非线性优化中 Lagrange 方法在安全强化学习中的应用

西安交通大学, 人工智能学院
吴天阳 [a], 郭涵伟 [b]
4124136039 [a], *3124136019* [b]

1 背景介绍

1.1 数学记号

定义 1 (Markov Decision Process, MDP). 设 $\mathcal{M} = \{\mathcal{S}, \mathcal{A}, \mathbb{P}, r, \mu, \gamma\}$，其中

- $\mathcal{S}$ 为有限状态集合,
- $\mathcal{A}$ 为有限动作集合,
- $\mathbb{P}(s'|s,a) : \mathcal{S}^2 \times \mathcal{A} \to [0,1]$ 为状态转移概率分布,
- $\mu(s) : \mathcal{S} \to [0,1]$ 为初始状态分布,
- $r(s) : \mathcal{S} \to \mathbb{R}$ 为奖励函数,
- $\gamma \in (0,1)$ 为折扣系数.

将 $\mathcal{M}$ 称为 *Markov* 决策过程，简称为 *Markov* 过程.

设 $\pi(a|s) : \mathcal{A} \times \mathcal{S} \to [0,1]$ 为参数化策略函数，表示在状态 s 下动作 a 执行的概率大小. 在深度强化学习中，我们通常会使用深度神经网络近似策略函数 π，因此通常也记为 π_θ 表示参数化的策略函数.

下面我们分别给出强化学习 (Reinforcement Learning, RL) 和安全强化学习 (Safe Reinforcement Learning, Safe RL) 中优化目标.

强化学习优化目标： 设 $\tau := (s_0, a_0, s_1, \cdots)$ 表示一段轨迹，$\tau \sim \pi$ 表示基于策略 π 采样得到的 τ，满足 $s_0 \sim \mu, a_t \sim \pi(\cdot|s_{t-1}), s_{t+1} \sim \mathbb{P}(\cdot|s_t, a_t)$. 记 $R(\tau) = \sum_{t=0}^{\infty} \gamma^t r(s_t)$ 表示折后回报，则强化学习的优化目标为最大化折后回报，即

$$\max_{\pi} \mathcal{J}^R(\pi) := \mathbb{E}_{\tau\sim\pi}[R(\tau)] = \mathbb{E}_{\tau\sim\pi}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t)\right] \tag{1.1}$$

安全强化学习优化目标： 设 $C_1, \cdots, C_m : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ 为 m 个成本函数 (Cost function)，$d_1, \cdots, d_m \in \mathbb{R}$ 为 m 个成本限制 (Cost limit)，成本函数的折后回报为

$$\mathcal{J}^{C_i}(\pi) = \mathbb{E}_{\tau\sim\pi}\left[\sum_{t=0}^{\infty} \gamma^t C_i(s_t, a_t, s_{t+1})\right] \tag{1.2}$$

则安全强化学习的优化目标为

$$\begin{aligned} &\max_{\pi} \mathcal{J}^R(\pi) \\ &s.t. \quad \mathcal{J}^{C_i}(\pi) \leqslant d_i, \quad (i = 1, \cdots, m) \end{aligned} \tag{1.3}$$

1.2 Lagrange 对偶

考虑如下具有一般性的最优化问题，也称原始问题 (Primal problem)

$$\begin{aligned} &\min_x f(x) \\ s.t. \quad &h_i(x) \leqslant 0, \quad (i=1,\cdots,m), \\ &l_j(x) = 0, \quad (j=1,\cdots,r). \end{aligned} \tag{1.4}$$

定义其对应的 Lagrange 对偶形式为

$$\mathcal{L}(x,\boldsymbol{u},\boldsymbol{v}) = f(x) + \sum_{i=1}^{m} u_i h_i(x) + \sum_{j=1}^{r} v_j l_j(x) \tag{1.5}$$

其中 $\boldsymbol{u} \in \mathbb{R}^m, \boldsymbol{v} \in \mathbb{R}^r$ 中的每个维度上的分量被成为 Lagrange 乘子.

