FUNDAMENTALS OF REINFORCEMENT LEARNING – ALBERTA UNIVERSITY - CANADA

LINK: https://www.coursera.org/learn/fundamentals-of-reinforcement-learning?shared=facebook#_=_

coursera Explore What do you want to learn? For Enterprise For Students Holger Rivera

Browse > Data Science > Machine Learning

Fundamentals of Reinforcement Learning

★★★★★ 4.8 1,332 ratings Share

Martha White +1 more instructor

Part of the **4-course series** **Reinforcement Learning Specialization**

Go To Course Already enrolled

29,712 already enrolled

Offered By
University of Alberta
Alberta Machine Intelligence Institute

About Instructors Syllabus Reviews Enrollment Options FAQ

About this Course

247,278 recent views

Reinforcement Learning is a subfield of Machine Learning, but is also a general purpose formalism for automated decision-making and AI. This course introduces you to statistical learning techniques where an agent explicitly takes actions and interacts with the world. Understanding the importance and challenges of learning agents that make decisions is of vital importance today, with more and more companies interested in interactive agents and intelligent decision-making.

This course introduces you to the fundamentals of Reinforcement Learning. When you finish this course, you will:
- Formalize problems as Markov Decision Processes
- Understand basic exploration methods and the exploration/exploitation tradeoff
- Understand value functions, as a general-purpose tool for optimal decision-making
- Know how to implement dynamic programming as an efficient solution approach to an industrial control problem

This course teaches you the key concepts of Reinforcement Learning, underlying classic and modern algorithms in RL. After completing this course, you will be able to start using RL for real problems, where you have or can specify the MDP.

This is the first course of the Reinforcement Learning Specialization.

Shareable Certificate
Earn a Certificate upon completion

100% online
Start instantly and learn at your own schedule.

Course 1 of 4 in the
Reinforcement Learning Specialization

Flexible deadlines
Reset deadlines in accordance to your schedule.

Intermediate Level
Probabilities & Expectations, basic linear algebra, basic calculus, Python 3.0 (at least 1 year), implementing algorithms from pseudocode.

from pseudocode.

WHAT YOU WILL LEARN

- Formalize problems as Markov Decision Processes
- Understand basic exploration methods and the exploration / exploitation tradeoff
- Understand value functions, as a general-purpose tool for optimal decision-making
- Know how to implement dynamic programming as an efficient solution approach to an industrial control problem

Approx. 15 hours to complete

English
Subtitles: English

SYLLABUS

WEEK

1

1 hour to complete

Welcome to the Course!

Welcome to: Fundamentals of Reinforcement Learning, the first course in a four-part specialization on Reinforcement Learning brought to you by the University of Alberta, Onlea, and Coursera. In this pre-course module, you'll be introduced to your instructors, get a flavour of what the course has in store for you, and be given an in-depth roadmap to help make your journey through this specialization as smooth as possible.

4 videos (Total 20 min), 2 readings SEE ALL

---

4 hours to complete

The K-Armed Bandit Problem

For the first week of this course, you will learn how to understand the exploration-exploitation trade-off in sequential decision-making, implement incremental algorithms for estimating action-values, and compare the strengths and weaknesses to different algorithms for exploration. For this week's graded assessment, you will implement and test an epsilon-greedy agent.

8 videos (Total 46 min), 3 readings, 2 quizzes SEE ALL

WEEK

2

3 hours to complete

Markov Decision Processes

When you're presented with a problem in industry, the first and most important step is to translate that problem into a Markov Decision Process (MDP). The quality of your solution depends heavily on how well you do this translation. This week, you will learn the definition of MDPs, you will understand goal-directed behavior and how this can be obtained from maximizing scalar rewards, and you will also understand the difference between episodic and continuing tasks. For this week's graded assessment, you will create three example tasks of your own that fit into the MDP framework.

7 videos (Total 36 min), 2 readings, 2 quizzes SEE ALL

WEEK

3

3 hours to complete

Value Functions & Bellman Equations

Once the problem is formulated as an MDP, finding the optimal policy is more efficient when using value functions. This week, you will learn the definition of policies and value functions, as well as Bellman equations, which is the key technology that all of our algorithms will use.

9 videos (Total 56 min), 3 readings, 2 quizzes SEE ALL

WEEK

4

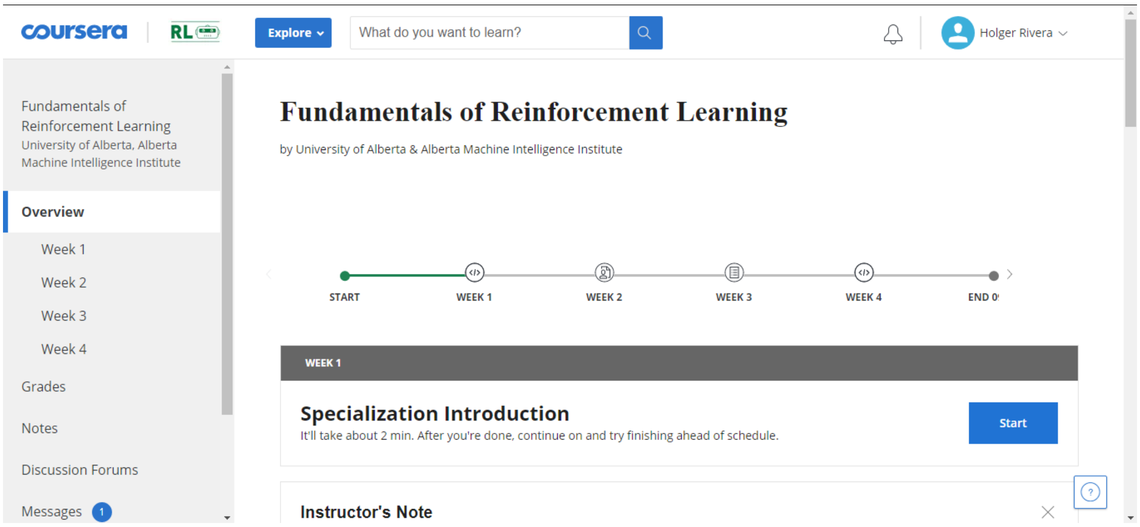
4 hours to complete

Dynamic Programming

This week, you will learn how to compute value functions and optimal policies, assuming you have the MDP model. You will implement dynamic programming to compute value functions and optimal policies and understand the utility of dynamic programming for industrial applications and problems. Further, you will learn about Generalized Policy Iteration as a common template for constructing algorithms that maximize reward. For this week's graded assessment, you will implement an efficient dynamic programming agent in a simulated industrial control problem.

10 videos (Total 72 min), 3 readings, 2 quizzes SEE ALL

INTRO


coursera
RL
Explore
What do you want to learn?
Holger Rivera
Fundamentals of Reinforcement Learning
University of Alberta, Alberta Machine Intelligence Institute
Overview
Week 1
Week 2
Week 3
Week 4
Grades
Notes
Discussion Forums
Messages 1
Fundamentals of Reinforcement Learning
by University of Alberta & Alberta Machine Intelligence Institute
START
WEEK 1
WEEK 2
WEEK 3
WEEK 4
END O
WEEK 1
Specialization Introduction
It'll take about 2 min. After you're done, continue on and try finishing ahead of schedule.
Start
Instructor's Note

Module 2 Learning Objectives

By the end of this module, you should be able to meet the following learning objectives:

Lesson 1: The K-Armed Bandit Problem

- Understand the temporal nature of the bandit problem
- Define k-armed bandit
- Define action-values
- Define reward

Lesson 2: What to Learn? Estimating Action Values

- Define action-value estimation methods
- Define exploration and exploitation
- Select actions greedily using an action-value function
- Define online learning
- Understand a simple online sample-average action-value estimation method
- Define the general online update equation
- Understand why we might use a constant stepsize in the case of non-stationarity

Lesson 3: Exploration vs. Exploitation Tradeoff

- Compare the short-term benefits of exploitation and the long-term benefits of exploration
- Understand optimistic initial values
- Describe the benefits of optimistic initial values for early exploration
- Explain the criticisms of optimistic initial values
- Describe the upper confidence bound action selection method
- Define optimism in the face of uncertainty

Compare bandits to supervised learning

How is the bandit problem similar or different to the supervised learning problem?

The k-armed bandits problem is a Reinforcement Learning problem different to Supervised Learning problem. Here check the differences:

- Reinforcement learning essentially is an **evaluative feedback**, otherwise Supervised Learning is an **instructive feedback**
- K-armed bandits problem do not have predefined responses (traditional labels), otherwise Supervised Learning have inputs and labels
- In Reinforcement learning the agent obtains the learning from the environment, otherwise Supervised Learning recognizes patterns between inputs and outputs, based in data
- Reinforcement learning depends entirely on the actions taken, otherwise Supervised Learning is independent of the action taken

Exploration/Exploitation

TOTAL POINTS 8

1. What is the incremental rule (sample average) for action values? 1 point

- ◯ $Q_{n+1} = Q_n + \frac{1}{n}[Q_n]$
- ◯ $Q_{n+1} = Q_n - \frac{1}{n}[R_n - Q_n]$
- ◯ $Q_{n+1} = Q_n + \frac{1}{n}[R_n + Q_n]$
- ◉ $Q_{n+1} = Q_n + \frac{1}{n}[R_n - Q_n]$

2. Equation 2.5 (from the SB textbook, 2nd edition) is a key update rule we will use throughout the Specialization. We discussed this equation extensively in video. This exercise will give you a better hands-on feel for how it works. The blue line is the target that we might estimate with equation 2.5. The red line is our estimate plotted over time. 1 point

$$q_{n+1} = q_n + \alpha_n [R_n - q_n]$$

Given the estimate update in red, what do you think was the value of the step size parameter we used to update the estimate on each time step?

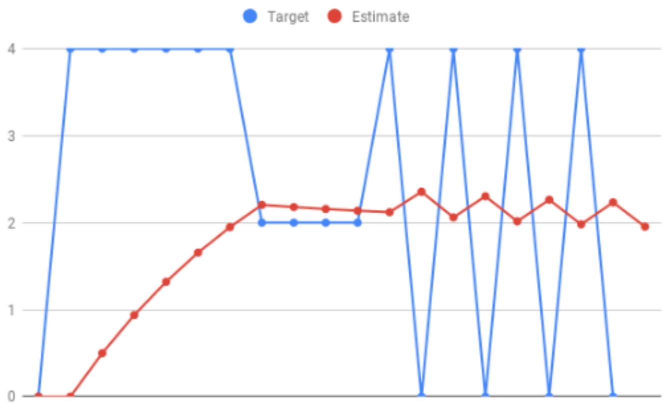
- ◯ 1.0
- ◉ 1/2
- ◯ 1/8
- ◯ 1 / (t - 1)

3. Equation 2.5 (from the SB textbook, 2nd edition) is a key update rule we will use throughout the Specialization. We discussed this equation extensively in video. This exercise will give you a better hands-on feel for how it works. The blue line is the target that we might estimate with equation 2.5. The red line is our estimate plotted over time.

