1 LECTURE OUTLINE

1 00:00:30,310 –> 00:00:35,300 okay let's get started let me welcome
2 00:00:35,300 –> 00:00:38,690 you to this last lecture last time the
3 00:00:38,690 –> 00:00:41,480 six lectures when approximate dynamic
4 00:00:41,480 –> 00:00:45,860 programming I have a feeling that my the
5 00:00:45,860 –> 00:00:49,250 sound is not very good let me just make
6 00:00:49,250 –> 00:00:56,239 an adjustment I'll put it over here and
7 00:00:56,239 –> 00:00:57,670 see what happens
8 00:00:57,670 –> 00:01:01,990 okay is that are you guys doing okay
9 00:01:01,990 –> 00:01:03,790 okay
10 00:01:03,790 –> 00:01:06,170 okay I think this a little better at
11 00:01:06,170 –> 00:01:08,710 least it sounds my ears a little better
开场白，没啥用，就不翻译了
12 00:01:08,710 –> 00:01:11,330 so this is the last lecture lecture six
13 00:01:11,330 –> 00:01:14,450 out of in an approximate dynamic programming
这是近似动态规划的第六次课，也是最后一次课
14 00:01:14,450 –> 00:01:18,440 and we have discussed quite
15 00:01:18,440 –> 00:01:22,970 a few topics so far and mostly having to
16 00:01:22,970 –> 00:01:26,330 do with policy duration cost function approximations
我们讨论了几个问题，主要包括策略迭代和成本函数近似
17 00:01:26,330 –> 00:01:29,479 today we're going to look
18 00:01:29,479 –> 00:01:32,380 at Q factors and model free
19 00:01:32,380 –> 00:01:34,310 implementations of approximate dynamic
20 00:01:34,310 –> 00:01:37,369 programming algorithms
今天我要讲 Q 函数和近似动态规划算法的无模型求解
21 00:01:37,369 –> 00:01:40,280 so we are going to review Q factors in the corresponding bellman equations
我要对 bellman 方程相关的 Q 函数进行一下综述
22 00:01:40,280 –> 00:01:43,910 then talk about value
23 00:01:43,910 –> 00:01:47,390 and policy duration for Q factors
然后讨论 Q 函数的值迭代和策略迭代
24 00:01:47,390 –> 00:01:50,899 and then discuss Q learning it's an
25 00:01:50,899 –> 00:01:53,450 important algorithm that can be viewed
26 00:01:53,450 –> 00:01:56,619 as a sampled version of value iteration
然后是 Q 学习，一种采样视角的值迭代
27 00:01:56,619 –> 00:02:00,679 you can do Q learning for Q factors
28 00:02:00,679 –> 00:02:02,420 but there is no corresponding algorithms
29 00:02:02,420 –> 00:02:04,340 are great for costs that's an interesting fact
你可以使用 Q 值运行 Q 学习，但是这俩不是一个东西，这个事情很有意思
30 00:02:04,340 –> 00:02:07,039 then we are going to
31 00:02:07,039 –> 00:02:09,949 talk about Q learning in combination
32 00:02:09,949 –> 00:02:12,620 with cost function approximation then
33 00:02:12,620 –> 00:02:15,110 say a few things about adaptive dynamic
34 00:02:15,110 –> 00:02:16,459 programming a few things about
35 00:02:16,459 –> 00:02:19,340 approximation policy space
然后我们要讲一些与成本函数近似结合的 Q 学习，然后是自适应动态规划中近似策略空间的内容
36 00:02:19,340 –> 00:02:19,700 and then we are
37 00:02:19,700 –> 00:02:23,450 to have a more general discussion to close the course
然后我们会讲一些比较通用的内容来结束这个课程

2 REVIEW

2.1 DISCOUNTED MDP

38 00:02:23,450 –> 00:02:30,200 so let's do a little review
我要简单回顾一下

39 00:02:30,200 –> 00:02:36,099 we continue to talk about a

40 00:02:36,099 –> 00:02:40,519 finite state system involving states 1

41 00:02:40,519 –> 00:02:43,459 up to n a finite set of controls for

42 00:02:43,459 –> 00:02:46,549 each state I the system is defined by

43 00:02:46,549 –> 00:02:48,680 transition probabilities that depend on control

我要谈论的是一个状态从 1 到 n 的有限状态系统，每一个阶段的控制都是一个有限集合，系统的转移概率依赖于控制

44 00:02:48,680 –> 00:02:51,860 and we consider policies which

45 00:02:51,860 –> 00:02:53,930 are sequences of functions from state to

46 00:02:53,930 –> 00:02:59,120 control nu 0 mu 1 and so on

考虑的策略是一个状态的函数序列，从 μ_0，μ_1 等等

47 00:02:59,120 –> 00:03:03,349 and given a policy starting at a state I we can

48 00:03:03,349 –> 00:03:05,840 consider the sequence of generated costs

49 00:03:05,840 –> 00:03:09,560 over time discounted by a factor of by a discount factor alpha

对于一个给定的起始状态 i 和策略，我可以考虑根据时间生成一个序列成本，这些成本被一个折扣因子 α 衰减

50 00:03:09,560 –> 00:03:13,579 and take the limit

51 00:03:13,579 –> 00:03:17,090 of the series as n goes to infinity this

52 00:03:17,090 –> 00:03:19,069 defines a number for every policy in every state

对于每一个策略和每一个状态，求 n 趋于无穷的时候的数值

53 00:03:19,069 –> 00:03:21,769 so J pi is a cost function

J_π 是一个成本函数

54 00:03:21,769 –> 00:03:23,209 we want to find the minimal cost

55 00:03:23,209 –> 00:03:25,489 function in a policy pi that attains the minimum

我想要找到一个策略 π 下的最小成本函数

56 00:03:25,489 –> 00:03:27,650 we have seen the importance

57 00:03:27,650 –> 00:03:30,109 of stationary policies policies where

58 00:03:30,109 –> 00:03:32,420 all the mutates are the same equal to some mu

我们需要知道平稳策略的重要性，平稳策略就是每一个阶段都用同一个策略 μ，也就是时间无关的策略

59 00:03:32,420 –> 00:03:37,910 and optimal optimal policies

60 00:03:37,910 –> 00:03:39,560 can be find among the stationary

61 00:03:39,560 –> 00:03:41,480 policies in those of the stationary

62 00:03:41,480 –> 00:03:43,430 policies and in the various algorithms

63 00:03:43,430 –> 00:03:46,930 such as policy duration and so on

用各种各样的算法，比如策略迭代什么的就可以在平稳策略中找到最优策略

64 00:03:46,930 –> 00:03:49,519 let me remind you also of our standard

65 00:03:49,519 –> 00:03:52,639 notation which we use occasionally to

66 00:03:52,639 –> 00:03:55,370 denote the dynamic programming mapping

67 00:03:55,370 –> 00:03:56,810 and the corresponding mapping

68 00:03:56,810 –> 00:03:59,930 corresponding policies Maps j Park

69 00:03:59,930 –> 00:04:02,120 functions or n-dimensional vectors in

70 00:04:02,120 –> 00:04:04,040 this case into other n-dimensional

71 00:04:04,040 –> 00:04:09,560 vectors TJ or T mu use of J

我要提醒你我们用过的标准符号, 表示动态规划映射，包括映射，相应的策略，函数 J 和 n 维向量，在这个例子中，另一个 n 维向量 TJ 或者 $T_\mu J$

2.2 BELLMAN EQUATIONS FOR Q-FACTORS

72 00:04:09,560 –> 00:04:13,639 and now let me remind you about Q factors we

73 00:04:13,639 –> 00:04:16,160 discussed those learning the second

74 00:04:16,160 –> 00:04:20,450 lecture so first the second lecture

我要提醒一下你关于 Q 函数，我们在第二个课程的时候讨论过了

75 00:04:20,450 –> 00:04:25,479 the given J star the optimal cost function

给定了最后成本函数 J^*

76 00:04:25,479 –> 00:04:30,860 the optimal Q factor at the state I can

77 00:04:30,860 –> 00:04:32,180 control you

78 00:04:32,180 –> 00:04:35,660 is given by this expression

状态 i 处控制 u 的最优 Q 函数被如下表达式定义

79 00:04:35,660 –> 00:04:38,919 it's the expected one stage cost starting from I

80 00:04:38,919 –> 00:04:44,600 using you plus the optimal cost to go

81 00:04:44,600 –> 00:04:48,410 starting from J so it's cost associated

82 00:04:48,410 –> 00:04:51,560 will start me state I but using you

83 00:04:51,560 –> 00:04:57,070 first and then optimal after that okay

他是一阶段成本加上后续最优长期成本 J 的期望，所以成本在当前状态与控制给定后最小化后续成本获得的

84 00:04:57,400 –> 00:05:00,560 so this is a bigger function than J star

这是一个比 J^* 更大的函数

85 00:05:00,560 –> 00:05:03,590 J star is defined for every step you

86 00:05:03,590 –> 00:05:05,180 started to find for every state and control

J^* 是定义在每一个阶段的，需要做的就是找每一个阶段的最优控制

87 00:05:05,180 –> 00:05:10,669 okay now we have balanced

88 00:05:10,669 –> 00:05:15,020 equation J star satisfying equal to this expression here

我们有 bellman 方程，J^* 也需要满足这个方程 (这一页第一个方程)

89 00:05:15,020 –> 00:05:18,080 therefore if we minimize

90 00:05:18,080 –> 00:05:21,169 on both sides of this expression we see

91 00:05:21,169 –> 00:05:23,930 that J starts of I is the minimum of

92 00:05:23,930 –> 00:05:26,570 the Q factors so if I have the optimal

93 00:05:26,570 –> 00:05:28,880 few factors I can obtain the optimal

94 00:05:28,880 –> 00:05:33,160 cost by minimizations over you

因此最小化这个表达式的等号两边，我们看到 $J^*(i)$ 是状态 i 下所有 u 中最小的 $Q(i,u)$

95 00:05:33,160 –> 00:05:36,169 moreover if I substitute this expression here in

96 00:05:36,169 –> 00:05:39,110 place of j* I have an equation that

97 00:05:39,110 –> 00:05:42,380 does not involve a star at all

如果我用这个表达式 ($J^*(i)=\min\limits_{u\in U(i)} Q^*(i,u)$) 代替上面的表达式中的 J^*，就可以得到一个完全不含有 J^* 的表达式

98 00:05:42,380 –> 00:05:46,130 it's an equation for Q factors

这是一个 Q 值的方程

99 00:05:46,130 –> 00:05:51,789 this equation holds for every I and you

这个方程对所有的 i 和 u 都成立

100 00:05:51,789 –> 00:05:54,889 so it is a number of nonlinear equations with equal number of unknowns

