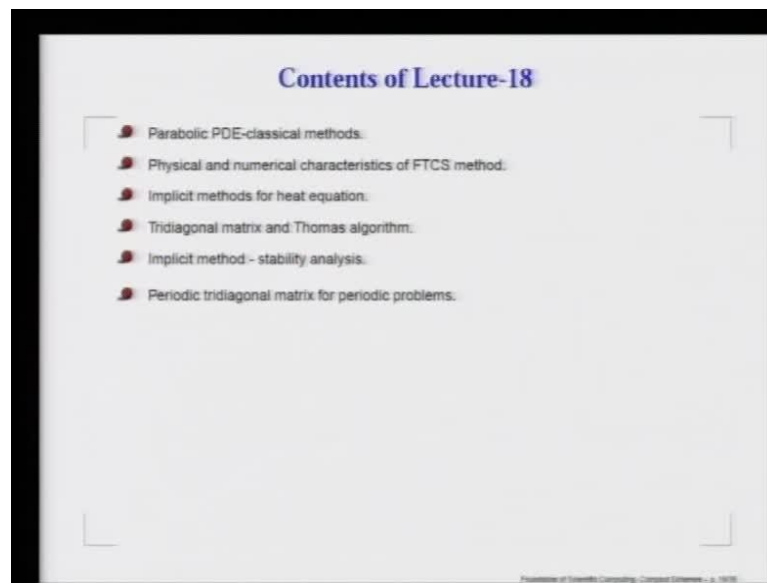


Foundation of Scientific Computing
Prof. T. K. Sengupta
Department of Aerospace Engineering
Indian Institute of Technology, Kanpur

Module No. # 01

Lecture No. # 18

(Refer Slide Time: 00:16)

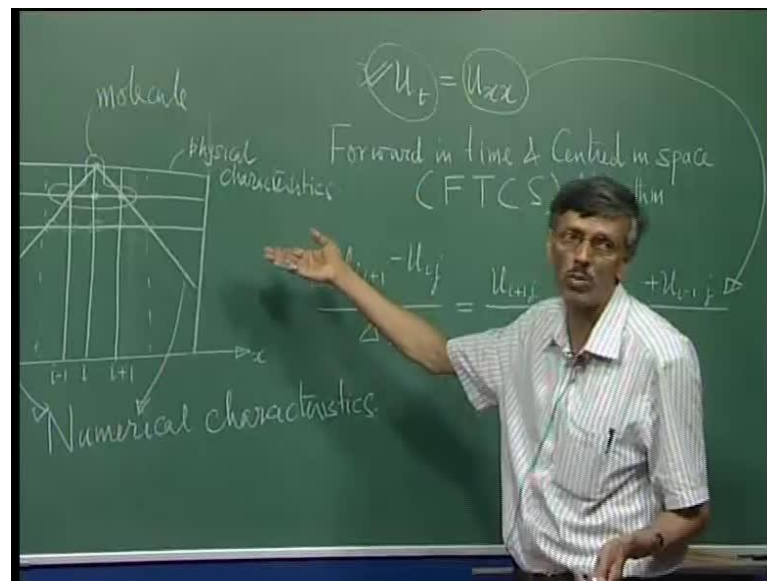


In today's lecture, **eighteenth**, we continue our discussion on solving parabolic partial differential equation by classical methods. These are essentially explicit methods and one of the properties of this explicit method, is that it brings in its own numerical characteristics, as opposed to the physical characteristics. And as demonstration, we are going to show the FTCS method, and we will see that it brings in significant restriction on the time steps and one of the way to avoid this problem of time step restriction is to switch over to implicit methods, and this is demonstrated today, by again, considering the heat equation.

One of the disadvantages of this implicit method is that we will have to be now solving matrix equation, but fortunately enough for the heat equation, this comes to be a tridiagonal matrix and which yields an exact solution by using Thomas algorithm.

Having adopted the implicit method, one would like to once again go through that stability analysis by the spectral analysis tool, and then we will establish its stability and its enhanced ability to take larger time step. However, at the same time when we have periodic boundary conditions or the periodic problems, we need to converge this tridiagonal matrix into its variation, called the periodic tridiagonal matrix and this will be studied in detail.

(Refer Slide Time: 02:04)



We are solving parabolic partial differential equation and today we just demonstrated (()) first taking a one dimensional heat equation as given here - partial derivative of u with respect to time is equal to the second derivative of u with respect to x , and this is defined in a x - t plane with spacing indicated by index i along the x direction and j along the time direction. And we specifically talk about a simplest possible explicit algorithm which is called as forward in time centered in space or FTCS algorithm, and that involves taking this time derivative u of t ; I am writing it in this form on the left hand side.

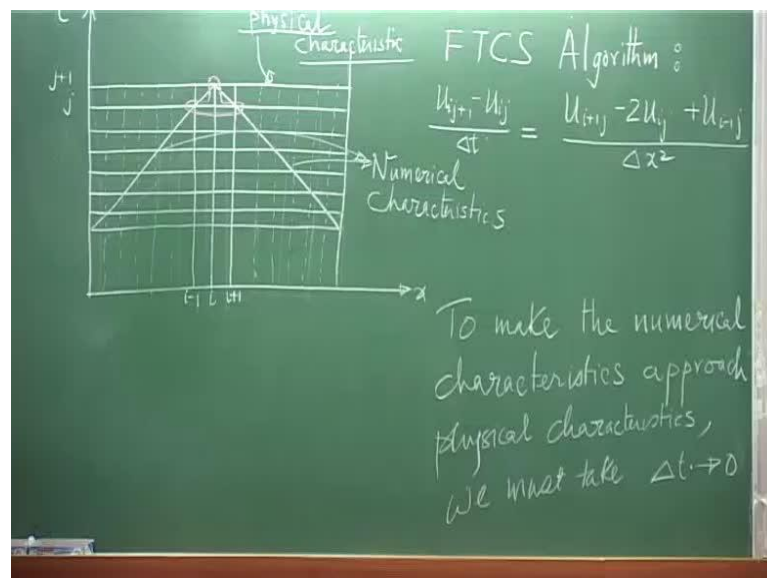
The same way, we can take the second derivative of u with respect to x on the right hand side and represent it by this difference expression. Having done that, we note that the points involved in this algorithm are given here by this dotted points, that is, $u_{i,j+1}$, which is at the advanced time level - j plus 1th time level, depends on other three points which are given on the line below.

In the language of competition, this participating node points constitute what is called as the computational molecule; so, this is what we are going to call as the computational molecule.

Now, in looking at this problem we know that this is a parabolic partial differential equation, so its physical characteristics are nothing but t equal to constant and this is what we have as the physical characteristics. Whereas we note that the node at the advance time level actually depends on this three points, and this three points again in turn depend on their corresponding three dependent points, that is, this one, this one and this one and for this - this point, this point and this point.

So, what actually means, then this point eventually depends on a cone which is defined by a line like this. The extremities of the cone is given by this and these are what are called as the numerical characteristics. Thus, we understand one interesting attribute of how computing is the distinction between the physical and the numerical characteristics. This is something very fundamental that we will be visiting again and again and we will point out that one of the goal for the accuracy of the solution depends on, how we bring this numerical characteristics as close to the physical characteristics as possible?

(Refer Slide Time: 05:42)



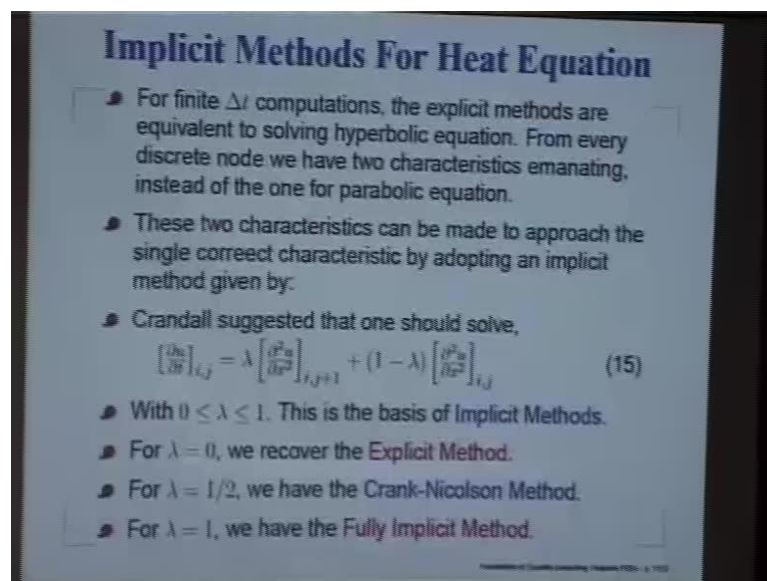
So, this is a single one, but here we end up getting two branches, that is what we get now. You can very clearly see, that to make, to **think** work for you, you should have these two sets of characteristics approach each other.

