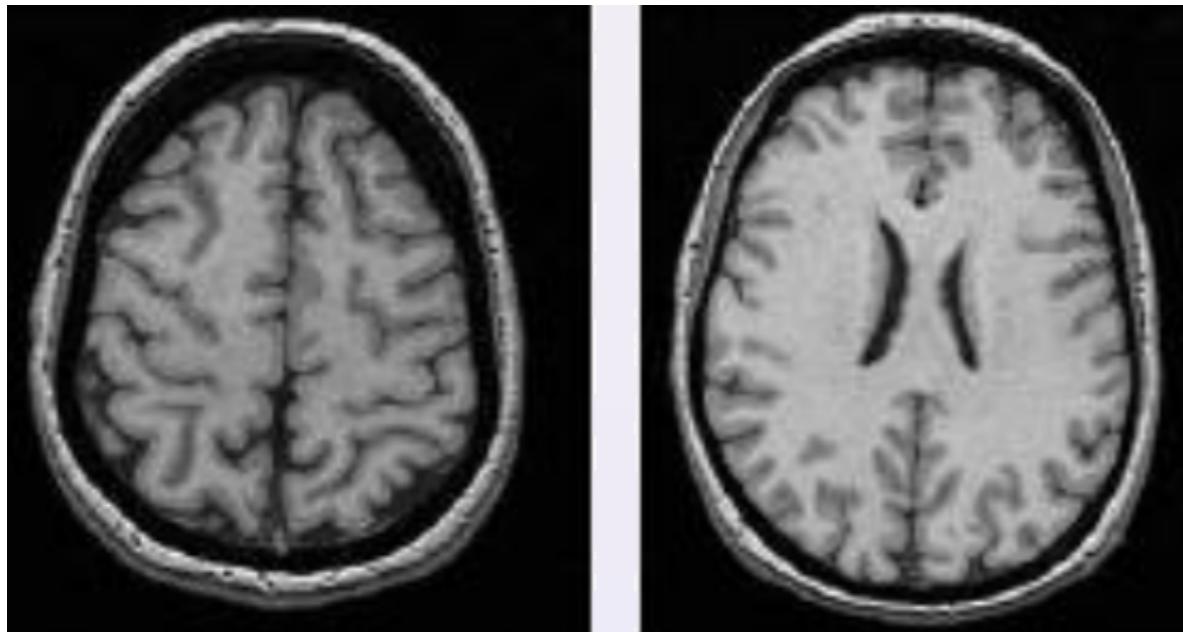


# IMN-530

Reconstruction et analyses d'images médicales  
-  
SEGMENTATION

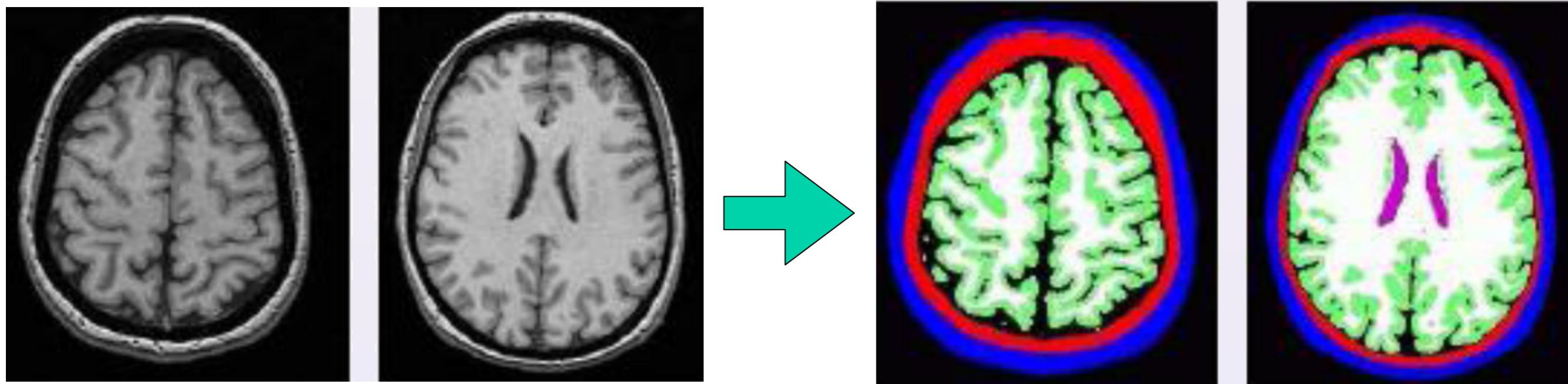
# Segmentation

- La segmentation vise à découper une image en régions connexes présentant une homogénéité selon un certain critère
- Différentes possibilités



# Segmentation

- La segmentation vise à découper une image en régions connexes présentant une homogénéité selon un certain critère
- Différentes possibilités



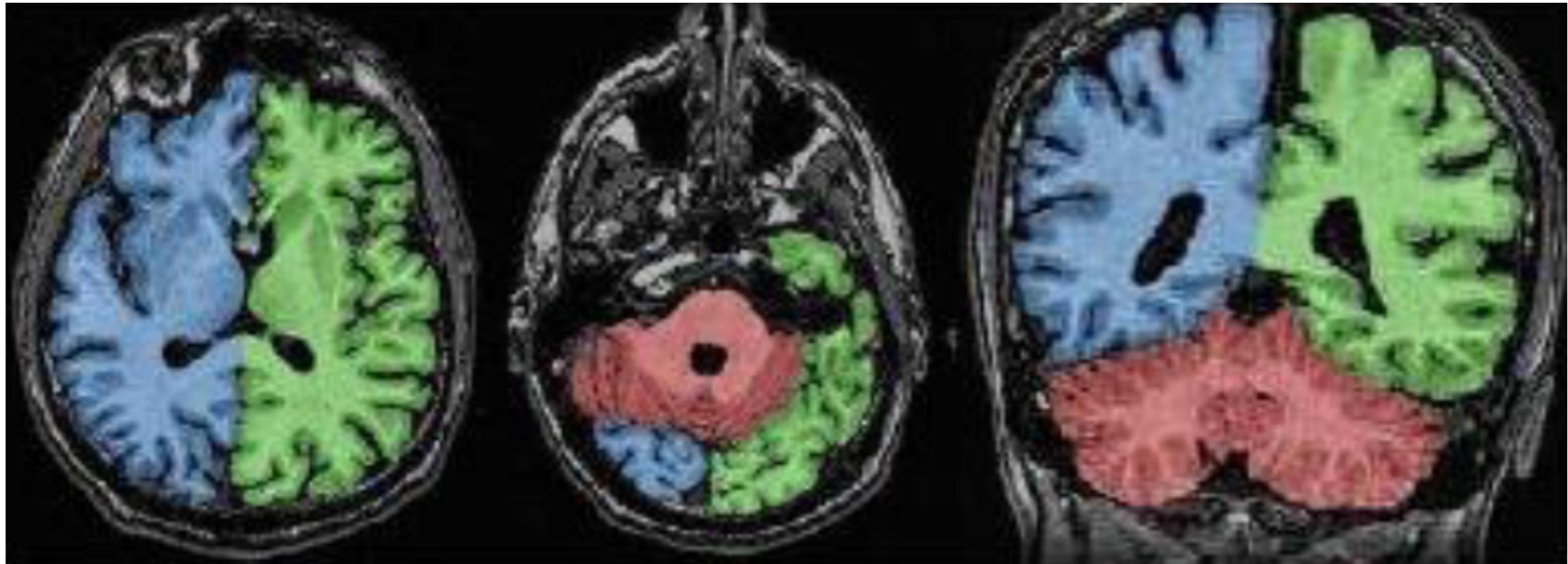
# Segmentation

- La segmentation vise à découper une image en régions connexes présentant une homogénéité selon un certain critère
- Différentes possibilités



Peau, os, LCR, matière grise, matière blanche, ventricules

# Segmentation



Hémisphère gauche,  
hémisphère droit, cervelet

# Segmentation

- Opération cruciale dans de nombreuses applications médicales

# Segmentation

- Opération cruciale dans de nombreuses applications médicales
- Imagerie fonctionnelle : quantification des volumes des tissus, des organes

# Segmentation

- Opération cruciale dans de nombreuses applications médicales
- Imagerie fonctionnelle : quantification des volumes des tissus, des organes
- Localisation d'une pathologie

# Segmentation

- Opération cruciale dans de nombreuses applications médicales
- Imagerie fonctionnelle : quantification des volumes des tissus, des organes
- Localisation d'une pathologie
- Étude d'une structure anatomique

# Segmentation

- Opération cruciale dans de nombreuses applications médicales
- Imagerie fonctionnelle : quantification des volumes des tissus, des organes
- Localisation d'une pathologie
- Étude d'une structure anatomique
- Planification d'un traitement

# Segmentation

- Opération cruciale dans de nombreuses applications médicales
- Imagerie fonctionnelle : quantification des volumes des tissus, des organes
- Localisation d'une pathologie
- Étude d'une structure anatomique
- Planification d'un traitement
- Chirurgie assistée par ordinateur

# Segmentation

- Opération cruciale dans de nombreuses applications médicales
- Imagerie fonctionnelle : quantification des volumes des tissus, des organes
- Localisation d'une pathologie
- Étude d'une structure anatomique
- Planification d'un traitement
- Chirurgie assistée par ordinateur
- Toutes les modalités de l'imagerie médicale sont concernées !

# Segmentation

- Description “haut-niveau” d’une image

# Segmentation

- Description “haut-niveau” d’une image
- Peut-être basée sur:
  - Les contours (discontinuités dans les image)
  - Les similitudes entre les régions (couleur, intensité, textures)

# Segmentation

- Description “haut-niveau” d’une image
- Peut-être basée sur:
  - Les contours (discontinuités dans les image)
  - Les similitudes entre les régions (couleur, intensité, textures)
- Pas de solution universelle

# Segmentation

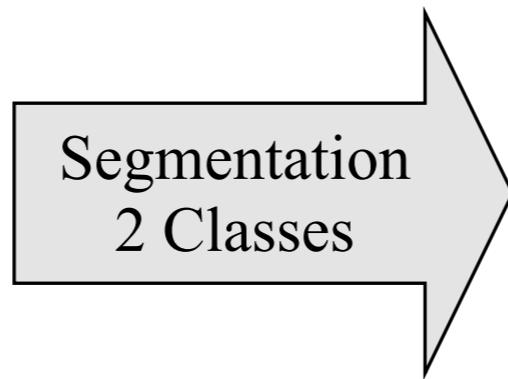
- Description “haut-niveau” d’une image
- Peut-être basée sur:
  - Les contours (discontinuités dans les images)
  - Les similitudes entre les régions (couleur, intensité, textures)
- Pas de solution universelle
- Différentes approches :
  - approches globales (seuillage, k-means, clustering)
  - approches par régions (magic wand, flood fill, markoviennes, ...)
  - approches par contours (contour actif, ...)

# Définition du problème

Concept de classe et d'étiquettes



y



x

Partant d'une image d'entrée « y », on cherche à estimer une image de sortie « x » dont chaque pixel contient une **étiquette de classe** (étiquette *terre* ou étiquette *mer*).

# Définition du problème

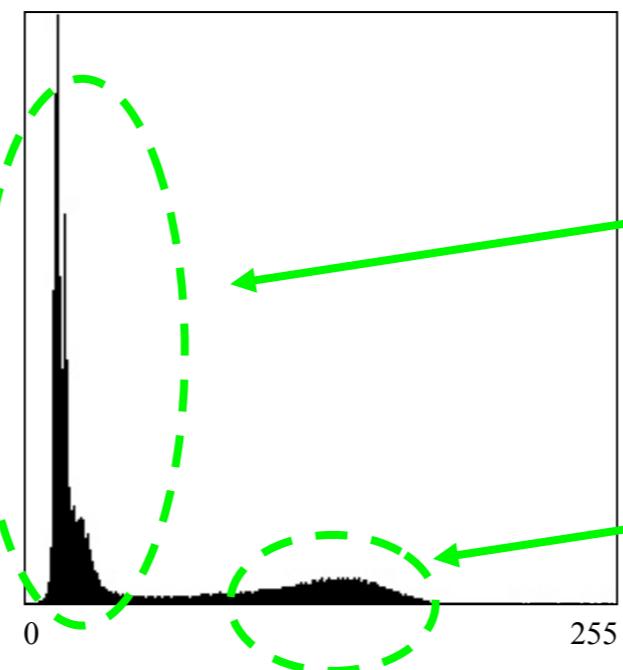
Segmentation, exemple: Terre Vs Mer



Image binaire : blanc = terre, noir = mer



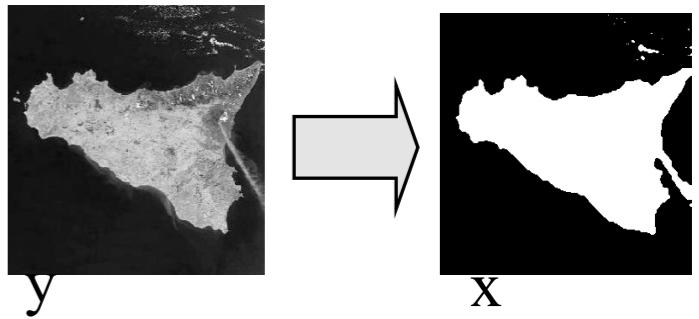
Histogramme normalisé  
des niveaux de gris



Niveaux de gris foncés = Mer

Niveaux de gris clairs = Terre

# Quelques définitions

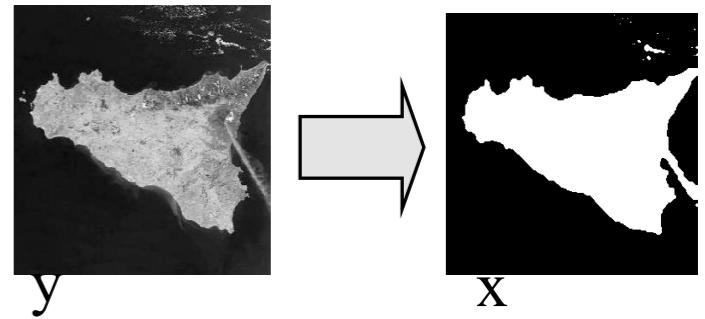


$y$  : un champ d'observations (image d'entrée)

$x$  : un champ d'étiquettes (image à estimer)

$i$

# Quelques définitions



$y$  : un champ d'observations (image d'entrée)

$x$  : un champ d'étiquettes (image à estimer)

**Site** : synonyme de pixel/voxel. Un site « s » possède les coordonnées  $(i,j)$  dans l'image.

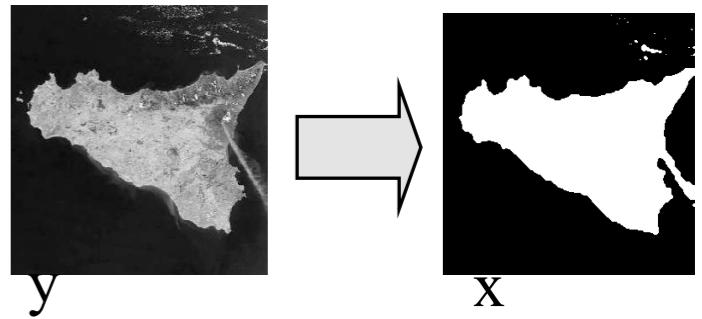
$$y_s = y(i, j) \quad \text{et} \quad y_s \in [0, 255]$$

$$x_s = x(i, j) \quad \text{et} \quad x_s \in \{\text{terre}, \text{mer}\}$$

$S$  : ensemble de tous les sites

$i$

# Quelques définitions



$y$  : un champ d'observations (image d'entrée)

$x$  : un champ d'étiquettes (image à estimer)

**Site** : synonyme de pixel/voxel. Un site «  $s$  » possède les coordonnées  $(i, j)$  dans l'image.

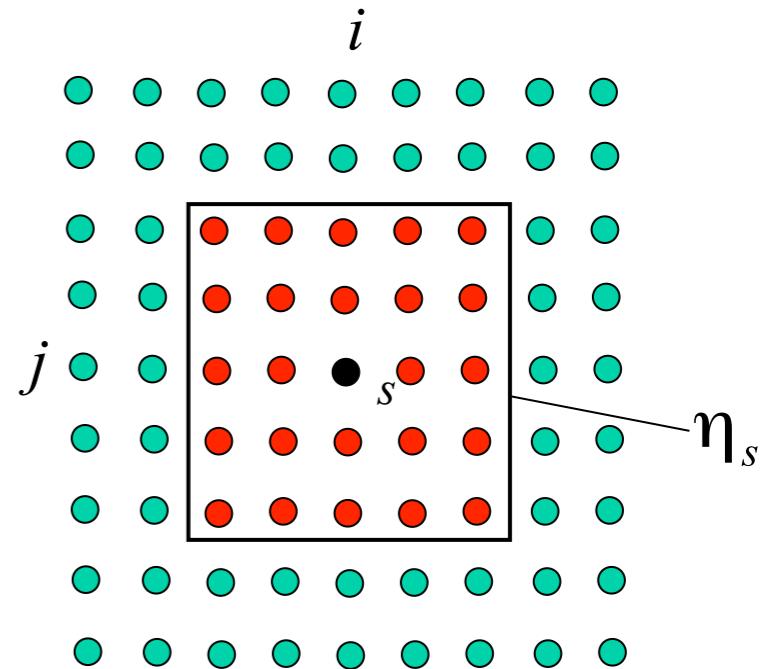
$$y_s = y(i, j) \quad \text{et} \quad y_s \in [0, 255]$$

$$x_s = x(i, j) \quad \text{et} \quad x_s \in \{\text{terre, mer}\}$$

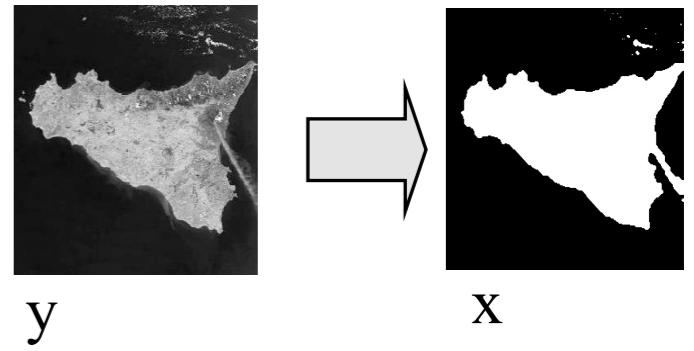
**$S$**  : ensemble de tous les sites

**Voisinage** "η <sub>$s$</sub> " : ensemble des pixels voisins de «  $s$  ».

η <sub>$s$</sub>  est généralement une fenêtre carrée  $N \times N$

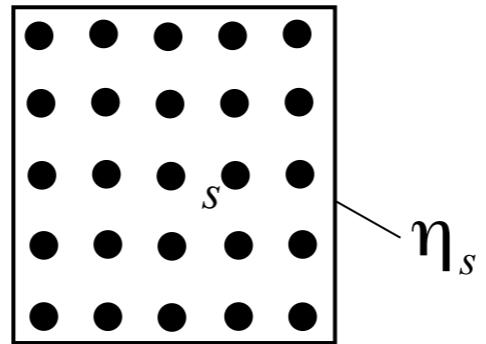


# Quelques définitions



**Cliques** : une clique «  $c$  » est un sous-ensemble de pixels/voxels tel que

- 1- «  $c$  » est un singleton ou
- 2- «  $c$  » est un ensemble de pixels/voxels voisins.



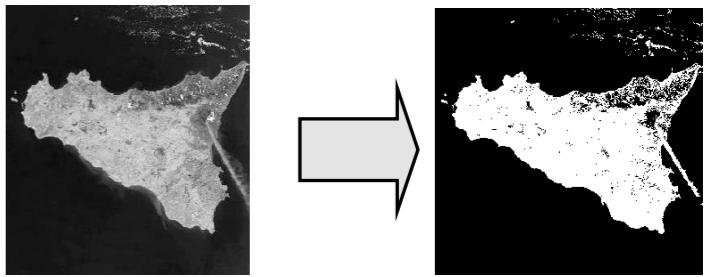
cliques: • —• —• —• ...

# Segmentation

## Approches globales

Algorithme par seuil

# L'algorithme du seuil



Un algorithme simple : le seuil

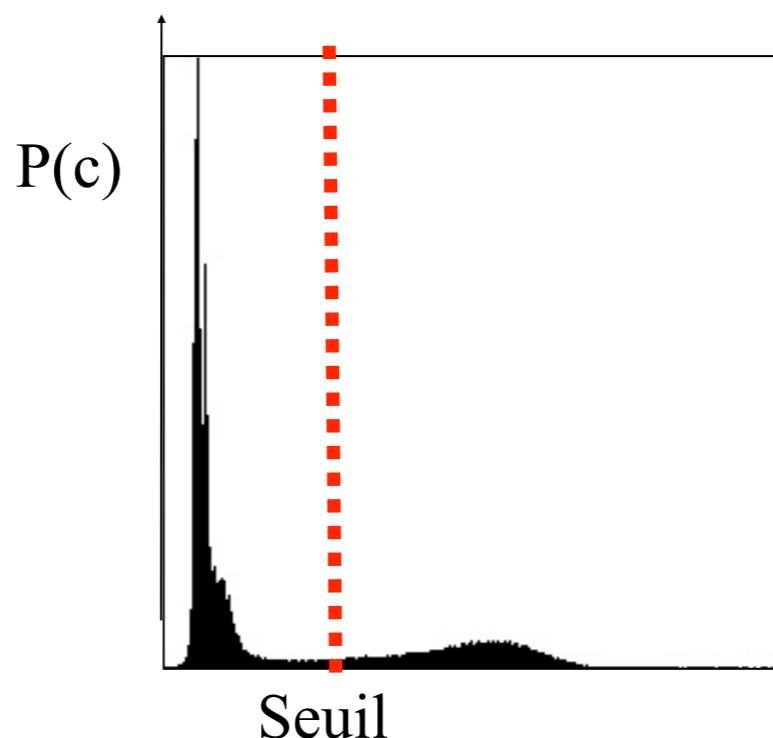
POUR CHAQUE site  $s$  du champ d'observations  $y$  FAIRE

SI  $y_s > \text{Seuil}$  ALORS

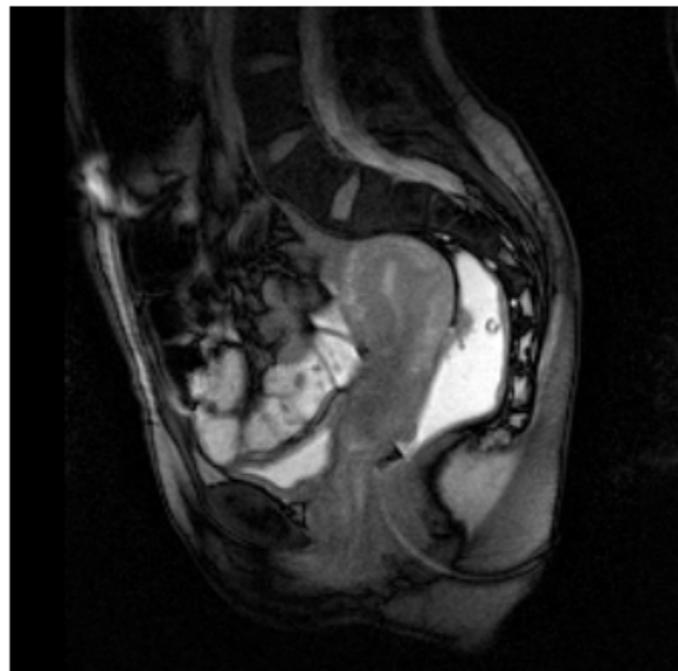
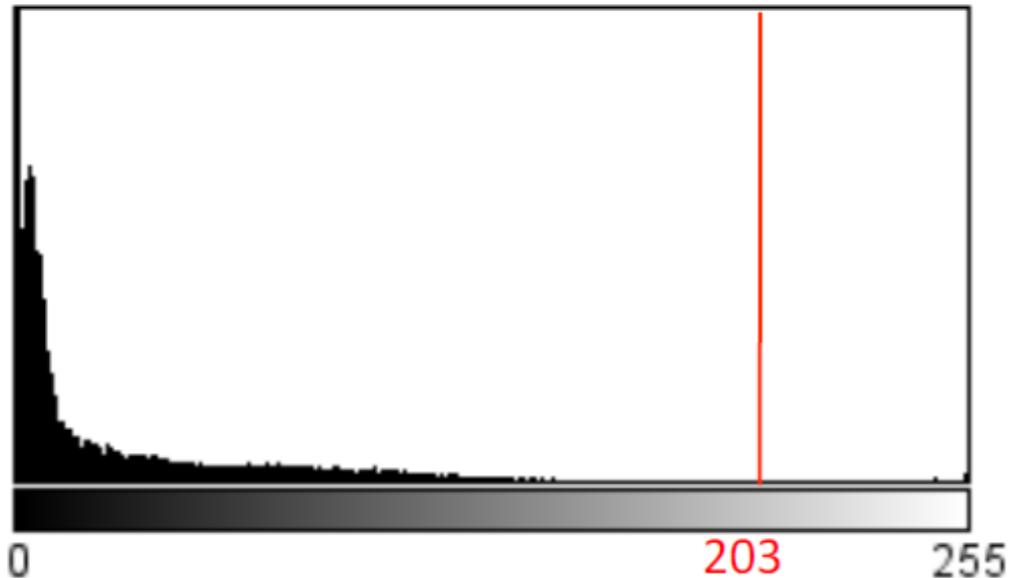
$x_s = 1$  /\* Étiquette « *terre* » au pixel  $(i,j)$  \*/

SINON

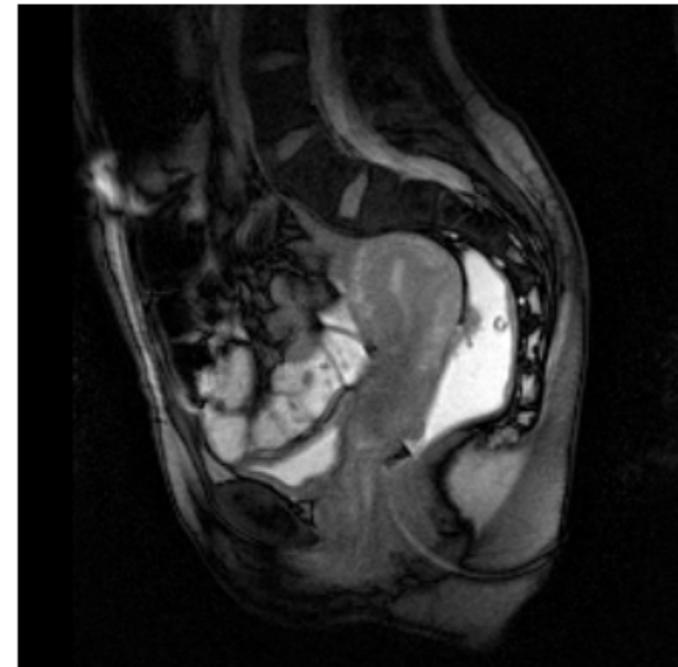
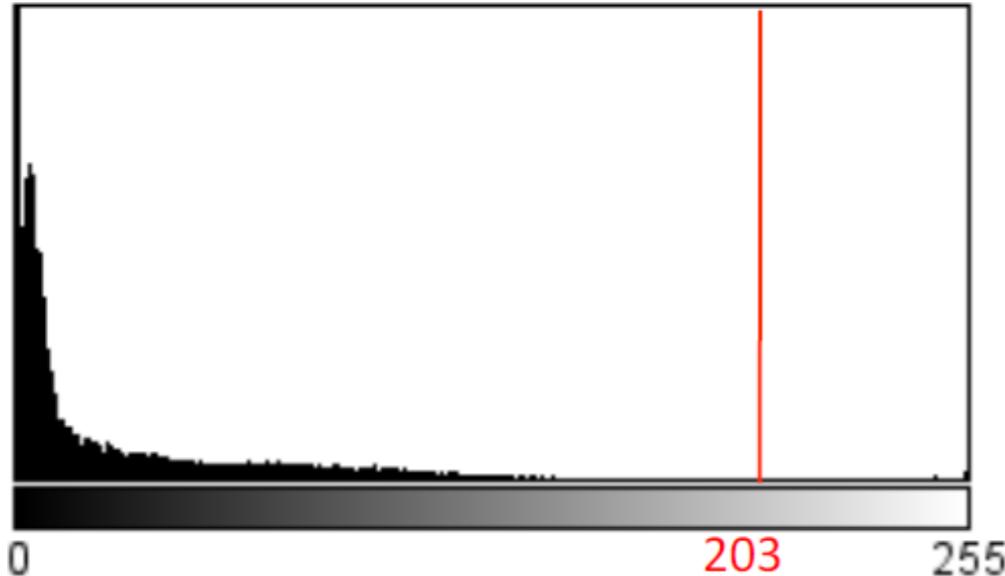
$x_s = 0$  /\* Étiquette « *mer* » au pixel  $(i,j)$  \*/



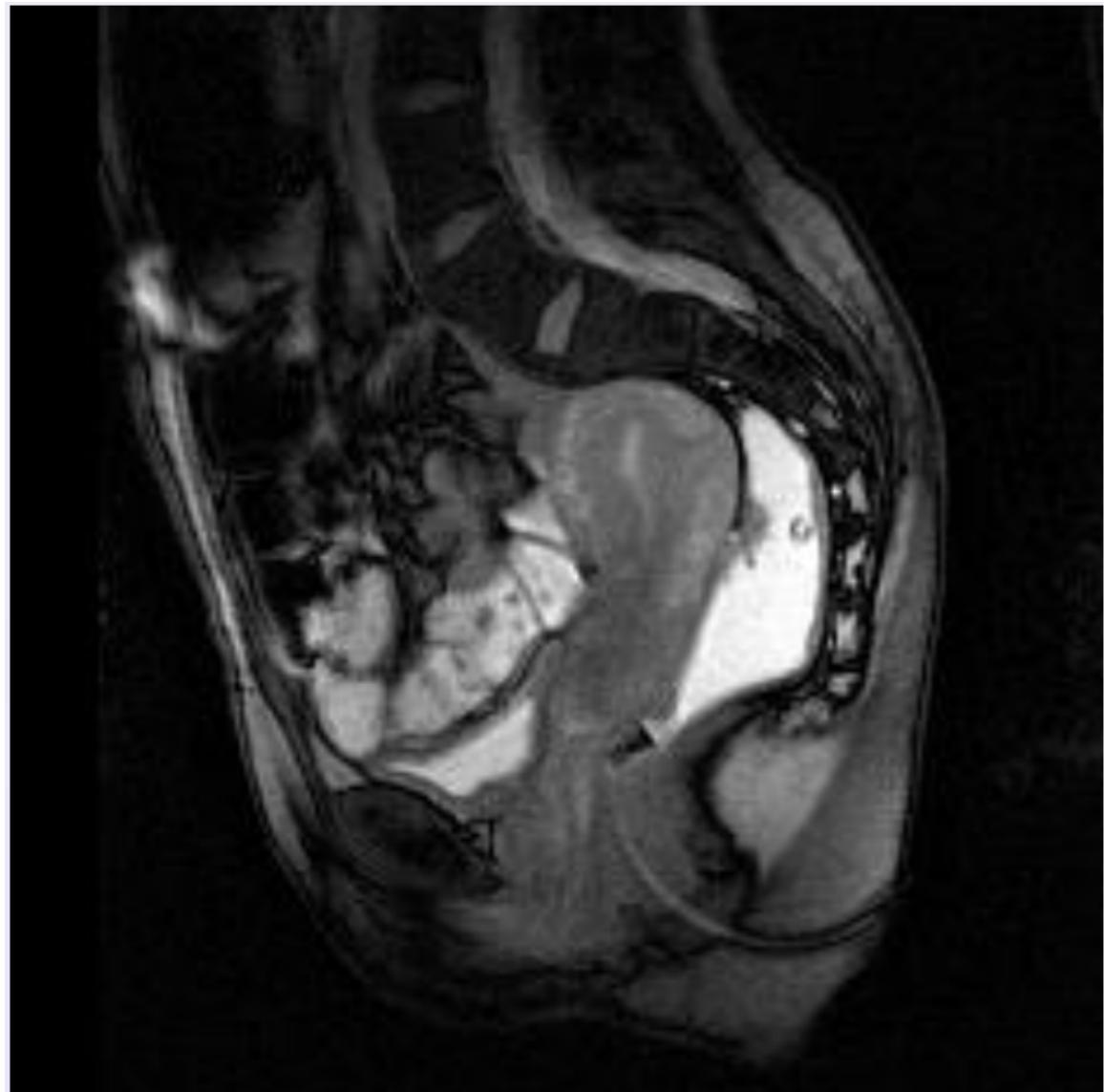
# Seuillage



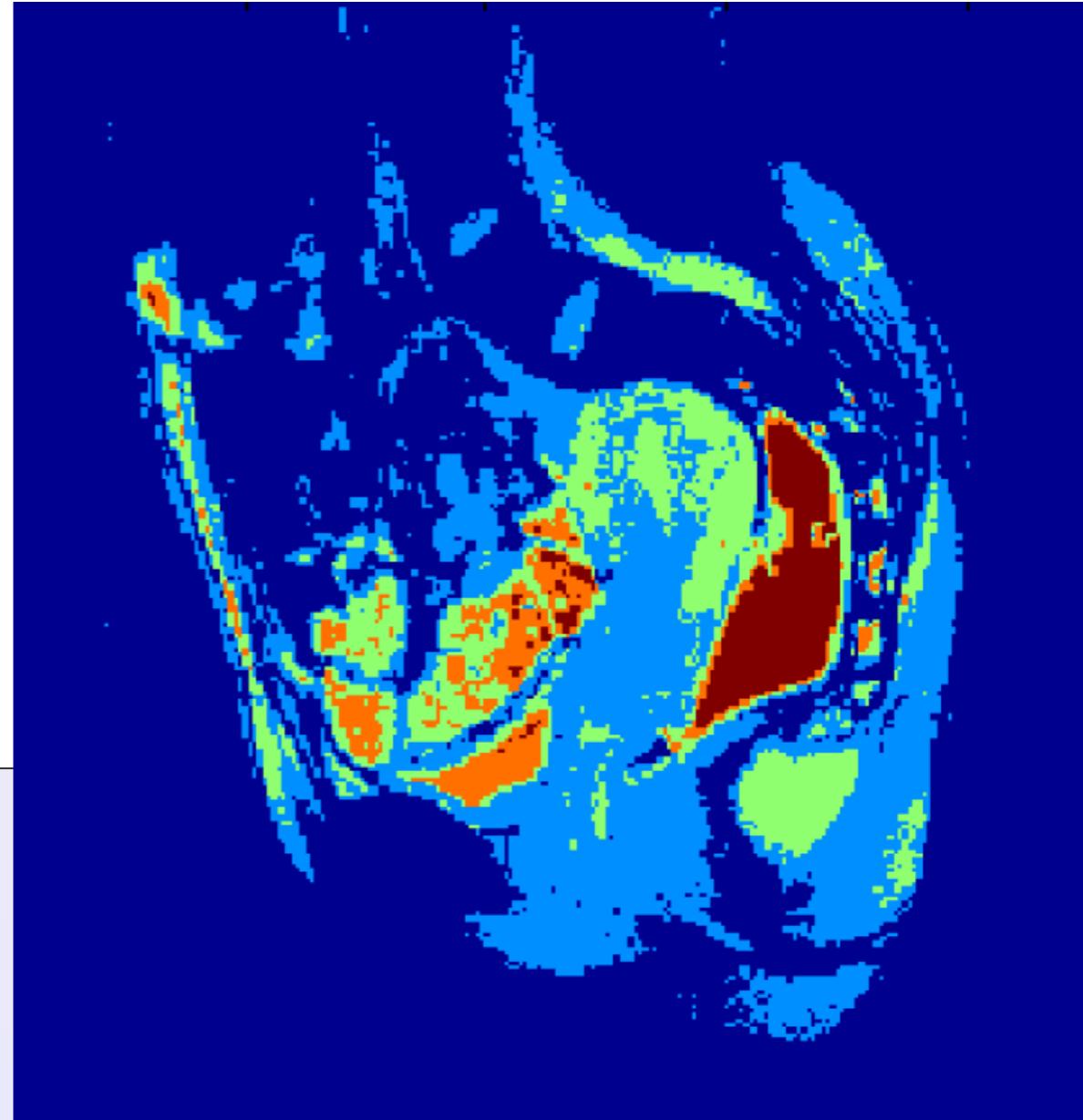
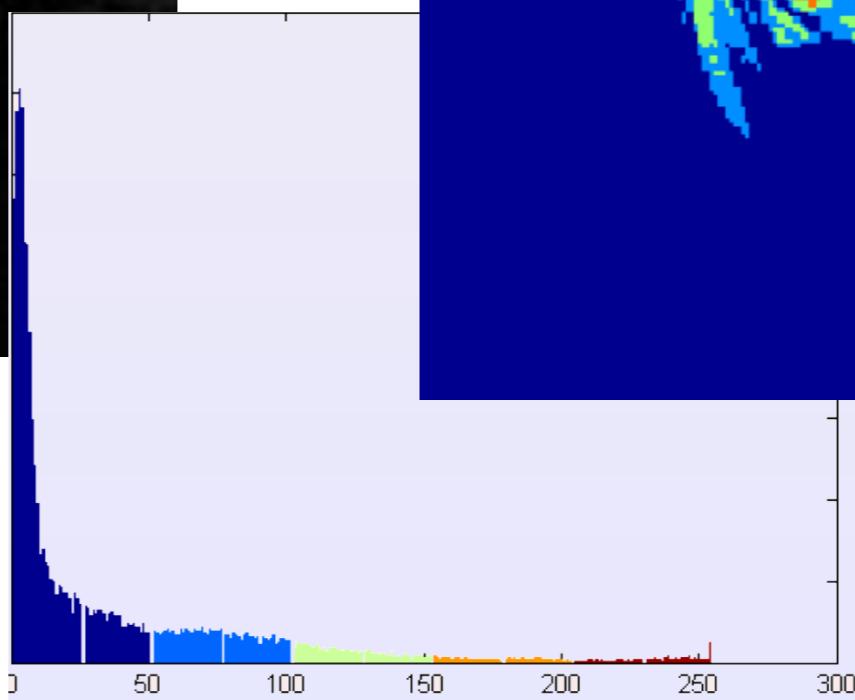
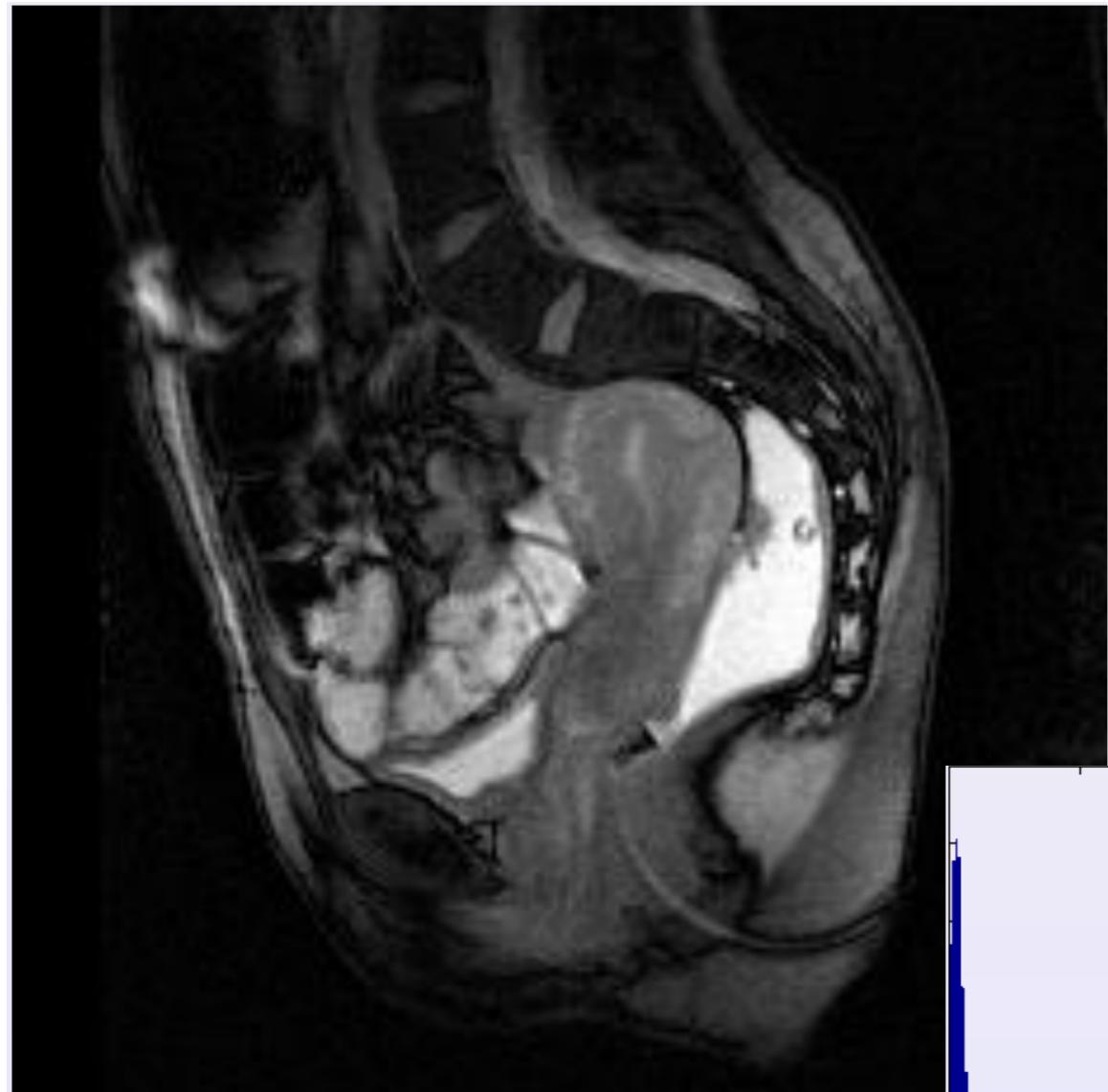
# Seuillage



# Seuillage



# Seuillage



# L'algorithme du seuil

Avantages:

- Trivial à implémenter;
- Très rapide

# L'algorithme du seuil

Avantages:

- Trivial à implémenter;
- Très rapide

Inconvénients:

# L'algorithme du seuil

Avantages:

- Trivial à implémenter;
- Très rapide

Inconvénients:

- Le seuil doit être fixé par un utilisateur

# L'algorithme du seuil

Avantages:

- Trivial à implémenter;
- Très rapide

Inconvénients:

- Le seuil doit être fixé par un utilisateur  
C'est un algorithme **supervisé**

# L'algorithme du seuil

Avantages:

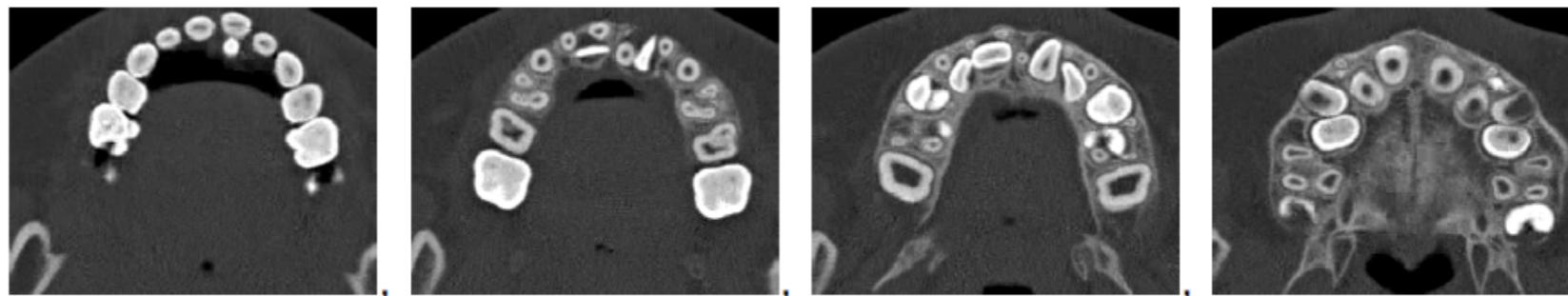
- Trivial à implémenter;
- Très rapide

Inconvénients:

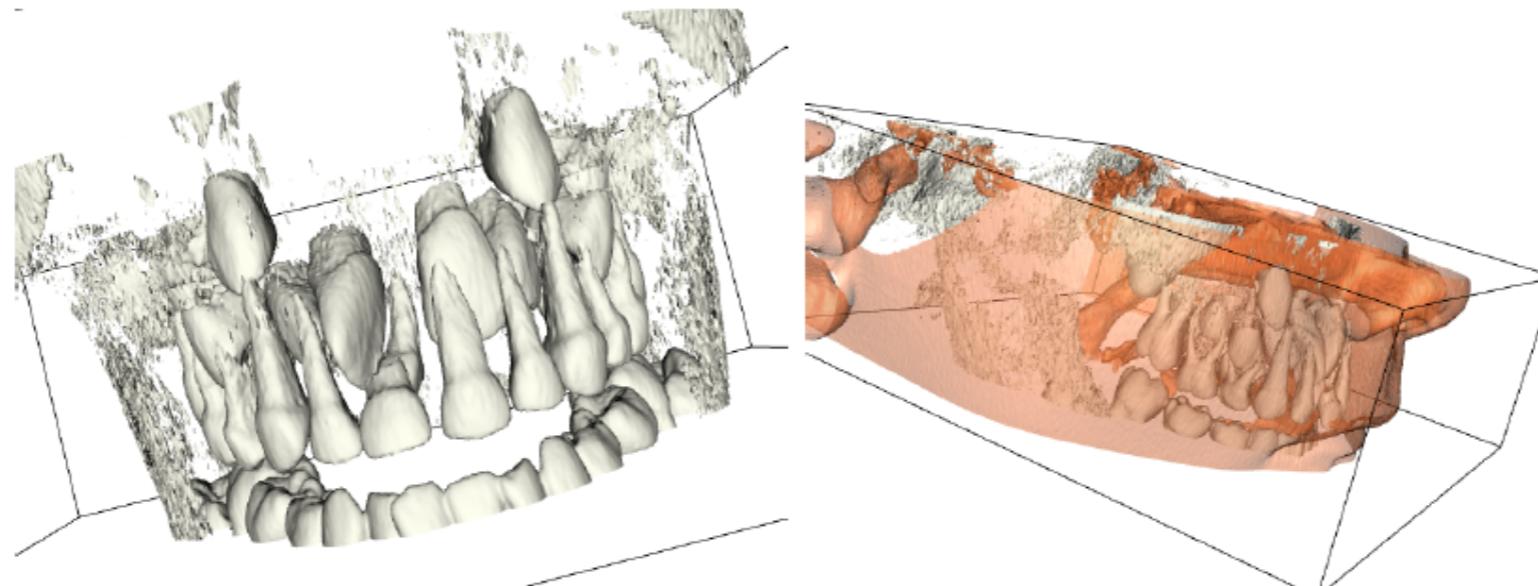
- Le seuil doit être fixé par un utilisateur  
C'est un algorithme **supervisé**
- Algorithme inefficace pour des images bruitées

# Algorithme du seuil

- Marche parfois très bien
- CT-scan grâce aux unités de Hounsfield



(a) 4 plans de coupe



(b) Seuillage à 80

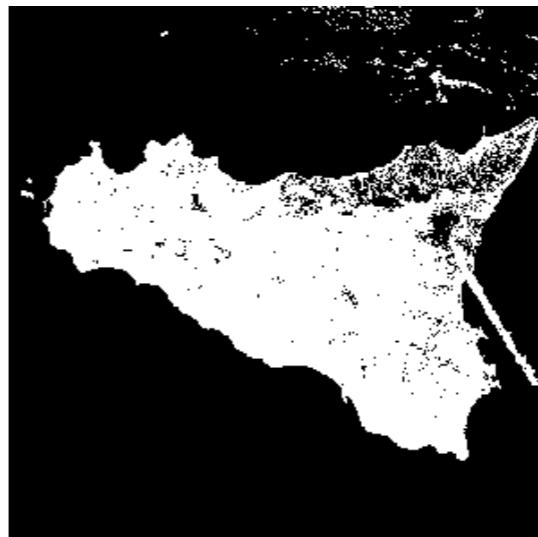
(c) Seuillage à 500

# L'algorithme du seuil

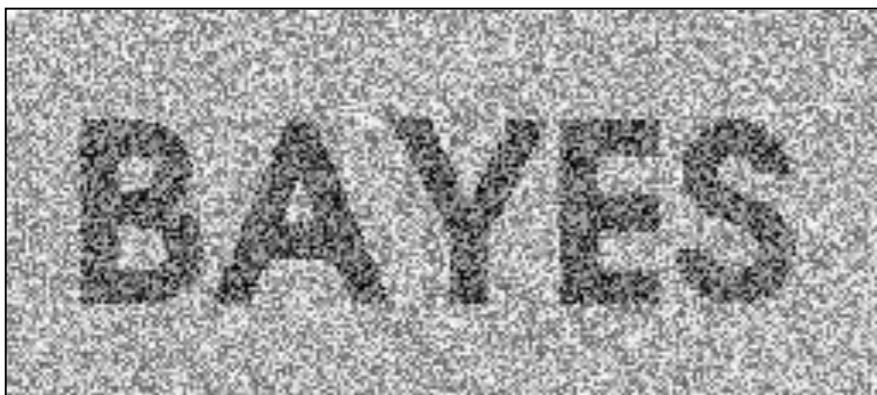
Algorithme inefficace pour des images bruitées



Segmentation  
2 Classes



Terre/Mer



Segmentation  
2 Classes



Lettres/Fond

# Segmentation

Seuillage d'histogramme automatique

# Seuillage automatique

- Choisir un seuil  $S$  initial (moyenne, médiane, ...)

# Seuillage automatique

- Choisir un seuil  $S$  initial (moyenne, médiane, ...)
- On seuille
  - 2 groupes de pixels/voxels de moyenne  $u_1$  et  $u_2$

# Seuillage automatique

- Choisir un seuil  $S$  initial (moyenne, médiane, ...)
- On seuille
  - 2 groupes de pixels/voxels de moyenne  $u_1$  et  $u_2$
- On calcule  $S = (u_1 + u_2)/2$

# Seuillage automatique

- Choisir un seuil  $S$  initial (moyenne, médiane, ...)
- On seuille
  - 2 groupes de pixels/voxels de moyenne  $u_1$  et  $u_2$
- On calcule  $S = (u_1 + u_2)/2$
- On itère jusqu'à ce que  $S$  soit constant

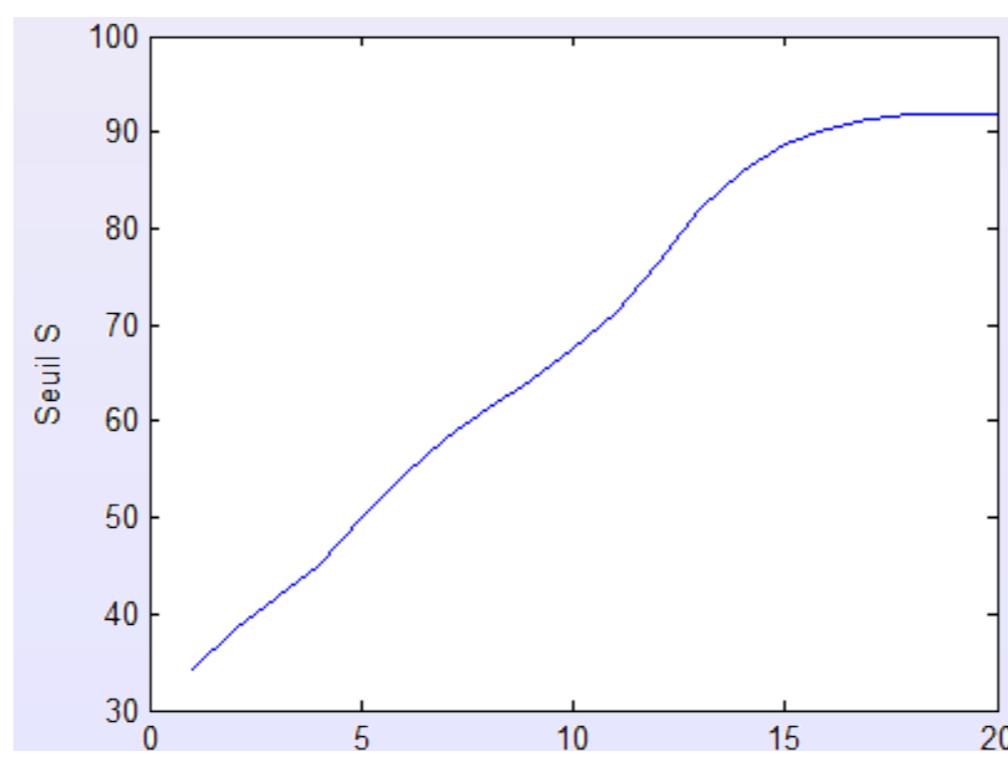
# Seuillage automatique

- Choisir un seuil  $S$  initial (moyenne, médiane, ...)
- On seuille
  - 2 groupes de pixels/voxels de moyenne  $u_1$  et  $u_2$
- On calcule  $S = (u_1 + u_2)/2$
- On itère jusqu'à ce que  $S$  soit constant



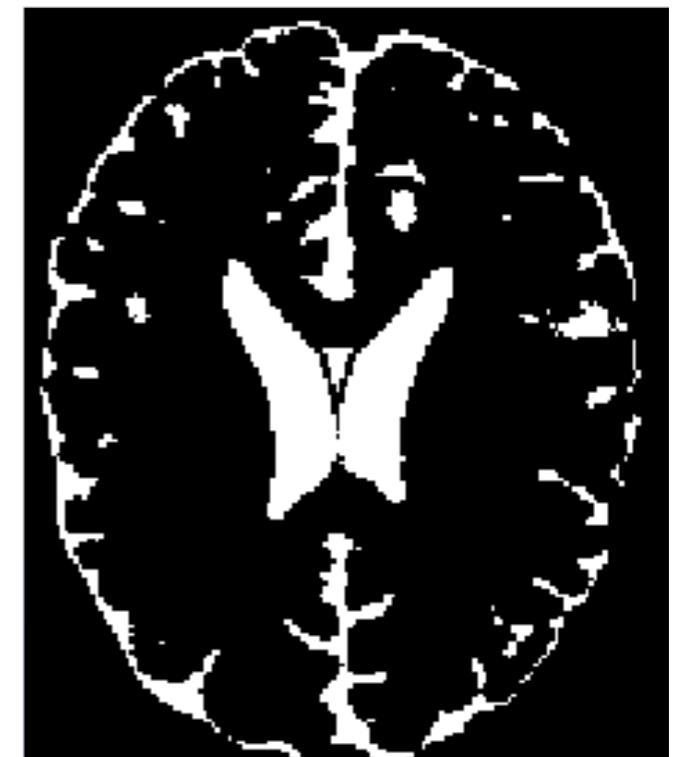
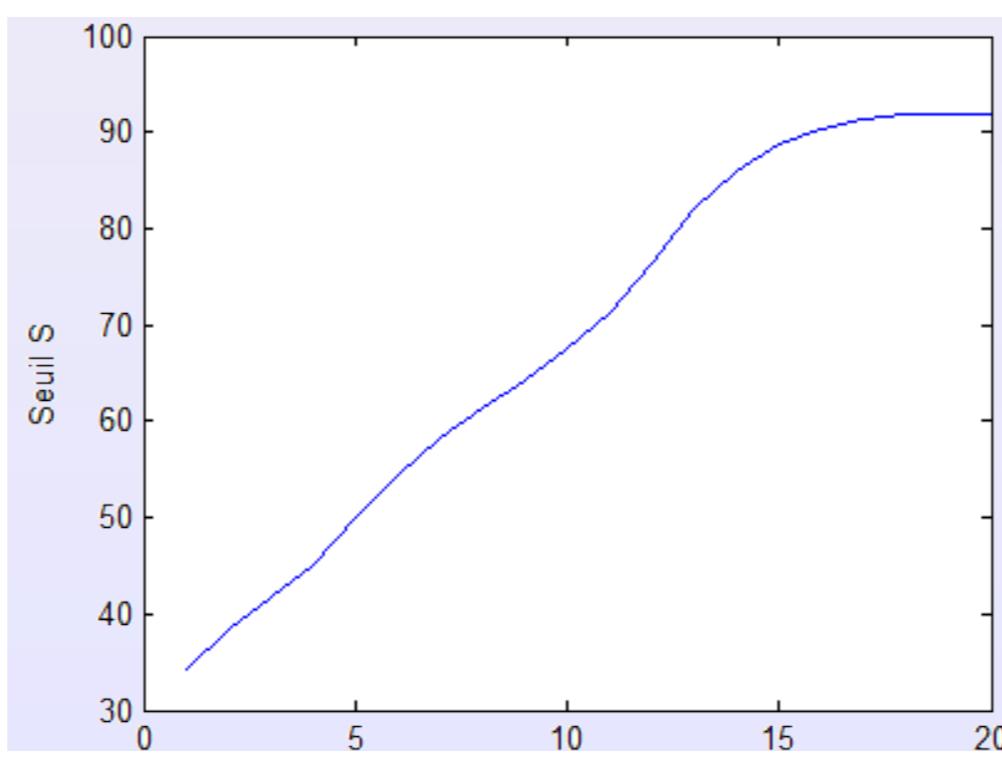
# Seuillage automatique

- Choisir un seuil  $S$  initial (moyenne, médiane, ...)
- On seuille
  - 2 groupes de pixels/voxels de moyenne  $u_1$  et  $u_2$
- On calcule  $S = (u_1 + u_2)/2$
- On itère jusqu'à ce que  $S$  soit constant



# Seuillage automatique

- Choisir un seuil  $S$  initial (moyenne, médiane, ...)
- On seuille
  - 2 groupes de pixels/voxels de moyenne  $u_1$  et  $u_2$
- On calcule  $S = (u_1 + u_2)/2$
- On itère jusqu'à ce que  $S$  soit constant



# Méthode d’Otsu (1979)

- Un seuil  $t$  définit deux groupes de pixels:  $C_1$  et  $C_2$
- On cherche alors le seuil qui minimise la variance intra-classe:

$$\sigma_w^2(t) = \omega_1(t)\sigma_1^2(t) + \omega_2(t)\sigma_2^2(t)$$

- Les poids  $w$  représentent la proba d’être dans la ième classe
- Les sigmas sont les variances de ces classes

# Méthode d'Otsu

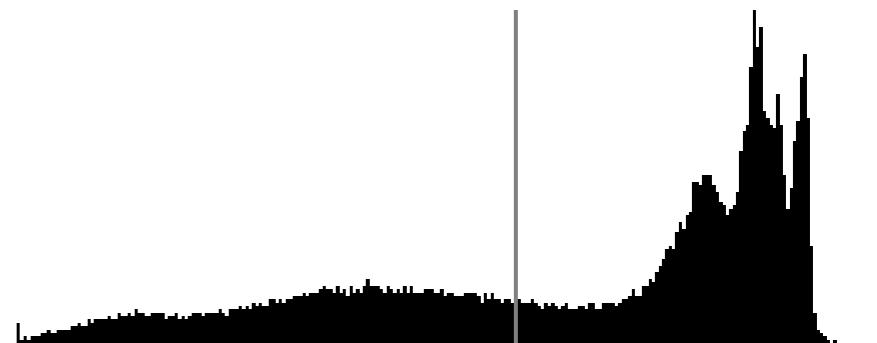
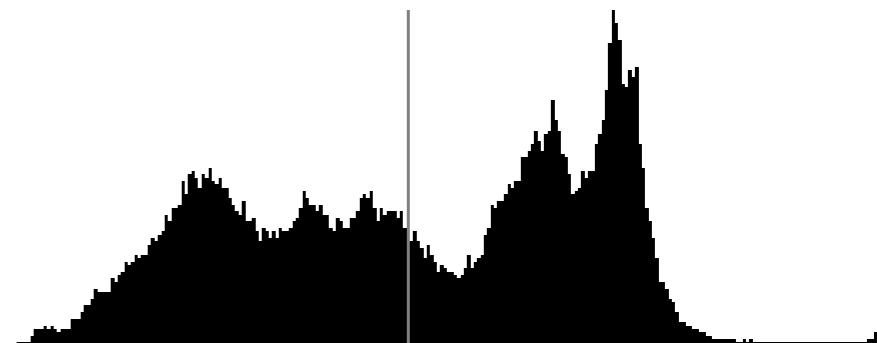
Greyscale Image

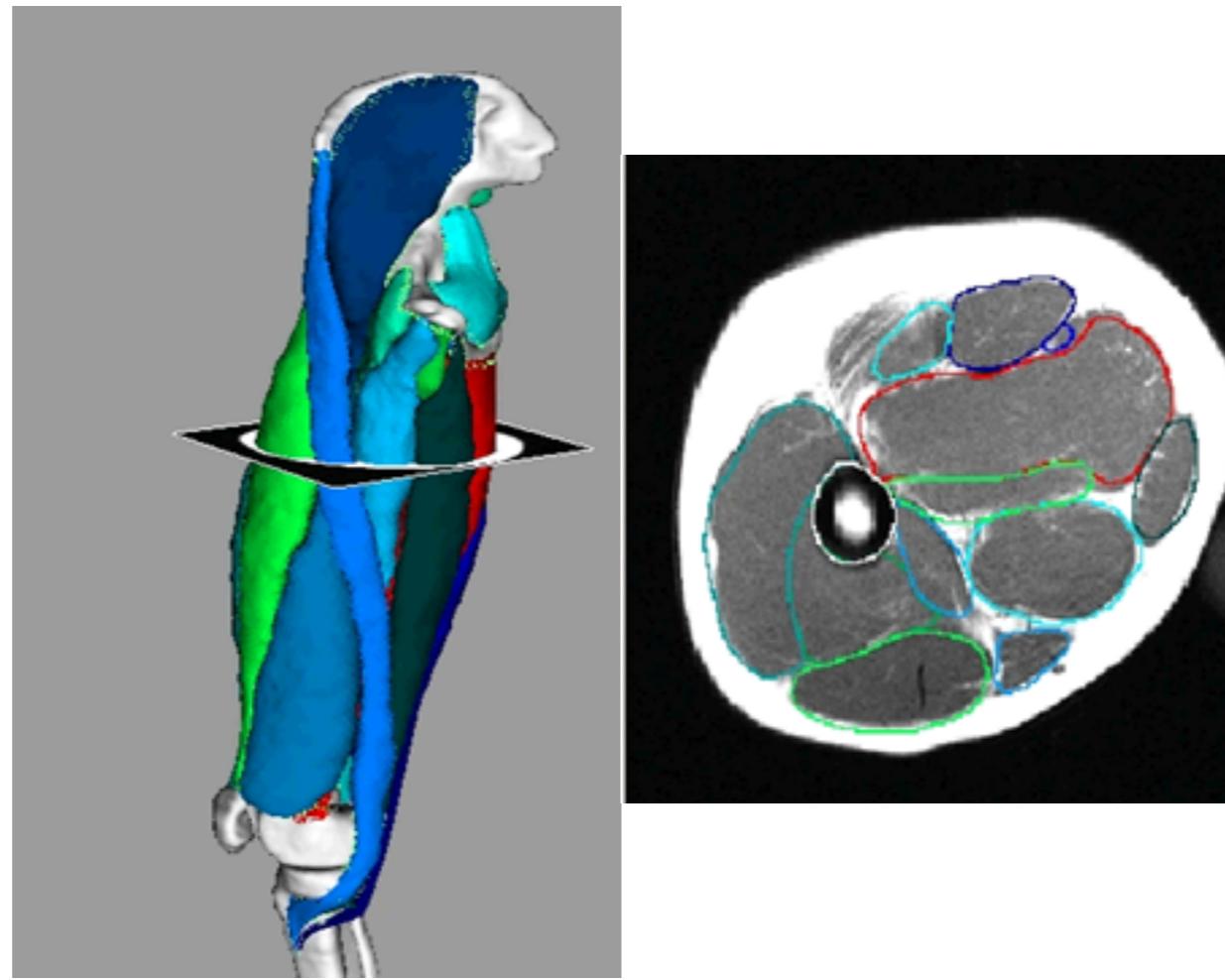


Binary Image



Histogram





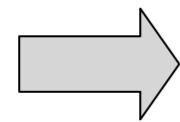
# Segmentation

*Approches par régions*

# Segmentation

Algorithme de la « baguette magique » (*magic wand*)  
ou d'inondation (*flood fill*)

# Baguette magique

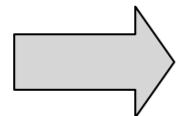


Segmentation « interactive » en vertu de laquelle l'utilisateur  
Sélectionne l'objet à segmenter.

## **Idée :**

- inonder la région entourant un (ou plusieurs) pixels sélectionnés

# Baguette magique



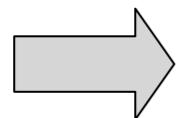
Segmentation « interactive » en vertu de laquelle l'utilisateur  
Sélectionne l'objet à segmenter.

## **Idée :**

- inonder la région entourant un (ou plusieurs) pixels sélectionnés

## **Critères :**

# Baguette magique



Segmentation « interactive » en vertu de laquelle l'utilisateur  
Sélectionne l'objet à segmenter.

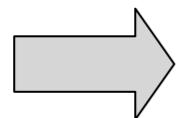
## **Idée :**

- inonder la région entourant un (ou plusieurs) pixels sélectionnés

## **Critères :**

- En fonction d'un pixel  $s$  sélectionné:

# Baguette magique



Segmentation « interactive » en vertu de laquelle l'utilisateur Sélectionne l'objet à segmenter.

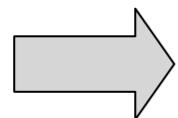
## **Idée :**

- inonder la région entourant un (ou plusieurs) pixels sélectionnés

## **Critères :**

- En fonction d'un pixel  $s$  sélectionné:
  - les pixels de la région segmentée doivent être **connectés** au pixel  $s$

# Baguette magique



Segmentation « interactive » en vertu de laquelle l'utilisateur  
Sélectionne l'objet à segmenter.

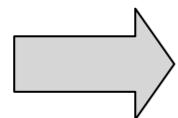
## **Idée :**

- inonder la région entourant un (ou plusieurs) pixels sélectionnés

## **Critères :**

- En fonction d'un pixel  $s$  sélectionné:
  - les pixels de la région segmentée doivent être **connectés** au pixel  $s$
  - pour tout pixel  $q$  de la région segmentée,

# Baguette magique



Segmentation « interactive » en vertu de laquelle l'utilisateur Sélectionne l'objet à segmenter.

## **Idée :**

- inonder la région entourant un (ou plusieurs) pixels sélectionnés

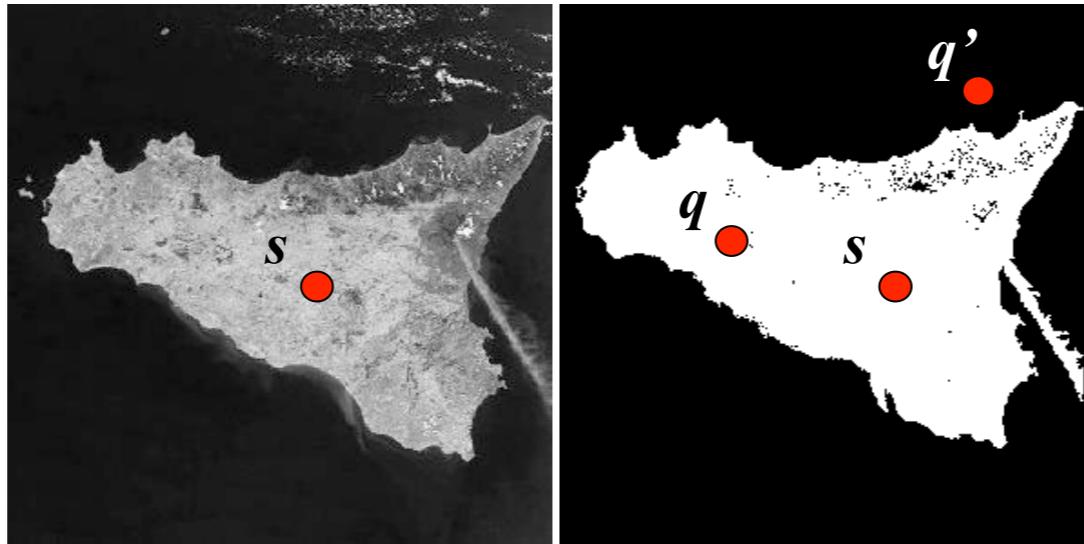
## **Critères :**

- En fonction d'un pixel  $s$  sélectionné:
  - les pixels de la région segmentée doivent être **connectés** au pixel  $s$
  - pour tout pixel  $q$  de la région segmentée,

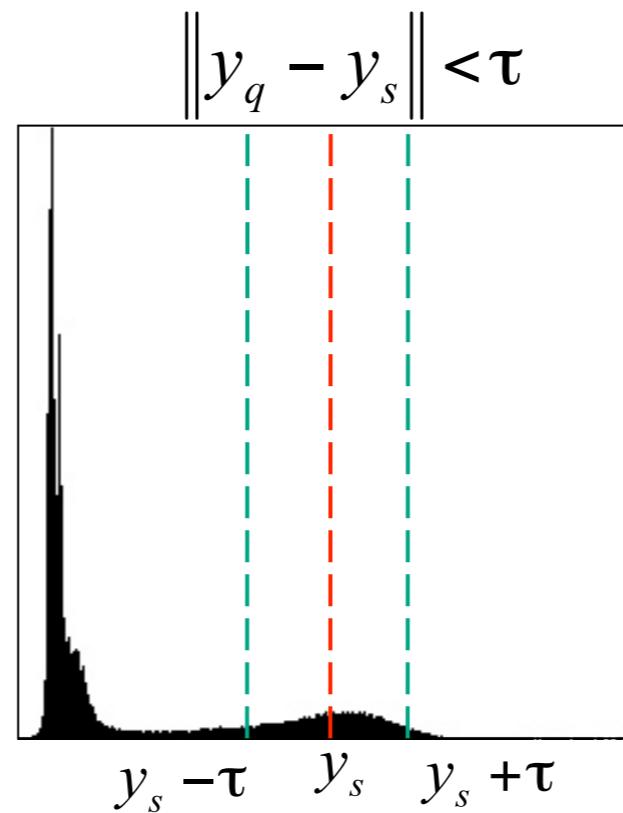
$$\|y_q - y_s\| < \text{Seuil}$$

# Baguette magique

Exemple:



s et q sont connectés, mais s et q' ne sont pas connecté (aucun chemin entre p et q' constitué exclusivement de pixels segmentés).



## Baquette magique

$x \leftarrow -1$

$s$  = pixel sélectionné par l'utilisateur

$x = \text{Inondation}(y, x, s, \tau)$

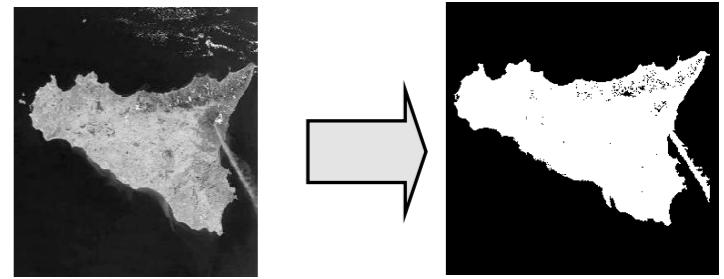
POUR CHAQUE site  $s$  du champ d'étiquettes FAIRE

SI  $x_s > 0$  ALORS

$x_s = 1$  /\* Étiquette « *terre* » au pixel (i,j) \*/

SINON

$x_s = 0$  /\* Étiquette « *mer* » au pixel (i,j) \*/



y

x

## Baquette magique

$x \leftarrow -1$

$s$  = pixel sélectionné par l'utilisateur

$x = \text{Innondation}(y, x, s, \tau)$

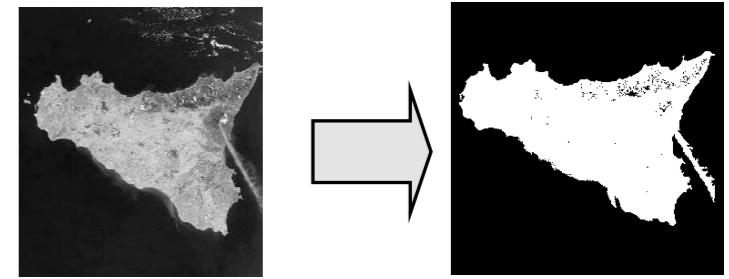
POUR CHAQUE site  $s$  du champ d'étiquettes FAIRE

SI  $x_s > 0$  ALORS

$x_s = 1$  /\* Étiquette « *terre* » au pixel (i,j) \*/

SINON

$x_s = 0$  /\* Étiquette « *mer* » au pixel (i,j) \*/



y

x

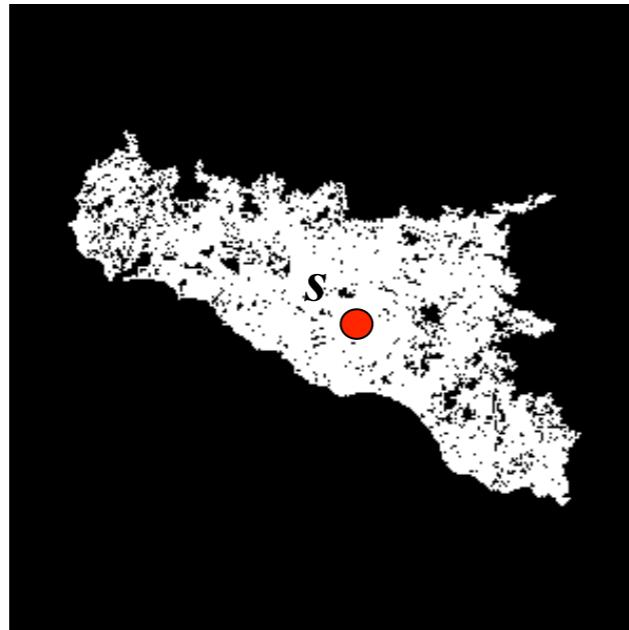
## Innondation

POUR chaque voisin  $r \in \eta_s$

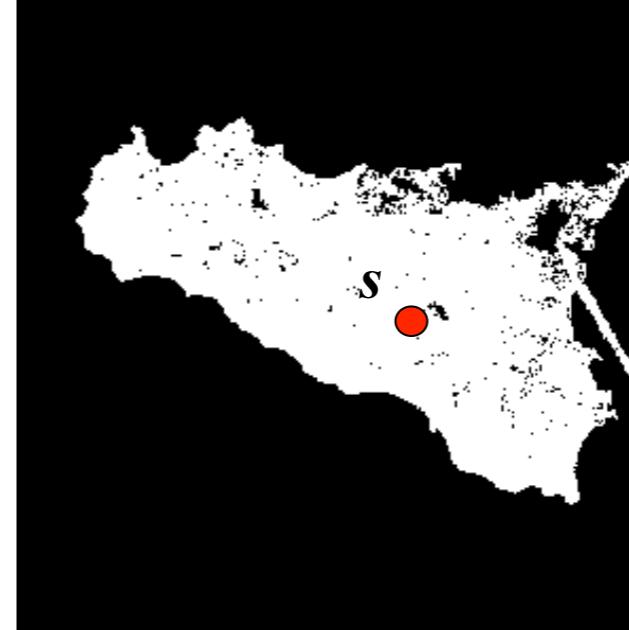
SI  $x_r < 0$  ET  $\|y_r - y_s\| < \delta$  ALORS

$x = \text{Innondation}(y, x, r, \tau)$

# Baguette magique



$\tau=50$



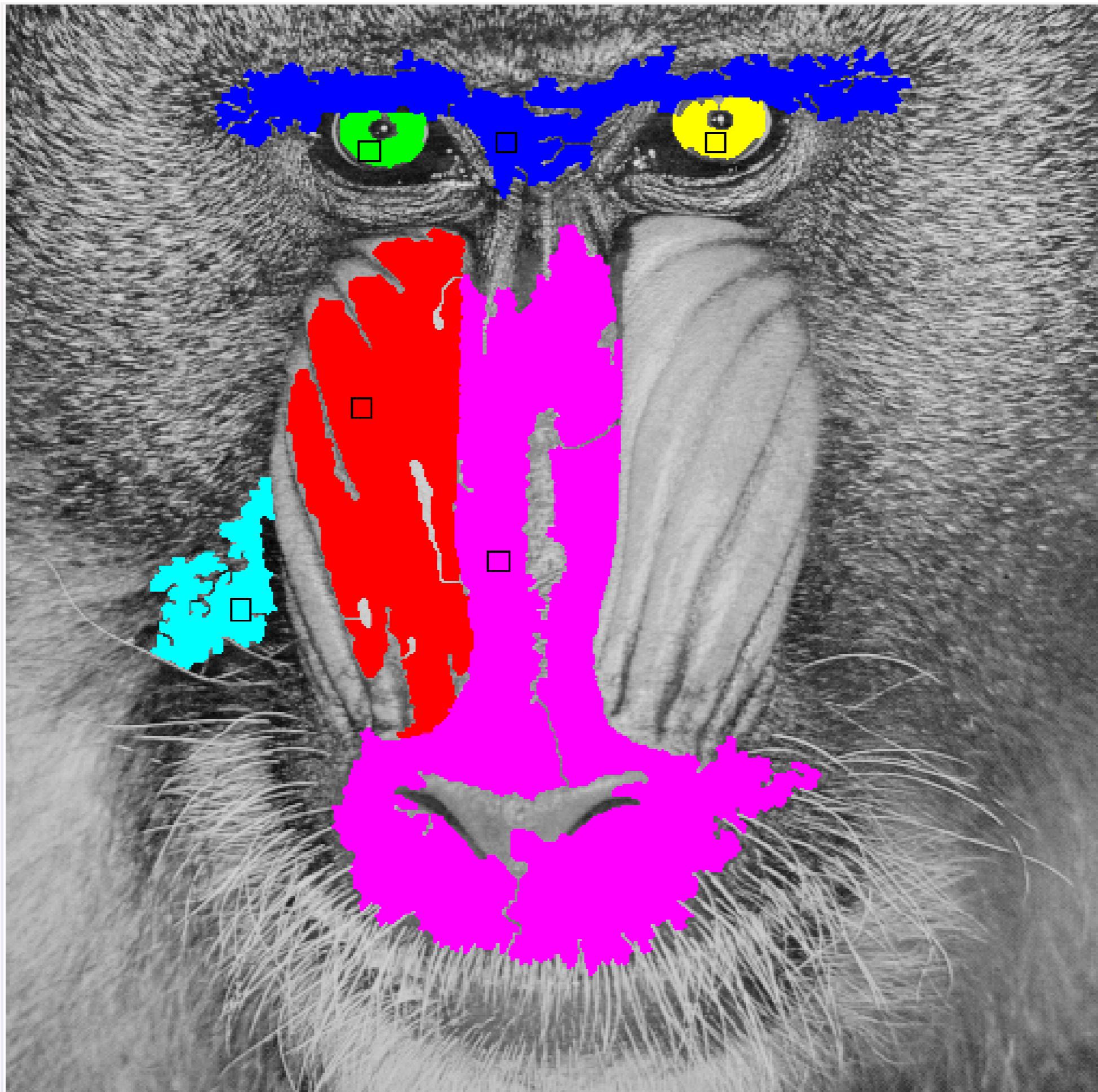
$\tau=80$



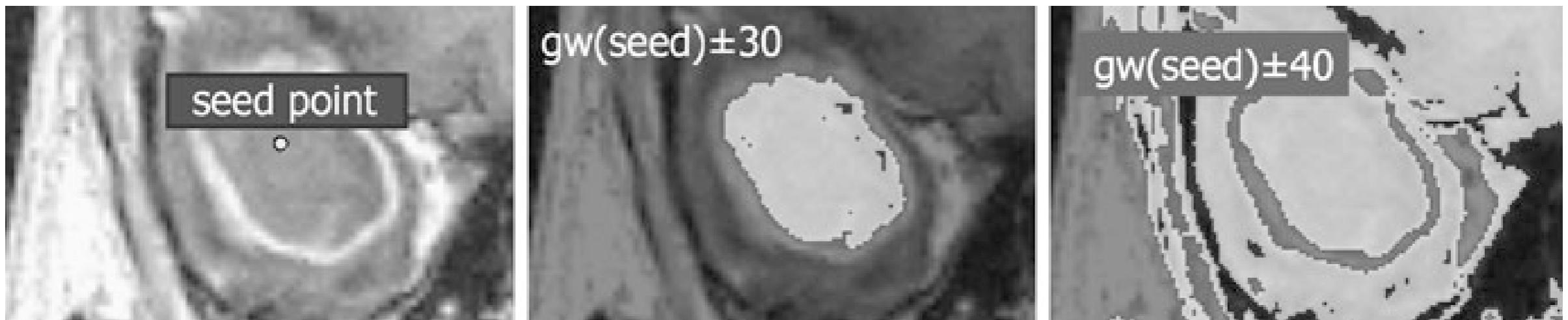
$\tau=120$



$\tau=140$



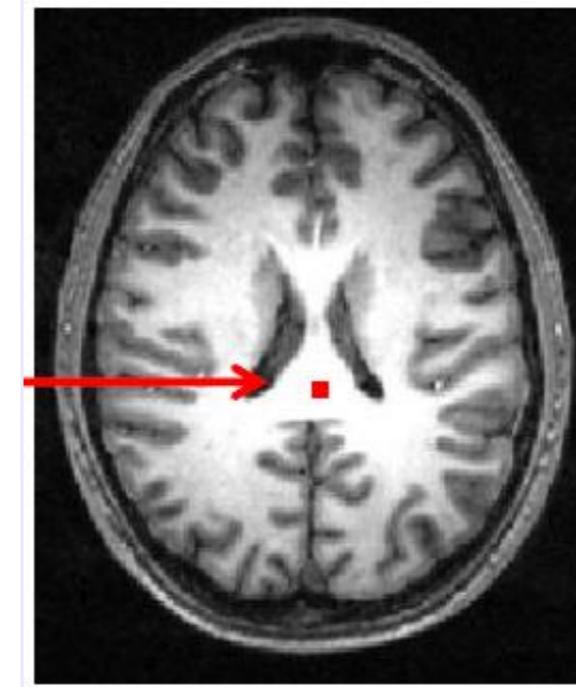
# Baguette magique (region growing)



p. 214 - Toennies Med IA 2012

# Limites et avantages

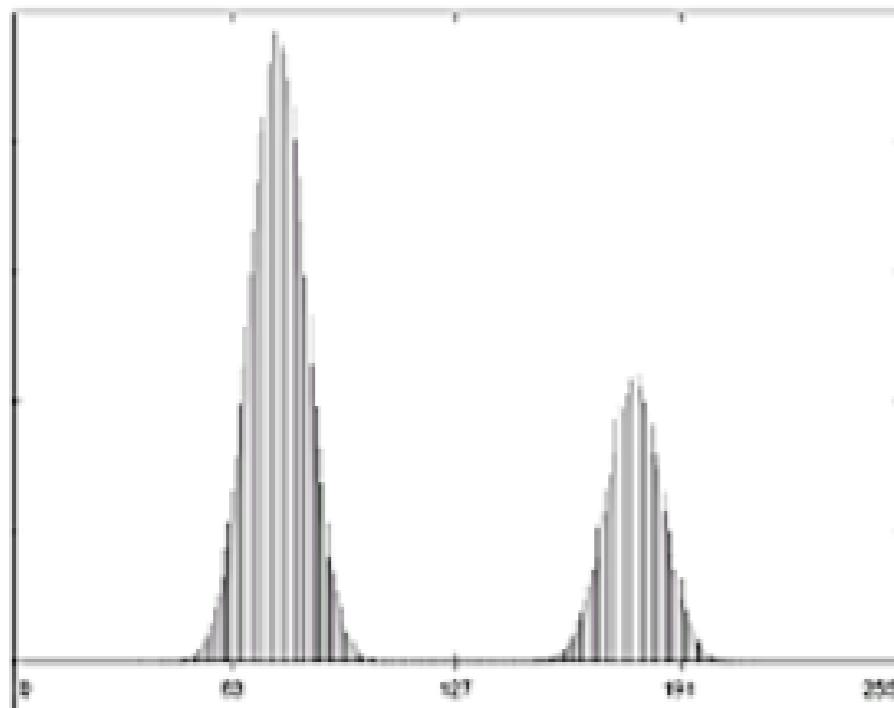
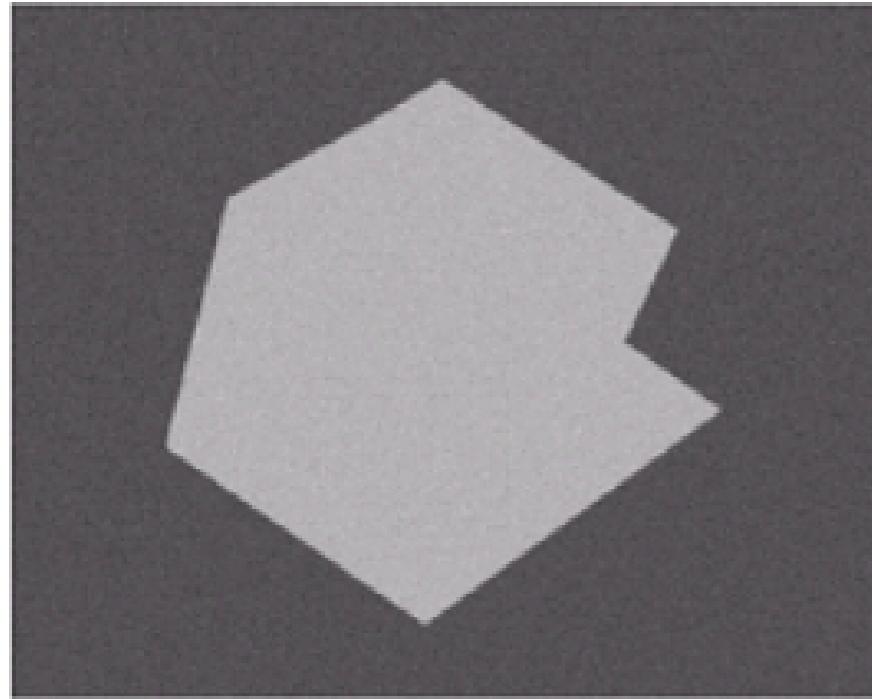
- Limites
  - Influence du choix des graines
  - Influence de l'ordre de parcours des points frontière
  - Choix du seuil
- Avantages
  - Implémentation très rapide si on utilise une structure de donnée adaptée (file d'attente)



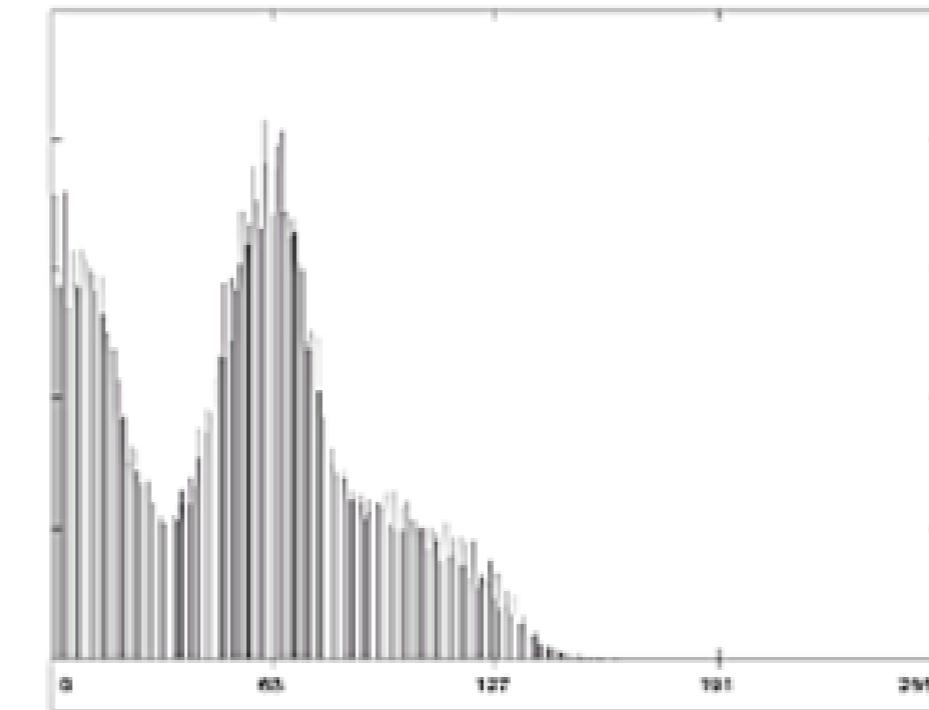
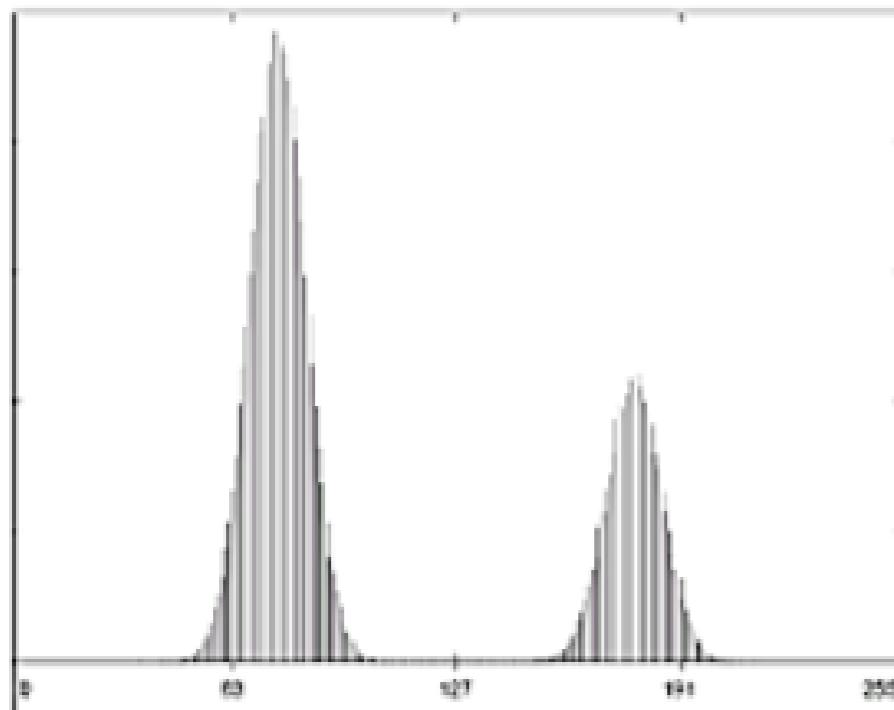
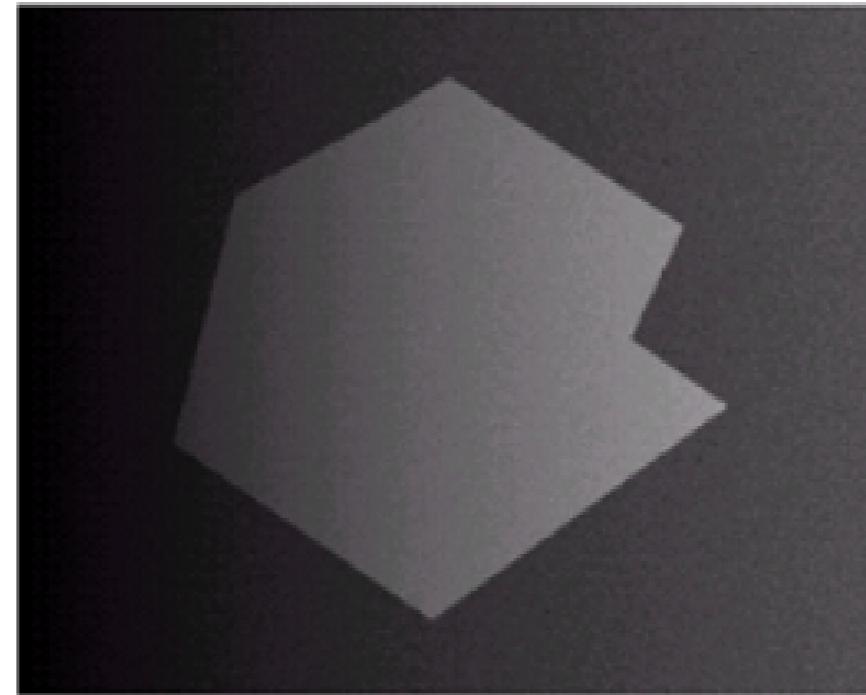
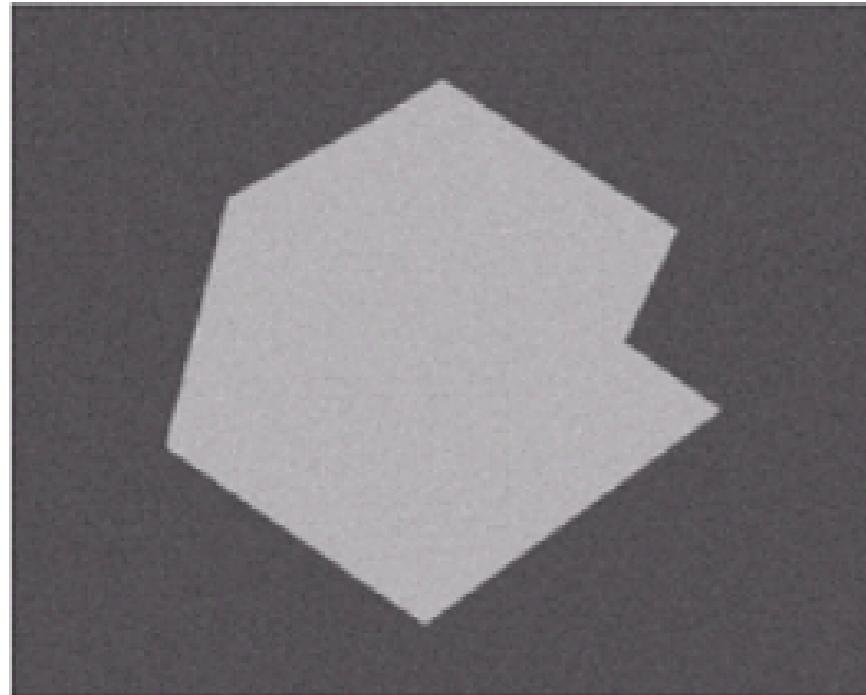
# Segmentation

*limites des approches globales*

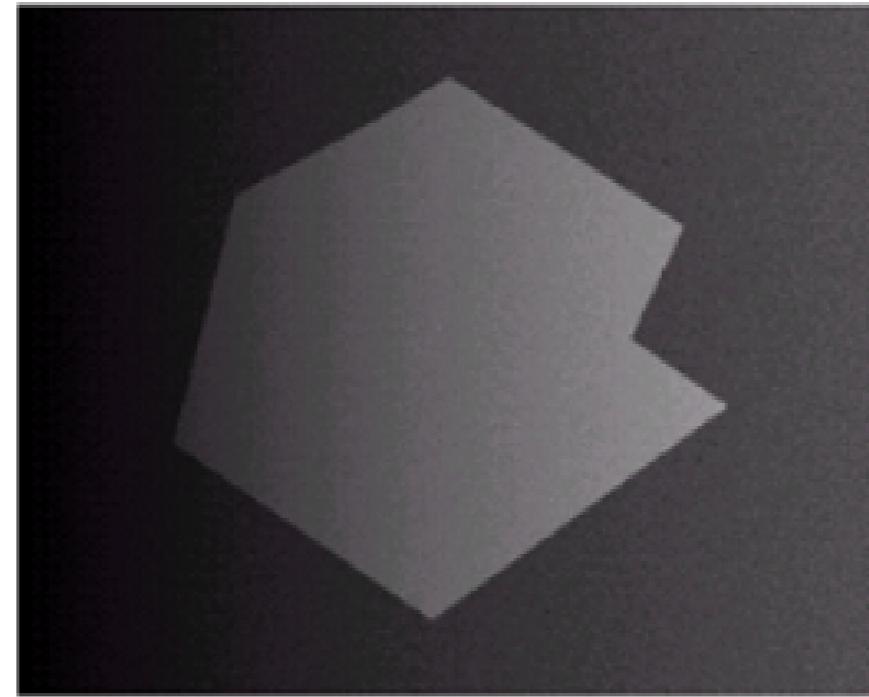
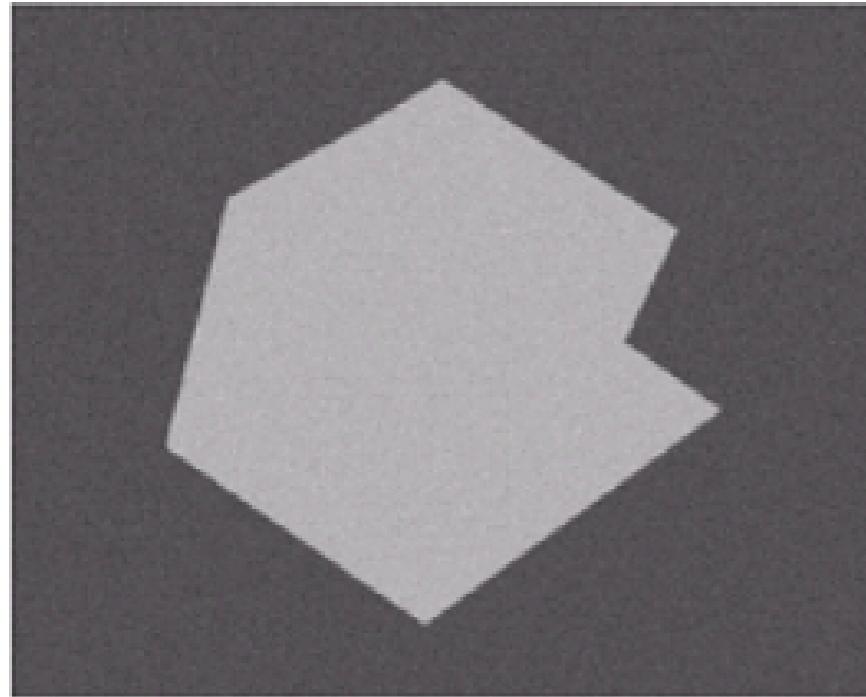
# Limites des approches globales



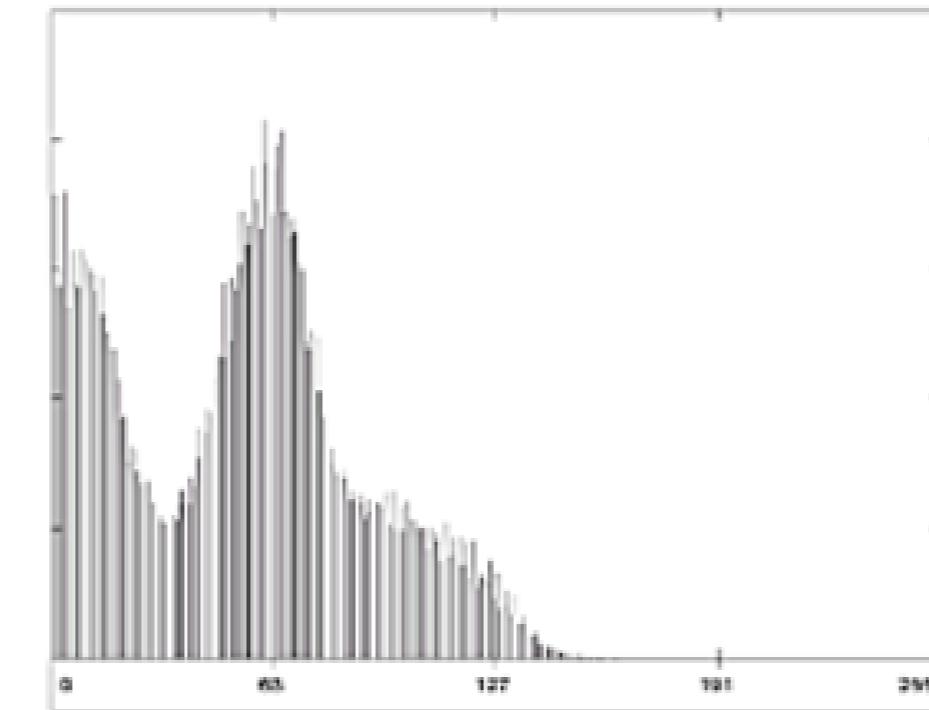
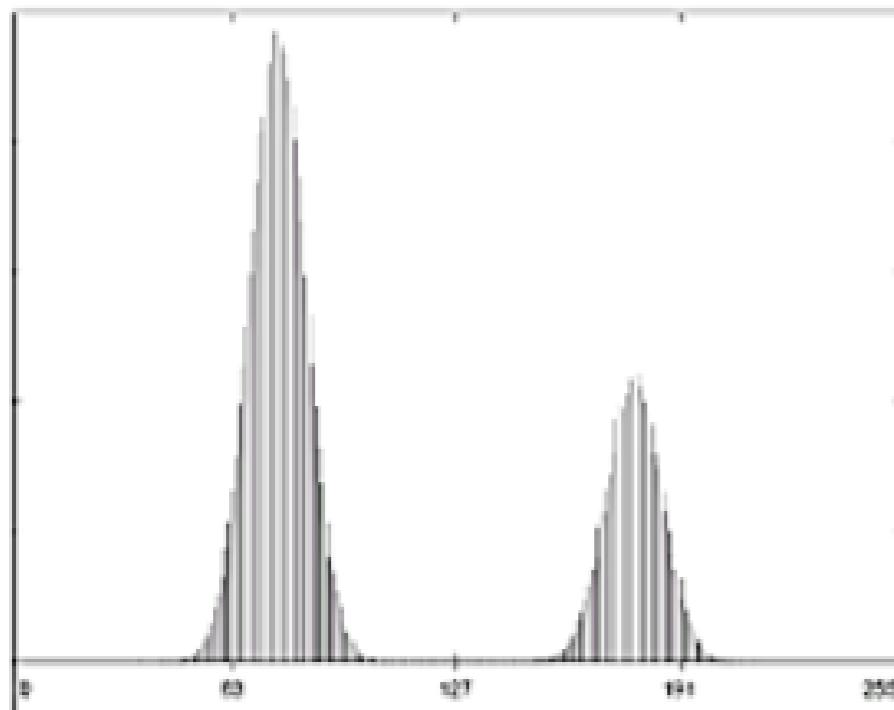
# Limites des approches globales



# Limites des approches globales

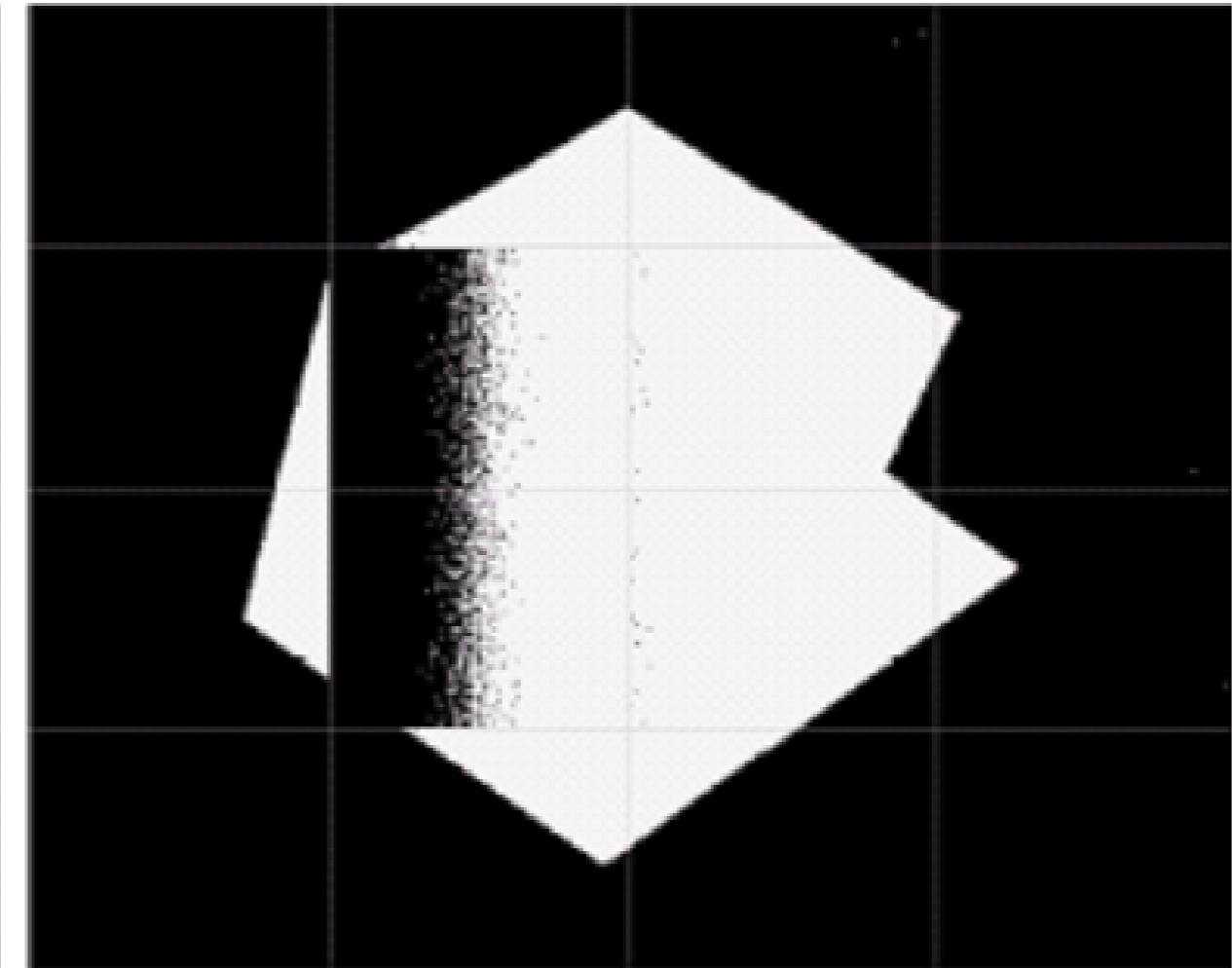
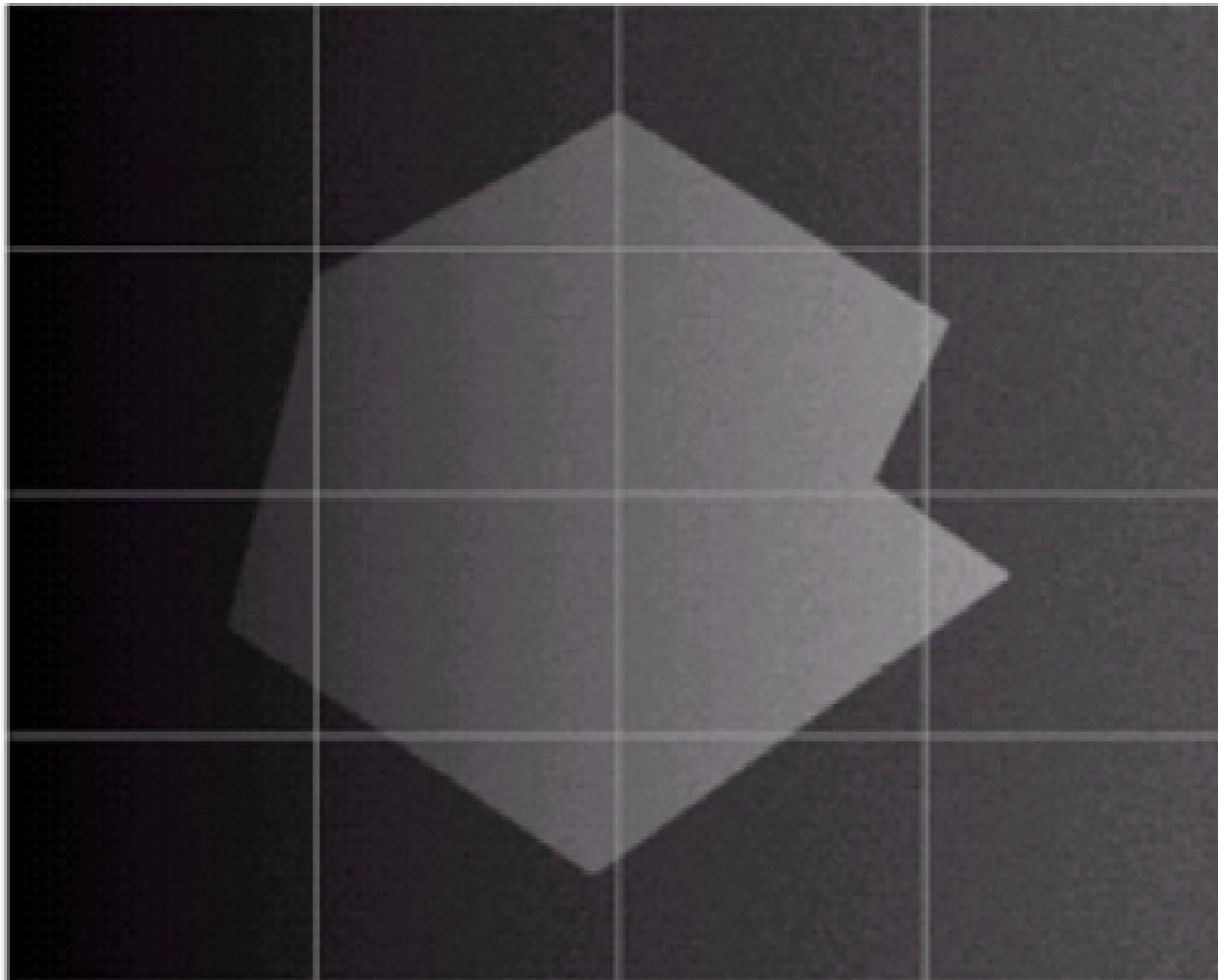


Otsu



# Seuillage adaptatif

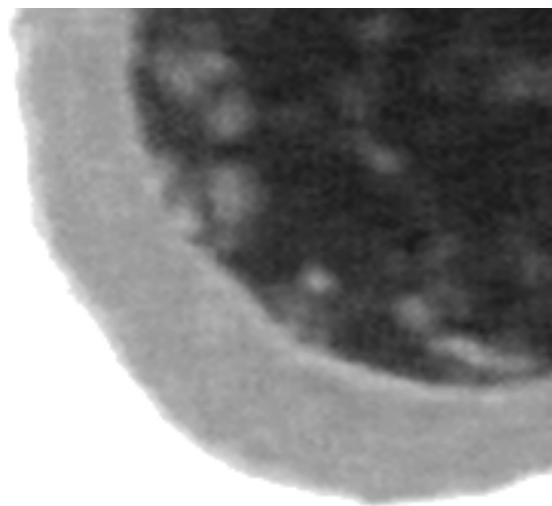
- On divise l'image en un certain nombre de sous-régions
- On seuil sur chaque région



# Seuillage d'hystérésis

- Compromis dans le choix du seuil de la norme du gradient
  - Fermeture des contours (peu de faux négatifs)
  - Immunité au bruit (peu de faux positifs)
- Choix de deux seuils, un haut (spécificité) et un bas (sensibilité)
  - On sélectionne les points au-dessus du seuil haut
  - Pour tout point  $<$  seuil haut et  $>$  seuil bas, on regarde s'il existe un point se son voisinage  $>$  seuil haut
  - Si oui, on connecte

# Seuillage d'hystérésis



Seuillage simple  
 $S=50$



Seuillage simple  
 $S=100$



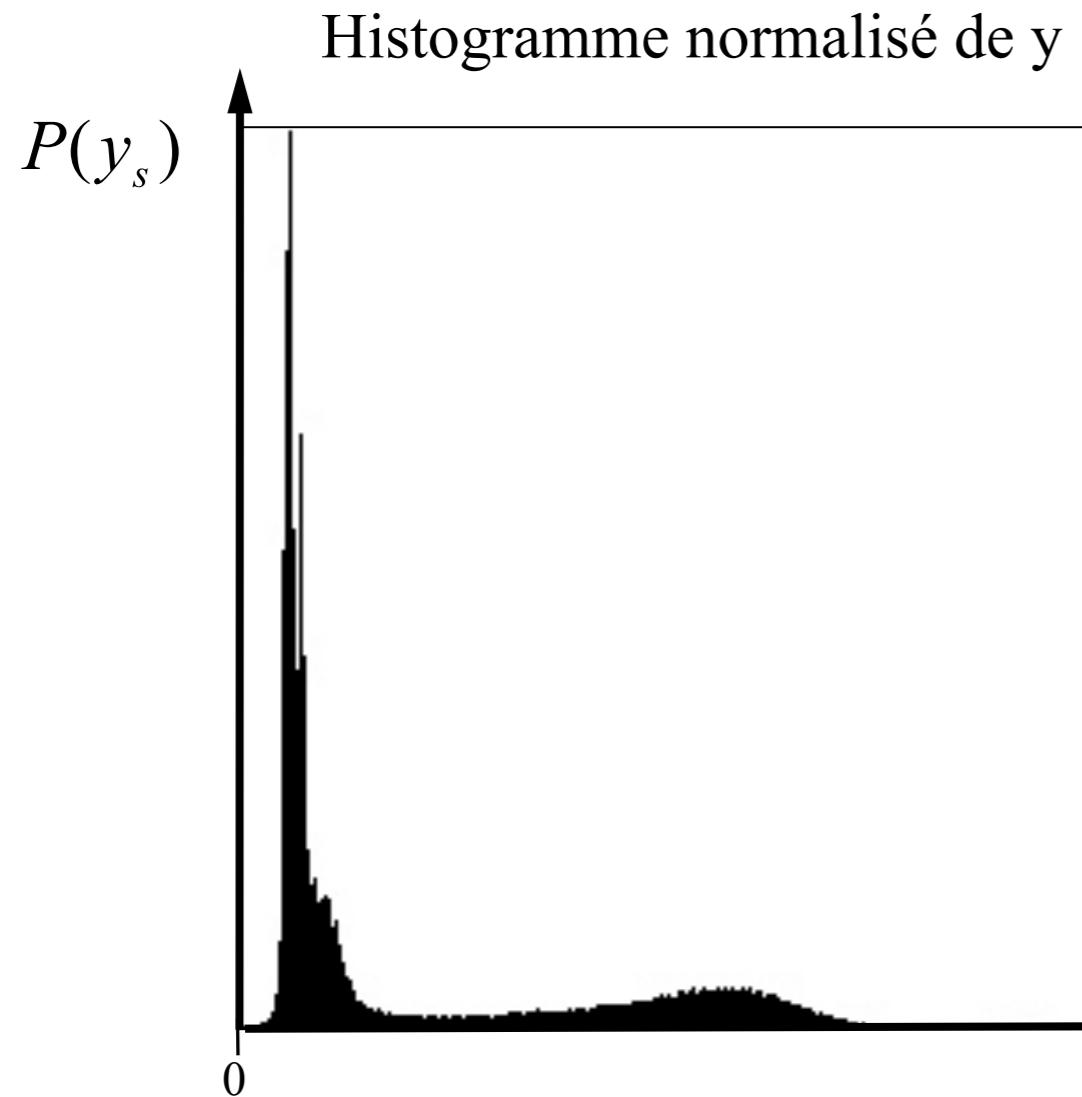
Seuillage par  
hystérésis  
 $Sh=100$ ,  $Sb=50$

# Segmentation

Version probabiliste de  
l'algorithme du seuil

# Trouver le *meilleur* seuil sans supervision

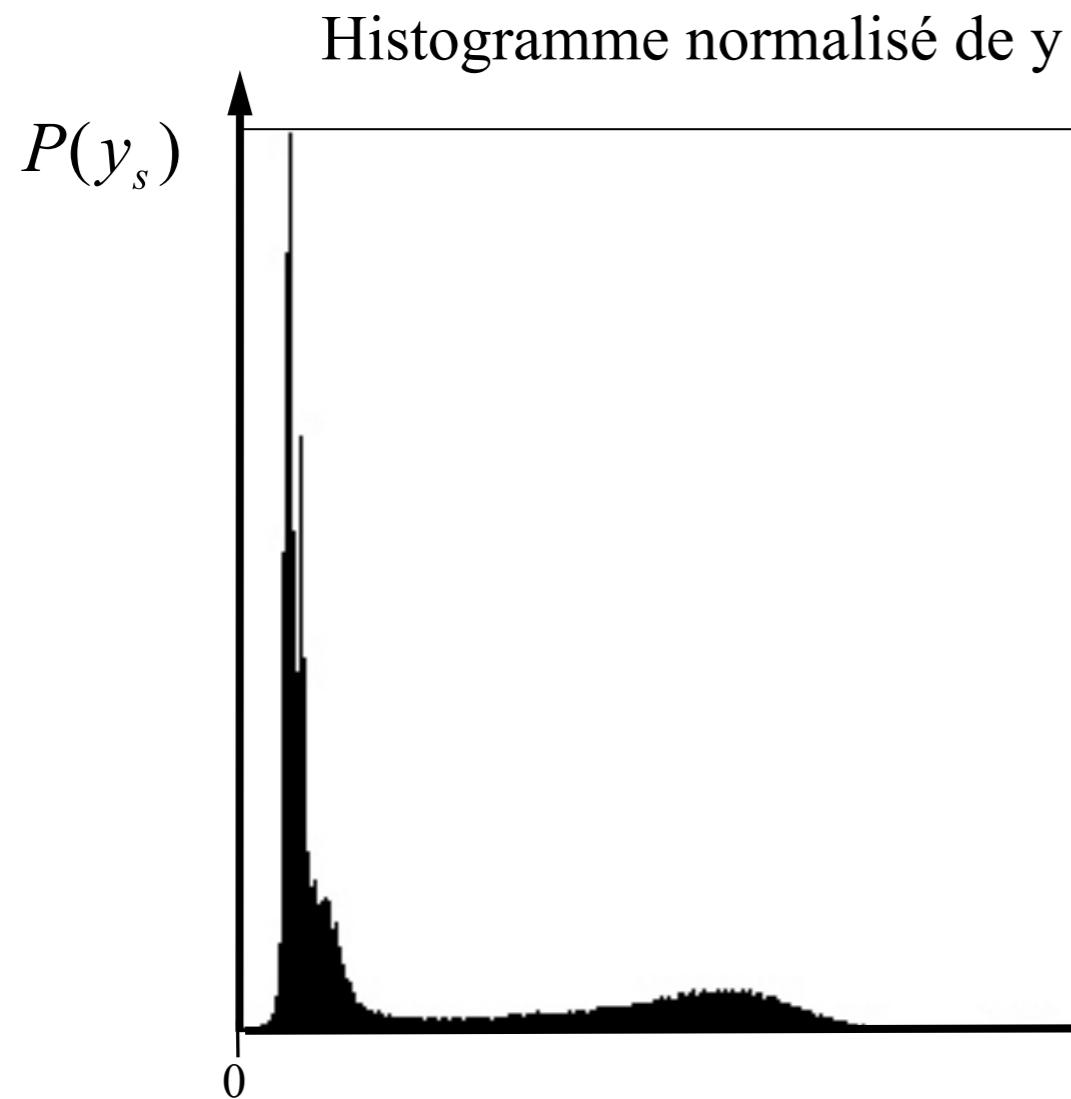
## Quelques définitions



$P(y_s)$  Distribution des niveaux de gris dans l'image  $y$ .

