seiko_results

1. schedule and result for MSS-SEIKO

1.1. schedule

total number = 19200, each outer loop, num_samples is dividable by 16.

- 1 outer-loop, num_samples=[19200]
- 2 outer-loop, num_samples=[6400, 12800]
- 3 outer-loop, num_samples=[2720, 5440, 11040]
- 4 outer-loop, num_samples=[1280, 2560, 5120, 10240]
- 5 outer-loop, num_samples=[560,1120,2240,4480,10800]
- 6 outer-loop, num_samples=[320, 640, 1200, 2400, 4800, 9840]
- 7 outer-loop, num_samples=[160, 320, 640, 1200, 2400, 4800, 9680]
- 8 outer-loop, num_samples=[160, 320, 640, 1200, 1760, 2720, 4160, 8240]
- (currently not planned to conduct) 9 outer-loop, num_samples=[64, 128, 256, 512, 1024, 1600, 3200, 4800, 7616]
- (currently not planned to conduct) 10 outer-loop, num_samples=[32, 64, 128, 256, 512, 1024, 1600, 3200, 4800, 7584]

4 seeds are chosen (31, 32, 41, 42). seed tuning is not really effective for experiment result. the fine-tuning trend is visualized. using 4 GPU, 4 set of experiment can be done with 2.5 days or so.

experiment schedule (with experiment result listed in sequence of avg_final_eval_reward, std, avg_time_spent[in hours]):

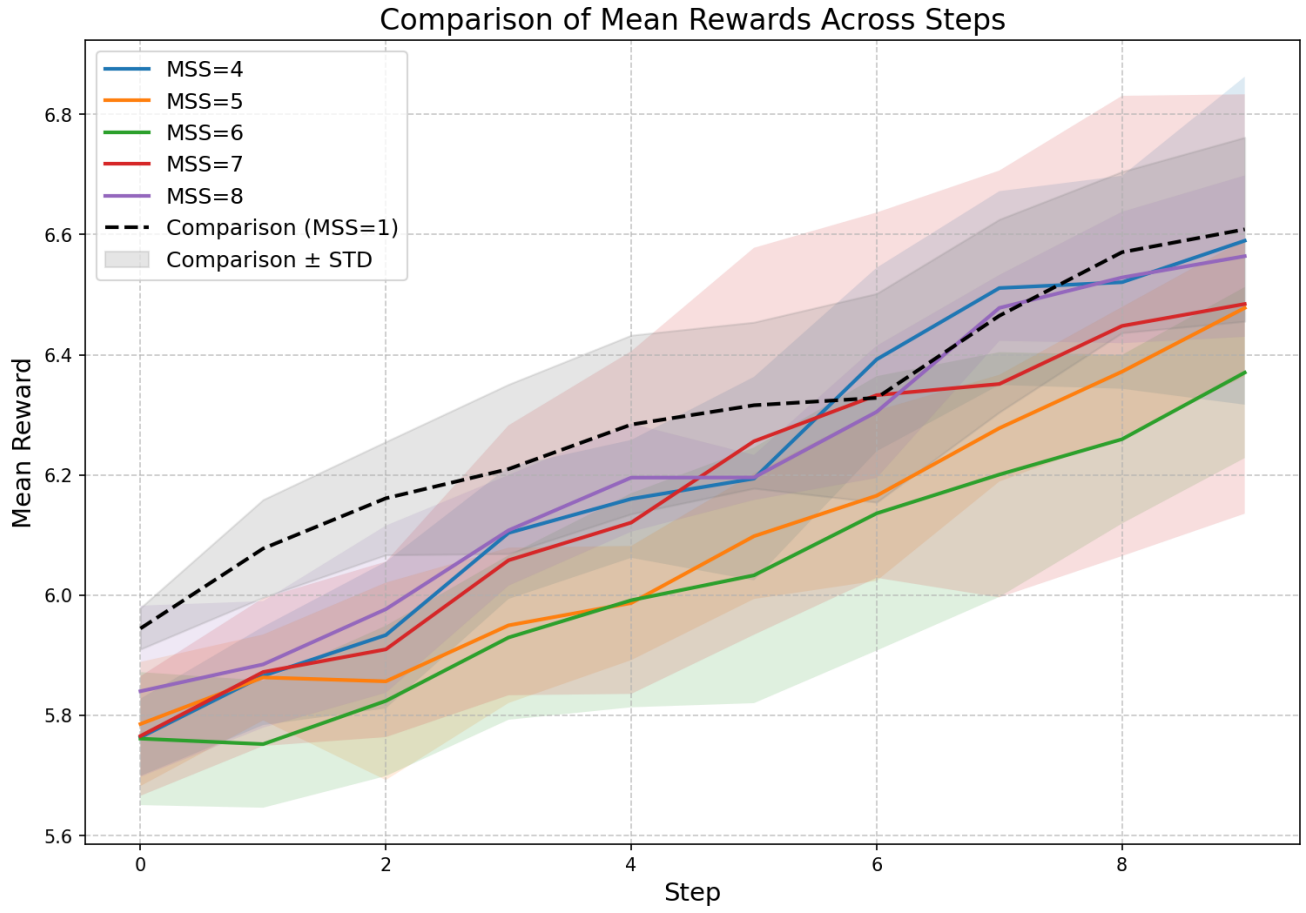
num-outer-loop / num-mss	1(yue)	4	5	6	7	8
1	✅ 6.61,0.153	6.59, 0.2728, 7.967	6.478, 0.1139, 7.054	6.37, 0.1422, 6.418	6.484, 0.3489, 5.983	6.564, 0.1341, 5.632
2	✅ 6.472, 0.2604	✅ 6.707, 0.3007, 10.76	✅ 6.464, 0.6474, 9.62	✅ 6.881, 0.2567, 8.965	✅ 6.743, 0.1894, 8.517	✅ 6.966, 0.1665, 8.086

