SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP

# Hands-on with OpenFOAM part I

G. Amati, CINECA
R. Ponzini, CINECA
A. Memmolo, CINECA

09/22

Univerza v Ljubljani

TECHNISCHE UNIVERSITÄT WIEN

CINECA

VSB TECHNICAL UNIVERSITY OF OSTRAVA | IT4INNOVATIONS NATIONAL SUPERCOMPUTING CENTER

Co-funded by the Erasmus+ Programme of the European Union

This project has been funded with support from the European Commission.
This publication [communication] reflects the views only of the author, and the Commission cannot be held responsible for anyuse which may be made of the information contained therein.

# Outline

SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP

**Circular cylinder in cross flow: CFD modelling**

- Problem description
- Problem physics
- CFD modelling with OpenFOAM toolbox: an introduction
  - Geometry management and meshing
  - Solver selection and setting
  - Monitoring the simulation, data post-processing visualization and animation
- Problem implementation in OpenFOAM
- Resources and References

# Problem description

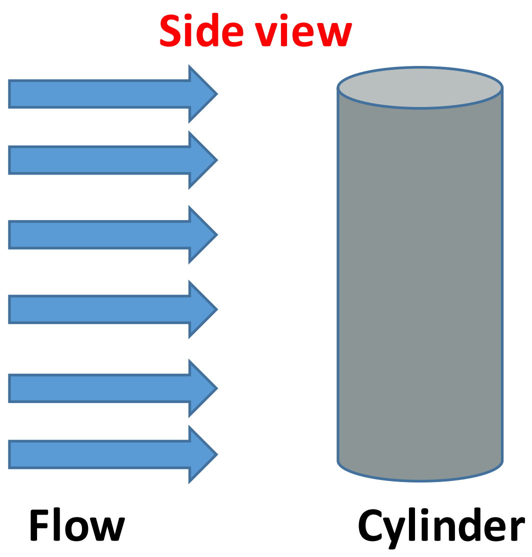
SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP

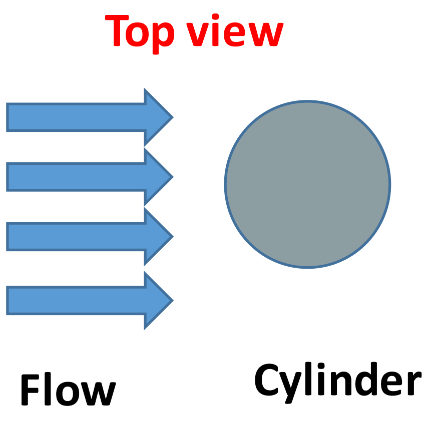
- The shedding of vortices behind bluff bodies, and particularly from two-dimensional circular cylinders, is perhaps one of the most studied subjects in fluid mechanics.
- Many acute observers of the physical reality (like Leonardo da Vinci) had already been attracted by this phenomenon
- Hundreds of papers were published in the scientific literature in the last century
- First recognized reference is by Von Karman in 1911
- Several reviews dealing with vortex shedding are present in the literature most of which regard in particular vortex shedding from a circular cylinder. This type of body is certainly the most studied one.

# Problem description

SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP

- Considering a bluff body having a plane of  symmetry in the flow direction
- Most of literature is related to cylinder but also other shapes have been investigated



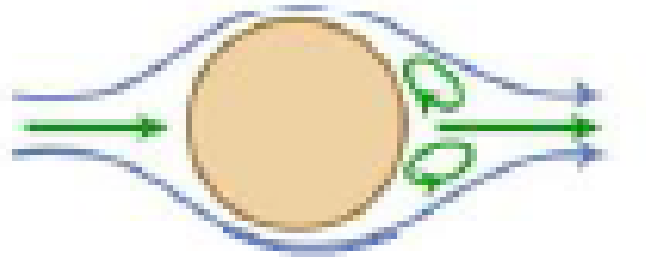



# Problem description

SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP

- There are a set of well-defined condition that occurs
- The changes in the condition is related to a non-dimensional fluid dynamics number, the Reynolds number (*Re*)

1. **Re is low:** two standing symmetrical vortices form behind the body are present. The shear layers separating from the body enclosing the boundary of the recirculation region containing the vortices. The length of the closed vortical region grown when increasing the Reynold number value



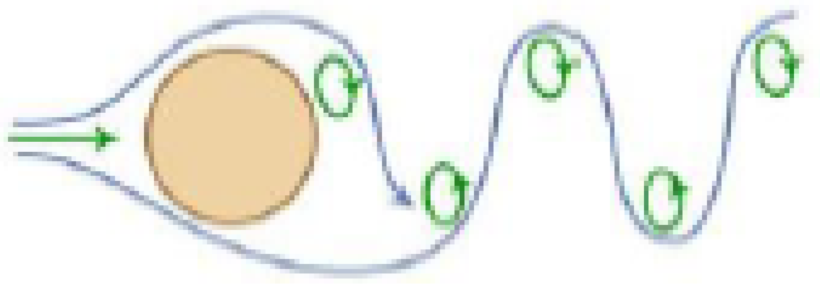
$$Re = VD/\nu$$

*D* is the lateral dimension of the body, *V* a reference velocity and ν the kinematic viscosity

# Problem description

SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP

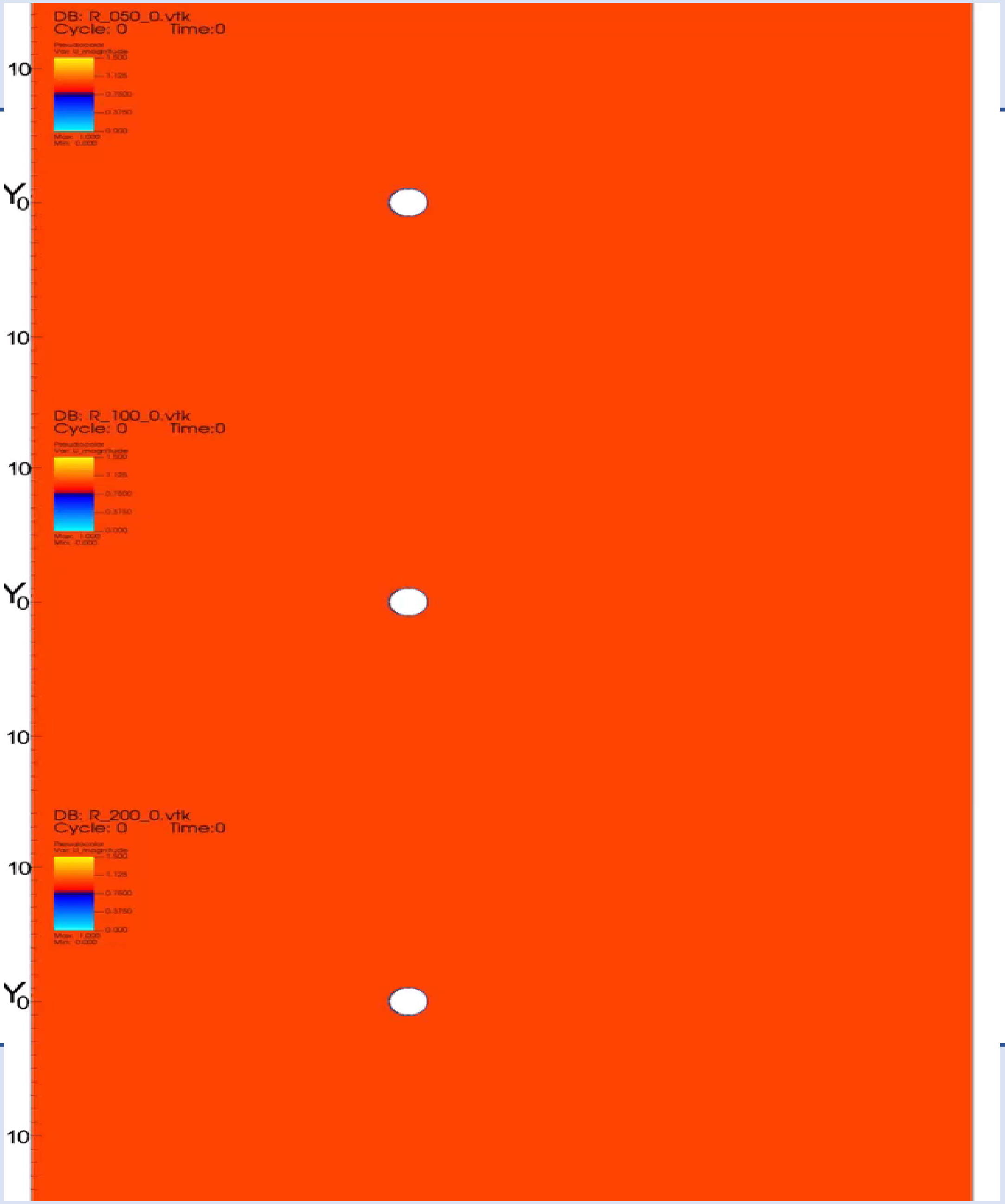
2. the Reynolds number is increased above the so-called critical value (*Re_critical*): the steady configuration becomes unstable, starting from the downstream end of the recirculating region, and a new time-dependent equilibrium flow is reached, which is characterized by the regular alternate shedding of vortices, with a definite frequency f, from the two sides of the body. Thus, generating the so-called vortex shedding phenomenon also known as vortex street or Von Karman street. *Re_critical* depends on the shape of the body; it is around 47 for a circular cylinder



# Implementation in OpenFOAM

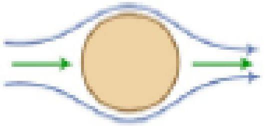


- Re=50
- Re=100
- Re=200



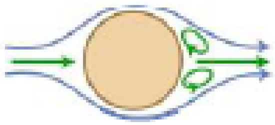
# Problem description





Creeping flow (no separation)
Steady flow

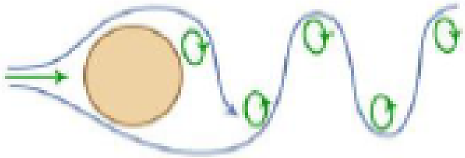
Re < 5



A pair of stable vortices in the wake
Steady flow

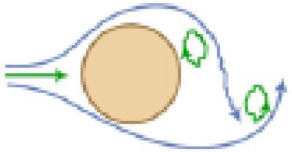
5 < Re < 40 - 46





Laminar vortex street (Von Karman street)
Unsteady flow

40 - 46 < Re < 150



Laminar boundary layer up to the separation point, turbulent wake
Unsteady flow

150 < Re < 300

Transition to turbulence

300 < Re < 3 x $10^5$

# Problem description

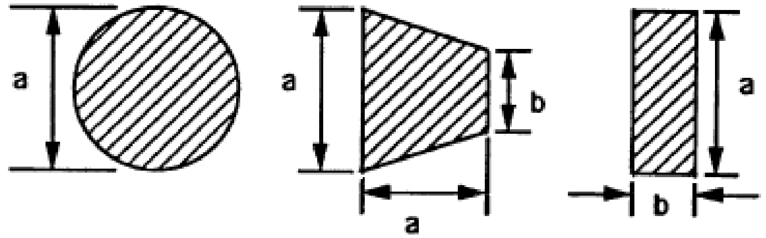


- The body shape is a driving parameter for the initiating of the physical phenomenon, and this explains why different Strouhal numbers (*St*) are found for different bodies.

$$St = f_s D/V$$

where fs is the frequency of vortex shedding

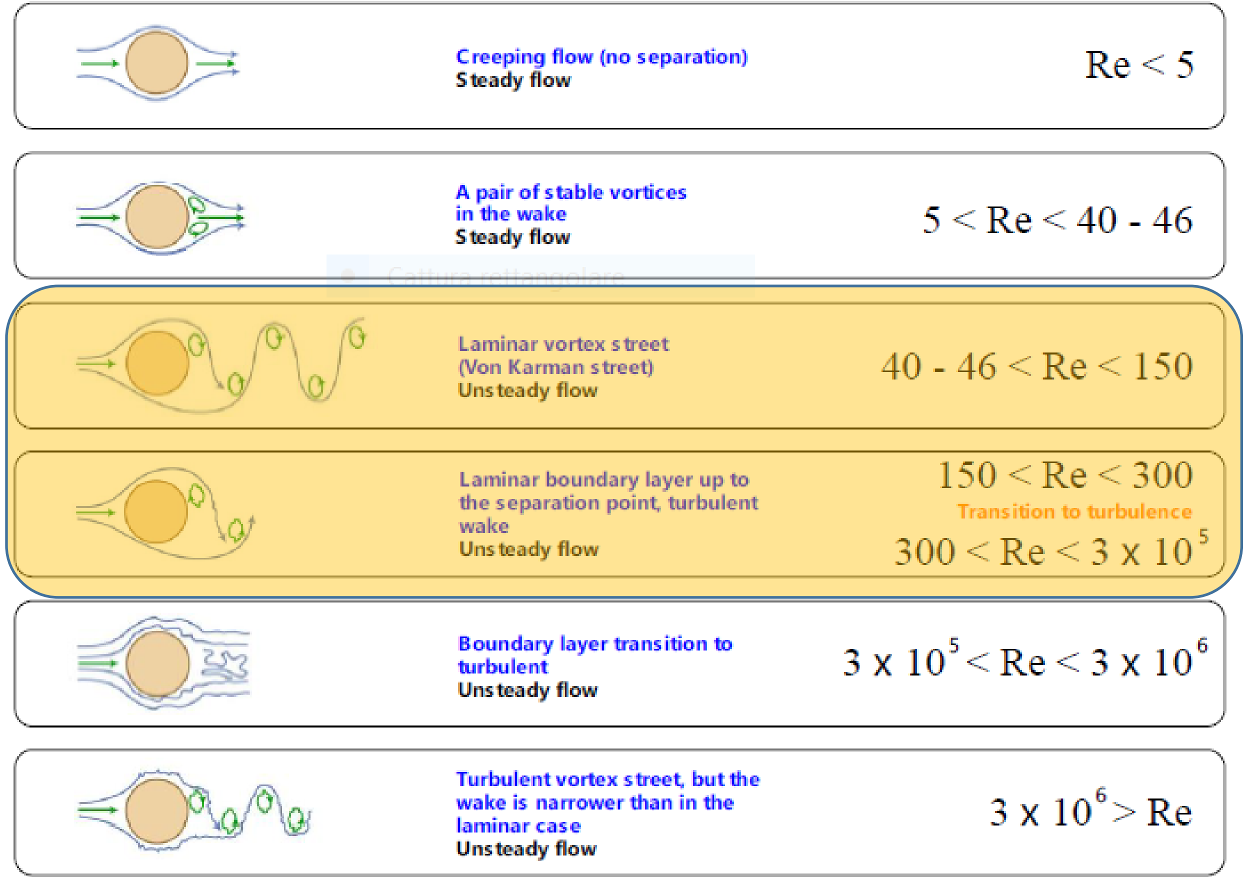
The geometry of the shedder bar determines the characteristic of the frequency of vortices. Three shapes for the shedder bar were selected for this study. They are circle, rectangular, and reversed wedge as shown in following figure 2.4.




