Optimal Control, Guidance and Estimation

Prof. Radhakant Padhi

Department of Aerospace Engineering

Indian Institute of Science, Bangalore

Lecture No. # 10

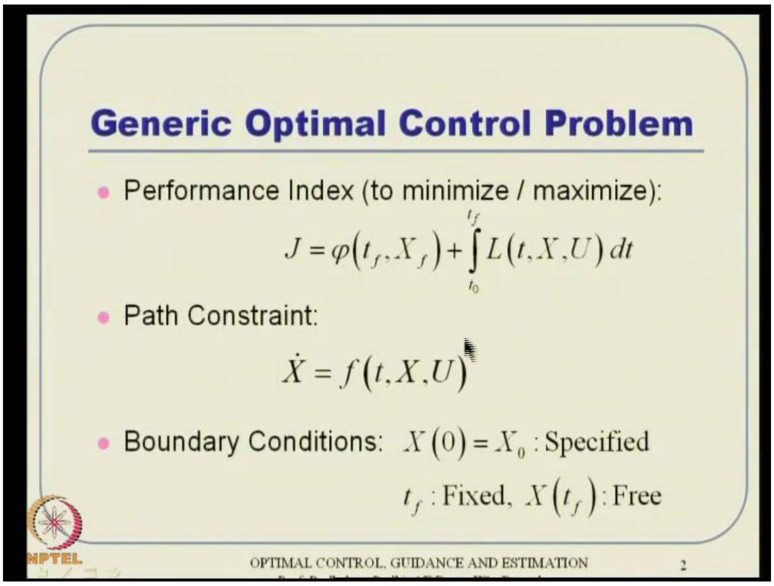
Linear Quadratic Regulator (LQR)-1

Hello everybody, we will continue our lecture in this optimal control course. And then, this particular lecture, we will talk about linear quadratic regulator and then subsequent two, three lectures also, we will keep on talking about these. There are several reasons for that and as you see in the previous couple of lectures, these are computationally complex problems in general, any optimal control formulation leads to these two point boundary value problems and all that.

So, that requires a lot of numerical intensive procedure and all. So, then the, then at least in 50's and 60's people started thinking, that is there any class of problems for which we would not really need all those kind of numerical techniques? And we can get solutions in, in the other computational efficient way, even though it is limited to a class of problems actually. So, that leads to this, this definition of linear quadratic regulator problems and extensively studied and extensively utilized as well. It is one of the very popular tricks, tricks of optimal controller or class of optimal control problem that is found in usage in industries as well, actually.

So, we will see that and then proceed further, some, some of this extension proofs and things like that in the subsequent lectures actually. So, let us see, understand, what is this LQR problem and then, furthermore will be subsequent lectures, anyway.

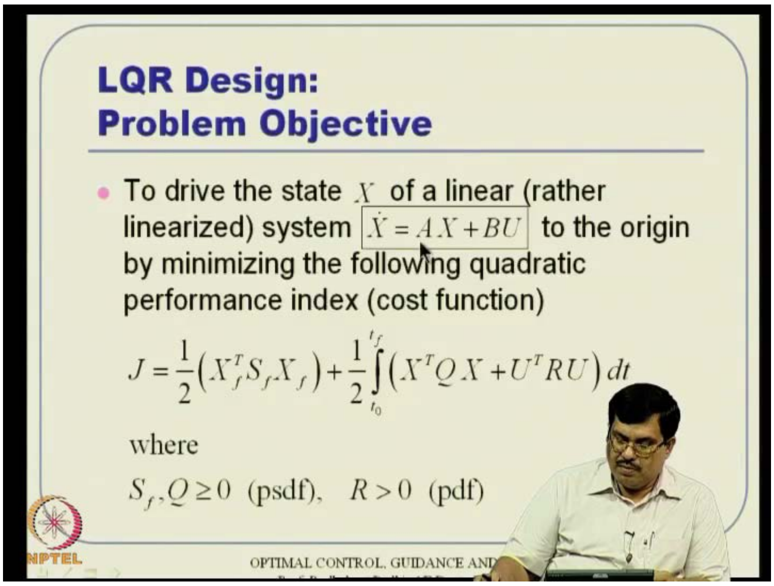
(Refer Slide Time: 01:33)



So, generic optimal control problem, as I discussed so many times before, this is what it is, J, J takes the form of some phi, this is terminal penalty plus some path penalty sort of thing. And then, along with that, that, there is a path constraint, which is a system dynamic equation, as well as boundary conditions, actually.

And we will consider, typically this X t f is free, X of t a, but some problems, it can be (( )) and also basically, actually, alright, anyway. So, this, but this particular LQR problem does not talk about generic formulation any more. Thus, it talks about a specific form of cross function and a specific form of system analysis.

(Refer Slide Time: 02:15)



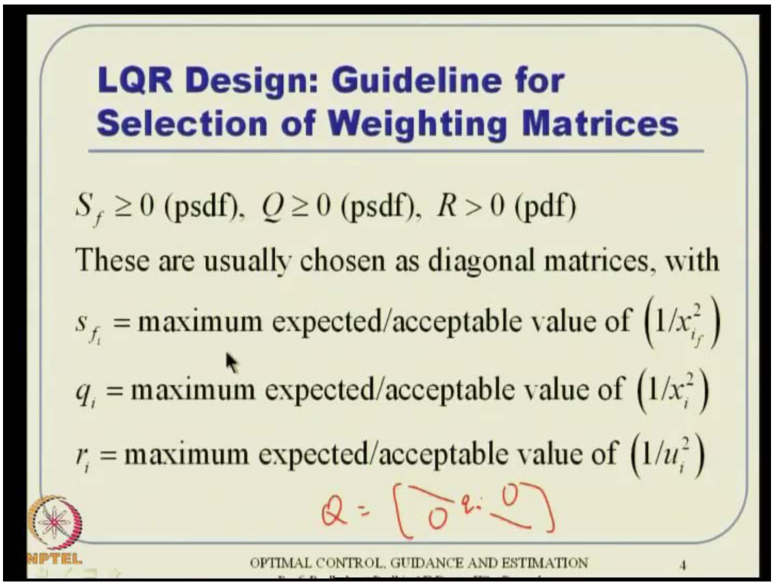
So, the, this problem can be defined this way. The objective is to drive the state of a linear or many times, linear is nothing, but a linearized system, anyway we know that. So, the objective is to drive the state X of a linear system, given by this equation, to the origin, that means, X of T f has to be 0 0, by, so how do you, so that it can be done by minimizing the following quadratic performance index actually.

So, this, the, this is, this is the terminal, terminal penalty X f transpose S f X f and this is the path penalty, where we want to maintain the state deviation small, small is possible throughout and we want to maintain the control as well, actually. The, we want to use as much as less control as possible and in general this, X and del, X and U are nothing, but delta X and delta U, remember that. When we, when you talk about linear system, X is not really the true state, it is the deviation state and U is not really the true control either, actually the deviation control.

So, what you want to minimize is deviation of the state should remain a close to 0 and deviation of control should also remain close to 0, this is the whole problem actually.

There are certain necessities as well, and this has to be an ultimately a meaningful performance index, has to be minimized. So, for that we need this S f and Q. These two matrices has to be positive semi-definite and this R matrix has to be positive definite. And not only that, there are, I mean, there are some one or two other conditions as well, we will talk in a couple of few minutes later.

(Refer Slide Time: 03:45)



So, just remember, that S f and Q are positive semi-definite and R has to be positive definite, so the, so that, I mean, how do you select this, this matrices now, actually. The question is, we are talking about selection of a matrix because there are the turning parameters for the LQR, I mean, I mean a design approach.

So, how do you select this matrices, and one of the, one of the way to select is just to select this as diagonal matrices and that is what, most of the time it is used that way. So, and once you select the any, any matrix as the, as diagonal matrix, the diagonal elements are nothing, but I mean, Eigen values.

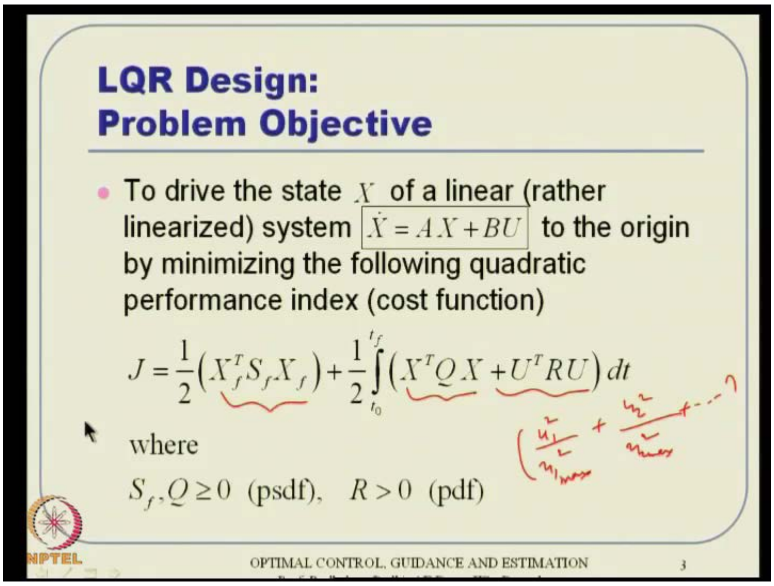
So, if you just put positive elements in diagonal matrices, it will give you positive definite matrix actually. So, that is the trick that people play and they put select diagonal matrices with positive entries, actually. Now, even in positive entries how will you select entries itself actually?

So, that is what I am talking, this for example, q, whatever q we are talking here, q is nothing, but something like a diagonal matrix with q, with q i, everything else is 0. Now, how will you select this q i? So, the guideline here is, recommended guideline turns out, that you expect a maximum deviation, maximum expected or acceptable value of, this value of, this value actually 1 by x i square.

q i is equal to maximum expected value of 1 by x i square or similarly, r i is maximum expected value or maximum acceptable value of 1 by u i square. In other words, if you do that, remember these functions are nothing, but u 1 square r 1 times by, u 1 square r 1 plus u 2 times r 2 like that, actually.

So, what it happens here is some sort of normalization actually.

