Reinforcement Learning

Prof. Volkan Cevher
volkan.cevher@epfl.ch

Lecture 1: Introduction + Dynamic Programming I

Laboratory for Information and Inference Systems (LIONS)
École Polytechnique Fédérale de Lausanne (EPFL)

EE-568 (Spring 2026)

lions@epfl

License Information for Reinforcement Learning (EE-568)

Acknowledgements

- Joint effort between ODI@ETHZ (Prof. Niao He) and lions@epfl (myself)
  - Anas Barakat, Batu Yardim, Jiawei Huang, Liang Zhang, Zebang Shen, Junchi Yang, Siqi Zhang, Yifan Hu and Adrian Müller.
  - Luca Viano, Igor Krawczuk, Ali Kavis, Ahmet Alacaoglu, Grigorios Chrysos, Pedro Abranches, Leello Dadi, Ali Ramezani, Stratis Skoulakis, Kimon Antonakopoulos, Fanghui Liu, Fabian Latorre, Thomas Pethick, Angeliki Kamoutsi, Yongtao Wu, Wanyun Xie, Elias Abad Rocamora, Zhenyu Zhu, Leyla Naz Candogan, Arshia Afzal, Andrej Janchevski, Frank Zhengqing Wu and Pol Puigdemont Plana.

Logistics

- **Credits:** 6
- **Lectures and exercises:** Thursdays 13:15-17:00 AAC231 + CM 1 4 (Overflow room)
- **Prerequisites:** Previous coursework in optimization, probability theory, and linear algebra is required (i.e., EE-556 Math of Data). Familiarity with deep learning and programming in python is useful.
- **Grading:** Attendance[1] (1pt), 3 Jupyter Notebooks (1pt each), Project (2pts) or Scribe (2pts, only PhD students may replace the project by writing lecture notes)
- **Moodle:** My courses > Genie electrique et electronique (EL) > Master
  > EE-568 Logistics & Course schedule & Learning materials
- **Details:** All details are explained in `https://go.epfl.ch/rl-moodle` (see Logistics & Course schedule)
- **TAs**: Leyla Naz Candogan (Head TA), Pol Puigdemont Plana, Ioannis Mavrothalassitis, Yongtao Wu, Wanyun Xie, Daniele Affinita, Mengxin Lin, Semih Zaman, Natasa Jovanovic, Antoine Pierre Julien Auberger, Sean Park, Fatih Selim Yilmaz, Giovanni Capano.

[1] We need to see evidence through Jupyter notebooks and the project that you are participating. No need to come physically to the class.

Logistics for online teaching

- **Moodle:** `https://go.epfl.ch/rl-moodle`
- **Zoom:** `https://go.epfl.ch/rl-zoom`
- **Switchtube:** `https://go.epfl.ch/rl-lectures`

Project guidelines

- You can choose *one* of the following options:

1. **Theory project:**
   - Read 3 theory papers in an active RL research area (we will provide pointers).
   - Summarize them, explaining which problems are still open (and maybe solve them!).
2. **Practical project:**
   - Either implementing existing algorithms in new environments or
   - Improve existing algorithms on common environments.
   - Practical projects are in cooperation with EPFL labs. We will provide a list of labs you are allowed to reach out to.

- In May, there will be a final report (1.5pt) and a poster presentation (0.5pt) of your project.
- You will work in groups of three people (both theory & practice). Registration opens on moodle soon.

- If and only if you are a PhD student, you may choose the following option instead:

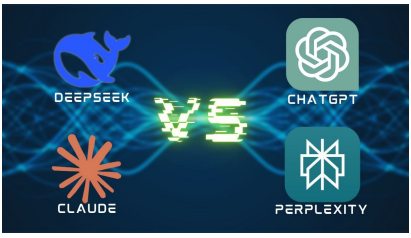
3. **Scribe:** Write lecture notes for a lecture assigned to you with a template we provide (2pts).

- For each option 1–3, please check moodle for the detailed and binding guidelines.

A paradigm shift in machine learning (ML) applications

- Self driving, industry automation, robotic manipulation, trading and finance, reasoning...

https://neptune.ai/blog/reinforcement-learning-applications



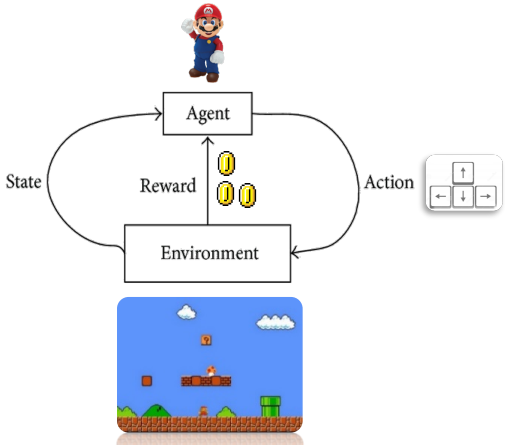
https://medium.com/nextgenllm/
deepseek-vs-chatgpt-vs-claude-comparison-354bb8487d63

What is reinforcement learning (RL)?

- Classical definitions:
  - Sutton and Barto: Reinforcement learning is learning what to do – how to map situations to actions – so as to maximize a numerical reward signal.
  - WIKIPEDIA: Reinforcement learning is an area of machine learning concerned with how intelligent agents ought to take actions in an environment in order to maximize the notion of cumulative reward.

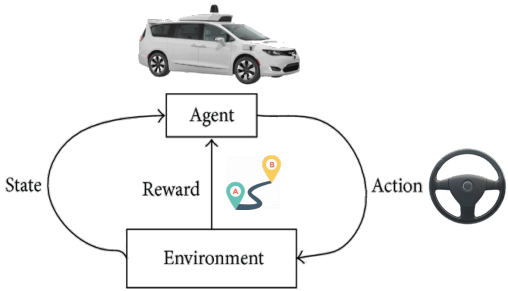
A common theme in RL

- An agent learns to act by interacting with an uncertain environment



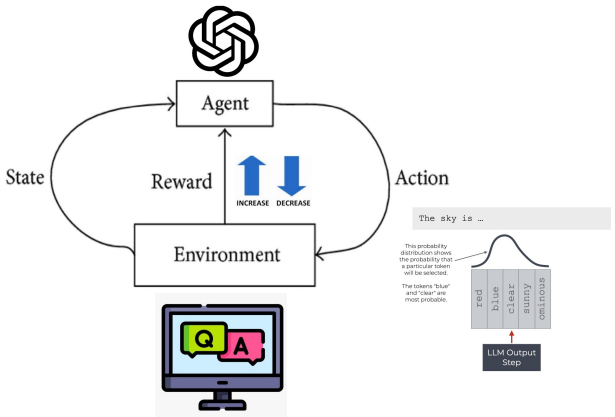
A common theme in RL

- An agent learns to act by interacting with an uncertain environment



A common theme in RL

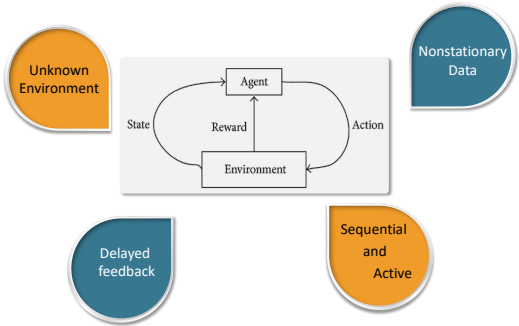
- An agent learns to act by interacting with an uncertain environment



