Discretisation & Grids, part 2

Rowan Gollan

August 19, 2019

The University of Queensland

Numerical discretisation

Having chosen a (continuous) mathematical model to represent the flow of interest, CFD involves formulating and solving a discrete approximation to the continous equations. The discretisation is in two (related) parts:

1. spatial discretisation $\rightarrow$ **meshes and grids**
2. equation discretisation $\rightarrow$ **numerical method**

All numerical methods involve the transformation of the mathematical model into an algebraic system of equations for mesh-related unknown quantities.

In this lecture...

- Equation discretisation
  - Finite-difference method (FDM)
  - A refresher on truncation error
  - Finite-volume method (FVM)
  - Truncation error for FVM
- The connection between truncation error and grids
- Gridding for CFD
  - Survey of structured grids
  - Survey of unstructured grids
  - Recommendations on grid quality

Equation discretisation

finite difference method

- replace the continuous derivatives with a finite difference estimate
- Euler first did this in 1768 for differential equations, in general
- In fluid dynamics, Richardson attempted weather predictions in the 1920s with methods that are considered the modern beginnings of CFD[1]
- limited to structured grids

finite volume method

- replace the continuous integrals with discrete summations
- most widely used method in CFD today
- used in most industrial CFD codes

finite element method

- dominant in solid mechanics, but less popular in fluid mechanics
- used to build higher-order methods (order 3 and higher)
- gaining popularity, but by no means standard in industry

[1] but without a modern computer

Finite difference method overview

We will (briefly) study the basis of finite difference methods because:

- Historically, CFD started with finite difference methods
- The theoretical ideas of truncation error are more easily explained in a finite difference context but have direct carry-over to finite volumes
- Finite differences are still in use today. They are used for highly efficient, high accuracy calculations on relatively simple geometries. This means they are used in a lot of direct numerical simulation (DNS) calculations.

Finite difference method

Consider an equation in one dimension $u(x)$, the derivative is defined as

$$\frac{\partial u}{\partial x} = \lim_{\Delta x \to 0} \frac{u(x + \Delta x) - u(x)}{\Delta x} \tag{1}$$

Removing the limit gives a finite difference approximation for the derivative.

When Δx is small but finite, we get an approximation to the true value of u_x. This approximation is improved when we reduce the size of Δx. How much that approximation is improved is linked to the **truncation error**. The truncation error should go to zero as Δx tends to zero.

Finite differences and Taylor series expansions

Consider the Taylor expansion around $u(x)$:

$$u(x+\Delta x) = u(x) + \Delta x \frac{\partial u}{\partial x} + \frac{\Delta x^2}{2}\frac{\partial^2 u}{\partial x^2} + \frac{\Delta x^3}{3!}\frac{\partial^3 u}{\partial x^3} + \dots \tag{2}$$

Rewrite as:

$$\frac{u(x+\Delta x) - u(x)}{\Delta x} = u_x(x) + \underbrace{\frac{\Delta x}{2}u_{xx}(x) + \frac{\Delta x^2}{6}u_{xxx}(x) + \dots}_{\text{truncation error}} \tag{3}$$

- RHS of Eq. 1 is indeed an approximation to u_x
- The remaining terms in the RHS represent the error when using this finite difference approximation

Truncation error

Consider the dominant term of the truncation error

$$\frac{u(x+\Delta x)-u(x)}{\Delta x} \approx u_x(x) + \frac{\Delta x}{2} u_{xx}(x) = u_x(x) + O(\Delta x) \qquad (4)$$

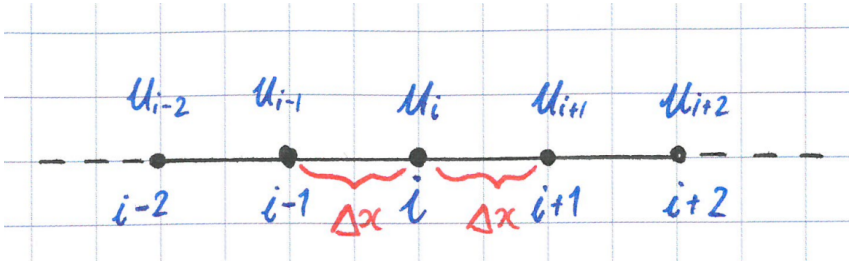
We use the notation $O(\Delta x)$ to represent a truncation error of first order: the error goes to zero like the first power of x.

Other approximations for the first derivative are possible...

central difference

$$\frac{\partial u}{\partial x} \approx \frac{u(x+\Delta x)-u(x-\Delta x)}{2\Delta x} + O(\Delta x^2) \qquad (5)$$

Truncation error and grid points



$$(u_x)_i \approx \frac{u_{i+1} - u_i}{\Delta x} + O(\Delta x)$$

$$(u_x)_i \approx \frac{u_{i+1} - u_{i-1}}{\Delta x} + O(\Delta x^2)$$

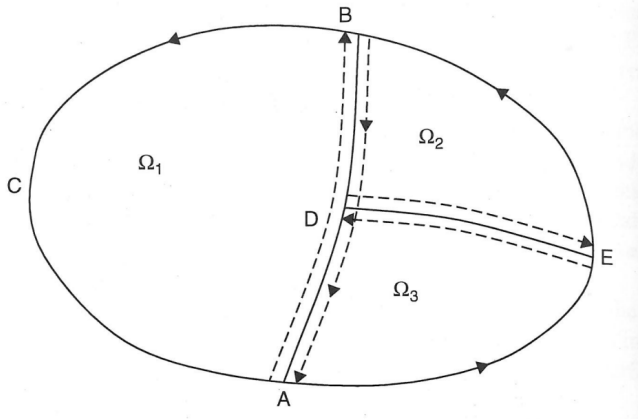
Thought experiment: Consider a domain on the x-axis from 0 to 1 with $\Delta x = 0.1$. There are 11 grid points. A first order method is used on those 11 grid points that has an error of 10%. A second order method is also used on that same grid. Its error is 1%. How many grid points are needed so that the first order method reduces to the same error as the second order method on 11 points?

