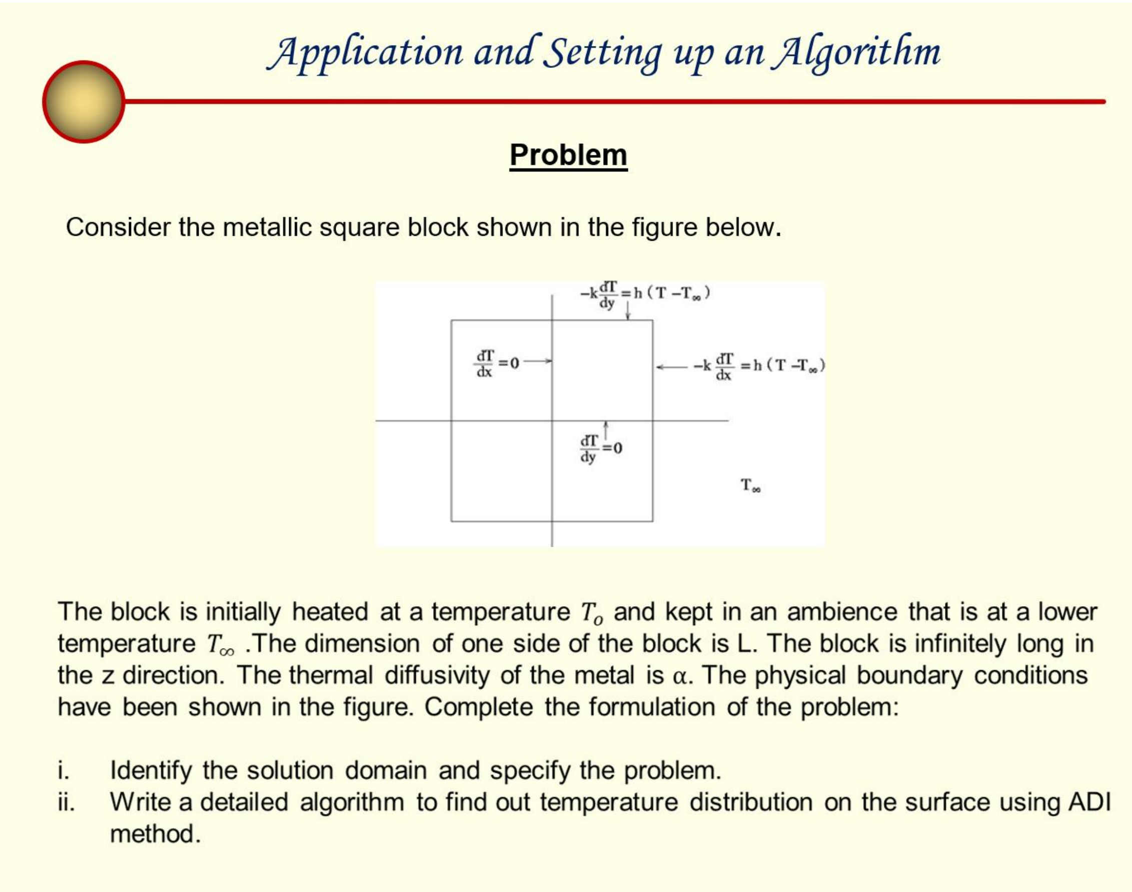
Computational Fluid Dynamics and Heat Transfer
Prof. Gautam Biswas
Department of Mechanical Engineering
Indian Institute of Technology, Kanpur

Lecture – 08
Applications of Our Knowledge to a Problem of Practical Interest and Setting up an Algorithm

Good afternoon, everybody, today we will do something little different. We will work on application of our knowledge to a problem of practical interest and setting up an algorithm or computer solution. How to set up the algorithm; we will experience that through today's lecture. And we will take up a problem which often occurs in metallurgical industry.

(Refer Slide Time: 01:02)



So, problem statement is the following, consider a metallic square block shown in the figure, and this square block is initially heated at a temperature T_o which is much higher than the ambient or local ambient temperature. The dimension of any side of this block is L. The block is infinitely long very long block in z direction.

So, we are looking at x and y direction, z direction infinitely long means, z direction dimension is much higher than x and y direction dimensions. The thermal diffusivity of the metal is α. The physical boundary conditions have been explained. See all around, it is being cooled by the local fluid and; obviously, this is $-k\frac{\partial T}{\partial x} = h(T - T_\infty)$, on this surface is convective boundary condition, which is basically in normal condition you can also say or mixed condition.

We have two tasks, one is identifying the solution domain and specify the problem and second is write a detailed algorithm to find out temperature distribution on the surface, surface of interest, we can see the surface using ADI (alternate direction implicit) method. So, first is setting up a problem that is quite simple.

(Refer Slide Time: 03:21)

Application and Setting up an Algorithm

Solution:

i. Considering the vertical symmetry and horizontal symmetry, the top right hand side quadrant can be considered as the solution domain.
The governing equation is:

$$\frac{\partial T}{\partial t} = \alpha\left(\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2}\right)$$

T is the temperature, α is the thermal diffusivity, k is the thermal conductivity and h is the convective heat transfer coefficient. The initial temperature of the block is T_0 .
It is a cooling problem. T_∞ is the ambient temperature and $T_0 > T_\infty$.

We can draw since all around the you know confining surfaces same boundary condition. So, there is a vertical symmetry and there is horizontal symmetry also. So, if we use these two symmetries; vertical symmetry and horizontal symmetry then we can make use of $\frac{\partial T}{\partial x} = 0$ on this surface and $\frac{\partial T}{\partial y} = 0$ on this surface (Refer Slide Time: 01:02), which in our computing language we can call symmetric boundary conditions.

So, considering the vertical symmetry and horizontal symmetry, the top right hand side quadrant can be considered as a solution domain. As such, any quadrant can be taken as a solution domain, we are taking top right-hand quadrant as a solution domain. And the governing equation is given by:

$$\frac{\partial T}{\partial t} = \alpha\left(\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2}\right)$$

Where, α is the thermal diffusivity, k is the thermal conductivity, h is a convective heat transfer coefficient.

We have already this geometry we have seen, and the initial temperature the block is T_o which is we have said higher than ambient temperature, hence it is a cooling problem and T_∞ is the ambient temperature, $T_o > T_\infty$. That is a physical interpretation of the problem.

(Refer Slide Time: 05:22)

Application and Setting up an Algorithm

The Part (ii): The ADI discretization of the governing equation :

Step 1

$$\frac{T_{i,j}^{n+\frac{1}{2}} - T_{i,j}^{n}}{(\Delta t/2)} = \alpha\left[\frac{T_{i+1,j}^{n+\frac{1}{2}} - 2T_{i,j}^{n+\frac{1}{2}} + T_{i-1,j}^{n+\frac{1}{2}}}{\Delta x^2}\right] + \alpha\left[\frac{T_{i,j+1}^{n} - 2T_{i,j}^{n} + T_{i,j-1}^{n}}{\Delta y^2}\right]$$

Step 2

$$\frac{T_{i,j}^{n+1} - T_{i,j}^{n+\frac{1}{2}}}{(\Delta t/2)} = \alpha\left[\frac{T_{i+1,j}^{n+\frac{1}{2}} - 2T_{i,j}^{n+\frac{1}{2}} + T_{i-1,j}^{n+\frac{1}{2}}}{\Delta x^2}\right] + \alpha\left[\frac{T_{i,j+1}^{n+1} - 2T_{i,j}^{n+1} + T_{i,j-1}^{n+1}}{\Delta y^2}\right]$$

Arranging both the equations in the form where the unknowns are on the left hand side and knowns on the right hand side.

