Numerical Simulation Methods

Part 2 - Applied Concepts

Lecture 10: Meshing - Further Concepts

Roadmap

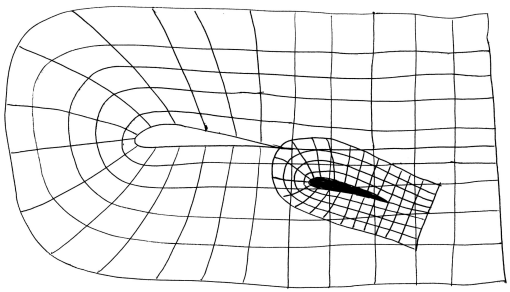
Applied Concepts

- Introduction to meshing
- **Today: Further details on meshing**
  - **Advanced mesh types: Hybrid & Chimera/Overset**
  - **Mesh adaption**
  - **Mesh generation methods**
  - **CFL implications in 2D/3D**
- Next Lecture: 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

Advanced mesh types - Chimera/Overset (overlapping)

An alternative to structured multiblock meshes is the Chimera/Overset approach.

- Single-block grids generated around each body independently
- Complete mesh is assembled by placing all meshes over each other: no matching boundaries
- Used for simulations with relative motion of bodies
- Special flow solver routines required to interpolated solution between Chimera/Overset blocks

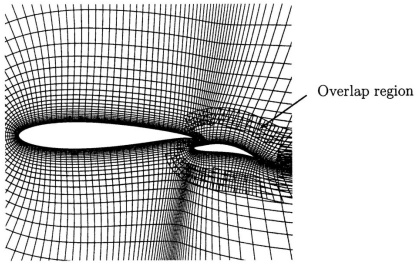


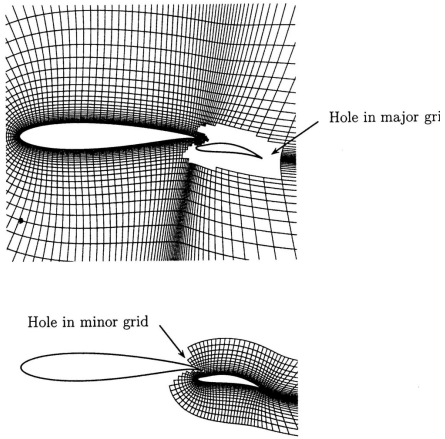
Advanced mesh types - Chimera/Overset

- Overlap region may be large - inefficient to solve for the solution in all cells in all blocks
  - Therefore, use an algorithm to **remove redundant cells** in each mesh - minimise size of the overlap region
- Also need a complex tagging algorithm to find neighbouring cells between different grid blocks. Cells are tagged as:
  - **Normal cell** - solution is stored and updated by the solver
  - **Fringe cell** - solution is stored by interpolated from another grid block
  - **Hole cell** - solution is not stored
- Special flow solver routines required to interpolated solution between overlapping Chimera/Overset blocks
  - Interpolation must be accurate to preserve overall spatial accuracy of the solver

Advanced mesh types - Chimera/Overset

Example: Aerofoil with flap





Advanced mesh types - Chimera/Overset

Example: Helicopter rotor

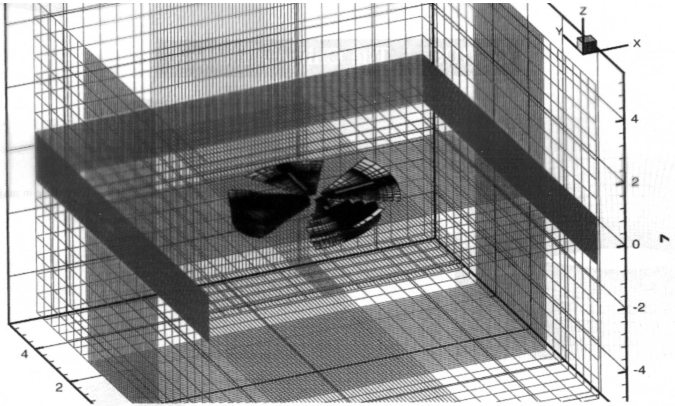


Figure 1 - Chimera grid system for the isolated rotor in forward flight

Advanced mesh types - Hybrid Meshes

Studies show that **unstructured meshes are not suitable for viscous computations**

