

MAPSI — cours 7 : Chaîne de Markov Cachée

Vincent Guigue, Thierry Artières
vincent.guigue@lip6.fr

LIP6 – Université Paris 6, France

Impossible d'évaluer les modèles sur les données qui ont servi à l'apprentissage !!!

⇒ Biais dans l'évaluation

- Pourtant : l'évaluation est aussi importante que le modèle lui-même
- ⇒ il faut faire les deux, c'est à dire diviser les données
 - beaucoup de données en apprentissage... Evaluation pauvre !
 - beaucoup de données en test... Modèle pauvre !

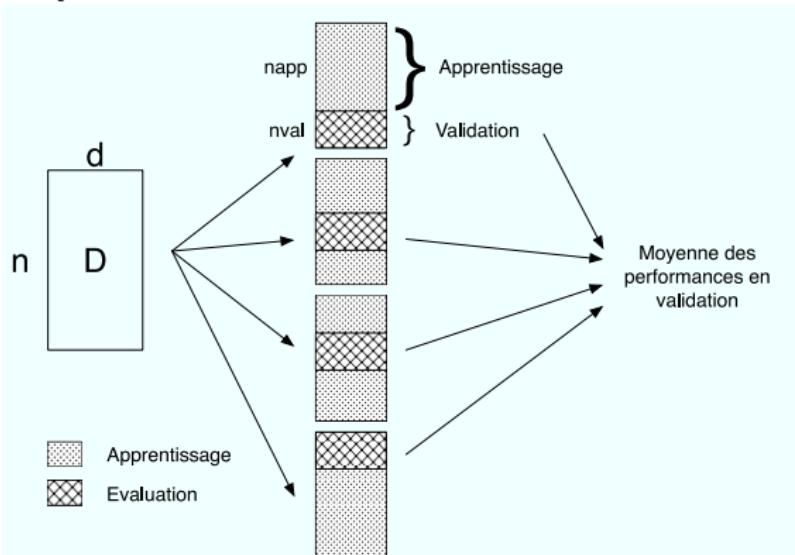
Comment évaluer les performances efficacement ?

- 3 Procédures :
 - **Beaucoup de données** (ou peu de temps) :
Apprentissage/test

Comment évaluer les performances efficacement ?

3 Procédures :

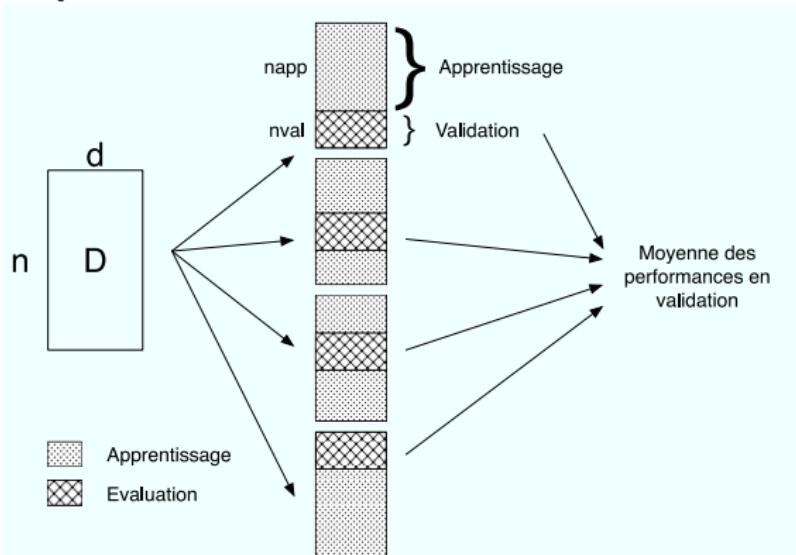
- **Beaucoup de données** (ou peu de temps) : Apprentissage/test
- **Le plus souvent** Validation croisée



Comment évaluer les performances efficacement ?

● 3 Procédures :

- **Beaucoup de données** (ou peu de temps) : Apprentissage/test
- **Le plus souvent** Validation croisée



- **Peu de données** Leave-one-out

Définition

Le sur-apprentissage consiste à extraire des règles qui sont efficaces en apprentissage mais pas en test.

Exemples :

- Transitions inconnues en apprentissage... $\Rightarrow \text{proba} = 0$
 - On est sur que ça ne nuit pas aux performances en apprentissage...
 - Mais ce n'est pas forcément pertinent en test
 - Proposition : toutes les matrices de comptage de transitions seront initialisées à 1 et pas 0. (Assez efficace en pratique)
- Transitions erronées (transitions étranges, erreurs de mesure...)
 - Bonne règle en apprentissage (cette transition n'existe nulle part ailleurs)
 - Mauvaise règle en test
 - Proposition : dans le codage des N-grams, on ne prend pas en compte les mots qui apparaissent très peu et on élimine les transitions associées.

