MAT 4506
Méthodes statistiques pour la segmentation dans les chaînes de Markov Cachées

Pierre Checchin & Erwan Le Blévec

23 novembre 2021

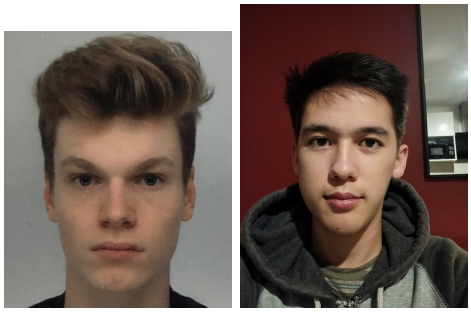


Table des matières

1 Introduction

Ce rapport rassemble les parties 4 et 5 du TP de modélisation. Dans un premier temps nous nous consacrerons à la mise en place de l'algorithme *forward-backward* simple dans cas d'une considération MPM du problème de modélisation puis nous discuterons des limites qu'il rencontre ainsi que des approches existantes pour les contourner. La seconde partie sera consacrée à une mise en pratique des algorithmes vus précédemment : à l'aide d'un parcours d'Hilbert Peano nous transformerons des images en chaînes (bruitées) que nous chercherons à segmenter à travers les trois approches que nous avons vu dans ce TP. Cette partie constitue la finalité de ce TP, elle nous permettra de conclure sur la pertinence des différentes segmentations en fonction des 5 bruits initiaux considérés.

2 Segmentation supervisée d'un signal grâce aux chaînes de Markov cachées

2.1 Introduction aux chaînes cachées de Markov

Les chaînes cachées de Markov peuvent intervenir dans le cas général d'un problème de lissage. C'est le cas de ce TP où, connaissant la valeur de la totalité des pixels (le bruit), nous cherchons à calculer la classe de chacun d'eux par le critère du maximum des lois a posteriori. L'expression directe de ces lois est :

$$p(x_n|y_{0..N}) = \frac{p(x_n,y_{0..N})}{p(y_{0..N})} \text{ avec } p(y_{0..N}) = \sum_{x_n} p(y_{0..N}, x_{0..N})$$

Calculer la loi de $y_{0..N}$ demanderait donc K^{N+1} opérations. À titre indicatif, pour une image de 30*30 pixels avec une segmentation selon deux classes, cela représenterait 2^{900} opérations soit $8.45 * 10^{270}$ opérations (environ). Sachant que les dernières cartes graphiques NVIDIA peuvent en réaliser $312 * 10^{12}$ par seconde cela demanderait (environ) $8.6 * 10^{248}$ années de calcul.

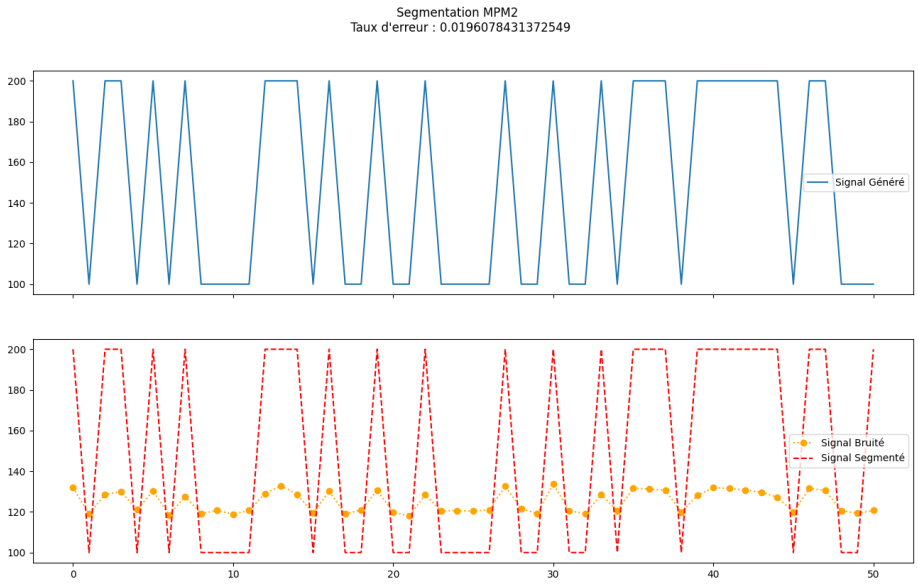
Tout l'intêret de modéliser la situation par des chaînes cachées réside dans la réduction du nombre de calculs nécessaires pour obtenir la vraisemblance (et donc le temps calcul).
Les chaînes de Markov cachées permettent d'obtenir l'expression suivante :

$$p(x_n|y_{0..N}) = \frac{p(y_{n+1..N}|x_n)p(y_{0..n},x_n)}{\sum_{x'_n} p(y_{n+1..N}|x'_n)p(y_{0..n},x'_n)}$$

Ce qui réduit le problème au calcul des $alpha_n(i) = p(y_{0..n}, x_n^i)$ et $beta_n(i) = p(y_{n+1..N}|x_n^i)$ que l'on peut exprimer recursivement (voir cours). Cette nouvelle expression de la vraisemblance conditionnelle ne demande plus que $(N+1) * K^2$ opérations (à un facteur proportionnel compris entre 1 et 10 près), soit dans le cas précédent environ 1200 opérations. La réduction est particulièrement conséquente.

