Dr. Pei (https://blogs.cuit.columbia.edu/zp2130/)

Electrical and Computer Engineering | STCO

Search

Search

+ Add post (https://blogs.cuit.columbia.edu/zp2130/wp-admin/post-new.php)

Menu

- Home (https://blogs.cuit.columbia.edu/zp2130/)
  - Teaching (https://blogs.cuit.columbia.edu/zp2130/teaching/)
  - SPCsim (https://blogs.cuit.columbia.edu/zp2130/spcsim/)
  - Service (https://blogs.cuit.columbia.edu/zp2130/service/)
  - RL (https://blogs.cuit.columbia.edu/zp2130/reinforcement_learning/)
  - Rankings (https://blogs.cuit.columbia.edu/zp2130/rankings/)
  - Publications (https://blogs.cuit.columbia.edu/zp2130/publications/)
  - Projects (https://blogs.cuit.columbia.edu/zp2130/projects/)
  - Posts (https://blogs.cuit.columbia.edu/zp2130/posts/)
  - Experience (https://blogs.cuit.columbia.edu/zp2130/experience/)
  - Contact (https://blogs.cuit.columbia.edu/zp2130/contact/)
  - Cacti++ (https://blogs.cuit.columbia.edu/zp2130/cacti/)
  - Bible (https://blogs.cuit.columbia.edu/zp2130/bible/)
  - Activities/Talks (https://blogs.cuit.columbia.edu/zp2130/activities_and_talks/)
  - About (https://blogs.cuit.columbia.edu/zp2130/about/)

Email Address: Google Scholar (https://scholar.google.com/citations?hl=en&user=-pdAvr4AAAAJ&view_op=list_works&sortby=pubdate)

zp2130@caa.columbia.edu (mailto:zp2130@caa.columbia.edu)

p@caa.columbia.edu (mailto:p@caa.columbia.edu)

Blog Stats

155,751 hits

State Action/Control

blogs.cuit.columbia.edu/p (https://blogs.cuit.columbia.edu/p/)

Meta

Site Admin (https://blogs.cuit.columbia.edu/zp2130/wp-admin/)

Log out (https://blogs.cuit.columbia.edu/zp2130/wp-login.php?action=logout&_wpnonce=67f6eed69a)

Entries feed (https://blogs.cuit.columbia.edu/zp2130/feed/)

Comments feed (https://blogs.cuit.columbia.edu/zp2130/comments/feed/)

WordPress.org (https://wordpress.org/)

Policy Gradient Methods for Reinforcement Learning with Function Approximation

Policy Gradient Methods for Reinforcement Learning with Function Approximation (http://blogs.cuit.columbia.edu/zp2130/files/2019/02/policy-gradient-methods-for-reinforcement-learning-with-function-approximation.pdf)

Math Analysis

Google Scholar (https://scholar.google.com/citations?hl=en&user=-pdAvr4AAAAJ&view_op=list_works&sortby=pubdate)

Markov Decision Processes and Policy Gradient (https://blogs.cuit.columbia.edu/zp2130/policy_gradient/)

So far in this book almost all the methods have been *action-value methods*; they learned the values of actions and then selected actions based on their estimated action values; their policies would not even exist without the action-value estimates. In this chapter we consider methods that instead learn a **parameterized policy** that can select actions **without consulting a value function**. A value function may still be used to **learn the policy parameter**, but is **not required for action selection**.

method	**Value function**	**Policy**
Action-value Methods	**Value of actions**	would not even exist
Policy Gradient Methods	without consulting a value function, or a value function may be used to **learn the policy parameter**, but is not required for action selection	learn a **parameterized** policy $\pi(a \mid s, \theta) = Pr\{A_t = a \mid S_t = s, \theta_t = \theta\}$

<Reinforcement Learning, An Introduction> Richard S. Sutton and Andrew G. Barto

这篇论文提出的策略(Policy)用它本身的FA(Function Approximator)来表现，策略与值函数无关，通过**期望回报**与**策略参数**的**梯度**来更新**策略。这篇论文主要的新成果是通过一个近似动作值或者高级函数，该梯度能写成适合估计的形式。**

$$\frac{\partial \rho}{\partial \theta} = \sum_s d^{\pi}(s) \sum_a \frac{\partial \pi(s, a, \theta)}{\partial \theta} Q^{\pi}(s, a)$$

Notation

$P^a_{ss'} = Pr\{s_{t+1} = s' \mid s_t = s, a_t = a\}$: state transition probabilities.

$R^a_s = E\{r_{t+1} | s_t = s, a_t = a\}$: expected rewards. $\forall s, s' \in S, a \in A$

$\pi(s, a, \theta) = Pr\{a_t = a \mid s_t = s, \theta\}$: A policy which the agent's decision making procedure at each time.

$\forall s \in S, a \in A, where\ \theta \in R^l, for\ l << |S|, is\ a\ paramter\ vector. \frac{\partial \pi(s,a)}{\partial \theta}\ exists,\ \pi(s,a)\ is\ for\ \pi(s,a,\theta)$

Google Scholar [https://scholar.google.com/citations?hl=en&user=-pdAvr4AAAAJ&view_op=list_works&sortby=pubdate]

$\rho(\pi)$: approximate action-value function, function approximation, long-term expected reward per step. two ways of formulating the agent's objective: average reward formulation and start state formulation. $\rho(\pi)$ is independent of state.

$d^{\pi}(s) = lim_{tn\to\infty} Pr\{s_t = s \mid s_0, \pi\}$: stationary distribution of states under **π.**

γ : [0, 1] a discount rate. In **start-state formulation**, we define $d^{\pi}(s)$ as a discounted weighting of states encountered starting at s_0 and then following **π** :

$$d^{\pi}(s) = \sum_{t=0}^{\infty} \gamma^t Pr\{s_t = s \mid s_0, \pi\}.$$

Q^{π} : the value of a state-action pair given a policy

π : **某策略的近似者函数。**

f_w : S x A → R be our approximation to **Q^{π},** with parameter **ω**. **某值函数的近似函数。**

$\hat{Q}^{\pi}(s_t, a_t)$: some unbiased estimator of **Q^{π}**(s_t, a_t), perhaps R_t.

