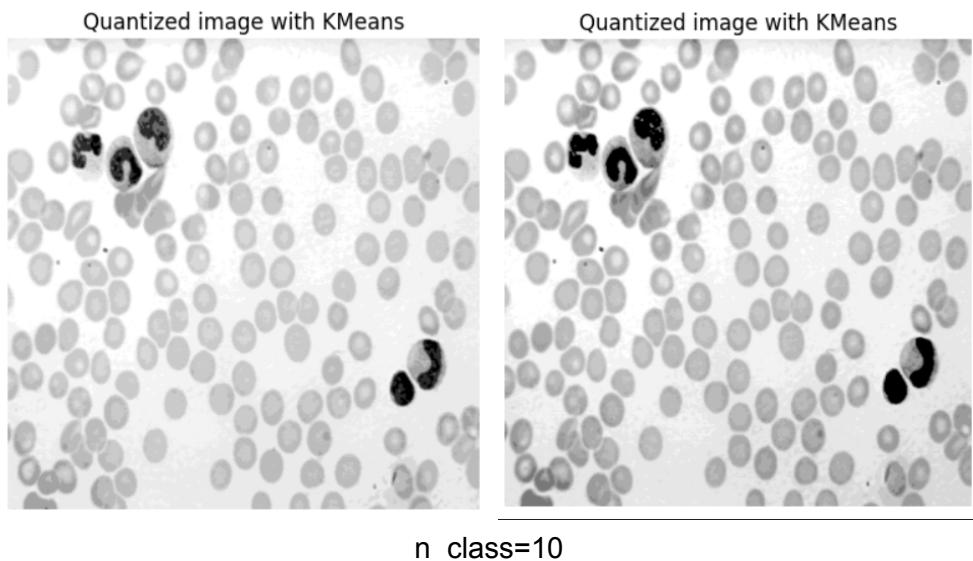
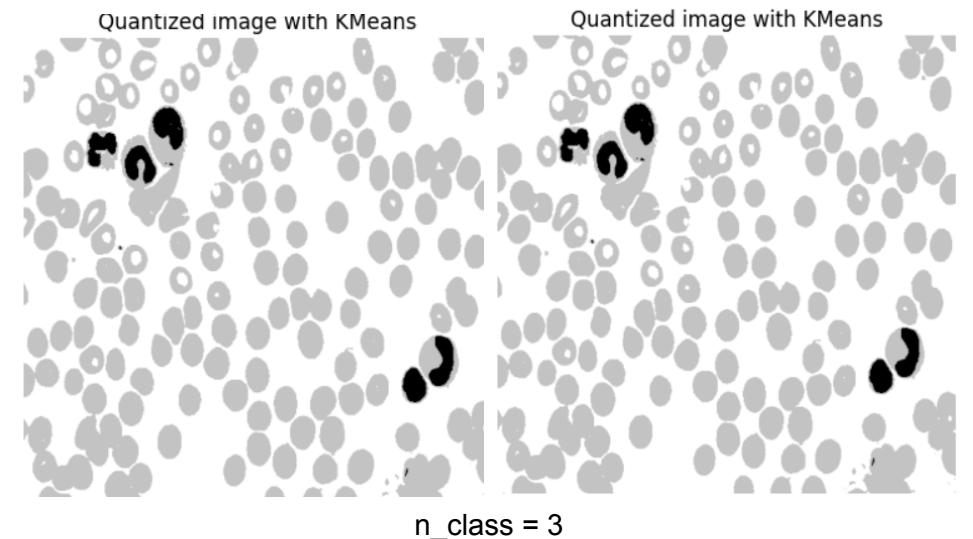
La classification est meilleure lorsque nous obtenons 3 classes pour les raisons mentionnées ci-dessus.

`random_state` : détermine la génération de nombres aléatoires pour l'initialisation du centroïde. Un int est utilisé pour que le hasard soit déterministe, c'est-à-dire que les mêmes résultats peuvent être reproduits.

`max_iter` : permet de définir le nombre d'itérations. En général, plus cette valeur est élevée, meilleurs sont les résultats.

