pUCoupledFoam - an open source coupled incompressible pressure-velocity solver based on foam-extend

Klas Jareteg[1], Vuko Vukčević[2], Hrvoje Jasak[2]

[1] klas.jareteg@chalmers.se,
Department of Applied Physics
Chalmers University of Technology, Sweden

[2] Faculty of Mechanical Engineering and Naval Architecture,
University of Zagreb, Croatia

June 23, 2014

Introduction

Background:

- General slow convergence of steady-state incompressible solvers
- Coupled solvers potentially give an increased convergence rate

pUCoupledFoam:

- Incompressible solver released with foam-extend-3.1
- Based on the block matrix format in foam-extend
- Extended block utilities and new operators allow a clear and efficient implementation

Implicit formulation

- Navier-Stokes, incompressible, steady-state equations:

$$\nabla \cdot (\mathbf{U}) = 0 \tag{1}$$

$$\nabla \cdot (\mathbf{U}\mathbf{U}) - \nabla(\nu \nabla \mathbf{U}) = -\frac{1}{\rho}\nabla p \tag{2}$$

- with the semi-discretized form:

$$\sum_{\text{faces}} \mathbf{U}_\text{f} \cdot \mathbf{S}_\text{f} = 0 \tag{3}$$

$$\sum_{\text{faces}} \left[\mathbf{U}\mathbf{U} - \nu \nabla \mathbf{U}\right]_\text{f} \cdot \mathbf{S}_\text{f} = -\sum_{\text{faces}} P_\text{f} \mathbf{S}_\text{f} \tag{4}$$

- Using a modified pressure:

$$\frac{p}{\rho} = P \tag{5}$$

- and Rhie-Chow interpolation in the continuity equation:

$$\sum \left[\overline{\mathbf{U}_\text{f}} - \overline{\mathbf{D}_\text{f}} \left(\nabla P_\text{f} - \overline{\nabla P_\text{f}}\right)\right] \cdot \mathbf{S}_\text{f} = 0 \tag{6}$$

Block matrix formulation

- Solution variable of length 4:

$$x^P = \begin{bmatrix} u^P \\ v^P \\ w^P \\ P^P \end{bmatrix} \tag{7}$$

Code 1: $FOAM_APP/solvers/coupled/pUCoupledFoam/createFields.H

```
volVector4Field Up
(
    IOobject
    (
        "Up",
        runTime.timeName(),
        mesh,
        IOobject::NO_READ,
        IOobject::AUTO_WRITE
    ),
    mesh,
    dimensionedVector4("zero", dimless, vector4::zero)
);
```

More information:

Training session: "Block coupled matrix solvers in foam-extend-3", Klas Jareteg and Ivor Clifford

Equation discretization - Momentum equation

- Momentum equation discretized using existing operators:

Code 2: $FOAM_APP/solvers/coupled/pUCoupledFoam/UEqn.H

```
fvVectorMatrix UEqn                                                    2
(                                                                      3
    fvm::div(phi, U)                                                   4
  + turbulence->divDevReff(U)                                          5
);                                                                     6
```

- and a new implicit gradient operator:

Code 3: $FOAM_APP/solvers/coupled/pUCoupledFoam/calculateCouplingMatrices.H

```
blockVectorMatrix pInU(fvm::grad(p));                                  1
```

Equation discretization - Continuity equation

- Continuity equation, with Rhie-Chow discretized in two different parts:

Code 4: $FOAM_APP/solvers/coupled/pUCoupledFoam/pEqn.H

```
fvScalarMatrix pEqn                                                    14
(                                                                      15
  - fvm::laplacian(rUAf, p)                                            16
 ==                                                                    17
  - fvc::div(presSource)                                               18
);                                                                     19
```

- and using a new implicit divergence operator:

Code 5: $FOAM_APP/solvers/coupled/pUCoupledFoam/calculateCouplingMatrices.H

```
blockVectorMatrix UInp(fvm::div(U));                                    2
```

- New operators allow the problem to be discretized without any hand assembling and with the generality of the run-time mechanisms

Matrix assembling

- Block matrix utilities extended such that the pressure and velocity equations can be automatically added:

Code 0.6: $FOAM_APP/solvers/coupled/pUCoupledFoam/pUCoupledFoam.C

```
blockMatrixTools::insertBlockCoupling(3, 0, UInp, U, A, b, false);          89
blockMatrixTools::insertBlockCoupling(0, 3, pInU, p, A, b, true);           90
```

- `blockMatrixTools` handles the coupled boundaries (e.g. processor)
- The utilities allows the pressure and velocity equations to modified without any further changes in the solver.

