# Stanford·CS234 | Reinforcement Learning (2019)

## CS234 (2019)· 课程资料包 @ShowMeAI

**视频**
中英双语字幕

**课件**
一键打包下载

**笔记**
官方笔记翻译

**代码**
作业项目解析

**视频 ·B 站 [ 扫码或点击链接 ]**
https://www.bilibili.com/video/BV1H64y1x7GH

**课件 & 代码 · 博客 [ 扫码或点击链接 ]**
http://blog.showmeai.tech/cs234

斯坦福 reinforcement learning 马尔可夫决策过程 DQN 强化学习 值函数方法 policy gradient Q-learning 梯度策略 Model free 蒙特卡洛搜索树

Awesome AI Courses Notes Cheatsheets 是 **ShowMeAI** 资料库的分支系列，覆盖最具知名度的 **TOP20+** 门 AI 课程，旨在为读者和学习者提供一整套高品质中文学习笔记和速查表。

**点击**课程名称，跳转至课程**资料包**页面，**一键下载**课程全部资料！

| 机器学习 | 深度学习 | 自然语言处理 | 计算机视觉 |
|---|---|---|---|
| Stanford · CS229 | Stanford · CS230 | Stanford · CS224n | Stanford · CS231n |
| **# Awesome AI Courses Notes Cheatsheets· 持续更新中** | | | |
| **知识图谱** | **图机器学习** | **深度强化学习** | **自动驾驶** |
| Stanford · CS520 | Stanford · CS224W | UCBerkeley · CS285 | MIT · 6.S094 |

## 微信公众号

资料下载方式 2：扫码点击底部菜单栏

称为 **AI 内容创作者？** 回复 [ 添砖加瓦 ]

# Lecture 3: Model-Free Policy Evaluation: Policy Evaluation Without Knowing How the World Works[1]

Emma Brunskill

CS234 Reinforcement Learning

Winter 2019

[1] Material builds on structure from David SIlver's Lecture 4: Model-Free Prediction. Other resources: Sutton and Barto Jan 1 2018 draft Chapter/Sections: 5.1; 5.5; 6.1-6.3

# Today's Plan

- Last Time:
  - Markov reward / decision processes
  - Policy evaluation & control when have true model (of how the world works)
- **Today**
  - **Policy evaluation without known dynamics & reward models**
- Next Time:
  - Control when don't have a model of how the world works

# This Lecture: Policy Evaluation

- Estimating the expected return of a particular policy if don't have access to true MDP models
- Dynamic programming
- Monte Carlo policy evaluation
    - Policy evaluation when don't have a model of how the world work
        - Given on-policy samples
- Temporal Difference (TD)
- Metrics to evaluate and compare algorithms

# Recall

- Definition of Return, $G_t$ (for a MRP)
  - Discounted sum of rewards from time step $t$ to horizon

$$G_t = \underline{r_t} + \overset{(0,1)}{\gamma} r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 r_{t+3} + \cdots$$

- Definition of State Value Function, $V^\pi(s)$
  - Expected return from starting in state $s$ under policy $\pi$

$$V^\pi(s) = \mathbb{E}_\pi[G_t | s_t = s] = \mathbb{E}_\pi[r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 r_{t+3} + \cdots | s_t = s]$$

- Definition of State-Action Value Function, $Q^\pi(s, a)$
  - Expected return from starting in state $s$, taking action $a$ and then following policy $\pi$

$$\begin{aligned} Q^\pi(s, a) &= \mathbb{E}_\pi[G_t | s_t = s, a_t = a] \\ &= \mathbb{E}_\pi[r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 r_{t+3} + \cdots | s_t = s, a_t = a] \end{aligned}$$

# Dynamic Programming for Policy Evaluation

given dynamics/transition $P$ & reward model $r$

- Initialize $V_0^\pi(s) = 0$ for all $s$
- For $k = 1$ until convergence $\quad ||V_k^\pi - V_{k-1}^\pi|| < \epsilon$
  - For all $s$ in $S$

$$V_k^\pi(s) = r(s, \pi(s)) + \gamma \sum_{s' \in S} p(s'|s, \pi(s)) V_{k-1}^\pi(s')$$

# Dynamic Programming for Policy $\pi$, Value Evaluation

- Initialize $V_0^\pi(s) = 0$ for all $s$
- For $k = 1$ until convergence
  - For all $s$ in $S$

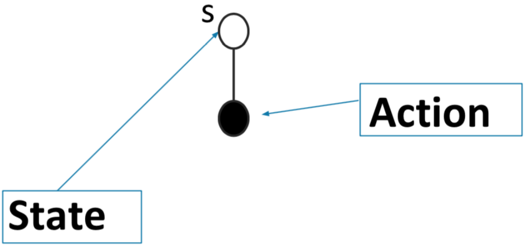
$$V_k^\pi(s) = r(s, \pi(s)) + \gamma \sum_{s' \in S} p(s'|s, \pi(s)) V_{k-1}^\pi(s')$$

- $V_k^\pi(s)$ is exact value of $k$-horizon value of state $s$ under policy $\pi$
- $V_k^\pi(s)$ is an estimate of infinite horizon value of state $s$ under policy $\pi$

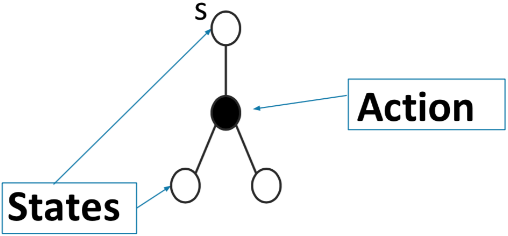
$$V^\pi(s) = \mathbb{E}_\pi[G_t|s_t = s] \approx \mathbb{E}_\pi[r_t + \gamma V_{k-1}|s_t = s]$$