Example of application in hydraulic flow meters
https://ntrs.nasa.gov/citations/19910010723

# Problem physics



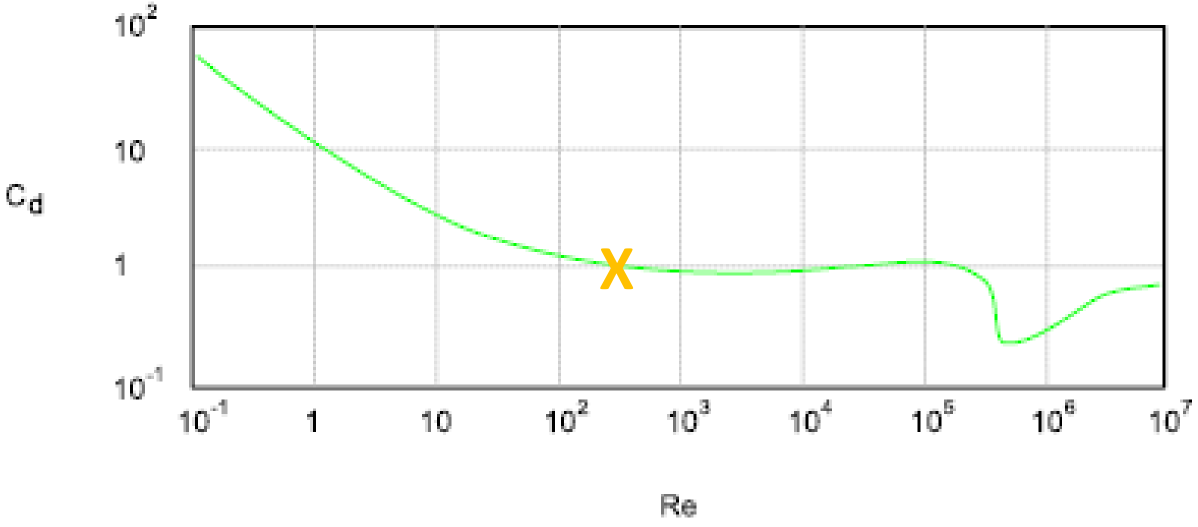



We are going to approach using OpenFOAM a 2D CFD model of a Laminar Vortex Street for Reynolds number around 200

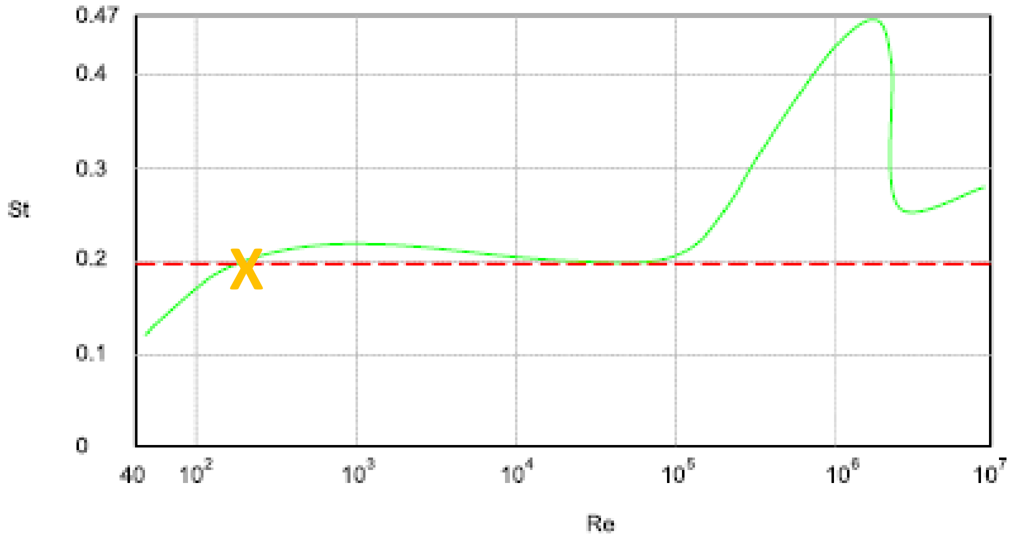
# Problem physics

SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP

Drag Coefficient as a function of Re



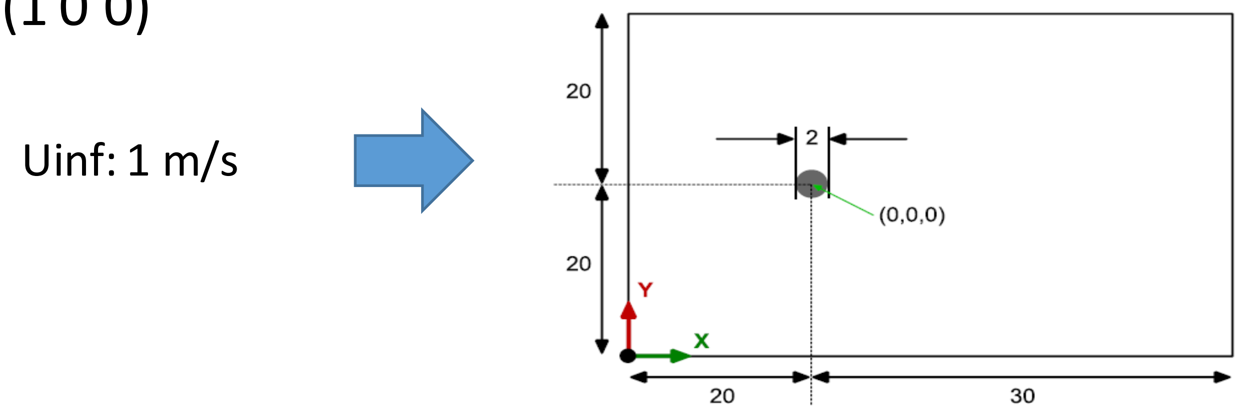

Strouhal number as a function of Re



# Re = 200



- In order to have a problem physics for a given Reynolds number we can play with several physical parameter.
- We decide to set the fluid as air (this choice will set density and viscosity)
- We decide to set the geometry dimension of the cylinder cross section equal to 2m and the overall domain extension as (50x40x1) m
- The given free stream velocity will be therefore defined by the *Re* definition as equal to 1 m/s in the x-direction (1 0 0)



# CFD modelling with OpenFOAM toolbox: an introduction

# What is OpenFOAM

SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP

- OpenFOAM was created by Henry Weller in 1989 under the name “FOAM” and was released open source as “OpenFOAM” by Henry Weller, Chris Greenshields and Mattijs Janssens in December 2004.
- OpenFOAM is today the standard de-facto, open-source software for computational fluid dynamics (CFD).
- The OpenFOAM Foundation distributes OpenFOAM exclusively under the General Public License (GPL). The GPL gives users the freedom to modify and redistribute the software and a guarantee of continued free use, within the terms of the license.
- The current version is 10. In 2014, the development line of OpenFOAM, known as “OpenFOAM-dev” was released publicly on GitHub.

# How to get it

SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP

From the OpenFOAM Foundation site: https://openfoam.org/

- The User Guide: https://cfd.direct/openfoam/user-guide/
- Download the toolbox: https://cfd.direct/openfoam/download/

# Classic CFD cycle

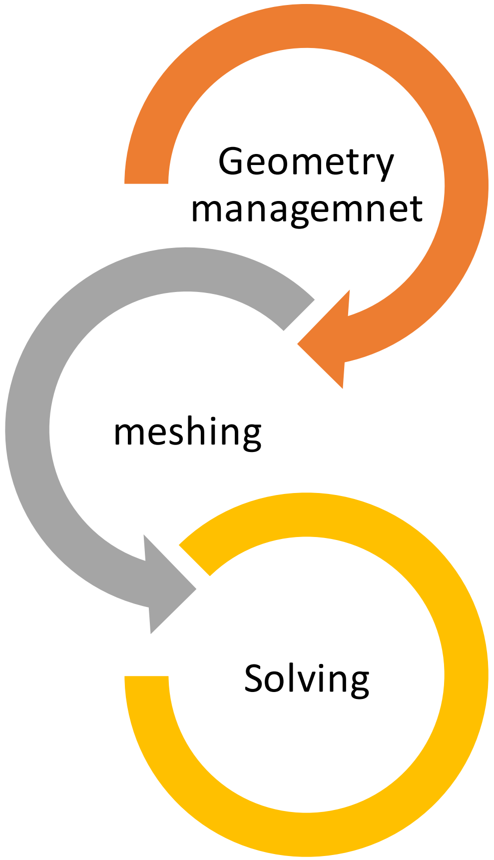


- Computational Fluid Dynamics (CFD) typical working cycle is made of a set of steps or elementary bricks that are equally relevant to allow engineers to perform their day-by-day activity:
  - Cad/geometry management
  - Meshing
  - Solving
  - Monitoring solution
  - Visualize/analyze results
- OpenFOAM (OF) is a complete toolbox in the sense that contains and embed all the necessary bricks to perform all the steps listed above

# Classic CFD cycle



- Usually, it is important to note that some activities are iterated in a cyclic way like that:
- This is a central point of common CFD workflow and the availability of robust and easy to use tools is essential
- OF is designed to allow the user to efficiently work and perform the necessary activities by means of a set of command lines functions
- Some functions works as stand-alone, other requires dictionary to be correctly executed



# Dictionary in OpenFOAM

SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP

- The concept of dictionary in OF is central
- A dictionary in OF is just a txt, human readable, file made of a set of entries (keys) that requires meaningful values in order to enable the correct execution of solvers and functions
- Each dictionary might contain sub-dictionaries each one with its own set of entries.
- If all the necessary entries are not given/defined the desired command or solver will not run and will exit with error
- Part of the complexity or unfriendliness of OF for beginners is related to that model; nevertheless, as we will see in the forthcoming there are *tutorials and templates* to support users setting up running cases with minimal effort

# Geometry management/1



- Geometry input is the basis for every CFD modelling
- To handle geometries OF requires two main kind of triangulated files: Object files (.obj) or stereolithography (.stl) either as binary or ascii format.
- The quality of the triangulated surface is relevant for the quality of the final meshing procedure, but the essential condition is that the geometry is watertight. This condition is not strictly mandatory, but it is more convenient to start with a "*watertight*" geometry to avoid meshing errors and/or wasting of time.
- OF have a set of command line to manage the geometry input but does not contain a geometry modeler

# Geometry management/2

SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP

- There are several valuable open-source geometry modeler that can be used to design geometries from scratch or to modify/simplify incoming 3D CAD files: **Salome, FreeCAD, Onshape, Blender** among the others

- OF have instead a set of command line tools to:
  - Evaluate the quality of the input geometry file
  - Translate/Rotate/Scale the geometry dimensions
  - Orient normal directions

# Geometry management/3



- *surfaceCheck [OPTIONS] <surface file>*

- Relevant info are:
  - Bounding box: you can check the main dimensions
  - Quality of triangulated surface
  - Opening/closeness of the geometry

```
Vertices     : 79219
Bounding Box : (-8.82013e-18 -0.212 3.05452e-09) (1.899 0.212 0.185222)

Region  Size
------  ----
Mesh_1  158434


Surface has no illegal triangles.

Triangle quality (equilateral=1, collapsed=0):
    0 .. 0.05  : 0.0050368
    0.05 .. 0.1  : 0.00136334
    0.1 .. 0.15  : 0.000984637
    0.15 .. 0.2  : 0.000940455
    0.2 .. 0.25  : 0.000984637
    0.25 .. 0.3  : 0.00085209
    0.3 .. 0.35  : 0.000965702
    0.35 .. 0.4  : 0.000934143
    0.4 .. 0.45  : 0.00124342
    0.45 .. 0.5  : 0.00213969
    0.5 .. 0.55  : 0.00227855
    0.55 .. 0.6  : 0.0022533
    0.6 .. 0.65  : 0.00249315
    0.65 .. 0.7  : 0.00302965
    0.7 .. 0.75  : 0.00405847
    0.75 .. 0.8  : 0.00650113
    0.8 .. 0.85  : 0.00717018
    0.85 .. 0.9  : 0.012188
    0.9 .. 0.95  : 0.0249946
    0.95 .. 1  : 0.919588

    min 4.70506e-07 for triangle 33187
    max 1 for triangle 88416

Edges:
    min 7.66607e-05 for edge 56509 points (1.89706 0.000336458 0.0628943)(1.89704 0.000262703 0.0628883)
    max 0.0122523 for edge 61338 points (1.61003 -0.20367 0.153604)(1.6221 -0.201712 0.152828)

Checking for points less than 1e-6 of bounding box ((1.899 0.424 0.185222) metre) apart.
Found 0 nearby points.

Surface is closed. All edges connected to two faces.
```

