APPROXIMATE POLICY ITERATION

maybe there's a way to bring other people in

不重要，不翻译了

okay so let's continue we have been talking about the approximations to the two basic methods of discount for discounted infinite horizon problems value iteration and policy duration and we saw the approximate version of value iteration

继续上课，我们之前讨论了对无限期折扣问题的两种近似方法，值迭代和策略迭代，并且已经讲过了近似值迭代

APPROXIMATE PI

now let's go into an approximate version of policy direction

现在我们来讲一下近似策略迭代

okay so policy Direction starts with a policy evaluated then computes an improved policy and so on

策略迭代先评价策略，然后改进策略

approximate policy is either iteration simply involves an approximation at the policy evaluation stage

近似策略迭代的策略评价的近似很简单

instead of solving the corresponding evaluation equation exactly which is a linear equation of gigantic potentially gigantic size we consider an approximation to that which involves our approximation architecture in the parameter R and then we use this approximation to generate and improve policy

取代求解大规模线性方程组进行精确策略评价的是参数r下的近似结构获得的近似解，然后用这个近似解进行策略改进

so how do we evaluate a typical policy mu in this box here

那么我们是怎么对框里的策略mu进行评价的呢

we will use a linear cost function approximation involving as vector matrix of features phi phi is a full rank and by n matrix with columns which are the basis functions and the i-th is defined is denoted by Phi of i prime same notation as we had earlier

我用一个线性成本函数进行近似，这个函数包括一个特征矩阵phi，phi是一个全排列n行矩阵，每一行都是一个基函数，第i个特征向量被定义维phi(i)’，这些符号与我们之前讲的一样

and the policy improvement once I have an approximation from this box here evaluating the current policy new policy improvement is obtained from this equation and I assume that this is done exactly you could consider approximations to this minimization but we'll assume that's done exactly

策略改进是，如果我有了对当前策略mu的近似策略评价，如果你想要精确地进行策略改进，就要最小化这个表达式(policy improvement下面的那个表达式)

what we want to focus you is how to deal with this approximate policy evaluation for the moment

目前我们需要关心的就只有如何进行近似策略评价

if you do this and if the policy evaluation is approximate to within Delta okay there's an error bound that says that the successive policies come to within this expression from the optimum

如果你这样做(近似策略评价和精确策略改进)如果你基于delta进行近似策略评价，那么这个策略表现的上界就由这个表达式给出

and by the way this is exactly the same error bound as approximate value direction

顺便说一下，这个误差上界和近似值迭代是一样的

so approximate value Direction an approximate policy duration give you the same error bound even though there are different methods it just turns out this way

所以近似值迭代和近似策略评价能够产生同样的gap，即使这是两个不同的算法

so that's the overall framework

这就是整个框架了

POLICY EVALUATION

and now we want to focus on evaluate approximate policy evaluation

现在我要讲近似策略评价了

so I mentioned last time that there are two approaches one direct the direct method and the other one is the indirect method

这是我最后一次提到这两个方法了，就是直接近似和间接近似

in the direct method we have this J\_mu that we want to approximate and we project it down within the approximation subspace and we could use simulation to do that as I mentioned earlier

直接近似方法是我想要近似这个J\_mu，我可以把它投影到近似子空间里，可以使用仿真来进行，我之前讲过

this approach is legitimate but it is not very popular and are quite a few reasons for that

这种方法合情合理但是不是很受欢迎，有几个原因

one of them is that to calculate this J\_mu samples of J\_mu you need to do simulation for a given state you need to calculate the corresponding cost of the policy so you have to run the policy many times in average to obtain some an approximate version of just one sample of J\_mu and then you have to do this for many states and there's a lot of noise in this a lot of simulation mode noise involved in this process and this noise shows up when you try to use this method you find that you need an extreme extreme amount of simulation so people don't use this method very much

一个原因是想要计算J\_mu的值你需要对一个给定的状态计算给定策略的成本，所以你需要使用这个策略执行很多次仿真然后求成本的平均数来计算近似成本，这样在仿真过程中会产生很多噪声，为了避免噪声的影响你必须进行非常多次的仿真，所以人们很少使用这个方法

instead the more popular method is the indirect method whereby instead of projecting the solution you project the equation itself this is called the galerkin method

另一个常用的是间接近似，使用投影方程而不是投影解(近似值)，也就是galerkin方法

and it amounts to solving this equation

需要求解这个方程(phi r=pi T\_mu(phi r))

if there was no approximation then we would not need a projection here so what we do is we project the equation and try to find an R that satisfies this equation