Lessons from finite-difference method

- There are many (in fact, infinite) ways to approximate the continuous derivatives with finite differences
- The truncation error is tied to the choice of approximation
- Higher-order approximations require information from more grid points (larger stencil) so in practice we use relatively low order approximations
- Still, despite that, there is a big advantage in using a second order method over a first order method in terms of computational efficiency

Finite volume method overview

$$\frac{\partial}{\partial t}\int_{\Omega} U d\Omega + \oint_{S} \vec{F}\cdot d\vec{S} = \int_{\Omega} Q d\Omega$$



Summation of individual control volumes within larger domain

$$\frac{\partial}{\partial t}\int_{\Omega_1} U d\Omega + \oint_{ABCA} \vec{F}\cdot d\vec{S} = \int_{\Omega_1} Q d\Omega$$

$$\frac{\partial}{\partial t}\int_{\Omega_2} U d\Omega + \oint_{DEBD} \vec{F}\cdot d\vec{S} = \int_{\Omega_2} Q d\Omega$$

$$\frac{\partial}{\partial t}\int_{\Omega_3} U d\Omega + \oint_{AEDA} \vec{F}\cdot d\vec{S} = \int_{\Omega_3} Q d\Omega$$

Basics of finite-volume method

We use direct discretisation of the integral form of the consevation laws

1. Subdivide the domain into finite (small) volumes
2. Apply the integral conservation laws to each of these finite volumes

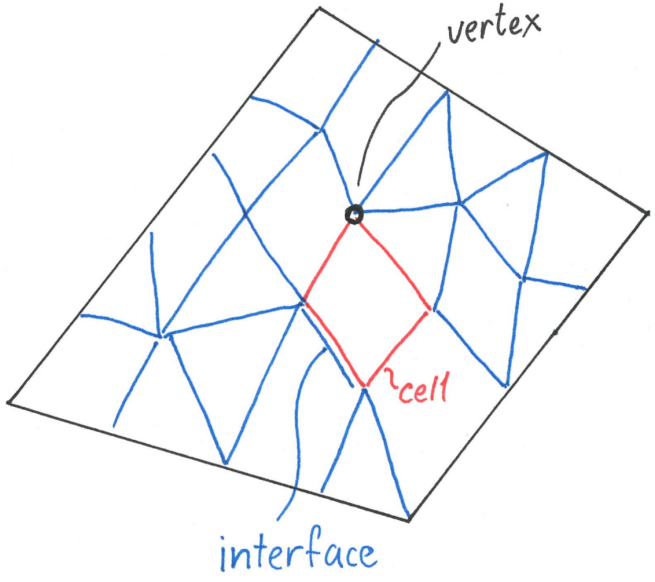
The key idea is that it should be "easy" to apply the conservation laws to these individual finite volumes, as compared to the domain as a whole.

$$\frac{\partial}{\partial t}\int_{\Omega} U d\Omega + \oint_{S} \vec{F}\cdot d\vec{S} = \int_{\Omega} Q d\Omega$$

when discretised becomes

$$\frac{\partial}{\partial t}(U_j\Omega_j) + \sum_{\text{faces}} \vec{F}\cdot\Delta\vec{S} = Q_j\Omega_j$$

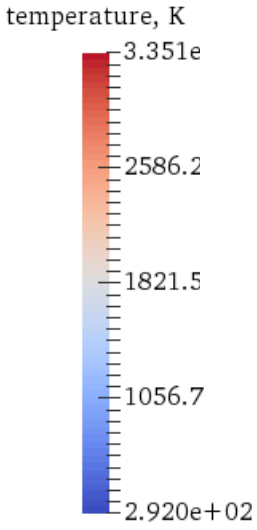
Discretization in space via finite volumes

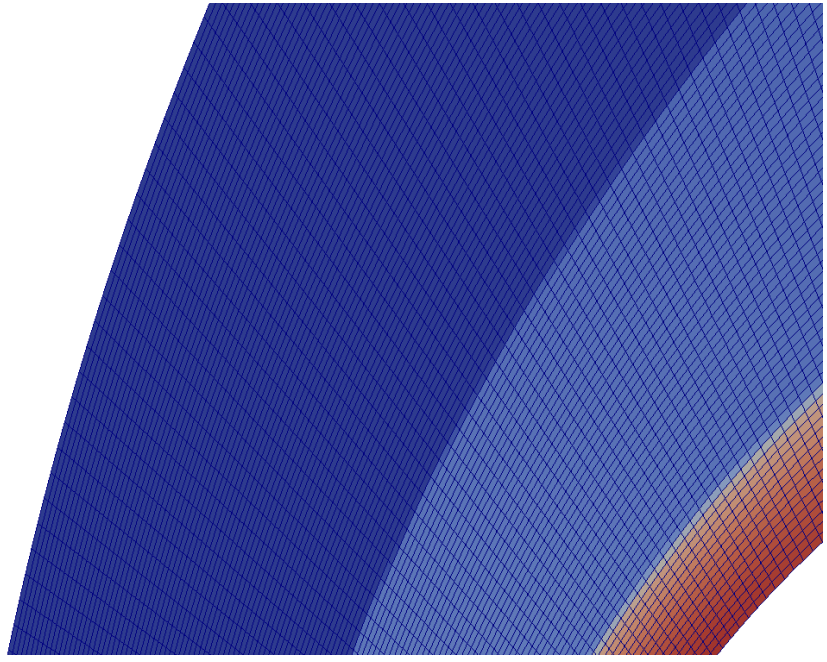


Some nomenclature

cells:	the finite volumes used to discretise the domain; the flow solution is computed as average quantities in the cells
interfaces:	the edges (or faces, in 3D) of cells
vertices:	points in space forming the corners of cells

