MECH4480
Computational Fluid Dynamics

CFD

Finite-Volume Formulation

Content

OpenFOAM

1. OpenFOAM
    1. Setting up and running a first case
    2. Setting Boundary Conditions
    3. Adjusting the solver
    4. Getting good solutions
2. Turbulence modelling
    1. What is turbulence
    2. Modelling the effects of turbulence
3. Solving the Governing equations:
   The solution process in OF

General Transport Eqn (revision)

$$\frac{d(\rho\phi)}{dt} + div(\rho\phi\mathbf{u}) = div(\Gamma\, grad\,\phi) + S_\phi \qquad \Gamma = \text{ diffusion coefficient}$$

This equation is the starting point for all computational procedures in the finite volume method.

The Finite Volume Method:

The above equation must hold for each control volume.
Hence integration over a control volume yields:

$$\int_{CV} \frac{d(\rho\phi)}{dt} dV + \int_{CV} div(\rho\phi\mathbf{u}) dV = \int_{CV} div(\Gamma\, grad\,\phi) dV + \int_{CV} S_\phi dV$$

$$\int_{CV} div\,\mathbf{a} dV = \int_{A} \mathbf{n}\,.\,\mathbf{a} dA$$ **Gauss' divergence theorem**

$$\frac{d}{dt}\left(\int_{CV} \rho\phi dV\right) + \int_{A} \mathbf{n}\,.\,(\rho\phi\mathbf{u}) dA = \int_{A} \mathbf{n}\,.\,(\Gamma\, grad\,\phi) dA + \int_{CV} S_\phi dV$$

Rate of increase of ϕ + Net rate of decrease of ϕ due to convection across the boundaries = Rate of increase of ϕ due to diffusion across the boundaries + Net rate of creation of ϕ

Finite Volume Method balances fluxes across boundaries

Some transport Eqn's

$$\frac{d}{dt}\left(\int_{CV} \rho\phi dV\right) + \int_A \mathbf{n}.(\rho\phi\mathbf{u})dA = \int_A \mathbf{n}.(\Gamma\, grad\,\phi)dA + \int_{CV} S_\phi dV$$

Rate of increase of ϕ + Net rate of decrease of ϕ due to convection across the boundaries = Rate of increase of ϕ due to diffusion across the boundaries + Net rate of creation of ϕ

Mass flow: (φ=1)

$$\frac{d}{dt}\left(\int_{CV} \rho dV\right) + \int_A \mathbf{n}.(\rho\mathbf{u})dA = \cancel{\int_A \mathbf{n}.(\Gamma\, grad\, 1)dA} + \int_{CV} S dV$$

Rate of mass increase + Net rate of decrease of mass due to convection across the boundaries = n/a + Creation of mass by sources

Momentum (x):

$$\frac{d}{dt}\left(\int_{CV} \rho u dV\right) + \int_A \mathbf{n}.(\rho u\mathbf{u})dA = \int_A \mathbf{n}.(\Gamma\, grad\, u)dA + \int_{CV} S_{Mx} dV$$

Rate of momentum increase + Net rate of decrease of momentum due to convection across the boundaries = Rate of increase of momentum due to diffusion across boundaries + Creation of mass by sources. Includes $\frac{dP}{dx}$ term

Solving the Governing Equation

Steady state solvers:

(Get to steady state solution as fast as possible)

- Solving a single transport equation
  - Diffusion problems
  - Sub-step for more complex methods
- Solving coupled transport equations
  - Convection problems, coupled convection-diffusion problems
  - Pressure – velocity coupling

Transient solvers:

(Time accurate solution)

- See Eilmer

Steady State - Pure Diffusion (e.g. Heat equation)

$$\frac{d}{dt}\left(\int_{CV}\rho\phi dV\right)+\int_{A}\mathbf{n}.(\rho\phi\mathbf{u})dA=\int_{A}\mathbf{n}.(\Gamma\, grad\,\phi)dA+\int_{CV}S_\phi dV$$

In a 1-D domain:
(e.g. variation in Φ as shown in Fig 1)

$$\frac{d}{dx}\left(\Gamma\frac{d\phi}{dx}\right)+S=0$$

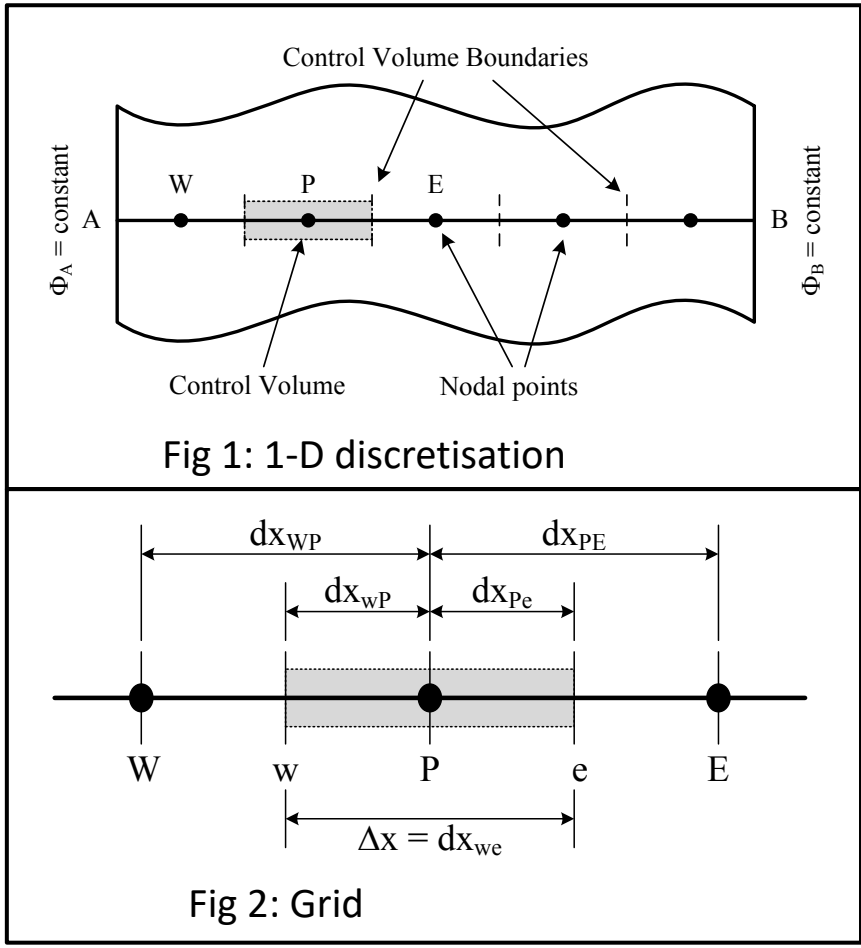


Fig 1: 1-D discretisation

1. Grid Generation:

dx_{WP}, dx_{WE} etc are defined individually to permit non-uniform grids

Fig 2: Grid

2. Discretisation at P:

$$\begin{aligned}0 &= \int_{A}\mathbf{n}.(\Gamma\, grad\,\phi)dA+\int_{CV}S_\phi dV\\ &= \left(\Gamma A\frac{d\Phi}{dx}\right)_e-\left(\Gamma A\frac{d\Phi}{dx}\right)_w+\bar{S}\,\Delta V\end{aligned}$$

