EQ2341 Pattern Recognition and Machine Learning
Assignment 1

Boyue Jiang
boyue@kth.se

Ziyue Yang
ziyuey@kth.se

May 03, 2024

Questions 2

We need to confirm that the state probabilities of the Markov chain do not change over time, suggesting that the initial state distribution q is stationary. The state distribution after one transition is computed as follows:

$$p^1 = q \cdot A = \begin{pmatrix} 0.75 \\ 0.25 \end{pmatrix}$$

Repeating this multiplication for additional time steps (e.g., $p^2, p^3, \ldots$) results in the same distribution q, confirming it as the stationary distribution:

$$p^2 = p^1 \cdot A = \begin{pmatrix} 0.75 \\ 0.25 \end{pmatrix}$$

Since the distribution remains constant after successive transitions, q is the stationary distribution, and $P(S_t = j)$ remains constant for all t.

When initialising a Markov chain, the start probability of each state and the likelihood matrix for transitioning from one state to another are given. Using this setup, a large number of simulations (10,000 steps in our case) can be performed on the Markov chain to generate a sequence of states representing the stochastic behaviour of the system over time.

Once the sequence has been generated, the next step is to count the number of occurrences of each state in the sequence to determine their relative frequencies. The relative frequency is obtained by dividing the count of each state by the total number of generated states. It can be seen that with a sufficiently large sample size, the relative frequencies are close to the theoretical probabilities defined by the transition matrix of the Markov chain.

The frequency of State 1, denoted as f_1, is given by:

$$f_1 = 0.75210$$

The frequency of State 2, denoted as f_2, is given by:

$$f_2 = 0.24790$$

Questions 3

To verify the functionality of our HMM random generation method, we start by determining the theoretical expected mean and variance for the output variable X, which is contingent on the state Z. Using the provided state probabilities $P(S1) = 0.75$ and $P(S2) = 0.25$, and the state-dependent means $M1 = 0$ and $M2 = 3$, the expected value $E[X]$ is calculated as follows:

$$E[X] = P(S1) \times M1 + P(S2) \times M2 = 0.75 \times 0 + 0.25 \times 3 = 0.75.$$

For the variance $\text{var}[X]$, we incorporate both the variance within each state and the variance due to the expectation of the output across states. State-dependent variances $\text{std1}^2 = 1$ for $S1$ and $\text{std2}^2 = 2$ for $S2$, the variance is calculated using the formula:

$$\text{var}[X] = P(S1) \times \text{std1}^2 + P(S2) \times \text{std2}^2 + P(S1) \times (M1 - E[X])^2 + P(S2) \times (M2 - E[X])^2$$

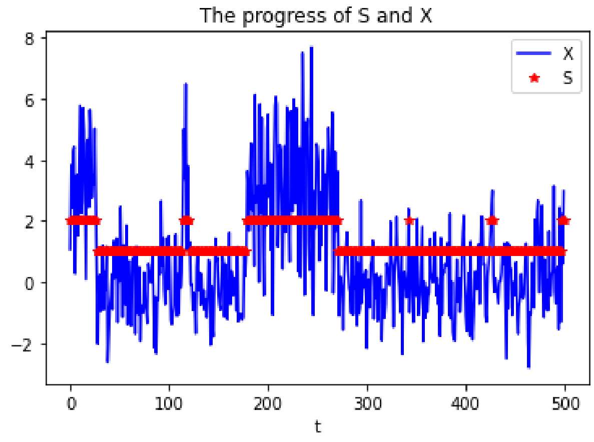
Simplifying, we find:

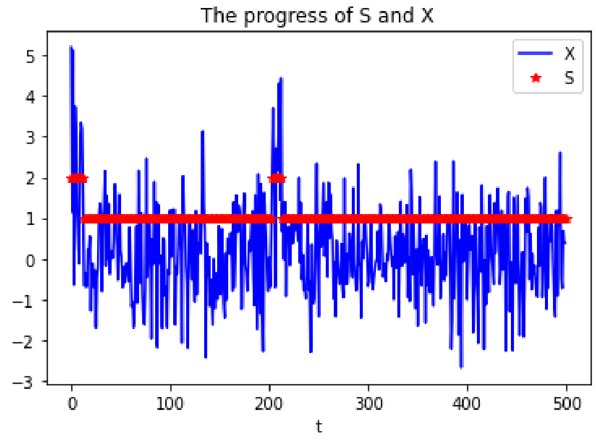
$$\text{var}[X] = 0.75 \times 1 + 0.25 \times 2 + 0.75 \times (0 - 0.75)^2 + 0.25 \times (3 - 0.75)^2 = 3.4375.$$

After completing these theoretical calculations, we simulate the HMM to generate a sequence of 10,000 output values and then calculate the empirical mean and variance. The estimated empirical mean and variance are 0.748 and 3.454, respectively, and these empirical values are very close to our calculated theoretical mean of 0.75 and variance of 3.4375, thus confirming the correctness of our HMM implementation.

