

Optimal Control Guidance and Estimation

Prof. Prof. Radhakant Pandi

Department of Aerospace Engineering

Indian Institute of Science, Bangalore

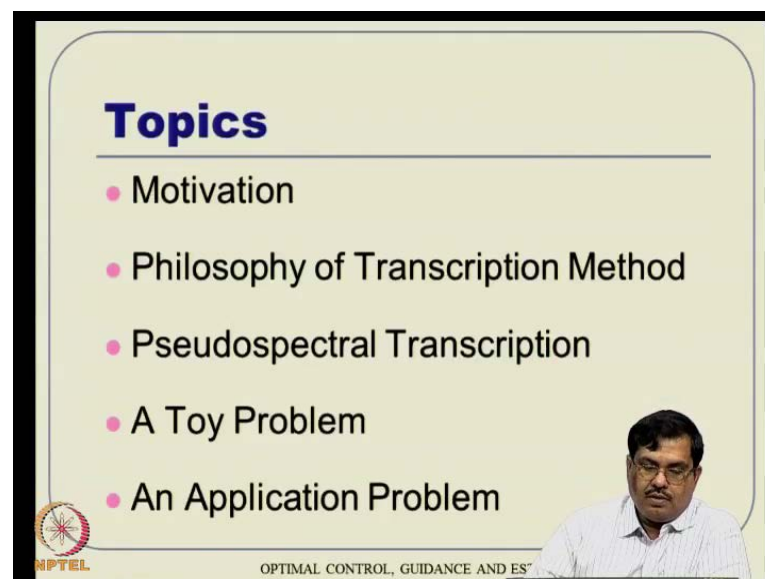
Lecture No. #22

Transcription Method to Solve Optimal Control Problems

Hello everybody, we will continue with the lecture number 22. Where, we will talk about this different idea of solving Optimal Control Problems, which is known as Transcription Method. And followed by generic philosophy will cover, not too much detail into that, and followed by will go to a very recently evolving concept called pseudo spectral transcription is essentially it borrows the idea from the transcription method, but it is a different way of putting the problem in the frame work.

So, the solution can be very fast, and hence the play music can do in real time **actually**, so that is very recent thing. So, will see the **the** concept first, what is transcription method and then slowly migrate to that and follow up with some **some** problem solutions and things like that.

(Refer Slide Time: 01:00)



The slide is titled "Topics" and lists five bullet points: Motivation, Philosophy of Transcription Method, Pseudospectral Transcription, A Toy Problem, and An Application Problem. It includes the NPTEL logo, the course title "OPTIMAL CONTROL, GUIDANCE AND ESTIMATION", and a small video inset of the professor in the bottom right corner.

Topics

- Motivation
- Philosophy of Transcription Method
- Pseudospectral Transcription
- A Toy Problem
- An Application Problem

NPTEL

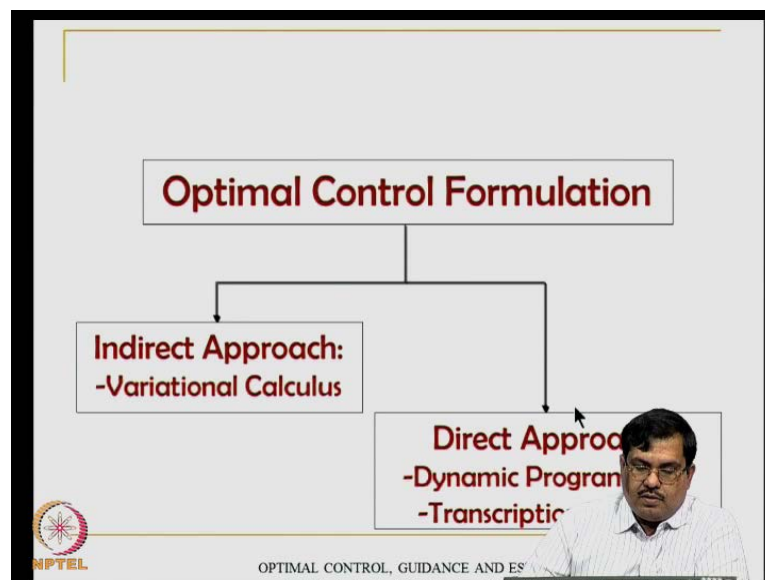
OPTIMAL CONTROL, GUIDANCE AND ESTIMATION

The topics are like this first, we will see some motivation and then, philosophy of transcription method, which is very different from well what I am talking here is **is** direct

transcription **actually**. So, this direct transcription is slightly different from what we have seen before, and then followed by this **this** pseudo spectral transcription.

We will demonstrate the idea pseudo spectral using a toy problem first then, we will go to a real application problem and see how this is useful and all that, then remember there are several, several, several problems, which have been solved by the different way of authors, recently using pseudo spectral and all that. So, we will not go too much into **into** that and I will list out a few references also for your further reading basically.

(Refer Slide Time: 01:41)



So, let us see the motivation first and in a generic view in a global view optimal control formulation can be thought of something like this, one is indirect approach, one is direct approach. An indirect approach we **we** studied something like variation calculus approach, where it leads to this two point revery problem and all that. Whereas, in the direct approach, we have this **this** dynamic programming which we have already studied and then we will have this **this** transcription method, which are going to study today **actually**.

(Refer Slide Time: 02:15)

Necessary Conditions of Optimality through Variational Calculus (Dualization)

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

NIPTEL OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 5

So, this **this** necessary conditions of optimality through variation calculus, if you see this is the idea that **that** is developed **actually**. First we have this idealization of the problem; that means, we have this state equation but, along with this state, now we assume another costae equation. Another set of equations with the same dimension like lambda dot and all that. So, we have this state equation along with that, is the costate equation, is optimal control equation, which are stationary equation **actually**.

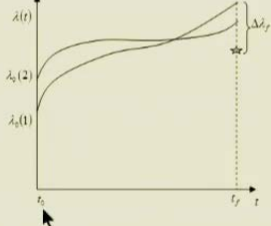
If you solve, it will get the solution of control as a function of state and costate, when associated there are boundary conditions and think like that **actually**. There is initial condition for the state, in the final condition, boundary condition either it is given for the state or it comes to this transversality conditions and think like that **actually**.

Now, the question is, we have gone through this difficulties and all that **actually** that state equation develops forward, costate equation backward, it is a split boundary condition problem. Hence it a two point revelry problem all sought of thing **actually**.

(Refer Slide Time: 03:17)

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



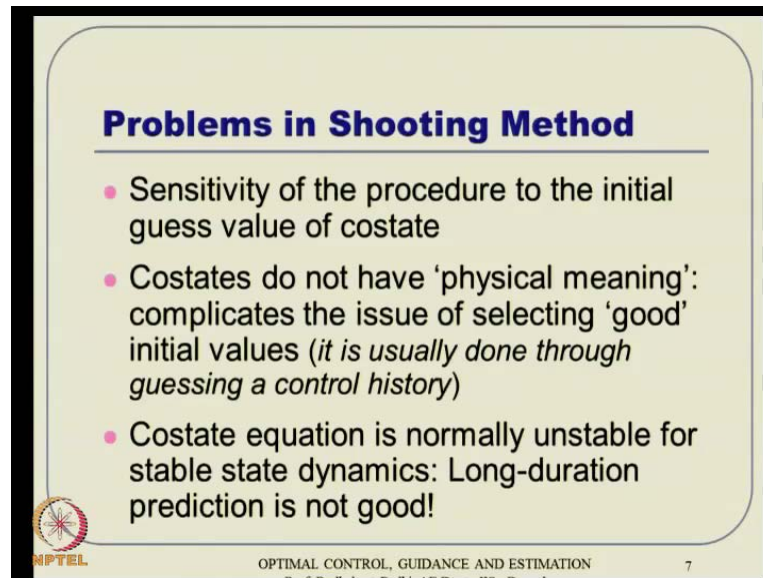
NPTEL OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 6

Now, the shooting method is one more **one more** method before we talked about gradient method and all that. But, shooting method is something like **something like** this, the whole idea here is we have this initial condition of the costate, which is not known.

So, we guess the costate initial condition of the costate, now if you have initial condition of the costate. Then state and costate equation can **can** be propagated forward together with respect to the guess costate value at in all. And then if you see when t goes to t_f your final value of lambda is somewhere here, **somewhere here** whereas the **the** boundary condition will give you some something else; let us say that is here.

So, utilizing this error, whatever error is happen now you update this initial condition lambda 1 to let us say lambda 0 1 to lambda 0 2. Now, repeat the procedure, now with the next iteration it will go little closer to that and again you can update further it will go further close to that, and then finally, converse here **actually**, that is the whole idea here **ok.**

(Refer Slide Time: 04:13)



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!

NPTTEL OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 7

So, if you do that then, what happens is the then I mean the difficulties, because typically costate equation happens to be unstable equation, and here integrated equation forwards. So, we have this the sensitivity issue; that means, is sensitive to the procedure and essentially to the initial guess value of the costate **actually**. And the typically the further difficulties, we cannot have a good guess of the costate.

Because, costate is typically do not have physical meaning; even though there is little bit mathematical meaning that costate is nothing but the optimal if you have optimal cause already. Then dynamic programming sense λ is $\frac{\partial J}{\partial x}$. So, gradient vector of optimal cause that does not help too much in guessing a number for the initial condition for the costate **actually**.

So, typically it is usually done through kind of guess a control history, some realistic control non optimal stabilizing control you use. And then use that predict it (Refer Slide Time: 05:07) I mean use that control from the initial condition of the state, both in the state sense, remember these are costate trajectory what you will operate on the state. And finally, you will get final state and once you get a final state you can evaluate this and then come back **come back** integrate the costate equation backward.

And hence you get some sought of a information about λ_0 , typically it is on that way. So, whole problem is costate equation is normally unstable and hence the long duration prediction is not good. If anywhere these are errors here, anywhere there is if

you not **not** very close to the real optimal $\lambda = 0$. Then we will have this difficulty of error amplification very fast, because the dynamic equation is unstable **actually**. So, typically non long duration prediction is not good.

