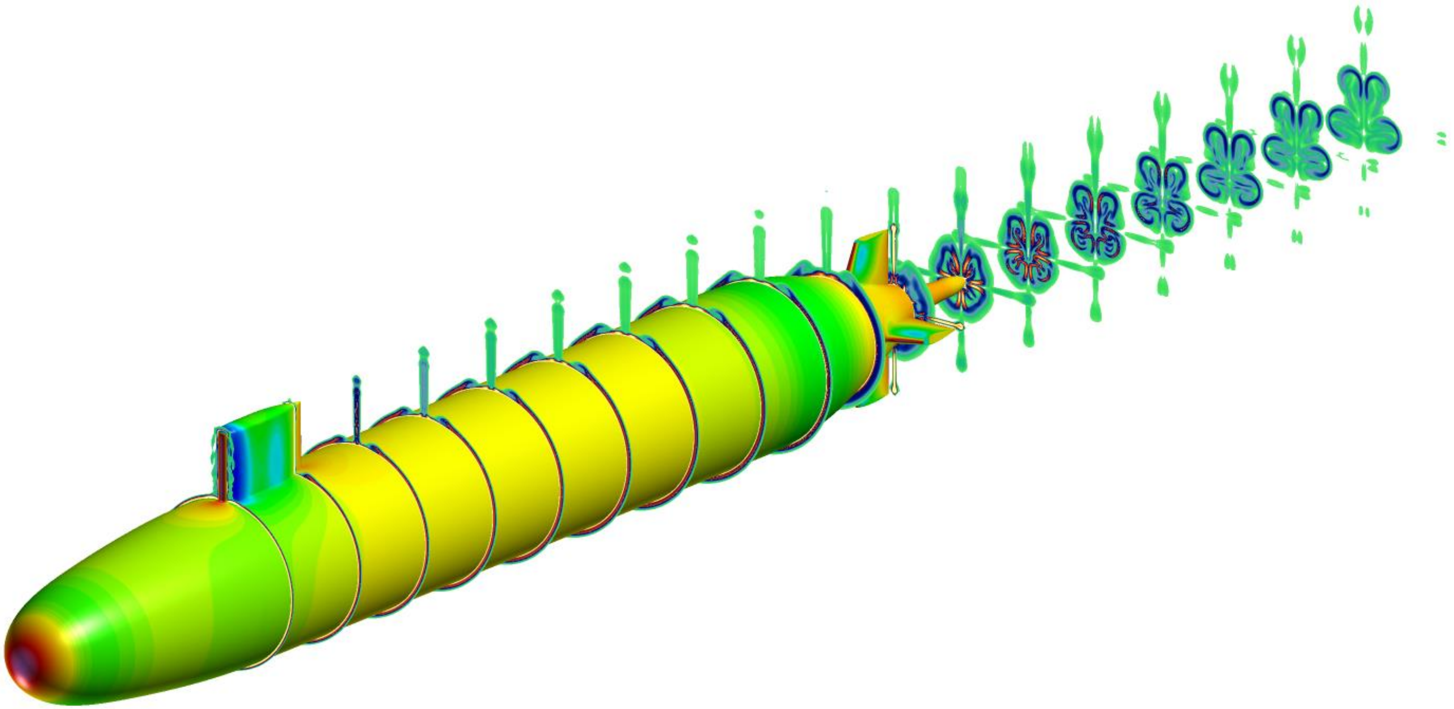


# Improvements of the Interpolation and Non-Orthogonal Correction Schemes in Caelus

Dr. Chris Sideroff
Dr. Darrin Stephens

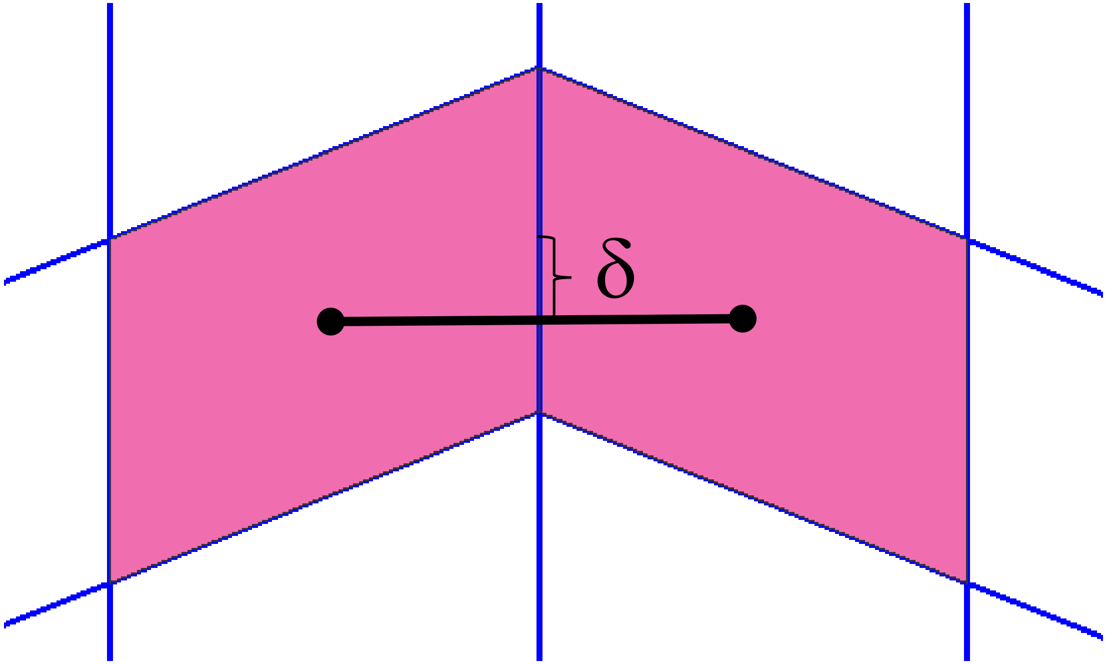
Applied CCM

# Mesh Quality Issues

- Sensitivity to mesh quality widely reported
- Two main metrics used to quantify “quality”
  - Non-orthogonality
  - Skewness
- Non-orthogonality *tends* affect gradient reconstruction, i.e. diffusion terms
- Skewness *tends* to affect face interpolation, i.e. convective terms

# Central Interpolation

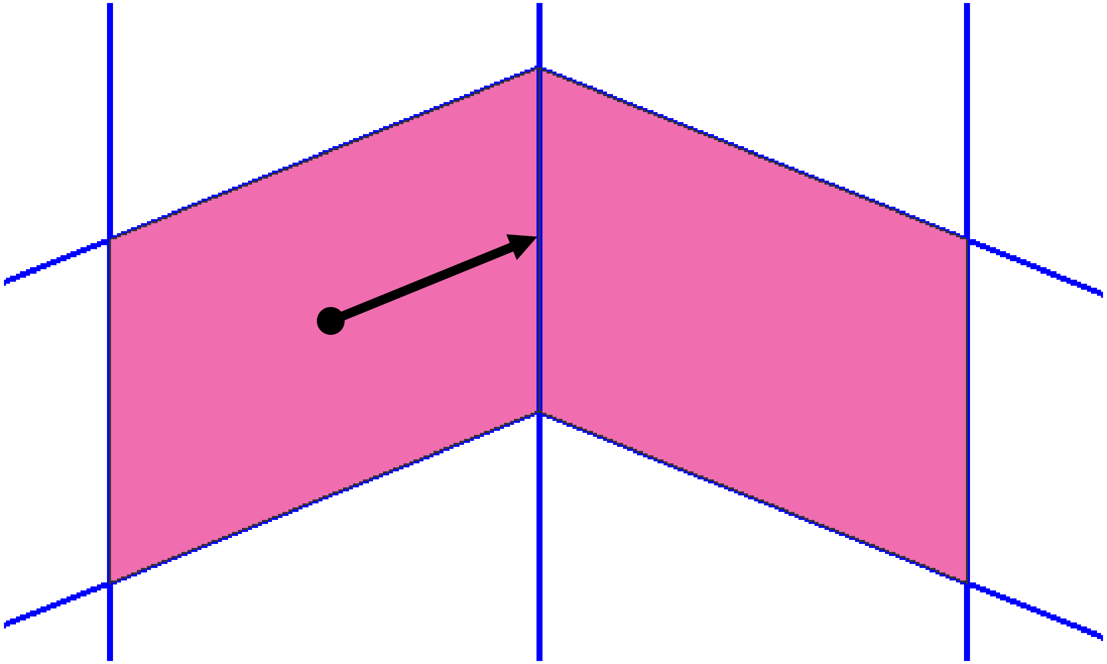
- Central-based schemes (linear, limitedLinear) are one dimensional – very susceptible to skewness