Le début du TP consiste en la mise en oeuvre des algorithmes de calcul des grandeurs utiles $(mat_f; alpha, beta)$

Exemple de signal segmenté par forward/backward :



2.2 Comparaison MAP et MPM_CC

2.2.1 Une erreur moyenne "moyenne" .. ?

La nouvelle approche que demande ce TP nécessite une légère modification dans la manière que nous avions d'obtenir nos résultats jusqu'à présent. Maintenant pour générer des signaux nous

devons fixer en amont cinq couples d'hyperparamètres $(A, a = p10)$ qui serviront de base à la réalisation de nos algorithmes. Or jusqu'à présent pour calculer l'erreur moyenne de segmentation sur un type de signal nous n'avions qu'à le bruiter un certain nombre de fois, calculer le taux d'erreur de chaque segmentation et en déduire le taux d'erreur moyen. Cette démarche nous a prouvée être pertinente dans les TP précédents du fait de la convergence du taux moyen obtenu (le nombre de segmentations finales était fixé à 500). Or ici il n'est plus question de calculer directement l'erreur moyenne sur un signal mais sur les hyperparamètres (A, a), la question sous jacente étant : *Existe-t-il des "formats" de signaux pour lesquels MAP est meilleure malgré son caractère local ou MPM_CC est-il toujours plus précis ?*
Générer simplement un signal X puis calculer l'erreur moyenne des deux méthode sur 500 segmentations reviendrait donc à réduire l'erreur moyenne sur (A, a) à celle sur un signal *généré* par (A, a). Or puisqu'il existe une infinité de signaux (approche stochastique) qui puissent en être issu, à la question "L'erreur moyenne sur ce signal généré par (A, a) est-elle représentative de l'erreur moyenne sur l'ensemble des signaux générés par ce coupe ?" il est sensé de répondre négativement (rien ne nous garanti que X est représentatif de cet ensemble de signaux, du fait du caractère aléatoire de sa génération). Pour garantir des résultats exploitables nous devons donc simuler un "grand" nombre de réalisations de (A, a) sur lesquelles il faudra calculer à chaque fois l'erreur moyenne des deux segmentations et enfin moyenner cette erreur. Avec cette approche seulement nous pourrons parler "d'erreur moyenne" sur (A, a).

2.2.2 Résultats

Pour obtenir ces résultats nous avons choisis : $\begin{cases} \text{Nombre de signaux générés par couple : } \mathbf{50} \\ \text{Nombre de segmentations pour chaque signal : } \mathbf{250} \end{cases}$

Les valeurs obtenues sont présentées dans le tableau suivant :

Bruit	Bruit 1	Bruit 2	Bruit 3	Bruit 4	Bruit 5
Segmentation	MAP, MPM-CC	*idem*	*idem*	*idem*	*idem*
(A1, a1)	0.0, 0.0	0.29, 0.2	0.6, 0.2	0.0, 0.0	0.65, 0.21
(A2, a2)	0.0, 0.0	0.19, 0.19	0.3, 0.3	0.0, 0.0	0.39, 0.4
(A3, a3)	0.0, 0.0	0.28, 0.21	0.27, 0.26	0.0, 0.0	0.3, 0.3
(A4, a4)	0.0, 0.0	0.13, 0.18	0.29, 0.27	0.0, 0.0	0.26, 0.28
(A5, a5)	0.0, 0.0	0.27, 0.13	0.37, 0.14	0.0, 0.0	0.56, 0.14

