多智能体系统与强化学习

主讲人：高阳、杨林、杨天培

https://reinforcement-learning-2025.github.io/

第五讲：基于模型的强化学习

模型学习

杨　林

大 纲

基于模型的强化学习介绍

基于模型的值函数优化

基于模型的策略函数优化

大 纲

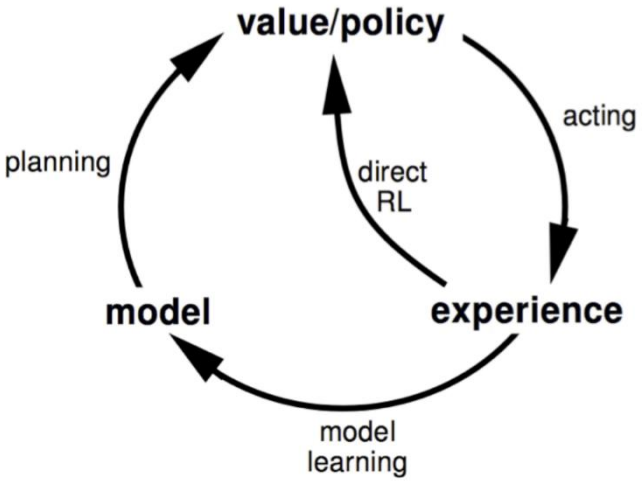
基于模型的强化学习介绍

基于模型的值函数优化

基于模型的策略函数优化

前瞻

- 无模型强化学习(Model-free RL)
  - ✓ 无需事先知道模型
  - ✓ 从经验中学习值/策略函数
- 基于模型强化学习(Model-based RL)
  - ✓ 从经验中学习一个模型
  - ✓ 从模型中规划一个值/策略函数



学习什么样的模型

- 模型 M 表示为一个参数化 η 的MDP
- 通常，对于模型 $M = (\mathcal{P}, \mathcal{R})$ ，其状态转换和奖励表示为

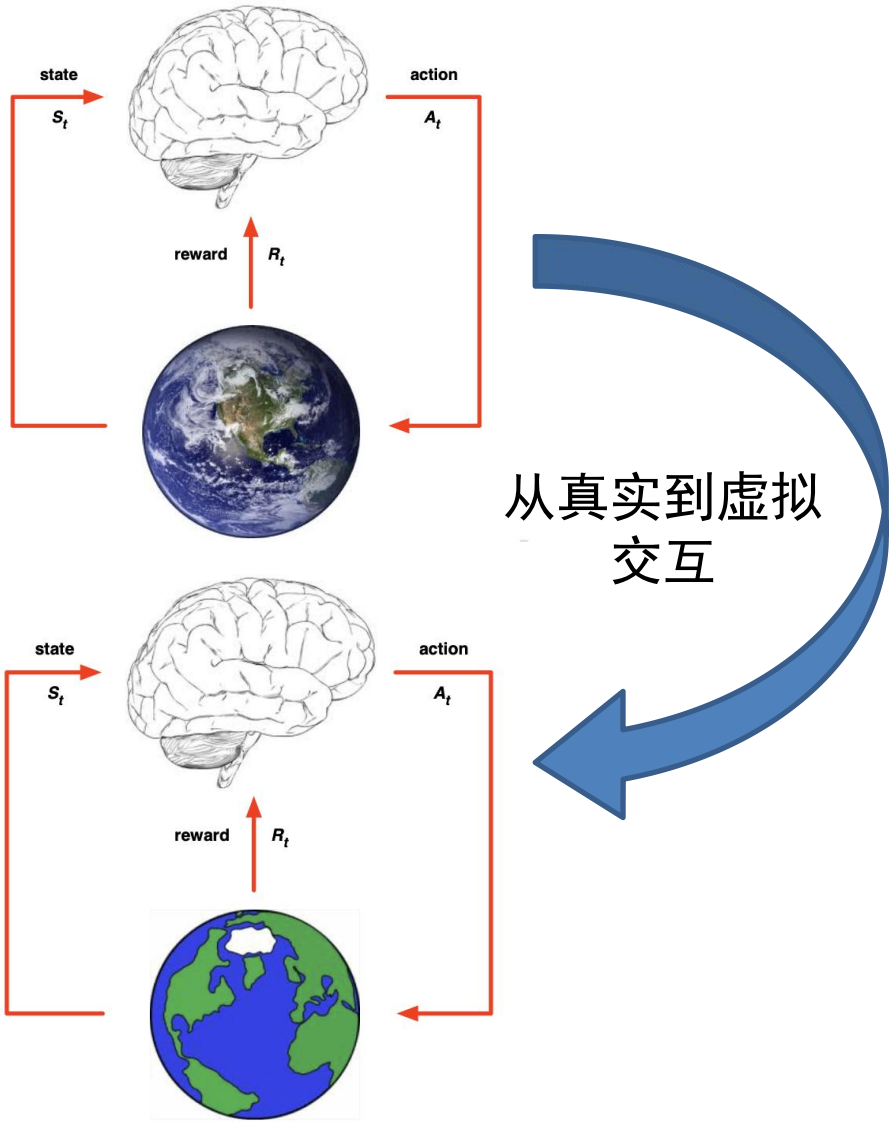
$$S_{t+1} \sim \mathcal{P}_\eta(S_{t+1}|S_t, A_t)$$
$$R_{t+1} = \mathcal{R}_\eta(R_{t+1}|S_t, A_t)$$

