目录

1 模版备用

算法[1]
1: 测试

[1]脚注

2 策略迭代算法

策略迭代算法

初始化状态价值函数$V(s)$和策略$\pi(s)$
策略估计：
repeat
  $\Delta \leftarrow 0$
  repeat
    $v \leftarrow V(s)$
    $V(s) \leftarrow \sum_{s',r} p\left(s', r \mid s, \pi(s)\right)\left[r+\gamma V\left(s'\right)\right]$
    $\Delta \leftarrow \max(\Delta, |v-V(s)|)$
  until 遍历所有的状态$s \in S$
until $\Delta < \theta$
策略改进：
$stable\_flag \leftarrow true$
repeat
  根据策略$\pi(a|s)$生成动作a_{temp}
  更新策略：$\pi(a|s) \leftarrow \arg\max_a \sum_{s',r} p\left(s', r \mid s, a\right)\left[r+\gamma V\left(s'\right)\right]$
  if $a_{temp} \neq \pi(a|s)$ **then**
    说明策略还未收敛，$stable\_flag \leftarrow false$
  end if
until 遍历所有的状态$s \in S$
if $stable\_flag \leftarrow true$ **then**
  结束迭代并返回最优策略$\pi \approx \pi_*$和状态价值函数$V \approx V_*$
else
  继续执行策略估计·
end if

3 价值迭代算法

价值迭代算法

初始化一个很小的参数阈值$\theta > 0$，以及状态价值函数$V(s)$，注意终止状态的$V(s_T) = 0$

repeat

  $\Delta \leftarrow 0$

  repeat

    $v \leftarrow V(s)$

    $V(s) \leftarrow \max_a \sum_{s',r} p\left(s', r \mid s, a\right)\left[r + \gamma V\left(s'\right)\right]$

    $\Delta \leftarrow \max(\Delta, |v - V(s)|)$

  until 遍历所有的状态$s \in S$

until $\Delta < \theta$

输出一个确定性策略$\pi \approx \pi_*$，
且$\pi(s) = \arg\max_a \sum_{s',r} p\left(s', r \mid s, a\right)\left[r + \gamma V\left(s'\right)\right]$

4 首次访问蒙特卡洛算法

首次访问蒙特卡洛算法

```
初始化价值函数 V(s)，一个空的回报列表 Returns(s_t)
for 回合数 = 1, M do
    根据策略π采样一回合轨迹τ = {s_0, a_0, r_1, ⋯ , s_{T−1}, a_{T−1}, r_T, }
    初始化回报 G ← 0
    for 时步 t = T − 1, , T − 2, ⋯ , 0 do
        G ← γG + R_{t+1}
        repeat
            将 G 添加到 Returns(s_t)
            V(S_t) ← average(Returns(S_t))
        until s_t 第二次出现，即与历史某个状态s_0, ⋯ , s_{t−1}相同
    end for
end for
```

[1]脚注

5 Q learning算法

Q-learning算法[1]

初始化Q表$Q(s,a)$为任意值，但其中$Q(s_{terminal},)=0$，即终止状态对应的Q值为0
for 回合数 $=1,M$ **do**
  重置环境，获得初始状态s_1
  for 时步 $=1,T$ **do**
    根据$\varepsilon-greedy$策略采样动作a_t
    环境根据a_t反馈奖励r_t和下一个状态s_{t+1}
    更新策略：
    $Q(s_t,a_t)\leftarrow Q(s_t,a_t)+\alpha[r_t+\gamma\max_a Q(s_{t+1},a)-Q(s_t,a_t)]$
    更新状态$s_{t+1}\leftarrow s_t$
  end for
end for

[1] Reinforcement Learning: An Introduction

6 Sarsa算法

Sarsa算法[1]

初始化Q表$Q(s,a)$为任意值，但其中$Q(s_{terminal},)=0$，即终止状态对应的Q值为0
for 回合数 $=1,M$ **do**
  重置环境，获得初始状态s_1
  根据$\varepsilon-greedy$策略采样初始动作a_1
  for 时步 $=1,t$ **do**
    环境根据a_t反馈奖励r_t和下一个状态s_{t+1}
    根据$\varepsilon-greedy$策略s_{t+1}和采样动作a_{t+1}
    更新策略：
    $Q(s_t,a_t)\leftarrow Q(s_t,a_t)+\alpha[r_t+\gamma Q(s_{t+1},a_{t+1})-Q(s_t,a_t)]$
    更新状态$s_{t+1}\leftarrow s_t$
    更新动作$a_{t+1}\leftarrow a_t$
  end for
end for

