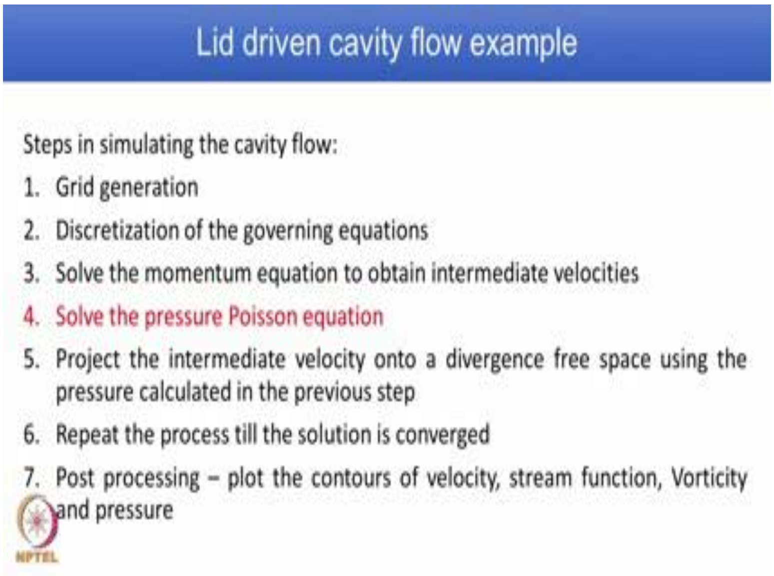
Foundation of Computational Fluid Dynamics
Dr. S. Vengadesan
Department of Applied Mechanics
Indian Institute of Technology, Madras

Lecture - 41

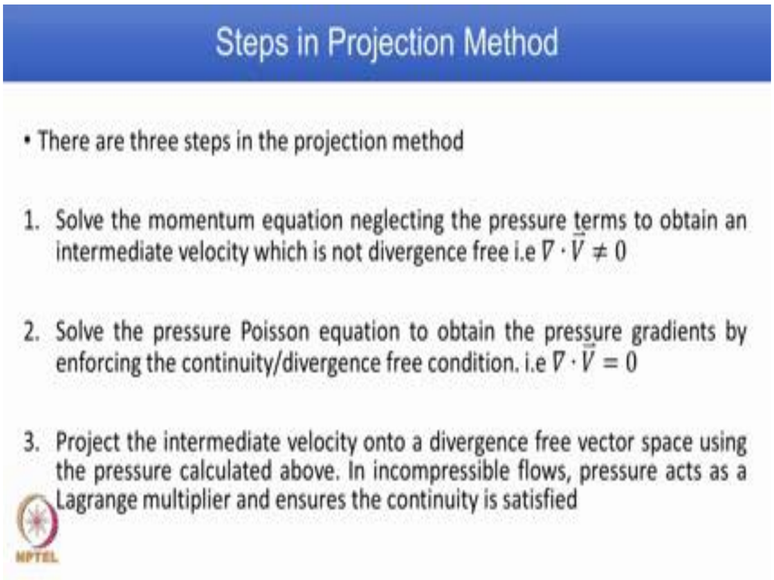
It is my pleasure to extend greetings again to all of you. We are now onto the module four of this week. In this week we mentioned, we will actually explain with a help of the working code, how to write a code and for that purpose, we have taken a test case problem of flow in a lid driven cavity. In the last three lectures, we explained about grid and then different components of u-momentum equation, v-momentum equation. In other words, how to writes separately for convection term, diffusion term and put term together to get u star, v star. And then we also explained how to enforce boundary condition as soon as you solve momentum equation, we have to impose boundary condition, and for the left wall, right wall, bottom wall and top wall respectively. Now in this module, we are going to talk particularly about the pressure or the projection method, how to solve pressure Poisson equation and enforcing boundary condition for pressure Poisson equation and corresponding code with explanation.

(Refer Slide Time: 01:36)



So for the sake of continuity, I am listing here all the steps that we mentioned to solve this particular problem. So, grid generation we have already done; discretization of the governing equations, separately convection term, diffusion term, solving the momentum equation to obtain intermediate velocities u star and v star. Now in this class, we are going to talk about how to solve pressure Poisson equation. Then we project the intermediate velocity onto the divergence free space using the pressure calculated through the previous step, correct the velocities and then we repeats the steps until the solution is converged. Last step, once we get all the variables we look into the flow to different post processing.

