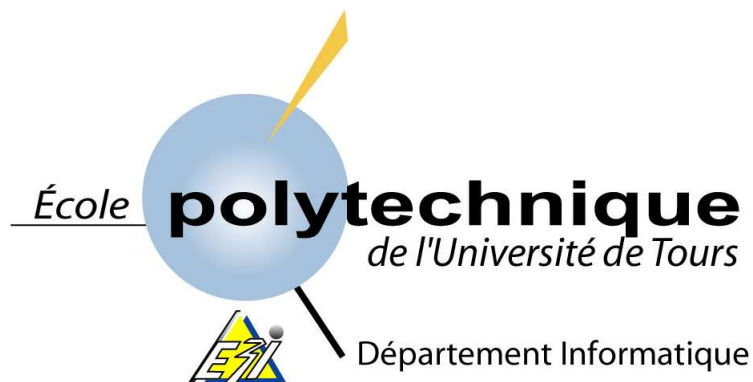


UNIVERSITE François Rabelais TOURS
Polytech'Tours-Département Informatique
64, Avenue Jean Portalis
37200 TOURS



Projet de Mathématiques : Chaînes de Markov cachées Algorithme de Baum-Welch

Encadré par :

M. SLIMANE, S. AUPETIT
Polytech' Tours/Département Informatique/Université Tours
64, avenue Jean Portalis
37200 TOURS

Présenté par :

Florian AGEN, Julien MICHOT
Promotion 2004-2007

Projet de Mathématiques : Chaînes de Markov cachées,
Algorithme de Baum-Welch

Florian AGEN - Julien MICHOT

Janvier 2005

Table des matières

1	Présentation générale	3
1.1	Les processus stochastiques	3
1.2	Les chaînes de Markov	4
1.3	Les chaînes de Markov cachées	5
2	L'algorithme de Baum-Welch	7
2.1	L'algorithme Forward	8
2.2	L'algorithme Backward	9
2.3	L'algorithme de Baum-Welch	10
2.4	Améliorations possibles	11
3	Présentation des programmes	13
4	Tests et résultats	14
5	Annexe1 : Interface DOS	22
6	Annexe2 : Interface FOX	23

Introduction

Les chaînes de Markov cachées (CMC) sont des processus stochastiques simples en applications, riches en propriétés et fondées sur des bases mathématiques solides. Elles permettent la modélisation de plusieurs phénomènes physiques.

Leur utilisation dans le domaine informatique est très répandue. A la fois novatrice dans la recherche associée à la parole ou à la reconnaissance des formes, les CMC sont aussi liées au développement des nouveaux modes de communication.

Plusieurs méthodes existent quant à la réestimation des paramètres d'une chaîne de Markov cachée. Nous présentons ici l'algorithme de Baum-Welch, proposé par ses auteurs dès 1967.

Notre but dans le cadre de ce projet de mathématiques sera d'explicitier le fonctionnement des modèles markoviens cachés, à travers une présentation générale dans un premier temps, puis par l'utilisation de l'algorithme implémenté. Les tests effectués sont fournis dans une dernière partie.

Chapitre 1

Présentation générale

Les processus stochastiques permettent de modéliser des systèmes dont le comportement n'est que partiellement prévisible. La théorie est fondée sur le calcul de probabilités et statistiques qu'il nous faut expliciter avant de définir le modèle markovien.

1.1 Les processus stochastiques

Un processus stochastique est une fonction, ou plus généralement une application $X(\omega, t)$, définie dans l'ensemble fondamental Ω à valeurs dans $F(t)$, ensemble des fonctions d'une variable t .

L'évolution d'un processus stochastique est une suite de transitions d'états : $s_0 s_1 \dots s_T$, pour laquelle on note s_0 l'état du processus à l'instant 0. Sa loi d'évolution est obtenue à l'aide de la probabilité $Pr(s_0 \dots s_T)$ définie successivement de la manière suivante :

$$\begin{aligned} Pr(s_0 \dots s_T) &= Pr(s_0 \dots s_{T-1}) \times Pr(s_T | s_0 \dots s_{T-1}) \\ &= Pr(s_0) \times Pr(s_1 | s_0) \times Pr(s_2 | s_0 s_1) \times \dots \times Pr(s_T | s_0 \dots s_{T-1}) \end{aligned} \quad (1.1)$$

La caractérisation du processus se résume donc par l'obtention des probabilités initiales $Pr(s_0)$ et des probabilités des états conditionnés par leurs évolutions antérieures. La loi de probabilité des états, à un instant t , dépend de l'histoire du processus qui garde la mémoire de son passé.

L'*espace des états* S est l'ensemble dénombrable des valeurs prises par l'ensemble des variables aléatoires du processus stochastique. Ces valeurs, tout comme celles prises dans l'*espace du temps* T , peuvent être discrètes ou continues, ce qui permet de les classer respectivement par rapport à ω et t :

- T et S sont continus : $X(\omega, t)$ est continu, on parle alors de **processus de renouvellement ou de diffusion**.
- T est continu, S est discret : $X(\omega, t)$ discontinu en ω , pour l'étude des **files d'attente**.

- T est discret, S est continu : $X(\omega, t)$ discontinu en t, pour l'étude des **séries temporelles**.
- T et S sont discrets : $X(\omega, t)$ est discontinu en ω et en t, ce sont les **processus markovien** ou **chaînes de Markov** qui nous intéressent particulièrement.

1.2 Les chaînes de Markov

Une chaîne de Markov est un processus stochastique qui vérifie les propriétés suivantes :

1. $X(\omega, t)$ ne change éventuellement de valeurs qu'à des instants déterminés (l'espace des temps étant discret), instants que l'on pourra toujours identifier par leurs indices. Le plus souvent, on note $X(\omega, t)$ simplement X_t .
2. L'espace des états S associé à $X(\omega, t)$ est fini et discret pour t fixé. Ainsi, X_t ne peut prendre que l'une des valeurs possibles $x_1, x_2, x_3, \dots, x_M$ pour un système à M états.
3. $X(\omega, t)$ possède la **propriété markovienne** : X_t ne dépend que du dernier état connu que l'on peut exprimer par :

$$Pr[X_t = j / X_{t-1} = i_{t-1} \dots X_{t-2} = i_{t-2} \dots X_0 = i_0] = Pr[X_t = j / X_{t-1} = i] \quad (1.2)$$

la probabilité de transition de l'état i_{t-1} à l'état j pour laquelle le dernier état connu l'est à l'instant t-1 (instant sans mémoire).