- 通常，我们假设状态转换和奖励之间具有有条件的独立性

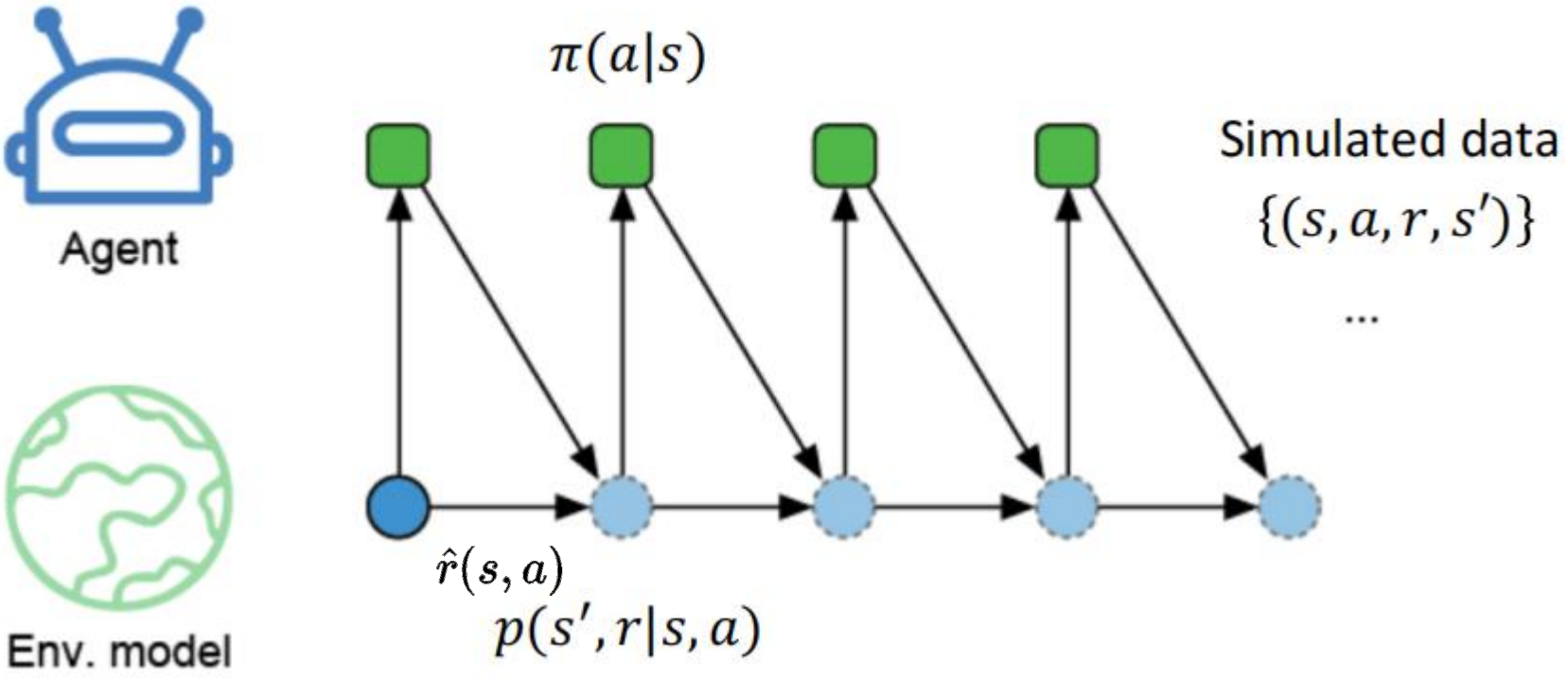
$$\mathcal{P}(S_{t+1}, R_{t+1}|S_t, A_t) = \mathcal{P}(S_{t+1}|S_t, A_t)\mathcal{P}(R_{t+1}|S_t, A_t)$$

