Markov Decision Processes and Solving Finite Problems

February 8, 2017
Overview of Upcoming Lectures

Feb 8: Markov decision processes, value iteration, policy iteration
Feb 13: Policy gradients
Feb 15: Learning $Q$-functions: $Q$-learning, SARSA, and others
Feb 22: Advanced $Q$-functions: replay buffers, target networks, double $Q$-learning

next... Advanced model learning and imitation learning
next... Advanced policy gradient methods, and the exploration problem
Overview for This Lecture

- This lecture assumes you have a known system with a finite number of states and actions.
- How to exactly solve for optimal policy
  - Value iteration
  - Policy iteration
  - Modified policy iteration
How Does This Lecture Fit In?

- Value Iteration \(\text{small updates + neural nets}\) \(\rightarrow\) deep Q-network methods
- Policy Iteration \(\text{small updates + neural nets}\) \(\rightarrow\) deep policy gradient methods
Markov Decision Process

Defined by the following components:

- **$S$: state space**, a set of states of the environment.
- **$A$: action space**, a set of actions, which the agent selects from at each timestep.
- **$P(r, s' | s, a)$**: a transition probability distribution.
  - Alternatively, $P(s' | s, a)$ and one of $R(s)$, $R(s, a)$ or $R(s, a, s')$
Partially Observed MDPs

- Instead of observing full state $s$, agent observes $y$, with $y \sim P(y \mid s)$.
- A MDP can be trivially mapped onto a POMDP
- A POMDP can be mapped onto an MDP:

  \[ \tilde{s}_0 = \{y_0\}, \quad \tilde{s}_1 = \{y_0, y_1\}, \quad \tilde{s}_2 = \{y_0, y_1, y_2\}, \ldots \]
Simple MDP: Frozen Lake

- Gym: FrozenLake-v0
- START state, GOAL state, other locations are FROZEN (safe) or HOLE (unsafe).
- Episode terminates when GOAL or HOLE state is reached
- Receive reward=1 when entering GOAL, 0 otherwise
- 4 directions are actions, but you move in wrong direction with probability 0.5.
Policies

- Deterministic policies $a = \pi(s)$
- Stochastic policies $a \sim \pi(a \mid s)$
Problems Involving MDPs

- **Policy optimization**: maximize expected reward with respect to policy $\pi$
  
  $$
  \text{maximize } \mathbb{E}_{\pi} \left[ \sum_{t=0}^{\infty} r_t \right]
  $$

- **Policy evaluation**: compute expected return for fixed policy $\pi$
  
  - return $:= \text{sum of future rewards in an episode (i.e., a trajectory)}$
    - Discounted return: $r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \ldots$
    - Undiscounted return: $r_t + r_{t+1} + \ldots + r_{T-1} + V(s_T)$
  
  - Performance of policy:
    $$
    \eta(\pi) = \mathbb{E} \left[ \sum_{t=0}^{\infty} \gamma^t r_t \right]
    $$
  
  - State value function:
    $$
    V^\pi(s) = \mathbb{E} \left[ \sum_{t=0}^{\infty} \gamma^t r_t \mid s_0 = s \right]
    $$
  
  - State-action value function:
    $$
    Q^\pi(s, a) = \mathbb{E} \left[ \sum_{t=0}^{\infty} \gamma^t r_t \mid s_0 = s, a_0 = a \right]
    $$
Value Iteration: Finite Horizon Case

- Problem:
  \[
  \max_{\pi_0} \max_{\pi_1} \ldots \max_{\pi_{T-1}} \mathbb{E}[r_0 + r_1 + \cdots + r_{T-1} + V_T(s_T)]
  \]

- Swap maxes and expectations:
  \[
  \max_{\pi_0} \mathbb{E} \left[ r_0 + \max_{\pi_1} \mathbb{E} \left[ r_1 + \cdots + \max_{\pi_{T-1}} \mathbb{E}[r_{T-1} + V_T(s_T)] \right] \right]
  \]

- Solve innermost problem: for each \( s \in S \)
  \[
  \pi_{T-1}(s), V_{T-1}(s) = \max_{a} \mathbb{E}_{sT} [r_{T-1} + V_T(s_T)]
  \]

- Original problem becomes
  \[
  \max_{\pi_0} \mathbb{E} \left[ r_0 + \max_{\pi_1} \mathbb{E} \left[ r_1 + \cdots + \max_{\pi_{T-2}} \mathbb{E}[r_{T-2} + V_{T-1}(s_{T-1})] \right] \right]
  \]
Value Iteration: Finite Horizon Case