A flow domain with discretisation





Spatial discretisation: semi-discretised equations

$$\frac{dU}{dt} = -\frac{1}{\Omega} \sum_{\text{cell-surface}} \vec{F} \cdot \hat{n}\, dA + Q \tag{6}$$

- The right-hand side (RHS) is now discretised — it forms a set of algebraic expressions.
- If our finite volumes completely fill the domain *(they should!)*, we see that this discretisation is inherently conservative: flux out of one cell is balanced as a flux in to an adjoining cell.
- The complete system is a system of ordinary differential equations.
- This system can be integrated forward in time using numerical methods for ODEs. We typically stick to low-order time methods because of the complexity and computer memory required by evaluation of the RHS.
- More details on how we compute $\sum_{\text{cell-surface}} \vec{F} \cdot \hat{n}\, dA$ are presented later in the course for incompressible and compressible flow, separately.

FVM and truncation error

On Cartesian grids, one can rewrite the FVM as FDM. Since we can determine the truncation error for FDM, the truncation error can be shown for this special case of FVM.

FVM on uniform Cartesian grids has second order truncation error provided we approximate the numerical fluxes to second order.

On non-uniform grids, the analysis of truncation error is more complicated. However, **the truncation error approaches second order** on good quality grids.

So we need to discuss what is a good quality grid.

Grids

Head of CFD Division at Boeing said (circa 2014):

> *Gridding in CFD is a first-oder error.*

More context...

> *I can give two CFD engineers the same geometry, the same flow conditions and they report back with completely different results. It almost always comes back to the differences in the grids that they generated.*

Grid types

Structured grids

> *families of intersecting lines, one family for each space dimension, where each mesh point is located at the intersection of one line, and only one line, of each family*

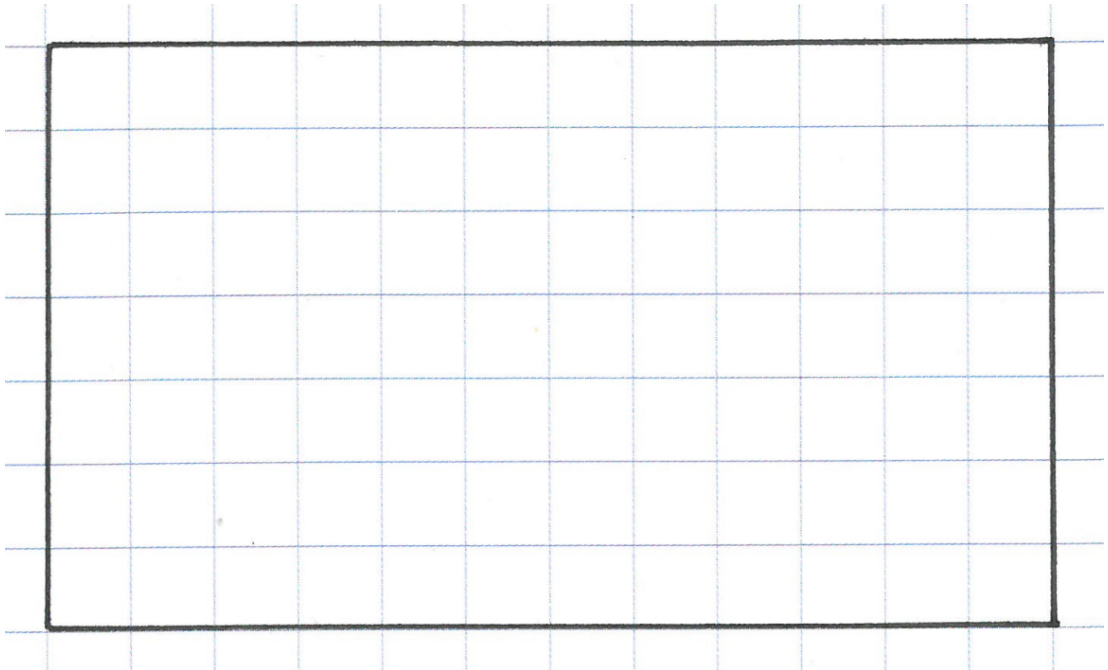
- Cartesian grids
- Body-fitted grids
- Multi-block grids

Unstructured grids

Hybrid grids

Cartesian grids

Cartesian grids with regular cell sizes are the ideal grid in terms of accuracy. The points are all equidistant and the discretised formulas have their highest possible accuracy on these grids.

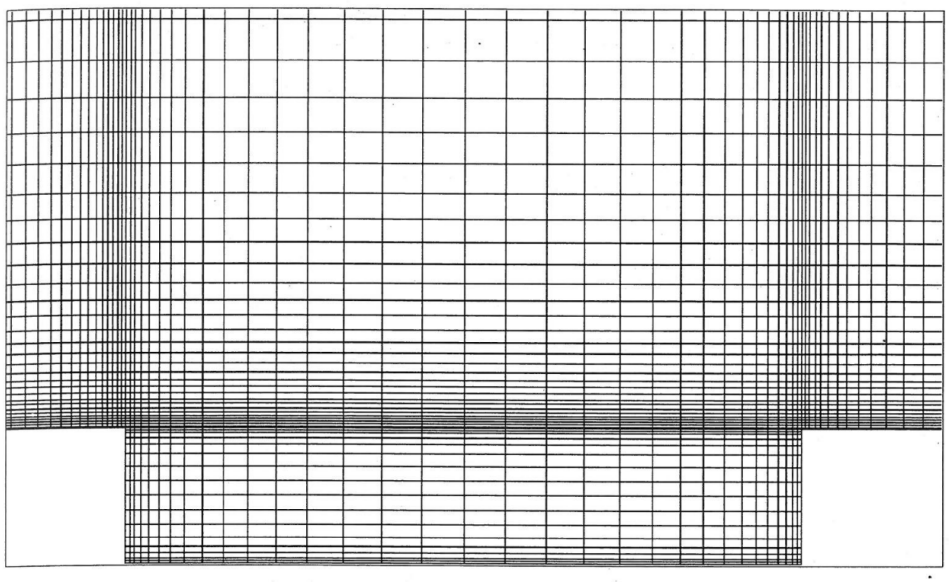


These are a good option when walls are parallel to Cartesian axes, or when there are no walls at all (simulating free space).