如果不近似的话我们就不需要投影了，但是间接近似是对方程组投影然后求解这个方程组(phi r=pi T\_mu(phi r))得到r的值

so the solution has the property that when you apply T to it and then project down you end up at the same point that's the concept of the solution here

求得的解有这样的性质：你对phi r=pi T\_mu(phi r)使用算子T，然后把它投影到低维空间能够得到同样的点，这就是这个解的概念

so let's accept that this is a legitimate method it has error bound it's it's something that does work to a certain extent practice

我们必须接受这种方法合情合理但是还是有误差的事实，不过它在一个场景下是能够工作的

how do we implement it how do we find approximately a solution to this by simulation that's our issue

我们该怎么实现这种方法呢，如何使用仿真找到一个近似解，这就是我们想知道的

well first of all generally projection as I mentioned can be implemented by simulation in least squares so somehow we have to find a way to factor a really simulation in least squares into this equation here

首先我提到的一般性的投影可以通过在最小二乘问题中使用仿真完成，无论如何要给投影都可以在最小二乘中找到一个仿真方法进行计算

PI WITH INDIRECT POLICY EVALUATION

okay so let's focus on on this method that uses indirect policy evaluation

我们来看一下间接策略评价的使用情况

given the current policy we solve this projected equation

给定一个策略后，我们求解这个投影方程组(phi r=pi T\_mu(phi r))

and then we approximate so basically we approximate the solution of bellman equation with a solution of projected equation

然后我们就可以用这个投影方程的解来近似bellman方程的解了

KEY QUESTIONS AND RESULTS

in color questions here well first of all does the projected equation have a solution how do we know that has a solution how do we know that has a unique solution

这里有几个问题，第一个是我们怎么知道投影方程有唯一解

so under and so this is the same as asking under what condition is this mapping PI T at contraction so that PI T has a unique fixed point

这和如果映射PI T有压缩性，所以PI T有唯一不动点的问题是相同的

and remember this is the same issue that we raised earlier

这和我们之前提到的话题是相同的

under what circumstances the mapping and the projection are matched so that PI T is a contraction

在映射T和投影是(范数, norm)匹配的，所以PI T有收缩性

so now here's the magical thing we are going to determine a weighted norm under which this is a contraction

这是一个很有意思的事情，为了保证收缩性，我们得确定一个权重范数

and here is the assumption we assume that the Markov chain corresponding to the current policy gadi

现在有一个假设，这个马尔科夫链现在使用的策略mu(没听清楚)

it has a single recurrent class and no transient states

他是一个单一递归和稳定状态的马尔科夫链

so you plug in this policy Markov chain runs and it visits all states infinitely often with certain long-term frequencies

所以使用这个策略，马尔科夫链长期运行并在long-term frequencies下无数次地访问所有状态

and these long-term frequencies are we steady-state probabilities

long-term frequencies指的是稳定状态概率

this is standard Markov chain theory you have a singular recurrent class no transient States then the steady state probabilities

这是一个标准的马尔科夫链，单循环递归，稳定状态，和平稳的状态概率

okay these are occupation measures or the average probability that you will be at state J if you start at state I in the limit this is independent of where you start and it is positive

这表示xi是occupation measures或者叫平均概率，表示你从状态i开始，状态j出现的平均概率不依赖于初始状态并且是一个正数

so this steady-state distribution which is the state with the distribution involved in the long term frequencies of the various states is a positive vector okay

所以这个稳定状态分布是各种状态的长期概率分布并且是一个正数

and now we are going to use projection with respect to this factor and things will work out this is a magical norm that we are used that we are looking for

现在我们要用这样的xi进行投影，这就是我们想要的范数

here is a proposition suppose that a projection is with respect to this distribution norm okay then PI T nu is a contraction of modules alpha with respect to this norm

有一个建议，假设这是一个在这样的分布范数下的投影，PI T\_mu是一个关于这个范数的模α的收缩映射

and moreover the unique fixed point of this mapping which is what we are going to use to approximate a mu has an error bound that satisfies this relation

我们用这个映射(PI T\_mu)获得的唯一不动点来近似策略mu有一个满足这个关系(这一页最下面的那个公式)的误差上界

okay notice that what you get is not the projection of J\_mu but the distance of J nu from this approximation is bounded by a scalar multiple of the distance or J\_mu from the approximation subspace

需要注意的是你得到的不是J\_mu的投影，而是J\_mu与这个近似函数的距离，这个距离被这个标量成语J\_mu与近似子空间的距离限制了上界

always with respect to this norm

通常与范数有关

for the moment this bound is not important what's important is that I have found some sampling distribution with respect to which to do the projection so that I get a solution of this equation and the process the ND and the divergence that I experienced in the previous method does not occur

