Rhie-Chow Interpolation

Mustafa Bhotvawala

- A **grid** is how variables are stored on a **mesh**. There are two types of these:

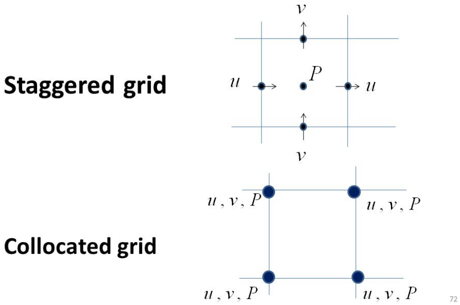

Figure: Collocated vs Staggered [1]

- In a collocated grid, all flow variables are stored at cell centres

Rhie-Chow Interpolation

- If you recall, the general discretization equation is:

$$a_p u_p = \sum a_i u_i + S_{up} + S_u$$

- The source term due to pressure evaluates to:

$$S_{up} = (P_W - P_E)\Delta y$$

- Hence, P_P itself does not contribute to u_P. This causes the checkerboard problem that we want to avoid by using staggered grids
- Staggered grids have velocity variables stored on cell faces and the pressure stored at cell centres
- But staggered grids are difficult to implement on unstructured meshes and add another layer of complexity to the code
- For these reasons, OpenFOAM uses collocated grids
- But how do they get around the checkerboard problem?

Rhie-Chow Interpolation

- This is by using **Rhie-Chow interpolation**, which is a correction applied to collocated grids to avoid checkerboard oscillations
- To recap, the N-S equations are:

$$\frac{\partial(\rho u_i)}{\partial t} + \frac{\partial(\rho u_i u_j)}{\partial x_j} = -\frac{\partial p}{\partial x_i} + \frac{\partial \tau_{ij}}{\partial x_j} \tag{1}$$

- The convection term $\frac{\partial(\rho u_i u_j)}{\partial x_j}$ is slightly different in OpenFOAM

```
fvm::div(phi, U)
```

- Where does `phi` (ϕ) come from?

Rhie-Chow Interpolation

- The convection term is discretized using the Gauss theorem as

$$\int_V \nabla \cdot (\mathbf{U}\underline{\mathbf{U}})dV = \int_S (\mathbf{U}\underline{\mathbf{U}})_f \cdot \mathbf{n}dS$$

- The convection term is non-linear in velocity. Let's distinguish them into two for now, **U** and $\underline{\mathbf{U}}$

$$\int_S (\mathbf{U}\underline{\mathbf{U}})_f \cdot \mathbf{n}dS = \sum_i \mathbf{U}_{f,i}\underline{\mathbf{U}}_{f,i} \cdot \mathbf{Sf}_i = \sum_i \mathbf{U}_{f,i}\phi_i$$

- We consider the scalar product of $\underline{U}_{f,i} \cdot \mathbf{Sf}_i$ as ϕ_i, and in general:

$$\phi = \underline{U}_f \cdot \mathbf{Sf}$$

- Hence, the flux ϕ is a scalar obtained from the dot product of the **face velocity that is kept constant during the Poisson equation** and the area vector at the face

Rhie-Chow Interpolation

- Rhie-Chow propose an intermediate velocity where the effect of pressure is removed, which is then interpolated from the cell nodes to cell faces
- If you recall,
$$\mathbf{U}^* = A^{-1}H - A^{-1}\nabla p$$
- One can see that $A^{-1}H$ is the component of the guess velocity $\mathbf{U}^*$ that does not contain pressure
- OpenFOAM interpolates this velocity $A^{-1}H$ (`HbyA`) to the face via a `flux` function as per Rhie-Chow:

```
surfaceScalarField phiHbyA("phiHbyA", fvc::flux(HbyA))
```

Rhie-Chow Interpolation

- The Poisson equation for pressure is eventually solved:

```
fvm::laplacian(rAtU(), p) == fvc::div(phiHbyA)
```

- The flux `phiHbyA` on the RHS is **just the flux of the HbyA term**. There is no pressure term in the flux. This uncouples the laplacian term in the LHS from any pressure term on the RHS.
- The Rhie-Chow method is completed by the snippet:

```
U = HbyA - rAtU()*fvc::grad(p)
```

- The gradient term is calculated on the cell faces and is used to correct the pressure-free velocity `HbyA`, which gives us the final velocity U
- This is an 'implicit' implementation of the Rhie-Chow method that may be different from how it's explained in theory [2]

References

1. Ohwada, T., Asinari, P. and Yabusaki, D., 2011. Artificial compressibility method and lattice Boltzmann method: Similarities and differences. Computers & Mathematics with Applications, 61(12), pp.3461-3474.
2. Ferziger, J.H., Perić, M. and Street, R.L., 2002. Computational methods for fluid dynamics (Vol. 3, pp. 196-200). Berlin: springer
3. Kärrholm, F.P., 2006. Rhie-chow interpolation in openfoam. Department of Applied Mechanics, Chalmers University of Technology: Goteborg, Sweden