# PGPE强化学习mountaincar总结

1. **任务概述：**

使用gym环境下的mountaincar组件，通过强化学习的方法使得小车刚好停在旗杆处。环境条件如下：

position:[-1.4,0.6]

velocity:[-0.07,0.07]

self\_init\_position（小车初始位置）:-0.6

self\_init\_velocity（小车初始速度）:0.0

goal\_position（目标位置）:0.5

我们选择使用基于参数探索的策略梯度方法进行强化学习，设定的规则如下：

if velocity >= 0: # velocity>=0 表示车辆正在向右移动  
 if (position > -0.5) and (abs(velocity) < abs(theta)): # 当到达右侧一定位置无法向右时, 则向左  
 action = 0  
 else:  
 action = 2  
elif velocity < 0: # velocity<0 表示车辆正在向左移动  
 if (position < beta) and (abs(velocity) < abs(theta)): # 当到达左侧一定位置无法向左时, 则向右  
 action = 2  
 else:  
 action = 0

我们设置了2个参数，theta和beta，主要任务就是针对这两个参数的优化，使得小车最终能够较好的完成任务。

1. **PGPE方法概述**

PGPE是“基于参数探索的策略梯度”的简称，其方法是通过参数化的规则划分状态空间，在每一段状态空间中定义一个确定性动作策略，然后通过参数探索的过程得到此组参数下的（平均）奖励值，再通过随机梯度法求得此时奖励函数对各个参数的梯度，然后通过梯度上升的方法更新参数，直到参数迭代收敛为止。

PGPE不要求动作策略对策略参数是可导的，而策略的探索过程则是通过对参数的大量取值来实现，也即策略梯度的计算从动作层面提升到了参数的分布层面，这就为我们在动作策略上采用参数化的规则提供了基本条件。

1、定义一个MDP过程：<S,A,p,R,S0,y>，S为状态空间，A为动作空间，p为状态转移概率，R为奖励函数，S0为状态初值，y为折扣率。

2、定义动作策略：以规则的形式定义确定性的动作策略，形式如下：“if s>s\_else, take a=a0, then take a=a1”，目的有二，一是划分状态空间，规则越多，参数越多，状态空间划分的就越精细；二是相当于定义了状态转移概率p。

3、目标函数：定义参数满足的分布 ：则目标函数的解析式为：

4、离散化目标函数：用马克洛夫过程近似目标函数。对每个参数来说，从分布中取得N个参数值，每个参数值运行M回合，得到M个奖励值，对这M个奖励值取平均得到参数的平均奖励值，然后再对N个参数的平均奖励值求平均作为参数 的奖励值。如下式：

5、参数的更新：使用梯度上升法更新参数，公式如下：。

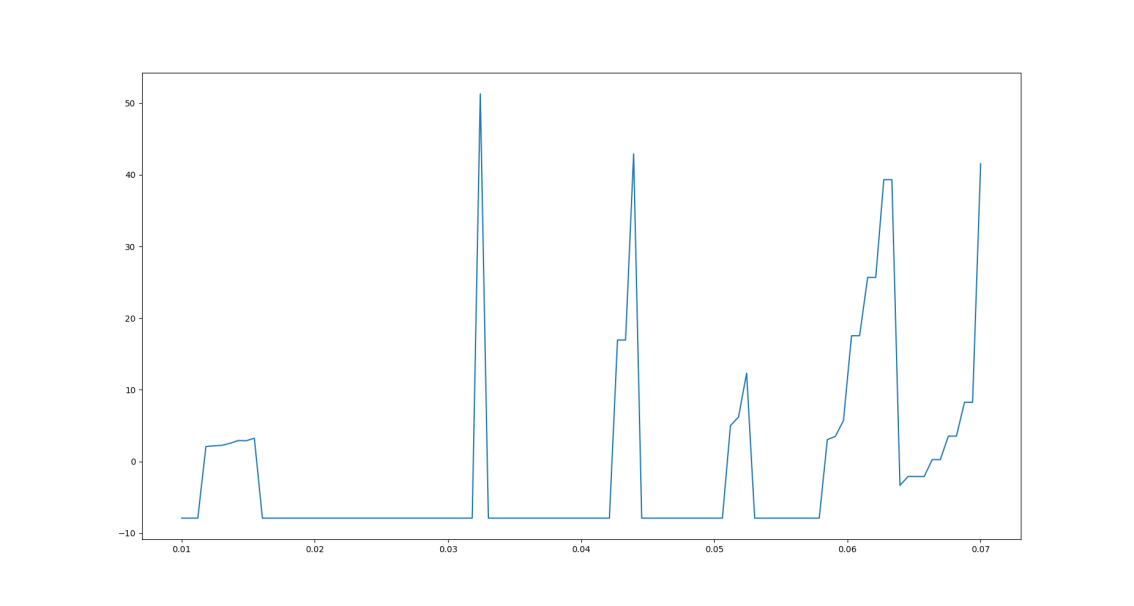
1. **奖励函数（reward）设置概述**

奖励函数的设置是强化学习中很重要的一环，特别是此任务对精度的要求很高，因此稀疏的奖励函数形式已经不能够满足要求，需要设计更加精细化、渐变的奖励函数。

考虑到奖励函数需要在旗杆处速度为0时达到最大，因此我们选择以小车到达旗杆处的速度的绝对值为自变量，用指数函数的变形来拟合，最终拟合得到的小车停在旗杆处的奖励函数如下：，这个奖励函数表示当小车速度为0时，奖励回报为100，小车速度为0.07（最大速度）时，奖励为0。完整的奖励函数如下：



为了观察奖励函数的性质，我们取beta=-0.7,theta=[0.00,0.07]来验证，theta-reward图像如下图：



可以看出，奖励函数的图像非常复杂，存在很多极值，这也是因为我们选择的动作策略会导致智能体（小车）在一个运行周期内在底部徘徊不同的次数后到达顶端旗杆，而奖励函数的设计重点是到达旗杆的速度，故经历不同时间到达旗杆时的小车只要速度相同，奖励函数相差不大。

