EQ2341 Pattern Recognition and Machine Learning Assignment 1

Ayushi Shah
ayushi@kth.se

Sri Janani Rengarajan
sjre@kth.se

April 16, 2020

Task 1

$$\begin{aligned} P[S_{t+1} = j] &= \sum_{t=1}^{N} P[S_{t+1} = j \cap S_t = i] \\ &= \sum_{t=1}^{N} \underbrace{P[S_{t+1} = j | S_t = i]}_{a_{ij}} P[S_t = i] \\ &= A^T p \end{aligned} \tag{1}$$

The initial probability of each state q = $\begin{bmatrix} 0.75 \\ 0.25 \end{bmatrix}$ and the state transition matrix $\mathbf{A} = \begin{bmatrix} 0.99 & 0.01 \\ 0.03 & 0.97 \end{bmatrix}$ are given. Using equation 1, probability for each state at t=1 is calculated as $p_1 = A^T * q = \begin{bmatrix} 0.75 \\ 0.25 \end{bmatrix}$. At t=2, $p_2 = A^T * p_1 = p_1$. Hence, $p_t = p_1 = p_2 = ... = p_{t-1}$, which is a stationary state distribution. It can also be proven that the given state distribution **q** is stationary by computing the eigen values of A^T and eigen vector corresponding to the eigen value 1. Eigen values of A^T = [1,0.96], eigen vector corresponding to eigen value 1 is **v**= $\begin{bmatrix} 0.9487 \\ 0.3162 \end{bmatrix}$. v is the stationary state distribution when scaled by a factor of 0.7906, p_t = 0.7906 * v.

Task 2

Using the markov chain rand function, a state sequence with 10000 values was generated. The state sequence consists of integers 1 and 2, indicating the states S_1 and S_2 respectively.

Number of times integer 1 occurred = 7341
Number of times integer 2 occurred = 2659
Relative frequency for State 1 = P[S = 1] = 0.7341
Relative frequency for State 2 = P[S = 2] = 0.2659

The relative frequency is close to the stationary state distribution p = $\begin{bmatrix} 0.75 \\ 0.25 \end{bmatrix}$

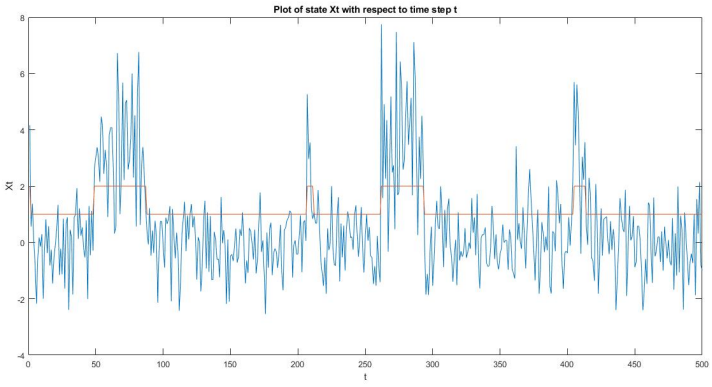

Figure 1: Sequence of 500 contiguous samples from the HMM

Task 3

Theoretical calculation for $E[X_t]$ and $var[X_t]$, where X_t is conditioned on S_t, is as follows:

$$\begin{aligned} E[X_t] &= E_{S_t}[E_{X_t}[X_t|S_t]] \\ &= P(S_t = 1).\mu_1 + P(S_t = 2).\mu_2 \\ &= 0.75 * 0 + 0.25 * 3 = 0.75 \end{aligned} \tag{2}$$

$$\begin{aligned} var[X_t] &= E_{S_t}[var_{X_t}[X_t|S_t]] + var_{S_t}[E_{X_t}[X_t|S_t]] \\ &= E_{S_t}[var_{X_t}[X_t|S_t]] + E[(E[X_t|S_t] - E[X_t])^2] \\ &= 0.75 * 1 + 0.25 * 4 + 0.75 * (0.75 - 0)^2 + 0.25 * (0.75 - 3)^2 \\ &= 3.4375 \end{aligned} \tag{3}$$

Mean and variance for the generated sequence of 10,000 output random scalars are 0.7626 and 3.4506 respectively.This result is close to the theoretical calculation 2,3.Therefore it can be concluded that the *HMM/rand* method works fine.

