Modèle de Mélange Gaussien pour la Segmentation des IRM Cérébrales

1 Introduction

Au cours des dernières décennies, l'imagerie par résonance magnétique (IRM) est devenue un outil central dans les études cliniques du cerveau. La plupart de ces études reposent sur une segmentation précise et robuste des images pour visualiser les structures cérébrales et calculer des mesures volumiques. Le marquage manuel du cerveau humain est un processus fastidieux et long : pour donner une idée, segmenter une image 3D du cerveau en IRM en 40 structures cérébrales peut prendre jusqu'à une semaine, un temps que les médecins et radiologues n'ont généralement pas. Pour ces raisons, de nombreux outils automatiques ont été proposés ces dernières années, comme *Freesurfer*, *SPM*, et *FSL*.

On rappelle le modèle de mélange gaussien (ou *Gaussian Mixture Model*, GMM), un modèle simple permettant de segmenter automatiquement les images IRM du cerveau en différentes structures : la substance blanche (WM), la substance grise (GM) et le liquide céphalorachidien (LCR).

2 Modèle de Mélange Gaussien

On suppose qu'un modèle de mélange gaussien génère nos données :

$$p(D) = \prod_{i=1}^{I} \sum_{k=1}^{K} \pi_k \mathcal{N}(d_i|\mu_k, \Sigma_k), \tag{1}$$

où $\sum_{k=1}^{K} \pi_k = 1$ avec $0 \leq \pi_k \leq 1$ constitue notre prior de segmentation, tandis que la distribution gaussienne multivariée standard

$$\mathcal{N}(d_i|\mu_k, \Sigma_k) = \frac{1}{(2\pi)^{N/2}|\Sigma_k|^{1/2}} \exp\left\{-\frac{1}{2}(d_i - \mu_k)^T \Sigma_k^{-1}(d_i - \mu_k)\right\} \tag{2}$$

est notre fonction de vraisemblance.

Les paramètres de notre modèle μ_k, Σ_k et π_k sont respectivement la moyenne, la matrice de covariance et le coefficient de mélange de la k-ième gaussienne.

Nous pouvons interpréter le coefficient de mélange π_k comme la probabilité d'occurrence de la structure cérébrale k dans le cerveau.

2.1 Ajustement des Paramètres aux Données

L'objectif est de trouver les paramètres qui maximisent la fonction de vraisemblance. Om peut écrire la fonction de log-vraisemblance sous la forme :

$$\ln p(D|\mu, \Sigma, \pi) = \sum_{i=1}^{I} \ln \left\{ \sum_{k=1}^{K} \pi_k \mathcal{N}(d_i|\mu_k, \Sigma_k) \right\}. \tag{3}$$

La sommation sur k, fait qu'il est impossible d'obtenir une solution analytique pour maximiser cette fonction. Une méthode efficace pour maximiser la log-vraisemblance est l'algorithme d'Expectation-Maximization (EM). On donne ici une explication intuitive de la mise à jour des paramètres dans l'EM.

Pour commencer, on annule la dérivée de la log-vraisemblance par rapport à μ :

$$\sum_{i=1}^{I} \frac{\pi_k \mathcal{N}(d_i|\mu_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(d_i|\mu_j, \Sigma_j)} \Sigma_k (d_i - \mu_k) = 0. \tag{4}$$

En réarrangeant l'équation, on peut écrire

$$\mu_k \leftarrow \frac{\sum_{i=1}^{I} w_{ik} d_i}{N_k}, \tag{5}$$

où nous avons défini

$$w_{ik} = \frac{\pi_k \mathcal{N}(d_i|\mu_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(d_i|\mu_j, \Sigma_j)} \quad \text{et} \quad N_k = \sum_{i=1}^{I} w_{ik}. \tag{6}$$

N_k peut être interprété comme le nombre de voxels assignés à chaque classe k, tandis que w_{ik} représente la responsabilité de la classe k pour générer le voxel i. On remarque que la moyenne de la distribution gaussienne associée à l'étiquette k est la moyenne pondérée des intensités des voxels attribués à cette étiquette.

