APPROXIMATE POLICY EVALUATION METHODS

sorry about all the delays hello okay you hear me okay very good so so we have talked about approximations in general then approximation architectures remember the two main issues that we have is how to select the architecture we have discussed this a little bit and now how do you train an architecture after you have selected it so that's what we're going to talk about now approximate policy evaluation methods

抱歉开始的时间被延后了，你们能听到我说话吧。我们已经讨论过了近似结构的问题，还记得近似结构的两个问题么，如何选择合适的近似结构和如何训练近似结构的参数，下面我要讲一些近似策略评估的方法

DIRECT POLICY EVALUATION

given a certain policy how do you approximate the cost of that policy the cost vector

给定了一个策略，你该怎么计算这个策略的成本向量呢

one possibility is to select this cost vector by projection

一种方法是通过投影选择成本向量

what we have here is this cost vector sitting in a very high dimensional space and we have an approximation subspace defined by a linear architecture

这个成本向量维度非常高，我们使用线性近似结构在子空间内进行近似

so think of J mu has been a gigantic vector in a gigantic space and this subspace being a lower dimensional one and we want to find an approximation within the lower dimensional space

J\_mu是一个高维度空间内的高维度向量，在高维空间内找到一个更低维度的空间并找到一个合理的近似

the most the the idea that immediately comes to mind is projection

最需要考虑的就是投影

define some kind of projection matrix some norm and project this down from J mu to PI sub J mu okay

定一某种投影矩阵范数，把原空间的策略J\_mu投影到pi\_{J\_mu}

now solving this pi is a Euclidean norm where condition assuming the Euclidean norm and Euclidean norms lead to quadratic least squares cost functions you minimize the sum of the square errors between J nu and its projection

这个待求解的pi是一个欧几里得范数，假设这个欧几里得范数会将目标函数变为最小二乘问题，那么这个问题就变成了最小化所有J\_mu和它的投影的平方误差之和

it's a quadratic problem that you may try to solve using data using some data associated with obtain but with policy mu

这是一个二次问题，你可以使用通过策略mu观测到的数据进行求解

DIRECT EVALUATION BY SIMULATION

so this is a sort of a but this comes to mind immediately and let's see how this can be done how can it be done by simulation

这个问题可以通过仿真求解

we can't have the if we're trying to evaluate the cost of a policy we'll try to evaluate the optimal cost is very difficult to obtain because that's what we're looking for

直接求解策略的成本非常难以至于无法求解

however and we can't evaluate the entire vector J mu what we can do is selectively look at some states and obtain the course starting from that state

由于无法计算J\_mu所有的成本向量，所以我们可以有选择性地计算一部分状态的成本

how can we do that we have a system and suppose we have a simulator for this system we simulate on the computer and we have a certain policy that again we can simulate we start at a certain state i and run a lot of trajectories with it

我们该怎么做呢，现在我们有一个系统，并且有这个系统的仿真器，我们可以使用这个模拟器对某个策略从状态i开始进行很多次仿真

of course the system is stochastic and we'll have a lot of generate a lot of different different trajectories but we can average the corresponding costs and obtain an approximation to the expected cost starting from the given state and using that policy

这个系统是一个随即系统，仿真过程中我们获得了很多不同的轨迹，可以计算这些轨迹的平均成本来近似从某个状态开始时某策略的期望成本

in other words ideally to evaluate the projection of J mu we need to solve this least squares problem okay r star just associate with the optimal projector it minimizes the Euclidean norm error between J nu and the approximation

换句话说，评估J\_mu的投影需要求解这个最小二乘问题，r\*是与最优投影相关的量，最小化J\_mu和近似值的欧几里得二范数误差可以得到r\*的值

xi is a weight that corresponds to the different components if xi is a vector of all ones then you've equally weigh the various components more generally you can introduce our weighted awake vector xi which gives you the which ways the different errors with the corresponding values

so that's what ideally what the exact projection is obtained by minimizing this this function here with respect to R

所以理想情况下，最小化这个表达式就可以获得精确投影r

now the minimization can be done by setting the gradient with respect to R to 0 right

最小化这个表达式可以通过让它关于r的梯度等于0

it is a it is a standard problem and it's solution can be written in closed form so this is the solution the exact solution of this projection problem

这是一个标准问题，它的解可以被写成这种形式，求得的是投影问题的精确解

okay you said here we certainly you create a matrix of derivatives and you invert that matrix and this is the exact expression for V for the for the weight of the projector R star

这个公式计算了导数矩阵并且求逆，然后就可以计算投影r\*的精确解

okay now this seems very simple but it has two major problems associated with it

这看起来非常简单但是有两个比较主要的问题需要注意

the first one is that you don't know the exact values of all the JMU's

第一个是你不知道所有J\_mu的精确值

remember that this J mus of Pi has many many components so you can't calculate all of them

还记得策略pi的J\_mu么，这个向量的维度非常大所以无法计算这个向量所有元素的值

the second is that n could be very very large so this involves a very large sum of of numbers and also a very large sum of matrices to embroil

第二个问题是n非常大，这时你需要计算非常多的数字累加和非常多的矩阵累加

so how are you going to calculate this

所以应该怎么计算这个表达式呢

remember what I was saying earlier simulation comes in handy when you want to calculate approximations to large sums

还记得我之前讲的么，在计算非常大规模的累加时，仿真可以派上用场

and if you look at these because xi is a probability vector this is an expected value

因为xi是一个概率矩阵，所以这个表达式算得的是期望值

okay it's expected value of this with the probabilities of various components are given by the disk size this is also an expected value of a matrix so simulation can be used to approximate these expected values and do low dimensional calculations

这个表达式的期望值与各元素的概率给出，这一项是矩阵的期望，所以仿真可以在低维度下近似他们的期望值

in particular we select some states we sample the state space we select states I 1 up to I K and we calculate the cost corresponding to these states by using simulation now you can do this exactly or you can do it approximately

实际计算时我使用仿真在状态空间中采样获得i1到ik并计算这些状态的成本，这样就可以计算他们的精确值或者近似值了

for example if you are to run an infinite number of simulations starting from I 1 then you would get the exact value of K mu at I 1 this should be 1 here

比如你想要从状态i1开始进行无限次模拟，你就可以计算策略mu从状态1开始在k个步骤内的精确成本，在这里这个值应该是1

however you may approximate this J news of i1 using only a few simulation samples maybe a hundred maybe ten okay