# Upwind Interpolation

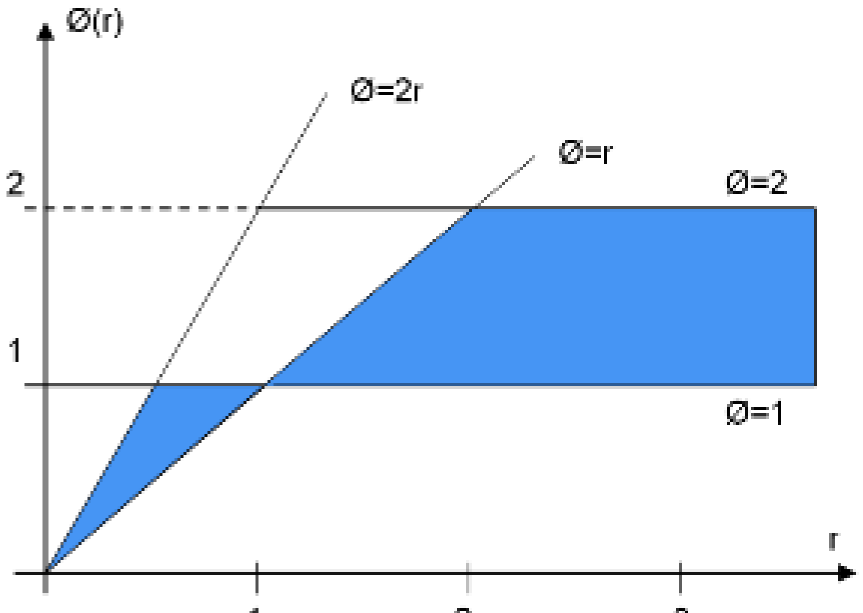
- Include gradient to get better than 1st order accuracy

$$\varphi_f = \varphi_c + \psi(\bar{x}_f - \bar{x}_c)\frac{\partial \varphi}{\partial \bar{x}}$$



# Interpolation Schemes Improvements

- `linearUpwind`
  - Does not limit the slope
  - Attempts to limit the fluxes
- Introduce commonly used slope limiters that adhere to the TVD condition




Admissible limiter region for second-order TVD schemes
(Sweby, 1984)

# New Interpolation Schemes

- 8 new interpolation schemes
  - Barth-Jespersen: `linearUpwindBJ{V}`
  - Monotonic central: `linearUpwindMC{V}`
  - Differential limiter: `linearUpwindDL{V}`
  - MG: `linearUpwindMG{V}`
  - Minmod: `linearUpwindMinmod{V}`
  - UMIST: `linearUpwindUMIST{V}`
  - VanAlbada: `linearUpwindVanAlbada{V}`
  - VanLeer: `linearUpwindVanLeer{V}`

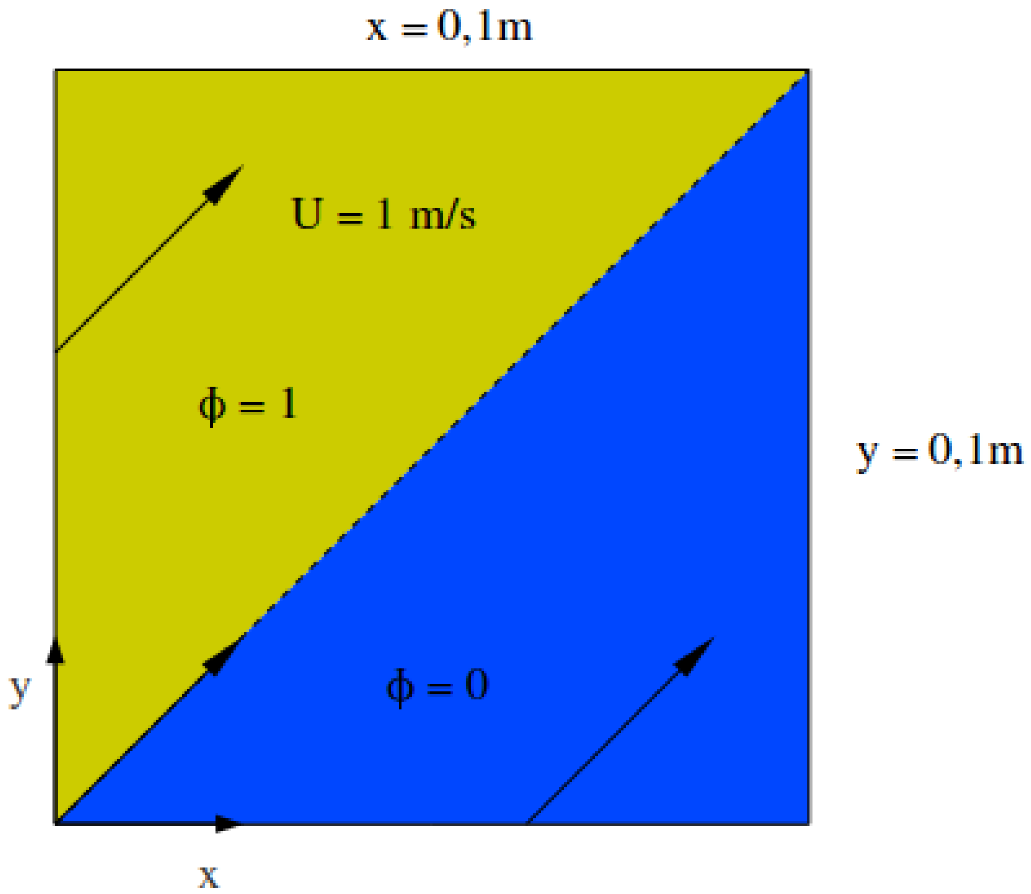
# New Interpolation Schemes cont.

- 2 implementations
  - Fully implicit
  - Deferred correction
- Deferred correction used for more difficult meshes
- Controlled with dictionary:

```
div(phi,U) deferredCorrection Gauss dcLinearUpwindBJV grad(U) 0.5
```

