Bilevel Reinforcement Learning via the Development of Hyper-gradient without Lower-Level Convexity

Yan Yang [†§] Bin Gao [†] Ya-xiang Yuan [†]

[†] Academy of Mathematics and Systems Science, CAS
[§] University of Chinese Academy of Sciences

Bilevel reinforcement learning

Parameterized MDP

- $\mathcal{M}_\tau(x) = (\mathcal{S}, \mathcal{A}, P, r(x), \gamma, \tau)$
- Soft value functions

$$V_s^\pi(x) = \mathbb{E}[\Sigma_{t=0}^\infty \gamma^t (r_{s_t a_t}(x) + \tau h(\pi_{s_t})) \mid s_0 = s, \pi, \mathcal{M}_\tau(x)]$$

$$Q_{sa}^\pi(x) = r_{sa}(x) + \gamma \mathbb{E}_{s' \sim P(\cdot|s,a)}[V_{s'}^\pi(x)]$$

with h as the entropy function $h(\pi_s) = -\Sigma_a \pi_{sa} \log \pi_{sa}$

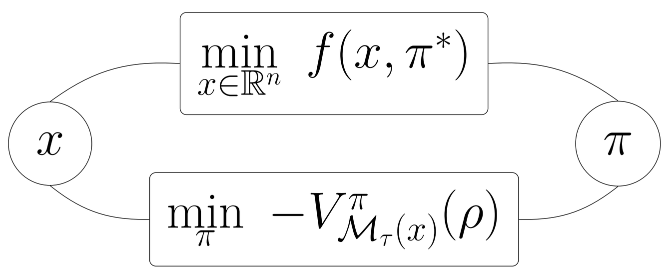
- The objective $V_{\mathcal{M}_\tau(x)}^\pi(\rho) = \mathbb{E}_{s\sim\rho}[V_s^\pi(x)]$, where ρ is the initial state distribution

Problem formulation of bilevel RL (BiRL)

$$\min_{x\in\mathbb{R}^n} \phi(x) := f(x, \pi^*(x))$$
$$\text{s. t. } \pi^*(x) = \arg\min_\pi -V_{\mathcal{M}_\tau(x)}^\pi(\rho)$$

- optimize the decision x based on the response policy π
- solve a RL problem in $\mathcal{M}_\tau(x)$ parameterized by x

The nested structure couples the upper and lower levels



Applications

- RL from human feedback (RLHF) [Christiano et al., 2017]

$$\min_x \mathbb{E}_{y,d_1,d_2\sim\rho(d;\pi^*(x))}[l_h(d_1, d_2, y; x)]$$
$$\text{s. t. } \pi^*(x) = \arg\min_\pi -V_{\mathcal{M}_\tau(x)}^\pi(\rho)$$

  - Trajectory sampled via $\pi^*(x)$: $d_i = \{(s_h^i, a_h^i)\}_{h=0}^{H-1}$ $(i = 1, 2)$
  - Preference for d_1 over d_2: $y \in \{0, 1\}$
- Reward shaping [Hu et al., 2020]

$$\min_x -V_{\mathcal{M}_\tau}^{\pi^*(x)}(\bar\rho)$$
$$\text{s. t. } \pi^*(x) = \arg\min_\pi -V_{\mathcal{M}_\tau(x)}^\pi(\rho)$$

- Apprenticeship learning [Arora and Doshi, 2021]
- Contract design [Wu et al., 2024]
- Robot navigation [Chakraborty et al., 2024]

Revisit bilevel optimization

General form

$$\min_{x\in\mathbb{R}^{d_x}} \phi(x) := f(x, y^*(x))$$
$$\text{s. t. } y^*(x) = \arg\min_{y\in\mathbb{R}^{d_y}} g(x, y)$$

- $f, g: \mathbb{R}^{d_x} \times \mathbb{R}^{d_y} \longrightarrow \mathbb{R}$, differentiable

Approximate implicit differentiation (AID)

- Hyper-gradient computation

$$\nabla\phi(x) = \nabla_x f(x, y^*(x)) + \nabla_x y^*(x)^\top \nabla_y f(x, y^*(x))$$

- Scenario of interest: $g(x, \cdot)$ is **strongly convex**

$$\nabla_x y^*(x) = \left[\nabla_{yy}^2 g(x, y^*)\right]^{-1} \nabla_{xy}^2 g(x, y^*)^\top$$

- Vanilla update rule

$$\mathbf{1}\times: x^+ = x - \beta\hat\nabla\phi(x)$$
$$\mathbf{N}\times: y^+ = y - \alpha\nabla_y g(x, y)$$

