Basic Courses (B1)

INTRODUCTION TO OPENFOAM

open Field Operation And Manipulation C++ libraries

INTRODUCTION TO OPENFOAM

open Field Operation And Manipulation C++ libraries

Rita F. Carvalho, MARE, Department of Civil Engineering, University of Coimbra, Portugal

UNIVERSIDADE D COIMBRA

FOAM@IBERIA 2019

MARE

OpenFOAM – what it is

SOLVE PARTIAL DIFFERENTIAL EQUATIONS (PDE)

RESOLUTION OF CURRENT "**ENGINEERING**" CHALLENGES

USE OF ADVANCED **TECHNOLOGY**

OpenFOAM – what it is

OpenFOAM is a powerful tool that allows the
➔ numerical solution of differential equations
➔ **easily code specific problem based on differential equations** (continuum mechanics)

OpenFOAM is a software toolbox licensed under the GNU General Public License trusted by many thousands of engineers and scientists in industry and academia worldwide.

OpenFOAM is an open source **C++ library**, prepared to **simple or parallel computing** to **create executable applications** (solvers & utilities) in
solvers designed to solve a **specific problem** in continuum mechanics
utilities designed to **perform tasks** that involve data manipulation.

OPENSOURCE, FREE, OPEN CODE, CONTINUOUS EVOLUTION

OpenFOAM – how can be used

- Users can **use** and **write simple to complex solver** with only few lines (depending of the knowledge in physics and programming techniques)

- Users can change the **existing solvers**, and, use them as the start point for the creation of a new solver - The **complete source of the code is available**!

- Users can profit from the numeric tools to solve EDPs, laplacian, gradient, divergent ...

$\neq$ Terms

$\neq$ Equations ➔ $\neq$ solvers

$\neq$ Set of Equations

OpenFOAM applications

Multi disciplinar

Mechanics of Fluids, Turbulence, Heat Transfers, Chemical Reactions, Electromagnetism, Financial

Besides…

➔ in hydraulic and CFD Fields …

- Aero and hydro dynamics of objects (vehicles, greenhouses, wind towers);
- Dimensioning of hydraulic structures;
- Optimization and design of river and maritime structures;
- Dimensioning of heat transfer devices;
- Influence of wind on structures;
- Propagation of polluting feathers;
- Simulation of sediment transport and location of erosion zones.

Different versions of OpenFOAM

http://openfoamwiki.net/index.php/Forks_and_Variants#Definitions

- **Fork**

The idea behind *forking* is that when **several changes to the code are made**, the distribution of the modified group of changes should use a **different name from the original project**, to avoid confusion, and should follow the policies and guidelines mentioned above. Fork (software development)

- **Variant**

This is the denomination mostly used when the **changes to the complete source code package don't seem substantial enough** to require a fully dedicated source code branch for supporting the changes made. This usually occurs for situations where the changes are done only a few times for a particular version of a software or even when the changes are only provided as *patch* files. Nonetheless, these variants must still abide to the policies and guidelines mentioned above.

Different versions of OpenFOAM

FORKS - number of different flavours of OpenFOAM that are all based on the original FOAM-code.

- **OpenFOAM (Foundation) OpenFOAM by the OpenFOAM foundation**
- **OpenFOAM+ OpenFOAM+ by the ESI-OpenCFD**
- **Foam-extend Foam-extend has a number of additional community-contributed features.**
- **Other**
    - Caelus-CML - version 5.10 (October 2015), it's officially a fork of OpenFOAM 2.1.1 - Hybrid commercial model, made available freely to the public.
    - ENGYS' own builds of OpenFOAM - Available for customers only/ Strictly commercial.
    - FreeFOAM - Completely open to the general public. Currently distributed with Debia
    - iconCFD - Strictly commercial.
    - RapidCFD and RapidCFD @ Sim-Flow- fork of OpenFOAM that has been specifically designed to be built with CUDA. Current version is a fork from OpenFOAM 2.3.1
    - RheologicRemix

Different versions of OpenFOAM

VERSIONS

- Latest **OpenFOAM** version: 6 (10th of July 2018)
- Latest Foam-Extend version: 4.0 (22nd of December 2016)
- Latest **OpenFOAM+** version: v1812 (20th December 2018)

The OpenFOAM Foundation

The OpenFOAM® Extend Project

OpenFOAM

- OpenFOAM Release Notes: **6** | **5.0** | 4.1 | **4.0** | 3.0.1 | 3.0.0 | 2.4.0| 2.3.1 | **2.3.0** | 2.2.2 | 2.2.1 | 2.2.0 | 2.1.1 | 2.1.0 | 2.0.1 | 2.0.0 | 1.7.1 | **1.7.0** | 1.6 | 1.5 | 1.4.1 | 1.4 | 1.3 | 1.2 | 1.1 | 1.0
- Foam-extend Release Notes: 4.0 | 3.2 | 3.1 | 3.0
- OpenFOAM+ Release Notes: **v1812** | v1806 | v1712 | v1706 | v1612+ | v1606+ | v3.0+

http://openfoamwiki.net/index.php/Main_Page

Different versions of OpenFOAM - Installation

OpenFOAM

- Linux - Ubuntu
- [Windows]
- [Mac OS]

Ubuntu Linux

Packaged installation for Ubuntu Linux **released on 10th July 2018**. A simple installation option for OpenFOAM, native to Ubuntu Linux.

Install on Ubuntu

Other Linux

Run on other Linux distributions, including RHEL, CentOS, Fedora, SLES and openSuSE, using Docker. Another simple installation option.

Install on Other Linux

macOS

Run on macOS Desktop v10.10.3+, using the Docker container system. Another simple installation option.

Install on macOS

Run on Windows 10

OpenFOAM packages for Ubuntu can be installed directly on Microsoft Windows 10 using Bash on Ubuntu on Windows

OpenFOAM on Windows 10

Run on a Virtual Machine (VM)

OpenFOAM is created for the GNU/Linux operating system and can run on the Windows operating system using virtualization.

OpenFOAM on a Virtual Machine

blueCFD®-Core

Public, open source software from blueCAPE Lda, providing OpenFOAM for updated Windows 64 bit OS, versions 7 to 10.

Download blueCFD-Core

OpenFOAM+ version: v1812

Installation

Instructions for installing OpenFOAM can be found below:

- Linux binary
- Mac binary
- Windows binary
- Windows 10 source and binary (native)
- Source

OpenFOAM - Structure

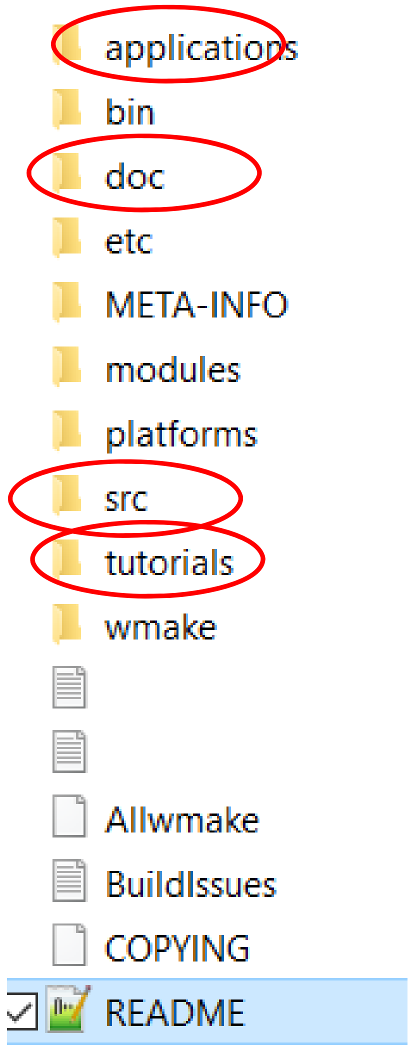
The OpenFOAM source code comprises of **four main components**:

src: the core OpenFOAM source code;

applications: collections of library functionality wrapped up into applications, such as solvers and utilities;

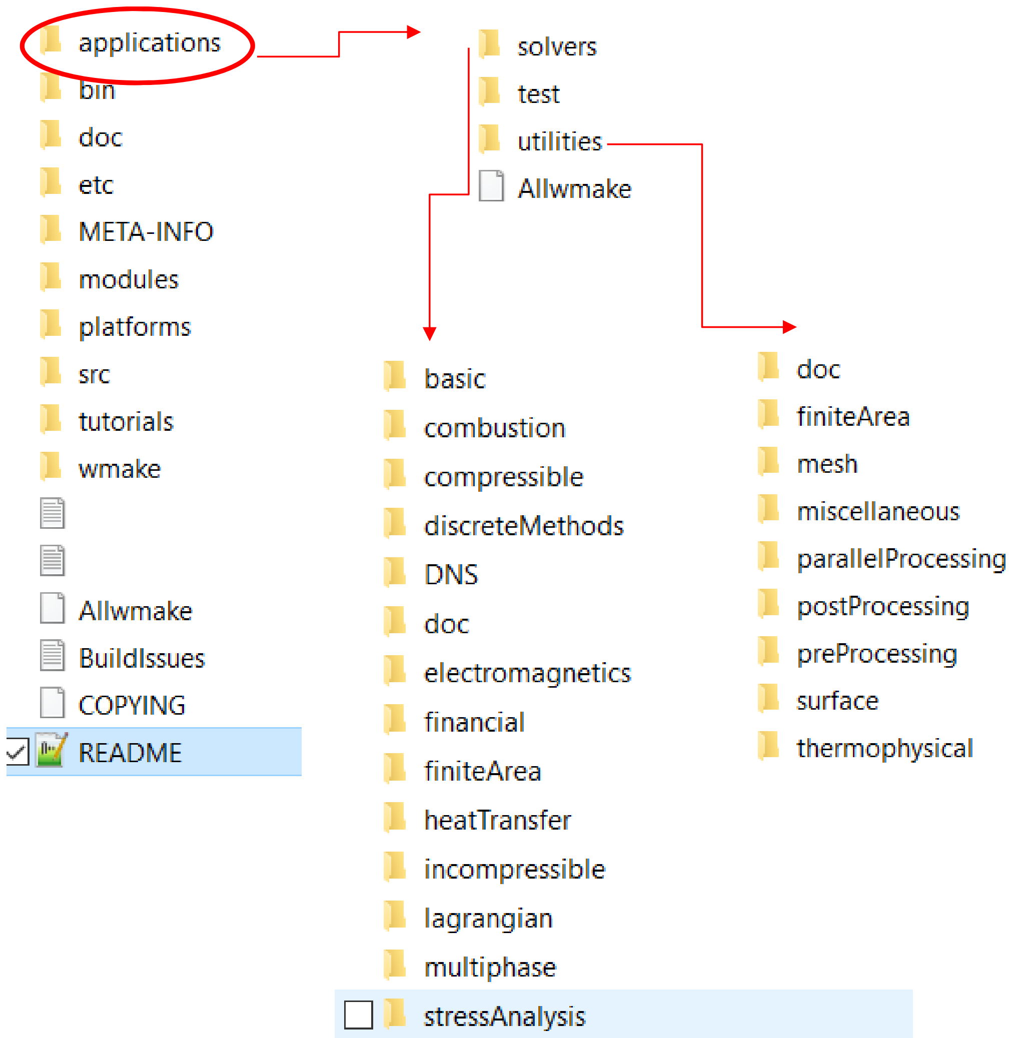
tutorials: a suite of test cases that highlight a broad cross-section of OpenFOAM's capabilities;

