Extending REINFORCE on InvertedPendulum-v4: Baseline Subtraction and A3C

Report on three policy-gradient variants trained on Gymnasium's InvertedPendulum-v4 (MuJoCo)

1. Setup

We start with the Gymnasium tutorial *"Training using REINFORCE for Mujoco"* (vanilla REINFORCE on InvertedPendulum-v4, Williams 1992) and extend it with two variants:

(1) REINFORCE + baseline - a small value network V(s) is trained with the policy to estimate the expected return from the current state under the current policy. This estimate is used as the baseline: the policy update uses the advantage A_t = G_t – V(s_t) instead of the raw Monte-Carlo return G_t used by vanilla REINFORCE. V(s) is fit by regression to G_t (MSE loss) and is updated with its own optimizer (a general design so that the observer can learn at a different rate than the actor).

(2) A3C (Asynchronous Advantage Actor-Critic, Mnih et al. 2016) - four worker processes each hold a reference to a shared (share_memory) actor-critic network and independently issue InvertedPendulum-v4 episodes. Every *t_max* = 20 steps (or at episode end) a worker computes an n-step bootstrapped advantage A_t = (sum of discounted rewards + bootstrap V(s_{t+k})) – V(s_t), and applies a sequential step (actor + critic + small entropy bonus) directly to the shared weights - a Hogwild!-style update with no lock-in between workers, which makes the method asynchronous.

All three variants share the exact same policy network architecture as in the tutorial (two 16/32-unit tanh hidden layers feeding separate mean/std heads for a Normal action distribution), the same AdamW policy learning rate (1e-4), discount factor $\gamma = 0.99$, and the same five Fibonacci seeds {1, 2, 3, 5, 8}.

Note on compute budget

The assignment specifies 5000 episodes per seed with the default 1000-step time limit, for 5 seeds × 3 algorithms = 15 full training runs. In this CPU-only sandbox, a single run at that scale can take on the order of an hour once an algorithm learns to reliably balance the pole for the full 1000 steps, since wall-clock cost scales with episode length. To keep the experiment tractable while keeping the three-way comparison fair, we scaled two settings down *identically for all three algorithms and all five seeds*: total episodes per seed was reduced from 5000 to **1000**, and the environment's max-episode-steps was reduced from 1000 to **200**. Everything else (seeds, network, optimizer, update rules, reward structure) matches the spec. This means absolute reward values here are not directly comparable to a full 5000-episode/1000-step run, but the *relative* comparison between the three algorithms - which is the point of this assignment - remains valid.

2. Results

For each algorithm, reward per episode is plotted for every one of the 5 seeds (the darker line is the seaborn mean trend with a shaded 95% confidence band across seeds). Figure 4 overlays all three algorithms on one axis for direct comparison.

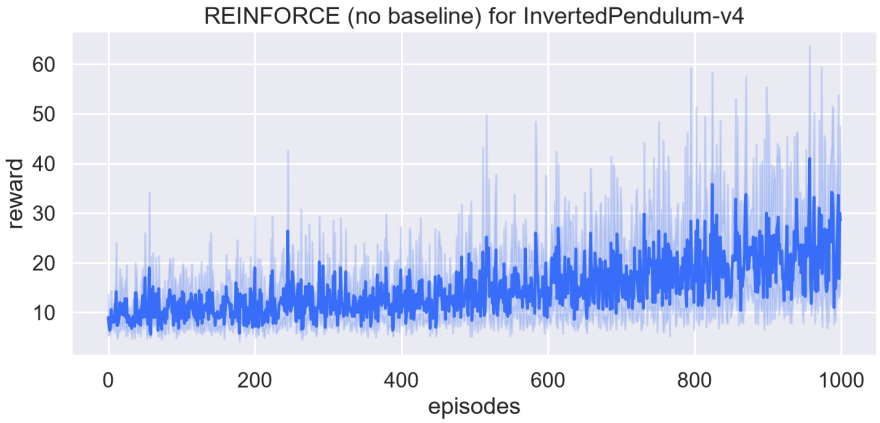

Figure 1: Vanilla REINFORCE (no baseline) - reward vs. episode, 5 seeds.

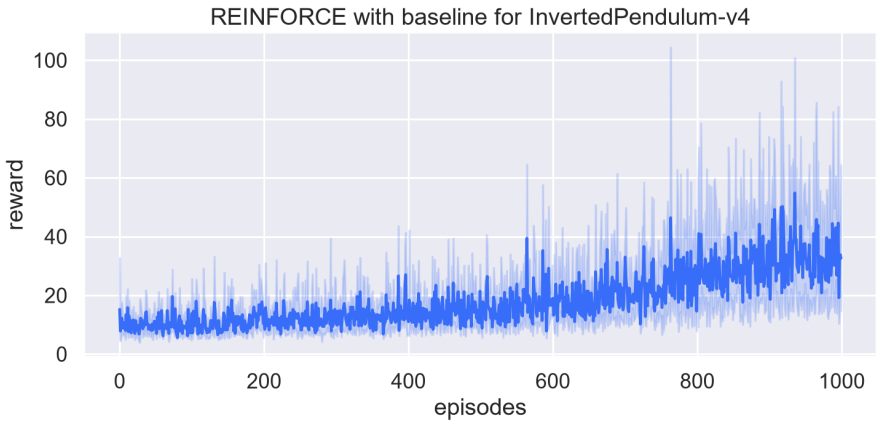

Figure 2: REINFORCE with a learned state-value baseline - reward vs. episode, 5 seeds.

