Optimal control guidance and estimation

Prof. Radhakant padhi

Department of Aerospace Engineering

Indian Institute of science, Bangalore

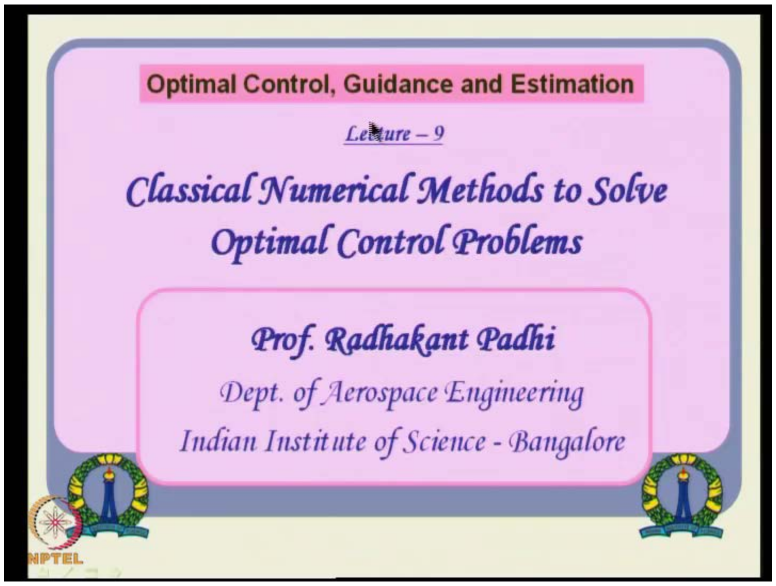
Module No. # 04

Lecture No. # 09

Classical numerical methods to solve optimal control problems

Hello every one, we will continue our lecture series in optimal control guidance in estimation course. In the last couple of lectures, we saw this optimal control formulation from calculus of variations and then we gave a couple of examples which motivated that in general we need numerical methods to solve all in optimal control problems.

(Refer Slide Time: 00:23)



Optimal Control, Guidance and Estimation

Lecture – 9

Classical Numerical Methods to Solve Optimal Control Problems

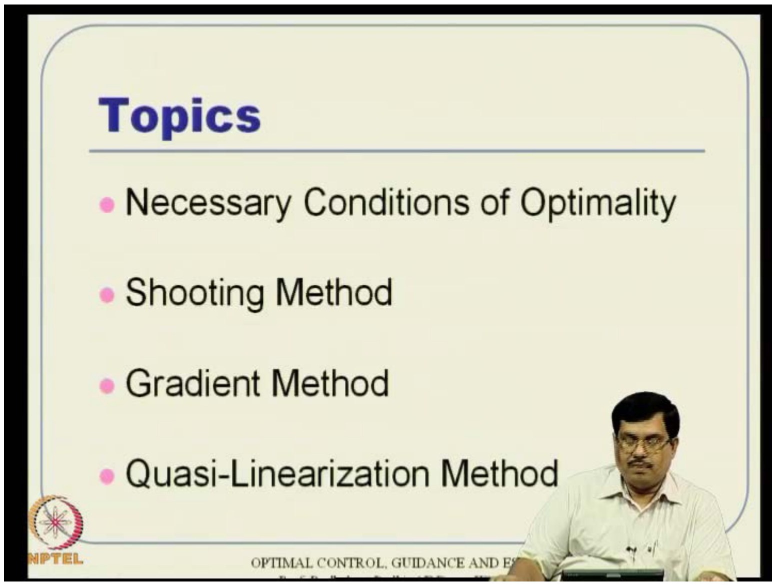
Prof. Radhakant Padhi

Dept. of Aerospace Engineering

Indian Institute of Science - Bangalore

So, let us see a couple of such numerical techniques and then we will follow it up with further things. Now, let's see the topics covered today.

(Refer Slide Time: 00:47)



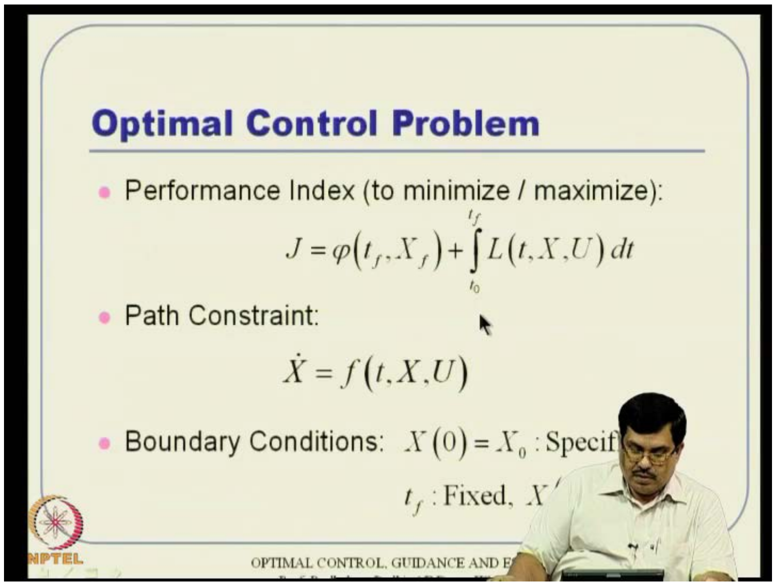
Topics

- Necessary Conditions of Optimality
- Shooting Method
- Gradient Method
- Quasi-Linearization Method

OPTIMAL CONTROL, GUIDANCE AND E

This lecture is first to very briefly review the necessary conditions of optimality, then we will talk about 23 methods and then the first shooting methods followed by gradient method and quasi linearization method.There are various other techniques as well but it's not necessary to continue talking about all the methods. You can, after knowing these techniques, probably see some literature or book or something to follow the other techniques as well.

(Refer Slide Time: 01:20)



Optimal Control Problem

- Performance Index (to minimize / maximize):

$$J = \varphi(t_f, X_f) + \int_{t_0}^{t_f} L(t, X, U)\, dt$$

- Path Constraint:

$$\dot{X} = f(t, X, U)$$

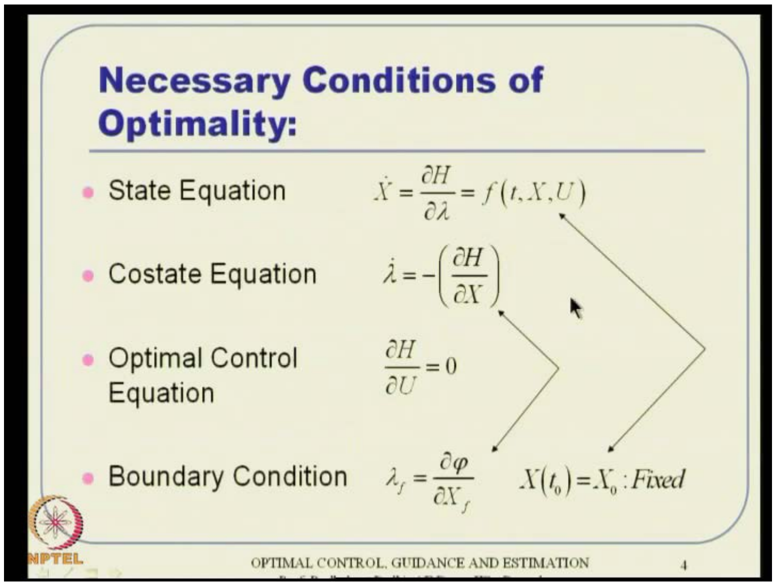
- Boundary Conditions: $X(0) = X_0$: Specif

t_f : Fixed, X

OPTIMAL CONTROL, GUIDANCE AND E

As just started, this is the generic optimal control problem that we are interested in that kind of optimizing or minimize or maximize certain performance index of this form (( )) and along with the performance index we have this associated path constraint as well as boundary conditions actually (( )).

(Refer Slide Time: 01:42)



Necessary Conditions of Optimality:

- State Equation $\dot{X} = \frac{\partial H}{\partial \lambda} = f(t, X, U)$
- Costate Equation $\dot{\lambda} = -\left(\frac{\partial H}{\partial X}\right)$
- Optimal Control Equation $\frac{\partial H}{\partial U} = 0$
- Boundary Condition $\lambda_f = \frac{\partial \varphi}{\partial X_f}$ $X(t_0) = X_0 : Fixed$

OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 4

So, the necessary conditions of optimality that we derived from previous lectures come out to be like that we have a state equation which is x dot be del H by del lambda which is the f since this sort of thing is coming from the system dynamics. Typically, when when followed by costate equation as well which is lambda dot is minus del H by del X and both of the these referred to be differential equations dynamic equations really and then the the third equation is optimal control equation which is a static equation. From this equation we should be able to solve optimal control U H is a function of X and lambda actually . So, that is how it develops now, for this 2 differential equations, we have this two boundary conditions as well.

So, the problem is well defined really and we can see that this state equation takes the initial condition whereas costate equation takes the final condition and that is how it lines up with those 2 point boundary value problems. And then, this 2 point boundary value problem should account for all this boundary conditions properly. Also review if a solution does not satisfy any one of the equations including the boundary conditions that means that the solution is non optimal.

So, we must get to further conditions; almost get to all the conditions with equivalent process itself. So, that's the point (()). So, in this lecture, before getting started with some of these techniques, we also discussed in last class that this state costate equations are dynamic equations and it turns out that if 1 is stable the other turns to be unstable in general (()).

The optional control is a stationary equation from which you need to solve for control and for which we need to know the lambda. Unless you know lambda, we cannot solve it from here and boundary conditions are going to be split. That's what I told you here, that state equation is initial condition whereas co-state equation is final condition (()) and typically because of this, the the state the state equation is double of forward whereas costate equation should be double of backward because you know the the final condition.

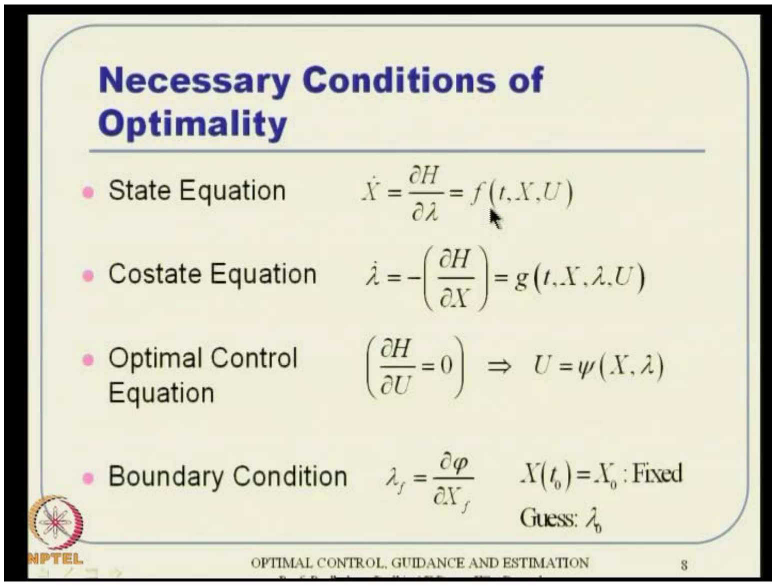
Basically, that's that's why in a typically costate equation we can integrated backwards because we know the boundary condition at the final thing in general. I mean the numerical techniques can circumference over this and then talk about forward integration above the equations and all that we will just see that in a in a while. Ok, so, with all this difficulty that the boundary conditions are split and if one is stable and other one is unstable, that it leads to this great difficulty of, I mean, this complexity issues and all enhance this this particular features that we are talking about, is also known as curse of complexity is optimal control.

So, we will end up with nice formulation but it's a part of necessary conditions we are leading towards this curse of complexity and traditionally, this two point boundary value problems require computationally intensive procedures and we are going to see some of these (()) techniques in a while. Anyway ultimately also remember that when every things are written down, it ultimately leads to this this open loop control structure actually.

Why so, because ultimately after we get the solution in iterative sense we are getting a solution for a particular initial condition. If the initial condition is changed, then the solution is not valid. (( )) So, because of that and once the initial condition is fixed for all features, time t not to t f we will control strategy and it also gets fixed actually. So, because of that, this is a future of getting getting into this open loop control structure actually (()).

So, here are some of the comments and difficulties associated with always. So let's proceed further and then talk about one particular method called shooting. This is the traditional method, very iterative obviously, but I utilize it on problems as well.

(Refer Slide Time: 05:48)



Necessary Conditions of Optimality

- State Equation $\dot{X} = \frac{\partial H}{\partial \lambda} = f(t, X, U)$
- Costate Equation $\dot{\lambda} = -\left(\frac{\partial H}{\partial X}\right) = g(t, X, \lambda, U)$
- Optimal Control Equation $\left(\frac{\partial H}{\partial U} = 0\right) \Rightarrow U = \psi(X, \lambda)$
- Boundary Condition $\lambda_f = \frac{\partial \varphi}{\partial X_f}$ $X(t_0) = X_0$: Fixed Guess: λ_0

OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 8

So, let us see what is the philosophy first and again just to summarize, you have the state equation, costate equation, optimal control and boundary conditions. You once again solve this optimal control equation ok, del is by del equal to 0 then I can end up with this u equal to psi of X and lambda. So, in general, you may or may not be able to solve it in explicit from. Ok, if you are not able to solve the explicit form, then you have to solve using numerical techniques like neutralizing method or something like that; but suppose many classes of problems, it is possible to solve u as an explicit function of x and lambda.

So, let us assume that it is available either symbolically or numerically. Either way, its actually ok once we solve it. You can if necessary, we put it back here in the lambda dot equation and if necessary, you can put it back here as well.

So, then, this u will be eliminated from both the equations. Then you will end up with Xdot is nothing but X and lambda if it is a function of X and lambda and lambda dot is a function of t x and lambda. Actually, most of the time, we will not discuss about this time dependency but if it is, I mean, if it is there, then it is not a matter of concern. Actually it can be handled directly.(())

Now, let's solve this boundary conditions. I told you that this is final boundary condition and this is initial condition; so, this strategy tells us that let us be little more intelligent and have a guess for lambda 0 actually.