How do you do that? A very simple observation is if I reduce delta t, then what will happen? This wedge will open up towards the theoretical limit, the physical characteristics; so, that is what we can do.

So, the way to do that, so may be if I have to make this observation, delta t should be as small as possible; so, that is the rule of the game, that is one of the way. But you already know from your experience in computing, what you have done just now I am going to submit today, that taking smaller time step also makes the process **very** very slow, so that is not a very good option.

That brings us to the question of trying to find out if there are other methods, which we will do that as desirably as possible, at the same time would not consume too much of time and resources.

(Refer Slide Time: 07:54)

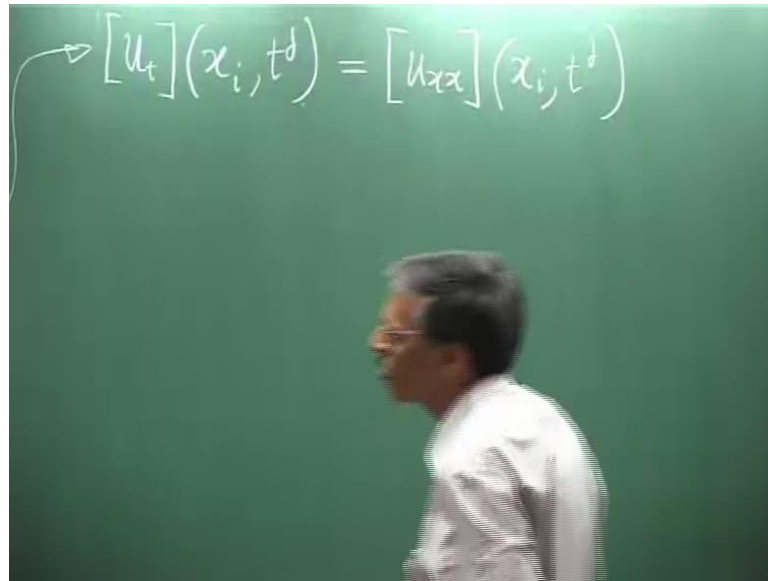


Implicit Methods For Heat Equation

- For finite Δt computations, the explicit methods are equivalent to solving hyperbolic equation. From every discrete node we have two characteristics emanating, instead of the one for parabolic equation.
- These two characteristics can be made to approach the single correct characteristic by adopting an implicit method given by:
- Crandall suggested that one should solve,
$$\left[\frac{\partial u}{\partial t} \right]_{i,j} = \lambda \left[\frac{\partial^2 u}{\partial x^2} \right]_{i,j+1} + (1 - \lambda) \left[\frac{\partial^2 u}{\partial x^2} \right]_{i,j} \quad (15)$$
- With $0 \leq \lambda \leq 1$. This is the basis of Implicit Methods.
- For $\lambda = 0$, we recover the Explicit Method.
- For $\lambda = 1/2$, we have the Crank-Nicolson Method.
- For $\lambda = 1$, we have the Fully Implicit Method.

In this context, we start talking about implicit methods and you can see the motivation actually comes from this diagram that we have here. We want this physical characteristics and the numerical characteristics approach each other and how do you do it?

(Refer Slide Time: 08:27)

A man in a white shirt is standing in front of a green chalkboard, pointing with his right hand towards a mathematical equation written on the board. The equation is
$$[u_t](x_i, t^j) = [u_{xx}](x_i, t^j)$$

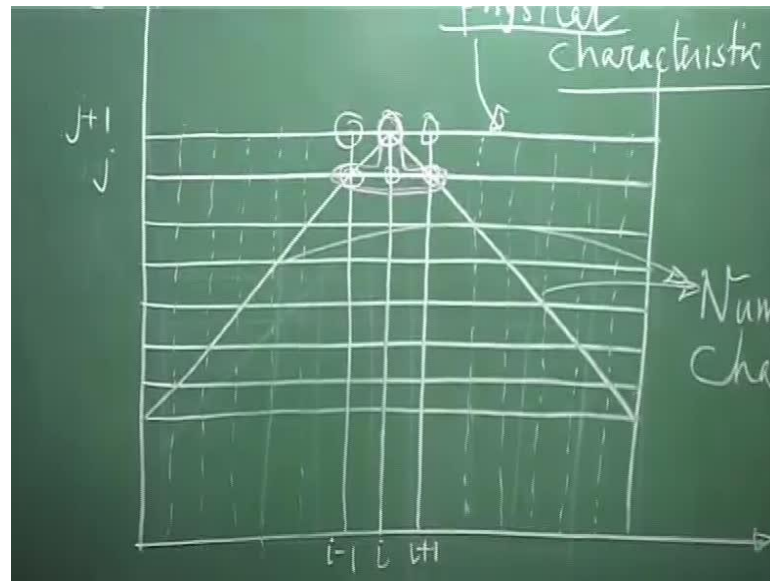
The chalkboard is green and the equation is written in white chalk. The man is seen from the side, wearing glasses and a white shirt.

That is noted here, that what we have done here when I wrote this FTCS algorithm, what I did? I evaluated this at say some node, which I called as x_i and I try to get the time level, which I called as t^j and what did I do is, I wrote this as u_{xx} evaluated; now, u_{xx} is evaluated at what level? At the j th level also.

So, that is also written for the i th node and this and as a consequence we had this wedge. Now, what we could do is, if I do the following, like what we have written down here in equation 15, that we will take the right hand side evaluated at two time levels - one is at the j th level and suitably blended with second derivative evaluated at the prior level.

So, basically then what we are doing? We are **doing**, rewriting the right hand side as a weighted average. How is the weighting done? λ has to be between 0 and 1 and then if I do this, what I am doing actually?

(Refer Slide Time: 10:04)



The first term, if I look at it, **put** involve these three points because I am evaluating that at the j plus 1th level, so that is what we are going to do. Whereas this $1 - \lambda$ times $u \times x$ are those three points that we had before.

So, what we are doing? We are now, instead of having a computational molecule involving four points, we are having it in terms of **6 points**, 6 points and what we are doing then? Instead of taking the value which defines the molecule at this three points, we are taking some kind of a weightage average between this point and that point, that point and that point and that this. So, this is the sequence.

So, what happens is, depending on the choice of λ we could do many things. For example, if I would set λ equal to 0, that is, I do not evaluate the derivative at the advance time level, get everything from the previous one, then we go back to our FTCS algorithm that we can very clearly see; so, this will be just simply that what we have written over there.

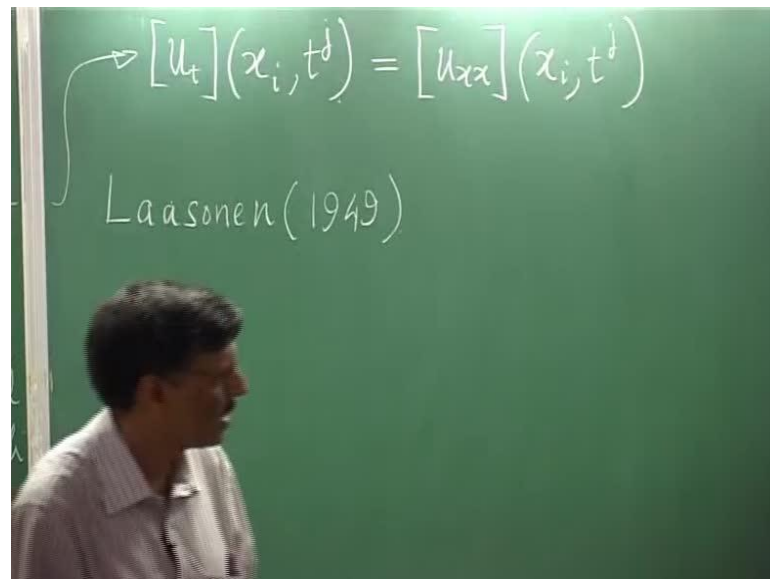
Now, this is what two gentlemen from Cambridge - Crank and Nicolson - they did it in mid-forties. What they say that take a kind of an unbiased average, take half and half contribution coming from each time level; that is your λ **equal to half means**.

So, I will take the second derivative evaluate it at the advance time level - fifty percent and the rest of it is coming from the previous time level.

Now, we could also, of course it is a whole continuum, you could choose a value of lambda anywhere between 0 and 1, but we are looking at some very specific cases and one of which, if I look at is, if I take lambda equal to 1. Now, of course, this terms switches off and all the contribution of the derivative would come from the advance time level.

So, basically, then you are going to actually have a computational molecule having this three points only and then what you see - the numerical characteristics is completely aligned with physical characteristics.

(Refer Slide Time: 12:48)



So, that should be the quite a desirable case and I think this was done by, attributed to Laasonen, if my source is correct, and this was I think done sometimes in that era; Crank and Nicolson suggested this sometime around 46 or 47. And this whole approach of blending the two was suggested by Crandall of MIT and then, that was a big game changer in the computing business. That we understand that there are better methods that we could do that will mimic physics better, and in the process, you could get your computational methodology also quite efficient and fast.

