# Mini-mémoire
# Notes articles HMM

Jérôme Hellinckx

11 décembre 2015

## Table des matières



## Glossaire

**chaîne de Markov** Processus stochastique subissant des transitions d'un état à l'autre dans l'espace des états et où la probabilité d'entrer à l'état suivant $j$ dépend uniquement de l'état courant $i$.

**probabilité d'émission** Probabilité qu'un état $i$ émette un symbole $x$. C'est donc la distribution des probabilités pour chaque symbole par état. Dénoté $e(x \mid i)$.

**probabilité de l'état initial** Pour tout état $i$, probabilité que $y_1$ prenne $i$ comme valeur. Dénoté $\pi(i)$.

**probabilité de transition** Probabilité de faire une transition d'un état $i$ à un état $j$. Dénoté $t(i, j)$.

**symbole** Événement observable d'un MMC.

**état** Facteur interne non-observable d'un MMC.

# Article 1: *Hidden Markov Models and their Applications in Biological Sequence Analysis*

## 1.1 Définition d'un MMC

Un *modèle de Markov caché* est un modèle statistique qui peut être utilisé pour décrire l'évolution d'événements observables qui sont dépendants de facteurs internes non-observables. On appelle l'événement observé un *symbole* et le facteur invisible sous-jacent l'observation un *état*. Un MMC consiste donc en deux processus stochastiques, d'une part un processus invisible d'états cachés, d'autre part un processus visible de symboles observables. Les états cachés forment une *chaîne de Markov* et la distribution de probabilité des symboles observés dépend de l'état sous-jacent.

Formellement, on dénote la séquence de symboles observés par $\mathbf{x} = x_1 x_2 \dots x_L$ et la séquence d'états sous-jacents par $\mathbf{y} = y_1 y_2 \dots y_L$ où $y_n$ est l'état sous-jacent de la $n-$ième observation $x_n$. Chaque symbole $x_n$ prend un nombre fini de valeurs possibles de l'ensemble d'observations $O = \{O_1, O_2, \dots, O_N\}$, et chaque état $y_n$ prend une valeur de l'ensemble d'états $S = \{1, 2, \dots, M\}$, où $N$ et $M$ dénotent respectivement le nombre d'observations distinctes et le nombre d'états distincts.
On suppose que la séquence d'états finis est une *chaîne de Markov en temps homogène de premier ordre*, ce qui implique que la probabilité d'entrer à l'état $j$ en $y_{n+1}$ dépend uniquement de l'état courant $i$ en $y_n$ et que cette probabilité ne change pas au cours du temps. Ainsi, on a

$$P\{y_{n+1} = j \mid y_n = i\} = t(i, j) \tag{1}$$

$\forall i, j \in S, \forall n \geq 1$. La probabilité de faire une transition d'un état $i$ à un état $j$ est appelée *probabilité de transition* qu'on dénote $t(i, j)$. À l'état initial $y_1$, on dénote la *probabilité de l'état initial* par $\pi(i) = P\{y_1 = i\}, \forall i \in S$. La probabilité que la $n-$ième observation soit $x_n = x$ dépend uniquement de l'état sous-jacent $y_n$, par conséquent

$$P\{x_n = x \mid y_n = i\} = e(x \mid i) \tag{2}$$

$\forall x \in O, \forall i \in S, \forall n \geq 1$. C'est ce qu'on appelle la *probabilité d'émission* de $x$ à l'état $i$, et on la dénote par $e(x \mid i)$.
Ces trois mesures de probabilités $t(i, j)$, $\pi(i)$, et $e(x \mid i)$ spécifient complètement un MMC. On dénote l'ensemble de ces paramètres par $\Theta$.

