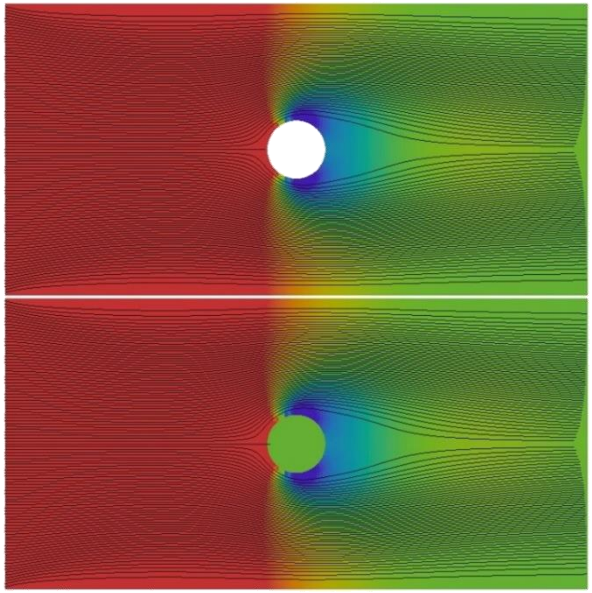
Geometric Immersed Boundaries (GIB) – A New Framework For Applying Boundary Conditions in OpenFOAM®

OpenFOAM User conference 2015

October 19 – 21, Stuttgart, Germany

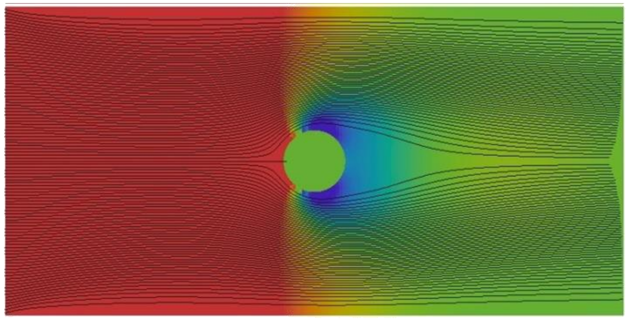


Georgios Karpouzas, ENGYS Ltd. – NTUA

Eugene de Villiers, ENGYS Ltd.

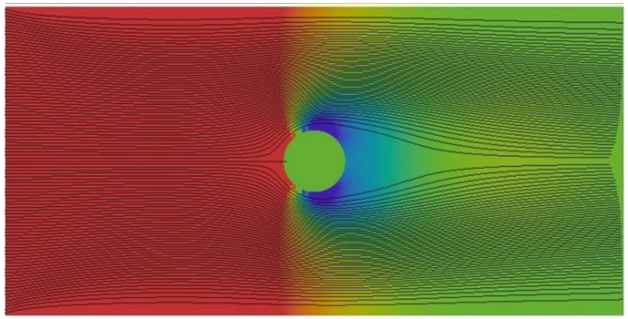
Contents

- Motivation
- Methodology
- Validation
- Applications
- Moving GIB
- Closing Comments



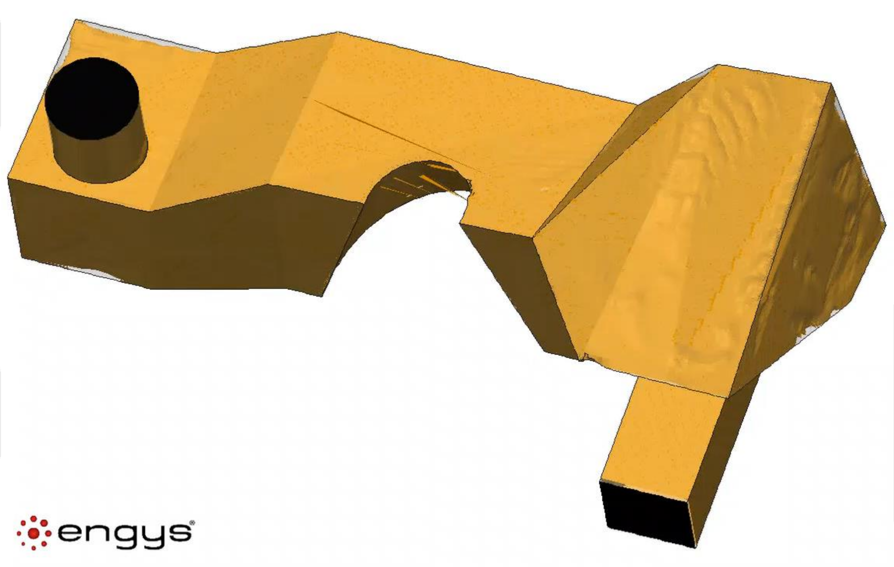
Contents

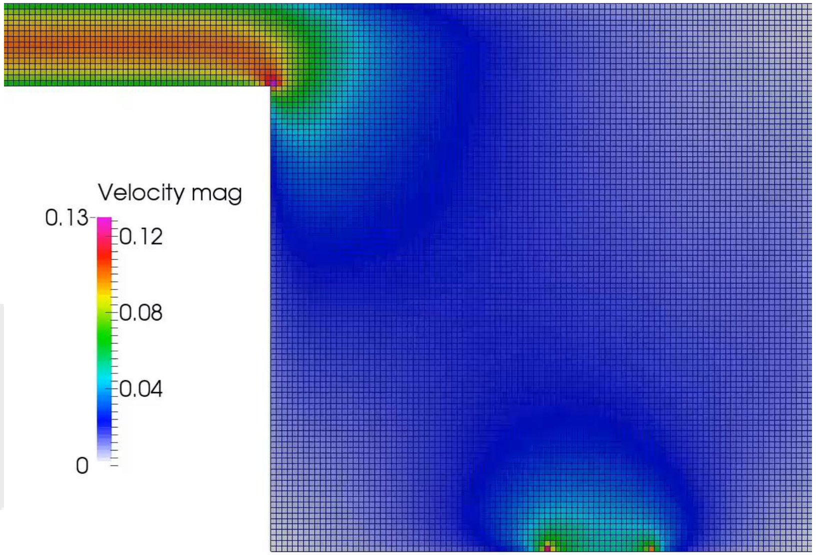
- Motivation
- Methodology
- Validation
- Applications
- Moving GIB
- Closing Comments



Motivation

- Topology optimization
- Level-set coupled with the continuous adjoint method





*G. Karpouzas, E.M. Papoutsis-Kiachagias, T. Schumacher, E. de Villiers, K.C. Giannakoglou, C. Othmer. “**Adjoint Optimization for Vehicle External Aerodynamics”,** JSAE - to be published soon*

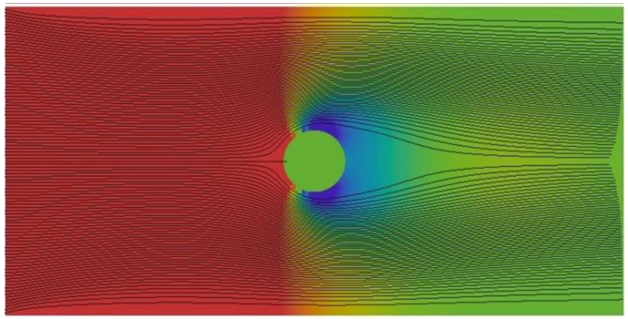
C. Othmer. **“Adjoint methods for car aerodynamics”, Journal of Mathematics in Industry 2014 4:6*

