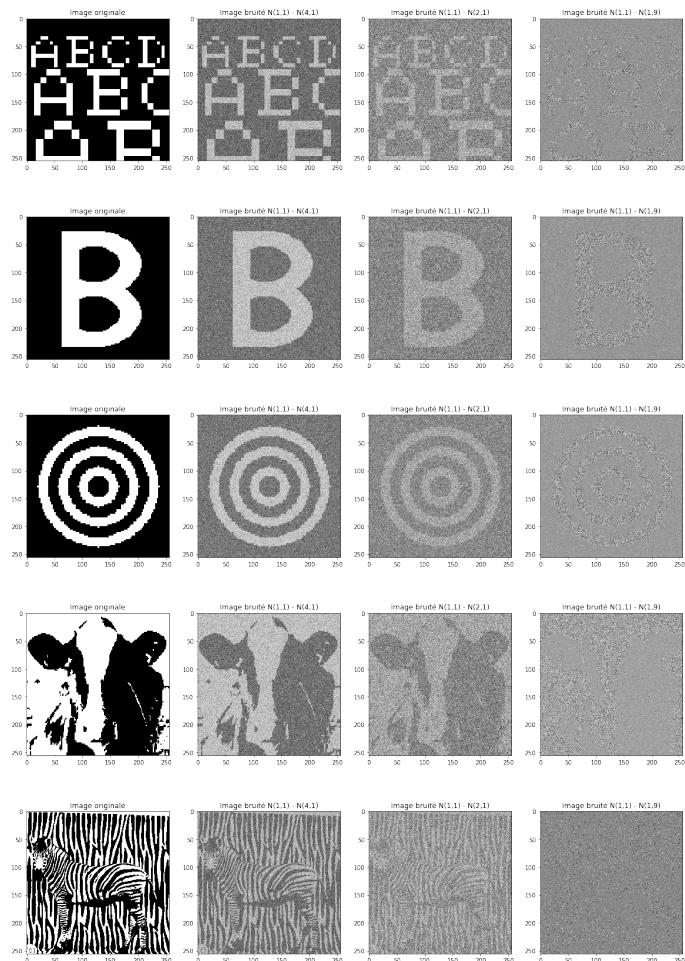


SEGMENTATION D'IMAGE

# Compte rendu : TP Segmentation d'image



Nicolas DUFOUR

Automne 2019

## Table des matières

<b>1</b>	<b>Mise en place des fonctions de bases</b>	<b>2</b>
<b>2</b>	<b>Segmentation basique par Kmeans</b>	<b>5</b>
<b>3</b>	<b>Segmentation MPM aveugle</b>	<b>6</b>
<b>4</b>	<b>Segmentation non supervisée aveugle</b>	<b>7</b>
<b>5</b>	<b>Segmentation par champs de Markov cachés supervisée</b>	<b>12</b>
<b>6</b>	<b>Segmentation par champs de Markov cachés non-supervisée</b>	<b>14</b>
<b>7</b>	<b>Segmentation réelle</b>	<b>15</b>

## Table des figures

<b>1</b>	<b>Code pour bruire notre image</b>	<b>2</b>
<b>2</b>	<b>Différentes images bruitées par des paramètres de bruits différents.</b>	<b>2</b>
<b>3</b>	<b>2 gaussiennes de paramètres <math>\mathcal{N}(1, 1)</math> et <math>\mathcal{N}(4, 1)</math></b>	<b>3</b>
<b>4</b>	<b>2 gaussiennes de paramètres <math>\mathcal{N}(1, 1)</math> et <math>\mathcal{N}(2, 1)</math></b>	<b>3</b>
<b>5</b>	<b>2 gaussiennes de paramètres <math>\mathcal{N}(1, 1)</math> et <math>\mathcal{N}(1, 9)</math></b>	<b>4</b>
<b>6</b>	<b>Segmentation par Kmeans de nos images bruité</b>	<b>5</b>
<b>7</b>	<b>Segmentation par MPM de nos images bruité</b>	<b>6</b>
<b>8</b>	<b>2 gaussiennes de paramètres <math>\mathcal{N}(1, 1)</math> et <math>\mathcal{N}(1, 9)</math></b>	<b>7</b>
<b>9</b>	<b>Convergences des différentes valeurs avec l'algorithme EM avec leur taux moyen d'erreur MPM aveugle associé</b>	<b>8</b>
<b>10</b>	<b>MPM non supervisée et EM avec initialisation naïve</b>	<b>9</b>
<b>11</b>	<b>MPM non supervisée et EM avec initialisation KMeans</b>	<b>9</b>
<b>12</b>	<b>Segmentation par MPM non supervisée par EM-Kmeans de nos images bruité</b>	<b>10</b>
<b>13</b>	<b>MPM non supervisée et EM avec initialisation KMeans</b>	<b>11</b>
<b>14</b>	<b>Segmentation par MPM non supervisée par SEM-Kmeans de nos images bruité</b>	<b>11</b>
<b>15</b>	<b>Generation de champs de Markov et segmentation par champs de markov cachée</b>	<b>12</b>
<b>16</b>	<b>Segmentation par champs de markov d'image réelles bruité supervisée</b>	<b>13</b>
<b>17</b>	<b>Segmentation par champs de markov d'image réelles bruité non supervisée</b>	<b>14</b>
<b>18</b>	<b>Segmentation d'une image satellite de l'europe</b>	<b>15</b>
<b>19</b>	<b>Segmentation d'une image d'une fleur de lotus</b>	<b>15</b>
<b>20</b>	<b>Segmentation d'une image satellite de fusée</b>	<b>15</b>

## Liste des tableaux

<b>1</b>	<b>Tableau récapitulatif des taux d'erreurs moyens sur 100 itérations</b>	<b>5</b>
<b>2</b>	<b>Tableau récapitulatifs des taux d'erreurs moyens sur 200 itérations</b>	<b>6</b>
<b>3</b>	<b>Tableau récapitulatifs des taux d'erreurs moyens sur 200 itérations</b>	<b>10</b>
<b>4</b>	<b>Tableau récapitulatifs des taux d'erreurs moyens sur 200 itérations</b>	<b>12</b>
<b>5</b>	<b>Tableau récapitulatifs des taux d'erreurs moyens sur 200 itérations</b>	<b>13</b>
<b>6</b>	<b>Tableau récapitulatifs des taux d'erreurs moyens sur 1 itérations</b>	<b>14</b>

# 1 Mise en place des fonctions de bases

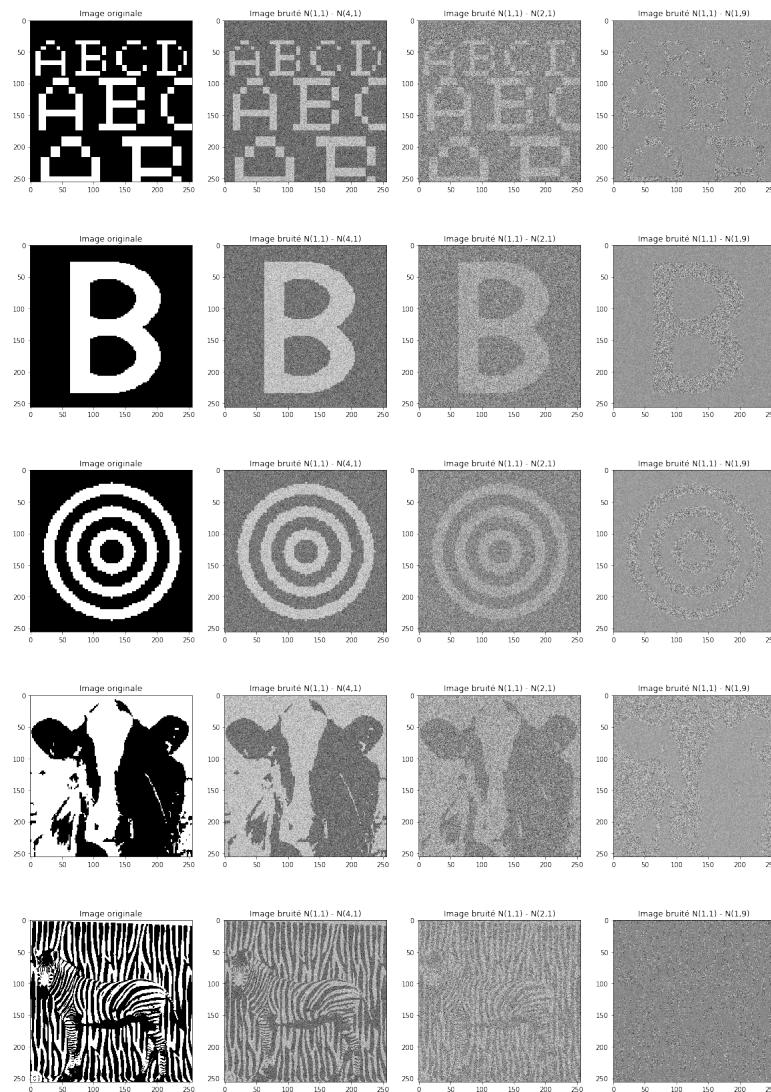
Pour cette section, j'ai du revenir en arrière car arrivé à la segmentation aveugle, pour 200 échantillons, mon code a mis 10h à tourner. Étant donné que c'est la segmentation la plus simple en terme de complexité, cela m'a inquiété pour la suite. Ainsi, en vectorisant mes fonctions avec numpy (à la place d'itérer sur l'image auparavant) j'ai pu réduire le temps de la segmentation aveugle sur 200 échantillons à 30 secondes, ce qui m'aidera considérablement par la suite. L'amélioration la plus importante est celle de la fonction de bruitage, `bruit_gauss`, donné ci-dessous :

```
def bruit_gauss(X,m,n,c11 ,c12 ,m1,sig1 ,m2,sig2):
    gauss_1=np.random.normal(m1,sig1 ,(m,n))
    gauss_2=np.random.normal(m2,sig2 ,(m,n))
    return(np.where(X==c11 ,gauss_1 ,gauss_2))
```

**FIGURE 1 – Code pour bruiter notre image**

Ainsi, en tirant directement toutes les gaussiennes possibles dans 2 matrices, puis en les choisissant vectoriellement, on a une accélération considérable même si au final, nous faisons 2 fois plus de tirages.

