


1. (9 points) SOLVING MDPs

Consider the gridworld MDP for which the actions **left** and **right** are always successful.

Specifically, the available actions in each state are to move to the neighboring grid squares. From state a , there is also an **exit** action available, which results in going to the terminal state and collecting a reward of 10. Similarly, in state e , the reward for the **exit** action is 1. (Exit actions are always successful.)

10				1
a	b	c	d	e

Let the discount factor be $\gamma = 1$. Fill in the following quantities.

$$V_0(d) = \underline{\mathbf{0}}$$

Explanation. When value iteration is initialized, the V_0 value of each state is 0.

$$V_1(d) = \underline{\mathbf{0}}$$

Explanation. At the first iteration, each state knows about the V_0 values of successor states. Because $V_0(e) = 0$, the state d will not know about the exit reward at state e . Call t the terminal state. Notice the transition probabilities do not show up in our equation, because the transitions are all deterministic.

$$V_1(a) = R(a, \text{exit}, t) + V_0(t) = 10,$$

$$V_1(d) = \max(V_0(c), V_0(e)) = 0,$$

$$V_1(b) = \max(V_0(a), V_0(c)) = 0,$$

$$V_1(e) = R(e, \text{exit}, t) + V_0(t) = 1,$$

$$V_1(c) = \max(V_0(b), V_0(d)) = 0$$

$$V_2(d) = \underline{\mathbf{1}}$$

Explanation. At the second iteration, each state knows about the V_1 values of successor states. State d will now know about the exit reward of state e , and $V_1(d)$ will be updated to 1.

$$V_2(a) = 10,$$

$$V_2(d) = \max(V_1(c), V_1(e)) = 1$$

$$V_2(b) = \max(V_1(a), V_1(c)) = 10,$$

$$V_2(e) = 1,$$

$$V_2(c) = \max(V_1(b), V_1(d)) = 1$$

$$V_3(d) = \underline{\mathbf{1}}$$

Explanation.

$$V_3(a) = 10,$$

$$V_3(d) = \max(V_2(c), V_2(e)) = 1$$

$$V_3(b) = \max(V_2(a), V_2(c)) = 10,$$

$$V_3(e) = 1,$$

$$V_3(c) = \max(V_2(b), V_2(d)) = 10$$

$$V_4(d) = \underline{\mathbf{10}}$$

Explanation. At the fourth iteration, state d will see the exit reward of state a , so $V_4(d)$ will be updated to 10.

$$V_4(a) = 10$$

$$V_4(d) = \max(V_3(c), V_3(e)) = 10$$

$$V_4(b) = \max(V_3(a), V_3(c)) = 10$$

$$V_4(e) = 1$$

$$V_4(c) = \max(V_3(b), V_3(d)) = 10$$

$$V_5(d) = \underline{\mathbf{10}}$$

Explanation. Nothing will change between the fourth and fifth iteration.

Alternatively, for a simple MDP like this one, the values could also be computed directly from the meaning of $V_i(d)$, which is the expected discounted sum of rewards if acting optimally for i time steps, starting from state d . With $i = 0$ and $i = 1$, no reward can be obtained from state d , so $V_0(d) = V_1(d) = 0$. With $i = 2$ and $i = 3$, a reward of 1 can be obtained through right, exit from state d , so $V_2(d) = V_3(d) = 1$. Because it takes four steps to obtain the reward of 10 (left, left, left, exit), $V_i(d) = 10$ for $i \geq 4$.

2. (9 points) VALUE ITERATION CONVERGENCE VALUES

Consider the gridworld where left and right actions are always successful. Specifically, the available actions in each state are to move to the neighboring grid squares. From state a , there is also an exit action available, which results in going to the terminal state and collecting a reward of 10. Similarly, in state e , the reward for the exit action is 1. (Exit actions are always successful.)

