1 ADAPTIVE CONTROL BASED ON ADP

1 00:00:37,079 –> 00:00:43,110 so okay so as we want to go through the
2 00:00:43,110 –> 00:00:45,170 subject of adaptive control control of
3 00:00:45,170 –> 00:00:48,180 dynamic systems with unknown dynamics
4 00:00:48,180 –> 00:00:49,860 and used Q learning for that purpose
5 00:00:49,860 –> 00:00:54,680 let me review this slide
讲自适应动态规划之前先复习一下这页的内容
6 00:00:54,680 –> 00:00:59,010 we want to do approximate policy iteration for Q
7 00:00:59,010 –> 00:01:02,760 factors where policy evaluation is done
8 00:01:02,760 –> 00:01:05,670 by solving the by minimizing the squared
9 00:01:05,670 –> 00:01:09,080 error in satisfying the bellman equation
我们想要做 Q 值得近似策略迭代，策略评价是通过最小化 bellman 二次误差完成的
10 00:01:09,080 –> 00:01:12,539 and so this is the mapping associated
11 00:01:12,539 –> 00:01:16,020 with mu and use a Euclidean norm circle to some distribution
这个是与策略 μ 相关的映射，使用某种分布求欧几里得范数
12 00:01:16,020 –> 00:01:18,600 and now we focus on
13 00:01:18,600 –> 00:01:20,729 deterministic systems while we consider
14 00:01:20,729 –> 00:01:23,880 samples of state and control because the
15 00:01:23,880 –> 00:01:25,560 system is deterministic the next state
16 00:01:25,560 –> 00:01:28,440 is completely determined and then the
17 00:01:28,440 –> 00:01:32,940 transition is to a Q factor involving
18 00:01:32,940 –> 00:01:37,800 the people the the current policy
现在我们关注确定性系统，考虑的下一个状态和控制样本是完全确定的，而且这个状态转移是依赖于当前策略的
19 00:01:37,800 –> 00:01:40,170 and we set up this least squares problem in
20 00:01:40,170 –> 00:01:43,590 this form in fact this would be exactly
21 00:01:43,590 –> 00:01:46,649 the same squares problem if we were to
22 00:01:46,649 –> 00:01:50,369 use many many samples of state control
23 00:01:50,369 –> 00:01:52,229 pairs but in practice we would use
24 00:01:52,229 –> 00:01:54,149 perhaps a limited set of state control pairs
我们把这个最小二乘问题写成这个形式 (下面的公式)，事实上如果我们使用非常多的样本进行计算这是一个与最小二乘问题相同的精确算法，但是在实际使用中我们会使用有限的样本集合
25 00:01:54,149 –> 00:01:58,110 solve this get r selenium
26 00:01:58,110 –> 00:02:00,899 squares problem and then that's the
27 00:02:00,899 –> 00:02:04,410 that's r mu for the for the for this
28 00:02:04,410 –> 00:02:06,629 policy and then we can use policy
29 00:02:06,629 –> 00:02:10,470 improvement to get to define the new
30 00:02:10,470 –> 00:02:12,030 policy which is going to be used to
31 00:02:12,030 –> 00:02:14,930 drive the next cycle of policy direction
求解这个表达式得到这个策略的 r，然后可以进行策略改进来得到新的策略，这个新策略会驱动新一次迭代的进行

1.1 LINEAR-QUADRATIC PROBLEM

32 00:02:14,930 –> 00:02:17,760 okay so now we're going to look into
33 00:02:17,760 –> 00:02:20,340 this in the context of continuous space
34 00:02:20,340 –> 00:02:22,890 and control systems and we're going to
35 00:02:22,890 –> 00:02:28,069 consider a classical example of control
36 00:02:28,129 –> 00:02:33,870 involving a linear system
现在我们来看一个连续状态和控制的系统，这是一个典型的线性系统的例子
37 00:02:33,870 –> 00:02:38,069 XK plus 1 equals a XJ plus bu K XK is a state vector
$x_{k+1} = Ax_k + Bu_k$，x_k 是一个状态向量
38 00:02:38,069 –> 00:02:40,500 the state vector is a vector in RN okay
状态向量是一个 $\Re^n$ 空间内的向量
39 00:02:40,500 –> 00:02:43,959 so it's a vector of n
40 00:02:43,959 –> 00:02:47,439 continues state variables okay has n components
这个向量由 n 个连续变量组成
41 00:02:47,439 –> 00:02:52,299 and UK is a vector of control

42 00:02:52,299 –> 00:02:55,750 components of M control components okay

u_k 是 $\Re^m$ 空间中的控制向量，有 m 个元素

43 00:02:55,750 –> 00:03:00,040 a is an N by n matrix B is an N by M matrix

A 是一个 $n \times n$ 矩阵，B 是一个 $n \times m$ 矩阵

44 00:03:00,040 –> 00:03:04,389 and this is the standard

45 00:03:04,389 –> 00:03:06,700 discrete-time linear system time

46 00:03:06,700 –> 00:03:08,889 invariant because a and B do not change over time

这是一个标准的时间无关的离散时间线性系统，因为 A 和 B 不随着时间变化

47 00:03:08,889 –> 00:03:12,250 and a classical problem is to

48 00:03:12,250 –> 00:03:15,159 find the control sequence from 0 to

49 00:03:15,159 –> 00:03:18,299 infinity that minimizes a quadratic cost

这个经典问题想要找到从 0 时刻到无穷时刻的控制序列来最小化二次成本

50 00:03:18,299 –> 00:03:23,260 ok now Q is an N by n positive semi-definite matrix

Q 是一个 $n \times n$ 半定矩阵

51 00:03:23,260 –> 00:03:27,099 X prime Q X is the

52 00:03:27,099 –> 00:03:30,669 quadratic form associated with Q

$x\prime Qx$ 是一个关于 Q 的二次项

53 00:03:30,669 –> 00:03:34,599 X prime is a row vector Q a column vector such a

54 00:03:34,599 –> 00:03:37,599 scalar quadratic form that penalizes

55 00:03:37,599 –> 00:03:42,579 large values of Xk

$x\prime$ 是一个行向量，Q 是一个列向量，标量二次形式是这样的，这一项表示关于 x_k 的惩罚

56 00:03:42,579 –> 00:03:46,120 similarly here we have a quadratic cost on control R is a

57 00:03:46,120 –> 00:03:49,510 positive definite matrix

相似地，有一个关于控制的二次项成本，R 是一个正定矩阵

58 00:03:49,510 –> 00:03:54,040 and it penalizes large values of control

这是一个关于控制的大惩罚项

59 00:03:54,040 –> 00:03:55,690 so basically we want to drive the state

60 00:03:55,690 –> 00:04:00,310 towards zero with small with relatively

61 00:04:00,310 –> 00:04:02,650 small amounts of control

所以比较基本的，我想要在比较小的控制下让状态趋于 0

62 00:04:02,650 –> 00:04:04,680 and we want to do it gradually over an infinite horizon

我想要在无限期内完成这件事

63 00:04:04,680 –> 00:04:07,209 it's a classical formulation and that

64 00:04:07,209 –> 00:04:10,030 meets a very elegant solution the

65 00:04:10,030 –> 00:04:13,150 optimal policy is linear

这是一个经典的公式，结果很优雅，是一个线性最优策略

66 00:04:13,150 –> 00:04:15,720 so the optimal policy is some matrix multiplying X