(Refer Slide Time: 05:54)

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"
- It brings in additional constraints of continuity and smoothness at the 'joining points'.
- Extension of this philosophy leads to "Transcription Method": A Direct Approach!

NPTEL OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 8

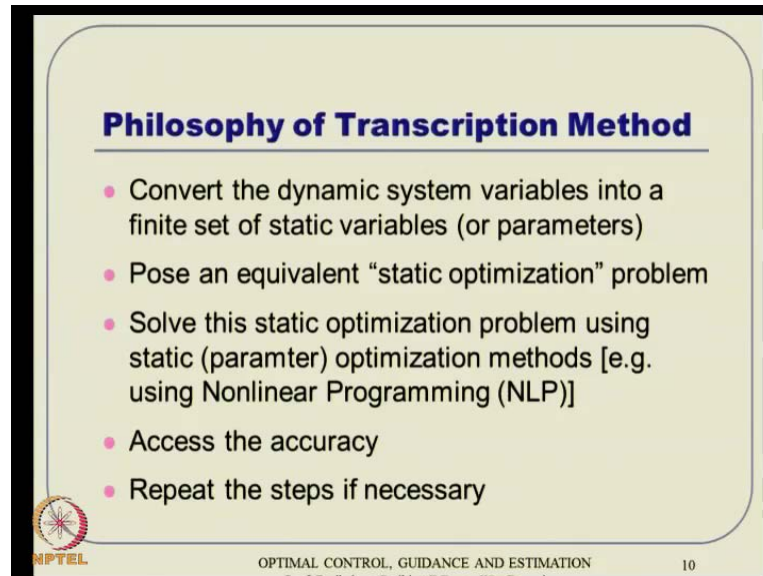
So, what you do? We have to do this idea of divide and rule; that means, instead of predicting for a very long time, how about predicting for a shorter duration. Let us, say t_0 to t_f I will divide into two parts t_0 to $t_{\text{intermediate}}$ and then $t_{\text{intermediate}}$ to t_f something. So, this approach is called something called multiple shooting, you have split that duration into several segments like this and but, when you do that it brings the additional constraint that around this point.

You have to make sure that this discontinuity does not come, I mean this point kind of falls the same point as well as the derivative smoothness should also be there. The derivative here should be equal to the derivative there at least the first you derivate **actually ok**.

So, these are **these are** some of the additional constraints that will put into the problem **actually**. So, these are the additional constraint at this joining points **actually**, but having said that you put this conditions in type, and try to solve it and think like that it essentially leads to this **this** concept of transcription method is kind of a direct approach. In other word if you think **think** about kind of segmenting it to a large number of segments **actually**, then essentially it goes down to some sought of a discretized version

of the problem and hence that is the approach of what is called as direct transcription method **actually**.

(Refer Slide Time: 07:14)



Philosophy of Transcription Method

- Convert the dynamic system variables into a finite set of static variables (or parameters)
- Pose an equivalent “static optimization” problem
- Solve this static optimization problem using static (parameter) optimization methods [e.g. using Nonlinear Programming (NLP)]
- Assess the accuracy
- Repeat the steps if necessary

NPTEL OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 10

So, this is philosophy of direct transcription essentially it **it** convert the dynamic system variable into a final set of **set of** static variables or parameters, and then pose an equivalent static optimization problem. And then solve this static optimization problem using this **this** static optimization methods and essentially there are methods for this, so called non-linear optimization or non-linear programming basically.

So, use that then solve the problem for the static optimization frame work **actually**. Then obsessively you have to assist the accuracy and repeat the steps if necessary and then there are concepts like fine tanning the grid points something like that. If the **the** course grid point is not good, you to go for final grid point and mass refining and think like that **actually**. **Alright**, so this is the whole idea of this transcription method **actually**.

(Refer Slide Time: 08:02)

Problem Objective & Philosophy

Objective:

$$\text{Minimize } J(\mathbf{x}(\cdot), \mathbf{u}(\cdot)) = E(\mathbf{x}(t_0), \mathbf{x}(t_f)) + \int_{t_0}^{t_f} L(\mathbf{x}(t), \mathbf{u}(t)) dt$$

Subject to $\dot{\mathbf{x}}(t) = f(\mathbf{x}(t), \mathbf{u}(t))$

with end point conditions

$$e(\mathbf{x}(t_0), \mathbf{x}(t_f)) = 0$$

and path constraints

$$h(\mathbf{x}(t), \mathbf{u}(t)) \leq 0$$

Philosophy: Select grid points, Discretize the states and control variables, Convert the problem to a nonlinear programming (NLP) problem and solve that problem, preferably in a computationally efficient manner.

NPTEL OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 11

Now in mathematical sense what is the thing, we have this objective of minimizing this notations are slightly different here. But essentially conveys the same message here the you have to minimize this **this** cost functions, is a function of state in control both, which is the end point condition and this is the path condition. Subject to this **this** dynamic equation and with end point, this is the soft condition for end points, where as these are hard conditions for the end point **actually ok**.

So, end point condition equal to 0 and think like that some functions of end points, essentially it tells that the desired value can be fixed. Initial condition can be known think, like that all these or everything will fall within this **this** functional form if you write it to that way. And there is also path constraints which are not discussed so, far, but will we will discuss around the subsequent lectures, where this problem can subject to the state and control inequality constraints all the way basically.

So, these constraint is also part of the problem formulation do not do not forget that **actually ok**. So, this is how the problem objective is and the whole idea here is we select a set of grid points discretized the state, and control variables. Convert the problem into a into a non-linear programming problem and solve that problem. And it should solve in such preferable it should solve that in a computationally efficient manner **actually**, that is what it the whole idea, I mean of direct transcription.

(Refer Slide Time: 09:29)

Key Components of Direct Transcription

1. Choose discretization points (grid);
2. Approximate the trajectory $x(t)$;
3. Discretize the state equation (derivative approximation);
4. Approximate the integration in the cost function.

Euler's Method:
 $t_{k+1} = t_k + h$

Approximate $x(t)$ by piecewise linear function.

Free variables :
 $[(x_0, u_0) \quad (x_1, u_1) \quad \dots \quad (x_N, u_N)]$

Ref: I. M. Ross, Lecture notes, Short course in AIAA GNC-2010, Toronto

NPTEL OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 12

So, to demonstrate the idea what we are talking here is let us say assume that time domain is **is** discretized into this several **several** points like that, here one here one here one here one like that **actually** (Refer Slide Time: 09:45). And if you have a continuous function like say x of t is some continuous function like this what you are telling is we select the value of x of t that point of this and then here and then here like that. So, then you will you join it linearly basically. So, this continuous function is approximated by a set of straight lines hope it is a continuous function you can see that way **actually ok.**

So, essentially approximate that trajectory that way and discretize the state equation and also you can discretize the control variables as well; similar to state variable, you can do that way **actually**. Then also we need a approximation of the integration of the cost function and we will proceed that will I means that is easy but, there are several improvements on that for basically. **Alright** how do you mechanize this **this** discretization of the state, you can very easily do that using this Euler's method.

And other words, if you take t_{k+1} is $t_k + \Delta t$ or $t_k + h$ or $t_k + \Delta t$ or $t_k + h$ or $t_k + \Delta t$ or $t_k + h$, then you get something like very **very** easily you can **you can** have a pair of x_k, u_k and think like that for the entire domain **actually**. Euler method is **is** very easy, because it all tells x_{k+1} is $x_k + \Delta t \cdot f(x_k, u_k, t_k)$ whatever it is x it tells that. If you have differential equation \dot{x} is **is** an f of x let us say x of u or whatever then x_{k+1} is $x_k + \Delta t \cdot f(x_k, u_k, t_k)$.

So, that is very easy to see on that **actually**, we can implement that and set of finite constraint this **this** is will act as constraint now on those variables **actually**. So, the system dynamics is accounted for all the grid points basically in some sense. **Alright**, so, this is how the way to discretize the state dynamics, so we got a discretized form of state dynamics using Euler method.

(Refer Slide Time: 11:44)

Key Components of Direct Transcription
 Reference: I. M. Ross, Lecture notes, Short course in AIAA GNC-2010, Toronto

Approximate the derivatives as: $\dot{\mathbf{x}}(t_k) \approx \frac{\mathbf{x}(t_{k+1}) - \mathbf{x}(t_k)}{h}$

In matrix form,

$$\frac{1}{h} \begin{pmatrix} -1 & 1 & 0 & \dots & 0 \\ 0 & -1 & 1 & \dots & \vdots \\ \vdots & \vdots & \vdots & \ddots & 0 \\ 0 & \dots & 0 & -1 & 1 \end{pmatrix} \begin{pmatrix} x_0 \\ x_1 \\ \vdots \\ x_N \end{pmatrix} \approx \begin{pmatrix} f(x_0, u_0) \\ f(x_1, u_1) \\ \vdots \\ f(x_N, u_N) \end{pmatrix}$$

Approximate the integration as:

$$\int_{t_0}^{t_f} F(t) dt \approx \sum_{k=0}^{N-1} h F(t_k)$$

$\approx \frac{h}{2} \{F(x_0, u_0) + 2F(x_1, u_1) + \dots + 2F(x_{N-1}, u_{N-1}) + F(x_N, u_N)\}$

The slide also includes a graph of state x versus time t showing a discrete approximation of a continuous trajectory.

Let us say, so, this is what is written here and in the matrix form if you put it this then it turns out to be something like this. And the approximate the integration as something like this; suppose you want there is a integration that needs to be done. This I and this is also like you can of this small error here if you to make it comfortable you can think that this is not a but, this is I **actually**, so this is I **ok**.

So, you can and all these are also like I basically anyway. So, that is way of implementing the trapezoidal rule really, but thinks may **may** be different also everybody does not have to use trapezoidal rule for say. But, to **to** visualize the concept **actually** you can see that state equation I can **i can** discretize using Euler method and cost function.

