强化学习2022
第9节

涉及知识点：
模型预测控制、基于模型的策略优化

基于模型的深度强化学习

张伟楠 – 上海交通大学

课程大纲

强化学习基础部分

1. 强化学习、探索与利用
2. MDP和动态规划
3. 值函数估计
4. 无模型控制方法
5. 规划与学习
6. 参数化的值函数和策略
7. 深度强化学习价值方法
8. 深度强化学习策略方法

强化学习前沿部分

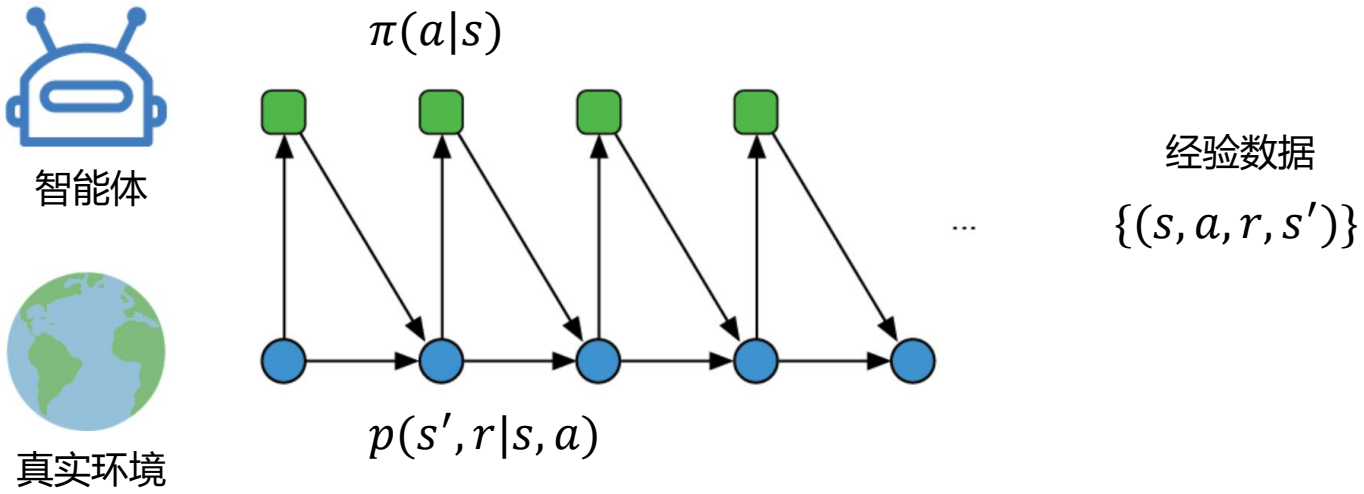
9. 基于模型的深度强化学习
10. 模仿学习
11. 离线强化学习
12. 参数化动作空间
13. 目标导向的强化学习
14. 多智能体强化学习
15. 强化学习大模型
16. 技术交流与回顾

