Assignment 2
Machine Learning in Robotics

Atussa Koushan - 03713848

August 2019

1 Learning data set using Gaussian mixture model

In this exercise the parameters of a GMM of size 4 are learned using the Expectation-Maximization algorithm. The learned priors π_k, means μ_k and convariances Σ_k are presented in equations 1, 2 and 3 respectively.

$$\pi_1 = 0.2400, \pi_2 = 0.2011, \pi_3 = 0.2972, \pi_4 = 0.2617 \tag{1}$$

$$\mu_1 = \begin{bmatrix} -0.0432 \\ 0.0446 \end{bmatrix}, \mu_2 = \begin{bmatrix} -0.0147 \\ -0.0796 \end{bmatrix}, \mu_3 = \begin{bmatrix} -0.0194 \\ -0.0166 \end{bmatrix}, \mu_4 = \begin{bmatrix} 0.0262 \\ 0.0617 \end{bmatrix} \tag{2}$$

$$\begin{aligned} \Sigma_1 = 10^{-3} * \begin{bmatrix} 0.1748 & 0.2615 \\ 0.2615 & 0.3975 \end{bmatrix}, \Sigma_2 = 10^{-3} * \begin{bmatrix} 0.3944 & 0.2166 \\ 0.2166 & 0.1276 \end{bmatrix}, \\ \Sigma_3 = 10^{-3} * \begin{bmatrix} 0.7437 & -0.5917 \\ -0.5917 & 0.6103 \end{bmatrix}, \Sigma_4 = \begin{bmatrix} 0.0011 & -0.0004 \\ -0.0004 & 0.0002 \end{bmatrix} \end{aligned} \tag{3}$$

The density values for inputs in range [-0.1, 0.1] are shown from different angles in figures 1-3. It is clear that the mixture model consists of four separate Gaussian distributions that together follow the pattern of the input data. As they cover roughly equally sized input spaces, it makes sense that the priors are very similar.

2 Human gesture recognition using hidden Markov model

In this exercise a set of gestures are classified based on 10 observation sequences. This is done using the forward procedure.

The gesture is determined from the log-likelihood of the observation sequence. As all the sequences got a log-likelihood less than -115, they are **all classified as gesture 2**.