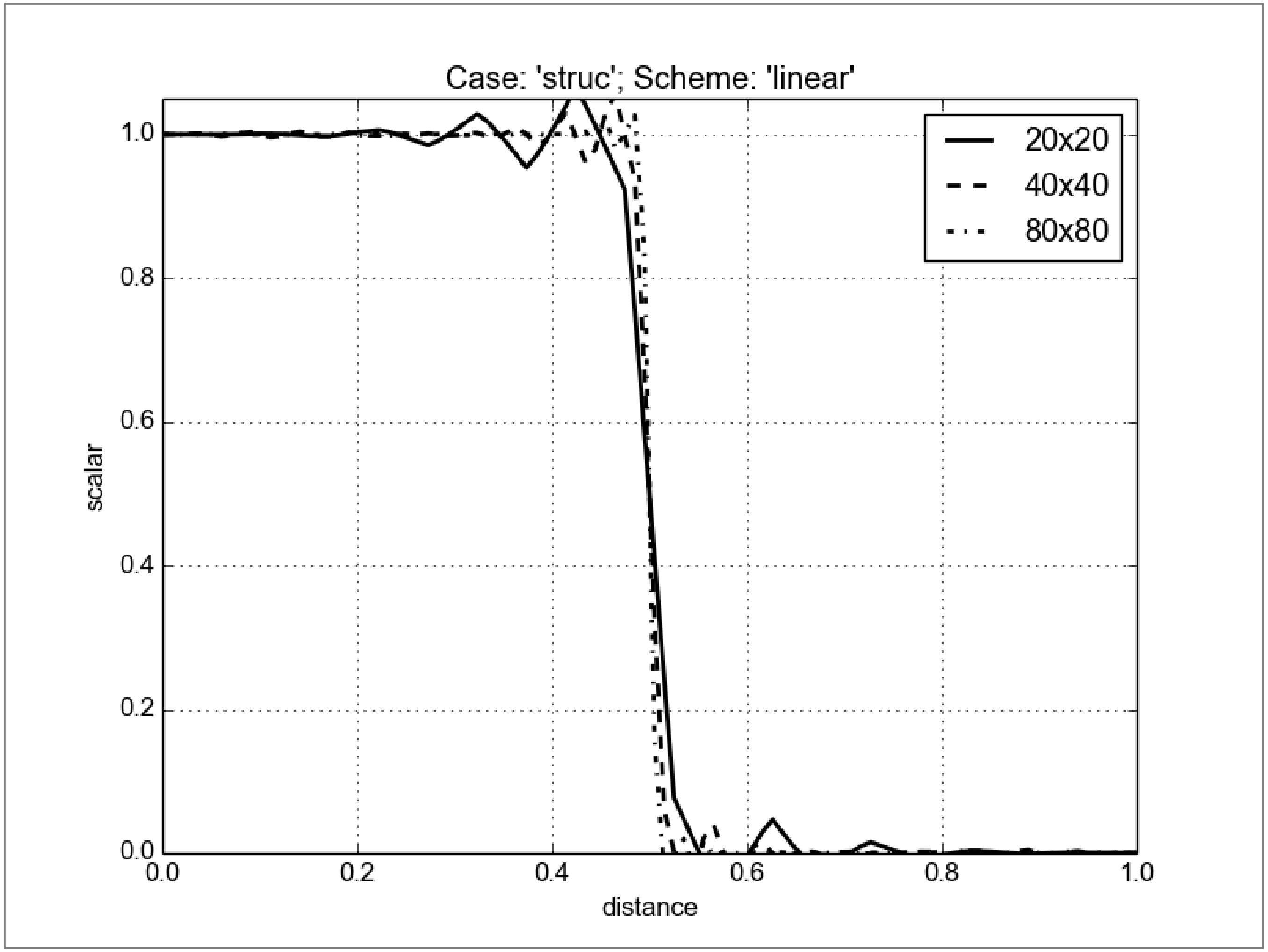
- 0.5 explicit relaxation factor that multiplies the explicit (high order) part of the discretization (0 – 1)