Ok, so, how about starting with a guess and again, this you need to emphasize here that queuing a lambda 0 value is not a trivial task. It requires a lot of insight into the problem or essentially it it takes tricks and techniques how to guess this this lambda 0. One way that I will recommend probably is that you can guess a iterative control history, ok you you can rest from t not to t a and once you guess, you then you can probably, I mean integrate, this equation state equation. Ok, x is not available and your control history is available, so, you can integrate from t not to t f. If using that particular control, guess control history, it is nowhere close to optimal in general.

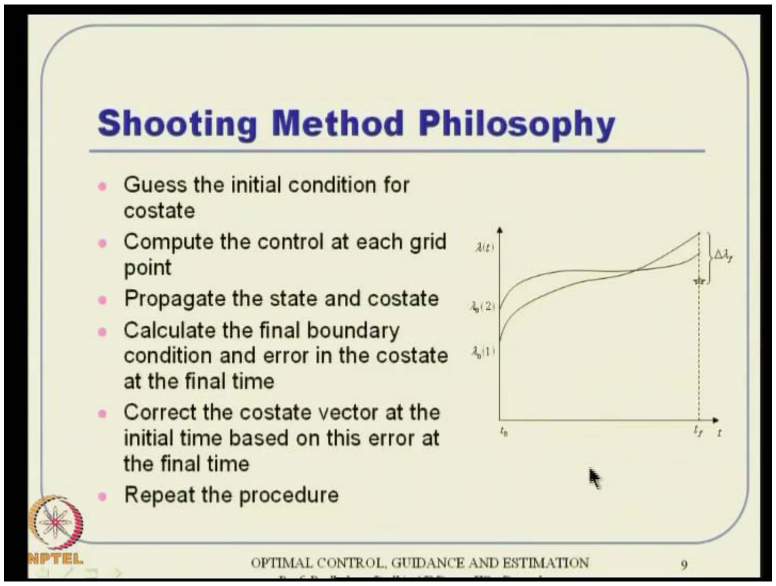
But we know that control history becaus it is an iterative control, anyway, so, if you integrate this state equation from t not to t f and get your x f, and once you get your x f, then you can get this lambda f. Once you get your lambda f then you can, I mean using this condition, you can integrate this equation backwards from t f to t not now, and that is how you get a guess value for lambda 0.

So, lambda 0 is not a trivial task because several several regions like that, and one of the primary region is lambda, in general does not have any physical value if it does have some some geometric value. We will see that later but, but when geometrically it will be the only sense that gives some sort of direction value. It does not talk about original value and lambda is not a physical variable. We cannot talk about position velocity, current voltage, nothing like that, so, we cannot really have a good guess for lambda in general. However, controlling the physical quantity, we can guess the control history from t not to t f. We have agent on stabilizing controller pid controller or just some heuristic controller from the from the problem of definition itself ok.

So, using that, or even sometimes I do recruitment using 0 control if the system is stable. Then, simply start with a 0 based history and and you can integrate the equations; the homogenous equation and then get it x f and then let it be coaxially. There is also a possibility, and again we have prolong dependent once we start working on a problem. We will see that in a later lecture on that, but the point here is, guessing lambda 0 is not trivial, so, give some special emphasis first for guessing a gold value of lambda 0. But once you guess it, (()) for a while you can ignore this and then take a look at these 2 differential equations, have a corresponding initial conditions available.

So, I can take these initial conditions and integrate this 2 equations forward state and costate equation forward, because, you have already eliminated in terms of x and lambda so these two are functions of x and lambda only I have two differential equations two sets of differential equations two sets of differential equations and two sets of initial condition, so you can use this and then integrating further, I will see that okay one side lambda and t equal t f and that lambda what I am getting here, and this lambda computed this they not be same and most of the time they will not be same. That is why, we were telling that this integration process is something like suiting, I mean, you are suiting again sort of thing towards target and and finally, it ands of somewhere with lambda f, but, the the goal point is somewhere here. Lambda is del phi by del f what is should be and then it we sense the error between the 2, and using that error we have to compute and improve this lambda 0.

(Refer Slide Time: 10:49)



Shooting Method Philosophy

- Guess the initial condition for costate
- Compute the control at each grid point
- Propagate the state and costate
- Calculate the final boundary condition and error in the costate at the final time
- Correct the costate vector at the initial time based on this error at the final time
- Repeat the procedure

$\lambda(t)$
$\lambda_0(2)$
$\lambda_0(1)$
$\Delta\lambda_f$
t_0
t_f
t
NPTEL

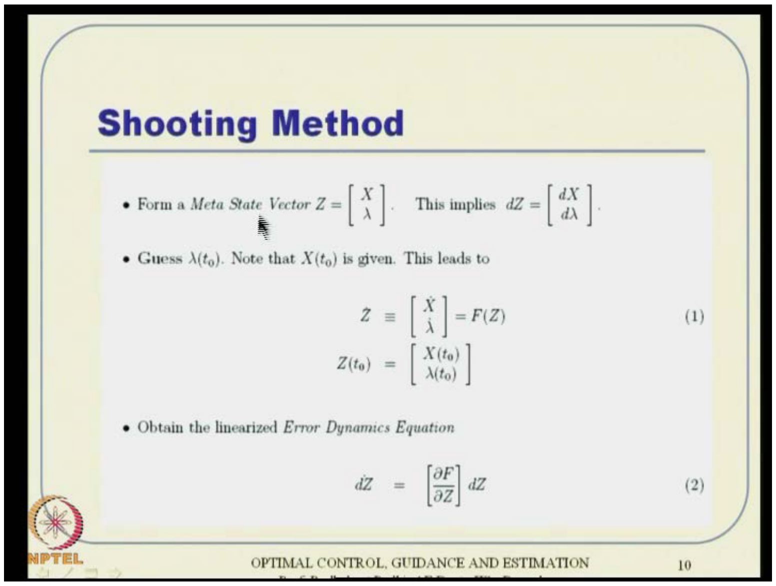
OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 9

How you do that is, this procedure here; so you start with some lambda 0 (()) lambda 0 for first guess sort of thing and then start integrating forward I not to t f you learned of somewhere; and then lambda f star (()) to be somewhere. This delta lambda f you have to compute and correct, this lambda 0 1 to lambda 0 2 and the procedure may go slightly more closer, (()) may be lower suite next time probably, usually the equations proceeds that way. Ok, so, to summarize, guess the initial condition for the costate first, compute the control at each grid point because, once you have the lambda 0 we have x 0 so u 0 is nothing but function of x 0 lambda 0. Once you have u 0, you can get from here; once once you have x one (()) and lambda, one is well. Once you have x one and lambda one, integrating this equations together, you can calculate u one; and then, if u one said u one, you can calculate x2 and anything like that usually.

Now, let us so compute the control at each grid point and then propagate the state and costate equations and then calculate the final boundary conditions and error in the costate at the final time (()) and then correct the costate vector at the initial time based on this error at the final time. Then repeat the procedure, keep repeating the procedure until and unless this lambda f, what you were getting from integrating this point, is very close to what it should be. Then then you tell okay, everything is satisfied because I am starting with initial conditions of the state anyway and then I satisfy whatever I started, I learned of there and this two values are very close and this is also satisfied and I used this control anywhere so this is satisfied and these two equations I am integrating as a part of the procedure anyway, so, all the conditions will be satisfied initially.

So little bit breadth now until... How do you do that actually?

(Refer Slide Time: 12:52)



Shooting Method

- Form a *Meta State Vector* $Z = \begin{bmatrix} X \\ \lambda \end{bmatrix}$. This implies $dZ = \begin{bmatrix} dX \\ d\lambda \end{bmatrix}$.
- Guess $\lambda(t_0)$. Note that $X(t_0)$ is given. This leads to

$$\dot{Z} \equiv \begin{bmatrix} \dot{X} \\ \dot{\lambda} \end{bmatrix} = F(Z) \quad (1)$$

$$Z(t_0) = \begin{bmatrix} X(t_0) \\ \lambda(t_0) \end{bmatrix}$$

- Obtain the linearized *Error Dynamics Equation*

$$d\dot{Z} = \left[\frac{\partial F}{\partial Z}\right] dZ \quad (2)$$

OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 10

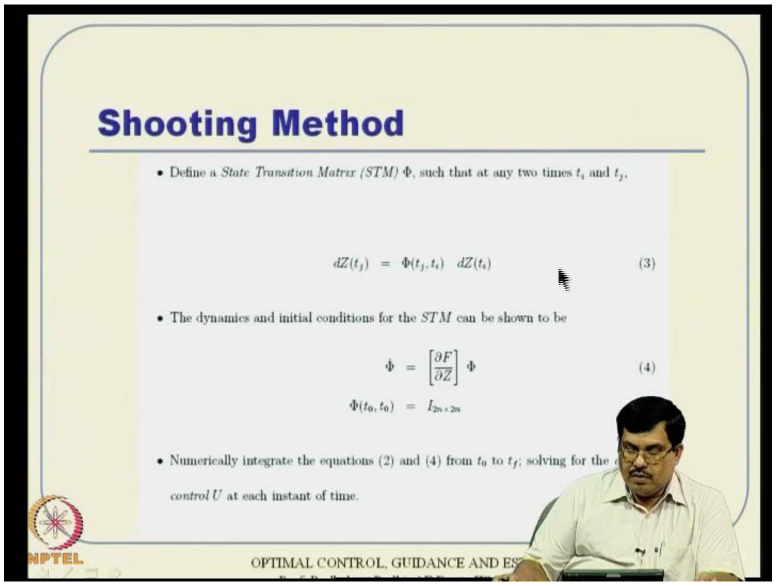
Okay, so, this what high recommend that form a meta state vector x and lambda, that means instead of talking these two separate equations we can always tell that is a function of x and lambda in a meta state; that we called as a meta state vector because u is already eliminated from here. So, this implies that the error in z dz is nothing but dx by d lambda, okay, and the information is used in the linear equations getting basically okay; will see that in a way.

Okay, one side you can find this meta state vector, what is z dot z dot is nothing but a and lambda dot is nothing but if I go back to here this is what in terms of to be f of three. Okay, so and I associated with that z of t not is now available. Because of t 0, I have already guessed structure, so now corresponding to this differential equation, which is knowing near, we can also obtain a corresponding linearised error dynamics which is, a dz dot is nothing but dx dot ah d lambda dot now see nothing but equal to del f by del z in to dz, using this taylors series linearized (()) structure.

So, here is a meta state vector differential equation associated with boundary condition and associated with these equation and and want to come in initial condition, we have this corresponding error dynamics as well. All these things, we have to take advantage of actually, okay?

Now, okay, the point here is this: we have a linearised version of a error dynamics, but remember that del f by del z is not a constant matrix, okay, so, this matrix is certainly not a constant matrix, that is time varying matrix. So, we cannot write this close form solution of if the power at n think like that, we cannot do like that, so, what what is the usual way of time varying linear systems?

(Refer Slide Time: 14:47)



Shooting Method

- Define a *State Transition Matrix (STM)* Φ, such that at any two times t_i and t_j,

$$dZ(t_j) = \Phi(t_j, t_i) \;\; dZ(t_i) \tag{3}$$

- The dynamics and initial conditions for the *STM* can be shown to be

$$\dot{\Phi} = \left[\frac{\partial F}{\partial Z}\right] \Phi \tag{4}$$

$$\Phi(t_0, t_0) = I_{2n \times 2n}$$

- Numerically integrate the equations (2) and (4) from t_0 to t_f; solving for the

control U at each instant of time.

OPTIMAL CONTROL, GUIDANCE AND ES

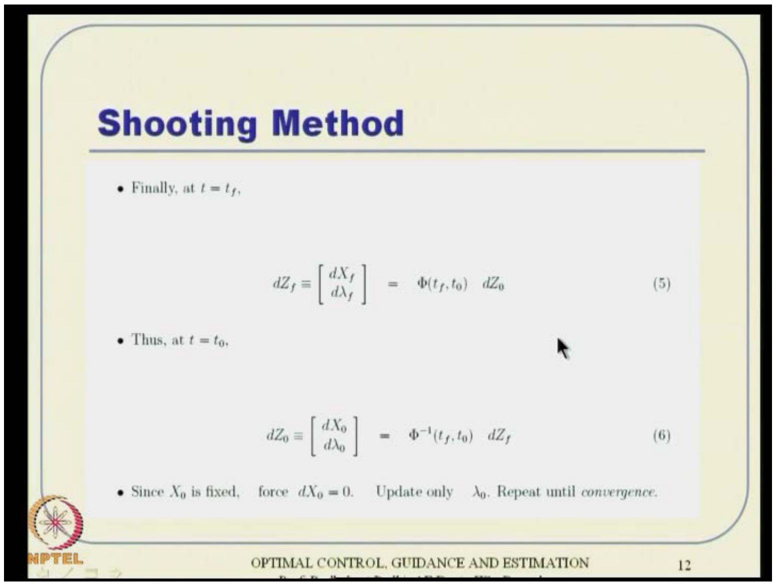
We know that it can be written in the form of state transition matrix. Ok, so, we have this linear equation here, but this time, a varying matrix is involved here and because of that the solutions terms out be something like that. So, dZ at any point t j is nothing but phi of t j t I and the corresponding value dZ of t I; that is how it proceeds this way ok?

Now, if you have this form, that means if you know the error at any point of time t I, you can calculated any other point of time t j. Now, the corresponding equations of the state transition matrix turns out to be phi of its t j t I and the dynamics and initial conditions. Okay, now that we have written in this form, we need a solution, need a number for this matrix actually. So, this is very standard linear system equations theory and then if you write a state transition matrix the same, I mean the state transition matrix ingenerate, will satisfy the same differential equation whatever differential equation we have. But, in the matrixes (()) d z d z is a vector were phi is a matrix usually but the differential equation turning out to be a (()) I mean, exactly in the same form basically So, you can take this d f by d z and plug in here and will satisfy this equation, okay?, and then we have this phi of t 0 t 0 initial condition is is invariably identity. So, if you see this t 0 t 0, both are same, then then this has to be identity usually.

So, the beauty here is that irrespective of the problem definition, irrespective of the initial condition of the problem, this particular phi can be obtained independently; actually form I t not to t f. This is a differential equation and this is a corresponding (()), I mean, initial condition. We need the information of del f by del z also all the way. That's why, it is coupled in a way, about in which segment you are talking about, was the value of del f bu del z can can vary depending on that basically; but those are details Having said this, what happens is, you have a state transition matrix and the corresponding initial conditions actually.

