Science and Technology Facilities Council

Scientific Computing

CCP-WSI

a Collaborative Computational Project in Wave Structure Interaction

Introduction to programming solvers in OpenFOAM®

Dr. Wendi Liu

Table of Content

Science and Technology Facilities Council
Scientific Computing

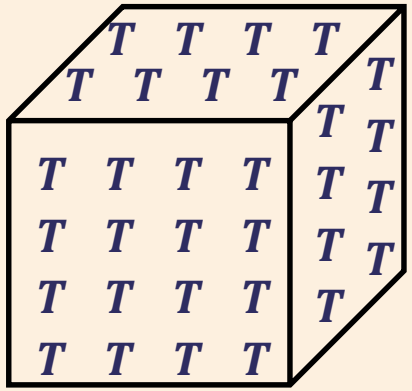
Scientific Computing

Building Blocks in OpenFOAM®

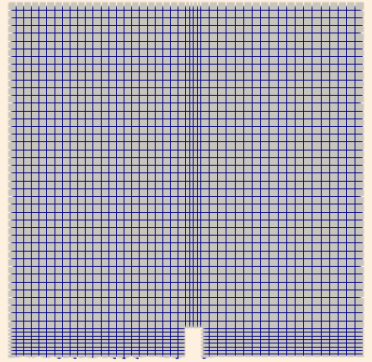
Building Blocks in OpenFOAM®

$$\boldsymbol{T} = T_{ij} = \begin{pmatrix} T_{11} & T_{12} & \cdots & T_{1n} \\ T_{21} & T_{22} & \cdots & T_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ T_{m1} & T_{m2} & \cdots & T_{mn} \end{pmatrix}$$

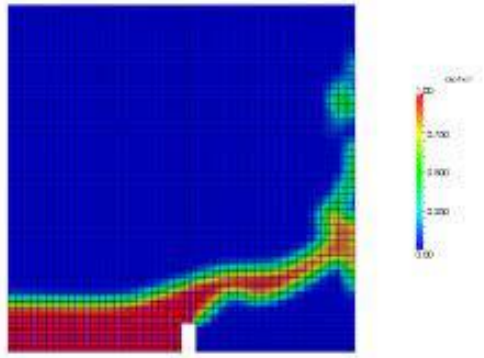
Tensor



Field



Discretisation



Solver

Science and Technology Facilities Council

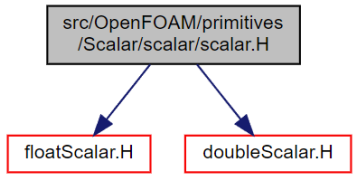
Scientific Computing

Building Blocks in OpenFOAM®

Tensor classes

$$s$$

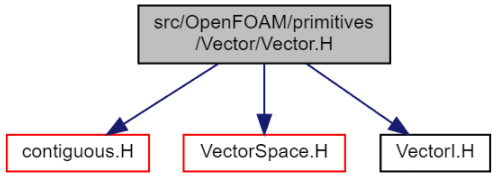
Rank 0



Scalar

$$\boldsymbol{v} = v_i = (v_1 \quad v_2 \quad v_3)$$

Rank 1

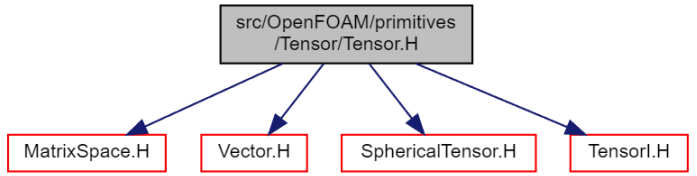


Access function:
x(), y(), z()

Vector

$$\boldsymbol{T} = T_{ij} = \begin{pmatrix} T_{11} & T_{12} & \cdots & T_{1n} \\ T_{21} & T_{22} & \cdots & T_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ T_{m1} & T_{m2} & \cdots & T_{mn} \end{pmatrix}$$

Rank 2



Access function:
xx(), xy(), xz() …

Tensor

Science and Technology Facilities Council
Scientific Computing

Building Blocks in OpenFOAM®

Algebraic tensor operations – snapshots from Programmer's Guide v2212

Operation	Comment	Mathematical Description	Description in OpenFOAM
Addition		$\mathbf{a}+\mathbf{b}$	`a + b`
Subtraction		$\mathbf{a}-\mathbf{b}$	`a - b`
Scalar multiplication		$s\mathbf{a}$	`s * a`
Scalar division		$\mathbf{a}/s$	`a / s`
Outer product	rank $\mathbf{a},\mathbf{b} >= 1$	$\mathbf{ab}$	`a * b`
Inner product	rank $\mathbf{a},\mathbf{b} >= 1$	$\mathbf{a}\cdot\mathbf{b}$	`a & b`

Continued on next page

Continued from previous page

Operation	Comment	Mathematical Description	Description in OpenFOAM
Double inner product	rank $\mathbf{a},\mathbf{b} >= 2$	$\mathbf{a}:\mathbf{b}$	`a && b`
Cross product	rank $\mathbf{a},\mathbf{b} = 1$	$\mathbf{a}\times\mathbf{b}$	`a ^ b`
Square		$\mathbf{a}^2$	`sqr(a)`
Magnitude squared		$\lvert\mathbf{a}\rvert^2$	`magSqr(a)`
Magnitude		$\lvert\mathbf{a}\rvert$	`mag(a)`
Power	$n = 0, 1, ..., 4$	$\mathbf{a}^n$	`pow(a,n)`
Component average	$i = 1, ..., N$	$\overline{a_i}$	`cmptAv(a)`
Component maximum	$i = 1, ..., N$	$\max(a_i)$	`cmptMax(a)`
Component minimum	$i = 1, ..., N$	$\min(a_i)$	`cmptMin(a)`
Scale		scale($\mathbf{a}$,$\mathbf{b}$)	`cmptMultiply(a,b)`
Geometric transformation	transforms $\mathbf{a}$ using tensor $\mathbf{T}$		`transform(T,a)`

Operations exclusive to tensors of rank 2

Transpose	$\mathbf{T}^T$	`T.T()`
Diagonal	diag $\mathbf{T}$	`diag(T)`
Trace	tr $\mathbf{T}$	`tr(T)`
Deviatoric component	dev $\mathbf{T}$	`dev(T)`
Symmetric component	symm $\mathbf{T}$	`symm(T)`
Skew-symmetric component	skew $\mathbf{T}$	`skew(T)`
Determinant	det $\mathbf{T}$	`det(T)`
Cofactors	cof $\mathbf{T}$	`cof(T)`
Inverse	inv $\mathbf{T}$	`inv(T)`
Hodge dual	$*\mathbf{T}$	`*T`

Operations exclusive to scalars

Sign (boolean)		sgn(s)	`sign(s)`
Positive (boolean)		$s >= 0$	`pos(s)`
Negative (boolean)		$s < 0$	`neg(s)`
Limit	n scalar	limit(s, n)	`limit(s,n)`
Square root		$\sqrt{s}$	`sqrt(s)`
Exponential		exp s	`exp(s)`
Natural logarithm		ln s	`log(s)`
Base 10 logarithm		$\log_{10} s$	`log10(s)`
Sine		sin s	`sin(s)`
Cosine		cos s	`cos(s)`
Tangent		tan s	`tan(s)`
Arc sine		asin s	`asin(s)`
Arc cosine		acos s	`acos(s)`
Arc tangent		atan s	`atan(s)`
Hyperbolic sine		sinh s	`sinh(s)`
Hyperbolic cosine		cosh s	`cosh(s)`
Hyperbolic tangent		tanh s	`tanh(s)`
Hyperbolic arc sine		asinh s	`asinh(s)`
Hyperbolic arc cosine		acosh s	`acosh(s)`
Hyperbolic arc tangent		atanh s	`atanh(s)`
Error function		erf s	`erf(s)`
Complement error function		erfc s	`erfc(s)`

Continued on next page

Continued from previous page

Operation	Comment	Mathematical Description	Description in OpenFOAM
Logarithm gamma function		ln Γs	`lgamma(s)`
Type 1 Bessel function of order 0		J_0 s	`j0(s)`
Type 1 Bessel function of order 1		J_1 s	`j1(s)`
Type 2 Bessel function of order 0		Y_0 s	`y0(s)`
Type 2 Bessel function of order 1		Y_1 s	`y1(s)`

$\mathbf{a}, \mathbf{b}$ are tensors of arbitrary rank unless otherwise stated
s is a scalar, N is the number of tensor components

Table 2.2: Algebraic tensor operations in OpenFOAM

Building Blocks in OpenFOAM®

Dimensional units

- Units are defined using the `dimensionSet` class

```
dimensionSet pressureDims(1, -1, -2, 0, 0, 0, 0);
```

No.	Property	Unit	Symbol
1	Mass	kilogram	k
2	Length	metre	m
3	Time	second	s
4	Temperature	Kelvin	K
5	Quantity	moles	mol
6	Current	ampere	A
7	Luminous intensity	candela	cd

S.I. base units of measurement

Building Blocks in OpenFOAM®

Field class

- Lists of the tensor classes are defined by `Field<Type>`
- All instances of `Field<Type>` are renamed using `typedef` as `tensorField, vectorField, etc.`

Field< Type > Class Template Reference

Generic templated field type. More...

Inherits **FieldBase**, and **List< Type >**.