I Will discretize using trapezoidal rule **actually**, then it becomes a easy problem sought of thing and the entire set of state equation, what you see in the grid points can we written in this form in that side **actually**.

(Refer Slide Time: 13:33)

Direct Transcription

End point conditions

$$\mathbf{x}(t_0) = \mathbf{x}_0^* \Rightarrow \mathbf{x}_0 = \mathbf{x}_0^*$$
$$\varphi(\mathbf{x}(t_f)) = 0 \Rightarrow \varphi(\mathbf{x}_N) = 0$$

Path constraints

$$h(\mathbf{x}(t), \mathbf{u}(t)) \leq 0 \Rightarrow \begin{bmatrix} h(\mathbf{x}_0, \mathbf{u}_0) \leq 0 \\ h(\mathbf{x}_1, \mathbf{u}_1) \leq 0 \\ \dots \\ h(\mathbf{x}_N, \mathbf{u}_N) \leq 0 \end{bmatrix}$$

NPTEL OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 14

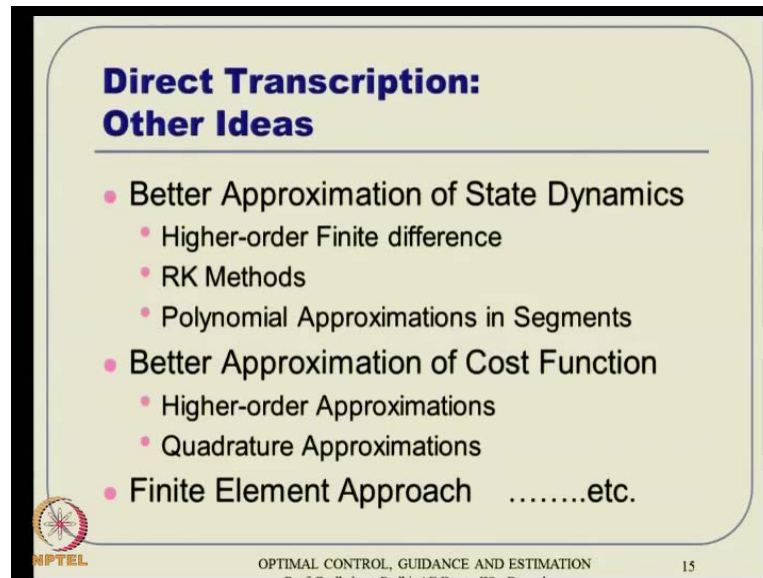
The end point conditions also you can put except u naught is if you put a grid point at the initial time and final time, then directly values are known to us. Then path constraints also given the these are the path constraints that needs to be satisfied. Remember all this things, now converted in the set of discrete form **actually** discrete variables really. So, everything will happen good in a discretized sense sought of thing.

(Refer Slide Time: 13:24) So, then you have set of set of compatible I mean optimization problem for the static programming sense. We have this discretized cost function to optimize; subject to this equality constraint and subject to this initial final condition constraint and subject to this path constraints. Everything's are in the everything happens in the form of discretized variables **actually**.

Now, we have a compatible non-linear programming and hence we can excite any of this **this** techniques which are which are available numerical techniques and all that to solve this problem **actually**. And just to recall, you can somebody can think of utilizing this mat lab optimization tool works and think like that **actually ok**.

This **this** there are tool boxes, there are various other tool boxes available in the market as well other than mat lab, mat lab also this is happening from functions and all that, which I have discussed before. Those thing can be excepted to **to** have solution of this discrete variables. Remember you are not solving a continuous optimization problem, we are solving the discretized optimization problem really here.

(Refer Slide Time: 14:24)



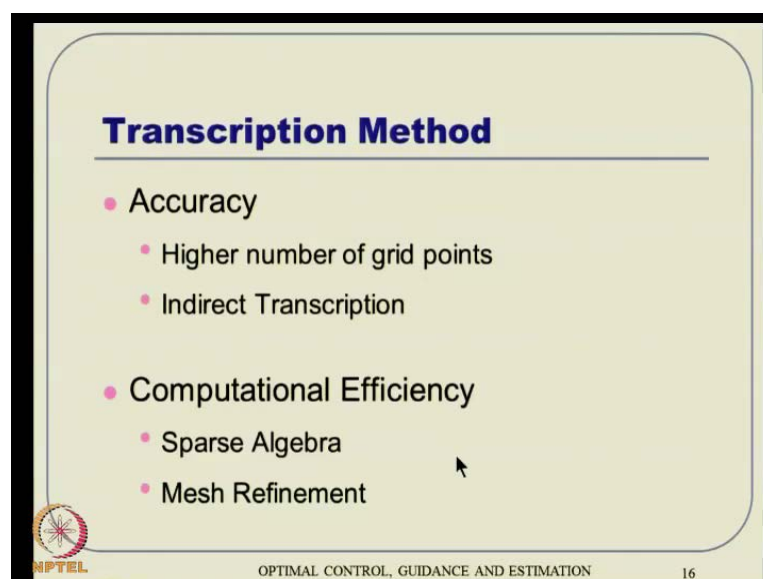
**Direct Transcription:
Other Ideas**

- Better Approximation of State Dynamics
 - Higher-order Finite difference
 - RK Methods
 - Polynomial Approximations in Segments
- Better Approximation of Cost Function
 - Higher-order Approximations
 - Quadrature Approximations
- Finite Element Approachetc.

NIPTEL OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 15

Then there are various **various** ideas about that. So, the how to the better approximation of the state dynamics or the integration is not good, you can think of higher order finite difference you can use RK 4 methods RK methods RK 2, RK 4 think like that; can use polynomial approximation in segment the polynomial and think like that way. Coming to the better approximation of cost function, you can use higher order approximations or you can think about using quadrature approximations as well these are mathematical **mathematically** more powerful **actually**. You can also talk about finite element approaches now days and **and** many different variations around that.

(Refer Slide Time: 15:10)



Transcription Method

- Accuracy
 - Higher number of grid points
 - Indirect Transcription
- Computational Efficiency
 - Sparse Algebra
 - Mesh Refinement

NIPTEL OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 16

So, in general someone can think about improving the accuracy using higher number of grid points. And then there is this indirect transcription method, which is always there with us, which is going through this **this** dualization of the problem through this lambda dot equation. And then, using some sought of numerical procedure for that and that **that** is is really indirect, so we are talking about direct transcription method here, so that is not that much relevant here. So, ultimately remember even if you go for indirect transcription, what you are getting solution is for the grid points only, so that is our I means restrictions let us say.

And coming to the computational efficiency point of view, people can think of using sparse algebra because if you see this matrix lot of zeros are there **actually** (Refer Slide Time: 16:00). Because, of this lot of zeros are available is appear naturally why to do zero computations, I mean 0 into 0 is 0 all the time. So, then this **this** sparse algebra can be excited towards speed of the algorithm and then also you can excite this **this** mesh refinement ideas.

So, this first you solve this with a course grid minimum **number of** number of grid points in less you can solve it and then the increase the number of grid points and then subsequently increase it further, then think like that, so that is the idea of mesh refinement **actually**. So, initially you will not waste too much of computational time and for working directly with some sought of a course.

Some sought of a fine grid lot of grid points, means large dimensional optimization problem and you do not want run into this difficulty of large number of local minimum. And then issues like computational time and think like that. So, it is always advisable to start with a course grid and then go slowly towards more and more number of grids **actually**. So, thus called mesh refinement.

(Refer Slide Time: 17:05) Many references are available based on this **this** whole idea of direct transcription and conceive many of this. First thing that probably comes to my mind is **is** this similar paper which come comes in 1987. And then little more better explanation sense and difference different problem and think like that appeared in 1992, 1993 little more a way. And then there are I mean there are several other papers available and like let us say this survey papers and then the more elaborate explanation.

And think like that available, you can see some of these. Especially the last one is also very good one. In fact, the author as written a book and **he is** he has also developed a commercial software for solving this **this** I mean industrial industry grade commercial software for solving optimal control problem using this indirect transcription approach **actually**.

So, you can see some of the concepts out there. Especially you can concentrate on this page numbers about well about 15 pages where you can get lot more insight into that **actually**. Anyway, lets come back to that and as I had told you in the beginning will concentrate on a different approach which is **which is** evolved recently last 10 year, I will say which is **which is** become very popular in **in** a industrial especially in **in** U S A. Because, of several I mean, several practical problem have applied and then, it is also been tried out in international space station, zero propeller an many over in several satellite minimum time many over and think like that **actually**.

And now it has also been applied for revisable launch vehicle guidance applications and it is also been used in robotics that think like that **actually**. So, here many things that are **that are** happening recently, so I thought it is a good idea to see this **this** philosophy. So, that if some of you are interested you can develop some **some** affinity towards this and probably you can think of that is the possible research direction for yourself also basically.

(Refer Slide Time: 19:12)

Problem Objective


Minimize $J = E(\mathbf{x}(t_0), \mathbf{x}(t_f)) + \int_{t_0}^{t_f} L(\mathbf{x}(t), \mathbf{u}(t)) dt$

Subject to $\dot{\mathbf{x}}(t) = f(\mathbf{x}(t), \mathbf{u}(t))$

with end point conditions $e(\mathbf{x}(t_0), \mathbf{x}(t_f)) = 0$

and path constraints $h(\mathbf{x}(t), \mathbf{u}(t)) \leq 0$

Philosophy: Discretize the states (and the control) using Pseudospectral (PS) method, Convert the problem to a "lower-dimensional" nonlinear programming (NLP) problem and solve that problem in a computationally efficient manner.

 OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 19

Alright, so, will go back to the problem definition and then then come back with this objective we have to we have to minimize this this cost function with soft end point consideration. I mean I mean this cost function contains an end point, I mean end point function and a path dependant functions actually. Subject to this system dynamics with end point want can reserve as well as path constraints like this; the philosophy here is discretization the state shown as before as well as the control. Now using pseudospectral method. So, you do not have to put blindly grid points and try to go discretization directly sort of thing and what is pseudospectral I will explain in the anyway.