# Advecting Step

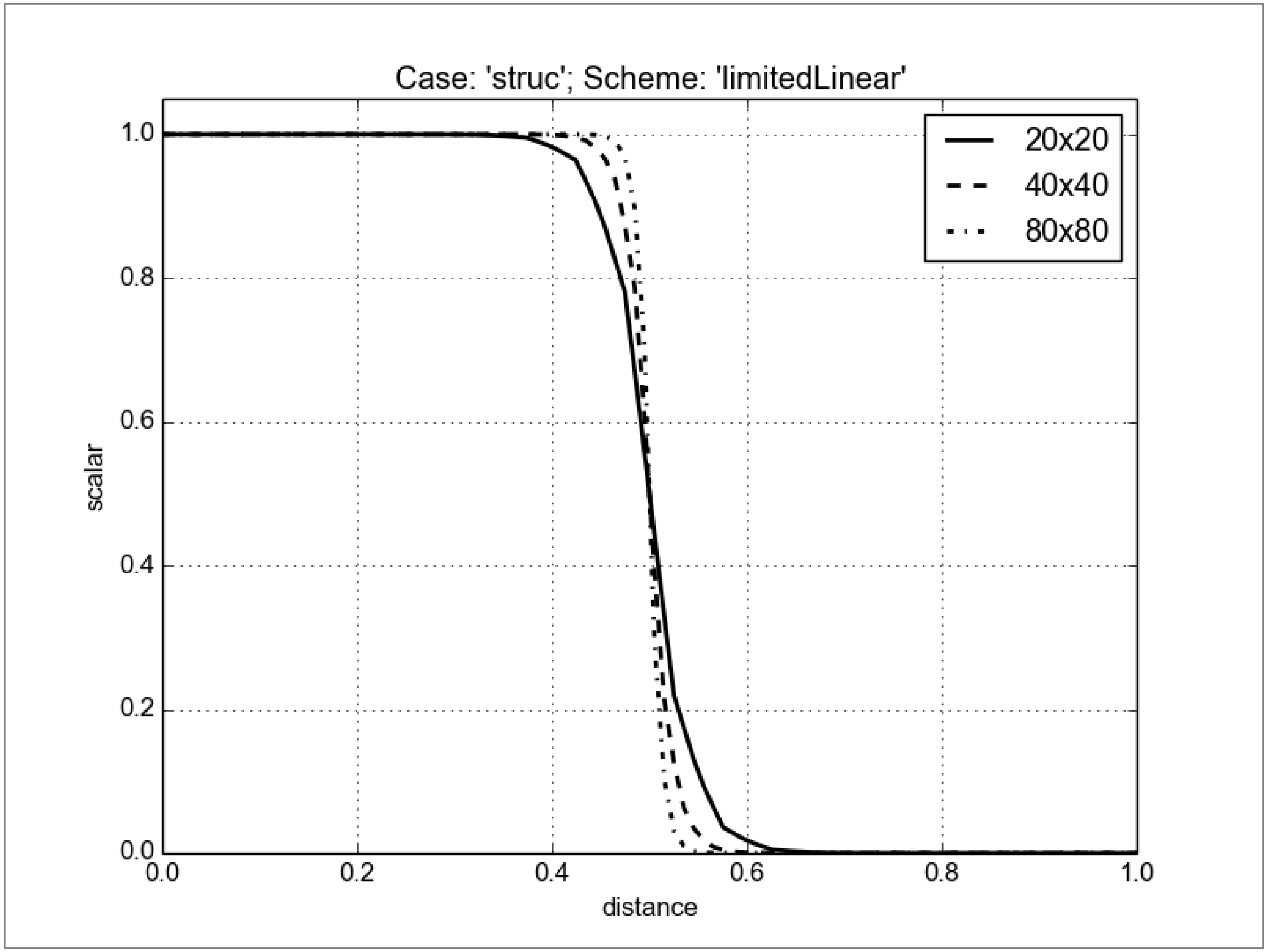



*Tobias Holzmann, http://www.holzmann-cfd.de/index.php/en/numerical-schemes*

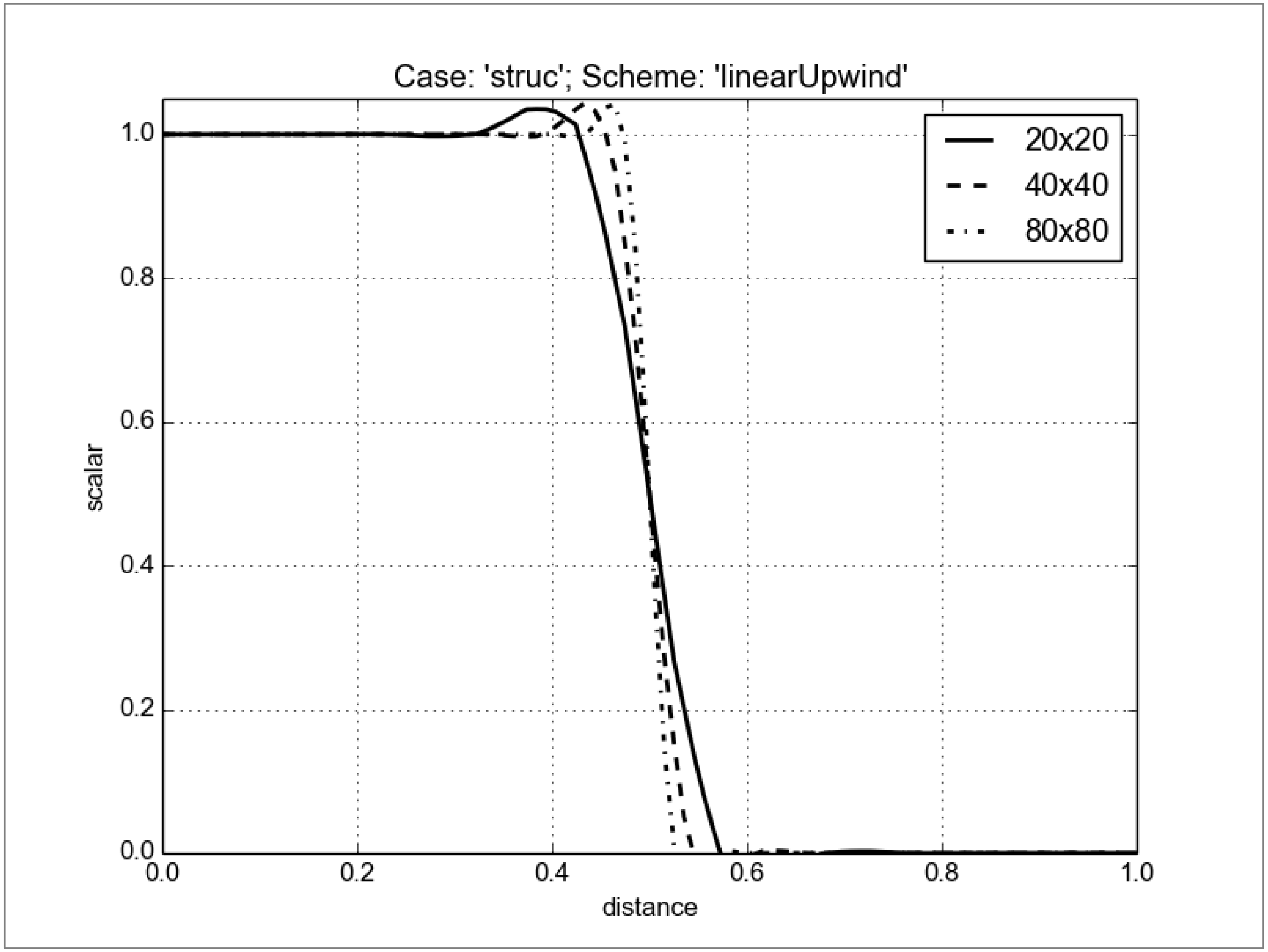
# Advecting Step



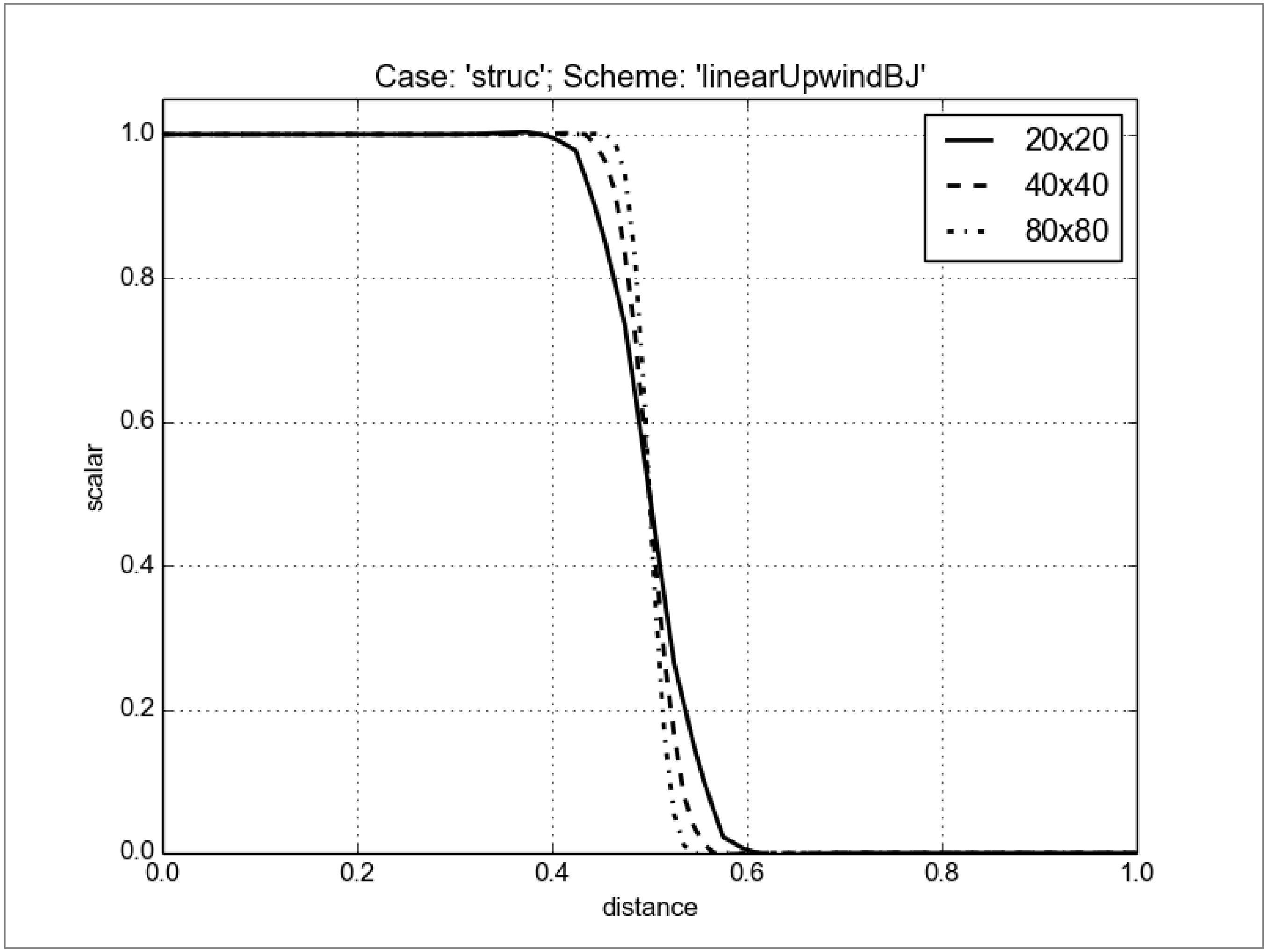
# Advecting Step



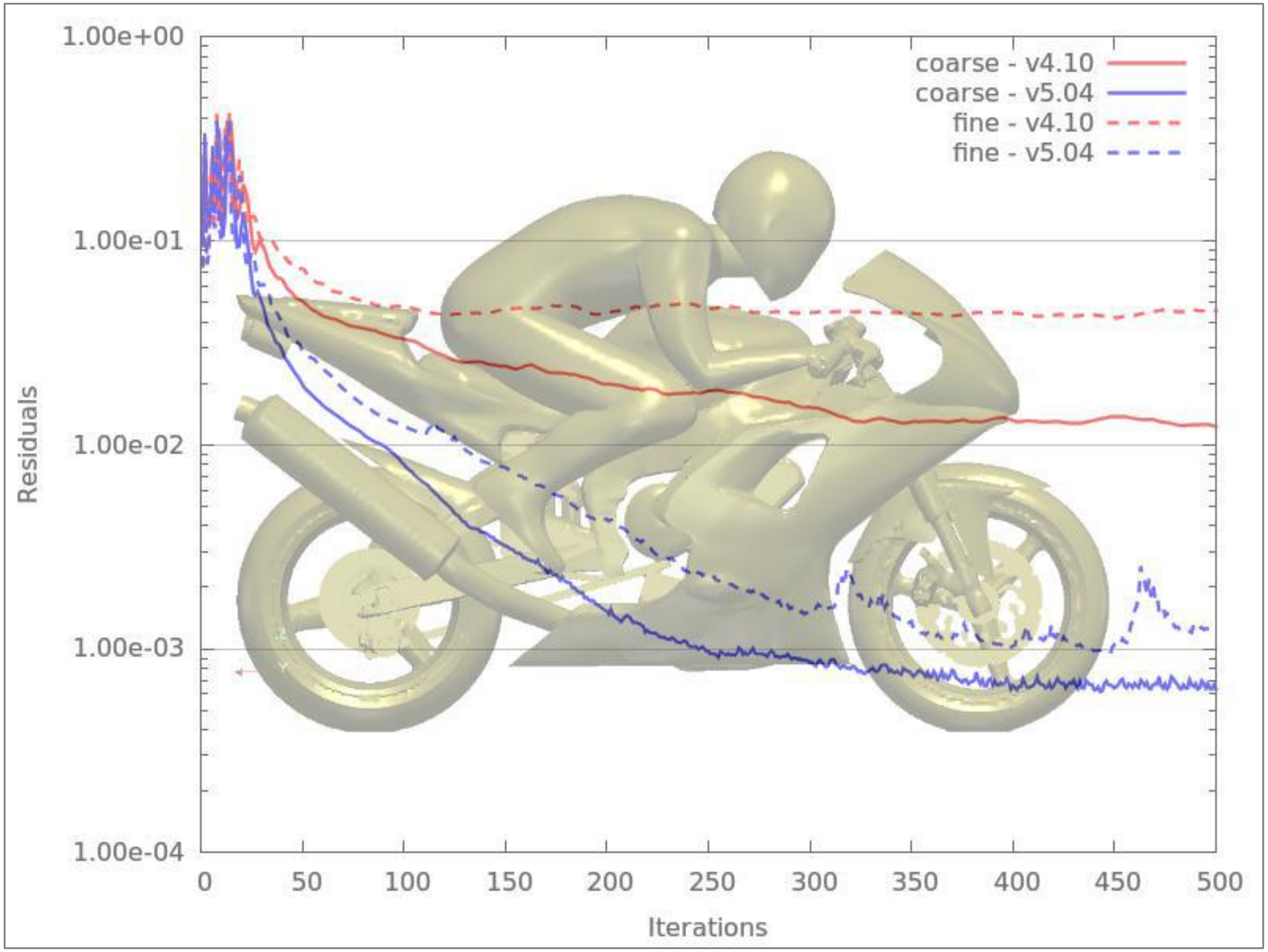
# Advecting Step



# Advecting Step



# motorBike Test Case



# Non-orthogonal Correction on Boundaries

- $1^{st}$ order accuracy near boundaries
  - Talks from OFWS 8 and 9
  - Personal communication
- Only when grid is non-orthogonal near boundaries
  - Almost always
