

See discussions, stats, and author profiles for this publication at: <https://www.researchgate.net/publication/311901454>

OpenFOAM 'advanced' tutorial

Technical Report · December 2016

DOI: 10.13140/RG.2.2.18360.34560

CITATIONS

0

READS

1,461

1 author:



Victor Pozzobon

Ecole Centrale Paris

11 PUBLICATIONS 22 CITATIONS

SEE PROFILE

Some of the authors of this publication are also working on these related projects:



Biomass gasification under high solar heat flux [View project](#)



OpenFOAM Tutoring [View project](#)

All content following this page was uploaded by [Victor Pozzobon](#) on 26 December 2016.

The user has requested enhancement of the downloaded file.

OpenFOAM tutorial

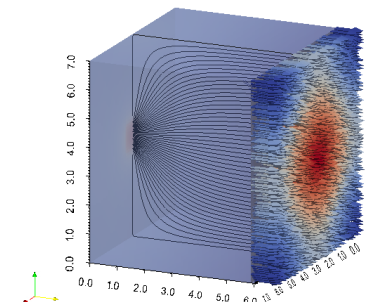
OpenFOAM tutorial

Discover it, tame it, use it

Advanced tutorial



by Victor Pozzobon
(victor.pozzobon@centralesupelec.fr)



26th December 2016

Version 1.00

Disclaimer

“This offering is not approved or endorsed by OpenCFD Limited, the producer of the OpenFOAM software and owner of the OPENFOAM® and OpenCFD® trade marks.”

Introduction

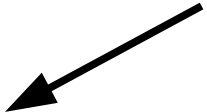
- This tutorial is a follow up of “*OpenFOAM tutorial Discover it, tame it, use it*”
- This document is a step by step guide
- It was done to be used on its own, there should be no need for a presenter (myself)
- It was designed for OpenFOAM 16.06+ (changes may appear in superior versions)

New aims

- This tutorial deals with
 - advanced meshing (simple modifications of meshes generated with OpenFOAM tools)
 - multiphase flow (Volume Of Fluid)
 - turbulence (only simple Reynolds Averaged Simulation)
- More is to come ...

How to use this tutorial

- Almost every command will be passed through the terminal. For example, when you see:
`gedit system/controlDict`
you type it in the terminal
- As OpenFOAM has no GUI, we will modify files. For example, when you see this kind of picture:
modify the file so that
its content is the same
before you save it



```
18 application    simpleFoam;  
19  
20 startFrom      startTime;  
21  
22 startTime      0;  
23  
24 stopAt         endTime;  
25  
26 endTime        100;  
27  
28 deltaT         1;
```

Battle plan

- Ex. 1: Bubble reactor
 - adding patches
- Ex. 2: Circulating reactor
 - removing cell
- Ex. 3: Rising bubble
 - auto refining mesh
- Ex. 4: Ozone tower
 - multiphase transport
- Ex. 5: Turbulent pipe
 - turbulence
- Ex. 6: Turbulent mixing length
 - more turbulence

Battle plan

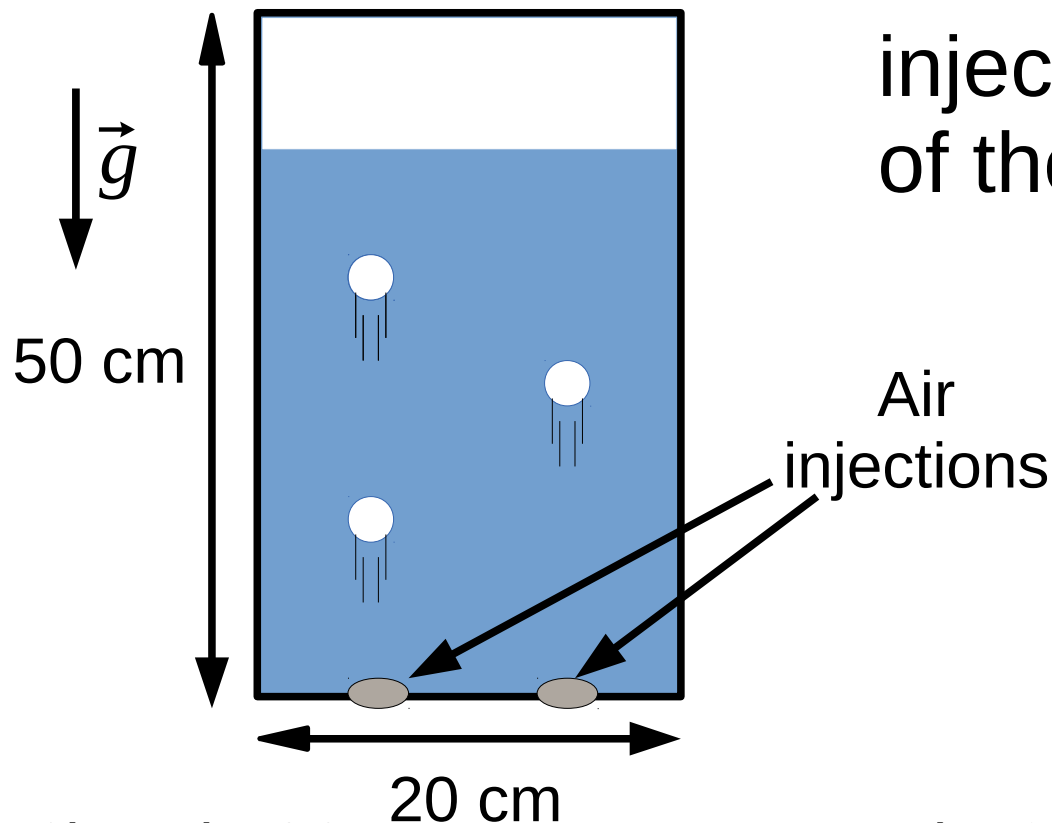
- From Ex. 1 to Ex. 3, we are going to learn how to modify an existing mesh to add new patches, add obstacles, and auto-refine the mesh
- In Ex. 1 and 2, we are going to set a 2D planar case that we will enrich as the exercises go on

Ex. 1: Bubble reactor - Objectives

- Creating a simple mesh
- Specifying new patches that are not entire faces
- Selecting faces based on their locations
- Discover multiphase flow

Ex. 1: Bubble reactor – Case setup

- Solving multiphase flow, Volume Of Fluid equations (VOF) to describe rising bubbles
- Air-Water system, with 2 air injection zones at the bottom of the reactor

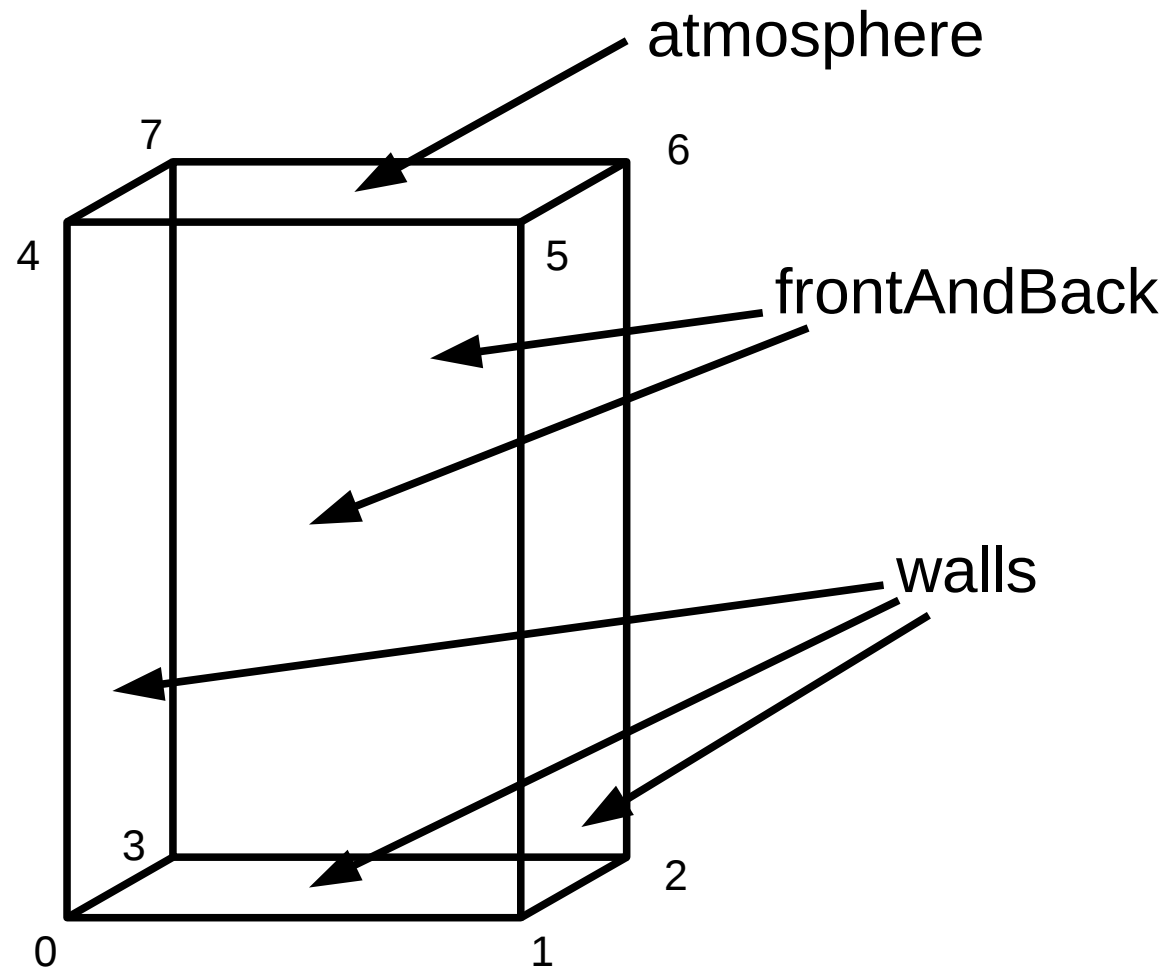


Ex. 1: Bubble reactor – A new case

- Go to your 'run' directory:
`run`
- Copy an existing case:
`cp -r`
`$FOAM_TUTORIALS/multiphase/interFoam/laminar/damBreak/damBreak Ex1`
- Move to the case directory:
`cd Ex1`
- Open the mesh file:
`gedit system/blockMeshDict`

Ex. 1: Bubble reactor – A new mesh

- The reactor is described as a 2D planar geometry



Ex. 1: Bubble reactor – A new mesh

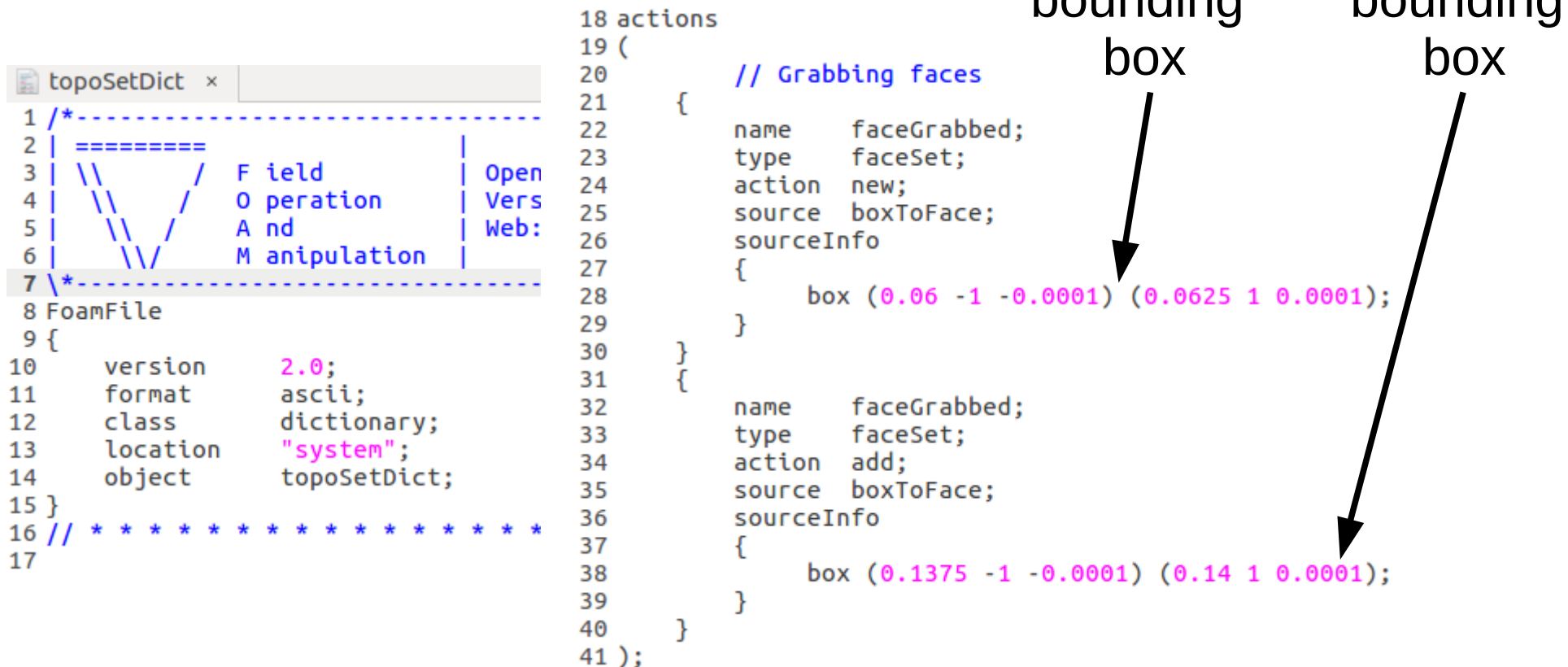
```
blockMeshDict x
1 /*----- C++ -----*/
2 |=====|
3 | \ \ / F i e l d | OpenFOAM: The Open Source CFD
4 | \ \ / O p e r a t i o n | Version: 3.0.0
5 | \ \ / A n d | Web: www.OpenFOAM.org
6 | \ \ / M a n i p u l a t i o n |
7 /*-----*/
8 FoamFile
9 {
10     version      2.0;
11     format        ascii;
12     class         dictionary;
13     object        blockMeshDict;
14 }
15 // *****
16
17 convertToMeters 0.01;
18
19 vertices
20 (
21     (0 0 0)
22     (20 0 0)
23     (20 1 0)
24     (0 1 0)
25     (0 0 50)
26     (20 0 50)
27     (20 1 50)
28     (0 1 50)
29 );
30
31 blocks
32 (
33     hex (0 1 2 3 4 5 6 7) (40 1 100) simpleGrading (1 1 1)
34 );
35
36 edges
37 (
38 );
39
40 boundary
41 (
42     walls
43     {
44         type wall;
45         faces
46         (
47             (0 1 2 3)
48             (1 2 6 5)
49             (0 3 7 4)
50         );
51     }
52     frontAndBack
53     {
54         type empty;
55         faces
56         (
57             (0 1 5 4)
58             (3 2 6 7)
59         );
60     }
61     atmosphere
62     {
63         type patch;
64         faces
65         (
66             (4 5 6 7)
67         );
68     }
69 );
70
71 mergePatchPairs
72 (
73 );
74
75 // *****
```

Ex. 1: Bubble reactor – A new case

- Build the mesh:
blockMesh
- In order to specify the inlet position, we are going to use two tools: topoSet and createPatch
- Copy the associated dictionaries:
cp
\$FOAM_TUTORIALS/multiphase/interFoam/laminar/mixerVessel2D/system/topoSetDict system/.
cp
\$FOAM_TUTORIALS/multiphase/interPhaseChangeDyMFoam/propeller/system/createPatchDict system/.