`n_init` : C'est le nombre de fois que l'algorithme K-means sera exécuté, donnant comme résultat celui qui a obtenu la meilleure performance.

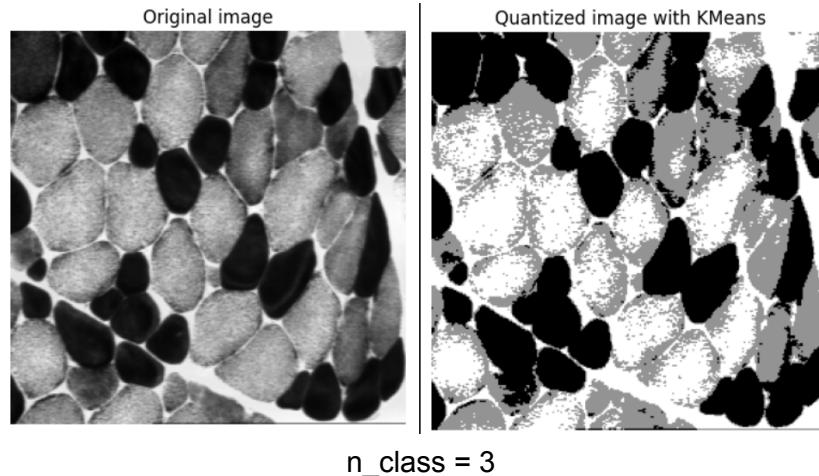
- La classification obtenue est-elle stable (même position finale des centres des classes) avec une initialisation aléatoire ? Testez sur différentes images à niveaux de gris et différents nombres de classes.



En raison de la nature de l'algorithme K-means, des centroïdes initiaux différents peuvent donner des résultats différents, mais dans ce cas, étant donné qu'il y a des couleurs principales, les différences ne sont pas si grandes. Néanmoins, on peut constater que plus le nombre de classes est élevé, plus la différence peut être importante.

Cristian Alejandro Chávez Becerra

- Quelles sont les difficultés rencontrées pour la segmentation des différentes fibres musculaires dans l'image muscle.tif ?



Même si l'on dispose de trois teintes dominantes et que l'on choisit trois classes, certaines parties des tissus ont des couleurs très différentes de celles qui les entourent et sont donc classées avec un centroïde différent de celui des tissus qui les entourent.

- Expliquez pourquoi le filtrage de l'image originale (filtre de la moyenne ou filtre median) permet d'améliorer la classification.

Cela revient à homogénéiser les couleurs d'un même tissu, ce qui permet de classer les pixels appartenant au même tissu dans une couleur plus proche.

### 3.2 Image en couleur

- Testez l'algorithme sur l'image fleur.tif pour une classification en 10 classes, les centres des classes initiaux étant tirés aléatoirement).



- Commentez la dégradation de l'image quantifiée par rapport à l'image initiale.

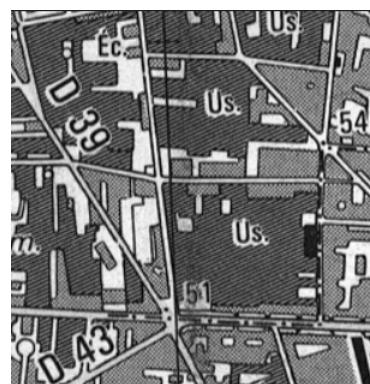
L'image originale contient beaucoup plus de nuances de couleurs que 10, de sorte que vous perdez des informations sur d'autres nuances, ce qui dégrade l'image. En utilisant le modèle RVB, on peut obtenir environ  $256 \times 256 \times 256 = 16\,777\,216$ . Quoi qu'il en soit, malgré la dégradation, il s'agit d'un bon résultat compte tenu de la quantité d'informations perdues.

- **Quel est le nombre minimum de classes qui donne un rendu visuel similaire à celui de l'image codée sur 3 octets ?**

Cela dépend du nombre de couleurs différentes utilisées par l'image, car toutes les images n'utilisent pas la gamme complète de couleurs codées sur 8 octets, mais, en général, je dirais 256, car dans ce cas nous aurions une réduction de 66,66 %, mais cela resterait similaire, comme vous pouvez le voir, mais cela a un coût de calcul.



