

Computational Fluid Dynamics
Prof. Sreenivas Jayanti
Department of Computer Science and Engineering
Indian Institute of Technology, Madras

Lecture - 56
Methods For Unstructured Grid Generation

In the last lecture, we have seen the formulation of the finite volume method for an irregular shaped domain. The idea was to divide the domain into small elements which may be tetrahedral or pyramidal or hexahedral elements. And integrate the governing equation written in the form of fluxes where, you have the advection and diffusion terms are brought together as divergence of flux and we can evaluate this equation the governing equation by integrating over a particular cell.

(Refer Slide Time: 00:25)

Formulation of the Finite Volume Method

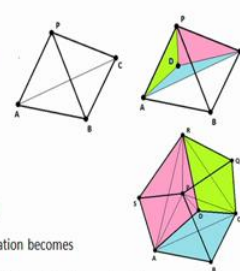
- General form of conservation equation:

$$\partial(\rho\phi)/\partial t + \nabla \cdot (\rho\mathbf{u}\phi) = \nabla \cdot (\Gamma_\phi \nabla \phi) + S_\phi$$
- In discrete form for cell,

$$[\partial(\rho\phi)/\partial t] V_{\text{cell}} + \Sigma(\mathbf{F} \cdot \mathbf{S})_i = S_\phi V_{\text{cell}}$$
- Evaluate it for cell j as

$$[(\rho\phi)^{n+1} - (\rho\phi)^n] V_j / \Delta t = (S_\phi)_j - \{ \Sigma(\mathbf{F} \cdot \mathbf{S})_i \}_j / V_{\text{cell}j}$$
- For cell j which is the tetrahedral PABC, the equation becomes

$$[(\rho\phi)^{n+1} - (\rho\phi)^n] V_j / \Delta t + (\mathbf{F} \cdot \mathbf{S})_{PAB} + (\mathbf{F} \cdot \mathbf{S})_{PBC} + (\mathbf{F} \cdot \mathbf{S})_{PCA} + (\mathbf{F} \cdot \mathbf{S})_{ABC} = (S_\phi)_j V_j$$
- Pyramid PABCD = tetrahedral PACD + tetrahedral PABC
- Hexhedral ABCDPQRS = Pyramids PABCD + PADRS + PCQRD
- The area and flux terms to be evaluated consistently so that the flux leaving cell j through a surface is equal to the flux that is entering a neighbouring cell through the same surface.
- Areas and volumes are evaluated using coordinates (x,y,z) of the vertices
- Fluxes are evaluated using the same neighbouring points



In the process we convert the divergence operator into a surface integration over each side of the surface. So, that we have an equation of this particular type rho phi j n plus 1 minus rho phi j n divide by delta t times volume of the cell, and the net outgoing flux through all the faces is equal to the source term and we made the point that we need to divide the control volume into small cells. We need to find the volumes in the areas and we saw how the volumes and areas could be determined from the coordinates of the

vertices. We also looked at evaluation of fluxes and how these could be evaluated consistently from the values of the cells which we are trying to determine and in that sense the formulation is complete except for generation of the points at which we would like to evaluate the solution. So, that is part of the grid generation and as part of the grid generation we would like to find the nodes of the vertices and all the information about it.

(Refer Slide Time: 01:34)

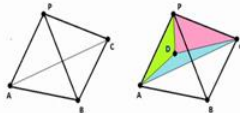
Evaluation of Areas

- Surface vector \mathbf{s}_{ABC} can be evaluated solely in terms of the coordinates of its vertices:

$$\mathbf{s}_{ABC} = \frac{1}{2} (\mathbf{x}_{AB} \times \mathbf{x}_{BC}) = \frac{1}{2} (\mathbf{x}_{BC} \times \mathbf{x}_{CA}) = \frac{1}{2} (\mathbf{x}_{CA} \times \mathbf{x}_{AB}) \text{ where } \mathbf{x}_{AB} = \mathbf{x}_B - \mathbf{x}_A$$
- The volume of the tetrahedral cell PABC can be obtained as