**Algorithm 1** Finite Horizon Value Iteration

```latex
\begin{algorithm}
\textbf{for} \ t = T - 1, \ T - 2, \ldots, \ 0 \ \textbf{do}
    \textbf{for} \ s \in S \ \textbf{do}
        \pi_t(s), V_t(s) = \max_{a} \mathbb{E}[r_t + V_{t+1}(s_{t+1})]
    \textbf{end for}
\textbf{end for}
\end{algorithm}
```
Discounted Setting

- Discount factor $\gamma \in [0, 1)$, downweights future rewards
- Discounted return: $r_0 + \gamma r_1 + \gamma^2 r_2 + \ldots$
- Coefficients $(1, \gamma, \gamma^2, \ldots) \Rightarrow$ informally, we’re adding up $1 + \gamma + \gamma^2 + \cdots = 1/(1 - \gamma)$ timesteps. *Effective time horizon* $1/(1 - \gamma)$.
- Want to solve for policy that’ll optimize discounted sum of rewards from each state.
- Discounted problem can be obtained by adding transitions to “sink state” (where agent gets stuck and receives zero reward)

\[
\tilde{P}(s' \mid s, a) = \begin{cases} 
    P(s' \mid s, a) & \text{with probability } \gamma \\
    \text{sink state with probability } 1 - \gamma 
\end{cases}
\]
Infinite-Horizon VI Via Finite-Horizon VI

- maximize $\max_{\pi_0\pi_1\pi_2\ldots} \mathbb{E} \left[ r_0 + \gamma r_1 + \gamma^2 r_2 + \ldots \right]$

- Can rewrite with nested sum

$$\max_{\pi_0} \mathbb{E} \left[ r_0 + \gamma \max_{\pi_1} \mathbb{E} \left[ r_1 + \gamma \max_{\pi_2} \mathbb{E} \left[ r_2 + \ldots \right] \right] \right]$$

- Pretend there’s finite horizon $T$, ignore $r_T, r_{T+1}, \ldots$
  - error $\epsilon \leq r_{\text{max}} \gamma^T / (1 - \gamma)$
  - resulting nonstationary policy only suboptimal by $\epsilon$
  - $\pi_0, V_0$ converges to optimal policy as $T \to \infty$. 
Infinite-Horizon VI

Algorithm 2 Infinite-Horizon Value Iteration

Initialize $V^{(0)}$ arbitrarily.
for $n = 0, 1, 2, \ldots$ until termination condition do
    for $s \in S$ do
        $\pi^{(n+1)}(s), V^{(n+1)}(s) = \max_{a} E_{sT}[r_{T-1} + \gamma V^{(n)}(s_T)]$
    end for
end for

Note that $V^{(n)}$ is exactly $V_0$ in a finite-horizon problem with $n$ timesteps.
Infinite-Horizon VI: Operator View

1. $V \in \mathbb{R}^{|S|}$
2. VI update is a function $\mathcal{T} : \mathbb{R}^{|S|} \rightarrow \mathbb{R}^{|S|}$, called backup operator

$$[\mathcal{T}V](s) = \max_a \mathbb{E}_{s' | s,a} [r + \gamma V(s')]$$

**Algorithm 3** Infinite-Horizon Value Iteration (v2)

- Initialize $V^{(0)}$ arbitrarily.
- for $n = 0, 1, 2, \ldots$ until termination condition do
  - $V^{(n+1)} = \mathcal{T}V^{(n)}$
- end for
Backup operator $\mathcal{T}$ is a contraction with modulus $\gamma$ under $\infty$-norm

$$\|\mathcal{T}V - \mathcal{T}W\|_\infty \leq \gamma \|V - W\|_\infty$$

By contraction-mapping principle, $B$ has a fixed point, called $V^*$, and iterates $V, \mathcal{T}V, \mathcal{T}^2V, \cdots \rightarrow V^*, \gamma$. 
Policy Evaluation

- Problem: how to evaluate fixed policy $\pi$:
  \[ V^{\pi,\gamma}(s) = \mathbb{E} \left[ r_0 + \gamma r_1 + \gamma^2 r_2 + \ldots \mid s_0 = s \right] \]

- Can consider finite-horizon problem
  \[ \mathbb{E} \left[ r_0 + r_1 + \cdots + r_{T-1} + V_T(s_T) \right] = \mathbb{E} \left[ r_0 + \gamma \mathbb{E} \left[ r_1 + \cdots + \gamma \mathbb{E} \left[ r_{T-1} + V_T(s_T) \right] \right] \right] \]