# Trouver le *meilleur* seuil sans supervision

## Quelques définitions

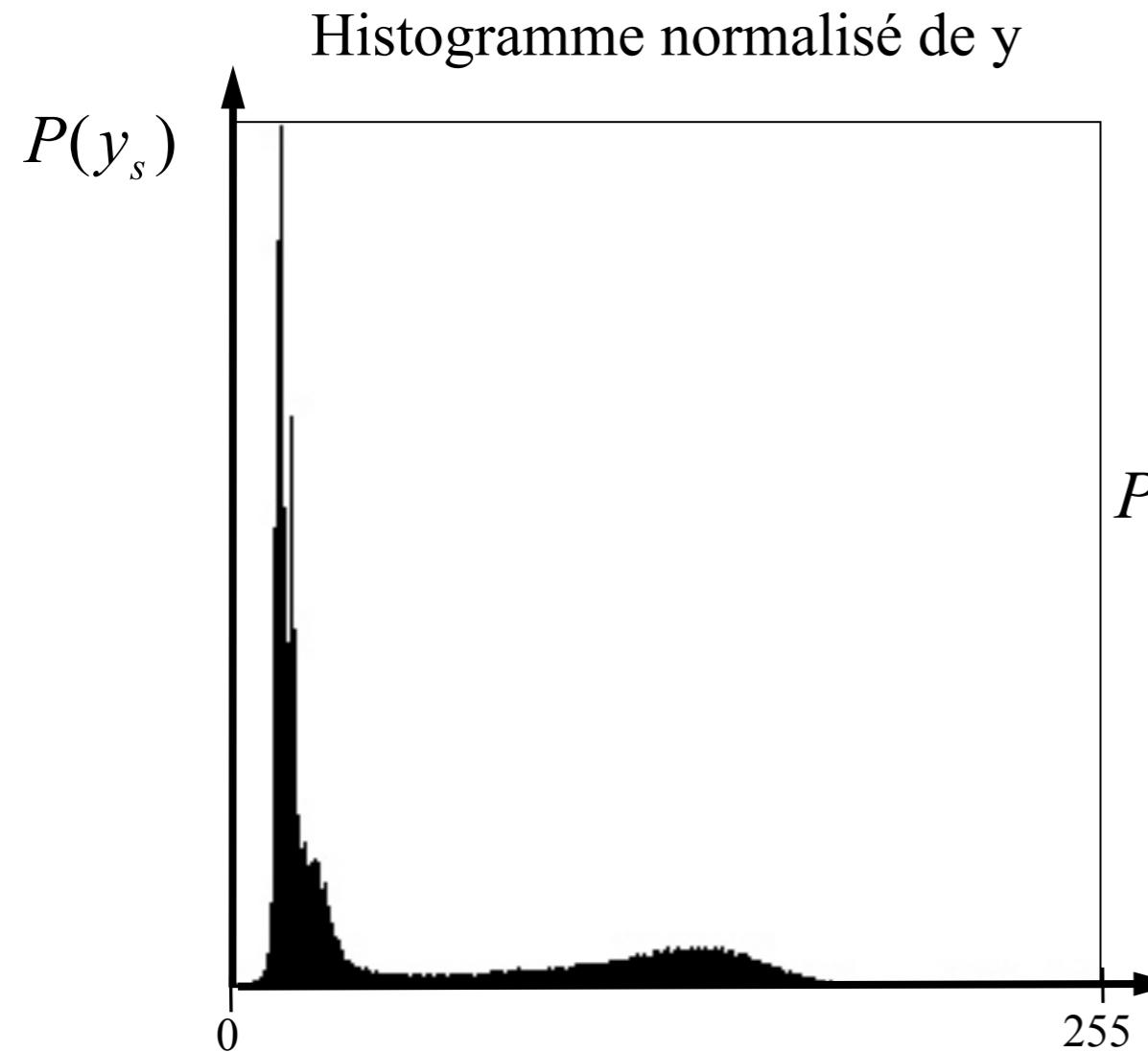


$P(y_s)$  Distribution des niveaux de gris dans l'image  $y$ .

$P(y_s = a)$  Peut se lire : probabilité d'observer un pixel de niveau de gris «  $a$  » dans l'image  $y$ .

# Trouver le *meilleur* seuil sans supervision

## Quelques définitions



$P(y_s)$  Distribution des niveaux de gris dans l'image  $y$ .

$P(y_s = a)$  Peut se lire : probabilité d'observer un pixel de niveau de gris «  $a$  » dans l'image  $y$ .

exemple: si  $P(y_s = 15) = 0.09$  alors

si je tire au hasard un pixel dans l'image  $y$ , j'aurai 9 pourcents de chance qu'il soit d'intensité 15.

# Trouver le *meilleur* seuil sans supervision

## Quelques définitions

$P(y_s = a)$  probabilité d'observer un pixel de niveau de gris a dans l'image  $y$

$P(y_s = a, mer)$  est une **probabilité jointe** qui se lit : la probabilité d'observer un pixel de niveau de gris a dans l'image  $y$  **ET** faisant partie de la classe mer.

$P(mer)$  probabilité d'observer un pixel appartenant à la classe *mer*.

# Trouver le *meilleur* seuil sans supervision

## Quelques définitions

$P(y_s = a)$  probabilité d'observer un pixel de niveau de gris  $a$  dans l'image  $y$

$P(y_s = a, mer)$  est une **probabilité jointe** qui se lit : la probabilité d'observer un pixel de niveau de gris  $a$  dans l'image  $y$  **ET** faisant partie de la classe mer.

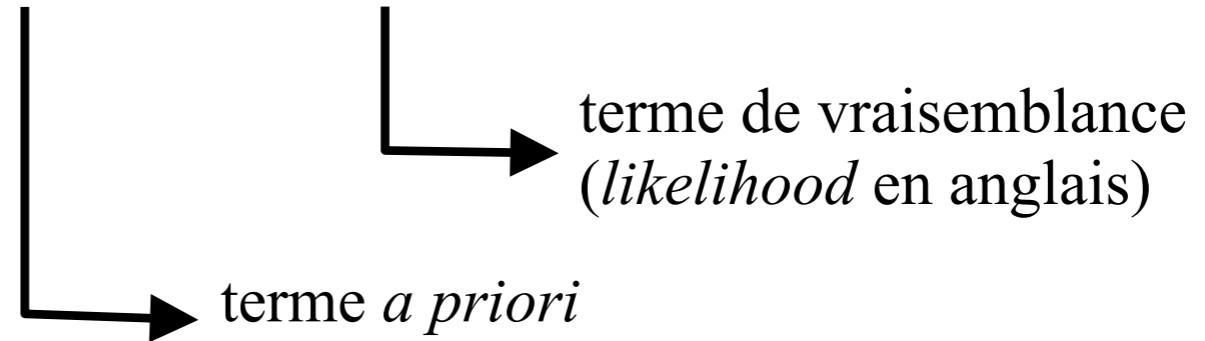
$P(mer)$  probabilité d'observer un pixel appartenant à la classe  $mer$ .

$$P(y_s = a, mer) = P(mer)P(y_s = a \mid mer)$$

$P(y_s = a \mid mer)$  est une **probabilité conditionnelle** qui se lit : la probabilité d'observer un pixel de niveau de gris  $a$  dans l'image  $y$  **ÉTANT DONNÉ** qu'il appartienne à la classe mer.

# Trouver le *meilleur* seuil sans supervision

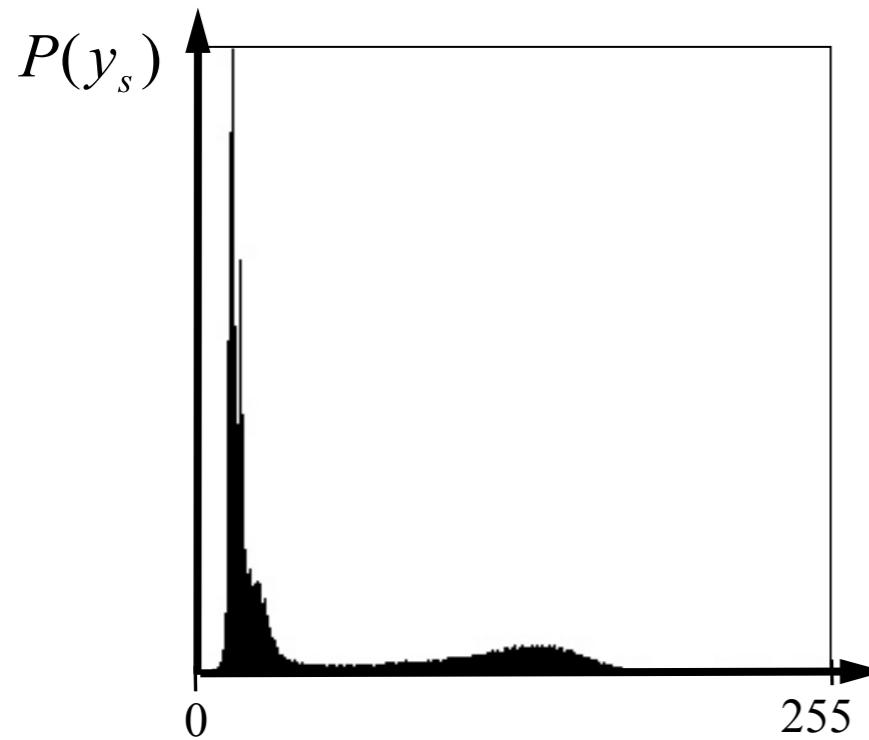
$$P(y_s, x_s) = P(x_s) \times P(y_s | x_s)$$



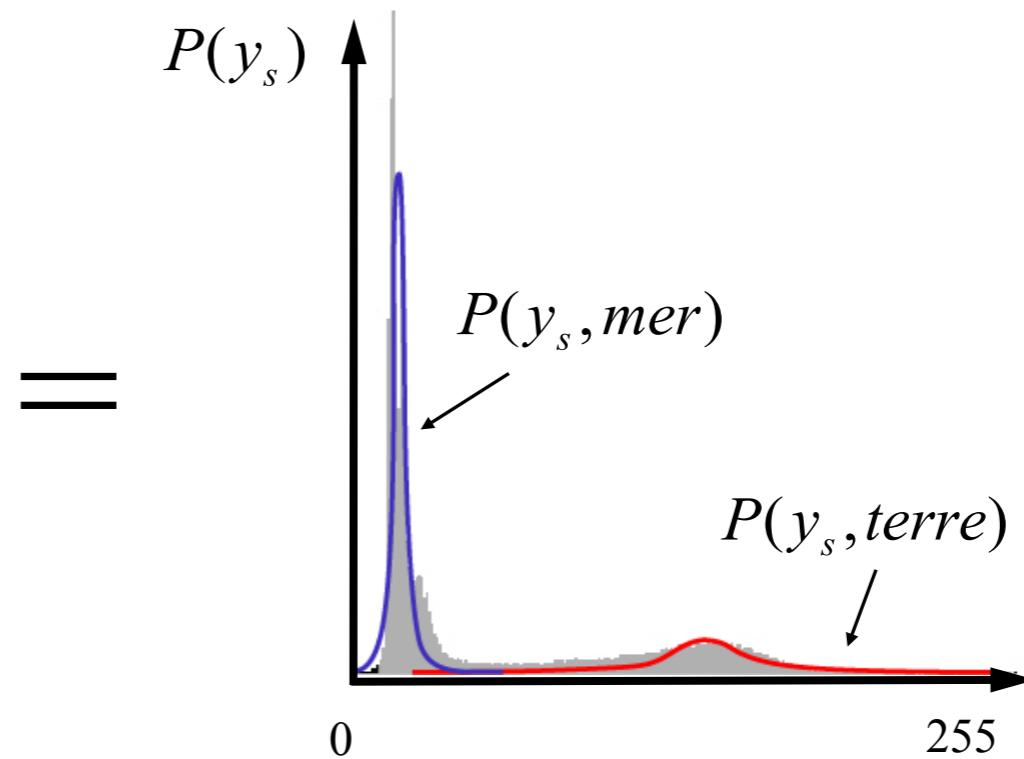
$$x_s \in \{\text{mer}, \text{terre}\}$$

# Trouver le *meilleur* seuil sans supervision

Histogramme normalisé de y



Histogramme normalisé de y



$P(y_s, terre)$

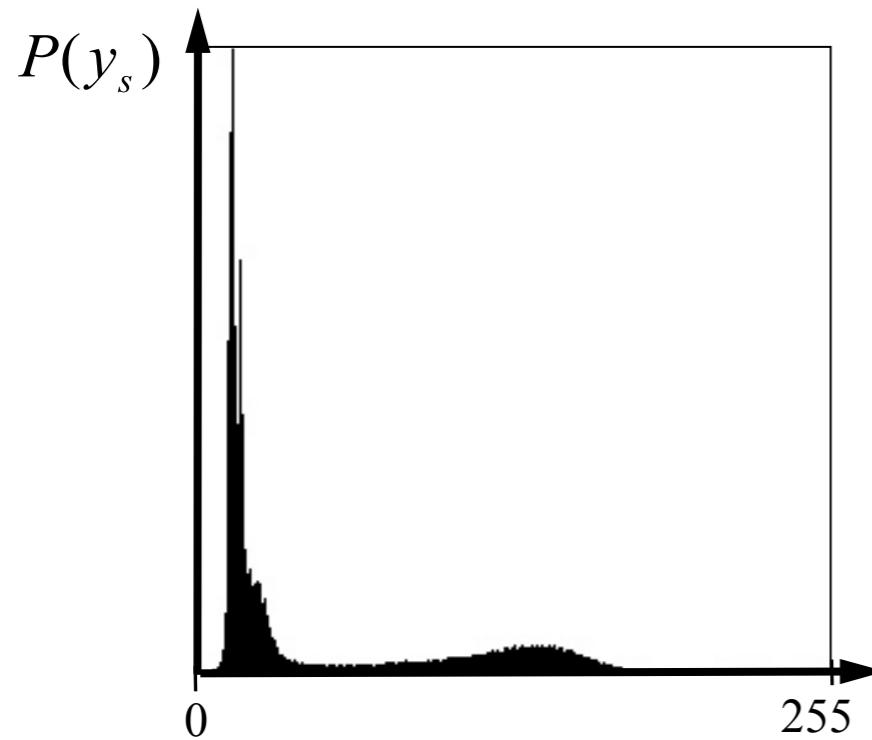
Distribution des niveaux de gris des pixels « terre »

$P(y_s, mer)$

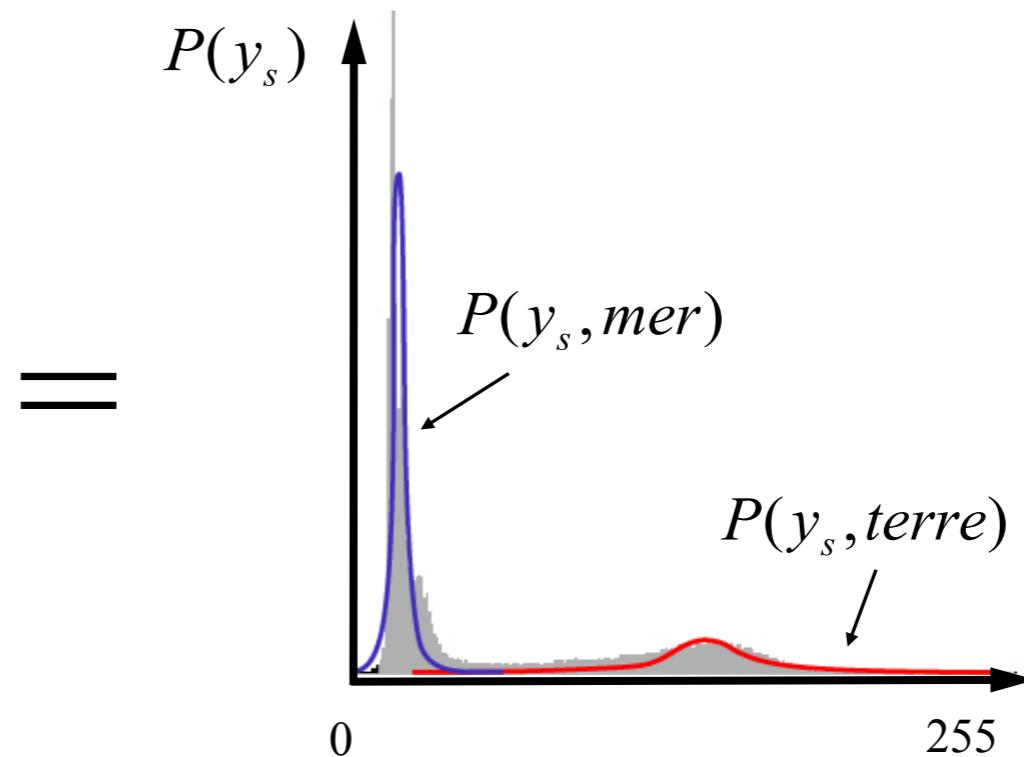
Distribution des niveaux de gris des pixels « mer »

# Trouver le *meilleur* seuil sans supervision

Histogramme normalisé de  $y$



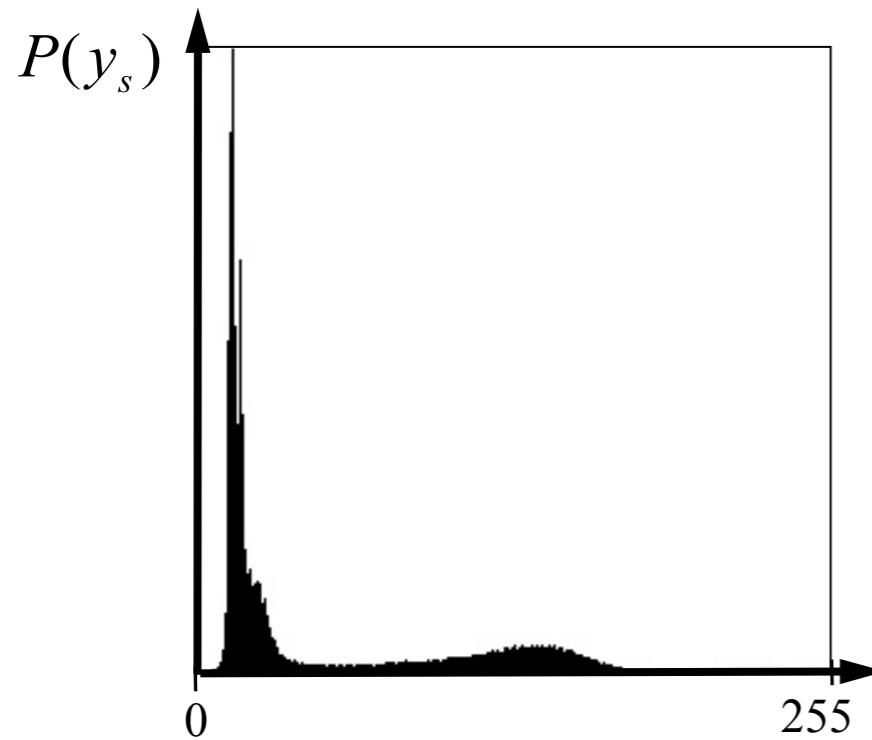
Histogramme normalisé de  $y$



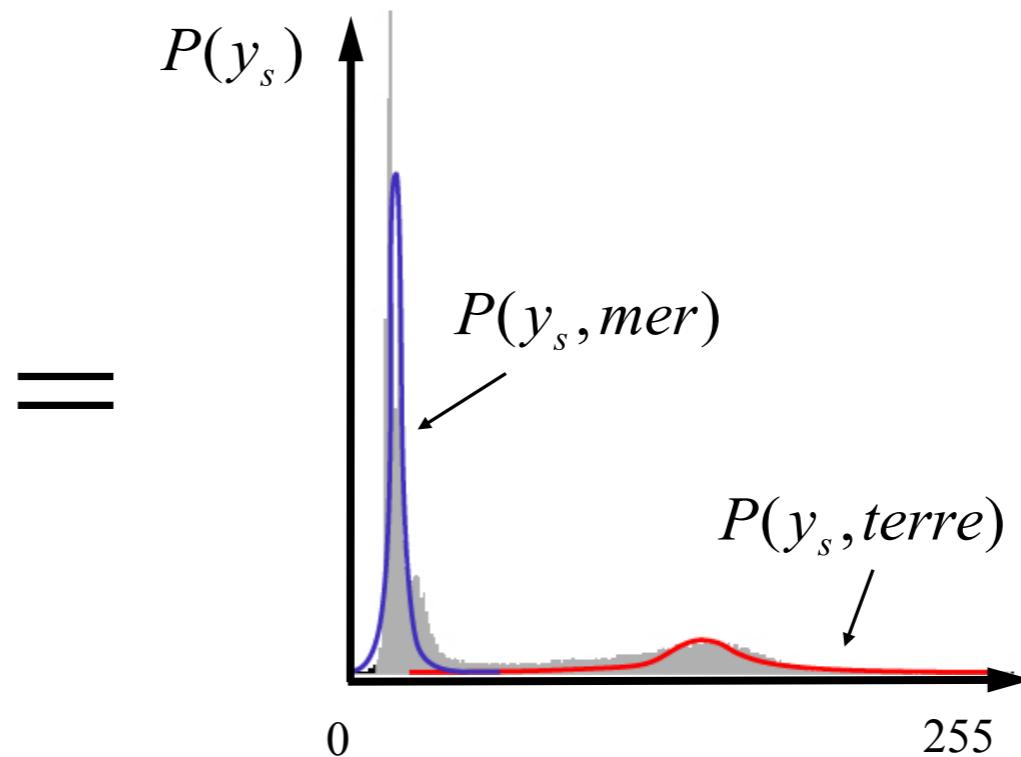
$$P(y_s) = P(y_s, mer) + P(y_s, terre)$$

# Trouver le *meilleur* seuil sans supervision

Histogramme normalisé de  $y$



Histogramme normalisé de  $y$



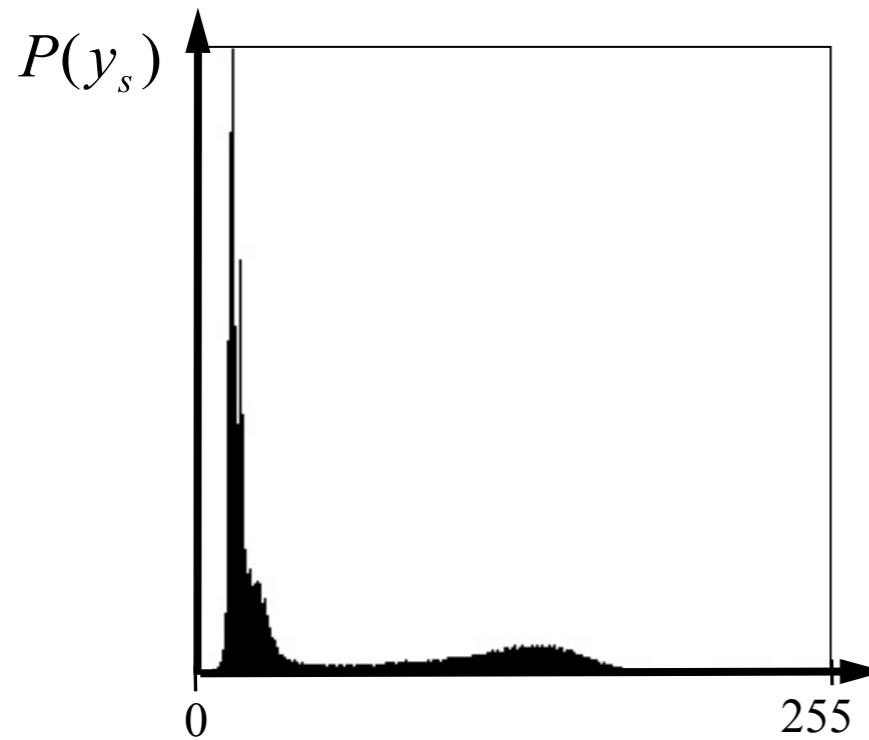
$$P(y_s) = P(y_s, mer) + P(y_s, terre)$$

Exemple

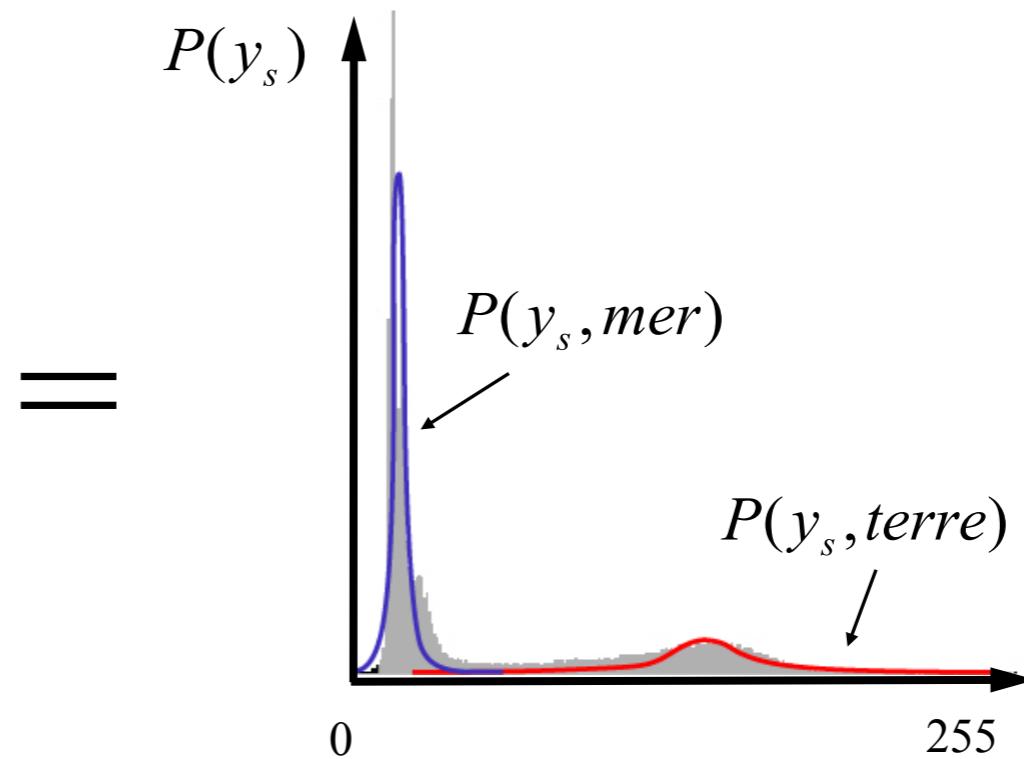
$P(y_s = 15, mer) = 0.08$  si je tire au hasard un pixel dans l'image  $y$ , j'ai 8 pourcents de chance qu'il soit d'intensité 15 **ET** qu'il appartienne à la classe **mer**

# Trouver le *meilleur* seuil sans supervision

Histogramme normalisé de  $y$



Histogramme normalisé de  $y$



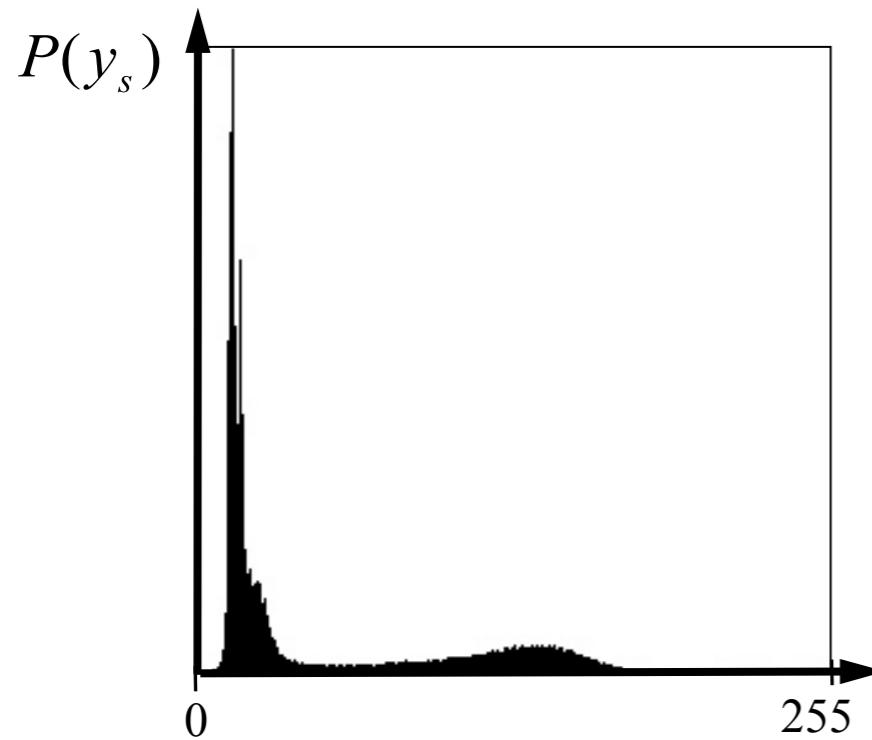
$$P(y_s) = P(y_s, mer) + P(y_s, terre)$$

Exemple

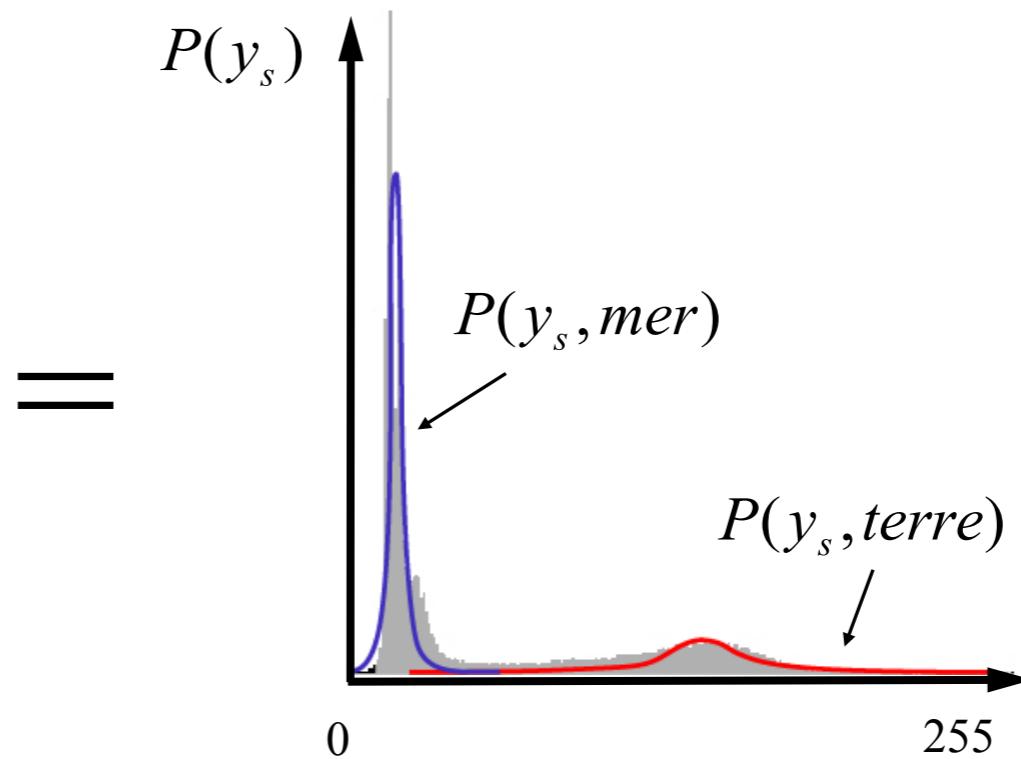
$P(y_s = 15, terre) = 0.01$  si je tire au hasard un pixel dans l'image  $y$ , j'ai 1 pourcent de chance qu'il soit d'intensité 15 **ET** qu'il appartienne à la classe **terre**

# Trouver le *meilleur* seuil sans supervision

Histogramme normalisé de  $y$



Histogramme normalisé de  $y$



$$P(y_s) = P(y_s, mer) + P(y_s, terre)$$

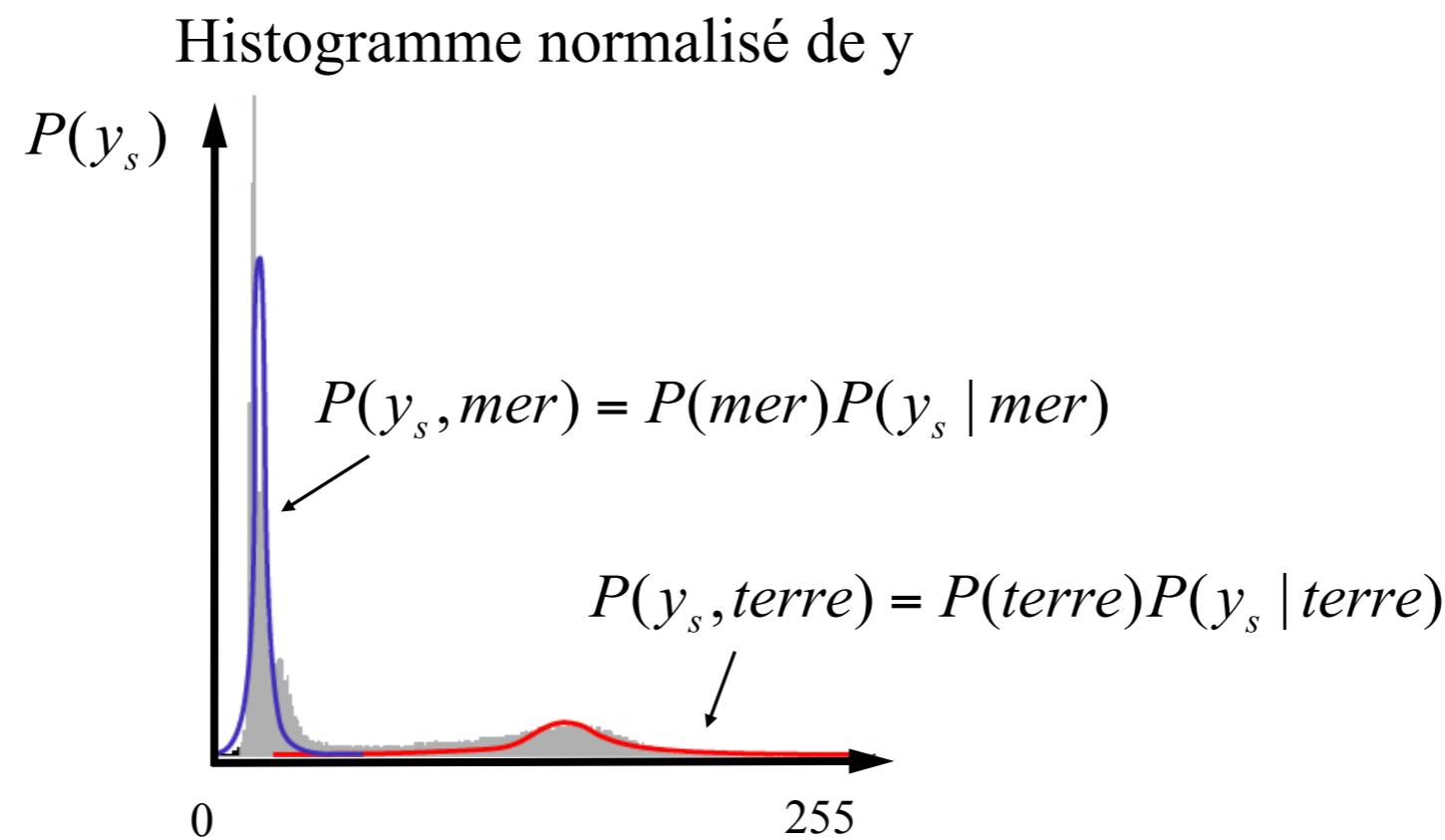
Exemple  $P(y_s = 15) = P(y_s = 15, mer) + P(y_s = 15, terre)$

$$0.09 = 0.08 + 0.01$$

# Trouver le *meilleur* seuil sans supervision

Une autre définition : **mixture de gaussiennes**

Si  $P(y_s | x_s)$  est une gaussienne  $N(y_s; \mu_{x_s}, \sigma_{x_s})$  alors



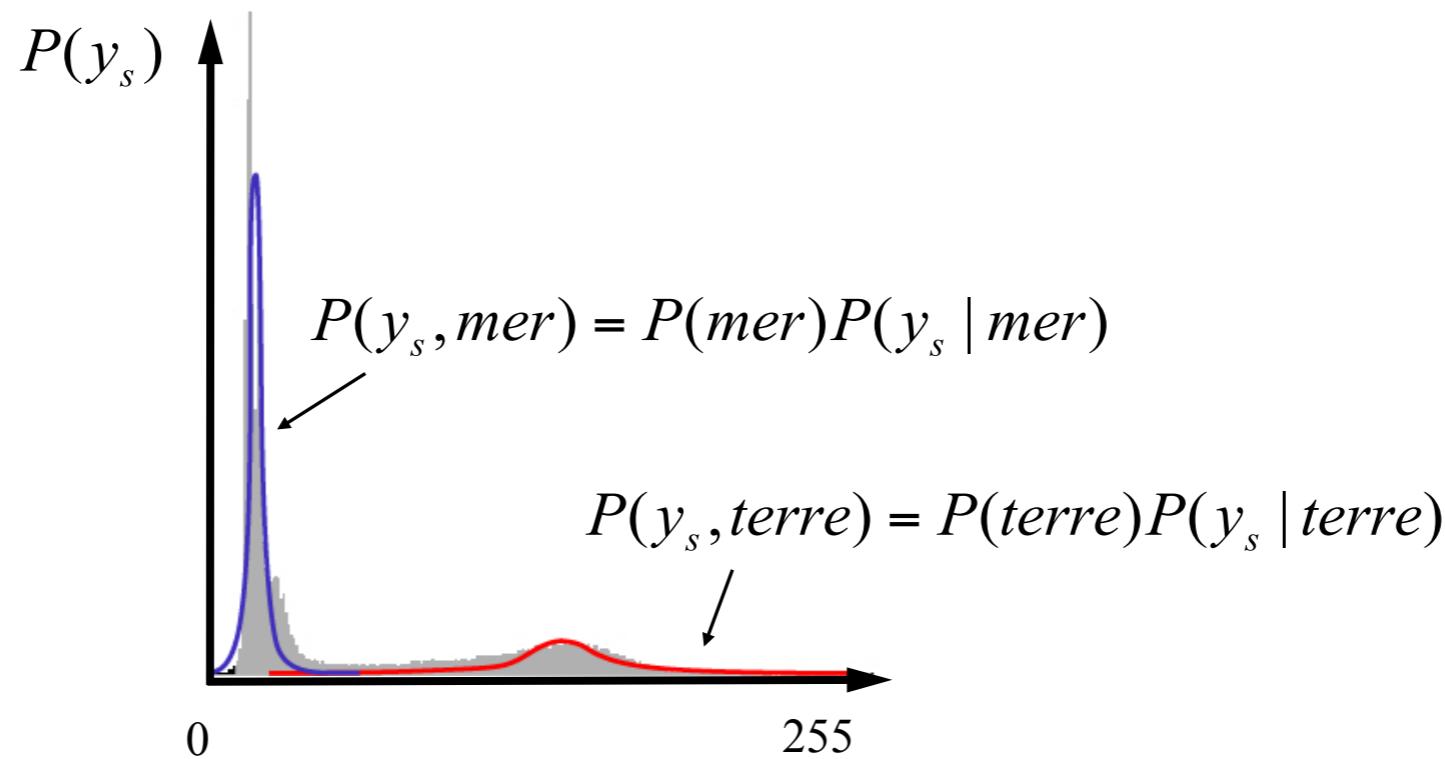
# Trouver le *meilleur* seuil sans supervision

Une autre définition : **mixture de gaussiennes**

Si  $P(y_s | x_s)$  est une gaussienne  $N(y_s; \mu_{x_s}, \sigma_{x_s})$  alors

$$\begin{aligned} P(y_s) &= P(y_s, \text{mer}) + P(y_s, \text{terre}) \\ &= P(\text{mer})P(y_s | \text{mer}) + P(\text{terre})P(y_s | \text{terre}) \\ &= P(\text{mer})N(y_s; \mu_{\text{mer}}, \sigma_{\text{mer}}) + P(\text{terre})N(y_s; \mu_{\text{terre}}, \sigma_{\text{terre}}) \\ &= \sum_{x_s \in \{\text{terre, mer}\}} P(x_s)N(y_s; \mu_{x_s}, \sigma_{x_s}) \end{aligned}$$

Histogramme normalisé de y



# Trouver le *meilleur* seuil sans supervision

Une autre définition : **mixture**

De façon plus générale, une mixture s'exprime mathématiquement de la façon suivante:

$$P(y_s) = \sum_c P(c)P(y_s | c)$$

Si  $P(y_s | c)$  est une gaussienne, alors on parlera d'une **mixture de gaussiennes**

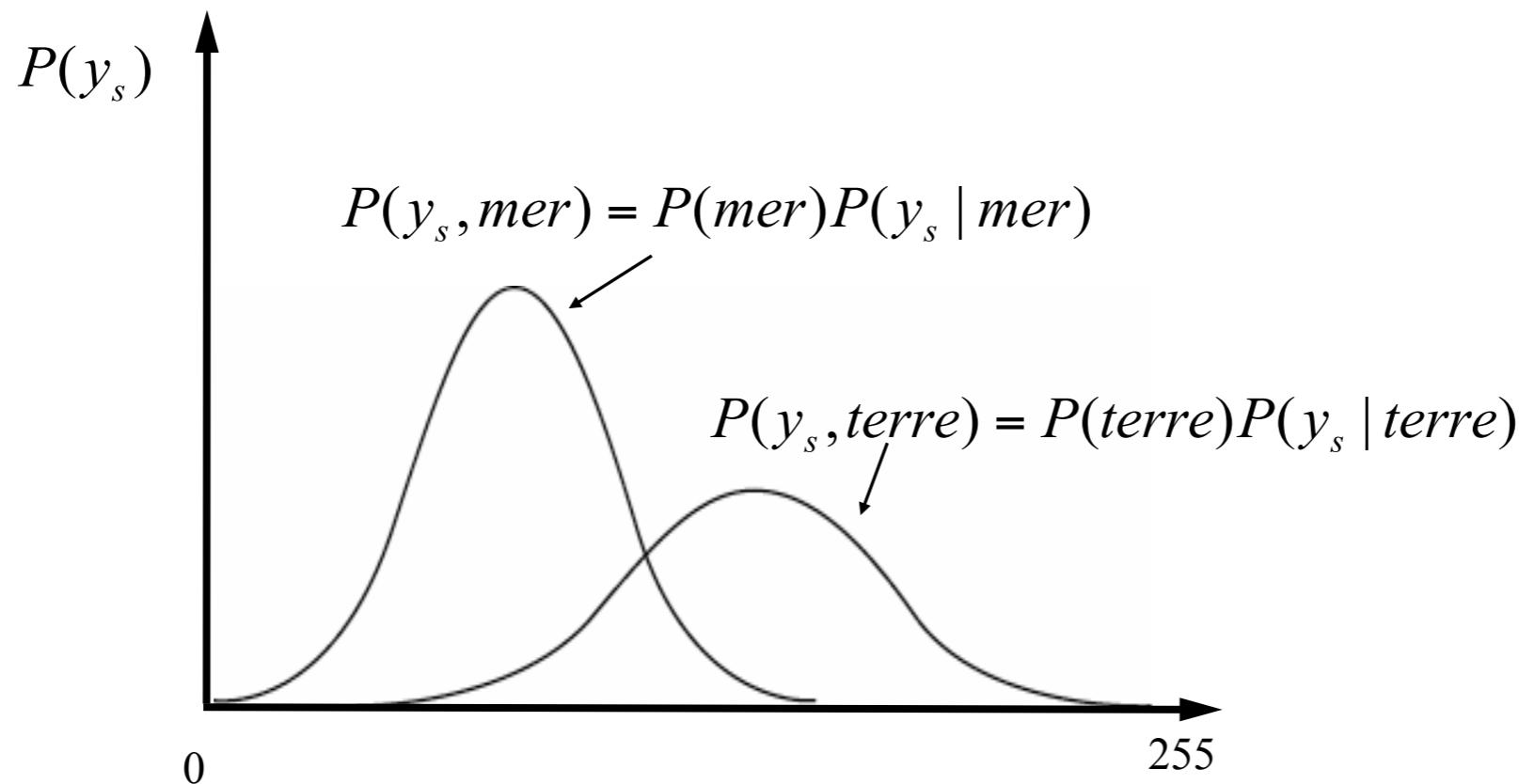
# Trouver le *meilleur* seuil sans supervision

Étant donné une mixture de gaussiennes dont on connaît les paramètres

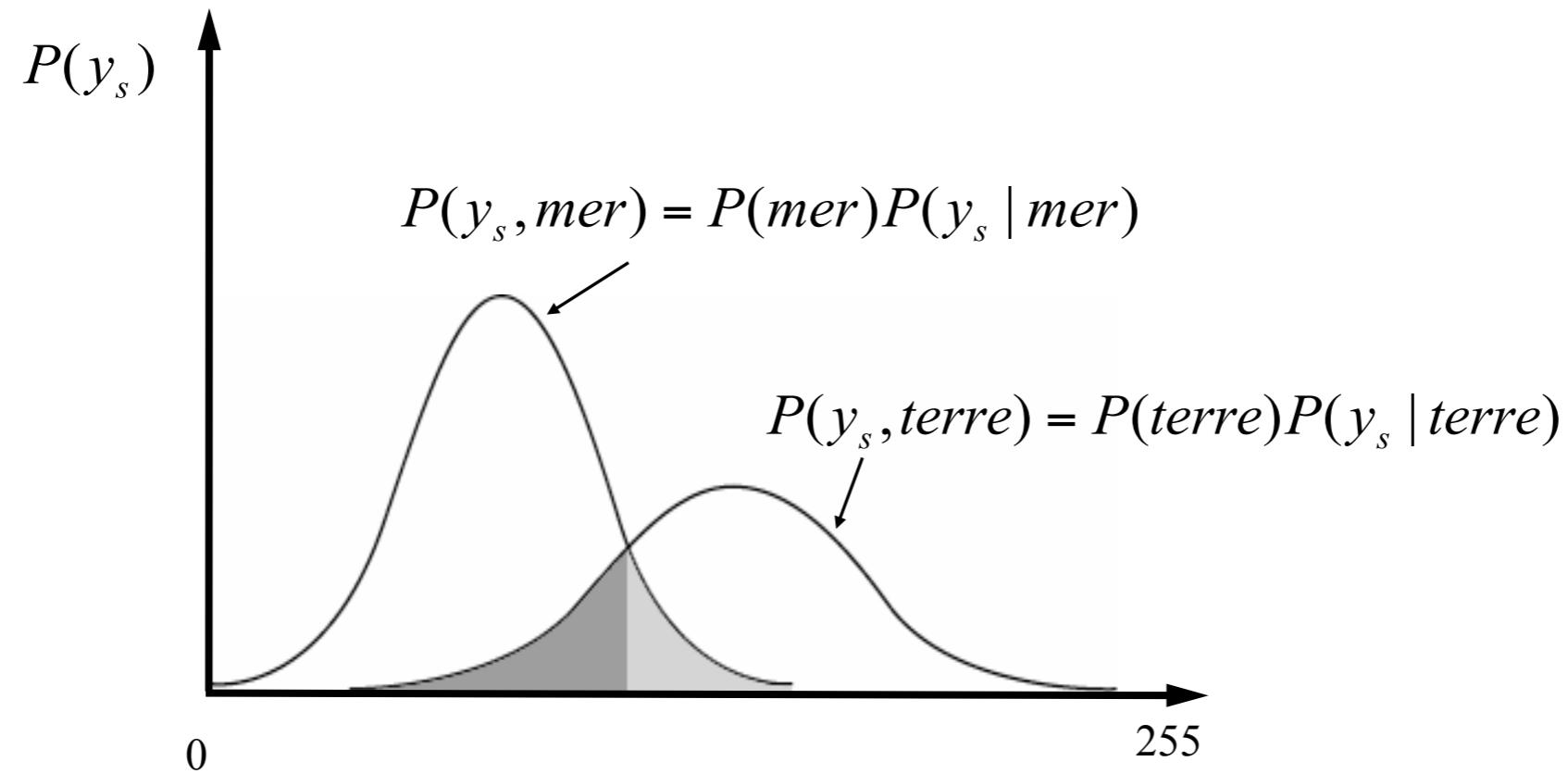
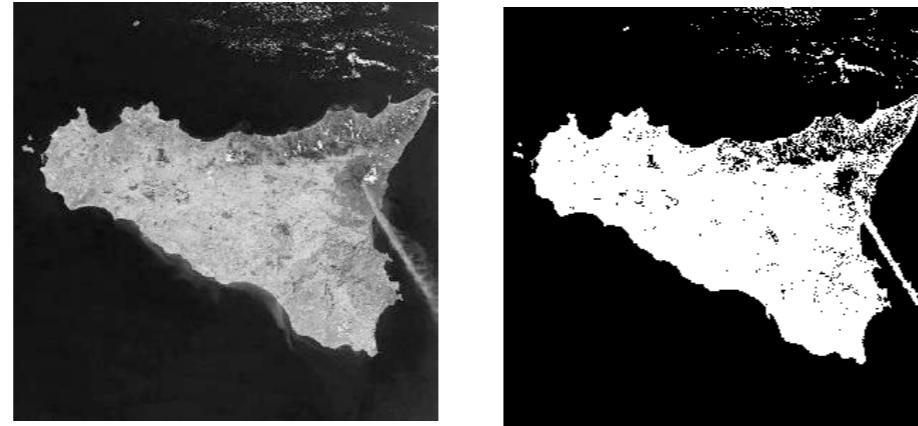
$$\{(P(\text{mer}), \mu_{\text{mer}}, \sigma_{\text{mer}}), (P(\text{terre}), \mu_{\text{terre}}, \sigma_{\text{terre}})\}$$

comment trouver le « meilleur » seuil  $T$ ?

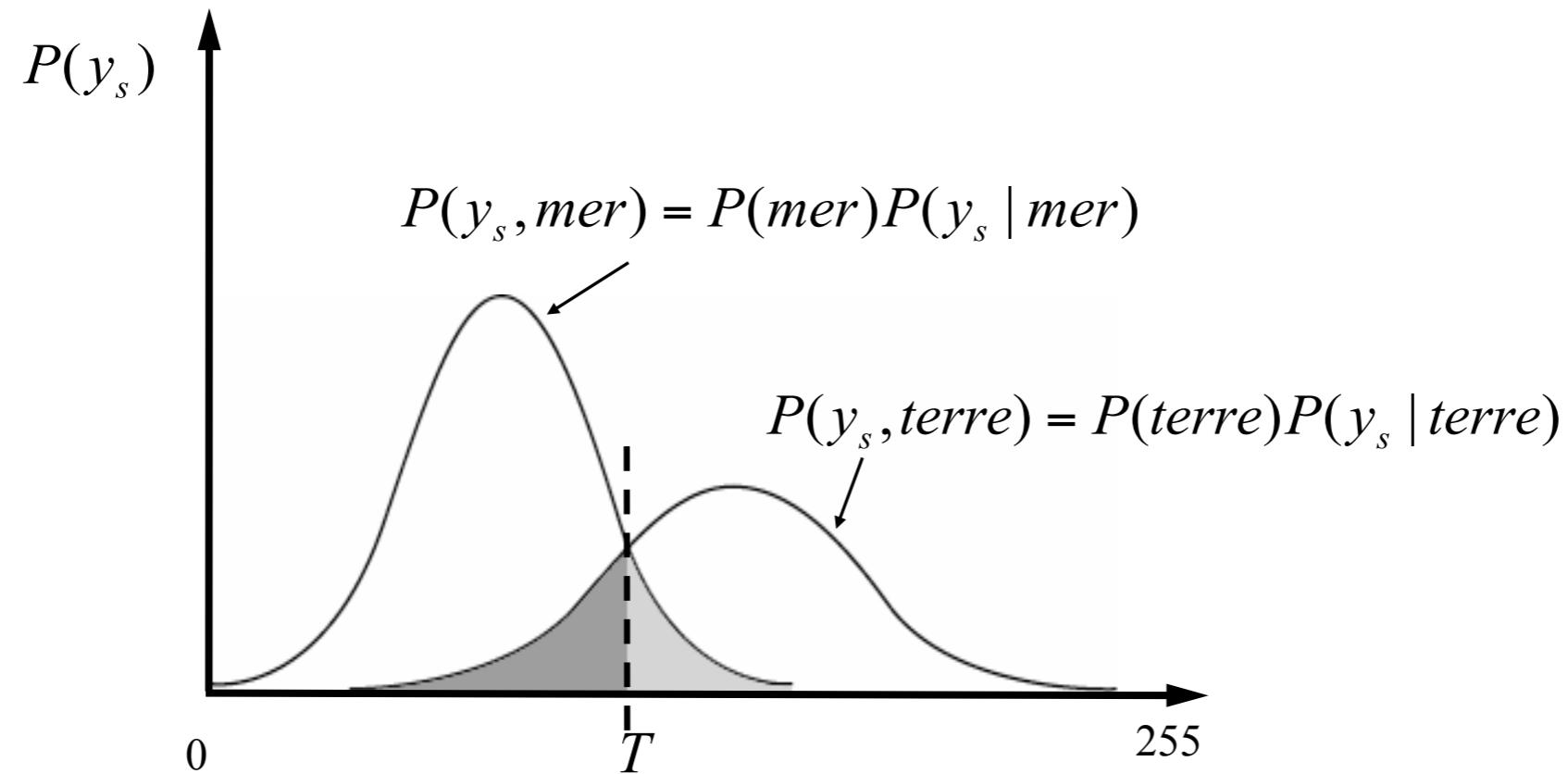
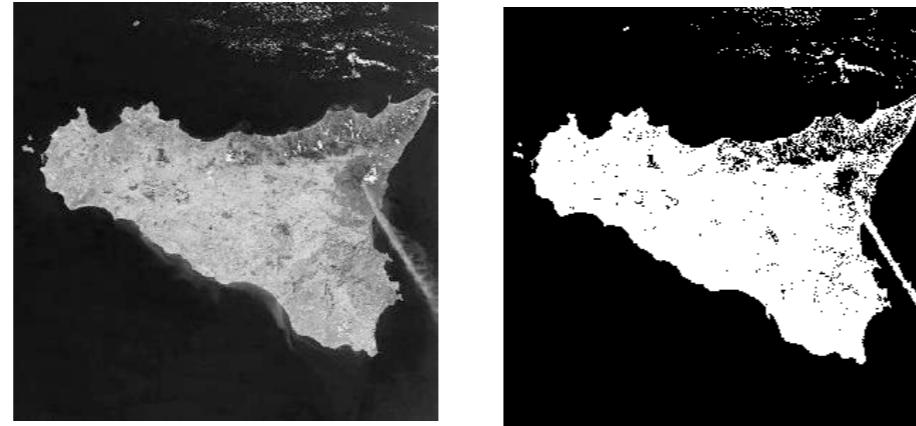
Pour y arriver on doit quantifier l'erreur de classification d'un seuil  $T$ . Le seuil retenu sera celui associé à la plus petite erreur.



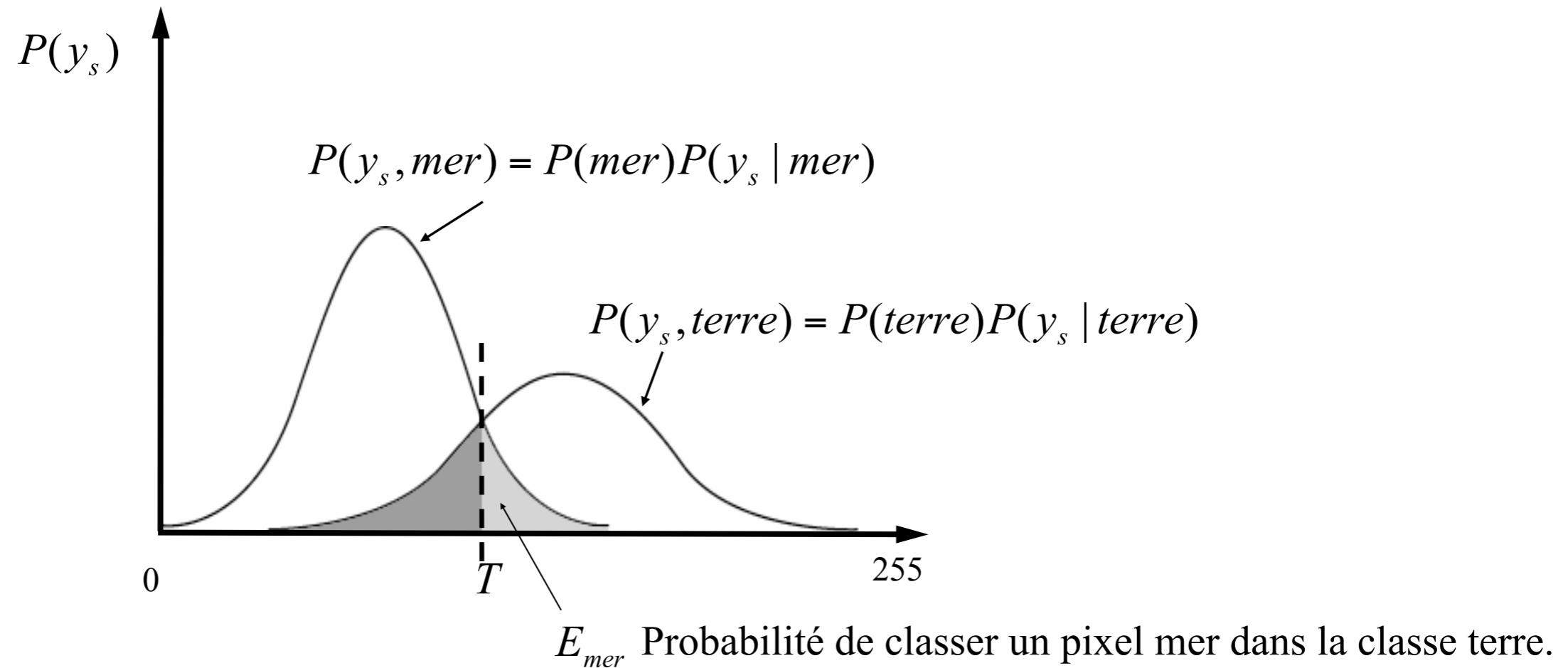
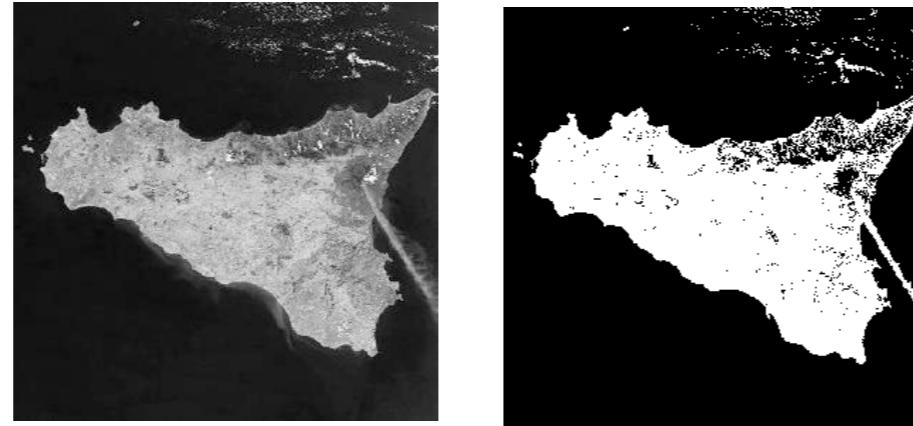
# Trouver le *meilleur* seuil sans supervision



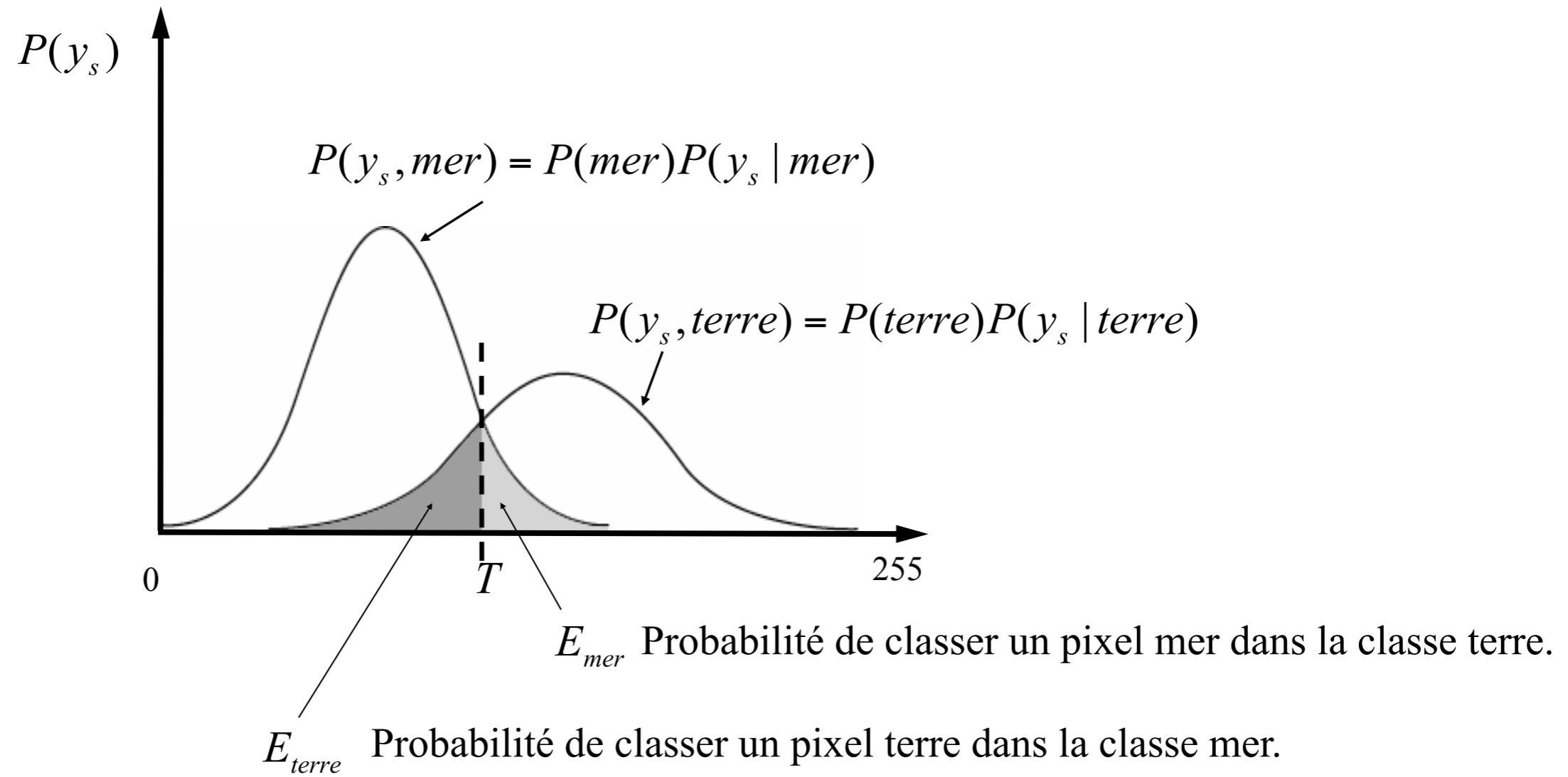
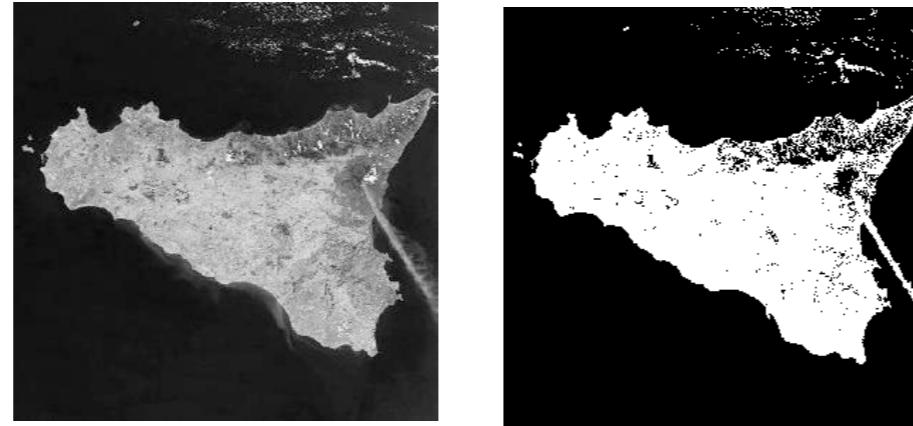
# Trouver le *meilleur* seuil sans supervision



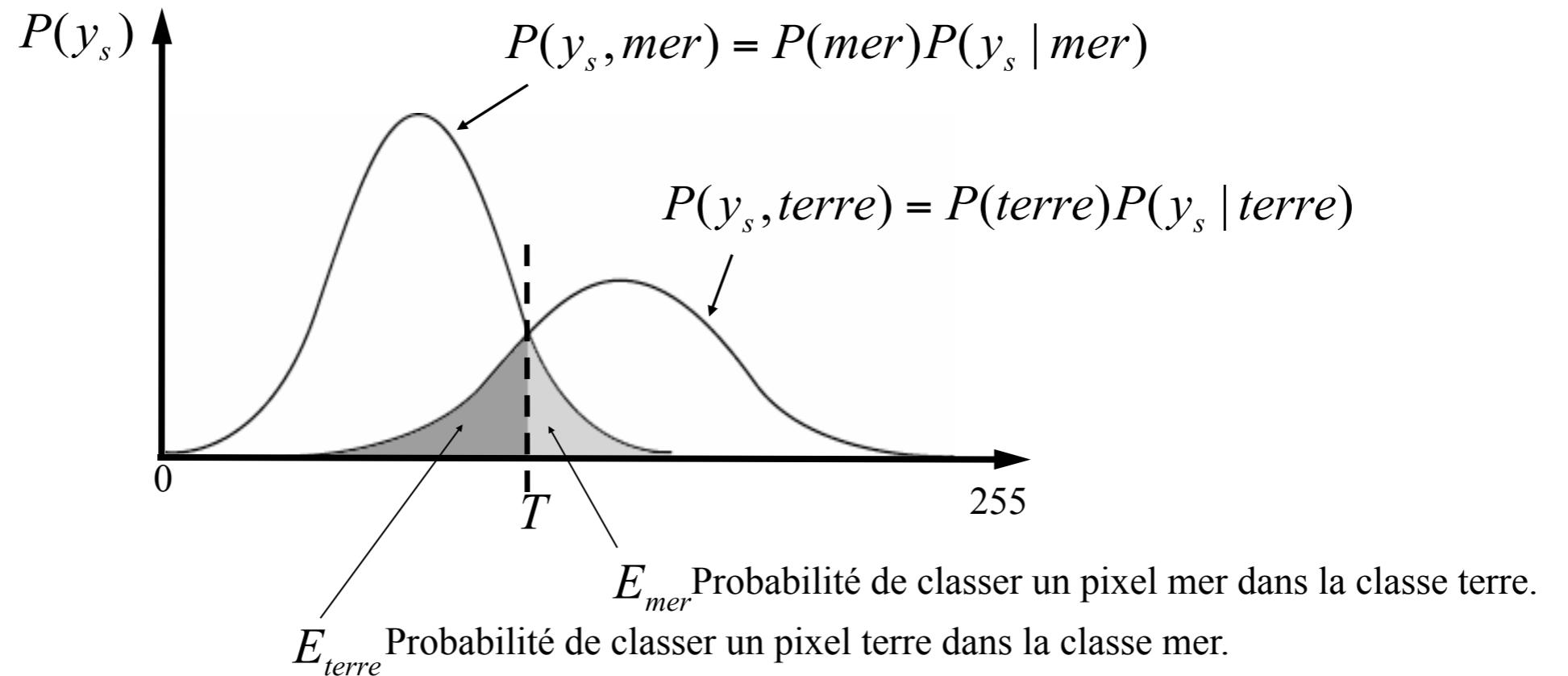
# Trouver le *meilleur* seuil sans supervision



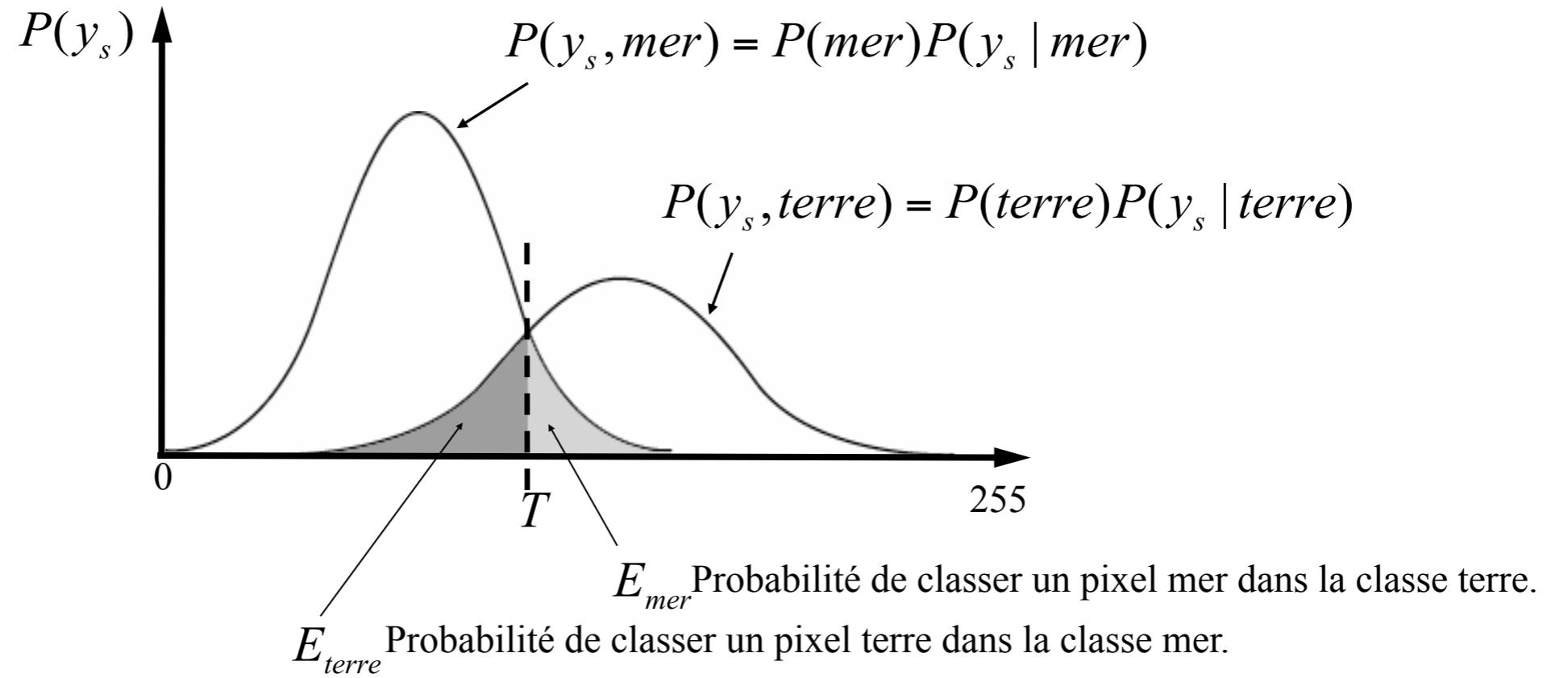
# Trouver le *meilleur* seuil sans supervision



# Trouver le *meilleur* seuil sans supervision



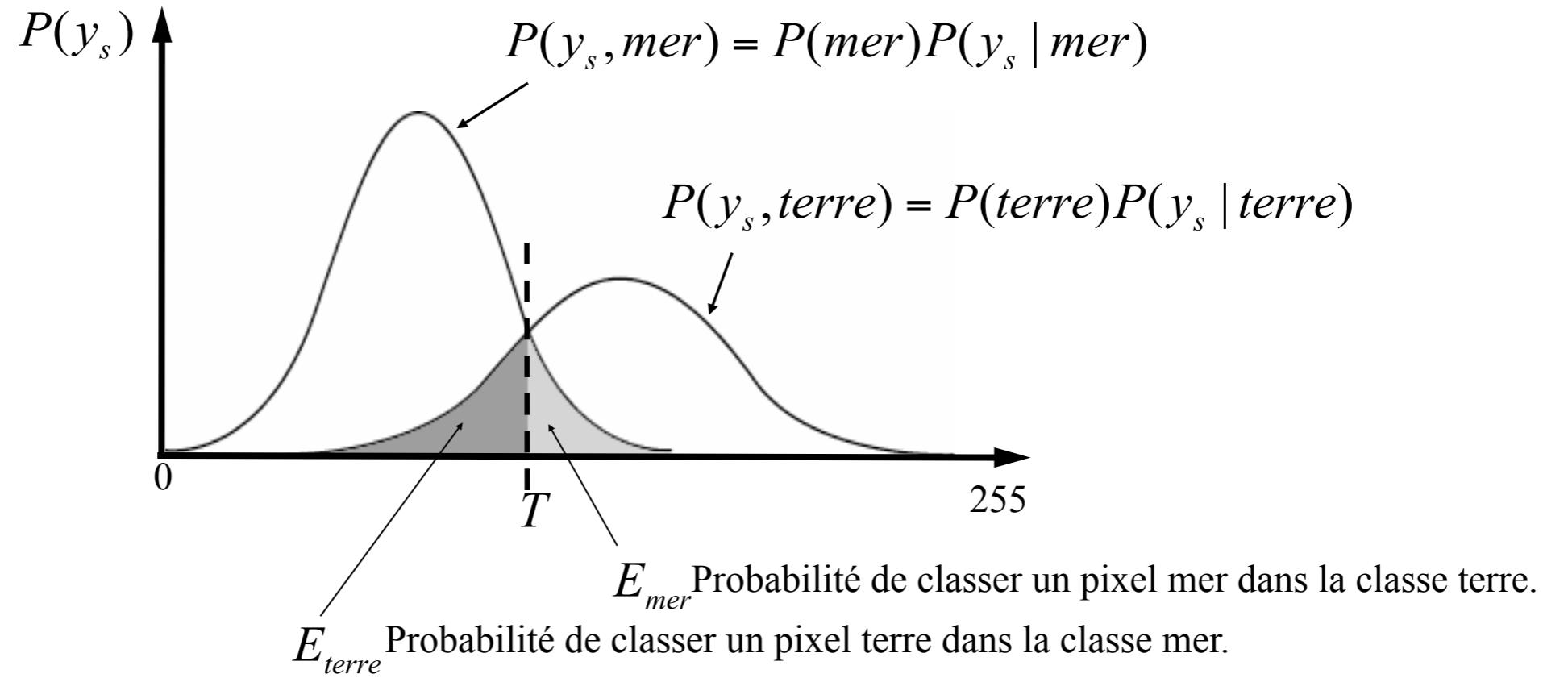
# Trouver le *meilleur* seuil sans supervision



$$E_{terre}(T) = \int_{-\infty}^T P(y_s, terre) dy$$

$$E_{mer}(T) = \int_T^{\infty} P(y_s, mer) dy$$

# Trouver le *meilleur* seuil sans supervision



$$E_{terre}(T) = \int_{-\infty}^T P(y_s, terre) dy$$

$$E_{mer}(T) = \int_T^{\infty} P(y_s, mer) dy$$

$$E_{totale}(T) = E_{terre}(T) + E_{mer}(T)$$

# Trouver le *meilleur* seuil sans supervision

Le meilleur seuil « T » est celui pour lequel l'erreur totale est minimale, c-à-d pour lequel :

$$\frac{dE_{total}(T)}{dT} = 0$$

# Trouver le *meilleur* seuil sans supervision

Le meilleur seuil « T » est celui pour lequel l'erreur totale est minimale, c-à-d pour lequel :

$$\frac{dE_{total}(T)}{dT} = 0$$

À l'aide de la règle de Leibniz, on obtient que

$$P(mer)P(y_s = T \mid mer) = P(terre)P(y_s = T \mid terre)$$

# Trouver le *meilleur* seuil sans supervision

Le meilleur seuil « T » est celui pour lequel l'erreur totale est minimale, c-à-d pour lequel :

$$\frac{dE_{total}(T)}{dT} = 0$$

À l'aide de la règle de Leibniz, on obtient que

$$P(\text{mer})P(y_s = T \mid \text{mer}) = P(\text{terre})P(y_s = T \mid \text{terre})$$

Ce qui revient au même d'effectuer le teste suivant

$$\frac{P(y_s \mid \text{mer})}{P(y_s \mid \text{terre})} \stackrel{\text{mer}}{\gtrless} \tau \frac{P(\text{terre})}{P(\text{mer})}$$

si  $\tau > 1$  alors on favorise la classe terre

si  $0 < \tau < 1$  alors on favorise la classe mer

# Trouver le *meilleur* seuil sans supervision

$$\frac{P(y_s \mid mer)}{P(y_s \mid terre)} \stackrel{\text{mer}}{\gtrless} \tau \frac{P(terre)}{P(mer)}$$

On appelle parfois ce type d'opération : **test d'hypothèses**

et le membre de gauche porte le nom de ***likelihood ratio***

# Trouver le *meilleur* seuil sans supervision

$$\frac{P(y_s | mer)}{P(y_s | terre)} \stackrel{\text{mer}}{\gtrless} \tau \frac{P(terre)}{P(mer)}$$

Très souvent, on suppose que la vraisemblance de chaque classe se distribue suivant une gaussienne:

$$P(y_s | mer) = \frac{1}{\sqrt{2\pi}\sigma_{mer}} \exp\left(-\frac{(y_s - \mu_{mer})^2}{2\sigma_{mer}^2}\right)$$

$$P(y_s | terre) = \frac{1}{\sqrt{2\pi}\sigma_{terre}} \exp\left(-\frac{(y_s - \mu_{terre})^2}{2\sigma_{terre}^2}\right)$$

où

$\mu_{mer}$  : intensité moyenne des pixels appartenant à la classe *mer*.

$\sigma_{mer}$  : écart - type de l'intensité des pixels appartenant à la classe *mer*.

# L'algorithme du seuil « optimal »

## Algorithme du seuil « optimal »

1. POUR CHAQUE site  $s$  du champ d'observations  $y$  FAIRE

$$P_m = \frac{P(\text{mer})}{\sqrt{2\pi}\sigma_{\text{mer}}} \exp\left(-\frac{(y_s - \mu_{\text{mer}})^2}{2\sigma_{\text{mer}}^2}\right)$$

$$P_t = \tau \frac{P(\text{terre})}{\sqrt{2\pi}\sigma_{\text{terre}}} \exp\left(-\frac{(y_s - \mu_{\text{terre}})^2}{2\sigma_{\text{terre}}^2}\right)$$

SI  $P_t > P_m$  ALORS

$x_s = 1$  /\* Étiquette « terre » au pixel (i,j) \*/

SINON

$x_s = 0$  /\* Étiquette « mer » au pixel (i,j) \*/

Note : nous verrons plus tard comment estimer  $P(\text{mer}), P(\text{terre}), \mu_{\text{mer}}, \mu_{\text{terre}}, \sigma_{\text{mer}}, \sigma_{\text{terre}}$

# Trouver le *meilleur* seuil sans supervision

Afin de réduire les temps de calculs, il est possible de précalculer le "meilleur" seuil T

$$P(\text{mer})P(y_s = T \mid \text{mer}) = P(\text{terre})P(y_s = T \mid \text{terre})$$
$$\frac{P(\text{mer})}{\sqrt{2\pi}\sigma_{\text{mer}}} \exp\left(-\frac{(T - \mu_{\text{mer}})^2}{2\sigma_{\text{mer}}^2}\right) = \frac{P(\text{terre})}{\sqrt{2\pi}\sigma_{\text{terre}}} \exp\left(-\frac{(T - \mu_{\text{terre}})^2}{2\sigma_{\text{terre}}^2}\right)$$

# Trouver le *meilleur* seuil sans supervision

Afin de réduire les temps de calculs, il est possible de précalculer le "meilleur" seuil T

$$P(\text{mer})P(y_s = T \mid \text{mer}) = P(\text{terre})P(y_s = T \mid \text{terre})$$
$$\frac{P(\text{mer})}{\sqrt{2\pi}\sigma_{\text{mer}}} \exp\left(-\frac{(T - \mu_{\text{mer}})^2}{2\sigma_{\text{mer}}^2}\right) = \frac{P(\text{terre})}{\sqrt{2\pi}\sigma_{\text{terre}}} \exp\left(-\frac{(T - \mu_{\text{terre}})^2}{2\sigma_{\text{terre}}^2}\right)$$

On peut facilement démontrer que

$$AT^2 + BT + C = 0$$

$$A = \sigma_{\text{mer}}^2 - \sigma_{\text{terre}}^2$$

$$B = 2(\mu_{\text{mer}}\sigma_{\text{terre}}^2 - \mu_{\text{terre}}\sigma_{\text{mer}}^2)$$

$$C = \mu_{\text{terre}}^2\sigma_{\text{mer}}^2 - \mu_{\text{mer}}^2\sigma_{\text{terre}}^2 + 2\sigma_{\text{mer}}^2\sigma_{\text{terre}}^2 \ln\left(\frac{\sigma_{\text{terre}}P(\text{mer})}{\sigma_{\text{mer}}P(\text{terre})}\right)$$

}

2 seuils = les 2 racines du polynôme.

# Trouver le *meilleur* seuil sans supervision

Afin de réduire les temps de calculs, il est possible de précalculer le "meilleur" seuil T

$$P(\text{mer})P(y_s = T \mid \text{mer}) = P(\text{terre})P(y_s = T \mid \text{terre})$$

$$\frac{P(\text{mer})}{\sqrt{2\pi}\sigma_{\text{mer}}} \exp\left(-\frac{(T - \mu_{\text{mer}})^2}{2\sigma_{\text{mer}}^2}\right) = \frac{P(\text{terre})}{\sqrt{2\pi}\sigma_{\text{terre}}} \exp\left(-\frac{(T - \mu_{\text{terre}})^2}{2\sigma_{\text{terre}}^2}\right)$$

On peut facilement démontrer que

$$AT^2 + BT + C = 0$$

$$A = \sigma_{\text{mer}}^2 - \sigma_{\text{terre}}^2$$

$$B = 2(\mu_{\text{mer}}\sigma_{\text{terre}}^2 - \mu_{\text{terre}}\sigma_{\text{mer}}^2)$$

$$C = \mu_{\text{terre}}^2\sigma_{\text{mer}}^2 - \mu_{\text{mer}}^2\sigma_{\text{terre}}^2 + 2\sigma_{\text{mer}}^2\sigma_{\text{terre}}^2 \ln\left(\frac{\sigma_{\text{terre}}P(\text{mer})}{\sigma_{\text{mer}}P(\text{terre})}\right)$$

Si  $\sigma_{\text{mer}} = \sigma_{\text{terre}} = \sigma$  alors

$$T = \frac{\mu_{\text{mer}} + \mu_{\text{terre}}}{2} + \frac{\sigma^2}{\mu_{\text{terre}} - \mu_{\text{mer}}} \ln\left(\frac{P(\text{mer})}{P(\text{terre})}\right)$$

2 seuils = les 2 racines du polynôme.

# L'algorithme du seuil « optimal »

## Algorithme du seuil optimal

calculer T

1. POUR CHAQUE site  $s$  du champ d'observations  $y$  FAIRE

SI  $y_s > T$  ALORS

$x_s = 1$  /\* Étiquette « *terre* » au pixel (i,j) \*/

SINON

$x_s = 0$  /\* Étiquette « *mer* » au pixel (i,j) \*/

# L'algorithme du seuil « optimal »

## Algorithme du seuil optimal

calculer T

1. POUR CHAQUE site  $s$  du champ d'observations  $y$  FAIRE

SI  $y_s > T$  ALORS

$x_s = 1$  /\* Étiquette « *terre* » au pixel (i,j) \*/

SINON

$x_s = 0$  /\* Étiquette « *mer* » au pixel (i,j) \*/

# Segmentation

L'estimation de paramètres robustes  
méthodes hybrides (locale/globale)

# Algorithme des K-moyennes (*K-means*)

Un des problèmes avec les algorithmes de classification « probabilistes » vus précédemment est qu'ils exigent de connaître les paramètres (moyenne et écart-type pour une gaussienne) de chaque classe avant de segmenter l'image d'entrée.

# Algorithme des K-moyennes (*K-means*)

Un des problèmes avec les algorithmes de classification « probabilistes » vus précédemment est qu'ils exigent de connaître les paramètres (moyenne et écart-type pour une gaussienne) de chaque classe avant de segmenter l'image d'entrée.

Ce sont des algorithmes dits supervisés. Ils ne sont pas autonomes et exigent que l'utilisateur fournisse des données fondamentales pour le traitement.

# Algorithme des K-moyennes (*K-means*)

Un des problèmes avec les algorithmes de classification « probabilistes » vus précédemment est qu'ils exigent de connaître les paramètres (moyenne et écart-type pour une gaussienne) de chaque classe avant de segmenter l'image d'entrée.

Ce sont des algorithmes dits supervisés. Ils ne sont pas autonomes et exigent que l'utilisateur fournisse des données fondamentales pour le traitement.

**Question:** comment segmenter automatiquement une image à l'aide d'une distribution probabiliste (ici une mixture de gaussiennes)? En d'autres mots, comment estimer SANS SUPERVISION les paramètres de la mixture de gaussienne:

# Algorithme des K-moyennes (*K-means*)

Un des problèmes avec les algorithmes de classification « probabilistes » vus précédemment est qu'ils exigent de connaître les paramètres (moyenne et écart-type pour une gaussienne) de chaque classe avant de segmenter l'image d'entrée.

Ce sont des algorithmes dits supervisés. Ils ne sont pas autonomes et exigent que l'utilisateur fournisse des données fondamentales pour le traitement.

**Question:** comment segmenter automatiquement une image à l'aide d'une distribution probabiliste (ici une mixture de gaussiennes)? En d'autres mots, comment estimer SANS SUPERVISION les paramètres de la mixture de gaussienne:

# Algorithme des K-moyennes (*K-means*)

Un des problèmes avec les algorithmes de classification « probabilistes » vus précédemment est qu'ils exigent de connaître les paramètres (moyenne et écart-type pour une gaussienne) de chaque classe avant de segmenter l'image d'entrée.

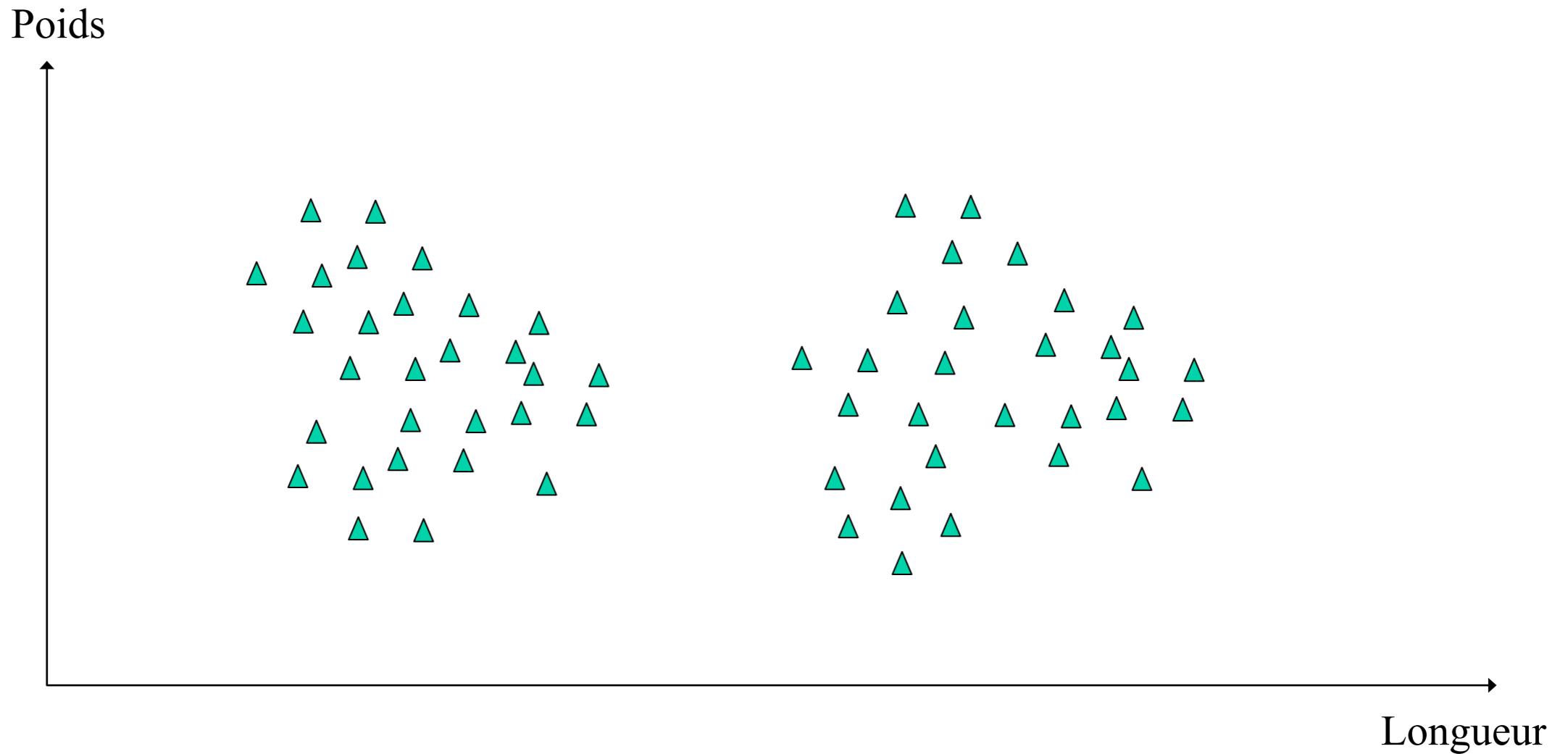
Ce sont des algorithmes dits **supervisés**. Ils ne sont pas autonomes et exigent que l'utilisateur fournisse des données fondamentales pour le traitement.

**Question:** comment segmenter automatiquement une image à l'aide d'une distribution probabiliste (ici une mixture de gaussiennes)? En d'autres mots, comment estimer SANS SUPERVISION les paramètres de la mixture de gaussienne:

$$\mu_{x_s}, \sigma_{x_s} \text{ et } P(x_s) ?$$

# Algorithme des K-moyennes (*K-means*)

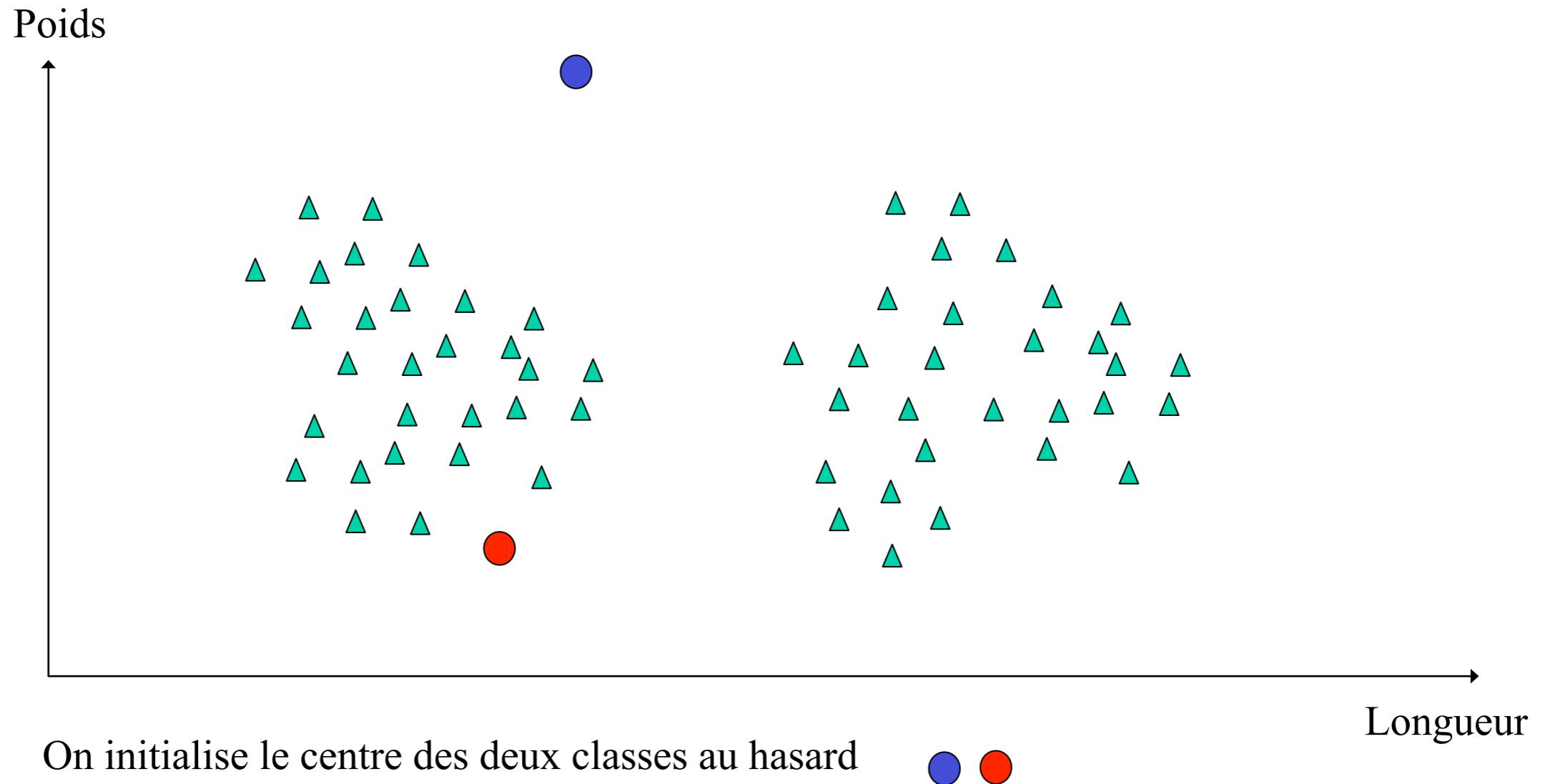
Exemple schématique de l'algorithme des K-moyennes



Voici 60 observations. Comment faire pour grouper ces données en deux classes et estimer la moyenne (ou le centre de masse) de ces classes ?

