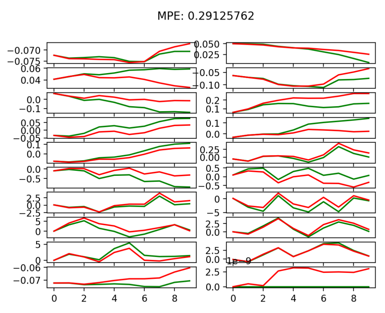
Homework 4

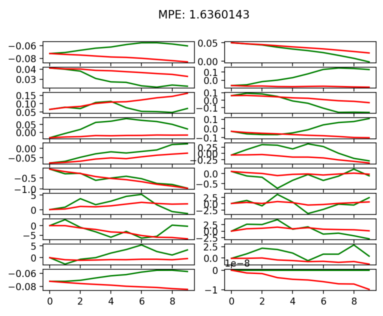
Juanwu Lu (3037432593)

(M.Sc. Civil Engineering, UC Berkeley)

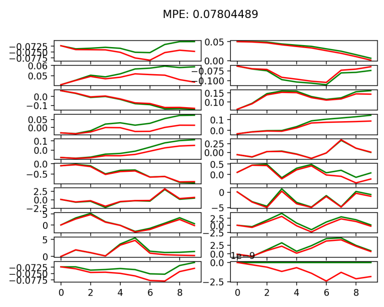
Problem 1



(a)



(b)



(c)

Figure 1. Results from different settings: (a) 1 hidden layer, 32 neurons, 500 steps; (b) 2 hidden layers, 250 neurons, 5 steps; (c) 2 hidden layers, 250 neurons, 500 steps.

Based on the results in Figure 1, training with a neural network consisting of two hidden layers of 250 neurons with 500 agent training steps each iteration achieves the best performance (i.e., with the lowewst MPE). Meanwhile, by comparing figure 1(b) and 1(c), I can tell that increasing the number of agent training steps per iteration has more significant improvement on the final performance than increasing the complexity of the neural network model.

```
echo "Running Homework 4 Problem 1";
python cs285/scripts/run_hw4_mb.py --exp_name q1_cheetah_n500_arch1x32 --env_name
    cheetah-cs285-v0 --add_sl_noise --n_iter 1 --batch_size_initial 20000
    --num_agent_train_steps_per_iter 500 --n_layers 1 --size 32 --scalar_log_freq -1
    --video_log_freq -1 --mpc_action_sampling_strategy random -gpu_id $1 &
python cs285/scripts/run_hw4_mb.py --exp_name q1_cheetah_n5_arch2x250 --env_name
    cheetah-cs285-v0 --add_sl_noise --n_iter 1 --batch_size_initial 20000
    --num_agent_train_steps_per_iter 5 --n_layers 2 --size 250 --scalar_log_freq -1
    --video_log_freq -1 --mpc_action_sampling_strategy random -gpu_id $1 &
python cs285/scripts/run_hw4_mb.py --exp_name q1_cheetah_n500_arch2x250 --env_name
    cheetah-cs285-v0 --add_sl_noise --n_iter 1 --batch_size_initial 20000
    --num_agent_train_steps_per_iter 500 --n_layers 2 --size 250 --scalar_log_freq -1
    --video_log_freq -1 --mpc_action_sampling_strategy random -gpu_id $1 &
wait;
echo "Done";
```