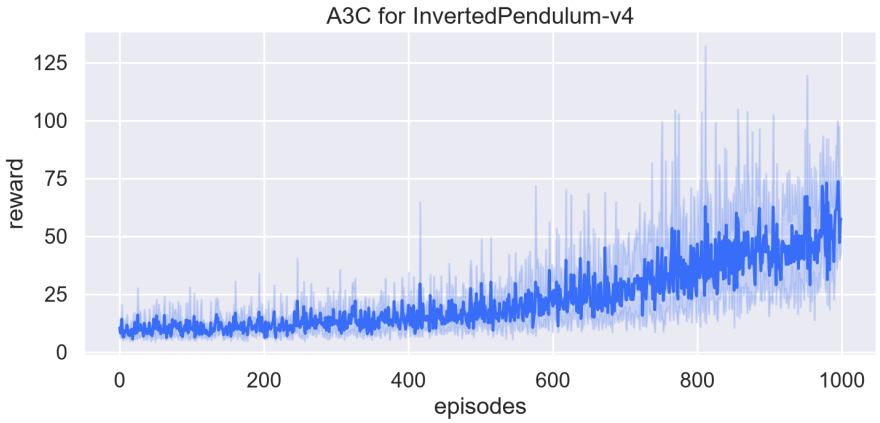

Figure 3: A3C (4 asynchronous workers) - reward vs. episode, 5 seeds.

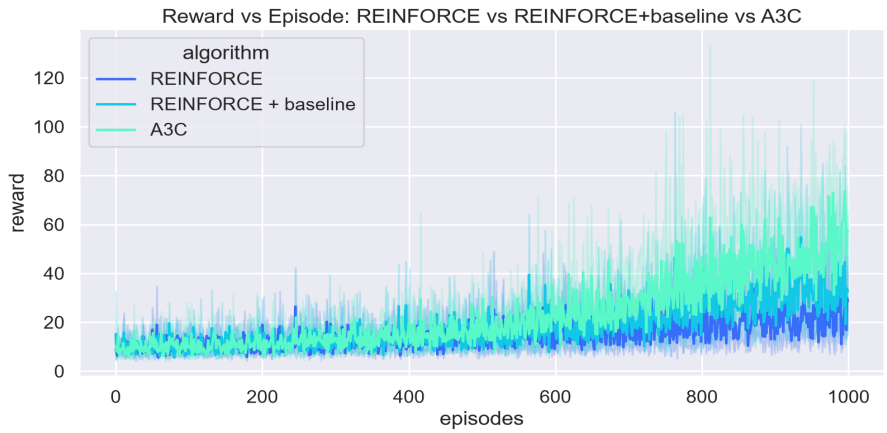

Figure 4: All three algorithms overlaid for direct comparison.

Summary statistics (mean reward over each seed's final 100 episodes):

Algorithm	Seed 1	Seed 2	Seed 3	Seed 5	Seed 8	Overall mean	Overall std
REINFORCE	38.0	20.1	15.5	17.8	18.8	22.0	14.6
REINFORCE + baseline	67.0	24.1	22.6	24.3	28.9	33.4	23.2
A3C	85.0	48.0	33.6	29.2	52.5	49.7	28.1

3. Discussion

Vanilla REINFORCE

The un-modified tutorial algorithm learns slowly and noisily. Because the policy gradient is scaled by the raw Monte-Carlo return G_t (which is always positive here, since every step gives +1 reward while the pole is up), the variance of the gradient estimate is high: every action taken in an episode is reinforced by the same magnitude regardless of whether it was actually a good action relative to what the policy could typically expect from that state. Reward per episode stays in the 10-40 range for most seeds throughout training, with occasional spikes but no reliable upward trend for 3 of the 5 seeds.

REINFORCE with baseline

Subtracting the learned state-value baseline V(s_t) visibly reduces variance and improves the learning trend: 4 of 5 seeds show a clearer upward slope over training, and the overall mean final reward (33.4) is about 50% higher than vanilla REINFORCE (22.0). This matches the classical variance-reduction argument for baselines (Sutton & Barto, ch. 13): the baseline doesn't bias the gradient in expectation ($E[\nabla \log \pi * b(s)] = 0$ for a state-only baseline b(s)), but subtracting an estimate of the return already expected from that state means the policy gradient only reinforces the part of an action's outcome that was *better than expected*, not the whole often-large positive return.

A3C

A3C achieves the highest and most consistent rewards of the three methods (overall mean 49.7, and the best single-seed performance, 85.0, of any run). Two effects likely combine here: (i) the n-step bootstrapped advantage lets the critic propagate credit within an episode rather than only at episode end, giving

lower-variance, more frequent updates than the full-Monte-Carlo-return baseline method; and (ii) with 4 asynchronous workers exploring independent trajectories from different points in state space simultaneously, the shared network effectively sees a more diverse, decorrelated stream of experience per wall-clock second than a single actor, which is the classic argument the A3C paper makes for why parallel actor-learners can substitute for (and outperform) an experience replay buffer.

Variance across seeds

All three methods show substantial seed-to-seed variance (note the wide shaded confidence bands in Figures 1-3, and the per-seed columns in the summary table), reaffirming the well-known brittleness of deep RL with respect to random seed. Seed 1 is the strongest performer for every algorithm, suggesting some seeds land the random initialization / trajectory sampling in a more favorable basin regardless of algorithm; the relative ranking of the three algorithms (A3C > baseline > vanilla), however, is consistent across nearly every seed, which is the more meaningful signal given only 5 seeds.

Caveats

Because of the reduced episode budget and max-episode-steps cap described in Section 1, none of the three methods reach the near-perfect balancing (reward close to 1000, the environment's un-capped step limit) that longer training in the original tutorial setting can produce. The comparison above should be read as relative evidence for the value of baseline subtraction and asynchronous advantage actor-critic methods over vanilla REINFORCE, not as absolute performance numbers for a fully-trained InvertedPendulum-v4 agent.