Motivation

- Currently simple immersed boundaries (IB) are applied on the fluid-solid interface
- Resistance/porosity is added to the solid cells of the matrix which blocks the velocity
- Lacks of accuracy especially in the turbulent cases
- In-situ IB primal results do not exactly match boundary fitted equivalent
- Results in approximate objective/optimal
- Solution: Implement immersed boundaries with the same accuracy as a real boundary

Contents

- Motivation
- Methodology
- Validation
- Applications
- Moving GIB
- Closing Comments

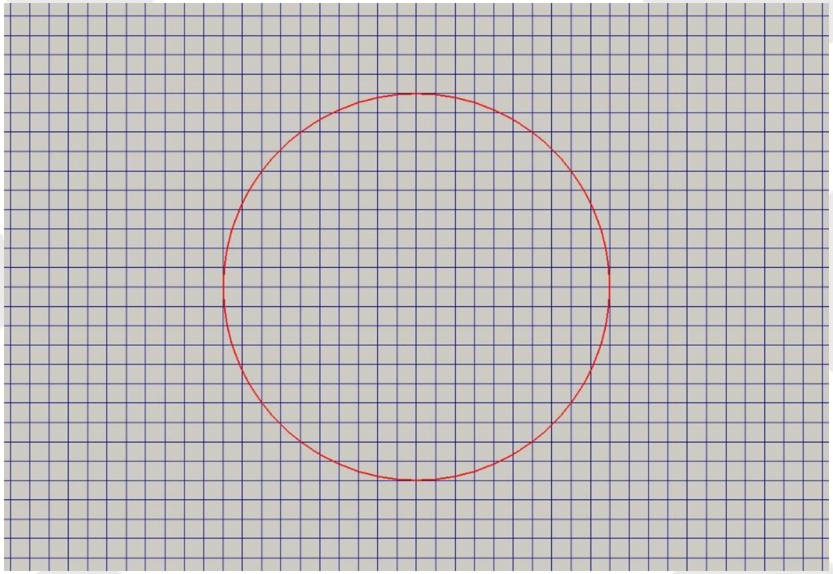


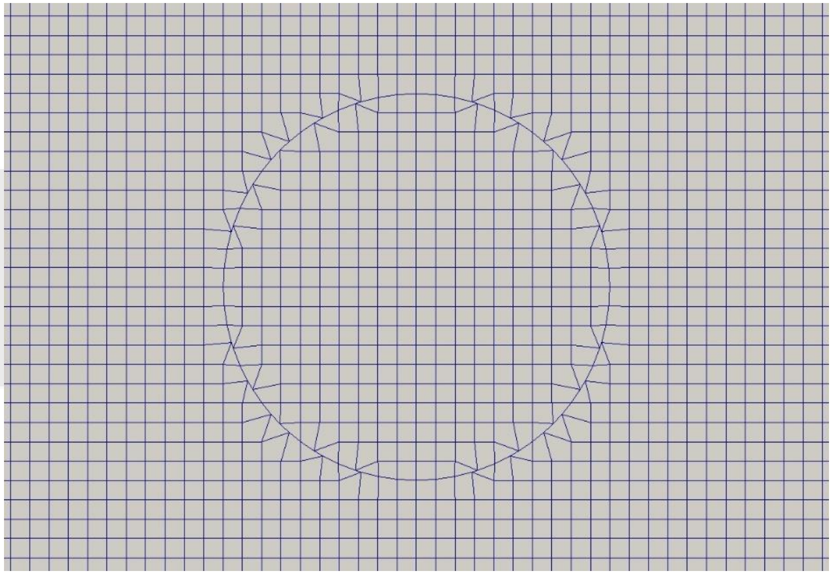
Methodology | Goals

- Same accuracy as body fitted meshes
- Automation: Work with every solver and operation
- Same interface as the other boundaries
- Apply the existing boundary conditions (`fixedValue`, `zeroGradient`…) on the immersed boundaries

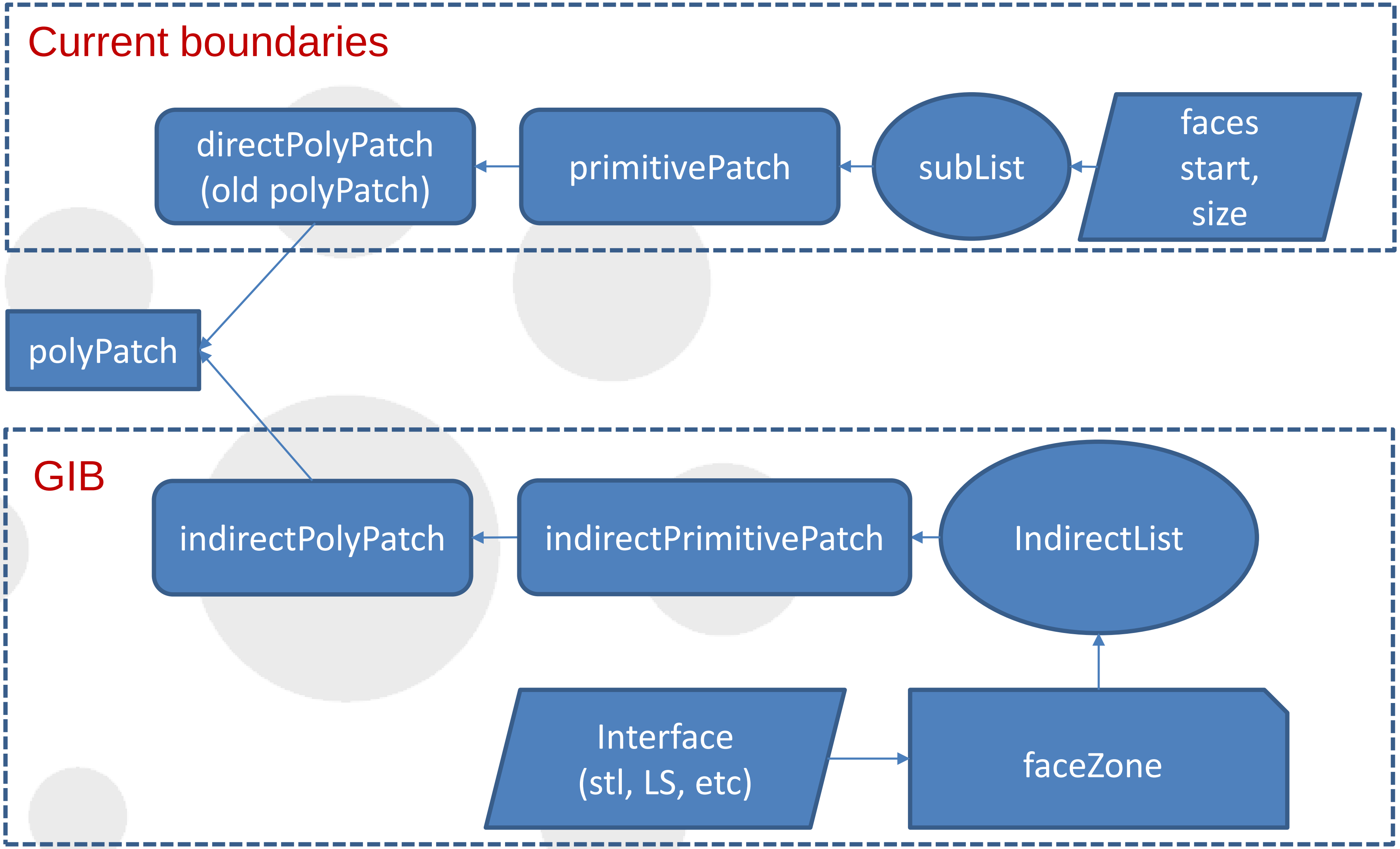
Methodology | Concept

- Perform snapping on the interface (LS, .stl, etc.)
- After snapping some of the faces are located on the interface.
- All the quantities needed from the finite volume are updated
- Problem: There is not a code structure in OpenFOAM® to apply boundary conditions in internal faces.