模型预测控制

张伟楠 – 上海交通大学

基于模型的强化学习

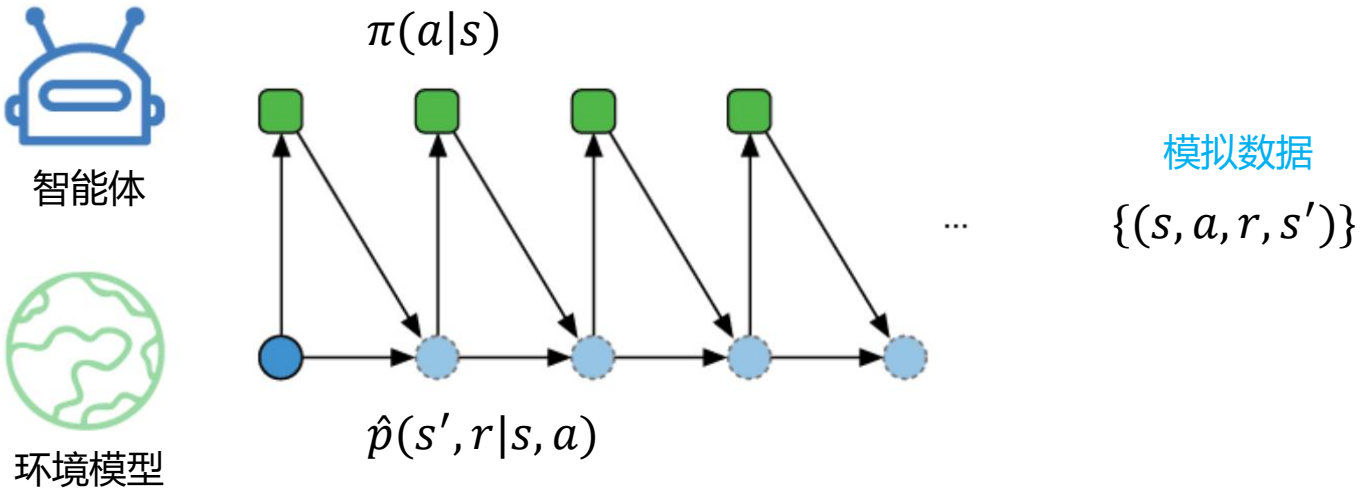
- 智能体和动态环境做交互的过程，可以使用一个环境模型（即一个模拟器）来替代



- 真实环境
  - 状态转移动态 $p(s'|s, a)$
  - 奖励函数 $r(s, a)$

基于模型的强化学习

- 智能体和动态环境做交互的过程，可以使用一个环境模型（即一个模拟器）来替代

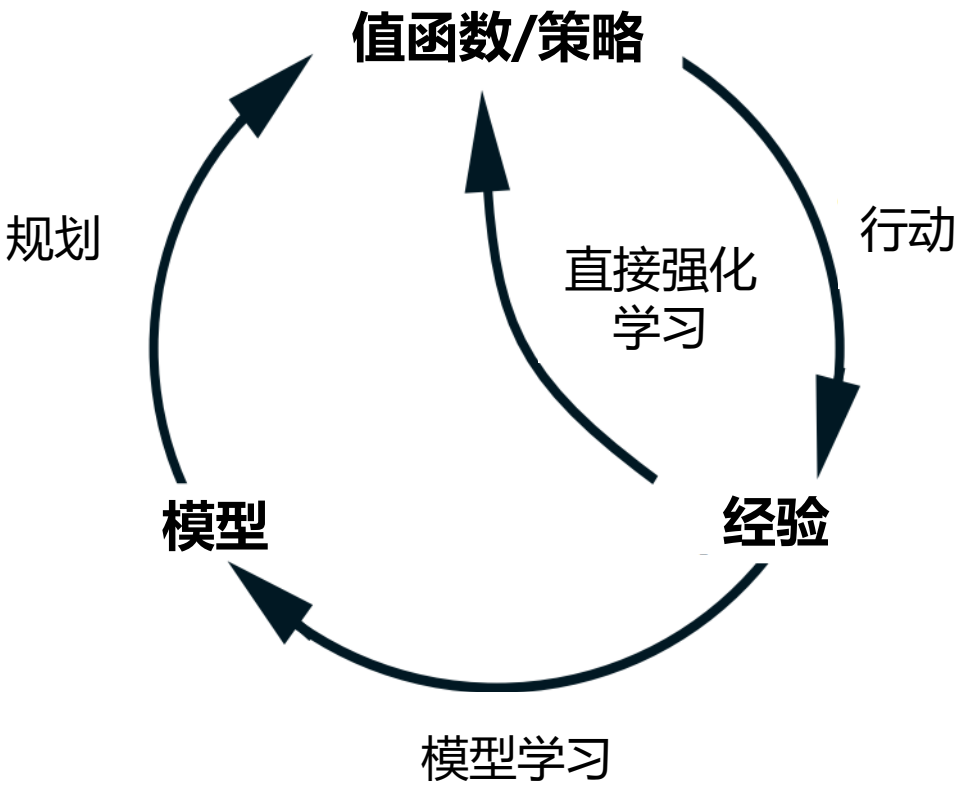


- 真实环境
  - 状态转移动态 $p(s'|s,a)$
  - 奖励函数 $r(s,a)$
- 环境模型
  - 状态转移动态 $\hat{p}(s'|s,a)$
  - 奖励函数 $\hat{r}(s,a)$

复习：Dyna (集成规划、决策和学习)

经验的不同用途

- 更新模型
  - 模型学习, 或间接强化学习
  - 对经验数据的需求少
- 更新值函数和策略
  - 直接强化学习（无模型强化学习）
  - 简单且不受模型偏差的影响

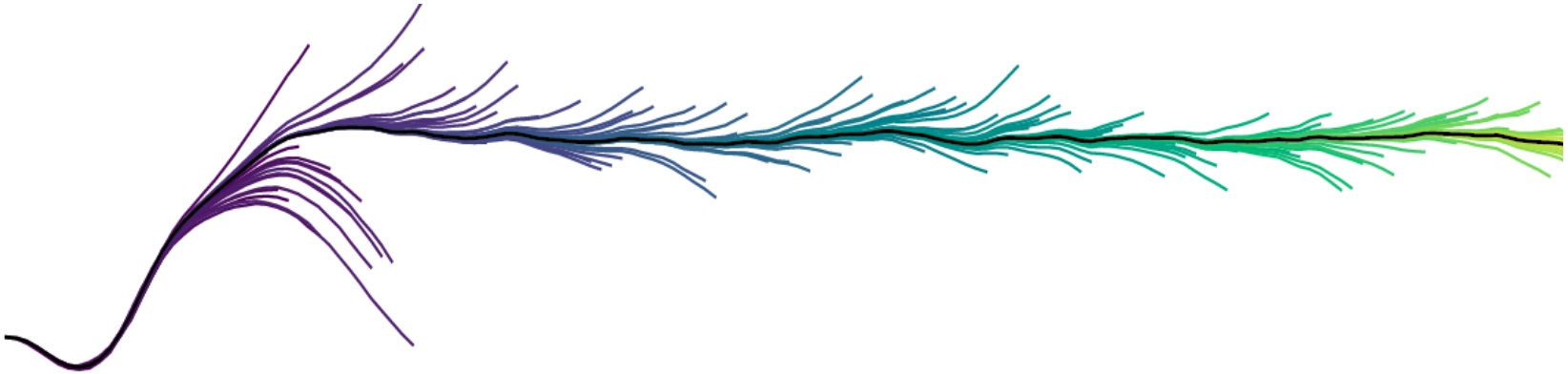


是否有一种方法不用学习值函数和策略，直接基于模型就可以工作呢？- **模型预测控制**

模型预测控制

基于模型的规划

- 模型预测控制MPC：在每个时间步，MPC算法采样多条轨迹，从这些轨迹中得到经验最优轨迹，然后取该轨迹的第一个动作来执行，如此往复，指导任务介绍。



模型预测控制方法不用学习值函数和策略，
直接基于模型就可以工作

目录

Contents

01

打靶法

打靶法

- 一种不用构建智能体的计算模块（值函数和策略），直接使用环境模型来规划下一个动作的方法
- 从当前状态s_0出发
- 对于每个候选动作a_0，附上一个长度为T的随机动作序列，得到

$$[a_0, a_1, a_2, \dots, a_T]$$

- 可以通过此动作序列与环境模型交互出一条轨迹

$$[s_0, a_0, \hat{r}_0, \hat{s}_1, a_1, \hat{r}_1, \hat{s}_2, a_2, \hat{r}_2, \dots, \hat{s}_T, a_T, \hat{r}_T]$$

- 基于此采样轨迹，可以对(s_0, a_0)计算得到一个经验性的价值

$$\hat{Q}(s_0, a_0) = \sum_{t=0}^{T} \gamma^t \hat{r}_t$$

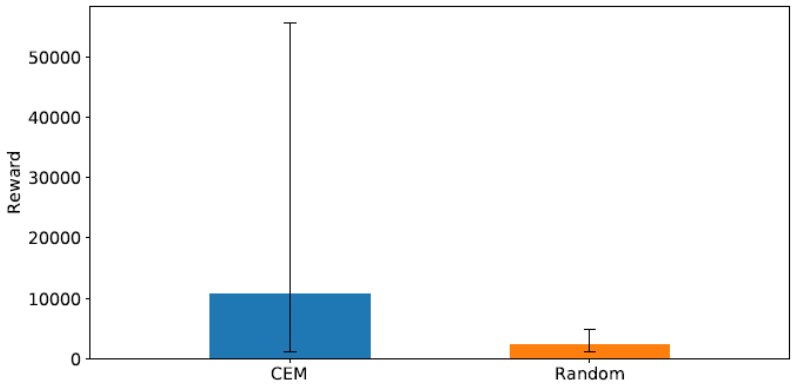
- 以上操作对每个候选动作计算一次，选择一个最高经验性价值的动作来执行

$$\pi(s) = \text{argmax}_a \ \hat{Q}(s_0, a)$$

打靶法

- 注：打靶法中每次动作序列$[a_0, a_1, a_2, \ldots, a_T]$是随机采样的
- 优点：
  - 实现十分简单
  - 无训练开销（例如没有梯度计算）
  - 不需要提前设置任务具体的时间步长度
- 缺点：很高的不确定性（方差大），可能无法选择到最优的动作

- 一种改进方法：交叉熵方法（Cross Entropy Method, CEM）
  - 交叉熵方法维护一个之前带来较高奖励值的动作分布，其打靶法中的随机动作序列从该动作分布中采样得到



02
PETS算法

PETS算法

- PETS算法全称为probabilistic ensembles with trajectory sampling，是使用MPC的一种基于模型的强化学习算法
- 集成中每个单模型是一个神经网络构建的高斯过程