Non-uniform Cartesian Grids

Regular Cartesian grids are very limiting in terms of geometry and can be be wasteful due to uniform cell size.

Variable mesh size A variant of the regular Cartesian grid allows for variable values of mesh spacing.



A common terminology is that we have *clustered* cells towards certain boundaries.

Diversion: clustering in a boundary layer

In laminar boundary layers, the ratio of velocity gradients is

$$\frac{\partial u}{\partial y} \Big/ \frac{\partial u}{\partial x} \sim \sqrt{\mathrm{Re}_x}$$

Suppose we want velocity variations of the same order over mesh distances Δx and Δy, then we need a cell aspect ratio of

$$\frac{\Delta y}{\Delta x} \sim \frac{1}{\sqrt{\mathrm{Re}_x}}$$

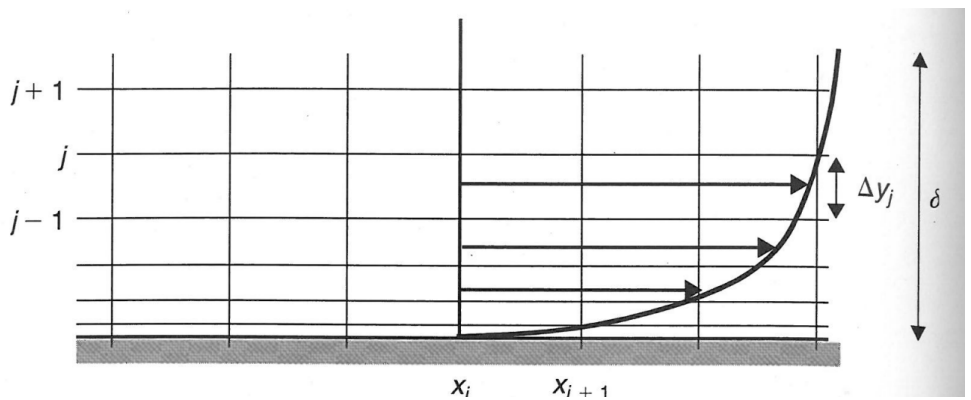
Thought experiment: Re = 1 million: the ratio between Δx and Δy is of the order of 1000! If we consider a plate of unit length and 100 cells in the x-direction with $\Delta x = 0.01$, what size should Δy be to keep velocity variations of the same order in the y-direction?

Diversion: clustering in boundary layers

From the previous example, we would require $\Delta y \sim 10^{-5}$. If we were to use a uniform Cartesian mesh, this would lead to millions of mesh points!

Instead, we should use meshes with varying cell size for efficiency reasons.

There is an extra lesson here: even in the *y*-direction, we should not use a uniform $\Delta y = 10^{-5}$ because we know that the strong gradients in the *y*-direction reduce progessively as move out of the boundary layer. Thus a non-uniform distribution in *y* is recommended. We call distributing points in a non-uniform manner *clustering*.

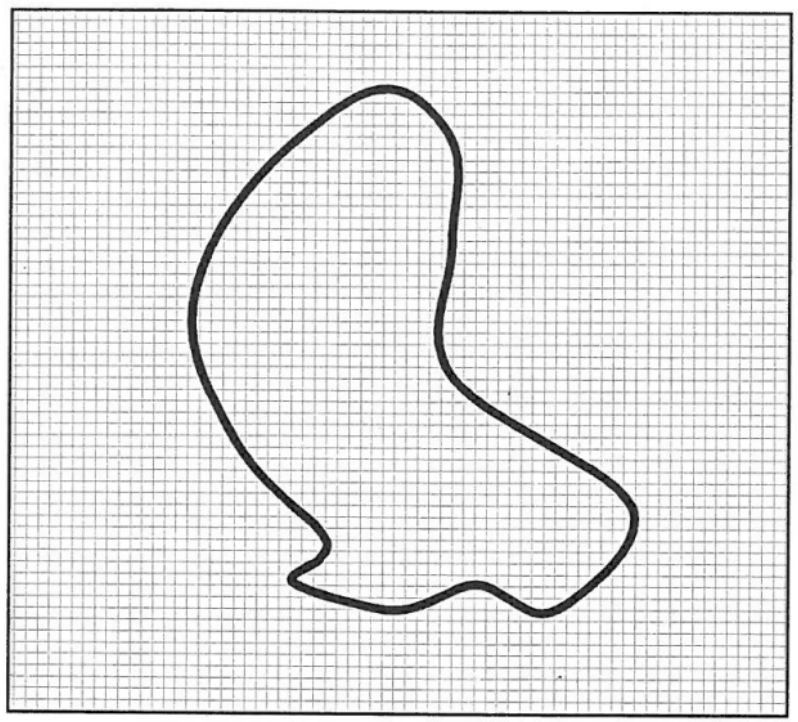


Note: we should choose an analytical means to distribute the cell sizes smoothly. This is what clustering functions attempt to do.

