Meeting 2021/08/18

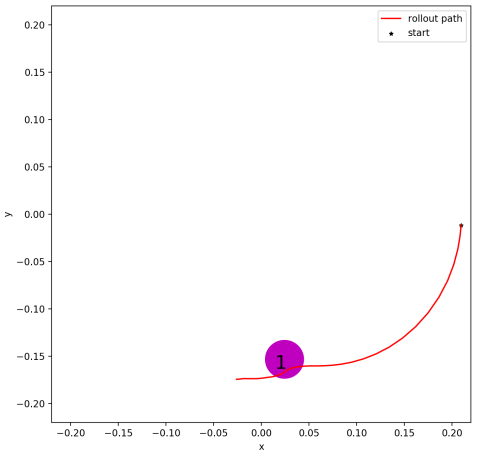
Shuo

Last Time Experiments

1.Dynamics Model 1(DM1): Trained with 1k data without bias

2. Dynamics Model 2(DM2): Trained with 100 data without bias

3. Dynamics Model 3(DM3): Trained with 50 data with bias



Bias model

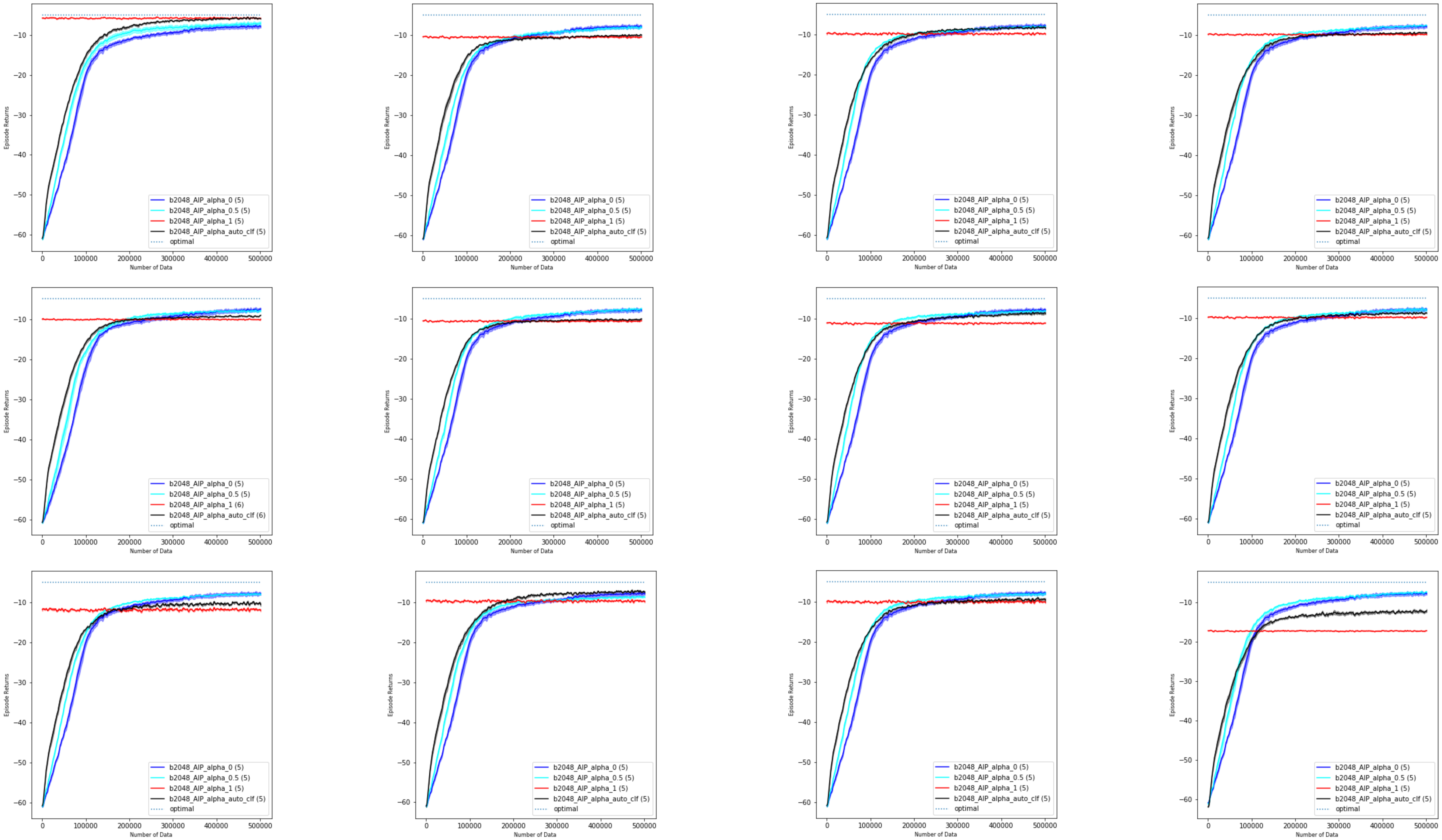
Reference deterministic model-based policy: 4 degrees of training using dynamics

1. Reference policy 1 (RP1): most well-trained, with 1e6 data
2. Reference policy 2 (RP2): with 1.5e5 data
3. Reference policy 3 (RP3): with 1e5 data
4. Reference policy 4 (RP4): most slightly-trained, with 5e4 data

We confirmed that AIP can improve the performance of model-based method.

Issue left:

Compared to model-free method, we want to Improve AIP so that it not only converges faster but also can match or outperforms model-free method's final performance



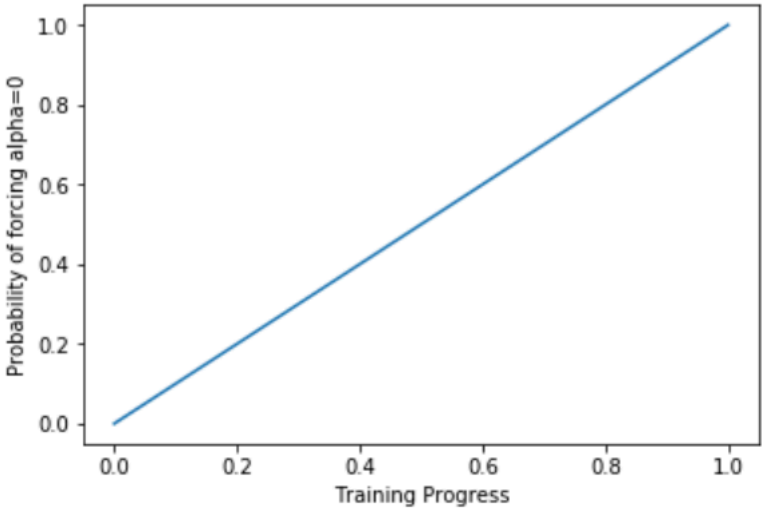
Episode Returns
Number of Data
b2048_AIP_alpha_0 (5)
b2048_AIP_alpha_0.5 (5)
b2048_AIP_alpha_1 (5)
b2048_AIP_alpha_auto_clf (5)
optimal
b2048_AIP_alpha_1 (6)
b2048_AIP_alpha_auto_clf (6)