So, this is the system of equation that can be integrated and you get phi of t z t I. Okay, to proceed further actually, so what it tells us is, you can now integrate this equation with corresponding, I mean (()), corresponding state transition matrix equation as well okay?, and from t not to t f and then once you once you know this actually, then your solution is almost ready because you you know this d z (()). Once you know this d z of t I, then you can, I mean, you can proceed further, and then you can integrate this equation. Also in a way (()) this this equation and the corresponding initial conditions are available. Similarly, this equation and corresponding initial conditions are available, so, this set of equations along with this set of equation can be integrated forward. Okay, once you integrate forward, then (()) you can compute this thing at any point of time, ok? (()) Now, let we just use this differential equation, and just using this algebraic equation, because the solution of this is now available from this actually (()).

(Refer Slide Time:18:33)



Shooting Method

- Finally, at $t = t_f$,

$$dZ_f \equiv \begin{bmatrix} dX_f \\ d\lambda_f \end{bmatrix} = \Phi(t_f, t_0) \quad dZ_0 \qquad (5)$$

- Thus, at $t = t_0$,

$$dZ_0 \equiv \begin{bmatrix} dX_0 \\ d\lambda_0 \end{bmatrix} = \Phi^{-1}(t_f, t_0) \quad dZ_f \qquad (6)$$

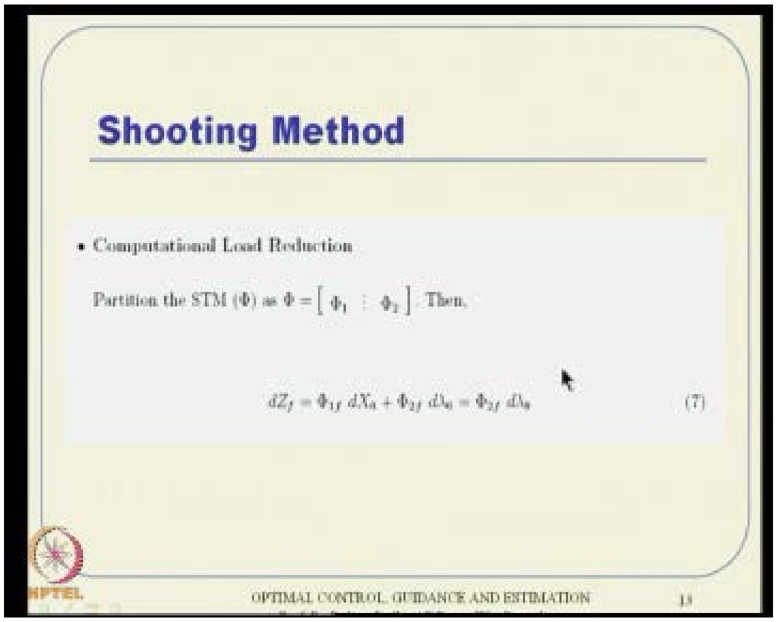
- Since X_0 is fixed, force $dX_0 = 0$. Update only λ_0. Repeat until *convergence*.

OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 12

Now, say this is the thing and then what we will get (()) finally, we are getting something like this actually. At t equal to t f dZ f by definition is d X f and d lambda f and this is simply when we put it, we will get phi of t f t 0 dZ 0. Okay, now the question it is we are actually interested in getting a value for dZ 0 because of what you know is dZ f actually. Ok, ultimately what you know is because lambda f will be embedded in lambda f I mean dZ f will contain in a d lambda f term and we using that will be interested to calculate d lambda 0. The idea actually (()) convert to that. dZ f is calculated this way and hence, in principle this dZ 0 is nothing but this this phi inverse t f t 0 d z f. Also remember state transition matrix are never singular and hence this inverse always exist actually. ok

So, this is always possible to compute, Now, the question here is, we do not want part of this d x not actually right, d x not is fixed and we want to preserve that. So, invariably no matter what we get from the solution (()), that we don't want to consider that actually. So, consider this d x is not equal to 0, increased only in updating lambda 0 okay? So, you can keep on doing that until some convergence happened. This is fine okay, this is fine. I mean, if (()) we can implement this way.

(Refer Slide Time: 20:06)



Shooting Method

- Computational Load Reduction

Partition the STM (Φ) as $\Phi = \left[\, \Phi_1 \;\vdots\; \Phi_2 \,\right]$. Then,

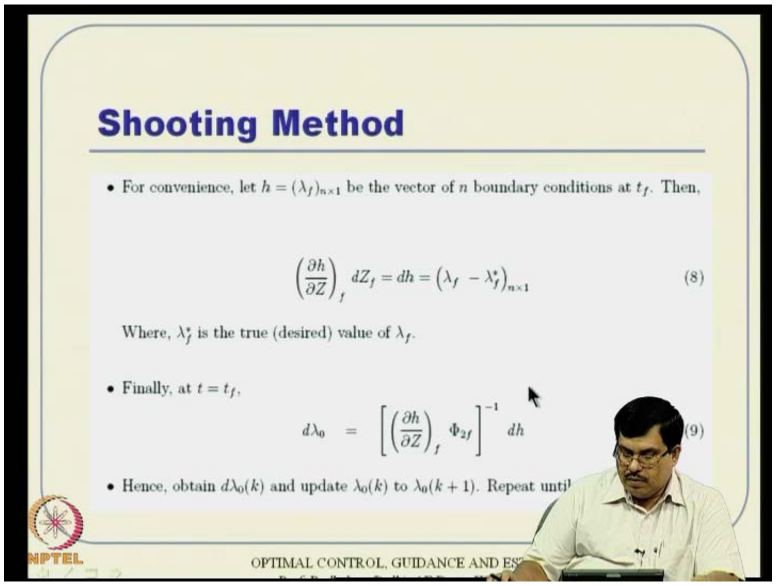
$$dZ_f = \Phi_{1f}\, dX_0 + \Phi_{2f}\, d\lambda_0 = \Phi_{2f}\, d\lambda_0 \qquad (7)$$

OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 13

However, there is a small computationally to equate the procedure, then we make this faster and this idea, therefore you can you can partition this state transition matrix this way: Half way phi one and phi 2 exactly the made loop vertically, then d z f tends out. This way you can write it phi

1 times d x not plus phi 2 f tends d lambda not and then you can write this equation. What you are looking for this equation it can be it can be we written in the form of phi 1 f d x not plus phi 2 f d lambda not actually. Okay, now we are interested in forcing this ah this ah lambda dX not is 0 anyway that's what we discovered in in here, Okay, so, because of that mm this terms are to be zero and hence we will apply actually okay?

(Refer Slide Time: 20:57)



Shooting Method

- For convenience, let $h = (\lambda_f)_{n\times 1}$ be the vector of n boundary conditions at t_f. Then,

$$\left(\frac{\partial h}{\partial Z}\right)_f dZ_f = dh = \left(\lambda_f - \lambda_f^*\right)_{n\times 1} \quad (8)$$

Where, λ_f^* is the true (desired) value of λ_f.

- Finally, at $t = t_f$,

$$d\lambda_0 = \left[\left(\frac{\partial h}{\partial Z}\right)_f \Phi_{2f}\right]^{-1} dh \quad (9)$$

- Hence, obtain $d\lambda_0(k)$ and update $\lambda_0(k)$ to $\lambda_0(k+1)$. Repeat unti

OPTIMAL CONTROL, GUIDANCE AND ES

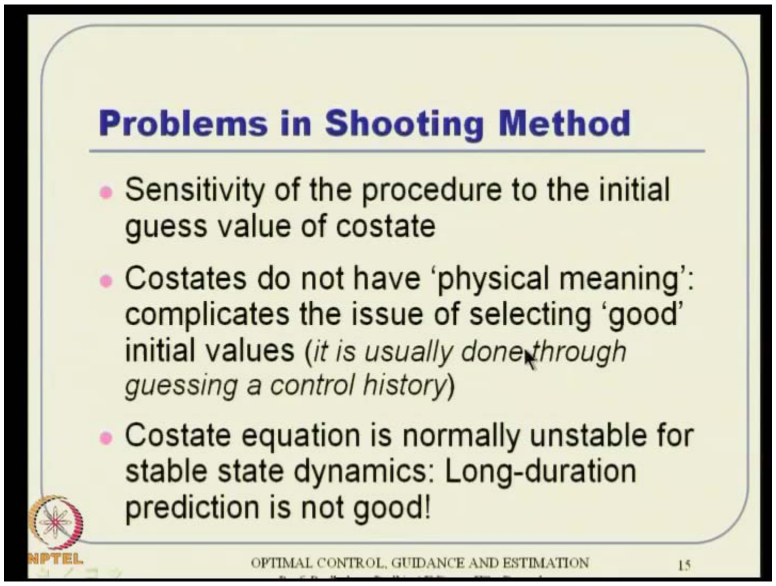
So, what we are looking for is, now you can locate; for convenience, you can think that H equal to some lambda f be the vector of boundary conditions at t f. That means the way that you compute the desired lambda f, we can consider that is a function of lambda f itself okay? in the okay You can generalize this problem a little bit further, I mean, it can be simply lambda f itself do not be a function directly emitting, okay?

So, anyway so the what I mean is this this condition is available, okay, what we see that this this equation is available, so, using this now you can think okay and I will consider that, I mean, call that is h or something that, okay, then you can look at d lambda d I, mean what you're saying here d z f is actually nothing but del h by del z into d z f okay Alright, so you can do that and then that tend turns out be lambda f minus lambda f star, but it should be, and then you can substitute all that and then tell if I substituted back here.

What I what I am getting here is this d z f is, I mean, substitute by here and then required it do and then I can compute it actually ok? So, the whole why it is so because, I see the here, you think about doing that directly. Then this will end up with two n by 2 n the (()) matrix inverse (()) the lambda is x is end and the lambda is n so dZ turns out to be 2 n by two n ah 2 n by one so phi tends out to be n by 2 n so 2 n by 2 n. Matrix inversion is required but here if we do this little bit algebra, it requires that it is this suppliants to be n by two n and this suppliants to be two n by n so that that means you will lend of with n by n matrix inversion only, and which is also logical because all that you need is n dimensional correction rest of then n dimension you don't need any way this is tin to 0, okay?

So, you do this operation and then just multiply these components and then take take the inverse and then you will be able to compute in a slightly lesser computational intensive way, okay? So, it'll just say that matrix inversion from two n by 2 n by n ok?; anyways this is what it is, so, you keep on doing this exercise, you keep on computing d lambda 0 and correct your lambda 0 k to lambda 0 k plus 1 using this d lambda 0 k and then keep on doing that repeating the procedure until convergence. Actually this is also team method, offers its

(Refer Slide Time: 23:45)



Problems in Shooting Method

- Sensitivity of the procedure to the initial guess value of costate
- Costates do not have 'physical meaning': complicates the issue of selecting 'good' initial values (*it is usually done through guessing a control history*)
- Costate equation is normally unstable for stable state dynamics: Long-duration prediction is not good!

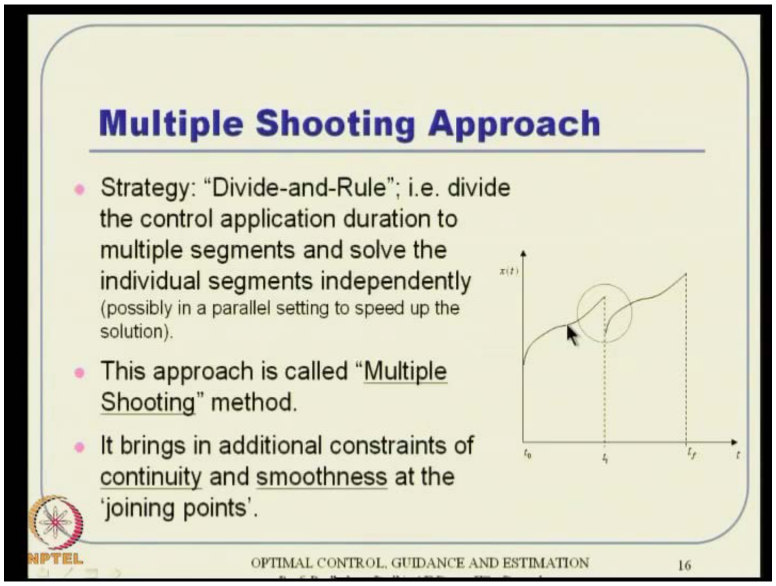
OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 15

Now, there several problems in this method, ok? First of all, as you get to notice, this particular method is is in general sensitive to the procedure, I mean the sensitivity of the procedure to the initial (()) costate Actually, that that becomes a major issue here and that is very apparent also like if you if you guess a wrong value of lambda 0, it turns out to be sensitive and you can impure it, can misleading also but the fact is, if your initial state equation is our system dynamic is stable, then the costate is actually an unstable, So, any amount of different lambda 0 then other than what it should be, then you are actually integrating an unstable differential equation numerically. So, there always is going to grow very fast actually, so that is why it is sensitive to that ok.

So, in general the procedure is sensitive to its initial conditions guess value, ok? Second thing is, we discussed costates do not have physical meaning in general and this complicates the issue of selecting good initial values actually ok?, and i'll explain the procedure how do you guess a value and it is usually done through guessing a control history rather actually; and invariably will end up with, I mean all this, classical methods and many of the numerical techniques you'll have requires that you guess essential control history to begin with. ok

Then, you update on that later actually. Alright, so this is what I already told, constate equation is normally unstable for stable dynamics. So, in general, long duration prediction is not good ok? (())

(Refer Slide Time: 25:32)



Multiple Shooting Approach

- Strategy: "Divide-and-Rule"; i.e. divide the control application duration to multiple segments and solve the individual segments independently (possibly in a parallel setting to speed up the solution).
- This approach is called "Multiple Shooting" method.
- It brings in additional constraints of continuity and smoothness at the 'joining points'.

OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 16

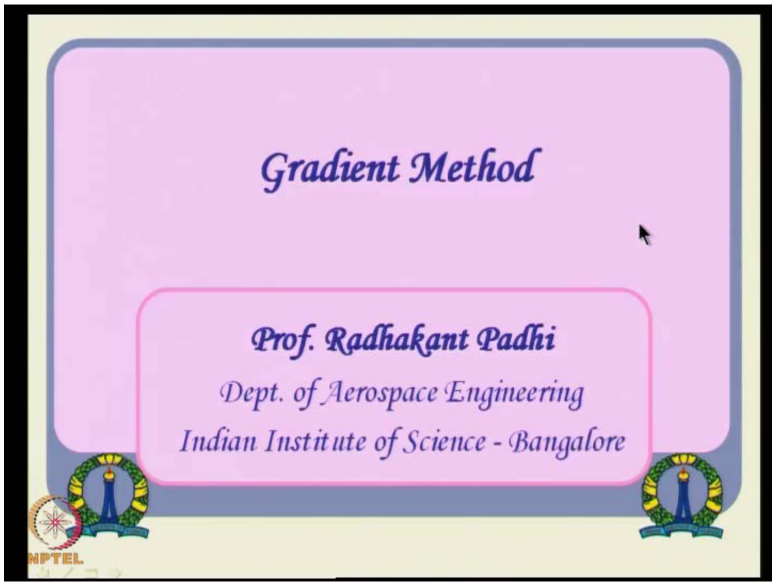
And suppose you have an unstable state, then then also have unstable system dynamic, then also have a problem because anyway you are integrating the state equation forwardised actually. So, we either integrate the costate equation forward or integrate the state equation forward on is stable of there is unstable, and hence everything is very sensitive issue here actually.

Anyways, (()) this is (()) do that (()) at this method is insensitive, so, one way (()) is the policy of divide and rule. You do not have (()) t zero to t f because, the integration error can grow very fast actually, and then all that (()) here is linear philosophy. Don't forget that also the error dynamics had to satisfy this linear equation or error dynamics actually, so, as I as as long as the errors are small, then the error dynamics are valid, otherwise not actually, ok?

So, be careful about that and one idea is, i'll do (()) this segment time duration t not to t f some two segments losses two to begin with t f, same two segments lesses 2 to begin with, or it can be in general multiple segments. 1 side divided to linear 2 segment (()) I j than actually third type, I mean, what is essentially needed is (()) the problem, as two different problems actually in one problem, operates from t 1 to t f, but, what essentially the same idea, is called multiple shooting, because we are not shooting all the way are shooting in a piece, I mean piece ends actually for parts ends about. It also brings addition of constraints of continuity and smoother, so, even though this is in the, remember these are all artificial division actually (()) of the problem (()) is not actually the problem operates from t not to t f in a continuous sense.

And this is the dynamics or non typically, non dynamic constraints, all that we discuss we for the system trajectory is not going to admit any any discontinuous or non smooth trajectory and thing like that, ok? So, it is our duty to enforce that where ever we are splitting the problem, we need to guarantee continuity and smoothness, then that brings in additional constrains at at designing points and this is possible. But I also mean don't keep on blindly applying an (()), just the message then, ok, but, this is the whole idea of multiple shooting, and most of the time this multiple shooting approach is also extended further and further and people talk about direct transition method. We will see all that later in the course, any way this is the whole idea of shooting method as well as multiple shooting actually ok.

(Refer Slide Time: 28:18)



Gradient Method

Prof. Radhakant Padhi

Dept. of Aerospace Engineering

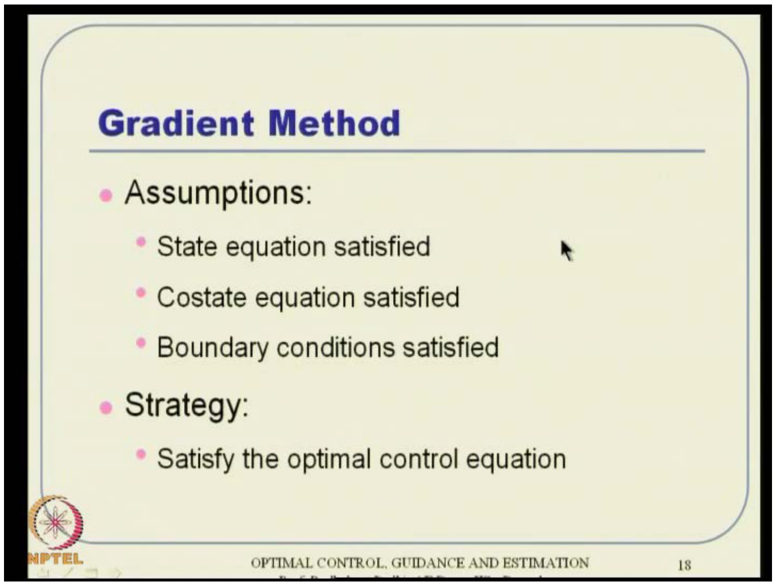
Indian Institute of Science - Bangalore

Now, that's about gradient method. Now, the the second method; remembered we talked about we we intened to discuss 3 methods theory actually, alright?

So, now compared to this, what is happening here I mean, the the whole idea propagate the state equation costate equation forward and in one direction, it ultimately it means. But if you if you see this equation, as we discussed before, ideally the state equation should be integrated forward were as the costate equation should be integrated with (()) and that future is not there in the shooting method actually ok.

Now, gradient method tries to exploit that feature and hence in general gradient method tends out to be more stable actually numerically. Alright, so, let us see what is gradient method d f first of all, ok?

(Refer Slide Time: 29:04)



Gradient Method

- Assumptions:
  - State equation satisfied
  - Costate equation satisfied
  - Boundary conditions satisfied
- Strategy:
  - Satisfy the optimal control equation

OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 18

The procedure is, (()) we start with the initial guess value of the control history and then (()) with respect to the guess the value of the control history. We are assuming that if we if you used that control history as an optimal control solution, I mean in general, then we are assuming that particular guess history or the previously updated, (()) satisfies the three equations actually, (()) satisfy the costate equation, it also satisfies the boundary condition. What it does not satisfy is, the optimal control equation. That's our assumption actually ok?, and because these are assumptions, (()) better of (()) better of (()) value satisfies there (())

Anyway, start with the guess value which will satisfies the equations, ok, and if you start with the state, I mean if you start with the control history, if we start with control history, incase start with some set of control history t not to t f forever okay, we start with a control history and then you integrate the equation state equation anyway right using this control history. So, obviously the state equation is satisfied okay?

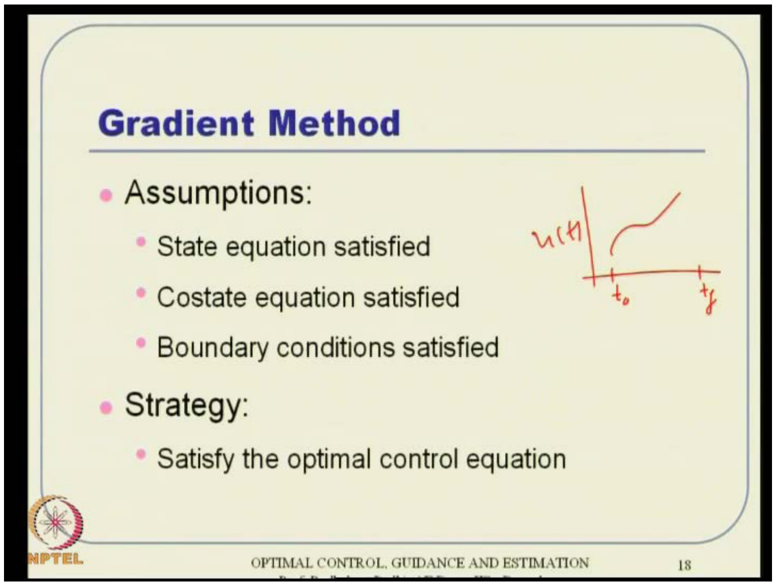
Now, you can calculate some costate to final final costate using that and then, compute to determine, I mean, I mean come back actually. We will start with that final condition in the costate you integrated backward, so costate equation is satisfied and you will start with the initial condition of the state and here, using the final boundary condition of the costate anyway.

So, the costate equations, I mean, the boundary conditions are also satisfied, okay? So so the out coming, what any (()) satisfied is optimum control equations actually, that is the whole idea of the assumption, alright?

So, the procedure is like this; you will guess a control history and then start with initial condition of the state integrated forward okay, and then you calculate the costate using the final condition of the costate lamada equal to del phi by del x that kind of thing and using that lambda f and the costate equations are now integrated backward, okay, and then you also get the entire costate trajectory.

So, now state trajectory and costate trajectory as well as the previous control trajectory are available, okay? The control trajectory, if we may or may not need anymore is depending on the hyteration that we are taking about. So, using all these, okay, we will be able to solve further required control because of using this this state equation of this optimal control equation, ok?

(Refer Slide Time: 31:46)



Gradient Method

- Assumptions:
  - State equation satisfied
  - Costate equation satisfied
  - Boundary conditions satisfied
- Strategy:
  - Satisfy the optimal control equation

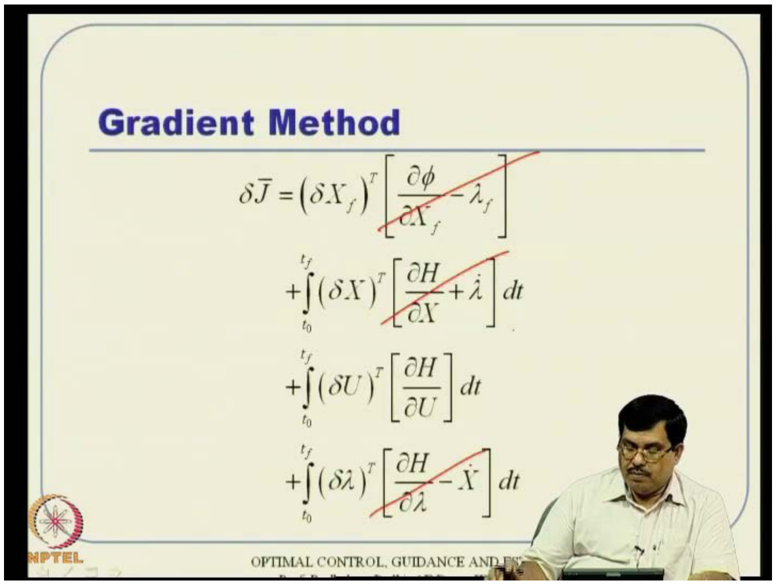
$u(t)$
t_0
t_f
NPTEL
OPTIMAL CONTROL, GUIDANCE AND ESTIMATION
18

This equation, we are using it to integrate forward, this equation okay? Let me summarize again; we are starting with a guess value of the control history anyway, then we are starting with initial condition of the state, okay?

So, using this state and that control whatever guess just control, we can integrate it forward okay? get an x of u and use that x of value to compute this lambda f once again lambda f and the state trajectory is also available also available for the entire decision lambda f. Hence, (()) lambda (()) is a function of (()) only lambda, the explicitly available (()), you can integrate it backwards and get the entire lambda history.

Now, once you have state history and lambda for the entire duration, you can go back and say that this equation can can be escalated further and then you can have error. You can correct that and keep proceeding that way actually, okay, that is the procedure of shooting method actually, sorry gradient methodology alright, so this is the procedure as how you do that basically.

(Refer Slide Time: 32:55)



Gradient Method

$$\delta\bar{J} = (\delta X_f)^T \left[\cancel{\frac{\partial \phi}{\partial X_f} - \lambda_f} \right]$$

$$+ \int_{t_0}^{t_f} (\delta X)^T \left[\cancel{\frac{\partial H}{\partial X} + \dot{\lambda}} \right] dt$$

$$+ \int_{t_0}^{t_f} (\delta U)^T \left[\frac{\partial H}{\partial U} \right] dt$$

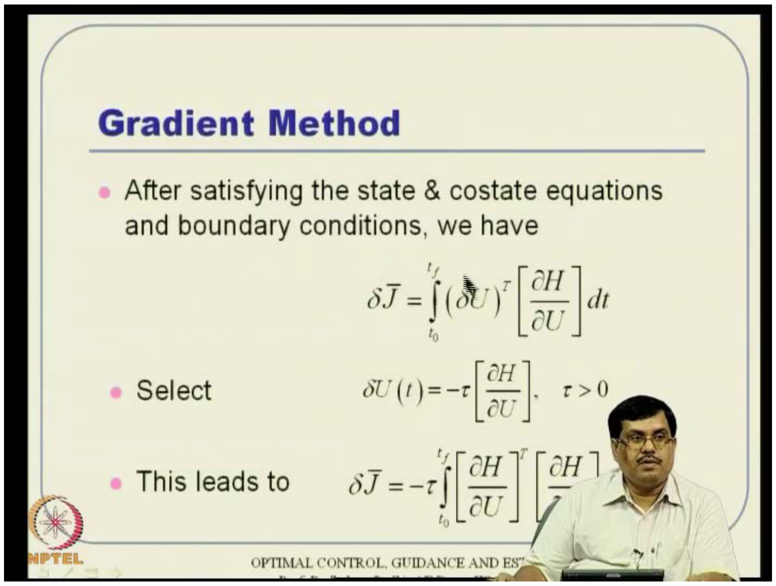
$$+ \int_{t_0}^{t_f} (\delta \lambda)^T \left[\cancel{\frac{\partial H}{\partial \lambda} - \dot{X}} \right] dt$$

NPTEL
OPTIMAL CONTROL, GUIDANCE AND ES

Now, we have to govern our derivation in the last class and then see the first variation of j bar okay; the first variation of j bar is first expressed in this form and hence we get all the necessary conditions right. We got lambda is equal to del phi by del x f, form here we got lambda dot equal to minus del h by del x from here. We got del h by del u equal to zero. Here we got del x dot equal to del h by del lambda here.

So, this is what we are, I mean, not what our first variation of j bar, okay? Now coming back to here, we are assuming that the state equation is satisfied, costate is satisfied as well as the boundary conditions are satisfied. That means this equation is satisfied ok, this equation satisfied okay, this equation satisfied, this equation satisfied, but that's what I have, I mean, that is not satisfied in this one. So, with respect to our, I mean procedure, all these three have automatically satisfied. What is left out is this this portion only, okay, so del j bar is nothing but this, this, this are only visible okay?