这个上界其实不重要，重要的是我找到了投影的采样分布，这个分布可以让我得到这个方程(PI T\_mu)的解，之前的方法会发散的情况就不会出现了

PRELIMINARIES: PROJECTION PROPERTIES

okay so let's say a few things about why this thing works

让我们来看看他为什么会工作

first of all let's talk about projections

首先我们来谈一下投影

an important property of projection onto a subspace spanned by the columns of a matrix phi is the tag Orient theorem okay

一个通过矩阵phi的列向子空间投影的重要性质是Orient定理

projection this vector is orthogonal to this vector here

对这个向量的投影(投影向量)与这个向量(横着的那个)正交

in the square of the this norm is equal to the sum of the squares of these two norms

这三个向量满足勾股定理

okay this is a an abstract extensions of the classical Pythagorean theorem for triangles for orthogonal triangles

这是直角三角形的经典勾股定理的抽象扩展。

and the orthogonality of courses with respect to the xi here

当然这个正交性是与xi相关的

and here's the second important fact for us projection is nonexpensive

第二个重要的性质是投影是nonexpensive的

if you project two vectors J and J bar with respect to a projection a weighted Euclidean projection norm then the distance of the projection is smaller or at least no larger than the distance of the vectors

如果你对向量J和J bar进行加权欧几里得范数投影，他们的距离小于，或者至少不大于原始向量J和J bar的距离

and the proof of this is one line

证明只有一行

the difference between the projections which is this because pi is linear is less than that plus some positive thing and by by the projection theorem by the Gordian theorem this sum here is equal to J minus J bar

投影的差距由于PI是线性的所以小于第一项加上某个正数，又由于Gordian 定理，他们的和等于J-J bar

let's not focus on this algebra just suffice it to say that's a one-line proof based on the Pythagorean theorem

我们不要把注意力放在代数上，只要说这是一个基于勾股定理的单行证明就足够了。

PROOF OF CONTRACTION PROPERTY

now here is the important theorem and applies to any Markov chain but these are body

这是一个对任何马尔科夫链都成立的重要定理

suppose that you have a Markov chain that corresponds let's say the one that corresponds to a policy mu

假设有一个使用策略mu的马尔科夫链

and P is its transition probability matrix

P是要给状态转移矩阵

and xi is the steady state probability vector corresponding to this Markov chain

xi是这个马尔科夫链的平稳状态概率向量

then you have this inequality holding

然后就有这个不等式

okay so take any vector Z apply this probability this transition probability sort of randomized according to T and we get something that's smaller a smaller norm

对于任何向量Z，使用这个状态转移概率之后都能得到更小的范数

okay the proof of this takes takes like five minutes to go through

这个证明大概需要用五分钟来讲

it involves you do the natural things take the square of that write it in terms of V of xi okay this is equal to this by definition

根据定义，小于号左边的项的平方是这样的

and then you use okay there's a there's a convexity property of a quadratic that you use here and using the steady state okay take this out here and then use the definition of steady state probabilities you get this equality and this is precisely the the the norm of Z squared

这个二次项是具有凸性的，你用这个凸性(Jenson 不等式)和平稳状态的性质等够得到小于号右边的这一项，然后根据平稳状态概率的定义可以得到等价的一项，也就是X的范数的平方

so it's a one-line proof a little tricky

这一行证明用了一些小技巧

and but it still fits in one paragraph

但是他仍然符合这个段落的内容

so that's not the time to do this detail proofs here but I want to show you that this is relatively simple

这不是详细讲这个证明的时间，但是我要告诉你这个证明相对简单

now once I have this relationship

我有了这个关系之后

then I used a non expensiveness of Phi in the definition

我使用phi的非扩张性的定义

the definition of T mu of J involves the cost vector of mu and also the transition probability matrix of mu

T\_mu J的定义包括mu的成本函数和mu的转移概率矩阵

and the difference okay I want you to see how to to look at the difference of Pi T applied on to generic vectors J and J bar

我希望你能够看到PI T应用在一般性向量J和J bar后的差分(difference)

because of this inequality here when you subtract these two this goes away and what's left is okay first we use the non expensiveness of the projection okay to get this relation in this relation is equal to this and now I use this lemma to get the last part of inequality which says that PI T is a contraction with modulus alpha

使用非扩张性可以得到这个不等式关系，然后不等号右边等于右边这项，再使用刚刚讲的定理，可以得到最右边的不等式关系，也就是说PI T是一个以alpha收缩的映射`

so that's it very simple proof to establish the right norm for projection which is the steady state distribution of the current policy

所以这个证明很简单地建立了当前策略平稳状态分布的映射的右范数

(Someone asking questions yes you use the property of their body because if Sai does not have positive components everywhere then this is not a norm anymore okay you have to have positive components oxide for it to qualify as a norm asking finished)