Case 1: Back facing step I

- Structured mesh, 4800 cells, laminar case
- Comparison of `simpleFoam` and `pUCoupledFoam`
- Under-relaxation in `pUCoupledFoam`: none in pressure, 0.995 in U

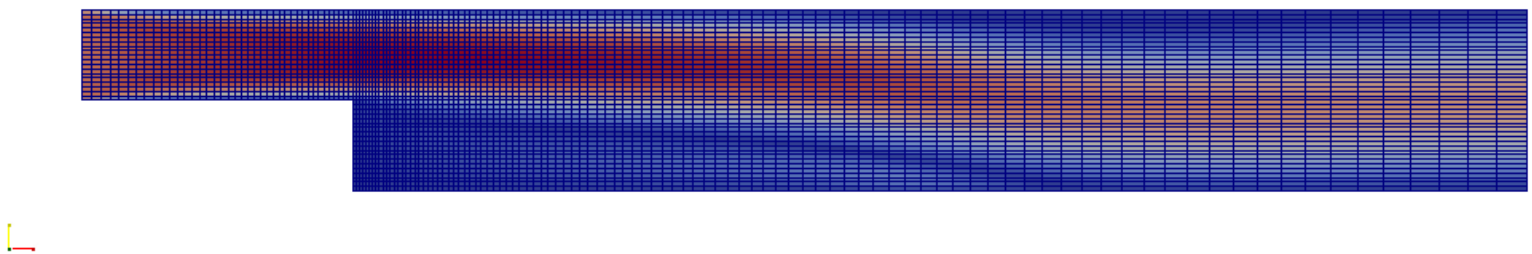

Figure: Geometry and velocity solution for back facing step case

- Major performance increase, both considering number of iterations and elapsed time

