文章编号: 1001-0920 (2015) 12-2161-07 DOI: 10.13195/j.kzyjc.2014.1489

基于对称扰动采样的 Actor-critic 算法

张春元[1,2], 朱清新[1]

(1. 电子科技大学 计算机科学与工程学院，成都 611731；2. 海南大学 信息科学技术学院，海口 570228)

摘 要: 针对传统 Actor-critic (AC) 方法在求解连续空间序贯决策问题时收敛速度较慢、收敛质量不高的问题, 提出一种基于对称扰动采样的 AC 算法框架. 首先, 框架采用高斯分布作为策略分布, 在每一时间步对当前动作均值对称扰动, 从而生成两个动作与环境并行交互; 然后, 基于两者的最大时域差分 (TD) 误差选取 Agent 的行为动作, 并对值函数参数进行更新; 最后, 基于两者的平均常规梯度或增量自然梯度对策略参数进行更新. 理论分析和仿真结果表明, 所提框架具有较好的收敛性和计算效率.

关键词: Actor-critic 方法；对称扰动采样；连续空间；强化学习

中图分类号: TP18 **文献标志码:** A

Actor-critic algorithms based on symmetric perturbation sampling

ZHANG Chun-yuan[1,2], *ZHU Qing-xin*[1]

(1. School of Computer Science and Engineering，University of Electronic Science and Technology of China, Chengdu 611731，China；2. College of Information Science and Technology，Hainan University，Haikou 570228, China．Correspondent：ZHANG Chun-yuan，E-mail：zhangcy@hainu.edu.cn)

Abstract: When solving the sequential decision-making problems in continuous spaces, the traditional actor-critic(AC) methods are often difficult to get good convergence speed and quality. To overcome the above weakness, an AC algorithm framework, which uses a Gaussian distribution as the policy distribution, is proposed based on the symmetric perturbation sampling. At each time step, the framework generates two actions through two symmetric perturbations on the current action mean, and takes them to interact with the environment in parallel. Then, the framework selects the Agent's behavior action and updates the value-function parameters based on the maximum temporal difference(TD) error, and updates the policy parameters based on the average regular gradient or the average incremental natural gradient. The theoretical analysis and simulation results show that the framework not only has a better convergence performance, but also has a high computational efficiency.

Keywords: Actor-critic method；symmetric perturbation sampling；continuous space；reinforcement learning

0 引 言

值函数逼近、策略搜索和 Actor-critic (AC) 方法作为求解连续状态与动作空间序贯决策问题 (简称连续空间问题) 的主要手段[1], 近年来一直是强化学习领域的研究热点. 其中, AC 方法可以看作是值函数逼近与策略搜索方法的混合[1], AC 算法框架由 Actor 和 Critic 两部分组成[2], 前者可以通过策略搜索方法进行策略改进, 后者可以通过值函数逼近方法进行策略评估. 这种独特的结构使得 AC 方法可以充分吸收上述两类方法的优点, 获得两者所不具备的能力. 相较值函数逼近方法, AC 方法不但可采用 Gibbs (或称为 Boltzmann) 分布[3]求解离散动作空间问题, 而且可采用高斯分布求解连续动作空间问题, 其应用范围更为广泛; 相较策略搜索方法, AC 方法不但可通过策略梯度下降进行策略参数更新, 而且可通过值函数的引入减小策略梯度的方差, 其收敛速度更为迅速[3-4]. 因此, 针对连续空间问题, AC 方法具有更好的应用前景.

然而, 在求解许多连续空间问题时, 传统 AC 方法的收敛速度和质量却差于值函数逼近方法. 造成这一现象的主要原因是: 尽管值函数逼近方法对连续动作空间的离散化处理降低了最 (次) 优动作的精度, 然而每一时间步基于 ε-greedy 或 Gibbs 策略选取行为动

收稿日期: 2014-09-27；**修回日期:** 2015-01-07.

基金项目: 国家自然科学基金项目(61100118, 60671033)；海南省自然科学基金项目(613153).

作者简介: 张春元(1973−), 男, 副教授, 博士生, 从事强化学习、信息检索等研究；朱清新(1954−), 男, 教授, 博士生导师, 从事计算运算学、生物信息学等研究.

作, 较好地解决了探索与利用的平衡问题. 而传统 AC 方法一般采用参数化高斯分布来表示连续策略[4-7], 每一时间步仅基于高斯噪声扰动随机生成一个动作进行学习, 从而实现值函数和策略参数的更新, 这种单动作采样方式虽然简单易用, 但在学习过程中存在重探索轻利用、策略参数反复修改、策略梯度估计偏差较大等问题, 从而影响了算法的收敛性能.

对于连续空间问题, 其状态与动作数量具有无穷多个, 因此如何提高探索效率并对现有知识加以利用对传统 AC 方法收敛性能的改善显得尤为重要. 近年来, 有两种非 AC 类型强化学习算法在求解这类问题时的良好表现引起了作者的注意: 一种是 Pazis 等[8]提出的折半动作查找算法, 该算法在每一时间步通过多次折半动作采样对连续动作空间进行探索, 有效地减少了探索的盲目性; 另一种是 Sehnke 等[9]提出的基于对称采样的策略梯度参数探索算法, 该算法每一 episode 通过对称扰动采样并行生成两条模拟轨迹, 解决了奖励基线误导问题, 有效地减小了策略梯度的估计方差. 实验结果表明, 两种算法均可显著提高原有算法的收敛速度和质量.

受上述两种算法的启发, 针对传统 AC 方法在求解连续空间问题时的收敛性能有限这一问题, 本文在文献 [3] 和文献 [4] 的基础上提出一种基于对称扰动采样的 AC 算法框架, 并从理论上分析框架的复杂度和收敛性, 通过小车爬山和倒立摆起摆仿真实验对框架集成的 4 种 AC 算法的有效性进行验证. 实验结果表明, 所提框架具有线性的时间与空间复杂度, 以及较好的收敛性能, 在有些情况下, 其资格迹采用替代迹, 较累加迹可以获得更好的收敛性能.

1 相关理论

1.1 Markov 决策过程

在强化学习中, 多维连续状态空间 $\mathcal{S}$ 和一维连续空间 $\mathcal{A}$ 序贯决策问题通常会形成将定义转化为一个离散时间 Markov 决策的过程 (MDP)[10]: 对于每一时间步 $t \in N$, Agent 观察状态 $s_t \in \mathcal{S}$, 根据随机策略 $\pi_\theta(a_t|s_t)$ 选取动作 $a_t \in \mathcal{A}$, 然后按概率 $P(s_{t+1}|s_t, a_t)$ 转移至下一状态 s_{t+1}, 并获得立即回报 $r_t \in R$. 其中: $\pi_\theta(a_t|s_t)$ 表示 a_t 在给定 s_t 和策略参数向量 $\theta \in R^M$ 时被选取的概率, 为了简化符号, 下文通过 π 表示 π_θ.