PROOF OF ERROR BOUND

okay so there is also this proof of the error bound and against a one-line proof which I will not go through

这是误差上界的证明，同样用一行来完成证明

the first equality here uses the Pythagorean theorem

第一个等式使用了毕达哥拉斯定理

the second inequality uses the second equality is because J\* use a fixed point of T and Phi R star the approximations a fix point of Pi T

然后使用PI T的不动点来近似phi r\*的不动点

and finally this by we by the fact that PI T is a contraction of modulus alpha gives you this

最后，由于PI T是一个基于alpha的收缩映射，可以得到这个不等式关系

take this on the other side and divide and you get the result that's that

取第一项和最后一项，移项，提公因式，把系数移到另一侧就得到这个误差上界了

so what it says is that if you project J\_mu according to this xi distribution the corresponding Markov chain

如果你根据这个马尔科夫链的分布xi对J\_mu进行投影

then the error that of that that that you get is something that is bounded by the minimum possible error over the approximation subspace

你得到的误差就被近似子空间内最小的概率误差限制

okay so all of this fuss was to show this proposition okay now what we're going to do is put this proposition into practice using simulation

上面所有的证明都是为了这个命题(Norm Matching Property，100页的那个)

SIMULATION-BASED SOLUTION OF

PROJECTED EQUATION

so we have to project this equation and when we want to see how do you solve it by simulation

我们对方程进行投影，然后看看怎么用仿真来求解

MATRIX FORM OF PROJECTED EQUATION

again here's the figure we have a given policy and the T\_mu mapping is given by this with PI being the distribution the transition probability matrix of mu and G being the cost vector

给定策略mu，投影T\_mu，策略mu的状态分布概率矩阵phi和成本向量g

and I want to find R where Phi R is an element of the approximation subspace that solves this equation here

我想要通过求解phi r=PI\_xi T\_mu(phi r)找到r，使用phi r在子空间里进行近似

okay now I have a projection here from T mu of phi R into phi R

我要做一个从T\_mu(phi r)到phi r的投影

and this project in this distance is orthogonal to be approximation subspace this is the orthogonality principle

这个投影和近似子空间是正交的，这就是正交原理

in particular the error the difference between this and that this broken line okay is orthogonal to the subspace spanned by the columns of phi

特殊的，原空间和近似子空间的差分与phi的列产生的子空间是正交的

okay so this error is orthogonal to the columns of phi

这个误差与phi的列是正交的

but orthogonal in the sense of the xi normally capital xi here is a diagonal matrix with the steady state probabilities along the diagonal

但在一般意义上，Xi通常是一个稳定状态概率的对角矩阵

okay so that's what we mean by orthogonality orthogonality with respect to the to the scaled norm of projection

这就是我们想要表达的投影的标量范数的正交性

so the projected equation is equivalent to a matrix equation

所以这个投影方程等价于一个矩阵方程

and this matrix equation is just this C R star equals to d

这个矩阵方程是Cr\*=D

with the matrix C being you read it from here

你可以在这里看到C的定义

Phi prime I times I minus AP and Phi and the constant is this times that

等价方程我就不写了，看这页最下面那个表达式，原方程中有一个常数d，计算方式看课件

so now this project equation is seeing a little abstract it has become very concrete it's just a matrix equation

现在这个项目方程式有点抽象，它已经变得非常具体了，它只是一个矩阵方程。

what's the dimension of this matrix equation it is small small because the row dimension because the column dimension of phi is small so this equation is s by s not n by n ok

这个矩阵方程的维度很小，因为phi的列维度很小，所以这个矩阵方程的维度是s乘以s，而不是n乘以n

it's a low dimensional equation

这是一个低维方程组

however computing this matrix so if I could compute this matrix C and the vector D that would be great it would that you can give to MATLAB

在计算的时候如果我可以计算矩阵C和向量D，那当然是很好了，你可以用matlab来计算·

okay it's a low dimensional matrix inversion give it to MATLAB it will give you the answer

你把低维矩阵给matlab，它会告诉你这个矩阵的逆是什么

but there is a catch here you cannot calculate this matrix C easily because it involves vector or matrix or matrix matrix multiplies that are high dimensional

但是这里有一个陷阱让你没法很轻松地计算矩阵C，因为这个表达式包括向量或者矩阵间的乘法计算，这是很高维度的计算

this phi prime matrix is like this it's multiplied with a big huge gigantic diagonal matrix xi and a gigantic matrix again n by n