Case 1: Back facing step II

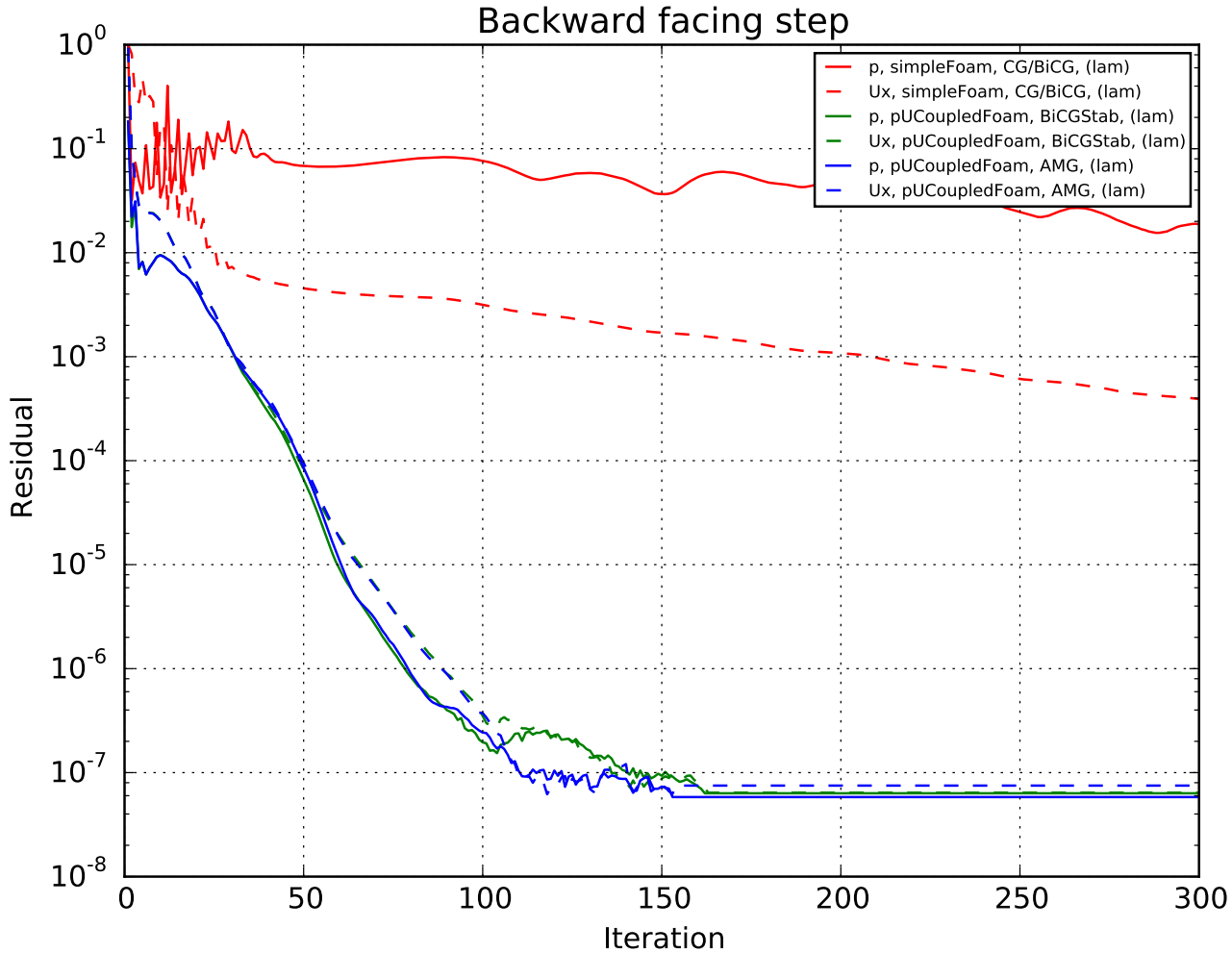

Figure: Performance of `simpleFoam` compared to `pUCoupledFoam`.