67 00:04:15,720 –> 00:04:19,899 that's the optimal gain matrix of the problem

最优策略是某个矩阵乘以 x，也就是这个问题得到的最优矩阵

68 00:04:19,899 –> 00:04:22,509 and if we can find that we

69 00:04:22,509 –> 00:04:24,280 measure the state then we multiply it

70 00:04:24,280 –> 00:04:26,320 with L and get to control the linear

71 00:04:26,320 –> 00:04:29,229 feedback control scheme and it works very nicely

如果我们得到了这个最优矩阵，就可以在某状态下用它诚意这个状态得到控制，这种线性反馈控制方案的很好

72 00:04:29,229 –> 00:04:36,130 ok now how about Q factors

Q 值是怎么起作用的呢

73 00:04:36,130 –> 00:04:39,909 well actually does the given any linear

74 00:04:39,909 –> 00:04:42,190 part suppose you have a linear policy it

75 00:04:42,190 –> 00:04:46,750 turns out that the Q factor of a state

76 00:04:46,750 –> 00:04:50,710 control pair is a quadratic involving

77 00:04:50,710 –> 00:04:53,260 some matrix chain mu the same matrix

78 00:04:53,260 –> 00:04:56,349 for all state control pairs

给定一个线性策略，事实证明状态控制对的 Q 值是一个带有某个矩阵的二次项，而且这个矩阵与状态和控制无关

79 00:04:56,349 –> 00:04:57,540 okay the optimal cost is actually

80 00:04:57,540 –> 00:05:00,030 quadratic function of X

最优成本是 x 的二次函数

81 00:05:00,030 –> 00:05:02,460 the optimal Q factor were more generally the Q factor
82 00:05:02,460 –> 00:05:06,830 of any linear policy is a quadratic
83 00:05:06,830 –> 00:05:12,510 involving some matrix here K mu that can
84 00:05:12,510 –> 00:05:16,950 be calculated if we knew a and B
最优 Q 值得方案更一般性，对于任何线性策略，Q 都是一个带有矩阵 K_μ 得二次项，如果你知道 A 和 B，这个 K_μ 是可以计算出来的
85 00:05:16,950 –> 00:05:18,960 however we want to look at the case where a and B are not known
我们想要研究的是 A 和 B 不知道的情况下该怎么办
86 00:05:18,960 –> 00:05:21,480 and when you want to
87 00:05:21,480 –> 00:05:24,060 evaluate policies by finding this K
88 00:05:24,060 –> 00:05:27,450 mu using a simulator of the system
89 00:05:27,450 –> 00:05:31,500 as opposed to using a and B
当你想要通过仿真找到这个 K_μ 而不是使用 A 和 B 对策略进行评价
90 00:05:31,500 –> 00:05:38,010 and for this we will use Q learning or Q factor policy iteration
我们要使用 Q 学习或者 Q 策略迭代来完成
91 00:05:38,010 –> 00:05:42,660 approximate policy duration
92 00:05:42,660 –> 00:05:45,390 with a Q factors represented as a linear
93 00:05:45,390 –> 00:05:50,250 combination of basis functions
带有 Q 值得近似策略迭代使用基函数表示为线性函数
94 00:05:50,250 –> 00:05:52,470 but here we are very fortunate because we know
95 00:05:52,470 –> 00:05:57,000 ahead of time that the Q factors of a
96 00:05:57,000 –> 00:06:00,900 policy are quadratic
但是幸运的是，我们知道 Q 得策略是二次的
97 00:06:00,900 –> 00:06:04,140 so the thing that's unknown here is this K nu but the basis
98 00:06:04,140 –> 00:06:06,510 functions are the quadratic functions
99 00:06:06,510 –> 00:06:08,400 the right but the right basis functions are quadratic functions
所以我们不知道的是 K_μ，但是这个函数是二次型我们是知道的
100 00:06:08,400 –> 00:06:10,770 in this K mu
101 00:06:10,770 –> 00:06:13,770 can be viewed as a vector of ways that
102 00:06:13,770 –> 00:06:18,330 ways the quadratic basis functions
在这个 K_μ 中，我们可以把它看作一个向量，也就是二次基函数
103 00:06:18,330 –> 00:06:20,220 so we're going to use as basis functions of these things here
我们可以使用这样的基函数
104 00:06:20,220 –> 00:06:25,470 all possible products
105 00:06:25,470 –> 00:06:28,170 of state components with other state
106 00:06:28,170 –> 00:06:32,520 components so X 1 squared X 1 2 X 1 3
107 00:06:32,520 –> 00:06:36,180 and so 1 X 2 squared and so on
所有状态之间的乘积都可能出现，x_1 得平方，x_2，x_3 什么的平方，x_1 乘以 x_2 之类的
108 00:06:36,180 –> 00:06:38,130 similarly we're going to use all quadratic
109 00:06:38,130 –> 00:06:41,940 components involving control u 1 square
110 00:06:41,940 –> 00:06:44,340 u 2 square and so one you want you to
111 00:06:44,340 –> 00:06:46,550 all the possible cross terms
相似地我们可以使用所有二次项控制，u_1 的平方，u_2 的平方之类的，所有这样的项都可以出现
112 00:06:46,550 –> 00:06:48,810 and similarly all the possible cross terms
113 00:06:48,810 –> 00:06:54,540 between state and control 。
还有所有控制和状态的乘积
114 00:06:54,540 –> 00:06:58,650 so what is the the the matrix phi here it is for a
115 00:06:58,650 –> 00:07:01,890 given X and u the row of phi of cost
116 00:07:01,890 –> 00:07:05,250 phi has an infinite number of component
117 00:07:05,250 –> 00:07:05,860 row Rose okay
所以这个矩阵 ϕ，对于给定的 x 和 u，这是一个有无数个元素的行向量
118 00:07:05,860 –> 00:07:08,949 but it has a finite number of
119 00:07:08,949 –> 00:07:11,620 columns which correspond to these basis
120 00:07:11,620 –> 00:07:15,400 functions and for a given X you the row
121 00:07:15,400 –> 00:07:18,039 of fee is part precisely of all these components
但是它的列是有限个的，而且与基函数相关，给定 x 和 u，ϕ 的行是这些元素组成的
122 00:07:18,039 –> 00:07:22,780 so if you can give me if I if

123 00:07:22,780 –> 00:07:24,669 you can hear me acts on you I can
124 00:07:24,669 –> 00:07:29,289 calculate the the corresponding row very easily
所以给定 x 和 u 之后，可以很轻松地计算行向量
125 00:07:29,289 –> 00:07:41,830 and the Q factor of a linear
126 00:07:41,830 –> 00:07:43,960 policy because we know that it has this
127 00:07:43,960 –> 00:07:47,139 form it can be exactly represented
128 00:07:47,139 –> 00:07:50,800 within the approximation subspace
由于我们知道了 Q 的策略是一个线性策略那么我们就可以精确地使用子空间近似来表示他
129 00:07:50,800 –> 00:07:53,050 okay we're very fortunate here because we
130 00:07:53,050 –> 00:07:55,449 know good basis functions ahead of time
这就是一个很幸运的事情，我们知道基函数的形式
131 00:07:55,449 –> 00:07:58,479 and if I can find the corresponding
132 00:07:58,479 –> 00:08:00,520 weight vector this is the same as
133 00:08:00,520 –> 00:08:03,610 finding this matrix chase of Nu
如果我能算出相关的权重向量，就可以得到矩阵 K_μ
134 00:08:03,610 –> 00:08:05,169 and Q learning is going to be used to find
135 00:08:05,169 –> 00:08:08,080 this way to weight vector without
136 00:08:08,080 –> 00:08:12,009 knowing and a and you just by use it you
137 00:08:12,009 –> 00:08:15,009 a and B just by using a simulator of a system
Q 学习被用来使用仿真来计算不知道 A 和 B 的时候的权重向量
138 00:08:15,009 –> 00:08:17,500 in other words instead of having
139 00:08:17,500 –> 00:08:20,349 area B I got the system and when I put
140 00:08:20,349 –> 00:08:22,300 in here to I measure the state you need
141 00:08:22,300 –> 00:08:24,669 to put in certain control it generates
142 00:08:24,669 –> 00:08:27,880 another state and that's all I know
143 00:08:27,880 –> 00:08:31,449 either all I need in order to apply Q learning
换句话说，取代知道 A 和 B 的情况，我知道当前状态给定一个控制后新状态是什么，就可以使用 Q 学习了

1.2 PI FOR LINEAR-QUADRATIC PROBLEM

144 00:08:31,449 –> 00:08:38,349 more precisely suppose I have a
145 00:08:38,349 –> 00:08:40,929 policy mu and I want to evaluate it and
146 00:08:40,929 –> 00:08:43,229 find the corresponding set of weights
假设我有一个策略 μ，我想要对他估值并找到相应的权重向量
147 00:08:43,229 –> 00:08:49,199 I set up the bellman error squared
我使用 bellman 误差平方
148 00:08:49,199 –> 00:08:53,550 I generate a set of state control pairs
149 00:08:53,550 –> 00:08:57,010 and write down the bellman equation
150 00:08:57,010 –> 00:09:00,490 error form this least squares problem
151 00:09:00,490 –> 00:09:02,910 which is linearly squares
我生成了一个状态控制对的集合，然后写出 bellman 方程误差的形式
152 00:09:02,910 –> 00:09:05,649 remember these are the rows of the matrix phi okay for a given X in U
给定 x 和 u，这些 $(\phi(x_k, u_k)')$ 是矩阵 Φ 的行
153 00:09:05,649 –> 00:09:09,040 and
154 00:09:09,040 –> 00:09:13,029 similarly here and R is a vector
155 00:09:13,029 –> 00:09:15,790 multiplying with growth
156 00:09:15,790 –> 00:09:17,740 a column that my prince Robert is
157 00:09:17,740 –> 00:09:20,170 awaiting before all this this basis functions
没听明白到底想说啥。。。
158 00:09:20,170 –> 00:09:22,990 this is the one stage cost so
这是一阶段成本
159 00:09:22,990 –> 00:09:24,460 this is a linear least squares problem
所以这是一个线性最小二乘问题
160 00:09:24,460 –> 00:09:29,020 and XJ U K XK plus one are many samples
161 00:09:29,020 –> 00:09:31,540 generated by a system or a simulator of the system
(x_k, u_k, x_{k+1}) 这类转移样本有很多，都是由系统或者系统的仿真生成的