Retour sur le traitement des séquences

Tâches :

- Classification / Clustering
- Etiquetage / Segmentation
- Génération de séquences
- Reconnaissance de chaîne de caractères

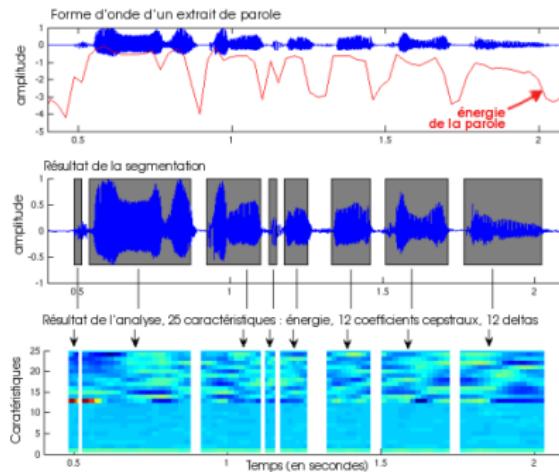


- Reconnaissance de paroles
- Génération/reconnaissance de mouvements
- Reconnaissance de mouvements (2)

Retour sur le traitement des séquences

Tâches :

- Classification / Clustering
- Etiquetage / Segmentation
- Génération de séquences
- Reconnaissance de chaîne de caractères
- Reconnaissance de paroles



Retour sur le traitement des séquences

Tâches :

- Classification / Clustering
- Etiquetage / Segmentation
- Génération de séquences
- Reconnaissance de chaîne de caractères
- Reconnaissance de paroles
- Génération/reconnaissance de mouvements

Retour sur le traitement des séquences

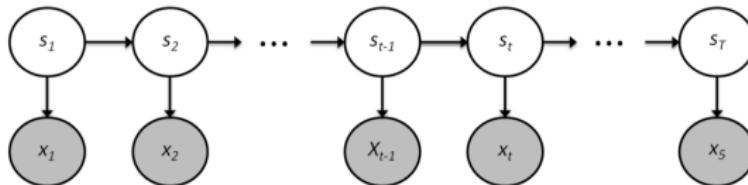
Tâches :

- Classification / Clustering
- Etiquetage / Segmentation
- Génération de séquences
- Reconnaissance de chaîne de caractères
- Reconnaissance de paroles
- Génération/reconnaissance de mouvements
- Reconnaissance de mouvements (2)



Modèles de Markov Cachés (MMC = HMM)

- Séparation des **observations** et des **états**
 - Une chaîne = une séquence d'observations...
 - ... chaque observation ayant été générée par un état
- 2 composantes
 - Chaîne de Markov d'états finis
 - Ensemble de lois de probabilité d'émission
- 2 points de vues
 - Génératif : la chaîne génère des états qui entraînent des observations
 - Analytique : les observations fournissent de l'évidence sur la suite d'états (décodage) et sur le modèle (apprentissage)