Case 1: Back facing step III

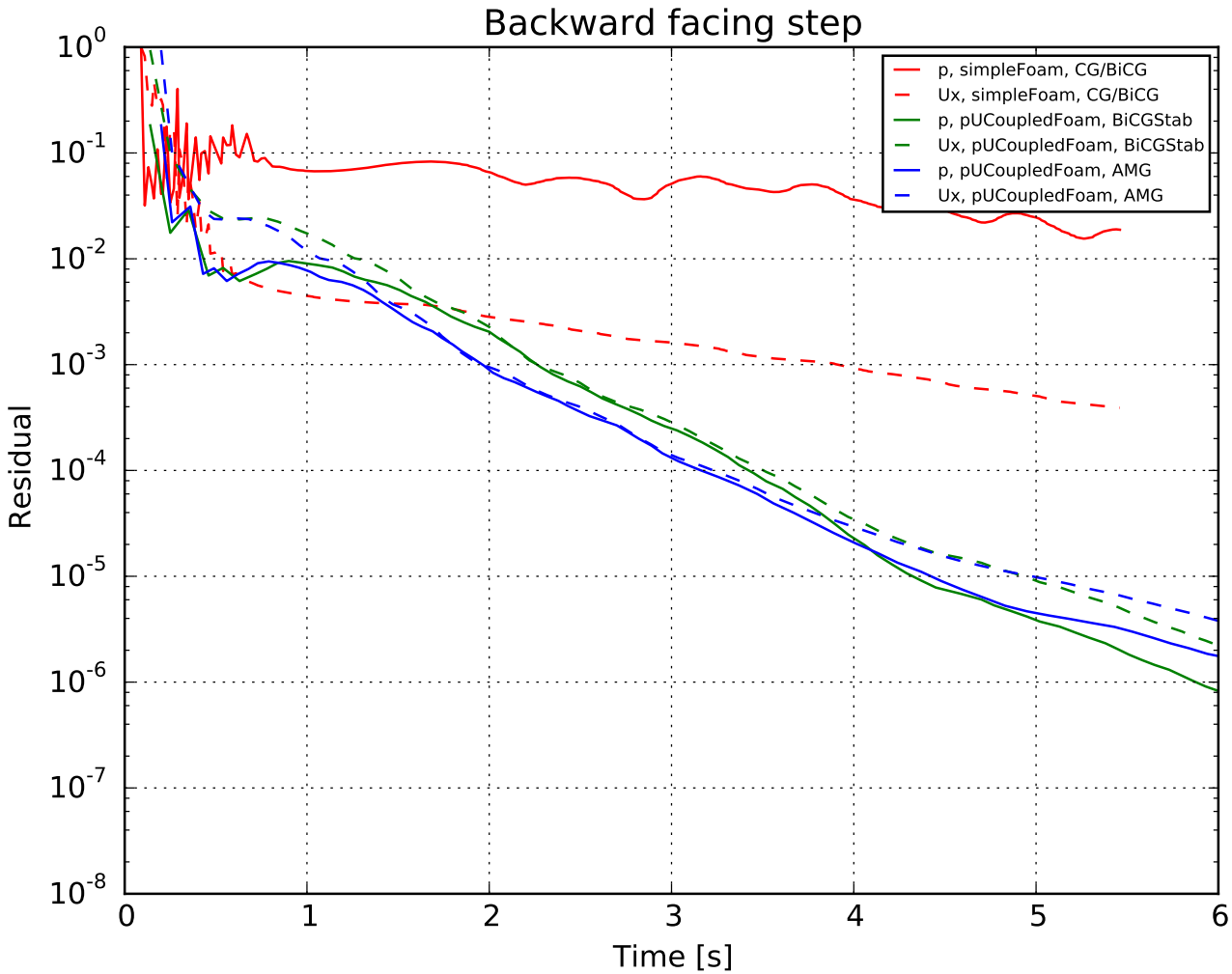

Figure: Performance of `simpleFoam` compared to `pUCoupledFoam`.

Case 2: Munk M3 airfoil I

- Unstructured mesh, 36410 cells, turbulence included
- Comparison of `simpleFoam` and `pUCoupledFoam`
- Under-relaxation in `pUCoupledFoam`: none in pressure, 0.85 in U

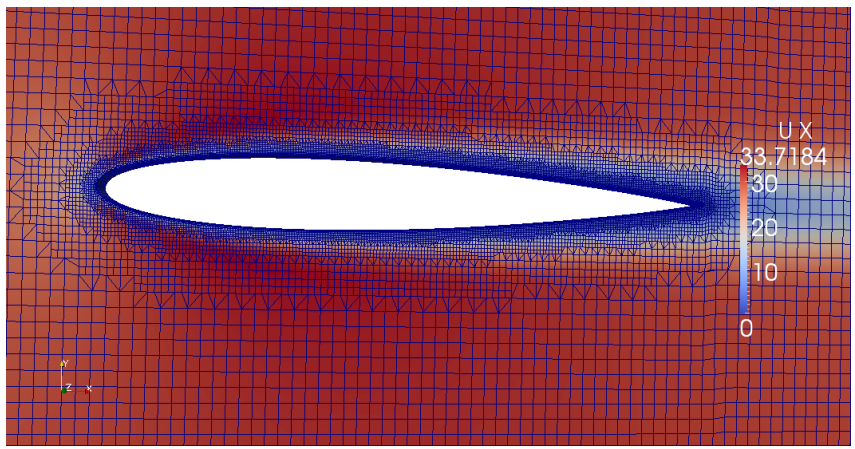

Figure: Geometry (zoomed) and pressure solution for Munk M3 airfoil case.

- Major performance increase, both considering number of iterations and elapsed time
- Best performance for the AMG solver