Now, we can see that character wise this is basically 2-D unsteady heat conduction problem. So, we have already experienced that in our course, we have done some inner exercise how to solve this problem. We will apply ADI method; that means alternate direction implicit.

So, and instead of going from one level say a n^{th} level to $(n+1)^{th}$ level through one time step Δt, we will divide the time step $\Delta t/2$. And through $\Delta t/2$ first we will progress $\left(n+\frac{1}{2}\right)^{th}$ level from a n^{th} level. And then from $\left(n+\frac{1}{2}\right)^{th}$ level again to we will go to $(n+1)^{th}$ level.

So, for when we go from n to $\left(n+\frac{1}{2}\right)^{th}$ level, in the right-hand side, the x direction derivatives, we express in terms of finite difference quotient and this is simple central difference, but at the time level we are writing $\left(n+\frac{1}{2}\right)^{th}$. And in y direction also you are applying simple central difference, but we are retaining the temperatures at n^{th} level.

So, from here it is possible to calculate the temperatures at $\left(n+\frac{1}{2}\right)^{th}$ level through implicit formulation. Having known $\left(n+\frac{1}{2}\right)^{th}$ level temperature we come to step 2, then x direction we retain again you know terms corresponding to $\left(n+\frac{1}{2}\right)^{th}$ level, and here the temperature is discretized $\frac{\partial^2 T}{\partial x^2}$ using central difference. But, in y direction we write the variable set $(n+1)^{th}$ level.

And; obviously, the temporal derivative is at (i,j) point. So, temporal derivative will be $\left(T_{i,j}^{n+1}-T_{i,j}^{n+\frac{1}{2}}\right)/(\Delta t/2)$ and this step we are going from $T_{i,j}^{n+\frac{1}{2}}$to $T_{i,j}^{n+1}$ level and time step is $\Delta t/2$. This is a usual thing we have learned. So, in two step we proceed first from n to $\left(n+\frac{1}{2}\right)^{th}$, then $\left(n+\frac{1}{2}\right)^{th}$ to $(n+1)^{th}$.

Arranging both equations in the form of so in the form, where unknowns are on the left-hand side and known are on the right-hand side. So, for step one all the equations, all the terms that are expressed at set $\left(n+\frac{1}{2}\right)^{th}$level will be taken on the left-hand side these (first term of right-hand side of equation step 1) are not known. And terms which are at n^{th} level we will be kept at right hand side this unknown. Having done that, when you go for step two all the terms that are written at set $(n+1)^{th}$ level will come to the left-hand side and term at set $\left(n+\frac{1}{2}\right)^{th}$ level we will be retained on the right-hand side.

(Refer Slide Time: 09:16)

Application and Setting up an Algorithm

Equation from step 1

$$-\frac{r_x}{2}T_{i-1,j}^{n+\frac{1}{2}} + (1+r_x)T_{i,j}^{n+\frac{1}{2}} - \frac{r_x}{2}T_{i+1,j}^{n+\frac{1}{2}} = T_{i,j}^{n} + \frac{r_y}{2}\left(T_{i,j+1}^{n} - 2T_{i,j}^{n} + T_{i,j-1}^{n}\right) \quad (1)$$

Equation from step 2

$$-\frac{r_y}{2}T_{i,j-1}^{n+1} + (1+r_y)T_{i,j}^{n+1} - \frac{r_y}{2}T_{i,j+1}^{n+1} = T_{i,j}^{n+\frac{1}{2}} + \frac{r_x}{2}\left(T_{i+1,j}^{n+\frac{1}{2}} - 2T_{i,j}^{n+\frac{1}{2}} + T_{i-1,j}^{n+\frac{1}{2}}\right) \quad (2)$$

Exactly, we do that. And here while doing so, this $\alpha\frac{\Delta t}{\Delta x^2}$ we call it r_x and $\alpha\frac{\Delta t}{\Delta y^2}$ we call it r_y. So, we are writing

$$-\frac{r_x}{2}T_{i-1,j}^{n+\frac{1}{2}} + (1+r_x)T_{i,j}^{n+\frac{1}{2}} - \frac{r_x}{2}T_{i+1,j}^{n+\frac{1}{2}} = T_{i,j}^{n} + \frac{r_y}{2}\left(T_{i,j+1}^{n} - 2T_{i,j}^{n} + T_{i,j-1}^{n}\right) \quad (1)$$

here you can see right hand side all the terms are retained at n^{th} level and all the terms are known.

Similarly, from equation 2 again,

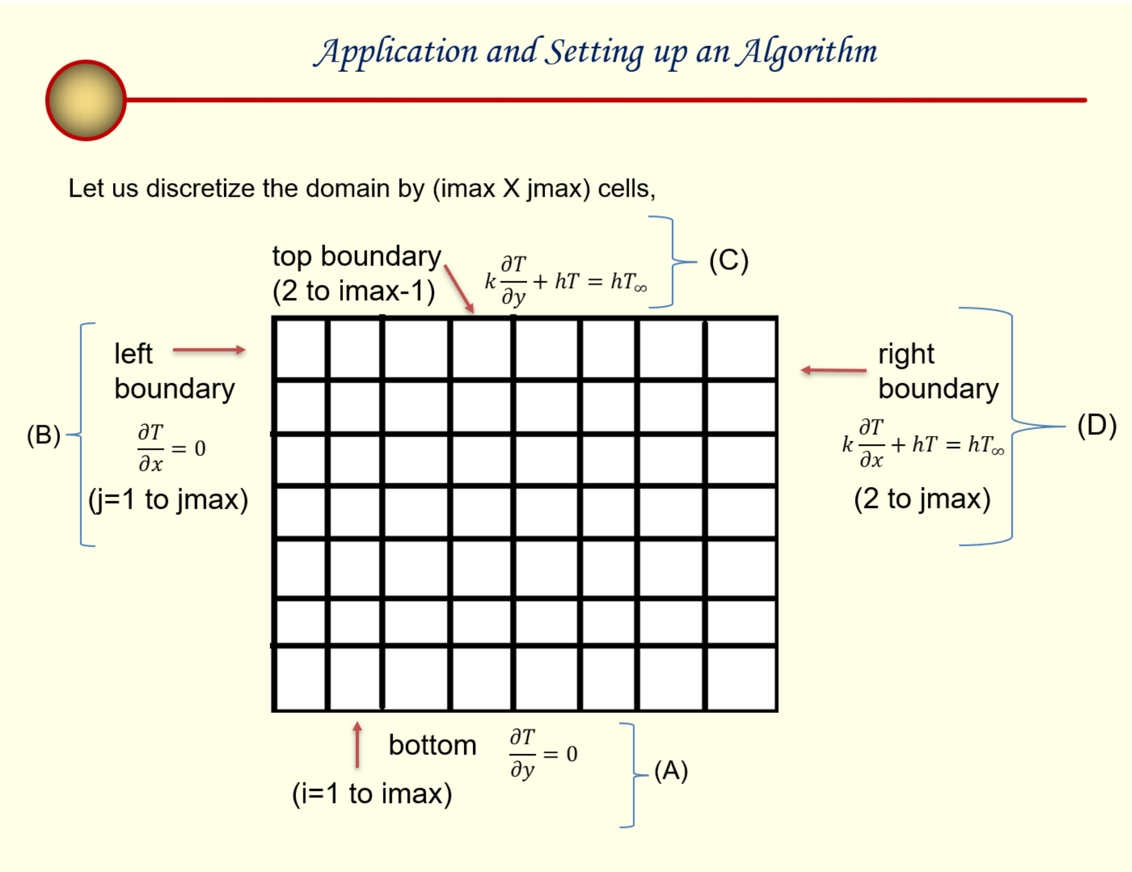
$$-\frac{r_y}{2}T_{i,j-1}^{n+1} + (1+r_y)T_{i,j}^{n+1} - \frac{r_y}{2}T_{i,j+1}^{n+1} = T_{i,j}^{n+\frac{1}{2}} + \frac{r_x}{2}\left(T_{i+1,j}^{n+\frac{1}{2}} - 2T_{i,j}^{n+\frac{1}{2}} + T_{i-1,j}^{n+\frac{1}{2}}\right) \quad (2)$$