TABLE 1 – Comparaison des erreurs moyennes "moyennes" des méthodes MAP et MPM_CC

Les résultats affichés ici sont d'autant plus encourageant que les écarts d'erreurs observés ne dépassent pas les 1% lorsque nous avons lancé plusieurs fois le programme (pour un même couple (A, a) et un même bruit).
Les couples utilisés sont :

$$\text{A1} = \begin{pmatrix} 0.05 & 0.95 \\ 0.25 & 0.75 \end{pmatrix} \quad a1 = 0,77 \qquad \text{A4} = \begin{pmatrix} 0.88 & 0.12 \\ 0.23 & 0.77 \end{pmatrix} \quad a4 = 0,8$$

$$\text{A2} = \begin{pmatrix} 0.34 & 0.66 \\ 0.48 & 0.52 \end{pmatrix} \quad a2 = 0,46 \qquad \text{A5} = \begin{pmatrix} 0.15 & 0.85 \\ 0.13 & 0.87 \end{pmatrix} \quad a5 = 0,56$$

$$\text{A3} = \begin{pmatrix} 0.64 & 0.36 \\ 0.17 & 0.83 \end{pmatrix} \quad a3 = 0,17$$

La première observation qui ressort de ce tableau est la supériorité de l'algorithme *forward-backward* sur celui du maximum *a posteriori*. Nous observons jusqu'à 44% de précision supplémentaire dans certains cas.
Pour A : Lorsque les termes de la matrice de transition sont proches de 0 ou 1 ($A1$ et $A4$) on observe des écarts d'erreurs entre MAP et MPM_CC bien plus importants que lorsqu'ils se rapprochent de 0.5. Dans ce cas ils semblent converger vers la même valeur.

Cela peut s'expliquer par le fait que pour une matrice $A = \begin{pmatrix} 0.5 & 0.5 \\ 0.5 & 0.5 \end{pmatrix}$ et pour tout n dans $[\![1; N]\!]$ les probabilités des $p(x_n = w_i)$ (où i $\in \{1, 2\}$) sont égales aux probabilités de la la racine, soient celles prises en compte par l'algorithme MAP.

Pour la racine : Il semblerait que les errreurs sont d'autant plus importantes que les probabilités sont proche de 0 ou 1 et à l'inverse d'autant plus faibles qu'elles sont proches de 0.5.

2.3 Et dans la pratique ?

La suite du TP nous approche un peu plus d'un "vrai" cas pratique de segmentation, à savoir celui où nous ne connaitrions que le signal bruité et les classes à déterminer.

Par exemple, utiliser l'algorithme *forward-backward* nécessite la connaissance de A et a afin de calculer les termes en $alpha_n$ et $beta_n$. Or depuis le début de ce TP nous définissons ces couples (A, a) servant à générer nos signaux, ils nous sont donc *connus* là où, en réalité, nous aurions à les estimer.

2.3.1 Problème de l'estimation de A

Le TP propose une approche à mi-chemin entre la connaissance parfaite de ces signaux et la réalité : nous connaitrons certe X mais pas A, que nous chercherons à estimer à partir de ce signal. Une première critique à l'encontre de cette approche est la mauvaise approximation qui résulte parfois de l'estimateur empirique. Pour certaines valeurs de A, comme le montre le tableau ci-dessous, l'erreur d'approximation est plus que conséquente.

Pour obtenir ces résultats nous nous sommes basés des signaux de longueur 50000, mais avons utilisé seulement les 50, 250 et 5000 premiers termes pour estimer A. Les valeurs prises par A sont celles introduites dans la partie précédente.

Longueur	50	250	5000
Coordonnées	a11, a12, a21, a22	*idem*	*idem*
A1	0.15, 0.15, 0.04, 0.04	0.15, 0.15, 0.04, 0.04	0.16, 0.16, 0.04, 0.04
A2	0.09, 0.09, 0.07, 0.07	0.09, 0.09, 0.06, 0.06	0.08, 0.08, 0.06, 0.06
A3	0.35, 0.35, 0.18, 0.18	0.32, 0.32, 0.16, 0.16	0.32, 0.32, 0.15, 0.15
A4	0.25, 0.25, 0.44, 0.44	0.22, 0.22, 0.42, 0.42	0.22, 0.22, 0.43, 0.43
A5	0.11, 0.11, 0.04, 0.04	0.05, 0.05, 0.02, 0.02	0.02, 0.02, 0.0, 0.0

TABLE 2 – Erreur d'estimation pour 200 générations de signaux par couple, avec a=0,6

Les valeurs ont été arrondies au centième près, $x.x$ indique une valeur exacte. Deux principales observations peuvent être tirées de ce tableau :

1. La première étant que l'erreur d'approximation de A est parfois extrêment lourde (32% en moyenne sur A3, 43% sur A4..), ce qui rendrait toute segmentation basée sur ces valeurs caduque.
2. La seconde est la faiblesse de la convergence de la loi des grands nombres dans notre cas : les écarts d'erreurs entre une longueur de 50 et une longueur de 5000 sont particulièrement faibles (de l'ordre de 1 à 4%..)

