Numerical Simulation Methods

Part 2 - Applied Concepts

Lecture 9: Introduction to Meshing

Roadmap

Part 2 - Applied Concepts

- **Today: Introduction to meshing**
  - **Different types of mesh**
- Next Lecture: Further details on meshing
- Coordinate transform, finite volume method, flux evaluation
- Solution storage approaches and their implications
- Advanced implicit methods
- Introduction to computer hardware and high performance computing
- Parallel decomposition and efficiency

Meshing

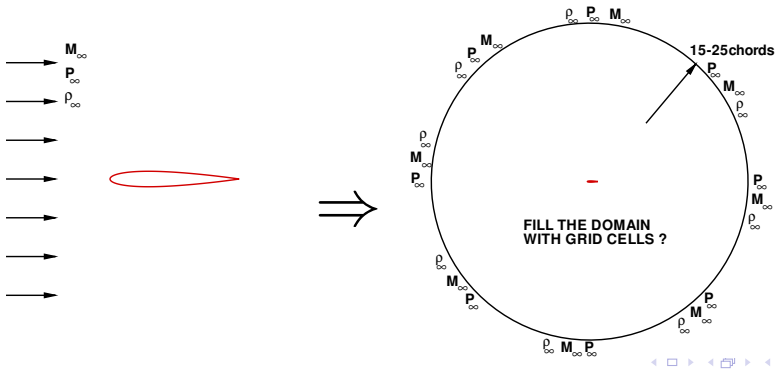
"Today, the generation of suitable meshes for CFD simulations about complex configurations constitutes **a principal bottleneck** *in the simulation workflow process."*

"Significant human intervention is often required in the mesh generation process due to **lack of robustness***. This is evidenced by the inability of current mesh generation software to consistently produce valid high-quality meshes of the desired resolution about complex configurations on the first attempt."*

- CFD Vision 2030 Study, NASA 2014

Meshing

- The generation of a 'good' mesh is as important as an accurate flow solver.
  - (What constitutes a good mesh will be considered later).
- However, the mesh generation process remains a lengthy and non-automated process requiring specialised user skill.

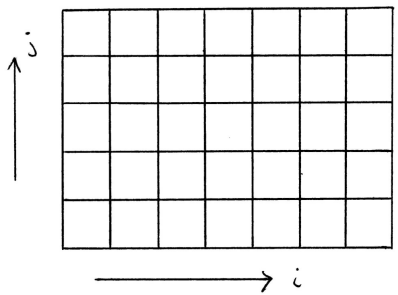


Mesh types - structured

There are two broad classes of mesh type - **Structured** and **Unstructured**.

Structured mesh

Grid lines in each coordinate direction are indexed by i, j, k integer indices. Every line in each family has the same number of points as every other line in the same family NI, NJ, NK.

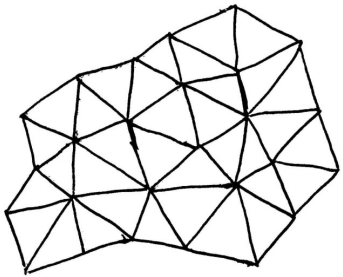


Mesh types - unstructured

There are two broad classes of mesh type - **Structured** and **Unstructured**.

Unstructured mesh

No regular or logical structure to grid point arrangement. Connectivity between points, faces and cells needs to specified for every element.



Mesh types - comparison

There are two broad classes of mesh type - **Structured** and **Unstructured**.

- **Structured Mesh**
  - Simple data structure
  - Regular (fast) data accesses
  - Neighbouring points known implicitly
  - Difficult and user-intensive generation process

Mesh types - comparison

There are two broad classes of mesh type - **Structured** and **Unstructured**.

- **Structured Mesh**
  - Simple data structure
  - Regular (fast) data accesses
  - Neighbouring points known implicitly
  - Difficult and user-intensive generation process
- **Unstructured Mesh**
  - 'Easier' to generate for complex geometries
  - Easy to apply automatic refinement
  - Connectivity data needs to be computed and stored
  - Lower-quality than structured grid - worse for viscous flows