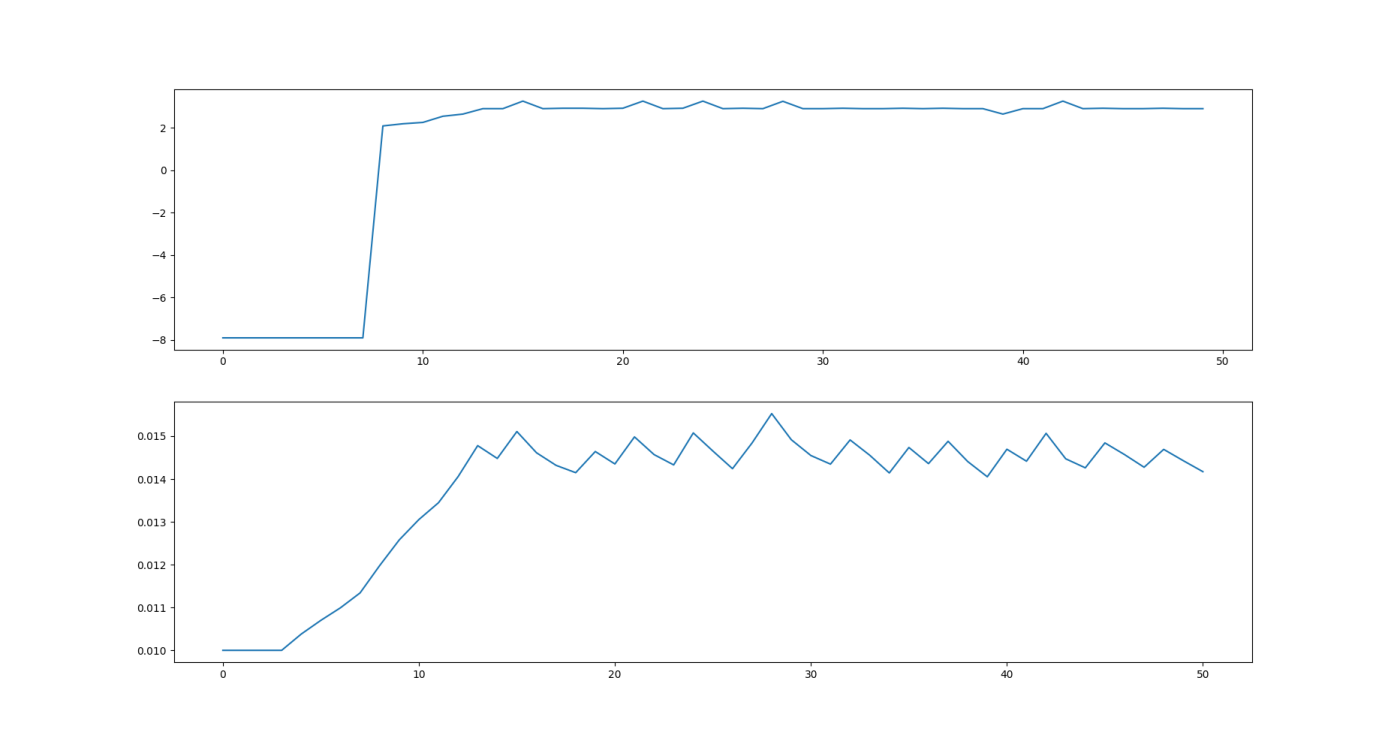
因为这是一个较为简单的模型，因此可以通过仿真模拟出奖励函数与参数的关系。随着参数数量的增加，以及强化学习系统的复杂程度的增加，直接仿真出奖励函数与参数的关系具备越来越小的可能性，需要通过机器学习算法来找到奖励函数极值。

1. **单参数theta强化学习过程和结果概述**

我们首先设定beta=-0.7，theta不确定，开始进行强化学习。下文中的”mu”是指参数theta的分布的均值，”sigmma”是指参数sigmma分布的方差。

1. theta=0.01开始强化学习

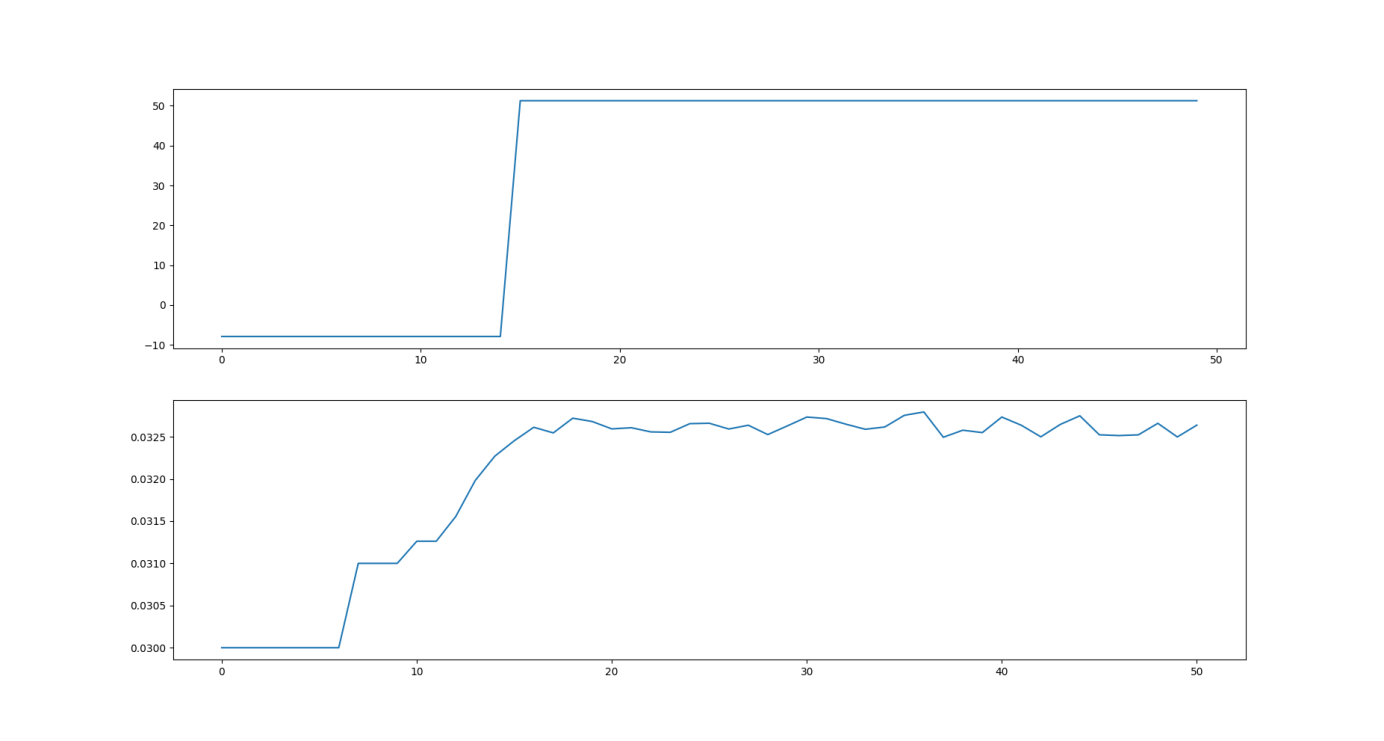
当sigmma=0.0006时开始进行梯度上升。在mu=0.0151时达到最大reward值3.256，迭代15轮后收敛，最终theta稳定在[0.0141,0.0152]之间，reward稳定在[2.900,3.256]之间。