通过 Agent 与环境的反复交互, 强化学习的目标是: 学习一个最优策略参数向量 θ^*, 从而最大化目标函数 $J(\pi)$. 根据强化学习问题的性质, $J(\pi)$ 可采用两种形式进行定义[11]: 第 1 种为平均回报形式

$$J(\pi) = \lim_{n\to\infty} \frac{1}{n} \mathrm{E}\Big[\sum_{t=0}^{n-1} r_t|\pi\Big]. \tag{1}$$

第 2 种为折扣回报形式 (或称起始状态形式[11])

$$J(\pi) = \mathrm{E}\Big[\sum_{t=0}^{\infty} \gamma^t r_t|s_0, \pi\Big]. \tag{2}$$

其中: s_0 为给定起始状态, $\gamma \in [0, 1]$ 为折扣因子.

1.2 策略梯度定理

对于任意状态-动作对 (x, u), 假设 $\pi(u|x)$ 对 θ 连续可微, 则常规策略梯度定理[11]扩展到连续空间可表述为

$$\nabla J(\pi) = \int_{\mathcal{S}} d^\pi(x) \int_{\mathcal{A}} \nabla\pi(u|x) Q^\pi(x, u) \mathrm{d}u\mathrm{d}x, \tag{3}$$

其中: ∇ 表示 ∇_θ, 下文表示与此相同; 对于平均回报形式, $d^\pi(x) = \lim\limits_{t\to\infty} P(s_t = x|s_0, \pi)$ 为 x 在给定 π 下的平稳分布, 且与 s_0 无关; 对于折扣回报形式, $d^\pi(x) = \sum\limits_{t=0}^{\infty} \gamma^t P(s_t = x|s_0, \pi)$. 若对于平均回报形式, 令 $\gamma = 1$, $\bar{r}(\pi) = J(\pi)$, 对于折扣回报形式, 令 $\bar{r}(\pi) = 0$, 则式 (3) 中的动作值函数 $Q^\pi(x, u)$ 可统一定义为[4]

$$Q^\pi(x, u) = \mathrm{E}\Big[\sum_{t=0}^{\infty} \gamma^t r_t - \bar{r}(\pi)|s_0 = x, a_0 = u, \pi\Big]. \tag{4}$$

由于 $\int_{\mathcal{A}} \nabla\pi(u|x)\mathrm{d}u = 0$, 可在式 (3) 中引入基线函数 $b(x) \in R$ 来减小策略梯度方差而不改变策略梯度值. 由文献 [3] 中的引理 2 可知, $Q^\pi(x, u)$ 的最小二乘方差基线 $b^*(x)$ 等于状态值函数 $V^\pi(x)$, 式 (3) 引入 $V^\pi(x)$ 并采用似然比技巧, 可重写为期望形式

$$\nabla J(\pi) = \mathrm{E}_{x\sim d^\pi, u\sim\pi}[A^\pi(x, u)\nabla\log\pi(u|x)]. \tag{5}$$

其中: $A^\pi(x, u) = Q^\pi(x, u) - V^\pi(x)$ 称为优势函数, $\nabla\log\pi(u|x)$ 为 $A^\pi(x, u)$ 的兼容特征向量[3].

自然策略梯度 $\widetilde{\nabla} J(\pi)$ 可由常规策略梯度 $\nabla J(\pi)$ 作线性变换计算得到[3], 即

$$\widetilde{\nabla} J(\pi) = G(\theta)^{-1} \nabla J(\pi), \tag{6}$$

其中: $G(\theta)$ 为 Fisher 信息矩阵, 其定义为

$$G(\theta) = \mathrm{E}_{x\sim d^\pi, u\sim\pi}[\nabla\log\pi(u|x)\nabla\log\pi(u|x)^{\mathrm{T}}]. \tag{7}$$

1.3 传统 AC 方法

近年来, AC 方法发展很快, 各种新算法也相继提出[1-7,10], 本文只对基于策略梯度的无模型在线在策略连续空间 AC 方法加以回顾, 称之为传统 AC 方法.

定义时域差分误差

$$\delta_t = r_t - \bar{r}_t + \gamma\hat{V}(s_{t+1}) - \hat{V}(s_t), \tag{8}$$

其中 $\bar{r}_t$、$\hat{V}(s_{t+1})$ 和 $\hat{V}(s_t)$ 分别为在给定 π 下的 $\bar{r}(\pi)$、$V^\pi(s_{t+1})$ 和 $V^\pi(s_t)$ 在 t 时间步的无偏估计值.

对于连续状态空间, Critic 常采用线性或非线性 TD(λ) 方法对式 (8) 中的 $\hat{V}(s_{t+1})$ 和 $\hat{V}(s_t)$ 逼近. 由于线性 TD(λ) 方法具有可靠的收敛保证[12], 令 $\hat{V}(x) = v^{\mathrm{T}}\phi(x)$. 其中: $v \in R^N$ 为值函数的参数向量, $\phi(x) \in R^N$ 为 x 对应的值函数特征向量. 记 $\lambda \in [0, 1]$、e_t 和 $\alpha_{v,t}$ 分别为 Critic 的资格迹衰减参数、资格迹向量和

v_t 的学习率, 则由线性 TD(λ) 方法[12]可知, e_t 和 v_t 的更新规则表示为

$$e_{t+1} = \gamma\lambda e_t + \phi(s_t), \tag{9}$$

$$v_{t+1} = v_t + \alpha_{v,t}\delta_t e_{t+1}. \tag{10}$$

记 $\alpha_{r,t}$ 为 $\bar{r}_t$ 的学习率, 则 $\bar{r}_t$ 的更新规则为[4]

$$\bar{r}_{t+1} = \bar{r}_t + \alpha_{r,t}\delta_t. \tag{11}$$

定义策略参数资格迹向量 z_t 的更新规则为

$$z_{t+1} = \gamma\lambda z_t + \nabla\log\pi(a_t|s_t). \tag{12}$$

参照文献 [5] 附录 A.2, 采用向前/向后观点分析, 不难推出基于 λ 回报的常规策略梯度, 可定义为

$$\nabla J(\pi) = \mathrm{E}_{x\sim d^\pi, u\sim\pi}[\delta_t z_{t+1}|s_t = x, a_t = u]. \tag{13}$$

对于连续动作空间, Actor 常用 $\nabla J(\pi)$ 或 $\widetilde{\nabla} J(\pi)$ 对 θ 更新. 记 $\alpha_{\theta,t}$ 为 θ_t 的学习率, 由式 (13) 和 (6) 可知, 基于 $\nabla J(\pi)$ 和 $\widetilde{\nabla} J(\pi)$ 的 θ_t 更新规则分别为[4]

$$\theta_{t+1} = \theta_t + \alpha_{\theta,t}\delta_t z_{t+1}, \tag{14}$$

$$\theta_{t+1} = \theta_t + \alpha_{\theta,t}\delta_t G(\theta_t)^{-1} z_{t+1}. \tag{15}$$

