

**Computational Fluid Dynamics Using Finite Volume Method**  
**Prof. Kameswararao Anupindi**  
**Department of Mechanical Engineering**  
**Indian Institute of Technology, Madras**

**Lecture - 15**  
**Finite Volume Method for Diffusion Equation: Linearization of source term,**  
**line-by-line TDMA**

(Refer Slide Time: 00:21)

Thomas algorithm (TDMA):

- 1) Calculate  $P_i = b_i/a_i$ ,  $Q_i = d_i/a_i$
- 2) Recurrence relations, calculate  
$$P_i = \frac{b_i}{a_i - c_i P_{i-1}}; \quad Q_i = \frac{d_i + c_i Q_{i-1}}{a_i - c_i P_{i-1}}$$
$$i = 2, 3, \dots, N-1, N$$
- 3)  $T_N = Q_N$
- 4)  $T_i = P_i T_{i+1} + Q_i$

Good morning. Let us get started. So, we are looking at the Thomas algorithm. or also known as the Tridiagonal Matrix Algorithm ok. Yesterday we have discussed how kind of this works, as a kind of special case of or a modification of the Gauss Seidel gaussian elimination process right. So, let us; let me kind of write the complete algorithm today. So, the first step is to calculate the coefficients  $P_i$  and  $Q_i$  right. So, the first one is  $P_1$ , what was  $P_1$ ?

Student: (Refer Time: 00:56).

$P_1 = \frac{b_1}{a_1}$  and  $Q_1$  is;  $Q_1 = \frac{d_1}{a_1}$  how much was  $Q_1$ ?  $d_1$  upon  $a_1$  is it? Can you help me write this? Was it  $d_1$  upon  $a_1$ ? Correct, ok. So, this is  $P_1$  and  $Q_1$ .

Then what do you do? Then you use the Recurrence relations and, calculate  $P_i$  as which is given by; how much was  $P_i$ ?  $P_i = \frac{b_i}{a_i - c_i P_{i-1}}$  is that correct? Ok? And then we have  $Q_i = \frac{d_i + c_i Q_{i-1}}{a_i - c_i P_{i-1}}$ . Is that correct? So, that is what we have.

So, essentially for  $i$  equals 2,3 all the way to  $N$  minus 1 to  $N$ . We are going to kind of use these recurrence relations and calculate what is  $P_i$  and  $Q_i$ , right. Ok? Alright. So, then I want to have  $P_i$  and  $Q_i$  what do we do? Back substitution.  $P_i$  and  $Q_i$  is known. Set what is  $T_N$  right. You start off what is  $T_N$  equals  $Q_N$  right. And what do we do next?

Student: (Refer Time: 02:26).

Use the  $T_i$  relations and calculate using back substitution, what is what is  $T_{N-1}$ ,  $T_{N-2}$  and so on to  $T_1$  right. So, essentially use then the relation  $T_i = P_i T_{i+1} + Q_i$  is it? Was it  $Q_i$ ? Right. Use this relation, and calculate for  $i$  equals  $N-1$ ,  $N-2$  and so on, all the way to 1 right or 2 whatever ok.

(Refer Slide Time: 02:54).

So, essentially you calculate all the temperatures. That is alright. You have obtained all the temperatures in the back substitution process ok. So, this is the algorithm for TDMA, right? Which you can kind of code and it will give you solution much faster compared to the gauss seidel right. Questions on this? Ok. So, would you be able to code this part? The algorithm is already there. You essentially need arrays for your  $a$   $b$   $c$   $d$  which you already have from the discretization and then you need arrays for  $P$  and  $Q$  right you would need arrays for  $P$  and  $Q$  and of course, temperature also would be there right.

So, if we have to write you have to write a kind of a function that takes in these arguments and calculates what is the temperature? ok. So, you need a small subroutine or a function that you would write, which will calculate this and then give you back the field temperature array, ok. That is what you would do. Fine.

Any questions? No? Alright then let us move on. So, we will look at; continue look at the solution of linear algebraic equations for the 2 dimensional and 3 dimensional situations ok. So, that is. So, this is again solution, a kind of linear algebraic equations in essentially 2D, of course, you can also extend this to; you can easily extend to 3D as well, ok. The same concept you can extended.

Now, as far as the solution of linear algebra equations in 1D is concerned, We have now learnt two methods; right. The first one was; what was the first method? Gauss Seidel right? This was, that was Gauss Seidel which we call it as a point by point method. Ok. That is, because you kind of go from one piece l to the next one and keep iterating. The other one we saw was the tridiagonal matrix algorithm ok. We looked at Tridiagonal matrix algorithm or the Thomas algorithm right? Which kind of gives you solution in like one shot essentially, ok. By solving this the algorithm that we just saw. Fine.