The right-hand side all the values are $\left(n+\frac{1}{2}\right)^{th}$ level and since all the $\left(n+\frac{1}{2}\right)^{th}$ level values are known to us. So, these are the equations; equation 1 and equation 2. And we can see that this is implicit formulation.

Here, if we can vary i from the first point to last point and j from first point to last point, we will get some matrix and we will be able to solve that and hopefully you know formulation should be such we will get tridiagonal matrix for step one. And also, we will get tridiagonal matrix from step two.

That is the whole purpose of doing ADI, because inverting tridiagonal matrix is simple and solution can be obtained very easily. We have already seen how it functions, and very simple algorithm only the diagonal elements are to be changed, and right-hand side vector resultant vector has to be changed. And then we will get the solution very easily through back substitution.

(Refer Slide Time: 12:16)



So, now the physical domain; that means, the metallic block. We are taking the top right hand side top coordinate and top section, top quadrant, where we can relay the grids and the boundary conditions are we have already said on this start from A (Refer Slide Time: 12:16). So, bottom surface; bottom surface if this direction x direction counts are i, so $i =$ 1 to imax, all the points, these are $\frac{\partial T}{\partial y} = 0$.

Symmetric condition $\frac{\partial T}{\partial y} = 0$. Then we will go to B. B is again the left extreme which is also symmetric condition. So, $j = 1$ to jmax, all the points, are $\frac{\partial T}{\partial x} = 0$ so, symmetric condition from the geometry. And then C, C is convective condition. So, since this point (i=1, j=jmax) has been covered by this boundary condition and this is likely to maintain the symmetry on this direction. So, we are here applying this from 2 to (imax-1).

We are leaving the last cell, because last cell, again boundary condition from right side will be defined. So, 2 to (imax-1), the boundary condition is $k\frac{\partial T}{\partial x} + hT = hT_{\infty}$. So, basically $k\frac{\partial T}{\partial x} + hT - hT_{\infty} = 0$, that we are translating in this way. And the right-hand boundary, again convective condition basically whatever is a temperature there is a heat flux based on the temperature at the end surface to the local ambient fluid and it will depend on the convective heat transfer coefficient h.

So, heat flux is $k\frac{\partial T}{\partial x}$ and the thermal energy, which is taken away by the fluid is $h(T - T_{\infty})$. So, basically, we can write this equation. Actual condition is $-k\frac{\partial T}{\partial x} = h(T - T_{\infty})$. Here also $-k\frac{\partial T}{\partial y} = h(T - T_{\infty})$.

So, these are the physical explanation of the boundary condition. And, this we have to and this again here 2 to jmax, we apply this condition (condition D). T is a temperature of the n node everywhere, all these n nodes, this T corresponds to that. And the total domain, this is 1 to imax, imax maybe you know 10, maybe 20, maybe 30, maybe 40 and this direction j equal to 1 to jmax; same, jmax can be 10, 20, 30, 40 whatever is a grid size we want to deploy.

(Refer Slide Time: 16:27)

Application and Setting up an Algorithm

B.C. for step 1

From (A) $$\frac{T_{i,2}^{n+\frac{1}{2}} - T_{i,1}^{n+\frac{1}{2}}}{\Delta y} = 0 \quad (3)$$

From (B) $$\frac{T_{2,j}^{n+\frac{1}{2}} - T_{1,j}^{n+\frac{1}{2}}}{\Delta x} = 0 \quad (4)$$

From (C) $$\frac{k\left(T_{i,jmax}^{n+\frac{1}{2}} - T_{i,jmax-1}^{n+\frac{1}{2}}\right)}{\Delta y} + hT_{i,jmax}^{n+\frac{1}{2}} = hT_\infty \quad (5)$$

From (D) $$\frac{k\left(T_{imax,j}^{n+\frac{1}{2}} - T_{imax-1,j}^{n+\frac{1}{2}}\right)}{\Delta x} + hT_{imax,j}^{n+\frac{1}{2}} = hT_\infty \quad (6)$$

Now, we have to this we have not done earlier, all the boundary conditions are to be converted into equations. How? A for example, and these are applicable for the equations 1 and 2. So, when this will be applicable to equation 1; all the temperature, boundary conditions will be found out from here, but temperatures are at $\left(n+\frac{1}{2}\right)^{th}$.

So, here basically $\frac{\partial T}{\partial y} = 0$; that means, temperature at this cell at this point, if we subtract this minus this and divided by Δy; that means, we are setting up $\frac{\partial T}{\partial y} = 0$. So, you have exactly done

$$\frac{T_{i,2}^{n+\frac{1}{2}} - T_{i,1}^{n+\frac{1}{2}}}{\Delta y} = 0 \quad (3)$$

Here, i we are keeping as dummy variable, i can run from 1 to imax or 2 to (imax-1); that means, for all these cells here 1 to imax we can run.

So, we are setting up basically symmetry boundary condition. Then, from B, B means this one;

$$\frac{T_{2,j}^{n+\frac{1}{2}} - T_{1,j}^{n+\frac{1}{2}}}{\Delta x} = 0 \qquad (4)$$

It is symmetry at this line. This is at n plus half, this is at $\left(n+\frac{1}{2}\right)^{th}$. This is equation 4.

Then from C we will write

$$\frac{k\left(T_{i,jmax}^{n+\frac{1}{2}} - T_{i,jmax-1}^{n+\frac{1}{2}}\right)}{\Delta y} + hT_{i,jmax}^{n+\frac{1}{2}} = hT_{\infty} \qquad (5)$$

So, from C we have been able to write the above equation (5). Then from D, again same thing

$$\frac{k\left(T_{imax,j}^{n+\frac{1}{2}} - T_{imax-1,j}^{n+\frac{1}{2}}\right)}{\Delta x} + hT_{imax,j}^{n+\frac{1}{2}} = hT_{\infty} \qquad (6)$$

This will be applicable for i =imax, where j will vary. These are all temperature set at $\left(n+\frac{1}{2}\right)^{th}$ level. So, these are boundary condition for step 1. Step 2 also we will apply same boundary condition, but there we will apply them at $(n+1)^{th}$ level and we have done that.

