|  |
| --- |
| 首行 |
| Spingup  Introduction to RL |
| 单位：NWPU |
| 时间2018年12月7日星期五 |

目录

[1 Part 1: Key Concepts in RL 2](#_Toc1622)

[1.1 What Can RL Do? 2](#_Toc3651)

[1.2 Key Concepts and Terminology 2](#_Toc30316)

[1.2.1 States and Observations 3](#_Toc4181)

[1.2.2 动作空间 3](#_Toc5342)

[1.3 策略 3](#_Toc29733)

[1.3.1 确定性策略 4](#_Toc9374)

[1.3.2 随机策略 4](#_Toc10616)

[1.3.3 轨迹(Trajectories) 5](#_Toc5260)

[1.4 奖励函数 6](#_Toc28951)

[1.5 值函数 7](#_Toc8719)

[1.6 优势函数 9](#_Toc7048)

[1.7 (Optional) Formalism 10](#_Toc10602)

[2 Part 2: Kinds of RL Algorithms 11](#_Toc29327)

[2.1 A Taxonomy of RL Algorithms（RL算法的分类） 11](#_Toc22963)

[2.1.1 Model-Free vs Model-Based RL 11](#_Toc19392)

[2.2 What to Learn 12](#_Toc29085)

[2.3 What to Learn in Model-Free RL 12](#_Toc29870)

[2.3.1 策略优化与Q-学习之间的权衡 13](#_Toc4977)

[2.4 What to Learn in Model-Based RL 13](#_Toc28023)

[2.5 Links to Algorithms in Taxonomy 14](#_Toc29160)

[3 Part 3: Intro to Policy Optimization 16](#_Toc19642)

[3.1 Deriving(导出) the Simplest Policy Gradient 16](#_Toc25800)

[3.2 Implementing the Simplest Policy Gradient 17](#_Toc16445)

[3.3 Expected Grad-Log-Prob Lemma（预期梯度-Log概率引理） 17](#_Toc11121)

[3.4 不要让过去分散你的注意力 18](#_Toc30670)

[3.5 Implementing Reward-to-Go Policy Gradient 18](#_Toc5304)

[3.6 Baselines in Policy Gradients 19](#_Toc29734)

[3.7 Other Forms of the Policy Gradient 20](#_Toc16072)

[3.8 Recap 21](#_Toc16848)

# Part 1: Key Concepts in RL

欢迎来到我们的介绍强化学习！在这里，我们的目标是让您了解用于讨论该主题的语言和符号，对RL算法的高层次解释(尽管我们基本上避免了它们是如何实现的)，以及一些核心数学中的一些作为算法基础的内容。简而言之，RL是对代理的研究，以及他们是如何通过尝试和错误来学习的。它正式化了这样一种观点，即奖励或惩罚代理人的行为使其更有可能在未来重复或放弃这种行为。

## What Can RL Do?

RL方法最近获得了各种各样的成功。例如，它被用来教计算机在仿真中控制机器人

它也被用来为复杂的战略游戏创造突破性的人工智能，最著名的是Go和Dota，教计算机玩原始像素的Atari游戏，以及训练模拟机器人来遵循人类的指令。

## Key Concepts and Terminology

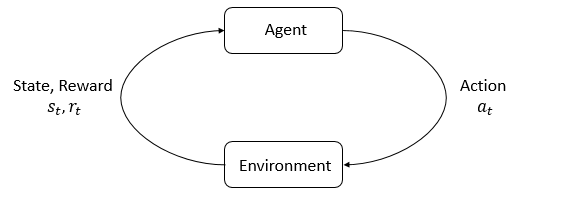


图 1-1

RL的主要特征是代理和环境。环境是代理人生活和相互作用的世界。在交互的每一步，代理都会看到(可能是部分的)对世界状态的观察，然后决定要采取的行动。当代理对其采取行动时，环境会发生变化，但也可能会自行改变。代理还接收来自环境的奖励信号，这个数字告诉它当前世界状态有多好或多坏。代理的目标是最大化其累积报酬，称为回报。强化学习方法是Agent学习行为以达到目标的一种方法。

为了更具体地讨论RL所做的事情，我们需要引入更多的术语。我们得谈谈

* states and observations,状态和观察
* action spaces,行动空间，
* policies,政策，
* trajectories,轨迹，
* different formulations of return,不同的回报公式，
* the RL optimization problem, RL优化问题
* and value functions. 和价值函数。

### States and Observations

A state s is a complete description of the state of the world. There is no information about the world which is hidden from the state. An observation o is a partial description of a state, which may omit information.

In deep RL, we almost always represent states and observations by a real-valued vector, matrix, or higher-order tensor. For instance, a visual observation could be represented by the RGB matrix of its pixel values; the state of a robot might be represented by its joint angles and velocities.

When the agent is able to observe the complete state of the environment, we say that the environment is fully observed. When the agent can only see a partial observation, we say that the environment is partially observed.