所以这是一个方程数与未知数数相等的非线性方程组

101 00:05:54,889 –> 00:05:57,740 but a bigger equation

102 00:05:57,740 –> 00:05:59,479 that Bellman's equation has more

103 00:05:59,479 –> 00:06:05,330 variables in more equations

但是 bellman 方程组规模更大，变量数量更多

104 00:06:05,330 –> 00:06:07,490 okay now we can use this equation also as a bellman equation

我们把这个方程也叫做 bellman 方程

105 00:06:07,490 –> 00:06:10,430 we can view Q star has been the

106 00:06:10,430 –> 00:06:13,159 fixed point of a certain mapping

我们用 Q^* 表示这个映射的不动点

107 00:06:13,159 –> 00:06:16,190 okay and the mapping is the one here on the right-hand side

这个映射在这个公式 (本页第二个公式) 的等号右边的内容

108 00:06:16,190 –> 00:06:20,810 so it is really a

109 00:06:20,810 –> 00:06:24,310 bellman equation such this one here and

110 00:06:24,310 –> 00:06:27,050 inherits the same properties as other bellman equations

这是一个 bellman 方程，与其他 bellman 方程具备的性质相同

111 00:06:27,050 –> 00:06:30,289 in fact you can

112 00:06:30,289 –> 00:06:34,370 consider i u as a state of a bigger

113 00:06:34,370 –> 00:06:39,889 system and we move from I you to other J

114 00:06:39,889 –> 00:06:41,490 u Prime

115 00:06:41,490 –> 00:06:43,650 and the minimum over here denotes the

116 00:06:43,650 –> 00:06:46,380 minimization over control

事实上你可以把 i，u 当成一个更大的系统的状态，然后从状态 (i，u) 转移到另一个状态 (i,u')，然后找到合适的 u 最小化新状态的 Q 值

117 00:06:46,380 –> 00:06:50,220 so there is an underlying dynamic programming problem
118 00:06:50,220 –> 00:06:53,069 on a bigger space that involves the Q
119 00:06:53,069 –> 00:06:55,860 factors in involves of states the pairs I nu
这是一个底层的在更大空间上的动态规划问题，状态空间是 (i，u)
120 00:06:55,860 –> 00:06:59,160 and the bellman equation for that
121 00:06:59,160 –> 00:07:01,349 problem is this one
这个是这个问题的 bellman 方程
122 00:07:01,349 –> 00:07:03,479 so everything we have said about bellman equations for J
123 00:07:03,479 –> 00:07:08,550 star or J mu applies to Q factors as well
J^* 和 J_μ 也作用在这个 Q 值上了
124 00:07:08,550 –> 00:07:11,550 so there's a unique solution to
125 00:07:11,550 –> 00:07:13,590 this equation this map here is a
126 00:07:13,590 –> 00:07:15,690 contraction mapping a lot of cool stuff
127 00:07:15,690 –> 00:07:18,139 that we discussed earlier apply
这是这个映射 (中间的红公式) 的唯一解，它是一个压缩映射，我们之前讨论过很多关于压缩映射的东西
128 00:07:18,139 –> 00:07:21,330 and there is also similar mapping describing
129 00:07:21,330 –> 00:07:24,690 the Q factors of a policy and let me get into those
这是一个相似的映射，Q 值的策略，我们来讲一下

2.3 BELLMAN EQ FOR Q-FACTORS OF A POLICY

130 00:07:24,690 –> 00:07:29,220 given a stationary policy
给定一个平稳策略
131 00:07:29,220 –> 00:07:31,830 we can consider some larger system that
132 00:07:31,830 –> 00:07:35,490 moves from IU to our use of J and then
133 00:07:35,490 –> 00:07:38,150 continues according to this mechanism
可以考虑在一个大系统中，根据这个转移机制 (概率) 从状态 (i,u) 转移到状态 $(u, \mu(j))$
134 00:07:38,150 –> 00:07:42,680 and the Q factors of a policy satisfy
135 00:07:42,680 –> 00:07:49,380 this equation
这个策略的 Q 值满足这个等式 (从上往下第一个公式)
136 00:07:49,380 –> 00:07:52,199 equivalently q mu is a fixed point of this mapping
Q_μ 是这个映射 (第二个公式) 的不动点
137 00:07:52,199 –> 00:07:54,960 this is a linear equation now
这是一个线性方程 (第一个方程)
138 00:07:54,960 –> 00:07:56,849 has all equations that have to do with single policies are
139 00:07:56,849 –> 00:07:59,340 a linear equation involving these
140 00:07:59,340 –> 00:08:02,370 unknowns defined for every policy in
141 00:08:02,370 –> 00:08:04,680 having an unknown for every I and u
所有的方程都是对于同一个策略来说的，这个线性方程 (第一个) 的未知数包括策略和 (i，u)
142 00:08:04,680 –> 00:08:08,490 and we can use this equation we can
143 00:08:08,490 –> 00:08:13,710 solve this equation we can evaluate the
144 00:08:13,710 –> 00:08:16,680 Q factors of a policy
我们可以评价这个策略的 Q 值
145 00:08:16,680 –> 00:08:20,060 and that's useful in policy direction
这可以用在策略迭代中
(someone asking
146 00:08:23,070 –> 00:08:35,820 I didn't understand yes it's a number of
147 00:08:35,820 –> 00:08:55,440 states that's true yes yes yes this in
148 00:08:55,440 –> 00:08:58,560 this is only one of the several possible
149 00:08:58,560 –> 00:09:01,020 transitions from I you okay
150 00:09:01,020 –> 00:09:04,920 it's a several address there are n
151 00:09:04,920 –> 00:09:17,610 possibilities yes okay P IJ you are the
152 00:09:17,610 –> 00:09:20,970 transition probabilities starting at I
153 00:09:20,970 –> 00:09:24,450 going to J over you will be using those
154 00:09:24,450 –> 00:09:29,360 for six lectures so so okay that's fine
asking completed)
155 00:09:34,430 –> 00:09:39,210 okay so we have Q factor optimal Q

156 00:09:39,210 –> 00:09:41,840 factors Q factors of a policy
我们求得了某个策略的 Q 值
157 00:09:41,840 –> 00:09:44,190 and generally you value direction in policy
158 00:09:44,190 –> 00:09:47,070 direction can be carried out in terms of Q factors
一般你可以用 Q 函数进行值迭代和策略迭代
159 00:09:47,070 –> 00:09:54,600 now if you do value duration
160 00:09:54,600 –> 00:09:56,970 for Q factors in all policy direction
161 00:09:56,970 –> 00:09:58,980 for Q factors you're not getting
162 00:09:58,980 –> 00:10:01,040 anything different than for course
163 00:10:01,040 –> 00:10:03,240 mathematically they're equivalent
如果你对 Q 值进行值迭代或者对 Q 值进行策略迭代，他们没什么区别，从数学上说他们是等价的
164 00:10:03,240 –> 00:10:05,340 you just operate on other things you're
165 00:10:05,340 –> 00:10:07,710 you're operating on you're using a
166 00:10:07,710 –> 00:10:10,250 transform versions of the same algorithm
你只是从不同的角度来看待这个算法
167 00:10:10,250 –> 00:10:16,170 so for exact so for exact policy
168 00:10:16,170 –> 00:10:18,960 evaluation policy direction and value
169 00:10:18,960 –> 00:10:21,920 iteration there is no difference between
170 00:10:21,920 –> 00:10:26,340 using two factors or policies except for a few things
所以对于精确策略迭代和值迭代来说，用 Q 值或者策略除了一些东西几乎没什么不同的

2.4 WHAT IS GOOD AND BAD ABOUT Q-FACTORS

171 00:10:26,340 –> 00:10:28,320 so let that leads us to a
172 00:10:28,320 –> 00:10:28,900 question
173 00:10:28,900 –> 00:10:31,390 why do we want to use Q factors what's
174 00:10:31,390 –> 00:10:35,440 good and what's bad about them
下面我们来讨论一个问题，为什么要用 Q 值，他有什么好处和坏处呢
175 00:10:35,440 –> 00:10:38,170 first of all all the exact theory and algorithms
176 00:10:38,170 –> 00:10:41,080 for costs applies to Q factors
首先，所有的精确理论和算法都用 Q 值计算成本
177 00:10:41,080 –> 00:10:42,760 bellman equations all the things that we were
178 00:10:42,760 –> 00:10:45,160 discussing the last in the preceding
179 00:10:45,160 –> 00:10:46,600 lectures contraction mappings
180 00:10:46,600 –> 00:10:47,920 contraction properties optimality
181 00:10:47,920 –> 00:10:50,320 conditions convergence and value of
182 00:10:50,320 –> 00:10:52,150 valuing policy direction all of these things apply
bellman 中的所有事情，我们之前讨论过的内容，压缩映射，收缩性，最优条件，收敛性，值迭代和策略迭代，都在用 Q 值
183 00:10:52,150 –> 00:10:56,020 all the approximate theory
184 00:10:56,020 –> 00:10:58,480 that we discussed also applies projected
185 00:10:58,480 –> 00:11:01,110 equation sampling exploration issues
186 00:11:01,110 –> 00:11:05,260 oscillation aggregation and all of these
187 00:11:05,260 –> 00:11:08,230 phenomena manifest themselves in Q
188 00:11:08,230 –> 00:11:11,850 factor land just as well as in cost land
我们讨论过的所有的近似理论也都在用，投影方程，采样，探索，震荡，聚合，所有的理论都在用 Q 值计算成本
189 00:11:11,850 –> 00:11:15,010 now here's one big thing about Q factors
现在关于 Q 值有一个很重要的事情
190 00:11:15,010 –> 00:11:18,850 it allows if you can find them a
191 00:11:18,850 –> 00:11:22,260 model free controller implementation
如果你能够找到一个模型无关的控制器
192 00:11:22,260 –> 00:11:25,420 once we calculate these Q factors for all I and u
如果我能计算所有 (i,u) 的 Q 值
193 00:11:25,420 –> 00:11:28,210 we can calculate an
194 00:11:28,210 –> 00:11:31,630 optimal policy not by minimizing on the
195 00:11:31,630 –> 00:11:33,570 right hand side of the bellman equation