4. Une chaîne de Markov est homogène dans le temps si les probabilités de transition sont indépendantes. Elle est définie par la donnée des probabilités de transition des états :

$$P_{ij} = P(X_t = j / X_{t-1} = i), \forall (i, j) \in S^2 \quad (1.3)$$

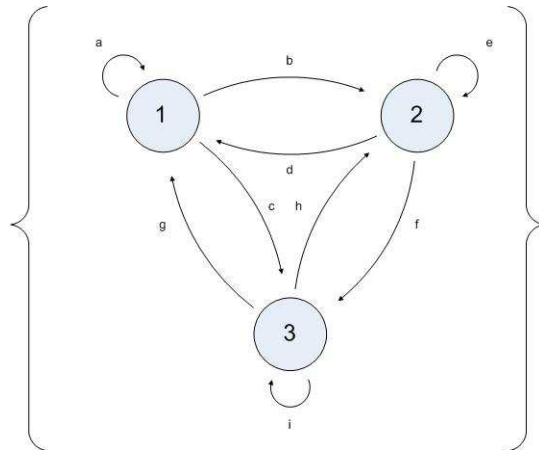
On associe à la chaîne de Markov $\{X_t\}$ un graphe G dont l'ensemble des sommets est une bijection avec l'ensemble des états S, et dont l'ensemble des arcs U orienté dans le sens de transition est défini par :

$$(i, j) \in U \iff P_{ij} = P(X_t = j / X_{t-1} = i) > 0 \quad (1.4)$$

Ainsi, pour un modèle de Markov M présentant une matrice de transition :

$$P = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix}$$

Le graphe G associé sera de la forme :



Cette modelisation permet de visualiser de manière optimale les probabilités de transition des différents états.

1.3 Les chaînes de Markov cachées

Les chaînes de Markov cachées (CMC) sont des modèles markoviens dont les états sont cachés.

Il existe plusieurs types de CMC. La première distinction se fait suivant la nature de la fonction de densité de probabilité utilisée pour la génération des observations. Lorsque la distribution est directement obtenue par quantification, les CMC sont qualifiés de *discrets*. L'utilisation d'une distribution continue, généralement approximée par une mixture de Gaussienne, conduit à des CMC *continus*. Il existe généralement un compromis entre ces deux familles appelé *CMC semi-continu*. En effet, l'utilisation d'une quantification induit une perte d'information qui peut être préjudiciable aux modèles. D'un autre côté, le passage à une modélisation continue conduit à une augmentation importante du nombre de paramètres à estimer. Les CMC semi-continus sont une alternative permettant d'optimiser le nombre global de paramètres du modèle.

Une autre distinction est effectuée entre les CMC, suivant le mode d'émission des observations. Généralement les observations sont produites par les états du modèle, on parle alors de *modèles d'états*. Il est cependant possible de considérer l'émission des observations lors du franchissement des transitions, il s'agit alors de *modèles d'arcs*. Le choix est guidé par l'application. Nous pouvons cependant mentionner qu'à nombre égal d'états, les modèles d'arcs permettent un plus grand nombre de possibilités quant à l'émission d'observations. Lors de la modélisation d'un phénomène par un modèle d'arcs, il peut être intéressant de permettre le franchissement de transitions sans émission d'observations, en particulier pour modéliser l'absence d'un évènement.

On définit différents éléments pour une CMC :

- \mathbf{N} est le nombre d'états cachés du modèle. On note $\mathbf{S} = \{ s_1, s_2, \dots, s_N \}$ l'ensemble des états cachés. A l'instant t un état est représenté par q_t ($q_t \in \mathbf{S}$);
- \mathbf{M} le nombre de symboles distincts que l'on peut observer dans chaque état. On les représente par l'ensemble $\mathbf{V} = \{ v_1, v_2, \dots, v_M \}$. A l'instant t un symbole observable est désigné par o_t ($o_t \in \mathbf{V}$);
- Une matrice de probabilité de transitions, notée $\mathbf{A} = [a_{ij}]$, où a_{ij} est la probabilité a priori de transition de l'état i vers l'état j . Dans une CMC stationnaire du 1^{er} ordre cette probabilité ne dépend pas de t . On définit $a_{ij} = P(q_{t+1} = s_j \parallel q_t = s_i)$, $1 \leq i, j \leq N$;
- Une matrice de distributions des probabilités, notée $\mathbf{B} = [b_j(k)]$, associée à chaque état où $b_j(k)$ est la probabilité d'observer le symbole v_k en étant à l'état s_j à l'instant t . On définit $b_j(k) = P(o_t = v_k \parallel q_t = s_j)$, $1 \leq i \leq N$, $1 \leq k \leq M$;
- Un vecteur $\mathbf{\Pi} = [\pi_i]$ de distributions des probabilités de transitions initiales, où π_i est la probabilité de commencer dans l'état i . On définit $\pi_i = P[q_1 = s_i]$ avec $1 \leq i \leq N$.

On peut dire qu'une CMC notée λ est définie complètement par $\lambda = (\mathbf{A}, \mathbf{B}, \mathbf{\Pi})$ (\mathbf{N} et \mathbf{M} sont sous entendus dans les matrices \mathbf{A} et \mathbf{B} ainsi que le vecteur $\mathbf{\Pi}$).

La génération des observations dans une CMC se fait par la procédure suivante :

1. $t = 1$ choix de l'état initial, $q_1 = s_i$ avec la probabilité π_i ;
2. Choix de l'observation $o_t = v_k$, avec la probabilité $b_i(k)$;
3. Transition vers le nouvel état $q_{t+1} = s_j$ avec la probabilité a_{ij} ;
4. $t = t+1$; si $t \leq T$, alors retour à l'étape 2, sinon fin de procédure (avec T la longueur d'une suite d'observations).

Afin de pouvoir exploiter le modèle λ , trois problèmes de base doivent être résolus :

1. **Le problème d'évaluation** : soit une séquence d'observations $O = o_0, o_1, \dots, o_{T-1}$ et un modèle $\lambda = (\mathbf{A}, \mathbf{B}, \mathbf{\Pi})$, comment calculer efficacement la probabilité de cette séquence étant donné le modèle $Pr(O|\lambda)$?
2. **Le problème de décodage ou de reconnaissance** : étant donné une séquence d'observations $O = o_0, o_1, \dots, o_{T-1}$ et un modèle $\lambda = (\mathbf{A}, \mathbf{B}, \mathbf{\Pi})$, comment trouver la séquence optimale d'états $Q = q_0, q_1, \dots, q_T$ qui a produit la séquence O ?
3. **Le problème de ré-estimation ou d'apprentissage** : étant donné un ensemble de séquences d'observations et un modèle initial λ , comment ré-estimer les différents paramètres du modèle afin d'augmenter la vraisemblance d'apparition de la séquence d'observation. La maximisation de la vraisemblance d'un modèle probabiliste est aujourd'hui effectuée par des algorithmes efficaces, comme l'algorithme EM (Expectation Maximization) ou l'algorithme de Baum-Welch, que nous allons présenter dans une deuxième partie.

