Homework 5 Template

Use this template to record your answers for Homework 5. 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 11 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 should also export your code as a .py 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 blanks 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.

Fill in the blank: What is the course number?

$\boxed{10703}$

Problem 0: Collaborators

Enter your team members' names and Andrew IDs in the boxes below. If you worked in a team with fewer than three people, leave the extra boxes blank.

Name 1:	Amy Jiang	Andrew ID 1:	amyjiang
Name 2:	Manyung Hon	Andrew ID 2:	mehon
Name 3:	Boxiang Fu	Andrew ID 3:	boxiangf

Problem 1: Model-Based Reinforcement Learning with PETS (105 pt)

1.1 Model-based Predictive Control (25 pts)

1.1.1 CEM (without MPC) with ground-truth dynamics (10 pt)

Success percentage 86%

1.1.2 Random sampling with ground-truth dynamics. (10 pt)

Success percentage without MPC 74%

Success percentage with MPC 92%

How does the performance of random sampling performance compare to that of CEM?

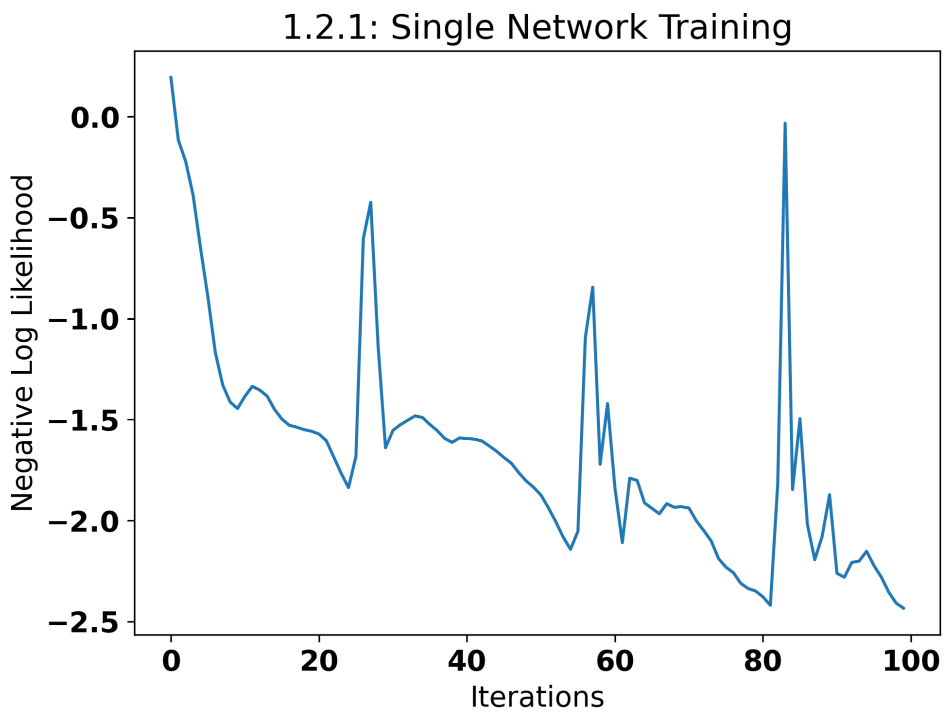
Without MPC, random action sampling performs worse than CEM. This makes sense since CEM works to move the action distribution closer to high performing elites and therefore increases chances of success. Random sampling with MPC performs better than CEM without MPC.

1.1.3 MPC vs. open-loop control (5 pt)

MPC is better in highly dynamic/noisy environments or when our dynamics model is incomplete/inaccurate because it allows us to replan at every timestep given new data. However, this is computationally expensive, so when the environment is well characterized and known it can be better to use open-loop control. Additionally, MPC is still inherently model-based so if there are egregious errors in the model, this can cause the resulting policy to fail.