162 00:09:31,540 –> 00:09:39,070 and after I find after I do
163 00:09:39,070 –> 00:09:41,800 the policy evaluation I can do policy
164 00:09:41,800 –> 00:09:44,080 improvement by doing this minimization
165 00:09:44,080 –> 00:09:47,370 at a given state X I minimize over u
我评估过策略之后，就可以通过给定状态 x 时在所有 u 中最小化这个表达式来进行策略改进
166 00:09:47,370 –> 00:09:50,650 but this minimization can be done in
167 00:09:50,650 –> 00:09:53,530 closed form because this FS here are
168 00:09:53,530 –> 00:09:58,780 what are this are quadratic in U so I
169 00:09:58,780 –> 00:10:04,270 can very simply do this calculation
但是这个最小化可以被这个形式的表达式完成，因为 $\phi(x,u)$ 关于 u 是二次的，所以我可以很轻松地进行计算
170 00:10:04,270 –> 00:10:08,890 so that exact policy duration for Q factors
171 00:10:08,890 –> 00:10:11,440 exact because I'm fortunate to know good
172 00:10:11,440 –> 00:10:13,960 basis functions and knowledge of alien B is not required
所以这是 Q 值的精确策略迭代，因为我知道比较好的基函数并且不需要知道矩阵 B
173 00:10:13,960 –> 00:10:19,660 it's quite remarkable
174 00:10:19,660 –> 00:10:21,070 actually because people have been
175 00:10:21,070 –> 00:10:22,750 working on adaptive control of linear
176 00:10:22,750 –> 00:10:24,970 systems since time immemorial since the 50s okay
这是一个很好的方法，因为人们关于线性系统自适应控制从 50 年代就开始了
177 00:10:24,970 –> 00:10:27,970 this approach is relatively new
178 00:10:27,970 –> 00:10:30,430 but actually discovered in the early
179 00:10:30,430 –> 00:10:33,160 days of approximate dynamic program in
180 00:10:33,160 –> 00:10:35,340 the early nineties but it has picked up
181 00:10:35,340 –> 00:10:39,130 it has been picked up by control in the
182 00:10:39,130 –> 00:10:47,230 control field quite strong recently okay
这个方法相对比较新，但是实际上早些年近似动态规划九十年代在控制领域就已经比较频繁地提出这个方法了
183 00:10:47,230 –> 00:10:49,150 so because all of this is essentially exact
因为这些信息都必须是精确的
184 00:10:49,150 –> 00:10:54,280 if you use a sufficient number of
185 00:10:54,280 –> 00:10:56,800 state control pairs here and in fact you
186 00:10:56,800 –> 00:10:58,450 need only a finite number in this
187 00:10:58,450 –> 00:11:00,580 particular case
如果你用足够数量的状态控制对，你只需要有限步迭代就可以让这个例子收敛了
188 00:11:00,580 –> 00:11:02,980 convergence to an optimal policy can be shown
而且可以看到收敛到最优解
189 00:11:02,980 –> 00:11:09,430 now the basic idea of this example
190 00:11:09,430 –> 00:11:11,230 has been carried further within the
191 00:11:11,230 –> 00:11:14,100 field of adaptive dynamic programming
这个例子的基本想法将来还会被自适应动态规划的研究者继续使用
192 00:11:14,100 –> 00:11:19,390 where you may have a nonlinear discrete
193 00:11:19,390 –> 00:11:21,700 time system or you may have a cost
194 00:11:21,700 –> 00:11:25,390 that's not quadratic and then what you
195 00:11:25,390 –> 00:11:27,390 need to do is basically approximate
196 00:11:27,390 –> 00:11:30,620 patience of these things here but still
197 00:11:30,620 –> 00:11:38,070 you may with with with model free policy
198 00:11:38,070 –> 00:11:42,300 duration ideas you can obtain viable
199 00:11:42,300 –> 00:11:46,100 schemes that people have been have been
200 00:11:46,100 –> 00:11:50,430 have been dealing with or experimenting
201 00:11:50,430 –> 00:12:12,270 with for quite a few years now
如果你有一个非线性离散时间系统或者一个成本布什尔茨的系统，你就会需要近似这些东西，用一个无模型策略迭代来求解，人们这些年就是这么做的
(asking quetsion
问题：为什么最小 bellman 二次误差没用 x_k，问题太简单，我就不记录了
202 00:12:12,270 –> 00:12:15,420 yes this is XJ plus one because that's this is
203 00:12:15,420 –> 00:12:17,820 the the cost approximation at the next
204 00:12:17,820 –> 00:12:20,910 state right according to the formula of

205 00:12:20,910 –> 00:12:26,580 Q learning so give an XK and UK you
206 00:12:26,580 –> 00:12:31,200 generate XK plus one from that okay and
207 00:12:31,200 –> 00:12:34,590 that's the yeah you need the XJ plus one
208 00:12:34,590 –> 00:12:38,580 here not XJ if you go back to the Q
209 00:12:38,580 –> 00:12:42,710 learning formulas let me just go back
210 00:12:44,900 –> 00:12:51,440 okay the next state comes in here and
211 00:13:07,450 –> 00:13:12,600 I'm sorry I can find it but
212 00:13:27,209 –> 00:13:30,459 yeah actually here okay it's the next
213 00:13:30,459 –> 00:14:06,940 state okay okay so you're looking the so
214 00:14:06,940 –> 00:14:09,839 I think your question is okay i i i i
215 00:14:09,839 –> 00:14:14,320 use this algorithm i calculate our new
216 00:14:14,320 –> 00:14:16,720 by solving this problem and that gives
217 00:14:16,720 –> 00:14:19,540 me the matrix chain new for the policy
218 00:14:19,540 –> 00:14:22,930 that policy and then but i generate a
219 00:14:22,930 –> 00:14:26,950 sequence of policies and the k news of
220 00:14:26,950 –> 00:14:28,959 the different policies will converge the
221 00:14:28,959 –> 00:14:31,600 k star let's say the k star
222 00:14:31,600 –> 00:14:35,520 corresponding to the optimal few factors
问题：算法会得到 r 序列和策略序列，这个序列会向何处收敛
回答：向最优矩阵收敛，也就是最优策略
223 00:14:51,050 –> 00:14:53,310 I'm not sure I understand complete your
224 00:14:53,310 –> 00:14:55,530 question you ask you the question of
225 00:14:55,530 –> 00:14:57,630 convergence of this algorithm this
226 00:14:57,630 –> 00:14:59,220 algorithm generates a sequence of
227 00:14:59,220 –> 00:15:00,990 policies and the sequence of weights
228 00:15:00,990 –> 00:15:04,140 weight vectors so where do these weight
229 00:15:04,140 –> 00:15:08,010 vectors converge to they will converge
230 00:15:08,010 –> 00:15:11,340 to the optimal weights that correspond
231 00:15:11,340 –> 00:15:20,760 to the optimal matrix can you not argue
232 00:15:20,760 –> 00:15:23,220 does not converge to you have our new
233 00:15:23,220 –> 00:15:27,660 zero R mu 1 mu 2 and so on and these are
234 00:15:27,660 –> 00:15:30,060 going to converge to our new star let's
235 00:15:30,060 –> 00:15:38,700 say okay the our new corresponds to a
236 00:15:38,700 –> 00:15:41,480 single policy
asking completed)
237 00:15:46,109 –> 00:15:48,369 incidentally I'm asserted here
238 00:15:48,369 –> 00:15:51,720 convergence but there is something that
239 00:15:51,720 –> 00:15:53,799 but-but-but we have discussed
240 00:15:53,799 –> 00:15:56,980 convergence of policy direction only for
241 00:15:56,980 –> 00:15:59,369 the case of a finite number of states
我要顺便说一些关于收敛的事情，之前我们提到收敛是关于有限状态集合的
242 00:15:59,369 –> 00:16:01,720 where you have convergence in a finite
243 00:16:01,720 –> 00:16:03,669 number of iterations right that's what
244 00:16:03,669 –> 00:16:05,759 the only thing that we have talked about
我们只讨论了有限状态数量的迭代的收敛性
245 00:16:05,759 –> 00:16:08,379 convergence of policy direction in an
246 00:16:08,379 –> 00:16:11,439 infinite space context is not a foregone
247 00:16:11,439 –> 00:16:12,850 conclusion okay
无穷空间中策略迭代的收敛性不是通用的结论
248 00:16:12,850 –> 00:16:16,480 it turns out however that for this
249 00:16:16,480 –> 00:16:18,970 linear quadratic problem and also for
250 00:16:18,970 –> 00:16:20,769 other related problems you get
251 00:16:20,769 –> 00:16:25,499 asymptotic convergence of the optimum of
252 00:16:25,499 –> 00:16:28,660 asymptotic convergence of the of the
253 00:16:28,660 –> 00:16:30,669 generated few factors to the optimal new
254 00:16:30,669 –> 00:16:32,980 factors and asymptotic convergence to an optimal policy
事实证明这个线性二次问题与其他问题关于 Q 值得渐进收敛性，是可以收敛到最优策略的
255 00:16:32,980 –> 00:16:35,649 but only because you have

256 00:16:35,649 –> 00:16:37,869 a lot of assumptions here it's a linear
257 00:16:37,869 –> 00:16:40,809 quadratic problem has involved a lot of structure
因为这个问题中你有很多假设，线性二次问题有很多结构可以保证
258 00:16:40,809 –> 00:16:45,279 so this again a result that
259 00:16:45,279 –> 00:16:47,439 goes back to the late 60s convergence of
260 00:16:47,439 –> 00:16:49,329 policy duration for the linear quadratic problem okay
线性二次问题策略迭代会收敛这个结论在六十年代就已经出现了

2 APPROXIMATION IN POLICY SPACE

261 00:16:49,329 –> 00:16:56,439 now if this is a vast
262 00:16:56,439 –> 00:17:01,269 subject and I we're not going to go into it
实际上这里有大量的主题，但是我不打算深入讲了
263 00:17:01,269 –> 00:17:05,939 let's talk about another major area
264 00:17:05,939 –> 00:17:08,470 but we also are not going to go in very
265 00:17:08,470 –> 00:17:10,869 deeply what just summarize we have been
266 00:17:10,869 –> 00:17:13,269 talking so far about approximation in
267 00:17:13,269 –> 00:17:16,329 value space or approximation in q-factor
268 00:17:16,329 –> 00:17:18,789 space now let's talk about approximation in policy space
我们来讨论一下另一个比较主要的领域，但是我不会讲的很深，只是一个总结，我们已经讲了很多关于值空间近似或者 Q 空间近似得问题了，现在我来讲一讲策略空间近似的内容