- Viscous terms are second differences, and normally require at least five points in each coordinate direction
- It is very awkward to obtain high accuracy over many cells for unstructured meshes
- Also there is no obvious cell direction normal to solid surfaces, so large gradients are difficult to main near surfaces (required for turbulence models)

To overcome these problems, hybrid meshes have been implemented.

Hybrid meshes

A structured grid is used around solid surfaces and unstructured meshes are used far away from solid surfaces. Exact matching at the boundary between the two means no interpolation is necessary.

Advanced mesh types - Hybrid Meshes

- A high-quality structured grid near the surface is very good at accurately resolving flow features, particular viscous layers
- An unstructured grid away from solid surfaces simplifies the mesh generation process, and allows coarser grid densities away from areas of interest

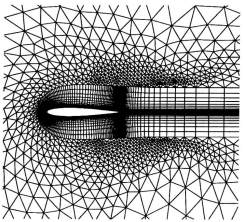


Fig. 8a: Mesh (7709 cells)

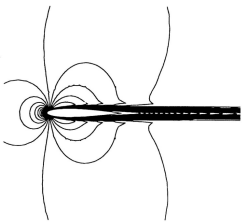
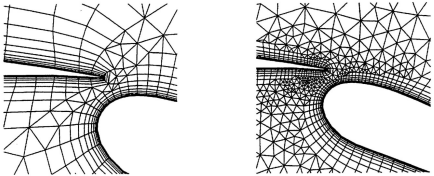


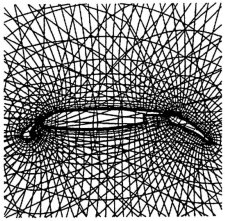
Fig. 8b: Iso-Mach lines



(a) Without Boundary Refinemet (a) With Boundary Refinemet

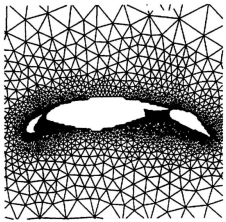
Figure 4 Effect of Boundary Refinement on Grid Quality

Advanced mesh types - Hybrid Meshes

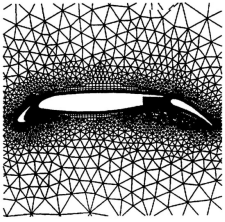


(a) Structured Grid Around Three Components

(b) Trimmed Structured Grid



(c) Unstructured Grid in the Void



(d) Resulting Hybrid Grid

Figure 5 Steps Involved in Generating Hybrid Grid for a Three Element Airfoil

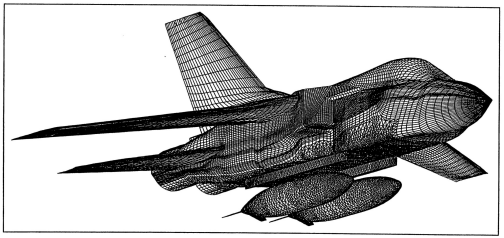


Figure 5 : Hybrid Grid for Release of Fuel Tanks from Tornado

Mesh adaption

Motivation

- Numerical dissipation present in all numerical schemes means that regions of the flow with large gradients (*e.g.* shocks)become smeared over multiple computation cells
- To make the solution 'sharper', we need to add more mesh points to reduce the mesh spacing

