Compte-rendu du devoir 3

Vincent Matthys

1 HMM implementation

1.0 Notations

On désignera par M le nombre d'états cachés (4 dans notre cas). On notera $\boldsymbol{u} = \{u_t\}_{t\in[\![1;T]\!]}$, $\boldsymbol{q} = \{q_t\}_{t\in[\![1;T]\!]}$, $\boldsymbol{\theta} = (\pi, A, \boldsymbol{\mu}, \boldsymbol{\Sigma})$ où $\boldsymbol{\mu} = \{\mu_k\}_{k\in[\![1;M]\!]}$ et $\boldsymbol{\Sigma} = \{\Sigma_k\}_{k\in[\![1;M]\!]}$. On utilisera également le *one-hot encoding* sur $\boldsymbol{q}$, de sorte que q_t^i, variable binaire, représente la i$^{\text{e}}$ composante de q_t.

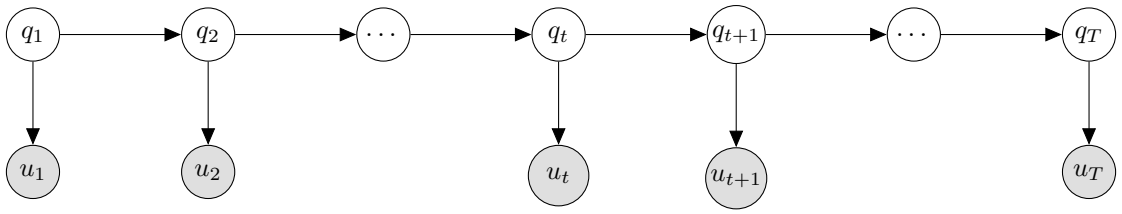


Figure 1 – Modèle graphique représentant le HMM considéré

1.1

Le problème de l'inférence dans la chaine de Markov modélisée, à savoir le calcul de $p(\boldsymbol{q} \mid \boldsymbol{u})$ est appréhendable par le calcul de la marginale $p(q_t \mid \boldsymbol{u})$ suivant :

$$
\begin{aligned}
p(q_t \mid \boldsymbol{u}) &= \frac{p(\boldsymbol{u} \mid q_t)p(q_t)}{p(\boldsymbol{u})} \\
&= \frac{p(u_1, \ldots, u_t \mid q_t)p(u_{t+1}, \ldots, u_T \mid q_t)p(q_t)}{p(\boldsymbol{u})} \quad u_1, \ldots, u_t \perp\!\!\!\perp u_{t+1}, \ldots, u_T \mid q_t \\
&= \frac{p(u_1, \ldots, u_t, q_t)p(u_{t+1}, \ldots, u_T \mid q_t)}{p(\boldsymbol{u})} \\
&= \frac{\alpha_t(q_t)\beta_t(q_t)}{p(\boldsymbol{u})} \\
&= \frac{\alpha_t(q_t)\beta_t(q_t)}{\sum_{q_t} \alpha_t(q_t)\beta_t(q_t)} \\
&= \frac{\alpha_t(q_t)\beta_t(q_t)}{\sum_{q_T} \alpha_T(q_T)} \quad \text{puisque} \quad p(\boldsymbol{u}) = \sum_{q_T} p(q_T, \boldsymbol{u}) = \sum_{q_T} \alpha_T(q_T)
\end{aligned}
\tag{1}
$$

avec $\alpha_t(q_t) = p(u_1, \ldots, u_t, q_t)$ et $\beta_t(q_t) = p(u_{t+1}, \ldots, u_T \mid q_t)$
On peut alors aussi exprimer $p(q_t, q_{t+1} \mid \boldsymbol{u})$ en fonction de α_t et β_t :

$$
\begin{aligned}
p(q_t, q_{t+1} \mid \boldsymbol{u}) &= \frac{p(\boldsymbol{u} \mid q_t, q_{t+1})p(q_t, q_{t+1})}{p(\boldsymbol{u})} \\
&= \frac{p(\boldsymbol{u} \mid q_t, q_{t+1})p(q_{t+1} \mid q_t)p(q_t)}{p(\boldsymbol{u})} \\
&= \frac{p(u_1, \ldots, u_t \mid q_t, q_{t+1})p(u_{t+1}, \ldots, u_T \mid q_t, q_{t+1})p(q_{t+1} \mid q_t)p(q_t)}{p(\boldsymbol{u})} \\
&= \frac{p(u_1, \ldots, u_t \mid q_t)p(u_{t+1}, \ldots, u_T \mid q_{t+1})p(q_{t+1} \mid q_t)p(q_t)}{p(\boldsymbol{u})} \\
&= \frac{p(u_1, \ldots, u_t \mid q_t)p(u_{t+1} \mid q_{t+1})p(u_{t+2}, \ldots, u_T \mid q_{t+1})p(q_{t+1} \mid q_t)p(q_t)}{p(\boldsymbol{u})} \\
&= \frac{p(u_1, \ldots, u_t, q_t)p(u_{t+1} \mid q_{t+1})p(u_{t+2}, \ldots, u_T \mid q_{t+1})p(q_{t+1} \mid q_t)}{p(\boldsymbol{u})} \\
&= \frac{\alpha_t(q_t)p(u_{t+1} \mid q_{t+1})\beta_t(q_{t+1})A_{q_t,q_{t+1}}}{p(\boldsymbol{u})}
\end{aligned}
\tag{2}
$$

où $A_{q_t,q_{t+1}} = p(q_{t+1} \mid q_t)$ et $p(u_{t+1} \mid q_{t+1})$ suit une loi gaussienne.