L'algorithme utilisé pour aboutir à ces résultats est détaillé dans l'annexe ; voir Programmes_ Annexes puis comp_ A_ emp.py

2.3.2 Problème de la disparition des grandeurs

Cette approche - par l'estimation de A en amont - appliquée aux signaux fournis dans l'énoncé nous amène très rapidement face à un nouveau problème : celui de la disparition des grandeurs $alpha_n$ et $beta_n$. Cette disparition est d'ordre purement technique : les $alpha_n$ et $beta_n$ diminuant avec les itérations, lorsque la taille de la chaîne est trop importante il arrive un instant où ils dépassent le seuil à partir duquel l'ordinateur les considère nuls (ce seuil est d'environ 10^{-23}). Toutes les vraisemblances devenant dès lors nulles l'algorithme n'est plus en mesure de trancher correctement sur la classe et son erreur moyenne tend vers 0.5 (une chance sur deux de se tromper puisque nous n'avons que deux classes possibles).

Bruit	Bruit 1	Bruit 2	Bruit 3	Bruit 4	Bruit 5	Longueur
Segmentation	MPM-CC	MPM-CC	MPM-CC	MPM-CC	MPM-CC	-
Signal 0	0.00	0.16	0.26	0	0.33	50
Signal 1	0.00	0.15	0.28	0	0.35	250
Signal 2	0.50	0.50	0.50	0.50	0.50	50000
Signal 3	0.50	0.50	0.50	0.50	0.50	50000
Signal 4	0.50	0.50	0.50	0.50	0.50	50000
Signal 5	0.50	0.50	0.50	0.50	0.50	50000

TABLE 3 – Erreur moyenne pour 500 segmentations sur chaque signal

En se référant au TP précédent on remarque immédiatement l'infériorité de MPM sur MAP en cas de disparition des grandeurs, ceci malgré le caractère local de MAP.

Or non seulement plus notre signal est long, meilleures seront (ou en tout cas sont censées être ...) nos estimations , mais surtout ce nombre d'itérations est un invariant puisqu'il est égal à la longueur du signal à estimer !
Dans ces conditions il a été proposé d'utiliser des termes dits "de mise à l'échelle". Cela consiste à multiplier chaque $alpha_n^i$ et $beta_n^i$ par un scalaire à définir pour empêcher ces grandeurs de diminuer trop rapidement. Les scalaires utilisés dans ce TP seront ceux classiques (vu en cours).

Nous obtenons alors les résultats suivants (avec l'erreur d'approximation sur A) :

Bruit	Bruit 1	Bruit 2	Bruit 3	Bruit 4	Bruit 5	Longueur
Segmentation	MPM-CC	MPM-CC	MPM-CC	MPM-CC	MPM-CC	-
Signal 0	0.00	0.16	0.26	0	0.34	50
Signal 1	0.00	0.15	0.28	0	0.36	250
Signal 2	0.00	0.16	0.29	0	0.36	250
Signal 3	0.00	0.16	0.28	0	0.36	50000
Signal 4	0.00	0.18	0.31	0	0.38	50000
Signal 5	0.00	0.18	0.31	0	0.38	50000

TABLE 4 – Erreur moyenne avec rescaling pour 500 segmentations sur chaque signal

Nous remarquons immédiatement la différence avec le simple *forward-backward* : les erreurs d'approximations sont bien plus faibles et surtout il est maintenant possible de considérer cette approche comme exploitable.

2.4 Conclusion sur les trois segmentations

Avant de conclure sur les différents taux d'erreur de nos trois méthodes en fonction du signal et du bruit apposé, nous proposons un bref état des lieux sur l'ensemble de ce que nous avons vu jusqu'à présent.
Le cadre de notre problème était le suivant : Placé dans un cas (supposé) d'inférence bayésienne, c'est à dire que nous n'avions pas accès aux données directement (le signal) mais à des données issues de ces dernières (le signal bruité), nous devions parvenir à retrouver ce signal initial.

Notre première approche fut celle du Bayes Naïf *(Les couples (Xi, Yi) représentants chaque pixel sont supposés indépendants)* avec un classificateur reposant sur le maximum de vraisemblance. Le critère de décision était donc le suivant :

$$\forall i \in [\![1; N]\!] \;\; \hat{w}_i = \arg \max_w \{ \; p(y|x_i = w_1), \; p(y|x_i = w_2) \; \}$$

Reste à exprimer $p(y|x_i)$. Dans la totalité du TP nous considérerons que cette densité conditionelle est une gaussienne de paramètres dépendants de la classe de x_i. Notre approche revient donc à simplement calculer la valeur de ces deux gaussiennes en chaque pixel et à choisir la classe de celle qui y est maximale.

