Problem 1. We will implement the deep deterministic policy gradient (DDPG) and soft actor-critic (SAC) to solve the `LunarLanderContinuous-v2` problem shown in Fig. 1

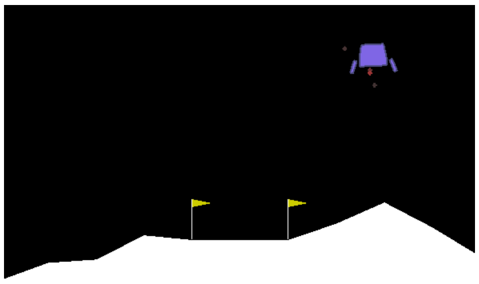


Fig. 1. The `LunarLanderContinuous-v2` environment

There are two continuous actions: the main engine and the side engine. The goal is to land smoothly on the landing pad centered between the two yellow flags. The problem is considered solved if the episodic return is 200 (land smoothly). We have provided you with the base code that includes the agent-environment interaction and actor-critic network initialization (in Python and PyTorch Framework). What you need to do is to implement the updates of the neural networks for DDPG and SAC. The code snippet can be found in the files `hw3_ddpg.ipynb` and `hw3_sac.ipynb`, or alternatively, in `ddpg.py` and `sac.py`.

```
def update(self, buffer):
    # sample from replay memory
    t = buffer.sample(self.batch_size)

    # TO DO: Perform the updates for the actor and critic networks
```

You are allowed to make certain modifications to the base code, such as adjusting the hyperparameters or the reward function. We do not restrict how many episodes it experiences to get to solving. Please submit your Python script `ddpg.py` and `sac.py` with the filled `update(self, buffer)` function, or submit your `hw3_ddpg.ipynb` and `hw3_sac.ipynb`. In addition, please attach the plot showing the episodic return over the course of training as we did in hw2. You can use any of the plotting libraries to generate the figure. One example is the Tensorboard toolkit that we discussed in hw2. (It is encouraged but not mandatory that you conduct a comparison between DDPG and SAC.)

Note that this environment can take a very long time to solve. Don't worry if your agent cannot achieve an episodic return of 200 as we value implementation and the progress toward learning the optimal policies.

1 Setup

This section will prepare all the prerequisites for running the homework program. If you are familiar with virtual environments and Python for deep learning please feel free to skip this section.

We will create a virtual environment to organize all the dependencies for our project. Then we will install the PyTorch deep learning framework, Gym reinforcement learning environment, and the Tensorboard visualization toolkit.

1.1 Create a conda Environment

1.1.1 Install Anaconda or Miniconda

We will be using Conda to create and manage our virtual environments. To use conda please install Anaconda on your system first:

https://docs.anaconda.com/anaconda/install/#installation

Anaconda contains many data science tools and will take 3GB. If your system has limited disk space, you can alternatively install the Miniconda instead of the full Anaconda:

https://docs.conda.io/en/latest/miniconda.html

1.1.2 Create and open a conda Environment

Step 1: After installing Anaconda, for Windows machines, please search and open "Anaconda Prompt" from the start menu. For Linux/Mac machines, just open the terminal.

Step 2: Create a conda environment by typing

```
$ conda create -n myenv python=3.10
```

in the Anaconda prompt or the terminal. You can replace `myenv` with your preferred name. The `$` symbol indicates the shell command.

More details about creating and managing an environment can be found at

https://docs.conda.io/projects/conda/en/latest/user-guide/tasks/manage-environments.html

Step 3: Activate the conda environment by typing the following in the Anaconda prompt or the terminal:

```
$ conda activate myenv
```

Now you should see something like `(myenv) $` which means the environment is activated in this terminal.