Source: https://x.com/Josh_Ebner/status/1765108810539024707

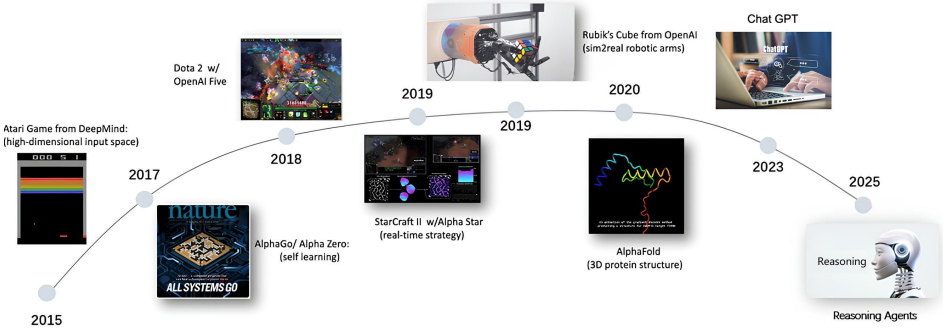
A common theme in RL

- An agent learns to act by interacting with an uncertain environment



Remarkable progress on ML applications

- Which one is not due to RL?



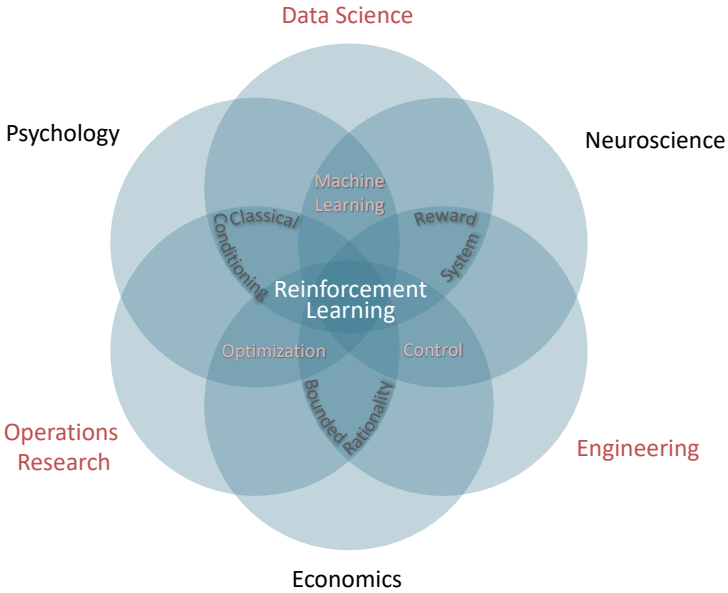
Perceptions of RL

- Roughly speaking...
  - For EE, it is control theory *mutatis mutandis*:

    control $\rightarrow$ action

    controller $\rightarrow$ agent or policy

    system or plant $\rightarrow$ environment
  - For CS, it is an ML paradigm along with supervised and unsupervised learning.
  - For others?


Data Science
Psychology
Neuroscience
Machine Learning
Classical Conditioning
Reward System
Reinforcement Learning
Optimization
Control
Bounded Rationality
Operations Research
Engineering
Economics

Challenges to RL

The New York Times

One Giant Step for a Chess-Playing Machine

The stunning success of AlphaZero, a deep-learning algorithm, heralds a new age of insight — one that, for humans, may not last long.

What is frustrating about machine learning, however, is that the algorithms can't articulate what they're thinking. We don't know why they work, so we don't know if they can be trusted. AlphaZero gives every appearance of having discovered some important principles about chess, but it can't share that understanding with us. Not yet, at least. As human beings, we want more than answers. We want insight. This is going to be a source of tension in our interactions with computers from now on.

- Theoretical foundations are more important than ever.

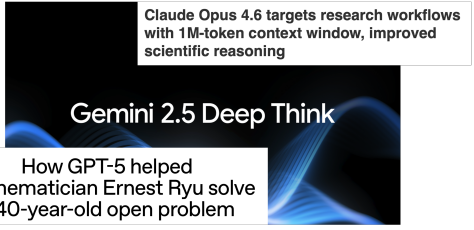
Challenges to RL

The New York Times

One Giant Step for a Chess-Playing Machine

The stunning success of AlphaZero, a deep-learning algorithm, heralds a new age of insight — one that, for humans, may not last long.

What is frustrating about machine learning, however, is that the algorithms can't articulate what they're thinking. We don't know why they work, so we don't know if they can be trusted. AlphaZero gives every appearance of having discovered some important principles about chess, but it can't share that understanding with us. Not yet, at least. As human beings, we want more than answers. We want insight. This is going to be a source of tension in our interactions with computers from now on.



- Theoretical foundations are more important than ever.
- Common challenges with ML: Robustness, interpretability, scalability, reproducibility, reward, etc.

Perceptions of our RL course (EE-568)

- Why are you taking this course?
  - Learn the basics of RL
  - Gain hands-on experience with RL implementations
  - Apply RL to my research
  - Might be useful for my future job
  - Just need the credits
  - Other reasons?
  - Let us know `https://go.epfl.ch/rl-expectations-2026`

What are our learning objectives?

- By the end of the course, participants will be able to
  - Define the key features of RL that distinguishes it from standard ML
  - Identify the strengths and limitations of various RL algorithms
  - Understand the theoretical properties of RL algorithms
  - Recognize the common, connecting boundary of optimization and RL
  - Formulate and solve sequential decision-making problems by applying relevant RL tools
  - Generalize or discover "new" applications, algorithms, or theories of RL towards conducting research

What EE-568 is really about: Theory and methods

Simplex (Dantzig)

Stochastic Approximation (Robbins & Monro)

Interior Point Method (Karmarker, Nemirovski, etc.)

Accelerated GD (Nesterov)

Boom of SGD and first-order methods

Kingma and Ba, Adam (ICLR, 2015)

Linear Programming

Stochastic Programming

Convex Programming

Stochastic Optimization

Nonconvex Optimization

1940s → 1950s → 1980s → 2000s → 2020s

Dynamic Programming

Value-based Methods

Policy Optimization

Deep Reinforcement Learning

DeepMind, DQN (Nature, 2015)

Boom of policy gradient methods

Policy Iteration (Howard)

Linear Programming

Q-Learning (Watkins)

TD Learning & Actor-critic (Sutton, Barto, etc.)

What EE-568 is *not* really about: Product development

- The following important topics are beyond the scope of this course:
  - Coding tricks and super practical implementations of RL
  - Product development of RL in real-world
  - RL engineering
  - Physically building autonomous robots
  - Physically building autonomous driving systems
  - Building GPT-5.2-scale systems with RLHF

Should I take this course?

- This course is right for you, if you
  - want to understand the RL foundations
  - have interest in performing RL research
  - have strong math background
  - want to gain hands-on experience with RL
- This course may not be right for you, if you
  - **only** want to gain hands-on experience with RL
  - are not interested in formulating RL in applications
  - have only basic math background
  - want to develop deep learning expertise

What is left for the course?