**（注：上为迭代次数- reward图像，下为迭代次数- mu图像，下同）**

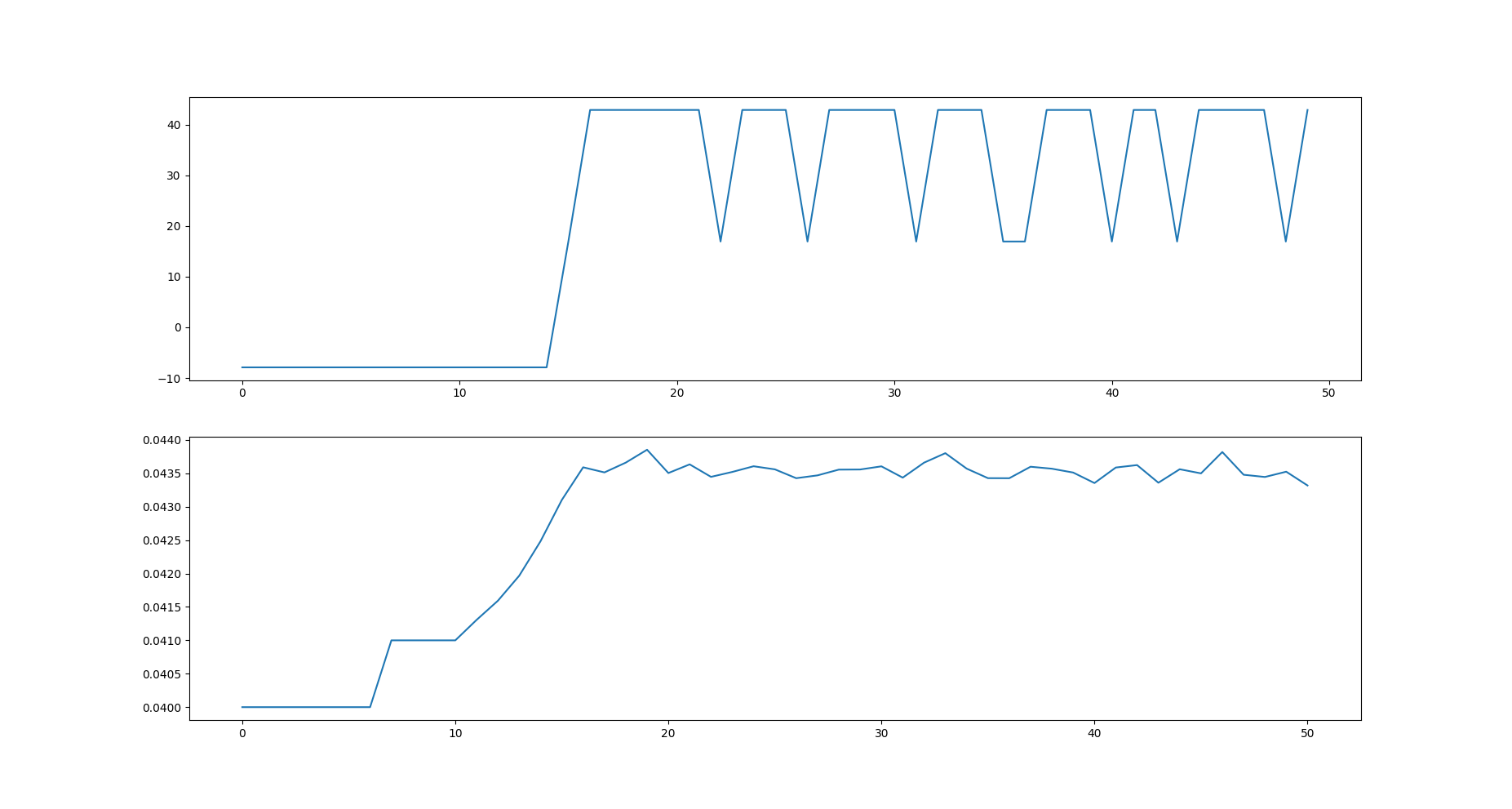
1. theta=0.03开始强化学习

当theta上升到0.031时开始进行梯度上升。在mu=0.0326时达到最大reward值51.275，迭代15轮后收敛，最终theta值稳定在[0.0325,0.0327]之间，reward稳定在51.275。