Chapitre 2

L'algorithme de Baum-Welch

De part le caractère caché des états d'un modèle Markovien, il n'est pas rare que les différents paramètres du modèle (la matrice de transition des états cachés, la matrice de génération des symboles suivant les états, ainsi que la matrice de probabilité de départ) soient eux aussi à déterminer, à estimer. Un algorithme particulièrement efficace dans l'estimation des paramètres d'une chaîne de Markov cachée est l'algorithme de Baum-Welch.

L'algorithme de Baum-Welch est dérivé de l'algorithme EM (Expectation Maximization). Dans ce dernier, l'objectif est de maximiser (ou minimiser) une probabilité du type $P(X/M)$, d'un modèle probabiliste (Markovien dans notre cas).

Une maximisation en mathématiques est généralement la solution de l'équation :

$$\frac{\partial P(X/M)}{\partial \lambda} = 0 \quad (2.1)$$

où λ est le vecteur de variables paramètres du modèle M .

La résolution de ce type d'équation est dans la pratique difficile, voire impossible à obtenir.

L'utilisation d'algorithmes itératifs, comme l'algorithme EM par exemple, permet néanmoins d'estimer de manière assez précise et rapide la solution optimale de l'équation.

L'algorithme de Baum-Welch est un algorithme à apprentissage. Etant donné un ensemble de séquences d'observations O et un modèle initial λ , l'algorithme de Baum-Welch entreprend une ré-estimation des paramètres du modèle de manière à augmenter la vraisemblance de génération des séquences d'observations. La maximisation de la vraisemblance $P(O/\lambda)$ peut donc se voir comme l'optimisation du modèle λ sachant que l'on a observé la ou les séquences O .

Afin de maximiser rapidement la vraisemblance $P(O/\lambda)$ du modèle, il est nécessaire d'utiliser un algorithme de calcul (de $P(O/\lambda)$) de complexité faible. Or, la manière la plus évidente pour déterminer $P(O/\lambda)$ est la suivante :

$$P(O/\lambda) = Pr(S_1 S_2 S_3 \dots S_T / \lambda \wedge O) + Pr(S_2 S_2 S_3 \dots S_{T+1} / \lambda \wedge O) + Pr(S_i S_j S_k \dots S_z / \lambda \wedge O) + etc \dots \quad (2.2)$$

avec $i, j, \dots, z \in 1 \dots N$.

Il nous faut donc additionner tous les arrangements avec répétition (de T termes) possibles des S_i , avec $i \in 0 \dots N$. Le nombre de séquences différentes (arrangements avec répétition)

étant alors de N^T . Ce qui revient à écrire :

$$P(O/\lambda) = \sum_{\text{Arrangements des } q_i} \pi_{q_1} b_{q_1}(o_1) a_{q_1 q_2} b_{q_2}(o_2) \dots a_{q_{T-1} q_T} b_{q_T}(o_T) \text{ avec } q_i \in S = \{S_1 \dots S_N\} \quad (2.3)$$

Cependant, cette formule possède une complexité en $2TN^T$, bien trop importante. Aussi, l'algorithme "Forward-Backward" a été conçu pour diminuer la complexité de calcul.

2.1 L'algorithme Forward

Le premier algorithme utilisé par l'algorithme Baum-Welch est l'algorithme Forward. Cet algorithme délivre deux informations : $P(O/\lambda)$ et $\alpha_t(i)$ où $\alpha_t(i)$ est la probabilité de la suite d'observations partielle $(o_1 o_2 \dots o_t)$, se terminant (à l'instant t) à l'état S_i

$$\alpha_t(i) = Pr(o_1 o_2 \dots o_t, q_t = S_i | \lambda) \quad (2.4)$$

et $P(O/\lambda)$, la probabilité d'apparition de la (ou des) séquence(s) observée(s) O , avec le modèle courant λ , (valeur à optimiser)

$P(O/\lambda)$ est telle que :

$$P(O/\lambda) = \sum_{i=1}^N \alpha_T(i) \quad (2.5)$$

L'algorithme Forward se détermine ainsi :

Pour $i = 1$ à N **Faire**

$$\alpha_1(i) = \pi_i b_i(o_1)$$

FinPour

Pour $t = 1$ à $T-1$ **Faire**

Pour $j = 1$ à N **Faire**

$$\alpha_{t+1}(j) = \left(\sum_{i=1}^N \alpha_t(i) a_{ij} \right) b_j(o_{t+1})$$

FinPour

FinPour

$$P(O/\lambda) = \sum_{i=1}^N \alpha_T(i)$$

La procédure Forward détermine $P(O/\lambda)$ en utilisant exactement la même formule (2.3) développée précédemment, mais de manière inductive, ce qui diminue fortement le temps et le nombre de calculs. En effet, l'algorithme Forward place dans un premier temps, dans la première 'ligne' α_1 , la probabilité d'obtenir l'état caché i sachant que l'on a observé le symbole o_1 (donc $\alpha_1(i) = \pi_i b_i(o_1)$, avec π_i la probabilité d'avoir l'état i en premier, et $b_i(o_1)$ la probabilité d'observer o_1 lorsque l'état i est 'apparu').

Par la suite, l'algorithme détermine la t ième ligne suivante, en s'appuyant sur la $t-1$ ième ligne de la matrice α . L'induction permet ainsi d'obtenir $\alpha_{t+1}(j)$, tel que $\alpha_{t+1}(j)$ soit égale à la

somme des probabilités d'avoir observé les t premiers symboles suivant tous les chemins des états cachés possibles, et de passer à l'état j , en observant le symbole o_{t+1} à l'instant $t+1$. La matrice α ainsi obtenue condense les calculs importants et redondants, qui seront nécessaires par la suite, dans l'algorithme de Baum-Welch.

De plus, il est important de remarquer que la somme des termes de la dernière ligne de la matrice α représente la probabilité recherchée $P(O/\lambda)$, puisque la dernière ligne représente la chaîne entièrement observée $(o_1 \dots o_T)$.

2.2 L'algorithme Backward

Le deuxième algorithme utilisé par l'algorithme Baum-Welch est l'algorithme Backward. Cet algorithme délivre une seule information : $\beta_t(i)$

$\beta_t(i)$ peut se voir comme la probabilité d'observer la suite partielle $(o_{t+1}o_{t+2} \dots o_T)$, qui commence à l'instant $t+1$ et se termine (à l'instant t) à l'état S_i

$$\beta_t(i) = Pr(o_{t+1}o_{t+2} \dots o_T, q_t = S_i | \lambda) \quad (2.6)$$

Cet algorithme permet aussi de calculer $P(O/\lambda)$ puisque

$$P(O/\lambda) = \sum_{i=1}^N \beta_1(i) \quad (2.7)$$

L'algorithme Backward se détermine ainsi :

Pour $i = 1$ à N **Faire**

$$\beta_T(i) = 1$$

FinPour

Pour $t = T-1$ à 1 **Faire**

Pour $i = 1$ à N **Faire**

$$\beta_t(i) = \sum_{j=1}^N a_{ij} \beta_{t+1}(j) b_j(o_{t+1})$$

FinPour

FinPour

L'algorithme Backward est conçu de manière similaire à l'algorithme Forward. Une seule chose diffère : l'ordre des lignes. En effet, contrairement au Forward, la dernière ligne de la matrice β , correspond à l'observation du premier symbole o_1 , la deuxième ligne pour l'observation des 2 premiers symboles o_1o_2 et ainsi de suite jusqu'à la première ligne où la séquence entière est considérée. L'algorithme fonctionne également de manière inductive et peut aussi fournir une estimation correcte de $P(O/\lambda)$.