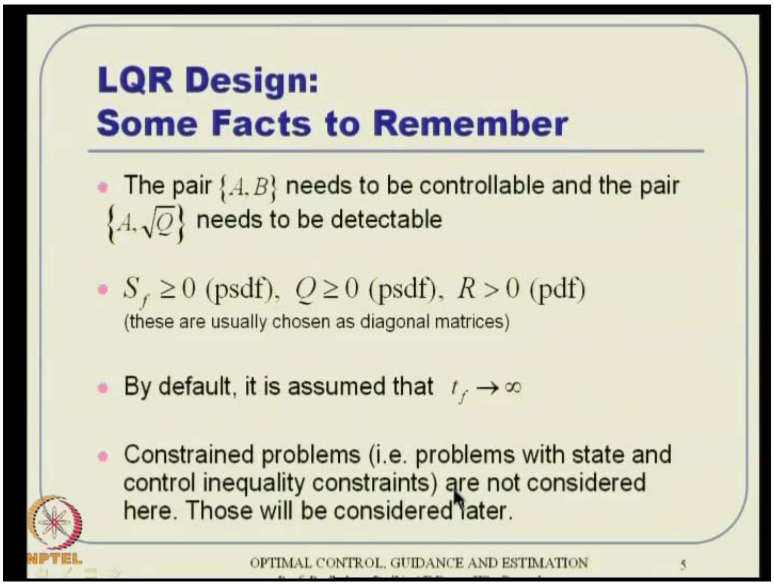
(Refer Slide Time: 05:43)



So, you are talking about something like u 1 square divided by u 1 max square plus u 2 square. This expression what we are looking at, this particular thing in that selection, what I just talked to, this will turn out to be something like u 1 square divided by u 1 max square plus u 2 square divided by u 2 max square, like that. So, that means, this is some set of a normalization actually.

Similarly, we are talking about doing similar operation here, is similar operation here as well, actually. So, that is what I want to tell actually by doing this.

(Refer Slide Time: 06:21)



There are some other kinds, I mean, some other facts to remember. I told you, that this pair A, B, obviously needs to be controllable, the very, very idea of controllability is always there. And if the pair A, B is not controllable, LQR control cannot be done. In fact, any other control cannot be done either, actually. So, the very fundamental requirement is A, B needs to be controllable and also it turns out, that A and square root of Q also needs to be detected.

So, these are all theoretical requirements, as some of you want to know, you can pick up a LQR related book, linear quadratic books are available and Anderson, Anderson and More is probably one of the standard books preferable, actually. And see, some of the proofs are reasons for that, anything like that, that way, anyway.

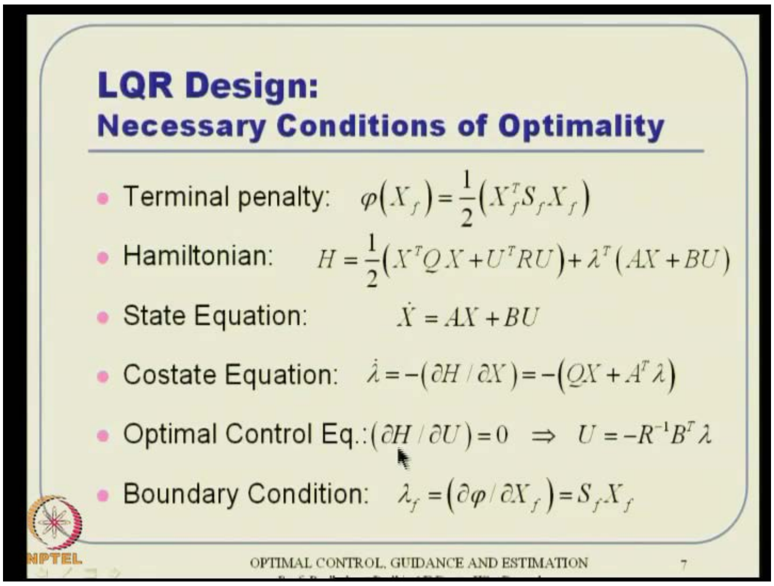
So, this pair A, B needs to be controllable and the pair A is n square root Q needs to be detectable and remember, as long as a matrix is positive semi-definite, you can talk about (( )) square root of Q; this is computable actually.

Alright, so, this is the first thing and this we already talked and by default it is assumed, that t f tends to infinity. In general, it can be finite time LQR, but, but, but if somebody does not tells you, that is, it is a final time LQR is, simply tells this LQR problem, then by default it is assumed, that t f goes to infinity actually. And also remember, that constraint problems, that means, problems with state and control inequality constraints are not considered here, we will consider them later, but not here right now. So, with these things to, with these things in mind, let us proceed further actually.

So, what is our performance index? This is, this is going to be our performance index, half of this plus integral of t naught to t f half of this quantity. So, sincerely, in our generic framework this happens to be our phi and this happens to be our l actually.

So, path constrained is a linear equation X dot equal to X plus BU and boundary condition is given by like this actually.

(Refer Slide Time: 08:22)



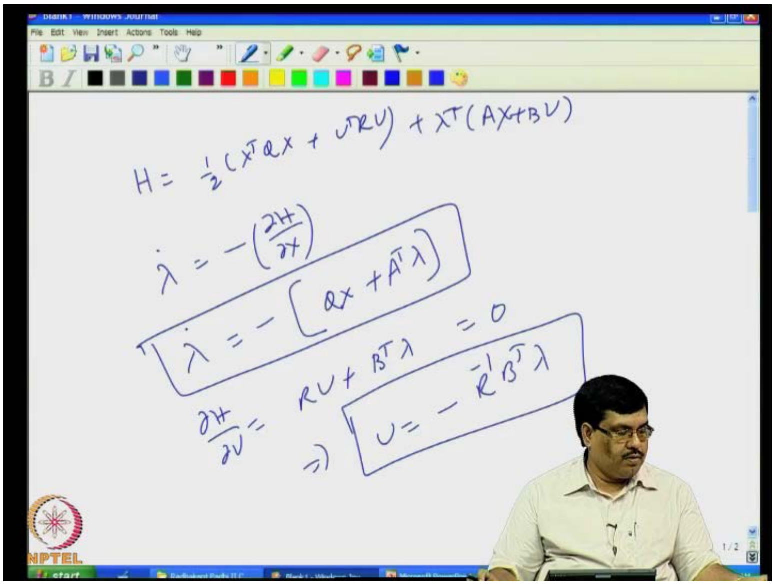
So, what is our terminal penalty? phi of X f is like this.

What is our Hamiltonian? Hamiltonian is nothing, but l plus lambda transpose f and remember, this is, this is our f. This is our f of X, small u, here, big U, well, and I, ever noticed any small thing is scalar and any big thing is a vector. So, just be careful about, that we are talking about vector variables in general, actually.

So, this is our phi and this is our l and this is our f and these are the conditions that are available to us actually. Alright, so this is phi, this is our Hamiltonian l, l coming from here plus lambda transpose f, f is this X plus BU. So, this is what it is. Once you know Hamiltonian and a phi, necessary conditions are all there with us.

The state equation is already there, X dot equal to X plus BU lambda dot equal to minus del H by del X. So, it happens, T be minus of Q X plus A transpose lambda and an optimal control equation is del H by del U equal to 0. So, that means, U equal to minus R inverse B transpose lambda because you can, you can see this from here, let me do this.

(Refer Slide Time: 10:05)



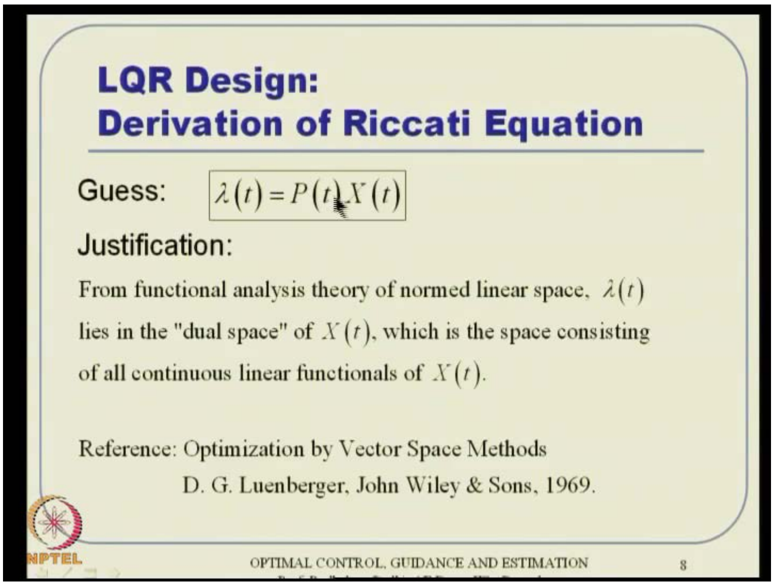
Anyways, let me, so let me quickly do this. Hamiltonian is nothing, but well, this alright, so Hamiltonian is nothing, but our the half of X transpose Q X plus U transpose RU l plus lambda transpose f, f is nothing, but AX plus BU; again this problem. So, this cross-function, sorry, this costate equation lambda dot is equal to minus del H by del X. So, if you have this, this H coming here, this is minus of del H by del X, one term comes from here, which is QX and other term from here, this is lambda transpose AX.

So, this lambda transpose AX, that I can talk about to derivative A transpose lambda basically. And similarly, if I take del H by del U, del H by del U is nothing, but one term comes from this half U transpose RU. So, that is, RU plus the other term is lambda transpose BU. So, that means, if I take derivative, it becomes B transpose lambda. So, then, this is to be equal to 0, that means, you can solve U equal to minus R inverse B transpose lambda. So, that is how it comes, this optimal control equation and this is for your costate equation comes actually. Alright, so this is what, what has been done here.

Alright, so X dot equal AX plus BU; lambda dot is equal to minus QX plus A transpose lambda and this equation del H by del U equal to 0, gives us this U equal to minus R inverse B transpose lambda, actually. The boundary conditions, this is, remember this is our terminal penalty, del phi by, del phi by del X f will turn out to be S f into X f. So, this is how it is.