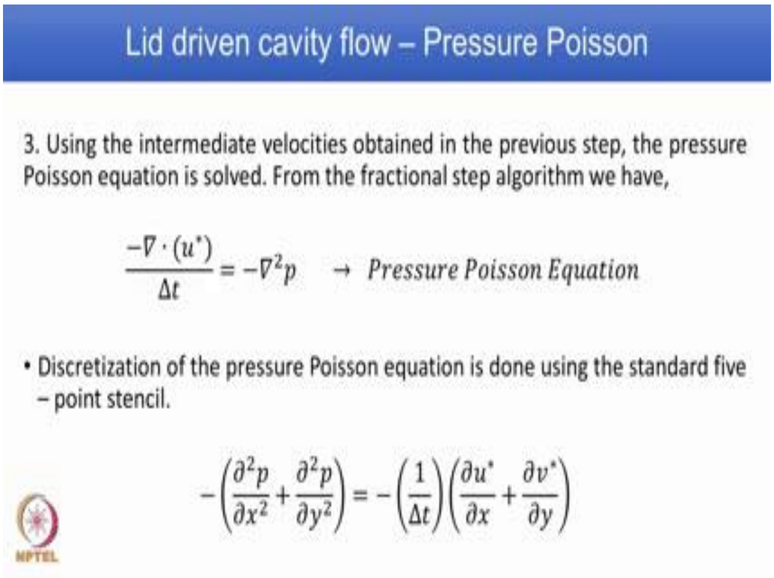
(Refer Slide Time: 02:25)



We mentioned we are following to do the pressure velocity coupling a procedure called projection method. In projection method, there are three steps involved; first - solve the momentum equation without considering the pressure term. So, we obtain what is known as intermediate velocity denoted as u star and v star, because we are not consider pressure term it will not satisfied the continuity equation. In other words, delta dot v is not equal to zero. Then you solve the pressure Poisson equation to obtain pressure gradients enforcing continuity or divergence free condition that is del dot v equal to zero.

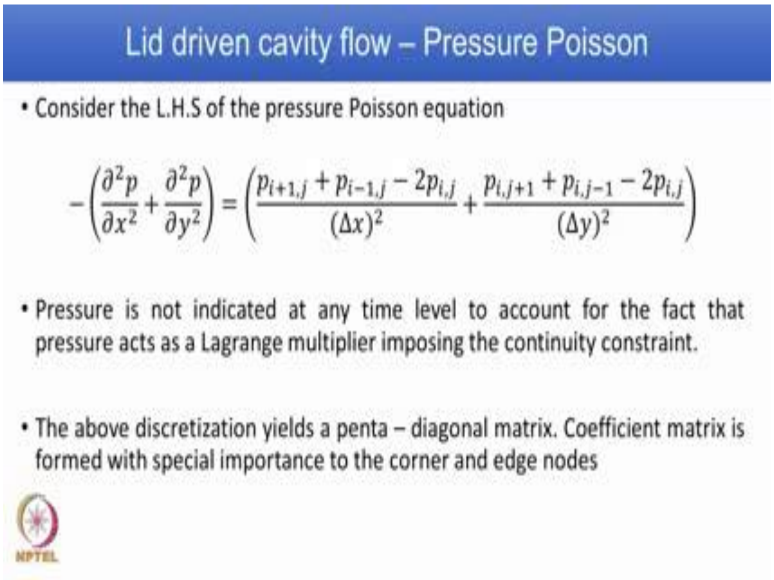
Project the intermediate velocity onto the divergence free vector space using the pressure calculated in the previous step. And in incompressible flows the pressure acts as a Lagrange multiplier and ensures the continuity is satisfied. This is the step that involved in pressure-velocity coupling. Please recall we listed four methods MAC algorithm, Marker and Cell algorithm, SIMPLE and different versions of SIMPLE that is SIMPLE or SIMPLE C and then projection method. For this demonstration, we are using projection methods.