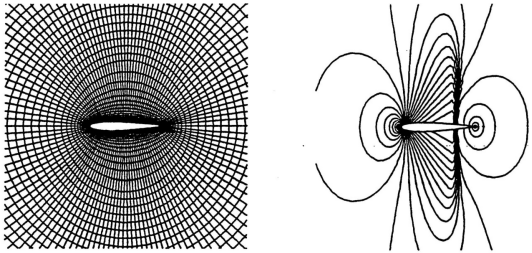
Refinement criteria Normally a set of PDEs are solved such that:

$$\Delta x_{i,j,k} \propto \left(\frac{\partial S_{i,j,k}}{\partial x}\right)^{-1} \qquad \Delta y_{i,j,k} \propto \left(\frac{\partial S_{i,j,k}}{\partial y}\right)^{-1} \qquad \Delta z_{i,j,k} \propto \left(\frac{\partial S_{i,j,k}}{\partial z}\right)^{-1} \tag{1}$$

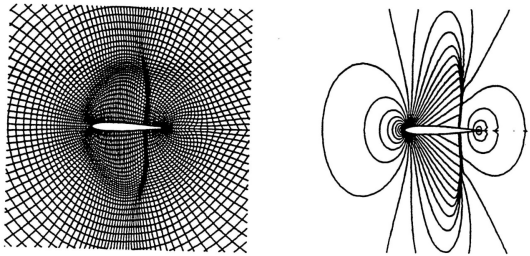
where S is some scalar quantity. Mach number is often used. The resulting equations are normally very complex, and require more time to solve than the flow equations.

Structured mesh adaption

To refine one area of a structured mesh we must **refine along all grid lines** that run through the area of refinement to keep the structured topology

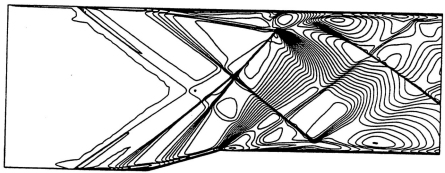


Original Grid and Solution

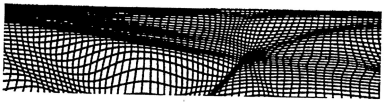


Adapted Grid and Solution

Structured mesh adaption

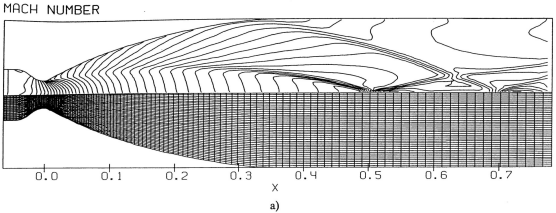


(b) density contours

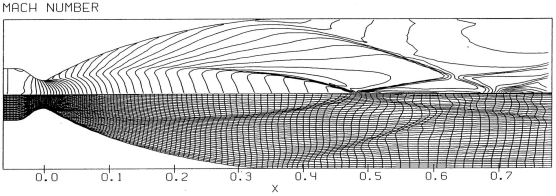


(c) Detail of grid at shock-wave BL interaction

Figure 2) Mach 2.0 laminar flow into a channel. Adaptation of 121X91 grid



a)



b)

Fig. 9. Nozzle flow problem. a) solution and grid without adaptation; b) adaptive solution and grid.

Unstructured mesh adaption

Unstructured mesh adaption is much simpler.

- Extra nodes, and hence cells, are simply added where the gradients are large (known as **enrichment**)
- Some cells can be removed where gradients are very small, but normally there will be more cells in an adapted unstructured mesh than the original one.
- This part is simple, but of course when new points are added new cells are created, and all the connectivity must be recomputed. This is extremely time consuming.