# Geometry management/4

SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP

- *surfaceTransformPoints [OPTIONS] <surface file> <output surface file>*

*options:*

*-rollPitchYaw <vector>:* transform in terms of '( roll pitch yaw )' in degrees

*-rotate <(vectorA vectorB)>:* transform in terms of a rotation between <vectorA> and <vectorB> - eg, '( (1 0 0) (0 0 1) )'

*-scale <vector>:* scale by the specified amount - eg, '(0.001 0.001 0.001)' for a uniform [mm] to [m] scaling

*-translate <vector>:* translate by the specified <vector> - eg, '(1 0 0)'

*-yawPitchRoll <vector>:* transform in terms of '( yaw pitch roll )' in degrees

SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP

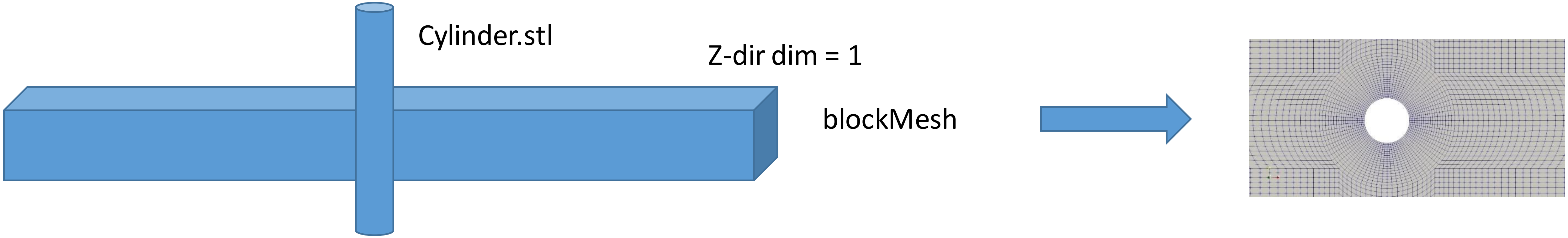
- *surfaceOrient [OPTIONS] <surface file> <output surface file> <visiblePoint>:*

```
(base) [a07lin00@r033c01s03 ~]$ surfaceOrient geometry.stl geometryOrineted.stl '(1e10 1e10 1e10)'
/*---------------------------------------------------------------------------*\
  =========                 |
  \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox
   \\    /   O peration     | Website:  https://openfoam.org
    \\  /    A nd           | Version:  8.0
     \\/     M anipulation  |
\*---------------------------------------------------------------------------*/
Build  : 8.0-56290025169b
Exec   : surfaceOrient geometry.stl geometryOrineted.stl (1e10 1e10 1e10)
Date   : Jun 22 2021
Time   : 15:58:58
Host   : "r033c01s03"
PID    : 37313
I/O    : uncollated
Case   : /galileo/home/usera07lin/a07lin00
nProcs : 1
sigFpe : Enabling floating point exception trapping (FOAM_SIGFPE).
fileModificationChecking : Monitoring run-time modified files using timeStampMaster (fileModificationSkew 10)
allowSystemOperations : Allowing user-supplied system call operations

// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //
Reading surface from "geometry.stl"
Orienting surface such that visiblePoint (1e+10 1e+10 1e+10) is outside
Did not flip orientation of any triangle of surface.
Writing new surface to "geometryOrineted.stl"
End
```

# Geometry management: Comment



- For the sake of interest of the 2D cylinder test case the geometry management tools are not of interest
- A 3D approach for meshing would be possible but the final mesh quality would not be optimal (presence of pyramid elements instead of a fully hexahedral mesh (optimal))
- 3D approaches are mandatory when geometry complexity is relevant



# Meshing

- Once we are fine with geometry definition of the shape we want to study or use to define our problem space, we must produce a valid discretization.
- OF have two main tools to produce high quality meshes for both simple and complex (industrial level) geometries
  - ***blockMesh****:* fully structured hexahedral mesher
  - ***snappyHexMesh****:* unstructured hexa-dominant mesher
- Both tools are instructed using the typical OF dictionary-based strategy
- Both tools produce high quality meshes

# Meshing: **blockMesh**

SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP

- ***blockMesh****:* is the basis of the meshing tool in OF. It can be used in conjunction with *snappyHexMesh* or as standalone tool.
- Meshes created with blockMesh are usually 100% made of hexahedra even if other mesh cell shape are allowed.
- The usual/average usage of the *blockMesh* tool is to create the background starting mesh of the computational domain before move to *snappyHexMesh* to finalize the mesh.
- For a simple computational domain *blockMesh* is well suited but for industrial complex domain is not suited and requires the usage of *snappyHexMesh* as additional step.

# Meshing: snappyHexMesh

SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP

- ***snappyHexMesh*** is the standard OF meshing approach for complex industrial geometries
- As for the *blockMesh* it works by means of a dictionary where the user can instruct several input parameters to handle different parts of the meshing workflow as desired
- The main steps are:
  - **Castellated**: the discretization
  - **Refinement**: the background discretization is refined where needed (usually surfaces or gaps)
  - **Snappage**: starting from the refined castellated mesh a set of algorithms is applied to project the mesh faces into the geometry description as defined by the stl triangulated surfaces
  - **Layering**: once the mesh is defined all over the domain, the user can decide to add a prismatic boundary layer to selected surfaces (typical case of RANS solvers with wall function enabled)

# Mesh quality



- After mesh is done, we can check it using the ***checkMesh*** function
- *checkMesh* is a very complete and useful synthetic mesh checker
- If the check is ok, then we are ok
- If some warning or error is pointed out it does not necessarily mean that we cannot run the mesh. Nevertheless, if the run crashes we know that the mesh can be the point.
- High quality meshes are also solver friendly and the opposite is also true.

# Solvers

SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP

- OF have a wide variety of solvers (about 60) included in the standard release
- The solvers are divided by physics and then by solver for a specific problem with this logic:

```
solvers
├── basic
├── combustion
├── compressible
├── discreteMethods
├── DNS
├── electromagnetics
├── financial
├── heatTransfer
├── incompressible
├── lagrangian
├── multiphase
└── stressAnalysis
```

```
solvers/incompressible/
├── adjointShapeOptimisationFoam
├── boundaryFoam
├── icoFoam
├── pimpleFoam
├── pisoFoam
├── shallowWaterFoam
└── simpleFoam
```

# Solvers

SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP

- OF comes also with a wide range of running tutorials
- A very convenient and practical way of doing is to:
  - Identify the physics of your problem
  - Look into the tutorial's directories for such physics or problem
  - Dive into the tutorial and try to understand it
  - Run it
  - Modify it to satisfy your needs
- There is also a more general approach based on given templates to solve different physical problems offered into the main distribution of OF under $FOAM_ETC/templates
- The templates can be used similarly to the tutorials to have a clean case base to be personalized
- In both cases (tutorials and templates) you will find a meaningful *README* file or an *AllRun* file that will show how to use the material

# Solvers

SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP

- The general case requires a set of mandatory directories o run in OF:
  - system/
  - constant/
  - 0/
- Depending on the solver type and thus on the physics that we want to solve there are a wide variety of required dictionary under these main directories

**OpenFOAM-10.0/etc/templates/inflowOutflow/**

```
├── 0
│   ├── k
│   ├── nut
│   ├── omega
│   ├── p
│   └── U
├── constant
│   ├── momentumTransport
│   ├── transportProperties
│   └── triSurface
├── README
└── system
    ├── blockMeshDict
    ├── controlDict
    ├── fvSchemes
    ├── fvSolution
    ├── meshQualityDict
    ├── snappyHexMeshDict
    └── surfaceFeaturesDict
```

# Data sampling

SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP

- Once the solver is set up and running there are several quantities that the user might want to monitor and or sampling
- OF have a convenient set of *functionObjects* to support a wide range of possible quantity sampling
- The full list is extremely large and you can access it by means of the command **`postProcess -list`**

```
Available configured functionObjects:
89
(
CourantNo
....
MachNo
PecletNo
.....
forceCoeffsCompressible
forceCoeffsIncompressible
forcesCompressible
forcesIncompressible
...
streamFunction
streamlinesLine
streamlinesPatch
streamlinesPoints
streamlinesSphere
....
turbulenceIntensity
....
vorticity
```

# Data sampling

SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP

- The starting quantities that should be monitored are the so-called *residuals*
- Residuals monitoring allows to get a primary understanding of the convergence of the numerical solution
- In few words residuals are a measure of the local imbalance of a conserved variable in each control volume. This value should tend to zero as the solving procedure evolves
- If residuals 'diverges' the numerical solution is not reliable
- If residuals 'converges' then from the numerical point of view, we are ok, but the physics must be checked

# Data sampling

SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP

- Other quantities that should be monitored to have a correct understanding of the physical meaning of the solution are the values of computed quantities at boundaries
- Usually when for instance we are imposing velocity values at a given boundary we want to monitor the computed pressure value and vice-versa if we prescribe pressure.
- Other physically meaningful quantity for a given problem should be sampled, for instance forces acting on the Cylinder in the case of vortex shedding or other related to measurement campaign or published data.

# Plotting during/after calculations

SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP

Once data are sampled and monitored during calculation, we can get a plot of those quantities

- A quick way of doing is using the *foamMonitor* function:

```
Monitor data with Gnuplot from time-value(s) graphs written by OpenFOAM
e.g. by functionObjects
- requires gnuplot, gnuplot_x11

Example:
  foamMonitor -l postProcessing/residuals/0/residuals.dat
```

- In OF is equally simple sample planes, point probes, line probes and numbers of relevant quantities.
- In order to add monitored quantities in an easy way we can use the *foamGet* function

```
Finds an example OpenFOAM case dictionary file in /cineca/prod/opt/applications/openfoam/6.0/intelmpi--2018--binary/OpenFOAM-6.0/etc/caseDicts and
copies it into the respective case directory, e.g.

  foamGet decomposeParDict
  foamGet extrudeMeshDict
  foamGet createPatchDict
  foamGet surfaces
```

# Output Visualization

SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP

- The standard way to have a look at your solution is using the open-source viewer *paraFoam* that is and extension of the more well-known viewer Paraview developed by Kitware (https://www.kitware.com/).
- Despite a set of convenient features available in *paraFoam* to support the OF user in visualizing the mesh you can obtain the same kind of visualization using the standard *Paraview* viewer.
- I will refer to *Paraview* in the forthcoming.

# Output Visualization

SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP

- In order to read and visualize an OF case output we can either read the *view.foam* file, i.e.: reading the native OpenFOAM data format, or we can convert the output data into the standard VTK file format by means of the *foamToVTK* function.
- In my experience using the native file format is more convenient for meshing analysis while for flow field quantities there is no differences between the file format.

# Recap

SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP

## You'll work on galile100 cluster (g100)

- Intel based cluster (Xeon(R) Platinum 8260 CPU @ 2.40GHz)
- Each node has 2 GPU for a total of 48 core wit 385 GB RAM (FAT node 3TB optane)
- [https://www.hpc.cineca.it/hardware/galileo100](https://www.hpc.cineca.it/hardware/galileo100)

## G100 has a module-based environment

- `cd $WORK`
- `cd $CINECA_SCRATCH`

## G100 has a batch system based on slurm

- `sbatch`
- `squeue`
- `scancel`

```
ssh -X <username>@login.g100.cineca.it
```

# Implementation in OpenFOAM

SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP

- In your working space you will find the directory named:

*Case_Re200_IcoFoam/*

the main dictionary are modified and adapted from Wolf Dynamics set of 2D cylinder cases (see references at the end of this document)

For visualization using paraview move your vtk file locally via scp:

```
scp <user>@login.m100.cineca.it:<path>/<vtkfile>   .
```

# Implementation in OpenFOAM



1. Copy in your working space this file

```
/g100_work/tra22_SCtrain/OF_HANDS_ON/Case_Re200_IcoFoam.final.tgz
/g100_work/tra22_SCtrain/OF_HANDS_ON/Case_Re200_IcoFoam_par.final.tgz
```

1. Untar the file

```
tar -zcvf Case_Re200_IcoFoam.final.tgz
```

the main dictionary are modified and adapted from Wolf Dynamics set of 2D cylinder cases (see references at the end of this document)

For visualization using paraview move your vtk file locally via scp:

```
scp <user>@login.m100.cineca.it:<path>/<vtkfile>    .
```

# Implementation in OpenFOAM



```
├── 0
│   ├── Co
│   ├── p
│   └── U
├── clean.sh
├── constant
│   ├── physicalProperties
│   └── polyMesh
│       ├── boundary
│       ├── faces
│       ├── neighbour
│       ├── owner
│       └── points
├── run_solver.sh
└── system
    ├── blockMeshDict
    ├── controlDict
    ├── copy.controlDict
    ├── cutPlaneSurface
    ├── forceCoeffsIncompressible
    ├── forcesIncompressible
    ├── fvSchemes
    ├── fvSolution
    └── residuals
```