doc: supporting documentation



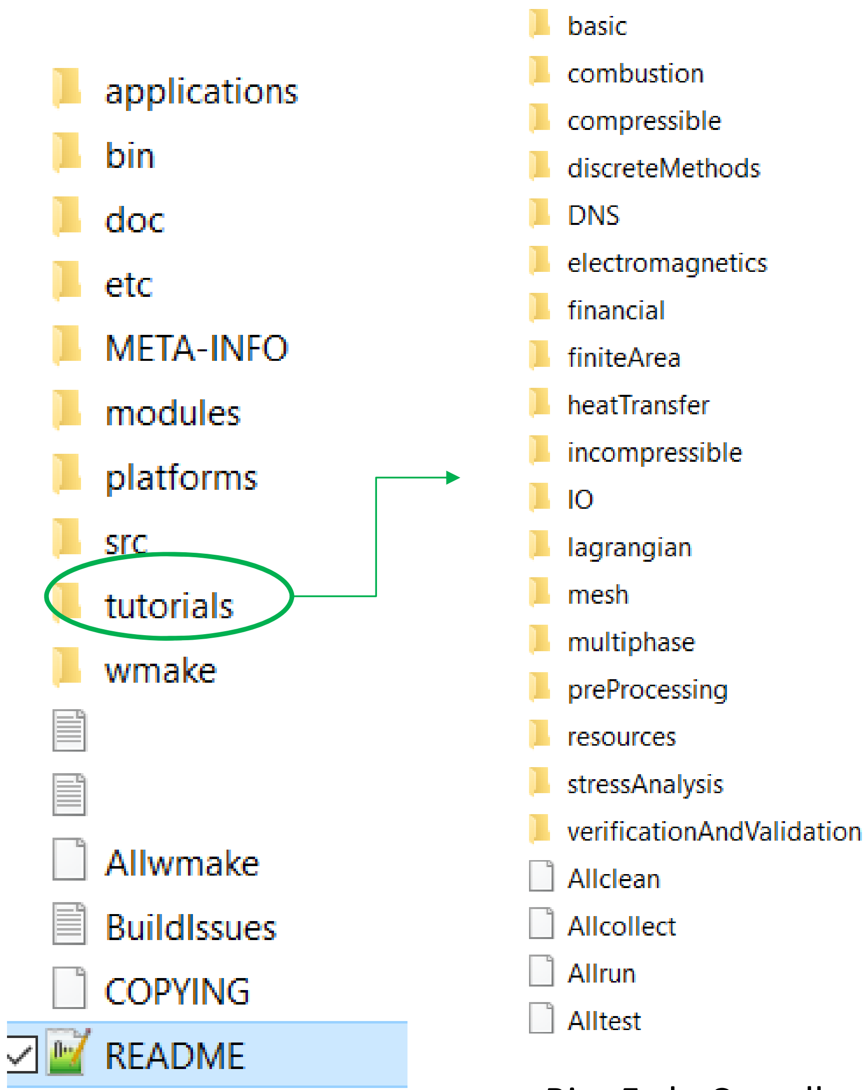
OpenFOAM - Structure

The OpenFOAM source code comprises of **four main components**:
src: the core OpenFOAM source code;
applications: collections of library functionality wrapped up into applications, such as solvers and utilities;
tutorials: a suite of test cases that highlight a broad cross-section of OpenFOAM's capabilities;
doc: supporting documentation



OpenFOAM - Structure

The OpenFOAM source code comprises of **four main components**:
src: the core OpenFOAM source code;
applications: collections of library functionality wrapped up into applications, such as solvers and utilities;
tutorials: a suite of test cases that highlight a broad cross-section of OpenFOAM's capabilities;
doc: supporting documentation



Rita F. de Carvalho

OpenFOAM – Installation and use

Users want

A "black box" software that they can use with **complete confidence** for general problem classes without having to understand the fine algorithmic details.

to be able to **tune** data structures for a particular application, even if the software is not as reliable as that provided for general methods

to **use a large body of numerical software** freely available 24 hours a day via electronic services - there are already dozens of libraries, technical reports on various parallel computers and software, test data, facilities to automatically translate FORTRAN programs to C, bibliographies, names and addresses of scientists and mathematicians, and so on.

Users should

Know what they are using, what is behind software

Be sure that they can have **confidence** in models

Look at data, analyse it and be able to **decide if it is trustable**

Have an idea of what kind of **uncertainty** is possible in data

Understand why so different models, solvers, …

OpenFOAM - Notes

OpenFOAM contains a suite of **numerical tools** to solve a range of problems. To solve equations for a **continuum**, OpenFOAM uses a numerical approach with the following features:

1. **Segregated, iterative solution**:
system of equations governing our problem of interest ➔
separate matrix equations are created for **each equation**, and are solved within an **iterative sequence** (as opposed to created one, big matrix equation for the entire system of equations).

2. **Finite volume method**: Matrix equations are constructed using the finite volume method applied to arbitrary shaped cells (any number of faces, any number of edges).

OpenFOAM - Notes

Physical approximations

Partial differential Equations, Initial and Boundary Conditions 3

CFD - Mass, momentum and energy conservation, inlet, outlet, walls,…

Depending of PDE – Physical phenomenon

Calculate **Position/Time;**

Variables: u, p, T, φ; **Properties**: ρ, μ, k, σ, Γ, c ➔ **definition** 2

Numerical solution techniques - Consideration of continuous material/domain represented by discrete particles - **Finite volumes**, finite elements ➔ **Geometry and Meshing** - acceptable element sizes and shapes 1

➔ accurate numerical approximations - Implicit Methods, precision, diffusion, relaxation and accuracy➔ **options** 4

CFD -Accurate treatment of momentum – advection term, coupling

OpenFOAM - Notes

Differential Equations

$$\frac{dy}{dt} = k\frac{y}{1+t}$$

$$\frac{dy}{dt} = -\frac{t}{y}$$

$$\frac{d^2y}{dt^2} = \frac{g}{l}siny$$

$$m\frac{d^2y}{dt^2} = F(y,t)$$

$$\frac{d^2y}{dx^2} = 9y$$

$$\frac{d^2y}{dt^2} = -b\frac{dy}{dt} - cy + w(t)$$

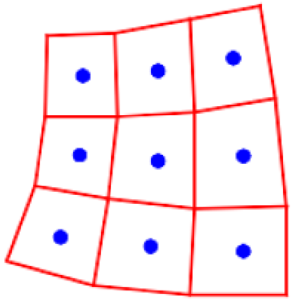
$$\frac{d^2y}{dx^2} = -\frac{-Q(L-x)}{EI}$$

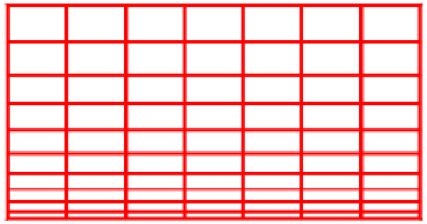
$$\frac{dT}{dt} = -k(T - T_a)$$

$$\frac{dV}{dt} = -A_0\sqrt{2gh}$$

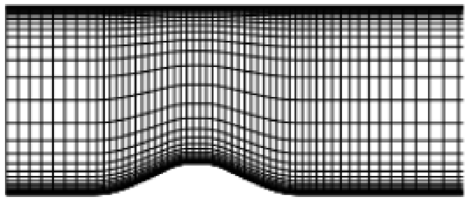
Involving:
Variables: dependent and indepedent, (x,t)
Properties

Domain / material represented as continuous and represented by discrete particles

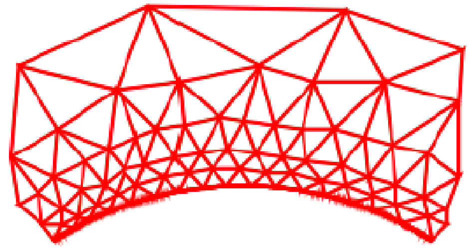




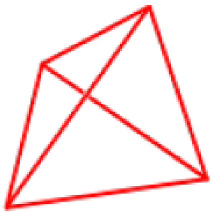
single-block structured Cartesian mesh



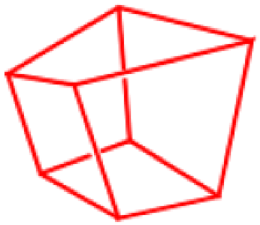
single-block structured curvilinear mesh



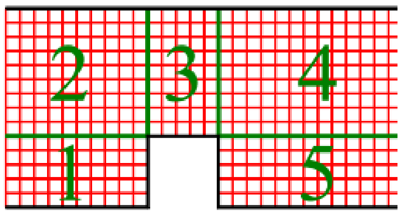
unstructured triangular mesh



tetrahedron



hexahedron



multi-block mesh

Rita F. de Carvalho

OpenFOAM - Notes

To solve equations for a continuum, OpenFOAM uses a numerical approach with the following features:

- **Parallel computation** are easy to perform (reduction of the computational time);
- You can create **simple meshes** with the mesh generator that comes with OpenFOAM. Also, you can convert to the OpenFOAM format, **meshes created with another software** (check the user guide for the available formats);
- It comes with **several utilities**;
- All components implemented in library form for easy re-use.