Ex. 1: Bubble reactor – Selecting faces

- With topoSet, we are going to select the faces that are going to be the air inlet of the reactor:
gedit system/topoSetDict



The image shows a gedit window editing the `topoSetDict` file. The file content is as follows:

```
1 /*-----  
2 |=====|  
3 | \\\ / | F ield | Open  
4 | \\\ / | O peration | Vers  
5 | \\\ / | A nd | Web:  
6 | \\\ / | M anipulation |  
7 |-----  
8 FoamFile  
9 {  
10     version      2.0;  
11     format        ascii;  
12     class          dictionary;  
13     location       "system";  
14     object         topoSetDict;  
15 }  
16 // *****  
17
```

Below the gedit window, the corresponding OpenFOAM code for selecting faces is shown:

```
18 actions  
19 (  
20     // Grabbing faces  
21     {  
22         name      faceGrabbed;  
23         type      faceSet;  
24         action     new;  
25         source     boxToFace;  
26         sourceInfo  
27         {  
28             box (0.06 -1 -0.0001) (0.0625 1 0.0001);  
29         }  
30     }  
31     {  
32         name      faceGrabbed;  
33         type      faceSet;  
34         action     add;  
35         source     boxToFace;  
36         sourceInfo  
37         {  
38             box (0.1375 -1 -0.0001) (0.14 1 0.0001);  
39         }  
40     }  
41 );
```

To the right of the code, a diagram of a bubble reactor is shown. It consists of a central vertical tube with two side inlets. The left inlet is labeled "Inlet 1 bounding box" and the right inlet is labeled "Inlet 2 bounding box". Arrows point from these labels to the corresponding `box` definitions in the code. The first `box` definition corresponds to Inlet 1, and the second `box` definition corresponds to Inlet 2.

Ex. 1: Bubble reactor – Selecting faces

- Then, select the face:
topoSet
(2 faces, should be selected)
- With createPatch, we are going to create the inlets patch:
gedit system/createPatchDict

Ex. 1: Bubble reactor – Creating patches

- Then, create the patch:
createPatch -overwrite

```
createPatchDict x topoSetDict x alpha.water.org x p_rgh x
1  /*-----* C++ *-----*/
2  |=====|
3  | \ \ / F i e l d | OpenFOAM: The Open Source CFD
4  | \ \ / O peration | Version: 2.1.0
5  | \ \ / A nd | Web: www.OpenFOAM.org
6  | \ \ / M anipulation |
7  |*-----*|
8  FoamFile
9  {
10     version      2.0;
11     format        ascii;
12     class         dictionary;
13     object        createPatchDict;
14 }
15
16 // *****
17
18 pointSync false;
19
20 // Patches to create.
21 patches
22 (
23     {
24         // Name of new patch
25         name inlets;
26
27         // Type of new patch
28         patchInfo
29         {
30             type patch;
31         }
32
33         // How to construct: either from 'patches' or 'set'
34         constructFrom set;
35
36         // If constructFrom = set : name of faceSet
37         set faceGrabbed;
38     }
39 );
```

Ex. 1: Bubble reactor – IC / BC

- Set initial/boundary conditions:
gedit 0/U 0/p_rgh 0/alpha.water.org

```

1  /*-----* C++ *-----*/
2  |=====|
3  | \ \ / F i e l d | OpenFOAM: The Open Sou
4  | \ \ / O p e r a t i o n | Version: 3.0.0
5  | \ \ / A n d | Web: www.OpenFOAM
6  | \ \ / M a n i p u l a t i o n |
7  /*-----*
8  FoamFile
9  {
10     version      2.0;
11     format        ascii;
12     class         volVectorField;
13     location      "0";
14     object        U;
15 }
16 // *****
17
18 dimensions      [0 1 -1 0 0 0 0];
19
20 internalField    uniform (0 0 0);
21
22 boundaryField
23 {
24     walls
25     {
26         type      fixedValue;
27         value      uniform (0 0 0);
28     }
29     frontAndBack
30     {
31         type      empty;
32     }
33     atmosphere
34     {
35         type      pressureInletOutletVelocity;
36         value      uniform (0 0 0);
37     }
38     inlets
39     {
40         type      fixedValue;
41         value      uniform (0 0 0.20);
42     }
43 }

```

Ex. 1: Bubble reactor – IC / BC

```
U x p_rgh x alpha.water.org x topoSetI
1 /*-----*-- C++ --*
2 | =====
3 | \ \ / F i e l d | OpenFOAM: The
4 | \ \ / O p e r a t i o n | Version: 3.0.
5 | \ \ / A n d | Web: www.
6 | \ \ / M a n i p u l a t i o n |
7 |*-----*
8 FoamFile
9 {
10     version      2.0;
11     format        ascii;
12     class         volScalarField;
13     object        p_rgh;
14 }
15 // *****
16
17 dimensions      [1 -1 -2 0 0 0 0];
18
19 internalField    uniform 0;
```

```
20
21 boundaryField
22 {
23     walls
24     {
25         type      fixedFluxPressure;
26         value      uniform 0;
27     }
28     frontAndBack
29     {
30         type      empty;
31     }
32     atmosphere
33     {
34         type      totalPressure;
35         p0         uniform 0;
36         U          U;
37         phi        phi;
38         rho        rho;
39         psi        none;
40         gamma      1;
41         value      uniform 0;
42     }
43     inlets
44     {
45         type      fixedFluxPressure;
46         value      $internalField;
47     }
48 }
```

Ex. 1: Bubble reactor – IC / BC

- After modifying the file, copy the water original field:

`cp 0/alpha.water.org 0/alpha.water`

```
U x p_rgh x alpha.water.org x t
1 /*-----*
2 | =====|
3 | \\      / F ield      | OpenFOAM
4 | \\      / O peration  | Version:
5 | \\      / A nd        | Web:
6 | \\      / M anipulation|
7 |*-----*
8 FoamFile
9 {
10     version      2.0;
11     format        ascii;
12     class         volScalarField;
13     object        alpha.water;
14 }
15 // *****
16
17 dimensions      [0 0 0 0 0 0 0];
18
19 internalField    uniform 0;
```

```
20
21 boundaryField
22 {
23     walls
24     {
25         type      zeroGradient;
26     }
27     frontAndBack
28     {
29         type      empty;
30     }
31     atmosphere
32     {
33         type      inletOutlet;
34         inletValue uniform 0;
35         value      uniform 0;
36     }
37     inlets
38     {
39         type      inletOutlet;
40         inletValue uniform 0;
41         value      uniform 0;
42     }
43 }
```

Ex. 1: Bubble reactor – Set water level

- Modify setFieldsDict to specify the initial water level in the reactor:
gedit system/setFieldsDict
- Then, apply it:
setFields

```
setFieldsDict x alpha.water.org x p_rgh x
1 /*----- C++ -----*/
2 |=====|
3 | \ \ / \ | F i e l d | OpenFOAM: The Op
4 | \ \ / \ | O p e r a t i o n | Version: 3.0.0
5 | \ \ / \ | A n d | Web: www.Op
6 | \ \ / \ | M a n i p u l a t i o n |
7 /*-----*/
8 FoamFile
9 {
10     version      2.0;
11     format        ascii;
12     class         dictionary;
13     location      "system";
14     object        setFieldsDict;
15 }
16 // *****
17
18 defaultFieldValues
19 (
20     volScalarFieldValue alpha.water 0
21 );
22
23 regions
24 (
25     boxToCell
26     {
27         box (0 -1 0) (0.20 1 0.40);
28         fieldValues
29         (
30             volScalarFieldValue alpha.water 1
31         );
32     }
33 );
34
35
36 // *****
```

Ex. 1: Bubble reactor – Gravity

- Modify the gravity vector so that it points downward on the vertical direction:
gedit constant/g

```
g x
1 /*-----*- C++ -*-
2 | =====
3 | \\      /  F i e l d      | OpenFOAM: The Ope
4 | \\      /  O peration    | Version:  3.0.0
5 | \\      /  A nd          | Web:      www.Ope
6 | \\      /  M anipulation  |
7 /*-----*-
8 FoamFile
9 {
10     version      2.0;
11     format        ascii;
12     class         uniformDimensionedVectorField;
13     location      "constant";
14     object        g;
15 }
16 // *****
17
18 dimensions      [0 1 -2 0 0 0 0];
19 value           (0 0 -9.81);
20
21
22 // *****
```

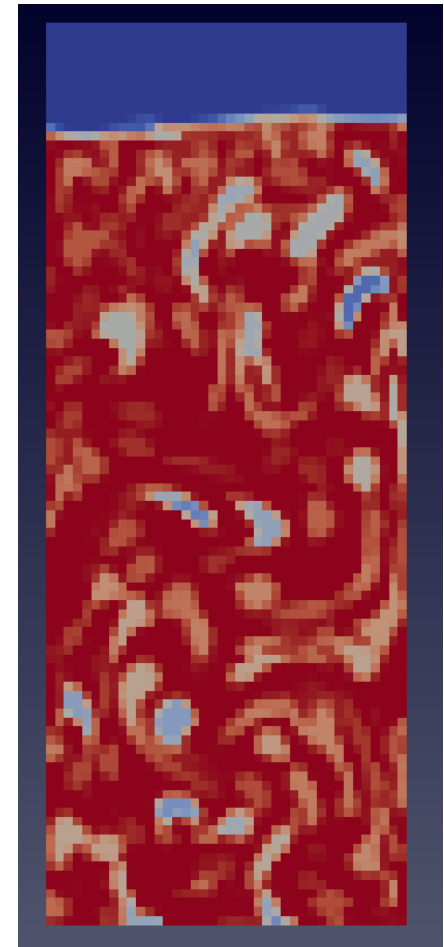
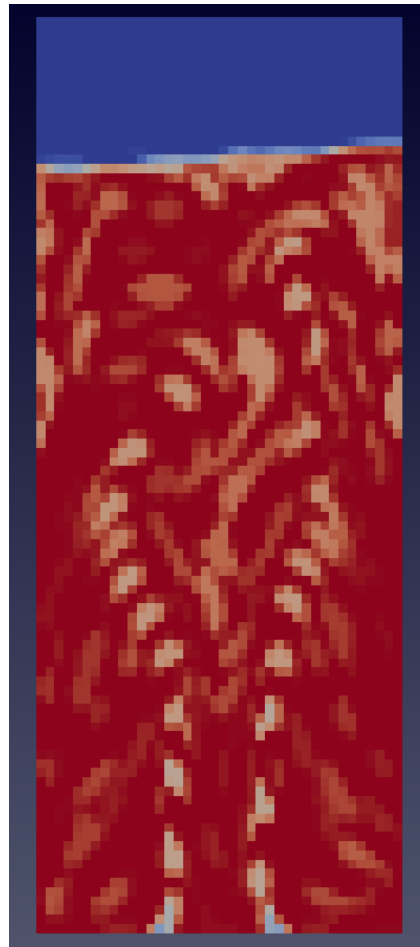
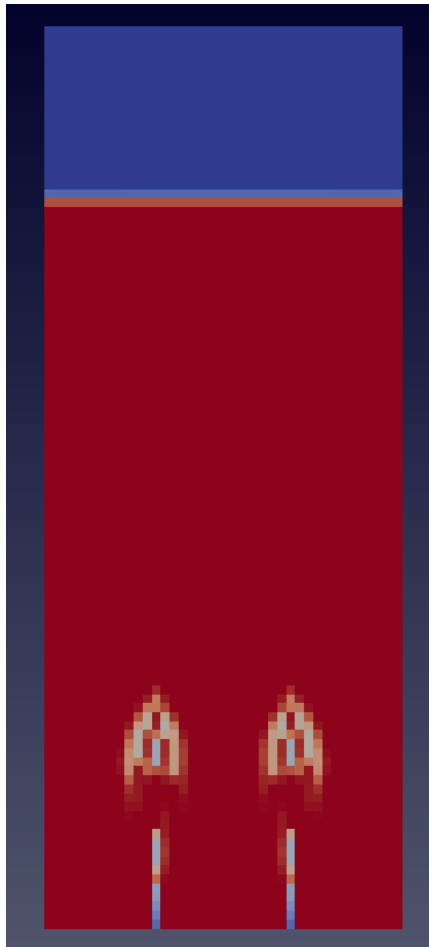
Ex. 1: Bubble reactor – Running

- Set the simulation final time to 10 s:
gedit system/controlDict
- Then, run the case:
interFoam

```
controlDict x
1 /*-----*
2 | =====|
3 | \ \ /   | F i e l d   | OpenFO
4 | \ \ /   | O p e r a t i o n   | Version
5 | \ \ /   | A n d   | Web:
6 | \ \ /   | M a n i p u l a t i o n   |
7 |*-----*
8 FoamFile
9 {
10     version      2.0;
11     format        ascii;
12     class         dictionary;
13     location      "system";
14     object        controlDict;
15 }
16 // *****
17
18 application      interFoam;
19
20 startFrom        startTime;
21
22 startTime        0;
23
24 stopAt           endTime;
25
26 endTime          10;
27
28 deltaT           0.001;
29
30 writeControl      adjustableRunTime;
31
32 writeInterval     0.05;
```

Ex. 1: Bubble reactor – Post processing

- Run paraFoam:
paraFoam

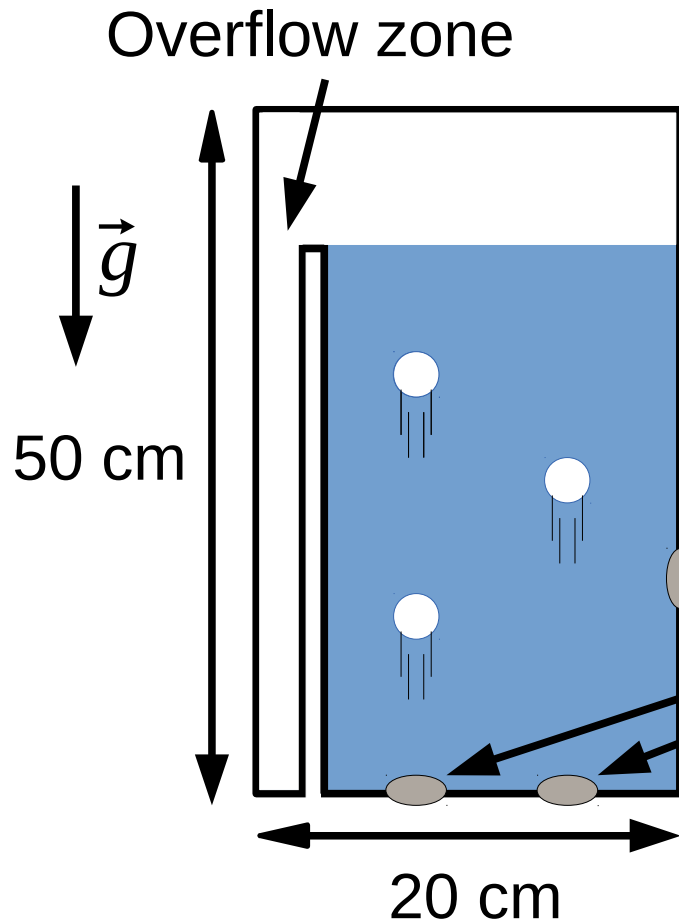


Ex. 2: Circulating reactor - Objectives

- Evolving an existing case
- Modifying mesh topology
- Selecting cell based on their locations
- Specifying the same boundary condition for several patches

