Week 5

Policy Iteration (Page 1-9)
Value Iteration (Page 10-18)
Model-Free RL (Page 20-31)

Prof. Subrahmanya Swamy

Iterative Policy Evaluation

- How to find V_π of a given policy π ?
- Iteratively apply the BE equation

$$V_{k+1}(s) = R_s^\pi + \sum_{s'} P_{ss'}^\pi V_k(s')$$

V_1 V_0

V_0

V_1

$$\boxed{\text{Repeat till } V_{k+1} = V_k \implies V_k = V_\pi}$$

Grid Example: Policy Evaluation

A	B
C	G 0

- **Deterministic** state transitions
- $R_t = -1$ on all transitions
- Terminal state value $V_\pi(G) = 0$
- Discount factor $\gamma = 1$
- **Uniform** Random Policy π

V_π

Uniform Policy Dynamics:

$$P^\pi_{A,A} = \frac{1}{2}, \quad P^\pi_{A,B} = \frac{1}{4}, \quad P^\pi_{A,C} = \frac{1}{4}$$

$$P^\pi_{B,A} = \frac{1}{4}, \quad P^\pi_{B,B} = \frac{1}{2}, \quad P^\pi_{B,G} = \frac{1}{4}$$

$$P^\pi_{C,A} = \frac{1}{4}, \quad P^\pi_{C,G} = \frac{1}{4}, \quad P^\pi_{C,C} = \frac{1}{2}$$

Iterative Policy Evaluation:

$$V_{k+1}(s) = R^\pi_s + \sum_{s'} P^\pi_{ss'} V_k(s')$$

$$\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \rightarrow \begin{cases} V_1(A) = -1 + \frac{1}{2}V_0(A) + \frac{1}{4}V_0(B) + \frac{1}{4}V_0(C) \\ V_1(B) = -1 + \frac{1}{4}V_0(A) + \frac{1}{2}V_0(B) + \frac{1}{4}V_0(G) \\ V_1(C) = -1 + \frac{1}{4}V_0(A) + \frac{1}{4}V_0(G) + \frac{1}{2}V_0(C) \end{cases} \rightarrow \begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix} \xrightarrow[k=1]{\text{update}} \begin{bmatrix} -2 \\ -1.75 \\ -1.75 \end{bmatrix} \xrightarrow[k=2]{\text{update}} \cdots$$

V_0 $\quad V_1 \quad V_2$

Initial estimate

Update at k=0

Repeat till $V_{k+1} = V_k$ $\implies$ $V_k = V_\pi$

Policy Evaluation gives V_{π}

How to find Optimal Policy V^* ?

π^*

$$\Downarrow$$

Policy Iteration

Policy Iteration Algorithm

$$\pi_0 \rightarrow V_{\pi_0} \rightarrow \pi_1 \rightarrow V_{\pi_1} \rightarrow \pi_2 \rightarrow V_{\pi_2} \rightarrow \ldots$$

Uniform Random

I.P.E.

P.I.

P.E.

Policy Evaluation

Policy Improvement

Repeat till

$$\pi_{k+1} = \pi_k$$

$$\Downarrow$$

$$\pi_k = \pi_*$$

- Policy Evaluation: Iteratively apply BE equation $V_{k+1}(s) = R_s^\pi + \sum_{s'} P_{ss'}^\pi V_k(s')$
- Policy Improvement: $\pi_{i+1}(s) \coloneqq argmax_a \; R_s^a + \sum_{s'} P_{ss'}^a V_{\pi_i}(s')$

optimal substructure

$\pi_1(s)$ — a_1, a_2, a_3

$R_s^a + \sum_{s'} V^{\pi_0}(s')$

Policy Iteration Algorithm

Repeat till

$\pi_{k+1} = \pi_k$

$\Downarrow$

$\pi_k = \pi_*$

$$\pi_0 \xrightarrow{} V_{\pi_0} \rightarrow \pi_1 \rightarrow V_{\pi_1} \rightarrow \pi_2 \rightarrow V_{\pi_2} \rightarrow \dots$$

Policy Evaluation

Policy Improvement

- Policy Evaluation: Iteratively apply BE equation $V_{k+1}(s) = R_s^\pi + \sum_{s'} P_{ss'}^\pi V_k(s')$
- Policy Improvement: $\pi_{i+1}(s) \coloneqq argmax_a \; R_s^a + \sum_{s'} P_{ss'}^a V_{\pi_i}(s')$
- $\boxed{V_{\pi_{i+1}} \geq V_{\pi_i}}$ due to Policy Improvement Theorem