Case 2: Munk M3 airfoil II

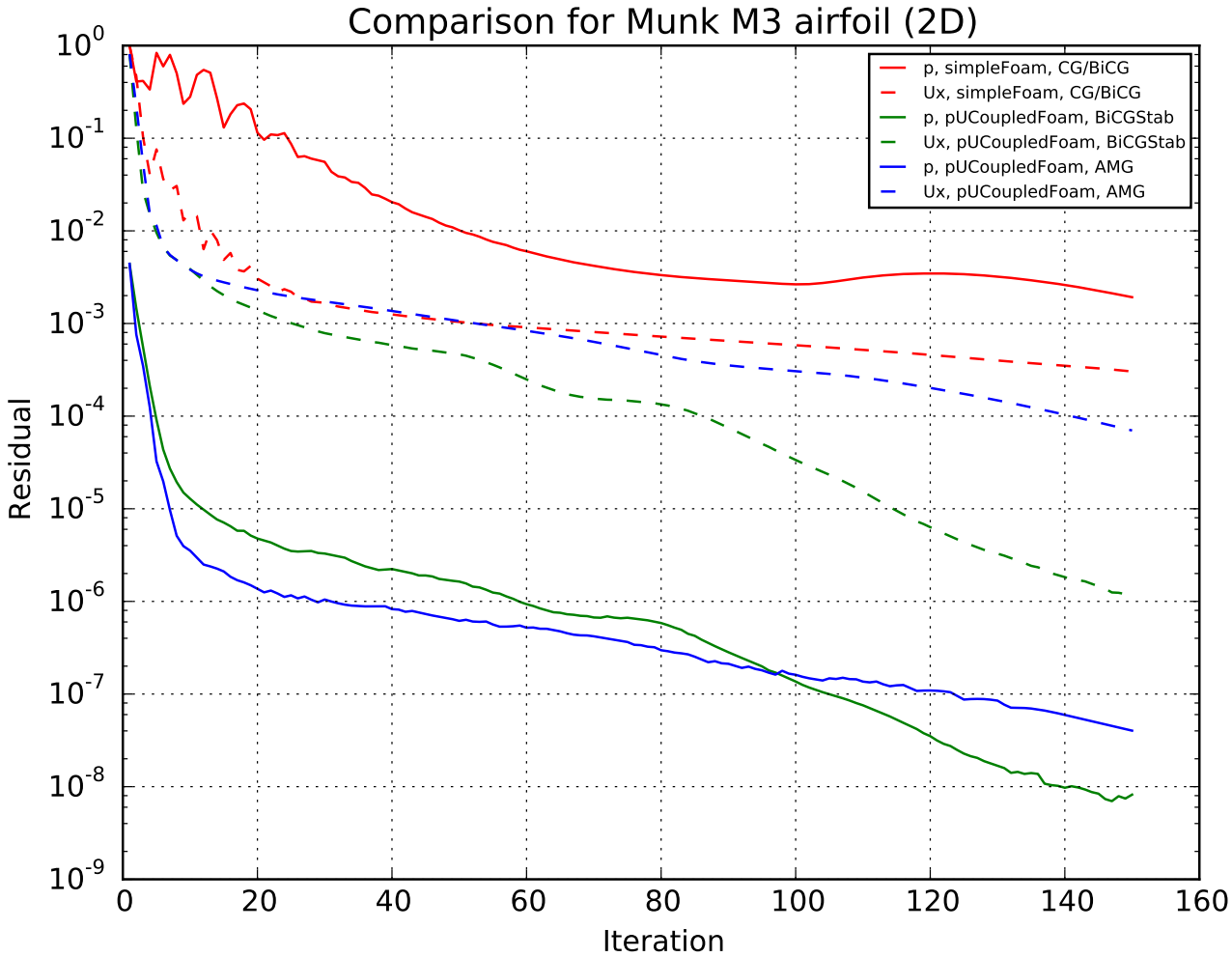

Figure: Performance of `simpleFoam` compared to `pUCoupledFoam`.

