okay can you hear me everything okay okay good so I'd like to welcome you to this second lecture second out of six on approximate hoops approximate dynamic programming

LECTURE OUTLINE

last time we talked about exact dynamic programming first finite horizon problems over n stages and then infinite horizon problems over an infinite number of stages and we focused on discounted problems

上一次我们讨论了精确的动态规划首先是n 个阶段的有限期问题, 然后是无限期问题, 主要介绍的是折扣问题，也关注了无限期折扣问题，然后介绍了一些简洁的符号

and we are going to review this discounted infinite horizon formulation and I introduced some shorthand notation also and then after we do this review we're going to focus on algorithms for exact discounted DP value iteration various forms of policy direction one particular type of it of policy direction which I call optimistic which is going to come up again in the future

接下来我们要讲的是折扣无限期问题，我会介绍他的速记符号形式的公式，然后要姐你改一下具体的折扣动态规划值迭代和不同形式的策略迭代与策略迭代中的一种短视策略

then I'm going to talk about Q factors in Q learning and introduction to these but we'll revisit them later

然后我们要讨论Q-learning，不过后面的课程我们还会讲到它

as well then we're going to talk about various models other than discounted in this short course we're going to focus almost exclusively on discounted problems which are the easiest class of problems they give you the strongest results not other models may not have a strong results but we discounted problems provide a benchmark of the best that you can achieve and it's a good starting point for us and we're going to follow that but we are going to introduce some other disk dynamic programming problems including some that are continuous space in continuous time

然后, 我们将谈论更多不同于折扣问题的模型，在这个课程中，我们会把主要注意力放在折扣问题上，这是最简单的那一类问题，因为它可以给你一个比较好的结果，通过对他的研究你可以得到最好的benchmark，这是我们开始研究的最好的问题，然后我们会介绍一下连续决策空间与连续状态空间与连续决策时间的问题。

then we are going to close this overview of exact dynamic programming with a couple of more advanced topics the first has to do the more abstract view of dynamic programming that allow us to allows us to develop the theory simultaneously in a unified way for all these models not just for discounted but for all of them in look in a more fundamental way as to whether as to why some results hold for some models and not for others

然后我们不在讨论具体的动态规划，而是从抽象的角度讨论动态规划，首先使用抽象角度对动态规划做一个综述，这让我们可以为动态规划开发一个统一的理论，而不是只能应用在折扣问题中，并且解是为什么一些结果只能应用在一部分模型上，另一些模型会失效

the last topic I want to touch upon briefly is a synchronous algorithms where you either have a parallel computing system executing value or policy duration or we have a simulation driven algorithm where not all states are treated at the same time it's a certain a synchronous a synchronous character to these algorithms and it turns out that they work both policy duration and value iteration when appropriately implemented they work in a synchronous format so that's where we're going

最后一个话题是，异步算法，如果你有一个并行机可以执行值迭代、策略迭代或者基于仿真的算法，所有的状态可以不同时被处理，那么算法的特点就是可以异步进行，而且不管是值迭代还是策略迭代都能够被证明异步算法可以生效

DISCOUNTED PROBLEMS/BOUNDED COST

and we're going to start with a review remember in infinite horizon in an inner equation setting we have a stationary system which gives you the next state at time k plus 1 given the current state the control that you apply in the random quantity which we call disturbance or noise whatever you want to call it it operates starting from 0 going to 1 2 and so on an infinite number of stages at each stage there is a cost that is incurred that depends on state control in this disturbance

我们首先回忆一下无限期问题，现在有一个平稳系统，他可以根据你在系统状态下执行的控制和随机扰动的影响给出新的系统状态，从阶段0开始，阶段1，阶段2，一直持续下去，阶段数量趋于无穷，每阶段的成本根据状态，控制与扰动产生

now we are looking at policies policies are sequences of functions one function for each time stage and this function Maps States into controls so we observe the state and the control function is a rule that tells us what control to apply so we are looking for sequences of such feedback control functions this museum new one and so on

现在我们来看一下策略，策略是一个序列的函数，每一个函数对应一个阶段，这些函数是状态到控制的映射，我们观测到状态的时候，策略会告诉我们应该采取什么控制，所以我们要做的是寻找这些反馈控制函数，即控制策略

now once we plug in such a policy into the cost function which by the way is discounted by alpha which is less than one in a discounted problem then we get a random number all this discounted costs added up over an infinite number of stages give you a random number random because they depend on this w here we take the expected value over this w and the limit as n goes to infinity and we get a number which is the cost of policy pi starting at the initial state x so for one state I have one cost for another state I have a different cost and so on so this J pi is a function of the state into the real numbers