- **Proposez une solution pour retrouver les planches-mères utilisées pour l'impression d'une carte IGN : carte.tif.**



carte.tif

En utilisant l'algorithme k-means, nous pourrions utiliser 4 classes pour les 4 couleurs différentes de l'image carte.tif.

## 4 Seuillage automatique : Otsu

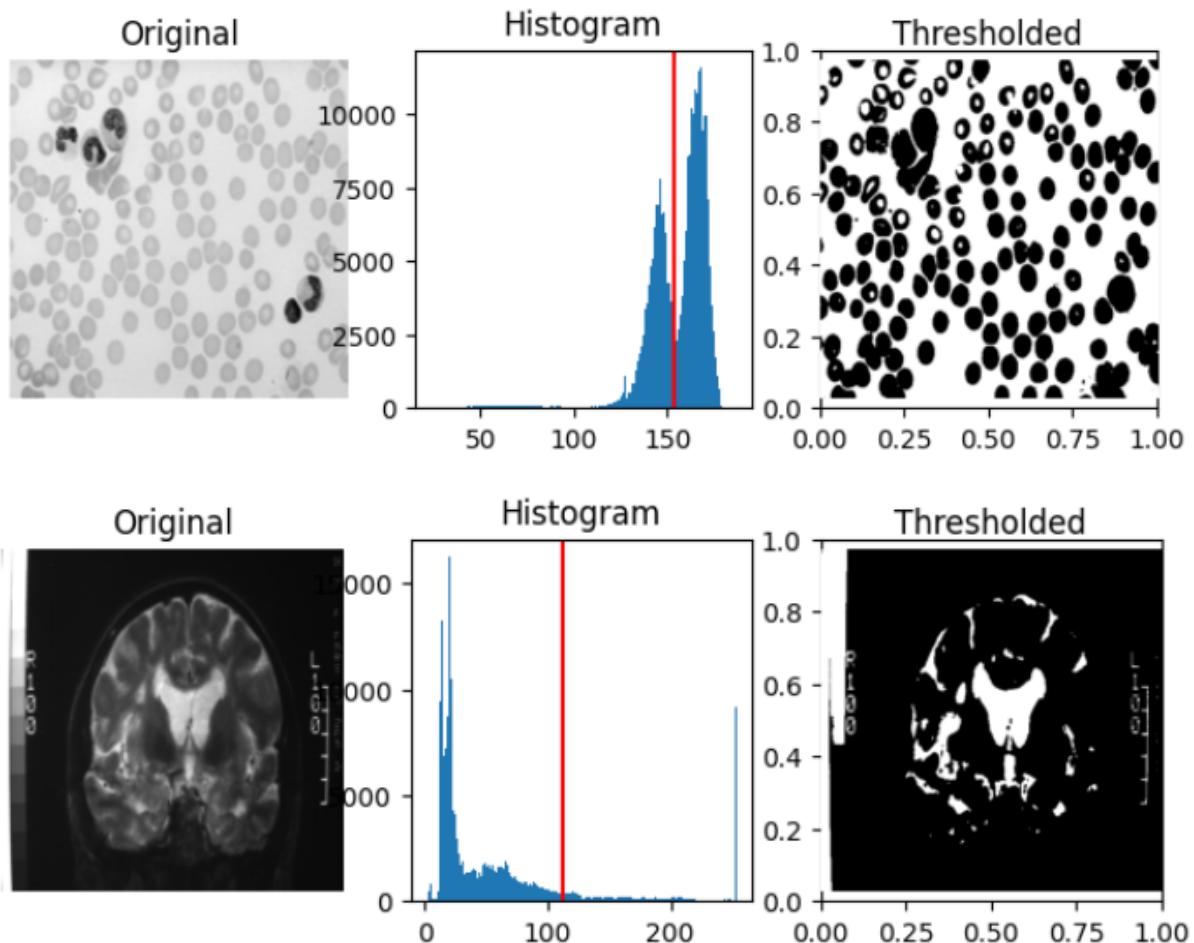
- Dans le script otsu.py quel critère cherche-t-on à optimiser ?