2.1 APPROXIMATION IN POLICY SPACE

269 00:17:18,789 –> 00:17:21,279 where instead of
270 00:17:21,279 –> 00:17:25,179 parameterizing costs or Q factors we parameterize policies
在这里我们要用参数化策略来代替参数化成本或者 Q 值
271 00:17:25,179 –> 00:17:28,559 so we parameterize
272 00:17:28,559 –> 00:17:33,519 policies by a vector R of weights and
273 00:17:33,519 –> 00:17:36,570 the corresponding weights or parameters
我们使用一个权重向量 r 来参数化策略
274 00:17:36,570 –> 00:17:39,519 this is an approximation architecture for policies
这是一个策略的近似结构
275 00:17:39,519 –> 00:17:44,049 so the policy that
276 00:17:44,049 –> 00:17:47,230 corresponds to a weight vector R gives
277 00:17:47,230 –> 00:17:50,740 you controls to apply at every possible
278 00:17:50,740 –> 00:17:55,389 state which depend on R
这个参数化的策略能够在所有可能出现的状态出现时根据权重向量 r 得到相应的控制
279 00:17:55,389 –> 00:17:57,559 now for a given R the policy is defined
280 00:17:57,559 –> 00:17:59,600 and therefore the corresponding cost
281 00:17:59,600 –> 00:18:03,220 vector is defined as a function of our
给定一个 r，策略就被定义好了，这样相应的成本向量也被定义成一个 r 的函数了
282 00:18:03,220 –> 00:18:05,059 so r
283 00:18:05,059 –> 00:18:07,580 give me policies and I want to find good
284 00:18:07,580 –> 00:18:10,039 r that give me good policies that's the idea
r 给我们一个策略，我想要做的就是找到一个好的 r，即好的策略，这就是这种方法的思路
285 00:18:10,039 –> 00:18:14,620 and to do this we may wish to
286 00:18:14,620 –> 00:18:17,600 optimize some measure of this cost
287 00:18:17,600 –> 00:18:22,250 corresponding to R over R
为了达到这个目标，我想要在所 r 中找能最小化成本的那一个
288 00:18:22,250 –> 00:18:24,830 for example we may formulate a cost function that
289 00:18:24,830 –> 00:18:29,240 involves the weighted sum of costs
290 00:18:29,240 –> 00:18:32,049 corresponding to the policy that
291 00:18:32,049 –> 00:18:36,080 corresponds to R and xi are some state
292 00:18:36,080 –> 00:18:38,900 dependent weights this is a scalar cost
293 00:18:38,900 –> 00:18:42,740 function that depends on R
比如我把优化的成本函数写成这样，包括某策略成本的加权累加，这个成本与 r 和 ξ 相关，ξ 是依赖于状态的权重，这是一个依赖于 r 的标量成本函数

294 00:18:42,740 –> 00:18:45,890 minimize this with respect to R gives you an optimal
295 00:18:45,890 –> 00:18:48,799 parameterization of policy in an optimal
296 00:18:48,799 –> 00:18:52,039 policy within this class
297 00:18:52,039 –> 00:18:54,140 optimal with respect to this cost function
最小化这个关于 r 的函数能够得到一个策略的最优参数，也就是这个问题的最优策略
298 00:18:54,140 –> 00:18:55,520 there is an issue here how do you weigh the
299 00:18:55,520 –> 00:18:57,679 various states and so on but let's bypass this ratio
这个话题是如何确定不同状态的权重，我们需要对这个比例进行调整
300 00:18:57,679 –> 00:19:03,080 you may use any of a
301 00:19:03,080 –> 00:19:07,460 large number of at least in principle
302 00:19:07,460 –> 00:19:10,460 you may use any of a large number of the
303 00:19:10,460 –> 00:19:12,679 term of optimization algorithms
304 00:19:12,679 –> 00:19:14,600 iterative optimization algorithms for minimizing subjecting
这种问题至少原则上你可以使用任意规模的项进行迭代来最小化目标函数
305 00:19:14,600 –> 00:19:16,880 for example a
306 00:19:16,880 –> 00:19:19,100 random search method for a gradient
307 00:19:19,100 –> 00:19:21,520 method or some other kind of method
比如一个梯度算法或者其他类型的算法的随机搜索
308 00:19:21,520 –> 00:19:25,549 that's the basic idea
这就是策略空间近似的基本想法
309 00:19:25,549 –> 00:19:27,049 now there's the question of how do you parameterize policies
现在的问题就是你该如何参数化策略
310 00:19:27,049 –> 00:19:31,340 generally the parameterization
311 00:19:31,340 –> 00:19:34,070 is problem dependent
一般参数化是依赖于问题的
312 00:19:34,070 –> 00:19:36,350 you look at your problem you know what is a good special
313 00:19:36,350 –> 00:19:42,919 structure you look at you you know that
314 00:19:42,919 –> 00:19:47,000 for simpler you have some idea about the
315 00:19:47,000 –> 00:19:49,610 structure of optimal policies you try to
316 00:19:49,610 –> 00:19:52,250 match the approximation architecture to that structure okay
你研究你的问题，然后知道这个问题最优策略的结构大概是什么样的，然后让近似结构匹配最优策略的结构
317 00:19:52,250 –> 00:19:55,789 but here's one
318 00:19:55,789 –> 00:19:59,480 general way to use state features to parameterize policies
但是这里有一个通用的方法，使用状态特征参数化策略
319 00:19:59,480 –> 00:20:03,230 introduce a cost
320 00:20:03,230 –> 00:20:06,320 approximation architecture we shall call
321 00:20:06,320 –> 00:20:07,240 V here
介绍了一种成本近似结构 V
322 00:20:07,240 –> 00:20:12,870 okay depending on a parameter vector R
323 00:20:12,870 –> 00:20:16,300 which and define indirectly a
324 00:20:16,300 –> 00:20:19,570 parameterization of policies by means of this minimization
以来这个参数向量 r，通过这个最小化表达式间接地定义参数化的策略
325 00:20:19,570 –> 00:20:23,350 so if you know good
326 00:20:23,350 –> 00:20:24,970 features you can still parameterize
327 00:20:24,970 –> 00:20:28,360 these policies this way
所以如果你知道比较好的特征，你依然可以使用这种方法参数化策略
328 00:20:28,360 –> 00:20:29,950 however it's also possible to use different
329 00:20:29,950 –> 00:20:32,350 parameterizations for state and for policies
然而同样可以使用不同的参数化来近似状态和策略
330 00:20:32,350 –> 00:20:35,559 in the parlance of this field a
331 00:20:35,559 –> 00:20:37,840 policy approximate is called an actor
332 00:20:37,840 –> 00:20:38,559 okay
333 00:20:38,559 –> 00:20:41,620 and the cost approximation is called a critic
这个领域的术语，把策略近似叫做行动，把成本近似叫做评价
334 00:20:41,620 –> 00:20:45,040 and many times and often these
335 00:20:45,040 –> 00:20:46,570 type of systems and involve our
336 00:20:46,570 –> 00:20:49,300 imaginations of one or both are called

337 00:20:49,300 –> 00:20:52,090 act or critic systems or critic systems

338 00:20:52,090 –> 00:20:54,940 or actor system actor only systems and so on

通常情况下，系统会包括一个或者两个同时包括，比如动作评价系统，评价系统或者动作系统之类的

2.2 APPROXIMATION IN POLICY SPACE METHODS

339 00:20:54,940 –> 00:21:03,880 now let's look more specifically

340 00:21:03,880 –> 00:21:10,440 at just be parameterization of policies

341 00:21:10,440 –> 00:21:13,780 and this kind of objective how do you minimize it

我们来看看更多的内容，有了参数化的策略和这个目标函数，你要如何最小化它

342 00:21:13,780 –> 00:21:18,280 one possibility is to use a random search method

一种可能的方法是随机搜索

343 00:21:18,280 –> 00:21:21,309 there are many many

344 00:21:21,309 –> 00:21:24,670 methods of this type very old and very

345 00:21:24,670 –> 00:21:28,630 straightforward to apply

有很多很老并且很直接的方法可以用

346 00:21:28,630 –> 00:21:31,570 the idea is that you are at a given point at a given set of parameters

这些方法的想法是给定了参数集合的某一个点之后

347 00:21:31,570 –> 00:21:34,720 you look around this

348 00:21:34,720 –> 00:21:39,600 set of parameters you search around and

349 00:21:40,679 –> 00:21:44,590 you look for a better r so I'm at the

350 00:21:44,590 –> 00:21:47,170 given r I know the cost here because I

351 00:21:47,170 –> 00:21:48,610 have evaluated by simulation or

352 00:21:48,610 –> 00:21:52,059 something then I generate different

353 00:21:52,059 –> 00:21:55,440 values around it I calculate the cost

354 00:21:55,440 –> 00:21:59,260 corresponding to each new value and I

355 00:21:59,260 –> 00:22:02,679 move to another point that has better cost

你看这个点周围的点，搜索一个更好的 r，当我在给定的 r 的时候，由于我可以通过仿真或者其他什么方法知道这个点的成本，然后我可以生成不同的 r，然后计算他们的成本，最后选择一个更好的成本的点移动到这个新的 r 上

356 00:22:02,679 –> 00:22:06,220 okay this the rough idea there are

357 00:22:06,220 –> 00:22:07,330 many many ways to construct

358 00:22:07,330 –> 00:22:08,800 neighborhoods to search among

359 00:22:08,800 –> 00:22:12,070 neighborhoods and is a method that has

360 00:22:12,070 –> 00:22:14,559 gotten good publicity recently called

361 00:22:14,559 –> 00:22:15,510 the cross enter

362 00:22:15,510 –> 00:22:17,730 method which you can find in the

363 00:22:17,730 –> 00:22:20,670 literature among others it has been very

364 00:22:20,670 –> 00:22:23,370 successful for this tetris problem the

365 00:22:23,370 –> 00:22:25,230 test case of the tetris problem that we

366 00:22:25,230 –> 00:22:28,590 have talked in earlier lectures and it

367 00:22:28,590 –> 00:22:33,270 beat by a large margin approximate

368 00:22:33,270 –> 00:22:37,230 policy duration methods value

369 00:22:37,230 –> 00:22:42,930 approximation value space

这种粗暴的构造邻域搜索的方法是一种最近被广泛使用的方法，被叫做交叉输入法 (cross enter), 你可以在文献中找到这种方法，这种方法非常成功，，比如俄罗斯方块，我们之前的课程讨论过的问题，这种方法在非常大的值空间中使用近似策略迭代与近似值迭代