On peut dès lors calculer la probabilité que le MMC générera la séquence d'observations $\mathbf{x} = x_1 x_2 \dots x_L$ avec la séquence d'états sous-jacente $\mathbf{y} = y_1 y_2 \dots y_L$. On a donc une probabilité jointe $P\{\mathbf{x}, \mathbf{y} \mid \Theta\}$ qui peut être calculée par

$$P\{\mathbf{x}, \mathbf{y} \mid \Theta\} = P\{\mathbf{x} \mid \mathbf{y}, \Theta\} P\{\mathbf{y} \mid \Theta\} \tag{3}$$

où

$$P\{\mathbf{x} \mid \mathbf{y}, \Theta\} = e(x_1 \mid y_1) e(x_2 \mid y_2) e(x_3 \mid y_3) \dots e(x_L \mid y_L) \tag{4}$$

$$P\{\mathbf{y} \mid \Theta\} = \pi(y_1) t(y_1, y_2) t(y_2, y_3) \dots t(y_{L-1}, y_L). \tag{5}$$

## 1.2 Les 3 problèmes basiques d'un MMC

### 1.2.1 Problème d'évaluation

Comment calculer la probabilité d'observation $P\{\mathbf{x} \mid \Theta\}$ en se basant sur un MMC donné ? On pourrait considérer toutes les séquences d'états $\mathbf{y}$ possibles pour $\mathbf{x}$ donné et additionner les probabilités

$$P\{\mathbf{x} \mid \Theta\} = \sum_{y} P\{\mathbf{x}, \mathbf{y} \mid \Theta\}. \tag{6}$$

Ceci est cependant très couteux en calcul puisqu'il existe $M^L$ séquences d'états possibles.

Le *forward algorithm* est un algorithme de programmation dynamique calculant efficacement $P\{\mathbf{x} \mid \Theta\}$. Cet algorithme définit une *forward variable* $f(n, i)$ déterminant la probabilité d'observer la séquence partielle de symboles $x_1 \dots x_n$ et d'arriver à l'état $i$ pour $y_n$, étant donné $\Theta$

$$f(n, i) = P\{x_1 \dots x_n, y_n = i \mid \Theta\}. \tag{7}$$

En $n = 1$, $f(1, i) = P\{x_1, i \mid \Theta\}$. Or cette probabilité d'observer $x_1$ avec $y_1 = i$ revient à joindre la probabilité d'émission de $x_1$ par $i$ et la probabilité initiale de l'état $i$. Ensuite, pour $n = 2, \dots, L$, afin de calculer la probabilité d'observer $x_1, \dots, x_{n-1}, x_n$ et d'arriver à l'état $i$, il suffit, pour chaque $j$ en $n - 1$, de récupérer son $f(n - 1, j)$ associé (probabilité d'observer $x_1, \dots, x_{n-1}$ et d'arriver à $j$) en y joignant la probabilité de transition $t(j, i)$ puisqu'on passe de $j$ à $i$ ainsi que le probabilité d'émission $e(x_n \mid i)$ étant donné qu'on observe $x_n$ avec $i$, et de tous les additionner. On en déduit la formule récursive

$$f(n, i) = \begin{cases} \pi(i)e(x_1 \mid i), & n = 1. \\ \sum_{j=1}^{M} [f(n-1, j)t(j, i)e(x_n \mid i)], & n = 2, \dots, L. \end{cases} \tag{8}$$

On finit par trouver la probabilité d'observation de $\mathbf{x}$ en additionnant les probabilités d'observer $x_1, \dots x_L$ et d'arriver à chaque état $i$

$$P\{\mathbf{x} \mid \Theta\} = \sum_{i=1}^{M} f(L, i). \tag{9}$$

La complexité de cet algorithme est seulement $\mathcal{O}(LM^2)$.

Les mêmes résultats peuvent être obtenus en utilisant un algorithme fonctionnant dans l'autre sens, appelé *backward algorithm*. Cet algorithme définit une *backward variable* donnant la probabilité d'observer la séquence de symbole $x_{n+1} \dots x_L$ après être arrivé à un état $i$ en $n$

$$b(n, i) = P\{x_{n+1} \dots x_L \mid y_n = i, \Theta\}. \tag{10}$$

Cette variable $b(n, i)$ peut être calculée récursivement comme suit

$$b(n, i) = \begin{cases} 1, & n = L. \\ \sum_{j=1}^{M} [t(i, j)e(x_{n+1} \mid j)b(n + 1, j)], & n = L - 1, L - 2, \dots, 1. \end{cases} \tag{11}$$