Problem 2

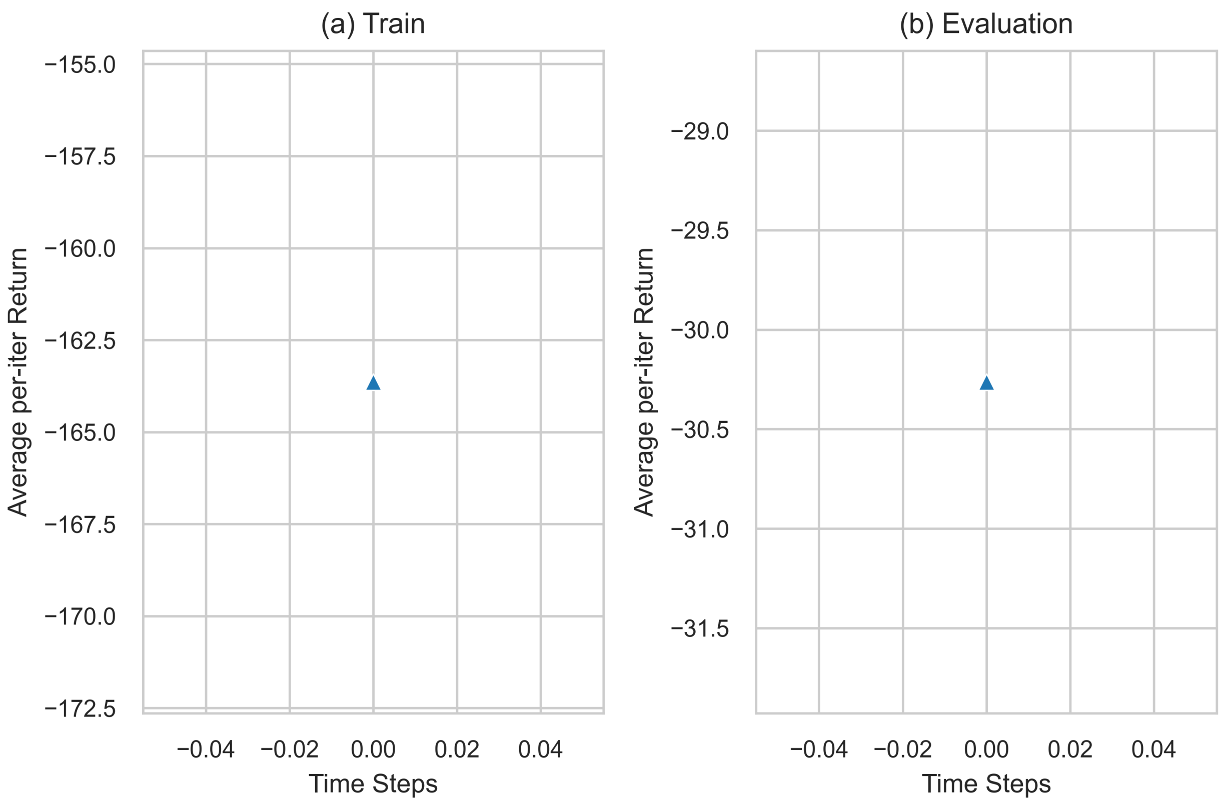


Figure 2. Average returns in (a) training and (b) evaluation procedure. As expected, the training one is -163.6 and the evaluation one is -30.26.

```
echo "Running Homework 4 Problem 2";
python cs285/scripts/run_hw4_mb.py --exp_name q2_obstacles_singleiteration --env_name
    obstacles-cs285-v0 --add_sl_noise --num_agent_train_steps_per_iter 20 --n_iter 1
    --batch_size_initial 5000 --batch_size 1000 --mpc_horizon 10 --video_log_freq -1
    --mpc_action_sampling_strategy random -gpu_id $1;
echo "Done";
```