OpenFOAM - Notes

OpenFOAM contains a suite of **numerical tools** to solve a range of problems. It includes **methods to solve problems where matter is represented as a continuum and where it is represented by discrete particles**.
To solve equations for a continuum, OpenFOAM uses numerical approaches

CFD - We replace the actual molecular structure by a **hypothetical continuous medium**, which at a point has the mean properties of the molecules surrounding the point - "**fluid particle" – control *volume* – *imaginary* region** in which dynamic forces are in equilibrium - we look at the forces in external surface of the control volume ➔ integral form of equations/ differential.
➔ we choose the most convenient in each case

Rita F. de Carvalho

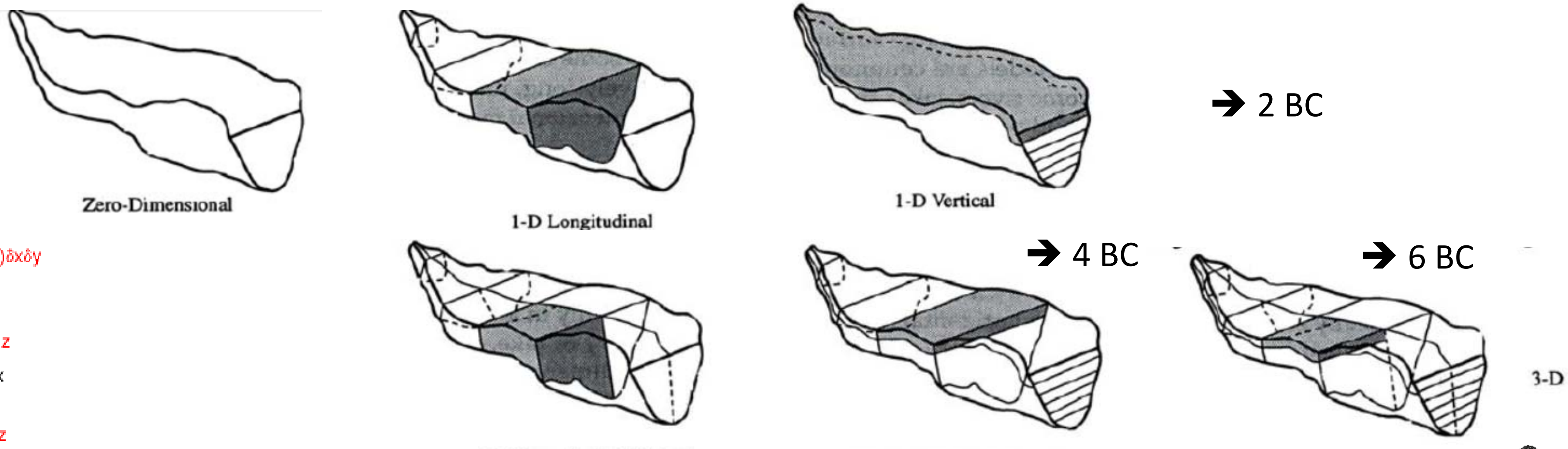
OpenFOAM - Notes

Differential Equations CFD

$$\boxed{\frac{1}{\rho}\frac{\partial \rho}{\partial t}+\frac{\partial u_i}{\partial x_i}=0}$$

$$\boxed{\rho\frac{\partial u_i}{\partial t}+\rho u_j\frac{\partial (u_i)}{\partial x_j}=\rho g_i-\frac{\partial p}{\partial x_i}+\frac{\partial \tau_{ji}}{\partial x_j}}$$

Domain / material represented as continuous and represented by discrete particles



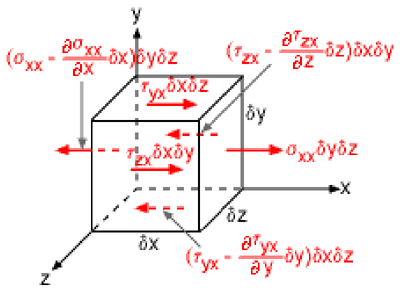
Zero-Dimensional

1-D Longitudinal

1-D Vertical

2-D Longitudinal-Lateral

2-D Longitudinal-Vertical

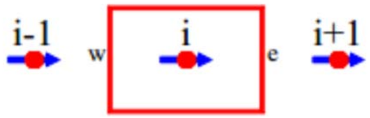


Rita F. de Carvalho
UNIVERSIDADE D COIMBRA
MARE

OpenFOAM - Notes

To solve equations for a continuum, OpenFOAM uses a numerical approach with the following features:

Collocated variables: The solution variable for each matrix equation is defined at cell centres.



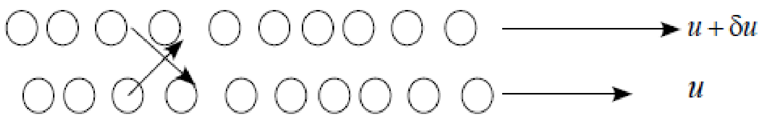
Different discretization schemes It uses second order schemes for the approximation of the different operators, but, many schemes are available, including high order schemes;

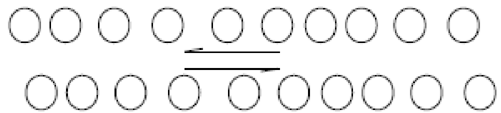
Equation coupling: The coupling between equations, particularly pressure and velocity is performed using adapted versions of well-known algorithms such as *e.g.* PISO and SIMPLE ➔ PIMPLE

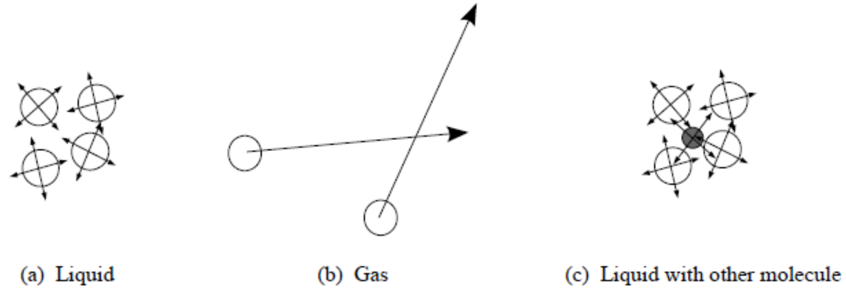
OpenFOAM - Notes

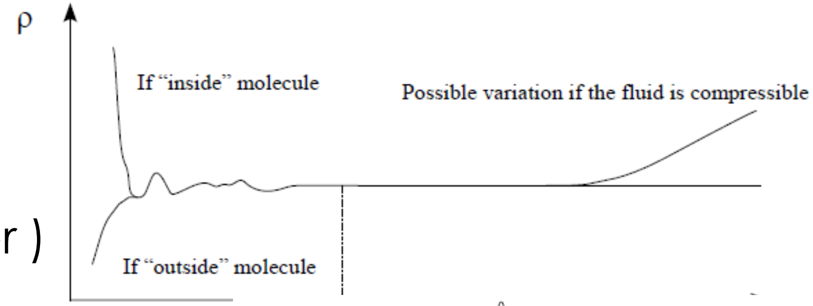
We assume the fluid is continuous, the continuum hypothesis represented by discrete volumes with average **Properties**

Density
Surface Tension
Bulk modulus / Compressibility
Diffusivity coefficient (spontaneous spreading of matter)
Viscosity (random motion comes from the molecular natures of the constituents)









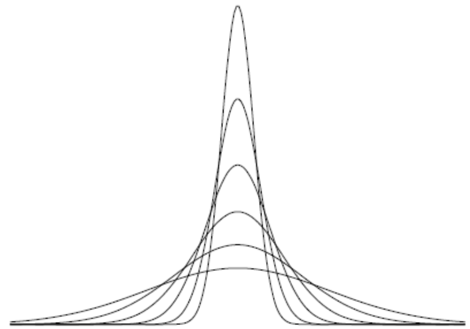


Figure 1-3. Typical behaviour of diffusion of an initial point concentration

OpenFOAM - Notes

General Equations – PDE - OpenFoam

•Mesh
•Properties
•Initial and Boundary Conditions
•Parameters, controls and options

Pre - processor

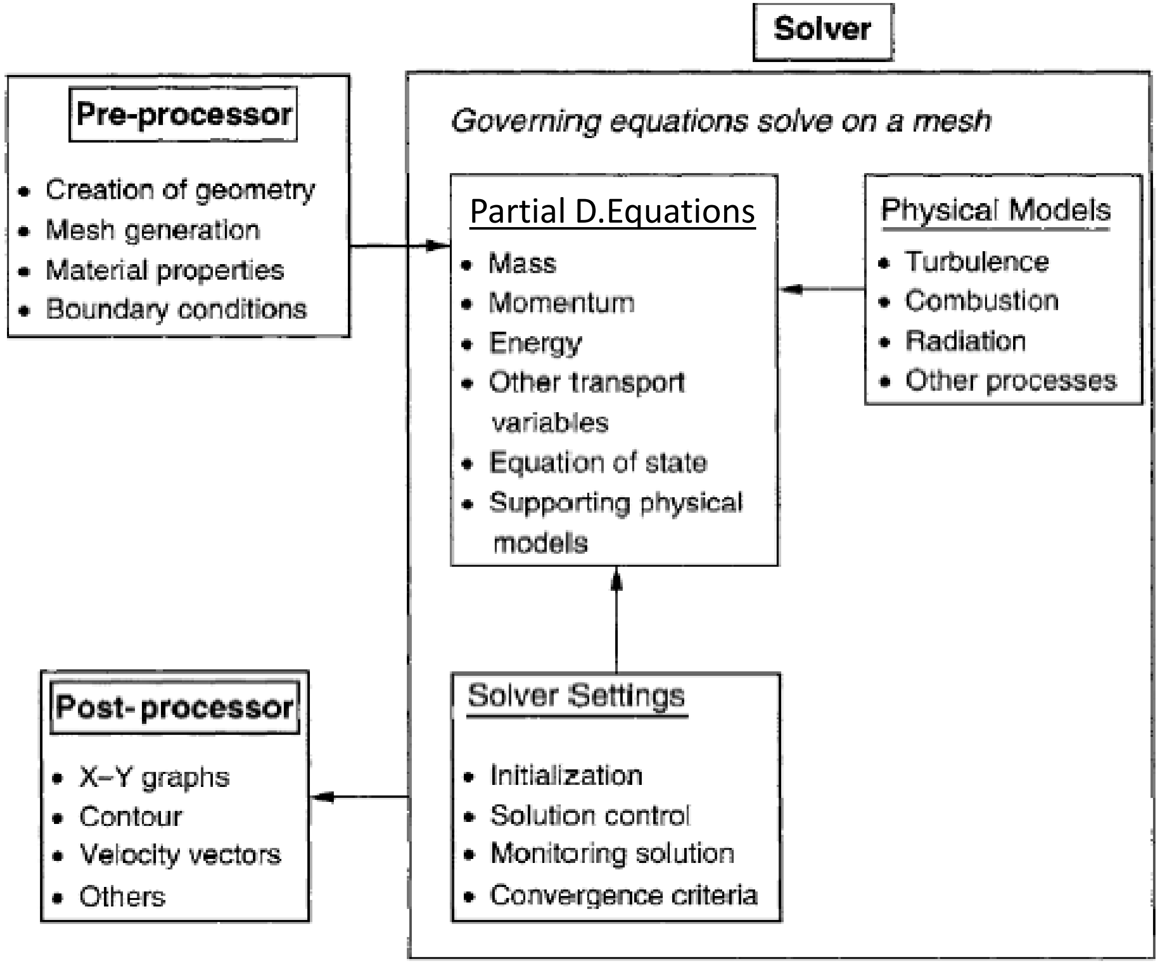
•Solve the equation
Gradients/divergents/ laplacians/interpolation