Motivation and challenge

Lower-level objective

$$V_{\mathcal{M}_\tau(x)}^\pi(\rho) = \mathbb{E}[\Sigma_{t=0}^\infty \gamma^t (r_{s_t a_t}(x) + \tau h(\pi_{s_t})) \mid \rho, \pi, \mathcal{M}_\tau(x)]$$

Main difficulties

- The lower-level problem of BiRL

$$\min_\pi -V_{\mathcal{M}_\tau(x)}^\pi(\rho)$$

is inherently **non-convex** [Agarwal et al., 2020; Lan, 2023 MP]
- Only the **non-uniform PL** condition holds [Mei et al., 2020]

Does hyper-gradient $\nabla\phi(x)$ exist?

The fixed-point equation

Softmax temporal value consistency

$$\pi_{sa}^*(x) = \exp\left(\tau^{-1}\left(r_{sa}(x) + \gamma\Sigma_{s'} P_{sas'} V_{s'}^*(x) - V_s^*(x)\right)\right)$$

$$V_s^*(x) = \tau \log\left(\Sigma_a \exp\left(\frac{r_{sa}(x) + \gamma\Sigma_{s'} P_{sas'} V_{s'}^*(x)}{\tau}\right)\right)$$

Take the derivative

$$\nabla\pi^*(x) = \tau^{-1}\operatorname{diag}(\pi^*(x))\left(\nabla r(x) - U\nabla V^*(x)\right)$$

Investigating $\nabla V^*(x)$

Consider the parameterized mapping

$$\varphi(x, v) = \tau \log\left(\exp\left(\frac{r(x) + \gamma P v}{\tau}\right) \cdot \mathbf{1}\right)$$

- Take the partial derivative

$$\frac{\partial\varphi_s(x, v)}{\partial v_{s'}} = \gamma\frac{\Sigma_a P_{sas'} \exp\left(\tau^{-1}\left(r_{sa}(x) + \gamma\Sigma_{s''} P_{sas''} v_{s''}\right)\right)}{\Sigma_a \exp\left(\tau^{-1}\left(r_{sa}(x) + \gamma\Sigma_{s''} P_{sas''} v_{s''}\right)\right)}$$

$\|\nabla_v\varphi(x, v)\|_\infty < 1 \Longrightarrow$ unique solution $V^*(x)$

- Differentiate both sides of $\varphi(x, V^*(x)) = V^*(x)$

$$\nabla V^*(x) = (I - \nabla_v\varphi(x, V^*(x)))^{-1}\nabla_x\varphi(x, V^*(x))$$

An observation in MDP: $\nabla_v\varphi(x, V^*(x)) = \gamma P^{\pi^*(x)}$

A model-based algorithm

Model-based hyper-gradient

$$\begin{aligned}\nabla\phi(x) &= \nabla_x f(x, \pi^*(x)) + \nabla\pi^*(x)^\top \nabla_\pi f(x, \pi^*(x)) \\ &= \nabla_x f(x, \pi^*(x)) + \tau^{-1}\nabla r(x)^\top \operatorname{diag}(\pi^*(x))\nabla_\pi f(x, \pi^*(x)) \\ &\quad - \tau^{-1}\nabla_x\varphi(x, V^*(x))^\top \left(I - \gamma P^{\pi^*(x)}\right)^{-\top} \\ &\quad \times U^\top \operatorname{diag}(\pi^*(x))\nabla_\pi f(x, \pi^*(x))\end{aligned}$$

Computing $\nabla\phi(x)$ is expensive!

Inexact strategy

- $V_k^* := V^*(x_k) \longleftarrow V_k \quad Q_k^* := Q^*(x_k) \longleftarrow Q_k$
  $\pi_k^* := \pi^*(x_k) \longleftarrow \pi_k$
- $A_k^{-1} b_k \longleftarrow w_k$ with
  $A_k = (I - \gamma P^{\pi_k})^\top$, $b_k = U^\top \operatorname{diag}(\pi_k)\nabla_\pi f(x_k, \pi_k)$

M-SoBiRL

$$\begin{aligned}\mathbf{1}\times&: w_k = w_{k-1} - \xi\left(\left(A_k^\top A_k\right) w_{k-1} - A_k^\top b_k\right) \\ \mathbf{1}\times&: V_k(s) = \tau\log\left(\Sigma_a \exp\left(Q_k(s, a)/\tau\right)\right) \\ \mathbf{1}\times&: x_{k+1} = x_k - \beta\tilde\nabla\phi(x_k, \pi_k, V_k, w_k) \\ \mathbf{N}\times&: Q_{k,n} = \mathcal{T}_{\mathcal{M}_\tau(x_{k+1})}(Q_{k,n-1}) \\ \mathbf{1}\times&: \pi_{k+1} = \mathtt{softmax}(Q_{k+1}/\tau)\end{aligned}$$