370 00:22:42,930 –> 00:22:45,630 you can apply this to discrete state spaces or

371 00:22:45,630 –> 00:22:47,520 continuous state spaces and control spaces

你可以把它应用在离散状态空间或者连续状态和连续控制空间空间

372 00:22:47,520 –> 00:22:52,800 and the performance of random

373 00:22:52,800 –> 00:22:55,410 search methods is notoriously inducing

374 00:22:55,410 –> 00:22:58,380 kradic you may get excellent results or

375 00:22:58,380 –> 00:23:02,640 you may get abysmal failures they can be

376 00:23:02,640 –> 00:23:05,120 very slow it can be very unpredictable

377 00:23:05,120 –> 00:23:12,360 but you can get success often enough to

378 00:23:12,360 –> 00:23:16,430 consider them as viable possibilities
随机搜索是一种臭名远扬的方法，你可能非常成功也可能完全失败，算法可能非常慢，它的效果没法预测，但是考虑到可实现性，你可以用他们成功运行
379 00:23:16,430 –> 00:23:18,900 they are very simple that's that's
380 00:23:18,900 –> 00:23:21,000 what's great about random search method very simple
随机搜索非常简单
381 00:23:21,000 –> 00:23:25,170 and they in theory in paper
382 00:23:25,170 –> 00:23:29,910 they they are guaranteed to converge
在文献中，他们可以从理论上保证收敛
383 00:23:29,910 –> 00:23:32,040 in practice that's not so but in theory
384 00:23:32,040 –> 00:23:33,960 they're guaranteed converge to a global optimum okay
实际中没法保证，但是理论上可以保证收敛到全局最优解
385 00:23:33,960 –> 00:23:42,330 an alternative to a random
386 00:23:42,330 –> 00:23:45,480 search is a gradient type of method like
387 00:23:45,480 –> 00:23:47,220 the ones that you use in nonlinear
388 00:23:47,220 –> 00:23:52,130 programming but perhaps adapted to the
389 00:23:52,130 –> 00:23:55,290 context that we have here including the
390 00:23:55,290 –> 00:23:57,680 stochastic context simulation so on
能够取代随机搜索的一种方法是梯度类型的方法，比如我们在非线性规划中使用的梯度方法，**比如我们之前用过的随机**
391 00:23:57,680 –> 00:24:00,120 these maps are called policy gradient
392 00:24:00,120 –> 00:24:01,260 methods they have also been used extensively
这些方法被叫做策略梯度方法，他们被广泛地应用
393 00:24:01,260 –> 00:24:03,450 they have been researched on
394 00:24:03,450 –> 00:24:06,090 extensively studied extensively again
395 00:24:06,090 –> 00:24:08,460 they are also very unpredictable in terms of their behavior
他们被广泛地研究但是表现同样不可预测
396 00:24:08,460 –> 00:24:11,040 the idea is to
397 00:24:11,040 –> 00:24:12,510 move along the direction of the gradient
398 00:24:12,510 –> 00:24:15,060 you have this cost function that depends
399 00:24:15,060 –> 00:24:17,940 on R you calculate its gradient or an
400 00:24:17,940 –> 00:24:20,190 approximation to the gradient perhaps a
401 00:24:20,190 –> 00:24:21,090 sampled
402 00:24:21,090 –> 00:24:23,070 version of the gradient by taking some
403 00:24:23,070 –> 00:24:25,980 samples of this this cost function and
404 00:24:25,980 –> 00:24:27,720 then you move in the direction of that
405 00:24:27,720 –> 00:24:31,350 gradient according to some step size
这种方法的思路是，沿着梯度方向移动，这个依赖于 r 的成本被梯度或者近似梯度计算得到。梯度可以使采样版本的梯度，使用样本的成本函数来计算梯度然后沿着这个梯度按照一定的步长移动参数
406 00:24:31,350 –> 00:24:33,330 in dynamic programming for particular
407 00:24:33,330 –> 00:24:35,820 formulations there are formulas or
408 00:24:35,820 –> 00:24:37,530 convenient formulas that give you
409 00:24:37,530 –> 00:24:39,780 expressions for this gradient that you
410 00:24:39,780 –> 00:24:41,760 can use and these formulas can be
411 00:24:41,760 –> 00:24:45,630 approximated by simulation
在动态规划中比较特殊的方程，或者说比较方便的方程，给你梯度的表达式，你就可以用这个梯度进行更新了，当然这个表达式也可以用仿真来近似
412 00:24:45,630 –> 00:24:47,850 all gradient methods suffer in principle by slow convergence
413 00:24:47,850 –> 00:24:51,060 local minima and these are no exception
所有梯度方法都存在收敛慢，局部最优等问题，这里也不例外
414 00:24:51,060 –> 00:24:53,130 and because these are
415 00:24:53,130 –> 00:24:56,790 implemented usually with simulation the
416 00:24:56,790 –> 00:24:58,970 simulation noise can be a real problem
因为这里常用仿真来计算，仿真的噪声是一个大问题
417 00:24:58,970 –> 00:25:01,980 there have been successes here but they
418 00:25:01,980 –> 00:25:04,950 have also been don't don't count on it
419 00:25:04,950 –> 00:25:07,530 okay don't count on success you may be

420 00:25:07,530 –> 00:25:14,730 likely but but but chances are that that
421 00:25:14,730 –> 00:25:17,910 to have success you need to experiment a
422 00:25:17,910 –> 00:25:23,480 lot and the outcome is still uncertain
这些方法获得了成功但是这些问题仍然没有解决，你需要很多次试验才能得到结果，结果的效果也是一个不确定的东西

2.3 COMBINATION WITH APPROXIMATE PI

423 00:25:26,010 –> 00:25:30,200 okay so now let's look how
424 00:25:30,200 –> 00:25:32,250 parameterization of point system work in
425 00:25:32,250 –> 00:25:33,960 the context of approximate policy direction
现在我们来看看参数化策略是如何在近似策略迭代中工作的
426 00:25:33,960 –> 00:25:39,480 in in in policy direction you
427 00:25:39,480 –> 00:25:42,210 have policy evaluation and you also have policy improvement
策略迭代中包括策略评价和策略改进
428 00:25:42,210 –> 00:25:44,940 policy improvement
429 00:25:44,940 –> 00:25:48,510 involves a minimization overall policies
430 00:25:48,510 –> 00:25:52,169 right over all controls
策略改进包括在所有策略和控制中找最小化目标函数的策略
431 00:25:52,169 –> 00:25:54,690 suppose we try to do the minimization only over the parametrized class
假设我们试图只在参数空间中最小化目标函数
432 00:25:54,690 –> 00:25:58,160 then you get
433 00:25:58,160 –> 00:26:01,620 approximate policy improvement
然后你就可以得到近似策略改进
434 00:26:01,620 –> 00:26:03,900 you may have approximate policy duration but you
435 00:26:03,900 –> 00:26:06,110 get approximate policy improvement
436 00:26:06,110 –> 00:26:09,480 involving the parametrized class
你会通过参数化近似策略改进进行近似策略迭代，
437 00:26:09,480 –> 00:26:10,950 so let's look at those a little bit more closely
我们来看一个很接近这个方法的例子
438 00:26:10,950 –> 00:26:19,590 given a certain policy defined
439 00:26:19,590 –> 00:26:24,059 by its it's a parameter vector
给定一个通过参数向量定义的策略
440 00:26:24,059 –> 00:26:26,640 we evaluated perhaps approximately with
441 00:26:26,640 –> 00:26:28,860 some critics some kind of approximate
442 00:26:28,860 –> 00:26:31,890 policy evaluation that produces some J tilde of mu
我们通过某种近似方法来进行近似策略评价得到 $\tilde{J}_\mu$
443 00:26:31,890 –> 00:26:35,790 okay so this is a legitimate
444 00:26:35,790 –> 00:26:40,669 cost approximator of policy
这是一个合理的近似策略成本
445 00:26:40,669 –> 00:26:43,950 and then instead of using the policy improvement
446 00:26:43,950 –> 00:26:46,799 process which is like so minimizing this expression
取代最小化这个表达式 (从上往下第一个公式) 进行策略改进的是
447 00:26:46,799 –> 00:26:49,860 we do it approximately by
448 00:26:49,860 –> 00:26:53,250 minimizing over a parametric class of policies
通过最小化策略的参数来进行近似策略改进
449 00:26:53,250 –> 00:26:58,110 now that means that we minimize
450 00:26:58,110 –> 00:27:01,200 over weight vectors of the parametric
451 00:27:01,200 –> 00:27:05,250 class some kind of cost function that
452 00:27:05,250 –> 00:27:09,530 weights the various components here
这就表示我们要在权重向量空间中最小化成本函数
453 00:27:09,530 –> 00:27:14,429 so you want to to minimize over R this
454 00:27:14,429 –> 00:27:17,010 expression and you take weight and you
455 00:27:17,010 –> 00:27:20,309 weigh the various states with with different weights
你想要在 r 空间中最小化这个表达式，这个权重 ξ 向量中每个状态对应的值都不同
456 00:27:20,309 –> 00:27:25,080 and you can try to do
457 00:27:25,080 –> 00:27:27,780 this by some means some type of gradient