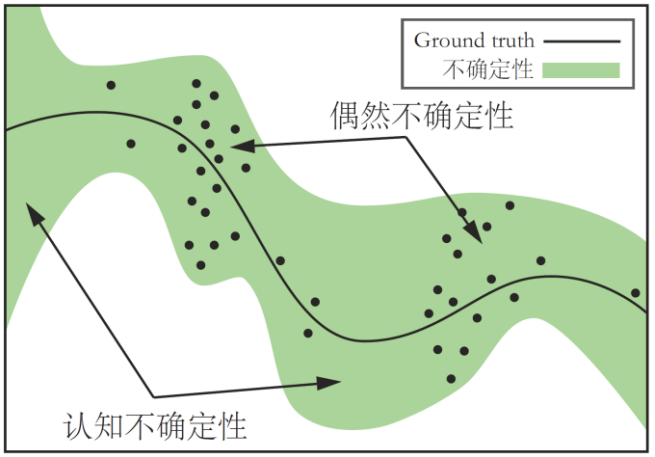
高斯过程 $\widetilde{f} = \Pr(\boldsymbol{s}_{t+1}|\boldsymbol{s}_t, \boldsymbol{a}_t) = \mathcal{N}\left(\boldsymbol{\mu}_\theta(\boldsymbol{s}_t, \boldsymbol{a}_t), \boldsymbol{\Sigma}_\theta(\boldsymbol{s}_t, \boldsymbol{a}_t)\right)$

集成损失 $\text{loss}_{\text{P}}(\boldsymbol{\theta}) = -\sum_{n=1}^{N} \log \widetilde{f}_{\boldsymbol{\theta}}(\boldsymbol{s}_{n+1}|\boldsymbol{s}_n, \boldsymbol{a}_n)$

- PE代表环境模型的概率集成，以刻画模型对环境的两种不确定性

1. 不同单模型之间的不一致性刻画出整个PE在数据缺少的区域的认知不确定性（epistemic uncertainty）

2. 每一个高斯过程建模真实随机环境的偶然不确定性（aleatoric uncertainty）



Chua, Kurtland, et al. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. NIPS 2018.

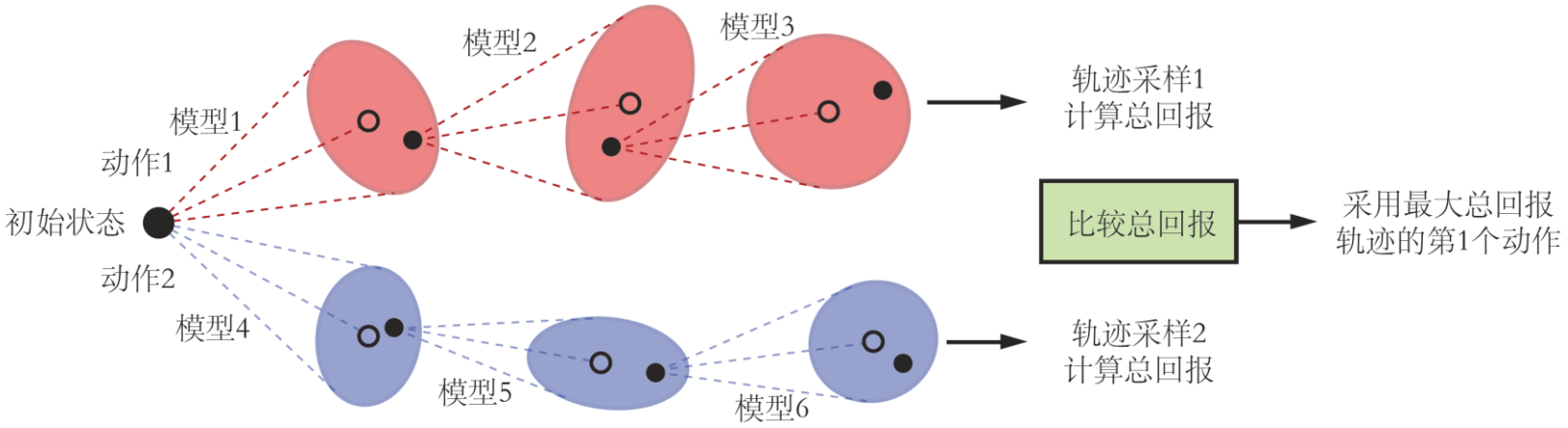
PETS算法

- PETS算法全称为probabilistic ensembles with trajectory sampling，是使用MPC的一种基于模型的强化学习算法
- 集成中每个单模型是一个神经网络构建的高斯过程

高斯过程 $\widetilde{f} = \Pr(\boldsymbol{s}_{t+1}|\boldsymbol{s}_t, \boldsymbol{a}_t) = \mathcal{N}\left(\boldsymbol{\mu}_\theta(\boldsymbol{s}_t, \boldsymbol{a}_t), \boldsymbol{\Sigma}_\theta(\boldsymbol{s}_t, \boldsymbol{a}_t)\right)$

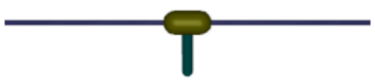
集成损失 $\text{loss}_{\text{P}}(\boldsymbol{\theta}) = -\sum_{n=1}^{N} \log \widetilde{f}_{\boldsymbol{\theta}}(\boldsymbol{s}_{n+1}|\boldsymbol{s}_n, \boldsymbol{a}_n)$

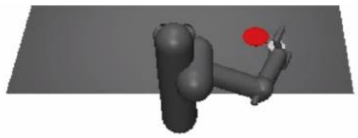
- TS代表集成环境模型的轨迹采样，每一次预测我们会从N个模型中挑选一个来进行预测，因此一条轨迹的采样会使用到多个环境模型

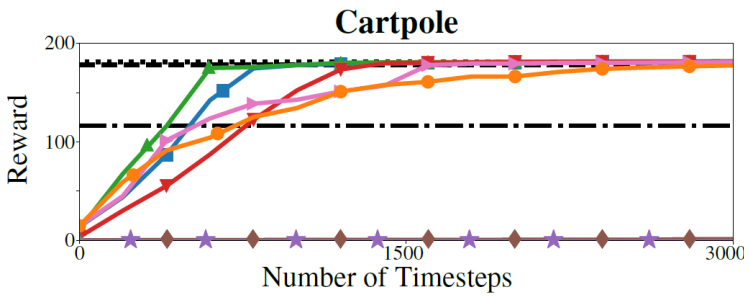


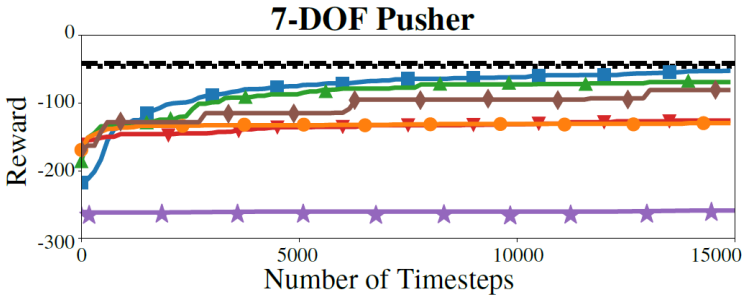
Chua, Kurtland, et al. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. NIPS 2018.