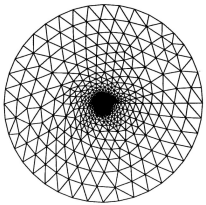


Fig. 6a: Initial mesh (5242 cells)

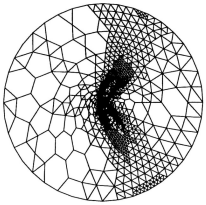


Fig. 6b: Final Mesh (10149 cells)

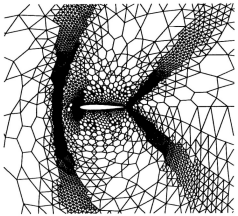


Fig. 6c: Details of the final mesh

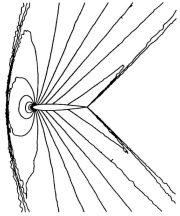


Fig. 6d: Iso-Mach lines
$(0 - 1.6, \Delta 0.066)$

Unstructured mesh adaption

We can also refine an cartesian type grid as an unstructured grid.
This makes recalculation of connectivity slightly simpler.

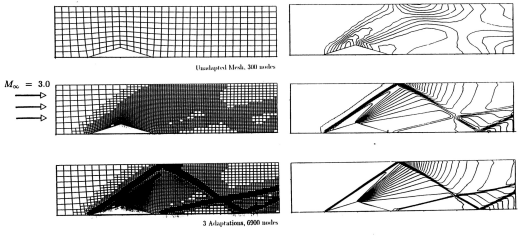


Figure 9. Sequence of grid and evolving solution of Mach 3 duct flow. Wedge half angle = 16.7°. Solution presented through pressure contours.

Unstructured cartesian type grid with enrichment

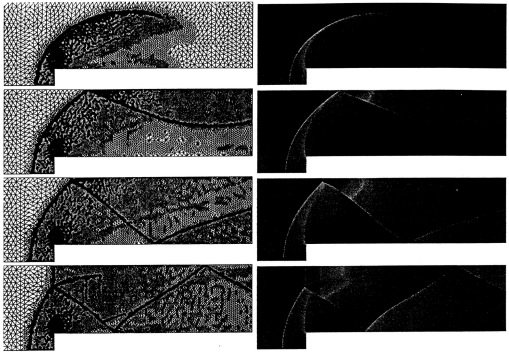
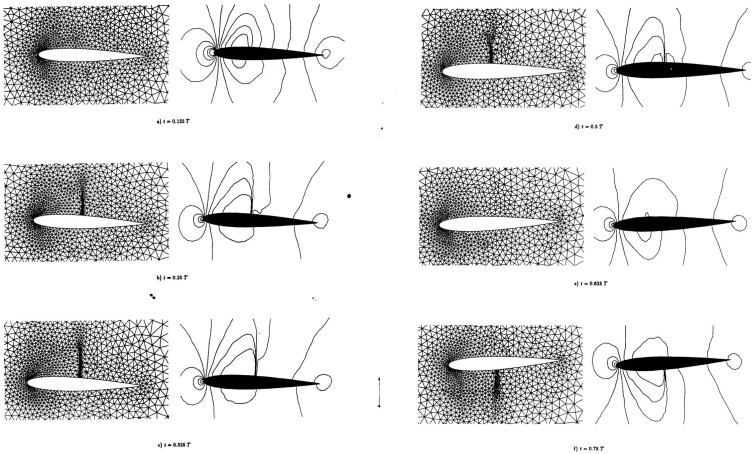


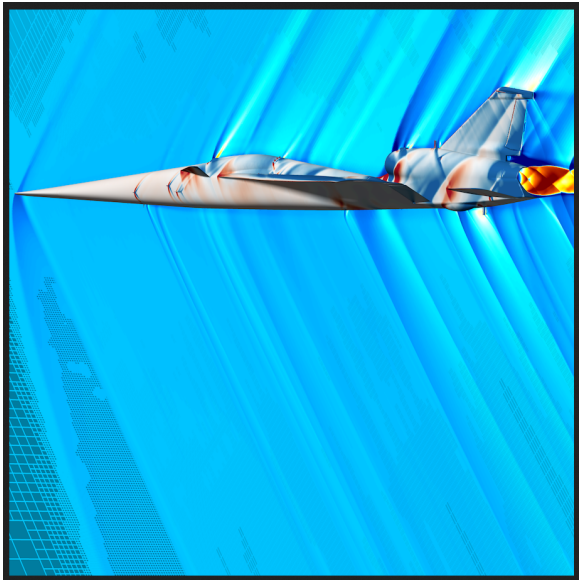
Figure 8: Evolution of the adaptive grid and corresponding solutions