- A preview of the course
  - Dynamic Programming
    - Value Iteration
    - Policy Iteration
    - Monte Carlo Methods
    - TD, SARSA, Q-learning
  - Linear Programming
    - Primal-Dual RL, REPS, Proximal Point
    - Applications to offline RL
  - Policy-based RL
    - Policy Gradient Method
    - Natural Policy Gradient Method
    - TRPO and PPO
  - Imitation Learning and Inverse RL
    - Behavior Cloning, GAIL
    - Interactive IL (DAgger, SMILe)
    - Max Margin and Max Entropy IRL
  - Deep and Robust RL
    - Deep Q Network and Extensions
    - Deep Actor-Critic (A3C, DDPG, TD3)
    - Robust DDPG/TD3
  - Alignment and Reasoning with RL
    - Language Models
    - RL from Human Feedback
    - Reasoning

Theory	Bellman Equations	Stochastic Approximation
	Policy Gradient Theorems	Optimization and Game Theory
	Performance Difference Lemma	Convergence Analysis

What's beyond?

- Episodic RL
- Strategic Exploration in RL
- Batch and Offline RL
- Safety in RL
- Multi-task RL
- Preference-based RL
- Causal RL
- Partially Observable Markov Decision Process (POMDP)
- ....

Where to go from here?

- Upcoming events:
  - Mannheim Workshop on Reinforcement Learning 2026
    https://www.wim.uni-mannheim.de/doering/conferences/rl-2026
  - RL Conference, 2026
    https://rl-conference.cc
  - Workshops at NeurIPS, ICML, ICLR, AAMAS, etc. (TBA)
- Recent workshops:
  - Foundations of RL and Control: Connections and New Perspectives, ICML 2024
    https://rl-control-theory.github.io
  - Aligning Reinforcement Learning Experimentalists and Theorists, ICML 2024
    https://arlet-workshop.github.io
  - Workshop on Open-World Agents, NeurIPS 2024
    https://sites.google.com/view/open-world-agents/home

Where to go from here? (cont'd)

- Seminars:
  - RL Theory Virtual Seminar:
    https://sites.google.com/view/rltheoryseminars/
  - Simons Institute Theory of RL:
    https://simons.berkeley.edu/programs/theory-reinforcement-learning
  - Simons Institute Learning and Games:
    https://simons.berkeley.edu/programs/games2022/workshops#simons-tabs

Questions

That's it! Any questions?

Questions

Let's start! Dynamic Programming I

A refresher on Markov chains – I

Definition (Markov Chain)

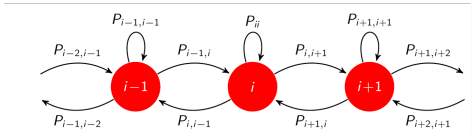
A (time-homogeneous) Markov chain is a stochastic process $\{X_0, X_1, \ldots\}$, taking values on a countable number of states, satisfying the so-called Markov property, i.e.,

$$\mathbb{P}[X_{t+1} = j | X_t = i, X_{t-1}, \ldots, X_0] = \mathbb{P}[X_{t+1} = j | X_t = i] = P_{ij}.$$

Markov Process

Markov process is a triple $\langle \mathcal{S}, \mathsf{P}, \mu \rangle$, where

- $\mathcal{S}$ is the set of all possible states;
- the matrix P with entries $[P]_{ss'} = \mathsf{P}(s'|s)$ is the transition matrix over $\mathcal{S}$;
- μ is the initial state distribution: $s_0 \sim \mu \in \Delta(\mathcal{S})$.



A refresher on Markov chains – II

Definition (Stationary distribution)

If a Markov chain is irreducible and aperiodic with finite states (i.e., ergodic), then there exists a unique stationary distribution $d^\star$ and $\{X_t\}$ converges to it, i.e., $\lim_{t\to\infty}[P^t]_{ij} = d^\star_j, \forall i, j$. We can represent this via $d^\star = d^\star P$ where $[P]_{ij} = P_{ij}$ and $d^\star$ is a row vector. Hence, $d^\star$ is the left principal eigenvector of P.

Remarks:

- Irreducibility:
  - A Markov chain is *irreducible* if it is possible to reach any state from any state.
  - Ensures the chain forms a single communicating graphical model.
- Aperiodicity:
  - A Markov chain is *aperiodic* if every state has a period of 1.
  - Prevents the chain from getting stuck in cycles, allowing thorough mixing.
- Practical Implications:
  - Convergence: Irreducible and aperiodic chains converge to a unique stationary distribution.
  - Ergodicity: Enables estimation of long-term averages by simulation.
  - Mixing Time: Affects the efficiency of simulations and probabilistic modeling.

Markov Decision Processes (MDPs)

Markov Decision Process

An MDP is a tuple $(\mathcal{S}, \mathcal{A}, \mathsf{P}, r, \mu, \gamma)$, where

- $\mathcal{S}$ is the set of all possible states.
- $\mathcal{A}$ is the set of all possible actions.
- For each action a, the matrix P^a with entries $[P^a]_{ss'} = \mathsf{P}(s'|s,a)$ is the transition matrix $(\mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S}))$.
- $r \colon \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function. We assume $r \in [0,1]$.
- μ is the initial state distribution: $s_0 \sim \mu \in \Delta(\mathcal{S})$.
- γ is the discount factor: $\gamma \in (0,1)$.

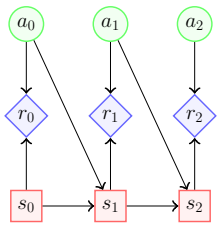


Figure: An MDP graphical model

MDPs: Discount factors and Ergodic chains

Discounting

Let $\mu \in \Delta(\mathcal{S})$ be the initial state distribution and $\gamma \in (0, 1)$ the discount factor. By discounting, we implicitly mean that we move forward in visiting the states in time, where we sample the next state in the chain using

$$s' \sim \begin{cases} \mathsf{P}(\cdot|s, a) & \text{with probability } \gamma; \\ \mu & \text{with probability } 1 - \gamma. \end{cases}$$

Remarks:

- In practice, this represents that the agent re-starts from scratch with probability $1 - \gamma$.
- The MDP resulting from this process is ergodic if $\mu_i > 0, \ \forall i = 1, 2, \ldots, |\mathcal{S}|$.

Example:

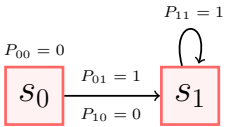


Figure: Non-ergodic chain

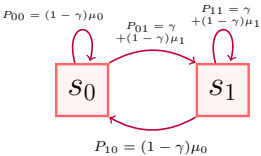


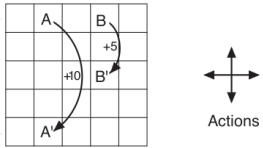
Figure: Ergodic chain

Exercise:

- Prove it!

Example: Gridworld

- State $\mathcal{S}$: the agent's position
- Action $\mathcal{A}$: moving north/south/east/west
- Reward r:
  - -1 if moving outside the world
  - $+10$ if moving to A
  - $+5$ if moving to B
  - 0 otherwise
- Transition model P:
  - move to the adjacent grid according to the direction
  - stay unchanged if moving toward the wall
  - transit to A' if moving into A, transit to B' if moving into B



Example: LLM reasoning

- State $\mathcal{S}$: output tokens in response to a question
- Action $\mathcal{A}$: generating the next token
- Reward r:
  - 0 if the response is not finished[a]
  - $+1$ if the response is finished and correct
  - -1 if the response is finished and wrong
- Transition model P:
  - generate the next token if the response is not finished
  - await evaluation of reward if the response is finished

[a] signified by the generation of an end-of-sentence token in implementation

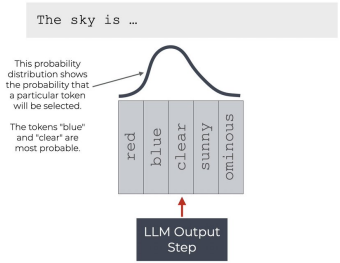


Figure: Next-token generation paradigm.
https://x.com/Josh_Ebner/status/1765108810539024707

MDPs: policies

What is our goal?