Grid Example: Policy Iteration

A	B
C	G

Deterministic Transitions

$$\pi_0 \to V_{\pi_0} \to \pi_1 \to V_{\pi_1} \to \pi_2 \to V_{\pi_2} \to \dots$$

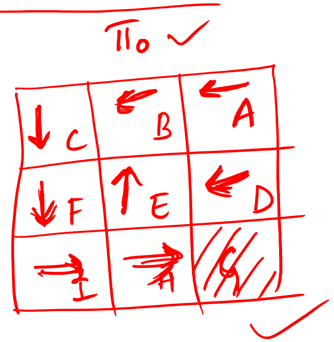
Policy Evaluation

Policy Improvement

-1

$\gamma = 1$

π_0 ✓



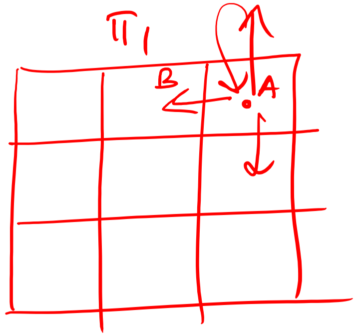
P.E. $\longrightarrow$

V_{π_0}

-4	-5	-6
-3	-6	-7
-2	-1	0

$\longrightarrow$

π_1



$$\pi_1(s) = \underset{a}{\operatorname{argmax}} \left\{ R_s^a + \gamma \sum_{s'} P_{ss'}^a \, V_{\pi_0}(s') \right\}$$

$$\pi_1(A) = \operatorname{argmax} \begin{cases} \text{Up}: & -1 + V_{\pi_0}(A) \\ \text{Left}: & -1 + V_{\pi_0}(B) \\ \text{Down}: & -1 + V_{\pi_0}(D) \\ \text{Right}: & -1 + V_{\pi_0}(A) \end{cases} = \underset{a}{\operatorname{argmax}} \begin{cases} U: & -1-6 \\ L: & -1-5 \; ✓ \\ D: & -1-7 \\ R: & -1-6 \end{cases}$$

$\Rightarrow \pi_1(A) = \text{Left}$

Grid Example: Policy Iteration

A	B
C	G

Deterministic Transitions

-1

$\gamma = 1$

$$\pi_0 \xrightarrow{\text{Policy Evaluation}} V_{\pi_0} \xrightarrow{\text{Policy Improvement}} \pi_1 \rightarrow V_{\pi_1} \rightarrow \pi_2 \rightarrow V_{\pi_2} \rightarrow \dots$$

π_0

↓ C	← B	← A
↓ F	↑ E	← D
→ I	→ H	G

P.E. →

V_{π_0}

-4	-5	-6
-3	-6	-7
-2	-1	0

→

π_1

↓	←	←
↓	↓	↓
→	→	

P.E. →

V_{π_1}

-4	-5	-6
-3	-2	-1
-2	-1	0

→

π_2

↓	↓	↓
↓	↓	↓
→	→	

↓

V_{π_2}

↓

π_3

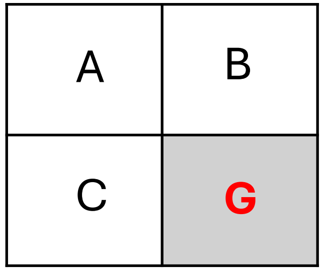
$$\pi_1(s) = \arg\max \left\{ R_s^a + \gamma \sum_{s'} P_{ss'}^a V_{\pi_0}(s') \right\}$$

$$\pi_1(E) = \arg\max \begin{Bmatrix} L: & -1 + V_{\pi_0}(F) \\ R: & -1 + V_{\pi_0}(D) \\ U: & -1 + V_{\pi_0}(B) \\ D: & -1 + V_{\pi_0}(H) \end{Bmatrix} = \arg\max_a \begin{Bmatrix} L: & -1-3 \\ R: & -1-7 \\ U: & -1-5 \\ D: & -1-1 \end{Bmatrix}$$

Exercise

- Take π_0 as a uniform random policy
- Apply Policy iteration algorithm
- Show the sequence of policies that we get

$\pi_0 \rightarrow \pi_1 \rightarrow \cdots$



Deterministic Transitions

Value Iteration

Value Iteration

Bellman Optimality (BO)

$$V^*(s) = \max_{a} R_s^a + \sum_{s'} P_{ss'}^a V^*(s')$$ (Optimal Substructure)

