Introductory OpenFOAM Workshop

Overview of the math inside

Timofey Mukha

KTH Engineering Mechanics

The European Centre of Excellence for Engineering Applications - EXCELLERAT

13 Partners

6 Core Codes

4 Application Areas

Aerospace, Automotive, Energy, Safety Applications

11 Use Cases

Core Codes and Use Cases

Codes	Application Areas	Use Cases
Alya	Automotive, Aerospace	Emission prediction of internal combustion (IC) and gas turbine (GT) engines
	Aerospace	Active flow control of aircraft aerodynamics including synthetic jet actuators
	Aerospace	Coupled simulation of fluid and structure mechanics for fatigue and fracture
AVBP	Aerospace, Energy	Combustion instabilities and emission prediction
	Safety Applications	Explosion in confined spaces
Coda	Aerospace	Design process and simulation of fully equipped aero planes
	Aerospace	CFD coupling with computational structural mechanics including elastic effects
Fluidity	Energy	Tidal Energy Generation Modelling of drag and sedimentation effects of tidal turbines
FEniCS	Automotive, Aerospace	Adjoint optimization in external aerodynamics shape optimization
NEK5000	Aerospace	Wing with three-dimensional wing tip
	Aerospace	High fidelity simulation of rotating parts

Overview

- Modest goal: set the stage so that one can understand what some of the options in OpenFOAM stand for.

- A bird-eye view of what is happening in the solver.

- By no means an exhaustive coverage of the topic, not even a comprehensive introduction.

Two things we need to look at

- Transforming a PDE to a linear system of equations using the Finite Volume Method.

- Solution procedure for the Navier-Stokes equations (pressure-velocity coupling).

FVM bird-eye view

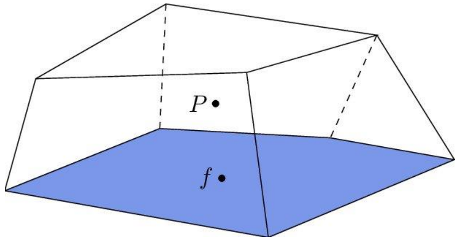
PDE in differential form
$\nabla \cdot (\rho \boldsymbol{u} \phi)$

$\xrightarrow{\text{Integrate}}$

PDE in integrated form
$\int \nabla \cdot (\rho \boldsymbol{u} \phi) \mathrm{d}V$

$\xrightarrow{\text{Gauss theorem}}$

Volume integrals are now surface integrals
$\oint (\rho \boldsymbol{u} \phi) \cdot \boldsymbol{n} \mathrm{d}s$



↓ The volume is a convex polyhedron

Surface integrals can be split across the volume faces

$$\sum_f \oint (\rho \boldsymbol{u} \phi) \cdot \boldsymbol{n} \mathrm{d}s_f$$

← Face values are approximated at face centres

Surface integrals replaced by evaluation at face centres

$$\sum_f (\rho \phi \boldsymbol{u} \cdot \boldsymbol{n})_f S_f$$

← Face values computed from volume centroid values

Discrete system in space

↓ Select time integration scheme

Fully discrete

Obtaining the integral form

$$\frac{\partial \phi}{\partial t} + \nabla \cdot (\rho \boldsymbol{u} \phi) = 0$$

$$\int \frac{\partial \phi}{\partial t} \mathrm{d}V + \int \nabla \cdot (\rho \boldsymbol{u} \phi) \mathrm{d}V = 0$$

$$\int \frac{\partial \phi}{\partial t} \mathrm{d}V + \oint \rho \phi (\boldsymbol{u} \cdot \boldsymbol{n}) \mathrm{d}S = 0$$

Gauss-Otrogradsky theorem

$$\int \nabla \cdot \mathbf{u} \mathrm{d}V = \oint \boldsymbol{u} \cdot \boldsymbol{n} \mathrm{d}S$$