This Week

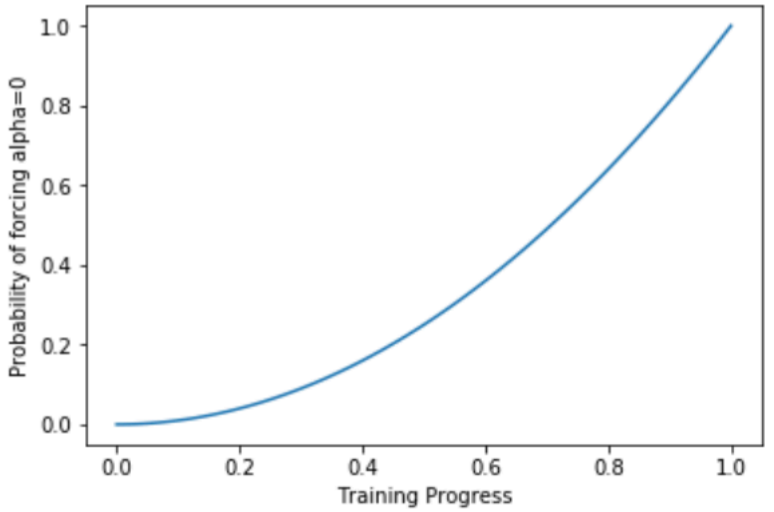
- Experiments with 6 different ways of tuning alpha in order to improve AIP's performance compared to model-free method

- Idea is that we want more alpha=0 during the training progress, so I add a probability p that forces alpha to be zero instead of the original output of alpha prediction network. And this p is going to be bigger and bigger. At the end of the training, p=1, which means we finally switch to model-free method.

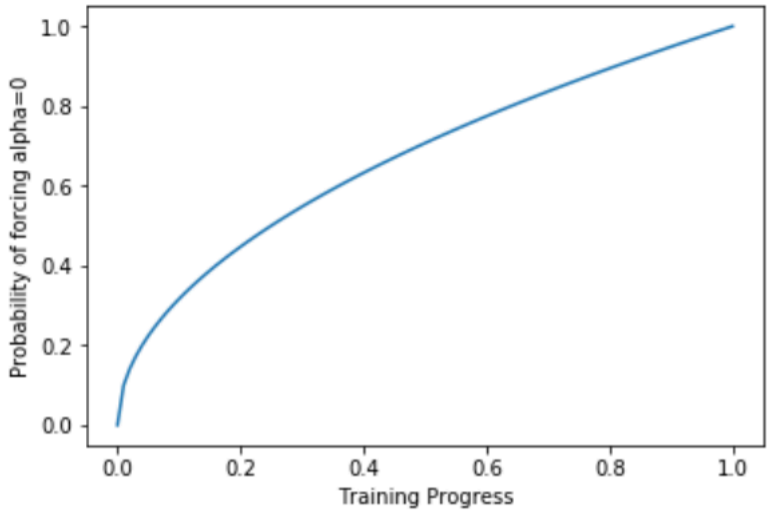
- Experimented 6 ways of p tuning



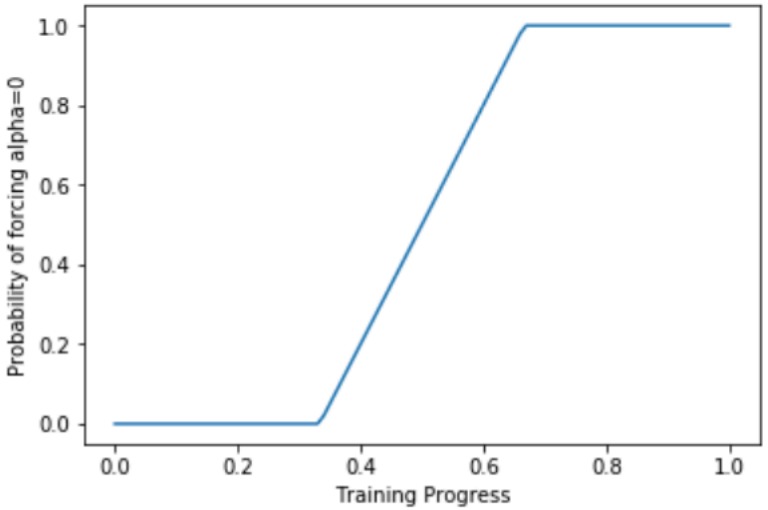
Linear (ln)



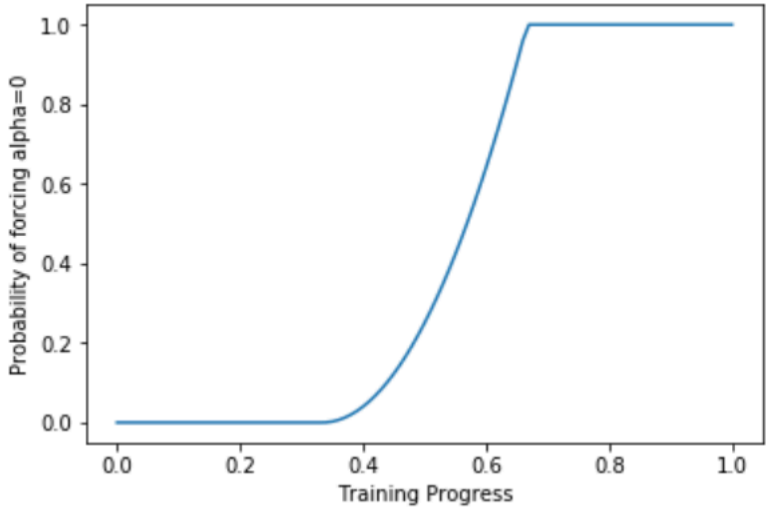
Square (sq)



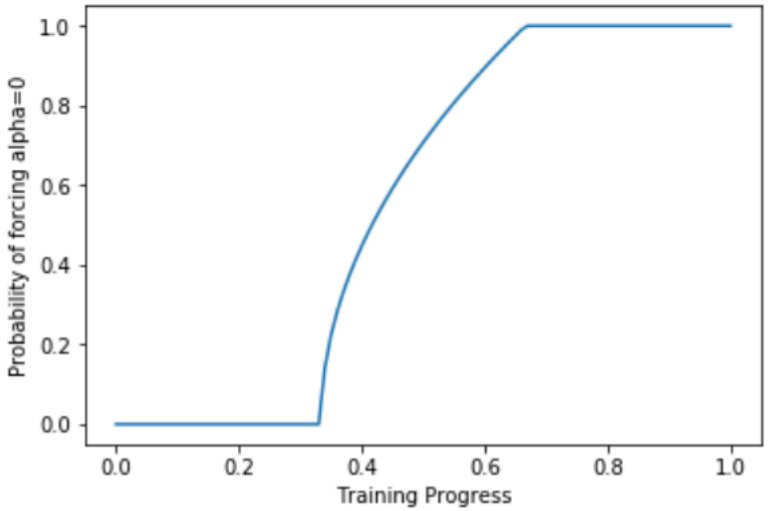
Square Root (sqrt)



ln_mid



sq_mid (still running)



sqrt_mid(still running)

Final Performance Summary(Average over 5 seeds)

	1.1	1.2	1.3	1.4	2.1	2.2	2.3	2.4	3.1	3.2	3.3	3.4
model free	-7.77	-7.77	-7.77	-7.77	-7.77	-7.77	-7.77	-7.77	-7.77	-7.77	-7.77	-7.77
model based	-5.71	-10.38	-9.79	-9.78	-9.99	-10.42	-10.99	-9.68	-11.91	-9.73	-10.05	-17.18
baseline aip	-5.80	-9.91	-8.30	-9.39	-9.03	-9.97	-8.56	-8.63	-10.43	-7.36	-9.45	-12.10
ln	-7.06	-8.26	-6.98	-7.42	-8.10	-8.91	-8.12	-7.82	-7.39	-7.01	-7.33	-8.18
ln_mid	-5.86	-8.28	-6.64	-7.44	-7.93	-8.35	-7.65	-7.25	-6.83	-6.08	-6.31	-7.08
sq	-6.08	-8.73	-7.14	-7.72	-8.63	-8.31	-8.03	-7.41	-6.23	-6.22	-6.48	-8.01
sqrt	-7.51	-7.79	-7.90	-7.60	-8.22	-8.35	-7.70	-7.18	-7.42	-6.86	-7.68	-8.17

- Generally, linear_mid works best.
- Since square works generally better than linear, sq_mid might work generally best.
- I am stilling running sq_mid and sqrt_mid, so the results are available soon.