BE → ① V_π

Value Iteration

Iteratively apply BO equation till $V_{k+1} = V_k$ ⇒ $V_k = V^*$

$$V_{k+1}(s) = \max_{a} R_s^a + \sum_{s'} P_{ss'}^a V_k(s')$$

$V_0 \xrightarrow[\text{update}]{\text{BO}} V_1 \xrightarrow[\text{update}]{\text{BO}} V_2 \ldots$

Optimal Policy from V^*

$$\pi^*(s) = \arg\max_{a} R_s^a + \sum_{s'} P_{ss'}^a V^*(s')$$

a_1

a_2

a_3

Implementation Details

$V_{k+1} \approx V_k$

- Issue: Takes a very long time to see $V_{k+1} = V_k$
- Solution: Stop when $||V_{k+1} - V_k||_\infty$ is small enough.

  $\varepsilon = 0.01$
- Typically, max-norm is used

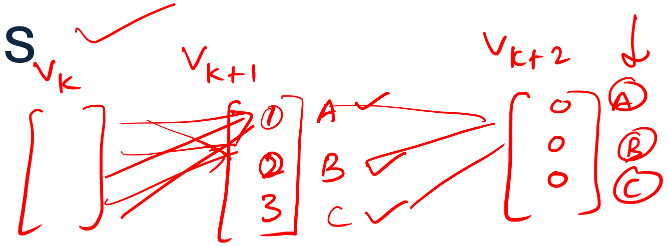
$$\begin{array}{cc} V_2 & V_1 \\ \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} & \begin{bmatrix} 3 \\ 4 \\ 7 \end{bmatrix} \end{array}$$

$$||V_2 - V_1||_\infty = \max\{|1-3|, |2-4|, |3-7|\}$$

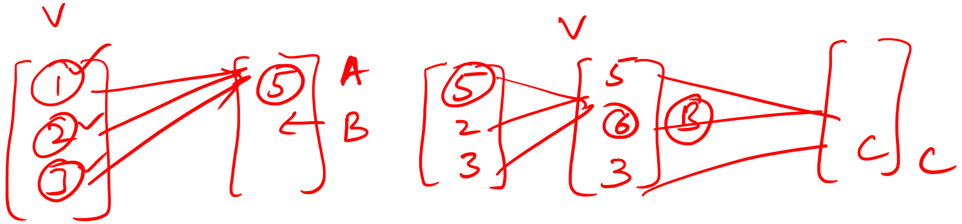
$$= 4$$

Implementation Details

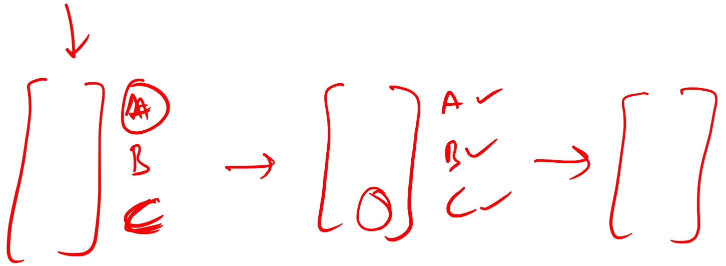
- Synchronous updates



- In-place updates



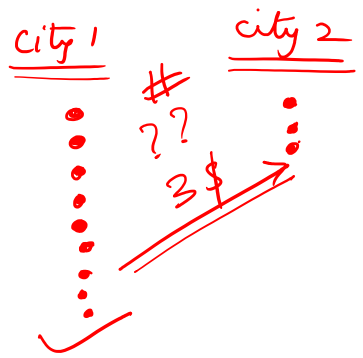
- Asynchronous updates



$\Rightarrow |V_{k+1}(A) - V_k(A)| = 10$ $\quad V_{k+2}$

$\Rightarrow |V_{k+1}(B) - V_k(B)| = 0.01$

Car rental Example



- A car rental company operates in two cities
- Customers arrive at these cities and rent a car for $10. If a customer arrives when cars are unavailable: Business Loss
- #car requests and returns are Poisson random variables
- At most, 20 cars can be parked at each location
- Upto 5 Cars be transferred overnight between cities at 3$

