Finite Volume Discretisation in OpenFOAM

Best Practice Guidelines

Hrvoje Jasak

`h.jasak@wikki.co.uk`

Wikki Ltd, United Kingdom

Outline

Objective

- Review the best practice guidelines for the Finite Volume (FV) discretisation in OpenFOAM and compare it with commercial CFD solvers
    - Background on discretisation
    - Default settings on dominantly hex and dominantly tet meshes

Topics

- Background
- Discretisation requirements: gradient scheme; convection; diffusion
- Proposed default settings: hexahedral meshes
- Proposed default settings: tetrahedral meshes
- Summary

Background

Best Practice Guidelines in CFD

- Commercial CFD codes offer robust set of default settings for the FVM: make the code run on a bad mesh and by inexpert users
- Priority is in producing a result: substantial improvements in solution quality and accuracy is possible
- ... but only for an expert user!
- Default settings are extremely important and change only after large validation and robustness testing campaigns

Default Settings in OpenFOAM

- ... are practically non-existent: the code is written by experts and defaults are changed on a whim
- Some tutorials have settings appropriate for the case, but not recommended in general
- To remedy this, we need automatic test loops with $5000+$ validation cases
- Improvements are in the pipeline: community effort and validation harness

Discretisation

Finite Volume Discretisation

- Main concerns of FVM accuracy are mesh structure and quality and choice of discretisation schemes
- Mesh structure determines the choice of appropriate gradient calculation algorithm
- For transport of bounded scalars, it is essential to use bounded differencing schemes: both for convection and diffusion

Gradient Scheme

Gauss Gradient Scheme

- Gradient calculated using integrals over faces

$$\int_{V_P} \nabla\phi \, dV = \oint_{\partial V_P} d\mathbf{s}\,\phi = \sum_f \mathbf{s}_f \phi_f$$

- Evaluate the face value of ϕ from cell centre values

$$\phi_f = f_x \phi_P + (1 - f_x)\phi_N$$

where $f_x = \overline{fN}/\overline{PN}$

- Expression is second-order accurate only if ϕ_f is the face centre value
- Accurate on hexahedral meshes, but looses accuracy on tetrahedra: large skewness error

Gradient Scheme

Least Squares Gradient: Second Order Accuracy On All Meshes

- Consider cell centre P and a cluster of points around it N. Fit a plane:

$$e_N = \phi_N - (\phi_P + \mathbf{d}_N \bullet (\nabla \phi)_P)$$

- Minimising the weighted error: second-order accuracy on all meshes

$$e_P^2 = \sum_N (w_N e_N)^2 \quad \text{where} \quad w_N = \frac{1}{|\mathbf{d}_N|}$$

  yields a second-order **least-square form of gradient**:

$$(\nabla \phi)_P = \sum_N w_N^2 \, \mathbf{G}^{-1} \bullet \mathbf{d}_N (\phi_N - \phi_P)$$

- $\mathbf{G}$ is a 3×3 symmetric matrix:

$$\mathbf{G} = \sum_N w_N^2 \, \mathbf{d}_N \mathbf{d}_N$$

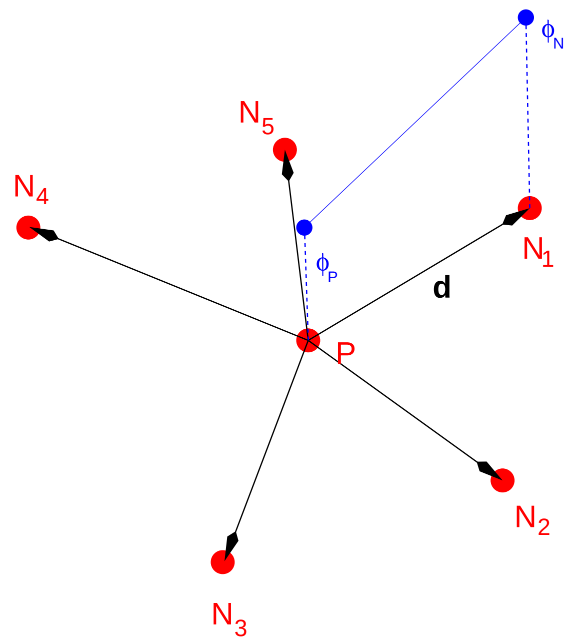
Gradient Limiter

Cell- and Face-Limited Gradient

- Gradient reconstruction may lead to local over- or under-shoots in reconstructed field:

$$min_N(\phi_N) \leq \phi_P + \mathbf{d}_N \bullet (\nabla \phi)_P \leq max_N(\phi_N)$$