这个phi prime矩阵比较小，但是它和一个很大的对角矩阵xi相乘的时候，就会产生一个n\*n的矩阵

so this is a huge calculation even though the end result is low dimensional

即使结果维度很低，计算量也会很大

this is a huge calculation similarly here

计算d的时候也会遇到同样的问题

I have this Phi prime which is like this multiplied with xi which is gigantic and multiplied with G which is also gigantic

我是用phi peime矩阵乘以很大的矩阵xi，再乘以一个很大的向量g

all this inner products end up you a low dimensional vector but carrying out these calculations is difficult

即使内积结果维度很低，但是计算也相当困难

what's the answer to this

那么该怎么计算呢

well last time I mentioned that the primary use of simulation within our context is to be able to make huge inner products make them manageable by sampling instead of adding all the terms

我最后说一次，可以初步使用仿真来代替大量的内积计算，用采样取代内积与求和让计算量可控

and similarly that's what we do here we calculate simulation based approximations of C and D that are manageable and then we use the approximations in place of the real things in this equation and get an approximation to our star okay

相似的方法，我们使用仿真来计算C和d的近似值，然后用这个近似值代替真实值来计算近似的r\*

SOLUTION OF PROJECTED EQUATION

so suppose that we have I give you C and D some hub in a magical way we have found C and D

假设你已经算出了C和d

I can solve this by matrix inversion assuming of course that c is invertible

假设C是可逆的，我就可以算出C逆是什么

okay which is not entirely a trivial issue assuming that is invertible you can use just matrix inversion

这并不是一个小问题，假设矩阵可逆，你可以直接求矩阵的逆

the alternative to matrix inversion is to solve this linear equation by iteration

可以使用迭代法解这个线性方程组替代矩阵求逆

linear equations are solved in two ways either by matrix inversion or by using some kind of an iterative method to solve them

线性方程组有两种解法，矩阵求逆或者用一些迭代法求解

in particular we may try to use this iteration which is called projected value iteration

在这里我要试着用一种叫做投影值迭代的方法求解

it's similar to be to the to the one that I discussed earlier fitted value iteration

这种方法与我之前讲过的拟合值迭代方法类似

how the direction K you have phi of RK and approximation of the what you're trying to compute me J\_mu

已经有了phi(r\_k)来近似你想要近似的，比如J\_mu

you back to apply PI T to it and then get a new approximation and keep going

使用PI T获得一个新的近似，然后一直使用PI T

so here's approximate excuse projected value Direction you are at phi of RK you go to T of phi of RK which is this expression here project it down yet RK plus one then get another one project another one project and so on

这就是投影值迭代的近似方法，现在有phi(r\_k)，使用T获得phi(r\_{k+1})，详细的表达式在这里，然后投影到低维空间获得r\_k，然后一直这么迭代下去

now this can be also written in terms of the matrices C and the vector D

这个过程同样可以被写成矩阵C和向量d

in particular the projected iteration tries to find the vector RK plus 1 such that the distance between these two is minimized

投影迭代想要找到让这个表达式(PVI下面那个计算r\_{k+1}的表达式)最小化的r\_{k+1}

okay so this is a least squares problem in r

这是一个决策r的最小二乘问题

setting the gradient of this objective to 0

让他的梯度等于0

you get this equation here

你可以得到这个方程

and after you manipulate it you see that the new vector RK plus 1 is given by the old vector times some some correction which is proportional to the residual the error that RK makes in satisfying the equation

在你完成计算后，你会看到，新的向量RK+1是由旧的向量加上一些修正给出的，这些修正与RK在满足方程时产生的误差成正比。

and there is also this matrix here so again this is a low dimensional direction but it involves some gigantic inner products in here in here in here in this gigantic in their products our next step is to address them by simulation to approximate them by simulation

这是一个低维空间的迭代，但是phi、C乘以r\_k和与d的计算都需要大量地计算，我们可以使用仿真来求他们的近似解

these are two major methods for solving projected equations

这就是求解投影方程组的两种主要方法

but of course the solution here is exact

直接T之后得到的是精确解

the first one leads to a method called LSTD the second least unethical LSPE and we're going to get into this method shortly

第一种方法被叫做LSTD，第二种方法被叫做LSPE，我们一会来讲这种方法

there are also some other methods but we can't we can't get into those present

其实还有其他方法可以求解，但是我们不讲了

SIMULATION-BASED IMPLEMENTATIONS

okay have any questions here

有什么问题么

so now what we're going to do is use sampling to approximate the matrix in the matrix and vector involved in this projected equation approximate C with Ck approximate D with Dk

下面我们就要用仿真来近似矩阵与向量，在这个投影方程里用Ck近似C，用Dk近似D

and then after we do that then the matrix inversion method is approximately implemented like so