À des fins d'implémentation, on considèrera les logarithmes des α et β, $\tilde{\alpha}$ et $\tilde{\beta}$ de sorte que les formules de récursion peuvent se réecrire :

$$
\begin{cases}
\ln \alpha_t(q_t) = \ln p(u_t \mid q_t) + \ln \sum_{q_{t-1}} \left(e^{\ln \alpha_{t-1}(q_{t-1})} A_{q_{t-1},q_t} \right) \\
\ln \beta_t(q_t) = \ln \sum_{q_{t+1}} \left(e^{\ln \beta_{t+1}(q_{t+1})} A_{q_t,q_{t+1}} p(y_{t+1} \mid q_{t+1}) \right)
\end{cases}
\tag{3}
$$

$$\begin{cases} \tilde{\alpha}_t(q_t) = \ln p(u_t \mid q_t) + \ln \sum_{q_{t-1}} \left(e^{\tilde{\alpha}_{t-1}(q_{t-1})} A_{q_{t-1},q_t} \right) \\ \tilde{\beta}_t(q_t) = \ln \sum_{q_{t+1}} \left(e^{\tilde{\beta}_{t+1}(q_{t+1})} A_{q_t,q_{t+1}} p(y_{t+1} \mid q_{t+1}) \right) \end{cases} \tag{4}$$

et que la tâche d'inférence dite de *smoothing* en équation (1) s'écrit :

$$\begin{aligned} \ln p(q_t \mid \boldsymbol{u}) &= \ln \alpha_t(q_t) + \ln \beta_t(q_t) - \ln \sum_{q_T} \alpha_T(q_T) \\ \ln p(q_t \mid \boldsymbol{u}) &= \tilde{\alpha}_t(q_t) + \tilde{\beta}_t(q_t) - \ln \sum_{q_T} e^{\tilde{\alpha}_T(q_T)} \end{aligned} \tag{5}$$

La tâche d'inférence dite de *filtering* est également ajoutée en normalisant α_t :

$$\alpha_t^{\text{norm}}(q_t) = p(q_t \mid u_1, \ldots, u_t) = \frac{\alpha_t(q_t)}{\sum_{q_t} \alpha_t(q_t)}$$

et est disponible par le calcul des α^{norm} dans le code fourni.

1.2

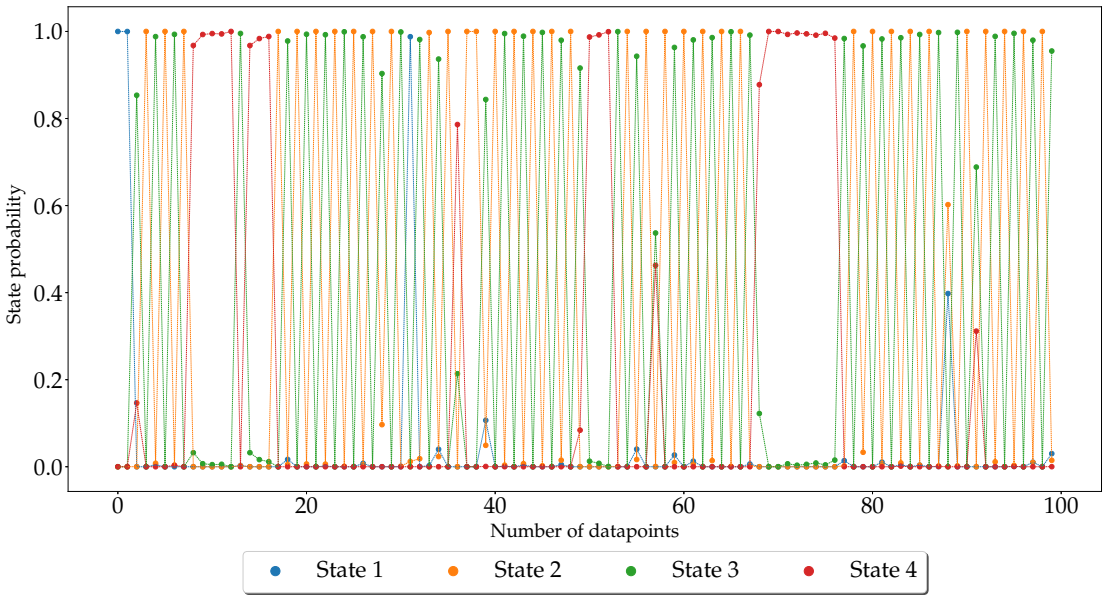


FIGURE 2 – Smoothing result for the first 100 datapoints

La figure donnant le résulat du smoothing des 100 premier points est présentée en figure 2. On y constate notamment une alternance importante entre les états 1 et 2, qui représentent également la majorité des états les plus probables.