Inherited by **CompactIOField< Foam::Field< label >, label >, CompactIOField< Foam::Field< scalar >, scalar >, CompactIOField< Foam::Field< Foam::Vector >, Foam::Vector >, CompactIOField< Foam::Field< Foam::SphericalTensor >, Foam::SphericalTensor >, CompactIOField< Foam::Field< Foam::SymmTensor >, Foam::SymmTensor >, CompactIOField< Foam::Field< Foam::Tensor >, Foam::Tensor >, DimensionedField< symmTensor, Foam::volMesh >, DimensionedField< symmTensor, volMesh >, DimensionedField< scalar, Foam::volMesh >, DimensionedField< scalar, volMesh >, DimensionedField< Type, pointMesh >, DimensionedField< vector, Foam::pointMesh >, DimensionedField< vector, Foam::volMesh >, DimensionedField< vector, volMesh >, DimensionedField< tensor, Foam::volMesh >, DimensionedField< tensor, volMesh >, DimensionedField< Foam::Vector, Foam::volMesh >, DimensionedField< Foam::Vector, Foam::pointMesh >, DimensionedField< scalar, Foam::surfaceMesh >, DimensionedField< scalar, surfaceMesh >, DimensionedField< scalar, Foam::pointMesh >, DimensionedField< scalar, pointMesh >, DimensionedField< vector, pointMesh >, DimensionedField< scalar, Foam::areaMesh >, DimensionedField< scalar, areaMesh >, DimensionedField< scalar, Foam::edgeMesh >, DimensionedField< Foam::Vector, Foam::areaMesh >, DimensionedField< Foam::Vector, Foam::edgeMesh >, DimensionedField< Foam::SphericalTensor, Foam::areaMesh >, DimensionedField< Foam::SphericalTensor, Foam::edgeMesh >, DimensionedField< Foam::SymmTensor, Foam::areaMesh >, DimensionedField< Foam::SymmTensor, Foam::edgeMesh >, DimensionedField< Foam::Tensor, Foam::areaMesh >, DimensionedField< Foam::Tensor, Foam::edgeMesh >, DimensionedField< Type, Foam::areaMesh >, DimensionedField< Type, Foam::edgeMesh >, DimensionedField< Foam::Vector, Foam::surfaceMesh >, DimensionedField< Foam::SphericalTensor, Foam::volMesh >, DimensionedField< Foam::SphericalTensor, Foam::surfaceMesh >, DimensionedField< Foam::SymmTensor, Foam::volMesh >, DimensionedField< Foam::SymmTensor, Foam::surfaceMesh >, DimensionedField< Foam::Tensor, Foam::volMesh >, DimensionedField< Foam::Tensor, Foam::surfaceMesh >, DimensionedField< Type, Foam::volMesh >, DimensionedField< Type, Foam::surfaceMesh >, DimensionedField< Type, Foam::pointMesh >, DimensionedField< vector, Foam::surfaceMesh >, DimensionedField< vector, surfaceMesh >, DimensionedField< scalar, triSurfacePointGeoMesh >, DimensionedField< Foam::SphericalTensor, Foam::pointMesh >, DimensionedField< Foam::SymmTensor, Foam::pointMesh >, DimensionedField< Foam::Tensor, Foam::pointMesh >, DimensionedField< tensor, Foam::areaMesh >, DimensionedField< tensor, areaMesh >, DimensionedField< vector, Foam::areaMesh >, DimensionedField< vector, areaMesh >, DimensionedField< scalar, edgeMesh >, DynamicField< Foam::Vector >, DynamicField< label >, GlobalIOField< vector >, IOField< label >, IOField< scalar >, IOField< Foam::Vector >, IOField< Foam::SphericalTensor >, IOField< Foam::SymmTensor >, IOField< Foam::Tensor >, IjkField< scalar >, IjkField< Foam::Vector >, IjkField< Foam::Vector< bool > >, IjkField< Foam::SymmTensor >, faPatchField< scalar >, faPatchField< Foam::Vector >, faPatchField< Foam::SphericalTensor >, faPatchField< Foam::SymmTensor >, faPatchField< Foam::Tensor >, faPatchField< tensor >, faPatchField< vector >, faePatchField< scalar >, faePatchField< Foam::Vector >, faePatchField< Foam::SphericalTensor >, faePatchField< Foam::SymmTensor >, faePatchField< Foam::Tensor >, fvPatchField< symmTensor >, fvPatchField< scalar >, fvPatchField< vector >, fvPatchField< tensor >, fvPatchField< Foam::Vector >, fvPatchField< Foam::SphericalTensor >, fvPatchField< Foam::SymmTensor >, fvPatchField< Foam::Tensor >, fvsPatchField< scalar >, fvsPatchField< Foam::Vector >, fvsPatchField< Foam::SphericalTensor >, fvsPatchField< Foam::SymmTensor >, fvsPatchField< Foam::Tensor >, fvsPatchField< vector >, valuePointPatchField< vector >, DimensionedField< Type, GeoMesh >, GlobalIOField< Type >, IOField< Type >, IjkField< Type >, faPatchField< Type >, faePatchField< Type >, fvPatchField< Type >, fvsPatchField< Type >, rawIOField< Type >,** and **valuePointPatchField< Type >**.

Collaboration diagram for Field< Type >:

Field class

Building Blocks in OpenFOAM®

Overview of discretisation

- Spatial discretisation

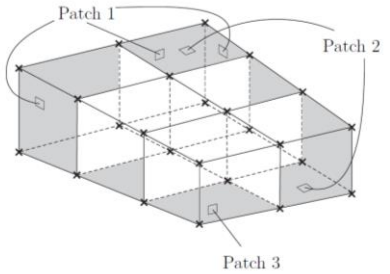
  Defining the solution domain by a set of points that fill and bound a region of space when connected;

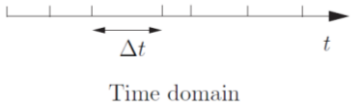
- Temporal discretisation

  (For transient problems) dividing the time domain into into a finite number of time intervals, or steps;

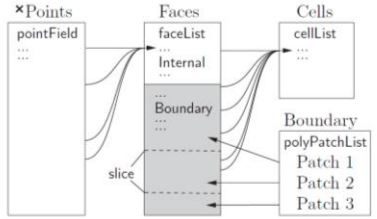
- Equation discretisation

  Generating a system of algebraic equations in terms of discrete quantities defined at specific locations in the domain, from the PDEs that characterise the problem.





Temporal discretisation



Spatial discretisation

Term description	Implicit / Explicit	Text expression	fvm::/fvc:: functions
Laplacian	Imp/Exp	$\nabla^2\phi$ $\nabla\bullet\Gamma\nabla\phi$	`laplacian(phi)` `laplacian(Gamma, phi)`
Time derivative	Imp/Exp	$\frac{\partial\phi}{\partial t}$ $\frac{\partial\rho\phi}{\partial t}$	`ddt(phi)` `ddt(rho,phi)`
Second time derivative	Imp/Exp	$\frac{\partial}{\partial t}\left(\rho\frac{\partial\phi}{\partial t}\right)$	`d2dt2(rho, phi)`
Convection	Imp/Exp	$\nabla\bullet(\psi)$ $\nabla\bullet(\psi\phi)$	`div(psi,scheme)`* `div(psi, phi, word)`* `div(psi, phi)`
Divergence	Exp	$\nabla\bullet\chi$	`div(chi)`

Gradient	Exp	$\nabla\chi$
		$\nabla\phi$
			`grad(chi)`
			`gGrad(phi)`
			`lsGrad(phi)`
			`snGrad(phi)`
			`snGradCorrection(phi)`

Grad-grad squared	Exp	$\lvert\nabla\nabla\phi\rvert^2$	`sqrGradGrad(phi)`
Curl	Exp	$\nabla\times\phi$	`curl(phi)`
Source	Imp Imp/Exp†	$\rho\phi$	`Sp(rho,phi)` `SuSp(rho,phi)`

†`fvm::SuSp` source is discretised implicit or explicit depending on the sign of `rho`.
†An explicit source can be introduced simply as a vol<Type>Field, *e.g.*`rho*phi`.
Function arguments can be of the following classes:
`phi`: vol<Type>Field
`Gamma`: scalar volScalarField, surfaceScalarField, volTensorField, surfaceTensorField.
`rho`: scalar, volScalarField
`psi`: surfaceScalarField.
`chi`: surface<Type>Field, vol<Type>Field.