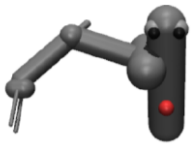
PETS算法的实验

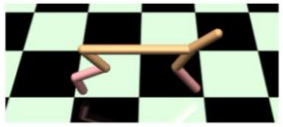


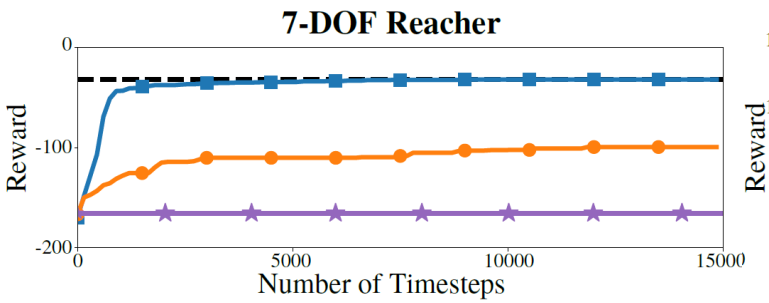


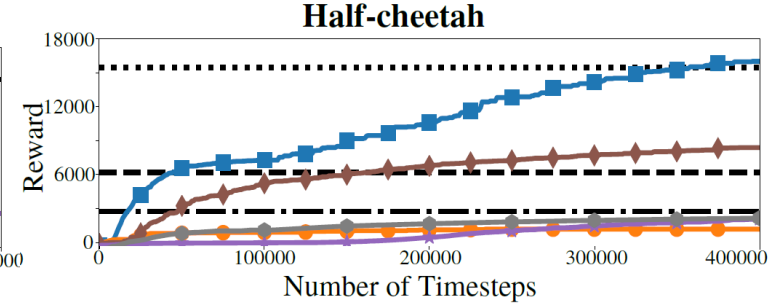












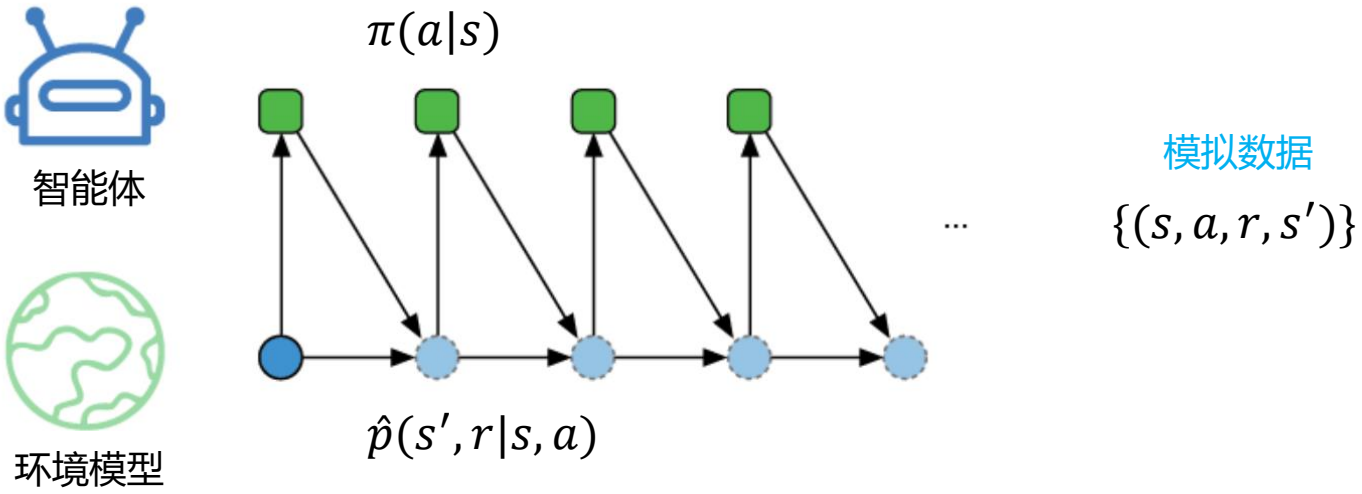
Chua, Kurtland, et al. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. NIPS 2018.

基于模型的策略优化

张伟楠 – 上海交通大学

复习：基于模型的强化学习

- 智能体和动态环境做交互的过程，可以使用一个环境模型（即一个模拟器）来替代



- 真实环境
  - 状态转移动态 $p(s'|s, a)$
  - 奖励函数 $r(s, a)$
- 环境模型
  - 状态转移动态 $\hat{p}(s'|s, a)$
  - 奖励函数 $\hat{r}(s, a)$

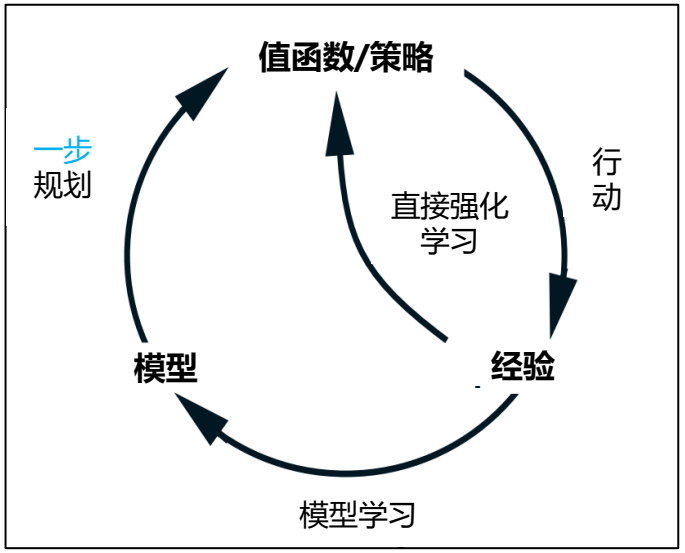
复习：Dyna (集成规划、决策和学习)

算法: 表格 $\mathbf{Dyna-Q}$

对于所有的 $s \in \mathcal{S}$ 和 $a \in \mathcal{A}(S)$，初始化值函数 $Q(s,a)$ 和模型 $Model(s,a)$

重复以下步骤:

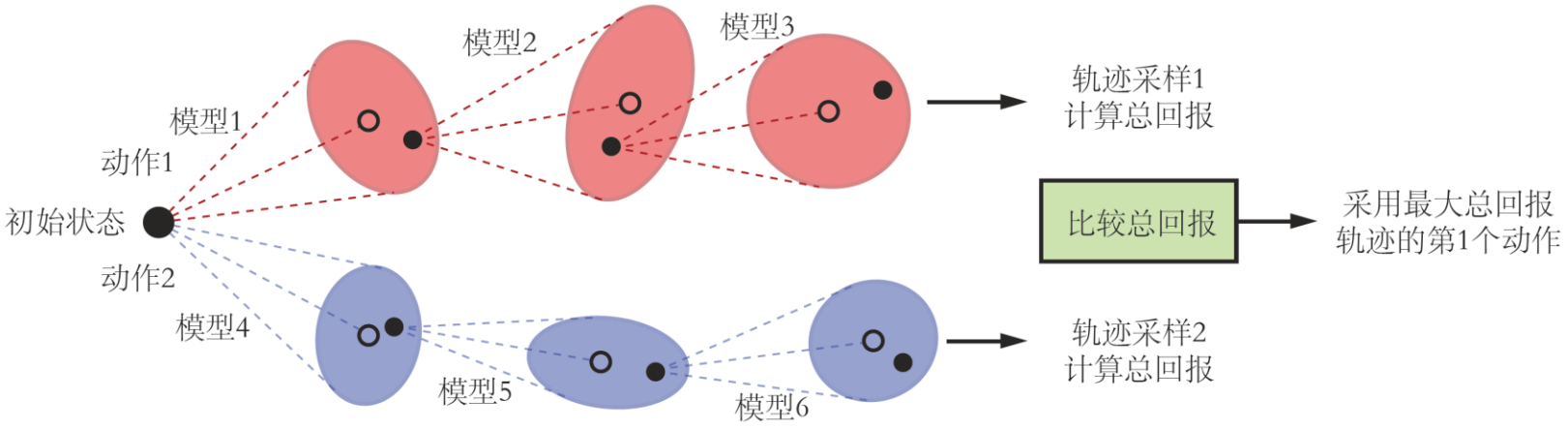
1. 令 $S \leftarrow$ 当前（非终止）状态
2. 令 $A \leftarrow \epsilon$-greedy(S,Q)
3. 做动作 A；得到奖励 R 和状态 S'
4. 令 $Q(S,A) \leftarrow Q(S,A) + \alpha\left[R + \gamma \max_a Q(S',a) - Q(S,A)\right]$
5. 令$Model(S,A) \leftarrow R, S'$（假设是确定性环境）
6. 重复以下步骤n次:
   a. 令 $S \leftarrow$ 随机采样之前见过的状态
   b. 令 $A \leftarrow$ 随机采样之前在状态S做过的动作
   c. 令 $R, S' \leftarrow Model(S,A)$
   d. $Q(S,A) \leftarrow Q(S,A) + \alpha\left[R + \gamma \max_a Q(S',a) - Q(S,A)\right]$



复习：PETS算法

- PETS算法全称为probabilistic ensembles with trajectory sampling，是使用MPC的一种基于模型的强化学习算法
- 集成中每个单模型是一个神经网络构建的高斯过程

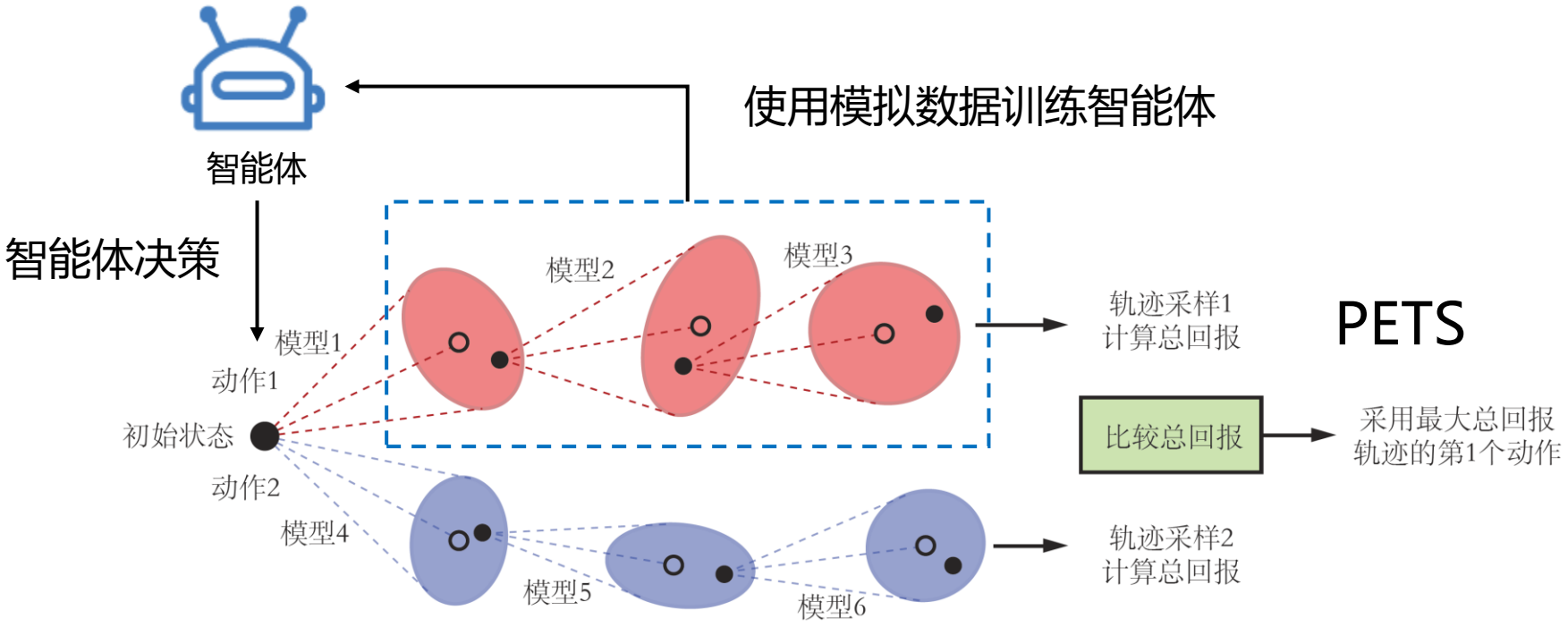
- PE代表环境模型的概率集成，以刻画模型对环境的两种不确定性
- TS代表集成环境模型的轨迹采样，每一次预测我们会从N个模型中挑选一个来进行预测，因此一条轨迹的采样会使用到多个环境模型



Chua, Kurtland, et al. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. NIPS 2018.

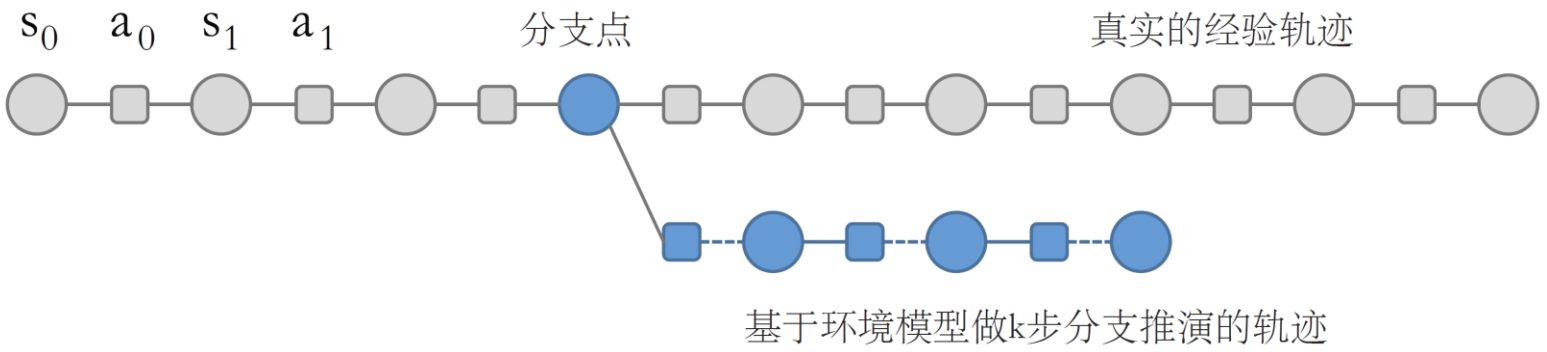
思考问题

- 问题：如果将PETS类的多步模拟数据用于训练智能体策略或价值函数，会怎么样？是否能提升智能体学习的样本效率？



分支推演

- 分支推演（Branched Rollout）
  - 从之前策略采样的真实轨迹中随机选择一个状态作为起点，使用环境模型和智能体当前策略做交互，采样k步得到一条分支轨迹