So, all this condition starting from here, this condition has to be satisfied for our control solution actually.

(Refer Slide Time: 12:20)



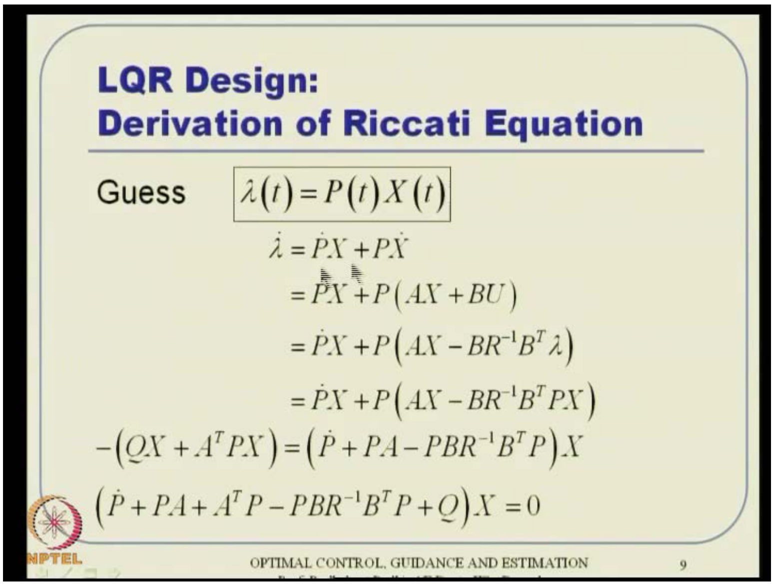
Now, here is the trick in this, this particular problem. This lambda of t is, guessed is, some time varying matrix P of t into X of t and there is, there are justifications for that actually, one into two engineering justification comes from here, that lambda f is nothing, but S f X f, that means, lambda f is a linear function of X f.

So, if, it final term t f, lambda f is a linear function of X f, then how about guessing, that every point of time it is also a linear function of X, that the motivation comes from there also. If you think a little bit, then it is LQR problems, such unique solutions. And so, if you are able to assume some thing, like this lambda of t is equal to P t times into X t, and finally get a solution, then that has to be the only solution, because, because it has a, it has a uniqueness property actually. But even going a little more further, it is, it has been formally proven from functional analysis theory of normed linear space and all that, that lambda of t f in the dual space of X actually, and hence, it has to be the linear functional of that.

So, we would not talk about too much on that, but somebody is interested, you can always see that classic book, very good book actually, Optimizing by vector space methods. If you see that, that the some of the, I mean, I mean proofs and all that will be available. But (( )) very engineering (()) science, it, it happens at the final time. So, we can assume, that it, it can happen in the, at every point of time, from t naught to t f and because LQR admits the uniqueness of the theorem. So, if, if at all we get a solution that has to be the only solution actually. The solution form can differ, I mean, you can, somebody can write in different symbolic sense and different math sense and all that.

So, it is something like sine theta equal to cosine of phi by 2 to minus theta. So, it, both are similar basically. You can, the expression seems to, appears to be different, but the ultimately, they are same actually, anyway. So, this is, this is, what it is. So, with that justification lambda is assumed to be some time varying matrix P times X of t actually. So, this is what it is.

(Refer Slide Time: 14:28)



Then, with respect to that selection, let us analyze or let us try to use all these equations, that we have each equally.

So, now lambda of t is, P t, P t X t. So, what is lambda dot? Lambda dot is P dot times X plus P times X dot and again, I emphasize, do not exchange this, this, I mean, multiplication sequence; sequence is important here actually. So, lambda dot is P dot time X plus P times X dot. So, let us keep this P dot time X, we do not change it, thus keep it like that, but X dot we know, that it is nothing, but state equations.

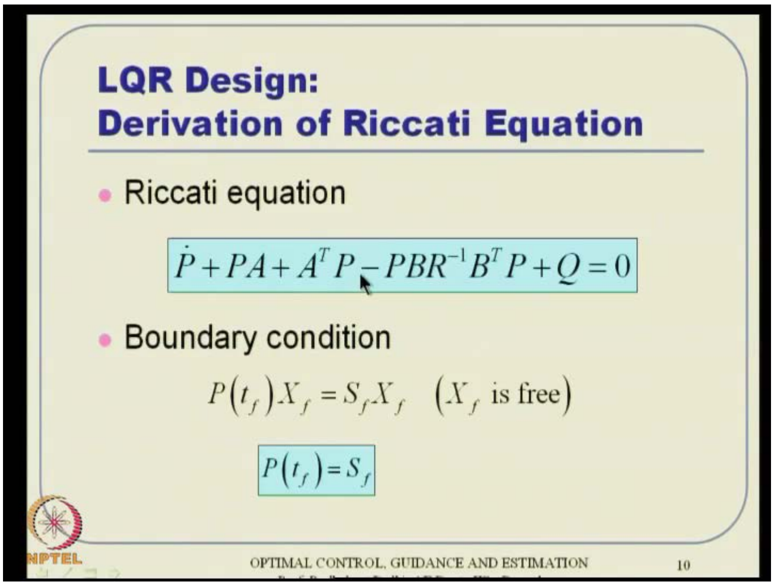
So, we put AX plus BU, that is state equation U, we just derive that. U is nothing, but R inverse B transpose lambda, so we will put that, that here, R inverse B transpose lambda. And then we, lambda, again we know it is nothing, but P times X. So, we will put P times X here.

Now, what is lambda of t? What is lambda dot of t? Lambda dot of t is nothing, but this costate equation minus Q X term, minus Q X plus A transpose lambda. So, we will put that, minus Q X plus A transpose lambda and lambda again, P times X that is put here actually.

So, this side of the story is costate equation; here we use state equation and here is optimal control equation actually. So, state equation getting used here, costate equation getting used here, this side of the story and then, optimal control equation is used here actually. So, all the three equations are embedded here actually.

Now, once you have it something like that, it is all in the some equations like this, you can take everything into the left side and make it something a big matrix times X equal to 0 and remember, X is not necessarily equal to 0. Basically, X is, X is a trajectory, which evolves with time and X in general is not, not 0, but this equation has to be true. So, in that sense and it has to be true for all possible value of X, that is, that also requirement actually. It is valid for all possible values of X, then the coefficient has to go to 0 and then, that is how will get it, Riccati equation actually, is famously called as Riccati equation. Italian mathematician who kind of developed this, I mean, came across this equation in a scalar variable sense for the first time actually. So, that is how it is named as Riccati equation actually, anyway.

(Refer Slide Time: 16:35)



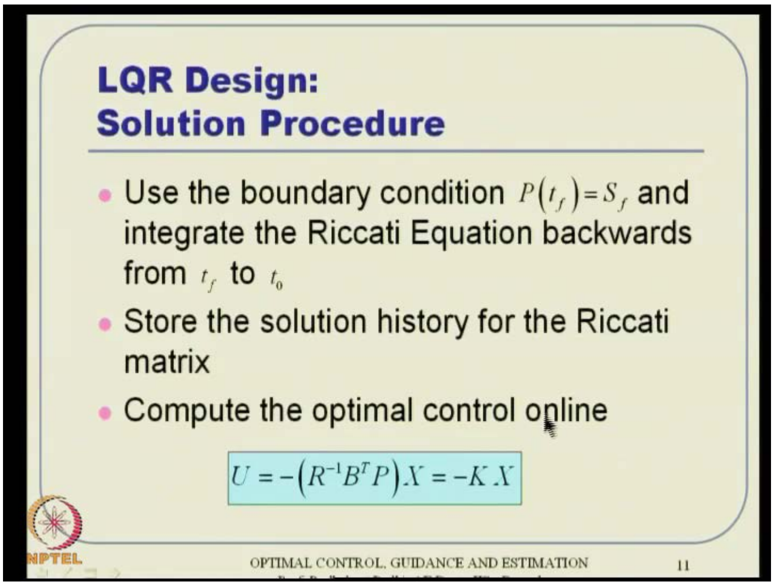
So, this is popularly known as differential Riccati equation and if you remember, this is P dot here. So, obviously, we need a boundary condition for P also and this boundary condition is obtained like this because we know lambda f equal to S f X f, alright.

This lambda f is nothing, but S f X f. But what is lambda f? Lambda is nothing, but P of t f into X of t f from here. So, you put that P of t f into X of t f or X f equal to S f X f. So, again X f is free, X f is not 0. So, hence, this two has to be equal actually. So, we get a differential equation, ok, metric differential equation with corresponding boundary condition at t f actually.

So, now, what is the beauty? Here this problem is, just see this equation and this boundary condition, they are independent of the problem definition. So, we do not really bother about which initial condition the problem operates and things like that. We simply start with this boundary condition and integrate this differential equation backward and store this values of P of t from t naught to t f and then start using, I mean, whenever the operation time starts at t naught.

So, we have available solution ready because P of t and X of t are available, your lambda is available and hence, once lambda is available, your control is also available. So, that is how the things proceed actually.

(Refer Slide Time: 18:17)

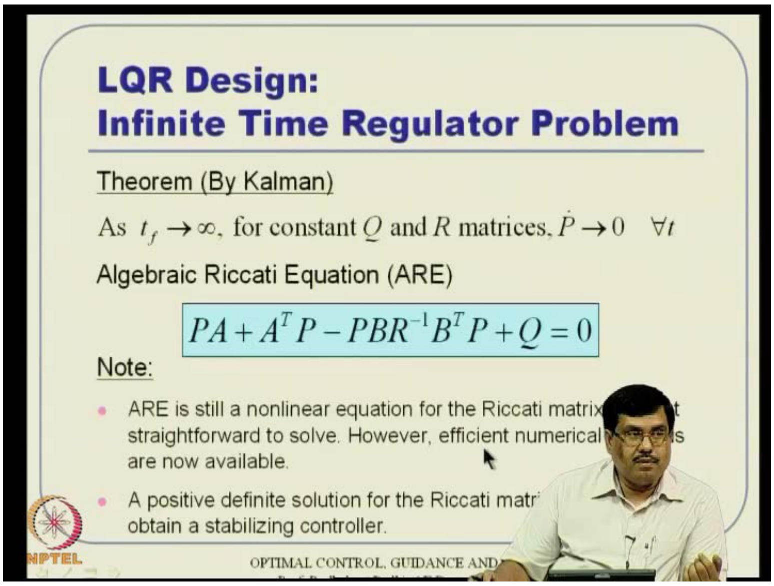


So, this is what is written here, you use the boundary conditions and integrate the Riccati equations backwards, from t f to t naught, store the solution history of the Riccati matrix from t naught to t f and compute the optimal control online, actually. When online, you, to compute it, compute, compute it equal to minus R inverse B transpose lambda, lambda is nothing, but P times X.