- No correction applied for all patch types except processor and coupled
- OpenFOAM 2.x and foam-extend 3.x

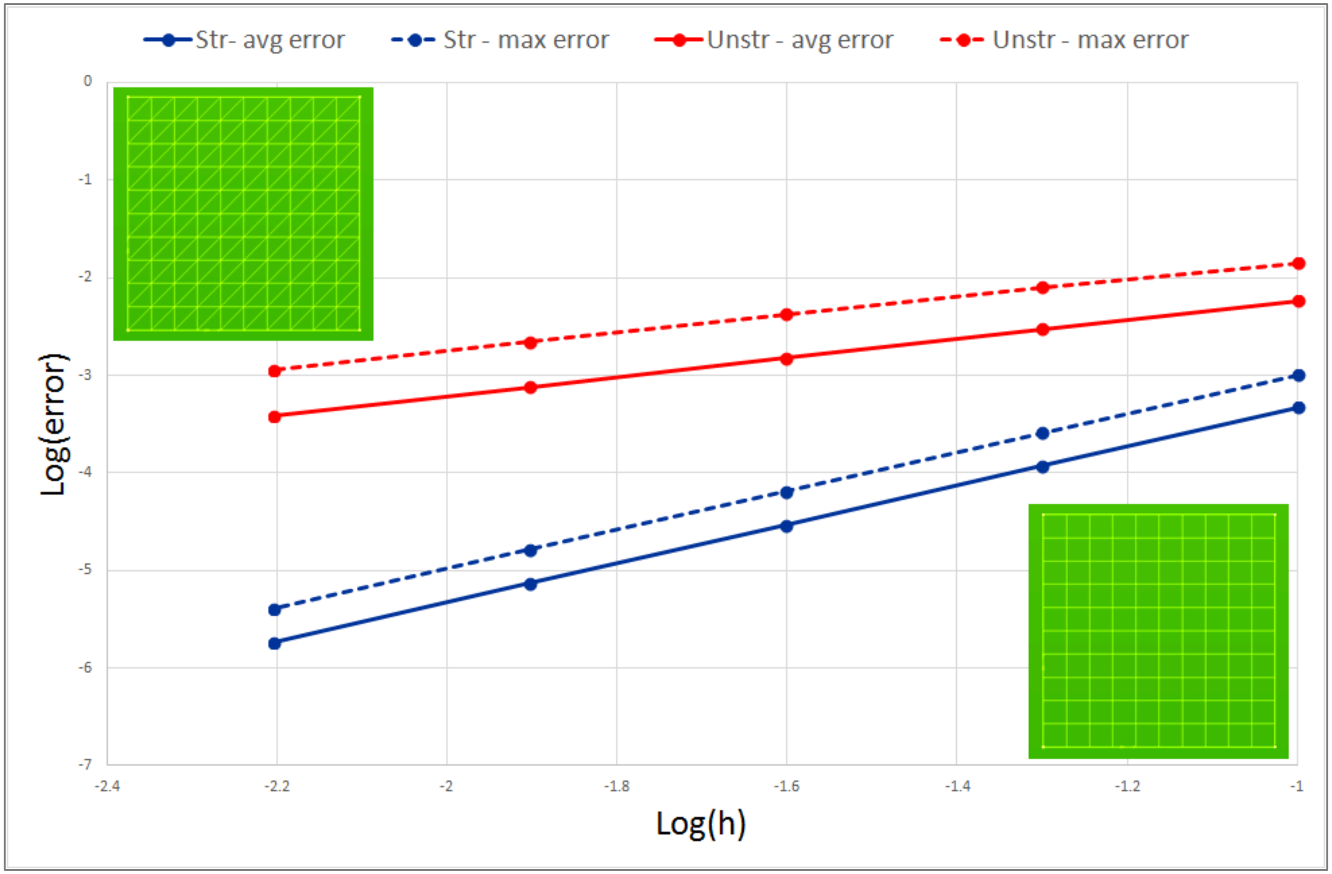
```
...
forAll(corrVecs.boundaryField(), patchi)
    {
        fvsPatchVectorField& patchCorrVecs =
corrVecs.boundaryField()[patchi];

        if (!patchCorrVecs.coupled())
        {
            patchCorrVecs = vector::zero;
        }
        else
        {
...
```

# N.O.C. on Boundaries cont.

- Doesn't work with wall functions < yet ;-) >
  - Wall functions work by modifying $\nu_t$ near wall cell
- Turned on with dictionary setting
  - `laplacian(DT,T) Gauss linear secondOrderCorrected;`