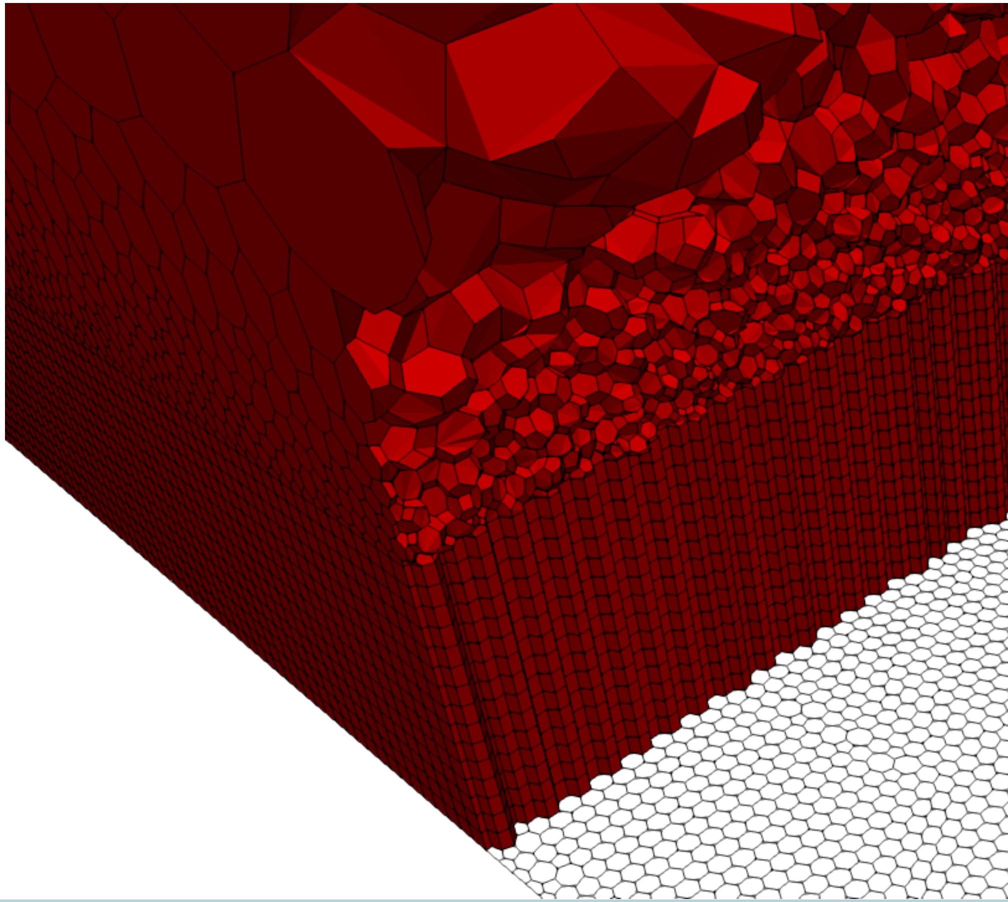
- Can be applied to gradient and diffusion terms as well.
- We replace the differential operators with the evaluation of the field at the surface of the volume.

Connection to meshing

- The integral form is valid for an arbitrary volume.

- When we mesh, we split the domain into many volumes, for each of them the integral equations are valid.

- In principle, as long as we are consistent in how we compute the surface integrals, our method is conservative.