Ex. 2: Circulating reactor – Case setup

- Solving multiphase flow, Volume Of Fluid equations (VOF) to describe rising bubbles



- Air-Water system, with 2 injection zones at the bottom of the reactor
- Water injection on the side

Water injection
Air injections

Ex. 2: Circulating reactor – Copying a case

- Go to your 'run' directory:
`run`
- Copy an existing case:
`cp -r Ex1 Ex2`
- Move to the case directory:
`cd Ex2`
- Clean the case directory:
`foamListTimes -rm`

Ex. 2: Circulating reactor – Modifying mesh

- Build the mesh:
`blockMesh`
- We are now going to draw the air and water inlets and the overflow
- Use `topoSet` to select faces and cells:
`gedit system/topoSetDict`

Ex. 2: Circulating reactor – topoSetDict

```
topoSetDict x createPatchDict x
1 /*----- C++ -----*/
2 |=====|
3 | \ \ / F i e l d | OpenFOAM: The Open Source
4 | \ \ / O p e r a t i o n | Version: 3.0.1
5 | \ \ / A n d | Web: www.OpenFOAM.or
6 | \ \ / M a n i p u l a t i o n |
7 /*-----*/
8 FoamFile
9 {
10     version      2.0;
11     format        ascii;
12     class         dictionary;
13     location      "system";
14     object        topoSetDict;
15 }
16 // *****
17
18 actions
19 (
20     // Selecting air inlets
21     {
22         name      faceGrabbed_1;
23         type      faceSet;
24         action     new;
25         source     boxToFace;
26         sourceInfo
27         {
28             box (0.10 -1 -0.0001) (0.1025 1 0.0001);
29         }
30     }
31     {
32         name      faceGrabbed_1;
33         type      faceSet;
34         action     add;
35         source     boxToFace;
36         sourceInfo
37         {
38             box (0.1475 -1 -0.0001) (0.15 1 0.0001);
39         }
40     }
41 )
```

```
42 // Selecting water inlet
43 {
44     name      faceGrabbed_2;
45     type      faceSet;
46     action     new;
47     source     boxToFace;
48     sourceInfo
49     {
50         box (0.199 -1 0.05) (0.201 1 0.06);
51     }
52 }
53
54 // Selecting overflow outlet
55 {
56     name      faceGrabbed_3;
57     type      faceSet;
58     action     new;
59     source     boxToFace;
60     sourceInfo
61     {
62         box (0.0 -1 -0.0001) (0.04 1 0.0001);
63     }
64 }
65
66 // Selecting overflow walls
67 {
68     name      overFlowWall;
69     type      cellSet;
70     action     clear;
71 }
72 {
73     name      overFlowWall;
74     type      cellSet;
75     action     invert;
76 }
77 {
78     name      overFlowWall;
79     type      cellSet;
80     action     delete;
81     source     boxToCell;
82     sourceInfo
83     {
84         box (0.04 -1 0) (0.05 1 0.40);
85     }
86 }
87 );
```

Ex. 2: Circulating reactor – Creating patches

- Modify the createPatchDict in order to create the patches:
gedit system/createPatchDict
- This dictionary is only used to create patches, the cell ablation will be handled by another tool
- The cell removing tool requires properly set boundary conditions. So, before calling it, we will also set the initial/boundary conditions

Ex. 2: Circulating reactor – Creating patches

```
createPatchDict x controlDict x topoSetDict x setFieldsDict
1 /*----- C++ -----*/
2 |=====|
3 | \ \ / F ield | OpenFOAM: The Open Source CFD
4 | \ \ / O peration | Version: 2.1.0
5 | \ \ / A nd | Web: www.OpenFOAM.org
6 | \ \ / M anipulation |
7 /*-----*/
8 FoamFile
9 {
10     version      2.0;
11     format        ascii;
12     class         dictionary;
13     object        createPatchDict;
14 }
15
16 // *****
17
18 pointSync false;
19
20 // Patches to create.
21 patches
22 (
23     {
24         // Name of new patch
25         name airInlets;
26
27         // Type of new patch
28         patchInfo
29         {
30             type patch;
31         }
32
33         // How to construct: either from 'patches' or 'set'
34         constructFrom set;
35
36         // If constructFrom = set : name of faceSet
37         set faceGrabbed_1;
38     }
39     {
40         // Name of new patch
41         name waterInlet;
42
43         // Type of new patch
44         patchInfo
45         {
46             type patch;
47         }
48
49         // How to construct: either from 'patches' or 'set'
50         constructFrom set;
51
52         // If constructFrom = set : name of faceSet
53         set faceGrabbed_2;
54     }
55     {
56         // Name of new patch
57         name overflow;
58
59         // Type of new patch
60         patchInfo
61         {
62             type patch;
63         }
64
65         // How to construct: either from 'patches' or 'set'
66         constructFrom set;
67
68         // If constructFrom = set : name of faceSet
69         set faceGrabbed_3;
70     }
71 );
```

Ex. 2: Circulating reactor – IC / BC

- Set initial/boundary conditions:
gedit 0/U 0/p_rgh 0/alpha.water.org

```
alpha.water.org x p_rgh x U x
1 /*----- C++ -----*/
2 |=====|
3 | \ \ / / | F i e l d | OpenFOAM: The Open Sou
4 | \ \ / / | O p e r a t i o n | Version: 3.0.0
5 | \ \ / / | A n d | Web: www.OpenFOAM
6 | \ \ / / | M a n i p u l a t i o n |
7 /*-----*/
8 FoamFile
9 {
10     version      2.0;
11     format        ascii;
12     class         volVectorField;
13     location      "0";
14     object        U;
15 }
16 // *****
17
18 dimensions      [0 1 -1 0 0 0 0];
19
20 internalField    uniform (0 0 0);
```

```
22 boundaryField
23 {
24     walls
25     {
26         type      fixedValue;
27         value      uniform (0 0 0);
28     }
29     frontAndBack
30     {
31         type      empty;
32     }
33     "(atmosphere|overflow)"
34     {
35         type      pressureInletOutletVelocity;
36         value      uniform (0 0 0);
37     }
38     airInlets
39     {
40         type      fixedValue;
41         value      uniform (0 0 0.20);
42     }
43     waterInlet
44     {
45         type      fixedValue;
46         value      uniform (-1 0 0);
47     }
48 }
49
50 // *****
```


Ex. 2: Circulating reactor – IC / BC

```
alpha.water.org x p_rgh x U x
1 /*-----* C++ -*
2 |=====|
3 | \ \ / F i e l d | OpenFOAM: The
4 | \ \ / O p e r a t i o n | Version: 3.0
5 | \ \ / A n d | Web: www
6 | \ \ / M a n i p u l a t i o n |
7 /*-----*
8 FoamFile
9 {
10     version      2.0;
11     format        ascii;
12     class         volScalarField;
13     object        p_rgh;
14 }
15 // *****
16
17 dimensions      [1 -1 -2 0 0 0];
18
19 internalField    uniform 0;
20
```

This syntax allows you you
to specify the same
boundary condition for
several patches

```
20
21 boundaryField
22 {
23     walls
24     {
25         type      fixedFluxPressure;
26         value      uniform 0;
27     }
28     frontAndBack
29     {
30         type      empty;
31     }
32     "(atmosphere|overflow)"
33     {
34         type      totalPressure;
35         p0        uniform 0;
36         U          U;
37         phi        phi;
38         rho        rho;
39         psi        none;
40         gamma      1;
41         value      uniform 0;
42     }
43     "(airInlets|waterInlet)"
44     {
45         type      fixedFluxPressure;
46         value      $internalField;
47     }
48 }
49
50 // *****
```



Ex. 2: Circulating reactor – IC / BC

- After modifying the file, copy the water original field:

cp 0/alpha.water.org 0/alpha.water


```
alpha.water.org x p_rgh x U x
1 /*-----*
2 |=====|
3 | \\\ / | F ield | OpenFOAM
4 | \\\ / | O peration | Version
5 | \\\ / | A nd | Web:
6 | \\\ / | M anipulation |
7 /*-----*
8 FoamFile
9 {
10     version      2.0;
11     format        ascii;
12     class          volScalarField;
13     location       "0";
14     object          alpha.water.org;
15 }
16 // *****
17
18 dimensions       [0 0 0 0 0 0 0];
19
20 internalField     uniform 0;
21
```

```
21
22 boundaryField
23 {
24     walls
25     {
26         type      zeroGradient;
27     }
28     frontAndBack
29     {
30         type      empty;
31     }
32     "(atmosphere|airInlets|overflow)"
33     {
34         type      inletOutlet;
35         inletValue uniform 0;
36         value      uniform 0;
37     }
38     waterInlet
39     {
40         type      inletOutlet;
41         inletValue uniform 1;
42         value      uniform 1;
43     }
44 }
```

Ex. 2: Circulating reactor – Modifying geometry

- Then, select the faces and the cells:
`topoSet`
- Create the patches:
`createPatch -overwrite`
- Delete the overflow cells from the mesh:
`subsetMesh -overwrite overFlowWall -patch walls`

Cell group name in
`topoSetDict`



Patch to which the
created boundary faces
will be attached to

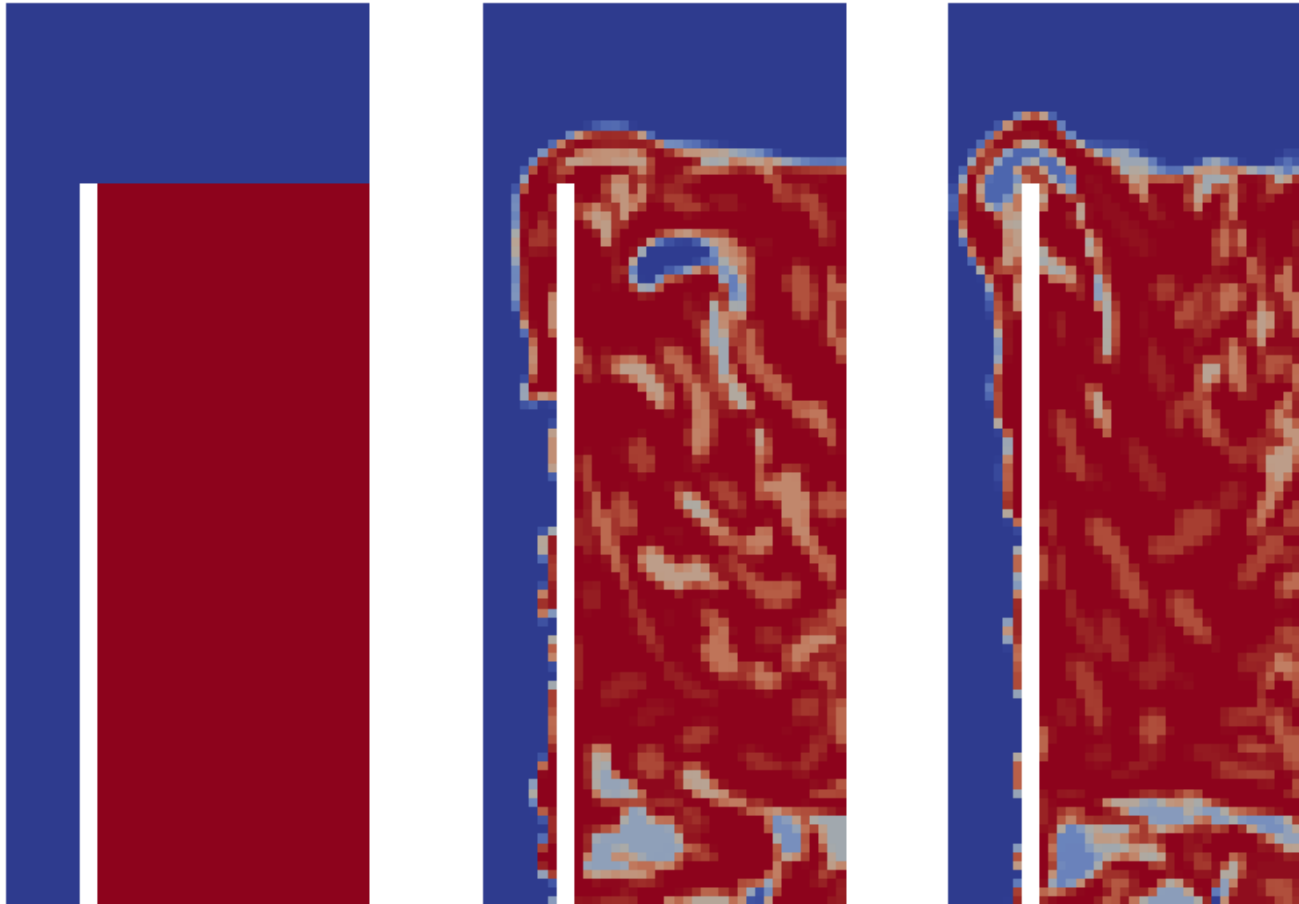
Ex. 2: Circulating reactor – Set water level

- Modify setFieldsDict to specify the initial water level in the reactor:
gedit system/setFieldsDict
- Then, apply it:
setFields
- Run the case:
interFoam

```
setFieldsDict x createPatchDict x controlDict x
1 /*----- C++ -----*/
2 |=====|
3 | \ \ \ \ / F i e l d | OpenFOAM: The Op
4 | \ \ \ \ / O p e r a t i o n | Version: 3.0.0
5 | \ \ \ \ / A n d | Web: www.Op
6 | \ \ \ \ / M a n i p u l a t i o n |
7 /*----- C++ -----*/
8 FoamFile
9 {
10     version      2.0;
11     format        ascii;
12     class         dictionary;
13     location      "system";
14     object        setFieldsDict;
15 }
16 // *****
17
18 defaultFieldValues
19 (
20     volScalarFieldValue alpha.water 0
21 );
22
23 regions
24 (
25     boxToCell
26     {
27         box (0.05 -1 0) (0.20 1 0.40);
28         fieldValues
29         (
30             volScalarFieldValue alpha.water 1
31         );
32     }
33 );
34
35
36 // *****
```

Ex. 2: Circulating reactor – Post processing

- Run paraFoam:
paraFoam

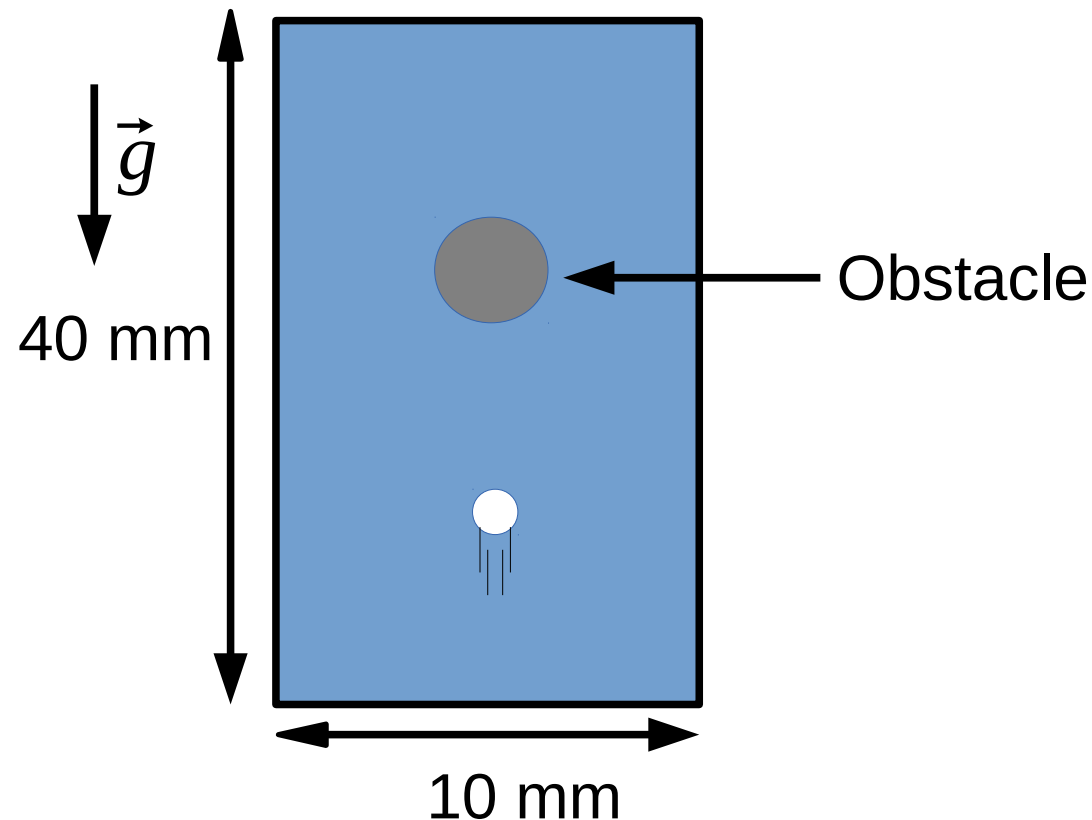


Ex. 3: Rising bubble - Objectives

- Describing the rise of an air bubble around an obstacle
- Setting up an auto refining mesh in order to finely track the bubble

