C1-18

Machine Learning
by Andrew Ng, Stanford Engineering

Xiaojie Zhou

szxjzhou@163.com

2016.10.3

第十九集：微分动态规划

- 强化学习算法的调试(Debugging Reinforcement Learning (RL) Algorithm)
- 微分动态规划(Differential Dynamic Programming (DDP))
- 卡尔曼滤波(Kalman Filter)
- 线性二次高斯(Linear Quadratic Gaussian (LQG))

第十九集：微分动态规划

- 强化学习算法的调试
  - 对于一个典型的强化学习问题来说大致需要解决下面的问题
    - 1、如何得到状态跳转关系
      - 给定t时刻的状态s_t和动作a_t的情况下怎么知道在下一时刻会以什么概率跳转到哪些状态
      - 通过设计学习模拟器方法可以解决
    - 2、选择一个合适的回报函数
      - 一种经典的做法是采用动力系统模型中的二次回报函数
    - 3、如何找到最优值函数和最优策略
  - 但解决了这三个问题以后效果仍然不佳，如何进行调试
    - 1、重新设计学习模拟器
      - 模拟器上运行的效果很好但实际飞行效果差
    - 2、更换回报函数
      - 将飞机调整到手动飞行模式，此时会得到一组非常好的飞行策略，将其代入值函数后发现手动飞行策略下回报值更低
    - 3、寻找最优值函数和最优策略的方法使用错误
      - 将飞机调整到手动飞行模式，此时会得到一组非常好的飞行策略，将其代入值函数后发现手动飞行策略下回报值更高

第十九集：微分动态规划

- 微分动态规划
  - 之前我们推导的马尔可夫决策过程的目标是尽可能使得值函数最优
    - 之前我们只要律师的值函数最优的策略，现在我们来看一种情况，如果我们强制要求马尔可夫决策过程必须经过某些状态，那么应该如何计算最优策略
      - 比如说直升机我们要求按某一轨迹行进

1、 我们首先需要得到一些标称轨迹（就是一开始希望直升机必须经过的状态和动作的序列，这些序列可以通过训练出的一个比较差的模拟器采集得到）

$$\overline{s_0}, \overline{a_0}, \overline{s_1}, \overline{a_1}, \ldots, \overline{s_T}, \overline{a_T}$$

2、 由于原模型可能是个非线性模型，因此需要通过如下方式对其进行线性化，从而得到线性模型

$$s_{t+1} \approx \left(\nabla_s f\left(\overline{s_t}, \overline{a_t}\right)\right)^T \left(s_t - \overline{s_t}\right) + \left(\nabla_a f\left(\overline{s_t}, \overline{a_t}\right)\right)^T \left(a_t - \overline{a_t}\right) + f\left(\overline{s_t}, \overline{a_t}\right) \Leftrightarrow A_t s_t + B_t a_t$$

3、 有了模型以后就可以计算最优值函数和最优策略

a) 初始化Φ_t和Ψ_t

$$\Phi_t = \Phi_T = -U_t, \Psi_t = \Psi_T = 0$$

b) 对于$t = T-1 \ldots 0$，通过下式计算Φ_t, Ψ_t

$$\Phi_t = A_t^T \left(\Phi_{t+1} + \frac{\Phi_{t+1} B_t B_t^T \Phi_{t+1}}{V_t - B_t^T \Phi_{t+1} B_t}\right) A_t - U_t$$

$$\Psi_t = \Psi_{t+1}$$

c) 通过下式计算$t = T-1 \ldots 0$的最优值函数和最优策略：

$$V_t^*(s) = s_t^T \Phi_t s_t + \Psi_t$$

$$a_t = \frac{s_t^T A_t^T \Phi_{t+1} B_t}{V_t - B_t^T \Phi_{t+1} B_t} = L_t s_t$$