其中: $G(\theta_t)^{-1}$ 计算费用高昂, 难以处理实时性强化学习任务. 为此, 在文献 [3] 的算法 3 中, 令 $\widetilde{\nabla} J(\pi) = w_{t+1}$ 对 θ_t 更新, w_{t+1} 为 $A^\pi(x,u)$ 在 t 时间步更新后的兼容权向量, 文献 [4] 称之为增量自然梯度. 记 $F_t = \nabla\log\pi(a_t|s_t)\nabla\log\pi(a_t|s_t)^{\mathrm{T}}$, 则式 (15) 可改写为

$$w_{t+1} = w_t - \alpha_{v,t}(F_t w_t - \delta_t z_{t+1}), \tag{16}$$

$$\theta_{t+1} = \theta_t + \alpha_{\theta,t} w_{t+1}. \tag{17}$$

2　对称扰动采样 AC 算法

2.1　高斯策略分布

对于连续动作空间问题, AC 方法习惯采用高斯分布来表示策略分布[4-7], 本文也将遵循这一习惯, 定义 $\pi(u|x)$ 为

$$\pi(u|x) = \frac{1}{\sqrt{2\pi}\sigma(x)}\exp\Big(-\frac{(u-\mu(x))^2}{2\sigma^2(x)}\Big). \tag{18}$$

其中: $\mu(x)$ 和 $\sigma(x)$ 为 x 对应的动作均值和标准差, 两者分别参数化定义为[4]

$$\mu(x) = {\theta_\mu}^{\mathrm{T}}\varphi_\mu(x), \tag{19}$$

$$\sigma(x) = \exp({\theta_\sigma}^{\mathrm{T}}\varphi_\sigma(x)). \tag{20}$$

其中: $\varphi(x) = (\varphi_\mu(x)^{\mathrm{T}}, \varphi_\sigma(x)^{\mathrm{T}})^{\mathrm{T}}$ 为策略特征向量, $\theta = ({\theta_\mu}^{\mathrm{T}}, {\theta_\sigma}^{\mathrm{T}})^{\mathrm{T}}$ 为策略参数向量. 相应地, $z_t = ({z_{\mu,t}}^{\mathrm{T}}, {z_{\sigma,t}}^{\mathrm{T}})^{\mathrm{T}}$, $w_t = ({w_{\mu,t}}^{\mathrm{T}}, {w_{\sigma,t}}^{\mathrm{T}})^{\mathrm{T}}$.

由上述定义可知, 兼容特征向量可计算如下:

$$\begin{cases} \nabla_{\theta_\mu}\log\pi(u|x) = \dfrac{(u-\mu(x))}{\sigma^2(x)}\varphi_\mu(x), \\ \nabla_{\theta_\sigma}\log\pi(u|x) = \dfrac{(u-\mu(x))^2-\sigma^2(x)}{\sigma^2(x)}\varphi_\sigma(x). \end{cases} \tag{21}$$

由式 (18) 可知, $\sigma(x)$ 表示策略的探索力度. 随着学习的进行, 如果算法收敛, 则探索力度将逐渐减弱, $\sigma(x)$ 将逐渐减小. 当 $\sigma(x) \to 0$ 时, $\nabla_{\theta_\mu}\log\pi(u|x) \to \infty$, $z \to \infty$, 这将会破坏算法的稳定性. 针对这一问题, Williams 建议采用 $\alpha\sigma^2(x)$ (α 为常数) 作为 θ 的学习率[13]. 顺应这一思路, 算法实现时, 固定 $\alpha_{\theta,t}$ 为常数, 直接按如下方式计算兼容特征向量:

$$\begin{cases} \nabla_{\theta_\mu}\log\pi(u|x) = (u-\mu(x))\varphi_\mu(x), \\ \nabla_{\theta_\sigma}\log\pi(u|x) = ((u-\mu(x))^2-\sigma^2(x))\varphi_\sigma(x). \end{cases} \tag{22}$$

记 $\eta \sim \mathcal{N}(0,1)$, 由式 (22) 可知, $\nabla_{\theta_\mu}\log\pi(u|x) = \eta\sigma(x)\varphi_\mu(x)$, $\nabla_{\theta_\sigma}\log\pi(u|x) = (\eta^2-1)\sigma^2(x)\varphi_\sigma(x)$. 由于 $\sigma(x)$ 和 $|\eta|$ 同样存在取值过大的可能, 仍会对算法的稳定性造成破坏. 为此, 本文提出一种经验性解决方案, 对 $\sigma(x)$ 和 η 的取值加以限定: 考虑到学习刚开始时, 策略的探索力度通常最大且 θ_0 一般初始化为零向量, 即 $\sigma(s_0) = 1$, 因此限定 $\sigma(x) \in [0,1]$; 当 $|\eta| > 3$ 时, 对 η 重新取值, 直至 $|\eta| \leqslant 3$.

2.2　单动作采样

传统 AC 方法普遍采用单动作采样方式学习: 在 t 时间步, Agent 由式 (18) 仅执行单一动作 $a_t = \mu(s_t) + \eta\sigma(s_t)$, 然后由式 (8) 计算 δ_t. 在此基础上, Critic 根据式 (9) $\sim$ (11) 分别对 e_t、v_t 和 $\bar{r}_t$ 进行更新, Actor 根据式 (12)、(14)、(22) 或 (16)、(17) 分别对 z_t 和 θ_t 进行更新. 当 $\eta > 0$ 时, 算法将对当前动作均值 $\mu(s_t)$ 实施正扰动, 反之实施负扰动. 当 $\lambda = 0$ 时, 如果 $\delta_t > 0$, 则 Actor 对当前扰动进行强化, $\mu(s_t)$ 朝当前扰动方向调整; 反之, 进行弱化, $\mu(s_t)$ 朝当前扰动相反方向调整.