我们这样做以后，矩阵求逆的结果就被近似了

this is the LSTD method probably the most popular method for solving projected equations

这就是LSTD方法，可能是最受欢迎的求解投影方程组的方法了

calculate by simulation approximations to C and D and just use the inverse

通过仿真计算C和d的近似解，然后使用他们的逆

the alternative is to consider this projected value iteration and replace this C with ck replace D with D K and replace this again the simulation based approximation which is possible to do by sampling

另一种方法是用C\_k代替C，用D\_k代替D，然后用G\_k代替(phi phi-1)，这样就有了一个近似投影方程，再用仿真来求解

and the key fact is that these approximations can be computed with low dimensional linear algebra of order of the number of basis functions in not only dimension of the space

这些方法的关键思路就是可以用低维线性代数来计算近似值

any questions

有什么问题么

SIMULATION MECHANICS

okay so now I have a matrix that I want to approximate by simulation what do I do well against the old trick if you want to calculate quantities by simulation you express them as expected values

我想要通过仿真近似一个矩阵，该怎么做呢，跟之前讲的一样，想要通过仿真计算一个数值，把它写成期望形式

somehow you put you you express them as expectation with respect to some distribution and then you sample according to that distribution and you make the approximate

你把这个数值写成在某分布下的期望，然后根据这个分布采样进行近似

it's similar here

跟这个过程是相似的

we collect now the main thing if what's important is to introduce a sampling method that involves the steady steady probabilities of the Markov chain so we can't generate samples arbitrarily but instead we use the Markov chain to generate the samples

我们需要的是一种在稳定概率分布下的马尔科夫链采样方法，我们不能任意采样，应该使用这个马尔科夫链进行采样

so we have this Markov chain I want to involve in the simulation the xi of that Markov chain so I run an infinitely long trajectory of this Markov chain it generates a sequence starting from this state it generates another state another state and so on because of the ergodicity that i have assumed all the states and all the transitions appear with long term frequencies xi(i) and P\_{ij} P\_{ij} being the transition probabilities

所以我有一个马尔科夫链，我想要在这个马尔科夫链的xi的分布下进行仿真，然后我生成了一个无穷长度的轨迹，从某状态开始，转移到新状态，再转移，由于状态的遍历性，我假设所有状态和所有状态转移都以长期概率xi(i)和P\_{ij}出现

with every transition that I generate I compute the corresponding row of phi this is a low dimensional vector and the corresponding cost component

对于所有的转移，我会计算相应的phi的行向量，它与相应的成本向量都是低维向量

so I try attic simulation infinitely long gives me a lot of samples of states and transitions with corresponding costs and the states appear with this distribution the transitions appear with a product of these two distributions

所以我尝试仿真无穷长度的轨迹能够获得大量状态和状态转移与相应的成本的样本状态根据这个分布(xi(i))产生，状态转移根据这两个概率(xi(i)和P\_{ij})产生

and now the formulas are given here

这些计算的表达式在这里

I want to approximate this vector here

我想要近似这个向量(phi prime g，第一行红公式最后一个等号左边的那项)

okay it involves this matrix xi and this vector I view this has an expected value

它包括一个矩阵xi和一个向量，我把它当作一个期望值

I write it like this okay if you expand P and P and so on you can be written exactly like so

你可以把它写成这种形式

and now you can see that it is a it is an expectation involving these probabilities

你可以把它当作概率分布xi\_i乘以p\_{ij}下的期望

now my simulation generates these probabilities they are the probabilities of these transitions

现在我的仿真生成了这些概率，也就是状态转移概率

xi is a probability of being a state i

xi是状态i出现的概率

p IJ is a probability of going to state J afterwards

p\_{ij}是从状态i转移到状态j的概率

so this is the steady state long term frequency of transitions like that

所以这就是平稳状态长期转移概率

and the Monte Carlo estimate of this is precisely this

蒙特卡洛估计的结果刚好是这个

so that's the simulation-based estimate of the D vector

这就是基于仿真的向量d估计方法

run an infinitely long sequence states and transitions occur with the right probabilities yeah if you average this expression by Monte Carlo up to time K will generate K samples a fine number of samples it's an approximation to this that converges to it for K going to infinity with an infinite number of samples

运行一个无穷长度轨迹，状态和状态转移以这个概率出现，如果你把这k个样本按照这个表达式通过蒙特卡洛平均得到的就是对目标的近似，同时如果k趋于无穷的时候，样本无穷多，算法也会收敛

notice that everything here is finite is low dimensional

注意这个(d右边的那一项)是低维的

Phi is a small dimensional vector

Phi是一个低维向量

and this is a number

g是一个数

so I'm just average in small dimensional vectors

所以我把他们平均成一个低维向量

and nowhere do I have to deal with a large dimension of a Markov chain it's the sampling that takes care of visiting the various states