(Refer Slide Time: 21:38)

Application and Setting up an Algorithm

B.C. for step 2

From (A) $$\frac{T_{i,2}^{n+1} - T_{i,1}^{n+1}}{\Delta y} = 0 \qquad (7)$$

From (B) $$\frac{T_{2,j}^{n+1} - T_{1,j}^{n+1}}{\Delta x} = 0 \qquad (8)$$

From (C) $$\frac{k\left(T_{i,jmax}^{n+1} - T_{i,jmax-1}^{n+1}\right)}{\Delta y} + hT_{i,jmax}^{n+1} = hT_{\infty} \qquad (9)$$

From (D) $$\frac{k\left(T_{imax,j}^{n+1} - T_{imax-1,j}^{n+1}\right)}{\Delta x} + hT_{jmax}^{n+1} = hT_{\infty} \qquad (10)$$

So, again the same we can see that $\frac{\partial T}{\partial y} = 0$; that means, you know here T is from $i = 1$ and j equal to 1, i equal to 2, i equal to 3, i equal to 4, so i will be varying. And here, again i equal to 2 i equal to 3, but j is all to all j's are 2 (indicating to slide Time: 12:16). So, all these temperatures and this nodal temperature is subtracted from the corresponding nodal temperature, $T_{i,2}$, where i is varying from 1 to imax. $T_{i,1}$ again i is varying, that is what we are saying. (refer Eq. 7)

$$\frac{T_{i,2}^{n+1} - T_{i,1}^{n+1}}{\Delta y} = 0 \qquad (7)$$

We are basically fixing j coordinate to 2 and 1 divided by Δy; that means, setting up $\frac{\partial T}{\partial y}$ here and that is equal to 0. Similarly, we are setting up here. We are fixing i coordinates; that means, all (i=1,1) to (i=2,1), (i=2,j) to (i=1,j), j is varying from 1 to jmax. So, basically through this we are setting up $\frac{\partial T}{\partial x}$ on this surface and that is $\frac{\partial T}{\partial x} = 0$.

So, basically, we are not really changing anything, only we are changing all the temperatures, we are writing at $(n+1)^{th}$ level. For step 2, we wrote the boundary conditions 3, 4, 5, and 6 all the temperatures at $\left(n+\frac{1}{2}\right)^{th}$ level. Similarly, on the same boundary we are writing boundary conditions at $(n + 1)^{th}$ level. We are just using the giving the equation number 7, 8, 9, and 10.

(Refer Slide Time: 24:30)

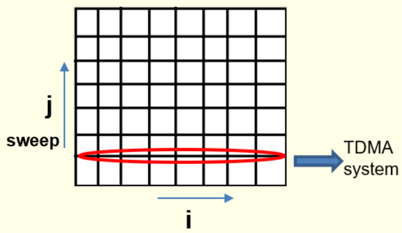
Application and Setting up an Algorithm

Solve the equations (1) and (2) together with the B.C. using Tri-diagonal matrix formulation.

Step 1 : Now for each j we can have TDM by varying i=2 to imax-1.
Use equation (1) for interior points and equations (4) and (6) for boundary points. For each j solve all i through TDMA.

$$\frac{T_{2,j}^{n+\frac{1}{2}} - T_{1,j}^{n+\frac{1}{2}}}{\Delta x} = 0$$

$$\frac{k\left(T_{imax,j}^{n+\frac{1}{2}} - T_{imax-1,j}^{n+\frac{1}{2}}\right)}{\Delta x} + hT_{imax,j}^{n+\frac{1}{2}} = hT_\infty$$



$$\begin{bmatrix} -1 & 1 & 0 & \cdots & \cdots & \cdots & 0 & 0 \\ -\frac{r_x}{2} & 1+r_x & -\frac{r_x}{2} & 0 & \cdots & \cdots & \cdots & 0 \\ 0 & -\frac{r_x}{2} & 1+r_x & -\frac{r_x}{2} & \cdots & \cdots & \cdots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \cdots & \cdots & \cdots & \vdots \\ \vdots & \cdots & \ddots & \ddots & \ddots & \cdots & \cdots & \vdots \\ \vdots & \cdots & \cdots & \ddots & \ddots & \ddots & \cdots & \vdots \\ \vdots & \cdots & \cdots & \cdots & \ddots & \ddots & \ddots & -\frac{r_x}{2} \\ 0 & \cdots & \cdots & \cdots & \cdots & 0 & -\frac{k}{\Delta x} & \left(\frac{k}{\Delta x}+h\right) \end{bmatrix} \begin{bmatrix} T_{1,j}^{n+\frac{1}{2}} \\ T_{2,j}^{n+\frac{1}{2}} \\ \vdots \\ \vdots \\ T_{imax,j}^{n+\frac{1}{2}} \end{bmatrix} = \begin{Bmatrix} 0 \\ T_{i,j}^{n} + \frac{r_y}{2}\left(T_{i,j+1}^{n} - 2T_{i,j}^{n} + T_{i,j-1}^{n}\right) \\ \vdots \\ i = 2\ to\ imax-1 \\ \vdots \\ hT_\infty \end{Bmatrix} \quad (11)$$

Now, we will see how we run the sweeps. So, solve equations 1 and 2 together with the boundary condition using tridiagonal matrix formulation. Step 1 is, now for each j, so if we are located at each j (indicating figures at slide Time: 24:30) we can have tridiagonal matrix by varying i= 2 to (imax-1).

So, 2 to (imax-1), why we are varying from 2 to (imax-1)? Because when we are at you know for example here, this is $T_{2,2}$, but $T_{i=2,j=2}$. But this is if this is $T_{i,j}$, the $T_{i,j}$ when we are writing equation and this will involve $T_{i-1,j}$ means this point (indicating in slide Time: 24:30).

Similarly, when we will come here this coordinate of this point is basically a ((imax-1),j), (imax-1) here j equal to 1, but (imax-1) is i^{th} location. So, if this is (i,j), this will involve (i+1, j); that means, (imax-1) will involve imax at j= 2, so j=2 we will vary i from 2 to (imax-1) and when we are at 2, i=2 eventually it will involve in the finite difference equations (i-1) which will be the boundary condition. When we are at i=(imax-1), then it will involve $T_{i+1,j}$ which will which is a boundary condition.

So, all i's, are varied from 2 to (imax-1), boundary conditions will be a you know taking care of (i-1) and (i+1) extremes. And this way we will be able to form at tridiagonal matrix, $T_{i,j=2}$ similarly j equal to 3, similarly j equal to 4. So, at each j we will form a tridiagonal matrix. System after solving at each j we will get the entire row. So, we are going from one j to another j and completing the enter row. And, since we are progressing in j direction, we will call it jsweep.

But while doing the j sweep observe two things; one is, at any j, i is varying from 2 to (imax-1), I have explained why, because this will take care of the boundary conditions at the left extreme and right extreme if we vary it from 2 to (imax-1) and found the tridiagonal matrix. Now, also when we are varying, we are doing j sweep, we will do it from j =2 to j=(jmax-1).