(Refer Slide Time: 13:39)

$$[u_t](x_i, t^j) = (1-\lambda)[u_{xx}](x_i, t^j) + \lambda[u_{xx}](x_i, t^{j+1})$$

$$\frac{u_{i,j+1} - u_{i,j}}{\Delta t} = \frac{(1-\lambda)}{\Delta x^2} [u_{i+1,j} - 2u_{i,j} + u_{i-1,j}] + \frac{\lambda}{\Delta x^2} [u_{i+1,j+1} - 2u_{i,j+1} + u_{i-1,j+1}]$$

$$Pe = \frac{\Delta t}{\Delta x^2}$$

$$\theta = Pe \lambda$$

Now, what does it do? What does it do is, it actually... Now, what we had done here as I had written in the slide before, so we are going at $u \times x$ evaluated at... ,so this is what we are writing. So, if we are writing that then, of course we can expand it and if I do, what I am going to get here is $u_{i,j} + 1$ minus $u_{i,j}$ delta t ; so, this is again Euler time integration, so, this is till the Euler time integration that we are resorting to. All the balancing or certainty is coming from how we treat the special derivatives at different time level.

If I of course, write this I will get... If I call that as Δx square and then I will get $u_{i,j+1} - u_{i,j}$ plus 1 minus 2 $u_{i,j}$ plus $u_{i-1,j}$, that is this part and from here I will get λ by Δx square and I will write down the same stencil, but now it is evaluated at j plus 1 th level; so, this is what we are going to get.

Now, quite like what we did for the convection equation, we can see some parameters are naturally coming out, that we already have defined and we define the peclet number. If you recall, in the last class that was simply the ratio of the time step by this.

Now, in addition we also have brought in a extra degree of freedom in terms of this parameter λ , so we could define a new quantity, which are called, that is the θ will be nothing but peclet number times λ .

So, that is simply nothing but $\lambda \Delta t$ over Δx square.

(Refer Slide Time: 16:47)

Implicit Method - Difference Equation.

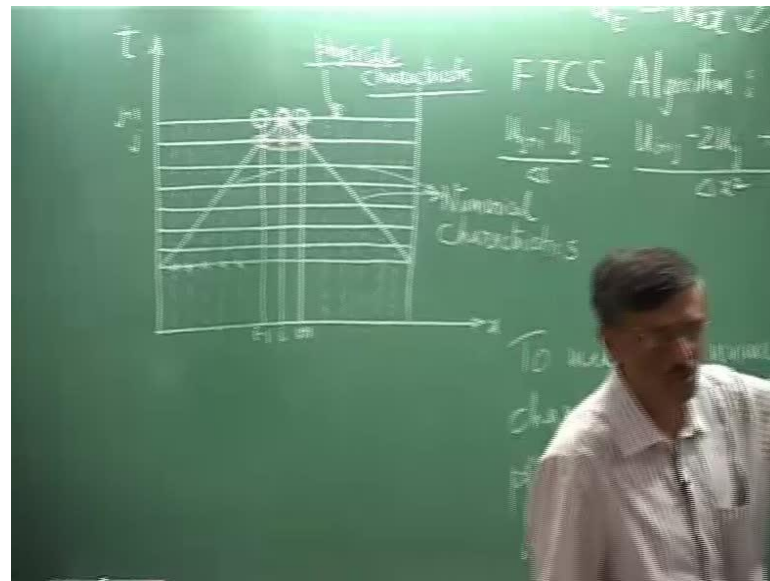
- The difference equation for (15) is given as,
$$-\theta u_{i-1,j+1} + (1 + 2\theta)u_{i,j+1} - \theta u_{i+1,j+1} = (Pe - \theta)u_{i+1,j} + [1 - 2Pe + 2\theta]u_{i,j} + (Pe - \theta)u_{i-1,j} \quad (16)$$
where $\theta = \lambda Pe$.
- Above is a coupled set of equations, when written for all the interior nodes, at any time level. This constitutes an algebraic equation,
$$[A]\{x\} = \{b\} \quad (17)$$
- $[A]$ is the sparse TRIDIAGONAL matrix with its entries given by the coefficients on the left hand side of (16). The vector $\{b\}$ is obtained from the right hand side of (16) and the applied boundary condition. The vector $\{x\}$ is composed of $u_{i,j}$'s.
- The above equation is solved exactly by Thomas' Algorithm- as described next.

Frontiers of Computer Science, Vol. 1, No. 1, 2007

So, you can immediately complain that I could simplify this and I could write down all the quantities which are at the advance time levels on the left hand side, that would involve this term and all this three terms, **those are written down here on the left hand side**, those have been written down there on the left hand side and put everything at the previous time level on the right hand side; this basically gives you this equation.

So, unlike the explicit method, you realize that now you cannot explicitly pick up the value of the unknown one at a time. What is happening now, your unknowns are all at the j plus 1th level, but they are appearing in a coupled manner, that is, the i minus 1 i and i plus 1 they are taken together.

(Refer Slide Time: 17:43)



So, if I look at that, what does it do? Well, you can very clearly see that you cannot solve it in isolation, so what we are going to do is **will keep...** Suppose I have a **(())** like condition, boundary conditions, so that is, u is prescribed on this two ends - at x equal to 0 and x equal to 1, they have prescribed, then I write down that equation for all the points.

So, if I keep doing it then what do I get? I am going to get a stack of equations. The first one will involve, if I look at it here, the first one will involve this point, this point and this point, so 1 2 and 3. What happens to 1? It is a divisional condition, it goes to the right hand side.

(Refer Slide Time: 18:27)

The chalkboard shows the following derivation:

$$\frac{u_{i,j+1} - u_{i,j}}{\Delta t} = \frac{(1-\lambda)}{\Delta x^2} [u_{i+1,j} - 2u_{i,j} + u_{i-1,j}] + \frac{\lambda}{\Delta x^2} [u_{i+1,j+1} - 2u_{i,j+1} + u_{i-1,j+1}]$$

Below this, the parameters are defined:

$$Pe = \frac{\Delta t}{\Delta x^2}$$

$$\theta = Pe \lambda$$

Then, the equation is rearranged for $i=2$:

$$-\theta u_{1,j+1} + (1+2\theta)u_{2,j+1} - \theta u_{3,j+1} = rhs_2$$

For the boundary condition at $i=2$, the right-hand side is augmented:

$$(1+2\theta)u_{2,j+1} - \theta u_{3,j+1} = rhs'_2$$

Similarly, for $i=3$:

$$-\theta u_{2,j+1} + (1+2\theta)u_{3,j+1} - \theta u_{4,j+1} = rhs_3$$

So, we are going to have a linear algebraic equation that has the diagonal term $u_{2,2}$ and say $u_{2,3}$ or it is like this. So, basically what I am saying that I will write this equation, let us say for any j ; so, there is no problem, but I will write it for say, i equal to 2.

If I write it for i equal to 2, then I am going to get minus θ and $u_{1,j+1}$ plus 1 plus 1 plus 2θ $u_{2,j+1}$ minus θ $u_{3,j+1}$, and everything that has gone on the right hand side, I will just simply write it as rhs_2 ; so, that is the whole contribution coming at the point two by right hand side.

Now, of course you can see if it is a divisional boundary condition, this can also be transported to the right hand side; so, that is what I said, that this equation would simply have 1 plus 2θ and I will have $u_{2,j+1}$ minus θ $u_{3,j+1}$. And I will write this rhs_2 and put a prime to indicate the boundary condition has augmented that what we had earlier on.

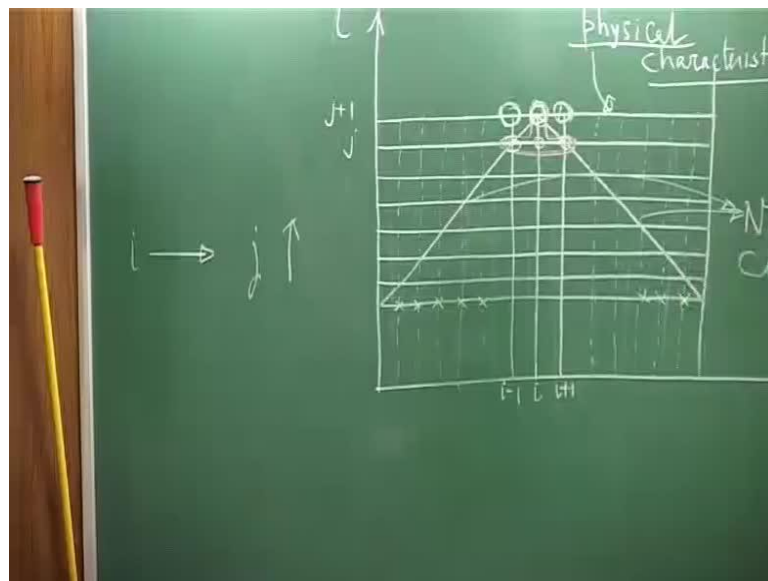
Now, I could write the same thing for i equal to 3 and now, you will notice that this term of course would be there, so, **you do not have to**, you do not need to take it to the right hand side; it is an unknown. **So, if that is so...**

And this I am going to write rhs_3 , so I could write all that. So, what I am going to do is I am going to start that equation, I will start writing for i equal 2 then i equal 3 and write it for all the unknown points, all the way up to here.