Proof Key Steps about Theorem 1 (Policy Gradient)

Define

$$V^{\pi}(s) = \sum_a \pi(s,a) Q^{\pi}(s,a) \quad (1)$$

For the start-state formulation:

$$Q^{\pi}(s,a) = R_s^a + \sum_{s'} \gamma P_{ss'}^a V^{\pi}(s') \quad (2)$$

so

$$\begin{aligned} \frac{\partial Q^{\pi}(s,a)}{\partial \theta} &= \frac{\partial \left[R_s^a + \sum_{s'} \gamma P_{ss'}^a V^{\pi}(s')\right]}{\partial \theta} \\ &= \sum_{s'} \gamma P_{ss'}^a \frac{\partial V^{\pi}(s')}{\partial \theta} \end{aligned} \quad (3)$$

Then, we consider (3) partial differential with respect to theta,

Google Scholar [https://scholar.google.com/citations?hl=en&user=-pdAvr4AAAAJ&view_op=list_works&sortby=pubdate]

$$\begin{aligned}\frac{\partial V^\pi(s)}{\partial\theta} &= \frac{\partial[\sum_a \pi(s,a)Q^\pi(s,a)]}{\partial\theta} \\ &= \sum_a \left[\frac{\partial\pi(s,a)}{\partial\theta}Q^\pi(s,a) + \pi(s,a)\frac{\partial Q^\pi(s,a)}{\partial\theta}\right] \\ &= \sum_a \left[\frac{\partial\pi(s,a)}{\partial\theta}Q^\pi(s,a) + \pi(s,a)\sum_{s'}\gamma P^a_{ss'}\frac{\partial V^\pi(s')}{\partial\theta}\right]\end{aligned} \tag{4}$$

so

$$\begin{aligned}\frac{\partial V^\pi(s')}{\partial\theta} &= \frac{\partial[\sum_{a'} \pi(s',a')Q^\pi(s',a')]}{\partial\theta} \\ &= \sum_{a'} \left[\frac{\partial\pi(s',a')}{\partial\theta}Q^\pi(s',a') + \pi(s',a')\frac{\partial Q^\pi(s',a')}{\partial\theta}\right] \\ &= \sum_{a'} \left[\frac{\partial\pi(s',a')}{\partial\theta}Q^\pi(s',a') + \pi(s',a')\sum_{s''}\gamma P^{a'}_{s's''}\frac{\partial V^\pi(s'')}{\partial\theta}\right]\end{aligned} \tag{5}$$

so

$$\begin{aligned}\frac{\partial V^\pi(s'')}{\partial\theta} &= \frac{\partial[\sum_{a''} \pi(s'',a'')Q^\pi(s'',a'')]}{\partial\theta} \\ &= \sum_{a''} \left[\frac{\partial\pi(s'',a'')}{\partial\theta}Q^\pi(s'',a'') + \pi(s'',a'')\frac{\partial Q^\pi(s'',a'')}{\partial\theta}\right] \\ &= \sum_{a''} \left[\frac{\partial\pi(s'',a'')}{\partial\theta}Q^\pi(s'',a'') + \pi(s'',a'')\sum_{s'''}\gamma P^{a''}_{s''s'''}\frac{\partial V^\pi(s''')}{\partial\theta}\right]\end{aligned} \tag{6}$$

so

Google Scholar [https://scholar.google.com/citations?hl=en&user=-pdAvr4AAAAJ&view_op=list_works&sortby=pubdate]

$$\begin{aligned}\frac{\partial V^{\pi}(s''')}{\partial\theta} &= \frac{\partial[\sum_{a'''}\pi(s''',a''')Q^{\pi}(s''',a''')]}{\partial\theta} \\ &= \sum_{a'''}\left[\frac{\partial\pi(s''',a''')}{\partial\theta}Q^{\pi}(s''',a''') + \pi(s''',a''')\frac{\partial Q^{\pi}(s''',a''')}{\partial\theta}\right] \\ &= \sum_{a'''}\left[\frac{\partial\pi(s''',a''')}{\partial\theta}Q^{\pi}(s''',a''') + \pi(s''',a''')\sum_{s''''}\gamma P^{a'''}_{s'''s''''}\frac{\partial V^{\pi}(s'''')}{\partial\theta}\right] \\ &\cdots\end{aligned} \tag{7}$$

Substitute (7) into (6) get 76, then, substitute 76 into (5), then, substitute 765 into (4),

$$\begin{aligned}\frac{\partial V^{\pi}(s)}{\partial\theta} &= \frac{\partial}{\partial\theta}\left[\sum_{a}\pi(s,a)Q^{\pi}(s,a)\right] \\ &= \sum_{a}\left[\frac{\partial\pi(s,a)}{\partial\theta}Q^{\pi}(s,a) + \pi(s,a)\frac{\partial Q^{\pi}(s,a)}{\partial\theta}\right] \\ &= \sum_{x}\sum_{k=0}^{\infty}\gamma^{k}\Pr(s\to x,k,\pi)\sum_{a}\frac{\partial\pi(x,a)}{\partial\theta}Q^{\pi}(x,a).\end{aligned}$$

(http://blogs.cuit.columbia.edu/zp2130/files/2019/02/rl_policy_gradient_equation.jpeg) Invalid Equation

so

$$\begin{aligned}\frac{\partial\rho}{\partial\theta} &= \frac{\partial}{\partial\theta}E\left\{\sum_{t=1}^{\infty}\gamma^{t-1}r_t \mid s_0,\pi\right\} = \frac{\partial V^{\pi}}{\partial\theta}(s_0) \\ &= \sum_{s}\sum_{k=0}^{\infty}\gamma^{k}Pr(s_0\to s,k,\pi)\sum_{a}\frac{\partial\pi(s,a)}{\partial\theta}Q^{\pi}(s,a)\end{aligned}$$

assume in start-state formulation:

$$d^{\pi}(s) = \sum_{k=0}^{\infty}\gamma^{k}Pr(s_0\to s,k,\pi)$$

Google Scholar (https://scholar.google.com/citations?hl=en&user=-pdAvr4AAAAJ&view_op=list_works&sortby=pubdate)

$$\begin{aligned}\frac{\partial \rho}{\partial \theta} &= \frac{\partial}{\partial \theta} E\left\{\sum_{t=1}^{\infty} \gamma^{t-1} r_t \mid s_0, \pi\right\} = \frac{\partial V^{\pi}}{\partial \theta}(s_0) \\ &= \sum_s \sum_{k=0}^{\infty} \gamma^k Pr(s_0 \to s, k, \pi) \sum_a \frac{\partial \pi(s,a)}{\partial \theta} Q^{\pi}(s,a) \\ &= \sum_s \sum_{k=0}^{\infty} \gamma^k Pr(s_0 \to s, k, \pi) \sum_a \frac{\partial \pi(s,a)}{\partial \theta} Q^{\pi}(s,a) \\ &= \sum_s d^{\pi}(s) \sum_a \frac{\partial \pi(s,a)}{\partial \theta} Q^{\pi}(s,a)\end{aligned}$$

(Q.E.D)