1.3

On exprime la log-vraisemblance incomplète de la manière suivante :

$$\begin{aligned}
\ell(\boldsymbol{\theta}; \boldsymbol{u}) &= \ln \sum_{\boldsymbol{q}} p\left(\boldsymbol{q}, \boldsymbol{u} \mid \boldsymbol{\theta}\right) \\
&= \ln \sum_{\boldsymbol{q}} r(\boldsymbol{q} \mid \boldsymbol{u}) \frac{p\left(\boldsymbol{q}, \boldsymbol{u} \mid \boldsymbol{\theta}\right)}{r(\boldsymbol{q} \mid \boldsymbol{u})} \quad \text{pour r quelconque} \\
&\geq \sum_{\boldsymbol{q}} r(\boldsymbol{q} \mid \boldsymbol{u}) \ln \frac{p\left(\boldsymbol{q}, \boldsymbol{u} \mid \boldsymbol{\theta}\right)}{r(\boldsymbol{q} \mid \boldsymbol{u})} \quad \text{par l'inégalité de Jensen} \\
&\geq \mathbb{E}_r[\ell_c(\boldsymbol{\theta}; \boldsymbol{q}, \boldsymbol{u})] - H(r(\boldsymbol{q} \mid \boldsymbol{u})) = \mathcal{L}(r, \boldsymbol{\theta}, \boldsymbol{u})
\end{aligned} \tag{6}$$

avec $\ell_c(\boldsymbol{\theta}; \boldsymbol{q}, \boldsymbol{u})$, log-vraisemblance complète, dont l'expression est la suivante :

$$\begin{aligned}
\ell_c(\boldsymbol{\theta}; \boldsymbol{q}, \boldsymbol{u}) &= \ln p\left(\boldsymbol{q}, \boldsymbol{u} \mid \boldsymbol{\theta}\right) \\
&= \sum_{t=1}^{T} \ln p(u_t \mid q_t, \boldsymbol{\mu}, \boldsymbol{\Sigma}) + \sum_{t=2}^{T} \sum_{i=1}^{M} \sum_{j=1}^{M} q_{t-1}^i q_t^j \ln A_{ij} + \sum_{i=1}^{M} q_1^i \ln \pi_i \\
&= \sum_{t=1}^{T} \sum_{k=1}^{M} q_t^k \left(-\ln 2\pi - \frac{1}{2} \ln |\Sigma_k| - \frac{1}{2} (u_t - \mu_k)^\intercal \Sigma_k^{-1} (u_t - \mu_k) \right) \\
&+ \sum_{t=2}^{T} \sum_{i=1}^{M} \sum_{j=1}^{M} q_{t-1}^i q_t^j \ln A_{ij} + \sum_{i=1}^{M} q_1^i \ln \pi_i
\end{aligned} \tag{7}$$

On a donc une borne inférieure de la log-vraisemblance incomplète, $\mathcal{L}(r, \boldsymbol{\theta}, \boldsymbol{u})$. L'algorithme EM consiste en une montée alternée de cette borne inférieure par rapport à r et $\boldsymbol{\theta}$. En choisissant $r(\boldsymbol{q} \mid \boldsymbol{u}) = p(\boldsymbol{q} \mid \boldsymbol{u}, \boldsymbol{\theta})$, on maximise celle-ci, de sorte que les deux étapes de l'algorithme se résument de la manière suivante :

$$\begin{aligned}
E - step \quad & r^{(s+1)} = \arg\max_r \mathcal{L}(r, \boldsymbol{\theta}^{(s)}, \boldsymbol{u}) = p(\boldsymbol{q} \mid \boldsymbol{u}, \boldsymbol{\theta}^{(s)}) \\
M - step \quad & \boldsymbol{\theta}^{(s+1)} = \arg\max_{\boldsymbol{\theta}} \mathcal{L}(r^{(s+1)}, \boldsymbol{\theta}, \boldsymbol{u}) = \arg\max_{\boldsymbol{\theta}} \mathbb{E}_{r^{(s+1)}}(\ell_c(\boldsymbol{\theta}; \boldsymbol{q}, \boldsymbol{u}))
\end{aligned} \tag{8}$$

où s dénote le nombre d'itérations de l'algorithme.

D'après l'équation (7), on observe que $m_{ij} \coloneqq \sum_{t=2}^{T} q_{t-1}^i q_t^j$ est la statistique exhaustive pour A_{ij}, que q_1^i est la statistique exhaustive pour π_i. Les estimateurs de maximum de log-vraisemblance complète pour A et π sont donc :

$$\widehat{\pi}_i = q_1^i \tag{9}$$

$$\widehat{A}_{ij} = \frac{m_{ij}}{\sum_{k=1}^{M} m_{ik}} \tag{10}$$

D'autre part, grâce aux calculs déjà effectués pour les estimateurs de vraisemblance de la mixture de gaussiennes, on obtient les estimateurs de log-vraisemblance complète des

paramètres d'émission :

$$\widehat{\mu_k} = \frac{\sum_{t=1}^{T} q_t^k u_t}{\sum_{t=1}^{T} q_t^k} \tag{11}$$

$$\widehat{\Sigma_k} = \frac{\sum_{t=1}^{T} q_t^k \left(u_t - \mu_k\right)\left(u_t - \mu_k\right)^\intercal}{\sum_{t=1}^{T} q_t^k} \tag{12}$$

Ces estimateurs doivent être pris sous l'espérance précise obtenue lors de l'étape E de l'algorithme EM, donnée par l'équation (8), afin que la maximisation de la log-vraisemblance complète induise une augmentation de la log-vraisemblance incomplète lors de l'itération $s+1$.