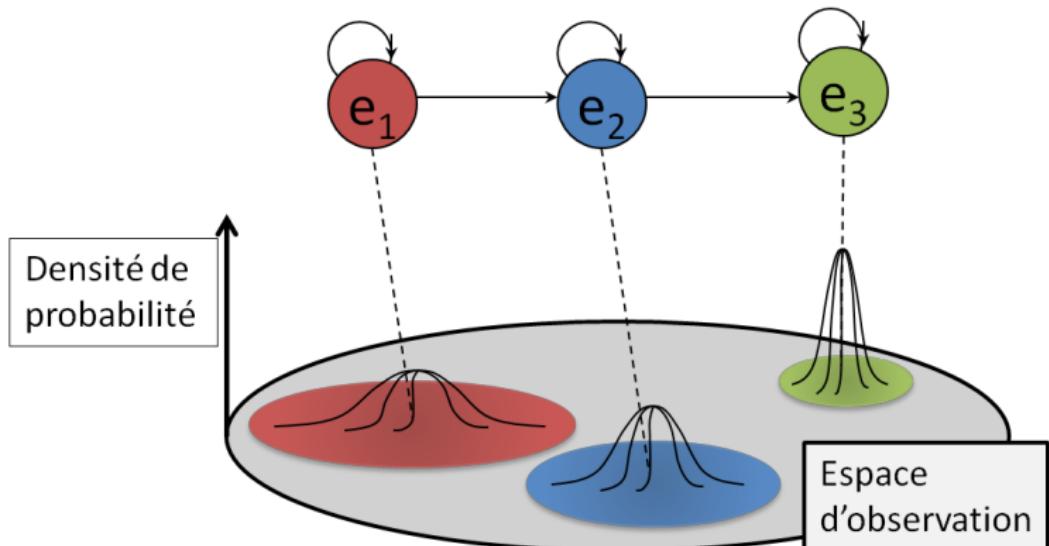


- La chaîne de Markov est toujours composée de :
 - d'une séquence d'**états** $S = (s_1, \dots, s_T)$
 - dont les valeurs sont tirées dans un ensemble fini $Q = (q_1, \dots, q_N)$
 - Le modèle est toujours défini par $\{\Pi, A\}$
 - $\pi_i = P(s_1 = q_i)$
 - $a_{ij} = p(s_{t+1} = q_j | s_t = q_i)$
- Les **observations** sont modélisées à partir des s_t
 - séquence d'observation : $X = (x_1, \dots, x_T)$
 - loi de probabilité : $b_j(t) = p(x_t | s_t = q_j)$
 - B peut être discrète ou continue
- MMC : $\lambda = \{\Pi, A, B\}$

Ce que l'on manipule

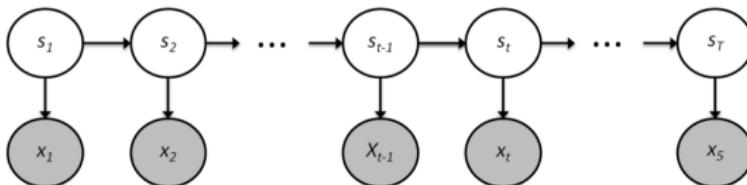
- Séquence d'observations
 - $X = (x_1, \dots, x_T)$
- Séquence d'états (**cachée = manquante**)
 - $S = (s_1, \dots, s_T)$

MMC à lois d'observation continues



- Note : la figure présente un modèle *Gauche/Droite...*
Quelle est sa particularité ?

Hypothèses MMC



- CM d'ordre 1 :

$$p(s_t | s_1, \dots, s_{t-1}) = p(s_t | s_{t-1})$$

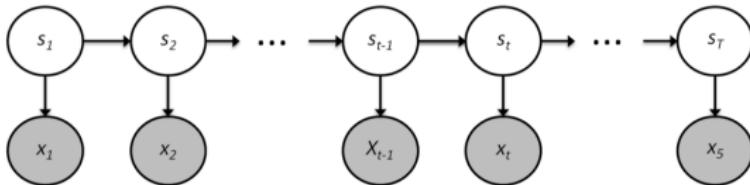
- Indépendance des observations :

$$p(x_t | x_{t-1}, s_t) = p(x_t | s_t)$$

i.e. : les observations successives sont indépendantes conditionnellement aux états !

Notations simplifiées (de a à b) :

$$s_1^t = s_1, \dots, s_t, \quad x_1^t = x_1, \dots, x_t$$



Comment calculer probabilités suivantes ?

- Séquence d'états
- Séquence d'observations (connaissant les états)
- Séquences obs + états :

Algorithm 1: Génération d'une séquence $\{x_1, \dots, x_T\}$

Data: A, B, Π

Result: x_1^T, s_1^T

$S \leftarrow [];$

$X \leftarrow [];$

Tirer s_1 en fonction de Π ;

Tirer x_1 en fonction de B et s_1 ;

$s_t \leftarrow s_1, x_t \leftarrow x_1, t \leftarrow 1;$

$S \leftarrow [S, s_t], X \leftarrow [X, x_t];$

while s_t n'est pas l'état final **do**

$s_{t+1} \leftarrow$ tirage selon $(A(s_t, :))$;

$x_{t+1} \leftarrow$ tirage selon $(B(s_{t+1}, :))$;