Stationary Distribution

平稳分布 (http://blogs.cuit.columbia.edu/zp2130/files/2019/02/平稳分布.pdf)

殊途同归

$$\pi = \pi P^n$$

or

$$\pi = \pi P$$

where

P: 转移概率矩阵

π: 平稳概率分布

例：设**状态空间**为S={0, 1, 2,}的马尔可夫链，其**一步转移概率矩阵**为

$$P = \begin{bmatrix} 0.5 & 0.4 & 0.1 \\ 0.3 & 0.4 & 0.3 \\ 0.2 & 0.3 & 0.5 \end{bmatrix}$$

试分析它的极限分布，平稳分布是否存在？并计算

Google Scholar [https://scholar.google.com/citations?hl=en&user=-pdAvr4AAAAJ&view_op=list_works&sortby=pubdate]

解：易知此链为不可约遍历链。

故极限分布存在，平稳分布存在唯一，且平稳分布就是其极限分布。

$$\begin{cases} \pi = \pi P \\ \pi_0 + \pi_1 + \pi_2 = 1 \end{cases} \Rightarrow \begin{aligned} \pi_0 &= \frac{21}{62} \\ \pi_1 &= \frac{23}{62} \\ \pi_2 &= \frac{18}{62} \end{aligned}$$

$$\Rightarrow \pi = (\pi_0, \pi_1, \pi_2) = \left(\frac{21}{62}, \frac{23}{62}, \frac{18}{62}\right)$$

用结果验证，

$$\begin{aligned} \pi P &= \left(\frac{21}{62}, \frac{23}{62}, \frac{18}{62}\right) \begin{bmatrix} 0.5 & 0.4 & 0.1 \\ 0.3 & 0.4 & 0.3 \\ 0.2 & 0.3 & 0.5 \end{bmatrix} \\ &= \left(\frac{21}{62} \cdot 0.5 + \frac{23}{62} \cdot 0.3 + \frac{18}{62} \cdot 0.2,\ \frac{21}{62} \cdot 0.4 + \frac{23}{62} \cdot 0.4 + \frac{18}{62} \cdot 0.3,\ \frac{21}{62} \cdot 0.1 + \frac{23}{62} \cdot 0.3 + \frac{18}{62} \cdot 0.5\right) \\ &= \left(\frac{12}{62}, \frac{23}{62}, \frac{18}{62}\right) \\ &= \pi \end{aligned}$$

$$\pi = \pi P$$

也就是说，可以将求平稳分布与求特征向量相“联系”起来。

$$\begin{aligned} \pi &= \pi P \\ \pi^T &= P^T \pi^T \\ \lambda x &= Ax \\ x &= Ax \end{aligned}$$

(http://blogs.cuit.columbia.edu/zp2130/files/2019/02/SD_eigen_STPMt.gif) Invalid Equation

Google Scholar [https://scholar.google.com/citations?hl=en&user=-pdAvr4AAAAJ&view_op=list_works&sortby=pubdate]

(A：i阶方阵，对应状态空间S的一步**转移概率矩阵的转置**，

λ: **A**的特征值，在这里为1，

x: 非零向量，对应**转移概率矩阵的转置的特征值为1的特征向量**，即**平稳分布**)

结论：求某状态空间S的马尔可夫链的**平稳分布**也就是
求其一步转移概率矩阵P^T的特征值为1的特征向量。

在状态空间S中，考虑到所有的动作a，进入到下一个状态S'，在本论文中平稳分布是d^π，根据以上有关**状态空间S的平稳分布**的说明，$\pi = \pi P$，则可以得出以下关系式：

For the average-reward formulation:

$$\sum_S d^\pi(s) \sum_a \pi(s,a) \sum_{S'} P^a_{ss'} = \sum_{S'} d^\pi(s')$$

Stationary distribution d^π, so sum of probability equals 1:

$$\sum_S d^\pi(s) = 1$$

$$\frac{\partial \rho(\pi)}{\partial \theta} \ \ is\ independent\ of\ s, \sum_S d^\pi(s) = 1$$

$$\sum_S d^\pi(s) \frac{\partial \rho}{\partial \theta} = 1 \cdot \frac{\partial \rho}{\partial \theta}$$

so

$$\sum_s d^\pi(s) \frac{\partial \rho}{\partial \theta} = \sum_s d^\pi(s) \sum_a \frac{\partial \pi(s,a)}{\partial \theta} Q^\pi(s,a) + \sum_{s'} d^\pi(s') \frac{\partial V^\pi(s')}{\partial \theta} - \sum_s d^\pi(s) \frac{\partial V^\pi(s)}{\partial \theta}$$

$$\frac{\partial \rho}{\partial \theta} = \sum_s d^\pi(s) \sum_a \frac{\partial \pi(s,a)}{\partial \theta} Q^\pi(s,a)$$

Google Scholar [https://scholar.google.com/citations?hl=en&user=-pdAvr4AAAAJ&view_op=list_works&sortby=pubdate]

(Q.E.D)

Methods that learn approximations to both **policy** and **value** functions are often called **actor–critic** methods, where '**actor**' is a reference to the **learned policy**, and '**critic**' refers to the **learned value function**, usually a **state-value function**.

1. Policy Gradient Theorem

Theorem 1 (Policy Gradient)

For any MDP, in either the average-reward or start-state formulations,

	average-reward formulation	start-state formulation
$\rho(\pi)$	$\pi(s,a,\theta) = Pr\{a_t = a \mid s_t = s, \theta\}$ $\rho(\pi) = lim_{n\to\infty}\frac{1}{n}E\{r_1 + r_2 + ... + r_n \mid \pi\}$ $= \sum_s d^\pi(s)\sum_a \pi(s,a)R_s^a$ $d^\pi(s) = lim_{n\to\infty}Pr\{s_t = s \mid s_0, \pi\}$	$\pi(s,a,\theta) = Pr\{a_t = a \mid s_t = s, \theta\}$ $\rho(\pi) = E\left\{\sum_{t=1}^{\infty}\gamma^{t-1}r_t \mid s_0, \pi\right\}$ $d^\pi(s) = \sum_{t=0}^{\infty}\gamma^t Pr\{s_t = s \mid s_0, \pi\}$ Define $d^\pi(s)$ as a **discounted** weighting of states encountered **starting at s_0** and then following π
$Q^\pi(s,a)$	$Q^\pi(s,a) = \sum_{t=1}^{\infty}E\{r_t - \rho(\pi) \mid s_0 = s, a_0 = a, \pi\}, \forall s \in S, a \in A$	$Q^\pi(s,a) = E\left\{\sum_{t=1}^{\infty}\gamma^{t-1}r_{t+k} \mid s_t = s, a_t = a, \pi\right\}$
Policy Gradient	$\frac{\partial\rho}{\partial\theta} = \sum_s d^\pi(s)\sum_a \frac{\partial\pi(s,a)}{\partial\theta}Q^\pi(s,a)$	$\frac{\partial\rho}{\partial\theta} = \sum_s d^\pi(s)\sum_a \frac{\partial\pi(s,a)}{\partial\theta}Q^\pi(s,a)$

Google Scholar [https://scholar.google.com/citations?hl=en&user=-pdAvr4AAAAJ&view_op=list_works&sortby=pubdate]

In any event, the key aspect of both expressions for the **gradient** is that their are no terms of the form $\frac{\partial d^{\pi}(s)}{\partial \theta}$:

the effect of **policy changes** on the **distribution of states** does **not appear**.