num-outer-loop / num-mss	1(yue)	4	5	6	7	8
3	✅ 7.151, 0.376, 29.8	✅ 7.054, 0.3638, 13.06	✅ 7.734, 0.6693, 11.87	✅ 7.439, 0.558, 11.21	✅ 7.343, 0.09005, 10.67	✅ 7.568, 0.8966, 10.4
4	✅ 7.513, 0.3966, 31.492	✅ 7.908, 0.5617, 15.29	✅ 7.839, 0.3943, 14.22	✅ 7.439, 0.7147, 13.37	✅ 7.842, 0.2195, 12.77	✅ 7.604, 0.293, 12.43
5	✅ 7.884, 0.3756, 33.3	✅ 7.963, 0.4212, 16.25	✅ 7.963, 0.4212, 16.25	✅ 7.658, 0.3913, 15.52	✅ 8.427, 0.368, 15.13	✅ 7.872, 0.5894, 14.59
6	✅ 8.177, 0.4994, 35.9	✅ 8.496, 0.5134, 19.46	✅ 8.227, 0.3063, 18.46	✅ 8.256, 0.1492, 17.65	✅ 8.214, 0.3876, 16.98	✅ 7.98, 0.2848, 16.83
7	✅ 7.821, 0.296	✅ 7.718, 0.6321, 16.76	on-going 8.329, 0.7042, 15.36	✅ 8.032, 0.1081, 14.38	✅ 8.003, 0.2528, 14.07	✅ 8.012, 0.1118, 13.51
8	✅ 8.014, 0.3279	✅ 8.258, 0.5172, 17.9	✅ 8.742, 0.1909, 16.52	✅ 8.357, 0.1319, 15.82	✅ 8.475, 0.3425, 15.6	✅ 8.586, 0.7267, 15.13

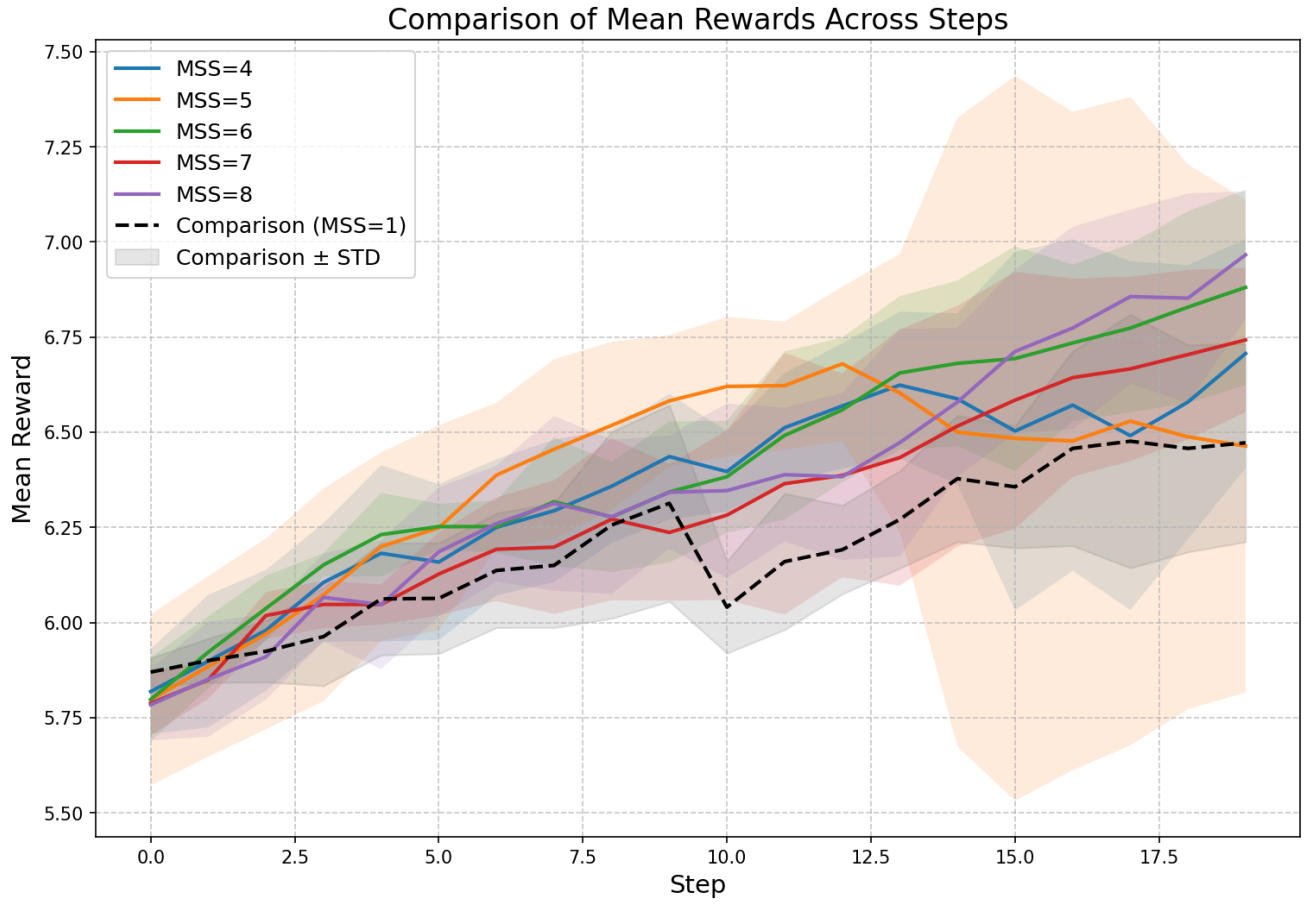
$\lambda = 7$ is found to be the best parametrization, and would be adopted for further experiment