La limite de cette approche est qu'elle sous-entend une équiprobabilité d'apparition entre les deux classes, ce qui peut (souvent) se révéler incorrect. Suite logique de ce constat, notre prochaine méthode de segmentation permet de prendre en compte cette potentielle disparité.
Le critère de décision est maintenant le suivant :

$$\forall i \in [\![1;N]\!] \;\; \hat{w}_i = \arg\max_w \{ \; p(y|x_i = w_1)p(x_1), \; p(y|x_2)p(x_i = w_2) \; \}$$

L'expression découle directement de l'expression de la vraisemblance conditionnelle $p(x|y) = \frac{p(y|x)p(x)}{p(y)}$*, les* $p(y)$ *étant présents de part et d'autre du maximum, on peut les simplifier.*

$\{p(y|x_i)\}_{k\in[\![1;2]\!]}$ étant connu, il restait à estimer la loi à priori de X. Nous avons utilisé deux approches : $\begin{cases} \text{Estimateur empirique} \\ (X_n)_{n\in[\![1,N]\!]} \text{ représentables par une chaîne de Markov} \end{cases}$

L'estimateur empirique sous-entend une indépendance entre chaque "classe" du signal de X, la viabilité de cette hypothèse reste donc à démontrer *(voir Annexe : "Une comparaison supplémentaire")*. À l'inverse les chaîne de Markov impliquent une dépendance de chaque terme vis à vis de son prédécesseur ce qui, pour un signal continu de plusieurs classes (une sinusoïde par exemple), est bien plus adapté (impossible de passer de la classe "du maximum" à la classe "du minimum" entre deux points).

L'étape suivante consistait en la prise en compte non plus de la seule valeur du point que nous souhaitions classifier mais de la totalité des valeurs du signal disponibles. Théoriquement il s'agissait d'une transition des méthodes **locales** vers les méthodes **globales**.
La forme du classificateur était la suivante :

$$\forall i \in [\![1;N]\!] \;\; \hat{w}_i = \arg\max_w \{ \; p(x_i = w_1|\; y_{0..N}), \; p(x_i = w_2|\; y_{0..N}) \; \}$$

Voir la partie 1 du TP pour plus d'explication sur l'origine de cette expression

Finalement, voici le tableau recensant le taux d'erreur moyen des 3 types de segmentations, pour les 6 singaux et les 5 bruits considérés tout au long du TP :

Bruit	Bruit 1	Bruit 2	Bruit 3	Bruit 4	Bruit 5
Segmentation	MV, MAP, MPM-CC	*idem*	*idem*	*idem*	*idem*
Signal 0	0.00, 0.00, 0.00	0.21, 0.20, 0.16	0.31, 0.28, 0.26	0, 0, 0	0.43, 0.46, 0.34
Signal 1	0.00, 0.00, 0.00	0.18, 0.18, 0.15	0.31, 0.31, 0.28	0, 0, 0	0.38, 0.39, 0.36
Signal 2	0.00, 0.00, 0.00	0.18, 0.18, 0.16	0.32, 0.31, 0.29	0, 0, 0	0.39, 0.38, 0.36
Signal 3	0.00, 0.00, 0.00	0.18, 0.18, 0.16	0.31, 0.31, 0.28	0, 0, 0	0.38, 0.38, 0.36
Signal 4	0.00, 0.00, 0.00	0.18, 0.18, 0.18	0.31, 0.31, 0.31	0, 0, 0	0.38, 0.38, 0.38
Signal 5	0.00, 0.00, 0.00	0.18, 0.18, 0.18	0.31, 0.31, 0.31	0, 0, 0	0.38, 0.38, 0.38

TABLE 5 – Comparaison des erreurs moyennes des méthodes MV, MAP et MPM-CC

On remarque que l'algorithme *forward-backward* se révèle être le plus performant d'un point de vue taux d'erreur moyen (ce qui était prévisible, l'estimateur étant plus fin). Passer d'une méthode locale à une méthode globale permet de gagner 1 à 9% de précision supplémentaire dans le cas de signaux cours (50 à 250 valeurs). Nous pouvons être en revanche être quelque peu déçu des résultats du MAP par rapport au MV, qui ne permet de gagner qu'1 à 2% de précision dans les meilleurs des cas, voire d'être parfois un moins bon estimateur (Bruit 5). L'hypothèse de d'équiprobabilité interclasses dans le cas de nos signaux (deux classes possibles) semble donc pouvoir être tenue.
On remarque enfin que les erreurs des estimateurs sur de très longs signaux (50000 valeurs) semblent converger vers une même valeur. Il n'y aurait donc, dans ce cas, plus d'intêret à privilégier notre méthode globale sur celle la plus simple : la maximum de vraisemblance.