Non-uniform Cartesian grids: immersed boundary method

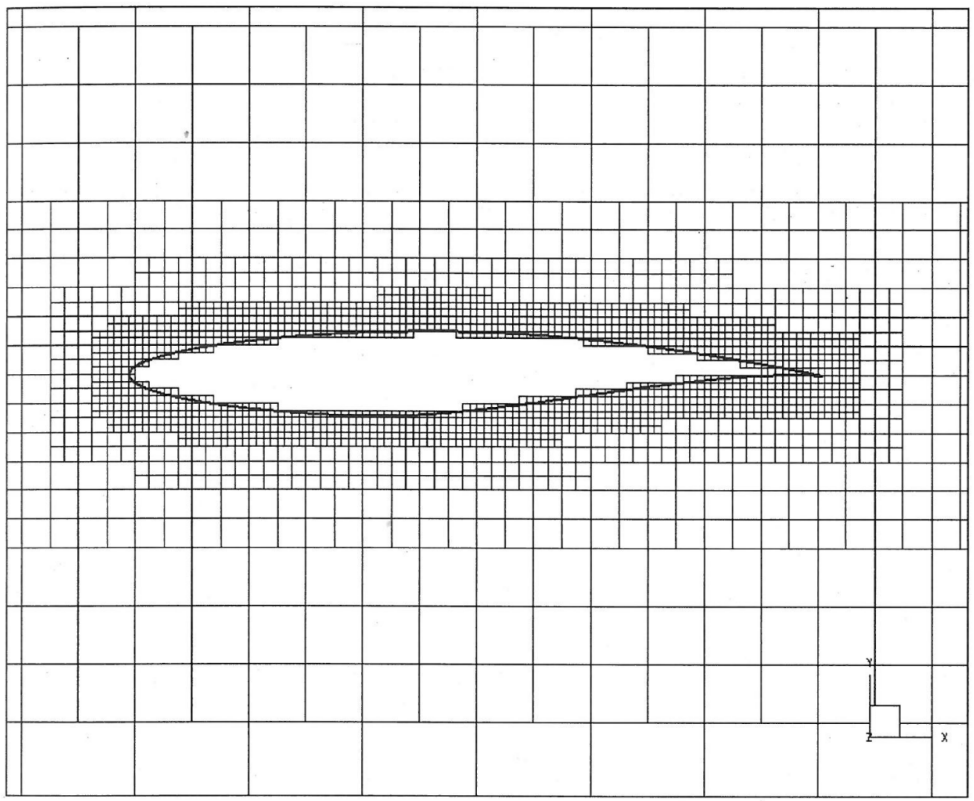
Curved boundaries are a problem for Cartesian grids. Cartesian grids are easy to generate, but we need then a special treatment at curved boundaries.

Cartesian grid used on both sides of solid surface. A special numerical treatment is used to handle the physical boundary condition.



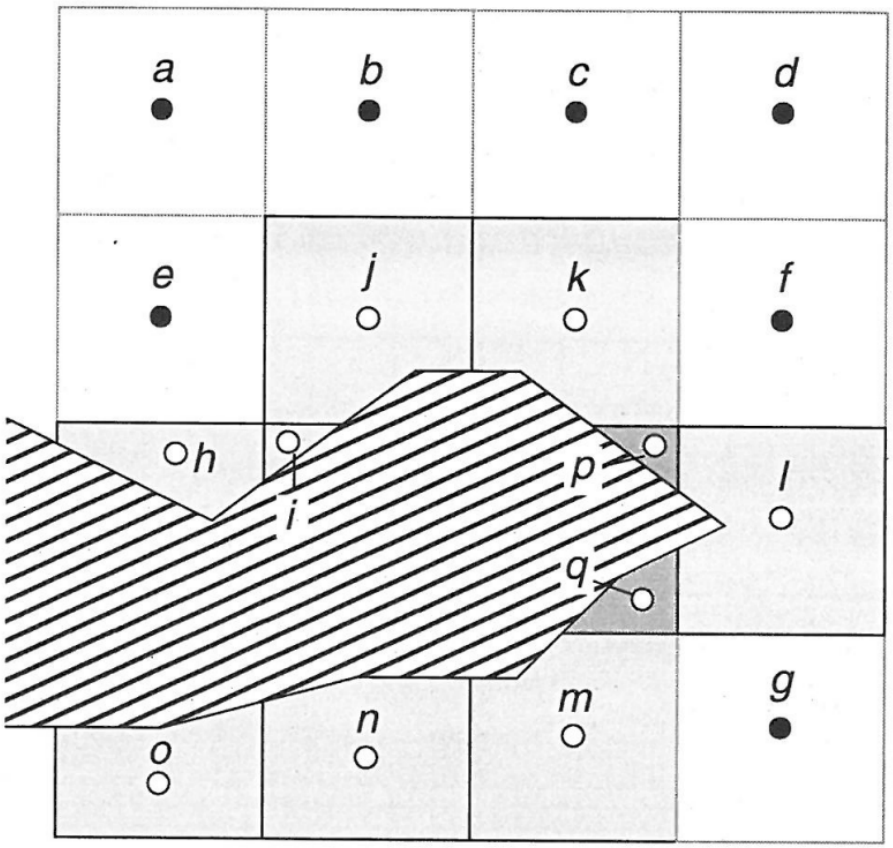
Non-uniform Cartesian grids: staircase method

Use local refinement near curved surface, but all cells are still regular, just subdivided from the Cartesian grid. Then remove cells inside the solid domain.



Non-uniform Cartesian grids: cutt-cell method

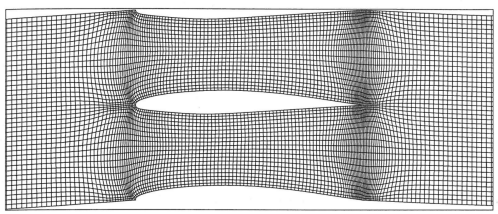
At the intersection of the solid boundary and the Cartesian grid, the cells are cut into arbitrary (non-Cartesian) shapes. Then apply a special form of the finite volume discretisation on the cut cells.