1.2. current results

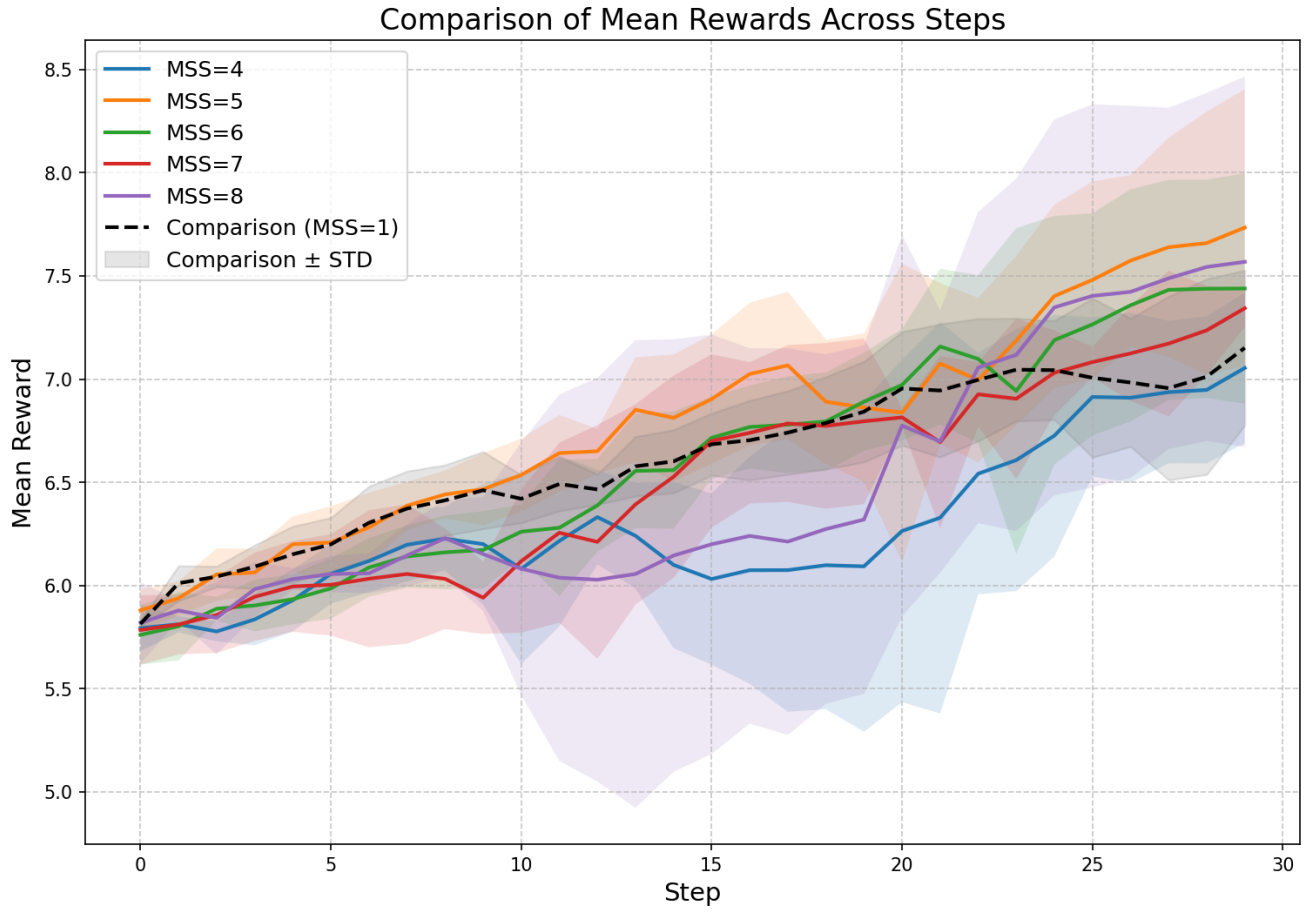
1 outer loop, num_samples = [19200]