Equation discretisation

Science and Technology Facilities Council

Scientific Computing

Solver Structure

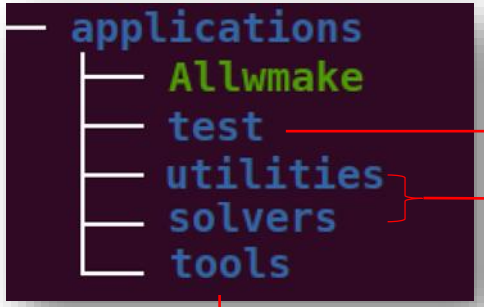
Science and Technology Facilities Council

Scientific Computing

Solver Structure

Folder structure of `applications/`

- Contains executable applications, utilities and unit tests

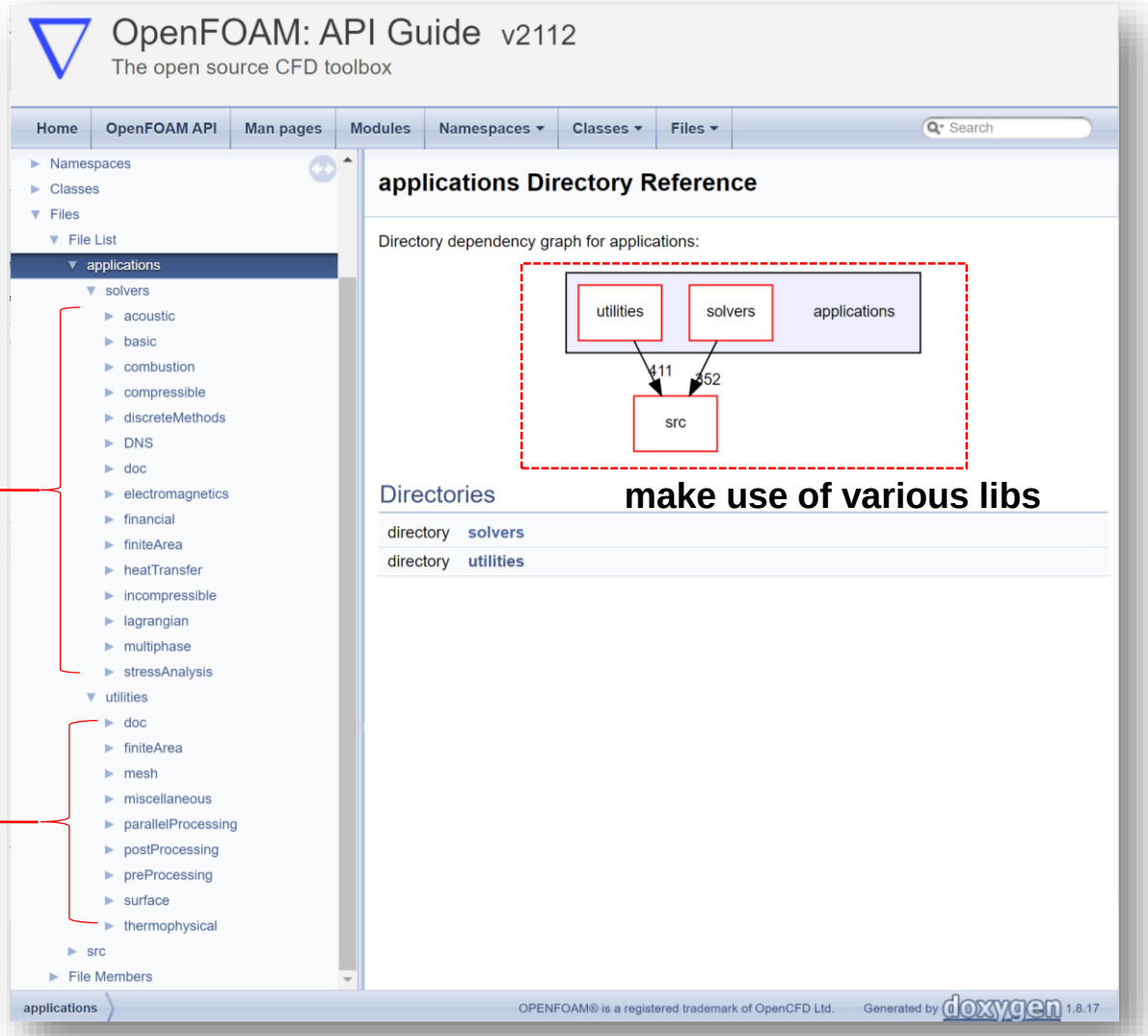


Unit tests for classes & functions. Useful resource to show the usage of classes & algorithms.

Tools that may be useful for developers or advanced users

Categorised in groups according to physical & mathematical models

Categorised in groups according to type of classes

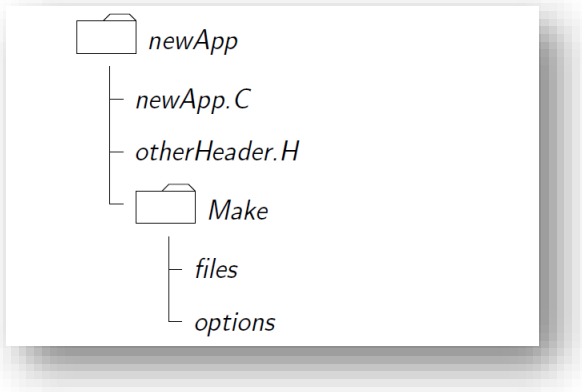


Science and Technology Facilities Council

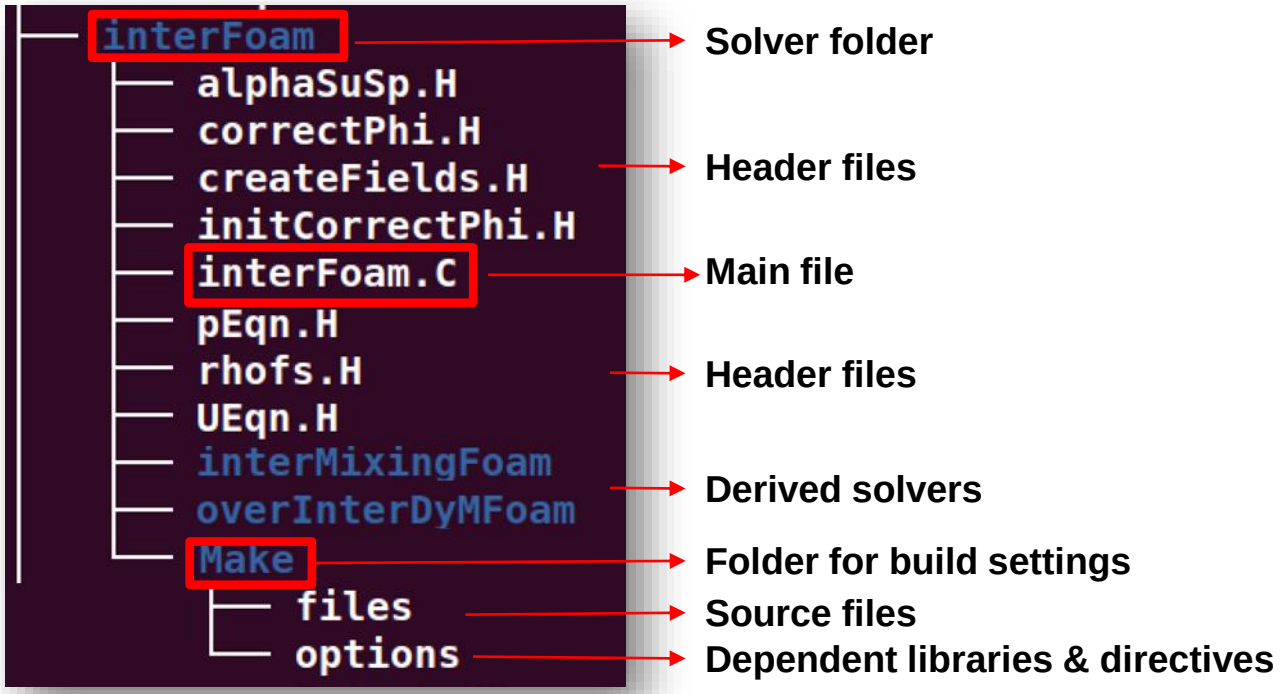
Scientific Computing

Solver Structure

Folder structure of `applications/solvers/multiphase/interFoam`



Directory structure for an application



Science and Technology Facilities Council

Scientific Computing

Solver Structure

High-level overview of `interFoam`

Hydrostatic pressure effects

For cases that the hydrostatic pressure contribution

$$\rho(\mathbf{g} \cdot \mathbf{h})$$

is important, e.g. for buoyant and multiphase cases, it is numerically convenient to solve for an alternative pressure defined by

$$p' = p - \rho(\mathbf{g} \cdot \mathbf{h}).$$

In OpenFOAM solver applications the p' pressure term is named `p_rgh`. The momentum equation

$$\frac{\partial}{\partial t}(\rho \mathbf{u}) + \nabla \cdot (\rho \mathbf{u} \otimes \mathbf{u}) - \nabla \cdot (\mu_{\text{eff}} \nabla \mathbf{u}) = -\nabla p + \rho \mathbf{g}$$

is transformed to use p':

$$p' = p - \rho(\mathbf{g} \cdot \mathbf{h}).$$

After the following substitutions:

$$\begin{aligned} -p &= -p' - \rho(\mathbf{g} \cdot \mathbf{h}) \\ -\nabla p &= -\nabla(p') - \nabla(\rho(\mathbf{g} \cdot \mathbf{h})) \\ &= -\nabla(p') - \rho \mathbf{g} \cdot \nabla \mathbf{h} - \mathbf{h} \cdot \nabla(\rho \mathbf{g}) \\ &= -\nabla(p') - \rho \mathbf{g} \cdot \mathbf{I} - \mathbf{g} \cdot \mathbf{h} \nabla(\rho) - \cancelto{0}{\rho \mathbf{h} \cdot \nabla(\mathbf{g})} \\ &= -\nabla(p') - \rho \mathbf{g} - \mathbf{g} \cdot \mathbf{h} \nabla \rho \end{aligned}$$

where, for CFD meshes the term $\nabla \mathbf{h}$ is given by the gradient of the cell centres, which equates to the tensor $\mathbf{I}$, the momentum equation becomes:

$$\frac{\partial}{\partial t}(\rho \mathbf{u}) + \nabla \cdot (\rho \mathbf{u} \otimes \mathbf{u}) - \nabla \cdot (\mu_{\text{eff}} \nabla \mathbf{u}) = -\nabla p' - \mathbf{g} \cdot \mathbf{h} \nabla \rho$$

