adjointShapeOptimizationFoam

Shenan Grossberg

University of Exeter
Hydro International

4th UK & Eire OpenFOAM User Meeting

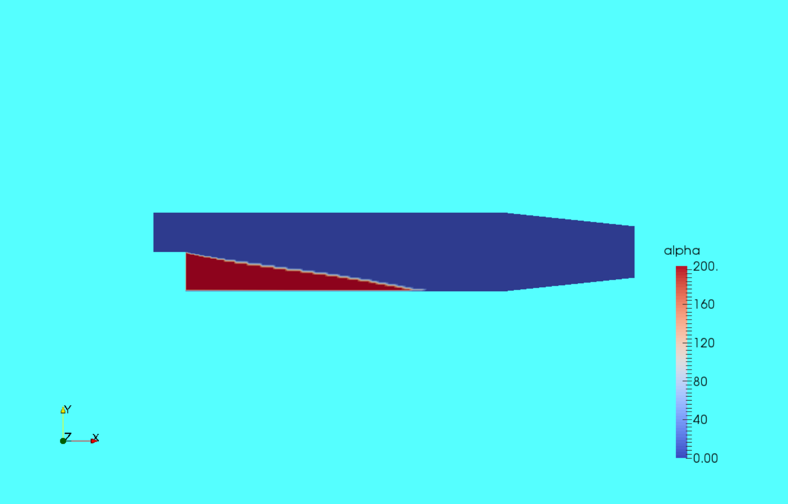
18th April 2016

RESEARCHER DEVELOPMENT

Aims

- Explain what adjointShapeOptimizationFoam does
- Show how the equations are coded in OpenFOAM

Demo



Equations

Primal Equations:

$$(\mathbf{v} \cdot \nabla)\mathbf{v} = -\nabla p + \nu \nabla^2 \mathbf{v} - \alpha \mathbf{v}$$
$$\nabla \cdot \mathbf{v} = 0$$

Adjoint Equations:

$$-(\mathbf{v} \cdot \nabla)\mathbf{u} - (\nabla \mathbf{u}) \cdot \mathbf{v} = -\nabla q + \nu \nabla^2 \mathbf{u} - \alpha \mathbf{u}$$
$$\nabla \cdot \mathbf{u} = 0$$

Steepest Descent Algorithm:

$$\alpha_{i+1} = \alpha_i - \mathbf{u} \cdot \mathbf{v} \; V \, \delta$$

Cost Function

The solver optimises for total pressure loss:

$$J = \int_{\Gamma_i} d\Gamma_i \left(p + \frac{1}{2}v^2\right) - \int_{\Gamma_o} d\Gamma_o \left(p + \frac{1}{2}v^2\right)$$

Cost Function

The solver optimises for total pressure loss:

$$J = \int_{\Gamma_i} d\Gamma_i \left(p + \frac{1}{2}v^2\right) - \int_{\Gamma_o} d\Gamma_o \left(p + \frac{1}{2}v^2\right)$$

It manifests in the following adjoint boundary conditions:

- adjoint inlet velocity
- adjoint outlet pressure
- adjoint outlet velocity

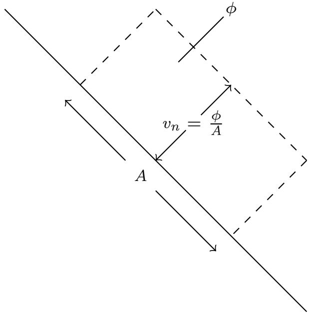
Adjoint inlet velocity

$$\mathbf{u}_t = 0$$
$$u_n = -1$$

Adjoint outlet pressure

$$q = (u_n - 1)v_n + \mathbf{v} \cdot \mathbf{u}$$

```
operator==((phiap/patch().magSf() - 1.0)*phip/patch().magSf() + (Up & Uap));
```



Adjoint outlet velocity

$$\mathbf{u} = u_n\mathbf{n} + \frac{\mathbf{v}_t}{v_n}$$

```
scalarField Un(mag(patch().nf() & Up));
vectorField UtHat((Up - patch().nf()*Un)/(Un + SMALL));
vectorField::operator=(phiap*patch().Sf()/sqr(patch().magSf()) + UtHat);
```

Thank you

Thank you for listening.