Pattern recognition system
Exercise Project

Report Assignment n°1

Student
Antoine Honoré
honore@kth.se

Student
Audrey Brouard
brouard@kth.se

2015 Period 1

I

Let us calculate $P(S_t = j)\ \forall j \in \{1, 2\}$ and for t = 1,2,3,...

We will do it for t = 1 and t = 2, and notice that the probabilities are constant $\forall t$.

— For t = 1, the probabilities are given by the matrix q_j. Thus, $P(S_1 = 1) = 0.75$ and $P(S_1 = 2) = 0.25$.

$\forall t \geq 2$, we will use the following formula :

— $P(S_t = j) = \sum_{i=1}^{2} P(S_t = j, S_{t-1} = i) = \sum_{i=1}^{2} P(S_t = j|S_{t-1} = i)P(S_{t-1} = i)$

For the case t = 2, we have :

— $P(S_t = j) = \sum_{i=1}^{2} a_{ij}q_i$

We thus obtain $P(S_2 = 1) = 0.75$ and $P(S_2 = 2) = 0.25$. We immediately notice that

$$\forall i\ P(S_2 = i) = P(S_1 = i)$$

By recurrency, we have $\forall t\ \Pr(S_t = j)$ constant.

II

After we generated 10 000 state integers, we found the following probabilities :

$$\Pr(S_t = 1) = 0.7525\ and\ \Pr(S_t = 2) = 0.2475$$

III @HMM/rand

III.1 Theorical calculation

Let us now calculate $E[X_t]$ and $Var[X_t]$.

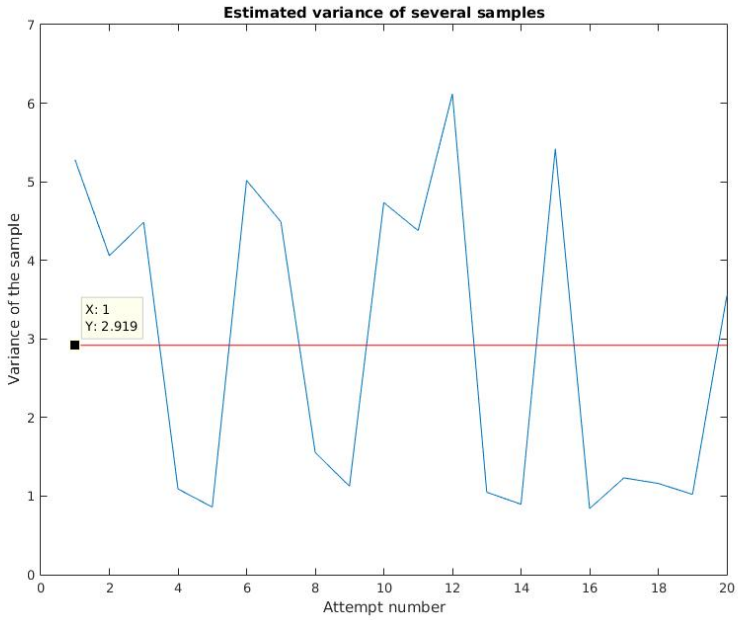
— $E[X_t]$: the book gives the formula $E[X] = E_S[E_X[X|S]]$, and according to the two different possible values of S, X density probability function is either b_1 or b_2.
Then, for j = 1, $E[X|S] = \mu_1$; and for j = 2, $E[X|S] = \mu_2$.
$E[X] = P(S = 1) * \mu_1 + P(S = 2) * \mu_2 = 0 + 3 * 0.25 = 0.75$.

— $Var[X_t]$: according to the book, $Var[X_t] = E_S[Var_X[X|S]] + Var_S[E_X[X|S]]$.
Thanks to the same observation as before, the expression becomes :
$Var[X_t] = E_S[Var_X[X|S]] + E_S[(E_X[X|S])^2] - E_S[E_X[X|S]]^2$
$Var[X_t] = [0.75 * \sigma_1^2 + 0.25 * \sigma_2^2] + [0.75 * \mu_1^2 + 0.25 * \mu_2^2] - 0.75^2 = 1.75 + 0.25 * 9 - 0.75^2 = 3.4375$.

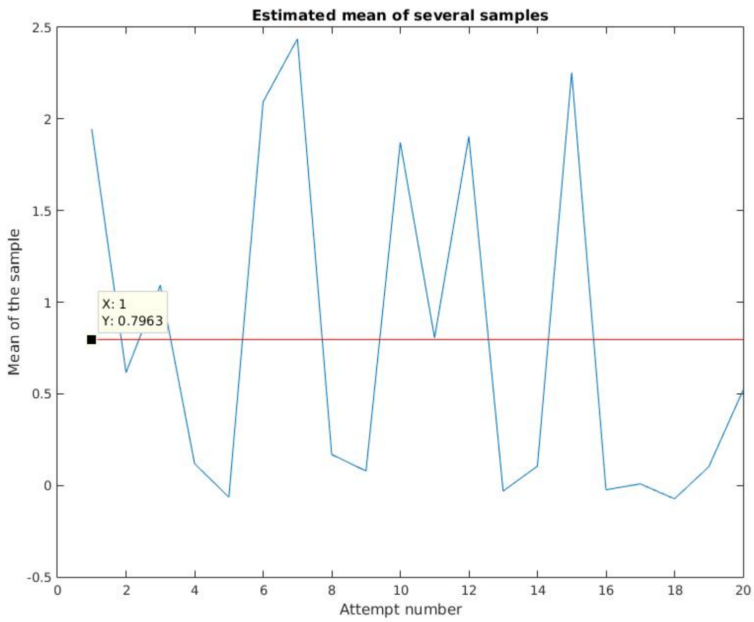
III.2 Measures

One point on a graph represent the mean (figure 1b) and the variance (figure 1a) of T=10000 scalar output. There is 20 points on each graph. This shows that the experimental values are close from the theoretical ones

As we can see on the figures the experimental values are close from the theoretical ones.