一个国家是对世界状况的完整描述。没有任何关于世界的信息是隐藏在国家的。观察o是对一种状态的部分描述，它可以省略信息。

在深度RL中，我们几乎总是用实值向量、矩阵或高阶张量来表示状态和观测.例如，视觉观察可以用它的像素值的RGB矩阵来表示；机器人的状态可以用它的关节角度和速度来表示。当代理能够观察到环境的完全状态时，我们说环境是完全被观察的。当代理人只能看到部分观测时，我们说环境是部分观测的。

您应该知道强化学习表示法有时会将状态符号o放在技术上更适合编写用于观察的符号o的地方。具体来说，在讨论代理如何决定动作时会发生这种情况：我们通常用符号表示动作是以状态为条件的，而在实践中，由于代理没有访问状态的权限，操作是以观察为条件的。在我们的指南中，我们将遵循标准的符号约定，但从上下文中应该可以清楚地看到它的含义。如果有什么不清楚，请提出一个问题！我们的目标是教书，而不是迷惑。

### 动作空间

不同的环境允许不同类型的动作。给定环境中所有有效操作的集合通常称为操作空间。有些环境，如Atari和Go，具有离散的操作空间，其中只有有限的移动对代理可用。其他环境，如代理在物理世界中控制机器人的环境，具有连续的动作空间。在连续空间中，动作是实值向量.这种区别对深度RL中的方法有一些相当深远的影响。有些算法族只能在一种情况下直接应用，并且必须对另一种情况进行大量的重新加工。

## 策略

策略是代理用来决定采取什么操作的规则。它可以是确定性的，在这种情况下，它通常用：来表示，或者它可能是随机的，在这种情况下，它通常用：来表示，因为策略本质上是代理的大脑，用“策略”代替“Agent”并不少见，例如说“策略试图最大化报酬”。在深度RL中，我们处理参数化策略：其输出是依赖于一组参数(如神经网络的权值和偏差)的可计算函数的策略，我们可以通过一些优化算法来调整这些参数来改变行为。我们通常用或表示这种策略的参数，然后将其写成策略符号的下标，以突出显示连接：



### 确定性策略

示例：确定性策略。下面是一个代码片段，用于为TensorFlow中的连续操作空间构建一个简单的确定性策略：

obs = tf.placeholder(shape=(None, obs\_dim), dtype=tf.float32)

net = mlp(obs, hidden\_dims=(64,64), activation=tf.tanh)

actions = tf.layers.dense(net, units=act\_dim, activation=None)

### 随机策略

深RL中最常见的两种随机策略，分别是范畴策略(categorical policies )和对角高斯策略(diagonal Gaussian policies)。范畴策略可用于离散动作空间，而对角高斯策略可用于连续动作空间。对于使用和训练随机策略，两个关键的计算是非常重要的：从策略中采样操作，以及计算特定操作的似然概率，。

在下面的内容中，我们将描述如何对分类和对角线高斯策略执行这些操作。

* 分类策略

分类策略就像对离散行为的分类器。您为一个分类策略构建神经网络，就像构建分类器一样：输入是观察，然后是若干层(可能是卷积的或紧密连接的，这取决于输入的类型)，然后有一个最后的线性层，它为每个动作提供日志，然后是一个Softmax来将logits转换为概率。

取样。考虑到每个动作的概率，像TensorFlow这样的框架有内置的采样工具。例如，请参阅thetf.Distributions.cluorical文档或tf.polyomial。

Log-可能性。将最后一层概率表示为。它是一个向量，但有许多条目，因为有动作，所以我们可以把动作作为向量的索引。然后，可以通过索引到向量：来获得动作a的log可能性。



* 对角高斯策略

多变量高斯分布(或多元正态分布，如果你愿意)由均值向量和协方差矩阵描述。对角高斯分布是协方差矩阵只有对角线上项的特例。因此，我们可以用向量来表示它。

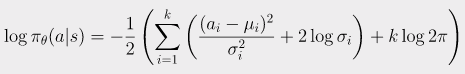
对角线高斯策略总是有一个神经网络，从观察映射到平均动作，。通常表示协方差矩阵的方法有两种。

第一种方法：有一个log标准差向量 ，它不是状态函数：是独立的参数。(您应该知道：我们的VPG、TRPO和PPO的实现就是这样做的。)

第二种方法：有一个从状态映射到log标准差的神经网络，。它可以选择与平均网络共享一些层。

注意，在这两种情况下，我们输出log标准差，而不是直接输出标准偏差。这是因为在￼中，log标准差可以自由地接受任何值，而标准差必须是非负的。如果不需要强制执行这些约束，那么训练参数就更容易了。标准偏差可以立即从对数标准差指数中得到，这样我们就不会损失任何东西。

1. 取样。给定平均作用和标准差，以及球面高斯()噪声的矢量，可以用,其中表示两个向量的元素乘积。标准框架已经内置了计算噪声向量的方法，如tf.random\_normal。或者，您可以直接向tf.distributions.Normal对象提供平均值和标准差，并使用它进行采样。
2. Log-可能性。给出了具有平均值和标准差的对角高斯的k维作用的对数似然公式：



### 轨迹(Trajectories)

轨迹是世界上状态和行为的序列，



世界的第一个状态，，是从起始状态分布随机抽样的，有时由￼：￼状态转换来表示(在时间t，状态和在时间t+1，状态之间的世界发生了什么)，受环境自然规律的支配，并且只依赖于最近的行动，。它们既可以是确定性的，

也可以是随机的，

行为来自于代理,根据其策略。

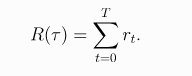
你应该知道轨迹也经常被称为插曲(episodes )或推出(rollouts)。

## 奖励函数

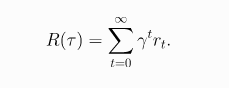
奖励函数R在强化学习中起着至关重要的作用。这取决于世界的当前状态、刚刚采取的行动以及世界的下一个状态：尽管这常常被简化为只依赖于当前状态、或状态-动作对。