[1] Reinforcement Learning: An Introduction

7 DQN算法

DQN算法

初始化当前网络参数 θ 和目标网络参数 $\hat{\theta} \leftarrow \theta$
初始化经验回放D
for 回合数 $m = 1, 2, \cdots, M$ **do**
  重置环境，获得初始状态s_0
  for 时步 $t = 1, 2, \cdots, T$ **do**
    交互采样：
    根据$\varepsilon - greedy$策略采样动作a_t
    环境根据a_t反馈奖励r_t和下一个状态s_{t+1}
    存储样本(s_t, a_t, r_t, s_{t+1})到经验回放D中
    更新环境状态$s_{t+1} \leftarrow s_t$
    策略更新：
    从D中随机采样一个批量的样本
    计算Q的期望值，即$y_i = r_t + \gamma \max_{a_{i+1}} Q(s_{i+1}, a; \hat{\theta})$
    计算损失$L(\theta) = \left(y_i - Q\left(s_i, a_i; \theta\right)\right)^2$，并关于参数$\theta$做随机梯度下降
    每C步复制参数到目标网络$\hat{\theta} \leftarrow \theta$
  end for
end for

8 DRQN算法

DRQN算法[1]

初始化策略网络参数θ
复制参数到目标网络$\hat{Q} \leftarrow Q$
初始化经验回放D
for 回合数 $= 1, M$ **do**
  重置环境,获得初始状态的观测o_t
  $h_0 \leftarrow 0$
  for 时步 $= 1, t$ **do**
    根据$\varepsilon - greedy$策略采样动作a_t
    环境根据a_t反馈奖励r_t和下一个状态，生成下一状态的观测o_{t+1}
    存储transition即(o_t, a_t, r_t, o_{t+1})到经验回放D中
    更新环境状态对应的观测$o_{t+1} \leftarrow o_t$
    更新策略：
    从D中采样一个batch的transition, 即
    $B = \left\{(s_j, a_j, r_j, s_j^{'}) \ldots (s_{j+\tau}, a_{j+\tau}, r_{j+\tau}, s_{j+\tau}^{'})\right\}_{j=1}^{\text{batch size}} \subseteq D$
    for 这个batch中的每个transition **do**
      $h_{j-1} \leftarrow 0$
      for $k = j$ to $k = j + \tau$ **do**
        更新LSTM网络的隐藏状态 $h_k = Q(s_k, h_{k-1}|\theta_i)$
      end for
      计算实际的Q值，即y_j[2]
      计算损失 $L(\theta) = \left(y_i - Q\left(s_{j+\tau}, a_{j+\tau}, h_{j+\tau-1}; \theta\right)\right)^2$
    end for
    关于参数θ做随机梯度下降[3]
    每C个回合复制参数$\hat{Q} \leftarrow Q$[4]]
  end for
end for

[1] Deep Recurrent Q-Learning for Partially Observable MDPs

[2] $y_j = \begin{cases} r_j & \text{对于终止状态} s_{i+1} \\ r_j + \gamma \max_{a'} Q\left(s_{j+\tau}, a_{j+\tau}, h_{j+\tau-1}; \theta\right) & \text{对于非终止状态} s_{i+1} \end{cases}$

[3] $\theta_i \leftarrow \theta_i - \lambda \nabla_{\theta_i} L_i\left(\theta_i\right)$

9 PER-DQN算法

PER-DQN算法

```
初始化当前网络参数 θ
复制参数到目标网络θ̂ ← θ
初始化经验回放D
for 回合数 m = 1, 2, ··· , M do
    重置环境，获得初始状态s_0
    for 时步 t = 1, 2, ··· , T do
        交互采样：
        根据ε − greedy策略采样动作a_t
        环境根据a_t反馈奖励r_t和下一个状态s_{t+1}
        存储样本(s_t, a_t, r_t, s_{t+1})到经验回放D中，并根据TD误差损失确定其优先级p_t
        更新环境状态s_{t+1} ← s_t
        模型更新：
        根据每个样本的优先级计算采样概率P(j) = p_j^α / Σ_i p_i^α，从D中采样一个批量的样本
        计算各个样本重要性采样权重 w_j = (N · P(j))^{−β} / max_i w_i
        计算TD误差δ_j；并根据TD误差更新优先级p_j
        计算Q的估计值，即y_j
        根据重要性采样权重调整损失 L(θ) = (y_j − Q(s_j, a_j; θ) · w_j)^2，并关于θ做随机梯度下降
        每C步复制参数Q̂ ← Q
    end for
end for
```