- Backwards recursion involves a backup operation $V_t = \mathcal{T}^{\pi} V_{t+1}$, where
  \[ [\mathcal{T}^{\pi} V](s) = \mathbb{E}_{s'} \mid s, a = \pi(s) \left[ r + \gamma V(s') \right] \]

- $\mathcal{T}^{\pi}$ is also a contraction with modulus $\gamma$, sequence
  \( V, \mathcal{T}^{\pi} V, (\mathcal{T}^{\pi})^2 V, \ldots \rightarrow V^{\pi,\gamma} \)

- $V = \mathcal{T}^{\pi} V$ is a linear equation that we can solve exactly:
  \[ V(s) = \sum_{s'} P(s' \mid s, a = \pi(s)) \left[ r(s, a, s') + \gamma V(s') \right] \]
Policy Iteration: Overview

- Alternate between
  1. Evaluate policy $\pi \Rightarrow V^\pi$
  2. Set new policy to be greedy policy for $V^\pi$

$$\pi(s) = \arg\max_a \mathbb{E}_{s'} | s, a [r + \gamma V^\pi(s')]$$

- Guaranteed to converge to optimal policy and value function in a finite number of iterations, when $\gamma < 1$
- Value function converges faster than in value iteration\(^1\)

---

Algorithm 4 Policy Iteration

Initialize $\pi^{(0)}$.

for $n = 1, 2, \ldots$ do

\[ V^{(n-1)} = \text{Solve} [V = T^{\pi^{(n-1)}} V] \]

\[ \pi^{(n)} = G V^{\pi^{(n-1)}} \]

end for
Policy Iteration: Convergence

- Policy sequence $\pi^{(0)}, \pi^{(1)}, \pi^{(2)}, \ldots$ is monotonically improving, with nondecreasing value function:
  $V_{\pi^{(0)}} \leq V_{\pi^{(1)}} \leq V_{\pi^{(2)}} \leq \ldots$. Informal argument:
  - Switch policy at first timestep from $\pi^{(0)}$ to $\pi^{(1)}$
    - Before: $V(s_0) = \mathbb{E}_{s_1} | s_0, a_0 = \pi^{(0)}(s_0) [r_0 + \gamma V^{\pi}(s_1)]$
    - After: $V(s_0) = \max_{a_0} \mathbb{E}_{s_1} | s_0, a_0 = \pi^{(0)}(s_0) [r_0 + \gamma V(s_1)]$
  - $V_{\pi^{(1)}}^{\pi^{(0)}} \geq V_{\pi^{(0)}}^{\pi^{(0)}} \pi^{(0)} \pi^{(0)} \pi^{(0)} \ldots$
  - $V_{\pi^{(1)}}^{\pi^{(1)}} \pi^{(0)} \pi^{(0)} \pi^{(0)} \pi^{(0)} \ldots \geq V_{\pi^{(1)}}^{\pi^{(0)}} \pi^{(0)} \pi^{(0)} \pi^{(0)} \pi^{(0)} \pi^{(0)} \ldots$
  - $\ldots \Rightarrow$
    $V_{\pi^{(1)}}^{\pi^{(1)}} \pi^{(1)} \pi^{(1)} \pi^{(1)} \ldots \geq V_{\pi^{(0)}}^{\pi^{(0)}} \pi^{(0)} \pi^{(0)} \pi^{(0)} \pi^{(0)} \ldots$
  - If the value function does not increase, then we’re done:
    $V_{\pi^{(n)}} = V_{\pi^{(n+1)}} \Rightarrow V_{\pi^{(n)}} = T V_{\pi^{(n)}} \Rightarrow \pi^{(n)} = \pi^*$. 
Modified Policy Iteration

- Update $\pi$ to be the greedy policy, then value function with $k$ backups ($k$-step lookahead)

**Algorithm 5 Modified Policy Iteration**

- Initialize $V^{(0)}$.

  for $n = 1, 2, \ldots$ do
  $\pi^{(n+1)} = GV^{(n)}$
  $V^{(n+1)} = (T^{\pi^{(n+1)}})^k V^{(n)}$, for integer $k \geq 1$

  end for

- $k = 1$: value iteration
- $k = \infty$: policy iteration
- See Puterman’s textbook\(^2\) for more details

---

The End

- Homework: will be released later today or early tomorrow, due on Feb 22
- Next time: policy gradient methods: infinitesimal policy iteration