单动作采样方式虽然简单易用, 但随之也带来了许多问题: 1) 每一时间步仅基于高斯噪声扰动 $\eta\sigma(x)$ 生成一个动作, η 取值的随机性使得算法无法确定扰动方向, 显然这并不是一种富有效率的探索方式, 也不利于值函数参数的快速收敛, 进而使得 TD 误差和梯度估计长时间处于较大偏差之下[3]; 2) 对于某些控制问题, 特别是在算法运行初期, 在某些状态下无论对当前动作均值采取正扰动还是负扰动, TD 误差都将小于或者大于 0, 由于扰动的随机性, 可能使得 θ 来回反复修改; 3) 由式 (13) 可知, 若 $s_t \sim d^\pi$, $a_t \sim \pi$, 则 $\nabla J(\pi)$ 的无偏估计为 $\int_{\mathcal{A}}\delta_t z_{t+1}\mathrm{d}a_t$, 因此在理论上, 对 $\nabla J(\pi)$ 估计时应该考虑该时间步下所有可能执行的动作, 而单动作采样仅考虑了动作 a_t, 即 $\nabla J(\pi) \approx \delta_t z_{t+1}$, 给 $\nabla J(\pi)$ 的估计带来了较大的偏差. 上述问题最终给传统 AC 方法的收敛速度和收敛质量造成了不利影响.

2.3　对称扰动采样

针对单动作采样存在的问题, 提出一种对称扰动采样方案, 即在每一时间步对当前动作均值同时实行正、负对称扰动, 从而生成两个动作与环境并行交互, 从中贪婪选取 Agent 的行为动作 (实际执行动作).

记 $C = \{+1, -1\}$, t 时间步高斯噪声扰动 $\epsilon_t =$

$\eta\sigma(s_t)$, 则该时刻生成的动作集 A_t 可定义为

$$A_t = \{a_t^i | a_t^i = \mu(s_t) + i\epsilon_t, i \in C\}, \tag{23}$$

由式 (8) 和 $\hat{V}(x) = v^{\mathrm{T}}\phi(x)$ 可知, 执行 a_t^i 时产生的 δ_t^i 可写为

$$\delta_t^i = r_t^i - \bar{r}_t + \gamma {v_t}^{\mathrm{T}}\phi(s_{t+1}^i) - {v_t}^{\mathrm{T}}\phi(s_t). \tag{24}$$

其中: s_{t+1}^i 和 r_t^i 分别为 Agent 在 s_t 下执行 a_t^i 时转移到的下一状态和所获得的立即回报.

令 a_t^i 对应的策略参数资格迹 z_t^i 更新规则为

$$z_{t+1}^i = \gamma\lambda z_t^i + \nabla\log\pi(a_t^i|s_t), \tag{25}$$

由于 $\mathcal{N}(0,1)$ 为对称分布, 且 a_t^{+1} 和 a_t^{-1} 关于 $\mu(s_t)$ 对称, 可令 $P(a_t^i|s_t,\pi) = 1/2$. 由式 (13) 和 (7) 可知, 时间步 t 下的 $\nabla J(\pi)$ 和 $G(\theta)$ 可分别估计为

$$\widehat{\nabla} J(\pi) = \sum_{i\in C}\delta_t^i z_{t+1}^i/2, \tag{26}$$

$$\widehat{G}(\theta) = \sum_{i\in C}\nabla\log\pi(a_t^i|s_t)\nabla\log\pi(a_t^i|s_t)^{\mathrm{T}}/2. \tag{27}$$

根据式 (26), 式 (14) 重新定义为

$$\theta_{t+1} = \theta_t + \alpha_{\theta,t}\sum_{i\in C}\delta_t^i z_{t+1}^i/2. \tag{28}$$

记 $F_t^i = \nabla\log\pi(a_t^i|s_t)\nabla\log\pi(a_t^i|s_t)^{\mathrm{T}}$, 根据式 (26) 和 (27), 式 (16) 重新定义为

$$w_{t+1} = w_t - \alpha_{v,t}\sum_{i\in C}(F_t^i w_t - \delta_t^i z_{t+1}^i)/2, \tag{29}$$

由式 (22) 可得

$$\nabla_{\theta_{\mu,t}}\log\pi(a_t^{+1}|s_t) = -\nabla_{\theta_{\mu,t}}\log\pi(a_t^{-1}|s_t),$$

$$\nabla_{\theta_{\sigma,t}}\log\pi(a_t^{+1}|s_t) = \nabla_{\theta_{\sigma,t}}\log\pi(a_t^{-1}|s_t),$$

则 $\sum_{i\in C}F_t^i w_t = 2F_t^{+1}w_t$. 又 z_0^{+1} 和 z_0^{-1} 一般初始化为零向量, 进而有 $z_{\mu,t+1}^{+1} = -z_{\mu,t+1}^{-1}$, $z_{\sigma,t+1}^{+1} = z_{\sigma,t+1}^{-1}$, 因此可将 z_{t+1}^{+1} 与 z_{t+1}^{-1} 合并, 只保留 z_{t+1}^{+1} 并对 θ_t 和 w_t 进行更新, 以节省内存空间. 记 $\delta_t^{\mu} = (\delta_t^{+1} - \delta_t^{-1})/2$, $\delta_t^{\sigma} = (\delta_t^{+1} + \delta_t^{-1})/2$, $\rho_{t+1} = ((\delta_t^{\mu}z_{\mu,t+1}^{+1})^{\mathrm{T}}, (\delta_t^{\sigma}z_{\sigma,t+1}^{+1})^{\mathrm{T}})^{\mathrm{T}}$, 则式 (28) 和 (29) 可分别重写为

$$\theta_{t+1} = \theta_t + \alpha_{\theta,t}\rho_{t+1}, \tag{30}$$

$$w_{t+1} = w_t - \alpha_{v,t}(F_t^{+1}w_t - \rho_{t+1}). \tag{31}$$

式 (31) 涉及 $F_t^{+1}w_t$ 的计算问题, 如果按顺序计算, 其时间复杂度为 $O(M^2)$. 考虑到

$$F_t^{+1}w_t = \nabla\log\pi(a_t^{+1}|s_t)\nabla\log\pi(a_t^{+1}|s_t)^{\mathrm{T}}w_t,$$

如果先计算右式的后两项, 则 $F_t^{+1}w_t$ 的时间复杂度为 $O(M)$, 本文在算法具体实现时将采用这一计算方式.

与 Actor 同时考虑 a_t^{-1} 和 a_t^{+1} 进行策略改进不同, Critic 基于 Agent 的行为动作 a_t, 对值函数参数进行更新. 其中 a_t 基于动作值函数从 A_t 贪婪选取, 即

$$a_t = \arg\max_{a_t^i\in A_t} Q^{\pi}(s_t, a_t^i), \tag{32}$$

由式 (4) 和 $\hat{V}(x) = v^{\mathrm{T}}\phi(x)$ 可知, $Q^{\pi}(s_t, a_t^i)$ 可估计为

$$\hat{Q}^{\pi}(s_t, a_t^i) = r_t^i - \bar{r}_t + \gamma {v_t}^{\mathrm{T}}\phi(s_{t+1}^i). \tag{33}$$

由式 (24) 可得 $\hat{Q}^{\pi}(s_t, a_t^i) = \delta_t^i + {v_t}^{\mathrm{T}}\phi(s_t)$, 因而 a_t 可基于 TD 误差贪婪选取. 记最大 TD 误差上标为

$$i_{\max} = \arg\max_{i\in C}\delta_t^i, \tag{34}$$

则 $a_t = a_t^{i_{\max}}$, Agent 执行 a_t 时, 转移到的下一状态 s_{t+1} 变为 $s_{t+1}^{i_{\max}}$, 相应的 TD 误差 δ_t 变为 $\delta_t^{i_{\max}}$, 式 (10) 和 (11) 分别重新定义为

$$v_{t+1} = v_t + \alpha_{v,t}\delta_t^{i_{\max}}e_{t+1}, \tag{35}$$

$$\bar{r}_{t+1} = \bar{r}_t + \alpha_{r,t}\delta_t^{i_{\max}}. \tag{36}$$

2.4 算法框架

由 2.1 节和 2.3 节的分析可知, 基于高斯策略分布的对称扰动采样 AC 算法框架如下.