如果我们在一个alpha小于1的折扣问题成本函数中使用某个策略，把所有阶段(阶段数是正无穷)的折扣成本累加起来可以获得一个随机数，因为成本函数中有一个参数w是随机变量，所以总成本是随机数，然后计算成本在阶段数量趋于无穷时的期望值就可以获得策略在初始状态x0时的总成本，如果我换一个初始状态，总成本会是另一个数，但他们总是一个实数

now one assumption that's critical for these discounted problems in addition to the fact that alpha is less than 1 is that the cost per stage is bounded in other words there is a number M that provides an upper bound to the absolute value of the cost for any X U and W the effect of this the mathematical effect of this is that it makes this series here summable summable to a real number ok because G's bonded alpha K is geometrically going to and the whole series is converging to a real number so J pi is a real valued function

有一个对于这些折扣问题非常重要的假设，就是折扣因子小于1，这样可以让每个阶段的成本是有界的，换句话说，上界M提供了一个关于状态x，控制u和随机变量w的成本的绝对值的上界，这在数学上是有影响的，它让总成本是一个实数，同时让g被alpha的k次方限制住并且成几何级数收敛到0，总成本也可以收敛到一个实数，保证了价值函数J\_pi是一个实函数

ok because there are some other implications which I'm going to get into that for this assumption of boundedness but let me first remind you of the shorthand notation that I introduced for dynamic programming mappings

这个假设的有界性还会有其他影响，但是我想先提醒你我之前介绍的动态规划映射的速记符号

now we are dealing here with functions of state and the dynamic programming algorithm both for finite horizon and also it's infinite horizon version takes functions and produces other functions now these other functions are denoted by T J so T is a box some abstract box that takes a function and produces another function through this operation

现在我们来处理状态函数和有限期问题与无限期问题的动态规划算法现在这些函数都统一使用TJ表示，T是一个抽象的盒子，能够作用在一个函数上产生另一个函数

now let me review this operation for you it involves at state X the first-stage cost and the cost from the next state onward as measured by the function J that we are operating on this expected value is a number minimize over all u and that gives you the value of TJ's of X it is the optimal cost for a one-stage problem involving this one stage cost plus a cost to go a final cost at the end of the stage

我们来回忆一下这个操作，它包括状态x开始时第一个阶段的成本，和第二阶段直到终止状态的成本的总和，因为有随机变量w，所以总成本是一个期望值，这个总成本通过函数J进行度量，J是在所有控制u下最小化成本函数的期望值，TJ(x)是当前成本加上后续成本的最小总成本

okay now this is the mapping this is dynamic programming dynamic programming starts with some J produces TJ t square J the square J is T applied on TJ t cube J which is T applied on T squared J and so on and so clearly it's a very important operation and it turns out to be very fundamental as well so that's no notation and it's much better to work with this expression than with that expression can get pretty pretty complicated

动态规划是一个映射，从J出发产生TJ，T^2J，T^2J表示T作用于TJ上，然后再计算T^3J，就这么一直计算下去，直到算法收敛，这是一个很重要也很基础的操作，使用速记符号表达这个操作要比详细的产开符号表达要好得多

okay now suppose that you have a fixed policy new and let's say the policy stationary stationary points are very important here stationary policies are pies for which mu does not change the same new at time sine U at time and so on so a stationary policies new Miu Miu Miu ok and if you plug it in here for a given J it gives you another function team use of J this is the one stage cost of applying mu at the first stage and then getting a terminal cost associated with J so these two are similar but this is for a single policy and this is for all policies the minimum over all policies okay so this is the shorthand notation and a nice thing about the shorthand notation it can give it it gives you the possibility to describe the entire theory of this in a couple of slides

假设有一个平稳策略mu，平稳策略指的是在每一期都是用同样的策略mu，如果你把这个策略放到成本函数中，你可以获得T\_mu J这是关于J的总成本，T\_mu与T相似，但是T是在所有策略中寻找最优策略，而T\_mu是计算给定策略的成本与T\_mu，速记符号可以同时用于这两种概念，并且可以用很少的篇幅把他们描述清楚

“SHORTHAND” THEORY – A SUMMARY

so this is the shorthand theory Bellman's equation J star the optimal cost function that we're looking for satisfies this equation J star is equal to t J star it says that J star is a fixed point of the mapping T more analytically it's this equation here this equation is one equation for each state okay so this is a functional equation an equation satisfied by the whole function J star ok so so if you have if X takes a finite number of values I'm sorry yeah if there are only a finite number of states 1 up to n there are n equations here J starts of 1 J star of 2 and so on so this becomes a vector equation but generally it's a functional equation in this equation is his for J star and there's another equation for J mu and we showed last time that J star is the unique solution of this equation where is J mu for any stationary mu is the unique solution of this equation that's the first fundamental result