196 00:11:33,570 –> 00:11:36,760 but rather by just looking at this list
197 00:11:36,760 –> 00:11:38,860 of Q factors and selecting the one
198 00:11:38,860 –> 00:11:40,540 that's numerically smaller
那么就不用通过最小化 bellman 方程来计算最优策略了，可以直接在 Q 值列表中找一个值最小的控制就行了
199 00:11:40,540 –> 00:11:43,720 okay so implementation once you have
200 00:11:43,720 –> 00:11:47,230 these Q stars is very simple
如果你能找到 Q^*，实施起来就非常简单
201 00:11:47,230 –> 00:11:49,510 and moreover I do not need to know the
202 00:11:49,510 –> 00:11:51,779 transition probabilities of the system
更多地，我不需要知道系统的状态转移概率
203 00:11:51,779 –> 00:11:55,060 ordinarily you need to minimize the
204 00:11:55,060 –> 00:11:58,240 expected value of the current stage cost
205 00:11:58,240 –> 00:12:01,720 plus the expected optimal costs ago that
206 00:12:01,720 –> 00:12:04,110 involves calculating an expectation and
207 00:12:04,110 –> 00:12:06,339 involves knowing the transition probabilities
按理说你需要最小化当前成本和最优长期成本的和的期望，同时还得知道状态转移概率
208 00:12:06,339 –> 00:12:08,500 here you don't need to
209 00:12:08,500 –> 00:12:10,360 know the transition probabilities you
210 00:12:10,360 –> 00:12:12,580 don't need to know the model
这里你不需要知道转移概率，也不需要知道模型
211 00:12:12,580 –> 00:12:16,089 if you can find these q stars okay
如果你能找到 Q^*
212 00:12:16,089 –> 00:12:20,620 that's the big incentive for using q factors
这是大家用 Q 值得一个很大的原因
213 00:12:20,620 –> 00:12:23,200 similarly if you calculate a parametric
214 00:12:23,200 –> 00:12:25,630 approximation of the q factors involved
215 00:12:25,630 –> 00:12:28,240 in some approximation architecture with
216 00:12:28,240 –> 00:12:31,420 a parameter vector r you can obtain a
217 00:12:31,420 –> 00:12:34,870 suboptimal policy by minimizing
218 00:12:34,870 –> 00:12:37,510 again these Q factors without the need for a model
相似地，如果你要计算一个 Q 值使用参数向量 r 和相应近似结构得参数近似问题，你可以不知道模型直接最小化 Q 来得到次优策略
219 00:12:37,510 –> 00:12:42,640 what are the bad things
220 00:12:42,640 –> 00:12:44,050 about Q factors
那么 Q 值得缺点是什么呢
221 00:12:44,050 –> 00:12:47,320 well greater dimension more storage and
222 00:12:47,320 –> 00:12:50,170 some complications in calculating these or that
更高的维度，更大的存储空间和更大的计算量
223 00:12:50,170 –> 00:12:55,390 that's a rough cut about what
224 00:12:55,390 –> 00:12:59,050 what we can expect from two factors in
225 00:12:59,050 –> 00:13:00,880 particular if you want to calculate
226 00:13:00,880 –> 00:13:04,240 exactly factors then the difficult is
227 00:13:04,240 –> 00:13:05,980 increased for large-scale problems and
228 00:13:05,980 –> 00:13:08,230 you have to think about aggregation or
229 00:13:08,230 –> 00:13:10,839 other approximation procedures in order
230 00:13:10,839 –> 00:13:19,690 to calculate Q factors
这是一个策略的划分，如果你想要计算精确的 Q 值，大规模会增加求解的难度，这时候你就不得不考虑聚合或者其他近似方法来求解 Q 值

3 Q-LEARNING

231 00:13:19,690 –> 00:13:21,190 okay now here's one thing about Q factors that we
232 00:13:21,190 –> 00:13:22,630 don't have for cause it's an algorithm called Q learning
下面要讲的关于 Q 值得算法叫做 Q 学习
233 00:13:22,630 –> 00:13:28,330 in this is in addition
234 00:13:28,330 –> 00:13:30,070 to everything else we discuss this is new
我们接下来要讲的是一个新东西

235 00:13:30,070 –> 00:13:33,880 and Q learning is a sampled form of value iteration

这是值迭代的采样形式

236 00:13:33,880 –> 00:13:36,460 it is a stochastic

237 00:13:36,460 –> 00:13:39,420 iterative algorithm

也是一个随机迭代算法

238 00:13:39,420 –> 00:13:42,850 that generates iterated values of these Q factors by

239 00:13:42,850 –> 00:13:47,310 using samples from the system

这个算法使用从系统中获取的样本对 Q 值进行迭代

240 00:13:47,310 –> 00:13:50,230 and it resembles value generation but it's not

241 00:13:50,230 –> 00:13:52,600 value duration it's a sampled form of value iteration

它很像值迭代，但是又不是值迭代，而是一种采样形式的值迭代

242 00:13:52,600 –> 00:13:56,020 okay so here's the

243 00:13:56,020 –> 00:13:58,270 description of Q learning

这是 Q 学习的介绍

244 00:13:58,270 –> 00:13:59,740 in the classical form there are many variations

245 00:13:59,740 –> 00:14:01,600 but this is the standard Q learning algorithm

Q 学习有很多不同的形式，这是标准 Q 学习

246 00:14:01,600 –> 00:14:05,040 remember we want to calculate

247 00:14:05,040 –> 00:14:09,970 Q of all I and you so we select a

248 00:14:09,970 –> 00:14:12,810 sequence of pairs of states and controls

249 00:14:12,810 –> 00:14:15,880 we have a system sample we get one state

250 00:14:15,880 –> 00:14:17,440 control pair here another one another

251 00:14:17,440 –> 00:14:20,170 one we have a whole large number of them

我们想要计算所有 (i,u) 对应的 Q 值，我们先选择一个序列的状态动作对，从系统中采样可以获得非常多的状态动作对 '

252 00:14:20,170 –> 00:14:23,050 and we can use any mechanism for doing

253 00:14:23,050 –> 00:14:25,660 that probabilistic mechanism or

254 00:14:25,660 –> 00:14:28,870 deterministic mechanism as long as all

255 00:14:28,870 –> 00:14:32,980 the state control pairs are represented

256 00:14:32,980 –> 00:14:36,850 and are sampled infinitely often was

257 00:14:36,850 –> 00:14:38,800 this an idealized algorithm practice we

258 00:14:38,800 –> 00:14:40,720 would not sound every state control pair

259 00:14:40,720 –> 00:14:42,900 infinitely often but that's the

260 00:14:42,900 –> 00:14:44,380 theoretical

261 00:14:44,380 –> 00:14:49,260 version of the algorithm

我们可以用任何机制进行采样，概率机制或者确定性机制来获得无数个状态动作对，实际上我们不能获得无数个状态动作对，但是这是算法从理论上的要求

262 00:14:49,260 –> 00:14:53,170 so a iteration case we had some two

263 00:14:53,170 –> 00:14:56,560 factors QK a whole big vector of Q factors

迭代是我们获得了巨大的 Q 值向量中的一部分

264 00:14:56,560 –> 00:15:00,370 and we update we said we have a

265 00:15:00,370 –> 00:15:04,360 sample we have this ik uk and we

266 00:15:04,360 –> 00:15:06,790 sample the next state according to this

267 00:15:06,790 –> 00:15:10,560 transition probabilities so we have I K

268 00:15:10,560 –> 00:15:16,060 UK and JK and we have we an and we

269 00:15:16,060 –> 00:15:19,180 update the Q factor of that particular

270 00:15:19,180 –> 00:15:22,660 pair according to this formula and we

271 00:15:22,660 –> 00:15:25,380 live only all the other Q factors unchanged

我们使用这个 i_k，u_k 和 j_k 根据这个表达式来更新这些 Q 值，被更新的只有这部分 Q 值，其他的没有变

272 00:15:25,380 –> 00:15:31,900 okay so here i am i select ik

273 00:15:31,900 –> 00:15:35,650 UK and i pick the Q factor of only

274 00:15:35,650 –> 00:15:40,750 that pair and i sample the next state to

275 00:15:40,750 –> 00:15:43,810 obtain JK and i calculate this

276 00:15:43,810 –> 00:15:47,170 expression which is a sample of the

277 00:15:47,170 –> 00:15:50,140 expected value that appears in bellman

278 00:15:50,140 –> 00:15:51,880 equation okay

279 00:15:51,880 –> 00:15:55,090 sample because it's not some over p IJ s

280 00:15:55,090 –> 00:16:01,120 but in rather just j k okay and we move
281 00:16:01,120 –> 00:16:05,080 the Q factor of only that pair in the
282 00:16:05,080 –> 00:16:08,290 direction of this sample but we also
283 00:16:08,290 –> 00:16:11,740 interpolate with a current value gamma K
284 00:16:11,740 –> 00:16:14,500 is a step size typically a small step
285 00:16:14,500 –> 00:16:17,740 size in fact it is necessary but the
286 00:16:17,740 –> 00:16:20,800 step size goes to 0 lightly at a rate of
287 00:16:20,800 –> 00:16:24,850 1 over K proportional to a constant over K
我有 i_k 和 u_k，选择这些状态和控制对应的 Q 值，然后对下一阶段状态进行采样得到 j_k，然后根据这个表达式计算样本的期望值，这里没有概率 p_{ij}，只有 j_k，现在我使用样本在当前的 γ_k 下更新 Q 值，这个步长必须非常小才行，一般与 $\frac{1}{k}$ 成正比
288 00:16:24,850 –> 00:16:31,690 so we calculate the sample which is like value direction
我们用样本来进行这个迭代时很像值迭代
289 00:16:31,690 –> 00:16:36,460 okay but it's only
290 00:16:36,460 –> 00:16:39,840 a sample and then we make this update
291 00:16:39,840 –> 00:16:43,180 leave all the others and change then I
292 00:16:43,180 –> 00:16:47,110 pick out the next sample i ka plus 1 u k
293 00:16:47,110 –> 00:16:51,370 plus 1 i do again a change for that you
294 00:16:51,370 –> 00:16:55,149 factor and keep going like the
只有一个样本时用这个样本更新相应的 Q 值，其他 Q 值不变，然后选择新的状态和控制进行迭代，迭代一直这么进行下去
295 00:16:55,149 –> 00:16:57,589 all the few factors are going to be
296 00:16:57,589 –> 00:17:00,560 updated infinitely often but only one at a time
所有的 Q 值都会被更新无数次，但是每次之更新一个
297 00:17:00,560 –> 00:17:08,770 okay so that's the idea
298 00:17:08,770 –> 00:17:14,329 value direction would move in that
299 00:17:14,329 –> 00:17:17,150 direction fully with step size equal to
300 00:17:17,150 –> 00:17:21,289 one Q learning moves in the direction of
301 00:17:21,289 –> 00:17:27,400 a sample of this by a step size gamma K
这个想法是这样的，值迭代会沿着步长是 1 的方向迭代，而 Q 学习会根据样本以 $gamma_k$ 的步长进行迭代