Nous cherchons à maximiser la variance intra-classe, dont la formule est la suivante

$$\text{var}(T) = P_0(T) \cdot P_1(T) \cdot (m_0(T) - m_1(T))^2$$

Ici,  $P_0(T)$  et  $P_1(T)$  représentent les probabilités des régions d'arrière-plan et d'avant-plan, respectivement. Nous définissons également  $m_0(T)$  et  $m_1(T)$  comme les valeurs moyennes d'intensité en niveaux de gris des régions d'arrière-plan et de premier plan, respectivement.

- Testez la méthode de Otsu sur différentes images à niveaux de gris, et commentez les résultats.



La méthode est efficace dans ce cas lorsque les régions ont des variances similaires et qu'il n'y a que deux pics dans l'histogramme de l'image.

- Cette méthode permet-elle de seuiller correctement une image de norme du gradient ?

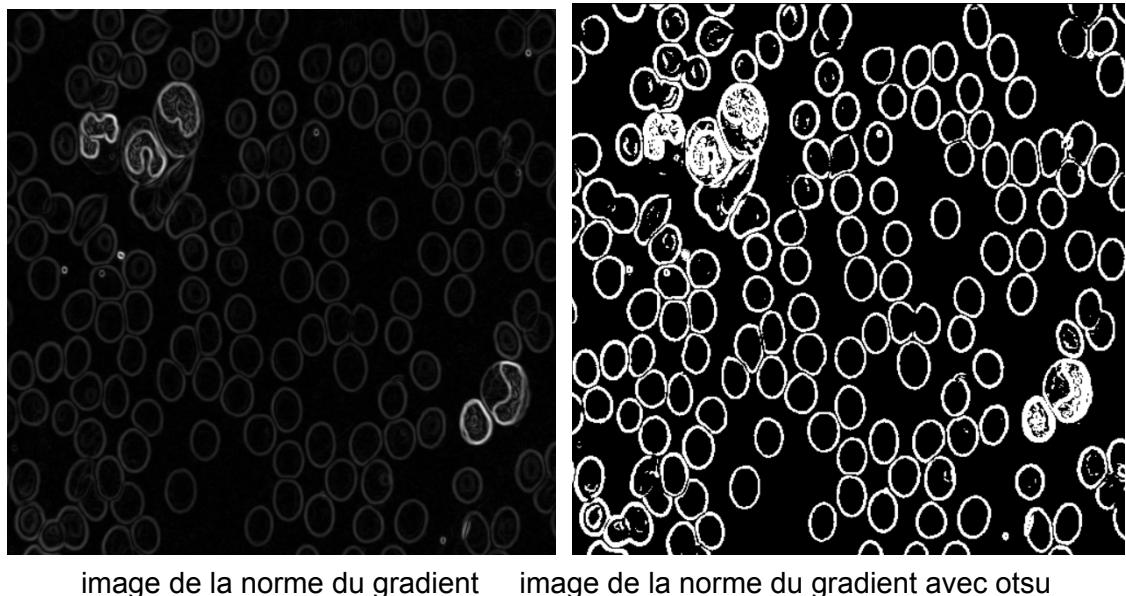


image de la norme du gradient

image de la norme du gradient avec otsu

En général, oui, une image à norme de gradient permet d'apprécier davantage la différence de contour/contraste (filtre passe-haut) des parties continues ou lisses (filtres passe-bas), ce qui permet d'avoir seulement une région de premier plan et une région d'arrière-plan.

Cependant, ce n'est pas une bonne solution si les régions de l'image originale doivent être segmentées en plus de deux classes.