1-D cont'd

$$\left(\Gamma A \frac{d\phi}{dx}\right)_e - \left(\Gamma A \frac{d\phi}{dx}\right)_w + \bar{S}\,\Delta V = 0$$

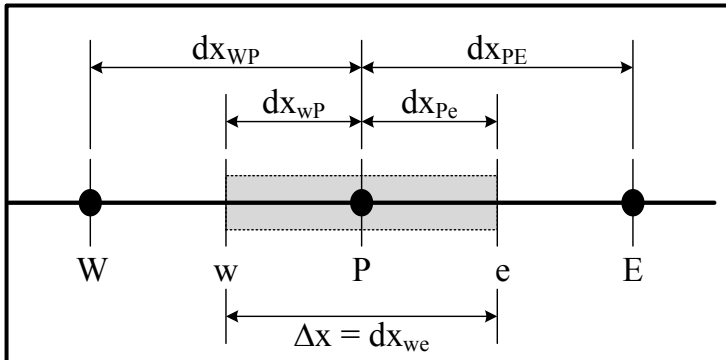


Fig 1: Grid

Determine values at control volume boundaries

$$\Gamma_w = \frac{\Gamma_W + \Gamma_P}{2} \qquad \Gamma_e = \frac{\Gamma_P + \Gamma_E}{2}$$

Average to get face properties

$$\left.\frac{d\Phi}{dx}\right|_w = \frac{\Phi_P + \Phi_W}{dx_{WP}} \qquad \left.\frac{d\Phi}{dx}\right|_e = \frac{\Phi_E + \Phi_P}{dx_{PE}}$$

Use Central difference to find gradients

Determine Source Term

$$\bar{S}\,\Delta V = S_u + S_P \Phi_P$$

Define two contributors for convenience

Discretised Transport equation

$$\Gamma_e A_e \left(\frac{\Phi_E - \Phi_P}{dx_{PE}}\right) - \Gamma_w A_w \left(\frac{\Phi_P - \Phi_W}{dx_{WP}}\right) + (S_u + S_P \Phi_P) = 0$$

$$\left(\frac{\Gamma_e}{dx_{PE}} A_e + \frac{\Gamma_w}{dx_{WP}} A_w - S_P\right) \Phi_P = \left(\frac{\Gamma_w}{dx_{WP}} A_w\right) \Phi_W + \left(\frac{\Gamma_e}{dx_{PE}} A_e\right) \Phi_E + S_u$$

$$a_P \Phi_P = a_W \Phi_W + a_E \Phi_E + S_u$$

I.e. Φ_P can be calculated in terms of Φ at surrounding points and source term.

Boundary Conditions – Ghost Cells

What happens at boundaries?
-> properties and gradients (e.g. at W) not available

$$\left(\frac{\Gamma_e}{dx_{PE}}A_e + \frac{\Gamma_w}{dx_{WP}}A_w - S_P\right)\phi_P = \left(\frac{\Gamma_w}{dx_{WP}}A_w\right)\phi_W + \left(\frac{\Gamma_e}{dx_{PE}}A_e\right)\phi_E + S_u$$

Ghost Cell Approach:
Add *ghost* cells and update ghost cell values as part of solution cycle

a) Set ghost cell to desired value
b) Calculate flux update using ghost cell data
c) Calculate new ghost cell value. Repeat...

Fixed Value (Dirichlet B/C)
- Ghost cell maintains fixed Value

Fixed Gradient (Neumann B/C)
- Calculate ghost cell value to maintain desired gradient of flux

E.g. zeroGradient $\Phi_W = \Phi_E$

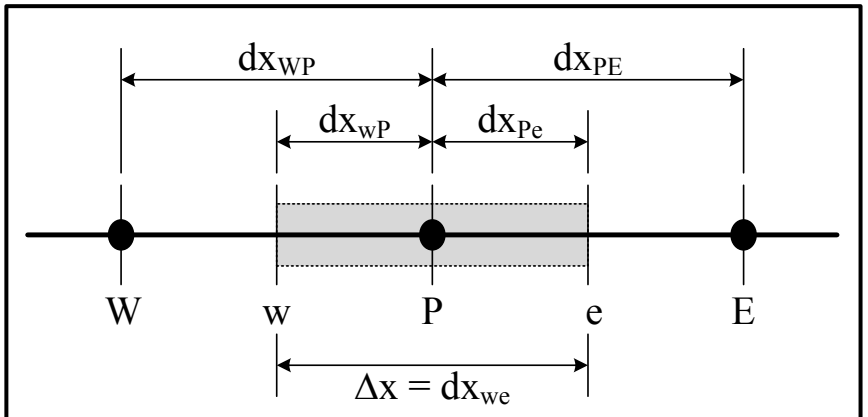


Fig 1: Normal Grid

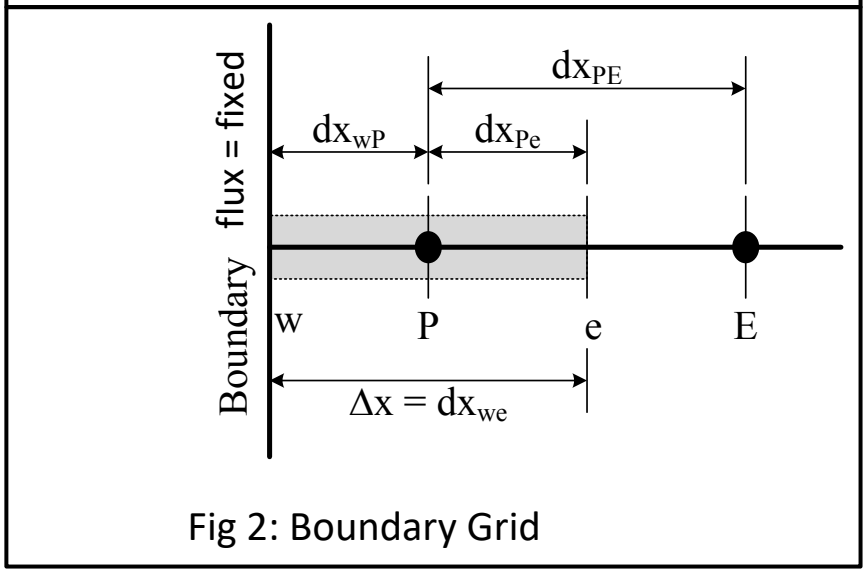


Fig 2: Boundary Grid

Boundary Conditions – Prescribed flux

What happens at boundaries?