这就是bellman方程的速记符号表达形式，J\*事满足bellman方程的最优成本函数，J\*=TJ\*，J\*是映射T的不动点。给出更多的分析，每一个状态都对应一个方程，这些函数方程，每一个都满足所有函数，所以如果状态x在有限集合内从1到n取值，bellman方程就有n个方程，最优成本就是J\_1\*，J\_2\*一直到J\_n\*，这就是个向量方程，一般地，这些方程是一个函数方程。另一种形式不是J\*，而是J\_mu，对于策略mu，J\_mu也是这个方程组的唯一不动点，这是一个基本的理论，不依赖于策略是什么

the second result is that if I give you J star and you minimize in this expression here and get a policy in this way then this policy is optimal it's an optimality condition find J star and then minimize in this expression or equivalently find mu such that team U and J star equals TJ star if you find such a mutant is optimal conversely if you have an optimal mu then it must satisfy this equation this is an if and only if condition

第二个结论是如果你知道J\*的值，你可以通过最小化下面这个表达式获得策略，这时候策略是最优策略，这就是最优条件，换一种说法，如果你能找到一个策略mu，让T\_mu J\*=TJ\*，这个mu也是最优策略。如果你想判断一个策略是不是最优策略，你需要看它是不是满足这个最优性条件，当且仅当这个条件满足时，它才是最优策略

the third major result is about the method of value iteration value iteration starts with J any J that is bounded generates TJ by applying this mapping here with the T mapping then generates this T squared J T cube J and so on in the limit value iteration has the property that it converges to the optimal value function some major algorithm for calculating J star and of course we want to calculate J star because this will also give us optimal policies so we prove this convergence last time or almost proved it

第三个主要的结论是关于值迭代算法的，值迭代算法从一个任意生成的有上界的J开始，使用映射T计算TJ的值，然后计算T^2J，T^3J，一直这么算下去直到收敛，有证明表明值迭代一定会收敛到最优值函数，这个算法能给我们最优值函数，也能得到最优策略，我会用它来计算J\*，而收敛性上次课我们已经证明过了，这里就不再讲了

and there's a second major algorithm policy direction which we described but we did not show its validity last time

这是第二种主要的算法，策略迭代，我介绍过他但是没有证明他的收敛性

in policy direction instead of generating cost functions JT JT squared J and so on we generate policies we start with some policy nu 0 and then generate another policy mu one another policy me 2 and so on so

在策略迭代中，我们没有生成成本函数然后计算J，TJ，T^2J，T^3J… 而是生成策略，从mu0开始，不断迭代生成mu1，mu2。。。

here's the typically duration of policy duration given a stationary policy mu J we find the cost of that policy and this cost is obtained as a fixed point of the mapping T\_mu this is called policy evaluation

这是策略迭代的经典迭代过程，给定一个平稳策略mu，我们找到这个映射T\_mu的不动点J\_mu，这个计算J\_mu的过程叫做策略评价

so we solve this functional equation here for MU for a given u K find J mu K and then we find mu k plus 1 that satisfies this equation in other words attains the minimum in this expression when J star is replaced by J mu J this is called policy improvement

然后我们求解下面这个方程，找到新策略mu\_{k+1}，换句话就是最小化这个表达式，然后使用计算得到的J\*代替J\_mu，这个过程被叫做策略改进

now policy duration value duration is going to be very important for us approximate versions of those are the and so on of approximate dynamic programming so we're going to get back to them quite frequently and but for the moment we have been looking at V at the exact versions we showed convergence of this and a few slides were going to look into this and show convergence of that as well

对于近似动态规划，策略迭代和值迭代是两个非常重要的算法，我们会非常频繁地回顾这两个算法，接下来我会用几个slide来证明他们的收敛性

okay now what are the major properties that make the theory work what is at the foundations of all the proofs and arguments and justifications it turns out that the fundamental properties are embedded in this mappings T and T mu

现在我要讲一些让这套理论工作的主要属性，包括所有的证明，理论和分析的基础，这些基础性质是存在于映射T和T\_mu中的

the first major property is that both T and T mu if I can take you back here are monotone

第一个主要的性质同时存在于T和T\_mu中，我把slide向上翻，就是单调性

if I increase J by a certain amount uniformly from J I take it up by some amount then TJ is also going to increase okay it's evident because it's a plus sign here similarly for T\_mu you it is monotone so that's that's a property that's very very basic and very important it's very simple but still very important