# Fixed Value Boundary Conditions

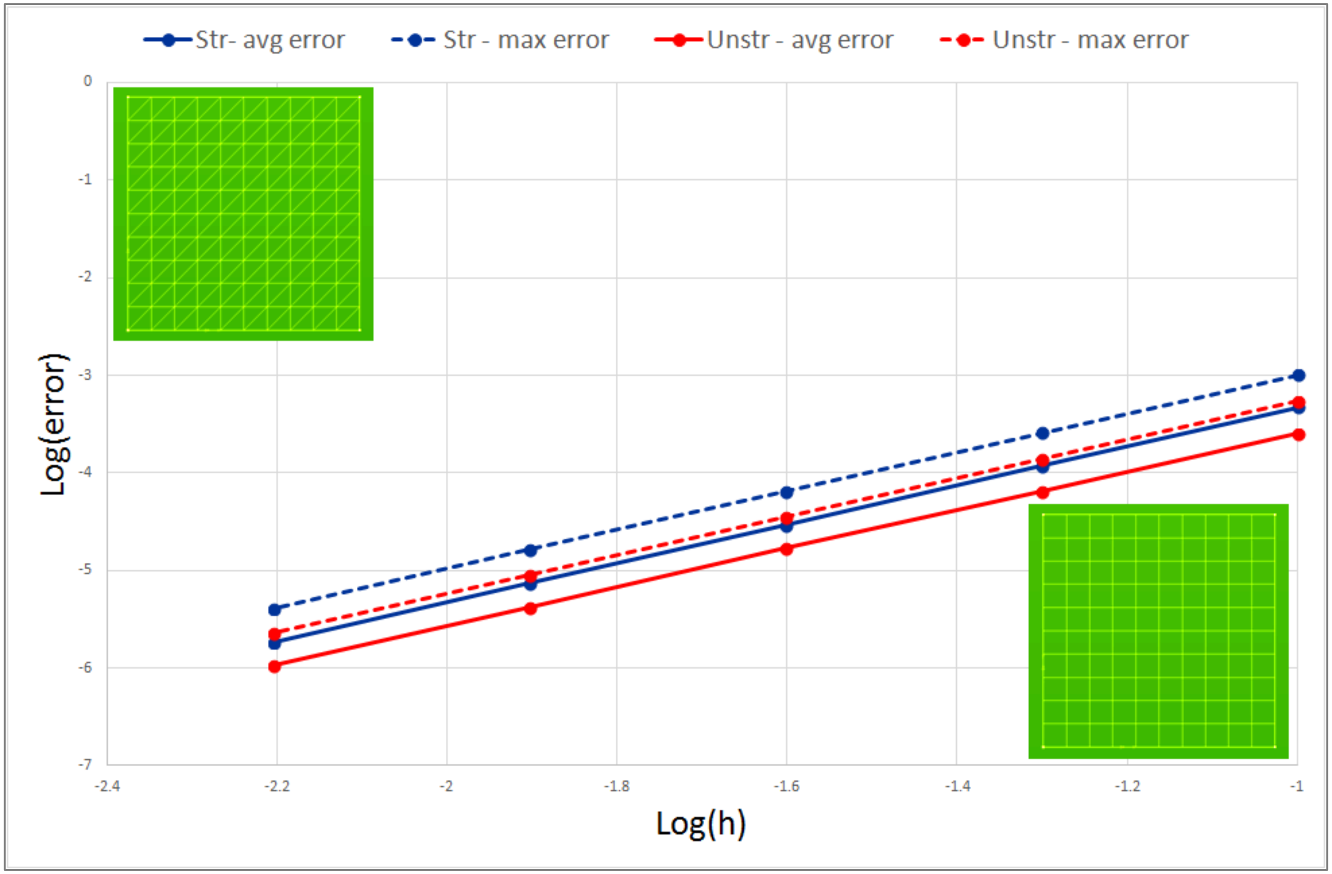
- Method of manufactured solutions




Caelus 4.10 (OpenFOAM, foam-extend)

