CS7641 Assignment 4
Markov Decision Processes
Spring 2025

1 Assignment Weight

The assignment is worth 15% of the total points.

Read everything below carefully as this assignment has changed term-over-term.

2 Objective

In some sense, we have spent the semester thinking about machine learning techniques for various forms of function approximation. It's now time to think about using what we've learned in order to allow an agent of some kind to act in the world more directly. This assignment asks you to consider the application of some of the techniques we've learned from reinforcement learning to make decisions.

The same ground rules apply for programming languages and libraries. You may program in any language that you wish insofar as you feel the need to program. As always, *it is your responsibility* to make sure that we can actually recreate your narrative, if necessary.

Please note, this class implements changes to the assignments term-over-term as we are calibrating the course incrementally. Please read through everything, even if you are submitting work from a previous semester as the requirements will likely have changed.

3 Procedure

3.1 The Problems Given to You

You are being asked to explore Markov Decision Processes (MDPs) using a combination of dynamic programming and reinforcement learning approaches.

1. You will analyze and solve two predefined MDPs:
   - **Blackjack (Discrete and Stochastic)** - A turn-based card game where actions affect future outcomes probabilistically. Gym Environment: Blackjack-v1
   - **CartPole (Continuous and Deterministic)** - A physics-based balancing problem where discretization of the state space is required for dynamic programming methods. Gym Environment: CartPole-v1

   These MDPs have distinct characteristics: **Blackjack is inherently discrete and stochastic**, whereas **CartPole is continuous and requires state discretization**. Your analysis should consider how these differences impact algorithm performance.
2. Solve both MDPs using:
   - **Value Iteration (VI)** and **Policy Iteration (PI)**: Compare convergence rates and assess how discretization influences results in the CartPole environment.
   - **SARSA and Q-Learning**: Implement and compare these model-free reinforcement learning approaches. You may use any Q-Learning variant, including **Deep Q-Networks (DQN)** or any component from the **Rainbow DQN** framework.

3. Analyze and compare results:
    - How many iterations does VI vs. PI take to converge?
    - Which method converges faster? Why?
    - How does discretization affect the CartPole solution?
    - How do SARSA and Q-Learning compare in performance?
    - What exploration strategies did you use, and how did they affect learning?

Extra Credit Opportunity:

For 5 points of extra credit, solve the Double Pendulum (Inverted Pendulum with Two Links) environment.

Warning: This is a challenging environment. Many students in the past have found this to be their favorite assignment of the course, as it can be extremely fun and rewarding. However, be aware that it requires significantly more computational resources and training time compared to Blackjack and CartPole. Proceed accordingly.

- You must use either **Deep Deterministic Policy Gradient (DDPG)** or **Soft Actor-Critic (SAC)**.
- Compare the training difficulty of the Double Pendulum with your previous MDPs.
- Provide a brief justification for your algorithm choice.

Gym Environment: Double Pendulum-v5

This extra credit is entirely optional but can be a rewarding challenge if you choose to take it on.

Additional Reading: If using a variant of DQN, you may refer to the 2017 Rainbow DQN study: (Link to study)

This extra credit is optional but requires additional time and computation.

Analysis writeup is limited to 8 pages. The page limit does include your citations. Anything past 8 pages will not be read. Please keep your analysis as concise while still covering the requirements of the assignment. As a final check during your submission process, download the submission to double check everything looks correct on Canvas. Try not wait until the last minute to submit as you will only be tempting Murphy's Law.

In addition, your report must be written in LaTeX on Overleaf. You can create an account with your Georgia Tech email (e.g. gburdell3@gatech.edu). When submitting your report, you are required to include a 'READ ONLY' link to the Overleaf Project. If a link is not provided in the report or Canvas submission comment, 5 points will be deduced from your score. Do not share the project directly with the Instructor or TAs via email. For a starting template, please use the IEEE Conference template.

3.2 Acceptable Libraries

The algorithms used in this assignment are relatively easy to implement. Existing implementations are easy to find too. Below are java and python examples.