# Dynamic Programming Policy Evaluation $V^{\pi}(s) \leftarrow \mathbb{E}_{\pi}[r_t + \gamma V_{k-1} | s_t = s]$

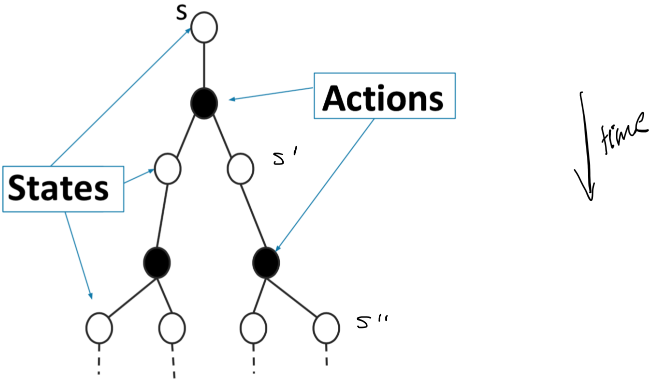
$\uparrow p(s'|s, \pi(s))$



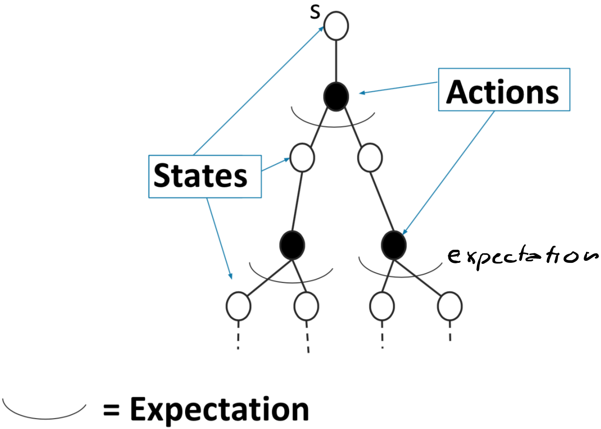
# Dynamic Programming Policy Evaluation $V^\pi(s) \leftarrow \mathbb{E}_\pi[r_t + \gamma V_{k-1} | s_t = s]$



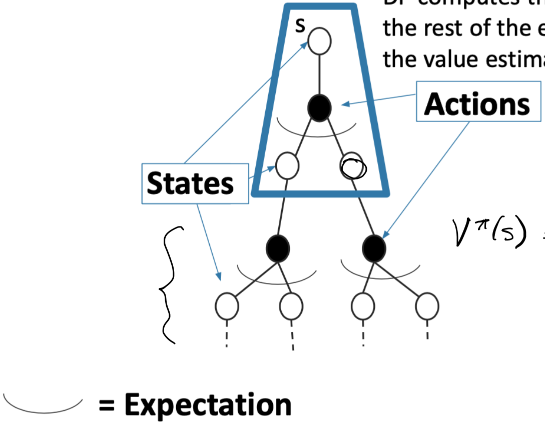
# Dynamic Programming Policy Evaluation $V^\pi(s) \leftarrow \mathbb{E}_\pi[r_t + \gamma V_{k-1}|s_t = s]$



# Dynamic Programming Policy Evaluation $V^{\pi}(s) \leftarrow \mathbb{E}_{\pi}[r_t + \gamma V_{k-1} | s_t = s]$



# Dynamic Programming Policy Evaluation $V^\pi(s) \leftarrow \mathbb{E}_\pi[r_t + \gamma V_{k-1} | s_t = s]$



DP computes this, **bootstrapping** the rest of the expected return by the value estimate $V_{k-1}$

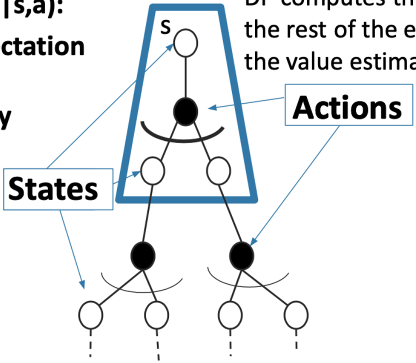
$$V^\pi(s) = r(s, \pi(s)) + \gamma \sum_{s'} p(s'|s,a) \underline{V^\pi_{k-1}(s')}$$

- Bootstrapping: Update for $V$ uses an estimate

# Dynamic Programming Policy Evaluation $V^{\pi}(s) \leftarrow \mathbb{E}_{\pi}[r_t + \gamma V_{k-1} | s_t = s]$

**Know model P(s'|s,a):**

**reward and expectation over next states computed exactly**



DP computes this, bootstrapping the rest of the expected return by the value estimate $V_{\underbrace{k-1}}$

**= Expectation**

- Bootstrapping: Update for $V$ uses an estimate

# Policy Evaluation: $V^\pi(s) = \mathbb{E}_\pi[G_t | s_t = s]$

- $G_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 r_{t+3} + \cdots$ in MDP $M$ under policy $\pi$
- Dynamic Programming
  - $V^\pi(s) \approx \mathbb{E}_\pi[r_t + \gamma V_{k-1} | s_t = s]$
  - **Requires model of MDP** $M$
  - Bootstraps future return using value estimate
  - Requires Markov assumption: bootstrapping regardless of history
- What if don't know dynamics model $P$ and/ or reward model $R$?
- **Today: Policy evaluation without a model**
  - Given data and/or ability to interact in the environment
  - Efficiently compute a good estimate of a policy $\pi$

# This Lecture Overview: Policy Evaluation

- Dynamic Programming
- **Evaluating the quality of an estimator**
- **Monte Carlo policy evaluation**
  - Policy evaluation when don't know dynamics and/or reward model
    - Given on policy samples
- Temporal Difference (TD)
- Metrics to evaluate and compare algorithms

# Monte Carlo (MC) Policy Evaluation

- $G_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 r_{t+3} + \cdots$ in MDP $M$ under policy $\pi$
- $V^\pi(s) = \mathbb{E}_{T \sim \pi}[G_t | s_t = s]$
  - Expectation over trajectories $T$ generated by following $\pi$
- Simple idea: Value = mean return
- If trajectories are all finite, sample set of trajectories & average returns