10 REINFORCE 算法

REINFORCE 算法

初始化策略参数 θ
for 迭代次数 $= 1, M$ **do**
  根据策略 π_θ 采样轨迹：$\tau = \{s_0, a_0, r_0, \cdots, s_T, a_T, r_T\}$
  for 时步 $t = 0, 1, \cdots, T-1$ **do**
    计算回报$G \leftarrow \sum_{k=t+1}^{T} \gamma^{k-t-1} r_k$
    更新策略$\theta \leftarrow \theta + \alpha\gamma^t G \nabla \ln \pi_\theta\left(a_t \mid s_t\right)$
  end for
end for

11 Advantage Actor Critic算法

Q Actor Critic算法

```
初始化Actor参数θ和Critic参数w
for 回合数 = 1, M do
    根据策略π_θ(a|s)采样一个(或几个)回合的transition
    更新Critic参数[1]
    for 时步 = t+1, 1 do
        计算Advantage，即δ_t = r_t + γQ_w(s_{t+1}, a_{t+1}) − Q_w(s_t, a_t)
        w ← w + α_w δ_t ∇_w Q_w(s_t, a_t)
        a_t ← a_{t+1}, s_t ← s_{t+1}
    end for
    更新Actor参数θ ← θ + α_θ Q_w(s, a) ∇_θ log π_θ(a | s)
end for
```

[1]这里结合TD error的特性按照从$t+1$到1计算法Advantage更方便

12 PPO-Clip算法

PPO-Clip算法

初始化策略网络(Actor)参数θ和价值网络(Critic)参数ϕ
初始化Clip参数ϵ
初始化epoch数K
初始化经验回放D
for 回合数 $= 1, 2, \cdots, M$ **do**
  使用策略π_θ采样C个时步数据,收集轨迹$\tau = s_0, a_0, r_1, ..., s_t, a_t, r_{t+1}, \cdots$到经验回放$D$中
  for epoch数 $k = 1, 2, \cdots, K$ **do**
    计算折扣奖励$\hat{R}_t$
    计算优势函数，即$A^{\pi_{\theta_k}} = V_{\phi_k} - \hat{R}_t$
    结合重要性采样计算Actor损失，如下：
    $L^{CLIP}(\theta) = \frac{1}{|D_k|T}\sum_{\tau\in D_k}\sum_{t=0}^{T} min(\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_k}(a_t|s_t)}A^{\pi_{\theta_k}}(s_t, a_t), g(\epsilon, A^{\pi_{\theta_k}}(s_t, a_t)))$
    梯度下降更新Actor参数：$\theta_{k+1} \leftarrow \theta_k + \alpha_\theta L^{CLIP}(\theta)$
    更新Critic参数:
    $\phi_{k+1} \leftarrow \phi_k + \alpha_\phi \frac{1}{|D_k|T}\sum_{\tau\in D_k}\sum_{t=0}^{T}(V_{\phi_k}(s_t) - \hat{R}_t)^2$
  end for
end for

13 PPO-KL散度算法

PPO-KL散度算法①②

初始化策略网络(Actor)参数θ和价值网络(Critic)参数ϕ
初始化KL散度参数λ
初始化回合数量M
初始化epoch数量K
初始化经验回放D
for 回合数 $= 1, 2, \cdots, M$ **do**
  根据策略π_{θ_m}采样一个或几个回合数据,收集(s_t, a_t, r_t)到经验回放$D_m = \{\tau_i\}$中
  for epoch数 $= 1, 2, \cdots, K$ **do**
    计算折扣奖励$\hat{R}_t$
    根据值函数V_{Φ_m},用某种优势估计方法计算优势函数$\hat{A}_t$
    通过最大化目标函数$J_{PPO}(\theta)$更新参数θ：
    $J_{PPO}(\theta) = \sum_{t=1}^{T} \frac{\pi_\theta(a_t|s_t)}{\pi_{old}(a_t|s_t)} \hat{A}_t - \lambda KL[\pi_{old}|\pi_\theta]$
    典型方法是Adam随机梯度上升
    根据均方误差回归拟合值函数，更新Critic参数:
    $\Phi_{m+1} \leftarrow \frac{1}{|D_m|T} \sum_{\tau \in D_m} \sum_{t=0}^{T} (V_{\Phi_m}(s_t) - \hat{R}_t)^2$
    运用某些梯度下降算法
    if $KL[\pi_{old}|\pi_\theta] > \beta_{high} KL_{target}$ **then**
      $\lambda \leftarrow \alpha\lambda$
    else if $KL[\pi_{old}|\pi_\theta] < \beta_{low} KL_{target}$ **then**
      $\lambda \leftarrow \frac{\lambda}{\alpha}$
    end if
  end for
end for

