Quiz 3 Review

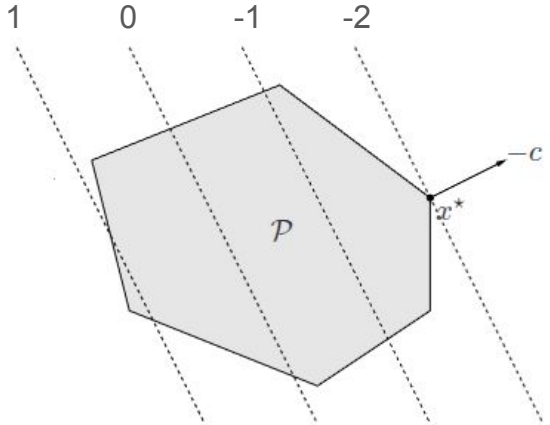
Overview

- **Formulating and Solving Optimization Problems**
  - Linear Programming
- **Reinforcement Learning**
  - Markov Decision Processes
  - The Principle of Optimality
  - Value Iteration

Linear Programming

- **Linear objective:** objective value improves along *-c*
- **Linear constraints:** each constraint defines a half-plane → when well-formed, the collection of all constraints defines a convex polyhedron
- **Solution lies on vertex of feasible region**

$$\begin{aligned} \min_{x} \quad & c^T x + d \\ \text{subject to} \quad & Gx \leq h \\ & Ax = b \end{aligned}$$



Linear Program Formulation: Example

Mars Rover Activity Planning

Activity	**Variable**	**Science reward**	**Energy use**	**Data generated**
Drive	x_d	5 pts/h	20 W	5 MB/h
Take Images	x_i	8 pts/h	10 W	20 MB/h
Analyze soil	x_a	12 pts/h	25 W	10 MB/h

Want to **maximize science reward** of spending x_d hours driving, x_i hours imaging and x_a hours analyzing soil.

Have **6 hours** of time available, **100 Wh** of energy, and **80 MB** of storage.

Linear Program Formulation: Example

Activity	Variable	Science reward	Energy use	Data generated
Drive	x_d	5 pts/h	20 W	5 MB/h
Take Images	x_i	8 pts/h	10 W	20 MB/h
Analyze soil	x_a	12 pts/h	25 W	10 MB/h

Available resources:

- **6 hours** of time
- **100 Wh** of energy
- **80 MB** of storage.

$$\max_{x_d, x_i, x_a} \quad 5x_d + 8x_i + 12x_a$$

subject to

$$x_d + x_i + x_a \leq 6$$

$$20x_d + 10x_i + 25x_a \leq 100$$

$$5x_d + 20x_i + 10x_a \leq 80$$

$$x_d, x_i, x_a \geq 0.$$

Solving the Example LP

$$\max_{x_d, x_i, x_a} \quad 5x_d + 8x_i + 12x_a$$

subject to

$$x_d + x_i + x_a \leq 6$$

$$20x_d + 10x_i + 25x_a \leq 100$$

$$5x_d + 20x_i + 10x_a \leq 80$$

$$x_d, x_i, x_a \geq 0.$$

We are now given an additional constraint that disallows driving, i.e. $\mathbf{x_d = 0}$:

Solving the Example LP

$$\max_{x_d, x_i, x_a} \quad 5x_d + 8x_i + 12x_a$$

subject to

$$x_d + x_i + x_a \leq 6$$

$$20x_d + 10x_i + 25x_a \leq 100$$

$$5x_d + 20x_i + 10x_a \leq 80$$

$$x_d, x_i, x_a \geq 0.$$

We are now given an additional constraint that disallows driving, i.e. $\mathbf{x_d = 0}$:

$$\max_{x_i, x_a} \quad 8x_i + 12x_a$$

subject to

$$x_i + x_a \leq 6$$

$$10x_i + 25x_a \leq 100$$

$$20x_i + 10x_a \leq 80$$

$$x_i, x_a \geq 0.$$

Solving the Example LP

$$\max_{x_d, x_i, x_a} \quad 5x_d + 8x_i + 12x_a$$

subject to

$$x_d + x_i + x_a \leq 6$$

$$20x_d + 10x_i + 25x_a \leq 100$$

$$5x_d + 20x_i + 10x_a \leq 80$$

$$x_d, x_i, x_a \geq 0.$$

We are now given an additional constraint that disallows driving, i.e. $\mathbf{x_d = 0}$:

$$\max_{x_i, x_a} \quad 8x_i + 12x_a$$

subject to

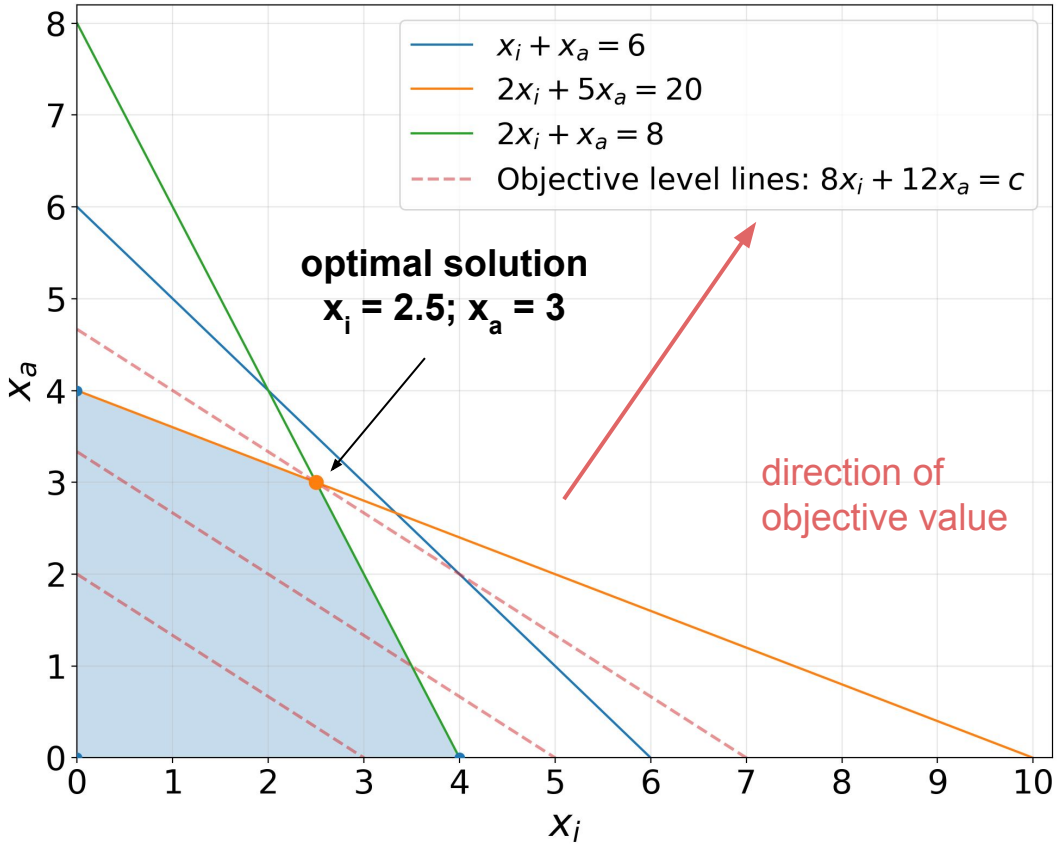
$$x_i + x_a \leq 6$$

$$2x_i + 5x_a \leq 20$$

$$2x_i + x_a \leq 8$$

$$x_i, x_a \geq 0.$$

Solving the Example LP



$$\max_{x_i, x_a} \quad 8x_i + 12x_a$$

subject to:

$$x_i + x_a \leq 6$$

$$2x_i + 5x_a \leq 20$$

$$2x_i + x_a \leq 8$$

$$x_i, x_a \geq 0.$$

Graphical Solution Was Easy in 2D, But…

…what happens when scaling the number of variables or constraints?