10				1
----	--	--	--	---

a b c d e Let the discount factor $\gamma = 0.2$. Fill in the following quantities.

$$V^*(a) = V_\infty(a) = \underline{\quad 10 \quad}$$

Explanation. The optimal action from a is to take the exit action. Call t the terminal state. Then, $V^*(a) = R(a, \text{exit}, t) + \gamma V^*(t) = 10$

$$V^*(b) = V_\infty(b) = \underline{\quad 2 \quad}$$

Explanation. From state b , it is quite clear that you should move toward the closer, larger reward at state a . Then, $V^*(b) = R(b, \text{left}, a) + \gamma V^*(a) = 2$

$$V^*(c) = V_\infty(c) = \underline{\quad .4 \quad}$$

Explanation. From state c , you are equally close to both rewards, so the optimal action is to move toward the larger reward in state a . Then, $V^*(c) = R(c, \text{left}, b) + \gamma V^*(b) = 0.4$

$$V^*(d) = V_\infty(d) = \underline{\quad .2 \quad}$$

Explanation. It is not immediately obvious which way we should go from state d , so we must do some calculations first.

$$V^*(d) = \max(R(d, \text{left}, c) + \gamma V^*(c), R(d, \text{right}, e) + \gamma V^*(e)) = \max(.08, .2) = 0.2$$

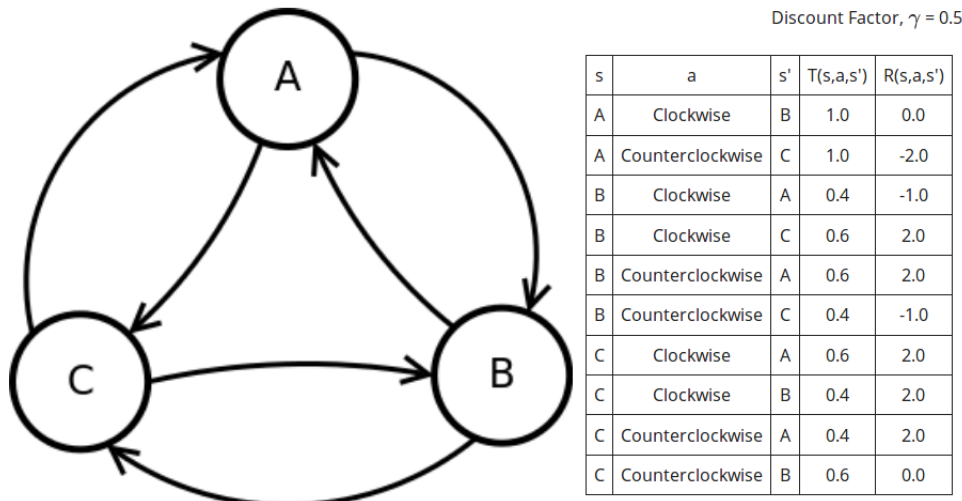
Notice that from d , we prefer the closer, smaller reward to the farther, larger reward. This is because our discount factor (0.2) is low enough for us to prefer the closer reward. If our discount factor was higher, we might prefer the farther reward instead.

$$V^*(e) = V_\infty(e) = \underline{\quad 1 \quad}$$

Explanation. In state e , we have a similar situation as state d , where we could go for the closer, smaller reward or the farther, larger reward. However, because we know the correct action from state d is right, we know that state e will also prefer the closer reward. Therefore, $V^*(e) = R(e, \text{exit}, t) + \gamma V^*(t) = 1$

3. (12 points) VALUE ITERATION: CYCLE

Consider the following transition diagram, transition function and reward function for an MDP.



Suppose after iteration k of value iteration we end up with the following values for V_k :

$V_k(A)$	$V_k(B)$	$V_k(C)$
0.400	1.400	2.160

(a) What is $V_{k+1}(A)$?

Answer. .7

Explanation.

$$V_{k+1}(A) = \max(Q_{k+1}(A, \text{clockwise}), Q_{k+1}(A, \text{counterclockwise}))$$

$$Q_{k+1}(A, \text{clockwise}) = T(A, \text{clockwise}, B)[R(A, \text{clockwise}, B) + \gamma V_k(B)] = .7$$

$$Q_{k+1}(A, \text{counterclockwise}) = T(A, \text{counterclockwise}, C)[R(A, \text{counterclockwise}, C) + \gamma V_k(C)] = -.92$$

Note that in general, we would have to sum over all possible s' for each action, but actions from A are deterministic in this problem.

(b) Now, suppose that we ran value iteration to completion and found the following value function, V^* .