A partir des variables $\alpha_t(i)$ et $\beta_t(i)$, nous sommes en mesure de calculer la vraisemblance

$P(O/\lambda)$ de la séquence d'observation O , pour le modèle λ , à chaque instant t :

$$P(O/\lambda) = \sum_{i=1}^N \alpha_t(i)\beta_t(i) \quad (2.8)$$

puisque,

$$\alpha_t(i) = P(o_1 \dots o_t, q_t = S_i | \lambda) \quad (2.9)$$

$$\beta_t(i) = P(o_{t+1} \dots o_T, q_t = S_i | \lambda) \quad (2.10)$$

on en déduit

$$\alpha_t(i)\beta_t(i) = P(o_1 \dots o_T, q_t = S_i | \lambda) \quad (2.11)$$

on peut alors exprimer $P(O/\lambda)$ en sommant ce produit sur l'ensemble des états cachés de la CMC.

2.3 L'algorithme de Baum-Welch

L'algorithme de Baum-Welch est un algorithme à apprentissage. Son but étant la maximisation de la vraisemblance d'un modèle λ , celui-ci modifie substantiellement les paramètres du modèle étudié afin d'augmenter sa vraisemblance. L'algorithme réalise son optimisation en ré-estimant les différents paramètres (A , B et Π), suivant la (ou les) séquence(s) observée(s).

Les estimations peuvent donc logiquement se concevoir ainsi :

$$\bar{\pi}_i = \text{probabilité d'être dans l'état } S \text{ à l'instant } t = 1 \quad (2.12)$$

$$\bar{a}_{ij} = \frac{\text{nombre de transitions de l'état } S_i \text{ vers } S_j}{\text{nombre de fois où l'on quitte } S_i} \quad (2.13)$$

$$\bar{b}_j(k) = \frac{\text{nombre de fois où l'on est dans l'état } S_j \text{ en observant le symbole } v_k}{\text{nombre de fois où l'on est dans l'état } S_j} \quad (2.14)$$

Il est à noter que lorsque les dénominateurs de \bar{a}_{ij} et $\bar{b}_j(k)$ sont nuls, les probabilités \bar{a}_{ij} et $\bar{b}_j(k)$ ne peuvent pas être calculées. Or, puisque nous réalisons des estimations sur ces coefficients, nous pouvons mettre ces probabilités à 0.

Pour calculer (estimer) ces nouvelles probabilités, l'algorithme de Baum-Welch utilise deux nouvelles matrices : Ξ et Γ . Les coefficients de Ξ , $\xi_t(i, j)$, représentent la probabilité d'être dans l'état S_i à l'instant t et de passer dans l'état S_j à l'instant $t+1$, d'après le modèle λ et la séquence d'observation O . Les coefficients de Γ , $\gamma_t(i)$, représentent la probabilité d'être dans l'état S_i à l'instant t , sachant l'observation O et le modèle λ .

Le calcul de ces coefficients est rapidement réalisé grâce aux deux matrices de paramètres α et β , délivrées par les algorithmes Forward et Backward.

Calcul des coefficients ξ de la matrice Ξ :

$$\xi_t(i, j) = \frac{\alpha_t(i)a_{ij}b_j(o_{t+1})\beta_{t+1}(j)}{P(O/\lambda)} \quad (2.15)$$

Calcul des coefficients γ de la matrice Γ :

$$\gamma_t(i) = \sum_{j=1}^N \xi_t(i, j) = \frac{\alpha_t(i)\beta_t(i)}{P(O/\lambda)} \quad (2.16)$$

Ainsi, la ré-estimation des paramètres (A, B et Π) du modèle Markovien λ est telle que :

$$\pi_i = \gamma_1(i) \quad 1 \leq i \leq N \quad (2.17)$$

$$\bar{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i, j)}{\sum_{t=1}^{T-1} \gamma_t(i)} \quad 1 \leq i \text{ et } j \leq N \quad (2.18)$$

$$\bar{b}_i(k) = \frac{\sum_{t=1}^T \mathbb{1}_{o_t=v_k} \gamma_t(i)}{\sum_{t=1}^T \gamma_t(i)} \quad 1 \leq i \leq N \quad (2.19)$$

Après avoir ré-estimer les différents paramètres du modèle d'origine, l'algorithme recalcule la vraisemblance du nouveau modèle. Il va ensuite ré-itérer les différentes opérations de ré-estimation avec ce nouveau modèle, tant que la vraisemblance courante n'est pas maximale ($P(O/\lambda)=1$).

L'algorithme de Baum-Welch se construit ainsi :

Faire

Appliquer les algorithmes **Forward** et **Backward**

Pour t = 1 à T **Faire**

Pour i = 1 à N **Faire**

Pour j = 1 à N **Faire**

 Calculer $\xi_t(i, j)$

FinPour

 Calculer $\gamma_t(i)$

FinPour

FinPour

 Ré-estimer $\lambda = (A, B, \Pi)$

TantQue (il y a augmentation de $P(O/\lambda)$) **ou** (Il y a encore des itérations à **Faire**)

2.4 Améliorations possibles

L'algorithme de Baum-Welch est un algorithme efficace et puissant. En quelques itérations, les différents paramètres du modèle markovien utilisé sont ajustés pour "coller" à l'observation désignée O. Or, l'algorithme de Baum-welch considère (en modifiant fortement les paramètres

du modèle) que si nous avons observé une séquence bien précise, c'est que celle-ci avait une forte probabilité d'apparition (d'où la maximisation de la vraisemblance). La séquence observée est cependant dépendante du hasard et l'idée selon laquelle une seule séquence d'observation permet de caractériser un modèle de markov entier reste illusoire. Aussi, il est beaucoup plus intéressant de considérer un grand nombre de séquence d'observations, et d'adapter ainsi l'algorithme de Baum-Welch avec ses considérations :

$$P(O/\lambda) = \sum_{l=1}^L P(O^l|\lambda) \quad (2.20)$$

$$\bar{a}_{ij} = \frac{\sum_{l=1}^L \left(\sum_{t=1}^{T-1} \xi_t^l(i, j) \right)}{\sum_{l=1}^L \left(\sum_{t=1}^{T-1} \gamma_t^l(i) \right)} \text{ et } \bar{b}_j(k) = \frac{\sum_{l=1}^L \left(\sum_{t=1 \cap o_t^l = v_k}^T \gamma_t^l(i) \right)}{\sum_{l=1}^L \left(\sum_{t=1}^T \gamma_t^l(i) \right)} \quad (2.21)$$

$$\text{où } \xi_t^l(i, j) = \frac{\sum_{t=1}^{T-1} \alpha_t^l(i) a_{ij} b_j(o_{t+1}^l) \beta_{t+1}^l(j)}{P(O^l|\lambda)} \text{ et } \gamma_t^l(i) = \sum_{j=1}^N \xi_t^l(i, j) \quad (2.22)$$

Cette dernière méthode permet d'obtenir un meilleur modèle de Markov.