Methodology | Implementation



Methodology | Implementation

- Two new boundaries (one from each side) are constructed based on the `faceZone` and the `flipMap`.
  - faceZone represents the addressing of the face list that constructs the GIB
  - The flipMap boolean list is used to define the two sides of the GIB
- Based on `faceZone` and the `flipMap`, the geometric characteristics of the boundary (`Cf`, `Sf` etc) are calculated from the internal faces.

Methodology | Implementation

- Existing boundary conditions can be used on the GIB.
- The GIB boundary faces give the appropriate contributions to the matrix.
- The GIB can behave like:
  - a pass-through (like not existing)
  - normal boundary (`fixedValue`, `zeroGradient` etc)
  - boundary with communication (CHT tempetarure boundary condition)

Methodology | Implementation

Boundary file sample

```
7
(
        Inlet
        {
                type patch;
                physicalType inlet;
                nFaces 100;
                startFace 39700;
        }
        …
        ib1
        {
                type indirectWall;
                neighbourPatch ib2;
                faceZone ib;
                indirectPolyPatchType master;
                startFace 80300;
        }
        ib2
        {
                type indirectWall;
                neighbourPatch ib1;
                faceZone ib;
                indirectPolyPatchType slave;
                startFace 80300;
        }
)
```

U Boundary Conditions sample

```
…
boundaryField
{
        Inlet
        {
                surfaceNormalFixedValue;
                redValue uniform -1;
        }
        …
        ib1
        {
                type fixedValue;
                value uniform (0 0 0);
        }
        ib2
        {
                type fixedValue;
                value uniform (0 0 0);
        }
}
```

p Boundary Conditions sample

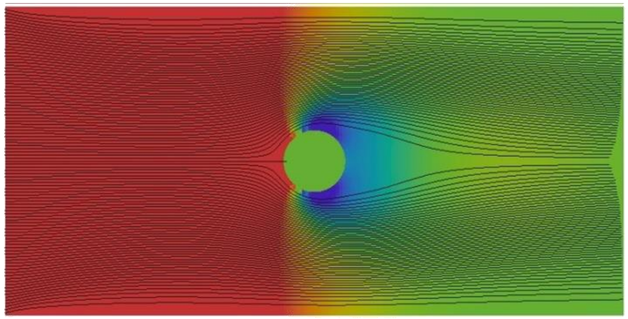
```
…
boundaryField
{
        Inlet
        {
                type zeroGradient;
        }
        …
        ib1
        {
                type zeroGradient;
        }
        ib2
        {
                type zeroGradient;
        }
}
```

Methodology | Implementation

- Changes in ~ 100 files
- OpenFoam library:
  - `polyMesh/patch` to insert the GIB classes
  - GeometricField to automate the operators
  - GAMG agglomerator
- finiteVolume library:
  - `fv(s)PatchField, fvPatch`
  - `fvm, fvc` operators
- Parallelization
- Mapping functions for moving GIB
- Wall distance for turbulence

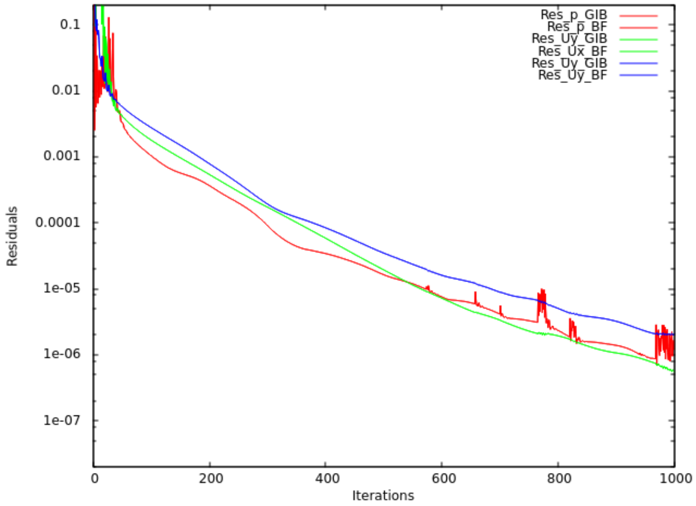
Contents

- Motivation
- Methodology
- Validation
- Applications
- Moving GIB
- Closing Comments



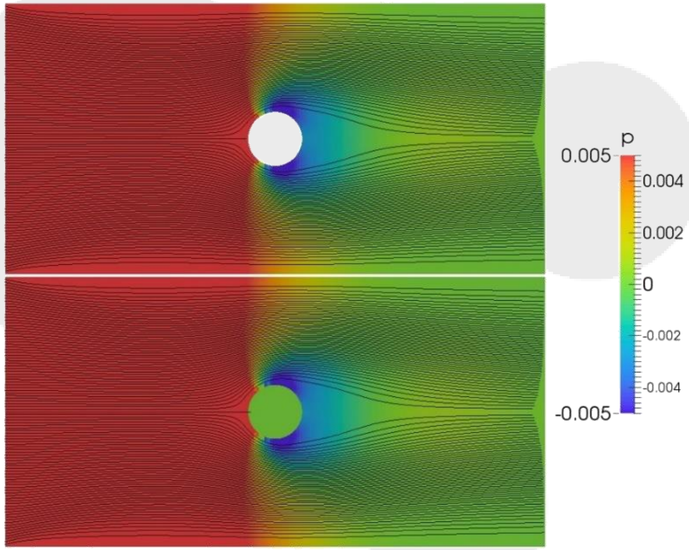
Validation | cylinder

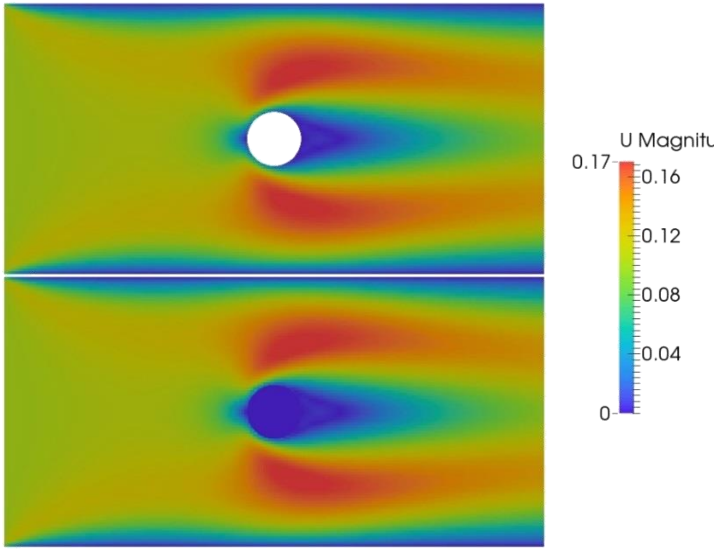
- Bodyfitted vs GIB cylinder results
- Identical residuals-results (machine accuracy)