$V^*(A)$	$V^*(B)$	$V^*(C)$
0.881	1.761	2.616

What is $Q^*(A, \text{clockwise})$?

Answer. .8805

Explanation.

$$Q^*(A, \text{clockwise}) = T(A, \text{clockwise}, B)[R(A, \text{clockwise}, B) + \gamma V^*(B)] = .8805$$

(c) What is $Q^*(A, \text{counterclockwise})$?

Answer. -.692

Explanation.

$$Q^*(A, \text{counterclockwise}) = T(A, \text{counterclockwise}, C)[R(A, \text{counterclockwise}, C) + \gamma V^*(C)] = -.692$$

(d) What is the optimal action from state A? ☒ **Clockwise** ☐ Counterclockwise

Explanation. The optimal action from A is the one that gives us the highest Q^* value.

4. (7 points) VALUE ITERATION PROPERTIES

Which of the following are true about value iteration? We assume the MDP has a finite number of actions and states, and that the discount factor satisfies $0 < \gamma < 1$.

- ☒ **Value iteration is guaranteed to converge.**
- ☒ **Value iteration will converge to the same vector of values (V^*) no matter what values we use to initialize V .**
- ☐ None of the above

Explanation.

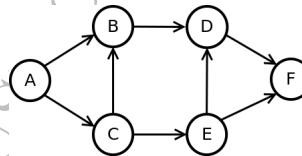
1. For discount less than 1, value iteration is guaranteed to converge.
2. At convergence, the following equation must be satisfied for all states:

$$V^*(s) = \max_a \sum_{s'} T(s, a, s') [R(s, a, s') + \gamma V^*(s')]$$

There will only be one set of values that satisfies this condition, so no matter where we start value iteration, we will always arrive at the same set of values on convergence.

5. (6 points) VALUE ITERATION CONVERGENCE

We will consider a simple MDP that has six states, A , B , C , D , E , and F . Each state has a single action, **go**. An arrow from a state x to a state y indicates that it is possible to transition from state x to next state y when **go** is taken. If there are multiple arrows leaving a state x , transitioning to each of the next states is equally likely. The state F has no outgoing arrows: once you arrive in F , you stay in F for all future times. The reward is one for all transitions, with one exception: staying in F gets a reward of zero. Assume a discount factor $= 0.5$. We assume that we initialize the value of each state to 0. (Note: you should not need to explicitly run value iteration to solve this problem.)



- (a) After how many iterations of value iteration will the value for state E have become exactly equal to the true optimum? (Enter inf if the values will never become equal to the true optimal but only converge to the true optimal.)

Answer. 2

- (b) How many iterations of value iteration will it take for the values of all states to converge to the true optimal values? (Enter inf if the values will never become equal to the true optimal but only converge to the true optimal.)

Answer. 4

Explanation. Because there are no moves from state F , we have the optimal value of F upon initializing. Since all the rewards are earned from transitions, finding the optimal value of a state amounts to finding the longest path from that state to F . For example, state D , whose longest path to F is only length 1, will find its optimal value after only one iteration.

$$V^*(D) = V_1(D) = R(D, \text{go}, F) + \gamma V^*(F) = 1$$

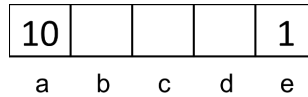
Similarly, the state A will find its optimal value after four iterations, because it will find out about its length 4 path to F after four iterations. Because A 's length 4 path

is the longest of the graph, it will take four iterations for all states to converge to their optimal values.

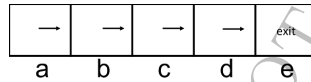
6. (10 points) POLICY EVALUATION

Consider the gridworld where Left and Right actions are always successful.

Specifically, the available actions in each state are to move to the neighboring grid squares. From state a , there is also an exit action available, which results in going to the terminal state and collecting a reward of 10. Similarly, in state e , the reward for the exit action is 1. (Exit actions are always successful.) The discount factor is $\gamma = 1$.



(a) Consider the policy π_1 shown below, and evaluate the following quantities for this policy.