For constant density applications this can be further simplified to

$$\frac{\partial}{\partial t}(\rho \mathbf{u}) + \nabla \cdot (\rho \mathbf{u} \otimes \mathbf{u}) - \nabla \cdot (\mu_{\text{eff}} \nabla \mathbf{u}) = -\nabla p'$$

interFoam

Overview

- Category: Multiphase
- transient
- incompressible
- multiphase: 2 immiscible phases
- isothermal
- Turbulence
- Finite volume options

Equations

- PIMPLE algorithm
- Hydrostatic pressure effects

Input requirements

Mandatory fields:

- p: pressure [Pa]
- p_rgh: pressure - hydrostatic contribution [Pa]
- U: velocity [m/s]

Physical models

- turbulence: `constant/turbulenceProperties`
- finite volume options : `constant/fvOptions` (optional)

Solution controls

- Schemes
- Linear equation solvers
- controlDict

Solver Structure

A walk-through of `interFoam`

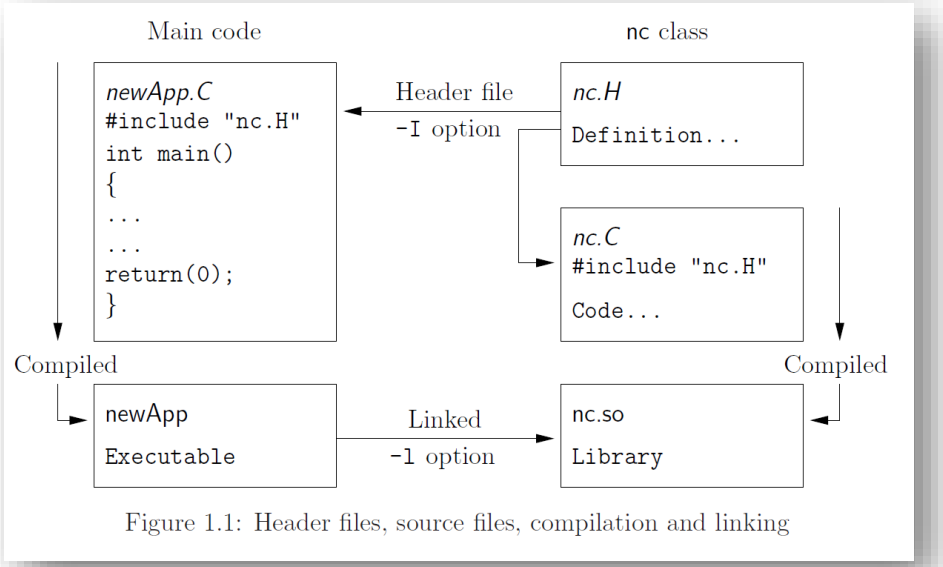
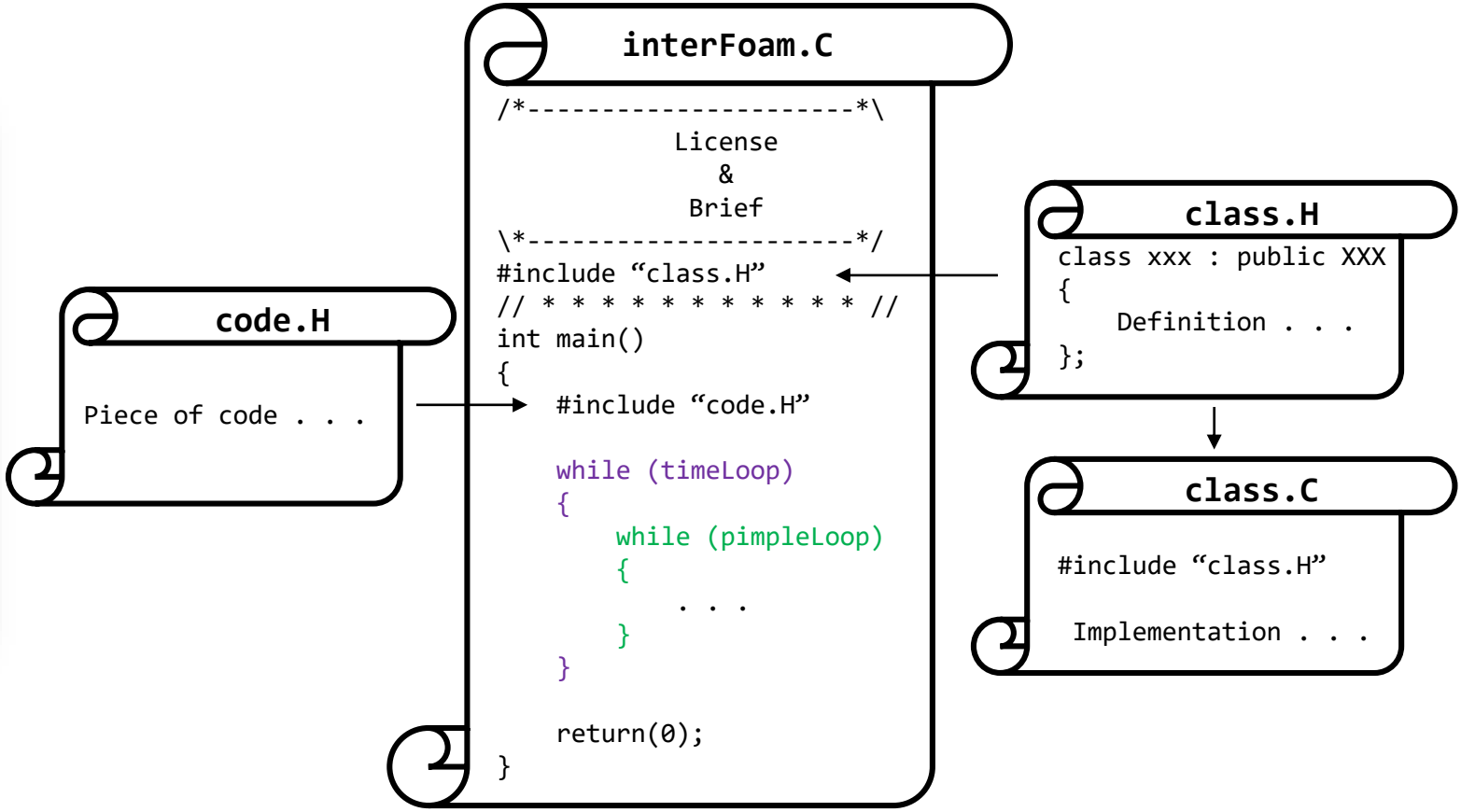


Figure 1.1: Header files, source files, compilation and linking

Header files, source files, compilation and linking



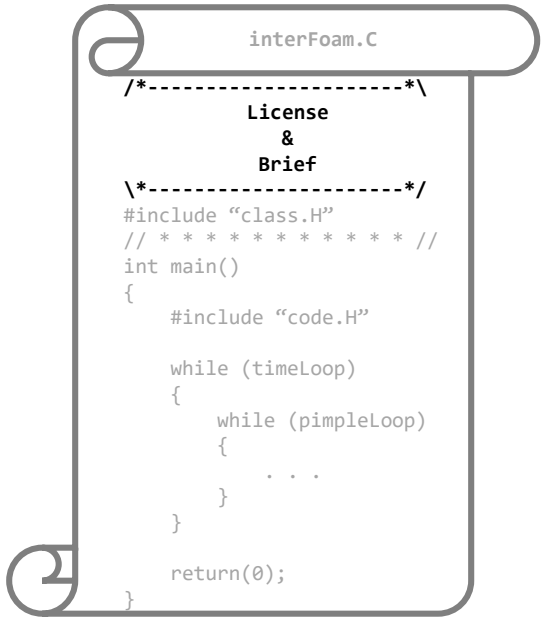
High-level structure of `interFoam`

Science and Technology Facilities Council

Scientific Computing

Solver Structure

A walk-through of `interFoam`



Science and Technology Facilities Council

Scientific Computing

OpenFOAM: API Guide v2112

The open source CFD toolbox

Home | OpenFOAM API | Man pages | Modules | Namespaces | Classes | Files

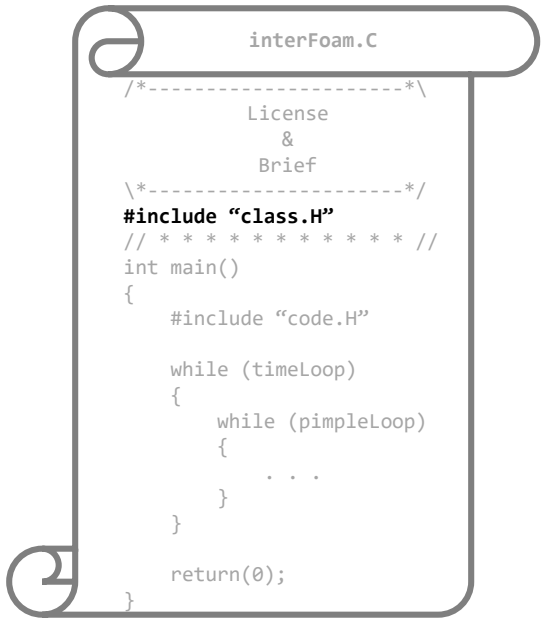
interFoam.C

Go to the documentation of this file.

```
/*---------------------------------------------------------------------------*\
  =========                 |
  \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox
   \\    /   O peration     |
    \\  /    A nd           | www.openfoam.com
     \\/     M anipulation  |
-------------------------------------------------------------------------------
    Copyright (C) 2011-2017 OpenFOAM Foundation
    Copyright (C) 2020 OpenCFD Ltd.
-------------------------------------------------------------------------------
License
    This file is part of OpenFOAM.

    OpenFOAM is free software: you can redistribute it and/or modify it
    under the terms of the GNU General Public License as published by
    the Free Software Foundation, either version 3 of the License, or
    (at your option) any later version.

    OpenFOAM is distributed in the hope that it will be useful, but WITHOUT
    ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or
    FITNESS FOR A PARTICULAR PURPOSE.  See the GNU General Public License
    for more details.

    You should have received a copy of the GNU General Public License
    along with OpenFOAM.  If not, see <http://www.gnu.org/licenses/>.

Application
    interFoam

Group
    grpMultiphaseSolvers

Description
    Solver for two incompressible, isothermal immiscible fluids using a VOF
    (volume of fluid) phase-fraction based interface capturing approach,
    with optional mesh motion and mesh topology changes including adaptive
    re-meshing.

\*---------------------------------------------------------------------------*/