(Refer Slide Time: 05:44)

extend to 2D

1D: 1) Gauss-Seidel point-by-point

2) TDMA (Thomas algorithm)

$$a_p \phi_p = \sum a_{nb} \phi_{nb} + b$$


---

G-S:  $\phi_p = \left( \sum a_{nb} \phi_{nb}^* + b \right)$

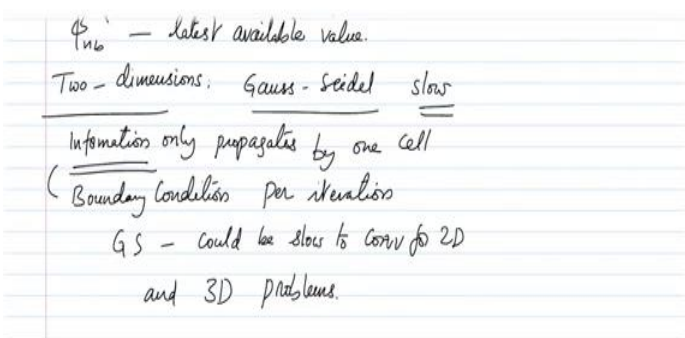
$\phi_{nb}^*$  — latest available value.  $a_p$

Now, if I kind of want to repeat a few things here. So, that we can use some terminology. So, in case of Gauss Seidel, let us say if you have  $a_p \phi_p = \sum a_{nb} \phi_{nb} + b$ , if this is our discretized linear equation that we have. Then in Gauss Seidel, what we did was, we kind of rearrange this equation for every cell and we wrote this as phi p which is the unknown at any cell ok. So, which is the unknown at any cell we calculated by writing it as a and b phi and b star plus b upon ap right.


Where I have now included a star, a superscript star, which denotes the latest available value ok; that means, if we are looking at a p cell, the west cell value would be the one that is just iterated, just updated and the e cell value would be the one from the previous iteration right? So, that is what we mean. And similarly, if you are let us say going from left to right and bottom to top, the south value would be the one which is just updated and the north value would be the one which would be updated ok. So, that is what we

mean by the star. The star is the latest value of  $\phi_n$  that is available in that cell, ok.  
Excuse me. Alright.

(Refer Slide Time: 07:10)



$\phi_n^*$  - latest available value.  
Two-dimensions: Gauss-Seidel slow  
Information only propagates by one cell  
(Boundary condition per iteration  
GS - could be slow to conv for 2D  
and 3D problems.



So, then this is the concept. Now, let us say how do I now extend it to two dimensions? Of course, in both 2D and 3D, you can have Gauss Seidel, as a piece, ok. You can still apply Gauss Seidel only thing is that, it will be kind of slow, because the information only propagates by one cell per iteration in Gauss Seidel. Right.

Because, you have one Sweep in which the information goes from one p cell to the e cell after the end of the iteration, ok. So, because information only propagates, ok., this information we mean that the boundary condition information. Right. Because we have specified some information. So, the boundary condition information that we have only propagates by one cell per iteration. As a result, Gauss Seidel could be slow to converge for 2D and 3D problems, ok.

So, what; So, what we kind of resort to is a is a method which is a combination of a direct method and this iterative method, that is the Gauss Seidel, ok. Now, of course, you are also welcome to develop a direct method for 2D and 3D problems. Only thing is that whatever the Thomas algorithm we have developed, that has to be kind of extended to a Penta diagonal matrix. Right. Instead of, let us say if you are looking at a 2D problem then, how does your discretization look like? Your discretization will give you a matrix that has five nonzero diagonals. Right. Nonzero coefficients on the five diagonals is what

would be there. So, one way is to of course, develop a solution algorithm that is similar to the Thomas algorithm, but for a Penta diagonal matrix, but that would require a lot more effort. So, as a result we will not go into that direction, rather we would kind of use a combination of combination of the TDMA and Gauss Seidel, ok.

(Refer Slide Time: 09:21)

Combination of TDMA and GS to solve  
for higher dimensional problems  
line-by-line method:

TDMA - y dir  
GS - x dir

So, we will use a combination of these two, to solve for 2D and 3D problems or higher dimension problems ok.

Fine. So, what we do is, we call this as; we give it a name, we call it as a line by line method ok? What we mean by line by line is basically, we choose one particular line and then solve TDMA along that line, thinking that that is a 1D problem ok? But with the coefficients taken from the 2D discretization, ok? And then we move on to the next line and so on, until we finish all the lines in the particular 2D domain ok?

So, now if I were to draw. So, I am drawing a, I am drawing lines that connect the p cells or e cells and west cells, all of them. And I am using different colours for different lines ok. So, essentially, we have; let me use one x ok. So, essentially ok. So, this is my x axis and this is my y axis. And then, we have some gridlines here ok. And then we have some gridlines in the vertical direction as well ok. So, we have some mesh here.

Now, these cell centroids that we have here, these are all the cell centroids, ok. This is we are looking at the cell centroids is, where these junctions are. Fine, ok; That means,

my actual cells are for example, if you are talking about a p cell, the p cell is somewhere here right and the e cell is somewhere here, this is the west cell. So, we have kind of connected all of these using these lines as the lines that connect the cell centroids. Fine. Now, we have two sets of lines; one is the black, black lines which are horizontal and the set of lines is vertical, which are the red lines.

Now, what we, what we do is we select; we kind of solve the TDMA along the red lines ok. So, essentially solve TDMA along y direction, ok; along the red lines and use Gauss Seidel along the x direction, fine. So, that is what we mean by a line by line method. We go from one line to another line.

(Refer Slide Time: 12:24)

The diagram shows a 4x4 grid of cells. Vertical lines are red and labeled 'TDMA - y dir'. Horizontal lines are black and labeled 'GS - x dir'. A coordinate system with x and y axes is shown. A legend indicates that 'X' marks are 'assumed to be known' and '•' marks are 'UNKNOWN'. Below the grid, the text reads: 'Traverse direction - TDMA dir - y dir' and 'Sweep direction - lines are visited x-dir'. At the bottom, a partial equation is visible:  $a_{i,j} \phi_{i,j} = a_{i,j-1} \phi_{i,j-1} + b_{i,j} + \phi_{i,j+1} +$ . In the bottom right corner, there is a small video inset of a man speaking.