Case 2: Munk M3 airfoil III

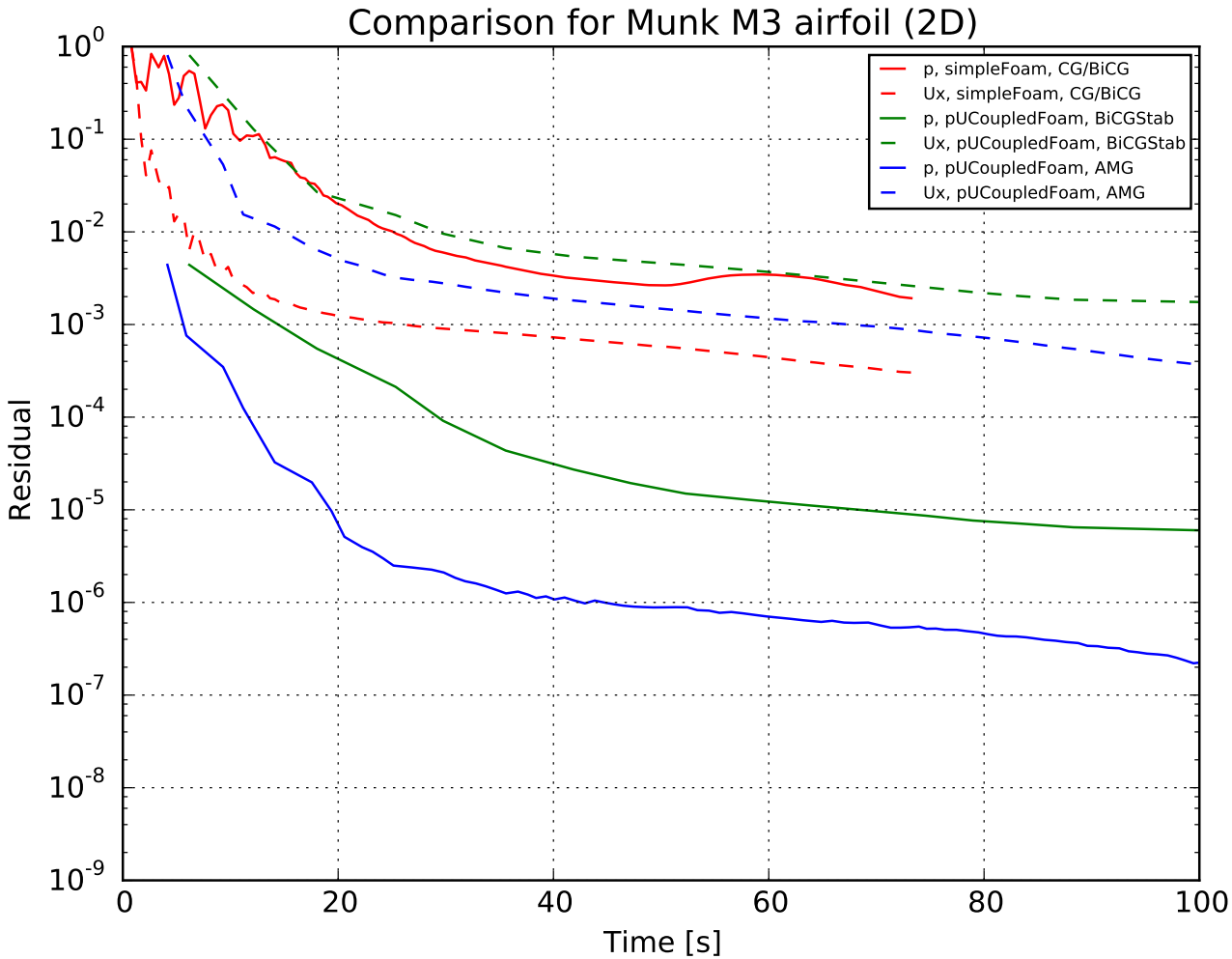

Figure: Performance of `simpleFoam` compared to `pUCoupledFoam`.

Case 3: Munk M3 wing I

- Untructured mesh, 1.4M cells, turbulence included
- Comparison of `simpleFoam` and `pUCoupledFoam`
- Under-relaxation in `pUCoupledFoam`: none in pressure, 0.85 (BiCG) and 0.7 (AMG) in U
- Large increase in performance, with best performance from the AMG solver.