如果你想要使用有限次仿真采样计算策略J\_mu在状态i1的近似成本，比如100次或者1000次

so you generate all these cross samples you go here you run the system get a cost value

then you run some go to another state run some cost the simulation get another cost value and so on

所以你通过交叉采样生成这些样本，然后计算他的成本值，然后转入另一个状态进行仿真计算它的成本，再转入其他的状态并计算成本，就这么一直算下去

and you collect all of these samples and you put them together in an approximation of the expected values that go into here

这样你就获得了所有的样本，然后你把它们放在一起，就可以得到这个近似的期望值了

so in place of this expected value we use a Monte Carlo average and instead of this expected value we use a Monte Carlo average

用蒙特卡洛平均，可以得到这两项的近似期望值

and that gives you an evaluation an approximate evaluation of the projection where the solution using the simulation based solution can be viewed as an approximation of the weighting corresponding to the projection

这样就可以计算得到投影的近似估值结果，这是一个基于仿真的近似投影权重

it's a typical example of how you use simulation in this field

这就是这个领域使用仿真的例子

now this is for the problem oh by the way let me also mention something else instead of it's also possible to show that that this this simulation answer can be obtained by solving a least squares problem

我要说一些其他的东西，你可以用这个最小二乘问题的解来代替仿真的结果，

whereby the cost samples the model for the cost samples is compared with the sample values that you get and the error is weighted in a least squares objective this formulation is the is equivalent to this

对成本进行采样，把误差加权累加的最小二乘目标和原问题的目标是等价的

so the original the exact problem of projection is a least squares problem like that the simulation based version is one whereby you replace exact values by sampled values

所以原来的精确问题是一个这样的最小二乘问题，基于仿真的问题就是把原问题的精确值换成采样值的问题

okay so this is the first method direct projection of something of interest like the cost of a policy it's perfectly viable simple but it's not the best method

这是第一种方法，直接对策略的成本投影，这是一种很完美的方法但不是最好的方法

INDIRECT POLICY EVALUATION

it is direct projection but there also have indirect approximation methods that are better suited for our context

相比直接近似，间接近似更适合我们要解决的问题

and one of them is so-called the Galerkin approximation

比如galerkin近似

galerkin approximation is an old method that's used very widely in numerical analysis for solving high dimensional equations like partial differential equations using lower dimensional approximations

galerkin近似是一种比较老的方法了，但是在数值分析中应用非常广泛，这种方法可以使用低维近似方法求解高维方程组，比如偏微分方程组

it's designed for this context approximating high dimensional solutions solutions of of high dimensional equations

在这个课程中这种方法被用来求解高维问题

now in indirect about policy evaluation in galerkin approximation particular instead of projecting the object that you want to approximate you project the equation that the object solves and you solve the projected equation

间接策略评估方法中，用gallery近似代替对你想要近似的目标进行投影，你对目标函数的解进行投影，然后解这个投影方程

so here the equation is X this J mu equals to T mu J nu

所以这个方程是X，这个J\_mu等于T\_mu J\_mu

you project this equation and using Phi R in place of of J mu

你对这个方程进行投影，然后用phi r代替J\_mu

and you solve this equation in place of the original policy of the bellman equation

然后你求解这个方程而不是bellman方程的原始策略

the projected form of bellman equation pi comes in here on the right hand side

bellman的投影形式中，策略pi在等号右手边

and to interpret what this does we're ears here you want to find a Phi R onto which J mu projects here you want to find a Phi R such that T mu Phi R projects is exactly on the Phi R because by the projection involves least-squares again this is a least-squares type of equation but it's not the same as the one that I had before

解释一下这个等式，你想要找到J\_mu的投影中的phi r，这时T\_mu phi r的投影在phi r上是一个精确值，因为这个投影包括了一个与我之前讲的不一样的最小二乘问题

there are several methods the most the most famous methods within this field for solving for policy evaluation out of this type are Garner in type

这里有一些很著名的进行策略评估的方法

and there's a method called TD lambda which is a stochastic iterative algorithm for solving this projected equation

这里有一个叫做TD lambda的方法，是一个用来求解投影方程的随机迭代算法

there's another method called LSTD lambda the lambda parameter let me not discuss what this is TD and LSTD they it's also a simulation based approximation of this equation I'll say a few things about what LSTD really does

另一个方法叫做LSTD lambda，参数lambda我没介绍过，TD和LSTD我也没介绍过，这是一个基于仿真的近似方程，我要说一下LSTD是怎么工作的

this is a projected equation if you make a simulation based approximation to that and solve the this approximation then you get the LSTD method

如果你进行基于仿真的近似的话，这是一个投影方程，然后你求解这个近似方程，就得到LSTD了

LSP methods call this response evaluation the simulation based form of projecting value iteration

LSTD方法把这样算得的评价值叫做基于方针的投影值迭代

instead of solving this equation what it does it it conceives an iteration involving this equation

and then solves this iteratively by by with some form some simulation involved

它不求解这个方程而是设计一个涉及这个方程的迭代过程，然后使用仿真在迭代过程中求解这个问题

okay so these are major names of methods in the field and it's not my intention now to discuss them but just to give you the thumbnails of where things fall this galerkin approximation is a major type of approximation methodologies based on simulation

这些是adp中主要方法的名字，我们现在不打算讨论他们而只是给你介绍一个概念，这些方法中galerkin近似是基于仿真的近似算法的一种主要方法

BELLMAN EQUATION ERROR METHODS

okay now let's go back to the bellman equation error method

现在让我们回到bellman方程误差法

where you minimize the the square of the error in satisfying Bellman's equation

你希望最小化满足bellman方程的平方误差

so within the approximation architecture you want to find one element that makes this error in satisfying the government equation may minimum

所以现在你希望用这个近似结构找到一个元素让这个误差满足bellman方程并且零目标函数值最小

the the quadratic objective with a weight with a vector xi with a weighted by a distribution xi which weighs differently the different states

二次目标函数的权重xi是一个分部形式的权重，不同的状态会有不同的权重

now this bellman equation method is actually a Galerkin method it's just that it uses a different projection norm than the Euclidean than the standard Euclidean norm there's a way that norm of a different kind that makes this a Galerkin method so everything you can say about the Galerkin methods you can also say about bellman equation error methods and that's that's that's an interesting thing

这种bellman误差法实际上是一种Galerkin方法，它用一个与欧式范数不同的投影范数，这种不同于正常的范数让他依然是一种Galerkin方法，所以你可以叫他Galerkin方法也可以叫他bellman误差方法，这是一个很有意思的事情

okay so now how do we implement this indirect method

现在我们看看如何执行这个间接近似方法

this projected equation and or bellman error methods using simulation again the idea is to cut to generate many random samples of states using the distribution some distributions xi either of this norm of the orthonormal projection