- Modifiez le script `otsu.py` pour traiter le problème à trois classes, i.e. la recherche de deux seuils.

```
def otsu_thresh2(im):

    h=histogram(im)
    m=0
    for i in range(256):
        m=m+i*h[i]

    mt1, mt2, maxk =0, 0, 0

    for t1 in range(256):
        for t2 in range(256):
            j0,j1,j2,a0,a1,a2=0,0,0,0,0,0
```

Cristian Alejandro Chávez Becerra

```

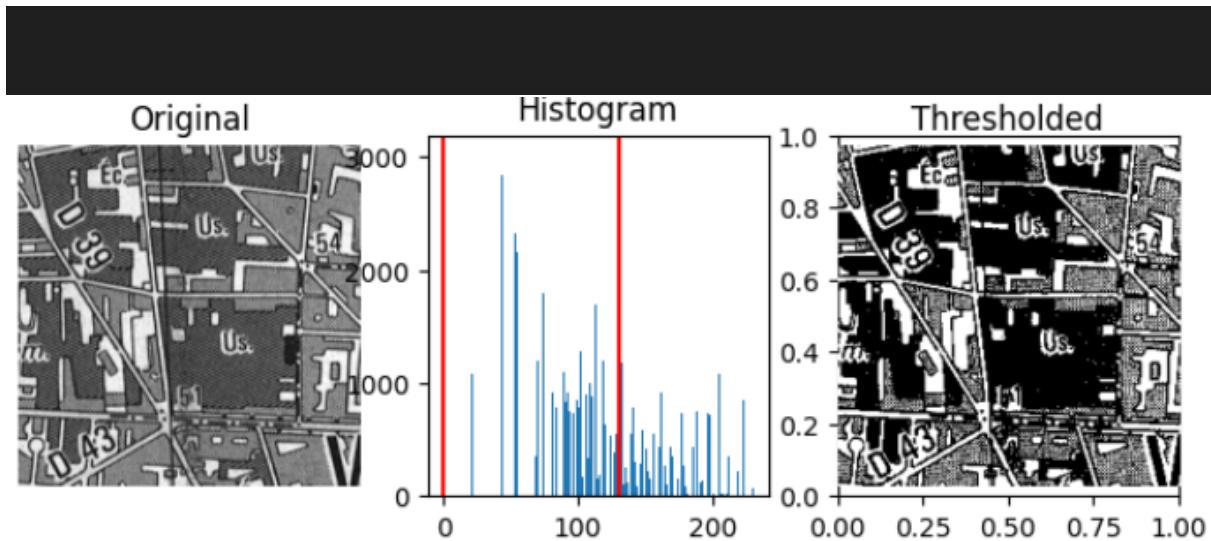
      #calculate mean grayscale intensity values for values below
#the 1st threshold
      for i in range(t1):
          j0=j0+h[i]
          a0=a0+i*h[i]
      if j0 > 0:
          a0=a0/j0

# calculate mean grayscale intensity values for values between the 2
thresholds
      for i in range(t1,t2):
          j1=j1+h[i]
          a1=a1+i*h[i]
      if w1 > 0:
          a1=a1/j1
      #calculate mean grayscale intensity values for values above
#the 2nd threshold
      for i in range(t2,256):
          j2=j2+h[i]
          a2=a2+i*h[i]
      if j2 > 0:
          a2=a2/j2
      #now, we calculate the between-class variance of 3 classes
      possible=j0*j1*(a0-a1)*(a0-a1) + j1*j2*(a1-a2)*(a1-a2)
      if possible > maxk:
          maxk=possible
          mt1=t1
          mt2=t2

      new_thresh=(mt1, mt2)
      return(new_thresh)

image = skio.imread('cell.tif')
thresh=otsu_thresh2(image)
#The pixels between the 2 thresholds will have the 126 value in the
#gray scale
binary = (image > thresh[0])*126
binary2 = (image > thresh[1])*255
binary = binary + binary2