So, due to discertizing, but discretize is in different way using pseudospectral method and so, do that you can convert the problem to a very lower dimensional non-linear programming problem. So, the whole idea here is how to formulate in a lower dimensional NLP problem actually, so that it can be solve much more faster and while solving you to solve in a very computational efficient manner. So, various branches of mathematics will come together here and essentially it will put together in such a way that ultimately results in fast solution basically.

(Refer Slide Time: 24:27)

Steps involved...

- 1. **Approximate $x(t)$ and/or $u(t)$?**
$$\hat{x}(t) = \sum_{n=0}^N a_n \phi_n(t) \quad \hat{u}(t) = \sum_{n=0}^N b_n \phi_n(t)$$
- 2. **Selection of grid points**
 - How are these points selected?
 - Uniform grid is not a very good choice!
- 3. **Discretize the differential equation using PS method**
 - Finite-difference Vs Spectral
 - Sparse Vs Dense differentiation matrix
- 4. **Approximate the integral equation**
 - Quadrature rules
- 5. **Apply an efficient finite optimization technique and solve the lower dimensional NLP problem.**

NPTEL OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 20

So, let us see the steps involved for this particular thing first of all you have approximate x of t and u of t but, how do you do it something like this. You do that remember this pseudospectral method is largely inspired from this solutions for parcel differential equation also (Refer Slide Time: 20:49). So, the parcel one way of the solving parcel differential equation these two basic functions and all that.

So, you use basic functions and one dimension and leave the other dimension for converting for p d is to o d is and all that probably that concept will see towards the end of this course in one or two lectures. Then, we will right now the borrowing the similar philosophy what we do here is, first we approximate the this state and control using some sort of basic functions ϕ_n which is ϕ_0, ϕ_1, ϕ_2 and think like that and do what difficulty this can be polynomial basic functions.

For example, and all that that is not very effective. So, the studies is tell us that in a the function approximate theory, if you go through that **that** side of story there are better function approximation basis functions like sparse polynomial, legend polynomial think like that, there are essentially kind of our series. But come terms taken together and think like that **actually**.

So, more on that you can study some methodic legend polynomial, sparse polynomial think like that **actually**, if you that kind of thing then, we take bunch of those. And

remember those polynomial have a very nice formula for recursive relationship. So, in the other words the programming sense also it can be very fast **actually**.

So, go there and tell x at of t something like this $x(t)$ and a_n and $v_n(t)$ something like that **actually**, but these are written like that in a finite number of basic functions. This dimension can be different for simplicity of algebra sake, will take it something same, but. here can be $n-1$ here, it can be $n-2$ does not have **actually**. Then, you have to selective set of grid points anyway and I will see summaries how to this set as how to select a set of grid points **actually**.

Now, how are these points selected in the **in that** we can some sought of question and then; obviously, it turns out that uniform grid points is a is not a very good choice. So, it have to oppose this called non uniform grid points for see, the whole idea is how to represent the same problem in a less number of grid points **actually**. So, is a large and large number of grid points is not a choice.

So, we want to have as possible and in turns out that you can really do a good job with less number of grid points but, we have to compromise the idea of uniform grid points. Let go through this non uniform grid points gauss location points and think like that **actually**. So, this now, discretize the differential equation is pseudospectral method and that method turns out to something like this we have this **this** finite difference versa pseudospectral thing; that means, finite difference is typically not good.

So, what you do is this differential equation we do this the spectral discretization sort of thing. So, sincerely instead of running into a large dimensional sparse matrix **actually**. What you are looking for is a small dimensional dense differential matrix. What is different in matrix is something, what we studied just now this **this** form (Refer Slide Time: 23:50) this is called a differentiation matrix, if you **if you** see that **actually**.

So, this can defined as differentiation matrix, so instead of a large differentiation matrix lot of zeros sparse **sparse** matrix, what you are looking for some sought of similar matrix which will lose similar all. But, we want a smaller dimension we at a dense matrix, we do not want see zeros very **actually**. Now, let us now if you do this approximate the integral equation also that, can be done even this **this** quadrature rules for that quadrature same all that **actually**.

Now, we have to use this **this** apply the efficient finite optimization, after that also we have do not done. Remember that even though if it represented is equalance smaller dimensional problem still that smaller dimensional problem has to be solve as fast as possible. So, we have to use this **this** algorithms, which has to be very powerful also remember that as, I mean whenever there is processor capability of the, I mean whenever there is increase of capability in the processor that also come very end **actually**.

So, in other words going this computational advantage of modern day computers of modern day processors think like that, **that** also comes very ending. But it is not intentionally solutions lies really in the good algebras **actually**, you may **sorry** good algorithm **actually**.

So, it too for something some **some** better capable algorithms to solve this N L P problem and always solving that a procedure which is more capable always better **actually**. So, this how the whole the idea is their **actually**, now will explain the steps little more clearly that here **actually**. So, how you do step one is approximation right, so how you do approximation.

(Refer Slide Time: 25:33)

1. Approximation

It can be thought of as a function \hat{x} which satisfies the boundary condition and makes the residual

$$\|R\| := \|\hat{x} - f\| \quad \text{SMALL?} \quad \dot{x}(t) = f(x(t), u(t))$$

- Define Trial functions: $\mathcal{P}_N := (\phi_0, \phi_1, \dots, \phi_N)$

$$\text{Approximate } \hat{x}(t) = \sum_{n=0}^N a_n \phi_n(t), \quad \hat{u}(t) = \sum_{n=0}^N b_n \phi_n(t)$$

- Define Test functions: $(\chi_0, \chi_1, \dots, \chi_N)$

$$\langle \chi_n, R \rangle = 0 \quad \forall n \in \{0, 1, \dots, N\}$$

NPTL OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 21

Now, these approximations is to be done in such way, so that residual between these two whatever none of that needs to be small **actually**. Now, how do you make sure that is small, so what you do this is already the approximation we have and now we defines some sought of test function.

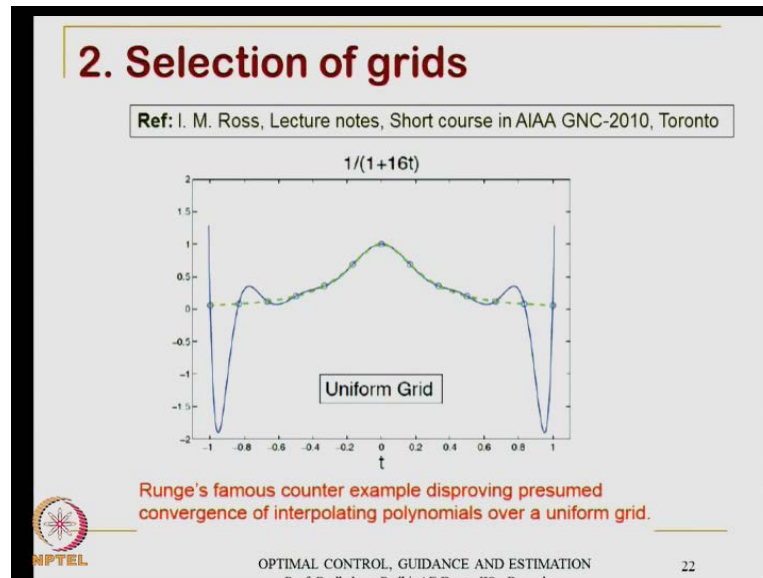
Some of the different functions the chi naught chi χ_1 χ_2 like that **actually ok**, right in such a way that it is a well this is χ_1 basically **alright**. So, this is we have to do it such way that this **this** residual error has to be minimum you know this is inner product. And this inner product of continues function in other words integral sense **actually**. So, this in inner product is define something **something** like this, we have two continues functions.

So, if you have two continues function f_1 and f_2 then inner product between them is some finite interval let us say 1 to t then 0 to t , then f_1, f_t and f_2, f_t sort of thing, that is what the inner product definition is that was we are using here **actually**.

Alright, so now the question is we have a test function series but, what test function you need to solve, so that algebra becomes simple, and it results in simple dynamics and all that **actually**. So, one idea turns out to mine that whenever you see the integral and all that why not trying out the delta function. If you try out the delta functions again, if you put this one of the function reference to be delta function, then the entire integral happens to be just function really basically, sometime people I mean define this is without this 1 to t is not there.

And if you have one of that something like delta function, $\delta(t - t_n)$ **actually**. Then, what happens the entire integral evolution turns out to be just f_1 of t_n , if the only condition is t_n is has to be strictly line between this interval **actually**. If it is one of the interval point, then it is to half of that **actually**. Anyway, but guessing that t_n is strictly inside this integral 0 to t , then the evolution of the integral happens to very easy and the residual happens to be the residual value of that point of the time **actually ok**. So, this our idea is and then the it mention further simplicity and all that.

(Refer Slide Time: 28:04)



Now, coming to the second point, how do you select the grid points and this is very nice demonstration, I taken from most of the lecture I taken from this material, which this short course in a I a a G N C which 800 myself. Some of the material are taken from that side of demonstrate it **actually**. Anyway, this is a **this is** very nice thing here happening. So, this is to see this function here and then just put uniform grid points.

Then what happens here is it is **is** not happening or other words it happens up to this grid point let say. In this centre, it is because, whatever the value you are getting you are getting it there anyway. But, the function value of the matches within that interval also whatever points you are not seeing within that, there also the error small **actually**, within this period I mean the time interval. But, has you go towards the boundary the function value is very different even though the grid point value is same.

So, there is some sort of a illusion rather basically; that means, the **the** value the grid point value is same as the function value but, in between the grid point there is a large dispatcher **actually**. So, obviously, this is not a good function approximation **actually**, but if you take the same function, you can do this I mean do this non uniform grid point and think like that will have a lot better advantage **actually**.

(Refer Slide Time: 30:00)

2. Selection of grids

$t_0 = -1$ $t_1, \dots, t_{N-1} : \text{arbitrary}$ $t_N = 1$

Grid of collocation points (or grid points) $t_n, n=0, \dots, N$ are points such that it satisfies the state equation exactly at these points.