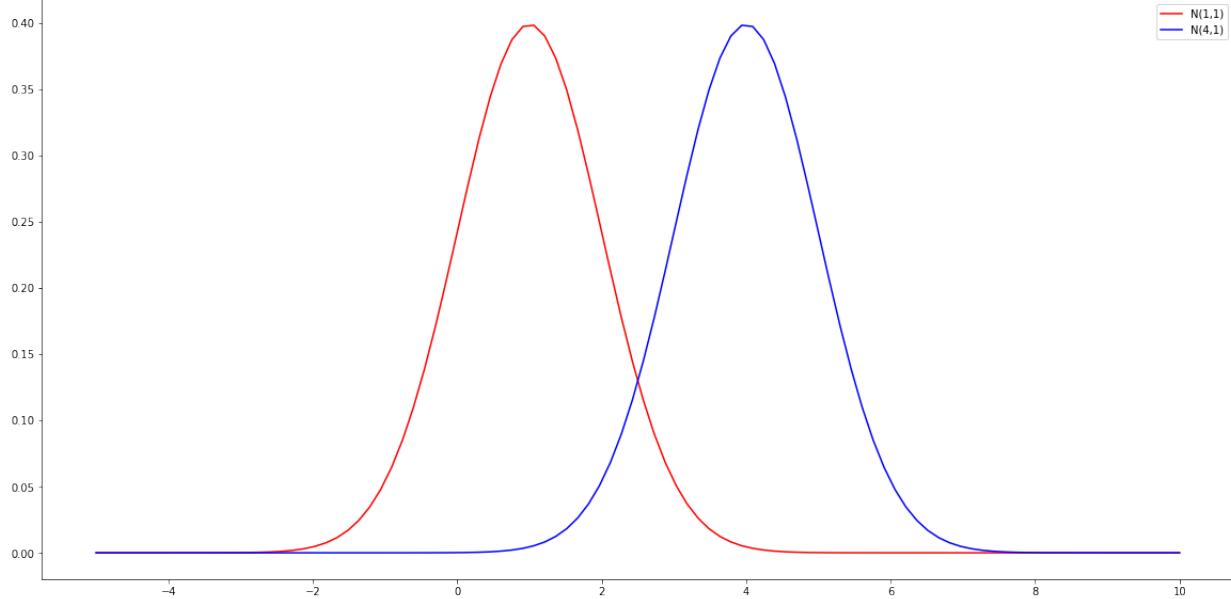
Ainsi, nous pouvons faire nos bruitages pour les différents bruits et pour 5 images. J'ai choisi 3 images avec des formes assez simples et 2 avec des formes plus complexes. On trouve le résultat suivant :



**FIGURE 2 – Différentes images bruitées par des paramètres de bruits différents.**

On se rend compte que plusieurs facteurs entre en compte vis-à-vis du bruit.

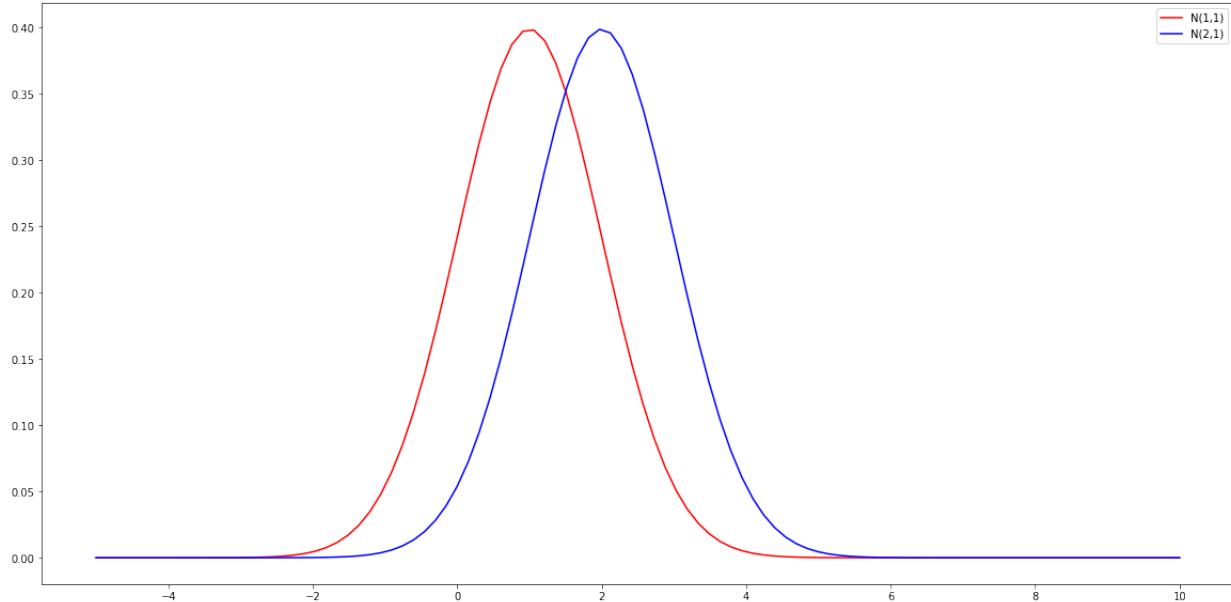
On se rend compte dans un premier temps que lorsque les 2 moyennes des gaussiennes sont suffisamment éloignées comme dans le premier cas  $\mathcal{N}(1, 1) \mid \mathcal{N}(4, 1)$ , on constate une différence visuelle importante. On différencie bien les 2 classes à l'il nu. Cela se comprend en traçant les distributions des 2 gaussiennes :



**FIGURE 3 – 2 gaussiennes de paramètres  $\mathcal{N}(1, 1)$  et  $\mathcal{N}(4, 1)$**

On voit donc bien que nos 2 gaussiennes sont assez éloignées. Ce qui explique que le bruit n'empêche pas la distinction des 2 classes visuellement.

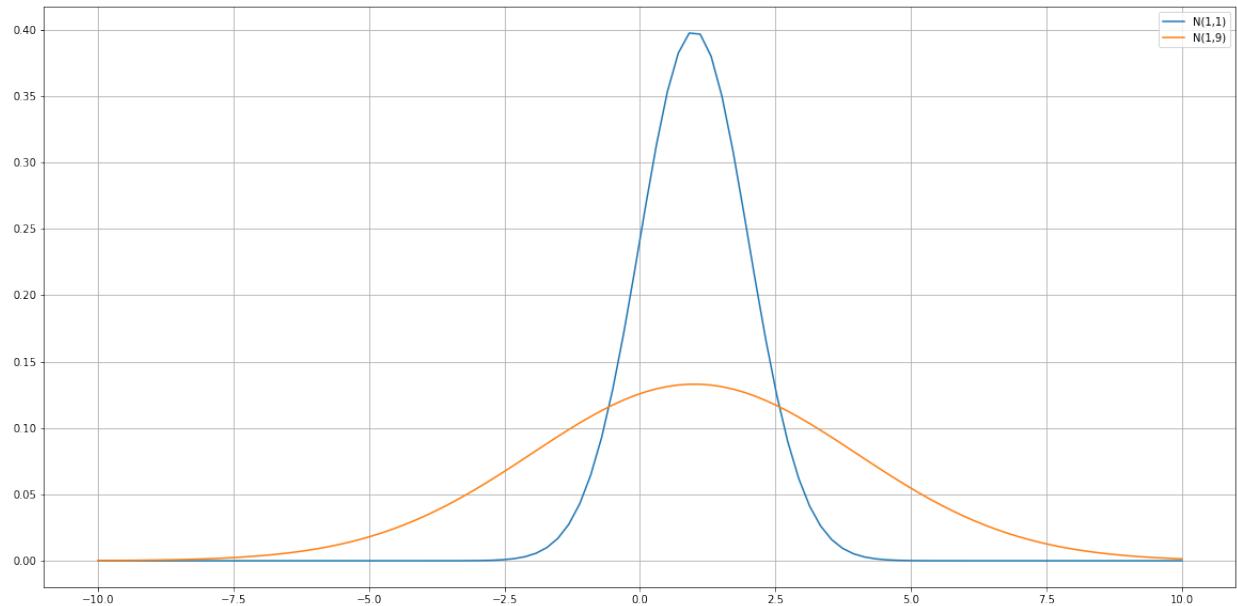
Pour le 2ème cas,  $\mathcal{N}(1, 1) \mid \mathcal{N}(2, 1)$ , on se rend compte que cela devient très compliqué de distinguer bruit de signal. Les moyennes sont trop proches, trop superposées, comme on peut le voir sur la figure suivante :



**FIGURE 4 – 2 gaussiennes de paramètres  $\mathcal{N}(1, 1)$  et  $\mathcal{N}(2, 1)$**

Cette difficulté due au bruitage se voit particulièrement sur les formes plus compliquées comme le zèbre ou la vache.