Solver

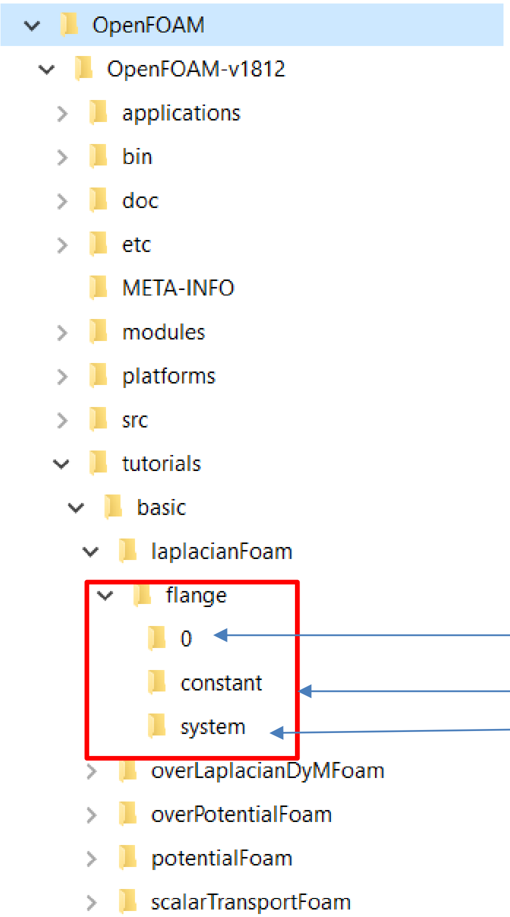
•Results analysis

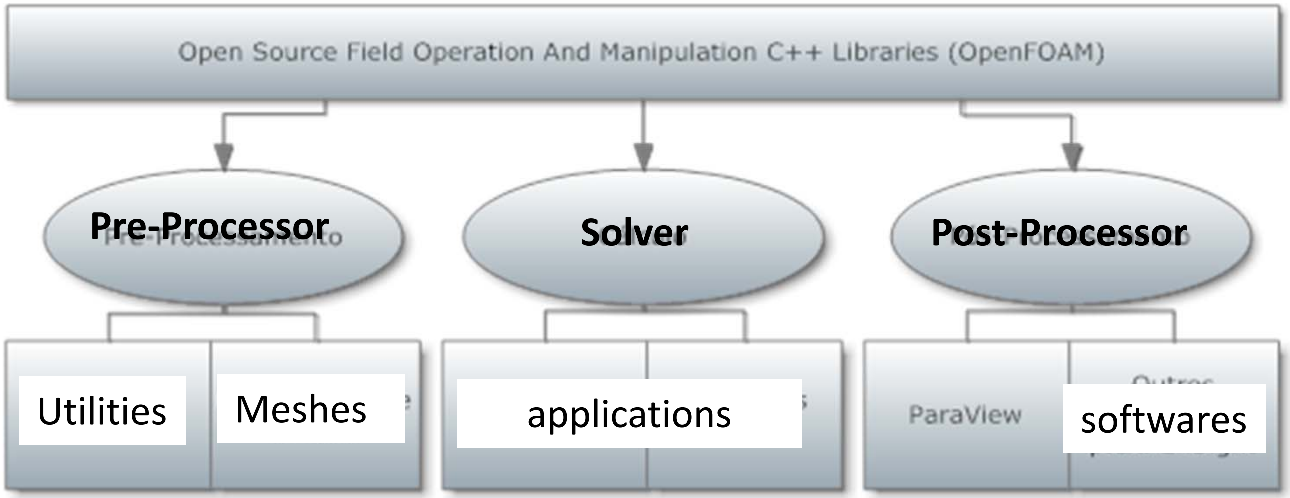
Post- processor

http://www.cfd-online.com/
http://www.openfoam.com/

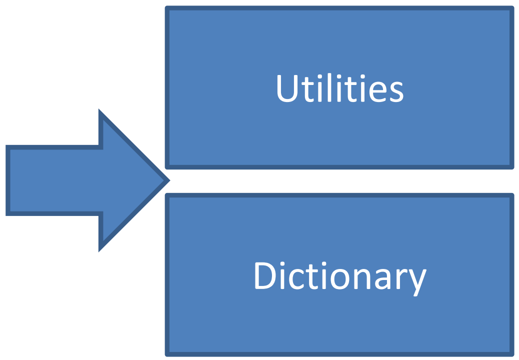


OpenFOAM - Use





- **Pre-Processor**
  - Mesh
  - IC + BC
  - Properties & parameters
- **Solver**
- **Post-Processor**



Rita F. de Carvalho

OpenFOAM - Equations

Equations in a solver

$$\frac{\partial \rho U}{\partial t} + \nabla \cdot \phi U - \nabla \cdot \mu \nabla U = -\nabla p$$

Ex: Solve (

```
        fvm::ddt(rho, U)
      + fvm::div(phi, U)
      - fvm::laplacian(miu, U)
   == - fvc::grad(p)
        );
```

Divergent vector → scalar

$$\nabla . V = \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z}$$

Laplacian vector (gradient of a divergent) → vector

$$\nabla^2 V = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} + \frac{\partial^2 w}{\partial z^2}$$

Gradient scalar → vector

$$\nabla p = \frac{\partial p}{\partial x}\hat{i} + \frac{\partial p}{\partial y}\hat{j} + \frac{\partial p}{\partial z}\hat{k}$$

it is fundamentally a tool for solving partial differential equations rather than a CFD package in the traditional sense

OpenFOAM - Solvers

Laplacian vector (gradient of a divergent) ➔ vector

$$\nabla^2 V = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} + \frac{\partial^2 w}{\partial z^2}$$

Basic' solvers

laplacianFoam/ overLaplacianDyMFoam Solves a simple Laplace equation, e.g. for thermal diffusion in a solid

$$\boxed{\frac{\partial T}{\partial t} = \nabla \cdot (\alpha \nabla T)} + \frac{q}{\rho c_p}, \qquad \alpha = \frac{k}{\rho c_p}$$

$\boxed{\dot{T} = \alpha \nabla^2 T}$ thermal diffusivity α is represented by DT

```
fvScalarMatrix TEqn
(
    fvm::ddt(T) –

    fvm::laplacian(DT, T)
      ==
    fvOptions(T)
);
```

➔ Initial and BC

ddtSchemes laplacianSchemes

➔ fvSolution, fvSchemes

interpolationSchemes

gradSchemes, divSchemes, snGradSchemes

T[K] is the absolute temperature field, ρ[kg/m3] is the density field, q[W/m3] is the rate of energy generation per unit volume, α [m2/s] is the thermal diffusivity, k[W/(m·K)] is the thermal conductivity and c_p[J/(kg·K)] is the specific heat at constant pressure

Rita F. de Carvalho

OpenFOAM - Solvers

Laplacian vector (gradient of a divergent) ➔ vector

$$\nabla^2 V = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} + \frac{\partial^2 w}{\partial z^2}$$

Basic' solvers

laplacianFoam Solves a simple Laplace equation, e.g. for thermal diffusion in a solid

$$\boxed{\frac{\partial T}{\partial t} = \nabla \cdot (\alpha \nabla T)} + \frac{q}{\rho c_p}, \qquad \alpha = \frac{k}{\rho c_p}$$

- flange
  - 0
  - constant
  - system

Pre-Processor

T: internal and boundary field;

transportProperties: DT 4e-05;

ControlDict, fvSchemes, fvSolution

Mesh and Solver

ansysToFoam (create polyMesh in constant)... laplacianFoam

Post-Processor: paraView

ofv6:
. /opt/OpenFOAM/of6_envars.sh

v1812:
. /opt/foam/of+1812_envars.sh

Lookfor:

cd $FOAM_RUN/tutorials/
/laplacianFoam/flange

See: 0, constant, system

Rita F. de Carvalho
UNIVERSIDADE D COIMBRA

OpenFOAM - Notes

Basic' solvers

scalarTransportFoam Solves a transport equation for a passive scalar (scalar convection-diffusion problem on a given velocity field.)

ex:pitzDaily

$$\frac{\partial T}{\partial t} = -\nabla \cdot (\phi T) + \nabla \cdot \left(\alpha \nabla \ T\right) + S$$

```
solve ( fvm::ddt(T) +
fvm::div(phi, T) -
fvm::laplacian(DT, T)
);
```

IBC - Mandatory fields:

- U: velocity [m/s] = ϕ
- T: scalar [-]

Pre-Processor

0: U, T

constant: transportProperties

System: blockMeshDict, controlDict, fvSchemes, fvSolutiojn

Mesh

blockMesh (blockMeshDict**)**

Solver: scalarTransportFoam

Post-Processor: paraView

potentialFoam Simple potential flow solver which can be used to generate starting fields for full Navier-Stokes codes;

OpenFOAM - Notes

Solvers capabilities
Incompressible
Compressible
Multiphase
Combustion
DNS and LES
lagrangian
Particle-tracking flows
Conjugate Heat transfer …
Buoyancy-driven flows

Particle methods (DEM, **DSMC**, **MD**)
Other (**Solid dynamics**, **electromagnetics**)

Rita F. de Carvalho

OpenFOAM - Notes

Standard Solvers

Incompressible flow

- adjointShapeOptimizationFoam
- boundaryFoam
- channelFoam
- icoFoam
- MRFSimpleFoam
- nonNewtonianIcoFoam
- pimpleDyMFoam
- pimpleFoam
- pisoFoam
- porousSimpleFoam
- shallowWaterFoam
- simpleFoam
- SRFSimpleFoam
- windSimpleFoam