So, that is the way the equations stack together and when you put them together, you will get a linear algebraic equation like this, where a would have the elements which are given here. So how many non-zero elements that you get? Only three.

So, you will get the point corresponds to whatever the i value that you have chosen, that is here, that is I will get it as along the diagonal and this term will go on to the super diagonal, that is the next point in hierarchy.

(Refer Slide Time: 21:36)



So, basically what we are doing, we are following this sequence that i will go from left to right and j will go from bottom to top. So, if I do that, that is what we are doing. We are fixing a value of j and then writing all the quantities from left to right and that is why we are getting this equation. a are those coefficients of that left hand side, x is the unknown vector, so the vector will be like this say, $2j + 1$, $3j + 1$, $4j + 1$, so that is a stack.

So, I am just simply, thus, economizing on space and calling it as x and this rhs that you have identified here, they all go to the right hand side. So, it is a very simple sort of a migration, that you can say that we have gone from an explicit method into a business of tackling a linear algebraic equation of this kind which is $a x = b$, but a as a very neat structure; I think, I have it here that we will tell you what it is, it has a structure like this.

(Refer Slide Time: 22:44)

Solving Tridiagonal Matrix Equation

The matrix equation (17) is solved by LU-decomposition of A matrix in Thomas Algorithm by writing,

$$[A] = [L][U] \quad (18)$$

where A matrix is tridiagonal and has the form,

$$\begin{bmatrix} b_1 & c_1 & 0 & 0 & 0 & \dots & 0 \\ a_2 & b_2 & c_2 & 0 & 0 & \dots & 0 \\ 0 & a_3 & b_3 & c_3 & 0 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & 0 & a_{n-1} & b_{n-1} & c_{n-1} \\ 0 & 0 & 0 & 0 & 0 & a_n & b_n \end{bmatrix}$$

$[L]$ is the lower di-diagonal matrix and $[U]$ is the upper bi-diagonal matrix, as shown next.

(Refer Slide Time: 23:07)

$$\frac{u_{i,j+1} - u_{i,j}}{\Delta x \Delta t} = \frac{(1-\lambda)}{\Delta x^2} [u_{i,j} - 2u_{i,j} + u_{i-1,j}] + \frac{\lambda}{\Delta x^2}$$

$Pe = \frac{\Delta t}{\Delta x^2}$
 $\theta = Pe \lambda$

$$- \theta u_{i,j+1} + (1+2\theta) u_{2j+1} - \theta u_{3j+1} = \chi s_2$$

$$(1+2\theta) u_{2j+1} - \theta u_{3j+1} = \chi s_2'$$

$$i=3: -\theta u_{2j+1} + (1+2\theta) u_{3j+1} - \theta u_{4j+1} = \chi s_2'$$

$\frac{1}{a_i} \quad \frac{1}{b_i} \quad \frac{1}{c_i}$

So, you have that diagonal elements and you have a super diagonal elements and you have a sub diagonal elements. So you can very clearly identify, so these are your like a i's, these are your a i's; so, if I have to write, so this will be my a i and this is b i and this is my c i.

So, for every point I am going to get this. Well, you might think why I am giving a subscript here, it is not needed, it is basically constant, for all ith point it is same.

But, let us discuss for a general case where you may have these things which could be non-dependent. Yes, let's discuss such a case and let me tell you that this equation despite its simple appearance, is a central methodology that is used in computing very often. You would come across this procedure of solving these tridiagonal matrix equations.

So, this is why we are calling it tridiagonal, we have the diagonal super and sub diagonal, so it constitutes three non-zero stacks, diagonal **arise**; so, that is why these are called tridiagonal matrix.

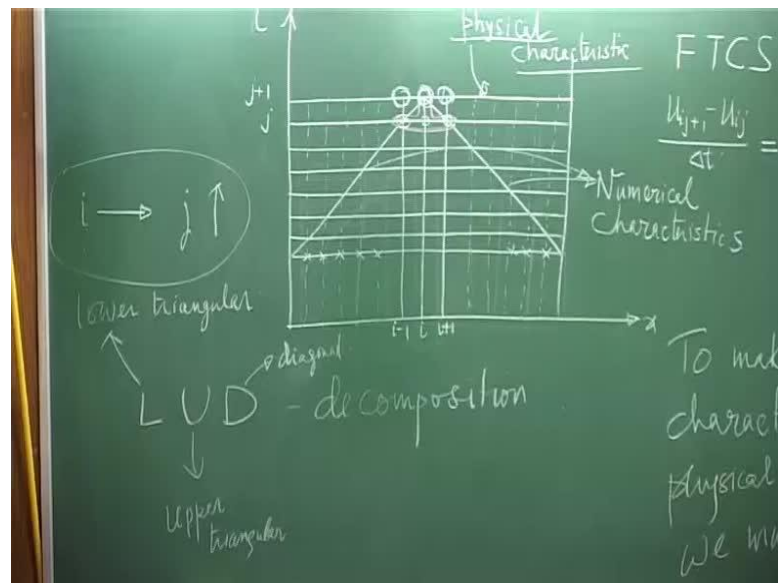
So, that is your structure of the A matrix and x vector. I noted to you how it is going to be accounted for, so will fix a j run through all the i 's from left to right, then migrate to the next j and so on, so forth.

So, this is the sequence that we will be following in cataloging this unknown x . So, whenever you have matrices with lots of 0s they are called sparse matrices and this particular A matrix is not only sparse, it is also tridiagonal matrix.

So, what happens is, good news in this story is that this equation can be solved exactly and this is what Thomas did, and it is well known as Thomas algorithm and this is what we are going to now discuss - how we solve this tridiagonal matrix equation?

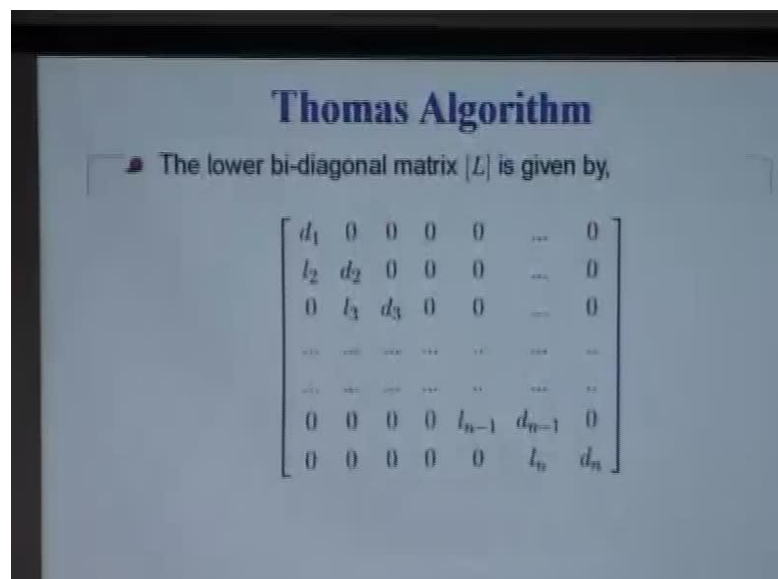
Well, the logic is simple. I have a matrix A , I could split it into two matrices - one corresponds to a lower triangular matrix, another would be upper triangular matrix. So, this is a very standard technique that you actually employ in linear algebra that if you are given a matrix equation with arbitrary A , what you try to do? You try to split it up into three parts like it is called LU-decomposition.

(Refer Slide Time: 25:57)



So, what you have is the basically lower triangular matrix and this is a upper triangular matrix and this is a diagonal matrix. So, in this case what has happened? We have dispensed with the d path; we are just simply writing it as the product of the lower triangular and an upper triangular matrix and let me just tell you how this is formally done.

(Refer Slide Time: 26:55)



Because of that sparseness, so many zeros all over, what we could do is, we could write down this term lower triangular matrix is like this; so, above the diagonal it is all 0, that is why, it is called lower triangular.

But because of the sparseness, what we notice also that even in the lower triangular matrix also, you have a large number of zeros barring only the diagonal and the sub diagonal entries.