DM3(RF det RF1)

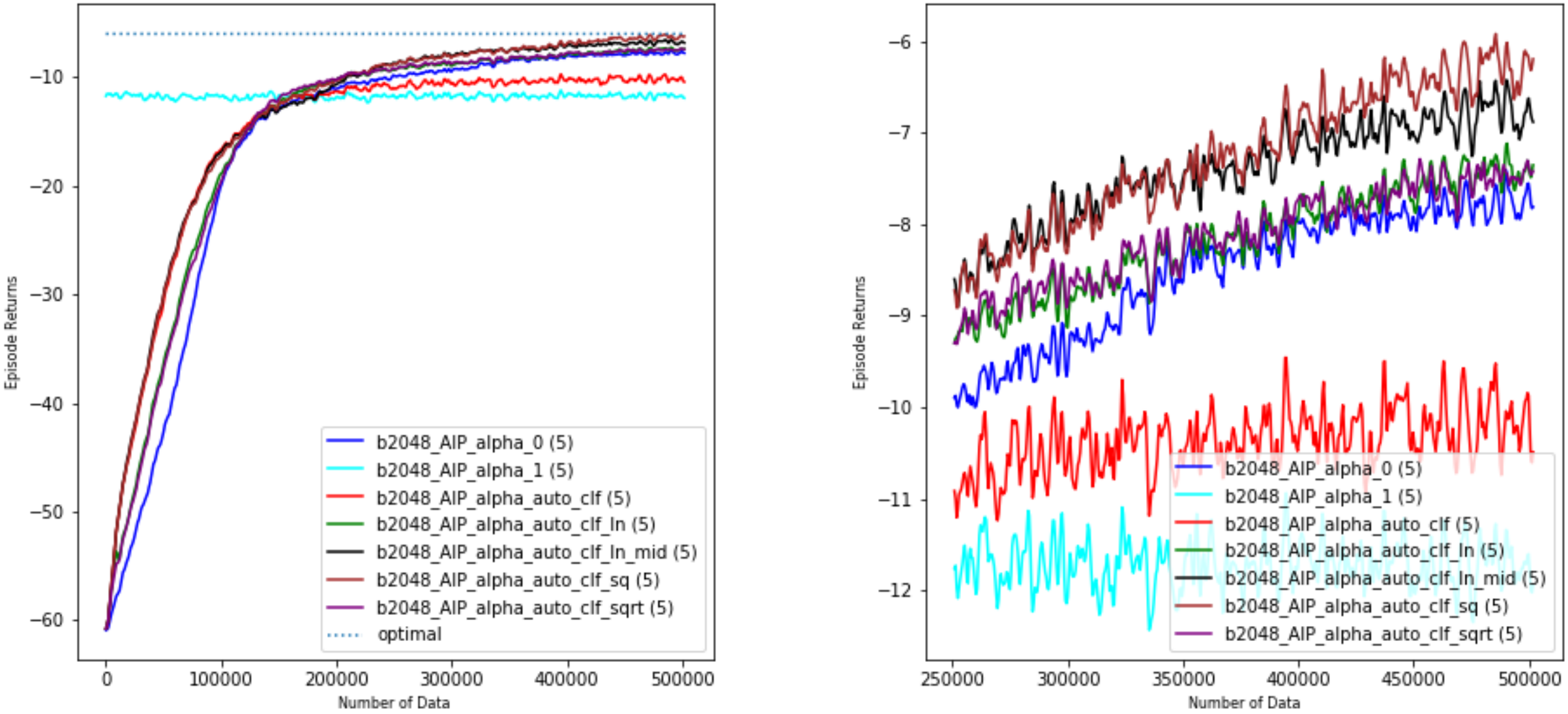
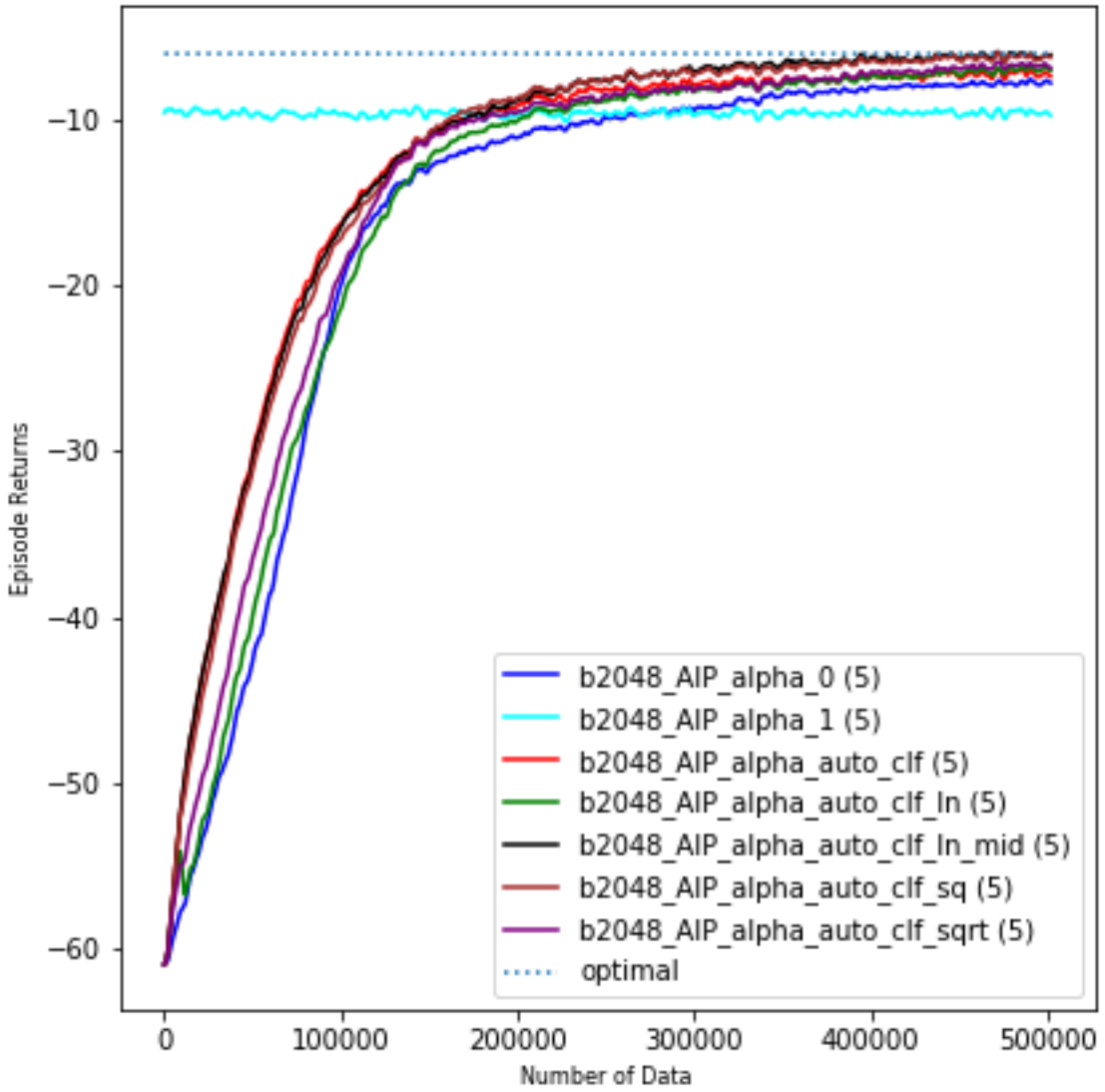