代理的目标是最大化某一轨道上累积报酬的概念，但这实际上意味着一些事情。我们将不使用来解释所有这些情况，或者从上下文中清楚说明我们指的是哪种情况，或者不重要(因为相同的等式将适用于所有情况)。

一种回报是有限水平的不折现回报（finite-horizon undiscounted return），也就是在一个固定的步骤窗口中获得的报酬之和：



另一种回报是无限层折扣回报（infinite-horizon discounted return），它是代理获得的所有报酬的总和，但根据它们在未来获得的距离来贴现。这个奖励的公式包括一个折扣因子：



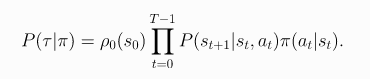
为什么我们会想要一个折扣因子呢？难道我们不想得到所有的奖励吗？是的，但折价因素在直觉上很吸引人，在数学上也很方便。

从直觉上看：现在奖励比以后奖励好。数学上：无限水平的报酬之和可能不会收敛到有限值，而且很难用方程式来处理。但在有折扣因子的情况下，在合理的条件下，无穷和收敛。

你应该知道，虽然这两种收益公式之间的界限在RL形式主义中是相当明确的，但深入的RL实践往往会使这条线变得模糊一些。例如，我们经常建立算法来优化未贴现的回报，但在估计价值函数时使用了折扣因素。

RL问题

无论选择的回报尺度(无论是无限水平折扣，还是有限水平不打折)，无论政策的选择，在RL的目标是选择一个策略，最大限度的预期回报，当代理人采取行动时。要讨论预期收益，我们首先要讨论轨道上的概率分布。假设环境变迁和策略都是随机的。在这种情况下，T步长轨迹的概率是：



然后，以表示的预期回报(以任何衡量标准表示)为：



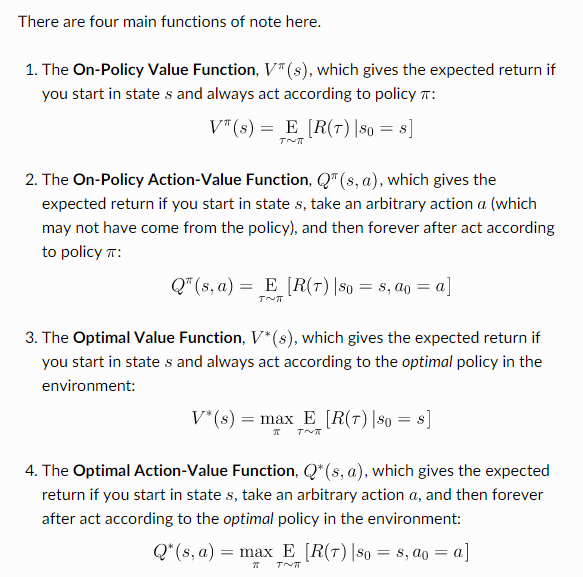
然后，rl中的中心优化问题可以用



以为最优策略。

## 值函数

了解状态或状态-动作对的值通常是有用的。所谓价值，我们指的是期望的回报，如果你开始在那个状态或状态-动作对，然后按照一个特定的策略行动以后永远。在几乎所有的RL算法中，都以某种方式使用值函数。