And then I would use another kind of terminology here, which is, I use two terms; one is called Traverse direction, and the other one is called Sweep direction ok.

Now, what we mean by Traverse is, the direction in which the TDMA is applied ok. So, this is the TDMA direction. In this particular case, the y direction happens to be the Traverse direction and the Sweep direction is its, the direction which these lines are progressed, ok. Let us say, in this particular case we, we move from x min to x max, we go from 0 to x max. So, we are going sweeping in the positive x direction ok. So, that is essentially the direction in which your Gauss Seidel are the lines are visited ok. So, this is the direction which lines are visited, ok. This happens to be the x direction.

Now, the question is, then how do I solve TDMA along these lines? That is very simple, if we look at the discretized equation, what would a discretized equation look here in terms of  $ij$ , if we indicate  $i$  and  $j$  as the  $p$  cell? So, what I have is, I have a  $ij$ ,  $\phi_{ij}$  would be my unknown for the  $p$  cell. Right. This is like a  $\phi_p = \sum_n b_n \phi_n + b$ . Now, on the right hand side, what do we have in terms of  $i$  and  $j$  indices, that would be some  $a_{ij+1} \phi_{ij+1} + \text{some } b_{ij-1} \phi_{ij-1} + \text{what else do we have?}$  plus some  $C_i + 1_j \phi_i + 1_j + d_i + 1_j \phi_i + 1_j$ , I am sorry, should be minus 1.

So, this is  $i-1, j$  and this is  $i-1, j+1$  plus we have some what else some constant let us call it constant is  $e_{ij}$  ok; that is our  $d$ , our earlier  $d$ . So, we have essentially links to East, west, north, south and the and the source term right. So, we have all these things.

Now, we want to apply a TDMA along the  $y$  direction, right? So, if you want to apply TDMA along the  $y$  direction, like along this direction; that means, our the coefficients  $abcd$  or our what would be unknowns? If I want to apply along the TDMA along the  $y$  direction, what will be my unknowns?  $ij$ , would this be an unknown? Yes, this is an unknown ok? So, this is an unknown, what else would be an unknown?  $ij+1$  right? So, this would be an unknown and what about  $i, j-1$ ? Is an unknown as well.

So, essentially along the  $y$  direction, I have all the unknowns aligned. Now, I kind of pretend that all the values on the left hand side and the right hand side are known to me, ok. Those are the latest values, are the guest values, they are known to me ok; that means, I am not solving for them along this line. Along this line, when I am solving for TDMA everything else is known to me, right. Other than what is aligned along that line; that means, I can write  $\phi_{i+1, j}$  with a star, ok. The star means that it is the latest value that we know and  $\phi_{i-1, j}$  is also with a star.

Ok; that means, if I look at here; So, you are this is your  $\phi_{ij}$  right. So, essentially, these black lines, these black dots are where the unknowns are aligned and the crosses, are where the values are assumed to be known right? So, these are assumed to be known, for the purpose of this iteration and these are the values which are unknowns. Right. So, essentially now we kind of came back to a 1D situation from a 2D situation. Now, can you extend the same concept to 3D? right.? So, we have one line and then you have four neighbours, right.

So, you have two more terms; which are on the top and bottom. They will also be treated as knowns, right. They will also be treated as known values. Fine.

(Refer Slide Time: 16:43)

Traverse direction — TDMA in  $y$  dir  
 Sweep direction — lines are visited  $x$ -dir

$$a_{ij} \phi_{ij} = a_{ij+1} \phi_{ij+1} + b_{ij-1} \phi_{ij-1} + \{c_{i+1,j} \phi_{i+1,j}^* + d_{i-1,j} \phi_{i-1,j}^* + e_{ij}\}$$

$$\rightarrow a_{ij} \phi_{ij} = a_{ij+1} \phi_{ij+1} + b_{ij-1} \phi_{ij-1} + g_{ij}$$

Knowns

Then, I can rewrite this equation as  $a_{ij} \phi_{ij} = a_{ij+1} \phi_{ij+1} + b_{ij-1} \phi_{ij-1} + \{c_{i+1,j} \phi_{i+1,j}^* + d_{i-1,j} \phi_{i-1,j}^* + e_{ij}\}$  1 plus some  $g_{ij}$ , where  $g_{ij}$  consists of what?. So,  $g_{ij}$  is a known value, right.

Now, can we apply TDMA for this line? Right? I can apply TDMA along this line. And then solve for all the  $\phi$  values in one shot, right, ok. And then I move on to the next line. So, essentially, we solve it for; we solve it for; let us say this is the first unknown. So, we solved TDMA along this line with guessed values and then we move on to the next line.

So, when we move on to the next line, what do we do? We update the  $\phi$  values that we have just found for these points, right? We update them and then we move on here. Then, we calculate these values and then, we move on to this particular line, and so on. So, our Traverse direction is in the  $y$  direction and the Sweep direction is in the  $x$  direction. The direction which the lines are visited ok?

Now, you may have a question, like why should it be  $y$  direction? why should it not; why cannot have  $x$  and  $y$  interchanged? You can certainly do that. You can interchange  $x$  and  $y$  directions, this Sweep and they Traverse directions. That is completely possible. In



fact, we will also see a kind of intelligent way of doing these things, such that we kind of get convergence faster, ok. That is also possible. Now. ok.

So, that is, that is line by line TDMA. Now questions on this part; that means, once you have a if you have a TDMA solver, you can easily now extend it to solve 2D and 3D and, and this will be much faster ok. Will we, will see in a little while why it is faster than Gauss Seidel?