Find a behaviour or rule to make decisions that maximize the expected return.

- In general, a policy selects an action based on the history $h_t := (s_{0:t}, a_{0:t-1}) := (s_0, a_0, \ldots, s_{t-1}, a_{t-1}, s_t)$
  - A stationary Markov policy is a mapping $\pi : \mathcal{S} \to \mathcal{A}$ or $\pi : \mathcal{S} \to \Delta(\mathcal{A})$,
  - Δ is the appropriate probability simplex.

Deterministic Policy

- Stationary policy $\pi : \mathcal{S} \to \mathcal{A}$, $a_t = \pi(s_t)$
- Markov policy $\pi_t : \mathcal{S} \to \mathcal{A}$, $a_t = \pi_t(s_t)$
- History-dependent policy $\pi_t : \mathcal{H}_t \to \mathcal{A}$
  - $\mathcal{H}_t$ is the set of histories up to time t.
  - $a_t = \pi_t(h_t)$

Randomized Policy:

- Stationary policy $\pi : \mathcal{S} \to \Delta(\mathcal{A})$, $a_t \sim \pi(\cdot|s_t)$
- Markov policy $\pi_t : \mathcal{S} \to \Delta(\mathcal{A})$, $a_t \sim \pi_t(\cdot|s_t)$
- History-dependent policy $\pi_t : \mathcal{H}_t \to \Delta(\mathcal{A})$
  - $\mathcal{H}_t$ is the set of histories up to time t.
  - $a_t \sim \pi_t(\cdot|h_t)$

Remarks:

- The infinite horizon objective can be maximized by a *stationary deterministic policy*.
- The finite horizon objective needs instead a (nonstationary) *deterministic Markov policy*.

From MDPs to performance criteria

Reminder:

- We have described the role of MDPs while establishing a performance criterion.
  - Finite Horizon: Cumulative reward and average reward.
  - Infinite Horizon: Discounted reward and average reward.
- In this course, we mainly focus on discounted infinite-horizon MDPs:

$$J(\pi) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \,\middle|\, s_0 \sim \mu,\ \pi\right].$$

- We use $\gamma \in (0, 1)$ to trade off past and present rewards.

Observations:

- If $\gamma = 1$, the total reward may be infinite, e.g., when the Markov process is cyclic.
- With $\gamma \in (0, 1)$, assuming bounded rewards, i.e., $r < \infty$, the total return will always be finite.

Value functions

Definition (State-Value Function)

$$V^{\pi}(s) := \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \mid s_0 = s,\ \pi\right]$$

Value functions

Definition (State-Value Function)

$$V^{\pi}(s) := \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \mid s_0 = s,\ \pi\right]$$

Definition (Quality Function / State-Action Value Function)

$$Q^{\pi}(s, a) := \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \mid s_0 = s,\ a_0 = a,\ \pi\right]$$

Observations:
- $V^{\pi}(s)$ represents the total expected return starting at state s under policy π.
- $Q^{\pi}(s, a)$ represents the total expected return when choosing action a in state s under policy π.
- For convenience, we may drop the π in RHS when it is clear from the context.

Remark:
- In the literature, state-value function and value function are used interchangeably.

Value functions (cont'd)

Pop quiz: ○ What is the relation between V^π and Q^π?

Value functions (cont'd)

Pop quiz: ○ What is the relation between V^π and Q^π?

Answer: ○ For any policy $\pi : \mathcal{S} \to \Delta(\mathcal{A})$, it holds that

$$Q^\pi(s,a) = r(s,a) + \gamma \sum_{s' \in \mathcal{S}} \mathsf{P}(s'|s,a) V^\pi(s') \tag{1}$$

$$V^\pi(s) = \sum_{a \in \mathcal{A}} \pi(a \mid s)\, Q^\pi(s,a) \tag{2}$$

Proof of equation (1)

Derivation:

$$
\begin{aligned}
Q^{\pi}(s,a) &= \mathbb{E}\Big[\sum\nolimits_{t=0}^{\infty} \gamma^t r(s_t,a_t) \mid s_0 = s,\, a_0 = a,\, \pi\Big] \\
&= r(s,a) + \mathbb{E}\Big[\sum\nolimits_{t=1}^{\infty} \gamma^t r(s_t,a_t) \mid s_0 = s,\, a_0 = a,\, \pi\Big]
\end{aligned}
$$

Proof of equation (1)

Derivation:

$$
\begin{aligned}
Q^{\pi}(s,a) &= \mathbb{E}\Big[\sum_{t=0}^{\infty} \gamma^t r(s_t,a_t) \mid s_0 = s,\, a_0 = a,\, \pi\Big] \\
&= r(s,a) + \mathbb{E}\Big[\sum_{t=1}^{\infty} \gamma^t r(s_t,a_t) \mid s_0 = s,\, a_0 = a,\, \pi\Big] \\
&= r(s,a) + \gamma \sum_{s' \in \mathcal{S}} \mathsf{P}(s'|s,a)\, \mathbb{E}\Big[\sum_{t=1}^{\infty} \gamma^{t-1} r(s_t,a_t) \mid s_0 = s,\, s_1 = s',\, a_0 = a,\, \pi\Big]
\end{aligned}
$$

Proof of equation (1)

Derivation:

$$
\begin{aligned}
Q^{\pi}(s,a) &= \mathbb{E}\Big[\sum_{t=0}^{\infty} \gamma^t r(s_t,a_t) \mid s_0 = s,\, a_0 = a,\, \pi\Big] \\
&= r(s,a) + \mathbb{E}\Big[\sum_{t=1}^{\infty} \gamma^t r(s_t,a_t) \mid s_0 = s,\, a_0 = a,\, \pi\Big] \\
&= r(s,a) + \gamma \sum_{s' \in \mathcal{S}} \mathsf{P}(s'|s,a)\, \mathbb{E}\Big[\sum_{t=1}^{\infty} \gamma^{t-1} r(s_t,a_t) \mid s_0 = s,\, s_1 = s',\, a_0 = a,\, \pi\Big] \\
&= r(s,a) + \gamma \sum_{s' \in \mathcal{S}} \mathsf{P}(s'|s,a)\, \mathbb{E}\Big[\sum_{t=1}^{\infty} \gamma^{t-1} r(s_t,a_t) \mid s_1 = s',\, \pi\Big] \qquad \text{(Markov assumption)}
\end{aligned}
$$

Proof of equation (1)

Derivation:

$$
\begin{aligned}
Q^\pi(s,a) &= \mathbb{E}\Big[\sum_{t=0}^{\infty} \gamma^t r(s_t,a_t) \mid s_0 = s,\, a_0 = a,\, \pi\Big] \\
&= r(s,a) + \mathbb{E}\Big[\sum_{t=1}^{\infty} \gamma^t r(s_t,a_t) \mid s_0 = s,\, a_0 = a,\, \pi\Big] \\
&= r(s,a) + \gamma \sum_{s' \in \mathcal{S}} \mathsf{P}(s'|s,a)\, \mathbb{E}\Big[\sum_{t=1}^{\infty} \gamma^{t-1} r(s_t,a_t) \mid s_0 = s,\, s_1 = s',\, a_0 = a,\, \pi\Big] \\
&= r(s,a) + \gamma \sum_{s' \in \mathcal{S}} \mathsf{P}(s'|s,a)\, \mathbb{E}\Big[\sum_{t=1}^{\infty} \gamma^{t-1} r(s_t,a_t) \mid s_1 = s',\, \pi\Big] \qquad \text{(Markov assumption)} \\
&= r(s,a) + \gamma \sum_{s' \in \mathcal{S}} \mathsf{P}(s'|s,a)\, \mathbb{E}\Big[\sum_{t=0}^{\infty} \gamma^{t} r(s_t,a_t) \mid s_0 = s',\, \pi\Big] \qquad \big(\text{i.e., } V^\pi(s')\big)
\end{aligned}
$$

