CME 241: Reinforcement Learning for Stochastic Control Problems in Finance

Ashwin Rao

ICME, Stanford University

Overview of the Course

- Theory of Markov Decision Processes (MDPs)
- Dynamic Programming (DP) Algorithms (a.k.a. model-based)
- Reinforcement Learning (RL) Algorithms (a.k.a. model-free)
- Plenty of Python implementations of models and algorithms
- Apply these algorithms to 3 Financial/Trading problems:
  - (Dynamic) Asset-Allocation to maximize utility of Consumption
  - Optimal Exercise/Stopping of Path-dependent American Options
  - Optimal Trade Order Execution (managing Price Impact)
- By treating each of the problems as MDPs (i.e., Stochastic Control)
- We will go over classical/analytical solutions to these problems
- Then introduce real-world considerations, and tackle with RL (or DP)

What is the flavor of this course?

An important goal of this course is to effectively blend:

- Theory/Mathematics
- Programming/Algorithms
- Modeling of Read-World Finance/Trading Problems

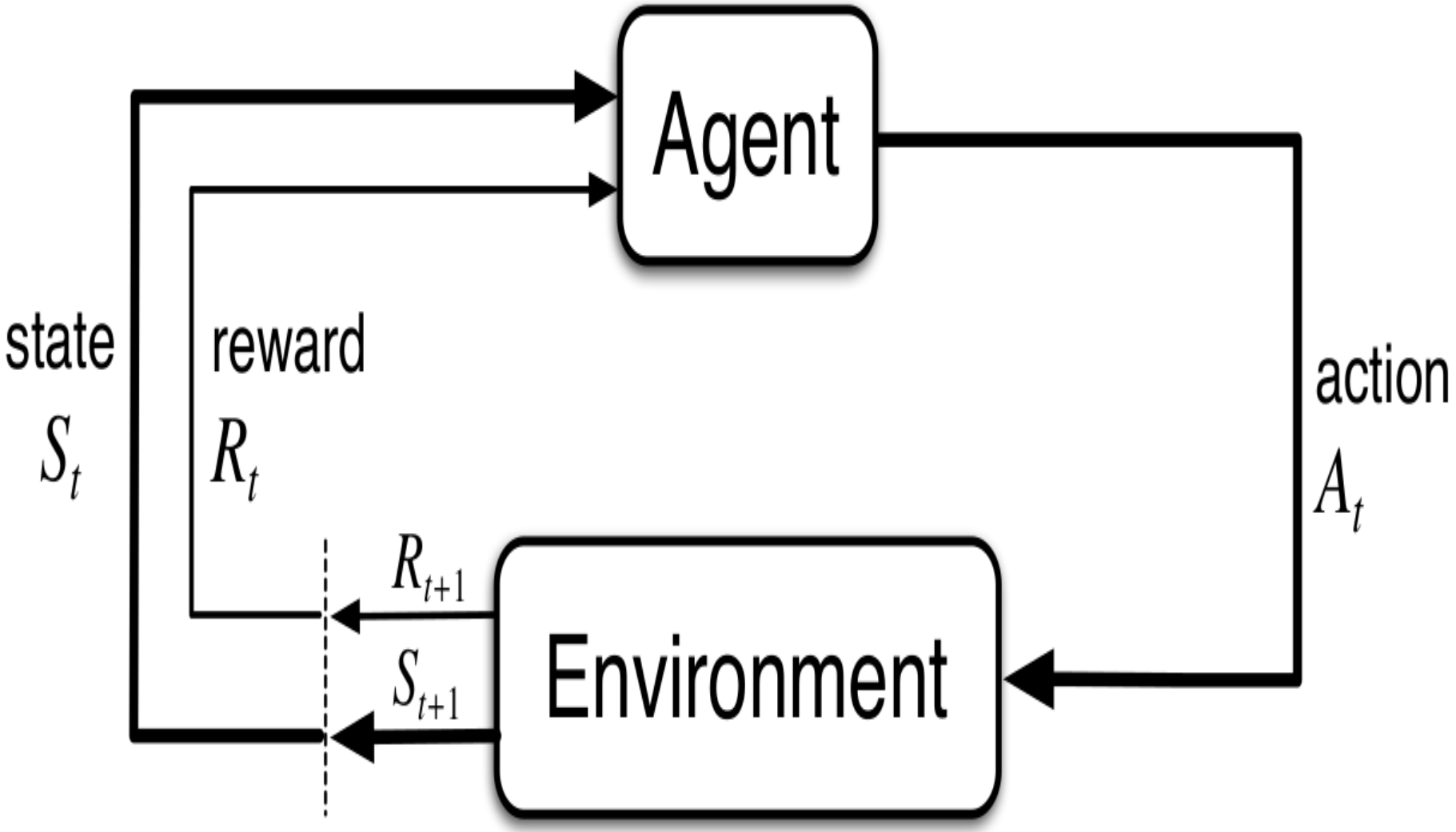
Pre-Requisites and Housekeeping

- Theory Pre-reqs: Optimization, Probability, Pricing, Portfolio Theory
- Coding Pre-reqs: Data Structures/Algorithms with numpy/scipy
- Alternative: Written test to demonstrate above-listed background
- Grade based on Class Participation, 1 Exam, and Programming Work
- Passing grade fetches 3 credits, can be applied towards MCF degree
- Wed and Fri 4:30pm - 5:50pm, 01/07/2019 - 03/15/2019
- Classes in GES building, room 150
- Appointments: Any time Fridays or an hour before class Wednesdays
- Use appointments time to discuss theory as well as your code

Resources

- I recommend Sutton-Barto as the companion book for this course
  - I won't follow the structure of Sutton-Barto book
  - But I will follow his approach/treatment
- I will follow the structure of David Silver's RL course
  - I encourage you to augment my lectures with David's lecture videos
  - Occasionally, I will wear away or speed up/slow down from this flow
- We will do a bit more Theory & a lot more coding (relative to above)