Fig 3.1 RF1

DM3(RF det RF2)



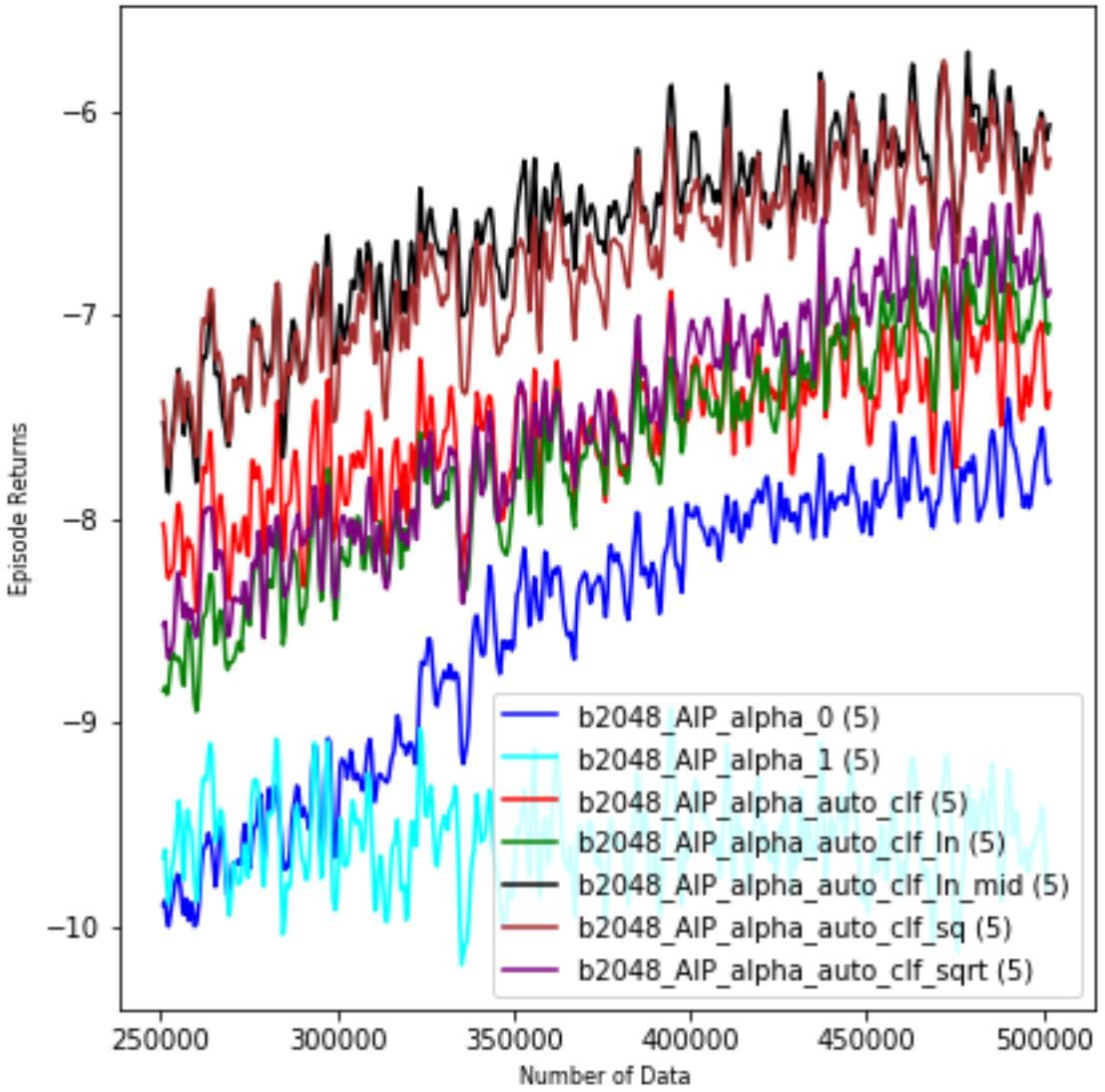


Fig 3.2 RF2

DM3(RF det RF3)