458 00:27:27,780 –> 00:27:30,240 method but in the space of parameter
459 00:27:30,240 –> 00:27:32,630 vectors
你可以在参数空间中用一些梯度方法进行寻优
460 00:27:34,840 –> 00:27:38,620 so here a gradient type of method given
461 00:27:38,620 –> 00:27:42,010 a set of parameters of the actor moves
462 00:27:42,010 –> 00:27:44,410 along the direction of this of the
463 00:27:44,410 –> 00:27:46,630 gradient of this expression to get a new
464 00:27:46,630 –> 00:27:49,210 actor value than a new actor value and
465 00:27:49,210 –> 00:27:49,960 so on
466 00:27:49,960 –> 00:27:53,200 perhaps one or more several steps
这个梯度方法给出一个参数集合，行动函数沿着这个表达式给出的梯度进行一个多多个步骤的移动得到新的行动函数
467 00:27:53,200 –> 00:27:57,430 so it's an alternation of critic implement
468 00:27:57,430 –> 00:28:03,190 critic approximations and actor steps
469 00:28:03,190 –> 00:28:10,930 in between the critic changes
这是使用近似评价进行行动改善的替代方法
470 00:28:10,930 –> 00:28:12,880 schemes like that have been used extensively in the
471 00:28:12,880 –> 00:28:14,860 field of adaptive dynamic programming
472 00:28:14,860 –> 00:28:16,810 for continuous pace deterministic
473 00:28:16,810 –> 00:28:19,290 problems particular this adaptive
474 00:28:19,290 –> 00:28:22,690 capable systems unknown parameters
这种方法被广泛地适用于连续确定性空间的自适应动态规划中，它适用于部分参数未知的系统
475 00:28:22,690 –> 00:28:24,790 it's a lot of research here very active field of research
这是一个很活跃的领域，有很多研究
476 00:28:24,790 –> 00:28:29,830 and there are considerable
477 00:28:29,830 –> 00:28:31,740 theoretical issues to be resolved
有很多可以考虑的理论待解决
478 00:28:31,740 –> 00:28:34,180 particularly for stochastic problems
479 00:28:34,180 –> 00:28:35,290 most of the work has been for deterministic problems
部分是随机问题，更多的工作是关于确定性问题的
480 00:28:35,290 –> 00:28:37,780 and even for
481 00:28:37,780 –> 00:28:42,400 those there is a lot of a lot of mystery
482 00:28:42,400 –> 00:28:50,050 about how these methods work
这个领域有很多秘密，比如这个方法是如何工作的
(asking questions
关于 ac 的，不重要，不记录了
483 00:28:50,050 –> 00:28:52,440 any questions
484 00:28:53,929 –> 00:29:06,330 yeah this can be different yeah this is
485 00:29:06,330 –> 00:29:09,120 an this is in the context of actor
486 00:29:09,120 –> 00:29:15,179 critic methods a critic is okay an actor
487 00:29:15,179 –> 00:29:18,809 is evaluated by a critic and then the
488 00:29:18,809 –> 00:29:24,200 actor tries to perform approximate
489 00:29:24,200 –> 00:29:26,640 policy improvement the critic does
490 00:29:26,640 –> 00:29:29,070 approximate policy evaluation the actor
491 00:29:29,070 –> 00:29:31,770 does approximate policy improvement each
492 00:29:31,770 –> 00:29:35,480 one within its own parameter space
493 00:29:35,480 –> 00:29:38,599 [Applause]
问题：p159 提到两个不同的近似是怎么回事
回答：从下往上第一个公式只有一个逼近器，用的参数 r，但是 r 是隐含地被从上往下第一个公式求的
494 00:29:41,850 –> 00:29:47,100 the nation policy space
495 00:29:57,960 –> 00:30:00,610 okay your question has to do with this
496 00:30:00,610 –> 00:30:02,710 okay we mentioned it's possible to have
497 00:30:02,710 –> 00:30:06,730 a different approximation for the cost
498 00:30:06,730 –> 00:30:08,380 function a different approximation for
499 00:30:08,380 –> 00:30:11,409 the policy now in this stage here in
500 00:30:11,409 –> 00:30:12,850 this equation there is only one
501 00:30:12,850 –> 00:30:16,179 approximator the approximator is in R

502 00:30:16,179 –> 00:30:19,570 but it is defined implicitly by an
503 00:30:19,570 –> 00:30:22,929 approximator of cost okay so this is
504 00:30:22,929 –> 00:30:26,139 really an actor system which uses
505 00:30:26,139 –> 00:30:30,570 acrylic in order to define the actor
506 00:30:38,820 –> 00:30:42,370 this R is defined implicitly through the
507 00:30:42,370 –> 00:30:45,669 critic approximation but you will use
508 00:30:45,669 –> 00:30:48,789 this in a scheme which minimizes this
509 00:30:48,789 –> 00:30:51,480 expression
510 00:30:56,440 –> 00:31:00,140 okay so let's take a break for 5-10
511 00:31:00,140 –> 00:31:03,350 minutes and we'll close the course and
512 00:31:03,350 –> 00:31:05,390 you may ask also your questions
513 00:31:05,390 –> 00:31:11,570 and and and so let's get back in the ten
514 00:31:11,570 –> 00:31:13,870 minutes
asking completed)

3 FINAL WORDS

515 00:31:31,620 –> 00:31:38,020 okay so we are very close to the end of the course
这个课程快要结束了
516 00:31:38,020 –> 00:31:43,270 and we carried a lot of
517 00:31:43,270 –> 00:31:46,000 material more than I thought actually we
518 00:31:46,000 –> 00:31:48,430 would but still there are many many
519 00:31:48,430 –> 00:31:50,620 topics that we have not covered
我讲了很多东西，但是还有很多内容没有讲
520 00:31:50,620 –> 00:31:53,290 I'll just make a list of those cases someone
521 00:31:53,290 –> 00:31:55,420 wants to ask some questions I don't know
522 00:31:55,420 –> 00:31:59,020 or after the end of the class
我列了一个清单，如果有没有想到的问题可以问我，课后问也行
523 00:31:59,020 –> 00:32:01,060 we have focused on discounted finite state problems
我们关注了有限状态空间的折扣问题
524 00:32:01,060 –> 00:32:05,980 but with but that there are
525 00:32:05,980 –> 00:32:07,750 continuous time version of those the
526 00:32:07,750 –> 00:32:10,690 semi Markov problems
也讨论过连续时间的问题和半马尔科夫问题
527 00:32:10,690 –> 00:32:12,670 the theory for those the approximation algorithms and
528 00:32:12,670 –> 00:32:15,280 all that are very similar to the
529 00:32:15,280 –> 00:32:16,870 discounted case it's just that the
530 00:32:16,870 –> 00:32:21,780 discount factor is a state dependent
这些近似算法的理论与折扣问题很相似，这个折扣因子是依赖于状态的
531 00:32:21,780 –> 00:32:23,980 we there are stochastic shortest path
532 00:32:23,980 –> 00:32:25,600 problems there are discounted and
533 00:32:25,600 –> 00:32:28,030 involves a terminal state to which you
534 00:32:28,030 –> 00:32:30,480 going to go with minimum expected cost
这个随机最短路径问题也是折扣问题，有一个终止状态，你希望用最小的期望成本到达终止状态
535 00:32:30,480 –> 00:32:33,820 they have similar theory to discounted problems
这个问题有与折扣问题相似的理论
536 00:32:33,820 –> 00:32:36,940 but it's it's a little bit more tricky
但是需要更多的技巧
537 00:32:36,940 –> 00:32:39,790 but a lot of the theory extends
扩展理论更多
538 00:32:39,790 –> 00:32:42,340 average cost problems are still more
539 00:32:42,340 –> 00:32:44,200 tricky but some of the theory expand extends
平均成本问题需要的技巧更多，基本理论也得到了扩展
540 00:32:44,200 –> 00:32:47,440 sequential games where in
541 00:32:47,440 –> 00:32:49,360 addition to the expectation it's the
542 00:32:49,360 –> 00:32:51,640 stochastic nature of the problem there's
543 00:32:51,640 –> 00:32:54,640 also an antagonistic opponent the the