(a) Variance of 10000 samples computed 20 times $E(Var(X)) \simeq 2.919$



(b) Mean of 10000 samples computed 20 times $E(E(X)) \simeq 0.7963$

Figure 1 – Validation of @HMM/rand
Blue : Plot of the 20 attempts
Red : Mean over the 20 attempts

IV HMM behavior 1/2

For this part, we changed the behavior of the @HMM/rand function, in order to get a vector from b_1 (not a scalar like the previous question). So here, each sample is a vector $x_t \sim N(\mu_j, \sigma_j^2)$ of size 500 (where j=1 or 2).

The following figure shows the 500 scalars from the process.

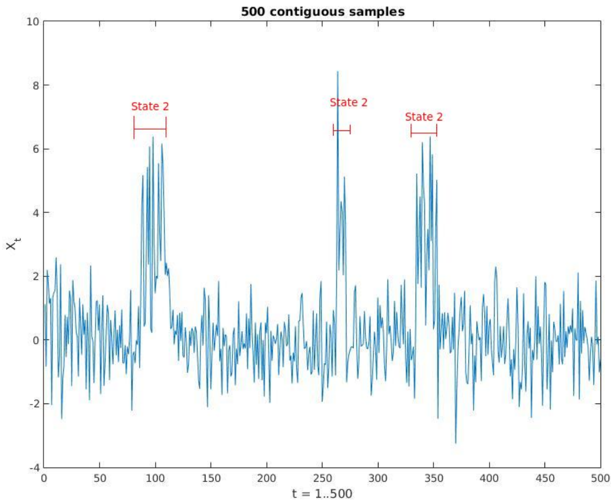


FIGURE 2 – Plot of 500 contiguous X_t, the two states are defined as follow : $b_1(x) = N(0, 1)$ and $b_2(x) = N(3, 2)$

Observation

After repeating the same code used to plot figure 2 several times, we confirmed the well behaving of the HMM. It is for likely to begin in state 1, and the system succeed in jumping into state 2, it will spend on this state than in state 1. When the means are different, it is pretty easy to find out the state sequence.

V HMM behavior 2/2

We now modify the mean of b_2, μ_2 so that it is equal to μ_1 : $\mu_1 = \mu_2 = 0$.

We can easily notice a change with the previous HMM. On the figure below are represented a sample produced while the state was 1 in blue, and when the state was 2 in red :

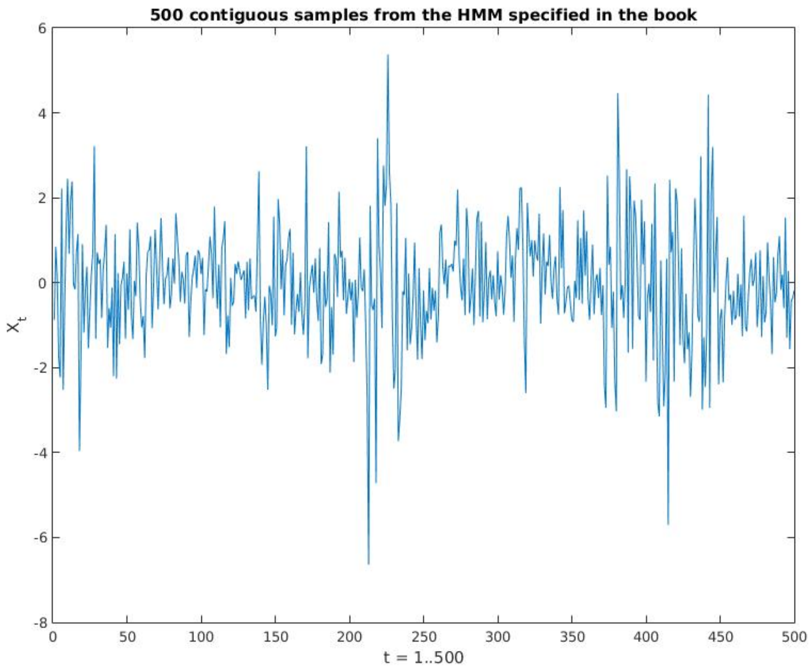


FIGURE 3 – Plot of 500 contiguous X_t, the two states are defined as follow : $b_1(x) = N(0, 1)$ and $b_2(x) = N(0, 2)$

Observation

The sequence of state should be equivalent [1] as the previous configuration. But only by considering the samples, when both means are equal, it is far more difficult to infer the state sequence. We can assume that the samples "far" from the mean are produced in state 2 (since the standard deviation is greater in state 2) but it is not a reliable assumption.

VI Finite-duration HMMs

The implementation works with finite duration Markov chain.

We defined a transition matrix $A = \begin{pmatrix} 0.6 & 0.3 & 0.1 \\ 0.5 & 0.4 & 0.1 \end{pmatrix}$ for a Markov chain, then a HMM based on this Markov chain, and the call to @HMM/rand returns the correct information. The difference is that we don't know how long the sequence is since we don't know precisely when the system will jump in the exit state (state 3). Here are three examples of several state sequences generated with three calls to @HMM/rand :

$S_1 = [12211212211211111123]$
$S_2 = [1112111222122211111121211111212111111222111122113]$
$S_3 = [1212112212213]$

1. in term of probability of jumping from one state to another.

VII Output vectors

The @HMM/rand function has been tested and works good when the Output Distributions returns both vectors and scalars.