# Monte Carlo (MC) Policy Evaluation

- If trajectories are all finite, sample set of trajectories & average returns
- Does not require MDP dynamics/rewards
- No bootstrapping
- Does not assume state is Markov
- Can **only** be applied to episodic MDPs
  - Averaging over returns from a complete episode
  - Requires each episode to terminate

# Monte Carlo (MC) On Policy Evaluation

- Aim: estimate $V^\pi(s)$ given episodes generated under policy $\pi$
  - $s_1, a_1, r_1, s_2, a_2, r_2, \ldots$ where the actions are sampled from $\pi$
- $G_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 r_{t+3} + \cdots$ in MDP $M$ under policy $\pi$
- $V^\pi(s) = \mathbb{E}_\pi[G_t | s_t = s]$
- MC computes empirical mean return
- Often do this in an incremental fashion
  - After each episode, update estimate of $V^\pi$

# First-Visit Monte Carlo (MC) On Policy Evaluation

Initialize $\overbrace{N(s)}^{\text{\# times visited a state}} = 0,\ G(s) = 0\ \forall s \in S$

Loop

- Sample episode $i = s_{i,1}, a_{i,1}, r_{i,1}, s_{i,2}, a_{i,2}, r_{i,2}, \ldots, s_{i,T_i}$
- Define $G_{i,t} = r_{i,t} + \gamma r_{i,t+1} + \gamma^2 r_{i,t+2} + \cdots \gamma^{T_i - 1} r_{i,T_i}$ as return from time step $t$ onwards in $i$th episode
- For each state $s$ visited in episode $i$
  - For **first** time $t$ that state $s$ is visited in episode $i$
    - Increment counter of total first visits: $N(s) = N(s) + 1$
    - Increment total return $G(s) = G(s) + G_{i,t}$
    - Update estimate $\underbrace{V^\pi(s)} = G(s)/N(s)$

# Bias, Variance and MSE

- Consider a statistical model that is parameterized by $\theta$ and that determines a probability distribution over observed data $P(x|\theta)$
- Consider a statistic $\hat{\theta}$ that provides an estimate of $\theta$ and is a function of observed data $x$ $\hat{\theta} = f(x)$
  - E.g. for a Gaussian distribution with known variance, the average of a set of i.i.d data points is an estimate of the mean of the Gaussian
- Definition: the bias of an estimator $\hat{\theta}$ is:

$$Bias_{\theta}(\hat{\theta}) = \underbrace{\mathbb{E}_{x|\theta}[\hat{\theta}]}_{} - \underbrace{\theta}_{}$$

- Definition: the variance of an estimator $\hat{\theta}$ is:

$$Var(\hat{\theta}) = \mathbb{E}_{x|\theta}[(\hat{\theta} - \mathbb{E}[\hat{\theta}])^2]$$

- Definition: mean squared error (MSE) of an estimator $\hat{\theta}$ is:

$$MSE(\hat{\theta}) = Var(\hat{\theta}) + Bias_{\theta}(\hat{\theta})^2$$

# First-Visit Monte Carlo (MC) On Policy Evaluation

Initialize $N(s) = 0,\ G(s) = 0\ \forall s \in S$

Loop

- Sample episode $i = s_{i,1}, a_{i,1}, r_{i,1}, s_{i,2}, a_{i,2}, r_{i,2}, \ldots, s_{i,T_i}$
- Define $G_{i,t} = r_{i,t} + \gamma r_{i,t+1} + \gamma^2 r_{i,t+2} + \cdots \gamma^{T_i-1} r_{i,T_i}$ as return from time step $t$ onwards in $i$th episode
- For each state $s$ visited in episode $i$
  - For **first** time $t$ that state $s$ is visited in episode $i$
    - Increment counter of total first visits: $N(s) = N(s) + 1$
    - Increment total return $G(s) = G(s) + G_{i,t}$
    - Update estimate $V^\pi(s) = G(s)/N(s)$

Properties:

- $V^\pi$ estimator is an unbiased estimator of true $\mathbb{E}_\pi[G_t|s_t = s]$
- By law of large numbers, as $N(s) \to \infty$, $V^\pi(s) \to \mathbb{E}_\pi[G_t|s_t = s]$

consistent

# **Every-Visit** Monte Carlo (MC) On Policy Evaluation

```
Initialize $N(s) = 0,\ G(s) = 0\ \forall s \in S$
Loop
    • Sample episode $i = s_{i,1}, a_{i,1}, r_{i,1}, s_{i,2}, a_{i,2}, r_{i,2}, \dots, s_{i,T_i}$
    • Define $G_{i,t} = r_{i,t} + \gamma r_{i,t+1} + \gamma^2 r_{i,t+2} + \cdots \gamma^{T_i-1} r_{i,T_i}$ as return from time step $t$ onwards in $i$th episode
    • For each state $s$ visited in episode $i$
        • For every time $t$ that state $s$ is visited in episode $i$
            • Increment counter of total first visits: $N(s) = N(s) + 1$
            • Increment total return $G(s) = G(s) + G_{i,t}$
            • Update estimate $V^{\pi}(s) = G(s)/N(s)$
```

# **Every-Visit** Monte Carlo (MC) On Policy Evaluation

Initialize $N(s) = 0,\ G(s) = 0\ \forall s \in S$

Loop

- Sample episode $i = s_{i,1}, a_{i,1}, r_{i,1}, s_{i,2}, a_{i,2}, r_{i,2}, \ldots, s_{i,T_i}$
- Define $G_{i,t} = r_{i,t} + \gamma r_{i,t+1} + \gamma^2 r_{i,t+2} + \cdots \gamma^{T_i - 1} r_{i,T_i}$ as return from time step $t$ onwards in $i$th episode
- For each state $s$ visited in episode $i$
    - For **every** time $t$ that state $s$ is visited in episode $i$
        - Increment counter of total first visits: $N(s) = N(s) + 1$
        - Increment total return $G(s) = G(s) + G_{i,t}$
        - Update estimate $V^\pi(s) = G(s)/N(s)$