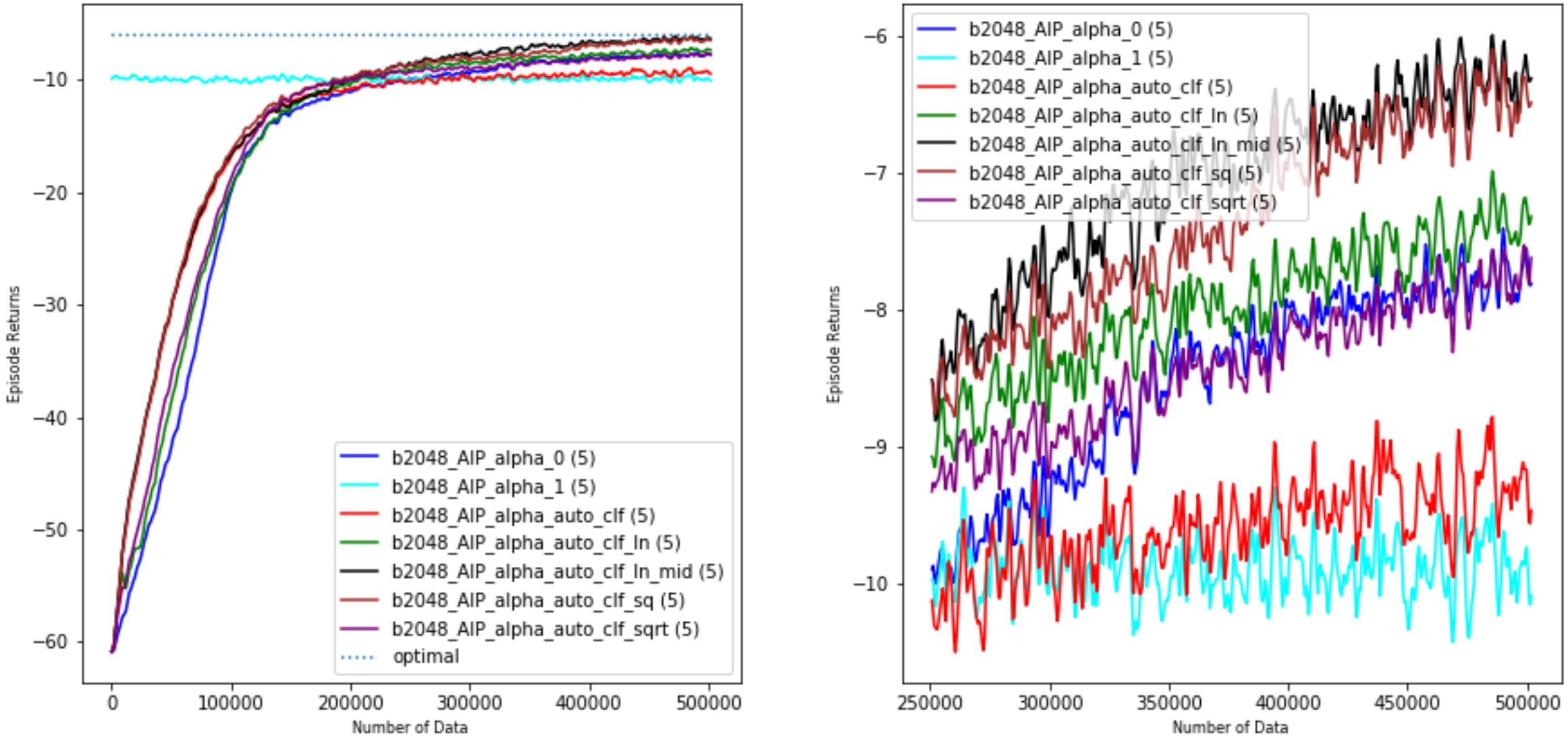
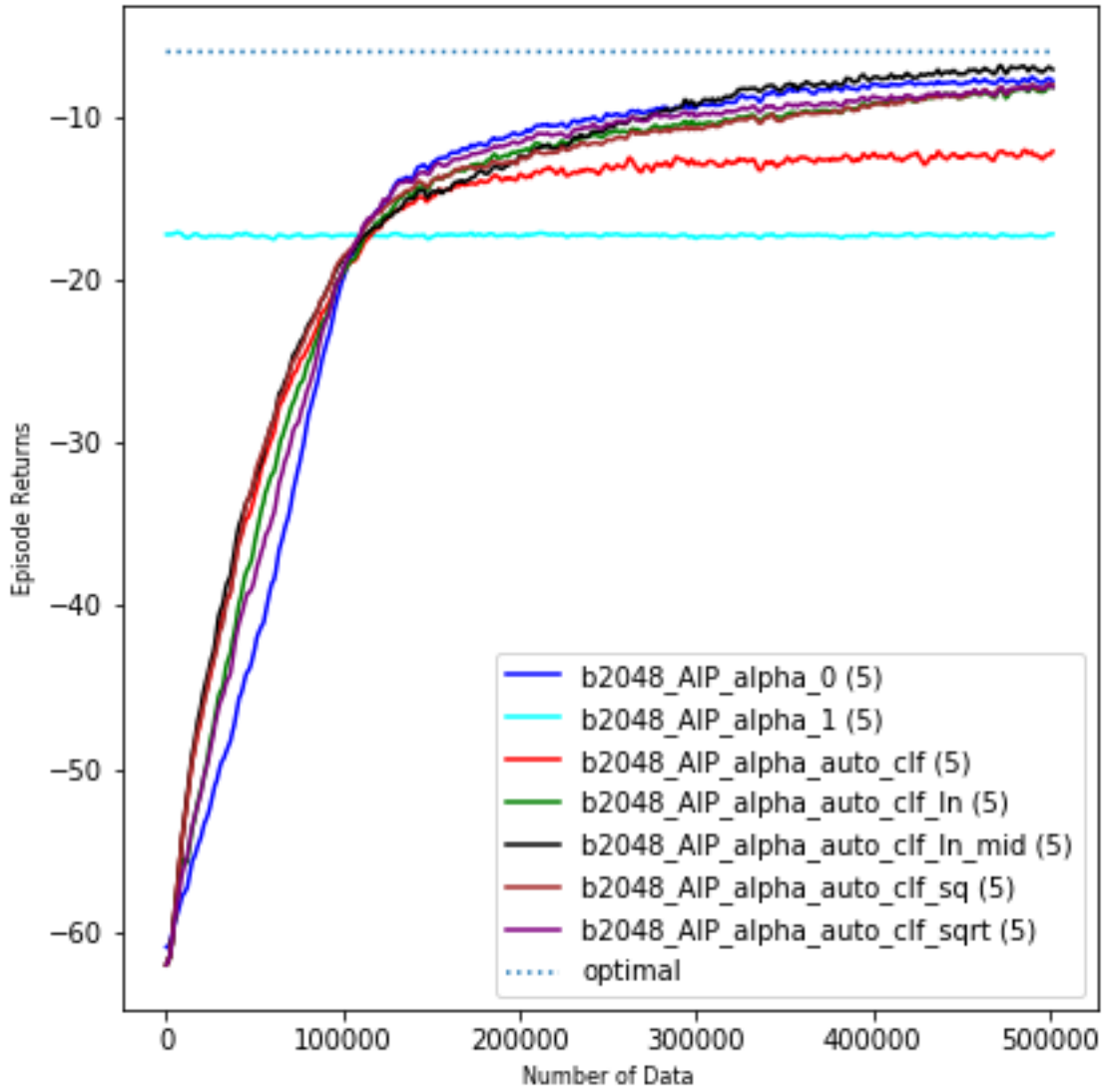


Fig 3.3 RF3

DM3(RF det RF4)