- Directory 0: time directory (initial condition)
- File: physicalProperties
- File: blockMeshDict
- File: controlDict
- File: fvSchemes
- File: fvSolutions
- Script: run_solver.sh

# run_solver.sh

SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP

```
#!/bin/bash

#load env
module purge
module load profile/eng autoload openfoam/10
# Source tutorial run functions
. $WM_PROJECT_DIR/bin/tools/RunFunctions
#cleanup
foamCleanTutorials
#meshing
runApplication blockMesh
runApplication checkMesh
#run solver
runApplication icoFoam
#logs & VTK
foamLog log.icoFoam > log.foamLog
foamToVTK > log.foamToVTK
```

Simple script

- Load correct environment
- Set-up simulation
  - Build mesh --> log.blockMesh
  - Check mesh --> log.checkMesh
- Run application
- Post processing
  - Extract info form log.icofoam
  - Produce vtk files

# blockMeshDict

SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP

```
vertices
(
    //back up
    (1 0 -0.5)                          //0
    (3 0 -0.5)                          //1
    (30 0 -0.5)                         //2
    (30 2.12132 -0.5)                   //3
    (2.12132 2.12132 -0.5)              //4
    (0.707107 0.707107 -0.5)            //5
    (30 20 -0.5)                        //6
    (2.12132 20 -0.5)                   //7
    (0 20 -0.5)                         //8
    (0 3 -0.5)                          //9
    (0 1 -0.5)                          //10
    (-1 0 -0.5)                         //11
    (-3 0 -0.5)                         //12
    (-20 0 -0.5)                        //13
    (-20 2.12132 -0.5)                  //14
    (-2.12132 2.12132 -0.5)             //15
    (-0.707107 0.707107 -0.5)           //16
    (-20 20 -0.5)                   //17
    (-2.12132 20 -0.5)                  //18

    //front up
    (1 0 0.5)                           //19
    (3 0 0.5)                           //20
    (30 0 0.5)                          //21
    (30 2.12132 0.5)                    //22
    (2.12132 2.12132 0.5)               //23
    (0.707107 0.707107 0.5)             //24
```

It controls the Mesh

- Vertices, blocks, edges
- boundaries

Do not modify

# fvSchemes

SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP

```
ddtSchemes
{
    //default          Euler;
    default          backward;
}

gradSchemes
{
    default           cellLimited leastSquares 1;
}

divSchemes
{
    default          none;
    div(phi,U)       Gauss linearUpwindV default;

laplacianSchemes
{
    default          Gauss linear limited 1;
}

interpolationSchemes
{
    default          linear;
}

snGradSchemes
{
    default          limited 1;
}
```

This dictionary controls the numerical Schemes used

e.g:

- Timed discretization scheme
- Gradient, divergence, laplacian, …
- Interpolation
- Other stuff....

Do not modify!

# fvSolutions

SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP

```
solvers
{
    p
    {
        solver           GAMG;
        tolerance        1e-6;
        relTol           0;
        smoother         GaussSeidel;
        nPreSweeps       0;
        nPostSweeps      2;
        cacheAgglomeration on;
        agglomerator     faceAreaPair;
        nCellsInCoarsestLevel 100;
        mergeLevels      1;
    }

    pFinal
    {
        $p;
        relTol          0;
    }

    U
    {
        solver          PBiCGStab;
        preconditioner  DILU;
        tolerance       1e-08;
        relTol          0;
    }
}

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

This dictionary controls solvers used

- You can choose different solvers
    - GAMG
    - PCGSTSM
    - …..
- Setting different resolutions, stopping criteria, min/max iterations and so on

# physicalProperties

SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP

```
/*--------------------------------*- C++ -*----------------------------------
  =========                 |
  \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox
   \\    /   O peration     | Website:  https://openfoam.org
    \\  /    A nd           | Version:  10
     \\/     M anipulation  |
\*---------------------------------------------------------------------------
FoamFile
{
    format      ascii;
    class       dictionary;
    location    "constant";
    object      physicalProperties;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

nu              [0 2 -1 0 0 0 0] 0.02;


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

- You can alter the viscosity (hence Re Number)

# controlDict

SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP

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

application     icoFoam;
startFrom       latestTime;
startTime       0;
stopAt          endTime;
endTime         1000;
deltaT          0.05;
writeControl    runTime;
writeInterval   100;
purgeWrite      0;
writeFormat     ascii;
writePrecision  8;
writeCompression off;
timeFormat      general;
timePrecision   6;
runTimeModifiable true;

// ***********************************************

functions
{
 #includeFunc CourantNo;
 #includeFunc residuals;
 #includeFunc forcesIncompressible;
 #includeFunc forceCoeffsIncompressible;
```

This dictionary controls all the simulation

- DeltaT
- I/O output
- Starting from
- Ending at
- Function to perform

# Implementation in OpenFOAM

- The mesh for the 2d case can be produced with efficiency and simplicity using the *blockMesh* utility
- By instructing the *blockMeshDict* dictionary file we can specify the necessary geometry components and spacing to obtain a fully hexahedral mesh
- Hexahedral meshes are very well suited for CFD since they show low non-hortogonality
- Using the *checkMesh* utility the user can verify the quality of the mesh
- Note that 2d mesh in OpenFOAM is a 3d mesh with 'thickness' e qual to 11 cell in one direction (z-dir (0 0 1) in our case)

# log.blockMesh

SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP

```
Create time

Reading "blockMeshDict"

Creating block mesh from
    "system/blockMeshDict"
Creating block edges
No non-planar block faces defined
Creating topology blocks
Creating topology patches

Creating block mesh topology

Check topology

        Basic statistics
                Number of internal faces : 28
                Number of boundary faces : 64
                Number of defined boundary faces : 64
                Number of undefined boundary faces : 0
        Checking patch -> block consistency

Creating block offsets
Creating merge list .

Creating polyMesh from blockMesh
Creating points with scale 1
    Block 0 cell size :
        i : 0.069170028 .. 0.13834006 0.069298242 .. 0.13859648 0.069170028 .. 0.
        j : 0.078519653 .. 0.078519653 0.083968924 .. 0.083968924 0.078519653 ..
24
        k : 1
```

```
Writing polyMesh
----------------
Mesh Information
----------------
  boundingBox: (-20 -20 -0.5) (30 20 0.5)
  nPoints: 18840
  nCells: 9200
  nFaces: 37020
  nInternalFaces: 18180
----------------
Patches
----------------
  patch 0 (start: 18180 size: 80) name: out
  patch 1 (start: 18260 size: 100) name: sym1
  patch 2 (start: 18360 size: 100) name: sym2
  patch 3 (start: 18460 size: 80) name: in
  patch 4 (start: 18540 size: 80) name: cylinder
  patch 5 (start: 18620 size: 9200) name: back
  patch 6 (start: 27820 size: 9200) name: front

End
```

# log.checkMesh

SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP

```
Mesh stats
    points:           18840
    internal points:  0
    faces:            37020
    internal faces:   18180
    cells:            9200
    faces per cell:   6
    boundary patches: 7
    point zones:      0
    face zones:       0
    cell zones:       0

Overall number of cells of each type:
    hexahedra:     9200
    prisms:        0
    wedges:        0
    pyramids:      0
    tet wedges:    0
    tetrahedra:    0
    polyhedra:     0

Checking topology...
    Boundary definition OK.
    Cell to face addressing OK.
    Point usage OK.
    Upper triangular ordering OK.
    Face vertices OK.
    Number of regions: 1 (OK).

Checking patch topology for multiply connected surfaces...
    Patch               Faces    Points   Surface topology
    out                 80       162      ok (non-closed singly connected)
    sym1                100      202      ok (non-closed singly connected)
    sym2                100      202      ok (non-closed singly connected)
    in                  80       162      ok (non-closed singly connected)
    cylinder            80       160      ok (non-closed singly connected)
    back                9200     9420     ok (non-closed singly connected)
    front               9200     9420     ok (non-closed singly connected)

Checking geometry...
    Overall domain bounding box (-20 -20 -0.5) (30 20 0.5)
    Mesh has 2 geometric (non-empty/wedge) directions (1 1 0)
    Mesh has 2 solution (non-empty) directions (1 1 0)
    All edges aligned with or perpendicular to non-empty directions.
    Boundary openness (1.0199166e-17 9.0703522e-18 8.2525597e-16) OK.
    Max cell openness = 2.1379307e-16 OK.
    Max aspect ratio = 6.8851226 OK.
    Minimum face area = 0.0056205169. Maximum face area = 1.764025.  Face area magnitudes OK.
    Min volume = 0.0056205169. Max volume = 1.764025.  Total volume = 1996.8616.  Cell volumes OK.
    Mesh non-orthogonality Max: 43.433983 average: 10.350523
    Non-orthogonality check OK.
    Face pyramids OK.
    Max skewness = 0.45511205 OK.
    Coupled point location match (average 0) OK.

Mesh OK.
```

# Implementation in OpenFOAM

SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP

- The solver *icoFoam* is transient incompressible solver
- The solver uses the *PISO* algorithm to solve the continuity equation and momentum equation
- The code is inherently transient, requiring an initial condition (such as zero velocity) and boundary conditions. The *icoFoam* solver can manage mesh non-orthogonality with successive non-orthogonality iterations.
- The number of PISO corrections and non-orthogonality corrections are controlled through user input in the *fvSolution* dictionary.

# log.icoFoam

SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP

```
Time = 44.55s

Courant Number mean: 0.10794366 max: 0.72004011
DILUPBiCGStab:  Solving for Ux, Initial residual = 0.00015634709, Final residual = 8.8823207e-10, No Iterations 1
DILUPBiCGStab:  Solving for Uy, Initial residual = 0.00041906609, Final residual = 8.746105e-10, No Iterations 1
GAMG:  Solving for p, Initial residual = 0.00063780924, Final residual = 7.0209122e-07, No Iterations 8
GAMG:  Solving for p, Initial residual = 0.00020047284, Final residual = 5.0431901e-07, No Iterations 7
GAMG:  Solving for p, Initial residual = 6.1373838e-05, Final residual = 7.4468901e-07, No Iterations 4
time step continuity errors : sum local = 1.483304e-11, global = -1.825935e-13, cumulative = 2.1311232e-10
GAMG:  Solving for p, Initial residual = 7.4222239e-05, Final residual = 8.2505545e-07, No Iterations 4
GAMG:  Solving for p, Initial residual = 2.3688108e-05, Final residual = 5.7740399e-07, No Iterations 4
GAMG:  Solving for p, Initial residual = 6.7073077e-06, Final residual = 4.3560485e-07, No Iterations 2
time step continuity errors : sum local = 8.6766123e-12, global = -7.8788058e-14, cumulative = 2.1303353e-10
ExecutionTime = 44.928231 s  ClockTime = 50 s

