EL2805, Exercise 6 (Recap. for exam)

Robert Mattila

Check the "Today's central concepts" from the previous sessions. My notes are available at:

rmattila.github.io.

Read through the lecture slides (these you can bring to exam, so familiarize yourself with them).

Solve the exercises we've solved here again on your own. Also try to solve the other exercises (solutions are available to all of them).

Good luck on the exam! :)

Ex 6.5 d

Prove that

$$\mathbb{E}_{\pi_\theta}\{\nabla \log \pi_\theta(s_t, a_t)\} = 0,$$

for $\mathcal{A}$ finite.

Solution:

We have that

$$\mathbb{E}_{\pi_\theta}\{\nabla \log \pi_\theta(s_t, a_t)\} =$$

$$\sum_{s\in\mathcal{S}}' \sum_{a\in\mathcal{A}}' \Pr\{s_t = s, a_t = a\} \nabla \log \pi_\theta(s,a) =$$

$$\sum_{s\in\mathcal{S}}' \sum_{a\in\mathcal{A}}' \Pr\{a_t = a \mid s_t = s\} \Pr\{s_t = s\} \nabla \log \pi_\theta(s,a) =$$

$$\sum_{s\in\mathcal{S}}' \Pr\{s_t = s\} \sum_{a\in\mathcal{A}}' \underbrace{\Pr\{a_t = a \mid s_t = s\}}_{= \pi_\theta(s,a)} \nabla \log \pi_\theta(s,a) =$$

$$\sum_{s\in\mathcal{S}}' \Pr\{s_t = s\} \sum_{a\in\mathcal{A}}' \pi_\theta(s,a) \nabla \log \pi_\theta(s,a) =$$

$$\sum_{s\in\mathcal{S}}' \Pr\{s_t = s\} \sum_{a\in\mathcal{A}}' \pi_\theta(s,a) \frac{\nabla \pi_\theta(s,a)}{\pi_\theta(s,a)} =$$

$$\sum_{s\in\mathcal{S}}' \Pr\{s_t = s\} \sum_{a\in\mathcal{A}}' \nabla \pi_\theta(s,a) =$$

$$\sum_{s\in\mathcal{S}}' \Pr\{s_t = s\} \nabla \underbrace{\sum_{a\in\mathcal{A}}' \pi_\theta(s,a)}_{=1} =$$

$\Big\{$ Recall that $\pi_\theta(s,a) = \Pr\{a_t = a \mid s_t = s\}$. "The probability that we take __some__ action is 1." $\Big\}$ =

$$\sum_{s\in\mathcal{S}}' \Pr\{s_t = s\} \underbrace{\nabla 1}_{=0} = 0.$$

Ex 6.4 | Consider a system with $\mathcal{S}=\{A,B,C\}$ and $\mathcal{A}=\{\varphi,\beta,\gamma\}$. We observe a trajectory

$$(\ldots,\ \underbrace{A,\ \varphi,\ 200}_{s_{t=5734},\ a_{t=5734},\ r_{t=5734}},\ B,\ \varphi,\ 150,\ A,\ \gamma,\ 140,\ C,\ \gamma,\ \ldots)$$

and

$$q^{(5734)}(s,a) = \begin{array}{c} \\ A \\ B \\ C \end{array} \begin{array}{c} \begin{array}{ccc} \varphi & \beta & \gamma \end{array} \\ \begin{bmatrix} 100 & 200 & 80 \\ 120 & 50 & 140 \\ 90 & 200 & 100 \end{bmatrix} \end{array} .$$

a, Provide updated q-tables for Q-learning. $\alpha = \frac{1}{2}$ $\lambda = 0.1$

b, What is the greedy policy?

c, Provide updated q-tables for SARSA. $\alpha = \frac{1}{2}$ $\lambda = 0.1$

d, What is the greedy policy?

e, With ε-greedy action selection, what q-tables will SARSA and Q-learning converge to? How can we obtain the optimal policy with SARSA?