-> properties and gradients (e.g. at W) not available

$$\left(\frac{\Gamma_e}{dx_{PE}}A_e + \frac{\Gamma_w}{dx_{WP}}A_w - S_P\right)\phi_P = \left(\frac{\Gamma_w}{dx_{WP}}A_w\right)\phi_W + \left(\frac{\Gamma_e}{dx_{PE}}A_e\right)\phi_E + S_u$$

Prescribed flux approach:

Remove dependency on conditions outside of mesh and then adjust source terms.

a) Set $dx_{WP} = 0,\ \Phi_W = 0$

b) Calculate desired flux across (w) interface

c) Add flux to source terms.

$$S'_u + S'_P\,\Phi_P = S_u + S_P\,\Phi_P + \mathrm{Flux}|_w$$

d) Solve equations. Repeat

Fixed Value (Dirichlet):

- calculate flux *as if* Φ_W at desired value

Fixed Flux:

- Set desired flux directly.

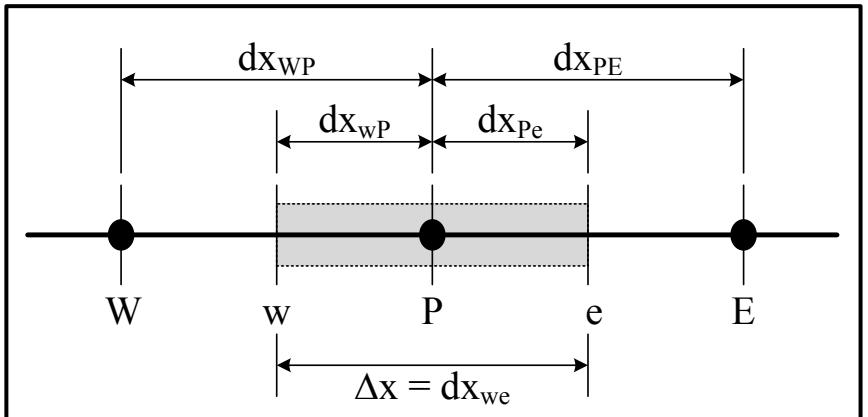


Fig 1: Normal Grid

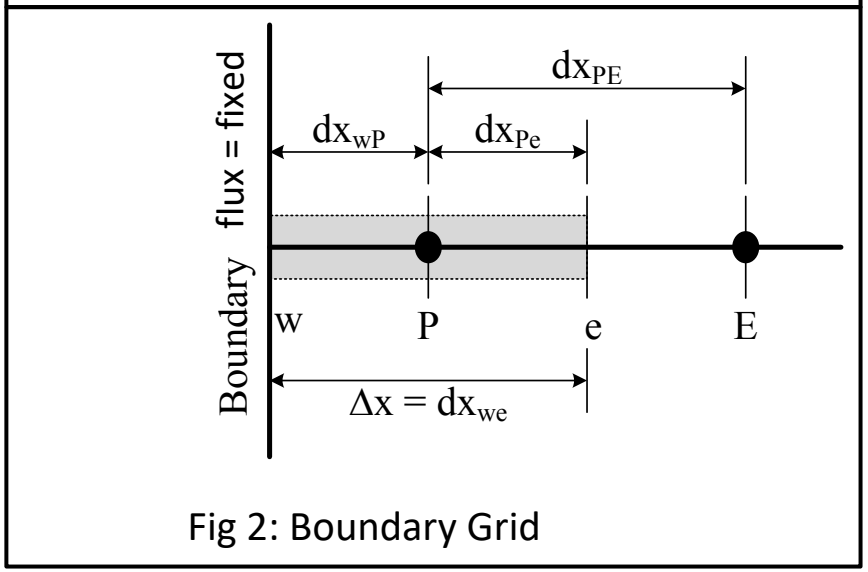


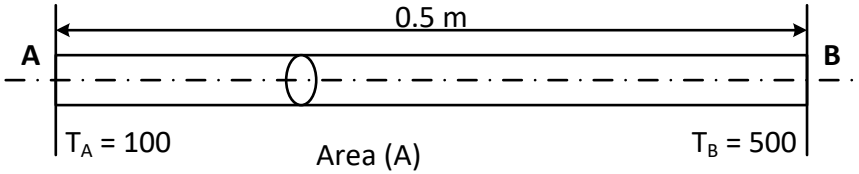
Fig 2: Boundary Grid

Example: 1-D Heat equation (steady)

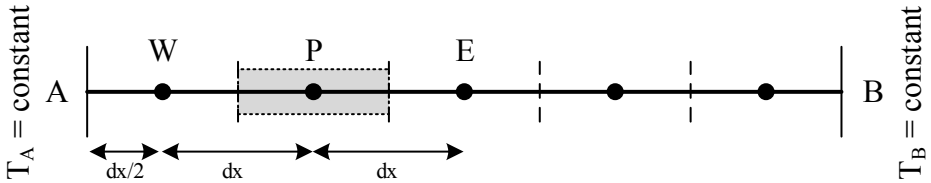
Consider the problem of source free heat conduction in an insulated rod whose ends are maintained at a constant temperature of 100°C and 500°C respectively.

Conductivity k = 1000 W/(mK), A = 10 x 10^{-3} m^2

(Example 4.1 from H K Versteeg & W Malalasekera , An introduction to Computational Fluid Dynamics)



$$\frac{d}{dx}\left(k\frac{dT}{dx}\right)+S=0$$



Create discretised equations:

$$0 \;=\; \frac{Flux_e - Flux_w}{dx} = \frac{d}{dx}\left(k\,A\,\frac{dT}{dx}\right) = 0$$

$$0 \;=\; Flux_e - Flux_w = \frac{k_e\,(T_E - T_P)}{dx_{PE}}\,A_e - \frac{k_w\,(T_P - T_W)}{dx_{WP}}\,A_w$$

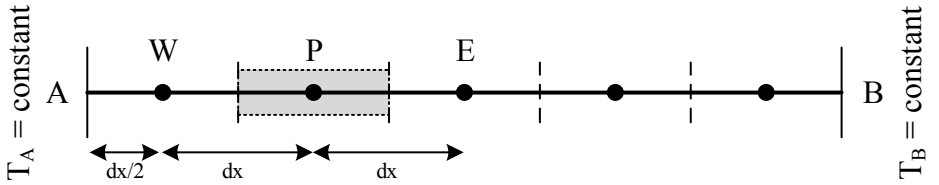
$$\left(\frac{k_e}{dx_{PE}}\,A_e + \frac{k_w}{dx_{WP}}\,A_w\right)T_P = \left(\frac{k_w}{dx_{WP}}\,A_w\right)T_W + \left(\frac{k_e}{dx_{PE}}\,A_e\right)T_E$$