4、 在得到最优策略后，在已知最优策略和初始状态的情况下我们又可以通过如下的方式得到新的标称轨迹（然后再回到第 2 步直至模型符合要求）

$$\overline{s_0} = initial \ \ state$$

$$\overline{a} = \pi(s)$$

$$\overline{s_{t+1}} = A_t \overline{s_t} + B_t \overline{a_t}$$

不难发现这个方法背后的思想是替换线性化的过程中表现不优秀的点$\left(\overline{s_t}, \overline{a_t}\right)$，使得对应的线性模型更加符合实际情况

第十九集：微分动态规划

- 卡尔曼滤波
  - 之前我们考虑的都是可观测状态的情况，也就是一个状态只会跳转到一个明确的状态中，而我们现在考虑无法观测状态而只能观测特征的情况
    - 比如对于直升机的自动驾驶控制中，我们的状态信息包括直升机的位置和速度，但目前只有一个精度并不高的雷达，只能得到一个近似的位置信息
      - 这里面的核心问题就是如何从特征中找到状态信息，也就是在给定观测结果下属于某一个状态的概率是什么（其中s_t为t时刻的隐状态，y为观测的结果）

$$p(s_t|y_1,y_2,\dots,y_t)$$

      - 对此一种比较经典的假设是认为状态与观测结果间服从混合高斯分布
        - 因此这给求解带来了一种思路，可以通过求解出这个混合高斯分布，再进而求解边缘概率分布和条件概率分布的办法直接将给定观测结果下属于某一个状态的概率解出，但这种方法运算量极大，极其低效
        - 而卡尔曼滤波给出了一种高效的解决方案

第十九集：微分动态规划

- 卡尔曼滤波

在卡尔曼滤波中目标系统将由以下式子所描述：

1、 隐状态的跳转（其中ω_t服从均值为0，协方差矩阵为Q的高斯分布，即$\omega_t \sim \mathcal{N}(0,Q)$）：

$$s_{t+1} = As_t + Ba_t + \omega_t$$

2、 隐状态和观测数据之间的关系（其中v_t服从均值为0，协方差矩阵为R的高斯分布，即$v_t \sim \mathcal{N}(0,R)$）：

$$y_t = Cs_t + v_t$$

主要核心在于两步：

1、 预测

$$\mathrm{p}(s_t|y_1,y_2,\dots,y_t) \xrightarrow{predict} \mathrm{p}(s_{t+1}|y_1,y_2,\dots,y_t)$$

其中$\mathrm{p}(s_t|y_1,y_2,\dots,y_t)$服从均值为$s_{t|t}$，协方差矩阵为$\Sigma_{t|t}$的高斯分布，即$\mathrm{p}(s_t|y_1,y_2,\dots,y_t) \sim \mathcal{N}\left(s_{t|t},\Sigma_{t|t}\right)$，而$\mathrm{p}(s_{t+1}|y_1,y_2,\dots,y_t)$服从均值为$s_{t+1|t}$，协方差矩阵为$\Sigma_{t+1|t}$的高斯分布，即$\mathrm{p}(s_{t+1}|y_1,y_2,\dots,y_t) \sim \mathcal{N}\left(s_{t+1|t},\Sigma_{t+1|t}\right)$，这二者间满足下列关系

a) 均值关系（预测状态）：

$$s_{t+1|t} = As_{t|t} + Ba_t$$

b) 方差关系（预测协方差）：

$$\Sigma_{t+1|t} = A\Sigma_{t|t}A^T + Q$$

2、 更新

$$\mathrm{p}(s_{t+1}|y_1,y_2,\dots,y_t) \xrightarrow{update} \mathrm{p}(s_{t+1}|y_1,y_2,\dots,y_t,y_{t+1})$$