算法 1 基于对称扰动采样的 AC 算法框架.

1) 输入: $\nabla\log\pi(u|x)$, $\phi(x)$, $\varphi(x)$, C; 初始化: γ, λ, $\alpha_{r,0}$, $\alpha_{v,0}$, $\alpha_{\theta,0}$, v_0, $\bar{r}_0$, θ_0, e_0, z_0^{+1}, w_0, s_0.

2) for $t = 0, 1, 2, \cdots$

3) 按式 (19) 和 (20) 分别计算 $\mu(s_t)$ 和 $\sigma(s_t)$;

4) 提取 $\eta \sim \mathcal{N}(0,1)$, 计算 $\epsilon_t = \eta\sigma(s_t)$.

5) for each $i \in C$

6) $a_t^i = \mu(s_t) + i\epsilon_t$;

7) 在 s_t 下执行 a_t^i, 获取 s_{t+1}^i 和 r_t^i;

8) 按式 (24) 计算 δ_t^i.

9) end for

10) 按式 (34) 确定 $i_{\max}$;

11) 按式 (9)、(35) 和 (36) 分别更新 e_t、v_t 和 $\bar{r}_t$;

12) $z_{t+1}^{+1} = \gamma\lambda z_t^{+1} + \nabla\log\pi(a_t^{+1}|s_t)$;

13) 计算 ρ_{t+1};

14) 按式 (30) 更新 θ_t (常规梯度), 或按式 (31) 和 (17) 更新 w_t 和 θ_t (自然梯度);

15) $s_{t+1} = s_{t+1}^{i_{\max}}$;

16) 分别更新 $\alpha_{r,t}$、$\alpha_{v,t}$ 和 $\alpha_{\theta,t}$.

17) end for

框架集成了平均回报常规梯度、平均回报增量自然梯度、折扣回报常规梯度和折扣回报增量自然梯度 4 种对称扰动采样 AC 算法, 为了便于后续讨论, 将上述方法分别简记为 ARAC-S、AIAC-S、DRAC-S 和 DIAC-S, 同类型的传统单动作采样 AC 算法分别简记为 ARAC-T、AIAC-T、DRAC-T 和 DIAC-T. 由 1.2 节中 $Q^{\pi}(x,u)$ 的定义可知, 对于 DRAC-S 和 DIAC-S 算法, $\bar{r}_t = 0$, 因此这两种算法所用各式中无 $\alpha_{r,t}$ 和 $\bar{r}_t$ 项.

虽然框架面向无限时域连续空间问题建立, 但稍作修改也可用于处理 episodic 任务. 对于这类任务, 当 s_{t+1}^i 为吸收终态时, 只需在算法 1 的第 8) 行计算 δ_t^i 时置 $\phi(s_{t+1}^i) = 0$, 然后在第 15) 行重新初始化 s_{t+1} 为起始状态, 并重置 $e_{t+1} = 0$, $z_{t+1}^{+1} = 0$ 即可. 此外, 尽管

本框架推导建立在累加迹之上, 但从本文后面的仿真验证可知, e_{t+1} 和 z_{t+1}^{+1} 也可采用替代迹形式.

3 性能分析

3.1 复杂度分析

由算法 1 和第 2.3 节分析可知, 框架中 Critic 和 Actor 均为线性模型, e_t 和 v_t 更新的时间与空间复杂度均为 $O(N)$, z_t^{+1}、w_t 和 θ_t 更新的时间与空间复杂度均为 $O(M)$. 在算法 1 的第 4) 行中, 在对 η 取值时, $|\eta| > 3$ 的概率只有 0.26%, 因此即便需要对 η 重新取值, 其所带来的计算量基本上可忽略不计. 相较传统 AC 方法而言, 本文方法通过贪婪策略确定 a_t, 有效地减少了正、负扰动的盲目性, 在每一时间步采用对称采样的最大 TD 误差对值函数参数进行更新, 在一定程度上可以加速值函数参数的收敛. 又由于 $\delta_t^{\mu} z_{\mu,t+1}^{+1} = (\delta_t^{+1} - \delta_t^{-1}) z_{\mu,t+1}^{+1}/2$, 由式 (30)、(31) 和 (17) 可知, $\mu(s_t)$ 倾向于朝获得更大的 TD 误差扰动方向进行调整, 同样可以加速值函数参数的收敛. 尽管本文方法在每一时间步需要执行两次采样, 但可以较好地解决传统 AC 算法存在的问题.

3.2 收敛性分析

记 Agent 行为策略为 π_g. 由式 (26) 可得, $\widehat{\nabla} J(\pi) = E_{x\sim d^{\pi_g}, A\sim\pi}\Big[\sum_i \delta^i z^i/2\Big]$, 动作集 A 仍由随机策略 π 确定, π_g 对梯度 $\widehat{\nabla} J(\pi)$ 的影响主要体现在 d^{π_g} 和对 δ^i 的逼近上. 又由于 $\pi_g \sim \pi$, π_g 实际上是一种建立在 π 上的局部贪婪随机策略, 而非确定性贪婪策略. 因此, π_g 的引入并不会使框架中各算法探索变得不充分而破坏其收敛性. 考虑到框架每时间步有多个非独立参数向量需要更新, 这里采用两时间尺度随机逼近算法[3]完成其收敛性证明, 为了简化证明, 此处仅分析 $\lambda = 0$ 情形. $\lambda > 0$ 时, 框架的收敛性可先参照文献 [5], 将策略梯度转换成向前形式, 再按照本节思路证明.