如果我使用一个均匀的，确定的数值加到J上，可以看到TJ的数值同样会增加，因为这里是一个加号，单调性对于T\_mu同样生效，这是一个非常基础也非常重要的性质

MAJOR PROPERTIES

for any function J and J Prime on the state space one bigger than the other then if you apply T to them to both of them being equality is preserved and similarly for T\_mu

对于状态空间中任意的J和J’，一个大于另一个，你在使用映射T时还是维持原关系不变，相同的，T\_mu也有这样的性质

now just about all dynamic programming models have this monotonicity property it's it's part and parcel structure of these problems

所有的动态规划模型都具有单调性 这是一个不可缺少的性质

however for discounted problems there's another very important property that is responsible for the good nature good character of these problems what makes discounted problems be relatively easy and having a strong theory

对于折扣问题，有能够给提供很好的性质和结构，让折扣问题很容易求解并且有很强的理论支的性质，

is that in addition to monotonicity there is a contraction property lurking in which we have already discussed

就是我们已经讨论过的收缩性

and it says that for any bounded functions J and prime then the distance between jet between J and J prime as measured by this expression here by the maximum distance between the two this distance is reduced when you apply T to these two functions applying T to J into J Prime brings them closer in this sense by a factor alpha alpha remember is a discount factor less than 1 similarly there's another property similar for MU of per team you square this

对于任意有界函数J和J’，他们之间的距离可以被这个表达式计算( |J(x)-J‘(x)| )，在你使用算子T对J和J’进行映射之后，他们的距离的最大值会变小，就是你看到的那个第一个表达式，不仅对于T有这样的性质，对于T\_mu也有这样的性质

and now if you want to translate it in even more compact notation introduce a norm introduce a norm on the space of functions J which is the maximum magnitude okay

如果你想要用更简洁的符号表达这些东西，我要介绍一下norm，函数空间J的norm指的是这个函数的最大值

so you have a function take the maximum magnitude and that's the norm of the function and what in terms of this norm this properties here translate are equivalent to in shorthand to this that the norm between TJ and TJ prime is less than alpha times the north between J and J Prime

然后你就可以看到，上面的两个不等式可以被简写成这两个

in mathematical terms this says that T is a contraction mapping on the space of functions J and similarly T\_mu use a contraction mapping now contraction mappings have great properties and that's why I discarded problems also have great properties

在数学上，T叫做函数空间J的压缩映射，相似地，也有压缩映射T\_mu，压缩映射有一些很好的性质，所以我要讨论的折扣问题也具有这些性质

THE TWO MAIN ALGORITHMS: VI AND PI

okay all of this is review up to now and let's go into the algorithms one is value iteration we discussed that it works for any starting function J

我们要回顾一下之前讲的算法，第一个是值迭代算法，它对于任意初始价值函数都是有效的·，都可以收敛到最优价值

in his policy duration given mu K evaluate mu K in other words find its cost function by solving the corresponding functional equation the bellman equation corresponding to mu K in shorthand it is like this in longhand it is like this okay then after you calculate this cost function of mu J you improve muche by finding another policy mu k plus 1 by minimizing in this expression so notice you have already calculated Jamie okay

策略迭代中，给定mu\_k后，计算mu\_k的总成本，也就是通过求解下面的bellman方程来计算策略mu的总成本，计算这个成本的复杂表达式和简洁表达式都在屏幕上了。在你算出这个成本函数后，你可以通过找一个满足下面表达式的策略mu\_{k+1}(即根据当前价值函数的值最小化总成本，找到相应的控制)来改进当前的策略，

you plug it in here and for every state X you do this minimization and you obtain a minimal a controller that is the minimum and that's your new policy you can write this expression in longhand or in shorthand it's like this as you can verify so that's policy direction

然后再把这个策略放到成本函数里，对每一个状态x做一次最小化成本函数获得一个相应的控制，这个最小化的控制器就是你得到的新的策略复杂形势和简单形式的表达都在这里了，你可以验证，这就是策略迭代

and now before we go into justifying this policy direction let's make an observation

在证明策略迭代之前，让我们先观察一下

suppose that you have n states okay not n dimensional system and states 1 2 up to n so there n costs associate will J\_mu J\_mu is a vector J\_mu 1 at state 1 2 and so on up to n so this J\_mu is a vector when you evaluate J mu using this equation you need to solve a linear system of equations there is one equation here for each X so there are n equations with n unknowns which are the components of this J mu K and this is a linear equation