- We know the solution lies on a corner point (vertex) of the feasible set, so we can solve using “brute force” by checking each vertex → how many vertices?
- With **n** variables and **m** constraints, at each vertex **n constraints are active**
- Therefore, the total number of vertices can be up to ***m choose n***

Key property that enables a more efficient algorithm:

- From every feasible vertex, there is a path (connecting adjacent feasible vertices) that **never decreases cost**.
- Therefore, **don’t have to check every vertex:** simply find the better neighboring vertex at each iteration (→ Simplex algorithm)

Reinforcement Learning

An **agent** interacts with its **environment** with the goal of maximizing **reward**.

At each timestep ***t***,

- The agent finds itself in state $\mathbf{s}^{t}$, and must choose an **action** $\boldsymbol{a}^{t}$ according to its **policy** $\boldsymbol{\pi}$, which specifies how to choose an action for a given state
- After executing this action in the environment, the environment produces an updated state $\boldsymbol{s}^{t+1}$
- We compute the **reward** $\boldsymbol{r}^{t}$ associated with the transition ($\boldsymbol{s}^{t}$, $\boldsymbol{a}^{t}$, $\boldsymbol{s}^{t+1}$)
  - For simplicity, we will consider only rewards based on the current state $\boldsymbol{s}^{t}$, that is, $\boldsymbol{r}^{t} = \boldsymbol{R}(\boldsymbol{s}^{t})$, where ***R*** is some reward function computing the reward for a given state.

Markov Decision Process

The setup from the previous slide can be described by a **Markov Decision Process** (MDP), which is a tuple ***M = (S, A, P, R)***, where…

- ***S*** is the set of possible states (the state space)
- ***A*** is the set of available actions (the action space)
- ***P*** describes the dynamics of the environment by specifying the probability of each state transition given the current state and the action to apply, i.e., ***P(s' | s, a)*** is the probability that the system transitions into state ***s'*** after taking action **a**.
- ***R*** is a reward function assigning a scalar reward value to each state (or, more generally, each state transition specified by the current and next states as well as the action taken)

Measuring Cumulative Reward

We would like to maximize the **cumulative reward** starting at state **s** and acting according to the **policy *π*** up to timestep ***H***:

$$V_\pi(s) = \mathbb{E}\left[\sum_{t=0}^{H} R(s_t) \mid s_0 = s\right]$$

- This is called the **value function**
- Why is this an expectation?

- What is the expectation over, i.e. what are the random variables and their distributions?

Measuring Cumulative Reward

We would like to maximize the **cumulative reward** starting at state **s** and acting according to the **policy π** up to timestep ***H***:

$$V_\pi(s) = \mathbb{E}\left[\sum_{t=0}^{H} R(s_t) \mid s_0 = s\right]$$

- This is called the **value function**
- Why is this an expectation?
  Because of nondeterministic environment dynamics.
- What is the expectation over, i.e. what are the random variables and their distributions?
  The expectation is taken over future states s^1, …, s^H whose distribution is determined by the environment transition model P and the policy π.

Measuring Cumulative Reward

What if transitions were **deterministic**?

- Let $s^{t+1} = f(s^t, \pi(s^t))$
- Then, $V_\pi^H(s) = R(s) + R(s^1) + \ldots + R(s^H)$, where $s^1 = f(s, \pi(s))$, $s^2 = f(s, \pi(s^1))$, …

Measuring Cumulative Reward

What if transitions were **deterministic**?

- Let $s^{t+1} = f(s^t, \pi(s^t))$
- Then, $V_\pi^H(s) = R(s) + R(s^1) + \ldots + R(s^H)$,
  where $s^1 = f(s, \pi(s))$, $s^2 = f(s, \pi(s^1))$, …

But $V_\pi^H(s) = R(s) + \underbrace{R(s^1) + \ldots + R(s^H)}_{= V_\pi^{H-1}(s^1)}$ has recursive structure!