首先证明 ARAC-S 算法的收敛性. 记 $(s_t, a_t, r_t) = (s_t^{i_{\max}}, a_t^{i_{\max}}, r_t^{i_{\max}})$, 假定 $\{(s_t, a_t, r_t)|s_t \in \mathcal{S}_g, a_t \in \mathcal{A}_g, r_t \in R, t = 0, 1, \cdots\}$ 为 MDP 在给定 π_g 下生成的状态-动作-立即回报序列, $\{A_t|A_t \subset \mathcal{A}, t = 0, 1, \cdots\}$ 为 MDP 在给定 π 下生成的动作集序列. 其中: $a_t \in A_t$, $\mathcal{S}_g = \{x_1, x_2, \cdots, x_n\}$, $\mathcal{A}_g = \{u_1, u_2, \cdots, u_m\}$, $\mathcal{S}_g \subseteq \mathcal{S}$, $\mathcal{A}_g \subseteq \mathcal{A}$, n、$m \to \infty$. 令

$$D = \mathrm{diag}\{d^{\pi_g}(x_i)\},$$

$$\Phi = [\phi^{\mathrm{T}}(x_1), \phi^{\mathrm{T}}(x_2) \cdots, \phi^{\mathrm{T}}(x_n)]^{\mathrm{T}},$$

$$R(x_i, u_k) = \sum_{x_j \in \mathcal{S}_g} P(s_{t+1} = x_j | s_t = x_i, a_t = u_k) r_t,$$

$$R^{\pi_g} = \Big[\sum_{u_k \in \mathcal{A}_g} \pi_g(u_k|x_1) R(x_1, u_k), \cdots, \sum_{u_k \in \mathcal{A}_g} \pi_g(u_k|x_n) R(x_n, u_k)\Big]^{\mathrm{T}},$$

$$P^{\pi_g} = \Big\{p^{\pi_g}(x_i, x_j) = \sum_{u_k \in \mathcal{A}_g} \pi_g(u_k|x_i) P(s_{t+1} = x_j | s_t = x_i, a_t = u_k)\Big\},$$

$$T(J) = R^{\pi_g} - J(\pi_g) i_e + P^{\pi_g} J.$$

其中: $\pi_g(u_k|x_i)$ 为给定 x_i 下按高斯分布生成 u_k 的概率, $J(\pi_g) = \sum_{x\in\mathcal{S}_g} d^{\pi_g}(x) \sum_{u\in\mathcal{A}_g} \pi_g(u|x) R(x, u)$, i_e 为 n 维全 1 列向量. 这里令 $\zeta_t^i = r_t^i - \bar{r}_t + \gamma v_t^{\pi_{g,t}\mathrm{T}} \phi(s_{t+1}) - v_t^{\pi_{g,t}\mathrm{T}} \phi(s_t)$, $\pi_{g,t}$ 为 θ_t 给定时 Agent 的行为策略. 令 Γ 和 $\hat{\Gamma}$ 为投影算子, 含义同文献 [5] 附录 A.3.

假设 1　对于任一给定 π_g, Markov 链 $\{s_t, t = 0, 1, \cdots\}$ 具有不可约性和非周期性.

假设 2　对于 $\forall t \in N$, $\phi(s_t)$、$\phi(s_{t+1})$ 和 r_t 的二阶矩均一致有界, Φ 为满秩阵, 且 $\Phi v \neq i_e$.

假设 3　对于任意确定且为正的 t、$\alpha_{v,t}$ 和 $\alpha_{\theta,t}$, $\sum_t \alpha_{v,t} = \sum_t \alpha_{\theta,t} = \infty$, $\sum_t \alpha_{v,t}^2 < \infty$, $\sum_t \alpha_{\theta,t}^2 < \infty$, $\dfrac{\alpha_{\theta,t}}{\alpha_{v,t}} \to 0$, $\alpha_{r,t} = \beta \alpha_{v,t}$, β 为正常数.

引理 1　在假设 1 ∼ 假设 3 的条件下, 任一给定 π_g、$\bar{r}_t$ 和 v_t 将以概率 1 分别收敛于 $J(\pi_g)$ 和 v^{π_g}, 其中 v^{π_g} 为 $\Phi^{\mathrm{T}} D \Phi v^{\pi_g} = \Phi^{\mathrm{T}} D T(\Phi v^{\pi_g})$ 的唯一解.

证明　根据两时间尺度随机逼近算法证明 $\bar{r}_t$ 和 v_t 的收敛时, 令 θ 固定, 即 π_g 确定, 式 (36) 可写为 $\bar{r}_{t+1} = (1-\alpha_{r,t})\bar{r}_t + \alpha_{r,t}(r_t + v_t^{\mathrm{T}} \phi(s_{t+1}) - v_t^{\mathrm{T}} \phi(s_t))$. 又不难证明 $\mathrm{E}[v_t^{\mathrm{T}} \phi(s_{t+1}) - v_t^{\mathrm{T}} \phi(s_t)|\pi_g] = 0$, 根据文献 [14] 中的定理 1, 令 $\lambda = 0$, 可得引理 1. □

定理 1　在假设 1 ∼ 假设 3 的条件下, θ_t 将以概率 1 收敛于 $\dot{\theta} = \hat{\Gamma}(\widehat{\nabla}\mathcal{J}(\pi))$ 的渐近稳定平衡点集, 其中 $\widehat{\nabla}\mathcal{J}(\pi) = \mathrm{E}_{x\sim d^{\pi_g}, A\sim\pi}\Big[\sum_i \zeta^i \nabla \log\pi(a^i|x)/2\Big]$.

证明　令 $\widehat{\nabla} J(\pi_{g,t}) = \sum_i \zeta_t^i \nabla \log\pi(a_t^i|s_t)/2$. 根据文献 [3] 的定理 2, 令 $\mathcal{F}_2(t) = \sigma(\theta_r, r \leqslant t)$. 由式 (26) 可知, 当 $\lambda = 0$ 时, $\widehat{\nabla} J(\pi_t) = \sum_i \delta_t^i \nabla \log\pi(a_t^i|s_t)/2$. 令

$$k(\theta_t) = \mathrm{E}[\widehat{\nabla} J(\pi_{g,t})|\mathcal{F}_2(t)],$$

$$\vartheta_t = \widehat{\nabla} J(\pi_t) - E[\widehat{\nabla} J(\pi_t)|\mathcal{F}_2(t)],$$

$$\varepsilon_t = \mathrm{E}[(\widehat{\nabla} J(\pi_t) - \widehat{\nabla} J(\pi_{g,t}))|\mathcal{F}_2(t)].$$

由式 (28) 可得, $\theta_{t+1} = \Gamma(\theta_t + \alpha_{\theta,t}(k(\theta_t) + \vartheta_t + \varepsilon_t))$, 其中 ODE 为 $\dot{\theta} = \hat{\Gamma}(\widehat{\nabla}\mathcal{J}(\pi))$. 令 $M^2(t) = \sum_{r=0}^{t-1} \alpha_{\theta,r} \vartheta_r, t \geqslant 1$. 由引理 1 和文献 [3] 中定理 2 的证明过程易知: 1) $\varepsilon_t = o(1)$, 2) $\{M^2(t)\}$ 为收敛鞅序列, 3) $k(\theta)$ 为 Lipschitz 连续函数, 4) $\dot{\theta} = \hat{\Gamma}(\widehat{\nabla}\mathcal{J}(\pi))$ 适定. 根据文献 [15] 可得定理 1. □

同理, 由文献 [12] 中的定理 1 和本文定理 1 可知, DRAC-S 算法收敛. 类似地, AIAC-S 算法和 DIAC-S

算法的收敛性可结合文献[14]和文献[12]的定理1及文献[3]的6.2节和6.3节所述方法予以证明, 在此不再赘述.