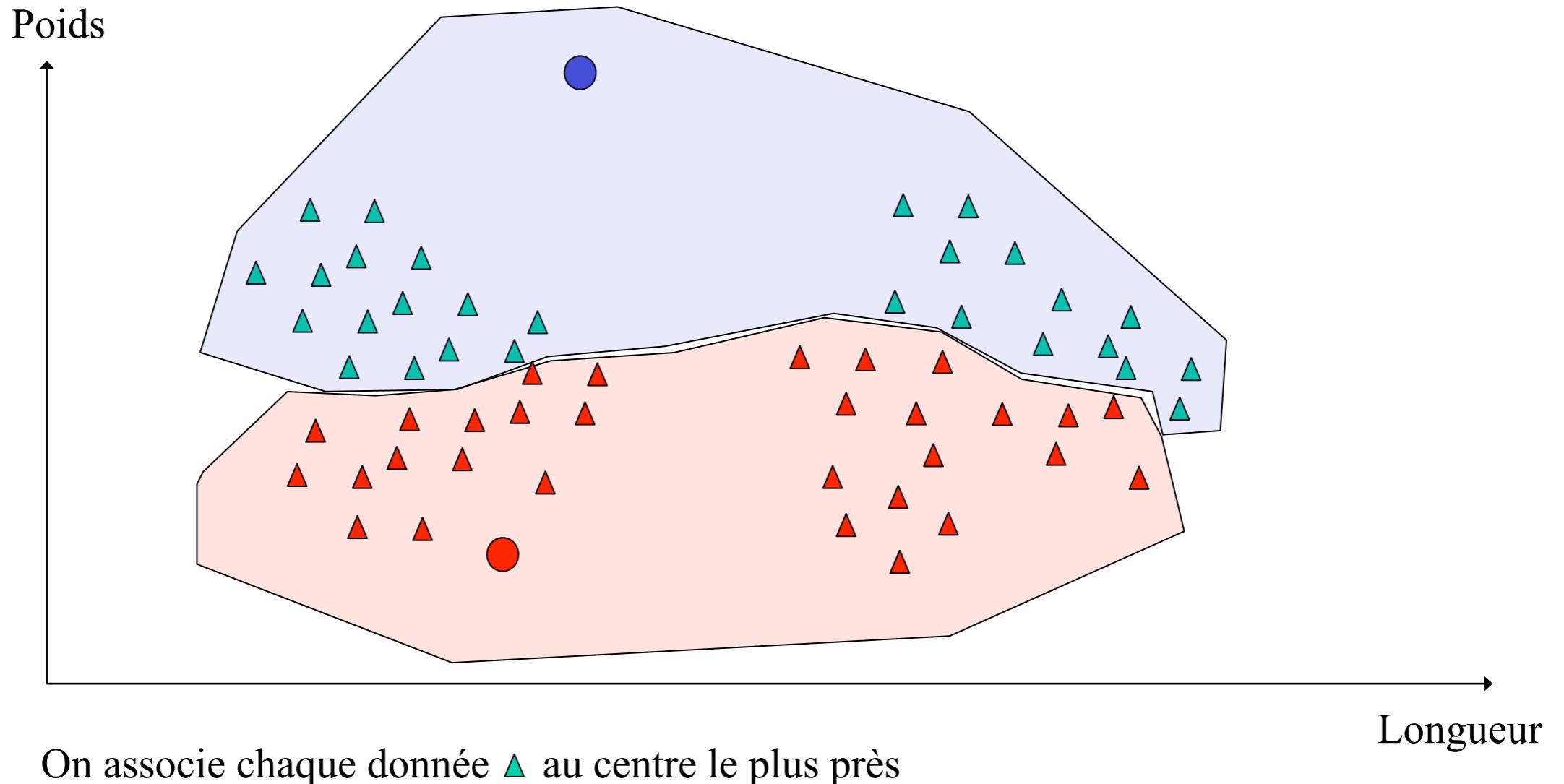
# Algorithme des K-moyennes (*K-means*)

Exemple schématique de l'algorithme des K-moyennes



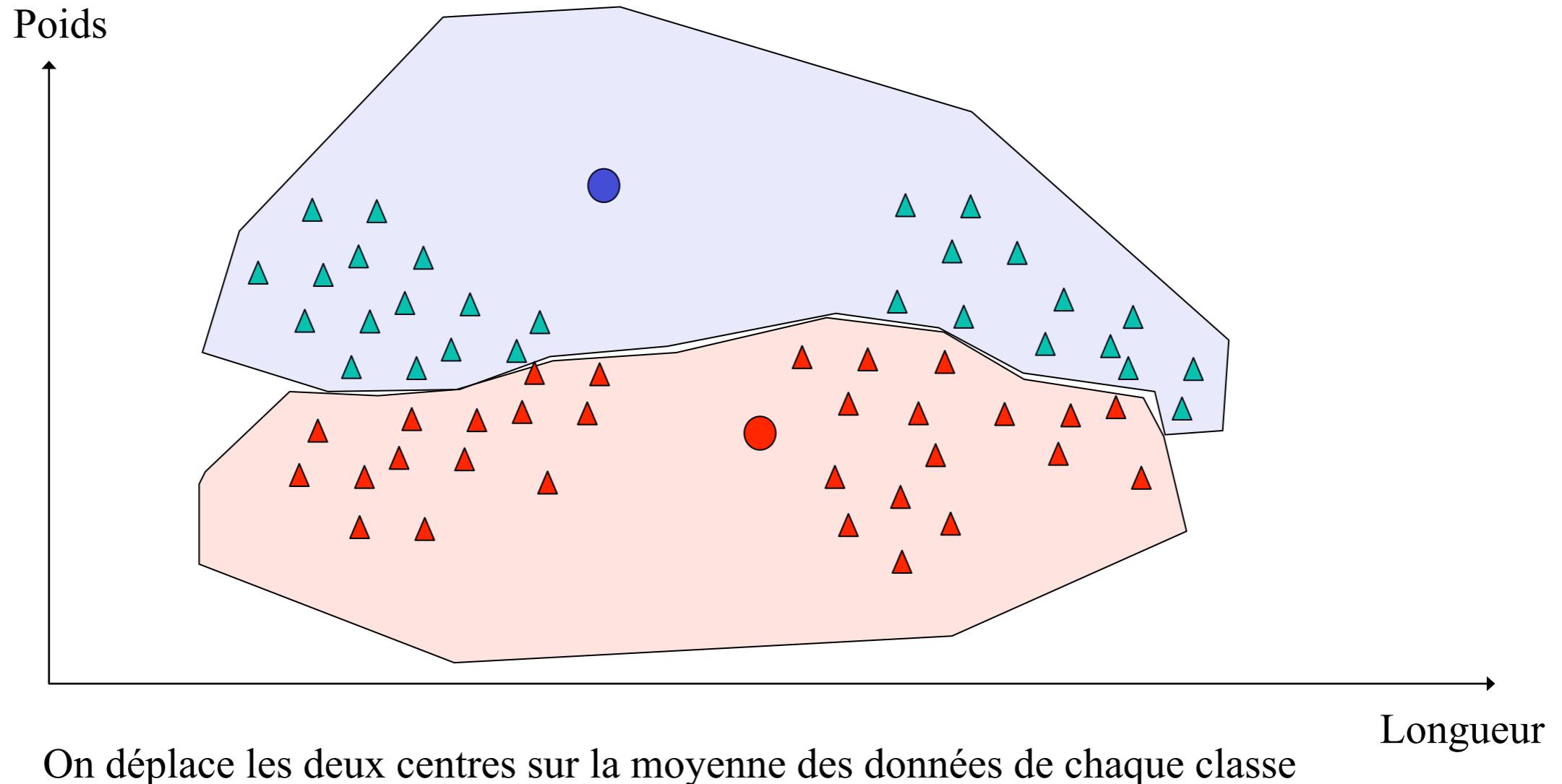
# Algorithme des K-moyennes (*K-means*)

Exemple schématique de l'algorithme des K-moyennes



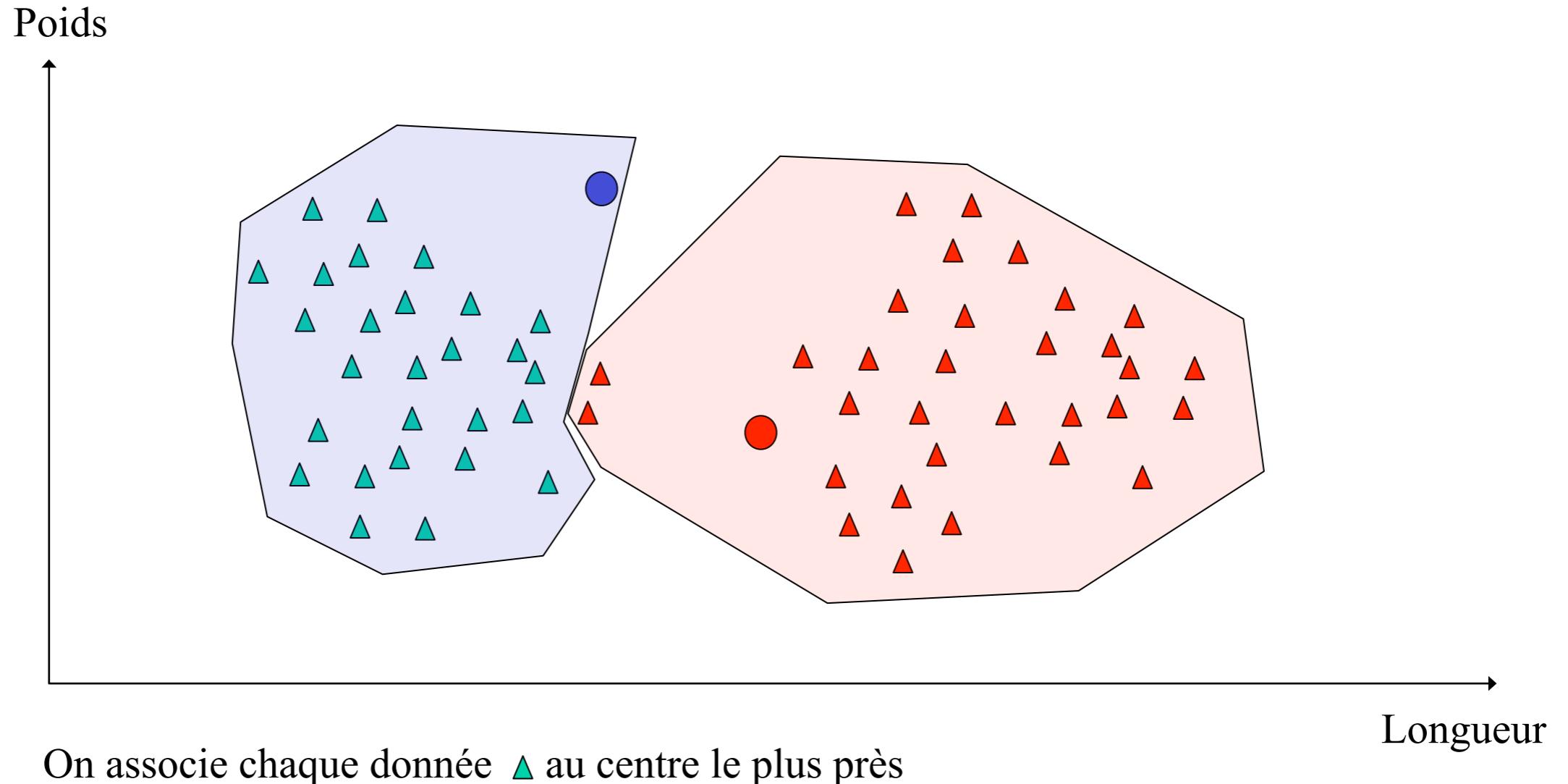
# Algorithme des K-moyennes (*K-means*)

Exemple schématique de l'algorithme des K-moyennes



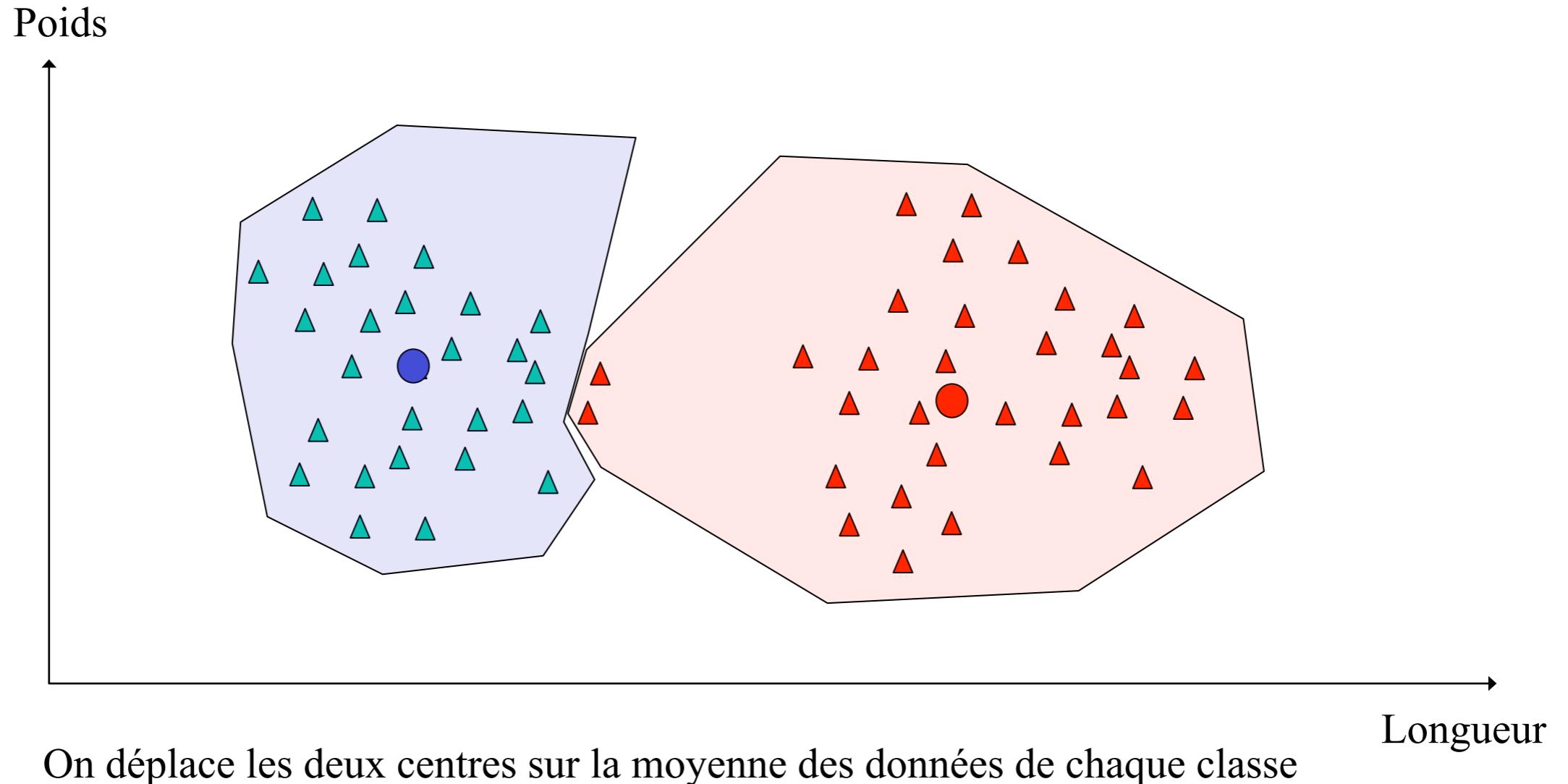
# Algorithme des K-moyennes (*K-means*)

Exemple schématique de l'algorithme des K-moyennes



# Algorithme des K-moyennes (*K-means*)

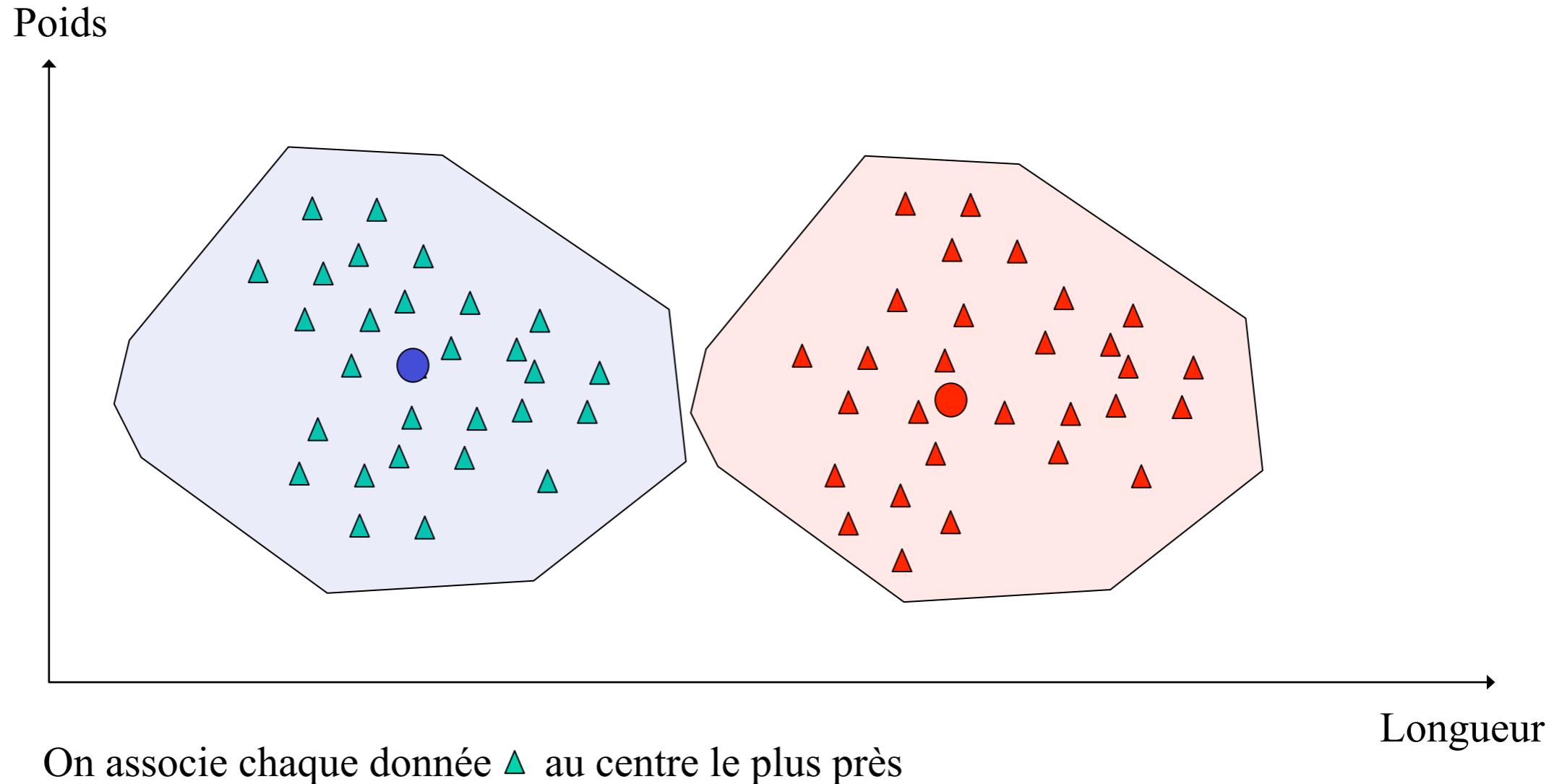
Exemple schématique de l'algorithme des K-moyennes



On déplace les deux centres sur la moyenne des données de chaque classe

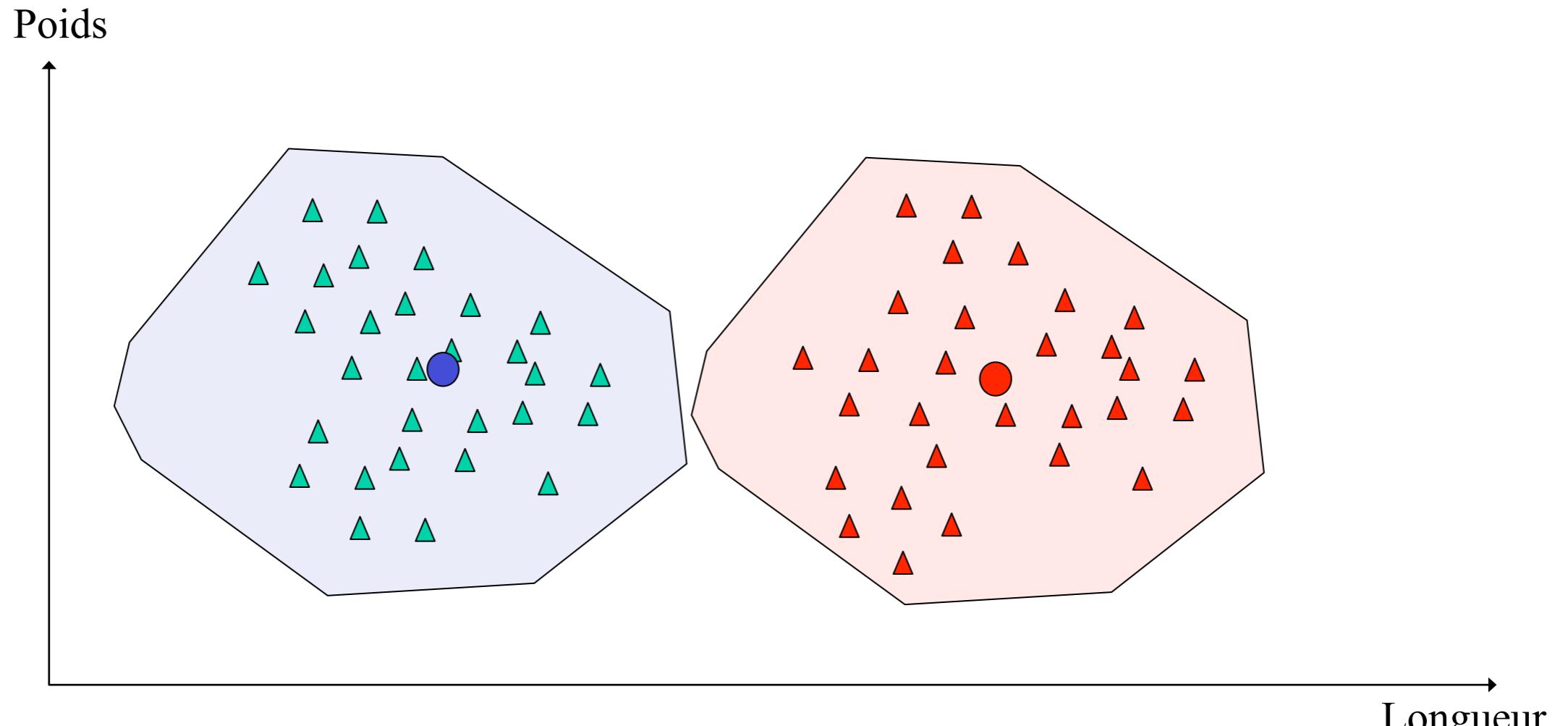
# Algorithme des K-moyennes (*K-means*)

Exemple schématique de l'algorithme des K-moyennes



# Algorithme des K-moyennes (*K-means*)

Exemple schématique de l'algorithme des K-moyennes



On déplace les deux centres sur la moyenne des données de chaque classe

Lorsque l'algorithme converge, ● et ○ correspondent à **la moyenne de chaque classe**

# Algorithme des K-moyennes (*K-means*)

L'algorithme des K-moyennes est un des algorithmes non supervisés parmi les plus simples. Parce qu'il est simple, cet algorithme fait de nombreuses hypothèses simplificatrices parmi lesquels

1.  $P(\text{mer}) = P(\text{terre})$
2. Les données de chaque classe se distribuent suivant des gaussiennes ayant **le même écart type**
3. Chaque donnée ne peut appartenir qu'à **une seule classe à la fois.**

# Algorithme des K-moyennes (*K-means*)

On a vu que le seuil « optimal »  $T$  est celui pour lequel

$$P(\text{terre})P(T \mid \text{terre}) = P(\text{mer})P(T \mid \text{mer})$$

# Algorithme des K-moyennes (*K-means*)

On a vu que le seuil « optimal »  $T$  est celui pour lequel

$$P(\text{terre})P(T \mid \text{terre}) = P(\text{mer})P(T \mid \text{mer})$$

$$P(\text{terre}) \times N(T; \mu_{\text{terre}}, \sigma_{\text{terre}}) = P(\text{mer}) \times N(T; \mu_{\text{mer}}, \sigma_{\text{mer}})$$

# Algorithme des K-moyennes (*K-means*)

On a vu que le seuil « optimal »  $T$  est celui pour lequel

$$P(\text{terre})P(T \mid \text{terre}) = P(\text{mer})P(T \mid \text{mer})$$

$$P(\text{terre}) \times N(T; \mu_{\text{terre}}, \sigma_{\text{terre}}) = P(\text{mer}) \times N(T; \mu_{\text{mer}}, \sigma_{\text{mer}})$$

$$N(T; \mu_{\text{terre}}, \sigma_{\text{terre}}) = N(T; \mu_{\text{mer}}, \sigma_{\text{mer}}) \quad (\text{hypothèse 1})$$

# Algorithme des K-moyennes (*K-means*)

On a vu que le seuil « optimal »  $T$  est celui pour lequel

$$P(\text{terre})P(T \mid \text{terre}) = P(\text{mer})P(T \mid \text{mer})$$

$$P(\text{terre}) \times N(T; \mu_{\text{terre}}, \sigma_{\text{terre}}^2) = P(\text{mer}) \times N(T; \mu_{\text{mer}}, \sigma_{\text{mer}}^2)$$

$$N(T; \mu_{\text{terre}}, \sigma_{\text{terre}}^2) = N(T; \mu_{\text{mer}}, \sigma_{\text{mer}}^2) \quad (\text{hypothèse 1})$$

$$\frac{1}{\sqrt{2\pi}\sigma_{\text{terre}}} e^{-\frac{(T-\mu_{\text{terre}})^2}{2\sigma_{\text{terre}}^2}} = \frac{1}{\sqrt{2\pi}\sigma_{\text{mer}}} e^{-\frac{(T-\mu_{\text{mer}})^2}{2\sigma_{\text{mer}}^2}}$$

# Algorithme des K-moyennes (*K-means*)

On a vu que le seuil « optimal »  $T$  est celui pour lequel

$$P(\text{terre})P(T \mid \text{terre}) = P(\text{mer})P(T \mid \text{mer})$$

$$P(\text{terre}) \times N(T; \mu_{\text{terre}}, \sigma_{\text{terre}}^2) = P(\text{mer}) \times N(T; \mu_{\text{mer}}, \sigma_{\text{mer}}^2)$$

$$N(T; \mu_{\text{terre}}, \sigma_{\text{terre}}^2) = N(T; \mu_{\text{mer}}, \sigma_{\text{mer}}^2) \quad (\text{hypothèse 1})$$

$$\frac{1}{\sqrt{2\pi}\sigma_{\text{terre}}} e^{-\frac{(T-\mu_{\text{terre}})^2}{2\sigma_{\text{terre}}^2}} = \frac{1}{\sqrt{2\pi}\sigma_{\text{mer}}} e^{-\frac{(T-\mu_{\text{mer}})^2}{2\sigma_{\text{mer}}^2}}$$

$$(T - \mu_{\text{terre}})^2 = (T - \mu_{\text{mer}})^2 \quad (\text{hypothèse 2})$$

# Algorithme des K-moyennes (*K-means*)

L'objectif des K-Moyennes: **minimiser la distance quadratique globale** suivante

$$J = \sum_s (y_s - \mu_{x_s})^2$$

En d'autres mots, donner l'étiquette « mer » ou « terre » à chaque pixel afin de minimiser  $J$ .

Pour ce faire, K-Moyennes doit:

1. Calculer le champ d'étiquettes  $x$ ;
2. Calculer l'intensité moyenne de chaque classe  $\mu_{terre}$  et  $\mu_{mer}$

# Algorithme des K-moyennes (*K-means*)

## 1. Calculer le champ d'étiquettes $x$

Si  $\mu_{terre}$  et  $\mu_{mer}$  sont connus, alors on peut facilement calculer  $x$ :

POUR CHAQUE site  $s$  du champ d'observations  $y$  FAIRE

SI  $(y_s - \mu_{terre})^2 < (y_s - \mu_{mer})^2$  ALORS  
 $x_s = 1$  /\* Étiquette « *terre* » au pixel (i,j) \*/

SINON

$x_s = 0$  /\* Étiquette « *mer* » au pixel (i,j) \*/

# Algorithme des K-moyennes (*K-means*)

## 2. Calculer $\mu_{terre}$ et $\mu_{mer}$

Si  $x$  est connu, alors calculer  $\mu_{terre}$  et  $\mu_{mer}$  est facile

$$\mu_{mer} = \mu_{terre} = 0$$

POUR CHAQUE site s du champ d'observations y FAIRE

SI  $x_s == 1$  ALORS

$$\mu_{terre} = \mu_{terre} + y_s$$

SINON

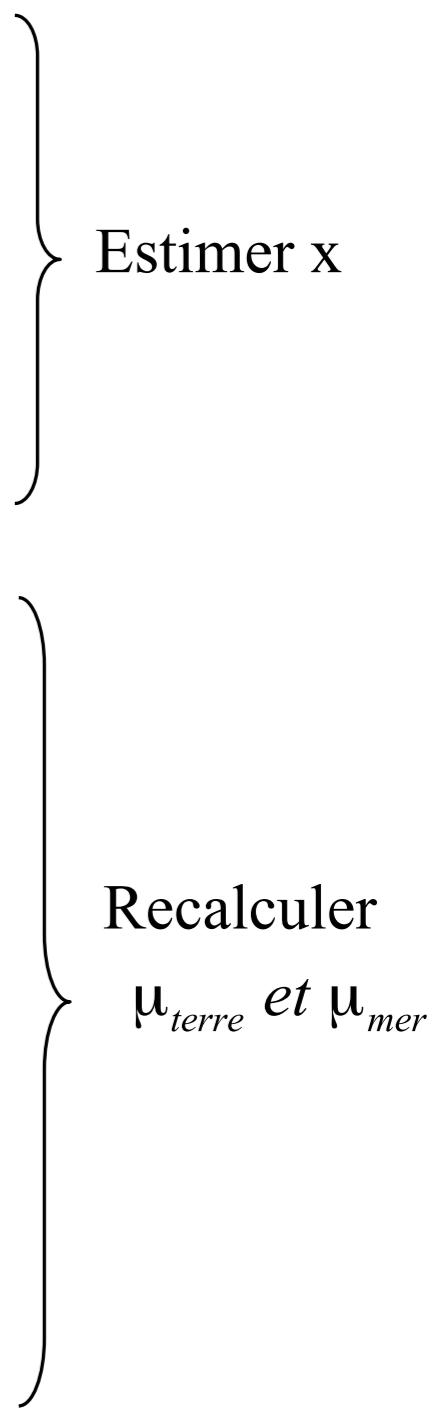
$$\mu_{mer} = \mu_{mer} + y_s$$

$$\mu_{mer} = \mu_{mer} / \text{Nombre de pixels "Mer"}$$

$$\mu_{terre} = \mu_{terre} / \text{Nombre de pixels "Terre"}$$

## Algorithme des K-Moyennes pour 2 classes

```
0.  $\mu_{mer}$  = un pixel  $y_s$  pris au hasard  
 $\mu_{terre}$  = un autre pixel  $y_s$  pris au hasard  
1. POUR CHAQUE site  $s$  du champ d'observations y FAIRE  
    SI  $(y_s - \mu_{terre})^2 < (y_s - \mu_{mer})^2$  ALORS  
         $x_s = 1$       /* Étiquette « terre » au pixel (i,j) */  
    SINON  
         $x_s = 0$       /* Étiquette « mer » au pixel (i,j) */  
2.  $\mu_{mer} = \mu_{terre} = 0$   
3. POUR CHAQUE site  $s$  du champ d'observations y FAIRE  
    SI  $x_s == 1$  ALORS  
         $\mu_{terre} = \mu_{terre} + y_s$   
    SINON  
         $\mu_{mer} = \mu_{mer} + y_s$   
4.  $\mu_{mer} = \mu_{mer} / \text{Nombre de pixels "Mer"}$   
 $\mu_{terre} = \mu_{terre} / \text{Nombre de pixels "Terre"}$   
5. SI  $\mu_{terre}$  et  $\mu_{mer}$  ne changent plus ALORS arrêter SINON retour à 1
```



# Algorithme des K-moyennes (*K-means*)

Algorithme des K-Moyennes pour un nombre arbitraire de classes

0. Initialiser la moyenne de chaque classe  $\mu_c$
1. POUR CHAQUE site  $s$  du champ d'observations  $y$  FAIRE  
 $x_s$  = étiquette de la classe dont la moyenne  $\mu_c$   
est la plus proche de  $y_s$ .
3. Recalculer la moyenne de chaque classe.
4. SI les moyennes ne changent plus ALORS arrêter SINON retour à 1

Pour en savoir plus au sujet de K-Means :

- Tutoriel d'andrew Moore : <http://www.autonlab.org/tutorials/kmeans.html>
- Livre de D.MacKay, « ***Information Theory, Inference, and Learning Algorithms*** »,  
Chapitre 20. (<http://www.inference.phy.cam.ac.uk/mackay/itprnn/book.html>)

# Algorithme des K-moyennes (*K-means*)

Algorithme des K-Moyennes pour un nombre arbitraire de classes

0. Initialiser la moyenne de chaque classe  $\mu_c$
1. POUR CHAQUE site  $s$  du champ d'observations  $y$  FAIRE  
 $x_s$  = étiquette de la classe dont la moyenne  $\mu_c$   
est la plus proche de  $y_s$ .
3. Recalculer la moyenne de chaque classe.
4. SI les moyennes ne changent plus ALORS arrêter SINON retour à 1

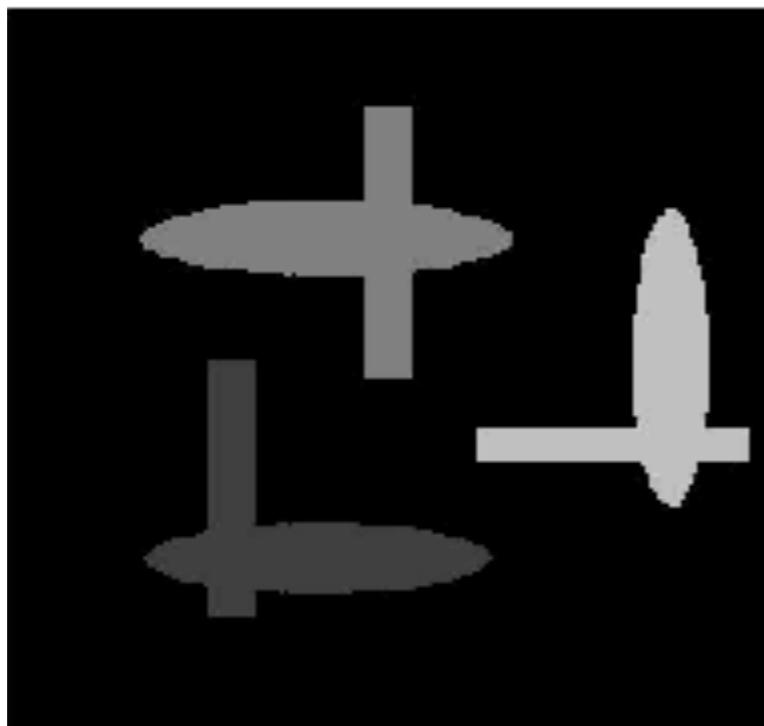
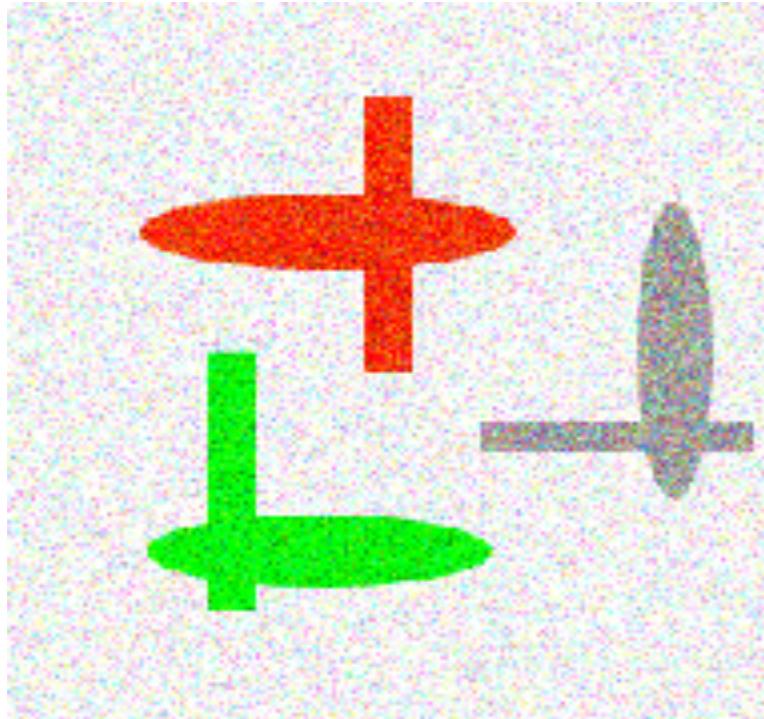
Pour en savoir plus au sujet de K-Means :

- Tutoriel d'andrew Moore : <http://www.autonlab.org/tutorials/kmeans.html>
- Livre de D.MacKay, « ***Information Theory, Inference, and Learning Algorithms*** »,  
Chapitre 20. (<http://www.inference.phy.cam.ac.uk/mackay/itprnn/book.html>)

DÉMO

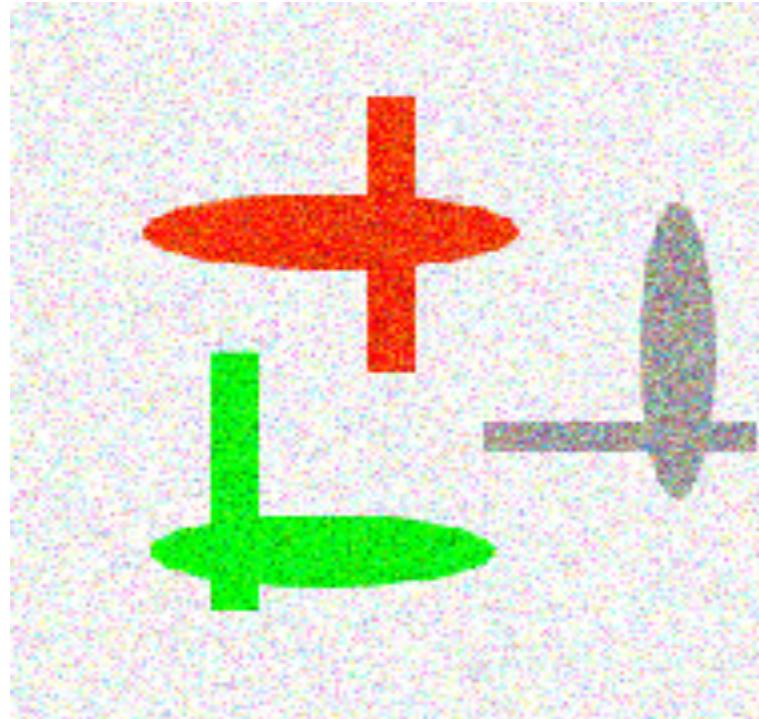
# Algorithme des K-moyennes (*K-means*)

Segmentation 4 classes

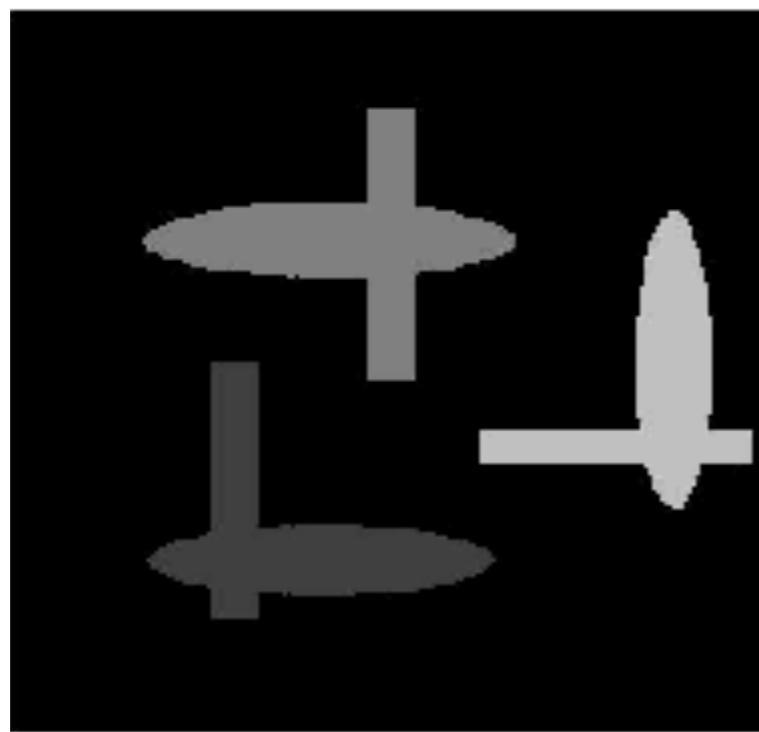


# Algorithme des K-moyennes (*K-means*)

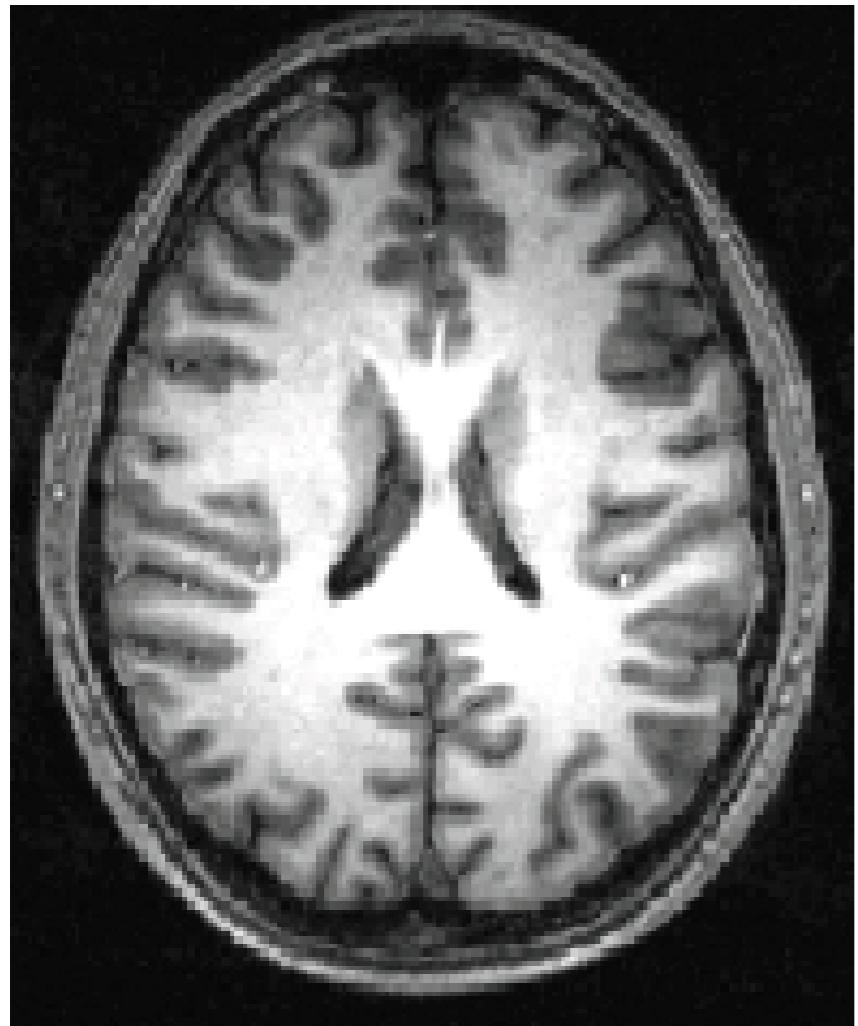
Segmentation 4 classes



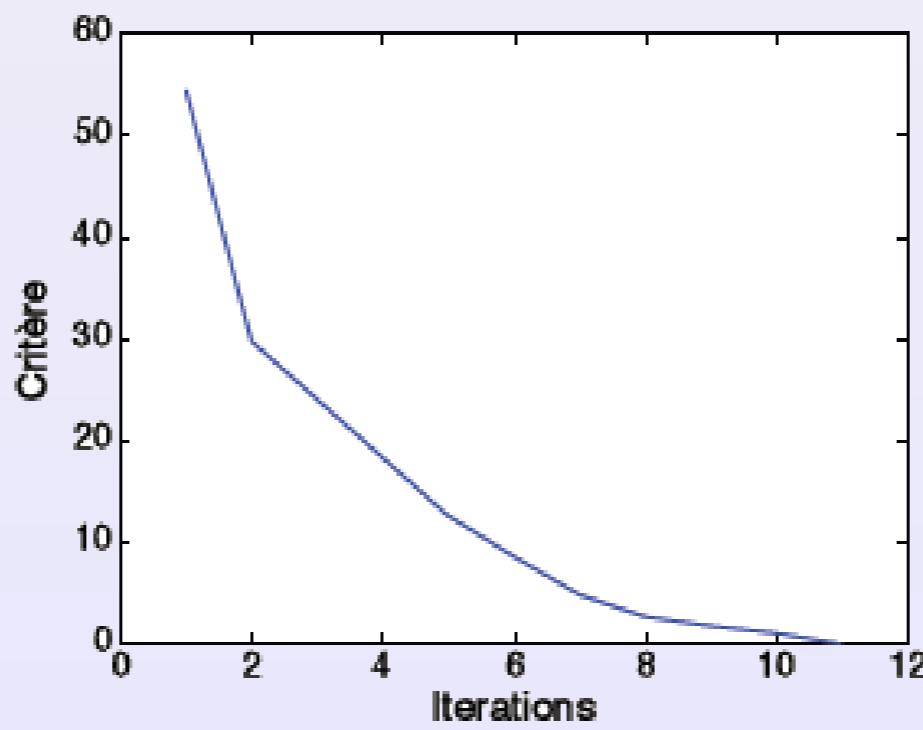
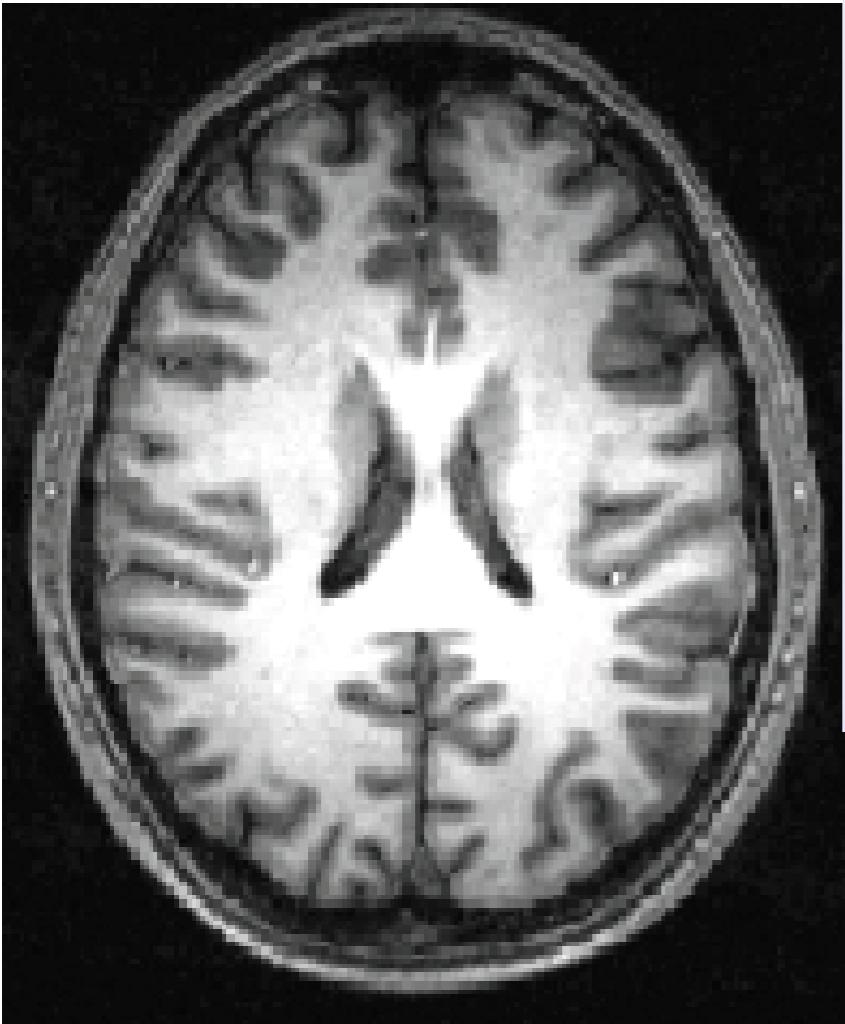
Segmentation 5 classes



# K-means

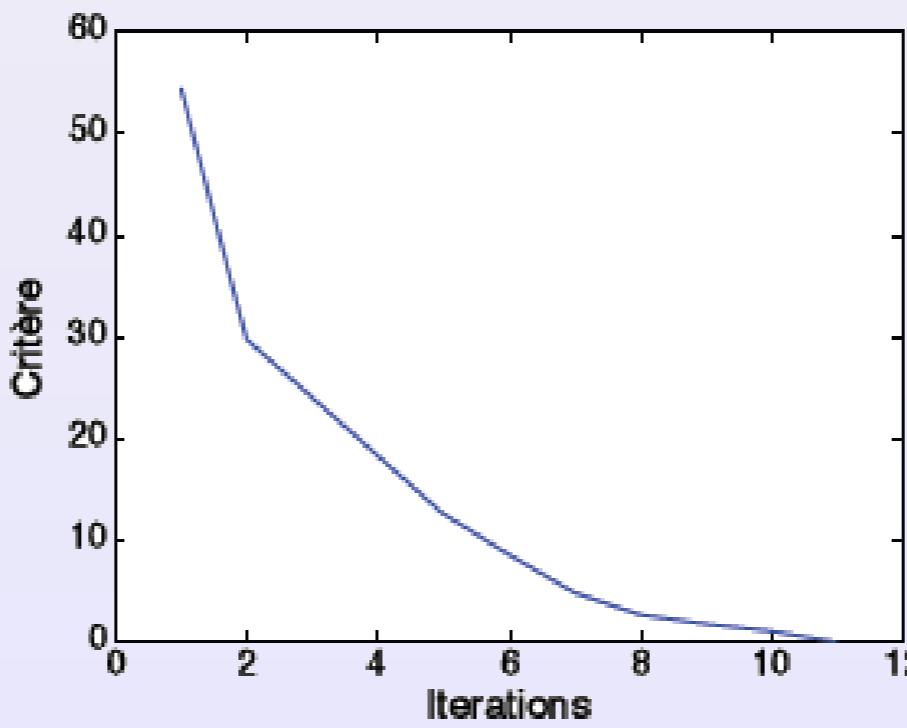
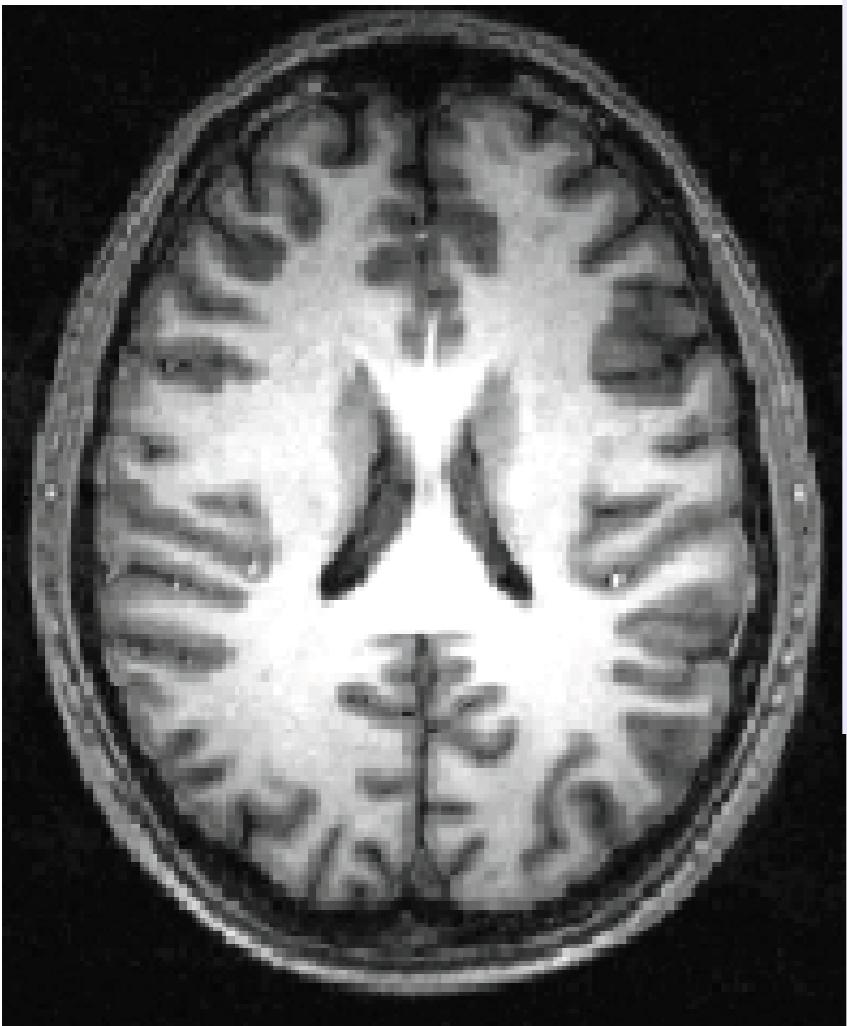


# K-means

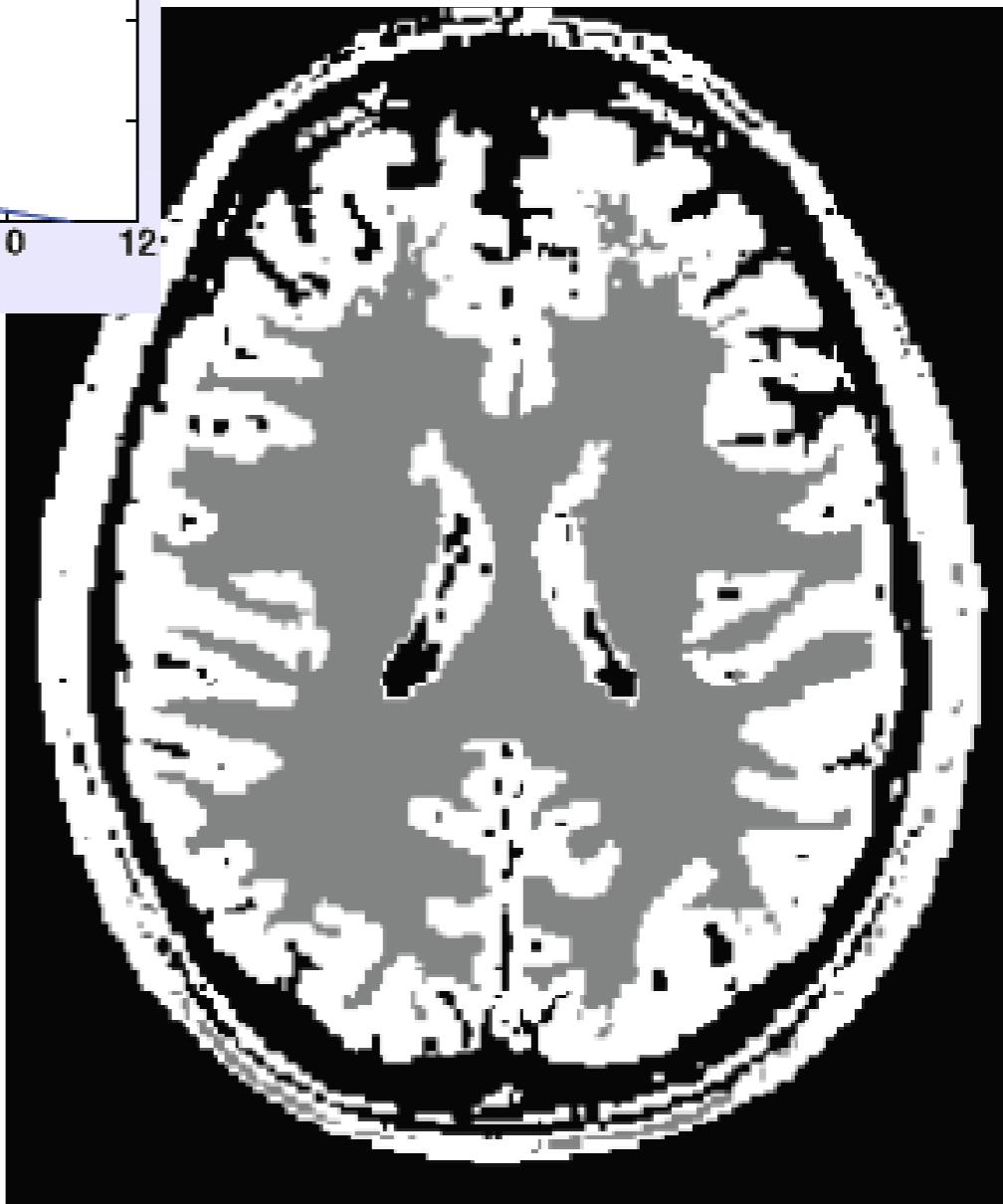


Convergence du critère

# K-means



Convergence du critère



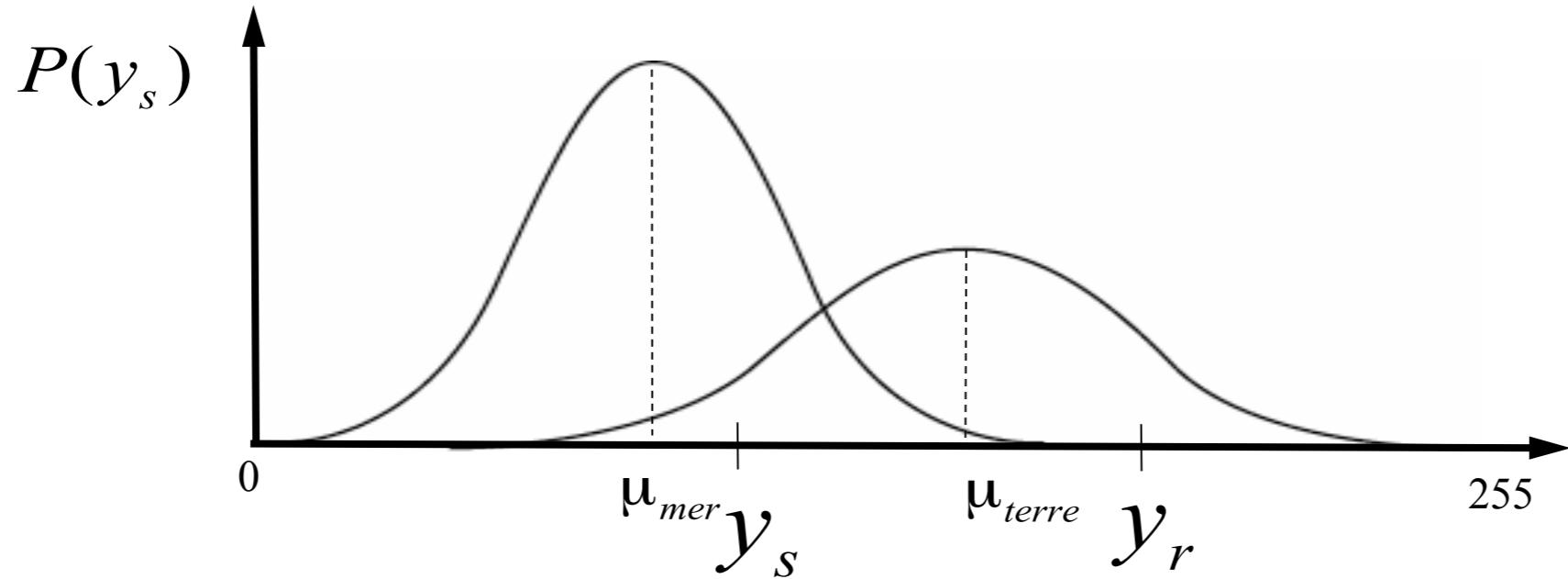
# Algorithme *Soft K-means* ou *Fuzzy c-means*

L'algorithme des K-Moyennes émet l'hypothèse que chaque site n'appartient qu'à une seule classe à la fois, ce qui est faux! Pour remédier à cette hypothèse parfois abusive, l'algorithme « *soft* » *k-means*, ou *fuzzy c-means* permet à chaque site d'appartenir à toutes les classes en même temps, mais avec des proportions différentes.

# Algorithme *Soft K-means* ou *Fuzzy c-means*

L'algorithme des K-Moyennes émet l'hypothèse que chaque site n'appartient qu'à une seule classe à la fois, ce qui est faux! Pour remédier à cette hypothèse parfois abusive, l'algorithme « *soft* » *k-means*, ou *fuzzy c-means* permet à chaque site d'appartenir à toutes les classes en même temps, mais avec des proportions différentes.

Exemple:



le site **S** appartient à la classe mer dans une proportion de 0.7 et à la classe terre dans une proportion de 0.3

le site **r** appartient à la classe mer dans une proportion de 0.05 et à la classe terre dans une proportion de 0.95

# Algorithme *Soft K-means*

Avec *K-means*, on cherche à minimiser l'erreur quadratique globale

$$J_{km} = \sum_s (y_s - \mu_{x_s})^2$$

# Algorithme *Soft K-means*

Avec *K-means*, on cherche à minimiser l'erreur quadratique globale

$$J_{km} = \sum_s (y_s - \mu_{x_s})^2$$

Avec *Soft K-means*, on cherche à minimiser l'erreur pondérée globale

$$J_{skm} = \sum_s \sum_{c=1}^{N_c} P(c | y_s) (y_s - \mu_c)^2$$

# Algorithme *Soft K-means*

Avec *K-means*, on cherche à minimiser l'erreur quadratique globale

$$J_{km} = \sum_s (y_s - \mu_{x_s})^2$$

Avec *Soft K-means*, on cherche à minimiser l'erreur pondérée globale

$$J_{skm} = \sum_s \sum_{c=1}^{N_c} P(c | y_s) (y_s - \mu_c)^2$$

Où c est une étiquette de classe (c = {mer, terre})

$N_c$  est le nombre total de classes (2 dans l'exemple terre-mer)

$P(c | y_s)$  est la proportion avec laquelle  $y_s$  appartient à la classe c.

$$P(c | y_s) = \frac{e^{-\beta |y_s - \mu_c|}}{\sum_r e^{-\beta |y_s - \mu_r|}}$$

## Algorithme *soft k-means* pour deux classes

0.  $\mu_{mer}$  = un pixel  $y_s$  pris au hasard

$\mu_{terre}$  = un autre pixel  $y_s$  pris au hasard

1. POUR CHAQUE site  $s$  du champ d'observations y FAIRE

$$d_t = \beta |y_s - \mu_{terre}|$$

$$d_m = \beta |y_s - \mu_{mer}|$$

$$P(mer | y_s) = \frac{e^{-d_m}}{e^{-d_m} + e^{-d_t}}$$

$$P(terre | y_s) = \frac{e^{-d_t}}{e^{-d_m} + e^{-d_t}}$$

2.  $\mu_{mer} = \mu_{terre} = T_{mer} = T_{terre} = 0$

3. POUR CHAQUE site  $s$  du champ d'observations y FAIRE

$$\mu_{mer} = \mu_{mer} + P(mer | y_s) \times y_s; T_{mer} = T_{mer} + P(mer | y_s)$$

$$\mu_{terre} = \mu_{terre} + P(terre | y_s) \times y_s; T_{terre} = T_{terre} + P(terre | y_s)$$

4.  $\mu_{mer} = \mu_{mer} / T_{mer}$

$\mu_{terre} = \mu_{terre} / T_{terre}$

5. SI  $\mu_{terre}$  et  $\mu_{mer}$  ne changent plus ALORS arrêter SINON retour à 1

} Estimer  $P(c | y_s)$  pour tous les pixels.

} Recalculer  $\mu_{terre}$  et  $\mu_{mer}$

# Algorithme *Soft K-means*

Algorithme de soft k-means pour un nombre arbitraire de classes

0. Initialiser la moyenne de chaque classe  $\mu_c$
1. POUR CHAQUE site  $s$  du champ d'observations y FAIRE

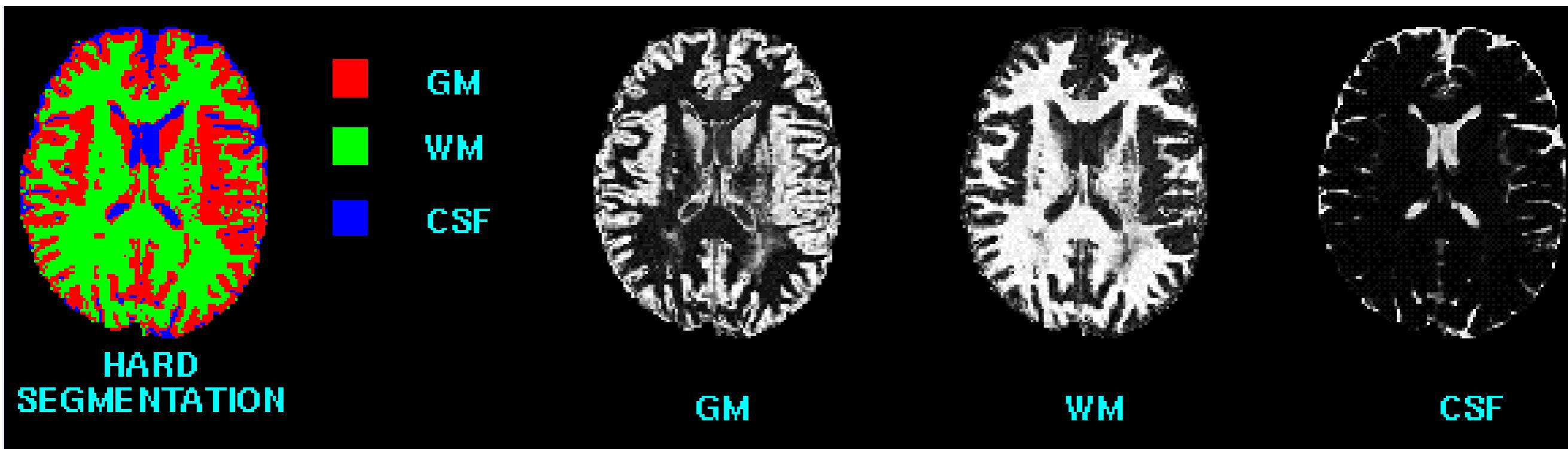
$$\text{Calculer pour chaque classe "c": } P(c | y_s) = \frac{e^{-d_c}}{\sum_r e^{-d_r}}$$

3. Calculer la moyenne de chaque classe « c »

$$\mu_c = \frac{\sum_s P(c | y_s) y_s}{\sum_s P(c | y_s)}$$

4. SI les moyennes  $\mu_c$  ne changent plus ALORS arrêter SINON retour à 1

# Soft k-means



# *Expectation-Maximisation*

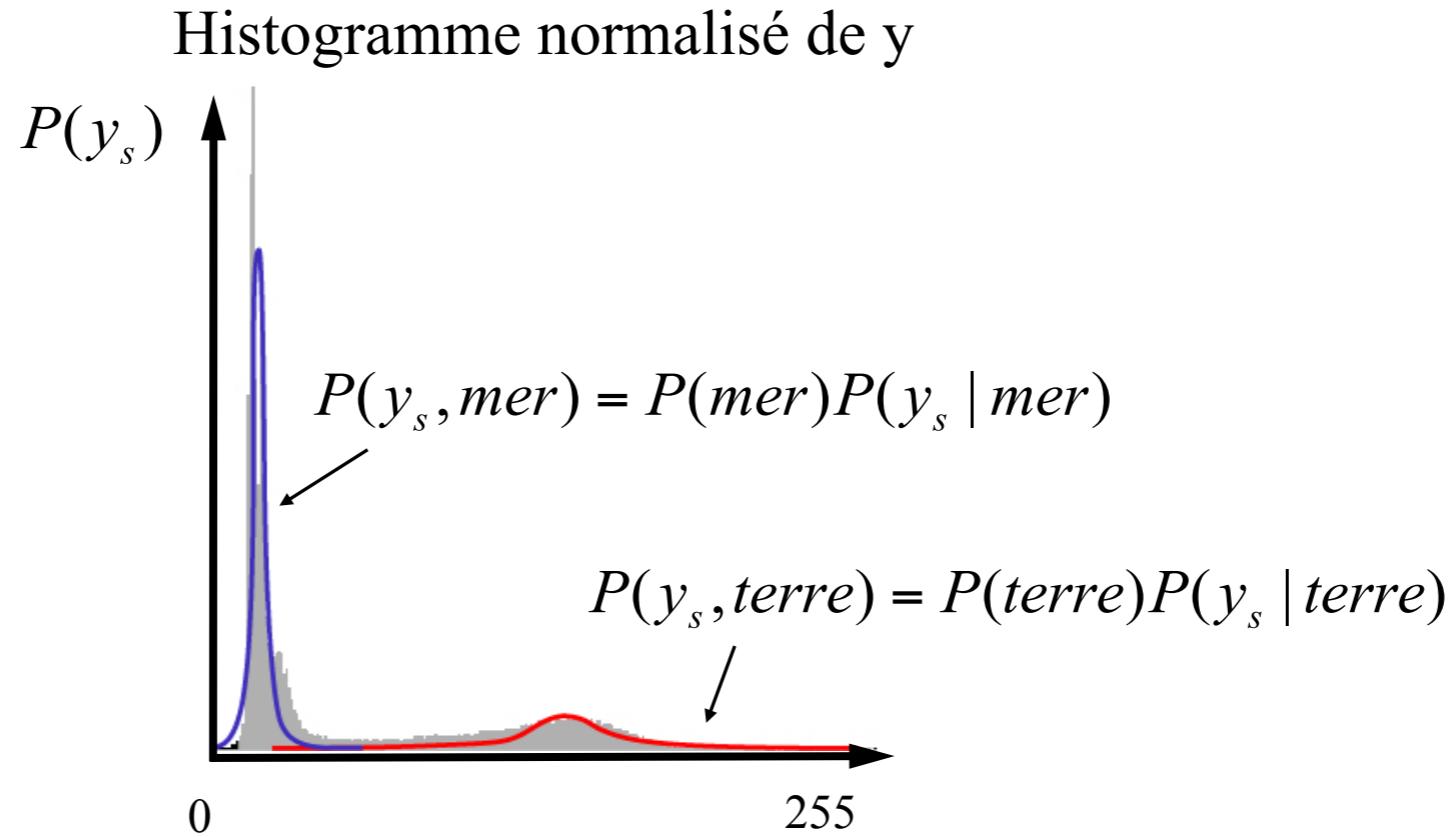
*K-means* et soft *k-means* sont deux algorithmes qui ne font qu'estimer la **moyenne** de chaque classe.

Un algorithme plus puissant du nom de «E-M» permet d'estimer TOUS les paramètres de la mixture de gaussiennes, c'est-à-dire:

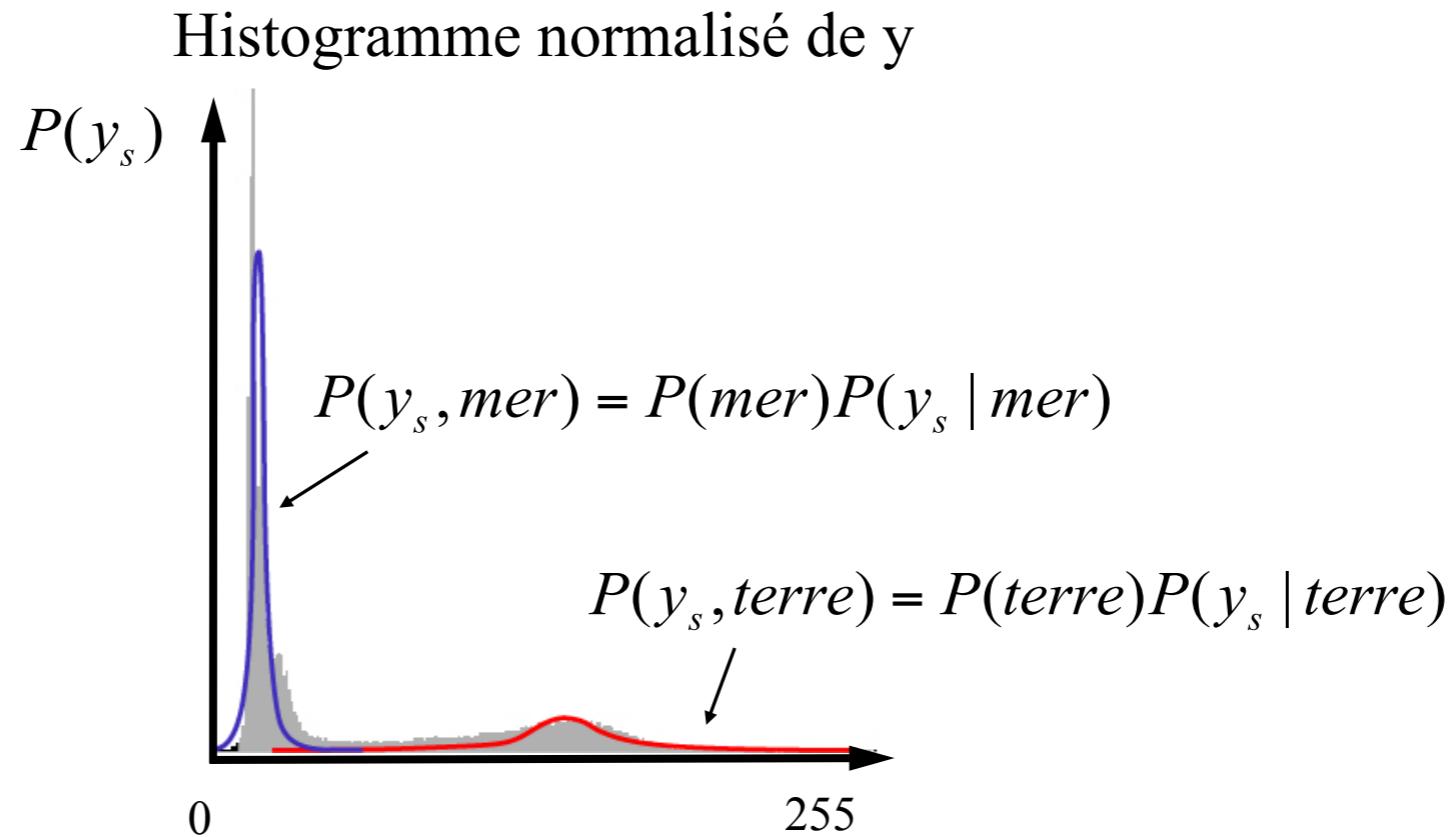
$$\mu_i, \sigma_i \text{ et } P(x_s = i)$$

pour chaque gaussienne.

# Expectation-Maximisation



# *Expectation-Maximisation*



$$P(y_s) = \sum_c P(c)P(y_s \mid c)$$

Puisqu'on a une mixture de gaussiennes

$$P(y_s | \theta) = \sum_c P(c) P(y_s | c, \theta_c)$$

où  $\theta_c = \{\mu_c, \sigma_c\}$

# *Expectation-Maximisation*

$$P(y_s | \theta) = \sum_c P(c)P(y_s | c, \theta_c)$$

probabilité d'observer un pixel  $y_s$

$$P(y | \theta) = \prod_s P(y_s | \theta)$$

probabilité d'observer tous les pixels de l'image  $y$



Les meilleurs paramètres  $\theta$  sont ceux qui **maximisent la vraisemblance**

$$\begin{aligned}\theta &= \arg \max_{\theta'} \prod_s P(y_s | \theta') \\ &= \arg \max_{\theta'} \sum_s \ln P(y_s | \theta')\end{aligned}$$

$$P(y_s | \theta) = \sum_c P(c)P(y_s | c, \theta_c)$$

# Expectation-Maximisation

Les paramètres  $\theta$  qui **maximisent la vraisemblance** sont ceux pour lesquels

$$\frac{d}{d\theta_i} \left( \sum_s \ln P(y_s | \theta) \right) = 0$$

$$\frac{d}{d\mu_i} \left( \sum_s \ln P(y_s | \theta) \right) = 0$$

$$\sum_s \frac{d}{d\mu_i} (\ln P(y_s | \theta)) = 0$$

$$\sum_s \frac{1}{P(y_s | \theta)} \frac{d}{d\mu_i} (P(y_s | \theta)) = 0$$

$$P(y_s | \theta) = \sum_c P(c)P(y_s | c, \theta_c)$$

# Expectation-Maximisation

Les paramètres  $\theta$  qui **maximisent la vraisemblance** sont ceux pour lesquels

$$\frac{d}{d\theta_i} \left( \sum_s \ln P(y_s | \theta) \right) = 0$$

Voyons ce qui arrive lorsque  $P(y_s | \theta)$  est une mixture de gaussiennes.

$$\frac{d}{d\mu_i} \left( \sum_s \ln P(y_s | \theta) \right) = 0$$

$$\sum_s \frac{d}{d\mu_i} (\ln P(y_s | \theta)) = 0$$

$$\sum_s \frac{1}{P(y_s | \theta)} \frac{d}{d\mu_i} (P(y_s | \theta)) = 0$$

# Expectation-Maximisation

$$\sum_s \frac{1}{P(y_s | \theta)} \frac{d}{d\mu_i} (P(y_s | \theta)) = 0$$

$$\sum_s \frac{1}{P(y_s | \theta)} \frac{d}{d\mu_i} \left( \sum_c P(c) \frac{1}{\sqrt{2\pi}\sigma_c} e^{-\frac{(y_s - \mu_c)^2}{2\sigma_c^2}} \right) = 0$$

$$\sum_s \frac{1}{P(y_s | \theta)} \frac{d}{d\mu_i} \left( P(i) \frac{1}{\sqrt{2\pi}\sigma_i} e^{-\frac{(y_s - \mu_i)^2}{2\sigma_i^2}} \right) = 0$$

$$\sum_s \frac{1}{P(y_s | \theta)} \frac{2P(i)}{2\sigma_i^2} (y_s - \mu_i) \frac{1}{\sqrt{2\pi}\sigma_i} e^{-\frac{(y_s - \mu_i)^2}{2\sigma_i^2}} = 0$$

$$\sum_s \frac{1}{P(y_s | \theta)} \frac{P(i)}{\sigma_i^2} (y_s - \mu_i) P(y_s | i, \theta_i) = 0$$

$$\sum_s \frac{P(i | y_s, \theta)}{P(y_s | i, \theta_i) P(i)} \frac{P(i)}{\sigma_i^2} (y_s - \mu_i) P(y_s | i, \theta_i) = 0 \quad \text{car } P(i | y_s, \theta_i) = \frac{P(y_s | i, \theta_i) P(i)}{p(y_s | \theta_i)}$$

(Bayes)

# *Expectation-Maximisation*

$$\sum_s \frac{P(i | y_s, \theta)}{P(y_s | i, \theta) P(i)} \frac{P(i)}{\sigma_i^2} (y_s - \mu_i) P(y_s | i, \theta) = 0$$

$$\frac{1}{\sigma_i^2} \sum_s P(i | y_s, \theta) (y_s - \mu_i) = 0$$

$$\mu_i = \frac{\sum_s P(i | y_s, \theta) y_s}{\sum_s P(i | y_s, \theta)}$$

# Expectation-Maximisation

Et si on fait la même chose pour  $\sigma_c$  et  $P(c)$ , c'est-à-dire:

$$\frac{d}{d\sigma_c} \left( \sum_s \ln P(y_s | \theta) \right) = 0$$

$$\frac{d}{dP(c)} \left( \sum_s \ln P(y_s | \theta) \right) = 0$$

on obtient que

$$P(c) = \frac{1}{N_s} \sum_s P(c | y_s, \theta)$$

$$\sigma_c^2 = \frac{\sum_s P(c | y_s, \theta) (y_s - \mu_c)^2}{\sum_s P(c | y_s, \theta)}$$

$N_s$  nombre total de pixels dans l'image

# Expectation-Maximisation

En résumé

$$\mu_c = \frac{\sum_s P(c | y_s, \theta) y_s}{\sum_s P(c | y_s, \theta)}$$

$$\sigma_c^2 = \frac{\sum_s P(c | y_s, \theta) (y_s - \mu_c)^2}{\sum_s P(c | y_s, \theta)}$$

$$P(c) = \frac{1}{N_s} \sum_s P(c | y_s, \theta)$$

$$P(c | y_s, \theta) = \frac{P(y_s | c, \theta_c) P(c)}{\sum_i P(y_s | i, \theta_i) P(i)}$$

$$P(y_s | c, \theta_c) = \frac{1}{\sqrt{2\pi}\sigma_c} e^{-\frac{(y_s - \mu_c)^2}{2\sigma_c^2}}$$

## Algorithme des E-M pour **2 classes**

0.  $P(\text{mer}), P(\text{terre}), \mu_{\text{mer}}, \sigma_{\text{mer}}, \mu_{\text{terre}}, \sigma_{\text{terre}} \leftarrow \text{Init}$

1. FAIRE

Calculer  $P(\text{mer} | y_s, \theta)$  et  $P(\text{terre} | y_s, \theta)$  pour tous les pixels "s":

$P(\text{mer}), P(\text{terre}), \mu_{\text{mer}}, \sigma_{\text{mer}}, \mu_{\text{terre}}, \sigma_{\text{terre}} \leftarrow 0$

2. POUR CHAQUE site s du champ d'observations y FAIRE

$$\mu_{\text{mer}} = \mu_{\text{mer}} + P(\text{mer} | y_s, \theta) y_s; \quad \mu_{\text{terre}} = \mu_{\text{terre}} + P(\text{terre} | y_s, \theta) y_s$$

$$\sigma_{\text{mer}}^2 = \sigma_{\text{mer}}^2 + P(\text{mer} | y_s, \theta) (y_s - \mu_{\text{mer}})^2; \quad \sigma_{\text{terre}}^2 = \sigma_{\text{terre}}^2 + P(\text{terre} | y_s, \theta) (y_s - \mu_{\text{terre}})^2$$

$$P(\text{mer}) = P(\text{mer}) + P(\text{mer} | y_s, \theta); \quad P(\text{terre}) = P(\text{terre}) + P(\text{terre} | y_s, \theta)$$

$$\mu_{\text{mer}} = \mu_{\text{mer}} / P(\text{mer}); \quad \mu_{\text{terre}} = \mu_{\text{terre}} / P(\text{terre})$$

$$\sigma_{\text{mer}}^2 = \sigma_{\text{mer}}^2 / P(\text{mer}); \quad \sigma_{\text{terre}}^2 = \sigma_{\text{terre}}^2 / P(\text{terre})$$

$$P(\text{mer}) = P(\text{mer}) / N_s; \quad P(\text{terre}) = P(\text{terre}) / N_s$$

3. TANT QUE  $\mu_c, \sigma_c$  et  $P(c)$  stabilisent

E

M

## Algorithme des E-M pour $K>2$ classes

0.  $P(c), \mu_c, \sigma_c \leftarrow$  initialiser les paramètres de chaque classe "c"

1. FAIRE

Calculer  $P(c | y_s, \theta)$  pour tous les pixels "s" et toutes les classes "c"

$$P(c | y_s, \theta) = \frac{P(y_s | c, \theta_c) P(c)}{\sum_i P(y_s | i, \theta_i) P(i)}$$

Calculer les paramètres de chaque gaussienne :

$$\mu_c = \frac{\sum_s P(c | y_s, \theta) y_s}{\sum_s P(c | y_s, \theta)}$$

$$\sigma_c^2 = \frac{\sum_s P(c | y_s, \theta) (y_s - \mu_c)^2}{\sum_s P(c | y_s, \theta)}$$

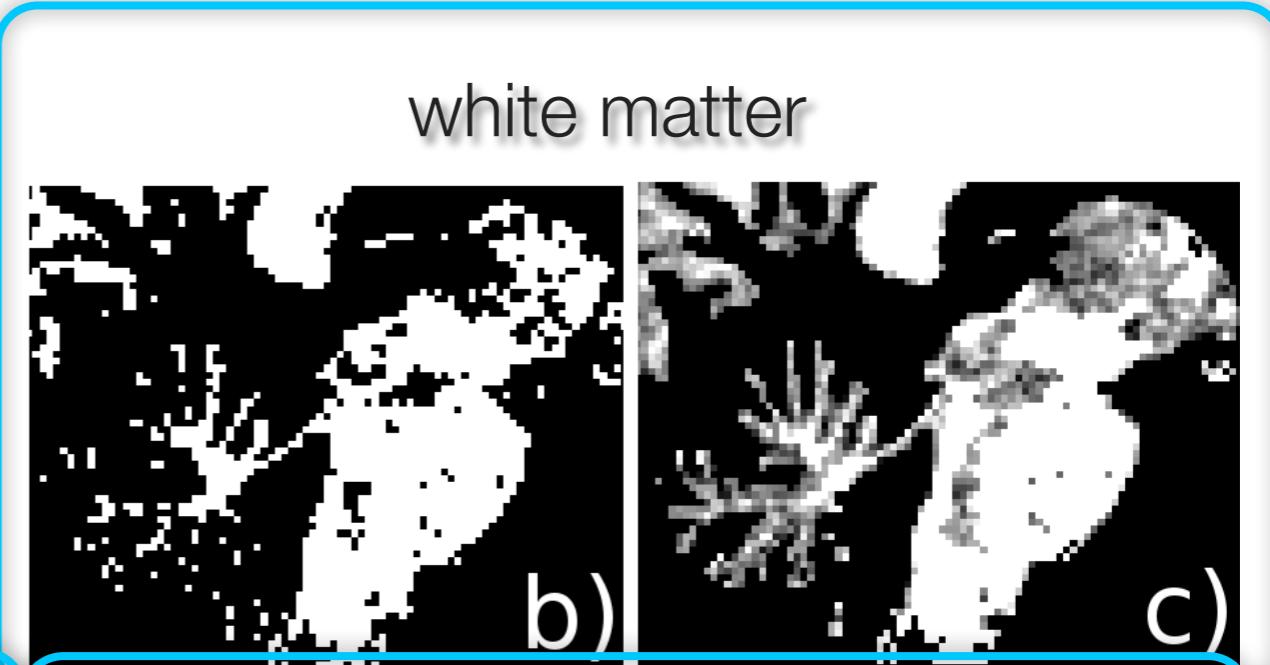
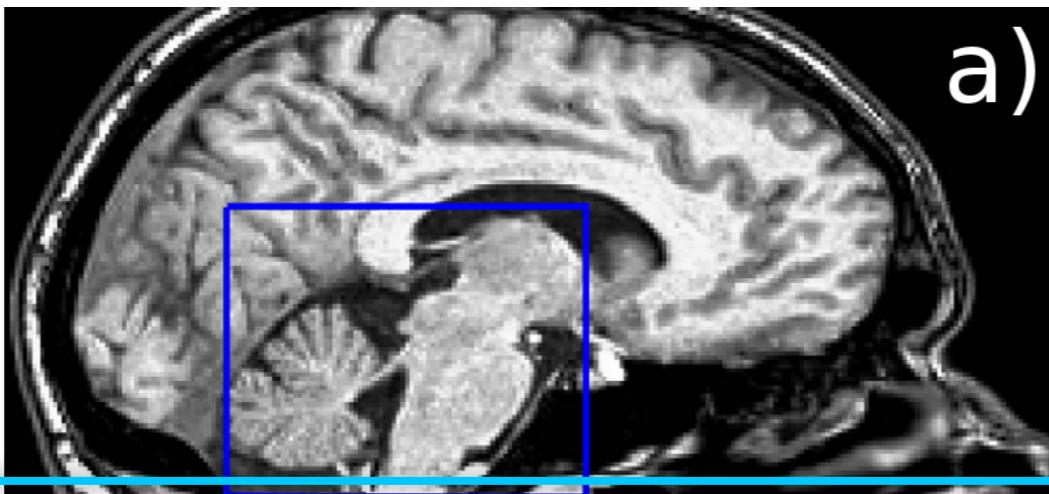
$$P(c) = \frac{1}{N_s} \sum_s P(c | y_s, \theta)$$

3. TANT QUE  $\mu_c, \sigma_c$  et  $P(c)$  ne sont pas stabilisés

E

M

# EM



gray matter

CSF

# En résumé...

Algorithme du seuil	Simple, mais le seuil doit être donné par l'usager <b>(approche supervisée).</b>
Algorithme probabiliste du seuil	Si on connaît la distribution probabiliste de chaque classe, le seuil optimal peut être calculé.
Algorithme des K-Moyennes	Algorithme permettant d'estimer automatiquement la <b>moyenne</b> de chaque classe (gaussiennes).
Algorithme Soft K-Means	Algorithme permettant d'estimer automatiquement la <b>moyenne</b> de chaque classe (gaussiennes).
Algorithme EM	Algorithme permettant d'estimer automatiquement la <b>moyenne, la variance et la proportion</b> de chaque classe (gaussiennes).

# Estimation du nombre de classes

Nous avons vu qu'EM permet d'estimer les paramètres d'une mixture de gaussiennes à savoir :  $\mu_c, \sigma_c$  et  $P(c)$

Toutefois, le nombre de classes «  $K$  » doit toujours être spécifié par un utilisateur.

Nous verrons donc 2 méthodes pour estimer  $K$  automatiquement

1. Test de « gaussianité »
2. MDL (*Minimum Description Length*)

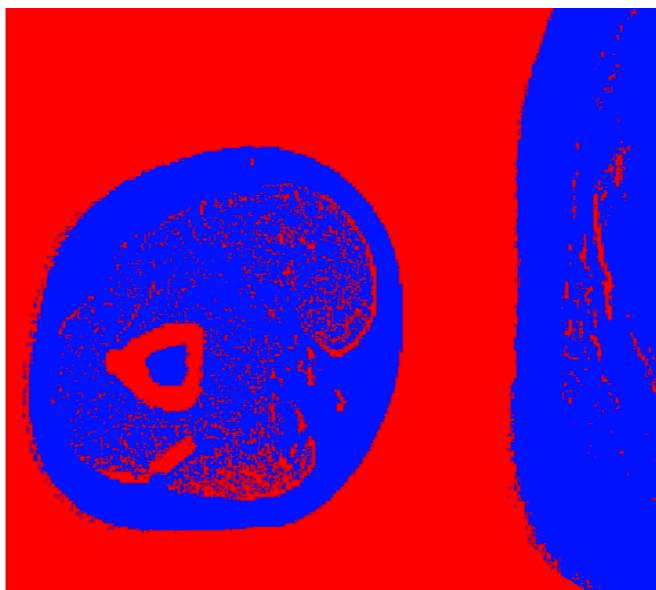
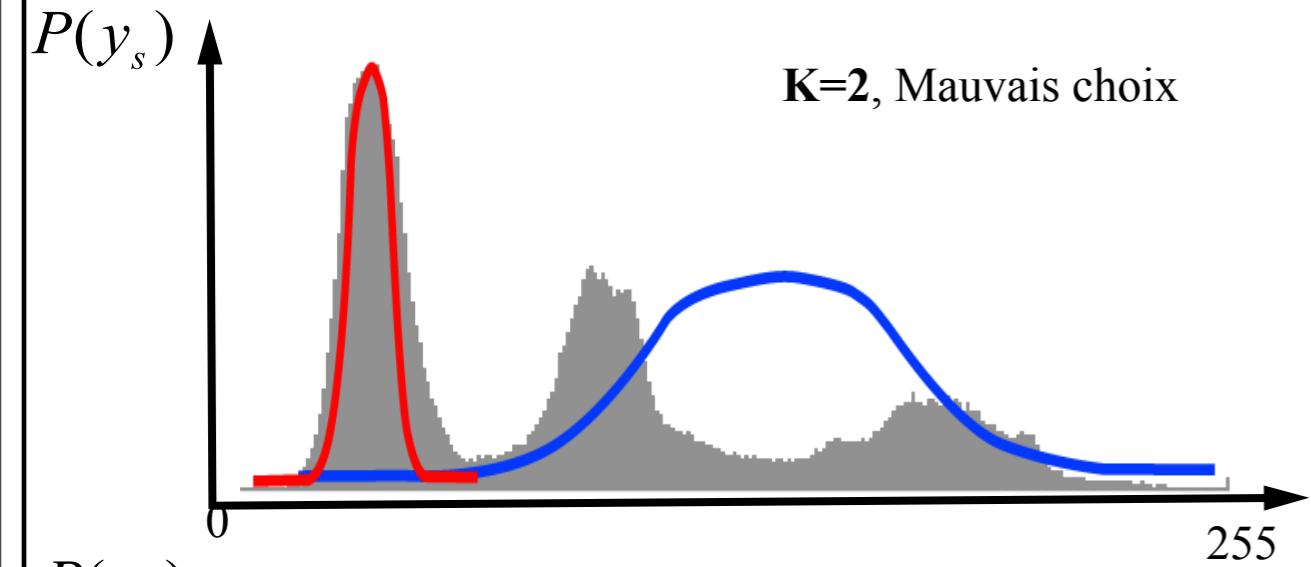
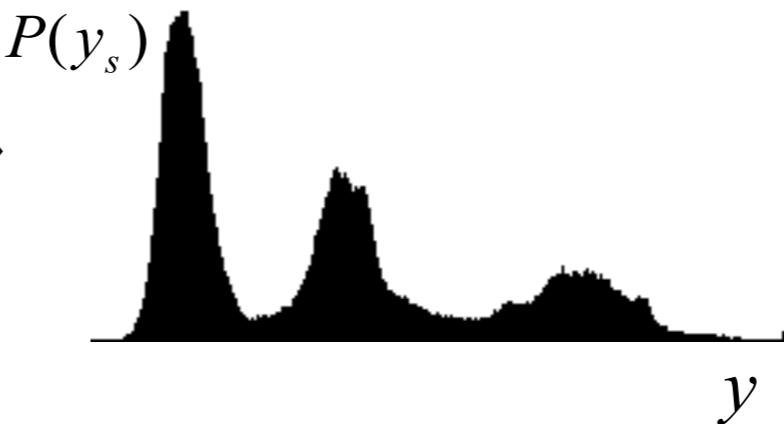
# Test de « gaussianité »

Soient des données  $\{y_s \mid s \in S\}$  se distribuant suivant une mixture de Gaussiennes dont les paramètres doivent être estimés (incluant  $K$  le nombre de classes).

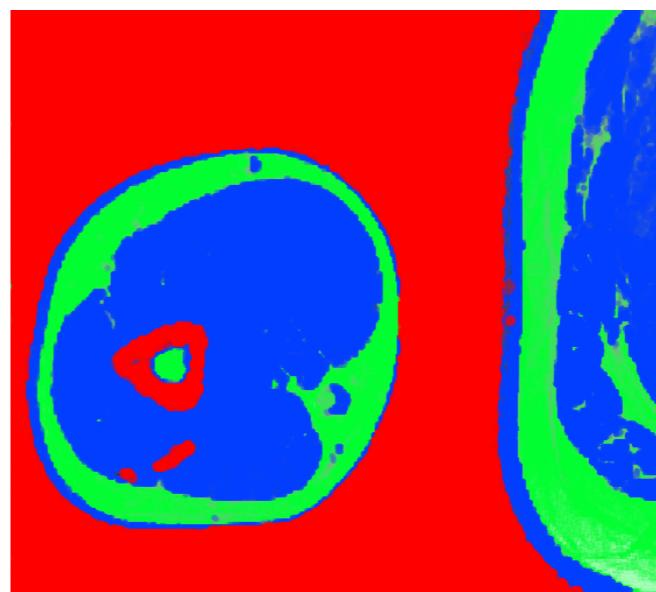
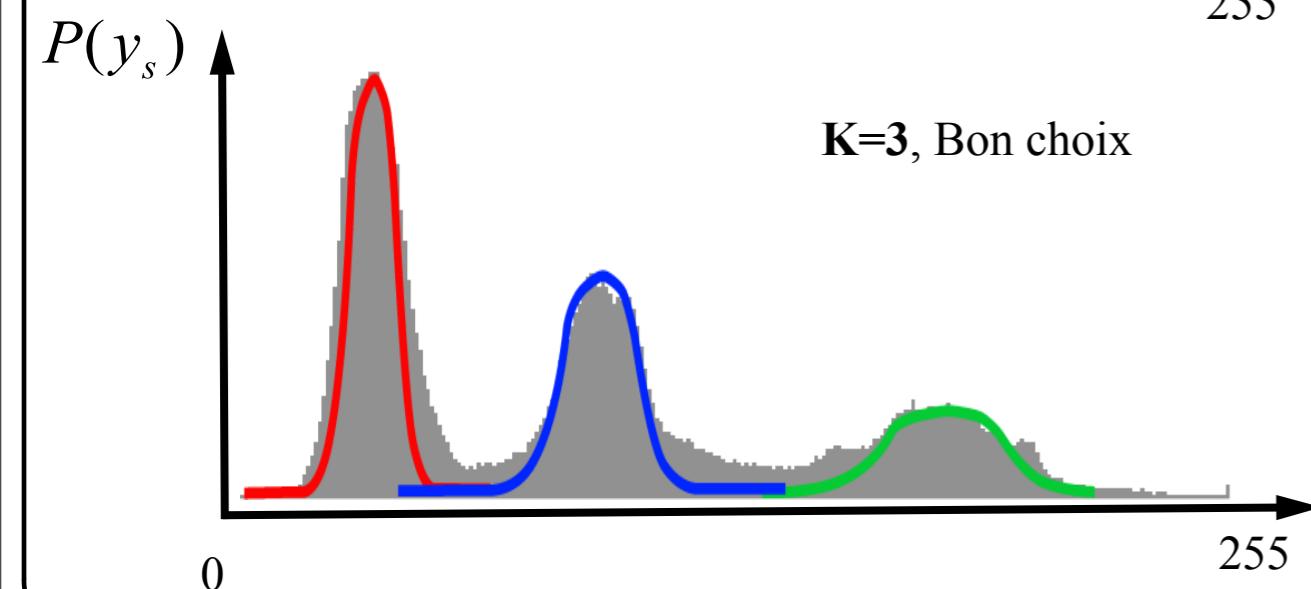
L'idée derrière le test Gaussien est de sélectionner un certain  $K$  pour lequel les données appartenant à une classe « i » se **distribuent vraiment selon la i-ème gaussienne**.

# Test de « gaussianité »

Exemple



- Dist. Gaussienne
- **Dist. Non Gaussienne**



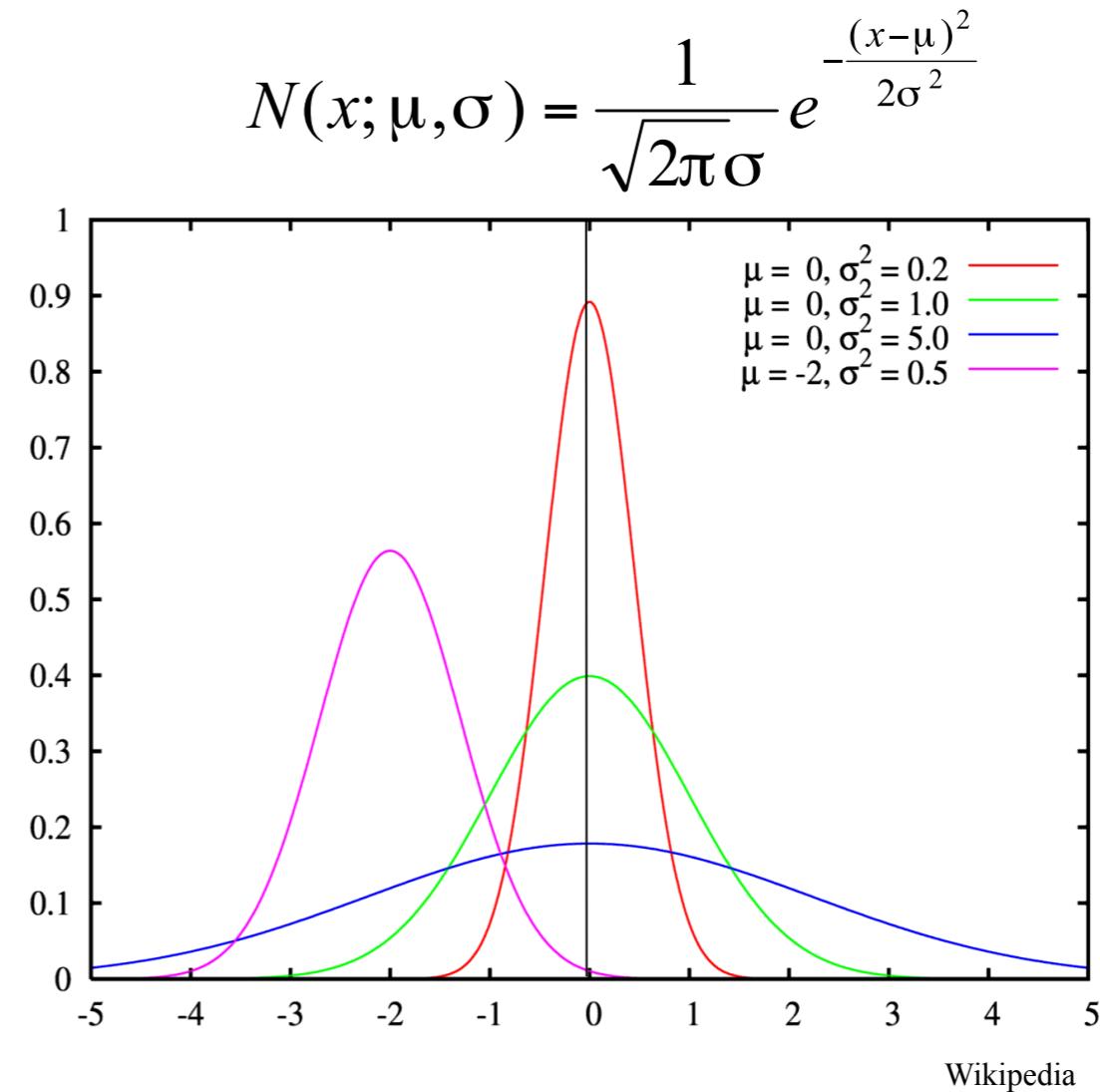
- Dist. Gaussienne
- Dist. Gaussienne
- Dist. Gaussienne

# Test de « gaussianité »

Comment savoir si les données d'une classe suivent une distribution gaussienne ?

Test de normalité ou test de « **goodness of fit** ».

$\sigma$	68.26894921371%
$2\sigma$	95.44997361036%
$3\sigma$	99.73002039367%



## Test de « gaussianité »

1- Étant donné une gaussienne  $N(\mu_c, \sigma_c^2)$  et un ensemble de  $N$  valeurs  $L = \{y_s\}$

2- TabCmpt[3] = {0,0,0};

3- POUR CHAQUE valeur  $y_s$  dans  $L$  FAIRE

$$dist = |y_s - \mu_c|$$

SI  $dist < \sigma$  ALORS

    TabCmpt[1]++;

SINON SI  $dist < 2\sigma$  ALORS

    TabCmpt[2]++;

SINON

    TabCmpt[3]++;

4- TabCmpt = TabCmpt/N;

5- SI  $\text{abs}(\text{TabCmpt}[1]-0.68) > \text{Seuil}$  OU  $\text{abs}(\text{TabCmpt}[1]-0.27) > \text{Seuil}$  OU  $\text{abs}(\text{TabCmpt}[1]-0.05) > \text{Seuil}$   
    RETOUR FAUX;

SINON

    RETOUR VRAI;

Estimer le nombre de classes  $K$  à l'aide d'un test de « gaussianité »

1. POUR  $K = 2$  à NB\_CLASSES\_MAX

$\{\mu_c, \sigma_c^2, P(c)\}_{c=1\dots K}$   $\neg$  Segmentation en  $K$  classes avec  $EM$

fin = VRAI;

2. POUR CHAQUE classes c FAIRE

$L = \{y_s\}_{x_s=c}$  // Mettre dans  $L$  tous les pixels associés à la classe “c”

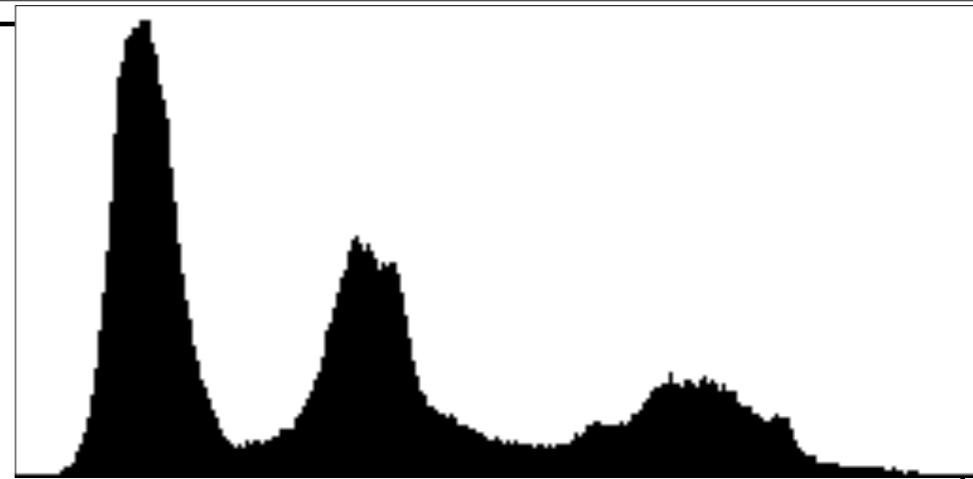
SI les pixels dans “L” NE SUIVENT PAS la distribution gaussienne  $N(\mu_c, \sigma_c^2)$  ALORS  
fin = FAUX;

SI fin == TRUE ALORS

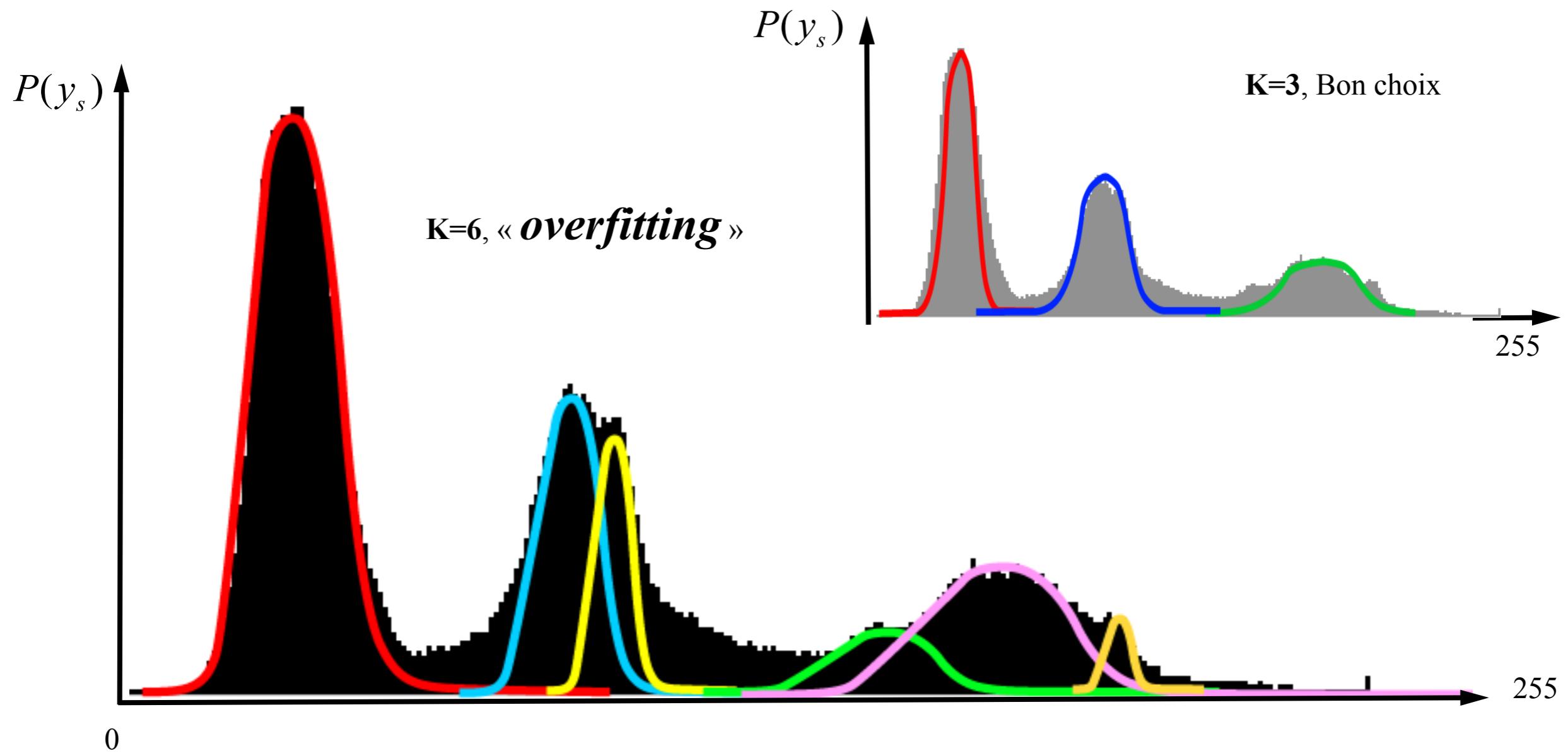
RETOUR K;

# MDL

*Minimum Description Length*



Malheureusement, les tests de normalité ont parfois tendance à choisir un grand nombre de classes « K » lorsque les données ne suivent pas exactement une mixture de gaussiennes



# MDL

*Minimum Description Length*

On cherche une mixture de gaussiennes dont les paramètres :

$$\theta = \{(\mu_1, \sigma_1, P(1)), \dots, (\mu_K, \sigma_K, P(K))\}$$

et le nombre de classes  $K$  vont satisfaire le critère suivant

$$K, \theta = \arg \min_{\theta, K} [L(y | K, \theta) + L(K, \theta)]$$

Mesure le « goodness of fit »

Pénalise les solutions avec un nombre de paramètres élevé.