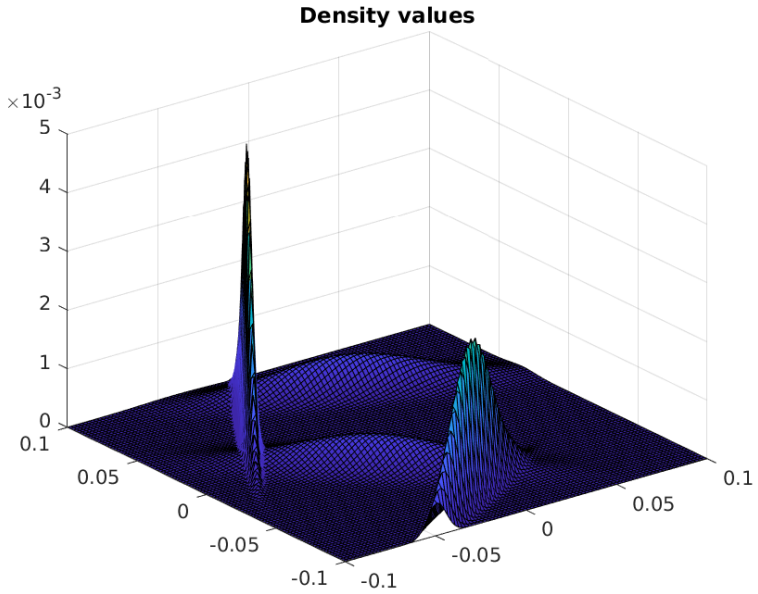

Figure 1: GMM density values

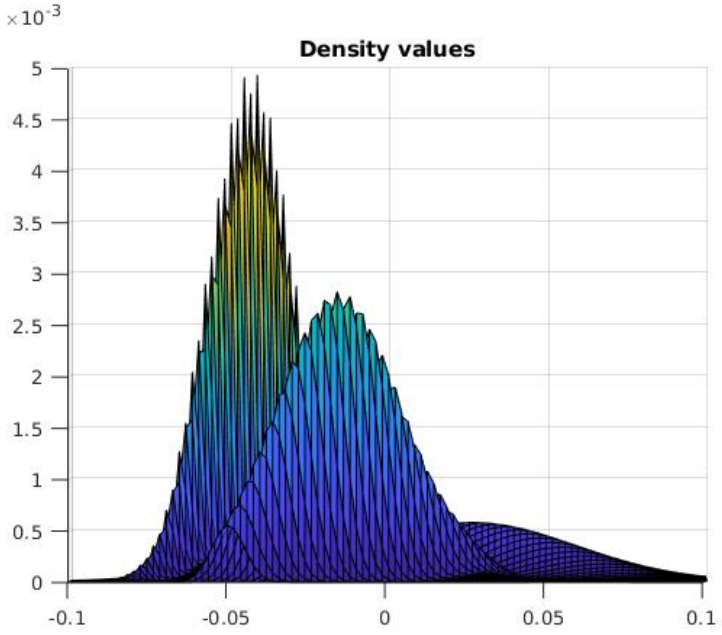

Figure 2: GMM density values side view

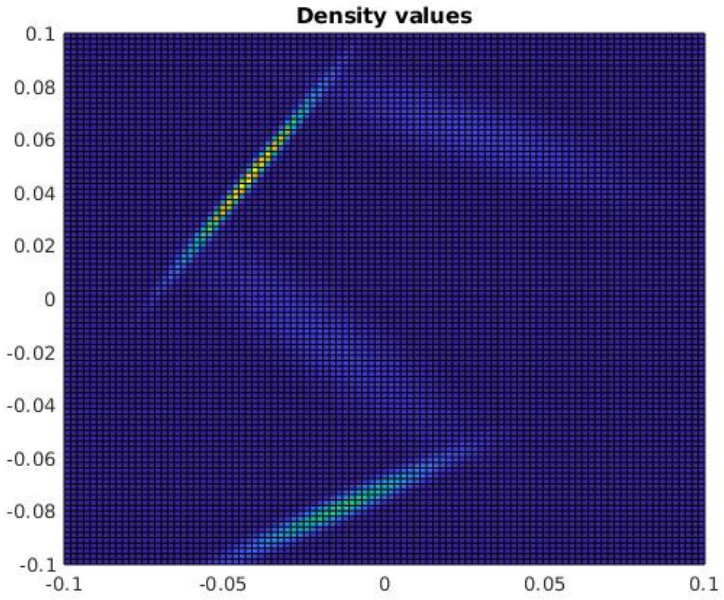

Figure 3: GMM density values top view

3 Learning a gait pattern for a humanoid robot using Reinforcement Learning

Task 1

For simplicity the reward function used only contains the values -1,0,1, where 1 is used for state-action pairs that move the robot forward, -1 for state-action pairs that should be avoided and 0 for the rest.

Task 2

Figure 4: Policy iteration from state 3

Figure 5: Policy iteration from state 10

1. Reward matrix, see (4).

$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & -1 \\ 0 & -1 & -1 & -1 \\ 0 & 0 & 0 & 0 \\ -1 & -1 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 \\ -1 & -1 & 0 & -1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ -1 & 1 & 0 & -1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 1 \\ 0 & -1 & -1 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{4}$$

2. Gamma used: $\gamma = 0.6$. Increasing gamma will make the algorithm run for more iterations and be unstable. This is because it can get a relatively high reward for new states. Decreasing gamma will however result in the algorithm not necessarily finding the optimal solution as it is not "curios" enough about exploring new states.

3. Approximately 3 iterations to converge.

4. See figure 4 and 5.

Task 3

Figure 6: Q learning starting from state 5

Figure 7: Q learning starting from state 12

1. $\epsilon = 0.4$, $\alpha = 1$ (deterministic).
2. Yes it matters. Too high values makes the algorithm run for more iterations and too low values might result in the algorithm getting stuck in a local minima.
3. Approximately 300 steps for convergence.
4. See figures 6 and 7