Questions 4

In this exercise, we will investigate the behaviour of a HMM using the provided Python code. The model is defined by two states controlled by a Markov chain transition matrix, where state transitions are infrequent (the transition probability of leaving each state is 0.01 and 0.03, respectively). The observable output of each state is modelled by a Gaussian distribution with different means and standard deviations (state 1: mean = 0, standard deviation = 1; state 2: mean = 3, standard deviation = 2). A plot generated for 500 observations for each of the two samples shows the underlying state sequence (‘S‘) and the corresponding observations (‘X‘). As can be seen from the plots, the states (‘S‘) tend to last longer due to lower transition probabilities, which is reflected in the fact that the observations (‘X‘) are clustered over longer periods of time around the mean value determined by the Gaussian distribution of the active states. This behavior exemplifies the memory characteristic and state-dependent output variability inherent in HMMs.



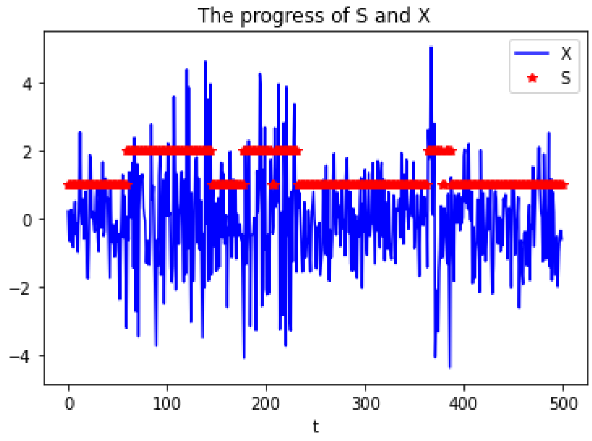


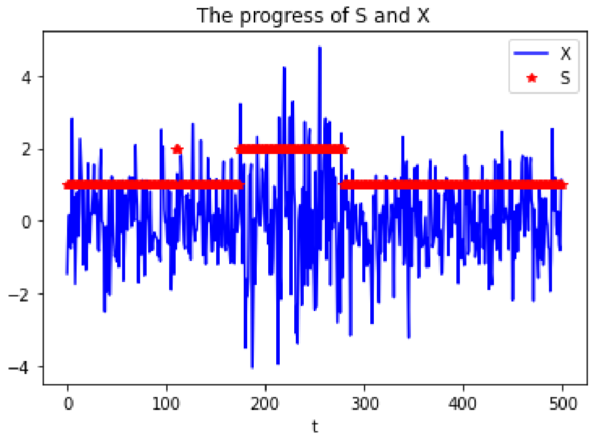
Questions 5

In the revised HMM, where both states share the same mean $\mu_1 = \mu_2 = 0$ but differ in their standard deviations $\sigma_1 = 1$ for state 1 and $\sigma_2 = 2$ for state 2, the output of the model behaves differently from the initial configuration where means differed. Despite the shared mean, the variability in observed outputs (X) depends on the state due to differing standard deviations. This creates a scenario where although the output mean remains constant, the spread or dispersion of values around that mean varies significantly between the two states.

The plots generated show the observed variables X and the state sequence S. The observations from the two states overlap around the same mean (0), but their dispersion varies, with state 2 showing a wider spread due to a higher standard deviation. This overlapping makes it more challenging to distinguish between states based solely on observed values, unlike in the original model where different means provided clearer differentiation.

Estimating the state sequence S from the observed variables X becomes more difficult in this setup. Because the means are identical, the distinction between states relies heavily on observing the variance in the data. Without clear mean differences, simple observation of output values becomes less informative, necessitating more sophisticated statistical methods or additional data to accurately infer the underlying state sequence. This highlights a critical aspect of HMMs where state differentiation is heavily dependent on the distinctness of the parameters defining the emissions from each state.