In more compact format, as most terms are constants.

$$a_P\,T_P = a_W\,T_W + a_E\,T_E$$

with $a_W = \frac{k}{dx}\,A; \quad a_E = \frac{k}{dx}\,A; \quad a_P = a_E + a_W$

Example cont'd



Consider Boundary Conditions (at A):

Prescribed flux approach

$$0 \quad = \quad Flux_e - Flux_w = \frac{k\,(T_E - T_P)}{dx}\,A - \frac{k\,(T_P - T_A)}{\frac{dx}{2}}\,A$$

$$\left(\frac{k}{dx}\,A + \frac{2\,k}{dx}\,A\right)\,T_P = 0 \times T_W + \left(\frac{k}{dx}\,A\right)\,T_E + \left(\frac{2\,k}{dx}\,A\right)\,T_A$$

This is similar to:

$$a_P\,T_P = a_W\,T_W + a_E\,T_E + S_u$$

with $a_W = 0; \quad a_E = \frac{k\,A}{dx}; \quad a_P = a_E + a_W - S_P; \quad S_P = -\frac{2\,k\,A}{dx}; \quad S_u = \frac{2\,k\,A}{dx}\,T_A$

Similarly for B:

$$a_P\,T_P = a_W\,T_W + a_E\,T_E + S_u$$

with $a_W = \frac{k\,A}{dx}; \quad a_E = 0; \quad a_P = a_E + a_W - S_P; \quad S_P = -\frac{2\,k\,A}{dx}; \quad S_u = \frac{2\,k\,A}{dx}\,T_B$

Determine constants: $$a_P \Phi_P = a_W \Phi_W + a_E \Phi_E + S_u$$

Node	a_W	a_E	S_u	S_p	$a_P = a_W + a_E - S_P$
1	0	100	$200\,T_A$	-200	300
2	100	100	0	0	200
3	100	100	0	0	200
4	100	100	0	0	200
5	100	0	$200\,T_B$	-200	300

Create set of linear simultaneous equations and rearange in matrix format to obtain $\mathbf{A}\,x = b$

$$\begin{bmatrix} 300 & -100 & 0 & 0 & 0 \\ -100 & 200 & -100 & 0 & 0 \\ 0 & -100 & 200 & -100 & 0 \\ 0 & 0 & -100 & 200 & -100 \\ 0 & 0 & 0 & -100 & 300 \end{bmatrix} \begin{bmatrix} T_1 \\ T_2 \\ T_3 \\ T_4 \\ T_5 \end{bmatrix} = \begin{bmatrix} 200\,T_A \\ 0 \\ 0 \\ 0 \\ 200\,T_B \end{bmatrix}$$

Use matrix solver to find T_1 to T_5.

Summary for diffusion

Derivation of discretised diffusion equation in 2-D and 3-D is available in text books.

Approach is identical, but all six neighbouring points need to be considered.

General solution is given by: (nb = neighbouring)

$$a_P \phi_P = \sum a_{nb} \phi_{nb} + S_u$$

$$a_P = \sum a_{nb} - S_P$$

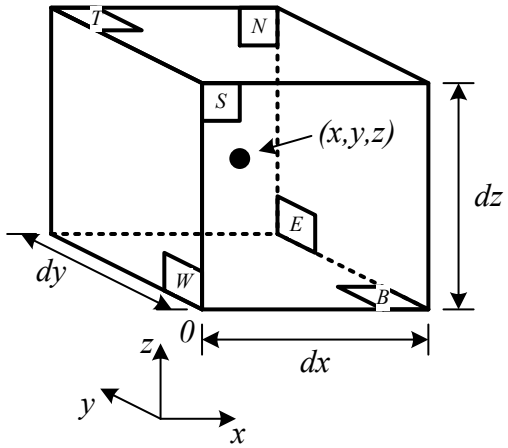


Fig 1: Cartesian Control Volume

Tab 1: coefficients for diffusion equation

	a_w	a_e	a_s	a_n	a_b	a_t
1-D	$\frac{\Gamma_w A_w}{dx_{WP}}$	$\frac{\Gamma_e A_e}{dx_{PE}}$				
2-D	$\frac{\Gamma_w A_w}{dx_{WP}}$	$\frac{\Gamma_e A_e}{dx_{PE}}$	$\frac{\Gamma_s A_s}{dx_{SP}}$	$\frac{\Gamma_n A_n}{dx_{PN}}$		
3-D	$\frac{\Gamma_w A_w}{dx_{WP}}$	$\frac{\Gamma_e A_e}{dx_{PE}}$	$\frac{\Gamma_s A_s}{dx_{SP}}$	$\frac{\Gamma_n A_n}{dx_{PN}}$	$\frac{\Gamma_b A_b}{dx_{BP}}$	$\frac{\Gamma_b A_b}{dx_{PT}}$

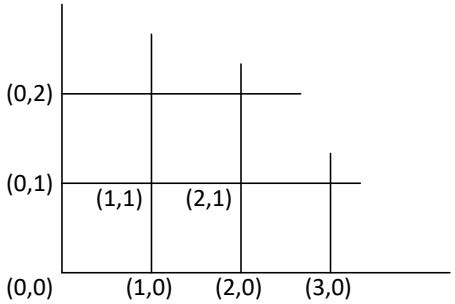
Boundary conditions:
Option 1:
Extend mesh and define ghost-cells. Set values in these to impose correct flow at boundary.

Option 2:
"Cut links" to outside cells and adjust source term.

B/C types:
- **fixed value (Dirichlet)**
- **fixed flux/ gradient (Neumann)**
- **Cauchy – simultaneous Dirichlet and Neumann**

2-D Diffusion problem

$$\left(\frac{\Gamma_w A_w}{dx_{WP}} + \frac{\Gamma_e A_e}{dx_{PE}} + \frac{\Gamma_s A_s}{dx_{SP}} + \frac{\Gamma_n A_n}{dx_{PN}} - S_P\right)\phi_{\mathbf{P}}$$
$$= \frac{\Gamma_w A_w}{dx_{WP}}\phi_{\mathbf{W}} + \frac{\Gamma_e A_e}{dx_{PE}}\phi_{\mathbf{E}} + \frac{\Gamma_s A_s}{dx_{SP}}\phi_{\mathbf{S}} + \frac{\Gamma_n A_n}{dx_{PN}}\phi_{\mathbf{N}} + S_u$$



E.g. for temperature:
$c_p T_P = c_n T_N + c_s T_S + c_w T_W + c_e T_E + c_{source}$
as c_n are constant, these linear equations can be combined into a matrix equation
$\mathbf{A}\, x = b$