So, this part of the things you can think about something like a gain matrix, you can, then hence, you can write U equal to minus K times X actually. So, that is how it is computed.

Now, the question is, do really need to do this? Because this is differential matrix equations and all that, so may not be very good to do that always. So, is there a simplified way of doing that? We do not have to keep on integrating, storing and things like that, actually.

(Refer Slide Time: 19:02)



Now, it turns out, that this great theorem by Kalman again has simplified the (( )) quite a lot and that is what is (( )), that when t f goes to infinity and you have select a, if somebody selects a constant Q and R matrix, they are time variable matrices, then P dot goes to 0 for all time, from beginning to end actually. That means, from P, P dot remains 0 throughout actually, it does not change. Essentially, the P turns out to be a constant matrix actually. If P is a constant matrix, then P dot is 0. Hence, if P dot is 0, suddenly the differential equation turns out to be an algebraic equation actually. That is what is very popular in (( )) everywhere, actually.

So, that is the reason why, if nobody tells t f, then by default we assume t f tends to infinity, actually. So, we can keep on using this as long as we wish, anyway. So, t f goes to infinity, P dot goes to 0 and hence, this differential Riccati equations turns out to be an algebraic Riccati equation, actually. But remember, that ARE or algebraic Riccati equation is still a non-linear equation for the Riccati matrix. And hence, it is not very straight forward to solve, it is not like, by the way, somebody little bit clever can see, that if this non-linear term is not there, this PA BR inverse B transpose P, that is not there, then this is nothing, but a (( )) equation, linear equation, actually. So, that can be solved extremely easily.

But in general here, this is a non-linear equation and it is, it is not that straightforward actually, but what happened is because this equation is so, I mean, it keeps on appearing in a number of problems throughout, across the field, people have done lot of research on that, how to, how to come up with efficient numerical methods, actually. So, it is all available now and some of these has also gone into some routines of these control systems toolbox (( )) actually.

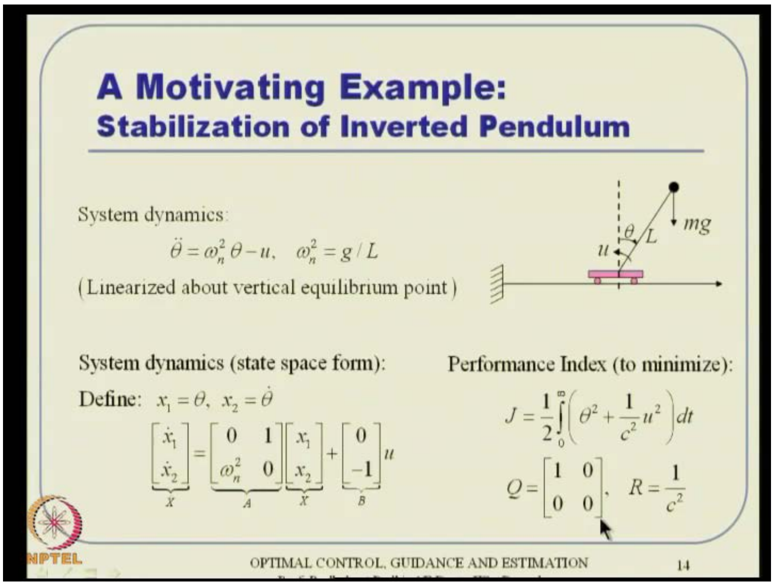
So, if you just use ARE, that is, algebraic Riccati equation, there is a command called ARE. These, these are common for LQR, also LQR, LQR 2, then discrete LQRD, LQR ARE is available for algebraic Riccati equation. So, things like that.

There are bunch of functions available in metal (()) also actually. So, ARE is still a non-linear equation for, for Riccati matrix, hence, but the question, I mean, the point here is, it has drawn quite a bit of attention and hence some good solution techniques are available for these actually. But also remember, that a positive definite solution for Riccati matrix is needed to obtain a stabilizing controller. So, you can, you can prove, that also, that with a positive definite P lot of good things do happen actually.

But it is, remember why, because these non-linear equations, it can admin multiple solutions actually. So, you have to discard all other solution and take one solution, which is positive definite actually. Then, you get lot of good things there, actually.

Before going to further things, which we will do it in next class anyway, I thought, in this class we will see some example problems, which will clarify our ideas in a good way, actually.

(Refer Slide Time: 22:12)



So, the first problem, that I really want people, all of you, to kind of put your hands on is a very standard, classical, benchmark problem of stabilization of inverted pendulum.

An inverted pendulum are, I mean, very intuitive, these problems. There are variety of inverted pendulum problems available and this problem, what we are talking, here is, there is an inverted pendulum on a cart, actually. This a cart and this an inverted pendulum and one way to stabilize that is to move the cart, but one way to stabilize them is also to apply a torque around that.

So, by moving the cart here and there, you are also indirectly applying a torque, actually. But in general, you can also directly mechanize some sort of a motor or something live, that if your, this torque can be generated actually.

So, this system dynamics, ok, this inverted pendulum is also drawn on little more interest in aerospace community because when the launch vehicle is lunched vertically, at that point of time it happens to be a kind of an inverted pendulum, actually. It is not that the string has 0 mass, the entire length will have a distributed mass and all that actually, that is a different issue.

For ultimately, the pay load will be there at the top and the instrumentation and a fuel pump, sorry, a fuel tank and everything will be towards there, I mean, towards the top, actually. So, and hence, you will consider that as some sort of inverted pendulum actually. So, the, anyway, so these are the some of the motivations why this inverted pendulum is problem is studied little bit in depth actually.

Alright, so the system dynamics, I mean, you can derive it using this Newton's law of motion and all that. But you, after you do all that, the dynamics turns out to be something like this in a linearized form. So, in linearized form if you take theta, the deflection from vertical axis is want you want to minimize, anyway.

So, if you, if you do this moment equation and all that, then it turns out, the theta double dot is nothing, but g, g over l into theta plus u or minus u, I mean, if u is in the opposite direction, then it is minus u, actually. So, that is, u is a control variable and theta is a state equation, I mean, this is system dynamics. So, theta and theta dot happens to be the state variable actually.

So, we write it that way, the state space form first we write it, this dynamics into in state space form, where u define x 1 is theta and x 2 as theta dot. So, we put that in x 1 dot and x 2 dot and then remember, this dynamics has to be written. So, it turn, it takes this form actually, x 1 dot, which is x 2. So, x 1 dot is 0 times x 1 plus x 2 plus 0 times u.

So, x 1 dot is x 2, whereas x 2 dot is theta double dot, which is nothing, but that. So, x 2 dot is omega n square into x 1 coming from here plus 0 times x 2, but minus 1 times u basically. That is how we get this a b matrices and all this have to be constant matrices, if there are constant matrices actually.

Now, what is our objective? Our objective is to minimize the performance index like these, why, because we want to minimize theta. It has to, it has to remain vertical. So, want to do, minimize the theta derivation with minimum application of control as well, actually. And this particular problem we write it this way.

So, this R equal to 1 over c square and Q happen to be 1 here because the theta is the x 1, so you have 1 here and 0 everywhere else, actually. And somebody can also think about minimizing theta dot as well.

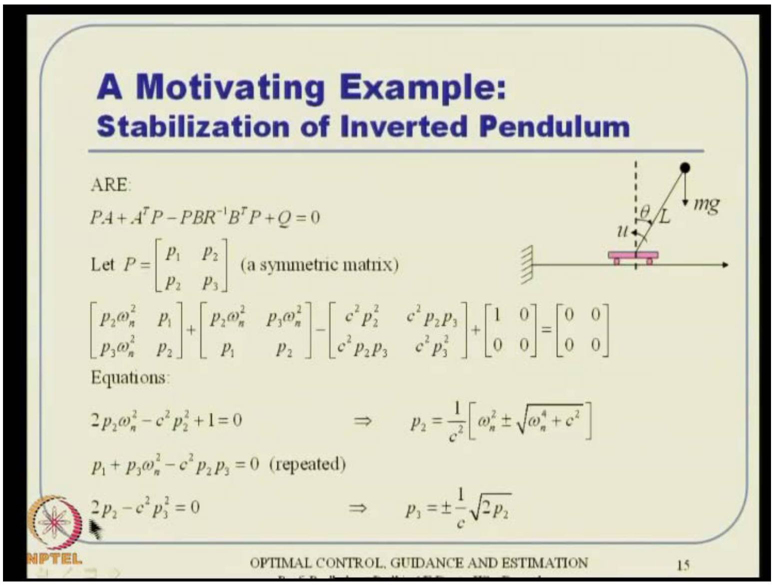
There are problems, there are good things and bad things about that, theta dot minimization means, you can also introduce a penalty for theta dot here, that means, some, something will appear in this diagonal elsewhere, element elsewhere. Good thing about that is, once it reaches the vertical point, that is a good thing to have, you know, you do not want theta dot to develop further. But if you minimize the theta dot on the way, then your response will be (( )) actually; response will not be fast.

So, if somebody is little bit clever, they can initially operate with this cos function and once it is a very small narrow boundary, then you can switch over to the other cos function and try to apply in the other controller actually, that is a possibility. But here, we are not talking about, that it is just a demonstrative problem, anyway, basically.

So, Q happens to be like this and R happens to be like this, 1 over c square and then, let us see, whether you can really apply our, our knowledge and then get some solution of how to do this actually.

So, here is a t goes to infinity. So, this is t f is infinity.

