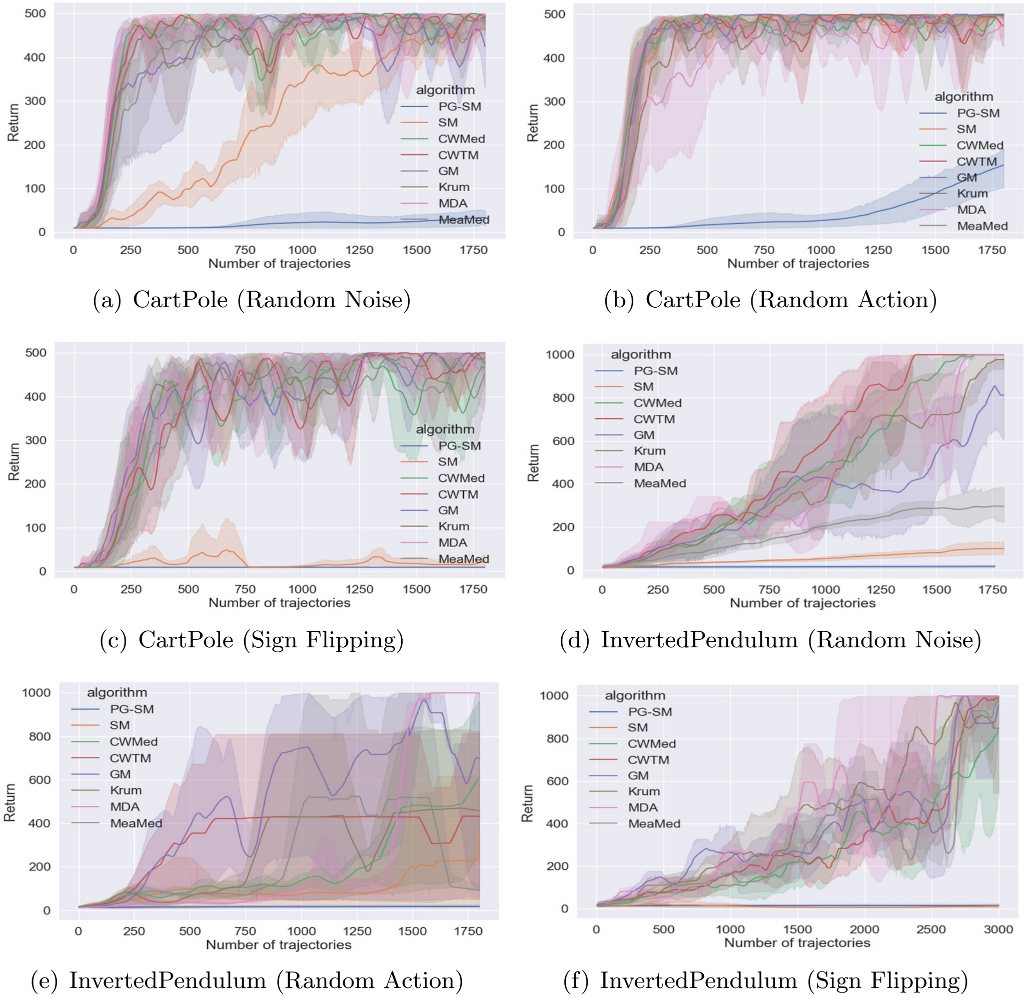


(a) CartPole (Random Noise)

(b) CartPole (Random Action)

(c) CartPole (Sign Flipping)

(d) InvertedPendulum (Random Noise)

(e) InvertedPendulum (Random Action)

(f) InvertedPendulum (Sign Flipping)

Figure 1: Evaluation results of Res-NHARPG on CartPole and InvertedPendulum. We test Res-NHARPG with the six aggregators as shown in Table 1. For baselines, we select Res-NHARPG with a simple mean (SM) function as the aggregator, which is equivalent to the original N-HARPG algorithm, and a vanilla policy gradient method with the simple mean aggregator (PG-SM). For each environment, there are ten workers, of which three are adversaries, and we simulate three types of attacks: random noise, random action, and sign flipping. It can be observed that N-HARPG outperforms PG and Res-NHARPG with those (f, λ) aggregators can effectively handle multiple types of attacks during the learning process. Note that each algorithm in each subfigure is run five times with different random seeds, for which the mean and 95% confidence interval are shown as the solid line and shadow area, respectively.