这个投影方程，或者叫bellman误差方法使用基于仿真的方法使用分布xi产生很多随机状态，或者是范数，或者是正交投影

calculate many samples of transitions using the policy that you are evaluating

使用你想评价的策略产生很多状态转移样本

then form some kind of simulation based approximation of the optimality condition for the projection problem or this problem here and you use sample averages in place of inner products

对于投影问题或者问题(\*)，一些最优性条件的基于仿真的近似方法可以使用采样平均的方法代替内积计算

and then solve this Monte Carlo approximation of the optimality condition to obtain a vector R

然后求解最优条件的蒙特卡洛近似来获得向量r

so let me summarize all this again

我来吧这些内容再进行一下总结

in direct methods ideally solve this equation or that equation

直接近似方法比较理想的是求解这个方程(p71的投影方程)或者这个方程(\*)

and there are simulation based approximation to these where you pick States

然后对你选中的状态进行基于仿真的近似

ok how you pick them is something that we may discuss next week

如何选择状态下周会讲

you pick States and you generate random transitions from the states

你选择了一些状态，然后根据这些状态随机生成了一些状态转移

and you obtain you can obtain various terms for this error term here okay

然后你可以观察到很多误差项

and and and then you weigh them appropriately you sum them up

然后你对这些误差项赋仪适当的权重并累加起来

and then you you form the corresponding in and then the corresponding simulation simulated equation involves Monte Carlo averages

然后你用蒙特卡洛平均之类的仿真方法进行仿真(译者理解是根据这个仿真方法构造最优性条件)

and you solve that and you obtain a solution

然后求解这个最优性条件获得一个解

I guess I'm used too many words to explain something's very simple

我好像用太复杂的表达描述一个很简单的问题

there's something exact we make a simulation based approximation to it with this problem and we solve the approximate problem

这是一个很明显的问题，我们使用基于仿真的近似方法来近似这个问题，然后求解这个近似问题

(someone asking yes your comment relates with how do we select random samples one possibility is to select them arbitrarily very regularly another possibility is to let the system run using the policy that you are evaluating and that will take you through and automatically pick the states pick the simulation States that's also a possibility in fact has some advantages and theoretical advantages however there are reasons why you may want to add to make the simulation richer not just go through the states that the policy takes you because the policy may have preference for some states but may never visit some other states and you want to be able to get a representative and rich enough simulation sample

通过策略选择状态理论上有好处但是策略会倾向于选择它喜欢的状态这样有些状态就访问不到了，因此让采样仿真随机性更大是有好处的

if you have one recurrent class yes in theory if you have a Markov chain to the singular recurrent class yes you're right if you have a singular recurrent class you're guaranteed to visit all states however with a fixed points with a given policy some states may be visited far more frequently than others it may be a very rare event that a state a particular state may be visited and and and then in a in a in practice that means that it may never be visited ok and so you may you may your approximation may be good in some parts of the state space but not for others it's a major problem the problem works explore explorations make sure that you're getting a representative sample from that policy that samples a lot of states so that you build an approximation that's good enough across the state space and not within a subspace on it

为了保证样本的代表性，使用给定策略进行采样时需要加强采样的多样性

have any questions here ok so here are the issues for indirectness how do you generate the samples ok and also how do you calculate the R vector the solution efficiently from the simulation based approximation so both of these are major issues asking and question finished)

这里讨论了关于间接近似方法的两个问题，第一个是如何进行采样，第二个是如何使用基于仿真的近似方法有效计算向量r的值

ANOTHER INDIRECT METHOD: AGGREGATION

ok let's talk about another indirect method

让我们来讨论另一个间接近似方法

this method is called aggregation it's an old method that appears in many different fields

叫做聚合法，这是一个老的方法但是被应用在很多领域

you have a large number of states now many of these states are going to be similar to each other ok think of a case of states pinch of graphically related you can think of many examples

状态空间非常大，但是很多状态都相似，比如图像之类的，类似的例子很多

the idea is to group many states that are similar grouped them together into single group States or aggregate States

想法是把相似的状态放到同一组里当成一个状态，或者叫聚合成一个状态

so the number of original States may be gigantic but you introduce only a few groups of states

原始状态数量可能非常大，但是你可以把他们聚合成很少组的状态

and the idea is to assign a common value of course to each group

为每组状态指定一个值(比如成本)

so like a piecewise constant approximation over groups

就像对每组状态进行分段近似

solve an aggregate dynamic programming involving the aggregate states to obtain the the cost associated with the groups and then use a common cost for every state within the group this is called hard aggregation there are many different types of aggregations the simplest type of aggregation

求解一个聚合状态的聚合动态规划，这种聚合被叫做hard aggregation，是众多聚合类型中最简单的一种

here's an example of hard aggregation

这是一个hard aggregation的例子，

here we have nine states and we introduce groups one group is this X 1 it has states 1 2 4 & 5 X 2 only two states and so on

有九个状态，把他们分组，第一组x1包括状态1，2，4，5，x2只包括两个状态，

and we're going to sort of formulate a problem that operates on the X space as opposed to the high space

我们要把这个问题限制在x空间内而不是原来的高维空间

we can encode these dependencies this membership relations within the different activities in this matrix here

我们把这些与不同行动相关的不独立状态进行编码产生这个矩阵

this matrix okay it has the row dimension is the number of states it's 9 in this particular case

这个矩阵的行数与状态的数量有关，这个例子中等于9

the column dimension is the number of aggregate States is 4

列的数量是聚合的状态数量，这个例子中等于4

so you see a 1 every time as the corresponding state belongs to the corresponding aggregate group okay

所以你每次可以看到一个属于某组的状态

so all of this can be encoded in this matrix field

所以所有的状态都可以被编码到这个矩阵中

and it is possible to generalize this idea and and within a more mathematical view solve a problem of this form involved in the current policy where PR is an approximation to the to the cost of policy mu okay

也可以从更一般或者数学的角度来看这个问题，求解当前策略mu的表达式/xi r=/xi DT\_/mu(/xi r)，其中/phi r是策略mu的近似成本

in the case of hard aggregation what this phi are is it's a piecewise constant approximation but generally if you choose a different matrix phi and also you choose a matrix D which is a probability distribution this type of equation generalizes the hard aggregation idea