Ex. 3: Rising bubble – Case setup

- Solving multiphase flow, Volume Of Fluid equations (VOF) to describe rising bubbles

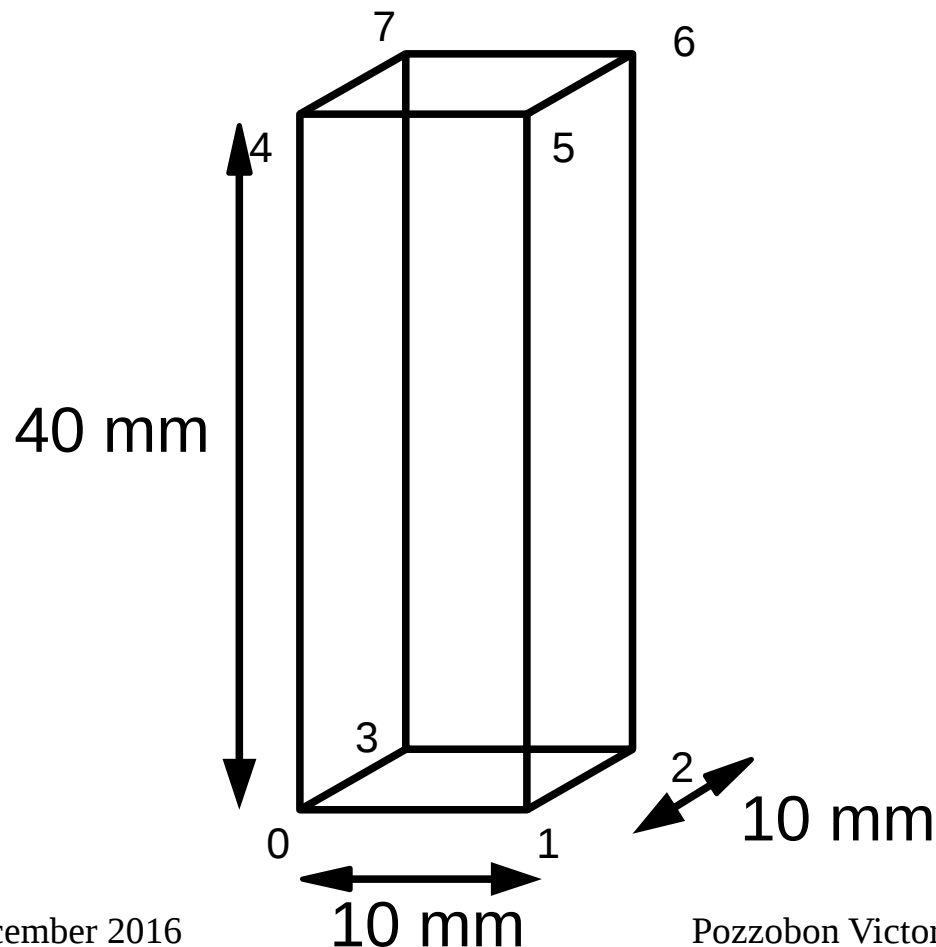


Ex. 3: Rising bubble – Copying a case

- Go to your 'run' directory:
`run`
- Copy an existing case:
`cp -r Ex2 Ex3`
- Move to the case directory:
`cd Ex3`
- Clean the case directory:
`foamListTimes -rm`

Ex. 3: Rising bubble – Modifying mesh

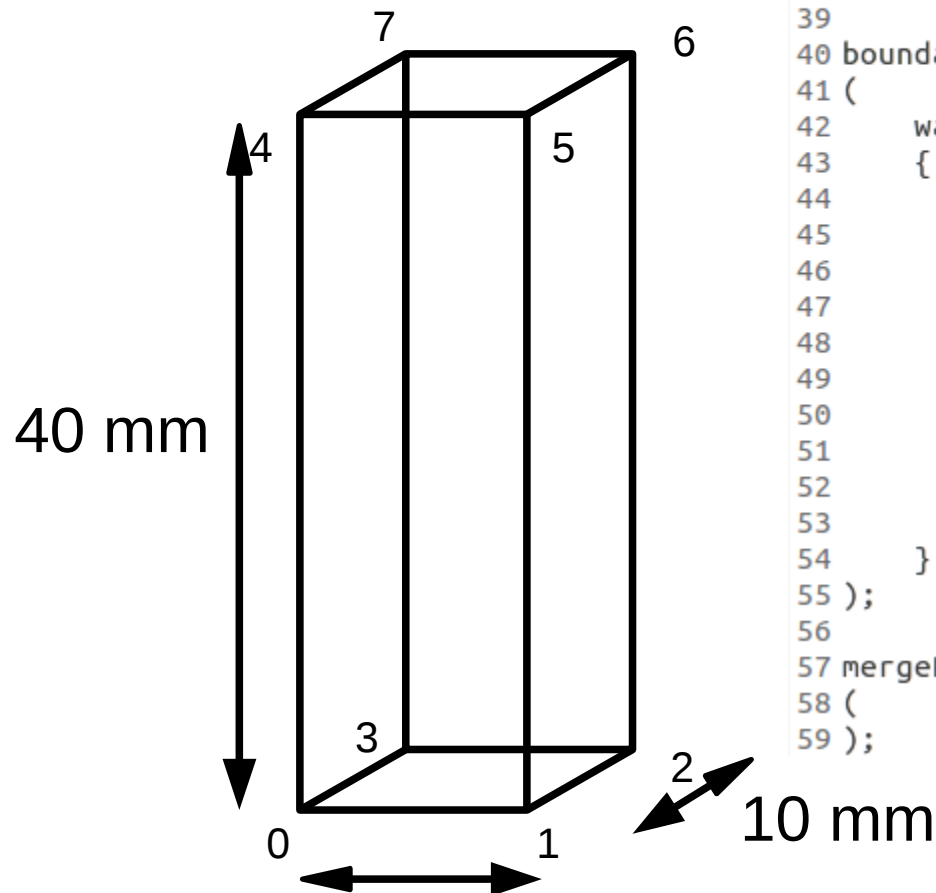
- Modify the mesh:
gedit system/blockMeshDict



```
blockMeshDict x
1 /*-----
2 | =====
3 | \\      /   F i e l d
4 | \\      /   O p e r a t i o n
5 | \\      /   A n d
6 | \\      /   M a n i p u l a t i o n
7 /*-----
8 FoamFile
9 {
10     version      2.0;
11     format        ascii;
12     class         dictionary;
13     object        blockMeshDict;
14 }
15 // *****
16
17 convertToMeters 0.001;
18
19 vertices
20 (
21     (0 0 0)
22     (10 0 0)
23     (10 10 0)
24     (0 10 0)
25     (0 0 40)
26     (10 0 40)
27     (10 10 40)
28     (0 10 40)
29 );
```

Ex. 3: Rising bubble – Modifying mesh

- Build the mesh:
blockMesh



```
31 blocks
32 (
33     hex (0 1 2 3 4 5 6 7) (10 10 40) simpleGrading (1 1 1)
34 );
35
36 edges
37 (
38 );
39
40 boundary
41 (
42     walls
43     {
44         type wall;
45         faces
46         (
47             (0 1 2 3)
48             (1 2 6 5)
49             (0 3 7 4)
50             (4 5 6 7)
51             (0 1 5 4)
52             (3 2 6 7)
53         );
54     }
55 );
56
57 mergePatchPairs
58 (
59 );
```

Ex. 3: Rising bubble – topoSetDict

- Select the cells to create the obstacle:
gedit system/topoSetDict
- Then, select the cells:
topoSet

```
topoSetDict x
1 /*-----*-- C++ *-----*/
2 |=====|
3 | \ \ \ / | F i e l d | OpenFOAM: The Open Source CFD Toolbox
4 | \ \ \ / | O p e r a t i o n | Version: 3.0.1
5 | \ \ \ / | A n d | Web: www.OpenFOAM.org
6 | \ \ \ / | M a n i p u l a t i o n |
7 |-----*--*/
8 FoamFile
9 {
10     version      2.0;
11     format        ascii;
12     class          dictionary;
13     location       "system";
14     object         topoSetDict;
15 }
16 // *****
17
18 actions
19 (
20     // Selecting obstacle cell
21     {
22         name      obstacle;
23         type       cellSet;
24         action     clear;
25     }
26     {
27         name      obstacle;
28         type       cellSet;
29         action     invert;
30     }
31     {
32         name      obstacle;
33         type       cellSet;
34         action     delete;
35         source     cylinderToCell;
36         sourceInfo
37         {
38             p1      (0.005 0.0 0.025); // start point on cylinder axis
39             p2      (0.005 1 0.025); // end point on cylinder axis
40             radius   0.002;
41         }
42     }
43 );
```

Ex. 3: Rising bubble – Creating the obstacle

- Restore alpha.water field:
`cp 0/alpha.water.org 0/alpha.water`
- Delete the formerly selected cells:
`subsetMesh -overwrite obstacle -patch walls`

Ex. 3: Rising bubble – Creating the bubble

- Create the bubble:
gedit system/setFieldsDict
- Then:
setFields

```
setFieldsDict x
1  /*----- C++ -----*/
2  |=====|
3  | \ \ \ \ | F i e l d | OpenFOAM: The Open Source CFD Toolbox
4  | \ \ \ \ | O p e r a t i o n | Version: 3.0.0
5  | \ \ \ \ | A n d | Web: www.OpenFOAM.org
6  | \ \ \ \ | M a n i p u l a t i o n |
7  | *-----*/
8  FoamFile
9  {
10     version      2.0;
11     format        ascii;
12     class         dictionary;
13     location      "system";
14     object        setFieldsDict;
15 }
16 // *****
17
18 defaultFieldValues
19 (
20     volScalarFieldValue alpha.water 0
21 );
22
23 regions
24 (
25     boxToCell
26     {
27         box (0.0 -1 0) (0.020 1 0.05);
28         fieldValues
29         (
30             volScalarFieldValue alpha.water 1
31         );
32     }
33     cylinderToCell
34     {
35         p1 (0.005 0.0 0.005); // start point on cylinder axis
36         p2 (0.005 1 0.005); // end point on cylinder axis
37         radius 0.002;
38         fieldValues
39         (
40             volScalarFieldValue alpha.water 0
41         );
42     }
43 );
```

Ex. 3: Rising bubble – IC / BC, running the case

- The case could be run as is. The initial and boundary conditions for walls patch have been set in the former exercise
- Instead, we will use an automatic mesh refinement routine to track the bubble interface and increase resolution accuracy at the bubble frontiers
- Copy the dynamic mesh dictionary:
cp
\$FOAM_TUTORIALS/multiphase/interDyMFoam/ras/damBreakWithObstacle/constant/dynamicMeshDict constant/.

Ex. 3: Rising bubble – dynamicMeshDict

Edit the dictionary:
gedit constant/dynamicMeshDict

```
35 // Stop refinement if maxCells reached
36 maxCells      200000;
37 // Flux field and corresponding velocity field. Fluxes on changed
38 // faces get recalculated by interpolating the velocity. Use 'none'
39 // on surfaceScalarFields that do not need to be reinterpolated.
40 correctFluxes
41 (
42     (alphaPhi none)
43     (phi none)
44     (nHatf none)
45     (rhoPhi none)
46     (ghf none)
47 );
48 // Write the refinement level as a volScalarField
49 dumpLevel      true;
50 }
```

Field that is used as refinement criterion

Between this bounds the mesh will be refined

Maximum number of successive refinement (≥ 1)

```
dynamicMeshDict x
1 /*-----*----- C++ *-----*/
2 |=====|
3 | \ \ \ \ / | F i e l d | OpenFOAM: The Open Sou
4 | \ \ \ \ / | O p e r a t i o n | Version: 3.0.0
5 | \ \ \ \ / | A n d | Web: www.OpenFOAM
6 | \ \ \ \ / | M a n i p u l a t i o n |
7 /*-----*-----*/
8 FoamFile
9 {
10     version      2.0;
11     format        ascii;
12     class          dictionary;
13     location       "constant";
14     object          dynamicMeshDict;
15 }
16 // * * * * *
17
18 dynamicFvMesh      dynamicRefineFvMesh;
19
20 dynamicRefineFvMeshCoeffs
21 {
22     // How often to refine
23     refineInterval  1;
24     // Field to be refinement on
25     field            alpha.water;
26     // Refine field inbetween lower..upper
27     lowerRefineLevel 0.001;
28     upperRefineLevel 0.999;
29     // If value < unrefineLevel unrefine
30     unrefineLevel    10;
31     // Have slower than 2:1 refinement
32     nBufferLayers    1;
33     // Refine cells only up to maxRefinement levels
34     maxRefinement    2;
35 }
```

Ex. 3: Rising bubble – fvSolution

- Copy the fvSolution file to prescribe the pressure for the solver:

```
cp
```

```
$FOAM_TUTORIALS/multiphase/interDyMFoam/ras/dam  
BreakWithObstacle/system/fvSolution system/.
```

```
gedit system/fvSolution
```


Ex. 3: Rising bubble – fvSolution

```

fvSolution x
1 /*----- C-
2 |=====
3 | \\ / F i e l d       | OpenFOAM
4 | \\ / O p e r a t i o n | Version:
5 | \\ / A n d             | Web:
6 | \\ / M a n i p u l a t i o n |
7 |-----
8 FoamFile
9 {
10     version      2.0;
11     format        ascii;
12     class         dictionary;
13     location      "system";
14     object        fvSolution;
15 }
16 // *****
17
18 solvers
19 {
20     "alpha.water.*"
21     {
22         nAlphaCorr      1;
23         nAlphaSubCycles 3;
24         cAlpha          1;
25     }
26
27     p_rgh
28     {
29         solver          GAMG;
30         tolerance       1e-08;
31         relTol          0.01;
32         smoother        DIC;
33         nPreSweeps      0;
34         nPostSweeps     2;
35         nFinestSweeps   2;
36         cacheAgglomeration false;
37         nCellsInCoarsestLevel 10;
38         agglomerator    faceAreaPair;
39         mergeLevels     1;
40     }
41
42     p_rghFinal
43     {
44         $p_rgh;
45         relTol          0;
46
47
48     "pcorr.*"
49     {
50         $p_rghFinal;
51         tolerance       0.0001;
52     }
53
54     U
55     {
56         solver          smoothSolver;
57         smoother        GaussSeidel;
58         tolerance       1e-06;
59         relTol          0;
60         nSweeps         1;
61     }
62
63     "(k|omega|B|nuTilda).*"
64     {
65         solver          smoothSolver;
66         smoother        symGaussSeidel;
67         tolerance       1e-08;
68         relTol          0;
69     }
70 }
71
72 PIMPLE
73 {
74     momentumPredictor no;
75     nCorrectors        3;
76     nNonOrthogonalCorrectors 0;
77
78     pRefPoint          (0.001 0.001 0.001);
79     pRefValue          0;
80 }