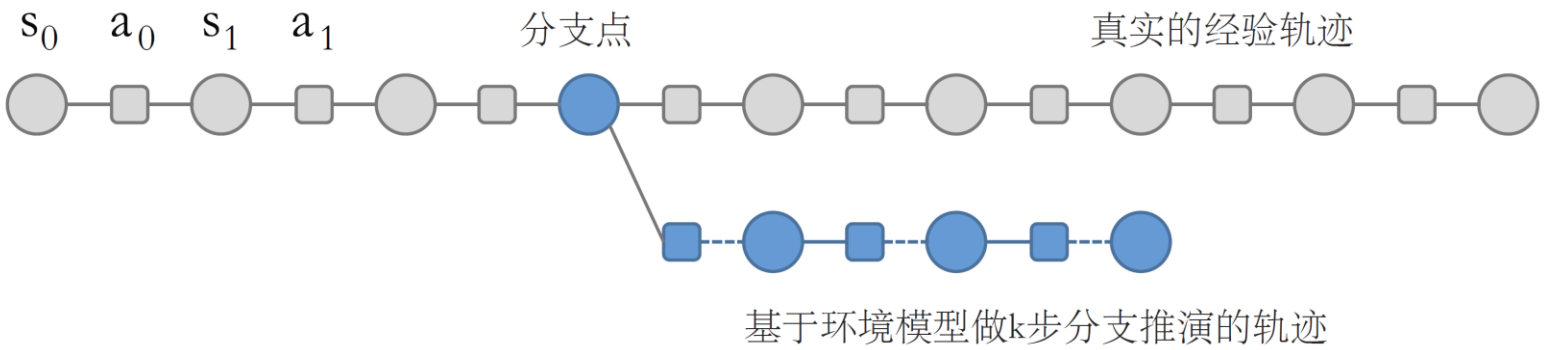


- 使用真实数据和推演数据共同训练智能体的策略
- Dyna算法可以看出是$k = 1$时的分支推演特例

Janner, Michael, et al. When to Trust Your Model: Model-Based Policy Optimization. NIPS 2019.c

分支推演

- 分支推演（Branched Rollout）
  - 从之前策略采样的真实轨迹中随机选择一个状态作为起点，使用环境模型和智能体当前策略做交互，采样k步得到一条分支轨迹



- 使用真实数据和推演数据共同训练智能体的策略
- Dyna算法可以看出是$k = 1$时的分支推演特例
- k越大，推演的数据也不准，价值越低；k越小，算法样本效率越低
- 问题：当环境模型不可避免地有一定不准确度时，如何做好 k 的选择?

Janner, Michael, et al. When to Trust Your Model: Model-Based Policy Optimization. NIPS 2019.

基于模型的强化学习中对策略价值误差的分析

- 策略价值误差 $|\eta[\pi] - \hat{\eta}[\pi]|$

  $\eta[\pi]$：策略在真实环境中的价值　　$\hat{\eta}[\pi]$：策略在环境模型中的价值

- 策略价值误差和算法样本效率之间的关系

策略价值误差$|\eta[\pi] - \hat{\eta}[\pi]|$越低 ➡ 更多使用模型做推演 ➡ 样本效率更高

- 策略价值误差与模型泛化误差以及策略偏移程度之间的定量关系

$$|\eta[\pi] - \hat{\eta}[\pi]| \leq C(\epsilon_m, \epsilon_\pi)$$

ϵ_m：模型泛化误差　　ϵ_π：策略偏移程度

基于模型的强化学习思路：

1. 推导策略价值误差界C
2. 设计算法降低C

Janner, Michael, et al. When to Trust Your Model: Model-Based Policy Optimization. NIPS 2019.

在分支推演对策略价值误差的分析

- 量化模型泛化误差和策略偏移程度为

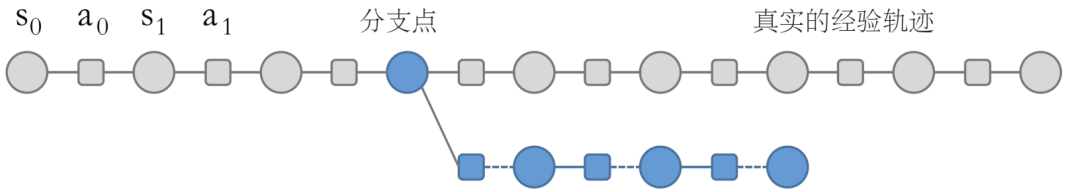
模型泛化误差 $\epsilon_{m'} = \max_t \mathbb{E}_{s\sim\pi_t}[D_{TV}(p(s',r|s,a)\|p_\theta(s',r|s,a))]$

策略偏移程度 $\epsilon_\pi = \max_s D_{TV}(\pi\|\pi_D)$

- 策略价值误差界可写为

$$\eta[\pi] \geq \eta^{\text{branch}}[\pi] - 2r_{\max}\left[\underbrace{\frac{\gamma^{k+1}\epsilon_\pi}{(1-\gamma)^2} + \frac{\gamma^k\epsilon_\pi}{(1-\gamma)}}_{\text{随}k\text{变大而变小}} + \underbrace{\frac{k}{1-\gamma}(\epsilon_{m'})}_{\text{随}k\text{变大而变大}}\right]$$

- 问题：多大的k使得误差界最小？



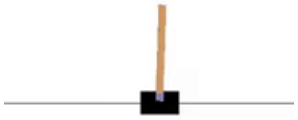
$$\hat{\epsilon}_{m'}(\epsilon_\pi) \approx \epsilon_m + \epsilon_\pi \frac{\mathrm{d}\epsilon_{m'}}{\mathrm{d}\epsilon_\pi}$$

- 推导结论：当比率 $\frac{\mathrm{d}\epsilon_{m'}}{\mathrm{d}\epsilon_\pi}$ 足够小时，k求导极值点为$k > 0$

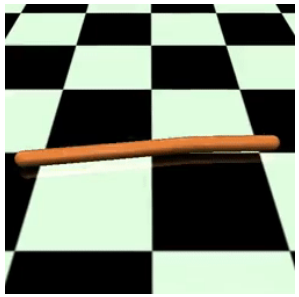
Janner, Michael, et al. When to Trust Your Model: Model-Based Policy Optimization. NIPS 2019.