3.1 NOTES AND QUESTIONS ABOUT Q-LEARNING

302 00:17:37,240 –> 00:17:39,850 so let's say let's before we go into
303 00:17:39,850 –> 00:17:42,400 questions as to why this works and so on
304 00:17:42,400 –> 00:17:44,010 let's just look at it and try to
305 00:17:44,010 –> 00:17:46,450 understand it a little bit better
我们讨论为什么它能工作之前先来看看这个好好理解一下
306 00:17:46,450 –> 00:17:49,240 here is the algorithm I've just restated the algorithm here
我要讲的算法在这里
307 00:17:49,240 –> 00:17:57,400 the first thing is that
308 00:17:57,400 –> 00:18:00,250 to implement the algorithm I don't need
309 00:18:00,250 –> 00:18:02,830 to have a detailed model of the system
首先实现这个算法的时候我不需要知道这个系统的细节
310 00:18:02,830 –> 00:18:05,350 I don't need to have the pIJ probabilities
我不需要知道概率 p_{ij}
311 00:18:05,350 –> 00:18:08,950 all I need is to have a simulator or
312 00:18:08,950 –> 00:18:12,880 some mechanism so that given AI K and UK
313 00:18:12,880 –> 00:18:16,030 I can generate the next state
我需要的是一个给定 i_k 和 u_k 能够生成新状态的模拟器
314 00:18:16,030 –> 00:18:18,580 so all I need is a box that samples next States
315 00:18:18,580 –> 00:18:21,610 and also gives me values of one state cost
所以我需要的是一个盒子，能够给我新状态和一阶段成本
316 00:18:21,610 –> 00:18:30,610 there is no need for a model
不需要知道模型的信息
317 00:18:30,610 –> 00:18:32,950 the other thing is that it operates at only
318 00:18:32,950 –> 00:18:35,020 one state control pair at a time and

319 00:18:35,020 –> 00:18:38,140 this is convenient if you want to
320 00:18:38,140 –> 00:18:40,600 generate the state control pairs by simulation
另一个事情就是每次通过仿真给定一个状态动作对很方便
321 00:18:40,600 –> 00:18:43,960 so you have one to state
322 00:18:43,960 –> 00:18:45,790 control pair that moves you to a next
323 00:18:45,790 –> 00:18:47,950 state and then you use that next state
324 00:18:47,950 –> 00:18:51,970 as as part of the next state control pair and so on
所以你有一个状态控制对，转移到下一个状态，然后继续转移到新状态，这样产生状态动作对
325 00:18:51,970 –> 00:18:55,030 so even though the
326 00:18:55,030 –> 00:18:58,060 mechanism for generating this this state
327 00:18:58,060 –> 00:19:00,460 control pair arbitrary often times
328 00:19:00,460 –> 00:19:02,380 you use us the simulator of the system
329 00:19:02,380 –> 00:19:05,830 and in that context it’s convenient that
330 00:19:05,830 –> 00:19:08,620 you operate at only one state control pair at a time
331 00:19:08,620 –> 00:19:11,110 because that’s consistent with simulation
这种每次通过系统仿真产生一个状态动作对的方法很方便
332 00:19:11,110 –> 00:19:16,060 however with this type
333 00:19:16,060 –> 00:19:18,640 of operation the algorithm becomes a synchronous
这种类型的操作让算法变成一个同步算法
334 00:19:18,640 –> 00:19:21,280 remember we talked about
335 00:19:21,280 –> 00:19:24,910 asynchronous algorithms backing in second
336 00:19:24,910 –> 00:19:27,520 lecture where we operate somewhat
337 00:19:27,520 –> 00:19:30,640 chaotically among the components of the
338 00:19:30,640 –> 00:19:33,550 vector that we update
记得我们在第二次课程讲的异步算法么，迭代更新时成本向量更新得很混乱 (不是一起更新的)
339 00:19:33,550 –> 00:19:36,310 so and one possibility is to update components one at a time
一种可能是每次更新一个元素
340 00:19:36,310 –> 00:19:39,070 well that’s what we do here we
341 00:19:39,070 –> 00:19:42,580 operate on Q factors of only one
342 00:19:42,580 –> 00:19:46,179 component at operate and only one factor at a time
这就是这个地方做的，更新时每次只更新一个 Q 值
343 00:19:46,179 –> 00:19:52,090 now there are no
344 00:19:52,090 –> 00:19:53,980 approximations in this out good we aim
345 00:19:53,980 –> 00:19:56,230 to find the exactly optimal Q factors
我们的目的是找到精确的最优 Q 值
346 00:19:56,230 –> 00:19:59,259 okay we don’t cook multiple suboptimal
347 00:19:59,259 –> 00:20:02,320 here we want to we aiming straight for
348 00:20:02,320 –> 00:20:05,769 the optimal Q factors
我们不想计算次优解，我们的目标就是得到最优 Q 向量
349 00:20:05,769 –> 00:20:09,100 and then the question is why does this work why does
350 00:20:09,100 –> 00:20:12,490 this algorithm converge to Q star the optimal Q factors
问题是为什么这个算法会收敛到最优 Q 值
351 00:20:12,490 –> 00:20:16,389 and also why can I
352 00:20:16,389 –> 00:20:19,179 not use a sample version of value
353 00:20:19,179 –> 00:20:23,700 iteration to calculate optimal costs
和为什么我不用采样版本的值迭代计算最优成本
354 00:20:23,700 –> 00:20:27,850 well the reason for this is actually Mathematica
355 00:20:27,850 –> 00:20:31,720 it’s just a fine point in the mathematics
原因是基于数学的一些比较好的理论
356 00:20:31,720 –> 00:20:35,220 generally speaking for
357 00:20:35,220 –> 00:20:38,460 stochastic iterative algorithms to work
一般来说，随即迭代算法工作时
358 00:20:38,460 –> 00:20:41,950 what you need to have is an expected
359 00:20:41,950 –> 00:20:43,779 value an iteration that involves an
360 00:20:43,779 –> 00:20:46,679 expected value that you can sample
你想要的是通过采样计算成本的期望值
361 00:20:46,679 –> 00:20:50,919 however in bellman’s equation you have a
362 00:20:50,919 –> 00:20:52,570 mapping that does not involve an

363 00:20:52,570 –> 00:20:55,119 expected value only but rather the
364 00:20:55,119 –> 00:20:58,389 minimum of an expected value if I were
365 00:20:58,389 –> 00:21:01,749 to sample this expected value and
366 00:21:01,749 –> 00:21:03,429 consider an algorithm that invoke
367 00:21:03,429 –> 00:21:06,220 minimization over example this would just not work
bellman 方程中你有一个不包括期望只有最小化操作的映射，如果我对这个期望进行采样然后对样本进行最小化，算法就不会工作
368 00:21:06,220 –> 00:21:10,059 the sample of minimum of
369 00:21:10,059 –> 00:21:13,419 minimum works but minimum sample the
370 00:21:13,419 –> 00:21:16,029 mathematics don't work
最小化的采样工作，但是最小化采样不工作
371 00:21:16,029 –> 00:21:18,340 because when you take expected values of a minimum of the
372 00:21:18,340 –> 00:21:19,990 sample you don't get the same as the
373 00:21:19,990 –> 00:21:22,990 minimum of the expected value
因为你计算样本的最小化期望值时，它的值与最小化期望值是不一样的
374 00:21:22,990 –> 00:21:25,269 so it's a fine mathematical point but it makes a
375 00:21:25,269 –> 00:21:27,789 big difference because I would like to
376 00:21:27,789 –> 00:21:30,490 work with I would like to have a sampled
377 00:21:30,490 –> 00:21:34,179 value iteration of involved in cost but
378 00:21:34,179 –> 00:21:36,759 I can't do it because the minimum and
379 00:21:36,759 –> 00:21:38,590 the expectation are in the wrong relation
这是一个数学上很好的概念，但是会导致这两种算法很大的差别，因为我想要进行采样值迭代的时候由于最小化和期望值的原因而不可行
380 00:21:38,590 –> 00:21:44,559 okay we're not going to go back
381 00:21:44,559 –> 00:21:47,470 to this point but instead let's consider
382 00:21:47,470 –> 00:21:49,210 why do we get convergence
我们不会再一次讲这个地方了，我们来考虑一下为什么会收敛吧
383 00:21:49,210 –> 00:21:54,990 I think questions at this point
有什么问题么

3.2 CONVERGENCE ASPECTS OF Q-LEARNING

384 00:22:07,360 –> 00:22:14,500 okay you can show convergence to the
385 00:22:14,500 –> 00:22:16,660 true exact Q factors and the fairly mild assumptions
在这里你可以看到精确 Q 值收敛和比较温和的假设
386 00:22:16,660 –> 00:22:20,220 and the line of proof
387 00:22:20,220 –> 00:22:23,650 involves two types of arguments two
388 00:22:23,650 –> 00:22:26,050 types of theories why is the theory of
389 00:22:26,050 –> 00:22:28,780 stochastic approximation iterations and
390 00:22:28,780 –> 00:22:32,230 the other is the theory of a synchronous algorithms
这一行证明包括两种类型 (包括了两种理论) 的讨论，一种是随机近似迭代，另一种是同步算法理论
391 00:22:32,230 –> 00:22:35,560 mathematically the key fact
392 00:22:35,560 –> 00:22:38,380 is that the q-learning mapping that
393 00:22:38,380 –> 00:22:38,860 map's
394 00:22:38,860 –> 00:22:43,090 Q into FQ is a contraction with respect
395 00:22:43,090 –> 00:22:44,740 to the soup norm okay the maximum norm
从数学上讲，关键的地方在 Q 学习映射上，也就是把 Q 映射到 FQ，这是一个带有 sup-norm 的压缩映射
396 00:22:44,740 –> 00:22:47,620 contraction is
397 00:22:47,620 –> 00:22:49,630 important in order to be able to use the stochastic approximation
为了使用随机近似方法，压缩性是很重要的
398 00:22:49,630 –> 00:22:52,510 sup norm is
399 00:22:52,510 –> 00:22:55,390 important because because of the a synchronous operation
sup norm 重要是因为同步操作
400 00:22:55,390 –> 00:22:57,430 you may remember
401 00:22:57,430 –> 00:22:59,320 on my discussion about a synchronous