Pour finir, on voit quand les moyennes sont superposées, on peut quand même différencier les formes car les variances sont différentes, même si on a du mal à différencier les classes pour des formes plus complexes. C'est le 3ème cas  $\mathcal{N}(1, 1) \mid \mathcal{N}(1, 9)$ . On comprend mieux en regardant le tracé des gaussiennes :

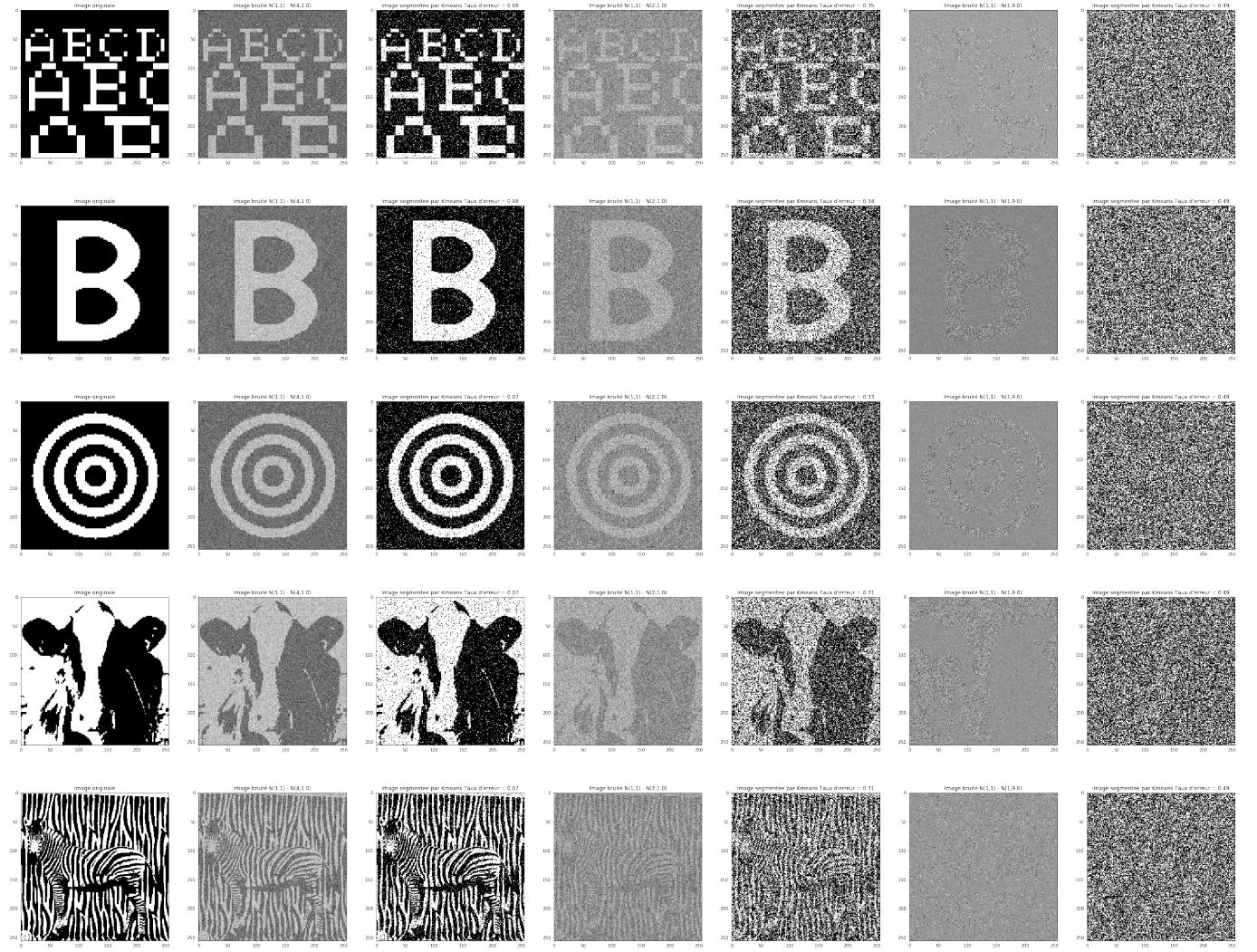


**FIGURE 5 – 2 gaussiennes de paramètres  $\mathcal{N}(1,1)$  et  $N(1,9)$**

Comme on peut le voir, comme la variance de la 2ème classe est très grande, au final les zones où l'on a des distributions très proches sont relativement réduites.

## 2 Segmentation basique par Kmeans

Avec le Kmeans le principe est de distribuer les pixels de l'image bruitée entre 2 classes. Les 2 classes ne sont pas nécessairement nos classes cl1,cl2. Kmeans groupe chaque pixel par proximité dans l'espace des nuances de gris. Les classes sont définis par les positions moyennes des pixels de cette classe. Ensuite, on pourra faire une bijection entre les classes et cl1,cl2. La bijection utiliser ici est d'associer  $\min(cl1, cl2)$  à la plus petite des 2 classes du Kmeans et  $\max(cl1, cl2)$  à la plus grande des classes. On effectue les classifications sur nos 5 images et nos 3 niveaux de bruit 100 fois et on obtient le tableau suivant avec les erreurs associées :



**FIGURE 6 – Segmentation par Kmeans de nos images bruité**

Image \ Distribution du bruit	$\mathcal{N}(1,1), \mathcal{N}(4,1)$	$\mathcal{N}(1,1), \mathcal{N}(2,1)$	$\mathcal{N}(1,1), \mathcal{N}(1,9)$
abcd	0,08	0,35	0,49
B	0,08	0,34	0,49
Cible	0,07	0,33	0,49
Vache	0,07	0,31	0,49
Zebre	0,07	0,31	0,49

**TABLE 1 – Tableau récapitulatif des taux d'erreurs moyens sur 100 itérations**

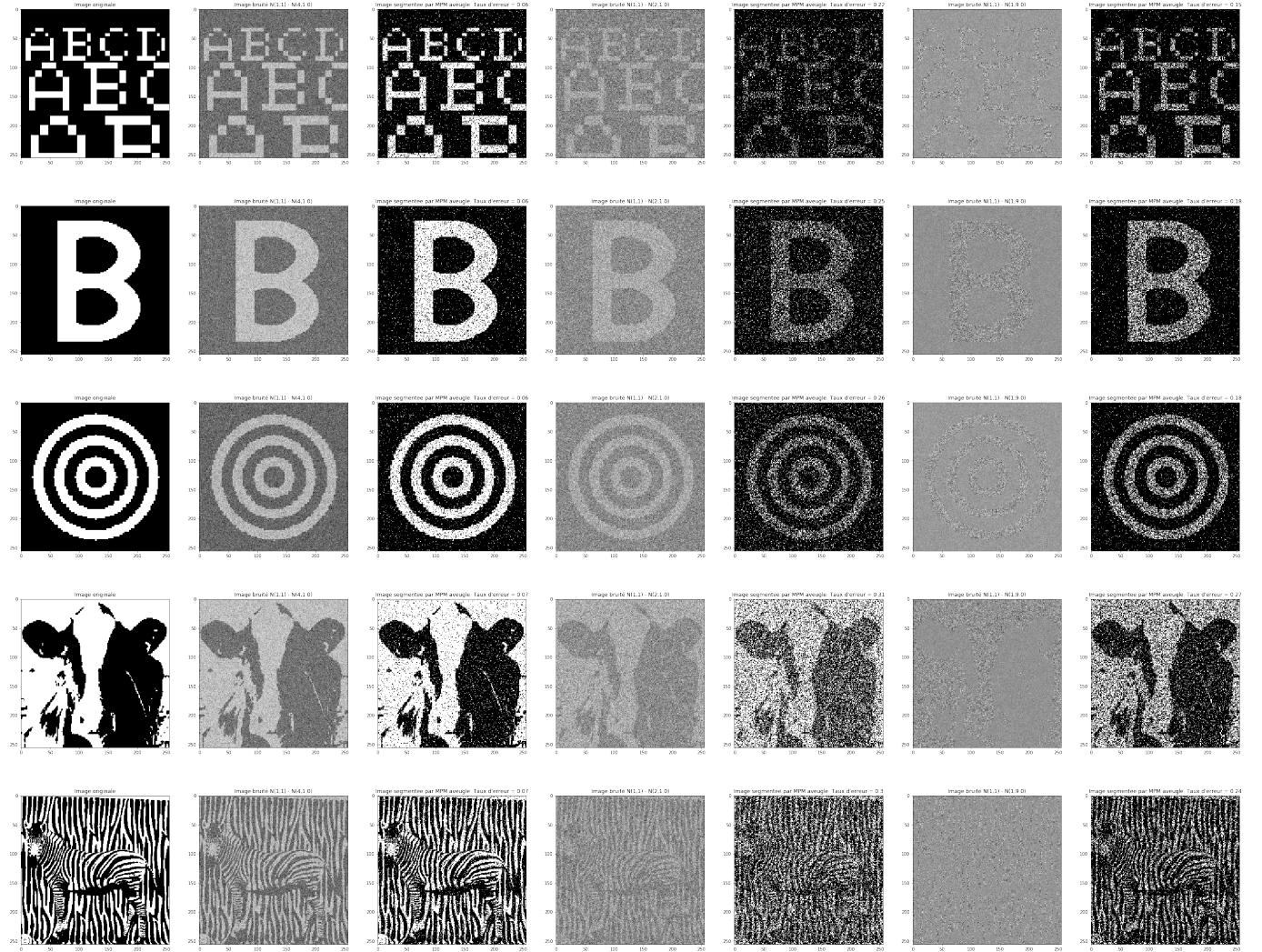
On peut observer que Kmeans arrive à des résultats corrects pour des situation de bruits optimales comme dans le premier cas de bruit. Cependant on observe bien qu'il segmente mal les zones bruitées de moyennes proche comme dans le 2ème cas de bruit. On voit que comme Kmeans ne prend pas en compte le coté stochastique de l'image, la segmentation est très bruitée. On voit aussi apparaître un autre problème dans

le cas 3 où les classes se voit inverser. Comme Kmeans ne prend pas en compte les connaissances sur nos niveaux de bruit, il ne sait rien sur les classes et on se retrouve avec une inversion de classes car les centroïdes ne sont pas directement définis par des connaissances sur nos classes.