Increasing Generality →

	Free Endpoints	One Endpoint Fixed	Arbitrary Endpoints
Gauss-Lobatto	Yes	Yes	Yes
Gauss-Radau	Yes	Yes	No
Gauss	Yes	No	No
Uniform	No	No	No

↓ Decreasing Applicability

OPTIMAL CONTROL, GUIDANCE AND ESTIMATION
23

So, this **this** grid points which are really non uniform located typically called as grid of collocation points, are normally grid points illusion where call that way. So, these are the typically done in very non uniform sense; which is something close here, something close here, something is sparse here, think like that **actually**. Depending on the several approach again **actually**.

So, anyway this is what you study here and then what **what** it the **the** literature claims that you have this the several ways of selecting that, something like uniform then, there is gauss and point gauss grid point and the gauss radeu grid point, gauss labatto grid point various things available in the math literature **actually**. And it turns out the uniform grid points is not good, because whether you have p n points or 1 n points or arbitrary end points no where it is approximate in a good way. But, if you slowly increase these like go to gauss **gauss** radeu in gauss labatto. Then it turns out the gauss labatto points will do a much better that in every where **actually**.

So, now, the **the** recommendation always go towards this **this** gauss labatto points **actually**. The exact formulas of that is always available in the literature, you can **you can** see it very clearly basically. **Alright**, so gauss is slightly better but, do not stop it the gauss level but, go towards the gauss labatto points sought of thing **alright**.

(Refer Slide Time: 30:59)

3. Approximating the differential equation

Approximations: $\hat{x}(t) = \sum_{n=0}^N a_n \phi_n(t), \quad \hat{u}(t) = \sum_{n=0}^N b_n \phi_n(t)$


State Equation Constraint:

$\dot{x} = f(x(t), u(t))$
 $\dot{\hat{x}} = f(\hat{x}(t), \hat{u}(t))$

$$\sum_{n=0}^N \dot{\phi}_n(t) a_n = f\left(\sum_{n=0}^N a_n \phi_n(t), \sum_{n=0}^N b_n \phi_n(t)\right)$$

Multiply with $\delta(t - t_n)$ on both sides:

$$\sum_{n=0}^N \underbrace{\dot{\phi}_n(t_n)}_{\text{A number}} a_n = f\left(\sum_{n=0}^N \underbrace{a_n \phi_n(t_n)}_{\text{A number}}, \sum_{n=0}^N \underbrace{b_n \phi_n(t_n)}_{\text{A number}}\right), \quad n = 0, 1, \dots, N$$


OPTIMAL CONTROL, GUIDANCE AND ESTIMATION
24

Alright so, now, coming back to approximation of the differential equation, here is something that we need really understand **actually**. Let us assume that, done the we selected the, this one we selected the basic functions, we have approximated that way and we have selected the grid points also now what **actually**. So, what we are looking for is approximation of the differential equation.

So, we have this state equation and control equation that way and then equation constraints will tell us these are the **are the** constraints what is happening here. So, I will put it I mean approximate value I will pit it here. whatever I am approximating I will put it here, the dot that dot remember this only function of time, this is constraint **actually**. So, the this dot will reflect in this dot really, and then we have this **this** quantity in the right hand side now.

If you multiply by with **with** delta functions on both sides then what happens the this happens to be some sought of a number and then we will end of with this coefficients only basically **ok**.

Now, let us **let us** see that in a little bit more elaborate sense let us take some something like a Chebyshev polynomial and combine ourselves to like five grid points only just to have this clarity of idea basically. We have this grid, I mean the nice thing about this is first to thinks you to write and the subsequent things you can write recursively.

Ultimately, it happens to be like this basically, now what happens is x rate of t lets say we **we** assume expanded it this way right. And B at is also I mean a rate is also expanded that way. So, we I have not telling that especially here but, then x dot of t when you do that then it is like t dot here t 1 dot t 2 dot, because a 0, a 1, a 2 these are all constant quantities **actually**.

Now, the good thing is this quantities once it is evaluated at a grid point it happens to be a number basically. So, if i **if i** look at this equation an at a particular grid point, let us say some 0 grid point whatever it is, 0 is something like this, because **these** these are now numbers **actually**. Similarly, x rate of t 1 happens to be all these are numbers and these the coefficient, which is which are typically unknown.

So, what will happen in the left hand side you will a differentiation matrix in the form of these but, these are all numbers now, t 1 t 2 t 3 are all grid points now basically. And right hand side also a number, because ultimately here we have this **this** grid point value are available basically.

(Refer Slide Time: 31:57)

3. Approximating the differential equation (Example)

- The Chebyshev polynomials

$$\begin{aligned} T_0(t) &= 1 \\ T_1(t) &= 2t - 1 \\ T_2(t) &= 4t^2 - 4t + 1 \\ T_3(t) &= 8t^3 - 8t^2 + 4t - 1 \\ T_4(t) &= 16t^4 - 32t^3 + 24t^2 - 8t + 1 \end{aligned}$$
- Polynomial Approximation

$$\begin{aligned} \hat{x}(t) &= a_0 T_0(t) + a_1 T_1(t) + a_2 T_2(t) + a_3 T_3(t) + a_4 T_4(t) \\ \hat{\dot{x}}(t) &= a_0 T_0'(t) + a_1 T_1'(t) + a_2 T_2'(t) + a_3 T_3'(t) + a_4 T_4'(t) \end{aligned}$$
- The differential yields

$$\begin{bmatrix} \hat{x}(t_0) \\ \hat{x}(t_1) \\ \hat{x}(t_2) \\ \hat{x}(t_3) \\ \hat{x}(t_4) \end{bmatrix} = \underbrace{\begin{bmatrix} T_0(t_0) & T_1(t_0) & T_2(t_0) & T_3(t_0) & T_4(t_0) \\ T_0(t_1) & T_1(t_1) & T_2(t_1) & T_3(t_1) & T_4(t_1) \\ T_0(t_2) & T_1(t_2) & T_2(t_2) & T_3(t_2) & T_4(t_2) \\ T_0(t_3) & T_1(t_3) & T_2(t_3) & T_3(t_3) & T_4(t_3) \\ T_0(t_4) & T_1(t_4) & T_2(t_4) & T_3(t_4) & T_4(t_4) \end{bmatrix}}_{\text{Differentiation matrix}} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \end{bmatrix} = f \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \end{bmatrix} : \text{An algebraic constraint}$$

NPTEL OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 25

So, once the grid points values are available those are **those are** become some sought of a numbers that are known, what are unknown is something like coefficients not **not** to be a 4 and b 4. So, this essentially even though it is differential equation constraints it results into some sought of algebraic constraints in the form of parameters **actually**. So, if you consider the problem to be kind of a unknown function in terms of a naught a 4 and b

naught to b 4 and it happens to be some sought of a algebraic constraint in in the form of coefficient, in fact everything done in form of coefficient actually.

The last one how do you do this discretization of the integral equation, you can do that, trapezoidal rule but, there are much better ways of doing that which is something like (O) rule. And that is nothing but an approximation of the definite integral function usually expressed something like a weighted sum, where the w I can be computed appropriately depending on what kind of we are talking actually 0.


So, ultimately it results in some sought of a discretization of the cost function also any way that what will need for the for the static optimization problem actually. So, we have got a algebraic constraint, which is converted into our is taken care of the state equation constraint.

(Refer Slide Time: 33:54)

4. Discretizing the integral equation
- Gauss Quadratures

A quadrature rule is an approximation of the definite integral of a function, usually stated as a weighted sum of function values at specified points within the domain of integration.

$$\int_{-1}^1 L(x(t), u(t)) dt \approx \sum w_n L(\hat{x}(t_n), \hat{u}(t_n))$$
$$J \cong J^N = E(\hat{\mathbf{x}}(t_N), \hat{\mathbf{u}}(t_N)) + \frac{t_f - t_0}{2} \sum_{n=0}^N w_n L(\hat{\mathbf{x}}(t_n), \hat{\mathbf{u}}(t_n))$$

 NPTEL

OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 26

And here it is something that will take care of the cost function other things are illustrate it direct actually.

(Refer Slide Time: 34:41)

Finally,


Minimize, $J^N = E(\hat{\mathbf{x}}(t_0), \hat{\mathbf{x}}(t_N)) + \frac{t_f - t_0}{2} \sum_{n=0}^N w_n L(\hat{\mathbf{x}}(t_n), \hat{\mathbf{u}}(t_n))$

Subject to, $\sum_{n=0}^N \underbrace{\phi_n^j(t_n)}_{\text{A number}} \mathbf{a}_n = f(\hat{\mathbf{x}}(t_n), \hat{\mathbf{u}}(t_n)) \quad 0 \leq n \leq N$

with end point conditions, $e(\hat{\mathbf{x}}(t_0), \hat{\mathbf{x}}(t_N)) = 0$

and path constraints, $h(\hat{\mathbf{x}}(t_n), \hat{\mathbf{u}}(t_n)) \leq 0 \quad 0 \leq n \leq N$

The optimal control problem has been simplified to a lower dimensional nonlinear programming problem.



OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 27

So, finally, what we have is something like discretized, version of the same problem which **which** talks about minimizing this cost functions objective this set algebraic equation now with end point conditions and path constraints. So, this problem can be I mean this **this** optimal can control problem has been now, simplified to a lower dimensional non-linear **non-linear** programming problem. And hence we can **we can** talk about some sought of a non-linear programming problem, solution techniques and all that to which you can I mean use to solve this problem **actually**. **Alright**, so this is now to demonstrate in a small toy problem first and then will solve it at a good application problem and wind up this lecture **actually**.