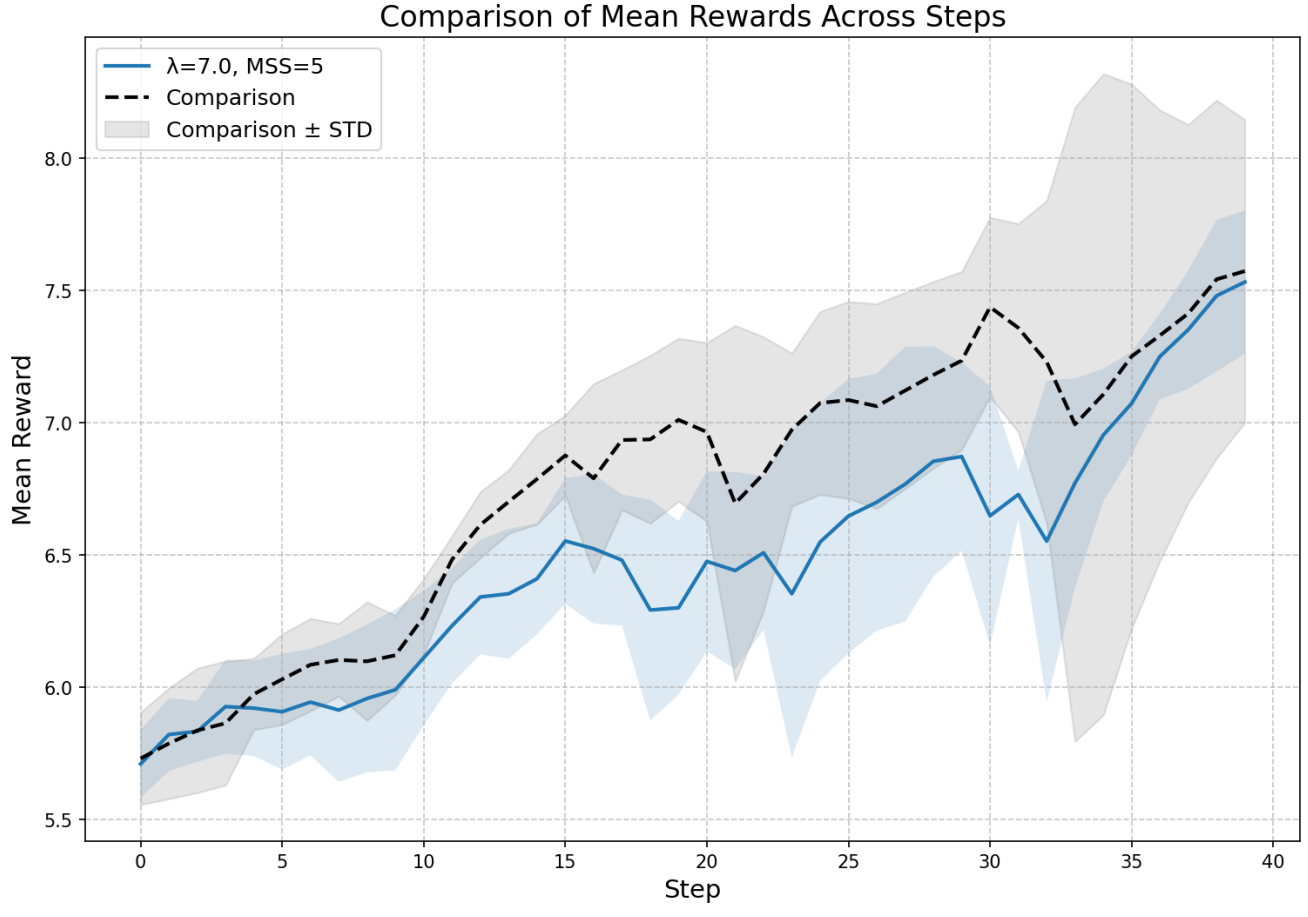
2 outer loop, num_samples = [6400, 12800]



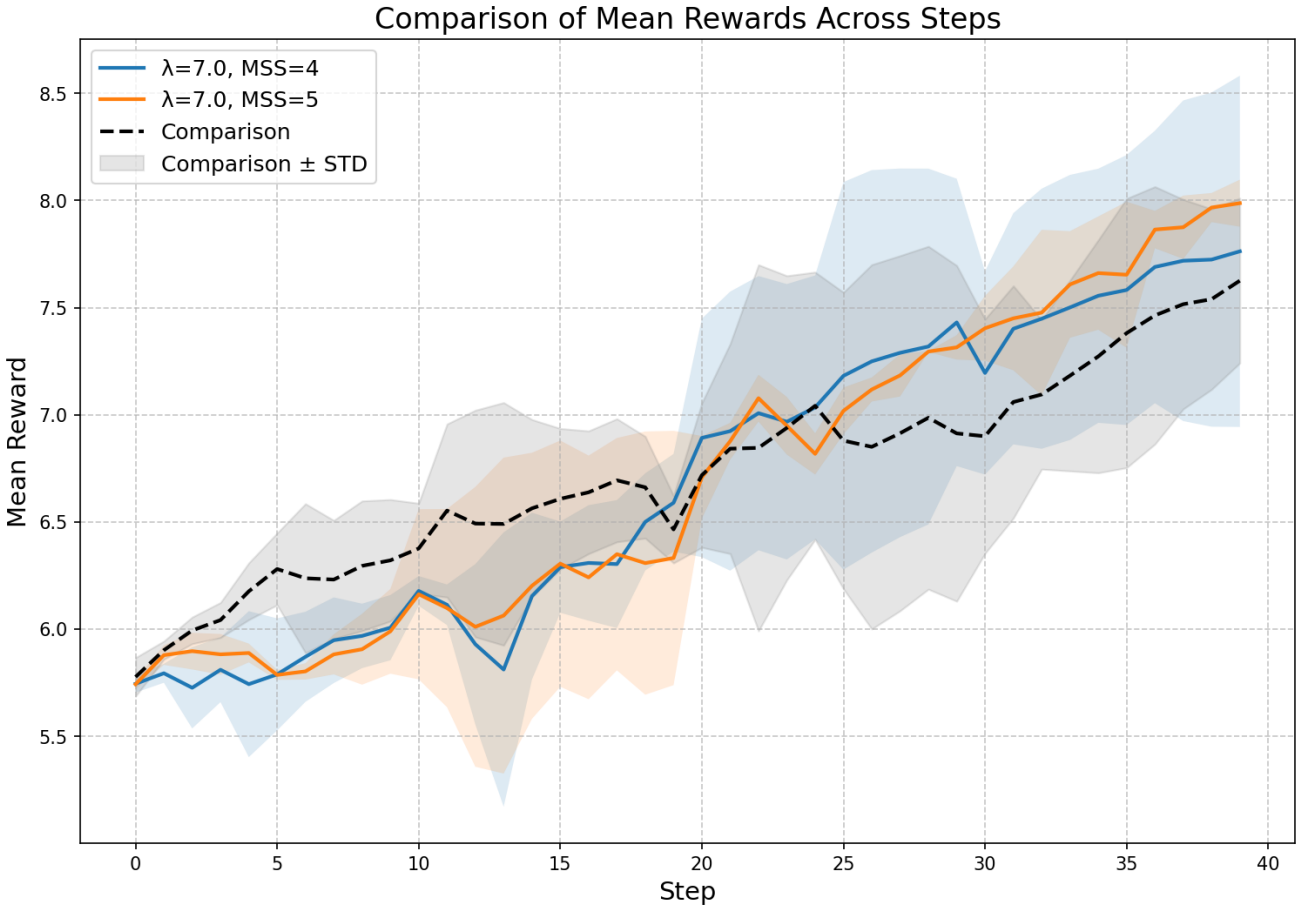
3 outer loops, num_samples=[2720, 5440, 11040]