换句话说就是，**策略变化**对于**状态分布没有影响。**

2. Policy Gradient with Approximation

Theorem 2 (Policy Gradient with Function Approximation)

If f_w satisfies

$$\sum_s d^{\pi}(s) \sum_a \pi(s,a)[Q^{\pi}(s,a) - f_{\omega}(s,a)] \frac{\partial f_{\omega}(s,a)}{\partial \omega} = 0 \quad (2a)$$

这里简单提下公式(2a)的来源，其实就是学习的近似值 f_w（对应值函数的真实值Q^{π}），通过策略π，通过下式的规则

$$\begin{aligned} \triangle \omega_t &\propto \frac{\partial}{\partial \omega}\left[\hat{Q}^{\pi}(s_t,a_t) - f_{\omega}(s_t,a_t)\right]^2 \\ &\propto \left[\hat{Q}^{\pi}(s_t,a_t) - f_{\omega}(s_t,a_t)\right] \frac{\partial}{\partial \omega} f_{\omega}(s_t,a_t) \end{aligned}$$

来更新ω，（近似值 f_w 与真实值Q^{π}的差的平方求ω偏导成正比），上式**红色部分**，当过程收敛到一个局部最佳，得到(2a)。

---

and is **compatible** with the policy parameterization in the sense that

$$\frac{\partial f_{\omega}(s,a)}{\partial \omega} = \frac{\partial \pi(s,a)}{\partial \theta} \cdot \frac{1}{\pi(s,a)} \quad (2b)$$

这个兼容条件compatibility condition很重要，起到'桥梁'作用，可能是由发现者从**结论"反推"**得到的

then Google Scholar [https://scholar.google.com/citations?hl=en&user=-pdAvr4AAAAJ&view_op=list_works&sortby=pubdate]

$$\frac{\partial \rho}{\partial \theta} = \sum_s d^\pi(s) \sum_a \frac{\partial \pi(s,a)}{\partial \theta} f_\omega(s,a) \quad (2c)$$

Proof:

Combining (2a) and (2b),

$$\sum_s d^\pi(s) \sum_a \pi(s,a)[Q^\pi(s,a) - f_\omega(s,a)] \frac{\partial \pi(s,a)}{\partial \theta} \cdot \frac{1}{\pi(s,a)} = 0$$

so

$$\sum_s d^\pi(s) \sum_a [Q^\pi(s,a) - f_\omega(s,a)] \frac{\partial \pi(s,a)}{\partial \theta} = 0$$

$$\sum_s d^\pi(s) \sum_a \frac{\partial \pi(s,a)}{\partial \theta} [Q^\pi(s,a) - f_\omega(s,a)] = 0 \quad (2d)$$

so

we use the theorem 1 – equation (2d)

$$\frac{\partial \rho}{\partial \theta} = \sum_s d^\pi(s) \sum_a \frac{\partial \pi(s,a)}{\partial \theta} Q^\pi(s,a)$$

get

$$\begin{aligned}
\frac{\partial \rho}{\partial \theta} &= \sum_s d^\pi(s) \sum_a \frac{\partial \pi(s,a)}{\partial \theta} Q^\pi(s,a) \\
&= \sum_s d^\pi(s) \sum_a \frac{\partial \pi(s,a)}{\partial \theta} Q^\pi(s,a) - 0 \\
&= \sum_s d^\pi(s) \sum_a \frac{\partial \pi(s,a)}{\partial \theta} Q^\pi(s,a) - \sum_s d^\pi(s) \sum_a \frac{\partial \pi(s,a)}{\partial \theta} [Q^\pi(s,a) - f_\omega(s,a)] \\
&= \sum_s d^\pi(s) \sum_a \frac{\partial \pi(s,a)}{\partial \theta} f_\omega(s,a)
\end{aligned}$$

(Q.E.D)

Google Scholar [https://scholar.google.com/citations?hl=en&user=-pdAvr4AAAAJ&view_op=list_works&sortby=pubdate]

3. Application to Deriving Algorithm and Advantages

Consider a policy that is a Gibbs distribution in a linear combination of features:

$$\pi(s,a) = \frac{e^{\theta^T \phi_{sa}}}{\sum_b e^{\theta^T \phi_{sb}}}$$

so

$$\begin{aligned}
\frac{\partial}{\partial \theta}\pi(s,a) \cdot \frac{1}{\pi(s,a)} &= \frac{\partial}{\partial \theta}\frac{e^{\theta^T \phi_{sa}}}{\sum_b e^{\theta^T \phi_{sb}}} \cdot \frac{1}{\pi(s,a)} \\
&= \frac{\phi_{sa} e^{\theta^T \phi_{sa}}(\sum_b e^{\theta^T \phi_{sb}}) - e^{\theta^T \phi_{sa}}(\sum_b \phi_{sb} e^{\theta^T \phi_{sb}})}{(\sum_b e^{\theta^T \phi_{sb}})^2} \cdot \frac{\sum_b e^{\theta^T \phi_{sb}}}{e^{\theta^T \phi_{sa}}} \\
&= \phi_{sa} - \frac{\sum_b e^{\theta^T \phi_{sb}} \phi_{sb}}{\sum_b e^{\theta^T \phi_{sb}}} \\
&= \phi_{sa} - \sum_b \pi(s,b)\phi_{sb}
\end{aligned}$$

so

$$f_\omega(s,a) = \omega^T \left[\phi_{sa} - \sum_b \pi(s,b)\phi_{sb} \right]$$

也就是说，除了**每个状态normalized为均值0（为什么？）**之外，f_w还与策略同样的特征必须是线性关系。

In other words, f_w must be linear in the same features as the policy, except **normalized to be mean zero (why?) for each state**.