(Refer Slide Time: 27:06)



So, we have this, the algebraic Riccati equation; that is what you have to bother about. So, we will put that and this is one equation. I will also recommend people taking this course should, should remember actually. While once you remember Riccati equation, you are remembering two equations simultaneously.

We are remember Riccati equation anyway, but you are also remembering (( )) equation. You take out this term and it is, it nothing, but (( )) equation actually and these two equations are very, very heavily appearing in many of our fields. So, I thought it is good to remember this equation actually.

So, this is our, this is our algebraic Riccati equation, P A plus A transpose P minus P B R inverse B transpose P plus Q is equal to 0. So, now, we are interested in solving for P and we want that kind of solutions. We should have symmetricity, as well as, positive definiteness.

So, let us start with that. So, we start with symmetric matrix P 2, P 2 and this P 1. So, that is, P 1 P 2 and P 2 P 2 here and then put it back here. So, P times A, this is P and that is A and sorry, this is A. So, if you multiply this P with, with that A, we will end up with this kind of thing, symmetric, plus A transpose P.

So, you will have this A, A times transpose times P and remember, P is the symmetric matrix. So, the, the, this happens to be the, just the transpose of this. This matrix, what you get here is nothing, but the transpose of that you do not have to really compute that; if you compute these, then you, immediately you can write these actually.

Minus P B inverse, sorry, PBR inverse B transpose P and you can compute all that and then come up with that actually. A is given as A, this way B is given as something like these, 0 minus 1 PBR R is given as something like these, 1 over c square. So, your R inverse is just c square and then B transpose and P are all availably actually.

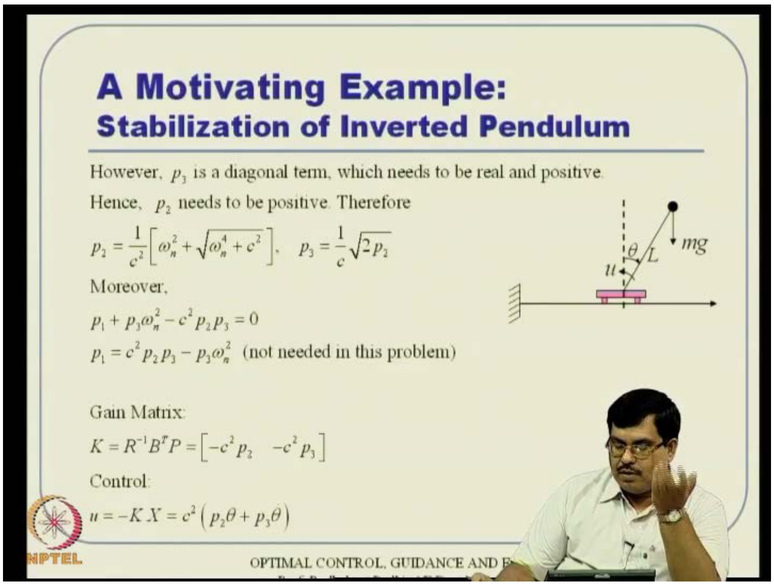
So, you compute this, this matrix multiplication terms, turns out to be like that actually. Plus Q and Q is like this, that is equal to 0. So, this equation turns out to like that and it, you see, now component by component, here just a comment, the, here if somebody wants, they can simply use, I mean, take a value of, numerical values of omega n square and all that and simply use these (( )) toolbox. Once you select value for omega n square and n c square, you can use these LQR solutions and get a solution. That is not a point, but what the point here is, can you, are you, can you able to, are you able to solve it using close form solutions, actually I mean, similarlic solution; that is what we are attempting for actually here.

So, once you write this equation, you write it in component by component. So, the first one, one element will give you us all that, that equation; one, two element will give us that way and one, three will give us that actually, sorry, one, one; one, two and two, two, that is all we need actually because other one will be repeated, actually. The, what you get from one, two will be same as what you get from two, one actually.

So, that is repeated equation you can ignore that because we have already selected this P 2, P 2 here, remember that. So, we have only 3, 3 free variables. So, we need 3, 3 equations only. So, when the (( )) equations are repeated we are not bothered about that actually. So, using these three equations, you have to solve for P 1, P 2, P 3 and remember, the equations are non-linear, in general. So, it can have multipilic solutions, all that.

So, if you use this, this things, remember this is a quadratic equation in P 2. So, we know the solution for that, P 2 equal to 1 over c square, this form actually and P 1 can be computed there. Once you know P 1 and P 3, P 1 is a direct function anyway and this equation if you use, the 3rd equation, you will get, P 3 is nothing, but plus or minus 1, 1 by C square root of 2 P 2 from here.

(Refer Slide Time: 31:36)



So, what it, what it happens? P 3 needs to be positive, so now, there is an ambiguity here and remember P 3 is a diagonal element. So, we are better off by selecting this positive element actually. I mean, this P 3, while I talk about something like a plus or minus term. So, we will select P 3 as positive term actually.

Once you say P 3 is positive term, then P 2 P 2 needs to be positive either actually because P 3 is a square root of 2 P 2. So, unless P 2 is positive, we will land up with some sort of an imaginary number and all that actually. So, we want real values at, at P 3 here and if it is real values, then better, that this has to be a square root of P 2 and P 2 needs to be positive also basically.

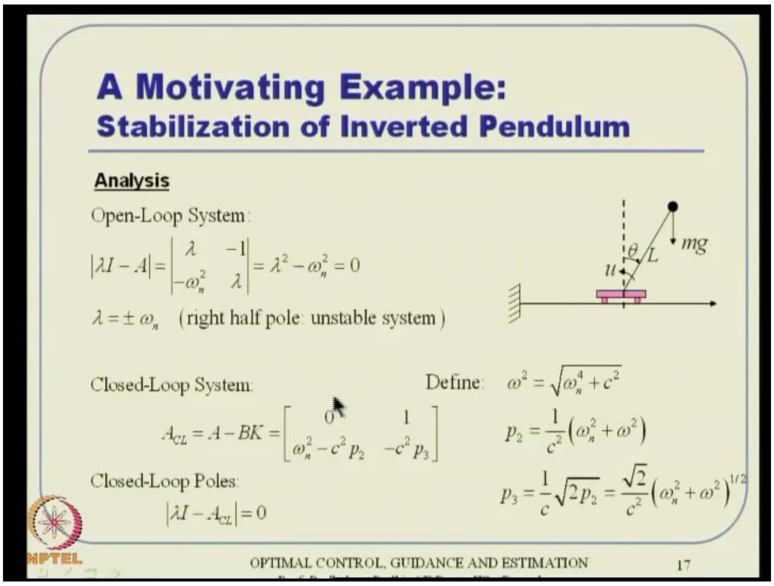
So, if P 2 is positive, then remember this term, what you are getting is more than that. We are better off by selecting the positive value, not the, not the negative sign; negative will be lesser than that basically because this is nothing, this omega n 4 plus c square square root, this is going to, magnitude wise it is going to be greater than omega n square. So, if you select a negative quantity here, then, then P 2 will turn out to be negative number and things like that.

Once P 2 is negative, P 3 will turn out to be imaginary quantity, so that kind of things we will try to avoid actually. So, we will take P 2 as some positive number and then, here also we will select a positive number actually. So, this is what it will happen actually. So, this is how we will eliminate that and P 1 as I told you, can solve it, but it turns out, that P 1 is essentially not needed because your gain matrix turns out to be a function of P 2, P 3 only.

Control gain, gain matrix is almost B transpose P, once you compute that, that turns out to be like this. So, this P 1 is essentially not needed, but it is needed for, for only cross checking purpose or some getting a P link for what is P 1 and all that actually, anyway. So, the control matrix can be computed that way, where P 2 is given something like this and P 3 is given something like this, the control is ultimately given in the form of minus K X. So, minus K X means this way and also notice, that even though we did not take this theta dot penalty here, that does not mean we do not need theta dot feedback, rather feedback is, here is necessary because it is a state feedback control solution anyway.

So, we, the the way to go ahead and do that is, I mean, to implement it is, we need both, theta and theta dot information for the controller, alright. So, what is the, what is the good thing about that? We have done some selection and all that; so final thing, final question is, have you made an unstable system stable actually, that is the, that is the question there. Inverted pendulum, by default, is unstable anyway; analysis will also show that it is unstable actually, we will see that.

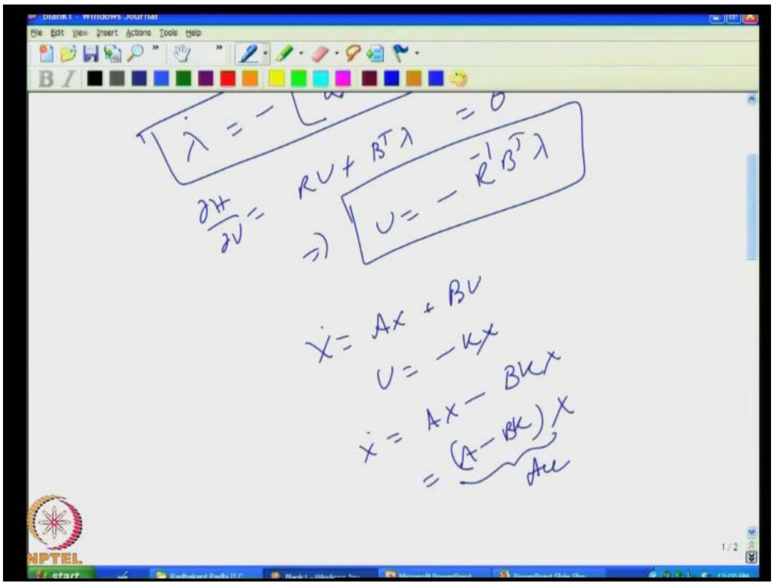
(Refer Slide Time: 34:49)



So, open loop system if we, if we talk about an open loop Eigen value, then the A matrix is only thing. So, if you take this A matrix and come, try to compute the Eigen values lambda I minus A, which turns out to be, like this is nothing, but omega is lambda square minus omega n square is equal to 0. So, that means, lambda is nothing, but plus or minus omega n.