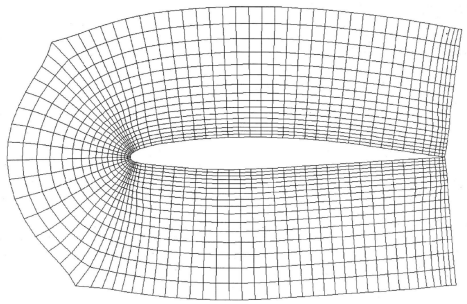
Cartesian mesh with embedded geometry

Body-fitted grids

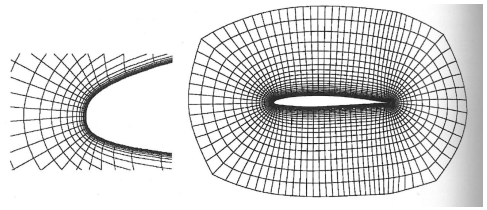
Grid lines become curvilinear to adapt to the shape of the body.



H grid



C grid



O grid

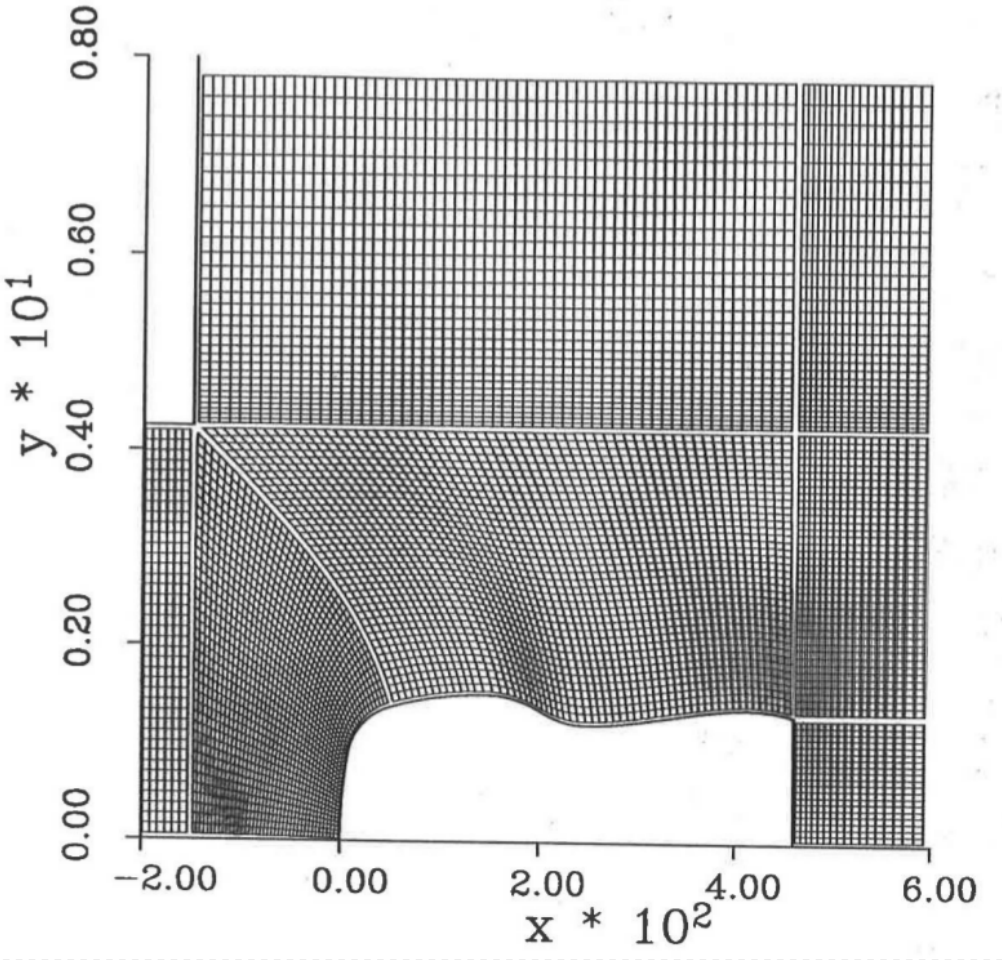
Note: Special emphasis is given in the tutorial on generating body-fitted grids in 2D since these will be used in OpenFOAM and Eilmer flow solvers.

Multi-block grids

Multi-block grids allow us greater flexibility in terms of topology and controlling mesh sizing in different parts of the domain.

There are two types:

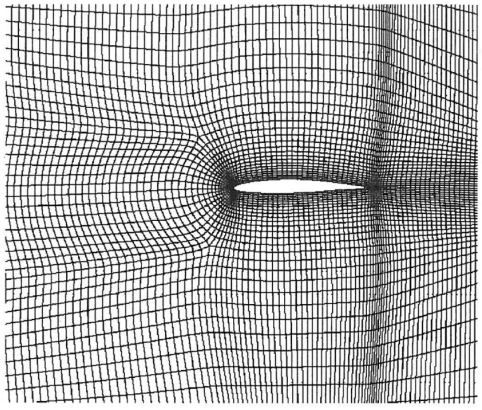
1. matching boundaries
2. non-matching boundaries



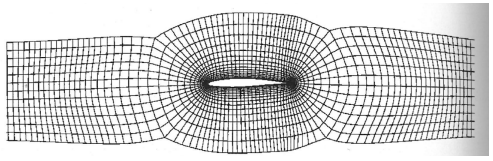
Note: Almost all of the grids you will be asked to build in this course will be multi-block, matching boundary grids.

Some common topologies with multi-block grids

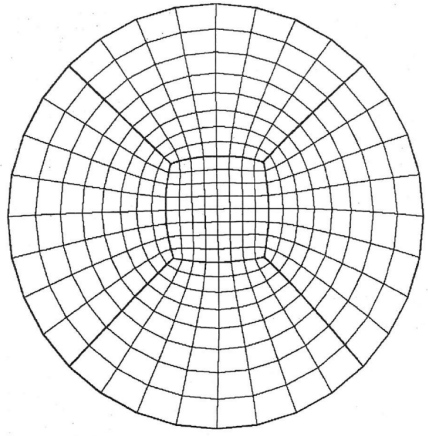
We use the term **topology** to refer to the arrangement of the blocks. There can be many valid topologies for a given a geometry, but some lead to better quality grids than others.