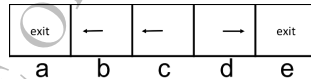


$$V^{\pi_1}(a) = \underline{\quad 1 \quad} \quad V^{\pi_1}(b) = \underline{\quad 1 \quad} \quad V^{\pi_1}(c) = \underline{\quad 1 \quad}$$

$$V^{\pi_1}(d) = \underline{\quad 1 \quad} \quad V^{\pi_1}(e) = \underline{\quad 1 \quad}$$

Explanation. Because there is no discounting and the only reward you receive is for taking an exit action, the value of a state is determined only by which exit will be taken from that state according to the policy. Because you will exit at state e from every state, the value of every state will be 1.

(b) Consider the policy π_2 shown below, and evaluate the following quantities for this policy.



$$V^{\pi_2}(a) = \underline{\quad 10 \quad} \quad V^{\pi_2}(b) = \underline{\quad 10 \quad} \quad V^{\pi_2}(c) = \underline{\quad 10 \quad}$$

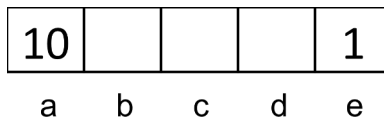
$$V^{\pi_2}(d) = \underline{\quad 1 \quad} \quad V^{\pi_2}(e) = \underline{\quad 1 \quad}$$

Explanation. From states a , b , and c , you will exit from state a , so the value from these states is 10. From states d and e , you will exit from state e , so the value of these states is 1.

7. (9 points) POLICY ITERATION

Consider the gridworld where left and right actions are always successful.

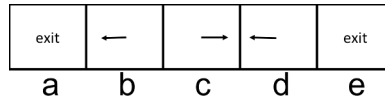
Specifically, the available actions in each state are to move to the neighboring grid squares. From state a , there is also an exit action available, which results in going to the terminal state and collecting a reward of 10. Similarly, in state e , the reward for the exit action is 1. (Exit actions are always successful.)



The discount factor is $\gamma = 0.9$.

We will execute one round of policy iteration.

Consider the policy π_i shown below, and evaluate the following quantities for this policy.



$$V^{\pi_i}(a) = \underline{\quad 10 \quad} \quad V^{\pi_i}(b) = \underline{\quad 9 \quad} \quad V^{\pi_i}(c) = \underline{\quad 0 \quad}$$

$$V^{\pi_i}(d) = \underline{\quad 0 \quad} \quad V^{\pi_i}(e) = \underline{\quad 1 \quad}$$

Explanation. From a, we take the exit action with reward 10. Thus, the value of state a is 10.

$$V^{\pi_i}(b) = \gamma V^{\pi_i}(a) = 9$$

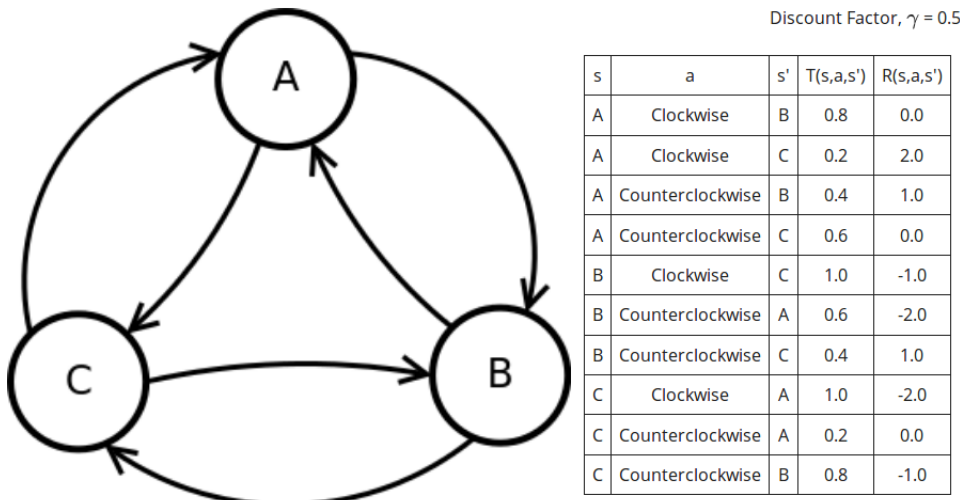
From c, we will never reach an exit state, according to the policy. Therefore, the value for this state is 0.

0, by the same reasoning as for state c.

From e, we take the exit action with reward 1.