Properties:

- $V^\pi$ every-vist MC estimator is an **biased** estimator of $V^\pi$
- But consistent estimator and often has better MSE

# Incremental Monte Carlo (MC) On Policy Evaluation

After each episode $i = s_{i,1}, a_{i,1}, r_{i,1}, s_{i,2}, a_{i,2}, r_{i,2}, \ldots$

- Define $G_{i,t} = r_{i,t} + \gamma r_{i,t+1} + \gamma^2 r_{i,t+2} + \cdots$ as return from time step $t$ onwards in $i$th episode
- For state $s$ visited at time step $t$ in episode $i$
  - Increment counter of total first visits: $N(s) = N(s) + 1$
  - Update estimate

$$V^\pi(s) = V^\pi(s)\frac{N(s)-1}{N(s)} + \frac{G_{i,t}}{N(s)} = V^\pi(s) + \frac{1}{N(s)}(G_{i,t} - V^\pi(s))$$

# Incremental Monte Carlo (MC) On Policy Evaluation, Running Mean

Initialize $N(s) = 0$, $G(s) = 0 \; \forall s \in S$

Loop

- Sample episode $i = s_{i,1}, a_{i,1}, r_{i,1}, s_{i,2}, a_{i,2}, r_{i,2}, \ldots, s_{i,T_i}$
- Define $G_{i,t} = r_{i,t} + \gamma r_{i,t+1} + \gamma^2 r_{i,t+2} + \cdots \gamma^{T_i - 1} r_{i,T_i}$ as return from time step $t$ onwards in $i$th episode
- For state $s$ visited at time step $t$ in episode $i$
    - Increment counter of total first visits: $N(s) = N(s) + 1$
    - Update estimate

$$V^\pi(s) = V^\pi(s) + \alpha(G_{i,t} - V^\pi(s))$$

- $\alpha = \frac{1}{N(s)}$: identical to every visit MC
- $\alpha > \frac{1}{N(s)}$: forget older data, helpful for non-stationary domains

# Check Your Understanding: MC On Policy Evaluation

Initialize $N(s) = 0$, $G(s) = 0$ $\forall s \in S$ $\quad V^{\pi}(s) = 0 \quad \forall s$

Loop

- Sample episode $i = s_{i,1}, a_{i,1}, r_{i,1}, s_{i,2}, a_{i,2}, r_{i,2}, \ldots, s_{i,T_i}$
- $G_{i,t} = r_{i,t} + \gamma r_{i,t+1} + \gamma^2 r_{i,t+2} + \cdots \gamma^{T_i - 1} r_{i,T_i}$
- For each state $s$ visited in episode $i$
  - For **first or every** time $t$ that state $s$ is visited in episode $i$
    - $N(s) = N(s) + 1$, $G(s) = G(s) + G_{i,t}$
    - Update estimate $V^{\pi}(s) = G(s)/N(s)$

Example: $\quad s1 \quad s7$

- Mars rover: R = [ 1 0 0 0 0 0 +10] for any action
- $\pi(s) = a_1$ $\forall s$, $\gamma \underline{= 1}$. any action from $s_1$ and $s_7$ terminates episode
- Trajectory = ($s_3$, $a_1$, 0, $s_2$, $a_1$, 0, $s_2$, $a_1$, 0, $s_1$, $a_1$, 1, terminal)
- First visit MC estimate of $V$ of each state?

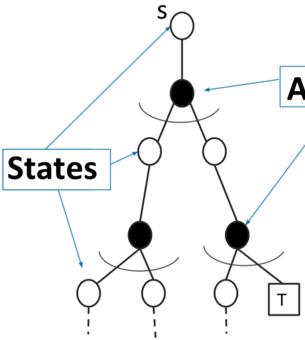
  $V(s_1) = 1 \quad V(s_2) = 1 \quad V(s_3) = 1 \quad V(s_4) = 0 \quad V(s_5) = 0$

  $V(s_6) = 0$

  $V(s_7) = 0$
- Every visit MC estimate of $s_2$? $\quad V(s_2) = 1$

# MC Policy Evaluation

$$V^\pi(s) = V^\pi(s) + \overset{\frac{1}{N(s)}}{\alpha}(G_{i,t} - V^\pi(s))$$



$s_1$ or $s_7$ Mars

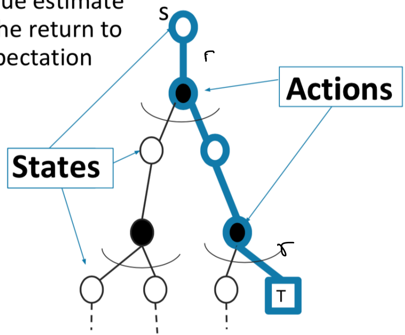
‿ = Expectation

T = **Terminal state**

# MC Policy Evaluation

$$V^\pi(s) = V^\pi(s) + \alpha(\overset{i \swarrow \quad \gamma}{G_{i,t}} - V^\pi(s))$$

MC updates the value estimate using a **sample** of the return to approximate an expectation



‿ = Expectation

$\boxed{\text{T}}$ **= Terminal state**

# Monte Carlo (MC) Policy Evaluation Key Limitations

- Generally high variance estimator
    - Reducing variance can require a lot of data
- Requires episodic settings
    - Episode must end before data from that episode can be used to update the value function

# Monte Carlo (MC) Policy Evaluation Summary

- Aim: estimate $V^{\pi}(s)$ given episodes generated under policy $\pi$
  - $s_1, a_1, r_1, s_2, a_2, r_2, \ldots$ where the actions are sampled from $\pi$
- $G_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 r_{t+3} + \cdots$ in MDP $M$ under policy $\pi$
- $V^{\pi}(s) = \mathbb{E}_{\pi}[G_t | s_t = s]$
- Simple: Estimates expectation by empirical average (given episodes sampled from policy of interest) or reweighted empirical average (importance sampling)
- Updates value estimate by using a **sample** of return to approximate the expectation
- No bootstrapping
- Converges to true value under some (generally mild) assumptions