# MDL

*Minimum Description Length*

$$MDL(\theta, K) = L(y | \theta, K) + L(\theta, K)$$

Suivant le critère de Rissanen

$$\begin{aligned} MDL(\theta, K) &= -\log P(y | \theta) + \frac{1}{2} R \log(ND) \\ &= -\log \prod_s P(y_s | \theta) + \frac{1}{2} R \log(ND) \end{aligned}$$

où

N = nombre total de pixels

$$R = K \left( \frac{(D+1)D}{2} + D + 1 \right) - 1, \quad K = \text{Nombre de classes}$$

D=1 pour images en niveaux de gris

D=3 pour images couleur RGB.

# MDL

*Minimum Description Length*

$$MDL(\theta, K) = -\log \prod_s P(y_s | \theta) + \frac{1}{2} R \log(ND)$$

Sachant que les données suivent une mixture de K gaussiennes

$$P(y_s | \theta, K) = \sum_{c=1}^K P(c)N(y_s | \mu_c, \sigma_c)$$

On peut dire que

$$MDL(\theta, K) = -\log \prod_s \left( \sum_{c=1}^K P(c)N(y_s | \mu_c, \sigma_c) \right) + \frac{1}{2} R \log(ND)$$

$$MDL(\theta, K) = -\sum_s \log \left( \sum_{c=1}^K P(c)N(y_s | \mu_c, \sigma_c) \right) + \frac{1}{2} R \log(ND)$$

# MDL

*Minimum Description Length*

$$MDL(\theta, K) = -\log \prod_s P(y_s | \theta) + \frac{1}{2} R \log(ND)$$

Sachant que les données suivent une mixture de K gaussiennes

$$P(y_s | \theta, K) = \sum_{c=1}^K P(c)N(y_s | \mu_c, \sigma_c)$$

On peut dire que

$$MDL(\theta, K) = -\log \prod_s \left( \sum_{c=1}^K P(c)N(y_s | \mu_c, \sigma_c) \right) + \frac{1}{2} R \log(ND)$$

$$MDL(\theta, K) = -\sum_s \log \left( \sum_{c=1}^K P(c)N(y_s | \mu_c, \sigma_c) \right) + \frac{1}{2} R \log(ND)$$

## Estimer le nombre de classes $K$ à l'aide de MDL

1. POUR  $K = 2$  à NB\_CLASSES\_MAX FAIRE
2.  $\theta = \{\mu_c, \sigma_c^2, P(c)\}_{c=1 \dots K}$   $\neg$  Segmentation en  $K$  classes avec *EM*
3.  $MDL[K] = -\sum_s \log \left( \sum_{c=1}^K P(c) N(y_s | \mu_c, \sigma_c) \right) + \frac{1}{2} R \log(ND)$
4.  $k_{optimal} = \arg \min_k MDL[k]$
5. RETOUR  $k_{optimal}$

J. Rissanen, « Universal Prior for Integers and Estimation by Minimum DescriptionLength » Annals of Statistics, vol. 11, no. 2, pp. 417-431, 1983.

C. Bouman, « CLUSTER: An Unsupervised Algorithm for Modeling Gaussian Mixtures », 2005  
Appendice A

# Exemple MDL



K= 2



K= 3



K= 4



K= 5



K= 12



K: 12	MDL: 2068.578461
K: 11	MDL: 2049.159599
K: 10	MDL: 2029.588927
K: 9	MDL: 2004.600938
K: 8	MDL: 1985.956678
K: 7	MDL: 1970.612420
K: 6	MDL: 1953.424010
<b>K: 5</b>	<b>MDL: 1909.510934</b>
K: 4	MDL: 1917.610814
K: 3	MDL: 1933.961104
K: 2	MDL: 1940.469298
K: 1	MDL: 2151.018760

# En résumé...

Algorithme du seuil	Simple, mais le seuil doit être donné par l'usager ( <b><u>approche supervisée</u></b> ).
Algorithme probabiliste du seuil	Si on connaît la distribution probabiliste de chaque classe, le seuil optimal peut être calculé.
Algorithme des K-Moyennes	Algorithme permettant d'estimer automatiquement la <b><u>moyenne</u></b> de chaque classe (gaussiennes).
Algorithme Soft K-Means	Algorithme permettant d'estimer automatiquement la <b><u>moyenne</u></b> de chaque classe (gaussiennes).
Algorithme EM	Algorithme permettant d'estimer automatiquement la <b><u>moyenne, la variance et la proportion</u></b> de chaque classe (gaussiennes).
Test Gaussien + MDL	Méthodes permettant d'estimer automatiquement le nombre de classes.

# *Mean-Shift*

Commanicu D, Meer P. “**Mean shift: a robust approach toward feature space analysis**”, PAMI 24(5), 2002

**Citations > 5000!**

# Mean-Shift

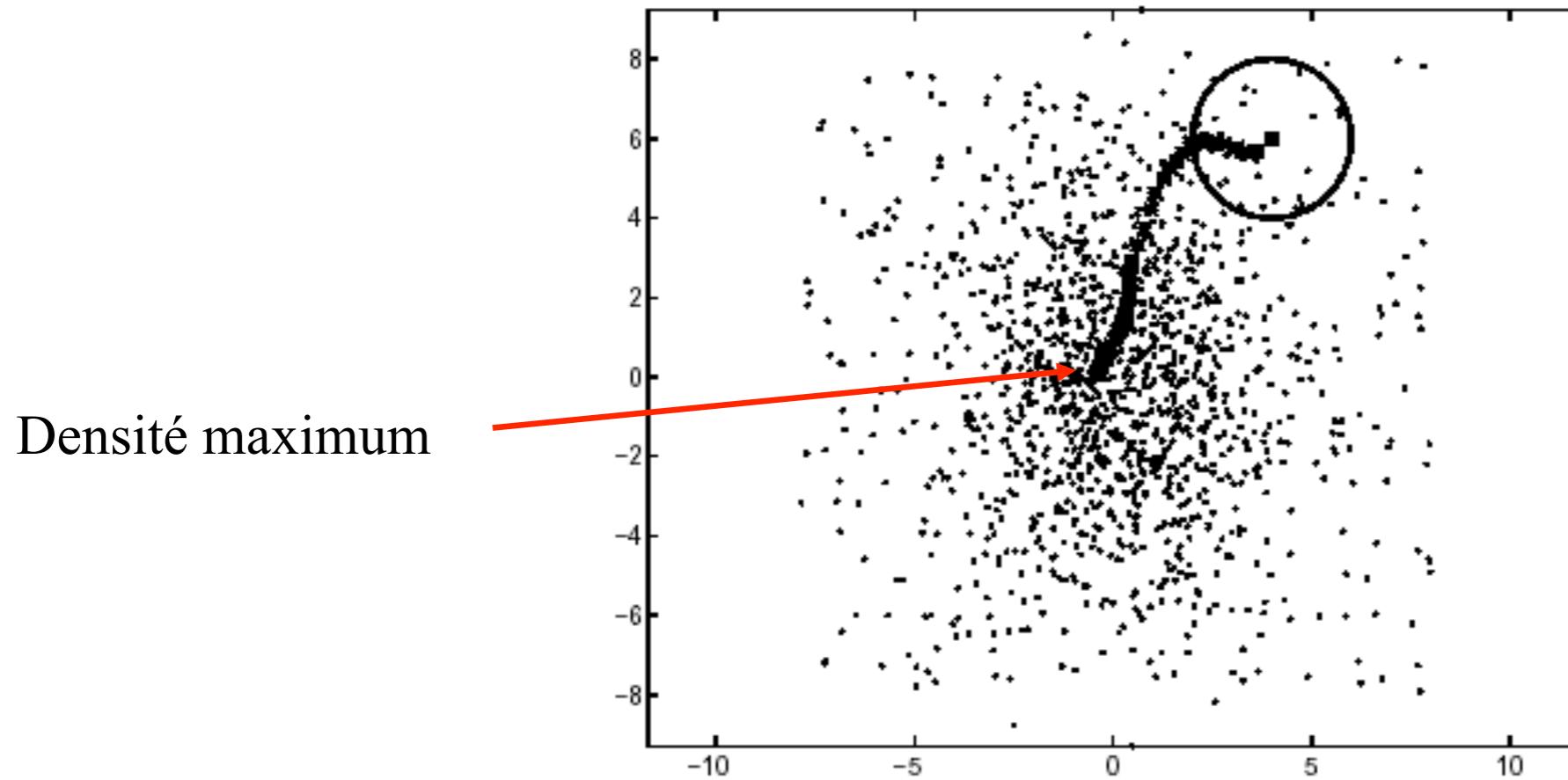
La méthode du Mean-shift en imagerie a été développée par Dorin Comaniciu et Peter Meer à la fin des années 90. Depuis, leur méthode a été utilisée pour résoudre un nombre considérable de problèmes en imagerie. L'application première du Mean-shift est le filtrage avec préservation de contours, mais on peut l'utiliser pour segmenter des images.

Le Mean-shift est une méthode basée sur **l'estimation de densité**

# Mean-Shift (chap. 7)

Pour la méthode du Mean-shift, on cherche la position qui, localement, possède la densité la plus élevée. Pour ce faire, on calcule la densité locale en un point et on déplace itérativement la fenêtre en direction du **gradient de densité maximum**.

Exemple 2D:

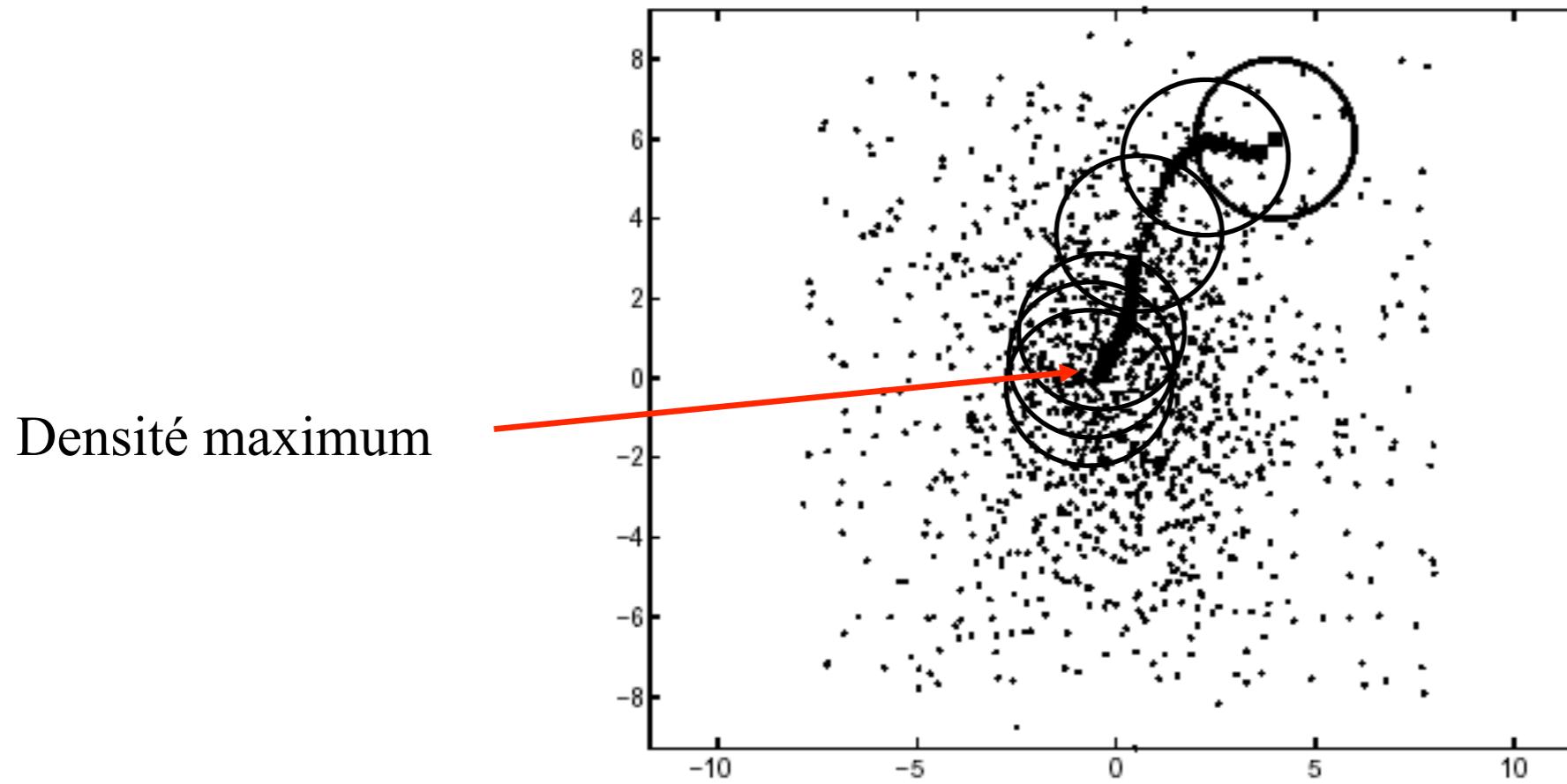


Comaniciu-Meer

# Mean-Shift (chap. 7)

Pour la méthode du Mean-shift, on cherche la position qui, localement, possède la densité la plus élevée. Pour ce faire, on calcule la densité locale en un point et on déplace itérativement la fenêtre en direction du **gradient de densité maximum**.

Exemple 2D:



Comaniciu-Meer

# Parenthèse Estimation de densité

# Estimation de densité

Soit  $Z = \{\vec{x}_1, \vec{x}_2, \dots, \vec{x}_n\}$  où  $\vec{x}_i$  iid  $P(\vec{x})$

Le but d'une méthode d'estimation de densité: retrouver  $P(\vec{x})$  étant donné Z.

**Approche paramétrique** : on connaît la forme de  $P(\vec{x})$  il ne manque que ses paramètres

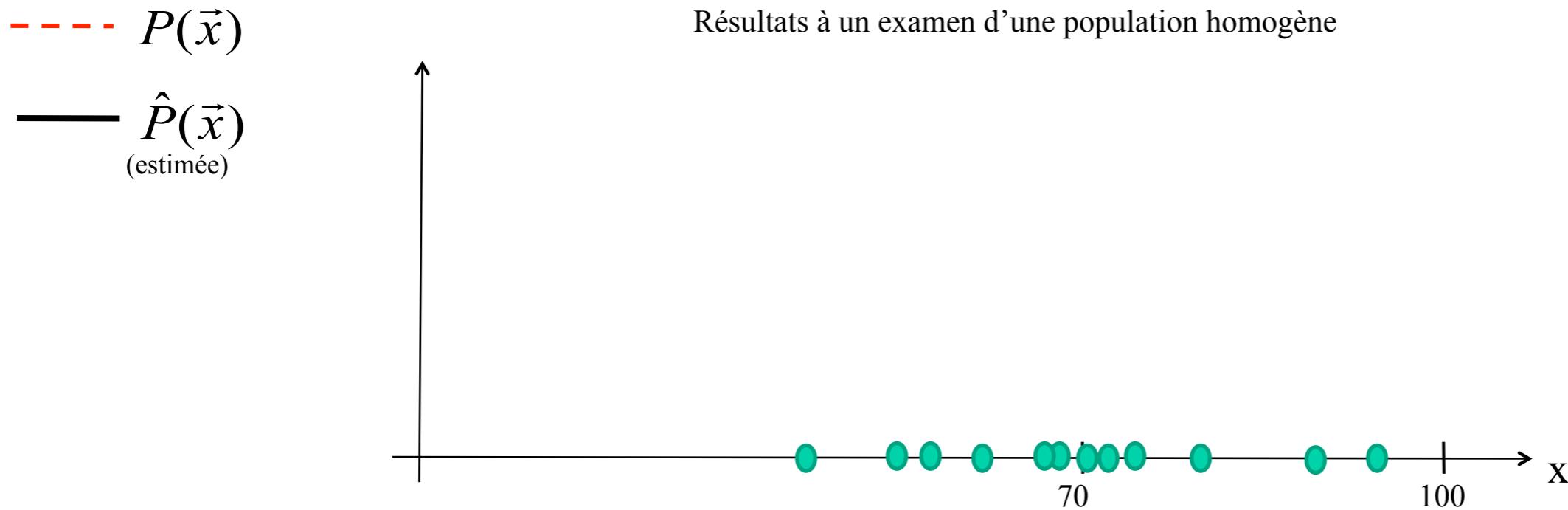
**Approche non-paramétrique** : aucune connaissance a priori quant à la forme de  $P(\vec{x})$

# Estimation de densité

Soit  $Z = \{\vec{x}_1, \vec{x}_2, \dots, \vec{x}_n\}$  où  $\vec{x}_i$  iid  $P(\vec{x})$

Le but d'une méthode d'estimation de densité: retrouver  $P(\vec{x})$  étant donné Z.

**Exemple approche paramétrique:** hypothèse gaussienne

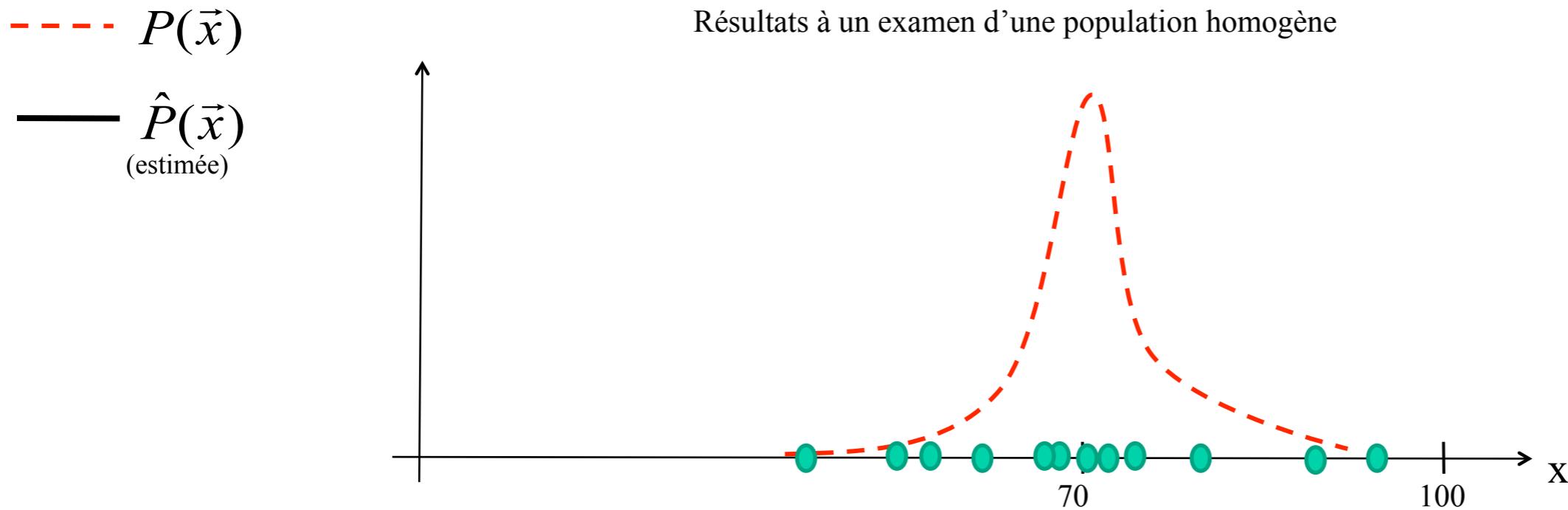


# Estimation de densité

Soit  $Z = \{\vec{x}_1, \vec{x}_2, \dots, \vec{x}_n\}$  où  $\vec{x}_i$  iid  $P(\vec{x})$

Le but d'une méthode d'estimation de densité: retrouver  $P(\vec{x})$  étant donné Z.

**Exemple approche paramétrique:** hypothèse gaussienne

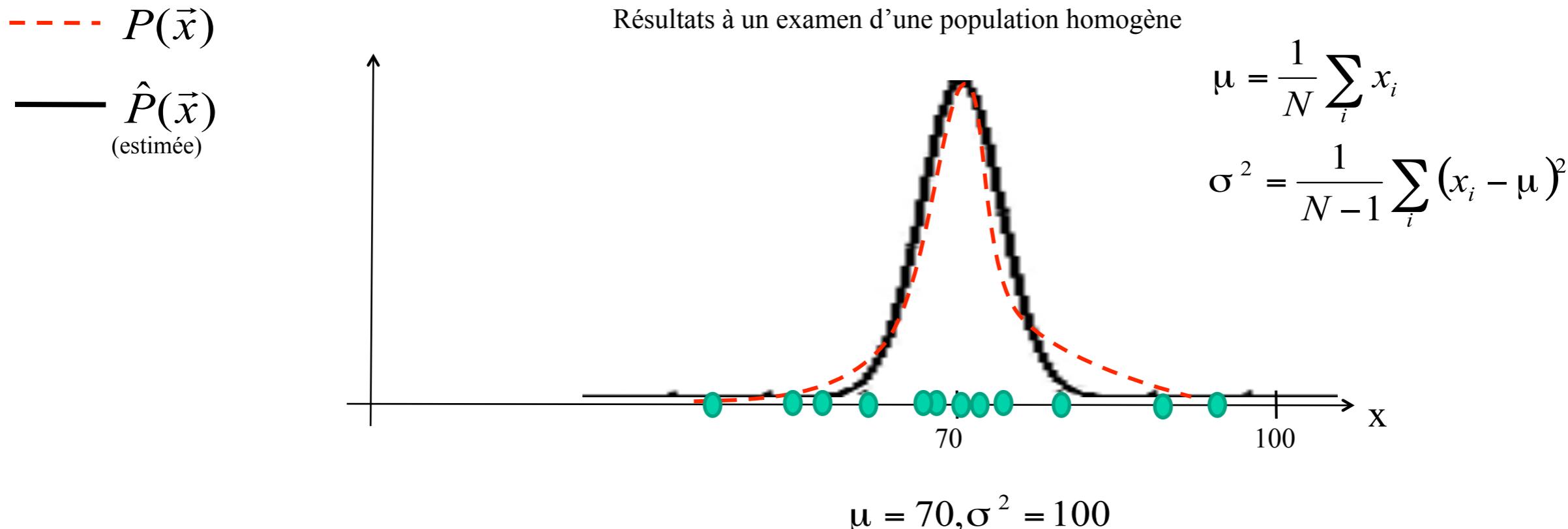


# Estimation de densité

Soit  $Z = \{\vec{x}_1, \vec{x}_2, \dots, \vec{x}_n\}$  où  $\vec{x}_i$  iid  $P(\vec{x})$

Le but d'une méthode d'estimation de densité: retrouver  $P(\vec{x})$  étant donné Z.

**Exemple approche paramétrique:** hypothèse gaussienne

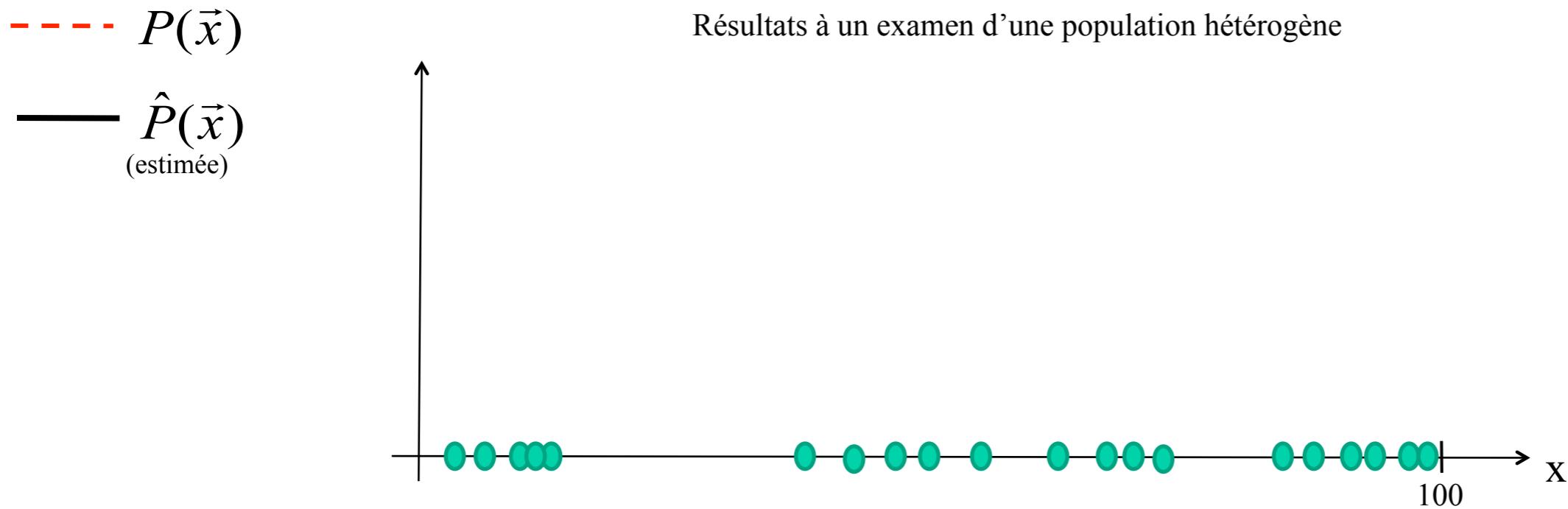


# Estimation de densité

Soit  $Z = \{\vec{x}_1, \vec{x}_2, \dots, \vec{x}_n\}$  où  $\vec{x}_i$  iid  $P(\vec{x})$

Le but d'une méthode d'estimation de densité: retrouver  $P(\vec{x})$  étant donné Z.

**Exemple approche non-paramétrique:** Méthode de Parzen

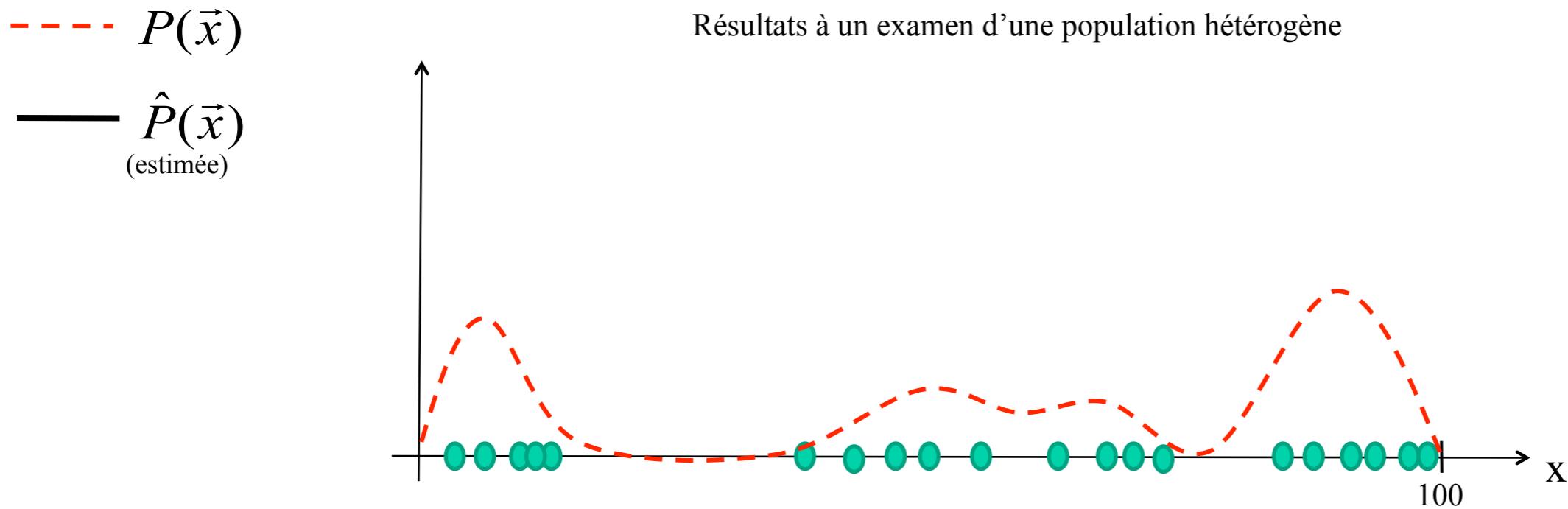


# Estimation de densité

Soit  $Z = \{\vec{x}_1, \vec{x}_2, \dots, \vec{x}_n\}$  où  $\vec{x}_i$  iid  $P(\vec{x})$

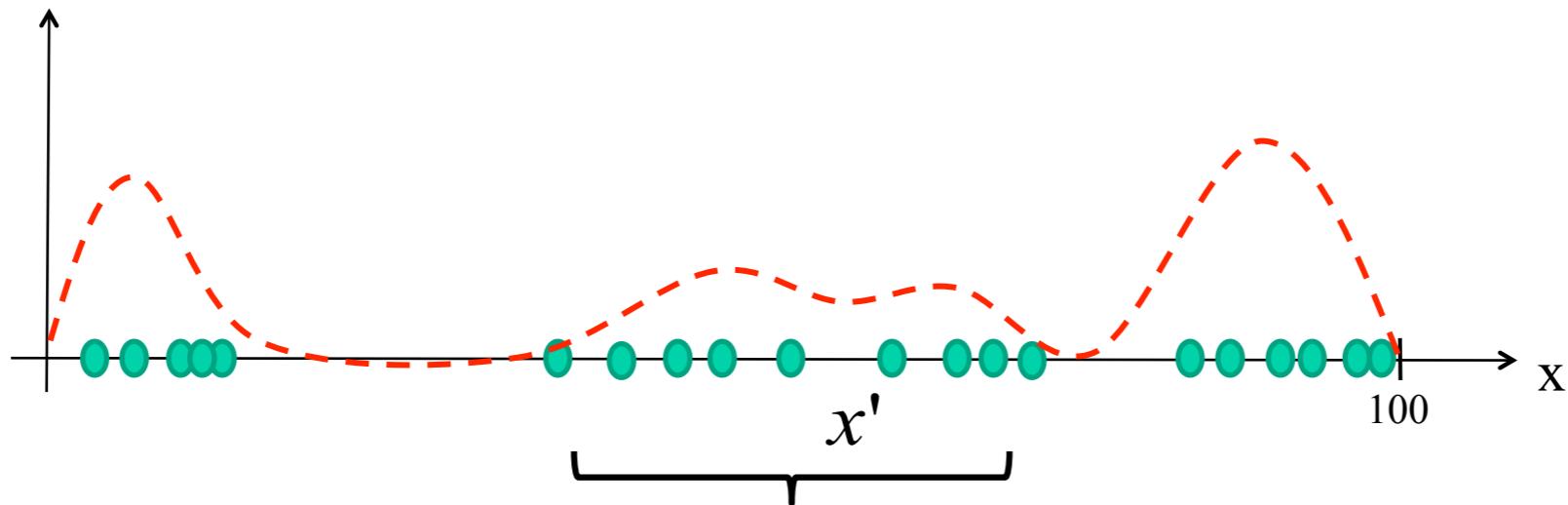
Le but d'une méthode d'estimation de densité: retrouver  $P(\vec{x})$  étant donné Z.

**Exemple approche non-paramétrique:** Méthode de Parzen



# Fenêtres de Parzen

Hypothèse:  $P(\vec{x}) \approx$  densité de points près de  $\vec{x}$

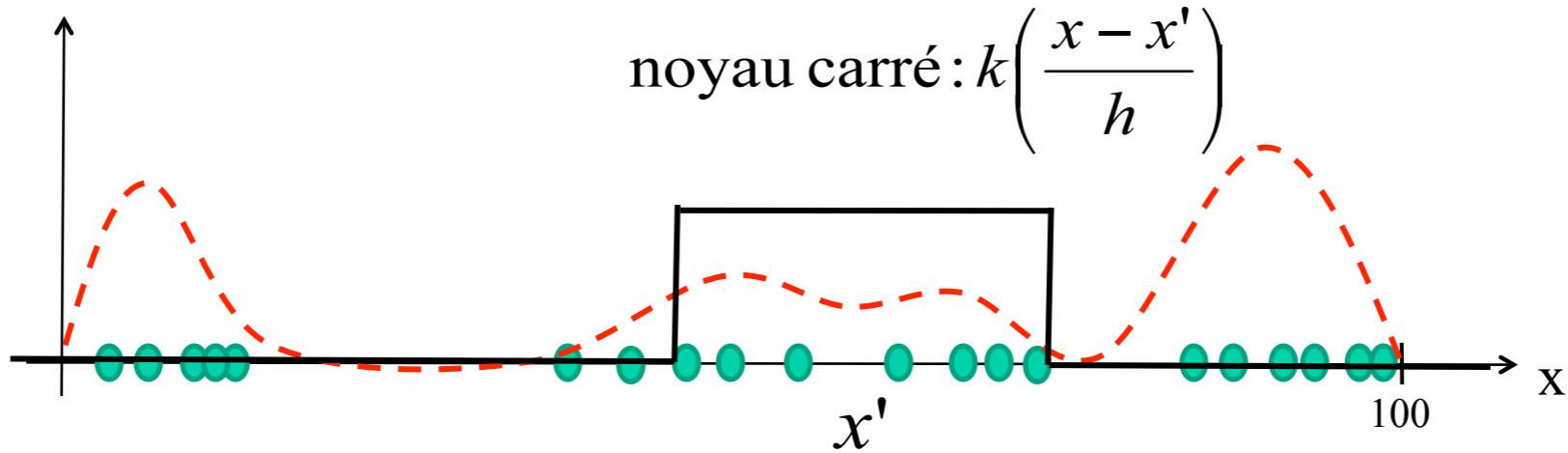


7 points près de  $x' \rightarrow P(x') \approx \frac{7}{20} \times \frac{1}{h}$  → Taille de la fenêtre

ici  $h = 25$

# Fenêtres de Parzen

Hypothèse:  $P(x) \approx$  densité de points près de  $x$



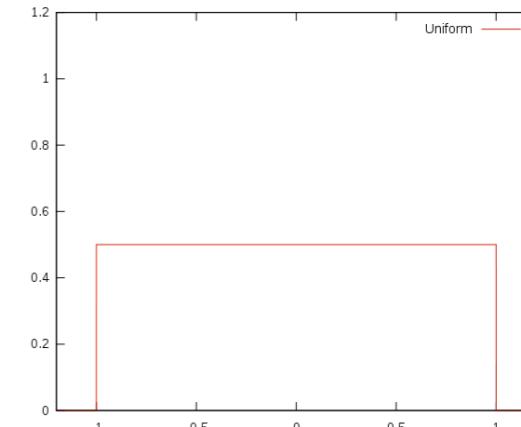
$$k\left(\frac{x - x'}{h}\right) = k(r) = \begin{cases} 1 & \text{si } |r| < 1/2 \\ 0 & \text{sinon} \end{cases}$$

$$P(x) \approx \frac{1}{nh} \sum_{i=1}^n k\left(\frac{x - x_i}{h}\right)$$

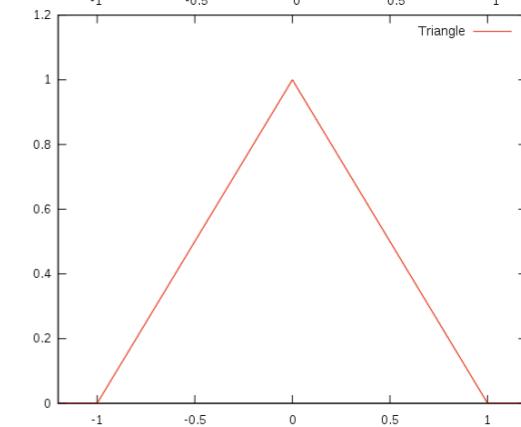
# Fenêtres de Parzen

## Autres noyaux

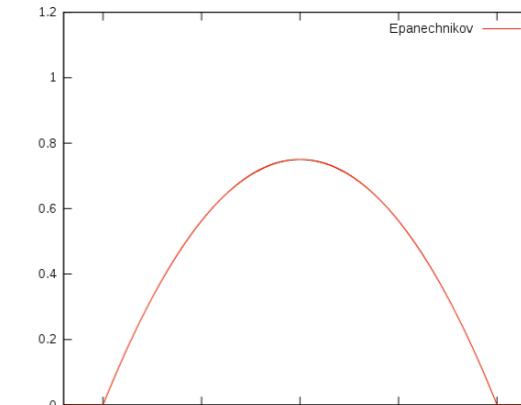
Carré     $k(r) = \begin{cases} 1/2 & \text{si } |r| < 1 \\ 0 & \text{sinon} \end{cases}$



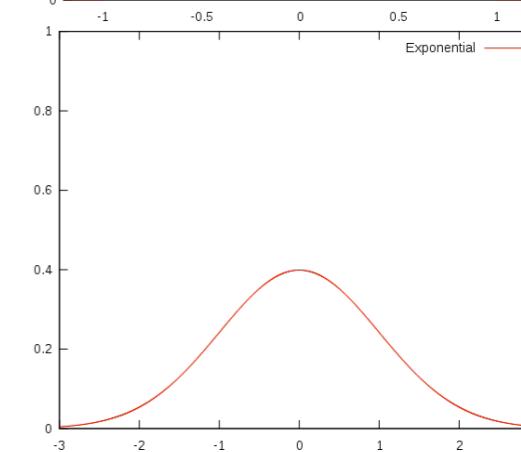
Triangulaire     $k(r) = \begin{cases} 1 - r & \text{si } |r| < 1 \\ 0 & \text{sinon} \end{cases}$



Epanechnikov     $k(r) = \begin{cases} 3/4(1 - r^2) & \text{si } r < 1 \\ 0 & \text{sinon} \end{cases}$



Gaussien     $k(r) = \frac{1}{\sqrt{2\pi}} \exp(-r^2)$



# Fenêtres de Parzen

## Propriété

Propriétés importantes d'un noyau:

$$\int_{-\infty}^{\infty} k(r)dr = 1$$
$$k(-r) = k(r) \quad \forall r$$

# Fenêtres de Parzen

## Propriété

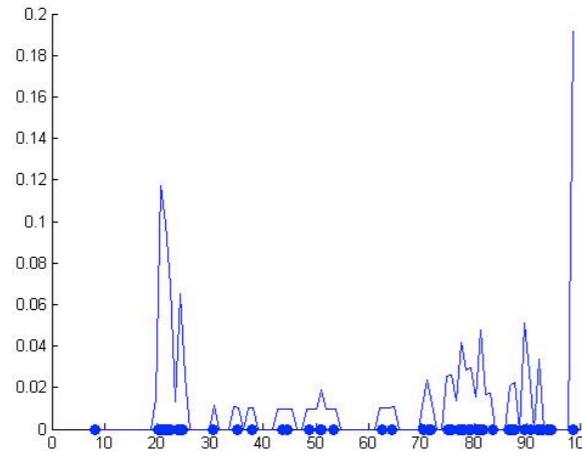
Propriétés importantes d'un noyau:

$$\int_{-\infty}^{\infty} k(r)dr = 1$$
$$k(-r) = k(r) \quad \forall r$$

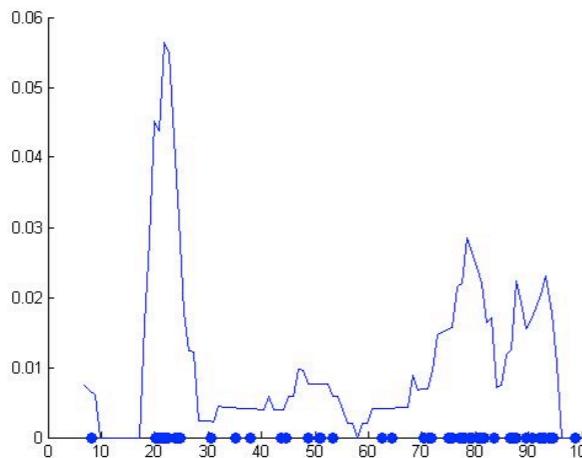
$$P(\vec{x}) \approx \frac{1}{nh^d} \sum_{i=1}^n k\left(\frac{\vec{x} - \vec{x}_i}{h}\right) = \frac{1}{nh^d} \sum_{i=1}^n k\left(\frac{\|\vec{x} - \vec{x}_i\|}{h}\right)$$

# Résultats 1D

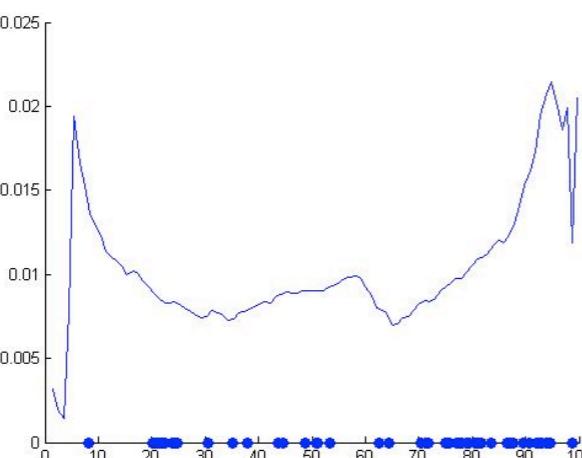
Noyau carré



$h$  faible

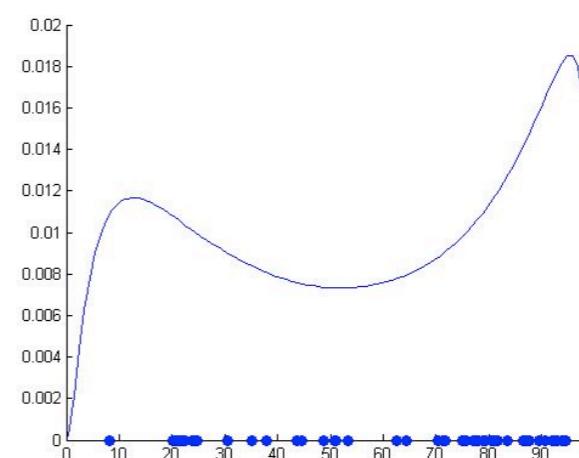
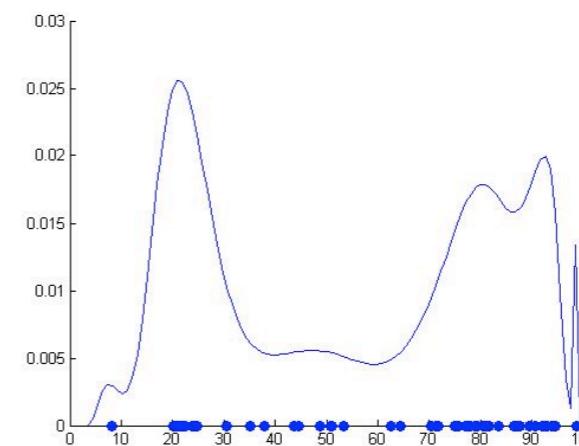
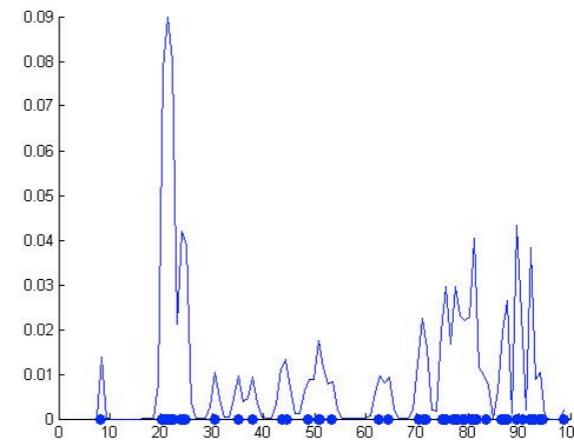


$h$  moyen



$h$  élevé

Noyau gaussien



# Fin de la parenthèse

# Mean-Shift

Le but de Mean-shift est de modéliser le **gradient de densité**

$$\nabla P(\vec{x}) = \frac{d}{d\vec{x}} \left( \frac{1}{nh^d} \sum_{i=1}^n k\left(\frac{\vec{x} - \vec{x}_i}{h}\right) \right)$$

# Mean-Shift

Le but de Mean-shift est de modéliser le gradient de densité

$$\nabla P(\vec{x}) = \frac{d}{d\vec{x}} \left( \frac{1}{nh^d} \sum_{i=1}^n k\left(\frac{\vec{x} - \vec{x}_i}{h}\right) \right)$$

Pour simplifier les calculs, nous utiliserons  $k\left(\frac{\|\vec{x} - \vec{x}_i\|^2}{h^2}\right)$

$$\begin{aligned} \nabla P(\vec{x}) &= \frac{d}{d\vec{x}} \left( \frac{1}{nh^d} \sum_{i=1}^n k'\left(\frac{\|\vec{x} - \vec{x}_i\|^2}{h^2}\right) \right) \\ &= \frac{2}{nh^{d+2}} \sum_{i=1}^n k'\left(\frac{\|\vec{x} - \vec{x}_i\|^2}{h^2}\right) (\vec{x} - \vec{x}_i) \end{aligned}$$

# Mean-Shift

Le but de Mean-shift est de modéliser le **gradient de densité**

$$\nabla P(\vec{x}) = \frac{2}{nh^{d+2}} \sum_{i=1}^n k' \left( \frac{\|\vec{x} - \vec{x}_i\|^2}{h^2} \right) (\vec{x} - \vec{x}_i)$$

# Mean-Shift

Le but de Mean-shift est de modéliser le **gradient de densité**

$$\nabla P(\vec{x}) = \frac{2}{nh^{d+2}} \sum_{i=1}^n k' \left( \frac{\|\vec{x} - \vec{x}_i\|^2}{h^2} \right) (\vec{x} - \vec{x}_i)$$

En posant  $g_i = -k' \left( \frac{\|\vec{x} - \vec{x}_i\|^2}{h^2} \right)$

$$\begin{aligned}\nabla P(\vec{x}) &= \frac{2}{nh^{d+2}} \sum_{i=1}^n g_i (\vec{x}_i - \vec{x}) \\ &= \frac{2}{nh^{d+2}} \sum_{i=1}^n g_i \vec{x}_i - \vec{x} \sum_{i=1}^n g_i \\ &= \frac{2}{nh^{d+2}} \sum_{i=1}^n g_i \left( \frac{\sum_{i=1}^n g_i \vec{x}_i}{\sum_{i=1}^n g_i} - \vec{x} \right)\end{aligned}$$

# Mean-Shift

Le but de Mean-shift est de modéliser le gradient de densité

$$\nabla P(\vec{x}) = \frac{2}{nh^{d+2}} \sum_{i=1}^n k' \left( \frac{\|\vec{x} - \vec{x}_i\|^2}{h^2} \right) (\vec{x} - \vec{x}_i)$$

En posant  $g_i = -k' \left( \frac{\|\vec{x} - \vec{x}_i\|^2}{h^2} \right)$

$$\nabla P(\vec{x}) = \frac{2}{nh^{d+2}} \sum_{i=1}^n g_i (\vec{x}_i - \vec{x})$$

$$= \frac{2}{nh^{d+2}} \sum_{i=1}^n g_i \vec{x}_i - \vec{x} \sum_{i=1}^n g_i$$

Coefficient

$$= \frac{2}{nh^{d+2}} \sum_{i=1}^n g_i \left( \frac{\sum_{i=1}^n g_i \vec{x}_i}{\sum_{i=1}^n g_i} - \vec{x} \right)$$

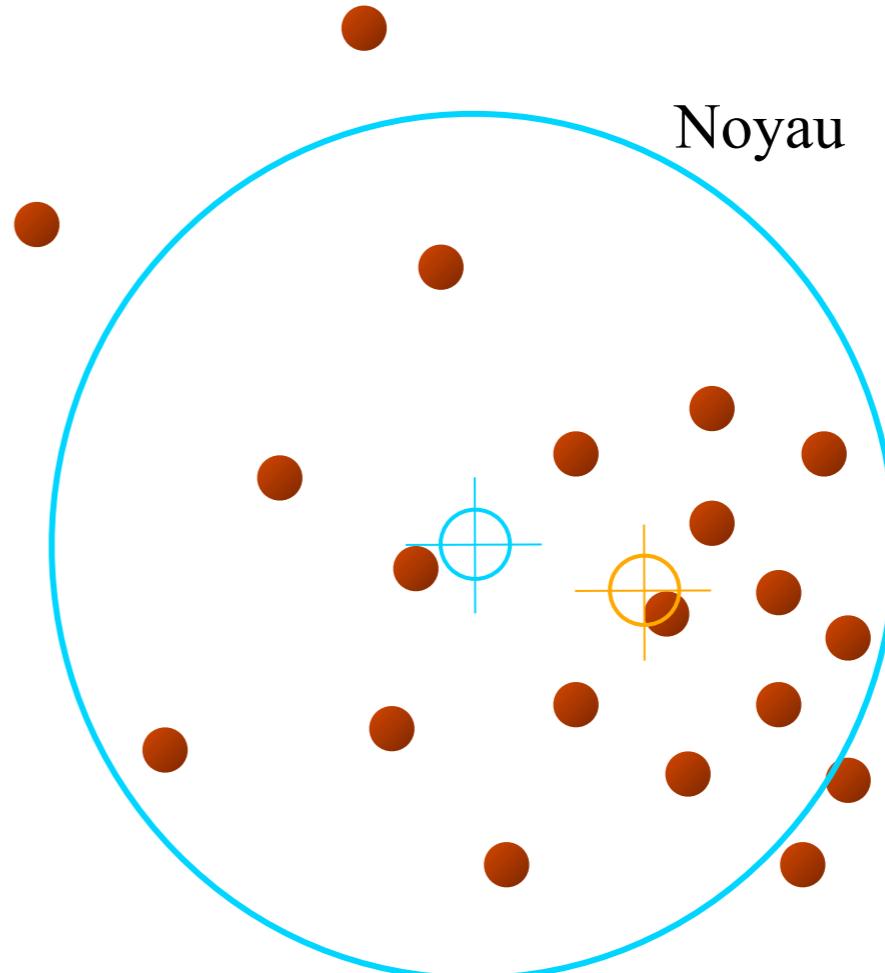
Vecteur mean-shift  $m(x)$   
tend vers 0 près du maximum

# Mean-Shift

Vecteur de Mean-shift

$$m(\vec{x}) = \frac{\sum_{i=1}^n g_i \vec{x}_i}{\sum_{i=1}^n g_i} - \vec{x}$$

$$g_i = -k' \left( \frac{\|\vec{x} - \vec{x}_i\|^2}{h^2} \right)$$



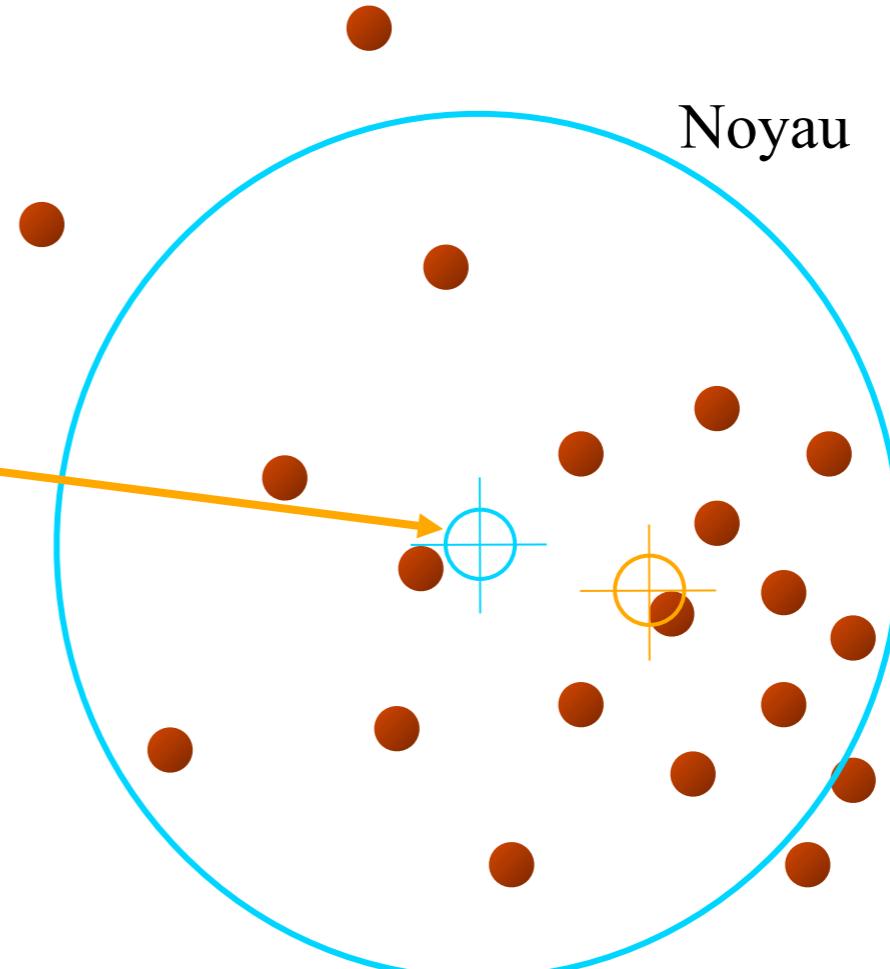
h est le paramètre d'échelle (rayon)

# Mean-Shift

Vecteur de Mean-shift

$$m(\vec{x}) = \frac{\sum_{i=1}^n g_i \vec{x}_i}{\sum_{i=1}^n g_i} - \vec{x}$$

$$g_i = -k' \left( \frac{\|\vec{x} - \vec{x}_i\|^2}{h^2} \right)$$



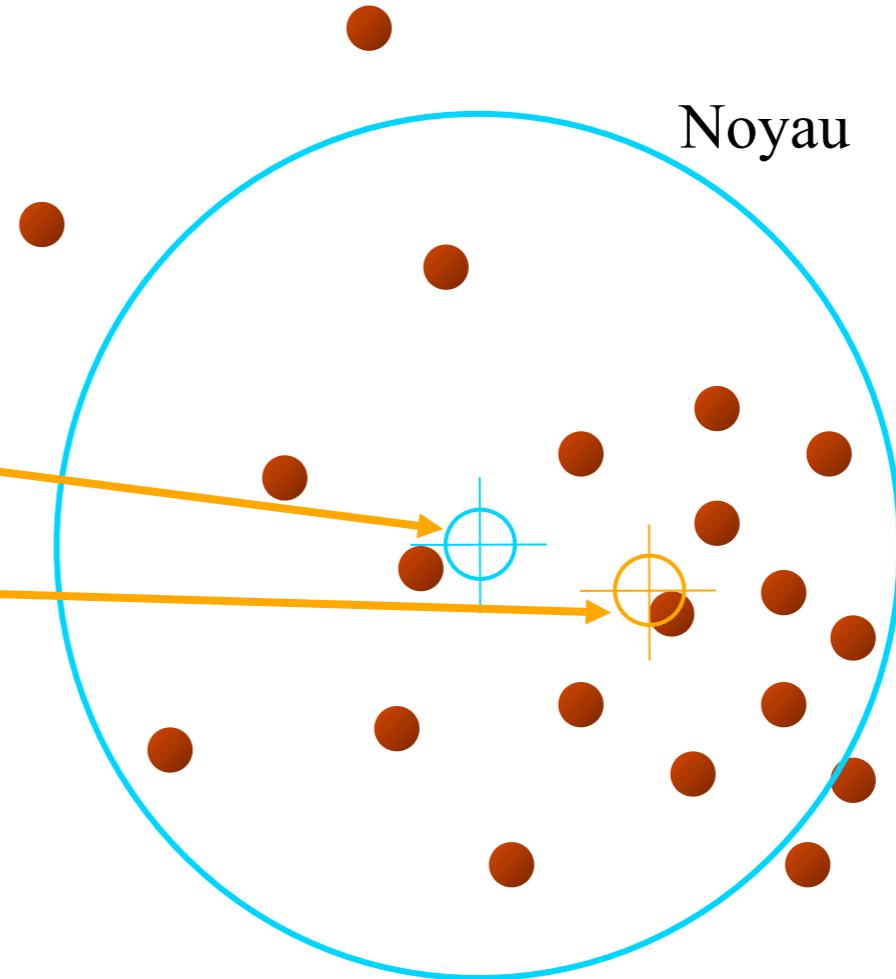
h est le paramètre d'échelle (rayon)

# Mean-Shift

Vecteur de Mean-shift

$$m(\vec{x}) = \frac{\sum_{i=1}^n g_i \vec{x}_i}{\sum_{i=1}^n g_i} - \vec{x}$$

$$g_i = -k' \left( \frac{\|\vec{x} - \vec{x}_i\|^2}{h^2} \right)$$



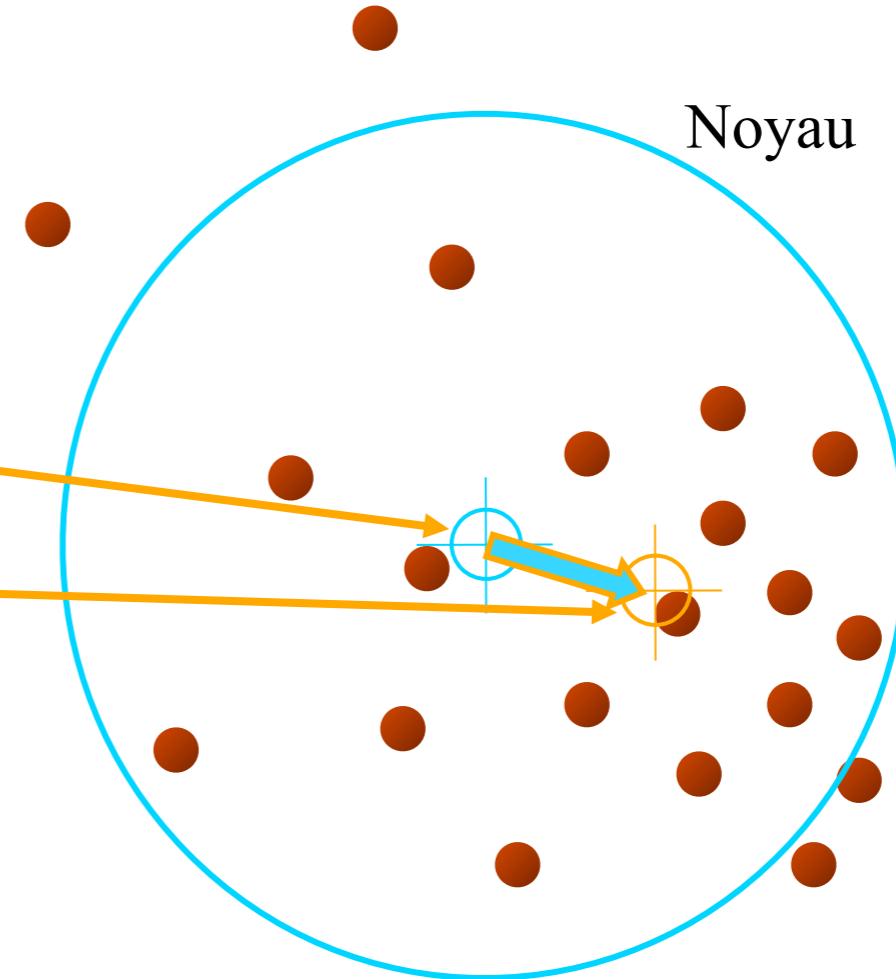
$h$  est le paramètre d'échelle (rayon)

# Mean-Shift

Vecteur de Mean-shift

$$m(\vec{x}) = \frac{\sum_{i=1}^n g_i \vec{x}_i}{\sum_{i=1}^n g_i} - \vec{x}$$

$$g_i = -k' \left( \frac{\|\vec{x} - \vec{x}_i\|^2}{h^2} \right)$$



$h$  est le paramètre d'échelle (rayon)

# Mean-Shift

Vecteur de Mean-shift

$$m(\vec{x}) = \frac{\sum_{i=1}^n g_i \vec{x}_i}{\sum_{i=1}^n g_i} - \vec{x}$$

$$g_i = -k' \left( \frac{\|\vec{x} - \vec{x}_i\|^2}{h^2} \right)$$

Noyaux usuels pour mean-shift

Epanechnikov  $g_i = \begin{cases} 1 & \text{si } \frac{\|\vec{x} - \vec{x}_i\|^2}{h^2} < 1 \\ 0 & \text{sinon} \end{cases}$

Gaussien  $g_i = \frac{1}{\sqrt{2\pi}} \exp \left( -\frac{\|\vec{x} - \vec{x}_i\|^2}{h^2} \right)$

Théorème :

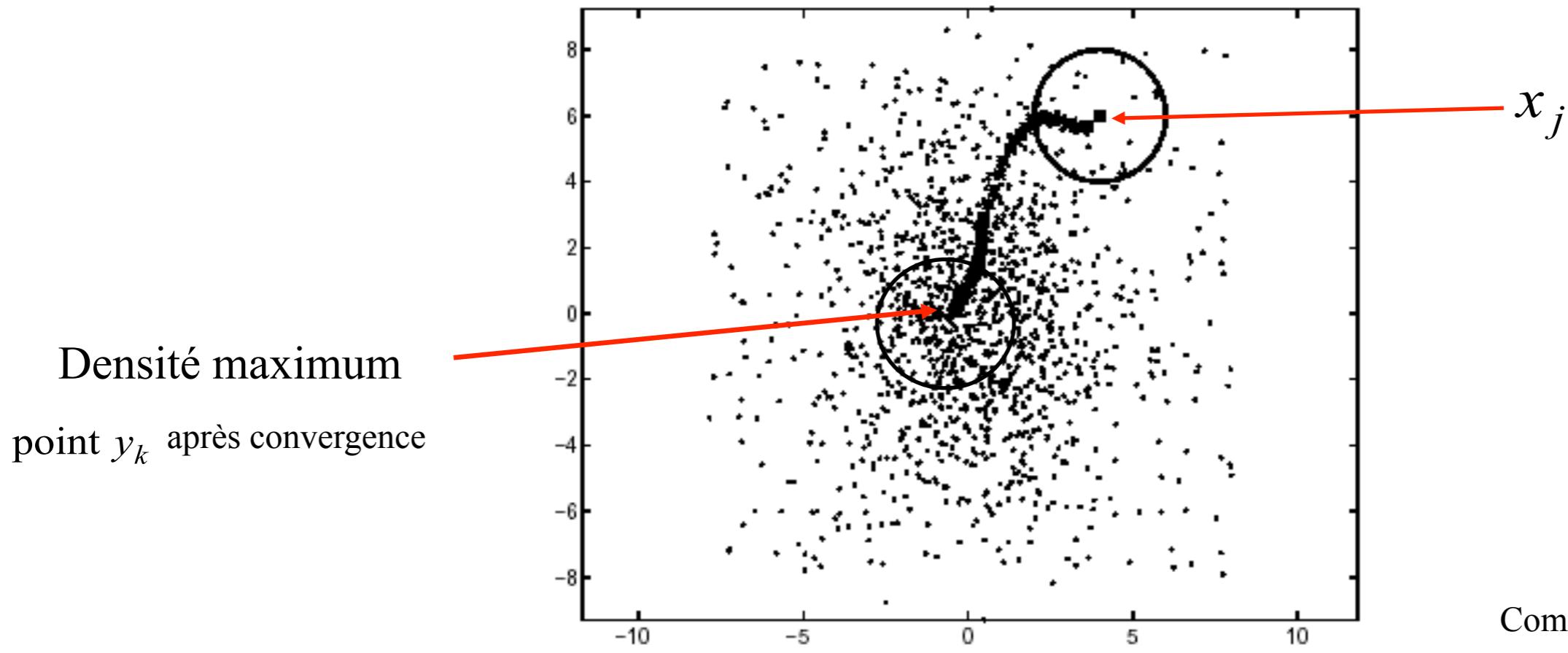
Si le noyau  $k$  est convexe et monotone décroissant, l'algorithme convergera et  $P(\vec{x})$  augmentera de façon monotone.

(voir article pour plus de détails)

# Mean-Shift

À l'aide du noyau d'Epanechnikov, on peut atteindre le point ayant une densité maximum locale à l'aide d'un algorithme très simple.

1.  $k = 1, y_k = x_j$
2. centrer la fenêtre  $S_h$  sur  $y_k$
3.  $y_{k+1} = \frac{1}{N_k} \sum_{\substack{i=0 \\ x_i \in S_h}}^{N-1} x_i , k = k + 1$
4. retour à 2 si  $y_{k+1} \neq y_k$



1. On choisit un point de départ  $x_i$
2.  $x_{i+1} \leftarrow$  la moyenne de tous les points situés dans un rayon  $R$  de  $x_i$ . Le vecteur  $x_{i+1} - x_i$  est alors appelé **Mean-Shift**.
3. On répète l'étape précédente tant que  $x_i$  bouge suffisamment. On s'arrête donc quand on a convergé vers un maximum local de la fonction de densité,  $x_{\text{final}}$ .
4. On étiquette alors tous les points à côté desquels on est passé durant l'évolution (dans un rayon  $R$ ) par la même étiquette
5. On recommence à la première étape avec un nouveau point, tant qu'il y a des points que l'on n'a pas encore étiquetés.

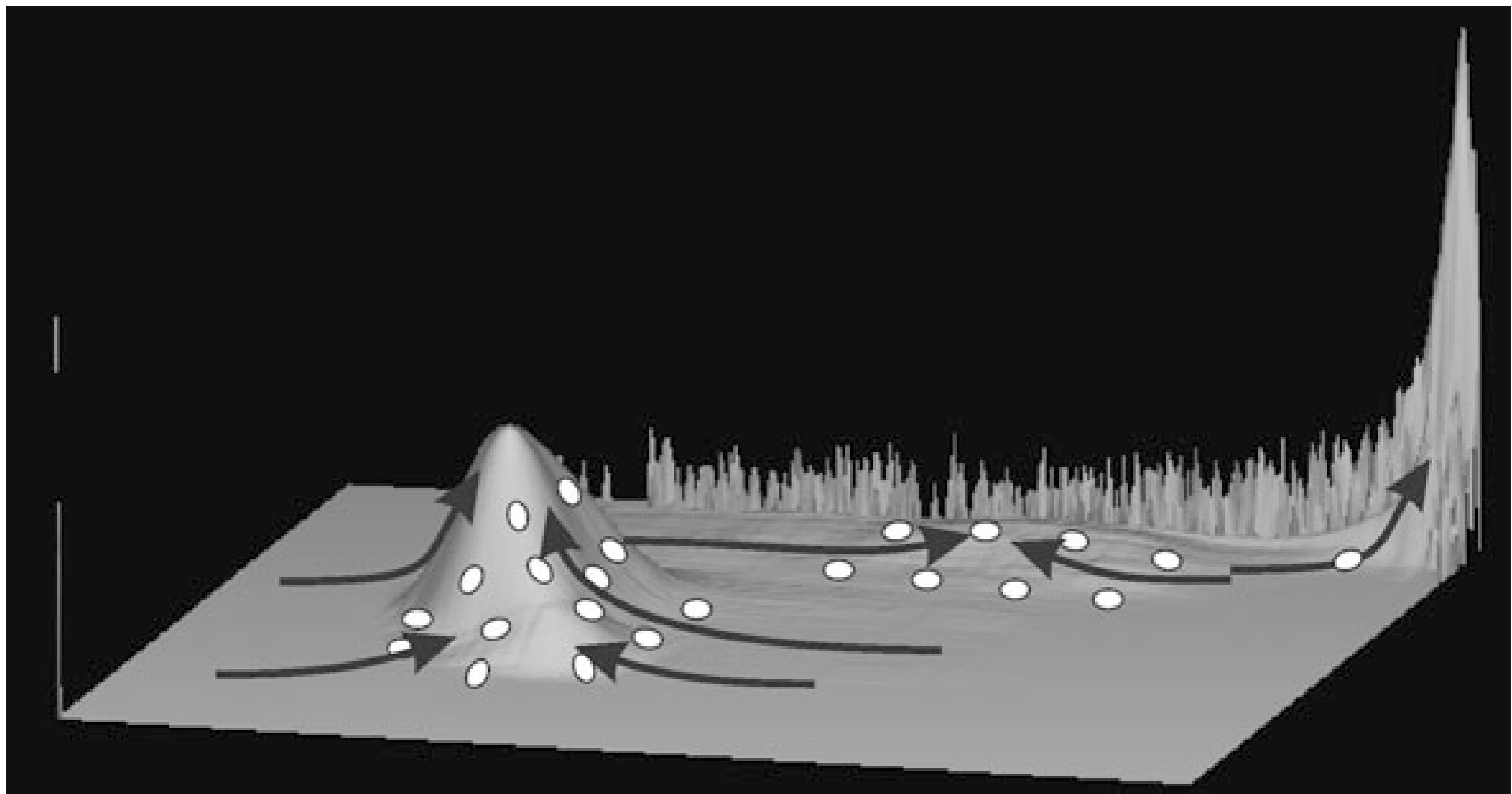
L'étiquette qu'on attribue à l'étape 4 est choisie avec les règles suivantes :

- si  $x_{\text{final}}$  (le point vers lequel on a convergé) est à une distance inférieure à  $R/2$  d'un point limite précédent (qu'il faut donc enregistrer au fur et à mesure), alors on donne la même étiquette que celle du cluster associé à ce point limite.
- sinon, on choisit une nouvelle étiquette

Une description "illustrée" de cet algorithme peut être trouvée ici

[http://www.enseignement.polytechnique.fr/informatique/INF562/Slides/A7\\_Slides5\\_MeanShiftClustering.pdf](http://www.enseignement.polytechnique.fr/informatique/INF562/Slides/A7_Slides5_MeanShiftClustering.pdf)

# Mean-shift



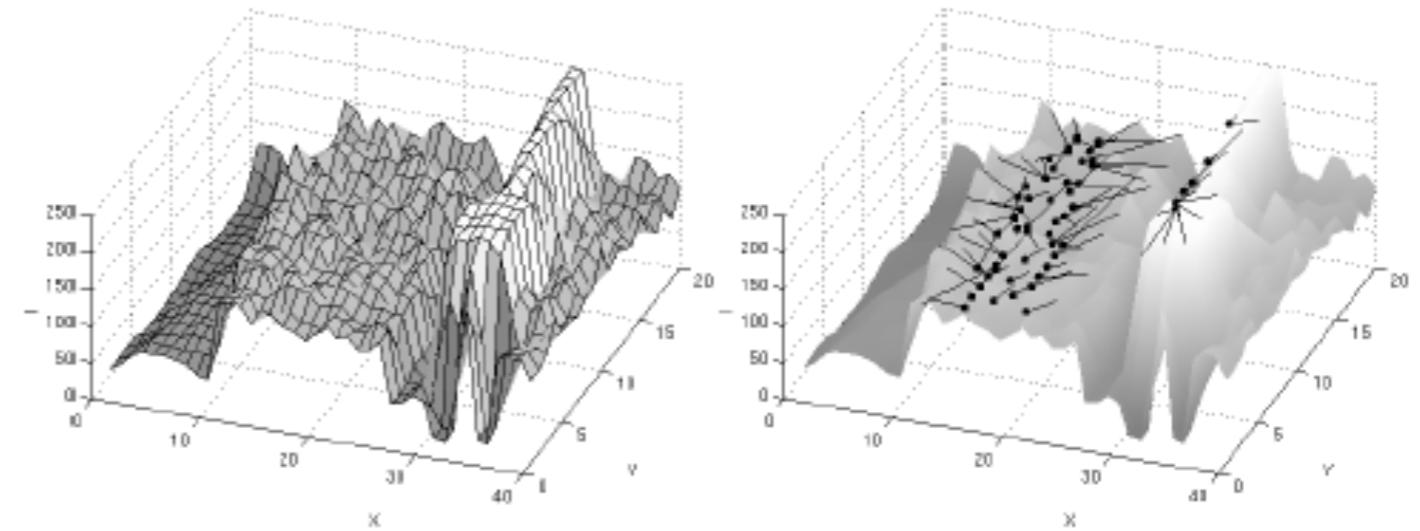
# Mean-Shift

Lors du filtrage d'une image en niveau de gris  $I(x,y)$ , chaque pixel est considéré comme un point dans un espace 3D. Le pixel  $I(i,j)$  correspond à un point 3D dont les coordonnées sont  $(i, j, I(i,j))$ .

Exemple



Avec l'algorithme du Mean-shift, les points 3D se décalent vers la densité maximale locale

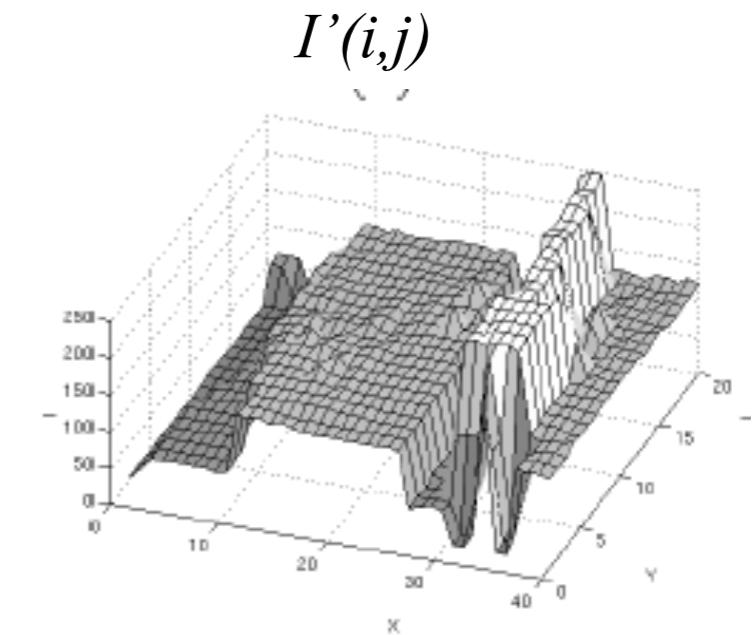
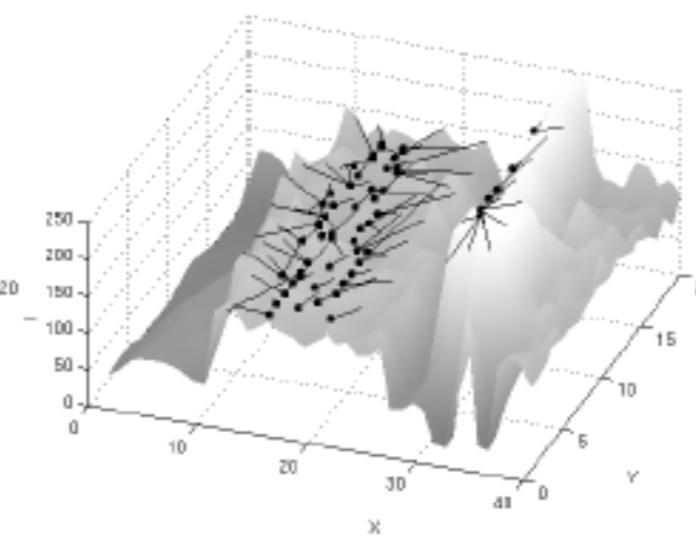
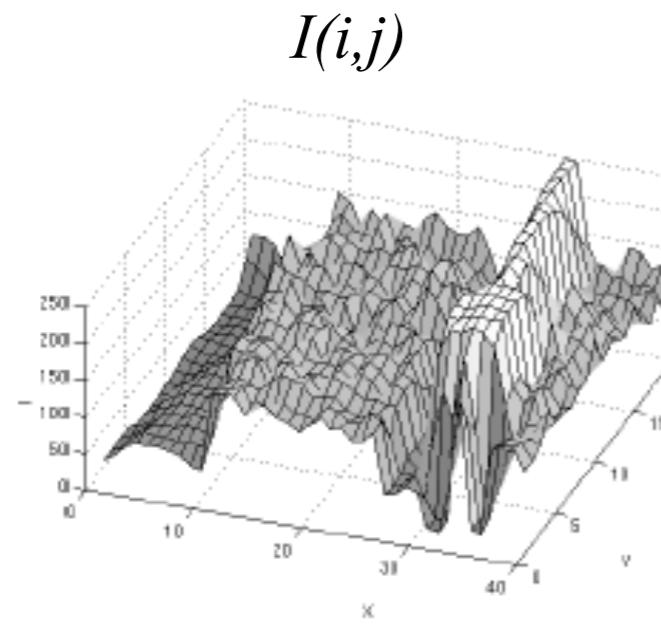


Comaniciu-Meer

# Mean-Shift

Lors du filtrage par mean-shift, pour chaque pixel  $I(i,j)$  de l'image

1.  $k = 1, y_k = (i, j, I(i, j))$
2. centrer la fenêtre 3D  $S_h$  sur  $y_k$
3.  $y_{k+1} = \frac{1}{N_k} \sum_{\substack{i=0 \\ x_i \in S_h}}^{N-1} x_i, k = k + 1$
4. retour à 2 si  $y_{k+1} \neq y_k$
5.  $I'(i, j) = y_k$



Comaniciu-Meer

# Mean-Shift

$S_h$  peut avoir différentes tailles et différentes formes

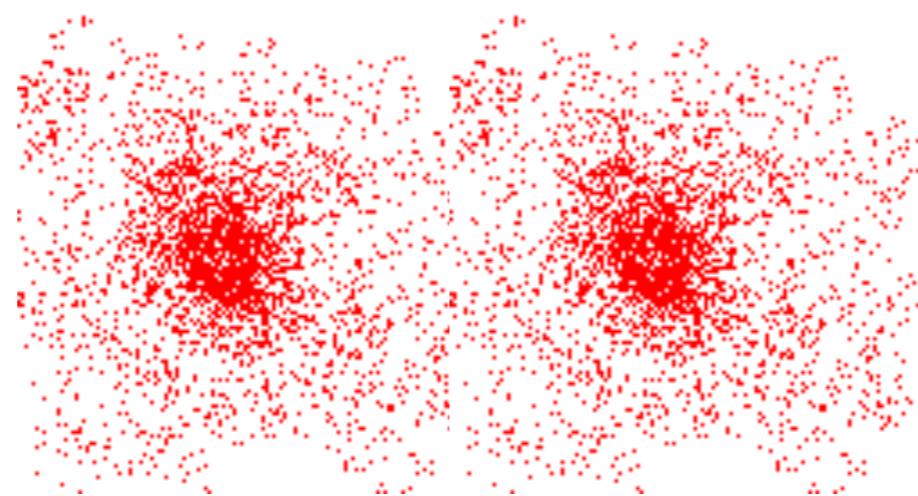


Comaniciu-Meer

# Mean-Shift

*Means-Shift* peut également servir d'outil de segmentation

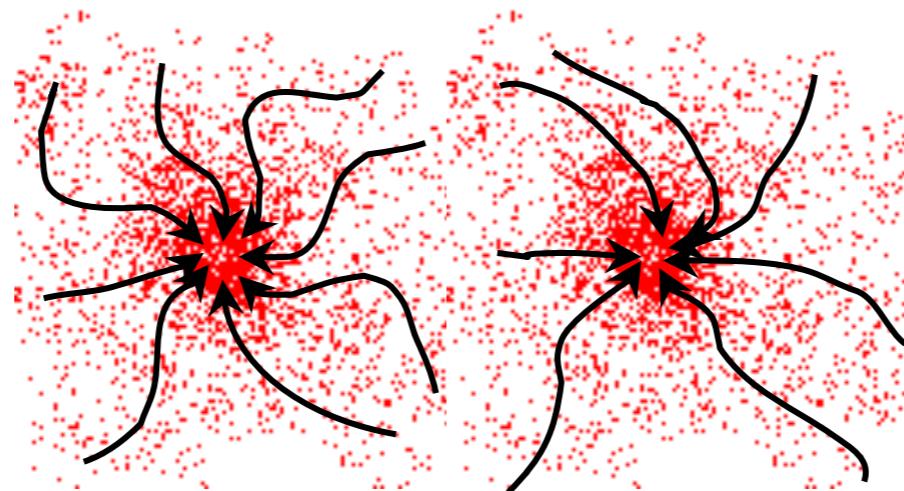
Idée : regrouper ensemble les points convergeants au même endroit.



# Mean-Shift

*Means-Shift* peut également servir d'outil de segmentation

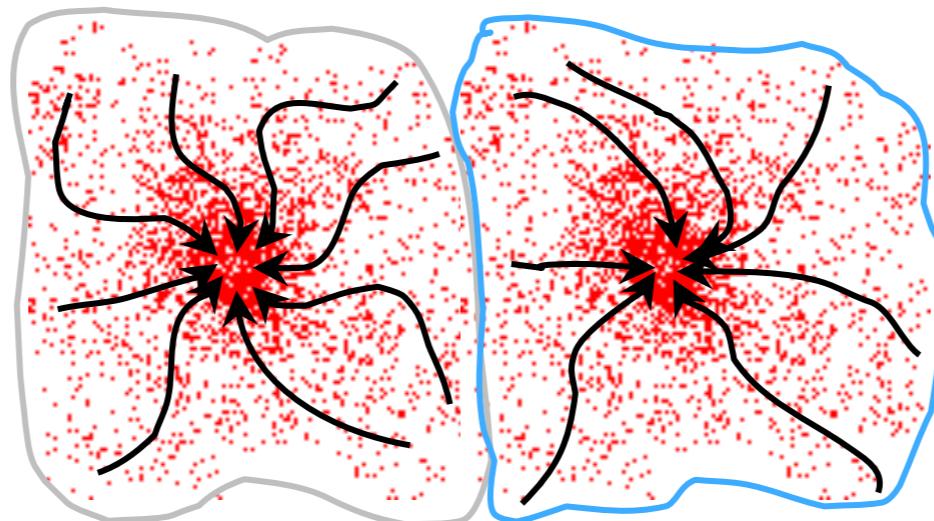
Idée : regrouper ensemble les points convergeants au même endroit.



# Mean-Shift

*Means-Shift* peut également servir d'outil de segmentation

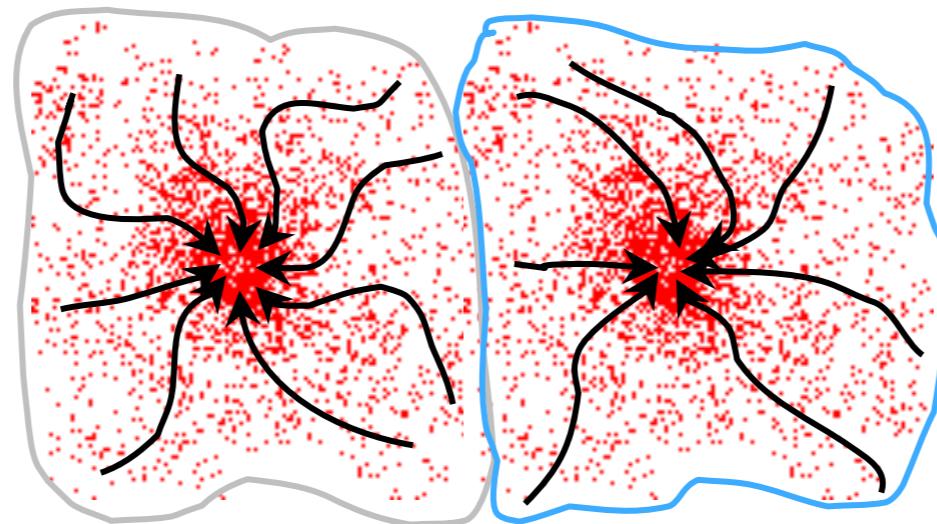
Idée : regrouper ensemble les points convergeants au même endroit.



# Mean-Shift

*Means-Shift* peut également servir d'outil de segmentation

Idée : regrouper ensemble les points convergeants au même endroit.

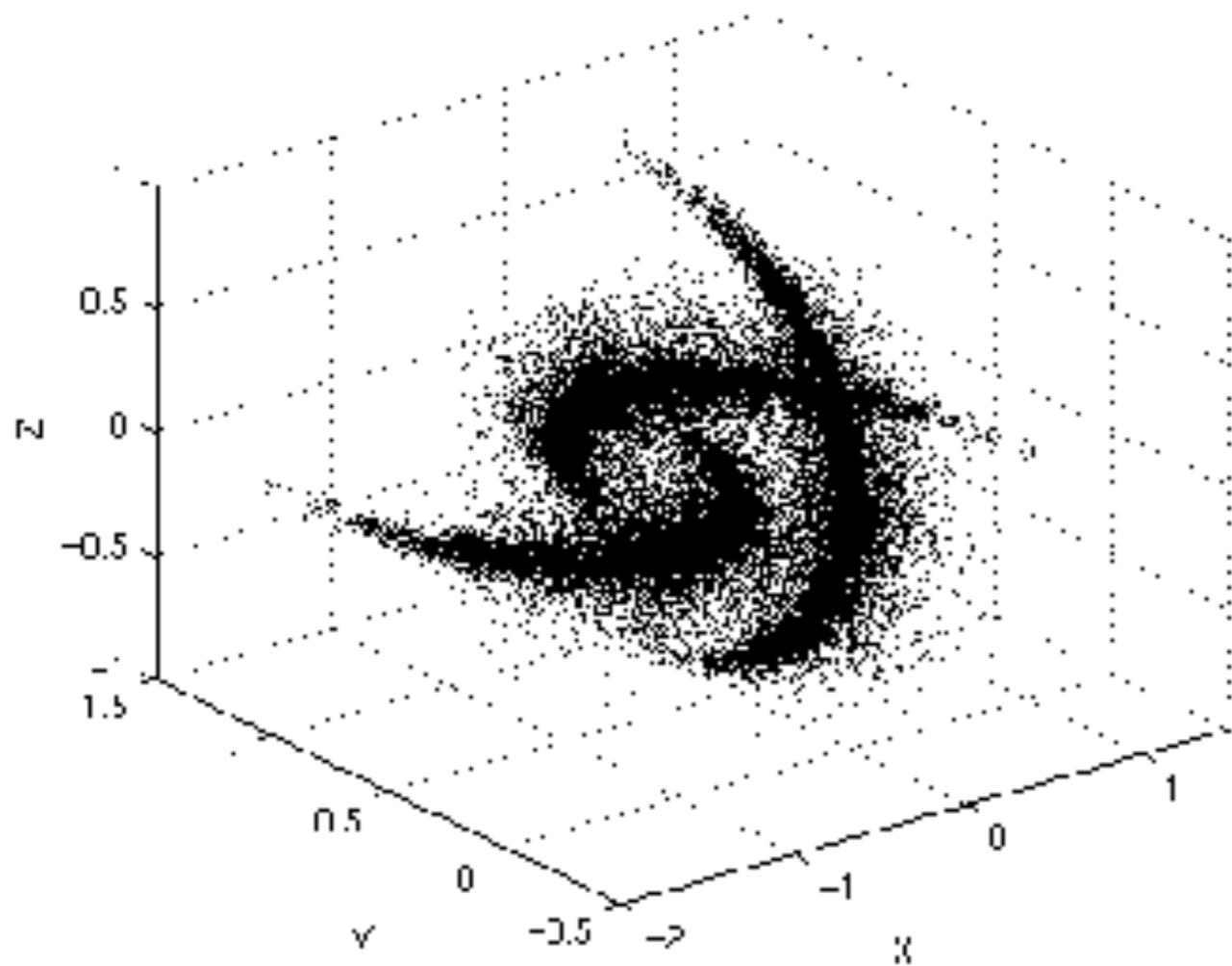


Dans cet exemple, les points convergent vers 2 endroits, il y a donc **2 classes**

# Mean-Shift

*Means-Shift* peut également servir d'outil de segmentation

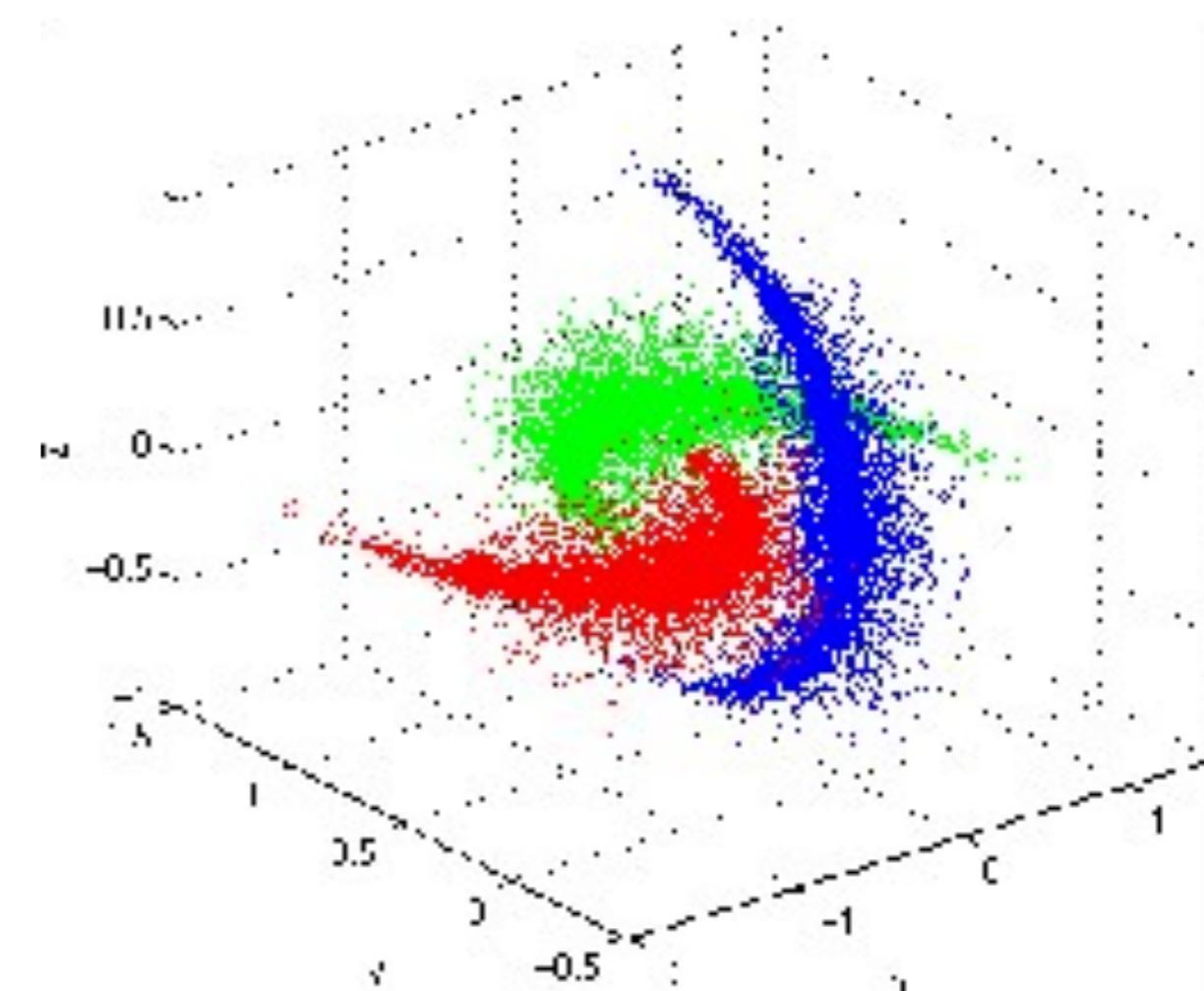
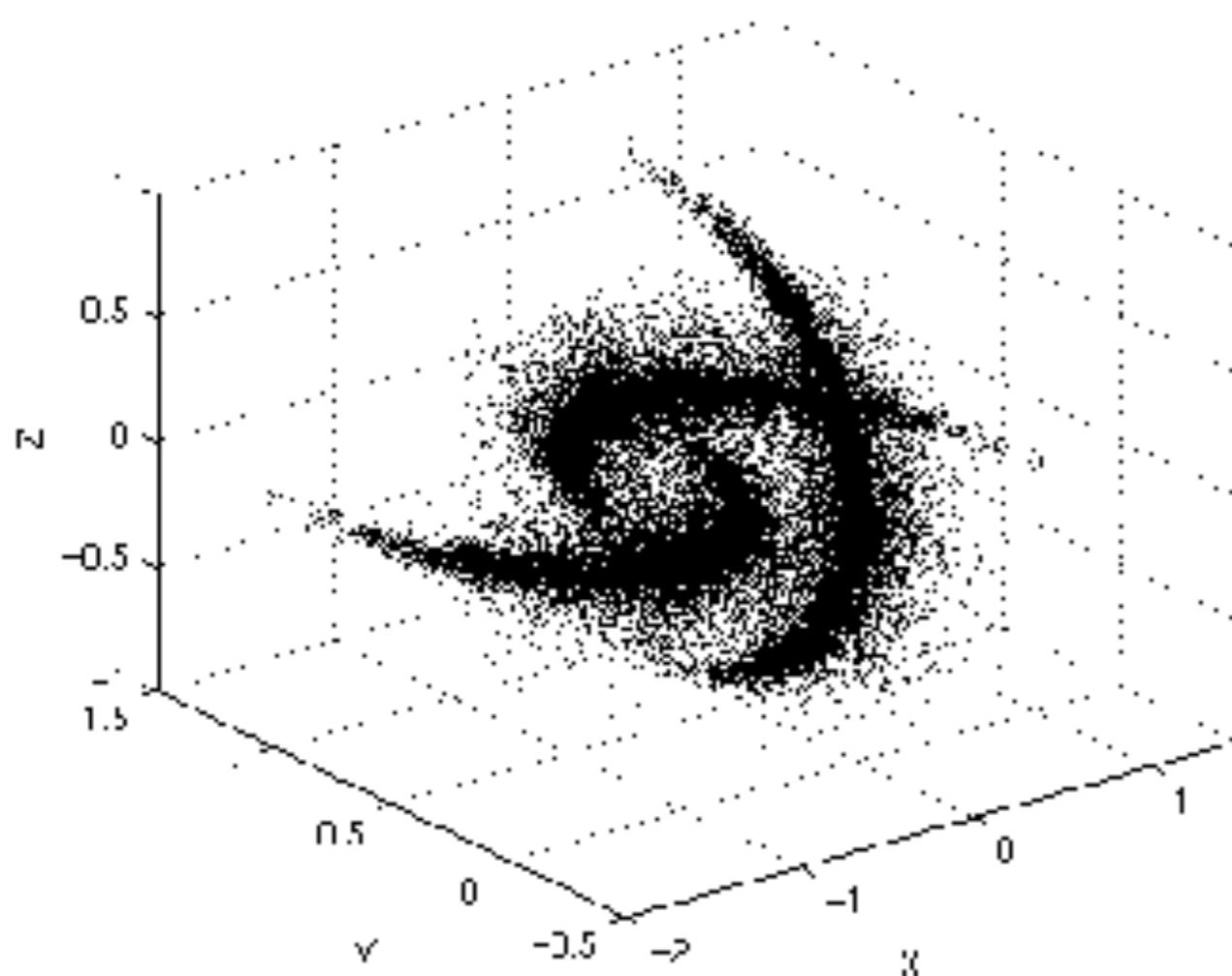
Fonctionne bien avec des classes complexes non gaussiennes.



# Mean-Shift

*Means-Shift* peut également servir d'outil de segmentation

Fonctionne bien avec des classes complexes non gaussiennes.



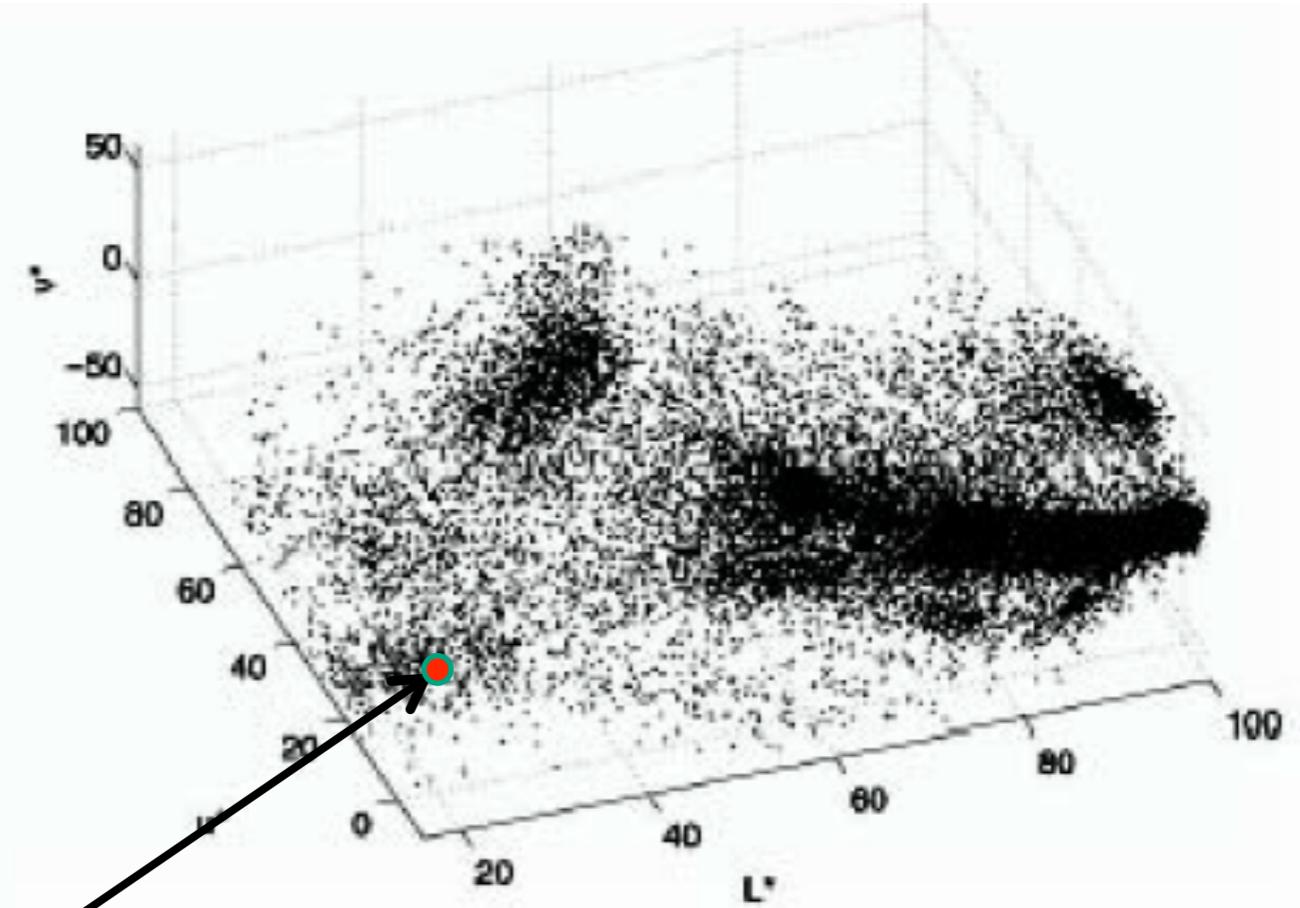
# Mean-Shift

*Means-Shift* peut également servir d'outil de segmentation



(r,g,b)

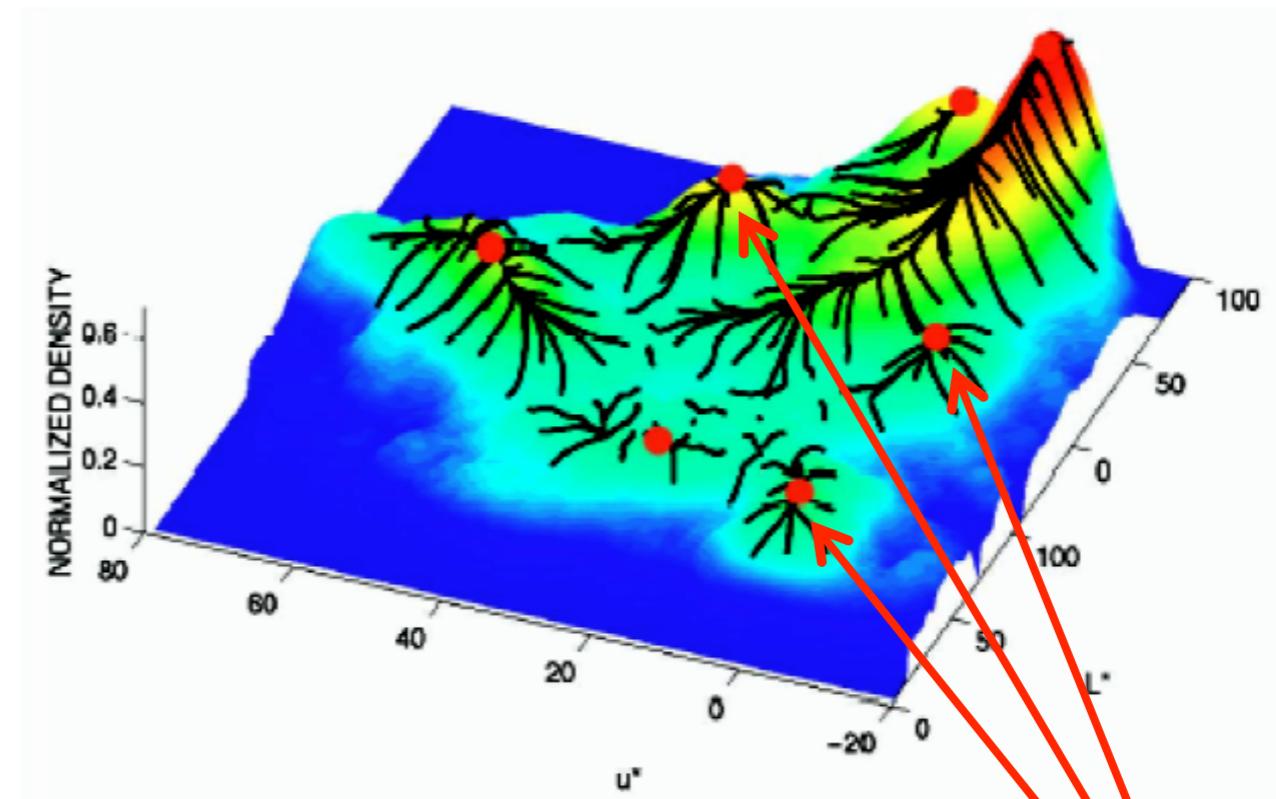
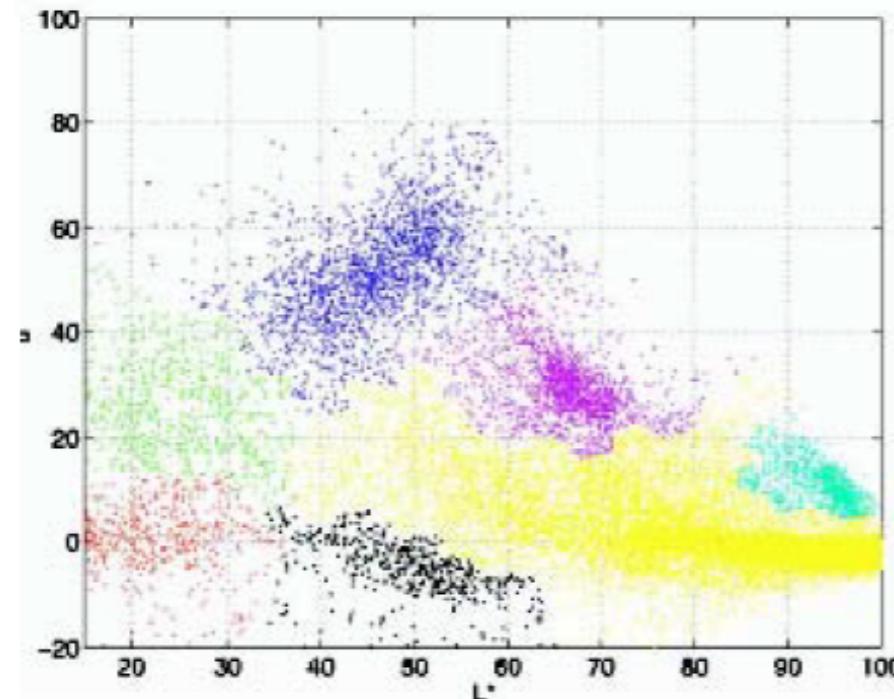
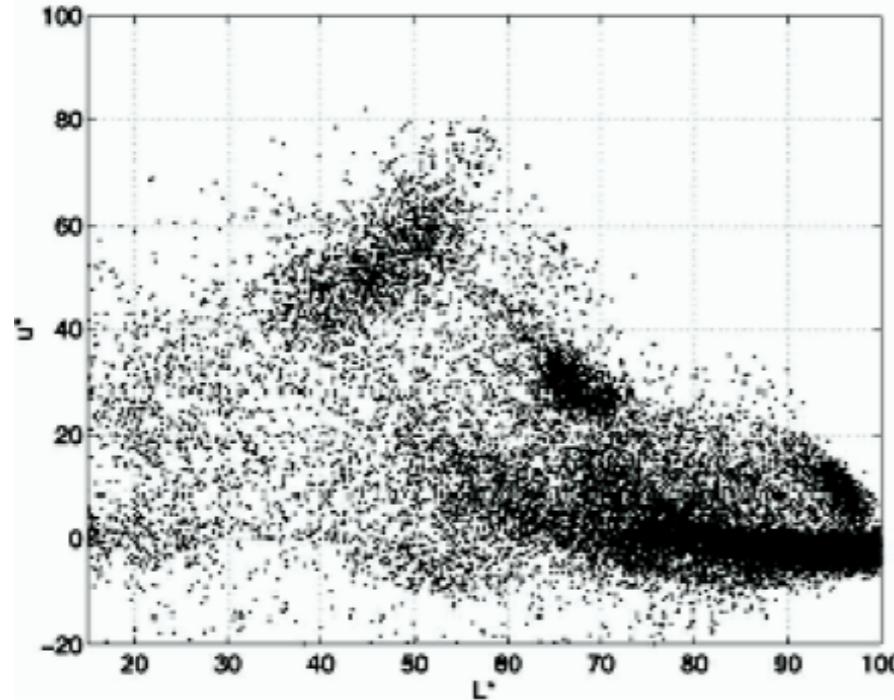
→ (l,u,v)



Comaniciu-Meer

# Mean-Shift

*Means-Shift* peut également servir d'outil de segmentation

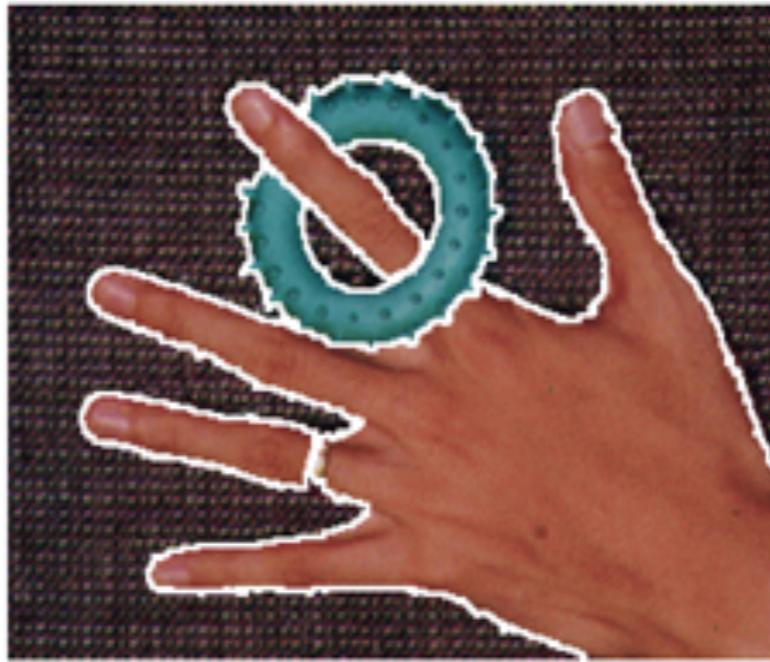


L'idée est de regrouper tous les pixels convergeant vers un même point

Comaniciu-Meer

# Mean-Shift

*Means-Shift* peut également servir d'outil de segmentation



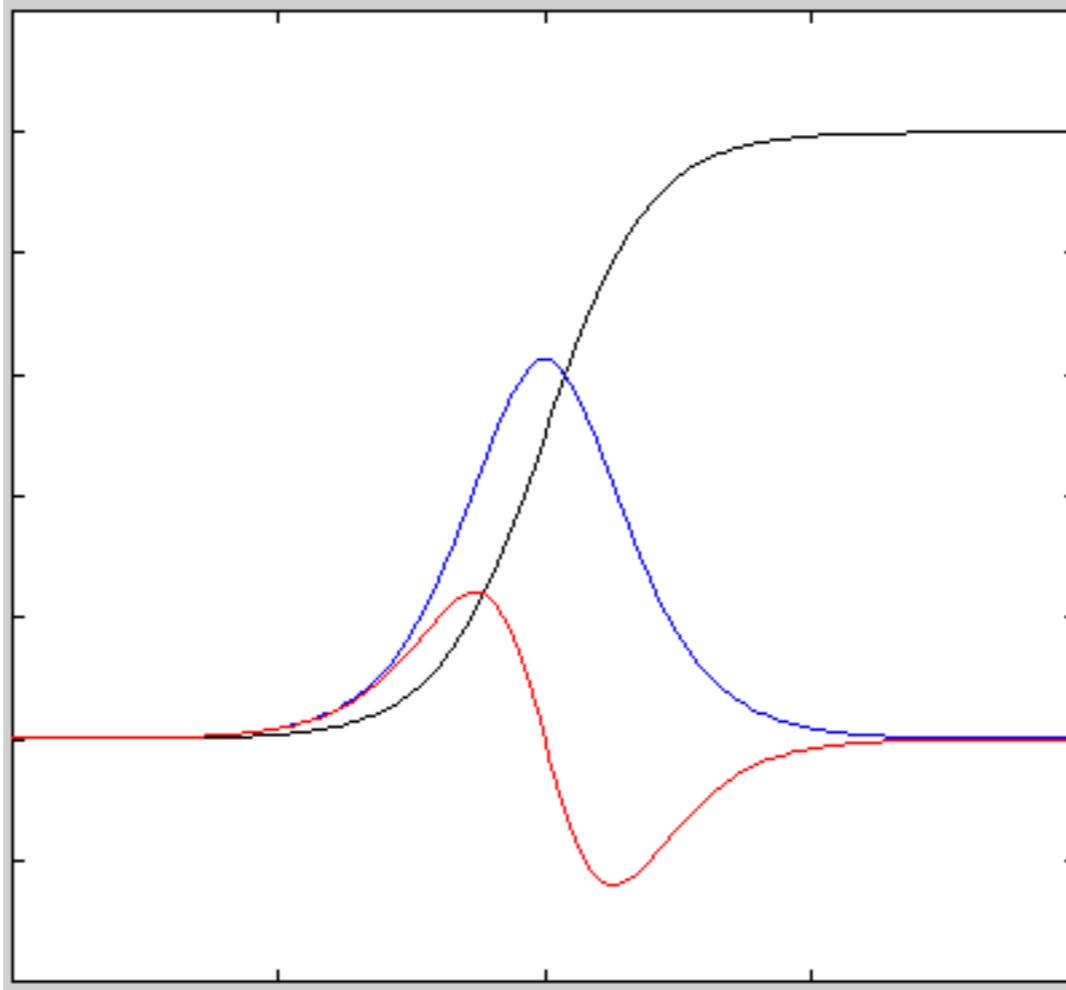
# Mean-Shift

Avantages <i>Mean-Shift</i>	Inconvénients <i>Mean-Shift</i>
<ul style="list-style-type: none"><li>• Trouve automatiquement le nombre de classes</li><li>• Aucun a priori quant à la distribution des données</li><li>• Ne dépend pas d'une « bonne initialisation »</li><li>• Robuste aux « outliers »</li></ul>	<ul style="list-style-type: none"><li>• Trouver le bon <math>h</math> peu être difficile<ul style="list-style-type: none"><li>• Trop petit : sur-segmentation</li><li>• Trop gros : fusion de modes.</li></ul></li></ul>

# Mean-Shift

Avantages <i>Mean-Shift</i>	Inconvénients <i>Mean-Shift</i>
<ul style="list-style-type: none"><li>• Trouve automatiquement le nombre de classes</li><li>• Aucun a priori quant à la distribution des données</li><li>• Ne dépend pas d'une « bonne initialisation »</li><li>• Robuste aux « outliers »</li></ul>	<ul style="list-style-type: none"><li>• Trouver le bon <math>h</math> peu être difficile<ul style="list-style-type: none"><li>• Trop petit : sur-segmentation</li><li>• Trop gros : fusion de modes.</li></ul></li></ul>

DÉMO



# Segmentation (chap 9)

*Approches par contours*

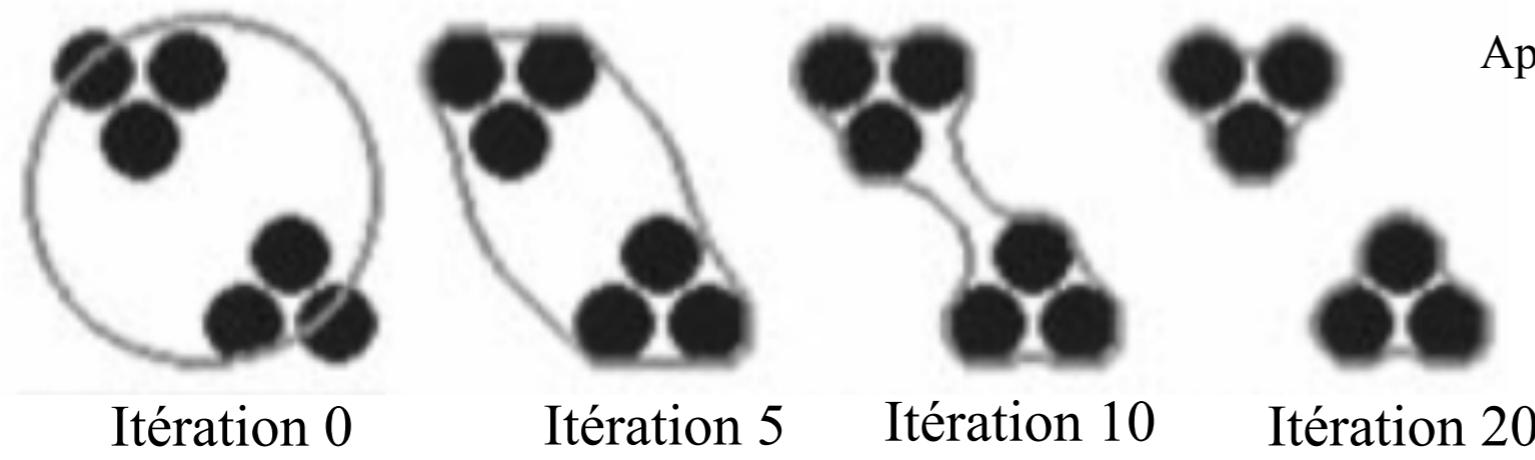
*Snakes et levelsets*

# Les contours actifs

**Objectifs des contours actifs:** faire évoluer une courbe afin de la faire coller aux contours d'un (ou plusieurs) objet que l'on souhaite segmenter. Fonctionne un peu comme un élastique.