(Refer Slide Time: 33:59)



Gradient Method

- After satisfying the state & costate equations and boundary conditions, we have

$$\delta\bar{J} = \int_{t_0}^{t_f} (\delta U)^T \left[\frac{\partial H}{\partial U}\right] dt$$

- Select $\qquad \delta U(t) = -\tau \left[\frac{\partial H}{\partial U}\right], \quad \tau > 0$

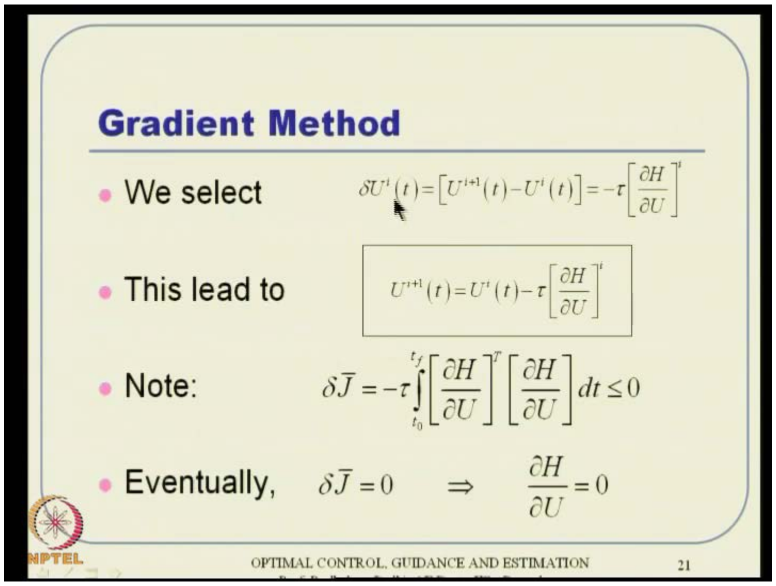
- This leads to $\qquad \delta\bar{J} = -\tau \int_{t_0}^{t_f} \left[\frac{\partial H}{\partial U}\right]^T \left[\frac{\partial H}{}\right.$

OPTIMAL CONTROL, GUIDANCE AND ES

Now, I have to compute a del u trajectory such that this del j bar is used to go towards minimum value basically. So, using this this concept of, I mean the gradient vector, and think like that and natural selection is del u a negative of some some factor tou times del s y del u again okay?

So, ultimately if we selected, that would end del j bar turns out to be like this and hence it is currently to decrease, actually del j bar is guaranteed to be negative, ok? With the whole of this this gradient methods, in even static optimization gives in that before. So, every grid point of time will will compute this this delta u that way and then update this this control trajectory like this.

(Refer Slide Time: 34:51)



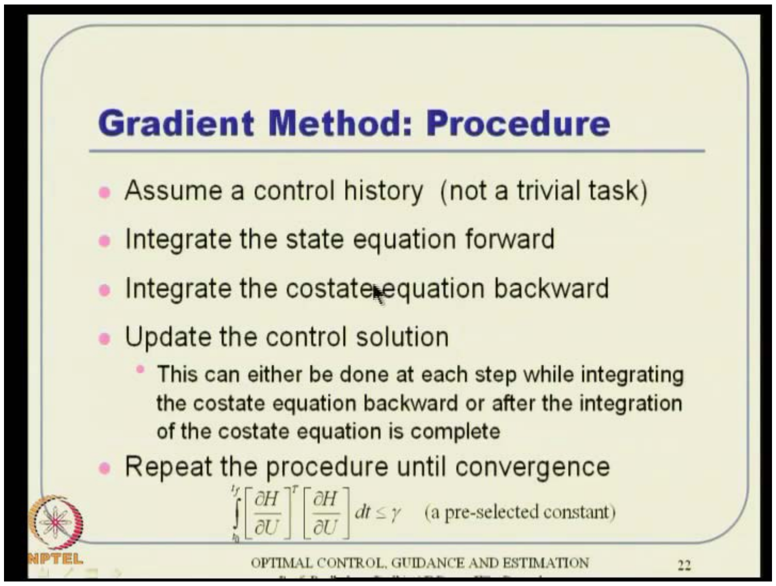
Gradient Method

- We select $\delta U^i(t) = \left[U^{i+1}(t) - U^i(t)\right] = -\tau\left[\frac{\partial H}{\partial U}\right]^i$
- This lead to $\boxed{U^{i+1}(t) = U^i(t) - \tau\left[\frac{\partial H}{\partial U}\right]^i}$
- Note: $\delta\bar{J} = -\tau\int_{t_0}^{t_f}\left[\frac{\partial H}{\partial U}\right]^T\left[\frac{\partial H}{\partial U}\right]dt \le 0$
- Eventually, $\delta\bar{J} = 0 \quad \Rightarrow \quad \frac{\partial H}{\partial U} = 0$

OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 21

The delta u y of t is nothing, but that okay, and that is already available, okay, and delta u I of t we just computed in that way, so, that is nothing but that... So, if I equate it to these 0 then, here, my updated control history is nothing but that okay?, so in a grid point at a time, I'll be able to compute it that way, ok?

Eventually if the procedure operates, it will keep upon decreasing and finally it will become 0 and when it becomes 0 here, del h by del u will also become 0 if this is a quadratic form actually, okay?



Gradient Method: Procedure

- Assume a control history (not a trivial task)
- Integrate the state equation forward
- Integrate the costate equation backward
- Update the control solution
  - This can either be done at each step while integrating the costate equation backward or after the integration of the costate equation is complete
- Repeat the procedure until convergence

$$\int_{t_0}^{t_f}\left[\frac{\partial H}{\partial U}\right]^T\left[\frac{\partial H}{\partial U}\right]dt \le \gamma \quad \text{(a pre-selected constant)}$$

OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 22

This this will be 0 only when the del h by del u equal to 0, that kind of philosophy we can bring in and equally in that okay?, if we are proper at the algorithm this way that del h by del u equal to zero will also be satisfied at the end of end of the procedure basically.

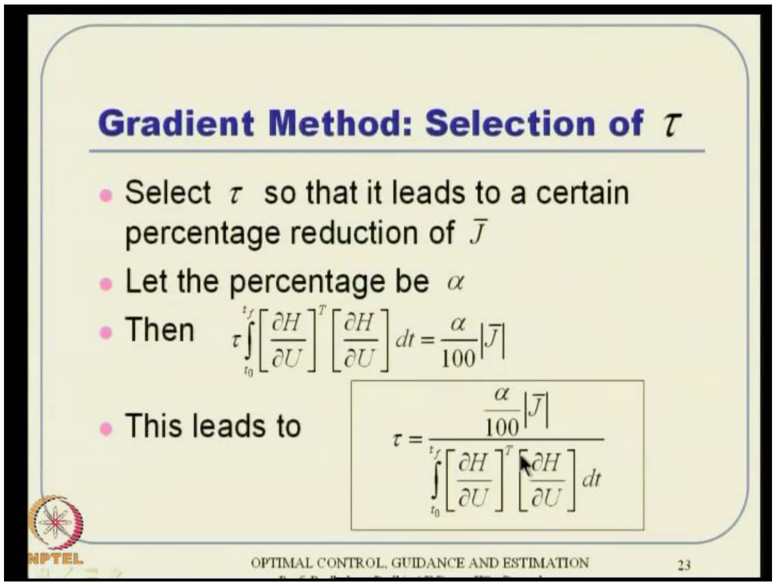
So, this is the summary of the procedure, you can assume a control history, integrate the state equation forward, integrate the costate equation backward and update the control solution and while updating the control solution it can be done in two ways actually. One way is you can integrate it completely backward, okay?

Okay, you can, well, likely to explain if we integrate it completely t f to t not and then update it, okay; but if you really want, I mean t f to t not, you can integrate this lambda equation backward and then at you can go update the control history. That’s that’s the possibility or while coming back, okay, in the backward integration procedure, in every point of time you can update the control, okay, at right there and then integrate it one delta t in backward actually, ok?

Then integrate, then update the control there and with updated control, you can integrate it one more step backward actually. That kind of a thing, you can also implement actually, okay? So, so there are two choices that you have there actually, okay?

Alright, so this is the is is what it summarizes and then you have to repeat the procedure until convergence and then convergence means this here, I mean, estimate this integral value is less than (()), the term selected constant okay, then you can assume that it has convergence to stop actually, okay?

(Refer Slide Time: 37:09)



Gradient Method: Selection of τ

- Select τ so that it leads to a certain percentage reduction of $\bar{J}$
- Let the percentage be α
- Then $\tau \int_{t_0}^{t_f} \left[\frac{\partial H}{\partial U}\right]^T \left[\frac{\partial H}{\partial U}\right] dt = \frac{\alpha}{100}\left|\bar{J}\right|$
- This leads to $\boxed{\tau = \dfrac{\frac{\alpha}{100}\left|\bar{J}\right|}{\int_{t_0}^{t_f} \left[\frac{\partial H}{\partial U}\right]^T \left[\frac{\partial H}{\partial U}\right] dt}}$

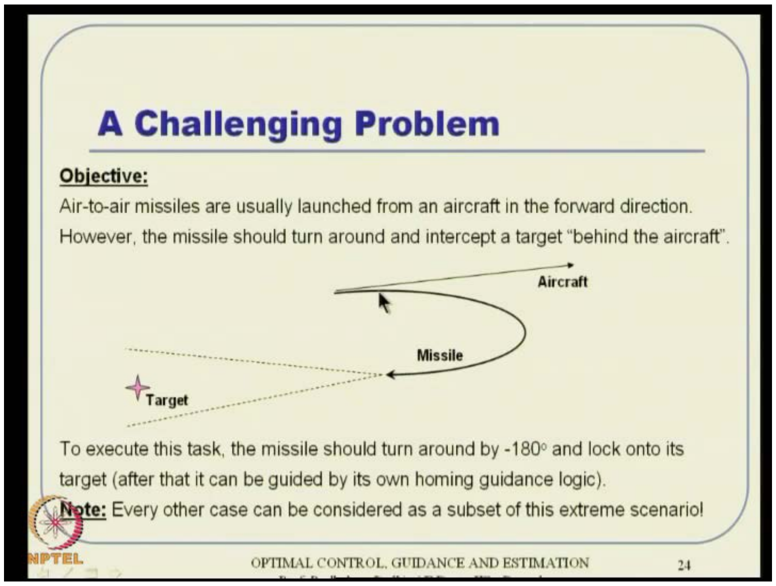
OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 23

When the question is how to select this tou fellow, I mean this tou factor, somebody can select arbitrary, somebody can select one or initially 11, later small value and think rather, that this also is possible. But one way of doing that is (()), and you can assume that should reduce by some percentage alpha the j bar should reduce by certain percentage alpha. I'll select a tou in such a way that'll be result. In that then, I will have the j bar value computed already before, okay ?

So, I i take the take the some factor alpha is a percentage. So, alpha by hundred is that okay, and then, this one what I am looking is tou in to that has to be equal to that.

So, tou can be computed that way, so that way it is a dynamic selected and I mean different different values for (()) iterations which will decide this remains is the pre selected value of some percentage value reduction basically of j bar. I think one, okay, little bit, again I am not emphasizing that every time somebody has to use it this way okay. You can (()) one way of adjusting alpha is (()).

(Refer Slide Time: 38:16)



A Challenging Problem

Objective:

Air-to-air missiles are usually launched from an aircraft in the forward direction. However, the missile should turn around and intercept a target "behind the aircraft".

Aircraft
Missile
Target

To execute this task, the missile should turn around by -180° and lock onto its target (after that it can be guided by its own homing guidance logic).

Note: Every other case can be considered as a subset of this extreme scenario!

OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 24

Further I thought, we can think about some sort of exercise problem and this is actually a challenging problem in a way. So this is an uploading literature, about in 120 years back and 200 evaluate 99 200 early 200, some of that anyway. So, this is the problem of somewhat like air to air missile sort of thing, this is aircraft to carry a missile it give some vane, okay? Typically it can get in finding a missile towards a target informed, okay? So, if your target is actually backward on the back side of the aircraft, the typical thing is you do a (( )) anywhere okay? You kind of integer an angle of an attack and hence enhance the drag drag increase and the speed reduces.

So, you come back okay, and then if you reduce the angle of attack and then and then come down actually okay? If you increase the angle of attack, the lift has increased. So, the aircraft goes up and it should also go a little bit, I mean (()), and then the drag is also more also. So it comes, I mean, I mean it reduces the speed and all.

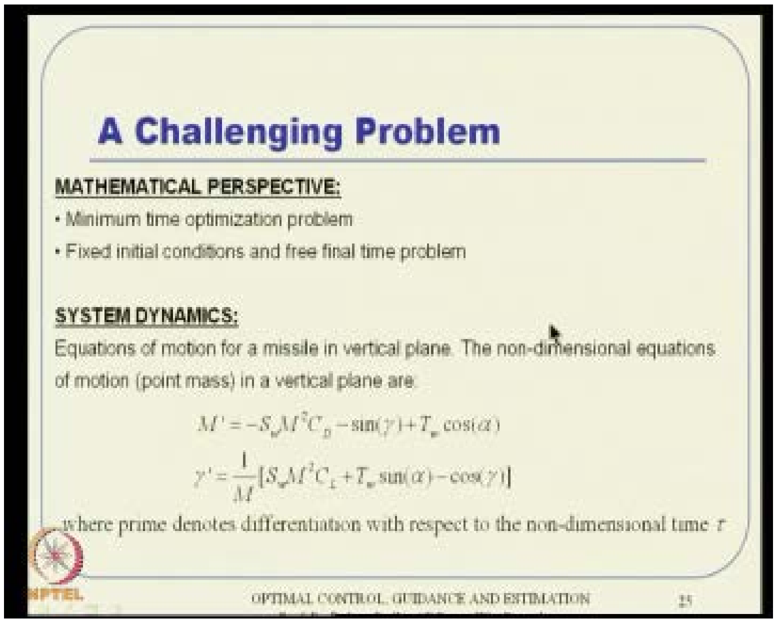
So, it comes back and then you will decrease the angle of attack and then you have the back side of the target and then you will be able to fire it actually. But, in the procedure, there's ther's a lot of, I mean, time is typically in such kind of applications. So, is it possible that you can still fire a missile in the forward direction but the the missile is to turn is possible towards the backward and then enlarge on the target.

So, the problem we are talking about is not enhancement of the target. That's a separate problem; what what we are talking about is, from relase to exactly turning out, I mean, turning in the backward direction okay, by one eighty degrees basically.

Because this may not be exactly um (()) horizontal it can have some some positive angle of attack, I mean it can have some some projective gamma rather, which is the flight path angle it it can go (()) little bit, I mean ascending set, to teach you basically. Then, it can release this is to turn in in is less less time is possibly actually okay, alright, also small convenience where cases can be considered as a subset of this extreme scenario.