En reprenant l'estimateur de π obtenu en équation (9), on obtient l'estimateur à l'itération $(s+1)$:

$$\begin{aligned} \widehat{\pi}_i^{(s+1)} &= \mathbb{E}_{\boldsymbol{q}|\boldsymbol{u},\boldsymbol{\theta}^{(s)}}[q_1^i] \\ &= \mathbb{E}_{q_1|\boldsymbol{u},\boldsymbol{\theta}^{(s)}}[q_1^i] \\ &= p(q_1^i = 1 \mid \boldsymbol{u}, \boldsymbol{\theta}^{(s)}) \coloneqq \gamma_1^{i,(s+1)} \end{aligned} \tag{13}$$

En reprenant l'estimateur de A_{ij} obtenue en équation (10), il faut déterminer $\mathbb{E}_{\boldsymbol{q}|\boldsymbol{u},\boldsymbol{\theta}^{(s)}}[m_{ij}]$. On a :

$$\begin{aligned} \mathbb{E}_{\boldsymbol{q}|\boldsymbol{u},\boldsymbol{\theta}^{(s)}}[m_{ij}] &= \sum_{t=2}^{T} \mathbb{E}_{\boldsymbol{q}|\boldsymbol{u},\boldsymbol{\theta}^{(s)}}\left[q_{t-1}^i q_t^j\right] \\ &= \sum_{t=2}^{T} p\left(q_{t-1}^i = 1, q_t^j = 1 \mid \boldsymbol{u}, \boldsymbol{\theta}^{(s)}\right) \\ &\coloneqq \sum_{t=2}^{T} \xi_{t-1,t}^{ij,(s+1)} \end{aligned}$$

D'où l'estimateur à l'itération $(s+1)$ pour A :

$$\widehat{A}_{ij}^{(s+1)} = \frac{\sum_{t=2}^{T} \xi_{t-1,t}^{ij,(s+1)}}{\sum_{k=1}^{M}\sum_{t=2}^{T} \xi_{t-1,t}^{ik,(s+1)}} = \frac{\sum_{t=2}^{T} \xi_{t-1,t}^{ij,(s+1)}}{\sum_{t=2}^{T} \gamma_{t-1}^{i,(s+1)}} \tag{14}$$

De la même façon, on obtient les estimateurs à l'itération $(s+1)$ pour les μ_k et Σ_k :

$$\begin{aligned} \widehat{\mu_k}^{(s+1)} &= \frac{\sum_{t=1}^{T} \gamma_t^{k,(s+1)} u_t}{\sum_{t=1}^{T} \gamma_t^{k,(s+1)}} \quad \forall k \in [\![1; M]\!] \\ \widehat{\Sigma_k}^{(s+1)} &= \frac{\sum_{t=1}^{T} \gamma_t^{k,(s+1)} \left(u_t - \mu_k^{(s)}\right)\left(u_t - \mu_k^{(s)}\right)^\intercal}{\sum_{t=1}^{T} \gamma_t^{k,(s+1)}} \quad \forall k \in [\![1; M]\!] \end{aligned} \tag{15}$$

L'implémentation du calcul des γ et ξ est succintement détaillée et fait intervenir le

calcul des α^{norm} du problème de *filtering*.

$$\begin{aligned}
\gamma_t(q_t) &= p(q_t \mid u_1, \ldots, u_T) \\
&= \sum_{q_{t+1}} p(q_t, q_{t+1} \mid u_1, \ldots, u_T) \\
&= \sum_{q_{t+1}} \frac{\alpha_t(q_t) A_{q_t,q_{t+1}}}{\sum_{q_t} \alpha_t(q_t) A_{q_t,q_{t+1}}} \gamma_{t+1}(q_{t+1}) \\
&= \sum_{q_{t+1}} \frac{\alpha_t^{\text{norm}}(q_t) A_{q_t,q_{t+1}}}{\sum_{q_t} \alpha_t^{\text{norm}}(q_t) A_{q_t,q_{t+1}}} \gamma_{t+1}(q_{t+1})
\end{aligned} \tag{16}$$

$$\begin{aligned}
\xi_{t,t+1}(q_t, q_{t+1}) &= p(q_t, q_{t+1} \mid \boldsymbol{u}) \\
&= \frac{\alpha_t(q_t) p(u_{t+1} \mid q_{t-1}) \gamma_{t+1}(q_{t+1}) A_{q_t,q_{t+1}}}{\alpha_{t+1}(q_{t+1})} \\
&= \frac{\alpha_t^{\text{norm}}(q_t) p(u_{t+1} \mid q_{t-1}) \gamma_{t+1}(q_{t+1}) A_{q_t,q_{t+1}}}{\alpha_t^{\text{norm}}(q_{t+1})} \times \frac{\sum_{q_t} \alpha_t(q_t)}{\sum_{q_{t+1}} \alpha_{t+1}(q_{t+1})}
\end{aligned} \tag{17}$$

Au niveau de l'implémentation, on notera qu'on utilise les α^{norm} qui ne changent aucunement la formule pour γ. Pour ξ, il faut prendre en compte la différence de normalisation entre les α^{norm}.