3 Une application à la segmentation d'image

3.1 Présentation du problème

Le seul impératif demandé par nos méthodes de segmentation porte sur le format de l'entrée : l'utilisation des chaînes de Markov requiert une dimension 1. Il est néanmois possible de prolonger ces dernières à des dimensions supérieures. En 2e dimension l'algorithme d'Hilbert-Peano par exemple permet de transformer une image en une chaîne, tout en conservant "le maximum d'information" au sens des chaînes de Markov.

Nous pouvons dès lors appliquer nos algorithmes de segmentations à des images bruitées, ce que propose de réaliser ce TP dans cette 5e et dernière partie.

3.2 Réalisation & Comparaison

Pour ne pas avoir à modifier l'intégralité de nos fonctions nous considérerons des images en noir et blanc. Le but sera d'estimer à partir d'images bruitées (en niveau de gris) la classe de chaque pixel, c'est à dire pixel noir (classe 1) ou pixel blanc (classe 2), ce qui permet de conserver la dualité des classes utilisées dans nos fonctions.

Dans le cas de la figure "beee.bmp"

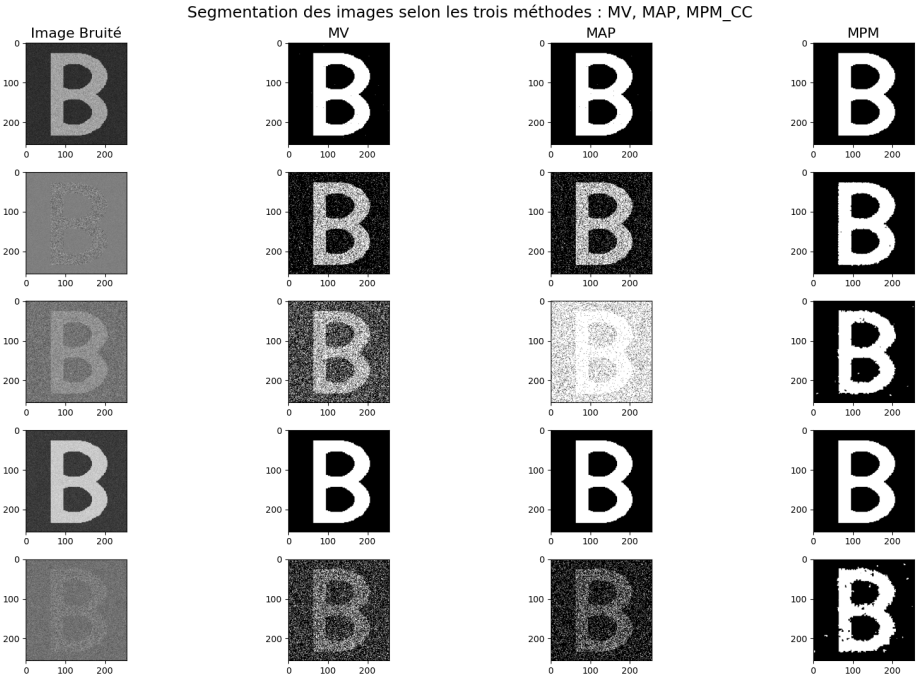


FIGURE 1 – Comparaison des trois segmentations sur une image représentant un B

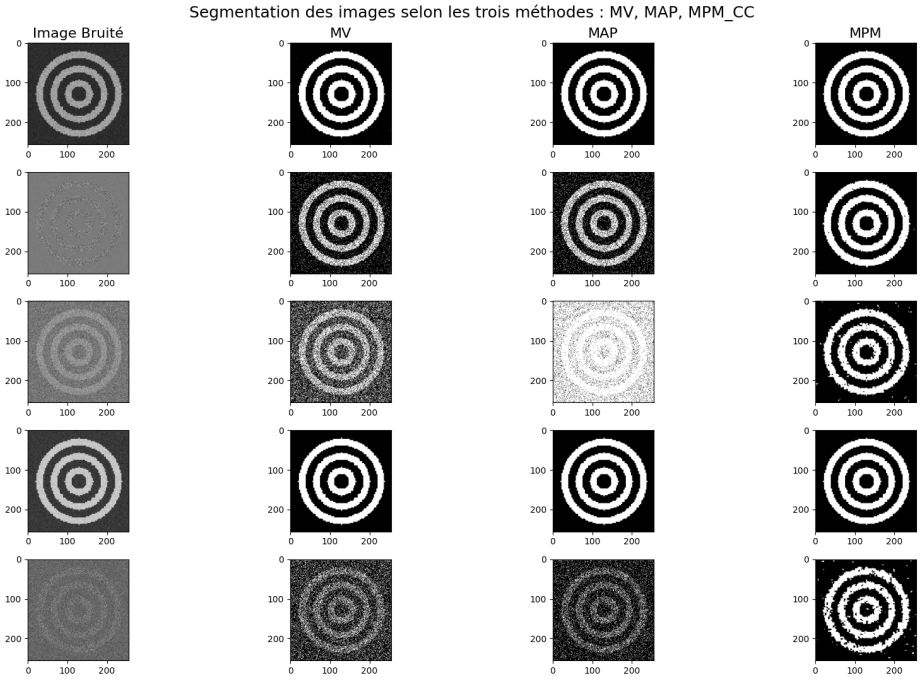


FIGURE 2 – Comparaison des trois segmentations sur une image représentant une cible