NB. A mesh that has been generated as a structured mesh, can be stored as an unstructured mesh - but not vice versa.

```
do ed=1,nedge

  F3=0.0
  G3=0.0
  F4=0.0
  G4=0.0
  FLX=0.0
  FLXF=0.0
  FLXG=0.0

  aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
    call GETFG(rho(edge(ed,3)),u(edge(ed,3)),                    &
&     v(edge(ed,3)),E(edge(ed,3)),p(edge(ed,3)),fvel(ed,:)       &
&     ,F3,G3)

    call GETFG(rho(edge(ed,4)),u(edge(ed,4)),                    &
&     v(edge(ed,4)),E(edge(ed,4)),p(edge(ed,4)),fvel(ed,:)       &
&     ,F4,G4)
  aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa


  ddddddddddddddddddddddddddddddddddddddddddddddddddddddddddddddd
  FLXF(:)=0.5*(F3(:)+F4(:))*                                     &
&   (grd(edge(ed,2),2)-grd(edge(ed,1),2))
  FLXG(:)=0.5*(G3(:)+G4(:))*                                     &
&   (grd(edge(ed,2),1)-grd(edge(ed,1),1))

  FLX(:)=FLXF(:)-FLXG(:)

  R(edge(ed,3),:)=R(edge(ed,3),:)+FLX(:)
  R(edge(ed,4),:)=R(edge(ed,4),:)-FLX(:)
  ddddddddddddddddddddddddddddddddddddddddddddddddddddddddddddddd


enddo
```

```
    if(i.ne.imax)then
      Fx(2,:,i,j)=0.5*(F(:,i,j)+F(:,i+1,j))
      Gx(2,:,i,j)=0.5*(G(:,i,j)+G(:,i+1,j))
    endif

    enddo
  enddo

  !pause

  do j=1,jmax
    do i=1,imax

    Popr(:,i,j)=+Fx(1,:,i,j)*(grd(2,i+1,j)-grd(2,i,j))                &
&           +Fx(2,:,i,j)*(grd(2,i+1,j+1)-grd(2,i+1,j))              &
&           +Fx(3,:,i,j)*(grd(2,i,j+1)-grd(2,i+1,j+1))              &
&           +Fx(4,:,i,j)*(grd(2,i,j)-grd(2,i,j+1))                  &
&           -Gx(1,:,i,j)*(grd(1,i+1,j)-grd(1,i,j))                  &
&           -Gx(2,:,i,j)*(grd(1,i+1,j+1)-grd(1,i+1,j))              &
&           -Gx(3,:,i,j)*(grd(1,i,j+1)-grd(1,i+1,j+1))              &
&           -Gx(4,:,i,j)*(grd(1,i,j)-grd(1,i,j+1))

     enddo
   enddo
```

Meshing strategies

Among the two mesh classes, there are several commonly-used strategies for meshing.

Structured

- **Simple cartesian,** not body-fitted (immersed boundary)
- **Single-block,** body-fitted
- **Multi-block,** body-fitted

Unstructured

- **Body-fitted**
- **Non-body-fitted,** volume mesh intersects parameterised surface (immersed boundary)

Meshing strategies

Among the two mesh classes, there are several commonly-used strategies for meshing.

Structured

- **Simple cartesian,** not body-fitted (immersed boundary)
- **Single-block,** body-fitted
- **Multi-block,** body-fitted

Unstructured

- **Body-fitted**
- **Non-body-fitted,** volume mesh intersects parameterised surface (immersed boundary)

The choice of meshing strategy is related to the capabilities of your flow solver - most flow solvers are EITHER structured OR unstructured.

Meshing strategies

Some terminology:

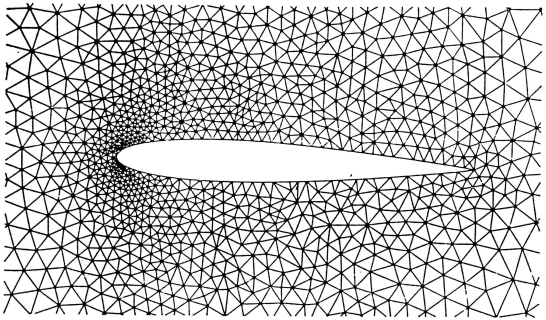
Body-fitted

The vertices and faces of the mesh conform to solid surfaces. *i.e.* The geometry of interest defines, and is defined by, one or more boundaries of the mesh.

- Body-fitted meshes are the norm since boundary conditions can be implemented as part of the existing spatial discretisation
- If a mesh is not body-fitted, then additional numerical methods are needed to apply boundary conditions (immersed boundary method) - this will not discussed in this course

Unstructured meshes

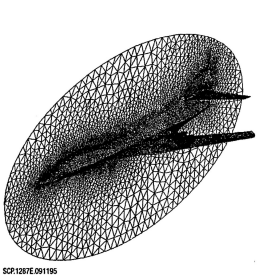
Example NACA0012 aerofoil unstructured grid with triangular elements.



How is this mesh stored?