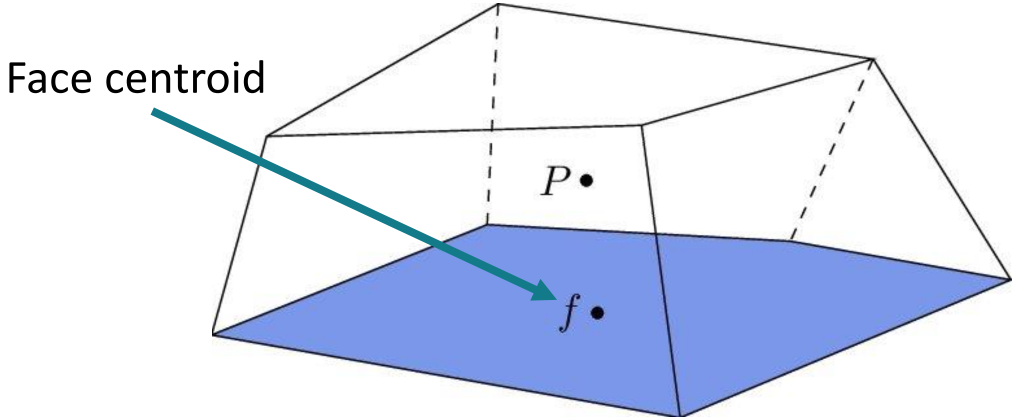


Towards a semi-discrete formulation

$$\int \frac{\partial \phi}{\partial t} \mathrm{d}V + \oint \rho\phi(\boldsymbol{u}\cdot\boldsymbol{n})\mathrm{d}S = 0$$

$$\int \frac{\partial \phi}{\partial t} \mathrm{d}V + \sum_f \oint \rho\phi(\boldsymbol{u}\cdot\boldsymbol{n})\mathrm{d}S_f = 0$$

$$\int \frac{\partial \phi}{\partial t} \mathrm{d}V + \sum_f \rho_f\phi_f(\boldsymbol{u}_f\cdot\boldsymbol{n}_f)S_f = 0$$

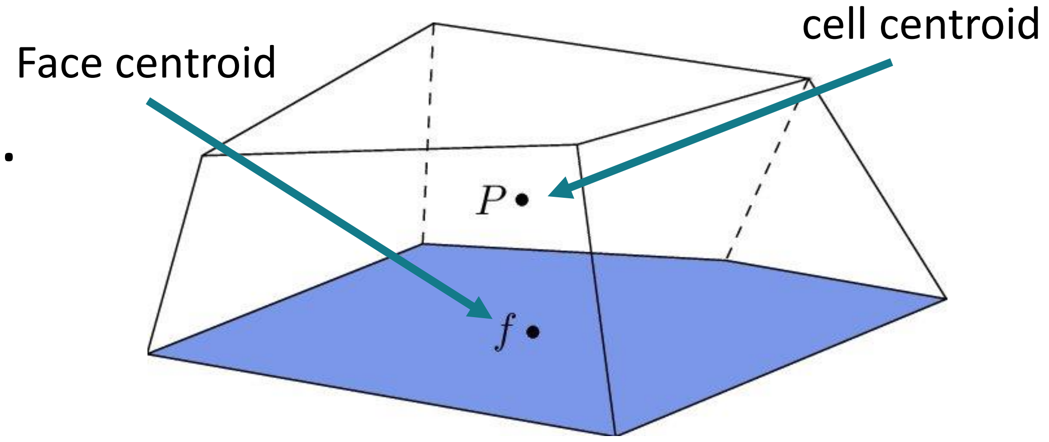


- Assume the volume is a convex poly.
- We replace the integral with a 2nd order accurate approximation.

Computing the face centroid values

- At this point, we need a recipe for computing the unknowns at the faces.

- In collocated FVM, the unknowns we solve for are located at the *cell* centroids.



- The task is then to compute the values at the *face* centroid given the values at the centroids of the two cells sharing the face!