We leave these endpoints, because these endpoints will come from boundary condition, we are not now in getting involved in it. We are j = 2 we will because if we then work on this tridiagonal matrix, we will see everything is known. And we will be able to find out values at all points at this row.

Similarly, j=3, we will be able to find out all values all temperature values at the second next row. So, for each row we need a tridiagonal matrix and j will take care of the variation of row, j will run from 2 to (jmax-1) and i also for each tridiagonal matrix we will run it from 2 to at (imax-1).

Now, while forming the matrix the boundary conditions, as I said that the left boundary it is taken care of $T_{2,j} – T_{i,j}/\Delta x$; that means, del T del x at any j, del T del x at n plus half level. So, the left-hand side boundary will be handled by this equation, which was equation 4.

So, this is equation 4 and this is equation 6. If you look at it and this is equation 6 will be basically the extreme right boundary. That is

$$k\frac{\partial T}{\partial x} + hT = hT_\infty$$

$-k\frac{\partial T}{\partial x} = h(T - T_\infty)$ that boundary condition is on the extreme right.

And then we can write down the entire matrix this way. Now, we will explain it again. You can see this boundary condition has been the first row. You can see, you know this is $T_{1,j}$ next term in this column is $T_{2,j}$ and this is -1, this is +1,. So, this matrix has been formed the first line of the matrix I am reading -1 $T_{1,j}$ + 1 $T_{2,j}$ = 0, equal to 0. Intermediate all points 2 to (imax-1) will be guided by equation 1, right.

So, this is we will again if you recall, we will be through equation 1. So, all intermediate points 2 to (imax-1) is handled by this element of this matrix and each for basically we will get a tridiagonal matrix; that means, each equation we will have 3 terms (i, j), ((i-1), j) and (i+1, j) at $\left(n+\frac{1}{2}\right)^{th}$ level unknown.

So, this will create 3 diagonals. It will run, and right-hand side all the terms are known will be this term, depending on again i=2, 3, 4 this i will keep on varying and this j has been kept fixed so these are all known terms. And finally, this n boundary condition extreme right boundary condition if you look into this equation, you can get term by term here; like you can get $(-k/\Delta x)T_{imax,j}^{n+\frac{1}{2}}$.

So, $(-k/\Delta x)T_{imax,j}^{n+\frac{1}{2}}$ this term you will see will be associated with this. And since, this is (imax, j) this is also (imax,j). So, here you can see $T_{imax,j}^{n+\frac{1}{2}}$ this term, here it has a coefficient $k/\Delta x$, here it has a coefficient h. So, $\frac{k}{\Delta x} + h$ this will be the last element multiplied by $T_{imax,j}^{n+\frac{1}{2}}$. And it is just previous element is you can see $(-k/\Delta x)T_{imax,j}^{n+\frac{1}{2}}$.

So, when we multiply with this column. So, this its multiplayer is basically $T_{imax-1,\ j}$ basically this element right. So, that will be the last equation. So, this term is here arising out of this $\frac{\partial T}{\partial x}$ term. And right-hand side is h T_∞. So, this explains how this matrix equation 11 is formed.

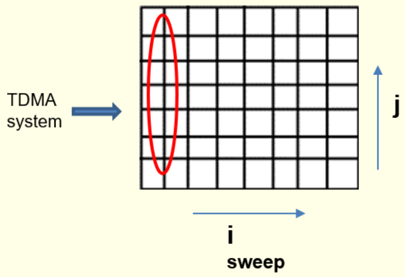
(Refer Slide Time: 35:16)

Application and Setting up an Algorithm

<u>Step 2</u> : For each " i " we can have TDM by varying j=2 to jmax.
Use equation (2) for interior points equations and (7) and (9) for boundary points.
For each i solve for all j through TDMA.

$$\frac{T_{i,2}^{n+1} - T_{i,1}^{n+1}}{\Delta y} = 0$$

$$\frac{k\left(T_{i,jmax}^{n+1} - T_{i,jmax-1}^{n+1}\right)}{\Delta y} + hT_{i,jmax}^{n+1} = hT_\infty$$



$$\begin{bmatrix} -1 & 1 & 0 & \cdots & \cdots & \cdots & 0 & 0 \\ -\frac{r_y}{2} & 1+r_y & -\frac{r_y}{2} & 0 & \cdots & \cdots & \cdots & 0 \\ 0 & -\frac{r_y}{2} & 1+r_y & -\frac{r_y}{2} & \cdots & \cdots & \cdots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \cdots & \cdots & \cdots & \vdots \\ \vdots & \cdots & \ddots & \ddots & \ddots & \cdots & \cdots & \vdots \\ \vdots & \cdots & \cdots & \ddots & \ddots & \ddots & \cdots & \vdots \\ \vdots & \cdots & \cdots & \cdots & \ddots & \ddots & \ddots & -\frac{r_y}{2} \\ 0 & \cdots & \cdots & \cdots & \cdots & 0 & -\frac{k}{\Delta y} & \left(\frac{k}{\Delta y}+h\right) \end{bmatrix} \begin{bmatrix} T_{i,1}^{n+1} \\ T_{i,2}^{n+1} \\ \vdots \\ \vdots \\ T_{i,jmax}^{n+1} \end{bmatrix} = \begin{Bmatrix} 0 \\ T_{i,j}^{n+\frac{1}{2}} + \frac{r_x}{2}\left(T_{i+1,j}^{n+\frac{1}{2}} - 2T_{i,j}^{n+\frac{1}{2}} + T_{i+1,j}^{n+\frac{1}{2}}\right) \\ \vdots \\ j = 2\ to\ jmax-1 \\ \vdots \\ hT_\infty \end{Bmatrix}$$

(12)

Then we go for step 2. It is just you know alternate direction. So now, we will have **isweep** we will progress from i = 2 to (imax-1) and j will vary and in equation 2 will vary j from 2 to (jmax-1). So this will form that tridiagonal matrix and values at $(n+1)^{th}$ level are not known. These are the unknown values $(n+1)^{th}$ level. And again, you can see the first equation that is in this direction $(T_{i,2} - T_{i,1})/\Delta y$.

So basically, when we are here, we have to apply basically symmetric condition here. So, this is a symmetric condition here $(T_{i,2} - T_{i,1})/\Delta y$. And when we are here again, we will get basically the top boundary which is again convective boundary. So, that is $(T_{i,jmax} - T_{i,jmax-1})/\Delta y$; that means,

