NPTEL

NPTEL ONLINE COURSE

REINFORCEMENT LEARNING

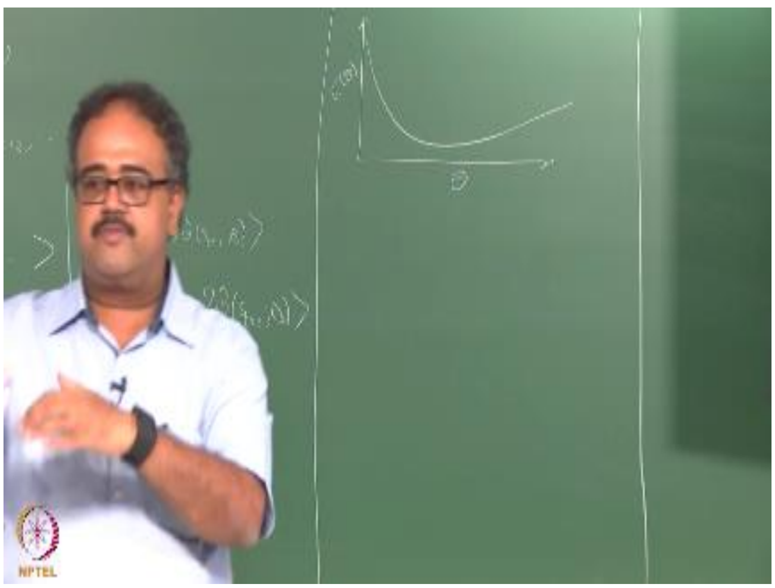
Linear Parameterization

Prof. Balaraman Ravindran

Department of Computer Science and Engineering

Indian Institute of Technology Madras

(Refer Slide Time: 00:16)

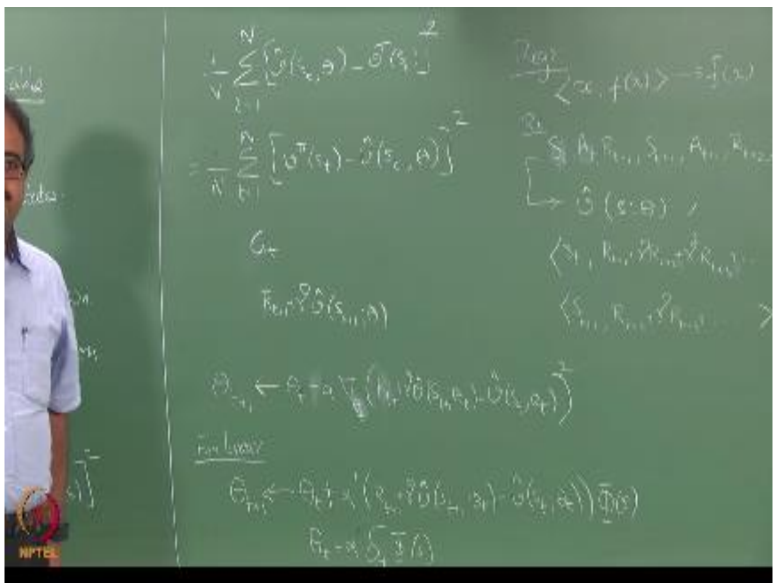


So for the time being so I am going to assume that we will do linear parameterization. So people have a proper visualization in mind when I am talking about the error surface and other things right when talking about error surface when talking about this gradient descent on the error surface and things like that right what is going to be my x-axis, what is going to be my y-axis.

Suppose I draw a picture like this right and I say I am going to find the lowest point on the error function right, what is my x-axis, what is my y-axis this is actually theta right. And this is error for theta right. So remember that when I am talking about functions right when I am talking about function approximations and looking at the response of a function to this I will be talking about the XY space, whatever is my input space right.

And then I will have a response to the function and so on. So but when I am talking about error okay I am talking about a space where it is defined on theta okay. It might sound like a very simple stupid thing to tell you, but the number of times people get completely thoroughly confused about this this innumerable.

(Refer Slide Time: 01:30)



So let us get that out of the way and talk about linear parameterization right. So I am going to assume that my state S some state S is going to be represented by a set of attributes or features right. So my state S will be represented by some key features V1, V2 up to VK. So what are these features depends I mean depends on the problem and so on so forth like I was telling you earlier.

So if you are thinking of some kind of a robotics task it could be very well that it is V1 is the x-coordinate, V2 is the y-coordinate or V1 is joint angle 1, V2 is joint angle to 2, V3 is joint angle 3 and so on so forth, that could be whole combination of it. If you think about say backgammon right V1 could be how many coins are there in position 1, V2 could be how many coins are there in position 2 and so on so forth.

But I remember so what is K in the backgammon case do you remember I told you that 192 okay, it is a very famous number nobody knows what the 102 features are except the V2. So and the IBM actually is kept it as a proprietary think so he was never allowed to me, it is written about one or some features it is never allowed to publish the all the 192 features anyway.

So the K can be rather large right, so even if you think of let us say grid world right, so I can actually make my K, x-coordinate, y-coordinate right and how many robots are to the left of me is how many how many obstacles or that to the right of me I can change it into any way I want right. So for example here is a representation for a separate world right, and there are some obstacles here.

