Lecture 7

EAIS S'26

Ken Nakamura

Last Time

- [ ] Grid-based
- [ ] Deepreach

This time

- [ ] Reachability RL
- [ ] online updates

Project Proposal Due Feb 16

Q: "Name two ways Deepreach modernized grid-based computation"

So far

$$0 = \min \left\{ \ell(s) - V(s,t),\ \frac{\partial V}{\partial t} + \underbrace{\max_a \frac{\partial V}{\partial s} \cdot f(s,a)}_{} \right\}$$

Grid	$\underbrace{\max_a \frac{\partial V}{\partial s} \cdot f(s,a)}_{}$	Grid nodes	backward in time
Deepreach	$\max_a \frac{\partial V}{\partial s} \cdot f(s,a)$	MLP	time curriculum

Example: Computing H

$$s = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} \quad \dot{s} = \begin{bmatrix} \cos\theta \\ \sin\theta \\ a \end{bmatrix} \quad \frac{\partial V}{\partial s} = \begin{bmatrix} P_x \\ P_y \\ P_\theta \end{bmatrix} \quad a \in [-1, 1]$$

$$\Rightarrow H(s,t) = \max_a \left[P_x \cos\theta + P_y \sin\theta + P_\theta \cdot a \right]$$

$$\Rightarrow a^* = \text{sign}\left[P_\theta \right]$$

$\Rightarrow$ control affine dyn w/ box ctrl limits is simple sign check

$$\max_a \frac{\partial V}{\partial s} \cdot f(s,a)$$

$$\dot{s} = f(s,a)$$

$$s_{t+1} = f(s_t, a_t)$$

- Real World rollout
- Simulator (eg Mujoco)
- World Model

① might not have $\dot{s} = f(s,a)$!

② computing $\max_a$ might be hard!

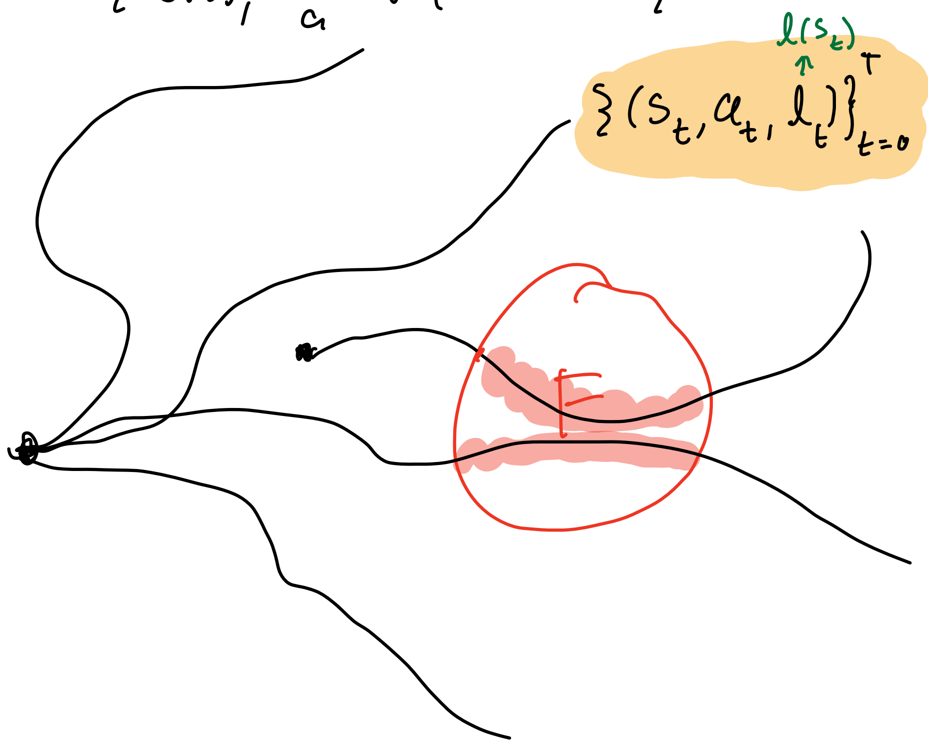
Setting

$$s' = f(s,a)$$

$$V(s) = \min \{ \ell(s), \max_a V(f(s,a)) \}$$

Goal Learn $V(s)$ via experience

discrete time HJ



"experience" in form of trajectory rollouts

To do this, need a few modifications...

1) Time discounting

Toy Ex: $V(s) = \min_{t \in [\tau, T]} \ell(s_t)$ as $T \to \infty$, $V(s) \to -\infty$!

$\Rightarrow$ bad

$T \to \infty$

$\ell(s) > \ell(s) > \ell(s) > \ell(s)$ $\quad \ell(s) = 0 \quad > \ell(s) > \ldots$

Add discount factor $\gamma \in [0, 1)$

↳ at each step, $(1-\gamma)$ chance the traj ends

$$V(s) = \underbrace{(1-\gamma)\,\ell(s)}_{\text{episode ends right now}} + \underbrace{\gamma \min\left\{\ell(s), \max_a V(f(s,a))\right\}}_{\text{w/ probability } \gamma \text{, we get to experience the future}}$$

Time-discount induces a "contraction"

Informally... for any initialization of $V(s)$, repeatedly applying the update across all states

$$V(s) \leftarrow (1-\gamma)\,\ell(s) + \gamma \min\left\{\ell(s), \max_a V(f(s,a))\right\}$$