$s_t \leftarrow s_1, x_t \leftarrow x_1;$

$S \leftarrow [S, s_t], X \leftarrow [X, x_t];$

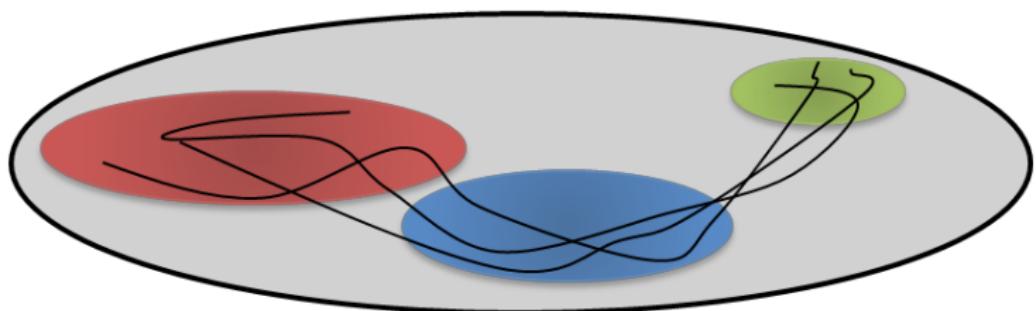
$t \leftarrow t + 1;$

Représentation d'une trajectoire

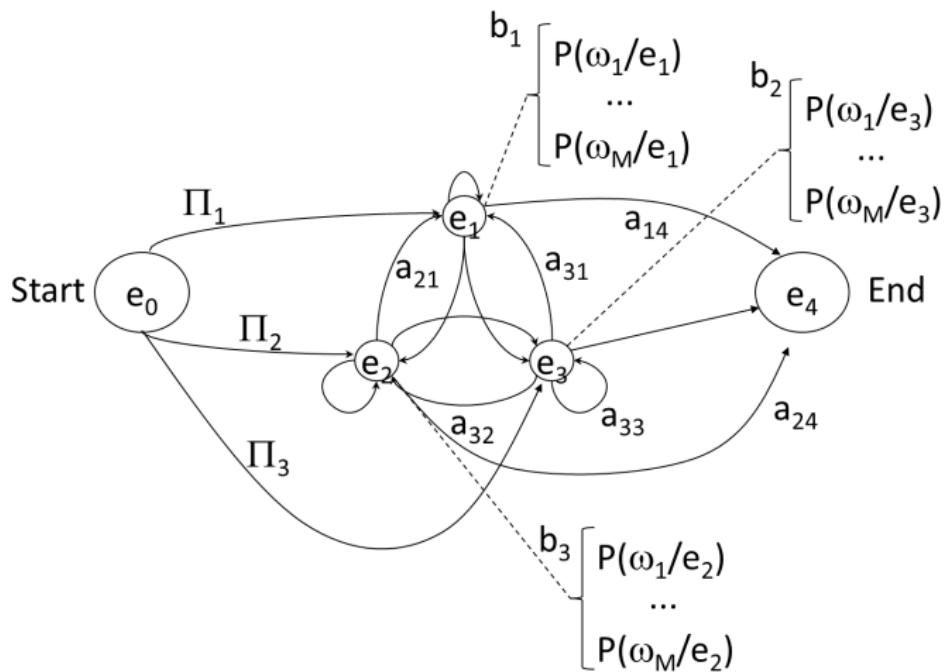
- trajectoire d'états



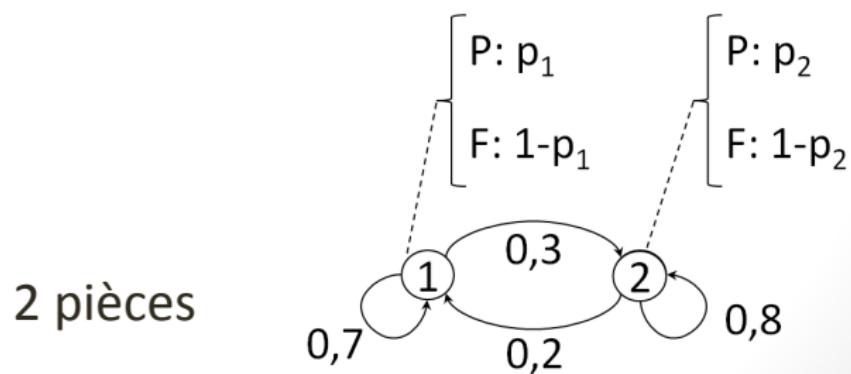
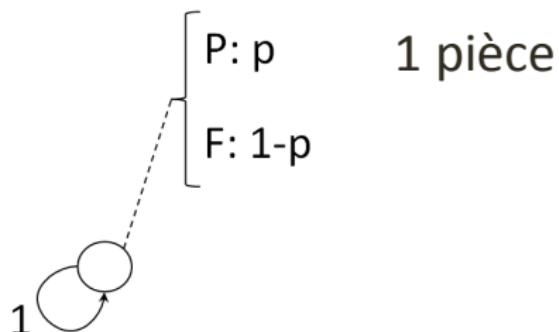
- trajectoire d'observations



Représentation d'un MMC



Modélisation des pièces...



