

## Introduction to OpenFOAM® Computational Library and Viscoelastic Fluid Flow Simulation

# P1 - Introduction do OpenFOAM

**J. Miguel Nóbrega**

[mnobrega@dep.uminho.pt](mailto:mnobrega@dep.uminho.pt)

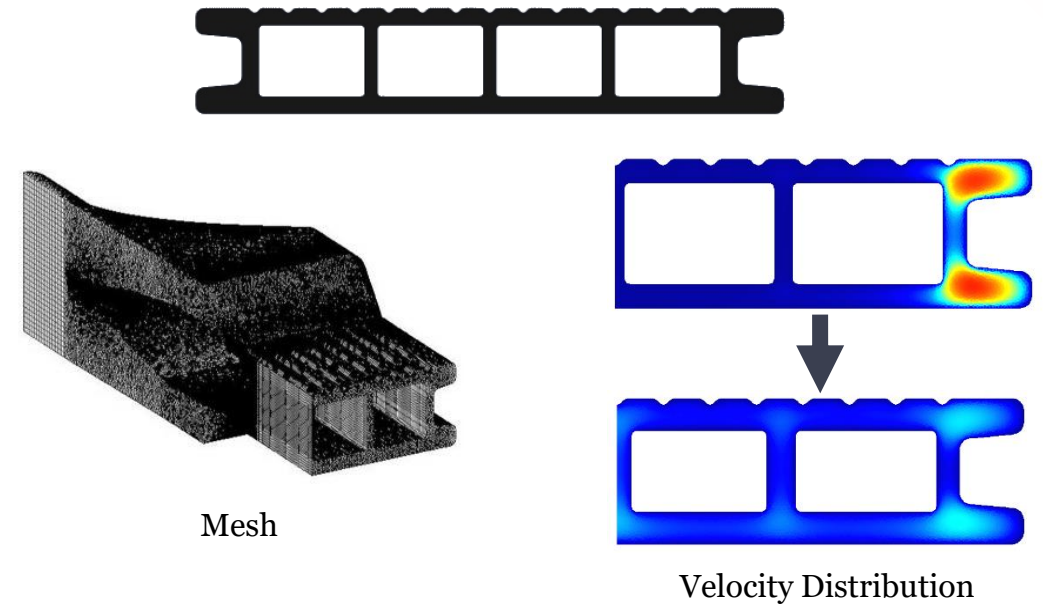
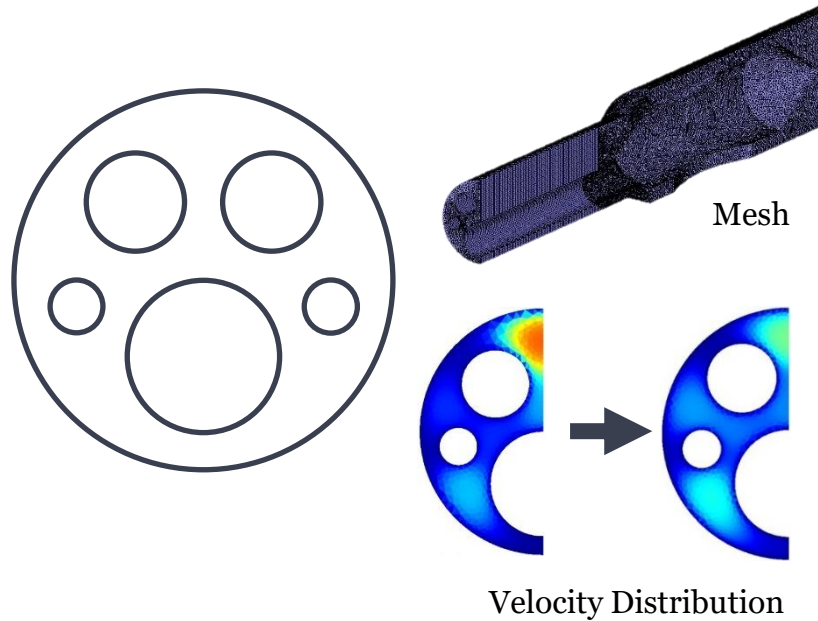
# Outline

9:00 – 10:30	<b>Introduction to OpenFOAM (P1)</b>
10:30 – 12:00	<b>Mesh generation and post-processing (P2)</b>
12:00 – 13:00	Lunch break
13:00 – 14:30	<b>Case studies: Single- and two-phase flow solvers (P3)</b>
14:30 – 16:00	<b>Case studies: Viscoelastic fluid flow solvers (P4)</b>

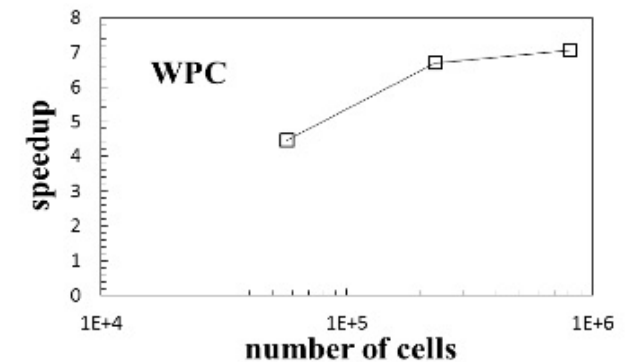
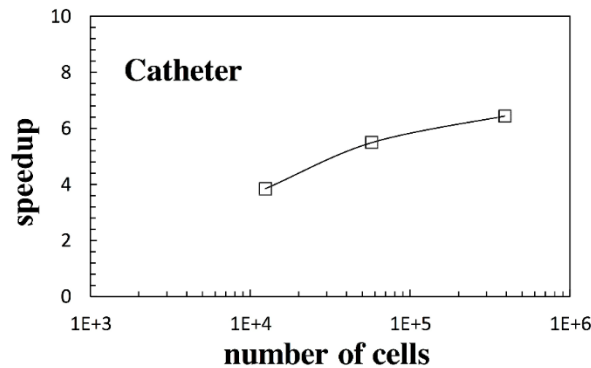


# Motivation

## Unstructured Numerical Modeling Code



## GPU Parallelization



# Motivation

**New feature → New researcher**



Hussaini Hanging Bridge, Pakistan



# The OpenFOAM® Computational Library

## Open▽FOAM

- **Open** source **F**ield **O**peration **A**nd **M**anipulation
  - C++ Computational Library, Finite Volume Method, Unstructured meshes (and mesh generators)
  - Multiphysics and Multiphase systems (FSI, Eulerian-Eulerian, Eulerian-Lagrangian)
  - Parallelized
  - Several **pre-compiled** solvers available
- ‘Basic’ CFD codes**  
**Incompressible flow**  
**Compressible flow**  
**Multiphase flow**  
**Large eddy simulation (LES)**  
**Combustion**  
**Particle-tracking flows**  
**Heat transfer**  
**Buoyancy-driven flows**  
**Molecular dynamics methods**  
**Direct simulation Monte Carlo methods**  
**Electromagnetics**  
**Solid Mechanics**  
**Viscoelastic**  
**Finance**  
...