1.2 Single probabilistic network (30 pts)

1.2.1 Training loss plot (4 pt)



1.2.2 MPC with random sampling (10 pt)

Success percentage	34%

1.2.3 MPC with CEM (10 pt)

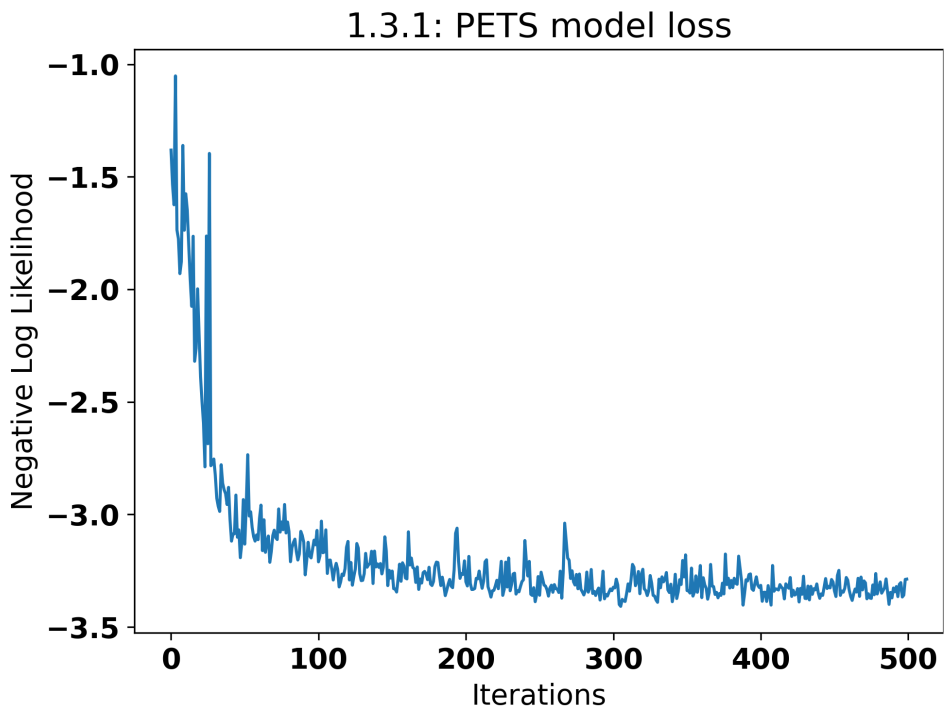
Success percentage	90%

1.2.4 Random sampling vs. CEM (6 pt)

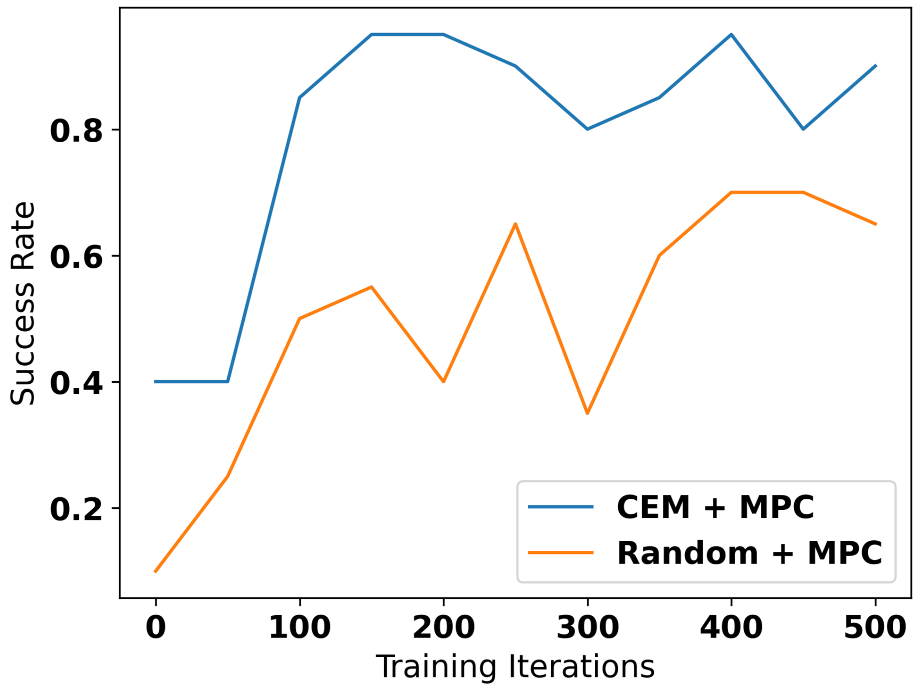
CEM performs better than random sampling, especially when they are used in the context of MPC on a learned dynamics model (compared to with the ground truth dynamics as discussed in the previous question). MPC with this model performed well and was able to beat some performance with ground truth dynamics without MPC. Generally, the derived policy succeeds when it is able to learn from a relatively accurate model of the world dynamics and is able to reason about it to pick the best action (through MPC, CEM). If the learned model is error-prone, the derived policy will also fail.

