TÉCNICO LISBOA

Learning and Decision Making 2015-2016

MSc in Computer Science and Engineering

Second test – June 4, 2016

Instructions

- You have 90 minutes to complete the test.
- Make sure that your test has a total of 7 pages and is not missing any sheets, then write your full name and student n. on this page (and all others if you want to be safe).
- The test has a total of 6 questions, with a maximum score of 20 points. The questions have different levels of difficulty. The point value of each question is provided next to the question number.
- *If you get stuck in a question, move on.* You should start with the easier questions to secure those points, before moving on to the harder questions.
- *No interaction with the faculty is allowed during the test.* If you are unclear about a question, clearly indicate it and answer to the best of your ability.
- Please provide your answer in the space below each question. If you make a mess, clearly indicate your answer.
- The test is open book and open notes. You may use a calculator, but any other type of electronic or communication equipment is not allowed.
- Good luck.

1 Partially observable MDPs

Question 1. (4 pts.)

Consider the Tiger problem discussed in class, represented as a POMDP $(\mathcal{X}, \mathcal{A}, \mathcal{Z}, \{\boldsymbol{P}_a\}, \{\boldsymbol{O}_a\}, c, \gamma)$, where $\mathcal{X} = \{x_\ell, x_r\}$ (x_ℓ indicates the tiger on the left and x_r the tiger on the right), $\mathcal{A} = \{a_0, a_\ell, a_r\}$ (a_0 is the action listen, a_ℓ and a_r are the actions for opening the left and right door, respectively), $\mathcal{Z} = \{z_\ell, z_r\}$, and the model parameters are given by:

$$\boldsymbol{P}_{a_0} = \begin{bmatrix} 1.0 & 0.0 \\ 0.0 & 1.0 \end{bmatrix} \qquad \boldsymbol{P}_{a_\ell} = \begin{bmatrix} 0.5 & 0.5 \\ 0.5 & 0.5 \end{bmatrix} \qquad \boldsymbol{P}_{a_r} = \begin{bmatrix} 0.5 & 0.5 \\ 0.5 & 0.5 \end{bmatrix}$$

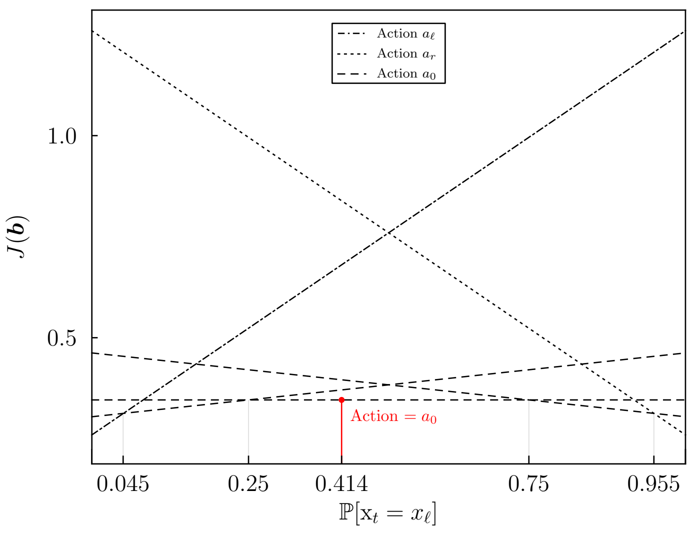
$$\boldsymbol{O}_{a_0} = \begin{bmatrix} 0.85 & 0.15 \\ 0.15 & 0.85 \end{bmatrix} \qquad \boldsymbol{O}_{a_\ell} = \begin{bmatrix} 0.5 & 0.5 \\ 0.5 & 0.5 \end{bmatrix} \qquad \boldsymbol{O}_{a_r} = \begin{bmatrix} 0.5 & 0.5 \\ 0.5 & 0.5 \end{bmatrix}$$

$$c = \begin{bmatrix} 0.1 & 1 & 0 \\ 0.1 & 0 & 1 \end{bmatrix},$$

where states, actions and observations are ordered as in the sets indicated above.

a) **(3 pts.)** Suppose that the belief of the agent at time-step t is given by $\boldsymbol{b}_t = [0.8, 0.2]^\top$, where the first component corresponds to the probability that $\mathrm{x}_t = x_\ell$ (the tiger is on the left) and the second component to the probability that $\mathrm{x}_t = x_r$ (the tiger is on the right). If the agent selects the action $a_t = a_0$ (listen) and observes $z_{t+1} = z_r$ (tiger on the right), determine the belief at time-step $t + 1$, $\boldsymbol{b}_{t+1}$. Indicate all relevant computations.

b) **(1 pt.)** Consider now the value function for the POMDP represented in the plot below, where α-vectors corresponding to different actions are plotted using different lines.