其中$\mathrm{p}(s_{t+1}|y_1,y_2,\dots,y_t)$服从均值为$s_{t+1|t}$，协方差矩阵为$\Sigma_{t+1|t}$的高斯分布，即$\mathrm{p}(s_{t+1}|y_1,y_2,\dots,y_t) \sim \mathcal{N}\left(s_{t+1|t},\Sigma_{t+1|t}\right)$，而$\mathrm{p}(s_{t+1}|y_1,y_2,\dots,y_t,y_{t+1})$服从均值为$s_{t+1|t+1}$，协方差矩阵为$\Sigma_{t+1|t+1}$的高斯分布，即$\mathrm{p}(s_{t+1}|y_1,y_2,\dots,y_t,y_{t+1}) \sim \mathcal{N}\left(s_{t+1|t+1},\Sigma_{t+1|t+1}\right)$，这二者间满足下列关系

a) Kalman 增益：

$$K_{t+1} = \Sigma_{t+1|t}C^T\left(C\Sigma_{t+1|t}C^T + R\right)^{-1}$$

b) 均值关系（更新状态）：

$$s_{t+1|t+1} = s_{t+1|t} + K_{t+1}\left(y_{t+1} - Cs_{t+1|t}\right)$$

c) 方差关系（更新协方差）：

$$\Sigma_{t+1|t+1} = \Sigma_{t+1|t} - K_{t+1}C\Sigma_{t+1|t}$$

第十九集：微分动态规划

- 线性二次高斯
  - 最后我们将卡尔曼滤波用来解决之前提到的问题

现在我们用卡尔曼滤波来解决给定观测结果下属于某一个状态的概率是什么（其中s_t为t时刻的隐状态，y为观测的结果）并以此求解出最优值函数和最优策略

$$\mathrm{p}(s_t|y_1,y_2,\dots,y_t)$$

1、 初始化（s_0为初始状态）：

$$s_{0|0}=s_0,\Sigma_{0|0}=0$$

2、 运行卡尔曼滤波算法进行迭代求解

a) 预测，从而得知$\mathrm{p}(s_{t+1}|y_1,y_2,\dots,y_t)$所属分布：

i. 均值关系（预测状态）：

$$s_{t+1|t}=As_{t|t}+Ba_t$$

ii. 方差关系（预测协方差）：

$$\Sigma_{t+1|t}=A\Sigma_{t|t}A^T+Q$$

b) 更新：

i. Kalman 增益：

$$K_{t+1}=\Sigma_{t+1|t}C^T\left(C\Sigma_{t+1|t}C^T+R\right)^{-1}$$

ii. 均值关系（更新状态）：

$$s_{t+1|t+1}=s_{t+1|t}+K_{t+1}\left(y_{t+1}-Cs_{t+1|t}\right)$$

iii. 方差关系（更新协方差）：

$$\Sigma_{t+1|t+1}=\Sigma_{t+1|t}-K_{t+1}C\Sigma_{t+1|t}$$

3、 运行线性二次型调节求解最优策略

a) 初始化Φ_t和Ψ_t

$$\Phi_t=\Phi_T=-U_t,\Psi_t=\Psi_T=0$$

b) 对于$t=T-1\dots0$，通过下式计算Φ_t,Ψ_t

$$\Phi_t=A_t^T\left(\Phi_{t+1}+\frac{\Phi_{t+1}B_tB_t^T\Phi_{t+1}}{V_t-B_t^T\Phi_{t+1}B_t}\right)A_t-U_t$$

$$\Psi_t=\Psi_{t+1}$$

c) 通过下式计算$t=T-1\dots0$的最优值函数和最优策略：

$$V_t^*(s)=s_{t|t}^T\Phi_ts_{t|t}+\Psi_t$$

$$a_t=\frac{s_{t|t}^TA_t^T\Phi_{t+1}B_t}{V_t-B_t^T\Phi_{t+1}B_t}=L_ts_{t|t}$$

第二十集：策略搜索

- 部分可观测的马尔可夫决策过程(Partially Observable Markov Decision Processes (POMDP))
- 策略搜索(Policy Search)
- 加强算法(Reinforced Algorithm)

