# TD : Modélisation : Chaînes de Markov et chaînes de Markov cachées

***Christelle Gonindard***

## 1. Discrimination des îlots CpG

### *A) Construction d'une Chaine de Markov CpG ($P^+$) et d'une chaîne de Markov non CpG ($P^-$)*

Dans le génome humain, la méthylation agit sur les dinucléotides CG, en méthylant la cytosine qui par la suite sera mutée en thymine. Pour cette raison, le taux des dinucléotides CpG (on met le "p" pour différencier ce dinucléotide de la paire C-G entre les deux brins de l'ADN) est anormalement bas. Néanmoins, il existe des régions dans lesquelles ce processus n'agit pas, et où le taux de CpG est plus élevé qu'alentours, régions nommées "îlots CpG". Ces régions ont des longueurs de quelques centaines à quelques milliers de bases.

Notre but est d'utiliser les chaînes de Markov pour discriminer des morceaux de séquence de souris, selon qu'ils possèdent ou non un îlot CpG [1]. L'idée est la suivante : à partir d'un jeu de séquences qui ne contiennent pas d'îlots CpG, on construit un processus de Markov, noté $P^- = (I^-, M^-)$, qui maximise la vraisemblance sur l'ensemble de ces données. A partir d'un autre de jeu de séquences qui ne contient que des îlots CpG, on calcule un autre modèle de Markov, noté $P^+ = (I^+, M^+)$, qui maximise la vraisemblance sur ces séquences :

$$L(\mathcal{S}|\mathcal{P}) = \mathcal{I}_{s_1} \times \prod_i \mathcal{M}_{s_{i-1}s_i}$$

La logvraisemblance est donnée par la formule suivante :

$$\mathcal{L}(\mathcal{S}|\mathcal{P}) = \ln \mathcal{I}_{s_1} + \sum_i \ln(\mathcal{M}_{s_{i-1}s_i}).$$

Ensuite, une séquence inconnue sera classée îlot CpG si sa vraisemblance est plus grande selon $P^+$ que selon $P^-$, c'est-à-dire si la différence entre ces deux logvraisemblances est positive. Sur une séquence S quelconque,

$$\Delta\mathcal{L}(\mathcal{S}) \quad = \mathcal{L}(\mathcal{S}|\mathcal{P}^+) - \mathcal{L}(\mathcal{S}|\mathcal{P}^-)$$

Nous possédons un fichier d'entraînement contenant des séquences d'îlots CpG chez la souris (mus_cpg_app.fa) et des séquences d'ADN de souris ne possédant pas d'îlots CpG (mus_temp_app.fa). Ces fichiers sont donnés au format fasta.

***Remarque :*** *L'estimation au Maximum de Vraisemblance nécessite le compte des mots de taille m+1 dans les séquences d'apprentissage. Un fichier au format fasta est un simple fichier* ***texte*** *de la forme suivante :*

*>nom sequence 1*
*AACCCTGGGC*
*>nom sequence 2*
*TTTTTTTTGCCCCGGTA*
*….*

1. *Afin d'effectuez l'apprentissage des paramètres des modèles pour chacune des régions, créer un programme* ***Compte*** *qui permet de compter les mots dans les séquences génomiques contenues dans un fichier au format fasta.*
2. *Construire un modèle M0 pour chaque type de séquences, noté* $P0^+$ *pour les séquences possédant des CpG et* $P0^-$ *pour les séquences sans CpG.*
3. *Construire ensuite des modèles M1, M2… pour chaque type de séquences (*$P1^+$, $P1^-$, $P2^+$, $P2^-$*…)*

## *B) Utilisation des modèles CpG et non CpG pour discriminer des séquences de jeu de données test*

Ces modèles étant construits, ils sont appliqués à deux jeux de séquences tests : un jeu de données contenant 1163 séquences d'îlots CpG chez la souris (mus_cpg_test.fa) et un second contenant 5137 séquences d'ADN de souris ne possédant pas d'îlots CpG (mus_tem_test.fa).

1. *Quel est l'intérêt de cette démarche ?*
2. *Créer un programme* ***Logvrais*** *qui permet de calculer la logvraisemblance d'une séquence génomique à partir d'un modèle de Markov.*
3. *Calculer le nombre Vrais Positifs, Faux Positifs, Vrais négatifs, Faux négatifs de chaque modèle* $P^+$ *et* $P^-$ *. Interpréter ces résultats. (Indication : vous pourrez par exemple étudier la spécificité et la sensibilité de la méthode.)*

4. *Au vu des résultats que vous avez obtenu que proposez vous comme modèle ?*

Maintenant, on souhaiterait étudier la succession des îlots CpG le long d'un chromosome entier de souris.