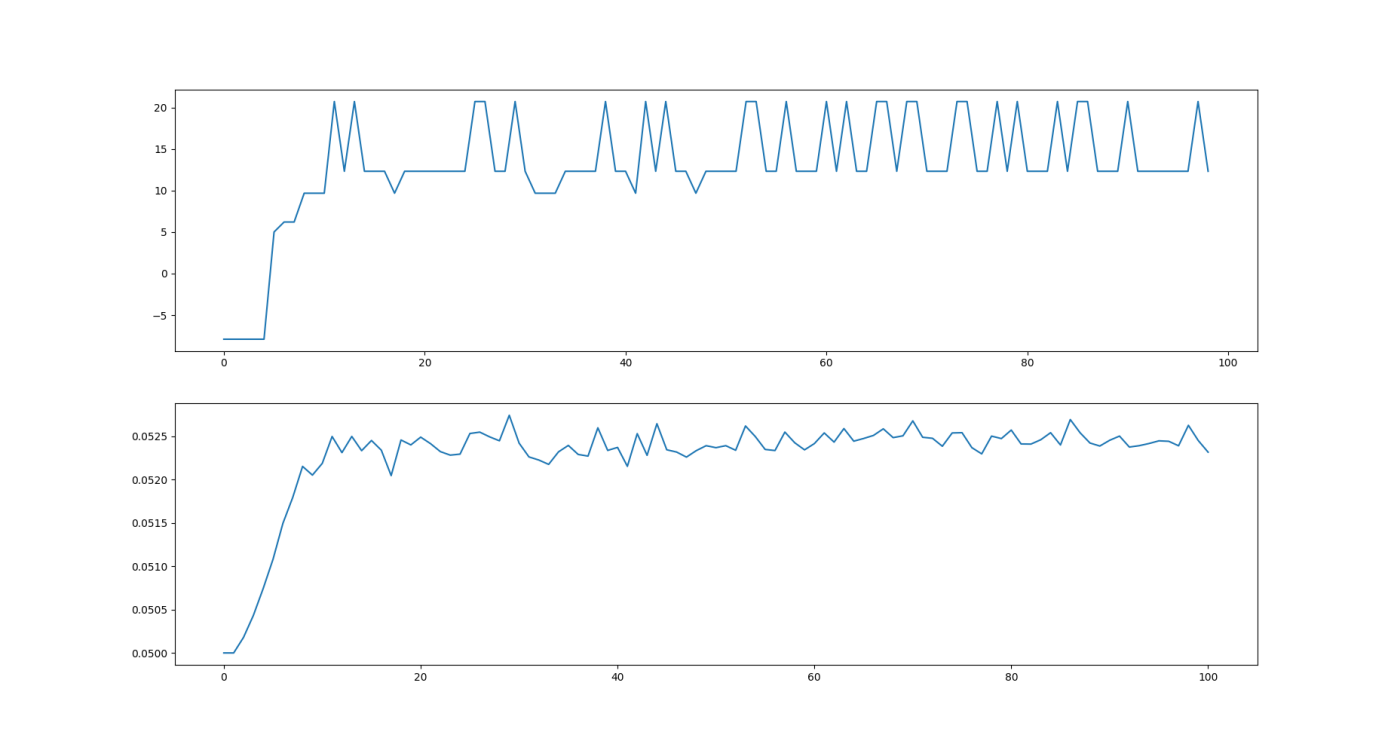
1. theta=0.04开始强化学习

当theta上升到0.041时开始进行梯度上升。在mu=0.0436时达到最大reward值42.919，迭代16轮后收敛，最终theta值稳定在[0.0430,0.0435]之间，reward值在16.947或42.919之间波动。



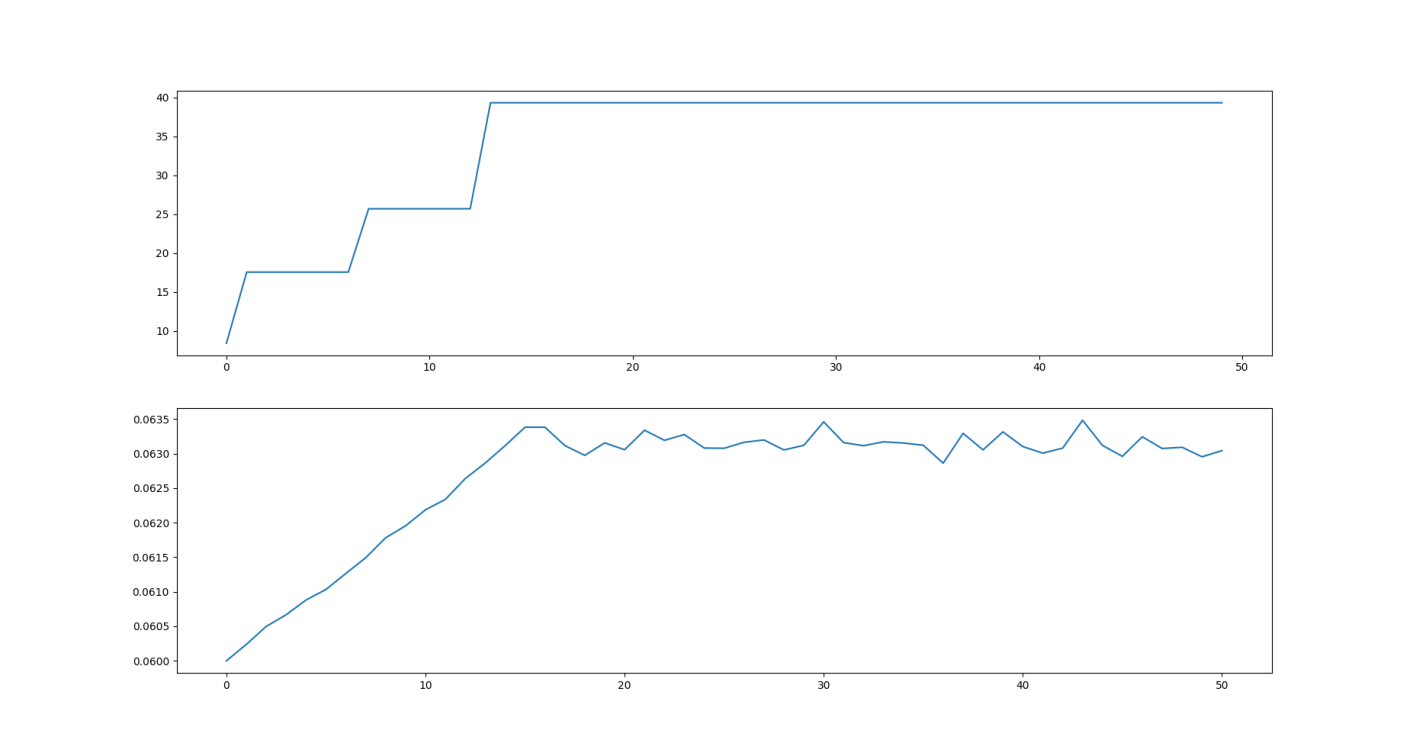
1. theta=0.05开始强化学习

当sigmma=0.0004时开始梯度上升。在mu=0.0525时达到最大reward值20.718，迭代11轮后开始进入波动状态，最终theta稳定在[0.0523,0.0526]之间，reward值在20.718和12.322之间波动。