Compressible flow

- rhoCentralFoam
- rhoCentralDyMFoam
- rhoPimpleFoam
- rhoPorousMRFLTSPimpleFoam
- rhoPorousMRFSimpleFoam
- rhoPorousMRFPimpleFoam
- rhoSimplecFoam
- rhoSimpleFoam
- sonicDyMFoam
- sonicFoam
- sonicLiquidFoam

OpenFOAM - Notes

Standard Solvers

multiphase flow

bubbleFoam
cavitatingFoam
compressibleInterFoam
interFoam
interDyMFoam
interMixingFoam
interPhaseChangeFoam
LTSInterFoam
MRFInterFoam
MRFMultiphaseInterFoam
multiphaseInterFoam
porousInterFoam
settlingFoam
twoLiquidMixingFoam
twoPhaseEulerFoam

combustion

chemFoam
coldEngineFoam
dieselEngineFoam
dieselFoam
engineFoam
fireFoam
PDRFoam
reactingFoam
rhoReactingFoam
XiFoam

OpenFOAM - Notes

Standard Solvers

Particle-tracking flow

coalChemistryFoam
icoUncoupledKinematicParcelDyMFoam
icoUncoupledKinematicParcelFoam
LTSReactingParcelFoam
porousExplicitSourceReactingParcelFoam
reactingParcelFilmFoam
reactingParcelFoam
uncoupledKinematicParcelFoam

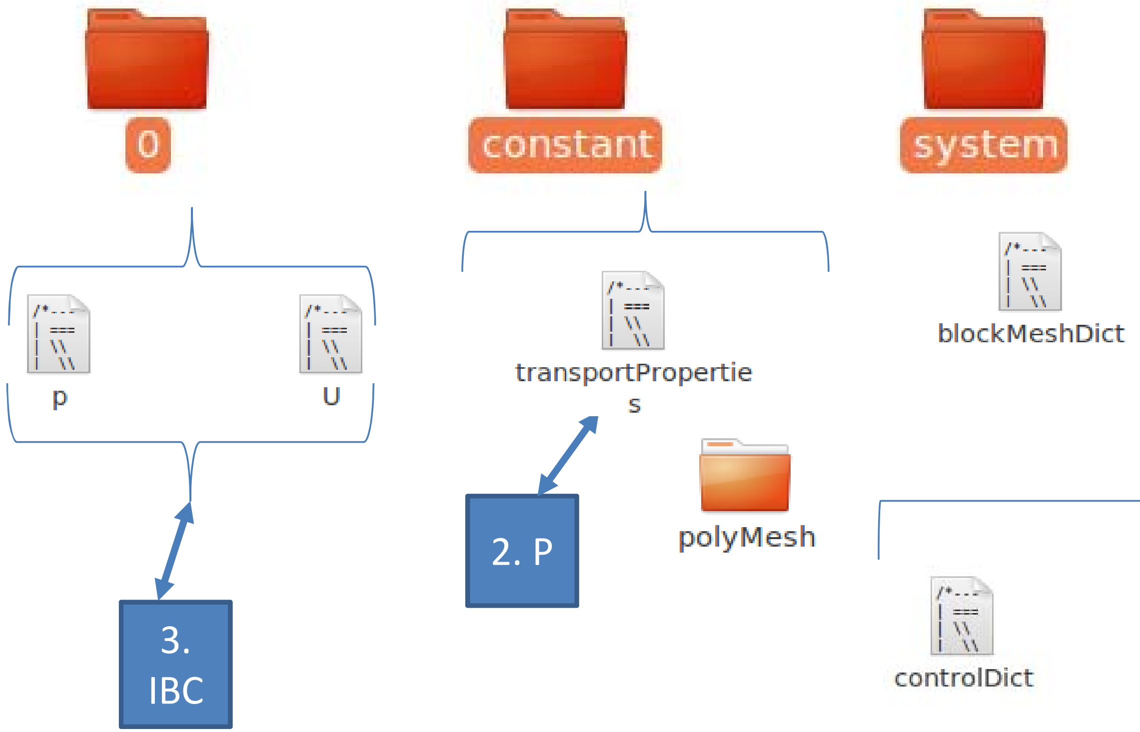
Direct simulation Monte Carlo methods

dsmcFoam	Direct simulation Monte Carlo (DSMC) solver for 3D, transient, multi- species flows

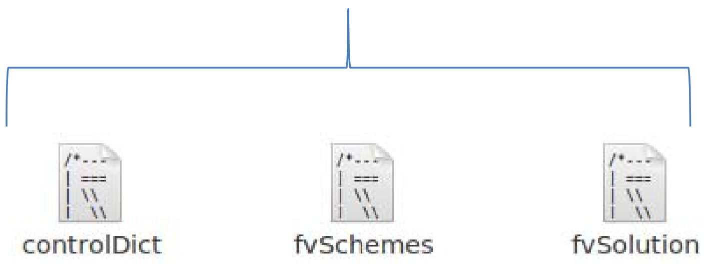
OpenFOAM - Use

- files

Pre-Processor:
Mesh (1), IC (3) +BC(1),Prop (2),
Parameters (4)
Solver → choice



1.
blockMesh (blockMeshDict in system folder)
cartesianMesh (MeshDict in system folder+stl in example)
snappyHexMesh - snappyHexMeshDict in system folder
ANSYS - ficheiro X.ans na pasta + ansysToFoam
→ polyMesh



4. Parameters,Solver

Rita F. de Carvalho

OpenFOAM - Use

```
cd $FOAM_RUN/tutorials/incompressible/icoFoam/cavity
blockMesh
icoFoam
paraFoam
```

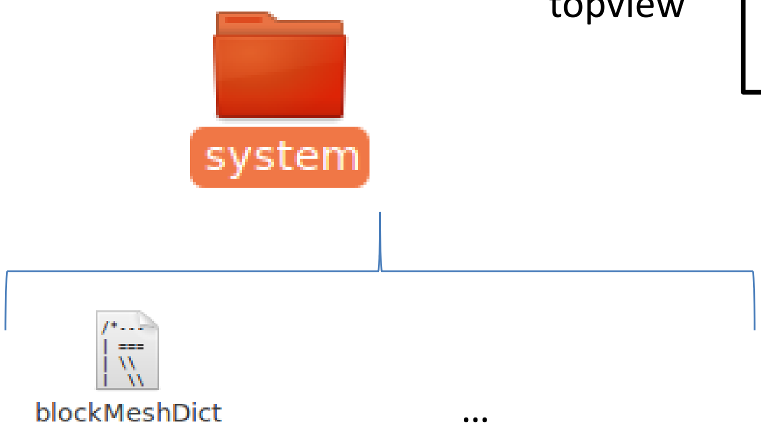
CFD – Open-Foam

Follow an example - laminar incompressible cavity

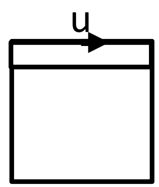
➔Change to directory of the example data

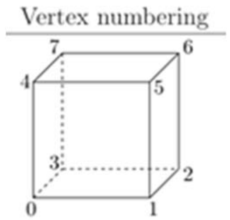
➔ dictionary **blockMeshDict– blockMesh**

1.
blockMesh (blockMeshDict in system folder)



topview





blockMeshDict

```
convertToMeters 0.1;

vertices
(
    (0 0 0)
    (1 0 0)
    (1 1 0)
    (0 1 0)
    (0 0 0.1)
    (1 0 0.1)
    (1 1 0.1)
    (0 1 0.1)
);

blocks
(
    hex (0 1 2 3 4 5 6 7) (20 20 1)
simpleGrading (1 1 1)
);

edges
(
);
```

blockMeshDict

```
boundary
(
    movingWall
    {
        type wall;
        faces
        (
            (3 7 6 2)
        );
    }
    fixedWalls
    {
        type wall;
        faces
        (
            (0 4 7 3)
            (2 6 5 1)
            (1 5 4 0)
        );
    }
    frontAndBack
    {
        type empty;
        faces
        (
            (0 3 2 1)
            (4 5 6 7)
        );
```

OpenFOAM - Use

```
cd $FOAM_RUN/tutorials/incompressible/icoFoam/cavity
```

CFD – Open-Foam

Follow an example - laminar incompressible cavity

➔Change to directory of the example data

➔ dictionary blockMeshDict– blockMesh

➔ verify other files if needed:

Properties (depend from solver) and choice of parameters and solvers (icoFoam)

IBC

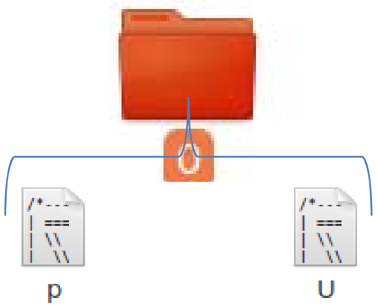
2. P

transportProperties

```
* //

nu              nu [ 0 2 -1 0 0 0 0 ] 0.01;
```

3. IBC



p

```
dimensions      [0 2 -2 0 0 0 0];

internalField   uniform 0;

boundaryField
{
    movingWall
    {
        type            zeroGradient;
    }

    fixedWalls
    {
        type            zeroGradient;
    }

    frontAndBack
    {
        type            empty;
    }
}
```

U

```
dimensions      [0 1 -1 0 0 0 0];

internalField   uniform (0 0 0);

boundaryField
{
    movingWall
    {
        type            fixedValue;
        value           uniform (1 0 0);
    }

    fixedWalls
    {
        type            fixedValue;
        value           uniform (0 0 0);
    }

    frontAndBack
    {
        type            empty;
    }
}
```