6. Evaluation

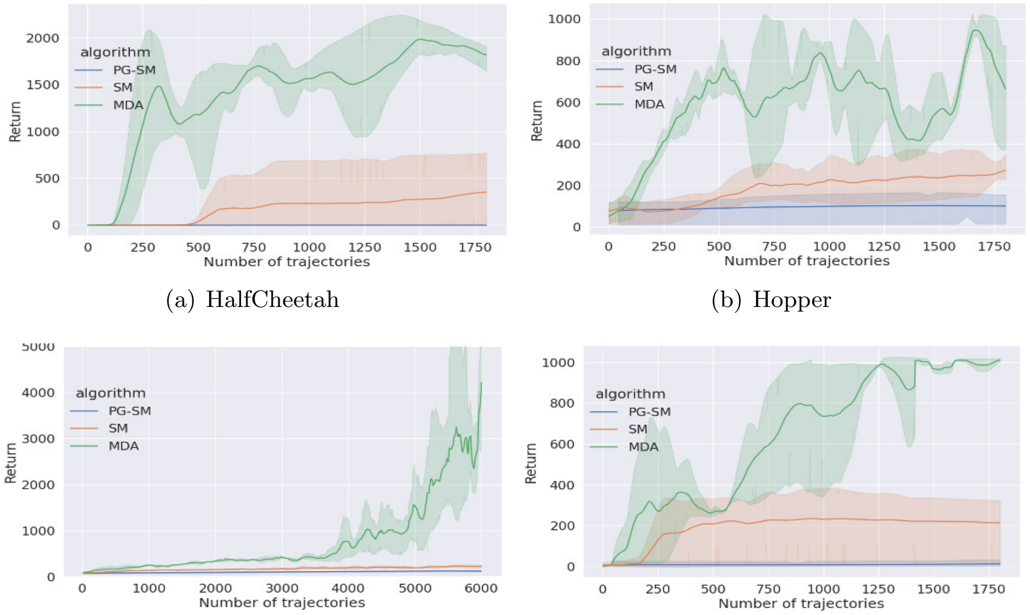
To show the effectiveness of our algorithm design (i.e., Res-NHARPG), we provide evaluation results on two commonly-used continuous control tasks: CartPole-v1 from OpenAI Gym (Brockman et al. (2016)) and InvertedPendulum-v2 from MuJoCo (Todorov et al. (2012)). For each task, there are ten workers to individually sample trajectories and compute gradients, and three of them are adversaries who would apply attacks to the learning process. Note that we do not know which worker is an adversary, so we cannot simply ignore certain gradient estimates to avoid the attacks. We simulate three types of attacks to the learning process: random noise, random action, and sign flipping. By 'random noise' or 'sign flipping', the real estimated policy gradients are altered by adding random noises or multiplying by a negative factor, respectively. While, for 'random action', adversarial workers would select random actions at each step, regardless of the state, when sampling trajectories for gradient estimations. Unlike the other attacks, 'random action' does not directly change the gradients, making it more challenging to detect. Also, 'random action' is different from the widely-adopted ϵ-greedy exploration method, since the action choice is fully random and the randomness does not decay with the learning process.

As shown in Figure 1, we compare among Res-NHARPG with the six aggregators shown in Table 1 (i.e., MDA, CWTM, CWMed, Krum, MeaMed, GM), Res-NHARPG with a simple mean (i.e., SM) aggregator, and Vanilla Policy Gradient with SM (i.e., PG-SM). Specifically, the simple mean aggregator only averages the estimates of gradients from all workers without considering the existence of adversaries, so Res-NHARPG with SM is equivalent to the state-of-the-art policy gradient method – N-HARPG (Fatkhullin et al. (2023)). Through these comparisons, we expect to show the necessity to utilize these (f, λ) aggregators in case of adversaries and the robustness of them for various types of environments and attacks.

In Figure 1, we present the learning process of the eight algorithms in two environments under three types of attacks. In each subfigure, the x-axis represents the number of sampled trajectories; the y-axis records the acquired trajectory return of the learned policy during evaluation. Each algorithm is repeated five times with different random seeds. The average performance and 95% confidence interval are shown as the solid line and shadow area, respectively. Codes for our experiments have been submitted as supplementary material and will be made public.

Comparing the performance of N-HARPG (i.e., SM) and Vanilla PG (i.e., PG-SM), we can see that N-HARPG consistently outperforms, especially in Figure 1(a) and 1(b) of which the task and attacks are relatively easier to deal with. For the 'random action' attack (Figure 1(b) and 1(e)), which does not directly alter the gradient estimates, N-HARPG shows better resilience. However, in more challenging tasks (e.g., InvertedPendulum) and under stronger attacks (e.g., sign flipping), both N-HARPG and Vanilla PG would likely fail, which calls for effective aggregator functions.

For CartPole, the maximum trajectory return is set as 500. Res-NHARPG, implemented with each of the six aggregators, can reach that expert level within 1000 trajectory samples, with slight difference in the convergence rate. As for InvertedPendulum, not all aggregators achieve the expert level (i.e., a trajectory return of 1000), yet they all demonstrate superior performance compared to those employing only the simple mean aggregator. It's worth noting that Res-NHARPG with the MDA aggregator consistently converges to the expert level across all test cases, showing its robustness. Moreover, the ‘random action’ attack brings more challenges to the aggregators, as the influence of random actions during sampling on the gradient estimates is indirect while all aggregators filter abnormal estimates based on gradient values.



(a) HalfCheetah

(b) Hopper

(c) Inverted Double Pendulum

(d) Walker

Figure 2: Res-NHARPG with the MDA aggregator consistently outperform the baselines: N-HARPG (i.e., SM) and Vanilla PG (i.e., PG-SM), on a series of MuJoCo tasks.

Previous evaluation results have shown the superiority of Res-NHARPG with the MDA aggregator. To further demonstrate its applicability, we compare it with the baselines: N-HARPG (i.e., SM) and Vanilla PG (i.e., PG-SM), on a series of more challenging MuJoCo tasks: HalfCheetah, Hopper, Inverted Double Pendulum, Walker, of which the result is shown as Figure 2. Our algorithm consistently outperforms the baselines when adversaries (specifically, random noise) exist, and relatively, N-HARPG performs better than Vanilla PG. Note that our purpose is not to reach SOTA performance but to testify the effectiveness of aggregators, so the three algorithms in each subfigure share the same set of hyperparameters (without heavy fine-tuning). In Figure 2, we illustrate the training progress, up to a maximum number of sampled trajectories (6000 for the Inverted Double Pendulum and 1800 for other tasks), for each algorithm by plotting their episodic returns. The advantage of Res-NHARPG is more significant when considering the peak model performance. For instance, the highest evaluation score achieved by Res-NHARPG on Inverted Double Pendulum can exceed 9000, i.e., the SOTA performance as noted in (Weng et al. (2022)), while the baselines’ scores are under 500.