$V(s)$ will eventually converge to a fixed pt

$\Rightarrow$ necessary for infinite horizon RL to converge

2) State-action Value fn

$V: S \to \mathbb{R}$ "worst case ℓ if you always do $\max_a V(f(s,a))$"

$Q: S \times A \to \mathbb{R}$ "worst-case ℓ if you first take a, then follow $\max_a V(f(s,a))$ for all following time"

$$Q(s,a) = (1-\gamma)\,\ell(s) + \gamma \min\{\ell(s), \max_{a' \in A} Q(s',a')\}$$

$s' = f(s,a)$

$$V(s) = \max_a Q(s,a) \qquad Q(s,a) = \min\{\ell(s), V(s')\}$$

$Q(s,a)$ allows us to learn the safety of any action from a given state s

Reachability RL

(A Kamatalu '18)
(Fisac '19 ICRA)

Objective: Jointly learn Q and Policy π

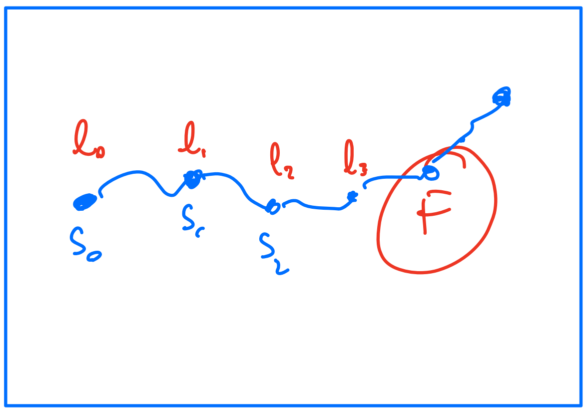
$Q(s,a) = \cdots\cdots$ see above

$\pi \approx \arg\max_a Q(s,a)$

① Initialize s_0
"env.reset()"

reset often times a Uniform distribution over states, or near unsafe states

② take actions in environment
↳ store (s, a, l, s') into a Replay Buffer



③ Fit Q: sample (s,a,l,s') from Buffer

Q_θ $\quad L(\theta) = \mathbb{E}_{(s,a,l,s')} \| Q_\theta(s,a) - y_{target} \|^2$

current policy π

$$y_{target} = (1-\gamma) l(s) + \gamma \left[\min \{ l(s), Q(s', \pi(s')) \} \right]$$

don't let gradients flow through y_{target}

④ Improve π_ψ $\quad L(\psi) = \mathbb{E}_{s} \left[-Q_\theta(s, \pi_\psi(s)) \right]$

minimize $-Q(s,a)$

$\Rightarrow$ maximize $Q(s,a)$

⑤ Repeat until done

Note: y_{target} is inaccurate estimate at the beginning of training

$$y_{target} = (1-\gamma) \underbrace{l(s) + \gamma \left[\min \{ l(s), \right.}_{\text{current state, accurate}} \left. \underbrace{Q(s', \pi(s'))}_{\text{estimate of future}} \} \right]$$

current state, accurate

estimate of future
↳ bad at first!

Idea (Fisac '19): anneal γ from low to high e.g. $\gamma = 0.8 \rightarrow 0.9999$

$\downarrow$

Can interpret this as doing backward-in-time computation!

Robust ver. ISAACS (Hsu et al '22)

actor $\pi_{safe}(s)$, disturbance $\pi_d(s)$, $Q(s, a, d)$

Sample (s, a, d, l, s') from replay buffer

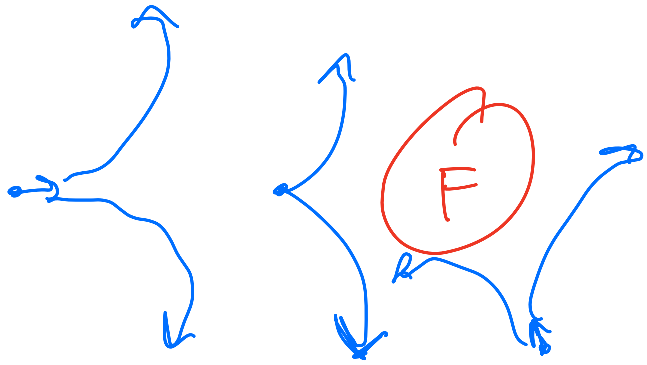
π_{safe} loss: $-Q(s, \pi_{safe}, d)$

π_d loss: $Q(s, a, \pi_d)$ $\rightarrow$ update π_d faster to give information adv

Critic loss $\| Q(s, a, d) - y_{targ} \|$

Observation:

Replay Buffer is filled primarily w/ safety-preserving actions



"always steer away from failure"

$\pi_{nom} : S \to A$ is different from π_{safe}

$Q(s, \underbrace{\pi_{nom}(s)}_{\text{OOD for } Q})$ might be bad / inaccurate

Alternative:

first $s' = f(s,a)$

↳ $Q(s,a) \approx \min \{ \ell(s), \overbrace{Q(s', \underbrace{\pi_{safe}(s')}_{\text{In-D for } Q})}^{\text{accurate}} \}$

Takeaway Learning is **not** a silver bullet, each method has trade-offs

	Scales?	Supervision Strength	dynamics	Accuracy
Grid	✗	✓	analytic & cont time	Grid resolution over states
Deepreach	✓	✗	analytic & cont time	Chosen Sampling dist. over states and time
RL	✓	✓	Simulator access + discrete time	Induced Sampling dist over states and actions