Bilevel Policy Optimization with Nyström Hypergradients

BROWN

Arjun Prakash* Naicheng He* Denizalp Goktas Amy Greenwald

Questions

How can we improve actor-critic algorithms by taking into account the interplay between the actor and critic? How can we efficiently and accuratley compute hypergradients?

Abstract

Actor-critic (AC) can be cast as a bilevel problem. We propose BLPO, which nests the critic and updates the actor with a Nyström hypergradient that accounts for critic adaptation. Under a linear critic, we prove polynomial-time convergence to a local strong-Stackelberg equilibrium. Empirically, BLPO matches or outperforms PPO across discrete and continuous control tasks.

Introduction

Given functions $f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$ and $g : \mathbb{R}^m \to \mathbb{R}$, a(n unconstrained) bilevel opmization problem can be formulated as follows:

$$\min_{\boldsymbol{x}\in\mathbb{R}^n} \Phi(\boldsymbol{x}) \doteq f(\boldsymbol{x}, \boldsymbol{y}^*(\boldsymbol{x})) \qquad \text{subject to } \boldsymbol{y}^*(\boldsymbol{x}) \in \mathcal{Y}^*_{\boldsymbol{x}} \doteq \arg\min_{\boldsymbol{y}\in\mathbb{R}^m} g_{\boldsymbol{x}}(\boldsymbol{y}) \tag{1}$$

A solution to a bilevel optimization problem (also known as a Stackelberg equilibrium) comprises a pair $(\boldsymbol{x}^*, \boldsymbol{y}^*) \in (\mathbb{R}^n, \mathbb{R}^m)$ s.t. $\boldsymbol{x}$ optimizes $\Phi(\boldsymbol{x})$ subject to the constraint that $\boldsymbol{y}^*$ optimizes $g_{\boldsymbol{x}}(\boldsymbol{y})$.

Hypergradient

To calculate the gradient of the leader, we must differentiate through the follower's best response:

$$\nabla f(\boldsymbol{x}, \boldsymbol{y}^*(\boldsymbol{x})) = \nabla_{\boldsymbol{x}} f(\boldsymbol{x}, \boldsymbol{y}) + \nabla \boldsymbol{y}^*(\boldsymbol{x}) \nabla_{\boldsymbol{y}} f(\boldsymbol{x}, \boldsymbol{y})$$

Which using the IFT becomes:

$$\nabla \boldsymbol{y}^*(\boldsymbol{x}) \nabla_{\boldsymbol{y}} f(\boldsymbol{x}, \boldsymbol{y}) = \underbrace{-\nabla^2_{\boldsymbol{xy}} g_{\boldsymbol{x}}(\boldsymbol{y}) \underbrace{(\nabla^2_{\boldsymbol{yy}} g_{\boldsymbol{x}}(\boldsymbol{y}))^{-1} \nabla_{\boldsymbol{y}} f(\boldsymbol{x}, \boldsymbol{y})}_{\boldsymbol{v}}}_{\text{Jacobian vector product}} \tag{2}$$

The Nyström method allows us to approximate the IHVP $\boldsymbol{v}$ by:

$$\hat{\boldsymbol{v}} = (H_q + \alpha \boldsymbol{I})^{-1} \nabla_{\boldsymbol{y}} f(\boldsymbol{x}, \hat{\boldsymbol{y}})$$

where

$$(H_q + \alpha \boldsymbol{I})^{-1} = \frac{1}{\alpha} \boldsymbol{I} - \frac{1}{\alpha^2} H_{[:,Q]} \left(H_{[Q,Q]} + \frac{1}{\alpha} H_{[:,Q]}^\top H_{[:,Q]} \right)^{-1} H_{[:,Q]}^\top$$

.