```

Solver Structure

A walk-through of `interFoam`



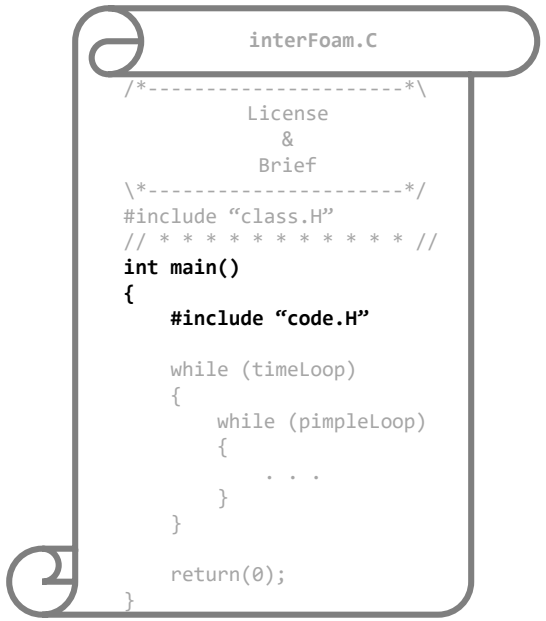

```
33 Description
34     Solver for two incompressible, isothermal immiscible fluids using a VOF
35     (volume of fluid) phase-fraction based interface capturing approach,
36     with optional mesh motion and mesh topology changes including adaptive
37     re-meshing.
38
39 \*---------------------------------------------------------------------------*/
40
41 #include "fvCFD.H"
42 #include "dynamicFvMesh.H"
43 #include "CMULES.H"
44 #include "EulerDdtScheme.H"
45 #include "localEulerDdtScheme.H"
46 #include "CrankNicolsonDdtScheme.H"
47 #include "subCycle.H"
48 #include "immiscibleIncompressibleTwoPhaseMixture.H"
49 #include "incompressibleInterPhaseTransportModel.H"
50 #include "turbulentTransportModel.H"
51 #include "pimpleControl.H"
52 #include "fvOptions.H"
53 #include "CorrectPhi.H"
54 #include "fvcSmooth.H"
55
56 // * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //
57
58 int main(int argc, char *argv[])
59 {
```

Science and Technology Facilities Council

Scientific Computing

Solver Structure

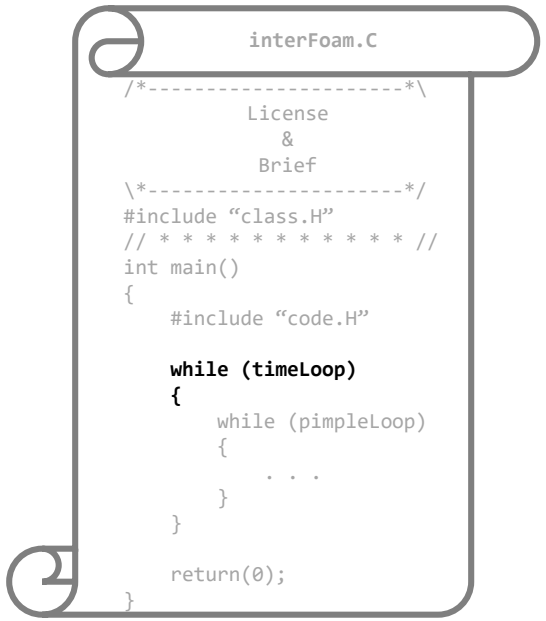
A walk-through of `interFoam`




```
58  int main(int argc, char *argv[])
59  {
60      argList::addNote
61      (
62          "Solver for two incompressible, isothermal immiscible fluids"
63          " using VOF phase-fraction based interface capturing.\n"
64          "With optional mesh motion and mesh topology changes including"
65          " adaptive re-meshing."
66      );
67
68      #include "postProcess.H"
69
70      #include "addCheckCaseOptions.H"
71      #include "setRootCaseLists.H"
72      #include "createTime.H"
73      #include "createDynamicFvMesh.H"
74      #include "initContinuityErrs.H"
75      #include "createDyMControls.H"
76      #include "createFields.H"
77      #include "createAlphaFluxes.H"
78      #include "initCorrectPhi.H"
79      #include "createUfIfPresent.H"
80
81      if (!LTS)
82      {
83          #include "CourantNo.H"
84          #include "setInitialDeltaT.H"
85      }
```

Solver Structure

A walk-through of `interFoam`




```
87      // * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //
88      Info<< "\nStarting time loop\n" << endl;
89
90      while (runTime.run())
91      {
92          #include "readDyMControls.H"
93
94          if (LTS)
95          {
96              #include "setRDeltaT.H"
97          }
98          else
99          {
100             #include "CourantNo.H"
101             #include "alphaCourantNo.H"
102             #include "setDeltaT.H"
103         }
104
105         ++runTime;
106
107         Info<< "Time = " << runTime.timeName() << nl << endl;
```

Science and Technology Facilities Council

Scientific Computing

Solver Structure

A walk-through of `interFoam`

interFoam.C

```
/*---------------------*\
          License
            &
          Brief
\*---------------------*/
#include “class.H”
// * * * * * * * * * * * //
int main()
{
    #include “code.H”

    while (timeLoop)
    {
        while (pimpleLoop)
        {
            . . .
        }
    }

    return(0);
}
```

createFields.H

```
33
34  Info<< "Reading transportProperties\n" << endl;
35  immiscibleIncompressibleTwoPhaseMixture mixture(U, phi);
```

```
109         // --- Pressure-velocity PIMPLE corrector loop
110         while (pimple.loop())
111         {
112             if (pimple.firstIter() || moveMeshOuterCorrectors)
113             {
114                 mesh.update();
115
116                 if (mesh.changing())
117                 {
118                     // Do not apply previous time-step mesh compression flux
119                     // if the mesh topology changed
120                     if (mesh.topoChanging())
121                     {
122                         talphaPhi1Corr0.clear();
123                     }
124
125                     gh = (g & mesh.C()) - ghRef;
126                     ghf = (g & mesh.Cf()) - ghRef;
127
128                     MRF.update();
129
130                     if (correctPhi)
131                     {
132                         // Calculate absolute flux
133                         // from the mapped surface velocity
134                         phi = mesh.Sf() & Uf();
135
136                         #include "correctPhi.H"
137
138                         // Make the flux relative to the mesh motion
139                         fvc::makeRelative(phi, U);
140
141                         mixture.correct();
142                     }
143
144                     if (checkMeshCourantNo)
145                     {
146                         #include "meshCourantNo.H"
147                     }
148                 }
149             }
```

Science and Technology Facilities Council

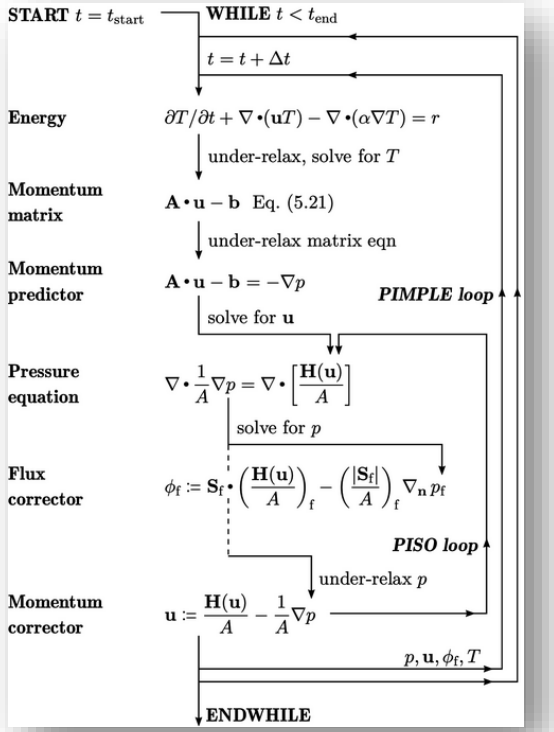
Scientific Computing

Solver Structure

A walk-through of `interFoam`

interFoam.C

```
/*---------------------*\
          License
            &
          Brief
\*---------------------*/
#include “class.H”
// * * * * * * * * * * * //
int main()
{
    #include “code.H”

    while (timeLoop)
    {
        while (pimpleLoop)
        {
            . . .
        }
    }

    return(0);
}
```