Student: (Refer Time: 18:45).

Yes right. Why do we say; So, the Gauss Seidel; the question is TDMA is quite visible, because along a y line, but; however, where is the Gauss Seidel coming into play here? So, the Gauss Seidel is coming into play, in terms of convergence. So, essentially, you Sweep along one y direction, right? And then, that is not the correct; that is not the final answer, right? Because we have started off with guess values on both sides. So, we are updating those line values and then we are moving onto the next line, right.

So, essentially imagine if the y direction was not there. If this was a; if this was just a 1D with only x directions, then you are sweeping in the x direction. Do you call that a Gauss Seidel? You are moving from point to point. Here only thing that you have collapsed the y line using a direct solver, right. So, all these values you got for all the cells in the y direction, we are just updating them and then moving on to the next set of lines, ok; and then you have to keep doing this process until you converge to phi values for the entire cells, all the range of cells, right?

Do you agree? If this was a direct solver in 2D you do not have to; you just calculate everything in one go. Right. Here, we have kind of a Gauss Seidel Sweep in the x direction and the TDMA is in the; along the y direction. Does it make it clear?

(Refer Slide Time: 20:14)

```
for i = 1 to N
{
   $\phi_i = \dots$ 
}
 $\phi_{[1:N]}$  Call TDMA
Check for conve
```

function TDMA ( N, a[N], b[N], c[N], d[N],  
submatrix T[N] )

So, essentially, let us say, what do I do? Let us say if I, if I write a kind of pseudocode for Gauss Seidel; how do you write a pseudocode? Essentially, you loop for, some for  $i$  equals 1 to  $i$  equals  $N$ , right? You kind of go through each of these things and you update what is  $\phi_i$ , from these values and then you would check for convergence right? that is what you would do. Right? This is what you do in a Gauss Seidel.

Now, in this also in 2D you are doing the same thing, but instead of calculating  $\phi_i$  directly from some expression you would call; what? You would make a call to TDMA function, which will give you, instead of  $\phi_i$  it will give you  $\phi_i$  for the entire range of  $n$  cells, right. It will give you for all the cells in one go. And then you would move onto the next line and so on; and then you would still check for convergence ok; So, that is kind of a line by line TDMA, you kind of check for convergence as well.

So, using the guess values, while calculating the TDMA, is where we call it as a Gauss Seidel, ok.; its kind of implicit in there, but not exactly, like Gauss Seidel ok. ok; So, yeah. So, the question is, if I have any iterative method, I would call it as a it is valid for anything right essentially the convergence checking and the iterations and why is it called Gauss Seidel? The question goes back to the difference between Jacobi and Gauss Seidel. Right. Essentially, we have if you are updating after you calculate you call it as Gauss Seidel or if you are not updating you want to keep those previous values then you

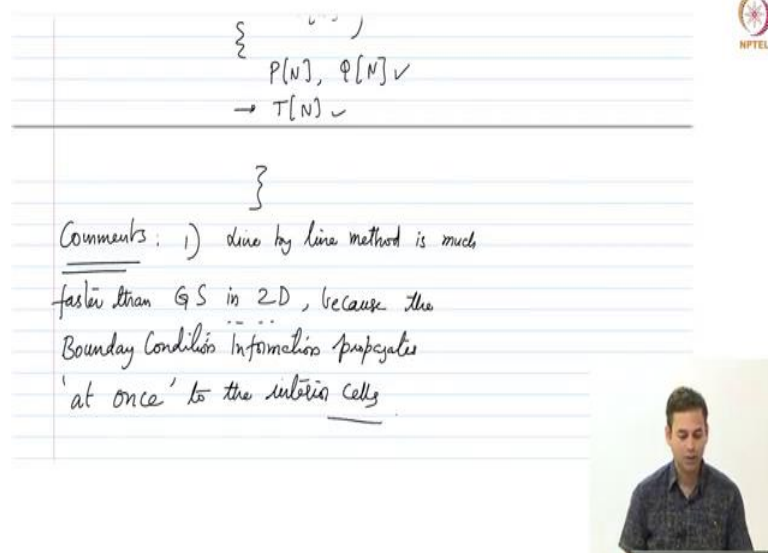
call it as a Jacobi method ok. I mean of course, you are welcome to name it anything, but its that is how its not how it is ok.

Other questions? You will be able to write this? Like essentially what I wanted to do is maybe in the next assignment, you would first develop a TDMA solver for 1D situation now that has to be generic enough such that . So, you have to write some kind of a function or a subroutine for solving TDMA ok. So, do not write it in the main program itself. So, you write a separate function, because you would be calling it many times and you would be using this one d thing in 2D and 3D situations later on ok.

So, TDMA, what all you want to pass it to? So, now, the TDMA should be generic enough such that you can either run it along y direction or along x direction. So, first thing you want to know is how many cells I want to kind of use right; and then you have to pass the coefficients that is a N, b N, c N right; and d N.

So, you have to kind of somehow pass all of these arguments. What else? Do you have to pass P i and Q i? No, you can calculate them inside from a b c d. What else? You want to also send what is the unknowns right, t N that is all.

(Refer Slide Time: 22:58)



The image shows a slide with handwritten notes on a lined background. At the top right, there is a small circular logo with the text 'NPTEL' below it. The notes are written in blue ink and consist of two main parts separated by a horizontal line. The first part shows a list of inputs:  $\{ P[N], Q[N] \}$  and  $\rightarrow T[N]$ . The second part is a comment: Comments: 1) line by line method is much faster than GS in 2-D, because the Boundary Condition information propagates 'at once' to the interior cells. In the bottom right corner, there is a small video inset showing a man in a dark shirt speaking.