1.4

La partie d'implémentation est effectuée toujours en python dans la classe *HMM*.

1.5

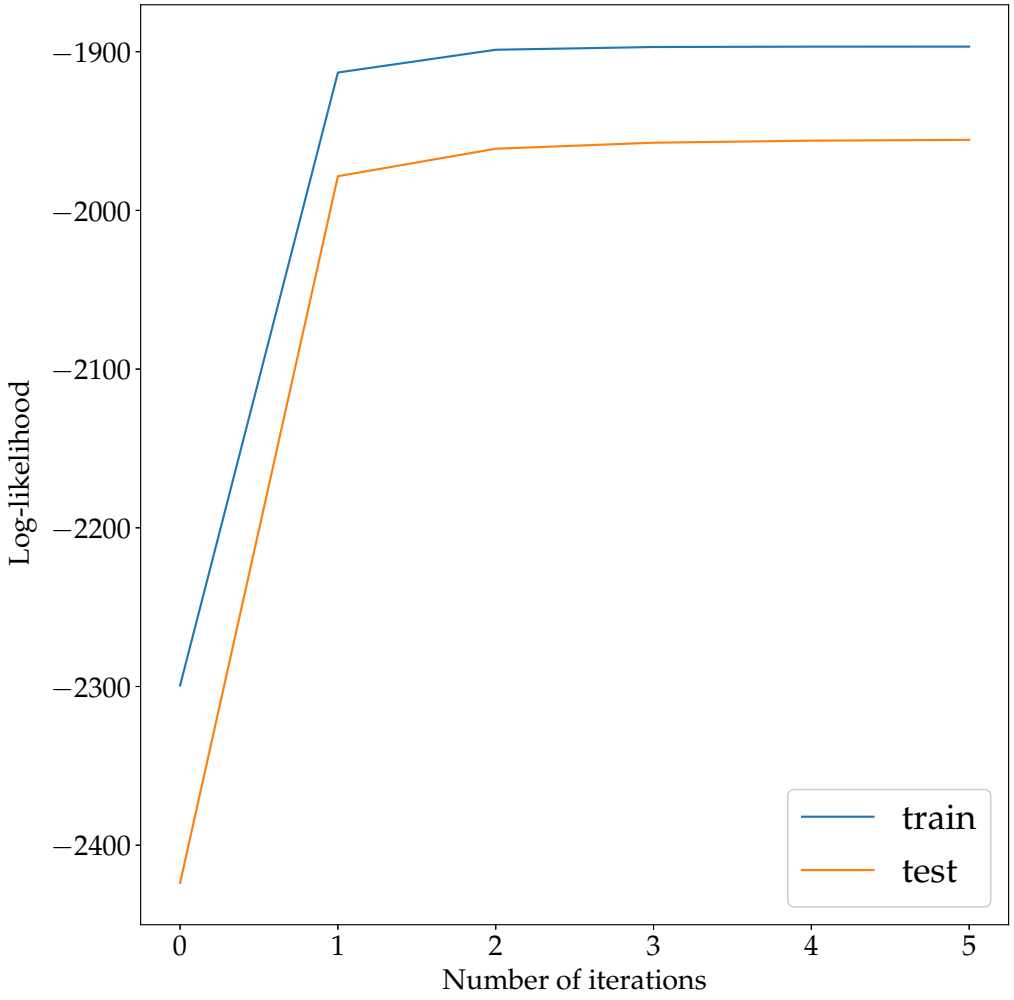


FIGURE 3 – Evolution de la log-vraisemblance en fonction du nombre d'itérations de EM pour le modèle de HMM

La log-vraisemblance en apprentissage et en test est représentée en figure 3. On constate qu'il suffit de 2-3 itérations de l'algorithme EM pour converger vers de nouveaux paramètres, en partant de ceux déterminés par mixture de gaussienne. On constate également que la vraisemblance en apprentissage est toujours supérieure à la vraisemblance de test, ce qui constitue un résultat attendu.

Modèle	Donées d'entraînement	Données de test
isotropic GM	-2.6396	-2.6146
general GM	-2.3277	-2.4092
HMM	**−1.8968**	**−1.9555**

TABLE 1 – Valeur $\times 10^3$ de la log-vraiemblance des données d'apprentissage et des données de test pour différents modèles. GM signifie Gaussian Mixture (isotropic pour des covariances diagonales, general pour des covariances quelconques), HMM signifie Hidden Markov Model

1.6

Les valeurs de log-vraisemblance des différents modèles rencontrés sont présentées en table 1. On constate que le modèle avec la meilleure log-vraisemblance est le modèle chaine de Markov, suivi du modèle de mixture de gaussiennes générales. C'est un résultat attendu, puisque l'odre correspond avec le degré de complexité du modèle : plus celui-ci est complexe (c'est-à-dire avec de degré de liberté), plus il peut coller aux données. On constate également que la log-vraisemblance sur les données de test est inférieure à celle obtenue sur les données d'entrainement, à l'exception du modèle de mixture gaussiennes isotropiques pour lequel la différence observée est minime entre les données d'entrainement et de test. On notera cette vraisemblance de GM isotropic en test est la seule vraisemblance différente de celle notée dans la solution du devoir 2 fourni par M. Obozinski, dans laquelle la vraissemblance en question est de $-2.6841\ 10^3$ et est bien inférieure à celle obtenue pour le même modèle sur les données d'entrainement.