```



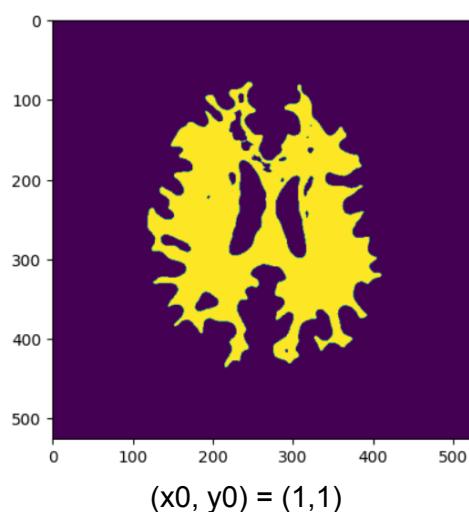
## 5 Croissance de régions

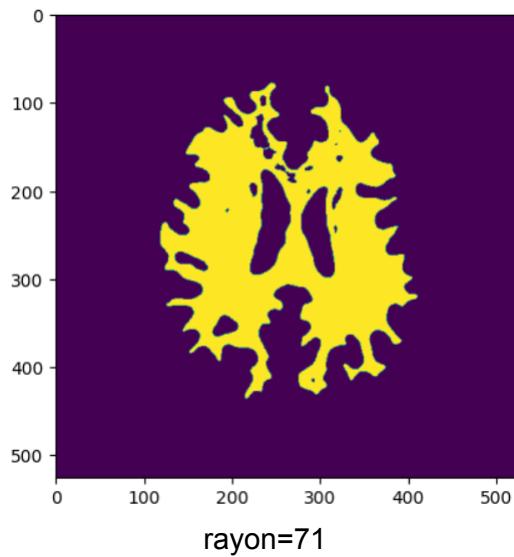
- Quelles contraintes doit vérifier un pixel pour être ajouté à l'objet existant ?

Il doit s'agir des pixels résultant de la matrice diff.

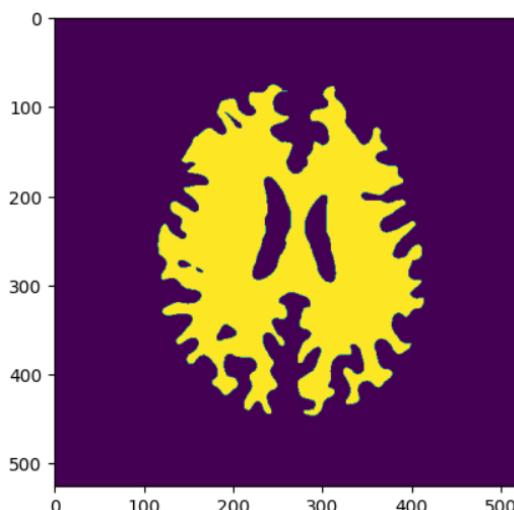
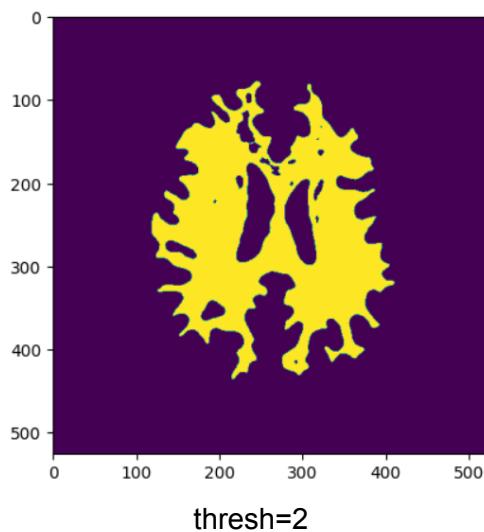
La différence entre la moyenne des pixels adjacents et la moyenne de la graine initiale avec ses voisins doit être inférieure au seuil multiplié par l'écart-type de la graine initiale avec ses voisins.

- Les param`etres `a fixer sont la position du point de départ ( $x_0, y_0$ ), un seuil thresh et le rayon qui définit le voisinage sur lequel sont estim`es la moyenne et l'écart-type locaux.