Capacités d'expression CM, MMC

- N urnes (de Rabiner)
 - chaque urne = distribution spécifique des couleurs de boules
- On tire successivement des boules dans les différentes urnes et on note les couleurs des boules



| | | |
|-----------------------------|-----------------------------|-----------------------------|
| $P(\text{red}) = b_1(1)$ | $P(\text{red}) = b_2(1)$ | $P(\text{red}) = b_N(1)$ |
| $P(\text{green}) = b_1(2)$ | $P(\text{green}) = b_2(2)$ | $P(\text{green}) = b_N(2)$ |
| $P(\text{yellow}) = b_1(3)$ | $P(\text{yellow}) = b_2(3)$ | $P(\text{yellow}) = b_N(3)$ |
| $P(\text{black}) = b_1(4)$ | $P(\text{black}) = b_2(4)$ | $P(\text{black}) = b_N(4)$ |
| ... | ... | ... |
| $P(\text{orange}) = b_1(M)$ | $P(\text{orange}) = b_2(M)$ | $P(\text{orange}) = b_N(M)$ |

- Modélisation complexe
 - mixture de sources émitrices
- Problèmes de sécançage des signaux
 - Parole, écrit,...
 - ADN,
 - Image

Les trois problèmes des MMC (Fergusson - Rabiner)

- ① **Evaluation** : λ donné, calcul de $p(x_1^T | \lambda)$
- ② **Décodage** : λ donné, quelle séquence d'états a générée les observations ?
- ③ **Apprentissage** : à partir d'une série d'observations, trouver λ^*

Et les applications associées...

- ① **Evaluation** : λ donné, calcul de $p(x_1^T | \lambda)$
- ② **Décodage** : λ donné, quelle séquence d'états a générée les observations ?

$$s_1^{T*} = \arg \max_{s_1^T} p(s_1^T | x_1^T, \lambda) = \arg \max_{s_1^T} p(x_1^T, s_1^T | \lambda)$$

- ③ **Apprentissage** : à partir d'une série d'observations, trouver λ^*

Et les applications associées...

Les trois problèmes des MMC (Fergusson - Rabiner)

- ① **Evaluation** : λ donné, calcul de $p(x_1^T | \lambda)$
- ② **Décodage** : λ donné, quelle séquence d'états a générée les observations ?

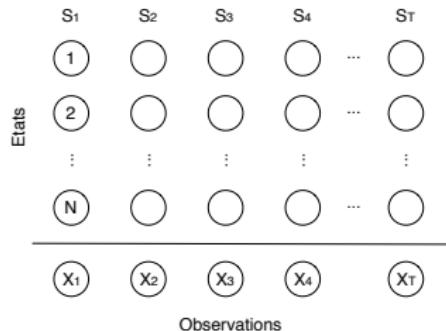
$$s_1^{T*} = \arg \max_{s_1^T} p(s_1^T | x_1^T, \lambda) = \arg \max_{s_1^T} p(x_1^T, s_1^T | \lambda)$$

- ③ **Apprentissage** : à partir d'une série d'observations, trouver λ^*

$$\lambda^* = \{\Pi^*, A^*, B^*\} = \arg \max_{\lambda} p(x_1^T | \lambda)$$

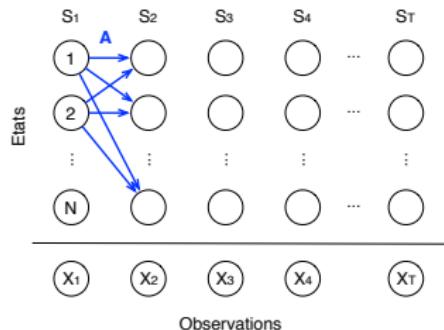
Et les applications associées...

PB1 : évaluation $p(x_1^T | \lambda)$



- Quel point de départ ?
- Complexité ?

PB1 : évaluation $p(x_1^T | \lambda)$

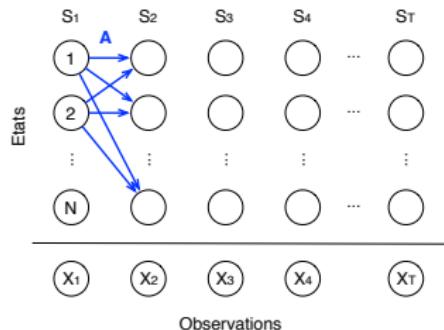


- Quel point de départ ? **Proba totales**

$$p(x_1^T | \lambda) = \sum_{s_1^T} p(x_1^T, s_1^T | \lambda)$$

- Complexité ?

PB1 : évaluation $p(x_1^T | \lambda)$

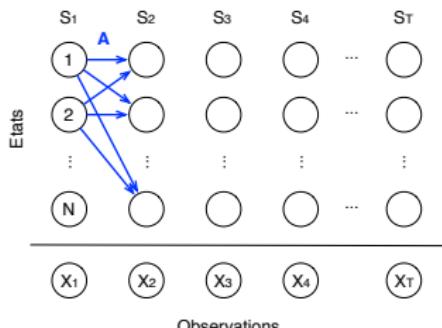


- Quel point de départ ? **Proba totales**

$$p(x_1^T | \lambda) = \sum_{s_1^T} p(x_1^T, s_1^T | \lambda)$$

- Simplifier la combinatoire...
- Complexité ?