①Proximal Policy Optimization Algorithms

②Emergence of Locomotion Behaviours in Rich Environments

14 DDPG算法

DDPG算法

初始化critic网络$Q\left(s, a \mid \theta^{Q}\right)$和actor网络$\mu(s|\theta^{\mu})$的参数$\theta^{Q}$和$\theta^{\mu}$
初始化对应的目标网络参数，即$\theta^{Q'} \leftarrow \theta^{Q}, \theta^{\mu'} \leftarrow \theta^{\mu}$
初始化经验回放 D
for 回合数 $=1, M$ **do**
  交互采样：
  选择动作$a_t=\mu\left(s_t \mid \theta^{\mu}\right)+\mathcal{N}_t$，$\mathcal{N}_t$为探索噪声
  环境根据a_t反馈奖励s_t和下一个状态s_{t+1}
  存储样本(s_t, a_t, r_t, s_{t+1})到经验回放 D 中
  更新环境状态$s_{t+1} \leftarrow s_t$
  策略更新：
  从 D 中取出一个随机批量的(s_i, a_i, r_i, s_{i+1})
  求得$y_i=r_i+\gamma Q'\left(s_{i+1}, \mu'\left(s_{i+1} \mid \theta^{\mu'}\right) \mid \theta^{Q'}\right)$
  更新 critic 参数，其损失为：$L=\frac{1}{N} \sum_i\left(y_i-Q\left(s_i, a_i \mid \theta^{Q}\right)\right)^2$
  更新 actor 参数：$\nabla_{\theta^{\mu}} J \approx \frac{1}{N} \sum_i \nabla_a Q\left(s, a \mid \theta^{Q}\right)\Big|_{s=s_i, a=\mu(s_i)} \nabla_{\theta^{\mu}} \mu\left(s \mid \theta^{\mu}\right)\Big|_{s_i}$
  软更新目标网络：$\theta^{Q'} \leftarrow \tau \theta^{Q}+(1-\tau) \theta^{Q'}$， $\theta^{\mu'} \leftarrow \tau \theta^{\mu}+(1-\tau) \theta^{\mu'}$
end for

15 SoftQ算法

SoftQ算法

初始化参数θ和ϕ
复制参数$\bar{\theta} \leftarrow \theta, \bar{\phi} \leftarrow \phi$
初始化经验回放D
for 回合数 $= 1, M$ **do**
    for 时步 $= 1, t$ **do**
        根据$\mathbf{a}_t \leftarrow f^{\phi}(\xi; \mathbf{s}_t)$采样动作，其中$\xi \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$
        环境根据a_t反馈奖励s_t和下一个状态s_{t+1}
        存储transition即(s_t, a_t, r_t, s_{t+1})到经验回放D中
        更新环境状态$s_{t+1} \leftarrow s_t$
        更新soft Q函数参数：
        对于每个$s_{t+1}^{(i)}$采样$\left\{\mathbf{a}^{(i,j)}\right\}_{j=0}^{M} \sim q_{\mathbf{a}'}$
        计算empirical soft values $V_{\text{soft}}^{\theta}(\mathbf{s}_t)$①
        计算empirical gradient $J_Q(\theta)$②
        根据$J_Q(\theta)$使用ADAM更新参数θ
        更新策略：
        对于每个$s_t^{(i)}$采样$\left\{\xi^{(i,j)}\right\}_{j=0}^{M} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$
        计算$\mathbf{a}_t^{(i,j)} = f^{\phi}\left(\xi^{(i,j)}, \mathbf{s}_t^{(i)}\right)$
        使用经验估计计算$\Delta f^{\phi}(\cdot; \mathbf{s}_t)$③
        计算经验估计$\frac{\partial J_\pi(\phi; \mathbf{s}_t)}{\partial \phi} \propto \mathbb{E}_{\xi}\left[\Delta f^{\phi}(\xi; \mathbf{s}_t) \frac{\partial f^{\phi}(\xi; \mathbf{s}_t)}{\partial \phi}\right]$，即$\hat{\nabla}_{\phi} J_\pi$
        根据$\hat{\nabla}_{\phi} J_\pi$使用ADAM更新参数$\phi$

    end for
    每C个回合复制参数$\bar{\theta} \leftarrow \theta, \bar{\phi} \leftarrow \phi$
end for