402 00:22:59,320 –> 00:23:01,510 algorithms a synchronous values are
403 00:23:01,510 –> 00:23:04,150 valid provided there's some kind of
404 00:23:04,150 –> 00:23:06,430 contraction light property with respect
405 00:23:06,430 –> 00:23:08,410 to the soup norm if you remember this
406 00:23:08,410 –> 00:23:11,740 box conditions and so on
你应该还记得我们讨论过的同步算法，sup norm 为同步值提供了收缩性，还记得那个方框么
407 00:23:11,740 –> 00:23:15,190 that's what in effect here when you look into in depth into the proof
如果你深入地看他的证明，就可以知道它的影响是什么了
408 00:23:15,190 –> 00:23:18,880 soup norm works well with
409 00:23:18,880 –> 00:23:21,070 a synchronous contraction works well
410 00:23:21,070 –> 00:23:25,960 with stochastic approximation okay
sup norm 在同步算法中工作的很好，收缩性在随机近似方法中过的得很好
411 00:23:25,960 –> 00:23:27,940 now let's understand the connection with
412 00:23:27,940 –> 00:23:34,300 stochastic approximation
我们来看看它与随机近似方法的联系
413 00:23:34,300 –> 00:23:36,700 let's step take one step back from Q learning and look
414 00:23:36,700 –> 00:23:39,130 at a generic fixed point problem
415 00:23:39,130 –> 00:23:42,850 involving an expectation
我们回到 Q 学习看一下带期望的通用的不动点问题
416 00:23:42,850 –> 00:23:46,360 we want to find a fixed point of this expected value of
417 00:23:46,360 –> 00:23:50,070 f mapping where W is a random variable
我们想要找到这个以 w 为随机变量的映射 f 的期望值的不动点
418 00:23:50,070 –> 00:23:54,550 okay so in our case of course this
419 00:23:54,550 –> 00:23:56,890 involves a bellman equation so one but
420 00:23:56,890 –> 00:23:59,410 let's in what in greater generality
在这个例子中这个表达式是一个一般性的 bellman 方程，
421 00:23:59,410 –> 00:24:01,330 and let's assume that this mapping is a
422 00:24:01,330 –> 00:24:03,250 contraction mapping with respect to some norm
我们假设这是一个与某些 nrom 相关的压缩映射
423 00:24:03,250 –> 00:24:06,820 so that this equation has a unique
424 00:24:06,820 –> 00:24:09,400 solution and also this iteration
425 00:24:09,400 –> 00:24:13,590 converges to the unique fixed point okay
所以这个方程有一个不动点，迭代过程会收敛与不动点
426 00:24:13,590 –> 00:24:17,710 now how would you work this how do you
427 00:24:17,710 –> 00:24:19,120 modify this algorithm
428 00:24:19,120 –> 00:24:21,760 so that instead of iterating with full
429 00:24:21,760 –> 00:24:24,520 expected values you just use samples sampled approximations
那么我们该如何修改这个算法让他使用采样近似代替迭代时的完全期望

3.3 STOCH. APPROX. CONVERGENCE IDEAS

430 00:24:24,520 –> 00:24:30,700 well we could
431 00:24:30,700 –> 00:24:35,050 generate a sequence of damages and use
432 00:24:35,050 –> 00:24:38,680 the sequence to approximate the expected
433 00:24:38,680 –> 00:24:43,180 value appears in this direction
我们生成了一个序列的样本，然后在迭代中用这个序列去近似期望值
434 00:24:43,180 –> 00:24:48,130 so the duration K to approximate this
435 00:24:48,130 –> 00:24:49,830 expression here this expected value
436 00:24:49,830 –> 00:24:54,070 let's use a Monte Carlo average using
437 00:24:54,070 –> 00:24:56,850 the first k samples okay
第 k 次迭代时用前 k 个样本进行蒙特卡洛平均近似期望值
438 00:24:56,850 –> 00:24:59,410 that's certainly an algorithm that makes sense
这是一个很有用的算法
439 00:24:59,410 –> 00:25:03,310 and as K increases this Monte
440 00:25:03,310 –> 00:25:06,100 Carlo approximation is going to be more
441 00:25:06,100 –> 00:25:08,140 and more accurate
随着 k 的值增加，蒙特卡洛近似越来越精确

442 00:25:08,140 –> 00:25:10,870 so asymptotically you're getting this algorithm in this
443 00:25:10,870 –> 00:25:13,090 it's very simple to show that this kind
444 00:25:13,090 –> 00:25:14,200 of algorithm works
从渐进角度上讲，这就是这个算法，可以很简单地说明它能够工作
445 00:25:14,200 –> 00:25:16,840 it's like iterating with a contraction
446 00:25:16,840 –> 00:25:20,230 fixed point iteration that converges
在迭代过程中收缩，并且收敛到不动点
447 00:25:20,230 –> 00:25:22,060 but there is a problem with this and the
448 00:25:22,060 –> 00:25:24,360 problem is the following
但是这种方法有一些问题，如下
449 00:25:24,360 –> 00:25:29,170 here at every Direction K I have to go
450 00:25:29,170 –> 00:25:31,960 back and calculate this expression for
451 00:25:31,960 –> 00:25:34,600 all the previous samples
每一次迭代 k，我都必须回去对所有之前的样本按照这个表达式进行一次计算
452 00:25:34,600 –> 00:25:37,230 so there's a lot of computation involved here
需要很大的计算量
453 00:25:37,230 –> 00:25:41,530 an increasing amount of averaging as the time progresses
随着迭代进行，计算量越来越大
454 00:25:41,530 –> 00:25:44,890 so we need to compute
455 00:25:44,890 –> 00:25:46,690 all these values here for all the previous WT
所以我们需要对所有 w_t 进行计算
456 00:25:46,690 –> 00:25:50,110 so a more convenient
457 00:25:50,110 –> 00:25:57,520 iteration is to replace this with the
458 00:25:57,520 –> 00:25:59,380 expression that you have already
459 00:25:59,380 –> 00:26:02,380 computed at the duration T
现在有一个更方便地方法，就是你用已经算过的数据来代替之前的数据
460 00:26:02,380 –> 00:26:04,900 that's a major simplification because you don't
461 00:26:04,900 –> 00:26:06,790 have to recompute all of these things
462 00:26:06,790 –> 00:26:10,690 again from time 1 to K but rather you
463 00:26:10,690 –> 00:26:14,520 can use the previous values that you had
这种方法非常简单因为你不需要重新计算所有值，只用之前算过的值就可以了
464 00:26:15,960 –> 00:26:18,790 so this is similar but requires much
465 00:26:18,790 –> 00:26:20,650 less computation because it leaves only
466 00:26:20,650 –> 00:26:25,180 one value of F per sample
这两种方法很相似，但是新的那个需要的计算量更少因为它只留下了一个每个样本的函数值
467 00:26:25,180 –> 00:26:27,010 now it turns out that this algorithm is kind of slow
468 00:26:27,010 –> 00:26:28,960 because it involves average
469 00:26:28,960 –> 00:26:34,929 and and then then the XS do not change
470 00:26:34,929 –> 00:26:38,919 very much over time
这种算法被证实比较慢因为它包括求平均，就导致了 x 的值随时间变化得非常慢
471 00:26:38,919 –> 00:26:41,679 so X of T at previous Direction is fairly close to
472 00:26:41,679 –> 00:26:44,590 the current XK and this thing is close
473 00:26:44,590 –> 00:26:46,779 enough to this so that this replacement is valid
所以之前的迭代中 x_t 与当前的 x_t 非常相近，而且函数 f 的值也非常相近，所以这个替换是有效的
474 00:26:46,779 –> 00:26:50,230 but that's what the successive
475 00:26:50,230 –> 00:26:52,960 approximation algorithm does
这就是近似算法成功的地方
476 00:26:52,960 –> 00:26:56,649 it replaces the naive Monte Carlo approximation with
477 00:26:56,649 –> 00:26:58,720 a more convenient computationally approximation
用一个更方便地近似计算方法来代替原始的蒙特卡洛
478 00:26:58,720 –> 00:27:02,769 now if you take this
479 00:27:02,769 –> 00:27:07,389 expression and you can rewrite it in a
480 00:27:07,389 –> 00:27:09,669 recursive form like this with gamma K equals 1 over K
现在你可以把这个表达式重新写成这样的，γ_k 等于 $\frac{1}{k}$
481 00:27:09,669 –> 00:27:13,960 so that you only need
482 00:27:13,960 –> 00:27:16,690 one sample represented at each direction
这样每次迭代你只需要一个样本就可以了

483 00:27:16,690 –> 00:27:19,419 and now you can see that this iteration
484 00:27:19,419 –> 00:27:22,179 is sort of the same thing as the Q learning direction
现在你可以看到这个迭代方法与 Q 学习迭代方法是一样的
485 00:27:22,179 –> 00:27:25,450 it involves moving at
486 00:27:25,450 –> 00:27:27,450 the edge in the direction of a sample
487 00:27:27,450 –> 00:27:29,980 doing interpolation between the current
488 00:27:29,980 –> 00:27:36,809 point a and the and the the other sample
在当前样本和其他样本间做插值让 x 从当前值变化到新的值
489 00:27:37,559 –> 00:27:40,600 so Q learning is stochastic
490 00:27:40,600 –> 00:27:43,600 approximation if it were done for all
491 00:27:43,600 –> 00:27:46,570 Q factors simultaneously
如果对所有的 Q 值都进行仿真，那么 Q 学习就是一种随机近似方法
492 00:27:46,570 –> 00:27:50,049 it is stochastic approximation with just this F mapping
493 00:27:50,049 –> 00:27:54,399 being this mapping here
这是一种随机近似，迭代公式中的 F 映射 (倒数第二个公式) 也就是这个映射 (最后一个公式)
494 00:27:54,399 –> 00:27:57,909 and using a single sample in here instead of the expected value
用一个样本代替这里面 (最后一个公式) 的期望值
495 00:27:57,909 –> 00:28:02,440 so this provides the
496 00:28:02,440 –> 00:28:04,419 connection between Q learning and stochastic approximation
所以这提供了 Q 学习和随机近似方法的联系
497 00:28:04,419 –> 00:28:08,919 to learning now
498 00:28:08,919 –> 00:28:11,980 has a little bit more into it because
499 00:28:11,980 –> 00:28:14,889 the durations are done 1q factor at that
500 00:28:14,889 –> 00:28:17,620 time but then again this will involve
501 00:28:17,620 –> 00:28:19,600 the theory of a synchronous computation
502 00:28:19,600 –> 00:28:22,600 and that's also what will come into the proof of convergence
现在将这个还有点早，因为每次迭代更新一个 Q 值，涉及同步计算，这部分也会出现在 Q 学习的收敛性证明里
503 00:28:22,600 –> 00:28:25,840 so this is not a
504 00:28:25,840 –> 00:28:28,659 proof of convergence by any means
从任何角度上讲这都不是收敛性的证明
505 00:28:28,659 –> 00:28:31,179 this step has to be justified and also there
506 00:28:31,179 –> 00:28:32,350 are other things that we have omitted
507 00:28:32,350 –> 00:28:34,480 but gives you an idea about the
508 00:28:34,480 –> 00:28:36,600 mathematical connections with other
509 00:28:36,600 –> 00:28:45,650 class of methodologies okay
这些东西都被省略了，我主要是想给你一个想法，关于它与其他方法有什么数学上的联系的