引理 1. *对于任何满足 (1.4) 中约束的 x，有 $f(x) = \max\limits_{u_i \geqslant 0, v_j} \mathcal{L}(x,\boldsymbol{u},\boldsymbol{v})$，并且右式取到最大值，当且仅当，$u_i h_i(x) = 0, (i=1,\cdots,m)$.*

证明. $\forall x$ 满足式 (1.4) 中约束条件，$\forall u_i \geqslant 0$，有 $\mathcal{L}(x,\boldsymbol{u},\boldsymbol{v}) = f(x) + \sum\limits_{i=1}^{m} u_i h_i(x) \leqslant f(x)$. 当且仅当，$u_i h_i(x) = 0, (i=1,\cdots,m)$ 时，$\mathcal{L}(x,\boldsymbol{u},\boldsymbol{v})$ 取到最大值. □

引理 2. *设 f^* 为原始问题最优解，则 $f^* = \min\limits_x \max\limits_{u_i \geqslant 0, v_j} \mathcal{L}(x,\boldsymbol{u},\boldsymbol{v})$.*

证明. 由引理 (1) 可知，只需证 $\min\limits_x$ 中取到的 x 满足式 (1.4) 中的约束条件. 假设存在 x 不属于可行域中，即存在 $h_{i_0}(x_0) > 0$ 或 $l_{j_0}(x_0) \neq 0$，则当 $u_{i_0} \to \infty$ 或 $v_{j_0} h_{j_0}(x_0) \to \infty$ 时，$\max\limits_{u_i \geqslant 0, v_j} \mathcal{L}(x,\boldsymbol{u},\boldsymbol{v}) \to \infty$，与 f^* 存在矛盾，故原命题成立. □

定义 2 (对偶问题). *设 $\theta_d(\boldsymbol{u},\boldsymbol{v}) = \min\limits_x \mathcal{L}(x,\boldsymbol{u},\boldsymbol{v})$，则原始问题的对偶问题为*

$$g^* = \max_{u_i \geqslant 0, v_j} \theta_d(\boldsymbol{u},\boldsymbol{v}) = \max_{u_i \geqslant 0, v_j} \min_x \mathcal{L}(x,\boldsymbol{u},\boldsymbol{v}) \tag{1.6}$$

注意到 $\theta_d(\boldsymbol{u},\boldsymbol{v})$ 是关于 $\boldsymbol{u},\boldsymbol{v}$ 的仿射函数且为逐点下确界（由 $\mathcal{L}$ 定义不难看出），因此 θ_d 为凹函数，故求解 g^* 属于凸优化问题.

命题 1 (弱对偶性). *上述定义的 f^*, g^* 满足弱对偶性 $g^* \leqslant f^*$.*

证明. 设 $x^* \in \mathbb{R}$ 为 f^* 取到时对应的值，$\boldsymbol{u}^* \in \mathbb{R}^m, \boldsymbol{v}^* \in \mathbb{R}^r$ 为 g^* 取到时对应的值，则

$$\begin{aligned} g^* &= \max_{u_i \geqslant 0, v_j} \min_x \mathcal{L}(x,\boldsymbol{u},\boldsymbol{v}) = \min_x \mathcal{L}(x,\boldsymbol{u}^*,\boldsymbol{v}^*) \leqslant \mathcal{L}(x^*,\boldsymbol{u}^*,\boldsymbol{v}^*) \\ &\leqslant \max_{u_i \geqslant 0, v_j} \mathcal{L}(x^*,\boldsymbol{u},\boldsymbol{v}) = \min_x \max_{u_i \geqslant 0, v_j} \mathcal{L}(x,\boldsymbol{u},\boldsymbol{v}) = f^* \end{aligned} \tag{1.7}$$

□

Lagrange 对偶问题转换通常是消去约束条件的方法，一般情况下我们不会讨论 $g^* = f^*$ 的情况，而通过梯度下降的方法求解 x，并在迭代过程中，如果 x 不属于可行域中，则将系数 $\boldsymbol{u}$ 进行放大，从而再利用梯度下降对 x 进行更新，最终将 x 限制到可行域中.