Quand on voit dans le 3ème cas que en moyenne le taux d'erreur est proche de 50%, cela veut dire que l'on fait presque comme si on segmentait chaque pixel aléatoirement par une Bernoulli équidistribuée. Ainsi, Kmeans n'est pas une approche satisfaisante pour segmenter ce genre d'images

### 3 Segmentation MPM aveugle

Ici, nous allons effectuer une segmentation MPM en supposant l'indépendance entre les pixels. On se retrouve avec les segmentations suivantes. Les taux d'erreurs moyens sont calculés sur 200 segmentation par image.

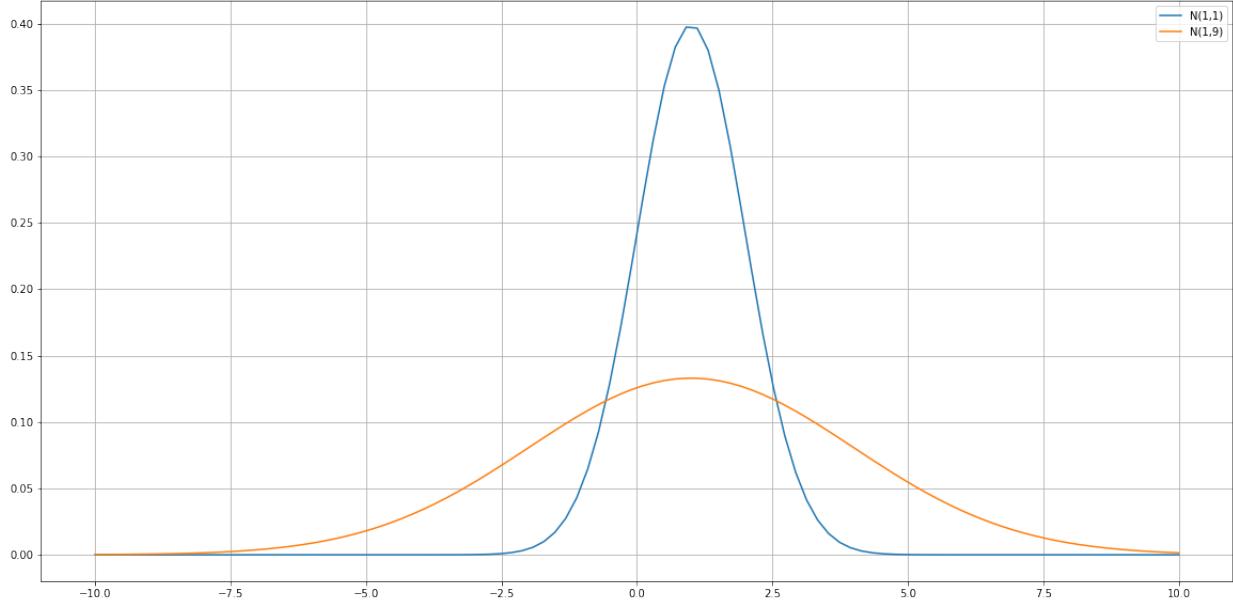


**FIGURE 7 – Segmentation par MPM de nos images bruitées**

Image \ Distribution du bruit	$\mathcal{N}(1,1), \mathcal{N}(4,1)$	$\mathcal{N}(1,1), \mathcal{N}(2,1)$	$\mathcal{N}(1,1), \mathcal{N}(1,9)$
abcd	0,06	0,22	0,15
B	0,06	0,25	0,18
Cible	0,06	0,26	0,18
Vache	0,07	0,31	0,27
Zebre	0,07	0,3	0,24

**TABLE 2 – Tableau récapitulatifs des taux d'erreurs moyens sur 200 itérations**

On n'observe pas de différence notable par la segmentation MPM par rapport à Kmeans pour les 2 premiers niveaux de bruit. Par contre, pour le dernier, la différence est considérable. Auparavant par Kmeans, comme les gaussiennes sont centré autour de la même moyenne, Kmeans ne voyait pas vraiment de différence entre les pixels. Avec l'introduction des connaissance sur les distributions, on peut voir que MPM arrive à différencier les pixels. Pour mieux comprendre, prenons 2 variable aléatoire suivant  $X_1 \sim \mathcal{N}(1, 1)$  et  $X_2 \sim \mathcal{N}(1, 9)$ . Pour la simplicité de compréhension, supposons que les lois à priori valent 0.5. On à ci dessous nos 2 lois :

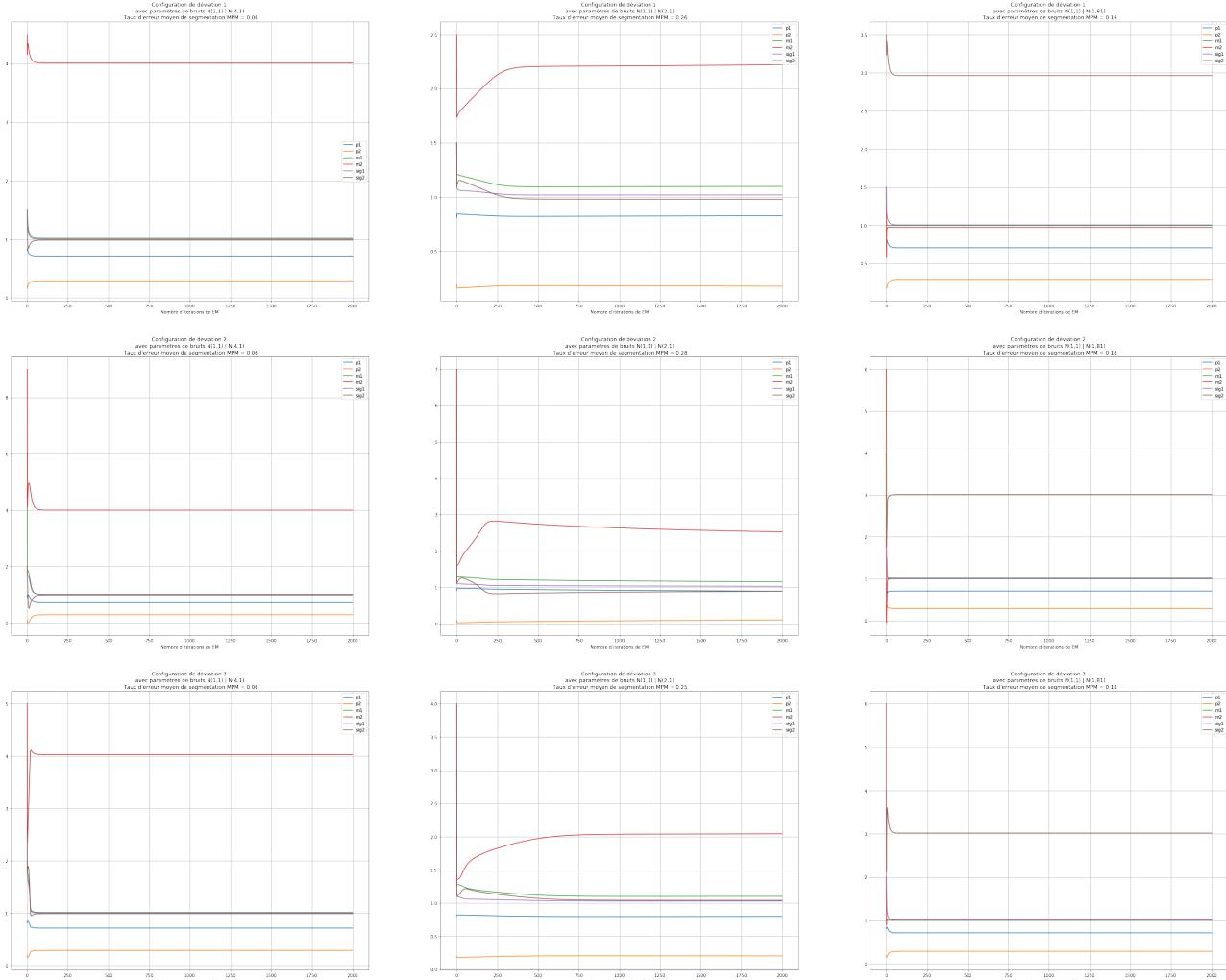


**FIGURE 8 – 2 gaussiennes de paramètres  $\mathcal{N}(1, 1)$  et  $\mathcal{N}(1, 9)$**

Ainsi, on voit que pour la plupart de la distribution de notre pixel noir (courbe rouge), il est prédominant par rapport à la distribution du pixel blanc. Cependant, comme le pixel blanc à une grande variance, la distribution n'est pas très importante sur cette zone et le pixel blanc arrive à être prédominant sur une grande partie de la distribution. Ce non empiétement sur l'autre distribution est responsable d'une meilleure classification. En plus MPM sait quelle distribution correspond à quelle classe contrairement à Kmeans qui trouve des centroïdes qui ne permettent pas toujours de trouver les bonnes classes. C'est en connaissant les lois à posteriori que le MPM peut être meilleur que le Kmeans