Questions 6

The output from the finite-duration HMM test indicates effective functionality of the `rand` function for handling sequence termination. In this case, the model has 1% chance to enter the exit state, except for the very beginning state. The HMM was set up with a maximum possible sequence length of 300, yet both the hidden state sequence S and the observed output sequence X concluded at 98 steps. This premature termination confirms the model's correct implementation of transition probabilities leading to an absorbing state, demonstrating the model's capability to manage finite-duration scenarios as expected.

```
mc = MarkovChain( np.array( [ 0.75, 0.25 ] ),
                  np.array( [ [ 0.5, 0.49, 0.01  ], [ 0.3, 0.69, 0.01 ] ] ) )
g1 = GaussD( means=[0], stdevs=[1] )
g2 = GaussD( means=[3], stdevs=[2] )
h  = HMM( mc, [g1, g2])
```

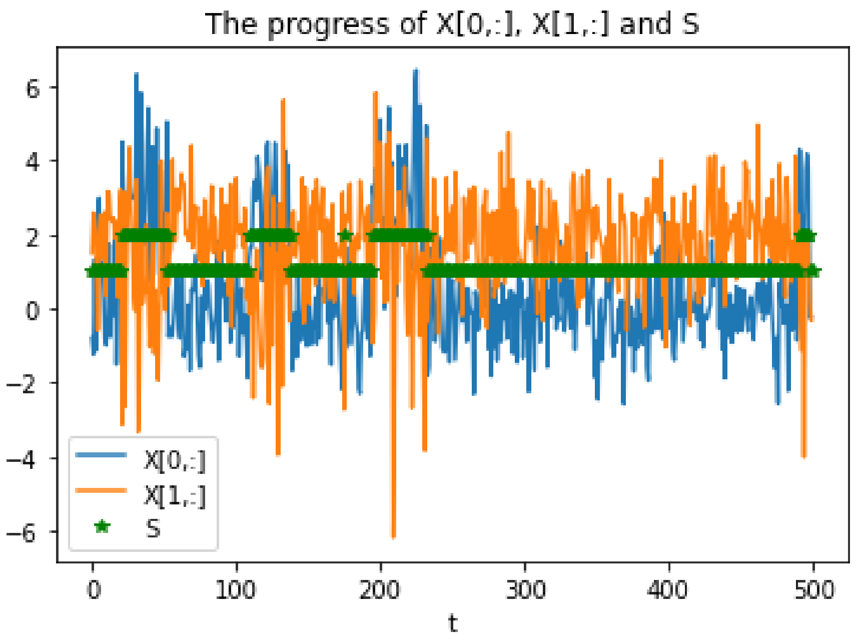
```
    maxLength = 300
    x,s = h.rand(maxLength)
    This is supposed to be a finite duration HMM,

Max number of states = 300
The length of S (hidden state) generated = 98
The length of X (output) generated = 98
```

The lengths of `s` and `x` being less than `maxLength` is reasonable because the HMM will terminate once it reaches a terminal state or when the maximum length is reached.

Questions 7

To demonstrate the capability of the `rand` function to handle vector outputs within a HMM, a new test was conducted with Gaussian vector distributions. The HMM uses two states, each defined by a Gaussian Distribution (GaussD) with distinct means and covariance matrices. Specifically, the first state has a mean of $[0, 2]$ with an identity covariance matrix, while the second state has a mean of $[3, 1]$ with a non-diagonal covariance matrix $\begin{bmatrix} 2 & 1 \\ 1 & 4 \end{bmatrix}$. This setup aims to verify that the `rand` function correctly generates vector outputs according to the defined state-dependent distributions. The generated plot visualizes two dimensions of output vectors $X[0,:]$ and $X[1,:]$ along with the state sequence S, demonstrating the function's ability to manage and differentiate outputs from complex Gaussian distributions within the stochastic framework of the HMM. The plot successfully shows how the observed outputs vary over time, closely adhering to the statistical characteristics defined by the respective state's distribution parameters.

```
mc = MarkovChain( np.array( [ 0.75, 0.25 ] ),
                  np.array( [ [ 0.99, 0.01 ], [ 0.03, 0.97 ] ] ) )
g1 = GaussD( means=[0, 2], cov=np.array([[1, 0],[0, 1]]) )
g2 = GaussD( means=[3, 1], cov=np.array([[2, 1],[1, 4]]) )
h  = HMM( mc, [g1, g2])
x, s = h.rand(500)
plt.plot(x[0])
plt.plot(x[1])
plt.plot(s, ’g*’)
plt.title(’The progress of X[0,:], X[1,:] and S’)
plt.legend([’X[0,:]’, ’X[1,:]’, ’S’])
plt.xlabel(’t’)
plt.show()
```


The progress of X[0,:], X[1,:] and S
6
4
2
0
−2
−4
−6
0
100
200
300
400
500
t
X[0,:]
X[1,:]
S