Proof of equation (1)

Derivation:

$$
\begin{aligned}
Q^\pi(s,a) &= \mathbb{E}\Big[\sum_{t=0}^{\infty} \gamma^t r(s_t,a_t) \mid s_0 = s,\, a_0 = a,\, \pi\Big] \\
&= r(s,a) + \mathbb{E}\Big[\sum_{t=1}^{\infty} \gamma^t r(s_t,a_t) \mid s_0 = s,\, a_0 = a,\, \pi\Big] \\
&= r(s,a) + \gamma \sum_{s' \in \mathcal{S}} \mathsf{P}(s'|s,a)\; \mathbb{E}\Big[\sum_{t=1}^{\infty} \gamma^{t-1} r(s_t,a_t) \mid s_0 = s,\, s_1 = s',\, a_0 = a,\, \pi\Big] \\
&= r(s,a) + \gamma \sum_{s' \in \mathcal{S}} \mathsf{P}(s'|s,a)\; \mathbb{E}\Big[\sum_{t=1}^{\infty} \gamma^{t-1} r(s_t,a_t) \mid s_1 = s',\, \pi\Big] \qquad \text{(Markov assumption)} \\
&= r(s,a) + \gamma \sum_{s' \in \mathcal{S}} \mathsf{P}(s'|s,a)\; \mathbb{E}\Big[\sum_{t=0}^{\infty} \gamma^{t} r(s_t,a_t) \mid s_0 = s',\, \pi\Big] \qquad \big(\text{i.e., } V^\pi(s')\big) \\
&= r(s,a) + \gamma \sum_{s' \in \mathcal{S}} \mathsf{P}(s'|s,a)\; V^\pi(s') \square
\end{aligned}
$$

Optimal value functions

- Let Π be the set of all (possibly non-stationary and randomized) policies.

Definition (Optimal Value Function)

$$V^\star(s) := \max_{\pi \in \Pi} V^\pi(s)$$

Definition (Optimal Action-Value Function)

$$Q^\star(s,a) := \max_{\pi \in \Pi} Q^\pi(s,a)$$

Pop quiz: - What is the relation between $V^\star$ and $Q^\star$?

Optimal value functions

- Let Π be the set of all (possibly non-stationary and randomized) policies.

Definition (Optimal Value Function)

$$V^\star(s) := \max_{\pi \in \Pi} V^\pi(s)$$

Definition (Optimal Action-Value Function)

$$Q^\star(s,a) := \max_{\pi \in \Pi} Q^\pi(s,a)$$

Pop quiz: - What is the relation between $V^\star$ and $Q^\star$?

Answer:

$$Q^\star(s,a) = r(s,a) + \gamma \sum_{s' \in \mathcal{S}} \mathsf{P}(s'|s,a) V^\star(s') \tag{3}$$

$$V^\star(s) = \max_{a \in \mathcal{A}} Q^\star(s,a) \tag{4}$$

- *Self-exercise*: prove equation (4).

Solving MDPs: find the optimal policy

Goal

Roughly speaking, the ultimate goal in RL can be summed up to finding an optimal policy $\pi^\star \in \Pi$ such that

$$V^{\pi^\star}(s) = V^\star(s) := \max_{\pi \in \Pi} V^\pi(s), \forall s \in \mathcal{S}.$$

Remark: ○ The optimal policy may not be unique, while $V^\star$ is unique.

Key Questions

- **Q1**: Does the optimal policy $\pi^\star$ exist?
- **Q2**: How to evaluate my current policy π, i.e., how to compute $V^\pi(s)$? *–policy evaluation*
- **Q3**: If $\pi^\star$ exists, how to improve my current policy π, i.e., how to find $\pi^\star$? *–policy improvement*

Bellman optimality conditions

- The optimal value function $V^\star$ is the unique fixed point of the following equation:

$$V^\star(s) = \max_{a \in \mathcal{A}} \left[r(s,a) + \gamma \sum_{s' \in \mathcal{S}} \mathsf{P}(s'|s,a) V^\star(s') \right].$$

Remarks:

- This requirement is also known as the Bellman optimality equation.
- We will show that there exists a deterministic optimal policy.
- Fixed-point perspective motivates value iteration (VI) and policy iteration (PI) methodologies.

Existence of an optimal policy

Theorem (Existence of an optimal policy [1] [2])

For an infinite horizon MDP $M = (\mathcal{S}, \mathcal{A}, P, t, \mu, \gamma)$, there exists a stationary and deterministic policy π such that for any $s \in \mathcal{S}$ and $a \in \mathcal{A}$, we have

$$V^{\pi}(s) = V^{\star}(s), \quad Q^{\pi}(s,a) = Q^{\star}(s,a).$$

Remarks:

- Finding $\pi^{\star}$ can be done by first computing $V^{\star}$ or $Q^{\star}$.
- Note that we can directly get a (deterministic and stationary) optimal policy from $Q^{\star}$:

$$\pi^{\star}(s) = \arg\max_{a \in \mathcal{A}} Q^{\star}(s,a).$$

- Note: Proof of the theorem can be found the supplementary slides #2.

Bellman consistency equation

Richard Ernest Bellman
(August 26, 1920 – March 19, 1984)

Theorem (Bellman Consistency Equation)

$$V^{\pi}(s) = \mathbb{E}_{a\sim\pi(\cdot|s)}\left[r(s,a) + \gamma \sum_{s'\in\mathcal{S}} P(s'|s,a)V^{\pi}(s')\right] \qquad \text{(BCE)}$$

Bellman consistency equation

Richard Ernest Bellman
(August 26, 1920 – March 19, 1984)

Theorem (Bellman Consistency Equation)

$$V^{\pi}(s) = \mathbb{E}_{a\sim\pi(\cdot|s)}\left[r(s,a) + \gamma \sum_{s'\in\mathcal{S}} P(s'|s,a)V^{\pi}(s')\right] \qquad \text{(BCE)}$$

Remarks:

- BCE is also known as Bellman expectation equation.
- BCE states the value of a state under a given policy π, which is
  - the expected return starting from that state, taking an action according to the policy,
  - ... and thereafter following the policy....

Bellman consistency equation

Richard Ernest Bellman
(August 26, 1920 – March 19, 1984)

Theorem (Bellman Consistency Equation)

$$V^{\pi}(s) = \mathbb{E}_{a\sim\pi(\cdot|s)}\left[r(s,a) + \gamma \sum_{s'\in\mathcal{S}} P(s'|s,a)V^{\pi}(s')\right] \qquad \text{(BCE)}$$

Matrix Form

We can concisely represent the Bellman consistency equation in the following matrix form: $V^{\pi} = R^{\pi} + \gamma P^{\pi} V^{\pi}$.