智能体与环境的交互

- 真实环境
  - ✓ 状态动态 $p\left(s' \mid s, a\right)$
  - ✓ 奖励函数 $r(s, a)$
- 环境模型
  - ✓ 状态动态 $\hat{p}\left(s' \mid s, a\right)$
  - ✓ 奖励函数 $\hat{r}(s, a)$



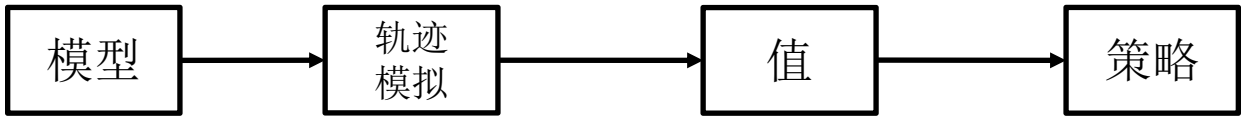
智能体与环境的交互



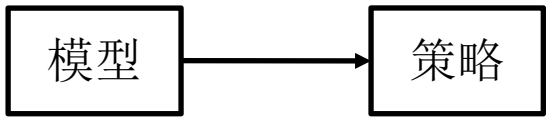
为了规划而对环境建模

- 规划是将模型作为输入，并通过与建模环境交互来生成或改进策略的计算过程

- 基于模型的值优化方法



- 基于模型的策略优化方法



大 纲

基于模型的强化学习介绍

基于模型的值函数优化

基于模型的策略函数优化

模型学习

- 目标：从经验 $\{S_1, A_1, R_1, \cdots, R_T\}$ 处学习模型 M_η
- 所以把它看作是一个监督学习问题

$$S_1, A_1 \rightarrow R_2, S_2$$
$$S_1, A_1 \rightarrow R_2, S_2$$
$$\vdots$$
$$S_1, A_1 \rightarrow R_2, S_2$$

- 学习 $s, a \rightarrow r$ 是一个回归问题
- 学习 $s, a \rightarrow s'$ 是一个概率密度估计问题
- 优化过程：
  - ✓ 选择一个损失函数，例如，均方误差，KL散度
  - ✓ 优化η，以最小化经验损失

查找表模型

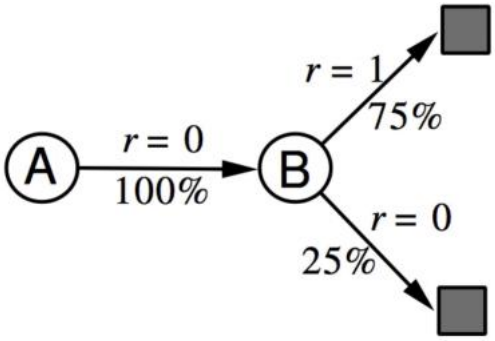
- 模型是一个显式的MDP，$\hat{\mathcal{P}}$ 和 $\hat{\mathcal{R}}$
- 对每个状态动作的对进行计数 $N(s,a)$ 访问

$$\hat{\mathcal{P}}^a_{s,s'} = \frac{1}{N(s,a)} \sum_{t=1}^{T} \mathbf{1}(S_t = s, A_t = a, S_{t+1} = s')$$

$$\hat{\mathcal{R}}^a_s = \frac{1}{N(s,a)} \sum_{t=1}^{T} \sum_{t=1}^{T} \mathbf{1}(S_t = s, A_t = a) R_t$$

一个只包含两状态的例子

- 两个状态A和B；没有折扣
- 观察了8次片段的经验
  - (状态，奖励，下一个状态，下一个奖励……)
  - (A, 0, B, 0), (B, 1), (B, 1), (B, 1), (B, 1), (B, 1), (B, 1), (B, 0)
- 因此，根据经验估计的一个表查找模型如下



基于采样进行规划

- 该模型仅用来生成（模拟）样本
- 一般过程：
  - ✓ 来自该模型的样本经验

$$S_{t+1} \sim \mathcal{P}_\eta(S_{t+1}|S_t, A_t)$$
$$R_{t+1} = R_\eta(R_{t+1}|S_t, A_t)$$