1 point

$q_{n+1} = q_n + \alpha_n[R_n - q_n]$

Given the estimate update in red, what do you think was the value of the step size parameter we used to update the estimate on each time step?



- (x) 1/8
- ( ) 1.0
- ( ) 1/2
- ( ) 1 / (t - 1)

---

4. Equation 2.5 (from the SB textbook, 2nd edition) is a key update rule we will use throughout the Specialization. We discussed this equation extensively in video. This exercise will give you a better hands-on feel for how it works. The blue line is the target that we might estimate with equation 2.5. The red line is our estimate plotted over time.

1 point

$q_{n+1} = q_n + \alpha_n[R_n - q_n]$

Given the estimate update in red, what do you think was the value of the step size parameter we used to update the estimate on each time step?

- ○ 1/8
- ○ 1/2
- ◉ 1.0
- ○ 1 / (t - 1)

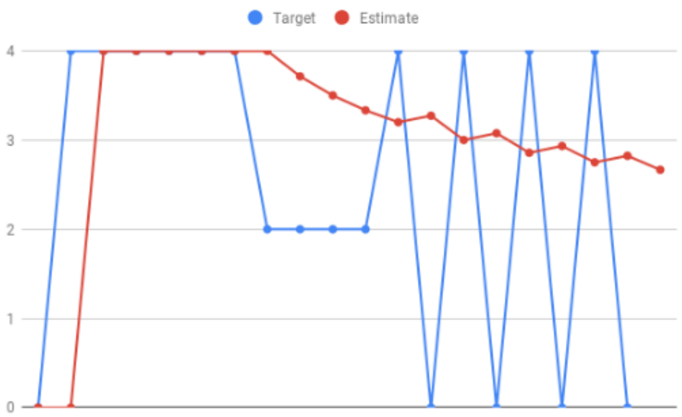
---

5. Equation 2.5 (from the SB textbook, 2nd edition) is a key update rule we will use throughout the Specialization. We discussed this equation extensively in [video](). This exercise will give you a better hands-on feel for how it works. The blue line is the target that we might estimate with equation 2.5. The red line is our estimate plotted over time.

1 point

$$q_{n+1} = q_n + \alpha_n [R_n - q_n]$$

Given the estimate update in red, what do you think was the value of the step size parameter we used to update the estimate on each time step?



- ○ 1.0
- ○ 1/2
- ○ 1/8
- ◉ 1 / (t - 1)

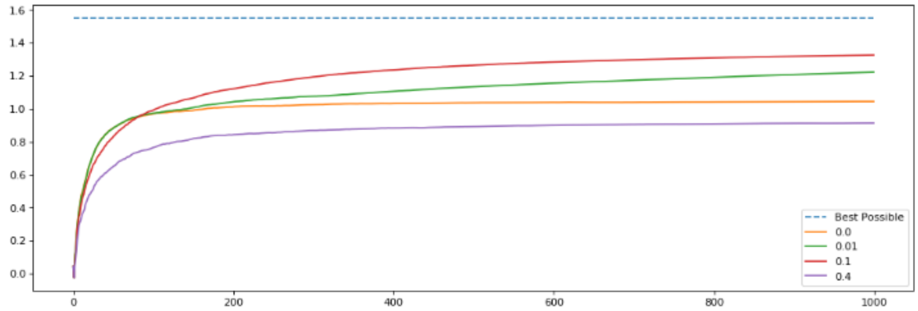
6. What is the exploration/exploitation tradeoff?

1 point

- ○ The agent wants to maximize the amount of reward it receives over its lifetime. To do so it needs to avoid the action it believes is worst to exploit what it knows about the environment. However to discover which arm is truly worst it needs to explore different actions which potentially will lead it to take the worst action at times.
- ○ The agent wants to explore the environment to learn as much about it as possible about the various actions. That way once it knows every arm's true value it can choose the best one for the rest of the time.
- ◉ The agent wants to explore to get more accurate estimates of its values. The agent also wants to exploit to get more reward. The agent cannot, however, choose to do both simultaneously.

7. Why did epsilon of 0.1 perform better over 1000 steps than epsilon of 0.01?

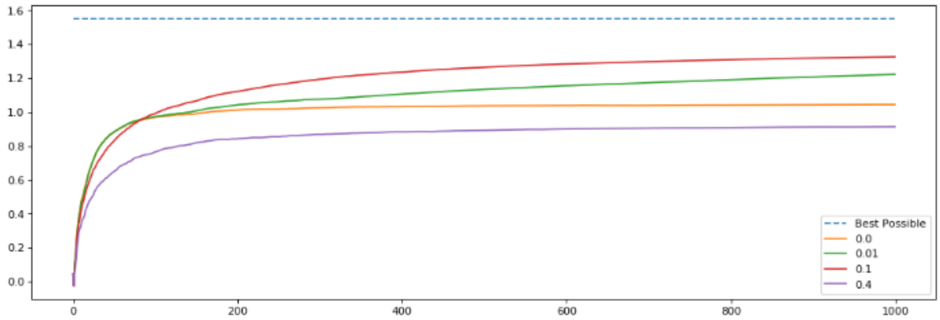
1 point



- [ ] The 0.01 agent explored too much causing the arm to choose a bad action too often.
- [ ] Epsilon of 0.1 is the optimal value for epsilon in general.
- [x] The 0.01 agent did not explore enough. Thus it ended up selecting a suboptimal arm for longer.

8. If exploration is so great why did epsilon of 0.0 (a greedy agent) perform better than epsilon of 0.4?

1 point



- [ ] Epsilon of 0.4 doesn't explore often enough to find the optimal action.
- [x] Epsilon of 0.4 explores too often that it takes many sub-optimal actions causing it to do worse over the long term.
- [ ] Epsilon of 0.0 is greedy, thus it will always choose the optimal arm.

REVISIÓN – QUIZ 01 - FEEDBACK

Exploration/Exploitation

TOTAL POINTS 8

1. What is the incremental rule (sample average) for action values?

1 / 1 point

- ( ) $Q_{n+1} = Q_n + \frac{1}{n}[Q_n]$
- ( ) $Q_{n+1} = Q_n - \frac{1}{n}[R_n - Q_n]$
- ( ) $Q_{n+1} = Q_n + \frac{1}{n}[R_n + Q_n]$
- (•) $Q_{n+1} = Q_n + \frac{1}{n}[R_n - Q_n]$

✓ **Correct**

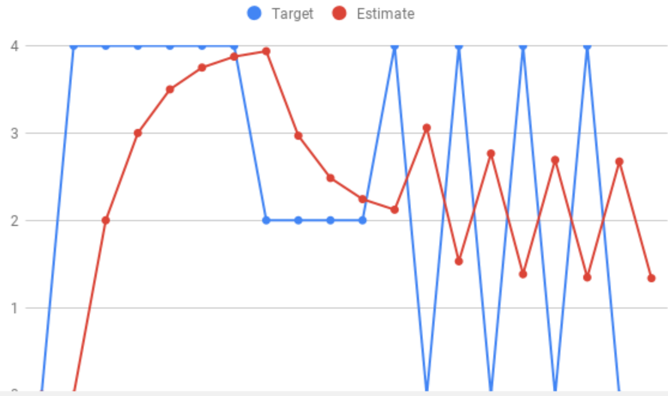
Correct! At each time step the agent moves its prediction in the direction of the error by the step size (here 1/n).

2. Equation 2.5 (from the SB textbook, 2nd edition) is a key update rule we will use throughout the Specialization. We discussed this equation extensively in video. This exercise will give you a better hands-on feel for how it works. The blue line is the target that we might estimate with equation 2.5. The red line is our estimate plotted over time.

1 / 1 point

$q_{n+1} = q_n + \alpha_n[R_n - q_n]$

Given the estimate update in red, what do you think was the value of the step size parameter we used to update the estimate on each time step?



- ( ) 1.0
- (•) 1/2
- ( ) 1/8
- ( ) 1 / (t - 1)

✓ **Correct**

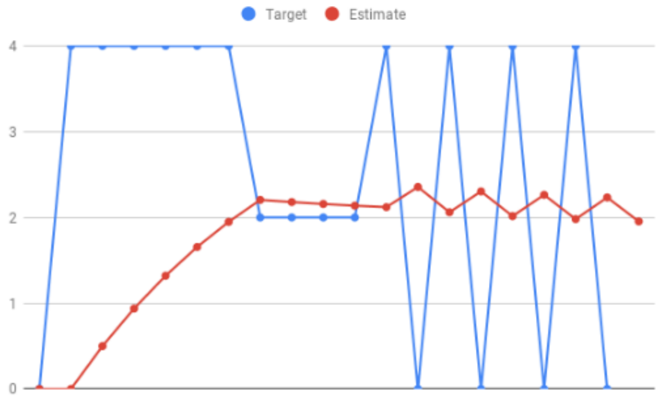
Correct! We can see that the estimate is updated by about half of what the prediction error is.

3. Equation 2.5 (from the SB textbook, 2nd edition) is a key update rule we will use throughout the Specialization. We discussed this equation extensively in [video](). This exercise will give you a better hands-on feel for how it works. The blue line is the target that we might estimate with equation 2.5. The red line is our estimate plotted over time.

1 / 1 point

$q_{n+1} = q_n + \alpha_n[R_n - q_n]$

Given the estimate update in red, what do you think was the value of the step size parameter we used to update the estimate on each time step?



- (•) 1/8
- ( ) 1.0
- ( ) 1/2
- ( ) 1 / (t - 1)

✓ **Correct**

Correct! We can see that the estimate is updated by ⅛ of the prediction error at each time step.

4. Equation 2.5 (from the SB textbook, 2nd edition) is a key update rule we will use throughout the Specialization. We discussed this equation extensively in [video](). This exercise will give you a better hands-on feel for how it works. The blue line is the target that we might estimate with equation 2.5. The red line is our estimate plotted over time.

1 / 1 point

$q_{n+1} = q_n + \alpha_n[R_n - q_n]$

Given the estimate update in red, what do you think was the value of the step size parameter we used to update the estimate on each time step?

1/8

1/2

1.0 (selected)

1 / (t - 1)

Correct

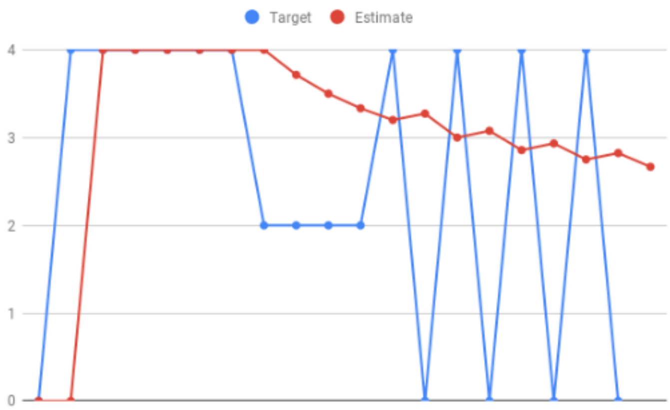
Correct! The estimate is updated to what the previous target was.

