

---

## CSE 291d. Assignment 7

**Out:** Tue Mar 6

**Due:** Tue Mar 13 (in class, no extensions)

---

### 7.1 Effective horizon time

Consider a Markov decision process (MDP) whose rewards  $r_t \in [0, 1]$  are bounded between zero and one. Let  $h = (1 - \gamma)^{-1}$  define an *effective* horizon time in terms of the discount factor  $0 \leq \gamma \leq 1$ . Consider the approximation to the (infinite horizon) discounted return,

$$r_0 + \gamma r_1 + \gamma^2 r_2 + \gamma^3 r_3 + \gamma^4 r_4 + \dots,$$

obtained by neglecting rewards from some time  $t$  and beyond. Recalling that  $\log \gamma \leq \gamma - 1$ , show that the error from such an approximation decays exponentially as:

$$\sum_{k \geq t} \gamma^k r_k \leq h e^{-t/h}.$$

Thus, we can view MDPs with discounted returns as similar to MDPs with finite horizons, where the finite horizon  $h = (1 - \gamma)^{-1}$  grows as  $\gamma \rightarrow 1$ . This is a useful intuition for proving the convergence of many algorithms in reinforcement learning.

---

### 7.2 Convergence of iterative policy evaluation

Consider an MDP with transition matrices  $P(s'|s, a)$  and reward function  $R(s)$ . In class, we showed that the state value function  $V^\pi(s)$  for a fixed policy  $\pi$  satisfies the Bellman equation:

$$V^\pi(s) = R(s) + \gamma \sum_{s'} P(s'|s, \pi(s)) V^\pi(s').$$

For  $\gamma < 1$ , one method to solve this set of linear equations is by iteration. Initialize the state value function by  $V_0(s) = 0$  for all states  $s$ . The update rule at the  $k^{\text{th}}$  iteration is given by:

$$V_{k+1}^\pi(s) \leftarrow R(s) + \gamma \sum_{s'} P(s'|s, \pi(s)) V_k^\pi(s').$$

Use a contraction mapping to derive an upper bound on the error

$$\Delta_k = \max_s |V_k(s) - V^\pi(s)|$$

after  $k$  iterations of the update rule. Your result should show that the error  $\Delta_k$  decays exponentially fast in the number of iterations,  $k$ , and hence that  $\lim_{k \rightarrow \infty} V_k(s) = V^\pi(s)$  for all states  $s$ .

---

---

### 7.3 Policy versus value iteration

Consider an MDP with  $n=6$  states, binary actions, and discount factor  $\gamma = 0.92$ . Download the ASCII files on the course web site that store the transition matrices and reward function for this MDP. The transition matrices are stored such that the rows sum to one. In this problem, you will compare the two classical algorithms derived in class for computing optimal policies in MDPs.

- (a) Compute the optimal policy  $\pi^*(s)$  for this MDP using the method of policy iteration, starting from the initial guess  $\pi_0(s) = 1$  for all states  $s$ . For the step of policy evaluation, you may use either the iterative method from problem 7.2 (allowing it to converge to a high degree of accuracy) or a black-box routine for solving linear systems of equations. Turn in your source code (in the language of your choice), as well as a print-out of the optimal policy  $\pi^*(s)$  and its state value function  $V^*(s)$ . Also note the number of iterations required to converge to the optimal policy.
- (b) Compute the optimal state value function  $V^*(s)$  using the method of value iteration. Turn in your source code, as well as a print-out of  $V^*(s)$ . Your result in part (b) should agree with your result in part (a); you can use this to check your work.

---

### 7.4 Stochastic approximation

In this problem, you will analyze an incremental update rule for estimating the mean  $\mu = E[X]$  of a random variable  $X$  from samples  $\{x_1, x_2, x_3, \dots\}$ . Consider the incremental update rule:

$$\mu_k \leftarrow \mu_{k-1} + \alpha_k(x_k - \mu_{k-1}),$$

with the initial condition  $\mu_0 = 0$  and step sizes  $\alpha_k = 1/k$ .