- We can derive from equations (1) and (2):

$$Q^{\pi}(s,a) = r(s,a) + \gamma \sum_{s'\in\mathcal{S}} \mathsf{P}(s'|s,a)V^{\pi}(s') \quad (1)$$

$$V^{\pi}(s) = \sum_{a\in\mathcal{A}} \pi(a \mid s)\, Q^{\pi}(s,a) \quad (2)$$

- We can write, with $|\mathcal{S}|$ being the cardinality of $\mathcal{S}$:

$$V^{\pi} \in \mathbb{R}^{|\mathcal{S}|} : V^{\pi}_s = V^{\pi}(s);$$

$$R^{\pi} \in \mathbb{R}^{|\mathcal{S}|}, R^{\pi}_s := \sum_{a\in\mathcal{A}} \pi(a|s) r(s,a);$$

$$P^{\pi} \in \mathbb{R}^{|\mathcal{S}|\times|\mathcal{S}|} : P^{\pi}_{s,s'} := \sum_{a\in\mathcal{A}} \pi(a|s)\mathsf{P}(s' \mid s,a).$$

Closed-form solution for policy evaluation

Closed-Form Solution of V^π

Given the matrix form of BCE, we have the following closed-form solution: $V^\pi = (I - \gamma P^\pi)^{-1} R^\pi$.

Remarks:

- This is one of exact solution methods for policy evaluation.
- Note that the matrix $I - \gamma P^\pi$ is always invertible for $\gamma \in (0, 1)$.
- The solution of Bellman equation is always unique.
- Computation cost: $\mathcal{O}(|\mathcal{S}|^3 + |\mathcal{S}|^2|\mathcal{A}|)$, which can be expensive for large state spaces.

Bellman expectation operator and fixed-point perspective

Definition (Bellman expectation operator)

The Bellman expectation operator $\mathcal{T}^\pi : \mathbb{R}^{|\mathcal{S}|} \to \mathbb{R}^{|\mathcal{S}|}$ is defined by the following expression

$$\mathcal{T}^\pi V := R^\pi + \gamma P^\pi V. \tag{5}$$

Remarks:

- BCE implies that V^π is the fixed point of $\mathcal{T}^\pi$: $\mathcal{T}^\pi V^\pi = V^\pi$.
- $\mathcal{T}^\pi$ is a linear operator and is a γ-contraction mapping.
- The solution of BCE is always unique.
- For the following iteration invariant $V_{t+1} = \mathcal{T}^\pi V_t$, $t = 0, 1, \ldots$, it holds that

$$\lim_{t \to \infty} (\mathcal{T}^\pi)^t V_0 = V^\pi.$$

Bellman optimality conditions

Theorem (Bellman optimality equation)

The optimal value and action-value functions satisfy the following equations:

$$V^\star(s) = \max_{a \in \mathcal{A}} \left[r(s,a) + \gamma \sum_{s' \in \mathcal{S}} P(s'|s,a) V^\star(s') \right],$$

$$Q^\star(s,a) = r(s,a) + \gamma \left[\sum_{s' \in \mathcal{S}} P(s'|s,a) \left(\max_{a' \in \mathcal{A}} Q^\star(s',a') \right) \right].$$

Remarks:

- These requirements are also known as Bellman optimality conditions.
- Obtained by combining equations (3) and (4).
- Fixed-point perspective motivates value iteration (VI) and policy iteration (PI) methodologies.

Bellman optimality operator

Definition (Bellman optimality operator)

We define the following operator $\mathcal{T}$, which will be useful when discussing value-iteration in the sequel:

$$(\mathcal{T}V)(s) := \max_{a \in \mathcal{A}} \left[r(s,a) + \gamma \sum_{s' \in \mathcal{S}} P(s'|s,a) V(s') \right]. \qquad \text{(Bellman operator)}$$

Remarks:

- The optimal value function $V^\star$ is the **fixed point** of $\mathcal{T}$, i.e.,

$$\mathcal{T}V^\star = V^\star.$$

- The Bellman optimality operator is a γ-contraction mapping w.r.t. ℓ_∞-norm (proof in slide #4):

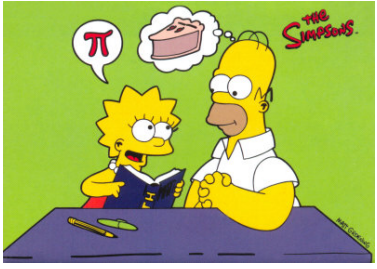
$$\left\| \mathcal{T}V' - \mathcal{T}V \right\|_\infty \leq \gamma \left\| V' - V \right\|_\infty.$$

- The Bellman operator is also monotonic (component-wise): $V' \leq V \;\Rightarrow\; \mathcal{T}V' \leq \mathcal{T}V$.

- We can define a similar Bellman operator on the Q-function and show similar properties.

Pause and reflect

- Before we move on, take a minute to reflect on these important notations:
  - π, $\pi^\star$, $V^\pi(s)$, $V^\star(s)$, $Q^\pi(s,a)$, $Q^\star(s,a)$, $\mathcal{T}^\pi$, $\mathcal{T}$



Solving MDPs

- What we talked about:
  - Optimal state-value function ($V^\star(s)$) and optimal action-value Function ($Q^\star(s,a)$).
  - Bellman consistency equation ($V^\pi = R^\pi + \gamma P^\pi V^\pi$).
  - Bellman expectation operator and fixed-point perspective ($\mathcal{T}^\pi V := R^\pi + \gamma P^\pi V$).
  - Bellman optimality equations and Bellman optimality operator.

- How do we use this to do "planning," i.e., finding an optimal policy via MDPs (our goal)?

Algorithm	**Component**	**Output**
Value Iteration (VI)	Bellman Optimality Operator $\mathcal{T}$	V_T such that $\|V_T - V^\star\| \leq \epsilon$
Policy Iteration (PI).	Bellman Operator $\mathcal{T}^\pi$ + Greedy Policy	$V^\star$ and $\pi^\star$

Observation: - These solutions require, and we assume throughout, that the transitions dynamics are known.

Value iteration (VI)

Algorithm: Value Iteration (VI) for solving MDPs

Start with an arbitrary guess V_0 (e.g., $V_0(s) = 0$ for any s)

for each iteration t **do**

Apply the BELLMAN OPERATOR $\mathcal{T}$ to the current value estimate V_t:

$$V_{t+1} = \mathcal{T} V_t.$$

end for

Remarks:

- Finding $V^\star$ or $\pi^\star$ is equivalent to finding a fixed point of $\mathcal{T}$.
- Value iteration can be therefore viewed as a fixed-point iteration.

Discussion on value iteration

- After obtaining $V^\star$ via VI, we can obtain an optimal policy from the greedy policy:

$$\pi^\star(s) = \arg\max_{a \in \mathcal{A}} \left[r(s,a) + \gamma \sum_{s' \in \mathcal{S}} \mathsf{P}(s'|s,a) V^\star(s') \right].$$

Discussion on value iteration

- After obtaining $V^\star$ via VI, we can obtain an optimal policy from the greedy policy:

$$\pi^\star(s) = \arg\max_{a \in \mathcal{A}} \left[r(s,a) + \gamma \sum_{s' \in \mathcal{S}} \mathsf{P}(s'|s,a) V^\star(s') \right].$$

- Alternatively, we can run Q-value iteration and compute $\pi^\star$ via

$$\pi^\star(s) = \arg\max_{a \in \mathcal{A}} \; Q^\star(s,a).$$

Remarks:

- Q-value iteration uses the following update derived from equations (1) and (2):

$$Q_{t+1}(s,a) = r(s,a) + \gamma \sum_{s' \in \mathcal{S}} \mathsf{P}(s'|s,a) \max_{a' \in \mathcal{A}} Q_t(s',a'), \tag{1}$$

- The Q-value iteration does not require knowledge of P to extract the policy $\pi^\star$.
- This observation is the starting point to develop "model-free" algorithms in the sequel.