# This Lecture: Policy Evaluation

- Estimating the expected return of a particular policy if don't have access to true MDP models
- Dynamic programming
- Monte Carlo policy evaluation
    - Policy evaluation when don't have a model of how the world work
        - Given on-policy samples
- **Temporal Difference (TD)**
- Metrics to evaluate and compare algorithms

# Temporal Difference Learning

- "If one had to identify one idea as central and novel to reinforcement learning, it would undoubtedly be temporal-difference (TD) learning." – Sutton and Barto 2017
- Combination of Monte Carlo & dynamic programming methods
- Model-free
- **Bootstraps and samples**
- Can be used in episodic or infinite-horizon non-episodic settings
- Immediately updates estimate of $V$ after each $(s, a, r, s')$ tuple

# Temporal Difference Learning for Estimating $V$

- Aim: estimate $V^\pi(s)$ given episodes generated under policy $\pi$
- $G_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 r_{t+3} + \cdots$ in MDP $M$ under policy $\pi$
- $V^\pi(s) = \mathbb{E}_\pi[G_t | s_t = s]$
- Recall Bellman operator (if know MDP models)

$$B^\pi V(s) = r(s, \pi(s)) + \gamma \sum_{s' \in S} \underbrace{p(s'|s, \pi(s))}\underbrace{V(s')}$$

- In incremental every-visit MC, update estimate using 1 sample of return (for the current $i$th episode)

$$V^\pi(s) = V^\pi(s) + \alpha(\underbrace{G_{i,t}} - V^\pi(s))$$

- Insight: have an estimate of $V^\pi$, use to estimate expected return

$$V^\pi(s) = V^\pi(s) + \alpha([\overbrace{r_t + \gamma V^\pi(s_{t+1})}] - V^\pi(s))$$

# Temporal Difference $[TD(0)]$ Learning

- Aim: estimate $V^\pi(s)$ given episodes generated under policy $\pi$
    - $s_1, a_1, r_1, s_2, a_2, r_2, \ldots$ where the actions are sampled from $\pi$
- Simplest TD learning: update value towards estimated value

$$V^\pi(s_t) = V^\pi(s_t) + \alpha(\underbrace{\overset{\sum_{s'} p(s'|s,a)\, \underline{V}^\pi(s')}{[r_t + \gamma V^\pi(s_{t+1})]}}_{\text{TD target}} - V^\pi(s_t))$$

- TD error:

$$\delta_t = \overbrace{r_t + \gamma \underline{V}^\pi(s_{t+1})} - \underline{V^\pi(s_t)} \swarrow \approx \text{expec over } S'$$

- Can immediately update value estimate after $(s, a, r, s')$ tuple
- Don't need episodic setting

# Temporal Difference [$TD(0)$] Learning Algorithm

Input: $\alpha$

Initialize $V^\pi(s) = 0, \forall s \in S$

Loop

- Sample **tuple** $(s_t, \overset{\pi(s_t)}{a_t}, r_t, s_{t+1})$
- $V^\pi(s_t) = V^\pi(s_t) + \alpha(\underbrace{[r_t + \gamma V^\pi(s_{t+1})]}_{\text{TD target}} - V^\pi(s_t))$

# Check Your Understanding: TD Learning

Input: $\alpha$
Initialize $V^\pi(s) = 0$, $\forall s \in S$
Loop

- Sample **tuple** $(s_t, a_t, r_t, s_{t+1})$
- $V^\pi(s_t) = V^\pi(s_t) + \alpha(\underbrace{[r_t + \gamma V^\pi(s_{t+1})]}_{\text{TD target}} - V^\pi(s_t))$

$\alpha = 1$

$V(s_3) = 0$
$V(s_2) = 0$
$V(s_2) = 0$
$V(s_1) = 1$

- $s_3$ $a_1$ $0$ $s_2$
$s_2$ $a_1$ $0$ $s_2$
$s_2$ $a_1$ $0$ $s_1$
$s_1$ $a_1$ $+1$ term

Example:

- Mars rover: R = [ 1 0 0 0 0 0 +10] for any action
- $\pi(s) = a_1$ $\forall s$, $\gamma = 1$. any action from $s_1$ and $s_7$ terminates episode
- Trajectory = ($s_3$, $a_1$, 0, $s_2$, $a_1$, 0, $s_2$, $a_1$, 0, $s_1$, $a_1$, 1, terminal)
- First visit MC estimate of $V$ of each state? [1 1 1 0 0 0 0]
- Every visit MC estimate of $V$ of $s2$? 1
- TD estimate of all states (init at 0) with $\alpha = 1$?

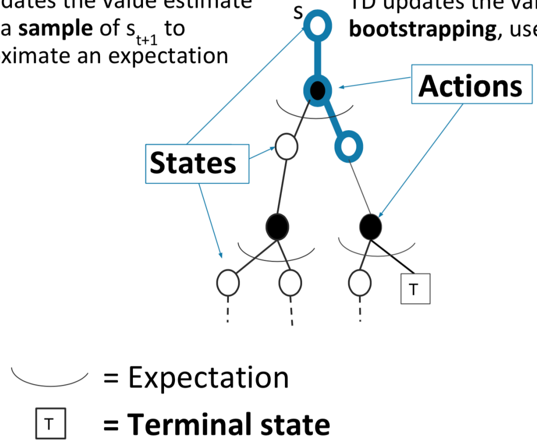
[1 0 0 0 0 0 0]

# Temporal Difference Policy Evaluation

$$V^\pi(s_t) = V^\pi(s_t) + \alpha([r_t + \gamma \underbrace{V^\pi(s_{t+1})]}_{} - V^\pi(s_t))$$

TD updates the value estimate using a **sample** of $s_{t+1}$ to approximate an expectation