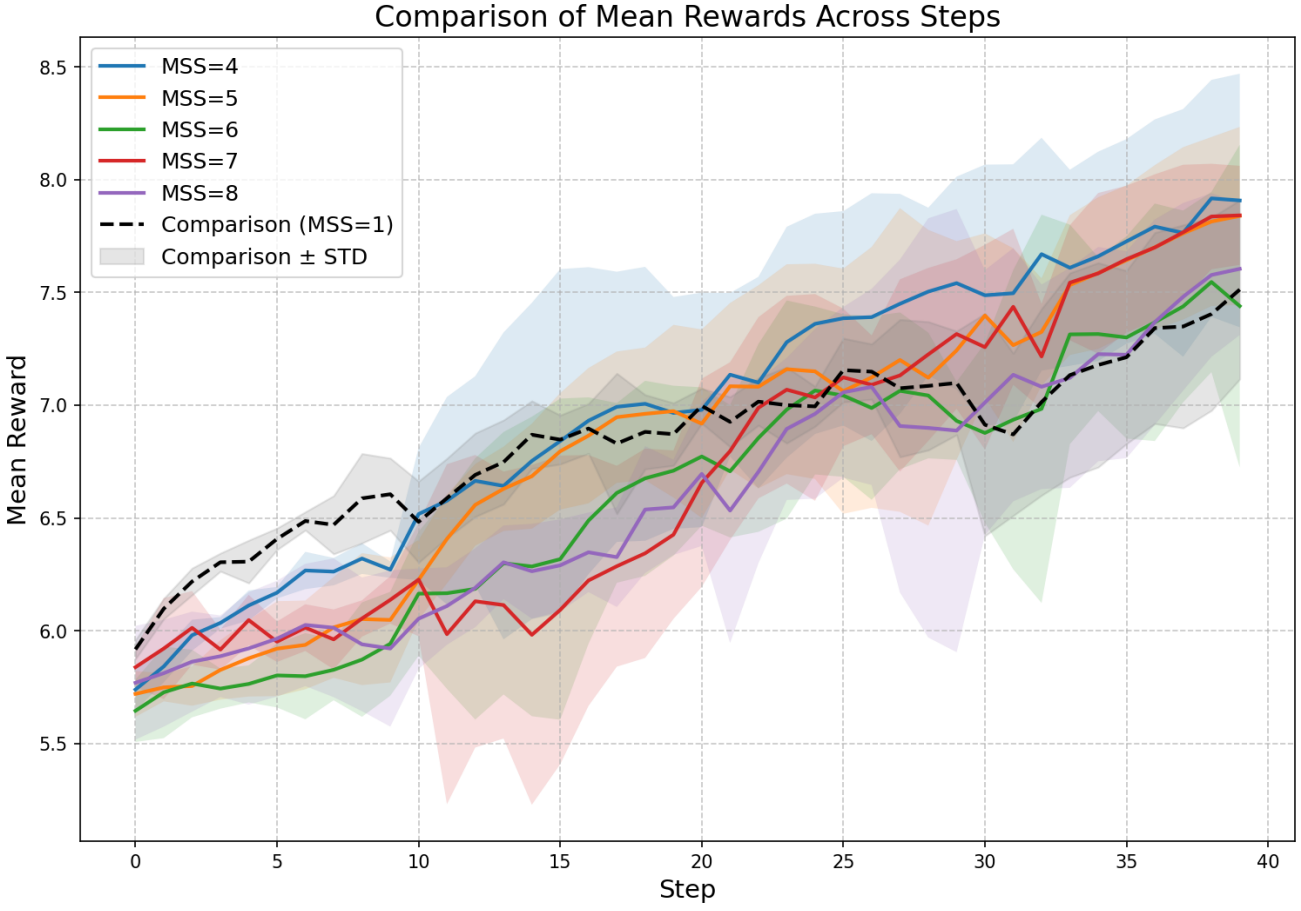
4 outer loops, num_samples=[320,640,1280,2560] vs /5, 14.5 h vs 10.5 h, 4 outer loop