Body-Fitted

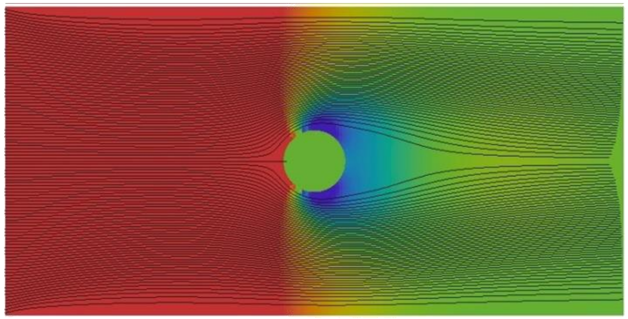
GIB





Contents

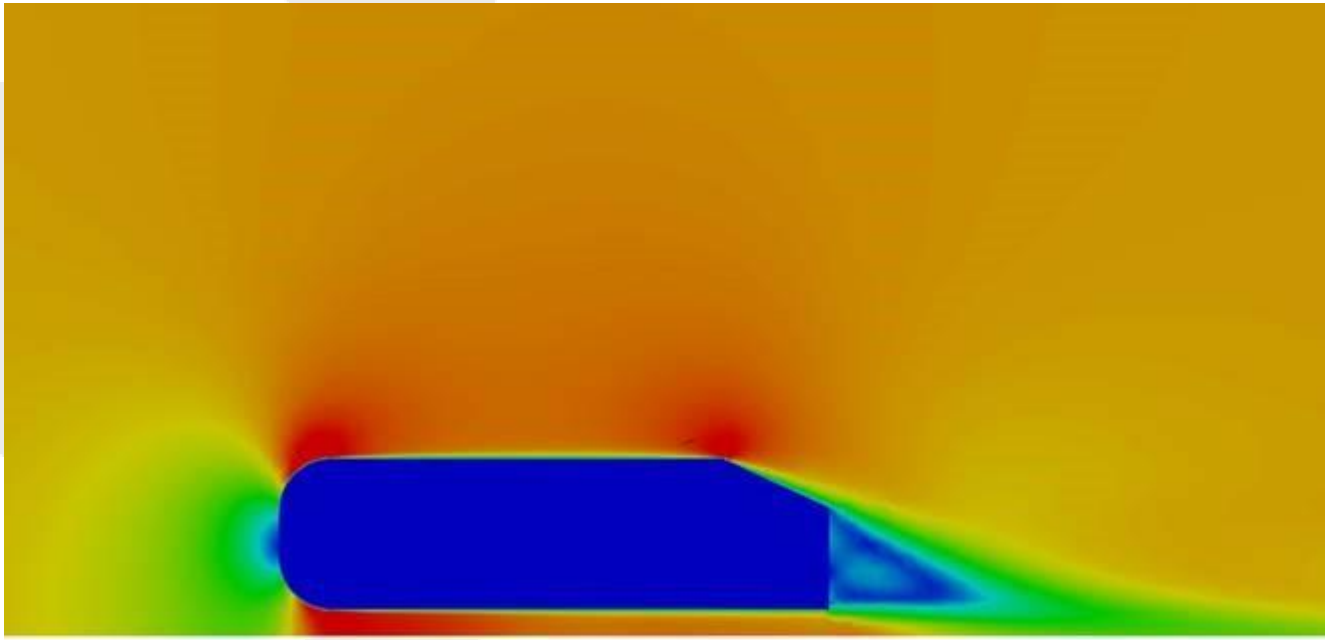
- Motivation
- Methodology
- Validation
- Applications
- Moving GIB
- Closing Comments



Applications | Ahmed

- Fully parallel
- Works with turbulence
- No top level change is required in the standard solvers

simpleFoam

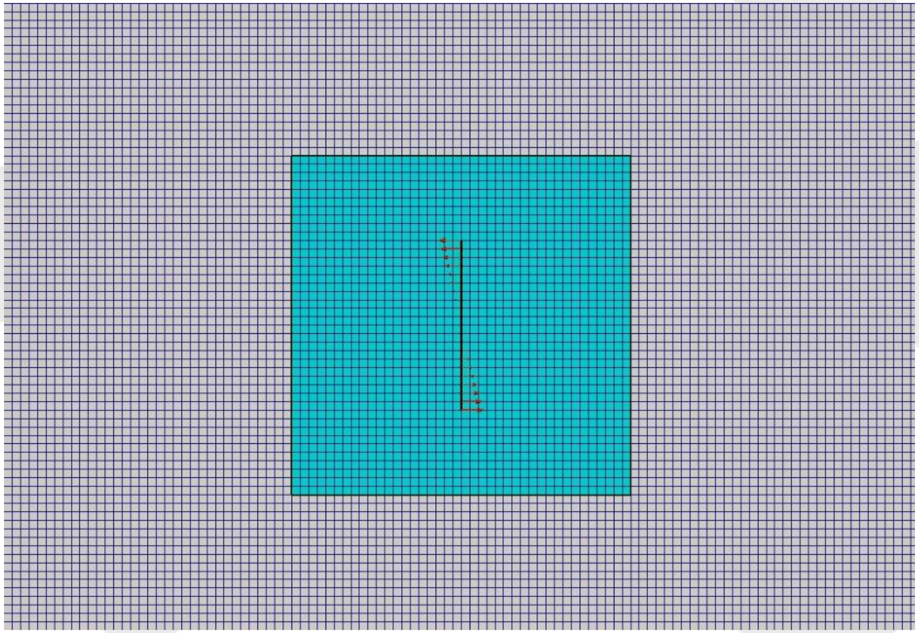


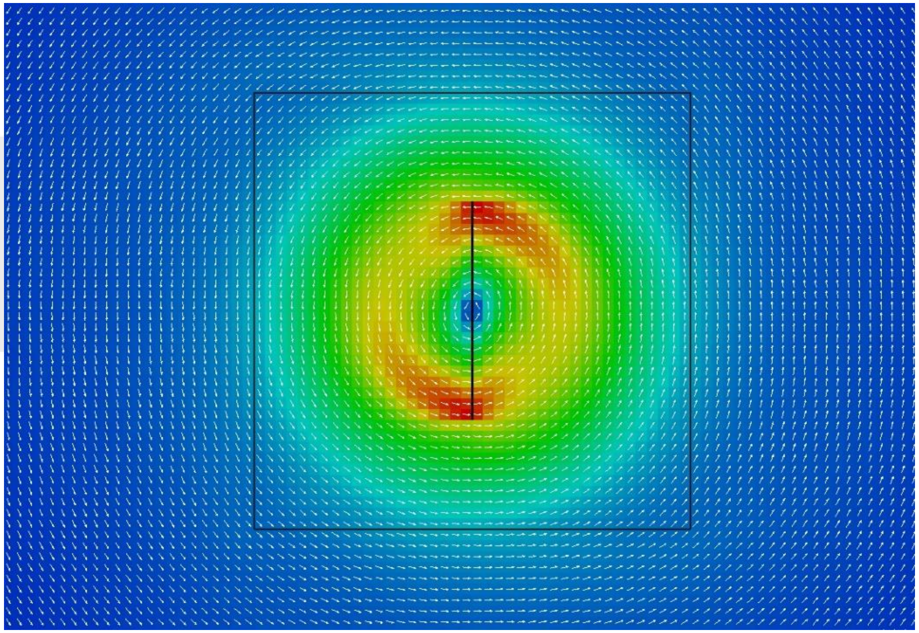
Applications | MRF | concept

- Current limitation:
  - The cellZone must be circular.
  - Reason: The relative and absolute fluxes should be the same at the interface of stationary and rotating part
- We apply GIB on the interface:
  - The pressure boundary is a pass-through
  - The velocity and the derived fields (phi, …) takes the value of the GIB wherever needed from the FV
  - The relative flux is added only in one side of the GIB