---

4. Convergence of Policy Iteration with Function Approximation

Theorem 3 (Policy Iteration with Function Approximation)

π, f_w 分别是任何的**某策略和某价值函数**的可微的近似者函数。同时它们满足公式（2b）即**兼容条件**，序列

Google Scholar [https://scholar.google.com/citations?hl=en&user=-pdAvr4AAAAJ&view_op=list_works&sortby=pubdate]

$$\{\rho(\pi_k)\}_{k=0}^{\infty}$$

由下面定义：任何θ_0，$\pi_k = \pi(.\ ,.\ ,\theta_k)$, and

$$\omega_k = \omega\ such\ that$$

$$\sum_s d^{\pi_k}(s) \sum_a \pi_k(s,a)[Q^{\pi_k}(s,a) - f_\omega(s,a)] \frac{\partial f_\omega(s,a)}{\partial \omega} = 0$$ 对应(2a)

$$\theta_{k+1} = \theta_k + \alpha_k \sum_s d^{\pi_k}(s) \sum_a \frac{\partial \pi_k(s,a)}{\partial \theta} f_\omega(s,a)$$对应(2c)

收敛至

$$lim_{k\to\infty} \frac{\partial \rho(\pi_k)}{\partial \theta} = 0$$

意味着存在局部最优

Soft max function, Softargmax, or Normalized Exponential Function

归一化指数函数

$$\pi(s,a) = \frac{e^{\theta^T \phi_{sa}}}{\sum_b e^{\theta^T \phi_{sb}}}$$

where ϕ_{sa}: an L-dimensional **feature vector** characterizing **state-action pair** s, a. **表征状态-动作对的一个L维特征向量。**

$\theta^T \phi_{sa}$: the inner product of θ and ϕ_{sa}.

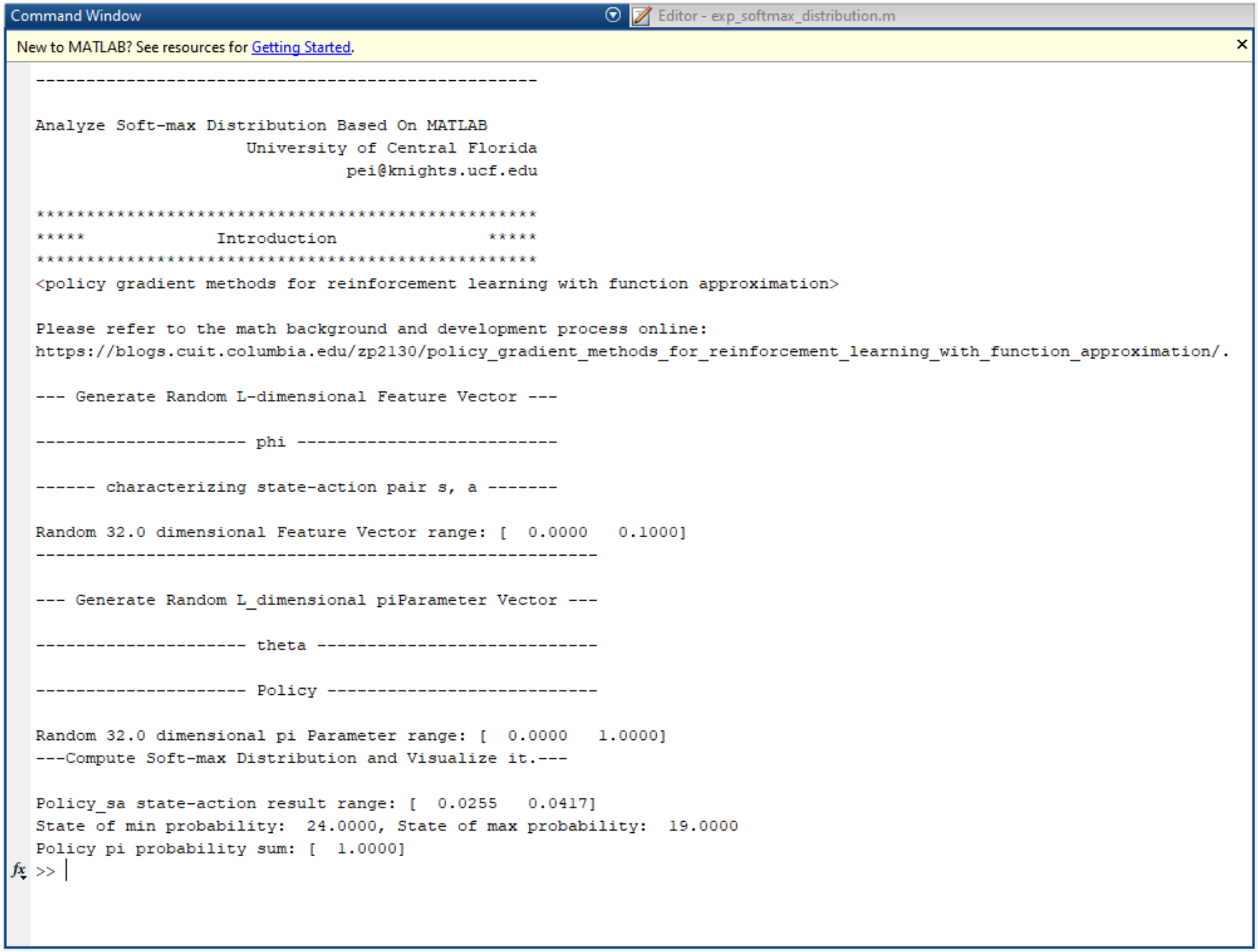
Softmax Distribution Matlab Code@github Private Repository (https://github.com/p9i/exp_softmax_distribution)