5. Equation 2.5 (from the SB textbook, 2nd edition) is a key update rule we will use throughout the Specialization. We discussed this equation extensively in video. This exercise will give you a better hands-on feel for how it works. The blue line is the target that we might estimate with equation 2.5. The red line is our estimate plotted over time.

1 / 1 point

$$q_{n+1} = q_n + \alpha_n [R_n - q_n]$$

Given the estimate update in red, what do you think was the value of the step size parameter we used to update the estimate on each time step?



1.0

1/2

1/8

1 / (t - 1) (selected)

Correct

Correct! We can see that the estimate is updated fully to the target initially, and then over time the amount that the estimate updates is reduced. This indicates that our step size is reducing over time.

6. What is the exploration/exploitation tradeoff?

1 / 1 point

○ The agent wants to maximize the amount of reward it receives over its lifetime. To do so it needs to avoid the action it believes is worst to exploit what it knows about the environment. However to discover which arm is truly worst it needs to explore different actions which potentially will lead it to take the worst action at times.

○ The agent wants to explore the environment to learn as much about it as possible about the various actions. That way once it knows every arm's true value it can choose the best one for the rest of the time.

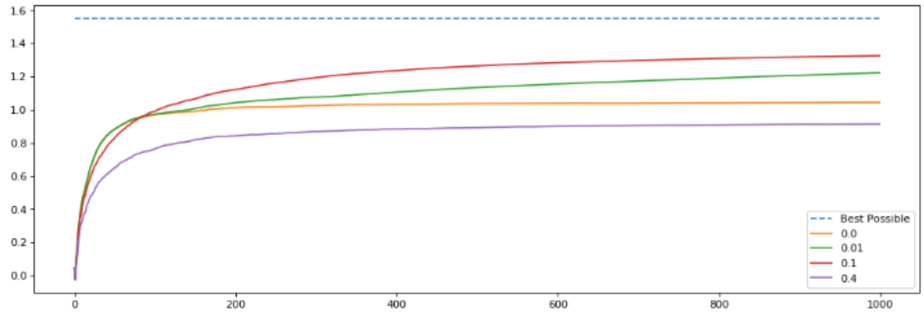
◉ The agent wants to explore to get more accurate estimates of its values. The agent also wants to exploit to get more reward. The agent cannot, however, choose to do both simultaneously.

✓ **Correct**

Correct! The agent wants to maximize the amount of reward it receives over time, but needs to explore to find the right action.

7. Why did epsilon of 0.1 perform better over 1000 steps than epsilon of 0.01?

1 / 1 point



○ The 0.01 agent explored too much causing the arm to choose a bad action too often.

○ Epsilon of 0.1 is the optimal value for epsilon in general.

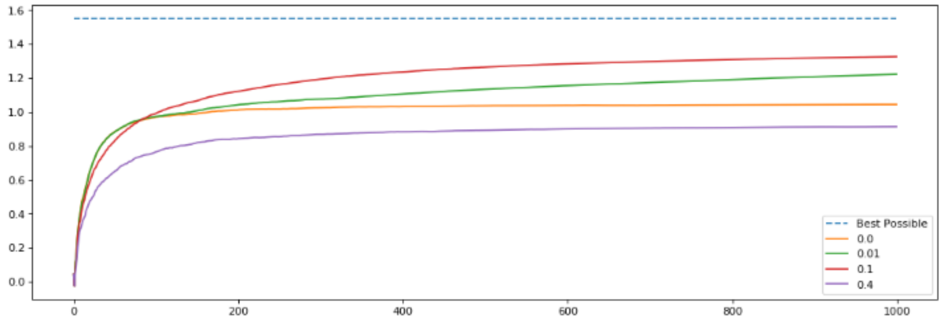
◉ The 0.01 agent did not explore enough. Thus it ended up selecting a suboptimal arm for longer.

✓ **Correct**

Correct! The agent needs to be able to explore enough to be able to find the best arm to pull over time. Here epsilon of 0.01 does not allow for enough exploration in the time allotted.

8. If exploration is so great why did epsilon of 0.0 (a greedy agent) perform better than epsilon of 0.4?

1 / 1 point



- [ ] Epsilon of 0.4 doesn't explore often enough to find the optimal action.
- [x] Epsilon of 0.4 explores too often that it takes many sub-optimal actions causing it to do worse over the long term.
- [ ] Epsilon of 0.0 is greedy, thus it will always choose the optimal arm.

Correct

Correct! While we want to explore to find the best arm, if we explore too much we can spend too much time choosing bad actions even when we know the correct one. In this case the action-value estimates are likely correct, however the policy does not always choose the action with the highest value.

Is the reward hypothesis sufficient?

The reward hypothesis states "that all of what we mean by goals and purposes can be well thought of as the maximization of the expected value of the cumulative sum of a received scalar signal (called reward)."
See http://www.incompleteideas.net/book/RLbook2018trimmed.pdf#page=75

Can you think of a situation that is not well-modeled by maximizing a scalar reward signal?

Your response has been submitted. Engage and discuss with other learners below!

Holger Rivera · 27 minutes ago
Sometimes the reward hypothesis it's NOT sufficient. A chess game is a example. If reward maximization were applied to chess, the player would seek to take the opponent's pieces or control the center of the board. This does not necessarily lead to victory, since the condition for winning the game is to eliminate the king.

QUIZ 2 – MARKOV DECISION PROCESS

MDPs

TOTAL POINTS 16

1. The learner and decision maker is the ______. 1 point

   - ○ Reward
   - ○ Environment
   - ○ State
   - ◉ Agent

2. At each time step the agent takes an ______. 1 point

   - ○ Reward
   - ○ Environment
   - ◉ Action
   - ○ State

3. What equation(s) define $q_\pi(S_t, A_t)$ in terms of subsequent rewards? 1 point

   - ☑ $q_\pi(s,a) = \mathbb{E}_\pi[G_t | S_t = s, A_t = a]$

     where: $G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \gamma^3 R_{t+4} \ldots$

   - ☐ $q_\pi(s,a) = [G_t | S_t = s, A_t = a]$

     where: $G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \gamma^3 R_{t+4} \ldots$

   - ☐ $q_\pi(s,a) = \mathbb{E}_\pi[G_t]$

     where: $G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \gamma^3 R_{t+4} \ldots$

   - ☑ $q_\pi(s,a) = \mathbb{E}_\pi[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \gamma^3 R_{t+4} \ldots | S_t = s, A_t = a]$

   - ☐ $q_\pi(s,a) = \mathbb{E}_\pi[R_{t+1} | S_t = s, A_t = a]$

4. Imagine the agent is learning in an episodic problem. Which of the following is true? 1 point

   - ○ The agent takes the same action at each step during an episode.
   - ◉ The number of steps in an episode is stochastic: each episode can have a different number of steps.
   - ○ The number of steps in an episode is always the same.

5. If the reward is always +1 what is the sum of the discounted infinite return when $\gamma < 1$ 1 point

$G_t = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$

- ( ) Infinity.
- (•) $G_t = \frac{1}{1-\gamma}$
- ( ) $G_t = 1 * \gamma^k$
- ( ) $G_t = \frac{\gamma}{1-\gamma}$

6. What is the difference between a small gamma (discount factor) and a large gamma? 1 point

- ( ) With a smaller discount factor the agent is more far-sighted and considers rewards farther into the future.
- (•) With a larger discount factor the agent is more far-sighted and considers rewards farther into the future.
- ( ) The size of the discount factor has no effect on the agent.

7. Suppose $\gamma = 0.8$ and we observe the following sequence of rewards: $R_1 = -3$, $R_2 = 5$, $R_3 = 2$, $R_4 = 7$, and $R_5 = 1$, with $T = 5$. What is G_0? Hint: Work Backwards and recall that $G_t = R_{t+1} + \gamma G_{t+1}$. 1 point

- (•) 6.2736
- ( ) 11.592
- ( ) 12
- ( ) -3
- ( ) 8.24

8. Suppose $\gamma = 0.8$ and the reward sequence is $R_1 = 5$ followed by an infinite sequence of 10s. What is G_0? 1 point

- (•) 45
- ( ) 15
- ( ) 55

9. Suppose reinforcement learning is being applied to determine moment-by-moment temperatures and stirring rates for a bioreactor (a large vat of nutrients and bacteria used to produce useful chemicals). The actions in such an application might be target temperatures and target stirring rates that are passed to lower-level control systems that, in turn, directly activate heating elements and motors to attain the targets. The states are likely to be thermocouple and other sensory readings, perhaps filtered and delayed, plus symbolic inputs representing the ingredients in the vat and the target chemical. The rewards might be moment-by-moment measures of the rate at which the useful chemical is produced by the bioreactor. Notice that here each state is a list, or vector, of sensor readings and symbolic inputs, and each action is a vector consisting of a target temperature and a stirring rate. Is this a valid MDP? 1 point

- ( ) Yes
- (•) No

10. Consider using reinforcement learning to control the motion of a robot arm in a repetitive pick-and-place task. If we want to learn movements that are fast and smooth, the learning agent will have to control the motors directly and have low-latency information about the current positions and velocities of the mechanical linkages. The actions in this case might be the voltages applied to each motor at each joint, and the states might be the latest readings of joint angles and velocities. The reward might be +1 for each object successfully picked up and placed. To encourage smooth movements, on each time step a small, negative reward can be given as a function of the moment-to-moment "jerkiness" of the motion. Is this a valid MDP? — 1 point

- (x) Yes
- ( ) No

11. Imagine that you are a vision system. When you are first turned on for the day, an image floods into your camera. You can see lots of things, but not all things. You can't see objects that are occluded, and of course you can't see objects that are behind you. After seeing that first scene, do you have access to the Markov state of the environment? Suppose your camera was broken that day and you received no images at all, all day. Would you have access to the Markov state then? — 1 point

- ( ) You have access to the Markov state before and after damage.
- (x) You have access to the Markov state before damage, but you don't have access to the Markov state after damage.
- ( ) You don't have access to the Markov state before damage, but you do have access to the Markov state after damage.
- ( ) You don't have access to the Markov state before or after damage.

12. What does MDP stand for? — 1 point

- (x) Markov Decision Process
- ( ) Meaningful Decision Process
- ( ) Markov Decision Protocol
- ( ) Markov Deterministic Policy

13. What is the reward hypothesis? — 1 point

- (x) Goals and purposes can be thought of as the maximization of the expected value of the cumulative sum of rewards received.
- ( ) Always take the action that gives you the best reward at that point.
- ( ) Goals and purposes can be thought of as the minimization of the expected value of the cumulative sum of rewards received.
- ( ) Ignore rewards and find other signals.

14. Imagine, an agent is in a maze-like gridworld. You would like the agent to find the goal, as quickly as possible. You give the agent a reward of +1 when it reaches the goal and the discount rate is 1.0, because this is an episodic task. When you run the agent its finds the goal, but does not seem to care how long it takes to complete each episode. How could you fix this? (Select all that apply) — 1 point

- [ ] Give the agent a reward of 0 at every time step so it wants to leave.
- [ ] Give the agent a reward of +1 at every time step.
- [x] Give the agent -1 at each time step.
- [x] Set a discount rate less than 1 and greater than 0, like 0.9.