第二十集：策略搜索

- 部分可观测的马尔可夫决策过程
  - 部分可观测的马尔可夫决策过程由下列七部分组成
    - 隐状态集合S
    - 动作集合A
    - 观测特征集合Y
    - 状态转换概率分布p_{sa}
    - 观测特征所属分布O_s
      - 对于某一时刻t来说，对应的观测特征记为y_t，对应的隐状态为s_t，该次观测y_t对应的分布为O_{st}（即$y_t \sim O_{st}$）
    - 持续时间T
    - 回报函数R

第二十集：策略搜索

- 策略搜索
  - 对于部分可观测的马尔可夫决策求解的一种方法是使用之前提到的卡尔曼滤波、线性二次高斯、线性二次型调节加以解决
    - 这种方法的原理是先找到最优值函数，然后再求最优策略，而与此同时还有一种策略搜索的办法。这种方法虽然不能保证找到的策略一定是最优的，但是实践证明这种方法效果不错
    - 首先先看策略搜索如何应用在马尔可夫决策模型中，而在此之前先做一些问题的定义
      - 在策略搜索这里，一开始不直接将策略看成是某个状态s一定要执行某一个动作a，而是采用随机策略的概念，用π(s,a)来表达在s状态下会执行a动作的概率
        - 因此π(s,a)需要满足：$\sum_a \pi(s,a) = 1, \pi(s,a) \geq 0$
        - 但就我们实际上看不可能在一个状态s下又做这个动作又同时做另一个动作，因此这个问题可以归结为一个分类问题
          - 我们可以算出s状态下所有动作执行的概率，然后选择一个最有可能执行的动作进行执行，目标是要尽量使得回报最大，即：

$$\max_\theta E\left[R(s_0,a_0) + \cdots + R(s_T,a_T) | \pi_\theta, s_0 = s\right]$$

其中$\pi_\theta(s,a)$为s状态下会执行a动作的概率，这一概率是与参数θ相关的，而我们的目的就是要像监督学习一样求出这样的θ使得回报函数最大化。对于π_θ的选取对于二元动作集来说可以选择 Sigmoid 函数，对于多元动作集来说可以选择 Softmax 函数（当然也可以根据实际情况选择其他函数）

第二十集：策略搜索

- 加强算法

加强算法的原理类似于分类里面的随机梯度下降法求参数，其具体过程如下：

$$\max_\theta E[R(s_0,a_0)+\cdots+R(s_T,a_T)|\pi_\theta,s_0=s]=\max_\theta\sum_{s_0,\dots,s_T\in S,a_0,\dots,a_T\in A}p(s_0,a_0,\dots,s_T,a_T)[R(s_0,a_0)+\cdots+R(s_T,a_T)]$$

$$=\max_\theta\sum_{s_0,\dots,s_T\in S,a_0,\dots,a_T\in A}p(s_0)\pi_\theta(s_0,a_0)p_{s_0,a_0}(s_1)\pi_\theta(s_1,a_1)\dots\pi_\theta(s_T,a_T)[R(s_0,a_0)+\cdots+R(s_T,a_T)]$$

因此：

$$\nabla_\theta E[R(s_0,a_0)+\cdots+R(s_T,a_T)|\pi_\theta,s_0=s]$$

$$=\sum_{s_0,\dots,s_T\in S,a_0,\dots,a_T\in A}\Big[p(s_0)\nabla_\theta\left(\pi_\theta(s_0,a_0)\right)p_{s_0,a_0}(s_1)\pi_\theta(s_1,a_1)\dots\pi_\theta(s_T,a_T)$$

$$+p(s_0)\pi_\theta(s_0,a_0)p_{s_0,a_0}(s_1)\nabla_\theta\left(\pi_\theta(s_1,a_1)\right)\dots\pi_\theta(s_T,a_T)+\cdots$$

$$+p(s_0)\pi_\theta(s_0,a_0)p_{s_0,a_0}(s_1)\pi_\theta(s_1,a_1)\dots\nabla_\theta\left(\pi_\theta(s_T,a_T)\right)\Big][R(s_0,a_0)+\cdots+R(s_T,a_T)]$$