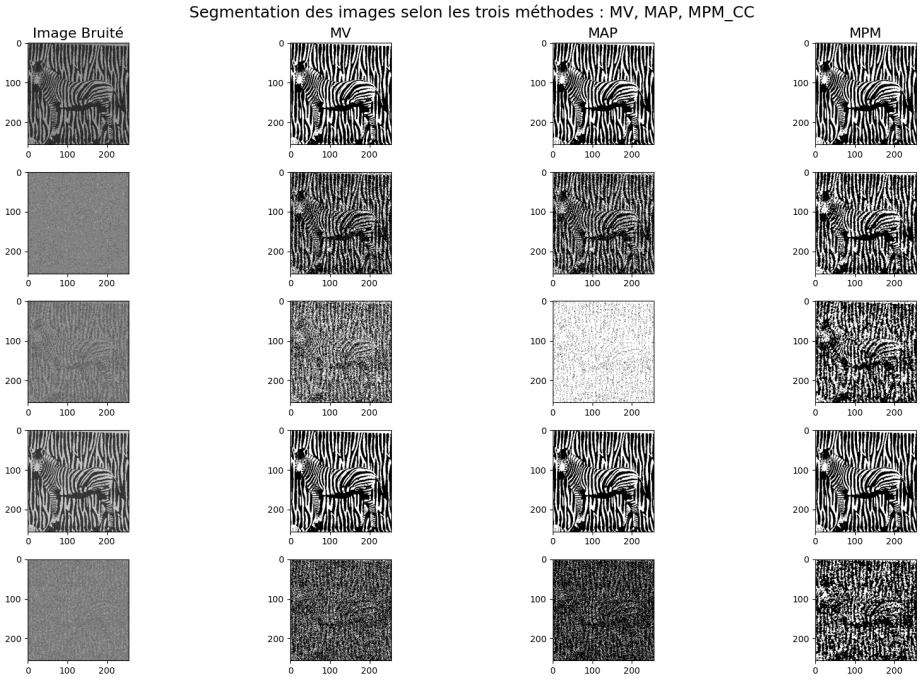


FIGURE 3 – Comparaison des trois segmentations sur une image représentant un zèbre

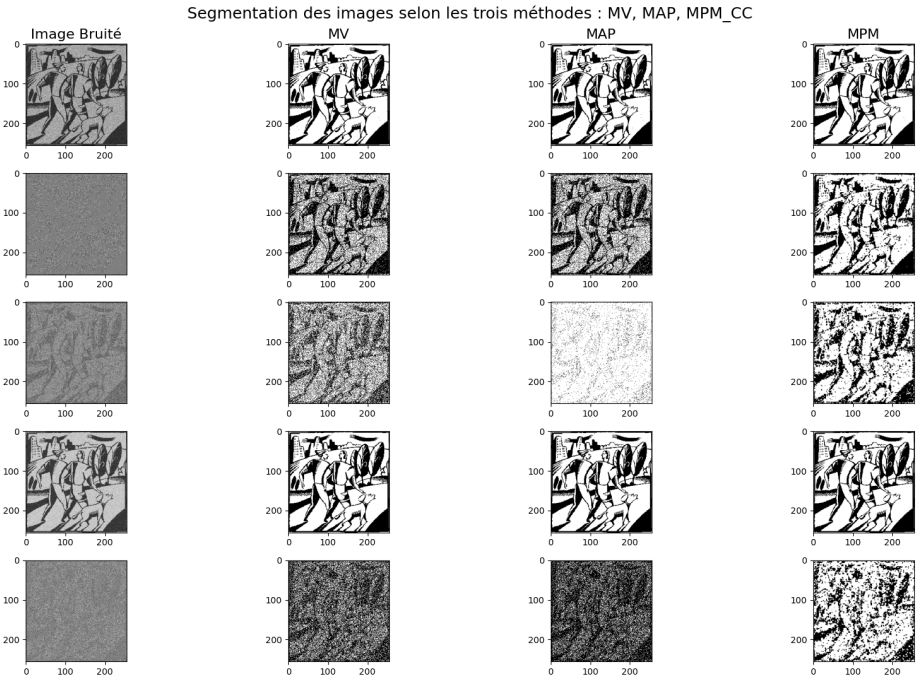


FIGURE 4 – Comparaison des trois segmentations sur une image représentant deux personnes

On observe dans la majorité des cas une franche avance de la part de MPM-CC sur les autres méthodes de segmentation, du point de vue du taux d'erreur moyen en tout cas. Cela est cohérent du fait du caractère "local" des deux premières méthodes, là ou MPM-CC est une globale, cette caractéristique étant d'autant plus importante pour une image où les pixels proches ont de grandes chances d'avoir des valeurs (donc classe ici) semblables.