Applications | MRF | simpleMixer

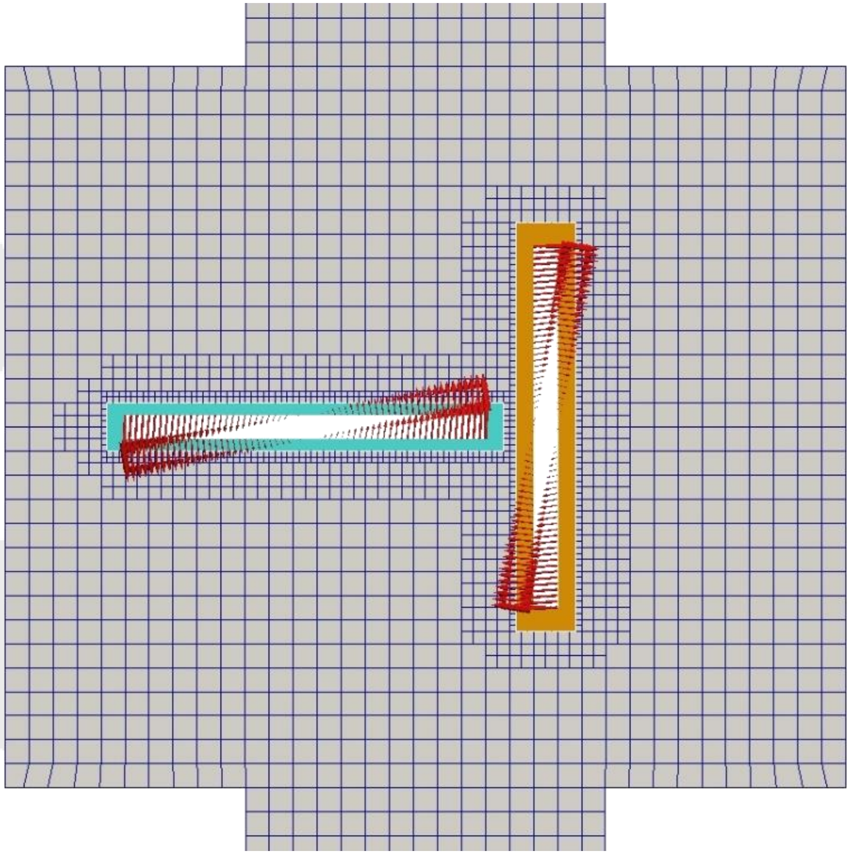
- Simple blockMesh geometry with a blade and a cellZone.
- GIB are applied at the perimeter of the cellZone



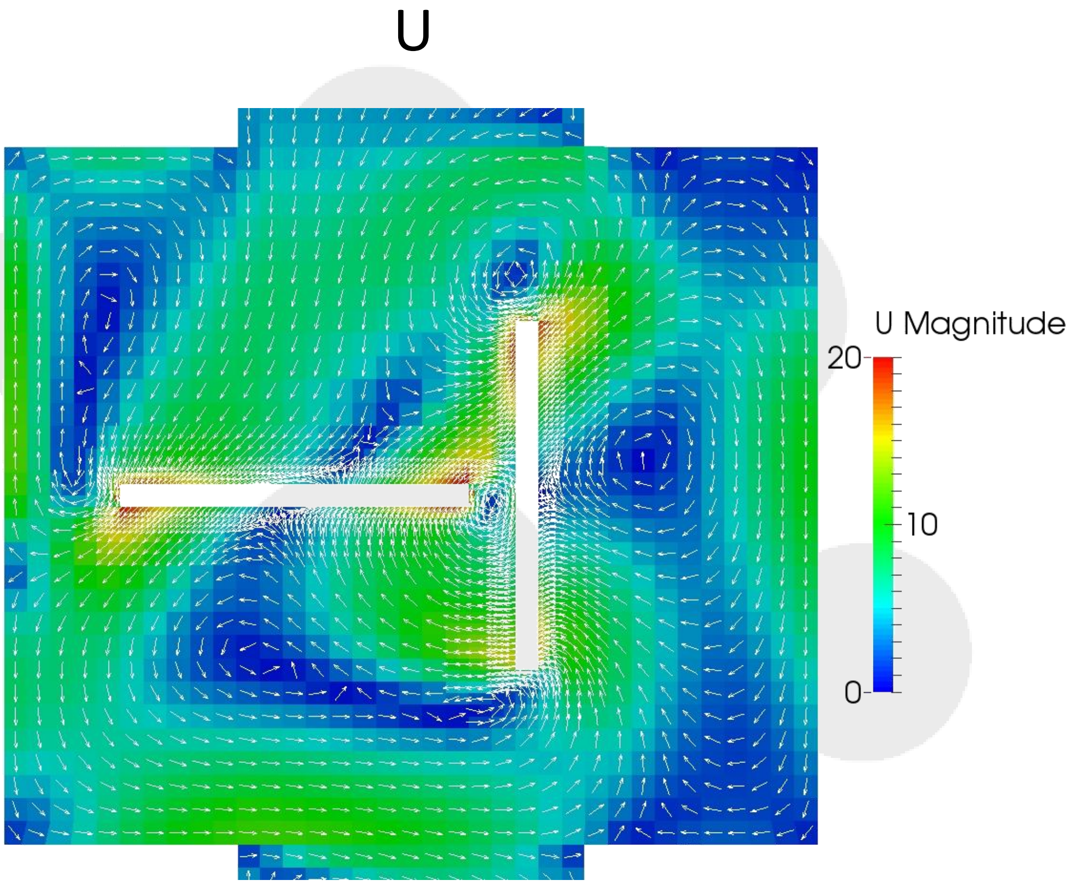


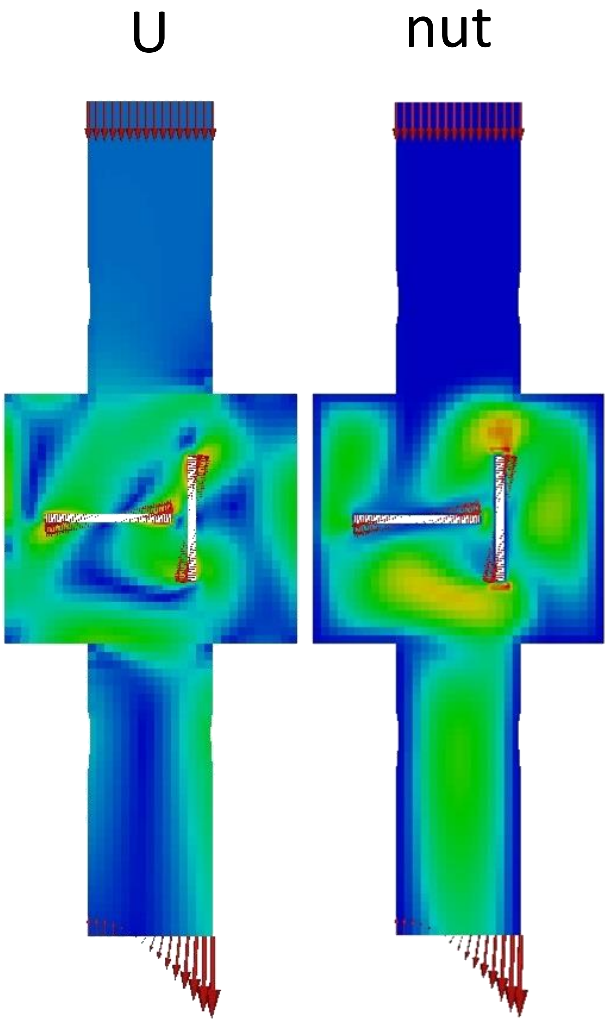
Applications | MRF | gearPump

- Pressure-pressure boundaries at the top and bottom
- Cyan and orange areas are two cellZones
- Two GIB are applied on the two interfaces (outside the two MRFzones).
- GIB boundary conditions are coupled (communication is required)
- The standard MRF method in OpenFOAM® not able to simulate the flow