适合的实验环境

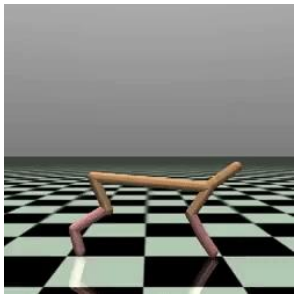
- 推导结论：当比率 $\frac{\mathrm{d}\epsilon_{m'}}{\mathrm{d}\epsilon_{\pi}}$ 足够小时，k求导极值点为$k > 0$
- 基于Gym和Mujoco的机器人控制和运动的环境满足此特性



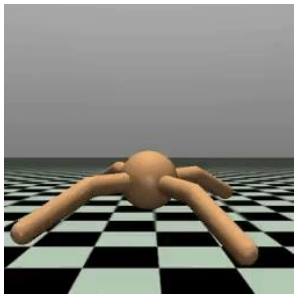
Cartpole



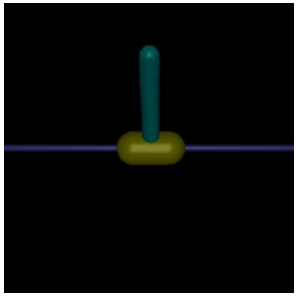
Swimmer



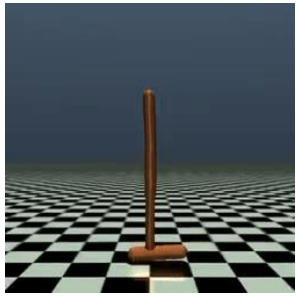
HalfCheetah



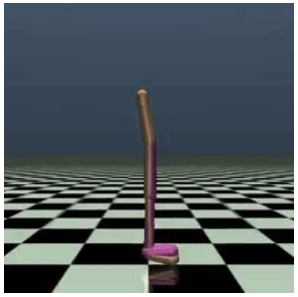
Ant



InvertedPendulum



Hopper



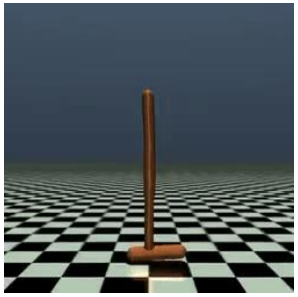
Walker2d

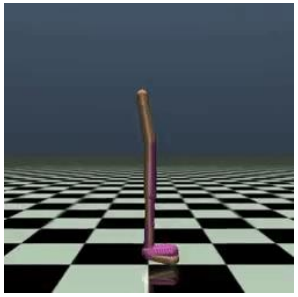
Humanoid

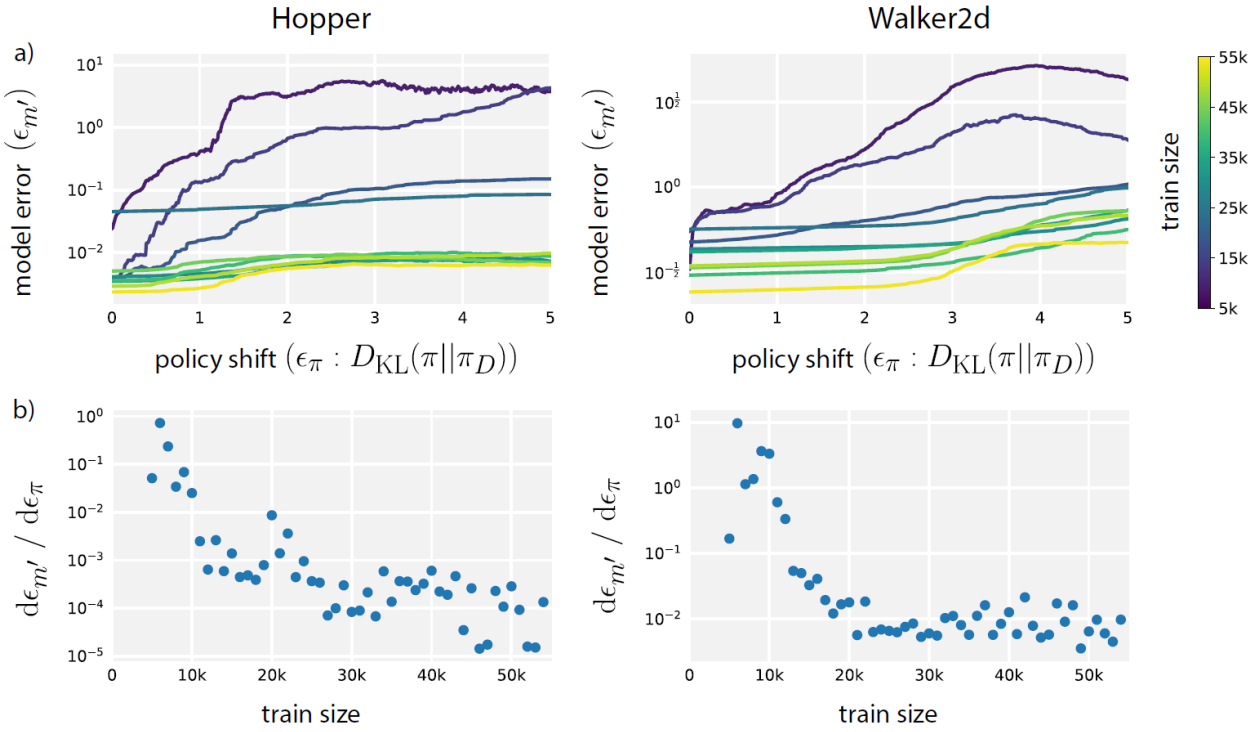
Janner, Michael, et al. When to Trust Your Model: Model-Based Policy Optimization. NIPS 2019.

适合的实验环境

- 当比率 $\frac{\mathrm{d}\epsilon_{m'}}{\mathrm{d}\epsilon_{\pi}}$ 足够小时，k 求导极值点为 $k > 0$







Janner, Michael, et al. When to Trust Your Model: Model-Based Policy Optimization. NIPS 2019.