假设你有n的状态，不是n维的状态，就会有n个成本函数J\_mu，J\_mu是一个向量函数，包括J\_mu^1，J\_mu^2，一直到J\_mu^n，然后你需要使用这个方程计算J\_mu的值，也就是说你需要解一个线性方程组，每一个方程都对应一个x，所以现在你需要处理的是一个有n个方程和n个未知数的线性方程组。

why is it linear well this is a constant term here and this is a term which involves this expectation now expectation is a linear operation right – right the expectation of a random variable you just write the product the weighted product of the values of the random variable with the prop corresponding probabilities

我说他是线性的原因是，每一个方程都是一个关于期望的函数，而期望是一个线性操作，这个线性操作是用函数值乘以相应的概率再累加

so this is a linear operation over here so you have n linear equations with n unknowns and you can write the system compactly like so

所以这是一个线性操作，你需要解的是一个n个方程和n个未知数的方程组，

this is a vector equation involving this constant vector involving this matrix here alpha P mu and P mu is the so-called transition probability matrix associated with mu let's not go into the details of that it is the matrix that comes out from this equation so you have a vector J nu here which is the unknown another vector which is the unknown and this is a linear equation a matrix equation that you can solve by inversion okay

这是一个向量方程，包括常数向量(g)、矩阵（alpha\*P\_mu），P\_mu被叫做关于mu的状态转移概率矩阵，我们不会深入细节，我只能说这个矩阵是来源于方程组的，所以你现在可以通过求逆的方式获得一个J\_mu向量

这是一个向量方程, 涉及这个常数向量涉及这个矩阵这里α p 亩和 P 亩是所谓的过渡概率矩阵与亩, 让我们不要进入细节, 它是从这个等式出来的矩阵, 所以你有一个 vector 这里是未知的另一个向量, 这是未知的, 这是一个线性方程, 你可以通过反演解决好的矩阵方程

you solve the code you invert the corresponding matrix and you get the solution in this way so mathematically speaking this is a very simple operation the policy evaluation is a very simple operation

你计算出这个矩阵的逆，然后用这种方式获得了解，在数学上这是一种非常简单的操作，策略评价也是一种很简单的操作

in practice however it's not so simple and the reason is that if you have a large number of states let alone if you have infinite number of states even if you have a finite number of states but if n is large and I'm talking really large here zillions and trillions and a huge number think of the hugest number that she can think more than the number of of molecules in the entire universe a gigantic number then it is impossible to solve this equation by giving it to MATLAB

实践中这并不简单，先不谈无限期问题，即使你面对的是一个很大的状态空间的有限期问题，状态数量n非常大，比如数以万亿计或无数个状态，你可以认为这个数比宇宙中所有分子的数量都多，这种问题根本没没有办法使用matlab解决

okay you can't do that the memory requirements are too big the computational requirements are too big so even though we are going to show shortly that this policy duration terminates in a finite number of steps applying it is out of the question applying it exactly you can't you can't apply it in practice you have to do some kind of an approximation

这种规模的问题需要的计算资源与存储资源实在是太多了，你没法满足这两个需求，如果策略迭代算法在有限步的迭代后就终止了，那么这种方法毫无疑问是可以用用的，但是实际问题中你根本没有办法使用策略迭代，必须做一些近似才能使用策略迭代

now the type of approximation that you we will use often and I'm going to get into that later

我过一会会讲我们经常用的近似方法的类型

is to solve this equation here iteratively not exactly by matrix inversion but iteratively by using value iteration use value iteration to solve this equation approximately with just a few iterations okay so instead of solving exactly solve it approximately in there are many many kind of approximations we can use including solution by value iteration okay

为了求解这个方程组，通过矩阵求逆求精确解已经是不可能的了，所以要使用迭代的方法去求他的解，比如值迭代，使用值迭代去求解这个方程组的近似解只需要很少次的迭代，所以有很多近似方法，比如值迭代可以取代求精确解的算法

so now what we would like to do is take a look at why this policy Direction method works it turns out that the most important thing about it is that there is policy improvement in other words you have the current policy new K you have its cost the cost function of the next policy is better is smaller so it uniformly decreases the cost function with every iteration

所以现在我们需要做的就是看一下为什么策略迭代方法能够起作用，最重要的就是策略改进，换句话说，你根据当前的策略mu\_k计算出了相应的成本函数，最小化策略改进函数可以获得更好/更小(极小化问题时)的策略，至少策略不会恶化，所以每一次迭代都可以让策略均匀地得到改善

JUSTIFICATION OF POLICY ITERATION

and the proof of this is a few lines which I'll give you here what we want to show is that with every direction I get a cost function that's less than what I had before a better policy now I could get the two of them to be equal but in that case I will have an optimal policy so I will show that either you have strict improvement at every direction or you have optimal 'ti