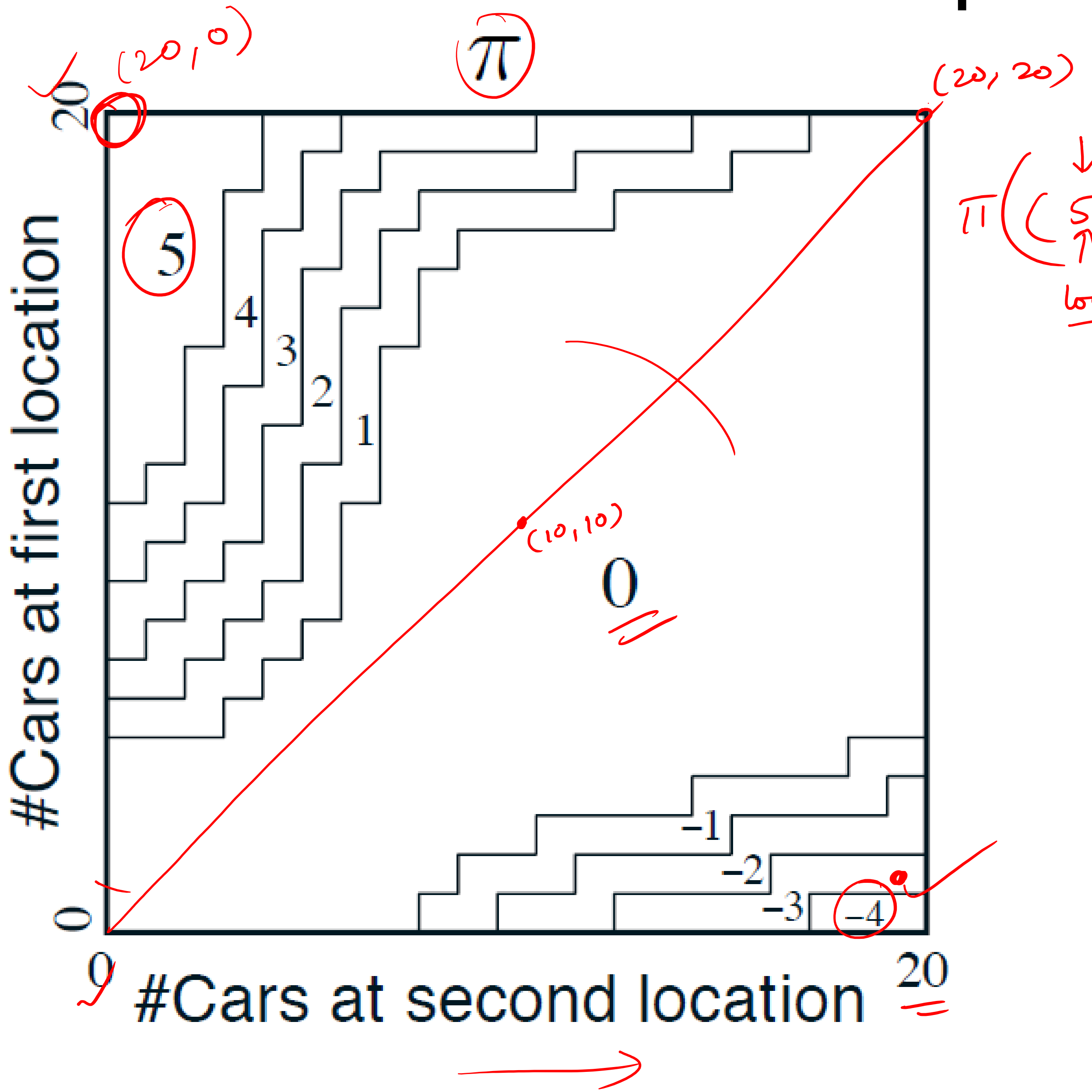
MDP

Car Rental Example

"Sutton & Barto"

Example 4.2: Jack's Car Rental Jack manages two locations for a nationwide car rental company. Each day, some number of customers arrive at each location to rent cars. If Jack has a car available, he rents it out and is credited \$10 by the national company. If he is out of cars at that location, then the business is lost. Cars become available for renting the day after they are returned. To help ensure that cars are available where they are needed, Jack can move them between the two locations overnight, at a cost of \$2 per car moved. We assume that the number of cars requested and returned at each location are Poisson random variables, meaning that the probability that the number is n is $\frac{\lambda^n}{n!}e^{-\lambda}$, where λ is the expected number. Suppose λ is 3 and 4 for rental requests at the first and second locations and 3 and 2 for returns. To simplify the problem slightly, we assume that there can be no more than 20 cars at each location (any additional cars are returned to the nationwide company, and thus disappear from the problem) and a maximum of five cars can be moved from one location to the other in one night. We take the discount rate to be $\gamma = 0.9$ and formulate this as a continuing finite MDP, where the time steps are days, the state is the number of cars at each location at the end of the day, and the actions are the net numbers of cars moved between the two locations overnight. Figure 4.2 shows the sequence of policies found by policy iteration starting from the policy that never moves any cars.