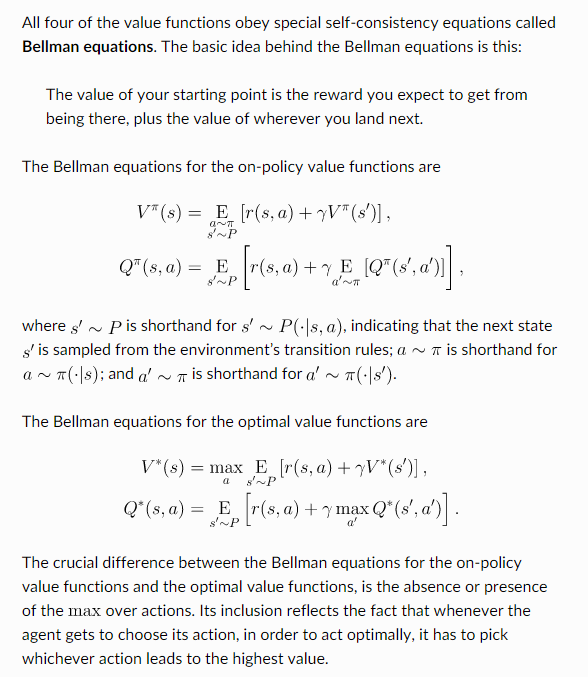


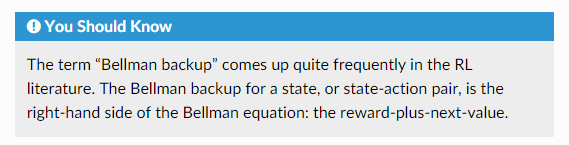
在Value函数和action-value函数之间经常出现两个关键的联系：



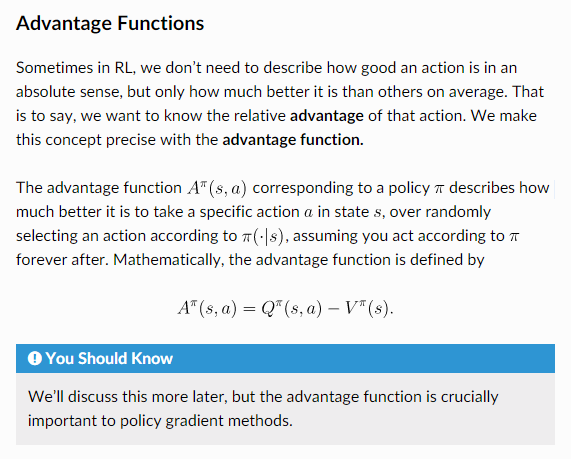


Bellman Equations

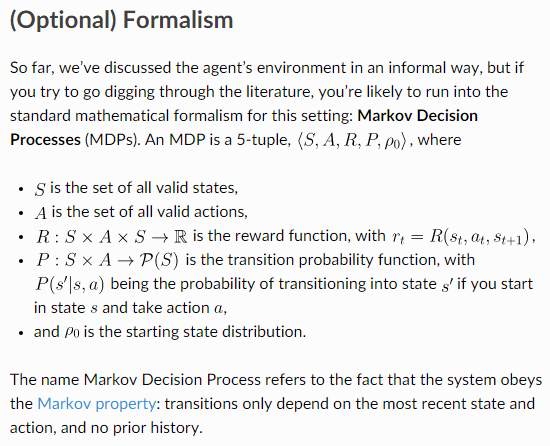




## 优势函数

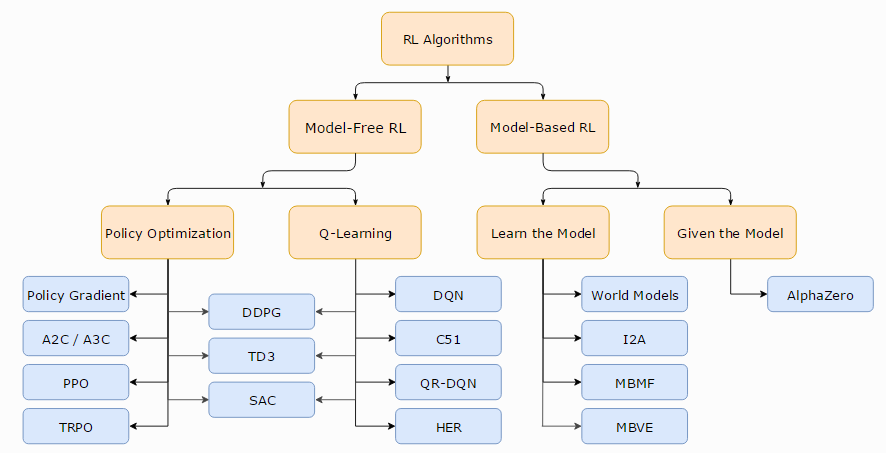


## (Optional) Formalism



# Part 2: Kinds of RL Algorithms

## A Taxonomy of RL Algorithms（RL算法的分类）



策略优化与Q-学习之间的权衡。策略优化方法的主要优势在于，它们是原则性的，也就是说，您可以直接针对您想要的东西进行优化。这往往使它们稳定可靠。相比之下，Q-学习方法只对Agent性能进行间接优化，通过训练￼满足自洽方程.这种学习有许多失败的模式，因此往往不太稳定。但是，Q-学习方法在工作时获得了更高的样本效率的优势，因为它们可以比策略优化技术更有效地重用数据。

策略优化与Q-学习之间的插值。巧合的是，策略优化和Q-学习并不是不相容的(而且在某些情况下，结果证明是等价的)，并且在这两个极端之间存在着一系列的算法。生活在这个频谱上的算法能够在任何一方的优点和弱点之间进行仔细的权衡。例如DDPG算法，它同时学习一个确定性策略和一个Q-函数，使用它们来改进对方；SAC，一种使用随机策略、熵正则化和其他一些技巧来稳定学习并在标准基准上得分高于DDPG的变体。

### Model-Free vs Model-Based RL

RL算法中最重要的分支点之一是代理是否能够访问(或学习)环境模型的问题。通过环境模型，我们指的是预测状态转换和奖励的函数。

建立一个模型的主要好处是，它允许代理通过预先思考、查看一系列可能的选择会发生什么来进行规划，并明确地在其选项之间作出决定。然后，代理可以将预先计划的结果提取为一项学习策略。这种方法的一个特别著名的例子是AlphaZero。当这起作用时，与没有模型的方法相比，它可以大大提高样本效率。

主要的缺点是环境的真实模型通常不适用于代理。如果一个代理想要在这种情况下使用模型，它必须完全从经验中学习该模型，这将带来几个挑战。最大的挑战是模型中的偏差可以被Agent利用，从而使Agent在学习的模型方面表现良好，但在实际环境中却表现出次优化(或超级可怕)。模型学习从根本上说是很难的，所以即使是紧张的努力-愿意投入大量的时间和计算-也可能无法得到回报。

使用模型的算法称为基于模型的方法，而那些不使用模型的算法称为无模型方法。虽然无模型方法放弃了使用模型在样本效率方面的潜在收益，但它们更容易实现和调整。在撰写本文时(2018年9月)，无模型方法比基于模型的方法更受欢迎，并且已经得到了更广泛的开发和测试。