3.4 Q-LEARNING COMBINED WITH OPTIMISTIC PI

510 00:28:49,669 –> 00:28:52,279 let me mention a variation of Q learning
511 00:28:52,279 –> 00:28:55,119 that's computationally more efficient
我来介绍一下各种计算效率更高的 Q 学习
512 00:28:55,119 –> 00:28:57,999 this is the Q learning direction right
这是 Q 学习迭代公式
513 00:28:57,999 –> 00:29:03,859 we pick a pair i k UK and we calculate
514 00:29:03,859 –> 00:29:05,749 this expression and we move in the
515 00:29:05,749 –> 00:29:08,450 direction of that expression
我们选择了 (i_k, u_k) 然后在迭代中计算这个表达式
516 00:29:08,450 –> 00:29:11,479 now this expression involves minimization over all controls
这个表达式包括对这个表达式进行最小化，找到合适的 u
517 00:29:11,479 –> 00:29:14,419 and if you have a lot of
518 00:29:14,419 –> 00:29:16,759 controls this may involve a substantial amount of computation
如果 u 非常多，计算量就会非常大
519 00:29:16,759 –> 00:29:21,200 it would be more
520 00:29:21,200 –> 00:29:24,379 efficient to remember the good controls
521 00:29:24,379 –> 00:29:27,469 instead of minimizing over all the

522 00:29:27,469 –> 00:29:29,629 controls minimize or over the good
523 00:29:29,629 –> 00:29:32,239 controls or perhaps just one control
把所有好的决策记住，不再在所有控制中选择最小化成本的控制而是在好的控制中选择
524 00:29:32,239 –> 00:29:35,239 this leads into the idea of maintaining
525 00:29:35,239 –> 00:29:38,719 a policy a sort of current policy which
526 00:29:38,719 –> 00:29:41,059 you use in here instead of do using a minimum
这就产生了一种想法，维护一个策略，在最小化时使用当前策略选择控制
527 00:29:41,059 –> 00:29:43,879 and if you try to connect it
528 00:29:43,879 –> 00:29:46,849 back to two policy direction and
529 00:29:46,849 –> 00:29:50,690 optimistic policy direction
如果你想要把它与策略迭代和乐观策略迭代联系起来
530 00:29:50,690 –> 00:29:53,359 you may consider a sample version of an
531 00:29:53,359 –> 00:29:55,690 optimistic policy Direction algorithm
你可以考虑采样版本的乐观策略迭代
532 00:29:55,690 –> 00:30:00,259 which evaluates the current policy we
533 00:30:00,259 –> 00:30:02,269 evaluate the Q factors of the current
534 00:30:02,269 –> 00:30:04,999 policy by using a finite number of
535 00:30:04,999 –> 00:30:07,639 values directions which would not be
536 00:30:07,639 –> 00:30:10,369 important in minimization here
使用有限次数的值迭代对策略进行评价，是不是最小化就不重要了
537 00:30:10,369 –> 00:30:12,709 and then improves the policy occasionally after
538 00:30:12,709 –> 00:30:16,329 new K steps by means of this iteration
然后使用这个公式对策略进行改进
539 00:30:16,329 –> 00:30:18,799 so this is a more efficient version of this
这是更有效率的算法
540 00:30:18,799 –> 00:30:21,109 but it turns out not to work
但是事实证明它不工作
541 00:30:21,109 –> 00:30:23,299 there’s an all the work
542 00:30:23,299 –> 00:30:26,749 dating back to 1993 which says that if
543 00:30:26,749 –> 00:30:28,879 you implement Q learning are
544 00:30:28,879 –> 00:30:33,379 synchronously with with maintaining both
545 00:30:33,379 –> 00:30:35,929 a policy and Q factors then you may
546 00:30:35,929 –> 00:30:37,089 get oscillation
追溯到 1993 年的工作就说明了在使用同步算法与策略和 Q 值的方式实现 Q 学习的时候会产生震荡
547 00:30:37,089 –> 00:30:39,589 however it turns out that this can be
548 00:30:39,589 –> 00:30:42,440 rectified by there’s a simple
549 00:30:42,440 –> 00:30:44,570 modification of this algorithm which is
550 00:30:44,570 –> 00:30:46,940 does not involve much overhead that
551 00:30:46,940 –> 00:30:49,489 overcomes this convergence problem
但是事实证明这个问题可以通过修改很少的东西校正这个算法
552 00:30:49,489 –> 00:30:52,519 and this is recent research and I refer you
553 00:30:52,519 –> 00:30:56,179 to a series of papers and I’ll leave it
554 00:30:56,179 –> 00:30:58,419 at that
这些工作可以看这些论文，我把他们给你们了
555 00:31:02,239 –> 00:31:05,519 okay so we have Q learning for exactly
556 00:31:05,519 –> 00:31:08,460 factors we have optimistic versions of Q
557 00:31:08,460 –> 00:31:09,989 learning combinations with policy
558 00:31:09,989 –> 00:31:12,809 iteration all of those for exact Q factors
我们讲了精确的 Q 学习，带有策略迭代的乐观版本的 Q 学习，都是用来计算精确值的

3.5 Q-FACTOR APPROXIMATIONS

559 00:31:12,809 –> 00:31:20,700 however we mentioned that for
560 00:31:20,700 –> 00:31:23,070 large-scale problems the exact form of Q
561 00:31:23,070 –> 00:31:26,669 learning is not is not it cannot be
562 00:31:26,669 –> 00:31:28,379 applied because of the storage and dimension problem
我提到过对于大规模问题精确 Q 学习由于存储空间和维度问题没法用

563 00:31:28,379 –> 00:31:30,690 so let's try to

564 00:31:30,690 –> 00:31:33,330 consider basis function approximations

所以我们考虑基函数近似

565 00:31:33,330 –> 00:31:40,759 so we introduce a Q factor approximation

566 00:31:40,759 –> 00:31:44,669 which is linear and involved this basis

567 00:31:44,669 –> 00:31:46,639 functions phi is a feature vector

568 00:31:46,639 –> 00:31:50,729 associated with I u and R is a of weights vector

我们介绍的近似是一种线性近似，基函数 ϕ 是关于 i 和 u 的特征向量，r 是权重向量

569 00:31:50,729 –> 00:31:56,359 and we can try to train r

570 00:31:56,359 –> 00:31:59,279 by for example approximate policy

571 00:31:59,279 –> 00:32:02,039 direction pick a policy calculate the

572 00:32:02,039 –> 00:32:04,109 corresponding r policy improve and so

573 00:32:04,109 –> 00:32:06,389 on and you can use several language for

574 00:32:06,389 –> 00:32:09,450 policy evaluation like LSTD LSPE and

575 00:32:09,450 –> 00:32:14,940 others that we have discussed

我们尝试样本近似策略迭代训练 r，选择一个策略计算相关的 r 然后进行策略改进，你可以用我们之前讲过的 LSTD，LSPE 什么的方法进行策略评价

576 00:32:14,940 –> 00:32:19,129 very often in practice people use optimistic

577 00:32:19,129 –> 00:32:21,749 versions of the policy duration method

实际操作种人们经常用乐观策略迭代

578 00:32:21,749 –> 00:32:28,589 whereby they have a policy they collect

579 00:32:28,589 –> 00:32:31,229 a few samples maybe one maybe ten maybe

580 00:32:31,229 –> 00:32:32,999 a hundred but not a very large number

581 00:32:32,999 –> 00:32:36,479 and then do an approximate evaluation

582 00:32:36,479 –> 00:32:38,999 based on the current weight the current

583 00:32:38,999 –> 00:32:42,239 policy then do a policy improvement or a

584 00:32:42,239 –> 00:32:45,149 step towards policy improvement collect

585 00:32:45,149 –> 00:32:48,629 more samples and so on

现在有一个策略，收集一些样本，可能一个，可能十个可能一百个，总之不是很多，然后基于当前的权重 r 和策略进行近似策略评价，再进行策略改进，然后继续收集样本评价和改进

586 00:32:48,629 –> 00:32:51,179 so policy evaluation with a few samples policy

587 00:32:51,179 –> 00:32:53,279 improvement by adjusting the weight

588 00:32:53,279 –> 00:32:56,489 vector again more policy evaluation and

589 00:32:56,489 –> 00:32:59,519 with samples and so on

根据已有的样本进行策略评价，通过调整权重向量改进策略，然后继续进行策略评价，这么一直进行下去

590 00:32:59,519 –> 00:33:01,529 the most extreme type of algorithm of this type uses just

591 00:33:01,529 –> 00:33:04,169 a single sample a single sample in

592 00:33:04,169 –> 00:33:07,850 between policy updates

这个算法的极端情况是只用一个样本进行策略的更新

593 00:33:07,850 –> 00:33:11,790 okay so here this algorithm generates a

594 00:33:11,790 –> 00:33:14,490 trajectory of state control pairs as follows

这个算法生成状态控制对的轨迹是这样的

595 00:33:14,490 –> 00:33:18,660 given the current weight vector

596 00:33:18,660 –> 00:33:22,470 that represents an estimate of the okay

597 00:33:22,470 –> 00:33:24,470 together with phi gives you an

598 00:33:24,470 –> 00:33:27,299 approximate cost function of a current policy

给定当前权重向量和当前策略的近似成本函数

599 00:33:27,299 –> 00:33:31,650 you simulate the transition using

600 00:33:31,650 –> 00:33:35,640 this this the transition probabilities

601 00:33:35,640 –> 00:33:39,030 of phi of the system

使用这个系统的 ϕ，也就是转移概率仿真生成状态状态转移样本

602 00:33:39,030 –> 00:33:43,740 then generate a control by minimizing the Q factors at the new point

然后在新的点通过最小化 Q 值产生控制

603 00:33:43,740 –> 00:33:48,990 and using that control you

604 00:33:48,990 –> 00:33:51,960 update the corresponding policy by

605 00:33:51,960 –> 00:33:56,120 changing the the weight vector R and

606 00:33:56,120 –> 00:33:59,880 this is this is done in the direction of

607 00:33:59,880 –> 00:34:04,410 a TD lambda type of iteration correction

608 00:34:04,410 –> 00:34:07,110 or lsbe direction some kind of iterative

609 00:34:07,110 –> 00:34:10,050 algorithm that makes a correction after

610 00:34:10,050 –> 00:34:14,219 a single sample

用这个控制通过改变权重向量 r 的值更新相关的策略，更新方式通过 LSPE 或者 TD 相关的算法给出

611 00:34:14,219 –> 00:34:19,859 so I have a state control pair I use that to generate a

612 00:34:19,859 –> 00:34:25,500 new new state then reevaluate the

613 00:34:25,500 –> 00:34:29,850 control at that state by minimizing over

614 00:34:29,850 –> 00:34:33,239 the Q factors then move again and so on

我有一个状态控制对，用这个产生一个新的状态，通过最小化 Q 值生成一个新的状态控制对，然后估值，改进，估值，一直这么进行下去

615 00:34:33,239 –> 00:34:37,199 a very chaotic type of algorithm with

616 00:34:37,199 –> 00:34:39,418 some connection to theory some

617 00:34:39,418 –> 00:34:43,109 connection to validity but very unclear properties