8. (14 points) POLICY ITERATION: CYCLE

Consider the following transition diagram, transition function and reward function for an MDP.



Suppose we are doing policy evaluation, by following the policy given by the left-hand side table below. Our current estimates (at the end of some iteration of policy evaluation) of the value of states when following the current policy is given in the right-hand side table.

A	B	C
Counterclockwise	Counterclockwise	Counterclockwise

$V_k^{\pi}(A)$	$V_k^{\pi}(B)$	$V_k^{\pi}(C)$
0.000	-0.840	-1.080

(a) What is $V_{k+1}^{\pi}(A)$? *Answer.* $-.092$

Explanation. Let cc denote counterclockwise.

$$V_{k+1}^{\pi}(A) = T(A, cc, B)[R(A, cc, B) + \gamma V_k^{\pi}(B)] + T(A, cc, C)[R(A, cc, C) + \gamma V_k^{\pi}(C)]$$

$$= -0.092$$

We only take into account the cc action from A, because that is the action according to the assumed policy.

(b) Suppose that policy evaluation converges to the following value function, V_{∞}^{π} .

$V_{\infty}^{\pi}(A)$	$V_{\infty}^{\pi}(B)$	$V_{\infty}^{\pi}(C)$
-0.203	-1.114	-1.266

Now let's execute policy improvement.

What is $Q_{\infty}^{\pi}(A, \text{clockwise})$? *Answer.* -0.1722

Explanation.

$$Q_{\infty}^{\pi}(A, \text{clockwise}) = T(A, \text{clockwise}, B)[R(A, \text{clockwise}, B) + \gamma V_{\infty}^{\pi}(B)] \\ + T(A, \text{clockwise}, C)[R(A, \text{clockwise}, C) + \gamma V_{\infty}^{\pi}(C)] = -0.1722$$

(c) What is $Q_{\infty}^{\pi}(A, \text{counterclockwise})$? *Answer.* -0.2026

Explanation. Let cc denote counterclockwise.

$$Q_{\infty}^{\pi}(A, \text{cc}) = T(A, \text{cc}, B)[R(A, \text{cc}, B) + \gamma V_{\infty}^{\pi}(B)] + T(A, \text{cc}, C)[R(A, \text{cc}, C) + \gamma V_{\infty}^{\pi}(C)] \\ = -0.2026$$

(d) What is the updated action for state A ? ☒ **Clockwise** ☐ Counterclockwise

Explanation. The updated action for state A will be the action that results in the higher Q_{∞}^{π} .

9. (7 points) WRONG DISCOUNT FACTOR

Bob notices value iteration converges more quickly with smaller γ and rather than using the true discount factor γ , he decides to use a discount factor of $\alpha\gamma$ with $0 < \alpha < 1$ when running value iteration. Mark each of the following that are guaranteed to be true:

- ☐ While Bob will not find the optimal value function, he could simply rescale the values he finds by $\frac{1-\gamma}{1-\alpha}$ to find the optimal value function.
- ☒ **If the MDP's transition model is deterministic and the MDP has zero rewards everywhere, except for a single transition at the goal with a positive reward, then Bob will still find the optimal policy.**
- ☐ If the MDP's transition model is deterministic, then Bob will still find the optimal policy.
- ☒ **Bob's policy will tend to more heavily favor short-term rewards over long-term rewards compared to the optimal policy.**
- ☐ None of the above.

Explanation.

Option 1: False. If Bob simply rescales all the values, the policy that he finds will be exactly the same.

Option 2: True. When the transitions are deterministic and there is a single reward at the goal, the optimal policy will be the shortest path to the goal both for discount factor γ and factor $\alpha\gamma$. Therefore, the optimal policy will not change.

Option 3: False. Consider the MDP from Question 8 for $\alpha = 0.1$ and $\gamma = 0.9$. The optimal policy in this MDP is to go left at state d . However, Bob's policy will tell you to go right at state d .

Option 4: True. Bob's policy is the optimal policy from running value iteration with a lower discount factor. The lower discount factor favors short term rewards.

10. (10 points) MDP PROPERTIES

(a) Which of the following statements are true for an MDP?