- List of node coordinates
- List of cells/faces joining nodes

Unstructured meshes



SCP.1287E.091195

Fig. 6 Unstructured grid for inviscid flow past B-1B configuration.

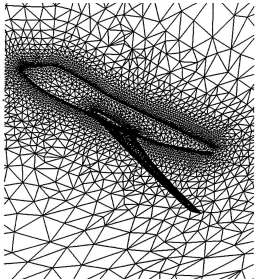


Figure 4. Viscous grid around Boeing 747 wing-body.

Surface and symmetry plane unstructured grid for aircraft

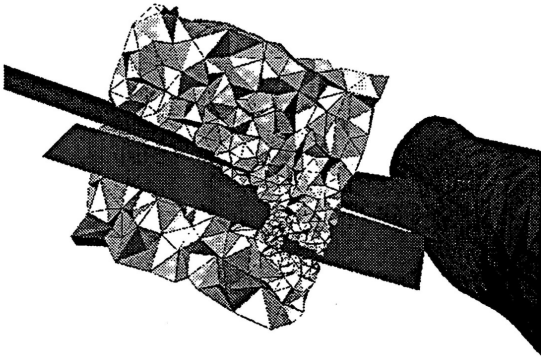
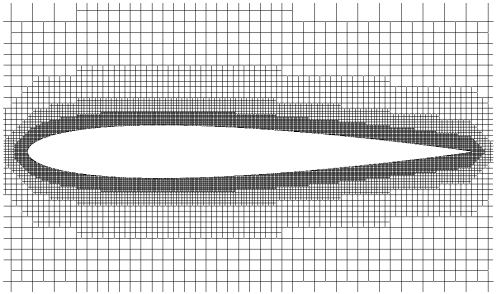


Figure 3. A fragment of the grid around Boeing 747

Unstructured meshes - cutcell mesh

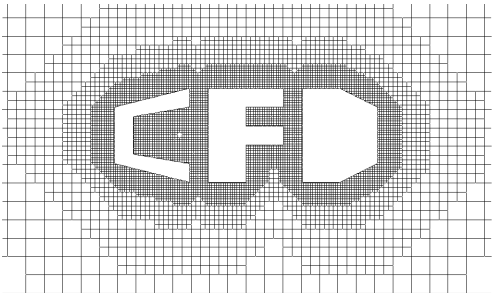
Example NACA0012 aerofoil unstructured grid with **cutcell** mesh.

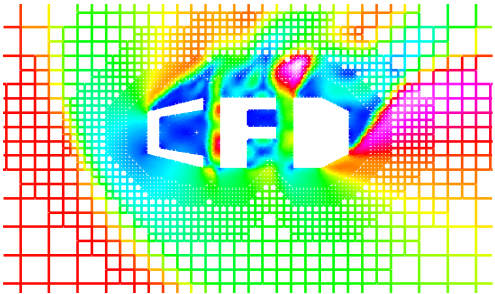


- **Unstructured:** note that some cells have more than four faces
- **Efficient:** refinement near surface where needed and fast coarsening out to the farfield
- **Mesh adaption:** extremely easy and quick to automate refinement and coarsening

Unstructured meshes - cutcell mesh

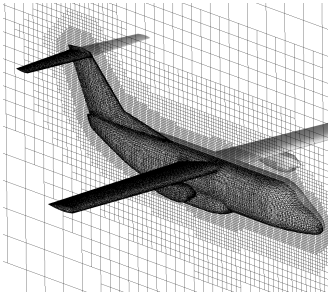
Cutcell meshing is automated: can mesh complex geometries and topologies