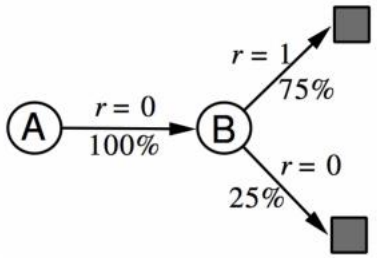
  - ✓ 将无模型RL来处理模拟样本：
    1. 蒙特卡洛方法
    2. 时序差分学习：Sarsa、Q-learning

规划示例

- 真实环境采样：8次片段的经验（状态，奖励，下一个状态…）

  (A, 0, B, 0), (B, 1), (B, 1), (B, 1), (B, 1), (B, 1), (B, 1), (B, 0)

- 构造模型：得到以下环境模型：



- 得到模拟经验：从该模型进行采样获取经验

  (B, 1), (B, 0), (B, 1), (A, 0, B, 1), (B, 1), (A, 0, B, 1), (B, 1), (B, 0)

- 无模型强化学习：基于蒙特卡洛方法的状态价值函数估计

  V(A) = 1, V(B) = 0.75

模型不准确问题

- 采样一般只能得到一个不完美的模型 $\langle \mathcal{P}_\eta, \mathcal{R}_\eta \rangle \neq \langle \mathcal{P}, \mathcal{R} \rangle$
- 基于模型的RL的性能受限于“近似MDP” $\langle S, A, \mathcal{P}_\eta, \mathcal{R}_\eta \rangle$ 的最优策略。当模型不准确时，规划过程将计算出一个次优策略
- 可能的解决方案:
  - ✓ 当模型精度较低时，使用无模型RL
  - ✓ 量化模型的不确定性（我们对状态估计的置信度有多大）：使用概率模型，如贝叶斯过程和高斯过程

关于真实和模拟经验

- 我们现在有两种经验来源

✓ **真实经验**：从环境中取样（真实MDP)

$$S', S \sim \mathcal{P}^a_{s,s'}$$
$$R = \mathcal{R}^a_s$$

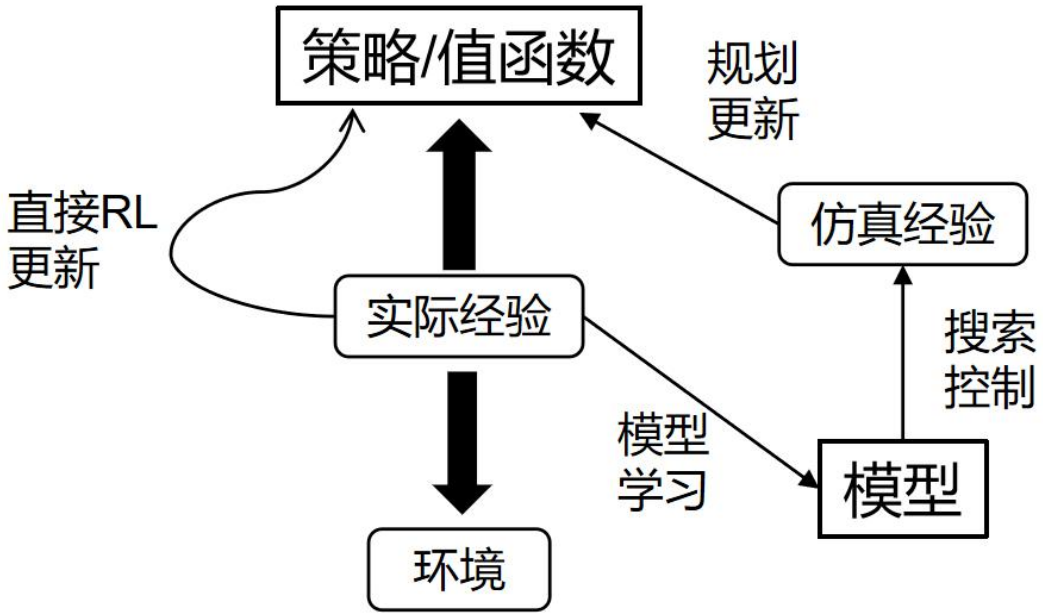
✓ **模拟经验**：从模型中采样（近似MDP)

$$\hat{S}', \hat{S} \sim \mathcal{P}_\eta(S'|S, A)$$
$$\hat{R} = \mathcal{R}_\eta(R|S, A)$$