15. When may you want to formulate a problem as episodic?

1 point

- ○ When the agent-environment interaction does not naturally break into sequences. Each new episode begins independently of how the previous episode ended.
- ◉ When the agent-environment interaction naturally breaks into sequences. Each sequence begins independently of how the episode ended.

16. When may you want to formulate a problem as continuing?

1 point

- ◉ When the agent-environment interaction does not naturally break into sequences. Each new episode begins independently of how the previous episode ended.
- ○ When the agent-environment interaction naturally breaks into sequences and each sequence begins independently of how the previous sequence ended.

QUIZ 2 – MARKOV DECISION PROCESS

FEEDBACK

MDPs
Practice Quiz • 45 min

✓ **Congratulations! You passed!**
TO PASS 80% or higher

Keep Learning

GRADE
87.50%

MDPs

TOTAL POINTS 16

1. The learner and decision maker is the ______. **1 / 1 point**

- ○ Reward
- ○ Environment
- ○ State
- ◉ Agent

✓ **Correct**
Correct!

2. At each time step the agent takes an ______. **1 / 1 point**

- ○ Reward
- ○ Environment
- ◉ Action
- ○ State

✓ **Correct**
Correct!

3. What equation(s) define $q_\pi(S_t, A_t)$ in terms of subsequent rewards? **1 / 1 point**

- [x] $q_\pi(s, a) = \mathbb{E}_\pi[G_t | S_t = s, A_t = a]$

  where: $G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \gamma^3 R_{t+4} \ldots$

  ✓ **Correct**
  Correct!

- [ ] $q_\pi(s, a) = [G_t | S_t = s, A_t = a]$

  where: $G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \gamma^3 R_{t+4} \ldots$

- [ ] $q_\pi(s, a) = \mathbb{E}_\pi[G_t]$

  where: $G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \gamma^3 R_{t+4} \ldots$

- [x] $q_\pi(s, a) = \mathbb{E}_\pi[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \gamma^3 R_{t+4} \ldots | S_t = s, A_t = a]$

  ✓ **Correct**

4. Imagine the agent is learning in an episodic problem. Which of the following is true?

1 / 1 point

- ◯ The agent takes the same action at each step during an episode.
- ◉ The number of steps in an episode is stochastic: each episode can have a different number of steps.
- ◯ The number of steps in an episode is always the same.

✓ **Correct**

Correct!

5. If the reward is always +1 what is the sum of the discounted infinite return when $\gamma < 1$

1 / 1 point

$G_t = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$

- ◯ Infinity.
- ◉ $G_t = \frac{1}{1-\gamma}$
- ◯ $G_t = 1 * \gamma^k$
- ◯ $G_t = \frac{\gamma}{1-\gamma}$

✓ **Correct**

Correct!

6. What is the difference between a small gamma (discount factor) and a large gamma?

1 / 1 point

- ◯ With a smaller discount factor the agent is more far-sighted and considers rewards farther into the future.
- ◉ With a larger discount factor the agent is more far-sighted and considers rewards farther into the future.
- ◯ The size of the discount factor has no effect on the agent.

✓ **Correct**

Correct!

7. Suppose $\gamma = 0.8$ and we observe the following sequence of rewards: $R_1 = -3$, $R_2 = 5$, $R_3 = 2$, $R_4 = 7$, and $R_5 = 1$, with $T = 5$. What is G_0? Hint: Work Backwards and recall that $G_t = R_{t+1} + \gamma G_{t+1}$.

1 / 1 point

- ◉ 6.2736
- ◯ 11.592
- ◯ 12
- ◯ -3
- ◯ 8.24

✓ **Correct**

Correct!

8. Suppose $\gamma = 0.8$ and the reward sequence is $R_1 = 5$ followed by an infinite sequence of 10s. What is G_0?

1 / 1 point

- (●) 45
- ( ) 15
- ( ) 55

Correct

Correct!

$G_2 = 10/(1 - 0.8) = 50$

$G_1 = 10 + .8 * (50) = 50$

$G_0 = 5 + .8 * 50 = 45$

9. Suppose reinforcement learning is being applied to determine moment-by-moment temperatures and stirring rates for a bioreactor (a large vat of nutrients and bacteria used to produce useful chemicals). The actions in such an application might be target temperatures and target stirring rates that are passed to lower-level control systems that, in turn, directly activate heating elements and motors to attain the targets. The states are likely to be thermocouple and other sensory readings, perhaps filtered and delayed, plus symbolic inputs representing the ingredients in the vat and the target chemical. The rewards might be moment-by-moment measures of the rate at which the useful chemical is produced by the bioreactor. Notice that here each state is a list, or vector, of sensor readings and symbolic inputs, and each action is a vector consisting of a target temperature and a stirring rate. Is this a valid MDP?

0 / 1 point

- ( ) Yes
- (●) No

Incorrect

Incorrect. Review section 3.1 in the textbook.

10. Consider using reinforcement learning to control the motion of a robot arm in a repetitive pick-and-place task. If we want to learn movements that are fast and smooth, the learning agent will have to control the motors directly and have low-latency information about the current positions and velocities of the mechanical linkages. The actions in this case might be the voltages applied to each motor at each joint, and the states might be the latest readings of joint angles and velocities. The reward might be +1 for each object successfully picked up and placed. To encourage smooth movements, on each time step a small, negative reward can be given as a function of the moment-to-moment "jerkiness" of the motion. Is this a valid MDP?

1 / 1 point

- (●) Yes
- ( ) No

Correct

Correct!

11. Imagine that you are a vision system. When you are first turned on for the day, an image floods into your camera. You can see lots of things, but not all things. You can't see objects that are occluded, and of course you can't see objects that are behind you. After seeing that first scene, do you have access to the Markov state of the environment? Suppose your camera was broken that day and you received no images at all, all day. Would you have access to the Markov state then?

0 / 1 point

- ○ You have access to the Markov state before and after damage.
- ◉ You have access to the Markov state before damage, but you don't have access to the Markov state after damage.
- ○ You don't have access to the Markov state before damage, but you do have access to the Markov state after damage.
- ○ You don't have access to the Markov state before or after damage.

! **Incorrect**

Incorrect. Because there is no history before the first image, the first state has the Markov property. The Markov property does not mean that the state representation tells all that would be useful to know, only that it has not forgotten anything that would be useful to know. The case when the camera is broken is different, but again we have the Markov property. The key in this case is that the future is impoverished. All the possible futures are the same (all blank), so nothing need be remembered in order to predict them.

12. What does MDP stand for?

1 / 1 point

- ◉ Markov Decision Process
- ○ Meaningful Decision Process
- ○ Markov Decision Protocol
- ○ Markov Deterministic Policy

✓ **Correct**

Correct!

13. What is the reward hypothesis?

1 / 1 point

- ◉ Goals and purposes can be thought of as the maximization of the expected value of the cumulative sum of rewards received.
- ○ Always take the action that gives you the best reward at that point.
- ○ Goals and purposes can be thought of as the minimization of the expected value of the cumulative sum of rewards received.
- ○ Ignore rewards and find other signals.

✓ **Correct**

Correct!

14. Imagine, an agent is in a maze-like gridworld. You would like the agent to find the goal, as quickly as possible. You give the agent a reward of +1 when it reaches the goal and the discount rate is 1.0, because this is an episodic task. When you run the agent its finds the goal, but does not seem to care how long it takes to complete each episode. How could you fix this? (Select all that apply)

1 / 1 point

- [ ] Give the agent a reward of 0 at every time step so it wants to leave.
- [ ] Give the agent a reward of +1 at every time step.
- [x] Give the agent -1 at each time step.

✓ **Correct**

Correct! Giving the agent a negative reward on each time step, tells the agent to complete each episode as quickly as possible.

- [x] Set a discount rate less than 1 and greater than 0, like 0.9.

✓ **Correct**

Correct! From a given state, the sooner you get the +1 reward, the larger the return. The agent is incentivized to reach the goal faster to maximize expected return.

---

15. When may you want to formulate a problem as episodic?

1 / 1 point

○ When the agent-environment interaction does not naturally break into sequences. Each new episode begins independently of how the previous episode ended.

◉ When the agent-environment interaction naturally breaks into sequences. Each sequence begins independently of how the episode ended.

✓ **Correct**

Correct!

16. When may you want to formulate a problem as continuing?

1 / 1 point

◉ When the agent-environment interaction does not naturally break into sequences. Each new episode begins independently of how the previous episode ended.

○ When the agent-environment interaction naturally breaks into sequences and each sequence begins independently of how the previous sequence ended.

✓ **Correct**

Correct!

Peer-graded Assignment: Graded Assignment: Describe Three MDPs

Was due Aug 23, 11:59 PM PDT

i

It looks like this is your first peer-graded assignment. Learn more

Submit Now

Your assignment was due on Aug 23, 11:59 PM PDT, but you still have a chance! Start now so you have time to submit and then review your peers' assignments - and so they have time to review yours.

1. **Instructions**
2. **My submission**
3. **Discussions**

For this assignment you will get experience thinking about Markov Decision Processes (MDPs) and how to think about them. You will devise three example tasks of your own that fit into the MDP framework, identifying for each its states, actions, and rewards. Make the three examples as different from each other as possible.

Review criteria
less

You will be graded on each MDP separately. The grading criteria is:

1. That you have described an MDP and that it is different than your other two.
2. That you have described the MDP's states.
3. That you have described the MDP's actions.
4. That you have described the MDP's rewards.

Example Submissions
less

An example of an MDP could be a self driving car. The states would be all of the sensor readings that car gets at each time step: LIDAR, cameras, the amount of fuel left, current wheel angle, current velocity, gps location. The actions could be accelerate, decelerate, turn wheels left, and turn wheels right. The rewards could be -1 at every time step so that the agent is encouraged to get to the goal as quickly as possible, but -1 billion if it crashes or breaks the law so that it knows not to do that.

Peer-graded Assignment: Graded Assignment: Describe Three MDPs

Was due Aug 23, 11:59 PM PDT

It looks like this is your first peer-graded assignment. Learn more

Submit Now

Your assignment was due on Aug 23, 11:59 PM PDT, but you still have a chance! Start now so you have time to submit and then review your peers' assignments - and so they have time to review yours.

MDP Example 1: Program for automation of the learning about topics of Calculus. The program consists of displaying a list of exercises of a multiple alternative about some Calculus topic with a correct alternative whose initial state is the "first topic in level 1". The states would be a sequential list of topics (organized according to a curriculum) and their difficulty levels. Actions would be to advance to the next difficulty level of a topic, go back in a difficulty level, advance to a new topic, go back a topic, or stay at the same level of a topic, displaying a set of questions corresponding this topic and level. The rewards would be +10 for each advance (level or topic with a condition to complete correctly more than 80% of questions), 0 for continue in same state (complete less than 80% of questions), and -1 for go back (level or topic if complete less than 50% of questions).