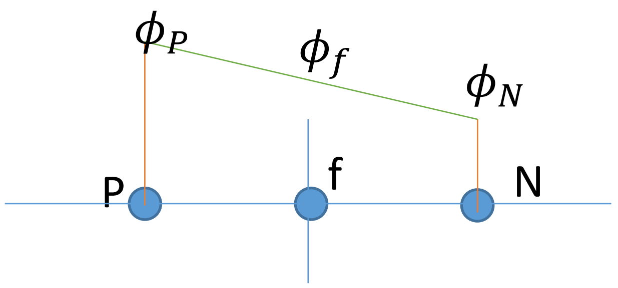
Computing the face centroid values

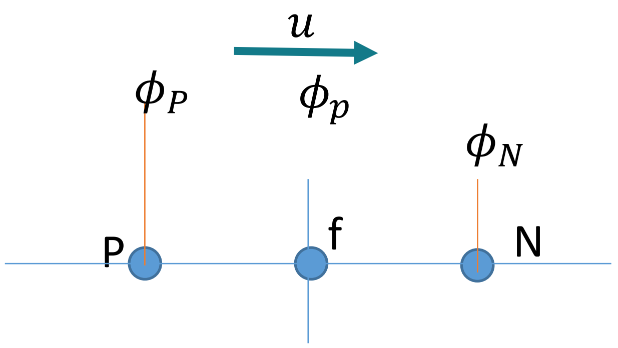
- The most obvious option is linear interpolation, 2nd order accurate.

- However, for convective fluxes it introduces oscillations.

- Can use the idea of upwinding instead.

- Many options for interpolating the convective flux exist. Often the most important numerical choice in the simulation!





Towards a discrete formulation

$$\int \frac{\partial \phi}{\partial t} \mathrm{d}V + \sum_f \rho_f \phi_f (\boldsymbol{u}_f \cdot \boldsymbol{n}_f) S_f = 0$$

$$\int_{t^n}^{t^{n+1}} \int \frac{\partial \phi}{\partial t} \mathrm{d}V \mathrm{d}t + \int_{t^n}^{t^{n+1}} \sum_f \rho_f \phi_f (\boldsymbol{u}_f \cdot \boldsymbol{n}_f) S_f = 0$$

$$\int \int_{t^n}^{t^{n+1}} \frac{\partial \phi}{\partial t} \mathrm{d}t \mathrm{d}V + \int_{t^n}^{t^{n+1}} \sum_f \rho_f \phi_f (\boldsymbol{u}_f \cdot \boldsymbol{n}_f) S_f = 0$$

Towards a discrete formulation

$$\int (\phi^{n+1} - \phi^{n})\mathrm{d}V + \int_{t^n}^{t^{n+1}} \sum_{f} \rho_f \phi_f (\boldsymbol{u}_f \cdot \boldsymbol{n}_f) S_f \,\mathrm{d}t = 0$$

$$(\phi_P^{n+1} - \phi_P^{n})V + \int_{t^n}^{t^{n+1}} \sum_{f} \rho_f \phi_f (\boldsymbol{u}_f \cdot \boldsymbol{n}_f) S_f \,\mathrm{d}t = 0$$

At this point, we have to select a time-integration scheme

Towards a discrete formulation

$$(\phi_P^{n+1} - \phi_P^n)V + \int_{t^n}^{t^{n+1}} \sum_f \rho_f \phi_f (\boldsymbol{u}_f \cdot \boldsymbol{n}_f) S_f \,\mathrm{dt} = 0$$

Spatial interpolation

$\longrightarrow$

Implicit time-scheme

$$a_p \phi_P^{n+1} + \sum_f a_n \phi_n^{n+1} = \mathrm{s_p}$$

$\downarrow$

Solve using an efficient algorithm

- ”Explicit Euler”: use value at t^n.
- ”Implicit Euler”: use value at t^{n+1}.
- ”Crank-Nicolson”: mean of t^n and t^{n+1}
- Some linear combination of t^n and t^{n+1}.
- 2nd order backward-differencing: use t^{n-1}, t^n and t^{n+1}

Time integration considerations

- Explicit or implicit: the latter means we have to solve a linear system at each time-step.

- Order of accuracy.

- Numerical stability, and implications for the time-step.