- bettermdptools (python) `https://github.com/jlm429/bettermdptools`
- BURLAP (java) `http://burlap.cs.brown.edu/`

4 Submission Details

The due date is indicated on the Canvas page for this assignment. Make sure you have set your timezone in Canvas to ensure the deadline is accurate. We are in the Eastern Time Zone for the course.

Due Date: **Indicated as "Due" on Canvas.** Please double check you understand the correct due date.

Late Due Date [20 point penalty per day]: **Indicated as "Until" on Canvas.** The late penalty is not on a racked scale, but rather wholistic day-to-day. Meaning, if you do utilize the late penalty, you have the full 24 hours before another 20 point penalty incurs.

You must submit:

- A file named README.txt containing instructions for running your code. This needs to be submitted to Canvas. We need to be able to get to your code and your data. Providing entire libraries isn't necessary when a URL would suffice; however, you should at least provide any files you found necessary to change and enough support and explanation so we can reproduce your results on a standard Linux machine.
- A file named yourgtaccount-analysis.pdf containing your writeup (GT account is what you log in with, not your all-digits ID). This file should not exceed 8 pages.
- A 'READ ONLY' link to share your Overleaf Project link and final commit (hash) for source code in your personal repository on Georgia Tech's private GitHub. These can be in your README.txt or commented to Canvas submission.

The file yourgtaccount-analysis.pdf should contain:

- *Brief description of the MDPs*
  - Provide an overview of Blackjack and CartPole, explaining their differences in terms of state space (discrete vs. continuous), action space, and reward structure.
  - Discuss why these environments are interesting to study in the context of Markov Decision Processes (MDPs).
- *Explanation of methods*
  - Describe Value Iteration (VI) and Policy Iteration (PI) in detail, explaining how they work and how you applied them to the two MDPs.
  - Explain how SARSA and Q-Learning function, including the variant(s) used (e.g., DQN, Rainbow DQN).
  - For CartPole, describe the discretization strategy used to apply Value and Policy Iteration.
- *Analysis of results*
  - Compare the performance of VI vs. PI:
    - How many iterations were needed for convergence?
    - Which algorithm converged faster? Why?
    - Did they produce the same optimal policy?
  - Compare SARSA vs. Q-Learning:
    - How did they perform in terms of reward maximization?
    - Which exploration strategy was used, and how did it affect learning?
    - If a deep RL approach (e.g., DQN) was used, how did it compare to tabular Q-Learning?
  - Discuss the effect of discretization on CartPole's solution:
    - Did different levels of discretization impact performance?
    - Were there trade-offs between computational efficiency and accuracy?
- *Visualizations and Data-Driven Evidence*
  - Graphs showing convergence rates for different algorithms.
  - Policy heatmaps or action distributions (for Blackjack).
  - Learning curves for SARSA/Q-Learning (showing cumulative rewards over time).
  - CartPole balancing performance (e.g., episode length over training iterations).
- *Extra Credit Analysis (if completed)*
  - Explanation of the Double Pendulum problem and its complexity.
  - Justification for using DDPG or SAC.
  - Performance comparison between Double Pendulum and the other MDPs.
  - Training curves and insights on computational difficulty.
- *Conclusion*
  - Summarize key findings.

- Discuss challenges encountered and possible improvements.
- Reflect on how different RL methods performed and what insights were gained.

It might be difficult to generate the same kinds of graphs for this part of the assignment as you did in previous assignments; however, you should come up with some clever way to describe the kinds of results you produce. If you can achieve this visually all the better. However, a note of caution. Figures should remain legible in a 100% zoom. Do not try to squish figures together in specific sections where axis labels become 8pt font or less. We are looking for clear and concise demonstration of knowledge and synthesis of results in your demonstrations. Any paper that solely has figures without formal writing will not be graded. Be methodical with your space.

You may submit the assignment as many times as you wish up to the due date, but, we will only consider your last submission for grading purposes.

Note: we need to be able to get to your code and your data. Providing entire libraries isn't necessary when a URL would suffice; however, you should at least provide any files you found necessary to change and enough support and explanation so we can reproduce your results on a standard linux machine.