Problem 3

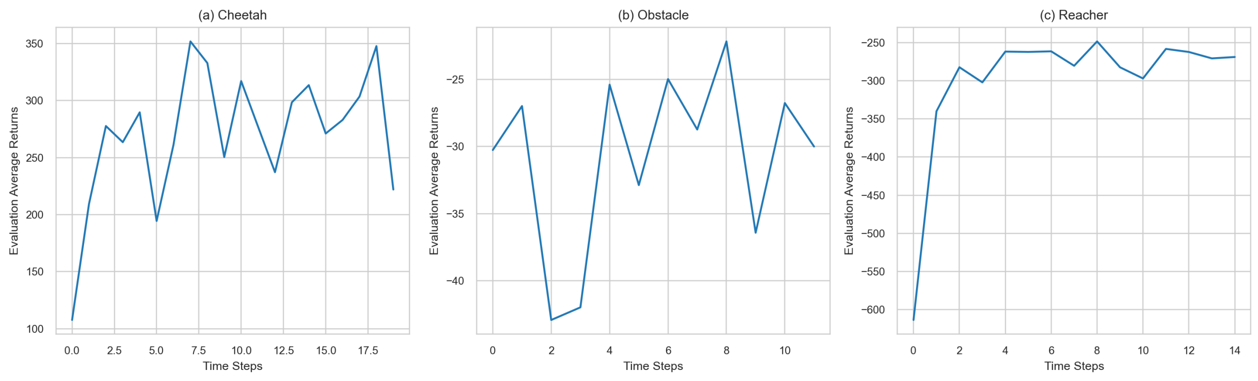


Figure 3. Average evaluation returns in (a) Cheetah, (b) Obstacles, and (c) Reacher environments. Final values are 277.3, -30, and -268.9, respectively.

```
echo "Running HW4 Problem 3";
python cs285/scripts/run_hw4_mb.py --exp_name q3_obstacles --env_name obstacles-cs285-v0
    --add_sl_noise --num_agent_train_steps_per_iter 20 --batch_size_initial 5000 --batch_size
    1000 --mpc_horizon 10 --n_iter 12 --video_log_freq -1 --mpc_action_sampling_strategy random
    -gpu_id $1 &
python cs285/scripts/run_hw4_mb.py --exp_name q3_reacher --env_name reacher-cs285-v0
    --add_sl_noise --mpc_horizon 10 --num_agent_train_steps_per_iter 1000 --batch_size_initial
    5000 --batch_size 5000 --n_iter 15 --video_log_freq -1 --mpc_action_sampling_strategy random
    -gpu_id $1 &
python cs285/scripts/run_hw4_mb.py --exp_name q3_cheetah --env_name cheetah-cs285-v0
    --mpc_horizon 15 --add_sl_noise --num_agent_train_steps_per_iter 1500 --batch_size_initial
    5000 --batch_size 5000 --n_iter 20 --video_log_freq -1 --mpc_action_sampling_strategy random
    -gpu_id $1 &
wait
echo "Done";
```

Problem 4

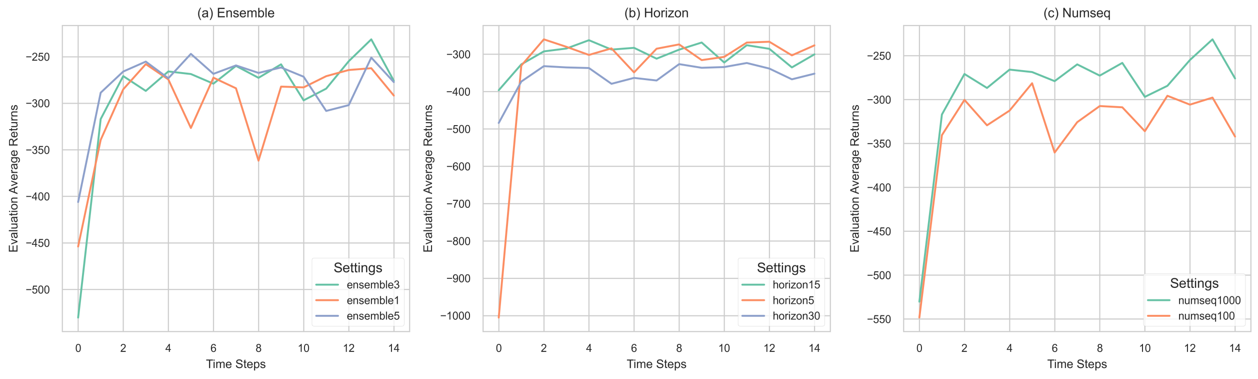


Figure 4. Average evaluation returns with different settings of (a) number of ensemble models; (b) model predictive control horizons; and (c) number of generated action sequence.

Based on the results in Figure 4, increasing the number of ensemble models (as shown in 4(a)) helps improve the performance but not much. Increasing prediction horizon would cause leads to higher estimation variance and prediction uncertainty, hence deteriorate the performance (as shown in 4(b)). Increasing number of generated action sequence helps explore more potential actions, which increases the possibility of achieving better score (as shown in 4(c)).

Codes are left out on purpose due to limited space.