(Refer Slide Time: 27:36)

Thomas Algorithm

The upper bi-diagonal matrix $[U]$ is given by,

$$\begin{bmatrix} 1 & u_1 & 0 & 0 & 0 & \dots & 0 \\ 0 & 1 & u_2 & 0 & 0 & \dots & 0 \\ 0 & 0 & 1 & u_3 & 0 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & 0 & 0 & 1 & u_{n-1} \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

- There is an alternative splitting of $[A]$ matrix, where diagonal entries of $[L]$ matrix are unity, while $[U]$ matrix has diagonal entries different from one.
- Multiplying the $[L]$ and $[U]$ matrices and equating the product with $[A]$ matrix, provides the entries of $[L]$ and $[U]$ matrices.

Foundations of Scientific Computing / Prof. Dr. P. K. K. S. - p. 2527

So, if I decide to call those entries as d_1 in the first row, then l_2, d_2 in the second and l_3, d_3 in the third and so on, so forth, so I have this formal structure. So, it would be our interest to find this out, I will tell you how we are going to find it out, but let me first tell you what we can do with the upper triangular matrix. Upper triangular matrix will also have a similar structure by analogy.

The lower triangular part will be 0 completely and the diagonal part here, I could have some value, but what I have done here? I have basically divided that, but the diagonal path, so that the diagonal entry has become unity and the super diagonal entries are written as u_1, u_2 up to u_{n-1} .

Now, how do I find these entries l and u lower case quantities? Well, simple thing for you to do is take this l and u matrices or take the product and equate it with what you have with u matrix entries.

(Refer Slide Time: 28:34)

THOMAS ALGORITHM

$$\begin{bmatrix} c_1 & b_1 & 0 & \dots & 0 \\ a_2 & b_2 & c_2 & & \\ 0 & a_3 & b_3 & c_3 & \\ & 0 & a_4 & b_4 & c_4 \\ & & 0 & a_5 & b_5 \end{bmatrix} = \begin{bmatrix} d_1 & & & & \\ l_2 & d_2 & & & \\ & l_3 & d_3 & & \\ & & l_4 & d_4 & \\ & & & l_5 & d_5 \end{bmatrix} \begin{bmatrix} 1 & u_1 & & & \\ & 1 & u_2 & & \\ & & 1 & u_3 & \\ & & & 1 & u_4 \\ & & & & 1 \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \end{bmatrix}$$

$[A] = [L][U] \rightarrow \text{upper triangular}$

Tridiagonal lower triangular

Next, we are going to talk about special algorithm called the Thomas algorithm which is used to solve in a direct fashion, a tridiagonal matrix which we have written here as the a matrix, which has three entries - branded entries b i's along the main diagonal, c i's along the super diagonal, and a i's along the sub diagonal. And one of the way in which this equation can be very easily solved, this matrix equation can be easily solved is by decomposing this tridiagonal matrix into a product of a lower triangular matrix times and upper triangular matrix given by this.

Please note the structure of this lower triangular matrix which has d i's along the diagonal and l i's along the sub diagonal elements, rest of the elements of this matrix is identically 0. Also note, that the upper triangular matrix has one along the diagonal and u i's along the super diagonal.

This is one possible way of decomposing the a matrix into l times u, but we could also do a complementary splitting where we could have one along the diagonal of the l matrix whereas the diagonal entries of the u matrix would be not equal to 1.

(Refer Slide Time: 31:10)



Both are essentially equivalently the same thing, but it is easy for **us** to realize that we can work out directly the entries of this l and u matrix by performing a product and for example, if I look at this entry a_{11} , that is, this b_{11} is nothing but product of this first line of l matrix multiplied by the first column of the u matrix and that is going to give us simply as **b_{11} , sorry b_1** , b_1 is equal to nothing but d_1 .

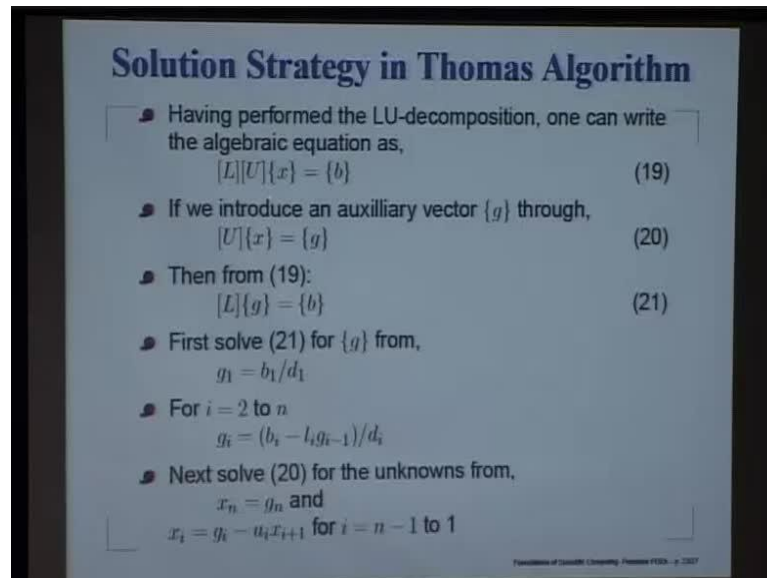
Similarly, the second entry of the a matrix c_1 would be nothing but product of the first row with the second column, that means, c_1 would be equal to nothing but $d_1 u_1$.

So, we can very clearly see that u_1 is nothing but equal to c_1 by b_1 itself. So, basically b_1 is defined like this and u_1 is defined like this. Now, we can go through this exercise on the second row of a matrix, for example, a_2 would be the product of the second row of l matrix multiplied by the first column of the u matrix.

So, that means a_2 would be simply equal to l_2 . Now, if I look at the entry b_2 of the a matrix, that will be a multiplication of the second row of l matrix multiplied by the second column of the u matrix.

So, that means i will get b_2 should be equal to $l_2 u_1$ plus d_2 and finally on the second row of a matrix, we have the entry c_2 and that would be **multiplied**, obtained by multiplying the second row of l matrix with the third column of the u matrix, and that would be nothing but equal to $d_2 u_2$.

(Refer Slide Time: 35:22)



Solution Strategy in Thomas Algorithm

- Having performed the LU-decomposition, one can write the algebraic equation as,
 $[L][U]\{x\} = \{b\}$ (19)
- If we introduce an auxiliary vector $\{g\}$ through,
 $[U]\{x\} = \{g\}$ (20)
- Then from (19):
 $[L]\{g\} = \{b\}$ (21)
- First solve (21) for $\{g\}$ from,
 $g_1 = b_1/d_1$
- For $i = 2$ to n
 $g_i = (b_i - l_i g_{i-1})/d_i$
- Next solve (20) for the unknowns from,
 $x_n = g_n$ and
 $x_i = g_i - u_i x_{i+1}$ for $i = n-1$ to 1

Foundations of Computer Engineering, Semester IV, 2007

So, what we can see that if we use this as the starting value here, d_1 is equal to b_1 and u_1 is equal to c_1 by b_1 and then we can easily obtain l_2 from this. Since u_1 is already available, l_2 is available; so, this equation will give us d_2 . Once d_2 is known, c_2 is the given entry, we can use the third relation to obtain u_2 . So, we can see that this procedure can be generalized; we can actually see the generalization emerging from this relation itself. For example, we could write a_j is equal to l_j and then we could write b_j is equal to l_j times u_{j-1} plus d_j and c_j is equal to d_j times u_j .

So, as before we would be able to use this equation to obtain l_j and this equation will be able to use to obtain d_j and the third equation would give us u_j . So, that completes the splitting of the a matrix into l and u . Find the entries of l and u matrices and that is your original equation $a \cdot x$ equal to b .

So, suppose I define u times x as g vector, u matrix multiplied by the unknown, if I call that as g vector then this equation is nothing but l times g equal to b . So, in equation 21 what you are seeing? You know the entries of l matrix, you know the b vectors, you can solve for. Why you can solve for? Because it is a simple structure, so what you can do is you can start from the top.

You can take the first row and evaluate g_1 , then you come to the second row, from there you can calculate g_2 because this equation will only involve g_1 . So, we we could go

through this, that is what I am saying that g_1 would be equal to nothing but b_1 by d_1 and then once you have that, you can go to any other lines from 2 to n .

So, basically in solving 21, you go from top to bottom, so you have exhausted knowing this vector g . So, once you have the g vector, then you will have to solve 20. That is easy because that also has this sparse by diagonal structure.

(Refer Slide Time: 37:07)

$$\begin{bmatrix} d_1 & & & \\ & d_2 & & \\ & & \ddots & \\ & & & d_n \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_n \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 - d_1 u_1 \\ \vdots \\ b_n - d_1 u_1 - d_2 u_2 - \dots - d_{n-1} u_{n-1} \end{bmatrix}$$

So, what you can do is now you can see the last row has only one known 0 entry and so you can now go from bottom up sequence. So, that is what we have done, that if I look at the last one, then this multiplied by g_n should be equal to x_n .

So, since I know g_n , so that means I have found out x_n . So, if I figure the doubt, then if I go to the previous line, that would involve here g_n minus 1 plus u_n minus 1 into g_n should be equal to x_n minus 1. So, that is what is the general form of that equation.