According to the cost-to-go function in the plot, indicate the action to take at time-step $t+1$, given the belief $\boldsymbol{b}_{t+1}$ that you computed in part (a). Explain your reasoning (you can mark the plot to explain your reasoning).

Note: If you did not complete part (a), use the belief $\boldsymbol{b}_{t+1} = [0.03, 0.97]^\top$.

Solution 1.

a) The belief update rule is given (componentwise) by

$$\boldsymbol{B}(z, a, \boldsymbol{b})_y = \frac{\sum_{x \in \mathcal{X}} b(x)\boldsymbol{P}(y \mid x, a)\boldsymbol{O}(z \mid y, a)}{\sum_{x, x' \in \mathcal{X}} b(x)\boldsymbol{P}(x' \mid x, a)\boldsymbol{O}(z \mid x', a)}$$

or, in vector notation, by

$$\boldsymbol{B}(z, a, \boldsymbol{b}) = \xi \boldsymbol{b}^\top \boldsymbol{P}_a \operatorname{diag}(\boldsymbol{O}_a(z)),$$

where ξ is a normalization constant given by

$$\xi = \boldsymbol{b}^\top \boldsymbol{P}_a \operatorname{diag}(\boldsymbol{O}_a(z))\boldsymbol{1}.$$

In our case, we have:

$$\boldsymbol{b}^\top \boldsymbol{P}_a \operatorname{diag}(\boldsymbol{O}_a(z)) = \begin{bmatrix} 0.8 & 0.2 \end{bmatrix} \begin{bmatrix} 1.0 & 0.0 \\ 0.0 & 1.0 \end{bmatrix} \begin{bmatrix} 0.15 & 0.0 \\ 0.0 & 0.85 \end{bmatrix} = \begin{bmatrix} 0.12 \\ 0.17 \end{bmatrix}$$

The final result is obtained by normalizing the resulting vector, yielding:

$$\boldsymbol{b}_{t+1} = \frac{1}{0.29} \begin{bmatrix} 0.12 \\ 0.17 \end{bmatrix} = \begin{bmatrix} 0.414 \\ 0.586 \end{bmatrix}.$$

b) The belief $\boldsymbol{b}_{t+1}$ computed in the previous question is marked in the plot above, as well as the corresponding α-vector, which corresponds to action a_0.

2 Sequential prediction

Question 2. (**2 pts.**)
Consider an agent that must, at each time-step t, select one of three possible actions: a_1, a_2 and a_3. After selecting the action, the agent pays a cost c_t that depends on the action selected at time-step t. Suppose that, for $t = 1, \ldots, 10$, the agent selected the actions:

$$a_1 \quad a_2 \quad a_3 \quad a_2 \quad a_1 \quad a_3 \quad a_2 \quad a_2 \quad a_1 \quad a_3$$

and paid the costs

$$0.56 \quad 0.30 \quad 0.62 \quad 0.19 \quad 0.78 \quad 0.74 \quad 0.45 \quad 0.37 \quad 0.68 \quad 0.62.$$

Indicate the action selected by the agent at time-step $t = 11$ if the agent follows the UCB algorithm.

Solution 2.
To compute the action selection for UCB, we first determine the average reward associated with each action. This yields

$$\hat{Q}(a_1) = 0.67 \qquad \hat{Q}(a_2) = 0.33 \qquad \hat{Q}(a_3) = 0.66.$$

The coefficients of each action, according to UCB, are then given by

$$\hat{Q}(a_1) - \sqrt{\frac{2\log(10)}{3}} = 0.67 - 1.24 = -0.57$$
$$\hat{Q}(a_2) - \sqrt{\frac{2\log(10)}{4}} = 0.33 - 1.07 = -0.74$$
$$\hat{Q}(a_4) - \sqrt{\frac{2\log(10)}{3}} = 0.66 - 1.24 = -0.58$$

and UCB will select action $A = a_2$.