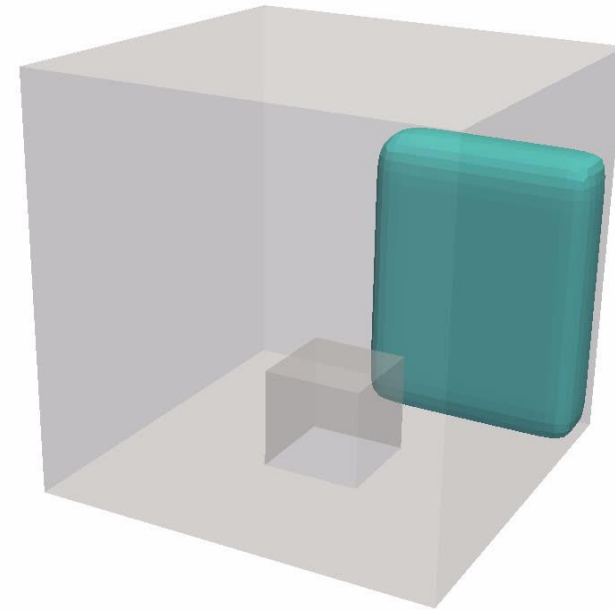
# The OpenFOAM® Computational Library

Open▽FOAM

## Fluid Dynamics



**Fluidized Bed**  
(Eulerian+Lagrangian)



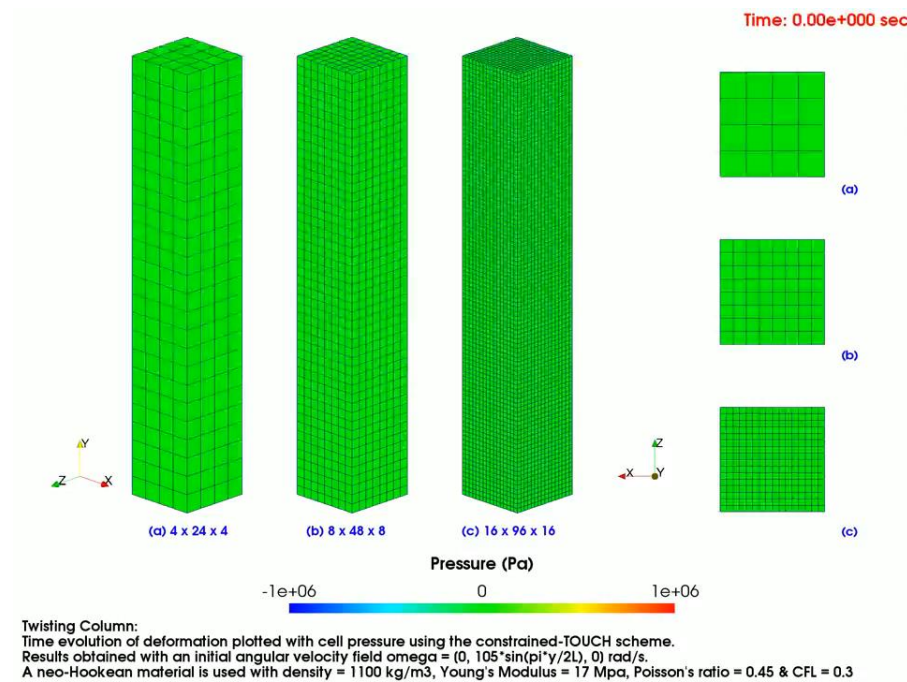
**Dam Break 3D - VOF**  
(Eulerian+Eulerian)



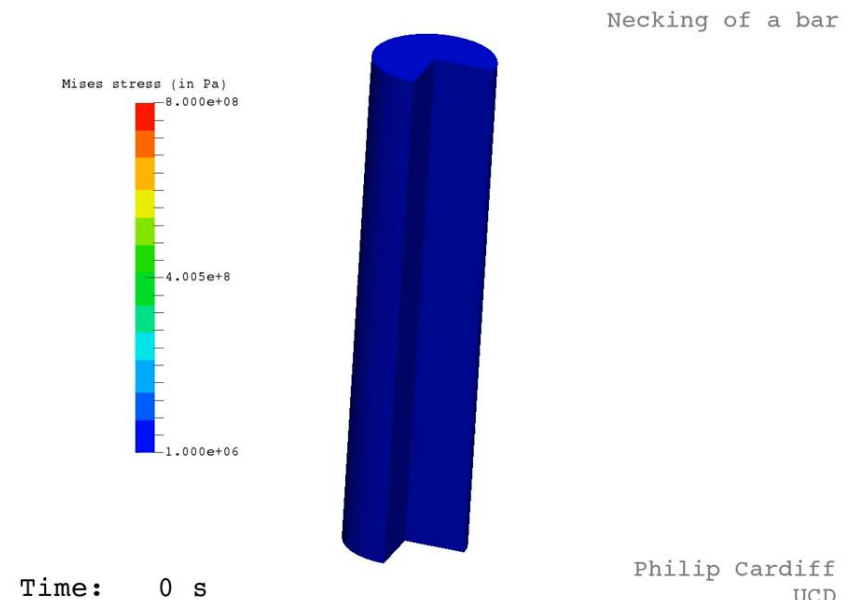
# The OpenFOAM® Computational Library

OpenFOAM

## Solid Mechanics



Twisting Column<sup>(1)</sup>



Necking of a bar <sup>(2)</sup>

- (1) From Jibran Haider's Youtube channel - <https://tinyurl.com/ybosqbtq>  
(2) From Philip Cardiff's Youtube channel - <https://tinyurl.com/pcardiff>