1.3 MBRL with PETS (15 pts)

1.3.1 Training loss plot (10 pt)

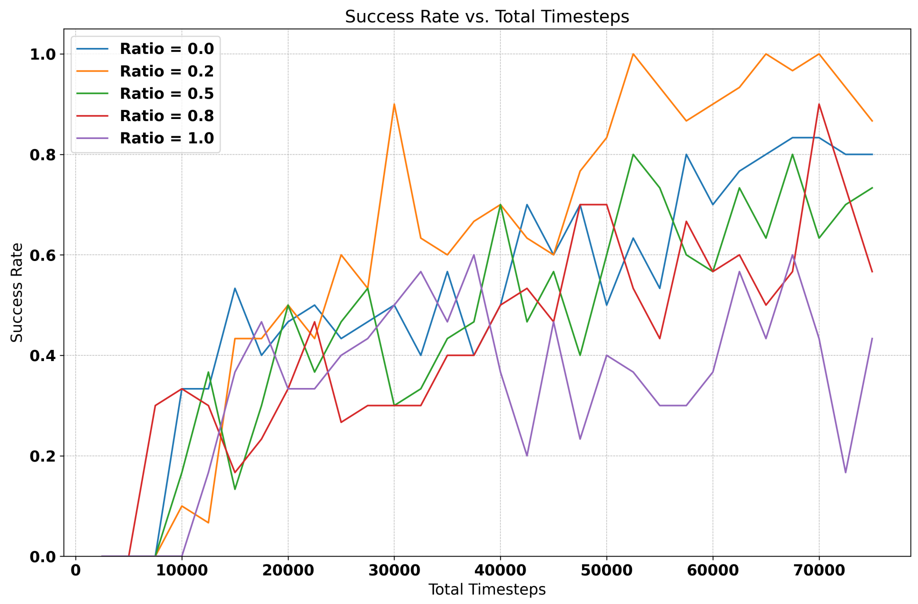


1.3.2 Test percentage of successes plot (5 pt)



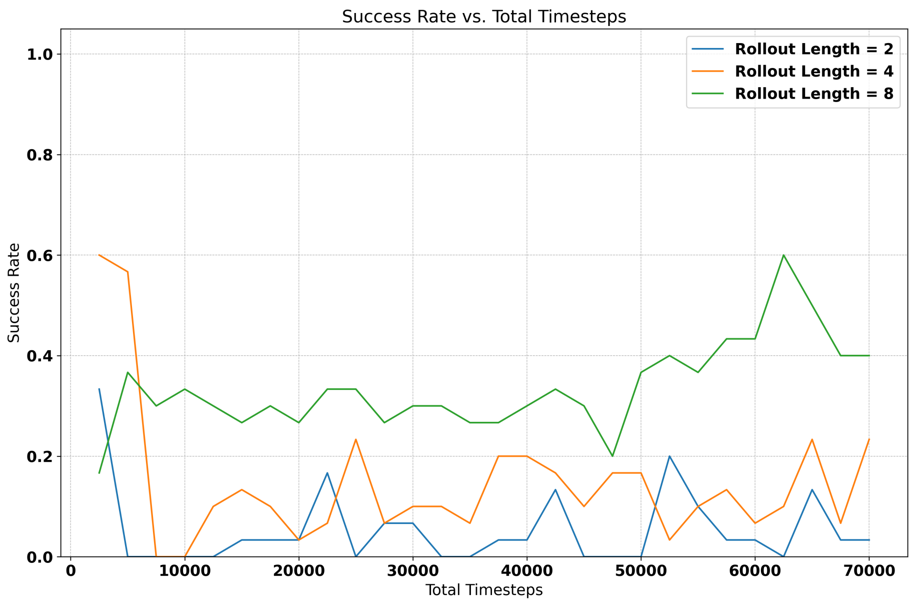
1.4 Training with synthetic data (30 pts)