这种算法很混乱，有的与理论相关，有的与有效性相关，总之没有很明确的性质

618 00:34:43,109 –> 00:34:46,379 very complex behavior and

619 00:34:46,379 –> 00:34:48,929 clear validity oscillations all over the place

620 00:34:48,929 –> 00:34:55,230 but people have used it a lot

有效性与震荡什么的很复杂，但是人们用的很多

621 00:34:55,230 –> 00:34:59,490 they claim claim success with it

他们说这个算法很成功

622 00:34:59,490 –> 00:35:01,050 you have to judge for yourself what kind of success

623 00:35:01,050 –> 00:35:06,450 this is or you know what it means

你需要自己判断这个成功到底是指什么

624 00:35:06,450 –> 00:35:09,359 there is however solid basis for a version of

625 00:35:09,359 –> 00:35:13,619 this algorithm for one for one type of

626 00:35:13,619 –> 00:35:16,790 problem problems of optimal stopping

在最优停止问题上，这个算法还是很有效的

627 00:35:16,790 –> 00:35:19,030 optimal stopping problems we have talked

628 00:35:19,030 –> 00:35:20,980 very much optimal stopping problems

629 00:35:20,980 –> 00:35:22,570 involved just two controls for each

630 00:35:22,570 –> 00:35:25,930 stage either stop or continue okay

我们经常会讨论最优停止问题，每一个阶段都有两个控制，继续或者停止

631 00:35:25,930 –> 00:35:29,320 an example is when you're trying to

632 00:35:29,320 –> 00:35:36,250 optimize optimize the time that you

633 00:35:36,250 –> 00:35:38,140 exercise an option like a financial

634 00:35:38,140 –> 00:35:40,660 option what you exercise the option you

635 00:35:40,660 –> 00:35:42,490 sell the option or you keep it for one more step

636 00:35:42,490 –> 00:35:44,850 you either stop or continue

637 00:35:44,850 –> 00:35:46,060 okay

一个例子就是你试图优化一个金融操作，卖出去或者留着，这就是继续或者停止的决定

638 00:35:46,060 –> 00:35:48,070 turns out that Q learning is used very

639 00:35:48,070 –> 00:35:49,890 widely in practice in financial

640 00:35:49,890 –> 00:35:53,500 engineering financial programs that that

641 00:35:53,500 –> 00:35:58,060 value options and and the optimal

642 00:35:58,060 –> 00:36:00,960 stopping formulation is the one that

643 00:36:00,960 –> 00:36:04,480 comes in there and there are and in this

644 00:36:04,480 –> 00:36:06,730 particular algorithm is valid for this

645 00:36:06,730 –> 00:36:12,210 for this for this problem

事实证明 Q 学习在金融问题中用的非常多，最优停止问题就是这个，上面的方法都可以很有效地用在这个问题上

3.6 BELLMAN EQUATION ERROR APPROACH

646 00:36:17,370 –> 00:36:21,100 okay now let's talk about another type

647 00:36:21,100 –> 00:36:23,710 of algorithm that involves Q factor

648 00:36:23,710 –> 00:36:25,740 approximation

我们来讨论另一个近似 Q 值得方法

649 00:36:25,740 –> 00:36:31,830 instead of using projected equations and

650 00:36:31,830 –> 00:36:34,180 the algorithmic we have discussed this

651 00:36:34,180 –> 00:36:35,680 another approach that we mentioned but

652 00:36:35,680 –> 00:36:37,750 it did not discuss very much the so

653 00:36:37,750 –> 00:36:40,530 called bellman equation error approach

能够代替投影方程但是我们讨论的比较少得算法就是 bellman 误差方法

654 00:36:40,530 –> 00:36:43,360 okay so we'll describe this approach for Q factors

我们要介绍一下这个方法关于 Q 值的版本

655 00:36:43,360 –> 00:36:47,950 given a policy new we

656 00:36:47,950 –> 00:36:51,150 approximate the Q factors of that policy

657 00:36:51,150 –> 00:36:53,880 using a linear approximation

658 00:36:53,880 –> 00:36:57,100 architecture that is obtained by

659 00:36:57,100 –> 00:36:59,980 minimizing the error in satisfying

660 00:36:59,980 –> 00:37:05,430 Bellman's equation okay

给定一个策略 μ，我们用线性结构对这个策略的 Q 值今昔，也就是满足这个最小化 bellman 误差的表达式

661 00:37:05,430 –> 00:37:13,120 so this is Phi R Phi R is the approximation to Q mu

Φr 是 Q_μ 的近似

662 00:37:13,120 –> 00:37:17,380 and F nm is the mapping associated with with

663 00:37:17,380 –> 00:37:20,770 the the policy mu involves expected

664 00:37:20,770 –> 00:37:24,280 value and so on

F_μ 是关于策略 μ 的映射，这个映射包括期望值

665 00:37:24,280 –> 00:37:28,890 so this is a least squares problem

这是一个最小二乘问题

666 00:37:28,890 –> 00:37:32,470 and the norm here is Euclidean work with respect to some

667 00:37:32,470 –> 00:37:35,950 distribution xi

这个范数是关于分布 ξ 的欧几里得范数

668 00:37:35,950 –> 00:37:38,200 this approach is related to somewhere to projected

669 00:37:38,200 –> 00:37:43,540 equation approach

这个方法与投影方程有关

670 00:37:43,540 –> 00:37:45,700 and and you can and it works as follows given a policy you

671 00:37:45,700 –> 00:37:48,910 solve this problem to find our new then

672 00:37:48,910 –> 00:37:52,600 you improve the policy

这个算法是这样工作的，给定一个策略，逆求解这个最小二乘问题找到 r，然后对策略进行改进

673 00:37:52,600 –> 00:37:53,980 and that allows you to solve this problem for the new

674 00:37:53,980 –> 00:37:59,410 policy and so on

然后用这个最小二乘问题计算新的策略

675 00:37:59,410 –> 00:38:02,680 now let's look at this for deterministic problems okay

我们来看看它是怎么作用在确定性问题上的

676 00:38:02,680 –> 00:38:04,840 this approach has found more application

677 00:38:04,840 –> 00:38:06,160 for deterministic problems rather than

678 00:38:06,160 –> 00:38:07,600 for stochastic problems and there are reasons for that

这个方法被发现相对于随机问题，更适用于确定性问题，原因如下

679 00:38:07,600 –> 00:38:09,430 let's look at the

680 00:38:09,430 –> 00:38:11,140 special case of deterministic problem

我们来看这个确定性问题的特殊例子

681 00:38:11,140 –> 00:38:13,960 now in the deterministic problem given a

682 00:38:13,960 –> 00:38:16,960 state in the control pair the next state

683 00:38:16,960 –> 00:38:19,900 is completely determined okay

对于一个确定性问题的状态控制对，下一个状态是完全确定的

684 00:38:19,900 –> 00:38:21,460 there are no transition probabilities there is a

685 00:38:21,460 –> 00:38:25,450 unique state which you go

没有转移概率，你将要得到的是一个唯一的状态

686 00:38:25,450 –> 00:38:27,440 so here's how this approach would be implemented
这就是这个方法是如何被使用的
687 00:38:27,440 –> 00:38:31,500 given a policy will generate a large set
688 00:38:31,500 –> 00:38:34,790 of sample pairs i K and UK
689 00:38:34,790 –> 00:38:38,120 corresponding deterministic costs
给定一个策略，产生很多状态控制对样本和相应的成本
690 00:38:38,120 –> 00:38:40,350 and corresponding transitions which are also
691 00:38:40,350 –> 00:38:46,290 deterministic to the state determines
692 00:38:46,290 –> 00:38:50,070 the state jk jk is the next state in
693 00:38:50,070 –> 00:38:53,540 news of JK is determined from the policy
状态转移会根据策略确定地到达状态 j_k
694 00:38:53,540 –> 00:38:55,980 now you can use a simulator of the
695 00:38:55,980 –> 00:38:59,310 deterministic system to generate such sample pairs
现在你可以使用这个确定性系统仿真器来生成这样的样本
696 00:38:59,310 –> 00:39:03,210 give an i and u the
697 00:39:03,210 –> 00:39:05,940 next u tell the simulator to generate JK for mu
给定一个 i 和 μ，可以生成控制 $\mu(j_k)$
698 00:39:05,940 –> 00:39:08,610 and you have a policy that
699 00:39:08,610 –> 00:39:12,720 gives you the control at the next state
700 00:39:12,720 –> 00:39:16,290 which you need for me you need to plug in here
然后这个策略就可以给你下一个状态的控制
701 00:39:16,290 –> 00:39:21,660 so given all these samples you
702 00:39:21,660 –> 00:39:24,030 put them together into a least squares
703 00:39:24,030 –> 00:39:29,010 problem that emulates this
给定了这些样本，把这些样本放到最小二乘问题中并计算
704 00:39:29,010 –> 00:39:33,030 so it's the sum of the errors in satisfying bellman
705 00:39:33,030 –> 00:39:34,350 equation right
这个是 bellman 误差的和
706 00:39:34,350 –> 00:39:37,650 it's a deterministic bellman equation
这是一个确定性 bellman 方程
707 00:39:37,650 –> 00:39:40,020 so this is the error corresponding this
708 00:39:40,020 –> 00:39:43,140 error here but instead of taking over
709 00:39:43,140 –> 00:39:45,450 all states you take it only over those samples
这是相关的 bellman 误差，与原来的对所有状态计算不同，这里对所有的样本进行计算
710 00:39:45,450 –> 00:39:48,900 if this summation includes all
711 00:39:48,900 –> 00:39:50,310 states these two would be a dime equivalent
如果这个表达式是对所有状态进行的，那么这两个表达式就是等价的
712 00:39:50,310 –> 00:39:54,450 so instead of sampling all
713 00:39:54,450 –> 00:39:56,370 the states I sample some representative
714 00:39:56,370 –> 00:39:59,340 states i form the error to satisfy
715 00:39:59,340 –> 00:40:00,420 balanced equation
716 00:40:00,420 –> 00:40:04,590 minimize that to obtain the weight
717 00:40:04,590 –> 00:40:08,180 vector that corresponds to the policy
用采集有代表性的样本取代对所有状态做计算，我这样处理误差并且最小化来得到相关策略的权重向量
718 00:40:08,180 –> 00:40:10,920 it's a linear least squares problem it
719 00:40:10,920 –> 00:40:13,590 can be solved the solution of this can
720 00:40:13,590 –> 00:40:17,130 be done in many different ways
这是一个线性最小二乘问题，可以被很多方法求解
721 00:40:17,130 –> 00:40:20,190 you need to have enough samples in order for the
722 00:40:20,190 –> 00:40:22,980 matrix involved here to be invertible to
723 00:40:22,980 –> 00:40:25,140 have a unique solution but this can be
724 00:40:25,140 –> 00:40:26,190 easily arranged
725 00:40:26,190 –> 00:40:31,910 it's not not a problem okay
你需要有足够的样本来保证矩阵可逆，然后得到一个唯一解，这很容易做到
726 00:40:31,910 –> 00:40:35,520 let me mention also that for stochastic
727 00:40:35,520 –> 00:40:37,500 problems is a similar approach but it's more complicated
随机问题和这个相似但是更复杂