# Les contours actifs

**Objectifs des contours actifs:** faire évoluer une courbe afin de la faire coller aux contours d'un (ou plusieurs) objet que l'on souhaite segmenter. Fonctionne un peu comme un élastique.



Approche « region-based »

[Chan-Vese 97]

# Les contours actifs

**Objectifs des contours actifs:** faire évoluer une courbe afin de la faire coller aux contours d'un (ou plusieurs) objet que l'on souhaite segmenter. Fonctionne un peu comme un élastique.



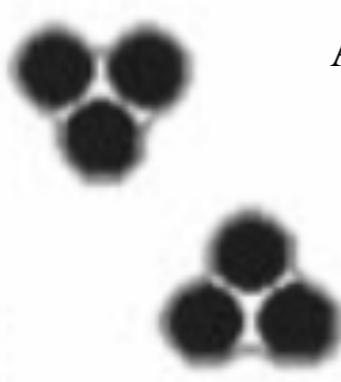
Itération 0



Itération 5



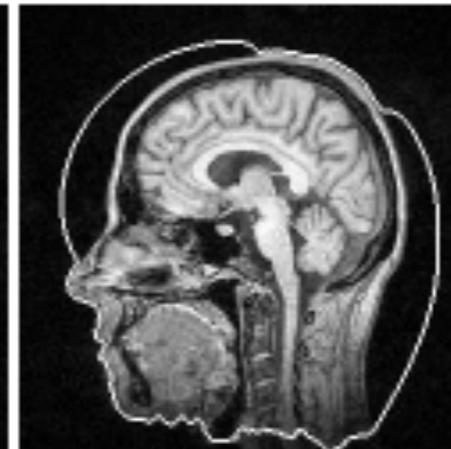
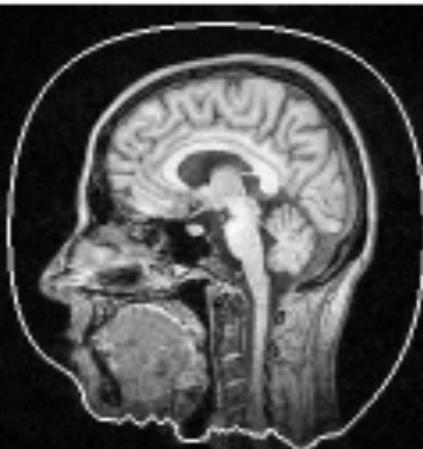
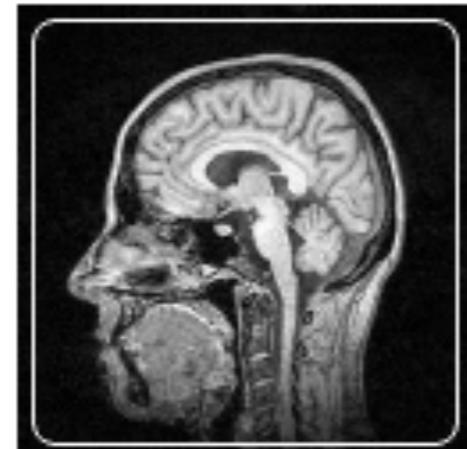
Itération 10



Itération 20

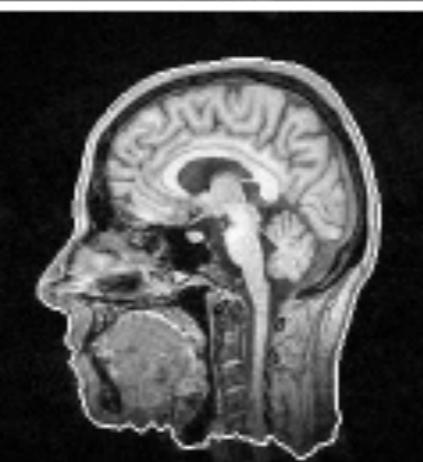
Approche « region-based »

[Chan-Vese 97]



Approche « edge-based »

[Weickert 00]



# Les contours actifs

## Deux grandes familles d'approches

**Snakes** : contours constitués de points de contrôle sur lesquels l'algorithme « pousse » en direction du contour le plus près.

**Level-set**: contours 2D est la courbe de niveau d'une fonction 3D. En déformant la fonction 3D, le contour se trouve « attirer » par le contour le plus près.

2D

Exemple animé

3D

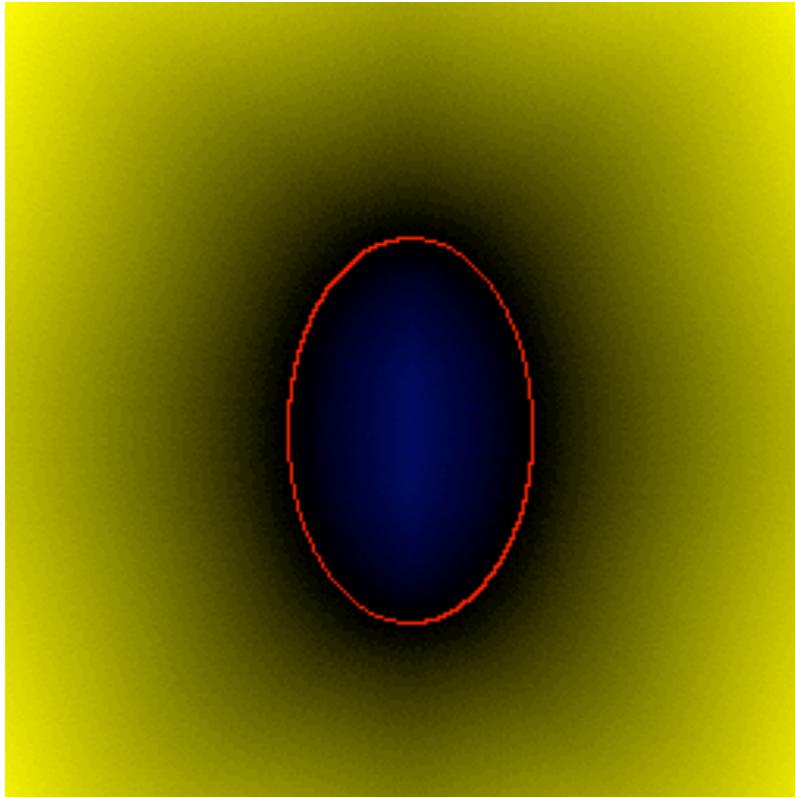
# Les contours actifs

## Deux grandes familles d'approches

**Snakes** : contours constitués de points de contrôle sur lesquels l'algorithme « pousse » en direction du contour le plus près.

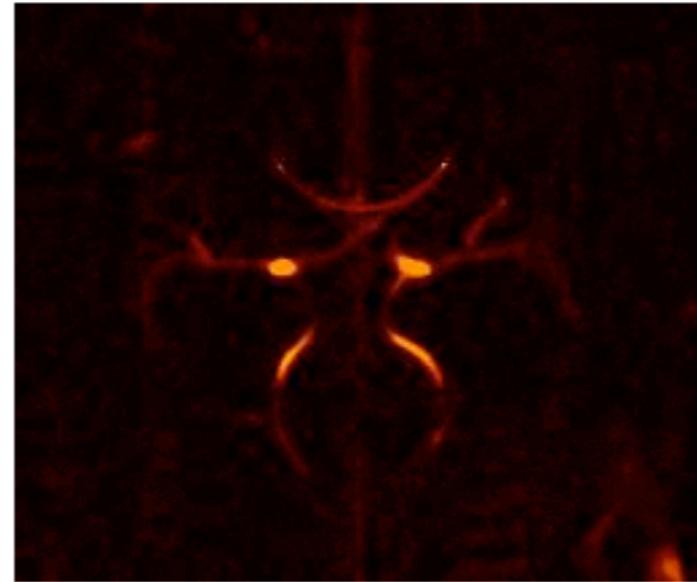
**Level-set**: contours 2D est la courbe de niveau d'une fonction 3D. En déformant la fonction 3D, le contour se trouve « attirer » par le contour le plus près.

2D



Exemple animé

3D



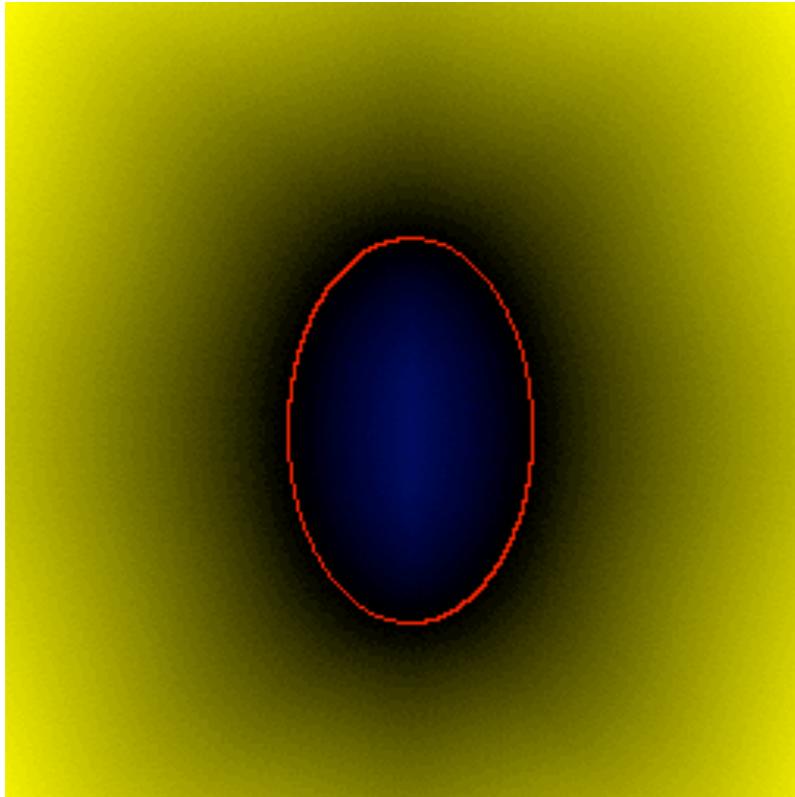
# Les contours actifs

## Deux grandes familles d'approches

**Snakes** : contours constitués de points de contrôle sur lesquels l'algorithme « pousse » en direction du contour le plus près.

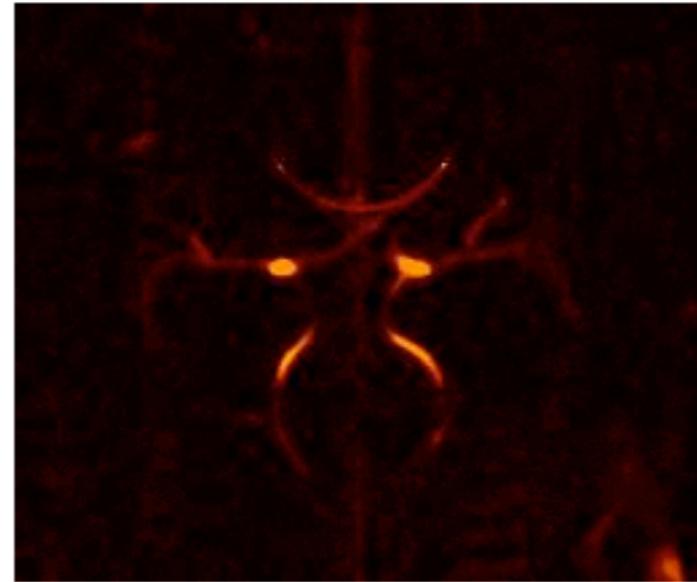
**Level-set**: contours 2D est la courbe de niveau d'une fonction 3D. En déformant la fonction 3D, le contour se trouve « attirer » par le contour le plus près.

2D



Exemple animé

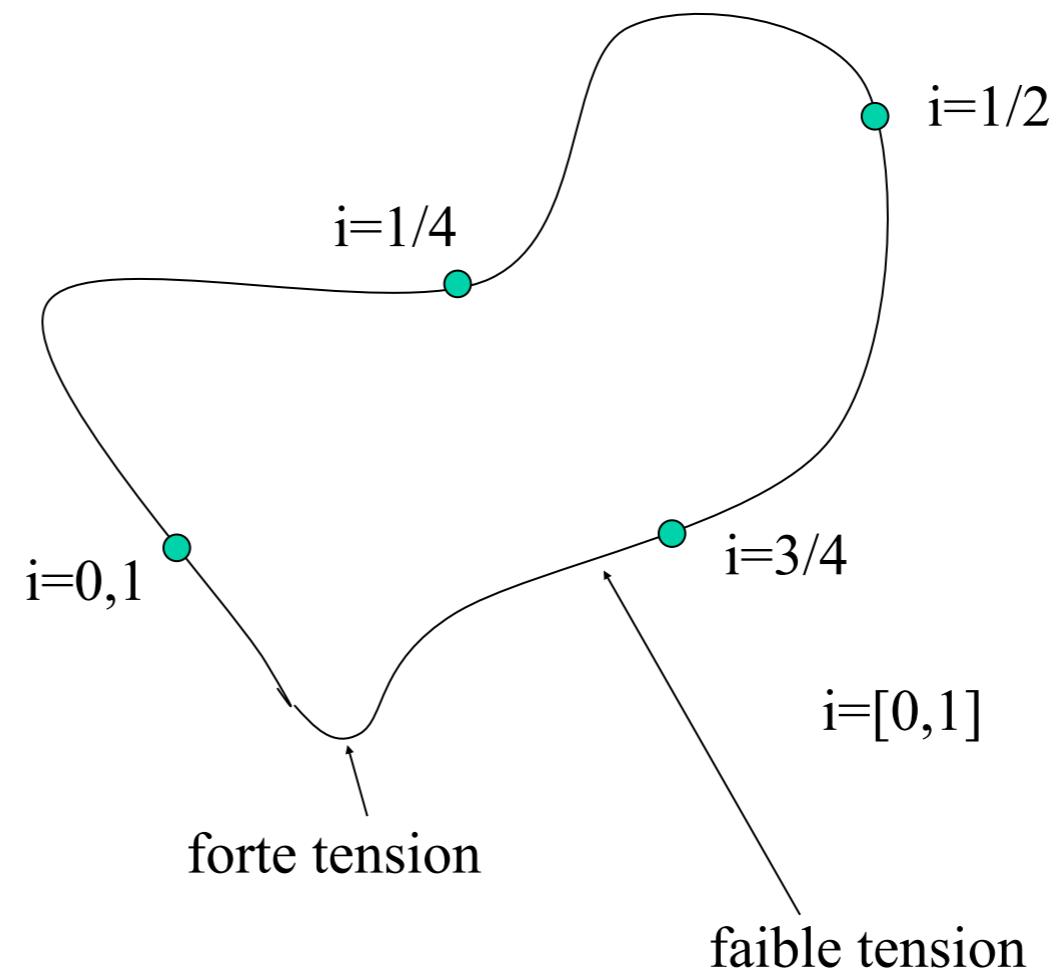
3D



# Les contours actifs

Soit  $C(i)$  une courbe paramétrique

$$C(i) \in IR^2$$



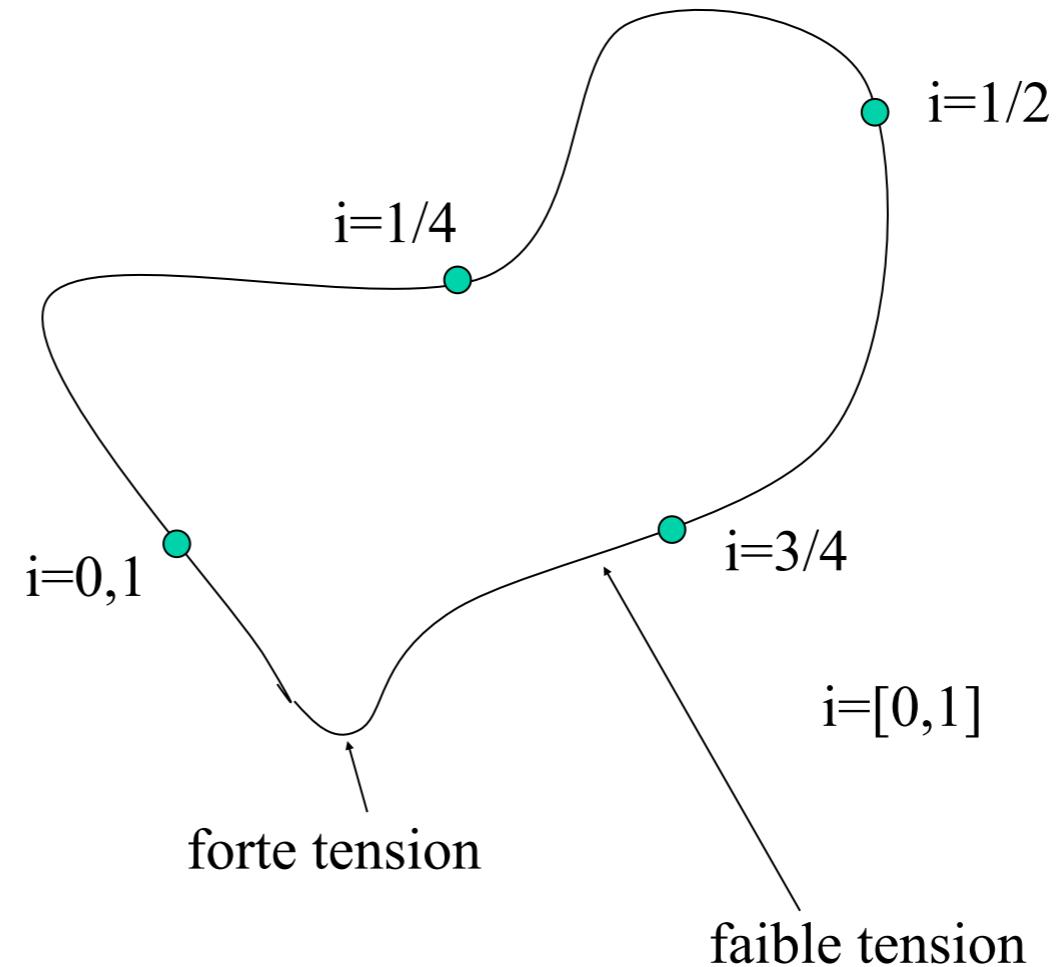
# Les contours actifs

Soit  $C(i)$  une courbe paramétrique

$$C(i) \in I\!R^2$$

$C'(i)$   $\Rightarrow$  mesure de "tension"

$C''(i)$   $\Rightarrow$  mesure de "rigidité" (stiffness)



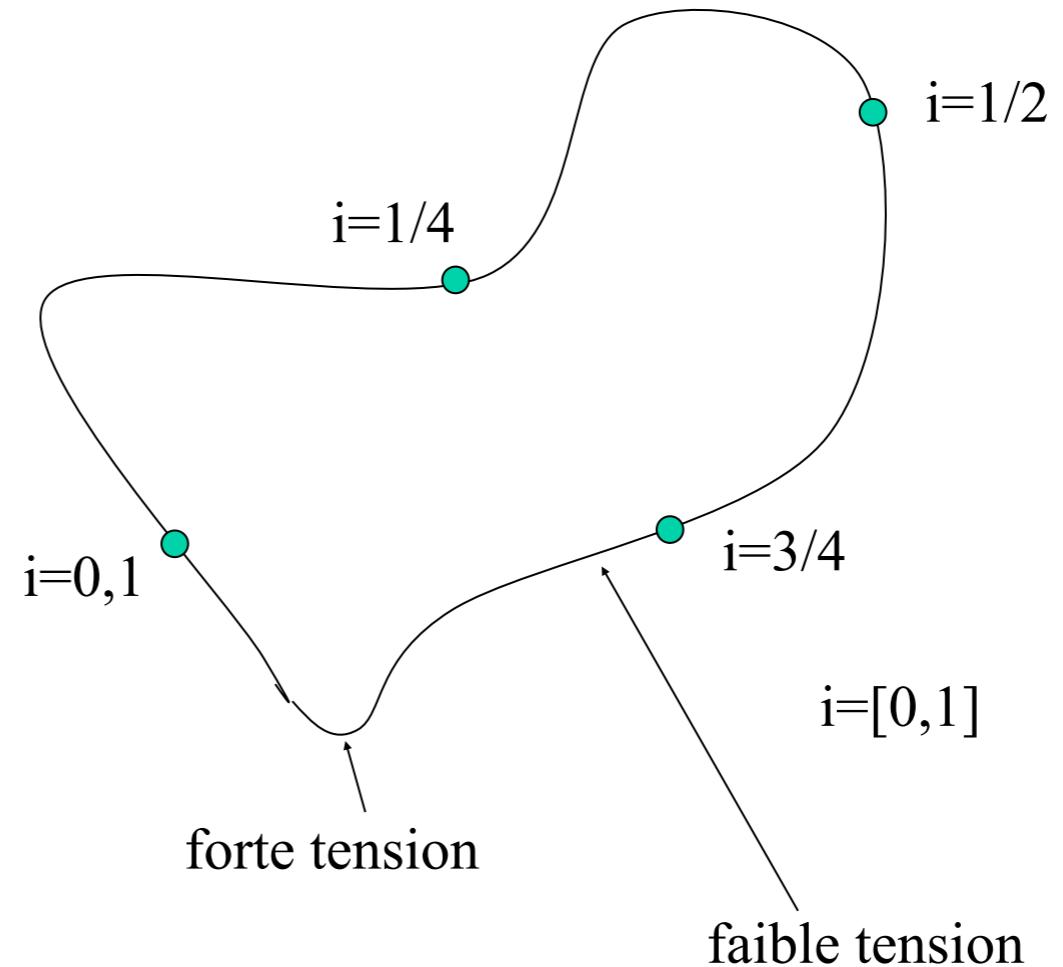
# Les contours actifs

Soit  $C(i)$  une courbe paramétrique

$$C(i) \in I\!R^2$$

$C'(i)$  ⇒ mesure de "tension"

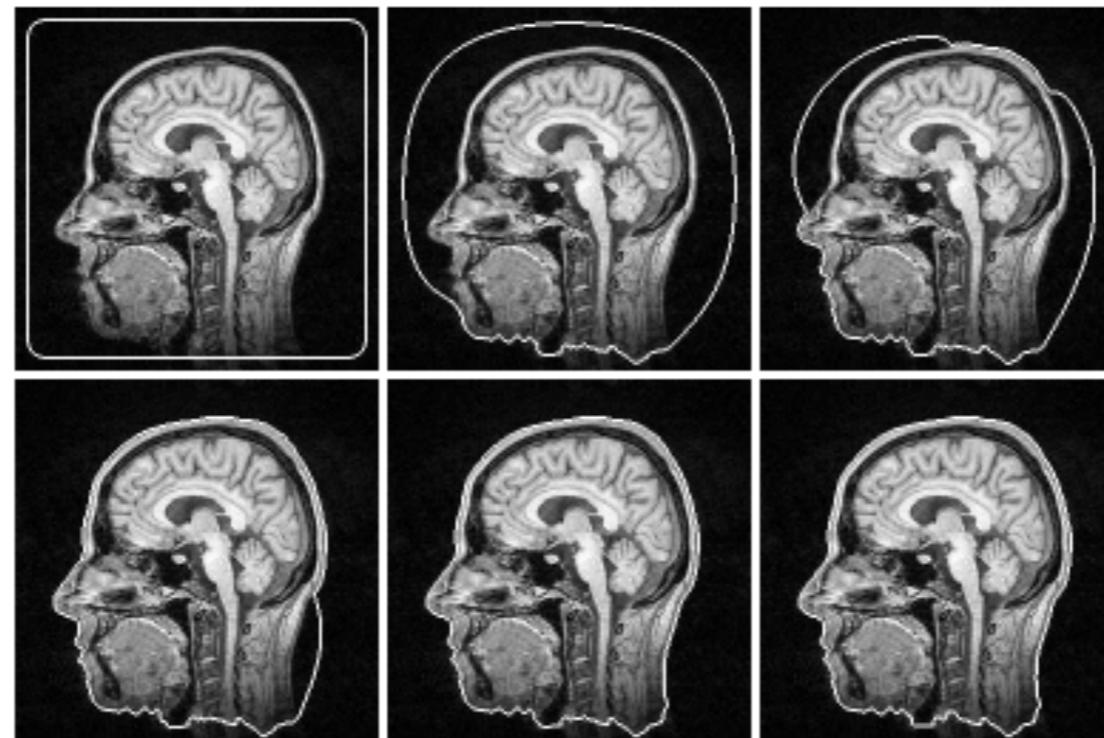
$C''(i)$  ⇒ mesure de "rigidité" (stiffness)



$\int_0^1 |C'(i)| di$  ⇒ mesure de la tension globale (minimiser cette fonction rend la courbe lisse et petite)

$\int_0^1 |C''(i)| di$  ⇒ mesure de la rigidité globale (minimiser cette fonction rend équidistant les pts de la courbe)

# Les contours actifs

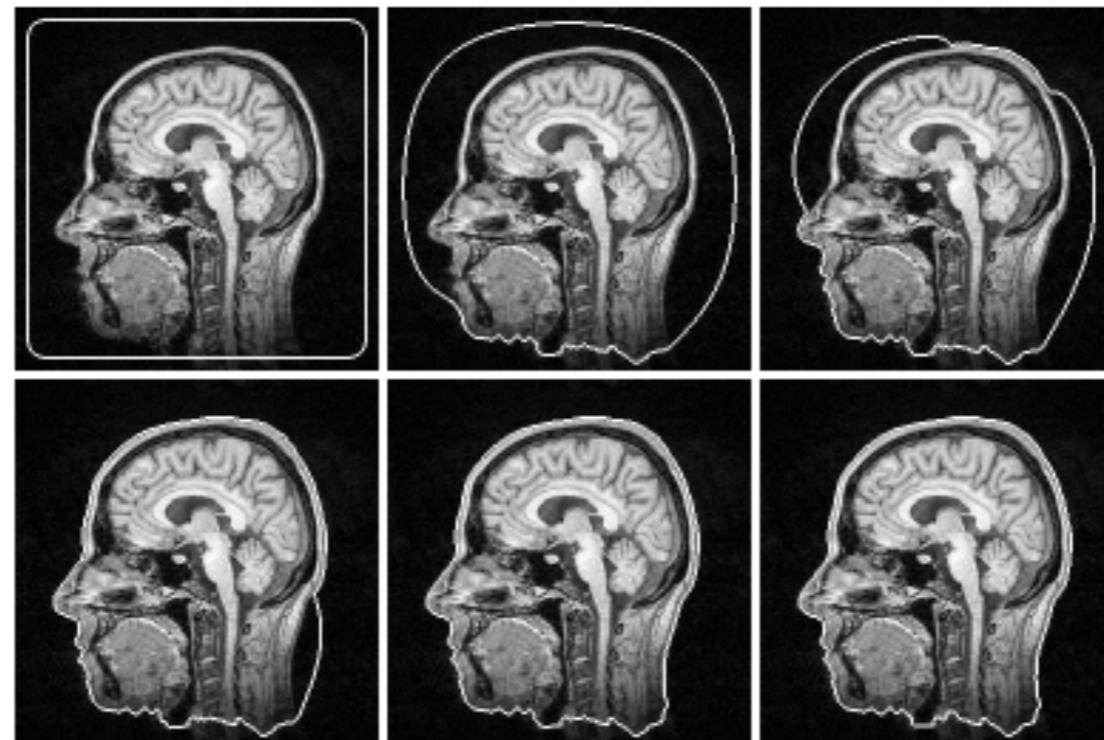


[Weickert 00]

On veut que  $C(q)$  évolue dans le temps jusqu'à **coller aux contours** des objets avoisinants. On souhaite aussi que la **courbure globale de  $C(q)$  soit « lisse »**. Donc, la « meilleure » courbe  $C$  étant donné  $I$  est

$$C = \arg \min_C E_{ext}(C, I) + E_{int}(C)$$

# Les contours actifs



[Weickert 00]

On veut que  $C(q)$  évolue dans le temps jusqu'à **coller aux contours** des objets avoisinants. On souhaite aussi que la **courbure globale de  $C(q)$  soit « lisse »**. Donc, la « meilleure » courbe  $C$  étant donné  $I$  est

$$C = \arg \min_C E_{ext}(C, I) + E_{int}(C)$$

Terme d'*a priori*  
(faible lorsque C est lisse)

Terme d'attache aux données  
(faible lorsque C colle aux contours de I)

# *Les snakes*

Kass M, Witkin A., Terzopoulos D. “Snakes: Active contour models”, IJCV, vol 4(1), 1988

**Plus de 13000 citations!!!**

# *Les Snakes*

$$E = E_{ext}(C, I) + E_{int}(C)$$

# *Les Snakes*

$$E = E_{ext}(C, I) + E_{int}(C)$$

$$E_{ext}(C, I) = \int_0^1 -|\nabla I(C(i))|^2 di$$

# Les Snakes

$$E = E_{ext}(C, I) + E_{int}(C)$$

$$E_{ext}(C, I) = \int_0^1 -|\nabla I(C(i))|^2 di$$

faible lorsque C colle aux contours de  $I$

# Les Snakes

$$E = E_{ext}(C, I) + E_{int}(C)$$

$$E_{ext}(C, I) = \int_0^1 -|\nabla I(C(i))|^2 di$$

$$E_{int}(C) = \int_0^1 \alpha (C'(i))^2 + \beta (C''(i))^2 di$$

faible lorsque C colle aux contours de  $I$

# Les Snakes

$$E = E_{ext}(C, I) + E_{int}(C)$$

$$E_{ext}(C, I) = \int_0^1 -|\nabla I(C(i))|^2 di$$

faible lorsque C colle aux contours de I

$$E_{int}(C) = \int_0^1 \alpha (C'(i))^2 + \beta (C''(i))^2 di$$

faible lorsque C a une tension et une rigidité faible

# Les Snakes

$$E = E_{ext}(C, I) + E_{int}(C)$$

$$E_{ext}(C, I) = \int_0^1 -|\nabla I(C(i))|^2 di$$

faible lorsque C colle aux contours de I

$$E_{int}(C) = \int_0^1 \alpha (C'(i))^2 + \beta (C''(i))^2 di$$

faible lorsque C a une tension et une rigidité faible

$$E = \int_0^1 -|\nabla I(C(i))|^2 + \alpha (C'(i))^2 + \beta (C''(i))^2 di$$

# Les Snakes

$$E = E_{ext}(C, I) + E_{int}(C)$$

$$E_{ext}(C, I) = \int_0^1 -|\nabla I(C(i))|^2 di$$

faible lorsque C colle aux contours de I

$$E_{int}(C) = \int_0^1 \alpha (C'(i))^2 + \beta (C''(i))^2 di$$

faible lorsque C a une tension et une rigidité faible

$$E = \int_0^1 -|\nabla I(C(i))|^2 + \alpha (C'(i))^2 + \beta (C''(i))^2 di$$

**Question :** comment trouver la meilleure courbe C afin de minimiser E?

# Les Snakes

$$E = E_{ext}(C, I) + E_{int}(C)$$

$$E_{ext}(C, I) = \int_0^1 -|\nabla I(C(i))|^2 di$$

faible lorsque C colle aux contours de I

$$E_{int}(C) = \int_0^1 \alpha (C'(i))^2 + \beta (C''(i))^2 di$$

faible lorsque C a une tension et une rigidité faible

$$E = \int_0^1 -|\nabla I(C(i))|^2 + \alpha (C'(i))^2 + \beta (C''(i))^2 di$$

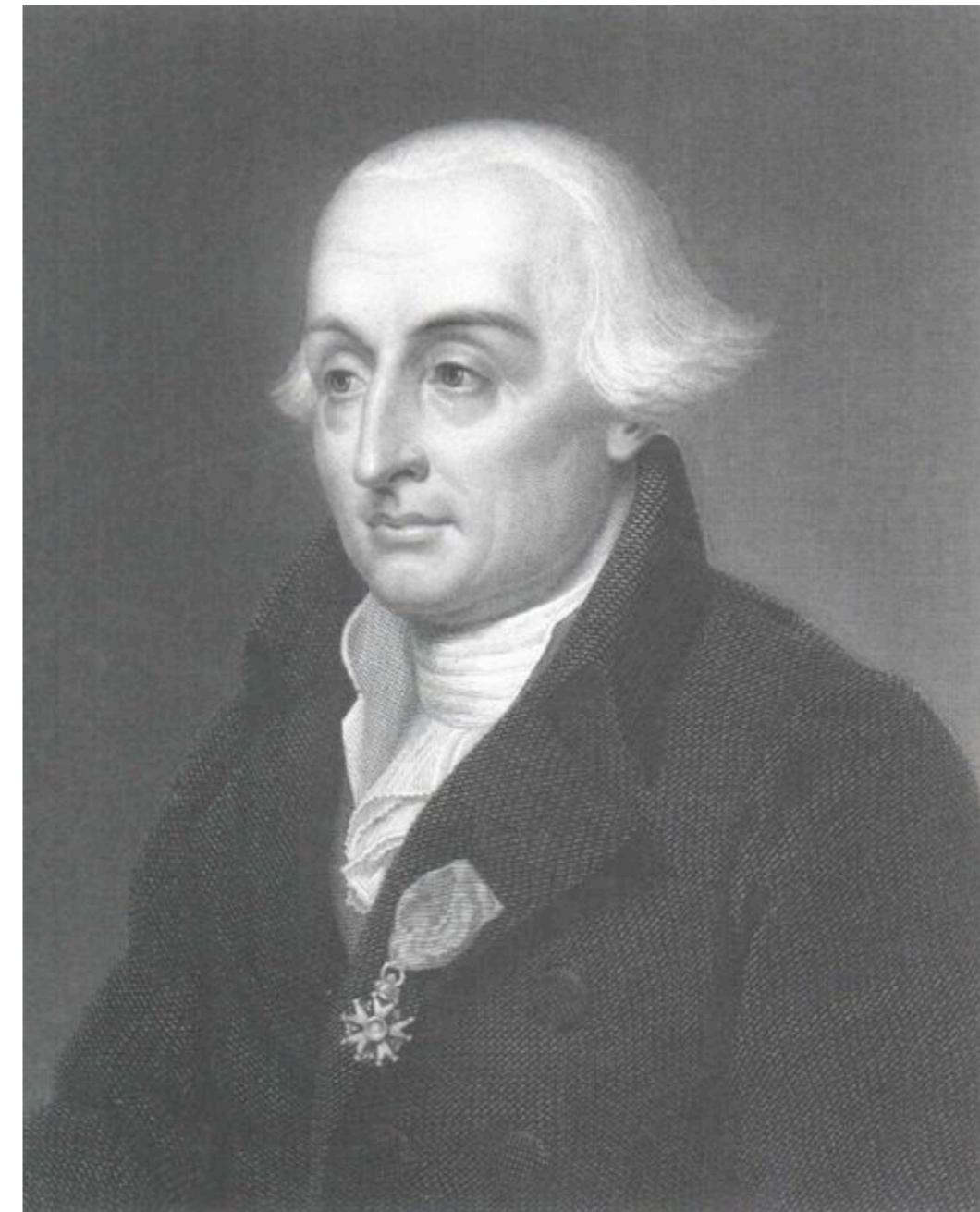
**Question :** comment trouver la meilleure courbe C afin de minimiser E?

Réponse : théorème d'Euler - Lagrange

# Le théorème d'Euler-Lagrange



Leonhard Euler, 1707 - 1783



Joseph-Louis Lagrange 1736 - 1813

# Le théorème d'Euler-Lagrange

**Calcul de nombres réels**

# Le théorème d'Euler-Lagrange

## Calcul de nombres réels

Soit  $h(x)$  une fonction continue qui transforme une valeur réelle en une autre valeur réelle.

# Le théorème d'Euler-Lagrange

## Calcul de nombres réels

Soit  $h(x)$  une fonction continue qui transforme une valeur réelle en une autre valeur réelle.

Si  $h$  possède un minimum local pour la valeur «  $a$  », alors  $\frac{dh(x = a)}{dx} = 0$

Si  $h$  est une fonction convexe, alors «  $a$  » est le seul minimum.

# Le théorème d'Euler-Lagrange

## Calcul de nombres réels

Soit  $h(x)$  une fonction continue qui transforme une valeur réelle en une autre valeur réelle.

Si  $h$  possède un minimum local pour la valeur «  $a$  », alors  $\frac{dh(x = a)}{dx} = 0$

Si  $h$  est une fonction convexe, alors «  $a$  » est le seul minimum.

## Calcul variationnel

Soit  $H(p(x))$  une fonction continue qui transforme une fonction  $p(x)$  en une valeur réelle.

Si  $H$  possède un minimum local pour une fonction  $p(x)$ , alors  $p(x)$  doit satisfaire l'**équation d'Euler-Lagrange**.

Si  $H$  est strictement convexe, alors «  $p$  » est le seul minimum.

# Le théorème d'Euler-Lagrange

## Théorème d'Euler-Lagrange (cas 1D)

Une fonction continue  $p(x), x \in [a, b]$  qui minimise la fonction  $H(p)$ :

# Le théorème d'Euler-Lagrange

## Théorème d'Euler-Lagrange (cas 1D)

Une fonction continue  $p(x), x \in [a, b]$  qui minimise la fonction  $H(p)$ :

$$H(p) = \int_a^b F(x, p, p', p'') dx$$

# Le théorème d'Euler-Lagrange

## Théorème d'Euler-Lagrange (cas 1D)

Une fonction continue  $p(x), x \in [a, b]$  qui minimise la fonction  $H(p)$ :

$$H(p) = \int_a^b F(x, p, p', p'') dx$$

doit satisfaire l'équation d'Euler-Lagrange que voici

# Le théorème d'Euler-Lagrange

## Théorème d'Euler-Lagrange (cas 1D)

Une fonction continue  $p(x), x \in [a, b]$  qui minimise la fonction  $H(p)$ :

$$H(p) = \int_a^b F(x, p, p', p'') dx$$

doit satisfaire l'équation d'Euler-Lagrange que voici

$$\frac{dF}{dp} - \frac{d}{dx} \left( \frac{dF}{dp'} \right) + \frac{d^2}{dx^2} \left( \frac{dF}{dp''} \right) = 0$$

# Le théorème d'Euler-Lagrange

$$E = \int_0^1 -|\nabla I(C(i))|^2 + \alpha(C'(i))^2 + \beta(C''(i))^2 di$$

# Le théorème d'Euler-Lagrange

$$E = \int_0^1 -|\nabla I(C(i))|^2 + \alpha(C'(i))^2 + \beta(C''(i))^2 di$$

$F(i, C(i), C'(i), C''(i))$

# Le théorème d'Euler-Lagrange

$$E = \int_0^1 - \underbrace{|\nabla I(C(i))|^2 + \alpha (C'(i))^2 + \beta (C''(i))^2}_{F(i, C(i), C'(i), C''(i))} di$$

$$\frac{dF}{dC} - \frac{d}{di} \left( \frac{dF}{dC'} \right) + \frac{d^2}{di^2} \left( \frac{dF}{dC''} \right) = 0$$

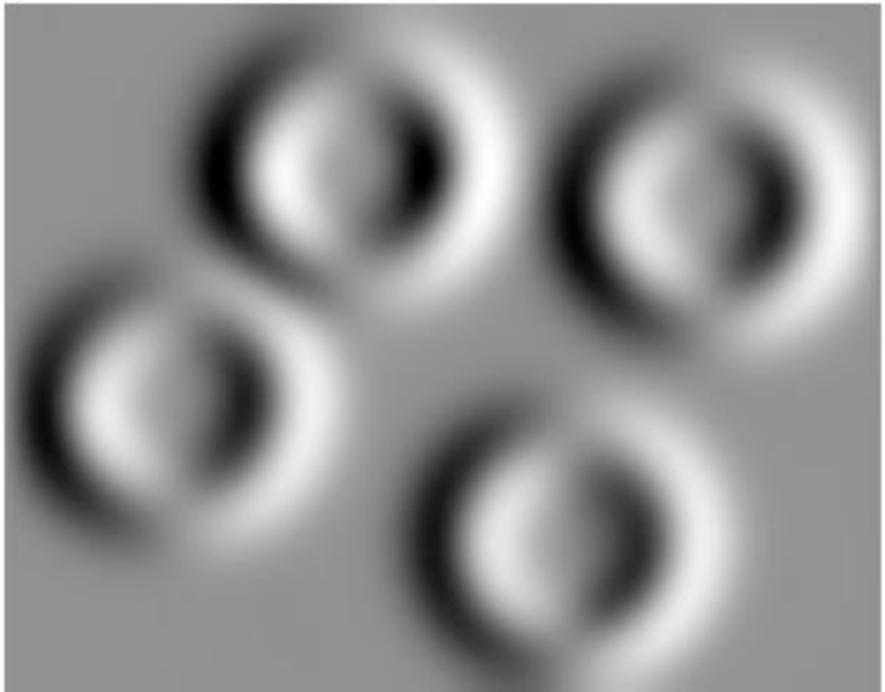
$$\frac{d}{dC} \left( - |\nabla I(C(i))|^2 \right) - \frac{d}{di} (2\alpha C') + \frac{d^2}{di^2} (2\beta C'') = 0$$

$$- \frac{d}{dC} \left( \left( \frac{\partial I(C)}{\partial x} \right)^2 + \left( \frac{\partial I(C)}{\partial y} \right)^2 \right) - 2\alpha C'' + 2\beta C''' = 0$$

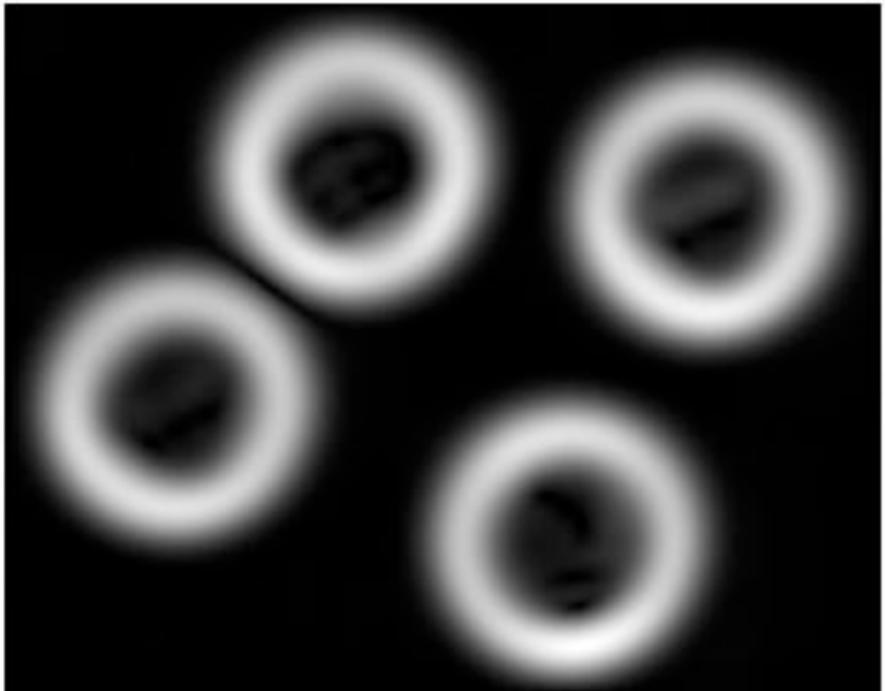
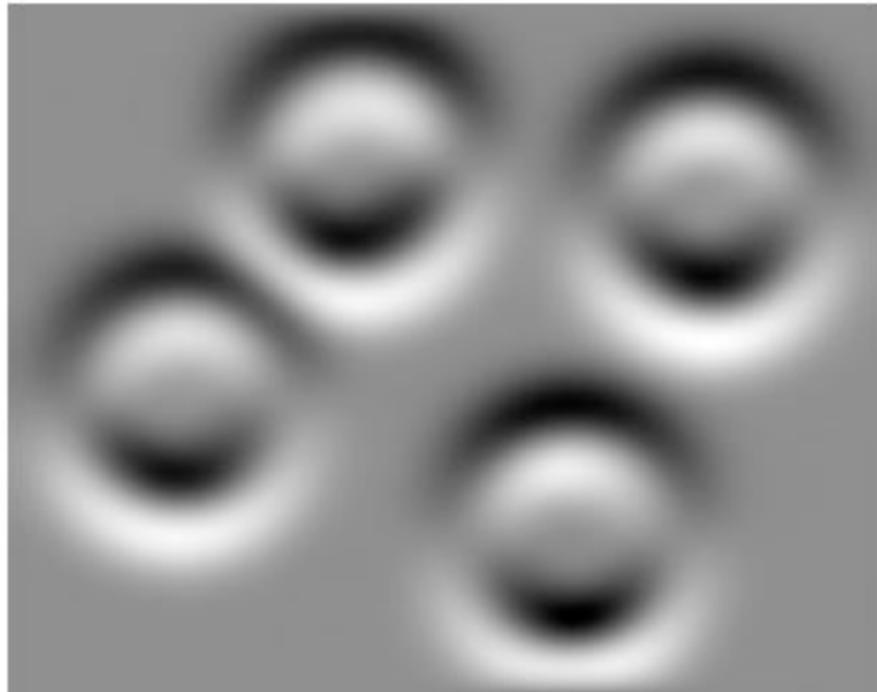
$$- \left( 2 \left( \frac{\partial^2 I(C)}{\partial x^2} \right) + 2 \left( \frac{\partial^2 I(C)}{\partial x \partial y} \right), 2 \left( \frac{\partial^2 I(C)}{\partial x \partial y} \right) + 2 \left( \frac{\partial^2 I(C)}{\partial y^2} \right) \right) - 2\alpha C'' + 2\beta C''' = 0$$
$$- (f_x(C), f_y(C)) - \alpha C'' + \beta C''' = 0$$

# Illustrer $(f_x, f_y)$

$$\frac{\partial I}{\partial x}$$



$$\frac{\partial I}{\partial y}$$



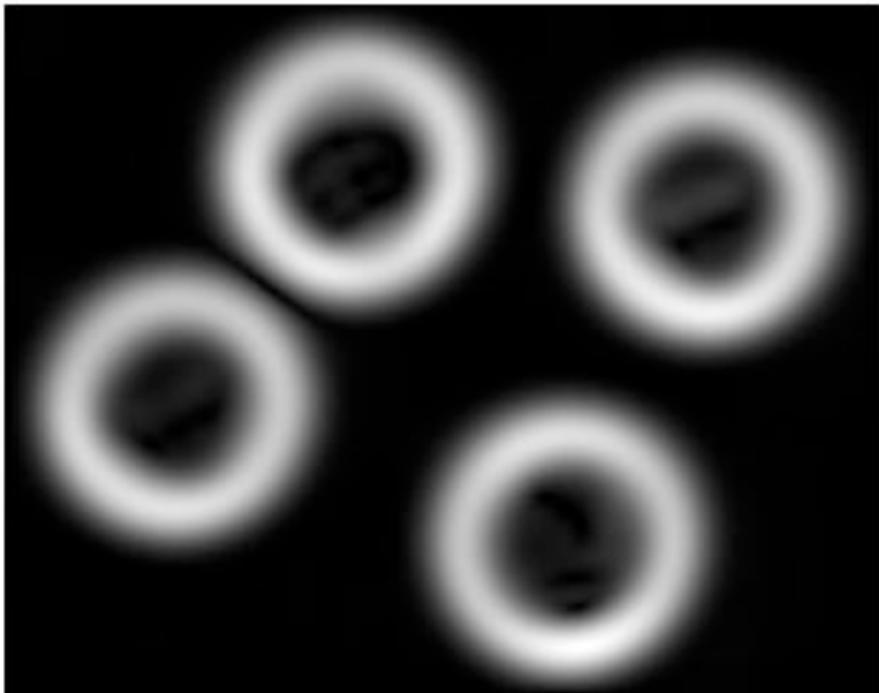
$$I$$



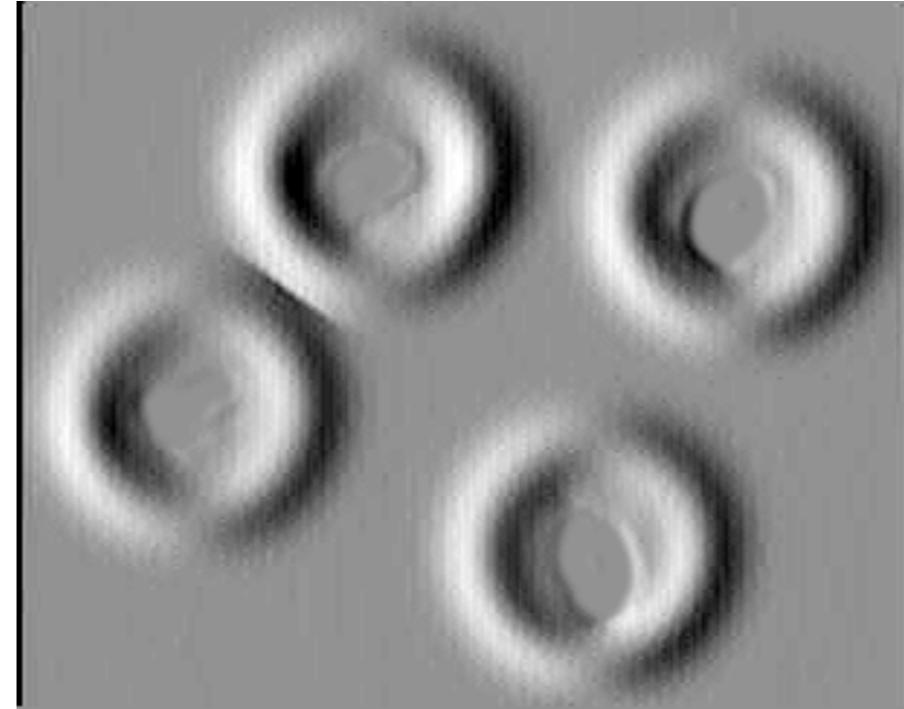
$$|\nabla I|^2 = \left( \frac{\partial I}{\partial x} \right)^2 + \left( \frac{\partial I}{\partial y} \right)^2$$

# Illustrer ( $f_x, f_y$ )

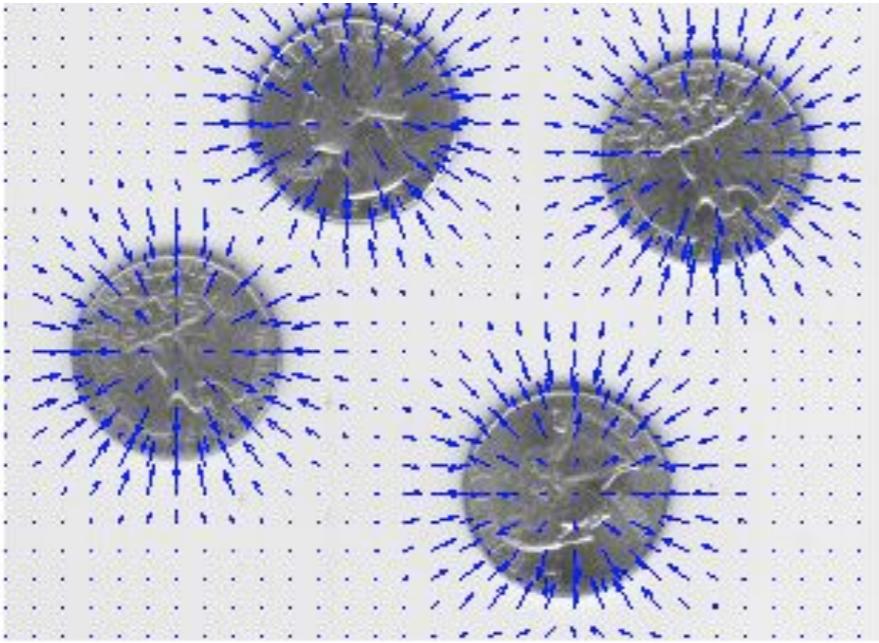
$$f = |\nabla I|^2$$



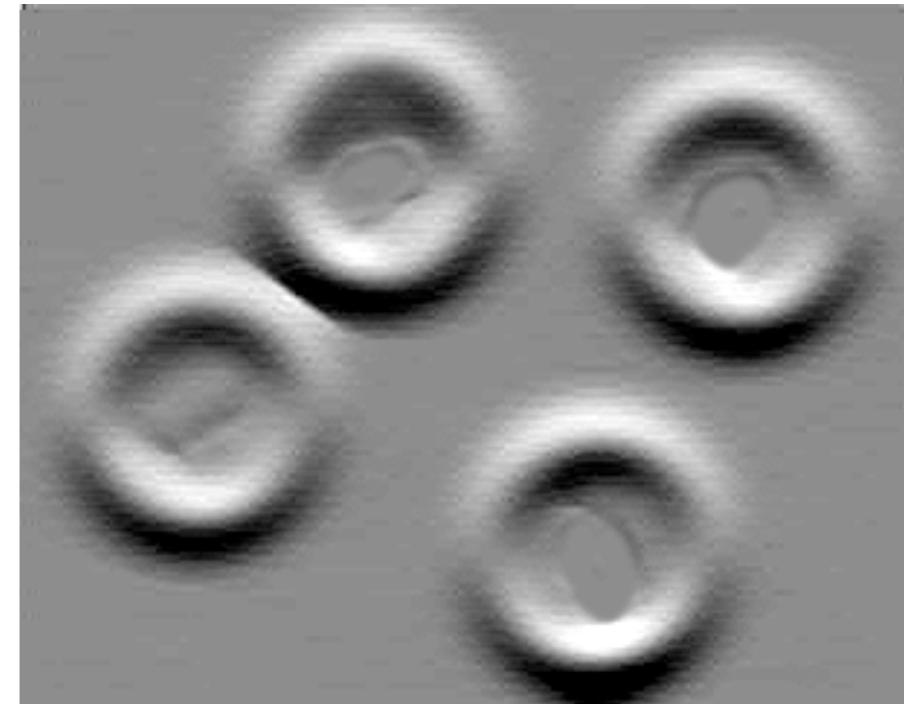
$$f_x = \partial f / \partial x$$



$$I + \left( \frac{\partial f}{\partial x}, \frac{\partial f}{\partial y} \right)$$



$$f_y = \partial f / \partial y$$



# Optimisation

**Descente de gradient (approche explicite):**

# Optimisation

**Descente de gradient (approche explicite):**

$$C^{t+1} = C^t - \delta \frac{dE}{dC}$$

# Optimisation

**Descente de gradient (approche explicite):**

$$C^{t+1} = C^t - \delta \frac{dE}{dC}$$

considérant que

$$\frac{dE}{dC} = -\left(f_x(C), f_y(C)\right) - \alpha C'' + \beta C'''$$

# Optimisation

**Descente de gradient (approche explicite):**

$$C^{t+1} = C^t - \delta \frac{dE}{dC}$$

considérant que

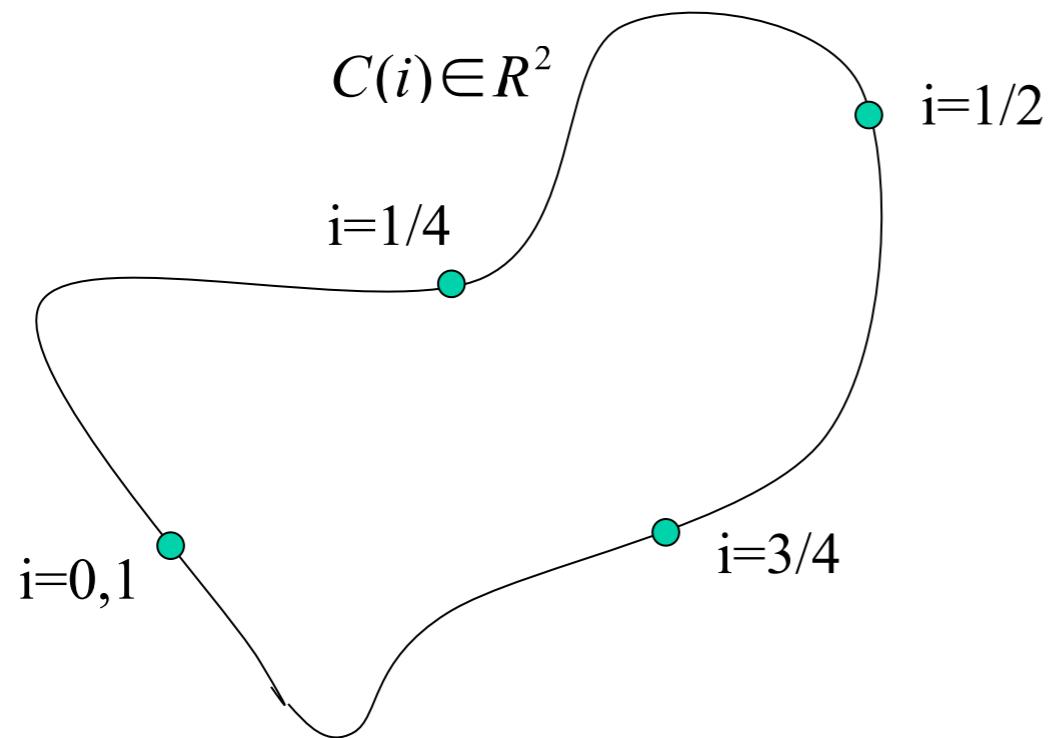
$$\frac{dE}{dC} = -\left(f_x(C), f_y(C)\right) - \alpha C'' + \beta C'''$$

on obtient :

$$C^{t+1} = C^t + \delta \left( \left( f_x(C^t), f_y(C^t) \right) + \alpha C^{t''} - \beta C^{t'''} \right)$$

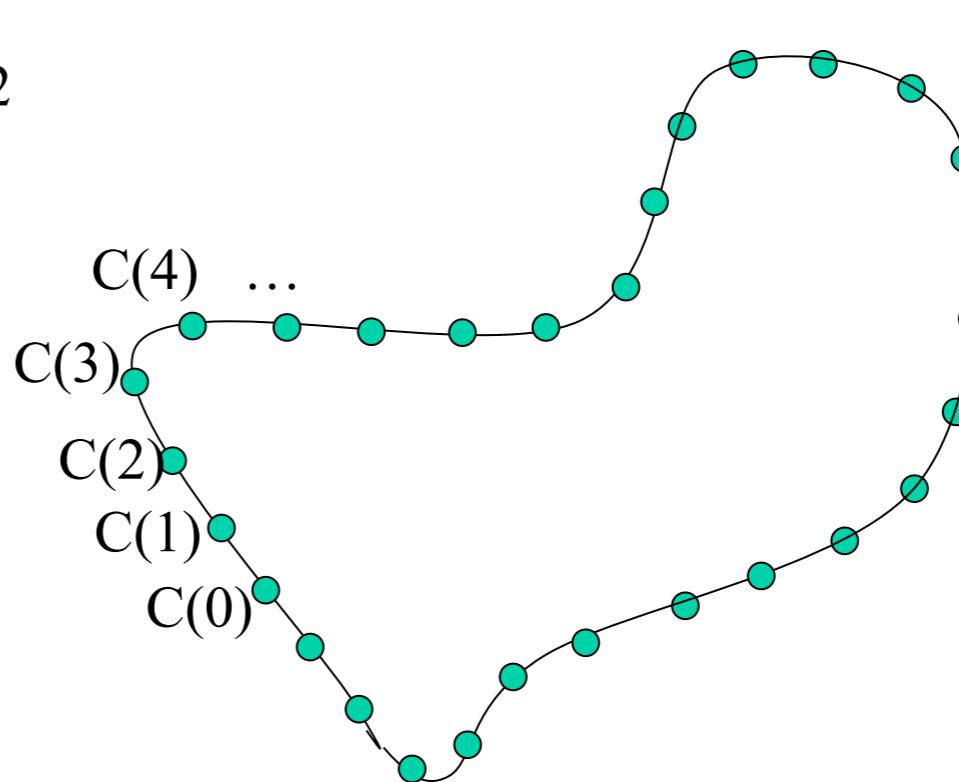
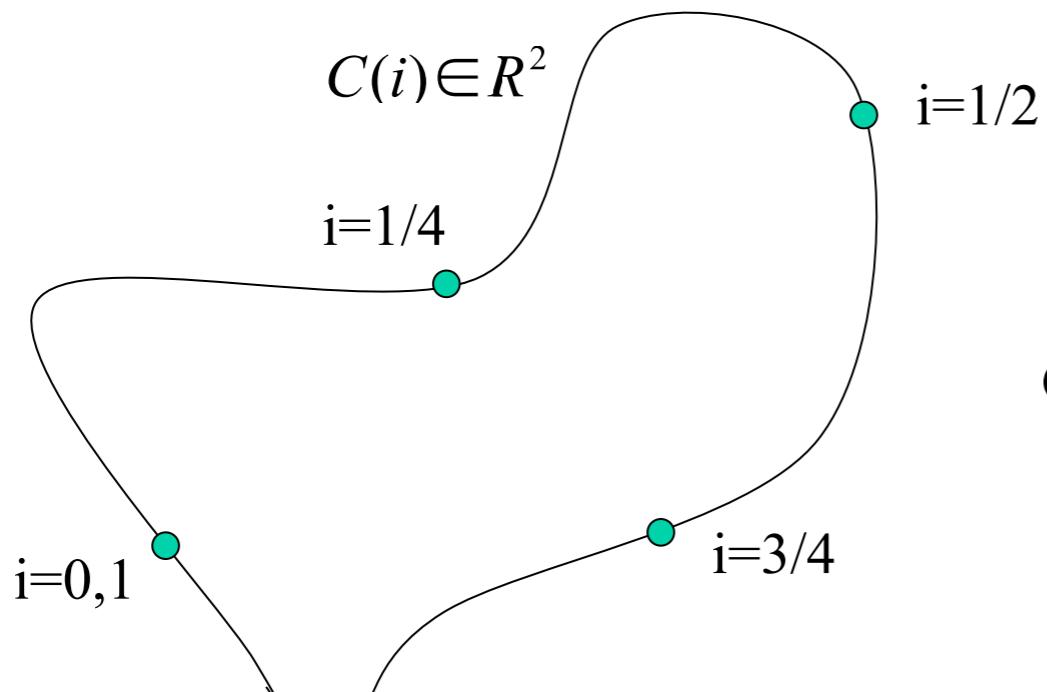
# Optimisation

## Approximation numérique



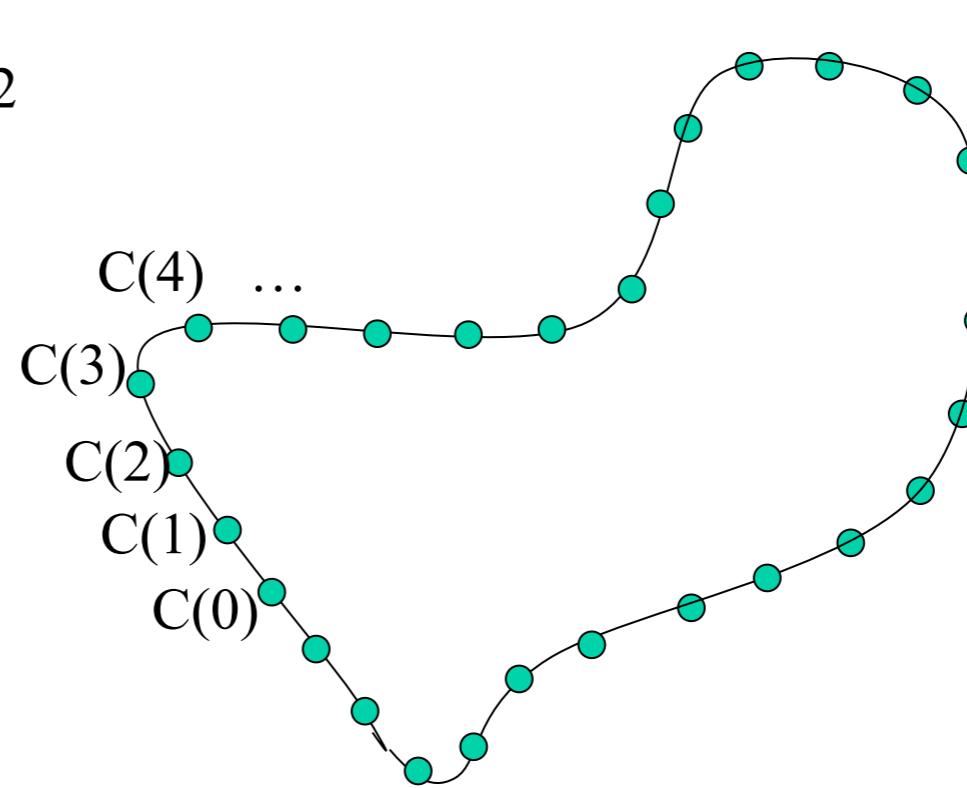
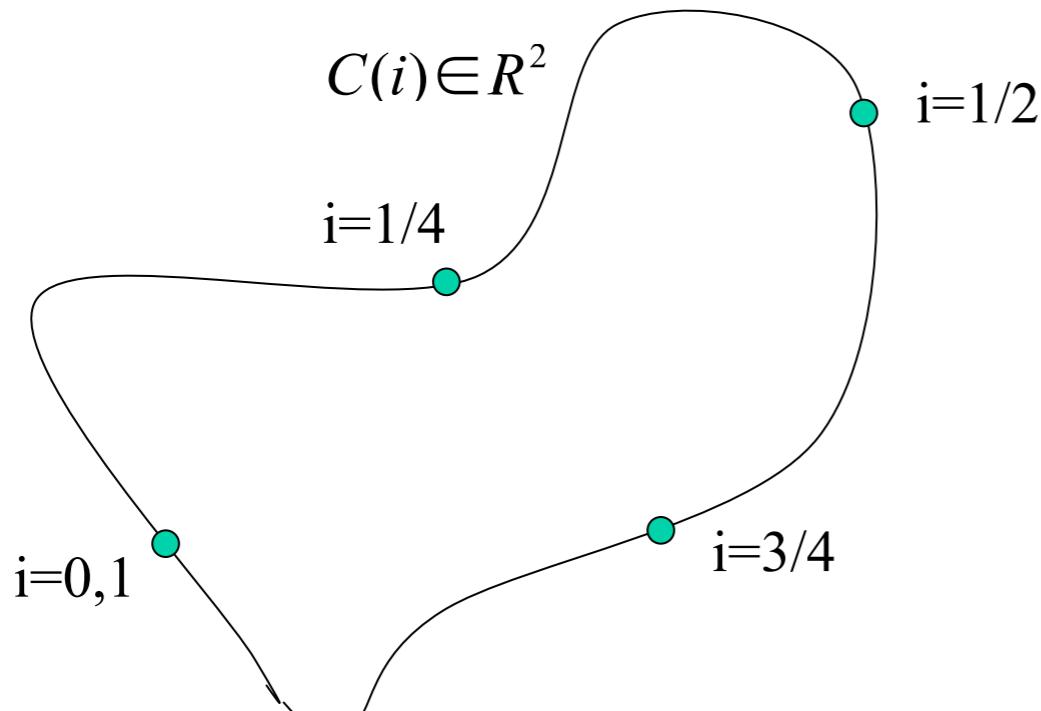
# Optimisation

## Approximation numérique



# Optimisation

## Approximation numérique



$$C'' = \frac{d^2 C}{di^2} \approx C^t(i+1) - 2C^t(i) + C^t(i-1)$$

$$C'''' = \frac{d^4 C}{di^4} \approx C^t(i+2) - 4C^t(i+1) + 6C^t(i) - 4C^t(i-1) + C^t(i-2)$$

# Optimisation

## Approximation numérique

$$C^{t+1} = C^t + \delta \left( \left( f_x(C^t), f_y(C^t) \right) + \alpha C'' - \beta C''' \right)$$

# Optimisation

## Approximation numérique

$$C^{t+1} = C^t + \delta \left( \left( f_x(C^t), f_y(C^t) \right) + \alpha C'' - \beta C''' \right)$$

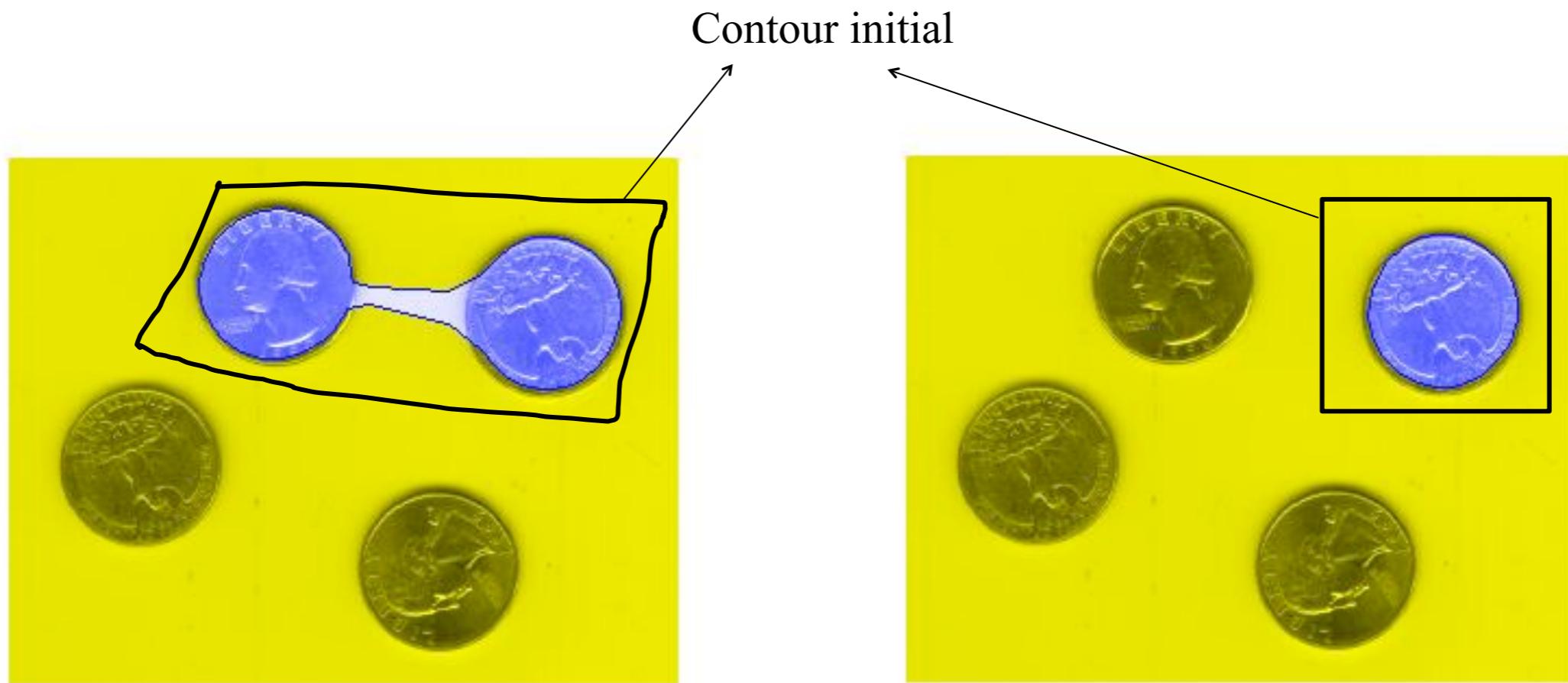
Est approximée par

$$C^{t+1} \approx C^t + \delta \left( \begin{aligned} & \left( f_x(C^t), f_y(C^t) \right) + \\ & \alpha \left( C^t(i+1) - 2C^t(i) + C^t(i-1) \right) + \\ & - \beta \left( C^t(i+2) - 4C^t(i+1) + 6C^t(i) - 4C^t(i-1) + C^t(i-2) \right) \end{aligned} \right)$$

## L'algorithme des snakes (méthode explicite )

1.  $I = \text{FiltragePassBasGaussien}(I)$  /\*  $I$  est l'image à segmenter \*/
2. Calculer  $f = |\nabla I|^2$
3. Calculer  $f = (f_x, f_y)$
4. Initialiser le contour  $C^0$  /\* Généralement donné par l'utilisateur \*/
4. **POUR**  $t$  allant de 0 à MAX\_ITER **FAIRE**
6. **POUR** point  $i$  de la courbe **FAIRE**
$$A = (f_x(C^t(i)), f_y(C^t(i)))$$
$$B = \alpha(C^t(i+1) - 2C^t(i) + C^t(i-1))$$
$$C = -\beta(C^t(i+2) - 4C^t(i+1) + 6C^t(i) - 4C^t(i-1) + C^t(i-2))$$
$$C^{t+1}(i) \approx C^t(i) + \delta(A + B + C)$$
7.  $C^t = C^{t+1}$

# Résultat



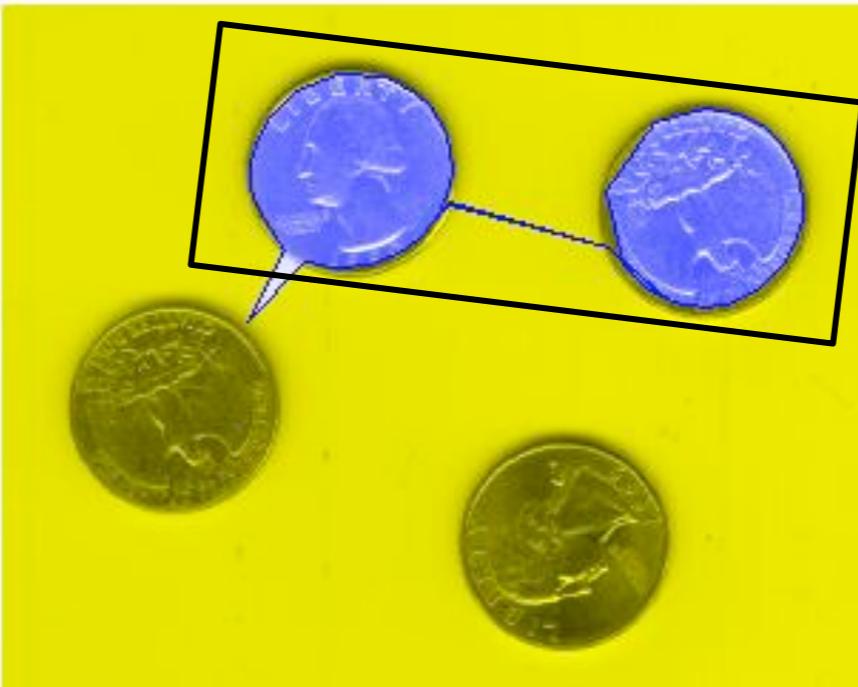
# Résultat

Avantage :

- segmentation locale et interactive

Inconvénients:

- incapacité pour une courbe de se scinder en 2
- aucune évolution de la courbe lorsqu'initialiser trop loin du contour
- la courbe peut « s'accrocher » à d'autres objets.
- pas évident de savoir quand arrêter l'évolution de la courbe.



# Résultat

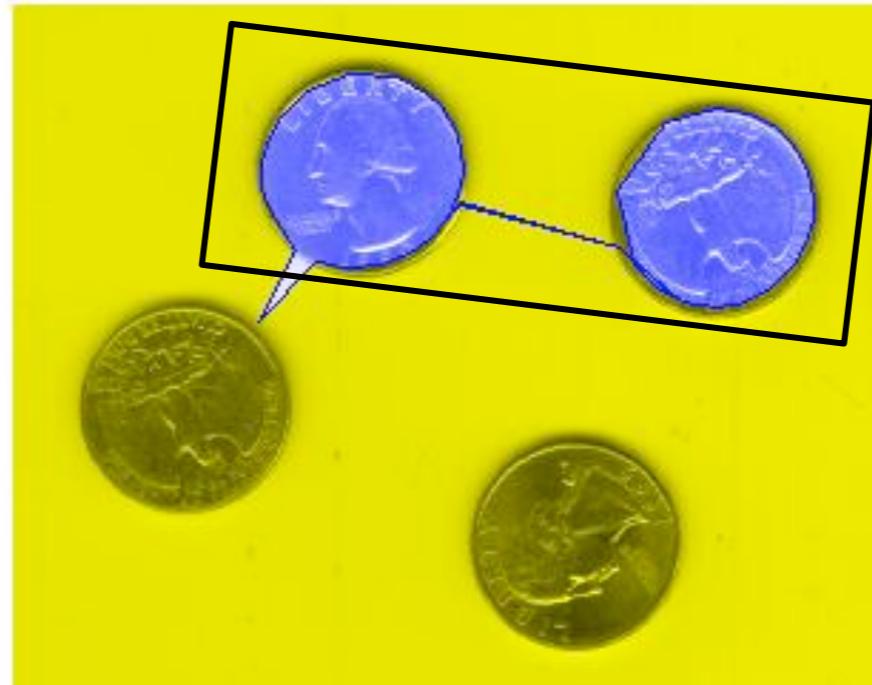
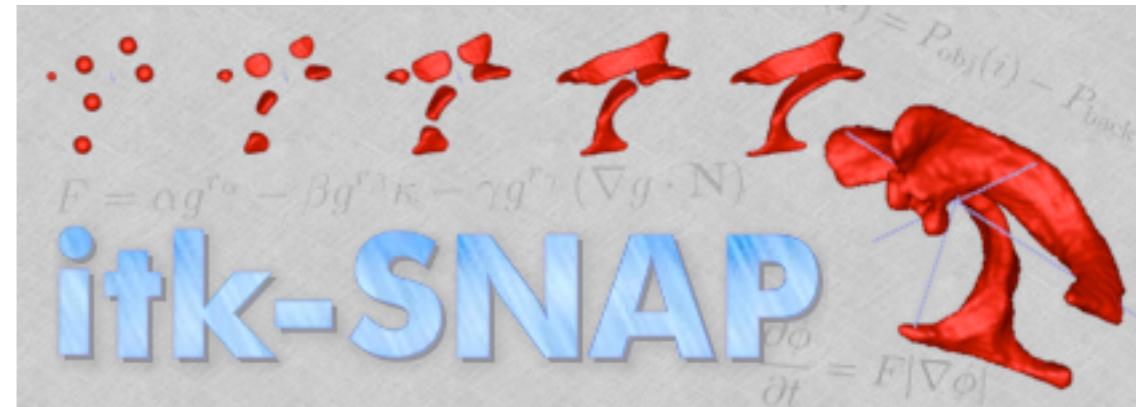
Avantage :

- segmentation locale et interactive

Inconvénients:

- incapacité pour une courbe de se scinder en 2
- aucune évolution de la courbe lorsqu'initialiser trop loin du contour
- la courbe peut « s'accrocher » à d'autres objets.
- pas évident de savoir quand arrêter l'évolution de la courbe.

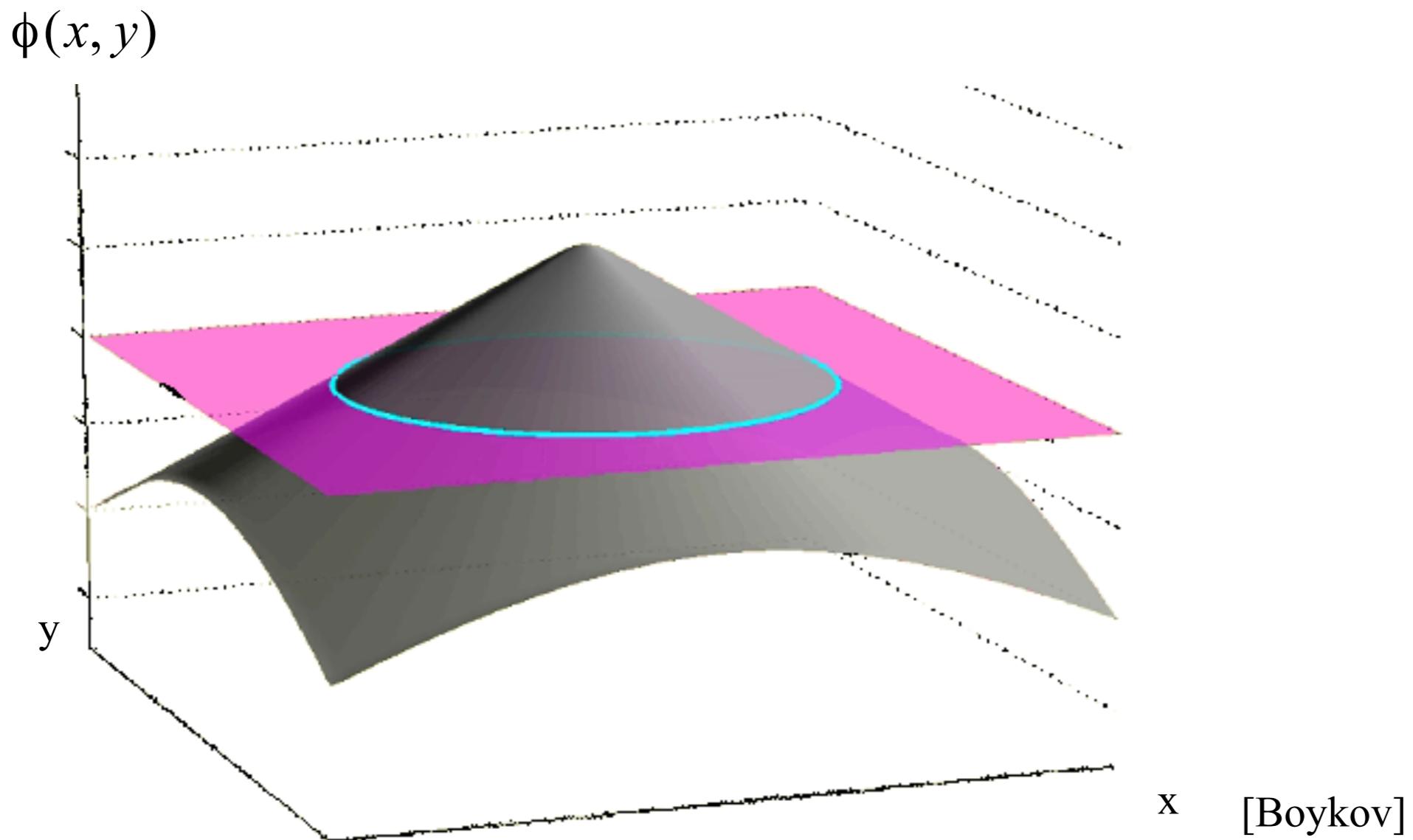
Essayez ITK-SNAP



# Les contours géodésiques actifs

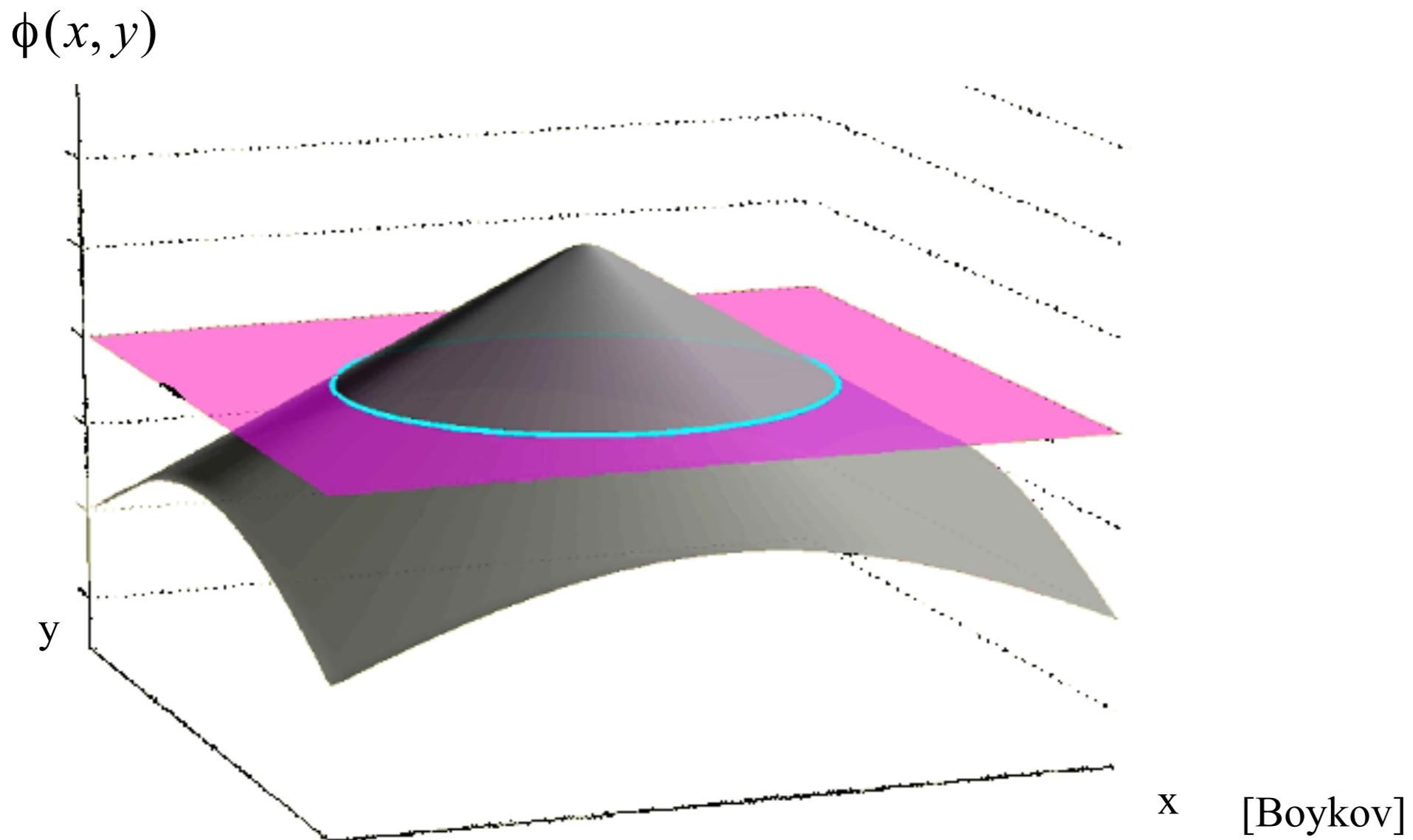
Caselles, Kimmel, Sapiro, “***Geodesic Active Contours***”, International Journal of Computer Vision, 22(1), 61-79, 1997.

# Les contours géodésiques actifs



$$C = \{(x, y) \mid \phi(x, y) = 0\}$$

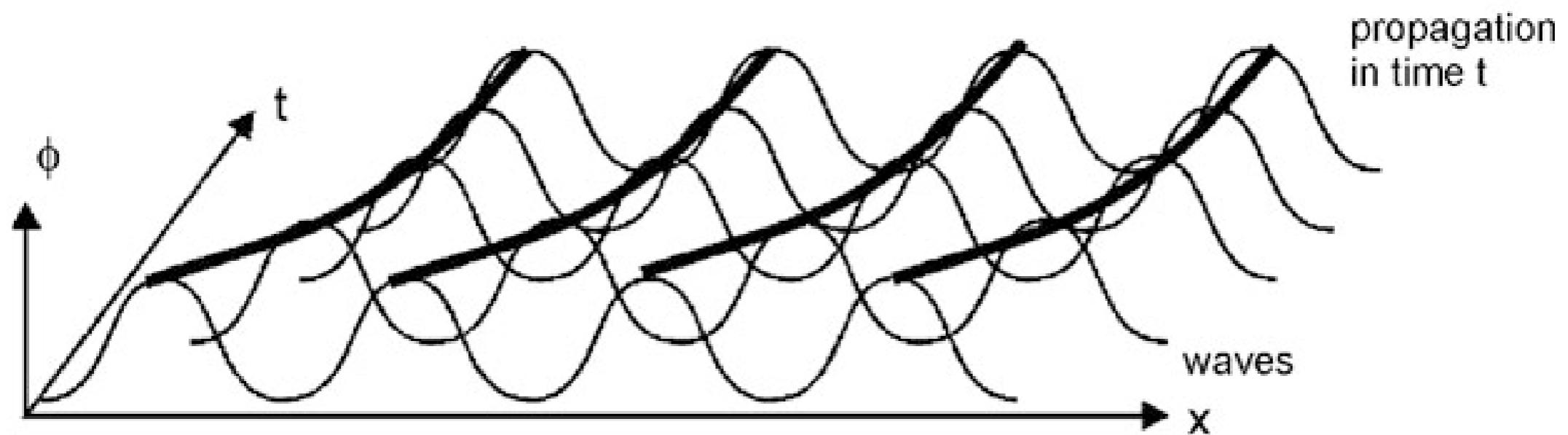
# Les contours géodésiques actifs



L'idée derrière les level-sets est d'associer la coube  $C(q)$  au niveau 0 d'une fonction  $\phi(x, y)$

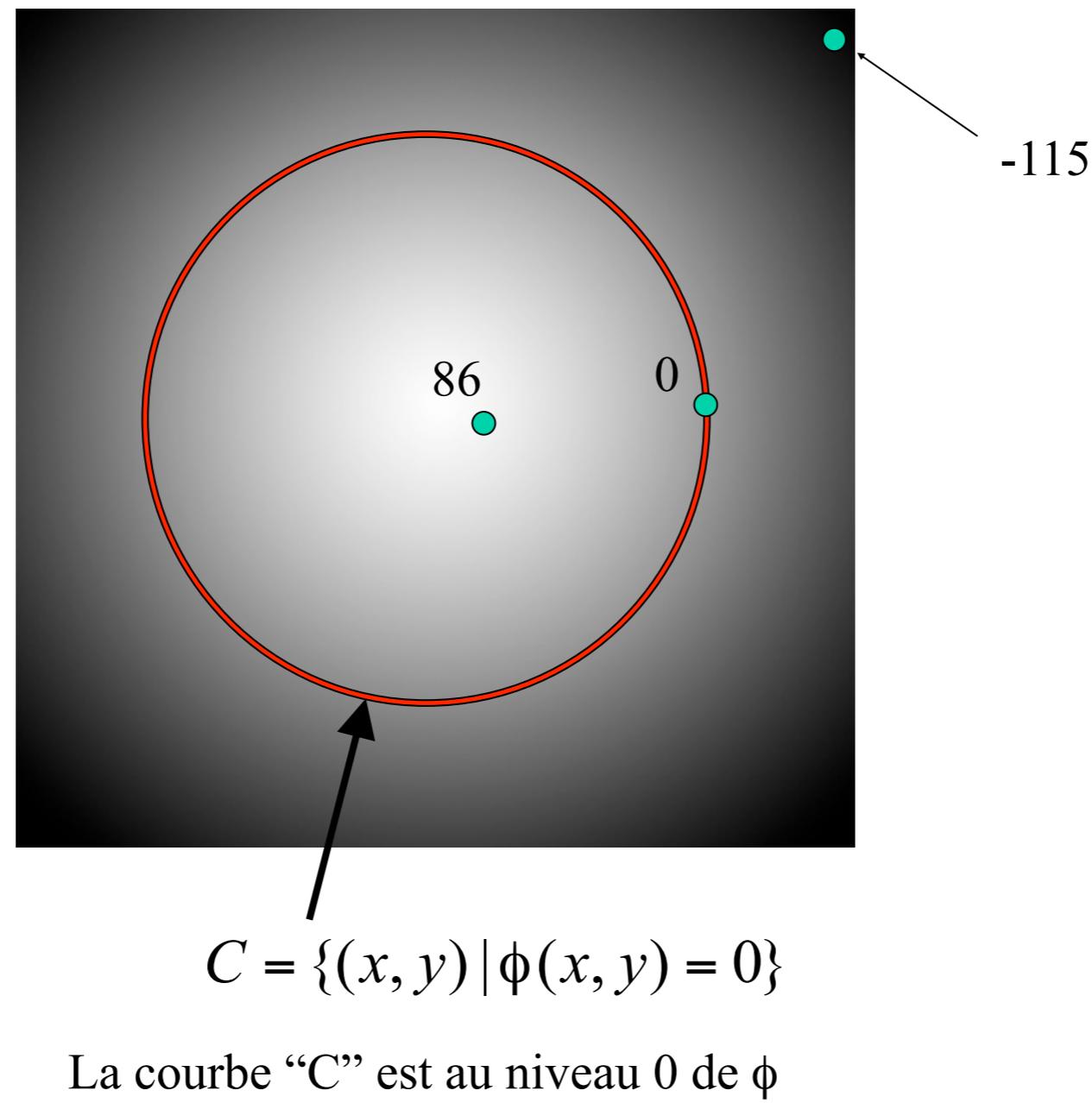
$$C = \{(x, y) \mid \phi(x, y) = 0\}$$

# Levelsets (courbes de niveaux)

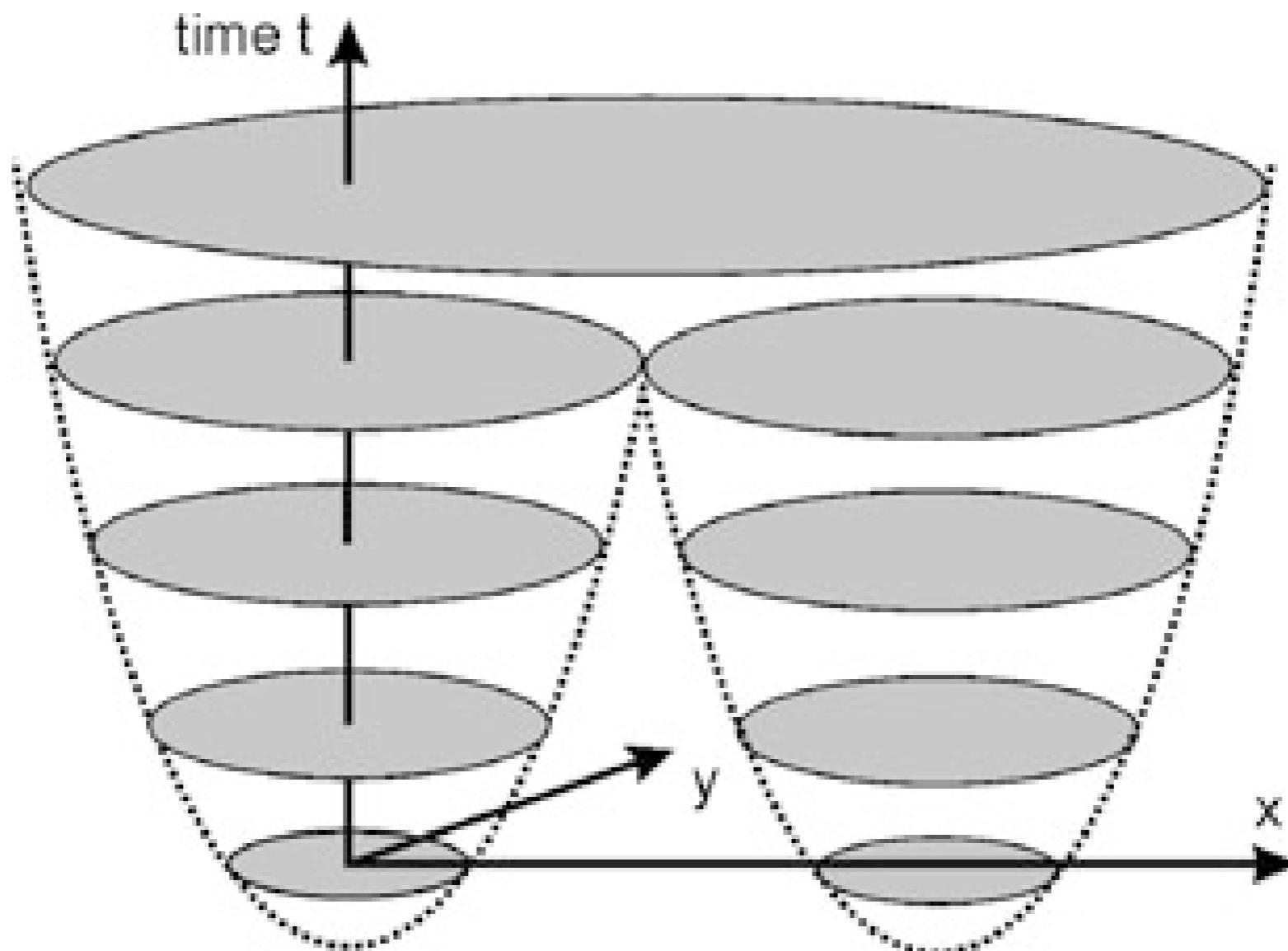


# Les contours géodésiques actifs

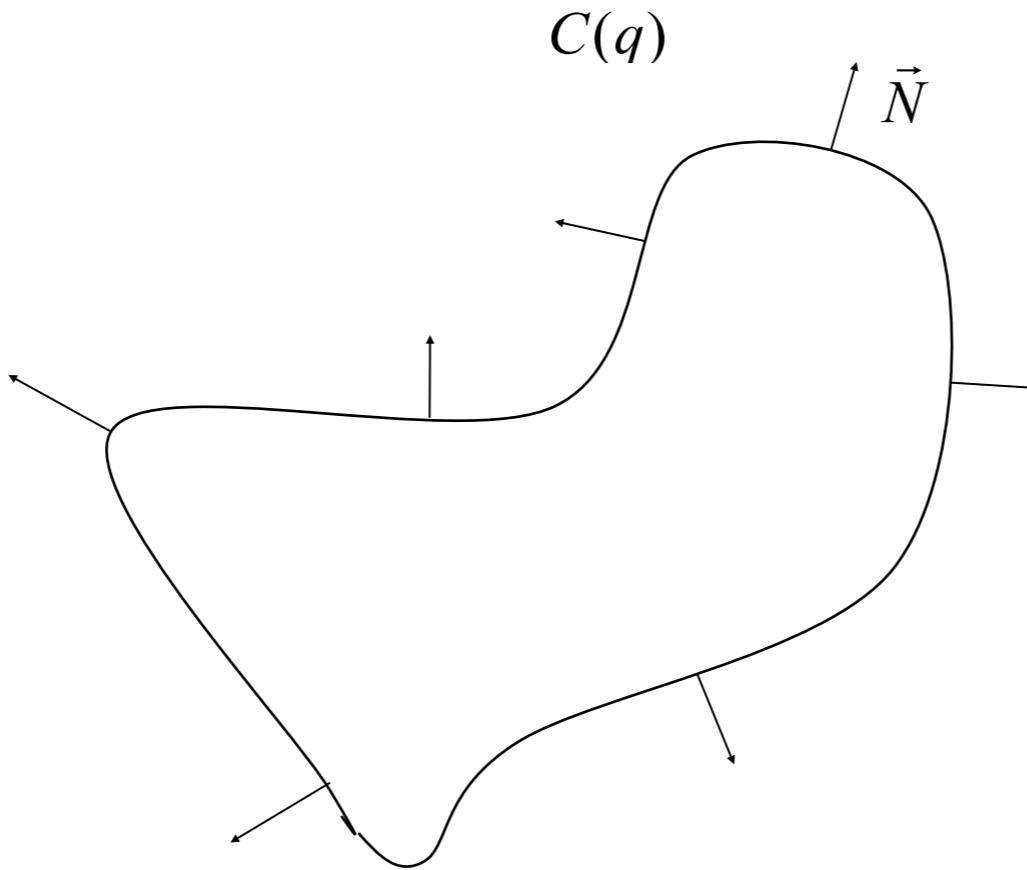
On représente souvent  $\phi(x, y)$  par une image “*height map*”. L’intensité d’un pixel correspond à la hauteur de  $\phi(x, y)$  à cet endroit. On appelle également  $\phi(x, y)$  une “*signed distance function*” car pour chaque pixel  $(x, y)$ , elle contient la distance au point le plus près sur la courbe.