所以我不需要处理高纬度马尔科夫链，只需要用采样来访问各种状态

now the projected equation involves a matrix C and vector D

投影方程包括矩阵C和向量d

the vector D is obtained like so

向量d可以这样获得(d\_k的计算表达式，第一个红公式)

the matrix C is this matrix here

矩阵C的计算方式是这样的(第二个红公式最后一个等号)

again low-dimensional but with gigantic inner products and matrix multiplies in here

低维向量((1-alpha P))与大量矩阵的内积与矩阵间的乘法

and it turns out ok it takes a little bit of algebra to see

这里还是有一些线性代数计算的

that this can be written again as an expected value

这一项(第二个红公式倒数第二项)可以被看作一个期望值

and this expected value can be approximated by Monte Carlo like this

这个期望值可以被蒙特卡洛像这样进行近似(第一个等号右边的表达式)

ok so look what this is

所以来看看这是什么

it's a small column vector multiplied with a small row vector

一个比较小的列向量乘以一个比较小的行向量

okay so these are small matrices small rank one matrices which I averaged over over a large number of transitions

所以这些都是小矩阵，我使用大量的状态转移求平均

and that's basically it these are the formulas and of course there are extensions but these are the basic formulas

这就是比较基本的方法了

if I also need to calculate this matrix here

如果我需要计算这个矩阵(最后一个公式的最后一项)

I can I can also the same simulation can give me a simulation based approximation to this and this is useful in the case of the iterative method

我同样可以使用同样的仿真来对这项进行近似，这在迭代中很有用

(someone asking questions yes yes right the question is the following do I need do I need to know sigh I don't need to know it I just let the Markov chain work and automatically it generates transitions and States according to science I write so I need a simulator of the Markov chain right yeah ok ok I'm right you're absolutely right your question has to do also with whether we do need this assumption we do and we don't okay we definitely need it for the theory as I gave it to you but there's a version of the theory that allows you to to allows approximations of Markov chains that are not exotic and then what you get with the simulation procedure is a little different but it's still usable basically you get you get that's right yeah ok you bringing up another interesting question if my simulation faithfully follows the Markov chain then I'm going to be encounter states that are preferred by that Markov chain so when I do my approximation I'm going to have more samples from the preferential States that this policy likes and the corresponding approximation is going to be biased towards those preferred States and so I may have large errors for the other states the problem of exploration which I touched upon you bring it up again it's a very very important problem and also probably deal with on Wednesday ok we're going to focus on policy issues of policy improvement on Wednesday today we're focusing only on issues of policy evaluation without regard of the fact that we're going to need the policy evaluation to do policy improvement with the next policy so that's let's let's postpone this question until until Wednesday so the first I understand we facility we've had this because the Markov chain is a positive dependent so we assume we have the right in January to be so laughter especially yes yes the question is that do we need to have do we need to to be able to implement this particular policy the answer is yes somehow either you need to have some analytical expression or more likely to be able to simulate transitions from a given policy now note that even though we do not have the policy has a vector of actions because that's too big the assumption is that at any state we can generate the action of that policy there's a computer routine that goes in and finds the action of that policy typically of course by just doing a policy improvement on the preceding policy so that's so so in other words you generate the action in the corresponding transitions as you go you don't need to have a model ahead of time and that's that's there's no way to avoid that yeah question so

实践时需要多少样本才能比较好地估计期望值：不知道，理论上越多越好，但是少到什么程度不可用谁都不知道

in the practice we always have a very finite number of samples so is there any results about onami simple passes prior to getting high quality restoration of yeah the question has to do with how many samples do we need to have an accurate accurate enough approximation of C and D for this method to work in there are a lot of unanswered questions around this issue impact in in theory you would like to have many many samples that would make the evaluation of each policy quite expensive in practice practitioners actually use very few samples and put their faith in God hope that this will work and I have a slide about that

well first of all it's very important to do exploration and to use a somewhat rich sample of States to visit also there are some algorithms that use relatively few samples with each policy evaluation but at the same time do not make the full change than the new policy okay they change the R vector a little bit okay and the iterative methods are better at that but there's a lot of art in here and there are few performance guarantees in theory you cannot guarantee very much theory only gives an indication of what tricks to try and then and then you try them and hopefully they work that's that's the way there is actually asking finished）

okay so convergence here depends on the law of large numbers requires an infinite number of samples

收敛性依赖于大量的样本

and there is also a certain theory of sometimes these methods are called temporal differences methods and I'm not going to get into what's a temporal difference is basically the sum of these two quantities here