Convergence of value iteration

Theorem (Linear Convergence of Value Iteration)

The value iteration algorithm attains a linear convergence rate, i.e.,

$$\|V_t - V^\star\|_\infty \leq \gamma^t \|V_0 - V^\star\|_\infty.$$

Convergence of value iteration

Theorem (Linear Convergence of Value Iteration)

The value iteration algorithm attains a linear convergence rate, i.e.,

$$\|V_t - V^\star\|_\infty \leq \gamma^t \|V_0 - V^\star\|_\infty.$$

Proof.

$$\|V_t - V^\star\|_\infty = \|\mathcal{T}V_{t-1} - \mathcal{T}V^\star\|_\infty \leq \gamma \|V_{t-1} - V^\star\|_\infty \leq \cdots \leq \gamma^t \|V_0 - V^\star\|_\infty.$$

□

Remarks:

- The complexity of applying $\mathcal{T}$ is $\mathcal{O}\left(|\mathcal{S}|^2|\mathcal{A}|\right)$.
- The number of iterations to reach ϵ accuracy is $\mathcal{O}\left(\log \epsilon^{-1}\right)$ due to linear convergence.

Directly update the policy

- Value iteration first finds $V^\star$, then computes the optimal policy $\pi^\star$ by the greedy policy.

Directly update the policy

- Value iteration first finds $V^\star$, then computes the optimal policy $\pi^\star$ by the greedy policy.
- We can also directly search for the optimal policy $\pi^\star$.

Some intuition: - Starting with an initial guess π, we can iteratively perform the following motions:

1. Evaluate policy: compute the value function V^π of the current policy
   $\Rightarrow$ Policy evaluation
2. Improve policy: update the guess by the greedy policy w.r.t. V^π
   $\Rightarrow$ Policy improvement

Policy improvement theorem

Theorem (Policy Improvement)

If a (deterministic) policy π' satisfies the following

$$Q^{\pi}(s, \pi'(s)) \geq V^{\pi}(s) \quad \forall\ s \in \mathcal{S}, \tag{6}$$

then we have $V^{\pi'}(s) \geq V^{\pi}(s)$ for any $s \in \mathcal{S}$.

Remarks:

- The same result holds for a stochastic policy π' if $\mathbb{E}_{a \sim \pi'(\cdot|s)} Q^{\pi}(s, a) \geq V^{\pi}(s) \quad \forall\ s \in \mathcal{S}$.
- Improving the current policy by one step everywhere, we can improve the whole policy.
- It suggests a natural way of improving the current policy via

$$\pi_{t+1}(s) \leftarrow \arg\max_{a \in \mathcal{A}} Q^{\pi_t}(s, a).$$

- Indeed, $V^{\pi_{t+1}}(s) \geq V^{\pi_t}(s), \forall\ s \in \mathcal{S}$, and the inequality is strict if π_t is suboptimal.

Policy iteration

Algorithm: Policy Iteration (PI) for solving MDPs

Start with an arbitrary policy guess π_0

for each iteration t **do**

(Step 1: Policy evaluation) Compute V^{π_t}:

(Option 1) Iteratively apply policy value iteration, $V_t \leftarrow \mathcal{T}^{\pi_t} V_t$, until convergence

(Option 2) Use the closed-form solution: $V^{\pi_t} = (I - \gamma P^{\pi_t})^{-1} R^{\pi_t}$

(Step 2: Policy improvement) Update the current policy π_t by the greedy policy

$$\pi_{t+1}(s) = \arg\max_{a \in \mathcal{A}} \left[r(s,a) + \gamma \sum_{s' \in \mathcal{S}} \mathsf{P}(s'|s,a) V^{\pi_t}(s') \right]. \tag{7}$$

end for

Remarks:

- Recall that we assume that there exists a deterministic optimal policy.
- Greedy policy achieves the optimal deterministic policy.

Comparison

Algorithm	Value Update	Policy Update
Value Iteration (VI)	$V_{t+1} = \mathcal{T} V_t.$	None
Policy Iteration (PI)	$V_{t+1} = \mathbb{E}\Big[r(s,a) + \gamma \sum_{s' \in \mathcal{S}} \mathrm{P}(s'|s,a) V(s') | \pi_t\Big]$	Greedy Policy

Algorithm	Per iteration cost	Number of iterations	Output
Value Iteration (VI)	$\mathcal{O}\big(\lvert\mathcal{S}\rvert^2 \lvert\mathcal{A}\rvert\big)$	$T = \mathcal{O}\big(\frac{\log(\epsilon(1-\gamma))}{\log \gamma}\big)$	V_T such that $\lVert V_T - V^\star \rVert \leq \epsilon$
Policy Iteration (PI)	$\mathcal{O}\big(\lvert\mathcal{S}\rvert^3 + \lvert\mathcal{S}\rvert^2 \lvert\mathcal{A}\rvert\big)$	$T = \mathcal{O}\big(\frac{\lvert\mathcal{S}\rvert(\lvert\mathcal{A}\rvert-1)}{1-\gamma}\big)$	$V^\star$ and $\pi^\star$

Observations:

- VI and PI are broadly dynamic programming approaches.
- PI converges in finite number of iterations [4] whereas VI does not [3].
- These solution mythologies are broadly known as model-based RL.
- Modified policy iteration [5] performs limited value-function updates for speed-ups.

Convergence of policy iteration (PI)

Theorem (Linear Convergence of Policy Iteration)

Policy iteration outputs the optimal policy after $\mathcal{O}\left(\frac{|\mathcal{S}||\mathcal{A}|}{1-\gamma}\right)$ *iterations.*

Proof.

- For simplicity, we provide just the proof sketch.
- The first step is to prove that PI identifies a suboptimal action at a certain state every $\mathcal{O}\left(\frac{1}{1-\gamma}\right)$.
- The proof is concluded noticing that there exists at most $|\mathcal{S}|\left(|\mathcal{A}|-1\right)$ suboptimal actions. □

Summary I

- Basic concepts of Markov decision process (MDP)
  - Policy, value functions, optimal value functions
  - Bellman equations and Bellman operators
  - Fixed point viewpoints
  - Existence and construction of optimal policy

Summary I

- Basic concepts of Markov decision process (MDP)
  - Policy, value functions, optimal value functions
  - Bellman equations and Bellman operators
  - Fixed point viewpoints
  - Existence and construction of optimal policy
- Exact solution methods for policy evaluation

Summary I

- Basic concepts of Markov decision process (MDP)
  - Policy, value functions, optimal value functions
  - Bellman equations and Bellman operators
  - Fixed point viewpoints
  - Existence and construction of optimal policy
- Exact solution methods for policy evaluation
- Exact solution methods for solving MDPs
  - Value iteration: iteratively apply Bellman operator
  - Policy iteration: alternatively execute policy evaluation and policy improvement

Wrap Up

- PI and VI are dynamic programming methods applicable when the transition matrix is known.
- The following week: What to do when the transition matrix is known!