Applications | MRF | gearPump



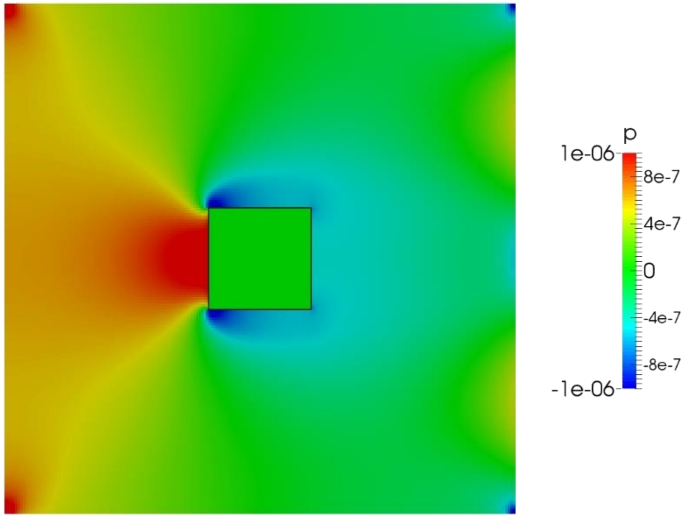


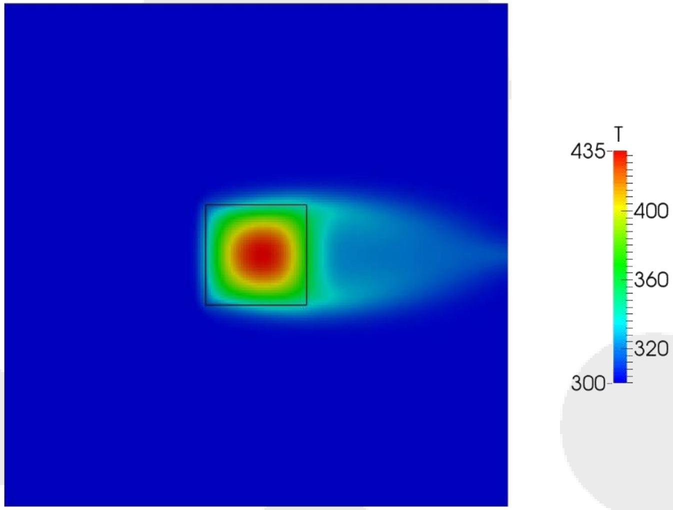
Applications | CHT | Current technique

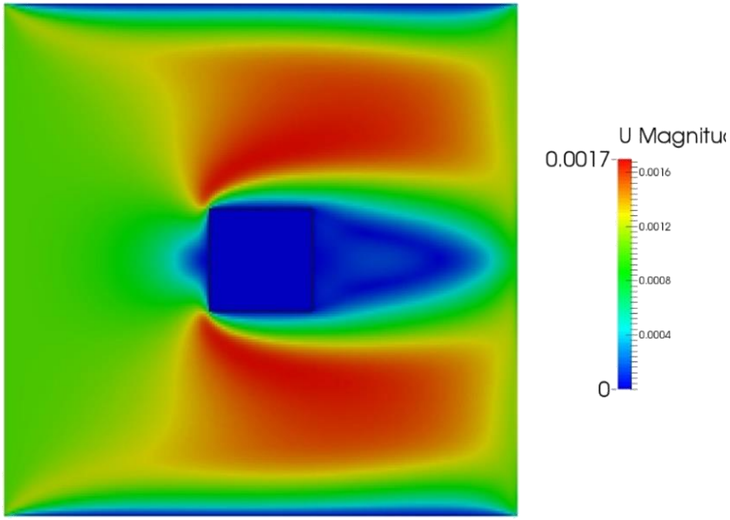
- Multi-region
- Everything is segregated. Basic equations:
  - For fluid:
    - Pressure
    - Velocity
    - Energy (enthalpy or temperature)
  - For solid:
    - Energy (enthalpy or temperature)
- Result: slow solution

Applications | CHT | GIB

- One region CHT
- The solid and fluid are communicating using GIB (black line)
- Boundary conditions for T or h are coupled because communication is required.
- Heat source is applied on the solid
- 1 matrix -> faster convergence

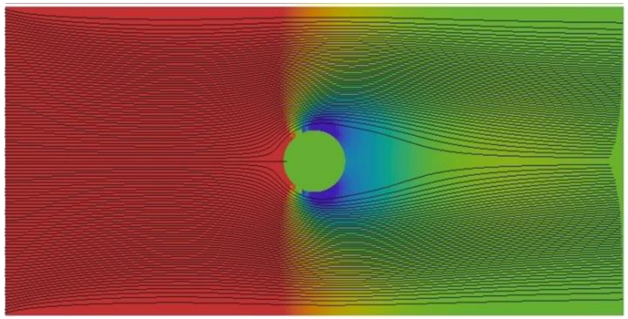






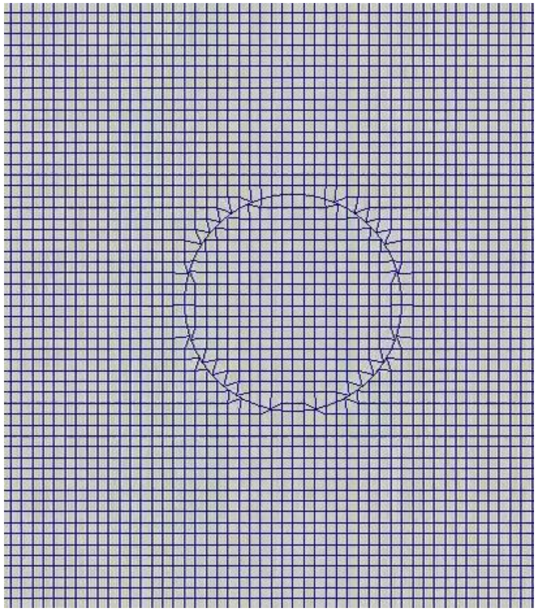
Contents

- Motivation
- Methodology
- Validation
- Applications
- Moving GIB
- Closing Comments