这个hard aggregation的例子中phi是一个分段常数近似，但是通常情况下，如果你选择了另一个矩阵phi，也就是选择了一个D，即概率分布，这种类型的方程一般就是硬聚合的思想

you can get some some intuition about what this phi Andy do is that if you consider the original system if you if you okay this is the system involving rows and columns if you combine different columns different rows using a probability distribution D that that reduces the row dimension if you combine different columns using a matrix phi then you get a smaller column dimension so in this way you shrink the dimension of the system and you obtain a system whose rows and columns are obtained by weighing the call rows and columns of the original

你可以得到一些关于phi在做什么的直观解释，如果你考虑一个原始系统，这个系统包括行和列，如果你用概率分布D把不同的行进行组合，就可以减少行的维度，如果用矩阵phi对不同的列进行组合，就可以得到更少的列维度，这种方法你可以缩小系统的维度，这个新系统的行和列可以通过对原问题的行和列加权获得

with some more general view of the aggregation idea and notice that the aggregation equation has a very similar form to the projected equation

从更一般的角度上讲，聚合思想和聚合方程与投影方程的形式很像

in fact if Phi D Phi the big matrix like this D are matrix like that if that matrix is a projection matrix then the aggregation method is a special case or a projection method

事实上，phi是一个大矩阵，D也是一个大矩阵，这个矩阵是一个投影矩阵，那么局和方法就是一个特殊情况下的投影方法

it turns out in fact that hard aggregation can be viewed as a projection method with a special choice of projection matrix

事实上，硬聚合可以被看做一种特别选择投影矩阵后的投影方法

okay now aggregation can be generalized a lot like I said any matrix phi and any matrix D whose called whose rows are probability distributions and columns and nd whose was also which its rows are probability distribution defines an aggregation method

现在聚合方法可以被概括为：任意矩阵phi和任意矩阵D，行是概率分布，列也是概率分布

so for example if this row here were not did not have a one in the first position and zeros and let's say I had a row of one fourth one fourth one fourth one fourth that would mean that state one has it can be viewed as a member of each one of this group with probability 1/4

举个例子，如果这一行第一个位置没有1，我就可以说我有一行1/4，1/4，1/4的数据，这表示状态1可以被看作组内每个状态出现概率都是1/4的状态

in other words the entries of this matrix more generally encode the degree of membership of a state within an aggregate state it's the same thing the same kind of interpretation the rows and columns of T

换句话说，这整个矩阵更一般地编码描述了状态与聚合状态的从属关系，这与矩阵T的行与T之间关系的解释是相同的

let me not go into this but I just want to tell you that there are many many different ways of doing aggregation and this is one way

我不会深入讲这个问题，我只是想告诉你状态聚合有很多方法，硬聚合只是其中一种

AGGREGATION AS PROBLEM APPROXIMATION

and there is also an interesting way to view aggregation as problem approximation

有一种比较有意思的想法是把状态聚合当成一种问题近似的方法

the original system is given here at the top

原系统在上面给出了

you go from state I to state J with probabilities P IJ and with costs GI of U and J

你以概率pij从状态i转移到状态j，带来了g(I,u,j)的成本

now if you establish a relationship between the original States and aggregate States

如果你建立了一个从原状态到聚合状态的关系

and introduce the system here then you can think of transitions of going from aggregate state to original state a transition and back to aggregate state

这个系统在这里介绍，你可以考虑一个从聚合状态到原状态的转移，和一个从原装胎到聚合状态的转移

and define through these entries of the matrix D and the matrix P define a system involving just the aggregate States

然后就可以使用矩阵D和矩阵P定义一个只包括聚合状态的系统

the more complicated system is approximated with the system whose transitions are between aggregate states

更复杂的原系统被近似成了聚合状态表示的系统

it's a little bit complicated context concept to grasp and we need to discuss more examples

这个概念有点复杂，我需要多举几个例子解释一下

however the key thing is that if you look at this bottom system it's a lower dimensional system

你看下面这个系统，是一个更低维度的系统

whose transition probabilities are determined by the transition probabilities of the original and whose cost function is also determined from the from the from the cost function of the original system

转移概率和成本都被原系统的转移概率和成本决定

and this transition probabilities the PI the P hats and the G hats can be set up so that you solve this lower dimensional problem an aggregate problem get the costs for that the cost of states of this aggregate problem and then use them to approximate the cost of the states of the original problem

转移概率P hat和G hat被这样设置，你求解这个低维的聚合问题的状态成本，然后用这个成本来近似原问题的状态成本

so here are the main elements of this methodology solve exactly or approximately this aggregate problem involving aggregate probabilities and aggregate costs

所以这就是这个系统的树妖部分，求这个包括聚合改率和聚合成本的聚合系统的精确解或者近似解

it's possible to do that with any kind of value duration or policy duration method

你可以使用任何形式的值迭代与策略迭代求解

including simulation based methods you build a simulator that moves from aggregate state to aggregate state using a simulator of the original

包括基于仿真的方法，你建立一个仿真器从一个聚合状态转移到另一个聚合状态，这个状态转移是原问题的仿真器进行状态转移，然后再做一次原状态到聚合状态的转换

and then you use the optimal cost of the aggregate problem which gives you a value for every state every aggregate state to approximate to make a piecewise or semi piecewise approximation for the cost of the original problem

然后你用聚合问题的最优成本给出每一个聚合状态的成本来做一个分段或者semi piecewise(不知道这是个啥)来近似原问题的成本

so this is also a problem approximation the original system where transitions take place between the original states is approximated by another system that involves these aggregate states and the transitions now are like this back like this go up and down and any simulator takes advantage of this kind of system and what you get is an approximation based on this structure

这是一个对原系统的问题近似，原状态转移被另一个系统的聚合状态转移代替，就像图上这样，任何一个仿真器对于这种类型的系统都能得到很好的效果，然后你就可以基于这种结构进行近似

now one nice thing about aggregation is that you can use the algorithm that use the policy duration algorithm you use is an approximation for the original problem but some exact I'll good for the aggregate problem

一个比较好的事情是你可以使用策略迭代计算原问题的近似解，但是聚合问题可以用精确算法求解

as a result it behaves more regularly than in the right general projection equation approach

因此它比一般的投影方法更有规律

the disadvantage of this aggregation approach is that it is kind of restrictive on the approximation architecture that you use because it involves this probability vectors and what is the projected equation approach I can use more general type of approximation architecture