Car Rental Problem: An Example Policy



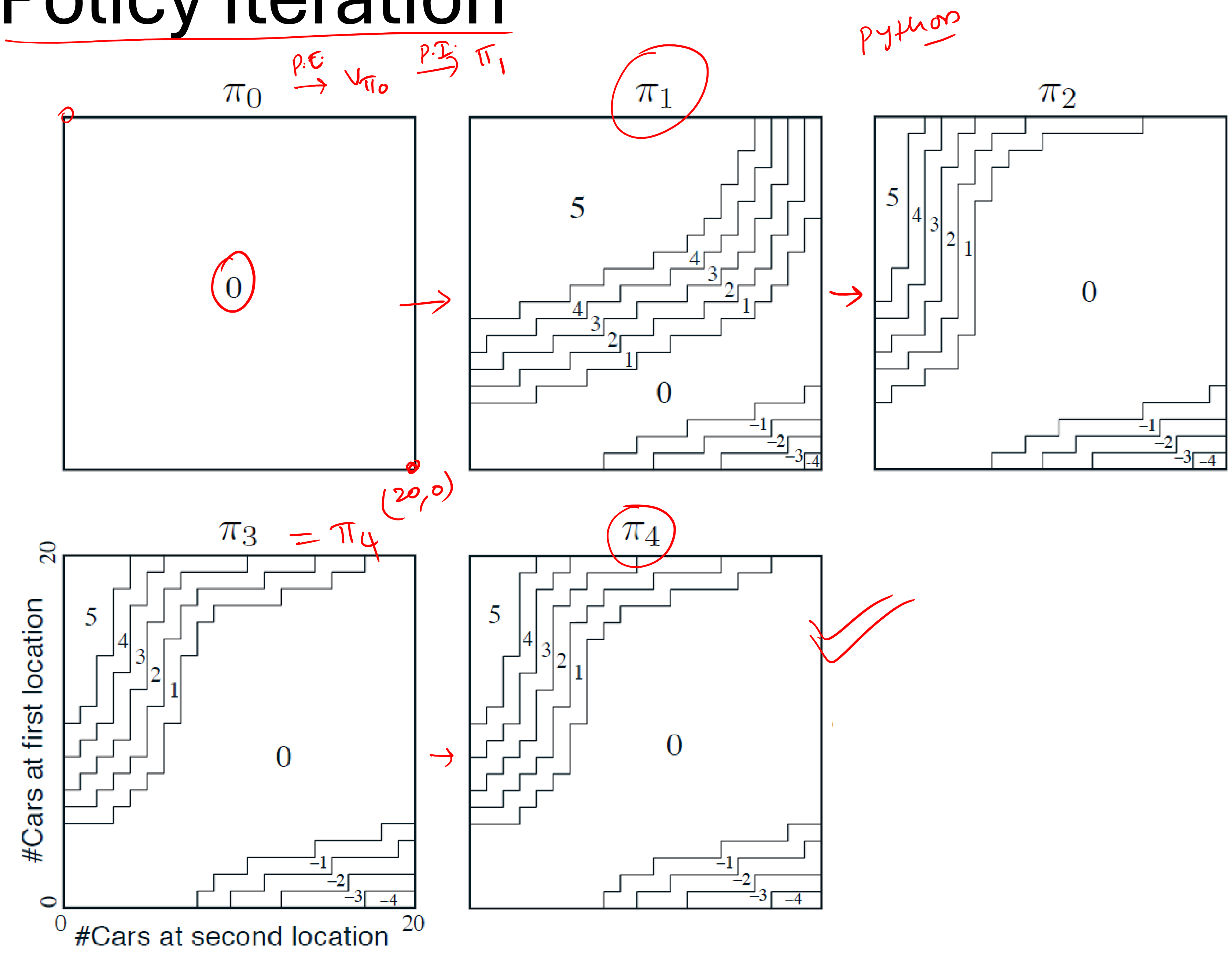
π((5, 10)) → +5 / +4 / −2

loc 1 → loc 2

loc 2 → loc 1

How many cars to transfer?