$$V_{PABC} = \frac{1}{6} \sum_{\text{faces}} (\mathbf{x} \cdot \mathbf{S})_{\text{faces}} = \frac{1}{6} [\mathbf{x}_{PA} \cdot (\mathbf{x}_{AB} \times \mathbf{x}_{BC})] = \frac{1}{6} [\mathbf{x}_{PA} \cdot (\mathbf{x}_{BC} \times \mathbf{x}_{CA})]$$

or
$$V_{PABC} = \frac{1}{6} \begin{vmatrix} x_P & y_P & z_P & 1 \\ x_A & y_A & z_A & 1 \\ x_B & y_B & z_B & 1 \\ x_C & y_C & z_C & 1 \end{vmatrix}$$



- Area of quadrilateral ABCD evaluated as sum of areas of triangles ABC and ADC:

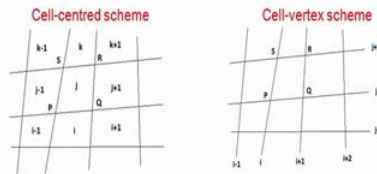
$$S_{ABCD} = \frac{1}{2} [(\mathbf{x}_{AB} \times \mathbf{x}_{BC}) + (\mathbf{x}_{CD} \times \mathbf{x}_{DA})]$$



(Refer Slide Time: 01:42)

Evaluation of Fluxes

- The fluxes vary locally with position and are usually a function of ϕ .
- Two possibilities depending on where the solution variable is evaluated:



- In a cell-centred scheme, the value of ϕ is associated with a cell, for example, ϕ_j and ϕ_k and fluxes across common faces, e.g., face RS, is evaluated using ϕ_j and ϕ_k .
- In a cell-vertex scheme, for face RS uniquely determined by its vertices R and S, the flux through the face can be evaluated as

$$F_{RS} = \frac{1}{2} [F(\phi_R) + F(\phi_S)] \quad \text{or also as} \quad F_{RS} = F[(\phi_R + \phi_S)/2]$$

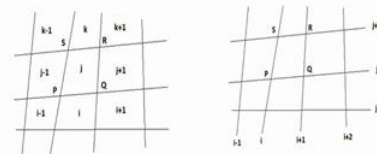


(Refer Slide Time: 01:56)

Evaluation of Diffusive and Convective Fluxes

- Evaluation of the diffusive flux requires the determination of the gradient. In a cell-centred method, the gradient can be estimated to second order accuracy using ϕ -values of the two cells on either side of the face, for example, cell j and cell k for the side RS:

$$[F_{diff} \cdot \mathbf{S}]_{RS} = [-D_{eff} \nabla \phi \cdot \mathbf{S}]_{RS} = -D_{eff}(\phi_j - \phi_k)/(x_j - x_k) S_{RSx} - D_{eff}(\phi_j - \phi_k)/(y_j - y_k) S_{RSy}$$



- The convective flux is usually evaluated using an upwind scheme; thus,

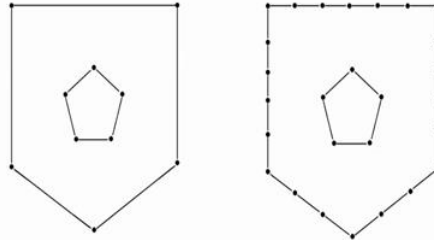
$$[F_{conv} \cdot \mathbf{S}]_{RS} = \begin{cases} [\rho \mathbf{u} \phi_j \cdot \mathbf{S}]_{RS} & \text{if } (\mathbf{u} \cdot \mathbf{S}) > 0 \\ [\rho \mathbf{u} \phi_k \cdot \mathbf{S}]_{RS} & \text{if } (\mathbf{u} \cdot \mathbf{S}) < 0 \end{cases}$$