(Refer Slide Time: 35:27)

Toy Problem

Minimize $J = \int_0^1 u^2(t) dt$


Subject to $\dot{x}(t) = x(t) + u(t)$

Boundary condition $(x(0), x(1)) = (1, e)$

$|u(t)| \leq 1$

- **Step 1. Decide on which polynomial we are planning as the trial function**
 - (i) **Chebyshev Polynomials of the first kind. Say take N=4 (First 5 polynomials)**

$T_0(t) = 1, T_1(t) = t, T_2(t) = 2t^2 - 1, T_3(t) = 4t^3 - 3t, T_4(t) = 8t^4 - 8t^2 + 1$

 OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 29

First this toy problem we want to minimize this control u square from 0 to 1, subject to this linear dynamics that is what is x plus t basically with the boundary conditions like this, and the control happens to I mean control constraint is given the modulus cannot exceed one **actually**. Now this **this** trivial toy problem is something, which interest me in a different sense if you really look at the I mean this cost function is linear to study equation and all that you can **actually** go and end scalar problem also that way.

So, we **actually** go back to the equation I mean whatever we know already. Assuming that this constraint is not there, let us you see that **actually** assuming that this additional constraint is not there you can **actually** go back to equation or whatever equation we have already known before that to solve this problem. And interestingly turns out that the solution is nothing but, equal to 0 throughout basically.

So, if u equal to 0 throughout; that means, no control the homogeneous solution happens to be the optimal solution for this. And u is 0 obviously, cost function is 0 also basically that is the best thing that you can also always have I mean without applying any control we will be able to do this how **actually**. But, remember this is an unstable problem x what is x dot equally to x , I mean plus x is **actually** unstable problem the trajectory will evolve in unstable sense **actually** ok.

This is time, this is x what we are telling is the initial condition is **is** 1 and the final condition at equal to 1, this is our final time. Initial condition is 1 and final time it has to

go and **and** go to e **actually**, this is value e. And you can see this **this** solution if I take equal to 0, while end of this and this **this** particular differential equation I as has nice solution directly you can see that x of t nothing but, if the power x of 0 but, x of 0 is 1. So, this of t, so when t equal to 1, then it is nothing but, e when t equal to 1.

So, we already suggest for the other one condition also x equal 1 is nothing but t. So, we are sought of arguments **actually ok**. So, what finally, what you are looking for is we try to solve this optimal control in a discretized manner using factorial but, ultimately you know the solution and the question is whether you are approaching towards that solution **actually**. So, here if you take chebyshev polynomial of first kind, second kind and think like what you have taken first kind, and you see the these are also kind of polynomial expressions.

But, in a different sense **actually** each of the basis function are different sought of polynomial consults, basically essentially if **if** we put them together. Essentially it is nothing but a basically in a way.

(Refer Slide Time: 38:20)

Toy Problem (Contd.)

➤ (ii) **Shifted Chebyshev polynomials for the interval [0,1]**

$$T_0(t) = 1$$

$$T_1(t) = 2t - 1$$

$$T_2(t) = 8t^2 - 8t + 1$$

$$T_3(t) = 32t^3 - 48t^2 + 18t - 1$$

$$T_4(t) = 128t^4 - 256t^3 + 160t^2 - 32t + 1$$

Step 2. Define the collocation (grid) points

➤ **Shifted** $t_i = \frac{1}{2}(a+b) + \frac{1}{2}(a-b)\left(\cos\left(\frac{i\pi}{N-1}\right)\right); \quad i = 0, 1, \dots, N-1$

The slide includes a diagram of a grid on the interval [0, 1] with points marked at 0, 0.4688, 0.9375, and 1. The NPTEL logo is in the bottom left, and the text 'OPTIMAL CONTROL, GUIDANCE AND ESTIMATION' and '30' are in the bottom center and right respectively.

Anyway you can **you can** use this and then try to go head and then see this **this** one is chebyshev polynomial first kind, other one is shifted chebyshev polynomial for the interval this that and all; so, details you can see that in **in** typically book **actually**. And the collocation grid points have been defined something like that, remember all these things whatever we are talking I forget to tell, one point probably that let me summarize it again

if I here (Refer Slide Time: 38:50) you talk about 0 to t 0 to t f but, the **the** entire grid points and all that will talk about something like minus 1 to plus 1.

So, this scaling is necessary to make it compactable **actually**, so that is why this scaling is **is** put here this plus b by 2 plus a minus b by 2 sought of thing. So, this is **this is** those are the grid points grid point 1, 2, 3, 4 see, we see how those things are spars those are slightly closer to each other these are farer than each other **actually**.

(Refer Slide Time: 39:22)

Toy Problem (Contd.)

Step 3. Approximate $x(t)$ and $u(t)$ using trial function

$$\hat{x}(t) = \sum_{n=0}^4 a_n T_n(t)$$

$$\hat{u}(t) = \sum_{n=0}^4 b_n T_n(t)$$

Step 4. Find the differentiation matrix and equate the state equation at the grid points

$$\frac{d}{dt} = \begin{bmatrix} 0 & 2 & 0 & 6 & 0 \\ 0 & 0 & 8 & 0 & 16 \\ 0 & 0 & 0 & 12 & 0 \\ 0 & 0 & 0 & 0 & 16 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\frac{d}{dt}(\hat{x}(t_i)) = \hat{x}(t_i) + \hat{u}(t_i)$$

NPTEL OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 31

So, like that, then you have this **this** state in control approximated like this polynomial thing. And if you see that you have d by d t of these it turns out to be something like this **actually**. You find out the differentiation matrix and equate the state equation at the grid points sought of things.

So, like that, then you have this **this** state in control approximated like this polynomial thing. And if you see that you have d by d t of these it turns out to be something like this **actually**. You find out the differentiation matrix and equate the state equation at the grid points sought of things.

So, like that, then you have this **this** state in control approximated like this polynomial thing. And if you see that you have d by d t of these it turns out to be something like this **actually**. You find out the differentiation matrix and equate the state equation at the grid points sought of things.

(Refer Slide Time: 39:40)


Toy Problem (Contd.)

Step 5. Compute $T_n(t)$ matrix for t_i and equate the state equation, $i=0,1,2,3,4$

$$T = \begin{bmatrix} 1 & -1 & 1 & -1 & 1 \\ 1 & -1/\sqrt{2} & 0 & 1/\sqrt{2} & -1 \\ 1 & 0 & -1 & 0 & 1 \\ 1 & 1/\sqrt{2} & 0 & -1/\sqrt{2} & -1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

$$\begin{bmatrix} \dot{\hat{x}}(t_0) \\ \dot{\hat{x}}(t_1) \\ \dot{\hat{x}}(t_2) \\ \dot{\hat{x}}(t_3) \\ \dot{\hat{x}}(t_4) \end{bmatrix} = (T \times D) \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \end{bmatrix}$$

Finally, we have

$$T^{-1} (T \times D - I) \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \end{bmatrix} = \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \end{bmatrix}$$


OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 32

And then, compute the T_n matrix something like this T and equate the state equation and also things we have as an inverse of that **actually** t into d and put it there. This constraint, what you see here can be approximated something **something** like this **actually**, sorry this **this** dot equation and all that **actually**. So, again I do not want to go step by step here but, I think you can knowing the concept you can **you can** easily derive all those **actually** I suggest that you do it yourself also basically. So, that is ultimately the constraint equation turns out to be something like this **actually**.

(Refer Slide Time: 40:16)

Toy Problem (Contd.)

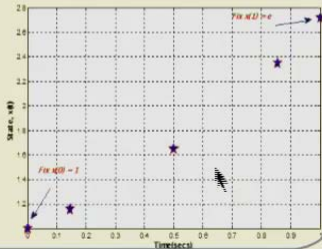

Step 6. Apply boundary conditions

$$\begin{bmatrix} 1 \\ x(t_1) \\ x(t_2) \\ x(t_3) \\ e \end{bmatrix} = \begin{bmatrix} 1 & -1 & 1 & -1 & 1 \\ 1 & -1/\sqrt{2} & 0 & 1/\sqrt{2} & -1 \\ 1 & 0 & -1 & 0 & 1 \\ 1 & 1/\sqrt{2} & 0 & -1/\sqrt{2} & -1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \end{bmatrix}$$

$(x(0), x(1)) = (1, e)$

This gives,

$$a_0 - a_1 + a_2 - a_3 + a_4 = 1$$

$$a_0 + a_1 + a_2 + a_3 + a_4 = 2.7182$$



OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 33

Now, apply the boundary condition, so we **we** apply all this grid point condition boundary condition all that whatever we know here. So, ultimately this gives us this **this** constraint equation that this to constraints has to be satisfied **actually**, initial condition, and final condition.


(Refer Slide Time: 40:34)

Toy Problem (Contd.)

Step 7. Approximate the integral equation

$$J = \int_0^1 u^2(t) dt \cong J^N = \frac{1}{2} \sum_{k=0}^4 w_k \hat{u}^2(t_k)$$

where

$$w_k = (b-a) \frac{\pi}{n+1} = \{1.57 \quad 0.7854 \quad 0.5236 \quad 0.3927 \quad 0.3142\}$$


OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 34

Now, this **this** discretization of the cost function; so discretized that way, where w was are selected something like this all these numbers are **are** given here and then, it results in a discretized cost function **actually ok**.

(Refer Slide Time: 40:49)

Toy Problem (Contd.)

Finally, we have

$$J^N = \frac{1}{2} \sum_{k=0}^4 w_k \hat{u}^2(t_k)$$


$$\hat{x}(t_k) + \hat{u}(t_k) - \hat{x}(t_{k+1}) = 0$$

$$\hat{u}(t_k) - 1 \leq 0 \quad \text{for } k = 0, 1, 2, 3, 4$$

Define the augmented cost function,

$$\begin{aligned} \bar{J}^N = & \frac{1}{2} \sum_{k=0}^4 w_k \hat{u}^2(t_k) + \sum_{k=0}^4 \lambda^T_k (\hat{x}(t_k) + \hat{u}(t_k) - \hat{x}(t_{k+1})) + \\ & \sum_{k=0}^4 \mu^T_k (\hat{u}(t_k) - 1) + v_1 (\hat{x}(0) - 1) + v_2 (\hat{x}(e) - 1) \end{aligned}$$

Apply any KKT (or any other static optimization technique) for solving the optimal control problem



OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 35

So, finally, we have this **this** cost function to optimized subject to these conditions what you have here and define the augmented cost function and think like that. And then we can apply this KKT condition integral condition or any other static optimization technique numerical optimization technique and think like that to solve this optimal control problem **actually**.