So, essentially this function takes in all these arguments and then it will kind of calculate what is p of N and q of N can be calculated and finally, obtain values for t N using TDMA, right.

Now, if you have to write a two dimensional situation depending on the number of cells you have in x and y you would send those particular a b c d, right; your d would either depend on d would be either, it would be either these three together or it would be these two; was this one together depending on the direction right. If you are sweeping along traversing along the y direction then, these two would be there otherwise it will be the other ones right?

So, accordingly you would pass to the subroutine all these values and then you will be able to Traverse and Sweep in any directions that you wish; that you choose right. Now, this TDMA function that you would write would become part of a bigger function that would be kind of a, what kind of a Gauss Seidel? right. So, essentially this guy would be called here right; within your Gauss Seidel. Fine.

Is the Traverse and Sweep make sense? Ok. Those are; So, the i minus values are known when we are iterating, yes. So, either they happen to the boundary values or they happen to be the near boundary or whatever, right. Well, yeah I mean; so, yeah I think the question is again, why is it Gauss Seidel? I think you are not happy with the word Gauss Seidel. So, the thing is we just have values. You think of a 1D situation; you have some cells and you calculate the value at p cell and you move onto the next cell, right. That is what you do in a Gauss Seidel.

The same thing is what we are doing in the line by line TDMA, except that instead of calculating one value you are getting values for a row right, but the entire y values and this is what we are getting in one shot to TDMA, ok. That is why I call it as Gauss Seidel, but we can just say line by line TDMA ok? If you are not happy ok. Other questions? No? You will be able to; yeah.

Student: Solve one example.

Solve one example for TDMA, ok. That is a tall order, I cannot solve it in the class. So, that will be your homework, ok. If you have questions, I will explain with that right. So, maybe we will spend two three weeks solving TDMA in the class.

Student: (Refer Time: 25:23).

Sorry.

Student: (Refer Time: 25:26).

Just one iteration? TDMA, coming yes. So, he says essentially just make the matrix and then substitute. I do not think we have to do it essentially you have. So, I have the; essentially, it is this one right. Calculate  $b_1$   $d_1$  if you want I can set up a problem and solve it maybe sometime next week or so, ok. But at probably not required, you know. Because I want you to do the rest of the things from here, ok., fine. Would you be able to do? Decode these things? Right. So, we can do one example, but I think that is your learning process, ok.

Other questions? It's very straight forward right? Just calculate  $p_j$  and the algorithm is solid. So, you just have to go through all these steps. Yes. how  $\phi_{i+1, j}$  is an updated value? No,  $\phi_{i+1, j}$  is not an updated value. So, this is  $i+1, j$ . Here the question is, how is this an updated value? Right. this is not an updated value, this is a value from the previous iteration. That is all.

So, we are pretending that that is known from the latest value, because we are solving along the line right yeah. So, that so, the question is, how slowed Gauss Seidel would compare to TDMA? That depends on the problems, of the size of the problem, as such who will come to that in terms of; I do not have numbers as such, but it will be much slower no? Ok.

So, let us kind of have a discussion or comments. So, essentially your TDMA or we will say the line by line method, ok is much faster than Gauss Seidel in; that is a Gauss Seidel in 2D problems or whatever. Because the boundary condition information propagates in one go right. Or at once to the interior cells yes.

Student: (Refer Time: 28:03).

$a_j \phi_{ij}$  yeah. Coefficient of  $\phi_{ij}$  this one.

Student: (Refer Time: 28:07).

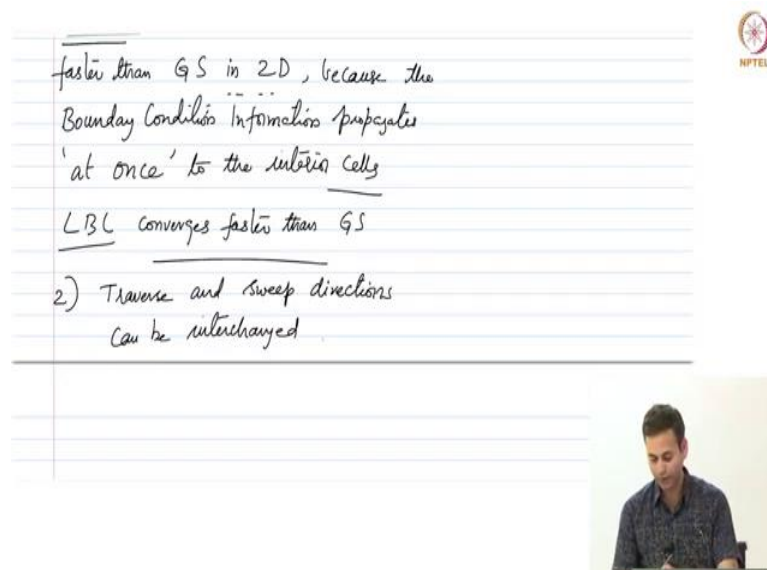
I am sorry, I should have written  $b$  yeah you are right. So, let me just change it to; So, the question is the coefficient should be different yeah, this is it should not be this should be something else. Let me call it as, central this is maybe let us call it as  $a_{ij}$  or something ok. This is certainly not the same something else ok, but that does not make a difference

anyway the coefficient power  $ij$  and  $ij + 1$  right  $i$  have this subscript anywhere there ok. So, the I should have started  $b c d$  instead of  $a b c$  ok. That is fine. Yeah of course, it is different. Its fine.

