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 9 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?

10-703

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:	Ethan Cheong	Andrew ID 1:	echeong
Name 2:	Al Hassan	Andrew ID 2:	akhassan
Name 3:	Clement Ou	Andrew ID 3:	clemento

Problem 1: Model-Based Reinforcement Learning with PETS (100 pt)

1.1 Model-based Predictive Control (25 pts)

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

Success percentage 80%

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

Success percentage without MPC 64%

Success percentage with MPC 88%

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

Random sampling performs worse than CEM without MPC. However, Random sampling combined with MPC performs better than CEM without MPC.

1.1.3 MPC vs. open-loop control (10 pt)

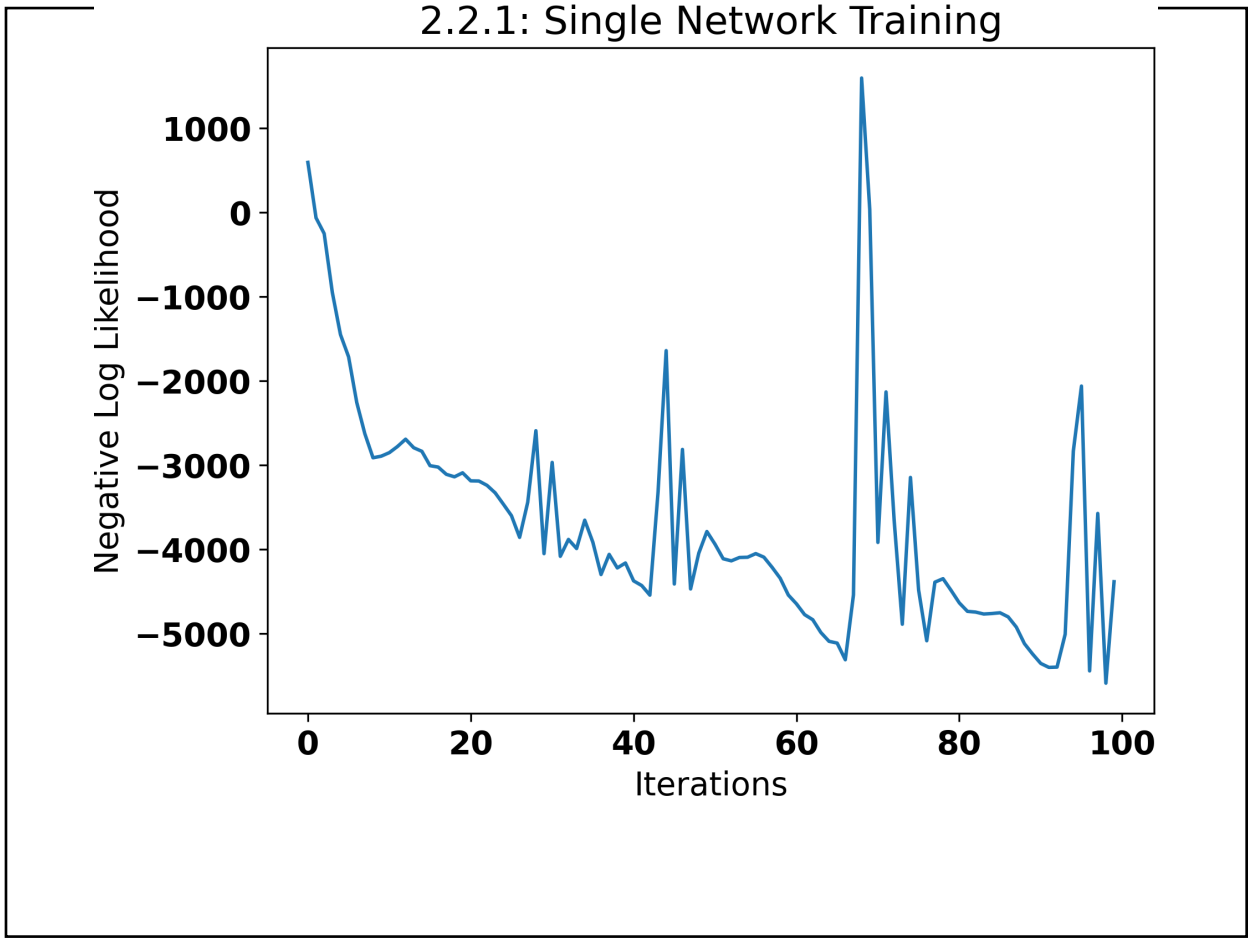
MPC performs better in environments with uncertainty and noise, while open-loop control works well in deterministic environments with perfect dynamics models.

MPC's main strengths lie in its adaptability. By replanning at each timestep, it can handle unexpected changes in the environment and maintain performance even with actuation noise. The continuous replanning also allows it to recover from suboptimal actions, as demonstrated by improving random sampling success from 64% to 88% in our experiments.

