CS 285 HW4

Cuiqianhe Du

Question 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

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

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

first run: q1\_cheetah\_n500\_arch1x32

A picture containing chart

Description automatically generated

Chart, line chart

Description automatically generated

Second run: q1\_cheetah\_n5\_arch2x250

Chart, line chart

Description automatically generated

Chart, diagram, line chart

Description automatically generated

Third run: q1\_cheetah\_n500\_arch2x250

Chart

Description automatically generated with medium confidence

Chart, line chart

Description automatically generated

From previous three experiments, we can tell that:

* The bigger the layer and the size, the better the performance (from experiment 1 and 3)
* The more the training step per iteration, the better the performance (from experiment 2 and 3)

Question 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 --mpc\_action\_sampling\_strategy random

Eval\_AverageReturn : -45.98809814453125

Train\_AverageReturn : -167.09857177734375

Question 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 --mpc\_action\_sampling\_strategy random

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 --mpc\_action\_sampling\_strategy random

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 --mpc\_action\_sampling\_strategy random

Chart, line chart

Description automatically generated

Chart, line chart

Description automatically generated

Chart, line chart

Description automatically generated

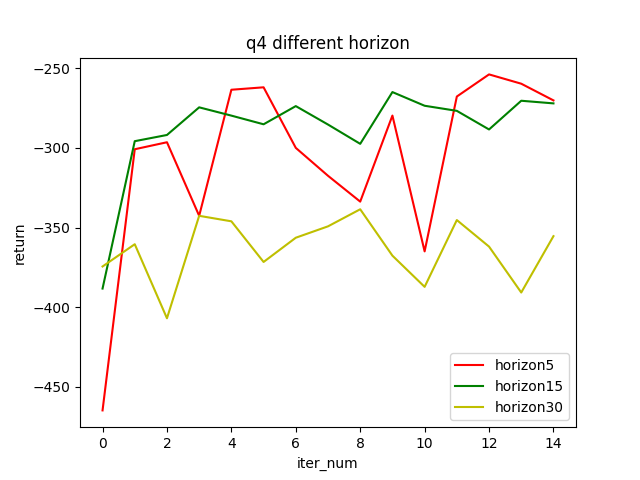
Question 4

python cs285/scripts/run\_hw4\_mb.py --exp\_name q4\_reacher\_horizon5 --env\_name reacher-cs285-v0 --add\_sl\_noise --mpc\_horizon 5 --num\_agent\_train\_steps\_per\_iter 1000 --batch\_size 800 --n\_iter 15 --mpc\_action\_sampling\_strategy random

python cs285/scripts/run\_hw4\_mb.py --exp\_name q4\_reacher\_horizon15 --env\_name reacher-cs285-v0 --add\_sl\_noise --mpc\_horizon 15 --num\_agent\_train\_steps\_per\_iter 1000 --batch\_size 800 --n\_iter 15 --mpc\_action\_sampling\_strategy random

python cs285/scripts/run\_hw4\_mb.py --exp\_name q4\_reacher\_horizon30 --env\_name reacher-cs285-v0 --add\_sl\_noise --mpc\_horizon 30 --num\_agent\_train\_steps\_per\_iter 1000 --batch\_size 800 --n\_iter 15 --mpc\_action\_sampling\_strategy random

performance is the worst when planning horizon is 30, and even though 15 and 5 have similar output, performance is more stable when horizon is 15.



python cs285/scripts/run\_hw4\_mb.py --exp\_name q4\_reacher\_numseq100 --env\_name reacher-cs285-v0 --add\_sl\_noise --mpc\_horizon 10 --num\_agent\_train\_steps\_per\_iter 1000 --batch\_size 800 --n\_iter 15 --mpc\_num\_action\_sequences 100 --mpc\_action\_sampling\_strategy random

python cs285/scripts/run\_hw4\_mb.py --exp\_name q4\_reacher\_numseq1000 --env\_name reacher-cs285-v0 --add\_sl\_noise --mpc\_horizon 10 --num\_agent\_train\_steps\_per\_iter 1000 --batch\_size 800 --n\_iter 15 --mpc\_num\_action\_sequences 1000 --mpc\_action\_sampling\_strategy random

Larger number of candidate action sequences have better performance

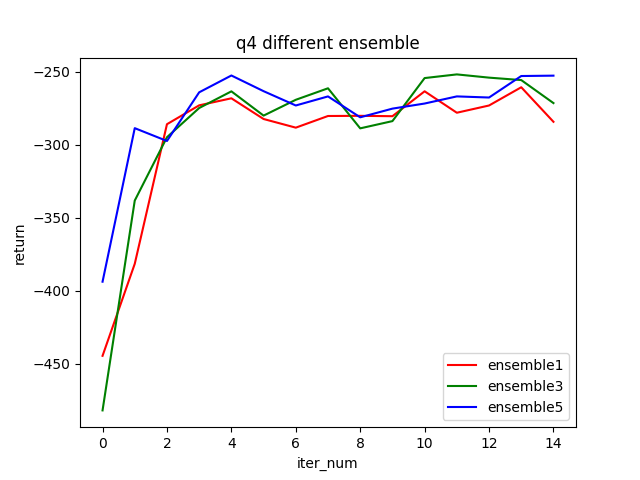
Chart, line chart

Description automatically generated

python cs285/scripts/run\_hw4\_mb.py --exp\_name q4\_reacher\_ensemble1 --env\_name reacher-cs285-v0 --ensemble\_size 1 --add\_sl\_noise --mpc\_horizon 10 --num\_agent\_train\_steps\_per\_iter 1000 --batch\_size 800 --n\_iter 15 --mpc\_action\_sampling\_strategy random

python cs285/scripts/run\_hw4\_mb.py --exp\_name q4\_reacher\_ensemble3 --env\_name reacher-cs285-v0 --ensemble\_size 3 --add\_sl\_noise --mpc\_horizon 10 --num\_agent\_train\_steps\_per\_iter 1000 --batch\_size 800 --n\_iter 15 --mpc\_action\_sampling\_strategy random

python cs285/scripts/run\_hw4\_mb.py --exp\_name q4\_reacher\_ensemble5 --env\_name reacher-cs285-v0 --ensemble\_size 5 --add\_sl\_noise --mpc\_horizon 10 --num\_agent\_train\_steps\_per\_iter 1000 --batch\_size 800 --n\_iter 15 --mpc\_action\_sampling\_strategy random

Larger ensemble size has better and more stable performance

Question 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 --mpc\_action\_sampling\_strategy random

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 --mpc\_action\_sampling\_strategy cem --cem\_iterations 2

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 --mpc\_action\_sampling\_strategy cem --cem\_iterations 4

CEM method with 4 iterations have best performance compared to CEM with 2 iterations and random shooting methodChart, line chart

Description automatically generated