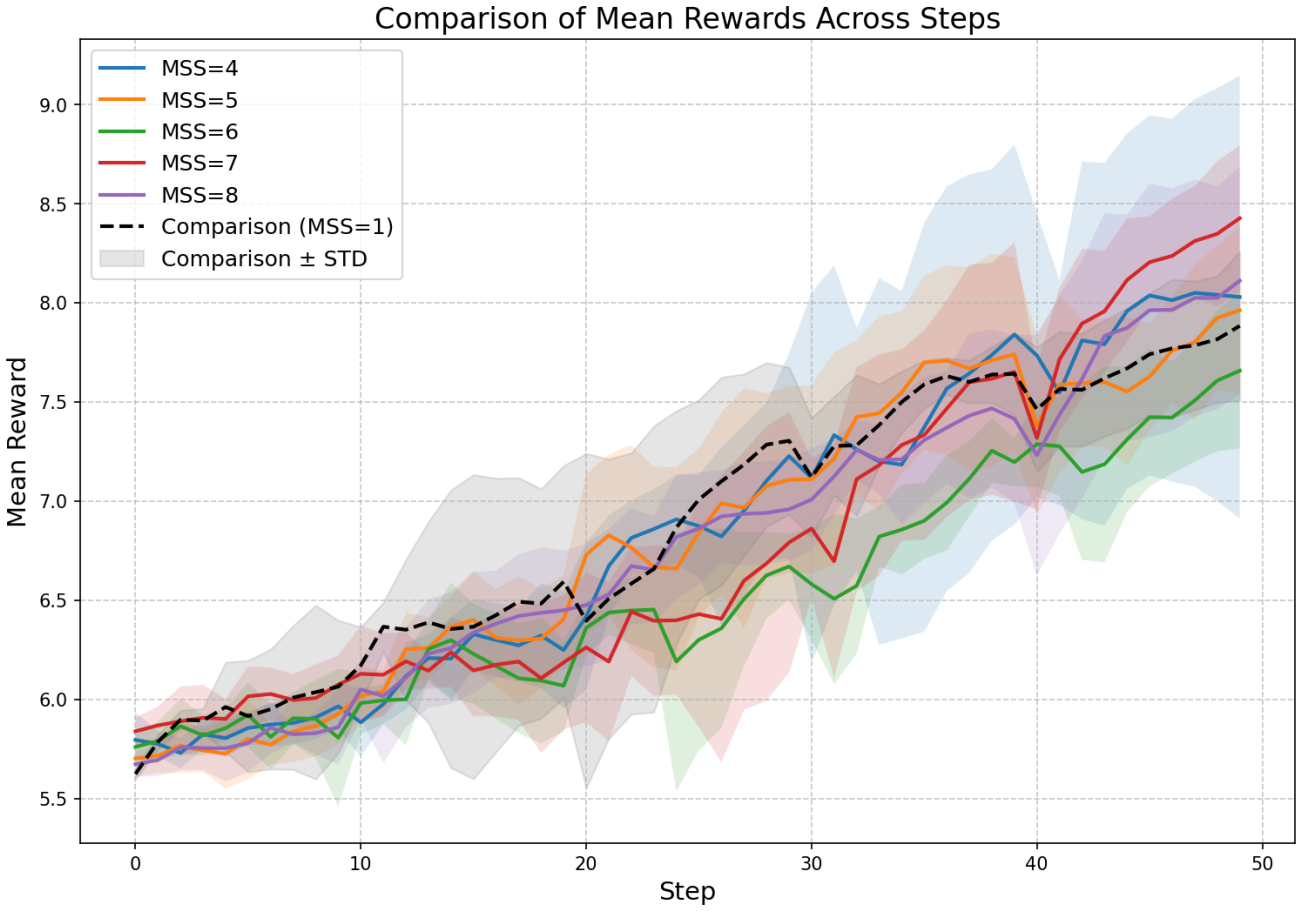
4 outer loops, num_samples=[640,1280,2560,5120] vs /4 & /5, 20h vs 12h vs 11.5 h, 4 outer loop