Policy Iteration



Model-Free RL

Prof. Subrahmanya Swamy

How to find V_π for a given π?

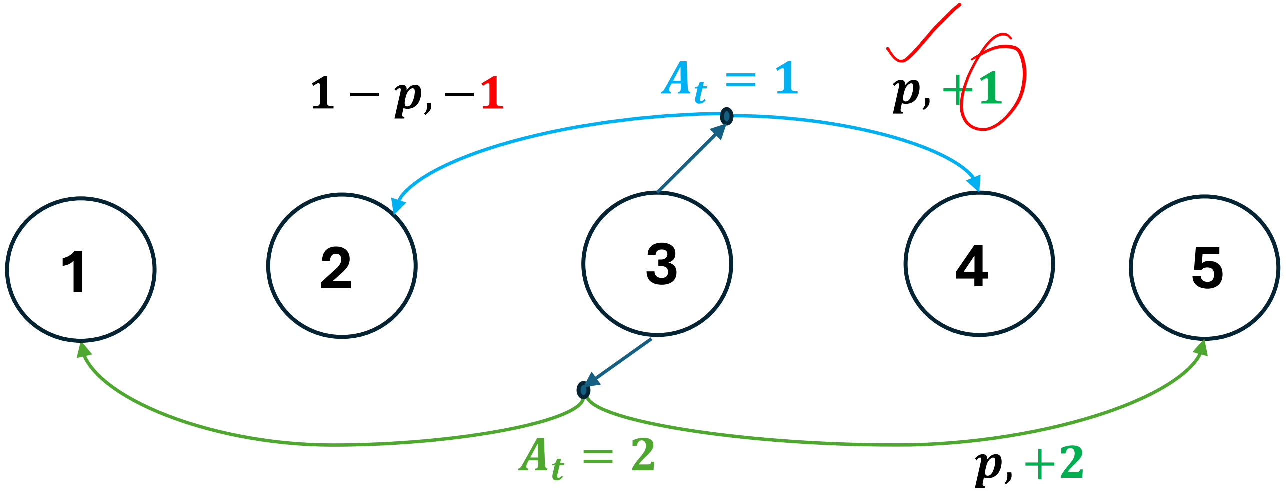
Policy Evaluation

$$V_{k+1}(s) = R_s^\pi + \sum_{s'} P_{ss'}^\pi V_k(s')$$

$V_{k+1} = V_k$.

$$R_s^\pi = \sum_a \pi(a \mid s)\, R_{ss'}^a \qquad P_{ss'}^\pi = \sum_a \pi(a \mid s)\, P_{ss'}^a$$

Requires complete knowledge of the environment: MDP model



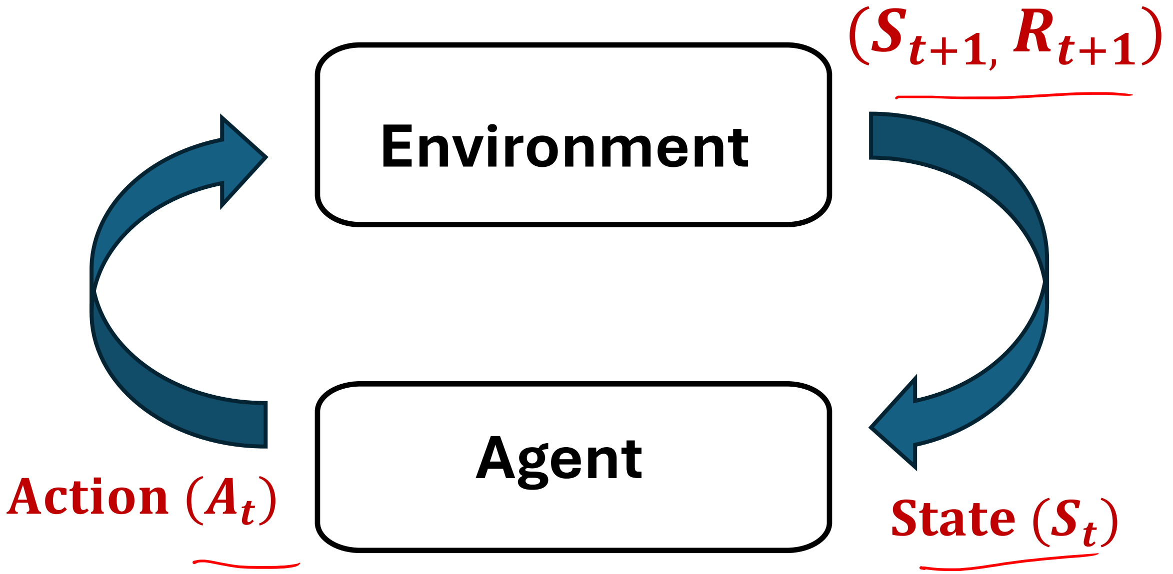
Model-Free RL: Unknown $R_s^a, P_{ss'}^a$