Navier-Stokes equations

$$\nabla \cdot u = 0$$

$$\frac{\partial u}{\partial t} + \nabla \cdot (u \otimes u) = -\nabla p + \nabla \cdot (\nu \nabla u)$$

- The first equation admits infinite number of solutions.
- No equation for pressure.
- The equation is non-linear, “velocity transports itself”.

Linearization

$$\nabla \cdot u = 0$$
$$\frac{\partial u}{\partial t} + \nabla \cdot (u \otimes u) = -\nabla p + \nabla \cdot (\nu \nabla u)$$

$$\nabla \cdot u = 0$$
$$\frac{\partial u}{\partial t} + \nabla \cdot (\boxed{\phi} \otimes u) = -\nabla p + \nabla \cdot (\nu \nabla u)$$

- The question is how to get ϕ.
- Simple solution: previous time-step. But this is 1^{st} –order accurate.
- We can also establish an iterative procedure. But this is additional cost.

Semi-discrete form

$$\frac{\partial u}{\partial t} + \nabla \cdot (\phi \otimes u) = -\nabla p + \nabla \cdot (\nu \nabla u)$$

$a_p u_p = H(u) - \nabla p$, where
$H(u) = \sum a_n u_n + s_p$

From here

$$u_p = \frac{H(u)}{a_p} - \nabla \mathrm{p}/a_p$$

interpolation

$$\nabla \cdot u = 0$$

$$\sum_f (u_f \cdot n_f) S_f = 0$$

Pressure equation

$$\sum_f \left(\left(\frac{H(u)}{a_p} \right)_f \cdot n_f \right) S_f = \sum_f \left(\frac{1}{a_p} \right)_f \nabla \mathrm{p_f} \cdot n_f S_f$$

Pressure-velocity coupling

1. Solve momentum equation with a guessed pressure field.

2. Assemble $H(u)$ and solve the pressure equation.

3. Correct velocity via $u_p = \frac{H(u)}{a_p} - \nabla \mathrm{p}/a_p$.

Note: given a new velocity field we can recompute $H(u) = \sum a_n u_n + u_p^n/\Delta t$.

- We can update the values of u_n.
- But also the coefficients a_n that come from ϕ, i.e. the linearization!

We arrive to an iterative procedure. Depending on what we update, the procedure s slightly different.

SIMPLE, simpleFoam, steady case

1. Solve momentum equation with a guessed pressure field.

2. Assemble $H(u)$ and solve the pressure equation.

3. Correct velocity via $a_p u_p = H(u) - \nabla p$.

Loop

- We loop across 1-3 until convergence.
- We update ϕ after obtaining a new velocity field to resolve the non-linearity.
- Under-relaxation can be used to stabilize the procedure.

PISO, pisoFoam, unsteady case

1. Solve momentum equation with a guessed pressure field.
2. Assemble $H(u)$ and solve the pressure equation.
3. Correct velocity via $a_p u_p = H(u) - \nabla p$.

Loop

- We do step 1 once.
- And loop across 2-3.
- Non-linearity error assumed to have small contribution for small time-steps.

PIMPLE, pimpleFoam, unsteady case

1. Solve momentum equation with a guessed pressure field.
2. Assemble $H(u)$ and solve the pressure equation.
3. Correct velocity via $a_p u_p = H(u) - \nabla p$.

Inner loop

Outer loop

- We loop across 1-3, ”outer iterations”.
- Within each outer iteration we also loop ”2-3” inner iterations.
- A hybrid, allowing for a larger time-step since we have iterations to resolve non-linearity.

Pressure-velocity coupling

- The parameters of the iterations are set in `fvSolution`.

```
PIMPLE
{
    nOuterCorrectors 1;
    nCorrectors      3;
    nNonOrthogonalCorrectors 1;
    pRefCell         1001;
    pRefValue        0;
}
```

- Quite a bit of subtleties that we haven't covered and that are actually difficult to fully understand.
    - Rhie-Chow interpolation.
    - Boundary conditions.
    - Cycle to reduce non-orthogonality errors.