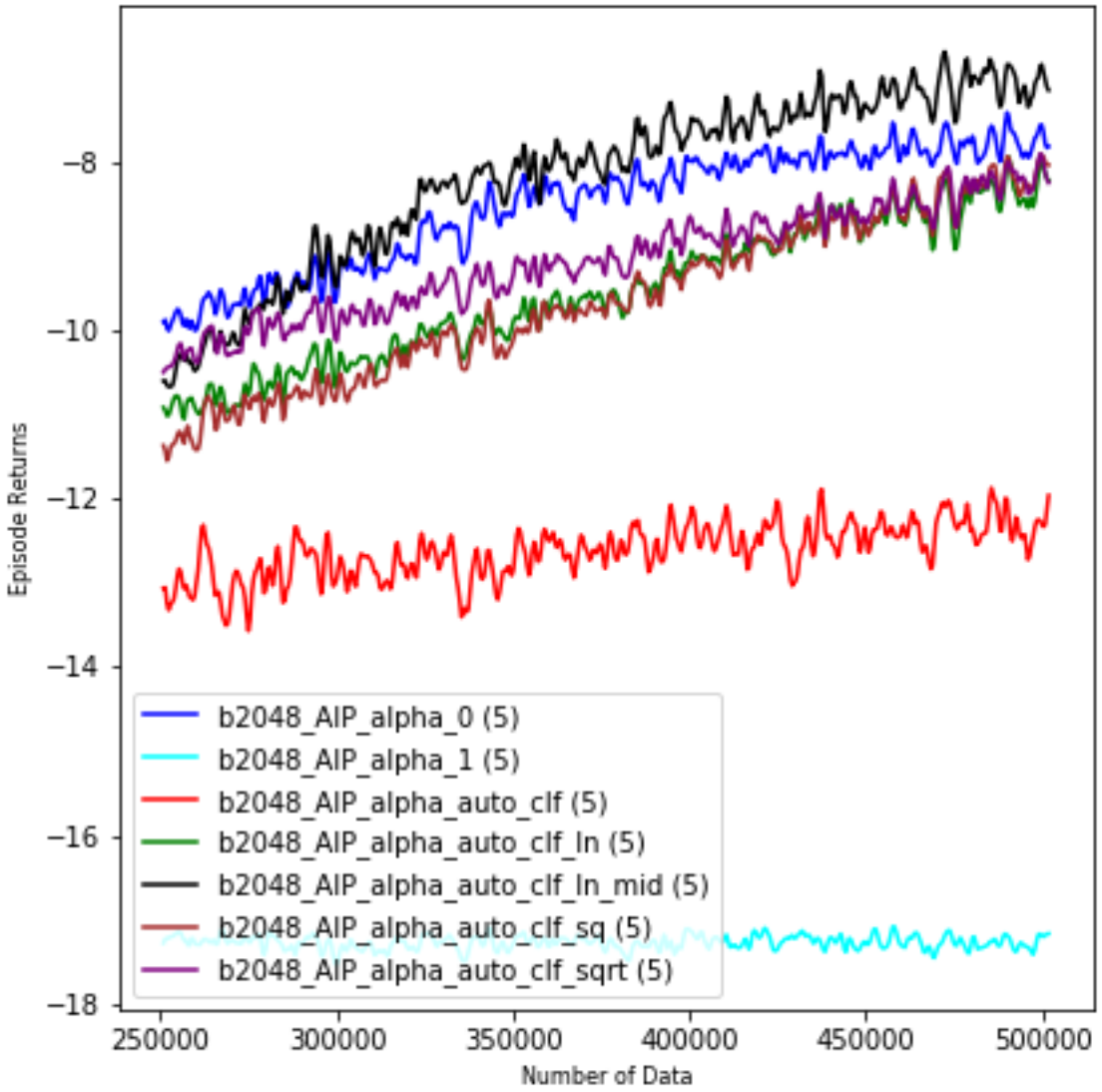


Fig 3.4 RF4

DM1(RF det)

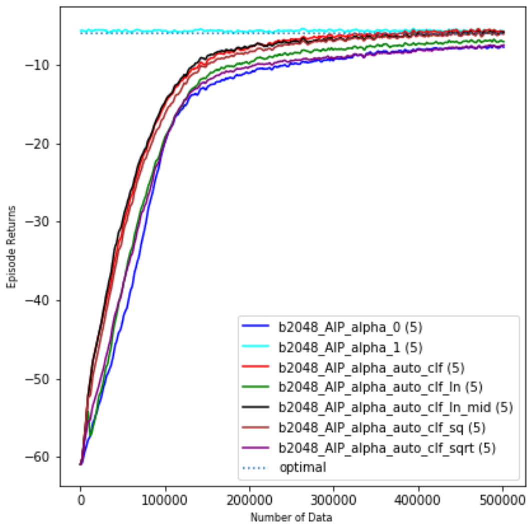


Fig 1.1 RF1

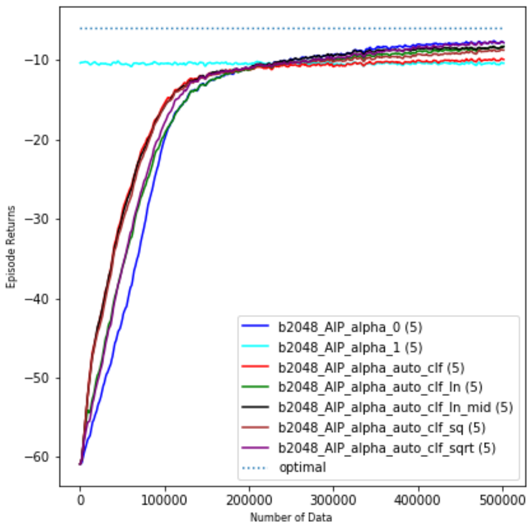


Fig 1.2 RF2

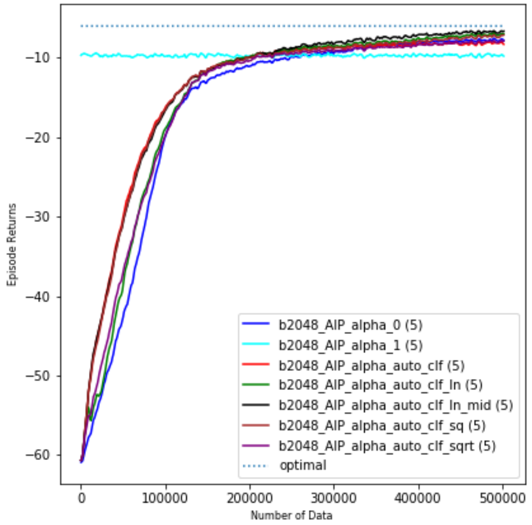


Fig 1.3 RF3

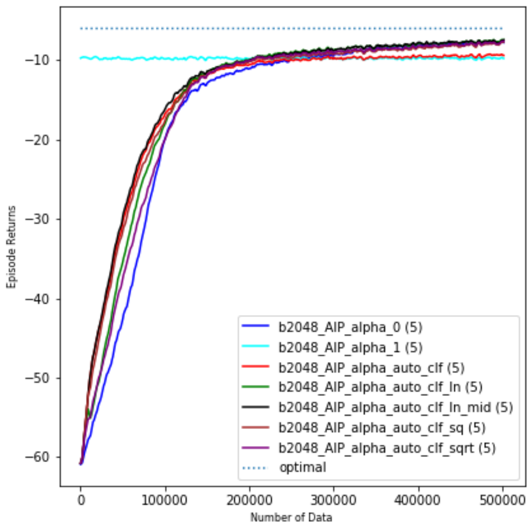
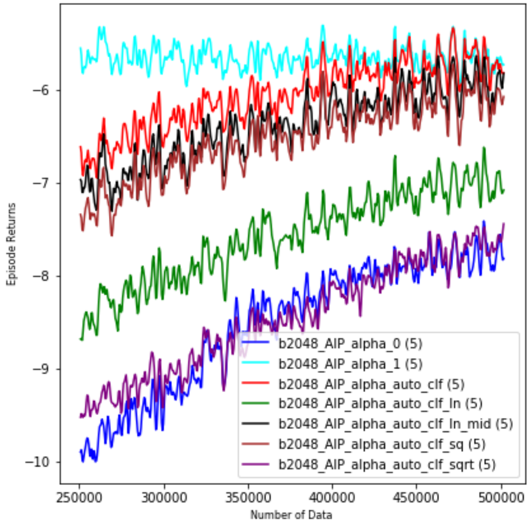
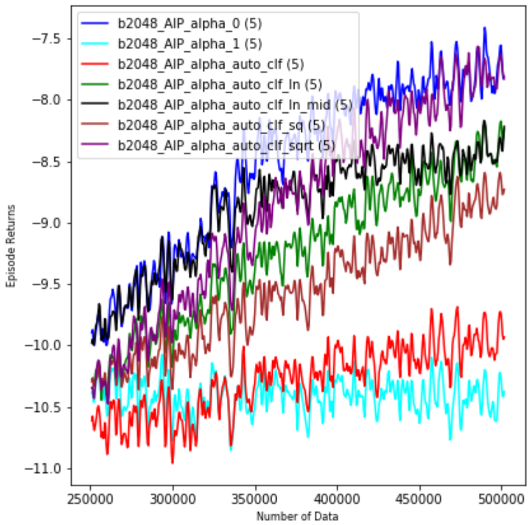
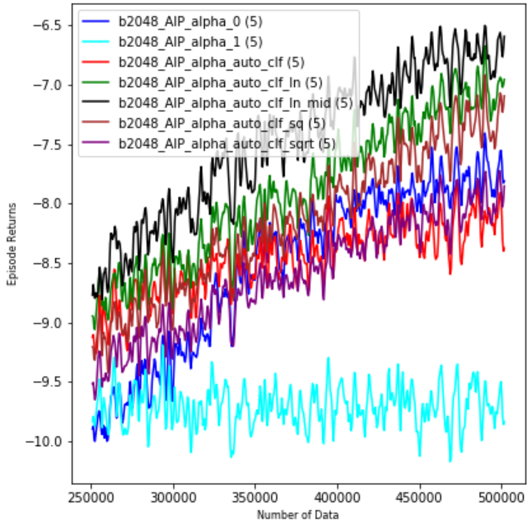
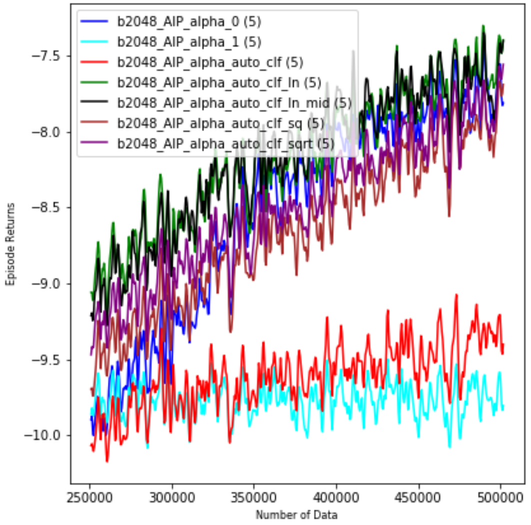


