COMPSCI/ECON 206

Computational Microeconomics

2022 Spring Term (Seven Week - Second)

Dear students in Econ/CS 206,

Congratulations on making fabulous milestones for Code Assignment 1. In general, most of your submissions are excellent. The major part that can be improved is the understanding of the dichotomy of environment and agents that we discussed in last week's lecture. However, as I had explained in the class, only a few in this world (including professors) are clear about the dichotomy. You don't need to feel any distress on points deducted based on a super high standard.

Please refer to the detailed feedback below.

The feedback is at your discretion to learn and grow. However, I do not expect you to submit any revisions. But if you want to consult further, I also would love to give you more advice.

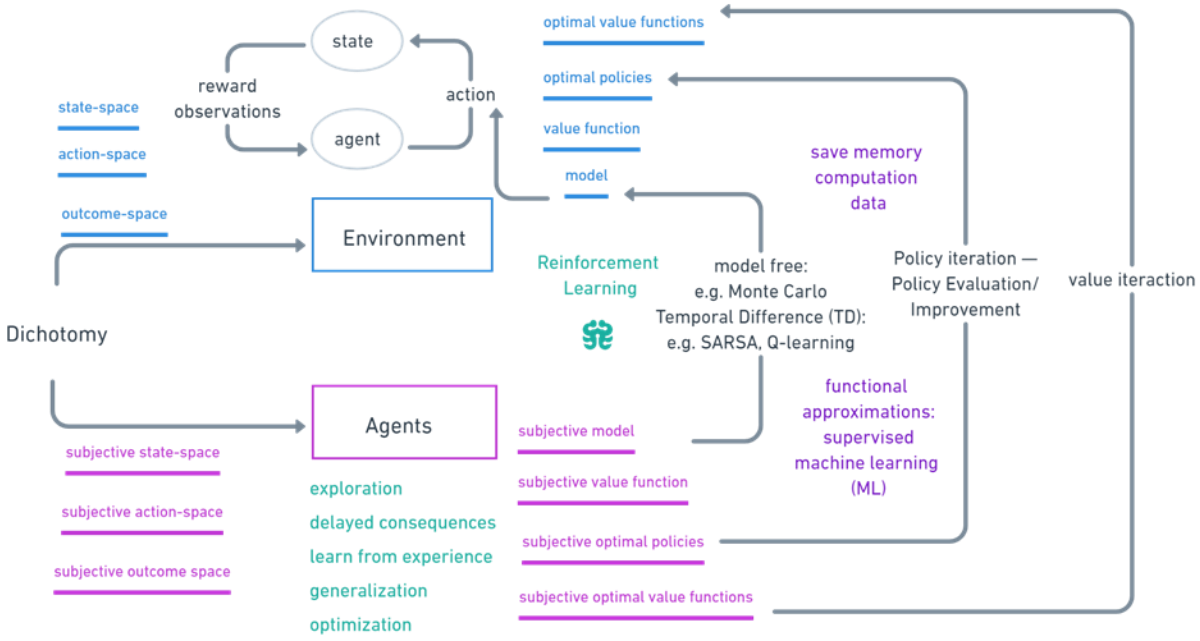
Hope you stay in good physical and mental health at this difficult time!

Cheers,

Luyao

General Feedback to "Code Assignment 1: The computational pipeline"

Research Question (RQ) 1: How to experiment with the Frozen-AI environment in openAI gym by the dichotomy of environment and agents?



As illustrated in this figure we discussed last week: any microeconomic problems can be described in a dichotomy of environment and agents. **Environment** usually models the objective situation, and **Agent** models the subjective belief of agents and inform agents' actually behavior. The actual components of Environment and Agents differ in the choices of models. For example, in reinforcement learning, **Environment** includes "state," "action," "rewards," and their interactions such as reward functions and models of state transitions; **Agent** includes the learning process to understand the environment (such as policy evaluations, state-value function, and action-state value function), the optimal sequential choices (coined optimal policy), and the optimized discounted rewards (coined optimal value function).

Lewis' (https://github.com/Lewis727/CS206/) answer with a bit revision is closest to the perfect answer:

"We shall first understand the components present in the Frozen Lake environment. The environment is the frozen lake displayed as a grid map with each grid having a state (frozen, hole, goal). **By the setting of this environment, the agent's action space must be discrete.** With different state, the agent may receive rewards or punishment as it makes a move. On the other hand, the agent receives observations as it interacts with the frozen lake environment, that is, making a move. **We can also experiment RL algorithms, such as Q learning, SARSA, and TD3, to model the agent's behaviors that models the agents' learning process of the environment and solves for agents' optimal policy.** By adjusting the environment and the agent's learning strategy (algorithm), we can experiment with the Frozen Lake environment in openAI gym by the dichotomy of environment and agents." **For example, we can change the environment by adjusting the probability of "slipping" and observe how it affects the agent's behavior.**

Research Question (RQ) 2: What are the general principles and references for lucid communication by professional markdowns and code formatting?