1. theta=0.06开始强化学习

开始进行梯度上升。在mu=0.0633时达到最大reward值39.317。迭代13轮后收敛，最终theta值稳定在[0.0630,0.0635]之间，reward值稳定在39.317。



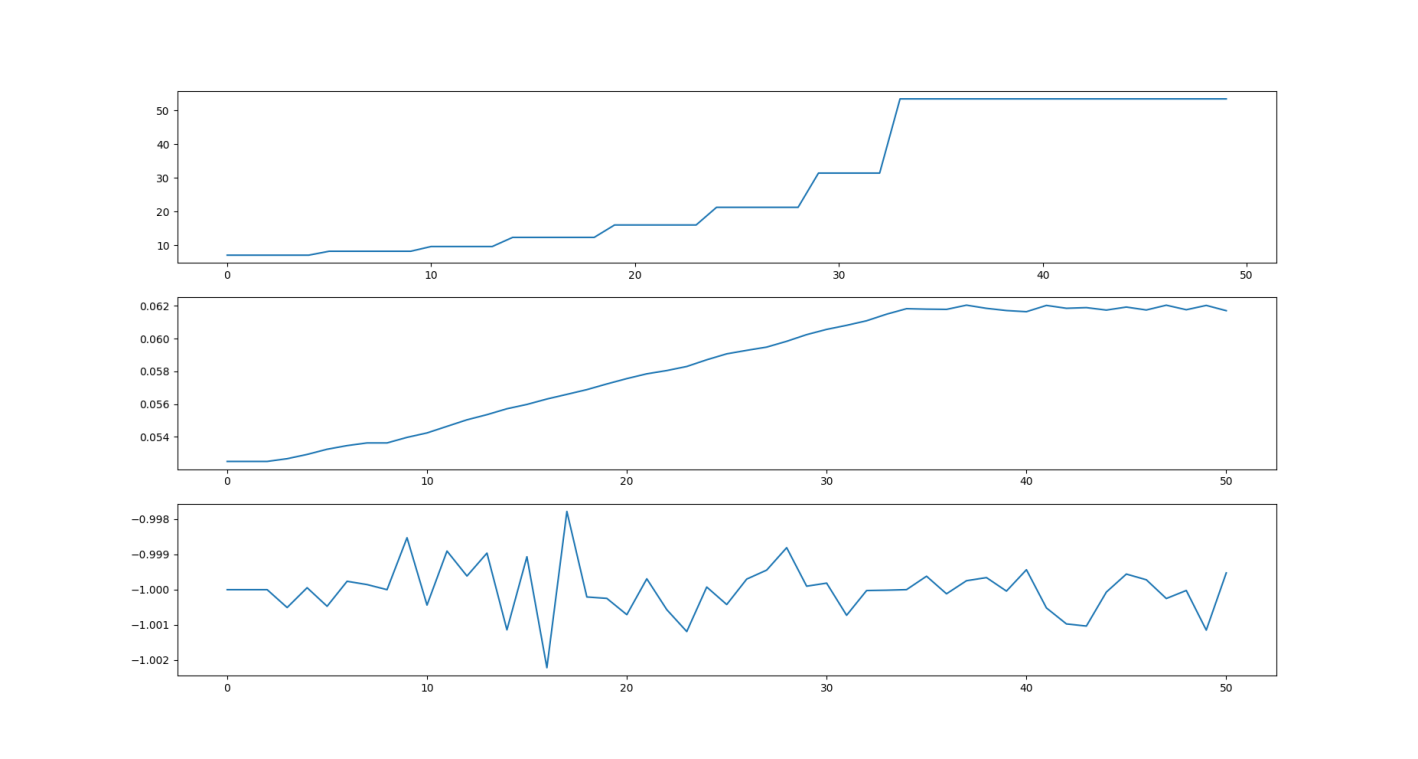
1. **双参数theta、beta强化学习过程和结果概述**

为了探究双参数对强化学习收敛过程的影响同时更加符合实际情况，我们使用两类初始值进行双参数的强化学习，一种是在单参数探索到的最大值对应的参数theta的基础上加上对beta的探索，另一种是直接在原有theta初值上加上对beta的探索。采用这两类方法的原因有二：一是比较两种方法梯度上升的速度；二是探索不同的初值带来的结果的不同，因为该梯度上升方法只能找到局部最优解，为了比较不同的局部最优解，需要多个不同的初值。