So, obviously, one root is there on the right of flow and hence the system is unstable and we know it very well, that inverted pendulum is unstable, in any case, basically. So, mathematically, it confirms to that, but that is not the point for this is open loop system. Anyway, the whole idea is, after putting the feedback, after applying the control, is the system stable? That means, we are talking about a closed loop system dynamics actually and closed loop system dynamics, it is very easy to see.

(Refer Slide Time: 35:48)

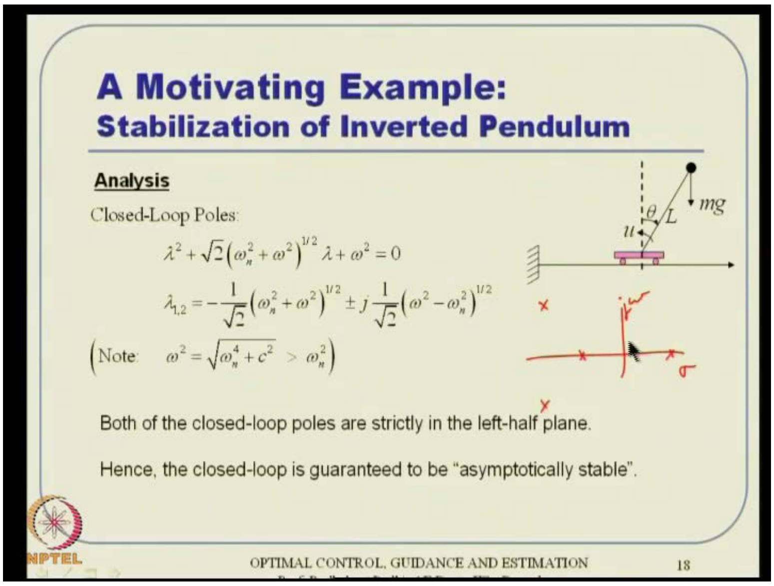


So, closed loop system dynamics, once you have this, this X dot is equal to A X plus BU, BU and U equal to minus KX. So, if you put that again these, then AX dot remains to be AX minus BK X. So, this is nothing, but A minus BK into X. So, this is what you are talking about is closed loop matrix, actually, all the time in linear systems.

So, this is what you are doing here actually, A minus BK, so A minus BK, BK. Now, K is available, so we put that, A is there, B is there, K is available. So, put it back and just turns to be like that and for further simplicity, you can, something like, define omega square is something like this.

So, p 2 and p 3 you can write in a simplified sense basically and closed loop poles should be given by this Eigen value equation anyway.

(Refer Slide Time: 36:45)



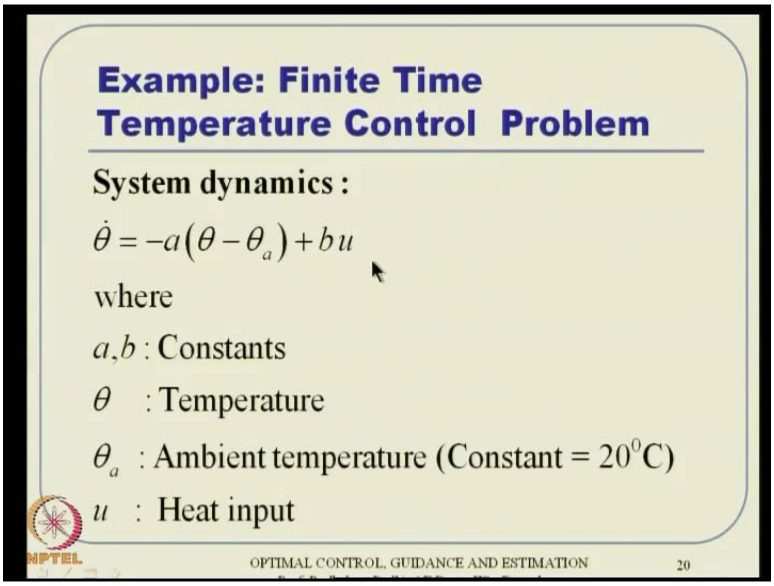
So, you try to expand that equation now and it turns out to be something like this, lambda square plus this quadratic function. I mean, this is the quadratic equation in terms of lambda and this solution turns out to be like this.

So, what happens here? The, the real part is negative actually, that means, both the roots have been shifted to the negative side actually. That means, if you really want to see in a picture sense, then initially, the open loop poles were something like this. And what you have done by doing this is, so we have shifted the poles somewhere here actually, open loop poles were somewhere, I mean, this is your sigma, sigma and j omega axis. So, initially the poles were somewhere like this and one was unstable, and then these two poles got shifted to this. Thus, both are stable actually, both are in a left hand side.

So, what we are telling here is, hence the closed loop system is guaranteed to be asymptotically stable, that is the message actually, alright. So, before stopping this lecture we will talk about another example, which is again a very interesting example actually.

And we talk about finite time problem now and finite time, this, that is an infinite time problem actually. Now, we talk about finite time problem and finite time problem, this is a very standard problem, a temperature control problem in a room, let us say.

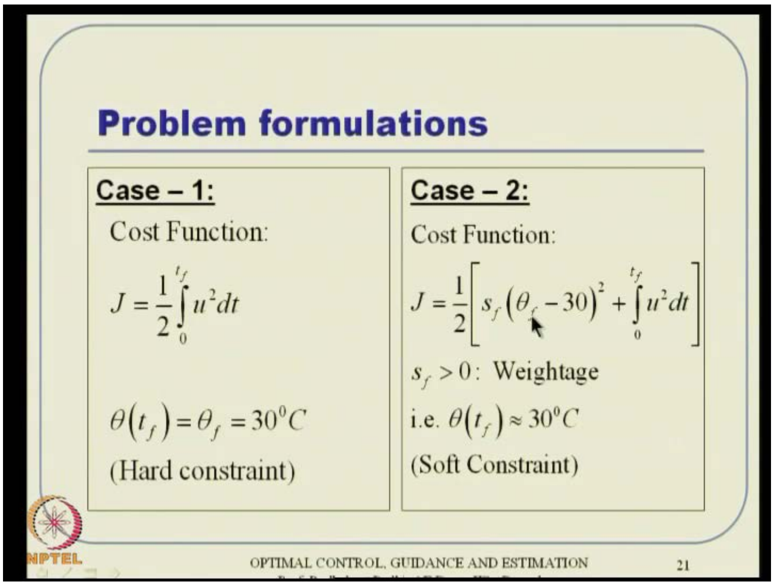
(Refer Slide Time: 38:16)



As the system dynamics turns out to be something like that, theta dot is minus of a into theta minus theta a plus bu, where theta a is the ambient temperature, whereas theta is the actual temperature, basically. And the way to control temperature is by heat input, I mean. So, we are assuming here, that either heat can be given or heat can be taken out, either way, actually.

So, if the temperature is lesser than what you desire, then you have to pump in some heat, so you have to give some heat input. If the temperature is already over, I mean, you want to reduce the temperature, then you have to pump out some heat actually, you take out heat actually.

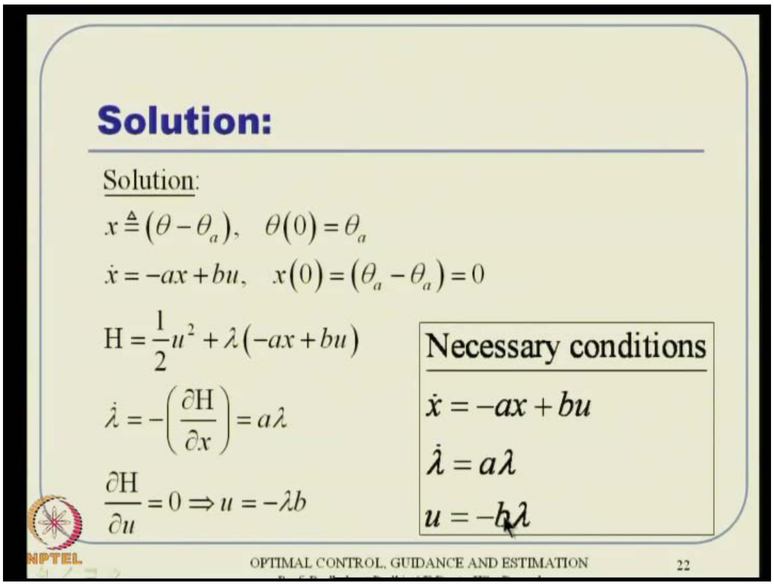
(Refer Slide Time: 38:54)



There are two ways of formulating this problem and we will see the two ways of, both the, both the ways of doing this actually. So, the, sorry, first, first problem is, case one, we will not, we will, we will talk about something like a hard constraint, where theta f final, final value of theta f, the temperature has to be equal to 30 degree. And this case two, we will talk about your soft constraint, where the final penalty is not these here, but hard constraint is impressed. But here, the final control, final constraint is not there, but your soft constraint is in place. Theta f has to be as close to 30 as possible, that is what we are comfortable with and then, we will put this, this cost function actually. So, this is the problem.

System dynamics is same, objective is nearly same when we talk about exactly 30 degree and the other one is approximately 30 degree. And as S f goes to infinite, these two problems are same actually. You just listen, when S f goes to infinity, then theta f has to be equal to 30 degree; that is the requirement actually. So, one is done in a hard constraint way, the other one is done in a soft constraint way actually.

(Refer Slide Time: 40:19)



So, let us see, for the solution of the, for, to, for our simplicity we will define this something like this, theta x equal to theta minus theta a and theta 0 happens to be theta a, that is our initial condition of the ambient temperature, actually.

Initially, the temperature is the ambient temperature. So, what is our x 0? x 0 is nothing, but theta of 0 minus theta a, theta of 0 is theta a anyway, that has to be theta a minus theta equal to 0. So, initial condition in state variable x is nothing, but 0 basically.

And what is the system dynamic? System dynamic is like this and theta a is a constant value, remember. So, theta a dot is 0. So, if I do theta a dot minus a is, so I am talking about x dot equal to minus a times x plus b u. This is what you do, x dot equal minus x plus b u actually.