Most of your have answers more than perfect:

For this part, I would love to highlight the details that **Qitong** (notebook) has provided:
"Both code cells and markdown cells are incorporated to allow developers to annotate their code and visualizations with mathematical expressions and other structures to keep track of code developments and logic. The first and foremost principle for lucid communication is standard styling. Perhaps the simplest element of markdown is the heading. The number symbol # is used to indicate various sizes of headings. Although italics and bolding are offered by most markdown editors, they're easy to write out with asterisks (* for italics, ** for bolding). Using block quotes can be used to quote something from a text or something said by someone. This is done with a right arrow >. Multiple right arrows can be used to form nested block quotes. Besides, we need to tell the story in a standard format with a smooth logic. That includes dividing the whole article into different sections and listing out the idea in a specific order (with numbers or bullet points). In terms of code formatting, we can import code when necessary to further demonstrate ideas and improve the interactions. Using proper layout helps identify parts of code that belong together, are dependent on one another, and more. Another part of making code readable is to optimize its content. Here, it is important that what you write makes sense and people can understand what each element, function or variable does. Additionally, we should involve code comments to clarify the markups, such that chances are good that whenever readers looks at the work, they will be able to understand what is going on quite quickly."

I also would love to highlight the professional citations that **Habib** (notebook) has noted:

"Markdown is a powerful and simple tool to style your text according to your needs, offering great support across a range of devices. The main principle for using Markdown when sharing your code is to make sure that your code is easily readable by other humans. Indenting your line in by the use of tab, leaving empty lines in between paragraphs, and bolding, italicizing, and underlying important information all serve to more professionally convey your information to the reader. First, in order to have principled markdown, one must learn how to use the tool. Two references for markdown tutorials are markdownguide.org and markdowntutorial.com. As for the principles of writing lucid communication as part of your code, a useful guide is the Engineer's Guide to Writing Code Comments available here. In it, the author emphasizes the importance of writing comments as you go, so as to not get bogged down in increased complexity and to keep your references contained in the current document."

ntwty5@gmail.com

fanz@cs.duke.edu

tw251@duke.edu

If you ask me to write the answer, see below:

A code notebook in general includes two parts: text and code cells. The lucid communication of our notebook to others, we provide professional markdown for the text cells and code formatting for the code cells.

The first principle is to make your notebook coherent in logics, self-content in glossaries, and comprehensive in references. A useful strategy is to "put yourself in the others' shoes." Image how you would understand your own notebook with out any prior information.

For *markdown*, we can refer to the guide (https://www.markdownguide.org/) for some basic golden rules:

The definition: John Gruber (2004) creates Markdown as a lightweight markup language to add formatting elements to plaintext text documents. Markdown is widely used, for example on Reddit and GitHub. Dilinger is one of the best online Markdown editors, that convert the markdown written down in .md file into HTML. Using markdown, you can typeset basic syntax such as:

- ✓ Headings for structure: # Heading one; ## Heading two; ### Heading three
- ✓ Font style: **text** for bold; *text* for italics
- ✓ Hyperlink: [text](URL)
- ✓ Horizontal Rule: ---
- ✓ Blockquote: >
- ✓ Ordered List and Unordered list
- ✓ Code: `content`
- ✓ Image ![alt text](image.jpg)

You can refer to the following documents for more typesetting options:

- ✓ Cheat-sheet: https://www.markdownguide.org/cheat-sheet/
- ✓ Basic Syntax: https://www.markdownguide.org/basic-syntax/
- ✓ Extended Syntax: https://www.markdownguide.org/extended-syntax/
- ✓ Hacks: https://www.markdownguide.org/hacks/

Here are the additional references for creating markdown:

- ✓ John Gruber's Markdown documentation. The original guide written by the creator of Markdown.
- ✓ Markdown Tutorial. An open-source website that allows you to try Markdown in your web browser.
- ✓ Awesome Markdown. A list of Markdown tools and learning resources.