1. 在单参数最大值附近进行双参数探索

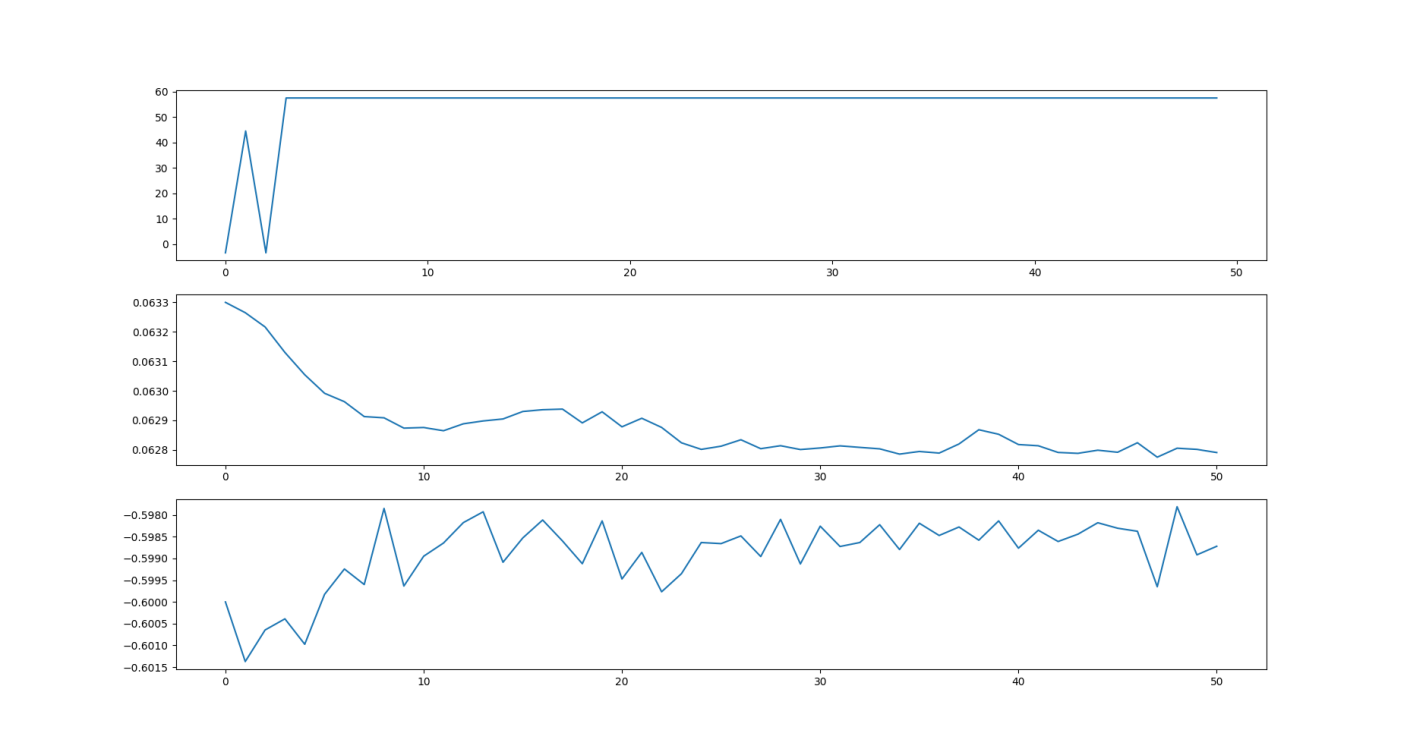
我们分别在theta=0.0151，0.0326，0.0433，0.0525，0.0633处对不同的beta值：-1.0，-0.8，-0.7，-0.6进行双参数探索，即一共探索20组初值。经过50轮的探索，有两组数据取得了优于单参数探索的性能，它们分别是(theta=0.053,beta=-1.0)与(theta=0.0633,beta=-0.6)组。

在(theta=0.053,beta=-1.0)组中，迭代33轮后收敛，达到最大reward值53.416，此时theta值稳定在0.0618附近，beta值在[-1.001,-0.999]之间波动。如下图：



**（注：最上图为迭代次数- reward图像，中间图为迭代次数- theta图像，下面图为迭代次数- beta图像，下文同）**

在(theta=0.0633,beta=-0.6)组中，迭代3轮即收敛，达到最大reward值57.515，此时theta值稳定在0.0628附近，beta值稳定在[-0.6,-0.598]之间。如下图：



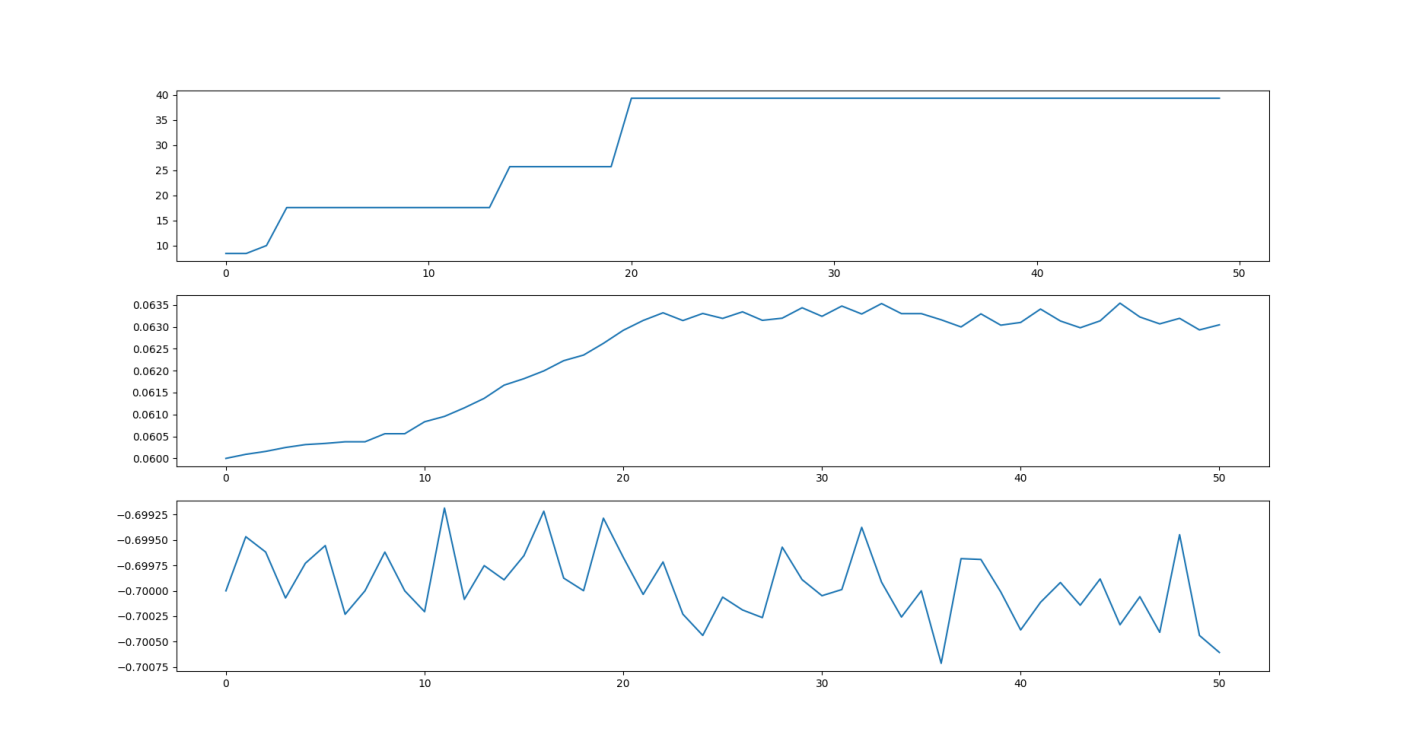
其他的各组数据收敛后没有超过原有的单参数的最大值，说明原有的beta=-0.7较为合理。

