Homework 4 Template

Use this template to record your answers for Homework 4. Add your answers using LaTeXand then save your document as a PDF to upload to Gradescope. You are required to use this template to submit your answers. **You should not alter this template in any way** other than to insert your solutions. You must submit all 7 pages of this template to Gradescope. Do not remove the instructions page(s). Altering this template or including your solutions outside of the provided boxes can result in your assignment being graded incorrectly. You may lose points if you do not follow these instructions.

You should also save your code as a .py or .zip file and upload it to the **separate** Gradescope coding assignment. Remember to mark all teammates on **both** assignment uploads through Gradescope.

Instructions for Specific Problem Types

On this homework, you must fill in (a) blank(s) for each problem; please make sure your final answer is fully included in the given space. **Do not change the size of the box provided.** For short answer questions you should **not** include your work in your solution. Only provide an explanation or proof if specifically asked. Otherwise, your assignment may not be graded correctly, and points may be deducted from your assignment.

Fill in the blank: What is the course number?

10-703

Problem 0: Collaborators

Enter your team's names and Andrew IDs in the boxes below. If you do not do this, you may lose points on your assignment.

Name 1:	Victoria Kalinovich	Andrew ID 1:	vkalinov
Name 2:	Madhuri Raman	Andrew ID 2:	madhurir
Name 3:		Andrew ID 3:	

Problem 1: TD3 (50 pts)

1.1 Prove that overestimation exists for discrete actions (5 pts)

$$Q(s,a) \leftarrow r + \gamma \max_{\tilde{a}} Q^{approx}(s', \tilde{a}) = r + \gamma \max_{\tilde{a}} (Q^{true}(s', \tilde{a}) + \epsilon)$$

where $\tilde{a}$ is the clipped noisy action, and s', r are the new state and reward respectively. But, because we use a function approximation, we know our approximated Q values have some amount of error ϵ. Each step we update using the maximum operator. By nature of the maximum estimator, our update prefer larger positive epsilons since this makes $Q^{true}(s', \tilde{a}) + \epsilon$ larger so a fixed $s^{'}$, $a^{'}$. This max operation does not preserve the original intention of the $E[\epsilon] = 0$. Therefore when we average over errors for a set state and action, we will select the largest positive ϵ's so:

$$\max_{a'} Q^{true}(s', a') \leq \max_{a'} Q^{true}(s', a') + \mathbb{E}_{\epsilon}[\epsilon] = \mathbb{E}_{\epsilon}[\max_{a'} Q^{true}(s', a') + \epsilon]$$

$$\leq \mathbb{E}_{\epsilon}[\max_{a'}(Q^{true}(s', a') + \epsilon)] \text{ by Jensen's inequality because } f(x) = max(x) + \epsilon \text{ is convex.}$$

1.2 Why is underestimation not a concern (5 pts)

Due to our update rule, $Q(s,a) = r + \gamma \max_{\tilde{a}} Q(s', \tilde{a})$, taking the maximum over our action-value estimates, we do not need to worry about propagating underestimation errors, since these under-estimated state-action values will most likely not be selected in our update rule when we take the max over state-action values. Therefore, under-estimation is much less of a concern since the error will not accumulate through our training.

1.3 Does variance affect target Q value? (5 pts)

Yes, because our update, $Q(s,a) = r + \gamma \max_{\tilde{a}} Q(s', \tilde{a})$, is equivalent to $Q(s,a) = r + V(s')$. Therefore, the variance of $V(s')$ will effect the variance of the updated Q-value $Q(s,a)$ by property of variance for sums of random variable. With large variances of $V(s')$ we can get very different values for our target Q, depending our batch samples. It's important to note that the $V(s')$ is affected by γ's, so this hyper-parameter could also effect target Q's.

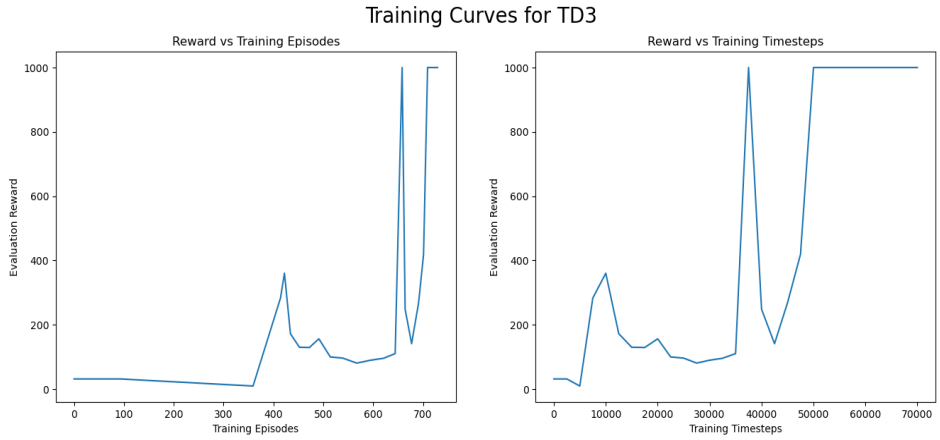
1.4 Propose a modification of Target Policy Smoothing for discrete actions (5 pts)

We can modify Target Policy Smoothing to be useful for discrete actions utilizing the entropy of the policy in our actor update. This will assure that we are not over-fitting to the value since we will be maximizing over not only the action-value, but also the entropy of the policy which will favor a flatter policy distribution over actions. Specifically our loss for the critic will be:

$$J(\phi) = \frac{1}{N} \sum Q_{\theta_1}(s, a)|_{a=\pi_\phi(s)} + H_{\pi_\phi}(a|s)$$

where $H_{\pi_\phi}(a|s)$ is the entropy of the policy.