$Node$	a_W	a_E	a_S	a_N	S_u	S_p	a_P
$0,0$	0	x	0	x	x	x	x
$0,1$	x	x	0	x	x	x	x
$0,2$	x	x	0	x	x	x	x
	.	.	.	.	.	.	.
I,J	.	.	.	.	.	.	.
	.	.	.	.	.	.	.
N,M	x	0	x	0	x	x	x

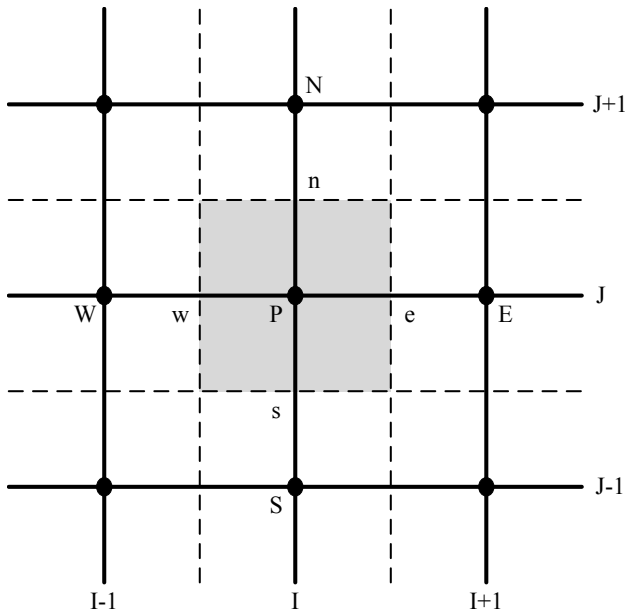


Fig 1: Cartesian 2-D grid

Solving single transport Equation

By using discretisation scheme (central difference in case of pure diffusion), *linear equations* can be developed between adjacent point. These equations can be combined to form sets of simultaneous linear equations, that fulfil the B/Cs and the transport equation (governing equation). Solving the resulting equations $\mathbf{A}\,x = b$ yields the solution.

Extra terms in transport equation can be solved in the same manner. I.e extra terms such as S_Φ, Convection, $\frac{d}{dt}$ (), etc... are added to linear equations.

Effectively the governing equations are solved by using piece-wise linear approximatins that relate the properties at a given point to the properties at the surrounding points. The closer the points are together (finer mesh), the less error is introduced by the linear approximations.

Discretisation Schemes for interfaces

Discretisation Schemes are used to find properties at the cell interfaces (e.g. at w between W and P)

For diffusion it is appropriate to use the *Central Difference* scheme:

$$\Phi_w = \frac{\Phi_W + \Phi_P}{2} \quad \text{(for unifom grid)}$$

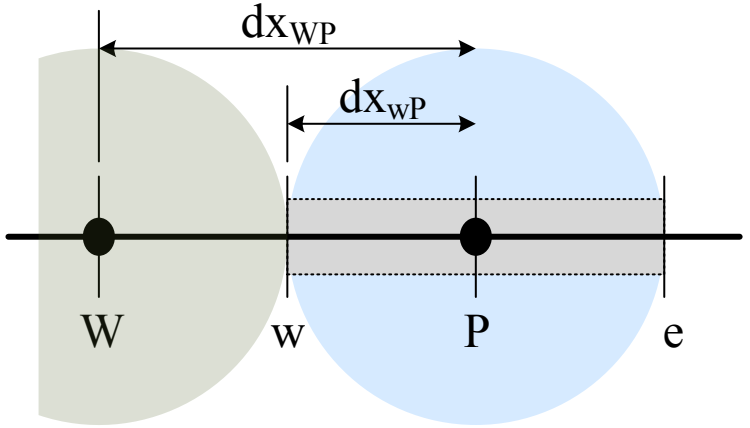
More advanced discretisation schemes are required for convection problems. Here properties are convected (blown) in the downstream direction. This effect is characterised by the Peclet number:

$$Pe = \frac{\rho\, u}{\frac{\Gamma}{dx}}$$

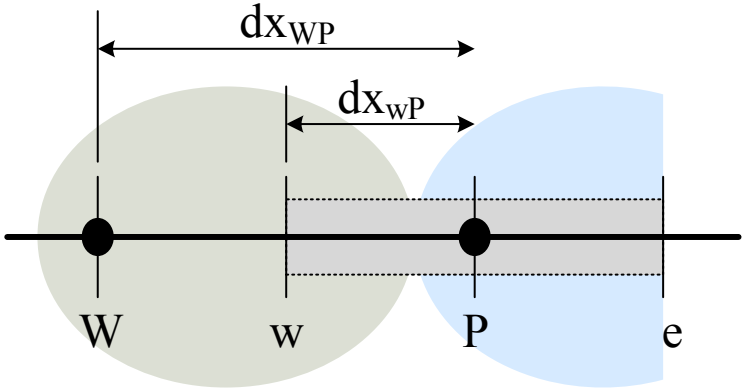
A discretisation scheme suitable for convection cases is the *Upwind* scheme:

$$\begin{aligned} \Phi_w &= \Phi_W \quad \text{if } u > 0 \\ \Phi_w &= \Phi_P \quad \text{if } u < 0 \end{aligned}$$

Higher order schemes, giving higher accuracy and better convergence are available.



Pure Diffusion $Pe \to 0$



Convection and Diffusion

$Pe \neq 0$

Coupled transport Equations

How do we efficiently solve the coupled transport equations for steady state?

$$\frac{d}{dx}(\rho u) + \frac{d}{dy}(\rho v) = 0 \quad \text{Continuity}$$

$$\frac{d}{dx}(\rho u u) + \frac{d}{dy}(\rho v u) = \frac{d}{dx}\left(\mu \frac{du}{dx}\right) + \frac{d}{dy}\left(\mu \frac{du}{dy}\right) - \underbrace{\frac{dp}{dx} + S_u}_{\text{Source Term}} \quad \text{x-momentum}$$

$$\frac{d}{dx}(\rho u v) + \frac{d}{dy}(\rho v v) = \frac{d}{dx}\left(\mu \frac{dv}{dx}\right) + \frac{d}{dy}\left(\mu \frac{dv}{dy}\right) - \underbrace{\frac{dp}{dy} + S_v}_{\text{Source Term}} \quad \text{y-momentum}$$

(In engineering flows the pressure gradient is the main source term. Despite typically being included in the source term, it has been shown above separately for clarity).

Solving these equations brings the following challenges:

- the convective terms of the momentum equation include nonliner terms (e.g. ρu^2)
- all three (or four in 3-D) equations are intrinsically coupled by the velocities
- resolving the role played by pressure is complex. It appears in both momentum equations, but there is no specific trasnport equation.
- due to nonlinearity and coupled equations we cannot create a single set of nonlienar equations.