Case 3: Munk M3 wing II

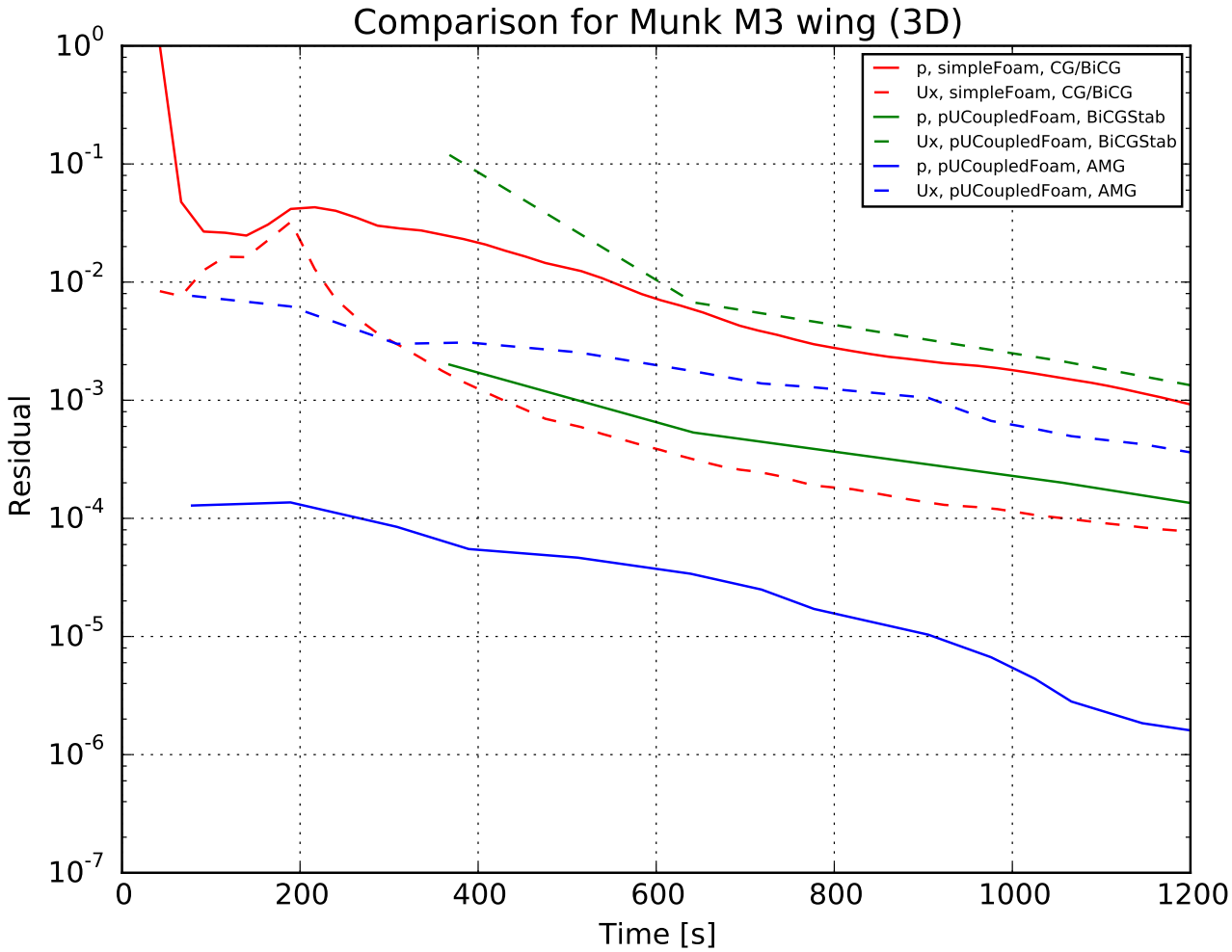

Figure: Performance of `simpleFoam` compared to `pUCoupledFoam`.

Case 3: Munk M3 wing III

- Comparison of parallel performance for `simpleFoam` and `pUCoupledFoam` (BiCG)

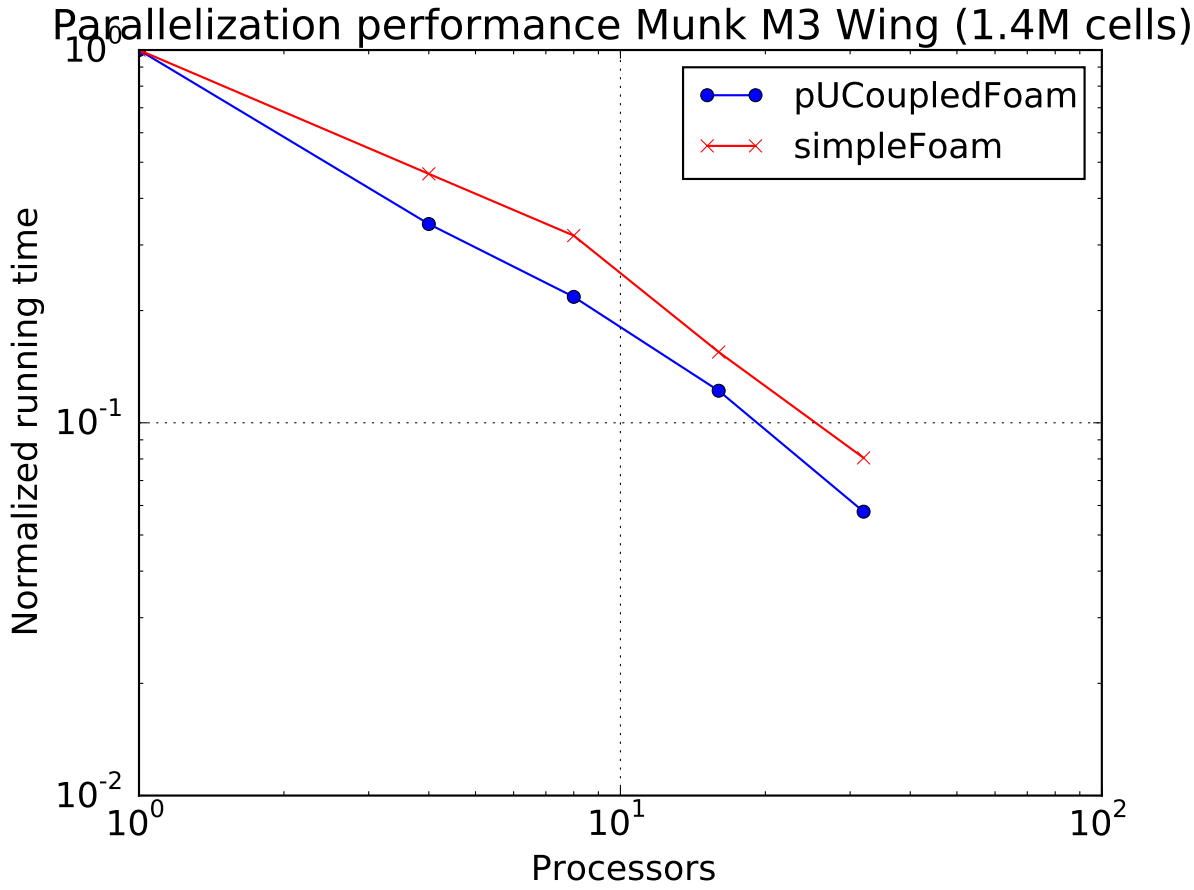

Figure: Speed up from parallelization.