B I

Create an MDP. Remember to describe the states, actions and rewards. Make sure your three MDPs are different from each other.

MDP Example 2: Program to control balance personal credit card. The program consists of managing the status of the credit card in such a way that it encourages payments to date while avoiding late payments or blocking the card for the financial health of the user. The states would be the condition of the card can be {created, paid daily, late paid, blocked, exceeded}. The actions of the system can be send payment warnings, block the card or provide more credit. The rewards would be +1 for more credit provided, a penalty for infractions such as -10 for each payment warning sent, -1000 for each block card operation.

B I

MDP Example 3: Program to automatic pilot drone for deliveries. The program consists of the drone being able to fly cities and using its GPS navigation system to reach a destination and make the delivery. The states would be the longitude and latitude of the current position and the target, battery level, the pixels frame of the images about the environment flown over. The actions are the controls up, down, right and left; increase and decrease the speed of the drone motor. The rewards can be +100000 each time drone reach the target coordinates, the penalties can be -1 for each action taken over time to fly over, -1000 each time the drone collides with an obstacle and -100 each time the drone requires recharging its battery.

B I

- [x] I understand that submitting work that isn't my own may result in permanent failure of this course or deactivation of my Coursera account.

Learn more about Coursera's Honor Code

Holger Rivera

FEEDBACK – 3 MDPS

Peer-graded Assignment: Graded Assignment: Describe Three MDPs

ⓘ It looks like this is your first peer-graded assignment. Learn more

You passed!

Congratulations. You earned 12 / 12 points. Review the feedback below and continue the course when you are ready. You can also help more peers by reviewing their submissions.

Review assignments

PROMPT

Create an MDP. Remember to describe the states, actions and rewards. Make sure your three MDPs are different from each other.

MDP Example 1: Program for automation of the learning about topics of Calculus. The program consists of displaying a list of exercises of a multiple alternative about some Calculus topic with a correct alternative whose initial state is the "first topic in level 1". The states would be a sequential list of topics (organized according to a curriculum) and their difficulty levels. Actions would be to advance to the next difficulty level of a topic, go back in a difficulty level, advance to a new topic, go back a topic, or stay at the same level of a topic, displaying a set of other questions corresponding this topic and level. The rewards would be +10 for each advance (level or topic with a condition to complete correctly more than 80% of questions), 0 for continue in same state (complete less than 80% of questions), and -1 for go back (level or topic if complete less than 50% of questions).

RUBRIC

Did the learner describe an MDP, and is it different than their other submissions?

- 0 points
  No
- **1 point**
  Yes

Are the **states** well-specified? Namely are they Markov and so can be used as MDP states.

- 0 points
  No
- **1 point**
  Yes

Are the **actions** well-specified? Namely can they used as actions in an MDP.

- 0 points
  No
- **1 point**
  Yes

Are the **rewards** well-specified? Namely to satisfy the requirements in the definition of an MDP with the described state and action set.

- 0 points
  No
- **1 point**
  Yes

PROMPT	RUBRIC
Create an MDP. Remember to describe the states, actions and rewards. Make sure your three MDPs are different from each other. **MDP Example 2: Program to control balance personal credit card.** The program consists of managing the status of the credit card in such a way that it encourages payments to date while avoiding late payments or blocking the card for the financial health of the user. The states would be the condition of the card can be {created, paid daily, late paid, blocked, exceeded}. The actions of the system can be send payment warnings, block the card or provide more credit. The rewards would be +1 for more credit provided, a penalty for infractions such as -10 for each payment warning sent, -1000 for each block card operation.	Did the learner describe an MDP, and is it different than their other submissions? 0 points No **1 point** **Yes** Are the **states** well-specified? Namely are they Markov and so can be used as MDP states. 0 points No **1 point** **Yes**
	Are the **actions** well-specified? Namely can they used as actions in an MDP. 0 points No **1 point** **Yes** Are the **rewards** well-specified? Namely to satisfy the requirements in the definition of an MDP with the described state and action set. 0 points No **1 point** **Yes**

PROMPT

Create an MDP. Remember to describe the states, actions and rewards. Make sure your three MDPs are different from each other.

MDP Example 3: Program to automatic pilot drone for deliveries. The program consists of the drone being able to fly cities and using its GPS navigation system to reach a destination and make the delivery. The states would be the longitude and latitude of the current position and the target, battery level, the pixels frame of the images about the environment flown over. The actions are the controls up, down, right and left; increase and decrease the speed of the drone motor. The rewards can be +100000 each time drone reach the target coordinates, the penalties can be -1 for each action taken over time to fly over, -1000 each time the drone collides with an obstacle and -100 each time the drone requires recharging its battery.

RUBRIC

Did the learner describe an MDP, and is it different than their other submissions?

- 0 points
  No
- **1 point**
  Yes

Are the **states** well-specified? Namely are they Markov and so can be used as MDP states.

- 0 points
  No
- **1 point**
  Yes

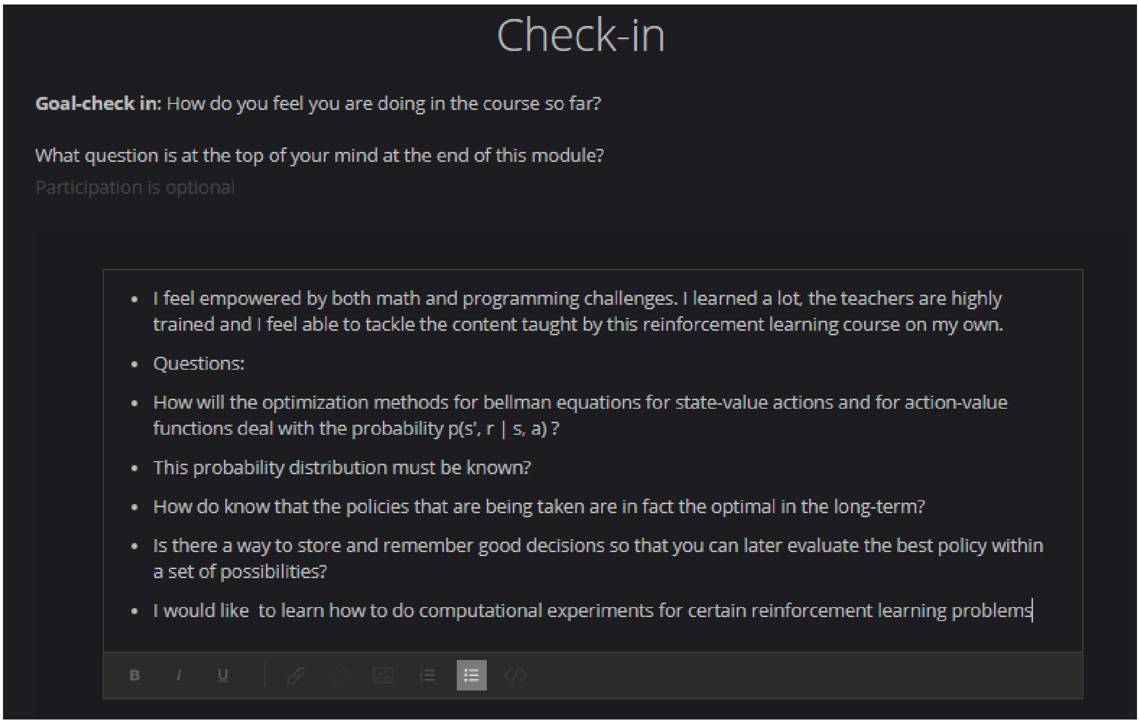
Are the **actions** well-specified? Namely can they used as actions in an MDP.

- 0 points
  No
- **1 point**
  Yes

Are the **rewards** well-specified? Namely to satisfy the requirements in the definition of an MDP with the described state and action set.

- 0 points
  No
- **1 point**
  Yes

CHECK-IN ABOUT COURSE OVERVIEW



Check-in

Goal-check in: How do you feel you are doing in the course so far?

What question is at the top of your mind at the end of this module?

Participation is optional

- I feel empowered by both math and programming challenges. I learned a lot, the teachers are highly trained and I feel able to tackle the content taught by this reinforcement learning course on my own.
- Questions:
- How will the optimization methods for bellman equations for state-value actions and for action-value functions deal with the probability p(s', r | s, a) ?
- This probability distribution must be known?
- How do know that the policies that are being taken are in fact the optimal in the long-term?
- Is there a way to store and remember good decisions so that you can later evaluate the best policy within a set of possibilities?
- I would like to learn how to do computational experiments for certain reinforcement learning problems

QUIZ 3 – VALUE FUNCTIONS AND BELLMAN EQUATIONS

Value Functions and Bellman Equations

TOTAL POINTS 10

1. A policy is a function which maps ____ to ____. 1 point

- ○ States to values.
- ○ States to probability distributions over actions.
- ○ Actions to probabilities.
- ◉ States to actions.
- ○ Actions to probability distributions over values.

2. The term "backup" most closely resembles the term ___ in meaning. 1 point

- ○ Value
- ○ Update
- ◉ Diagram

3. At least one deterministic optimal policy exists in every Markov decision process. 1 point

- ○ False
- ◉ True

4. The optimal state-value function: 1 point

- ◉ Is unique in every finite Markov decision process.
- ○ Is not guaranteed to be unique, even in finite Markov decision processes.

5. Does adding a constant to all rewards change the set of optimal policies in episodic tasks? 1 point

- ○ Yes, adding a constant to all rewards changes the set of optimal policies.
- ◉ No, as long as the relative differences between rewards remain the same, the set of optimal policies is the same.

6. Does adding a constant to all rewards change the set of optimal policies in continuing tasks? 1 point

- ◉ Yes, adding a constant to all rewards changes the set of optimal policies.
- ○ No, as long as the relative differences between rewards remain the same, the set of optimal policies is the same.

7. Select the equation that correctly relates v_* to q_*. Assume π is the uniform random policy. 1 point

- ( ) $v_*(s) = \sum_{a,r,s'} \pi(a|s)p(s',r|s,a)[r + q_*(s')]$
- (•) $v_*(s) = max_a q_*(s,a)$
- ( ) $v_*(s) = \sum_{a,r,s'} \pi(a|s)p(s',r|s,a)q_*(s')$
- ( ) $v_*(s) = \sum_{a,r,s'} \pi(a|s)p(s',r|s,a)[r + \gamma q_*(s')]$

8. Select the equation that correctly relates q_* to v_* using four-argument function p. 1 point

- ( ) $q_*(s,a) = \sum_{s',r} p(s',r|a,s)[r + v_*(s')]$
- ( ) $q_*(s,a) = \sum_{s',r} p(s',r|a,s)\gamma[r + v_*(s')]$
- (•) $q_*(s,a) = \sum_{s',r} p(s',r|a,s)[r + \gamma v_*(s')]$

9. Write a policy π_* in terms of q_*. 1 point

- ( ) $\pi_*(a|s) = q_*(s,a)$
- ( ) $\pi_*(a|s) = \max_{a'} q_*(s,a')$
- (•) $\pi_*(a|s) = 1 \text{ if } a = \text{argmax}_{a'} q_*(s,a'), \text{ else } 0$

10. Give an equation for some π_* in terms of v_* and the four-argument p. 1 point

- ( ) $\pi_*(a|s) = \sum_{s',r} p(s',r|s,a)[r + \gamma v_*(s')]$
- ( ) $\pi_*(a|s) = 1 \text{ if } v_*(s) = \sum_{s',r} p(s',r|s,a)[r + \gamma v_*(s')], \text{ else } 0$
- ( ) $\pi_*(a|s) = \max_{a'} \sum_{s',r} p(s',r|s,a')[r + \gamma v_*(s')]$
- (•) $\pi_*(a|s) = 1 \text{ if } v_*(s) = \max_{a'} \sum_{s',r} p(s',r|s,a')[r + \gamma v_*(s')], \text{ else } 0$

FEEDBACK – QUIZ 3

Congratulations! You passed!