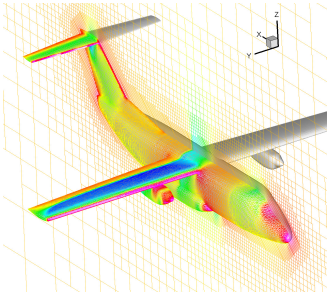


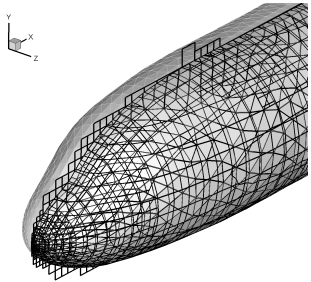


Unstructured meshes - cutcell mesh

Cutcell meshing is automated: can mesh complex geometries and topologies





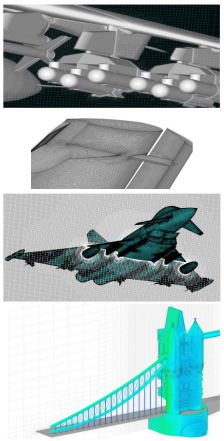


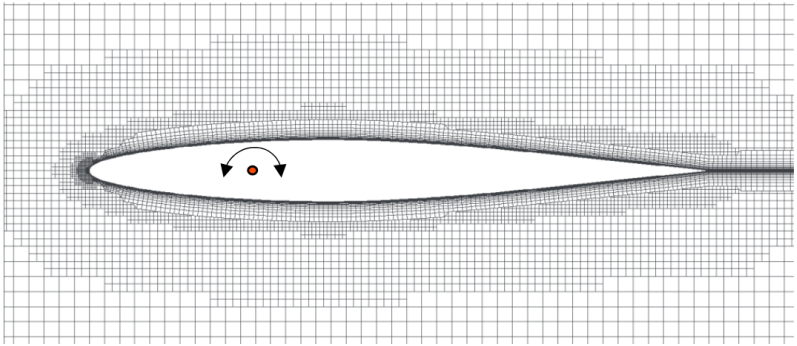
How would you make a cutcell mesh?

- Import surface - probably as an STL triangulation, but could also be CAD
- Make a background grid and detect where intersection occurs
- Refine cells with intersections until you are happy with the resolution
- Split mesh edges and re-insert in mesh - detecting any new edges created
- Split surface edges and insert in mesh - detecting any new edges created
- Add all edges to loops around faces
- Detect any split faces
- Detect any split cells
- Ta-da!

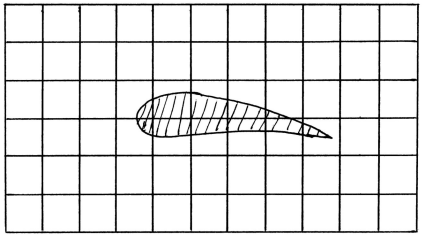
- How do you know if a cell (let's say 2D) is split or not?
- Pick a vertex. Colour the edges that meet at that vertex
- Colour any vertices that are part of a coloured edge
- Repeat this process until no more vertices or edges are coloured on a particular step
- Are there any vertices or edges left uncoloured? If no, you are finished and the cell is unsplit. If yes, you have traced out one loop, and the next loop must be another (new) cell that has been split off from it
- In 3D, exactly the same process can tell you if a cell has been split too, but it requires detecting all of the split faces first, which requires knowing all the split edges first
- This 'bootstrap' integer logic works fine, but only if you make zero errors. A single error will fail the entire process
- Consider how many types of edge, face and cell splits can exist in general. There is almost no limit to their complexity. Your code must handle them all - this is no small challenge!
- Often people avoid these issues; relatively few cartesian methods resolve all splits. Cart3D (NASA) and the one we gave you do, though

- Cartesian meshes are suitable for inviscid or low-Re cases. To handle high-Re, you need to insert boundary layer cells that are appropriately stretched. The SOLAR system from BAe is a nice example of this





Structured meshes - simple cartesian

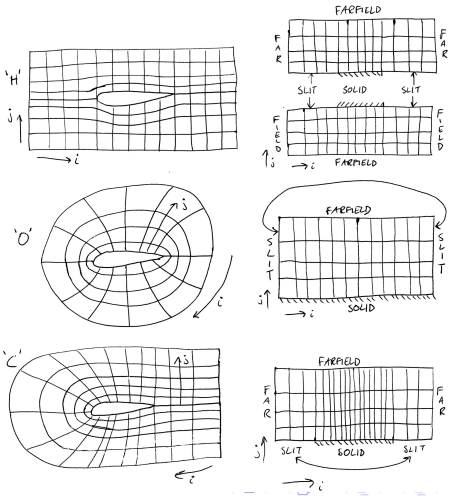


- **Not body-fitted:** poor surface representation or need special parameterisation methods at surface (immersed boundary)
- Rarely used on their own - need body-fitted grids
- Used for background grid in Chimera/Overset methods

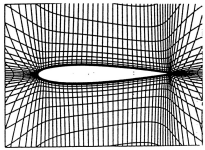
Structured meshes - body-fitted

There are three main mesh topologies when considering body-fitted structured grids.