Task 4

Figure 1 is generated by HMM signal source. To understand the output of this HMM, we also plot the state transition overlapping on the HMM output.
Following behaviors are observed from the figure1.

- Because of the fact that the two sub-sources are Gaussian distributions with mean 0 and 3 respectively, samples X_t mainly oscillates around 0 and 3.
- We observed differences due to difference in variances.For Gaussian Distribution with $\mu_1 = 0, \sigma_1 = 1$ there is smaller jitter around 0 and for the gaussian distribution with $\mu_2 = 3, \sigma_2 = 2$ there is a large jitter around 3.
- With every state S_t transition X_t changes it value. The HMM tends to be trapped within a state for longer time as $P[S_{t+1} = 1|S_t = 1] = 0.99$ and $P[S_{t+1} = 2|S_t = 2] = 0.97$ for all t

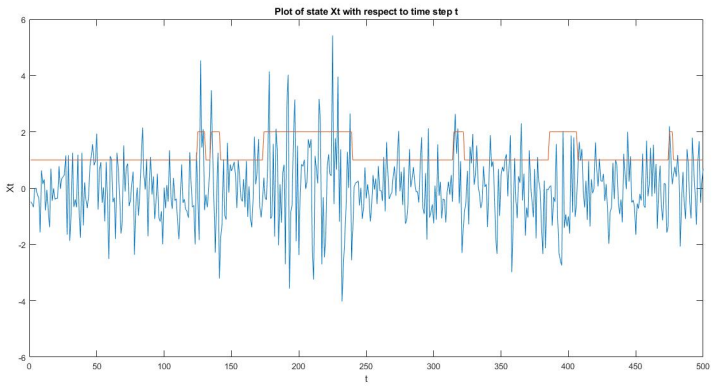

Figure 2: Sequence of 500 contiguous samples from the HMM with same mean

Task 5

Figure 2 is generated by HMM signal source which has $\mu_1 = 0\mu_2 = 0$. Similarity in both the HMM's is that they tend to be trapped within a state for longer time as $P[S_{t+1} = 1|S_t = 1] = 0.99$ and $P[S_{t+1} = 2|S_t = 2] = 0.97$ for all t. Also higher spread appears when state transits from 1 to 2 and a lower spread appears when state transits from 2 to 1, that is because of difference in the variances. Unlike the previous HMM, it is almost impossible to find the relation between outputs and states by pure observation. After 500 observations we notice that the average variance of the new sequence is much lower than the previous HMM, which is more like a white noise thus makes this harder to estimate the state sequence.

Task 6

To test our rand-function for *finite-duration* HMMs, we re-define our HMM source with the following parameters:

$$\text{q}=\begin{pmatrix}0.75\\0.25\end{pmatrix};\ \text{A}=\begin{pmatrix}0.5 & 0.4 & 0.1\\0.3 & 0.4 & 0.3\end{pmatrix};\ \text{B}=\begin{pmatrix}b_1(x)\\b_2(x)\end{pmatrix}$$

we run our program to generate a sequence of output with an expected length of 500 samples.To verify that our function returns reasonable results, we executed this HMM repeatedly, for 1000 times, and recording counts of different lengths of sequence when the process reaches the exit state, and then calculating the probabilities of exiting HMM process after different steps.
The statistical results are listed below:

```
Probability of occurrence of sequence of length 1 :0.163
Probability of occurrence of sequence of length 2 :0.14
Probability of occurrence of sequence of length 3 :0.135
Probability of occurrence of sequence of length 4 :0.097
Probability of occurrence of sequence of length 5 :0.092
Sequence of length 1 has maximum Probability of occurrence
```