If you want to kind of predefine some selected, where many not necessarily one eighty degree, but any where lesser than that, then it's up its substituted that actually mathematical perspective.

(Refer Slide time: 40:45)



A Challenging Problem

MATHEMATICAL PERSPECTIVE:

- Minimum time optimization problem
- Fixed initial conditions and free final time problem

SYSTEM DYNAMICS:

Equations of motion for a missile in vertical plane. The non-dimensional equations of motion (point mass) in a vertical plane are

$$M' = -S_w M^2 C_D - \sin(\gamma) + T_w \cos(\alpha)$$

$$\gamma' = \frac{1}{M}[S_w M^2 C_L + T_w \sin(\alpha) - \cos(\gamma)]$$

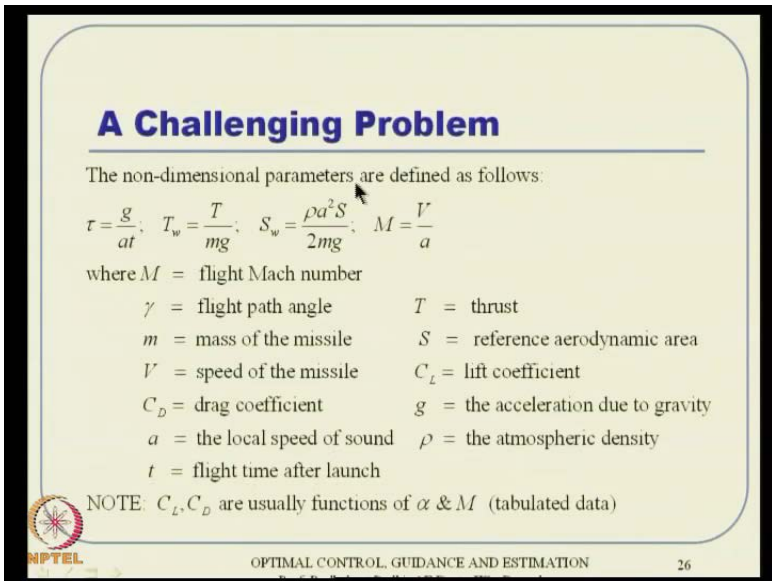
where prime denotes differentiation with respect to the non-dimensional time τ

OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 25

It is actually a minimum time per um um optimization problem, but that is typically difficult and we will say that later. Minimum time problems solving when controls and think either later will see them.

Alright, now the system dynamics associated with that is, this given in this, I mean given like this, where I assume that this, some of few like this, precisely an aero space problem and this m is mach number and then gamma is flight path angle and alpha is angle of attack first is at t w is thrust a number that way. But this also is in a normalize form, okay and there various variables are are given like this, the system dynamics is already in normalized form, okay?,

(Refer Slide Time: 41:23)



A Challenging Problem

The non-dimensional parameters are defined as follows:

$\tau = \frac{g}{at}$; $T_w = \frac{T}{mg}$; $S_w = \frac{\rho a^2 S}{2mg}$; $M = \frac{V}{a}$

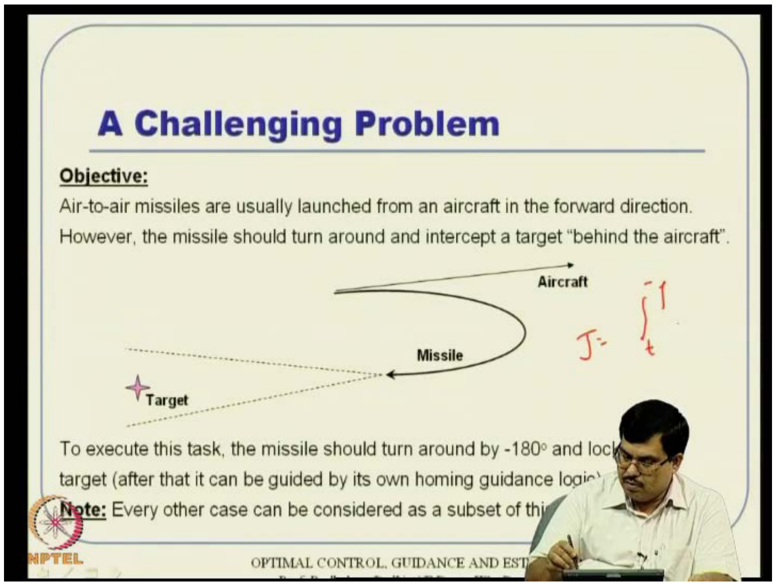
where M =	flight Mach number		
γ =	flight path angle	T =	thrust
m =	mass of the missile	S =	reference aerodynamic area
V =	speed of the missile	C_L =	lift coefficient
C_D =	drag coefficient	g =	the acceleration due to gravity
a =	the local speed of sound	ρ =	the atmospheric density
t =	flight time after launch		

NOTE: C_L, C_D are usually functions of α & M (tabulated data)

OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 26

So, tou is, these are derivatives with respect to tou, the tou is normalize like this, okay? and t w is thrust by weight okay, this kind of normalized thrust sort of thing s w like this am mach number is by definition, normalize velocity of the vehicle divided by sound velocity. So, using this variables okay, you have the system dynamics okay, and then ah ah I mean our objective is to minimize the time taken.

(Refer Slide Time:41:57)



A Challenging Problem

Objective:

Air-to-air missiles are usually launched from an aircraft in the forward direction. However, the missile should turn around and intercept a target "behind the aircraft".

To execute this task, the missile should turn around by -180° and lock target (after that it can be guided by its own homing guidance logic

Note: Every other case can be considered as a subset of thi

NPTEL

OPTIMAL CONTROL, GUIDANCE AND EST

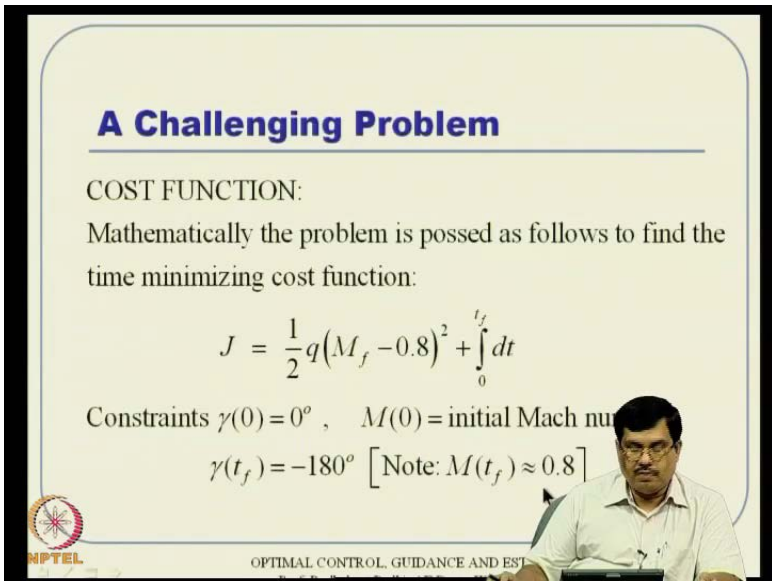
So, minimize the t f minus t 0 sort of thing actually and we have seen that that ah the corresponding cross function, we have seen that j equal to this, we can talk about t 0 2 t 0 2 t f.

Let me (()) into t zero to if one d t sort of thing so t f minus t zero it will become like that okay? So, we can do that, ah and ah but it it is ah done in a little bit different way, a little bit other clever way. Will see that way but, also remember like that, due to certain certain engagement requirement and all, we cannot allow this vehicle, to kind of this missile, to drought velocity fast actually.

So, it is the end of the turning it should also have certain velocity so that it can engage with the target in a good way okay? So, we can remember by while turning, this is also consuming thrust actually and for untill engagement it does not burn thrust is is burn thrust in intial segment, then that procedure has to be remembered.

So, we cannot effort to lose too much of energy because any turning also revolves this induced drag and all that it ah it ah results in ah loss of energy. We cannot afford to do that.

(Refer Slide Time: 43:13)



A Challenging Problem

COST FUNCTION:

Mathematically the problem is possed as follows to find the time minimizing cost function:

$$J = \frac{1}{2}q\left(M_f - 0.8\right)^2 + \int_0^{t_f} dt$$

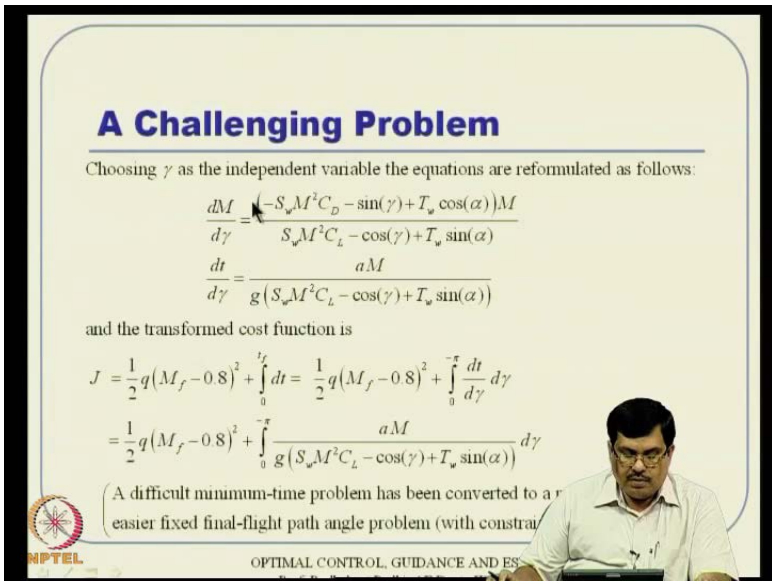
Constraints $\gamma(0) = 0^o$, $M(0) =$ initial Mach nu

$\gamma(t_f) = -180^o$ $\left[\text{Note: } M(t_f) \approx 0.8\right]$

OPTIMAL CONTROL, GUIDANCE AND EST

So, there is a constant for that, m f has to be equal to point eight actually okay, but will instead of that, equally pointed will tell approximately point eight is okay. And then, this kind of a sub constant is alright actually, so you put that is a sub constant it anyway close to point eight is fine actually, thus, the meaning okay, but it goes to an ideal higher values then m f will go closer and closer to point 8 actually and this part will guarantee okay that minimum time is there t f minus 0 that is t f has to be minimum actually okay?

(Refer Slide Time: 43:47)



A Challenging Problem

Choosing γ as the independent variable the equations are reformulated as follows:

$$\frac{dM}{d\gamma} = \frac{\left(-S_w M^2 C_D - \sin(\gamma) + T_w \cos(\alpha)\right)M}{S_w M^2 C_L - \cos(\gamma) + T_w \sin(\alpha)}$$

$$\frac{dt}{d\gamma} = \frac{aM}{g\left(S_w M^2 C_L - \cos(\gamma) + T_w \sin(\alpha)\right)}$$

and the transformed cost function is

$$J = \frac{1}{2}q\left(M_f - 0.8\right)^2 + \int_0^{t_f} dt = \frac{1}{2}q\left(M_f - 0.8\right)^2 + \int_0^{-\pi} \frac{dt}{d\gamma} d\gamma$$

$$= \frac{1}{2}q\left(M_f - 0.8\right)^2 + \int_0^{-\pi} \frac{aM}{g\left(S_w M^2 C_L - \cos(\gamma) + T_w \sin(\alpha)\right)} d\gamma$$

(A difficult minimum-time problem has been converted to a r
easier fixed final-flight path angle problem (with constrai

OPTIMAL CONTROL, GUIDANCE AND ES

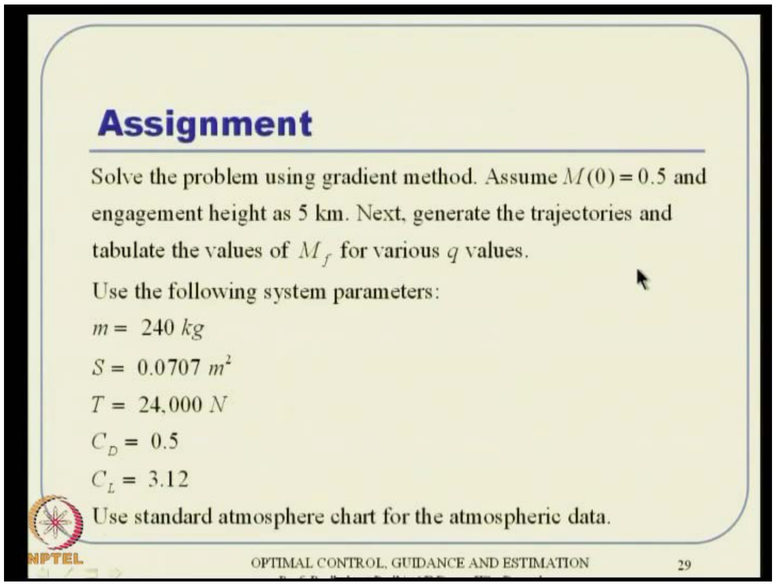
But this is done in a little bit clever way. You can first (()). This is, instead of dou is independent variable you can think of gamma is an independent variable. First write b m by d gamma d t by d gamma like that and remember this ah one d t can be thought about something likes this. d t is nothing but d t by d gamma into d gamma and d t by d gamma is nothing but one by, I mean, this d t by d gamma is available here okay?

So, this equation, this d t by d gamma, this expression can be substituted here okay, and then this ah m f minus point eight and all that is anyway available. So, ultimately because this equation comes to the cross function, where as this equation becomes the system dynamically actually.

So, we have a cross function in this form, subject to one scalar differential equation ever independent variable is gamma actually. Okay, so this kind of a scalar sort of problem, we can think about where m is the only state vector and gamma evolves from zero to minus five, and this expression has already come here to minimum, I mean, in the in the cross function minimization part of d t it is already utilized.

So, sincerely you will minimize this cross function subject to this first differential equation actually, okay? alright

(Refer Slide Time: 45:01)



Assignment

Solve the problem using gradient method. Assume $M(0) = 0.5$ and engagement height as 5 km. Next, generate the trajectories and tabulate the values of M_f for various q values.