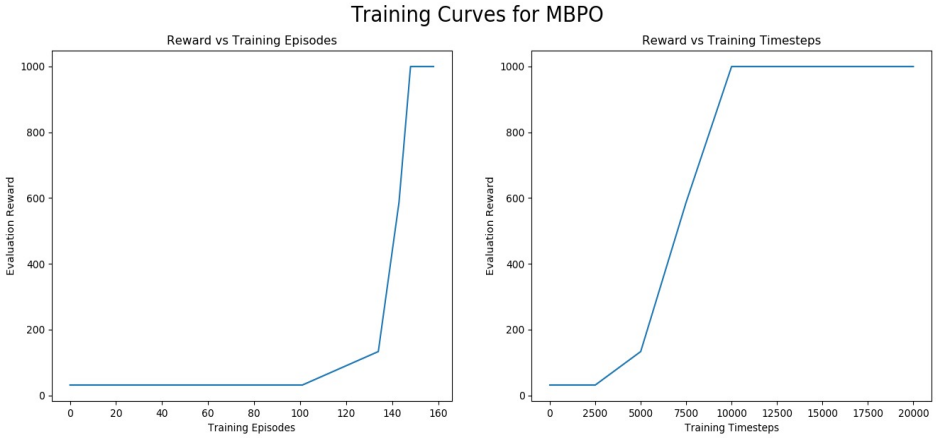
1.5 Training Curves (30 pts)



Note that we reached convergence (as defined by the TA's and starter code) around 60k training time-steps, but we ran about 10k longer to verify the behavior.

Problem 2: MBPO (43 pts)

2.1 Training Curves (30 pts)



We see that for both algorithms the evaluation rewards start very low, around 200, and they very quickly increase. However, TD3 has periods of high evaluation reward before converging, whereas MBPO does not. MPBO uses far less real environment steps, it converged in about 20k whereas TD3 takes 60k real environment time-steps. Ultimately, they both converged with 5 in a row evaluation rewards over 990.

2.2 Wall clock time for TD3 vs. MBPO (5 pts)

Wall clock time until convergence for MBPO: 2 hours
Wall clock time until convergence for TD3: 20 minutes
(Note hardware: MacBook Pro (15-inch, 2018), Processor 2.2 GHz 6-Core Intel Core i7, Memory 16 GB 2400 MHz DDR4, Graphics Intel UHD Graphics 630 1536 MB)

We think MBPO takes longer by wall clock time mainly due to the $B \times M$ total model roll-outs we perform for each real environment step. This is step doesn't occur in TD3, so it saves clock time each iteration.

2.3 Environment in which MBPO runs faster than TD3 (3 pts)

MBPO would likely run faster by wall clock time than TD3 when the environment is very slow to query/collect data and thus it takes long to fill the environment experience buffer. MBPO queries the environment less often (only every G timesteps) than TD3 (every timestep) in terms of policy network updates.

2.4 Addressing model exploitation (5 pts)

One way we handle model exploitation is by using the hyper-parameter k or the roll-out horizon. This is helpful since it restricts model roll-outs to shorter steps which are more likely to accurately reflect the environment dynamics, since there is less uncertainly in these small roll-outs. Therefore our models has less opportunity to take advantage of any dynamics model errors.
Another way we handle model exploitation is by using an ensemble of probabilistic dynamics models instead of a single dynamics model. These assure that our policy model can generalize over all of these ensemble models, thus it cannot over-fit or exploit inaccuracies in just one of the dynamics models.

Extra (2pts)

Feedback (1pts): You can help the course staff improve the course for future semesters by providing feedback. You will receive a point if you provide actionable feedback **for each of the following categories**.

What advice would you give to future students working on this homework?

> The most confusing part for us was the MBPO implementation, since the starter code did not match how we were implementing it. Recitation was very helpful, but it would have been great it if was earlier since we had already started. Also the variable type explanations in the starter code was confusing as times (e.g. the predict function in pe_model.py). For future students we would recommend reading/understanding the starter code early (before implementing) so it's easier when you do code, and you don't waste time since they gave us gave us so many helper functions.

Time Spent (1pt): How many hours did you spend working on this assignment? Your answer will not affect your grade.

Alone	1.5
With teammates	8
With other classmates	0
At office hours	1