En appliquant le même principe pour Σ et π_k, on obtient

$$\Sigma_k \leftarrow \frac{1}{N_k} \sum_{i=1}^{I} w_{ik} (d_i - \mu_k)(d_i - \mu_k)^T \tag{7}$$

et

$$\pi_k \leftarrow \frac{N_k}{I}. \tag{8}$$

On peut interpréter la mise à jour de la covariance de la distribution gaussienne associée à l'étiquette k comme la variance pondérée des intensités des

voxels attribués à cette étiquette. Le prior pour chaque classe cérébrale est quant à lui la fraction des voxels actuellement assignés à cette classe.

Remarque : Les équations de mise à jour que nous avons dérivées ne sont pas en forme fermée, car elles dépendent des responsabilités w_{ik}. Malheureusement, cela ne garantit pas que notre algorithme convergera vers un optimum global. Néanmoins, nous avons des garanties théoriques que la vraisemblance augmente à chaque itération de l'algorithme EM (voir référence).

3 Algorithme EM : Vue d'Ensemble

On peut résumer l'algorithme EM en les étapes suivantes :

1. **Initialisation** : Initialiser les paramètres du GMM (μ_k, Σ_k et π_k).
2. **Étape d'Espérance (E-step)** : Estimer les responsabilités w_{ik} en utilisant les valeurs actuelles des paramètres.
3. **Étape de Maximisation (M-step)** : Réestimer les paramètres μ_k, Σ_k et π_k en utilisant les mises à jour actuelles des responsabilités w_{ik} calculées à l'étape E.
4. **Itération** : Alterner entre les étapes E et M jusqu'à convergence (ou un nombre prédéfini d'itérations).

4 Exemple de Segmentation du Cerveau avec un GMM

Voyons maintenant ce qu'un GMM est capable de faire avec des images du cerveau. Pour l'exemple suivant, j'utilise une image générée par BrainWeb. Voici l'image, montrant une coupe en vue axiale, coronale et sagittale :

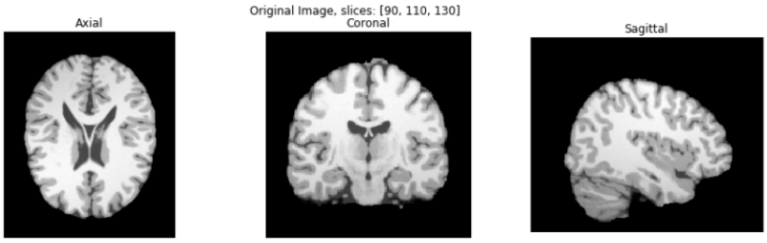

FIGURE 1 – Exemple de segmentation d'une image IRM cérébrale montrant les vues axiale, coronale et sagittale.

Cet exemple démontre comment un GMM peut être utilisé pour segmenter différentes structures cérébrales en fonction des intensités des voxels dans l'image IRM.

5 Segmentation de l'image

Analysez le squelette de programme du fichier LabSession_MM_EM.py. Celui-ci fait appel à des fonctions qui sont toutes décrites dans le fichier func.py. Dans ce dernier fichier, trois fonctions sont vides : à vous de les compléter en suivant les indications ci-dessous !

A faire :

Examinez la structure du programme selon ces 5 phases :

- La lecture dese images.
- L'initialisation des paramètres du mélange.
- L'estimation des paramètres du mélange par l'algorithme EM.
- La classification de l'image selon la décision bayésienne.
- Le dessin des courbes.

À chaque itération, nous prenons soin de sauvegarder les paramètres, ainsi que l'erreur de classification, pour dessiner leur évolution lors de la dernière phase.

1. Complétez la fonction d'initialisation des paramètres avec un algorithme à définir.
2. Complétez la fonction qui itère EM avec les formules de re-estimation vues en cours.
3. Ajoutez une fonction de calcul de la vraisemblance et de la fonction auxiliaire Q, à toutes les itérations de EM.

Astuce :

Réduisez à 3 ou 4 le nombre d'itérations de EM, le temps de réaliser vos tests lors de la phase de développement des algorithmes. Notez qu'il faut environ 20-50 itérations pour obtenir une bonne estimation des paramètres du mélange prévu dans le programme de bruitage de l'image.

Segmenter les images qui se trouvent dans le répertoire *sources*.