(Refer Slide Time: 03:56)



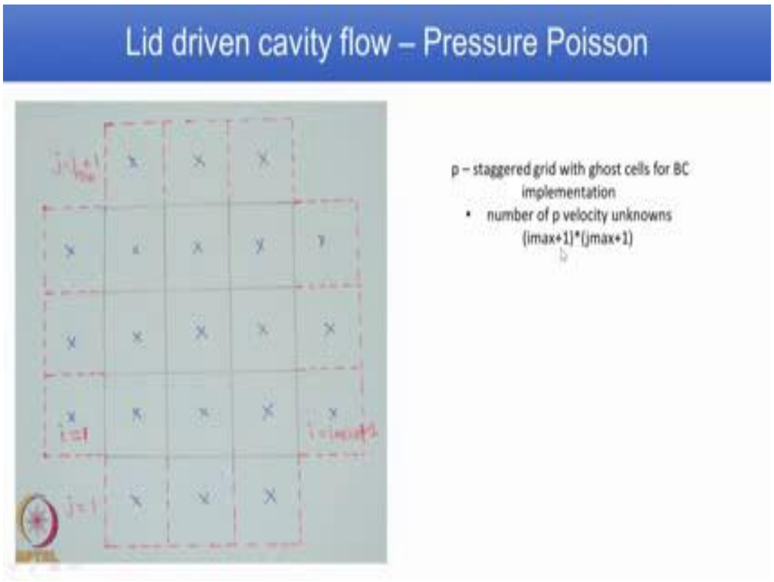
This is the third step that is using the intermediate velocities obtain in the previous step that is by solving u momentum and v momentum equation, pressure Poisson equation is set up and it is solved. So, we have del dot u star by delta t equal to del square p which is the pressure Poisson equation. Discretization of the pressure Poisson equation is done using the standard five point stencils. Such as dou squared p by dou x square plus dou square p by dou y square minus and is equal to minus one upon delta t multiplying dou u star by dou x plus dou v star by dou y, this is the pressure Poisson equation, this is separately obtain we already listed how to obtained this pressure Poisson equation.

(Refer Slide Time: 05:00)



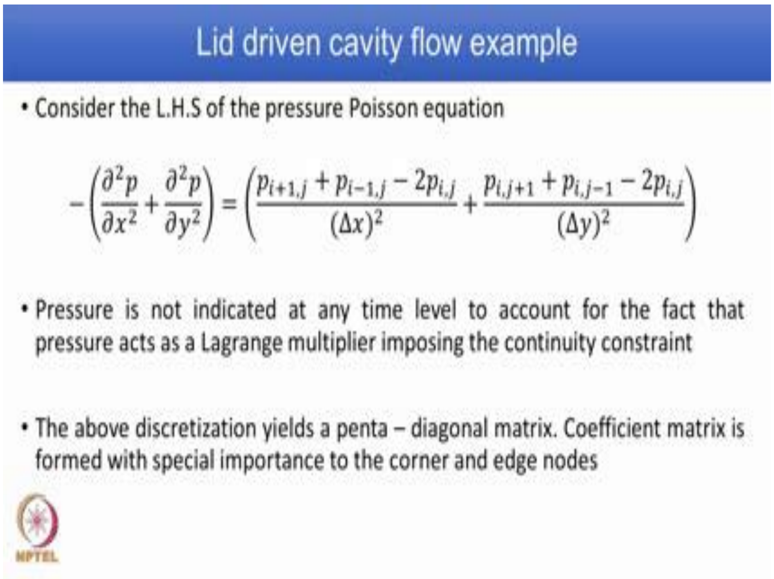
So, you consider the left hand side of the pressure Poisson equation that is minus dou square p by dou x square plus dou square p by dou y square, because it is secondary derivative we use second order central different switch scheme for pressure. So, we get p at i plus 1 comma j plus p i minus 1 comma j minus 2 p i comma j by delta x square for the first term. Now for the second term that is second derivative in the y-direction, we have p i comma j plus 1 plus p i comma j minus 1 minus 2 p i comma j by delta y square. Pressure is not indicated at any time level to account for the fact that pressure acts as a Lagrange multiplier imposing the continuity constraint that mean the pressure is only local at the particular time. The above discretization results in what is known as penta diagonal matrixes; again please recall we mentioned different matrixes possible; tridiagonal matrix, just diagonal and penta diagonal matrixes of different form. So, this pressure Poisson equation discretization of the pressure Poisson equation will result in penta diagonal matrix. Coefficient matrix is formed with special importance to the corner as well as edge nodes.