On obtient alors la probabilité d'observation de $\mathbf{x}$

$$P\{\mathbf{x} \mid \Theta\} = \sum_{i=1}^{M} b(1, i)\pi(i)e(x_1 \mid i). \tag{12}$$

### 1.2.2 Problème de décodage

Étant donné une séquence de symbole $\mathbf{x}$ et un modèle $\Theta$, quelle est la séquence d'états $\mathbf{y}$ qui explique au mieux la séquence de symboles observés ? On recherche donc le chemin [d'états] optimal $\mathbf{y}^*$ qui maximise la probabilité d'observation de la séquence de symbole $\mathbf{x}$. Soit formellement

$$\mathbf{y}^* = \max_y P\{\mathbf{y} \mid \mathbf{x}, \Theta\}. \tag{13}$$

L'*algorithme de Viterbi* permet de trouver un tel chemin. Cet algorithme définit la variable $\gamma(n, i)$ qui est le score maximum le long d'un chemin d'état $y_1 \dots y_{n-1}$ arrivant à l'état $i$ en $y_n$ et émettant les symboles $x_1 \dots x_n$. C'est-à-dire formellement

$$\gamma(n, i) = \max_{y_1,\dots,y_{n-1}} P\{x_1 \dots x_n, y_1 \dots y_{n-1} y_n = i \mid \Theta\}, \tag{14}$$

calculée récursivement selon la formule

$$\gamma(n, i) = \begin{cases} \pi(i) e(x_1 \mid i), & n = 1. \\ \max\limits_j [\gamma(n-1, j) t(j, i) e(x_n \mid i)], & n = 2, \dots, L. \end{cases} \tag{15}$$

On conclut en obtenant la probabilité d'observation maximale comme suit

$$P^* = \max_y P\{\mathbf{x}, \mathbf{y} \mid \Theta\} = \max_i \gamma(L, i), \tag{16}$$

et on retrouve le chemin d'états optimal $\mathbf{y}^*$ en remontant les récursions qui ont menées à la probabilité maximale. L'algorithme de Viterbi trouve la séquence d'états optimale en un temps $\mathcal{O}(LM^2)$.
Il est à noter que les probabilités peuvent devenir de très petits nombres *float* pour de longues séquences, menant potentiellement à des problèmes pour la représentation machine. Ceci peut être résolu en changeant les probabilités en logarithmes de probabilités.

### 1.2.3 Problème d'entrainement

Soit un ensemble de séquences de symboles observés $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_T\}$ qu'on souhaite représenter par un MMC. Comment peut-on choisir de manière raisonnable et adéquate les paramètres $\Theta = (\pi(i), t(i, j), e(x \mid i))$ du MMC en se basant sur ces observations ? Bien qu'il n'existe pas de manière optimale d'estimer les paramètres à partir d'un nombre limité de séquences finies de symboles, il est possible de trouver des paramètres maximisant localement la probabilité d'observation (*Baum-Welch*, *stochastic EM*).

# Article 2: *A linear memory algorithm for Baum-Welch training*

## 2.1 Entrainement *Baum-Welch*

Rappelons que le problème d'entrainement est de trouver un ensemble de paramètres $\Theta = (\pi(i), t(i, j), e(x \mid i))$ satisfaisant pour un MMC. L'algorithme Baum-Welch définit une procédure itérative dans laquelle les probabilités d'émission $e(x \mid i)$ et de transition $t(i, j)$ dans l'itération $n+1$ sont mises au nombre de fois que chaque transition et émission est attendue à être utilisée lors de l'analyse de la séquence d'entrainement avec l'ensemble des probabilités d'émission et de transition dérivées dans l'itération précédente $n$.