1.3 Lagrange 方法

我们将上述的 Lagrange 对偶方法与安全强化学习结合，对安全强化学习优化目标 (1.3) 转化为 Lagrange 对偶问题

$$\min_{\lambda_i \geqslant 0} \max_{\pi} \left[\mathcal{J}^R(\pi) - \lambda_i(\mathcal{J}^{C_i}(\pi) - d_i)\right], \quad (i = 1, \cdots, m) \tag{1.8}$$

该问题为无约束的强化学习问题，因此可以使用任何强化学习算法解决，通常强化学习算法会基于当前与环境的交互，估计得到状态对应的折后回报期望，从而优化 π 使其最大化折后回报，因此我们只需要将 $-\lambda_i(\mathcal{J}^{C_i}(\pi) - d_i)$ 项加入到之前的折后回报 $\mathcal{J}^R(\pi)$ 中，使用任何强化学习算法对 π 进行更新，若新的 π 不满足约束条件，则增大 λ_i.

以一个约束条件 C 为例，记第 t 次迭代的成本误差为 $e_t = \mathcal{J}^C(\pi_t) - d$，下面给出一种最简单的调整 λ 的方法

$$\lambda_{t+1} = \max\left(\lambda_t + \eta e_t, 0\right) \tag{1.9}$$

其中 η_i 为 λ_i 对应的学习率.

这里的 Lagrange 乘子更新策略可以更加复杂，例如使用 PID(Proportion Integration Differentiation) 控制算法：

$$\begin{aligned} \lambda_{t+1} &= \lambda_t + K_p e(t) + K_i \int e(t)\mathrm{d}t + K_d \frac{\mathrm{d}e(t)}{\mathrm{d}t}, && \text{（连续形式）} \\ \lambda_{t+1} &= \lambda_t + K_p e(t) + K_i \sum_{n=0}^{t} e(t) + K_d \left(\mathcal{J}^C(\pi_t) - \mathcal{J}^C(\pi_{t-1})\right). && \text{（离散形式）} \end{aligned} \tag{1.10}$$

具体实现中通常会用指数平滑 (Exponential Moving Average, EMA) 代替 $e(t)$ 和 $\mathcal{J}^C(\pi_t)$.

2 实验步骤与结果分析

Safety Gymnasium是在MuJoCo（机器人仿真环境）上加入成本函数的可视化，我们仅考虑其中两个包含速度限制的环境

环境名称	速度阈值	状态维度	动作维度
`SafetyAntVelocity-v1`	2.6222	27	8
`SafetyHumanoidVelocity-v1`	1.4149	376	17

表 1: 使用的两个环境的参数

安全强化学习算法我们分别考虑了两个在线与离线的经典算法 PPO 和 SAC，分别使用简单比例控制和 PID 控制[1]对 Lagrange 乘子进行调整，分别称为 PPOLag, SACLag, CPPOPID 和 SACPID.

[1]参考论文：Responsive Safety in Reinforcement Learning by PID Lagrangian Methods

环境名称	PPO	PPOLag	CPPOPID	SAC	SACLag	SACPID
Ant-Ret	5977.73 ±885.65	3261.87 ±80.00	3213.36 ±146.78	5456.31 ±156.04	1897.32 ±1213.74	1940.55 ±482.41
Ant-Cost	958.13 ±134.5	12.05 ±6.57	14.30 ±7.39	943.10 ±47.51	5.73 ±7.83	13.73 ±7.24
Humanoid-Ret	9115.93 ±596.88	6624.46 ±25.9	6579.26 ±55.70	6039.77 ±167.82	5940.04 ±121.93	6107.36 ±113.24
Humanoid-Cost	960.44 ±7.06	5.87 ±9.46	3.76 ±3.61	41.42 ±49.78	17.59 ±6.24	6.20 ±10.14

表 2: 四种不同算法在两个环境上得到的总回报与总成本，PPO 算法训练 10^7 步骤，SAC 算法训练 3×10^6，速度成本阈值均为 25.

3 结论与讨论