Task	Model Available	Model Unknown
Policy Evaluation V_π	Iterative Policy Evaluation	??
Optimal Policy π^*	Policy Iteration, Value Iteration	??

✓Monte-Carlo (MC)

Temporal Difference (TD)

Learn through real-time interaction with environment



Monte-Carlo (MC) method to estimate V_π

π.

Prediction

- $V_\pi(s) = \mathbb{E}_\pi[G_t \mid S_t = s]$

$\mu(a) = E[R^a]$

$R_1, R_2, \dots R_N$ $\quad \mu(a) \approx \frac{R_1 + R_2 + \dots R_N}{N}$

- Interact with the environment and generate multiple episodes of data
  - Episode 1: $S_0 = s, A_0 \sim \pi, R_1, S_1, A_1 \sim \pi, R_2, S_2, \dots, S_T$ $\to G^{(1)}$
  - Episode 2: $S_0 = s, A_0 \sim \pi, R_1, S_1, A_1 \sim \pi, R_2, S_2, \dots, S_T$ $\to G^{(2)}$
  - ...
  - ...

$V_\pi(s)$

$G^{(N)}$

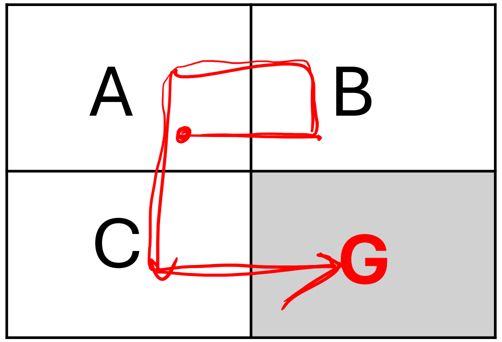
- Compute sample returns of each episode from state s
  - $G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \dots$

- $V_\pi(s) \approx$ sample avg of the returns ✓

Monte-Carlo: Grid Example

$V_\pi(s) = E_\pi[G_t \mid S_t = s]$

$S_0 = s$



Uniform Random Policy

$V_\pi(A)$

Sample Episode

$A, Right, -1, B, Left, -1, A, Down, -1, C, Right, -1, G$

$t=2$ $\quad$ $t=3$ $\quad$ $t=4$

1/4 R

1/4 L

-4 $\quad$ -2 $\quad$ -2

$V_\pi(A)$

$V_\pi(B) \to -3$

$V_\pi(C) \; ? \to -1$

-4 $\quad$ -2

MC- First visit

MC - Every-visit

MC First-Visit

- Two states: $\{A, B\}$

- Observed episodes:
  - A, 1, B, -2, B, 4, A, 0, B, -2 → Terminated
  - B, -1, B, 3, A, 2, B, 0, A , -3 → Terminated

- Observed returns
  - Episode 1: Return from first-visit of state A: $1 - 2 + 4 + 0 - 2 = 1$
  - Episode 1: Return from first-visit of state B: $-2 + 4 + 0 - 2 = 0$
  - Episode 2: Return from first-visit of state A: $2 + 0 - 3 = -1$
  - Episode 2: Return from first-visit of state B: $-1 + 3 + 2 + 0 - 3 = 1$

- MC estimates (average of observed returns):
  - $V(A) \approx \frac{1}{2}(1 - 1) = 0$
  - $V(B) \approx \frac{1}{2}(0 + 1) = \frac{1}{2}$

MC Every-Visit

- Two states: $\{A, B\}$

- Observed episodes:
  - A, 1, B, -2, B, 4, A, 0, B, -2 → Terminated
  - B, -1, B, 3, A, 2, B, 0, A , -3 → Terminated

- Observed returns
  - Episode 1: Returns from all-visits of state A: 2 return samples
  - Episode 1: Return from all-visits of state B: 3 return samples for B.
  - Episode 2: Return from all-visits of state A:
  - Episode 2: Return from all-visits of state B:

- MC estimates (average of observed returns):
  - $V(A) \approx$
  - $V(B) \approx$

MC: Disadvantages

- Requires G_t

$V(\cdot) \longrightarrow$ online fashion

- Can be obtained only after the episode terminates – Offline method

- Not suitable for Continuing MDPs

Episodic
Continuing ✓ S_T

- Alternate approach: Temporal Difference (TD) methods!