Fig 1.4 RF4









DM2(RF det)

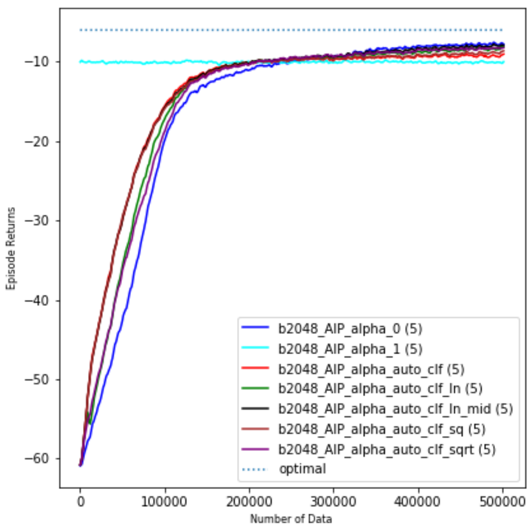


Fig 2.1 RF1

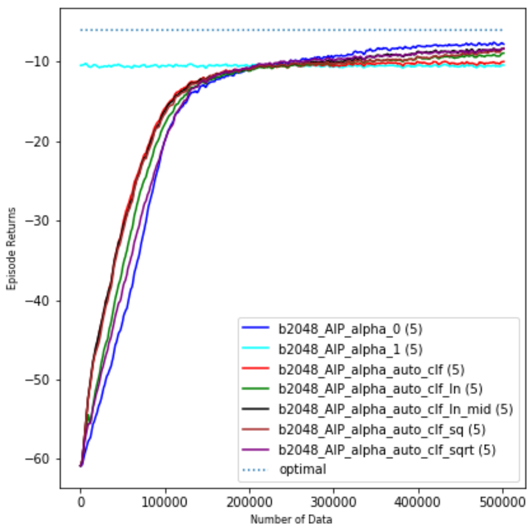


Fig 2.2 RF2

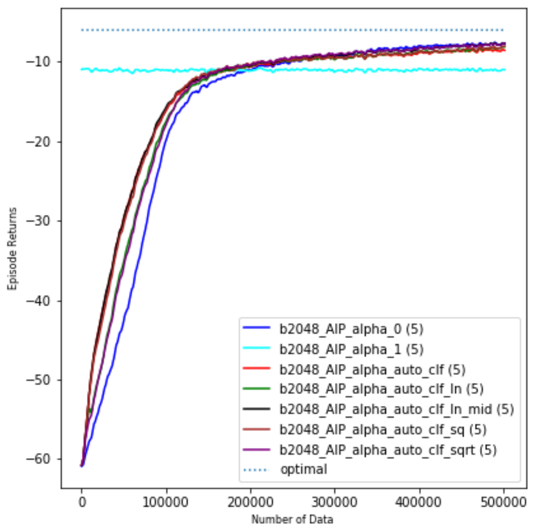


Fig 2.3 RF3

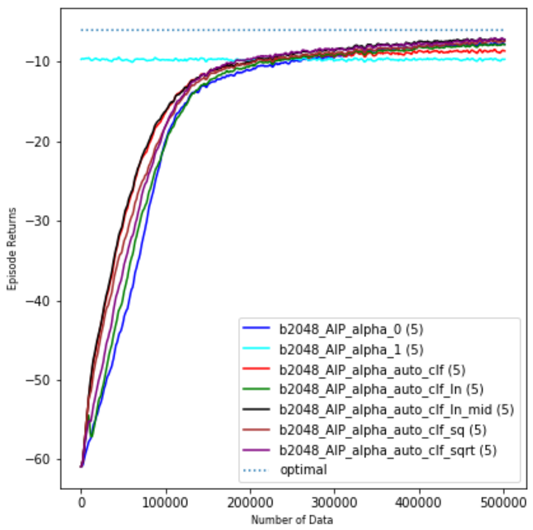
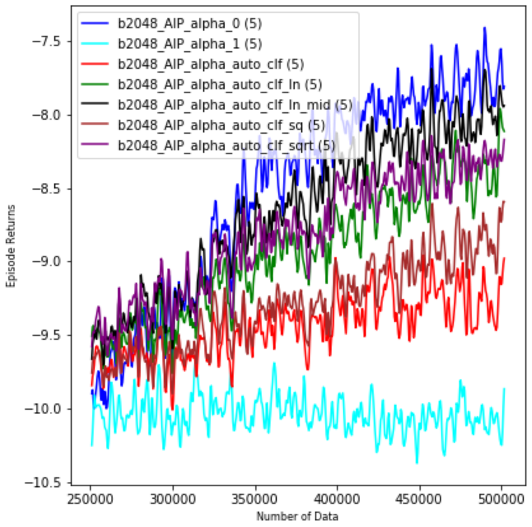
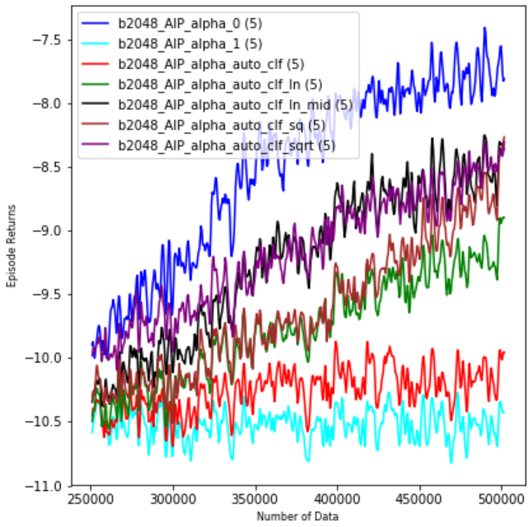
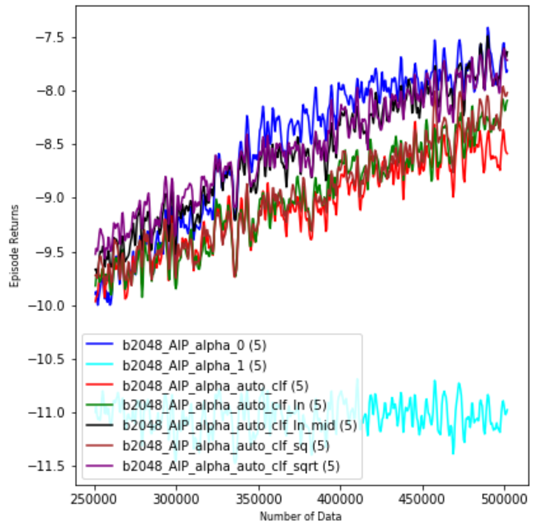
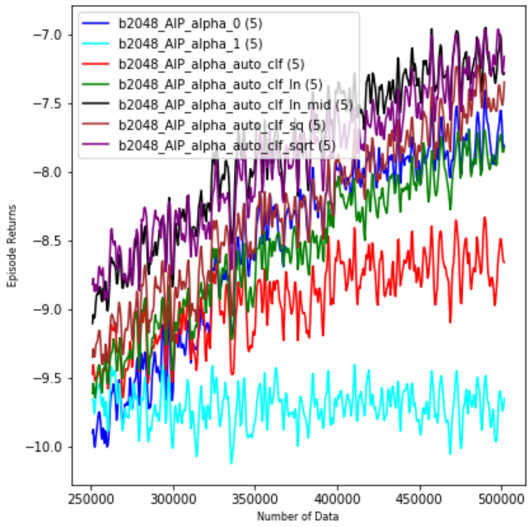


Fig 2.4 RF4

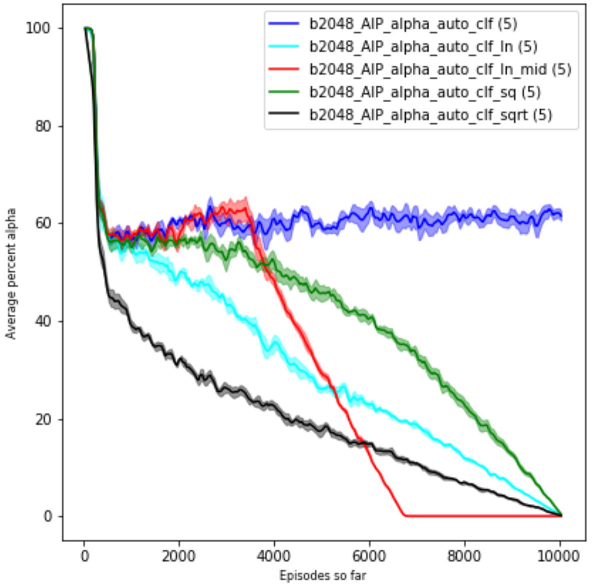