Bruit	Bruit 1	Bruit 2	Bruit 3	Bruit 4	Bruit 5
Segmentation	MV, MAP, MPM-CC	*idem*	*idem*	*idem*	*idem*
B	0.00, 0.00, 0.00	0.11, 0.10, 0.01	0.30, 0.25, 0.03	0, 0, 0	0.29, 0.27, 0.02
Cible	0.00, 0.00, 0.00	0.15, 0.13, 0.02	0.30, 0.27, 0.04	0, 0, 0	0.30, 0.36, 0.07
Zèbre	0.00, 0.00, 0.00	0.24, 0.20, 0.08	0.31, 0.25, 0.14	0, 0, 0	0.31, 0.25, 0.10
Promenade	0.00, 0.00, 0.00	0.18, 0.18, 0.07	0.31, 0.28, 0.16	0, 0, 0	0.40, 0.37, 0.23

TABLE 6 – Taux d'erreur sur chaque image en fonction du bruit apposé

4 Annexe

4.1 Une petite étude supplémentaire

Modélisation de la loi a priori de X dans le cas du MAP :

Dans la partie 3 de ce TP nous avons mis en place le classificateur par maximum *a posteriori* or, comme mentionné précedemment, il fait apparaître la loi *a priori* de X et demande donc de l'estimer.

Nous avons utilisé deux estimateurs pour cela :

— *L'Estimateur empirique :* $\overline{p(X = w_i)} = \frac{\text{Nombre d'occurence de la classe i}}{\text{Nombre de valeurs total}}$

— *L'Estimateur découlant des chaînes de Markov :*

$\forall n \in [\![1; N]\!]\ p(X_n = w_i) = \sum_{j\in\{1,2\}} p(X_n = w_i | X_{n-1} = w_j) p(X_{n-1} = w_j)$

Qui nous permet de calculer toutes les probabilités de manière récursive à condition de fixer $p(X_0 = w_{1,2})$ et d'estimer A.

Nous avions fait la remarque que l'estimateur empirique implique une indépendance entre les choix des classes qui se suivent, et que cela pouvait porter préjudice pour certaines segmentations (ce qui a été particulièrement mis en évidement sur les taux d'erreur des images) mais nous n'avions pas vérifié si cela pénalisait réellement la qualité de l'estimateur MAP. Nous avons donc réalisé une petite étude comparant les taux d'erreurs moyens pour une loi *a priori* estimée empiriquement et pour cette même loi estimée via les chaînes de Markov.

Voici les résultats :

Signal	Bruit 1	Bruit 2	Bruit 3	Bruit 4	Bruit 5
Signal 1	0.0	0.199	0.279	0.0	0.36
Signal 2	0.0	0.178	0.311	0.0	0.384
Signal 3	0.0	0.176	0.309	0.0	0.381
Signal 4	0.0	0.177	0.309	0.0	0.382
Signal 4	0.0	0.177	0.309	0.0	0.382
Signal 6	0.0	0.176	0.308	0.0	0.381

(a) Estimateur Empirique

Signal	Bruit 1	Bruit 2	Bruit 3	Bruit 4	Bruit 5
Signal 1	0.0	0.198	0.287	0.0	0.36
Signal 2	0.0	0.178	0.306	0.0	0.38
Signal 3	0.0	0.176	0.309	0.0	0.381
Signal 4	0.0	0.177	0.308	0.0	0.382
Signal 5	0.0	0.177	0.309	0.0	0.382
Signal 6	0.0	0.176	0.308	0.0	0.381

(b) Chaînes de Markov

TABLE 7 – Comparaison des erreurs de MAP selon la modélisation de la loi *a priori* de X

On remarque ainsi que la différence est plus que négligeable. L'hypothèse d'indépendance mutuellement des classes dans le cas d'une segmentation d'un signal à deux classes peut être validée.

Le résultat serait très probablement différent dans le cas d'une segmentation sur une image, même dans le cas des deux classes.

4.2 Codes

L'ensemble des fonctions, modules et figures utilisés pour réaliser ou approfondir ce TP sont disponibles à l'adresse suivante : *https://github.com/Erwanlbv/TSP_P1.git*