(Refer Slide Time: 02:18)

Unstructured Grid Generation

- Need to divide computational domain into large number of cells in such a way that they
 - Do not overlap
 - Do not leave any "holes", i.e., all the surface is covered and the whole volume is filled
 - Cells are of similar size or in a distribution of our choice, e.g., small cells in areas of large gradients
- Grid generation is non-trivial for generic shapes and one would have to have a generic approach to grid generation, not something that will work only for certain shapes



So, we are going to look at methods for unstructured grid generation. When you look at grid generation, we need to have the basic theme incorporated in a procedure which is to divide the computational domain into large number of small cells. This is to be done in such a way that the cells do not overlapped, they do not leave out any holes that is all the bounding surface should be covered without any gaps. At the same time they should not be any overlaps and the whole volume needs to be filled by the cells that we are chosen. So, from the individual cells we should be able to reconstruct the entire bounding surface and the entire volume through which the flow is flowing. So, the computational domain must be recreatable from the cells that we subdivide into and the cell should be roughly of similar size. We cannot have very large cells and very small cells in an arbitrary way and if they are not of the similar size they should be in a distribution of our choice.

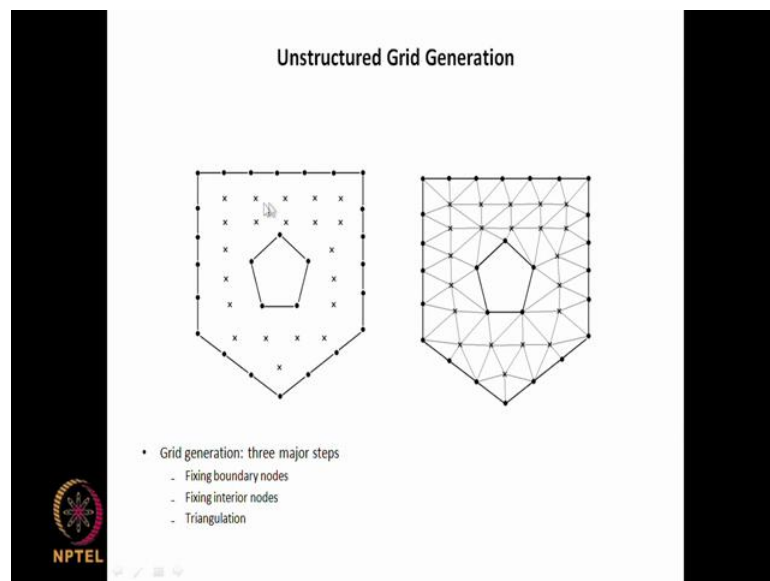
For example, small sense in areas of large gradients of the variable that we are looking for is a very desirable feature of CFD. Because if you want to resolve a large gradient a steep gradient you need have large number of grid points because a truncation error is proportional to Δx or Δx square times the gradient value. So, if the gradient is large the truncation error will be large. So, you would like to reduce the Δx there, as to reduce the overall truncation error.

So, we would like to have good spacing, good cells at the same time when we are looking at large gradients we would like to resolve the flow better. So, we would like to

have the small only small cells in those regions. So, in that sense all these factors must be considered when we generate grid. Generation of a grid for a fairly simple geometry that can be fit into cartesian coordinate system is a not so very difficult, it can be done intuitively. It can be done by inspection but when we have a more complicated domain, then we need to we cannot do it in an arbitrary way and we have to do it algorithmically. So, grid generation is non-trivial or generic shapes. And one would have to have a generic approach for grid generation not something that would work only for certain number of shapes, certain types of shapes. So, just to see what the issues are in terms of grid generation the major issues.