## A (very) brief history...

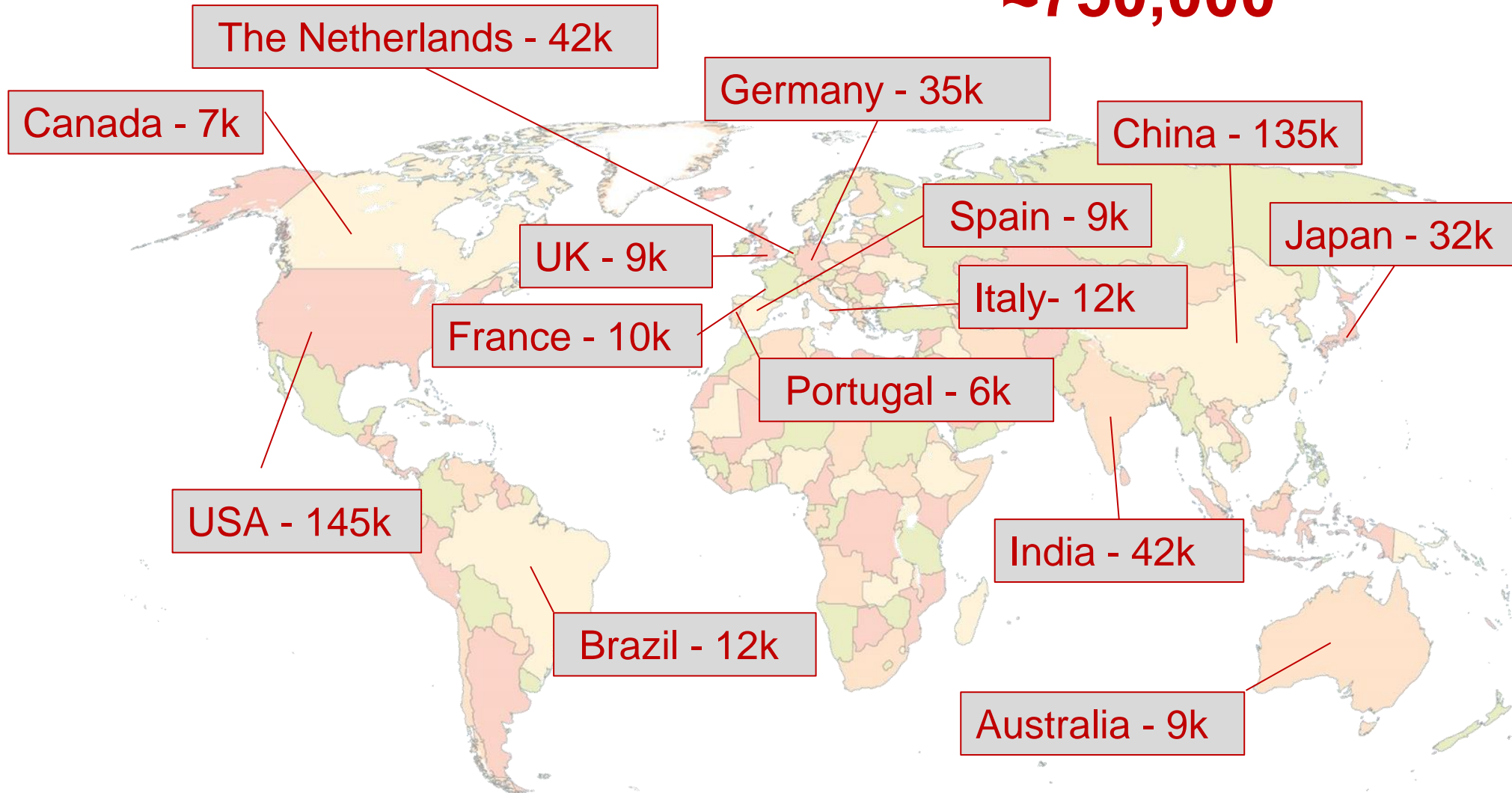




# The OpenFOAM® Computational Library

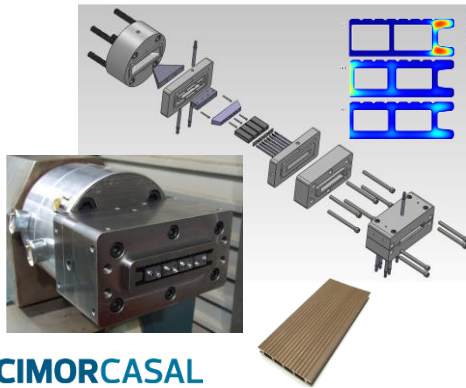
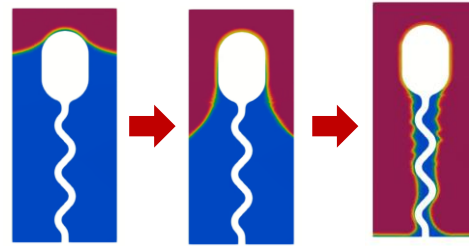
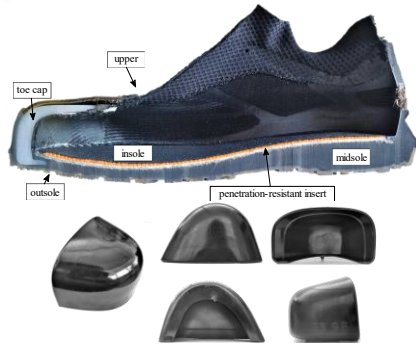
Downloads during 2023 of the three major forks