# Robuste aux changements de topologies



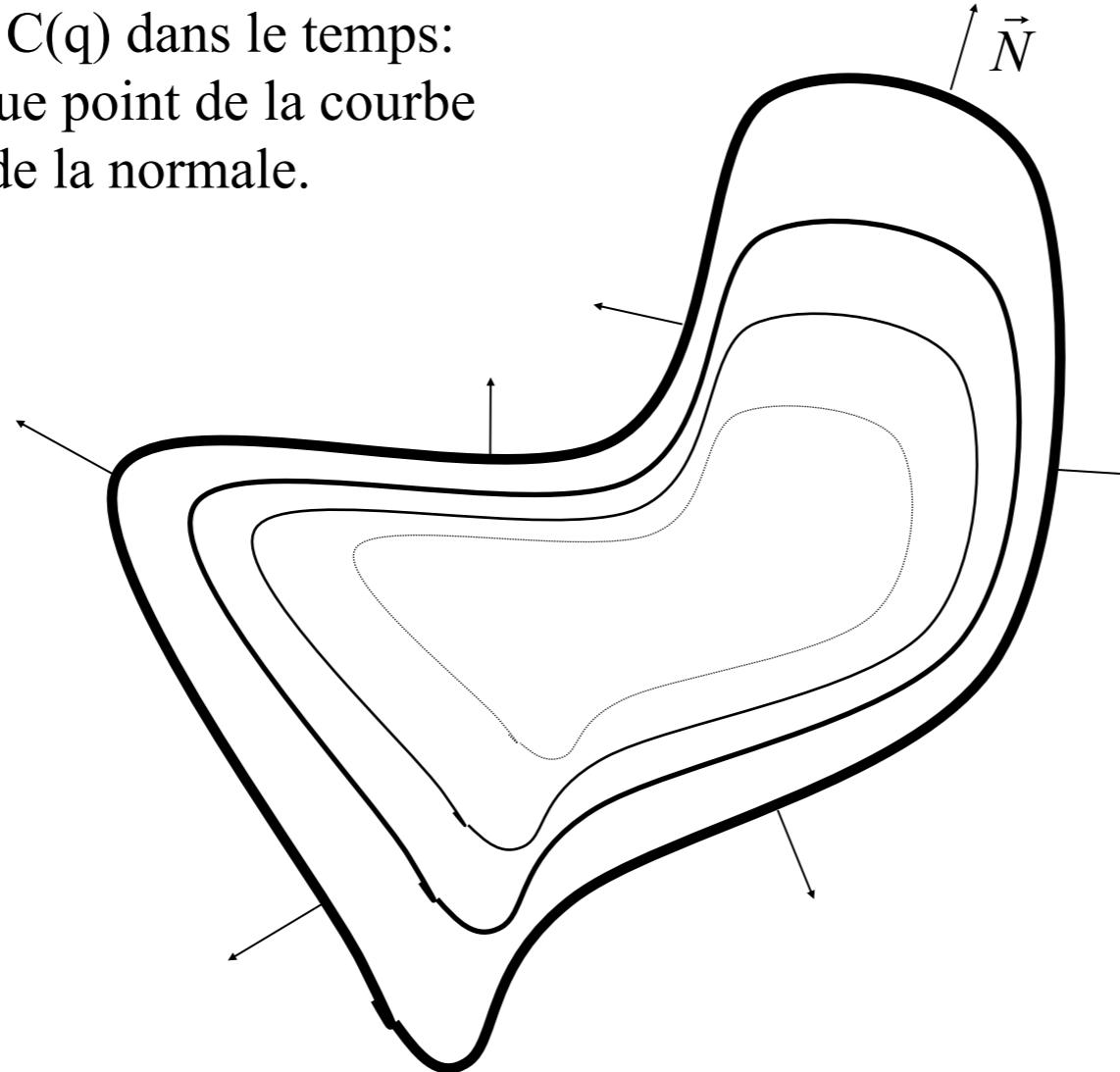
# Les contours géodésiques actifs



La courbe possède une normale  $\vec{N}$  en chacun de ses points

# Les contours géodésiques actifs

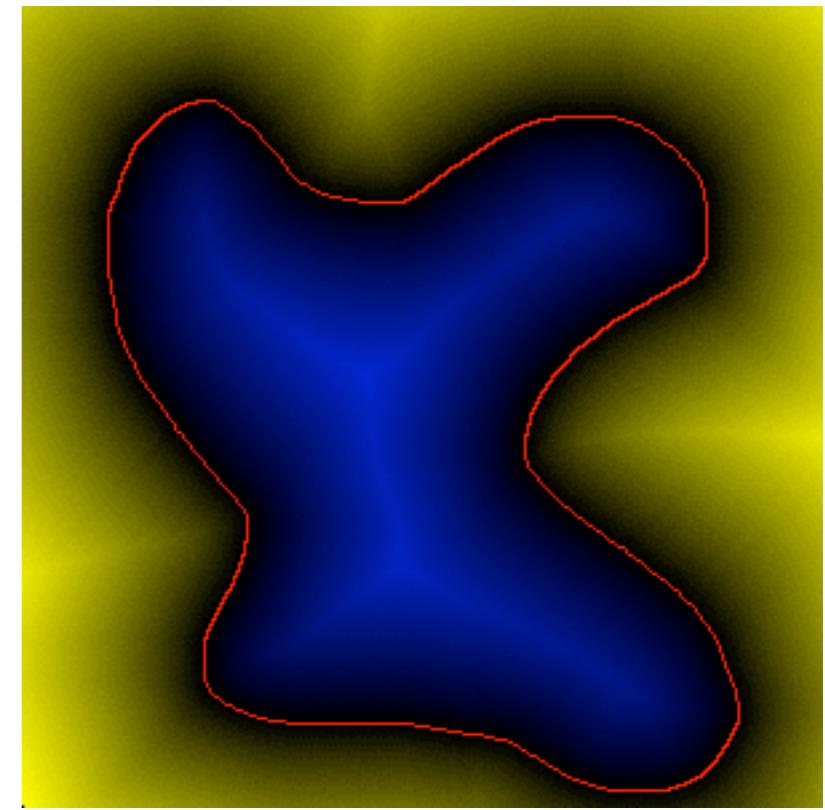
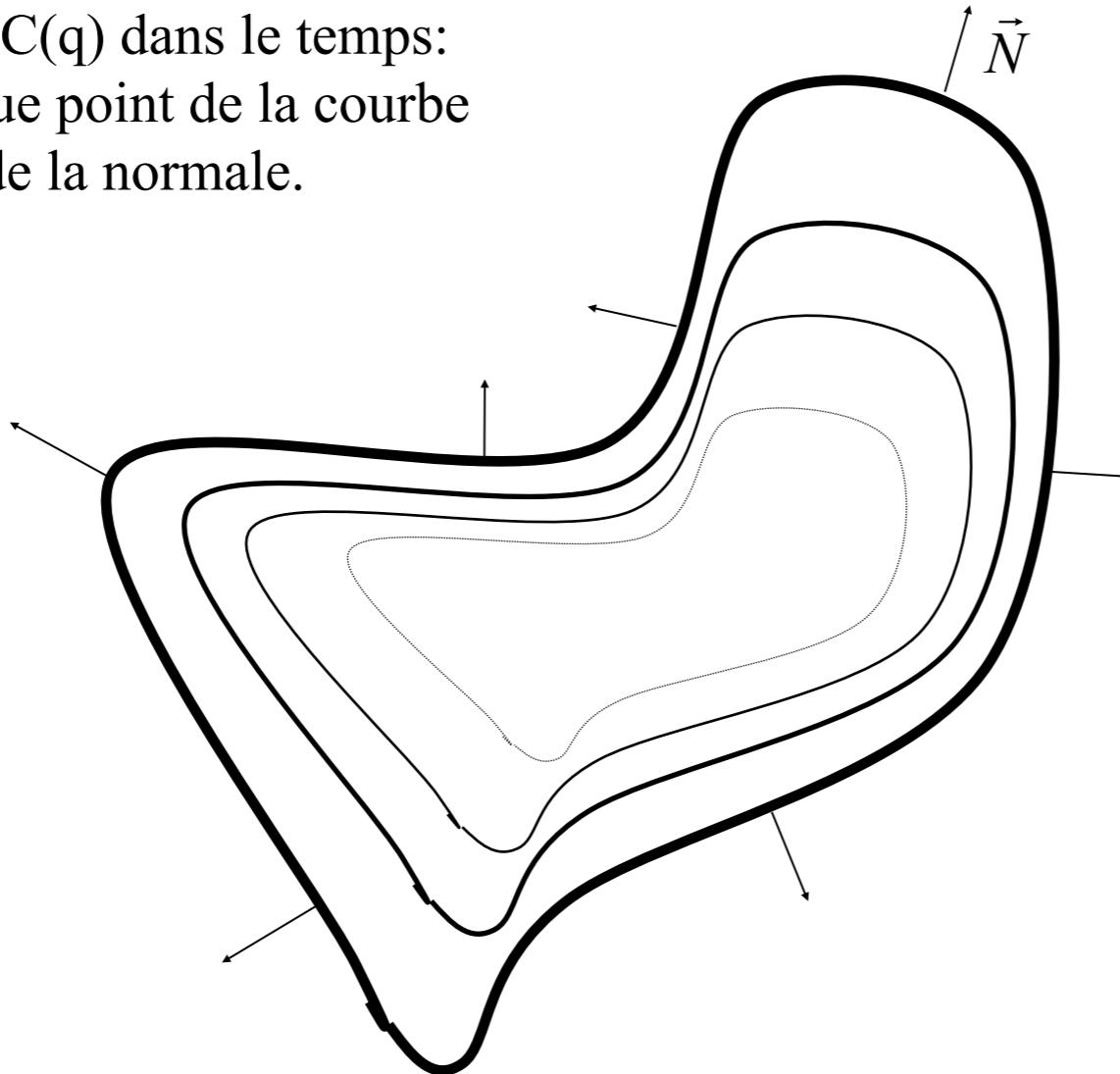
Évolution de  $C(q)$  dans le temps:  
on étire chaque point de la courbe  
en direction de la normale.



$$\frac{\partial C}{\partial t} = F \vec{N}$$

# Les contours géodésiques actifs

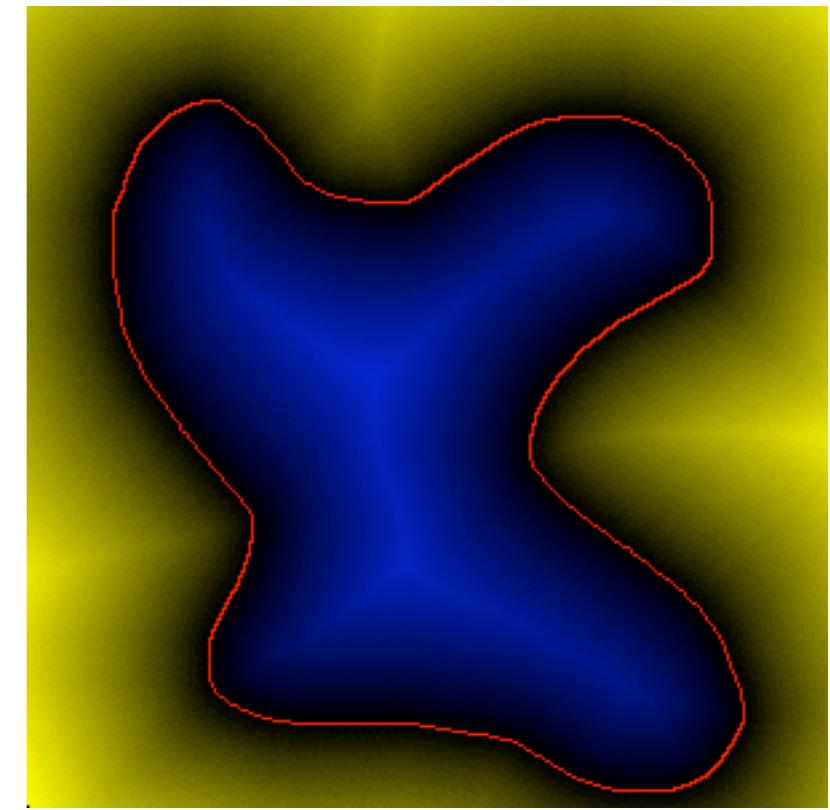
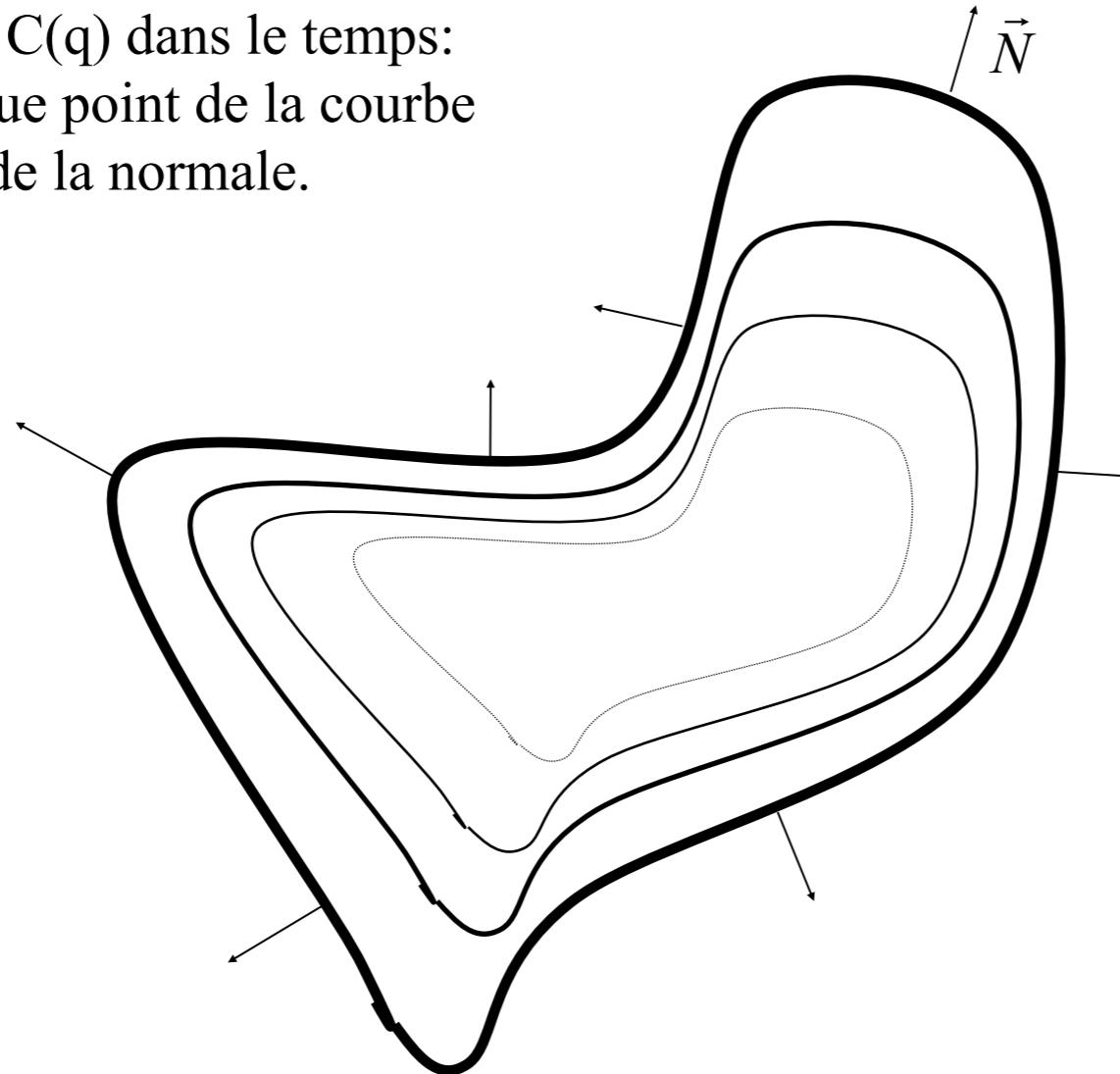
Évolution de  $C(q)$  dans le temps:  
on étire chaque point de la courbe  
en direction de la normale.



$$\frac{\partial C}{\partial t} = F \vec{N}$$

# Les contours géodésiques actifs

Évolution de  $C(q)$  dans le temps:  
on étire chaque point de la courbe  
en direction de la normale.

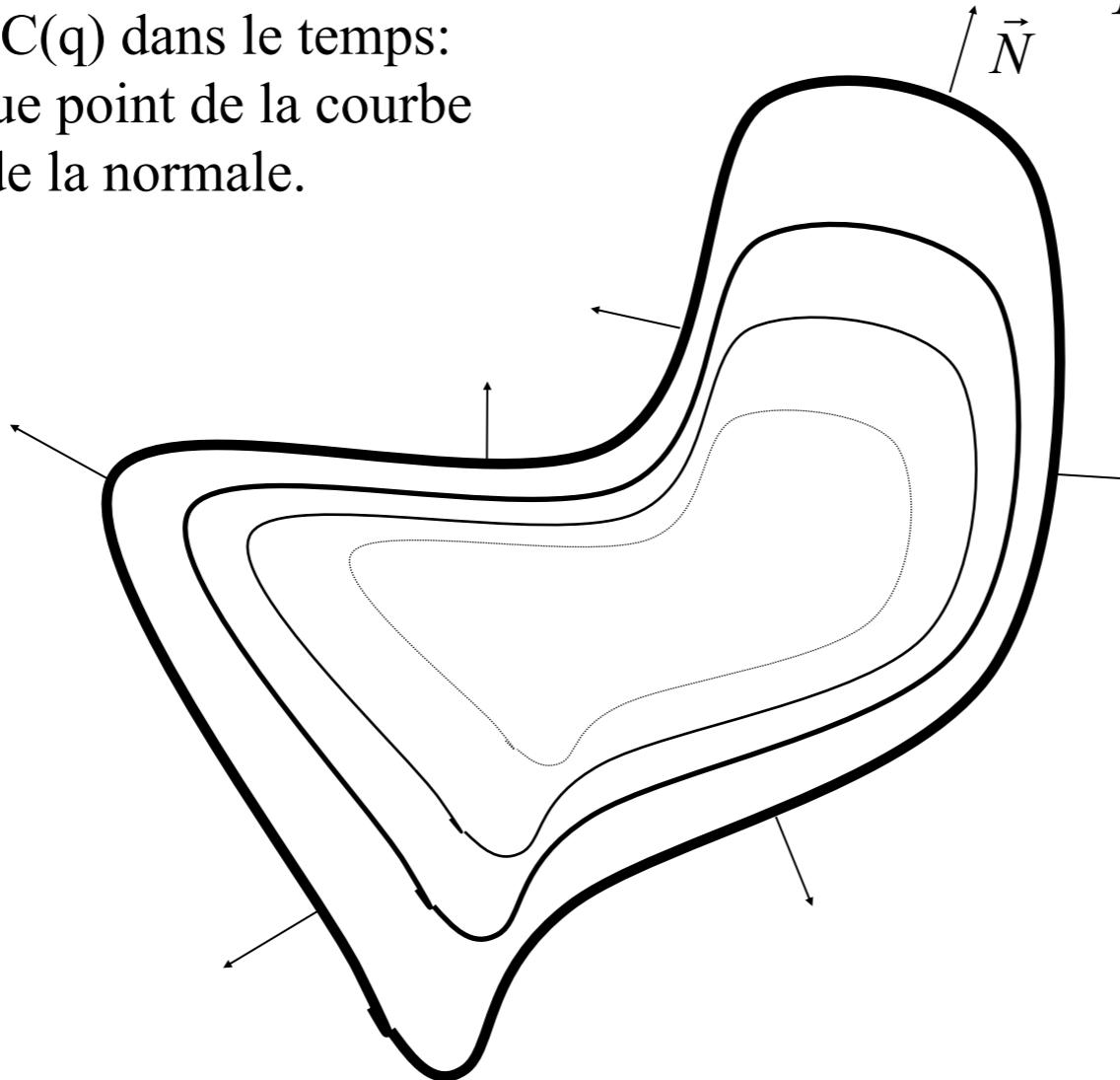


$$\frac{\partial C}{\partial t} = F \vec{N}$$

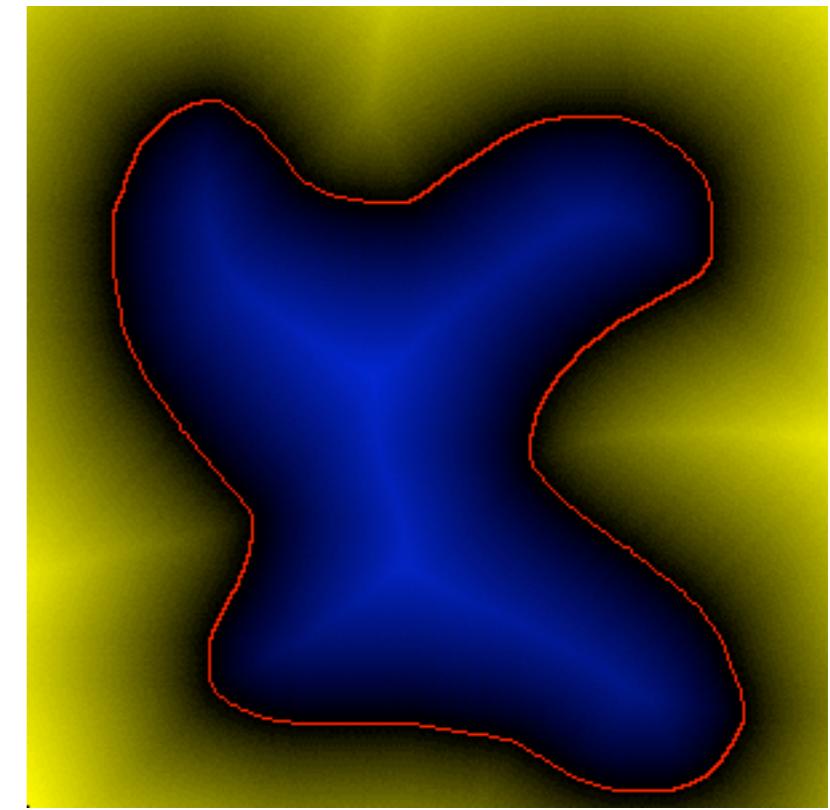
Si  $F > 0 \rightarrow$  la courbe s'étire  
Si  $F < 0 \rightarrow$  la courbe rétrécit  
Si  $F = 0 \rightarrow$  la courbe ne bouge plus

# Les contours géodésiques actifs

Évolution de  $C(q)$  dans le temps:  
on étire chaque point de la courbe  
en direction de la normale.



$$\vec{N} = \frac{\nabla\phi}{|\nabla\phi|}$$



$$\frac{\partial C}{\partial t} = F\vec{N}$$

Si  $F>0 \rightarrow$  la courbe s'étire  
Si  $F<0 \rightarrow$  la courbe rétrécit  
Si  $F=0 \rightarrow$  la courbe ne bouge plus

# Les contours géodésiques actifs

On suppose que le contours évolue dans le temps suivant :  $\frac{dC}{dt} = F\vec{N}$

À tout temps, la courbe « C » est au niveau zéro de  $\phi$  :  $\phi(C(t), t) = 0$

En combinant ces deux équations, on obtient :

$$0 = \frac{d\phi(C(t), t)}{dt} = \nabla\phi \frac{\partial(C)}{\partial t} + \frac{\partial\phi}{\partial t}$$

et donc

$$\frac{\partial\phi}{\partial t} = -\nabla\phi \frac{d(C)}{dt} = -\nabla\phi F\vec{N}$$

Étant donné que  $\vec{N} = \frac{\nabla\phi}{|\nabla\phi|}$  on obtient l'équation des level sets:

$$\boxed{\frac{\partial\phi}{\partial t} = -F|\nabla\phi|}$$

# Les contours géodésiques actifs

$$\frac{\partial \phi}{\partial t} = -F |\nabla \phi|$$

La grande question est de déterminer de façon appropriée la FORCE «  $F$  » à appliquer sur  $\phi$

Le tout dépend de l'application :

- simulation de fluides compressibles (air, fumée, etc.)
- simulation de fluides incompressible (eau, vase, lave, etc)
- segmentation 2D, 3D.
- ...

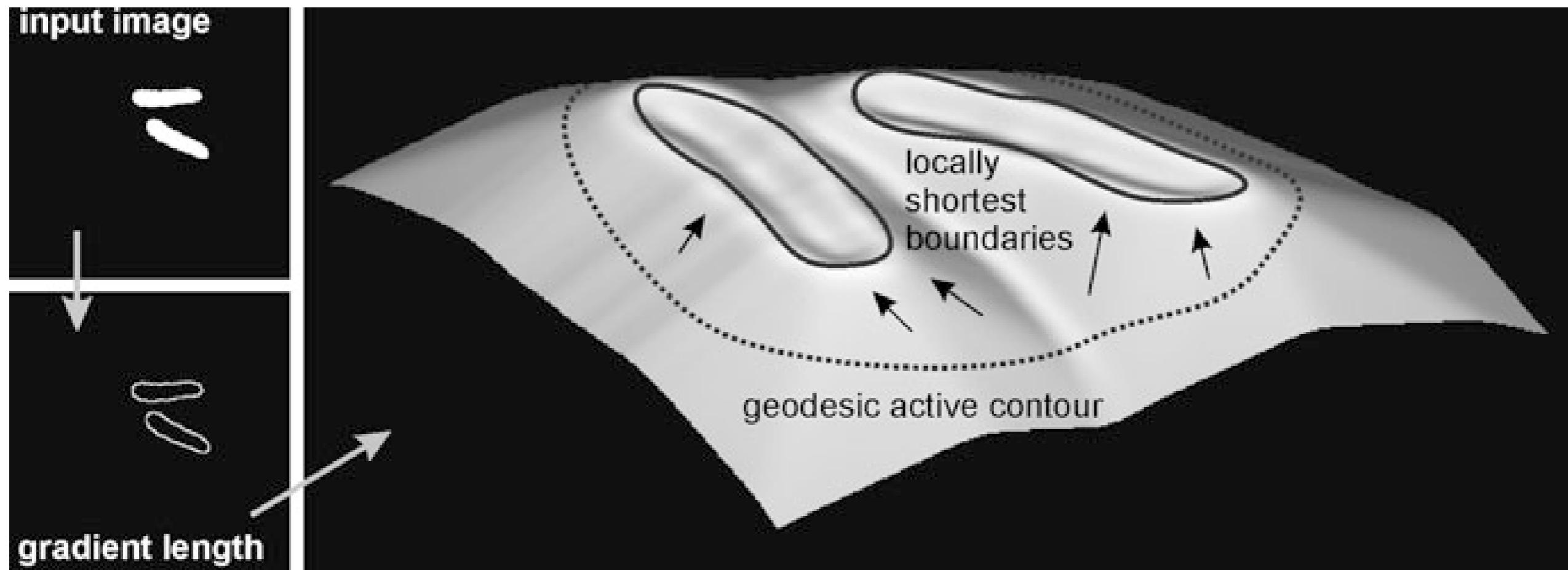
# Les contours géodésiques actifs

$$\frac{\partial \phi}{\partial t} = -F |\nabla \phi|$$

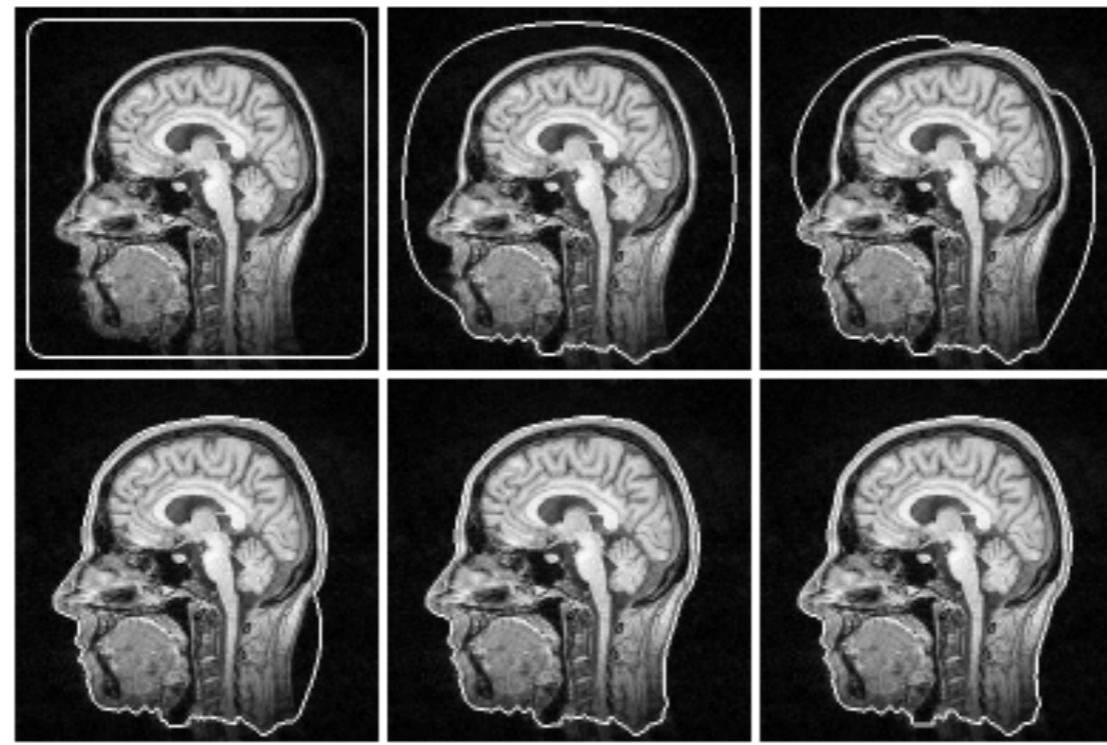
La réponse par l'approche « contours géodésiques actifs » :

$$\frac{\partial \phi}{\partial t} = |\nabla \phi| \operatorname{div} \left( g(\nabla I) \frac{\nabla \phi}{|\nabla \phi|} \right)$$

# Les contours géodésiques actifs (GAC)



# Les contours géodésiques actifs



[Weickert 00]

$$C = \arg \min E_{ext}(C, I) + E_{int}(C)$$

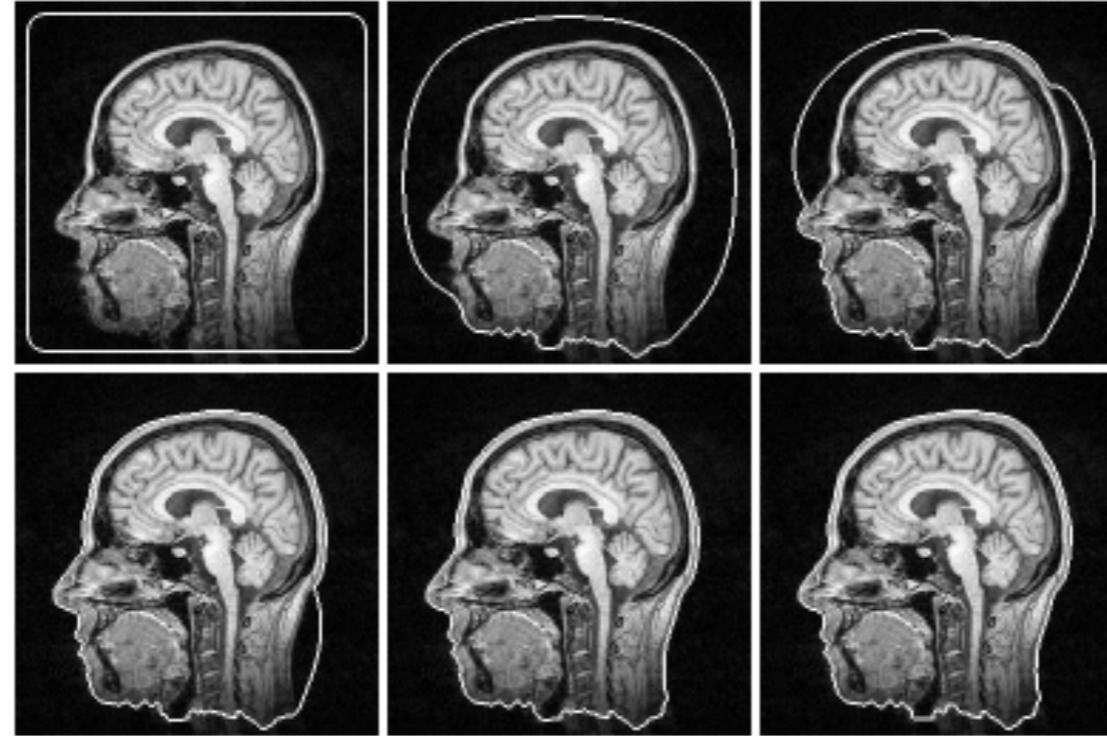
$$= \arg \min \int_0^1 g(\nabla I(C(q))) dq + \lambda \int_0^1 |C'(q)|^2 dq$$

$$\int_0^1 g(\nabla I(C(q))) dq$$

retourne une petite valeur lorsque  $C(q)$  suit le contours d'un objet dans  $I$ . Sinon retourne une valeur élevée.

$$\int_0^1 |C'(q)|^2 dq \quad \text{faible lorsque la courbure globale de } C \text{ est « smooth »}.$$

# Les contours géodésiques actifs



[Weickert 00]

$$\begin{aligned} C &= \arg \min E_{ext}(C, I) + E_{int}(C) \\ &= \arg \min \int_0^1 g(\nabla I(C(q))) dq + \lambda \int_0^1 |C'(q)|^2 dq \end{aligned}$$

On peut démontrer (mais ce n'est pas trivial) que cette dernière équation est équivalente à

$$C = \arg \min \int_0^1 g(\nabla I(C(q))) C'(q) | dq$$

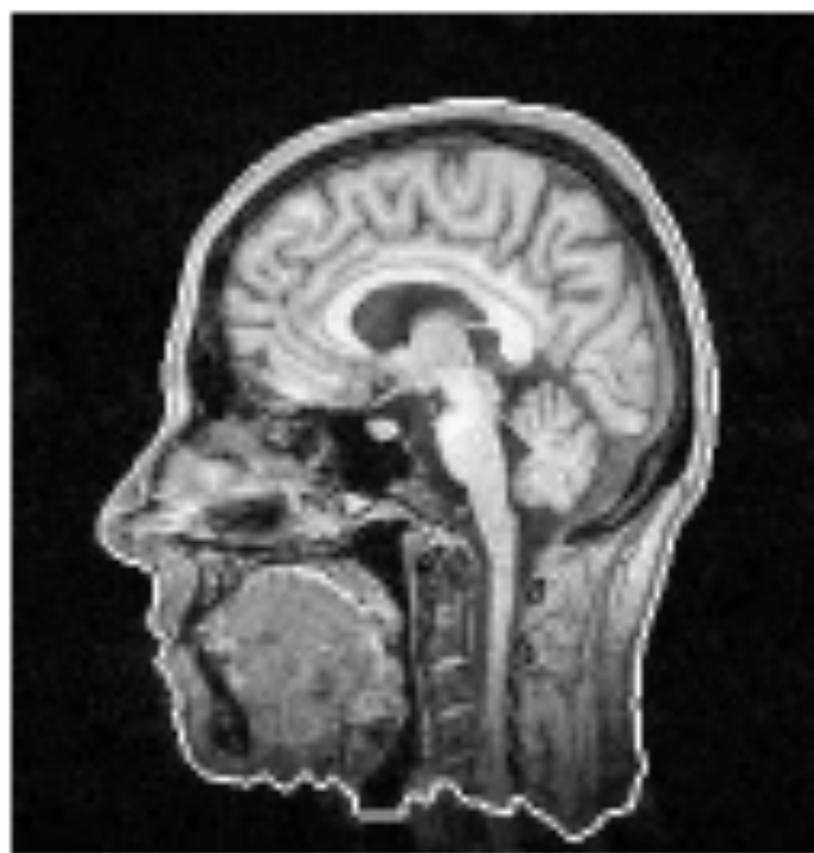
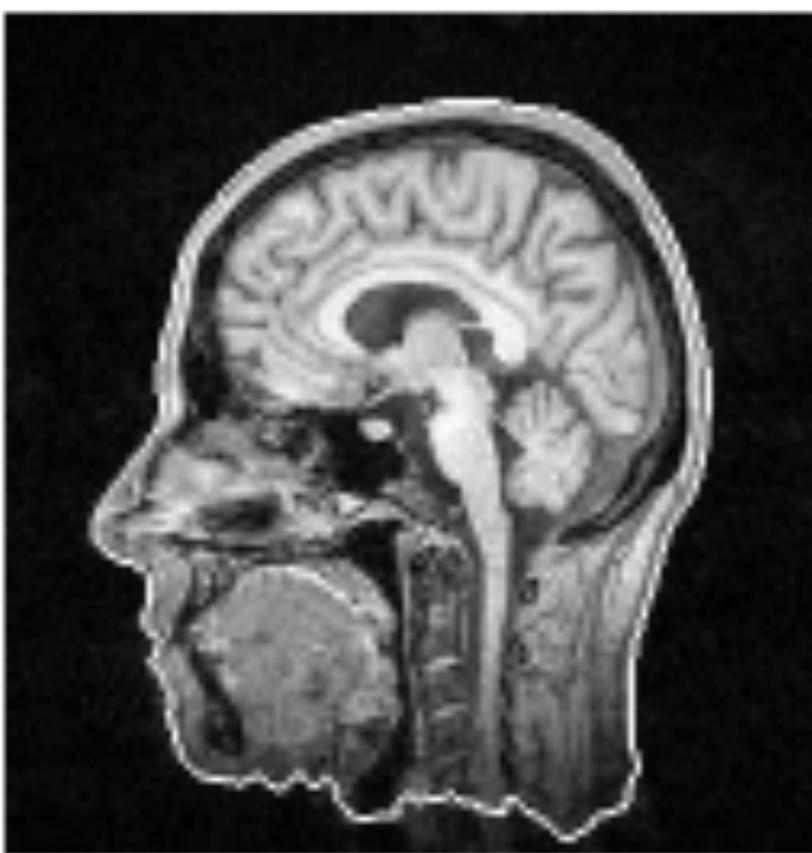
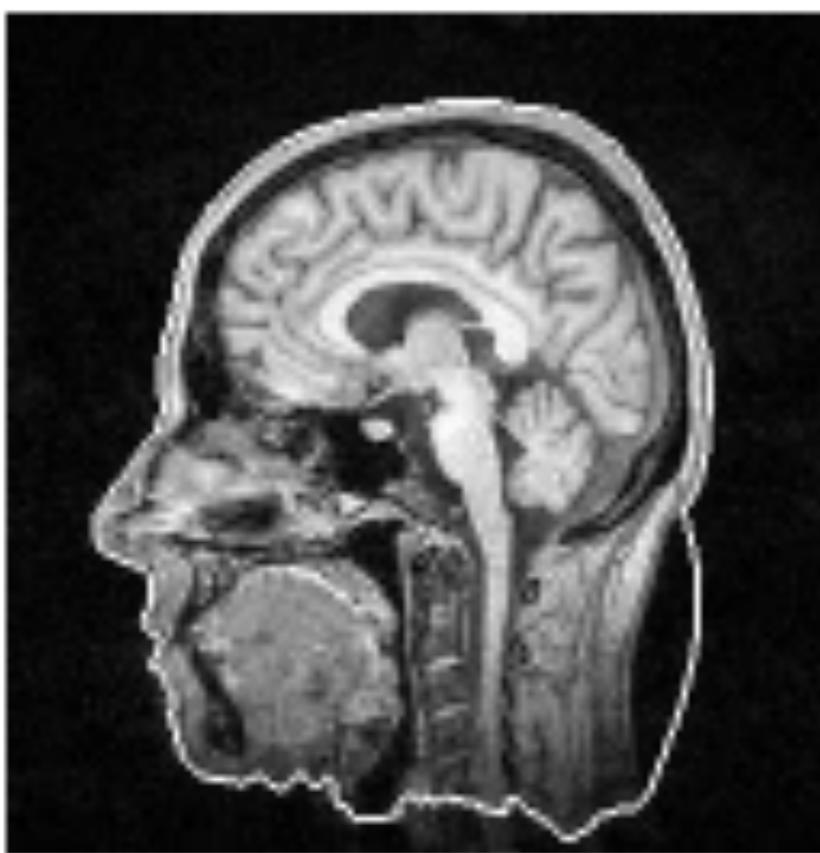
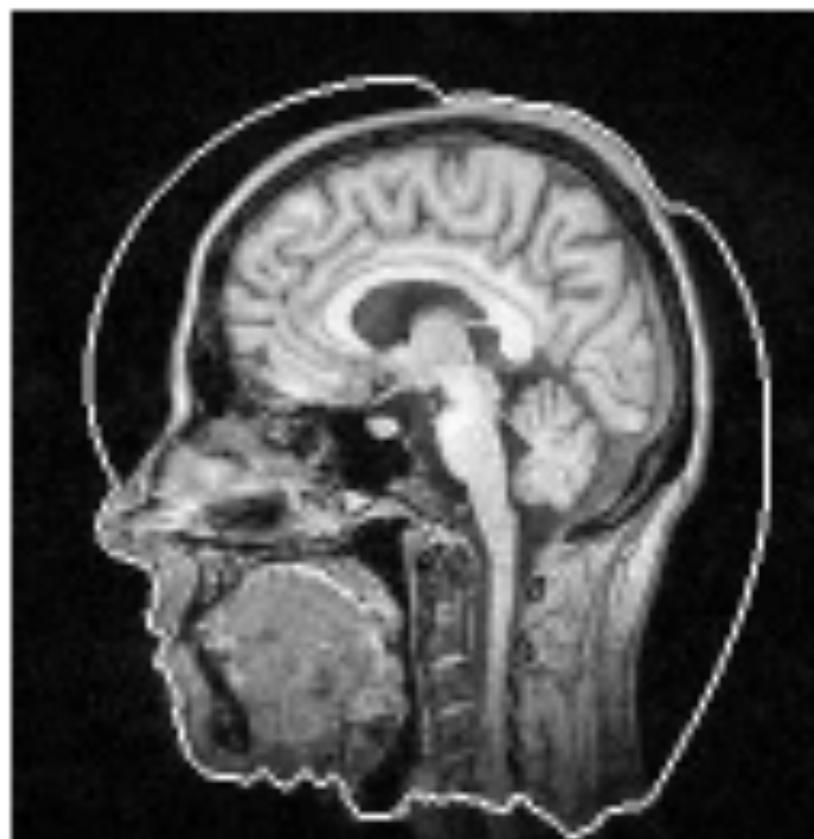
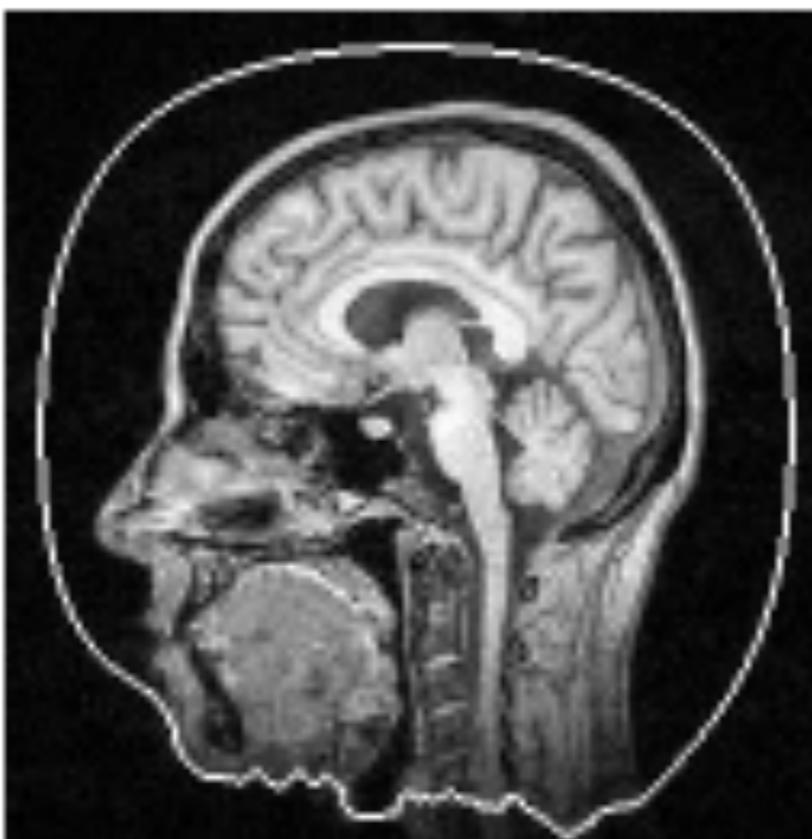
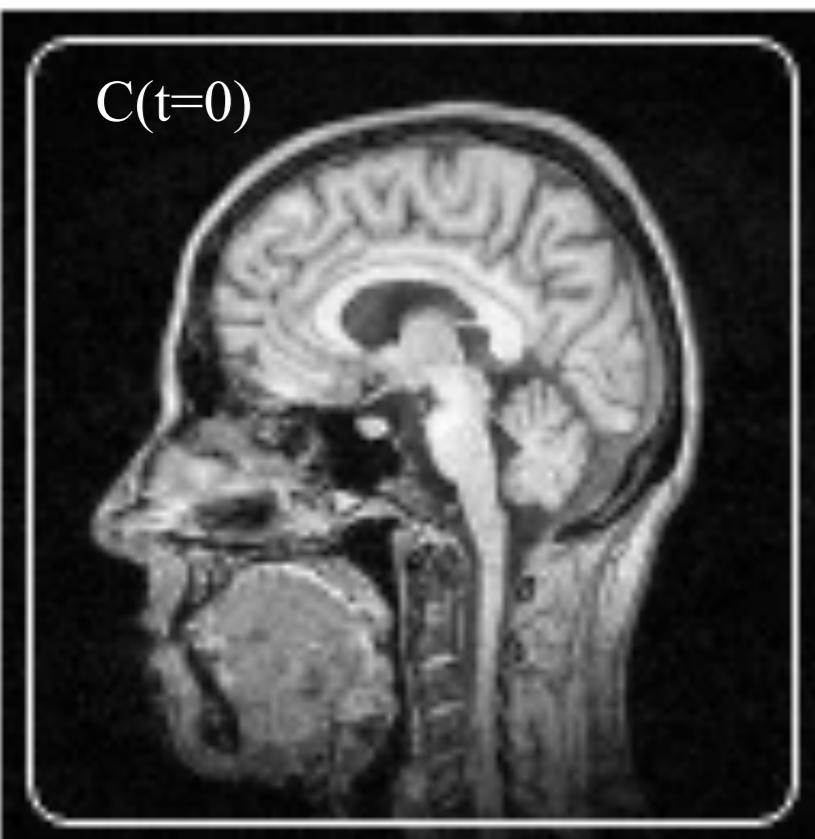
# Les contours géodésiques actifs

$$C = \arg \min \int_0^1 g(\nabla I(C(q)) C'(q) | dq$$

À l'aide de la formule **d'Euler-Lagrange**, on peut démontrer que partant d'une courbe  $C(t=0)$  on peut atteindre la courbe "optimale" en faisant évoluer  $C$  suivant l'équation suivante:

$$\frac{\partial C(t)}{\partial t} = g(\nabla I) \kappa \vec{N} - (\nabla g(\nabla I) \times \vec{N}) \vec{N}$$

où  $\kappa$  est la "courbure euclidienne" de  $C$ .



# Les contours géodésiques actifs

En résumé, un contours C qui évolue suivant :  $\frac{\partial C}{\partial t} = F \vec{N}$

est équivalent à l'évolution du levelset  $\phi$  :  $\frac{\partial \phi}{\partial t} = -F |\nabla \phi|$

$$\frac{\partial C(t)}{\partial t} = g(\nabla I) \kappa \vec{N} - (\nabla g(\nabla I) \times \vec{N}) \vec{N}$$

En combinant ces trois équations ensemble, on obtient l'équation suivante:

$$\frac{\partial \phi}{\partial t} = |\nabla \phi| \operatorname{div} \left( g(\nabla I) \frac{\nabla \phi}{|\nabla \phi|} \right)$$

C'est l'équation qui décrit comment  $\phi$  évolue dans le temps

# Les contours géodésiques actifs

$$\frac{\partial \phi}{\partial t} = |\nabla \phi| \operatorname{div} \left( g(\nabla I) \frac{\nabla \phi}{|\nabla \phi|} \right)$$

La question maintenant est de savoir **comment résoudre cette équation.**

La façon la plus facile de résoudre cette équation est de façon « **explicite** »

Sinon, on le fait de façon « **implicite** » avec les « levelsets »

# Les contours géodésiques actifs

$$\frac{\partial \phi}{\partial t} = |\nabla \phi| \operatorname{div} \left( g(\nabla I) \frac{\nabla \phi}{|\nabla \phi|} \right)$$

La question maintenant est de savoir **comment résoudre cette équation.**

La façon la plus facile de résoudre cette équation est de façon « **explicite** »

Sinon, on le fait de façon « **implicite** » avec les « levelsets »

[Osher: Levelset methods]

[Notes IMN559/IMN786 Pierre-Marc Jodoin]

# Les contours géodésiques actifs

Nous savons que

$$g(\nabla I|_{ij}) \begin{cases} \text{petite valeur lorsque } |\nabla I|_{ij} \text{ est élevée} \\ \text{grosse valeur lorsque } |\nabla I|_{ij} \text{ est faible} \end{cases}$$

Pour ce faire, [Caselles et al. 97] recommandent

$$g(\nabla I|_{ij}) = \frac{1}{1 + \frac{|\nabla I|_{ij}^2}{\lambda^2}}$$

# Les contours géodésiques actifs

Ce dernier algorithme résout l'équation suivante:

$$\frac{\partial \phi}{\partial t} = |\nabla \phi| \operatorname{div} \left( g(\nabla I) \frac{\nabla \phi}{|\nabla \phi|} \right)$$

Malheureusement, cet algorithme est TRES TRES TRES (...) TRES lent. Pour accélérer la convergence, [Caselles et al. 97] proposent l'équation suivante:

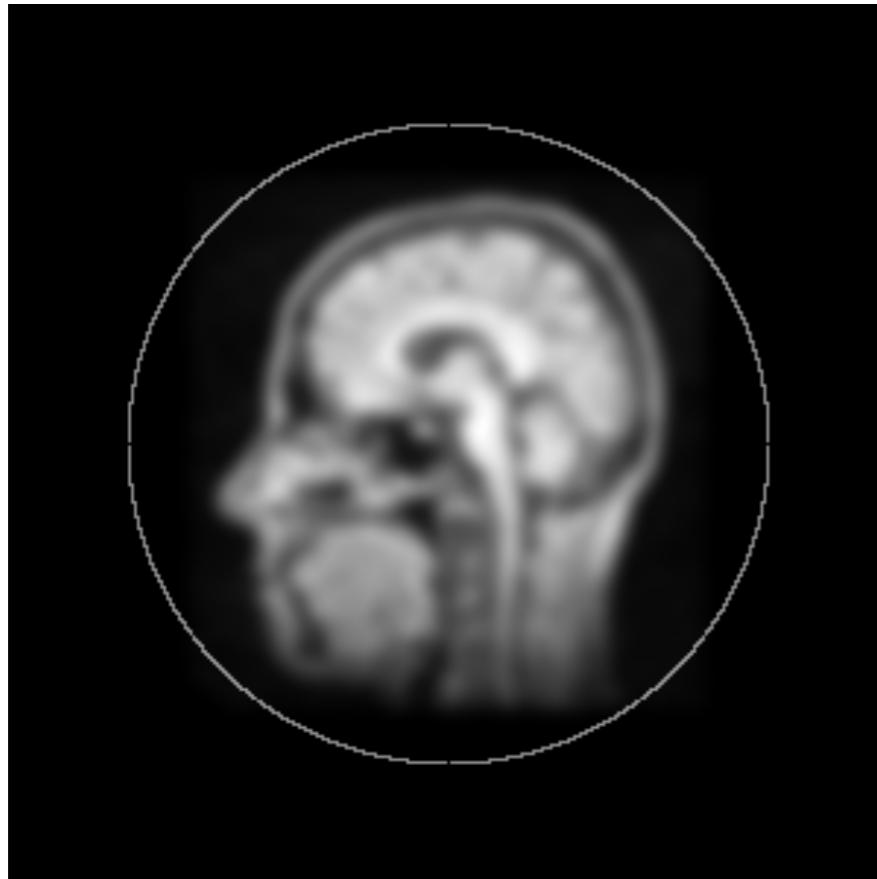
$$\frac{\partial \phi}{\partial t} = |\nabla \phi| \operatorname{div} \left( g(\nabla I) \frac{\nabla \phi}{|\nabla \phi|} \right) + \underline{c |\nabla \phi| g(\nabla I)}$$

où « c » pousse la courbe vers l'extérieur lorsque  $c>0$  et vers l'intérieur lorsque  $c<0$

# Les contours géodésiques actifs

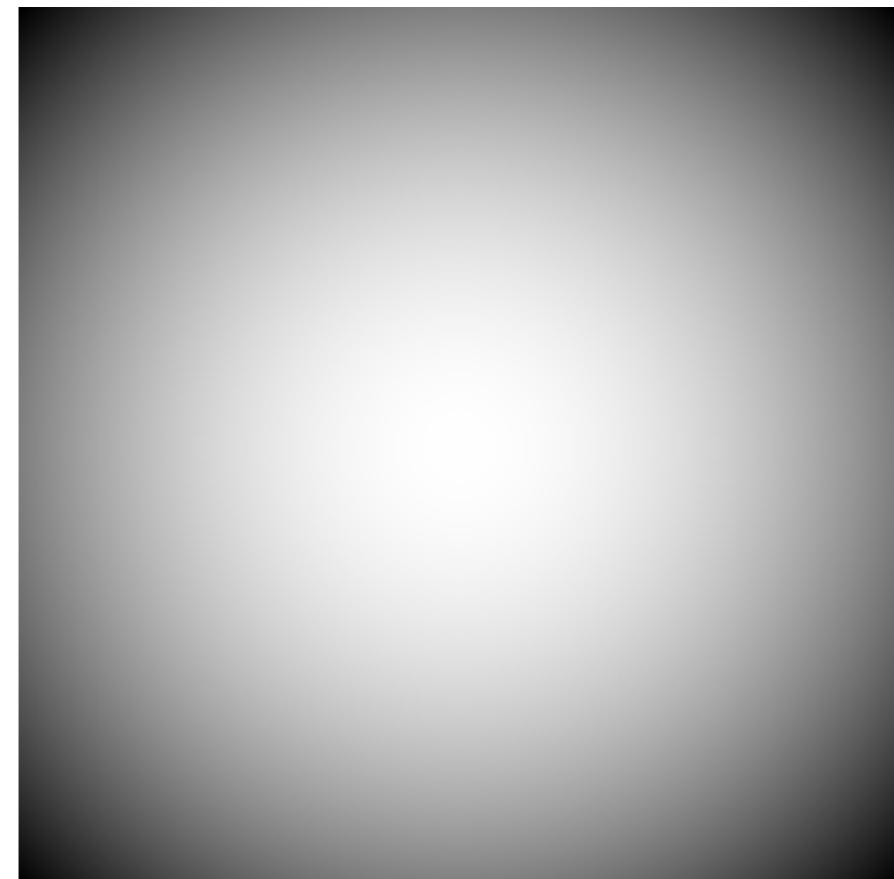
Animations

$$\sigma = 3, \lambda = 3, \tau = 0.25 \text{ et } c = -0.1$$



$$I_\sigma + C$$

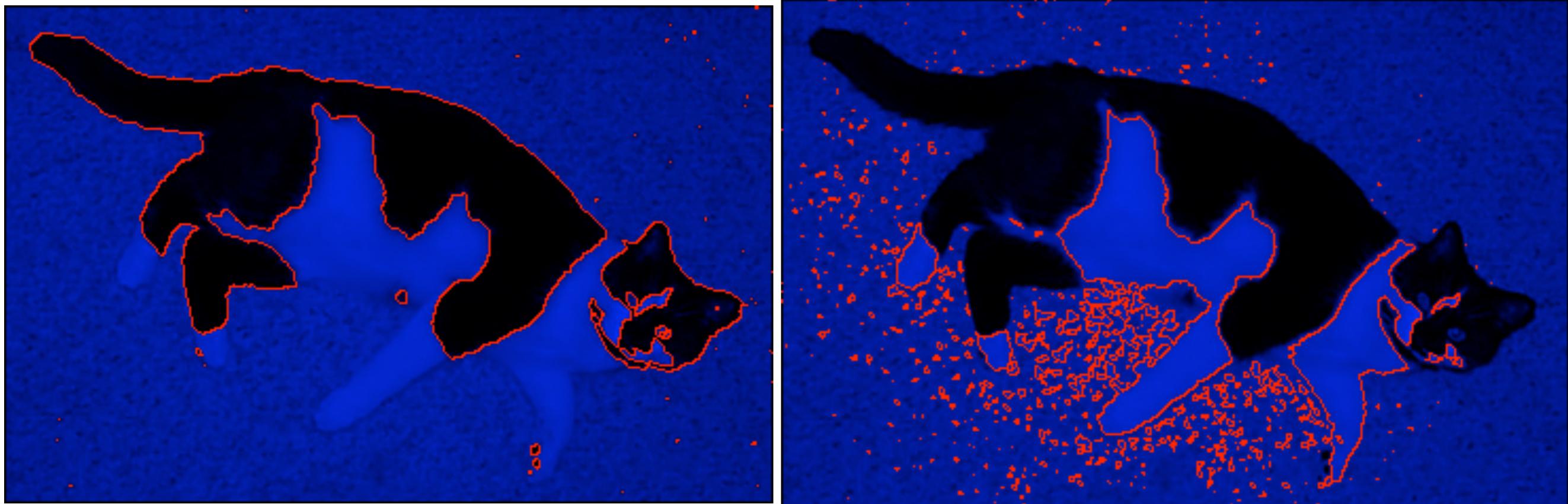
$$\text{où } C = \{(x, y) \mid \phi(x, y) = 0\}$$



$$\phi(x, y)$$

# Les contours géodésiques actifs

Animations



# Les contours géodésiques actifs

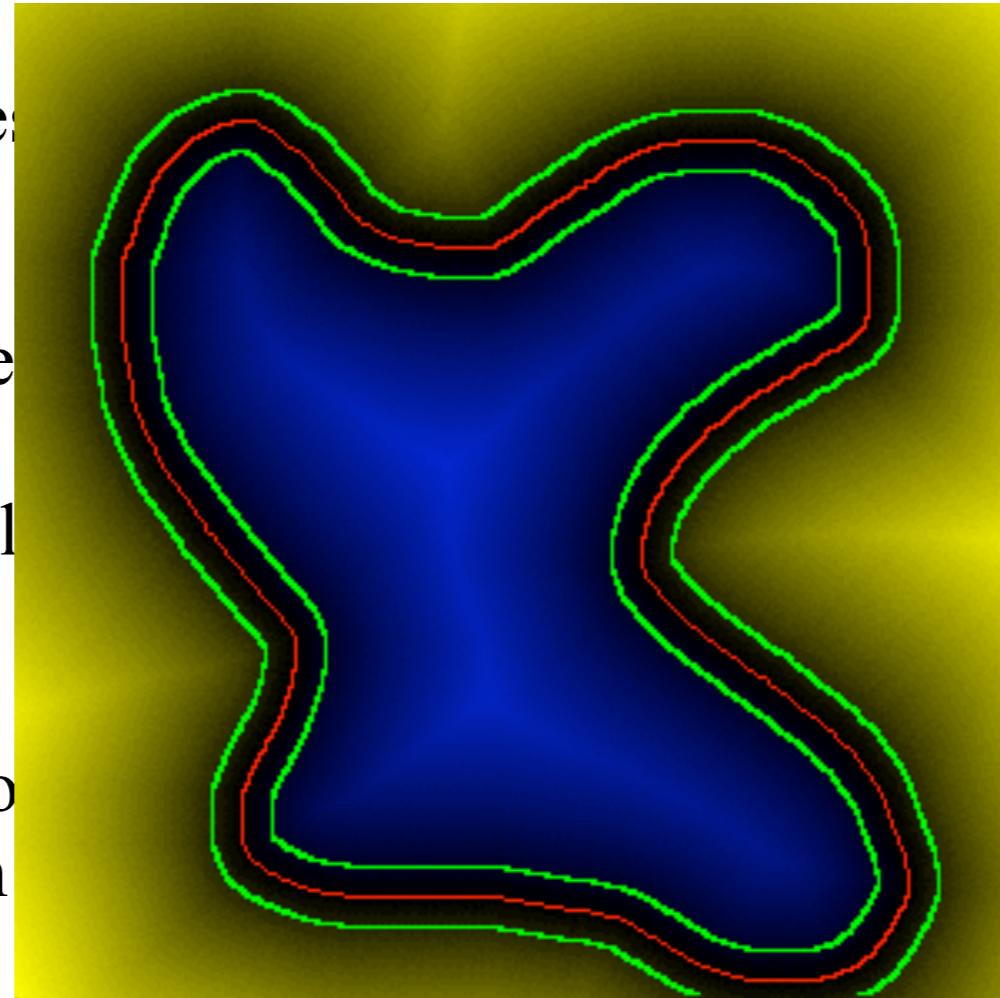
Ce qu'il faut retenir des GAC:

1. un coutour est le niveau Zéro d'une fonction d'élévation  $\phi(x, y)$
2. On cherche à solutionner la fonction:  $\frac{\partial \phi}{\partial t} = |\nabla \phi| \operatorname{div} \left( g(\nabla I) \frac{\nabla \phi}{|\nabla \phi|} \right) + c |\nabla \phi| g(\nabla I)$
3. Lorsqu'on approxime la dérivé et les gradients par une version numérique, on obtient l'algorithme
4. **NOTE:** Après un certain nombre d'itérations de l'algorithme des GAC, la fonction d'élévation  $\phi(x, y)$  finie par prendre des valeurs très positives à l'intérieur de la courbe et très négative à l'extérieur de la courbe. Il est donc préférable de réinitialiser  $\phi(x, y)$  après chaque X itérations afin d'éviter tout débordement. “*narrow band implementation*”

# Les contours géodésiques actifs

Ce qu'il faut retenir de:

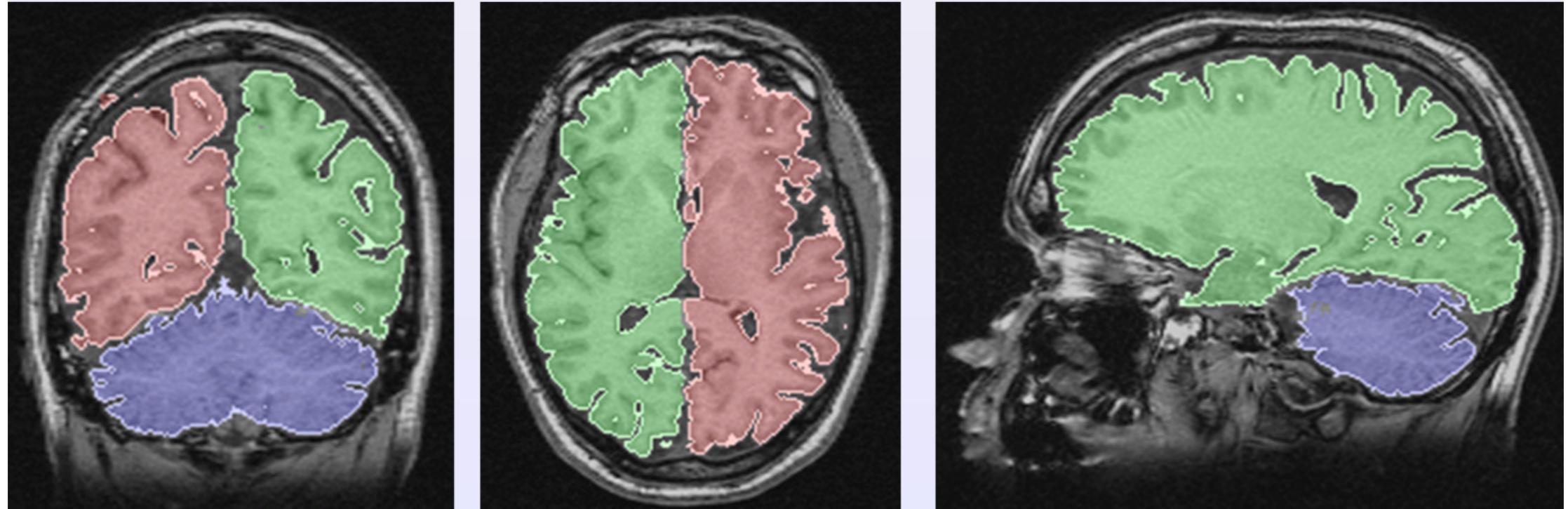
1. un contour est le niveau d'une fonction d'élévation  $\phi(x, y)$
2. On cherche à solve l'équation aux dérivées partielles :  
$$\gamma \left( g(\nabla I) \frac{\nabla \phi}{|\nabla \phi|} \right) + c |\nabla \phi| g(\nabla I) = 0$$
3. Lorsqu'on approche la solution par une version numérique, on obtient :
4. **NOTE:** Après un certain nombre d'itérations de l'algorithme des GAC, la fonction d'élévation  $\phi(x, y)$  finie par prendre des valeurs très positives à l'intérieur de la courbe et très négative à l'extérieur de la courbe. Il est donc préférable de réinitialiser  $\phi(x, y)$  après chaque X itérations afin d'éviter tout débordement. “*narrow band implementation*”



# Les contours géodésiques actifs

Pour en apprendre plus au sujet des *geodesic active contours (snakes)*:

Caselles, Kimmel, Sapiro, “**Geodesic Active Contours**”, International Journal of Computer Vision, 22(1), 61-79, 1997.





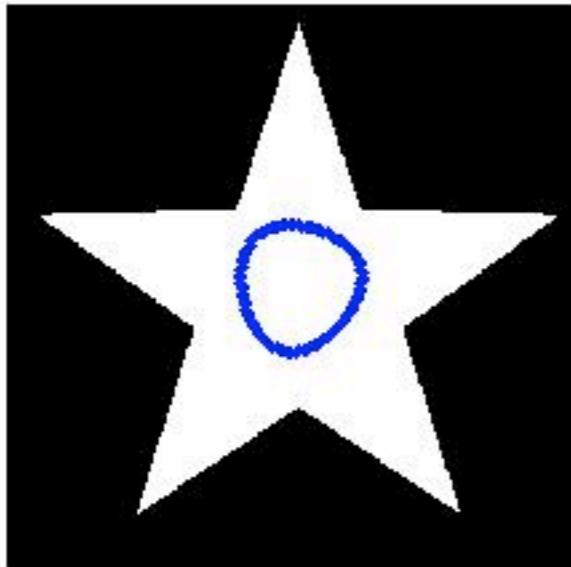




# Exemple SNAKES

- Voir démo.

The image with initial contour



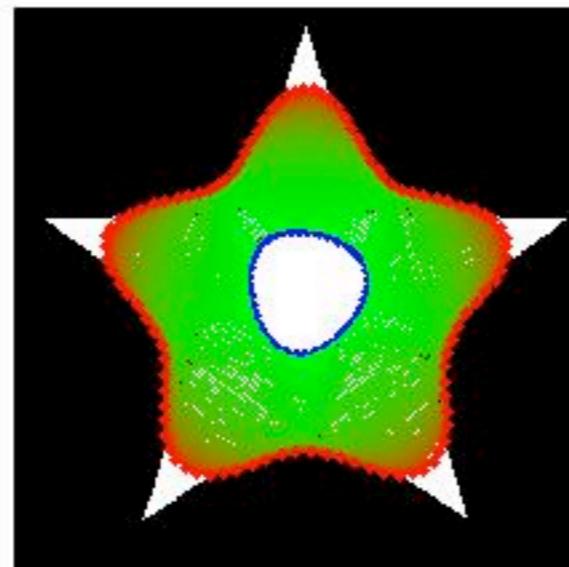
The external energy



The external force field



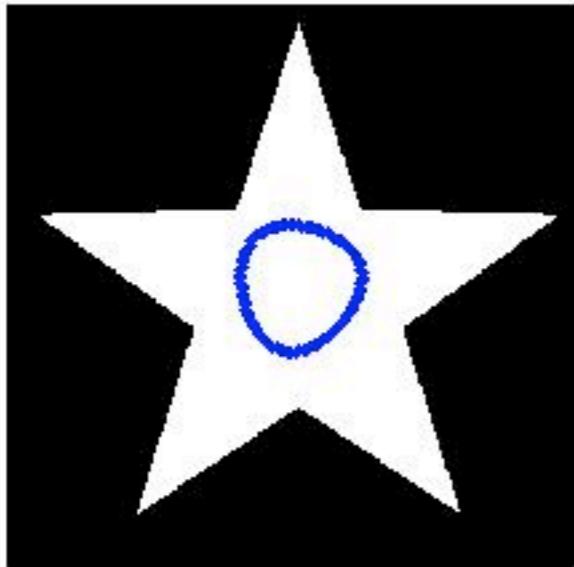
Snake movement



# Exemple SNAKES

- Voir démo.

The image with initial contour



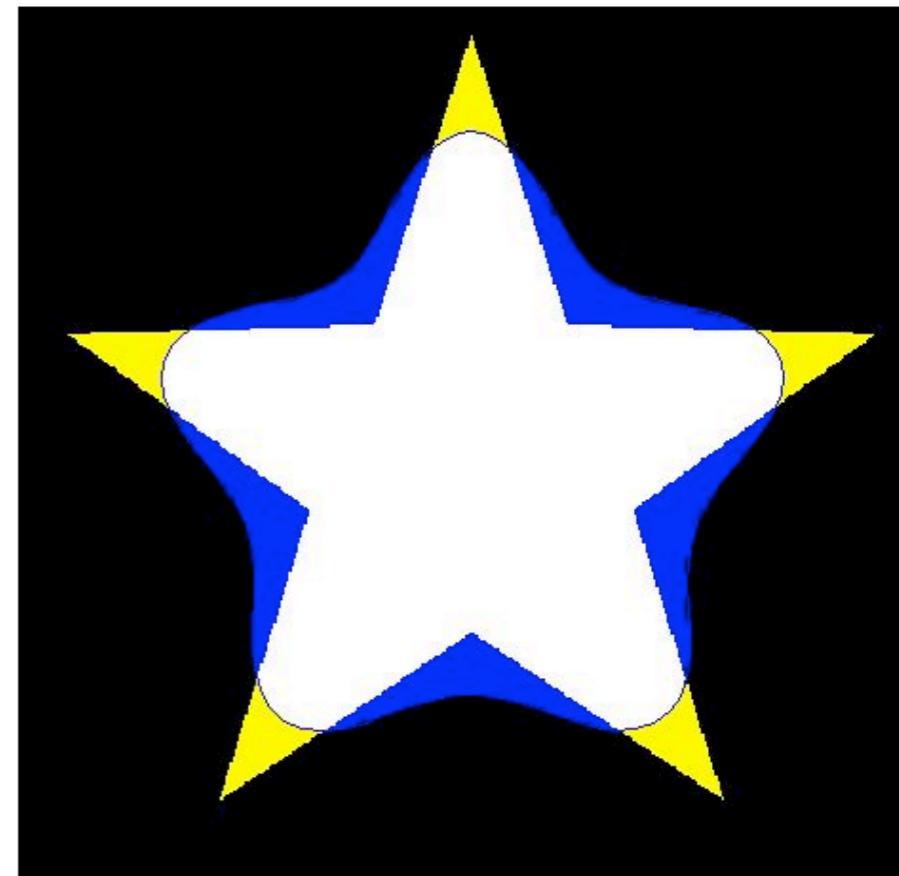
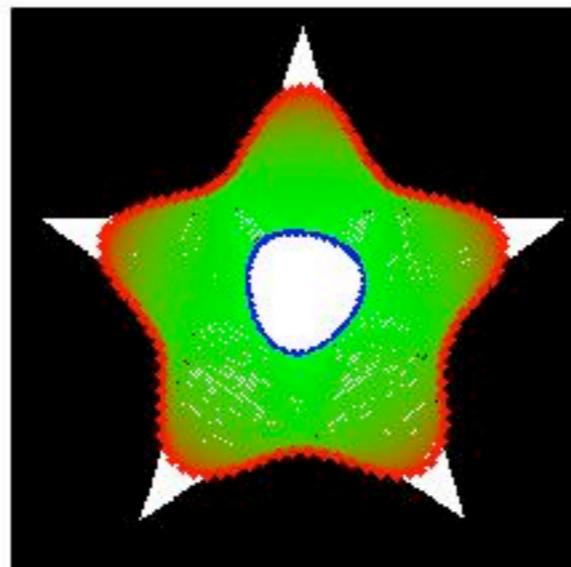
The external energy



The external force field



Snake movement



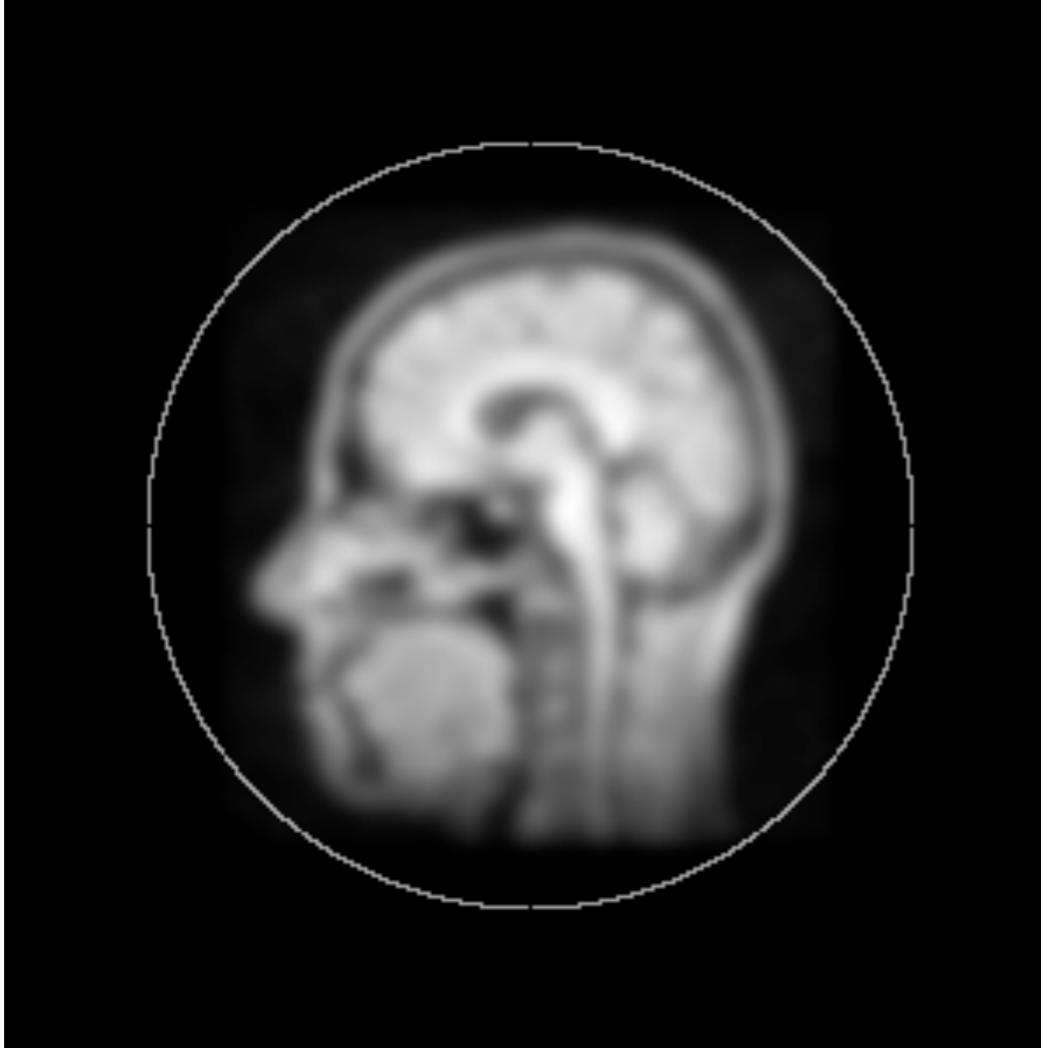
# Exemple GAC

$$\frac{\partial \phi}{\partial t} = |\nabla \phi| \operatorname{div} \left( g(\nabla I) \frac{\nabla \phi}{|\nabla \phi|} \right) + c |\nabla \phi| g(\nabla I)$$

$$g(\nabla I|_{ij}) = \frac{1}{1 + \frac{|\nabla I|_{ij}^2}{\lambda^2}}$$

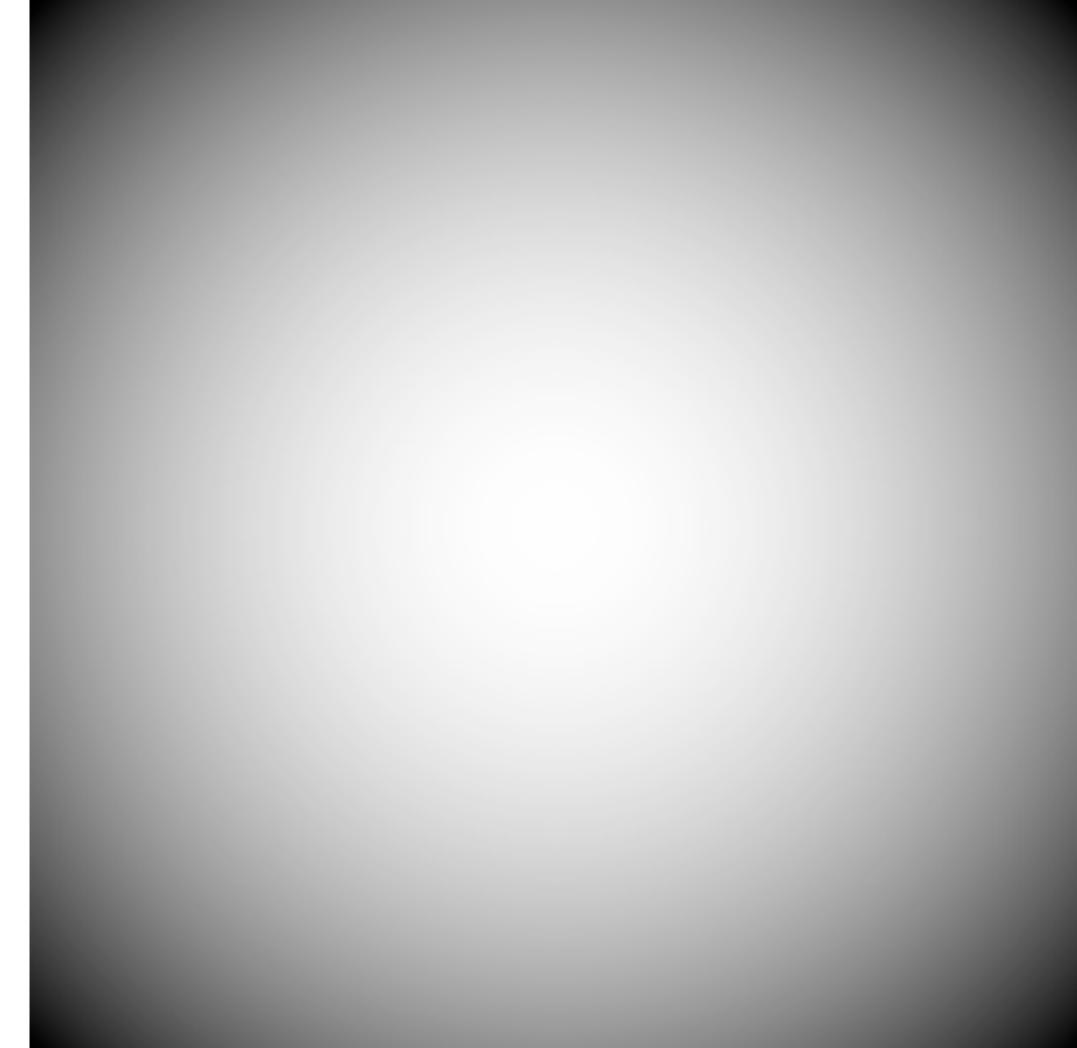
Animations

$$\sigma = 3, \lambda = 3, \tau = 0.25 \text{ et } c = -0.1$$



$$I_\sigma + C$$

$$\text{où } C = \{(x, y) \mid \phi(x, y) = 0\}$$



$$\phi(x, y)$$

Voir démo

# Local shape description

---

- Hessian matrix

$$H = \begin{bmatrix} I_{xx} & I_{xy} & I_{xz} \\ I_{yx} & I_{yy} & I_{yz} \\ I_{zx} & I_{zy} & I_{zz} \end{bmatrix}$$

- Encodes shape information, i.e., how the normal to the iso-intensity manifold changes locally

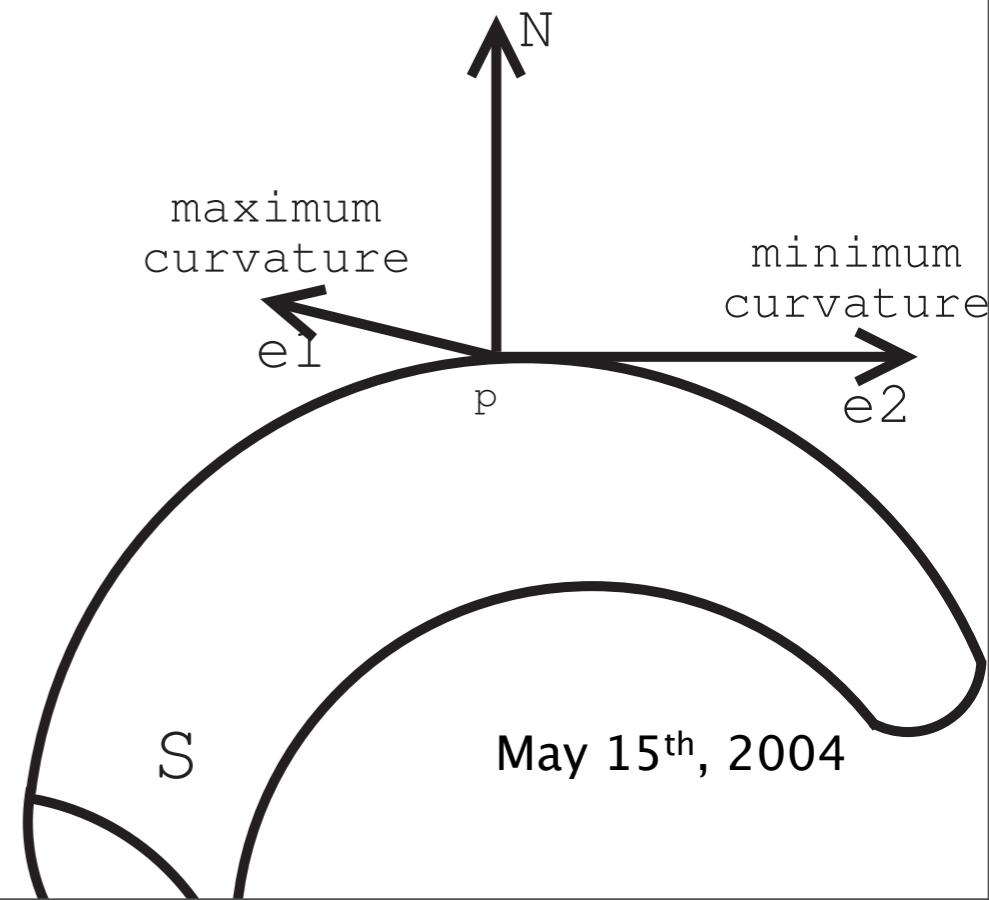
[Descoteaux et al. MEDIA 2008]

# Local shape description

- Hessian matrix

$$H = \begin{bmatrix} I_{xx} & I_{xy} & I_{xz} \\ I_{yx} & I_{yy} & I_{yz} \\ I_{zx} & I_{zy} & I_{zz} \end{bmatrix}$$

- Encodes shape information, i.e., how the normal to the iso-intensity manifold changes locally



[Descoteaux et al. MEDIA 2008]

# Local structure classification

Eigen value conditions	local structure	examples
$\lambda_1 \approx 0, \lambda_2 \approx \lambda_3 \gg 0$	tube-like	vessel, bronchus
$\lambda_1 \approx \lambda_2 \approx 0, \lambda_3 \gg 0$	sheet-like	cortex, skin
$\lambda_1 \approx \lambda_2 \approx \lambda_3 \gg 0$	blob-like	nodule
$\lambda_1 \approx \lambda_2 \approx \lambda_3 \approx 0$	noise-like	noise

$$R_B = \frac{|\lambda_1|}{\sqrt{|\lambda_2 \lambda_3|}} \quad R_A = \frac{|\lambda_2|}{|\lambda_3|} \quad S = \sqrt{\lambda_1^2 + \lambda_2^2 + \lambda_3^2}$$

**blob vs others**

**sheet vs others**

**noise vs others**

[Descoteaux et al. MEDIA 2008]

# Frangi's vesslness measure

---

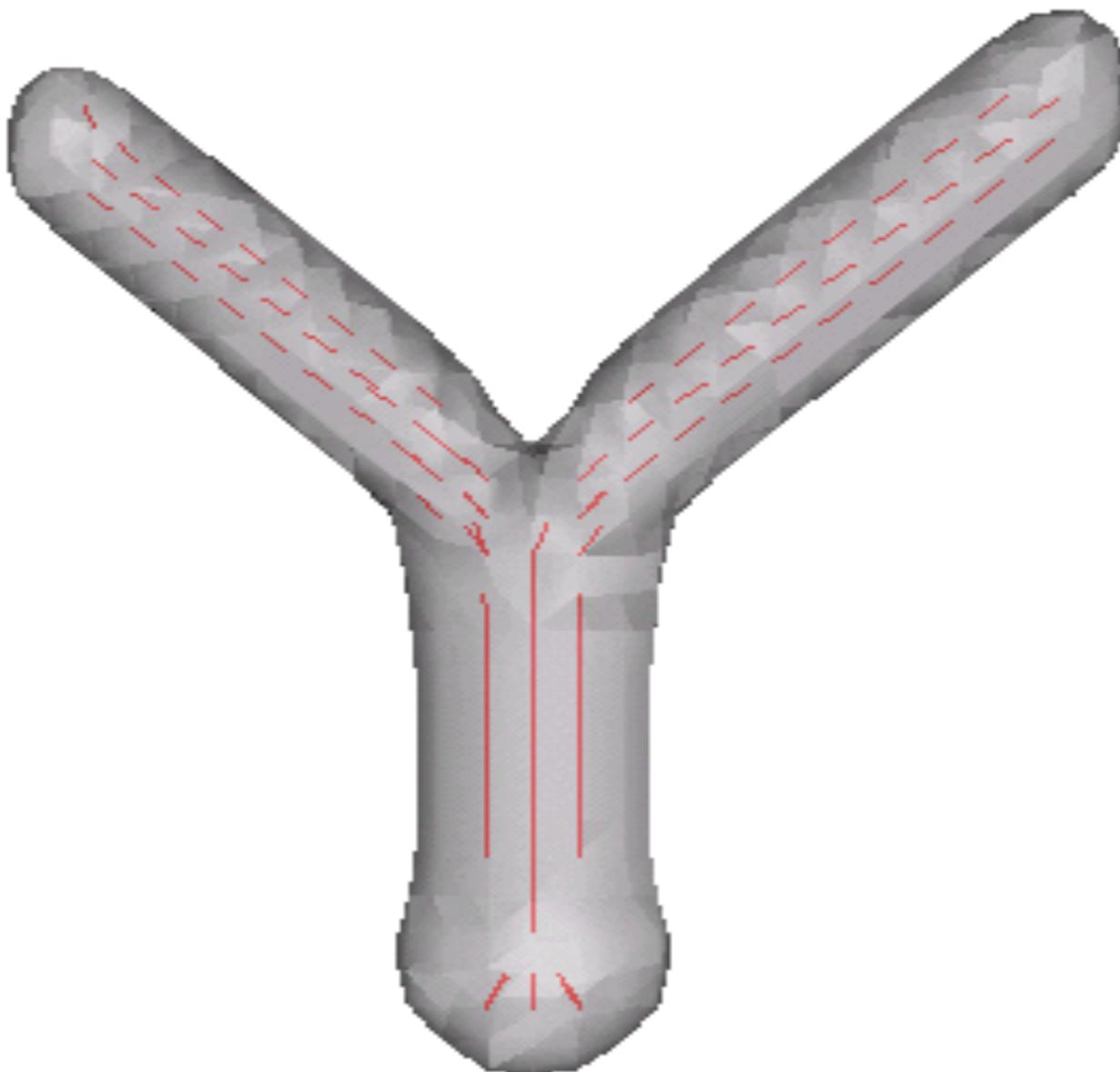
$$V(\sigma) = \begin{cases} 0 & \text{if } \lambda_2 < 0 \text{ or } \lambda_3 < 0 \\ (1 - \exp\left(-\frac{R_A^2}{2\alpha^2}\right)) \exp\left(-\frac{R_B^2}{2\beta^2}\right) (1 - \exp\left(-\frac{S^2}{2c^2}\right)) & \text{otherwise} \end{cases}$$

- Maximum along centerlines of tubular structures
- Close to zero outside vessel-like regions
- $\operatorname{argmax}( V(\sigma) ) = \text{radius of vessel}$

[Descoteaux et al. MEDIA 2008]

# Synthetic branch example

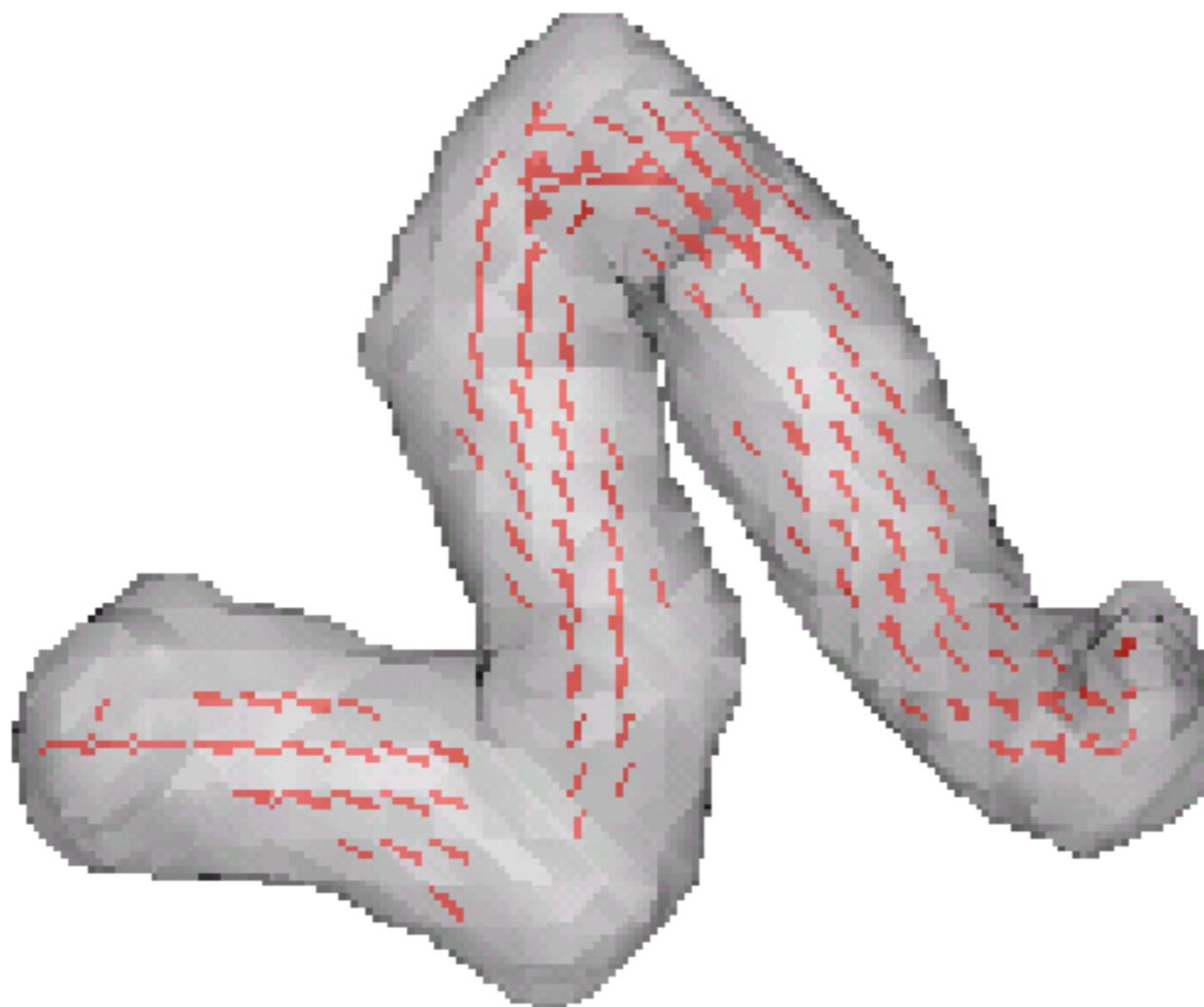
---



[Descoteaux et al. MEDIA 2008]

# Cropped MRA region

---



[Descoteaux et al. MEDIA 2008]

# Flux maximizing geometric flow

---

- Used to direct the evolution of a curve/surface so that its normals are aligned with a given vector field

$$\mathcal{S}_t = \operatorname{div}(\vec{\mathcal{V}}) \vec{\mathcal{N}}$$

[Vasilevkiy, Siddiqi, PAMI 02]

May 15<sup>th</sup>, 2004

# Multi-scale geometric flow

---

- Consider the vector field
- The associated flux maximizing flow

$$\vec{\mathcal{V}} = \phi \frac{\nabla \mathcal{I}}{|\nabla \mathcal{I}|}$$

$$\begin{aligned} S_t &= \operatorname{div}(\vec{\mathcal{V}}) \vec{\mathcal{N}} \\ &= \left[ \left\langle \nabla \phi, \frac{\nabla \mathcal{I}}{|\nabla \mathcal{I}|} \right\rangle + \phi \operatorname{div} \left( \frac{\nabla \mathcal{I}}{|\nabla \mathcal{I}|} \right) \right] \vec{\mathcal{N}} \\ &= \left[ \left\langle \nabla \phi, \frac{\nabla \mathcal{I}}{|\nabla \mathcal{I}|} \right\rangle + \phi \kappa_{\mathcal{I}} \right] \vec{\mathcal{N}}. \end{aligned}$$

[Descoteaux et al. MEDIA 2008]

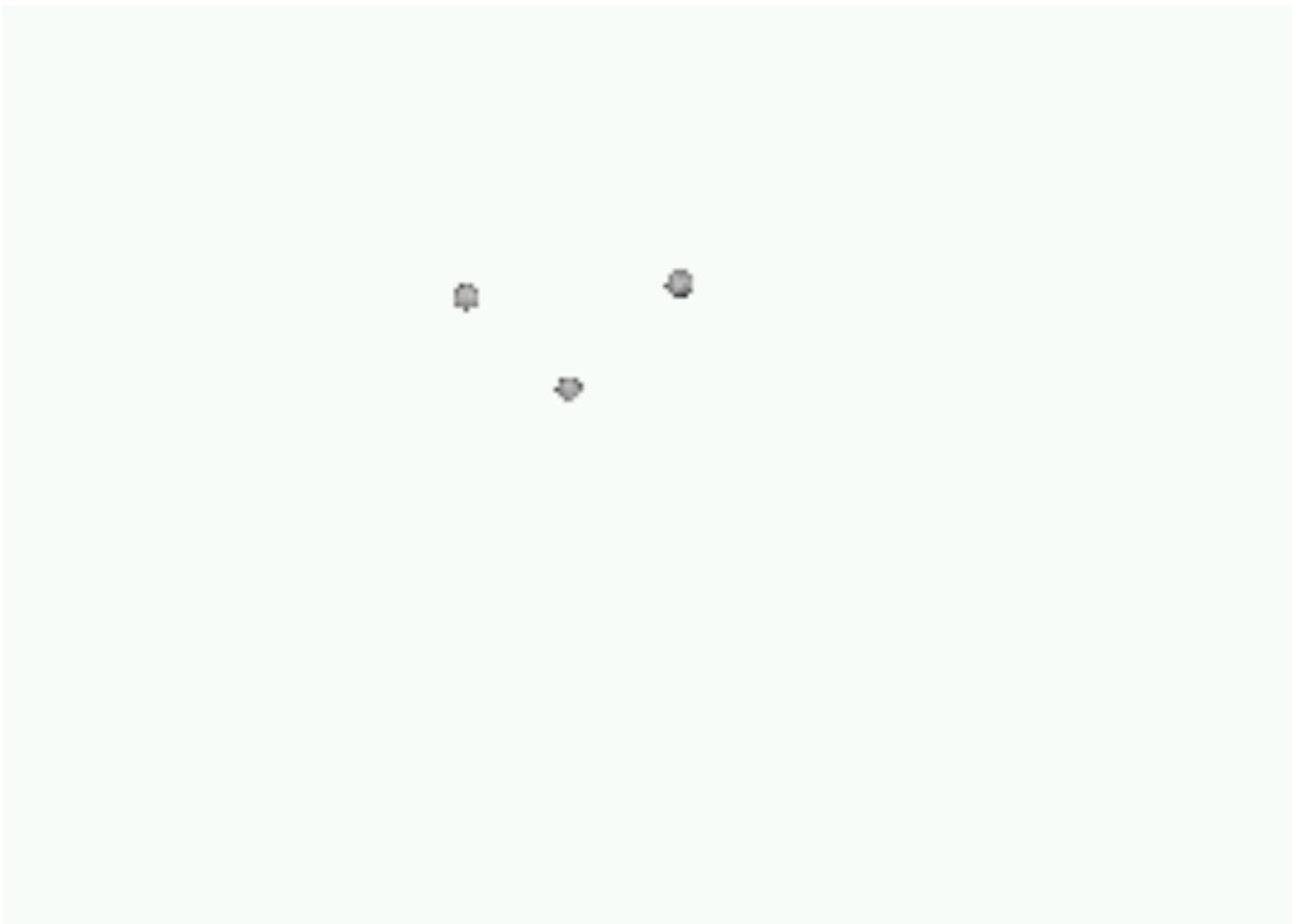
# Surface evolution

---

[Descoteaux et al. MEDIA 2008]

# Surface evolution

---

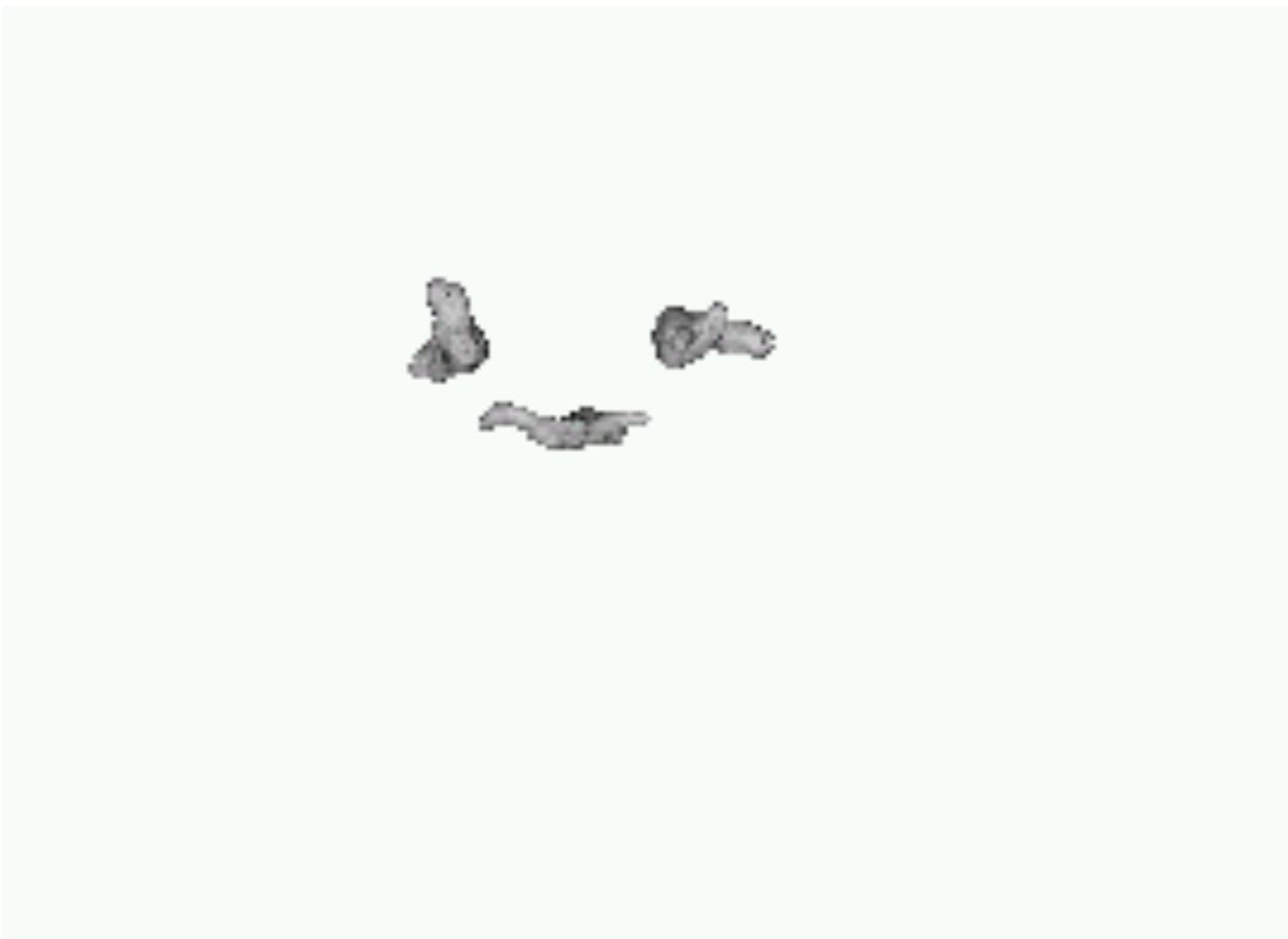


$T = 0$

[Descoteaux et al. MEDIA 2008]

# Surface evolution

---



**T = 100**

[Descoteaux et al. MEDIA 2008]

# Surface evolution

---

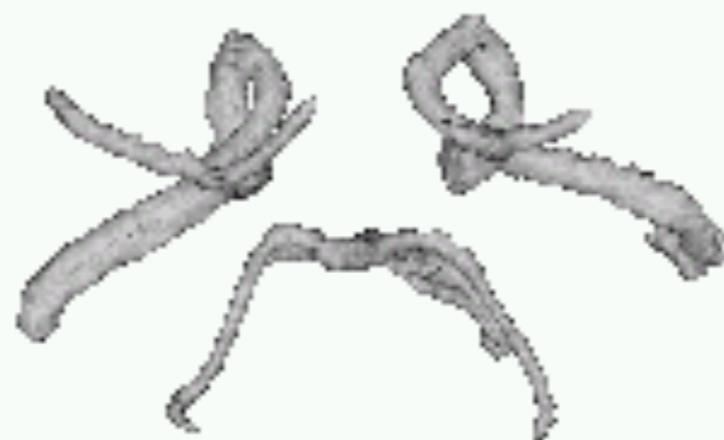


**T = 200**

[Descoteaux et al. MEDIA 2008]

# Surface evolution

---

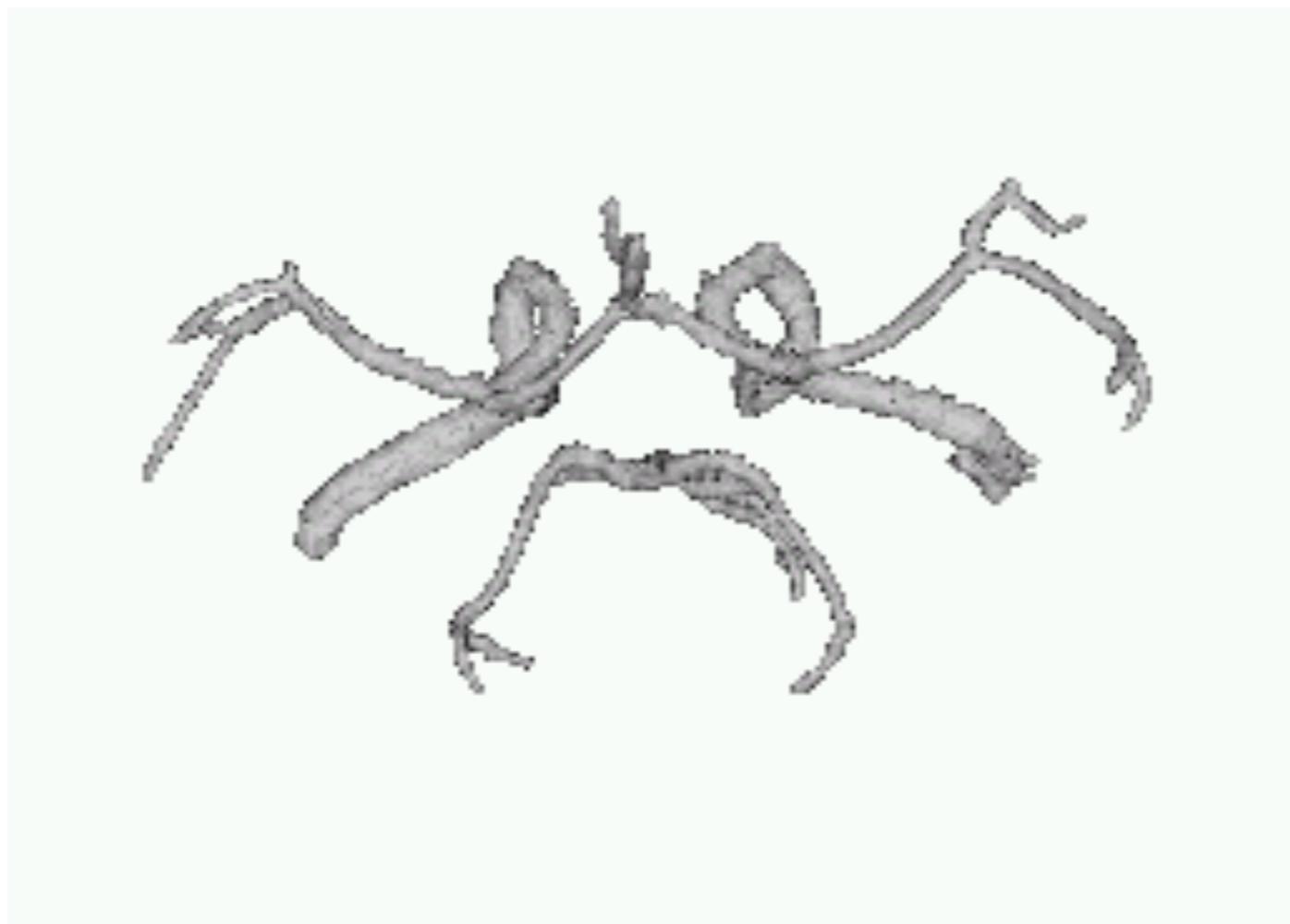


**T = 500**

[Descoteaux et al. MEDIA 2008]

# Surface evolution

---

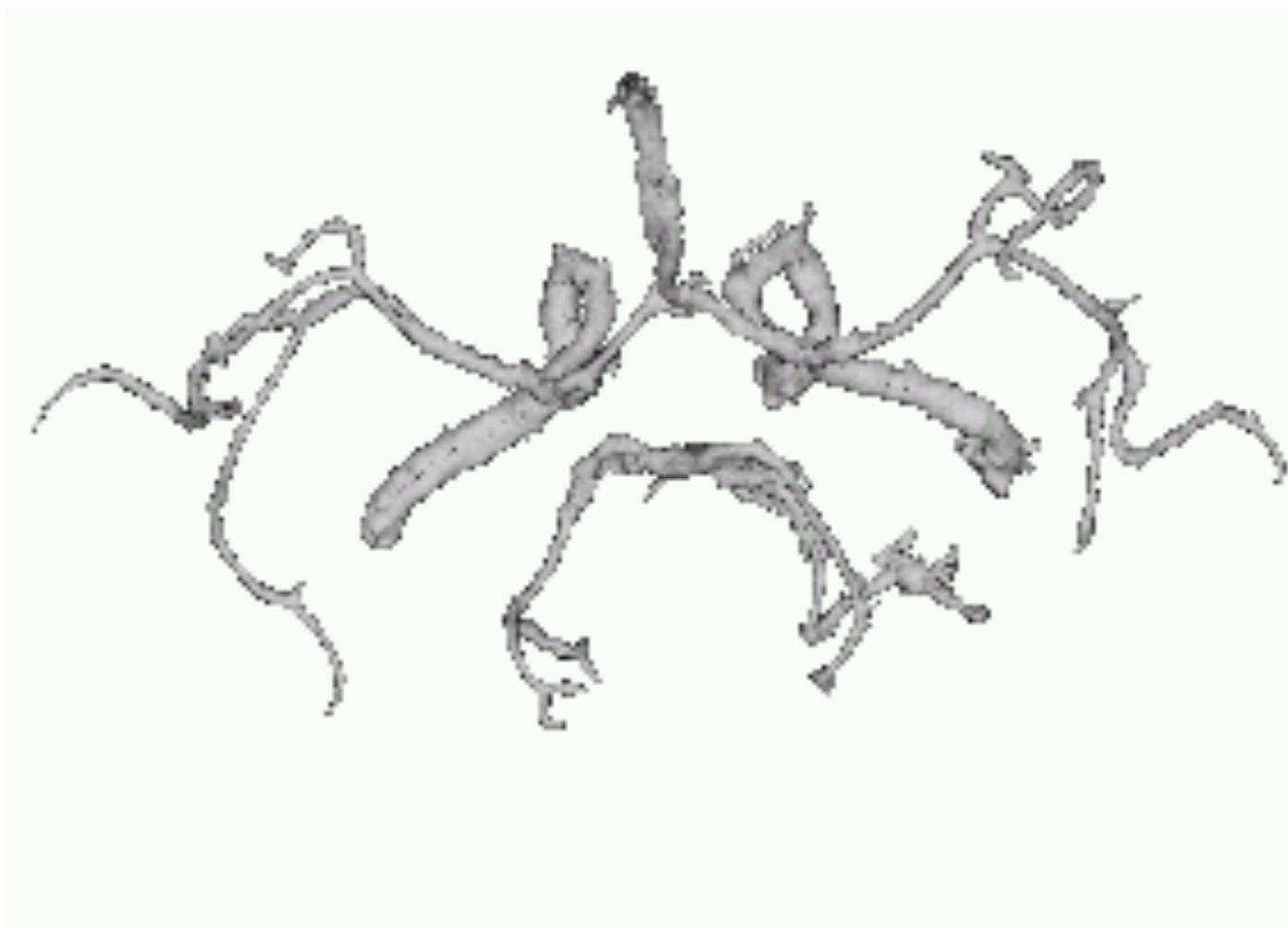


**T = 1000**

[Descoteaux et al. MEDIA 2008]

# Surface evolution

---

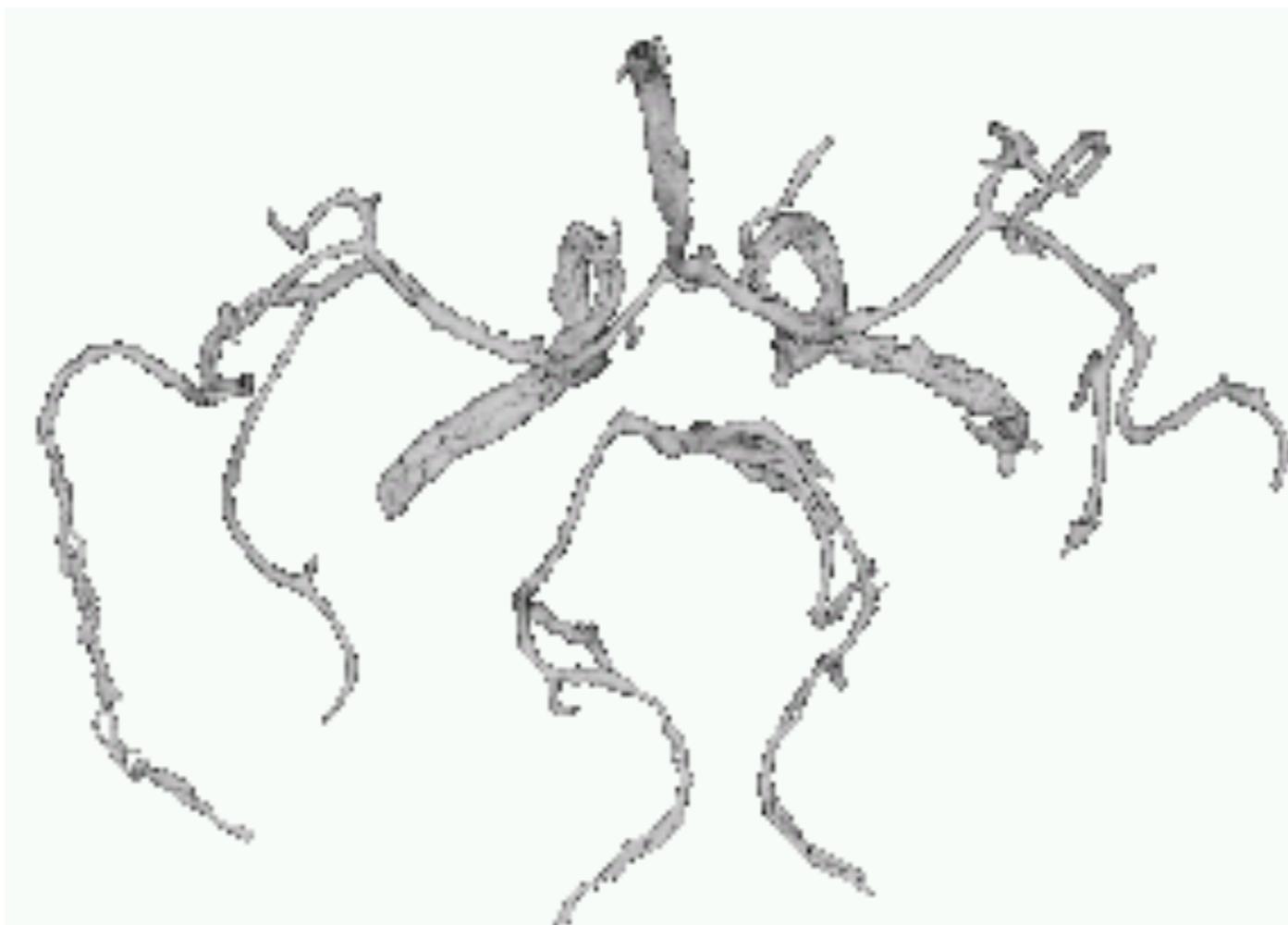


**T = 3000**

[Descoteaux et al. MEDIA 2008]

# Surface evolution

---



**T = 5000**

[Descoteaux et al. MEDIA 2008]

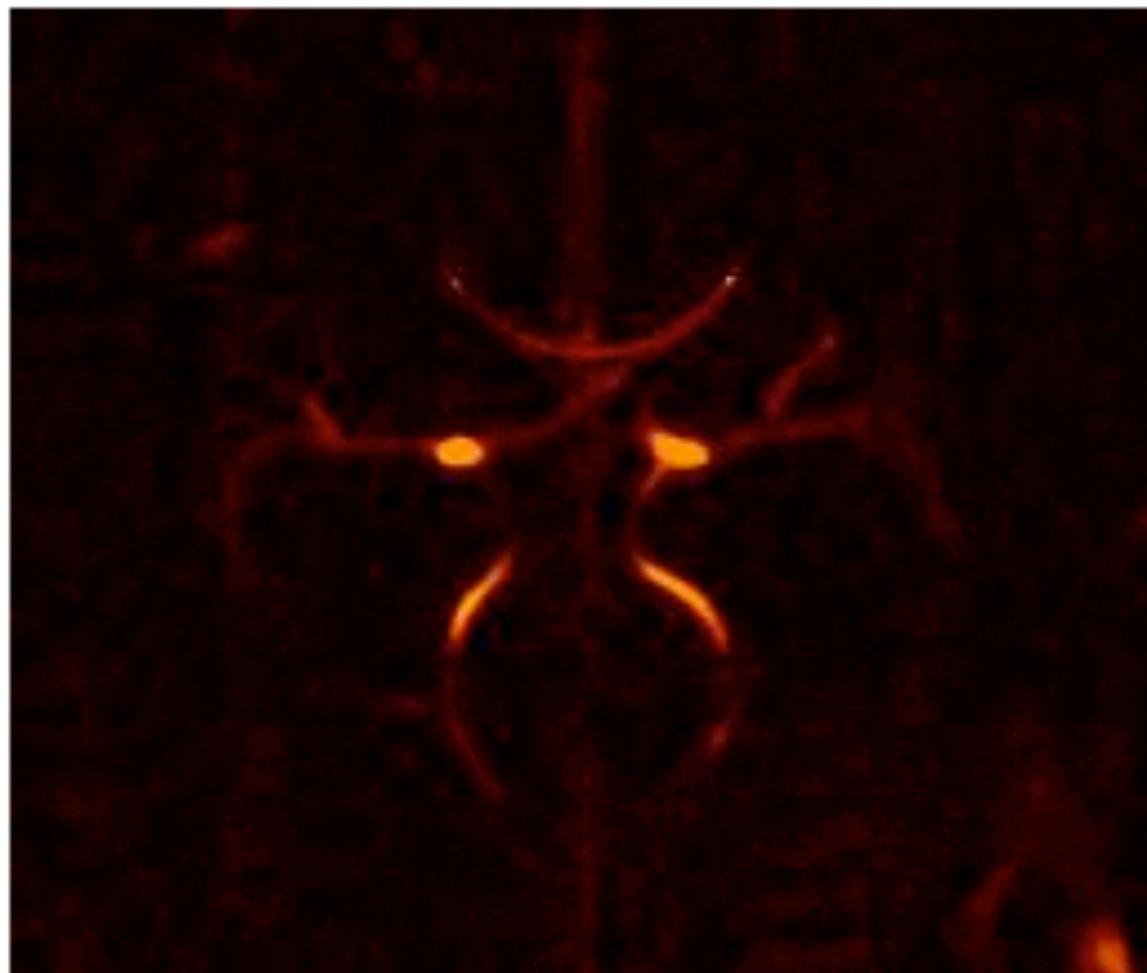
# Surface evolution

---

[Descoteaux et al. MEDIA 2008]

# Surface evolution

---



[Descoteaux et al. MEDIA 2008]

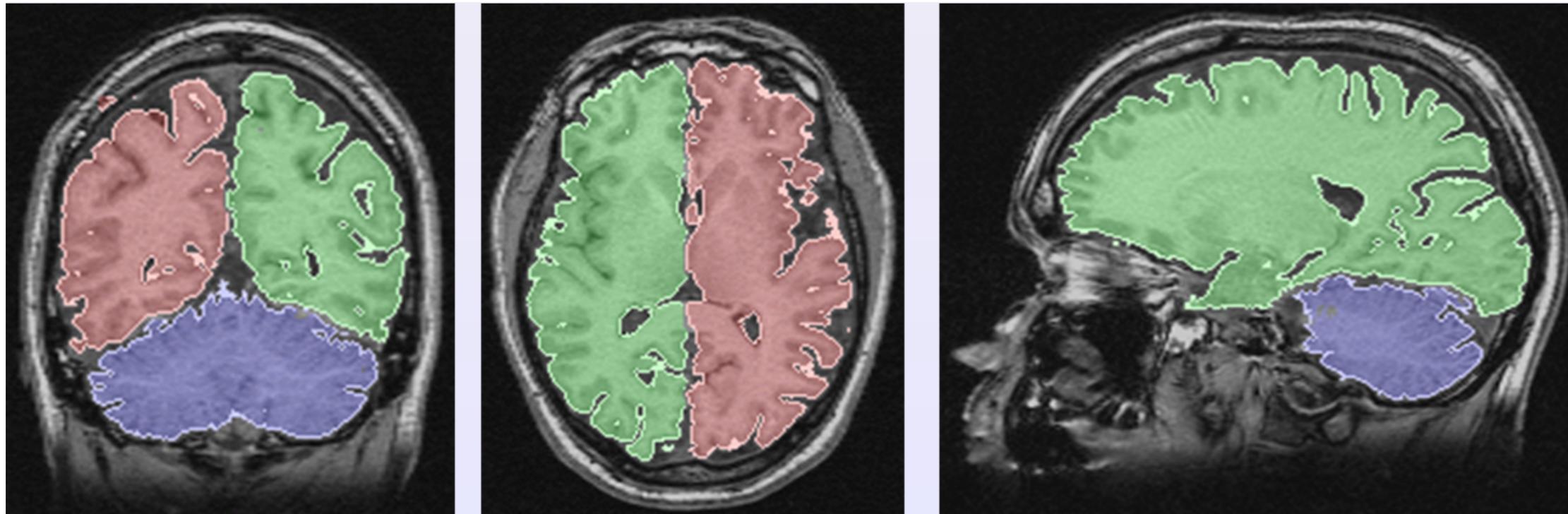
# Résultat GAC

Avantage :

- segmentation locale et interactive
- robuste aux changements de topologies (la courbe peut se scinder en 2, 3, ...)
- flexible. Peut injecter des *a priori* de formes et d'évolution (balloon term)
- existe des milliers de variantes

Inconvénients:

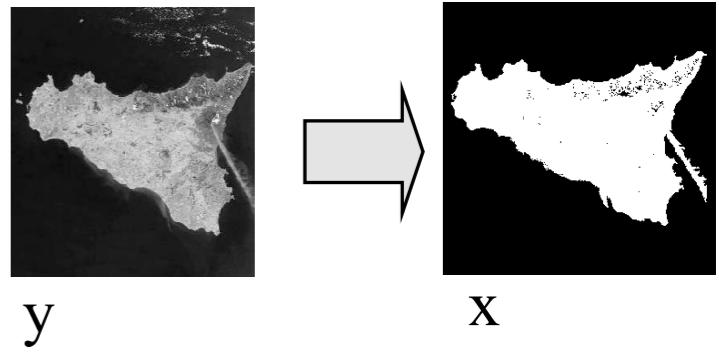
- pas évident de savoir quand arrêter l'évolution de la courbe.
- convergence!!!
- lourd en temps de calcul



# Segmentation

Méthode probabiliste avec *a priori*

# Méthodes markoviennes



- Modélisation stochastique des images : sur l'entrée  $y$  et sur le résultat  $x$  on définit une grille discrète de pixel et un système de voisinage associé
- Modélisation pour un problème donné
  - Loi *a priori* sur le résultat  $x$  :  $P(x)$
  - Vraisemblance/transition  $P(y | x)$  : modélise le processus de formation de l'image

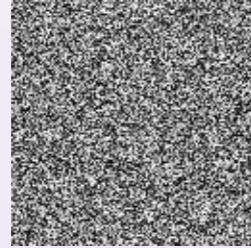
# *A priori* $P(x)$

- Notre chance de mettre nos connaissances *a priori* sur ce que l'on cherche
  - Tube, cylindre, forme d'oeuf, résultat lisse, régulier, ...