Notes on Computational Fluid Dynamics: General Principles

```
151             #include "alphaControls.H"
152             #include "alphaEqnSubCycle.H"
153
154             mixture.correct();
155
156             if (pimple.frozenFlow())
157             {
158                 continue;
159             }
160
161             #include "UEqn.H"
162
163             // --- Pressure corrector loop
164             while (pimple.correct())
165             {
166                 #include "pEqn.H"
167             }
168
169             if (pimple.turbCorr())
170             {
171                 turbulence->correct();
172             }
173         }
174
175         runTime.write();
176
177         runTime.printExecutionTime(Info);
178     }
179
180     Info<< "End\n" << endl;
181
182     return 0;
183 }
184
185
186 // ************************************************************************* //
```

Science and Technology Facilities Council

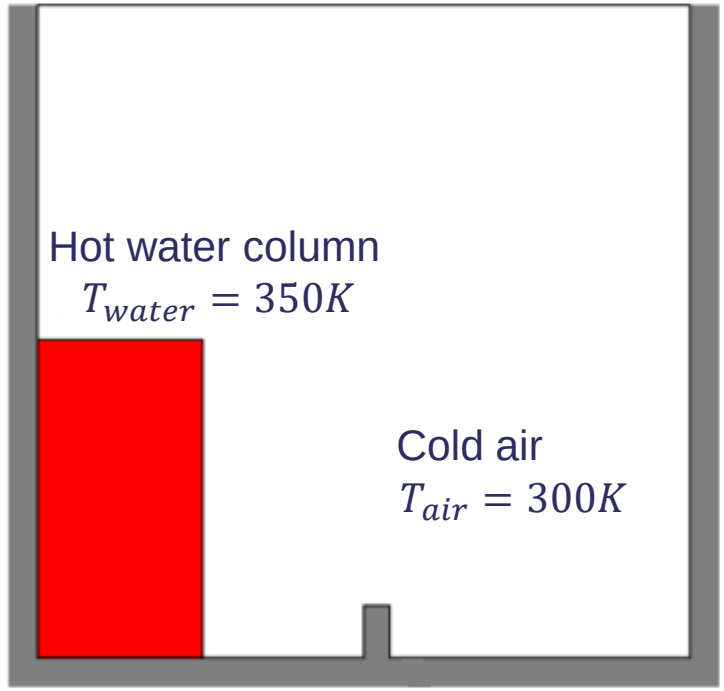
Scientific Computing

Hands On Session

Hands On Session

A 2-D dam break with hot water column

A column of hot water with a temperature of $T_{water} = 350K$ at rest located behind a membrane on the left side of a tank. The rest of the tank is filled with cold air with a temperature of $T_{air} = 300K$. At time $t = 0s$, the membrane is removed and the column of hot water collapses. During the collapse, the hot water impacts an obstacle at the bottom of the tank and exchanges heat continuously with the surrounding cold air.



Geometry of the dam break

Hint: Modify the `interFoam` solver to implement unsteady passive scalar transport equations for heat exchange.

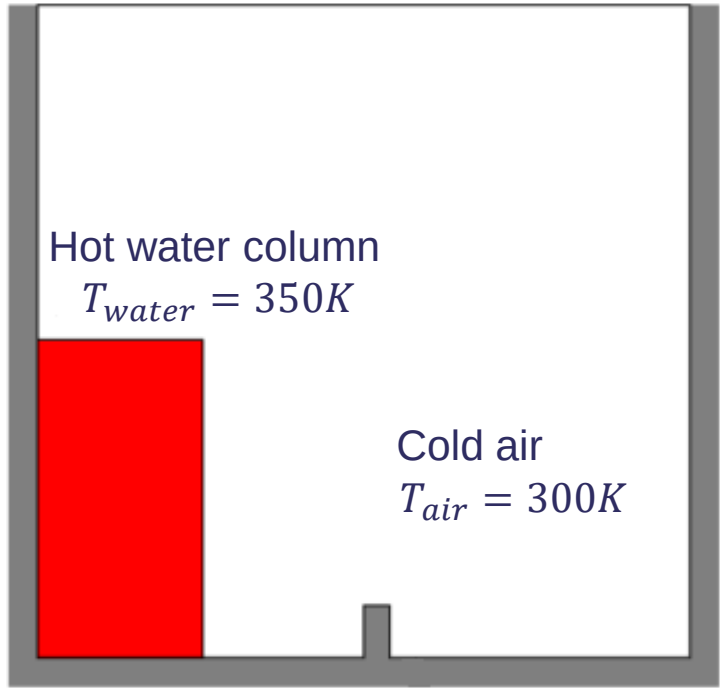
Hands On Session

Step 0: Analysis and planning

$$\frac{\partial \rho T}{\partial t} + \nabla \cdot (\rho \boldsymbol{U} T) - \nabla \cdot \left(D_{eff} \nabla T\right) = 0$$

$$D_{eff} = \frac{\alpha k_1}{C_{v1}} + \frac{(1-\alpha) k_2}{C_{v2}}$$

- T: Temperature
- D_{eff}: Effective thermal conductivity coefficient of the phase mixture
- k: Conduction coefficient
- C_v: Specific heat capacity



Geometry of the dam break

Science and Technology Facilities Council
Scientific Computing

Caution: It is a toy case to demonstrate how to modify a solver. It cannot be used for productive simulations due to omitting a number of important formulations.

Hands On Session

Step 1: Preparation

- Open the “*OpenFOAM Sandbox*” terminal environment from *Applications >> Software*

- Source the compiled copy of OpenFOAM from the course_material directory

```
source /mnt/ceph-training/course_materials/OpenFOAM/OpenFOAM-v2212/etc/bashrc
```

Hands On Session

Step 1: Preparation

- Create OpenFOAM user directory and a subfolder for the modified solver

```
mkdir -p $WM_PROJECT_USER_DIR/applications && cd $WM_PROJECT_USER_DIR/applications
```

```
cp -r /mnt/ceph-training/course_materials/OpenFOAM/OpenFOAM-v2212/applications/solvers/multiphase/interFoam/ interThermoFoam
cp -r /mnt/ceph-training/course_materials/OpenFOAM/OpenFOAM-v2212/applications/solvers/multiphase/VoF/ .
```

```
rm -r interThermoFoam/interMixingFoam interThermoFoam/overInterDyMFoam
```

Hands On Session

Step 1: Preparation

- Create a subfolder in the OpenFOAM® user directory for the test case

```
cd $WM_PROJECT_USER_DIR && mkdir run && cd run

cp -r /mnt/ceph-training/course_materials/OpenFOAM/OpenFOAM-

v2212/tutorials/multiphase/interFoam/laminar/damBreakPermeable/ .
```

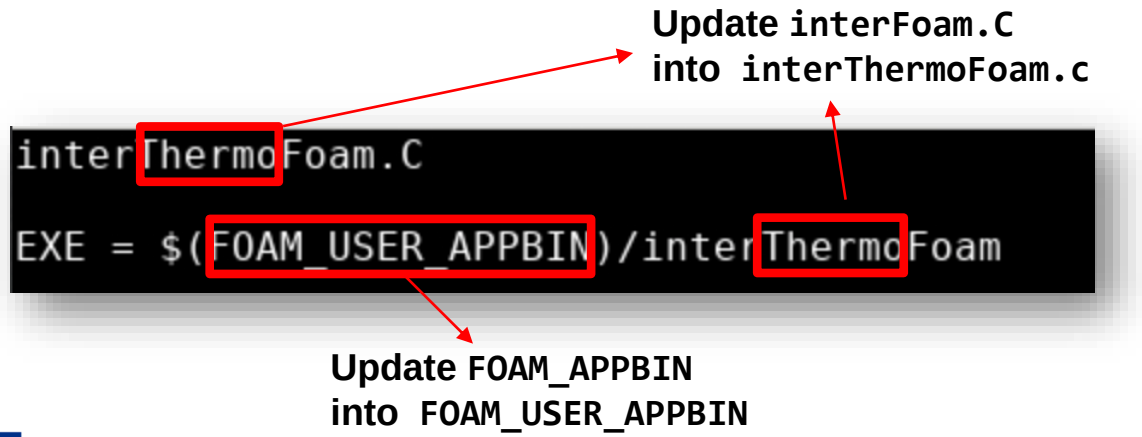
Hands On Session

Step 2: Rename the application

- Rename source file `interFoam.C`

```
cd $WM_PROJECT_USER_DIR/applications/interThermoFoam

mv interFoam.C interThermoFoam.C
```

- Update config file `Make/files`



Hands On Session

Step 2: Rename the application

- Update source file comments

Update `interFoam.C` into `interThermoFoam.c`

```
Application
    interThermoFoam

Group
    grpMultiphaseSolvers

Description
    Solver for two incompressible, isothermal immis
    (volume of fluid) phase-fraction based interfac
    with optional mesh motion and mesh topology cha
    re-meshing, with simplified thermal equations.

\*---------------------------------------------------
```