References I

[1] Alekh Agarwal, Nan Jiang, Sham M Kakade, and Wen Sun.
Reinforcement learning: Theory and algorithms.
CS Dept., UW Seattle, Seattle, WA, USA, Tech. Rep, 2019.
50

[2] Martin L Puterman.
Markov decision processes: discrete stochastic dynamic programming.
John Wiley & Sons, 2014.
50

[3] Satinder Singh Richard and Richard C. Yee.
An upper bound on the loss from approximate optimal-value functions.
In *Machine Learning*, pages 227–233, 1994.
69

[4] Bruno Scherrer.
Improved and generalized upper bounds on the complexity of policy iteration.
Mathematics of Operations Research, 41(3):758–774, 2016.
69

[5] Bruno Scherrer, Victor Gabillon, Mohammad Ghavamzadeh, and Matthieu Geist.
Approximate modified policy iteration.
arXiv preprint arXiv:1205.3054, 2012.
69

Supplementary material

Existence of an optimal policy (proof)

Proof Sketch

Assume a start from $(s_0, a_0, r_0, s_1) = (s, a, r, s')$*, then*

1. *Define "offset" policy* $\tilde{\pi}(a_t = a \,|\, h_t) := \pi(a_{t+1} = a \,|\, (s_0, a_0) = (s, a), h_t)$*, Markov property implies*

$$\mathbb{E}\left[\sum_{t=1}^{\infty} \gamma^t r(s_t, a_t) \,|\, (s_0, a_0, r_0, s_1) = (s, a, r, s'),\, \pi\right] = \gamma V^{\tilde{\pi}}(s').$$

2. *With all* $(s_0, a_0, r_0) = (s, a, r)$*, the set* $\{\tilde{\pi} \,|\, \Pi\}$ *will just be* Π *itself.*
3. *Show that the optimal value from* s_1 *onward is independent of* $(s_0, a_0, r_0) = (s, a, r)$*,*

$$\max_{\pi \in \Pi} \mathbb{E}\left[\sum_{t=1}^{\infty} \gamma^t r(s_t, a_t) \,|\, (s_0, a_0, r_0, s_1) = (s, a, r, s'),\, \pi\right] = \gamma \max_{\pi \in \Pi} V^{\tilde{\pi}}(s') = \gamma \max_{\pi \in \Pi} V^{\pi}(s') = \gamma V^{\star}(s').$$

Existence of an optimal policy (proof)

Proof Sketch (cont.)

4. *Let* $\pi(s) = \arg\max_{a \in \mathcal{A}} \max_{\pi' \in \Pi} Q^{\pi'}(s,a)$, *show this deterministic and randomized policy is optimal*

$$
\begin{aligned}
V^\star(s_0) &= \max_{\pi' \in \Pi} \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \mid s_0 = s,\ \pi\right] = \max_{\pi' \in \Pi} \mathbb{E}\left[r(s_0,a_0) + \sum_{t=1}^{\infty} \gamma^t r(s_t,a_t) \mid \pi\right] \\
&= \max_{\pi' \in \Pi} \mathbb{E}\left[r(s_0,a_0) + \mathbb{E}\left[\sum_{t=1}^{\infty} \gamma^t r(s_t,a_t) \mid (s_0,a_0,r_0,s_1), \pi\right]\right] \qquad (8) \\
&\leq \max_{\pi' \in \Pi} \mathbb{E}\left[r(s_0,a_0) + V^\star(s_1)\right] \quad \Longleftarrow \quad \textit{Step 3 above} \\
&= \mathbb{E}\left[r(s_0,a_0) + V^\star(s_1) \mid \pi\right] \quad \Longleftarrow \quad \textit{Definition of } \pi \textit{ above}
\end{aligned}
$$

5. $V^\star(s_0) \leq \mathbb{E}\left[r(s_0,a_0) + V^\star(s_1) \mid \pi\right] \leq \mathbb{E}\left[r(s_0,a_0) + \gamma r(s_1,a_1) + \gamma^2 V^\star(s_1) \mid \pi\right] \leq \cdots \leq V^\pi(s_0)$, *so* $\mathbf{V}^\pi = \mathbf{V}^\star$, *i.e., the proposed* π *is optimal.*

□

Link back to the referring slide #38.

Contraction of bellman optimality operator (proof)

Proof.

For any $\mathbf{V}', \mathbf{V} \in \mathbb{R}^{|\mathcal{S}|}$ and $s \in \mathcal{S}$, we have

$$
\begin{aligned}
& \left|\big(\mathcal{T}\mathbf{V}'\big)(s) - (\mathcal{T}\mathbf{V})(s)\right| \\
&= \left|\max_{a\in\mathcal{A}}\left[r(s,a) + \gamma \sum\nolimits_{s'\in\mathcal{S}} \mathsf{P}(s'|s,a)\mathbf{V}'(s')\right] - \max_{a'\in\mathcal{A}}\left[r(s,a') + \gamma \sum\nolimits_{s'\in\mathcal{S}} \mathsf{P}(s'|s,a')\mathbf{V}(s')\right]\right| \\
&\leq \max_{a\in\mathcal{A}}\left|\left(r(s,a) + \gamma \sum\nolimits_{s'\in\mathcal{S}} \mathsf{P}(s'|s,a)\mathbf{V}'(s')\right) - \left(r(s,a) + \gamma \sum\nolimits_{s'\in\mathcal{S}} \mathsf{P}(s'|s,a)\mathbf{V}(s')\right)\right| \\
&\leq \max_{a\in\mathcal{A}} \gamma \sum\nolimits_{s'\in\mathcal{S}} \mathsf{P}(s'|s,a)\left|\mathbf{V}'(s') - \mathbf{V}(s')\right| \\
&\leq \left\|\mathbf{V}' - \mathbf{V}\right\|_\infty \max_{a\in\mathcal{A}} \gamma \sum\nolimits_{s'\in\mathcal{S}} \mathsf{P}(s'|s,a) = \gamma\left\|\mathbf{V}' - \mathbf{V}\right\|_\infty,
\end{aligned}
$$

which concludes the proof. Link to slide #43. □

Policy improvement theorem (proof)

Theorem (Policy Improvement)

If a (deterministic) policy π' satisfies that,

$$Q^{\pi}(s, \pi'(s)) \geq V^{\pi}(s) \quad \forall\ s \in \mathcal{S}, \tag{9}$$

then $V^{\pi'}(s) \geq V^{\pi}(s)$ for any $s \in \mathcal{S}$.

Proof.

Follow the property, for any $s \in \mathcal{S}$, (denote $s' \sim \mathsf{P}(\cdot|s, \pi'(s))$ as $s' \sim \pi'$)

$$\begin{aligned} V^{\pi}(s) \leq\ Q^{\pi}(s, \pi'(s)) &= \mathbb{E}_{\pi'}\left[r(s_0, \pi'(s_0)) + \gamma V^{\pi}(s_1) | \, s_0 = s\right] \\ &\leq \mathbb{E}_{\pi'}\left[r_0 + \gamma Q^{\pi}(s_1, \pi'(s_1)) \, | \, s_0 = s\right] \\ &\leq \mathbb{E}_{\pi'}[r_0 + \gamma r_1 + \gamma V^{\pi}(s_1) \, | \, s_0 = s] \\ &\leq \cdots \\ &\leq \mathbb{E}_{\pi'}\left[r_0 + \gamma r_1 + \gamma^2 r_2 + \cdots \, | \, s_0 = s\right] = V^{\pi'}(s). \end{aligned} \tag{10}$$

□

Link to slide #50.