Now, Hamiltonian is nothing, but I mean, case one, what we are talking here, l plus lambda transpose f and l plus lambda transpose f is, is common to both, anyway. Only phi as a different, this was this phi and that phi 0 and all that actually. anyway.

So, coming back, this is your Hamiltonian, comes half of u square, coming from this term, plus lambda times f, f is nothing, but minus x plus b u basically, that, that is coming from this state equation actually; this is a Hamiltonian.

So, what is our lambda dot? Lambda dot t is minus del H by del X and if you do this del H by del X from this expression, it turns out to be this one. Only this term will contribute actually. So, that is why, in del X is nothing, but minus a, a lambda and minus of that is again plus a lambda.

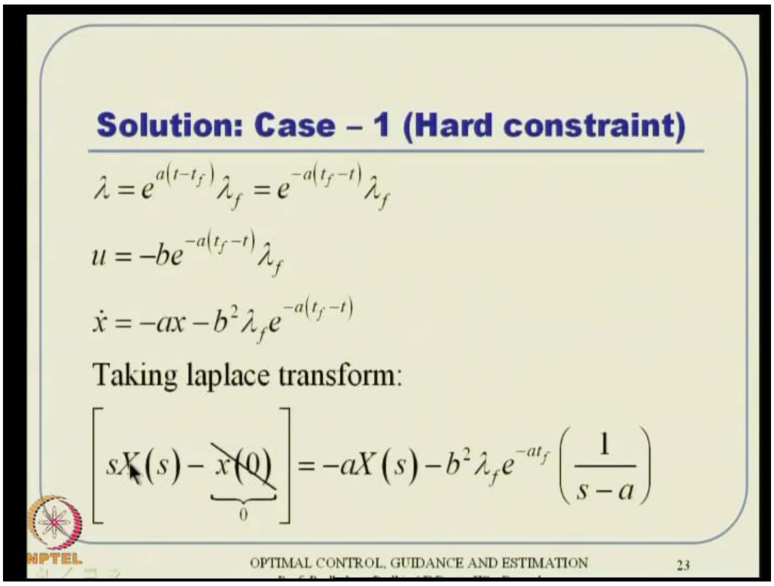
So, lambda is lambda dot, each a times x and here, you can also, I mean, very interesting you can see that. Let us assume that a is positive, then what happens when the homogenous system, u is not there, u is 0, then X dot on one side, you have this, sorry, one side you have this x dot equal to a x and other side you have lambda dot equal to minus, sorry, x dot equal to minus a x and lambda dot equal to a x, a lambda.

So, what you are looking here is, if a is a positive number, then x is state equation, is positive, I mean, stable. If x is a positive number, then, then x dot equal to minus a x, that means, this equation, stable equation, whereas this equation lambda dot is a lambda, a is a positive number and hence, this is unstable equation actually. So, see, that these kind of things are available.

Alright, so, this is what it is, but anyway, coming back, coming back, this is our state equation, this our costate equation and this happens to be our control equation, u equal to minus lambda. So, as long as we know minus lambda times b, so the b is known to us. So as long as we know lambda, we are known actually.

So, necessary conditions to summarize happen to be like this, x dot equal to minus ax plus bu; lambda dot is a lambda and u equal to minus b lambda.

(Refer Slide Time: 43:46)



So, the solution of the, of the first case R constant K, we can proceed this way, this equation to, happen to be kind of an independent equation here, which is a lot of good thing because you can do close more solution easily now.

So, lambda dot is a lambda, so with respect to lambda f, lambda f is something that is known to us typically. So, with respect to lambda f, this solution f is to be like that. U to the power a t minus t f into lambda f is which equal to u. The power, just to, just to put minus sign here, t f minus t. Typically, in all our missile guidance problem and all, this function appears heavily, t f minus t is nothing, but t go, basically, time to go basically.

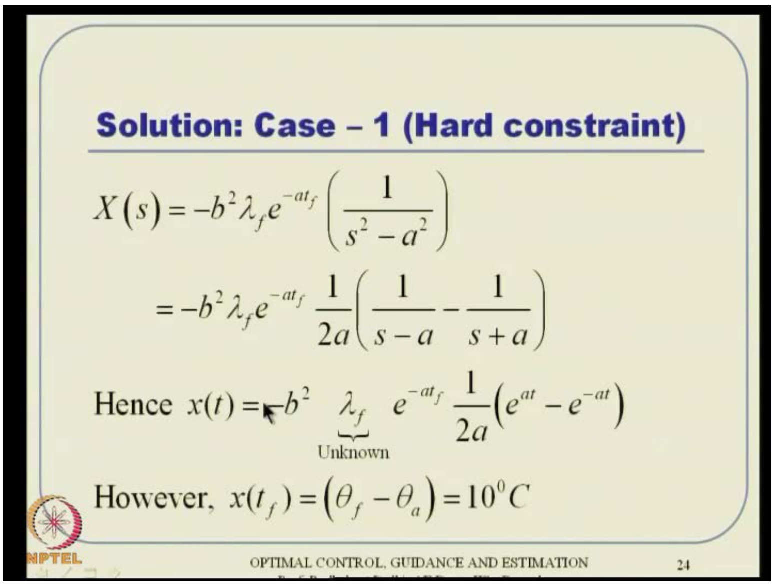
How much more time is available for our control application? So, that is concept of time to go basically. So, this lambda l is nothing, but u to the power minus a times t f minus t into lambda f. And hence, I, if I know my, know my lambda, my control is now known, minus b times lambda.

So, minus b times lambda, lambda is this, but remember, lambda f is still not known actually, so that we have to compute. Anyway, b is available, lambda is available, so what is my x dot? Now, x dot is minus a x plus b times u and u is available now. So, this expression is available. So, you put a break here. Now, this equation is nothing, but linear time invariant system with a forcing function actually.

So, I can try to solve it and one way to solve that is using this Laplace transform way of solving and all that. So, you can take Laplace transform of this equation both sides and one side it will be x times x of s minus x of 0 in time thing and x of 0 is nothing, but 0. So, put z 0 and I just said, in minus I times, that minus b square lambda f and this kind of thing, whatever this, Laplace transform of this function actually turns out to be like that.

So, you solve it, I mean, take this s minus a, I mean, this two term, take it to one side. So, this will turn out be s plus a into x of s.

(Refer Slide Time: 45:55)



And hence, you can solve it using this inverse Laplace transform ideas and all that.

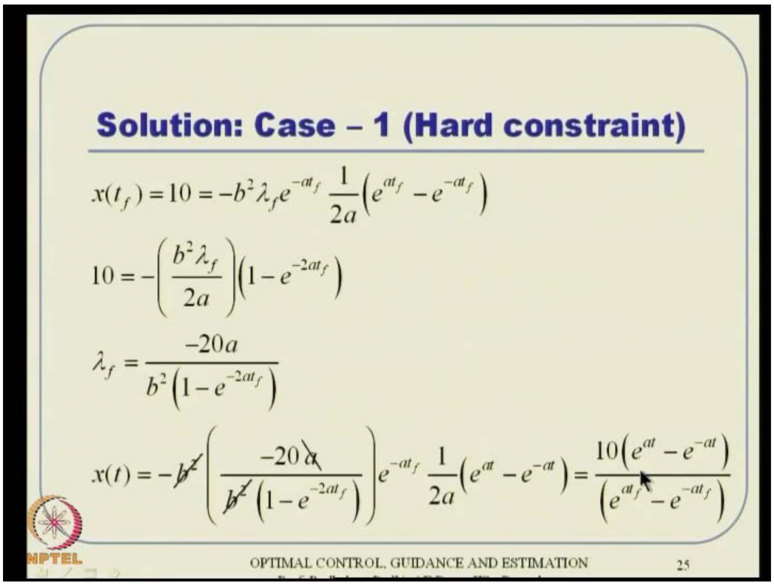
So, x of s turns out to be, if you, if you take it and divide it by s plus a everywhere and all that. So, this is s minus a into s plus a, s plus a will come from this term and that term one side and then divide it by that. So, we will have a of s, s minus a into s plus a is, that s square minus a square sort of thing. So, you will have this, this term. We, you can, I again suggest all of you to, to carry out this algebra yourself (( )) actually

So, x of s is nothing, but this one and then this expression can be done in partial fraction way. You can, if you want to solve, it is very easy rather. So, this partial fraction happens to be like this and hence, you can take the inverse transform and get it that way.

Now, you will be able to see, you will be able to your final hard constraint and hard constraint happens to be this way. This hard constraint in the, in the state space form, in whatever state we have defined, theta minus theta a turns out to be tan actually.

So, x of t f because x of f is now available in the close form way, but lambda f is unknown. Remember, that, that you have to, we have to solve, we are able to solve it using this boundary condition, anyway. So, at this, at this, using this expression put equal to t f and t when you put equal to t f, x of t f is nothing, but 10 degrees.

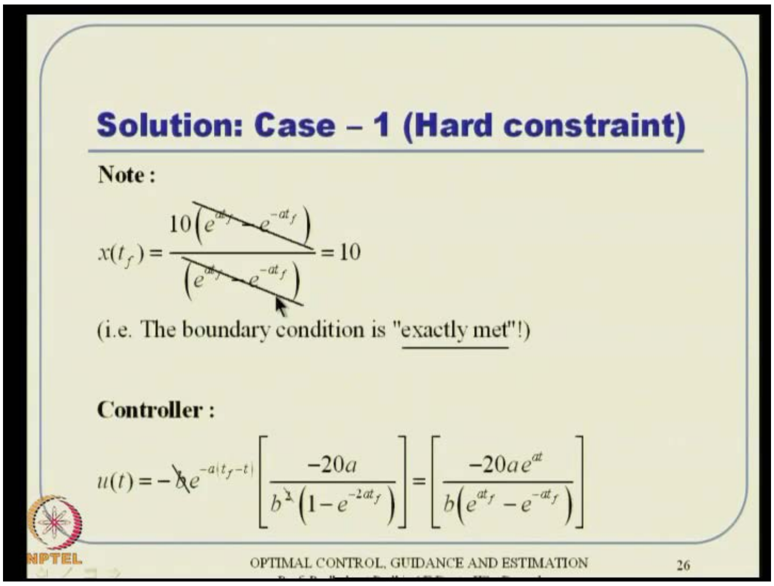
(Refer Slide Time: 47:18)



So, you put 10 and then, you solve it from, solve for lambda f from this equation. So, 10 equal to all that and hence, lambda f turns out to be like that. So, x of t happens to be like this actually.