Il est aussi à noter que, lorsqu'un grand nombre d'itérations est sélectionné, une perte de définition de différentes valeurs apparaît. En effet, le calcul des α_{ij} étant des produits de probabilités, les valeurs ainsi manipulées tendent vers 0 de manière exponentielle. La précision de la machine devient dès lors trop faible. Pour résoudre le problème et augmenter ainsi la précision des différentes composantes du modèle, une méthode à été inventée : le rescaling. Le principe du rescaling est le suivant : normaliser les coefficients des matrices α et β , afin de conserver une certaine précision. Une des manières les plus courantes réside dans la normalisation des α_{ij} et β_{ij} de façon à avoir la somme des éléments d'une ligne égale à 1.

Chapitre 3

Présentation des programmes

Deux logiciels ont été implémentés lors de la réalisation de ce projet, apportant chacun différentes fonctionnalités. Les deux programmes ont été écrits en langage C. L'interface du deuxième logiciel est conçue en C++.

Le logiciel *MMCS*hell, dont une représentation est visible en annexe 1 page 22, est une simple fenêtre de saisie, où l'on entre les différents paramètres (matrices A, B, Π , α , β et Γ) du modèle Markovien λ . L'affichage des résultats de chaque itération de l'algorithme de Baum-Welch se fait dans la fenêtre de commandes, mais également dans le fichier *resultat.txt* présent dans le répertoire courant.

L'interface *MMCFox* présentée en annexe 2 page 23 intègre une visualisation de matrices sous forme d'image. Cette dernière version a été conçue avec la librairie graphique FOX-Toolkit, qui intègre une interface simple et épurée, et qui gère rapidement la création d'images (impensable sous une fenêtre de commandes).

Dans cette version, nous retrouvons à gauche la zone de saisie des paramètres du modèle Markovien, au milieu, les différents résultats pour chaque itération de l'algorithme de Baum-Welch (et dans le fichier de sortie *resultat.txt*). Dans la partie droite du programme, une visualisation de matrice a été introduite. Les matrices A, B, Π , Alpha, Beta et Gamma peuvent être ainsi observée dans une certaine itération (choisie par l'utilisateur). Plus une zone de l'image ainsi générée est noire, et plus la valeur (probabilité) de la case correspondante de la matrice sélectionnée est proche de 0. Inversement, plus la case est blanche, et plus la probabilité tend vers 1.

Cette version du logiciel limite les dimensions des paramètres du modèle Markovien (le nombre d'états cachés N se limite à 4, tout comme le nombre de symboles observables M, et le nombre maximum d'observation T est de 5).

Ces majorations viennent de la restriction imposée par l'interface. Nous ne pouvons pas générer des zones de saisie (inputBox) infiniment, sans altérer l'affichage. Ce pourquoi, nous avons gardé l'ancienne version du programme qui permet de manipuler des dimensions de matrices plus importantes.

Chapitre 4

Tests et résultats

Pour tester l'efficacité de l'algorithme Baum-Welch, nous avons procédé à de multiples tests. Deux exemples simples vont être présentés dans cette dernière partie.

Exemple 1 :

Soit un modèle Markovien à trois états cachés 1,2 et 3, et deux symboles observables P (pile) et F (face). Ce modèle représente un lancé d'une pièce de monnaie (pile ou face), possédant 3 états cachés.

Matrice de transition des états cachés , $A = [a_{ij}]$:

$$\begin{bmatrix} 0.3 & 0.5 & 0.2 \\ 0 & 0.3 & 0.7 \\ 0 & 0 & 1 \end{bmatrix}$$

Matrice de probabilités de génération des symboles $B = [b_j(k)]$:

$$\begin{bmatrix} 1 & 0 \\ 0.5 & 0.5 \\ 0 & 1 \end{bmatrix}$$

Matrice O :

$$\begin{bmatrix} 0 & 0 & 1 & 1 \end{bmatrix}$$

Matrice $\Pi = [\Pi_i]$:

$$\begin{bmatrix} 0.6 & 0.4 & 0 \end{bmatrix}$$

$$P(O/\lambda) = \mathbf{0.223200}$$

Calcul de l'évolution des paramètres du modèle sur 1 itération

Calcul des matrices Alpha et Beta :

A l'aide des formules suivantes, nous sommes en mesure de calculer les $\alpha_t(i)$. $\alpha_1(i) = \pi_i b_i(o_1)$

$$\alpha_{t+1}(j) = \left(\sum_{i=1}^N \alpha_t(i) a_{ij} \right) b_j(o_{t+1})$$

AN.

$$\alpha_1(1) = \pi_1 b_1(o_1) = 0.6 \times 1.0 = 0.6$$

$$\alpha_1(2) = \pi_2 b_2(o_1) = 0.4 \times 0.5 = 0.2$$

$$\alpha_1(3) = \pi_3 b_3(o_1) = 0.0 \times 0.0 = 0.0$$

$$\alpha_2(1) = \alpha_1(1) a_{11} b_1(o_2) + \alpha_1(2) a_{21} b_1(o_2) + \alpha_1(3) a_{31} b_1(o_2) = 0.6 \times 0.3 \times 1.0 + 0.2 \times 0.0 \times 1.0 + 0.0 \times 0.0 \times 1.0 = 0.18$$

...

