**强化学习理论学习与代码实现**

# Trust Region Policy Optimization,TRPO

John等人[1]提出了一个具有保证单调改进的优化策略迭代过程，该算法类似于自然策略梯度法，对神经网络等大型非线性策略的优化是有效的。

策略梯度算法PG、DDPG等在离散动作空间和连续动作空间都取得了很好的成果，这一系列算法的梯度更新满足如下关系：



$$

\theta\_{n e w}=\theta\_{o l d}+\alpha \nabla\_{\theta} J

$$

策略梯度的方法取得好的结果存在着一些难度，因为这类方法对迭代步骤数（步长）非常敏感：如果选得太小，训练过程就会慢得令人绝望；如果选得太大，反馈信号就会淹没在噪声中，甚至有可能让模型表现雪崩式地下降。这类方法的采样效率也经常很低，学习简单的任务就需要百万级至十亿级的总迭代次数。

所谓合适的步长是指当策略更新后，回报函数的值不能更差。如何选择这个步长？或者说，如何找到新的策略使得新的回报函数的值单调增，或单调不减。TRPO的核心就是解决不好确定Learning rate (或者 Step size)的问题。

TRPO的做法是将新的策略所对应的回报函数分解成旧的策略所对应的回报函数+其他项。只要新的策略所对应的其他项大于等于零，那么新的策略就能保证回报函数单调不减。



$$

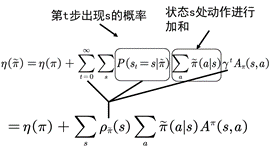
\eta(\tilde{\pi})=\eta(\pi)+\mathbb{E}\_{s\_{0}, a\_{0}, \cdots \sim \tilde{\pi}}\left[\sum\_{t=0}^{\infty} \gamma^{t} A\_{\pi}\left(s\_{t}, a\_{t}\right)\right]

$$

其中表示旧的策略，为新的策略， $A\_{\pi}(s, a)=Q\_{\pi}(s, a)-V\_{\pi}(s)$

上述reward可以展开为下式：





$$

\begin{aligned}

\eta(\tilde{\pi}) &=\eta(\pi)+\sum\_{t=0}^{\infty} \sum\_{s} P\left(s\_{t}=s \mid \tilde{\pi}\right) \sum\_{a} \tilde{\pi}(a \mid s) \gamma^{t} A\_{\pi}(s, a) \\

&=\eta(\pi)+\sum\_{s} \sum\_{t=0}^{\infty} \gamma^{t} P\left(s\_{t}=s \mid \tilde{\pi}\right) \sum\_{a} \tilde{\pi}(a \mid s) A\_{\pi}(s, a) \\

&=\eta(\pi)+\sum\_{s} \rho\_{\tilde{\pi}}(s) \sum\_{a} \tilde{\pi}(a \mid s) A\_{\pi}(s, a)

\end{aligned}

$$

注意，这时状态s的分布由新的策略产生，对新的策略严重依赖。

TRPO第一个技巧

这时，我们引入TRPO的第一个技巧对状态分布进行处理。我们忽略状态分布的变化，依然采用旧的策略所对应的状态分布。这个技巧是对原代价函数的第一次近似。其实，当新旧参数很接近时，我们将用旧的状态分布代替新的状态分布也是合理的。这时，原来的代价函数变成了：



$$

L\_{\pi}(\tilde{\pi})=\eta(\pi)+\sum\_{s} \rho\_{\pi}(s) \sum\_{a} \tilde{\pi}(a \mid s) A\_{\pi}(s, a)

$$

我们再看(7.5)式的第二项策略部分，这时的动作a是由新的策略产生。可是新的策略是带参数的，这个参数是未知的，因此无法用来产生动作。这时，我们引入TRPO的第二个技巧。

1. Schulman, J., et al. *Trust Region Policy Optimization*. 2015. arXiv:1502.05477.

<https://www.jianshu.com/p/f4d383b0bd4c?utm_source=desktop&utm_medium=timeline>

<https://zhuanlan.zhihu.com/p/26308073>