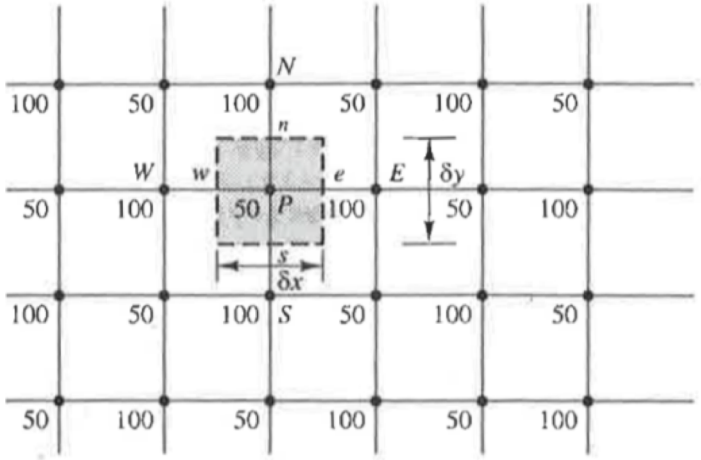
The SIMPLE algorithm is an iterative solution approaches to solve this.

Discretising the pressure and velocity

First the pressure and velocity fields need to be discretised. Here a problem arises if pressure and velocity is stored at the same nodes. E.g.Look at checker board pressure field.

$$\left.\frac{dp}{dx}\right|_P = \frac{p_e - p_w}{dx} = \frac{\left(\frac{p_E+p_P}{2}\right) - \left(\frac{p_P+p_W}{2}\right)}{dx}$$

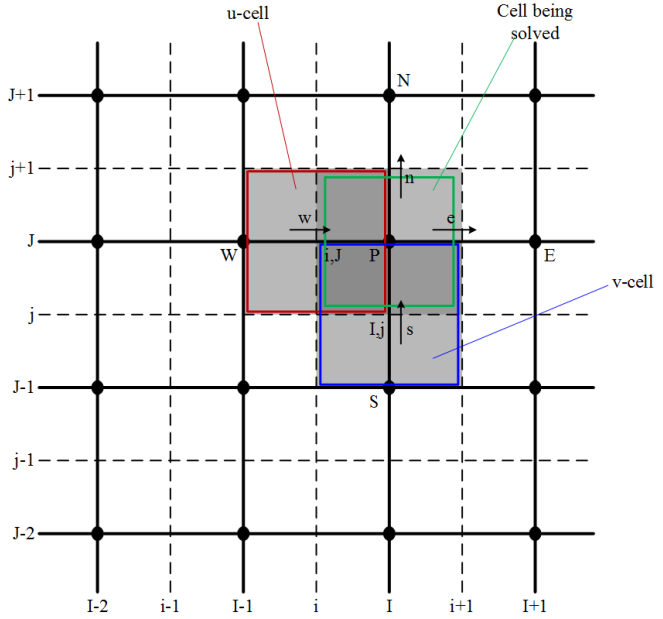
$$= \frac{p_E - p_W}{2\,dx}$$



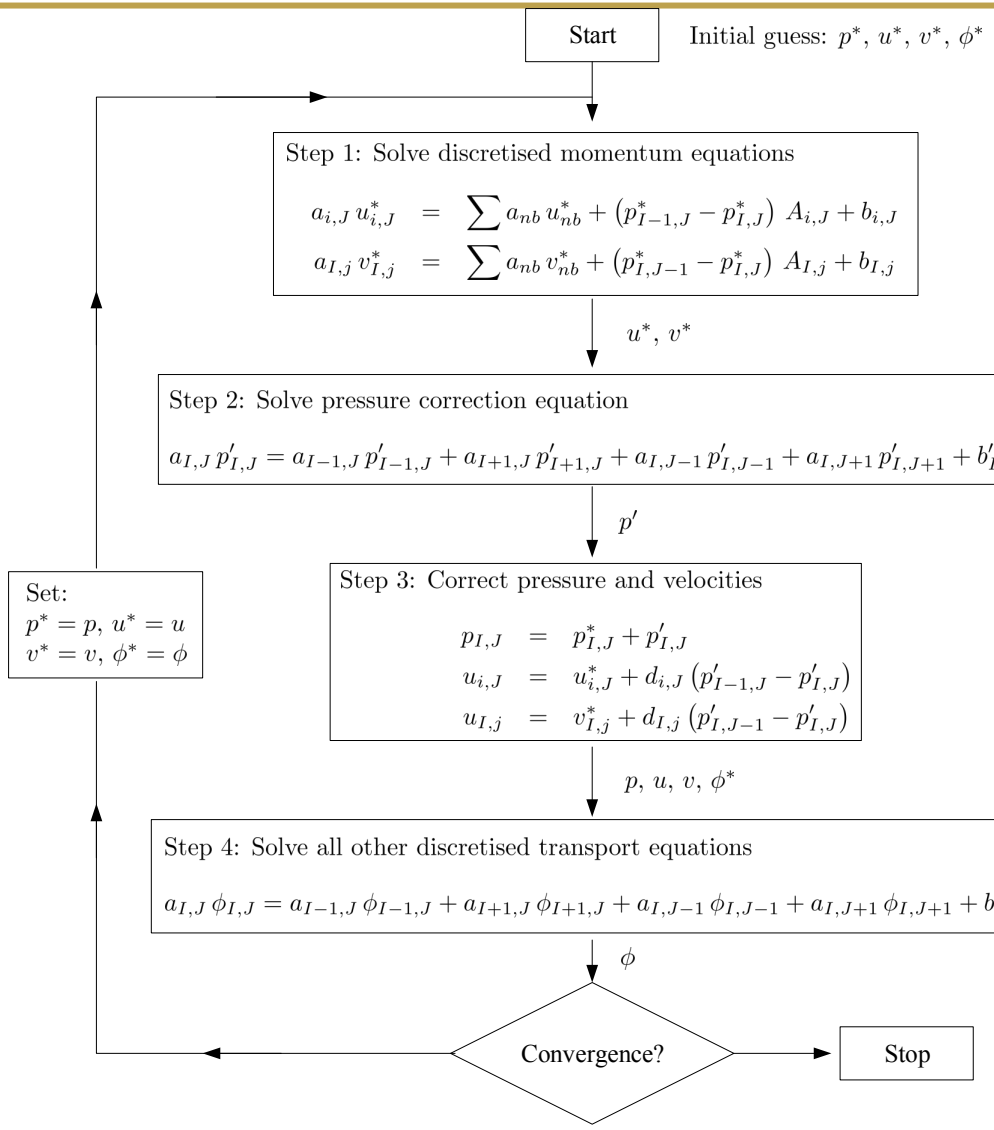
Obviously this field has $\frac{dp}{dx} \neq 0$ and $\frac{dp}{dy} \neq 0$, but with linear discretisation one obtains $\frac{dp}{dx} = \frac{dp}{dy} = 0$.