≈750,000



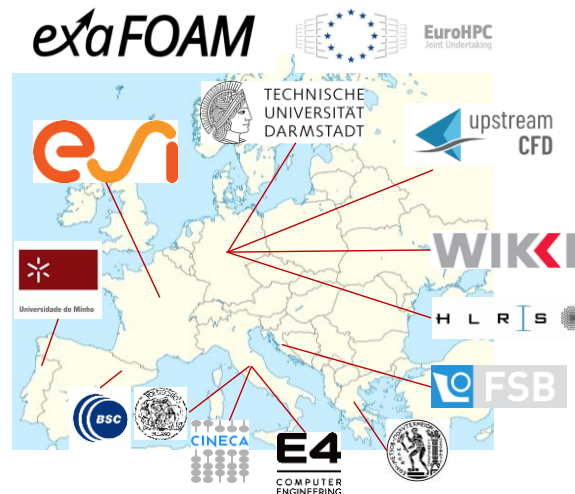
# Introduction

**FAMEST**  
— Footwear Advanced Materials, Experiments  
and Software Technologies —



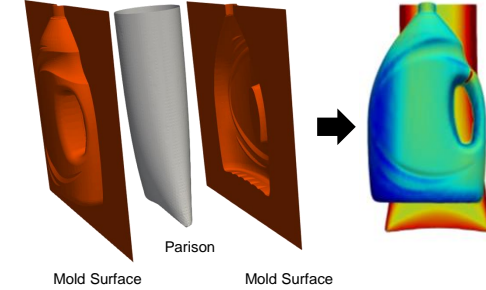
**SOCIMORCASAL**

**exaFOAM**

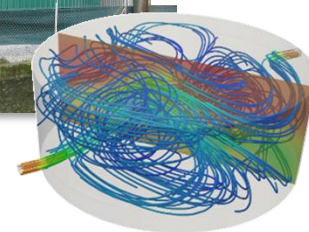


upstream  
CFD

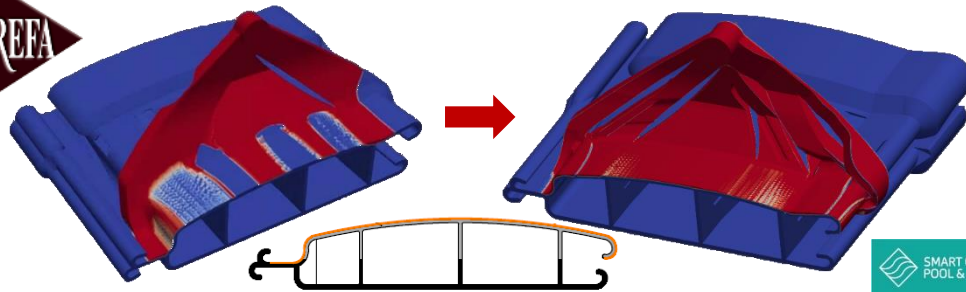
**LOGOPLASTE**



**FORTISSIMO**



**SOPREFA**

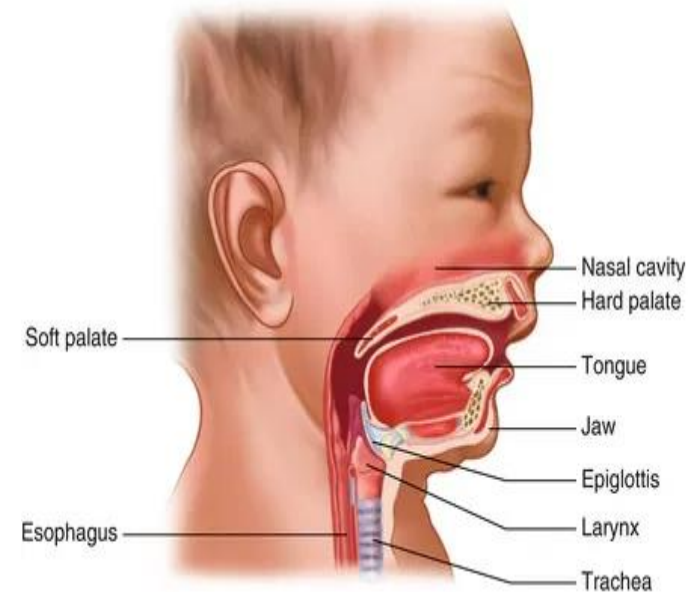




# Introduction

## Pacifiers Assessment

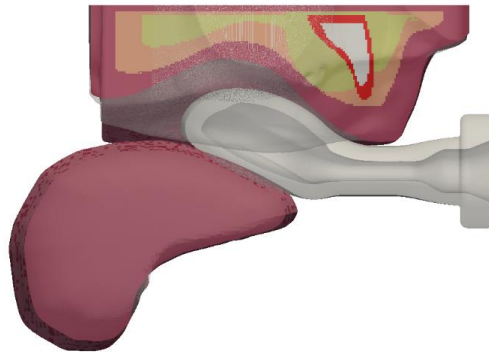
Development of a computational methodology capable of predicting the newborn oral cavity behavior during pacifier sucking