Time = 44.6s

Courant Number mean: 0.10794476 max: 0.72001149
DILUPBiCGStab:  Solving for Ux, Initial residual = 0.00015616353, Final residual = 8.833415e-10, No Iterations 1
DILUPBiCGStab:  Solving for Uy, Initial residual = 0.00041861057, Final residual = 8.7001898e-10, No Iterations 1
GAMG:  Solving for p, Initial residual = 0.00056688804, Final residual = 7.6265588e-07, No Iterations 8
GAMG:  Solving for p, Initial residual = 0.00020313214, Final residual = 5.1936215e-07, No Iterations 7
GAMG:  Solving for p, Initial residual = 6.1602802e-05, Final residual = 7.4539767e-07, No Iterations 4
time step continuity errors : sum local = 1.4843943e-11, global = -2.1508481e-13, cumulative = 2.1281845e-10
GAMG:  Solving for p, Initial residual = 6.6743752e-05, Final residual = 7.6728688e-07, No Iterations 4
GAMG:  Solving for p, Initial residual = 2.3907559e-05, Final residual = 5.9732614e-07, No Iterations 4
GAMG:  Solving for p, Initial residual = 6.7401699e-06, Final residual = 4.3778735e-07, No Iterations 2
time step continuity errors : sum local = 8.7182041e-12, global = -5.5905266e-14, cumulative = 2.1276254e-10
ExecutionTime = 44.972376 s  ClockTime = 50 s
```

# Using the **`foamMonitor`** function

SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP

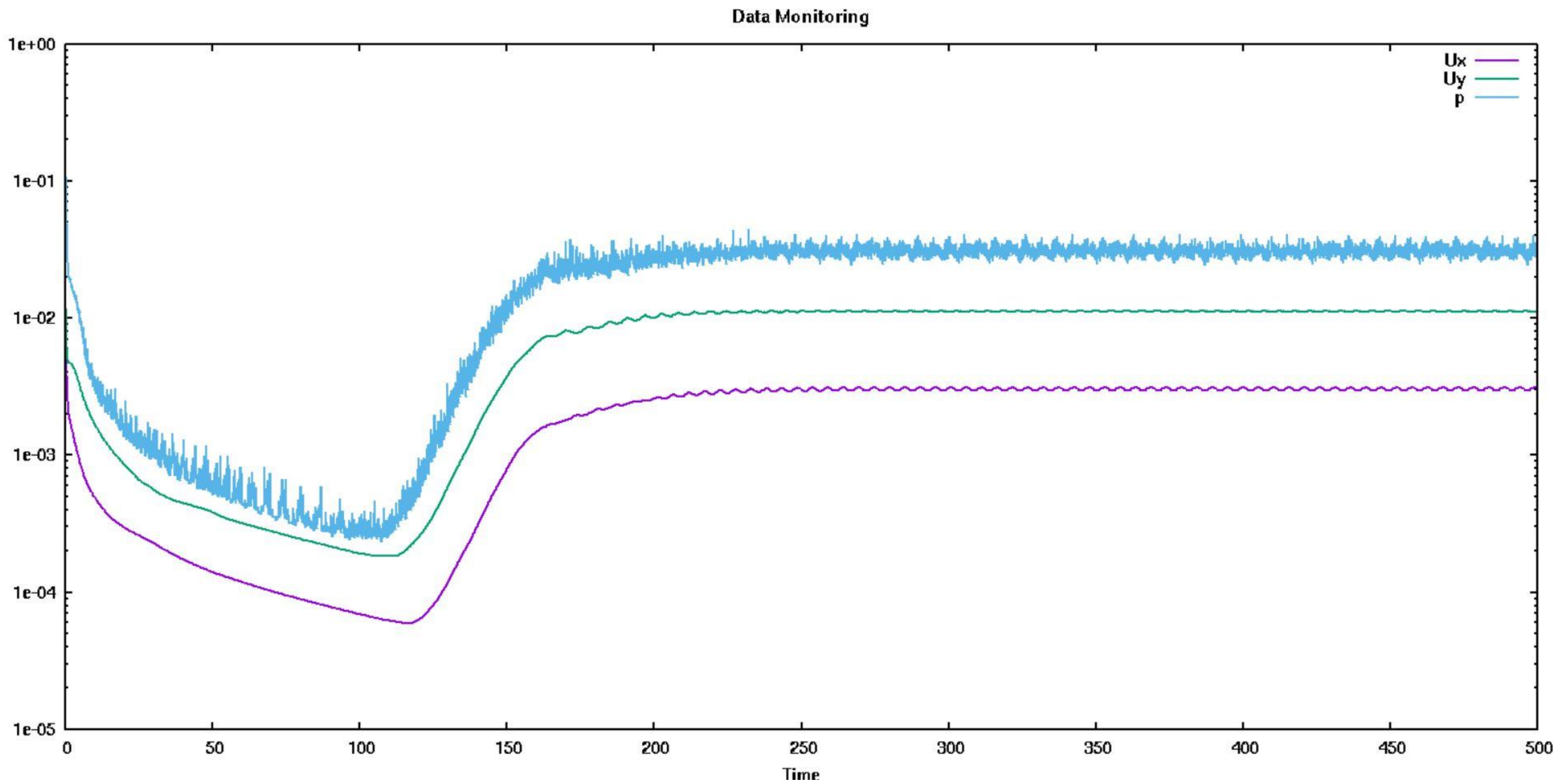
- As introduced above thanks to the *foamMonitor* function we can plot quantities using the gnuplot graphic engine
- In what follows we will ser this function in action by means of these commands:

```
foamMonitor postProcessing/residuals/0/residuals.dat
foamMonitor postProcessing/forcesIncompressible/0/forces.dat
foamMonitor postProcessing/forceCoeffsIncompressible/0/forceCoeffs.dat
```

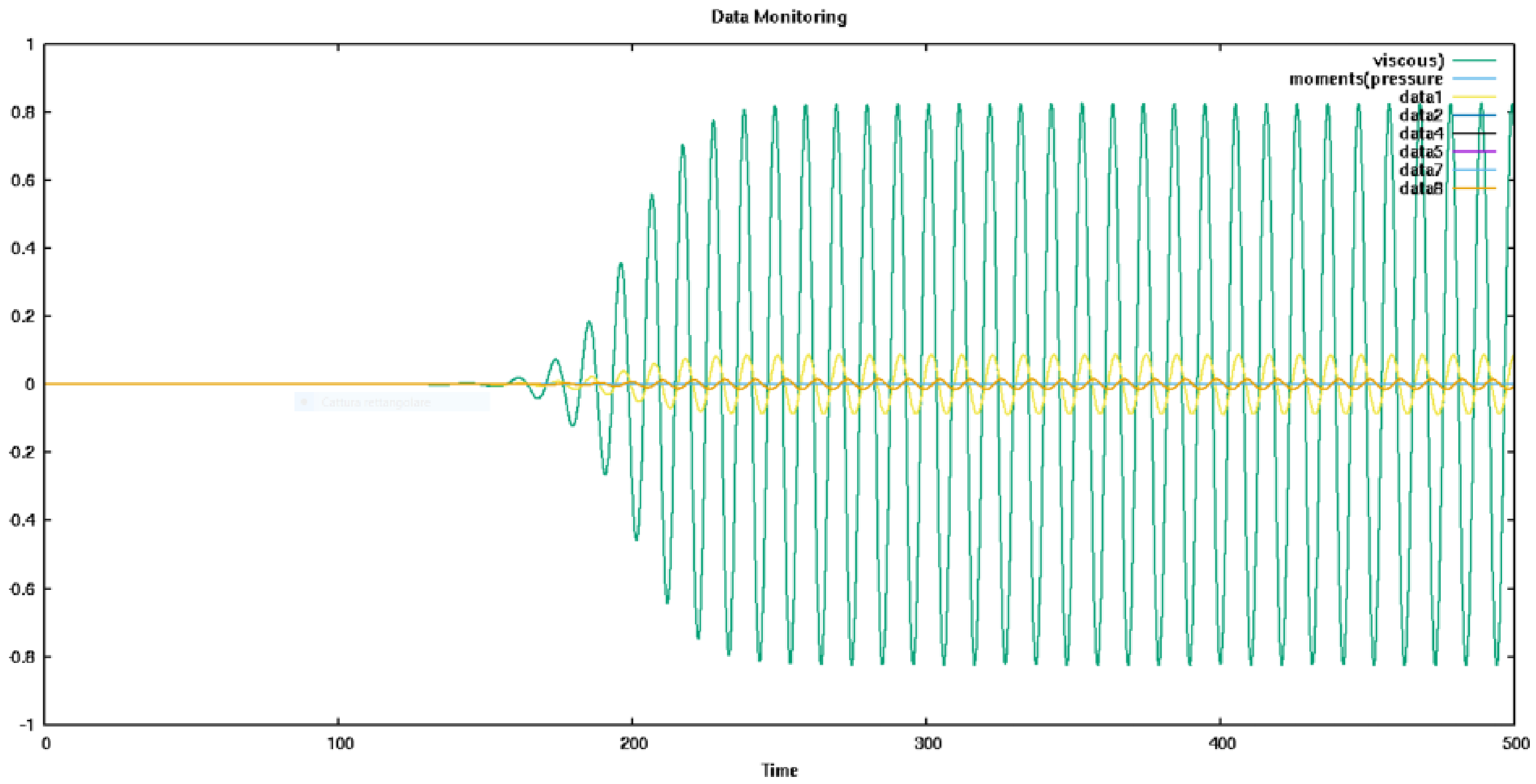
# Implementation in OpenFOAM

SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP

```
foamMonitor postProcessing/residuals/0/residuals.dat
```

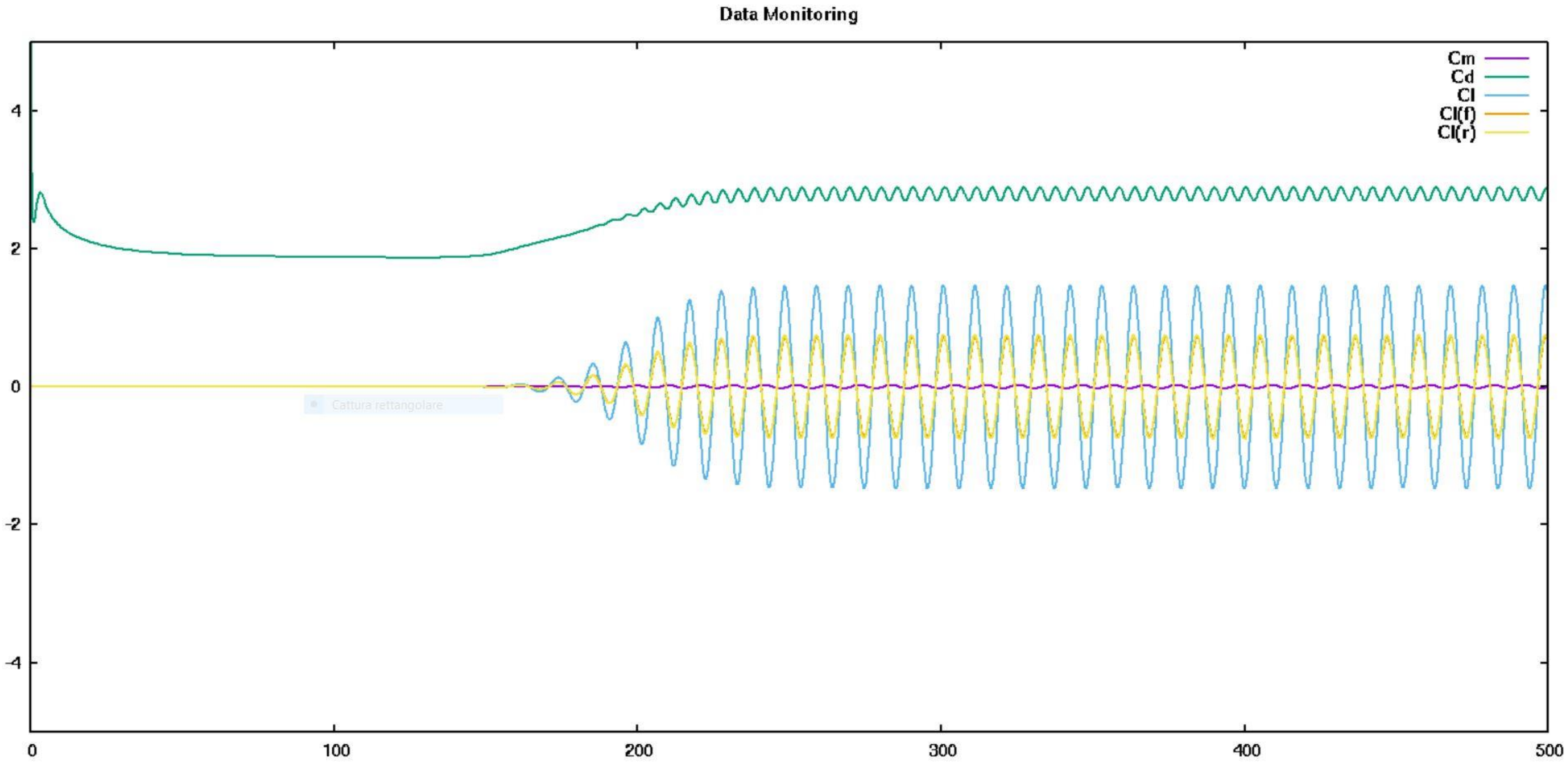