1. 直接进行双参数探索

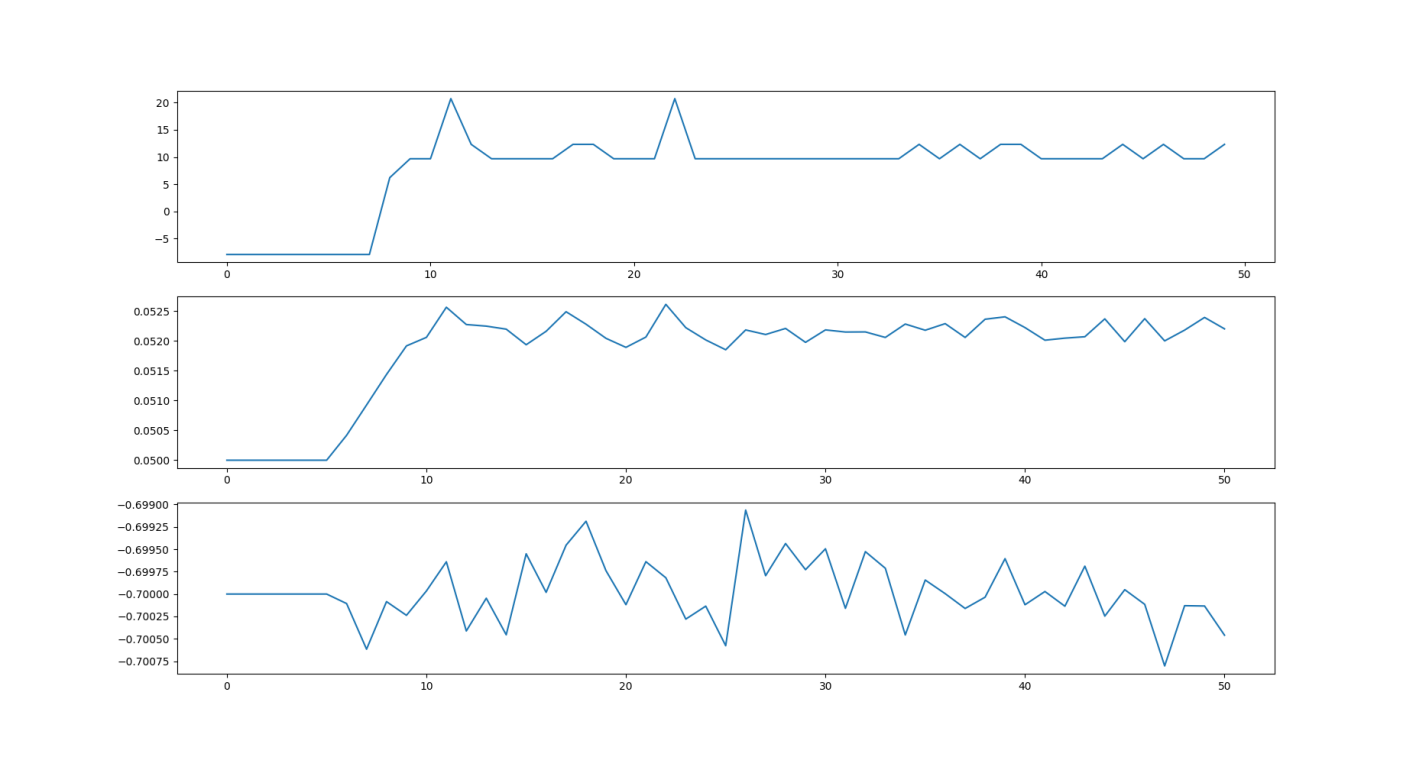
我们在theta=0.03,0.04,0.05,0.06,beta=-0.7开始直接进行双参数探索，其他beta初值的情况由于电脑限制没有进行仿真。

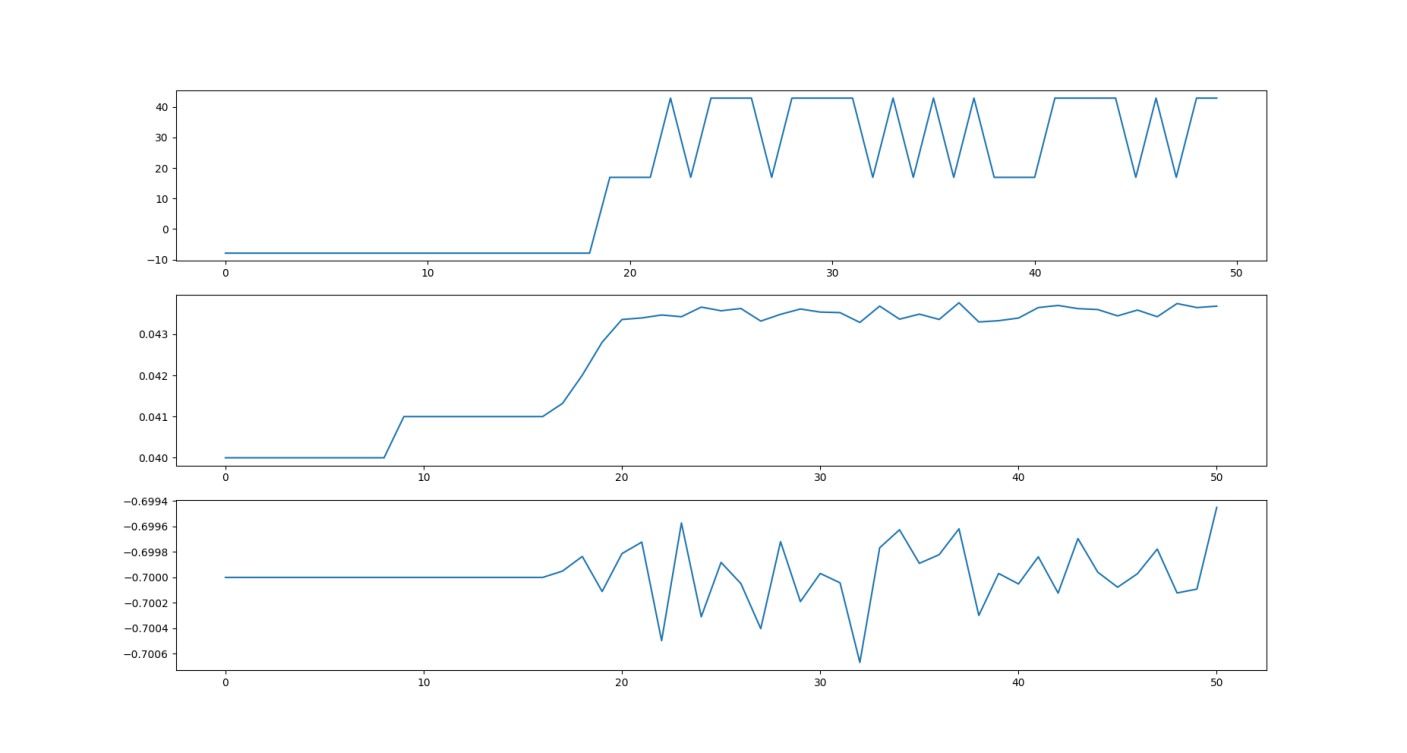
直接进行双参数探索的结果没有超过单参数的最优值，且迭代收敛时间明显增加，但收敛后的波动程度明显下降。这说明二元函数的随机梯度下降法速度下降，这是因为二元函数空间多了两个维度，因此需要更加充足的探索。不同处值的探索图像如下所示：