基于模型的策略优化算法MBPO

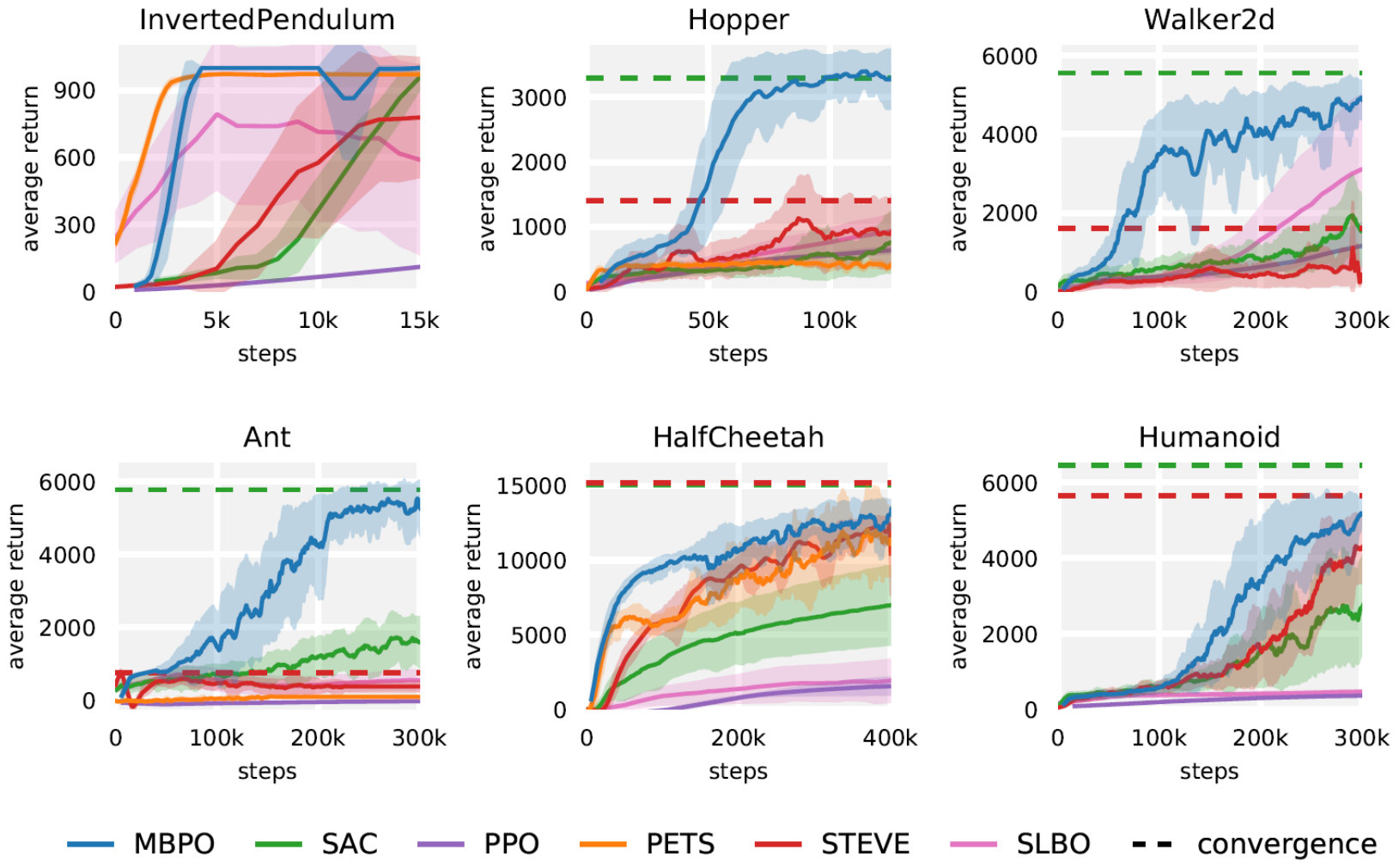
Algorithm 2 Model-Based Policy Optimization with Deep Reinforcement Learning

Initialize policy π_ϕ, predictive model p_θ, environment dataset $\mathcal{D}_{\text{env}}$, model dataset $\mathcal{D}_{\text{model}}$
for N epochs **do**
  Train model p_θ on $\mathcal{D}_{\text{env}}$ via maximum likelihood
  for E steps **do**
    Take action in environment according to π_ϕ; add to $\mathcal{D}_{\text{env}}$
    for M model rollouts **do**
      Sample s_t uniformly from $\mathcal{D}_{\text{env}}$
      Perform k-step model rollout starting from s_t using policy π_ϕ; add to $\mathcal{D}_{\text{model}}$
    for G gradient updates **do**
      Update policy parameters on model data: $\phi \leftarrow \phi - \lambda_\pi \hat{\nabla}_\phi J_\pi(\phi, \mathcal{D}_{\text{model}})$

- MBPO算法要点
  - 分支推演的起点是从真实经验轨迹中采样任何的状态，而不是一定从初始状态开始推演
  - 分支推演的步数k取决于环境模型和策略
  - 得到推演数据后，使用Soft AC算法训练策略

Janner, Michael, et al. When to Trust Your Model: Model-Based Policy Optimization. NIPS 2019.

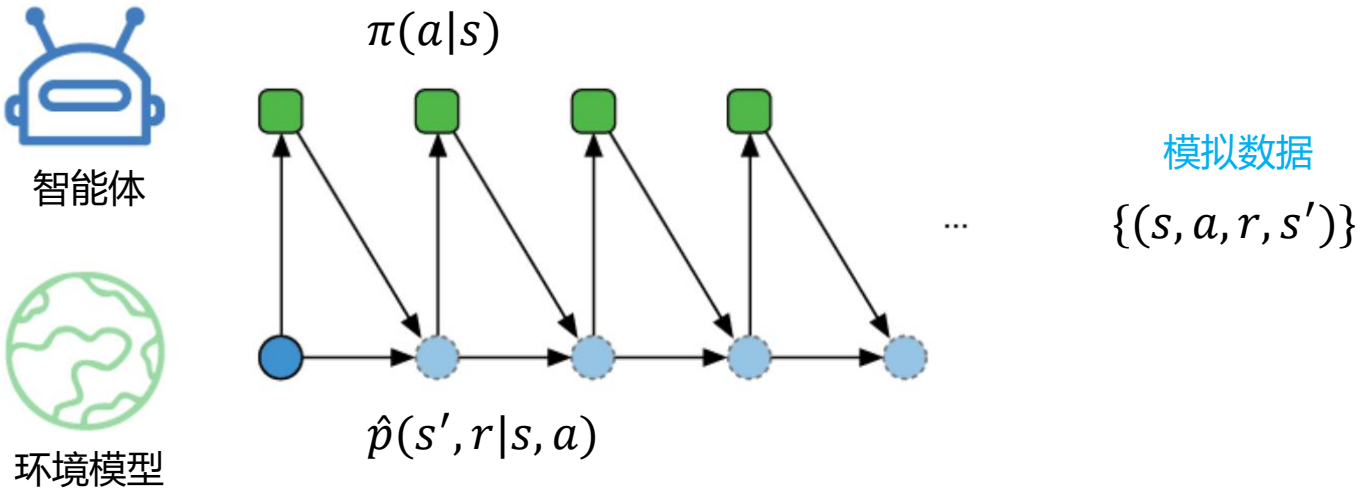
基于模型的策略优化算法MBPO的实验结果



Janner, Michael, et al. When to Trust Your Model: Model-Based Policy Optimization. NIPS 2019.

总结基于模型的强化学习

- 智能体和动态环境做交互的过程，可以使用一个环境模型（即一个模拟器）来替代



- 基于模型的强化学习是提升强化学习样本效率的主流方法
  - 模型预测控制方法：PETS
  - 基于模型的策略优化方法：MBPO

THANK YOU