Solution:

a, Recall the Q-learning update:

$$q(s,a) \leftarrow q(s,a) + \alpha \left[r(s,a) + \gamma \max_{a'} q(s',a') - q(s,a) \right].$$

Then:

$$q^{(5734)}: \begin{array}{c|ccc|} & \varphi & \beta & \gamma \\ \hline A & 100 & 200 & 80 \\ B & 120 & 50 & 140 \\ C & 90 & 200 & 100 \\ \hline \end{array}$$

$s = A$
$a = \varphi$
$r = 200$
$s' = B$

- $q(s,a) = q(A,\alpha) \leftarrow$

$$q(s,a) + \alpha \left[r(s,a) + \gamma \max q(s',a') - q(s,a) \right] =$$

$$q(A,\varphi) + \alpha \left[r(A,\varphi) + \underbrace{\gamma \max_{a'} q(B,a')}_{140 \text{ for } a' = \gamma} - q(A,\varphi) \right] =$$

$$100 + \frac{1}{2} \left[200 + 0.1 \times 140 - 100 \right] =$$

$$100 + \frac{1}{2} \left[100 + 14 \right] = 157$$

$$q^{(5735)}: \begin{array}{c|ccc|} & \varphi & \beta & \gamma \\ \hline A & 157 & 200 & 80 \\ B & 120 & 50 & 140 \\ C & 90 & 200 & 100 \\ \hline \end{array}$$

$s = B$
$a = \varphi$
$r = 150$
$s' = A$

- $q(B,\varphi) \leftarrow$

$$q(B,\varphi) + \alpha \left[r(B,\varphi) + \underbrace{\gamma \max_{a'} q(A,a')}_{200 \text{ for } a' = \beta} - q(B,\varphi) \right] =$$

$$120 + \frac{1}{2} \left[150 + 0.1 \times 200 - 120 \right] =$$

$$120 + \frac{1}{2} \left[150 - 100 \right] = 145$$

$$q^{(5736)}: \begin{array}{c|ccc|} & \varphi & \beta & \gamma \\ \hline A & 157 & 200 & 80 \\ B & 145 & 50 & 140 \\ C & 90 & 200 & 100 \\ \hline \end{array}$$

$s = A$
$a = \gamma$
$r = 140$
$s' = C$

- $q(A,\gamma) \leftarrow$

$$q(A,\gamma) + \alpha \left[r(A,\gamma) + \underbrace{\gamma \max_{a'} q(C,a')}_{200 \text{ for } a' = \beta} - q(A,\gamma) \right] =$$

$$80 + \frac{1}{2} \left[140 + 0.1 \times 200 - 80 \right] =$$

$$80 + \frac{1}{2} \left[140 - 60 \right] = 120$$

$$q^{(5737)}: \quad \begin{array}{c|ccc} & \varphi & \beta & \gamma \\ \hline A & 157 & 200 & 120 \\ B & 145 & 50 & 140 \\ C & 90 & 200 & 100 \end{array}$$

And then we don't have more data.

b, This corresponds to a greedy policy:

- $\pi^{(5737)}(s=A) = \arg\max_a q^{(5737)}(A, a) = \beta$
- $\pi^{(5737)}(s=B) = \varphi$
- $\pi^{(5737)}(s=C) = \beta$

↳ Recall the SARSA update:

$$q(s,a) \leftarrow q(s,a) + \alpha \left[r(s,a) + \lambda q(s',a') - q(s,a) \right]$$

Then:

$q^{(5734)}$:

	φ	β	γ
A	100	200	80
B	120	50	140
C	90	200	100

$s = A$
$a = \varphi$
$r = 200$
$s' = B$
$a' = \varphi$

- $q(s,a) = q(A,\varphi) \leftarrow$

$$q(s,a) + \alpha \left[r(s,a) + \lambda q(s',a') - q(s,a) \right] =$$

$$q(A,\varphi) + \alpha \left[r(A,\varphi) + \lambda q(B,\varphi) - q(A,\varphi) \right] =$$

$$100 + \frac{1}{2}\left[200 + 0.1 \times 120 - 100 \right] =$$

$$100 + \frac{1}{2}\left[100 + 12 \right] = 156$$

$q^{(5735)}$:

	φ	β	γ
A	156	200	80
B	120	50	140
C	90	200	100

$s = B$
$a = \varphi$
$r = 150$
$s' = A$
$a' = \gamma$

- $q(B,\varphi) \leftarrow$

$$q(B,\varphi) + \alpha \left[r(B,\varphi) + \lambda q(A,\gamma) - q(B,\varphi) \right] =$$

$$120 + \frac{1}{2}\left[150 + 0.1 \cdot 80 - 120 \right] =$$

$$120 + \frac{1}{2}\left[30 + 8 \right] = 139$$

$q^{(5736)}$:

	φ	β	γ
A	156	200	80
B	139	50	140
C	90	200	100

$s = A$
$a = \gamma$
$r = 140$
$s' = C$
$a' = \gamma$

- $q(A,\gamma) \leftarrow$

$$q(A,\gamma) + \alpha \left[r(A,\gamma) + \lambda q(C,\gamma) - q(A,\gamma) \right] =$$

$$80 + \frac{1}{2}\left[140 + 0.1 \times 100 - 80 \right] =$$

$$80 + \frac{1}{2}\left[60 + 10 \right] = 115$$

$$q^{(5737)}: \begin{array}{c|ccc} & \varphi & \beta & \gamma \\ \hline A & 156 & 200 & 115 \\ B & 139 & 50 & 140 \\ C & 90 & 200 & 100 \end{array}$$

And then there is no more data.

d, This corresponds to a greedy policy:

$$\pi^{(5737)}(s=A) = \beta$$
$$\pi^{(5737)}(s=B) = \gamma$$
$$\pi^{(5737)}(s=C) = \beta$$

e, with ε-greedy action selection, SARSA will converge to the optimal policy taking into account that we do random exploration with probability ε. By letting $\varepsilon \to 0$, we will tend to the optimal policy (which is what Q-learning is converging to).

Ex 6.3 An order is received w.p. p at each time step. One can choose to process all orders (setup cost $K > 0$), or wait (costs $c > 0$ per order on hold). The maximum number of unfilled orders is n. There is a discount factor λ. Characterize an optimal processing policy.

Solution:

State-space: $S = \{0, 1, \dots, n\}$

\# of unfilled orders when starting the period

Actions:

$A_n = \{P\}$ — process all orders

$A_i = \{W, P\}$ — wait, process all orders — $i = 0, 1, \dots, n-1$

Rewards:

In this case, we deal with costs:

- $r(i, P) = -K$
- $r(i, W) = -ci$ — $i = 0, 1, \dots, n-1$
- $r(n, P) = -K$

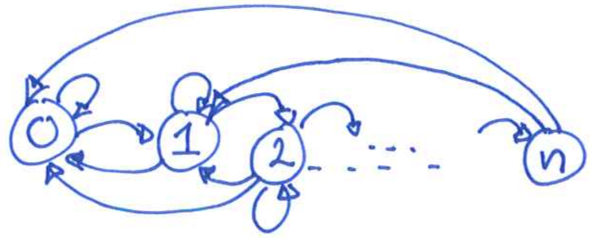
Time-horizon and objective:

∞-horizon, discounted:

$$\mathbb{E}\left\{ \sum_{t=0}^{\infty} \lambda^t r(s_t, a_t) \right\}$$

(Remark: We can interpret $1-\lambda$ as the probability of "going out of business" (e.g., bankruptcy).)

Transitions:



- $p(i \mid i, W) = 1-p$

  wait, and no new order

- $p(i+1 \mid i, W) = p$ $\qquad\qquad i = 0, 1, \ldots, n-1$

  wait, and a new order

- $p(0 \mid i, P) = 1-p$

  process everything, and no new order

- $p(1 \mid i, P) = p$

  process everything, and receive a new order

- $p(0 \mid n, P) = 1-p$
- $p(1 \mid n, P) = p$

Bellman equation:

$$u^*(s) = \max_{a \in A_s} \left\{ r(s,a) + \lambda \sum_{s' \in S} p(s' \mid s, a)\, u^*(s') \right\}$$