4 仿真研究

4.1 仿真问题与设置

根据框架集成的各AC算法的具体特性, 先选用小车爬山问题[2]对DRAC-S算法和DIAC-S算法进行验证, 再选用受限力矩倒立摆起摆问题[16]对ARAC-S算法和AIAC-S算法进行验证, 并与相应的传统AC算法进行比较. 在两个仿真实验中, 各算法的资格迹除了使用累加迹外, 还将尝试使用替代迹.

小车爬山问题的学习目标是确定一种油门策略, 使小车能以尽量短的时间从山谷最低点到达右侧山顶. 其动力学方程如文献[2]所述, 连续状态变量 $x_t \in [-1.5, 0.5]$ 和 $\dot{x}_t \in [-0.07, 0.07]$ 分别为 t 时刻小车的位置和速度, 连续动作变量 $a_t \in [-1, 1]$ 为 t 时刻对小车油门实施的控制. 若 $x_{t+1} \geqslant 0.5$, 则表明小车已到达山顶, 此时获得立即回报 $r_t = 0$, 否则 $r_t = -1$; 若 $x_{t+1} \leqslant -1.5$, 则置 $\dot{x}_{t+1} = 0$. 在仿真中, 对每次实验共执行100个episode, 在每一episode小车均从初始状态 $(x_0, \dot{x}_0) = (-0.5, 0)$ 出发, 当其到达山顶或运行步数达到1 000时间步时, 该episode学习结束.

倒立摆起摆问题的学习目标是确定一力矩策略, 使摆杆尽可能迅速地从某一位置举起并长时间保持平衡. 其动力学方程如文献[16]所述, 连续状态变量 $\theta_t \in [-\pi, \pi]$ 和 $\omega_t \in [-78.54, 78.54]$ 分别为 t 时刻摆杆偏离垂直方向的角度和角速度, 连续动作变量 $u_t \in [-2, 2]$ 为 t 时刻对摆杆绕旋转轴施加的力矩. 如果 $\theta_{t+1} > \pi$, 则置 $\theta_{t+1} = \theta_{t+1} - 2\pi$; 如果 $\theta_{t+1} < -\pi$, 则置 $\theta_{t+1} = \theta_{t+1} + 2\pi$. 在仿真中, 采用Euler法模拟动态系统, 离散时间步长为0.01 s, 摆杆初始状态 $(\theta_0, \omega_0) = (\pi/2, 0)$, 立即回报 $r_t = \cos\theta_{t+1}$, 每次实验共执行100个时间段, 每段包括1 000个时间步, 且每运行完一段, 重置摆杆回初始状态然后继续运行.

类似于文献[4], 两个仿真实验的参数设置为: $\gamma = 1$, α_r、α_v、α_θ 为常数; 值函数和策略参数逼近均基于Tile-coding[2]实现, 设 $\phi(x) = \varphi_\mu(x) = \varphi_\sigma(x)$, $\phi(x)$ 采用10个 11×11 tilings进行量化编码; 初始化 $v_0 = \theta_{\mu,0} = \theta_{\sigma,0} = e_0 = z_0^{+1} = 0$, $\bar{r}_0 = 0$. 为了使验证更加客观, 各AC算法均选用最优 α_r、α_v、α_θ、λ 重复运行30次. 最优 α_r、α_v、α_θ、λ 通过参数扫描法确定. 其中: 参数 α_r、α_v、α_θ 所采用的扫描集为 $\{0.000\,1, 0.000\,5, 0.001, \cdots, 0.5, 1\}$, 参数 λ 采用的扫描集为 $\{0, 0.5, 0.75, 0.875, 0.937\,5\}$. 参数扫描时, 最优 α_r、α_v、α_θ、λ 的确定标准为: 1) 小车爬山问题, 综合参考第10个和第100个episode学习得到的时间步大小; 2) 倒立摆起摆问题, 综合参考第10个和第100个时间段内学习得到的平均回报值. 此外, 根据第3.2节的假设3可知, 扫描期间应使 $\alpha_\theta < \alpha_v$.

4.2 仿真结果与分析

两个问题重复执行30次的平均仿真实验结果分别如表1、表2和图1、图2所示, 算法名称后缀"-A"或"-R"表示该算法资格迹采用累加迹或替代迹. 在表1和表2中: 各参数为参数扫描法确定的最优参数, Steps-I和Steps-II分别为各折扣回报AC算法在最优参数下于第10个和第100个episode学习得到的平均时间步与标准差大小, Reward-I和Reward-II分别为各平均回报AC算法在最优参数下于第10个和第100个时间段内每一时间步学习得到的平均回报值与标准差大小. 图1和图2分别为各折扣回报和平均回报AC算法在最优参数下的学习曲线.

表1 折扣回报AC算法小车爬山实验

算法	α_v	α_θ	λ	Steps-I	Steps-II
DRAC-T-A	1	0.05	0	255.0±45.5	128.7±27.8
DRAC-T-R	0.5	0.05	0	269.2±39.9	126.9±24.0
DRAC-S-A	0.5	0.005	0.75	210.1±40.1	118.3±22.5
DRAC-S-R	0.5	0.01	0.875	215.7±39.0	118.5±18.2
DIAC-T-A	0.1	0.000 1	0.5	595.5±193.1	178.7±20.2
DIAC-T-R	0.1	0.000 1	0.5	722.5±204.2	191.4±34.8
DIAC-S-A	0.1	0.000 1	0.875	226.5±42.1	112.3±2.7
DIAC-S-R	0.1	0.000 5	0.9375	212.3±40.5	107.2±2.6

表2 平均回报AC算法倒立摆起摆实验

算法	α_r	α_v	α_θ	λ	Reward-I	Reward-II
ARAC-T-A	0.05	0.5	0.1	0.5	0.853±0.137	0.969±0.004
ARAC-T-R	0.05	1	0.1	0.75	0.910±0.059	0.968±0.008
ARAC-S-A	0.1	0.5	0.1	0.75	0.875±0.159	0.965±0.008
ARAC-S-R	0.1	1	0.1	0.75	0.942±0.022	0.967±0.005
AIAC-T-A	0.05	0.05	0.001	0.937 5	0.371±0.192	0.735±0.317
AIAC-T-R	0.05	0.05	0.001	0.937 5	0.348±0.142	0.716±0.311
AIAC-S-A	0.01	0.1	0.001	0.75	0.774±0.319	0.956±0.024
AIAC-S-R	0.01	0.1	0.001	0.75	0.701±0.349	0.953±0.031