```
foamMonitor postProcessing/forcesIncompressible/0/forces.dat
```



SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP

**foamMonitor postProcessing/forceCoeffsIncompressible/0/forceCoeffs.dat**



# Implementation in OpenFOAM

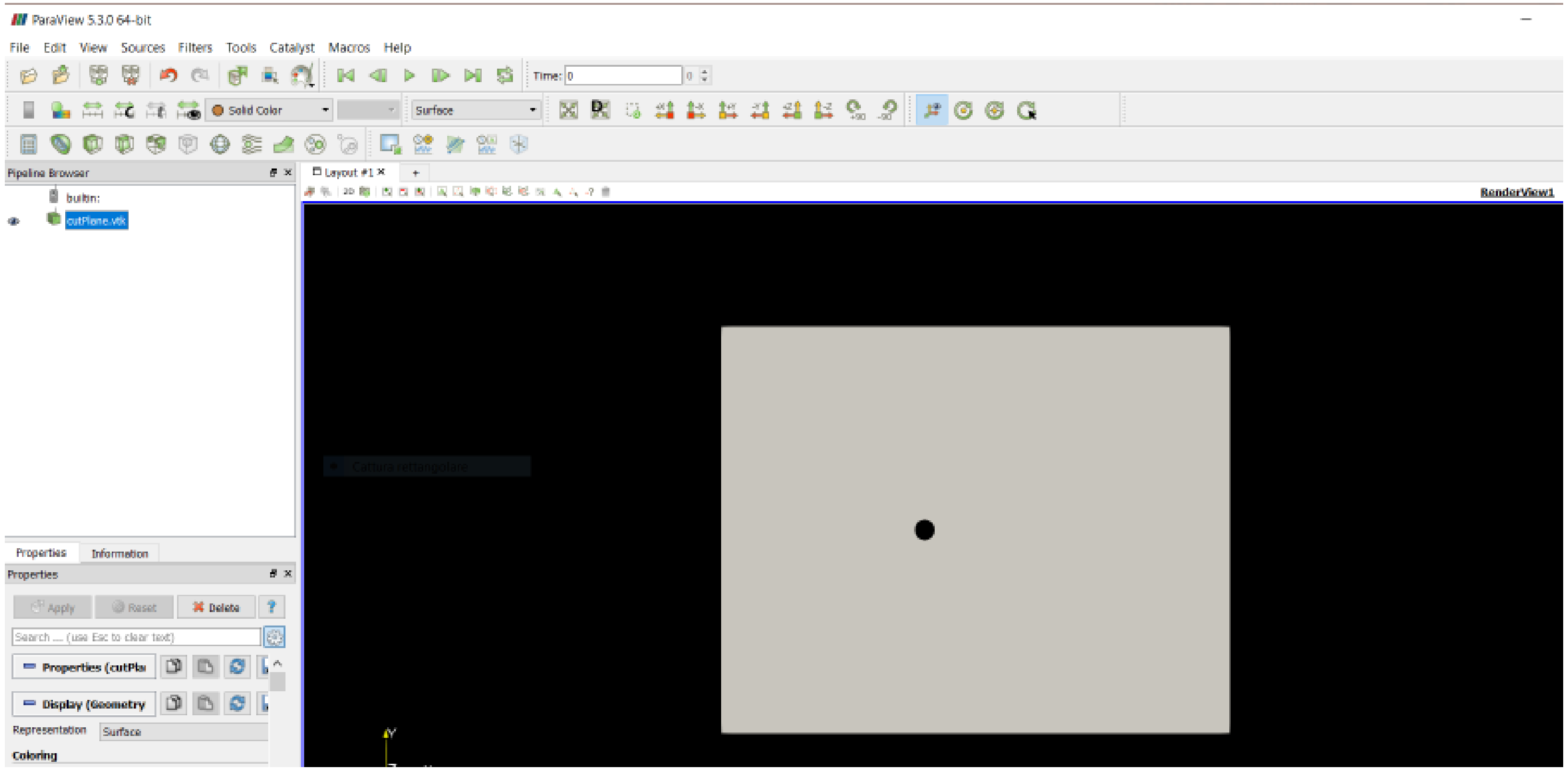
Open a VTK file in Paraview:

**File/Open/**

*(then select your vtk file (s))*

*Or*

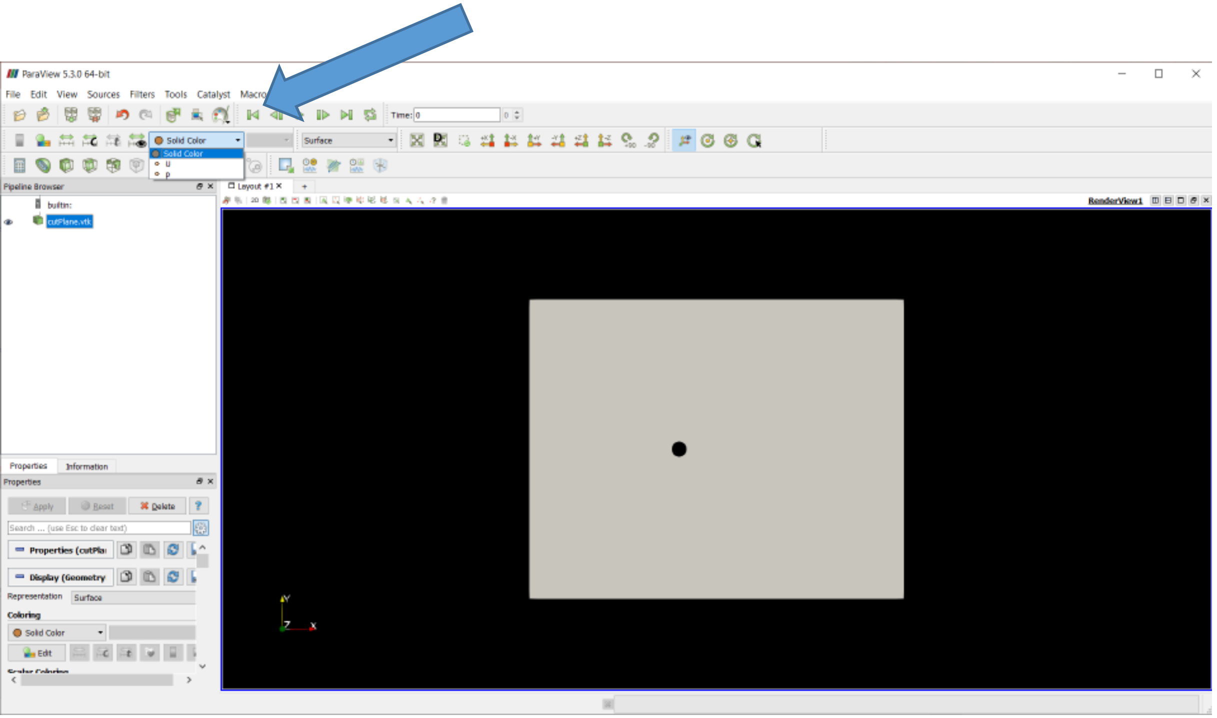
*(then select your .foam file)*



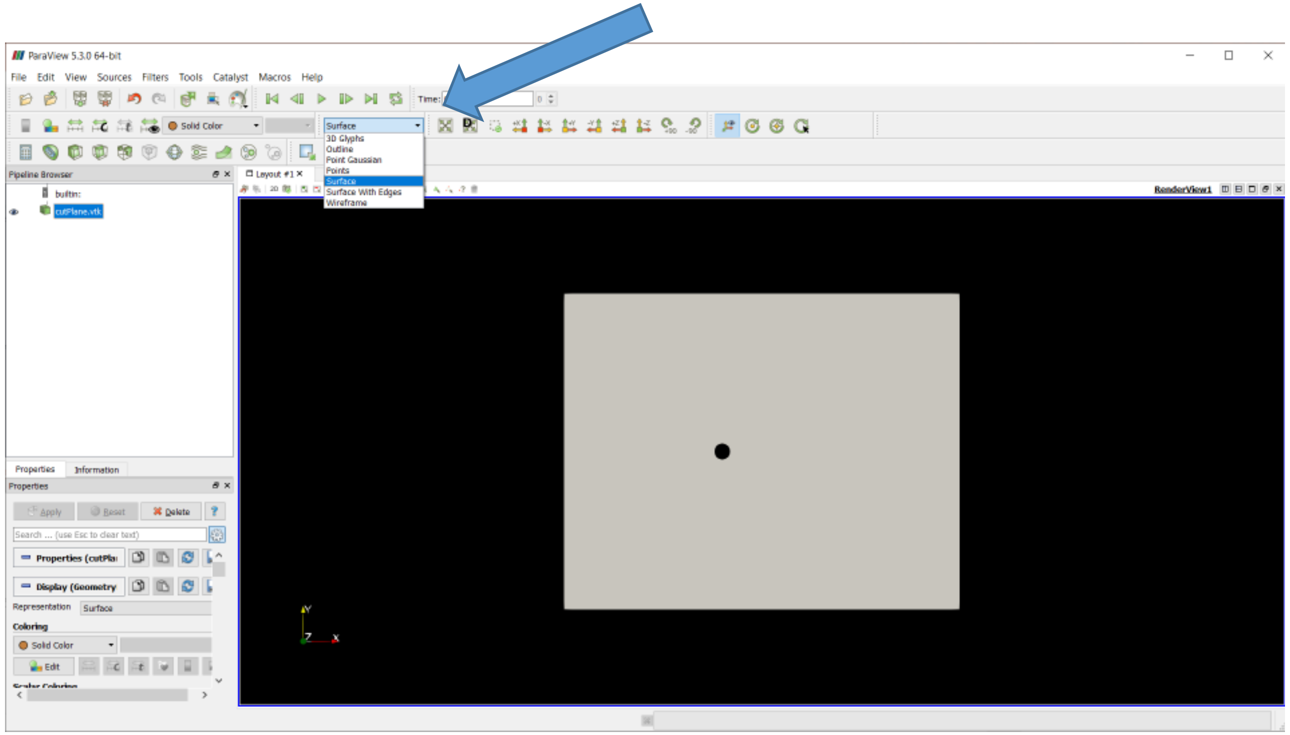
# Implementation in OpenFOAM

SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP

- Select the desired field to be visualized

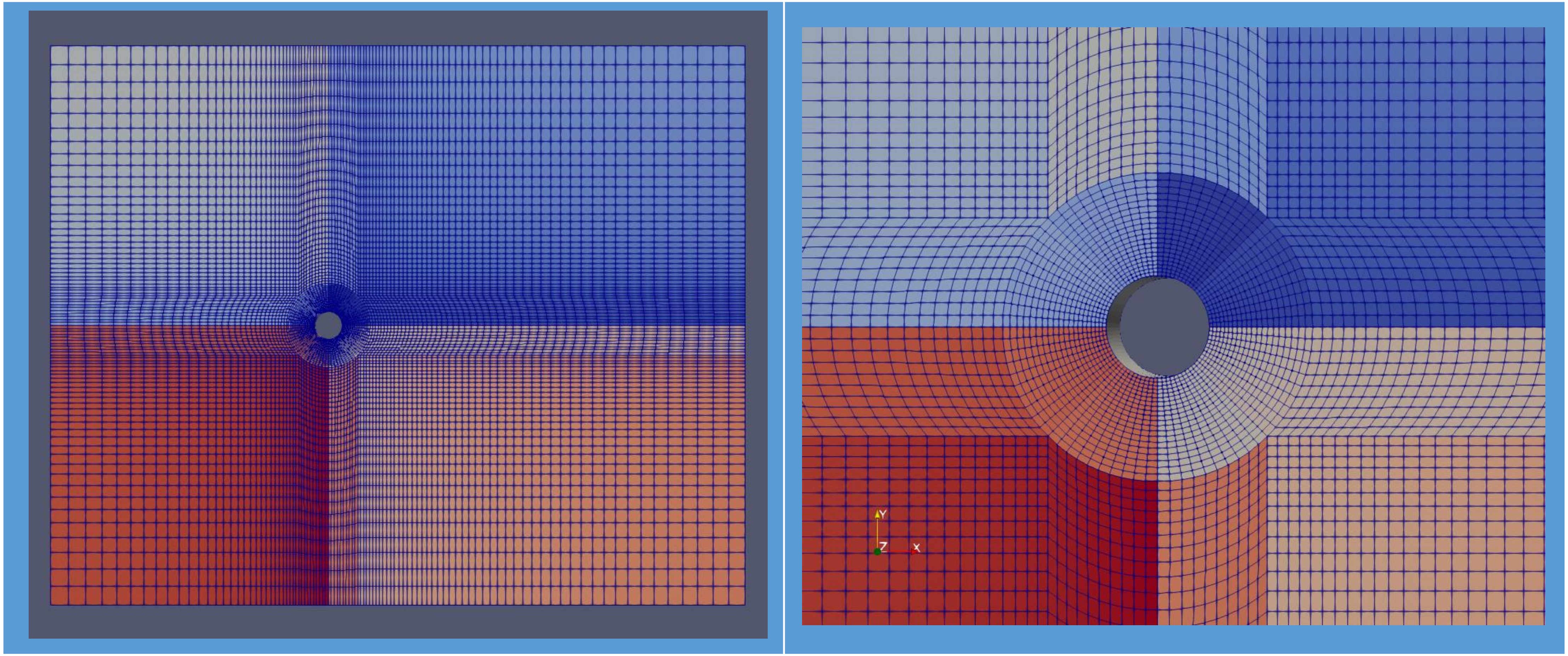


- Select the desired type of visualization (surface)



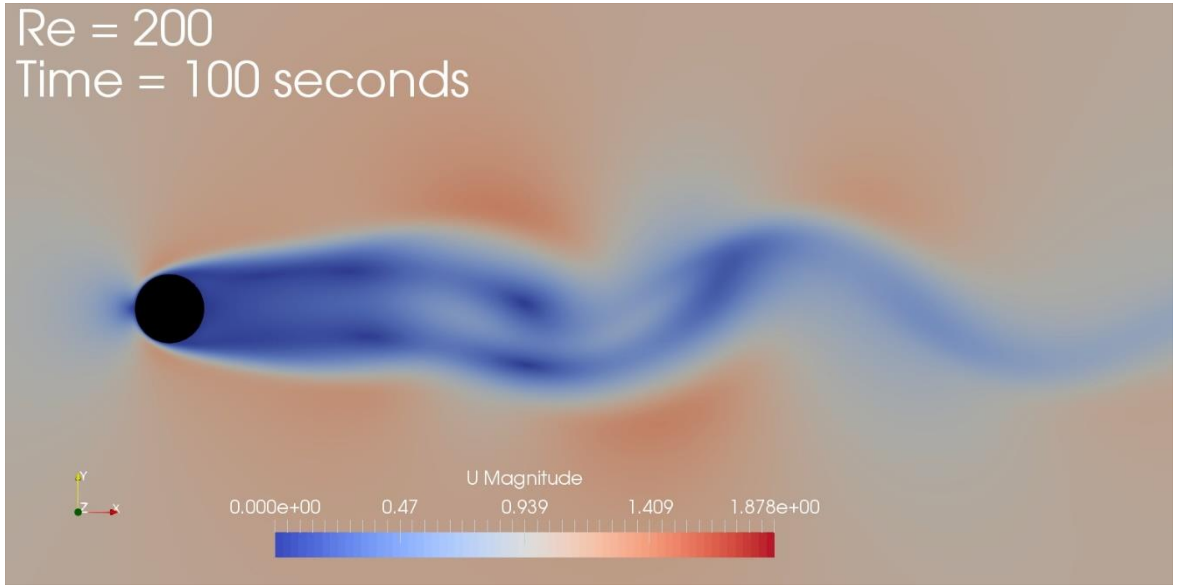
# Resulting grid

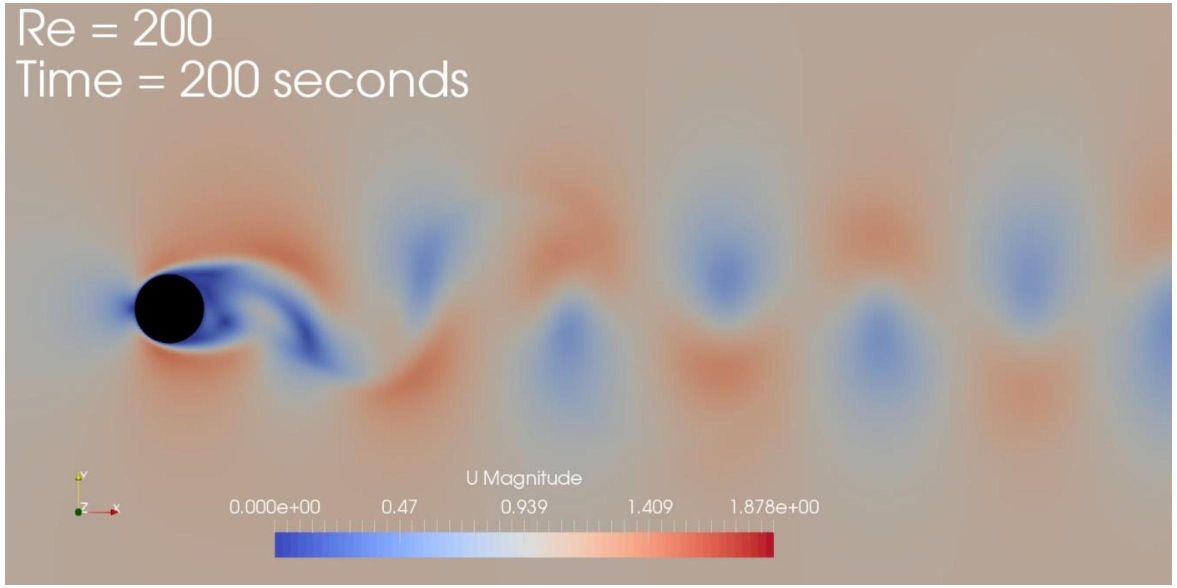
SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP



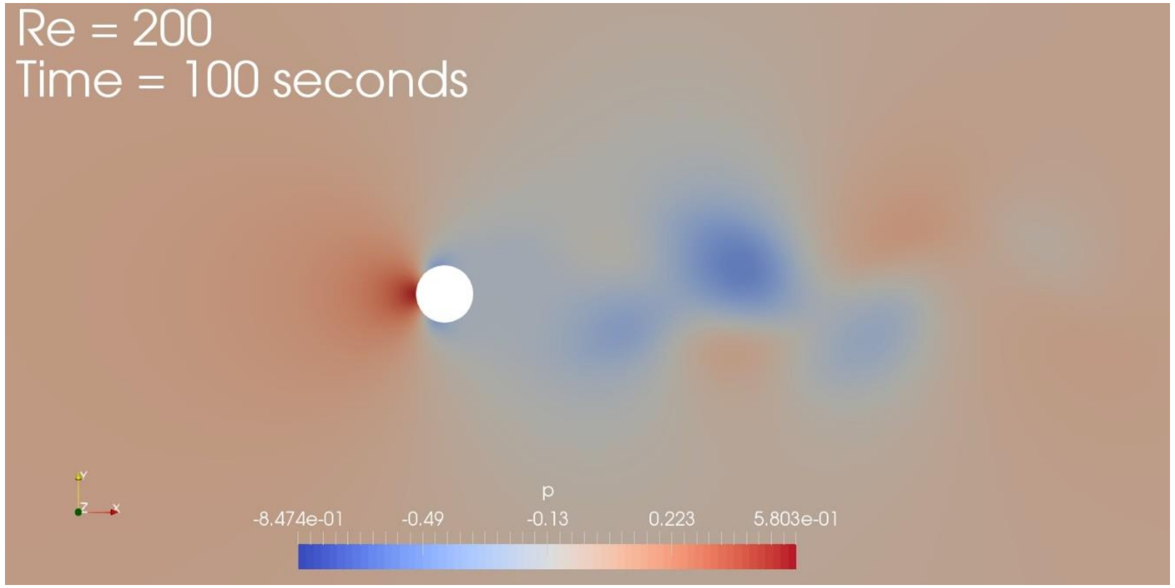
# Implementation in OpenFOAM

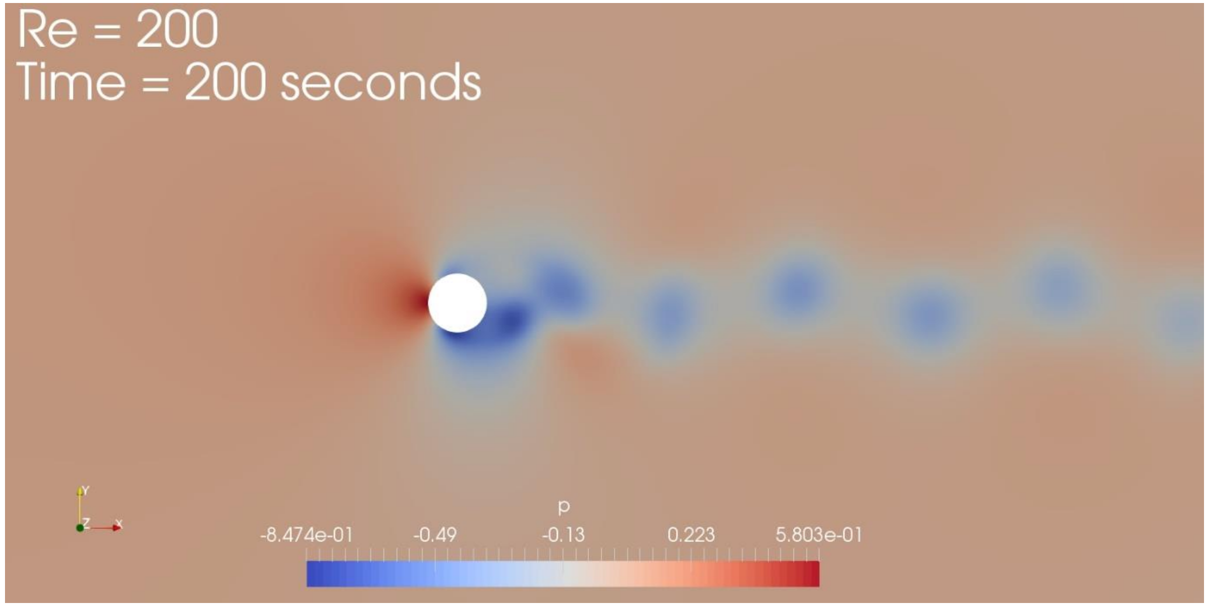
- U field over time



- P field over time



# Hands-on: howto

SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP

To run vortex shedding we will:

1. Connect to the HPC cluster:
   ```
   ssh –X login.g100.cineca.it
   ```
2. Request a job in interactive mode including graphic X11*:*
   ```
   srun --x11 -N1 --ntasks-per-node=1 -A tra22_Sctrain -p g100_usr_prod
   --reservation=s_tra_sc1 (--reservation=s_tra_sc2) --time=1:30:00 --pty /bin/bash
   ```
3. Load the required modules:
   OpenFOAM v10
   Gnuplot (for *foamMonitor* purposes)
   ```
   module load profile/eng autoload openfoam/10 gnuplot
   ```
4. Go into the test case directory
5. Run the workflow and other commands

# Hands-on suggested activities

SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP

We suggest to:

1. try to obtain similar results with the existing templates by running the workflow and thus taking confidence with:
   - OpenFOAM dictionaries (change the data sampling value)
   - 1D Data plotting
   - Paraview 2D data plotting and animation
2. Modify the test case by changing the ***0/U*** dictionary (for instance) to decrease the Reynolds number to 30 and re-run the case (please note the differences (no shedding))
3. Modify the test case increasing the deltaT up to crashing the solver (comment why?)

# foamLog



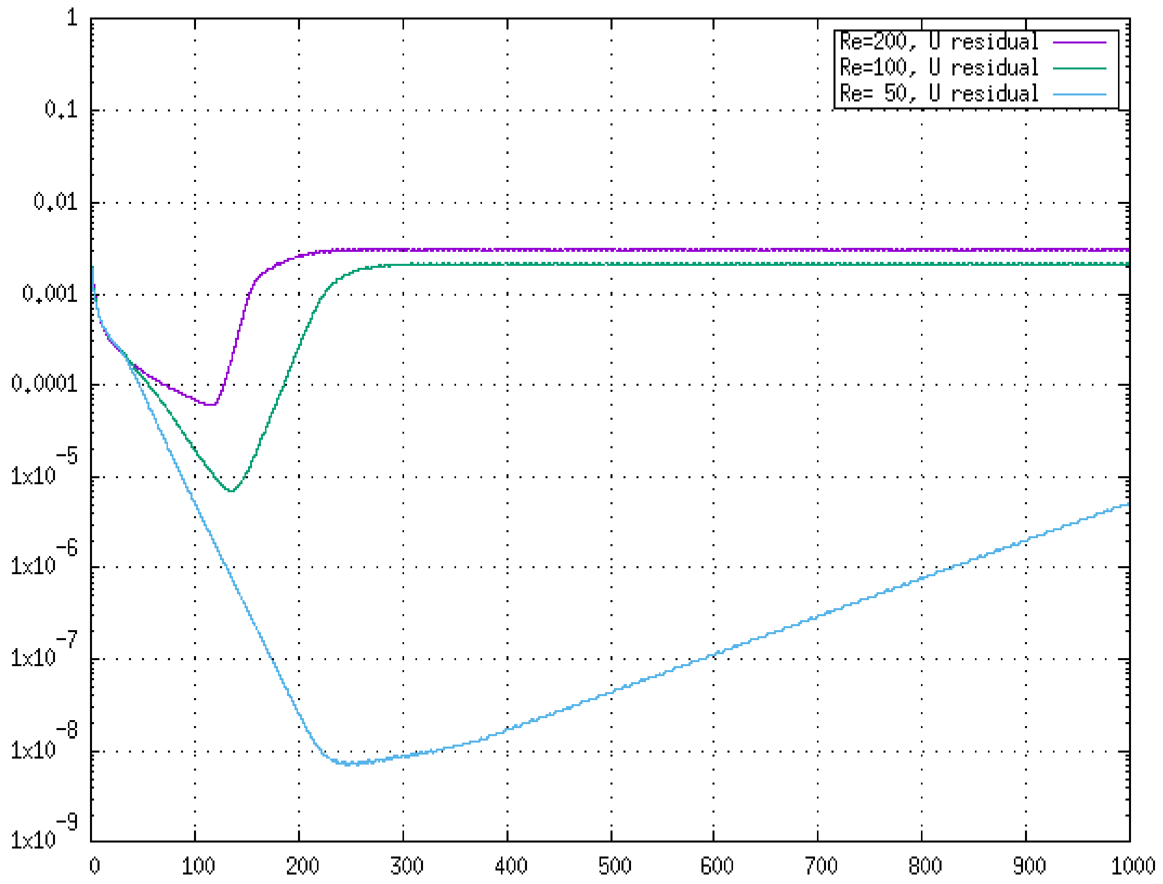
```
Executing: awk -f ./logs/foamLog.awk log.icoFoam