- This is important for bounded variables, especially when gradients are used in further discretisation or coupling terms
- Solution: based on the gradient, calculate min and max neighbourhood value and apply gradient limiter to preserve bounds in cell centres



Convection Discretisation

Convection Operator

- Convection operator splits into a sum of face integrals (integral and differential form)

$$\oint_S \phi(\mathbf{n}\bullet\mathbf{u})dS = \int_V \nabla\bullet(\phi\mathbf{u})\,dV = \sum_f \phi_f(\mathbf{s}_f\bullet\mathbf{u}_f) = \sum_f \phi_f F$$

  where ϕ_f is the face value of ϕ and

$$F = \mathbf{s}_f\bullet\mathbf{u}_f$$

  is the **face flux**: measure of the flow through the face

- Simplest face interpolation: **central differencing**. Second-order accurate, but causes oscillations

$$\phi_f = f_x\phi_P + (1 - f_x)\phi_N$$

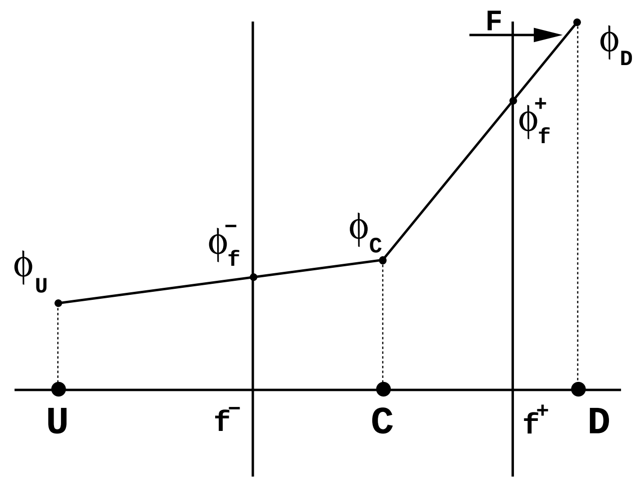
- **Upwind differencing**: taking into account the transportive property of the term: information comes from upstream. No oscillations, but smears the solution

$$\phi_f = max(F, 0)\,\phi_P + min(F, 0)\,\phi_N$$

Face Interpolation

Face Interpolation Scheme for Convection

- In order to close the system, we need a way of evaluating ϕ_f from the cell values ϕ_P and ϕ_N: **face interpolation**
- In order to preserve the iteration sequence, the convection operator for bounded (scalar) properties must preserve boundedness
- There exists a large number of schemes, trying to achieve good accuracy while preserving boundedness: *e.g.* TVD, and NVD families: $\phi_f = f(\phi_P, \phi_N, F, \ldots)$



- Special differencing schemes for strictly bounded scalars: switching to UD when a variable violates the bound. Example: Gamma01

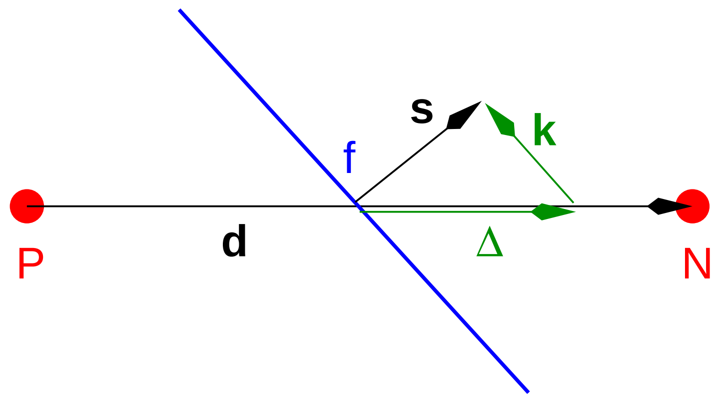
Diffusion Discretisation

Diffusion Operator and Mesh Non-Orthogonality

- Diffusion term is discretised using the Gauss Theorem

$$\oint_S \gamma(\mathbf{n} \bullet \nabla \phi) dS = \sum_f \int_{S_f} \gamma(\mathbf{n} \bullet \nabla \phi)\, dS = \sum_f \gamma_f\, \mathbf{s}_f \bullet (\nabla \phi)_f$$