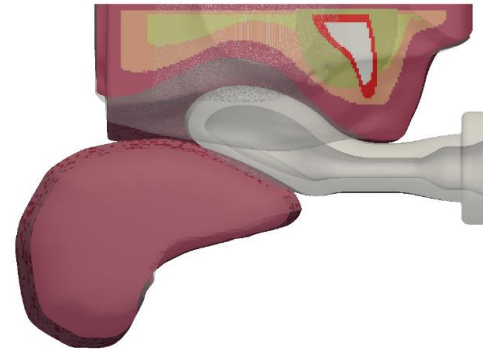
# Introduction

## Pacifiers Assessment

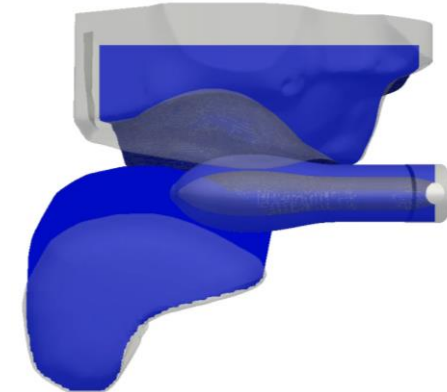
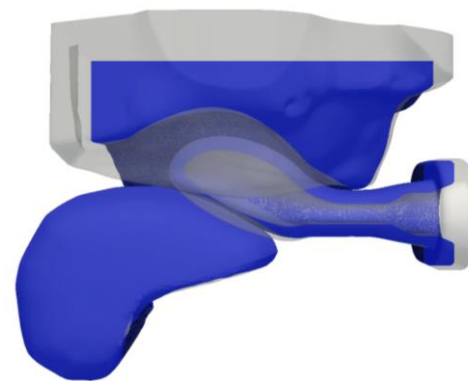
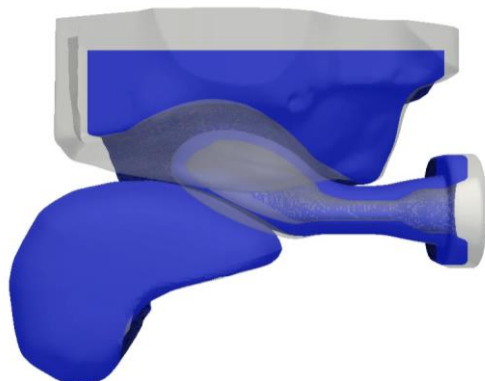
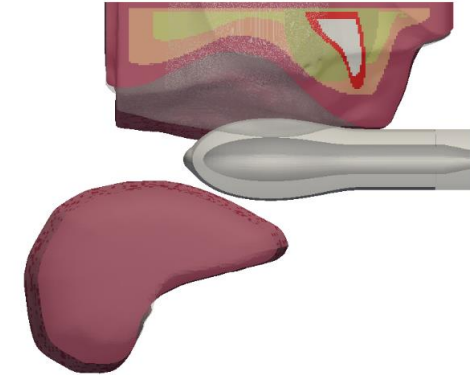
**NUK Genius Pacifier**



**NUK Standard Pacifier**



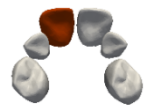
**Conventional Pacifier**



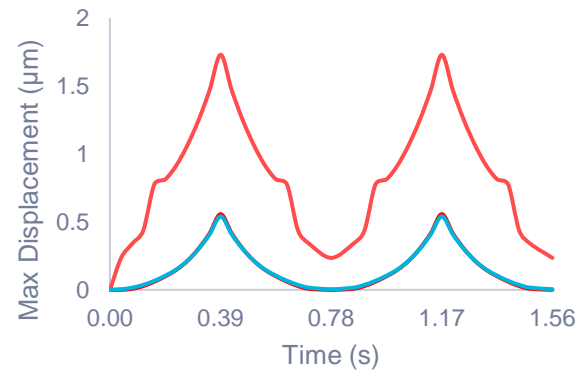


# Introduction

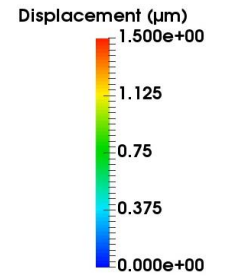
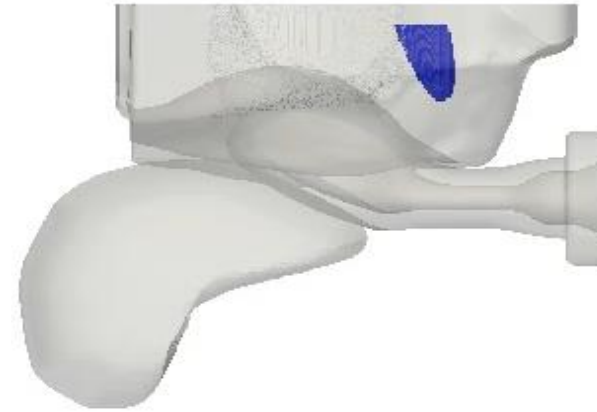
## Pacifiers Assessment



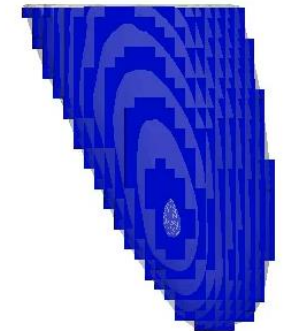
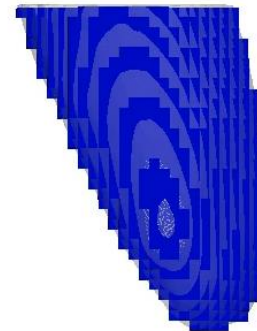
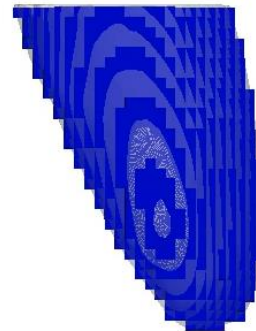
Central Incisor