(theta=0.06,beta=-0.7):

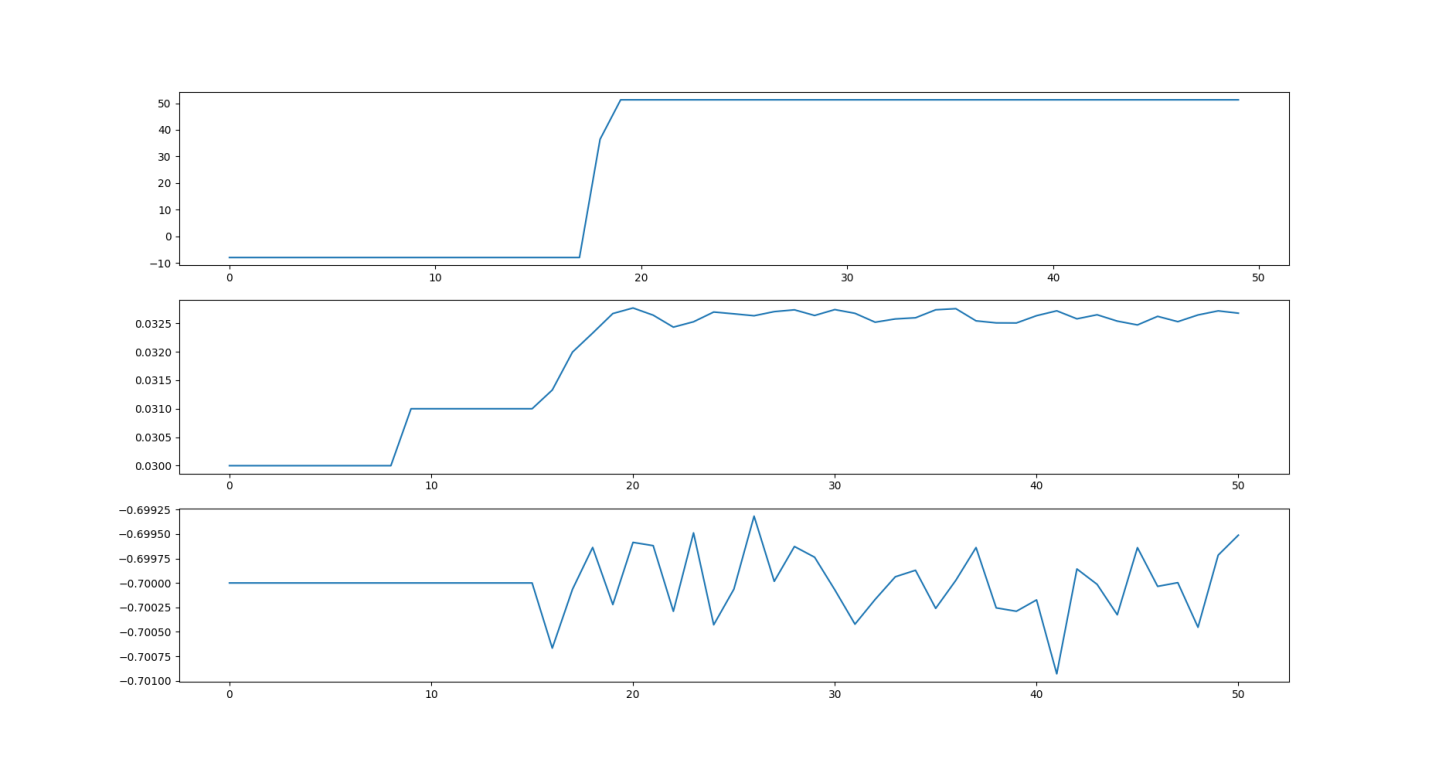


(theta=0.05,beta=-0.7):



(theta=0.04,beta=-0.7):

(theta=0.03,beta=-0.7):



1. **总结**
2. 规则划分状态动作空间为若干离散空间，规则中的参数的探索-优化过程等同于传统强化学习过程不断探索-调整Q(s,a)函数的过程，而动作为确定性动作（与传统强化学习最大的不同），故动作不存在探索过程。
3. 为了得到较为准确的梯度结果，参数点分布的方差取得很小，导致梯度上升只能得到局部最优解；如果将方差取大，可能会使参数向错误的方向运动，最后不会收敛。为了得到满意的局部最优解，可以同时运行多个初始值开始运动（需要根据经验取值），也可以在收敛到一个局部最优时人为改编参数的值，使其向下一个局部最优点运行。

3、对于此任务来说，一个参数可以完成任务，两个参数可以更好的完成任务，但目前只能针对固定的初始状态完成任务，若每回合初始状态改变，现有的规则不足以使智能体很好的完成任务，因此如果要想满足更加复杂的情况，还需要设计更多的规则。