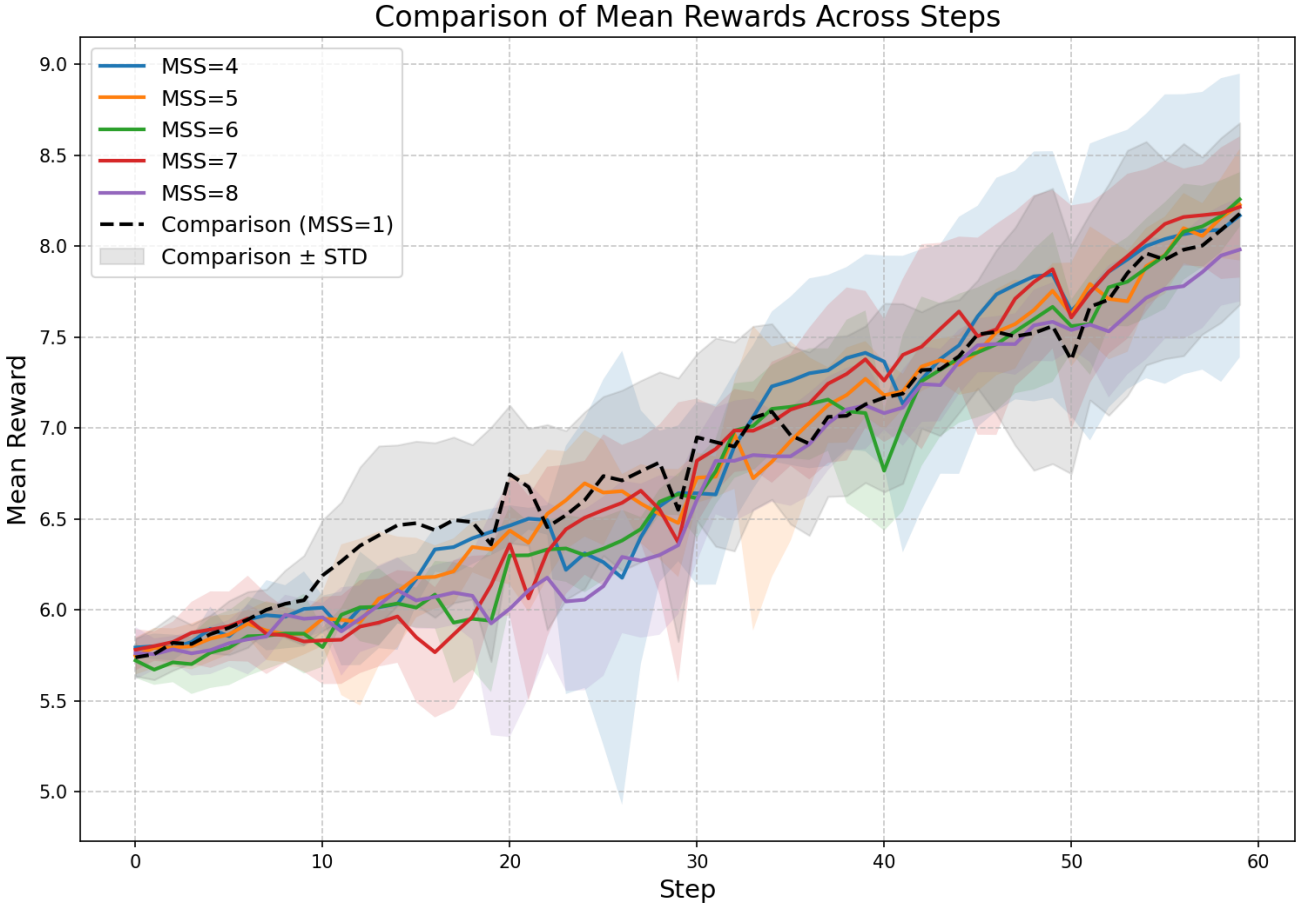
4 outer loops, num_samples = [1280, 2560, 5120, 10240] vs /5, 31.5h vs 14.5 h, 4 outer loops



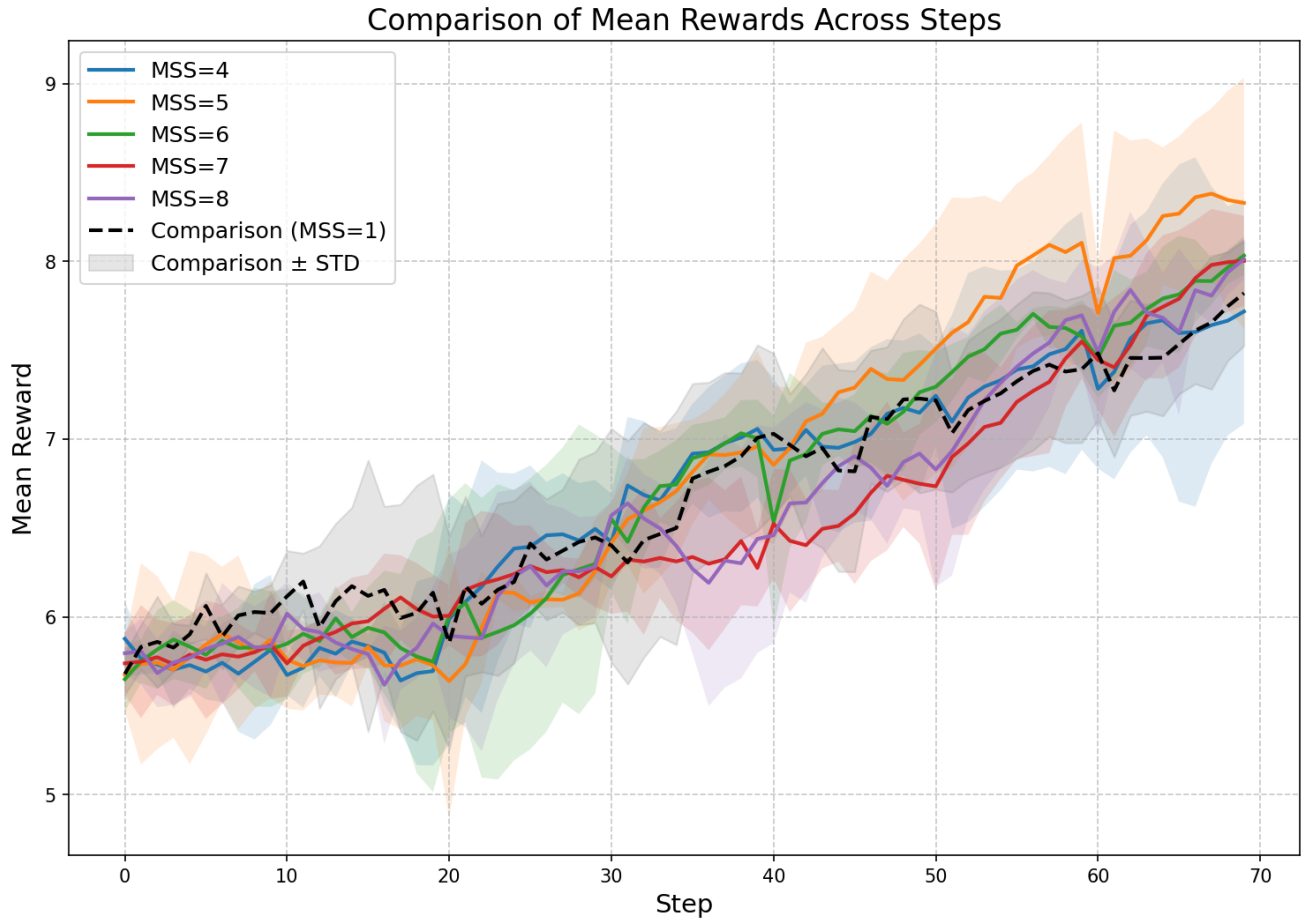
5 outer-loop, num_samples=[620, 1240, 2480, 4960, 9900] v.s. [560,1120,2240,4480,10800] / 5, 6, 7. time used: 33.3h vs 16h, 15.5h, 15h