## 4 Segmentation non supervisée aveugle

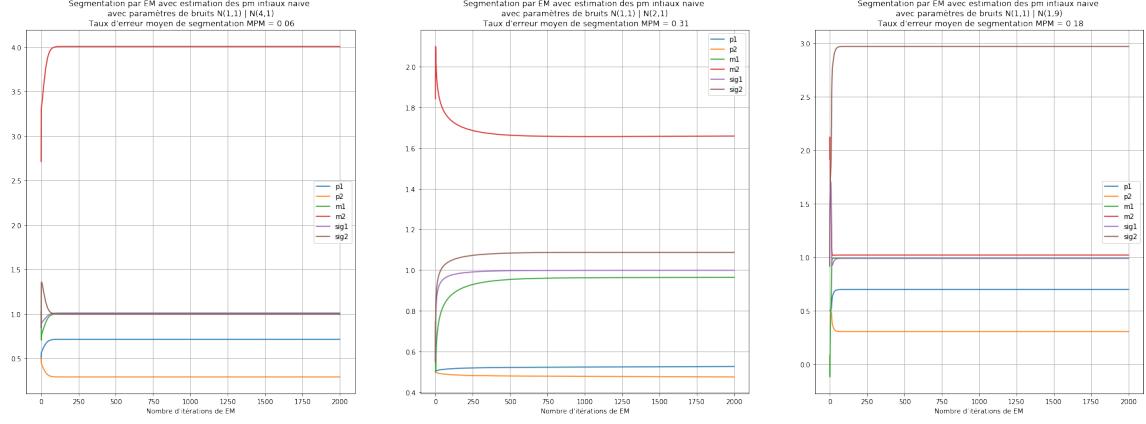
Le but de cette partie est d'estimer les paramètres des lois à priori et à posteriori à partir de l'image observée. Nous allons dans un premier temps implémenter une approche EM (Expectation - Maximisation). Nous allons dans un premier temps étudier l'algorithme en nous basant sur l'image du B majuscule. Sur un algorithme EM sur 2000 itérations, nous étudions pour nos 3 profils de bruit utilisés auparavant avec des déviations par rapport au valeurs initiales ( $p_1, p_2, m_1, m_2, sig_1, sig_2$ ). Nos 3 profils de déviations sont :  $(p_1^0, p_2^0, m_1^0, m_2^0, sig_1^0, sig_2^0) = (1)(p_1 + 0.1, p_2 - 0.1, m_1 + 0.5, m_2 + 0.5, sig_1 + 0.5, sig_2 + 0.5); (p_1 + 0.2, p_2 - 0.2, m_1 + 5, m_2 + 5, sig_1 + 1, sig_2 + 1); (p_1 + 0.1, p_2 - 0.1, m_1 + 1, m_2 + 1, sig_1 + 3, sig_2 + 3)$ . Ainsi, le profil (1) correspond à une déviation légère des vrais valeurs. Le profil (2) nous montre ce qu'il se passe pour un gros écart autour des moyennes et (3) une grosse déviation des écarts types. On obtient les courbes de convergence de EM suivante associées à leurs taux moyen d'erreur pour une segmentation MPM aveugle.



**FIGURE 9 – Convergences des différentes valeurs avec l'algorithme EM avec leur taux moyen d'erreur MPM aveugle associé**

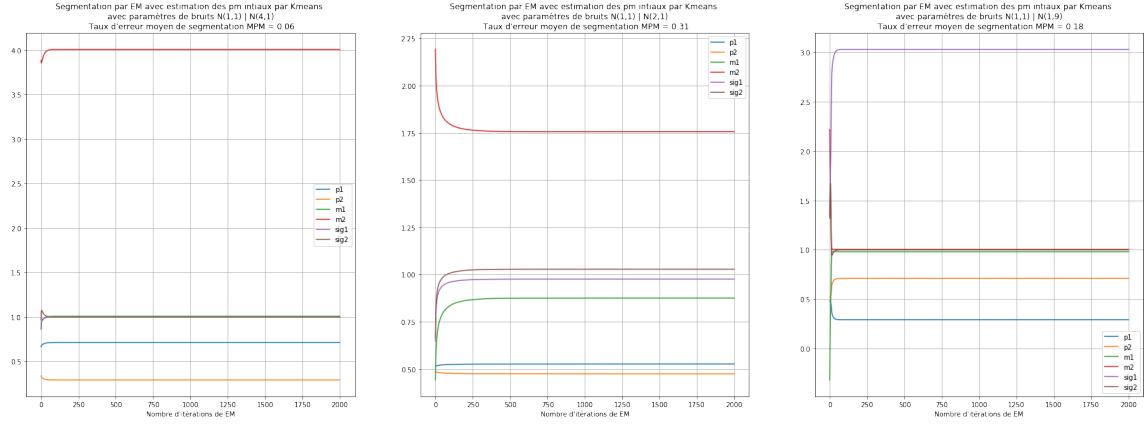
On se rend compte que pour les profils de bruits (1) et (3), la convergence de EM se fait assez facilement. Les valeurs des erreurs de la segmentation sont les mêmes que pour du supervisé. Pour le profil (2), la convergence est plus difficile. On voit que comme nous l'avons expliqué dans la dernière partie, le fait que les 2 gaussiennes soit confondues complique la segmentation. En effet, on voit que les valeurs ont du mal à convergé. Quand on utilise le profil de déviation (2), on se rend compte que EM diverge même de sa valeur "vraie". On peut penser que notre algorithme EM est attiré par un autre extremum local. On voit aussi que les taux d'erreurs moyens reste quand même proche à la valeur du MPM aveugle supervisé. Cela s'explique sûrement par le fait que comme de base notre segmentation n'est pas très bonne, une erreur sur les paramètres estimés n'est pas si grave si elle reste proche de la vrai valeur. En plus, on peut considérer que cette part d'erreur peut dans certains cas compenser l'erreur déjà présente.

Maintenant, nous allons pouvoir effectuer une segmentation totalement non supervisées. L'objectif va être ici d'estimer les valeurs initiales seulement avec l'image observée. On a 2 approches. La premières est une approche naïve qui à calculer  $m_{image}$  et  $sig_{image}$ . On retiendra ensuite les valeurs suivantes :  $(p_1^0, p_2^0, m_1^0, m_2^0, sig_1^0, sig_2^0) = (0.5, 0.5, m_{image} - \frac{sig_{image}}{2}, m_{image} + \frac{sig_{image}}{2}, \frac{sig_{image}}{2}, \frac{sig_{image}}{2})$ . On trouve les résultats suivants :



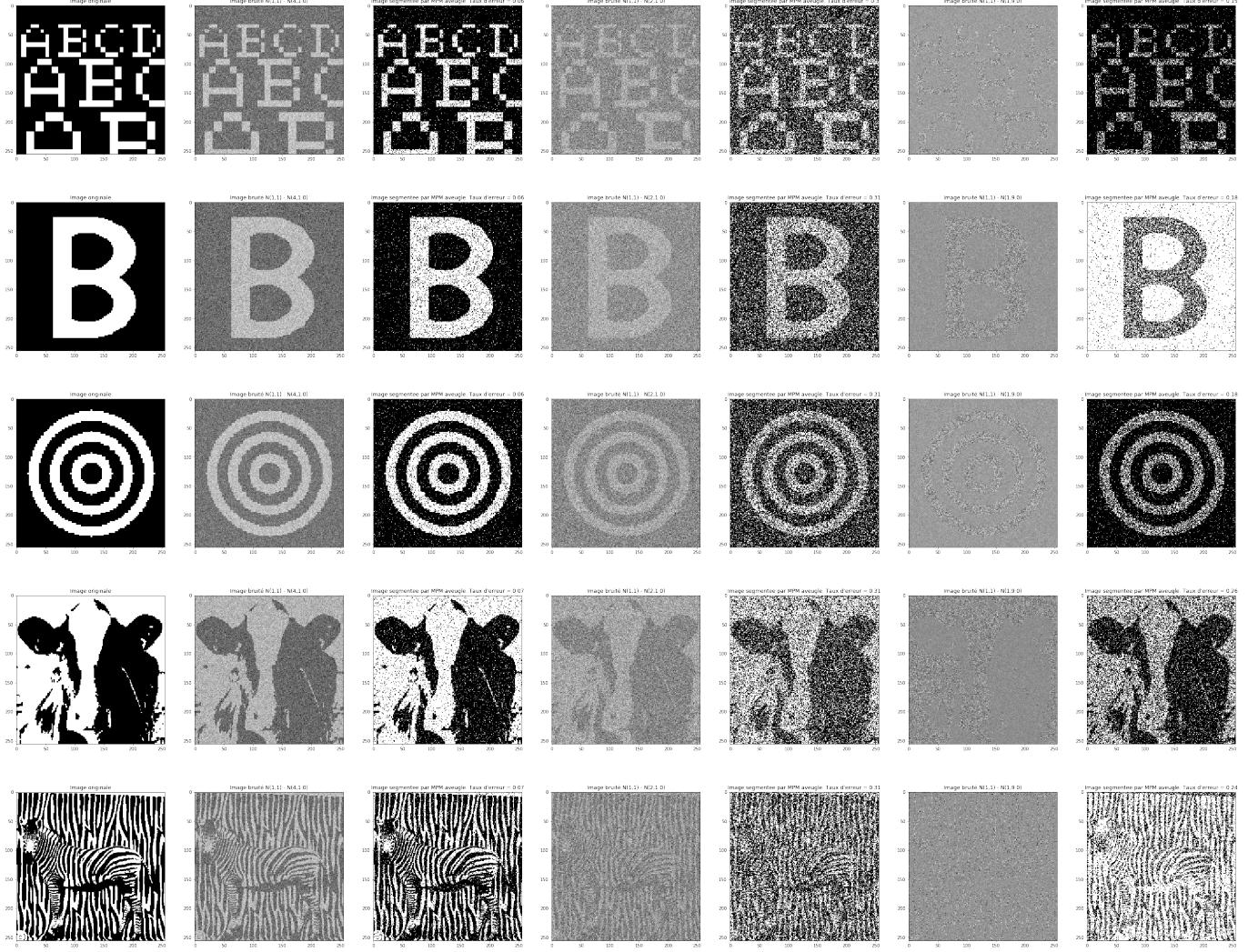
**FIGURE 10 – MPM non supervisée et EM avec initialisation naïve**