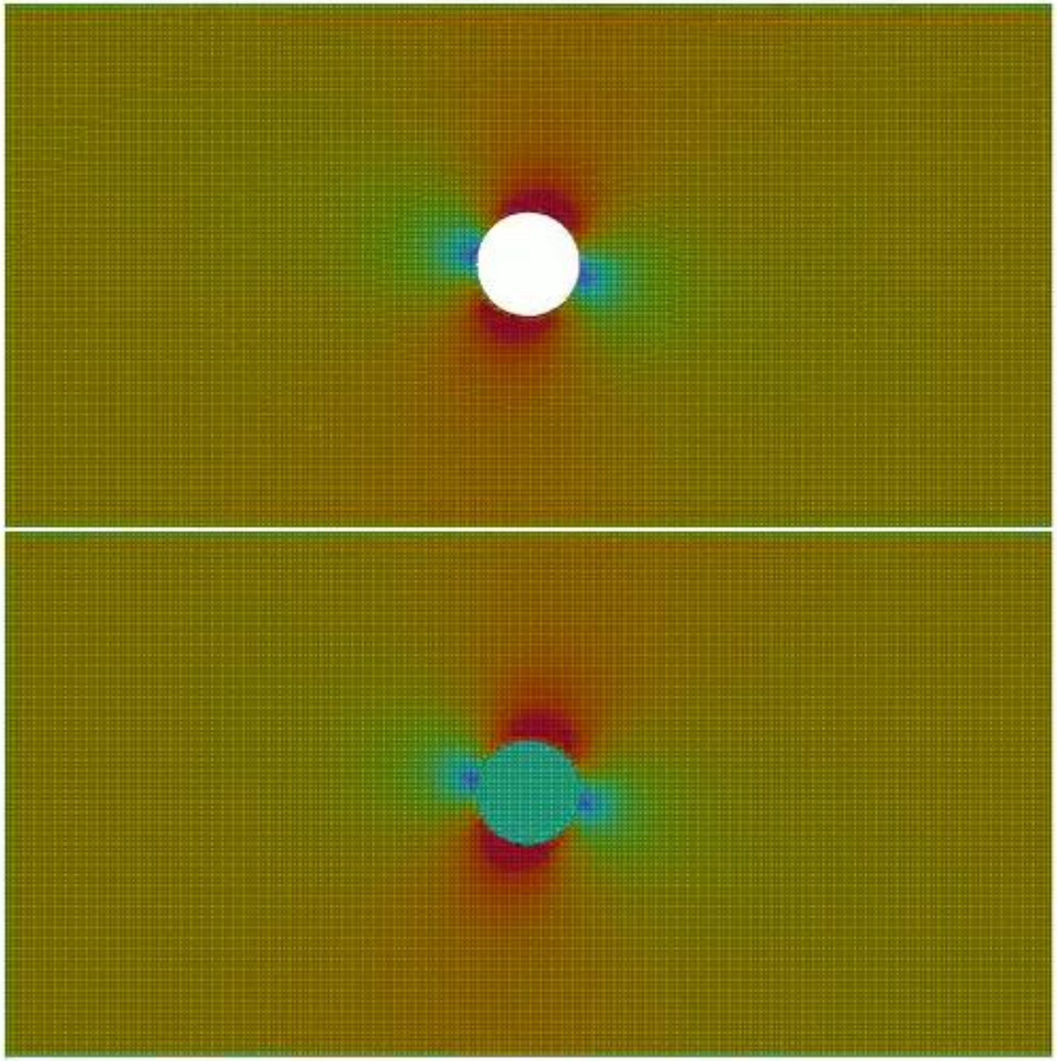


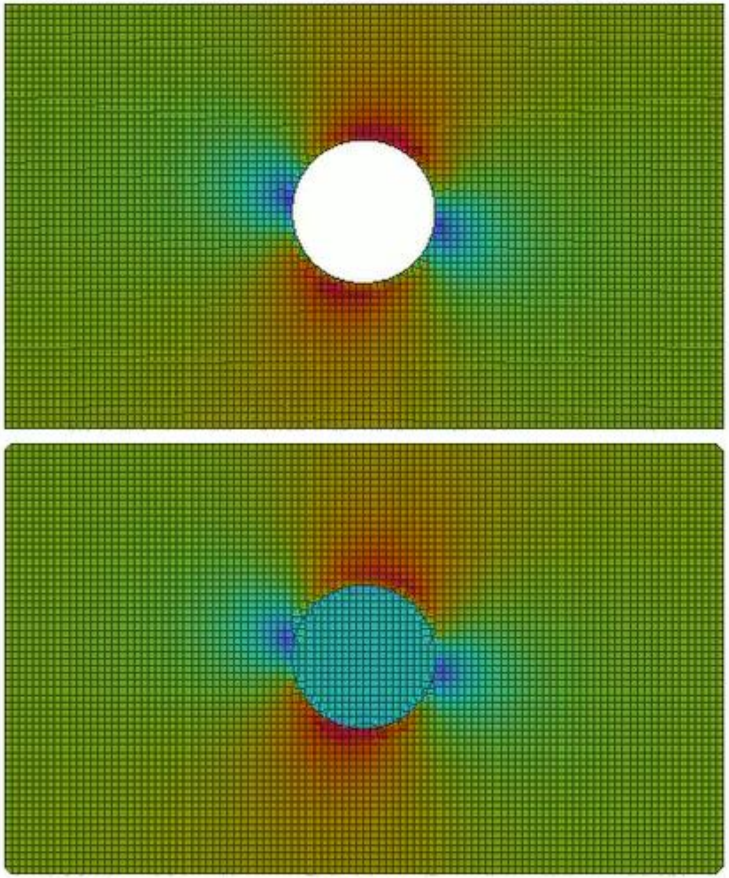
Moving GIB | Basic steps

- New location of the interface
- Perform snapping from the base mesh to the new interface
- faceZone update
- polyPatches class update
- GIB patch Fields update with mapping
- Special treatment for the freshly solid/fluid cells is needed