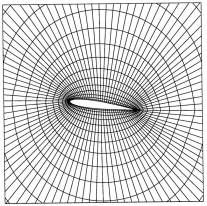
- Each topology maps a regular cartesian grid around a body such that the body forms a boundary of the grid
- O-Type meshes for inviscid flows
- C- and H-type for RANS



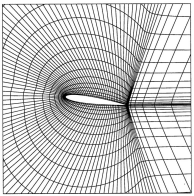
Structured meshes - body-fitted

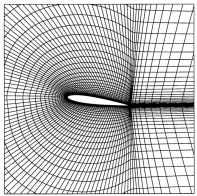


H- type structured grid



O- type structured grid



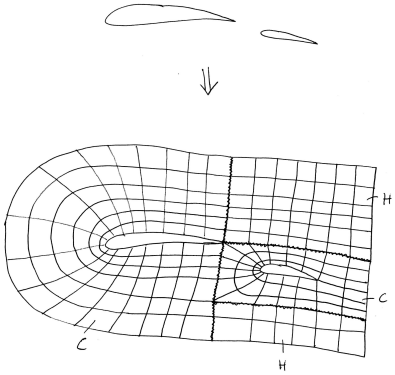


C- type structured grids, Euler and Navier-Stokes

Structured meshes - multiblock

For real configurations we most-often need multiple structured blocks to mesh our domain: a complex mesh is generated from many simple meshes.

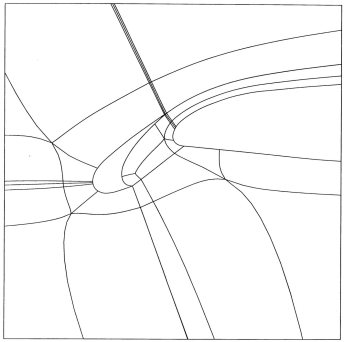
Example: aerofoil with flap



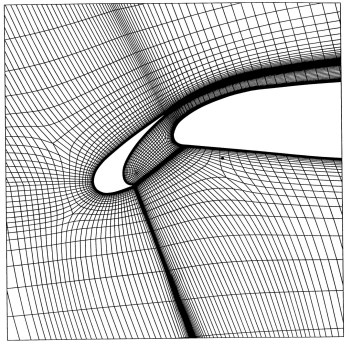
Structured meshes - multiblock

Important

Mesh point distributions at each block boundary must match exactly: therefore no interpolation of the solution is required.



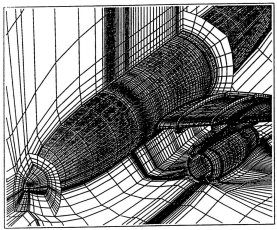
Multiblock grid - block boundaries



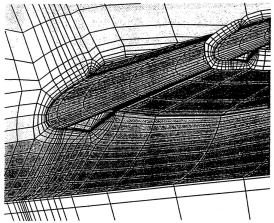
Multiblock grid - block grids

Structured meshes - multiblock

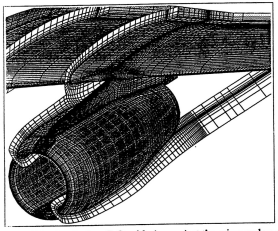
For complex configurations, placing and matching structured blocks is a skilled and time-consuming process



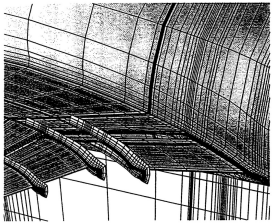
DLR-F6 Navier-Stokes grid (coarse), C-O grids, embedded in H-O grids



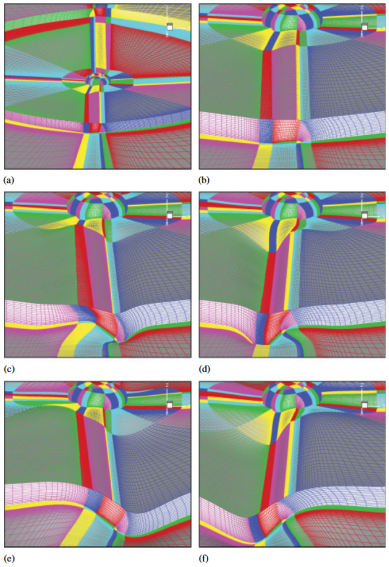
DLR-ALVAST Navier-Stokes grid (coarse), slat region



DLR-F6 Navier-Stokes C-O grids (coarse) at the wing, pylon and nacelle



DLR-ALVAST Navier-Stokes grid (coarse), flap region


(a)
(b)
(c)
(d)
(e)
(f)

Summary

- Mesh generation is the first step towards performing CFD
  - Mesh generation is a manual, time-consuming process
  - Skill and experience required to generate 'good' quality meshes
- Many options available for meshing strategy
  - Structured: cartesian, single-block, multi-block
  - Unstructured: triangular, cutcell

Next Lecture: Further details on meshing

- Advanced mesh types: Hybrid & Chimera/Overset
- Mesh adaption
- Mesh generation methods
- CFL implications in 2D/3D