1.7

L'algorithme de décodage de Viterbi permet de déterminer la séquence d'états la plus probable $\boldsymbol{q} = q_1, \ldots, q_T$ étant donnée une séquence d'observation $\boldsymbol{u} = u_1, \ldots, u_t$ pour des paramètres de modèle donnés $\boldsymbol{\theta}$. Au lieu de calculer la log-vraisemblance d'une séquence d'états étant donnée les observations, ce qui reviendrait à faire une α récursion pour chaque séquence d'états, on utilise l'algorithme $max - product$ sur le graphe du modèle de Markov en figure 1, en prenant comme racine de l'arbre le noeud q_1. Ainsi, on obtient la relation de récurence suivante :

$$\delta_{t\to t+1}(q_{t+1}) = \max_{q_t} \left(\delta_{(t-1)\to t}(q_t) A_{q_t,q_{t+1}}\right) p(u_{t+1} \mid q_{t+1}) \tag{18}$$

$$\log \delta_{t\to t+1}(q_{t+1}) = \max_{q_t} \left(\log \delta_{(t-1)\to t}(q_t) + \log A_{q_t,q_{t+1}}\right) + \log p(u_{t+1} \mid q_{t+1}) \tag{19}$$

Une fois estimée ce message pour chaque point, on considère la succession d'états qui a permis d'obtenir ce message, notée $\boldsymbol{q}^* = \arg\max_{\boldsymbol{q}} p(\boldsymbol{q} \mid \boldsymbol{u})$, que l'on obtient en partant du noeud q_T :

$$q_T^* = \arg\max_{q_T} \delta_{(T-1)\to T}(q_T) \tag{20}$$

$$q_t^* = \arg\max_{q_t} \delta_{(t-1)\to t}(q_t) A_{q_t, q_{t+1}^*} \quad \forall t \in [\![1; T-1]\!] \tag{21}$$

D'où le pseudo-code suivant :

Algorithm 1: Viterbi decoding

Result: $\boldsymbol{q}^*$

$\delta_1(j) = \pi_j p(u_1 \mid q_1 = j)$;
for $i = 2$ *to* T **do**
  $\delta_t(j) = \max_i (\delta_{t-1}(i) + \log A_{i,j}) + \log p(u_t \mid q_t = j)$;
end
$q_T^* = \arg\max_j \delta_T(j)$;
for $i = T-1$ *to 1* **do**
  $q_t^* = \arg\max_j \delta_t(j) A_{j, q_{t+1}^*}$;
end

1.8

Le résultat du décodage de Viterbi sur les 100 premiers points des données d'entrainement avec les paramètres du modèle HMM obtenus par EM sur ces mêmes données est présenté en figure 4. On a également superposé les centres des clusters ainsi que l'ellipsiode correspondant à 90 % de la masse de chaque gaussienne associée. On constate que l'attribution des points à chaque cluster est cohérent avec les résultats précédemment obtenus lors de la modélisation par mixture de gaussiennes générales.