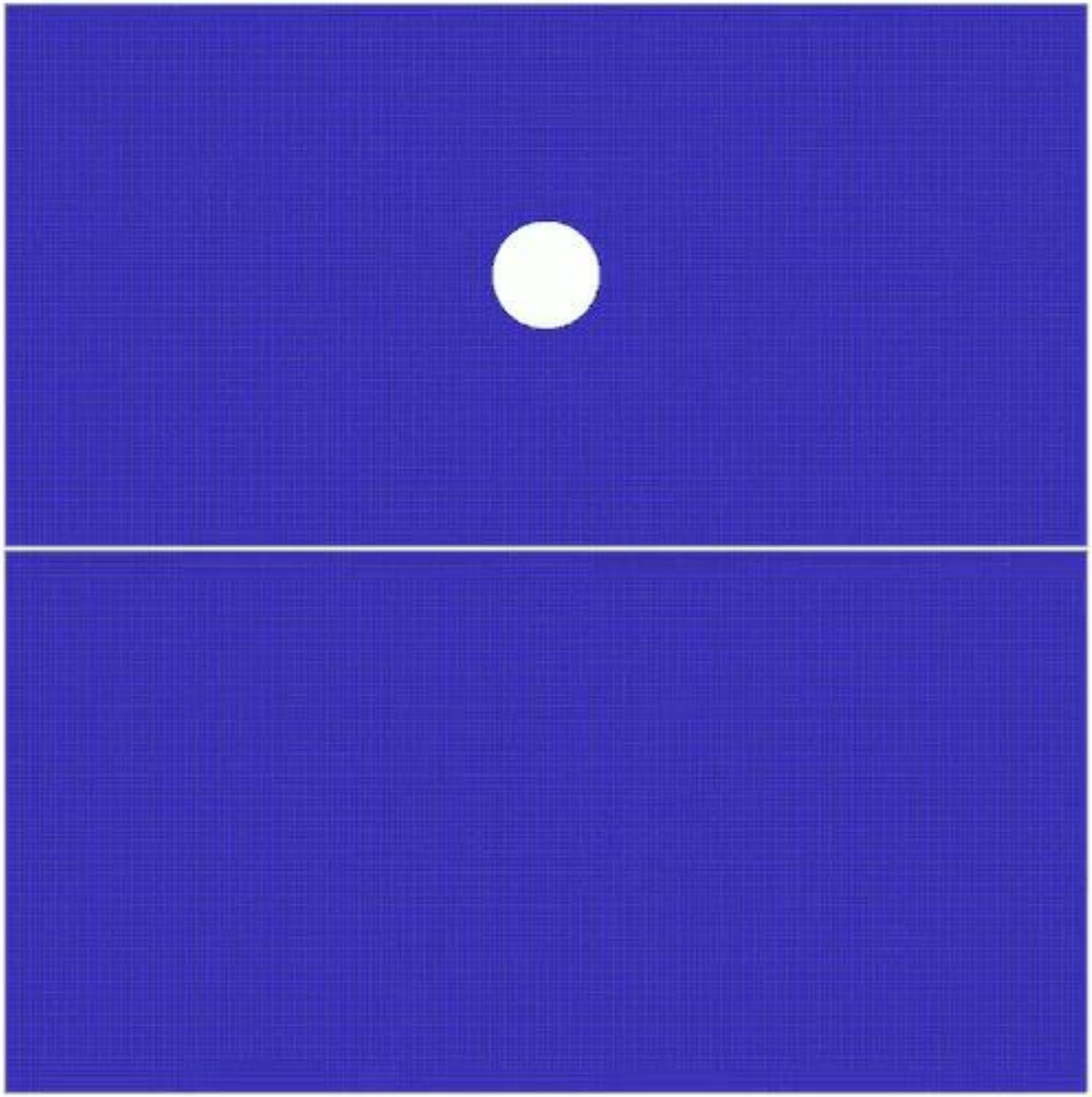


Moving GIB | movingCylinderBenchmark

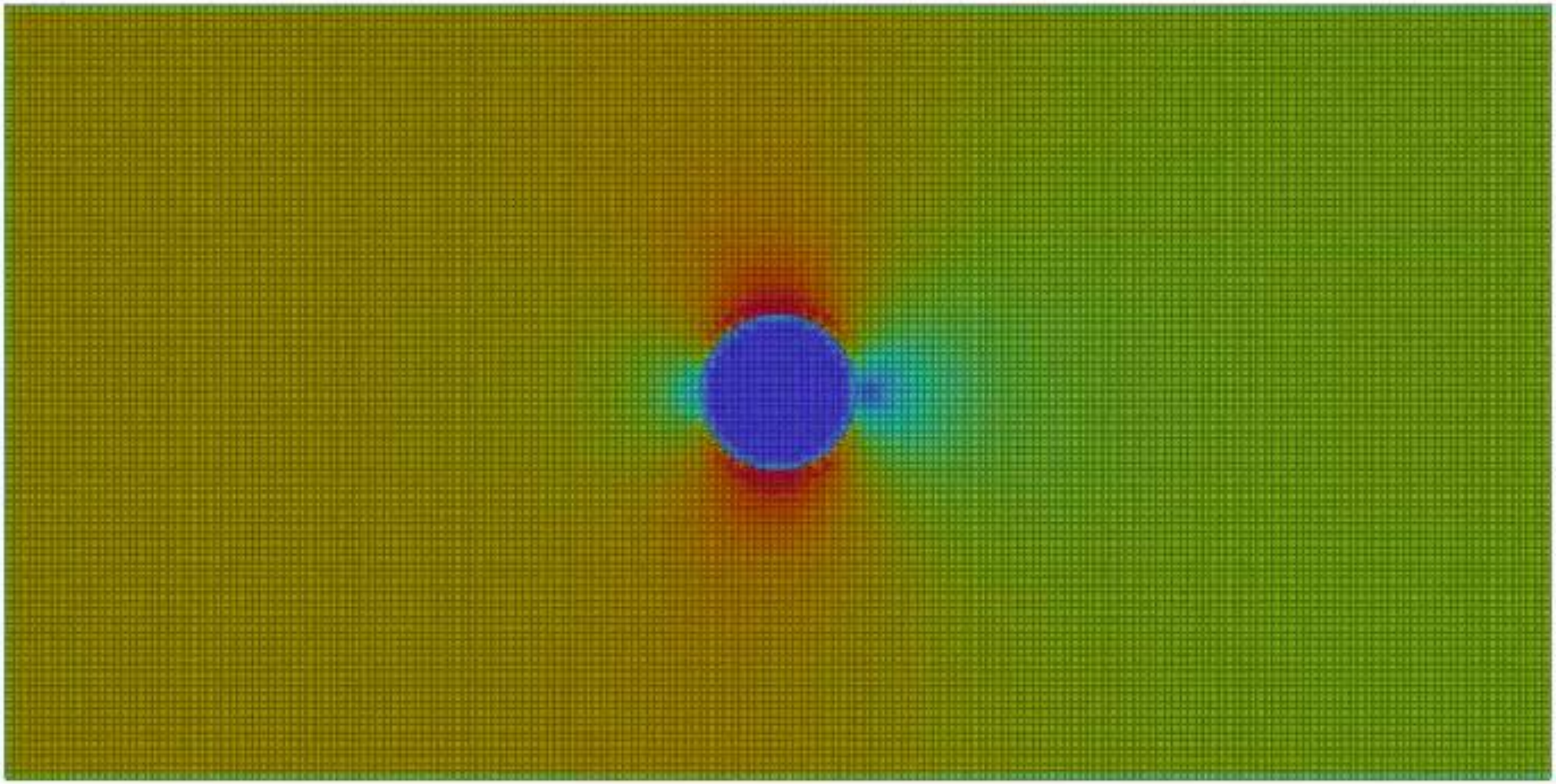




Moving GIB | crashingCylinder

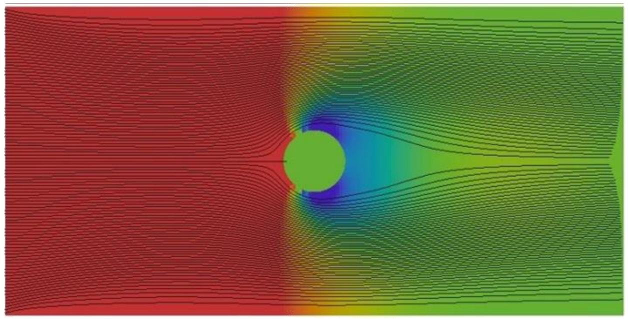


Moving GIB | growShrink Cylinder



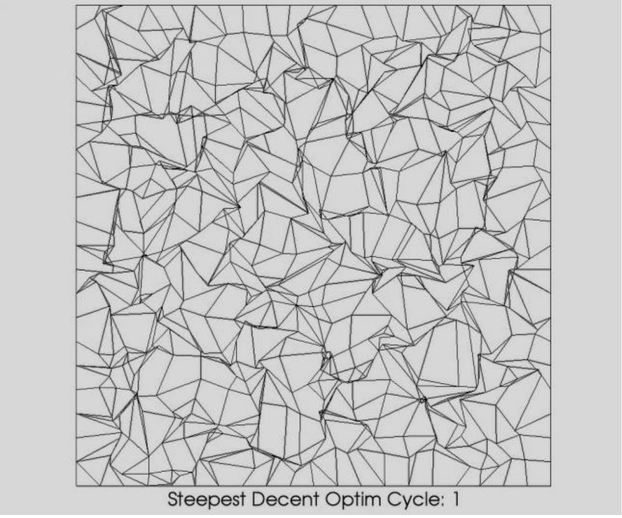
Contents

- Motivation
- Methodology
- Validation
- Applications
- Moving GIB
- Closing Comments



Closing comments | Next steps

- Closing comments:
  - New framework for applying boundary condition in internal group of faces is implemented
  - General implementation. Top level change is not required.
- Next steps:
  - Coupling with the adjointFoam engine
  - Mesh optimization for improving the mesh quality near the interface
  - Mesh adaptation



Looking forward

- It can be applied in every application with a static/moving interface.
- Applications such as:
    - CHT
    - MRF
    - FSI
    - multiphase
    - Gear pumps
    - 6 DoF
- Adjoint version of them
- Challenges: Add layers to the GIB with overset grids

The end

Thanks for your time! Any questions?