Exponential Soft-max Distribution

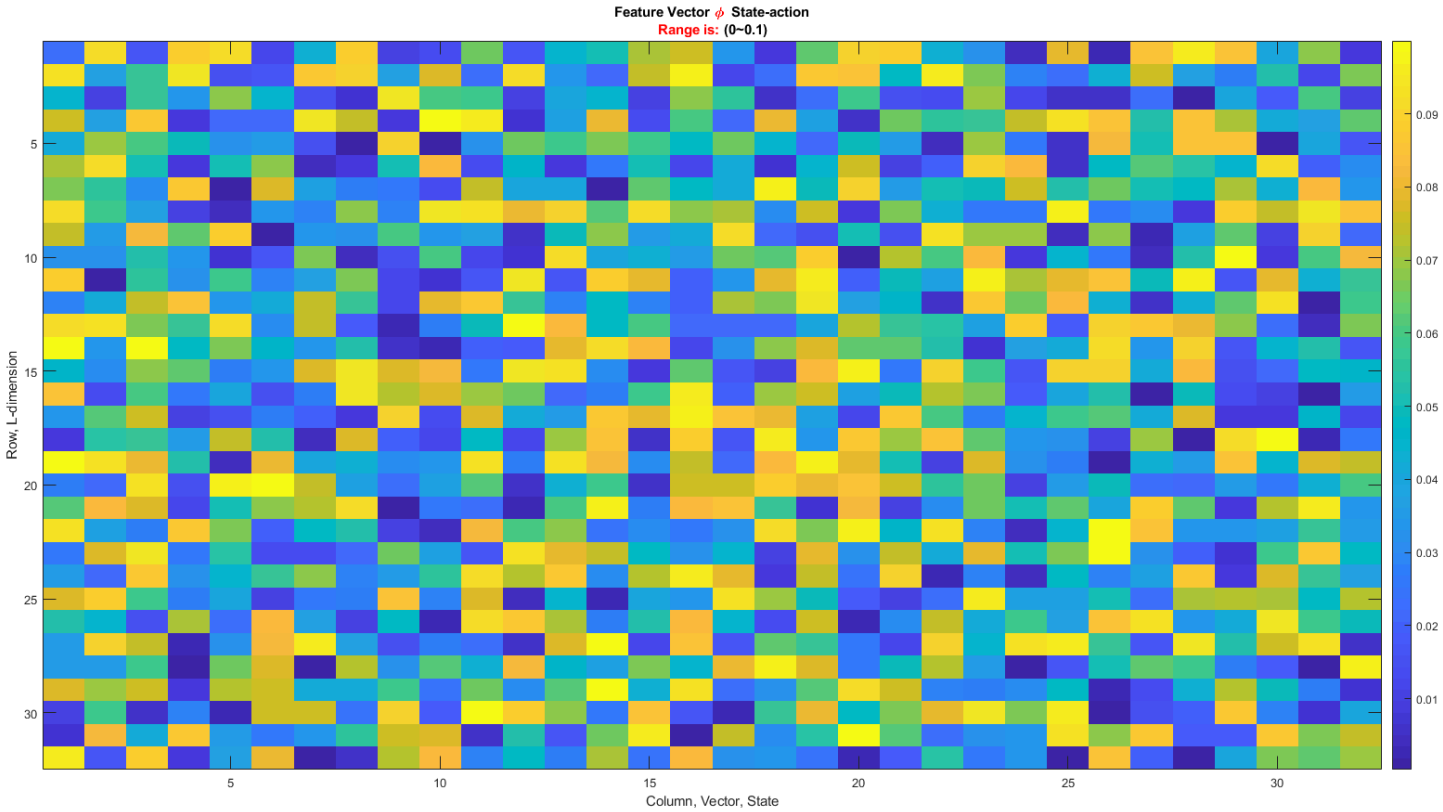
Column is the vector, state.

Google Scholar [https://scholar.google.com/citations?hl=en&user=-pdAvr4AAAAJ&view_op=list_works&sortby=pubdate]

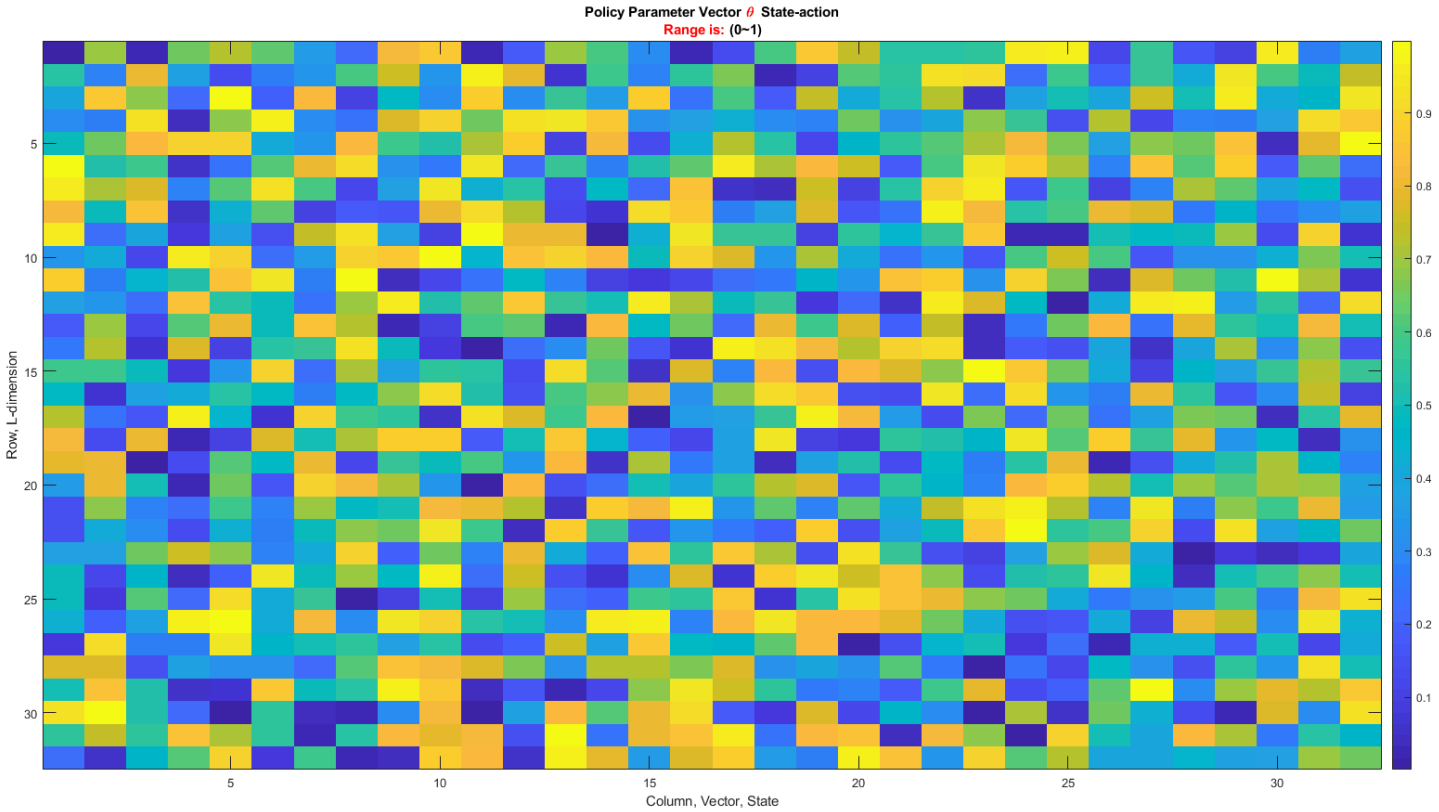
Google Scholar (https://scholar.google.com/citations?hl=en&user=-pdAvr4AAAAJ&view_op=list_works&sortby=pubdate)