Consider this domain here that we have which has these 5 surfaces. It is like a pentagon not quite regular pentagon. Inside which you have another pentagon like this. And whatever the domain is in between is a computational domain. So, it is this looks very regular to me, but it is a way non trivial kind of thing and it obviously, cannot these surfaces here cannot be represented in simple cartesian meshes. So, if you want to generate domain like this a grid for this it is a 2-D shape and in such a way that we break it up into small number of triangular cells. Which when put everything put together will make up the entire shape without leaving any holes, then we need to first of all see the where we would like to have the boundary nodes.

(Refer Slide Time: 06:46)

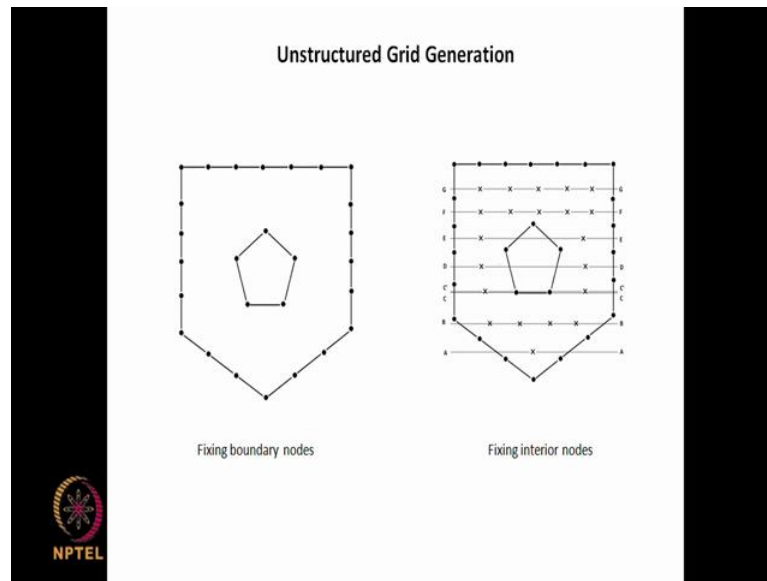


For example, we can put boundary nodes here, overall the boundaries and may be on the interior also. And then we can try to put some internal nodes like this. Then we can join them to generate a grid like this. So, we are going for a given shape like this. So, we are going through these steps here to finally, end up with something like this. We have got a triangulated mesh and this whole domain is now decomposed into smaller triangular elements and all of this put together will make up the entire shape. We will fill the entire control volume and every part of the boundary here. Every part of this boundary and this boundary is part of is the side of 1 of these triangles and the whole of that boundary domain is put together in this. So, we would like to go from shape like this or computational domain like this into shape triangulation like this.

Now this is a very coarse grid and we really cannot use it for CFD kind of solutions with any kind of accuracy, but the idea is to go from there to here. Here is where, we have a grid which meets those kind of considerations that all the boundaries are part of this grids and the whole boundaries there and all the computational domain is included as the area of the individual cells. The some of the areas of all the triangle triangular cells is equal to the some of the area of the computational domain.

So, when you want to go from just a boundary into triangulated mesh like this then, we would have three major steps to reach up to this particular point. First is fixing the boundary nodes, and second is fixing the interior nodes and then third is triangulation ok. So, that is joining these individual things to make up a triangle. We may look at these as fairly simple to do, but one would have to take a lot of care in making sure the for example, all these points are well distributed. So, what will do in this lecture? Given that grid generation is nontrivial. We look at how to go from this domain shape into the stage where we can put the boundary nodes and the interior nodes and then we look at triangulation in the next step because there is that is definitely nontrivial exercise.

(Refer Slide Time: 09:44)



So, we are looking at fixing the boundary nodes and fixing the interior nodes. So, here what we have is a domain and we also know the boundary condition. We also know the equation that needs to be solved, but how to fix a grid for this, that depends on number of factors it depends on how at what stage of solution you are. If you are making a first cut solution first time you are solving a problem maybe it is a good idea to have a fairly course grid just to see that what kind of solution you get and then where you think are what kind of flow is taking place is this the correct way to formulate the problem have you got all the boundary conditions, have you got all the information that is needed for the solution.