Problem 5

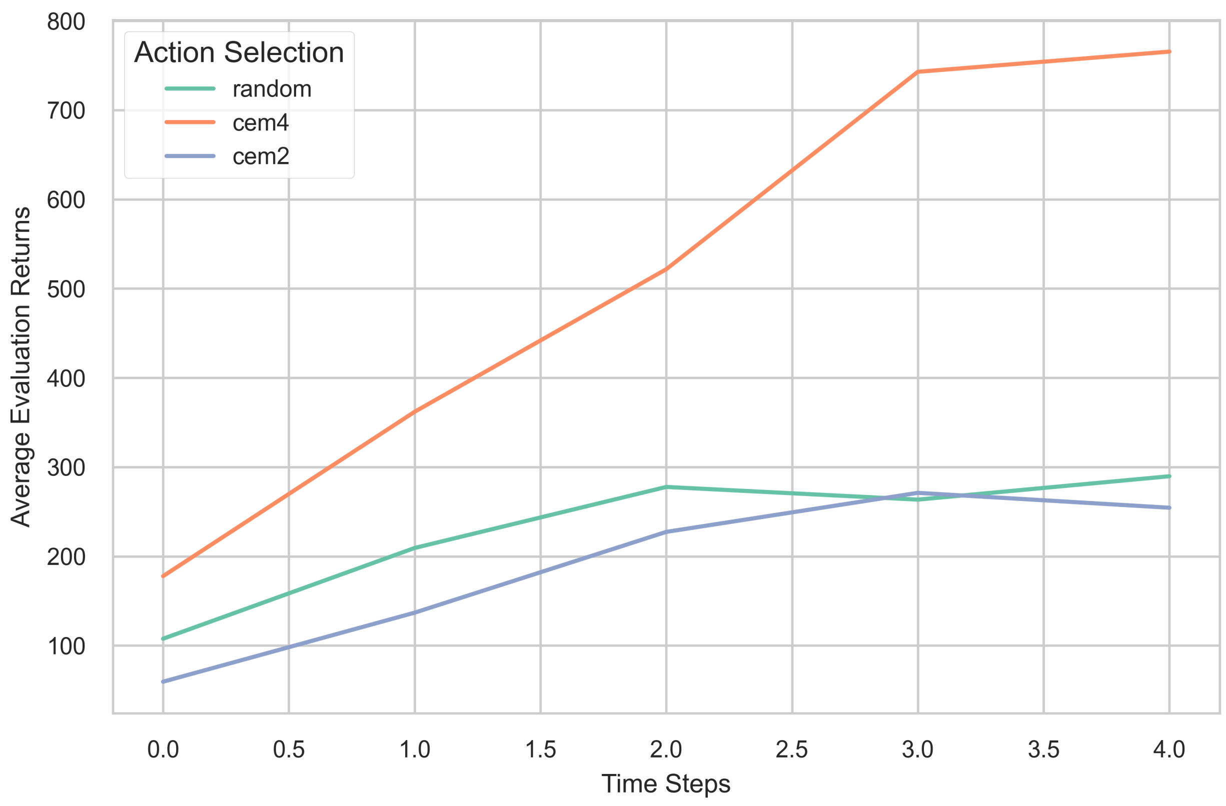


Figure 5. Average evalutaion returns with different action selection settings. Model with 4 CEM action sampling iterations achieves the best performance at around 765.5. However, implementing 2 CEM iterations achieves lower returns than random one in my experiement. My explannation for it is that CEM performance could be affected by the initial action sequence guess. And in my case, taking only 2 CEM iterations is not sufficient to guarentee that sampled actions are the optimal.

```
echo "Running Homework 4 Problem 5"
python cs285/scripts/run_hw4_mb.py --exp_name q5_cheetah_random --env_name cheetah-cs285-v0
    --mpc_horizon 15 --add_sl_noise --num_agent_train_steps_per_iter 1500 --batch_size_initial
    5000 --batch_size 5000 --n_iter 5 --video_log_freq -1 --mpc_action_sampling_strategy random
    -gpu_id $1 &
python cs285/scripts/run_hw4_mb.py --exp_name q5_cheetah_cem_2 --env_name cheetah-cs285-v0
    --mpc_horizon 15 --add_sl_noise --num_agent_train_steps_per_iter 1500 --batch_size_initial
    5000 --batch_size 5000 --n_iter 5 --video_log_freq -1 --mpc_action_sampling_strategy cem
    --cem_iterations 2 -gpu_id $1 &
python cs285/scripts/run_hw4_mb.py --exp_name q5_cheetah_cem_4 --env_name cheetah-cs285-v0
    --mpc_horizon 15 --add_sl_noise --num_agent_train_steps_per_iter 1500 --batch_size_initial
    5000 --batch_size 5000 --n_iter 5 --video_log_freq -1 --mpc_action_sampling_strategy cem
    --cem_iterations 4 -gpu_id $1 &
wait
echo "Done!"
```