它的证明会用几行，我想要给你看的就是每一次迭代获得的成本函数都小于我之前的策略的成本函数，如果这两个值相等，那么现在的这个策略就是最优策略了，所以每一次迭代，你要么对策略进行改进，要么发现你已经得到了最优策略

and here's the proof for a given equation

这是这个表达式的证明

suppose we are at the direction K then we know that we that but that that the policy evaluation equation holds J mu J satisfies this equation

假设程序迭代到第k次，现在我们知道策略评价的结果，J\_mu满足这个表达式(J\_{mu^k}= T\_{mu^k} J\_{mu^k})

when I minimize this expression here in involved in key against J mu J I get something that's smaller and by policy improvement I have that this is equal to that okay so just from the definition of the method I have that when you apply T mu K + 1 to j nu K it takes it down

我最小化这个表达式(T^N\_{mu^{k+1}} J\_{mu^k})就可以得到一个更好的策略，策略改进之后

从这些的定义我可以得到T\_{mu^{k+1}} J\_{mu^k}会变得更小

so you start with general K applying P mu K + 1 I get something that smaller apply it again and again get something smaller and you get convergence to something so that's due to the monotonicity property J mu K is less than this here apply mu K plus 1 again and then again and you get a monotonically decreasing sequence and in the limit you get by value iteration you get J mu k plus 1 so that's that J mu K is greater than J near k plus 1 it's two lines proof okay

所以从一个迭代次数k开始使用T mu k+1，我可以获得更好的策略，然后这么不断地重复下去，因为单调性的存在，可以一直获得更小的成本直到收敛，也就是说J mu k 小于J mu K+1 ，J mu K+1 小于J mu k+2，获得一个单调递减序列，这个单调递减序列的极限可以使用值迭代算法计算出来这就是证明这个结论的两种方法

now if the two are equal then equality holds throughout here which which among other shows that J nu K sauce balance equations the T equation because it's a teeth some pony yeah have equality here therefore J mu K is equal to J star

这两种证明是等价的，J mu k等于J mu k+1，后面所有的不等号就都变成等号了，J mu k也就是bellman方程的解，因此J mu k也就等于J\*

so in the end I did iteration K I've every algorithm generates a strictly improved policy strict inequality in at least one state or else it finds an optimal policy

第k次迭代任何一个算法都能够严格地改进策略，获得严格地不等式关系，否则这个算法就已经找到最优策略了

now this is true for an arbitrary state space

这个结论对于任意状态空间都是成立的

if you have a finite spaces MDP or a finite state MVP then the algorithm terminates with an optimal policy

如果你有一个有限空间的MDP或者有限状态的MDP，算法都可以终止于最优策略

why is that going to put up for a finite space MDP finite number of states final number of controls at each state how many policies are there altogether the finite number finite number of policies at every direction I have strict improvement or else optimality therefore termination in a finite number of steps with an optimal policy

原因是一个有限空间的MDP有有限个状态和有限个控制，这样就有有限个策略，每次迭代时都能够严格地改进当前策略，也就是说算法最终会经过有限步之后收敛于最优策略（最坏的情况是遍历策略空间）

if you have an infinite state space then you can show convergence asymptotic in an infinite number of iterations that's a little trickier to show but it is true okay

如果你有一个无限状态空间的问题，你可以看到他会在无限次迭代中渐进收敛，这很难以理解但确实是事实

so this policy duration method has very solid properties for the discounted problem

所以对于折扣问题，策略迭代方法有很多很好用的性质

okay now there is we mentioned value duration as one algorithm and then policy direction and the fact that sometimes it's better to use a few directions of a few value iterations to evaluate a policy rather than to evaluate it exactly because we may have a very large number of states this is called optimistic policy iteration it is policy duration where the policy evaluation is done approximately with a few directions with a few value iterations

现在我们已经讲过了值迭代和策略迭代，实际上用少量迭代来评价策略要好于计算策略的精确价值函数，因为我们可能由于状态过多无法计算精确值，这种少量值迭代进行策略评价的方法被叫做乐观策略迭代(optimistic policy iteration)，也就是策略评价的时候使用值迭代算出来的近似值来评价策略

OPTIMISTIC POLICY ITERATION

so here it is optimistic policy duration this is policy duration where the policy evaluation is done approximately the finite number of valid directions so we approximate J nu with a few it value directions involving mu starting from any J M can be one in which case we turns out that we get value iteration algorithm or it could be two three any number M equals infinity is exact policy direction

乐观策略迭代是策略评价时使用mu下少量次数的值迭代近似策略价值，对于任意J，M可以是1，可以是2，可以是3，当M的值是无穷的时候，乐观策略迭代就变成精确策略迭代了

so this optimistic is is this optimistic method is intermediate between the two extremes value and policy duration and in practice it works better than both