Question 3. (2 pts.)
Explain the main differences between UCB and EXP3, indicating in which situations each of the two algorithms is more adequate.

Solution 3.
UCB is an algorithm for *stochastic bandits*, while EXP3 is an algorithm for *adversarial bandits*. This means that

- UCB assumes that the rewards associated with each action *follow some underlying distribution*, and the goal of the agent is to track the best arm as soon as possible. For this reason, UCB relies on an upper confidence bound for the mean reward associated with each arm, and adopts a *deterministic policy* that simply selects the action with the highest value for this bound. UCB should thus be used in situations where the rewards are "well-behaved".
- EXP3 does not make any assumptions regarding the process by which rewards are generated. As such, it uses a *randomized policy* that selects each action with a probability that depends on its average reward. The use of a randomized policy ensures that an adversarial generation of rewards is unable to exploit the agent. EXP3 should thus be used in situations where there is significant variability in the process by which the rewards are generated, or when assuming an underlying distribution is unreasonable.

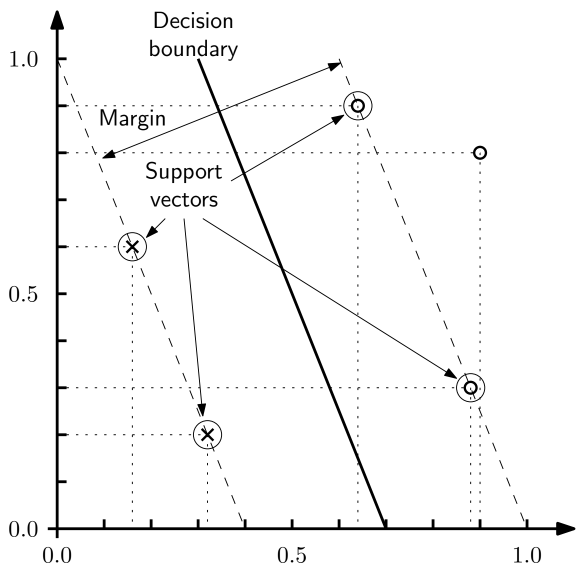
3 Supervised learning

Question 4. (6 pts.)
A hospital ran a survey among its patients, regarding their cholesterol level and exercise habits. Each patient was described by two numbers between zero and one: the first number represents the danger in the cholesterol level of the patient; the second represents the regularity with which they practice exercise. The hospital also recorded whether or not the patient had suffered from coronary disease.

The data is summarized in the table below:

Cholesterol level	0.32	0.88	0.16	0.90	0.64
Regularity of exercise	0.20	0.30	0.60	0.80	0.90
Risk of coronary disease	No	Yes	No	Yes	Yes

a) **(3 pt.)** Represent your data in a *scatter plot*, where each patient corresponds to a data point defined by the two numeric attributes in the table above. Data points corresponding to patients with risk of coronary disease should be represented with the symbol "○", and the remaining data points with the symbol "×".

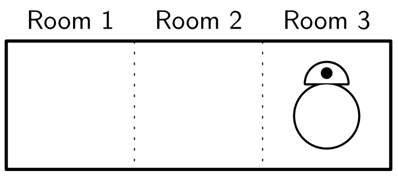


b) **(3 pts.)** Suppose you train an SVM classifier with the data above. Mark in your plot:

- How the decision boundary would be;
- The support vectors;
- The margin.

4 Reinforcement learning

Question 5. (**4 pts.**)
Consider an agent moving in an office environment with 3 areas, as depicted in the figure below.



The agent has available, at each time-step, two actions: Move Left (L) and Move Right (R). Each action moves the agent to the adjacent room in the corresponding direction (if there is one) with a certain probability, but sometimes fails, leaving the position of the agent unchanged. This problem can be modeled as an MDP with 3 states (corresponding to the 3 positions of the robot) and the two actions referred above.