728 00:40:37,500 –> 00:40:39,750 it involves taking Q
729 00:40:39,750 –> 00:40:42,090 samples per iteration it's it's more and
730 00:40:42,090 –> 00:40:46,440 it's it's it's more complicated and and
731 00:40:46,440 –> 00:40:48,330 there are reasons for that
732 00:40:48,330 –> 00:40:53,580 refer you to the text
每一个阶段都要计算 Q 值，这就是复杂的原因
733 00:40:53,580 –> 00:40:57,090 however the approach is model free and for the
734 00:40:57,090 –> 00:40:58,650 deterministic problems it is very
735 00:40:58,650 –> 00:41:01,260 attacked attractive for adaptive control
736 00:41:01,260 –> 00:41:05,070 of systems with unknown dynamics
确定性问题的方法不需要知道模型，同样非常适合于不知道动态性的自适应系统
737 00:41:05,070 –> 00:41:06,920 for example you may have a linear system
738 00:41:06,920 –> 00:41:10,430 like a process or some kind of a robot
比如你有一个线性系统，比如一个过程或者某种类型的机器人
739 00:41:10,430 –> 00:41:13,260 that does things but because of the
740 00:41:13,260 –> 00:41:16,290 circumstances which operates you don't
741 00:41:16,290 –> 00:41:18,150 know the dynamics of the system may be
742 00:41:18,150 –> 00:41:20,370 linear but you don't know them the
743 00:41:20,370 –> 00:41:22,350 coefficients the coefficients that
744 00:41:22,350 –> 00:41:25,110 multiply the state coefficient that
745 00:41:25,110 –> 00:41:29,760 multiplies the control and so on
由于系统的参数你不知道系统的动态性你也不知道，状态和控制的系数都不知道，只知道这是一个线性系统
746 00:41:29,760 –> 00:41:33,360 this approach provides an avenue for dealing
747 00:41:33,360 –> 00:41:35,970 with such problems without knowing the
748 00:41:35,970 –> 00:41:38,970 dynamics you go directly towards finding
749 00:41:38,970 –> 00:41:41,550 an optimal policy what more Q factors optimal policy
这个方法提供了一个不知道系统细节的情况下直接计算最优策略的途径

4 ADAPTIVE CONTROL BASED ON ADP

750 00:41:41,550 –> 00:41:47,010 so that's the next topic
751 00:41:47,010 –> 00:41:48,690 I want to talk about but we'll take a
752 00:41:48,690 –> 00:41:52,680 break this relates to something called
753 00:41:52,680 –> 00:41:55,500 adaptive dynamic programming which is
754 00:41:55,500 –> 00:41:57,420 really adaptive control based on
755 00:41:57,420 –> 00:42:00,090 approximate dynamic programming adaptive
756 00:42:00,090 –> 00:42:02,670 dynamic programming a DP has the same
757 00:42:02,670 –> 00:42:03,930 acronym as approximate dynamic
758 00:42:03,930 –> 00:42:08,790 programming which is a DP also but
759 00:42:08,790 –> 00:42:10,890 adaptive dynamic programming is really a
760 00:42:10,890 –> 00:42:14,070 subset of approximate dynamic program so
761 00:42:14,070 –> 00:42:15,900 we'll talk about that after we take a
762 00:42:15,900 –> 00:42:20,330 break any questions
这是我要讲的下一个话题，但是我要先休息一会，然后讲自适应动态规划。一种基于近似动态规划的自适应控制。自适应动态规划与近似动态规划相同，或者说自适应动态规划是近似动态规划的一个子集。休息之后再讲吧，谁有什么问题么
(questions
问题：连续状态空间是否有精确 Q 学习算法
回答：没有
763 00:42:31,240 –> 00:42:34,320 [Music]
764 00:42:35,020 –> 00:42:38,440 okay but that's a good question
765 00:42:38,440 –> 00:42:41,260 we have been talking so far about
766 00:42:41,260 –> 00:42:44,780 discrete space systems finite state
767 00:42:44,780 –> 00:42:48,320 systems however this approach also has
768 00:42:48,320 –> 00:42:50,570 an analogue for continuous space systems
769 00:42:50,570 –> 00:42:52,700 and in fact it is continuous space
770 00:42:52,700 –> 00:42:55,520 systems that involve for which adaptive

771 00:42:55,520 –> 00:42:58,820 control is most applicable basically you
772 00:42:58,820 –> 00:43:00,920 do the same thing for continuous space
773 00:43:00,920 –> 00:43:16,850 systems you just sample States okay we
774 00:43:16,850 –> 00:43:19,880 have to distinguish between okay your
775 00:43:19,880 –> 00:43:21,740 question is the following suppose I have
776 00:43:21,740 –> 00:43:23,690 a continuous space problem to begin with
777 00:43:23,690 –> 00:43:26,240 does there exist an exact Q learning
778 00:43:26,240 –> 00:43:28,160 algorithm that operates on the
779 00:43:28,160 –> 00:43:30,710 continuous space of states the answer is
780 00:43:30,710 –> 00:43:34,880 no we don't I don't play the literature
781 00:43:34,880 –> 00:43:38,270 has there's no non version of the
782 00:43:38,270 –> 00:43:39,800 algorithm probably it's much more
783 00:43:39,800 –> 00:43:43,040 complicated perhaps you can you can
784 00:43:43,040 –> 00:43:44,960 figure out how it might be but there's
785 00:43:44,960 –> 00:43:47,570 no analysis of such an algorithm now
786 00:43:47,570 –> 00:43:49,790 that's one thing Q learning per
787 00:43:49,790 –> 00:43:51,950 continuous space system in exact form
788 00:43:51,950 –> 00:43:54,440 that we don't know how it how it
789 00:43:54,440 –> 00:43:57,190 operates but it is possible to apply
790 00:43:57,190 –> 00:44:00,770 your learning for a continuous space
791 00:44:00,770 –> 00:44:03,140 system with cost function approximation
792 00:44:03,140 –> 00:44:06,470 and the R vector then the vector of
793 00:44:06,470 –> 00:44:08,360 weights the basis functions is only a
794 00:44:08,360 –> 00:44:11,300 finite number and for that theory so
795 00:44:11,300 –> 00:44:14,500 there has been a lot of work

问题：如何保证这个算法的性能

回答：没法保证，收敛和收敛率 (convergence rate) 都没法保证，最后一段在说成本函数 g 有解析表达的时候会不会有帮助，没听清，但是再说解析表达会有帮助的

796 00:44:21,630 –> 00:44:24,519 okay so the question is how fast
797 00:44:24,519 –> 00:44:26,079 what's the put what are the performance
798 00:44:26,079 –> 00:44:34,950 guarantees for good for disabilities III
799 00:44:34,950 –> 00:44:36,190 okay
800 00:44:36,190 –> 00:44:37,539 the Q learning algorithm involves
801 00:44:37,539 –> 00:44:41,740 several complicating factors one is the
802 00:44:41,740 –> 00:44:43,630 resemblance to stochastic approximation
803 00:44:43,630 –> 00:44:46,569 in the use of samples and the other is
804 00:44:46,569 –> 00:44:50,380 one Q factor at a time if we were to do
805 00:44:50,380 –> 00:44:52,869 all two factors simultaneously and
806 00:44:52,869 –> 00:44:54,579 obtain basically a stochastic
807 00:44:54,579 –> 00:44:58,200 approximation algorithm then there are
808 00:44:58,200 –> 00:45:00,519 convergence rate results about
809 00:45:00,519 –> 00:45:02,529 stochastic approximation algorithms
810 00:45:02,529 –> 00:45:04,630 however the Q learning algorithm more
811 00:45:04,630 –> 00:45:07,089 complicated because it involves a
812 00:45:07,089 –> 00:45:14,230 chaotic kind of chaotic trajectories
813 00:45:14,230 –> 00:45:16,390 over state control pairs and you update
814 00:45:16,390 –> 00:45:18,640 only one at a time and I don't believe
815 00:45:18,640 –> 00:45:21,910 that there are interesting convergence
816 00:45:21,910 –> 00:45:25,829 rate results for this type of algorithm
817 00:45:25,829 –> 00:45:29,079 if you want a qualitative answer the
818 00:45:29,079 –> 00:45:31,930 algorithm is very slow very slow very
819 00:45:31,930 –> 00:45:36,190 often and that's typical of stochastic
820 00:45:36,190 –> 00:45:38,829 approximation algorithms so if you are
821 00:45:38,829 –> 00:45:43,359 interested in high accuracy then that's
822 00:45:43,359 –> 00:45:44,890 probably something that you cannot yet
823 00:45:44,890 –> 00:45:46,809 but sometimes you get convergence
824 00:45:46,809 –> 00:45:49,240 reasonably fast particularly if you've
825 00:45:49,240 –> 00:45:51,339 got no good starting points if you have

826 00:45:51,339 –> 00:45:54,430 if you have if you do the sampling in an
827 00:45:54,430 –> 00:46:01,119 intelligent way if you've your first
828 00:46:01,119 –> 00:46:03,190 experiences with Q learning are likely
829 00:46:03,190 –> 00:46:07,470 to be frustrating frustratingly slow
830 00:46:07,870 –> 00:46:24,110 form of it it if G were continued okay
831 00:46:24,110 –> 00:46:27,080 so so what kind of G function here would
832 00:46:27,080 –> 00:47:16,790 help perhaps yeah would be also if you
833 00:47:16,790 –> 00:47:19,190 use cost function approximation if you
834 00:47:19,190 –> 00:47:21,020 know the form of G you may be able to
835 00:47:21,020 –> 00:47:23,360 exploit it in order to pick your basis
836 00:47:23,360 –> 00:47:25,850 functions appropriately and that problem
837 00:47:25,850 –> 00:47:28,540 is going to be helpful
838 00:47:30,670 –> 00:47:34,360 any other questions
839 00:47:36,400 –> 00:47:38,319 okay so let's take a break for ten
840 00:47:38,319 –> 00:00:00,000 minutes come back