① $V_{\text{soft}}^{\theta}(\mathbf{s}_t) = \alpha \log \mathbb{E}_{q_{\mathbf{a}'}}\left[\frac{\exp\left(\frac{1}{\alpha} Q_{\text{soft}}^{\theta}(\mathbf{s}_t, \mathbf{a}')\right)}{q_{\mathbf{a}'}(\mathbf{a}')}\right]$

② $J_Q(\theta) = \mathbb{E}_{\mathbf{s}_t \sim q_{\mathbf{s}_t}, \mathbf{a}_t \sim q_{\mathbf{a}_t}}\left[\frac{1}{2}\left(\hat{Q}_{\text{soft}}^{\bar{\theta}}(\mathbf{s}_t, \mathbf{a}_t) - Q_{\text{soft}}^{\theta}(\mathbf{s}_t, \mathbf{a}_t)\right)^2\right]$

③
$$\begin{aligned}\Delta f^{\phi}(\cdot; \mathbf{s}_t) = & \mathbb{E}_{\mathbf{a}_t \sim \pi^{\phi}}\left[\kappa\left(\mathbf{a}_t, f^{\phi}(\cdot; \mathbf{s}_t)\right) \nabla_{\mathbf{a}'} Q_{\text{soft}}^{\theta}(\mathbf{s}_t, \mathbf{a}')\Big|_{\mathbf{a}'=\mathbf{a}_t}\right. \\ & \left. + \alpha \nabla_{\mathbf{a}'} \kappa\left(\mathbf{a}', f^{\phi}(\cdot; \mathbf{s}_t)\right)\Big|_{\mathbf{a}'=\mathbf{a}_t}\right]\end{aligned}$$

16 SAC-S算法

SAC-S算法

初始化参数$\psi, \bar{\psi}, \theta, \phi$
for 回合数 $= 1, M$ **do**
  for 时步 $= 1, t$ **do**
    根据$\boldsymbol{a}_t \sim \pi_\phi\left(\boldsymbol{a}_t \mid \mathbf{s}_t\right)$采样动作$a_t$
    环境反馈奖励和下一个状态，$\mathbf{s}_{t+1} \sim p\left(\mathbf{s}_{t+1} \mid \mathbf{s}_t, \mathbf{a}_t\right)$
    存储transition到经验回放中，$\mathcal{D} \leftarrow \mathcal{D} \cup\left\{\left(\mathbf{s}_t, \mathbf{a}_t, r\left(\mathbf{s}_t, \mathbf{a}_t\right), \mathbf{s}_{t+1}\right)\right\}$
    更新环境状态$s_{t+1} \leftarrow s_t$
    更新策略：
    $\psi \leftarrow \psi-\lambda_V \hat{\nabla}_\psi J_V(\psi)$
    $\theta_i \leftarrow \theta_i-\lambda_Q \hat{\nabla}_{\theta_i} J_Q\left(\theta_i\right)$ for $i \in\{1,2\}$
    $\phi \leftarrow \phi-\lambda_\pi \hat{\nabla}_\phi J_\pi(\phi)$
    $\bar{\psi} \leftarrow \tau \psi+(1-\tau) \bar{\psi}$
  end for
end for