Main bottleneck

- ☹ Relying explicitly on the **black-box** P
- ☹ Involving large-scale **matrix multiplications**

Model-free bilevel RL

Problem formulation

$$\min_{x\in\mathbb{R}^n} \phi(x) := f(x, \pi^*(x))$$
$$\text{s. t. } \pi^*(x) = \arg\min_\pi -V_{\mathcal{M}_\tau(x)}^\pi(\rho)$$

with $f(x, \pi) = \mathbb{E}_{d_i\sim\rho(d;\pi)}[l(d_1, d_2, ..., d_I; x)]$

Absorb P into an expectation

$$\begin{aligned}\nabla V_s^*(x) &= (I - \nabla_v\varphi(x, V^*(x)))_s^{-1}\nabla_x\varphi(x, V^*(x)) \\ &= \sum_{t=0}^\infty \gamma^t \sum_{s'} P(s_t = s' | s_0 = s, \pi^*(x)) \sum_a \nabla r_{s'a}(x)\pi_{s'a}^*(x) \\ &= \mathbb{E}\left[\sum_{t=0}^\infty \gamma^t \nabla_x r_{s_t a_t}(x) \mid s_0 = s, \pi^*(x), \mathcal{M}_\tau(x)\right]\end{aligned}$$

Log probability trick

$$\begin{aligned}&\sum_{(d_1,\ldots,d_I)} l(d_1, d_2, ..., d_I; x)\nabla\Pi_{i=1}^I P(d_i; \pi^*(x)) \\ &= \tau^{-1}\mathbb{E}_{d_i\sim\rho(d;\pi^*(x))}\left[l(d_1, d_2, ..., d_I; x)\left(\sum_i\sum_h \nabla\left(Q_{s_h^i a_h^i}^*(x) - V_{s_h^i a_h^i}^*(x)\right)\right)\right]\end{aligned}$$

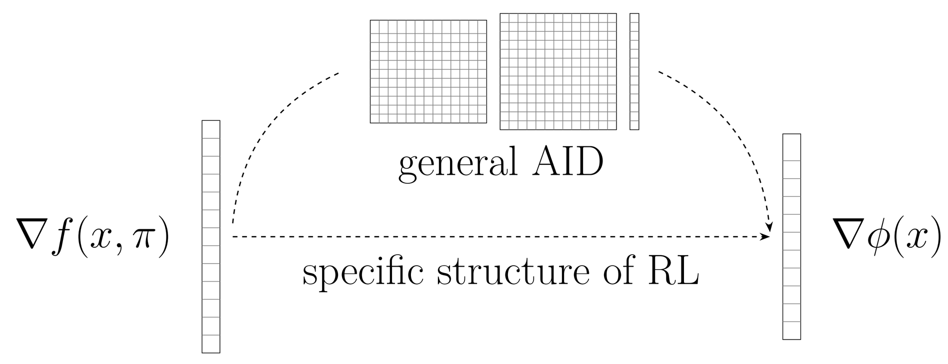
The model-free hyper-gradient

Hyper-objective

$$\begin{aligned}\phi(x) &= \mathbb{E}_{d_i\sim\rho(d;\pi^*(x))}[l(d_1, d_2, ..., d_I; x)] \\ &= \sum_{(d_1,\ldots,d_I)} l(d_1, d_2, ..., d_I; x)\,\Pi_{i=1}^I P(d_i; \pi^*(x))\end{aligned}$$

Estimate via sampling first-order information

$$\begin{aligned}\nabla\phi(x) = &\,\mathbb{E}_{d_i}[\nabla l(d_1, d_2, ..., d_I; x)] \\ &+ \tau^{-1}\mathbb{E}_{d_i}\left[l(d_1, d_2, ..., d_I; x)\left(\sum_i\sum_h \nabla\left(Q_{s_h^i a_h^i}^*(x) - V_{s_h^i}^*(x)\right)\right)\right]\end{aligned}$$



Model-free algorithms

Deterministic Soft BiRL algorithm

Algorithm SoBiRL

Input: iterations K, step size β, accuracy ϵ

1: **for** $k = 1, \ldots, K$ **do**
2: Solve the lower-level problem approximately with initial guess π_{k-1} such that $\|\pi_k - \pi^*(x_k)\|_2^2 \le \epsilon$
3: Compute the approximate hyper-gradient $\tilde\nabla\phi(x_k, \pi_k)$
4: Implement an inexact hyper-gradient descent step $x_{k+1} = x_k - \beta\tilde\nabla\phi(x_k, \pi_k)$
5: **end for**