PB1 : évaluation $p(x_1^T | \lambda)$



- Quel point de départ ? **Proba totales**

$$p(x_1^T | \lambda) = \sum_{s_1^T} p(x_1^T, s_1^T | \lambda)$$

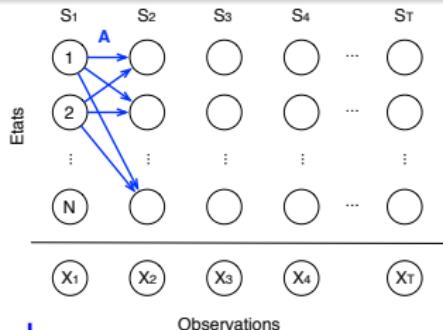
- Simplifier la combinatoire... **Hypothèse MMC ordre 1**

$$p(x_1^T | \lambda) = \sum_{s_1^T} \prod_{t=1}^T p(s_t | s_{t-1}) p(x_t | s_t)$$

- Complexité ?

PB1 : évaluation $p(x_1^T | \lambda)$

A chaque pas de temps,
envisager toutes les
combinaisons d'états qui ont pu
emmené ici...



- Quel point de départ ? **Proba totales**

$$p(x_1^T | \lambda) = \sum_{s_1^T} p(x_1^T, s_1^T | \lambda)$$

- Simplifier la combinatoire... **Hypothèse MMC ordre 1**

$$p(x_1^T | \lambda) = \sum_{s_1^T} \prod_{t=1}^T p(s_t | s_{t-1}) p(x_t | s_t)$$

- Complexité ? **$\mathcal{O}(N^T)$**

Calcul par récurrence :

- On pose :

$$\alpha_t(i) = p(x_1^t, s_t = i | \lambda)$$

- Que vaut alors $p(x_1^T | \lambda)$?

Calcul par récurrence :

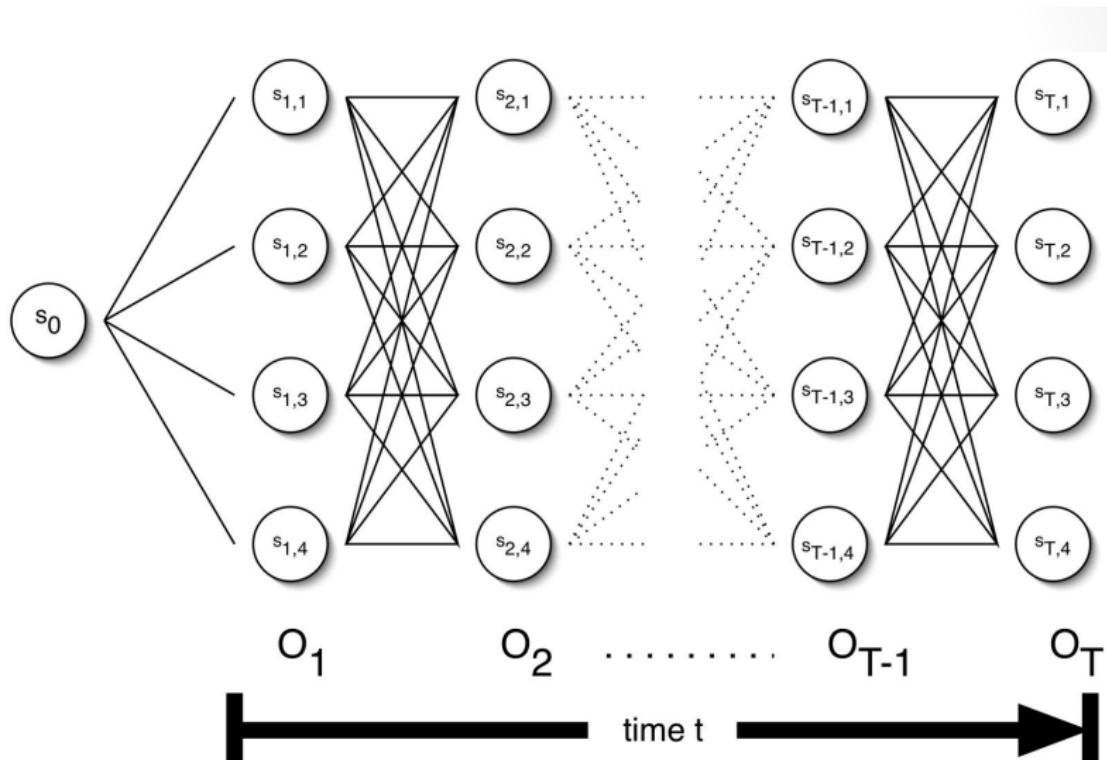
- On pose :