- You can freely use my open-source code for your coding work
  - I expect you to duplicate the functionality of above code in this course
- We will go over some classical papers on the Finance applications
- To understand in-depth the analytical solutions in simple settings
- I will augment the above content with many of my own slides
- All of this will be organized on the course web site

The MDP Framework (that RL is based on)



Components of the MDP Framework

- The *Agent* and the *Environment* interact in a time-sequenced loop
- *Agent* responds to [*State*, *Reward*] by taking an *Action*
- *Environment* responds by producing next step's (random) *State*
- *Environment* also produces a (random) scalar denoted as *Reward*
- Goal of *Agent* is to maximize *Expected Sum* of all future *Rewards*
- By controlling the ($Policy : State \rightarrow Action$) function
- *Agent* often doesn't know the *Model* of the *Environment*
- *Model* refers to state-transition probabilities and reward function
- So, *Agent* has to learn the *Model* AND learn the Optimal *Policy*
- This is a dynamic (time-sequenced control) system under uncertainty

Many real-world problems fit this MDP framework

- Self-driving vehicle (speed/steering to optimize safety/time)
- Game of Chess (Boolean *Reward* at end of game)
- Complex Logistical Operations (eg: movements in a Warehouse)
- Make a humanoid robot walk/run on difficult terrains
- Manage an investment portfolio
- Control a power station
- Optimal decisions during a football game
- Strategy to win an election (high-complexity MDP)

Why are these problems hard?

- *Model* of *Environment* is unknown (learn as you go)
- *State* space can be large or complex (involving many variables)
- Sometimes, *Action* space is also large or complex
- No direct feedback on "correct" *Actions* (only feedback is *Reward*)
- *Action*s can have delayed consequences (late *Reward*s)
- Time-sequenced complexity (eg: *Actions* influence future *Actions*)
- *Agent Action*s need to tradeoff between "explore" and "exploit"

Why is RL interesting/useful to learn about?