- ✓ Typesetting Markdown. A multi-part series that describes an ecosystem for typesetting Markdown documents using pandoc and ConTeXt.
- ✓ A Crash Course: https://www.youtube.com/watch?v=HUBNt18RFbo
- ✓ The Github: https://docs.github.com/en/get-started/writing-on-github/getting-started-with-writing-and-formatting-on-github/basic-writing-and-formatting-syntax

Here are the markdown authoring applications:

- ✓ **Mac:** MacDown, iA Writer, or Marked 2
- ✓ **iOS / Android:** iA Writer
- ✓ **Windows:** ghostwriter or Markdown Monster
- ✓ **Linux:** ReText or ghostwriter
- ✓ **Web:** Dillinger or StackEdit

For *code formatting*, the golden rules are to improve expandability. You can confirm the check list:

- ✓ Write your code in simple and logical structure (e.g., following the PEP-8 Standard)
- ✓ Write *comments* begin with a hash (number sign) (#) for signal line explanations
- ✓ Write *docstring* that describe modules, classes and functions

"""
docstring
docstring
docstring
"""

- ✓ Add necessary indention, black spaces, line spaces manually or using the tools such as "black"

you can refer to:

- ✓ **PEP-8 Standard**: https://realpython.com/python-pep8/
- ✓ **Format the code with black**:(Blanks) https://github.com/psf/black (YouTube)
- ✓ **Github documentations**: https://github.com/realpython/python-guide/blob/master/docs/writing/documentation.rst#id19

Finally, you can also consider Sphinx (https://www.sphinx-doc.org/en/master/) for intelligent and beautiful (python) documentation.

Research Question (RQ) 3: How to tell stories using the Frozen-AI environment in openAI gym with lucid communication by professional markdowns and code formatting in Jupyter Notebook?

All of you did a great job to share your own insights:

Lewis Highlight

"Writing this tutorial in Jupyter Notebook with markdowns and code formatting significantly increased the communication efficiency and interactivity, compared to plain text stories. First, by mixing code cells and markdowns, we are able to break down steps and explain code in vivid details. Second, a good story should be clearly structured. In the case of the *Frozen Lake Experiment*, the author introduces the environment, stochastic vs. deterministic, maps, and conclusion. **Third, the use of images and insertion of hyperlinks not only make this more visual but also give readers more references if they are interested.** Moreover, the code comments make the code more understandable."

Qitong highlight

"The article can be a great example to tell the whole story. Firstly, it should have a clear logic flow, mainly on how to build the environment and then how to create the game. The article starts with creating the Frozen Lake environment and develops step by step according to the algorithm towards rhe conclusion. That offers readers an acceptable learning curve when reading the article and makes it easier to digest abstract concepts by explaining one by one. **Secondly, the article has a good structure with different sub-sections. Having got diverse sub-titles for each section, the article successfully improves the readability by breaking a giant goal into small pieces and leaves readers with several small tasks to complete.** Thirdly, it illustrates code, hyperlinks and generated diagrams to make the story more vivid and understandable. That provides readers chances to have a try themselves or to check the results to have a better insight in their concept and given code. The hyperlink also makes the further readings more accessible to readers and improves the efficiency."

Habib highlight

"For my assignment, I used Colab. Colab was a great tool as it allowed me to have cells for code and cells for text, allowing for different programming languages for each use. The principle for effective storytelling using the Frozen-AI environment in openAI gym is to explain the intuition of the algorithm in every step of the way. The reader should be introduced, through a text cell, to the what, the how and the why of each code cell. In this way, the logical reasoning of the author is transmitted to the reader through logical steps, without any jumps or holes in mutual understanding. This is very useful when the story is complex as the reader can be lost, lose interest, and therefore lucid communication is not achieved. **A good example in The Frozen Lake article (Mendes 2019) is the Stochastic vs Deterministic section. The act of contrasting the two different settings gracefully elucidates the probabilistic aspect of the random trials.**"

Ray highlight

"See above. I think the best communication is to combine text explanation with code demonstration. We first tell the background, procedures, and what we are doing in text in markdown blocks. Then, we insert the coding blocks to present how it is actually implemented and

what the result will be like. **To make it more understandable, we might need to use metaphors or refer to relevant examples to help explain it. Besides, it is also notable to follow the references and conventions listed in research question 2**."

Appendix: Code Assignment 2 Breakdown

1. **Replicate the assigned content** (https://reinforcement-learning4.fun/2019/06/16/gym-tutorial-frozen-lake/) **in a Júpiter notebook**. (A VM with Ubuntu 20.04 is preferred but Google Colab is also acceptable.) ---(8 points)
2. **You must properly cite the source in your markdown** ---(1 point)
   [reference: https://www.scribbr.com/citing-sources/cite-a-website/]
3. **Provide a brief reflection on the following three questions directly in the Jupiter notebook**

- **Research Question (RQ) 1**: How to experiment with the Frozen-AI environment in openAI gym by the dichotomy of environment and agents? (2 points)
- **Research Question (RQ) 2**: What are the general principles and references for lucid communication by professional markdowns and code formatting? (2 points)
- **Research Question (RQ) 3**: How to tell stories using the Frozen-AI environment in openAI gym with lucid communication by professional markdowns and code formatting in Jupyter Notebook? (2 points)