聚合方法的缺点是对你能使用的近似结构有限制，因为它包含这个概率向量但是在投影问题方法中我可以使用更一般的近似结构

the two major methods for policy evaluation for approximation value space and policy evaluation are these projected equation types in aggregation types

值空间近似和策略评估这两种主要的策略评估方法都是投影法中的聚合问题的方法

(someone asking questions so now let me say a few things you have any questions please they can be completely unrelated yeah they can be completely unrelated of course if there are relations you can get different specific methods but we okay the disaggregation probabilities are the elements of this matrix D and the aggregation problems all the elements of this matrix fish so the aggregation probabilities are what you see here in this particular case and the only requirement is that the rows of V and D are probability distributions that's the only requirement for different choices of course we get particular interest particular methods that are interesting so this hard aggregation soft aggregation the theory okay

如何选择phi和D 通过直觉，观察误差界限

so what you're saying is how do we choose P and D that's the that's the question and the answer is by intuition to a great extent you can you can look at error bounds there are error bounds that are associated with this methodology but when it all comes down to application it's really intuition how what makes sense from an intuitive point of view as an aggregate state and that's if you have a particular problem like a queueing problem for example you can you can think of good aggregate States pretty easily in one aggregate state is when the system is nearly empty if it has zero customers one customer to the say up to five okay that's one aggregate state another aggregate state when it's mid loaded another state wins the heavy loading fail okay that's so you can take a system you can take a queue that has many many states and and reduce it down to four or five aggregate States that would be an example of intuition and similarly if you have a queuing network for example that might be okay so we're not doing very well with time with all the delays so I have a few more slides

APPROXIMATE POLICY ITERATION

ISSUES

and let me cover the first let me cover a few rather quickly okay so we have issues of approximate policy evaluation now policy Direction has two phases policy evaluation and policy improvement

我们要讲近似策略评估的问题，首先我快速讲一下接下来我们会讲的内容，现在策略评估有两个部分，策略评估和策略改进

THEORETICAL BASIS OF APPROXIMATE PI

the policy evaluation the questions actually are quite easy

策略评估问题很简单

we it's the policy improvement questions that are harder in fact

策略改进问题要难得多

okay so what's the theoretical basis for approximate policy Direction I mentioned last time

我要最后一次讲策略迭代的理论基础

suppose that you do the policy evaluation approximately within Delta

假设你用delta进行近似策略评估

and you do the policy improvement also approximately within epsilon could be the epsilon equal to zero

用epsilon进行近似策略改进，这个epsilon趋于0

then there's an error bound the sequence of policys that we generate come within this bound of the optimal

我们生成的策略序列与最优解间会有一个上界

so what this says that is that eventually you're going to get cost functions of policies that we are within a zone of the optimum of this size

所以最后你获得的策略的成本函数会在这个最优解附近的区间内

typically the way these methods behave in practice is that you make steady progress up to a point you may start with a bad policy that has a cost that's high and then like the next iteration may get something that's better and better and better up to the point where you get within this zone close to the open and then you typically may oscillate within that zone

通常在实践中这些方法可以从一个成本比较高的坏策略稳步前进，每一次迭代都能让策略越来越好，即成本越来越低，最后进入靠近最优解的区域，在这个区域内震荡收敛

the oscillations are quite unpredictable it's very hard to tell what's going to happen it may be that there are no oscillations at all

震荡是不可预测的，很难说它什么时候会发生，也可能完全震荡

aggregation methods actually do not have any oscillations at all

实际上聚合方法是完全不震荡的

they they terminate somewhere within that zone

他们最终会收敛到这个区间内

however projection equation metastatic lead to oscillations and there's some very discouraging examples of oscillations that have been constructed

但是投影法的转移性会导致震荡，这里构造了一些让人沮丧的震荡例子

where you don't get good performance at all the method gets tacked into a barrel slide

如果你没有获得好的表现，那么这些方法都会被放到xx里

in practice people have not have not seem to worry very much about oscillations far from the optimum which this bomb is kind of loose here okay it's a gigantic number if alpha is very close to one this is a huge number

实践中人们并没有很担心震荡偏离最优解太原，这种发散很松散，如果alpha接近1，那么此时误差就是一个非常大的数字

but people don't seem to worry about that the practical studies that people do the case studies that people do they do not seem to report any substantial problems with oscillations

但是人们看起来并不担心实际案例的研究，人们进行研究时没有报告任何实质性的震荡问题

there are some arguments about this but in any case there is you may read about some very bad examples of oscillations and with fairly simple systems okay so this is the theoretical basis

有一些关于震荡的讨论，但是任何情况下如果你看到了非常坏的简单系统震荡的例子，这就是策略迭代的理论基础

THE ISSUE OF EXPLORATION

now here's an here's an issue that came up just a little while ago with your question having to do with exploration how do you construct simulation samples for a given policy

这是一个你们不久前刚刚问过的问题，如何进行探索，如何对给定策略构造一个仿真进行采样

a given policy has a tendency to go to some states perhaps but not to others

一个给定的策略会趋于访问一些特定的状态，不访问另外的状态

so if we generate the core samples using that policy this bias is the simulation by under representing states that are unlikely to occur under this policy as a result the cost estimates for the underrepresented States may be highly inaccurate

如果我们使用一个策略生成核心样本，估计偏差会由这个策略导致的不太可能出现的状态产生的，这样导致的后果是代表性不足的状态估值可能会非常不准确

we form a least-squares objective involved in simulation samples if the simulation samples do not do not represent some states then there is nothing in our optimization this quest optimization that we'll provide that will provide an incentive for good approximation of the states

我们构造了一个涉及仿真样本的最小二乘目标函数，如果仿真样本没有代表一些状态的话，他们就不会出现在我们的优化目标中，这鼓励我们对状态进行更好的近似

so if you have an inaccurate inaccurate cost approximation for some states when you do policy improvement this may give you nonsense for the improved policy

所以如果你对状态的成本近似不准确，那么当你进行策略改进时可能会导致无意义的改进

so serious problem in fact the most serious problem for a practical point of view is how to construct a rich enough simulation sample to simply original set of samples to make sure that the space is sufficiently well explored

事实上这些这么严重的问题从实践的角度来看变成了如何构造一个足够丰富的仿真样本来简化原样本集来保证采样对空间进行足够的探索

so this is known as inadequate exploration if you have a deterministic system or if you have a nearly deterministic system that's a very acute problem because a deterministic policy will not randomized at all between states it will just go along a single trajectory and as a result we the some states will never be visited if you just follow this policy if you just follow single trajectories of this of this policy what you need is to is to create a more artificially create a rich sample by introducing states and transitions that do not are not associated with that policy