所以optimistic 策略迭代是一种处于值迭代与策略迭代中间又比这两者都好的算法

and experimentally this has been verified the some analysis that indicates that and you can think of extreme cases where it would be better than either one it's not difficult to do that there's a shorthand notation for this for this method since I'm big on shorthand notations I have it here and you can see what it is this is the policy improvement in aeration and this is the policy evaluation direction involving Mk number of value iterations and for MK ungentle equal to one you can verify that this exactly value duration for MK infinity it becomes policy direction

这里面的一些分析已经被实验验证了，你可以想到的极端情况下，它的表现都比这两者好，这并不难验证，这里有一些间接表达式，你可以在slide上看到，这个是策略改进，这个是策略评价，到底是策略迭代还是值迭代依赖于M的值，当M等于1时，这个算法是值迭代，当M的值时无穷时，这个算法时策略迭代

you can show convergence for both finite and infinite space discounted problems in the case of in both cases both in fine and in play if you can shoot an infinite number of directions you need an infinite number of directions because for the special case where M K is equal to 1 it becomes very duration and value iteration requires an infinite number of iterations typically it works faster than badly duration in policy duration particularly for large problems when we are going to talk about approximations later

无论是有限空间还是无限空间的折扣问题，如果你进行无限次数的迭代，都可以看到策略迭代的收敛性，因为特殊情况下，策略迭代和值迭代都需要无穷次迭代，一般策略迭代的工作速度是最快的，特别是对于大规模问题，我们后续讨论近似的时候还要谈到

an approximate policy duration it's always some kind of optimistic method that you are using we never use the two extremes what very seldom we use the extreme of value duration and never exact policy duration in approximate dynamic programming

近似策略迭代一般是optimistic方法的基础上发展来的，我们从来都不会使用精确的值迭代和策略迭代

ok let's say one more thing about policy duration with an eye to the future starting in the next lecture we're going to be talking about approximations and we're going to be talking about approximate versions of this policy duration algorithm

关于策略迭代，我要再说一点内容，下一次可我要讲一些近似方法，从近似的角度来进行策略迭代

APPROXIMATE PI

so let's have a preview of that and what's the basis for that ok

我们来做一个预习，告诉你们一些基础内容

so well then we have approximations they can be approximations in two ways

我们想要做近似的时候，有两种方法

the policy evaluation may be approximate in the sense that instead of computing the cost of a policy

策略评价使用近似方法计算策略的成本

we compute some other function JK which is uniformly within Delta this is the maximum norm so the maximum distance by value distance between JK and J mu K is bound by Delta

在我们计算函数J\_k的时候，可以知道他的值被delta限制，即J\_k和J\_{mu^k}这两个函数的距离不大于delta

so I have this J nu K that I want to evaluate but instead I using something that's within Delta uniformly from that Delta is some number whatever it is okay so this is approximate policy evaluation

所以现在我想计算J\_{mu^k}的值，我可以通过计算一个与delta有关的数值来代替，这就是近似策略评价

and suppose also that the policy improvements approximate so when I take the minimum in in in in the minimization operation corresponding to JK that I have that I don't find the exact minimum but rather something that's within epsilon of the minimum so policy improvement is exact within Epsilon okay

近似策略改进是这样的，当我最小化这个与J\_k相关的表达式，我不去找精确的最小值，而是找一个满足这个不等式的值就可以了，这就是近似的epsilon策略改善

so both of these approximations of course throw off the convergence you can't expect convergence to be exact optimum

当然了这两个近似可以收敛，不过你不能期望他们能收敛到精确的全局最优

but what you have is a bound the bound says that the sequence of policies generated by this approximate policy duration method satisfies this inequality

这个收敛之后的值，也是有上界的，上界表明近似策略迭代过程中生成的所有策略都满足这个不等式关系

so these policies do not converge to anything that's optimal but come to within this factor this number this bound from the optimal okay

所以这些策略不会收敛到全局最优，但是可以收敛到这些因素作用下的最优解

so the optimal is somewhere over here and I get something that's higher than that but by no more than this

最优解在这个位置，我获得了比当前解更高的解，但是他没有超过最优解

now epsilon is what you'd be approximation involved in the policy improvement

epsilon是参与近似策略改进的因素

Delta is the approximation the error involved in the policy evaluation

Delta对近似策略评估的误差有影响

and there is a factor 1 minus alpha squared in the denominator alpha is the discount factor this factor in the denominator worrisome because alpha may be something like okay point nine nine okay very close to one and this denominator is going to cause this bound to be very high