- NUK Genius Pacifier
- NUK Standard Pacifier
- Conventional Pacifier



Conventional Pacifier



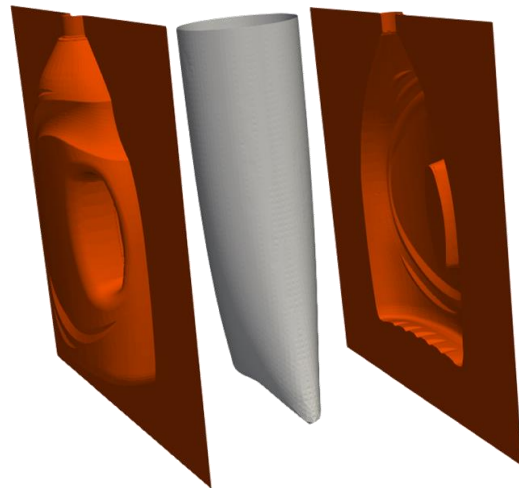
Displacement 1000x





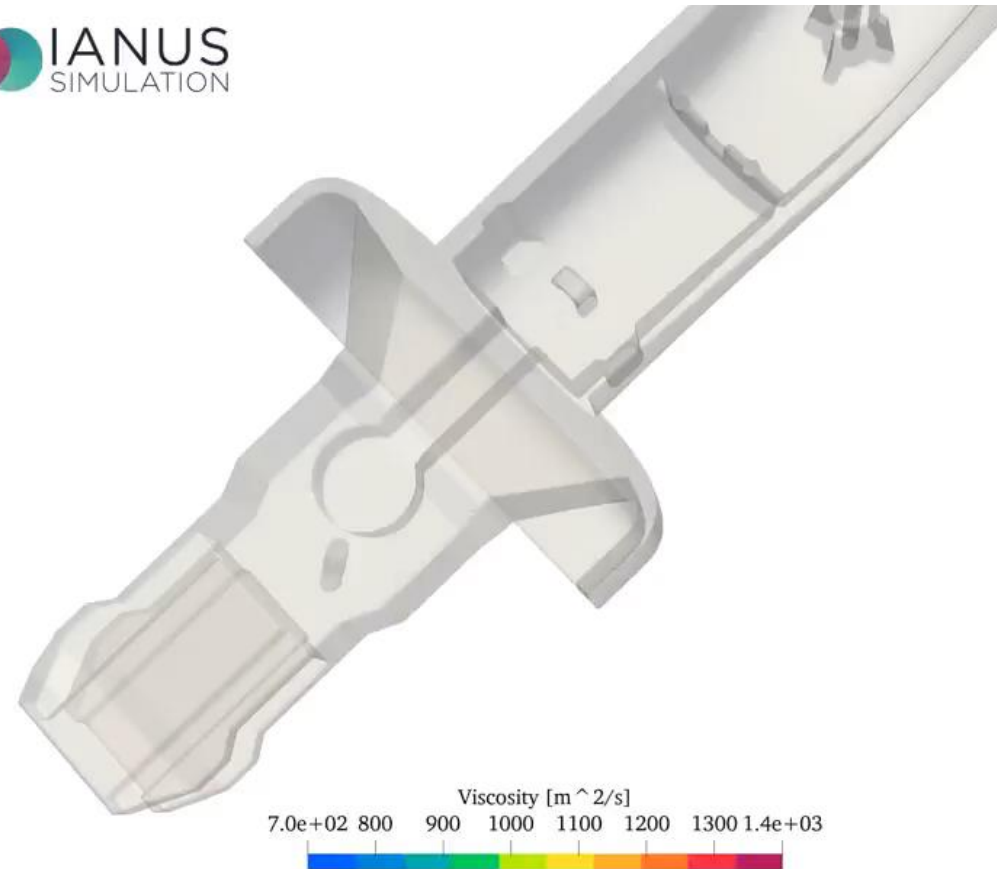
# Introduction

## Extrusion Blow Molding Simulator



# Introduction

## Injection Molding



<https://www.linkedin.com/feed/update/urn:li:activity:7187341849596940288/>



# Computational Modelling

## Notation

- Scalars – plain characters - e.g.  $p, \rho, \eta_p, \lambda$
- Tensors [ ], vectors { } or bold characters – e.g.  $\mathbf{U}, \boldsymbol{\tau}$
- Nabla (  $\nabla$  ) the following differential operator

$$\nabla = \left\{ \begin{array}{c} \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} \\ \frac{\partial}{\partial z} \end{array} \right\}$$



# Computational Modelling

## Governing Equations

- Continuity

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{U}) = 0$$

- Momentum

$$\frac{\partial (\rho \mathbf{U})}{\partial t} + \nabla \cdot (\rho \mathbf{U} \mathbf{U}) + \nabla \cdot (p \mathbf{I}) + \nabla \cdot \boldsymbol{\tau} = 0$$

- Constitutive Model (Oldroyd-B)

$$\boldsymbol{\tau} = \boldsymbol{\tau}_S + \boldsymbol{\tau}_P$$

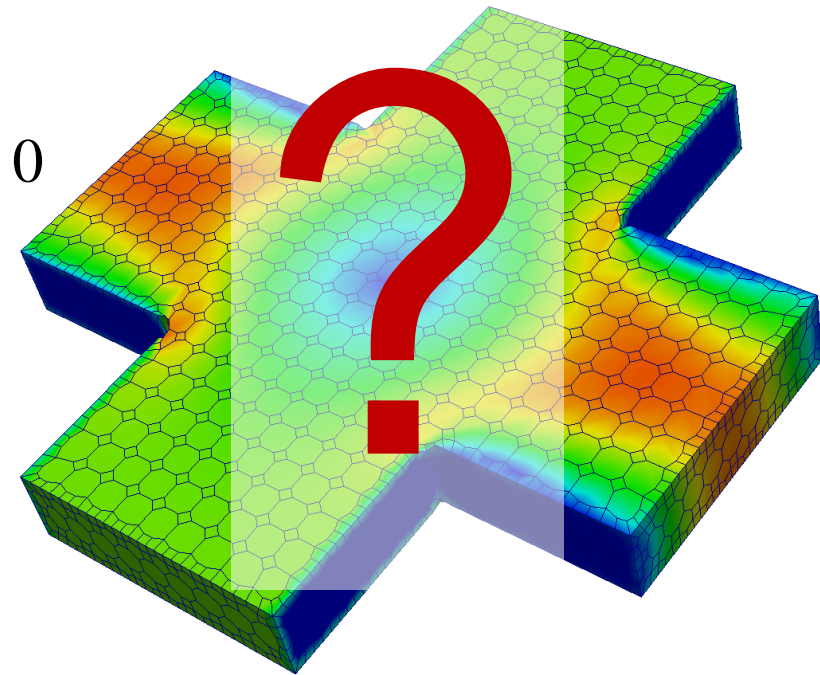
Solvent

$$\boldsymbol{\tau}_S = \eta_S \left( \nabla \mathbf{U} + (\nabla \mathbf{U})^T \right)$$

Polymeric

$$\boldsymbol{\tau}_P + \lambda \left( \frac{\partial \boldsymbol{\tau}_P}{\partial t} + \nabla \cdot (\mathbf{U} \boldsymbol{\tau}_P) \right) = \eta_P \left( \nabla \mathbf{U} + \nabla \mathbf{U}^T \right) + \lambda \left( \boldsymbol{\tau}_P \cdot \nabla \mathbf{U} + \boldsymbol{\tau}_P \cdot (\nabla \mathbf{U})^T \right)$$

Velocity magnitude distribution



# Computational Modelling

- Continuity

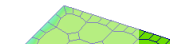
$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{U}) = 0$$

- Momentum

$$\frac{\partial (\rho \phi)}{\partial t} + \nabla \cdot (\rho \mathbf{U} \phi) - \nabla \cdot (\rho \Gamma_{\phi} \nabla \phi) = S_{\phi}(\phi)$$