So I can choose to represent the current world as the x-coordinate and the y-coordinate or I can say wherever I am standing its third obstacle above me is there obstacle below me is there obstacle to the left is that obstacle to the right. Now I made it into a four dimensional representation.

So V1 will be one whenever there is an obstacle above me right, so for example if I am here that will be 2, 2 right. So the location will be 2,2 or so one state representation is 2,2 another representation is to say going clockwise from about right, there is nothing about there is something to the right, there is something below and nothing to the left this is another state representation right.

So once I start moving into this kind of domain right, so where I am saying it is some vector V1 to Vk that represents the state I can take any problem I can take any mdp any description that is given to me and think of different ways in which I can encode this right. And what is the right encoding you should be using, depends it is truly dependent is one of those things which is really no right answer right.

So you have to depends on what your problem domain is right and there are some mathematically desirable qualities for the ϕ 1 to VK right, what are the mathematical desirable qualities regression, regression, regression, linear regression. As soon as a linear parameterization are going to linear regression so what would I want of those k features linearly independent right.

So I would expect them to be linearly independent okay just because you finished one course last semester does not mean you have to forget all about it right. So that is a basic course you will be using it again and again and again and again. So it is not something new I am asking about okay, so I said mathematically desirable quality which is it has to be linearly independent okay.

And I have shown some of the guys who did not do our luckily when ml were in PR right. Now between the two courses if you should have known all of this but no. Anyway so when I say linearly independent here is another thing. So what I am asking to be linearly independent values of the features what do you mean missing values of the features are linearly independent linearly I am talking about vector C right.

I am talking about linearly independent vectors so what are these vectors I am talking about exactly all the observations of V1 all the observations of V2 and so on so forth right. So I have S right I but basically I have S1, S2, S3 all the way up till some SM right some states. So if I can write them one below the other right and take the first column that is the V1 vector take the second column that is a V2 vector.

So I want these ϕ 1, ϕ 2, ϕ 3 all the way to this K this K vectors I want them to be linearly independent make sense great. But that is only if I want all this linear regression business right, so if I want to let us say use regression trees right, so what happens if the features are not linearly independent when I am using regression trees, if you remember way back when we started the class I said this is the only chapter where I will be using ideas from ml.

So this is one chapter which you would find a little hard if you do not have a strong background in machine learning, but that is why I am going slow right. In fact it looks like I cannot assume that even the people who did the ML course have background and machine learning but that is another issue. But so people remember regression trees no, yes or no okay yeah.

So what would happen if my features are linearly dependent in the regression tree, what unlike linear regression having linearly dependent features increase will not be catastrophic right. So there might be multiple features which give me the same split value and it is one of them and split and the rest of the features become redundant but you will still be carrying them around but usually they will not give you a good split value once you split on one of those linearly dependent variable.

So it does not matter right, it just you will be doing additional work but the stability or the correctness of the solution all of those things do not matter. So this linear independence essentially matters only if you are solving it using either some kind of gradient computations right or the more numerically intensive methods right. So you can throw a lot of features at it right.

And you can still be hope into it will work right anyway, so I can summarize this as $\phi(S)$. So when I say $\phi(S)$ it is essentially the whole vector ϕ evaluated at state S right. If I want to do that what is that so the first column evaluated on the entire is a vector right $\phi 1$ is a vector so the first $\phi 1$ is small $\phi 1$ evaluated on each and every state okay. If at all I need to use this I will use this notation.

So capital ϕ 1 is a vector where ϕ with the feature ϕ 1 is evaluated at the entire state space ϕ (S) is a vector comprising of V1 to VK evaluated on S okay. So what is the dimension of capital $\phi(S)$ what is the dimension of ϕ 1 m, m where m is the number of states okay. So just keep things straight and that will help us.

So V hat of okay, so this is a linear parameterization so I can take V hat there and plug it in here, so what will I get here is a slight of hand that I am going to do right. I am just going to treat this as target- current estimate and when I take the gradient I will take the gradient of the current estimate only like I will not take the gradient of the target okay. I mean assuming the target is coming from somewhere else right.

If this had been GT if the target had been GT like this that is a perfectly valid thing to do right. But in this case what is going to happen for the linear case anything else something else, you are getting there so I have evaluated the gradient right. So the gradient of V hat will essentially be ϕ - ϕ so that is why that became a plus that okay.

So remember that is a vector equation maybe this is a set of equations. So it will be theta T+ 1 1= theta T+1, T1 so on so forth right. The first component of theta T+1 is equal to the first component of theta T+α times the error times ϕ (1(S)) right, likewise theta 2 will get updated with the same error times ϕ 2(s) and so on so forth this actually set of equations written there right.

And what is this TDA right, so I can essentially write this out this for compactly everyone on board great. So there are two things that we can do from here one we can talk about specific forms of linear function approximate as which is essentially would meet different choices for ϕ 5(T) for different choices or five when I say I am talking about different linear parameterizations it essentially means different choices of ϕ.

So there are many, many ways in which you can choose ϕ and each one deals a different kind of function approximate. So the most popular is to just have some kind of arbitrary function like this, but then there are some very structured forms you can choose for ϕ which gives us different kinds of function of product one thing is to explore that. The second thing is to talk about the use of eligibility traces basically the backward view of TD lambda when you are using function approximation these are two things that we need to do next.

IIT Madras Production

Funded by
Department of Higher Education