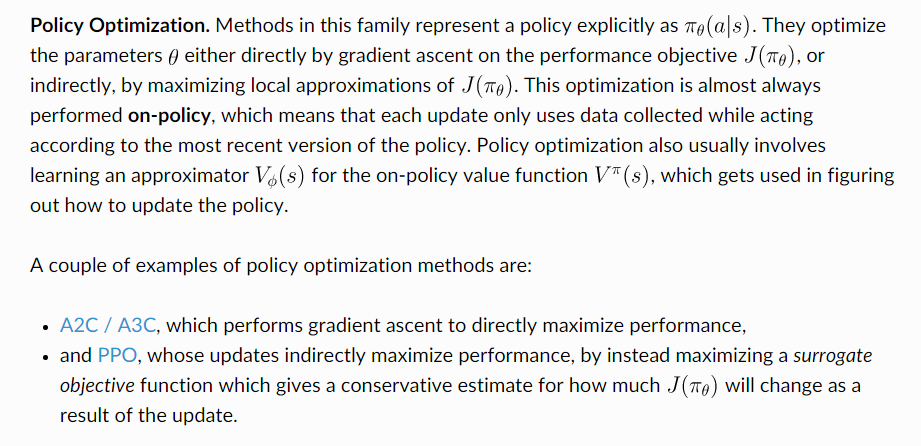
## What to Learn

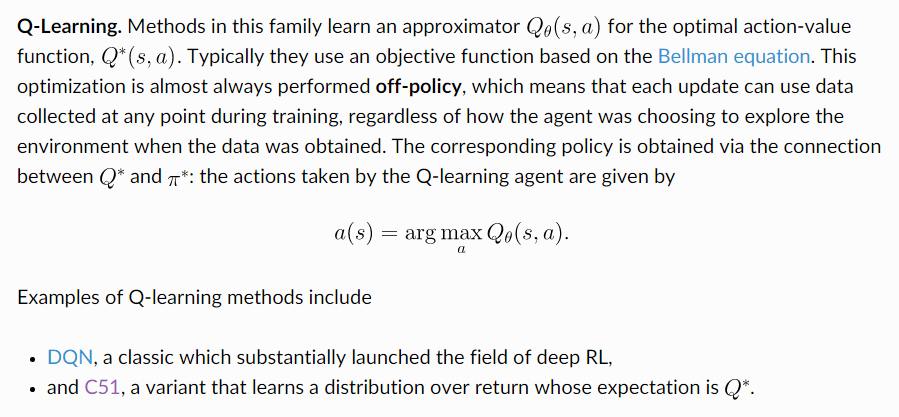
RL算法中的另一个关键分支点是要学习什么。通常的名单包括

* policies, either stochastic or deterministic,
* action-value functions (Q-functions),
* value functions,
* and/or environment models.

## What to Learn in Model-Free RL

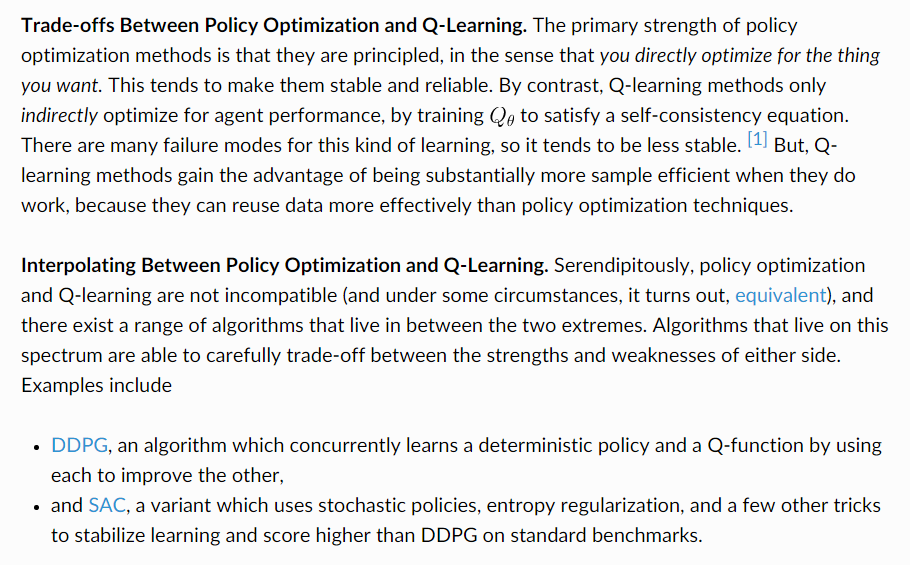
有两种使用无模型RL表示和培训代理的主要方法：





### 策略优化与Q-学习之间的权衡

策略优化方法的主要优势在于，它们是原则性的，也就是说，您可以直接针对您想要的东西进行优化。这往往使它们稳定可靠。相比之下，Q-学习方法仅通过训练￼来满足自洽方程，从而间接地优化了Agent的性能.这种学习有许多失败的模式，因此往往不太稳定。[1]但是，Q-学习方法在工作时获得了更高的样本效率的优势，因为它们可以比策略优化技术更有效地重用数据。



## What to Learn in Model-Based RL

与无模型RL不同的是，对于基于模型的RL，有许多易于定义的方法集群：有许多正交的使用模型的方法。我们将举几个例子，但这个列表还远没有详尽无遗。在每种情况下，都可以给出或学习模型。