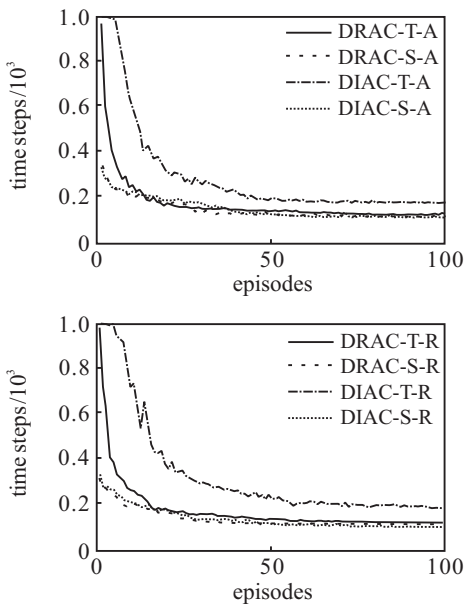

图1 折扣回报AC算法小车爬山实验

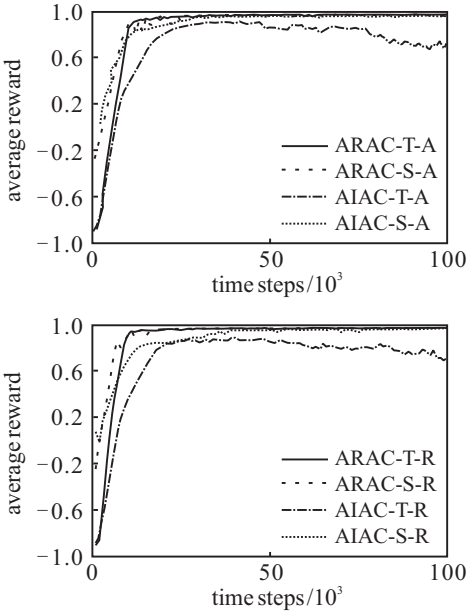

图 2　平均回报 AC 算法倒立摆起摆实验

由仿真实验结果可得出如下结论: 1) 对称扰动采样可有效提高传统 AC 方法的收敛速度和质量, 特别是对传统增量自然梯度 AC 方法. 单纯从表 2 和图 2 来看, ARAC-T 与ARAC-S 算法性能非常接近, 但就参数扫描过程而言, ARAC-S 算法对不同 α_r、α_v、α_θ、λ 的适应能力优于 ARAC-T 算法. 2) 各对称扰动采样算法的最优参数与传统 AC 算法比较接近, 表明前者在收敛性能上的提高主要得益于对称扰动采样机制的引入. 3) 各 AC 算法的资格迹采用替代迹也是可行的, 在有些情况下可有效改善算法的收敛性能. 4) 除 DIAC-S 算法外, 增量自然梯度 AC 方法的表现并不如常规梯度 AC 方法.

5　结　论

针对传统 AC 方法求解连续空间问题收敛速度和收敛质量不高这一问题, 本文提出了一种基于高斯策略分布的对称扰动采样 AC 算法框架, 集成了 ARAC-S、AIAC-S、DRAC-S 和 DIAC-S 四种算法. 各算法在每一时间步通过对称高斯扰动生成两个动作与环境并行交互, 在此基础上, Critic 基于两者的最大 TD 误差确定 Agent 的行为动作并对值函数参数进行更新, Actor 基于两者的平均常规梯度或增量自然梯度对策略参数进行更新. 理论分析和仿真研究表明, 所提出的框架具有较好的收敛性能和计算效率. 本文的研究成果表明, 可以通过增加动作扰动的贪婪性来改善传统 AC 方法的收敛速度和质量.

参考文献(References)

[1] Xu X, Hou Z, Lian C, et al. Online learning control using adaptive critic designs with sparse kernel machines[J]. IEEE Trans on Neural Networks and Learning Systems, 2013, 24(5): 762-775.

[2] Sutton R S, Barto A G. Reinforcement learning: An introduction[M]. Cambridge: MIT Press, 1998: 151-215.

[3] Bhatnagar S, Sutton R S, Ghavamzadeh M, et al. Natural actor-critic algorithms[R]. Canada: Department of Computing Science, University of Alberta, 2009.

[4] Degris T, Pilarski P M, Sutton R S. Model-free reinforcement learning with continuous action in practice[C]. 2012 American Control Conf. Montreal: IEEE Press, 2012: 2177-2182.

[5] Degris T, White M, Sutton R S. Off-policy actor-critic[C]. The 29th Int Conf on Machine Learning. Edinburgh: Omnipress, 2012: 457-464.

[6] Silver D, Lever G, Heess N, et al. Deterministic policy gradient algorithms[C]. The 31st Int Conf on Machine Learning. Beijing: JMLR W & CP, 2014: 387-395.

[7] Lee D, Lee J. Incremental receptive field weighted actor-critic[J]. IEEE Trans on Industrial Informatics, 2013, 9(1): 62-71.

[8] Pazis J, Lagoudakis M G. Binary action search for learning continuous-action control policies[C]. The 26th Int Conf on Machine Learning. Montreal: ACM Press, 2009: 793-800.

[9] Sehnke F, Osendorfer C, Rückstieβ T, et al. Parameter-exploring policy gradients[J]. Neural Networks, 2010, 23(4): 551-559.

[10] Peters J, Schaal S. Natural actor-critic[J]. Neurocomputing, 2008, 71(7/8/9): 1180-1190.

[11] Sutton R S, McAllester D A, Singh S P, et al. Policy gradient methods for reinforcement learning with function approximation[C]. Advances in Neural Information Processing Systems 12. Denver: MIT Press, 2000: 1057-1063.

[12] Tsitsiklis J N, Van Roy B. An analysis of temporal-difference learning with function approximation[J]. IEEE Trans on Automatic Control, 1997, 42(5): 674-690.

[13] Williams R J. Simple statistical gradient-following algorithms for connectionist reinforcement learning[J]. Machine Learning, 1992, 8(3/4): 229-256.

[14] Tsitsiklis J N, Van Roy B. Average cost temporal-difference learning[J]. Automatica, 1999, 35(11): 1799-1808.

[15] Kushner H J, Clark D S. Stochastic approximation methods for constrained and unconstrained systems[M]. New York: Springer, 1978: 191-196.

[16] Doya K. Reinforcement learning in cointinuous time and space[J]. Neural Computation, 2000, 12(1): 219-245.

(责任编辑: 闫　妍)