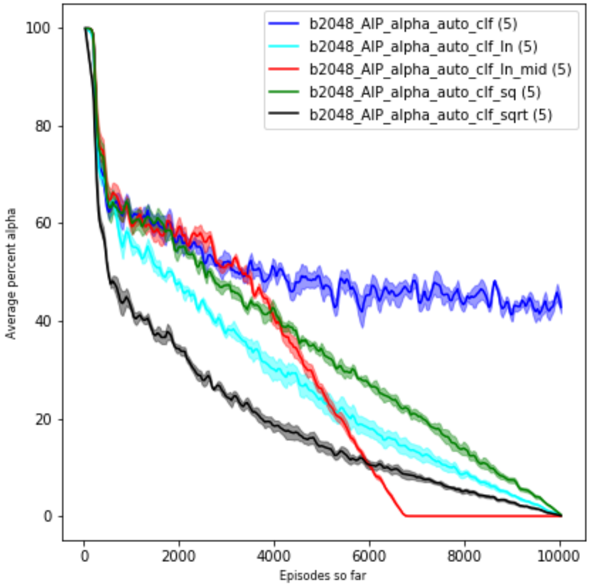


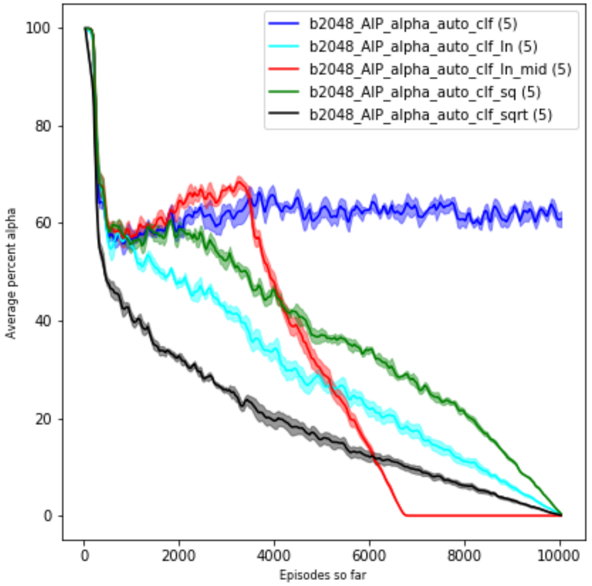


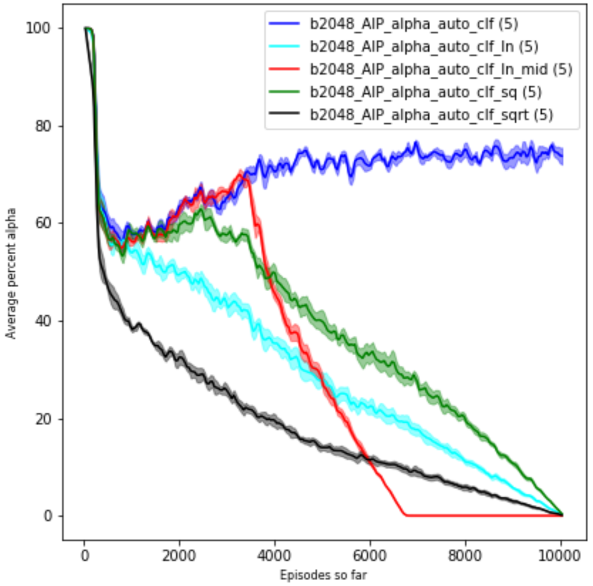


Alpha comparison (DM3)









Next plan

- Reacher with obstacles?
- Other environments: such as Fetch-Reacher in openai?

- Monte-Carlo rollout(using Q in equation 3) will take much more time (half horizon times more than current one-step reward) since for every one data, we need a full-horizon rollout to acquire true Q.