So, we could write the doubt. So, since I have a starting value, I can go back once from bottom up and get the unknowns x . Now, so this is the way that we solve.

(Refer Slide Time: 37:52)

Implicit Method - Stability Analysis

- The difference equation for (15) is given as,

$$-\theta u_{i-1,j+1} + (1 + 2\theta) u_{i,j+1} - \theta u_{i+1,j+1} = (Pe - \theta) u_{i+1,j} + [1 - 2Pe + 2\theta] u_{i,j} + (Pe - \theta) u_{i-1,j} \quad (16)$$
 where $\theta = \lambda Pe$.
- Defining $u_{m,n} = \int U(k, t^n) e^{ikx_m} dk$ and a numerical amplification factor, $G = U(k, t^{n+1})/U(k, t^n)$, one can obtain the equation governing $G(kh, Pe, \theta)$ as,

$$-\theta G e^{-ikh} + (1 + 2\theta) G - \theta G e^{ikh} = (Pe - \theta) e^{ikh} + [1 - 2Pe + 2\theta] + (Pe - \theta) e^{-ikh}$$
- This can be further simplified to,

$$G(kh, \lambda, Pe) = \frac{[1 - 4Pe(1 - \lambda) \sin^2(kh/2)]}{[1 + 4Pe\lambda \sin^2(kh/2)]} \quad (22)$$

Foundations of Scientific Computing, Pearson-PECS-a-2027

Why would we take the trouble of going from an explicit method to implicit method, provided we do not get dividend?

So, let us try to find out what we gain and the best thing for us to really find out, what we are getting in terms of its stability property? Because we recall that the fact that we have to, apart from accuracy, will also have to ensure numerical stability. For example, in your assignment, you must have seen that if you take lesser number of point which is blows up; so, that is a sort of a defining sequence of numerical instability.

So, here also we do the same thing. What we do here is just write down the difference equation and we will follow the same language by defining this unknowns in terms of its special dependence in terms of Fourier Laplace transform. And we are looking at the m th node, so that is why, we are writing x of m and this is evaluated at n th time step.

So, we will do this and we will substitute it here. If I do that, well, actually this is a little clumsy work, I should have used some other subscripts because this i and this ι should not be confused, this e to the power $i k x$ is the square root of minus 1, I think I will go back and correct it and load the correct one, I will exchange this subscripts in these equations.

So, if I do that, what I am going to get? As we have done in the spectral analysis of the 1 d convection equation, I will get the same thing. Now, what you are seeing here that I

have a quantity that would be evaluated at $t_n + 1$ and the right hand side would be all evaluated at t_n .

So, if I divide the both sides by U_k of t_n , then this one will give me a G and because it is also shifted to the left by 1 grid point. So, that is why I will have this e to the power minus ikh and the diagonal term is, of course, we are looking at i th point. So, diagonal term will not have any such shift, whereas the last one is shifted to the right, that will give me a e to the power ikh here. And of course, you understand θ is retained as it is and this u_{j+1} and divide by u_j , that will give me G .

Right hand side is a clean expression. So, of course, what you can see here then... well if, if, if, if, if, if you look at it, I could club the first and third term on the left hand side together, so I have e to the power minus ikh plus e to the power plus ikh and then I have the quantity $1 + 2\theta$ into G , 2θ into G , that is on the left hand side and on the right hand side, we could take $P e^{-\theta}$ and that will give us e to the power ikh plus e to the power minus ikh , and the other quantity is $1 - 2P e + 2\theta$.

(Refer Slide Time: 41:15)

The chalkboard shows the following steps:

$$- \theta G \left[\frac{e^{-ikh} + e^{ikh}}{2 \cos kh} \right] + (1 + 2\theta) G = (P e - \theta) \left[\frac{e^{ikh} + e^{-ikh}}{2 \cos kh} \right] + (1 - 2P e + 2\theta) G$$

$$\left[-2\theta \cos kh + (1 + 2\theta) \right] G = 2(P e - \theta) \cos kh + (1 - 2P e + 2\theta) G$$

$$\left[1 + 2\theta \frac{(1 - \cos kh)}{2 \sin^2 kh} \right] G = 1 + 2(P e - \theta) [1 - \cos kh]$$

So, this is nothing but $2 \cos kh$, right, will you agree with me? Same thing here, this is $2 \cos kh$ and what we could do then? We could write there minus $2\theta \cos kh$ and there is a G here, there is a G here, so I will have $1 + 2\theta$ into G is on the left hand side, I will have $P e - \theta$ and there is a 2 here $\cos kh$ plus $1 - 2P e + 2\theta$.

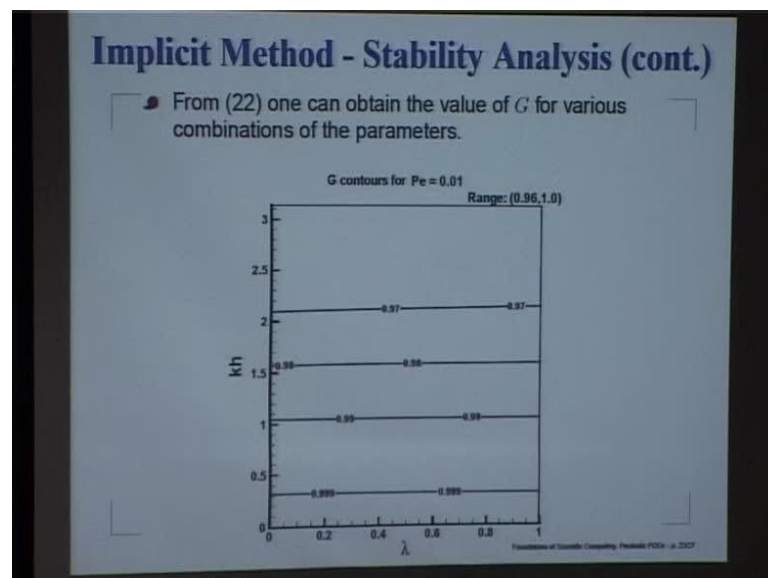
So, this I could write it as 1 minus, well, we can write it 1 plus 2 theta, then I will have 1 minus cos k h into G and on this side I could also get 1 here, and then I could write 2 times theta minus P e, this will give me 1 and from here I will get cos k h.

So, of course, this is 2 sign square k h by 2, so that is what you are seeing there. So, I am going to get, this is multiplied by 1 plus 4 theta sign square k h by 2.

So, that is what you are seeing here, down below because p into lambda is theta, right? So, that is what you see, the denominator and the numerator is also, similarly, is written in terms of this.

So, what you can notice is that this is also a bounded on the upper side by 1 because I can manipulate it and write it as 1 minus something. So, it is never going to be above plus 1, that is guaranteed for you. So, to guard against instability, all you have to worry about, that it should not fall below minus 1.

(Refer Slide Time: 44:57)



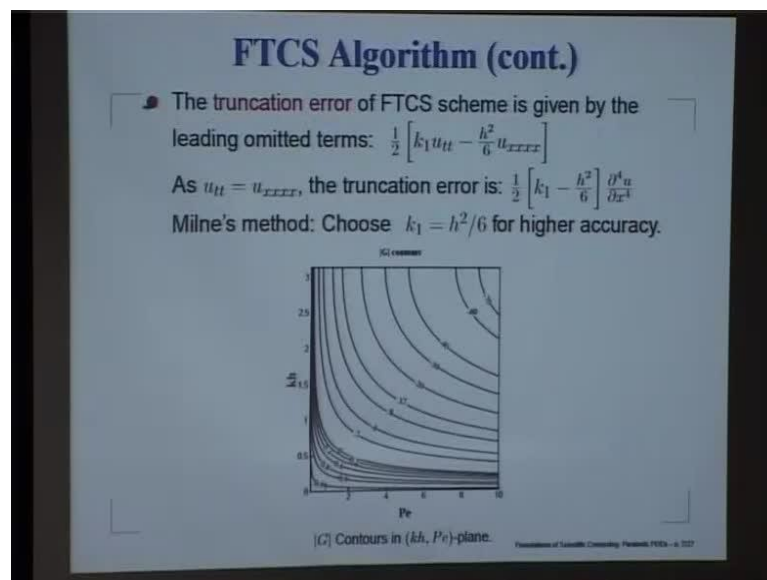
So, that is what we will have to do it and what you notice, that apart from the peclet number, now you also have lambda as a control for you or you can take the product of the two as theta as the parameter.

And now, what we can do is, we can obtain the value of G and for various combinations of the parameters, ideally I could have done it in the three dimensional space; and on one

side I will have lambda, another side I will have k h and on the third direction I will have the P e.

But to make the things little more understandable, I will show you by slices. So, take a very moderate value, low values of P e, that is, about 0.01 and then plot the G contours in k h lambda plane; so, this is how you get. The interesting bit is, what is written on the top right corner, here is a **range**, range of values of G in the whole space.