## Links to Algorithms in Taxonomy

|  |  |
| --- | --- |
| [2] | [A2C / A3C](https://arxiv.org/abs/1602.01783) (Asynchronous Advantage Actor-Critic): Mnih et al, 2016 |

|  |  |
| --- | --- |
| [3] | [PPO](https://arxiv.org/abs/1707.06347) (Proximal Policy Optimization): Schulman et al, 2017 |

|  |  |
| --- | --- |
| [4] | [TRPO](https://arxiv.org/abs/1502.05477) (Trust Region Policy Optimization): Schulman et al, 2015 |

|  |  |
| --- | --- |
| [5] | [DDPG](https://arxiv.org/abs/1509.02971) (Deep Deterministic Policy Gradient): Lillicrap et al, 2015 |

|  |  |
| --- | --- |
| [6] | [TD3](https://arxiv.org/abs/1802.09477) (Twin Delayed DDPG): Fujimoto et al, 2018 |

|  |  |
| --- | --- |
| [7] | [SAC](https://arxiv.org/abs/1801.01290) (Soft Actor-Critic): Haarnoja et al, 2018 |

|  |  |
| --- | --- |
| [8] | [DQN](https://www.cs.toronto.edu/~vmnih/docs/dqn.pdf) (Deep Q-Networks): Mnih et al, 2013 |

|  |  |
| --- | --- |
| [9] | [C51](https://arxiv.org/abs/1707.06887) (Categorical 51-Atom DQN): Bellemare et al, 2017 |

|  |  |
| --- | --- |
| [10] | [QR-DQN](https://arxiv.org/abs/1710.10044) (Quantile Regression DQN): Dabney et al, 2017 |

|  |  |
| --- | --- |
| [11] | [HER](https://arxiv.org/abs/1707.01495) (Hindsight Experience Replay): Andrychowicz et al, 2017 |

|  |  |
| --- | --- |
| [12] | [World Models](https://worldmodels.github.io/): Ha and Schmidhuber, 2018 |

|  |  |
| --- | --- |
| [13] | [I2A](https://arxiv.org/abs/1707.06203) (Imagination-Augmented Agents): Weber et al, 2017 |

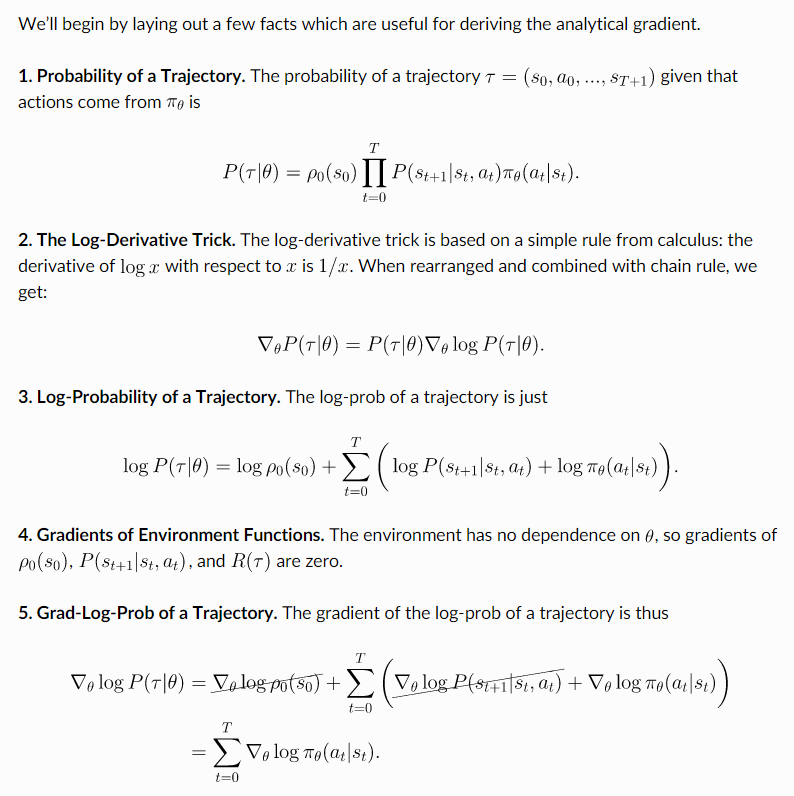
|  |  |
| --- | --- |
| [14] | [MBMF](https://sites.google.com/view/mbmf) (Model-Based RL with Model-Free Fine-Tuning): Nagabandi et al, 2017 |

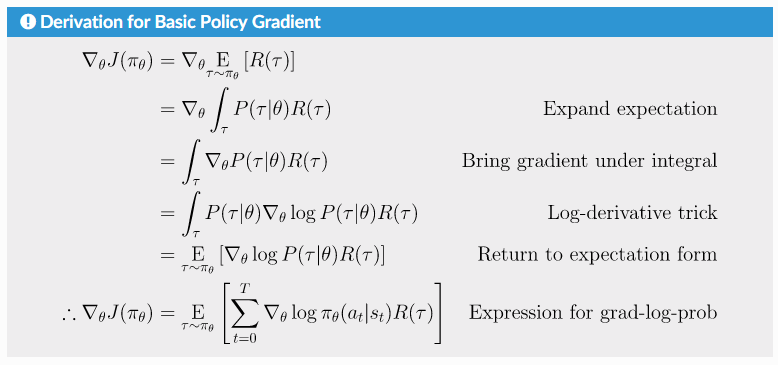
|  |  |
| --- | --- |
| [15] | [MBVE](https://arxiv.org/abs/1803.00101) (Model-Based Value Expansion): Feinberg et al, 2018 |

|  |  |
| --- | --- |
| [16] | [AlphaZero](https://arxiv.org/abs/1712.01815): Silver et al, 2017 |