Rita F. de Carvalho

cd $FOAM_RUN/tutorials/incompressible/icoFoam/cavity

OpenFOAM - Use

4. Parameters,Solver

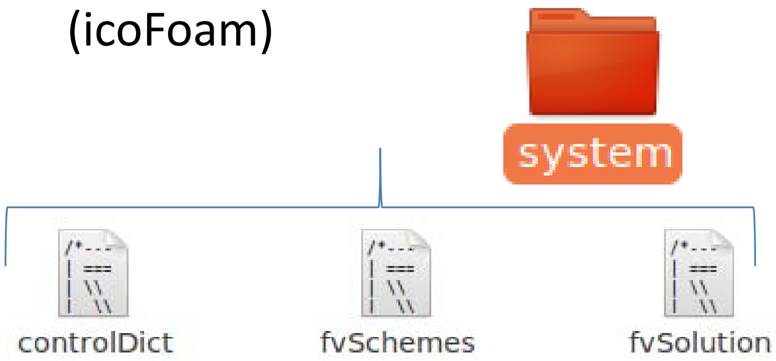
CFD – Open-Foam

Follow an example - laminar incompressible cavity

➔Change to directory of the example data

➔ dictionary blockMeshDict– blockMesh

➔ verify other files if needed: **IBC Properties** (depend from solver) and choice of parameters and solvers (icoFoam)



controlDict

```
application     icoFoam;
startFrom       startTime;
startTime       0;
stopAt          endTime;
endTime         0.5;
deltaT          0.005;
writeControl    timeStep;
writeInterval   20;
purgeWrite      0;
writeFormat     ascii;
writePrecision  6;
writeCompression off;
timeFormat      general;
timePrecision   6;
runTimeModifiable true;
```

```
ddtSchemes
{   default         Euler;}
gradSchemes
{   default         Gauss linear;
    grad(p)         Gauss linear;}
divSchemes
{   default         none;
    div(phi,U)      Gauss linear;}
laplacianSchemes
{   default         none;
    laplacian(nu,U)
    Gauss linear corrected;
    laplacian((1|A(U)),p)
    Gauss linear corrected;}
interpolationSchemes
{   default         linear;
    interpolate(HbyA) linear;}
snGradSchemes
{   default         corrected;}
fluxRequired
{   default         no;
    p               ;}
```

fvSolution

```
solvers
{
    p
    {
        solver          PCG;
        preconditioner  DIC;
        tolerance       1e-06;
        relTol          0;
    }

    U
    {
        solver          PBiCG;
        preconditioner  DILU;
        tolerance       1e-05;
        relTol          0;
    }
}

PISO
{
    nCorrectors     2;
    nNonOrthogonalCorrectors 0;
    pRefCell        0;
    pRefValue       0;
}
```

Rita F. de Carvalho

OpenFOAM - Use

- Create a new example
  - New folder with 3 subfolders

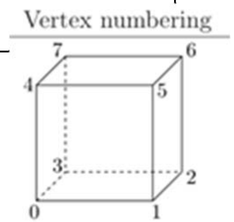

```
mkdir (nome do problema)
cd (nome do problema)
mkdir 0
mkdir constant
mkdir system
cd constant
cp –r …
```

  - Through home folder use copy paste for the chosen solver - interFoam

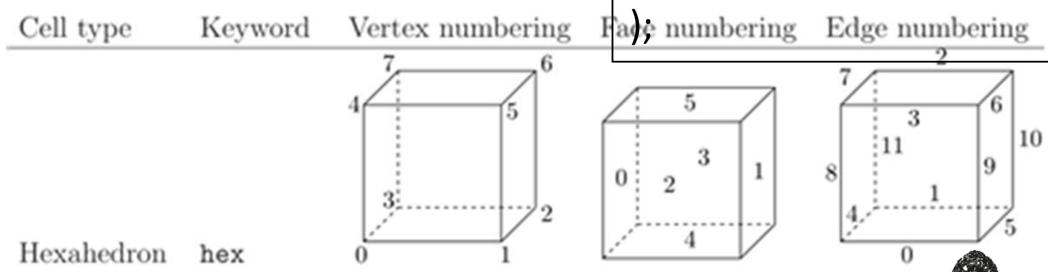
1.
blockMesh (blockMeshDict in system folder)

- Always 3D
- First in OpenFOAM is “0” and not “1”
- ConvertToMeters/ vertices / blocks / …

```
convertToMeters 0.001;
```




```
vertices
(
   (0 0 0)
   (1 0 0)
   (1 1 0)
   (0 1 0)
  …
);
```



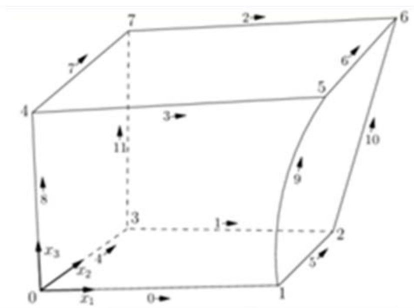
Rita F. de Carvalho

OpenFOAM - Use

- Create a new example
  - New folder with 3 subfolders

- blocks
- .../ edges (linear edges can be omitted) -

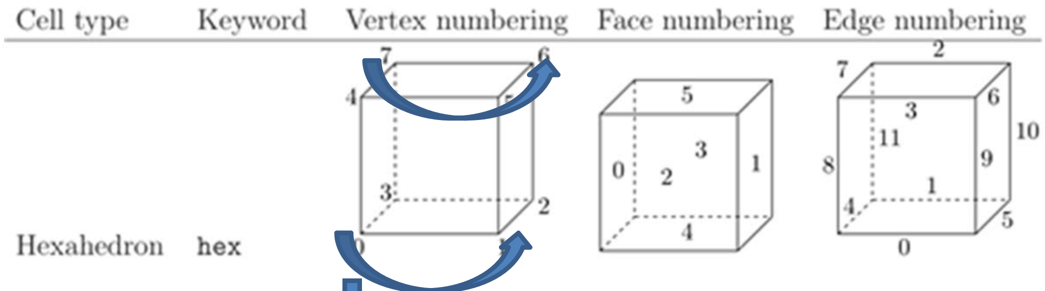
NOTA: non linear edge defined by 2 vertices followed by 2 or more interpolation points




```
edges
(
Arc 1 5 (2.0 -1.0 0.2)
);
```

- Arc (arc center),
- simpleSpline (int.p.)
- polyLine (int.p.)
- polySpline (int.p.)
- Line (0)

1.
blockMesh (blockMeshDict in system folder)




```
blocks
(
   hex (0 1 2 3 4 5 6 7) (100 50 25) simpleGrading (1 2 3)
);
```

```
simpleGrading (1 2 3)
```

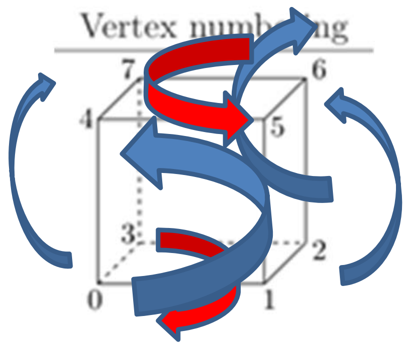
```
edgeGrading (1 1 1 1 2 2 2 2 3 3 3 3)
```

Rita F. de Carvalho

OpenFOAM - Use

1.
blockMesh (blockMeshDict in system folder)

- Create a new example
  - New folder with 3 subfolders



- .../patch

```
inlet (0 4 7 3); outlet(1 2 6 5);
walls (0 1 5 4),(0 3 2 1),(3 7 6 2),(4 5 6 7)
```


```
patches
(   patch inlet    (       (0 3 7 4))
    patch outlet    (       (1 5 6 2))
    wall wall    (  (0 1 2 3)  (0 4 5 1) (2 6 7 3) (4 7 6 5)));
mergePatchPairs();
```

```
patches
(
   left
   {
    type cyclic;
    neighbourPatch right;
    faces ((0 4 7 3));
   }
   right
   {
    type cyclic;
    neighbourPatch left;
    faces ((1 5 6 2));
   }
);
```

Rita F. de Carvalho

OpenFOAM - Use

```
cd $FOAM_RUN/tutorials/incompressible/icoFoam/cavity
blockMesh
icoFoam
paraFoam
```

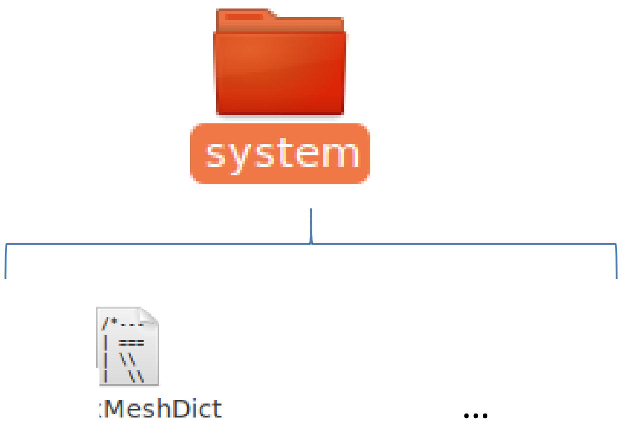
CFD – Open-Foam

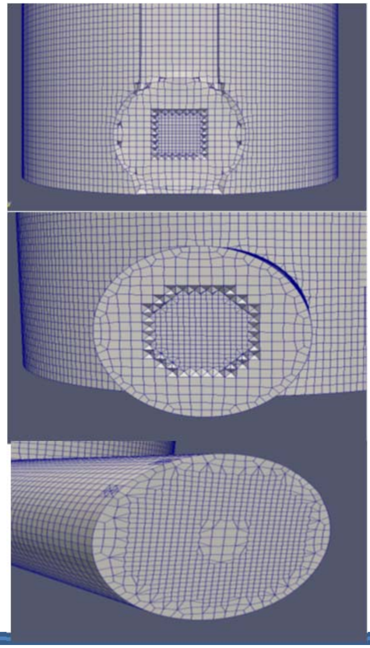
Follow an example - laminar incompressible cavity

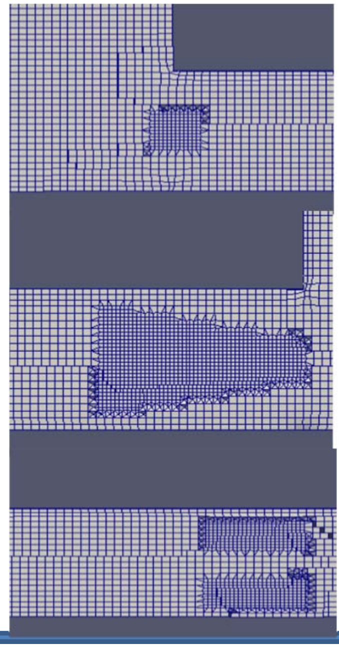
➔Change to directory of the example data

➔ dictionary **blockMeshDict– blockMesh**

1.
Salome to do the geometry and stl files
Use **surfaceFeatureEdges** and **MeshDict**







```
objectRefinements
{
    inlet_connection_box
    {
        cellSize 0.0022; //cellSize 0.002;
        type box;
        centre  (0.1175 0 0.0375);
        lengthX  0.02;
        lengthY  0.02;
        lengthZ  0.02;
    }

    outlet_connection_cone
    {
        cellSize 0.0022; //cellSize 0.002;
        type cone;
        p0  (-0.1175 0 0.0375);
        p1  (-0.2175 0 0.0375);
        radius0  0.01;
        radius1  0.03;
    }

    inlet_pipe_hollow_cone
    {
        cellSize 0.0022; //cellSize 0.002;
        type hollowCone;
        p0  (0.25 0 0.0375);
        p1  (0.3 0 0.0375);
        radius0_Inner  0.01;
        radius0_Outer  0.03;
        radius1_Inner  0.01;
        radius1_Outer  0.03;
    }
}
```