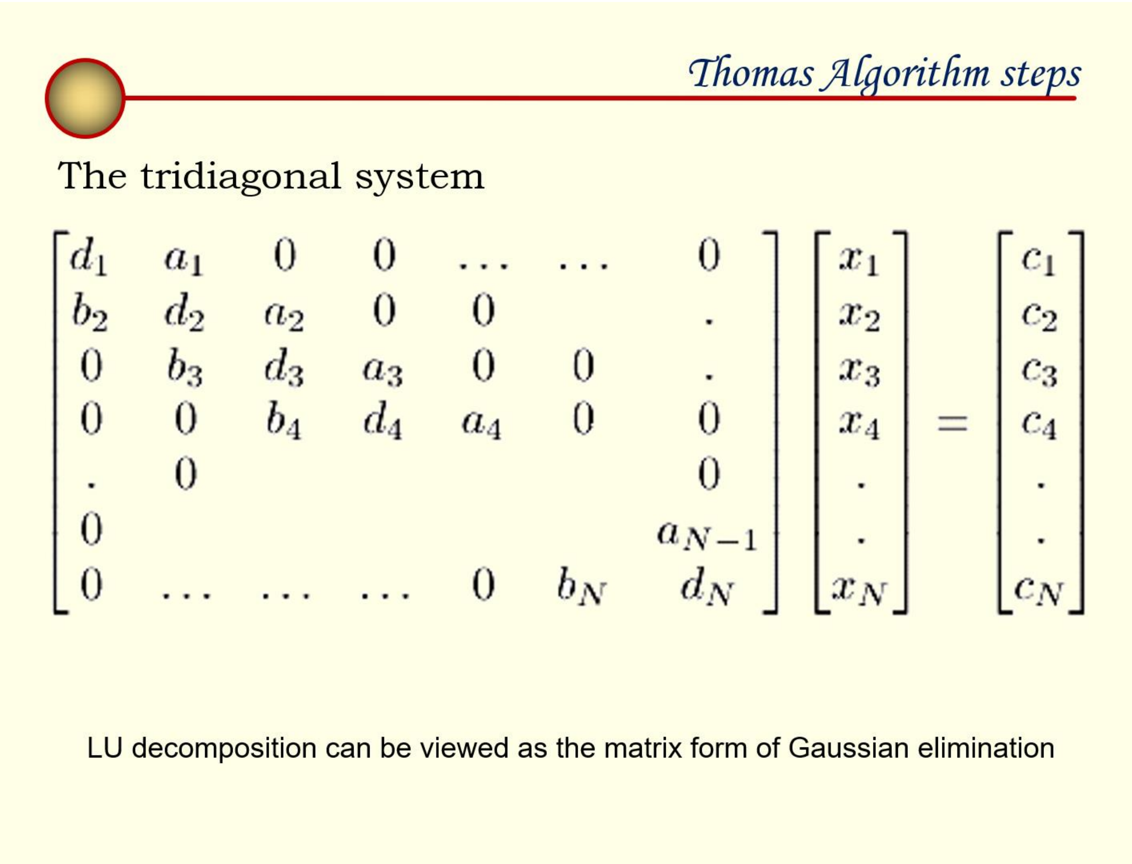
$$\frac{k\partial T}{\partial y} + ht = hT_\infty$$

And here at the bottom it is $T_{i,2} – T_{i,1}$, set up del T del y equal to 0. And again, all intermediate points again you know will be guided by equation 2; if you recall just say equation 2 will be the all-intermediate point. So, these are j = 2 to jmax-1 and $–(r_y/2)\ T_{i,j-1}$ $(1+r_y)\ T_{i,j} – (r_y/2)\ T_{i,j+1}$, those are unknown terms and right-hand side is known and that is how this tridiagonal matrix is set up.

Again, like we checked in the last case. See first equation you can get here $T_{i,1}$ and $T_{i,2}$ in this column. Now, we know basically j we are varying and i is kept fixed for each i we will vary j. So this is I sweep, i will first we will fix i equal to 2, will find all the values then go to i equal to 3, find all the values, i equal to 4 all the values and we are fixing i we are basically forming a tridiagonal matrix for that i.

So, here we have done that. And the here you can see again the last equation like $\frac{k\partial T}{\partial y} + ht = hT_\infty$, here also last equation is basically interpretation of this. You can see, minus k del T del y with jmax-1 term. Then k by delta y plus h with T i, j max term and right hand side is h T_∞, equation 12.

(Refer Slide Time: 39:39)



$$\begin{bmatrix} d_1 & a_1 & 0 & 0 & \dots & \dots & 0 \\ b_2 & d_2 & a_2 & 0 & 0 & & . \\ 0 & b_3 & d_3 & a_3 & 0 & 0 & . \\ 0 & 0 & b_4 & d_4 & a_4 & 0 & 0 \\ . & 0 & & & & & 0 \\ 0 & & & & & & a_{N-1} \\ 0 & \dots & \dots & \dots & 0 & b_N & d_N \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ . \\ . \\ x_N \end{bmatrix} = \begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \\ . \\ . \\ c_N \end{bmatrix}$$

So, basically, we are supposed to get a tridiagonal matrix where this x vector is unknown and the c vector is known and this tridiagonal matrix d is the diagonal, b is the immediate sub diagonal and a is the super diagonal.

(Refer Slide Time: 40:08)

Application and Setting up an Algorithm

Algorithm:

1. Initialize Δx, Δy, h, k, Δt, Told [imax, jmax], Tnew [imax, jmax].
2. Apply initial condition Told [imax, jmax] = T_0 for interior and boundary points.
3. Two stages of time stepping:

   (a.) Sweep for each j (except for j = 1 and j = jmax)

Sweep initially in the y-direction for each j (all interior points), i.e. except for j = 1 and j = jmax, call the Tri-diagonal subroutine.
Tri(a, b, c, d, r_x), where the array variables namely a, b, c, d, r_x are obtained from equation (11).

$$c(1) = 0 \;,\; c(imax) = T_\infty \;\; and \;\; c(i) = T_{i,j}^n + \frac{r_y}{2}\left(T_{i,j+1}^n - 2T_{i,j}^n + T_{i,j-1}^n\right)$$

$$for \; 2 \le i \le imax - 1$$

$$a(1) = 0 \;,\; d(1) = -1 \;,\; b(1) = 1$$

$$a(i) = -\frac{r_x}{2} \;,\; d(i) = 1 + r_x \;,\; b(i) = -\frac{r_x}{2}$$

So, we will set it up. We will initialize Δx, Δy, h, k, Δt, T_{old}. Now we are setting up the algorithm. So, basically these are the arrays or variables we are discussing and we are initializing. We are defining Δx, Δy, h, k, Δt, T_{old} this is a subscripted array we declare and T_{new} another subscripted array we did declare and dimension of this array at the maximum points in arrays are maximum points in i and j direction.

Apply initial condition T_{old}[i=1 to imax, j=1 to jmax]=T_o. That means the initial temperature which is higher than the ambient temperature. Now, two sweeps for each j, except for j=1 and jmax for each j we form the tridiagonal matrix. So, sweep initially in the y-direction for each j that is except for j=1 and j=jmax we have seen.

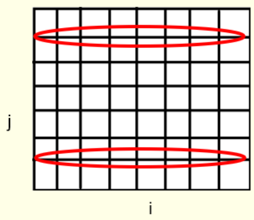
So that is why we are calling in j, we are going from j = 2 to j = (jmax-1), but for each j we are forming tridiagonal matrix. And this way we define a Tri(a, b, c, d, r_x), and these are the values you know array variables namely a, b, c, d, r_x are obtained from equation 1. And you can see, $c(1)$ is 0, like if you are equation 1 or equation 11. $c(1)$ is 0 these are all intermediate $c(2), c(3), c(4) \ldots$ value and this is my finally, you know $c(n)$ value.

So, $c(max)$ is T_∞, and all these intermediate c_i values at this. Similarly, you know these diagonal values if we calculate. So, we can get from here also these are all diagonals. So, you can see that first value is $d(1)$ is -1. All intermediate d_i's are $1 + r_x$. and finally, $d(max)$ is $\frac{k}{\Delta x} + h$.