However, MPC comes with tradeoffs. It requires significant computational resources for replanning at each timestep, needs accurate state estimation to be effective, and is more complex to implement than open-loop control.

1.2 Single probabilistic network (40 pts)

1.2.1 Training loss plot (10 pt)



1.2.2 MPC with random sampling (12 pt)

Success percentage 20%

1.2.3 MPC with CEM (12 pt)

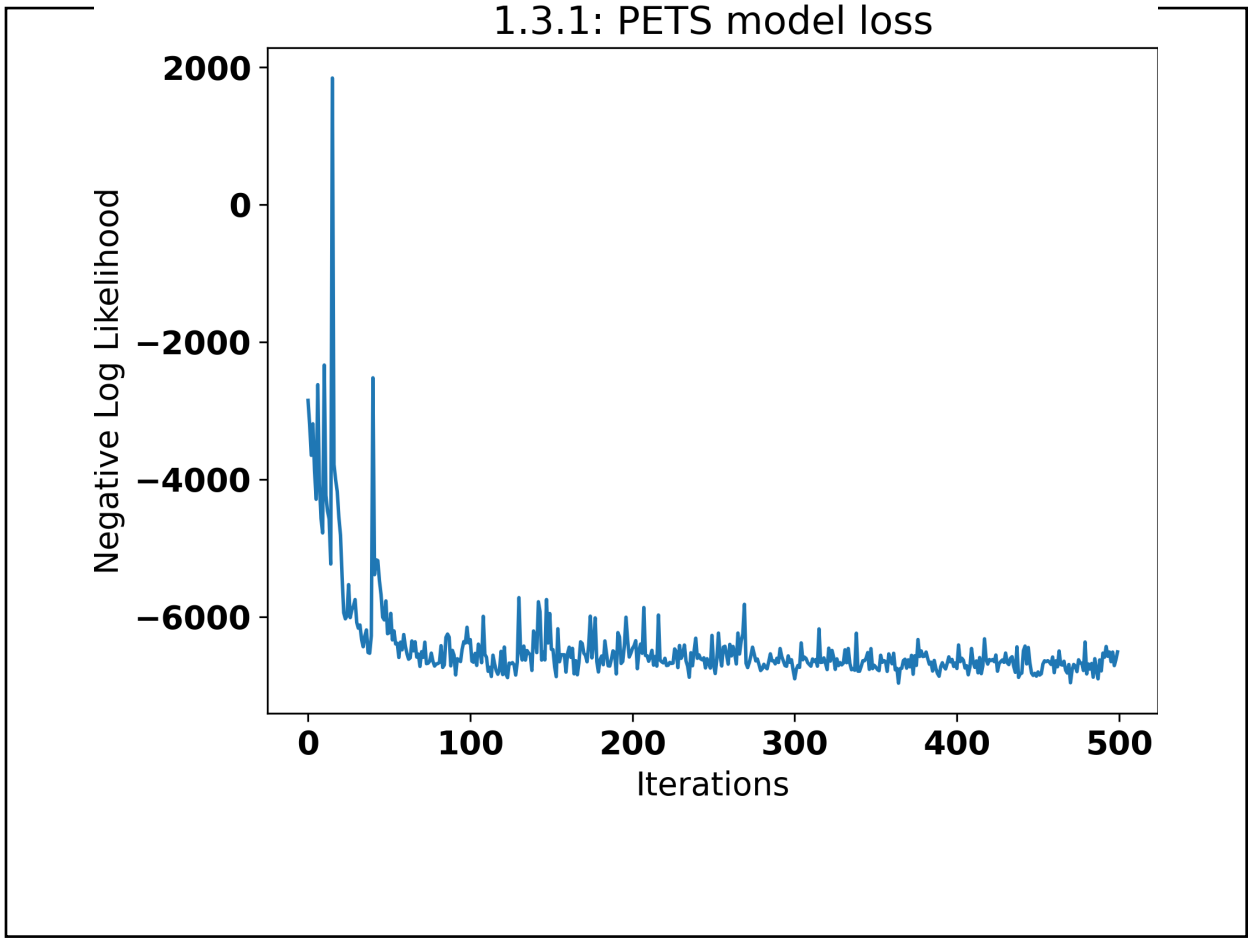
Success percentage 38%

1.2.4 Random sampling vs. CEM (6 pt)

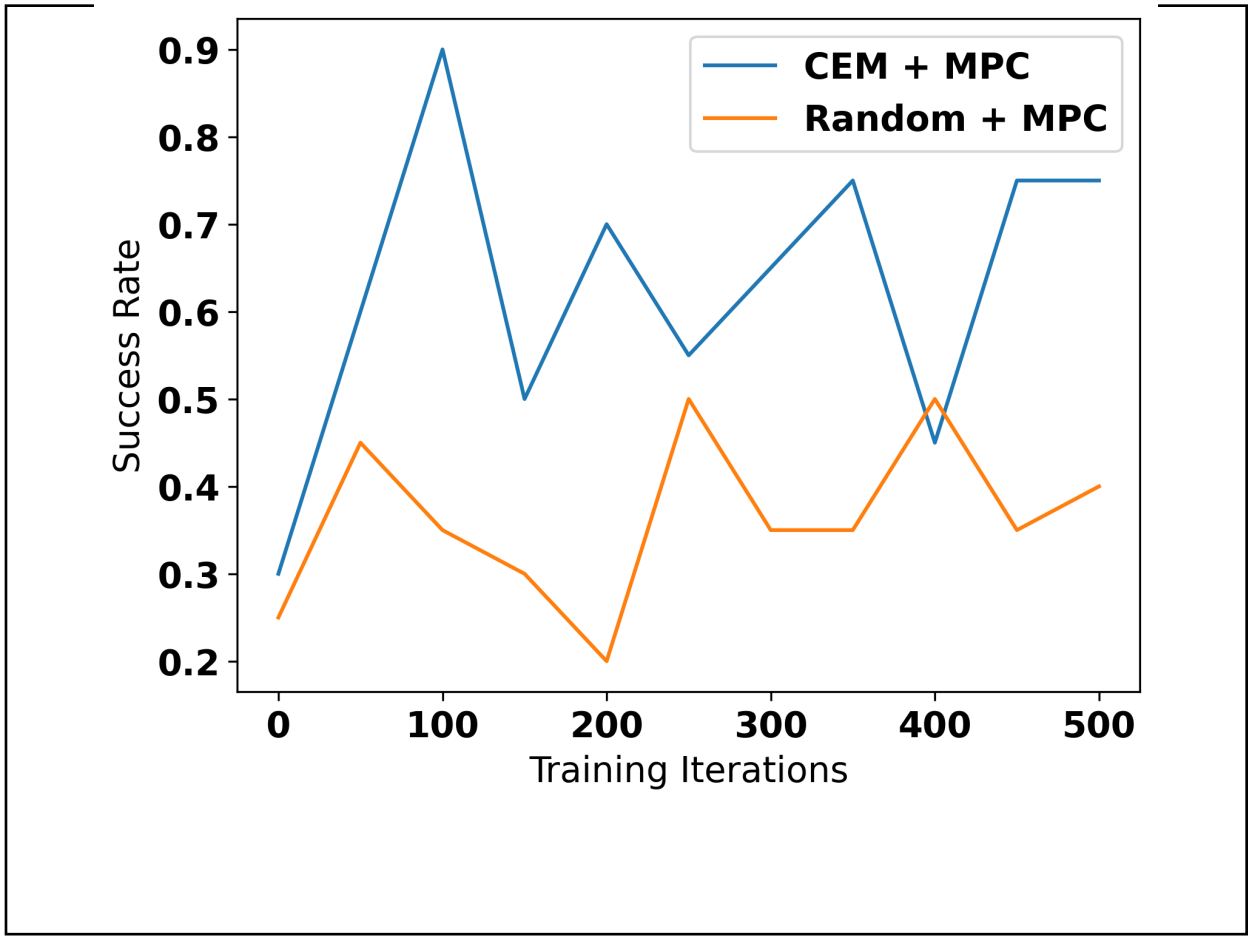
CEM outperforms random sampling with our learned model, achieving a 38% success rate compared to random sampling's 20%. This aligns with our expectations since CEM iteratively refines its action distribution, while random sampling maintains a fixed distribution throughout.
However, both methods perform significantly worse with the learned model compared to using ground truth dynamics (where they achieved 80% and 88% success rates respectively). This suggests our learned model has considerable inaccuracies that affect planning.
The policy tends to succeed in situations requiring simple, direct pushing motions. It struggles with more complex scenarios that require precise control or multiple contact points, likely due to the model's difficulty in accurately predicting complex physical interactions and contact dynamics.

1.3 MBRL with PETS (35 pts)

1.3.1 Training loss plot (10 pt)



1.3.2 Test percentage of successes plot (10 pt)



1.3.3 Limitations of MBRL (5 pt)

MBRL has several key limitations. First, it struggles with complex dynamics that are difficult to model accurately, as shown by our pushing task where contact dynamics caused significant modeling errors. Second, it requires substantial computational resources for online planning, making it challenging for real-time applications with fast dynamics.

However, MBRL would be preferred over policy gradient methods when: (1) we have limited environment interactions available, since MBRL is more sample efficient by learning an explicit model, (2) the environment dynamics are relatively simple and predictable, and (3) we need interpretable policies where we can understand the planning process. Our experiments showed MBRL achieving 75% success rate after just 350 iterations, while policy gradient methods typically require thousands of episodes.