Generated XY files for:
clockTime
contCumulative
contGlobal
contLocal
CourantMax
CourantMean
executionTime
p
pFinalRes
pIters
Separator
Time
Ux
UxFinalRes
UxIters
Uy
UyFinalRes
UyIters
End
```

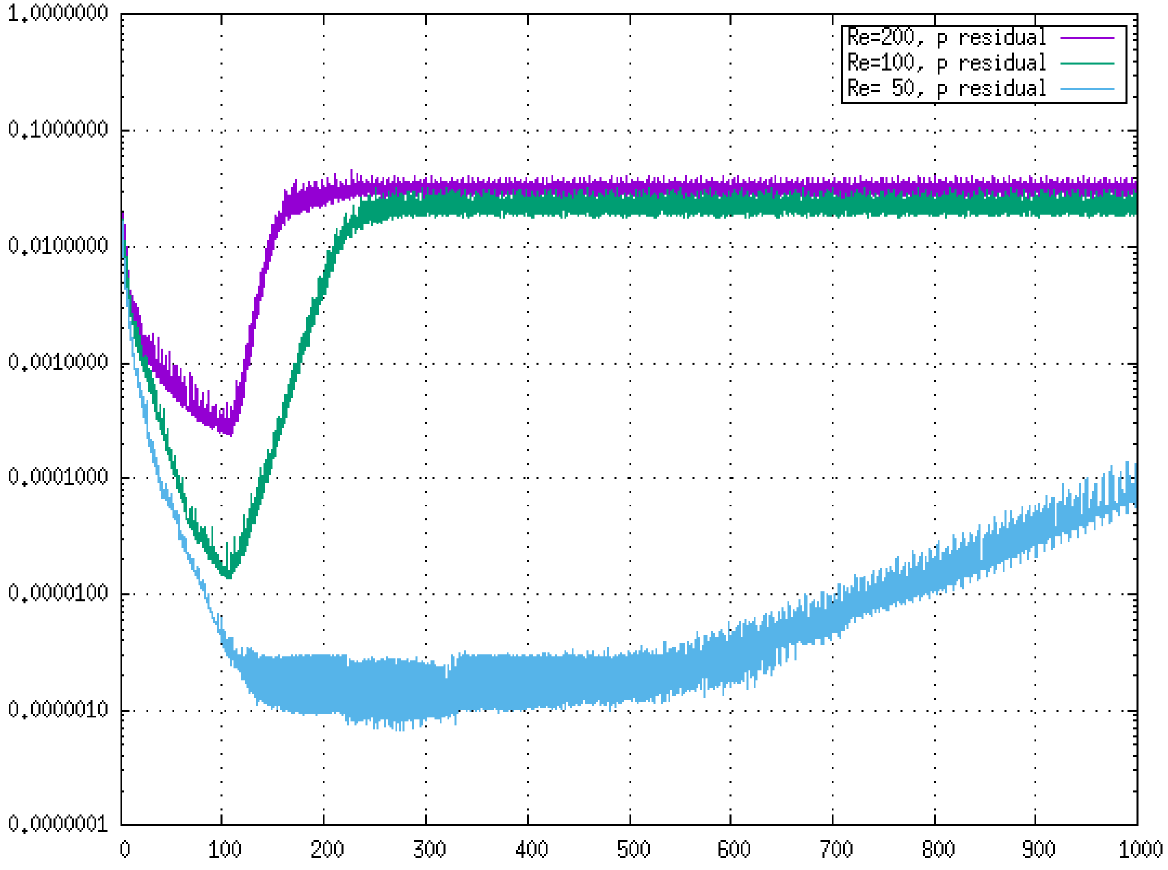
Extracts for info from log file, usually very verbose....

- Residuals
- Clocktime
- Courant Numbers
- Solver's iterations and so on...

# Residuals

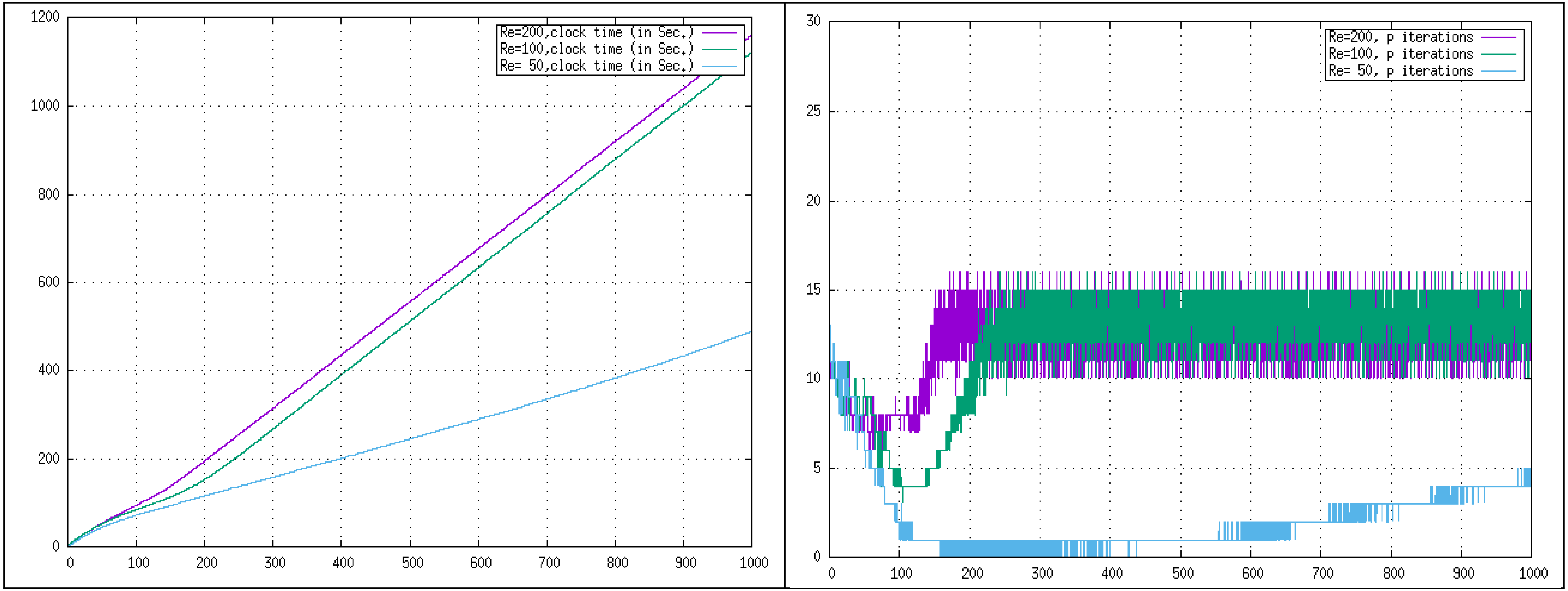
SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP



# Clocktime





Re=200,clock time (in Sec.)
Re=100,clock time (in Sec.)
Re= 50,clock time (in Sec.)
0
200
400
600
800
1000
1200
0
100
200
300
400
500
600
700
800
900
1000
Re=200, p iterations
Re=100, p iterations
Re= 50, p iterations
0
5
10
15
20
25
30
0
100
200
300
400
500
600
700
800
900
1000

# postProcess

SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP

- **postProcess -func streamFunction**
- **postProcess -func vorticity**