C-H grid

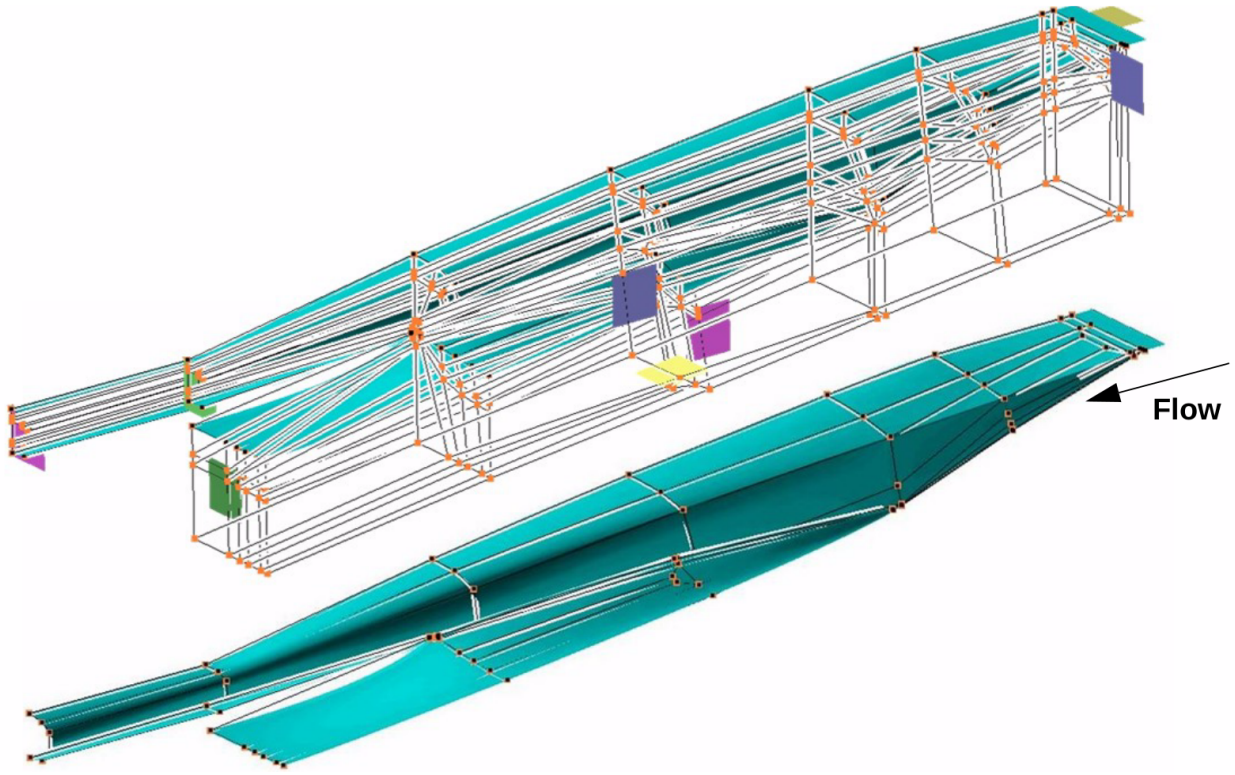


H-O-H grid

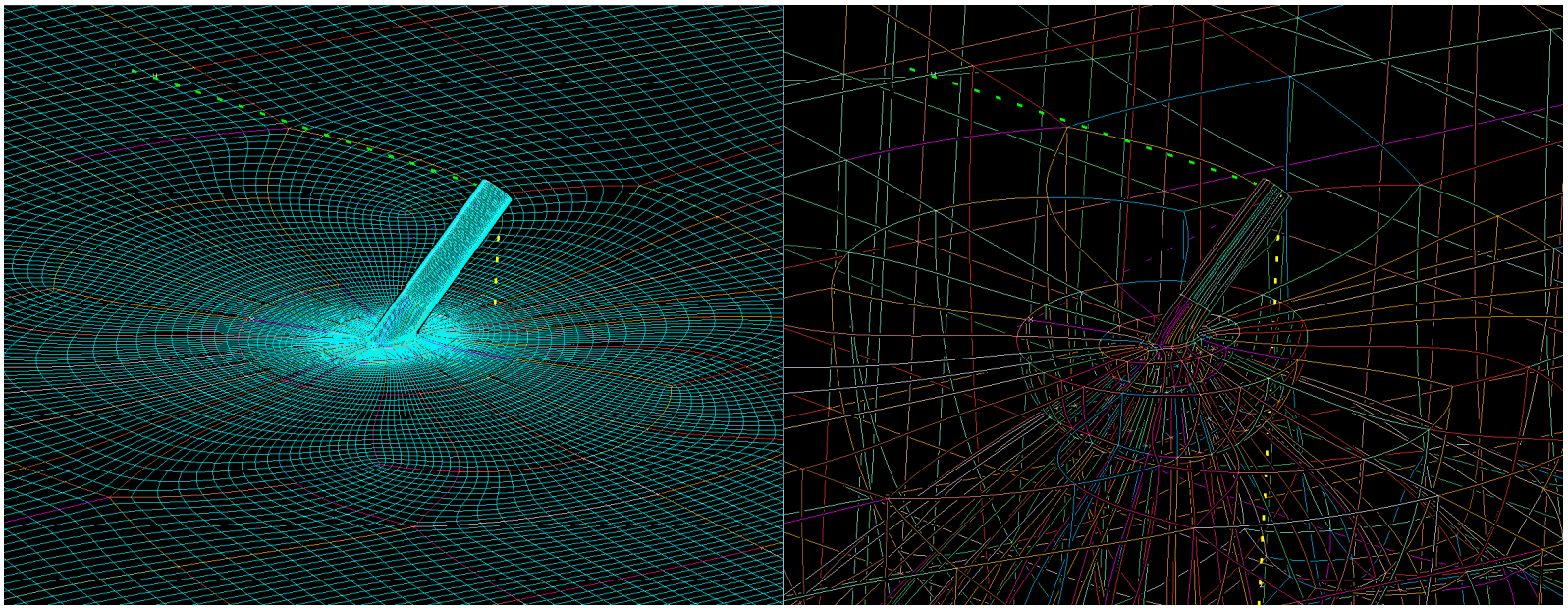


‘butterfly’ grid

Multi-block grids can get complicated



… very complicated



…but can take a long time to generate (on the order of weeks). This cost might be prohibitive in an industrial setting.

Hence, unstructured grids are popular industry because they can be generated rapidly in an automatic manner.

Unstructured grids

Unlike structured grids, there is no family of mesh lines to define cell edges. Cells can be arbitrary shaped polygons (in 2D) and polyhedra (in 3D).

These arbitrary shapes lend themselves to being automatically placed in complex geometries. So grid generation can be rapid.

Unstructured grids: the accuracy cost

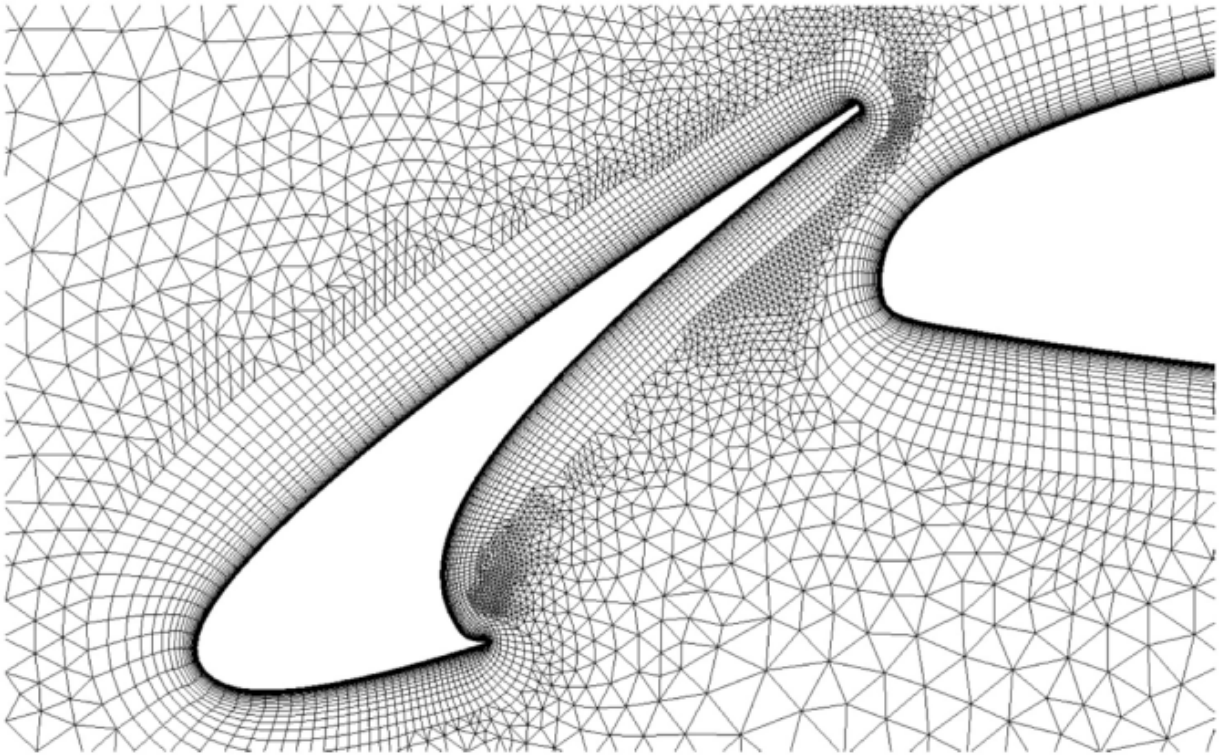
The cost one pays for using unstructured grids is a loss of accuracy, or to put this another way, one typically requires many many more unstructured grid cells to achive the same accuracy as a structured grid.

A quote from Hirsch:

> *It has to be emphasised that structured grids will, compared to unstructured grids, often be more efficient from a CFD point of view, in terms of accuracy, CPU time and memory requirement.*

Hybrid grids

Use high quality structured grids near solid boundaries and unstructured grids to fill the far-field domain.



Truncation error and its connection to grids

- The discretized formulas for CFD methods are often derived assuming regular equidistant mesh points.
- So, an ideal Cartesian grid has the highest quality because it matches the assumptions in the discretised formulas.
- All distortions of cells from the ideal Cartesian grid affect the accuracy of the CFD (negatively)
    - changes in cell size
    - changes in cell shape
- We cannot avoid these distortions on geometries of practical interest
- But we need to be aware of the effect of these distortions on CFD accuracy
- At worst case, severe distortions can reduce the accuacy to zero order

Grid quality recommendations

Taken from Hirsch:

- Avoid *absolutely* discontinuites in grid cell size. Any sudden jump in grid size could reduce the local accuracy to order zero.
- Ensure that the grid sizes vary in a continuous way in all directions.
- Minimise grid distortion, avoiding concave cells or cells with angles between adjacent edges that are too far away from orthogonality. If these angles are reduced to a few degrees, poor accuracy is guaranteed.
- Avoid cells with one or more very short edges, except in boundary layers where high aspect ratio cells are acceptable, provided the cells are sufficiently close to orthogonality to the solid surface.