Case 3: Munk M3 wing IV

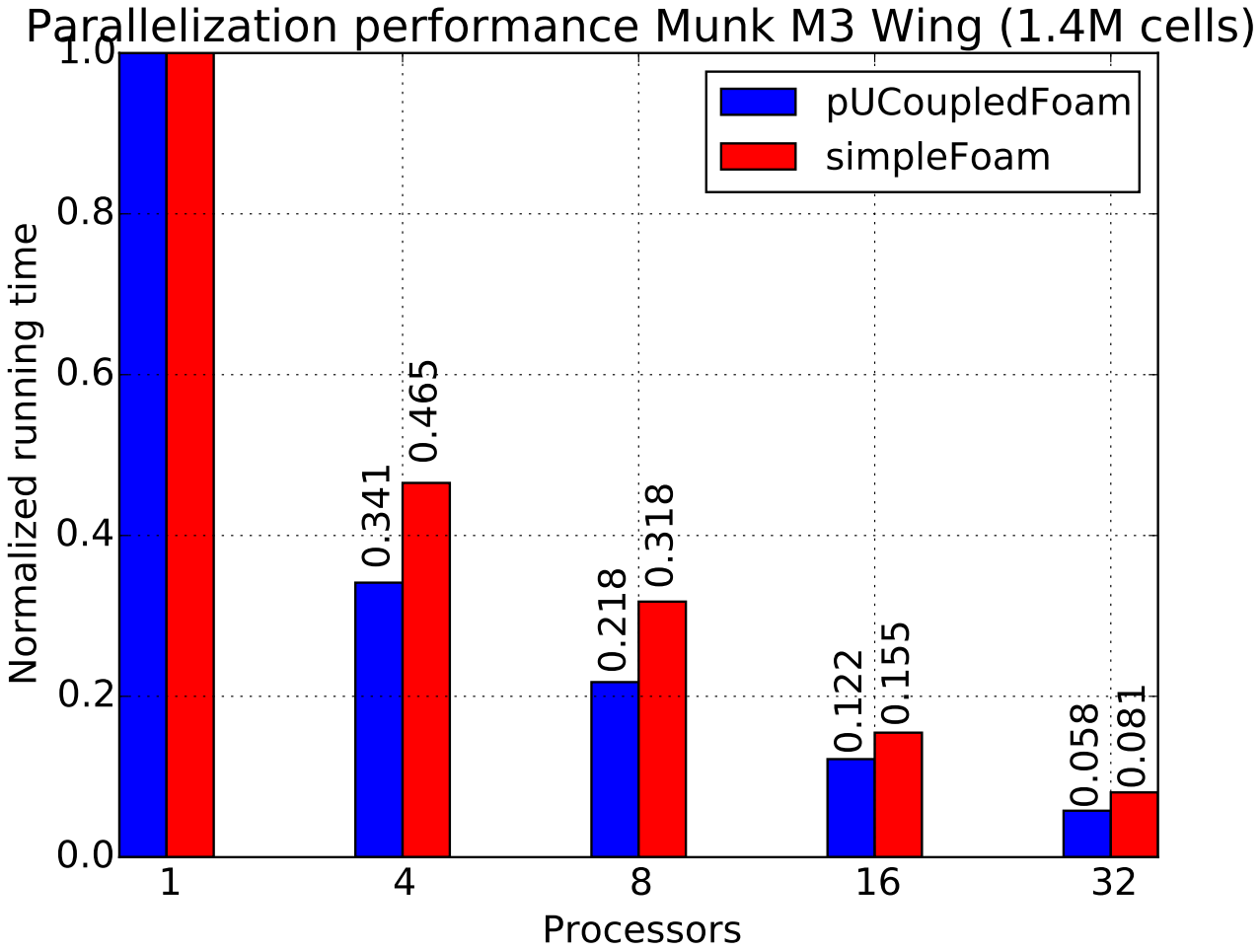

Figure: Relative elapsed time as compared to single processor.

Summary

- `puCoupledFoam`: an open source incompressible block coupled solver
- General implementation, relying on new implicit operators
- Convergence rate increased as compared to segregated solver, for single CPU as well as in parallel.
- Easy to extend due to the use of operators and block utilities