Other questions? No? ok. So, what I just wrote here is; I say, the line by line method is much faster compared to Gauss Seidel if you go for let us say 2D or 3D problems that is, because in the line by line method you actually using TDMA, which transmits the information of the boundaries into the all the interior cells in one go right. In at once you do not have iterations right? Whereas, in Gauss Seidel, the boundary condition information is updated to the cells by one cell per Sweep right? Because if you have done one sweep, then the information would have moved by oneself right from left cell to right cell and so on.

So, that is why the line by line method would converge faster than a pure Gauss Seidel ok?

(Refer Slide Time: 29:33)



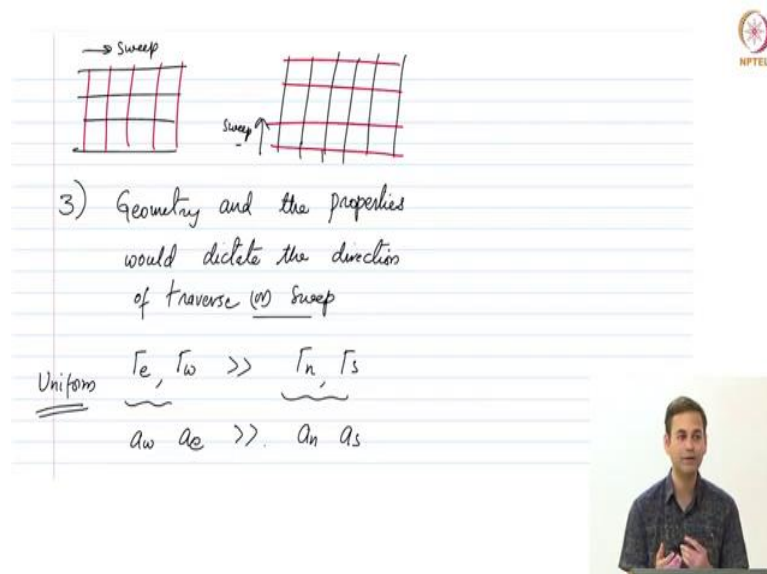
Handwritten notes on a lined paper. The text reads: "faster than GS in 2-D, because the Boundary Condition Information propagates 'at once' to the interior cells". Below this, it says "LBL converges faster than GS". A second point is listed: "2) Traverse and Sweep directions can be interchanged". In the top right corner of the paper is a small circular logo with the letters "NPTEL". In the bottom right corner, there is a small inset video showing a man in a blue shirt speaking.

So, LBL, which is the line by line method converges faster than Gauss Seidel, because of the information being sent across the boundary condition information being sent across in at once, ok. Now of course, the. The, The Traverse and Sweep directions can be interchanged, ok. For example, for, let us say for one iteration you could use a Traverse in the  $y$  direction and the next iteration you could use a Traverse in the  $x$  direction, ok.

You could kind of interchange the Sweep and Traverse directions to increase convergence.

Essentially, because the TDMA sends information of the boundaries very quickly to the interior, if you exchange the direction, then the information of the left and the right boundaries also would come into the interior much more quickly ok. So, as a result instead of just sweeping in the y direction let us say for 100 iterations; you could actually what you can do is you can just exchange the order of the Sweep and Traverse.

(Refer Slide Time: 30:51)



The slide contains handwritten notes on a grid background. At the top left, there are two 4x4 grids. The first grid has a horizontal arrow labeled 'Sweep' above it. The second grid has a vertical arrow labeled 'Sweep' to its left. In the top right corner, there is a small circular logo with 'NPTEL' written below it. Below the grids, the text reads: '3) Geometry and the properties would dictate the direction of traverse (or) sweep'. Underneath this, there are two mathematical expressions:  $\Gamma_e, \Gamma_w \gg \Gamma_n, \Gamma_s$  and  $a_w, a_e \gg a_n, a_s$ . The word 'Uniform' is written to the left of the first expression. At the bottom right of the slide, there is a small video inset showing a man speaking.

That means initially you can go with this is the Traverse and this is the Sweep. And the next iteration you could actually do this is the Sweep and your Traverse could be along x axis.

You could do this. So, essentially your Sweep is in this, your Sweep is along this direction here and your Sweep is along this direction here, ok. Along x and along y you could even alternate this, because the TDMA will bring in the boundary condition information to the interior cells in one go, ok. As a result, you would get convergence is much faster ok. I will try to; I think your next assignment would kind of involve these things, ok.

So, your TDMA function has to be generic enough, such that you can do all these patterns, ok. You could do, left to right and then top to bottom or you could do just a top

to bottom something like that, ok. or bottom to top all these things. Fine. So, that kind of patterns can be used to increase convergence, ok.

Now, sometimes the geometry and the properties, would kind of dictate the direction of sweep, direction of Traverse or Sweep, ok. So, what do we mean by geometry and the properties? Ok. Essentially, what we mean by this, is for example, let us consider you have gamma east, gamma west, and you have gamma north and gamma south, ok.

So, the diffusion coefficients; let us say the East and West diffusion coefficients are much larger than the North and South ok; that means, these are much larger than this; what would; and if we have a kind of a uniform mesh, ok. Or uniform cells. then what would be the coefficients a East and a West in comparison to a north and a south would be? These would be larger, right? These would be larger as a result, you would like to choose the Traverse in which direction? in the east west direction right? That is clear.