Use the following system parameters:

$m = 240\ kg$

$S = 0.0707\ m^2$

$T = 24{,}000\ N$

$C_D = 0.5$

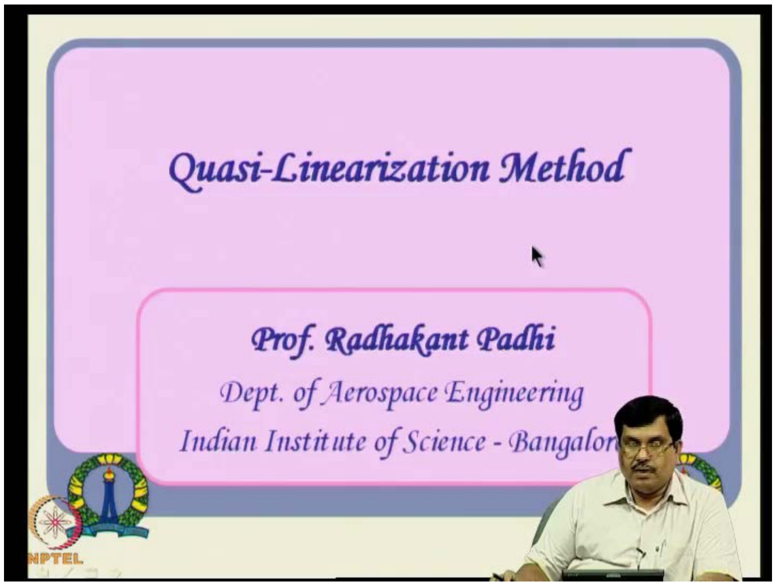
$C_L = 3.12$

Use standard atmosphere chart for the atmospheric data.

OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 29

So I will leave that here. This problem, it does mention seriously all of you to attempt this okay, using some of this typical numbers actually. You can see see some of this parameter numbers, you can use some standard atmospheric data at the altitude of 5 kilo meter. You have to select this ah this atmospheric density okay, and then proceed with all this numbers and then generates some results using ah this ah gradient method actually, alright?

(Refer Slide Time: 45:28)

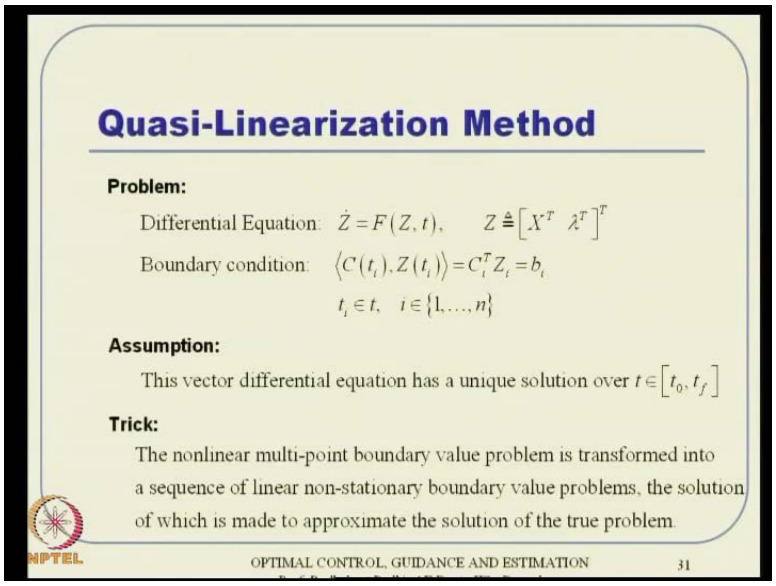


Quasi-Linearization Method

Prof. Radhakant Padhi
Dept. of Aerospace Engineering
Indian Institute of Science - Bangalore

So, the last technique that in this class I want to to talk about is this quassi linearization. so Completely different approach again and this also (( )) beauty actually alright.

(Refer Slide Time: 45:41)



Quasi-Linearization Method

Problem:

Differential Equation: $\dot{Z} = F(Z, t)$, $\quad Z \triangleq \left[X^T \;\; \lambda^T\right]^T$

Boundary condition: $\langle C(t_i), Z(t_i) \rangle = C_i^T Z_i = b_i$

$t_i \in t, \quad i \in \{1, \ldots, n\}$

Assumption:

This vector differential equation has a unique solution over $t \in \left[t_0, t_f\right]$

Trick:

The nonlinear multi-point boundary value problem is transformed into a sequence of linear non-stationary boundary value problems, the solution of which is made to approximate the solution of the true problem.

OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 31

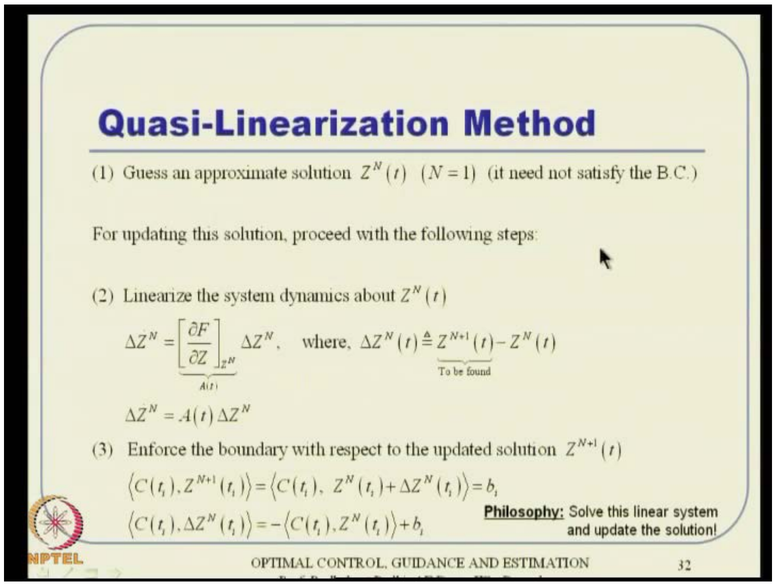
So let us see that. I'll not talk about optimal control problem in general here, but we will ah increase the, I mean general, it is in will in increase the general little little more actually, okay? Alright, so the differential equation in general, we can talk about like this Z dot equal to half F of Z t where Z is nothing but X and lambda it taken together actually, okay, boundary condition, it is, this includes ah a set of boundary conditions in different parts of time. Okay, if can be only reasonable final time, if can also be something on the way as well basically.

So, these kinds of conditions are also possible. For example (( )), vehicle is there for going to moon then you just go to your parking orbit and then go to the moon actually. Alright, so that kind of thing if can be as is split either artificially or naturally, whatever be actually.

So, the problem that we are talking about is this ways Z dot of Z t, where the boundary equations are given that actually and we also missile that this vector differential equation as a unique solution over the time domain. So, the trick here is this ah non linear multi point boundary value problem now is not necessarily only 2 point valuable problem

This non linear multipoint multi point boundary value problem, which transform into a sequence of linear non stationary boundary value problems, okay, and the solution will be found from there and than if it will be approximately the solution of the true problem actually.

(Refer Slide Time: 47:17)



Quasi-Linearization Method

(1) Guess an approximate solution $Z^N(t)$ $(N=1)$ (it need not satisfy the B.C.)

For updating this solution, proceed with the following steps:

(2) Linearize the system dynamics about $Z^N(t)$

$$\Delta \dot{Z}^N = \underbrace{\left[\frac{\partial F}{\partial Z}\right]_{Z^N}}_{A(t)} \Delta Z^N, \quad \text{where, } \Delta Z^N(t) \triangleq \underbrace{Z^{N+1}(t)}_{\text{To be found}} - Z^N(t)$$

$$\Delta \dot{Z}^N = A(t)\,\Delta Z^N$$

(3) Enforce the boundary with respect to the updated solution $Z^{N+1}(t)$

$$\left\langle C(t_i), Z^{N+1}(t_i)\right\rangle = \left\langle C(t_i),\ Z^N(t_i) + \Delta Z^N(t_i)\right\rangle = b_i$$

$$\left\langle C(t_i), \Delta Z^N(t_i)\right\rangle = -\left\langle C(t_i), Z^N(t_i)\right\rangle + b_i$$

Philosophy: Solve this linear system and update the solution!

OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 32

So, how do how does it proceed; will ah guess an approximate solution first ok, that in invariably there is a requirement. So, you guess an approximate solution Z n of t N equal to 1 to begin with and then we have to update the solution and to update the solution, we have to proceed this way actually okay?

About this guess value, what about, we have Z n t we can always write a linearize system dynamics and that terms to be like this, where by definition delta z n is nothing but Z n plus 1 minus z n actually. This is available with us either as a guess or as a previous solution. This has to be found actually, okay? So, this is like (( )) certain method this is, like a time baring matrix basically.

So, you can always write this way um. Then, we have to enforce the boundary conditions with respect to the updated solution, even the guess solution mean may not, the but while you update the solution. Our aim is that we should update in such a way that the updated solution satisfies the boundary condition actually.

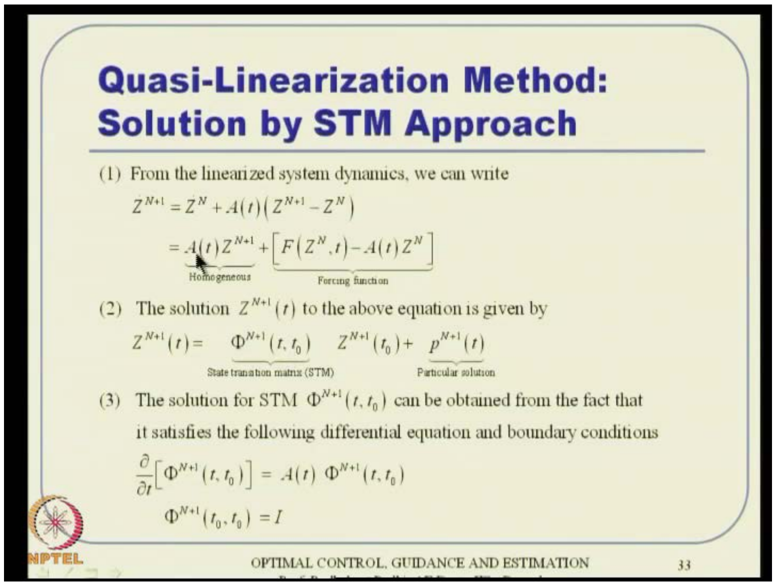
So, how do you do that? Because the, remember this is the boundary condition, so in this boundary condition, I will substitute z n plus 1 here and then I will see okay, c of t I z n plus 1 ah t I okay, no problem as to equal to b I, thus the condition.

But now, z n plus one, I can divide into that this two because that is by definition in like that, Okay? So, I can write that ah this enough dou t nothing but minus (( )) dou plus v I. we can use split out this this enough product of a and b plus c is nothing but n product of a b plus enough product of a C.

So, using that, I will able to split that and then I will be able to write that ah C I C or t I and Z n of t I this enough, dou t I nothing but z n cross v I anyway. So, what you where are you hearing actually, okay, where is that will be apparently? (( )) actually.

So, what is the what is it that from this differential equation Z delta Z n ah dot equal to these will be able to write this way delta Z n dot is nothing but z n plus one dot minus z n dot So, z n plus one dot is nothing but z n dot plus this equation right this a t time delta z n delta z n is this one by definition. So, I will substitute these 2 and it will right of that way.

(Refer Slide Time: 49:48)



Quasi-Linearization Method: Solution by STM Approach

(1) From the linearized system dynamics, we can write

$$Z^{N+1} = Z^N + A(t)\left(Z^{N+1} - Z^N\right)$$

$$= \underbrace{A(t)Z^{N+1}}_{\text{Homogeneous}} + \underbrace{\left[F\left(Z^N, t\right) - A(t)Z^N\right]}_{\text{Forcing function}}$$

(2) The solution $Z^{N+1}(t)$ to the above equation is given by

$$Z^{N+1}(t) = \underbrace{\Phi^{N+1}(t, t_0)}_{\text{State transition matrix (STM)}} \; Z^{N+1}(t_0) + \underbrace{p^{N+1}(t)}_{\text{Particular solution}}$$

(3) The solution for STM $\Phi^{N+1}(t, t_0)$ can be obtained from the fact that it satisfies the following differential equation and boundary conditions

$$\frac{\partial}{\partial t}\left[\Phi^{N+1}(t, t_0)\right] = A(t)\;\Phi^{N+1}(t, t_0)$$

$$\Phi^{N+1}(t_0, t_0) = I$$

NPTEL

OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 33

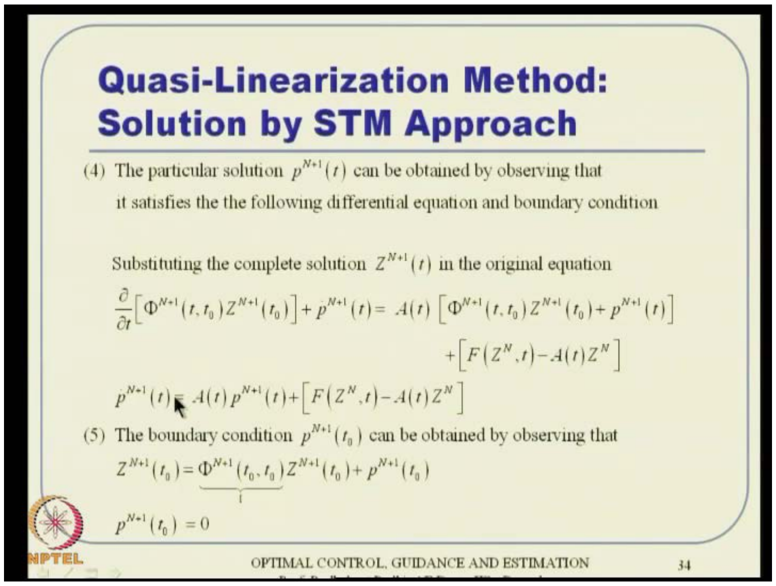
So, I few can interrupt this this z n z n plus one dot it has this homogeneous part a of t times Z n Z n plus one plus follows this forcing function actually which is already available with us ok. So, previous guess is to previous solution, is available with ah or previous solution or guess solution is available with us.

So, this is the forcing function and this a homogenous equation. So, if you really want to solve, then we need the state transition matrix of the mean. So, this and this solution, ah because it is a time varying matrix, the newest state transition matrix ok, first a particular solution because there is a forcing function ok.