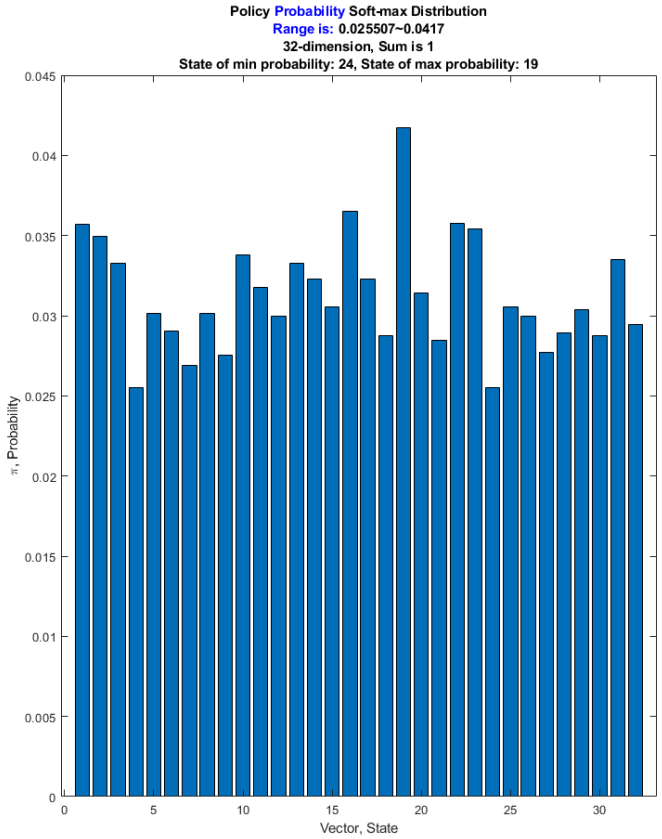
Google Scholar (https://scholar.google.com/citations?hl=en&user=-pdAvr4AAAAJ&view_op=list_works&sortby=pubdate)

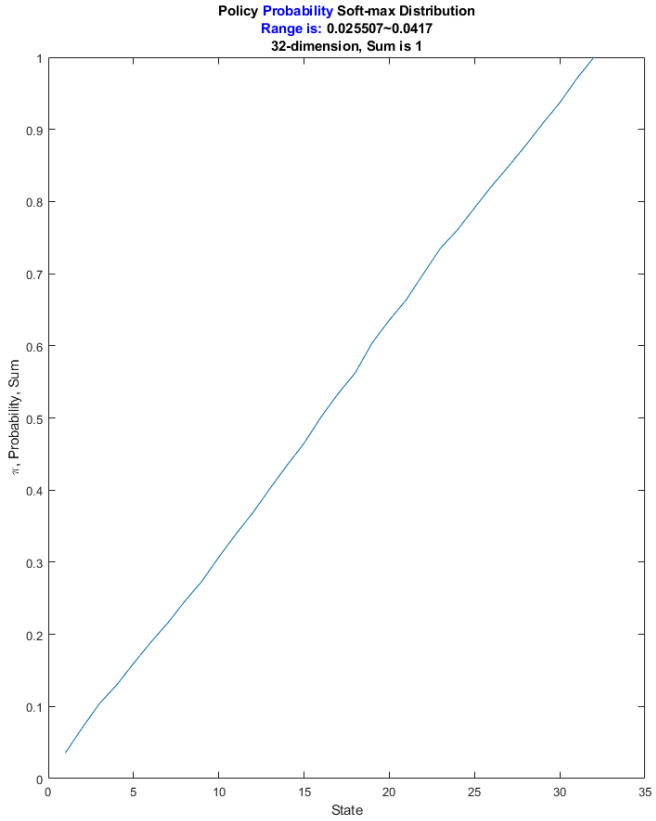


Google Scholar (https://scholar.google.com/citations?hl=en&user=-pdAvr4AAAAJ&view_op=list_works&sortby=pubdate)



Google Scholar (https://scholar.google.com/citations?hl=en&user=-pdAvr4AAAAJ&view_op=list_works&sortby=pubdate)





Major Components of an RL Agent

- An RL agent may include one or more of these components:
  - Policy: agent's behaviour function
  - Value function: how good is each state and/or action
  - Model: agent's representation of the environment

Google Scholar (https://scholar.google.com/citations?hl=en&user=-pdAvr4AAAAJ&view_op=list_works&sortby=pubdate)

Policy

- A policy is the agent's behaviour
- It is a map from state to action, e.g.
- Deterministic policy: $a = \pi(s)$
- Stochastic policy: $\pi(a|s) = \mathbb{P}[A_t = a|S_t = s]$

Value Function

- Value function is a prediction of future reward
- Used to evaluate the goodness/badness of states
- And therefore to select between actions, e.g.

$$v_\pi(s) = \mathbb{E}_\pi \left[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \dots \mid S_t = s\right]$$

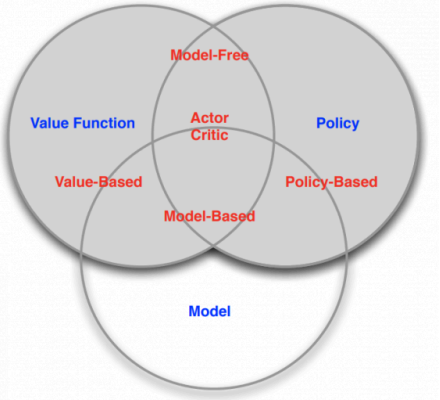
Model

- A model predicts what the environment will do next
- $\mathcal{P}$ predicts the next state
- $\mathcal{R}$ predicts the next (immediate) reward, e.g.

$$\mathcal{P}^a_{ss'} = \mathbb{P}[S_{t+1} = s' \mid S_t = s, A_t = a]$$
$$\mathcal{R}^a_s = \mathbb{E}\left[R_{t+1} \mid S_t = s, A_t = a\right]$$

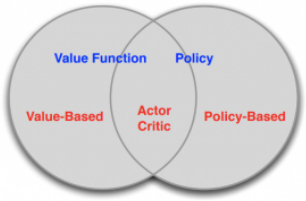
Google Scholar (https://scholar.google.com/citations?hl=en&user=-pdAvr4AAAAJ&view_op=list_works&sortby=pubdate)

RL Agent Taxonomy



Value-Based and Policy-Based RL

- Value Based
  - Learnt Value Function
  - Implicit policy (e.g. ϵ-greedy)
- Policy Based
  - No Value Function
  - Learnt Policy
- Actor-Critic
  - Learnt Value Function
  - Learnt Policy