TO PASS 80% or higher

Keep Learning

GRADE
80%

Value Functions and Bellman Equations

TOTAL POINTS 10

1. A policy is a function which maps ____ to ____.

1 / 1 point

- ○ States to actions.
- ◉ States to probability distributions over actions.
- ○ Actions to probability distributions over values.
- ○ States to values.
- ○ Actions to probabilities.

✓ Correct
Correct!

2. The term "backup" most closely resembles the term ___ in meaning.

1 / 1 point

- ○ Value
- ◉ Update
- ○ Diagram

✓ Correct
Correct!

3. At least one deterministic optimal policy exists in every Markov decision process.

1 / 1 point

- ○ False
- ◉ True

✓ Correct
Correct! Let's say there is a policy π_1 which does well in some states, while policy π_2 does well in others. We could combine these policies into a third policy π_3, which always chooses actions according to whichever of policy π_1 and π_2 has the highest value in the current state. π_3 will necessarily have a value greater than or equal to both π_1 and π_2 in every state! So we will never have a situation where doing well in one state requires sacrificing value in another. Because of this, there always exists some policy which is best in every state. This is of course only an informal argument, but there is in fact a rigorous proof showing that there must always exist at least one optimal deterministic policy.

4. The optimal state-value function:

1 / 1 point

- (selected) Is unique in every finite Markov decision process.
- Is not guaranteed to be unique, even in finite Markov decision processes.

✓ **Correct**

Correct! The Bellman optimality equation is actually a system of equations, one for each state, so if there are N states, then there are N equations in N unknowns. If the dynamics of the environment are known, then in principle one can solve this system of equations for the optimal value function using any one of a variety of methods for solving systems of nonlinear equations. All optimal policies share the same optimal state-value function.

5. Does adding a constant to all rewards change the set of optimal policies in episodic tasks?

1 / 1 point

- (selected) Yes, adding a constant to all rewards changes the set of optimal policies.
- No, as long as the relative differences between rewards remain the same, the set of optimal policies is the same.

✓ **Correct**

Correct! Adding a constant to the reward signal can make longer episodes more or less advantageous (depending on whether the constant is positive or negative).

6. Does adding a constant to all rewards change the set of optimal policies in continuing tasks?

0 / 1 point

- No, as long as the relative differences between rewards remain the same, the set of optimal policies is the same.
- (selected) Yes, adding a constant to all rewards changes the set of optimal policies.

! **Incorrect**

Incorrect. Since the task is continuing, the agent will accumulate the same amount of extra reward independent of its behavior.

7. Select the equation that correctly relates v_* to q_*. Assume π is the uniform random policy.

1 / 1 point

- $v_*(s) = \sum_{a,r,s'} \pi(a|s)p(s',r|s,a)q_*(s')$
- $v_*(s) = \sum_{a,r,s'} \pi(a|s)p(s',r|s,a)[r + \gamma q_*(s')]$
- (selected) $v_*(s) = max_a q_*(s,a)$
- $v_*(s) = \sum_{a,r,s'} \pi(a|s)p(s',r|s,a)[r + q_*(s')]$

✓ **Correct**

Correct!

8. Select the equation that correctly relates q_* to v_* using four-argument function p.

1 / 1 point

- $q_*(s,a) = \sum_{s',r} p(s',r|a,s)[r + v_*(s')]$
- $q_*(s,a) = \sum_{s',r} p(s',r|a,s)\gamma[r + v_*(s')]$
- (selected) $q_*(s,a) = \sum_{s',r} p(s',r|a,s)[r + \gamma v_*(s')]$

✓ **Correct**

Correct!

9. Write a policy π_* in terms of q_*.

1 / 1 point

- ( ) $\pi_*(a|s) = q_*(s,a)$
- ( ) $\pi_*(a|s) = \max_{a'} q_*(s,a')$
- (•) $\pi_*(a|s) = 1 \text{ if } a = \text{argmax}_{a'} q_*(s,a'), \text{ else } 0$

Correct

Correct!

10. Give an equation for some π_* in terms of v_* and the four-argument p.

0 / 1 point

- ( ) $\pi_*(a|s) = 1 \text{ if } v_*(s) = \sum_{s',r} p(s',r|s,a)[r + \gamma v_*(s')], \text{ else } 0$
- (•) $\pi_*(a|s) = 1 \text{ if } v_*(s) = \max_{a'} \sum_{s',r} p(s',r|s,a')[r + \gamma v_*(s')], \text{ else } 0$
- ( ) $\pi_*(a|s) = \sum_{s',r} p(s',r|s,a)[r + \gamma v_*(s')]$
- ( ) $\pi_*(a|s) = \max_{a'} \sum_{s',r} p(s',r|s,a')[r + \gamma v_*(s')]$

! Incorrect

Incorrect. This equation will give a probability of 1 to every action.

EXAM – WEEK 3

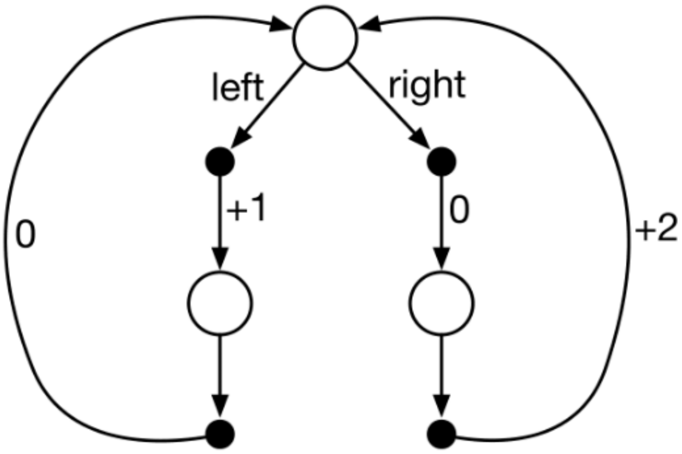
Value Functions and Bellman Equations

TOTAL POINTS 11

1. A function which maps ___ to ___ is a value function. [Select all that apply] 1 point

- [ ] Values to actions.
- [x] State-action pairs to expected returns.
- [ ] Values to states.
- [x] States to expected returns.

2. Consider the continuing Markov decision process shown below. The only decision to be made is in the top state, where two actions are available, left and right. The numbers show the rewards that are received deterministically after each action. There are exactly two deterministic policies, π_{left} and π_{right}. Indicate the optimal policies if $\gamma = 0$? If $\gamma = 0.9$? If $\gamma = 0.5$? [Select all that apply] 1 point



- [x] For $\gamma = 0, \pi_{\text{left}}$
- [ ] For $\gamma = 0.9, \pi_{\text{left}}$
- [x] For $\gamma = 0.5, \pi_{\text{right}}$
- [x] For $\gamma = 0.9, \pi_{\text{right}}$
- [ ] For $\gamma = 0, \pi_{\text{right}}$
- [x] For $\gamma = 0.5, \pi_{\text{left}}$

3. Every finite Markov decision process has __. [Select all that apply]

1 point

- [x] A unique optimal value function
- [ ] A unique optimal policy
- [x] A deterministic optimal policy
- [ ] A stochastic optimal policy

4. The ___ of the reward for each state-action pair, the dynamics function p, and the policy π is _____ to characterize the value function v_π. (Remember that the value of a policy π at state s is $v_\pi(s) = \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a)[r + \gamma v_\pi(s')]$.)

1 point

- ( ) Distribution; necessary
- (•) Mean; sufficient

5. The Bellman equation for a given a policy π: [Select all that apply]

1 point

- [x] Expresses state values $v(s)$ in terms of state values of successor states.
- [x] Holds only when the policy is greedy with respect to the value function.
- [ ] Expresses the improved policy in terms of the existing policy.

6. An optimal policy:

1 point

- (•) Is not guaranteed to be unique, even in finite Markov decision processes.
- ( ) Is unique in every Markov decision process.
- ( ) Is unique in every finite Markov decision process.

7. The Bellman optimality equation for v_*: [Select all that apply]

1 point

- [x] Holds when $v_* = v_\pi$ for a given policy π.
- [x] Holds for the optimal state value function.
- [x] Holds when the policy is greedy with respect to the value function.
- [x] Expresses state values $v_*(s)$ in terms of state values of successor states.
- [x] Expresses the improved policy in terms of the existing policy.

8. Give an equation for v_π in terms of q_π and π.

1 point

- (•) $v_\pi(s) = \sum_a \pi(a|s) q_\pi(s,a)$
- ( ) $v_\pi(s) = \max_a \gamma\pi(a|s) q_\pi(s,a)$
- ( ) $v_\pi(s) = \max_a \pi(a|s) q_\pi(s,a)$
- ( ) $v_\pi(s) = \sum_a \gamma\pi(a|s) q_\pi(s,a)$

9. Give an equation for q_π in terms of v_π and the four-argument p.

1 point

- ○ $q_\pi(s,a) = \sum_{s',r} p(s',r|s,a)\gamma[r + v_\pi(s')]$
- ○ $q_\pi(s,a) = \sum_{s',r} p(s',r|s,a)[r + v_\pi(s')]$
- ○ $q_\pi(s,a) = \max_{s',r} p(s',r|s,a)\gamma[r + v_\pi(s')]$
- ○ $q_\pi(s,a) = \max_{s',r} p(s',r|s,a)[r + \gamma v_\pi(s')]$
- ○ $q_\pi(s,a) = \max_{s',r} p(s',r|s,a)[r + v_\pi(s')]$
- ◉ $q_\pi(s,a) = \sum_{s',r} p(s',r|s,a)[r + \gamma v_\pi(s')]$

10. Let $r(s,a)$ be the expected reward for taking action a in state s, as defined in equation 3.5 of the textbook. Which of the following are valid ways to re-express the Bellman equations, using this expected reward function? [Select all that apply]

1 point

- [x] $v_*(s) = \max_a [r(s,a) + \gamma \sum_{s'} p(s'|s,a) v_*(s')]$
- [x] $q_\pi(s,a) = r(s,a) + \gamma \sum_{s',a'} p(s'|s,a)\pi(a'|s')q_\pi(s',a')$
- [x] $q_*(s,a) = r(s,a) + \gamma \sum_{s'} p(s'|s,a) \max_{a'} q_*(s',a')$
- [x] $v_\pi(s) = \sum_a \pi(a|s)[r(s,a) + \gamma \sum_{s'} p(s'|s,a) v_\pi(s')]$

11. Consider an episodic MDP with one state and two actions (left and right). The left action has stochastic reward 1 with probability p and 3 with probability $1 - p$. The right action has stochastic reward 0 with probability q and 10 with probability $1 - q$. What relationship between p and q makes the actions equally optimal?