Now, the straight line transition matrix as we know, it is it is satisfies the same differential equations with this boundary condition. So, this is this is clear we can ah studies for this initial, this differential equation with this boundary condition.

So, we can integrate it and then get some up grated values and all that so this value will be available with us. Now, that we have value value of this one actually, so, let us see how to get that for getting the particular solution. You take the entire solution, full solution, and substituted break in the original differential equation ok, and and if we do that, this is the what you are getting

(Refer Slide Time: 51:01)



Quasi-Linearization Method: Solution by STM Approach

(4) The particular solution $p^{N+1}(t)$ can be obtained by observing that it satisfies the the following differential equation and boundary condition

Substituting the complete solution $Z^{N+1}(t)$ in the original equation

$$\frac{\partial}{\partial t}\left[\Phi^{N+1}(t,t_0)Z^{N+1}(t_0)\right]+\dot{p}^{N+1}(t)= A(t)\left[\Phi^{N+1}(t,t_0)Z^{N+1}(t_0)+p^{N+1}(t)\right]$$
$$+\left[F(Z^N,t)-A(t)Z^N\right]$$

$$\dot{p}^{N+1}(t)= A(t)p^{N+1}(t)+\left[F(Z^N,t)-A(t)Z^N\right]$$

(5) The boundary condition $p^{N+1}(t_0)$ can be obtained by observing that

$$Z^{N+1}(t_0)=\underbrace{\Phi^{N+1}(t_0,t_0)}_{I}Z^{N+1}(t_0)+p^{N+1}(t_0)$$

$$p^{N+1}(t_0)=0$$

OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 34

This is the z dot sort of thing ok, the z dot is nothing but z n plus one dot and that is nothing but with this of kind of things actually. ok I mean this this entire thing basically, so, if we this one z n plus one dot is nothing but with this for the whatever comes out from here and remember this is a number, is not a function, but this is a function and this is also a function.

So, using that, we will be able to write this left side of the thing, ah right side of the thing is anywhere there this one plus this one ok, so I am writing that.

Now what is happening here? This equation dealing satisfied here, this here state transitions matrix will ah will satisfies this equation anyway. So this will get cancelled out actually, the first term will be cancelled out.

So, you left out with this term in the left hand side where in the right hand side we contain this one into that one here actually ah sorry, and this into the entire thing here actually ah sorry, this is this is only for that, for this one into this one which coming here and plus this one in the always. So this is the how we get for the the differential equations for this, for that is not everything, we will also need a boundary condition for that ok?

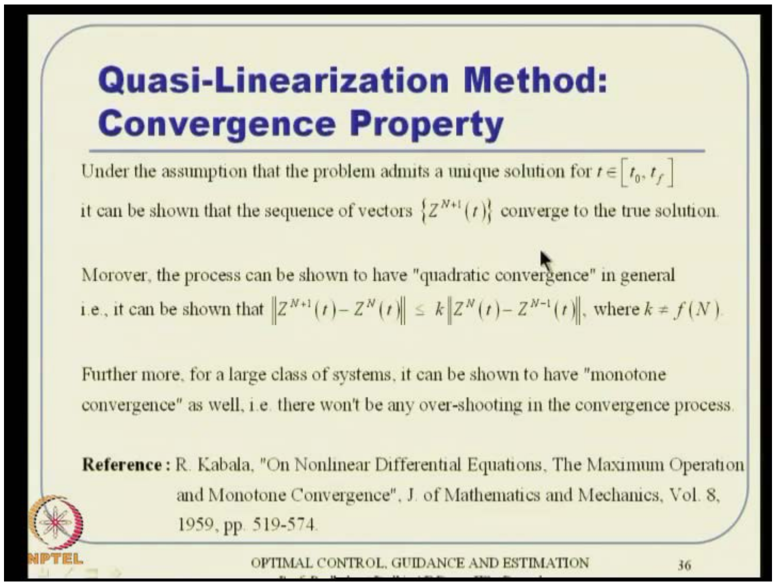
So, how do we complete that actually? So, that can be done this way but vary to the this z n plus t zero ok, is given like this ok, it can be here and we substitute zero. Once we substitute to zero, this we can identity ok, and their this is equal this, become z n plus t zero this is z n plus t zero, this will be cancelled out and this will be left out.

So, this left out, as we left out, so that is the way get it (()). So, p n plus t zero anything is zero and p n plus one dot is available from this equation ok? So, this differential equation and this boundary condition if I integrate, I will get p p n plus one t as well. Actually this one ok and I i if I solve this equation, I will be this one so now the solution is available basically (()) and it equals the boundary condition, for this can can be obtained in this way. Then, we can put it the boundary condition is the real delta z is like this the way

So, you use this way and (( )) and then ultimately we will also something like this actually, this is where the solution, we can put it here and then expand this in order to (( )) so to solve this, we will get initial condition for Z n plus 1 actually. So, then you can, ah we can integrate this equation, I mean, sorry, we need this Z n plus 1 t 0 to get some get value basically. So this z n plus 1 t 0 if we know, then if we know this and know this, then you done z n plus t 0 you have tell this way from the boundary conditions ok?

Now, this one will obtain to this, and this one will obtain through this differential equation, with this boundary condition ok? That is how we get the solution like this actually. Alright this is the summary of this quasi linearization method actually.

(Refer Slide Time: 54:05)



Quasi-Linearization Method: Convergence Property

Under the assumption that the problem admits a unique solution for $t \in [t_0, t_f]$ it can be shown that the sequence of vectors $\{Z^{N+1}(t)\}$ converge to the true solution.

Morover, the process can be shown to have "quadratic convergence" in general i.e., it can be shown that $\|Z^{N+1}(t) - Z^{N}(t)\| \le k \|Z^{N}(t) - Z^{N-1}(t)\|$, where $k \ne f(N)$.

Further more, for a large class of systems, it can be shown to have "monotone convergence" as well, i.e. there won't be any over-shooting in the convergence process.

Reference : R. Kabala, "On Nonlinear Differential Equations, The Maximum Operation and Monotone Convergence", J. of Mathematics and Mechanics, Vol. 8, 1959, pp. 519-574.

OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 36

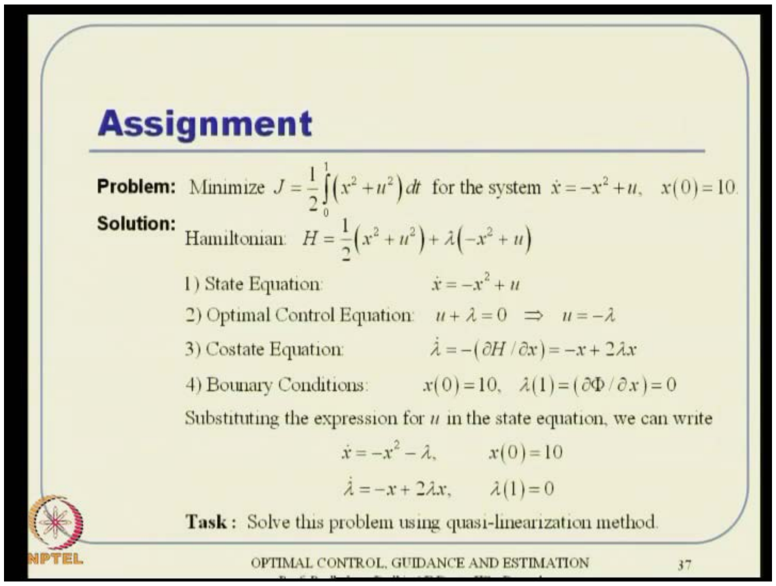
Now, this method is good under certain assumption, so ah which are not that as actually in the can be shown that the sequence, what you have getting it does converge to the two solution, and not only does not only converges but also it converges at that something called quadratic convergence property and where it can be shown that this property is good actually. ok

So, that means it converges fast actually, huh uh here new ton law of method of root finding typically, for satisfies this quadratic convergence. So, these kind of routine also satisfies this quadratic convergence property property actually, or in that you can ah you can see this actually here.

So, this also is very interesting, but we are further more for a large class of system. It can also be shown that it converges monotonically, it doesn't over shoot and then come back and convergence all of z convergence from one side only actually, which is even measured actually.

So, even though the method is since a little bit more complicated and all that, in that still (( )) that you can think of using this and then (( ))...

(Refer Slide Time: 55:08)



Assignment

Problem: Minimize $J = \frac{1}{2}\int_0^1 \left(x^2 + u^2\right) dt$ for the system $\dot{x} = -x^2 + u$, $x(0) = 10$.

Solution: Hamiltonian: $H = \frac{1}{2}\left(x^2 + u^2\right) + \lambda\left(-x^2 + u\right)$

1) State Equation: $\dot{x} = -x^2 + u$

2) Optimal Control Equation: $u + \lambda = 0 \Rightarrow u = -\lambda$

3) Costate Equation: $\dot{\lambda} = -(\partial H / \partial x) = -x + 2\lambda x$

4) Bounary Conditions: $x(0) = 10$, $\lambda(1) = (\partial \Phi / \partial x) = 0$

Substituting the expression for u in the state equation, we can write

$$\dot{x} = -x^2 - \lambda, \qquad x(0) = 10$$
$$\dot{\lambda} = -x + 2\lambda x, \qquad \lambda(1) = 0$$

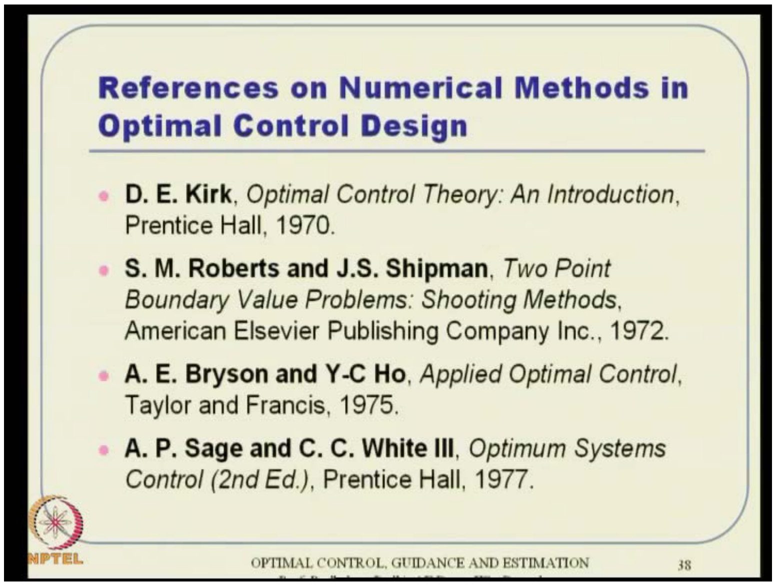
Task : Solve this problem using quasi-linearization method.

OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 37

So, this is the corresponding small assignment problem in which we have as scalar problem, with quadratic gauss function and these are the some necessary condition and all that. So, ultimately end up this taken this costate equation in this form because, control equation has to be eliminated as a function form minus one and you have the 1 equation as well so 0 the task is to solve this problem using quasi linearization method.

So, I seriously suggest all you to kind of ah go to convergertive programming and then try to core this ah the algorithms and then we have a feeling of ah how this now this algorithm operates of actually.

(Refer Slide Time: 55:46)



References on Numerical Methods in Optimal Control Design

- **D. E. Kirk**, *Optimal Control Theory: An Introduction*, Prentice Hall, 1970.
- **S. M. Roberts and J.S. Shipman**, *Two Point Boundary Value Problems: Shooting Methods*, American Elsevier Publishing Company Inc., 1972.
- **A. E. Bryson and Y-C Ho**, *Applied Optimal Control*, Taylor and Francis, 1975.
- **A. P. Sage and C. C. White III**, *Optimum Systems Control (2nd Ed.)*, Prentice Hall, 1977.

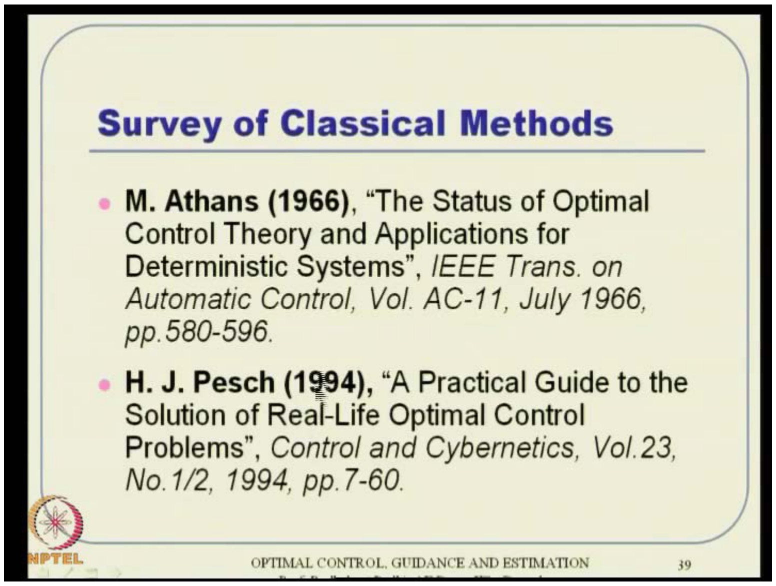
NPTEL OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 38

Some of the references for numerical methods I have taking from this, and lastly, this is this gradient method and quasi linearization are taken from this book where very beautiful for classical solution of optimal optimal control problems.

So, this is way then all other things are available for ah and then for especially the shooting method of they consist this let this actually good book also the tow point boundary value problems it actually So, the various various other thing that the book talks, book meant for that actually.

Now, we as I told in the very beginning class Bryson and y-c ho is always a standard book you can refer to, many of this used, sage and white is also the book

(Refer Slide Time: 56:33)



Survey of Classical Methods

- **M. Athans (1966)**, "The Status of Optimal Control Theory and Applications for Deterministic Systems", *IEEE Trans. on Automatic Control, Vol. AC-11, July 1966, pp.580-596.*
- **H. J. Pesch (1994),** "A Practical Guide to the Solution of Real-Life Optimal Control Problems", *Control and Cybernetics, Vol.23, No.1/2, 1994, pp.7-60.*

OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 39

And there are some like this quizzical and very good ah class valident papers also rather, is a very old paper and this is not the tool that is also related old, but these are very well written papers. So, we can see lot of these ideas there in this lecture next I will wanted to talking this this particular class Thanks a lot.