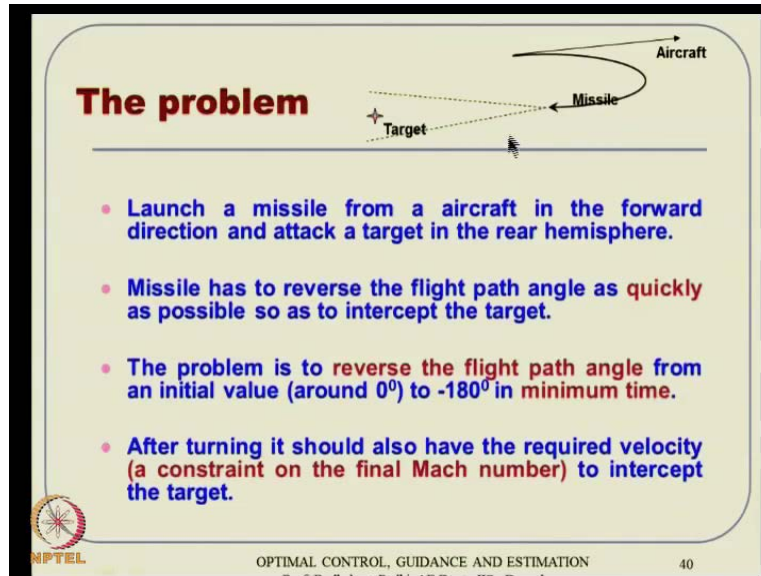
Now, (Refer Slide Time: 41:16) the solution turns out to be very **very** close to what we expect it starts with 1 and then 1 here, 1 here like that but, once you put that the solution form. Remember there is a solution is a polynomial **actually** ultimate **ultimately** the solution is a polynomial, what you are solving for is the coefficient a_n and b_n what you already know is the basis function T_n **actually**.

So, once you know the coefficients **actually**, we know the solution all over **actually**. Now one idea is, we know the solution for control you know the initial condition for the state. So, you can integrate it using RK4 and then, you get a different sequence of numbers and that sequence of numbers have to fall on this, these polynomial what you know **actually**. So, that is the validation technique, so let us assume that exercise is done **actually** in a good way **ok**.

So, anyway, so, coming back to that this is what it is the numbers that we are getting is for the grid points but, associated with that we also have polynomial solution for this, and it happens to be very closely there with respect to the exact solution **actually**. One is interpolated solution; one is collocation point; and one is exact solution. So, the all the things are together **actually**, so that gives us lot of good continuous there, what about control remember these are numerical technique. So, 0 is not really exactly 0, but the turns out to be something like turns upon minus 3 which is **which is** very small very close to 0 **actually**. And that is why it is not disturbing this **this** solution chebyshev basically.

Alright, so, that is some sought of confidence for **for** going further and then what you do is we going to elaborate more realistic problem or something called front to back turning back an air launch missile using pseudo-spectro method. So, let us say how do you do that inverse the problem and think like that for rest of the time that I that I have **actually**. So, this reference (Refer Slide Time: 43:04) **actually** we have recently kind of this presented these in the IFAC workshop which has done in **in** Bangalore right in a **actually**.

(Refer Slide Time: 43:15)



The problem

- Launch a missile from a aircraft in the forward direction and attack a target in the rear hemisphere.
- Missile has to reverse the flight path angle as quickly as possible so as to intercept the target.
- The problem is to reverse the flight path angle from an initial value (around 0°) to -180° in minimum time.
- After turning it should also have the required velocity (a constraint on the final Mach number) to intercept the target.

NPTTEL OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 40

Anyway this **this** is the problem where you have a carrier aircraft I mean taking a missile now typically what happens is aircraft and the whatever missiles they carry they are all front looking; that means, any target that **that** appears within the front side. They can be launched and they can be engaged **actually**. But what happens some **some** target appears in the back, then the whole idea of that means, this angular I mean this aircraft can increase its angle of attack.

So, let us say and then it also has more drag, so the speed comes down it goes up with a lesser speed by that time this goes front and then, it **it** drag the reverse maneuver and; that means, angle of attack is reduced it comes down again. And so, roughly it **it** kind of becomes behind the target. And the maneuver that we are talking about, first going up and then coming down and think like that this is nothing but, a cobra maneuver. So, I mean this real typical terms use in the air **actually**.

Alright so, but that takes a lot of time first, we have to go up, and then increase the angle go up reduce come back and then align yourself. So, that you look at the target in front of you **actually**. So, by that time target is not passive, it is also looking at you here. You remember that for the target, if it also happens to be in aircraft and all that in a **in a** earlier you are already in front of him; that means, this target has an advantage of heading towards **towards** the aircraft then, you **you** I mean we getting in advantage to fight towards the target **actually**.

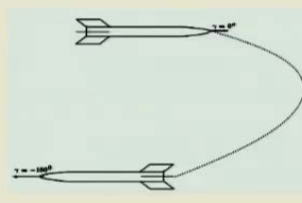
So, there is a problem happens then that can I **can I** really not launch a missile here, after all that is also an aerospace vehicle, with larger capability aircraft turning is much lesser capability compared to the missile turning basically. So, this is larger turning capability, so I will launch it and then quickly turn it backwards. So, that now I can the missile target and hence it is **it is** equivalent to like a attack basically.

So, the problem here is how to term this, from I mean turn this vehicle from front side front ward going to the backward sought of things. In other words the problem is to reverse the flight path angle from around 0 degree to something like minus 180 degree and minimum time, cannot take too much of time to do the job also. Because, I mean in a real come back scenario time is everything **actually**. So, we turn it very quickly and then the problem how about the turning problem.

Hence here after that it can be guided it is in a different sense **actually**. And also remember by the time it turns, it cannot take too much of in this drag, I mean turning also excides this drag conception and all that. And you want to turning as this penalty of in this drag. So, it is not a too much of that. So, ultimately what the constraint is 1 gamma goes to minus 180 degree which should also have some **some** desired final mach number basically. So, that you can still continue to move towards that **actually**.

(Refer Slide Time: 46:19)

**The problem -
Cost function and boundary conditions**




Minimize $J = \int_0^{t_f} dt$

subject to the constraint

$\gamma(t_0) = \text{known (around } 0^\circ), \gamma(t_f) = -180^\circ$
 $M(t_0) = \text{known}, M(t_f) = 0.8$

The problem is to reverse the flight path angle from 0° to -180° maintaining a final Mach number of 0.8 in minimum time.



OPTIMAL CONTROL, GUIDANCE AND ESTIMATION

41

Now the problem is something like that minimize this 0 to t f different. So, minimize this cost function subject to the constraint that initial gamma is around 0 final gamma is

minus 180 initial mach number is known, when final mach number let us say equal to 0.8 hard constraint **actually**. The problem is to reverse the flight path angle from 0 to minus 180 maintaining the final mach number or **0.1**, 0.8 in and it has to be done in minimum time **actually ok**.

(Refer Slide Time: 46:58)

System Dynamics

The point mass equations of motion:

$$\dot{V}(t) = -\frac{D}{m} - \frac{W}{m} \sin \gamma + \frac{T}{m} \cos(\alpha + \delta_b)$$

$$\dot{\gamma}(t) = \frac{1}{mV} (L - W \cos \gamma + T \sin(\alpha + \delta_b))$$

$$\dot{h}(t) = V \sin \gamma$$

$$\dot{x}(t) = V \cos \gamma$$

OPTIMAL CONTROL, GUIDANCE AND ESTIMATION
42

So, the system dynamics is first and we see that vehicle once it is launched it has to go this **this** kind of a thing. So, we have this **this** dynamics in this scenario, we have some gamma coming here and some alpha and think like that. So, and also there is a thrust defluxion angle that we are assuming, the thrust is not happening in the direction it has to happen some sought of a different angle sought of thing that can be done typically through various mechanism of defluxion. Can be some side v t side injection susceptible control or may be some **some** fin reflection at the end something like that **actually**; or that there can be jet veins or whatever it is **actually**.

Some mechanism adjusts, so that the thrust can be reflected. So, that **actually** arguments to the angle of attack turning basically, thrust vector is much more powerful vector if it turns then, the vehicle turns very quickly also basically in a way. And typically these are not restricted to very **very** low degrees also I mean thrust reflection angle that we are talking here can be little bit higher, than the **than the** alpha and **actually** that way.

Anyway so, here this **this** point mass equation $v \dot{\gamma}$ and $s \dot{n} \times \dot{\gamma}$. So, these two represent dynamic level equation, then these two remain kinematic equations like that it happens in two descents here.

(Refer Slide Time: 48:14)

System Dynamics
 Ref: Han et al., "State constrained Agile Missile Control with Adaptive Critic Based Neural Networks", JGCD, 2002

The non-dimensional parameters were considered as:

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

Non-dimensional point mass equations of motion:

$$\begin{aligned} M'(\tau) &= -S_w M^2 C_D - \sin \gamma + T_w \cos(\alpha + \delta_b) \\ \gamma'(\tau) &= \frac{1}{M} (S_w M^2 C_l - \cos \gamma + T_w \sin(\alpha + \delta_b)) \\ h'(\tau) &= \frac{a^2 M \sin \gamma}{g} \\ x'(\tau) &= \frac{a^2 M \cos \gamma}{g} \end{aligned}$$

NPTEL OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 43

Then following this **this** reference the idea here, is to non dimensionalize the entire problem, so that the problem can be more and more **actually**. So, you take care of that first, we normalize I mean define this non dimensional quantities and starting from this we **actually**, write this equation now $m \dot{m}$ prime means $\frac{dm}{d\tau}$ basically but, how we define something like this.

So, we write it the same system dynamics in an non dimensional quantities something like this. And then remember there is a difficult problem, because we do not want to excide this control and think like that **actually** this control **actually** that way. So, we have to I then again going by this idea which applied here in this **this** literature, (Refer Slide Time: 49:00) the idea here is to convert this free final time problem to some sought of a state constraint problem with **with** bounds on control.