L'autre approche consiste à segmenter notre image par un Kmeans et à estimer empiriquement les valeurs recherchées. On trouve les résultats suivants :



**FIGURE 11 – MPM non supervisée et EM avec initialisation KMeans**

Même si en moyenne les taux d'erreurs sont similaires, on se rend compte que Kmeans converge plus vite. On utilisera donc Kmeans comme méthode d'initialisation. On va maintenant pouvoir effectuer notre segmentation aveugle non supervisée sur l'ensemble des images et des bruits pour comparer les résultats avec 2000 itérations de EM. On obtient le tableau suivant :

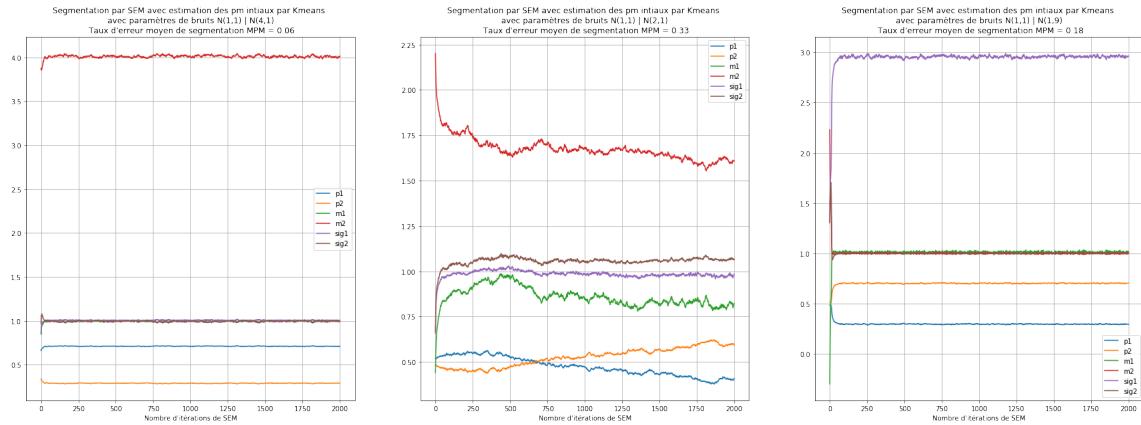


**FIGURE 12 – Segmentation par MPM non supervisée par EM-Kmeans de nos images bruité**

Image \ Distribution du bruit	$\mathcal{N}(1,1), \mathcal{N}(4,1)$	$\mathcal{N}(1,1), \mathcal{N}(2,1)$	$\mathcal{N}(1,1), \mathcal{N}(1,9)$
abcd	0,06	0,3	0,15
B	0,06	0,31	0,18
Cible	0,06	0,31	0,18
Vache	0,07	0,31	0,26
Zebre	0,07	0,31	0,24

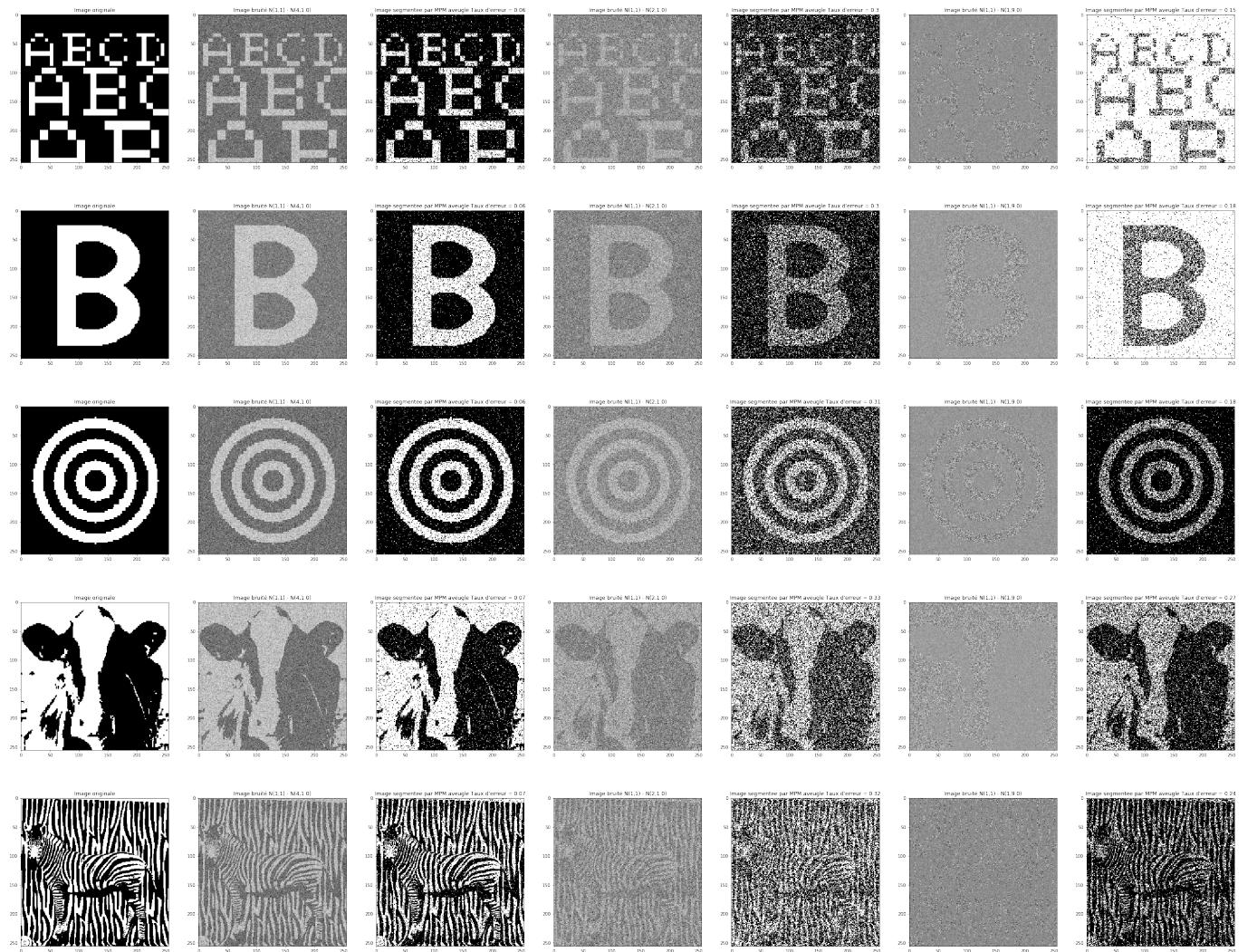
**TABLE 3 – Tableau récapitulatifs des taux d'erreurs moyens sur 200 itérations**

On se rend compte que ici, on a augmenter en erreur sur le cas de bruit  $N(1,1)$ ,  $N(2,1)$  comme expliqué précédemment. Lors du texte, on trouve aussi des cas où EM échoue et donc on ne peut pas segmenter. Cela s'explique par le fait que EM à du se bloquer dans un extremum local. Nous allons donc implémenter une autre méthode dites SEM qui de part son coté stochastique peut résoudre ce problème. Nous allons d'abord tester SEM sur l'image du B pour comparer les convergences avec EM.



**FIGURE 13 – MPM non supervisée et EM avec initialisation KMeans**

On peut voir le coté stochastique sur les courbes. On voit aussi que SEM converge à peu près à la même vitesse que EM. Il présente aussi les mêmes problèmes pour le bruit (2) de difficultés à converger. On peut maintenant segmenter nos images par SEM :



**FIGURE 14 – Segmentation par MPM non supervisée par SEM-Kmeans de nos images bruité**

Image \ Distribution du bruit	$\mathcal{N}(1, 1), \mathcal{N}(4, 1)$	$\mathcal{N}(1, 1), \mathcal{N}(2, 1)$	$\mathcal{N}(1, 1), \mathcal{N}(1, 9)$
abcd	0,06	0,3	0,15
B	0,06	0,31	0,18
Cible	0,06	0,31	0,18
Vache	0,07	0,33	0,27
Zebre	0,07	0,32	0,24

TABLE 4 – Tableau récapitulatifs des taux d'erreurs moyens sur 200 itérations

On voit que même si en moyenne, on obtient des convergences similaires, SEM permet d'éviter que l'algorithme converge vers un extremum local non global. On pourrait s'affranchir de l'ajout du stochastique sur le résultat final de SEM tout en gardant l'avantage sur l'algorithme en prenant comme valeur finale une moyenne des k dernières valeurs.