# Fixed Value Boundary Conditions

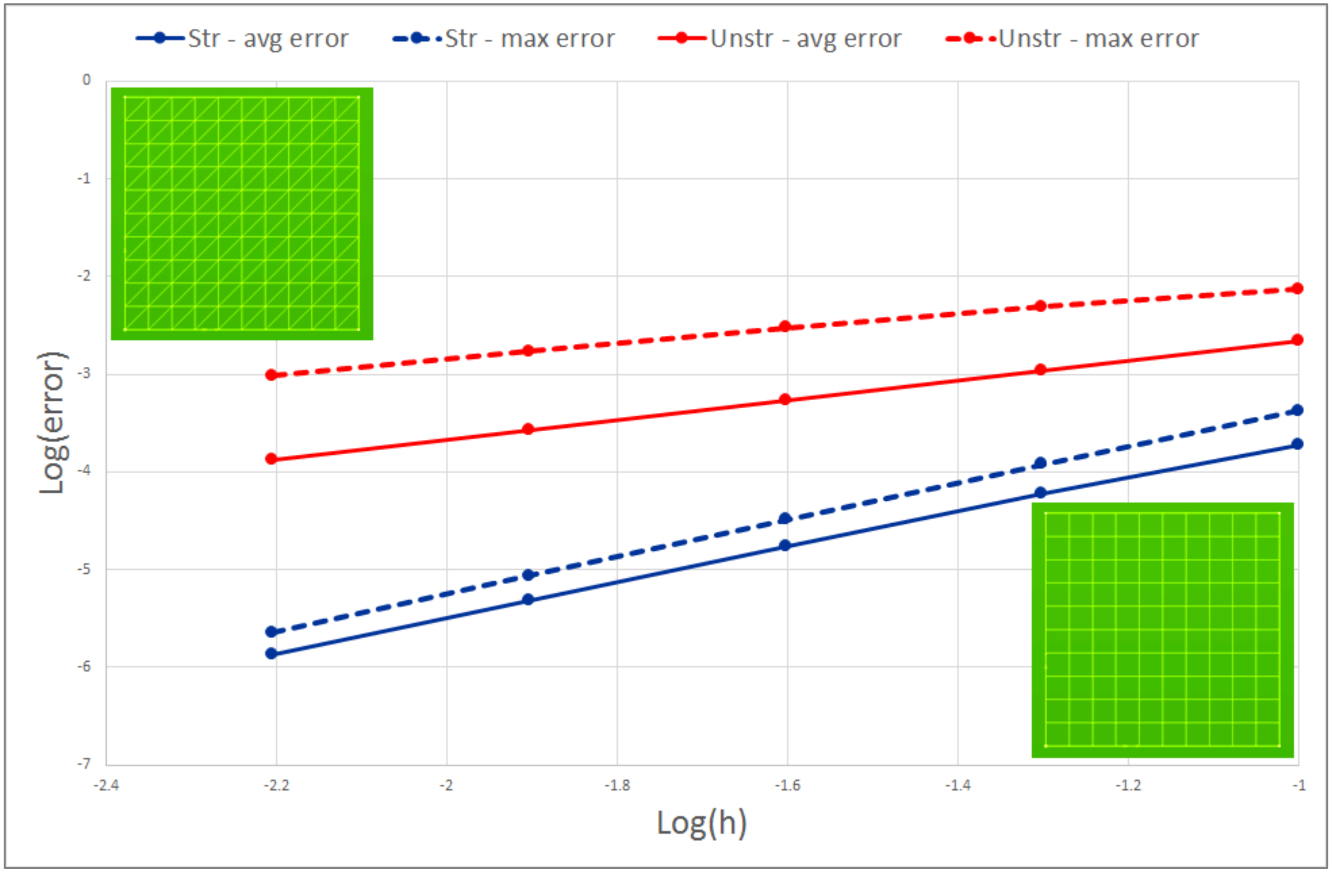
- Method of manufactured solutions




Caelus 5.04

# Fixed Gradient Boundary Conditions

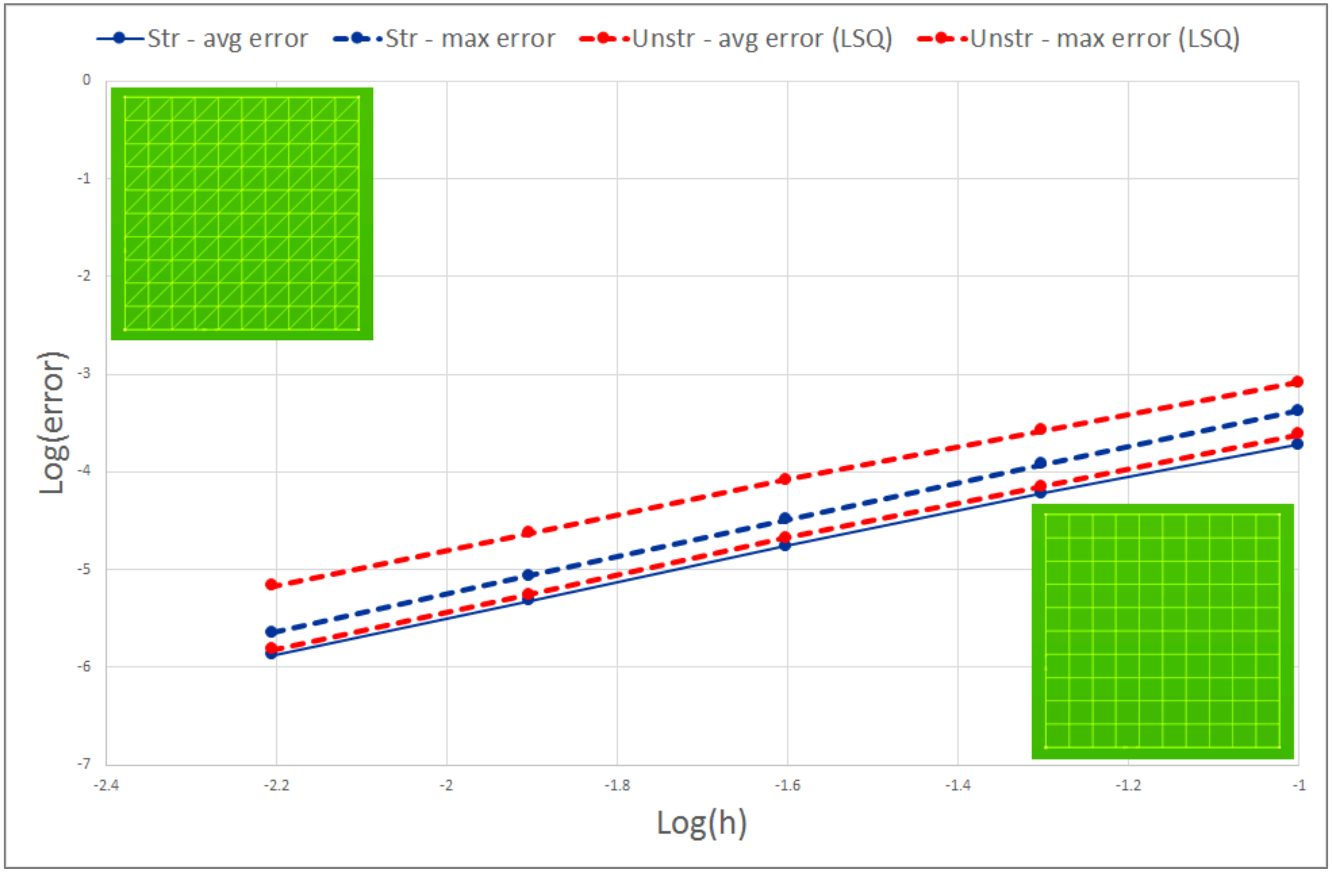
- Method of manufactured solutions




Caelus 4.10 (OpenFOAM, foam-extend)

# Fixed Gradient Boundary Conditions

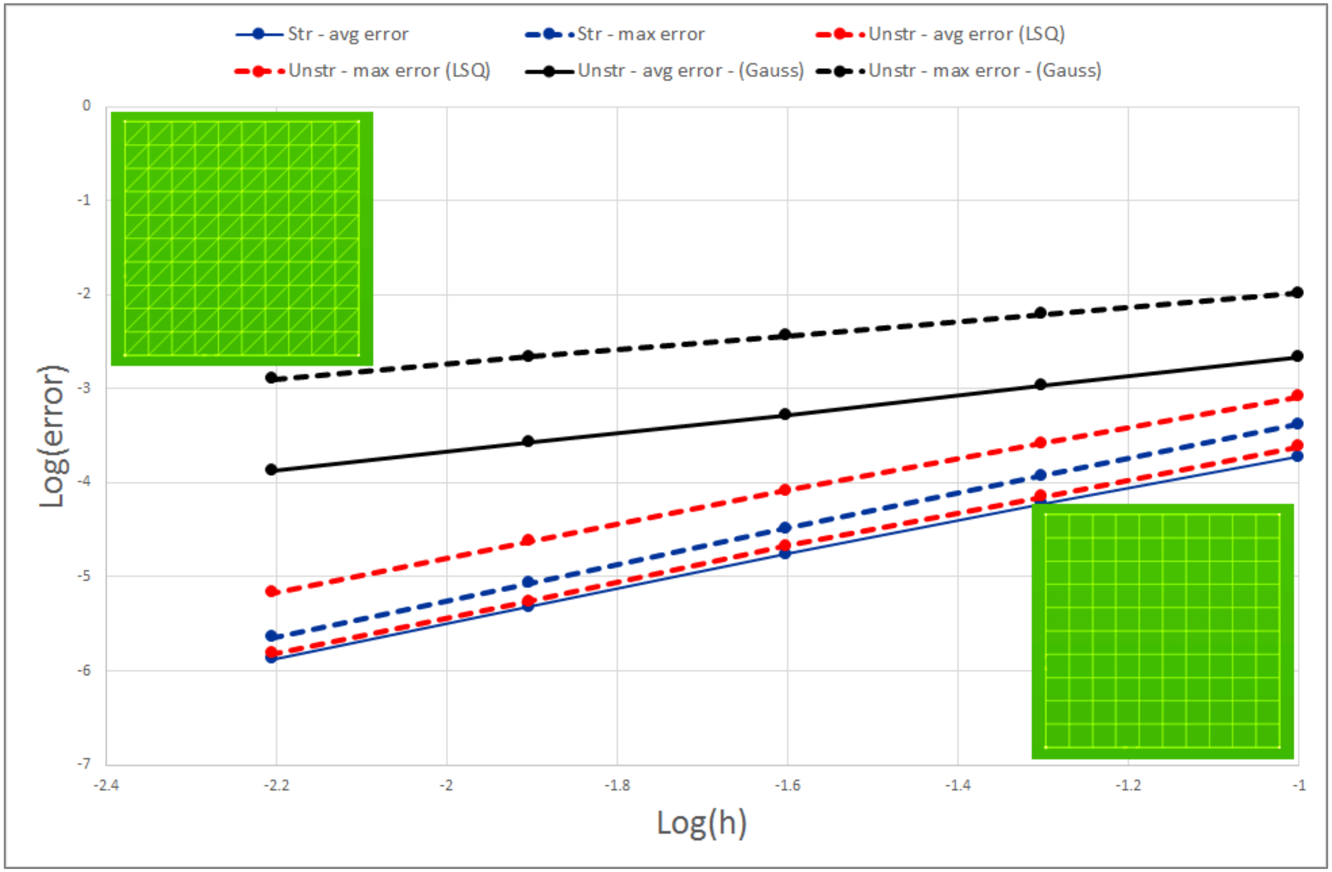
- Method of manufactured solutions




Caelus 5.04

# corrGauss Gradient Scheme

- Face value unknown – iterate a few times (default = 2)




Caelus 5.04

# What is Caelus?

- Caelus is a fork of OpenFOAM
- Free and open: www.caelus-cml.com

4.10

- Supports multiple platforms (Windows, Linux and Mac)
- Robust compilation (Scons: www.scons.org)
- Easy to install
- Verified and validated turbulence models (conforms to most commonly used variants published in open literature)
- Documentation and validation cases