```

Ex. 3: Rising bubble – Running the case

- Modify the controlDict:
gedit system/controlDict
- Then, run the case:
interDyMFoam



(DyM stands for DYnamic Mesh)

```
18 application      interDyMFoam;
19
20 startFrom         startTime;
21
22 startTime         0;
23
24 stopAt            endTime;
25
26 endTime           0.40;
27
28 deltaT            0.001;
29
30 writeControl       adjustableRunTime;
31
32 writeInterval     0.01;
33
34 purgeWrite        0;
35
36 writeFormat       ascii;
37
38 writePrecision     6;
39
40 writeCompression  uncompressed;
41
42 timeFormat         general;
43
44 timePrecision      6;
45
46 runTimeModifiable yes;
47
48 adjustTimeStep     yes;
49
50 maxCo              0.75;
51 maxAlphaCo         0.75;
52 maxDeltaT          1;
```

Ex. 3: Rising bubble – Post processing

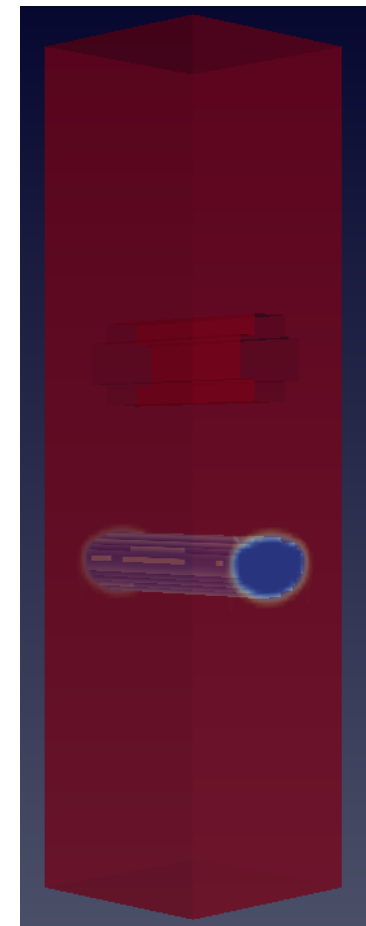
- Run paraFoam:

$t = 0.10 \text{ s}$

Without
refinement



With
refinement



Ex. 3: Rising bubble – Post processing

$t = 0.20 \text{ s}$

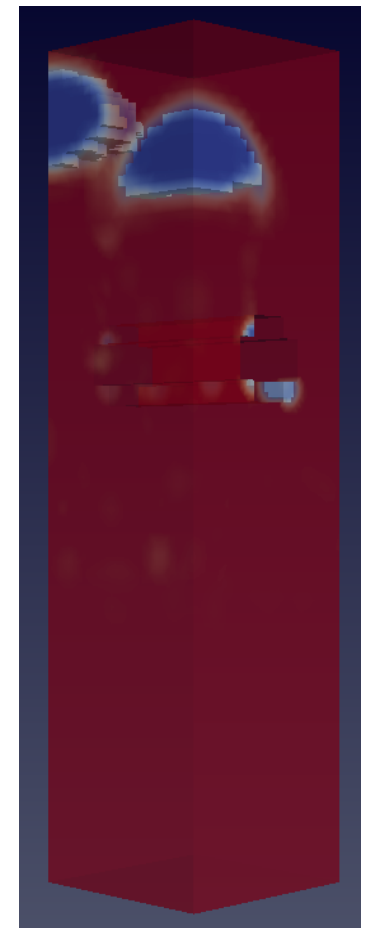
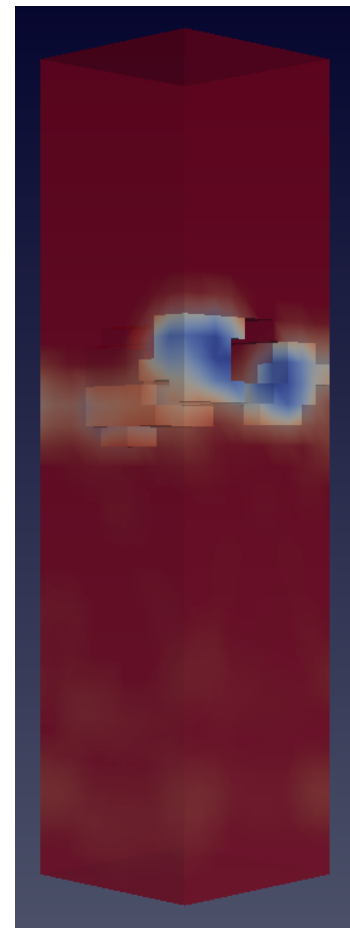
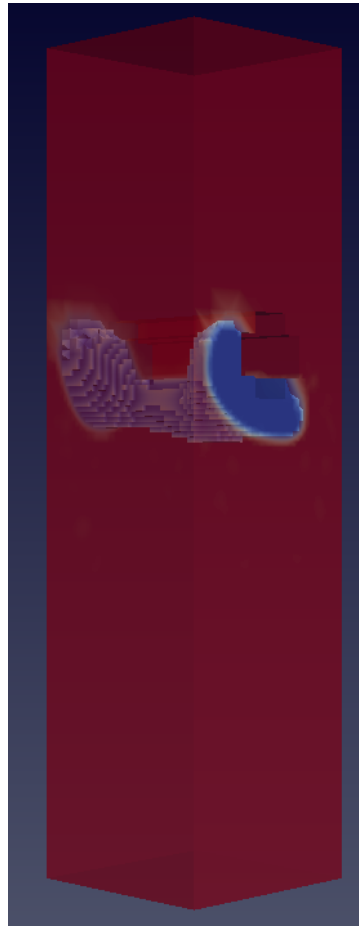
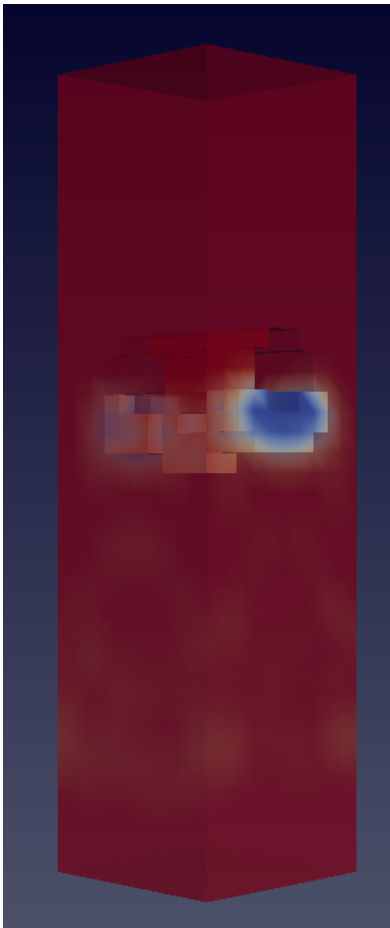
$t = 0.30 \text{ s}$

Without
refinement

With
refinement

Without
refinement

With
refinement

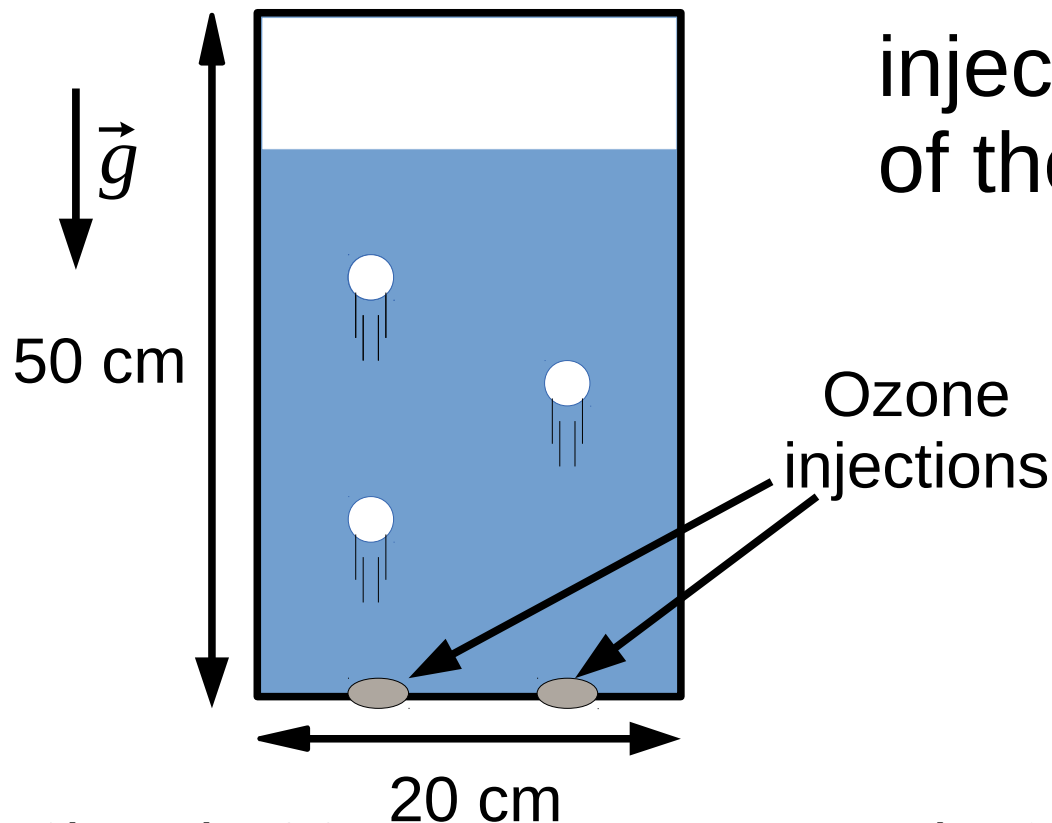


Ex. 4: Ozone tower - Objectives

- Adding mass transfer to a multiphase case
- Modifying a multiphase solver

Ex. 4: Ozone tower – Case setup

- Solving multiphase flow, Volume Of Fluid equations (VOF) to describe rising bubbles
- Air-Water system, with 2 air injection zones at the bottom of the reactor
- Mass transfer between phase is taken into account



Ex. 4: Ozone tower – Case setup

- Ozone transport equation has to account for the interface gap (through Henry constant) and the difference of mass diffusivity in gas and liquid

$$\frac{dO_3}{dt} + \nabla \cdot (\vec{U} O_3) = \nabla^2 \left(\frac{D_l D_g}{\alpha D_g + (1 - \alpha) D_l} \frac{1 - H}{\alpha H + (1 - \alpha)} \alpha O_3 \right)$$

- H being Henry's constant and α the liquid phase indicator

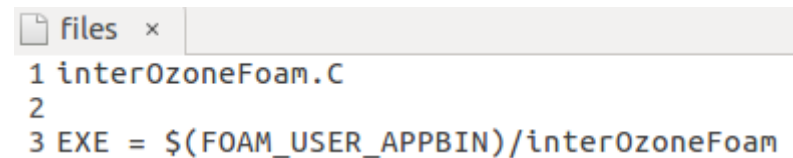
(Credit to Cyp)

Ex. 4: Ozone tower – Solver creation

- Reach 'run' directory:
`run`
- Move to solver directory:
`cd solvers`
- Copy interFoam:
`cp -r $FOAM_APP/solvers/multiphase/interFoam/ .`
`mv interFoam interOzoneFoam`
- Move to the new solver directory:
`cd interOzoneFoam`

Ex. 4: Ozone tower – Solver creation / modification

- Clean the directory:
`rm -r interDyMFoam/ interMixingFoam/`
- Clean the directory:
`wclean`
- Rename interFoam:
`mv interFoam.C interOzoneFoam.C`
- Change compilation file:
`gedit Make/files`



A screenshot of a text editor window titled 'files'. The window shows the following content:

```
1 interOzoneFoam.C
2
3 EXE = $(FOAM_USER_APPBIN)/interOzoneFoam
```

Ex. 4: Ozone tower – Solver modification / createFields.H

- Add O3 field and physical properties to createFields.H:
gedit createFields.H

```
136 // MULES Correction
137 tmp<surfaceScalarField> talphaPhiCorr0;
138
139 Info<< "Reading field O3\n" << endl;
140 volScalarField O3
141 (
142     IOobject
143     (
144         "O3",
145         runTime.timeName(),
146         mesh,
147         IOobject::MUST_READ,
148         IOobject::AUTO_WRITE
149     ),
150     mesh
151 );
152
153 IOdictionary transportProperties
154 (
155     IOobject
156     (
157         "transportProperties",
158         runTime.constant(),
159         mesh,
160         IOobject::MUST_READ_IF_MODIFIED,
161         IOobject::NO_WRITE
162     )
163 );
164
165 dimensionedScalar H
166 (
167     transportProperties.lookup("H")
168 );
169
170 dimensionedScalar Diff_l
171 (
172     transportProperties.lookup("Diff_l")
173 );
174
175 dimensionedScalar Diff_g
176 (
177     transportProperties.lookup("Diff_g")
178 );
```

Ex. 4: Ozone tower – modifying solver

- Add the scalar transport equation:
gedit interOzoneFoam.C

```
117         // --- Pressure corrector loop
118         while (pimple.correct())
119         {
120             #include "pEqn.H"
121         }
122
123         if (pimple.turbCorr())
124         {
125             turbulence->correct();
126         }
127     }
128
129     //- Ozone field
130     solve
131     (
132         fvm::ddt(O3)
133         + fvm::div(phi, O3)
134         ==
135         fvc::laplacian(O3 * Diff_l * Diff_g / (alpha1 * Diff_g + (1 - alpha1) * Diff_l)
136                     * (1 - H) / (alpha1 * H + (1 - alpha1))
137                     , alpha1)
138     );
139
140     runTime.write();
141
142     Info<< "ExecutionTime = " << runTime.elapsedCpuTime() << " s"
143         << "   ClockTime = " << runTime.elapsedClockTime() << " s"
144         << nl << endl;
```

Ex. 4: Ozone tower – Solver compilation / case creation

- Clean the directory and compile:
`wclean; wmake`
- Move to cases directory:
`run`
- Copy Ex1 directory:
`cp -r Ex1 Ex4`
- Move to the case directory:
`cd Ex4`

Ex. 4: Ozone tower – Add physical properties

- Clean the directory and compile:
gedit constant/transportProperties

```
18 phases (water air);
19
20 water
21 {
22     transportModel    Newtonian;
23     nu                 [0 2 -1 0 0 0 0] 1e-06;
24     rho                [1 -3 0 0 0 0 0] 1000;
25 }
26
27 air
28 {
29     transportModel    Newtonian;
30     nu                 [0 2 -1 0 0 0 0] 1.48e-05;
31     rho                [1 -3 0 0 0 0 0] 1;
32 }
33
34 sigma                [1 0 -2 0 0 0 0] 0.07;
35
36 H                    H [0 0 0 0 0 0 0] 0.10;
37 Diff_l               Diff_l [0 2 -1 0 0 0 0] 1e-9;
38 Diff_g               Diff_g [0 2 -1 0 0 0 0] 1e-6;
```