1 point

- ○ $7 + 2p = -10q$
- ○ $13 + 3p = -10q$
- ○ $13 + 2p = 10q$
- ○ $13 + 3p = 10q$
- ◉ $7 + 2p = 10q$
- ○ $7 + 3p = -10q$
- ○ $13 + 2p = -10q$
- ○ $7 + 3p = 10q$

- [ ] I understand that submitting work that isn't my own may result in permanent failure of this course or deactivation of my Coursera account.

Learn more about Coursera's Honor Code

Holger Rivera

EXAM 3 – FEEDBACK WEEK 3

Congratulations! You passed!

TO PASS 80% or higher

Keep Learning

GRADE
81.81%

Value Functions and Bellman Equations

LATEST SUBMISSION GRADE
81.81%

1. A function which maps ___ to ___ is a value function. [Select all that apply]

1 / 1 point

- [ ] Values to actions.
- [x] State-action pairs to expected returns.

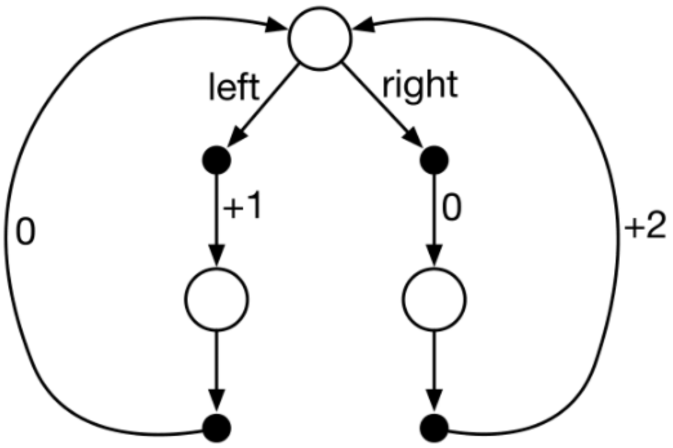
Correct
Correct! A function that takes a state-action pair and outputs an expected return is a value function.

- [ ] Values to states.
- [x] States to expected returns.

Correct
Correct! A function that takes a state and outputs an expected return is a value function.

2. Consider the continuing Markov decision process shown below. The only decision to be made is in the top state, where two actions are available, left and right. The numbers show the rewards that are received deterministically after each action. There are exactly two deterministic policies, π_{left} and π_{right}. Indicate the optimal policies if $\gamma = 0$? If $\gamma = 0.9$? If $\gamma = 0.5$? [Select all that apply]

1 / 1 point



- [x] For $\gamma = 0$, π_{left}

Correct

Correct! Since both policies return to the top state every two time steps, to determine the optimal policy, it suffices to consider the reward accumulated over the first two time steps. For the policy left, this is equal to 1; for the policy right, this is equal to 0.

- [ ] For $\gamma = 0.9$, π_{left}
- [x] For $\gamma = 0.5$, π_{right}

Correct

Correct! Since both policies return to the start state every two time steps, to determine the optimal policy, it suffices to consider the reward accumulated over the first two time steps. For the policy left, this is equal to 1; for the policy right, this is equal to 1.

- [x] For $\gamma = 0.9$, π_{right}

Correct

Correct! Since both policies return to the top state every two time steps, to determine the optimal policy, it suffices to consider the reward accumulated over the first two time steps. For the policy left, this is equal to 1; for the policy right, this is equal to 1.8.

- [ ] For $\gamma = 0$, π_{right}
- [x] For $\gamma = 0.5$, π_{left}

Correct

Correct! Since both policies return to the start state every two time steps, to determine the optimal policy, it suffices to consider the reward accumulated over the first two time steps. For the policy left, this is equal to 1; for the policy right, this is equal to 1.

3. Every finite Markov decision process has __. [Select all that apply]

1 / 1 point

- [x] A unique optimal value function

Correct

Correct! The Bellman optimality equation is actually a system of equations, one for each state, so if there are N states, then there are N equations in N unknowns. If the dynamics of the environment are known, then in principle one can solve this system of equations for the optimal value function using any one of a variety of methods for solving systems of nonlinear equations. All optimal policies share the same optimal state-value function.

- [ ] A unique optimal policy

- [x] A deterministic optimal policy

Correct

Correct! Let's say there is a policy π_1 which does well in some states, while policy π_2 does well in others. We could combine these policies into a third policy π_3, which always chooses actions according to whichever of policy π_1 and π_2 has the highest value in the current state. π_3 will necessarily have a value greater than or equal to both π_1 and π_2 in every state! So we will never have a situation where doing well in one state requires sacrificing value in another. Because of this, there always exists some policy which is best in every state. This is of course only an informal argument, but there is in fact a rigorous proof showing that there must always exist at least one optimal deterministic policy.

- [ ] A stochastic optimal policy

4. The __ of the reward for each state-action pair, the dynamics function p, and the policy π is ____ to characterize the value function v_π. (Remember that the value of a policy π at state s is $v_\pi(s) = \sum_a \pi(a|s) \sum_{s',r} p(s', r|s, a)[r + \gamma v_\pi(s')]$.)

1 / 1 point

( ) Distribution; necessary

(•) Mean; sufficient

Correct

Correct! If we have the expected reward for each state-action pair, we can compute the expected return under any policy.

5. The Bellman equation for a given a policy π: [Select all that apply]

0 / 1 point

- [x] Expresses state values $v(s)$ in terms of state values of successor states.

Correct

Correct!

- [x] Holds only when the policy is greedy with respect to the value function.

! This should not be selected

Incorrect. Take another look at the lesson: Optimal Policies.

- [ ] Expresses the improved policy in terms of the existing policy.

6. An optimal policy:

1 / 1 point

(•) Is not guaranteed to be unique, even in finite Markov decision processes.

( ) Is unique in every Markov decision process.

( ) Is unique in every finite Markov decision process.

Correct

Correct! For example, imagine a Markov decision process with one state and two actions. If both actions receive the same reward, then any policy is an optimal policy.

7. The Bellman optimality equation for v_*: [Select all that apply]

0 / 1 point

- [x] Holds when $v_* = v_\pi$ for a given policy π.

This should not be selected

Incorrect. Take another look at the lesson: Optimal Value Functions & Bellman Optimality Equation.

- [x] Holds for the optimal state value function.

Correct

Correct!

- [x] Holds when the policy is greedy with respect to the value function.

This should not be selected

Incorrect. Take another look at the lesson: Optimal Value Functions & Bellman Optimality Equation.

- [x] Expresses state values $v_*(s)$ in terms of state values of successor states.

Correct

Correct!

- [x] Expresses the improved policy in terms of the existing policy.

This should not be selected

Incorrect. Take another look at the lesson: Optimal Value Functions & Bellman Optimality Equation.

8. Give an equation for v_π in terms of q_π and π.

1 / 1 point

- (●) $v_\pi(s) = \sum_a \pi(a|s) q_\pi(s,a)$
- ( ) $v_\pi(s) = \max_a \gamma\pi(a|s) q_\pi(s,a)$
- ( ) $v_\pi(s) = \max_a \pi(a|s) q_\pi(s,a)$
- ( ) $v_\pi(s) = \sum_a \gamma\pi(a|s) q_\pi(s,a)$

Correct

Correct!

9. Give an equation for q_π in terms of v_π and the four-argument p.

1 / 1 point

- [ ] $q_\pi(s,a) = \sum_{s',r} p(s',r|s,a)\gamma[r + v_\pi(s')]$
- [ ] $q_\pi(s,a) = \sum_{s',r} p(s',r|s,a)[r + v_\pi(s')]$
- [ ] $q_\pi(s,a) = \max_{s',r} p(s',r|s,a)\gamma[r + v_\pi(s')]$
- [ ] $q_\pi(s,a) = \max_{s',r} p(s',r|s,a)[r + \gamma v_\pi(s')]$
- [ ] $q_\pi(s,a) = \max_{s',r} p(s',r|s,a)[r + v_\pi(s')]$
- [x] $q_\pi(s,a) = \sum_{s',r} p(s',r|s,a)[r + \gamma v_\pi(s')]$

Correct

Correct!

10. Let $r(s,a)$ be the expected reward for taking action a in state s, as defined in equation 3.5 of the textbook. Which of the following are valid ways to re-express the Bellman equations, using this expected reward function? [Select all that apply]

1 / 1 point

- [x] $v_*(s) = \max_a[r(s,a) + \gamma \sum_{s'} p(s'|s,a)v_*(s')]$

Correct

Correct!

- [x] $q_\pi(s,a) = r(s,a) + \gamma \sum_{s',a'} p(s'|s,a)\pi(a'|s')q_\pi(s',a')$

Correct

Correct!

- [x] $q_*(s,a) = r(s,a) + \gamma \sum_{s'} p(s'|s,a) \max_{a'} q_*(s',a')$

Correct

Correct!

- [x] $v_\pi(s) = \sum_a \pi(a|s)[r(s,a) + \gamma \sum_{s'} p(s'|s,a)v_\pi(s')]$

Correct

Correct!

11. Consider an episodic MDP with one state and two actions (left and right). The left action has stochastic reward 1 with probability p and 3 with probability $1-p$. The right action has stochastic reward 0 with probability q and 10 with probability $1-q$. What relationship between p and q makes the actions equally optimal?

1 / 1 point

- ○ $7+2p=-10q$
- ○ $13+3p=-10q$
- ○ $13+2p=10q$
- ○ $13+3p=10q$
- ◉ $7+2p=10q$
- ○ $7+3p=-10q$
- ○ $13+2p=-10q$
- ○ $7+3p=10q$

✓ **Correct**

Correct!

FEEDBACK EXAM – WEEK 4

Dynamic Programming

TOTAL POINTS 10

1. The value of any state under an optimal policy is ___ the value of that state under a non-optimal policy. [Select all that apply]

1 / 1 point

- [ ] Strictly greater than
- [x] Greater than or equal to

Correct
Correct! This follows from the policy improvement theorem.

- [ ] Strictly less than
- [ ] Less than or equal to

2. If a policy π is greedy with respect to its own value function v_π, then it is an optimal policy.

1 / 1 point

- (•) True
- ( ) False

Correct
Correct! If a policy is greedy with respect to its own value function, it follows from the policy improvement theorem and the Bellman optimality equation that it must be an optimal policy.