544 00:32:54,640 –> 00:32:58,150 approximation theory for those is is
545 00:32:58,150 –> 00:33:00,450 more complicated as far as I know
546 00:33:00,450 –> 00:33:03,670 incomplete because this is they're just
547 00:33:03,670 –> 00:33:06,460 more difficult problems
序贯博弈除了问题的随机性，还有对手的因素需要考虑，由于这个问题更复杂，所有相关的近似理论也更复杂
548 00:33:06,460 –> 00:33:08,590 and this is the maximization operation that makes things
549 00:33:08,590 –> 00:33:12,390 complicated when you try to do sampling
在我们尝试使用采样来求解问题的时候最大化操作会让求解变得很复杂
550 00:33:12,630 –> 00:33:15,130 continuous space problems well we talked
551 00:33:15,130 –> 00:33:18,940 about them this at some points in the class
我们讨论过一点连续空间问题的内容
552 00:33:18,940 –> 00:33:26,430 we mentioned that that when well
553 00:33:26,430 –> 00:33:28,660 they are interesting in the context of
554 00:33:28,660 –> 00:33:31,540 adaptive dynamic programming and we
555 00:33:31,540 –> 00:33:32,310 haven't gone too
556 00:33:32,310 –> 00:33:34,680 into this subject it's a very fertile
557 00:33:34,680 –> 00:33:36,570 ground for further research and application
我们还关注了自适应动态规划，但是并没有讲的太深入，因为这是一个待研究和应用的领域
558 00:33:36,570 –> 00:33:40,650 continuous time problems to
559 00:33:40,650 –> 00:33:42,810 have with continuous space these are
560 00:33:42,810 –> 00:33:47,300 continuous time optimal control problems
561 00:33:47,300 –> 00:33:51,420 they're very interesting and we have not
562 00:33:51,420 –> 00:33:53,520 discussed them and they also have this
563 00:33:53,520 –> 00:33:57,090 interesting idea of Amy to approximate
564 00:33:57,090 –> 00:33:59,670 the derivative of the cost to go
565 00:33:59,670 –> 00:34:01,740 function as opposed to the cost to the
566 00:34:01,740 –> 00:34:05,010 cost function itself
连续空间的连续时间问题也是连续时间的最优控制问题，他们很有趣但是我们没有讨论太多内容，这种问题的近似是对成本函数的导数进行近似而不是对成本函数本身进行近似
567 00:34:05,010 –> 00:34:08,940 this is an area where there's a lot of a lot of things
568 00:34:08,940 –> 00:34:13,620 to be done to clarify
这个领域还有很多事情待解决
569 00:34:13,620 –> 00:34:16,920 and and some of you may want to look into those
你们中一些人可能希望研究这些内容
570 00:34:16,920 –> 00:34:20,429 adaptive dynamic programming for deterministic
571 00:34:20,429 –> 00:34:23,520 optimal control problems there is the
572 00:34:23,520 –> 00:34:26,190 issue of again if you have continuous
573 00:34:26,190 –> 00:34:28,590 time problems then there's the issue of
574 00:34:28,590 –> 00:34:30,780 cost function approximation of cost
575 00:34:30,780 –> 00:34:32,219 functions everybody's or cost function
576 00:34:32,219 –> 00:34:34,290 differences I think this is an area
577 00:34:34,290 –> 00:34:38,280 where that's not very clear and needs a lot of research
确定性最优控制问题的自适应动态规划，的话题是连续时间问题，关注的是对成本函数的导数或者差分进行近似，我认为这个领域研究地还不是很明白，还需要进一步研究
578 00:34:38,280 –> 00:34:42,060 approximation policy
579 00:34:42,060 –> 00:34:44,280 space I think from my comments you may
580 00:34:44,280 –> 00:34:46,290 have realized that it's not a failsafe
581 00:34:46,290 –> 00:34:48,239 type of method there's a lot of research
582 00:34:48,239 –> 00:34:52,770 to be done in random search and gradient type methods
策略空间的近似你可以从我的评价中看到，不是一个完善的领域，关于随机搜索与梯度方法还有很多内容要做
583 00:34:52,770 –> 00:34:56,400 here's a subject that we did not discuss
这是一个我们没有讨论过的主题
584 00:34:56,400 –> 00:34:59,630 how do we pick the the
585 00:34:59,630 –> 00:35:02,820 the approximation architecture
如何选择近似结构
586 00:35:02,820 –> 00:35:05,340 how do we pick the basis functions

如何选择基函数
587 00:35:05,340 –> 00:35:07,320 suppose that we have our collection of basis functions
588 00:35:07,320 –> 00:35:08,750 to choose from which are parameterized
589 00:35:08,750 –> 00:35:11,310 can we make an optimal choice between
590 00:35:11,310 –> 00:35:14,760 them bye-bye within this parametric
591 00:35:14,760 –> 00:35:18,000 class is it possible to generate basis
592 00:35:18,000 –> 00:35:20,250 functions automatically some very very
593 00:35:20,250 –> 00:35:21,210 important questions
594 00:35:21,210 –> 00:35:25,920 but not enough has been done in this
595 00:35:25,920 –> 00:35:30,830 area yet fertile ground for research
假定我们有一个参数化的基函数集合，现在我们要从参数空间中选择一个最好的，这样就可以自动地生成基函数，这是一个待研究的领域，当前的工作并不是很多
596 00:35:31,310 –> 00:35:33,990 okay here's another area from discussed
597 00:35:33,990 –> 00:35:37,410 at all a lot of the things that we
598 00:35:37,410 –> 00:35:40,290 covered had to do with solving linear
599 00:35:40,290 –> 00:35:44,100 problems such as linear equations or
600 00:35:44,100 –> 00:35:45,060 least squares
601 00:35:45,060 –> 00:35:47,670 squares problems and solve them by simulation
这是一个我们到论的很多东西都包含的内容，我们用它来求解线性问题，比如线性方程组或者线性最小二乘问题，都可以通过仿真求解
602 00:35:47,670 –> 00:35:51,120 in other words using Monte
603 00:35:51,120 –> 00:35:54,180 Carlo methods to solve linear problems
换句话说，用蒙特卡洛方法求解这些线性问题
604 00:35:54,180 –> 00:35:58,290 now these problems arose in dynamic
605 00:35:58,290 –> 00:36:01,470 programming in this class
这些问题出现在动态规划中
606 00:36:01,470 –> 00:36:04,770 but the ideas extend much more generally
但是这些问题的扩展问题会更具一般性
607 00:36:04,770 –> 00:36:07,380 so you might talk about a field that you might name
608 00:36:07,380 –> 00:36:11,100 Monte Carlo linear algebra which is the
609 00:36:11,100 –> 00:36:13,680 use of Monte Carlo methods to solve
610 00:36:13,680 –> 00:36:18,270 linear problems that arise in fields
611 00:36:18,270 –> 00:36:20,610 other than dynamic programming not only dynamic programming
所以你可以谈论一个叫做蒙特卡洛线性代数的领域，使用蒙特卡罗方法来求解线性问题而不仅仅是动态规划问题
612 00:36:20,610 –> 00:36:25,290 there is the husband
613 00:36:25,290 –> 00:36:27,270 work in this area it's an old field by
614 00:36:27,270 –> 00:36:30,480 the way he dates Monte Carlo methods for
615 00:36:30,480 –> 00:36:32,790 solving linear equations had had been
616 00:36:32,790 –> 00:36:36,960 advocated by fond Newman 91950
这个领域是一个老领域了，有数千项工作，蒙特卡洛求解线性方程组可以追溯到 1950 年了
617 00:36:36,960 –> 00:36:38,880 it's an old idea to use simulation to solve
618 00:36:38,880 –> 00:36:41,520 linear problems but it has received a
619 00:36:41,520 –> 00:36:45,660 lot of attention recently and that means
620 00:36:45,660 –> 00:36:52,560 an interesting field for research ok
用仿真来求解线性问题是一个很老的方法了，但是最近几年又被人频繁地使用，这就表示这还是一个有趣的领域

3.1 CONCLUDING REMARKS

621 00:36:52,560 –> 00:36:54,750 now let's focus on the things that we have covered
我们来看一下讲过的内容
622 00:36:54,750 –> 00:36:58,260 we cover met we went
623 00:36:58,260 –> 00:37:00,720 through many methods
我们讲过很多方法
624 00:37:00,720 –> 00:37:02,190 and I want to tell you right ahead of time with certainty
625 00:37:02,190 –> 00:37:03,720 that there is no clear winner there is

626 00:37:03,720 –> 00:37:05,340 no best method if you're looking for a

627 00:37:05,340 –> 00:37:08,010 best method there is no such thing

我要明确地告诉你，没有唯一的赢家也没有最好的方法，如果你想找一个最好的方法，不存在的

628 00:37:08,010 –> 00:37:10,350 there are good methods for some problems

有一些问题是由好的方法的

629 00:37:10,350 –> 00:37:12,780 and it's a variety of methods and it's

630 00:37:12,780 –> 00:37:15,750 important to understand them both

631 00:37:15,750 –> 00:37:20,310 theoretically and and be aware however

632 00:37:20,310 –> 00:37:23,190 that they do not provide higher on clad performance guarantees

有很多种方法，比较重要的是理解他们，不仅理解他们的理论，还要知道他们不能提供严格的性能的保证

633 00:37:23,190 –> 00:37:27,270 you may try a

634 00:37:27,270 –> 00:37:29,820 method but there's no guarantee it's

635 00:37:29,820 –> 00:37:32,070 going to work you may have to try

636 00:37:32,070 –> 00:37:34,530 several methods or you may have to be

637 00:37:34,530 –> 00:37:36,600 more creative about how you apply it or

638 00:37:36,600 –> 00:37:39,360 maybe invent a new method for your problem

你可能试了一种方法但是他不能保证工作，你可能尝试几种方法或者对这个方法进行改进甚至是创造一个新的方法来解决你的问题

639 00:37:39,360 –> 00:37:43,050 all types of methods have major flaws

所有类型的方法都有自已的缺陷

640 00:37:43,050 –> 00:37:46,880 in approximate policy duration

641 00:37:46,880 –> 00:37:51,120 projected equations particular you have

642 00:37:51,120 –> 00:37:54,810 the issue of oscillations in the issue

643 00:37:54,810 –> 00:37:56,350 of exploration

近似策略迭代中投影方程探索会存在震荡问题

644 00:37:56,350 –> 00:38:00,520 in aggregation you have restrictions on

645 00:38:00,520 –> 00:38:02,830 the approximation architecture in other

646 00:38:02,830 –> 00:38:04,480 words the basis functions happy to be

647 00:38:04,480 –> 00:38:05,890 defined in terms of probability

648 00:38:05,890 –> 00:38:10,810 distribution and that's a restriction

在聚合问题中近似结构会受到限制，换句话说基函数需要按照概率分布来设计，这就是我说的限制

649 00:38:10,810 –> 00:38:12,580 if you want to do approximation in policy

650 00:38:12,580 –> 00:38:15,580 space optimization policy space then all

651 00:38:15,580 –> 00:38:18,220 the methods available for this at flaky

652 00:38:18,220 –> 00:38:21,910 are unreliable gradient methods based on

653 00:38:21,910 –> 00:38:25,830 simulation random search these methods

654 00:38:25,830 –> 00:38:32,860 this may or may not work for you

如果你想在近似策略空间中进行优化，所有的方法都或多或多少的不可靠，基于仿真的梯度方法，随即搜索什么的，都可能不工作

655 00:38:32,860 –> 00:38:37,410 despite all this discarding and disappointing

656 00:38:37,410 –> 00:38:40,720 picture there have been impressive

657 00:38:40,720 –> 00:38:44,170 successes with enormous Lee complex

658 00:38:44,170 –> 00:38:47,080 problems for which often there is no

659 00:38:47,080 –> 00:38:51,100 alternative methodology

尽管这些缺陷让人失望，但是在非常复杂的问题中这些方法获得了非常大的成功，而且这些问题通常没有任何代替方法

660 00:38:51,100 –> 00:38:53,710 we are looking at difficult problems for which there