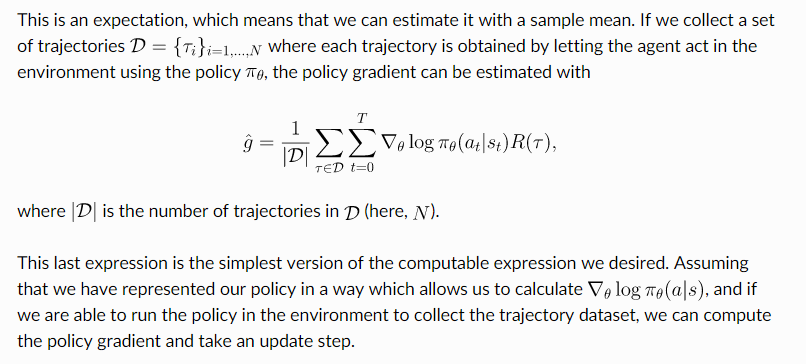
# Part 3: Intro to Policy Optimization

## Deriving(导出) the Simplest Policy Gradient





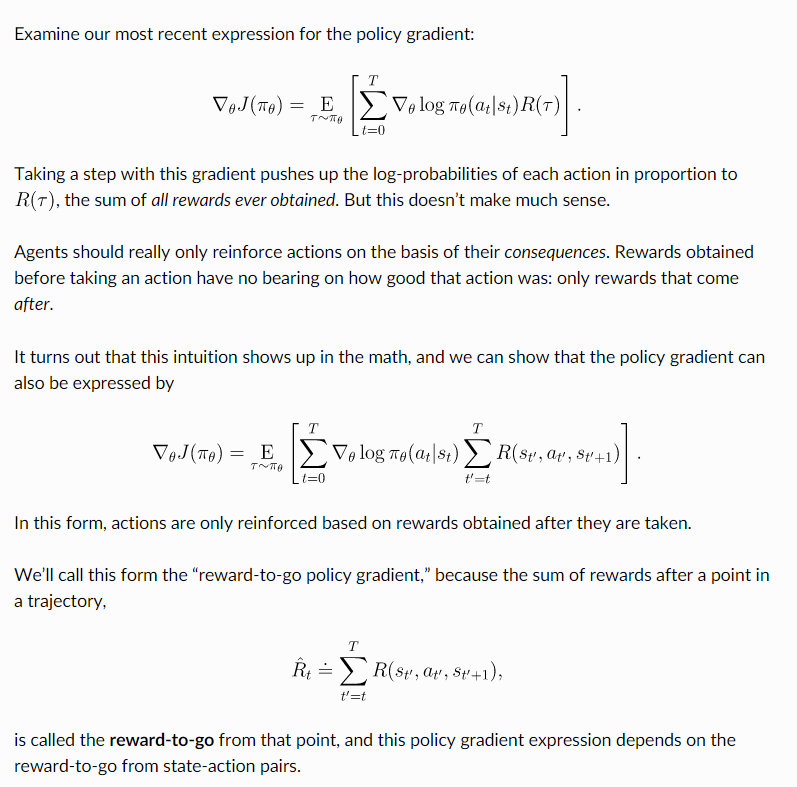
这是一个期望，这意味着我们可以用一个样本平均值来估计它。如果我们收集一组轨迹，其中每个轨迹都是通过让代理在环境中使用策略来获得的，则可以用以下方法估计策略梯度：



## Implementing the Simplest Policy Gradient

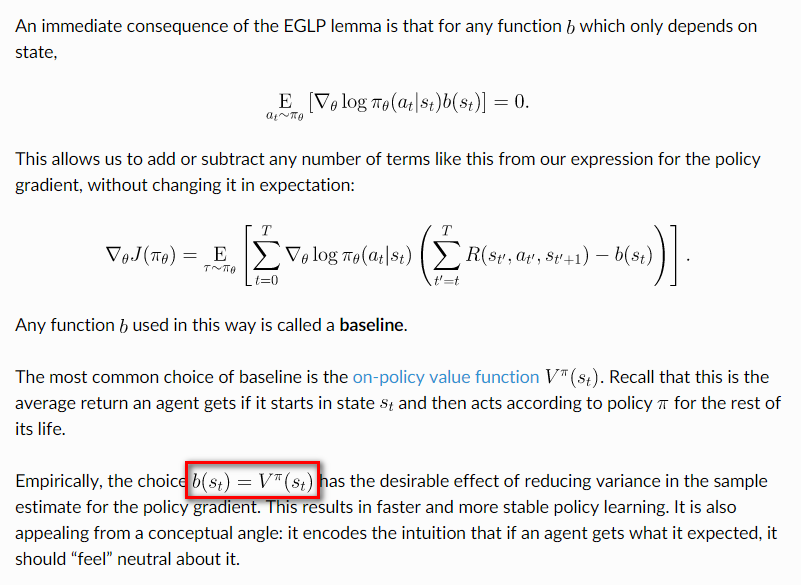
## Expected Grad-Log-Prob Lemma（预期梯度-Log概率引理）