Nous obtenons alors les valeurs suivantes :

$$\alpha_1(1) = 0.6$$

$$\alpha_1(2) = 0.2$$

$$\alpha_1(3) = 0.0$$

$$\alpha_2(1) = 0.18$$

$$\alpha_2(2) = 0.18$$

$$\alpha_2(3) = 0.0$$

$$\alpha_3(1) = 0.0$$

$$\alpha_3(2) = 0.072$$

$$\alpha_3(3) = 0.162$$

$$\alpha_4(1) = 0.0$$

$$\alpha_4(2) = 0.0108$$

$$\alpha_4(3) = 0.2124$$

La procédure de calcul des $\beta_t(i)$ est identique, avec :

$$\beta_T(i) = 1 \text{ et } \beta_t(i) = \sum_{j=1}^N a_{ij} \beta_{t+1}(j) b_j(o_{t+1})$$

AN.

$$\beta_4(1) = 1.0$$

$$\beta_4(2) = 1.0$$

$$\beta_4(3) = 1.0$$

$$\beta_3(1) = a_{11} \beta_4(1) b_1(o_4) + a_{12} \beta_4(2) b_2(o_4) + a_{13} \beta_4(3) b_3(o_4) = 0.3 \times 1.0 \times 0.0 + 0.5 \times 1.0 \times 0.5 + 0.2 \times 1.0 \times 1.0 = 0.45$$

...

Nous obtenons alors les valeurs suivantes :

$$\beta_1(1) = 0.330625$$

$$\beta_1(2) = 0.124125$$

$$\beta_1(3) = 0.0$$

$$\beta_2(1) = 0.4125$$

$$\beta_2(2) = 0.8275$$

$$\beta_2(3) = 1.0$$

$$\beta_3(1) = 0.45$$

$$\beta_3(2) = 0.85$$

$$\beta_3(3) = 1.0$$

$$\beta_4(1) = 1.0$$

$$\beta_4(2) = 1.0$$

$$\beta_4(3) = 1.0$$

Ré-estimation des matrices A, B et Π :

Ré-estimation de A :

$$\bar{a}_{11} = \frac{\sum_{t=1}^{T-1=4-1=3} \xi_t(i=1, j=1)}{\sum_{t=1}^{T-1=4-1=3} \gamma_t(i=1)} = \frac{\sum_{t=1}^{T-1=3} \alpha_t(1) a_{11} b_1(o_{t+1}) \beta_{t+1}(1)}{\sum_{j=1}^{N=3} \alpha_t(1) a_{1j} b_j(o_{t+1}) \beta_{t+1}(j)}$$

$$\begin{aligned} \bar{a}_{11} = & (0.600000 \times 0.300000 \times 1.000000 \times 0.412500 + 0.180000 \times 0.300000 \times 0.000000 \times 0.450000 + \\ & 0.000000 \times 0.300000 \times 0.000000 \times 1.000000) / (0.600000 \times 0.300000 \times 1.000000 \times 0.412500 + \\ & 0.600000 \times 0.500000 \times 0.500000 \times 0.827500 + 0.600000 \times 0.200000 \times 0.000000 \times 1.000000 + \\ & 0.180000 \times 0.300000 \times 0.000000 \times 0.450000 + 0.180000 \times 0.500000 \times 0.500000 \times 0.850000 + \\ & 0.180000 \times 0.200000 \times 1.000000 \times 1.000000 + 0.000000 \times 0.300000 \times 0.000000 \times 1.000000 + 0.000000 \times \\ & 0.500000 \times 0.500000 \times 1.000000 + 0.000000 \times 0.200000 \times 1.000000 \times 1.000000) = 0.272352 \end{aligned}$$

Le procédé est alors répété pour toutes les valeurs \bar{a}_{ij} . Nous obtenons les résultats suivants :

$$\bar{a}_{11} = 0.272352$$

$$\bar{a}_{12} = 0.595598$$

$$\bar{a}_{13} = 0.132050$$

$$\bar{a}_{21} = 0.000000$$

$$\bar{a}_{22} = 0.249282$$

$$\bar{a}_{23} = 0.750718$$

$$\bar{a}_{31} = 0.000000$$

$$\bar{a}_{32} = 0.000000$$

$$\bar{a}_{33} = 1.000000$$

Ré-estimation de Π :

$$\bar{\pi}_1 = \frac{\alpha_1(1)\beta_1(1)}{P(O|\lambda)} = \frac{0.600000 \times 0.330625}{0.223200} = 0.888777$$

$$\bar{\pi}_2 = \frac{\alpha_1(2)\beta_1(2)}{P(O|\lambda)} = \frac{0.200000 \times 0.124125}{0.223200} = 0.111223$$

$$\bar{\pi}_3 = \frac{\alpha_1(3)\beta_1(3)}{P(O|\lambda)} = \frac{0.000000 \times 0.000000}{0.223200} = 0.000000$$

Ré-estimation de B :

$$\bar{b}_1(1) = \frac{\sum_{t=1}^T \gamma_{t(o_t=v_1)} \gamma_t(1)}{\sum_{t=1}^T \gamma_t(1)} = \frac{\sum_{t=1}^T \alpha_t(1) \beta_t(1)}{\sum_{t=1}^T \alpha_t(1) \beta_t(1)}$$

$$\bar{b}_1(1) = (0.600000 \times 0.330625 + 0.180000 \times 0.412500) / (0.600000 \times 0.330625 + 0.180000 \times 0.412500 + 0.000000 \times 0.450000 + 0.000000 \times 1.000000) = 1.000000$$

Le procédé est alors répété pour toutes les valeurs $\bar{b}_i(j)$. Nous obtenons les résultats suivants :

$$\bar{b}_1(1) = 1.000000$$

$$\bar{b}_2(1) = 0.707049$$

$$\begin{aligned}\bar{b}_3(1) &= 0.000000 \\ \bar{b}_1(2) &= 0.000000 \\ \bar{b}_2(2) &= 0.292951 \\ \bar{b}_3(2) &= 1.000000\end{aligned}$$

Résultats de l'algorithme de Baum-Welch sur 3 itérations

Itération numéro 1 :

Matrice A :

$$\begin{bmatrix} 0.272352 & 0.595598 & 0.132050 \\ 0.000000 & 0.249282 & 0.750718 \\ 0.000000 & 0.000000 & 1.000000 \end{bmatrix}$$

Matrice B :

$$\begin{bmatrix} 1.000000 & 0.000000 \\ 0.707049 & 0.292951 \\ 0.000000 & 1.000000 \end{bmatrix}$$

Matrice O :

$$\begin{bmatrix} 0 & 0 & 1 & 1 \end{bmatrix}$$

Matrice Π :

$$\begin{bmatrix} 0.888777 & 0.111223 & 0.000000 \end{bmatrix}$$

Matrice Alpha :

$$\begin{bmatrix} 0.600000 & 0.200000 & 0.000000 \\ 0.180000 & 0.180000 & 0.000000 \\ 0.000000 & 0.072000 & 0.162000 \\ 0.000000 & 0.010800 & 0.212400 \end{bmatrix}$$

Matrice Beta :

$$\begin{bmatrix} 0.330625 & 0.124125 & 0.000000 \\ 0.412500 & 0.827500 & 1.000000 \\ 0.450000 & 0.850000 & 1.000000 \\ 1.000000 & 1.000000 & 1.000000 \end{bmatrix}$$