## Governing Equations

Velocity magnitude distribution

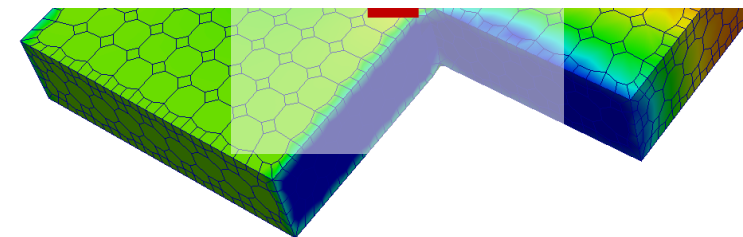


Solvent

$$\boldsymbol{\tau}_S = \eta_S (\nabla \mathbf{U} + (\nabla \mathbf{U})^T)$$

Polymeric

$$\boldsymbol{\tau}_P + \lambda \left( \frac{\partial \boldsymbol{\tau}_P}{\partial t} + \nabla \cdot (\mathbf{U} \boldsymbol{\tau}_P) \right) = \eta_P (\nabla \mathbf{U} + \nabla \mathbf{U}^T) + \lambda \left( \boldsymbol{\tau}_P \cdot \nabla \mathbf{U} + \boldsymbol{\tau}_P \cdot (\nabla \mathbf{U})^T \right)$$





# Computational Modelling

## The Standard Governing Equation

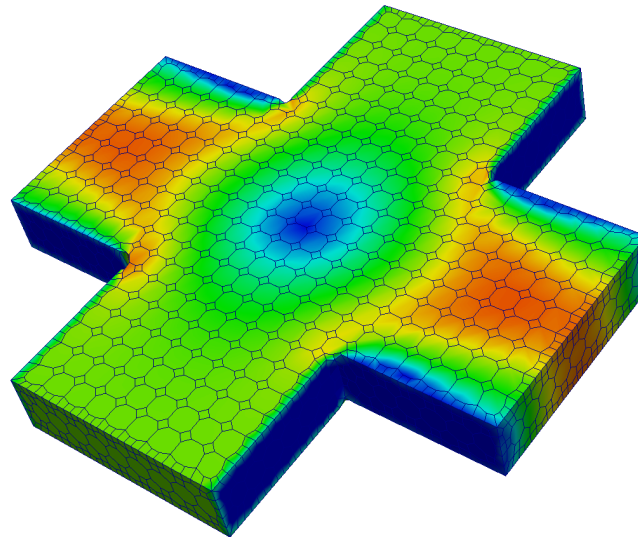
$$\underbrace{\frac{\partial(\rho\phi)}{\partial t}}_{\text{Time evolution}} + \underbrace{\nabla \cdot (\rho \mathbf{U} \phi)}_{\text{Advection}} - \underbrace{\nabla \cdot (\rho \Gamma_{\phi} \nabla \phi)}_{\text{Diffusion}} = \underbrace{S_{\phi}(\phi)}_{\text{Source}}$$



# Computational Modelling

## The Standard Governing Equation

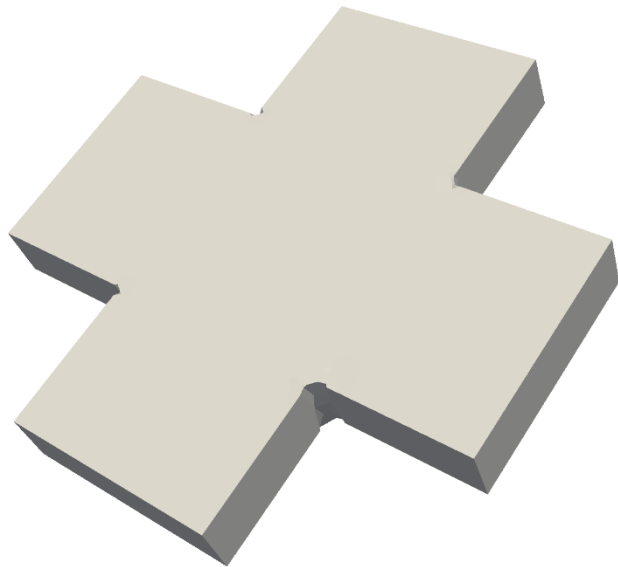
$$\underbrace{\frac{\partial(\rho\phi)}{\partial t}}_{\text{Time evolution}} + \underbrace{\nabla \cdot (\rho \mathbf{U} \phi)}_{\text{Advection}} - \underbrace{\nabla \cdot (\rho \Gamma_{\phi} \nabla \phi)}_{\text{Diffusion}} = \underbrace{S_{\phi}(\phi)}_{\text{Source}}$$



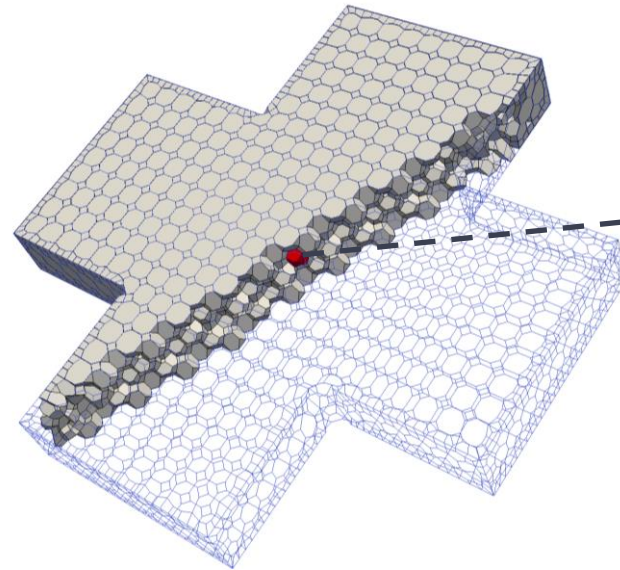
# Computational Modelling

## The Finite Volume Method

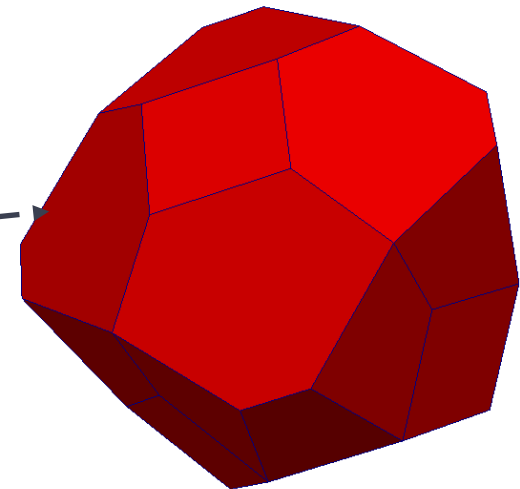
$$\frac{\partial(\rho\phi)}{\partial t} + \nabla \cdot (\rho \mathbf{U} \phi) - \nabla \cdot (\rho \Gamma_{\phi} \nabla \phi) = S_{\phi}(\phi)$$