661 00:38:53,710 –> 00:38:56,710 are few methods and you should not be

662 00:38:56,710 –> 00:39:00,790 discouraged by the fact that that we

663 00:39:00,790 –> 00:39:02,020 don't have a methodology that's foolproof

我们研究一个问题时可以用的方法很少，不应该因为这个感到沮丧

664 00:39:02,020 –> 00:39:09,370 we have focused on approximate

665 00:39:09,370 –> 00:39:11,200 policy duration approximate value

666 00:39:11,200 –> 00:39:14,160 duration q-learning

667 00:39:14,160 –> 00:39:17,710 versions amusing few factors but there
668 00:39:17,710 –> 00:39:19,120 are also some other methods that we have
669 00:39:19,120 –> 00:39:22,090 not covered at all
我们讲过了近似策略迭代，近似值迭代，Q 学习，不同种类的 Q 值的算法，但是还有很多方法我们完全没有提到过
670 00:39:22,090 –> 00:39:25,270 we mentioned rollout which is very simple often successful
671 00:39:25,270 –> 00:39:28,750 and generally reliable
我们提到过的 rollout 方法就是很简单很成功而且可靠性很高的方法
672 00:39:28,750 –> 00:39:30,460 if I had to pick the most reliable method within this
673 00:39:30,460 –> 00:39:34,810 landscape but rollout is the one
如果我想选择一个最可靠的方法，那么 rollout 方法就是我想要的
674 00:39:34,810 –> 00:39:37,170 it it doesn't it's not very ambitious but
675 00:39:37,170 –> 00:39:39,910 reliable and often will give you very good results
这个方法用起来不是很消耗资源，但是非常可靠并且能够给出很好的结果
676 00:39:39,910 –> 00:39:43,330 approximate linear
677 00:39:43,330 –> 00:39:46,350 programming which uses linear
678 00:39:46,350 –> 00:39:49,300 architectures to approximate a certain
679 00:39:49,300 –> 00:39:52,570 linear programming formulation of the
680 00:39:52,570 –> 00:39:57,010 bellman equation for MDP is is also
681 00:39:57,010 –> 00:39:59,650 another methodology that has its
682 00:39:59,650 –> 00:40:03,580 successors and has received attention
683 00:40:03,580 –> 00:40:04,990 both fear
684 00:40:04,990 –> 00:40:07,240 it can be impractically it's another
685 00:40:07,240 –> 00:40:13,119 alternative worth considering
近似线性规划用线性结构来近似一个确定的 bellman 线性规划模型这是另一种很成功并且收到很多关注的方法，这是另一个值得关注的方法
686 00:40:13,119 –> 00:40:15,010 I hope I gave the message to you that even though
687 00:40:15,010 –> 00:40:16,480 this is a field where there's a lot of
688 00:40:16,480 –> 00:40:18,460 trial and error theoretical
689 00:40:18,460 –> 00:40:20,710 understanding is very important in the fear is non-trivial
我希望给你们一个信息，虽然这是试错的领域，但是理解理论知识也是很重要的
690 00:40:20,710 –> 00:40:24,520 you need to to dig
691 00:40:24,520 –> 00:40:27,310 into the theory to be able to apply
692 00:40:27,310 –> 00:40:31,030 these methods in practice
你需要去挖掘理论知识来把这些方法应用到实践中
693 00:40:31,030 –> 00:40:33,100 still however for all the theory that you may know
694 00:40:33,100 –> 00:40:36,970 practice is a challenge it's an art in a
695 00:40:36,970 –> 00:40:42,580 challenge to creativity
对于所有理论，实践是一个很大的挑战，这是对创造力的挑战
696 00:40:42,580 –> 00:40:47,530 very difficult problems you should expect
也是你会遇到的很困难的问题

4 Q&A

问题：能不能用线性规划来求解动态规划问题

回答：只能求解有限状态问题，但是没法判断近似线性规划和近似策略迭代哪一个表现更好，这是一个没有理论保证的话题

697 00:40:47,530 –> 00:40:50,550 that so that's it that's all I have to say and
698 00:40:50,550 –> 00:40:53,020 now it's a good time to ask questions
699 00:40:53,020 –> 00:40:55,780 and incidentally if someone has a longer
700 00:40:55,780 –> 00:40:59,170 question perhaps we can discuss it after
701 00:40:59,170 –> 00:41:01,810 the class or during the next couple of
702 00:41:01,810 –> 00:41:03,400 weeks that I'm going to be here I think
703 00:41:03,400 –> 00:41:07,030 why you may send me an email and try to
704 00:41:07,030 –> 00:41:13,440 make a arrange a meeting so questions
705 00:41:19,940 –> 00:41:20,950 yes
706 00:41:20,950 –> 00:41:24,070 [Music]
707 00:41:27,640 –> 00:41:30,820 [Music]

708 00:41:49,049 –> 00:41:54,500 we knew already the pro-soviet
709 00:41:56,390 –> 00:42:03,410 so my question is what is the efficient
710 00:42:06,090 –> 00:42:09,199 [Music]
711 00:42:14,530 –> 00:42:17,780 okay so your question has to do with a
712 00:42:17,780 –> 00:42:19,850 linear programming approach toward
713 00:42:19,850 –> 00:42:23,270 solving dynamic programming problems and
714 00:42:23,270 –> 00:42:26,890 whether this is an efficient approach
715 00:42:26,890 –> 00:42:30,530 for exact implementation and also for
716 00:42:30,530 –> 00:42:33,860 approximate implementation okay let me
717 00:42:33,860 –> 00:42:37,310 say a few things about linear program
718 00:42:37,310 –> 00:42:40,880 this refers to this Bellman’s equation
719 00:42:40,880 –> 00:42:43,430 can be equivalently written as the
720 00:42:43,430 –> 00:42:45,590 solution of a linear program and this
721 00:42:45,590 –> 00:42:48,980 linear program involves involves a large
722 00:42:48,980 –> 00:42:52,910 dimension potentially there is a
723 00:42:52,910 –> 00:42:57,050 variable for every every state and
724 00:42:57,050 –> 00:42:59,930 control pair okay and there’s a
725 00:42:59,930 –> 00:43:03,350 constraint for I’m sorry there’s a
726 00:43:03,350 –> 00:43:05,540 variable for every state and there’s a
727 00:43:05,540 –> 00:43:07,190 constraint for every state and control
728 00:43:07,190 –> 00:43:12,950 pair so so so but it is a legitimate
729 00:43:12,950 –> 00:43:18,080 method for solving exactly finite state
730 00:43:18,080 –> 00:43:20,120 dynamic programming problems it cannot
731 00:43:20,120 –> 00:43:21,740 be applied to infinite state dynamic
732 00:43:21,740 –> 00:43:23,930 programming problems because there are
733 00:43:23,930 –> 00:43:26,300 no infinite dimensional linear programs
734 00:43:26,300 –> 00:43:27,530 there are no algorithms for infinite
735 00:43:27,530 –> 00:43:29,270 dimensional linear programs that are
736 00:43:29,270 –> 00:43:34,820 effective the actual linear programming
737 00:43:34,820 –> 00:43:36,980 and policy direction are related in
738 00:43:36,980 –> 00:43:39,080 other words one can show that the policy
739 00:43:39,080 –> 00:43:41,150 Direction is an execution of some
740 00:43:41,150 –> 00:43:43,280 variant of the simplex method that
741 00:43:43,280 –> 00:43:45,230 solves the same linear programming
742 00:43:45,230 –> 00:43:49,780 problem people have used have looked at
743 00:43:49,780 –> 00:43:52,930 this the issue of comparative efficiency
744 00:43:52,930 –> 00:44:00,080 by complexity analysis and there has
745 00:44:00,080 –> 00:44:01,970 been some interesting work I can’t
746 00:44:01,970 –> 00:44:04,670 remember exactly the details but linear
747 00:44:04,670 –> 00:44:06,470 programming is a polynomial algorithm
748 00:44:06,470 –> 00:44:08,690 and people have worked on complex
749 00:44:08,690 –> 00:44:10,369 analysis for policy direction they have
750 00:44:10,369 –> 00:44:14,480 made comparisons when it comes to
751 00:44:14,480 –> 00:44:17,359 approximations it’s really very hard to
752 00:44:17,359 –> 00:44:20,660 tell which whether approximate linear
753 00:44:20,660 –> 00:44:24,260 programming or approximate policy
754 00:44:24,260 –> 00:44:26,270 direction will work better on a given
755 00:44:26,270 –> 00:44:30,560 problem it’s it’s just one of those
756 00:44:30,560 –> 00:44:32,390 things for which there are no guarantees
757 00:44:32,390 –> 00:44:34,280 in this field you can try both of them
758 00:44:34,280 –> 00:44:36,410 and see which one works better and there
759 00:44:36,410 –> 00:44:38,060 are a lot of cases that is that argue
760 00:44:38,060 –> 00:44:39,859 for approximate linear programming and
761 00:44:39,859 –> 00:44:43,839 also case studies that argue against it
762 00:44:43,839 –> 00:44:47,270 so that’s all I can say there is no
763 00:44:47,270 –> 00:44:48,710 clear answer for the case of
764 00:44:48,710 –> 00:44:57,970 approximations some other questions
765 00:45:03,430 –> 00:45:05,810 okay I want to thank you for your

766 00:45:05,810 –> 00:45:08,540 attendance and like I said you may
767 00:45:08,540 –> 00:45:12,460 contact me after the lecture
768 00:45:12,700 –> 00:00:00,000 [Applause]