Problem 6

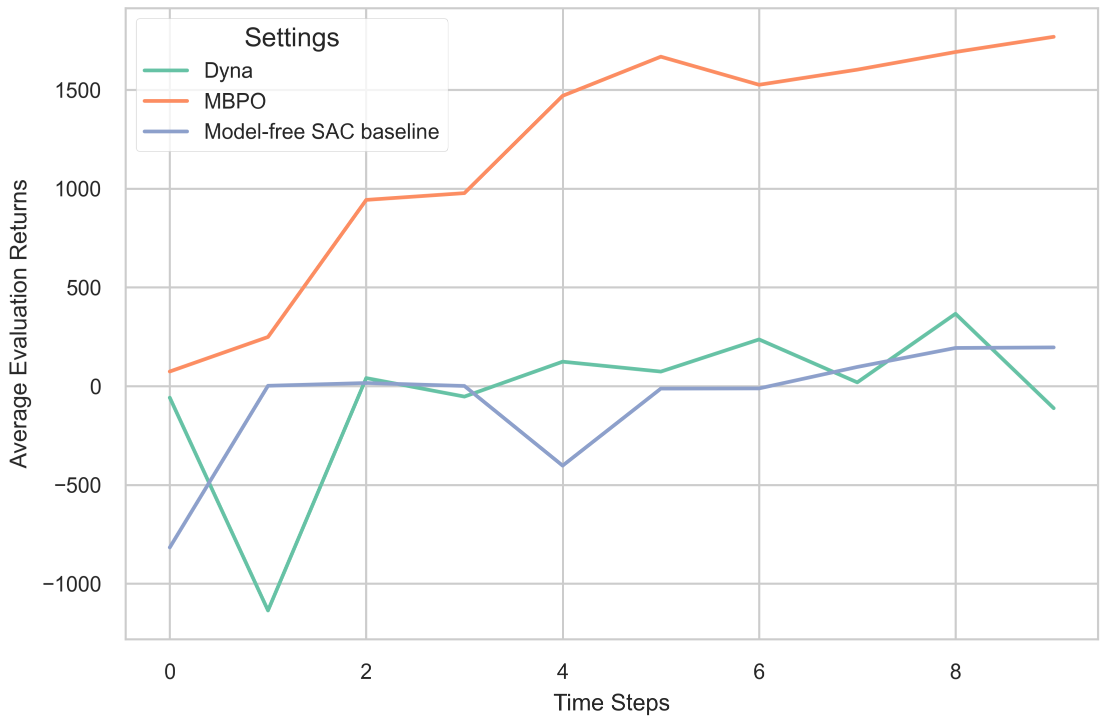


Figure 6. Results of with three different settings. As expected the Model-free SAC perform the worst among three, followed by Dyna style single-step rollouts. This means adding only single-step rollouts from the model has limited improvement. On the other hand, MBPO with 10-step rollouts significantly improves the long-term performance, proving the dynamic model has successfully learned the observed dynamic.

```
echo "Running Homework 6"
python cs285/scripts/run_hw4_mbpo.py --exp_name q6_cheetah_rlen10 --env_name cheetah-cs285-v0
    --add_sl_noise --num_agent_train_steps_per_iter 1500 --batch_size_initial 5000 --batch_size
    5000 --n_iter 10 --video_log_freq -1 --sac_discount 0.99 --sac_n_layers 2 --sac_size 256
    --sac_batch_size 1500 --sac_learning_rate 0.0003 --sac_init_temperature 0.1 --sac_n_iter
    1000 --mbpo_rollout_length 0 -gpu_id $1 &
python cs285/scripts/run_hw4_mbpo.py --exp_name q6_cheetah_rlen1 --env_name cheetah-cs285-v0
    --add_sl_noise --num_agent_train_steps_per_iter 1500 --batch_size_initial 5000 --batch_size
    5000 --n_iter 10 --video_log_freq -1 --sac_discount 0.99 --sac_n_layers 2 --sac_size 256
    --sac_batch_size 1500 --sac_learning_rate 0.0003 --sac_init_temperature 0.1 --sac_n_iter
    5000 --mbpo_rollout_length 1 -gpu_id $1 &
python cs285/scripts/run_hw4_mbpo.py --exp_name q6_cheetah_rlen10 --env_name cheetah-cs285-v0
    --add_sl_noise --num_agent_train_steps_per_iter 1500 --batch_size_initial 5000 --batch_size
    5000 --n_iter 10 --video_log_freq -1 --sac_discount 0.99 --sac_n_layers 2 --sac_size 256
    --sac_batch_size 1500 --sac_learning_rate 0.0003 --sac_init_temperature 0.1 --sac_n_iter
    5000 --mbpo_rollout_length 10 -gpu_id $1 &
wait
echo "Done!"
```