Ex. 4: Ozone tower – IC / BC

- Set initial/boundary conditions for ozone:
cp 0/p_rgh 0/O3
gedit 0/O3

```
8 FoamFile
9 {
10     version      2.0;
11     format        ascii;
12     class         volScalarField;
13     object        03;
14 }
15 // * * * * *
16
17 dimensions       [1 -3 0 0 0 0 0];
18
19 internalField     uniform 0;
20
21 boundaryField
22 {
23     walls
24     {
25         type      zeroGradient
26     }
27     frontAndBack
28     {
29         type      empty;
30     }
31     atmosphere
32     {
33         type      inletOutlet;
34         inletValue uniform 0.0;
35         value      uniform 0.0;
36     }
37     inlets
38     {
39         type      inletOutlet;
40         inletValue uniform 1;
41         value      uniform 1;
42     }
43 }
```

Ex. 4: Ozone tower – Numerical schemes

- Add O3 convection to the numerical schemes:
gedit system/fvSchemes

```
18 ddtSchemes
19 {
20     default          Euler;
21 }
22
23 gradSchemes
24 {
25     default          Gauss linear;
26 }
27
28 divSchemes
29 {
30     div(rhoPhi,U)    Gauss upwind;
31     div(phi,alpha)   Gauss vanLeer;
32     div(phirb,alpha) Gauss linear;
33     div(phi,k)        Gauss upwind;
34     div(phi,omega)    Gauss upwind;
35     div(((rho*nuEff)*dev2(T(grad(U)))))) Gauss linear;
36     div(phi,O3)      Gauss vanLeer;
37 }
```

Ex. 4: Ozone tower – Solver settings

- Add O3 to the solved fields:
gedit system/fvSolution

```
63     "(k|omega|B|nuTilda).*"
64     {
65         solver      smoothSolver;
66         smoother     symGaussSeidel;
67         tolerance    1e-08;
68         relTol        0;
69     }
70     O3
71     {
72         solver      BICCG;
73         preconditioner DILU;
74         tolerance    1e-06;
75         relTol        0;
76     }
77 }
78
79 PIMPLE
80 {
81     momentumPredictor no;
82     nCorrectors      3;
83     nNonOrthogonalCorrectors 0;
84
85     pRefPoint        (0.51 0.51 0.51);
86     pRefValue         0;
87 }
```

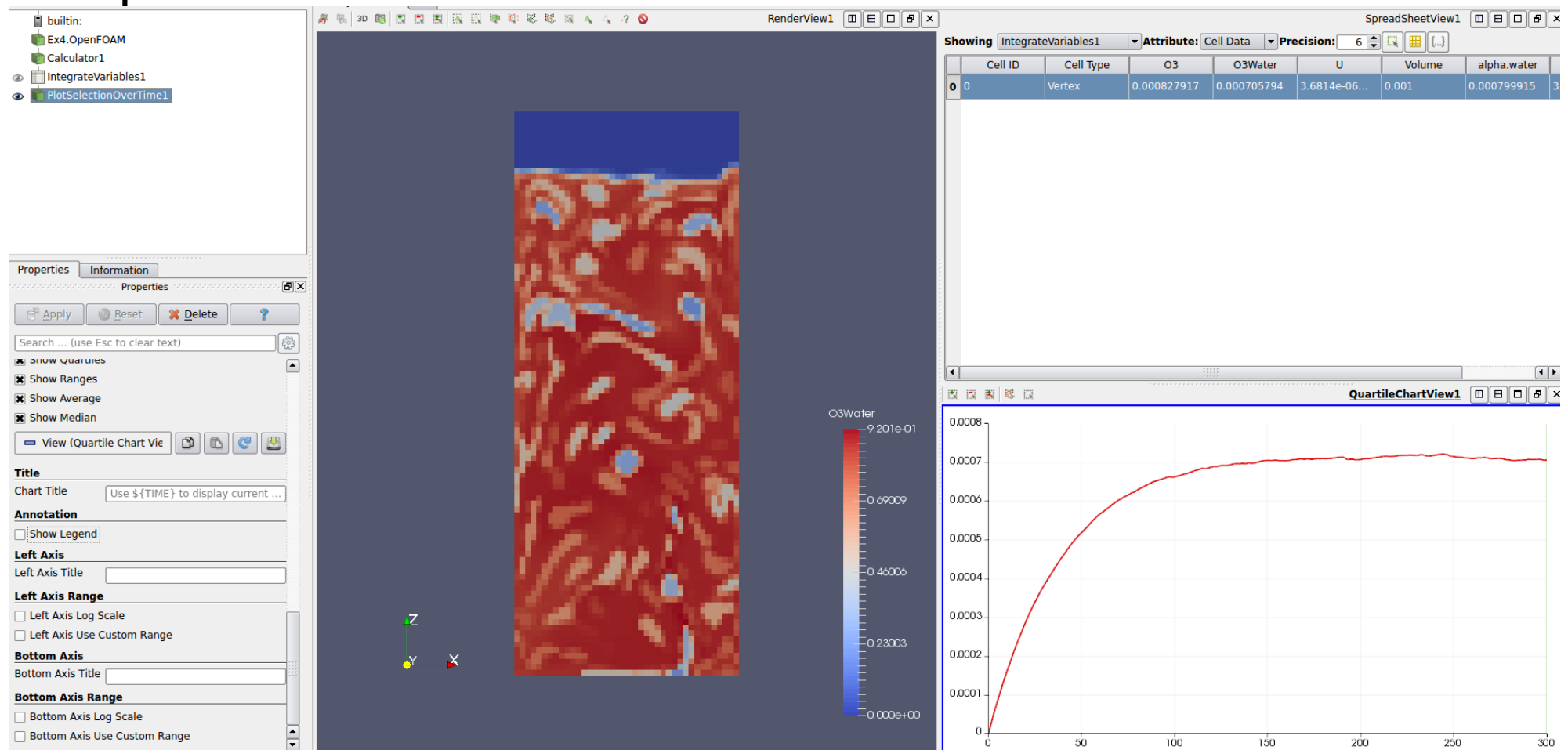

Ex. 4: Ozone tower – Run settings

- Change the run duration and the timestep extraction interval:
gedit system/controlDict

```
8 FoamFile
9 {
10     version      2.0;
11     format        ascii;
12     class         dictionary;
13     location      "system";
14     object        controlDict;
15 }
16 // * * * * *
17
18 application      interFoam;
19
20 startFrom        startTime;
21
22 startTime        0;
23
24 stopAt           endTime;
25
26 endTime          300;
27
28 deltaT           0.001;
29
30 writeControl      adjustableRunTime;
31
32 writeInterval     1;
```

Ex. 4: Ozone tower – Process

- Plot the amount of O3 dissolved in water:
paraFoam



Ex. 5: Turbulent pipe - Objectives

- Dealing with turbulence with RANS approach
- Evolving a case from laminar to turbulent flow
- We are going to use “Exercise 5 – 2D pipe” from the first tutorial

Ex. 5: Turbulent pipe – Case setup

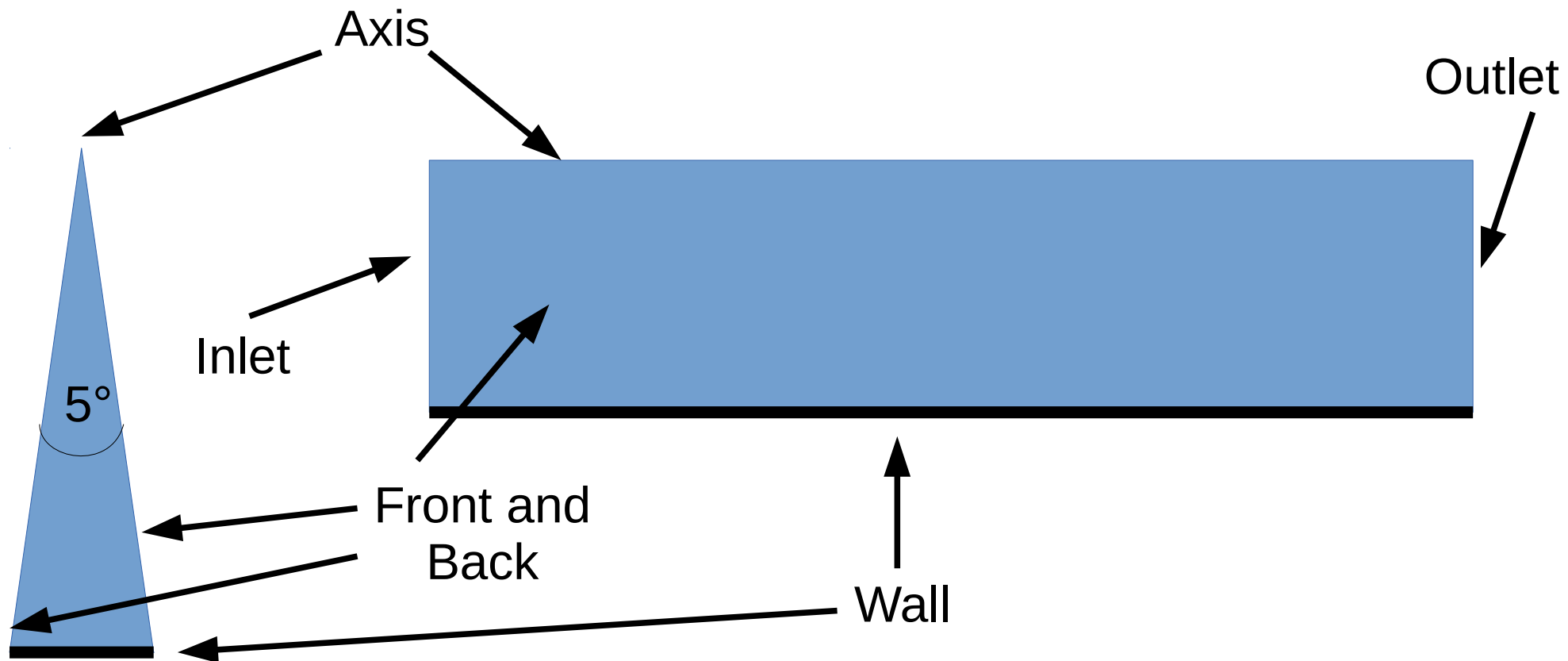
- Solving incompressible turbulent flow in a 2D axisymmetrical pipe, in steady state

$$\frac{d\bar{U}}{dt} + \bar{U} \nabla \cdot \bar{U} = -\nabla \frac{\bar{p}}{\rho} + \eta \nabla^2 \bar{U} + \text{Turbulence}$$



Ex. 5: Turbulent pipe – Mesh

- The geometry looks like that:



Ex. 5: Turbulent pipe – Case creation

- Reach 'run' directory:
`run`
- Copy an existing case:
`cp -r First/Tutorial/Cases/Folder/Ex5 Ex5`
- Go to the new case directory:
`cd Ex5`
- Clean the case directory:
`foamListTimes -rm`

Ex. 5: Turbulent pipe – Case creation

- Modify the mesh:
gedit system/blockMeshDict

```
41 boundary
42 (
43     inlet
44     {
45         type patch;
46         faces
47         (
48             (0 5 6 0)
49         );
50     }
51     outlet
52     {
53         type patch;
54         faces
55         (
56             (4 7 8 4)
57         );
58     }
59     wall
60     {
61         type wall;
62         faces
63         (
64             (5 6 8 7)
65         );
66     }
67     front
68     {
69         type wedge;
70         faces
71         (
72             (0 4 7 5)
73         );
74     }
75     back
76     {
77         type wedge;
78         faces
79         (
80             (0 4 8 6)
81         );
82     }
83     axis
84     {
85         type patch;
86         faces
87         (
88             (0 4 4 0)
89         );
90     }
91 );
```

Ex. 5: Turbulent pipe – Setting turbulence model

- Modify the viscosity so that the fluid is now water:
gedit constant/transportProperties

```
1  /*-----*-- C++
2  |=====|
3  | \ \ /  | F i e l d      | OpenFOAM: T
4  | \ \ /  | O peration    | Version: 3
5  | \ \ /  | A nd          | Web:      w
6  | \ \ /  | M anipulation  |
7  /*-----*--
8  FoamFile
9  {
10     version      2.0;
11     format        ascii;
12     class         dictionary;
13     location      "constant";
14     object         transportProperties;
15 }
16 // *****
17
18 transportModel  Newtonian;
19
20 nu              [0 2 -1 0 0 0 0] 1e-06;
21
22 // *****
```


Ex. 5: Turbulent pipe – Setting turbulence model

- Modify the turbulence model:
gedit constant/turbulenceProperties

```
turbulenceProperties x
1 /*-----*
2 | ===== |
3 | \ \ \ \ / | F i e l d | O p e n F O A
4 | \ \ \ \ / | O p e r a t i o n | V e r s i o n
5 | \ \ \ \ / | A n d | W e b :
6 | \ \ \ \ / | M a n i p u l a t i o n |
7 /*-----*
8 FoamFile
9 {
10     version      2.0;
11     format        ascii;
12     class         dictionary;
13     location      "constant";
14     object        turbulenceProperties;
15 }
16 // *****
17
18 simulationType RAS;
19
20 RAS
21 {
22     RASModel      kEpsilon;
23
24     turbulence     on;
25
26     printCoeffs   on;
27
28     kEpsilonCoeffs
29     {
30         Cmu        0.09;
31         C1          1.44;
32         C2          1.92;
33         sigmaEps    1.44;
34     }
35 }
```

Ex. :: Turbulent pipe – Modifying IC / BC

- Create file for k and epsilon file. Increase inlet velocity so that the flow becomes turbulent:

cp 0/p 0/k

cp 0/p 0/epsilon

cp 0/p 0/nut

gedit 0/U 0/k 0/epsilon 0/nut

Ex. 5: Turbulent pipe – Modifying U

- Increase inlet velocity up to 1 m/s:

```
1 /*-----*
2 | =====
3 | \\\ / F i e l d       | O p e n F
4 | \\\ / O p e r a t i o n | V e r s i
5 | \\\ / A n d           | W e b :
6 | \\\ / M a n i p u l a t i o n |
7 |-----*
8 FoamFile
9 {
10     version      2.0;
11     format        ascii;
12     class         volVectorField;
13     object        U;
14 }
15 // *****
16
17 dimensions       [0 1 -1 0 0 0 0];
18
19 internalField     uniform (0 0 0);
20
```

```
21 boundaryField
22 {
23     inlet
24     {
25         type       fixedValue;
26         value       uniform (0 0 1);
27     }
28
29     outlet
30     {
31         type       zeroGradient;
32     }
33
34     wall
35     {
36         type       fixedValue;
37         value       uniform (0 0 0);
38     }
39
40     "(front|back)"
41     {
42         type       wedge;
43     }
44     axis
45     {
46         type       zeroGradient;
47     }
48 }
```