Performance

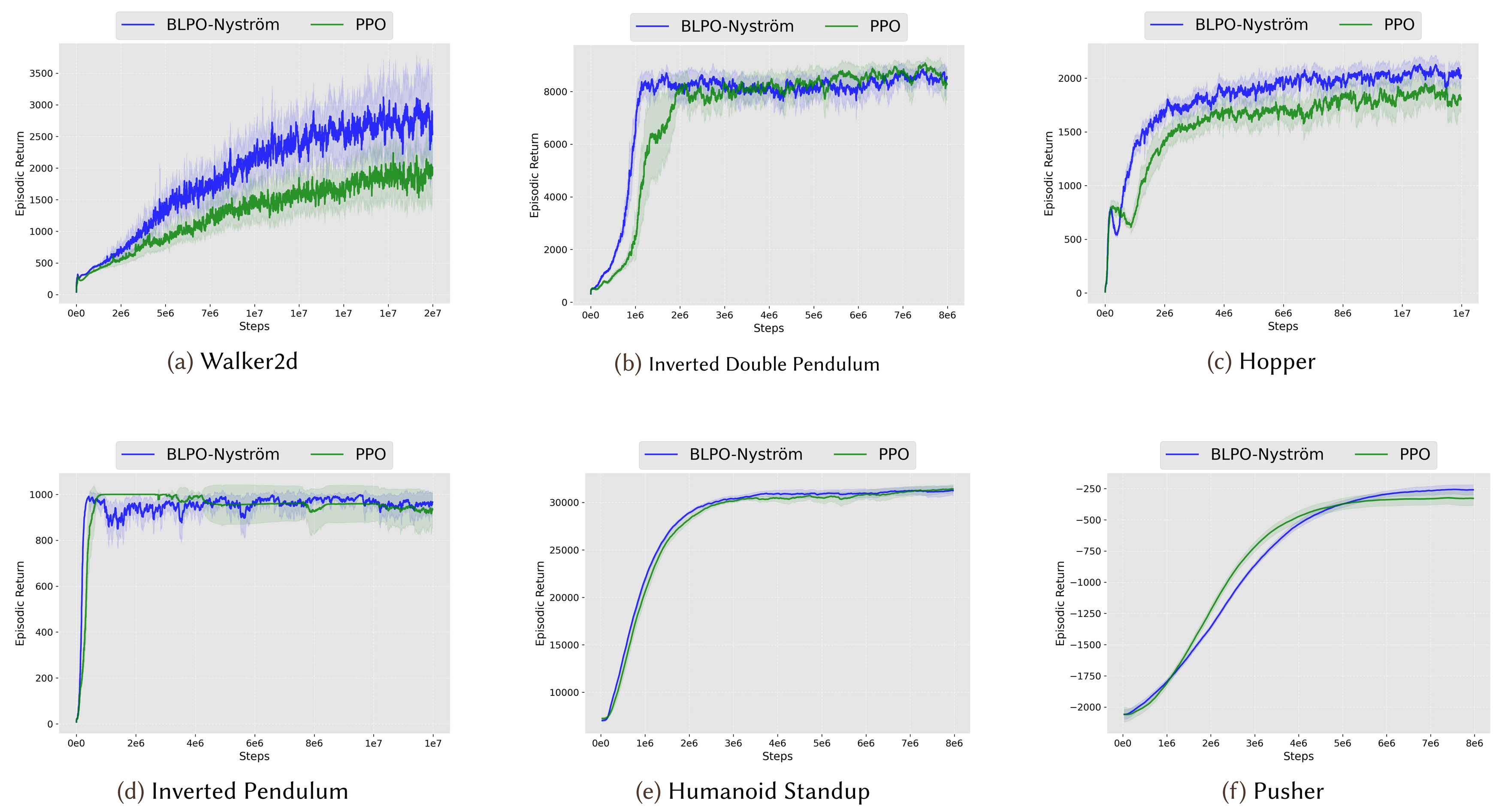

Figure 1. In continuous control tasks, BLPO either outperforms PPO or performs comparably.