So, you need to fix all those things and then may be look for a final solution. So, it would be a good idea to start with course grid and then you can refine it further and further. So, when you are looking at course grid what how course it is. We are looking for a CFD solution, we are looking at solution of non-linear equations and highly coupled equations. So, we would like to have good accuracy, it cannot be few grid points. So, that we can get a quick solution and so, that is where we would like to see what kind of flow scales we want to resolve. So, that is one particular constant, if you are looking at certain if you already are expecting certain features as a flow and these need to be resolved then if there is a certain region in which you are expecting say large velocity change may be

50 percent velocity change then so that means, that as you when you look at velocity change you are saying that at this point a velocities is here and at this point velocities here. You would like this velocity variation to be resolved. So, that is you would like to have few grid points may be three four grid points here. So, that we can say velocity at this point is here, the velocity at this point is here, the velocity at this point is here and when you join them you get this straight line.

So, the most steep the gradient is the smaller will be the grid point, the more regular the variation is, the few of the number of grid points. If you have a linear variation then you need only 2 points. This point the velocity at this point and the velocity at this point to make up the stream the straight line, but how do you know in somewhere in the interior you have a linear variation. You don't know. You would like it to come out of the solutions as part of the solution. So, you should have sufficient number of grid points to resolve velocity profiles that are going to be generated by resolution. So, you should want to have decent number of grid points.

So, based on for example, if this is the flow domain shape and if I am expecting some flow file here, then I would like to have maybe ten points in between. So, if I am looking at 10 points based on that I can say that I would like to have a grids spacing of so much. it is not possible in the general unstructured mesh to fix uniform grids spacing which will work for all the cells, but they should be an estimate of the size of the cell. In this case we are looking at a triangular cell. What will be the side of the triangle side length of the triangle that you are looking at? So, that can be beginning basis to fix your boundary points.

Let us say that in this particular case I have taken large size so as to make the solution clear, but this much variation this much Δx is not such a good thing. So, let us say that this is a size that you want to have. You start at some point in the boundary, you move along the boundary and then you move that much distance say small l and then you fix the point here. You move another small l is tends you come here and then you can keep on putting this markers along the boundary and say that this is where you want to have the cell. That is one possibility. For more complicated things it may become you may need to think more imaginatively as to how you can fix this.

For example, you can draw lines separated by that measure distance and find out the intersection of these horizontal lines with the boundary points and along that we can put. So, you can find out the distance from here along the boundary. So, in that way you can fix certain you can fix the boundary points. So, you fix the boundary points on all of these. So now, how do you fix the interior points? So, the interior points should be fixed in such a way that there are points all over the place. So, that is over the entire computational domain we should have those points and this points cannot be in half hazard location, they should be roughly of equal distance from each other. Obviously, they should not be very close to the surfaces to the boundary surfaces because if you had a point here and then if you joined it by like this to make a triangle, then you would have a highly distorted triangle and something like that is not very desirable.

So, you would like to have method of placing the interior nodes in such a way that the distributed all over the computational domain and that there is sufficient gap between the points and between the points and the boundary.

So, with these kind of things it is possible to come up with simple way of doing this nodalisation that is fixing the interior nodes. So, what we can do is at for a shape like this, you know the total maximum height and minimum height. And you have some characteristic distance that you would like to have the small l that you would like to have between the grid points which can be the estimated size of there of an equilateral triangle that you would like your triangular cell to be. So, with that thing and starting with the bottom most point, you draw horizontal lines which are dis separated by the distance l . So, you have line a, b, b, c, c and d, d, e, f, f, g, g. All these things are these kind of a horizontal lines which are roughly spread as set a part by distance small l .