If $s = n$:

$$u^*(n) = \max_{\underbrace{a \in A_n}_{=\{P\}}} \left\{ \underbrace{r(n,a)}_{=-K} + \lambda \sum_{s' \in S} p(s' \mid n, a)\, u^*(s') \right\}$$

$$= -K + \lambda \left[(1-p)\, u^*(0) + p\, u^*(1) \right] \qquad (*)$$

If $s = i < n$:

$$u^*(i) = \max_{\underbrace{a \in A_i}_{=\{P,W\}}} \left\{ r(i,a) + \lambda \sum_{s' \in S} p(s'|i,a)\, u^*(s') \right\}$$

Evaluate the actions separately:

If $a = P$:

$$\underbrace{r(i,P)}_{=-K} + \lambda \sum_{s' \in S} p(s'|i,P)\, u^*(s') =$$

$$-K + \lambda \left[(1-p)u^*(0) + pu^*(1) \right] \overset{\text{def.}}{=} \gamma$$

If $a = W$:

$$\underbrace{r(i,W)}_{=-ci} + \lambda \sum_{s' \in S} p(s'|i,W)\, u^*(s') =$$

$$-ci + \lambda \left[(1-p)u^*(i) + pu^*(i+1) \right]$$

So that:

$$u^*(i) = \max \Big\{ \underbrace{\gamma}_{P},\ \underbrace{-ci + \lambda \left[(1-p)u^*(i) + pu^*(i+1) \right]}_{W} \Big\}. \qquad (**)$$

It is reasonable to assume that the optimal policy is of threshold type. Can we prove that it is?

A threshold policy is such that

$$a^*(i) = P \implies a^*(i+1) = P.$$

This is equivalent to

$$\gamma > -ci + \lambda\left[(1-p)u^*(i) + pu^*(i+1)\right] \implies$$
$$\gamma > -c(i+1) + \lambda\left[(1-p)u^*(i+1) + pu^*(i+2)\right]$$

Note that this holds if

$$u^*(i) \geqslant u^*(i+1), \qquad (\square)$$

since $c > 0$.

Because then:

$$\gamma > -ci + \lambda\left[(1-p)u^*(i) + pu^*(i+1)\right] \quad \text{since } c>0$$
$$> -c(i+1) + \lambda\left[(1-p)\underbrace{u^*(i)}_{\geqslant u^*(i+1)} + p\underbrace{u^*(i+1)}_{\geqslant u^*(i+2)}\right]$$
$$\geqslant -c(i+1) + \lambda\left[(1-p)u^*(i+1) + pu^*(i+2)\right]$$

Can we prove that $(\square)$ holds?

We will use value iteration to prove $(\square)$.

Base case:

Let $u_0(s) = 0$ for all $s \in \mathcal{S}'$.

Then $u_0(i) \geq u_0(i+1)$ holds trivially.

Induction:

Assume that $u_k(i) \geq u_k(i+1)$.

(We will show that then $u_{k+1}(i) \geq u_{k+1}(i+1)$ holds.)

If $s = i+1 < n$: (see p. 15 for an alternative proof.)

It is clear that the following holds:

$$\underbrace{-c(i+1)}_{\substack{\leq -ci \\ \text{since} \\ c>0}} + \lambda[(1-p)\underbrace{u_k(i+1)}_{\substack{\leq u_k(i) \\ \text{by assumption}}} + p\underbrace{u_k(i+2)}_{\substack{\leq u_k(i+1) \\ \text{by} \\ \text{assumption}}}] \leq -ci + \lambda[(1-p)u_k(i) + pu_k(i+1)] \qquad (\Delta)$$

Let $F_k(x)$ be defined as

$$F_k(x) = \max\{-K + \lambda[(1-p)u_k(0) + pu_k(1)], x\}.$$

This function is increasing in x.