所以这被称为不充分的探索，如果你有一个确定性系统或者几乎是确定性的系统，这是一个很棘手的问题因为一个确定性的策略不会随机地访问所有的状态，他只会沿着一条轨迹进行决策，导致的后果就是如果你只使用这条轨迹或者这个策略，你只能访问轨迹上的状态，而不在轨迹上的状态从来都不会被访问，为了解决这个问题，你需要一个更有创造性更智能的富样本集，多进行与策略无关的状态转移

so there are some remedies that we are going to discuss later restart the simulation and ensure that the initial States a form a rich and representative subset

因此，我们将在稍后重新开始模拟并确保初始状态形成一个丰富且有代表性的子集时，会有一些补救措施

occasionally generate transitions that use a randomly selected control rather than the one that's dictated by the policy there are also other methods that use two Markov chains one for selecting generating States and the other is to generate state transitions and all of this is one of these are very important topics for research within this field

偶尔使用随机选择控制进行状态转移而不是一直使用策略规定的转移规则，还有其它方法能达到同样的效果，比如用两个马尔科夫链，一个选择新的状态，一个选择状态转移，这都是研究时很重要的问题

APPROXIMATING Q-FACTORS

okay we have been talking about Jan about approximating cost functions

我们已经讨论过成本函数近似了

however if you get this approximation then the policy improvement requires a model knowledge of transition probabilities for all the controls

如果你近似状态的成本，进行策略改善的时候就需要知道模型信息，即所有控制下的状态转移概率

and if instead we approximate Q factors then we don't need a model

但是如果我近似Q值的话，就不需要知道模型了

that's the advantage of Q factors I mentioned that earlier

这就是我之前讲过的Q值的好处

and it is possible to use a parametric approximation for Q factors which would be with have features of both state and control and weigh them with corresponding weights are

近似的时候就可以用一个参数化的方式来近似Q值，特征中包括状态和控制，对他们进行参数化赋以相应的权重

any method that that I mentioned earlier projected equation segregation bellman error and so on can be adaptive to adapt it to work with Q factors rather than States

我之前提到的任何方法，包括投影法，聚合法，bellman误差法之类的都可以应用到Q值上

the reason is that there is a there is a system who state our state costs are the Q factors you may construct a Markov chain that has States not the state I of the original but state control pairs define transition probabilities between state control pairs to other state control pairs as dictated by the policy and you can apply all these methods I gave earlier to this other system that involves a state the state control pairs

原因是我们有一个Q值表示成本，你可以构造一个不使用原系统状态i的马尔科夫链，此时马尔科夫链的状态是状态-控制对，再定义某策略两个状态-控制对之间的转移概率，这时你就可以使用我之前讲的所有的方法来求解这个状态控制对做系统状态的新系统了

so we can talk about the cost function approximation and everything applies to to Q factor approximation the methods transfer through

所以我们之前讨论的成本状态近似的所有方法都可以适用于Q值近似

on the other hand with Q factors for a given policy exploration is a bigger concern because from a state I u the only kind of state that you can get to is a state that you this would be J here by the way these states where the control is the one assigned by the policy at J so from I you you cannot go to many states you can on Google fewer states only a subset of states and there's a problem of exploration

换种说法，对于一个给定策略的Q值，探索是一个更大的需要考虑的问题，因为从一个状态i-u，你能到达的新状态是被策略指定的，顺便说一下，某个状态的控制是被策略指定的，所以从状态i-u出发你不能到达很多状态，你只能到达一个很少的状态子集中的状态，所以这会对探索状态空间造成一些困难

SOME GENERAL ISSUES

okay so let's take a five minute break and then I have two or three more slides and then we'll finish some general issues on on on on this approximation methodology you may also think of some questions I'm sorry I had to rush a little bit in the last part of this

我们要休息五分钟，接下来差不多还有两三页slide吧，我要讲一点近似方法，你可以想一些问题因为接下来讲的会比较快

STOCHASTIC ALGORITHMS: GENERALITIES

okay let me turn on this mic hello okay their way okay

让我把麦克风打开

some generic issues that do not fit into what I discussed so far but they are important

一些不符合我讲的内容的通用的但是很重要的话题

one has to do with with algorithms user use simulation

一个是用仿真的算法

remember that what we were interested in is solving some linear systems of equation using sampling like the linear system of equation that we solve is the evaluation equation the bellman equation for a given policy this a linear system

还记得我们之前关心的使用采样求解一些线性系统方程嘛，线性系统方程比如给定策略的评价方程，bellman方程之类的线性系统

and it's generally of this type x equals B plus ax X is not because confusable a state X is the variable of this generic linear system okay

这个一般性的系统是这样的(x=b+Ax)，不要搞混了，x不是系统状态，而是通用线性系统的变量

so any linear system can be written into this form a is a square matrix here and B is a vector and we want to find the vector X that solves this equation we can do it by matrix inversion but many people use simulation to solve such systems in specific contexts

任何一个线性系统都可以被写成这种形式，A是一个二次矩阵，b是一个向量，我想要解这个方程获得向量x，我可以通过矩阵求逆来解，但是在特定的场景下，很多人使用仿真来求解这样的系统

suppose that we don't have access to a and B ok a and B is something that that can be calculated only approximately within some sampling error instead what you have is simulation samples of B and a so B is observed M times with error wk and a is observed K M times with with with error WK n WK and capital W K random they're just the simulation noise so all we have is this simulation samples and we want to solve approximately the system of equations

假设我们无法访问A和b，而且A和b是可以在一定采样误差内近似计算的，如果你有B的一些仿真样本，b被误差w\_k下观测(采样)m次，A也在误差W\_k下观测m次，w\_k和W\_k都是随机变量，被称作仿真噪声，所以我们只有这些仿真样本，我们想要近似求解这个系统方程

what kind of methods are there for doing so

这些方法是这么做的

well we have the particular context in mind where this is the system is the approximate value is the policy evaluation system either the projected equation or the aggregated equation creation equations

我们有特定的场景，这个系统的近似值是由投影方程或者聚合方程构造的策略评价系统

now there are two kinds of methods

有两种类型的方法

one is called stochastic approximation

一种叫随机近似

in the stochastic iterative algorithm

在随机迭代算法中

whereby you start with a certain vector X 0 some guess of the solution

你从一个解向量x0开始

and then you use the current sample ok

然后用当前的样本

of evaluate the sample version of the right-hand side of this equation

等号右边的表达式进行估值

ok the sampled B and the sampled a multiplied with XK

样本b和样本A乘以x\_k

that gives you sort of a direction to go

给你一个新解的迭代方向

however because this term is random you should not go too far if you should some use some kind of a step size gamma K