TD updates the value estimate by **bootstrapping**, uses estimate of $V(s_{t+1})$



# This Lecture: Policy Evaluation

- Estimating the expected return of a particular policy if don't have access to true MDP models
- Dynamic programming
- Monte Carlo policy evaluation
  - Policy evaluation when don't have a model of how the world work
    - Given on-policy samples
    - Given off-policy samples
- Temporal Difference (TD)
- **Metrics to evaluate and compare algorithms**

# Check Your Understanding: For Dynamic Programming, MC and TD Methods, Which Properties Hold?

| | DP | MC | TD |
|---|---|---|---|
| Usable when no models of current domain | | ✓ | ✓ |
| Handles continuing (non-episodic) domains | ✓ | | ✓ |
| Handles Non-Markovian domains | | ✓ | |
| consistent<br>Converges to true value in limit [1] (Markov) | ✓ | ✓ | ✓ |
| Unbiased estimate of value | | ✓<br>1st visit<br>every visit biased | |

[1] For tabular representations of value function. More on this in later lectures

# Some Important Properties to Evaluate Policy Evaluation Algorithms

- Usable when no models of current domain
    - DP: No MC: Yes TD: Yes
- Handles continuing (non-episodic) domains
    - DP: Yes MC: No TD: Yes
- Handles Non-Markovian domains
    - DP: No MC: Yes TD: No
- Converges to true value in limit [2]
    - DP: Yes MC: Yes TD: Yes
- Unbiased estimate of value
    - DP: NA MC: Yes TD: No

[2] For tabular representations of value function. More on this in later lectures

# Some Important Properties to Evaluate Model-free Policy Evaluation Algorithms

- Bias/variance characteristics
- Data efficiency
- Computational efficiency

# Bias/Variance of Model-free Policy Evaluation Algorithms

- Return $G_t$ is an unbiased estimate of $V^\pi(s_t)$
- TD target $[r_t + \gamma V^\pi(s_{t+1})]$ is a biased estimate of $V^\pi(s_t)$
- But often much lower variance than a single return $G_t$
- Return function of multi-step sequence of random actions, states & rewards
- TD target only has one random action, reward and next state
- MC
  - Unbiased
  - High variance
  - Consistent (converges to true) even with function approximation
- TD
  - Some bias
  - Lower variance
  - TD(0) converges to true value with tabular representation
  - TD(0) does not always converge with function approximation

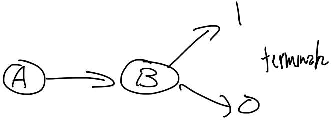
| $s_1$ | $s_2$ | $s_3$ | $s_4$ | $s_5$ | $s_6$ | $s_7$ |
|---|---|---|---|---|---|---|
| $R(s_1) = +1$<br>*Okay*<br>*Field Site* | $R(s_2) = 0$ | $R(s_3) = 0$ | $R(s_4) = 0$<br> | $R(s_5) = 0$ | $R(s_6) = 0$ | $R(s_7) = +10$<br>*Fantastic*<br>*Field Site* |

- Mars rover: R = [ 1 0 0 0 0 0 +10] for any action
- $\pi(s) = a_1 \; \forall s$, $\gamma = 1$. any action from $s_1$ and $s_7$ terminates episode
- Trajectory = $(s_3, a_1, 0, s_2, a_1, 0, s_2, a_1, 0, s_1, a_1, 1, \text{terminal})$
- First visit MC estimate of $V$ of each state? [1 1 1 0 0 0 0]
- Every visit MC estimate of $V$ of $s_2$? 1
- TD estimate of all states (init at 0) with $\alpha = 1$ is [1 0 0 0 0 0 0]
- TD(0) only uses a data point $(s, a, r, s')$ once
- Monte Carlo takes entire return from $s$ to end of episode

# Batch MC and TD

- Batch (Offline) solution for finite dataset
    - Given set of $K$ episodes
    - Repeatedly sample an episode from $K$
    - Apply MC or TD(0) to the sampled episode
- What do MC and TD(0) converge to?

# AB Example: (Ex. 6.4, Sutton & Barto, 2018



- Two states $A, B$ with $\gamma = 1$
- Given 8 episodes of experience:
  - $A, 0, B, 0$ ↙ r
  - $B, 1$ (observed 6 times)
  - $B, 0$
- What are $V(A), V(B)$?

$(s, a, r, s')$

$= (1-\alpha)V + \alpha(r + \gamma V(s'))$

MC estimate

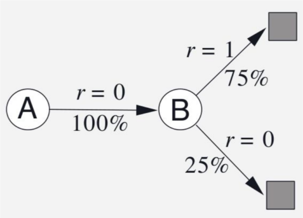
$V(B) = 3/4 = 6/8$

TD $V(B) = 6/8 = 3/4$

$0 + \gamma V(B)$

| | MC | TD |
|---|---|---|
| $V(A) =$ | $0$ | $3/4$ |

# AB Example: (Ex. 6.4, Sutton & Barto, 2018)



- Two states $A, B$ with $\gamma = 1$
- Given 8 episodes of experience:
  - $A, 0, B, 0$ ←
  - $B, 1$ (observed 6 times)
  - $B, 0$
- $V(B) = 0.75$ by TD or MC
- What about $V(A)$?