- ☐ If the only difference between two MDPs is the value of the discount factor then they must have the same optimal policy.
- ✓ ***For an infinite horizon MDP with a finite number of states and actions and with a discount factor γ that satisfies $0 < \gamma < 1$, value iteration is guaranteed to converge.***
- ☐ When running value iteration, if the policy (the greedy policy with respect to the values) has converged, the values must have converged as well.
- ☐ None of the above

Explanation.

Option 1: False. Consider the MDP for Question 8 with two discount factors $\gamma_1 = 0.9$ and $\gamma_2 = 0.2$. For γ_1 , the optimal policy will be to go left at state d . For γ_2 , the optimal policy will be to go right at state d .

Option 2: True. With a discount factor less than 1, value iteration is guaranteed to converge, as shown in lecture.

Option 3: False. Consider an MDP with three states A , B , and C . There are deterministic transitions from A to B , B to C , and C to A . The reward for each transition is 1. For this MDP, we will know from the beginning what the policy is, because there is only one action from each state, and so there is only one possible policy. However, value iteration only converge much later.

(b) Which of the following statements are true for an MDP?

- ✓ ***If one is using value iteration and the values have converged, the policy must have converged as well.***
- ☐ Expectimax will generally run in the same amount of time as value iteration on a given MDP.
- ✓ ***For an infinite horizon MDP with a finite number of states and actions and with a discount factor γ that satisfies $0 < \gamma < 1$, policy iteration is guaranteed to converge.***
- ☐ None of the above

Explanation.

Option 1: True. If value iteration has converged, the values of each state will not change anymore. This means that the policy from iteration to iteration will not change either.

Option 2: False. Let's consider the cost of computing the value of a single state s , where a horizon H is needed for the values to converge. With expectimax, we will construct a tree with a maximum branching factor of AS , for every (a, s') pair, and a depth of H , for the horizon, so the complexity of expectimax will be $(AS)^H$. Now consider value iteration. For every step of value iteration, we will look at AS^2 values, for every (s, a, s') tuple. There will be a total of H iterations, so complexity of value iteration is AS^2H . Thus, expectimax will generally run much slower than value iteration.

Option 3: True. For policy iteration, we are guaranteed to find a better policy every iteration until we converge. Because we improve the policy every iteration, and there are a finite number of policies for a given MDP, we are guaranteed to eventually converge.

11. (7 points) John, James, Alvin and Michael all get to act in an MDP $(S, A, T, \gamma, R, s_0)$.

John runs value iteration until he finds V^* which satisfies

$$\forall s (s \in S \rightarrow V^*(s) = \max_{a \in A} \sum_{s'} T(s, a, s') (R(s, a, s') + \gamma V^*(s')))$$

and acts according to

$$\pi_{\text{John}} = \arg \max_{a \in A} \sum_{s'} T(s, a, s') (R(s, a, s') + \gamma V^*(s')).$$

James acts according to an arbitrary policy π_{James} .

Alvin takes James's policy π_{James} and runs one round of policy iteration to find his policy π_{Alvin} .

Michael takes John's policy and runs one round of policy iteration to find his policy π_{Michael} .

Note: One round of policy iteration = performing policy evaluation followed by performing policy improvement.

Mark all of the following that are guaranteed to be true:

- ☐ It is guaranteed that $\forall s \in S : V^{\pi_{\text{James}}}(s) \geq V^{\pi_{\text{Alvin}}}(s)$
- ☒ ***It is guaranteed that*** $\forall s \in S : V^{\pi_{\text{Michael}}}(s) \geq V^{\pi_{\text{Alvin}}}(s)$
- ☐ It is guaranteed that $\forall s \in S : V^{\pi_{\text{Michael}}}(s) > V^{\pi_{\text{John}}}(s)$
- ☐ It is guaranteed that $\forall s \in S : V^{\pi_{\text{James}}}(s) > V^{\pi_{\text{John}}}(s)$
- ☐ None of the above.

Explanation.

Option 1: False. Actually, the reverse is true. In policy iteration, we are guaranteed to improve every step until convergence.

Option 2: True. Because John's policy is optimal, running policy iteration on it will return the same optimal policy. Therefore, Michael's policy is optimal, while Alvin's may not be optimal.

Option 3: False. John and Michael have the same policy.

Option 4: False. John's policy is optimal, so there cannot be a policy that is better than it.