$$\alpha_t(i) = p(x_1^t, s_t = i | \lambda)$$

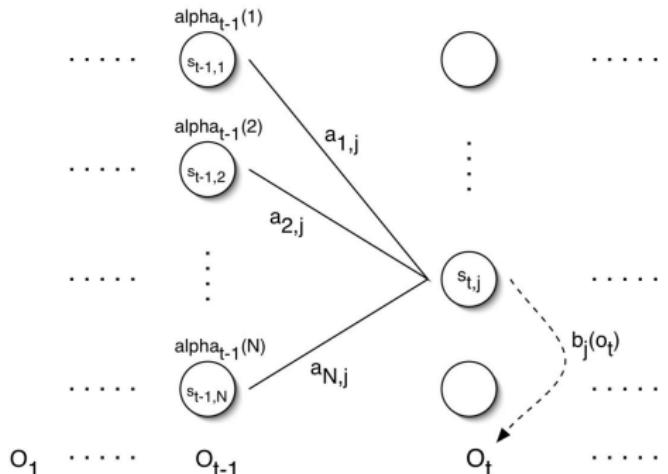
- Que vaut alors $p(x_1^T | \lambda)$?

$$p(x_1^T | \lambda) = \sum_{i=1}^N \alpha_T(i)$$

Envisager tous les états à tous les pas de temps (+ transitions) = trellis d'hypothèses



PB1 : Algorithme forward

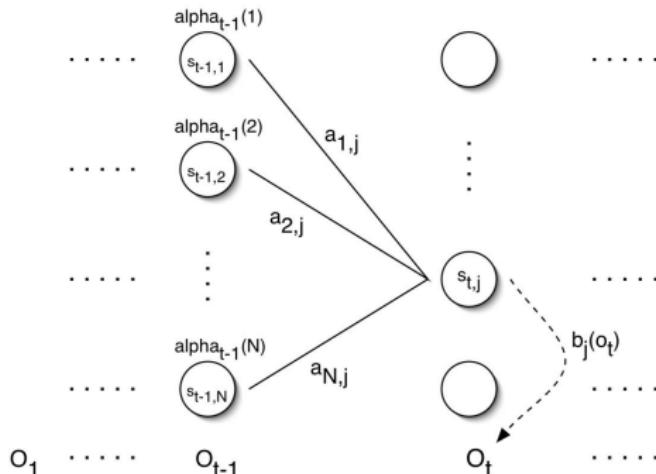


$$\alpha_t(j) = p(x_1^t, s_t = j | \lambda)$$

Exprimer en fonction des α_{t-1}

Formalisation récursive = briser la complexité

PB1 : Algorithme forward



$$\alpha_t(j) = p(x_1^t, s_t = j | \lambda)$$

$$\alpha_t(j) = \left[\sum_{i=1}^N \alpha_{t-1}(i) a_{ij} \right] b_j(x_t)$$

Formalisation récursive = briser la complexité

PB1 : Algorithme *forward*

- Initialisation :

$$\alpha_{t=1}(i) = p(x_1^1, s_1 = i | \lambda) = \dots$$

PB1 : Algorithme *forward*

- Initialisation :

$$\alpha_{t=1}(i) = p(x_1^1, s_1 = i | \lambda) = \dots$$

$$= \pi_i b_i(x_1)$$

- Itération :

$$\alpha_t(j) = \left[\sum_{i=1}^N \alpha_{t-1}(i) a_{ij} \right] b_j(x_t)$$

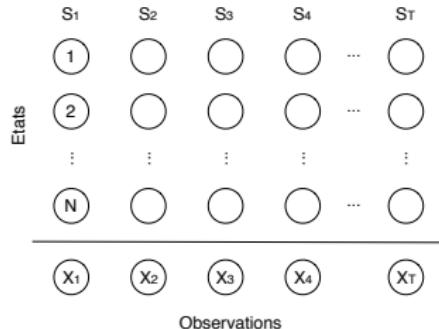
- Terminaison :

$$p(x_1^T | \lambda) = \sum_{i=1}^N \alpha_T(i)$$

- Complexité linéaire en T

- Usuellement : $T \gg N$

PB2 : décodage



$$s_1^{T*} = \arg \max_{s_1^T} p(x_1^T | \lambda)$$

Avec la formule précédente (hyp. MMC ordre 1) :

$$s_1^{T*} = \arg \max_{s_1^T} \prod_{t=1}^T p(s_t | s_{t-1}) p(x_t | s_t)$$

- Même schéma que précédemment = **algorithme de Viterbi**

- On pose :