只使用其中一种会出现哪些问题？

Dyna: 一种经验集成的框架

- Dyna架构



Dyna-Q算法

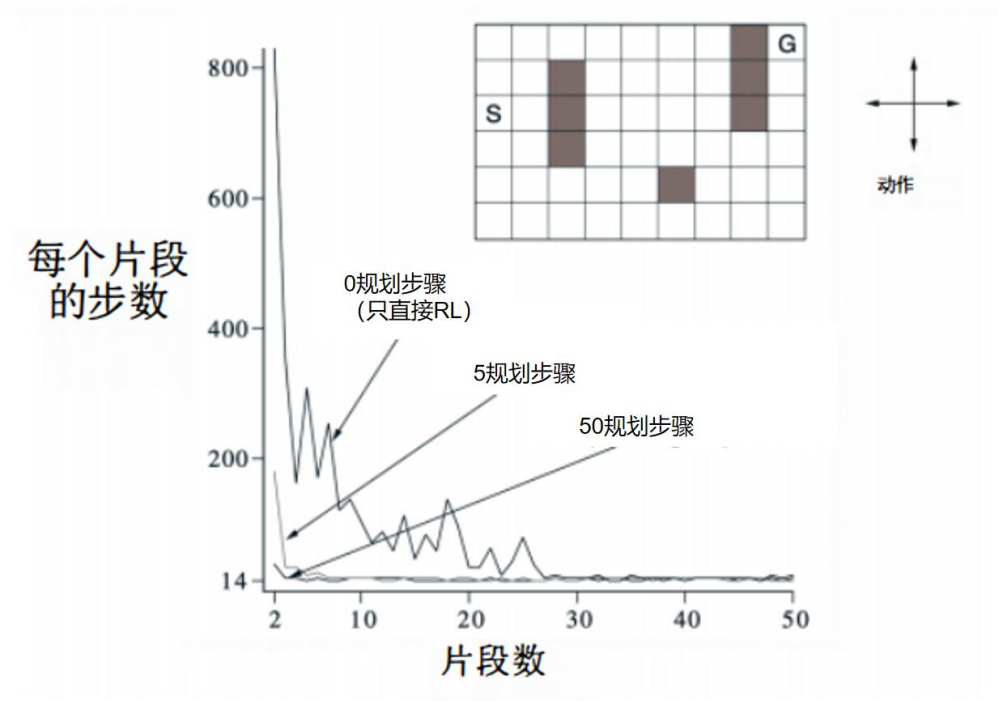
- 结合直接RL、模型学习和规划

Tabular Dyna-Q:
初始化: Q(s, a) 和模型(s, a)
循环：

(a) S ←当前状态
(b) $A \leftarrow \epsilon - \text{greedy}(S, Q)$
(c) 执行动作A，观测结果奖励R和状态S
(d) $Q(S, A) \leftarrow Q(S, A) + \alpha[R + \gamma \max_a Q(S', A) - Q(S, A)]$
(e) 模型$(S, A) \leftarrow R, S'$(假设确定性环境)
(f) 重复N次
（1）S ← 之前观测的随机状态
（2）A ←之前执行的随机动作
（3）R, S' ←模型(S,A)
（4）$Q(S, A) \leftarrow Q(S, A) + \alpha[R + \gamma \max_a Q(S', A) - Q(S, A)]$