Google Scholar (https://scholar.google.com/citations?hl=en&user=-pdAvr4AAAAJ&view_op=list_works&sortby=pubdate)

http://blogs.cuit.columbia.edu/zp2130/policy_gradient/

Policy Gradient and Q-learning

RL两大类算法的本质区别？（Policy Gradient 和 Q-learning) Q-learning 是一种基于值函数估计的强化学习方法，Policy Gradient是一种策略搜索强化学习方法。两者是求解强化学习问题的不同方法，如果熟悉监督学习，前者可类比Naive Bayes——通过估计后验概率来得到预测，后者可类比SVM——不估计后验概率而直接优化学习目标。 回答问题：1. 这两种方法的本质上是否是一样的（解空间是否相等）？比如说如果可以收敛到最优解，那么对于同一个问题它们一定会收敛到一样的情况？ 两者是不同的求解方法，而解空间（策略空间）不是由求解方法确定的，而是由策略模型确定的。两者可以使用相同的模型，例如相同大小的神经网络，这时它们的解空间是一样的。 Q-learning在离散状态空间中理论上可以收敛到最优策略，但收敛速度可能极慢。在使用函数逼近后（例如使用神经网络策略模型）则不一定。Policy Gradient由于使用梯度方法求解非凸目标，只能收敛到不动点，不能证明收敛到最优策略。 2. 在Karpathy的blog中提到说更多的人更倾向于Policy Gradient，那么它们两种方法之间一些更细节的区别是什么呢？ 基于值函数的方法（Q-learning, SARSA等等经典强化学习研究的大部分算法）存在策略退化问题，即值函数估计已经很准确了，但通过值函数得到的策略仍然不是最优。这一现象类似于监督学习中通过后验概率来分类，后验概率估计的精度很高，但得到的分类仍然可能是错的，例如真实正类后验概率为 0.501，如果估计为0.9，虽然差别有0.3，如果估计为0.499，虽然差别只有0.002，但分类确是错的。 尤其是当强化学习使用值函数近似时，策略退化现象非常常见。可见 Tutorial on Reinforcement Learning slides中的例子。 Policy Gradient不会出现策略退化现象，其目标表达更直接，求解方法更现代，还能够直接求解stochastic policy等等优点更加实用。 （3. 有人愿意再对比一下action-critic就更好了(: Actor-Critic就是在求解策略的同时用值函数进行辅助，用估计的值函数替代采样的reward，提高样本利用率。 ———————— 作者：ForABiggerWorld 来源：CSDN 原文：https://blog.csdn.net/zjucor/article/details/79200630 版权声明：本文为博主原创文章，转载请附上博文链接！

Dr. Pei

Google Scholar [https://scholar.google.com/citations?hl=en&user=-pdAvr4AAAAJ&view_op=list_works&sortby=pubdate]

Download: pdf (http://blogs.cuit.columbia.edu/zp2130/files/2019/02/rl_2.pdf)

edit (https://blogs.cuit.columbia.edu/zp2130/wp-admin/post.php?post=4486&action=edit)

Author: Z Pei (https://blogs.cuit.columbia.edu/zp2130/author/zp2130/) on February 17, 2019

Categories: AI (https://blogs.cuit.columbia.edu/zp2130/category/ai/), Function Approximation (https://blogs.cuit.columbia.edu/zp2130/category/function-approximation/), Policy Gradient Methods (https://blogs.cuit.columbia.edu/zp2130/category/policy-gradient-methods/), Reinforcement Learning (https://blogs.cuit.columbia.edu/zp2130/category/reinforcement-learning/), RL (https://blogs.cuit.columbia.edu/zp2130/category/rl/), Stationary Distribution (https://blogs.cuit.columbia.edu/zp2130/category/stationary-distribution/)

Tags: AI (https://blogs.cuit.columbia.edu/zp2130/tag/ai/), Function Approximation (https://blogs.cuit.columbia.edu/zp2130/tag/function-approximation/), Policy Gradient Methods (https://blogs.cuit.columbia.edu/zp2130/tag/policy-gradient-methods/), Reinforcement Learning (https://blogs.cuit.columbia.edu/zp2130/tag/reinforcement-learning/), RL (https://blogs.cuit.columbia.edu/zp2130/tag/rl/), Stationary Distribution (https://blogs.cuit.columbia.edu/zp2130/tag/stationary-distribution/)

Other posts

Metric spaces (https://blogs.cuit.columbia.edu/zp2130/metric_spaces/) «» Actor-Critic Algorithms (https://blogs.cuit.columbia.edu/zp2130/actor-critic_algorithms/)

Last posts

- Neuromorphic Computing (https://blogs.cuit.columbia.edu/zp2130/neuromorphic_computing/)
- Circuit Elements (https://blogs.cuit.columbia.edu/zp2130/circuit_elements/)
- Columbia VLSI (https://blogs.cuit.columbia.edu/zp2130/columbia_vlsi/)
- Resume (https://blogs.cuit.columbia.edu/zp2130/resume/)
- Why does inserting or removing buffers help fix timing violations? (https://blogs.cuit.columbia.edu/zp2130/why_does_inserting_or_removing_buffers_help_fix_timing_violations/)
- Software and Hardware vs Time by Grok (https://blogs.cuit.columbia.edu/zp2130/software_and_hardware_vs_time_by_grok/)
- Technology Node vs Year (https://blogs.cuit.columbia.edu/zp2130/technology_node_vs_year/)
- Symbolic Netlist to Innovus-friendly Netlist (https://blogs.cuit.columbia.edu/zp2130/symbolic_netlist_to_innovus-friendly_netlist/)
- Finite-Sample Convergence Rates for Q-Learning and Indirect Algorithms (https://blogs.cuit.columbia.edu/zp2130/finite-sample_convergence_rates_for_q-learning_and_indirect_algorithms/)

Google Scholar (https://scholar.google.com/citations?hl=en&user=-pdAvr4AAAAJ&view_op=list_works&sortby=pubdate)

- Solving H-horizon, Stationary Markov Decision Problems In Time Proportional To Log(H) (https://blogs.cuit.columbia.edu/zp2130/paul_tseng_1990/)

© **Dr. Pei (https://blogs.cuit.columbia.edu/zp2130)** | powered by the WikiWP theme (http://wikiwp.com) and WordPress (http://wordpress.org/). | RSS (https://blogs.cuit.columbia.edu/zp2130/feed/)

Google Scholar (https://scholar.google.com/citations?hl=en&user=-pdAvr4AAAAJ&view_op=list_works&sortby=pubdate)