## 5 Segmentation par champs de Markov cachés supervisée

Grace à un échantillonneur de Gibbs, nous allons pouvoir simuler un champs de Gibbs suivant une distribution donnée. On a donc pu générer un champs de paramètre  $\alpha = 1$ . Ensuite, on le bruite et on peut le segmenter par la méthode des champs. On peut voir ici, le résultat :

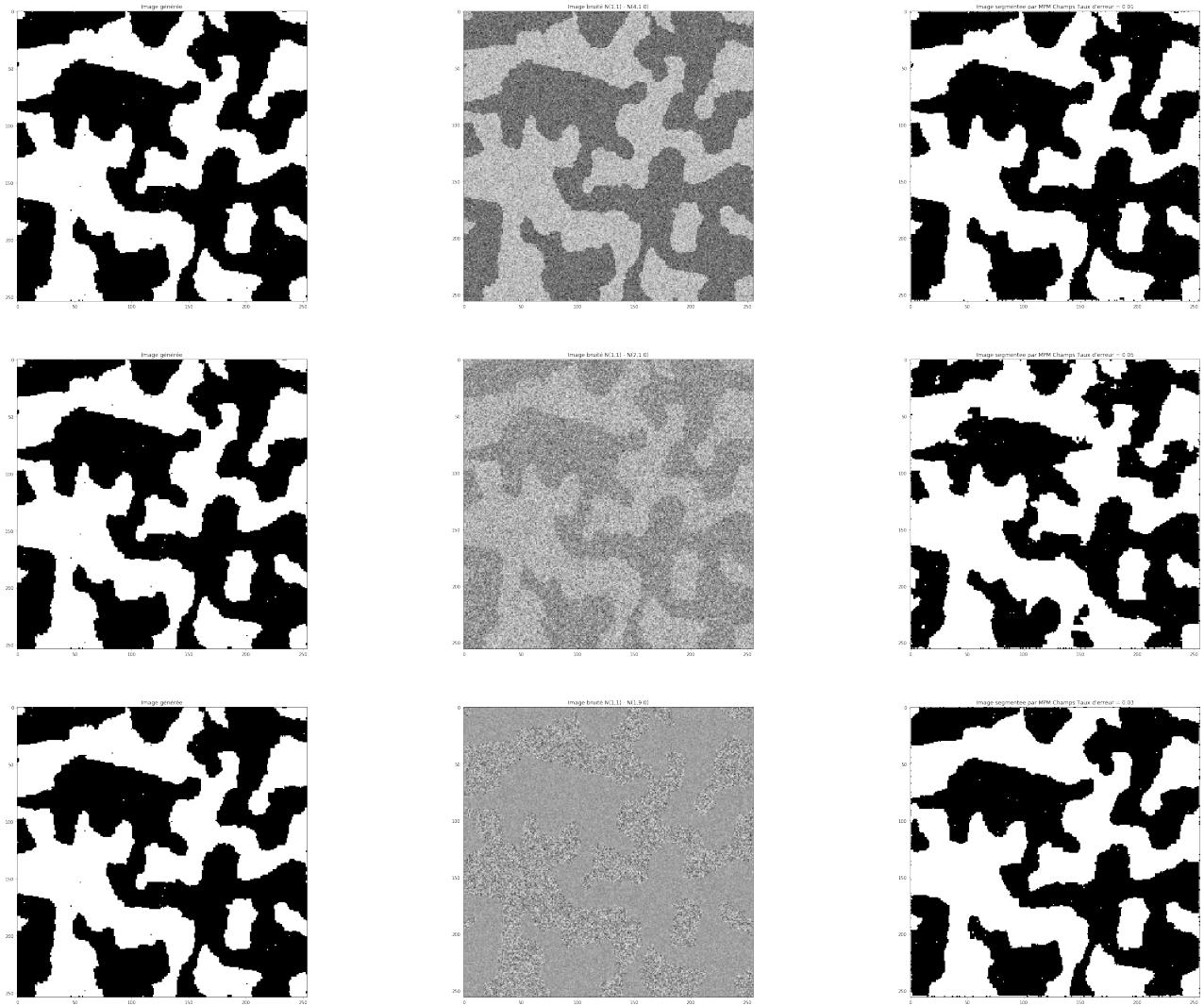
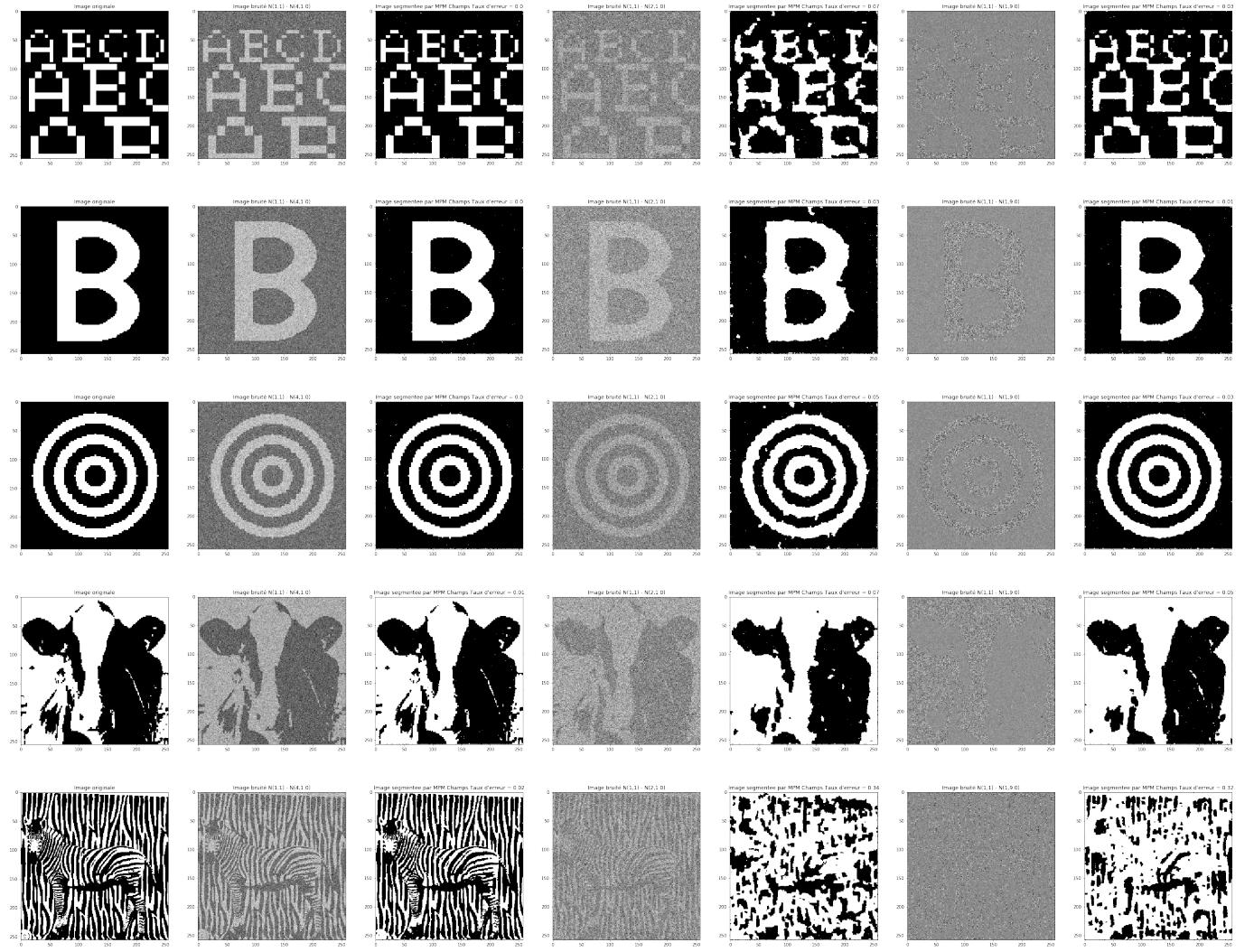


FIGURE 15 – Génération de champs de Markov et segmentation par champs de markov cachée

On voit que dans le cas de la segmentation de champs de markov, notre modèle atteint des taux d'erreur très faibles. Cependant, dans le cas d'image réelles, on ne connaît pas la probabilité conditionnellement au

voisinage et on va donc devoir l'estimer. Dans un premier temps, nous allons considérer que cette probabilité est fixé et nous allons segmenter nos images comme dans les cas précédents. Pour des problèmes de complexité, nous allons effectuer notre étude sur des images 128x128. On obtient le résultat suivant :



**FIGURE 16 – Segmentation par champs de markov d’image réelles bruité supervisée**

On remarque que à part pour le zèbre, on obtient des taux d’erreurs très bas. On voit cependant apparaître certains artifices lors de la segmentation (surtout pour le profil de bruit où les gaussiennes sont très proches). Pour le zèbre, cela peut s’expliquer par le fait que l’image comprend une certaine texture qui n’est pas inclue dans le modèle.

On remarquera que ici, on a effectué une approximation sur la proba conditionnellement aux voisnages qui peut avoir induit des erreurs de segmentation. Ceci sera réglé lors de l’approche non supervisée. On obtient le tableau des erreurs suivant :