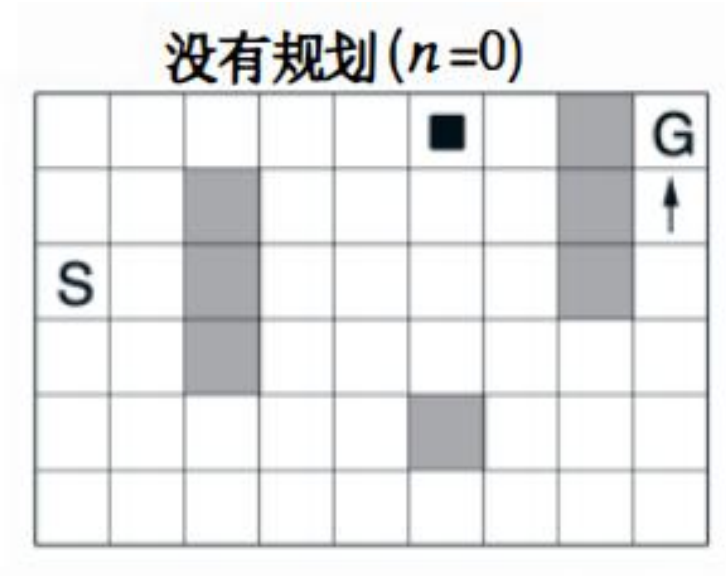
Dyna的实验结果（1）

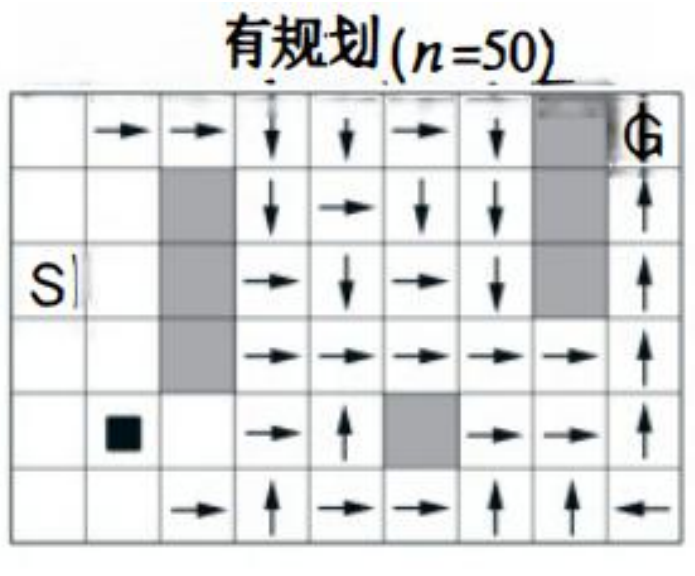
- 一个简单的迷宫环境：尽快从S走到G
- 学习曲线随着规划步骤数而改变：



Dyna的实验结果（2）

- 由基于规划和非规划的Dyna-Q算法在第二幕时发现的策略（无箭头代表等概率动作）：





总结：MBRL的优势

- 无模型强化学习的数据样本效率非常低
  - ✓ 依赖试错学习
- 不支持多步预测和长期规划
  - ✓ 棋类游戏中走子模拟，优化全局策略
- 基于模型的强化学习（Model-based RL）
  - ✓ 样本效率高
  - ✓ 支持长期规划
  - ✓ 但可能会引入模型误差

大 纲

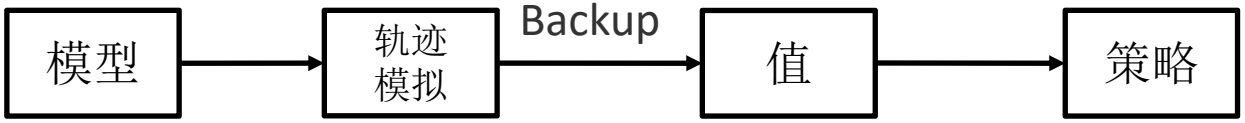
基于模型的强化学习介绍

基于模型的值函数优化

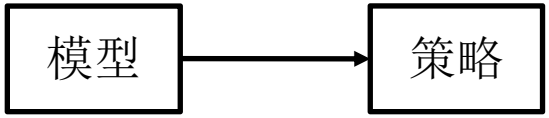
基于模型的策略优化

使用基于模型的RL进行策略优化

- 以前的基于模型的基于值的RL：



- 我们是否可以优化策略并直接学习模型，而不是估计价值？



RL中基于模型的策略优化

- 在策略梯度中，作为一个无模型的RL，只关心策略 $\pi_\theta(a_t|s_t)$ 和预期回报

$$\tau = \{s_1, a_1, s_2, a_2 \ldots, s_T, a_T\} \sim \pi_\theta(a_t|s_t)$$

$$\arg\max_\theta \mathbb{E}_{\tau\sim\pi_\theta}\left[\sum_t \gamma^t r(s_t, a_t)\right]$$

- 在计算策略梯度中，不需要 $p(s_{t+1}|s_t, a_t)$（无论它是已知的或未知的）

$$p(s_1, a_1, \ldots, s_t, a_T) = p(s_1)\prod_{t=1}^{T}\pi_\theta(a_t|s_t)p(s_{t+1}|s_t, a_t)$$

如果我们知道模型或者能够学习模型，我们能做得更好吗？

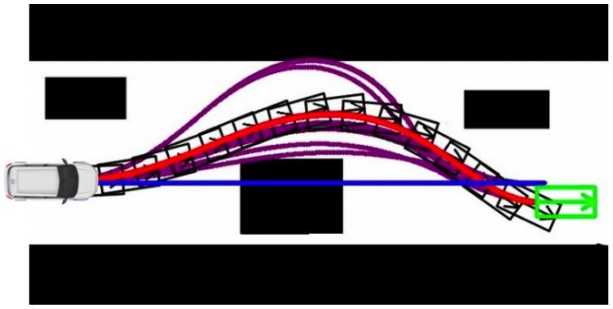
基于模型的策略优化

- RL中基于模型的策略优化类似于最优控制问题
- 控制理论使用系统动力学方程 $s_t = f(s_{t-1}, a_{t-1})$ 来描述系统状态的迁移，“最优控制”最小化价值函数：

$$\arg\min_{a_1,\ldots,a_T} \sum_{t=1}^{T} c(s_t, a_t) \text{ subject to } s_t = f(s_{t-1}, a_{t-1})$$