Matrice Γ :

$$\begin{bmatrix} 0.888777 & 0.111223 & 0.000000 \\ 0.332661 & 0.667339 & 0.000000 \\ 0.000000 & 0.274194 & 0.725806 \end{bmatrix}$$

[0.000000 0.048387 0.951613]

P(O/λ) = **0.381487**

Itération numéro 2 :

Matrice A :

[0.152756 0.774101 0.073144]
[0.000000 0.103466 0.896534]
[0.000000 0.000000 1.000000]

Matrice B :

[1.000000 0.000000]
[0.837402 0.162598]
[0.000000 1.000000]

Matrice O :

[0 0 1 1]

Matrice Π :

[0.970538 0.029462 0.000000]

Matrice Alpha :

[0.888777 0.078640 0.000000]
[0.242060 0.388140 0.000000]
[0.000000 0.070580 0.323348]
[0.000000 0.005154 0.376333]

Matrice Beta :

[0.416582 0.142920 0.000000]
[0.275778 0.810874 1.000000]
[0.306531 0.823745 1.000000]
[1.000000 1.000000 1.000000]

Matrice Γ :

[0.970538 0.029462 0.000000]
[0.174986 0.825014 0.000000]
[0.000000 0.152403 0.847597]
[0.000000 0.013511 0.986489]

P(O/λ) = **0.603545**

Itération numéro 3 :

Matrice A :

$$\begin{bmatrix} 0.044302 & 0.938471 & 0.017227 \\ 0.000000 & 0.020091 & 0.979909 \\ 0.000000 & 0.000000 & 1.000000 \end{bmatrix}$$

Matrice B :

$$\begin{bmatrix} 1.000000 & 0.000000 \\ 0.954970 & 0.045030 \\ 0.000000 & 1.000000 \end{bmatrix}$$

Matrice O :

$$\begin{bmatrix} 0 & 0 & 1 & 1 \end{bmatrix}$$

Matrice Π :

$$\begin{bmatrix} 0.996770 & 0.003230 & 0.000000 \end{bmatrix}$$

Matrice Alpha :

$$\begin{bmatrix} 0.970538 & 0.024671 & 0.000000 \\ 0.148255 & 0.631273 & 0.000000 \\ 0.000000 & 0.029281 & 0.576801 \\ 0.000000 & 0.000493 & 0.603052 \end{bmatrix}$$

Matrice Beta :

$$\begin{bmatrix} 0.619858 & 0.079009 & 0.000000 \\ 0.188106 & 0.911900 & 1.000000 \\ 0.199011 & 0.913358 & 1.000000 \\ 1.000000 & 1.000000 & 1.000000 \end{bmatrix}$$

Matrice Γ :

$$\begin{bmatrix} 0.996770 & 0.003230 & 0.000000 \\ 0.046206 & 0.953794 & 0.000000 \\ 0.000000 & 0.044311 & 0.955689 \\ 0.000000 & 0.000816 & 0.999184 \end{bmatrix}$$

$$P(O/\lambda) = \mathbf{0.878811}$$

Conclusion

Les tests réalisés montrent que l'algorithme de Baum-Welch est efficace pour la ré-estimation des paramètres.

Références

- Rapport interne n°262 - décembre 2002, **Les modèles de Markov cachés à substitution de symboles**, S. AUPETIT, M. SLIMANE, N. MONMARCHE

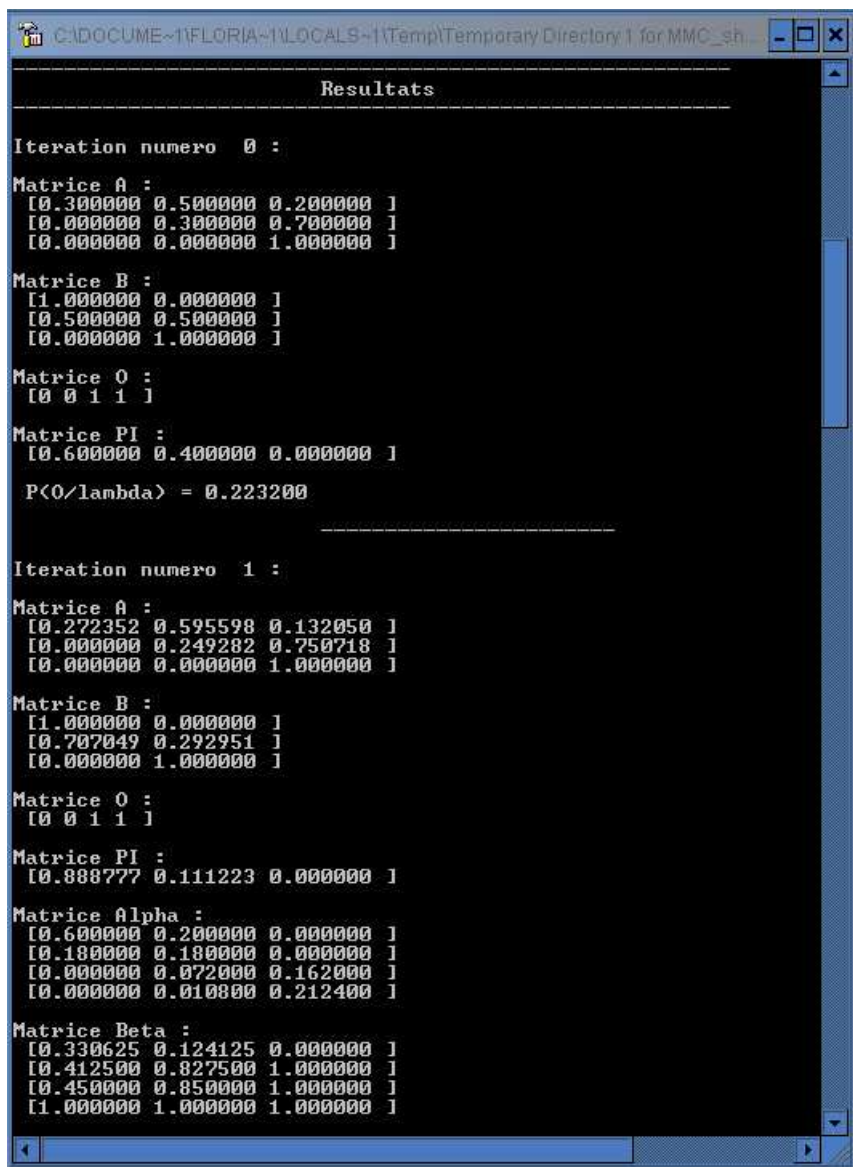
- Rapport interne n°260 - octobre 2002, **Les Chaînes de Markov cachées : définitions, algorithmes, architectures**, Mohamed SLIMANE

- Cours Ecole Polytechnique universitaire de Tours Département Informatique - janvier 2005, **Les processus stochastiques**, M. SLIMANE.