1.4.1 Ablation over model vs real data (10 pt)



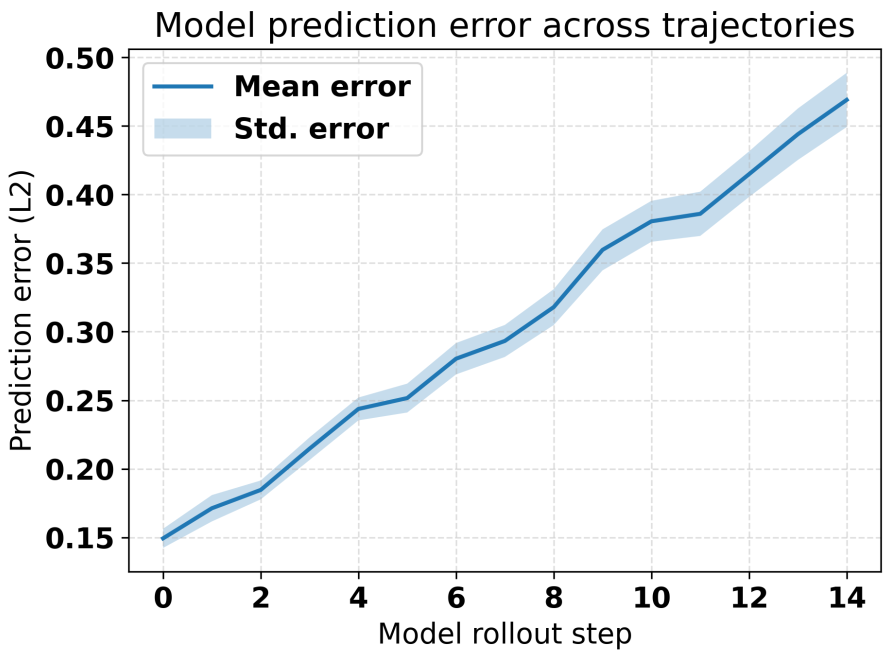
The success rate of trained policies over training timesteps is best with a data ratio of 0.2 and decreases as the ratio is increased beyond that optimum. This is caused by compounding errors from model inaccuracies. When synthetic data occupies more and more of the experience buffer, errors in the learned dynamics model that result in faulty state transitions and rewards can mislead the policy to learn more incorrect actions. However, having some synthetic data in the buffer can help generate more diverse experiences to help the policy learn faster, as shown by the performance improvement of the 0.2 ratio agent over the 0.0 ratio agent.

1.4.2 Ablation over the rollout lengths (10 pt)



Increasing the rollout length increases the success rate of trained policies. This effect is close to uniform across all training timesteps. With longer rollouts, the agent has more data about the long-horizon impacts (rewards) of its actions and there is reduced bias from bootstrapped value predictions. This allows the agent to learn better policies and increase its success rate. Additionally, longer rollout lengths increases sample efficiency by yielding more data from the same number of real environment interactions.

1.4.3 Model error over rollout lengths (10 pt)



Yes, increasing rollout length clearly increases the mean model prediction error even considering the variance bounds.

1.4.4 MBRL vs Policy Gradient (5pt)

MBRL is preferred to policy gradient methods when permitted interaction with the environment is limited (ex. ”failing” is expensive or there is limited time to explore), since MBRL is very sample efficient. Additionally, we would prefer MBRL when we want the model to learn actions to achieve different types of goals in the same environment or similar related environments because we can reuse training progress from the learned dynamics model. Moreover, the integration with MPC allows MBRL methods to reason about unseen states, disturbances, and constraints through an understanding of the environment’s dynamics.

Feedback

Feedback: You can help the course staff improve the course for future semesters by providing feedback. You will receive a point of you provide actionable feedback. What was the most confusing part of this homework, and what would have made it less confusing?

The most confusing part was that the algorithms presented in the instructions PDF differed from the provided implementation in the skeleton code in several functions, causing it to be unclear which parts of the algorithm were already implemented for us and which we needed to implement. It would also be helpful to mention that the Pushing2D environment is a custom environment that is included in the starter code, not from Gym/Gymnasium libraries, and to look there for helpful functions for getting observations.

Collaboration: Detail the work division amongst your group below.

Each group member worked on a different code file independently at first: model.py, mpc.py, or train.py, and conferred with each other to debug and answer each others' questions. Since the code parts had to be run together, we got together to merge, debug, and run the code on one person's computer.

Time Spent: How many hours did you spend working on this assignment? Your answer will not affect your grade. Please average your answer over all the members of your team.

Alone	15
With teammates	5
With other classmates	0
At office hours	0