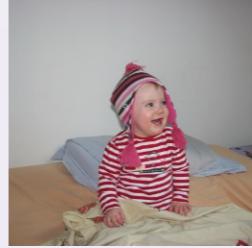
# *A priori* $P(x)$

- Notre chance de mettre nos connaissances *a priori* sur ce que l'on cherche
  - Tube, cylindre, forme d'oeuf, résultat lisse, régulier, ...

en général



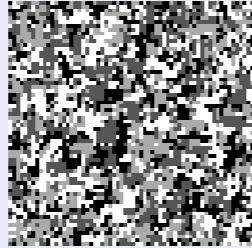
moins probable que



(si images “naturelles”)

débruitage : image régulière plus probable que du bruit

segmentation :

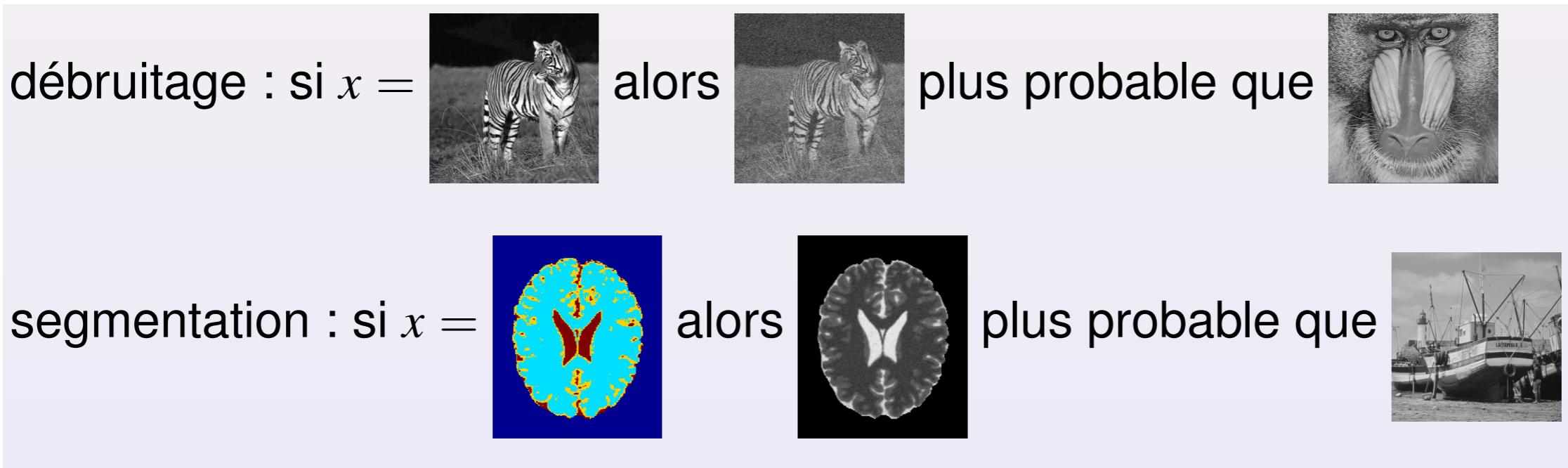


moins probable que



# Vraisemblance $P(y|x)$

- Modélise le processus de formation de l'images



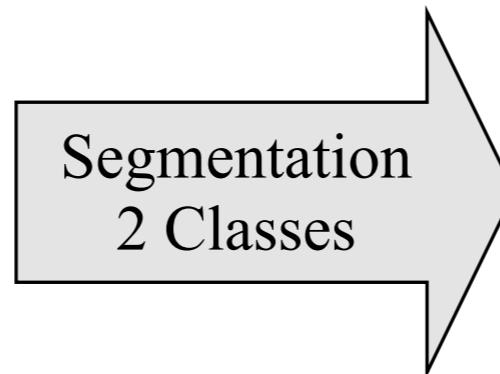
- Hypothèse de base:  $P(y|x)$  et  $P(x)$  plus facile à modéliser que  $P(x|y)$

# Maximum *a posteriori*

Théorème de Bayes + Markov : résumé



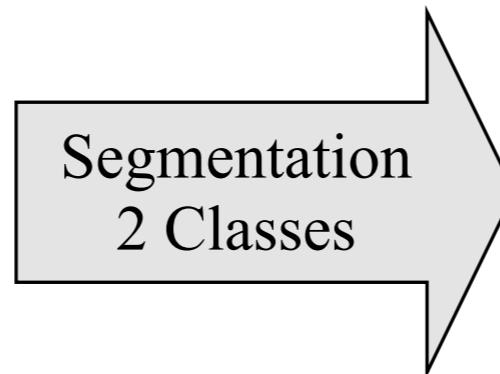
y



x

# Maximum *a posteriori*

Théorème de Bayes + Markov : résumé



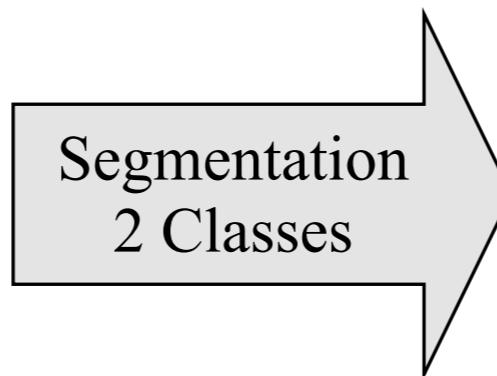
MAP:  $\hat{x} = \arg \max_x P(x | y)$

# Maximum *a posteriori*

Théorème de Bayes + Markov : résumé



y



x

MAP:  $\hat{x} = \arg \max_x P(x | y)$

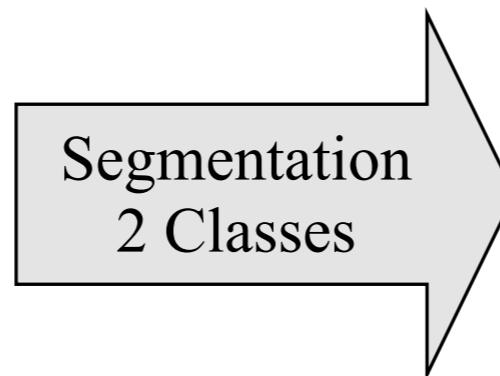
Bayes:  $\hat{x} = \arg \max_x P(y | x)P(x)$

# Maximum *a posteriori*

Théorème de Bayes + Markov : résumé



y



x

MAP:  $\hat{x} = \arg \max_x P(x | y)$

Bayes:  $\hat{x} = \arg \max_x P(y | x)P(x)$

Markov:  $\hat{x} = \arg \min_x \sum_{s \in S} U(x_s, y_s) + W(x_s)$

# Maximum a posteriori

$$U(\text{terre}, y_s) = -\ln(N(y_s; \mu_{\text{terre}}, \sigma_{\text{terre}}))$$
$$U(\text{mer}, y_s) = -\ln(N(y_s; \mu_{\text{mer}}, \sigma_{\text{mer}}))$$

Gaussienne

$$W(x_s) = \beta \sum_{t \in \eta_s} (1 - \delta(x_s, x_t)) \quad \text{où } \delta(x_s, x_t) = \begin{cases} 1 & \text{si } x_s = x_t \\ 0 & \text{sinon} \end{cases}$$

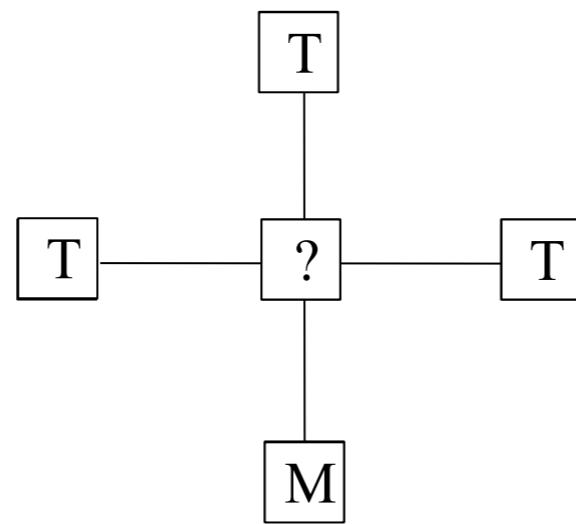
Modèle de *Ising*

# Maximum a posteriori

Exemple : fonction *a priori*, modèle de *Ising*

$$W(x_s) = \beta \sum_{t \in \mathcal{N}_s} (1 - \delta(x_s, x_t)) \quad \text{où } \delta(x_s, x_t) = \begin{cases} 1 & \text{si } x_s = x_t \\ 0 & \text{sinon} \end{cases}$$

Voisinage d'ordre 1  
(4 voisins)



$$W(T) = \beta$$

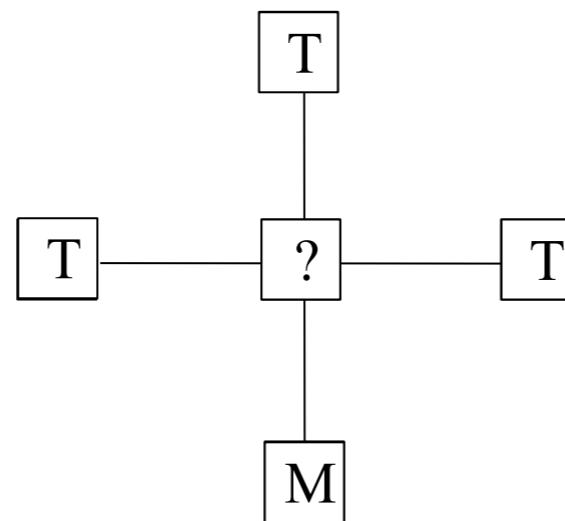
$$W(M) = 3\beta$$

# Maximum a posteriori

Exemple : fonction *a priori*, modèle de *Ising*

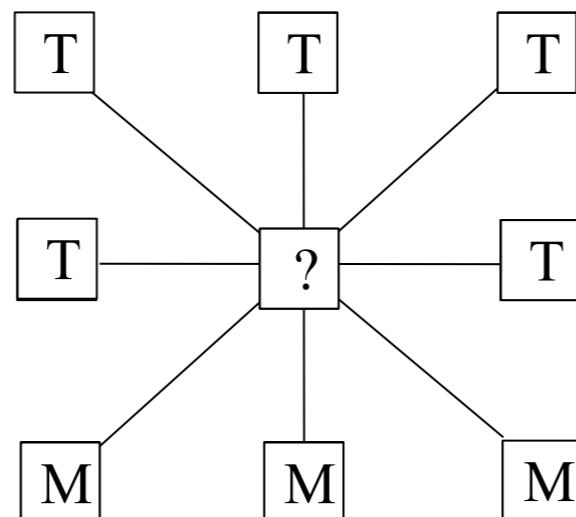
$$W(x_s) = \beta \sum_{t \in \mathcal{N}_s} (1 - \delta(x_s, x_t)) \quad \text{où } \delta(x_s, x_t) = \begin{cases} 1 & \text{si } x_s = x_t \\ 0 & \text{sinon} \end{cases}$$

Voisinage d'ordre 1  
(4 voisins)



$$W(T) = \beta$$
$$W(M) = 3\beta$$

Voisinage d'ordre 2  
(8 voisins)

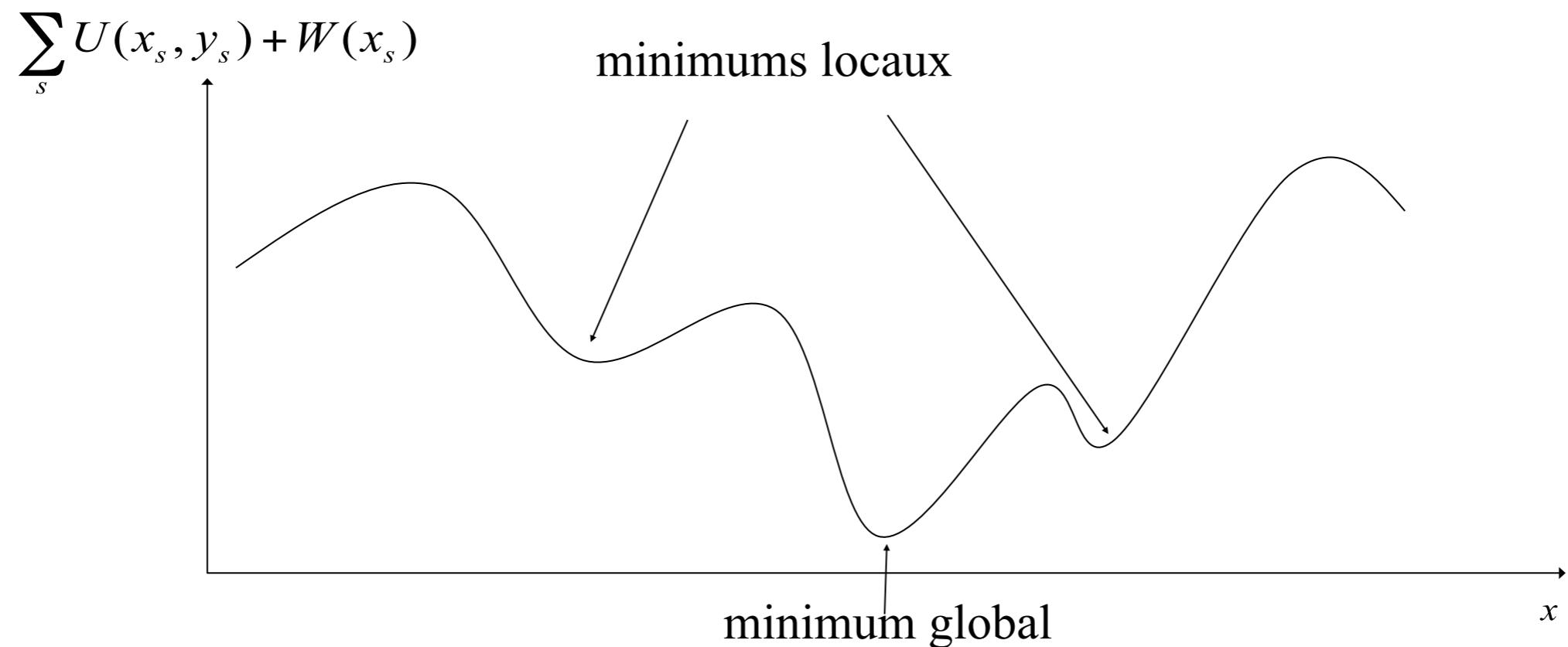


$$W(T) = 3\beta$$
$$W(M) = 5\beta$$

$$\hat{x} = \arg \min_x \sum_{s \in S} U(x_s, y_s) + W(x_s)$$

# Maximum a posteriori

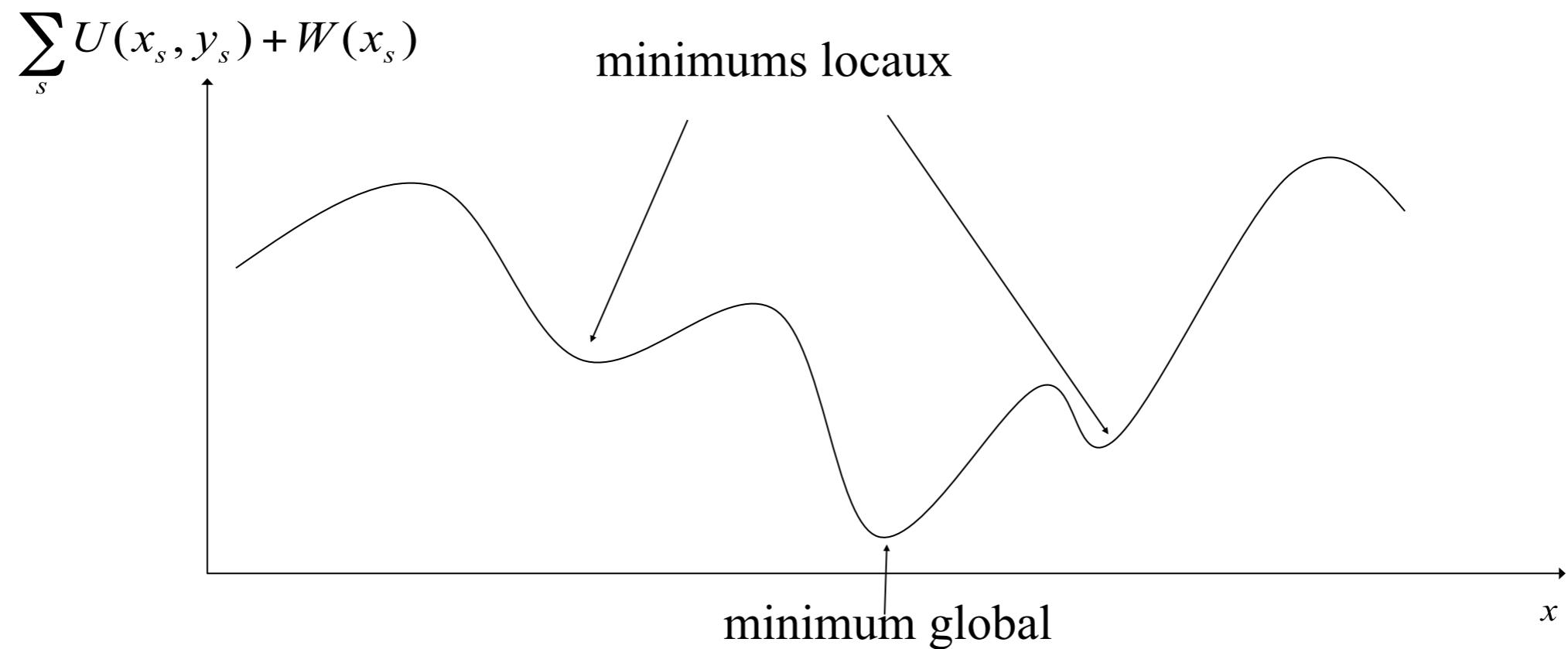
Trouver le « meilleur » champ d'étiquettes  $x$  au sens du MAP est un problème classique de minimisation de fonction d'énergie.



$$\hat{x} = \arg \min_x \sum_{s \in S} U(x_s, y_s) + W(x_s)$$

# Maximum a posteriori

Trouver le « meilleur » champ d'étiquettes  $x$  au sens du MAP est un problème classique de minimisation de fonction d'énergie.



**Algorithmes déterministes** : rapides mais convergent vers le minimum le plus près.  
 Ces algorithmes doivent donc être initialisés près du minimum global (ex. **ICM**, **HFC**).  
**Algorithmes stochastiques** : très lents mais convergent « *toujours* » vers le minimum global, peu importe le point de départ. (ex. **recuit simulé**).

# Maximum a posteriori

## Algorithme ICM (déterministe)

/\* Initialiser le champ d'étiquettes et les paramètres des gaussiennes (ex.: *K-Means, EM*)\*/

$$\{x, \mu_{mer}, \sigma_{mer}, \mu_{terre}, \sigma_{terre}\} \leftarrow \{\hat{x}, \hat{\mu}_{mer}, \hat{\sigma}_{mer}, \hat{\mu}_{terre}, \hat{\sigma}_{terre}\}$$

FAIRE

$$x_{prev} \leftarrow x$$

POUR CHAQUE site  $s$  de  $x$  FAIRE

$$U_{terre} = U(terre, y_s) + W(terre)$$

$$U_{mer} = U(mer, y_s) + W(mer)$$

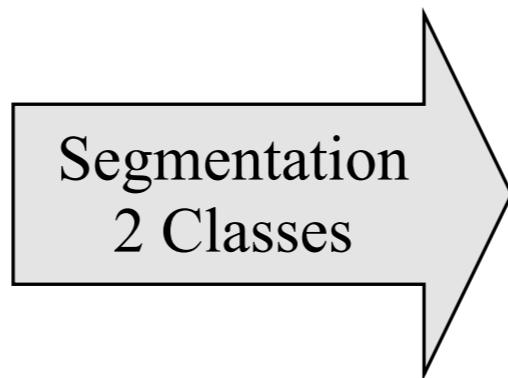
SI  $U_{terre} < U_{mer}$  ALORS

$x_s = 1$  /\* Étiquette « *terre* » au pixel (i,j) \*/  
SINON

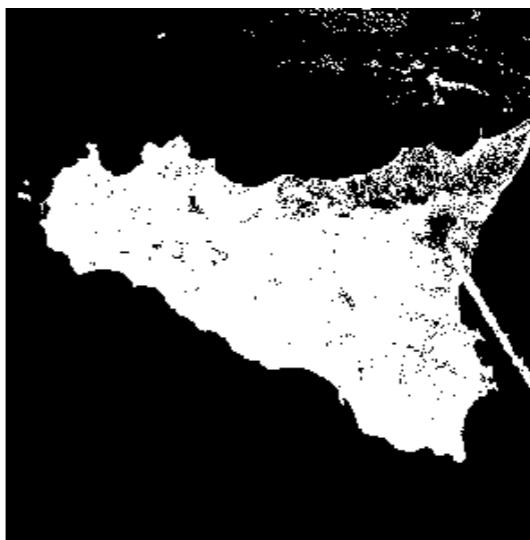
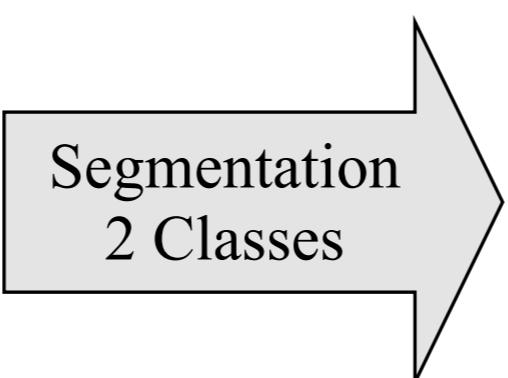
$x_s = 0$  /\* Étiquette « *mer* » au pixel (i,j) \*/

TANT QUE  $x_{prev} \neq x$

# Résultats



ICM



K-Moyennes

x Terre/Mer

# Maximum a posteriori

## Algorithme du recuit simulé

$\{\mu_{mer}, \sigma_{mer}, \mu_{terre}, \sigma_{terre}\} \leftarrow \{\hat{x}, \hat{\mu}_{mer}, \hat{\sigma}_{mer}, \hat{\mu}_{terre}, \hat{\sigma}_{terre}\}$  /\* *K – Means* \*/

$x \leftarrow random$

$T = T_{\max}$

FAIRE

POUR CHAQUE site  $s$  de  $x$  FAIRE

$$P_{terre} = e^{\frac{-(U(terre, y_s) + W(terre))}{T}}$$

$$P_{mer} = e^{\frac{-(U(mer, y_s) + W(mer))}{T}}$$

Normaliser  $P_{terre}$  et  $P_{mer}$  afin que  $P_{terre} + P_{mer} = 1$

$p$  = valeur aléatoire entre 0 et 1

SI  $p < P_{terre}$  ALORS

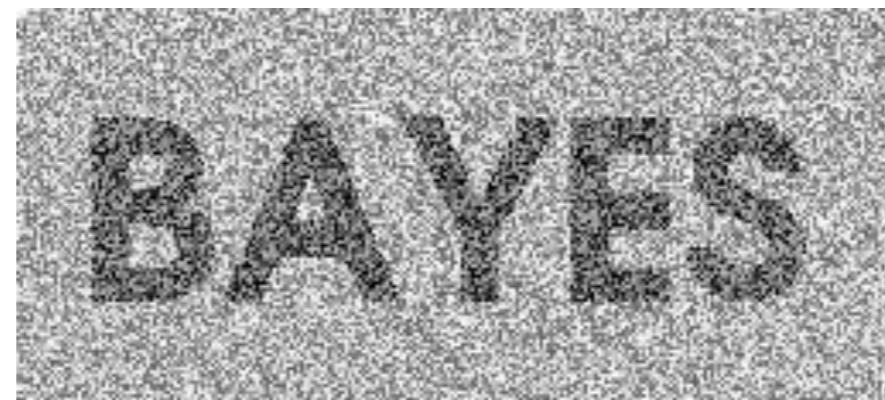
$x_s = 1$  /\* Étiquette « *terre* » au pixel (i,j) \*/  
SINON

$x_s = 0$  /\* Étiquette « *mer* » au pixel (i,j) \*/

$T = T \times \text{Taux\_de\_refroidissement}$

TANT QUE  $T > T_{\min}$

# Maximum a posteriori



# Maximum a posteriori

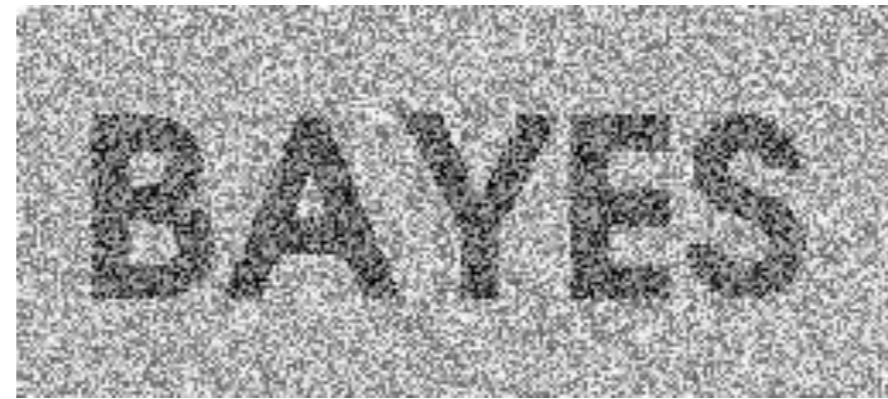


ICM



8 itérations  
(énergie globale = 384 232)

# Maximum a posteriori



ICM



8 itérations  
(énergie globale = 384 232)

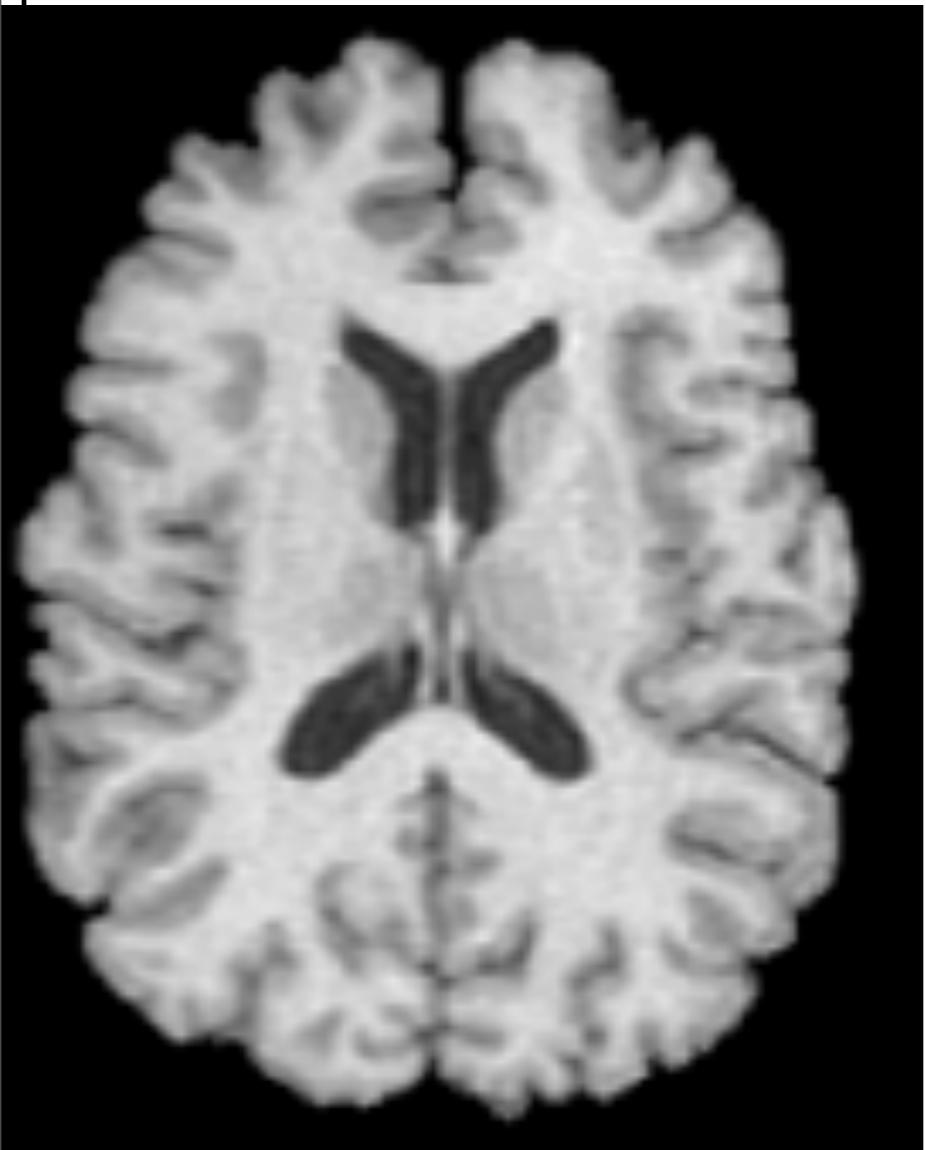
Recuit simulé



248 itérations  
(énergie globale = 383 782)

# K-means versus Markov

Original



k-means



Markov



# Maximum a posteriori

Calcul de l'énergie globale d'une paire (x,y)

$$U_{tot} = 0$$

POUR CHAQUE site  $s$  de  $x$  FAIRE

$$U_{tot} += U(x_s, y_s) + W(x_s)$$

Note: avec ICM l'énergie globale doit **TOUJOURS-TOUJOURS-TOUJOURS** diminuer d'une itération à l'autre, jusqu'à convergence

# Méthodes markoviennes

- Méthodes ont l'avantage d'être assez souples: on modifie l'énergie, on initialise, on optimise  
(recuit simulé, ICM, descente de gradient)

# Méthodes markoviennes

- Méthodes ont l'avantage d'être assez souples: on modifie l'énergie, on initialise, on optimise  
(recuit simulé, ICM, descente de gradient)
- Par rapport aux approches continues/variationnelles :  
modélisation du bruit, ajout d'*a priori*

# Méthodes markoviennes

- Méthodes ont l'avantage d'être assez souples: on modifie l'énergie, on initialise, on optimise  
(recuit simulé, ICM, descente de gradient)
- Par rapport aux approches continues/variationnelles : modélisation du bruit, ajout d'*a priori*
- Très à la vogue dans les années 85-95 : elles ont été moins utilisées lorsque les images sont devenues trop grandes

# Méthodes markoviennes

- Méthodes ont l'avantage d'être assez souples: on modifie l'énergie, on initialise, on optimise  
(recuit simulé, ICM, descente de gradient)
- Par rapport aux approches continues/variationnelles : modélisation du bruit, ajout d'*a priori*
- Très à la vogue dans les années 85-95 : elles ont été moins utilisées lorsque les images sont devenues trop grandes
  - Marche assez bien pour l'ultrason

# Méthodes markoviennes

- Méthodes ont l'avantage d'être assez souples: on modifie l'énergie, on initialise, on optimise  
(recuit simulé, ICM, descente de gradient)
- Par rapport aux approches continues/variationnelles : modélisation du bruit, ajout d'*a priori*
- Très à la vogue dans les années 85-95 : elles ont été moins utilisées lorsque les images sont devenues trop grandes
  - Marche assez bien pour l'ultrason
- Regain d'intérêt depuis 5-10 ans grâce à l'apparition des techniques comme GraphCut qui permettent une optimisation rapide

# GraphCut

- Voir notes extras  
(si temps, retour dans les sujets avancés à la fin du cours)
- Voir chapitre 8 (Toennies Med IA 2012)
- Optimum global
- Plus rapide
- Reste quand même lourd sur une image 3D médicale

# Sursegmentation

## Partage des eaux (*Watersheds*)

# Partage des eaux

Utile pour effectuer une **sursegmentation** et ainsi retrouver des **super-pixels**.

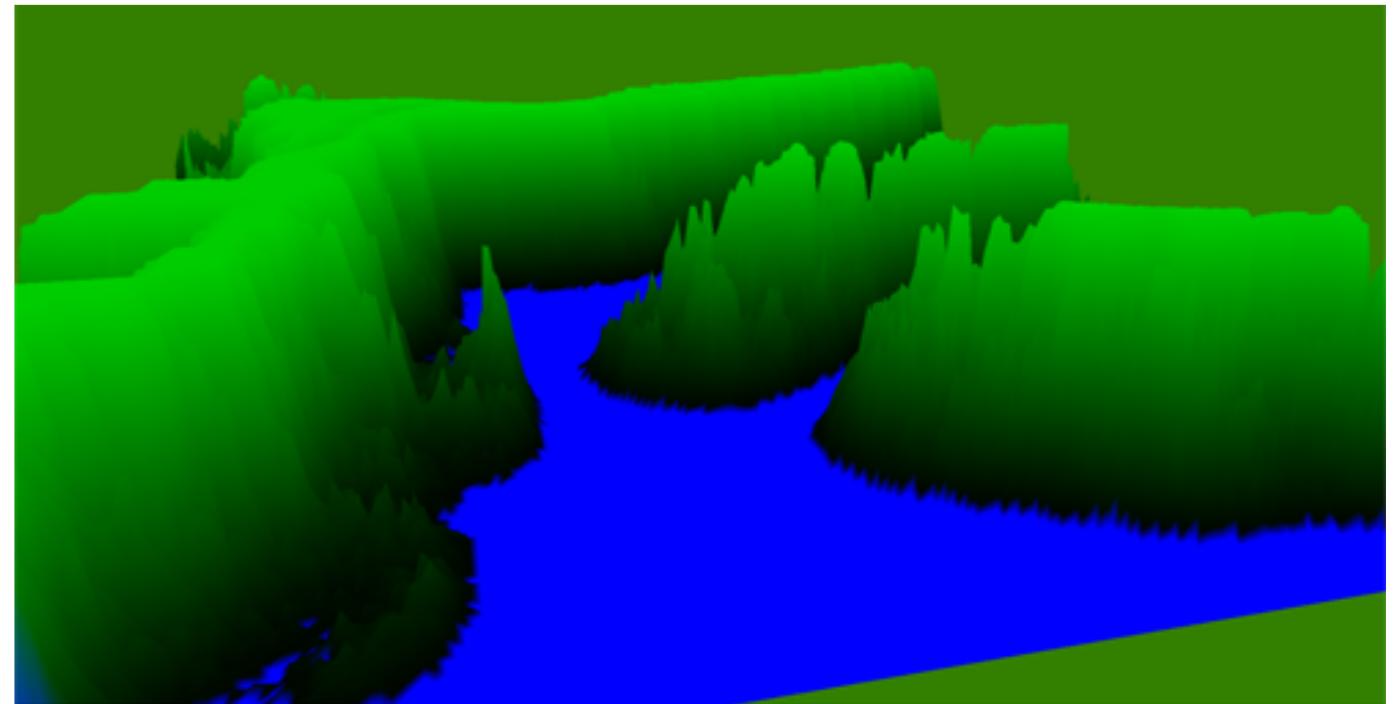
Pour ce faire, voir une image comme une **carte topologique**

L'algorithme consiste à “**inonder**” l'image en sélectionnant les pixels les plus sombres en premier.  
Le but étant de trouver **les lignes de partage des eaux**.

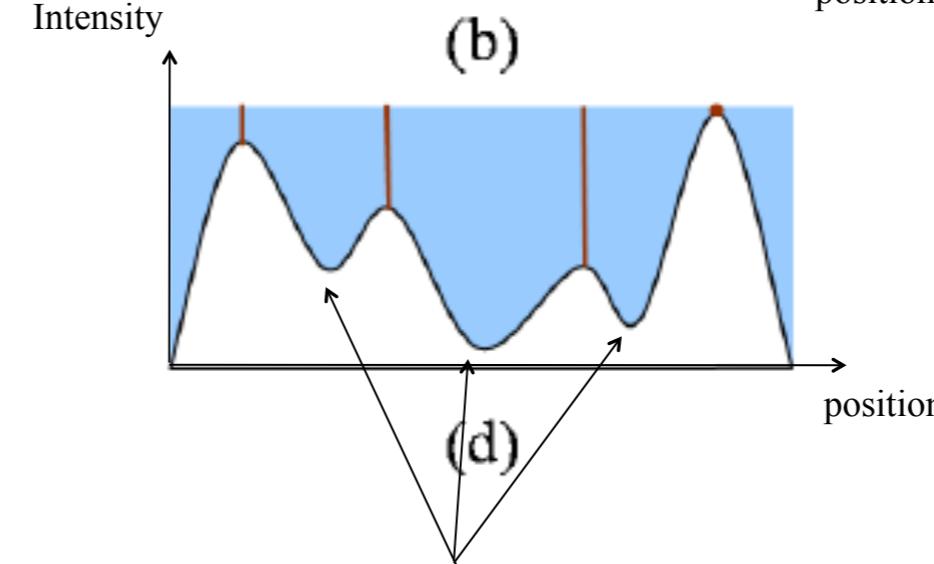
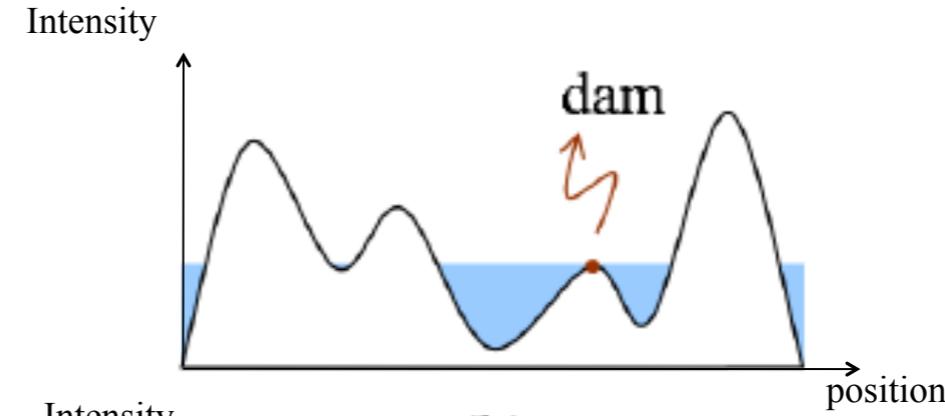
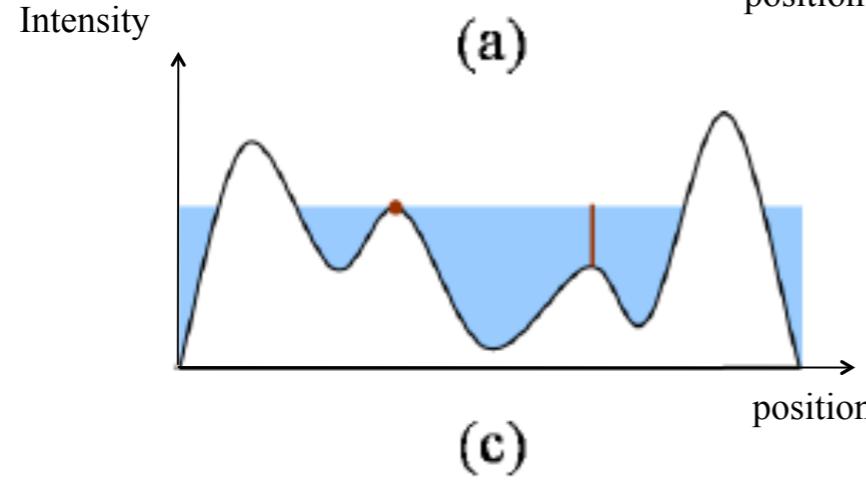
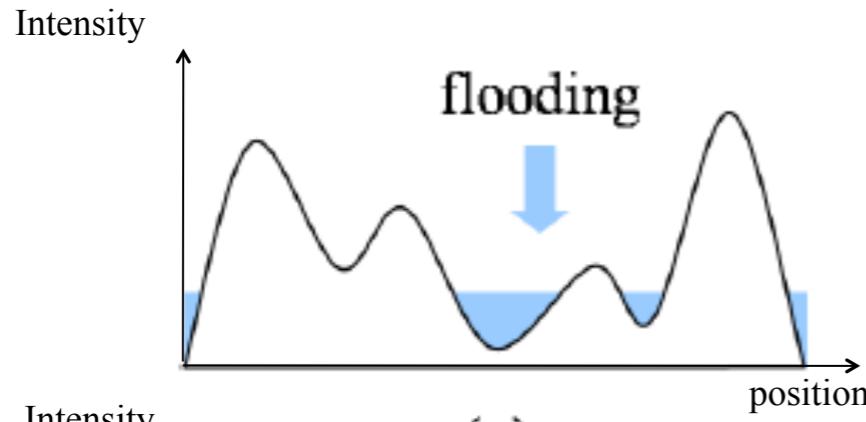
Image d'entrée



Version « topologique » de l'image



# Partage des eaux



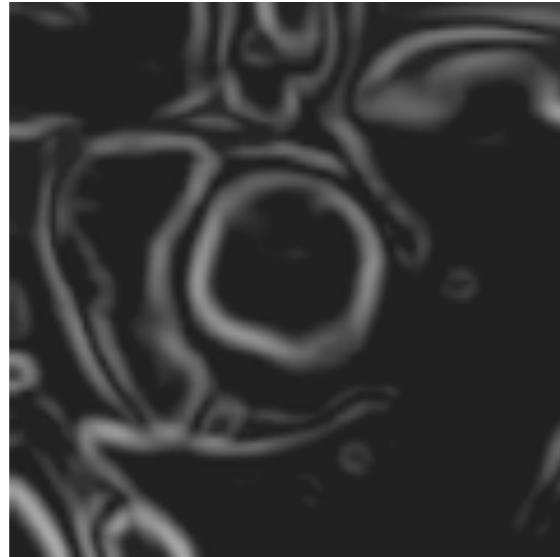
Bassins de rétention (*watersheds*)

# Partage des eaux

Pour les détails de l'algorithme...

Voir Gonzalez et woods. « **Digital Image Processing 2nd Edition** », 2002

Gradient



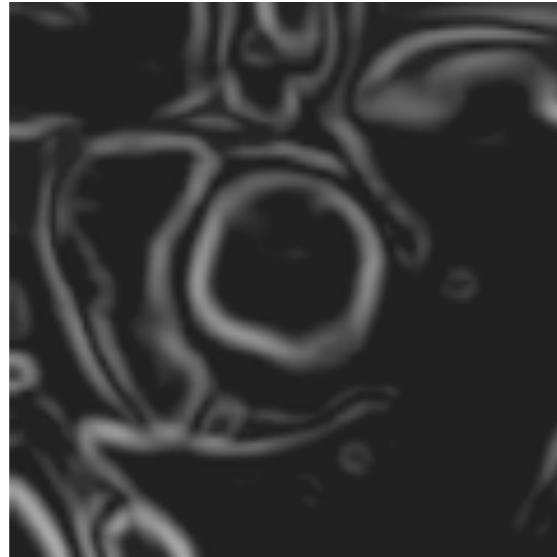
Très bon détecteur de contours

# Partage des eaux

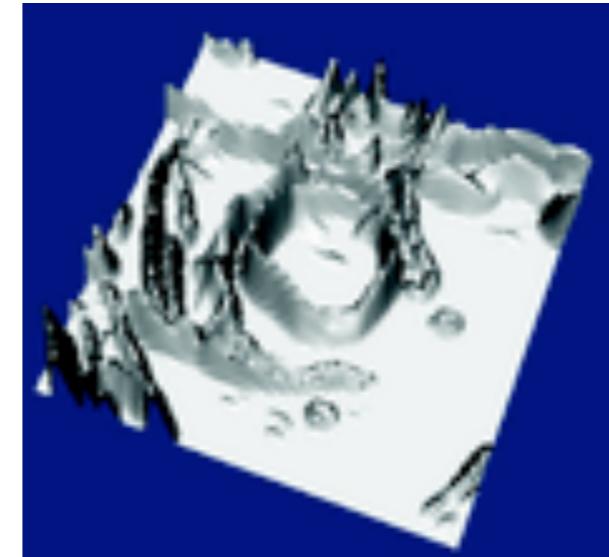
Pour les détails de l'algorithme...

Voir Gonzalez et woods. « **Digital Image Processing 2nd Edition** », 2002

Gradient



Topographie du gradient



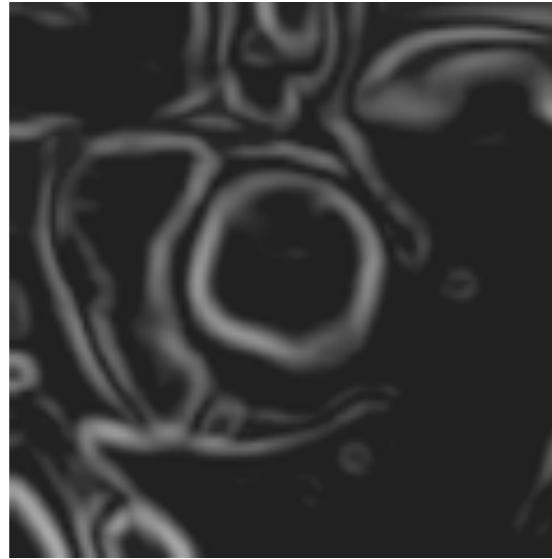
Très bon détecteur de contours

# Partage des eaux

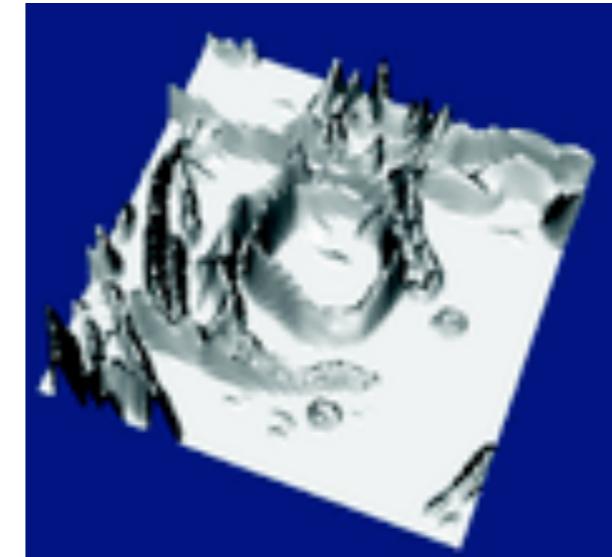
Pour les détails de l'algorithme...

Voir Gonzalez et woods. « **Digital Image Processing 2nd Edition** », 2002

Gradient



Topographie du gradient



Partage des eaux  
du gradient



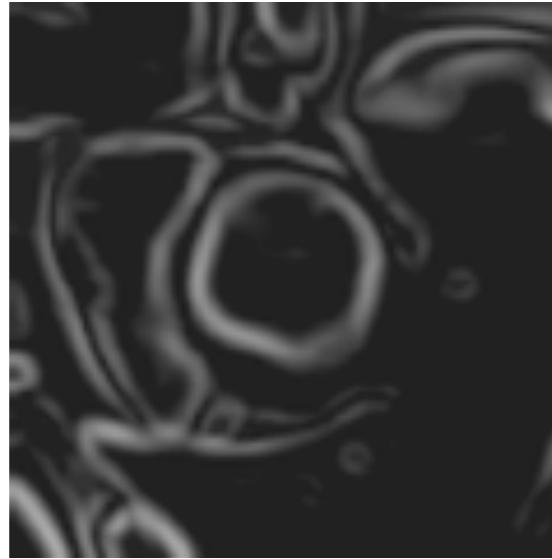
Très bon détecteur de contours

# Partage des eaux

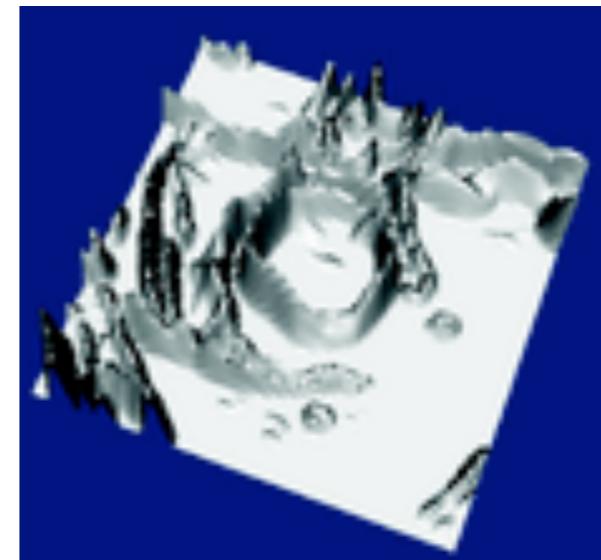
Pour les détails de l'algorithme...

Voir Gonzalez et woods. « **Digital Image Processing 2nd Edition** », 2002

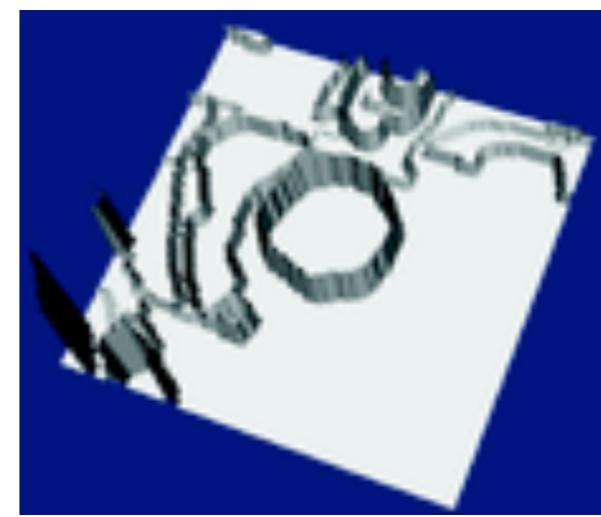
Gradient



Partage des eaux  
du gradient



Topographie du gradient



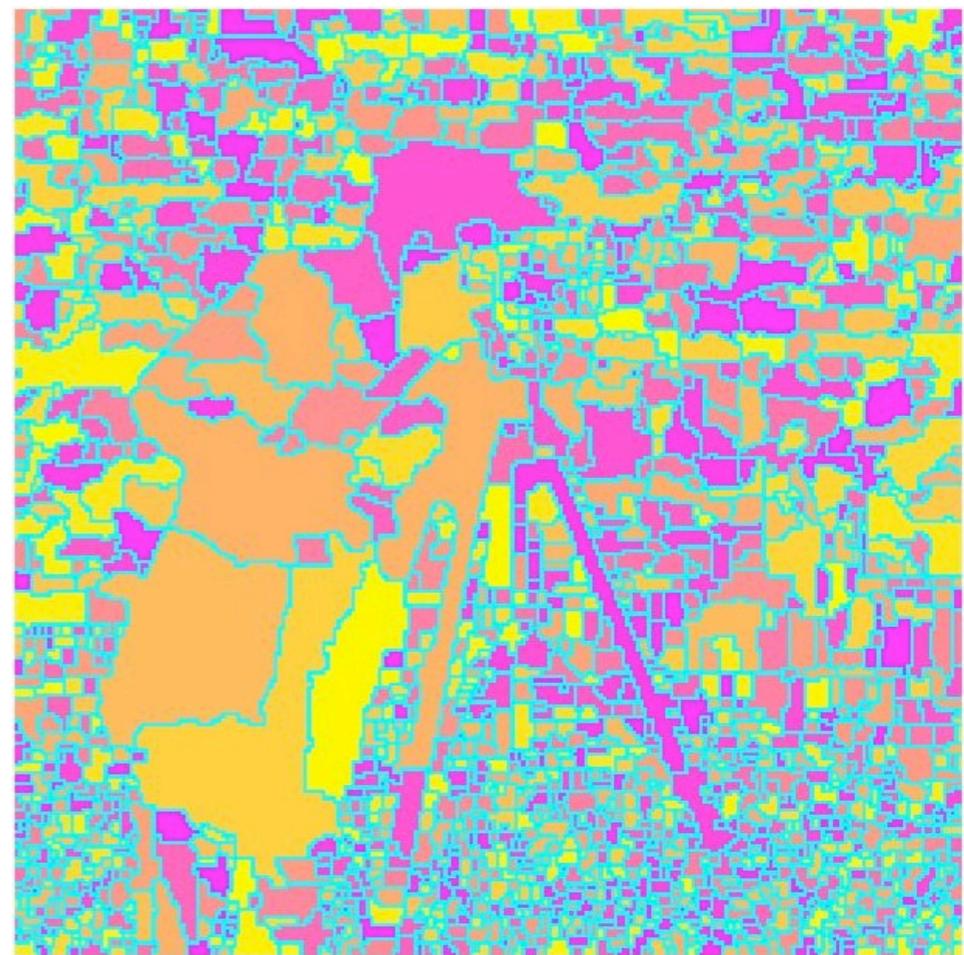
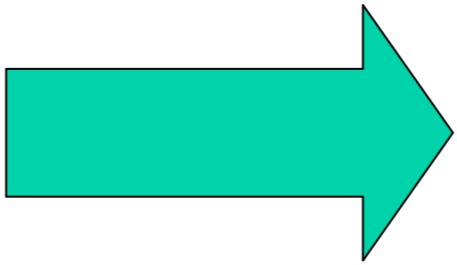
Relief du partage  
des eaux du gradient

Très bon détecteur de contours

# Partage des eaux



# Partage des eaux



## Sur-segmentation

$x \leftarrow 0$

Étiquette = 1;

POUR CHAQUE site  $s$  de l'image y FAIRE

SI  $x_s < 0$  ALORS

Innondation ( $x, y, s, \tau, \text{étiquette}$ )

Étiquette ++

## Innondation

$x_s = \text{étiquette}$

POUR chaque voisin  $r \in \eta_s$

SI  $x_r < 0$  ET  $\|y_r - y_s\| < \delta$  ET  $\|r-s\| < MAX\_SIZE$  ALORS

$x = \text{Innondation}(y, x, r, \tau, \text{iterMax}, \text{iter}+1)$

# Sursegmentation

$\|r-s\|$ : distance de Manhattan

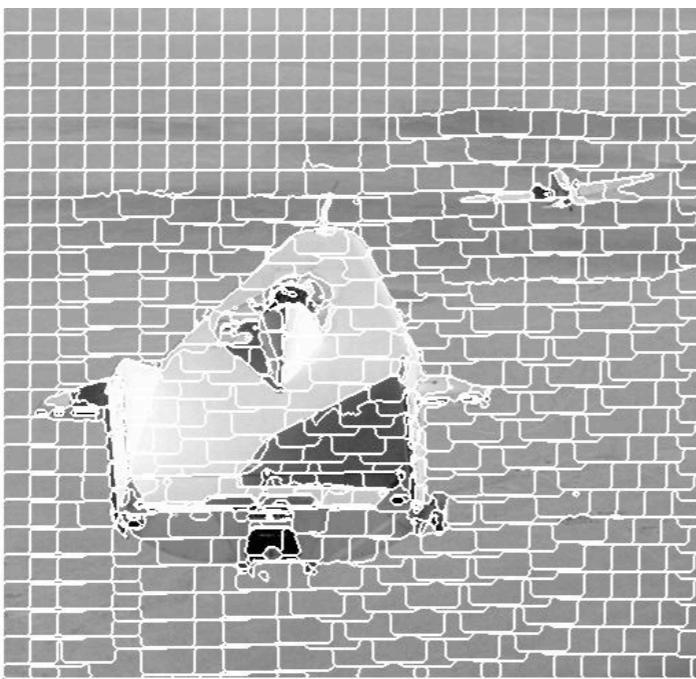
(Distance du  
chauffeur de taxi)

A  
 $(X_A, Y_A)$

B  
 $(X_B, Y_B)$



$$d(A, B) = |X_B - X_A| + |Y_B - Y_A| .$$



# Sursegmentation

$\|r-s\|$ : distance de Manhattan



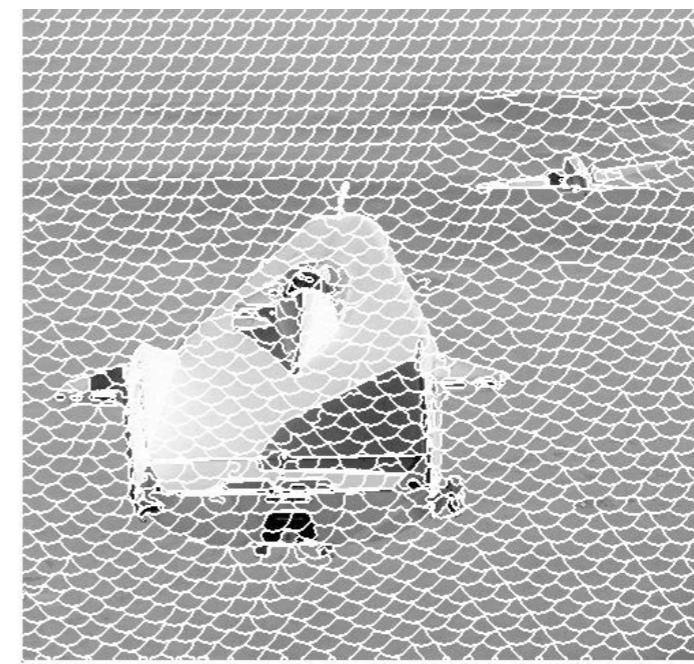
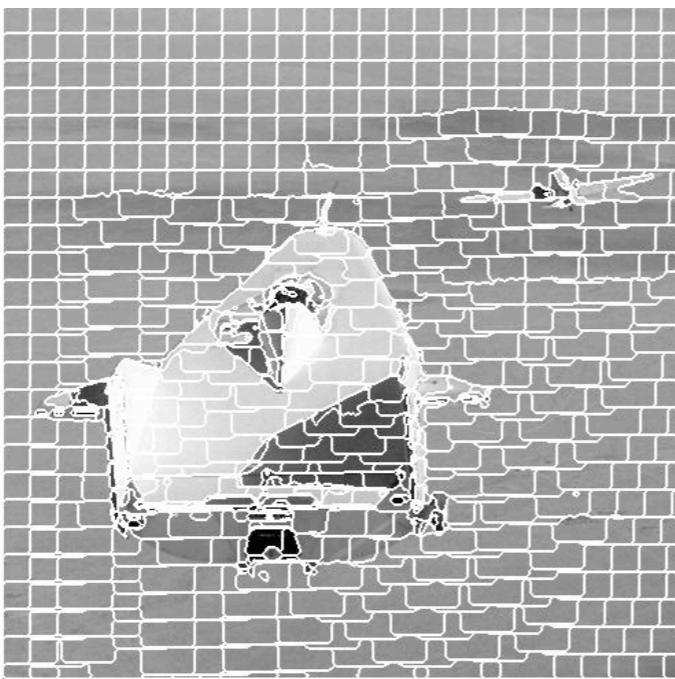
A  
( $X_A, Y_A$ )

B  
( $X_B, Y_B$ )

$\|r-s\|$ : distance Euclidienne



$$d(A, B) = |X_B - X_A| + |Y_B - Y_A| .$$



# Watershed

- Algorithme non local

# Watershed

- Algorithme non local
- Problème de sur-segmentation

# Watershed

- Algorithme non local
- Problème de sur-segmentation
- Sensibilité à tout minimum local, au bruit

# Watershed

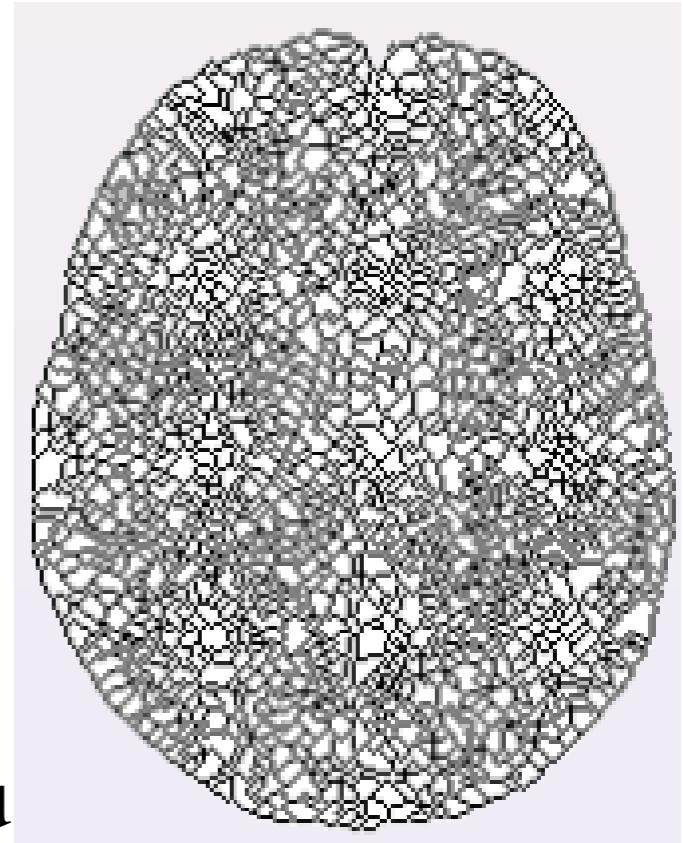
- Algorithme non local
- Problème de sur-segmentation
- Sensibilité à tout minimum local, au bruit
- Multirésolution facile à implémenter

# Watershed

- Algorithme non local
- Problème de sur-segmentation
- Sensibilité à tout minimum local, au bruit
- Multirésolution facile à implémenter
- Peut servir d'initialisation pour d'autres algos ou information complémentaire pour d'autres

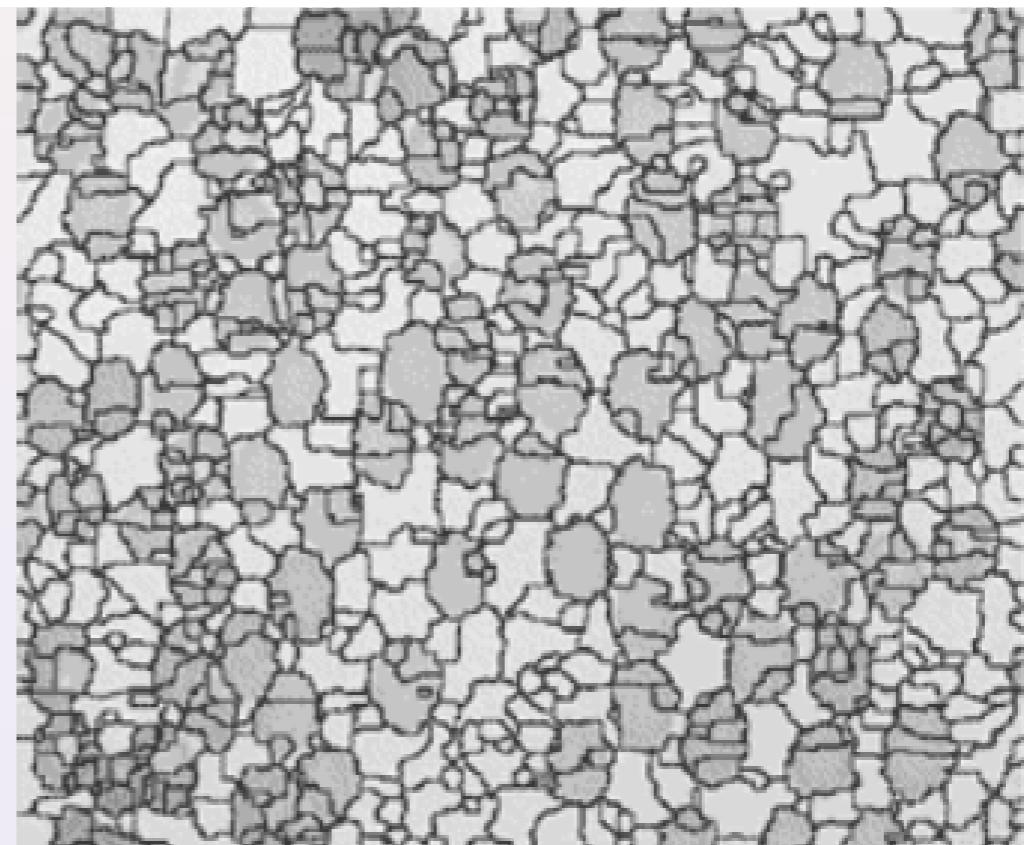
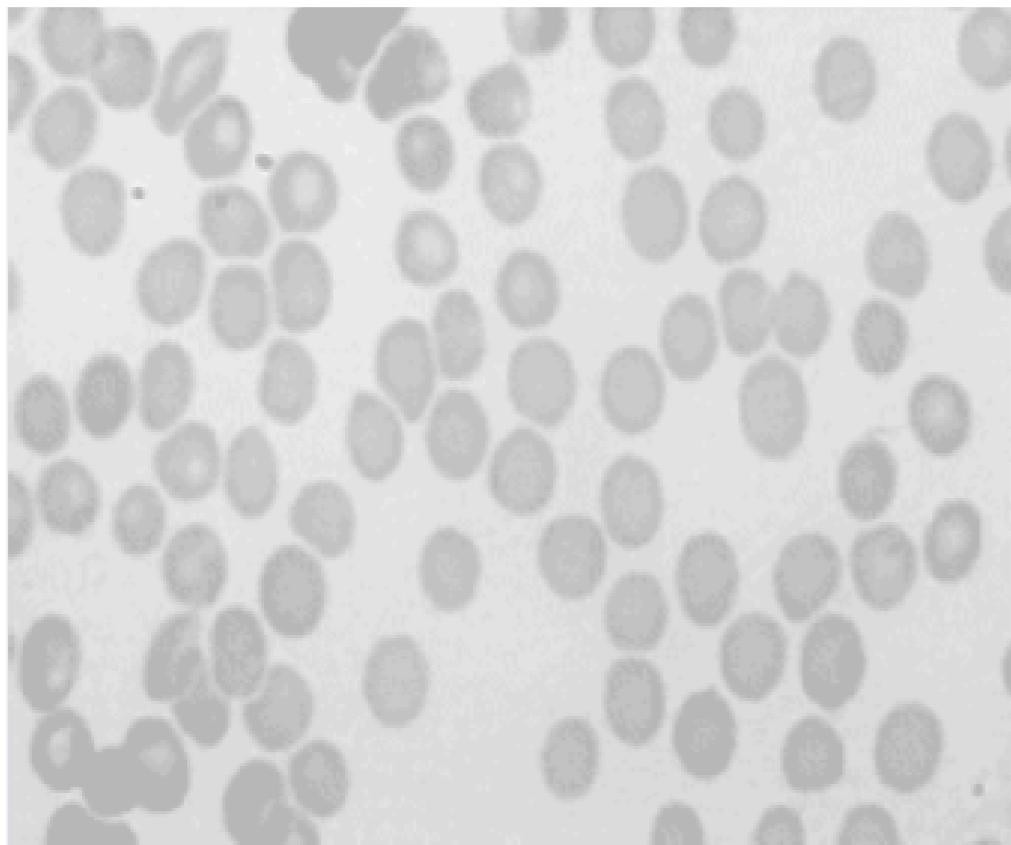
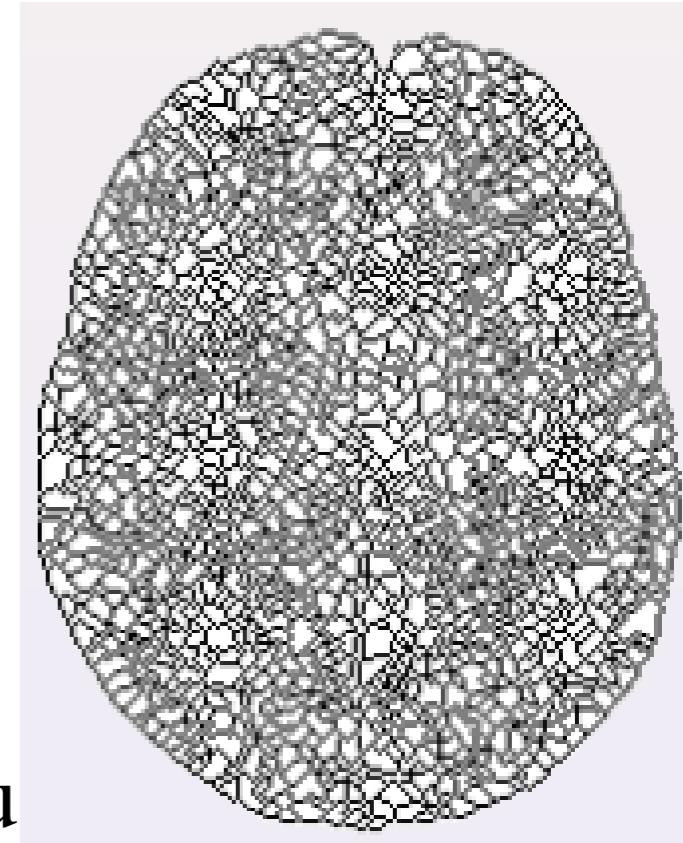
# Watershed

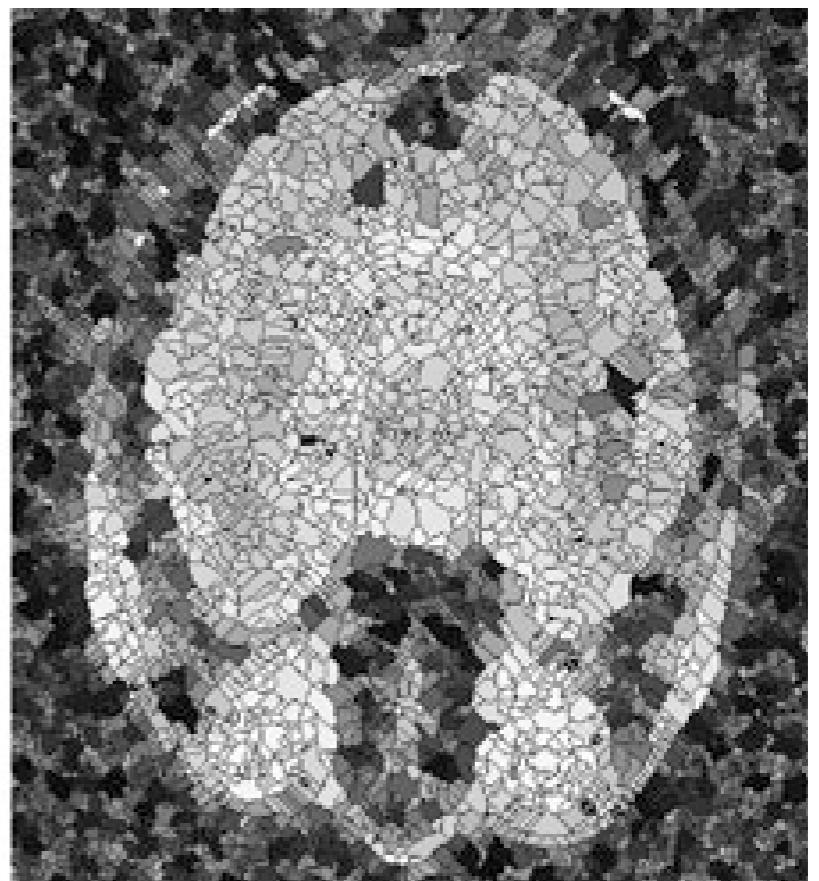
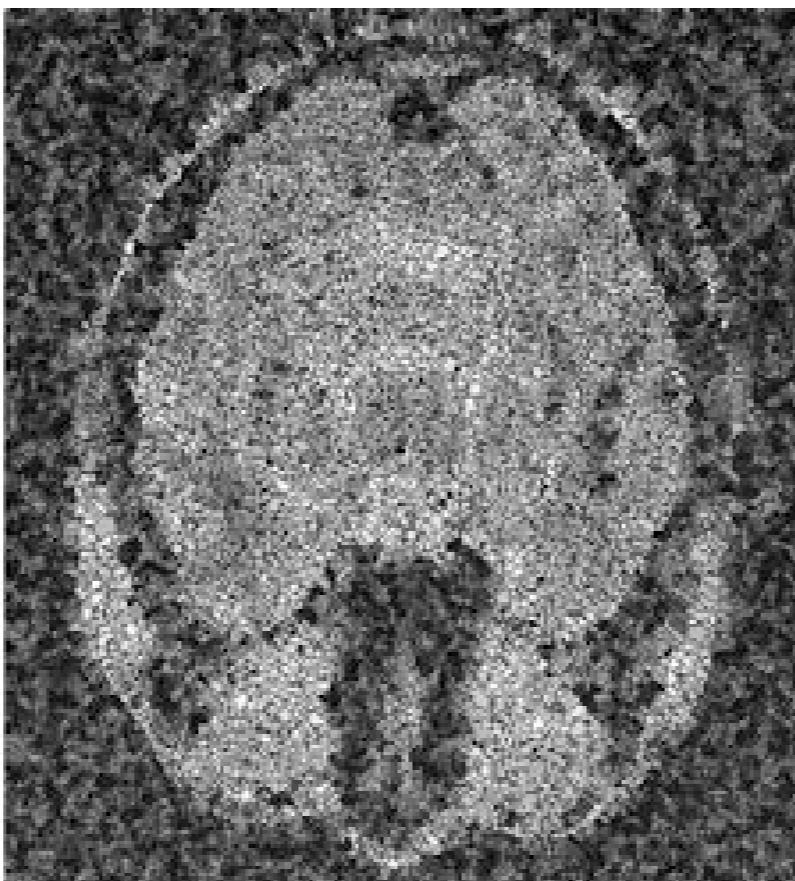
- Algorithme non local
- Problème de sur-segmentation
- Sensibilité à tout minimum local, au bruit
- Multirésolution facile à implémenter
- Peut servir d'initialisation pour d'autres algos ou information complémentaire pour d'autres

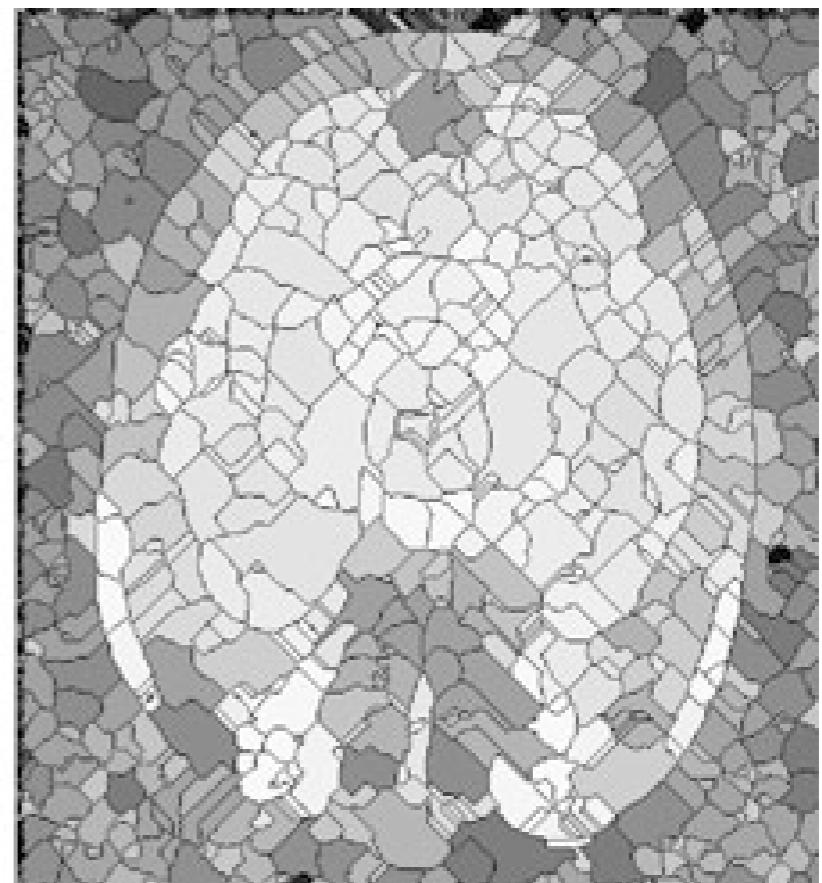
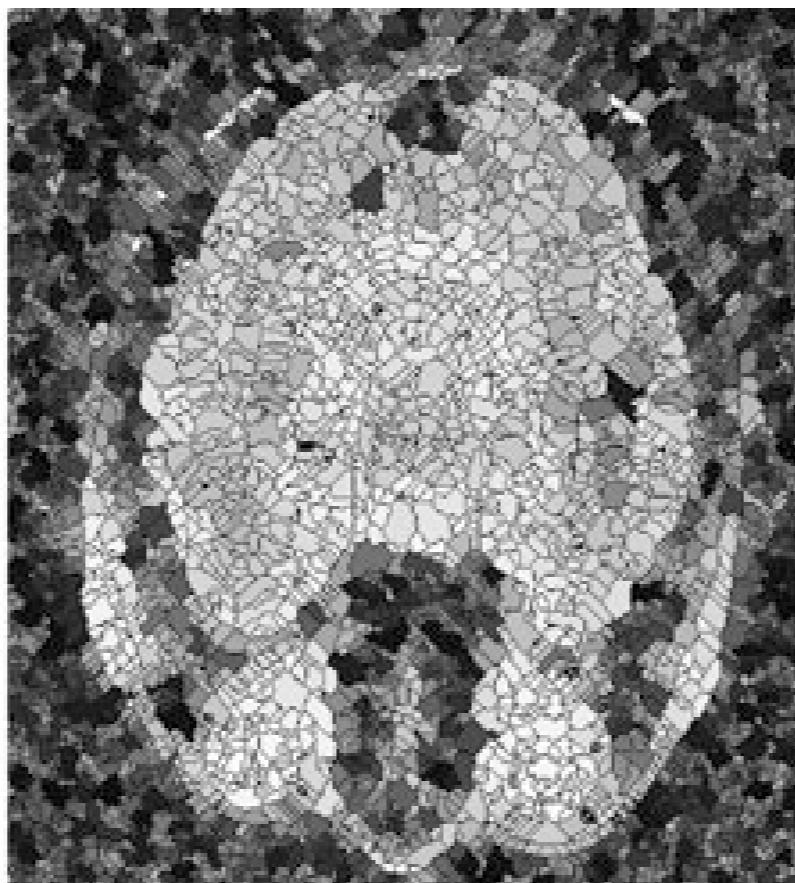
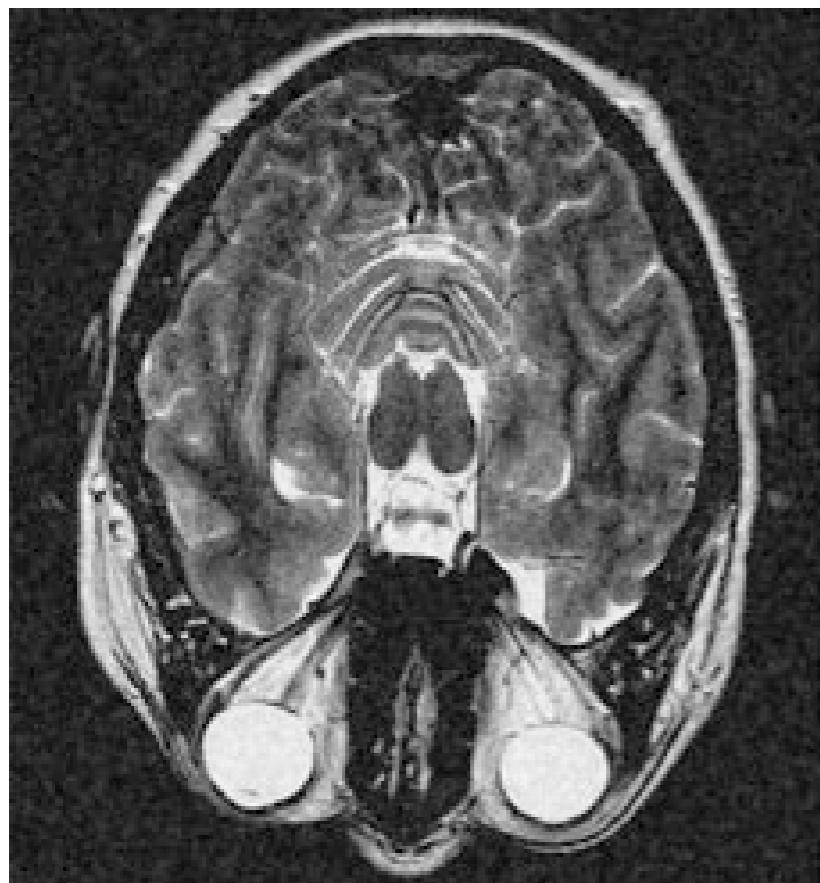
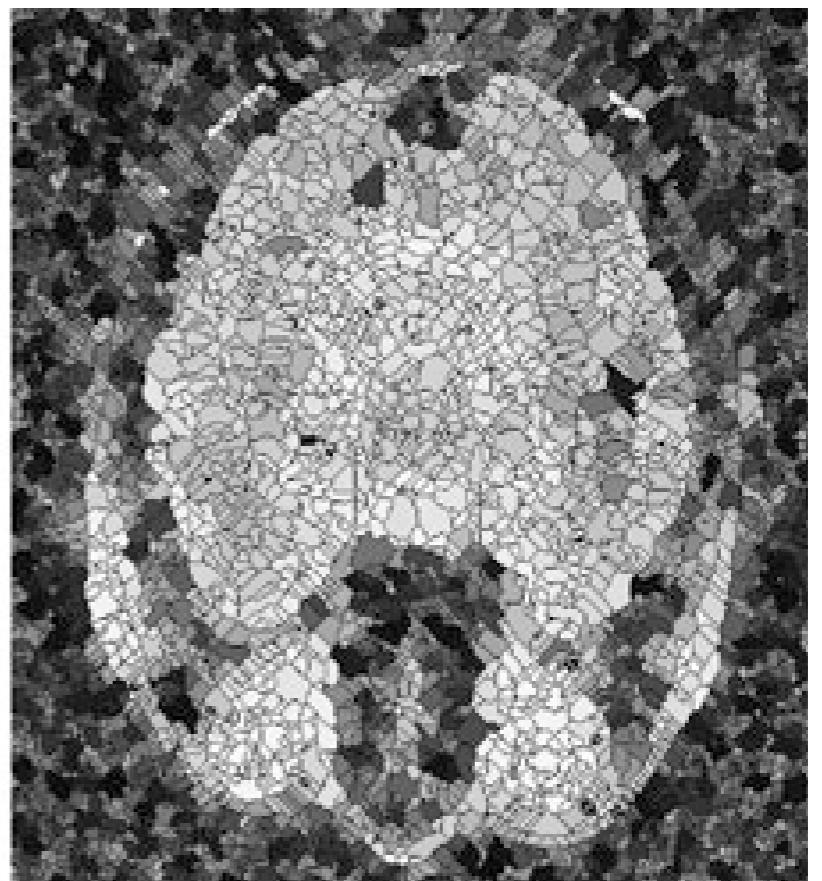
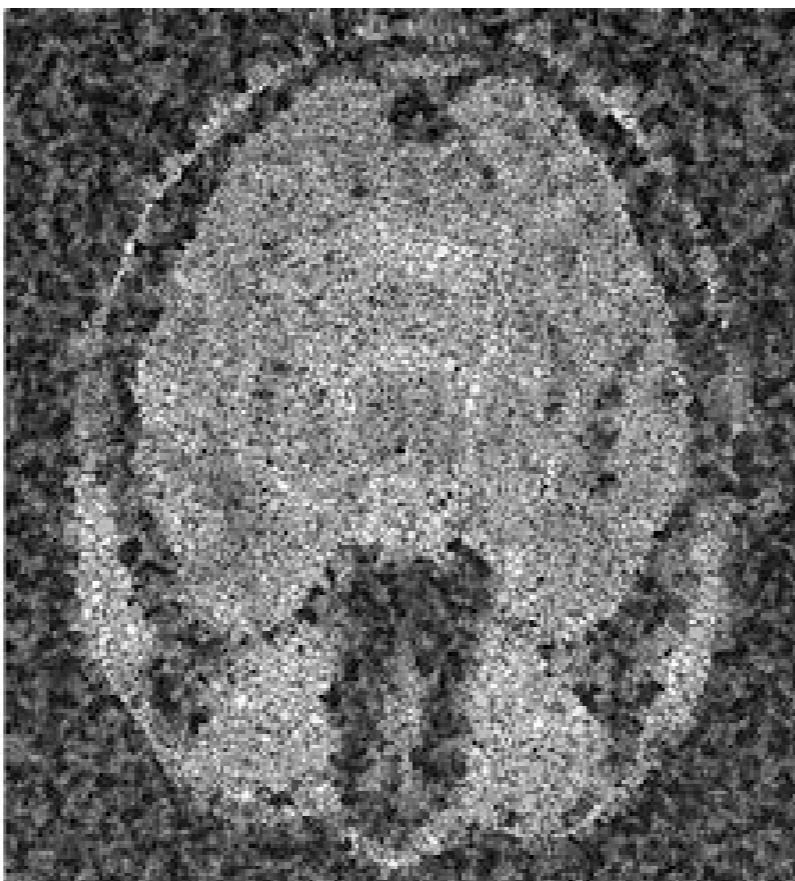


# Watershed

- Algorithme non local
- Problème de sur-segmentation
- Sensibilité à tout minimum local, au bruit
- Multirésolution facile à implémenter
- Peut servir d'initialisation pour d'autres algos ou information complémentaire pour d'autres



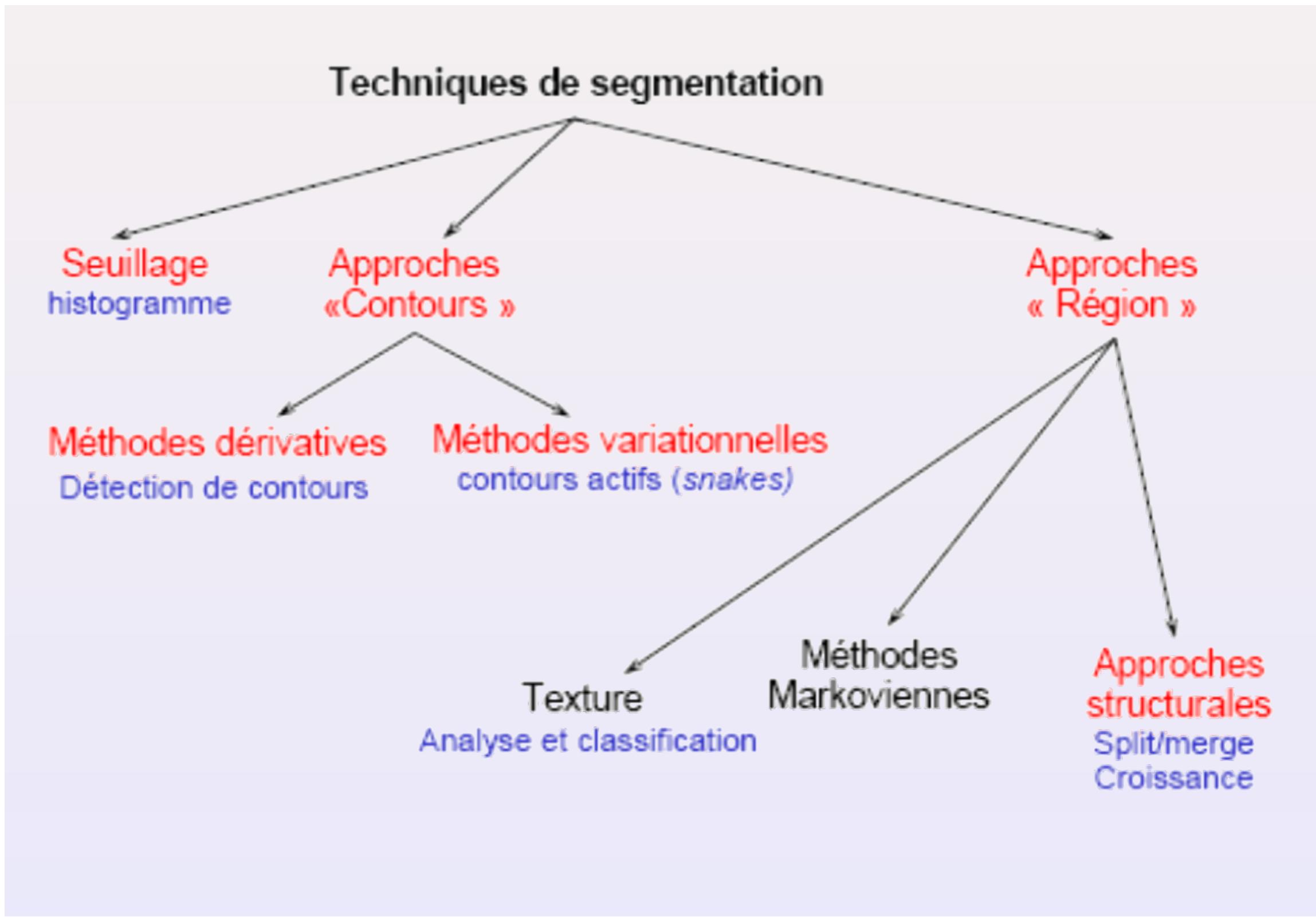


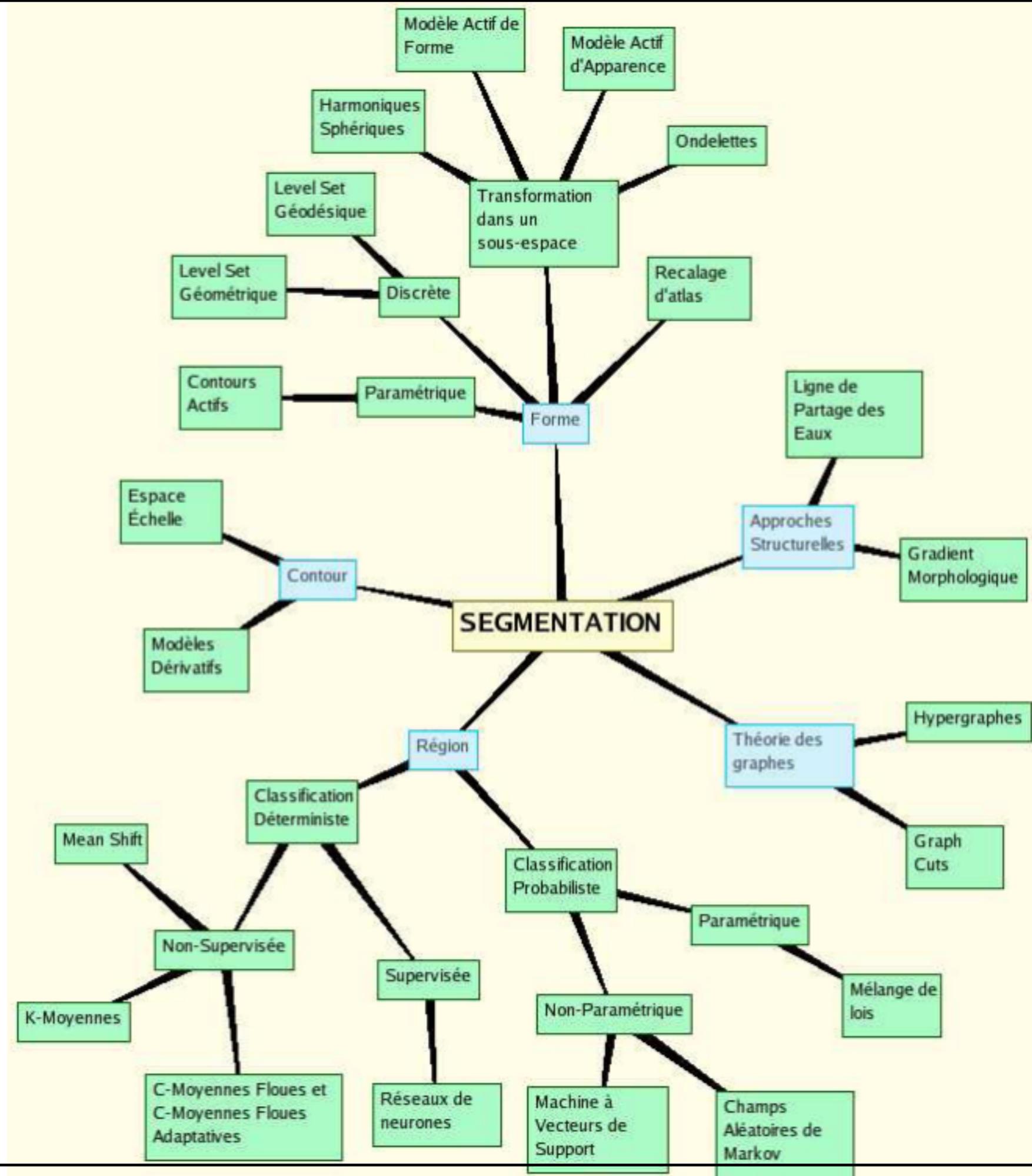


# Approches structurelles

- Morphologie mathématique (érosion, dilatation, ...)
- Partage des eaux
- Intéressant pour corriger légèrement les segmentations
- Voir notes extras