(Refer Slide Time: 06:34)



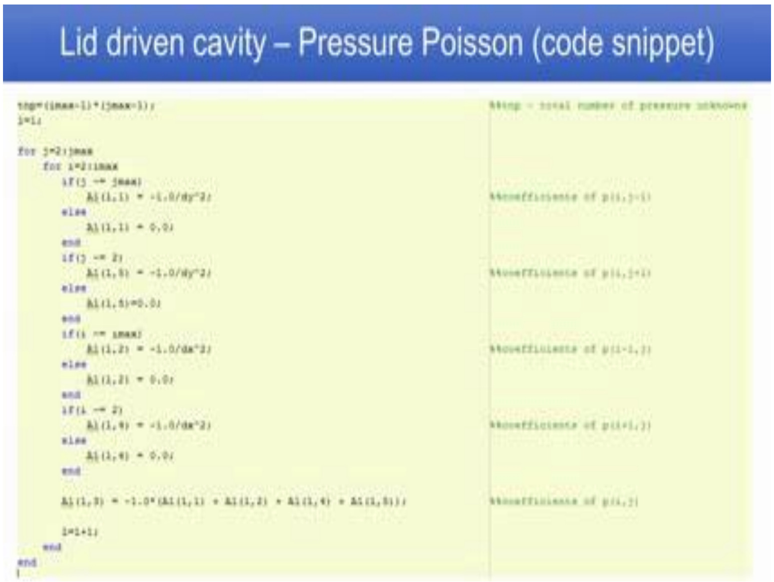
So, we have again ghost grid and ghost cells strategy to impose boundary condition. Please recall what we learned in the two previous modules; separately we have done for u velocity and separately we have done for v velocity. Then we respectively solved u momentum equation and v momentum equation, because we are following staggered grid arrangement of variable storage, we need to have one extra grid on either side for the respective equation. And pressure is it the centre of the shell in staggered grid arrangement. So, we have ghost grid as well as ghost shell on all the sides that is both x as well as in y. And those are shown by this a red colour line. So, these are all extra cell in x-direction similarly in other x-direction; and for y-direction at the top as well as at the bottom. So, p staggered grid with ghost cell for boundary condition implementation because of this extra ghost node or ghost cell number of p velocity unknowns now becomes i max plus 1 into j max plus .

(Refer Slide Time: 08:00)



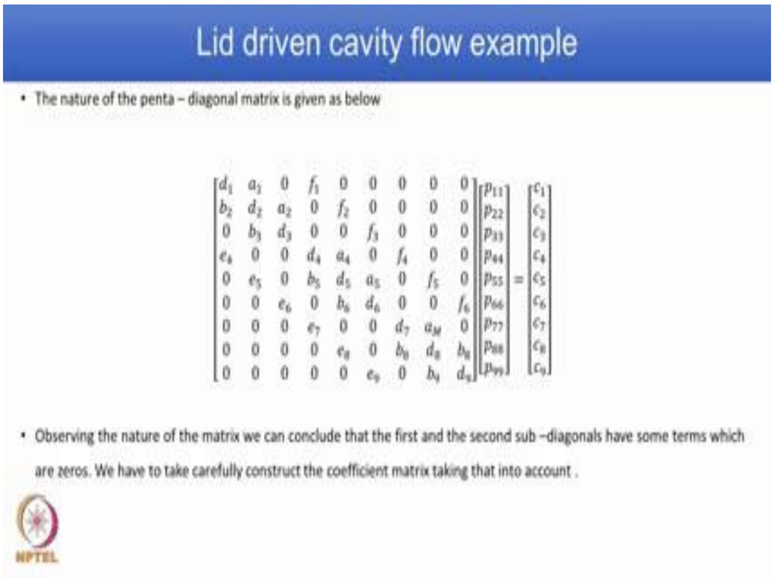
Consider the left hand side of the pressure Poisson equation, so again that is what I am showing here, we have already mentioned we are following second order central difference scheme to get the second derivative, and then we mentioned it will result in what is known as penta diagonal matrix.

(Refer Slide Time: 08:20)