```
drwxr-xr-x 2 gamati01 interactive   4096  8 set 11.58 uniform
-rw-r--r-- 1 gamati01 interactive 351785  8 set 11.58 U_0
-rw-r--r-- 1 gamati01 interactive 351909  8 set 11.58 U
-rw-r--r-- 1 gamati01 interactive 217708  8 set 11.58 phi_0
-rw-r--r-- 1 gamati01 interactive 217586  8 set 11.58 phi
-rw-r--r-- 1 gamati01 interactive 114970  8 set 11.58 p
-rw-r--r-- 1 gamati01 interactive 104563  8 set 11.58 Co
-rw-r--r-- 1 gamati01 interactive 205139  8 set 17.07 streamFunction
-rw-r--r-- 1 gamati01 interactive 414995  8 set 17.08 vorticity
```

# FoamToVTK



```
VTK/
├── back
├── cylinder
├── front
├── in
├── out
├── PART_1_0.vtk
├── PART_1_10000.vtk
├── PART_1_12000.vtk
├── PART_1_14000.vtk
├── PART_1_16000.vtk
├── PART_1_18000.vtk
├── PART_1_20000.vtk
├── PART_1_2000.vtk
├── PART_1_22000.vtk
├── PART_1_24000.vtk
├── PART_1_26000.vtk
├── PART_1_28000.vtk
├── PART_1_30000.vtk
├── PART_1_4000.vtk
├── PART_1_6000.vtk
├── PART_1_8000.vtk
├── sym1
└── sym2
```

A VTK directory is created with:

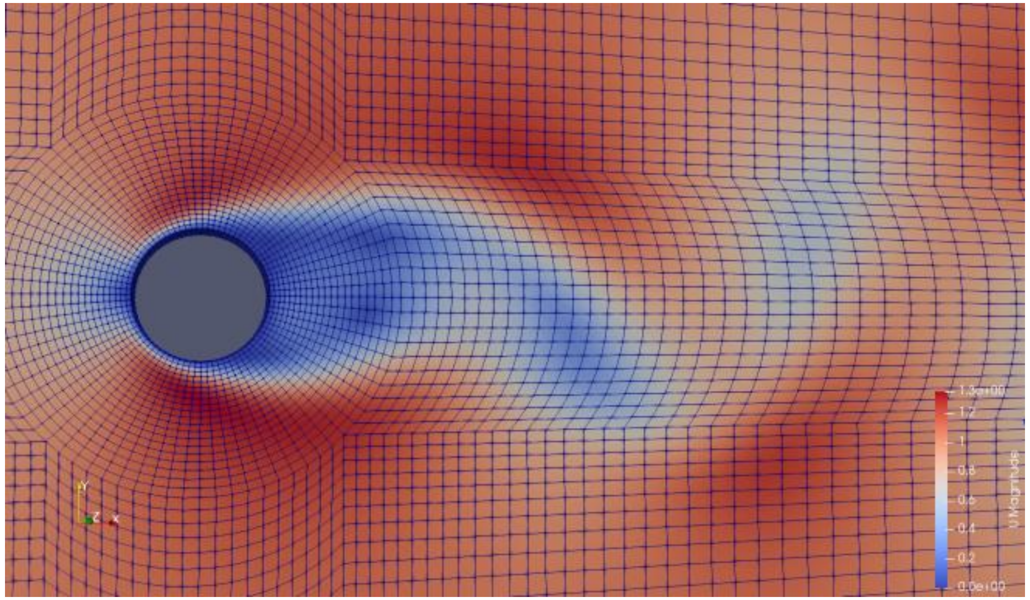
**`<dir_name>_<index>.vtk files`**

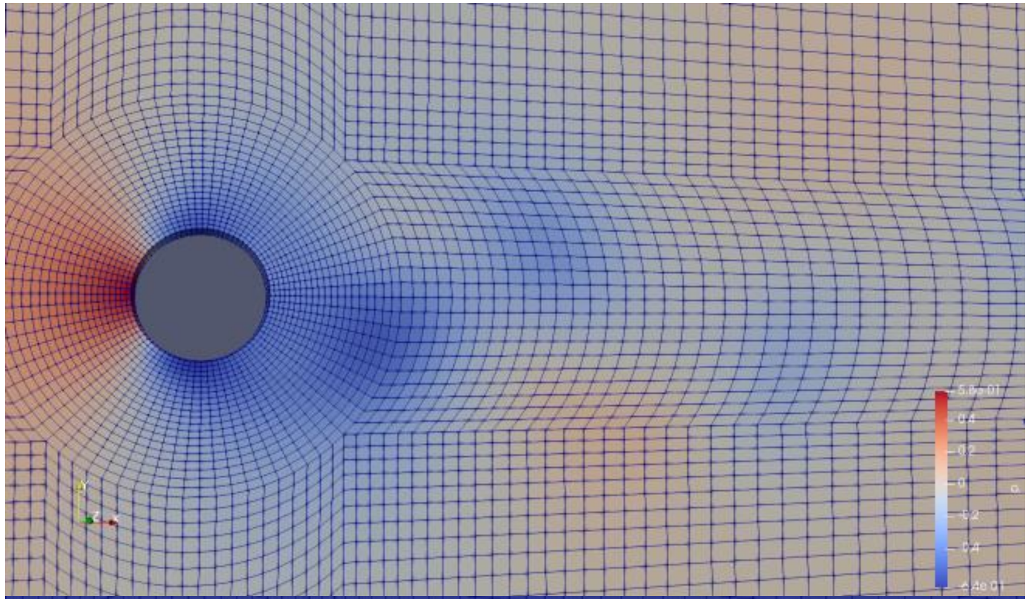
- <index> is the iteration, not the time!

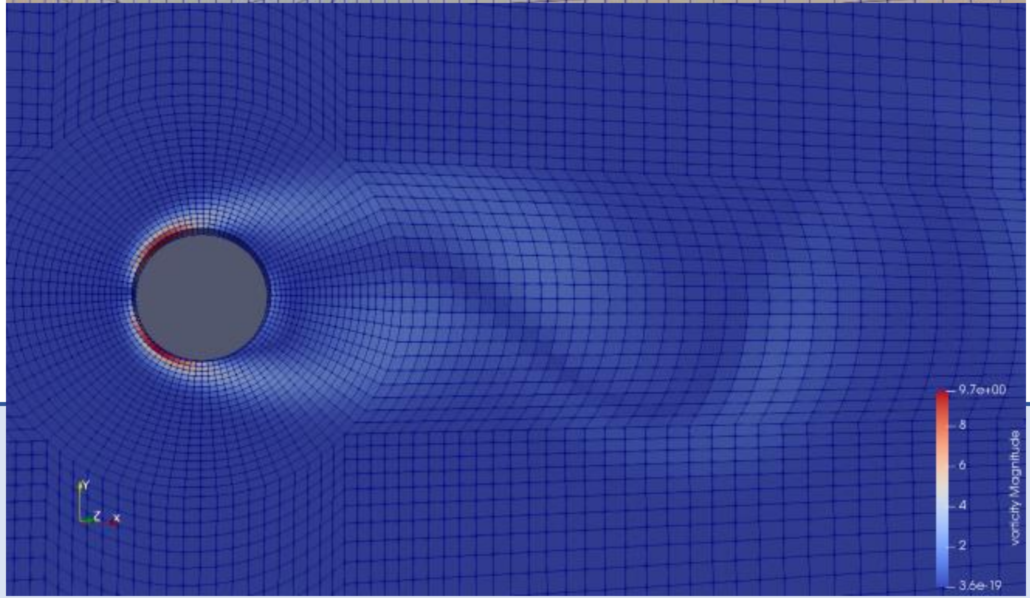
# Some pictures...

SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP

- Pressure, velocity magnitude, vorticity







# Resources and references



- CAD modeler open-source/free:
  - Freecad: https://www.freecadweb.org/
  - Salome: https://www.salome-platform.org/
  - Blender: https://www.blender.org/
  - Onshape: https://www.onshape.com/en/products/free
- OpenFOAM Official Material:
  - User Guide: https://cfd.direct/openfoam/user-guide/
  - Download: https://cfd.direct/openfoam/download/

SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP

- References:
  - Case 2d cylinder laminar: adapted and updated to v10 from Wolf Dynamics (http://www.wolfdynamics.com/) public tutorial material: http://www.wolfdynamics.com/wiki/vortex_shedding.tar.gz

# Thank you for your attention!

http://sctrain.eu/

Univerza v Ljubljani
TECHNISCHE UNIVERSITÄT WIEN
CINECA
VSB TECHNICAL UNIVERSITY OF OSTRAVA | IT4INNOVATIONS NATIONAL SUPERCOMPUTING CENTER

Co-funded by the Erasmus+ Programme of the European Union

This project has been funded with support from the European Commission.
This publication [communication] reflects the views only of the author, and the Commission cannot be held responsible for anyuse which may be made of the information contained therein.