[①]Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor

17 SAC算法

SAC算法

初始化网络参数θ_1, θ_2以及ϕ
复制参数到目标网络$\bar{\theta_1} \leftarrow \theta_1, \bar{\theta_2} \leftarrow \theta_2$,
初始化经验回放D
for 回合数 $= 1, M$ **do**
  重置环境，获得初始状态s_t
  for 时步 $= 1, t$ **do**
    根据$\boldsymbol{a}_t \sim \pi_\phi\left(\boldsymbol{a}_t \mid \mathbf{s}_t\right)$采样动作$a_t$
    环境反馈奖励和下一个状态，$\mathbf{s}_{t+1} \sim p\left(\mathbf{s}_{t+1} \mid \mathbf{s}_t, \mathbf{a}_t\right)$
    存储transition到经验回放中，$\mathcal{D} \leftarrow \mathcal{D} \cup\left\{\left(\mathbf{s}_t, \mathbf{a}_t, r\left(\mathbf{s}_t, \mathbf{a}_t\right), \mathbf{s}_{t+1}\right)\right\}$
    更新环境状态$s_{t+1} \leftarrow s_t$
    更新策略：
    更新Q函数，$\theta_i \leftarrow \theta_i-\lambda_Q \hat{\nabla}_{\theta_i} J_Q\left(\theta_i\right)$ for $i \in\{1,2\}$
    更新策略权重，$\phi \leftarrow \phi-\lambda_\pi \hat{\nabla}_\phi J_\pi(\phi)$
    调整温度因子，$\alpha \leftarrow \alpha-\lambda \hat{\nabla}_\alpha J(\alpha)$
    更新目标网络权重，$\bar{\theta}_i \leftarrow \tau \theta_i+(1-\tau) \bar{\theta}_i$ for $i \in\{1,2\}$
  end for
end for

18 GAIL算法

GAIL算法[1]

采样专家轨迹$\tau_E \sim \pi_E$，初始化网络模型参数θ_0和判别器D参数ω_0
for 回合数 $i = 1, 2, \cdots$ **do**
  采样策略轨迹$\tau_i \sim \pi_{\theta_i}$
  使用梯度下降更新判别器D的参数ω_i，梯度为：

$$\hat{\mathbb{E}}_{\tau_i}\left[\nabla_w \log\left(D_w(s,a)\right)\right] + \hat{\mathbb{E}}_{\tau_E}\left[\nabla_w \log\left(1 - D_w(s,a)\right)\right] \tag{1}$$

  使用判别器D对策略轨迹τ_i的输出作为奖励更新策略π_{θ_i}[2]
end for

[1] Generative Adversarial Imitation Learning

[2] 策略更新方式与策略模型π_θ有关，如PP0-Clip等.

19 MAPPO算法

MAPPO算法

初始化每个智能体u的Critic网络Q_{ϕ^u}和参数为θ^u的Actor网络，$u \in U$
初始化每个智能体u的目标Actor网络π^u_{old}的参数$\theta^u_{old} \leftarrow \theta^u$和目标Critic网络$Q_{\overline{\phi}^u}$的参数$\overline{\phi}^u \leftarrow \phi^u$
初始化epoch数K
初始化经验回放D
for 回合数 $= 1, 2, \cdots, M$ **do**
    初始化状态s_1
    每个智能体u都根据各自策略采样C个时步数据,收集轨迹$\tau^u = \{o^u_t, a^u_t, r_{t+1}\}^T_{t=1}$
    对每个时步的每条轨迹
    计算折扣奖励$\{\hat{R}^u_t\}^T_{t=1}$
    计算优势函数$\{A^u_t = V^u_{\phi_t} - \hat{R}^u_t\}^T_{t=1}$
    计算$y^u_t = V^u_{\phi_t} + A^u_t$
    将每个时步的数据$\{[o^u_t, a^u_t, y^u_t, A^u_t]^U_{u=1}\}^T_{t=1}$ 都存储到经验回放D中
    for epoch数 $k = 1, 2, \cdots, K$ **do**
        打乱D中数据顺序并重新编号
        for $j = 0, 1, \cdots, \frac{T}{B} - 1$ **do**
            选择B条数据$\{o^u_i, a^u_i, y^u_i, A^u_i\}^{B(j+1)}_{i=1+Bj}$
            计算梯度：
            $\triangle\theta^u = \frac{1}{B}\sum^B_{i=1}\{\bigtriangledown_{\theta^u} f(r_i(\theta^u), A^u_i)\}$
            $\triangle\phi^u = \frac{1}{B}\sum^B_{i=1}\{\bigtriangledown_{\phi^u}(y^u_i - V_{\phi^u}(o^u_i))^2\}$
            Adam梯度上升方法计算θ^u，Adam梯度下降方法计算ϕ^u
        end for
    end for
    更新$\theta^u_{old} \leftarrow \theta^u$, $\overline{\phi}^u \leftarrow \phi^u$
end for