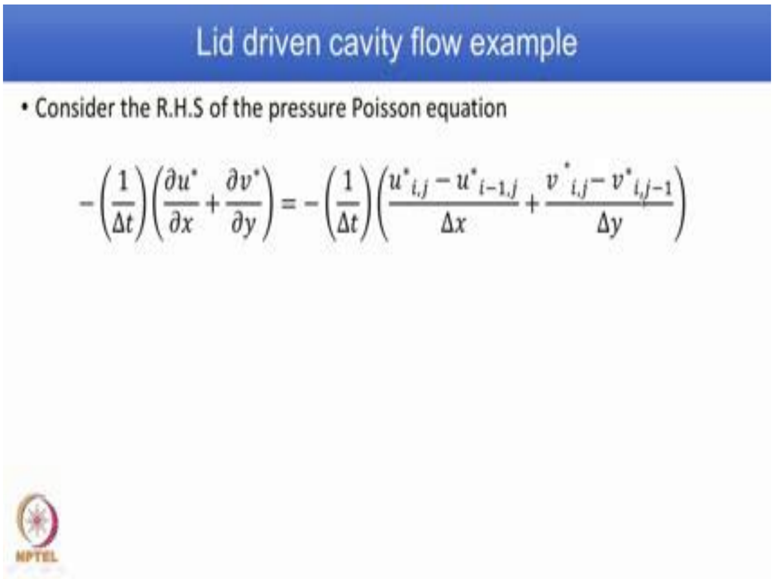
So, corresponding code is displayed here. So, tnp is number is nodes i max minus one into j max minus one that is the total number of pressure nodes; and for j 2 to j max, and for i two to i max if j is not equal to j max then we have define this. So, this for coefficients of pressure for the node i comma j minus 1. We have written similar thing for each location. So, this for example, this particular line is for coefficients of pressure at i comma j plus 1, we have written similarly for each. So, coefficient for p at general nodes i comma j is shown here, and I would like emphasize once again these special treatments that is i j minus 1 i j plus 1 i minus 1 j and i plus 1 j these are for corner nodes and edge nodes and this particular line is for all internal nodes.

(Refer Slide Time: 09:34)



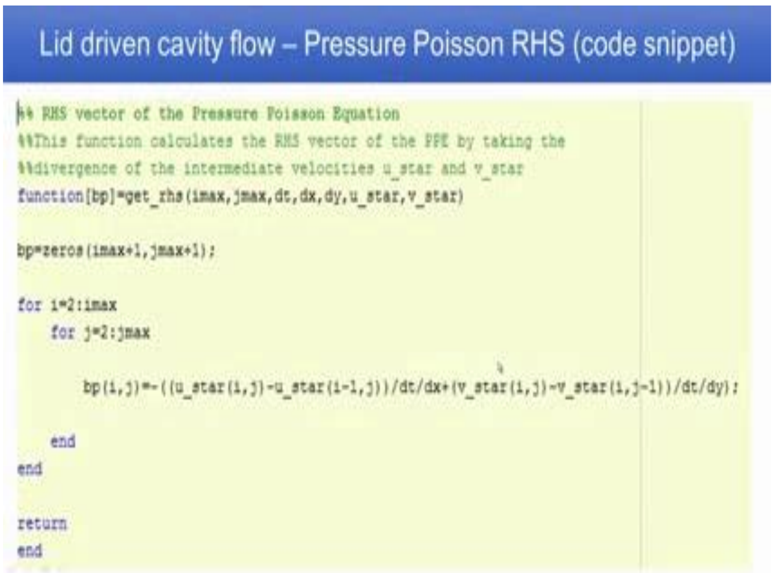
So, the nature of the penta diagonal matrix after the coefficients are evaluated is shown here. So, we have d 1 a 1 then there is the element 0 then there is the value f 1. Along the diagonal of the coefficient matrix, we have value; immediately below sub diagonal, we have value; immediately above sub diagonal, we have value, then we haves zeros and then values f 1 to f 5 f 6; similarly on the lower side, e 4 to e 9 after the zero values. And pressure column vector, unknown column vector is multiplying the coefficient matrix equal to central differences scheme known value is written as a known vector on the right side. If you observe the nature of the matrix, we conclude that first and second sub diagonal have some terms that is what is this which are zeros and we have to carefully construct coefficient matrix taking that into account.

(Refer Slide Time: 10:49)