（其中，状态转移方程作为优化问题的条件来使用）

轨迹优化的最优控制



$$\min_{a_1,\ldots,a_T} \sum_{t=1}^{T} c(s_t, a_t) \text{ subject to } s_t = f(s_{t-1}, a_{t-1})$$

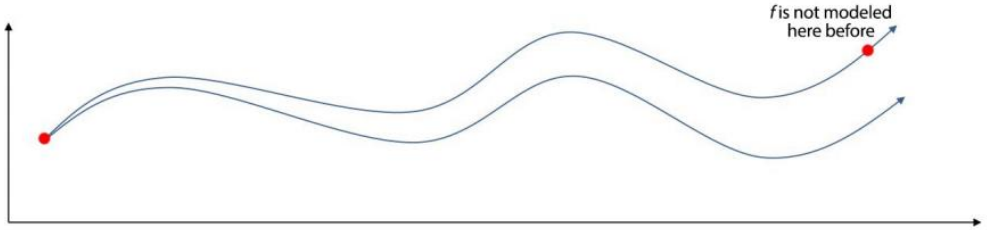
- 如果动力学方程已知，上述问题是一个最优控制问题
- 代价函数是RL问题的负回报
- 在一些简化的假设下，最优解可以通过优化方法求解

轨迹优化的模型学习：算法1

- 如果动力学模型未知，我们可以将模型学习和轨迹优化相结合
- 算法步骤：
  - ✓ 步骤一：运行策略 $\pi_0(a_t|s_t)$（随机策略）来收集数据 $\mathcal{D} = \{(s, a, s')_i\}$
  - ✓ 步骤二：学习动力学模型$s' = f(s, a)$以最小化 $\sum_i ||f(s_i, a_i) - s'_i||^2$
  - ✓ 步骤三：通过$f(s, a)$进行规划来优化行动选择
- 步骤二采用监督学习来训练一个模型，以最小化来自采样数据的最小二乘误差
- 步骤三利用模型、代价函数和迭代优化算法（基于值或策略）计算出最优轨迹

轨迹优化的模型学习：算法2

- 上一个解决方案容易受到模型误差的影响，一个微小的误差会沿着轨迹快速累积



- 通过迭代来改进算法:
  - ✓ 运行基本策略$\pi_0(a_t|s_t)$（随机策略）来收集 $\mathcal{D} = \{(s, a, s')_i\}$
  - ✓ 循环：
    - ① 学习动态模型$s' = f(s, a)$，以最小化 $\sum_i ||f(s_i, a_i) - s'_i||^2$
    - ② 通过$f(s, a)$进行计划来选择行动
    - ③ 执行这些操作，并将结果数据 $\{(s, a, s')_i\}$添加到 $\mathcal{D}$

轨迹优化的模型学习：算法3

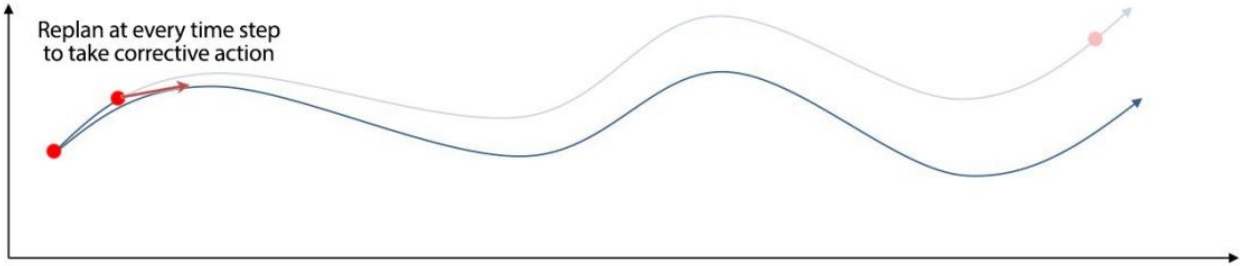
- 然而，上述方法在拟合模型之前执行所有规划的动作。后面的动作可能包含了较多的误差
- 可以使用模型预测控制（Model Predictive Control, MPC）来优化整个轨迹