# Sommaire

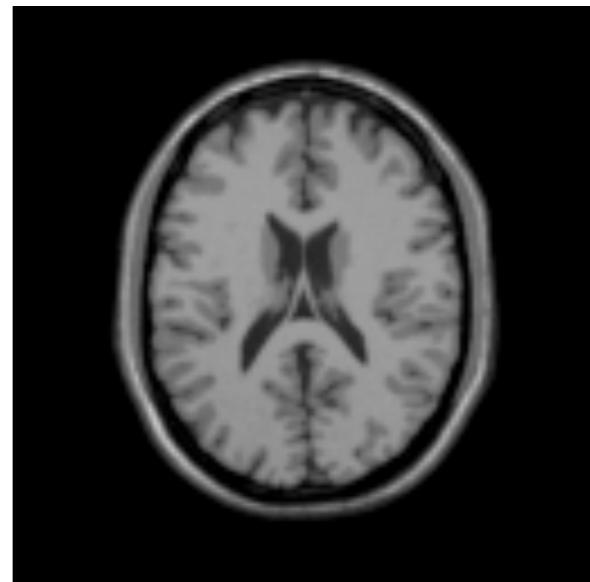
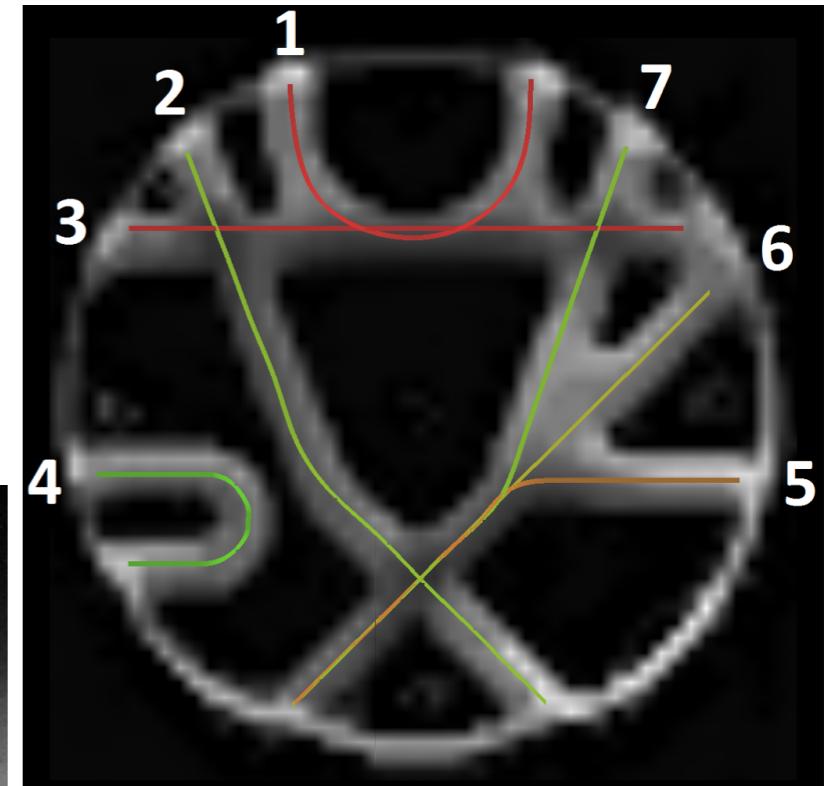
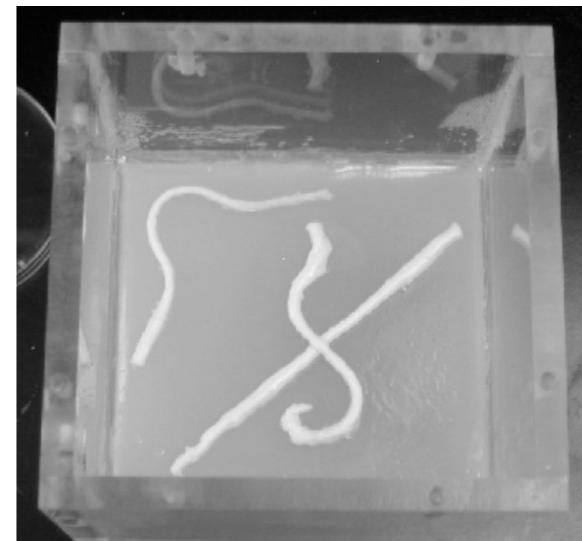




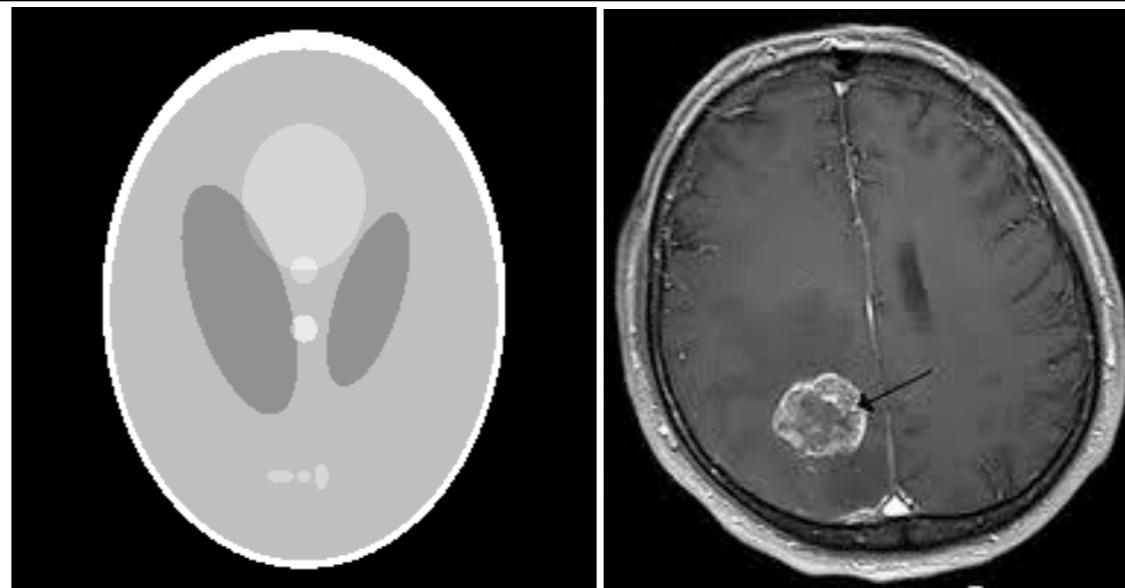
# Évaluation des segmentations

# Validation

- Fantôme numérique
  - Données synthétiques
- Fantôme ex vivo
  - biologique
  - synthétique
- Fantôme ou données terrains réelles
  - Super-cerveau ou données moyennées  
(BrainWeb et autres)

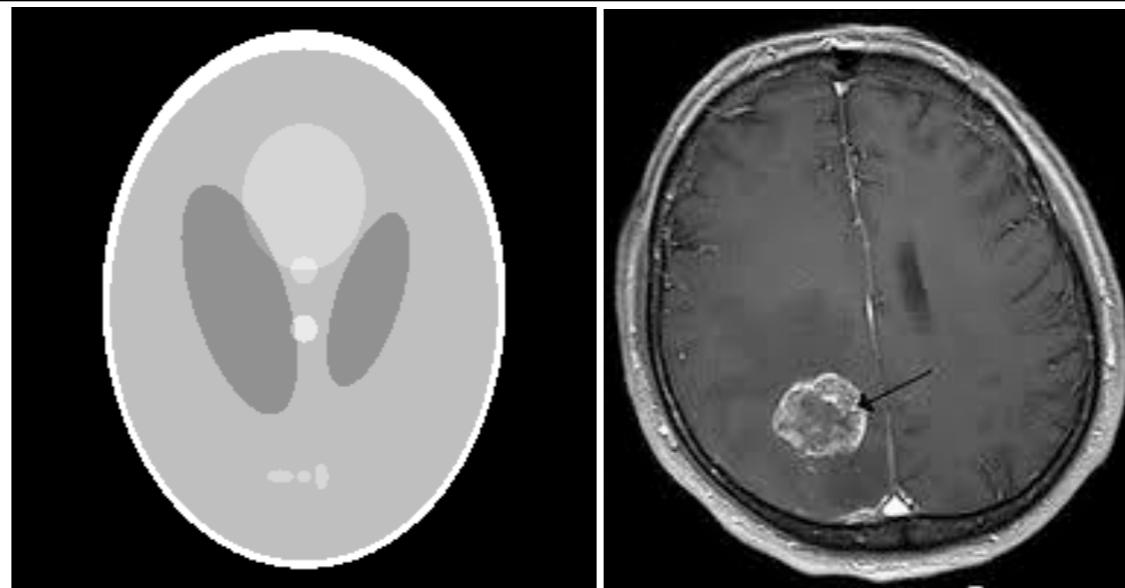


# Évaluation



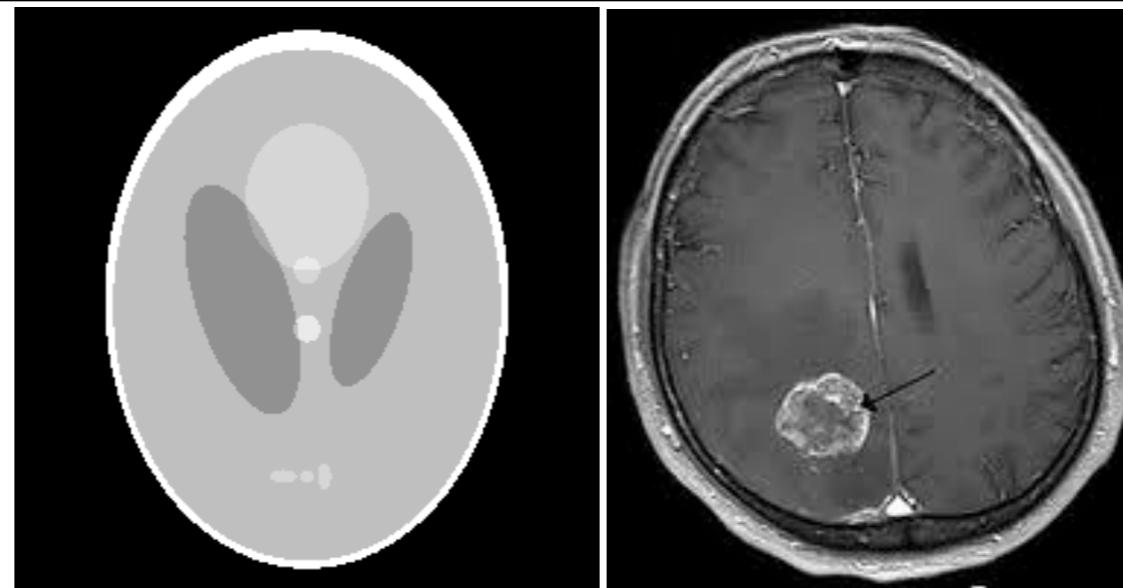
- On n'a jamais la vérité terrain (*ground truth*) dans des vraies applications médicales

# Évaluation



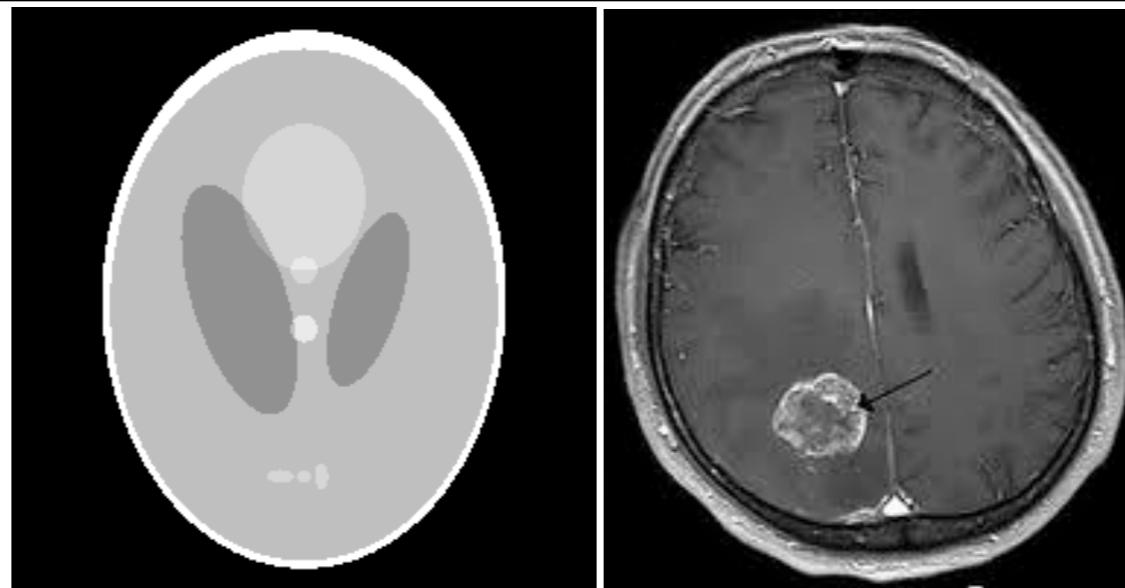
- On n'a jamais la vérité terrain (*ground truth*) dans des vraies applications médicales
- On a potentiellement une segmentation d'un ou plusieurs experts (... se méfier ... des “experts”)

# Évaluation

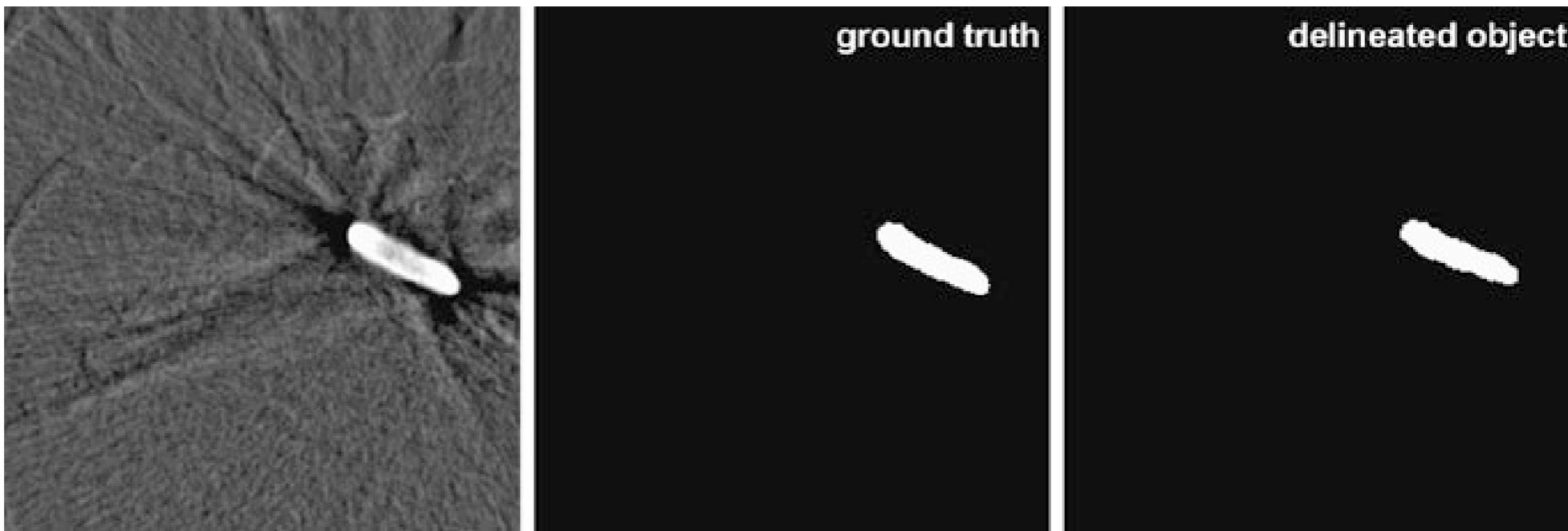


- On n'a jamais la vérité terrain (*ground truth*) dans des vraies applications médicales
- On a potentiellement une segmentation d'un ou plusieurs experts (... se méfier ... des “experts”)
- Quand on a cette segmentation d'un expert, c'est souvent sur un sous-échantillon des données

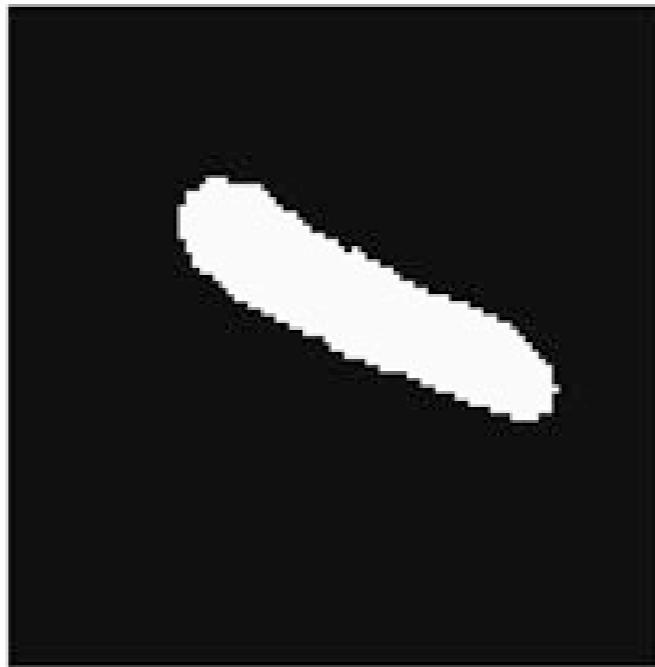
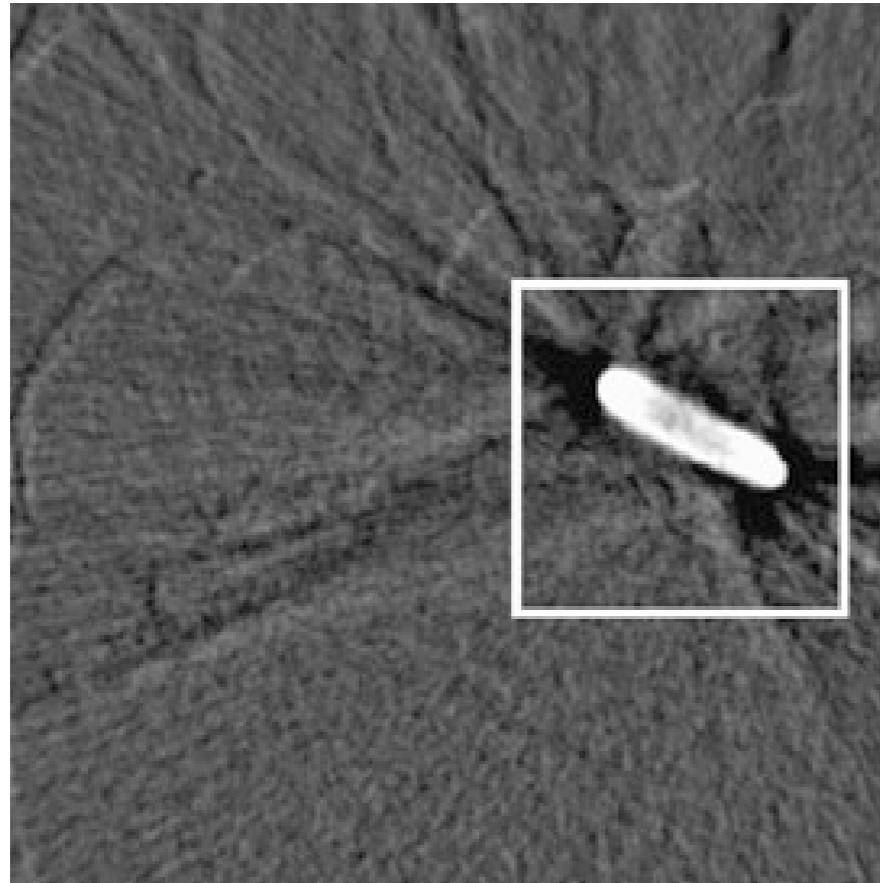
# Évaluation



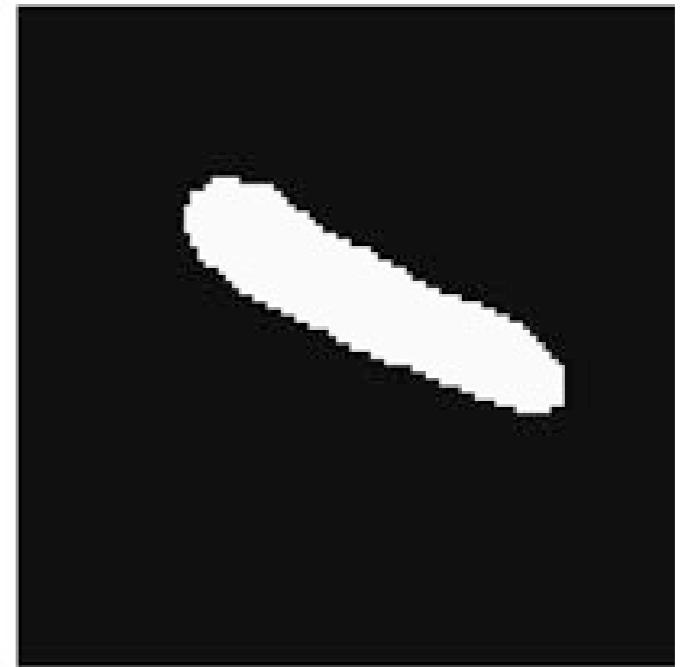
- On n'a jamais la vérité terrain (*ground truth*) dans des vraies applications médicales
- On a potentiellement une segmentation d'un ou plusieurs experts (... se méfier ... des “experts”)
- Quand on a cette segmentation d'un expert, c'est souvent sur un sous-échantillon des données



# Différence de volumes

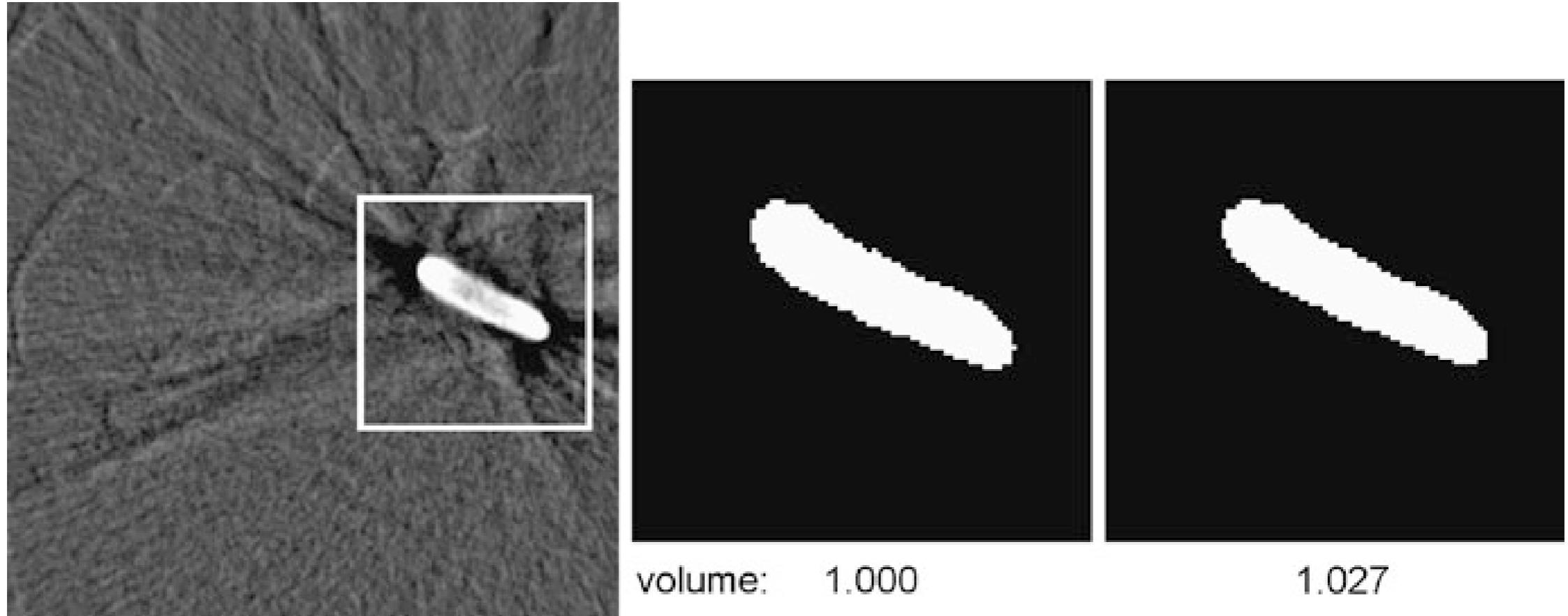


volume: 1.000



1.027

# Différence de volumes



Ne quantifie pas les différences de formes. Ne reflète pas la sur ou sous segmentation

# Métriques d'évaluation

- VP : vrai positif - nombre de voxels de l'objet bien segmenté

		object present	object not present
object found	object present	True Positive	False Positive Type I Error
	object not present	False Negative Type II Error	True Negative

matrice de confusion

# Métriques d'évaluation

- VP : vrai positif - nombre de voxels de l'objet bien segmenté
- FP : faux positif - nombre de voxels du fond faussement identifié comme de l'objet (sur-segmentation)

		object present	object not present
object found	object present	True Positive	False Positive Type I Error
	object not present	False Negative Type II Error	True Negative

matrice de confusion

# Métriques d'évaluation

- VP : vrai positif - nombre de voxels de l'objet bien segmenté
- FP : faux positif - nombre de voxels du fond faussement identifié comme de l'objet (sur-segmentation)
- FN : faux négatif - nombre de voxels de l'objet faussement identifiés comme dans le fond (sous-estimation)

		object present	object not present
object found	object found	True Positive	False Positive Type I Error
	object not found	False Negative Type II Error	True Negative
matrice de confusion			

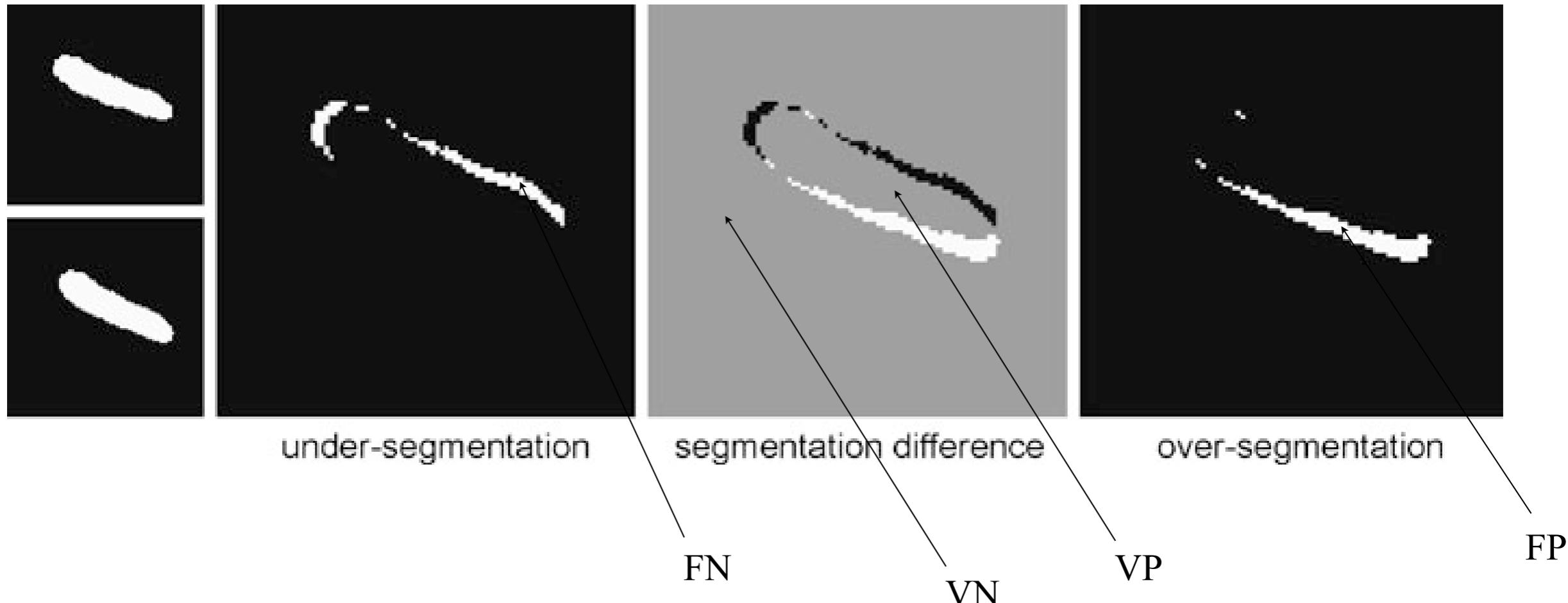
# Métriques d'évaluation

- VP : vrai positif - nombre de voxels de l'objet bien segmenté
- FP : faux positif - nombre de voxels du fond faussement identifié comme de l'objet (sur-segmentation)
- FN : faux négatif - nombre de voxels de l'objet faussement identifiés comme dans le fond (sous-estimation)
- VN : vrai négatif - nombre de voxels dans le fond bien identifié comme dans le fond

		object present	object not present
object found	object found	True Positive	False Positive Type I Error
	object not found	False Negative Type II Error	True Negative
matrice de confusion			

# Métriques d'évaluation

- $VP + FN =$  nombre de voxels dans l'objet
- $VN + FP =$  nombre de voxels dans le fond
- $VP + FN + VN + FP =$  nombre total de voxels



# Précision

- VP : vrai positif - nombre de voxels de l'objet bien segmentés
- FP : faux positif - nombre de voxels de l'objet mal segmentés
- FN : faux négatif - nombre de voxels du fond mal identifiés comme dans le fond
- VN : vrai négatif - nombre de voxels dans le fond bien identifié comme dans le fond

$$Pr = \frac{VP}{VP + FP}$$

# Précision

- VP : vrai positif - nombre de voxels de l'objet bien segmentés
- FP : faux positif - nombre de voxels de l'objet mal segmentés
- FN : faux négatif - nombre de voxels du fond mal identifiés comme dans le fond
- VN : vrai négatif - nombre de voxels dans le fond bien identifié comme dans le fond

$$Pr = \frac{VP}{VP + FP}$$

Haute Pr indique que peu de voxels du fond ont été classés comme de l'objet

# Rappel ou sensibilité

- VP : vrai positif - nombre de voxels de l'objet bien segmentés
- FP : faux positif - nombre de voxels de l'objet mal segmentés
- FN : faux négatif - nombre de voxels du fond mal identifiés comme dans le fond
- VN : vrai négatif - nombre de voxels dans le fond bien identifié comme dans le fond

$$Ra = \frac{VP}{VP + FN}$$

# Rappel ou sensibilité

- VP : vrai positif - nombre de voxels de l'objet bien segmentés
- FP : faux positif - nombre de voxels de l'objet mal segmentés
- FN : faux négatif - nombre de voxels du fond mal identifiés comme dans le fond
- VN : vrai négatif - nombre de voxels dans le fond bien identifié comme dans le fond

$$Ra = \frac{VP}{VP + FN}$$

Haute Pr indique que la plupart des voxels de l'objet ont été bien classés

# Spécificité

- VP : vrai positif - nombre de voxels de l'objet bien segmentés
- FP : faux positif - nombre de voxels de l'objet mal segmentés
- FN : faux négatif - nombre de voxels du fond mal identifiés comme dans le fond
- VN : vrai négatif - nombre de voxels dans le fond bien identifié comme dans le fond

$$Sp = \frac{VN}{VN + FP}$$

# Spécificité

- VP : vrai positif - nombre de voxels de l'objet bien segmentés
- FP : faux positif - nombre de voxels de l'objet mal segmentés
- FN : faux négatif - nombre de voxels du fond mal identifiés comme dans le fond
- VN : vrai négatif - nombre de voxels dans le fond bien identifié comme dans le fond

$$Sp = \frac{VN}{VN + FP}$$

Haute Pr indique que peu de voxels du fond ont été classés dans l'objet

# Autres métriques

- f-mesure :  $2(\text{Pr} * \text{Ra}) / (\text{Pr} + \text{Ra})$ 
  - aussi appelé le coefficient de Dice
- Taux de VN, FP et FN

$$T_{FN} = \frac{FN}{VP + FN}$$

$$T_{FP} = \frac{FP}{FP + VN}$$

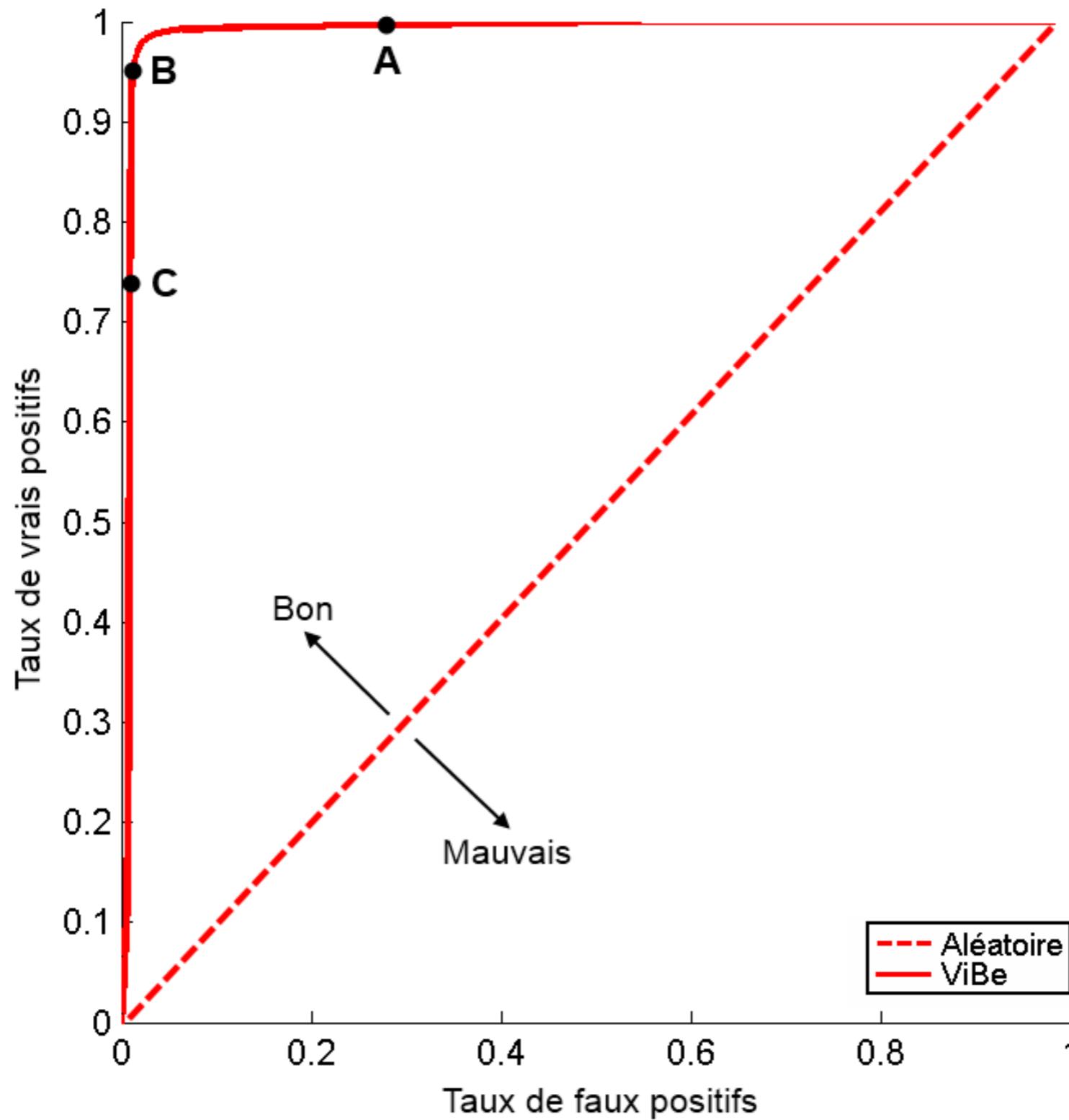
- Pourcentage de bonnes et mauvaises classification

$$PBC = 100 \frac{VP + VN}{VP + FN + FP + VN}$$

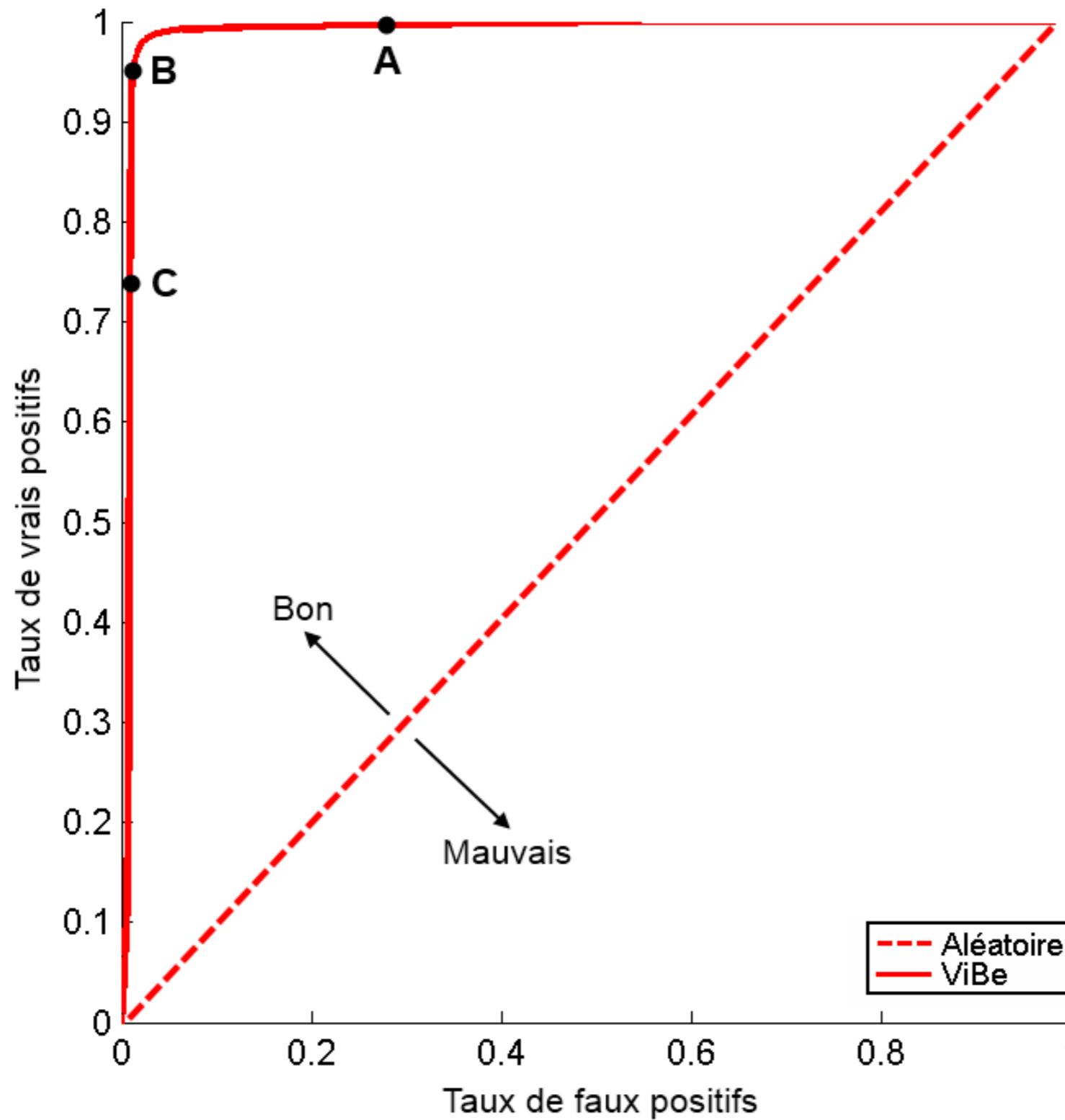
$$PMC = 100 \frac{FN + FP}{VP + FN + FP + VN}$$

# Courbe ROC (receiver operating characteristic)

# Courbe ROC (receiver operating characteristic)



# Courbe ROC (receiver operating characteristic)



Plusieurs auteurs préfèrent la courbe Précision vs Rappel

# Métriques - jamais unanime

(chap 13 Toennies Med IA 2012)

- Coefficients de Jaccard :

$$C_J = \frac{VP}{VP + FP + FN}$$

- Coefficients de Yule :

$$C_Y = \left| \frac{VP}{VP + FP} + \frac{VN}{VN + FN} - 1 \right|$$

# Métriques avancées

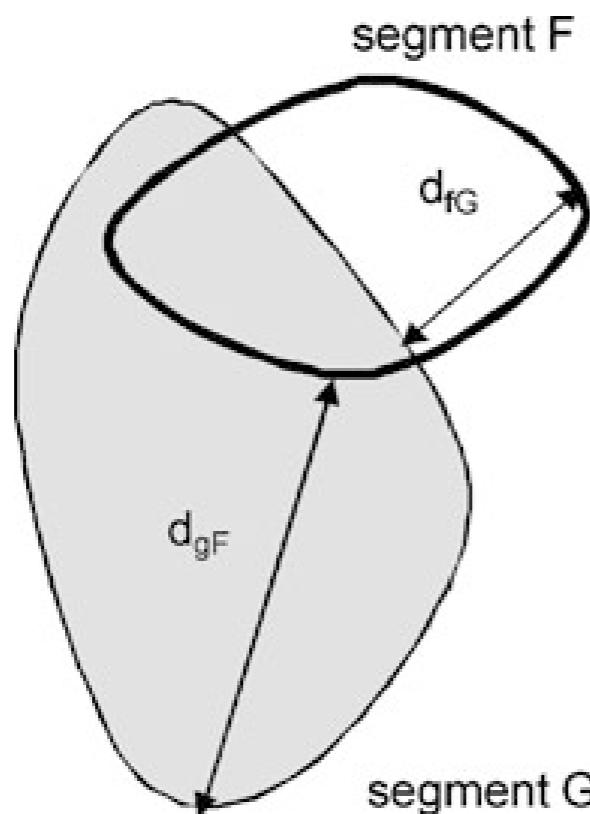
- Distance Euclidienne entre les volumes (formes)
  - Besoin de la fonction de distance (comme dans les levelsets)  
[Descoteaux et al. Media 2008]

# Métriques avancées

- Distance Euclidienne entre les volumes (formes)
  - Besoin de la fonction de distance (comme dans les levelsets)  
[Descoteaux et al. Media 2008]
- Mesure d'Hausdorff

# Métriques avancées

- Distance Euclidienne entre les volumes (formes)
  - Besoin de la fonction de distance (comme dans les levelsets)  
[Descoteaux et al. Media 2008]
- Mesure d'Hausdorff

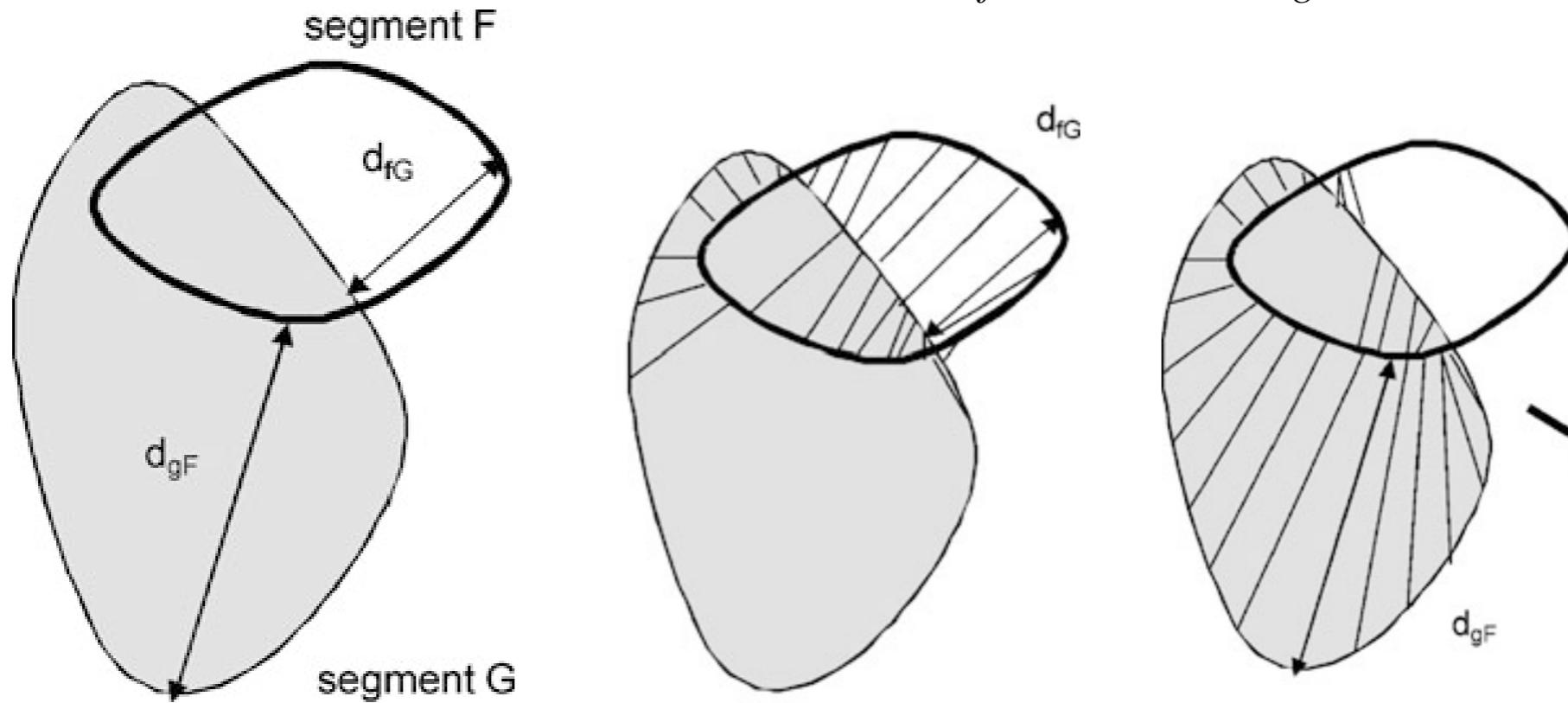


# Métriques avancées

- Distance Euclidienne entre les volumes (formes)
  - Besoin de la fonction de distance (comme dans les levelsets)  
[Descoteaux et al. Media 2008]

- Mesure d'Hausdorff

$$h = \max \left( \inf_{f \in F} d(f, G), \inf_{g \in G} d(g, F) \right).$$



# Évaluations de multiples experts

- Les experts ne sont pas toujours d'accord

# Évaluations de multiples experts

- Les experts ne sont pas toujours d'accord
- Variabilité inter-observateur

# Évaluations de multiples experts

- Les experts ne sont pas toujours d'accord
- Variabilité inter-observateur
- Comment quantifié cette variabilité entre experts?

# Évaluations de multiples experts

- Les experts ne sont pas toujours d'accord
- Variabilité inter-observateur
- Comment quantifié cette variabilité entre experts?
- Chevauchement maximal atteint par un vote majoritaire  
[Warfield et al 1995]

# Évaluations de multiples experts

- Les experts ne sont pas toujours d'accord
- Variabilité inter-observateur
- Comment quantifié cette variabilité entre experts?
- Chevauchement maximal atteint par un vote majoritaire [Warfield et al 1995]
- Soit on cherche l'unanimité ou un entre-deux

# Évaluations de multiples experts

- Les experts ne sont pas toujours d'accord
- Variabilité inter-observateur
- Comment quantifié cette variabilité entre experts?
- Chevauchement maximal atteint par un vote majoritaire  
[Warfield et al 1995]
- Soit on cherche l'unanimité ou un entre-deux
- STAPLE :  
Simultaneous Truth Performance Level Estimation.  
[Warfield et al 2004]

# Intersection, union et vote majoritaire

- L'intersection ou l'union entre les segmentations donnent une idée de la variabilité entre les segmentations

# Intersection, union et vote majoritaire

- L'intersection ou l'union entre les segmentations donnent une idée de la variabilité entre les segmentations
- Peu de chance que ça donne une représentation “moyenne” intéressante...

# Intersection, union et vote majoritaire

- L'intersection ou l'union entre les segmentations donnent une idée de la variabilité entre les segmentations
- Peu de chance que ça donne une représentation “moyenne” intéressante...

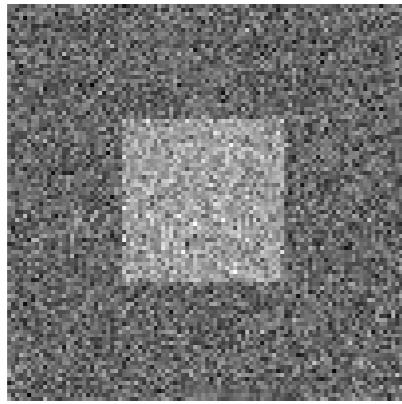
# Intersection, union et vote majoritaire

- L'intersection ou l'union entre les segmentations donnent une idée de la variabilité entre les segmentations
- Peu de chance que ça donne une représentation “moyenne” intéressante...
- Le vote majoritaire est une stratégie plus évoluée
  - On applique la règle du “majority vote”
  - L'étiquette qui revient le plus fréquemment dans les différentes segmentation initiales est conservée à chaque site

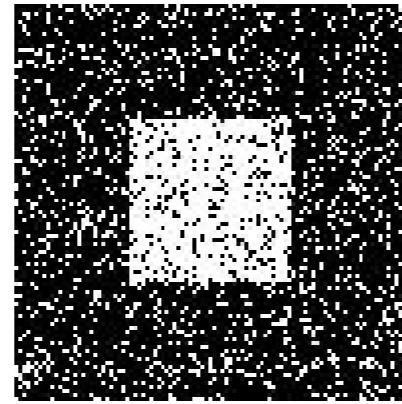
# Intersection, union et vote majoritaire

- L'intersection ou l'union entre les segmentations donnent une idée de la variabilité entre les segmentations
- Peu de chance que ça donne une représentation “moyenne” intéressante...
- Le vote majoritaire est une stratégie plus évoluée
  - On applique la règle du “majority vote”
  - L'étiquette qui revient le plus fréquemment dans les différentes segmentation initiales est conservée à chaque site
- Démo

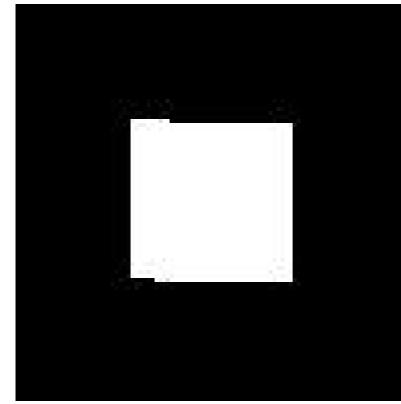
# Exemple jouet



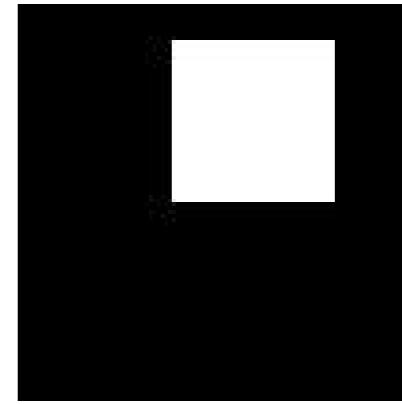
I\_toy



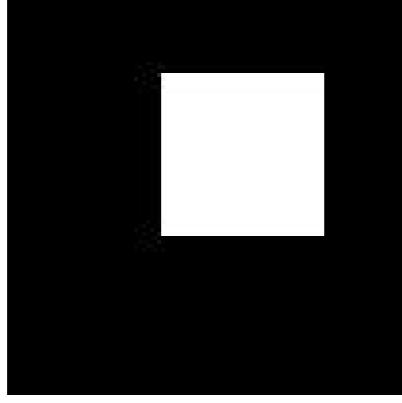
Segm\_toy(:,:,1)



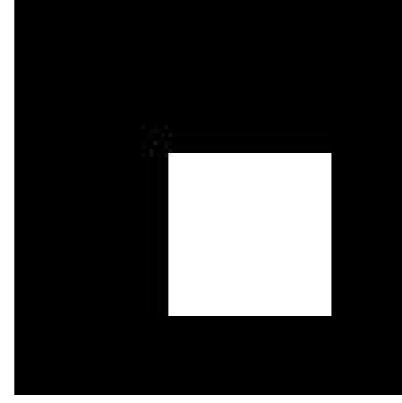
Segm\_toy(:,:,2)



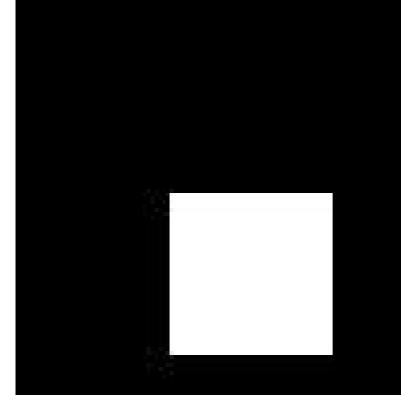
Segm\_toy(:,:,3)



Segm\_toy(:,:,4)



Segm\_toy(:,:,5)

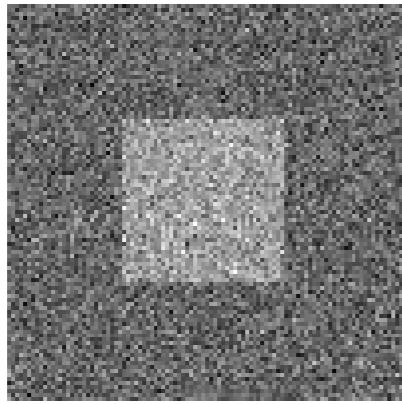


Segm\_toy(:,:,6)

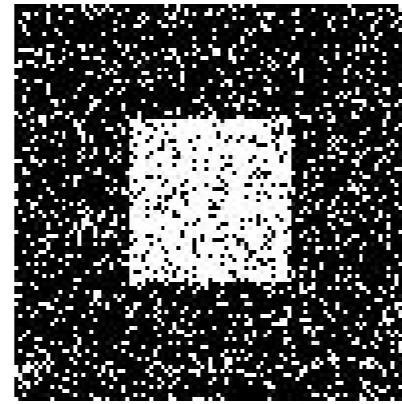


Segm\_toy(:,:,7)

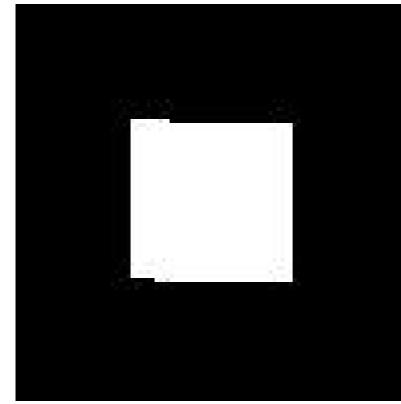
# Exemple jouet



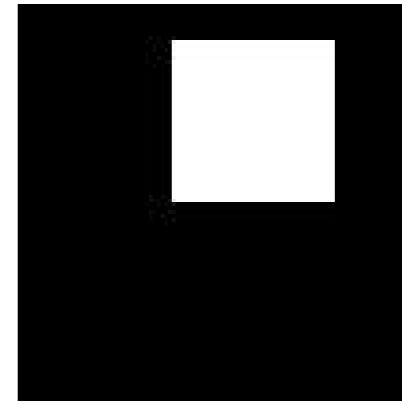
I\_toy



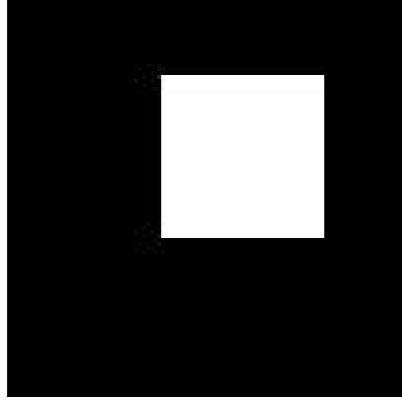
Segm\_toy(:,:,1)



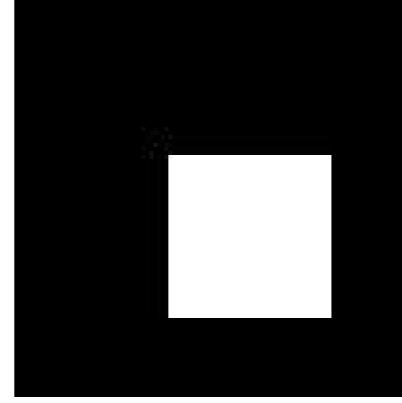
Segm\_toy(:,:,2)



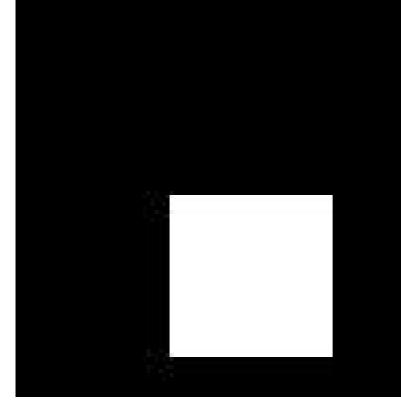
Segm\_toy(:,:,3)



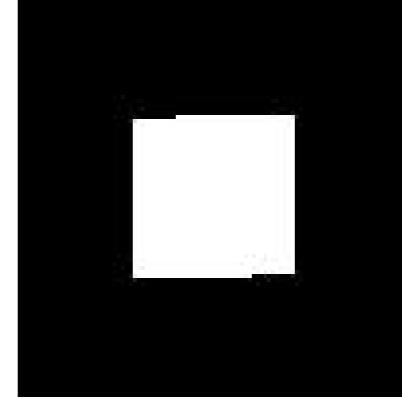
Segm\_toy(:,:,4)



Segm\_toy(:,:,5)



Segm\_toy(:,:,6)

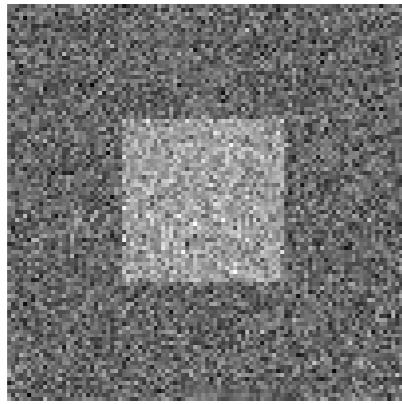


Segm\_toy(:,:,7)

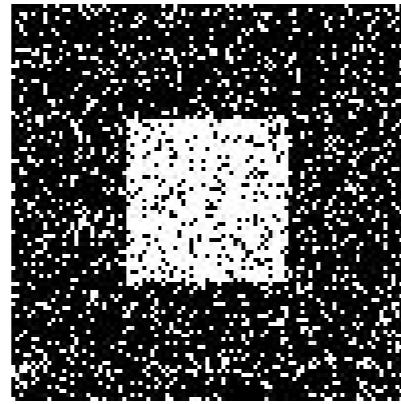


Intersection

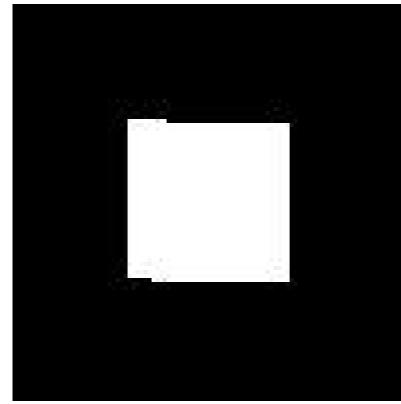
# Exemple jouet



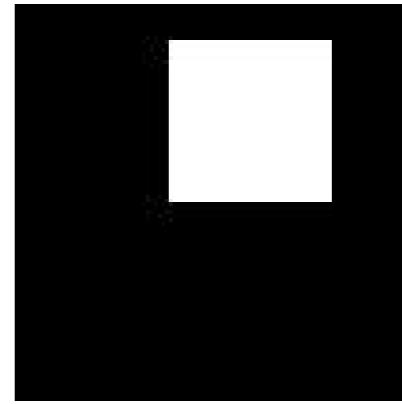
I\_toy



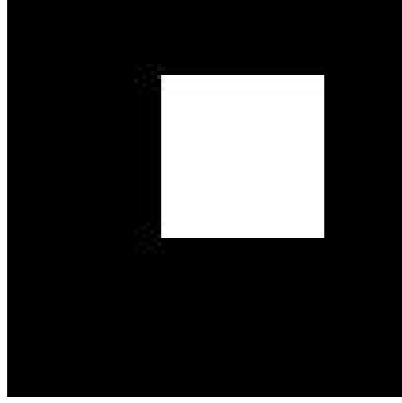
Segm\_toy(:,:,1)



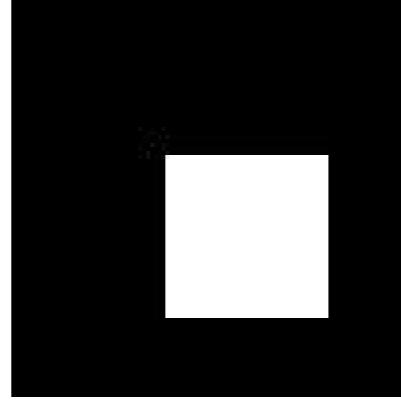
Segm\_toy(:,:,2)



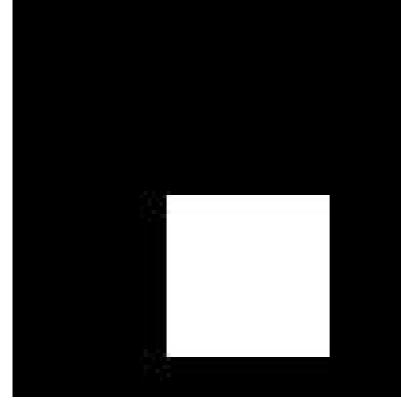
Segm\_toy(:,:,3)



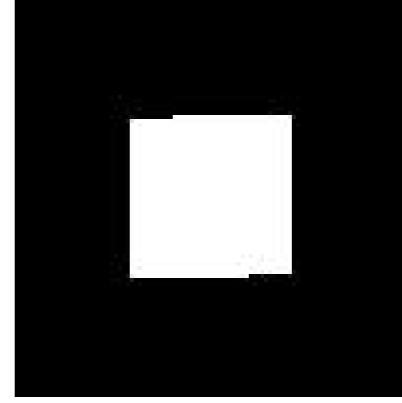
Segm\_toy(:,:,4)



Segm\_toy(:,:,5)



Segm\_toy(:,:,6)



Segm\_toy(:,:,7)

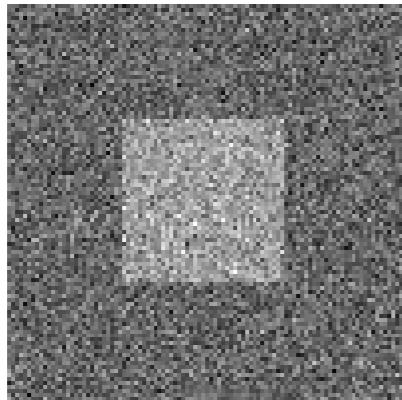


Intersection

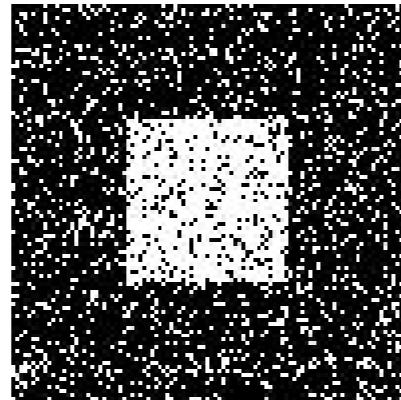


Union

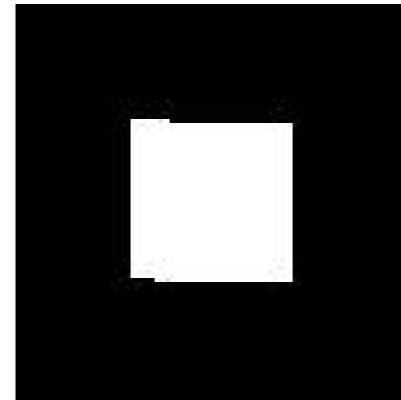
# Exemple jouet



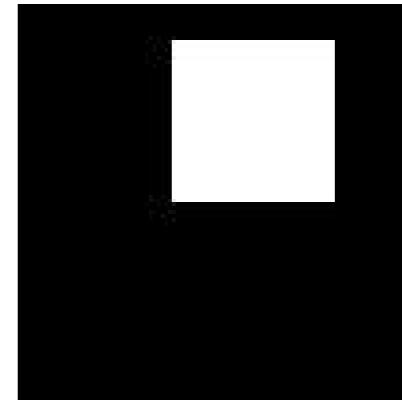
I\_toy



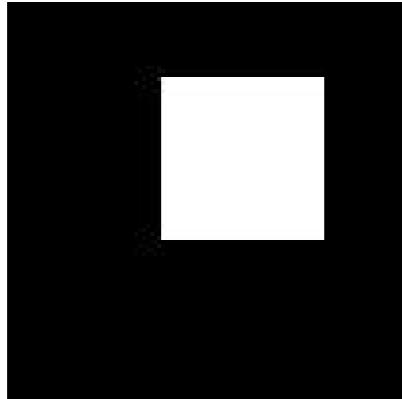
Segm\_toy(:,:,1)



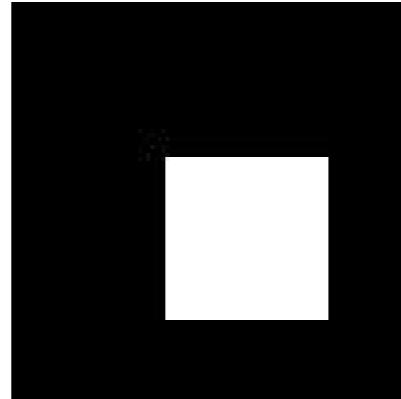
Segm\_toy(:,:,2)



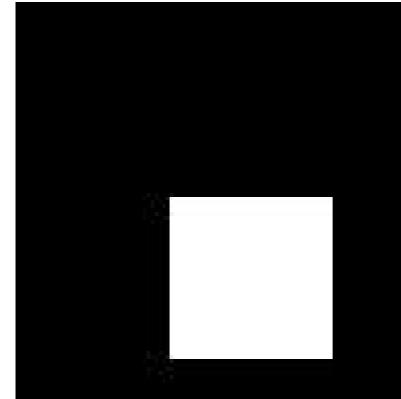
Segm\_toy(:,:,3)



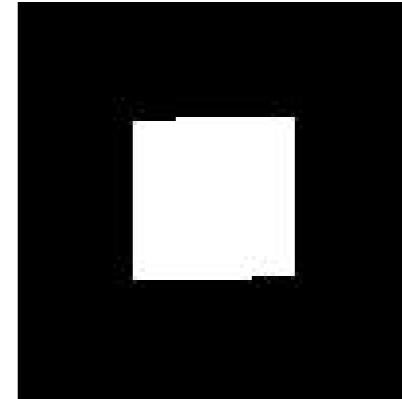
Segm\_toy(:,:,4)



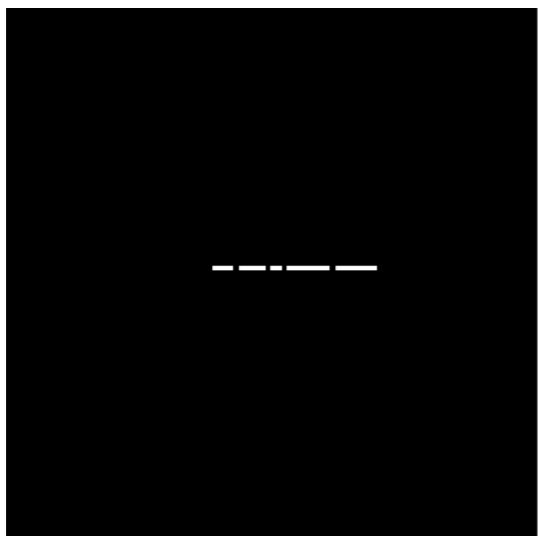
Segm\_toy(:,:,5)



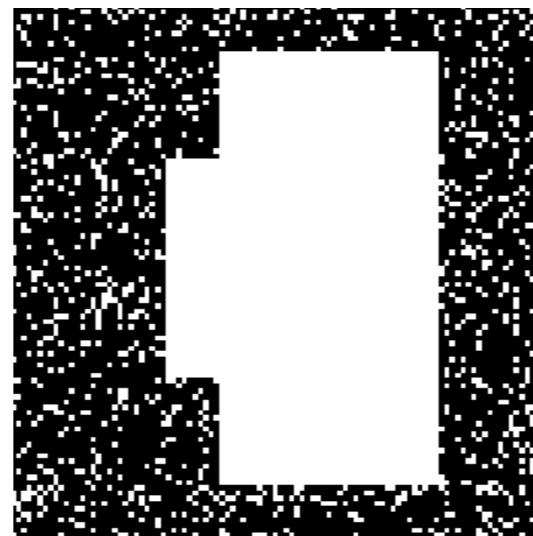
Segm\_toy(:,:,6)



Segm\_toy(:,:,7)



Intersection



Union



Vote majoritaire

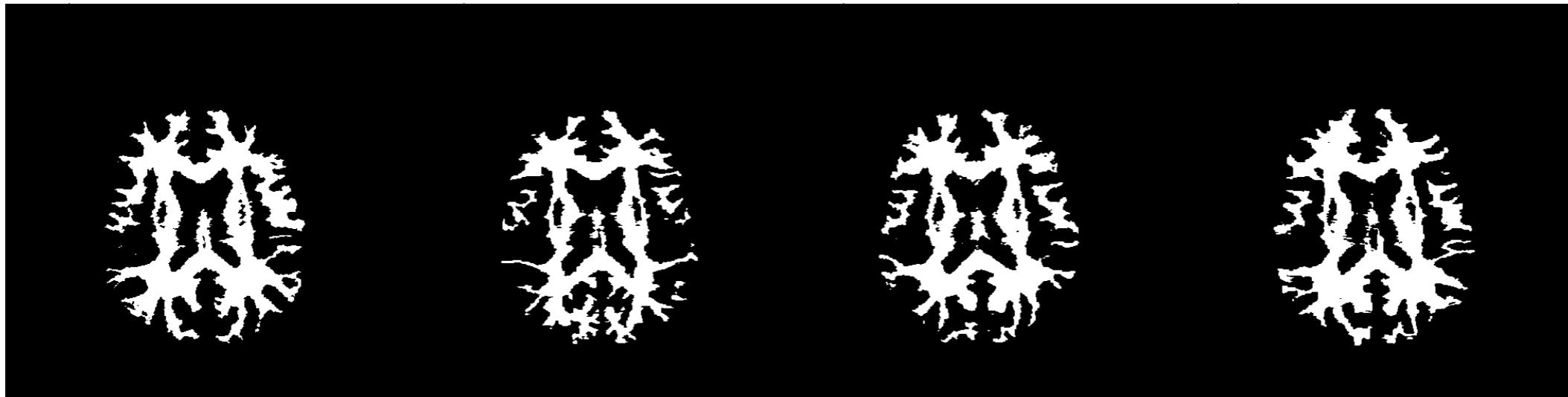
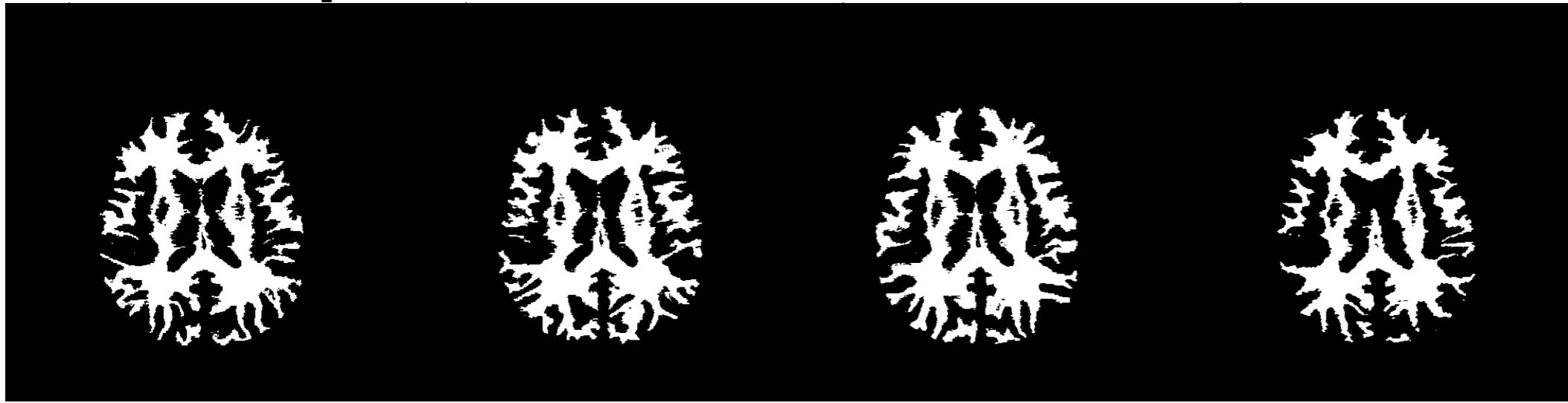
# Exemple cerveau binaire

- Voir données dans la démo.
- 15 segmentations différentes binaires

# Exemple cerveau binaire

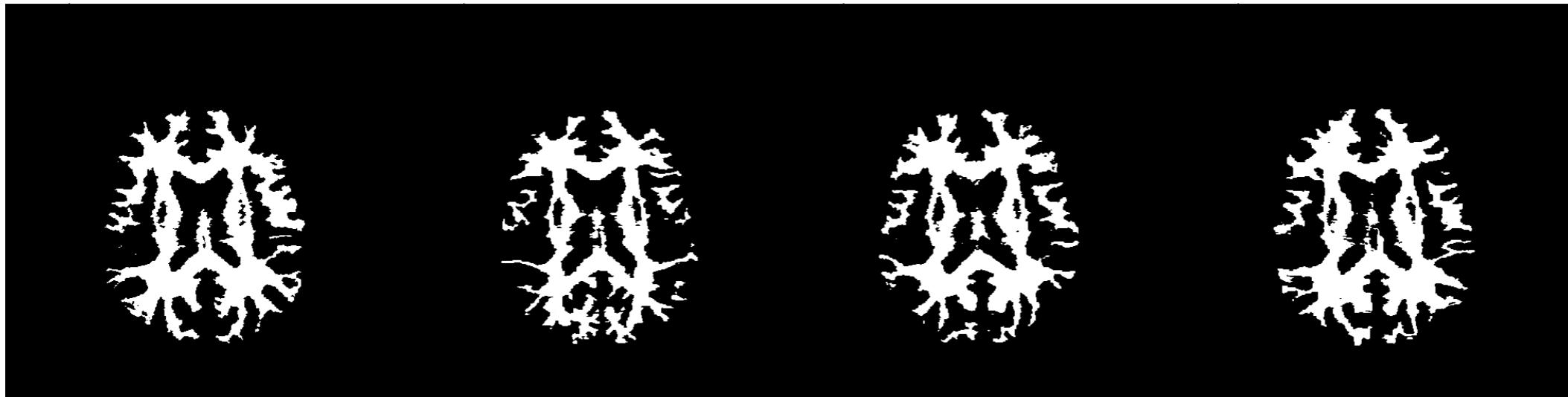
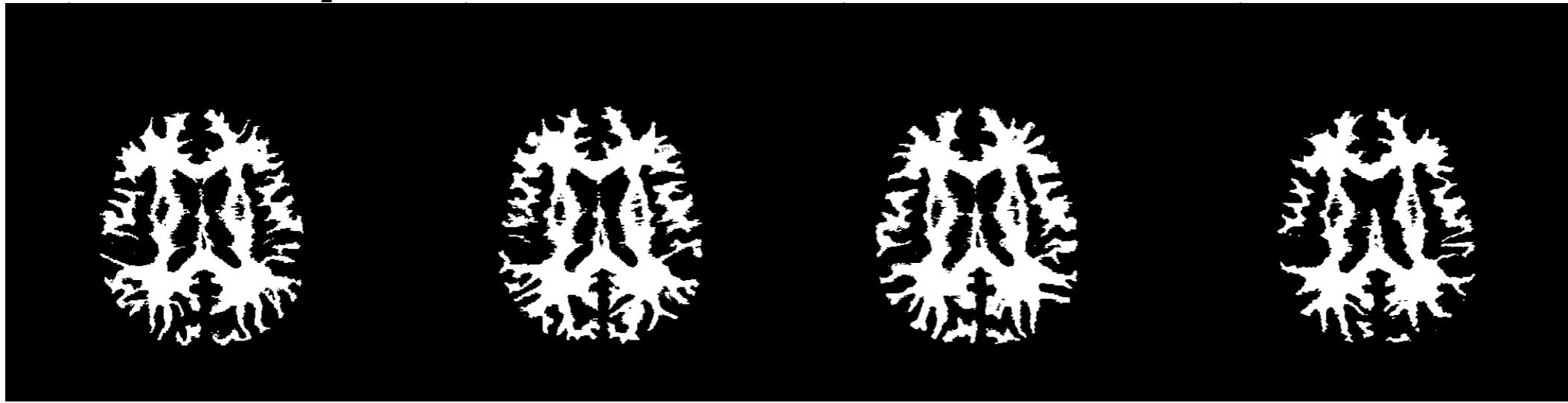


# Exemple cerveau binaire



Intersection

# Exemple cerveau binaire



Intersection



Union

# Exemple cerveau binaire



Intersection



Union



Vote majoritaire

# Exemple cerveau 4 classes

- Voir données dans la démo.
- 15 segmentations différentes 4 classes  
(e.g. résultats de FSL, Freesurfer, AFNI, SPM, ...)

# Exemple cerveau 4 classes

- Voir données dans la démo.
- 15 segmentations différentes 4 classes  
(e.g. résultats de FSL, Freesurfer, AFNI, SPM, ...)



Intersection

# Exemple cerveau 4 classes

- Voir données dans la démo.
- 15 segmentations différentes 4 classes  
(e.g. résultats de FSL, Freesurfer, AFNI, SPM, ...)



Intersection



Union

# Exemple cerveau 4 classes

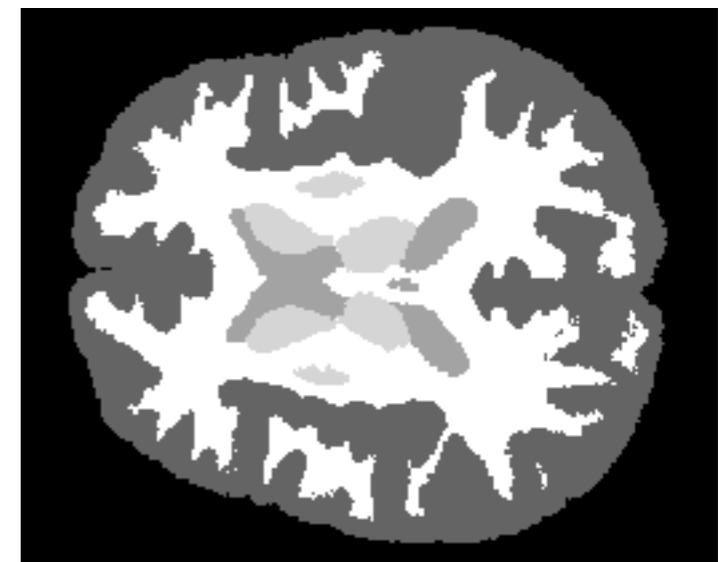
- Voir données dans la démo.
- 15 segmentations différentes 4 classes  
(e.g. résultats de FSL, Freesurfer, AFNI, SPM, ...)



Intersection



Union



Vote majoritaire

# Algorithme STAPLE

- Permet de simultanément combiner  $n$  segmentations et estimer le niveau de performance de chacune des segmentations initiales

# STAPLE

- Input:
  - Ensemble de  $N$  voxels étiquetés venant de  $R$  différents experts (ou algorithmes de segmentation)
  - Précision et fiabilité inconnues  
(on vient de voir que ça peut varier beaucoup)

# STAPLE

- Input:
  - Ensemble de  $N$  voxels étiquetés venant de  $R$  différents experts (ou algorithmes de segmentation)
  - Précision et fiabilité inconnues  
(on vient de voir que ça peut varier beaucoup)
- Reconstruit la vérité terrain la plus “probable” en utilisant un maximum de vraisemblance. Estimation de:
  - $\mathbf{T} = (T_1, \dots, T_N)$  la vrai segmentation venant des  $R$  segmentations  $D_{ij}$   
 $\mathbf{D}_{ij} = (D_{1,j}, \dots, D_{N,j})$
  - sensibilité  $\mathbf{p} = (p_1, \dots, p_R)$  et la spécificité  $\mathbf{q} = (q_1, \dots, q_R)$  pour chacune des  $R$  segmentation

# STAPLE

- Input:
  - Ensemble de  $N$  voxels étiquetés venant de  $R$  différents experts (ou algorithmes de segmentation)
  - Précision et fiabilité inconnues  
(on vient de voir que ça peut varier beaucoup)
- Reconstruit la vérité terrain la plus “probable” en utilisant un maximum de vraisemblance. Estimation de:
  - $\mathbf{T} = (T_1, \dots, T_N)$  la vrai segmentation venant des  $R$  segmentations  $D_{ij}$   
 $\mathbf{D}_{ij} = (D_{1,j}, \dots, D_{N,j})$
  - sensibilité  $\mathbf{p} = (p_1, \dots, p_R)$  et la spécificité  $\mathbf{q} = (q_1, \dots, q_R)$  pour chacune des  $R$  segmentation
- Comment reconstruire  $\mathbf{T}, \mathbf{p}, \mathbf{q}$ ?

# STAPLE

- Input:
  - Ensemble de  $N$  voxels étiquetés venant de  $R$  différents experts (ou algorithmes de segmentation)
  - Précision et fiabilité inconnues  
(on vient de voir que ça peut varier beaucoup)
- Reconstruit la vérité terrain la plus “probable” en utilisant un maximum de vraisemblance. Estimation de:
  - $\mathbf{T} = (T_1, \dots, T_N)$  la vrai segmentation venant des  $R$  segmentations  $D_{ij}$   
 $\mathbf{D}_{ij} = (D_{1,j}, \dots, D_{N,j})$
  - sensibilité  $\mathbf{p} = (p_1, \dots, p_R)$  et la spécificité  $\mathbf{q} = (q_1, \dots, q_R)$  pour chacune des  $R$  segmentation
- Comment reconstruire  $\mathbf{T}, \mathbf{p}, \mathbf{q}$ ?
- Algorithme de type EM (expectation-maximization)

# STAPLE

- Dans le cas binaire, la qualité de la segmentation  $j$  (ou de l'expert associé) sera mesuré via la sensibilité  $p_j$  et la spécificité  $q_j$

$$p_j = \Pr(D_{ij} = 1 | T_i = 1) \quad , \quad q_j = \Pr(D_{ij} = 0 | T_i = 0)$$

ou  $T_i$  est la segmentation “idéale”  
 $D_{ij}$  est la segmentation (0 ou 1 ici)  
donnée par l’expert  $j$  au pixel  $i$   
(e.g. si  $i = (x,y)$  on a  $D_{ij} = \text{Segm\_toy}(x,y,j)$ )

- On veut donc estimer  $T$ ,  $\mathbf{p}$  et  $\mathbf{q}$  simultanément

# STAPLE - Algorithme EM

1. On initialise  $p_j^{(0)}$  et  $q_j^{(0)}$ .
2. étape E : à l'étape k, on suppose qu'on connaît la sensibilité  $p_j^{(k)}$  et la spécificité  $q_j^{(k)}$  de chaque expert et on calcule pour chaque pixel  $i$  la probabilité qu'il fasse partie de la classe 1 que l'on note  $W_i^{(k)}$ .
3. étape M : on suppose qu'on connaît  $W_i^{(k)}$ , la probabilité qu'un pixel  $i$  fasse partie de la classe 1, et on met à jour la sensibilité  $p_j^{(k+1)}$  et la spécificité  $q_j^{(k+1)}$  de chaque expert.
4. On itère jusqu'à convergence. La segmentation finale est alors obtenue en affectant 1 aux pixels pour lesquel  $W \geq$  à un certain seuil

# STAPLE - Algorithme EM

- Étape E : prédiction

$$W_i^{(k)} = \frac{a_i^{(k)}}{a_i^{(k)} + b_i^{(k)}}$$

avec

$$a_i^{(k)} = \Pr(T_i = 1) \prod_{j|D_{ij}=1} p_j^{(k)} \prod_{j|D_{ij}=0} (1 - p_j^{(k)})$$

$$b_i^{(k)} = \Pr(T_i = 0) \prod_{j|D_{ij}=0} q_j^{(k)} \prod_{j|D_{ij}=1} (1 - q_j^{(k)})$$

ou  $\Pr(T_i = 1)$  et  $\Pr(T_i = 0)$  sont les probabilités *a priori* d'avoir l'une ou l'autre des classes. Estimées en calculant la proportion de 1 et 0 dans les segmentations initiales.

# STAPLE - Algorithme EM

- Étape M : maximisation, consiste à estimer la sensibilité et la spécificité de chaque expert à partir des probabilités  $W_i^{(k)}$

$$p_j^{(k+1)} = \frac{\sum_{i|D_{ij}=1} W_i^{(k)}}{\sum_i W_i^{(k)}} \quad \text{et} \quad q_j^{(k+1)} = \frac{\sum_{i|D_{ij}=0} (1 - W_i^{(k)})}{\sum_i (1 - W_i^{(k)})}$$

# STAPLE

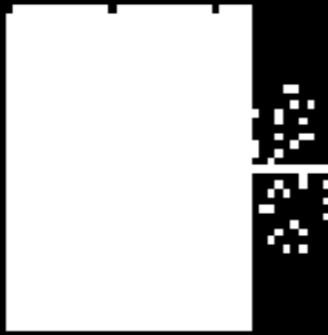
- Détail d'implémentation
  - Voir p. 426-427 - Toennies et al MedIA 2012
  - Warfield et al. 2004

# STAPLE

- Détail d'implémentation
  - Voir p. 426-427 - Toennies et al MedIA 2012
  - Warfield et al. 2004
- Démo

# STAPLE

- Détail d'implémentation
  - Voir p. 426-427 - Toennies et al MedIA 2012
  - Warfield et al. 2004
- Démo



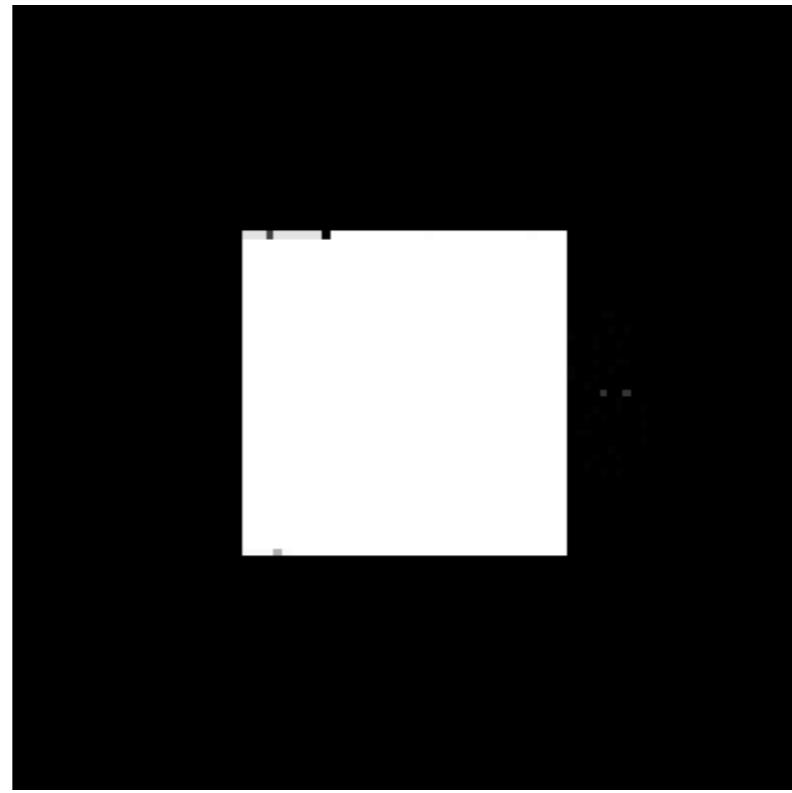
Vote majoritaire

# STAPLE

- Détail d'implémentation
  - Voir p. 426-427 - Toennies et al MedIA 2012
  - Warfield et al. 2004
- Démo



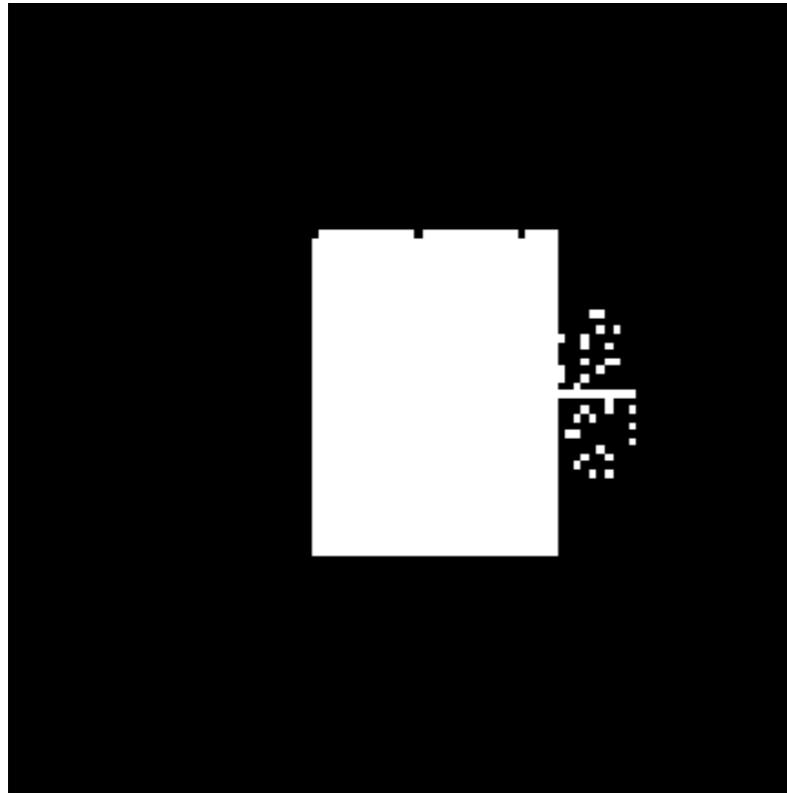
Vote majoritaire



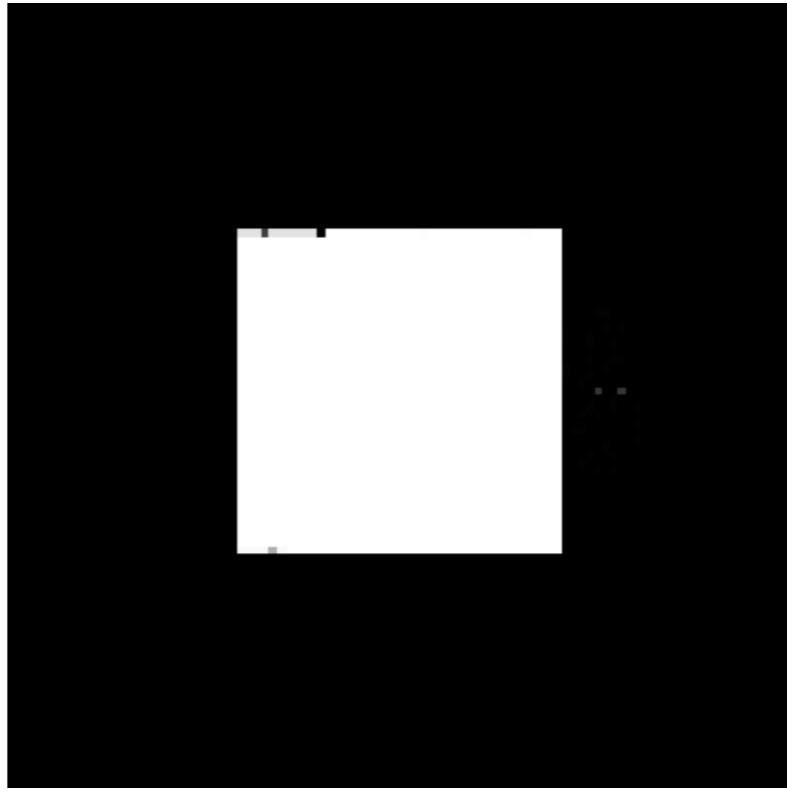
STAPLE - W

# STAPLE

- Détail d'implémentation
  - Voir p. 426-427 - Toennies et al MedIA 2012
  - Warfield et al. 2004
- Démo



Vote majoritaire



STAPLE - W



STAPLE -  $W > 0.5$

# Exemple de courbe ROC

- On souhaite segmenter  $I_{toy}$  par seuillage

# Exemple de courbe ROC

- On souhaite segmenter  $I_{toy}$  par seuillage
- On décide de choisir le meilleur résultat en terme de sensibilité/spécificité

# Exemple de courbe ROC

- On souhaite segmenter  $I_{toy}$  par seuillage
- On décide de choisir le meilleur résultat en terme de sensibilité/spécificité
- Pour cela, on trace la courbe ROC

# Exemple de courbe ROC

- On souhaite segmenter  $I_{toy}$  par seuillage
- On décide de choisir le meilleur résultat en terme de sensibilité/spécificité
- Pour cela, on trace la courbe ROC
- Comment on trouver le meilleur seuil?

# Exemple de courbe ROC

- On souhaite segmenter  $I_{toy}$  par seuillage
- On décide de choisir le meilleur résultat en terme de sensibilité/spécificité
- Pour cela, on trace la courbe ROC
- Comment on trouver le meilleur seuil?
- On cherche une sensibilité = 1 et une spécificité = 1 ( $1 - \text{spécificité} = 0$ )

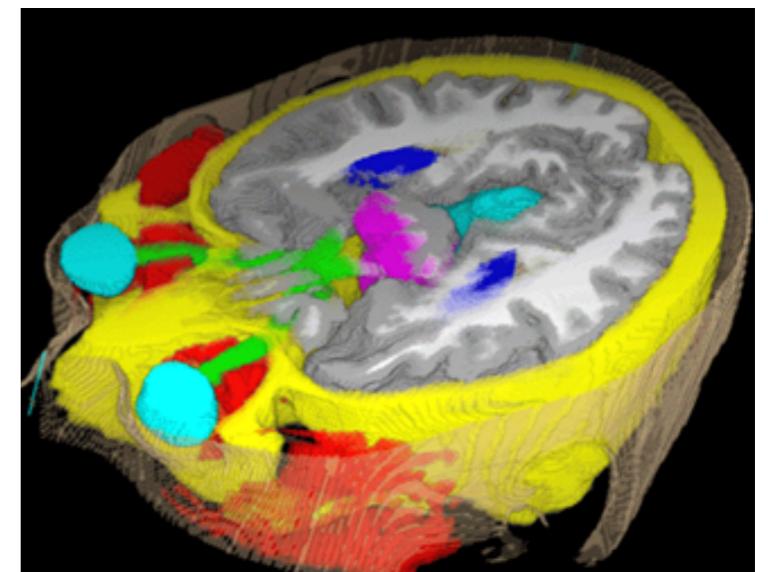
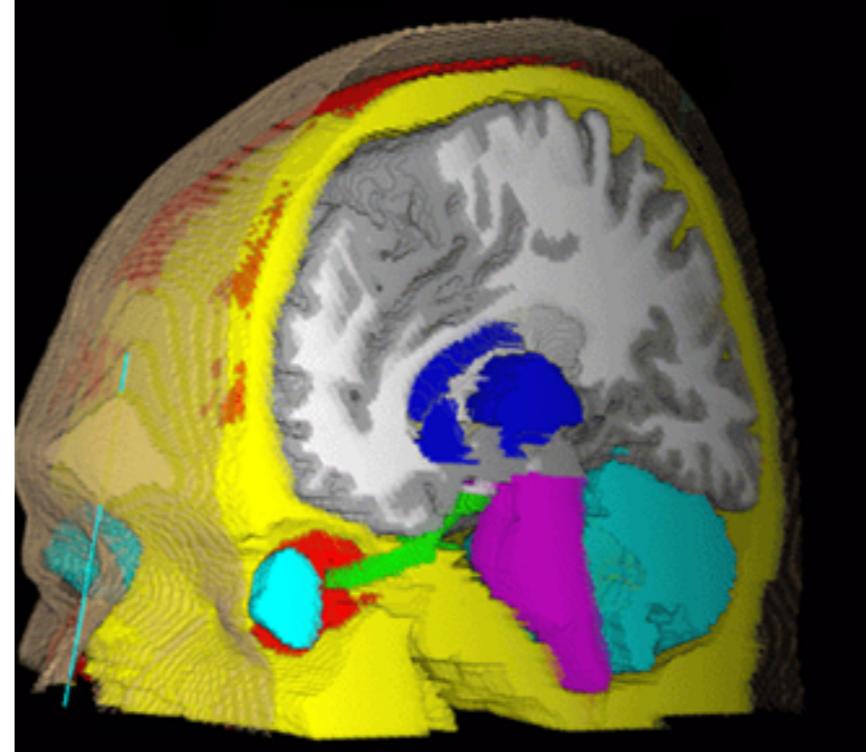
# Exemple de courbe ROC

- On souhaite segmenter  $I_{toy}$  par seuillage
- On décide de choisir le meilleur résultat en terme de sensibilité/spécificité
- Pour cela, on trace la courbe ROC
- Comment on trouver le meilleur seuil?
- On cherche une sensibilité = 1 et une spécificité = 1 ( $1 - \text{spécificité} = 0$ )
- Démo

# Segmentation basé sur des atlas

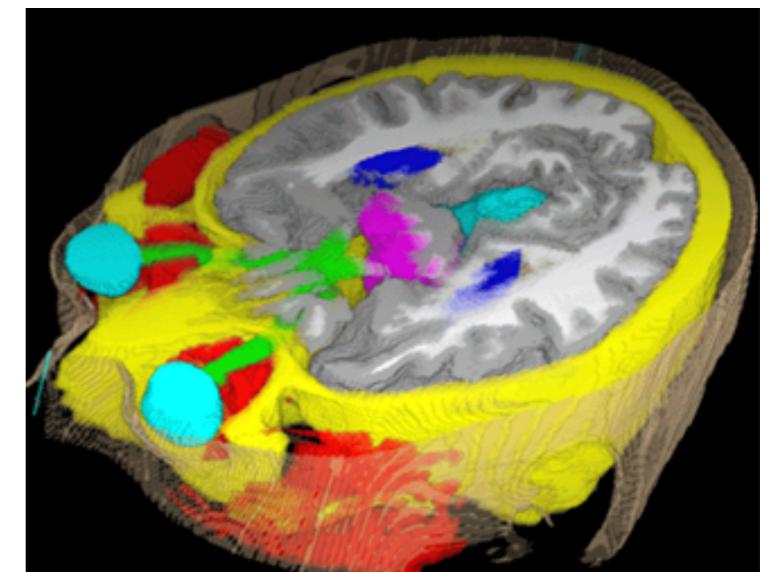
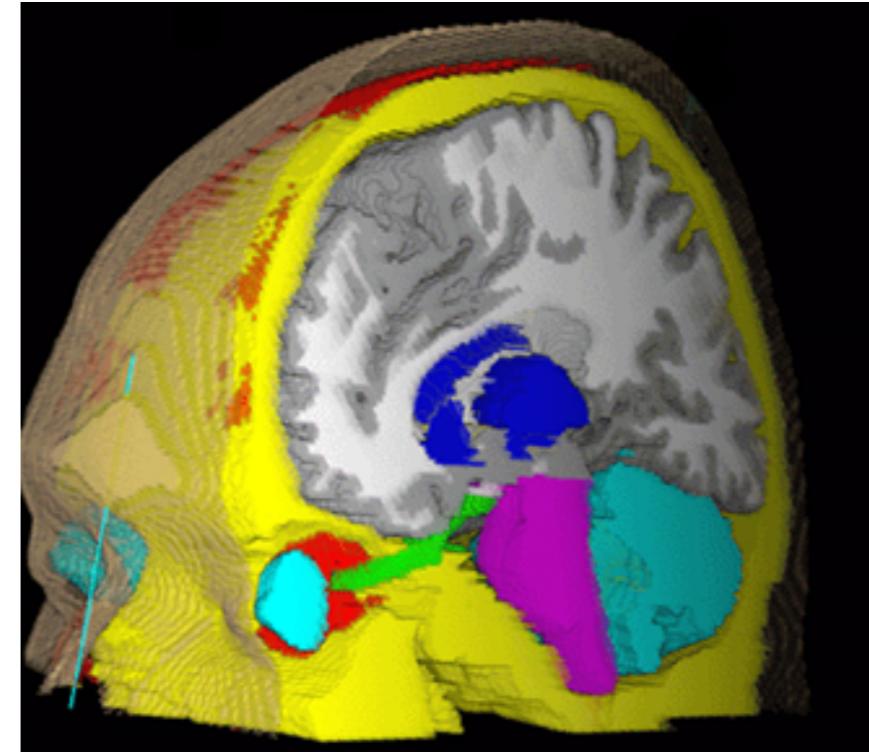
# Segmentation basée sur les atlas

- Atlas anatomique
  - Image d'une anatomie moyenne
  - Segmentation fait par un expert



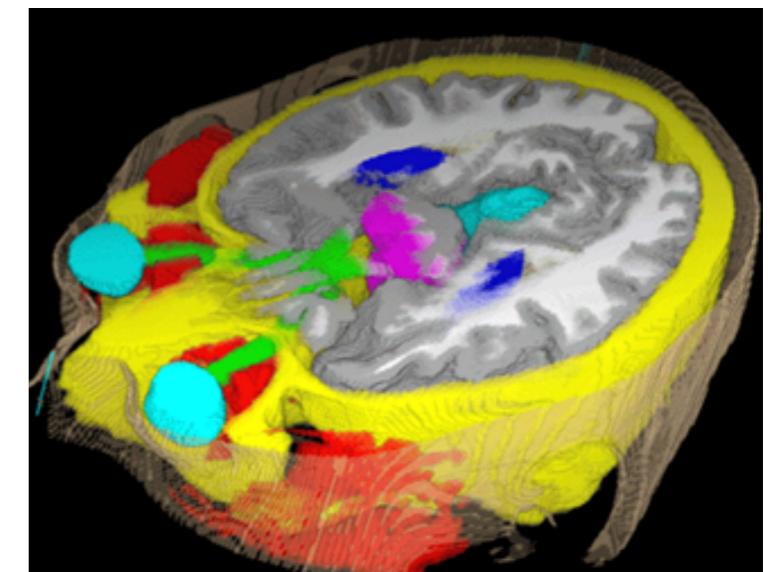
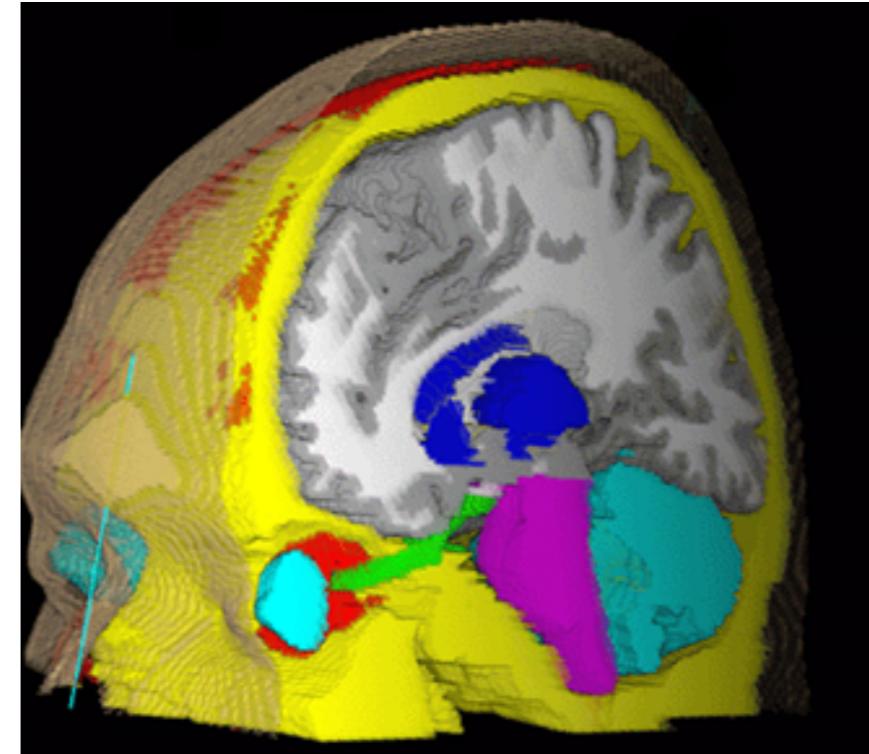
# Segmentation basée sur les atlas

- Atlas anatomique
  - Image d'une anatomie moyenne
  - Segmentation fait par un expert
- Objectif
  - Utilisation de l'atlas pour segmenter une nouvelle image (patient)



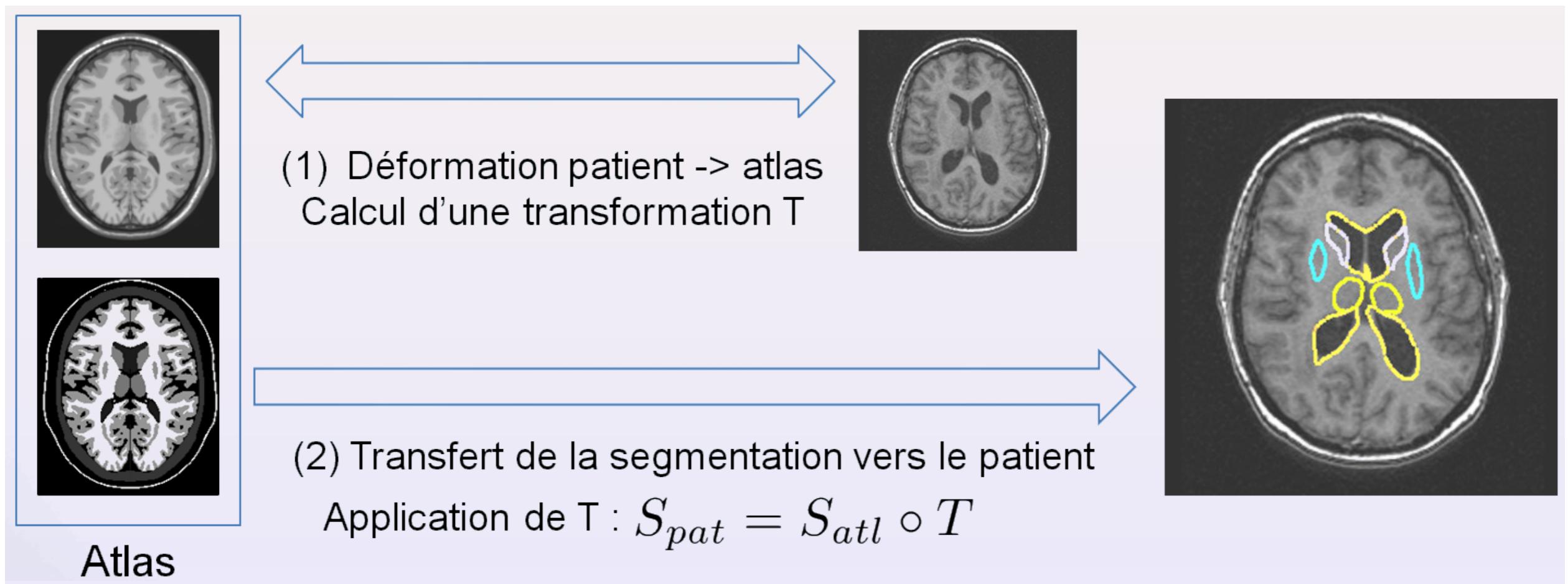
# Segmentation basée sur les atlas

- Atlas anatomique
  - Image d'une anatomie moyenne
  - Segmentation fait par un expert
- Objectif
  - Utilisation de l'atlas pour segmenter une nouvelle image (patient)
- Avantage
  - Prise en compte d'*a priori* spécifiques au résultat recherché
  - Segmentation de multiples structures en une fois
  - Possibilité d'estimer des structures peu visible



# Segmentation par atlas

- On transforme le problème de segmentation en un problème de **recalage**



- Il faut donc de bons algorithmes de recalage (à venir...)