File produced by Nazmul Beg

Rita F. de Carvalho

OpenFOAM - Use

- Create a new exemple:

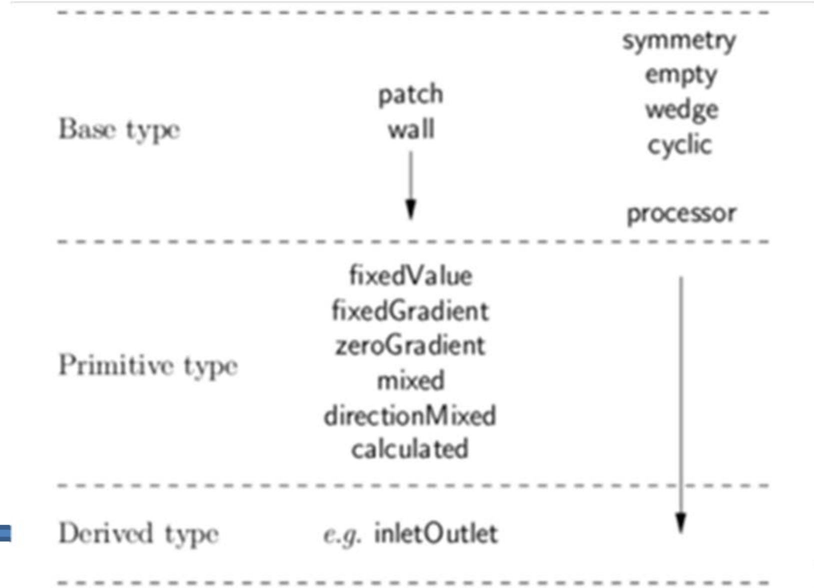
3. IBC

0 – Define BC – boundaries should be defined by patch accordingly physical properties

2. P

Properties (depends from solver)

g
transportProperties
turbulenceProperties
RASProperties
LESProperties
environmentalProperties
waveProperties
MRFZones



cd $FOAM_RUN/tutorials/incompressible/icoFoam/cavity

OpenFOAM - Use

3.
IBC

CFD – Open-Foam

Follow an example - laminar incompressible cavity

➔ Change to directory of the example data

- ➔ dictionary blockMeshDict– blockMesh
- ➔ verify other files if needed:

Properties (depend from solver) and choice of parameters and solvers (icoFoam)

IBC – Initial conditions through setFields

```
defaultFieldValues
(
    volScalarFieldValue alpha.water 0
    volVectorFieldValue U (0 0 0)
);

regions
(
        cylinderToCell
        {
        p1        (0.0 0.0 0.5); // start point on cylinder axis
        p2        (1.5 0.0 0.5);   // end point on cylinder axis
        radius    0.01856;
        fieldValues
         (
             volScalarFieldValue alpha.water 1
             volVectorFieldValue U (4.6 0 0)
        );
    }
```

Rita F. de Carvalho

OpenFOAM - Use

4. Parameters,Solver

- Create a new exemple:

- fvScheme -Define the approximations for the operators
- fvSolution - Define how to solve the system of equations

```
 object      fvSolution;
}
// * * * * * * * * * * * * * * * * * * //

solvers
{
   T
   {
      solver                       PCG;
      preconditioner               DIC
      tolerance                    1e-07;
      relTol                       0.1;
   }
}

SIMPLE
{
   nNonOrthogonalCorrectors 2;
}
```

iterative method

Depending on the structure of your matrix, this will allow a reduction on the number of iterations

Tolerance for stopping the iterative procedure

relTol is the relative tolerance between the initial and the final residual

```
PCG  : Solving for T, Initial residual = 0.584235, Final residual = 1.0582e-07, No Iterations 3
```

$$r = \|AT_0 - B\| \qquad r = \|AT_{final} - B\|$$

solver

number of iterations

Rita F. de Carvalho

OpenFOAM - Use

- Create a new exemple:

➔ controlDict

```
libs (
  "libswakFunctionObjects.so"
  "libsimpleFunctionObjects.so"
  "libgroovyBC.so"
);
```

4. Parameters,Solver

- Other functions to include variable boundaries: swak4foam

Others dictionaries

» controlDict
» setFieldsDict
» topoSetDict
» refinemeshDict
» decomposeParDict

Rita F. de Carvalho

OpenFOAM - Use

CFD – Open-Foam

Follow an example - laminar incompressible cavity

➔Change to directory of the example data

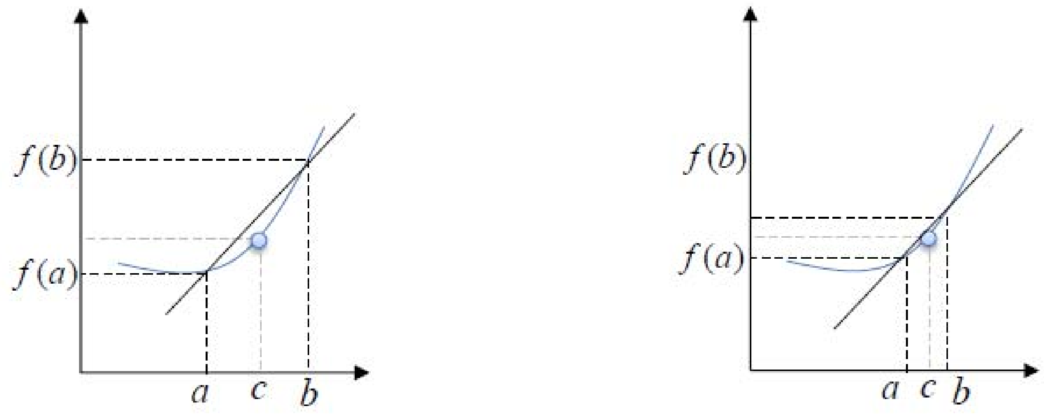
➔ dictionary blockMeshDict– blockMesh

➔ verify other files if needed: **Properties** (depend from solver) and choice of parameters and solvers (icoFoam)

IBC – Initial conditions through setFields

➔ **Try to change mesh – look at resuts and parameters**

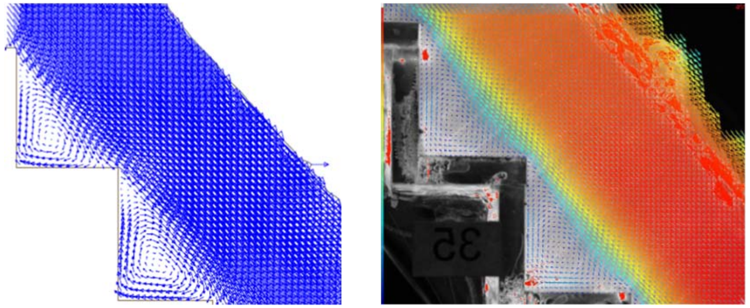
4. Run diferente meshes – usually 3

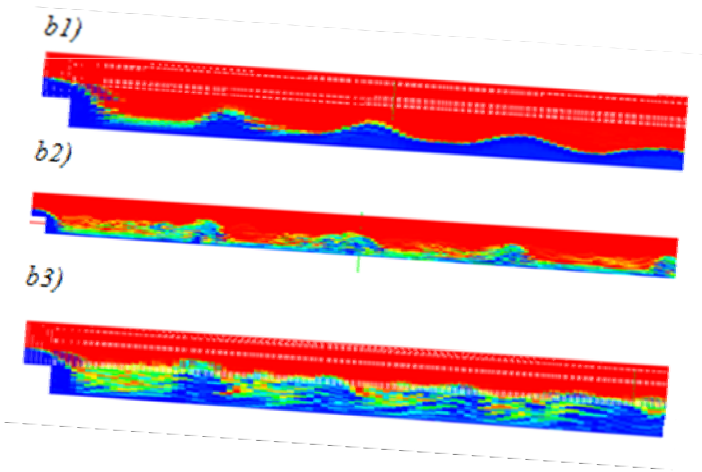


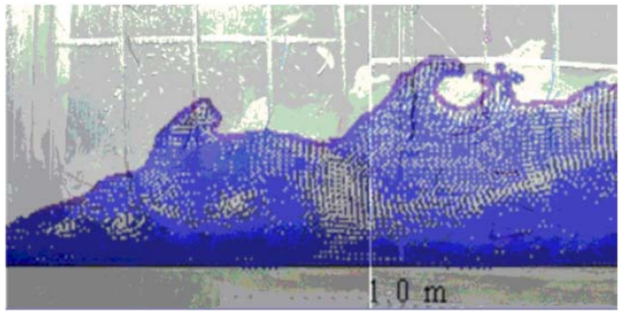
$$f'(c) \approx \frac{f(b)-f(a)}{b-a} \qquad f'(c) \approx \frac{f(b)-f(a)}{b-a}$$

Rita F. de Carvalho

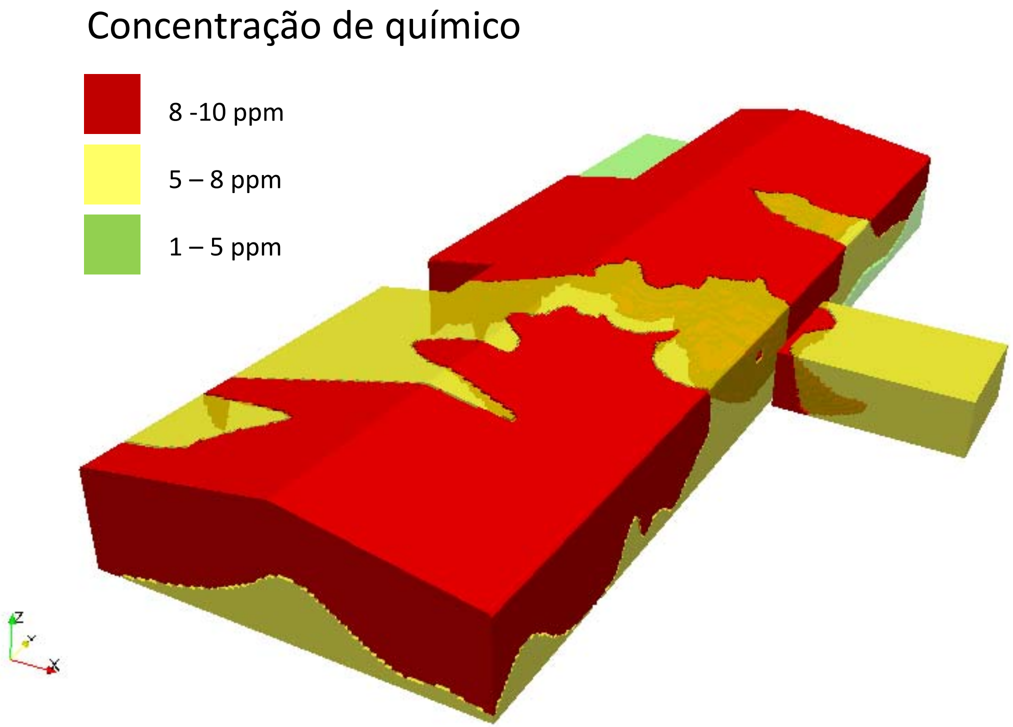
EXAMPLES: Roughness Channel / Hydraulic Jump