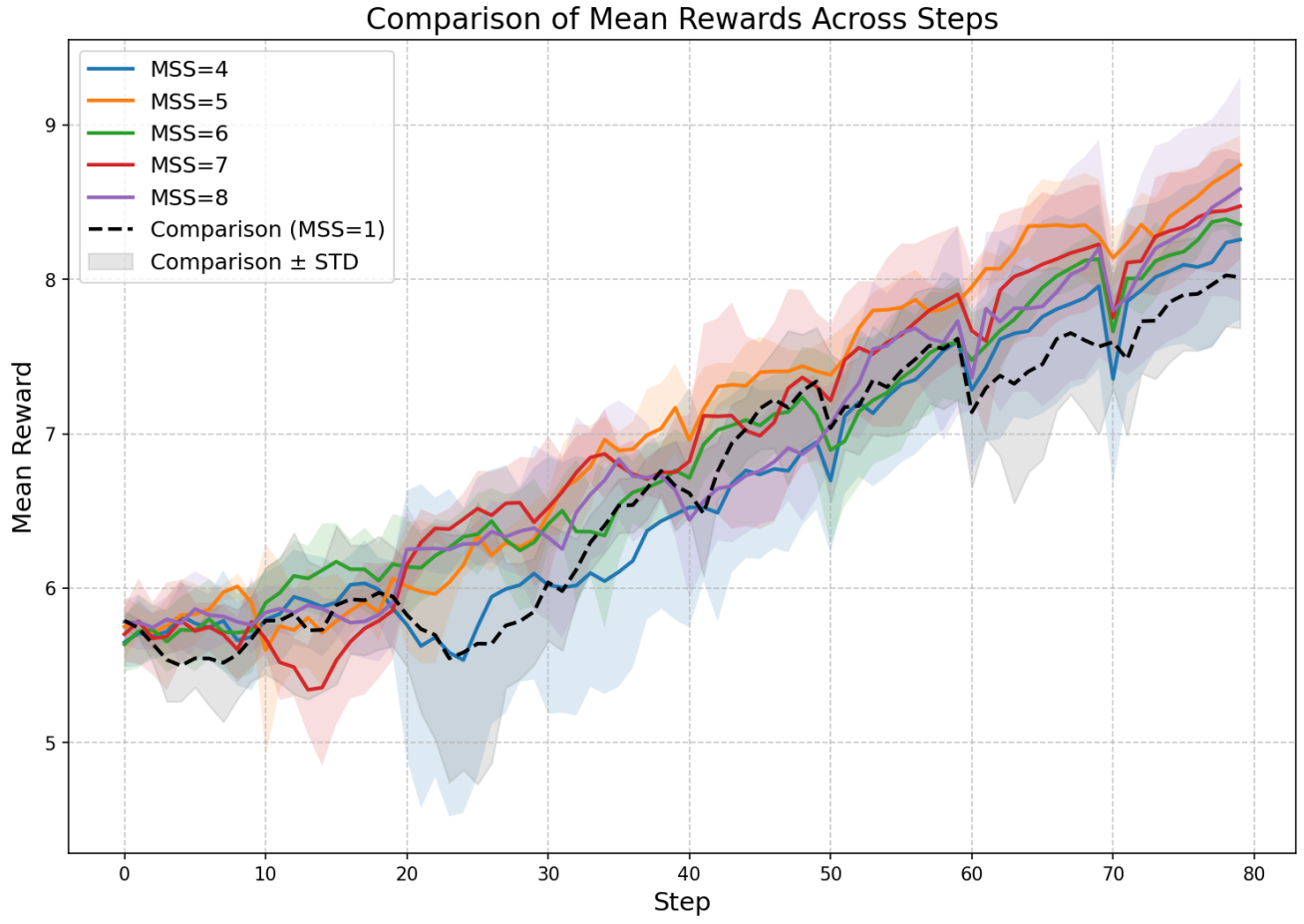
6 outer loop, vs num_sample = [320, 640, 1200, 2400, 4800, 9840]/5,6,7. time used: 35.9 vs 18.5, 17.5, 17



7 outer loop



8 outer loop



selected generated figures are attached here