This problem is overcome by using a staggered grid. Here pressure, p, and other scalars are stored at the nodes and velocity, u & v, are stored at the interfaces.

This is especially convenient, as for the finite volume method, you want to calculate the fluxes (velocities) across the cell boundaries when solving the transport equations.



The SIMPLE Algorithm



SIMPLE or Semi-Implicit Method for Pressure Linked Equations, is a guess and correct procedure for calculating pressure on the staggered field.

1. Using guessed/previous pressure field p^*, solve the momentum equations to yield the velocity field u^* and v^*.
2. Define corrections for pressure and velocities, $p = p^* + p'$, $u = u^* + u'$, and $v = v^* + v'$
   Solve pressure correction equation:
   - create equations for u' and v' by subtracting localised equations of u and $u*$ and same for v
   - with some simplifications it can be shown that $u_{i,J} = u^*_{i,J} + d_{i,J}\left(p'_{I-1,J} - p'_{I,J}\right)$ (and same for v) where $d_{i,J} = \frac{A_{i,J}}{a_{i,J}}$
   - these equations are combined using the continuity equation and solved for p'

   (this equation usually requires some under relaxation to maintain stability)
3. Update p, u, and v equation
4. Solve other transport equations

More details on solution algorithms are available in:Patankar, S.V. (1980). Numerical Heat Transfer and Fluid Flow, Hemisphere Publishing Corporation, Taylor & Francis Group, New York.

SIMPLE Equations

Step 1: Solve discretised momentum equations

Initial guess: p^*, u^*, v^*, ϕ^*

$$a_{i,J}\, u^*_{i,J} = \sum a_{nb}\, u^*_{nb} + \left(p^*_{I-1,J} - p^*_{I,J}\right) A_{i,J} + b_{i,J}$$

$$a_{I,j}\, v^*_{I,j} = \sum a_{nb}\, v^*_{nb} + \left(p^*_{I,J-1} - p^*_{I,J}\right) A_{I,j} + b_{I,j}$$

u^*, v^*

p'

p, u, v, ϕ^*

Step 2: Solve pressure correction equation

$$a_{I,J}\, p'_{I,J} = a_{I-1,J}\, p'_{I-1,J} + a_{I+1,J}\, p'_{I+1,J} + a_{I,J-1}\, p'_{I,J-1} + a_{I,J+1}\, p'_{I,J+1} + b'_{I,J}$$

ϕ

Step 3: Correct pressure and velocities

$$p_{I,J} = p^*_{I,J} + p'_{I,J}$$

$$u_{i,J} = u^*_{i,J} + d_{i,J}\left(p'_{I-1,J} - p'_{I,J}\right)$$

$$u_{I,j} = v^*_{I,j} + d_{I,j}\left(p'_{I,J-1} - p'_{I,J}\right)$$

Set:

$p^* = p$, $u^* = u$

$v^* = v$, $\phi^* = \phi$

Step 4: Solve all other discretised transport equations

$$a_{I,J}\, \phi_{I,J} = a_{I-1,J}\, \phi_{I-1,J} + a_{I+1,J}\, \phi_{I+1,J} + a_{I,J-1}\, \phi_{I,J-1} + a_{I,J+1}\, \phi_{I,J+1} + b'_{I,J}$$

Other Solution Algorithms

SIMPLER - SIMPLE-Revised

Has a better correction Algorithm compared to the SIMPLE algorithm

SIMPLEC - SIMPLE-Consistent

Uses different velocity correction that includes extra terms

PISO Pressure Implicit with Splitting of Operators

Originally an unsteady solver that has been adapted for iterative solution of steady state problems. Effectively this is the simple algorithm with an added correction step.

Generally solutions can be found with all algorithms. The more complex ones are better at updating the coupling effects. The main difference is speed of convergence. More complex algorithms, with up to 30 % more calculations can converge 30-50 % more quickly. This is very subjective and also depends on problem and relaxation.

The SIMPLE algorithm in OF

Looking at SIMPLE Algorithm in OpenFOAM:

- Change to application directory `$ app`
  `solvers` contains solver source code, `test` contains code to test/validate solved, `utilities` is utility source code
- Have a look at `icoFoam.C`
  - Line 60: Transport equation for momentum is defined
  - Line 69: Transport equation for is solved with `\grad(p)`
  - etc...
- Have a look at `simpleFoam.C`
  You will find the code for the velocity and pressure equations in `UEqn.H` and `pEqn.H`
  - Line 63 & 64: Files containing velocity and pressure equation are called.
  - Line 67: Inclusion of the turbulence model. I.e. after SIMPLE loop is complete effect of turbulence is updated.
- If you want to build your own solver, use existing OpenFOAM C^{++} libraries and use these to solve continuum mechanics equations.

Feel free to look at other solvers in your spare time...

More OpenFOAM solvers

For other applications, (e.g. thesis projects, work in industry) there are many other pre-build OpenFOAM solvers. Here are some examples, but the list grows every day. Full list is available in `foam/applications/solvers/...`

- Single and Multiple Rotating Reference Frame FOAM.
  Generally used for turbomachinery
  - Solve flow in rotating frames
  - Correction the momentum to include Centrifugal and Coriolis forces
  - Sliding mesh interfaces
- porous media
- LES and DNS
- heatTransfer
- multiphase
  - Eulerian - Lagrnagian simulation. Track liquid packets that flow through domain.
  - Volume of Fluid (VoF) - Solve for mixture faction
- combustion and chemical reactions
- solidMechanics
- electromagnetics
- financial
- many more...