So, we are able to actually solve a hard constraint problem using this some of the simplistic ideas and primarily, the key point here to note is, this lambda equation turns out to be independent equation, it does not, is not coupled with x and all that. So, that, that made our life simpler, actually.

(Refer Slide Time: 47:49)

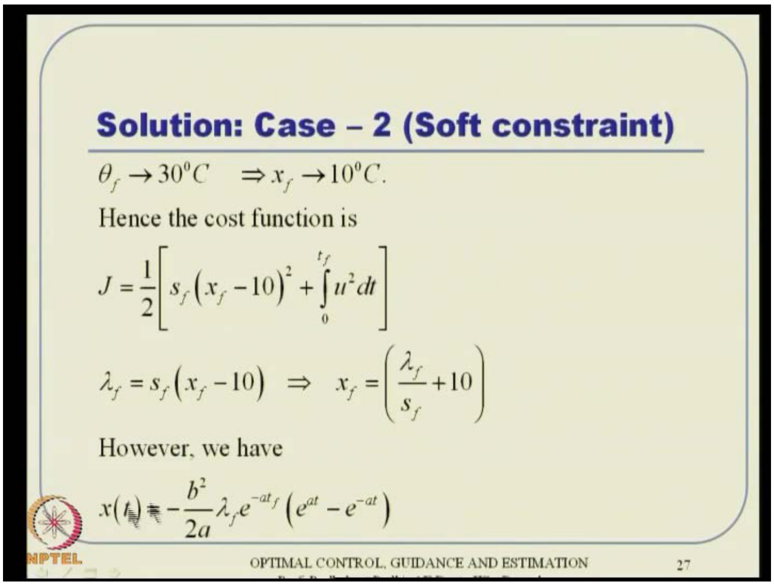


But the point here is, using this expression if I put x of t f, that you can very clearly note, that x of t f we put is nothing, but u to the power t f. The numerator and denominator is the same term, this will cancel out and you will get exactly 10. That means, the boundary condition is exactly met actually.

What is the controller? Controller expression you already have, this is our controller and all. Now, lambda f is an expression available; lambda f, you have solved for lambda f. So, putting that, you will get a controller expression actually. So, this controller if you use for about, whatever, I mean, t f is given, alright. For t equal to t f, t 0 to t f if we apply, t, t equal to 0 to t f if we apply that exactly, you will be able to get it actually and you remember, this, this is a symbolic solution. So, t f is still a variable.

What you can, you can use various values of t f and see this actually, b c this is 1 actually. No matter what t of value you take, it will be exactly satisfied actually at that point of t f.

(Refer Slide Time: 49:04)



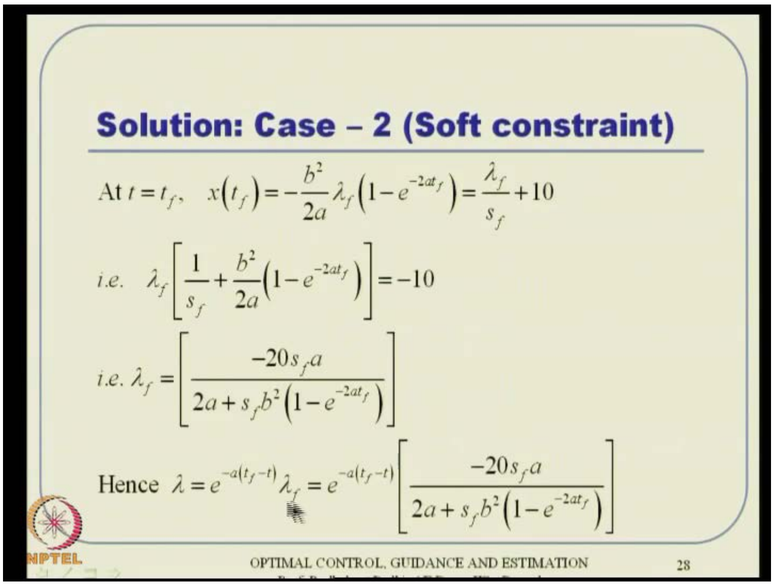
Now, what about the case two, which is soft constraint that we are not, bothered about that, that kind of control actually. But we are not…

So, what, I mean, there is, there is a little bit of drawback here, if you use this, I mean, typically with hard constraint we will have that drawback, anyway. This is not term, I mean, this particular problem it is still ok, but then in general, the, if you have the hard constraint problem, the control requirement at the end turns out to be infinity, which is not possible to meet actually. There is a control singularity at the (( )) actually.

Anyway, so coming back to this, this is a soft constraint approach. Soft constraint means, we are interested in x f going to be approximately 10 degree and there is a corresponding cross function appears to be like this. So, then, lambda f is nothing, but del phi by del x f. So, that comes out to be something like this. So, from here you can solve for x f is something like this.

But x of t we have already solved, that part of the solution remaining same actually. So, we solve it like this, I mean, x of t is available. Now, you can put equal to t f and then substitute this and then, you try to solve for lambda f equal to t f, x will become x f, x f is nothing, but that. So, put it there and that will become an equation in terms of lambda f actually.

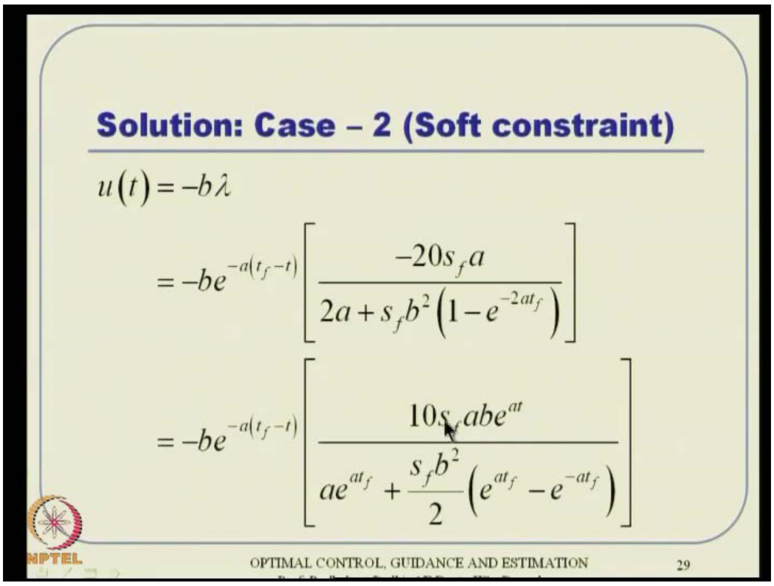
(Refer Slide Time: 50:31)



So, we solve for lambda f; that is what we done here actually. Solve for the, put that equation as there and then solve for lambda f. So, what is happening here? Only the lambda f expression is different, other things, other things are same. One case, the lambda f is transferred to be like this; the other case, the lambda f is turns out to be, we have just solved it here, somewhere turns out to be like this.

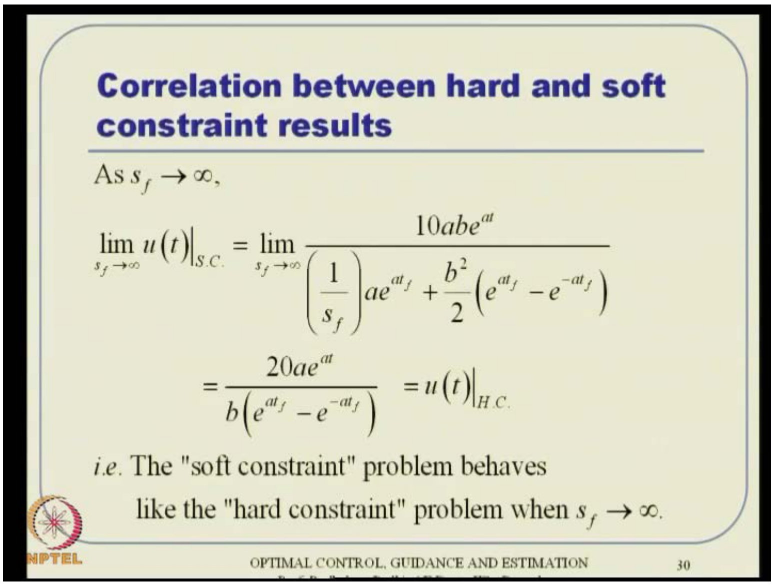
So, these two expressions are different from each other and hence, the control and everything will come different actually, that way.

(Refer Slide Time: 51:08)



Alright, so lambda f is, is like this here and hence, lambda turns out to be like this and hence your control is nothing, but b, minus b lambda and happens to be like that. The two controllers will be different depending on whether we are talking about the soft constraint or hard constraint actually.

(Refer Slide Time: 51:19)



Now the question is, if S f tends to infinity, then soft constraint is nothing, but hard constraint, so does it satisfy that actually? Because its cost, cost function if u see, if you started with the that cost function, S f goes to infinity, as I told these two problems are same, so does it happens that way?

We can see, that the limit, when it do that limiting calculations u of t under soft constraint, when it goes, infinity happens to be like this and hence, you can, you can simplify this expression, you can carry out this algebra and very clearly say, that some of these expression will nullify and ultimately you land up with same hard constraint controlled actually. This expression, this expression, simplify, we have 1 by S f 0. So, this is gone and we are left out with these two (( )), this 20s and 1 b will cancel out. So, we land up with, this expression is nothing, but the expression of the hard constraint control actually.

So, the summary is soft constraint problem behaves like the hard constraint problem when t S f goes to infinity, so that compatibility check is also there actually. Alright, so I think, that these two examples will give us some ideas, that how to handle this, both in terms of Riccati equations as well as hard constraints and soft constraints and things like that actually.

More, most on LQR control and extensions proofs, then all that we talk in subsequent lecture actually, that is all I want to… (( ))