5 Feedback Requests

When your assignment is scored, you will receive feedback explaining your errors and successes in some level of detail. This feedback is for your benefit, both on this assignment and for future assignments. It is considered a part of your learning goal to internalize this feedback. We strive to give meaningful feedback with a human interaction at scale. We have a multitude of mechanisms behind the scenes to ensure grading consistency with meaningful feedback. This can be difficult, however sometimes feedback isn't always as clear as you need. If you are confused by a piece of feedback, please start a private thread on Ed and we will jump in to help clarify.

Previously, we have had a different rescore policy in this class which usually resulted in the same grade or lower. Many times there is a disconnect between what may be important or may have been missed in analysis. For this reason, we will not be conducting any rescore requests this term.

6 Plagiarism and Proper Citation

The easiest way to fail this class is to plagiarize. **Using the analysis, code or graphs of others in this class is considered plagiarism**. The assignments are designed to force you to immerse yourself in the empirical and engineering side of ML that one must master to be a viable practitioner and researcher. It is important that you understand why your algorithms work and how they are affected by your choices in data and hyperparameters. The phrase "as long as you participate in this journey of exploring, tuning, and analyzing" is key. We take this very seriously and you should too.

What is plagiarism?

If you copy any amount of text from other students, websites, or any other source without proper attribution, that is plagiarism. The most common form of plagiarism is copying definitions or explanations from wikipedia or similar websites. We use an anti-cheat tool to find out which parts of the assignments are your own and there is a near 100 percent chance we will find out if you copy or paraphrase text or plots from online articles, assignments of other students (even across sections and previous courses), or website repositories.

What does it mean to be original?

In this course, we care very much about your analysis. It must be original. Original here means two things: 1) the text of the written report must be your own and 2) the exploration that leads to your analysis must be your own. Plagiarism typically refers to the former explicitly, but in this case it also refers to the latter explicitly.

It is well known that for this course we do not care about code. We are not interested in your working out the edge cases in k-nn, or proving your skills with python. While there is some value in implementing algorithms yourselves in general, here we are interested in your grokking the practice of ML itself. That practice is about the interaction of algorithms with data. As such, the vast majority of what you're going to learn in order to master the empirical practice of ML flows from doing your own analysis of the data, hyper parameters, and so on; hence, you are allowed to steal ML code from libraries but are not allowed to steal code written explicitly for this course, particularly those parts of code that automate exploration. You will be tempted to just run said code that has already been overfit to the specific datasets used by that code and will therefore learn very little.

How to cite:

If you are referring to information you got from a third-party source or paraphrasing another author, you need to cite them right where you do so and provide a reference at the end of the document [Col]. Furthermore, "if you use an author's specific word or words, you must place those words within quotation marks and you must credit the source." [Wis]. It is good style to use quotations sparingly. Obviously, you cannot quote other people's assignment and assume that is acceptable. Speaking of acceptable, citing is not a get-out-of-jail-free card. You cannot copy text willy nilly, but cite it all and then claim it's not plagiarism just because you cited it. Too many quotes of more than, say, two sentences will be considered plagiarism and a terminal lack of academic originality.

Your README file will include pointers to any code and libraries you used.

If we catch you...

We report all suspected cases of plagiarism to the Office of Student Integrity. Students who are under investigation are not allowed to drop from the course in question, and the consequences can be severe, ranging from a lowered grade to expulsion from the program.

7 Version Control

- 03/24/2025 - TJL updated broken link to Double Pendulum Problem v5.
- 03/03/2025 - TJL final updates for Spring 2025 and posting to class.

References

[Col] Williams College. *Citing Your Sources: Citing Basics*. URL: `https://libguides.williams.edu/citing`.

[Wis] University of Wisconsin - Madison. *Quoting and Paraphrasing*. URL: `https://writing.wisc.edu/handbook/assignments/quotingsources`.

Original assignment by Charles Isbell. Updated for Spring 2024 by John Mansfield and Theodore LaGrow. Modified for LaTeX by John Mansfield.