(Refer Slide Time: 43:17)

Application and Setting up an Algorithm

$$\frac{T^{n+1/2}_{i,\mathrm{jmax}} - T^{n+1/2}_{i,\mathrm{jmax}-1}}{\Delta y} + hT^{n+1/2}_{j\max} = hT\infty$$



$$\frac{T^{n+1/2}_{i,2} - T^{n+1/2}_{i,1}}{\Delta y} = 0$$

We are calculating for each j, j = 2 to jmax - 1.

For, $2 \le i \le imax - 1$

$a(imax) = -\frac{k}{\Delta x}$, $d(imax) = \frac{k}{\Delta x} + h$, $b(imax) = 0$

The Tri-diagonal system returns the x(i) array.

- Store the Tnew [i, j] = x(i).
- Apply B.C. for j = 1 and j = jmax from equations (3) and (5).
- Substitute Told [i, j] by Tnew [i, j].

(b.) Sweep for each i (except for i = 1 and i = imax)

Here the 2nd Tri-diagonal sweep has to be done. For all i except i=1 and i=imax, call the subroutine Tri(a, b, c, d, r_y), where the array variables namely a, b, c, d, r_y takes the values from equation (12), which again returns x(i).

- Update Tnew [i, j] = X(i).
- Apply B.C. for i = 1 and i = imax from equations (8) and (10).

Now, these a values, we are calling it a we as you can see $a(1)$ is 0, a_i i's are $-\frac{r_x}{2}$ and $a(imax) = -\frac{k}{\Delta x}$. Although, we are patterning it after this you know tridiagonal matrix, we have called our d arrays are diagonal arrays our c arrays are same, but in this problem, this sub diagonals are defined as in our current problem $a's$. Just as little change $a(1) =$

0, $a(i)$'s are all $-\frac{r_x}{2}$ and $a(imax) = -\frac{k}{\Delta x}$. And b's are just you know we are calling super diagonal set b. So, you can see $b(1) = 1$, b all $b(i)$'s are $-\frac{r_x}{2}$ and $b(imax) = 0$, and that is how we get the you know all the arrays of the diagonal, sub diagonal, super diagonal these arrays.

Immediate sub diagonal, immediate super diagonal, these arrays and c are the column vectors. And once we get that, we basically we have got the entire tridiagonal matrix and from that tridiagonal matrix the finally, the vector that we evaluate is $x(1)$ to $x(n)$. So, that is what we will see.

So that tridiagonal system returns $x(i)$ array. So, this is how each tridiagonal matrix we give, we will give us $x(i)$ array, means all these x points temperature points. So, store $T_{new}[i,j]$ equal to $x(i,j)$. So, that will give the temperature distribution at each point. Then again, we will go to next j; we will get next all points. So, we will keep on varying j.

So, but this new array we will store and for each array after storing, we what will do? We will apply boundary condition which is which we have not applied so far. Because if you look at the boundary condition for equation 1, see we have applied 4 and 6, but we have not applied 3 and 5.

So, 3 and 5 can be applied now, because we have found out these, we have found out next j point all the up to we have got (jmax-1). All these points we have found out, but we have not found out these points and these points (referring to Slide Time: 43:17). But this points, this from 3 and this is from equation 5. So, this is basically here again at each point now we have to vary i and set $\frac{\partial T}{\partial y} = 0$ at each point.

Similarly, we have to vary i and $-k\frac{\partial T}{\partial y} = h(T - T_\infty)$ we have to set at all these points. Then we complete the task we for the j sweep, and sweep for each i, then we again change the direction. So, for each i then we go for formulation of tridiagonal matrix in j direction.

Like you know this for each i, i sweep will do now tridiagonal matrix in j direction. So, that is what is the next level. But this is just what we did in 3, (a) sweep for each j we will form tridiagonal matrix from i equal to 1 to i equal to (imax-1) point.

Here also we will repeat the same thing in alternate direction; that means, we will fix i points except for i equal to 1 and imax, but any intermediate point, say i=3. Then we will vary j=2 to (jmax-1) and we will form, from equation 12, a tridiagonal matrix. We will solve that all the points will be evaluated will go to, i=3.

And again, we will vary j=2 to (jmax-1) will find that tridiagonal matrix will solve it will move to i=4, again j=2 to (jmax-1) again tridiagonal matrix again we will get the values of at all the points. Here we have written that second tridiagonal sweep has to be done for all i is except i=1 and imax, call the subroutine Tri(a, b, c, d, r_y),, but the array variables a, b, c, d, r_y, takes the values from equation 12, which again returns $x(i)$ the unknown matrix and update T_{new}[i,j]=x(i). And then again apply boundary condition equation from equation 8 and 10.

Like here, after having evaluated everything, we had to apply 3 and 5 in order to say these two boundaries right, these two boundaries correct. So, similarly when i = 1 to i = (imax-1), we will sweep and we will form tridiagonal matrix for each i-point involving all j's. After completing this we this column has to be done, but this point's column or points have to be calculated from equation 8 and 10. Like here these rows of points we calculate it from 3 and 5. So, similarly 8 and 10 we will have to calculate all this confining boundary conditions are set up set up confining boundary correctly.

(Refer Slide Time: 51:45)

Application and Setting up an Algorithm

4. If it is required to predict the temperature field at a specified time level, then divide the total time duration by four/five equal divisions and each division is counted as the time step, Δt.Use step 3 to complete the calculation for each time step. Having calculated the temperature field after each time step, initialize the temperature field by the most recent value and progress.If the desired time level is reached, go to step 6.

5. If you are required to reach the steady state temperature,then take a time step that is reasonable and calculate the temperature field iteratively using step 3.At the end of each time step, compare the temperature values at each location between two time steps.Check if the maximum discrepancy at any (i, j) location falls below a specified small value, say ε, effectively, $\max(dis_{i,j}) < \varepsilon$, then go to step 6. Else go to step 3 and repeat.

6. Stop.

And you know, then one complete time step is over, we will go to next time step, we will go to next time step. That is how the calculation will progress. If it is required to predict the temperature field at a specified time level, then divide the total time duration by four or five equal divisions and each division is counted as a time step Δt. Use step 3; that means, these two sweeps, j sweep and i sweep.

So, use step 3 to complete the calculation for each time step. Having calculated the temperature field after each time step, initialize the temperature field by the most recent value. That means, for example, you have to go find out a hot plate temperature after 10 seconds. So, maybe you divide 10 seconds you know by 4, hence after each 2.5 second you find out the total distribution. And now this will be the initial distribution from here again you go to Δt go.

This will be from n to again go to (n+1). When you have gotten (n+1), then again that will be n you go to next (n+1). So, that is how if you know when you reach the desired time level you go to 6; that means, stop. Some problems are such that you know such cooling problem; you just define it little differently. Instead of knowing temperature after some given time you want to reach a steady state temperature.

In that case, if you are required to reach the steady state temperature then take a time step that is reasonable, because here you have no restriction by stability, you can take larger time steps. But this all this cannot be also you know too large a time step. I will discuss this point later. Here even though no restriction, we cannot arbitrarily take a very large time step.