One iteration of value iteration is (compare the computations that led to $(**)$):

$$u_{k+1}(i+1) = \max\{-K + \lambda[(1-p)u_k(0) + pu_k(1)], \\ -c(i+1) + \lambda[(1-p)u_k(i+1) + pu_k(i+2)]\}$$

$$= F_k(-c(i+1) + \lambda[(1-p)u_k(i+1) + pu_k(i+2)])$$

$$\leq F_k\left(-ci + \lambda\left[(1-p)u_k(i) + pu_k(i+1)\right)\right)$$

by (Δ) and that $F_k(x) \uparrow x$

$$= \max\left\{-K + \lambda\left[(1-p)u_k(0) + pu_k(1)\right],\right.$$
$$\left. -ci + \lambda\left[(1-p)u_k(i) + pu_k(i+1)\right]\right\}$$

$$= u_{k+1}(i).$$

by definition of one iteration of value iteration, compare (**).

<u>If $s = n$:</u>

Again, one iteration of value iteration (compare computation of (*)) yields:

$$u_{k+1}(n) = -K + \lambda\left[(1-p)u_k(0) + pu_k(1)\right]$$

obviously smaller than the max of itself and something else

$$\leq \max\left\{-K + \lambda\left[(1-p)u_k(0) + pu_k(1)\right],\right.$$
$$\left. -c(n-1) + \lambda\left[(1-p)u_k(n-1) + pu_k(n)\right]\right\}$$

compare (**)

$$= u_{k+1}(n-1).$$

In summary, we have now shown that

$$u_k(i) \geq u_k(i+1) \Rightarrow u_{k+1}(i) \geq u_{k+1}(i+1)$$

for all $i = 0, \ldots, n-1$.

Taking the limit:

We have shown that

- $u_0(i) \geq u_0(i+1)$
- $u_k(i) \geq u_k(i+1) \Rightarrow u_{k+1}(i) \geq u_{k+1}(i+1)$.

Since it is well known that

$$u^*(s) = \lim_{k\to\infty} u_k(s),$$

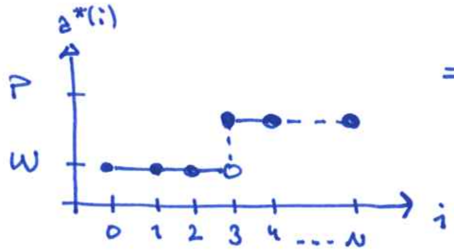
when u_k is computed using value iteration for discounted problems, we conclude that

$$u^*(i) \geq u^*(i+1),$$

which is $(\square)$.

See, e.g., p. 84 in Vol. 2 of Bertsekas' "Dynamic Program. and Optimal Control".

In summary, this proves that $a^*(i) = P \Rightarrow a^*(i+1) = P$



$\Rightarrow$ To find the optimal policy, we just need to check which of the $N+1$ possible threshold policies is the best.

(This entails performing the policy evaluation step of policy iteration $N+1$ times.)

If $s = i+1 < n$: (Alternative)

One iteration of value iteration is:

$$u_{k+1}(i+1) = \max_{a \in \mathcal{A}_{i+1}} \left\{ r(i+1,a) + \sum_{s' \in S}' p(s' | i+1, a) u_k(s') \right\}$$

$$= \left\{ \text{compare calculations for } (**) \right\}$$

$$= \max \Big\{ \underbrace{\overset{P}{-K} + \lambda \left[(1-p) u_k(0) + p u_k(1) \right]}_{\overset{\text{def.}}{=} \gamma_k \text{, constant in } i},$$

$$\overset{W}{-c(i+1)} + \lambda \left[(1-p) u_k(i+1) + p u_k(i+2) \right] \Big\}$$

$$= \max \left\{ \overset{P}{\gamma_k}, \overset{W}{-c(i+1)} + \lambda \left[(1-p) u_k(i+1) + p u_k(i+2) \right] \right\}$$

$$= \left\{ \underline{\text{Note:}} \ \max\{\gamma_k, x\} \text{ is increasing in } x \right\}$$

$$\overset{(\Delta)}{\leq} \max \left\{ \overset{P}{\gamma_k}, \overset{W}{-ci} + \lambda \left[(1-p) u_k(i) + p u_k(i+1) \right] \right\}$$

$$= \max \Big\{ \overset{P}{-K} + \lambda \left[(1-p) u_k(0) + p u_k(1) \right],$$

$$\overset{W}{-ci} + \lambda \left[(1-p) u_k(i) + p u_k(i+1) \right] \Big\}$$

$$= u_{k+1}(i).$$