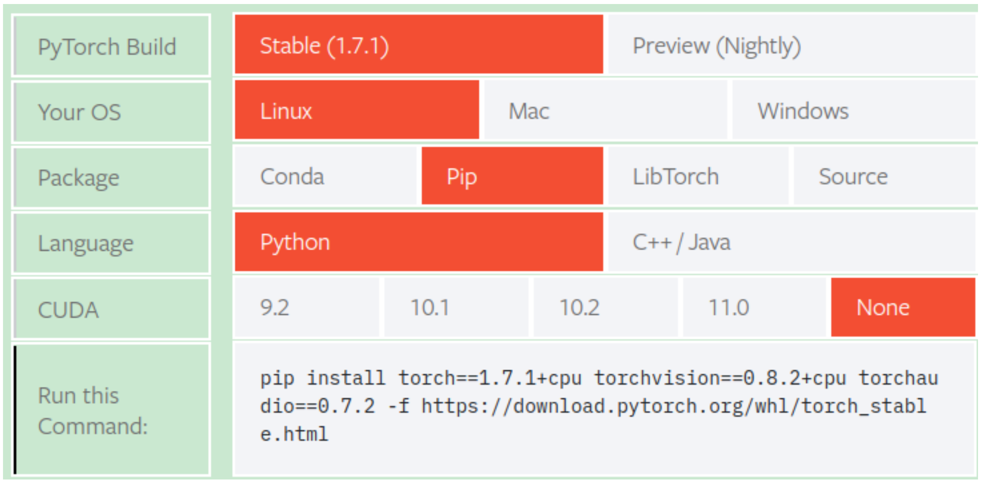
We will install all the following packages in the environment we've just created. So please keep the environment activated and use the same terminal/prompt for all the following installations.

1.2 Install PyTorch

Go to

https://pytorch.org/get-started/locally/

and select your appropriate setup. For example, mine looks like the following:



PyTorch Build	Stable (1.7.1)	Preview (Nightly)			
Your OS	Linux	Mac	Windows		
Package	Conda	Pip	LibTorch	Source	
Language	Python	C++ / Java			
CUDA	9.2	10.1	10.2	11.0	None

Run this Command:

```
pip install torch==1.7.1+cpu torchvision==0.8.2+cpu torchaudio==0.7.2 -f https://download.pytorch.org/whl/torch_stable.html
```

Please copy and paste the Run this Command in your terminal/prompt and run it. By selecting CUDA=None, the neural network training will be done on the CPU. If your machine has a CUDA supporting GPU please feel free to select the appropriate CUDA version and modify the code to run the training process on GPU.

1.3 Install OpenAI Gym

Run the following in your terminal/prompt:

```
(myenv)$ pip install swig==4.1.1
(myenv)$ pip install gymnasium[box2d]
```

More details about Gym can be found at https://gymnasium.farama.org/

You may need to download and install the Microsoft C++ build tools in:

https://visualstudio.microsoft.com/visual-cpp-build-tools/

1.4 Install Tensorboard

Run the following in your terminal/prompt:

```
(myenv)$ pip install tensorboard==2.13.0
```

More details can be found at

https://pytorch.org/tutorials/intermediate/tensorboard_tutorial.html

1.5 Install Stable-Baselines3 (Optional)

Stable-Baselines is a set of high-quality reinforcement learning algorithm implementations. In our homework, we will not use them as we will develop our own. However, you can compare your results with these implementations. To install Stable-Baselines run the following in your terminal/prompt:

```
(myenv)$ pip install "stable-baselines3[extra]>=2.0.0a4"
```

More details can be found at

https://stable-baselines3.readthedocs.io/en/master/index.html

2 Run the Codes

In the opened terminal, navigate to the directory containing the homework files. Then run the code:

```
(myenv)$ python run_ddpg.py
```

Or:

```
(myenv)$ python run_sac.py
```

3 Visualize the Results

To visualize the training curves we could use Tensorboard. Please open a new terminal and activate the conda environment:

```
$ conda activate myenv
```

Then navigate to the homework directory containing a folder called `tensorboard` and run

```
(myenv)$ tensorboard --logdir=tensorboard
```

The terminal will display an HTTP path. Copy and paste that path to your browser. Then Tensorboard should lively show and update the training logs we've declared in the `run_ddpg.py` or `run_sac.py` file.