- Evaluation of the face-normal gradient. If $\mathbf{s}$ and $\mathbf{d}_f = \overline{PN}$ are aligned, use difference across the face. For non-orthogonal meshes, a correction term may be necessary



$$\mathbf{s}_f \bullet (\nabla \phi)_f = |\mathbf{s}_f| \frac{\phi_N - \phi_P}{|\mathbf{d}_f|} + \mathbf{k}_f \bullet (\nabla \phi)_f$$

Diffusion Operator

Limiting Non-Orthogonal Correction in a Laplacian

- Decomposition of face gradient into “orthogonal component” and “non-orthogonal correction” depends on mesh quality: mesh non-orthogonality is measured from $\overline{PN}$ and $\mathbf{s}_f$
- Mathematically, a Laplacian is a perfect operator: smooth, bounded, self-adjoint. Its discretisation yields a symmetric matrix
- In contrast, non-orthogonal correction is explicit, unbounded and unsigned
- Limited non-orthogonal correction: explicit part clipped to be smaller than its implicit counterpart, base on the current solution

$$\lambda \frac{|\mathbf{s}_f|}{|\mathbf{d}_f|}(\phi_N - \phi_P) > \mathbf{k}_f \bullet \nabla(\phi)_f$$

  where λ is the limiter value

- Treatment of mesh non-orthogonality over 90^o: mesh is formally invalid
  - This corresponds to a Laplacian operator with negative diffusion
  - Stabilise the calculation and remove non-orthogonal correction term
  - Note: This is a “rescue procedure”: reconsider mesh and results!

Discretisation Settings

Proposed Settings for Hexahedral Meshes

- Gradient scheme: Gauss or Gauss with limiters
- Convection scheme
  - In initial settings or unknown mesh quality, always start with Upwind. If this fails, there are problems elsewhere in case setup
  - Momentum equation: for second order, recommend linear upwind. with optional gradient limiters
  - TVD/NVD schemes for bounded scalars (eg. turbulence); optionally, use deferred correction formulation
- Diffusion scheme: settings depend on max non-orthogonality
  - Below 60 deg, no special practice: Gauss linear corrected
  - Above 70 deg, non-orthogonality limiter: Gauss linear limited 0.5
- In all cases, monitor boundedness of scalars and adjust convection and diffusion schemes to remove bounding messages

Discretisation Settings

Proposed Settings for Tetrahedral Meshes

- On tetrahedral meshes, cell neighbourhood is minimal: a tet has only 4 neighbours
- Skewness and non-orthogonality errors are larger and with substantial effect on the solution: it is essential to re-adjust the discretisation
- Gradient scheme: least squares; in most cases without limiters
- Convection scheme
  - On simple cases, use upwinding; nature of discretisation error changes due to lack of mesh-to-flow alignment
  - For highly accurate simulations, special (reconstructed) schemes are used
- Diffusion scheme: always with non-orthogonality limiters. Control limiter based on boundedness messages on scalars

Summary

Summary

- Discretisation settings in tutorials are a good starting point
- Variation in mesh structure (tetrahedral, hexahedral and polyhedral) means that no single choice will work for all meshes
- In complex physics, consider physical properties of variables: boundedness and conservation
- OpenFOAM is regularly set up for high accuracy rather than convergence to steady-state: The fact that a solver converges does not necessarily mean the results are correct (or physical!)
- “Special applications” like LES require additional care: energy conserving numerics, low diffusion and dispersion errors
- Guidance provided for main mesh types: hex and tet. Polyhedral meshes use hex settings
- Further complications may be introduced by moving mesh and topological changes