- **An inequality with applications to statical estimation for probabilistic functions of Markov processes and to a model for ecology**- Bull American Mathematical Society - 1967, L.E BAUM.

Chapitre 5

Annexe1 : Interface DOS



```
-----
                                Resultats
-----

Iteration numero 0 :

Matrice A :
[0.300000 0.500000 0.200000 ]
[0.000000 0.300000 0.700000 ]
[0.000000 0.000000 1.000000 ]

Matrice B :
[1.000000 0.000000 ]
[0.500000 0.500000 ]
[0.000000 1.000000 ]

Matrice O :
[0 0 1 ]

Matrice PI :
[0.600000 0.400000 0.000000 ]

P(O/lambda) = 0.223200

-----

Iteration numero 1 :

Matrice A :
[0.272352 0.595598 0.132050 ]
[0.000000 0.249282 0.750718 ]
[0.000000 0.000000 1.000000 ]

Matrice B :
[1.000000 0.000000 ]
[0.707049 0.292951 ]
[0.000000 1.000000 ]

Matrice O :
[0 0 1 ]

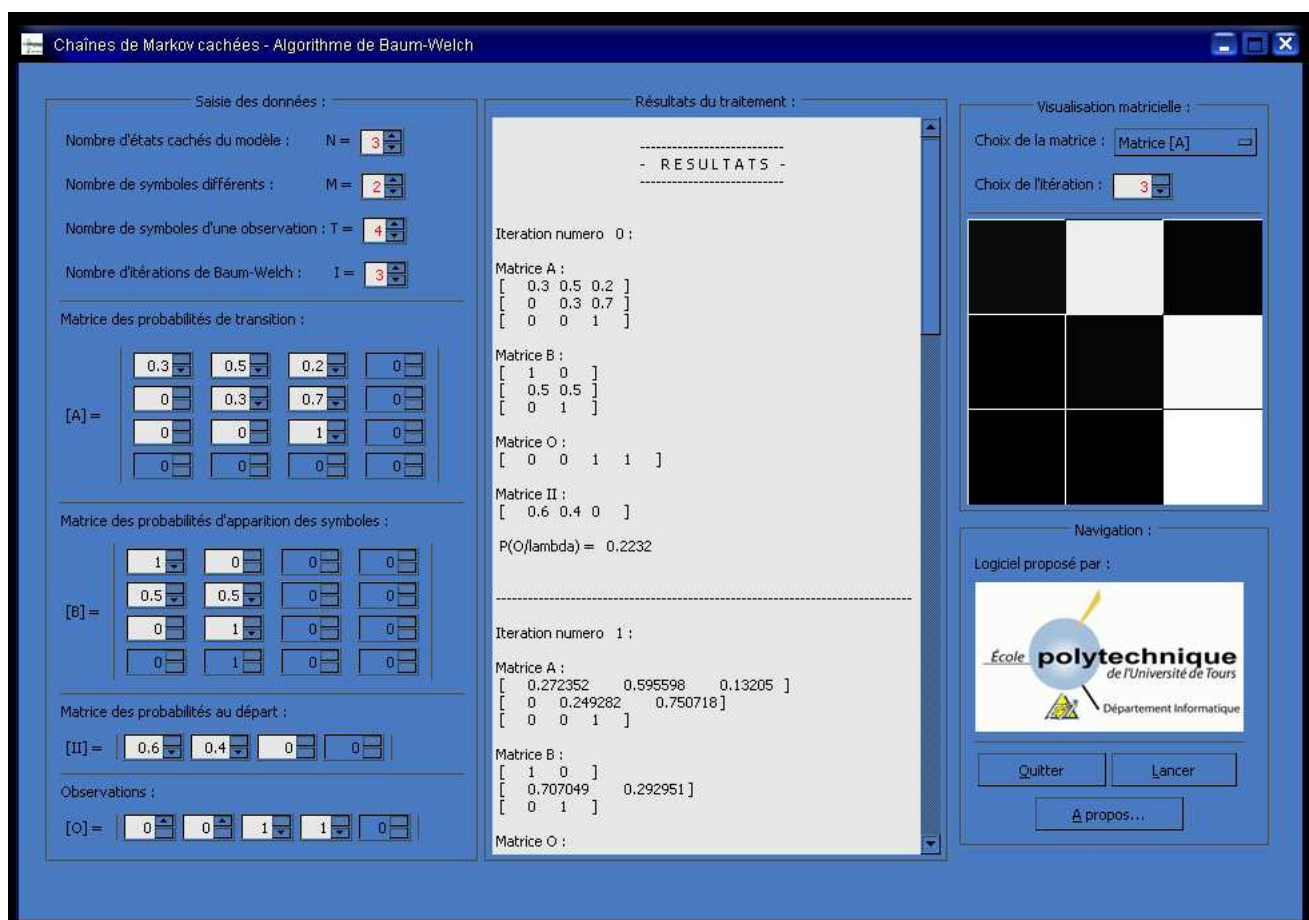
Matrice PI :
[0.888777 0.111223 0.000000 ]

Matrice Alpha :
[0.600000 0.200000 0.000000 ]
[0.180000 0.180000 0.000000 ]
[0.000000 0.072000 0.162000 ]
[0.000000 0.010800 0.212400 ]

Matrice Beta :
[0.330625 0.124125 0.000000 ]
[0.412500 0.827500 1.000000 ]
[0.450000 0.850000 1.000000 ]
[1.000000 1.000000 1.000000 ]
```


Chapitre 6

Annexe2 : Interface FOX



Résumé :

Les chaînes de Markov cachées (CMC) sont des outils statistiques d'apprentissage. Elles permettent de modéliser divers systèmes physiques probabilistes, à partir d'un ensemble d'observations, et dont le nombre d'états du système peut ne pas être connu. L'estimation des paramètres est parfois difficile à obtenir. Nous présentons ici l'algorithme de Baum-Welch qui améliore le maximum de vraisemblance en ré-estimant les différents paramètres. Cette méthode efficace, nécessite néanmoins d'être optimisée pour augmenter la précision des différentes composantes du modèle.

Mots clés :

Processus stochastiques, Chaînes de Markov, Chaînes de Markov Cachées (CMC), Maximum de vraisemblance.

Abstract :

Hidden Markov models (HMM) is basically a Markov chain whose internal state cannot be observed directly but only through some probabilistic function. That is, the internal state of the model only determines the probability distribution λ of the observed variable. The Baum-Welch algorithm is a method of adjusting the λ parameters to maximize the likelihood of the training set. That's efficient, but not optimum without some improvements such as the rescaling method, which provides a larger accuracy.

Key words :

Stochastic processes, Markov Models, Hidden Markov Models (HMM), Maximum Likelihood.