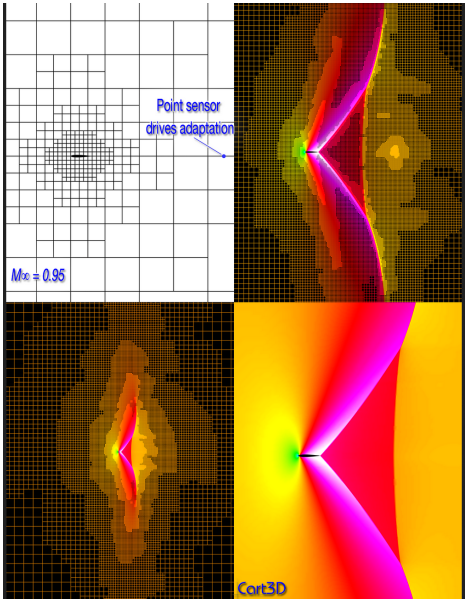
Unstructured adapted grid

Unstructured mesh adaption

Grid-refinement at each real time-step for unsteady flow.



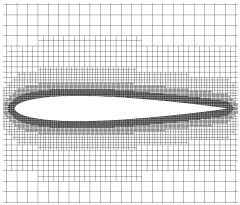


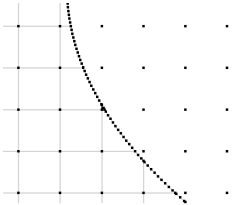

Point sensor
drives adaptation
M∞ = 0.95
Cart3D

Lecture 10: Meshing - Further Concepts

Part 2

Mesh Generation Methods - Cut Cell Meshes





- Unlike other methods, the cut cell method does not require a surface mesh to generate a volume mesh
- Only a 'surface representation' (*e.g.* CAD surface) is required which allows the computation of intersections with cells
- The volume mesh is generated by starting from a single cell, and refining the mesh from 'outer to inner' and refining near the surface

Mesh Generation Methods - Cut Cell Meshes

- This approach needs a completely arbitrary cell shape unstructured solver, with a particular data storage method, but removes the mesh generation problem.
- Not good for viscous flows though, so a hybrid approach can be used; viscous mesh around the surface, with the cut cell mesh intersecting the outer edge of the viscous layer. But this removes the advantage of not having to generate a surface mesh.

General unstructured meshes (*e.g.* tetrahedral) are generated by 'advancing front' or Delauney triangulation procedures, which move away from solid surfaces; these method therefore require a good surface mesh definition.

Mesh Generation Methods - Surface Meshes

Important

Most mesh generation methods are driven by the surface mesh distribution. The structure, topology and quality of the volume mesh is highly dependent on that of the surface mesh.

- If you have a poor quality surface you will always get a poor quality volume mesh
- For example, there must be a small cell size at aerofoil leading and trailing edges. This is where the flow gradients are largest, and so need more points to capture them
- Will often begin with some kind of CAD model, but this will not always be in a usable form for a CFD surface generator - a lot of time spent 'CAD-fixing'

NASA researchers estimate that of the total simulation time, from receiving a geometry to actually having usable post-processed data from the flow-solver, up to 85% of the total time is taken up in the surface mesh generation stage.

General Volume Mesh Generation

There are then many methods to compute the coordinates of the field mesh, based on the generated suface mesh. Three of the most popular are:

1. Solve a set of P.D.E.'s

Numerically solving a set of equations, with for example terms defining orthogonality and stretching. The equations can be some combination of Laplace, Poisson, and Equidistribution equations. This is a three-parameter problem, i.e. i,j,k points all generated simultaneously.