分母是(1-alpha)的平方，alpha是折扣因子，这个折扣因子会对算法造成比较大的影响，因为如果alpha的值是0.9的话，因为他很接近一，分母就会非常小导致算法误差非常高

it turns out that you can find specialized examples where this inequality is sharp okay however typically in practice this bound is conservative

事实证明你可以找到让这个不等式相差幅度很大的特殊的例子，但是一般来说，实践表明这个上界很保守

and what you get is you start out typical behavior in practice you start out with some initial policy and you have some cost functions like like that then you apply one step of approximate policy direction you get something that's lower and lower and lower up until you get to within some zone from the optimum you get in there you sort of bounce back okay

在实践中你获得了一些初始策略和这些策略的成本，然后你多次使用近似策略迭代，可以看到成本越来越低直到在某些最优解附近的趋于成本开始反弹

the method does not converge it just oscillates within this zone here that's the typical behavior in practice and this zone is not as high it's not as big as this bound suggests it is something typically smaller

这个方法没有收敛，他的成本只是在这个区域内震荡，这是一种很典型的实际问题的成本变化状况，这个震荡区域没有那么大，它是由bound决定的，甚至比bound还要小

there's another bound that I want to mention if this sequence of policies terminates at some policy mu bar in other words you get to some point and you keep generating the same policy it's not going to be optimal it's going to be optimal within some error bound and this error bound is this here and it is better by a factor of 1 minus alpha than this other bound

我还想介绍一下另外一个界，策略迭代可以获得一个策略序列，如果一个策略迭代过程获得了mu bar，那么策略迭代也就终止了，换种说法，如果你发现一直在产生相同的策略，那么这可能不是最优的策略，她会有一些误差，但他一定比1-alpha决定的其他策略要好

there are some approximate policy duration methods for which you can guarantee termination and have this better error bound there are some others where you cannot guarantee termination you get oscillation generically and then and then this is this bound here that that is in effect

有一些近似策略迭代你可以保证迭代终止并且可以保证能够获得更好的误差，有一些方法你没法保证迭代终止，甚至不能保证震荡终止

now you should not take this bounce too seriously okay they are not quantitative bounds that you can use

不要把这些界看的太重要，他们不是你使用这些方法的界限

you can't first of all you cannot guarantee that you get when you do approximate evaluation who knows what this Delta is you can't get a hold on on numerical value of Delta similarly you cannot get a hold of you numerical value of epsilon and less epsilon is equal to zero okay

你不能保证在你做近似策略评估的时候就知道delta的值是多少，你不能一直用一个delta，需要尝试什么值最合适，同样，sesilon的值也需要调整，最小的epsilon等于0

and then so since Delta is unknown you can't really say much about that's quantitative using this bounce all you can say is that at least the method does not explode does not go unstable its stable within some bounds okay

因为delta的值为止，所以你没法说这个值用多少能够保证结果能够在一定范围内震荡，你只能保证这种方法至少不会让结果变得非常大

it's not going to explode on you which is not very much but it's something it's not quite negligible okay because there are methods that do approximations and then they explode as as the overtime okay

它不会变得很大但并不是没有影响，因为有些方法近似的时候这个值会随着迭代时间变得非常大

so that's a good point to take a little break are there any questions about policy direction we have done policy direction so far and now we're going to go into Q factors and we have closed the subject for the moment of policy direction at least exact

现在我们要休息一会，你们有什么关于策略迭代的问题就可以问了，接下来我们要讲Q函数了

Q&A

and have any questions yes yeah your question has to do with this optimistic policy direction okay I'm telling you that it's typically works faster than value duration and policy duration but can you prove it the answer is that there are some partial results that that that you can interpret them as faster convergence the most convincing evidence is really the practical evidence there is the theoretical results are not so definitive there are some but they are not definitive it's mostly based on practice but it is a sure thing okay you can have no doubt about it yes okay equals MK is equal an integer or greater yeah positive I'm not sure I understand what you mean should be positive okay okay no I don't think you're right okay your question is the following suppose that M K is equal to infinity then I claim that this this relation here gives you this relation gives you J nu K plus okay because you do an infinite number of it M K goes to infinity so this the sequence converges to J mu K J mu K I'm sorry J nu K so so this is exact policy evaluation and therefore when followed by policy improvement you get the policy duration I think it's correct as it is okay in the last slide the last slide here second equation it's you know it's G second it's policy yes ah there's a typo here and it should be m'kay okay you're right here's M superscript K yes you're right yeah it's a type of I'll correct it as I as was mentioned earlier this slides change up to five minutes before the lecture so sometimes there are some mistakes and even though we are posting slides before the lecture for you to take a look at at the end of the lecture I make corrections to typos and post them again okay so let's take a break for ten minutes and then we'll come back and continue