Add a description of the new functionality

```
/*---------------------------------------------------------------------------*\
  =========                 |
  \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox
   \\    /   O peration     |
    \\  /    A nd           | www.openfoam.com
     \\/     M anipulation  |
-------------------------------------------------------------------------------
    Copyright (C) 2011-2017 OpenFOAM Foundation
    Copyright (C) 2020 OpenCFD Ltd.
-------------------------------------------------------------------------------
License
    This file is part of OpenFOAM.

    OpenFOAM is free software: you can redistribute it and/or modify it
    under the terms of the GNU General Public License as published by
    the Free Software Foundation, either version 3 of the License, or
    (at your option) any later version.

    OpenFOAM is distributed in the hope that it will be useful, but WITHOUT
    ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or
    FITNESS FOR A PARTICULAR PURPOSE.  See the GNU General Public License
    for more details.

    You should have received a copy of the GNU General Public License
    along with OpenFOAM.  If not, see <http://www.gnu.org/licenses/>.

Application
    interThermoFoam

Group
    grpMultiphaseSolvers

Description
    Solver for two incompressible, isothermal immiscible fluids using a VOF
    (volume of fluid) phase-fraction based interface capturing approach,
    with optional mesh motion and mesh topology changes including adaptive
    re-meshing, with simplified thermal equations.

\*---------------------------------------------------------------------------*/

#include "fvCFD.H"
#include "dynamicFvMesh.H"
#include "CMULES.H"
-- INSERT --
```

interThermoFoam.C

Hands On Session

Step 3: Adding new Entries into source codes

- Add definitions of k and C_v in `createFields.H`

Add k and C_v after the definition of `mixture`

A instance named k_1

Name of the dimensioned scalar is k

The dimension of k: $[W/(m \cdot K)] = [M^1 L^1 t^{-3} T^{-1} Q^0 C^0 l^0]$

Lookup `transportProperties` under the sub-dictionary `phase1Name()`

Lookup keyword k

A instance named C_{v1}

Name of the dimensioned scalar is C_v

The dimension of C_v: $[J/(kg \cdot K)] = [M^0 L^2 t^{-2} T^{-1} Q^0 C^0 l^0]$

Lookup `transportProperties` under the sub-dictionary `phase1Name()`

Lookup keyword C_v

```
immiscibleIncompressibleTwoPhaseMixture mixture(U, phi);

volScalarField& alpha1(mixture.alpha1());
volScalarField& alpha2(mixture.alpha2());

const dimensionedScalar& rho1 = mixture.rho1();
const dimensionedScalar& rho2 = mixture.rho2();

dimensionedScalar k1
(
    "k",
    dimensionSet(1, 1, -3, -1, 0, 0, 0),
    mixture.subDict(mixture.phase1Name()),
    "k"
);

dimensionedScalar k2
(
    "k",
    dimensionSet(1, 1, -3, -1, 0, 0, 0),
    mixture.subDict(mixture.phase2Name()),
    "k"
);

dimensionedScalar Cv1
(
    "Cv",
    dimensionSet(0, 2, -2, -1, 0, 0, 0),
    mixture.subDict(mixture.phase1Name()),
    "Cv"
);

dimensionedScalar Cv2
(
    "Cv",
    dimensionSet(0, 2, -2, -1, 0, 0, 0),
    mixture.subDict(mixture.phase2Name()),
    "Cv"
);
```

createFields.H

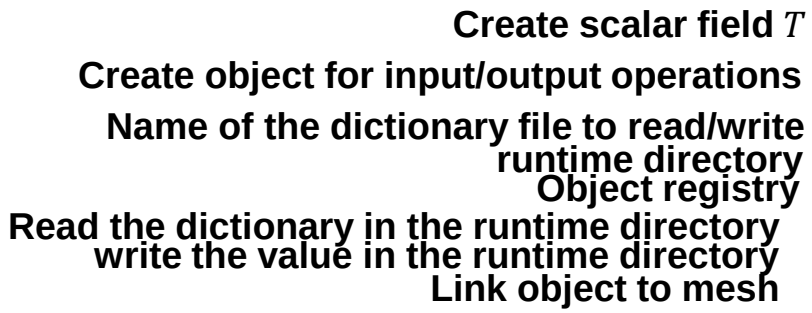
Science and Technology Facilities Council

Scientific Computing

Hands On Session

Step 3: Adding new Entries into source codes

- Initialise the temperature field in `createFields.H`




```
(
    "Cv",
    dimensionSet(0, 2, -2, -1, 0, 0, 0),
    mixture.subDict(mixture.phase1Name()),
    "Cv"
);

dimensionedScalar Cv2
(
    "Cv",
    dimensionSet(0, 2, -2, -1, 0, 0, 0),
    mixture.subDict(mixture.phase2Name()),
    "Cv"
);

volScalarField T
(
    IOobject
    (
        "T",
        runTime.timeName(),
        mesh,
        IOobject::MUST_READ,
        IOobject::AUTO_WRITE
    ),
    mesh
);


// Need to store rho for ddt(rho, U)
volScalarField rho
(
    IOobject
    (
        "rho",
        runTime.timeName(),
        mesh,
        IOobject::READ_IF_PRESENT
    ),
    alpha1*rho1 + alpha2*rho2
);
rho.oldTime();
```

createFields.H

Hands On Session

Step 4: Implement model equations

- Create a new header file `TEqn.H` in the source folder

```
cd $WM_PROJECT_USER_DIR/applications/interThermoFoam

touch TEqn.H
```

- Define D_{eff} in `TEqn.H` and solve the passive temperature PDE

$$D_{eff} = \frac{\alpha k_1}{C_{v1}} + \frac{(1-\alpha)k_2}{C_{v2}}$$

$$\frac{\partial \rho T}{\partial t} + \nabla \cdot (\rho \boldsymbol{U} T) - \nabla \cdot \left(D_{eff} \nabla T\right) = 0$$

```
volScalarField Deff
(
    "Deff"
    (alpha1*k1/Cv1 + (scalar(1)-alpha1)*k2/Cv2)
);

solve
(
    fvm::ddt(rho, T)
   +fvm::div(rhoPhi, T)
   -fvm::laplacian(Deff, T)
);
```

TEqn.H

Hands On Session

Step 5: Include `TEqn.H` into solution algorithm

```
mixture.correct();

if (pimple.frozenFlow())
{
    continue;
}

#include "UEqn.H"
#include "TEqn.H"

// --- Pressure corrector loop
while (pimple.correct())
{
    #include "pEqn.H"
}

if (pimple.turbCorr())
{
```

Include `TEqn.H`

interThermoFoam.C

Hands On Session

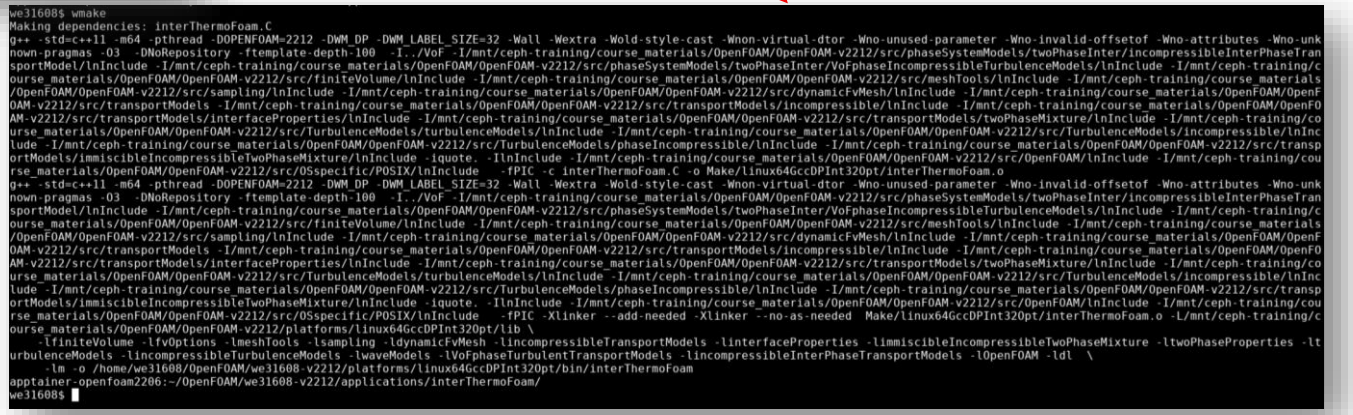
Step 6: Compile `interThermoFoam`

```
cd $WM_PROJECT_USER_DIR/applications/interThermoFoam

wmake
```

Failed?

or

Succeeded?



Hint: Check spelling, and semicolons if failed

Science and Technology Facilities Council

Scientific Computing

Hands On Session

Step 7: Prepare the test case

- Setup temperature field T in `0.orig` folder

```
cd $WM_PROJECT_USER_DIR/run/damBreakPermeable

cp 0.orig/alpha.water 0.orig/T
```

Change object to T

Change dimension of T

Change internal Field value as 300

Change top boundary value as 300

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  v2212                                 |
|   \\  /    A nd           | Website:  www.openfoam.com                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       volScalarField;
    object      T;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

dimensions      [0 0 0 1 0 0 0];

internalField   uniform 300;

boundaryField
{
    leftWall
    {
        type            zeroGradient;
    }

    rightWall
    {
        type            zeroGradient;
    }

    lowerWall
    {
        type            zeroGradient;
    }

    atmosphere
    {
        type            inletOutlet;
        inletValue      uniform 300;
        value           uniform 300;
    }
```

Science and Technology Facilities Council

Scientific Computing

Hands On Session

Step 7: Prepare the test case

- Setup initial condition of water temperature T in `setFieldsDict`

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  v2212                                 |
|   \\  /    A nd           | Website:  www.openfoam.com                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      setFieldsDict;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

defaultFieldValues
(
    volScalarFieldValue alpha.water 0
);

regions
(
    boxToCell
    {
        box (0 0 -1) (0.2461 0.292 1);
        fieldValues
        (
            volScalarFieldValue alpha.water 1
            volScalarFieldValue T 350
        );
    }
);


// ************************************************************************* //
```

Set water temperature as 350

setFieldsDict

Hands On Session

Step 7: Prepare the test case

- Setup fluid thermal properties in `transportProperties`

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  v2212                                 |
|   \\  /    A nd           | Website:  www.openfoam.com                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      transportProperties;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

phases          (water air);

water
{
    transportModel  Newtonian;
    nu              1e-06;
    rho             1000;
    k               0.66;
    Cv              3900;
}

air
{
    transportModel  Newtonian;
    nu              1.48e-05;
    rho             1;
    k               0.03;
    Cv              718;
}

sigma           0.07;