3. Let v_π be the state-value function for the policy π. Let $v_{\pi'}$ be the state-value function for the policy π'. Assume $v_\pi = v'_\pi$. Then this means that $\pi = \pi'$.

0 / 1 point

- ( ) True
- (•) False

Incorrect
Correct! For example, two policies might share the same value function, but differ due to random tie breaking.

4. What is the relationship between value iteration and policy iteration? [Select all that apply]

1 / 1 point

- [ ] Policy iteration is a special case of value iteration.
- [ ] Value iteration is a special case of policy iteration.
- [x] Value iteration and policy iteration are both special cases of generalized policy iteration.

Correct
Correct!

5. The word synchronous means "at the same time". The word asynchronous means "not at the same time". A dynamic programming algorithm is: [Select all that apply]

1 / 1 point

- [x] Asynchronous, if it updates some states more than others.

> **Correct**
> Correct! Only algorithms that update every state exactly once at each iteration are synchronous.

- [x] Asynchronous, if it does not update all states at each iteration.

> **Correct**
> Correct! Only algorithms that update every state exactly once at each iteration are synchronous.

- [x] Synchronous, if it systematically sweeps the entire state space at each iteration.

> **Correct**
> Correct! Only algorithms that update every state exactly once at each iteration are synchronous.

6. All Generalized Policy Iteration algorithms are synchronous.

1 / 1 point

- (●) False
- ( ) True

> **Correct**
> Correct! A Generalized Policy Iteration algorithm can update states in a non-systematic fashion.

7. Which of the following is true?

1 / 1 point

- (●) Asynchronous methods generally scale to large state spaces better than synchronous methods.
- ( ) Synchronous methods generally scale to large state spaces better than asynchronous methods.

> **Correct**
> Correct! Asynchronous methods can focus updates on more relevant states, and update less relevant states less often. If the state space is very large, asynchronous methods may still be able to achieve good performance whereas even just one synchronous sweep of the state space may be intractable.

8. Why are dynamic programming algorithms considered planning methods? [Select all that apply]

1 / 1 point

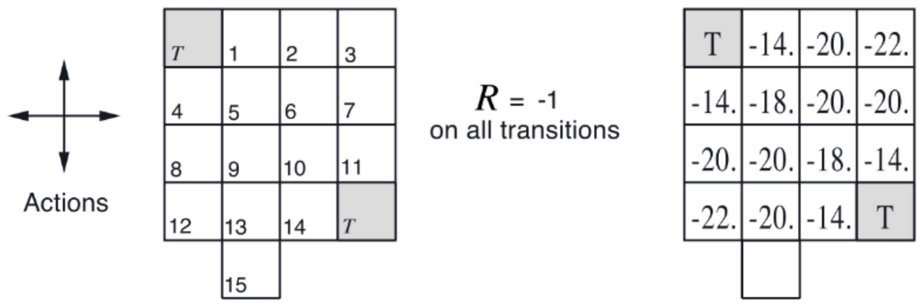
- [ ] They compute optimal value functions.
- [x] They use a model to improve the policy.

> **Correct**
> Correct! This is the definition of a planning method.

- [ ] They learn from trial and error interaction.

9. Consider the undiscounted, episodic MDP below. There are four actions possible in each state, A = {up, down, right, left}, which deterministically cause the corresponding state transitions, except that actions that would take the agent off the grid in fact leave the state unchanged. The right half of the figure shows the value of each state under the equiprobable random policy. If π is the equiprobable random policy, what is $q(7, \text{down})$?

1 / 1 point



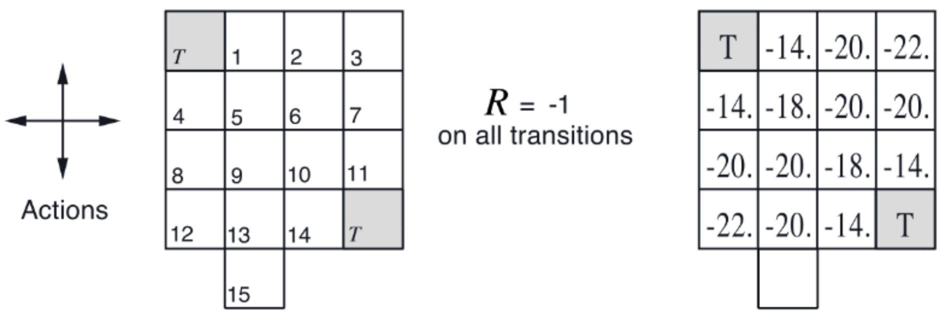
- ○ $q(7, \text{down}) = -14$
- ○ $q(7, \text{down}) = -20$
- ○ $q(7, \text{down}) = -21$
- ◉ $q(7, \text{down}) = -15$

✓ **Correct**

Correct! Moving down incurs a reward of -1 before reaching state 11, from which the expected future return is -14.

10. Consider the undiscounted, episodic MDP below. There are four actions possible in each state, A = {up, down, right, left}, which deterministically cause the corresponding state transitions, except that actions that would take the agent off the grid in fact leave the state unchanged. The right half of the figure shows the value of each state under the equiprobable random policy. If π is the equiprobable random policy, what is $v(15)$? Hint: Recall the Bellman equation $v(s) = \sum_a \pi(a|s) \sum_{s',r} p(s', r|s, a)[r + \gamma v(s')]$.

1 / 1 point



- [ ] $v(15) = -21$
- [ ] $v(15) = -22$
- [ ] $v(15) = -25$
- [ ] $v(15) = -23$
- [x] $v(15) = -24$

Correct

Correct! We can get this by solving for the unknown variable $v(15)$. Let's call this unknown x. We solve for x in the equation $x = 1/4(-21) + 3/4(-1 + x)$. The first term corresponds to transitioning to state 13. The second term corresponds to taking one of the other three actions, incurring a reward of -1 and staying in state x.

FINAL CONSIDERATIONS

Where can you use dynamic programming?

Where can you use dynamic programming? Discuss problems that you have encountered that could be solved with dynamic programming methods. What are the advantages of using DP to solve these? What are the disadvantages**?**

Your response has been submitted. Engage and discuss with other learners below!

View My Response

HR

Holger Rivera · 2 minutes ago

- Finance such as Stock Market Prediction (Time series and Dynamic Programming)
- Problems that can be modeled with linear or integer programming that define an objective function
- Optimization problems that are initially modeled by recursive methods and then can be transformed into iterative DP Methods
- Gaming problems: DOTA, GO, chess and ATARI games use Dynamic Programming with approach
- Problems of Industrial Control
- Robotics and Automation

Advantages:

- More efficient to use computational resources such as space and time.
- Dynamic Programming is a good way to transform recursive problem in iterative problem efficiently
- In long complex problems DP requires flexibility to achieve better results

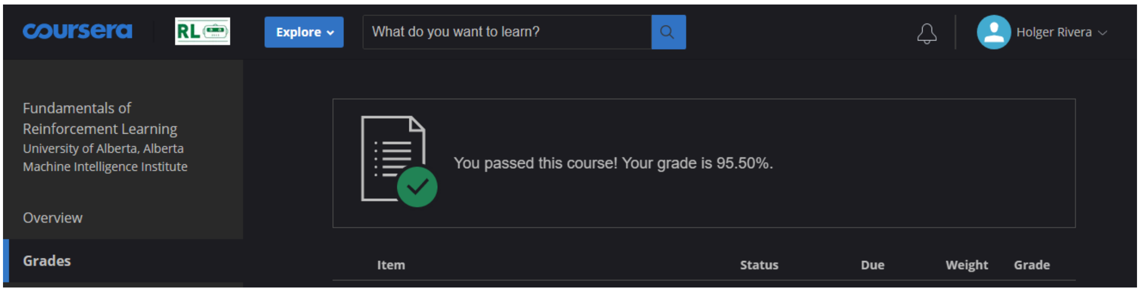
Disadvantages:

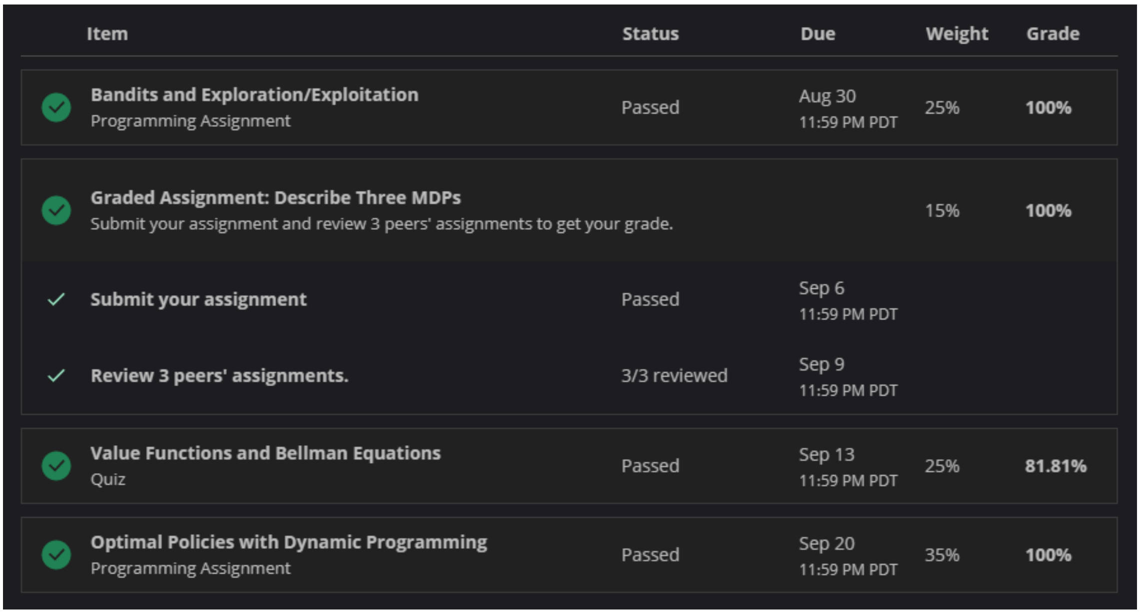
- Some problems a complexity scale, and DP is not sufficient to cover all the complexity. This requires a combination of other approaches as time series, genetic algorithms or graph theory to achieve more robust models and algorithms.

0 Upvotes

Reply

FINAL RESULTS





Item	Status	Due	Weight	Grade
Bandits and Exploration/Exploitation Programming Assignment	Passed	Aug 30 11:59 PM PDT	25%	**100%**
Graded Assignment: Describe Three MDPs Submit your assignment and review 3 peers' assignments to get your grade.			15%	**100%**
Submit your assignment	Passed	Sep 6 11:59 PM PDT		
Review 3 peers' assignments.	3/3 reviewed	Sep 9 11:59 PM PDT		
Value Functions and Bellman Equations Quiz	Passed	Sep 13 11:59 PM PDT	25%	**81.81%**
Optimal Policies with Dynamic Programming Programming Assignment	Passed	Sep 20 11:59 PM PDT	35%	**100%**