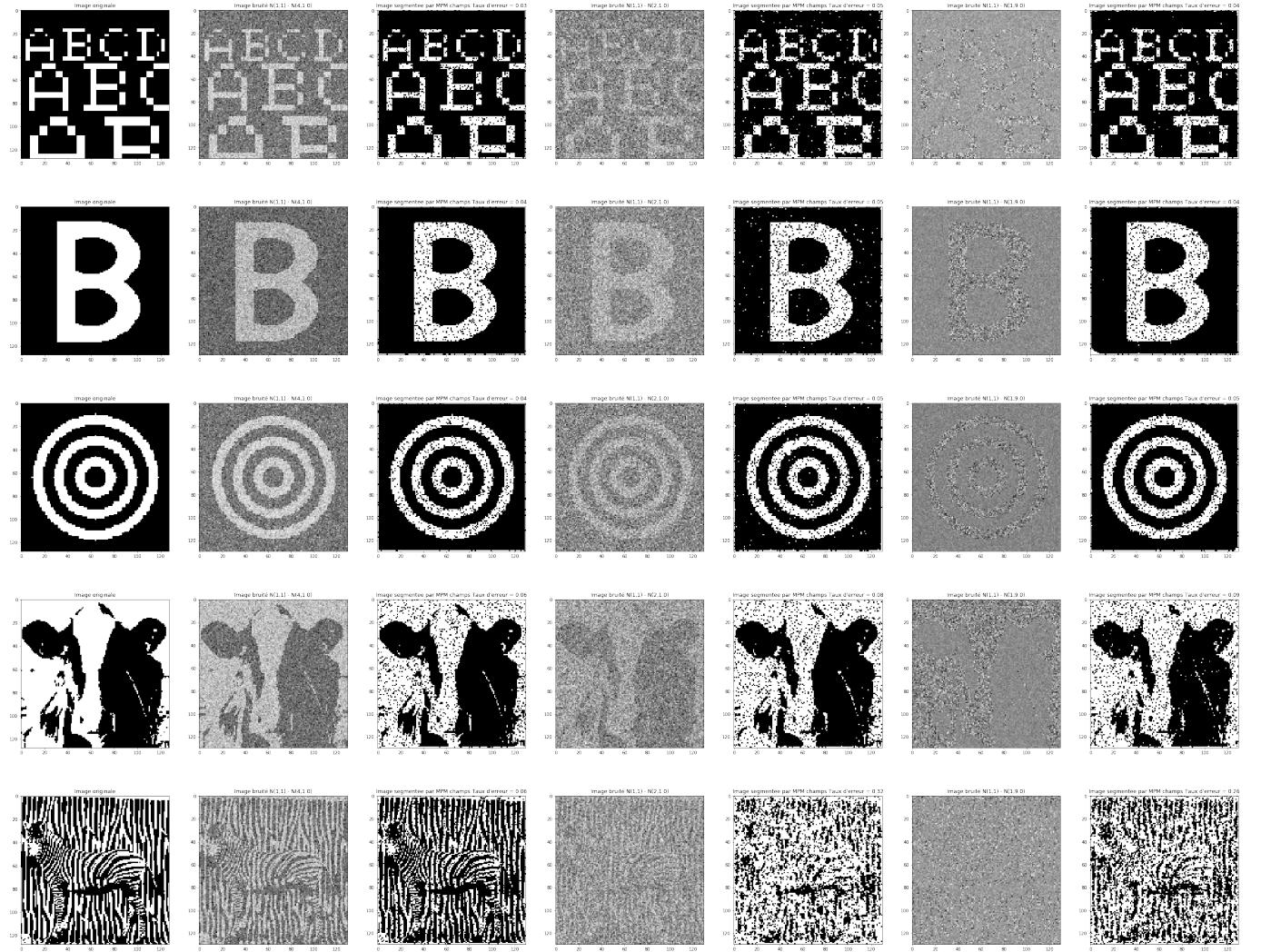
Image \ Distribution du bruit	$\mathcal{N}(1,1), \mathcal{N}(4,1)$	$\mathcal{N}(1,1), \mathcal{N}(2,1)$	$\mathcal{N}(1,1), \mathcal{N}(1,9)$
abcd	0,005	0,07	0,03
B	0,005	0,03	0,01
Cible	0,005	0,05	0,03
Vache	0,02	0,07	0,05
Zebre	0,02	0,34	0,32

**TABLE 5 – Tableau récapitulatifs des taux d’erreurs moyens sur 200 itérations**

A part le zèbre, nos images ont une segmentation très bonne par rapport aux résultats du supervisé aveugle

## 6 Segmentation par champs de Markov cachés non-supervisée

Pour cette partie, la complexité de notre algorithme est devenue prohibitive. Chaque itérations sur les 15 différents cas précédents mets plus de 24h sur des images 128x128. Nous avons essayé d'effectuer quelques itérations mais l'ordinateur à fini par shutdown le kernel. Nous allons donc devoir nous contenter de taux d'erreur propre à l'image et non moyens. On trouve le résultat suivant :



**FIGURE 17 – Segmentation par champs de markov d'image réelles bruité non supervisée**

On remarque que le résultat final n'a pas le même aspect que le résultat supervisé. Ma théorie est que notre EM aurait besoin de plus d'itération pour bien faire converger la probabilité conditionnelle aux voisinages car on remarque la présence de pixels isolées qui peuvent être due à une proba qui ne sanctionne pas assez les pixels à zéros voisins similaires. Cependant, on obtient quand même des taux d'erreurs meilleurs que la segmentation aveugle. On obtient le tableau suivant :

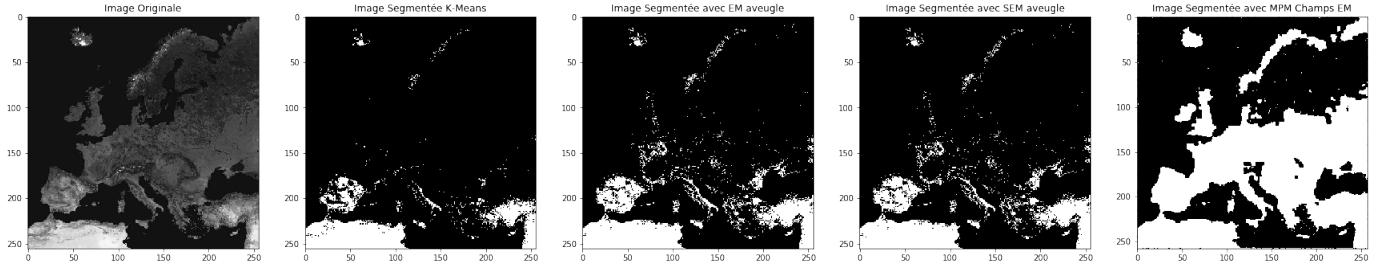
Image \ Distribution du bruit	$\mathcal{N}(1,1), \mathcal{N}(4,1)$	$\mathcal{N}(1,1), \mathcal{N}(2,1)$	$\mathcal{N}(1,1), \mathcal{N}(1,9)$
abcd	0,03	0,05	0,04
B	0,04	0,05	0,04
Cible	0,04	0,05	0,05
Vache	0,06	0,08	0,09
Zebre	0,06	0,32	0,26

**TABLE 6 – Tableau récapitulatifs des taux d'erreurs moyens sur 1 itérations**

Comme on peut le voir, nos résultats sont un peu moins bons mais on constate que l'on observe moins

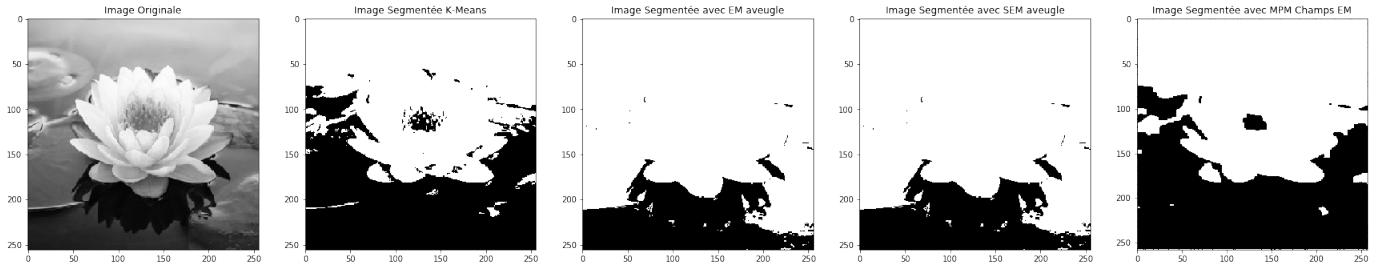
d'artifact visuels que sur le supervisé. Ceci est du au fait que l'on estime les proba conditionnelle au voisinage. Ainsi, on peut supposer que si on arrive à avoir suffisament d'itérations pour se débarasser des pixels isoler, on devrait arriver à obtenir de meilleurs résultats.

## 7 Segmentation réelle



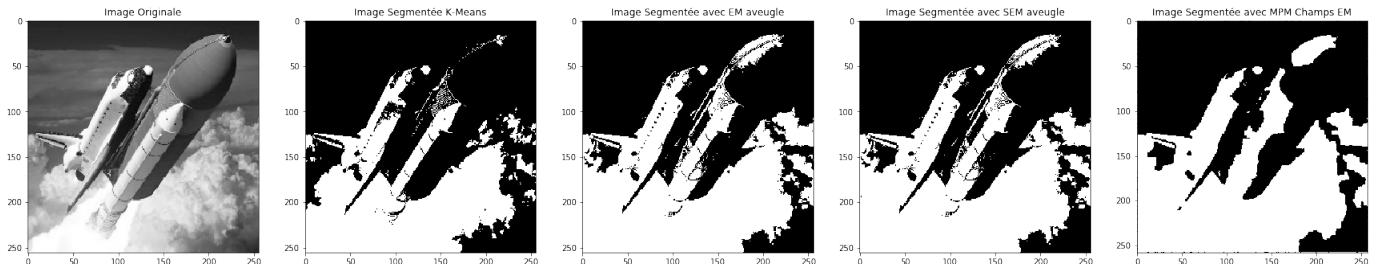
**FIGURE 18 – Segmentation d'une image satellite de l'europe**

On voit que la segmentation champs de l'europe est bien meilleure que les autres méthodes. A part la norvège, le continent européen est plutot bien segmentée. Ici, 2 classes devait clairement être identifiée : L'eau et la terre.



**FIGURE 19 – Segmentation d'une image d'une fleur de lotus**

Ici, on voit que K-Means est la méthode qui marche le mieux. En effet, sur MPM, l'algorithme à du mal à différencier la fleur de l'eau. Peut être que pour améliorer la segmentation, on devrait la faire sur 3 classes pour distinguer les pétales, des feuilles et de l'eau pour avoir une meilleur segmentation



**FIGURE 20 – Segmentation d'une image satellite de fusée**

Ici, on voit que l'algorithme segmente assez bien la fusée mais identifie les nuages en tant que fusée. Comme la fleur, on devrait essayer avec plusieurs classes. On voit aussi un problème qui se pose pour la segmentation d'objets ayant différents attributs comme la fusée qui à des zones blanches et des zones noires.