CSE 542: Statistical Reinforcement Learning

Lecture 1: Course Logistics & MDP Fundamentals

Kevin Jamieson

Paul G. Allen School of Computer Science & Engineering
University of Washington

What is this course about?

Reinforcement learning (RL): how should an agent act in an uncertain, sequential environment to maximize cumulative reward?

This course develops the mathematical and algorithmic foundations of RL:

- Provably efficient methods and their theoretical guarantees
- From classical tabular settings to modern function approximation
- Both offline (batch) and online (interactive) learning

By the end, you will be positioned to **read and contribute to RL research**.

Topics

Foundations

- MDP theory: value functions, Bellman equations
- Policy iteration, value iteration
- Planning complexity

Online RL

- Model-based/free exploration, UCB for tabular and linear MDPs with sample complexity guarantees
- Policy gradient

Offline RL

- Fitted value iteration
- Offline policy evaluation & optimization

Function Approximation

- Linear MDPs, LSVI-UCB
- Bellman rank, Eluder dimension
- Provably efficient algorithms

Prerequisites

Required:

- Linear algebra, probability, calculus (see HW0 self-test)
- Concentration inequalities: Hoeffding, Bernstein, Azuma–Hoeffding

Strongly recommended:

- Online learning and multi-armed bandits (CSE 541 or equivalent)
- Consult [Szepesvári–Lattimore] or CSE 541 (Winter 2026) materials

Self-test: Complete HW0 (not graded) in the first week to gauge your readiness.

Course Materials

Primary Textbook:

- *Reinforcement Learning: Theory and Algorithms*, Agarwal, Brantley, Jiang, Kakade, Sun
  `rltheorybook.github.io/rltheorybook_ABJKS.pdf`

Recommended Background:

- *Bandit Algorithms*, Szepesvári & Lattimore
  `https://tor-lattimore.com/downloads/book/book.pdf`

Additional References (Practical RL):

- *Reinforcement Learning: An Introduction*, Sutton & Barto
- *Reinforcement Learning: An Overview*, Murphy (2024)

Logistics at a Glance

Meeting Times

Mon/Wed 10:00–11:20 AM
ECE 045

Office Hours

Instructor: Tue 2:30–3:30, CSE 340
Mars Gao: TBD
Kevin Huang: TBD

Communication

Discussion board: **Ed** (preferred)
Email:
`cse542-staff@cs.washington.edu`

Grading

Homework 1	20%
Homework 2	20%
Homework 3	20%
Final Project	40%

Submission

Single PDF to **Gradescope**
Typeset (LaTeX recommended)
No photos/scans

Final Project

- Choose a topic in reinforcement learning
- Write a **summary and literature review**: a primer for someone about to enter research in that area
- Worth **40%** of your grade
- Details forthcoming

Late Policy (Homeworks)

24-hour grace period, no questions asked.
Multiple days: email instructor *before* the due date.
Late policy does *not* apply to the final project.

Collaboration & LLM Policy

Homeworks are individual

You may collaborate to discuss ideas, but **write your own solutions**.
List all collaborators on your submission.

LLM Use Policy

LLMs (ChatGPT, etc.) are allowed as a *learning aid*.
If you use an LLM, you must attach a link to the full transcript.

If you find yourself copying substantial derivations from the LLM $\Rightarrow$ you've crossed the line.

No transcript attached + *LLM suspected* $\Rightarrow$ report filed.

Academic Integrity

Do not use pre-existing solutions from the web, past courses, or other textbooks.
Violations are reported to Community Standards and Student Conduct.

Today's Plan

Motivation: Sequential Decision Making

Many problems involve **sequential decisions under uncertainty**:

- Robot navigation: reach a goal efficiently
- Conversational agents: complete user tasks
- Games: chess, Go, poker
- Resource allocation, clinical trials, ...

Key challenges:

- Actions affect future states (not i.i.d.!)
- Reward may be delayed
- Need to balance exploration vs. exploitation

The (Infinite-Horizon Discounted) MDP

A **Markov Decision Process** is a tuple $M = (\mathcal{S}, \mathcal{A}, P, r, \gamma, \mu)$:

- $\mathcal{S}$: state space (assume finite)
- $\mathcal{A}$: action space (assume finite)
- $P : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$: transition kernel
  $P(s' \mid s, a)$: prob. of moving to s'
- $r : \mathcal{S} \times \mathcal{A} \to [0, 1]$: reward function
- $\gamma \in [0, 1)$: discount factor
- $\mu \in \Delta(\mathcal{S})$: initial state distribution

Interaction Protocol

$s_0 \sim \mu$. At each step $t = 0, 1, 2, \ldots$:

1. Observe s_t
2. Select $a_t \in \mathcal{A}$
3. Receive $r_t = r(s_t, a_t)$
4. Transition $s_{t+1} \sim P(\cdot \mid s_t, a_t)$

Goal: Maximize $\mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \;\middle|\; s_0 = s\right]$

Examples

Navigation

State = current location. Actions = {N, S, E, W}. Deterministic transitions.
Reward = 1 at goal, else 0. Discount γ incentivizes shortest paths.
Optimal policy: greedy shortest path. Value $\approx \gamma^d$ for path length d.

Inventory Management

State = current stock level. Action = how much inventory to order each day.
Demand is stochastic: reward = sales revenue (only if item is in stock) − ordering cost.
Tension: understocking loses sales; overstocking ties up capital.

Strategic Games

State = board position. Actions = legal moves.
Reward = eventual win/loss.
RL has achieved superhuman performance in Go, Chess, Poker, Backgammon.

Planning vs. Statistical RL

Planning (dynamics known)

The model P, r is given. The challenge is purely *computational*: how do we efficiently find the optimal policy?

Examples:

- Shortest path / navigation
- Board games (Chess, Go): simulator provides exact next state
- Video games: emulator is the model

Statistical RL (dynamics unknown)

We don't know P or r. We must *learn* what actions lead to what outcomes by interacting with the environment.

Examples:

- Personalized medical treatment: how does this patient's condition evolve if we prescribe drug A vs. drug B?
- Inventory management: what is the true demand distribution?

Planning vs. Statistical RL

Planning (dynamics known)

The model P, r is given. The challenge is purely *computational*: how do we efficiently find the optimal policy?

Examples:

- Shortest path / navigation
- Board games (Chess, Go): simulator provides exact next state
- Video games: emulator is the model

Statistical RL (dynamics unknown)

We don't know P or r. We must *learn* what actions lead to what outcomes by interacting with the environment.

Examples:

- Personalized medical treatment: how does this patient's condition evolve if we prescribe drug A vs. drug B?
- Inventory management: what is the true demand distribution?

Focus of this course

We study the **statistical** problem: how many interactions does it take to learn a near-optimal policy? Planning serves as a subroutine, but the core challenge is exploration and generalization.

Policies and Value Functions

History: $\tau_t = (s_0, a_0, r_0, s_1, \ldots, s_t)$.
A **policy** π maps histories to distributions over actions.

Stationary Policy

$\pi : \mathcal{S} \to \Delta(\mathcal{A})$ — actions depend only on current state s_t.
A **deterministic** stationary policy is $\pi : \mathcal{S} \to \mathcal{A}$.

Value function $V^\pi : \mathcal{S} \to \mathbb{R}$:

$$V^\pi(s) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \,\middle|\, \pi,\, s_0 = s\right], \qquad 0 \le V^\pi(s) \le \frac{1}{1-\gamma}.$$

Action-value (Q-value) function $Q^\pi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$:

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

Goal: find π maximizing $V^\pi(s)$ for all s (or for $s \sim \mu$).