Runtime

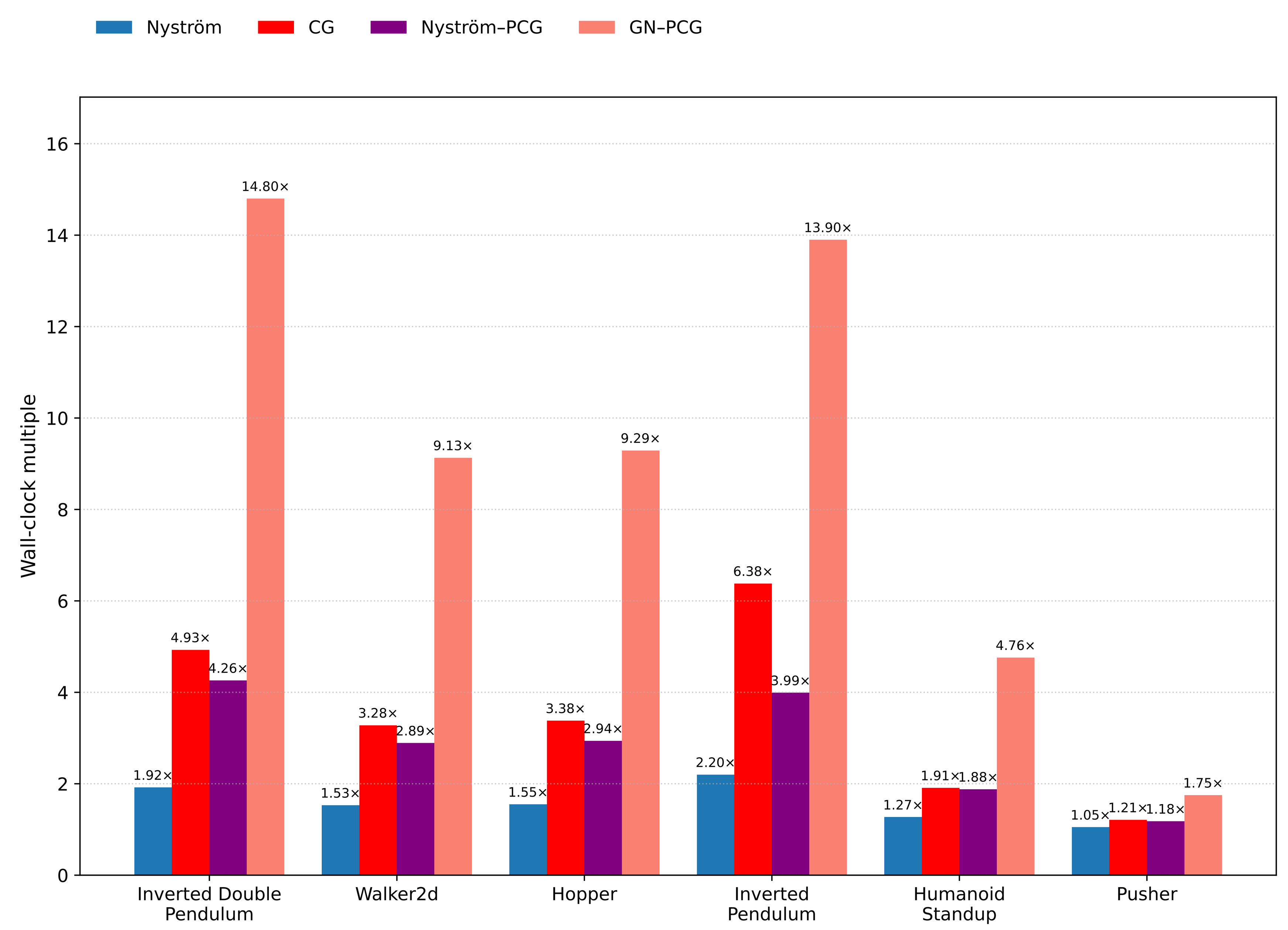

Figure 2. Runtimes relative to PPO. The The Nyström method is faster than CG (max 50 iters) and preconditioned variants. All methods achieve comparable performance.