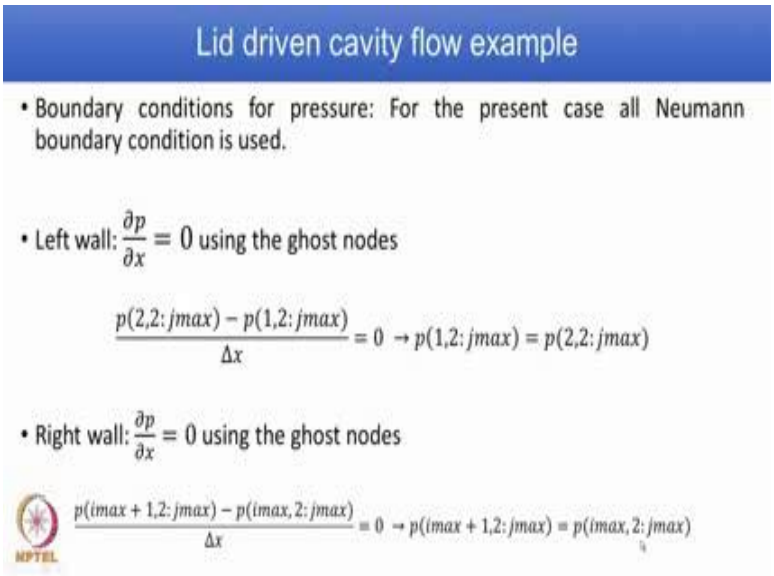
We have explained how do to the L. H. S of the pressure Poisson equation; now we do for right hand side of pressure Poisson equation. The right hand side pressure Poisson equation as actually the source term, which is related to intimidate velocities u star and v star as shown here that is minus one upon delta t equal to dou u star by dou x plus dou v star by dou y. We write to finite difference form of the source term as shown here. So, one upon delta t u star evaluated at i comma j node minus u star from i minus comma j node by delta x plus v star from i comma j node minus v star from i comma j minus 1 node by delta y.

(Refer Slide Time: 11:41)



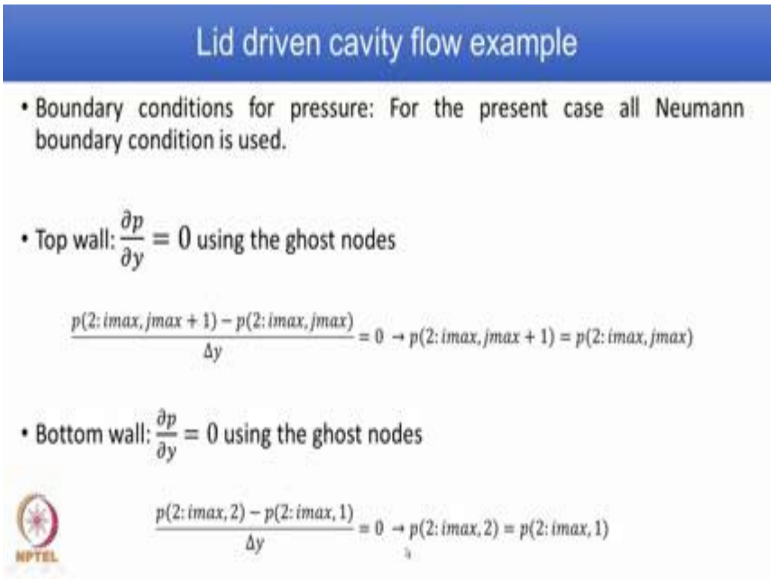
So, corresponding code is displayed here. So, the right side is actually right side column vector is actually given as b p commands are returns first few lines. So, R.H.S vector of the pressure Poisson equation, this function calculates the R.H.S vector of pressure Poisson equation by considering divergence of the intermediate velocities u star and v star. So, we define the function with corresponding arguments as shown here. Initially they are set to zeros with the memory size related to number of nodes as shown here. Now for i 2 to i max and for j 2 to j max, b p is calculated as shown here. This is exactly what we explained in the previous slide; only thing in it is written in mat lab code form as shown here. So, we have calculated separately the right side term and separately left side term.

(Refer Slide Time: 12:45)