Suppose that you are using the model-based approach used in Lab 9 to compute the optimal Q-function for this MDP. After 20 time steps, you have collected the following data:

$$\hat{\boldsymbol{P}}_L = \begin{bmatrix} 1 & 0 & 0 \\ 0.75 & 0.25 & 0 \\ 0 & 1 & 0 \end{bmatrix} \qquad \hat{\boldsymbol{P}}_R = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0.33 & 0.67 \\ 0 & 0 & 1 \end{bmatrix} \qquad \hat{c} = \begin{bmatrix} 1 & 1 \\ 1 & 1 \\ 0 & 0 \end{bmatrix}$$

and

$$N = \begin{bmatrix} 4 & 2 \\ 4 & 3 \\ 3 & 4 \end{bmatrix} \qquad \hat{Q} = \begin{bmatrix} 2.71 & 1.9 \\ 1.94 & 1.21 \\ 0.87 & 0 \end{bmatrix}.$$

Suppose that the agent is in Room 2, executes action R (which succeeds and moves agent to Room 3), and pays a cost of 1. Assuming that $\gamma = 0.9$, compute a step of model-based reinforcement learning, updating the estimates $\hat{\boldsymbol{P}}$, $\hat{c}$ and $\hat{Q}$.

Solution 5.

We observed a transition $(2, R, 1, 3)$. We start by updating $\hat{\boldsymbol{P}}$. Only $\hat{\boldsymbol{P}}_R(\cdot \mid 2)$ will be updated using

$$\hat{\boldsymbol{P}}(y \mid x_t, a_t) \leftarrow \left(1 - \frac{1}{N(x_t, a_t) + 1}\right) \hat{\boldsymbol{P}}(y \mid x_t, a_t) + \frac{1}{N(x_t, a_t) + 1}\mathbb{I}(x_{t+1} = y),$$

yielding

$$\hat{\boldsymbol{P}}(1 \mid 2, R) \leftarrow \left(1 - \frac{1}{4}\right) \hat{\boldsymbol{P}}(1 \mid 2, R) = 0$$
$$\hat{\boldsymbol{P}}(2 \mid 2, R) \leftarrow \left(1 - \frac{1}{4}\right) \hat{\boldsymbol{P}}(2 \mid 2, R) = \frac{1}{4}$$
$$\hat{\boldsymbol{P}}(3 \mid 2, R) \leftarrow \left(1 - \frac{1}{4}\right) \hat{\boldsymbol{P}}(3 \mid 2, R) + \frac{1}{4} = \frac{3}{4}.$$

As for $\hat{c}$, we have

$$\hat{c}(x_t, a_t) \leftarrow \hat{c}(x_t, a_t) + \frac{1}{N_t(x_t, a_t) + 1}\big(r_t - \hat{r}(x_t, a_t)\big),$$

yielding

$$\hat{c}(2, R) \leftarrow \hat{c}(2, R) + \frac{1}{4}\big(1 - 1\big) = 1.$$

Finally, for Q we get

$$\hat{Q}(x_t, a_t) \leftarrow \hat{c}(x_t, a_t) + \gamma \sum_y \hat{\boldsymbol{P}}(y \mid x_t, a_t) \min_{a'} Q^{(t)}(y, a'),$$

yielding

$$\hat{Q}(2, R) \leftarrow 1 + 0.9 \times \begin{bmatrix} 0 & \frac{1}{4} & \frac{3}{4} \end{bmatrix} \times \begin{bmatrix} 1.9 \\ 1.21 \\ 0 \end{bmatrix} = 1.27.$$

Question 6. (2 pts.)

In class, it was explained that Q-learning is an *off-policy method*, while SARSA is an *on-policy method*. Explain the difference between on-policy and off-policy methods, indicating in particular the differences that they induce in terms of the update rules for the two methods.

Solution 6.

Off-policy refers to reinforcement learning methods that learn the value of a policy while following a different policy. *On-policy* refers to reinforcement learning methods that learn the value of the policy that the agent is following.

In Q-learning, the update rule is

$$Q^{(k+1)}(x_t, a_t) = Q^{(k)}(x_t, a_t) + \alpha_t(c_t + \gamma \boxed{\min_{a \in \mathcal{A}} Q^{(k)}(x_{t+1}, a)} - Q^{(k)}(x_t, a_t))$$

and we can observe that the update is based on the value of the *greedy policy at the next state*, independently of the policy that the agent is actually following. Conversely, on SARSA, the update rule is

$$Q^{(k+1)}(x_t, a_t) = Q^{(k)}(x_t, a_t) + \alpha_t(c_t + \gamma \boxed{Q^{(k)}(x_{t+1}, a_{t+1})} - Q^{(k)}(x_t, a_t))$$

and we can observe that the update is based on the value of the *actual policy followed by the agent at the next state*. For this reason, Q-learning is an off-policy method while SARSA is an on-policy method.