And then equation of motion are formulated using flight path angle as independent variable basically. What you are assuming here is the turning happens in one direction only; that means, the flight path angle is monotonic **actually**.

So, if you assume that then you can take the flight path angle as independent variable. And this leads to this **this** fixed final condition, where the initial condition of gamma I know and final condition is minus 180 degree. So, it happens to be fixed final condition now **actually**; for fixed final time sought of thing now **actually** that way. So, that becomes solutions **(O)** using this final time optimal control theory basically. So, the entire assumptions for do that flight path angle is monotonic and continuous function with respect to time.

So, that is the inner and assumptions based on which can attempt to solve this problem (Refer Slide Time: 50:05). So, now, when gamma is independent but, it is you have this **this** first define this d m by d gamma something by something m prime by gamma prime d t by d gamma 1 by gamma prime like that **actually**. And then we this modified cost function is something given like this 0 to t f is equally written something like d gamma integration of d gamma. But, d gamma it is something like d t has to be converted as d gamma. So, d t by d gamma, then this is 1 over d gamma by d t and think like that. So, d gamma by d t is available from here supported here.

So, j is available essential of a complex function, but ultimately it is **it is** time minimizing cost function **actually**. And then the, this is hard constraint, so m of gamma of is to 0.8 **actually**.

(Refer Slide Time: 50:45)

Nonlinear programming problem

Minimize $J^N = \frac{t_f - t_0}{2} \sum_{k=0}^N w_k \frac{a \hat{M}(\gamma_k)}{g(S_w [\hat{M}(\gamma_k)]^2 C_L + T_w \sin(\hat{\alpha}(\gamma_k) + \hat{\delta}_o(\gamma_k)) - \cos \gamma_k)}$

Subject to $\sum_{k=0}^N a_k T_k(\gamma_k) = \frac{-S_w [\hat{M}(\gamma_k)]^2 C_D + T_w \cos(\hat{\alpha}(\gamma_k) + \hat{\delta}_o(\gamma_k)) - \sin \gamma_k}{g(S_w [\hat{M}(\gamma_k)]^2 C_L + T_w \sin(\hat{\alpha}(\gamma_k) + \hat{\delta}_o(\gamma_k)) - \cos \gamma_k)}$

Subject to path constraints

$\hat{M}(\gamma_k) = \sum_{i=0}^k a_i T_i(\gamma_i); \hat{\alpha}(\gamma_k) = \sum_{i=0}^k \alpha_i T_i(\gamma_i); \hat{\delta}_o(\gamma_k) = \sum_{i=0}^k \delta_{oi} T_i(\gamma_i)$
 $\hat{M}(\gamma_0) = \sum_{i=0}^0 a_i T_i(\gamma_i); \hat{M}(\gamma_N) = \sum_{i=0}^N a_i T_i(\gamma_i)$
 Grid points, $\gamma_i, k=0, \dots, N$

with end point conditions

$e_0(x_0) = \hat{M}(\gamma_0) = M_0$ where $M_0 \in [0.3, 0.8]$

$e_f(x_f) = \hat{M}(\gamma_N) = 0.8$

OPTIMAL CONTROL, GUIDANCE AND ESTIMATION
46

And then go through this **this** long algebra of discretization discretized mathematics and minimize this cost function rule. Subject to this, **this** derivative approximation; subject to this equality constraint equation; subject to this path constraint also that alpha cannot be more than 20 degree and delta cannot be more than 72 degree, that is what I told delta can be much more than the alpha basically **ok**.

So, and essentially I mean this end point conditions are direct and 0 can be 0.3 to 0.3; however, m f has to be 0.8 that is the problem **actually** (Refer Slide Time: 51:26). So, these are some of the experiments, we tried out with various different phases and all that. So, remember no matter whatever is initial condition of Mach number, the final Mach number turns out to be 0.8 **actually**.

So, this is very nice observation first thing to see and if you see the kind of height verses down range sought of thing this is the trajectory that you see. So, this is last function somewhere here and it turns out I mean it nicely turns, where you want **actually** the turning happens nicely basically **ok**.

Then, we have this **this** angle of attack constraint, which is also more than twenty is not allowed something like that, this **this** constraint is coming in the picture, so that is enforced here. And **and** this all within the bound **actually**, whereas, delta b which is also this thrust control mechanism sought of thing this is **actually** helping that to **to** remain within the bound **actually**. So, delta b turns out to be like this all are smooth trajectories that also, and then the point here is the final mach number is 0.8 no matter where you start; so velocity drop is not **not** there **actually**.

Then the mach number verses angle of attack something like mach number verses flight path and angle of attack verses flight path. All the things you can plot in different **different** variable sense and adds lot of more meaning **actually**. And remember this has to be seen from the right to left **actually**. No matter what you start you all end at the 0.8 flight path angle evolves from 0 to minus 180 degree.

So, this plot have to be seen in the reverse right to left sense **actually**, this also has to be seen right to left sense angle of attack how it evolves as the trajectory flight path angle per time how it evolves **actually** then this always has this 0 to minus 180. But, depending on different cases different things are then how do you generate this plot by the way.

Because, once flight path angle becomes independent quantity **ok**, then time becomes some sought of a state equation.

So, $d\gamma/dt$ equation is also put into the problem formulation, so corresponding to particular gamma you have the corresponding value of time **actually**, so both are **both** cannot be independent quantities only one has to be independent quantity. So, according to the **according to** that the other one varies **actually**. So, that is how the time is generated and it is plotted something like this here. So, now you can think effect of reducing angle of attack and think like that. So, this **this** depends on the angle again this **this** is more that can be less and **and** think like that **actually ok**.

Now, (Refer Slide Time: 54:26) this what is here shear angle and angle I some of these things are aerodynamic concepts you can see these are this is with respect to the body is this is the body angle δ_b and with respect to the velocity vector. This is a different angle that is shear angle **actually** velocity vector and thrust vector, that can be even higher than the **actually** according to that. I **alright**, so this is what it is **actually**, then effect of the reducing angle of attack along with the flight path, I mean the all details are there in the paper you can read some of the information there also.

So, let not too much elaborate here, it is all happening in a in a good sense you can plot flight path verses time, how it evolves for math number verses time. How it goes from 0.5 to 0.8 depending on different cases **actually**. That means, we put various bounds and all for we have this initial bound is some 20 degree, now we put various bounds on that and depending on that the other things gets **actually**.

Now **now** the question is how it is a function of a grid points now. So, initial start with the course grid points only not good, you can see some of these things are to quite looking sought of things. So, but then this history implies, when use that as a guess history and then go for five in grid and all that **actually**. So, 5 to 10, 10 to 12, 12 to 20, 30 like that, now you can observe that beyond 20 is not much of an issue. So, going beyond 20 is no point **actually**. So, we convert we take that 20 grid points are of this solution and we leave it the 20 grid point solution basically.

(Refer Slide Time: 56:08)


Selection of number of grids

No. of grids	Computational time
5	0.899679
10	1.022969
12	1.102450
20	1.652399
30	3.314869

Intel® Core™ 2 Quad CPU
Q6600 @2.40 GHz, 1.98GB RAM
Software: MATLAB 7.4

- The number of grids for analysis: 20
- 20 grids are comparable with 30

Note: Real-time implementation in "C" or "Assembly language" is expected to be much (at least 50 times) faster than the MATLAB Code



OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 53

So, you can see the number of grid point verses computational time. And all that if your five; obviously, your computational time is very small, there are all evaluated in this hardware settings which is **which is** a regular with mat lab 7.4 and all that. Not a dedicated procedure the whole idea is if you have a real **real** time normal processor with a dedicated processor all that this and your program is written terms of missile level language then it can be much faster than that **actually**.

Alright so, but even this setting we have number of grid point think five this is the computational time and 10, 12, 20 it starts increasing. But, beyond a point is not beyond 20, we do not recommend, so we **we** reject here **actually**. So, the real time implementation is c or assembly language is expected to be much faster and hence it is thing that can implemented in real time **actually ok**.

Now, what was the idea of chebyshev and legendre it turns out that it takes a chebyshev polynomial or legendre polynomial both happens to be like giving us the same result **actually**. So, the one way or other there is no advantages for say basically that that exercise tells us. So, another consistency check also you can think about that **actually**.

So, (Refer Slide Time: 57:18) conclusions for the missile turning problem is **are** like this the hemisphere engagement if feasible. So, physically speaking in the no need of dog fight provided of course, you can you can implement these **these** high shear angle values

and all that, this **this** values, what you are seeing here what is delta b or c angle. So, thrust control augmentation that has to be implemented.

Now, that has to be seen subject to that condition it is possible to see have this conclusion basically. Minimum time fight path angle reversal is feasible with **with** realistic control force that is what we think and then promising numerical results; that means, computationally they are efficient and viable tool for optimal guidance for say and chebyshev and legendre polynomial lead to the identical results **actually**.

So, this gives some **some** degree of confidence that pseudo spectral method can be used in various application. As I told before there are various other things that Michael Ross and I M Ross and this group **actually** demonstrated. There are other groups also working on the pseudo spectral method but, this team has **actually** demonstrated variety of problems and including hardware implementation and implementation in **in** international space station also. So, some of these references you can see for more details and my recommendation will be probably the first one, which appeared very recently 2012 with some sought of a very good overview of the method itself and from to flight and all that **actually**.

So, if you can read some of these papers it is also written in a way that is not too much mathematically involved. Whereas, other papers can be little bit mach **math** overwhelming also basically now this is very written where relatively long paper and summarizes the when the concept in necessary mathematic sense and all that **actually**. So, my strong, I mean strong recommendation is that you should read this paper and understand more **actually**. **Alright** so, this is what I wanted to talk in this particular lecture further more and at what we have developed on these ideas and all that calling this M P S P method and M P S C subsequent things and think like that, we will discuss in the in the next **next** lecture **actually**, **alright**, thanks a lot.