# Article 3: *Implementing EM and Viterbi algorithms for Hidden Markov Model in linear memory*

## 3.1 Définition d'un MMC

Un MMC peut être décrit comme une machine à état fini stochastique pour laquelle chaque transition entre états cachés se termine par une émission de symbole. Un MMC peut être représenté par un graphe dirigé avec $N$ états où chaque état peut émettre soit un caractère discret, soit une valeur continue tirée depuis une fonction de densité de probabilité ($PDF$). Les paramètres suivants décrivent l'implémentation conventionnelle d'un MMC :

- Un ensemble d'états $S = \{S_1, \ldots, S_N\}$ avec $q_t$ l'état visité à l'instant $t$ ;
- Un ensemble de fonctions de densité de probabilité $B = \{b_1(o), \ldots, b_N(o)\}$, décrivant les probabilités d'émission $b_j(o_t) = P(o_t \mid q_t = S_j)$ pour $1 \leq j \leq N$, où $o_t$ est l'observation au temps $t$ de la séquence d'observations $O = \{O_1, \ldots, O_T\}$ ;
- Une matrice de probabilités de transition [d'états] $A = \{a_{i,j}\}$ pour $1`i, j \leq N$, où $a_{i,j} = P(q_{t+1} = S_j \mid q_t = S_i)$ ;
- Un vecteur de distributions initiales d'états $\Pi = \{\pi_1, \ldots, \pi_N\}$.

# Article 4: *Using hidden Markov models to analyze gene expression time course data*

## 4.1 Définition formelle d'un problème de *clustering*

Soit $n$ séquences $O^i$ pouvant être de longueur différente, avec un ensemble d'indices $\mathcal{I} = \{1, 2, \ldots, n\}$ et un entier fixé $K \ll n$. Calculer une partition $\mathcal{C} = (C_1, C_2, \ldots, C_K)$ de $\mathcal{I}$ et les MMCs $\lambda_1, \ldots, \lambda_K$ qui maximisent la fonction

$$f(\mathcal{C}) = \prod_{k=1}^{K} \prod_{i \in C_k} L(O^i \mid \lambda_k), \tag{17}$$

où $L(O^i \mid \lambda_k)$ dénote la *likelihood function*, la densité de probabilité de générer la séquence $O^i$ avec le modèle $\lambda_k : (O^i \mid \lambda_k) := P(O^i \mid \lambda_k)$.

## 4.2 Proposition de résolution

L'approche de la vraisemblance maximum (*maximum likelihood*) pour résoudre un *cluster problem* de MMC est proposée, étant donnée une collection $K$ initiale de MMCs $\lambda_1^0, \ldots, \lambda_K^0$.

1. Itération $t \in \{1, 2, \ldots\}$ :
   (a) Générer un nouveau partitionnement des séquences en assignant chaque séquence $O^i$ au modèle $k$ pour lequel la vraisemblance $L(O^i \mid \lambda_k^{t-1})$ est maximale.
   (b)
   Calculer les nouveaux paramètres $\lambda_1^t, \ldots, \lambda_K^t$ en utilisant un algorithme de réestimation avec leur paramètres précédents $\lambda_1^{t-1}, \ldots, \lambda_K^{t-1}$ et leurs séquences assignées.
2. Stop si (1) l'amélioration de la fonction est plus petite qu'une limite donnée $\varepsilon$, (2) le groupement des séquences est inchangé ou (3) la limite d'iterations est atteinte.

### 4.3 Déterminer le nombre de *clusters*

Déterminer le nombre de *clusters* pose un problème particulier dans ce cas puisqu'il faut ici spécifier une *collection* de modèles. Généralement on choisit des modèles initiaux en se basant sur des prototypes de comportement. Le nombre de modèles suit ensuite deux règles simples.

1. Si un cluster contient très peu de profils, supprimer le cluster et réassigner les profils aux clusters restants.
2. Si un cluster contient trop de profils, le séparer en deux. Pour cela, on copie le modèle en changeant aléatoirement les paramètres des copies de façon uniforme et indépendante.

### 4.4 Données manquantes

On ajoute aux probabilités d'émission une valeur constante représentant la fréquence de données manquantes. Cette constante n'est pas modifiée par l'étape Baum-Welch dans l'algorithme de clustering.

## Article 5: *Robust inference of groups in gene expression time-courses using mixtures of HMMs*