有一个卢纶被叫做时域差分，我不会详细讲，只能告诉你们他是这两种方法的累加

and because this temporal differences appear in all kinds of implementations of this methodology very often people view the temporal differences has been an important important quantity and and and they call the methods temporal difference methods to me temporal differences are not all that important they just happen to appear in various mathematical calculations but you may make up your own mind if you wish to go a little further into this field

而且由于这种时域差分方法出现在该方法的各种实现中，人们常常把时域差分看作一个重要的方法，他们称之为时域差分方法，对我来说，时域差分并不重要。它们只是刚好出现在各种数学计算中，但如果你想进一步研究这个领域，你可以自己做决定。

OPTIMISTIC VERSIONS

okay so now let me come to a question simulation

现在我们来讲一讲仿真的问题

this is the standard version of approximate policy direction with an infinite number of samples in between policy evaluations

这是一个标准的近似策略迭代方法，策略评估迭代无穷次

instead of calculating nearly exact approximations it is possible to do a less accurate approximation based on just a few simulation samples as I mentioned earlier and this is called optimistic policy direction

可以用一种准精确近似方法进行评估，进行比较少次数的仿真来近似，我之前提过这种方法，被叫做optimistic policy direction(直译是乐观策略迭代)

why is it called optimistic well first of all we can't under the name optimistic earlier

为什么要叫乐观，之前我们一直不理解乐观的意思

as the value of the policy Direction could use only a few value iterations to to to up to to do a policy evaluation

在策略迭代时可以只进行少量值迭代来进行策略评估

here it's a similar setting but a lot different because the first number of simulation samples there are optimistic methods that just is one sample and they immediately change the the R vector with one sample

有一种比较相似但是很多地方不一样的方法，因为这种方法在策略迭代时之进行一次采样就马上求新的r

others that use a hundred samples other that use a thousand samples very few methods in practice use an infinite or the extremely large number of samples

其他人进行成百上千次的采样，很少方法在实际使用中进行无穷次或者非常多次的采样

often you get better results but there's no Theory really for these methods okay when you have approximation

通常你可以得到一个比较好的结果，但是实际上这些近似方法没有理论支撑

and and often times the behavior is very complex and in the next lecture I'm going we're going to talk about how oscillations may occur in how these optimistic methods may make the oscillations even worse

这些方法通常表现很复杂，下节课我会将什么时候会震荡，什么时候这些乐观方法会让震荡很严重

now here is something that you can say genetically the matrix inversion method inverts an approximation matrix Ck but if the accuracy of this approximation is low the CK matrix will have a lot of noise if you try to invert the matrix that has a lot of noise in it then we get nonsense very often particularly if the matrix is nearly singular as they often are here

逆矩阵法求近似矩阵C\_k的逆，但是如果近似精度较低的时候矩阵C\_k会有噪声，同样逆矩阵会带有很多噪声，这样会导致计算没有意义，特别是矩阵C是奇异矩阵的时候

so there's a serious problem with with the matrix inversion method because if you have limited sampling

所以这种有限次采样求矩阵的逆会导致一系列的问题

particularly for Co conditioned the LSPD method does not involve this matrix inversion tends to cope a little better because it's also iterative in nature it adjusts the vector R little by little and in the meantime accumulates more relevant samples

LSPE就不涉及这个矩阵求逆的问题，他会比LSTD好一点，因为它本质上也是迭代方法，一点一点地调整向量r同时积累相关样本

it's also possible to put a little step size in front of the correction of the iterative method this gamma is a step size which can be taken to be small so that with few samples you do only you do only a small correction and the noises sort of tends to attenuate the simulation noise this gamma step sighs

也可以在迭代公式前加一个比较小的步长gamma，这个补偿可以很小，能够在样本比较小的时候能够少量校正，同时样本造成的噪声也能够被衰减

there's a lot of art involved here and and don't expect success the first time you try this

这些方法中有很多技巧，不要期望一次就成功

(someone asking questions well you have okay so what is the simulation now it's a question is quite the simulation noise and and whether it is what is it harmful well there's a we want to approximate D by something that involves the noise here so the noise is DK minus D and similarly here the more samples that you get the noise is reduced on the other hand still agree from the smaller amount of noise this matrix may become singular or give you very different inverse that this matrix here so if you're going to use this matrix inversion method it's important that you sort of match the the the amount of simulation that you do to the to be the condition number of that matrix but clearly from the point of view of inversion the more samples you get the better off you are but how many you need is not very clear

MULTISTEP PROJECTED EQUATIONS

okay now there are some generalizations of this methodology that I want to cover in the last part of this lecture so let's take a break for ten minutes I have a few more slides