B r=1 6

B r= 0 2

# Batch MC and TD: Converges

- Monte Carlo in batch setting converges to min MSE (mean squared error)
  - Minimize loss with respect to observed returns
  - In AB example, $V(A) = 0$
- TD(0) converges to DP policy $V^\pi$ for the MDP with the maximum likelihood model estimates
  - Maximum likelihood Markov decision process model

$P(B|A) = 1$

$$\hat{P}(s'|s,a) = \frac{1}{N(s,a)} \sum_{k=1}^{K} \sum_{t=1}^{L_k-1} \mathbb{1}(s_{k,t} = s, a_{k,t} = a, s_{k,t+1} = s')$$

$$\hat{r}(s,a) = \frac{1}{N(s,a)} \sum_{k=1}^{K} \sum_{t=1}^{L_k-1} \mathbb{1}(s_{k,t} = s, a_{k,t} = a) r_{t,k}$$

$r(B) = 3/4$
$r(A) = 0$

  - Compute $V^\pi$ using this model
  - In AB example, $V(A) = 0.75$

# Some Important Properties to Evaluate Model-free Policy Evaluation Algorithms

- Data efficiency & Computational efficiency
- In simplest TD, use $(s, a, r, s')$ once to update $V(s)$
  - $O(1)$ operation per update
  - In an episode of length $L$, $O(L)$
- In MC have to wait till episode finishes, then also $O(L)$
- MC can be more data efficient than simple TD
- But TD exploits Markov structure
  - If in Markov domain, leveraging this is helpful

# Alternative: Certainty Equivalence $V^\pi$ MLE MDP Model Estimates

- Model-based option for policy evaluation without true models
- After each $(s, a, r, s')$ tuple
  - Recompute maximum likelihood MDP model for $(s, a)$

$$\hat{P}(s'|s,a) = \frac{1}{N(s,a)} \sum_{k=1}^{K} \sum_{t=1}^{L_k - 1} \mathbb{1}(s_{k,t} = s, a_{k,t} = a, s_{k,t+1} = s')$$

$$\hat{r}(s,a) = \frac{1}{N(s,a)} \sum_{k=1}^{K} \sum_{t=1}^{L_k - 1} \mathbb{1}(s_{k,t} = s, a_{k,t} = a) r_{t,k}$$

  - Compute $V^\pi$ using MLE MDP [3] (e.g. see method from lecture 2)

[3] Requires initializing for all $(s, a)$ pairs

# Alternative: Certainty Equivalence $V^\pi$ MLE MDP Model Estimates

- Model-based option for policy evaluation without true models
- After each $(s, a, r, s')$ tuple
  - Recompute maximum likelihood MDP model for $(s, a)$

$$\hat{P}(s'|s,a) = \frac{1}{N(s,a)} \sum_{k=1}^{K} \sum_{t=1}^{L_k - 1} 1(s_{k,t} = s, a_{k,t} = a, s_{k,t+1} = s')$$

$$\hat{r}(s,a) = \frac{1}{N(s,a)} \sum_{k=1}^{K} \sum_{t=1}^{L_k - 1} 1(s_{k,t} = s, a_{k,t} = a) r_{t,k}$$

  - Compute $V^\pi$ using MLE MDP [4] (e.g. see method from lecture 2)
- Cost: Updating MLE model and MDP planning at each update ($O(|S|^3)$ for analytic matrix solution, $O(|S|^2|A|)$ for iterative methods)
- Very data efficient and very computationally expensive
- Consistent
- Can also easily be used for off-policy evaluation

| $s_1$ | $s_2$ | $s_3$ | $s_4$ | $s_5$ | $s_6$ | $s_7$ |
|---|---|---|---|---|---|---|
| $R(s_1) = +1$<br>*Okay*<br>*Field Site* | $R(s_2) = 0$ | $R(s_3) = 0$ | $R(s_4) = 0$<br> | $R(s_5) = 0$ | $R(s_6) = 0$ | $R(s_7) = +10$<br>*Fantastic*<br>*Field Site* |

- Mars rover: R = [ 1 0 0 0 0 0 +10] for any action
- $\pi(s) = a_1 \; \forall s$, $\gamma = 1$. any action from $s_1$ and $s_7$ terminates episode
- Trajectory = $(s_3, a_1, 0, s_2, a_1, 0, s_2, a_1, 0, s_1, a_1, 1, \text{terminal})$
- First visit MC estimate of $V$ of each state? [1 1 1 0 0 0 0]
- Every visit MC estimate of $V$ of $s_2$? 1
- TD estimate of all states (init at 0) with $\alpha = 1$ is [1 0 0 0 0 0 0]
- What is the certainty equivalent estimate?

# Some Important Properties to Evaluate Policy Evaluation Algorithms

- Robustness to Markov assumption
- Bias/variance characteristics
- Data efficiency
- Computational efficiency

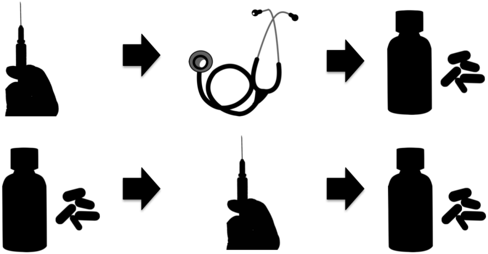
# Summary: Policy Evaluation

- Dynamic Programming
- Monte Carlo policy evaluation
  - Policy evaluation when we don't have a model of how the world works
    - Given on policy samples
    - Given off policy samples
- Temporal Difference (TD)
- Metrics to evaluate and compare algorithms

# This Lecture: Policy Evaluation

- Estimating the expected return of a particular policy if don't have access to true MDP models
- Dynamic programming
- Monte Carlo policy evaluation
    - Policy evaluation when don't have a model of how the world work
        - Given on-policy samples
        - **Given off-policy samples**
- Temporal Difference (TD)
- Metrics to evaluate and compare algorithms

# MC Off Policy Evaluation



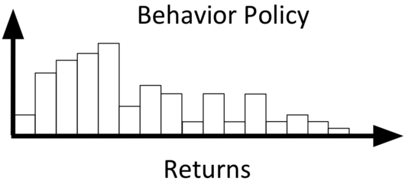
- Sometimes trying actions out is costly or high stakes
- Would like to use old data about policy decisions and their outcomes to estimate the potential value of an alternate policy