Following is the code for *finite- duration* HMM.

```
mc=MarkovChain([0.75;0.25], [0.5 0.4 0.1;0.3 0.4 0.3]);%State generator
g1=GaussD("Mean",0,"StDev",1); %Distribution for state=1
g2=GaussD("Mean",3,"StDev",2); %Distribution for state=2
h=HMM(mc, [g1; g2]); %The HMM

lenX=zeros(1,1000); %Matrix to store the length of each sequence
for i = 1 : 1000
    [x,s]=rand(h, 100);
    lenX(i)=length(x);
end
prob=zeros(1,max(lenX));
for j = 1: max(lenX)
    prob(j)=length(find(lenX==j))/1000;
    if j<=5
        disp("Probability of occurence of sequence of length "+j+" :"+prob(j));
    end
end
maxProb=find(prob==max(prob));
disp("Sequence of length "+maxProb+ " has maximum Probability of occurence")
```

Task 7

To verify that our rand-function also works when the state-conditional output distributions generate random vectors, we change the output probability distributions into the following form:

$$\text{B}=\begin{pmatrix} b_1(x) \\ b_2(x) \end{pmatrix} b_1(x) \sim \mathcal{N}\left(\mu_1 = \begin{bmatrix} 0 \\ 10 \end{bmatrix}, C1 = \begin{bmatrix} 2 & 1 \\ 1 & 4 \end{bmatrix}\right), b_2(x) \sim \mathcal{N}\left(\mu_1 = \begin{bmatrix} 5 \\ 20 \end{bmatrix}, C2 = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix}\right)$$

We generate a sequence after 500 time steps, and obtain a 2x500 matrix as output since each sub-source is a two-dimensional Gaussian distribution.Figure 3 we can easily recognize state transitions based on value changes of both features. The two Gaussian distributions are well separated. Thus, we can say that our implementation of rand-function works well in this case.

Following is the code for HMM using 2-Dimensional Gaussian Distribution.

```
g1=GaussD("Mean",[0 ; 10],"Covariance",[2 1 ;1 4]);
g2=GaussD("Mean",[5 ; 20],"Covariance",[1 0 ;0 2]);

mc=MarkovChain([0.5;0.5], [0.9 0.1;0.05 0.95]);
h=HMM(mc, [g1; g2]);
[x,s]=rand(h,500);
t=[1 : 1: 500];
subplot(2,1,1);
plot(t,s);
title('Plot of state St with respect to time step t')
xlabel('t'); ylabel('St')
```

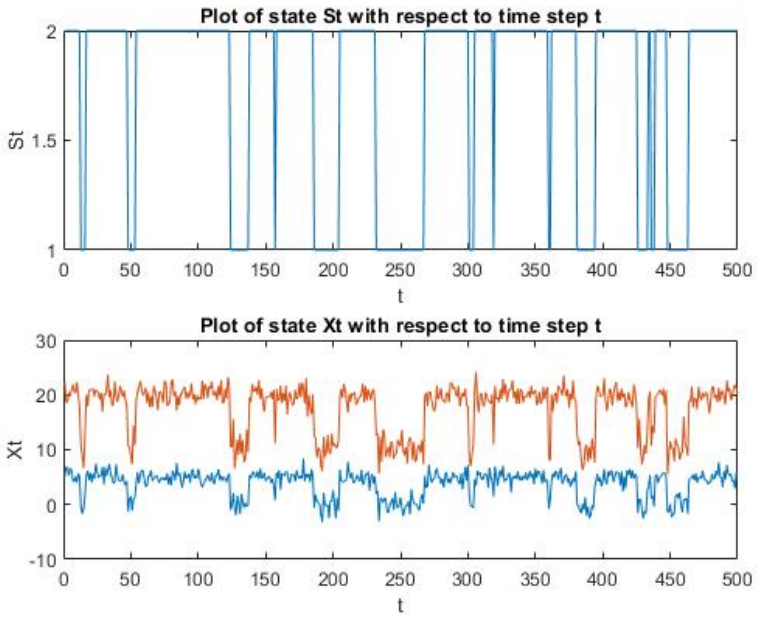

Figure 3: Sequence of 500 contiguous samples from the HMM with non-diagonal covariance matrix

```
subplot(2,1,2);
plot(t,x);
title('Plot of state Xt with respect to time step t')
xlabel('t'); ylabel('Xt')
```