You want to choose Traverse in East West, why is it? So, because the coefficients are larger, because the boundary condition information will come in quickly and what else? Because a north and a south are so small, the errors that you introduce by guessing the values would be insignificant compared to what you would have from a East and a West right. So, as a result you would kind of go along Traverse along the x direction right. Because the guess values you would use in the north and south directions.

(Refer Slide Time: 33:55)

$\phi^*$  along n and s would have insignificant effect on  $\phi_p$  Traverse — x-dir  
 along the direction of larger coefficients

Grid diagram showing nodes  $w$ ,  $p$ ,  $e$  and arrows indicating the direction of the traverse.





So, the phi along phi star along, n and s would have insignificant effect on ap or on phi p. Right. So, as a result Traverse should be along x direction.

So, or in general Traverse should be along the direction which you have larger coefficients ok. So, it should be along the direction of larger coefficients ok; because the small coefficients would not change it much in terms of the solution convergence. Yes, ok; the question is, is it rule for fast convergence? Or does it make the matrix better right? This is essentially only for faster convergence because as you can see the a p. Of course, it would also depend on the a north and a south coefficients, but I do not think the diagonal dominance would increase, because you have chosen it this way right. Because you would have a source term essentially it is far faster convergence. Fine.

Other questions? No? Ok. Let us then; what about the property? So, properties is done what about the geometry? We said two comments right. One is geometry, other one is properties. These would kind of, indicate that we should choose that the Traverse in a particular direction. Coming to the geometry, Let us say we have a mesh that is very, with a large aspect ratio ok. So, essentially I have very thin cells, which is which is quite possible, in several applications ok. So, we have a mesh that is like this, which is really thin, ok.

It I just made it non uniform because I could not draw them much better. So, this is what we have; and the cells are like this ok. So, these are the cells ok. So, this is my P cell this is east this is north this is west and this is south ok. So, we have delta x which is much larger than the delta y and we also have the del x e and del x w and we also have the del y north and del y south ok.

So, you can see the delta x is greater than delta y somewhat greater three four times and delta xe delta x w also three four times greater than delta y north and delta y south ok.

(Refer Slide Time: 36:57)

along the direction of larger coefficients

Geometry:

$\Gamma$  is the same everywhere

$$a_e = \frac{\Gamma_e \Delta y}{\delta x_e} \quad \text{small}$$

$$a_n = \frac{\Gamma_n \Delta x}{\delta y_n} \quad \text{larger}$$

So, then how do the coefficients look like? If I look at a east and a north, ok. One for each direction, a east would be what gamma east. Let us say the gamma is the same; is the same everywhere. Then what would be a east coefficient?  $a_e = \frac{\Gamma_e \Delta y}{\delta x_e}$  and what would be a north?  $a_n = \frac{\Gamma_n \Delta x}{\delta y_n}$

So, which is now greater? So, which is now greater and in what direction would he choose the traverse?  $a_e$  is it greater.

Student: (Refer Time: 37:35).

$a_e$  smaller east west is smaller, because delta y is small. This is, this is small and this is delta x is large, in comparison to here, which is larger and this is smaller right. As a result, the north coefficients would come out to be larger than the east and west right? So, in which direction would you choose the Traverse? In the y direction? Right? Essentially that is going to; So, your Traverse would be in along this direction, this is your Traverse direction and you would Sweep would be in the in the x direction, ok. Fine.

So, essentially you choose it along the larger coefficients. Now, these are all only kind of tricks for little bit faster convergence, that is all. Now, these would help especially, if we have a big 3 dimensional problem or a big 2 dimensional problem, right. And if we have unsteady as well, then you have to kind of solve all this for every time step, ok. In that

situations, cumulatively you will have a much quicker solution obtained, if you make use of these tricks rather than blindly just running it with a the Gauss Seidel you know, something like that ok.

Student: (Refer Time: 38:41).

So, the question is, if you use TDMA along the more number of cells, which is the y direction you may run out of RAM, but yeah I mean if you have some technical problems then you have to accordingly change it. Fine. Here we are assume that we have enough RAM to run.

Other questions? Ok alright then, we looked at the comments, then let us make a statement about few more things, essentially we looked the geometry and the properties, ok. Both of them. We talked about the Traverse direction based on the coefficients.

(Refer Slide Time: 39:25)

The slide contains handwritten notes and a diagram. At the top, two coefficients are compared:  $a_e = \frac{\Gamma_e \Delta y}{\Delta x}$  (small) and  $a_n = \frac{\Gamma_n \Delta x}{\Delta y}$  (large). Below this is a diagram of a rectangular domain. The left boundary is labeled 'BCs' with a temperature  $T=T_2$ . The top boundary is labeled  $T=T_1$ . The bottom boundary is labeled  $T=T_3$ . The right boundary is labeled 'Adiabatic'. A vertical red line with an upward arrow is labeled 'T'. Horizontal arrows pointing right are labeled 'Sweep' and 'flow'. The text 'Sweep direction' is written above the diagram. Below the diagram, it says 'Sweep should bring (Known values)' and 'Correction - Sweep is the same dir as Conv.'. In the bottom right corner, there is a small video inset of a man speaking.

Now, similarly we can also have dependency on the Sweep direction, ok? The Sweep direction also could be dictated by the geometry that we have.