Ex. 5: Turbulent pipe – Modifying k

- Set up turbulent kinetic energy field:

```
1 /*-----*
2 |=====|
3 | \\\ / | F ield | OpenF
4 | \\\ / | O peration | Versi
5 | \\\ / | A nd | Web:
6 | \\\ / | M anipulation |
7 |-----*
8 FoamFile
9 {
10     version      2.0;
11     format        ascii;
12     class         volScalarField;
13     object        k;
14 }
15 // *****
16
17 dimensions       [0 2 -2 0 0 0];
18
19 internalField     uniform 3.38E-02;
```

```
21 boundaryField
22 {
23     inlet
24     {
25         type      fixedValue;
26         value      uniform 3.75E-03;
27     }
28
29     outlet
30     {
31         type      zeroGradient;
32     }
33
34     wall
35     {
36         type      zeroGradient;
37         //type      kqRWallFunction;
38         //value      uniform 3.38E-02;
39     }
40
41     "(front|back)"
42     {
43         type      wedge;
44     }
45
46     axis
47     {
48         type      zeroGradient;
49     }
```

Ex. 5: Turbulent pipe – Modifying epsilon

- Set up dissipation field:

```
epsilon x k x p x U x
1 /*-----
2 |=====
3 | \ \ \ \ \ / F i e l d   | O p e r
4 | \ \ \ \ \ / O p e r a t i o n   | V e r s
5 | \ \ \ \ \ / A n d   | W e b:
6 | \ \ \ \ \ / M a n i p u l a t i o n   |
7 |-----
8 FoamFile
9 {
10     version      2.0;
11     format        ascii;
12     class         volScalarField;
13     location      "0";
14     object        epsilon;
15 }
16 // *****
17
18 dimensions       [0 2 -3 0 0 0 0];
19
20 internalField     uniform 2.04E-01;
21
```

```
22 boundaryField
23 {
24     inlet
25     {
26         type      fixedValue;
27         value      uniform 7.55E-03;
28     }
29
30     outlet
31     {
32         type      zeroGradient;
33     }
34
35     wall
36     {
37         type      epsilonWallFunction;
38         value      uniform 2.04E-01;
39     }
40
41     "(front|back)"
42     {
43         type      wedge;
44     }
45     axis
46     {
47         type      zeroGradient;
48     }
49 }
```

Ex. 5: Turbulent pipe – Modifying nut

- Set up turbulent viscosity field:

```
nut x
1 /*-----*/
2 |=====|
3 | \      / | F i e l d | O p e n |
4 |  \    /  | O p e r a t i o n | V e r s |
5 |   \  /   | A n d | W e b : |
6 |    \/    | M a n i p u l a t i o n |
7 |-----*/
8 FoamFile
9 {
10     version      2.0;
11     format        ascii;
12     class         volScalarField;
13     location      "0";
14     object        nut;
15 }
16 // *****
17
18 dimensions      [0 2 -1 0 0 0 0];
19
20 internalField    uniform 0;
```

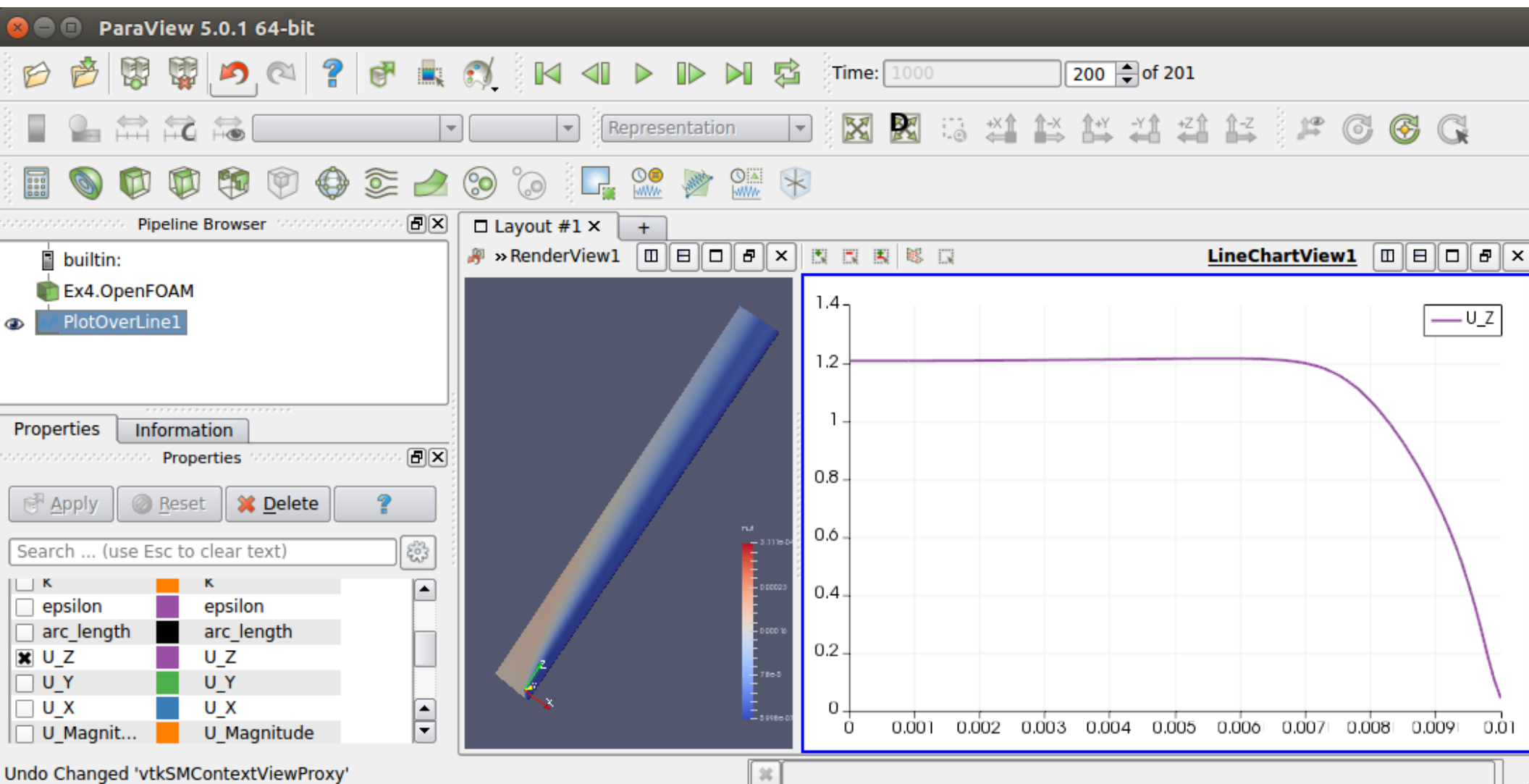
```
22 boundaryField
23 {
24     inlet
25     {
26         type      calculated;
27         value      uniform 0;
28     }
29
30     outlet
31     {
32         type      calculated;
33         value      uniform 0;
34     }
35
36     wall
37     {
38         type      calculated;
39         value      uniform 0;
40     }
41     "(front|back)"
42     {
43         type      wedge;
44     }
45     axis
46     {
47         type      zeroGradient;
48     }
49 }
50 }
```

Ex. 5: Turbulent pipe – Running the case

- Increase the number of iterations:
gedit system/controlDict
- Run the case:
simpleFoam
- And process the results:
paraFoam

```
16 // * * * * *
17
18 application      simpleFoam;
19
20 startFrom        startTime;
21
22 startTime        0;
23
24 stopAt           endTime;
25
26 endTime          1000;
27
28 deltaT           1;
29
30 writeControl      timeStep;
31
32 writeInterval     5;
33
34 purgeWrite        0;
35
36 writeFormat       ascii;
37
38 writePrecision    6;
39
40 writeCompression  off;
41
42 timeFormat         general;
43
44 timePrecision     6;
45
46 runTimeModifiable true;
47
48 // *****
```

Ex. 5: Turbulent pipe – Postprocessing

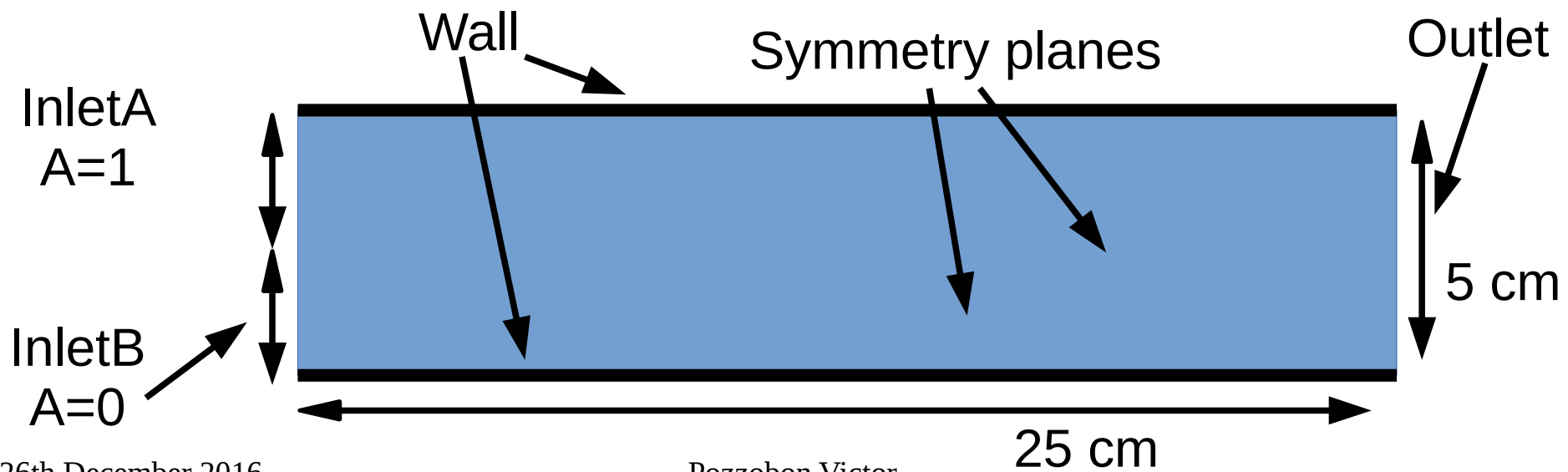


Ex. 6: Turbulent mixing length – Case setup

- Solving incompressible turbulent flow in a 2D plane with scalar transport, in steady state

$$\frac{d\bar{U}}{dt} + \bar{U} \nabla \cdot \bar{U} = -\nabla \frac{\bar{p}}{\rho} + \eta \nabla^2 \bar{U} + \text{Turbulence}$$

$$\frac{d\bar{A}}{dt} + \bar{U} \nabla \cdot \bar{A} = \alpha_t \nabla^2 \bar{A} \quad Sc_t = \frac{\nu_t}{\alpha_t}$$

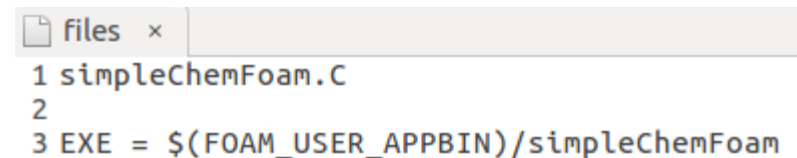


Ex. 6: Turbulent mixing length – Solver creation

- Reach 'run' directory:
`run`
- Move to solver directory:
`cd solvers`
- Copy simpleFoam:
`cp -r $FOAM_APP/solvers/incompressible/simpleFoam .`
`mv simpleFoam simpleChemFoam`
- Move to the new solver directory:
`cd simpleChemFoam`

Ex. 6: Turbulent mixing length – Solver creation / modification

- Clean the directory:
`rm -r porousSimpleFoam/ SRFSimpleFoam/ simpleFoam.dep`
- Clean the directory:
`wclean`
- Rename simpleFoam:
`mv simpleFoam.C simpleChemFoam.C`
- Change compilation file:
`gedit Make/files`



```
files x
1 simpleChemFoam.C
2
3 EXE = $(FOAM_USER_APPBIN)/simpleChemFoam
```

Ex. 6: Turbulent mixing length – Solver modification / createFields.H

- Add A field and turbulent Schmidt number to createFields.H:
gedit createFields.H

```
createFields.H x
1 Info<< "Reading field p\n" << endl;
2 volScalarField p
3 (
4     IOobject
5     (
6         "p",
7         runtime.timeName(),
8         mesh,
9         IOobject::MUST_READ,
10        IOobject::AUTO_WRITE
11    ),
12    mesh
13 );
14
15 Info<< "Reading field A\n" << endl;
16 volScalarField A
17 (
18     IOobject
19     (
20         "A",
21         runtime.timeName(),
22         mesh,
23         IOobject::MUST_READ,
24         IOobject::AUTO_WRITE
25     ),
26     mesh
27 );
```

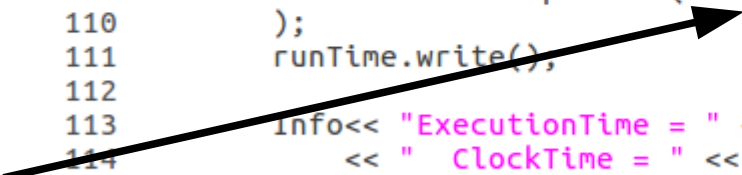
```
29 Info<< "Reading field U\n" << endl;
30 volVectorField U
31 (
32     IOobject
33     (
34         "U",
35         runtime.timeName(),
36         mesh,
37         IOobject::MUST_READ,
38         IOobject::AUTO_WRITE
39     ),
40     mesh
41 );
42
43 #include "createPhi.H"
44
45
46 label pRefCell = 0;
47 scalar pRefValue = 0.0;
48 setRefCell(p, simple.dict(), pRefCell, pRefValue);
49 mesh.setFluxRequired(p.name());
50
51
52 singlePhaseTransportModel laminarTransport(U, phi);
53
54 autoPtr<incompressible::turbulenceModel> turbulence
55 (
56     incompressible::turbulenceModel::New(U, phi, laminarTransport)
57 );
58
59 IOdictionary transportProperties
60 (
61     IOobject
62     (
63         "transportProperties",
64         runtime.constant(),
65         mesh,
66         IOobject::MUST_READ_IF_MODIFIED,
67         IOobject::NO_WRITE
68     )
69 );
70
71
72 Info<< "Reading diffusivity Sct\n" << endl;
73
74 dimensionedScalar Sct
75 (
76     transportProperties.lookup("Sct")
77 );
```

Ex. 6: Turbulent mixing length – modifying solver

- Add the scalar transport equation:
gedit
simpleChemFoam.C

Accessing turbulent viscosity

```
91  while (simple.loop())
92  {
93      Info<< "Time = " << runTime.timeName() << nl << endl;
94
95      // --- Pressure-velocity SIMPLE corrector
96      {
97          #include "UEqn.H"
98          #include "pEqn.H"
99      }
100
101      laminarTransport.correct();
102      turbulence->correct();
103
104      // --- Scalar A field
105      solve
106      (
107          fvm::ddt(A)
108          + fvm::div(phi, A)
109          - fvm::laplacian(turbulence->nut() / Sct, A)
110      );
111      runTime.write();
112
113      Info<< "ExecutionTime = " << runTime.elapsedCpuTime()
114          << " ClockTime = " << runTime.elapsedClockTime()
115          << nl << endl;
116  }
117
118  Info<< "End\n" << endl;
119
120  return 0;
121 }
```



Ex. 6: Turbulent mixing length – Solver compilation / case creation

- Clean the directory and compile:
`wclean; wmake`
- Move to cases directory:
`run`
- Copy Ex5 directory:
`cp -r Ex5 Ex6`
- Move to the case directory:
`cd Ex6`