Maximizing Cumulative Reward

Using the value function, we have established how to measure how good a given policy is. But how do we find the **optimal** policy π^*?

- From the previous slide, we have that

  $V_\pi^H(s) = R(s) + V_\pi^{H-1}(s')$ with $s' = f(s, \pi(s))$

- But what if instead of committing to some policy π we acted always according to the **best action at that step?**

  $V^{H,*}(s) = R(s) + max_a\, V^{H-1,*}(f(s, a))$

The Principle of Optimality

The value (cumulative reward) is maximized if we always act according to the **best action at each step:**

$$V^{H,*}(s) = R(s) + \max_a V^{H-1,*}(f(s, a))$$

Principle of Optimality: An optimal policy has the property that whatever the initial state and initial decision are, the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decision. *(Bellman, 1957, Chap. III.3.)*

Value Iteration

The value (cumulative reward) is maximized if we always act according to the **best action at each step:**

$$V^{H,*}(s) = R(s) + \max_a V^{H-1,*}(f(s, a))$$

By “unrolling” the recursion above we can compute the optimal value function by going “backwards”!

$$V^{0,*}(s) = R(s)$$

$$V^{1,*}(s) = R(s) + \max_a V^{0,*}(f(s, a))$$

$$V^{2,*}(s) = R(s) + \max_a V^{1,*}(f(s, a))$$, …

Extracting the Optimal Policy

Once we have computed the optimal value function, how can we use this to determine how to act?

- We can simply look at the optimal value function that we have already computed and pick the best action by checking all of the precomputed values:

  $$\pi^*(s) = \text{argmax}_a \, V^{H,*}(f(s, a))$$

  …that is, the optimal action is the one that leads to the best expected next-state value.

Return to the Stochastic Case

With deterministic transitions, we had

$$V^{H,*}(s) = R(s) + \max_a V^{H-1,*}(f(s, a))$$

Returning to our stochastic transition model $P(s' \mid s, a)$ we can easily adapt the above:

$$V^{H,*}(s) = R(s) + \max_a E_{s' \sim P(\cdot \mid s, a)} V^{H-1,*}(s')$$

(note that the optimal value function is deterministic even when the transition model is stochastic!)

What about H → ∞?

In many cases, we would like to find policies which are optimal for an infinite horizon. However, note that the value function

$$V_\pi(s) = \mathbb{E}\left[\sum_{t=0}^{H} R(s_t) \mid s_0 = s\right]$$

could blow up (→ ∞) if we simply consider the infinite sum as H→ ∞.

Standard practice to avoid this notices that we generally would like to weigh **earlier rewards more heavily than very distant ones**, and introduces a so-called **discount factor** γ

Discounted Reward

Using the **discount factor γ,** we have

$$V_\pi(s) := \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t) | s_0 = s\right]$$

so the infinite series will converge as long as $\gamma < 1$ and our rewards are bounded.

Value Iteration: Stochastic Transitions and Infinite Horizon

Putting everything together, we arrive at the value iteration algorithm from the previous lecture:

Value Iteration

$V_0(s) \leftarrow 0$ for all $s \in S$;
for $k = 1, 2, \ldots$ **do**
  for *all* $s \in S$ **do**
    $V_{k+1}(s) \leftarrow \max_{a \in A} \left[R(s) + \gamma \sum_{s' \in S} P(s'|s, a) V_k(s') \right]$;

So, is RL solved?

Value Iteration **optimally solves** an MDP, but…

- Assumes we can enumerate all states
- Assumes we can enumerate all actions
- Uses knowledge of the state and transition dynamics

These are **huge obstacles** in practice! More in the next lecture.

Value Iteration

$V_0(s) \leftarrow 0$ for all $s \in S$;
for $k = 1, 2, \ldots$ **do**
  for *all* $s \in S$ **do**
    $V_{k+1}(s) \leftarrow \max_{a \in A} \left[R(s) + \gamma \sum_{s' \in S} P(s'|s,a) V_k(s') \right]$;