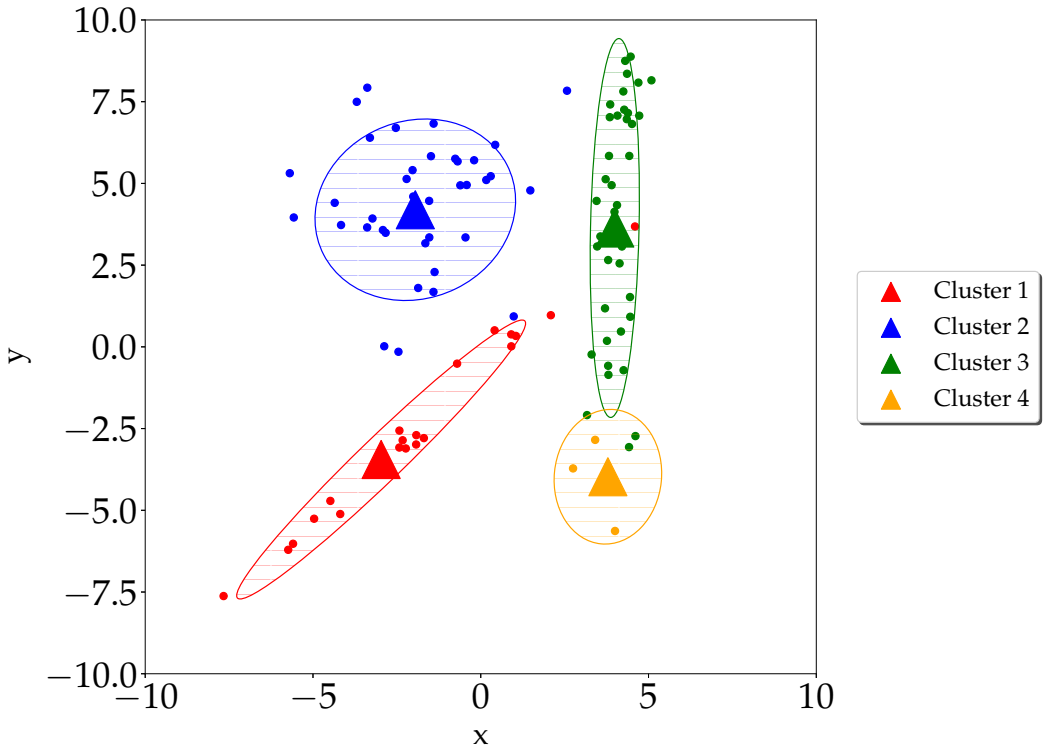


FIGURE 4 – Visualisation de la séquence d'états la plus probable sur les 100 premiers points du jeu d'entrainement obtenu par décodage de Viterbi

1.9

Le calcul des probabilités marginales de chaque état q_t étant donnée la séquence complète des observations $\boldsymbol{u}$ a été décrite en équation (1) lors de l'$\alpha - \beta$ récursion. On utilise les paramètres α et β obtenus après convergence de EM sur les données d'entrainement, et on représente les marginales de chacun des 4 états des 100 premiers points du jeu de test en figure 5. On peut y observer une alternance presque exclusive des états 2 et 3.

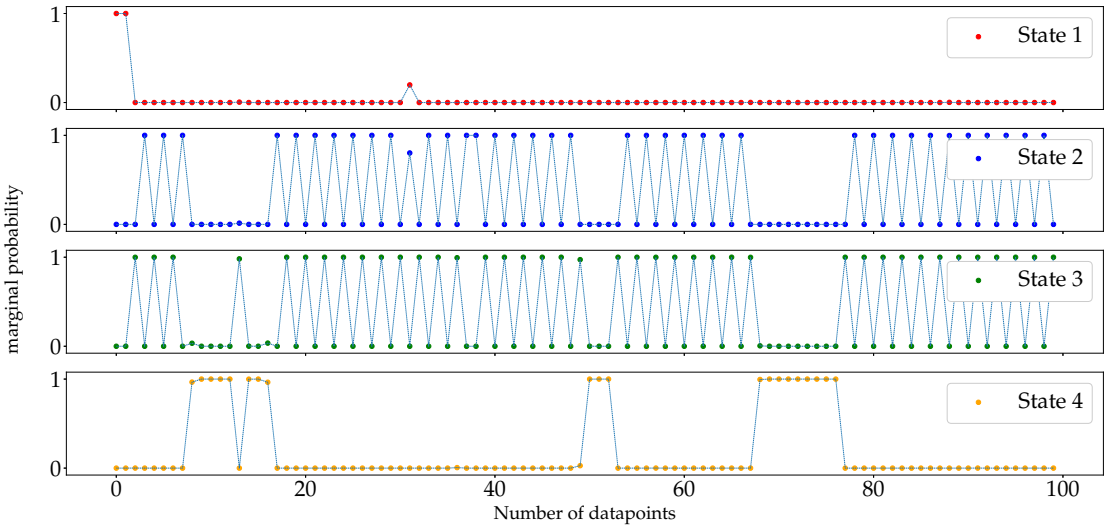


FIGURE 5 – Probabilités marginales des états sur le jeu de test pour les 100 premiers points

1.10

L'état le plus probable pour chacun de ces mêmes points est représenté en figure 6. On y constate clairement l'alternance entres les états 2 et 3, entrecoupés de périodes de 2 à 9 points en état 4.