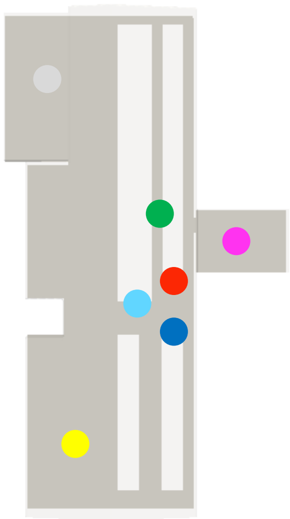


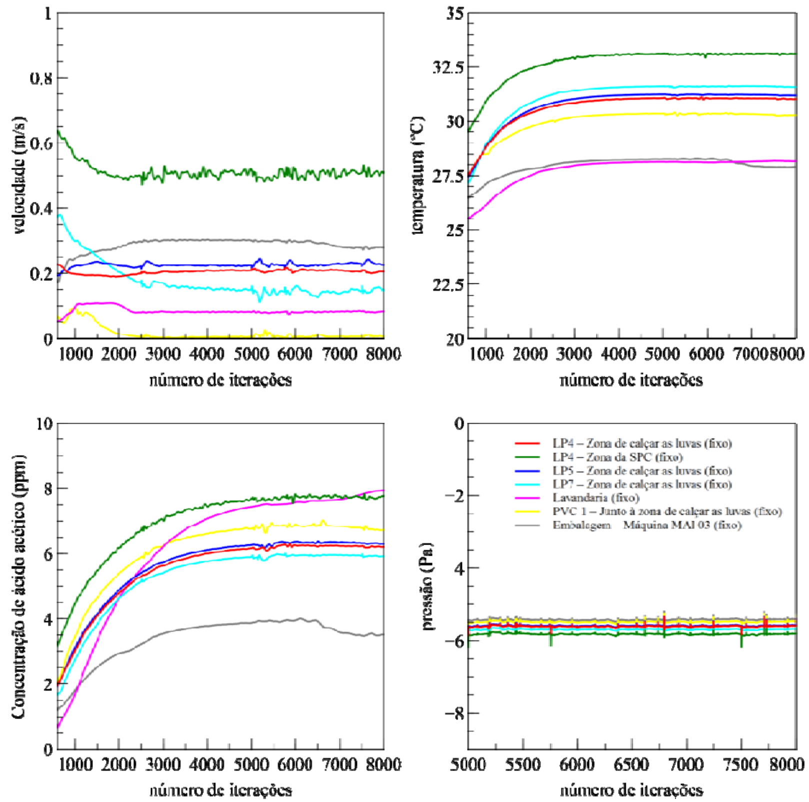




EXAMPLE: Industrial building

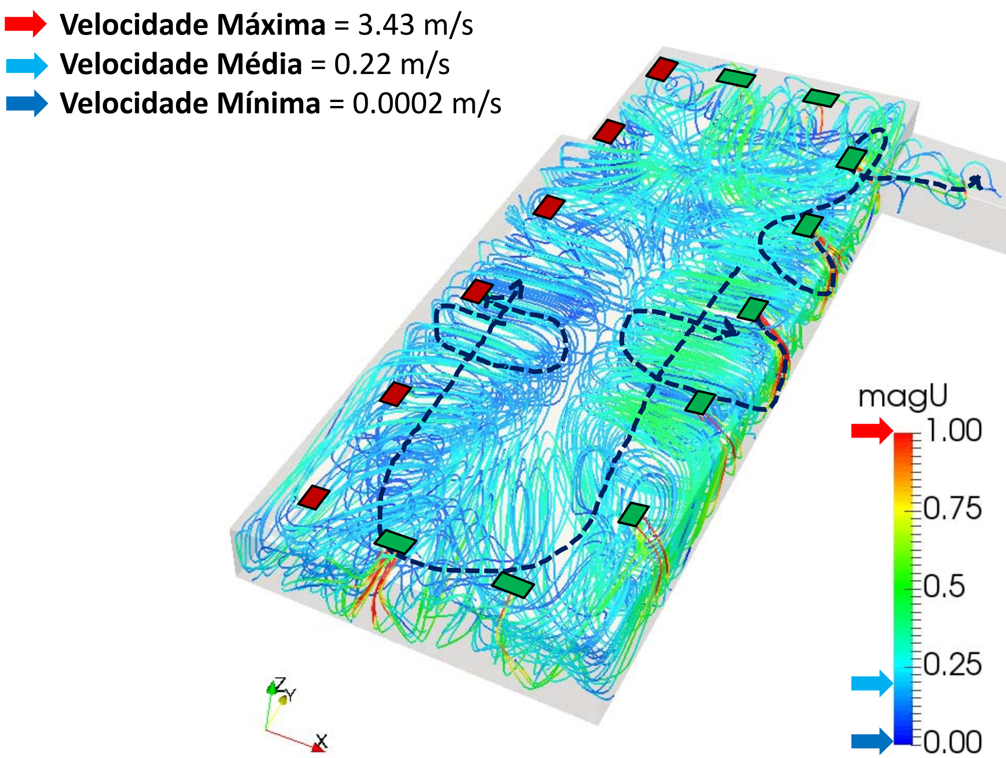


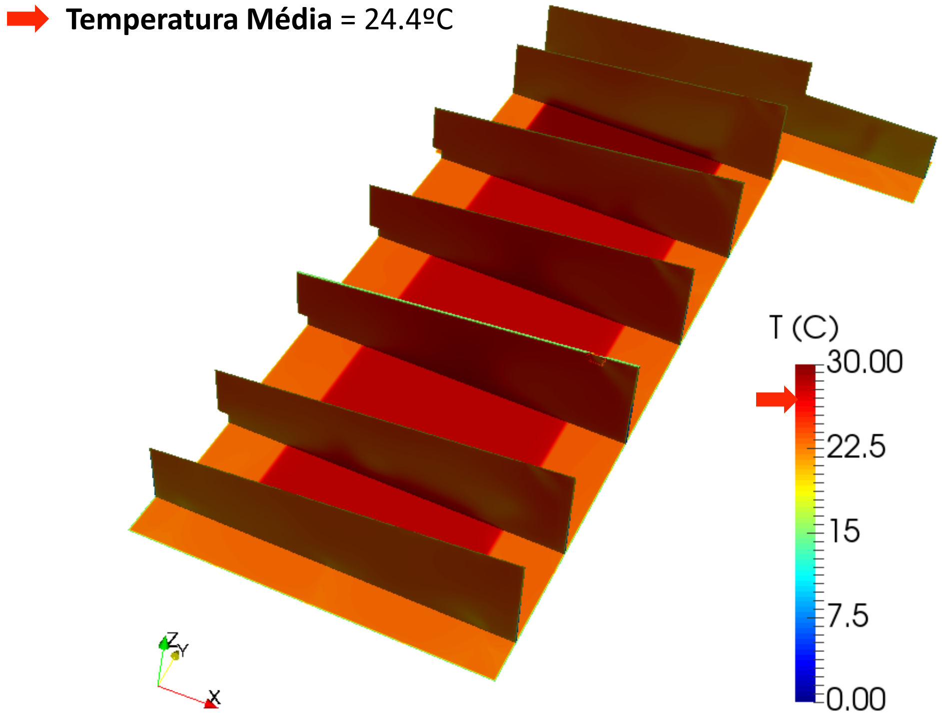




Model constructed by Pedro Lopes

EXAMPLE: Municipal swimming-pool





Model constructed by Pedro Lopes

EXAMPLE: Other hydraulic aplications

https://www.youtube.com/watch?v=gJBsgdNTRDw&list=PL6BfJWHH53_pLUOzU8uuzXgvrhR6uBAR6&index=4
https://www.youtube.com/watch?v=Q6XyTrhaGxc&list=PL6BfJWHH53_pLUOzU8uuzXgvrhR6uBAR6&index=7
https://www.youtube.com/watch?v=6Wc2lJ4N-2s&list=PL6BfJWHH53_pLUOzU8uuzXgvrhR6uBAR6&index=9
https://www.youtube.com/watch?v=6Wc2lJ4N-2s&list=PL6BfJWHH53_pLUOzU8uuzXgvrhR6uBAR6&index=9
https://www.youtube.com/watch?v=bR3rulCQS08&list=PL6BfJWHH53_pLUOzU8uuzXgvrhR6uBAR6&index=11

https://www.youtube.com/watch?v=OH0l5CkY_aY
https://www.youtube.com/watch?v=Ky7Ksdf9ihs
https://www.youtube.com/watch?v=SCQL57wHVq4&index=4&list=PL6BfJWHH53_pNtHuG0JAEGsW7WzteQbsL

https://www.youtube.com/watch?v=FhkGIa8J9b0

OpenFOAM – Conclusions

What is openFoam
Applications
Current Forks and versions
Structure and Organization
what is behind software
Main Characteristics, from equations to Pre-processing mesh, properties, options, choose a solver and see results
Equations in a Solver and code
Basic Solvers: LaplacianFoam, scalarTransportFoam
Other Solvers
Prepare a case: ideas of mesh construction; properties, IBC, options
Understand the importance of fvScheme (approximations for the operators) and fvSolution (solve the system of equations)
Understand the importance of study a mesh
Examples and Potential of OpenFoam

OpenFOAM – Next Basic Courses

Basic Meshing Wagner Galuppo (11:30-13:00)

Post-processing Célio Fernandes (14-15:30)

Multi-phase flows József Nagy (16-17:30)