但是因为这一项是随机的，所以步长gamma\_k不可以太大

now this step size should be diminishing actually in order to average the effect of randomness here typically gamma K is taken to be proportional to 1 over k ok something like that

为了平衡随机因素的影响，步长需要逐渐减小，一般步长会取1/k

and you you form the XK plus 1 is XK plus gamma K times the error between the sample the sampled the sample the error in the sampled version of the equation the sampling okay this quantity minus X\_k

这个迭代式中，x\_{k+1}等于x\_k加gamma k乘以样本时间的误差，也就是这一项减掉x\_k

now this is a very old methodology dates the 50s it has very strong theory people still working on it

这是一个很老的方法，可以追溯到50年代但是他又很强的理论支持，人们还在用

and it's one possibility within our context stochastic approximation

一种可能是根据上下文，进行随机近似

another possibility is to use this samples of b and A to construct Monte Carlo averages ok

另一种可能是使用这些b和A的样本来构造蒙特卡洛平均

this is the Monte Carlo average of the samples of B ok

这是蒙特卡洛平均的样本b

and this is the Monte Carlo average the sample of A

这是蒙特卡洛的样本A

this is some vector B M and a m

这是一些向量b\_m和A\_m

and then you form an approximate sample equation where the samples are replacing we the sample averages are replacing the true quantities

你用这些向量b\_m和A\_m代替系统方程中的真实数据构造了一个近似方程

and then once you have this equation you solve it either by matrix inversion or you solve it by using some kind of iteration

一旦你获得了这个近似方程，你就可以使用矩阵求逆或者用其他迭代方法求解

both of these methods are important within our context

在我们的课程中，这些方法都很重要

I mentioned earlier this method of TD lambda and q-learning

我之前提到过这个方法，TD(lambda)和q-learning

TD lambda Q learning are examples of this approach here

TD(lambda)和q-learning是这个方法的两个例子

you collect samples and you make iterations of either r values weight values or J or Q values Q factors

你收集样本，进行r权重或者Q值的迭代，

in Q learning X is a Q factor vector

q-learning中x是一个Q值的向量

and we actually Don iterate on the entire vector but we iterate one state control component at times and a synchronous version of this stochastic approximation that's what Q learning will turn out to be

我们迭代时实际上是对整个向量进行迭代，但是我们每次只对一个状态控制对进行迭代，这是一种同步角度的随机近似，这就是Q-learning

TD lamda operates in TD lambda the X vector here is the R vector the vector weight

在TD(lambda)中x向量是r向量，也就是权重向量

it operates on weights using samples in a an iteration that is a special case of this form

在一次迭代是，他使用样本对权重进行更新，这是这种形式的一个特殊情况

however there are other methods I mentioned LSTD and LSPE

还有其他方法，比如我提到的LSTD和LSPE

they are of the second type

这就是另一种类型了

you collect the samples you form averages of the matrix and vector that's involved in the evaluation equation

你收集到了样本，用这些样本计算了评估方程中的矩阵和向量的平均值

and then you use matrix inversion for this one all iterations for this one

然后你用矩阵求逆在所有迭代过程中进行计算

as a fundamental division to different kinds of methods and both of them are useful within our context and they allow you to solve linear equations where the composite be the data of the equation is unknown and can only be approximated by simulation so we will come back to this type of methods but I'm giving you now an overview of where they fit

作为一种对不同方法的基本划分，他们在我们的课程中都是有用的，他们让你能求解一个数据复合而且未知的线性方程组，这种方程组只能用仿真来求解，所以我们会回忆这种方法但是我要给你进行一次概述，讲一讲他们适合什么情况

COSTS OR COST DIFFERENCES?

the second topic I want to mention is that is whether we should be approximating cost well it's a good idea to approximate cost functions

第二个话题我想讲我们为什么要近似成本，近似成本是一个很好的想法

should we approximate cost functions or something different

我们应该近似成本函数或者其他的么

there is an argument that says that we should approximate cost function differences

有一种说法是我们应该近似成本函数的差

in other words not the cost at a given state X but rather the difference between the cost at two different states

换句话，不是近似状态x处的成本，而是两个不同的成本间的成本差值

why do we want to do that

我们为什么要这么做呢

well suppose that we are at a state X and we want to compare two controls a U and u Prime in policy improvement

假设我们现在处于状态x，想要在策略改进中对比两个控制u和u’

then we should minimize expected value of G Plus alpha J mu

然后我们应该最小化g+alpha J\_mu的值

but to evaluate the two controls we should evaluate this quantity and subtract what this quantity and the answer depends on the sine of this okay

但是为了评估两个控制，我们应该计算他们的值并相减，结果是这个

so what enters into this side is the cost difference not the cost values

所以等号右边的应该是成本的差值而不是成本值

in other words if I take a cost approximation and I raise everything by a gigantic amount I'm going to get exactly the same new policy

换句话说，如果我近似了成本函数并且把所有成本都提高了非常大，那么我会得到一个相同的新策略

the policy improvement operation does not involve does not depend on a cost constant shift of the of the function that you approximate

策略改进不依赖于你近似的成本常数

therefore it is cost function differences that are important not costs themselves okay

这时候成本函数的差值而不是成本就很重要了

look at this expression is the bellman equation expression for you and the bellman equation expression for u Prime where X bar and X bar prime are the next state corresponding to u and u Prime

看这个表达式(第一个圆点的那个)，这个表达式是bellman形式的

so this cost function different is what's the important thing

所以成本函数的插值很重要

and and of course it's possible to evaluate cost function different by approximating separately

所以可以通过分别近似估算成本函数差值

this approximating separately that and take the difference but the difference is subject to noise okay

分别近似差值，但是这个差值是依赖于噪声的

you're subtracting two quantities that involve noise

你用两个带有噪声的数相减

you're likely to get garbage if the two quantities are similar to each other it's a serious concern

如果这两个数大小相近，那么你可能会得到一个没有用处的数，这是一个很严重的问题

now everything that I've said about approximating costs also approach applies to approximating cost differences instead of calculating an approximation to this I can calculate an approximation to D mu corresponding to two different states