(Refer Slide Time: 46:05)

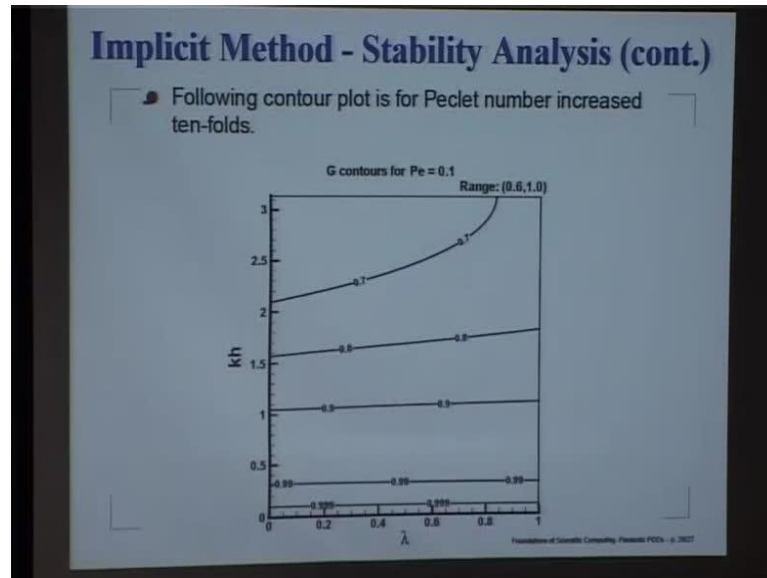


If you now go back, look at this range, it is ranging between 0.96 and 1 and if we look back to our FTCS algorithm, you know I did not write it, but you can see that it had gone from 1 to some value here, almost 0, I mean, there is a 0 line and on this side we have an instability line, which had gone all the way up to have the last contour, that we show here is minus 70.

So, you can see the **dramatic transformation**, dramatic transformation because this method was unstable in this part of the domain. So, even I am talking about Peclet number of 0.01, I would be looking at there, so **it is quite o.k.**, it is quite o.k. that it will remain stable, no problem there. However, look at the value of G, this will take all the way from 1 to virtually lets say 0.2, 0.15, that kind of value.

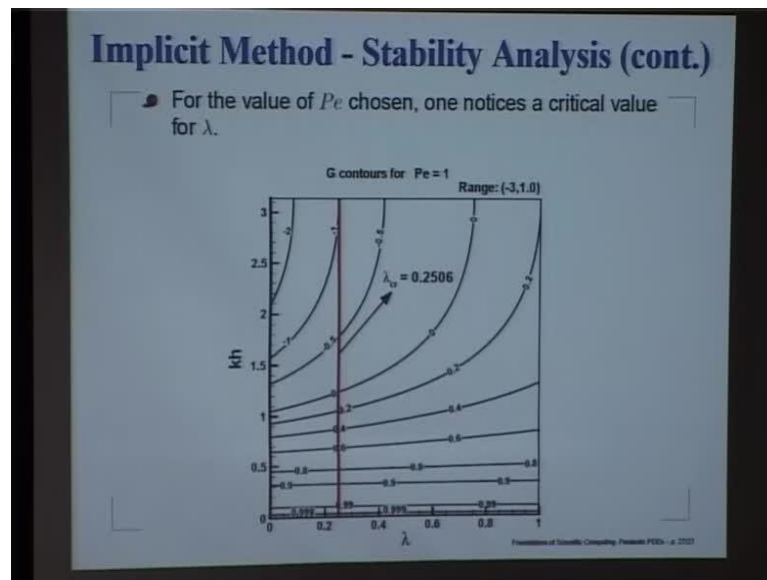
Whereas now by doing this implicit method, I could actually bracket that G between 0.96 and 1, a tremendous improvement, **right**? So, this is what we get by migrating from explicit method to implicit methods in terms of numerical stability property.

(Refer Slide Time: 47:32)



Now, if I increase the peclet number by a factor of ten, so that is what we have done. From 0.01, we have now gone to 0.1 and what do we find? The range is still not too bad, it is still quite, **it is between 0.6 and 1**, it is between 0.6 and 1 and you can realize that this is the story for virtually this is your explicit method, will correspond to what? Λ equal to 0.

(Refer Slide Time: 48:35)



So, you can see this is your explicit method and this is your fully implicit method and you can see that there is a significant improvement here that we do not go below 0.6, whereas in FTCS we could have gone all the way up to 0, **right**? And what happens if you take a much larger value of peclet number, that is of the order of 1, and then what you notice that finally you started seeing instability.

So, now the range is gone from minus 3 to plus 1, so what I have done here? **I have drawn that minus 1 line here**, we have drawn a minus 1 line here, so to the left of which this region is your unstable region.

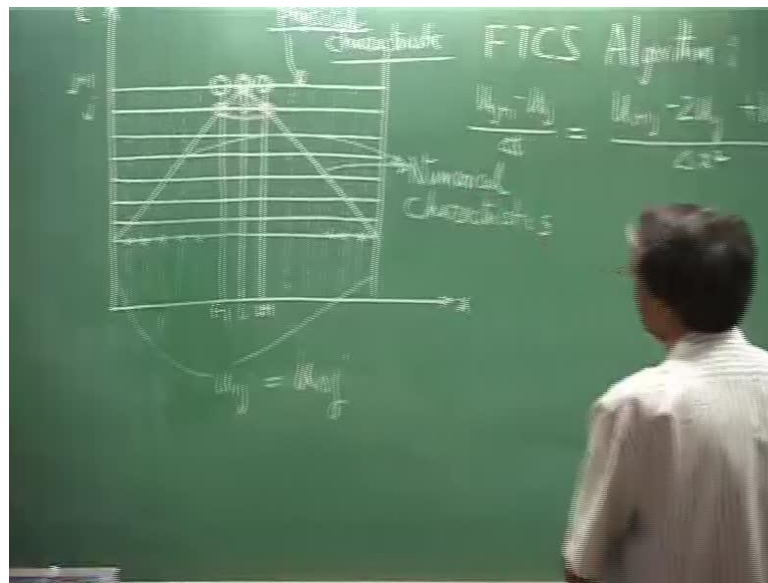
So, if you take explicit methods for value of lambda below this critical value, what will happen? You have to worry about some values of kh which are going to be unstable; so, that is something you have to worry about. However, if you take value of lambda above this critical value, then you have a fully stable method for all the kh range that you have resolved with your grade, so this critical value happens to be about 0.2506.

So, all that you are able to see now that you can get a very **very** spectacular improvement in your ability to take much larger time step. See, this values of peclet number are few orders of magnitude, larger then what you can do with the explicit method.

So, we are talking about improvement of the order of say 1000 times, 10000 times, that type of improvement in computing. So, this is what is all about the classical way of looking at parabolic equation and their computations.

There is only one thing that I have not talked about, which I would like to tell you is, this is more from practicality point of view and not specifically related to parabolic equation, but in many computing problems, what happens is we come across conditions where we state that the variables are periodic in nature, so what does it mean?

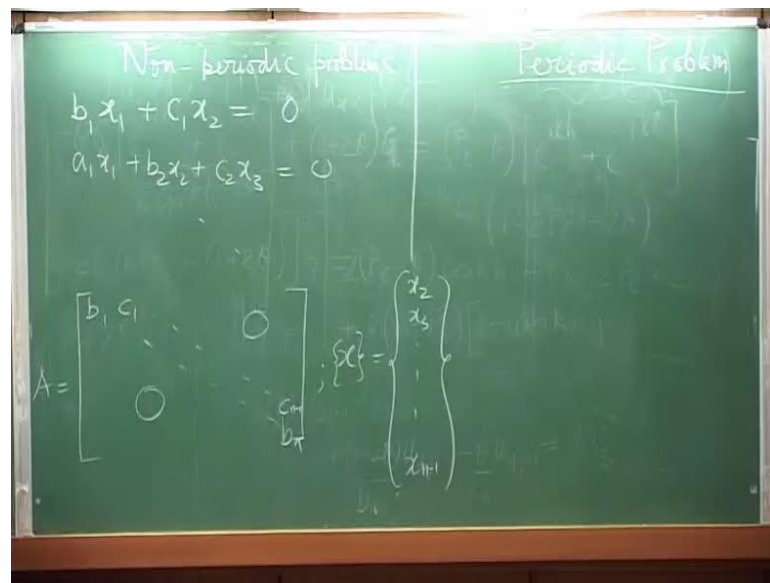
(Refer Slide Time: 51:27)



When I say **the very**, the variables are periodic in nature, what we are talking about is that this values are repeated; so, whatever the value I have here, so if I call that as u_1 that will be equal to say u_n .

So, the function is repeating itself, that is what you shall do with your Fourier series analysis. If a function is periodic, then you say look this is my **the** wavelength, so it is going to be periodic like this and then we should be doing this.