Now, how do we say this? For example, let us say I have coming to the boundary conditions; let us say we have a domain, like this rectangular domain on which we have some isothermal boundary conditions on the three faces and then on one of the faces on this side we have an adiabatic boundary condition, ok. This is these are the boundary conditions that we have. In this, in this scenario how do we choose the Sweep direction?

Usually, you start off Sweep from the direction where you have a known boundary condition, like a Dirichlet boundary condition ok. So, the Sweep direction should be from left to right. Let us say, if I am using my Traverse in this direction this is my Traverse, then my Sweep could be from left to right. Because as I am sweeping I am starting off with a known temperature and that would be brought in inside ok, but if I start from right to left then, because it is adiabatic I do not have a temperature right.

So, the information of the isothermal boundary condition or something like would not be brought in very quickly right. Because this itself is not a fixed value ok, as a result this if I Sweep in the other direction from right to left would be kind of slower than a Sweep in the from left to right ok.

So, now, these are all kind of observational things. So, if you write a code which you would in the assignment and then you can kind of play with them and then add it up with this insight that we are getting here.

So essentially, you should always start the Sweep kind of should, should bring known values ok. So, essentially that is in the direction of the known values ok. That is what we would do. Now, for example, let us say if you have convection. Let us say there is a flow that is going in this particular direction. Let us say there is a flow, then how would you sweep? Would you Sweep in the direction of the flow? Or would you Sweep against the direction of the flow?

Let us say flow is happening from left to right. What would be your Sweep direction? Left to right. You should go with the flow direction, because then you have information coming from the flow right? So, you should always go in the direction of convection for your sweeps rather than against ok? If you go against, will the problem converge? It will still converge. You may be doing some more iterations. That is all, ok.

So, convection then, then Sweep in the same direction ok. as convection fine. So, these are essentially some tricks by which you can choose the direction of the Sweep and the Traverse, fine? which will kind of help you get a faster convergence, then if you are chosen it without these things, ok. Questions?

No questions? Ok. Yeah number of cells yes. Do the number of cells effect the direction of sweep, Is it? So, a question is ok. If you have more number of cells, should I choose it

that way? That is possible, but usually you would not have; I mean depends on the problem you are solving ok? Ideally, because this TDMA is only a 1D problem. So, it would not make a lot of difference if you have hundred cells or let us say five hundred cells or so. It would still come out very quickly right? So, there are lot of things which we can affect here, but these are some broad ones ok. Suddenly number of cells and the aspect ratio all these things will definitely affect how fast it can kind of converge ok.

Other questions? So, the takeaway you have to write your subroutine such that it is generic enough and you can play with it ok. Sweep and Traverse you can kind of alternate or you can kind of call it in x direction, as well as y direction and things like that. Fine? No other questions? ok.

(Refer Slide Time: 44:01)

Source term linearization:

$$S(T) = f(T, T^2, T^3, \dots)$$

Non-linear Source term.

linearize it.  $S(T) \approx S_c + S_p T_p$

linear model

- 1) The nominally linear framework only allows for linear models to be there
- 2) linearization is better than just a constant.

Then, let us move on to one more thing. That is basically Source term linearization ok. Fine. So, we have always said, if there is a source term S of, let us say Temperature or something, S of T, which is a non-linear function of temperature ok.  $S(T) = f(T, T^2, T^3, \dots)$  S of T is a non-linear function. So, it depend on some constants etcetera. So, we have a non-linear source term. Right. Then we always said we have to kind of linearize it, and we said S of T would be linearized as some constant S c plus S p times T p right.

So, essentially we have a linear model, for this non-linear source term right. So, we have a linear model.

Then the question is, why do we have to linearize? And if we have to linearize, is there a good way of linearizing the source term? Right? Or can I linearize it in infinitely many ways? The thing is we have to linearize, because the ultimate system we would get is a linear system right. The entire framework is essentially, you would get  $ax$  equal to  $b$  right? So, you cannot have a non-linear thing with in there right.

Your  $x$  is always linear. So, essentially, the nominally linear framework only allows for linear models to be there, ok? That is why we kind of; That is why we must linearize the source terms right? And the second thing is, linearizing it is better than just having a constant in there, right. You could also say, I would use a constant for the source term, but linearizing is a better model compared to just having a constant source term ok. So, linearization is better than just a constant, ok? So, these are the two main motivations to linearize the source term and use it in the fashion that we have discussed. Ok.

(Refer Slide Time: 46:59)

2) linearization is better than just a constant.

General way to linearize: Taylor Series expansion

linear model:  $S(T) \approx (S_c + S_p T_p)$

{ good representation of  $S(T)$

$(S_p \leq 0)$  Condition.

$S(T) = S^* + \left(\frac{\partial S}{\partial T}\right)^* (T - T^*)$

$S^*$  - expanding about  $*$  values

Now, so, the general one of the a general way to linearize any source term is to use Taylor series expansion. Now, of course, the Linear model that we get should be a you know,  $S(T) \approx (S_c + S_p T_p)$  This should be a good representation of the original non-linear function right it should be somewhat a good representation of the original non-linear function and it should also satisfy the  $S_p \leq 0$  condition right. This we have kind of stated. So, if these two are there, then the linearization model we would use would be kind of a good model ok.

So, a general way to you do is using Taylor series expansion for the source term; that means, if I have some  $S(T) = S^* + \left(\frac{\partial S}{\partial T}\right)^* (T - T)^*$  . So, these star all evaluated. So, essentially, I have some S star, I am expanding about S star, expanding about star values, ok. What are star values? Known values, ok. We are expanding about the known values and then obtaining a model for our non-linear source term, ok.