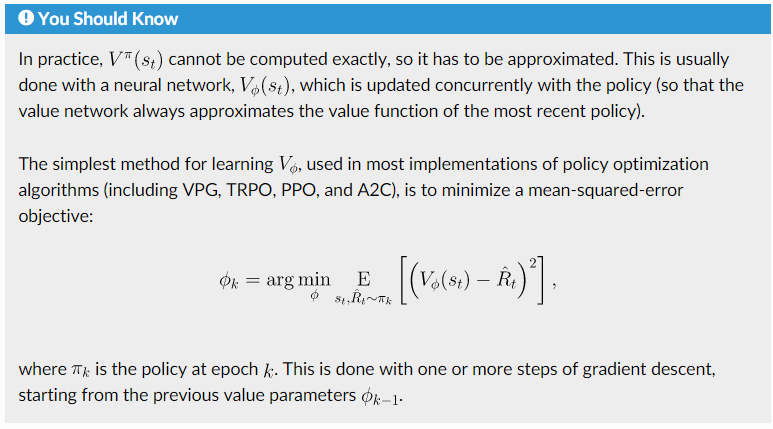
## 不要让过去分散你的注意力



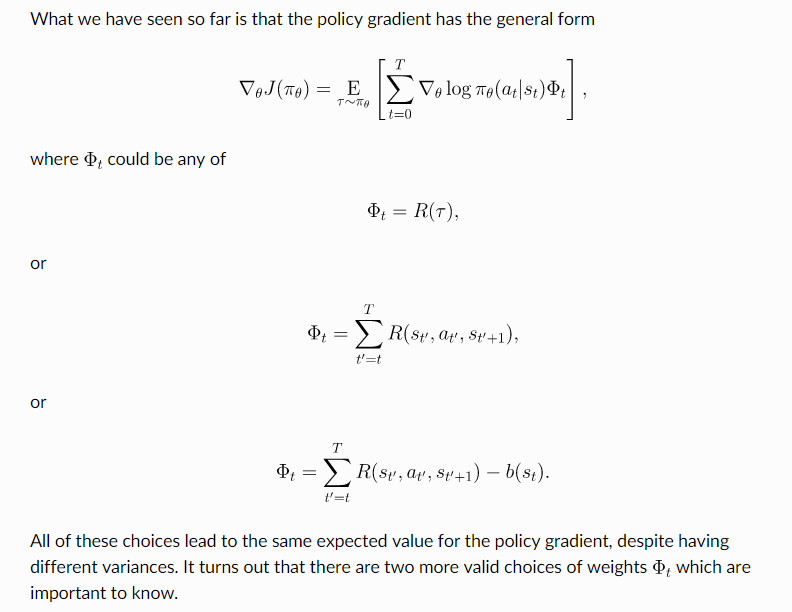
## Implementing Reward-to-Go Policy Gradient

## Baselines in Policy Gradients

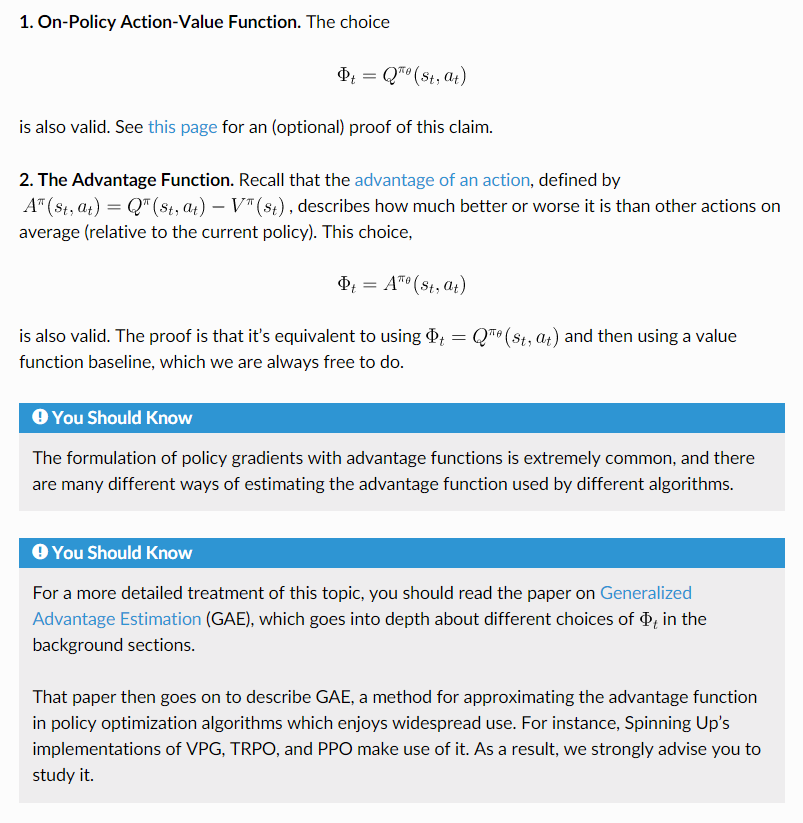




## Other Forms of the Policy Gradient



所有这些选择导致政策梯度的期望值相同，尽管有不同的差异。结果表明，有两个更有效的选择，权重，这是很重要的知道。



## Recap

在本章中，我们描述了策略梯度方法的基本理论，并将一些早期的结果与代码示例联系起来。从这里开始，有兴趣的学生应该继续研究后来的结果(价值函数、基线和政策梯度的优势制定)如何转化为旋转的香草政策梯度的实施。



图 1‑1 我是一张图

表 1‑1 我是一张表

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |

1‑1