Vanilla Actor-Critc

AC algorithms like PPO [3] and SAC [2] update the actor and critic simultaneously, meaning each updates its network parameters during iteration $t+1$, given the other's parameters at iteration t. Simultaneous updating corresponds to a mutual better-response dynamic, which, in the event of convergence, would find a solution to the following simultaneous-move game:

$$\arg\min_{\boldsymbol{\theta}\in\mathbb{R}^n} -J(\boldsymbol{\theta}, \boldsymbol{\omega}) \qquad \arg\min_{\boldsymbol{\omega}\in\mathbb{R}^m} L(\boldsymbol{\omega}, \boldsymbol{\theta}) \tag{3}$$

However, simultaneous training dynamics are known to cycle [1].

BLPO

Partially inspired by [4], we recognize the fact that AC algorithms *should be* bilevel, and define the critic's loss function as a *parameterized* function of the actor's policy:

$$\min_{\boldsymbol{\theta}\in\mathbb{R}^n} \Phi(\boldsymbol{\theta}) \doteq -J(\boldsymbol{\theta}, \boldsymbol{\omega}^*(\boldsymbol{\theta})) \qquad \text{subject to } \boldsymbol{\omega}^*(\boldsymbol{\theta}) \in \arg\min_{\boldsymbol{\omega}\in\mathbb{R}^m} L_{\boldsymbol{\theta}}(\boldsymbol{\omega}) \tag{4}$$

Algorithm 1 BLPO with Nyström Hypergradients

for $k = 0, 1, \ldots, K_{\boldsymbol{\theta}} - 1$ **do**
  for $d = 0, 1, \ldots, K_{\boldsymbol{\omega}} - 1$ **do**
    $\boldsymbol{\omega}^{(d+1)} \leftarrow \boldsymbol{\omega}^{(d)} - \eta_{\boldsymbol{\omega}} \nabla_{\boldsymbol{\omega}} \hat{L}_{\boldsymbol{\theta}}(\boldsymbol{\omega}^{(d)})$ {Update critic}
  end for
  $\boldsymbol{\omega}^{(k)} \leftarrow \boldsymbol{\omega}^{(K_{\boldsymbol{\omega}})}$
  $\hat{\boldsymbol{v}}_{AC} \leftarrow (\nabla^2_{\boldsymbol{\omega}} \hat{L}_{\boldsymbol{\theta}}(\boldsymbol{\omega}^{(k)}))^{-1} \nabla_{\boldsymbol{\omega}} \hat{J}(\boldsymbol{\theta}^{(k)}, \boldsymbol{\omega}^{(k)})$ {Estimate the IHVP via the Nyström method}
  $\nabla_{\boldsymbol{\theta}} \hat{J}^{(k)} \leftarrow \nabla_{\boldsymbol{\theta}} \hat{J}(\boldsymbol{\theta}^{(k)}, \boldsymbol{\omega}^{(k)}) - \nabla_{\boldsymbol{\theta\omega}} \hat{L}(\boldsymbol{\theta}^{(k)}, \boldsymbol{\omega}^{(k)}) \hat{\boldsymbol{v}}_{AC}$ {Calculate hypergradient}
  $\boldsymbol{\theta}^{(k+1)} \leftarrow \boldsymbol{\theta}^{(k)} + \eta_{\boldsymbol{\theta}} \nabla_{\boldsymbol{\theta}} \hat{J}^{(k)}$ {Update actor}
end for

References

[1] T. Fiez, B. Chasnov, and L. Ratliff. Implicit Learning Dynamics in Stackelberg Games: Equilibria Characterization, Convergence Analysis, and Empirical Study. In *Proceedings of the 37th International Conference on Machine Learning*, pages 3133–3144. PMLR, Nov. 2020. ISSN: 2640-3498.

[2] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pages 1861–1870. PMLR, 2018.

[3] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *CoRR*, abs/1707.06347, 2017.

[4] L. Zheng, T. Fiez, Z. Alumbaugh, B. Chasnov, and L. J. Ratliff. Stackelberg actor-critic: Game-theoretic reinforcement learning algorithms. In *Proceedings of the AAAI conference on artificial intelligence*, volume 36, pages 9217–9224, 2022.