✓ **预测模型**：基于当前状态和系统动态模型，预测未来一段时间内的系统行为

✓ **优化求解**：在预测时域内，求解满足约束条件的最优控制序列（如最小化能耗或误差）

✓ **滚动执行**：仅执行第一个控制动作，随后重复预测和优化（闭环反馈）

- 在MPC中，我们优化了整个轨迹，但只采取第一个行动。我们再次观察并重新计划。重新规划使我们有机会在再次观察到当前状态后采取纠正行动

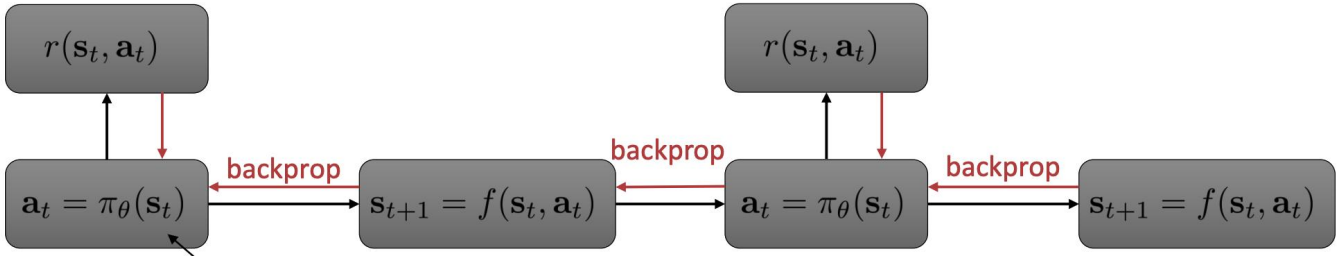


轨迹优化的模型学习：算法3

- 带MPC的算法3：
  - ✓ 运行基本策略$\pi_0(a_t|s_t)$以收集 $\mathcal{D}=\{(s,a,s')_i\}$
  - ✓ N个步骤循环一次
    1. 学习动态模型 $s'=p(s,a)$，以最小化$\sum_i ||f(s_i,a_i)-s'_i||^2$
    2. 循环每个步骤
       ① 通过$f(s,a)$规划选择行动
       ② 执行第一个规划的操作并观察结果状态 s'（MPC）
       ③ 将$\{(s,a,s')_i\}$附加到数据集$\mathcal{D}$

轨迹优化的模型学习：算法4

- 最后，我们可以将策略学习与模型学习相结合



- 算法4：同时学习模型和策略

1. 运行基本策略 $\pi_0(a_t|s_t)$（随机策略）来收集 $\mathcal{D} = \{(s, a, s')_i\}$
2. 循环

① 学习动态模型 $f(s, a)$，以最小化 $\sum_i ||f(s_i, a_i) - s'_i||^2$

② 通过 $f(s, a)$ 反向传播到策略中，以更新优化 $\pi_\theta(a_t|s_t)$

③ 运行 $\pi_\theta(a_t|s_t)$，将所访问的 (s, a, s') 附加到 $\mathcal{D}$

例子：MBPO算法

Algorithm 1 基于模型的策略优化（MBPO）

初始化策略 π_ϕ，预测模型 p_θ，环境数据集 $\mathcal{D}_{\text{env}}$，模型数据集 $\mathcal{D}_{\text{model}}$
for N 轮训练 **do**
  在环境数据集 $\mathcal{D}_{\text{env}}$ 上通过最大似然训练模型 p_θ
  for E 步 **do**
    根据策略 π_ϕ 在环境中执行动作，并添加到 $\mathcal{D}_{\text{env}}$
  end for
  for M 轮模型展开 **do**
    从 $\mathcal{D}_{\text{env}}$ 中均匀采样 s_t
    从 s_t 开始使用策略 π_ϕ 进行 k 步模型展开，并添加到 $\mathcal{D}_{\text{model}}$
  end for
  for G 轮梯度更新 **do**
    在模型数据集上更新策略参数：
    $\phi \leftarrow \phi - \lambda_\pi \hat{\nabla}_\phi J_\pi(\phi, \mathcal{D}_{\text{model}})$
  end for
end for

模型学习

模型展开

策略优化

思考和讨论

1. 理解基于模型/无模型强化学习算法区别
2. 理解MBPO算法
3. 可以通过哪些方式构建模型？

谢 谢 ！