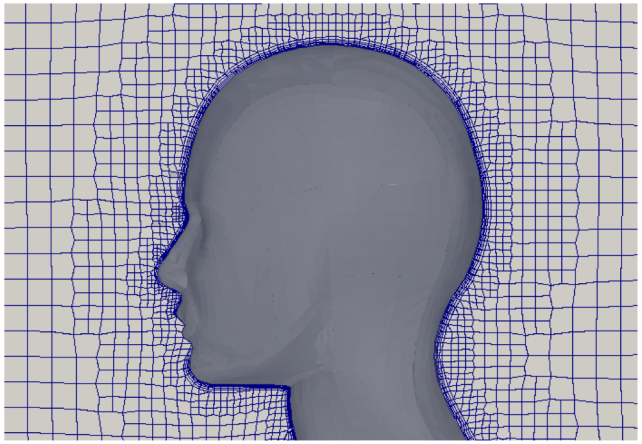
Other OpenFOAM Tools

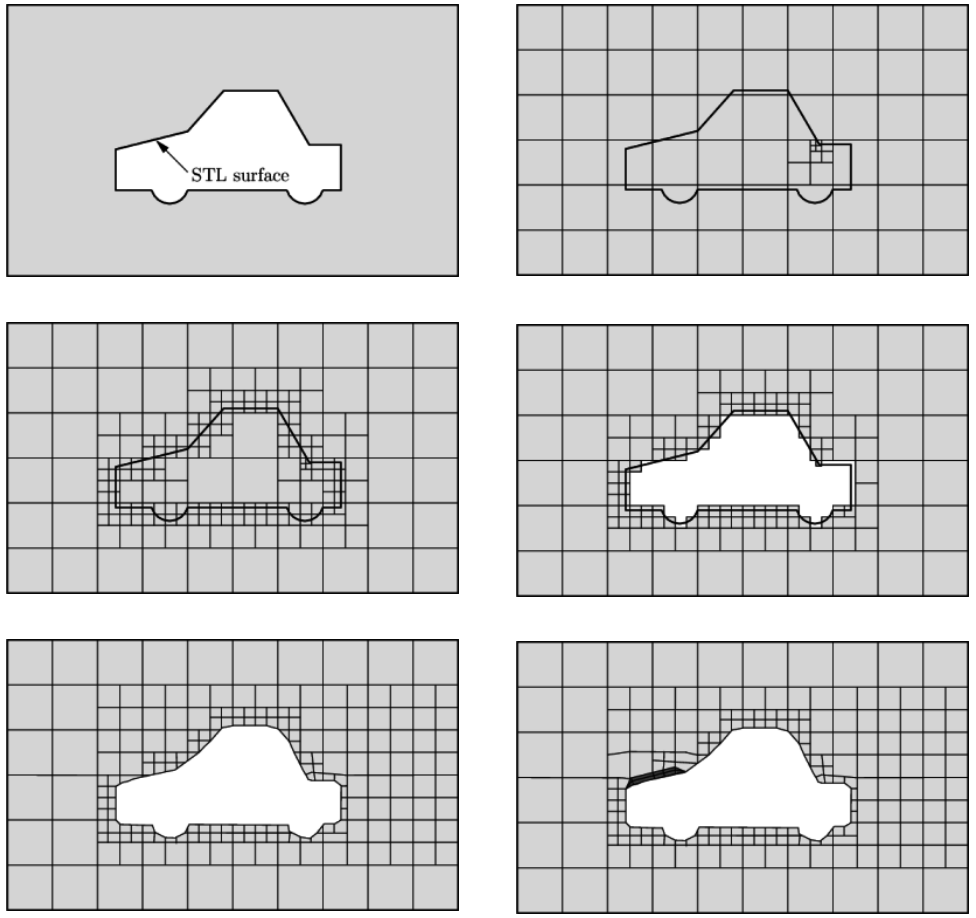
- Useful post-processing utilities. See OF Userguide section 6.2
  Typical usage, for example `simpleFoam -postProcess -func yPlus`
  - Mach
  - yPlus
  - probes
  - and several runtime postprocessing fucntions.
- Parallel simulations (multiple CPUs) See OF Userguide section 3.4
  - Decompose mesh: Separate simulation into compnents for each CPU
  - Running decomposed case
  - Post-processing and Reconstruction
- Snappy Hex Mesh

Snappy Hex Mesh

SnappyHexMesh is a automated grid generator that is part of the OpenFoam package. If youa re interested, read up in the OpenFoam userguide.
The process:

- Create base mesh
- Refine base mesh
- Remove unused cells
- Snap mesh to surface
- Add layers



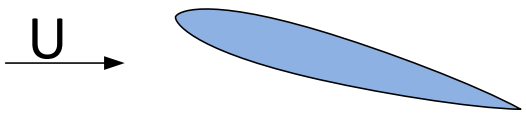


Source: OpenFoam UserGuide

How good is CFD (RANS)?

Real World with Real Physics

So complicated it warps your mind



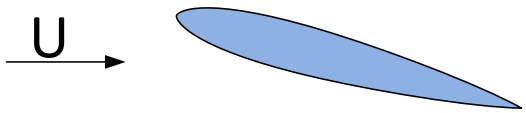
CFD (RANS) with medium complexity physics + turbulence model

Flow governed by

$$\frac{d\rho}{dt} + div(\rho \mathbf{u}) = 0$$

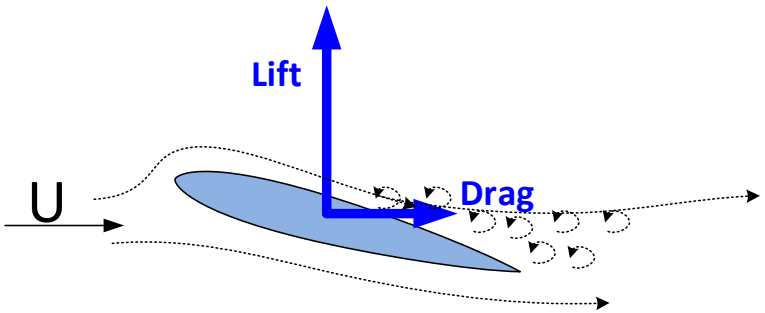
$$\frac{d(\rho u)}{dt} + div(\rho u \mathbf{u}) = -\frac{dp}{dx} + div(\mu\, grad\, u) + S_{Mx}$$

$$\frac{d(\rho v)}{dt} + div(\rho v \mathbf{u}) = -\frac{dp}{dy} + div(\mu\, grad\, v) + S_{My}$$



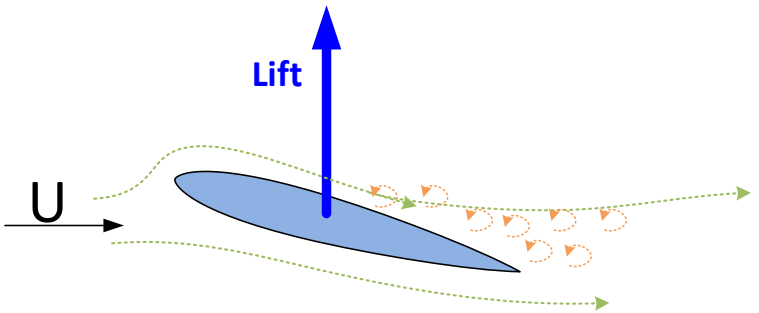
Real Experiment

- Lift
- Drag
- Complex flow field



Numerical Experiment

- Lift → **OK**
- Drag → **OK**
- Complex flow field → **Large Eddies Resolved**
  Small eddies and turbulence modelled



The end …

(of MECH4480 incompressible flow)