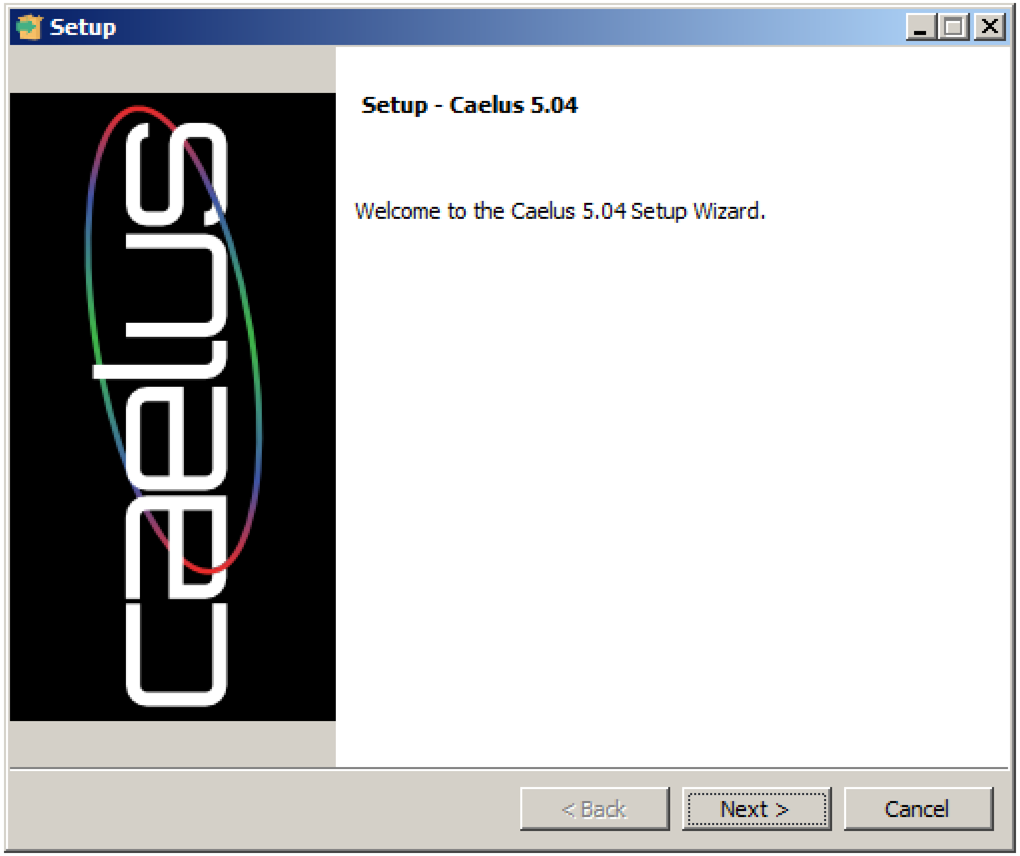
5.04

- Improving algorithmic robustness on non-"perfect" meshes
  - Multi-dimensional interpolation
- Improved accuracy on non-"perfect" meshes
  - Non-orthogonal boundary correction
- Swak, Python wrapping, etc

5.10*

- New wall functions, new compressible solvers, more numerical improvements

# What is Caelus?



# What is Caelus?

2. Theory — Caelus Docume ×

www.caelus-cml.com/userdoc/2_Theory.html#sst-model

Apps | Bookmarks | PW | Glyph2 | Tcl 8.5 | CFD CFD Online | CFD PW Forum | CFD OF Forum | CFD Rev | Gmail | Tools | Usuals | Archive

Caelus Documentation | 4.10 | Menu | Page | « 1. Introduction | 3. Validation an... » | Search

## 2.1.4. $k$–$\omega$ SST Model

**Model description**

The $k$–$\omega$ shear stress transport (SST) turbulence model was introduce by Menter [16] as an improvement on the original two equation $k$–$\omega$ model of Wilcox [14]. The SST formulation uses the $k$–$\omega$ formulation in the inner parts of the boundary layer and it switches to a $k$–$\varepsilon$ behaviour in the free-stream portions of the flow domain. Thus, the $k$–$\omega$ SST turbulence model does not require a damping function close to the wall enabling its use with near-wall resolved meshes without modification. At the same time, the $k$–$\omega$ SST models reverts to a $k$–$\varepsilon$ model behaviour away from the walls thus removing the sensitivity of the original $k$–$\omega$ model to the $\omega$ free-stream boundary value.

The implementation of the $k$–$\omega$ SST model in Caelus 4.10 is that of Kuntz et al [3]. The transport equations for turbulence kinetic energy and specific dissipation are:

$$\partial_t k + \partial_j(u_j k) = \frac{P}{\rho} - \beta^* \omega k + \partial_j[(\nu + \sigma_k \nu_t)\partial_j k]$$

$$\partial_t \omega + \partial_j(u_j \omega) = \frac{\gamma P}{\rho \nu_t} - \beta \omega^2 + \partial_j[(\nu + \nu_t)\partial_j \omega] + 2(1 - F_1)\frac{\sigma_{\omega 2}}{\omega}\partial_j k \partial_j \omega$$

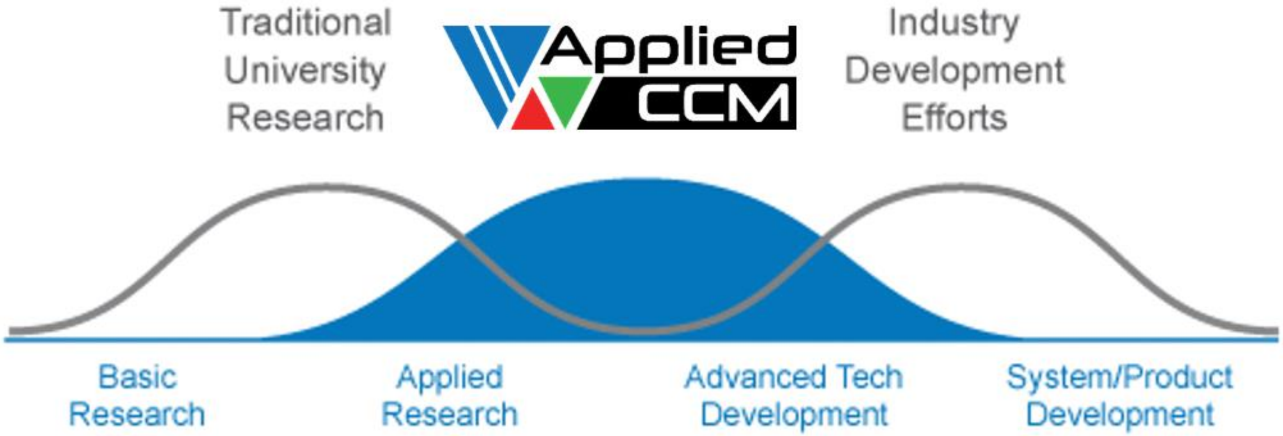
It should be noted that this set of transport equations only applicable for incompressible flows as density was assumed constant. Terms appearing in the $k$–$\omega$ SST model have the following definitions:

**Model definitions**

| Symbol | Definition | Formula |
|---|---|---|
| $\nu_t$ | Turbulent viscosity | $\frac{a_1 k}{max(a_1 \omega, SF_2)}$ |
| $S_{ij}$ | Symmetric part of stress tensor | $\frac{1}{2}(\partial_j u_i + \partial_i u_j)$ |
| $S$ | Strain invariant | $S = \sqrt{S_{ij} S_{ij}}$ |
| $F_1$ | Blending function | $tanh(arg_1^4)$ |
| $arg_1$ | Argument for $F_1$ function | $min\left[max\left(\frac{\sqrt{k}}{\beta^* \omega d}, \frac{500\nu}{d^2 \omega}\right), \frac{4\sigma_{\omega 2} k}{CD_{k\omega} d^2}\right]$ |
| $CD_{k\omega}$ | Function | $max\left(2\sigma_{\omega 2}\frac{1}{\omega}\frac{\partial k}{\partial x_j}\frac{\partial \omega}{\partial x_j}, 10^{-10}\right)$ |

# About Applied CCM

- Specialise in the application, support and development of OpenFOAM-based computational mechanics
- Main developers of Caelus
- People
  - Darrin Stephens, Chris Sideroff and Aleks Jemcov*
- Locations
  - Australia, Canada and USA
- Engage with customers as their technology partner



# Questions

**Applied CCM Canada**
Dr Chris Sideroff

Phone: 613 276 7472
Email: c.sideroff@appliedccm.ca
Web: www.appliedccm.ca

**Applied CCM Pty Ltd**
Dr Darrin Stephens

Phone: 03 8376 6962
Email: d.stephens@appliedccm.com.au
Web: www.appliedccm.com.au