model-free solution of finite MDP / SOCP

goal: derive algorithms to solve MDP/SOCP when model is "unknown"

refs: *Neuro-Dynamic Programming*

Dimitri P. Bertsekas and John N. Tsitsiklis

Chapter 5

- given finite MDP/SOCP $(X, \mathcal{U}, P, c)$ – infinite-horizon, exponentially-discounted

i.e. $\min_{u} E[c(x,u)]$ $\quad c(x,u) = \sum_{t=0}^{\infty} \gamma^t \cdot \mathcal{L}(x_t, u_t)$

s.t. $x^+ \sim P(x,u)$, $|X|, |\mathcal{U}| < \infty$ $\quad \gamma \in (0,1)$

- recall that the value $v^\pi : X \to \mathbb{R}$ of policy $\pi : X \to \Delta(\mathcal{U})$

is defined $\forall x \in X$: $v^\pi(x_0) = E[c(x,u) \mid x(0) = x_0, u(t) \sim \pi(x(t))]$

and satisfies Bellman eqn: $= \sum_{u_0 \in \mathcal{U}} \pi(u_0 | x_0) \sum_{x_1 \in X} P(x_1 | x_0, u_0)(\mathcal{L}(x_0,u_0) + \gamma \cdot v^\pi(x_1))$

$= E[\mathcal{L}(x_0,u_0) + \gamma \cdot v^\pi(x_1) \mid x_1 \sim P(x_0,u_0), u_0 \sim \pi(x_0)]$

$$= E\left[\mathcal{L}(x_0, u_0) + \gamma \cdot v^\pi(x_1) \,\middle|\, x_1 \sim P(x_0, u_0),\ u_0 \sim \pi(x_0) \right]$$

- so, removing subscripts: $\forall x \in X : v^\pi(x) = E\left[\mathcal{L}(x,u) + \gamma \cdot v^\pi(x^+) \right]$
(& suppressing distributions)

* this form of Bellman equation suggests we can approximate v^π
from trajectory data using Monte Carlo estimation

---

aside: Monte Carlo estimation (Robbins-Monroe)

- let $w : \Omega \to \mathbb{R}$ be a random variable whose mean we want to estimate

* given samples $\{w^n\}_{n=1}^N$ of this rv then $E[w] \simeq \underbrace{\overline{w}_N := \frac{1}{N} \sum_{n=1}^{N} w^n}_{\text{sample mean}}$

→ show that the sample mean can be computed recursively
(find a way to write $\overline{w}_{N+1}$ i.t.o. $\overline{w}_N$ and w^{N+1})

– $\overline{w}_{N+1} = \frac{1}{N+1} \sum_{n=1}^{N+1} w^n = \overline{w}_N + \frac{1}{N+1}\left(w^{N+1} - \overline{w}_N \right)$

- note that, by rearranging, we get $\overline{w}_{N+1} = \left(1 - \frac{1}{N+1}\right) \overline{w}_N + \frac{1}{N+1} w^{N+1}$
which has the form $x^+ = (1-\alpha) x + \alpha u$

* for any $\alpha \in (0,1)$, if u is rv with mean $E[u]$
then x is rv with mean $E[x] = E[u]$

∘ recalling that $\forall x \in X: v^\pi(x_0) = E[c(x,u) \mid x(0) = x_0]$
we can apply Monte Carlo estimation given trajectories $\{(x^n, u^n)\}_{n=1}^{N+1}$
generated by applying policy π from initial condition $x^n(0) = x_0$:

$$v_N^\pi(x_0) = \frac{1}{N}\sum_{n=1}^{N} c(x^n, u^n), \quad v_{N+1}^\pi(x_0) = v_N^\pi(x_0) + \alpha\cdot\left(c(x^{N+1}, u^{N+1}) - v_N^\pi(x_0)\right)$$

$\alpha \in (0,1)$, eg $\alpha = \frac{1}{N+1}$

stylize

$$v^+(x_0) = v(x_0) + \alpha\left(c(x,u) - v(x_0)\right), \quad c(x,u) = \sum_{t=0}^{\infty} \gamma^t \cdot \ell(x_t, u_t)$$

$$\begin{aligned} &= v(x_0) + \alpha\cdot\Big[\gamma^0\cdot\left(\ell(x_0,u_0) + \gamma v(x_1) - v(x_0)\right) \\ &\qquad + \gamma^1\cdot\left(\ell(x_1,u_1) + \gamma\cdot v(x_2) - v(x_1)\right) \\ &\qquad + \dots \\ &\qquad + \gamma^t\cdot\left(\ell(x_t,u_t) + \gamma\cdot v(x_{t+1}) - v(x_t)\right) \\ &\qquad + \dots\Big] \end{aligned}$$

$$* \quad v^+(x_0) = v(x_0) + \alpha\cdot\sum_{t=0}^{\infty}\gamma^t\cdot\underbrace{\left(\ell(x_t,u_t) + \gamma\cdot v(x_{t+1}) - v(x_t)\right)}_{=: d_t \text{ - temporal differences (TD)}}$$

↳ this is TD(0) – generalizes to TD(λ)

Q-learning

∘ given policy $\pi: X \to \Delta(U)$ with value $v^\pi: X \to \mathbb{R}$,
consider state-action policy quality

$Q^\pi: X \times U \to \mathbb{R}$ – expected cost of: 1°. choosing u in x; 2°. using π from then on

$$: (x,u) \mapsto \sum_{x^+ \in X} P(x^+ \mid x,u)\cdot\left(\ell(x,u) + \gamma\cdot v^\pi(x^+)\right)$$

∘ note that optimal quality function satisfies the Bellman equation

$x' \in X$

- note that optimal quality function satisfies the Bellman equation

$$q^*(x,u) = E\left[\mathcal{L}(x,u) + \gamma \cdot \min_{w \in \mathcal{U}} q^*(x,w) \right] \quad \text{b/c} \quad v^*(x) = \min_{u \in \mathcal{U}} q^*(x,u)$$

* applying Monte-Carlo estimation yields <u>q-learning</u> algorithm

$$q^+(x,u) = (1-\alpha)\, q(x,u) + \alpha \cdot \left(\mathcal{L}(x,u) + \gamma \cdot \min_{w \in \mathcal{U}} q(x,w) \right)$$

$\alpha \in (0,1)$ – learning rate