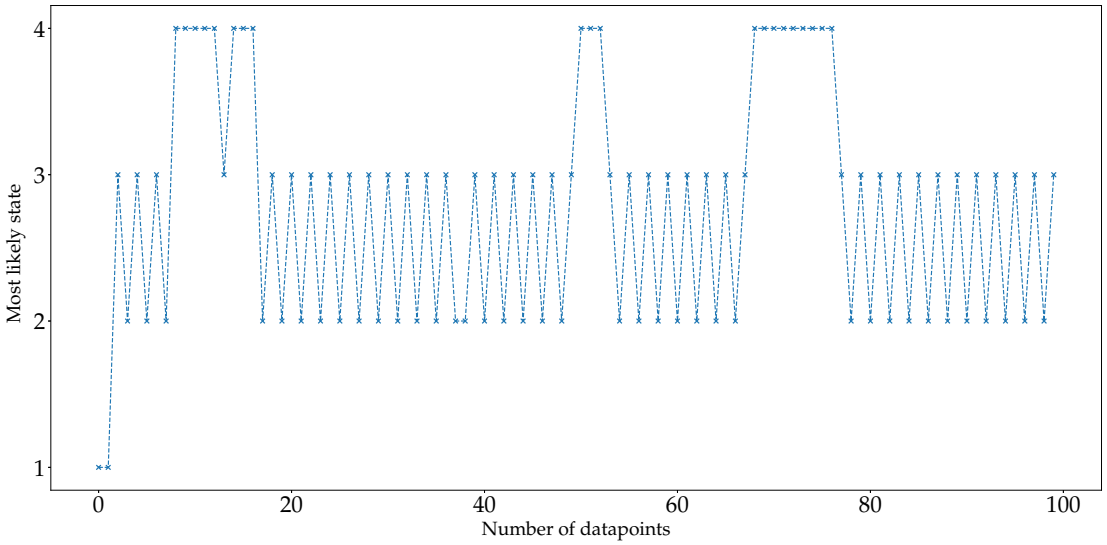
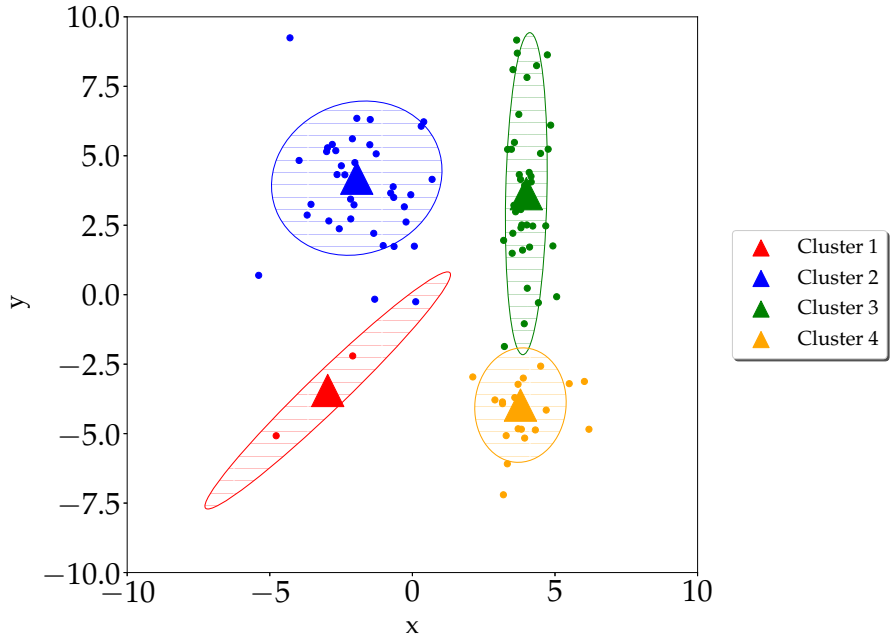
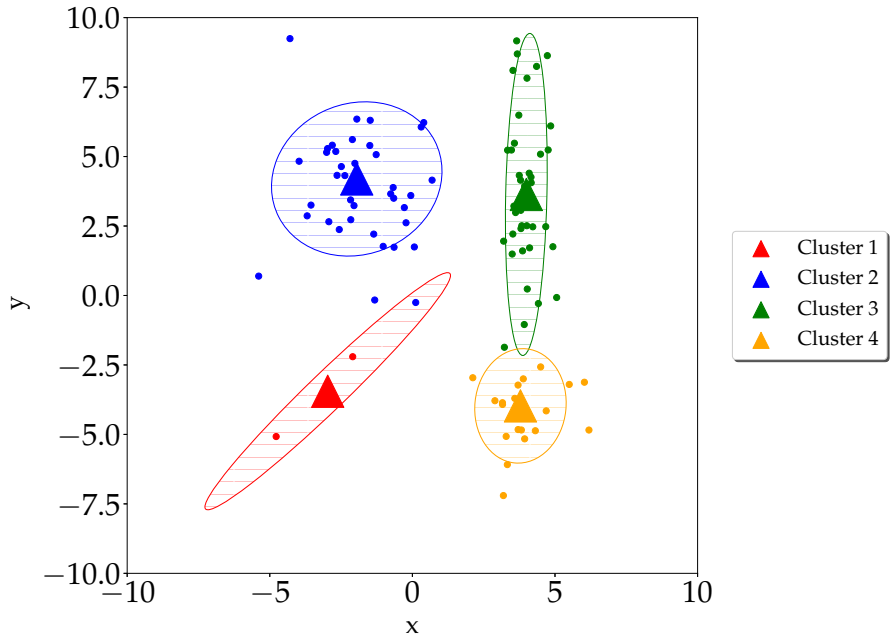


FIGURE 6 – Etat le plus probable pour chacun des 100 premiers points du jeu de test

1.11



(a) Décodage selon Viterbi



(b) Décodage selon les probabilités marginales

FIGURE 7 – Décodage des 100 premiers points du jeu de test suivant Viterbi ou les probabilités marginales

Les décodages obtenus par Vierbi et par les probabilités marginales pour les 100 premiers points du jeu de test sont présentés en figure 7. On remarque que les décoadges par les deux méthodes sont identiques.

1.12

Si le nombre d'états n'était pas connu, il faudrait faire une sélection de modèle en comparant la log-vraisemblance obtenue par les différents modèles HMM (à différents nombre d'états) et conserver le modèle donnant la meilleure vraisemblance en test.