(Refer Slide Time: 52:13)

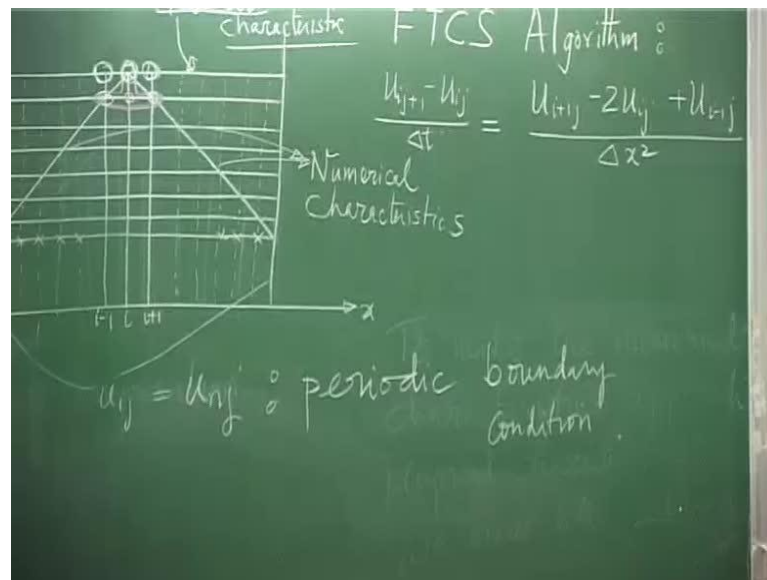


Now, all that I am going to ask you now, to help me in writing down the discrete equation. See, the earlier what we had, earlier we had written $b_1 x_1 + c_1 x_2 = 0$ was the first line, so this was when we did not have any periodicity involved. So, I will call them as non-periodic problem.

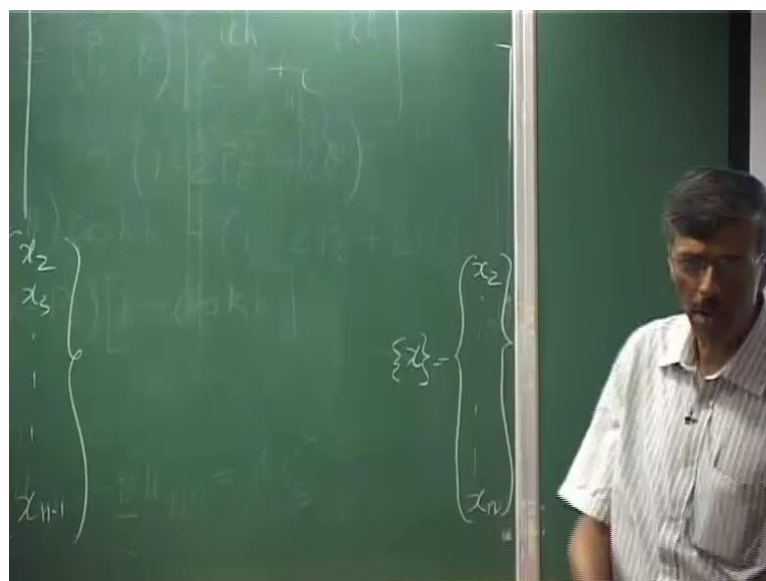
And the second line, add a 1×1 plus $b_2 x_2 + c_2 x_3$ equal to 0 and so on, so forth. That is why you had that A matrix that when we wrote, I wrote b_1 all the way up to b_n , then we had here c_1 all the way up to c_{n-1} , there is 0 0 and what was your x vector x vector and $x_2 \times 3$ all the way up to x_{n-1} , that is what we had.

Now, if I look at periodic problem what I am talking about? Now my unknowns, basically I do not have a deviational condition; instead, a boundary condition is the periodic boundary condition, that is what I wrote.

(Refer Slide Time: 54:02)



(Refer Slide Time: 54:27)



So, basically my unknowns will be what? If I have to write x vector, so I could start from x_2 all the way up to x_n . Now, you see the difference because why did not I write x_1 because x_1 equal to x_n . So, I am just simply noticing that here the number of unknowns are n minus 2, here it is n minus 1, so dimension is bigger.

(Refer Slide Time: 55:05)

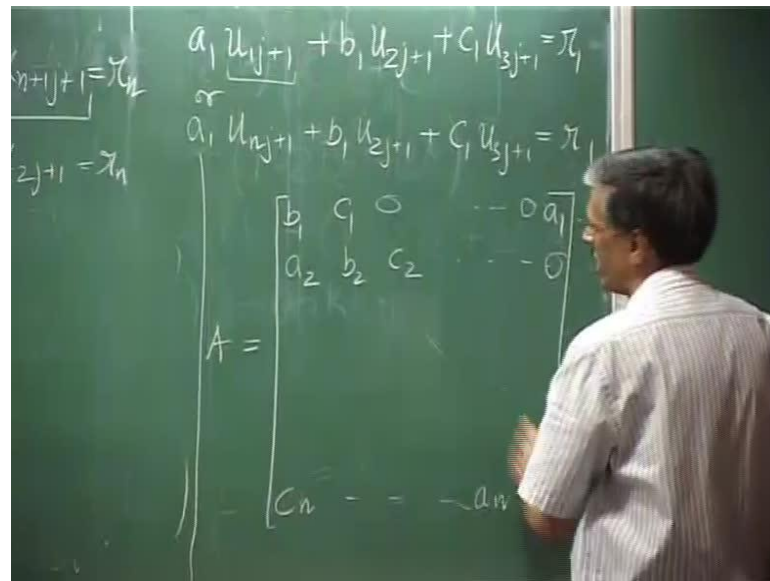
Non-periodic problems

$$b_1 x_1 + c_1 x_2 = r_1'$$
$$a_1 x_1 + b_2 x_2 + c_2 x_3 = r_2$$
$$A = \begin{bmatrix} b_1 & c_1 & 0 \\ a_1 & b_2 & c_2 \\ 0 & 0 & 0 \end{bmatrix}$$
$$\{x\} = \begin{bmatrix} x_2 \\ x_3 \end{bmatrix}$$

What about the a matrix? What will be the a matrix? Well I wrote something totally wrong, none of you protested, but let me write it as this r_1 , r_2 and so on, so forth, so that there is a right hand side, **right**? That is how we got that implicit.

So, now if I look at this, what did I do? I had a r_1 prime, what was r_1 prime? If you remember what I said? r_1 prime was whatever r_1 that I had, I also have a quantity called a_1 , if I look at the point. So, if I have this as i equal to 1 and I am now writing the equation for what? **I am, I am**, I am writing the equation from 2 onwards, that is how I have catalogued the unknowns.

(Refer Slide Time: 56:07)



So, if I write the equation for 2, then I have what here? So, **i**, for i equal to 2, let us write one of the equation that you will appreciate. You are going to get u_{1j+1} , that is an unknown, that is multiplied by a 1. Then I have b_{1j+1} and c_{1j+1} is equal to r_1 .

Now, what you are noticing that this quantity is not a derivational quantity, like, earlier we could put it on the other side, so what should I do now? Here I use the periodic condition that is given there, so I will write it as u_1 .

So, what has happened now to the A matrix? You can see that corresponding to the last entry I have a non-zero a_1 , **right?** So, a matrix if I have to write it like this. I will remove this, so since you have noted is now, I will expect that you remember what we are doing.

So, now, if I were to write down the A matrix, the diagonal entry still remains the same b_{1j+1} , then I have the super diagonal, what about the a_1 quantity? a_1 quantity is nothing but it will come here.

Do you see that there are stacks of 0s? And then **you have here**, then of course once we are in the other line there is no such problem. Same way, what will be my last equation? The last equation - if you allow me to erase this part, so we are not talking about non periodic part is easy and done - now, if I try to write the last equation, well, what would be that? That would be a_n , tell me, $n - 1j + 1$ plus b_n , well, r_n .

Now, this looks quite o.k. except the fact that what is this? Our unknowns are from 2 to n , so what is $n+1$ doing there, what is the meaning of $n+1$? It will be what? So, x_{n+1} would be some point here, some fictitious point here. Listen it, if I write it for this, this is involving all this three nodes, what about this point? This actually will be here because it is periodic.

So, that is what we are going to do that I am just simply going to write the same thing, but here I will write $c_{n+2} x_{n+1} + b_n x_n + a_n x_{n-1} = r_n$. So, if I look at the last equation, then what we have here is that b_n remains here and a_n remains here and c_n goes where? c_n comes here because that corresponds to x_{n+2} ; so, this is the structure.

So, what we have here is a periodic tridiagonal matrix.

So, in the next class I will just, tomorrow I will tell you how we handle periodic tridiagonal matrix. This is not exactly the replica of what we have done for non-periodic cases and this periodic problems are far too many for us to really understand and digest.

O.K.