5. *Expliquez comment vous procéderiez pour répondre à cette question grâce aux modèles qui viennent d'être définis ?*

6. *Quelles sont les limites de cette méthode ?*

## 2. Segmentation d'une séquence en îlots CpG [1]

Certes, il est fort utile de pouvoir discriminer des séquences selon leurs propriétés, mais il serait encore plus utile de pouvoir partager de grandes séquences en de plus petites qui, elles, obéissent aux propriétés recherchées. Les chaînes de Markov à états cachés sont une technique pour faire cela.

Le critère de pertinence d'une telle chaîne à décrire une séquence sera pour nous le maximum de vraisemblance. Les probabilités au sein de chaque état sont déterminées comme précédemment, par apprentissage à partir de séquences déjà identifiées. Ainsi, dans notre cas, les probabilités des transitions au sein des deux états (îlots / non-îlots) sont celles que l'on a déjà calculées.

Les probabilités de transitions entre états sont calculées de la même manière : pour une transition à partir d'un état A vers un autre état (ou vers lui-même) on compte le nombre de passages entre ces deux états sur des séquences déjà segmentées, et l'on divise ce nombre par le nombre total de transitions effectuées à partir de cet état A. Dans les exemples que l'on étudie, on peut déterminer ces facteurs grâce aux longueurs des îlots par rapport aux longueurs qui séparent ces îlots. Les moyennes de ces valeurs sont données dans le tableau suivant.

| | îlots CpG | entre îlots |
|---|---|---|
| longueur moyenne | 1000 | 125000 |

### *A) Modélisation de la séquence:*

1. *Définir la problématique, les hypothèses et proposer une modélisation du problème.*

2. *On souhaite représenter la structure d'une séquence nucléique en îlots CpG par un modèle HMM (comme décrite ci-dessus).*

Dans un premier temps, vous vous intéresserez à un modèle HMM ou au sein de chaque état, vous aurez affaire à un modèle de type M1.

a) *Dessinez le modèle formel sous la forme d'un graphe représentant les changements d'états.*

*b) Notez les paramètres de transition et les probabilités d'émission correspondantes.*

## *B) Estimations des paramètres*

L'estimation des paramètres se fait par maximum de vraisemblance, qui nécessite le compte des mots de taille m+1 dans les séquences d'apprentissage.

a) **Probabilités d'émission :** Effectuez l'apprentissage de vos paramètres d'émission pour chacune des régions.

b) **Probabilités de transition** : Les probabilités de transitions entre états sont calculées de la même manière : pour une transition à partir d'un état A vers un autre état (ou vers lui-même) on compte le nombre de passages entre ces deux états sur des séquences déjà segmentées, et l'on divise ce nombre par le nombre total de transitions effectuées à partir de cet état A. Dans les exemples que l'on étudie, on peut déterminer ces facteurs grâce aux longueurs des îlots par rapport aux longueurs qui séparent ces îlots. Les moyennes de ces valeurs sont données dans le tableau suivant.

| | îlots CpG | entre îlots |
|---|---|---|
| longueur moyenne | 1000 | 125000 |

*1. Déterminer les probabilités de transitions entre états.*

## *C) Chemin optimal et discussion*

1. *Programmer l'algorithme de Viterbi [2], qui est un algorithme de segmentation de séquences à partir d'un modèle HMM. Faites attention, les séquences étudiées peuvent être très grandes.*
2. *En utilisant l'algorithme de Viterbi, déterminez quelle est la structure des séquences tests (mus1.fa, mus2.fa, mus3.fa) la plus probable étant donné le modèle estimé.*
3. *Recommencer en utilisant un modèle HMM ou au sein de chaque état, vous aurez affaire à un modèle de type M2, M3 …*
4. *Conclusion : Comparer les résultats obtenus lors des questions 2 et 3 ? Qu'en concluez vous ?*

5. *Vous pourrez également discuter de la pertinence de la base de données choisie, et du problème des séquences d'apprentissage en général.*

## 3. Références

1. R. Durbin, S. Eddy, A. Krogh, G. Mitchison, Biological sequence analysis, Probabilistic models of proteins and nucleic acids. Cambridge.

2. R. Shamir, 2001, Hidden Markov Models

## 4. Quelques indications

### *A) Indication programmation*

Pour les fonctions de bases pour la lecture de fichier fasta par exemple, il est possible d'utiliser la librairie ***seqinr***

*# Lecture d'un fichier fasta :*

*seq=read.fasta(file= "mus_cpg_app.fa")*

*# Donne le nombre de séquences dans le fichier*

*length(seq)*

*# Extraction de la première séquence du fichier*

*s1=seq[[1]]*

*# Fonction compte :*

*count(s1,1)*

*a c g t*

*92 131 39 252*

*count(s1,2)*

*aa ac ag at ca cc cg ct ga gc gg gt ta tc tg tt*

*17 22 4 49 19 25 6 81 9 8 2 20 47 76 27 101*

## *B) Indication statistique*

Sensibilité = VP/(VP+FN) et Spécificité = VN/(VN+FP)