# Monte Carlo (MC) Off Policy Evaluation

- Aim: estimate value of policy $\pi_1$, $V^{\pi_1}(s)$, given episodes generated under behavior policy $\pi_2$
  - $s_1, a_1, r_1, s_2, a_2, r_2, \dots$ where the actions are sampled from $\pi_2$
- $G_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 r_{t+3} + \cdots$ in MDP $M$ under policy $\pi$
- $V^\pi(s) = \mathbb{E}_\pi[G_t | s_t = s]$
- Have data from a different policy, behavior policy $\pi_2$
- If $\pi_2$ is stochastic, can often use it to estimate the value of an alternate policy (formal conditions to follow)
- Again, no requirement that have a model nor that state is Markov

# Monte Carlo (MC) Off Policy Evaluation: Distribution Mismatch

- Distribution of episodes & resulting returns differs between policies



# Importance Sampling

- Goal: estimate the expected value of a function $f(x)$ under some probability distribution $p(x)$, $\mathbb{E}_{x \sim p}[f(x)]$
- Have data $x_1, x_2, \ldots, x_n$ sampled from distribution $q(s)$
- Under a few assumptions, we can use samples to obtain an unbiased estimate of $\mathbb{E}_{x \sim q}[f(x)]$

$$\mathbb{E}_{x \sim q}[f(x)] = \int_x q(x) f(x)$$

# Importance Sampling (IS) for Policy Evaluation

- Let $h_j$ be episode $j$ (history) of states, actions and rewards

$$h_j = (s_{j,1}, a_{j,1}, r_{j,1}, s_{j,2}, a_{j,2}, r_{j,2}, \ldots, s_{j,L_j(terminal)})$$

# Importance Sampling (IS) for Policy Evaluation

- Let $h_j$ be episode $j$ (history) of states, actions and rewards

$$h_j = (s_{j,1}, a_{j,1}, r_{j,1}, s_{j,2}, a_{j,2}, r_{j,2}, \dots, s_{j,L_j(terminal)})$$

$$\begin{aligned} p(h_j|\pi, s = s_{j,1}) =& p(a_{j,1}|s_{j,1})p(r_{j,1}|s_{j,1}, a_{j,1})p(s_{j,2}|s_{j,1}, a_{j,1}) \\ & p(a_{j,2}|s_{j,2})p(r_{j,2}|s_{j,2}, a_{j,2})p(s_{j,3}|s_{j,2}, a_{j,2}) \dots \\ =& \prod_{t=1}^{L_j-1} p(a_{j,t}|s_{j,t})p(r_{j,t}|s_{j,t}, a_{j,t})p(s_{j,t+1}|s_{j,t}, a_{j,t}) \\ =& \prod_{t=1}^{L_j-1} \pi(a_{j,t}|s_{j,t})p(r_{j,t}|s_{j,t}, a_{j,t})p(s_{j,t+1}|s_{j,t}, a_{j,t}) \end{aligned}$$

# Importance Sampling (IS) for Policy Evaluation

- Let $h_j$ be episode $j$ (history) of states, actions and rewards, where the actions are sampled from $\pi_2$

$$h_j = (s_{j,1}, a_{j,1}, r_{j,1}, s_{j,2}, a_{j,2}, r_{j,2}, \ldots, s_{j,L_j(terminal)})$$

$$V^{\pi_1}(s) \approx \sum_{j=1}^{n} \frac{p(h_j|\pi_1, s)}{p(h_j|\pi_2, s)} G(h_j)$$

# Importance Sampling for Policy Evaluation

- Aim: estimate $V^{\pi_1}(s)$ given episodes generated under policy $\pi_2$
  - $s_1, a_1, r_1, s_2, a_2, r_2, \ldots$ where the actions are sampled from $\pi_2$
- Have access to $G_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 r_{t+3} + \cdots$ in MDP $M$ under policy $\pi_2$
- Want $V^{\pi_1}(s) = \mathbb{E}_{\pi_1}[G_t | s_t = s]$
- IS = Monte Carlo estimate given off policy data
- Model-free method
- Does not require Markov assumption
- Under some assumptions, unbiased & consistent estimator of $V^{\pi_1}$
- Can be used when agent is interacting with environment to estimate value of policies different than agent's control policy
- More later this quarter about batch learning

# Today's Plan

- Last Time:
  - Markov reward / decision processes
  - Policy evaluation & control when have true model (of how the world works)
- Today
  - Policy evaluation when don't have a model of how the world works
- **Next Time**:
  - **Control when don't have a model of how the world works**

# Stanford·CS234 | Reinforcement Learning (2019)

## CS234 (2019)· 课程资料包 @ShowMeAI

**视频**
中英双语字幕

**课件**
一键打包下载

**笔记**
官方笔记翻译

**代码**
作业项目解析

**视频 ·B 站 [ 扫码或点击链接 ]**
https://www.bilibili.com/video/BV1H64y1x7GH

**课件 & 代码 · 博客 [ 扫码或点击链接 ]**
http://blog.showmeai.tech/cs234

斯坦福 reinforcement learning 马尔可夫决策过程 DQN 强化学习 值函数方法 policy gradient Q-learning 梯度策略 Model free 蒙特卡洛搜索树

Awesome AI Courses Notes Cheatsheets 是 ShowMeAI 资料库的分支系列，覆盖最具知名度的 **TOP20+** 门 AI 课程，旨在为读者和学习者提供一整套高品质中文学习笔记和速查表。

**点击**课程名称，跳转至课程**资料包**页面，**一键下载**课程全部资料！

| 机器学习 | 深度学习 | 自然语言处理 | 计算机视觉 |
|---|---|---|---|
| Stanford · CS229 | Stanford · CS230 | Stanford · CS224n | Stanford · CS231n |
| **# Awesome AI Courses Notes Cheatsheets· 持续更新中** | | | |
| **知识图谱** | **图机器学习** | **深度强化学习** | **自动驾驶** |
| Stanford · CS520 | Stanford · CS224W | UCBerkeley · CS285 | MIT · 6.S094 |

**微信公众号**

资料下载方式 2：扫码点击底部菜单栏

称为 **AI 内容创作者？** 回复 [ 添砖加瓦 ]