So, reasonable and calculate the temperature field iteratively using step 3. At the end of each time step compare the temperature values at each location between two-times steps. Each location; that means e.g., $T_{2,2}$, to $T_{2,2}$. Same location, what was the temperature at n and what is at $(n+1)^{th}$ level. And, this discrepancy you surveyed, over the scan, over the entire field.

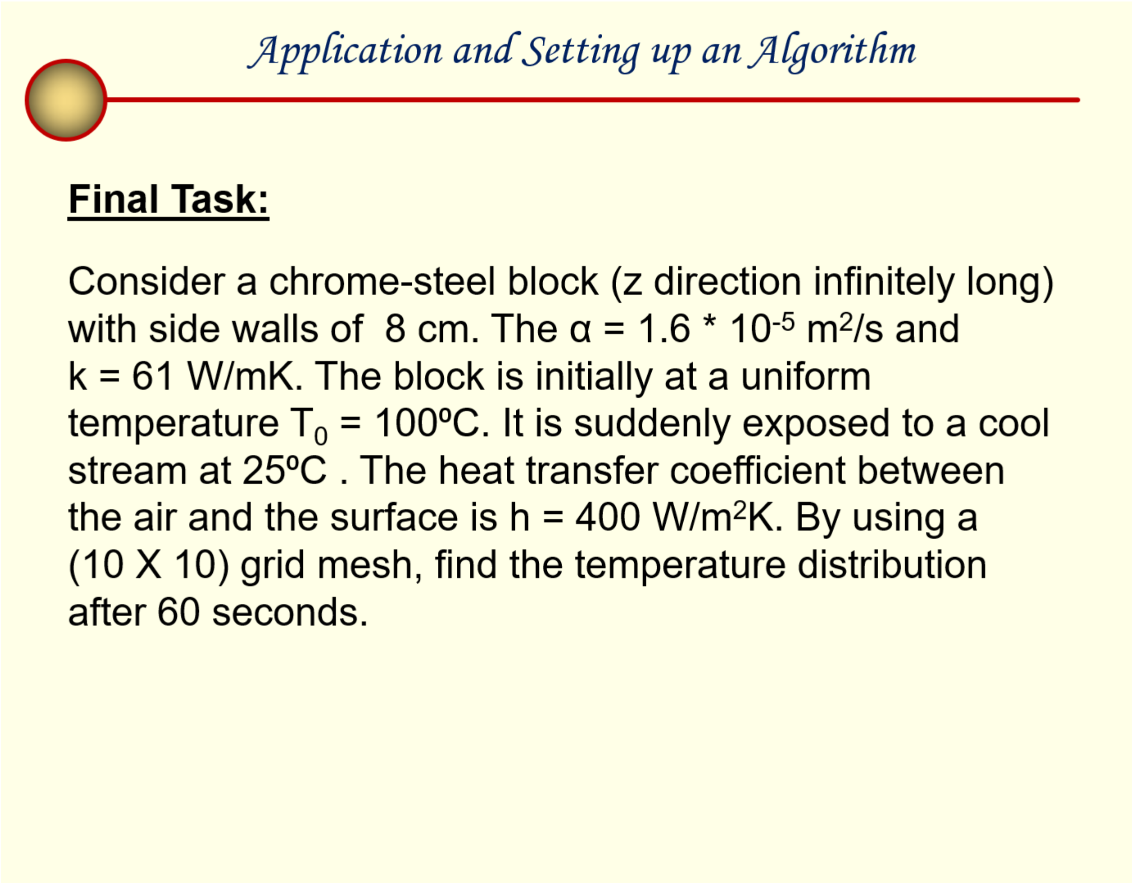
Now check if the maximum discrepancy at any i,j location, you have to keep on calculating, that till this and this maximum discrepancy can occur at any point. So, wherever it occurs, this you will keep on iterating and applying boundary condition till it falls below a specified value say ϵ. ϵ will be 10^{-3} to 10^{-4}, something like that.

So, basically, we are reaching a stage where temperature is not changing after third decimal place or forth decimal place, there is no change in temperature. So, basically you know specified as I said that check the maximum discrepancy at any (i, j) location falls below a specified small value, say epsilon, effectively, maximum discrepancy at any point, (i, j) point. It can occur anywhere, after each time step it has to be can till the maximum discrepancy fall false below ϵ.

And if it does that; that means, you have reached study state you go to step 6, else go to step 3 and repeat. So, I have just taken up this example, because this is a way you have to set up a problem, this is the way you have to plan an algorithm and then you go for programming. For me programming can be done in any language whether you use C or C++ or Python or Fortran 77, Fortran 95, Fortran 77, all are scientific programming language. These languages are used by several numerical algorithm subroutines.

But there is no restriction if you know, you can do that otherwise use C, C++, some of you are you know these days conversant with Python. You can even use MATLAB programming. What, but main focus is you set up your algorithm correctly. If you can set up your algorithm correctly, whatever language you use does not matter. You will get the correct result.

(Refer Slide Time: 57:20)



So, we have done this all this arising out of this discussion. It excites me to give you a problem. Also probably, it will be very you know satisfying experience for you if you can solve this problem by yourself. So, you really now know everything. You have to again just what we did repeat that problem with numerical data.

Consider a chrome steel block z-direction is infinitely long, again I should have written something else. Not infinitely long it is very long. It is so long that it is not comparable with x and y direction dimensions. So, whatever will happen at any x and y it will keep on repeating if j direction is very long. So, only end effect will be something, but it will not influence the entire route. There may be some little end effect.

So, it is a read infinitely long block with a very long block, not infinitely long block with sidewalls 8 centimetre. So, x and y dimensions are 8 centimetres. α is $1.6 \times 10^{-5}\ m^2/s$ Thermal conductivity is 61W/mK watt per meter second. And so thermal diffusivity and thermal conductivity you can see, the block is initially at a uniform temperature T_0, as I said it is a heated block. Temperature is $325^o\ C$ it is suddenly exposed to cool stream at $25^o\ C$. The heat transfer coefficient between the air and the surface that is h is 400 W/m^2K. By using a 10 by 10 grid mesh, find the temperature distribution after 20 seconds. So, now you know Δx you know Δy, you know h, you know initial temperature, you know the time interval after which you have to determine the temperature all property values are known after 20 seconds.

Maybe you can make smaller time steps, maybe after each 2 if you set up a program it does not matter. Maybe after each 2 seconds, 2 seconds can be your intermediate your Δt. So, or maybe 4 second can be your Δt. So, then it will be 5 iterations in time direction.

So, after you choose any value, you can take 2 you can take 4 seconds, you can take 2 seconds and, how about for each 2 seconds or for each 4 seconds, you have to progress you know through a time step of $\Delta t/2$ from n to (n+1/2) then (n+1/2) to (n+1), then that will be the you know initial value.

So, this way you know after 20 seconds what is the temperature distribution on the block. That means, temperature at all 100 points; 10 into 10 grid mesh you have to find out and submit. I will formally give this problem as an assignment so that all of you can get credit for this, but practice it. Practicing it will be really useful.

It will form steps for confidence building. Once you can solve you get really a result from your computer for all 100 points temperature and they are reasonably correct, correct to second decimal place, that is good enough. You will have the confidence that you can formulate a problem and you can solve the problem you know which was seemingly unknown to you even few days back. Thank you very much, I will stop here today.

Thank you.