Ex. 6: Turbulent mixing length – Create case directory

- Clean the case directory:
foamListTimes -rm
- Modify physical properties:
gedit constant/transportProperties

```
8 FoamFile
9 {
10     version      2.0;
11     format        ascii;
12     class         dictionary;
13     location      "constant";
14     object        transportProperties;
15 }
16 // * * * * *
17
18 transportModel    Newtonian;
19
20 nu                nu [0 2 -1 0 0 0 0] 1e-05;
21 Sct                Sct [0 0 0 0 0 0] 1;
22
23 // *****
```

Ex. 6: Turbulent mixing length – Create mesh

- Modify the mesh: gedit system/blochMeshDict

```
17 convertToMeters 0.05;
18
19 vertices
20 (
21     (0 0 0) //0
22     (1 0 0)
23     (1 1 0)
24     (0 1 0)
25     (0 0 5) //4
26     (1 0 5)
27     (1 1 5)
28     (0 1 5) //7
29 );
30
31 blocks
32 (
33     hex (0 1 2 3 4 5 6 7) (80 1 250)
34         simpleGrading
35         (
36             (
37                 (0.1 0.2 4) // 10% y-dir, 20% cells, expansion = 466
38                 (0.8 0.6 1) // 80% y-dir, 60% cells, expansion = 167
39                 (0.1 0.2 0.25) // 10% y-dir, 20% cells, expansion = 0.25 (1/4)
40             )
41             1 // y-direction expansion ratio
42             1 // z-direction expansion ratio
43         )
44 );
```

Mesh refinement on
the two sides of the
x axis



```
46 edges
47 (
48 );
49
50 boundary
51 (
52     inletA
53     {
54         type patch;
55         faces
56         (
57             (0 1 2 3)
58         );
59     }
60     outlet
61     {
62         type patch;
63         faces
64         (
65             (4 5 6 7)
66         );
67     }
68     wall
69     {
70         type wall;
71         faces
72         (
73             (1 2 6 5)
74             (0 3 7 4)
75         );
76     }
77     symmetryWall
78     {
79         type symmetry;
80         faces
81         (
82             (0 4 5 1)
83             (3 2 6 7)
84         );
85     }
86 );
87
88 mergePatchPairs
89 (
90 );
```


Ex. 6: Turbulent mixing length – Modifying the mesh

- We are now going to create InletB patch using topoSetDict and createPatchDict:
gedit
system/topoSetDict

```
topoSetDict x
1  /*-----*-- C++ --*-----*/
2  |=====|
3  | \ \ \ \ | F i e l d | OpenFOAM: The Open Source
4  | \ \ \ \ | O p e r a t i o n | Version: 3.0.1
5  | \ \ \ \ | A n d | Web: www.OpenFOAM.org
6  | \ \ \ \ | M a n i p u l a t i o n |
7  |-----*-----|
8  FoamFile
9  {
10     version      2.0;
11     format        ascii;
12     class         dictionary;
13     location      "system";
14     object        topoSetDict;
15 }
16 // *****
17
18 actions
19 (
20     // Grabbing faces
21     {
22         name      faceGrabbed;
23         type      faceSet;
24         action     new;
25         source     boxToFace;
26         sourceInfo
27         {
28             box (0.00 0.0 -0.0001) (0.025 0.05 0.0001);
29         }
30     }
31 );
```

Ex. 6: Turbulent mixing length – Modifying the mesh

- Then createPatchDict:
gedit
system/createPatchDict

```
8 FoamFile
9 {
10     version      2.0;
11     format        ascii;
12     class         dictionary;
13     object        createPatchDict;
14 }
15
16 // * * * * *
17
18 pointSync false;
19
20 // Patches to create.
21 patches
22 (
23     {
24         // Name of new patch
25         name inletB;
26
27         // Type of new patch
28         patchInfo
29         {
30             type patch;
31         }
32
33         // How to construct: either from 'patches' or 'set'
34         constructFrom set;
35
36         // If constructFrom = set : name of faceSet
37         set faceGrabbed;
38     }
39 );
```

Ex. 6: Turbulent mixing length – Modifying the mesh

- Then, select the inletB faces:
topoSet
- Create the new patch:
createPatch -overwrite

Ex. 6: Turbulent mixing length – Modifying the solver

- Modify the numerical schemes to account for A specie transport:
gedit system/fvSchemes

```
8 FoamFile
9 {
10     version      2.0;
11     format        ascii;
12     class         dictionary;
13     location      "system";
14     object        fvSchemes;
15 }
16 // * * * * *
17
18 ddtSchemes
19 {
20     default       steadyState;
21 }
22
23 gradSchemes
24 {
25     default       Gauss linear;
26 }
27
28 divSchemes
29 {
30     default       none;
31     div(phi,U)    bounded Gauss linearUpwind grad(U);
32     div(phi,k)     bounded Gauss limitedLinear 1;
33     div(phi,epsilon) bounded Gauss limitedLinear 1;
34     div(phi,omega) bounded Gauss limitedLinear 1;
35     div(phi,v2)    bounded Gauss limitedLinear 1;
36     div((nuEff*dev2(T(grad(U))))) Gauss linear;
37     div(nonlinearStress) Gauss linear;
38     div(phi,A)     bounded Gauss linear;
39 }
```

Ex. 6: Turbulent mixing length – Modifying the solver

- Modify the solvers to account for A specie transport:
gedit system/fvSolution

```
18 solvers
19 {
20     p
21     {
22         solver          GAMG;
23         tolerance        1e-06;
24         relTol           0.1;
25         smoother         GaussSeidel;
26         nPreSweeps        0;
27         nPostSweeps       2;
28         cacheAgglomeration on;
29         agglomerator      faceAreaPair;
30         nCellsInCoarsestLevel 10;
31         mergeLevels       1;
32     }
33
34     "(U|k|epsilon|omega|f|v2)"
35     {
36         solver           smoothSolver;
37         smoother          symGaussSeidel;
38         tolerance         1e-05;
39         relTol             0.1;
40     }
41
42     A
43     {
44         solver            BICCG;
45         preconditioner     DILU;
46         tolerance         1e-05;
47         relTol             0.1;
48     }
49 }
```

Ex. 6: Turbulent mixing length – Modifying the solver

- Carry on ...

```
51 SIMPLE
52 {
53     nNonOrthogonalCorrectors 0;
54     consistent yes;
55
56     residualControl
57     {
58         p 1e-2;
59         U 1e-3;
60         "(k|epsilon|omega|f|v2|A)" 1e-3;
61     }
62 }
63
64 relaxationFactors
65 {
66     equations
67     {
68         U 0.9; // 0.9 is more stable but 0.95 more convergent
69         ".*" 0.9; // 0.9 is more stable but 0.95 more convergent
70     }
71 }
```

Ex. 6: Turbulent mixing length – IC / BC

- Modify velocity boundary conditions:
gedit 0/U

```
8 FoamFile
9 {
10     version      2.0;
11     format        ascii;
12     class         volVectorField;
13     object        U;
14 }
15 // * * * * *
16
17 dimensions      [0 1 -1 0 0 0 0];
18
19 internalField    uniform (0 0 0);
20
21 boundaryField
22 {
23     "(inlet.*)"
24     {
25         type      fixedValue;
26         value      uniform (0 0 1);
27     }
28
29     outlet
30     {
31         type      zeroGradient;
32     }
33
34     wall
35     {
36         type      fixedValue;
37         value      uniform (0 0 0);
38     }
39
40     symmetryWall
41     {
42         type      symmetry;
43     }
44 }
```

Ex. 6: Turbulent mixing length – IC / BC

- Modify pressure boundary conditions:
gedit 0/p

```
8 FoamFile
9 {
10     version      2.0;
11     format        ascii;
12     class         volScalarField;
13     object        p;
14 }
15 // *****
16
17 dimensions       [0 2 -2 0 0 0 0];
18
19 internalField     uniform 0;
20
21 boundaryField
22 {
23     "(inlet.*)"
24     {
25         type      zeroGradient;
26     }
27
28     outlet
29     {
30         type      fixedValue;
31         value      uniform 0;
32     }
33
34     wall
35     {
36         type      zeroGradient;
37     }
38
39     symmetryWall
40     {
41         type      symmetry;
42     }
43 }
```


Ex. 6: Turbulent mixing length – IC / BC

- Modify dissipation boundary conditions:
gedit 0/epsilon

```
8 FoamFile
9 {
10     version      2.0;
11     format        ascii;
12     class         volScalarField;
13     location      "0";
14     object        epsilon;
15 }
16 // * * * * *
17
18 dimensions      [0 2 -3 0 0 0 0];
19
20 internalField    uniform 2.04E-01;
21
22 boundaryField
23 {
24     "(inlet.*)"
25     {
26         type      fixedValue;
27         value      uniform 7.55E-03;
28     }
29
30     outlet
31     {
32         type      zeroGradient;
33     }
34
35     wall
36     {
37         type      epsilonWallFunction;
38         value      uniform 2.04E-01;
39     }
40
41     symmetryWall
42     {
43         type      symmetry;
44     }
45 }
```

Ex. 6: Turbulent mixing length – IC / BC

- Modify turbulent kinetic energy boundary conditions:
gedit 0/k

```
8 FoamFile
9 {
10     version      2.0;
11     format        ascii;
12     class         volScalarField;
13     object        k;
14 }
15 // * * * * *
16
17 dimensions       [0 2 -2 0 0 0 0];
18
19 internalField     uniform 3.38E-02;
20
21 boundaryField
22 {
23     "(inlet.*)"
24     {
25         type       fixedValue;
26         value       uniform 3.75E-03;
27     }
28
29     outlet
30     {
31         type       zeroGradient;
32     }
33
34     wall
35     {
36         type       zeroGradient;
37     }
38
39     symmetryWall
40     {
41         type       symmetry;
42     }
43 }
```

Ex. 6: Turbulent mixing length – IC / BC

- Modify turbulent viscosity boundary conditions:
gedit 0/nut

```
8 FoamFile
9 {
10     version      2.0;
11     format        ascii;
12     class         volScalarField;
13     location      "0";
14     object        nut;
15 }
16 // * * * * *
17
18 dimensions       [0 2 -1 0 0 0 0];
19
20 internalField     uniform 0;
21
22 boundaryField
23 {
24     "(inlet.*)"
25     {
26         type       calculated;
27         value       uniform 0;
28     }
29
30     outlet
31     {
32         type       calculated;
33         value       uniform 0;
34     }
35
36     wall
37     {
38         type       calculated;
39         value       uniform 0;
40     }
41
42     symmetryWall
43     {
44         type       symmetry;
45     }
46 }
```

Ex. 6: Turbulent mixing length – IC / BC

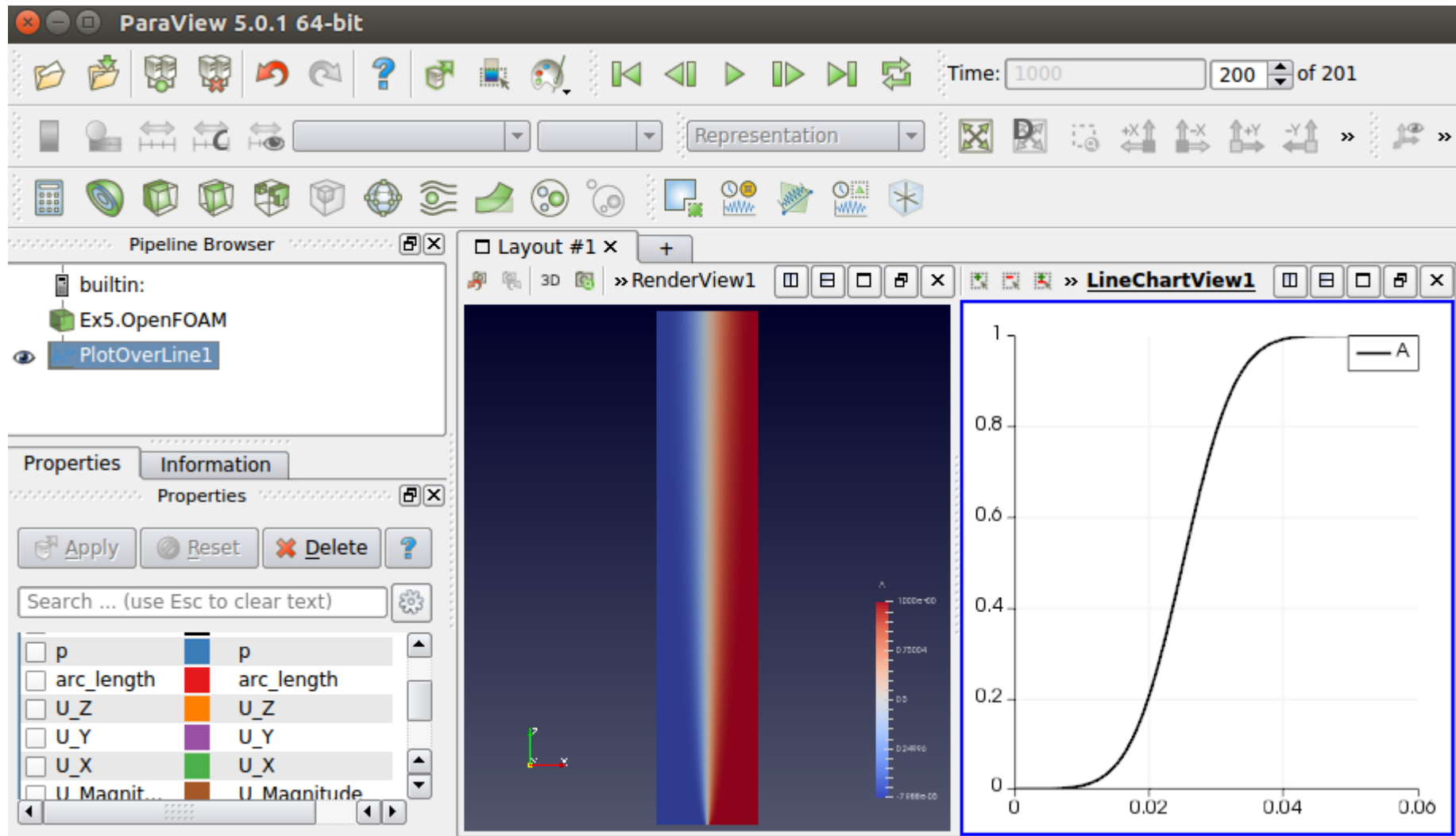
- Modify A field boundary conditions:
gedit 0/A

```
8 FoamFile
9 {
10     version      2.0;
11     format        ascii;
12     class         volScalarField;
13     object        A;
14 }
15 // *****
16
17 dimensions      [1 -3 0 0 0 0 0];
18
19 internalField    uniform 0;
20
21 boundaryField
22 {
23     inletA
24     {
25         type      fixedValue;
26         value      uniform 1;
27     }
28     inletB
29     {
30         type      fixedValue;
31         value      uniform 0;
32     }
33     outlet
34     {
35         type      zeroGradient;
36     }
37
38     wall
39     {
40         type      zeroGradient;
41     }
42
43     symmetryWall
44     {
45         type      symmetry;
46     }
47 }
```

Ex. 6: Turbulent mixing length – Running the case

- Run the case:
`simpleChemFoam`
- And process the results:
`paraFoam`

Ex. 6: Turbulent mixing length – Postprocessing



It's over

- This tutorial is over, thank you for your attention
- I hope you enjoyed it
- Please feel free to contact me:
victor.pozzobon@centralesupelec.fr

The extra mile

- The open source software, I use to draw and mesh complex geometries:
SALOME: www.salome-platform.org
- Another open source software which can be used to process high volume results:
Visit: <https://visit.llnl.gov>
- Where I ask for help:
CFD Online: www.cfd-online.com/Forums/openfoam

It's over

Again, thank you for your attention.