Meilleur score pour un chemin au temps t , se terminant à l'état i :

$$\delta_t(i) = \max_{s_1^{t-1}} p(s_1^{t-1}, s_t = i, x_1^t | \lambda)$$

A t , la probabilité du meilleur chemin est la combinaison :

- d'un des meilleurs chemins précédents...
- ... et de la transition vers s_j + observation de x_t à partir de s_j

- On pose :

Meilleur score pour un chemin au temps t , se terminant à l'état i :

$$\delta_t(i) = \max_{s_1^{t-1}} p(s_1^{t-1}, s_t = i, x_1^t | \lambda)$$

A t , la probabilité du meilleur chemin est la combinaison :

- d'un des meilleurs chemins précédents...
- ... et de la transition vers s_j + observation de x_t à partir de s_j

- Initialisation :
- Récurrence :

- On pose :

Meilleur score pour un chemin au temps t , se terminant à l'état i :

$$\delta_t(i) = \max_{s_1^{t-1}} p(s_1^{t-1}, s_t = i, x_1^t | \lambda)$$

A t , la probabilité du meilleur chemin est la combinaison :

- d'un des meilleurs chemins précédents...
- ... et de la transition vers s_j + observation de x_t à partir de s_j

- Initialisation :

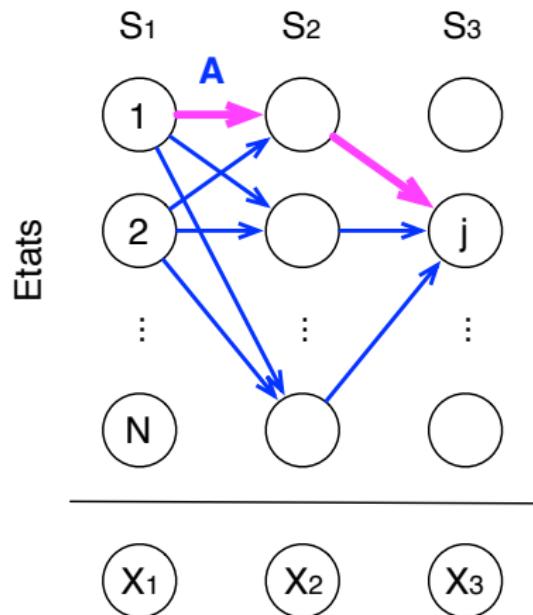
$$\delta_1(i) = \pi_i b_i(x_1)$$

- Récurrence :

$$\delta_t(j) = \left[\max_i \delta_{t-1}(i) a_{ij} \right] b_j(x_t)$$

PB2 : Viterbi (suite)

- δ = stockage de la probabilité de certaines situations...
- Ce qui nous intéresse c'est la séquence d'états associée
⇒ stockage à prévoir



- δ = stockage de la probabilité de certaines situations...
- Ce qui nous intéresse c'est la séquence d'états associée
⇒ stockage à prévoir
- Stockage des indices des états dans un tableau Ψ

$$\Psi_t(j) = \arg \max_{i \in [1, N]} \delta_{t-1}(i) a_{ij}$$

Pour arriver en j à t , quel était le meilleur état à $t - 1$? ??

- ⇒ A parcourir à l'envers pour retrouver le chemin optimal !

- δ = stockage de la probabilité de certaines situations...
- Ce qui nous intéresse c'est la séquence d'états associée
⇒ stockage à prévoir
- Stockage des indices des états dans un tableau Ψ

$$\Psi_t(j) = \arg \max_{i \in [1, N]} \delta_{t-1}(i) a_{ij}$$

- Complexité en $\mathcal{O}(N^2 T)$

PB2 : Viterbi (récapitulatif)

$$\delta_t(i) = \max_{s_1^{t-1}} p(s_1^{t-1}, s_t = i, x_1^t | \lambda)$$

➊ Initialisation

$$\begin{aligned}\delta_1(i) &= \pi_i b_i(x_1) \\ \Psi_1(i) &= 0\end{aligned}$$

➋ Récursion

$$\begin{aligned}\delta_t(j) &= \left[\max_i \delta_{t-1}(i) a_{ij} \right] b_j(x_t) \\ \Psi_t(j) &= \arg \max_{i \in [1, N]} \delta_{t-1}(i) a_{ij}\end{aligned}$$

➌ Terminaison

$$S^* = \max_i \delta_T(i)$$

➍ Chemin

$$\begin{aligned}q_T^* &= \arg \max_i \delta_T(i) \\ q_t^* &= \Psi_{t+1}(q_{t+1}^*)\end{aligned}$$