我之前说过的所有近似成本的方法都可以应用在成本差值的近似商，我可以计算一个近似的两个状态间的成本差值D\_mu来代替近似成本的差

and the reason is very simple if you be the the cost difference function satisfies the bellman equation and it's a bellman equation corresponding to a system that has a states pairs of states of original system in particular d mu the cost function difference satisfies this equation

原因很简单，如果对应于原系统的状态对的成本差值函数满足bellman方程，那么成本差值函数D\_mu也满足这个方程

where the one-stage cost is the difference of one stage cost that comes in here plus alpha times the difference of costs at the next states X bar is the next date starting projects in human policy mu X bar prime corresponds to the next state starting from X prime

一步成本的差值加上alpha乘以后续累加成本的差值也满足bellman方程，这里x bar是策略mu下的下一个状态，x bar prime是x prime相关的下一个状态

so if I view X X prime as the state of a new system and use as cost function this then the bellman equation for that system satisfies this equation and gives you the cost function differences

如果我把x和xprime当成新系统的状态并把它们用于成本函数，那么这个系统的bellman方程就会满足这个方程并且能够给出成本函数差值

so everything I said before about direct approximation and direct approximation projected equations and related equations bellman error applies to this system as well

我之前讲的所有内容，直接近似，间接近似，投影方程，bellman误差都可以应用在这个问题上

and that's a way this is called differential training and it is you can you can learn these of em this of mu with with the standard methods

这种方法可以叫误差学习，你可以使用标准方法学习D\_mu

now let me show you an example where cost function differences are important I mentioned earlier in the previous lecture that when it comes to continuous time optimal control it's the gradient of the cost to go function that's important and not the cost to go function because the hamilton-jacobi bellman equation involves the gradient of not J and here's an example

现在我要给你举个我之前讲过的成本函数差值很重要的例子，这是一个连续时间最优控制系统，相比cost to go函数，cost to go的梯度更重要，因为HJB方程包含J的梯度而没有J

AN EXAMPLE (FROM THE NDP TEXT)

this is a very simple comes from a very simple optimal control problem continues time optimal control problem

这是一个非常简单的连续时间最优控制问题

you have a differential equation that's the simplest that you can think of

你有一个你能想到的最简单的微分方程

X dot equals u

X的导数等于u

okay X is the state is a scalar

x是系统状态，是一个标量

and it evolves according to this equation U is the control

u是控制

and one way to address this this problem is to discretize the time okay so we've talked about the screens ation

一种处理这个问题的方法是把它离散化就像我们之前讲过的那样

so let's discretize I have a typo here this should not be XK it should be UK okay I'm sorry

okay there's a typo here instead of XK it should be UK

这个地方不应该是x\_k，应该是u\_k(就是状态转移那块，乘以delta的不应该是x\_k)，把他换成u\_k

so this is the delta discretization of this differential equation

这就是微分方程进行delta离散的结果

and within a delta discretization interval the cost is quadratic let's take a quadratic cost

在delta离散化区间内，成本是二次的，所以我们采用二次成本

Delta is the increment of time increment multiplying the value of the quadratic

Delta是时间的增量乘以二次项的值

so now let's consider policy and let me take up just a some policy it's minus 2 sub X it's a stabilizing policy

我们现在考虑策略，假设一个策略是负2x，这是一个稳定策略

it's as good as any other policy to try to evaluate

这个策略和其他策略一样容易评价

I've taken this particular policy because this example is done in more detail in the in the textbook so I've chosen these particular numbers ok

我用这个特殊的策略因为这个例子在教材中有详细的说明所以我选择了这些特殊的数字

so now if you calculate this cost function its exact cost function

如果你计算这个策略的精确成本函数

it has this form it is quadratic it involves Delta okay

他是一个包含delta的二次函数

it is it is okay it involves Delta because because Delta comes into the dynamics here

他包含delta因为delta是一个动态项

and to first order it is quadratic in the state and linear in Delta and it also has some second-order terms that are negligible

首先它在状态中是二次的并且在Delta中是线性的，并且它还具有可忽略的一些二阶项

so for small Delta is the first term here

delta比较小的时候，第一项是这样的

the Q factor associated with you if you can calculate it has this form here

如果你需要计算的话，Q值是这个样子的

now a Q factor involves at term in the state had term in another term that that that is negligible when Delta is small

现在Q值包括个状态相关的项，另一项在dalta很小的时候可以忽略不计

and the control part is this one here and is proportional to Delta

控制项在这里并且与delta成比例

so if you are at state X and consider applying u the Q factor involves a big term in the state and a small term in the control small because it's weighted by Delta

现在你在状态x想要应用控制u，Q值包括一个与状态相关的很大的项和一个与控制相关的很小的项，很小是因为这一项被delta影响

so it's a quadratic function with a little perturbation on top of it

所以它是一个有一点扰动的二次函数

now suppose you approximate this quadratic function

现在假设你要近似这个二次函数

the approximation is going to try to match itself to this first term it's going to treat this as noise deglet able noise

近似趋向于匹配第一项，然后把第二项当成噪声

least squares methods tend to mimic the course behavior of the functions approximated

and little variations are being washed out in any least squares minimization

最小二乘法倾向于模仿近似函数的过程行为，并且在任何最小二乘最小化中都会消除很少的变化

so if you try to apply TD lambda Q learning whatever width approximation then the important part of this Q factor is going to be washed up completely

所以你在尝试TD lambda Q-learning之类的近似方法时，这个Q值的重要部分将被完全冲洗掉

for comparing policies at state X it's only this last term that's significant that part is constant

为了比较状态x的策略，只有最后一项才是有意义的，这部分是一个常数

it's only the part that depends on u and Delta dependent

这部分依赖于u和delta

so you may be approximating something that's big and the important part may be a small variation on that but the approximation does not capture that at all

所以你可能近似了一个非常大的数值，但是重要的部分变化很小所以近似完全没有发现这点

people who work in approximate in adaptive dynamic programming in approximate approximate dynamic programming with continuous-time systems are well aware of this problem

自适应动态规划领域的研究者在处理连续时间系统的时候也知道这个问题

and they do not use to not use the standard methods in the same way that people who work on artificial intelligence - and the reason is precisely this

所以他们不使用与人工智能领域研究者们相同的标准方法，原因就是这个

you should be approximating in some context differences of cost ago values and not cost to go values

在一些场景中，你需要近似cost to go 的差值而不是cost to go 的值本身

there's some questions on this or the previous part okay it's late I'm sorry that this has been sort of a little bit rough session with all the difficulties in getting the projector to work and all that and we'll meet again on Monday and we'll go into more specifics [Applause]