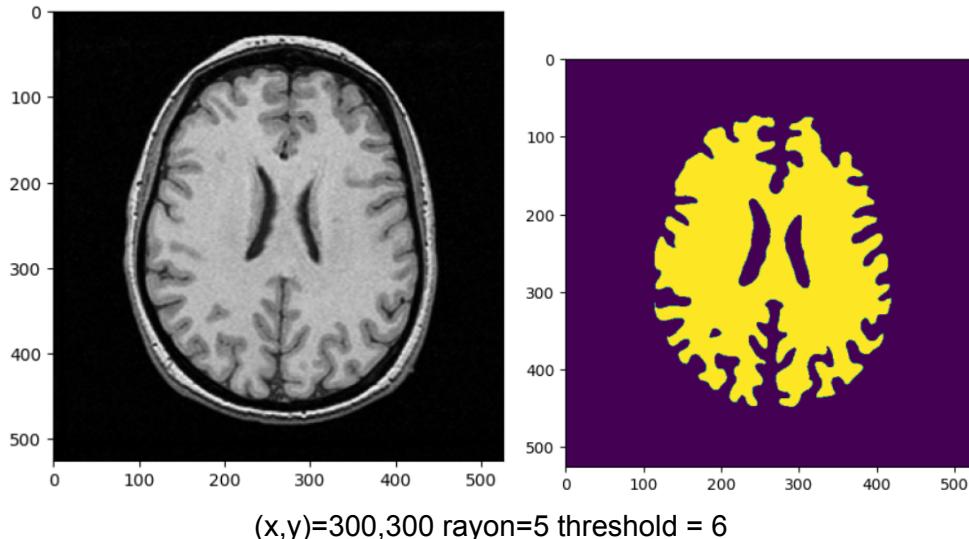
- Quel est l'effet du param`etre thresh sur le r`esultat de segmentation ?



thresh=4

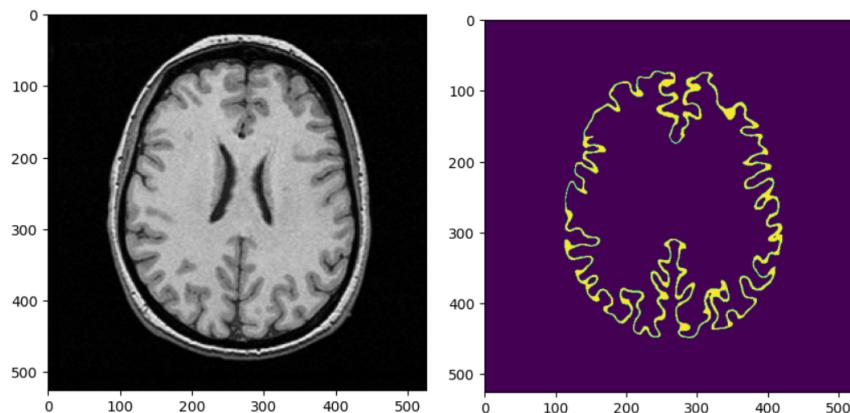
Plus la région est grande, plus le nombre de pixels autorisés à y pénétrer est élevé.

- Quels paramètres permettent de segmenter correctement la matière blanche ?



- Parvenez-vous à segmenter la matière grise également ?

Oui, cela dépend de la valeur de la graine. En choisissant une valeur de graine qui correspond à une valeur similaire aux valeurs des pixels représentant la matière grise, c'est possible.



- Quel est le prédicat mis en place dans ce script ?

Si la différence entre la moyenne des pixels adjacents du pixel courant et la moyenne des pixels adjacents du pixel de départ est inférieure à la moyenne, cela signifie qu'ils appartiennent au même type.

Cristian Alejandro Chávez Becerra

- Proposez un autre algorithme qui n'utilise pas la croissance de régions, mais qui donne le même résultat.

On ne peut évaluer que la valeur de la différence des pixels par rapport au pixel de départ et que la différence est inférieure à la multiplication du seuil par l'écart-type et que les pixels sont connectés (qu'il y a un chemin entre eux à partir de pixels appartenant déjà à la région) à un autre pixel appartenant à la région.

- Proposez un prédicat qui nécessite réellement un algorithme de croissance de région.

Les pixels doivent être connectés (il doit y avoir un chemin entre eux à partir de pixels appartenant déjà à la région) à un autre pixel appartenant à la région.