// ************************************************************************* //
```

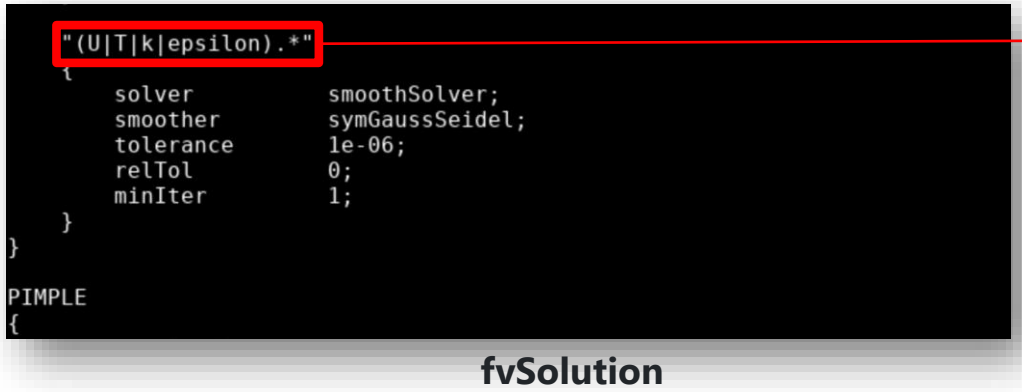
transportProperties

Setup k and C_v for both phases

Hands On Session

Step 7: Prepare the test case

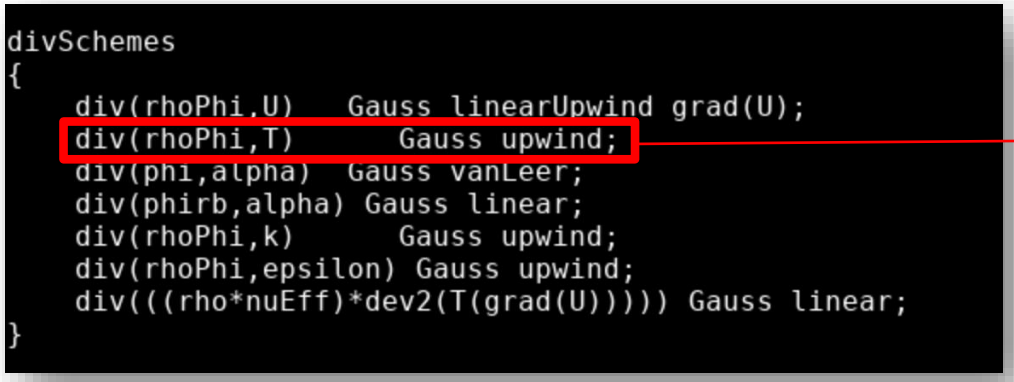
- Setup linear solver for T in `fvSolution`



Add T in smooth linear solver

fvSolution

- Setup divSchemes for $div(rhoPhi, T)$ in `fvSchemes`



Setup $div(rhoPhi, T)$ by Gauss upwind

fvSchemes

Hands On Session

Step 7: Prepare the test case

- Change the application name and set the endTime as 10s in `controlDict`

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  v2212                                 |
|   \\  /    A nd           | Website:  www.openfoam.com                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      controlDict;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

application     interThermoFoam;

startFrom       startTime;

startTime       0;

stopAt          endTime;

endTime         10;
```

Change the application name as `interThermoFoam`

Extend the endTime to 10s

controlDict

Hands On Session

Step 8: Execute the test case

```
cd $WM_PROJECT_USER_DIR/run/damBreakPermeable

./Allrun
```

```
apptainer-openfoam2206:~/OpenFOAM/we31608-v2212/run/damBreakPermeable/
we31608$ ./Allrun
Restore 0/ from 0.orig/
Running blockMesh on /home/we31608/OpenFOAM/we31608-v2212/run/damBreakPermeable
Running setFields on /home/we31608/OpenFOAM/we31608-v2212/run/damBreakPermeable
Running decomposePar on /home/we31608/OpenFOAM/we31608-v2212/run/damBreakPermeable
Running interThermoFoam (4 processes) on /home/we31608/OpenFOAM/we31608-v2212/run/damBreakPermeable
```

Running ...

```
apptainer-openfoam2206:~/OpenFOAM/we31608-v2212/run/damBreakPermeable/
we31608$ ./Allrun
Restore 0/ from 0.orig/
Running blockMesh on /home/we31608/OpenFOAM/we31608-v2212/run/damBreakPermeable
Running setFields on /home/we31608/OpenFOAM/we31608-v2212/run/damBreakPermeable
Running decomposePar on /home/we31608/OpenFOAM/we31608-v2212/run/damBreakPermeable
Running interThermoFoam (4 processes) on /home/we31608/OpenFOAM/we31608-v2212/run/damBreakPermeable
Running redistributePar (4 processes) on /home/we31608/OpenFOAM/we31608-v2212/run/damBreakPermeable
apptainer-openfoam2206:~/OpenFOAM/we31608-v2212/run/damBreakPermeable/
we31608$
```

Done

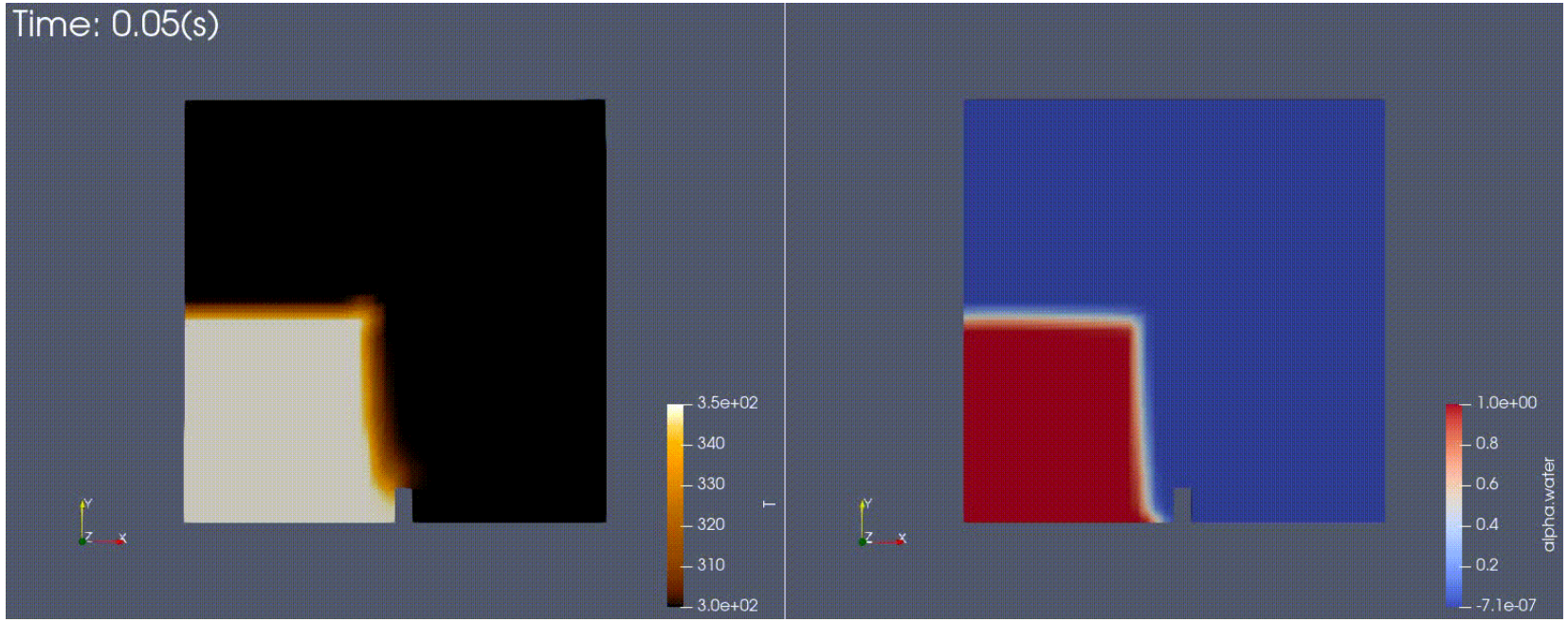
Hands On Session

Step 9: Check the result

```
cd $WM_PROJECT_USER_DIR/run/damBreakPermeable

touch open.foam
```

- Open the "*Paraview*" terminal from

  Applications >> Utilities



Temperature Contour **Phase Contour**

Science and Technology Facilities Council
Scientific Computing

Caution: It is a toy case to demonstrate how to modify a solver. It cannot be used for productive simulations due to omitting a number of important formulations.

UKRI Science and Technology Facilities Council

Scientific Computing

Questions?

References

OpenFOAM: Programmer’s Guide v2212
https://sourceforge.net/projects/openfoam/files/v2212/ProgrammersGuide.pdf/download

OpenFOAM: Extended Code Guide v2112
https://www.openfoam.com/documentation/guides/v2112/doc/

Maric, T., Höpken, J. and Mooney, K.G., *The OpenFOAM® Technology Primer*. Sourceflux, 2014.

UKRI Science and Technology Facilities Council

Scientific Computing

Thank you

scd.stfc.ac.uk @SciComp_STFC STFC Scientific Computing wendi.liu@stfc.ac.uk