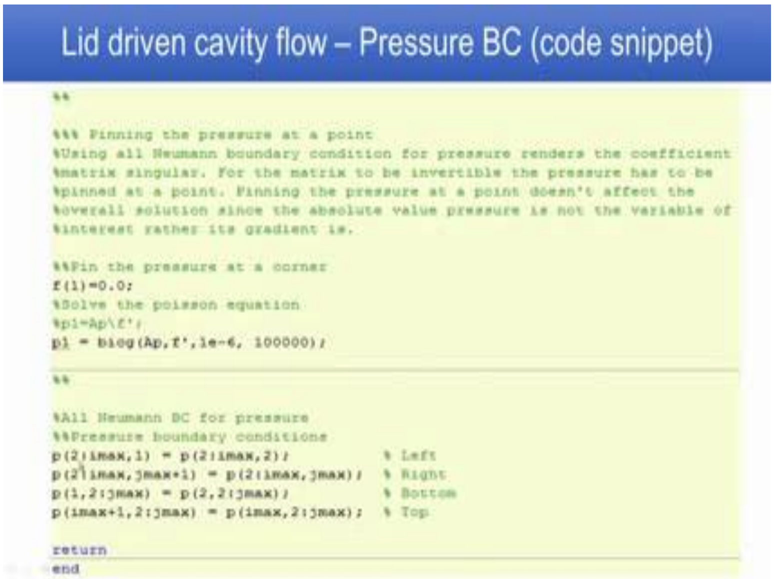
Now we have to enforce boundary condition for pressure. For the present case that is for the pressure we enforce boundary condition in form of Neumann type; for velocity, we enforce through Dirichlet boundary condition. So, the left side wall dou p by dou x equal zero and using the ghost nodes, we need to have extra ghost nodes enforced boundary condition. So, p at i is equal 2, and for j 2 to j max minus p at i is equal to one against j 2 to j max by delta x equal to zero. So, this is actually dou p by dou x equal to zero. So, if you rewrite this equation, so we have p at 1 comma 2 to j max equal p at 2 comma 2 to j max. Similarly for the right side wall, again Neumann type boundary condition for pressure dhow p by dou x equal to 0, we have i as i max plus 1 2 to j max for j, inside of that wall, so p i max and j 2 to j max by delta x equal to 0. So, if you rearrange, we get value for the ghost cell node such as p i max plus 1 j going from 2 to j max equal to p i max that is inside the right side wall and for j it is 2 to j max.

(Refer Slide Time: 14:39)



Similarly, the top wall dou p by dou y equal to 0; again similar arrangement only thing you have to now pay attention to j value. So, it is j max plus 1, which is above the top wall and j max which is just below the top wall equal to delta y, so that will result in dou p by dou y equal to zero. And if you rewrite p 2 to i max for x-direction and j max plus one is just above the lid equal to p 2 to i max for i, and j max in to just below the lit. Bottom wall dou p by dou y equal to zero, again using the ghost nodes now we have to pay attention carefully again for value of j. So, i is from 2 to i max, j is two minus again p at 2 to i max j is at 1 by delta y is equal to zero. And you rearrange, we get expression as shown here.

(Refer Slide Time: 15:53)



So, there are all implemented as shown in this code. We will come to this command after explain in this. So, all Neumann type boundary condition for pressure, pressure boundary conditions p at 2 to i max comma one equal to p 2 to i max comma two on the left side. Similarly for the right side wall, bottom wall, top wall as we explained in the previous slide; only thing it is return in mat lab code language. Now there is the procedure called pinning the pressure at a particular point. So, when we you are using Neumann boundary type of boundary condition for pressure it tenders the coefficient matrix singular. So, if when get a matrix as a singular, it is very difficult to solve. So, to avoid that situation and to make the matrix invertible pressure has to be pinned at a point that is we are actually interested in a problem only the pressure difference not the actual pressure itself. We use this so we say at any one point pressure is made to zero and all other pressures are referred with respect to that point and this process is called pinning the pressure.

So, for the matrix to invertible the pressure has to be pinned at a point, pinning the pressure at a point does not affect the overall solution, because the absolute pressure does not matter, what matter is only the pressure gradient. So, if you look at u momentum equation, v momentum equation, we have only dou p by dou x and dou p by dou y. So, all that it matter is the pressure gradient; actual pressure itself is not that much important, and this is used to set the procedure called pinning and that helps to make the coefficient matrix as invertible. So, pin the pressure at a corner, so we can pin pressure at any point, in this example, you have pin the pressure at one corner that is what is f at 1 equal to 0.0.

Now solve the pressure Poisson equation as shown here. We can again set convergence criteria separately for pressure Poisson equation and then return and end.

So, in this module, we have learned in detail how to solve to a pressure Poisson equation from the predicted velocity u star and v star. The pressure Poisson equation has a left side as well as right side term. We looked at left side term and right side term separately. We also learned a procedure called pinning the pressure, and pinning the pressure helps to make the matrix invertible then we also learn how to enforce boundary condition for pressure. For the pressure, we follow Neumann type of boundary condition and to impose the boundary condition, because we are following staggered grid arrangement we need to have a ghost node on all the sides for pressure. This is appropriately accounted and we looked at corresponding lines in the code. In the next module, we are going to see complete assembly of the code and solution obtains how to do pose processing from the solution corresponding display of the code.

Thank you.