$$=\sum_{s_0,\dots,s_T\in S,a_0,\dots,a_T\in A}p(s_0)\pi_\theta(s_0,a_0)p_{s_0,a_0}(s_1)\pi_\theta(s_1,a_1)\dots\pi_\theta(s_T,a_T)\left[\frac{\nabla_\theta\left(\pi_\theta(s_0,a_0)\right)}{\pi_\theta(s_0,a_0)}+\frac{\nabla_\theta\left(\pi_\theta(s_1,a_1)\right)}{\pi_\theta(s_1,a_1)}\right.$$

$$\left.+\cdots+\frac{\nabla_\theta\left(\pi_\theta(s_T,a_T)\right)}{\pi_\theta(s_T,a_T)}\right][R(s_0,a_0)+\cdots+R(s_T,a_T)]$$

$$=\sum_{s_0,\dots,s_T\in S,a_0,\dots,a_T\in A}p(s_0,a_0,\dots,s_T,a_T)\left[\frac{\nabla_\theta\left(\pi_\theta(s_0,a_0)\right)}{\pi_\theta(s_0,a_0)}+\frac{\nabla_\theta\left(\pi_\theta(s_1,a_1)\right)}{\pi_\theta(s_1,a_1)}+\cdots\right.$$

$$\left.+\frac{\nabla_\theta\left(\pi_\theta(s_T,a_T)\right)}{\pi_\theta(s_T,a_T)}\right][R(s_0,a_0)+\cdots+R(s_T,a_T)]$$

$$=E\left[\left[\frac{\nabla_\theta\left(\pi_\theta(s_0,a_0)\right)}{\pi_\theta(s_0,a_0)}+\frac{\nabla_\theta\left(\pi_\theta(s_1,a_1)\right)}{\pi_\theta(s_1,a_1)}+\cdots+\frac{\nabla_\theta\left(\pi_\theta(s_T,a_T)\right)}{\pi_\theta(s_T,a_T)}\right][R(s_0,a_0)+\cdots+R(s_T,a_T)]\right]$$

由此通过随机梯度上升的办法即可找到对应最优的θ使得下式成立：

$$\max_\theta E[R(s_0,a_0)+\cdots+R(s_T,a_T)|\pi_\theta,s_0=s]$$

其算法具体步骤为：

循环执行如下算法直至收敛

a) 采样得到一组样本

$$s_0,a_0,s_1,a_1,\dots,s_T,a_T$$

b) 计算总回报

$$payoff=R(s_0,a_0)+\cdots+R(s_T,a_T)$$

c) 使用随机梯度下降更新θ的值

$$\theta=\alpha E\left[\left[\frac{\nabla_\theta\left(\pi_\theta(s_0,a_0)\right)}{\pi_\theta(s_0,a_0)}+\frac{\nabla_\theta\left(\pi_\theta(s_1,a_1)\right)}{\pi_\theta(s_1,a_1)}+\cdots+\frac{\nabla_\theta\left(\pi_\theta(s_T,a_T)\right)}{\pi_\theta(s_T,a_T)}\right][R(s_0,a_0)+\cdots+R(s_T,a_T)]\right]$$

$$=\theta+\alpha\left[\frac{\nabla_\theta\left(\pi_\theta(s_0,a_0)\right)}{\pi_\theta(s_0,a_0)}+\frac{\nabla_\theta\left(\pi_\theta(s_1,a_1)\right)}{\pi_\theta(s_1,a_1)}+\cdots+\frac{\nabla_\theta\left(\pi_\theta(s_T,a_T)\right)}{\pi_\theta(s_T,a_T)}\right]payoff$$

从中可以看出策略搜索往往运行速度比先找到最优值函数，然后再求最优策略的方法要来得快，虽然它并不能保证这是一个最优策略，这对于需要快速决策的事件来说是很有价值的

而对于部分可观测的马尔可夫决策过程来说同样可以使用此方法，而在此之前需要用卡尔曼滤波得到t时刻的隐状态的估计值$s_{t|t}$，然后对于$s_{t|t}$使用这一方法