- (a) Show that the step sizes  $\alpha_k = 1/k$  obey the conditions (i)  $\sum_{k=1}^{\infty} \alpha_k = \infty$  and (ii)  $\sum_{k=1}^{\infty} \alpha_k^2 < \infty$ , thus guaranteeing that the update rule converges to the true mean. (You are not required to prove convergence.)
- (b) Show that for this particular choice of step sizes, the incremental update rule yields the same result as the sample average:  $\mu_k = (1/k)(x_1 + x_2 + \dots + x_k)$ .

---

## 7.5 Value function for a random walk

Consider a Markov decision process (MDP) with discrete states  $s \in \{0, 1, 2, \dots, \infty\}$  and rewards  $R(s) = s$  that grow linearly as a function of the state. Also, consider a policy  $\pi$  whose action in each state either leaves the state unchanged or with equal probability yields a transition to the next state:

$$P(s'|s, \pi(s)) = \begin{cases} \frac{1}{2} & \text{if } s' = s \\ \frac{1}{2} & \text{if } s' = s+1 \\ 0 & \text{otherwise} \end{cases}$$

Intuitively, this policy can be viewed as a right-drifting random walk. As usual, the value function for this policy,  $V^\pi(s) = \mathbb{E} [\sum_{t=0}^{\infty} \gamma^t R(s_t) | s_0 = s]$ , is defined as the expected sum of discounted rewards starting from state  $s$ .

- (a) Assume that the value function  $V^\pi(s)$  satisfies a Bellman equation analogous to the one in MDPs with finite state spaces. Write down the Bellman equation satisfied by  $V^\pi(s)$ .
  - (b) Show that one possible solution to the Bellman equation in part (a) is given by the linear form  $V^\pi(s) = as + b$ , where  $a$  and  $b$  are coefficients that you should express in terms of the discount factor  $\gamma$ . (*Hint*: substitute this solution into both sides of the Bellman equation, and solve for  $a$  and  $b$  by requiring that both sides are equal for all values of  $s$ .)
  - (c) *Extra credit*: justify that the value function  $V^\pi(s)$  has this linear form. (In other words, rule out other possible solutions to the Bellman equation for this policy.)
-

---

## 7.6 Multiplicative rewards

Consider a Markov decision process (MDP) with discrete states  $s \in \mathcal{S}$ , discrete actions  $a \in \mathcal{A}$ , deterministic rewards  $R(s)$ , and transition probabilities  $P(s'|s, a)$ . Let  $W^\pi(s)$  denote the expected *product* of rewards over a *finite* horizon of  $T$  steps, starting from state  $s$  and following policy  $\pi$ ; more precisely,

$$W^\pi(s) = \mathbb{E}^\pi \left[ \prod_{t=0}^{T-1} R(s_t) \mid s_0 = s \right],$$

where the expectation on the right hand side is over all  $T$ -step paths through state space, as weighted by their probabilities under policy  $\pi$ . More explicitly, we can write this expected product of rewards as:

$$W^\pi(s) = R(s) \sum_S \prod_{t=1}^T \left[ P(s_t | s_{t-1}, \pi(s_{t-1})) R(s_t) \right].$$

where the sum on the right hand side is over all paths  $S = (s_1, s_2, \dots, s_T)$ .

### (a) One-step and two-step horizons

Evaluate the expected product of rewards for the special cases of one and two-step horizons; that is, compute  $W^\pi(s)$  for  $T=1$  and  $T=2$ , explicitly writing out the sums over paths  $S$  in each case.

### (b) Dynamic programming

Consider how to compute the expected product of rewards  $W^\pi(s)$  over a finite horizon of  $T \geq 2$  steps. Specifically, show how to fill in a matrix  $\{\alpha_{it}\}$  for states  $i \in \mathcal{S}$  and times  $t \geq 0$ , such that the expected product of rewards after  $T$  steps is related in a simple way to the elements of  $\alpha_{iT}$ . The computation time of your algorithm should scale as  $O(n^2T)$ , where  $n$  is the number of states.