MC Incremental form

TD

$G^{(1)}, G^{(2)}, \ldots G^{(N)}$

Estimate after observing returns from N episode: $V_N(s)$

$$V_N(s) = \frac{G^{(1)} + G^{(2)} + \ldots G^{(N)}}{N}$$

Observed return in next episode: $G^{(N+1)}$

"N+1"th

$$V_{N+1}(s) = \frac{\text{"}G^{(1)} + G^{(2)} + \ldots G^{(N)} + G^{(N+1)}\text{"}}{N+1}$$

What is $V_{N+1}(s)$?

$$V_{N+1}(s) = \frac{N V_N(s) + G^{(N+1)}}{N+1}.$$

$$V_{N+1}(s) = \frac{V_N(s)(N+1) - V_N(s) + G^{(N+1)}}{N+1}$$

$$\rightarrow V_{N+1}(s) = V_N(s) + \frac{1}{N+1}\left[G^{(N+1)} - V_N(s)\right]$$

MC Incremental form

Update: $V_{N+1}(s) = V_N(s) + \frac{1}{N+1}(G_t^{(N+1)} - V_N(s))$

$$V_{new}(s) = V_{old}(s) + \alpha(G_t - V_{old}(s))$$

G_t

New Estimate

Old Estimate

Error between current sample and old estimate

Different Sample estimates => Different algorithms

MC

TD

Temporal Difference (TD) method for V_π

$$V_\pi(s) = \mathbb{E}_\pi[G_t \mid S_t = s]$$

$$= E_\pi\left[R_{t+1} + \gamma G_{t+1} \mid S_t = s\right]$$

$$= E_\pi\left[R_{t+1} \mid S_t = s\right] + \gamma E_\pi\left[G_{t+1} \mid S_t = s\right]$$

MC update: $V_{new}(s) = V_{old}(s) + \alpha\,(G_t - V_{old}(s))$

TD update: $V_{new}(s) = V_{old}(s) + \alpha\,(\,??\,- V_{old}(s))$

Temporal Difference (TD) method for V_π

$$
\begin{aligned}
V_\pi(s) &= \mathbb{E}_\pi[G_t \mid S_t = s] \\
&= \mathbb{E}_\pi[R_{t+1} + \gamma G_{t+1} \mid S_t = s] \\
&= \mathbb{E}_\pi[R_{t+1} \mid S_t = s] + \gamma \mathbb{E}_\pi[G_{t+1} \mid S_t = s] \\
&\approx R_{t+1} + \gamma V_\pi(S_{t+1})
\end{aligned}
$$

$S_t, A_t, R_{t+1}, S_{t+1}$

V_{old} s s'

$s, \dots\dots S_T \rightarrow G_t$

MC update: $V_{new}(s) = V_{old}(s) + \alpha\,(G_t - V_{old}(s))$

TD update: $V_{new}(s) = V_{old}(s) + \alpha\,(R_{t+1} + \gamma V_{old}(s') - V_{old}(s))$

MC

TD

V_π

"online"

"continuing"

π^*

How to find π^* in model-free setting?

$Q_\pi(s, a)$

Policy Iteration

$\pi_0 \rightarrow V_{\pi_0} \rightarrow \pi_1 \rightarrow V_{\pi_1} \rightarrow \pi_2 \rightarrow V_{\pi_2} \rightarrow \ldots$

Policy Evaluation

TD | MC

Policy Improvement

MC
TD

Repeat till
$\pi_{k+1} = \pi_k$
$\Downarrow$
$\pi_k = \pi_*$

- Policy Evaluation: Iteratively apply BE equation $V_{k+1}(s) = R_s^\pi + \sum_{s'} P_{ss'}^\pi V_k(s')$
- Policy Improvement: $\pi_{i+1}(s) := argmax_a \; R_s^a + \sum_{s'} P_{ss'}^a V_{\pi_i}(s')$

Can we use Policy Iteration in a model-free setting?

- Replace PE with MC/TD-based V_π estimation
- What about Policy Improvement?