Now given this line you can find out what is the where this line is intersecting the boundary. For example, this line is a intersecting the boundary at this point and at this point. Line b, b is intersecting the boundary here and here. So, if you know the boundary equation, then if you know this equation and you can find out where the point of intersection is. So, once you do that then you are now in a position to see. What is the length between the intersections, which is part of the competition domain?

For example, if I take this line here, this part from here to here is inside the computational domain and this part is outside the computational domain. So, I can find out from this intersection what is the length of this line which lies inside the computational domain? Now along this line I fix as many number of grid points as I need to have in order to have a separation distance of about one. For example, here along this line is this length short may be I will have just 2 points here, 2 intervals here. Along this line I have wider width. So, I can put one distance, 1 second distance 1, third distance 1, fourth distance 1 and fifth distance 1. So, along this I can put four points. If I come to line d , it is intersecting the boundary at 0.1, 0.2, 0.3, 0.4 at 4 points. Out of this, when we have an even number of intersections like this that means that if you have one internal space, which is not part of the domain.

Here you have only this part is part of the computational domain and this part is part of the computational domain. You do not want to put any grid points outside the computational domain because if you do that if you have a grid point it has to be connected to other grid points make a triangle. So, if you have a grid point inside here and then if I join it like this to make a triangle, then part of the triangulated domain lies outside the computational domain and you will be violet in the principle that when you put all the rectangular all the triangular together, you should get exactly the area of the computational domain. We will now you will be getting a bit higher. So, you have to make sure that you have do not have any points outside this computational domain. So, you say that I take this length and then I put to 1 point here to make into 2 equal sides of roughly 1 and here also I do this. So, I can continue like this, but if I look at point c here, then this is passing through this almost passing through like this and so, in such a case I can move it up or down slightly and then put the points here.

So, in a case where I feel that the points are very close to existing points, then I can move it I can move points up or down because I know that it is not possible to get exactly the same length 1 all the time. It is ok to have sometimes may be to have 0.7 1 distance or 1.3 distance 1 distance. So, accordingly we have to move the interior points around after using this general guideline to place them inside.

So, this is one simple way for a 2 d case of fixing this kind of domain. You can also extend this logic to three d and then you can do this. So, in that sense nodalisation which is the task of putting bound, putting points inside the computational domain, so that all points are inside the computational domain and the spread throughout the computational domain requires some estimation of the distance between the points that you would like to have. Once you have that kind of distance then and once you have a representation of the boundaries in the boundary nodes, then we can go through this kind of exercise to get all the internal points. So, once you have all the internal points, you have got something like this.

Now the next task is to join these to form triangles. Now joining this it looks like a very trivial think to do, but if you have to do it with your eyes closed. If you have to if you know only the x y coordinates of this, and then which one to join with what is something that needs to be determined.

For example; if you joined here like this and then, if you join this one like this then you are overlapping, if you join this and if you join this, but don't join these 2 then you have some area which is ah left out and if by chance you join this and this and then if you join this and this there is again going to some overlap. So, in that sense there is possibility of overlap, there is a possibility of living out certain part of the domain. And there is also the desirable feature that if you are making into triangles then you would like the triangles to be as equilateral possible. We do not want have 2 distorted triangles. So, these are the kind of constants that are present when you are doing triangulation, will look at triangulation in the next lecture.

So, what we have done in this lecture is to understand what is there, what is the kind of issues that we have to face when you are doing unstructured grid generation. Given the fact that, you have domain which does not fit into regular constant x, constant y kind of things as shown here for this simple case, and given that they may not be any regularity in terms of this, you need to have some ah imaginative way of trying to put interior points throughout the domain in such a way that they are separated roughly by a distance a desirable distance a part and that there everywhere and only inside the computational domain and not external. We also need to make sure that the entire boundary is

completely divide into segments and these segments are also such that there is a desirable distance between one segment and the next segment. So, with these kind of ideas we can do nodalization and triangulation is something that will see in the next lecture.