$\alpha_{i0} = ?$
$\alpha_{i,t+1} = ?$
$W^\pi(s) = ?$

Naturally, your algorithm should reproduce your solution in part (a) for the cases  $T=1$  and  $T=2$ .

---

---

## 7.7 Explore or exploit

Consider a Markov decision process (MDP) with discrete states  $s \in \{1, 2, \dots, n\}$ , binary actions  $a \in \{0, 1\}$ , deterministic rewards  $R(s)$ , and transition probabilities:

$$P(s'|s, a=0) = \frac{1}{n} \text{ for all } s',$$
$$P(s'|s, a=1) = \begin{cases} 1 & \text{for } s=s', \\ 0 & \text{otherwise.} \end{cases}$$

In this MDP, the “explore” action  $a=0$  leads to uniformly random exploration of the state space, while the “exploit” action  $a=1$  leaves the agent in its current state.

The simple structure of this MDP allows one to calculate its value functions, as well as the form of its optimal policy. This problem guides you through those calculations.

### (a) Local reward-mining

As usual, the state-value function  $V^\pi(s)$  is defined as the expected sum of discounted rewards starting from state  $s$  at time  $t=0$  and taking action  $\pi(s_t)$  at time  $t$ :

$$V^\pi(s) = \mathbb{E}^\pi \left[ \sum_{t=0}^{\infty} \gamma^t R(s_t) \mid s_0 = s \right].$$

Consider the subset of states  $\mathcal{S}_1 = \{s \mid \pi(s) = 1\}$  in which the agent chooses the “exploit” action. Compute  $V^\pi(s)$  for states  $s \in \mathcal{S}_1$  by evaluating the expected sum of discounted rewards.

### (b) Bellman equation for exploration

Consider the subset of states  $\mathcal{S}_0 = \{s \mid \pi(s) = 0\}$  in which the agent chooses the “explore” action. Write down the Bellman equation satisfied by the value function  $V^\pi(s)$  for states  $s \in \mathcal{S}_0$ . Your answer should substitute in the appropriate transition probabilities from these states.

### (c) State-averaged value function

Assume that the reward function, averaged over the state space, has zero mean:  $\frac{1}{n} \sum_s R(s) = 0$ .

Let  $\mu = \frac{1}{n} |\mathcal{S}_0|$  denote the fraction of states in which the agent chooses to explore.

Let  $r = \frac{1}{n} \sum_{s \in \mathcal{S}_1} R(s)$  denote the average reward in the remaining states.

Let  $v = \frac{1}{n} \sum_s V^\pi(s)$  denote the expected return if the initial state is chosen uniformly at random.

Combining your results from parts (a-b), show that the state-averaged expected return  $v$  is given by:

$$v = \frac{\gamma r}{(1-\gamma)(1-\gamma\mu)}$$

(d) **Value of exploration**

Using the results from parts (b) and (c), express the state-value function  $V^\pi(s)$  for states  $s \in \mathcal{S}_0$  in terms of the reward function  $R(s)$ , the discount factor  $\gamma$ , and the quantities  $r$  and  $\mu$ .

(e) **Optimal policy**

The form of this MDP suggests the following intuitive strategy: in states with low rewards, opt for random exploration at the next time step, while in states with high rewards, opt to remain in place, exploiting the current reward for all future time. In this problem, you will confirm this intuition.

Starting from the observation that an optimal policy is its own greedy policy, infer that the optimal policy  $\pi^*$  in this MDP takes the simple form:

$$\pi^*(s) = \begin{cases} 1 & \text{if } V^*(s) > \theta \\ 0 & \text{if } V^*(s) \leq \theta \end{cases}$$

where the threshold  $\theta$  is a constant, independent of the state  $s$ . As part of your answer, express the threshold  $\theta$  in terms of the state-averaged optimal value function  $v^* = \frac{1}{n} \sum_s V^*(s)$ .