Output: (x_{K+1}, π_{K+1})

Stochastic extension (Stoc-SoBiRL)

- Characterize the bias and variance
- Replace $\tilde\nabla\phi$ with its stochastic counterpart

$$\begin{aligned}\bar\nabla\phi(\mathbf{D}_k, \xi_k, \zeta_\mathbf{k}; x_k, \pi_k) = &\frac{1}{M}\sum_{m=1}^M \nabla l(\mathbf{d}_k^m; x_k) \\ &+ \frac{1}{\tau M}\sum_{m=1}^M l(\mathbf{d}_k^m; x_k)\sum_i\sum_h \bar\nabla\left(Q_{s_h^{m,i} a_h^{m,i}}^{\xi_k} - V_{s_h^{m,i}}^{\zeta_\mathbf{k}}\right)(x_k, \pi_k)\end{aligned}$$

- Maintain a momentum-instructed h_k for acceleration

$$h_k = \bar\nabla\phi_k + (1 - \mu_k)\left(h_{k-1} - \bar\nabla\phi(\mathbf{D_k}, \boldsymbol{\xi}_\mathbf{k}, \boldsymbol{\zeta}_\mathbf{k}; \mathbf{x_{k-1}}, \boldsymbol{\pi}_\mathbf{k-1})\right)$$

- The upper-level update rule: $x_{k+1} = x_k - \beta_k h_k$

Convergence results

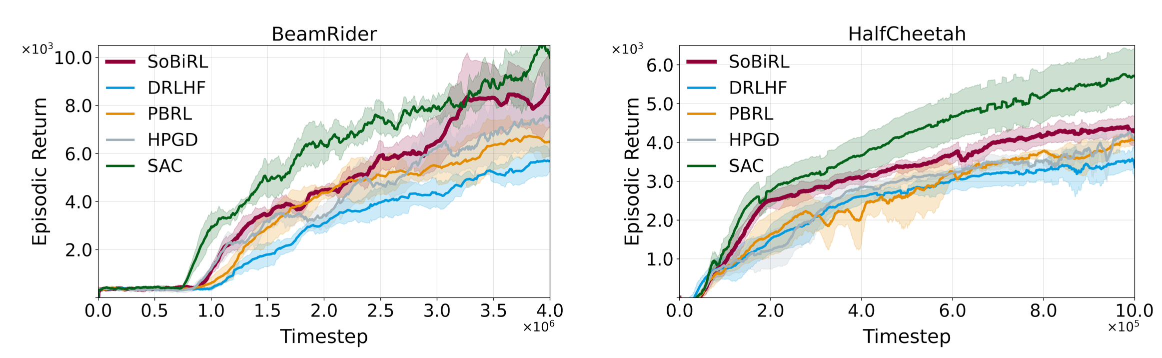
Related work

- An AID-based bilevel RL framework, PARL [Chakraborty et al., ICLR 2024]
- A penalty-based method, PBRL [Shen et al., ICML 2024]
- A stochastic bilevel RL method, HPGD [Thoma et al., NeurIPS 2024]

Comparison among biRL methods

Algorithm	Deter. or Stoch.	Conv. Rate	Inner Iter.	Oracle
PARL	Deter.	$\mathcal{O}(\epsilon^{-1})$	$\mathcal{O}(\log\epsilon^{-1})$	1st+2nd
PBRL	Deter.	$\mathcal{O}(\lambda\epsilon^{-1})$	$\mathcal{O}(\log\lambda^2\epsilon^{-1})$	1st
HPGD	Stoch.	$\mathcal{O}(\epsilon^{-2})$	$\mathcal{O}(\log\epsilon^{-1})$	1st
M-SoBiRL	Deter.	$\mathcal{O}(\epsilon^{-1})$	$\mathcal{O}(1)$	1st
SoBiRL	Deter.	$\mathcal{O}(\epsilon^{-1})$	$\mathcal{O}(\log\epsilon^{-1})$	1st
Stoc-SoBiRL	Stoch.	$\tilde{\mathcal{O}}(\epsilon^{-1.5})$	$\mathcal{O}(\log\epsilon^{-1})$	1st

Numerical results



- SoBiRL is comparable to the baseline SAC
- SoBiRL obtains higher returns than other bilevel methods
- The accuracy of preference prediction of DRLHF achieves 85%, while it hovers around 54% for SoBiRL

☺ $\nabla\phi(x)$ **integrates exploitation and exploration!**

More experiments are referred to our paper...