- RL solves MDP problem when *Environment Model* is unknown
- Or when *State* or *Action* space is too large/complex
- **Promise of modern A.I. is based on success of RL algorithms**
- Potential for automated decision-making in real-world business
- In 10 years: Bots that act or behave more optimal than humans
- RL already solves various low-complexity real-world problems
- RL will soon be the most-desired skill in the Data Science job-market
- Possibilities in Finance are endless (we cover 3 important problems)
- Learning RL is a lot of fun! (interesting in theory as well as coding)

Optimal Asset Allocation to Maximize Consumption Utility

- You can invest in (allocate wealth to) a collection of assets
- Investment horizon is a fixed length of time
- Each risky asset has an unknown distribution of returns
- Transaction Costs & Constraints on trading hours/quantities/shorting
- Allowed to consume a fraction of your wealth at specific times
- Dynamic Decision: Time-Sequenced Allocation & Consumption
- To maximize horizon-aggregated Utility of Consumption
- Utility function represents degree of risk-aversion
- So, we effectively maximize aggregate Risk-Adjusted Consumption

MDP for Optimal Asset Allocation problem

- *State* is [Current Time, Current Holdings, Current Prices]
- *Action* is [Allocation Quantities, Consumption Quantity]
- *Action*s limited by various real-world trading constraints
- *Reward* is Utility of Consumption less Transaction Costs
- *State*-transitions governed by risky asset movements

Optimal Exercise of Path-Dependent American Options

- RL is an alternative to Longstaff-Schwartz algorithm for Pricing
- *State* is [Current Time, History of Spot Prices]
- *Action* is Boolean: Exercise (i.e., Payoff and Stop) or Continue
- *Reward* always 0, except upon Exercise (= Payoff)
- *State*-transitions governed by Spot Price Stochastic Process
- Optimal Policy $\Rightarrow$ Optimal Stopping $\Rightarrow$ Option Price
- Can be generalized to other Optimal Stopping problems

Optimal Trade Order Execution (controlling Price Impact)

- You are tasked with selling a large qty of a (relatively less-liquid) stock
- You have a fixed horizon over which to complete the sale
- Goal is to maximize aggregate sales proceeds over horizon
- If you sell too fast, *Price Impact* will result in poor sales proceeds
- If you sell too slow, you risk running out of time
- We need to model temporary and permanent *Price Impacts*
- Objective should incorporate penalty for variance of sales proceeds
- Which is equivalent to maximizing aggregate Utility of sales proceeds

MDP for Optimal Trade Order Execution

- *State* is [Time Remaining, Stock Remaining to be Sold, Market Info]
- *Action* is Quantity of Stock to Sell at current time
- *Reward* is Utility of Sales Proceeds (i.e., proceeds-variance-adjusted)
- *Reward* & *State*-transitions governed by *Price Impact Model*
- Real-world *Model* can be quite complex (Order Book Dynamics)

Week by Week (Tentative) Schedule

- Week 1: Markov Decision Processes & Overview of Finance Problems
- Week 2: Bellman Equations & Dynamic Programming Algorithms
- Week 3: Model-free Prediction (RL for Value Function Estimation)
- Week 4: Model-Free Control (RL for Optimal Value Function/Policy)
- Week 5: RL with Function Approximation (including Deep RL)
- Week 6: Optimal Asset Allocation problem
- Week 7: Optimal Exercise of American Options problem
- Week 8: Optimal Trade Order Execution problem
- Week 9: Batch Methods (DQN, LSTDQ/LSPI), and Gradient TD
- Week 10: Policy Gradient Algorithms

Sneak Peek into a couple of lectures in this course

- Policy Gradient Theorem and Compatible Approximation Theorem
- HJB Equation and Merton's Portfolio Problem

Landmark Papers we will cover in detail

- Merton's solution for Optimal Portfolio Allocation/Consumption
- Longstaff-Schwartz Algorithm for Pricing American Options
- Bertsimas-Lo paper on Optimal Execution Cost
- Almgren-Chriss paper on Optimal Risk-Adjusted Execution Cost
- Original DQN paper and Nature DQN paper
- Lagoudakis-Parr paper on Least Squares Policy Iteration
- Sutton et al's paper on Policy Gradient
- Gradient TD Learning Algorithms

Similar Courses offered at Stanford

- AA 228/CS 238 (Mykel Kochenderfer - Autumn 2018)
- CS 234 (Emma Brunskill - Winter 2019)
- MS&E 251 (Edison Tse - Spring 2019)
- CS 332 (Emma Brunskill - Autumn 2018)
- MS&E 338 (Ben Van Roy - Spring 2019)
- MS&E 348 (Gerd Infanger - Winter 2020)
- MS&E 351 (Ben Van Roy - Winter 2019)