2. Hyperbolic marching from inner boundary

Define the inner boundary distribution, and inner boundary normal direction only. The hyperbolic scheme then 'marches' away from the inner boundary. This can be useful for concave inner boundaries, since there is a repulsion type term, which ensures no mesh crossover is possible. However, the definition of the outer boundary cannot be determined prior to the generation, and this can cause problems. For example, when generating a multiblock mesh the block boundaries would be defined first, then the interior points computed. Hyperbolic is a two parameter problem, i.e. for each k plane moving from inner to outer, i,j points are generated.

General Volume Mesh Generation

3. Interpolation between the inner and outer boundaries.

Define the inner and outer boundary distributions, and using a weighted interpolation, which also includes a defined inner boundary normal direction term. This is much simpler than 1) or 2) since the mesh positions are available directly, i.e. the equations give $(x, y, z) = F(inner, outer, normal)$. This is a one-parameter problem, as for each i,j point on the surface, the interpolation generates each set of k points independently. Not very robust !

CFL Condition in 2D/3D

We have encountered the CFL condition in one-dimension which gives the following upper-bound on allowable timestep:

$$1-\mathrm{D}: \quad \Delta t \leq \frac{\Delta x}{\mathrm{MAX}(\mathrm{wavespeed}_x)} \tag{2}$$

In higher dimensions we have wavespeed(s), or signal propagation speed(s), in each coordinate direction.

CFL Condition in 2D/3D

We have encountered the CFL condition in one-dimension which gives the following upper-bound on allowable timestep:

$$1-\mathrm{D}: \quad \Delta t \leq \frac{\Delta x}{\mathrm{MAX}(\mathrm{wavespeed}_x)} \tag{2}$$

In higher dimensions we have wavespeed(s), or signal propagation speed(s), in each coordinate direction. If the mesh spacings are $\Delta x, \Delta y, \Delta z$, then the CFL condition becomes:

$$2-\mathrm{D}: \quad \Delta t \leq \mathrm{MIN}\left(\frac{\Delta x}{\mathrm{MAX}(\mathrm{wavespeed}_x)}, \frac{\Delta y}{\mathrm{MAX}(\mathrm{wavespeed}_y)}\right) \tag{3}$$

$$3-\mathrm{D}: \quad \Delta t \leq \mathrm{MIN}\left(\frac{\Delta x}{\mathrm{MAX}(\mathrm{wavespeed}_x)}, \frac{\Delta y}{\mathrm{MAX}(\mathrm{wavespeed}_y)}, \frac{\Delta z}{\mathrm{MAX}(\mathrm{wavespeed}_z)}\right) \tag{4}$$

CFL Condition in 2D/3D - implications

- The computation of three-dimensional flows using explicit schemes is the most straightforward
- But it is **very time-consuming** due to the CFL condition resulting in **a large number of time-steps**
- Hence, implicit schemes are normally used for steady flows, where we don't care about the solution progression to steady state.
  - The size of the resulting matrix means that implicit schemes for 3-D problems need multi-node parallelism or efficient matrix approximations
  - For example, **operator splitting** is often used, wherein the scheme is split into a sequence of '1-D type' schemes; more details in future lectures.

Summary

- Advanced mesh configurations
  - Chimera/Overset allows for relative rigid-body motion
  - Hybrid meshes combine the advantages of structured and unstructured meshes
- Grid adaption allows the redistribution of cells to where we need extra spatial resolution
  - Simpler to apply to unstructured meshes
- Several approaches to mesh generation
  - Cut-cell: doesn't require pre-existing surface mesh
  - Advancing front for general unstructured meshes
  - Elliptic and hyperbolic solve PDEs for mesh point locations
  - Interpolation is simpler, no PDEs to solve

Next Lecture: Numerical methods for real problems: coordinate transform, finite volume method and solution storage