Geometry



Mesh



Cell

# Computational Modelling

## The Finite Volume Method

$$\frac{\partial(\rho\phi)}{\partial t} + \nabla \cdot (\rho \mathbf{U} \phi) - \nabla \cdot (\rho \Gamma_{\phi} \nabla \phi) = S_{\phi}(\phi)$$



$$\begin{bmatrix} a_{11} & a_{12} & \dots & a_{1N} \\ a_{21} & a_{21} & \dots & a_{21} \\ \dots & \dots & \dots & \dots \\ a_{N1} & a_{N2} & \dots & a_{NN} \end{bmatrix} \begin{Bmatrix} \phi_1 \\ \phi_2 \\ \dots \\ \phi_N \end{Bmatrix} = \begin{Bmatrix} b_1 \\ b_2 \\ \dots \\ b_N \end{Bmatrix}$$

# Computational Modelling

## The Finite Volume Method

$$\frac{\partial(\rho\phi)}{\partial t} + \nabla \cdot (\rho \mathbf{U} \phi) - \nabla \cdot (\rho \Gamma_{\phi} \nabla \phi) = S_{\phi}(\phi)$$



$$\begin{bmatrix} a_{11} & a_{12} & \dots & a_{1N} \\ a_{21} & & & \\ \dots & & & \\ a_{N1} & a_{N2} & \dots & a_{NN} \end{bmatrix} \begin{bmatrix} \phi_1 \\ \phi_2 \\ \vdots \\ \phi_N \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_N \end{bmatrix}$$

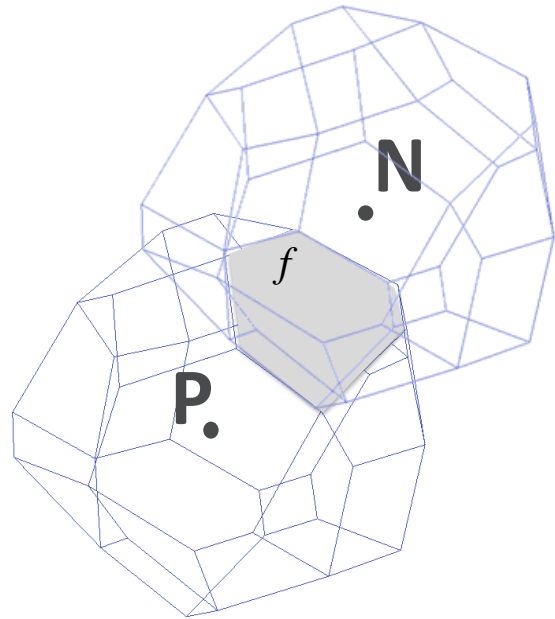
**$[A] \{\phi\} = \{b\}$**



# Computational Modelling

## The Finite Volume Method

$$\frac{\partial(\rho\phi)}{\partial t} + \nabla \cdot (\rho \mathbf{U} \phi) - \nabla \cdot (\rho \Gamma_{\phi} \nabla \phi) = S_{\phi}(\phi)$$



Cell P and N share face  $f$

Discretization (fvSchemes)

$$\frac{\phi_N - \phi_P}{|d|} = \frac{-1}{|d|} \phi_P + \frac{1}{|d|} \phi_N$$

Add to P cell equation

# Computational Modelling

## System of Equations

Solve the system(s) of equations (**fvSolution**)

$$[A]\{\phi\} = \{b\}$$



# Computational Modelling

## System of Equations

Solve the system(s) of equations (**fvSolution**)

$$[A]\{\phi\} = \{b\}$$

- Direct Solvers
- Iterative Solvers



# Computational Modelling

## System of Equations – Iterative Solvers



$$[A]\{\phi\} - \{b\} = \{R\}$$

- $|R|$  quantifies the error of the current solution
  - Relative tolerance
  - Absolute Tolerance



# Linux/OpenFOAM - Memory Aid

## General

- ☐  WSL App ## Open Ubuntu Terminal
- ☐  ParaView App ## open Paraview for post processing
- ☐ right mouse button ## paste clipboard contents in command line
- ☐ tab key ## complete commands
- ☐ Ctrl+R ## Repeat previous Commands
- ☐ Arrow up or down ## Browse by previous commands
- ☐ >> code . ## open VSCode in the current folder
- ☐ >> shopt -s direxand #allow tab in WSL
- ☐ >> explorer . ## Open the current folder in windows explorer

## Linux

- ☐ >> cd <name> ## Change to directory <name>
- ☐ >> cd .. ## Change to previous folder
- ☐ >> cd ## Change to home folder
- ☐ >> pwd ## print current (working) directory
- ☐ >> rm <file> ## remove file named <file>
- ☐ >> rm -rf <folder> ## remove folder named <folder>
- ☐ >> chmod +x <file> ## make the <file> executable
- ☐ >> touch x.foam ## create empty file named x.foam

## OpenFOAM

- ☐ >> openfoam2206 ## load OpenFOAM variables
- ☐ \$FOAM\_TUTORIALS ## Folder for tutorial cases
- ☐ \$FOAM\_RUN ## folder to run cases
- ☐ >> tut ## change to tutorial folder
- ☐ >> run ## change to run folder
- ☐ >> <solverName> ## run solver named <solverName>
- ☐ >> <solverName> > log & ## run solver named <solverName> in background and send output to log file





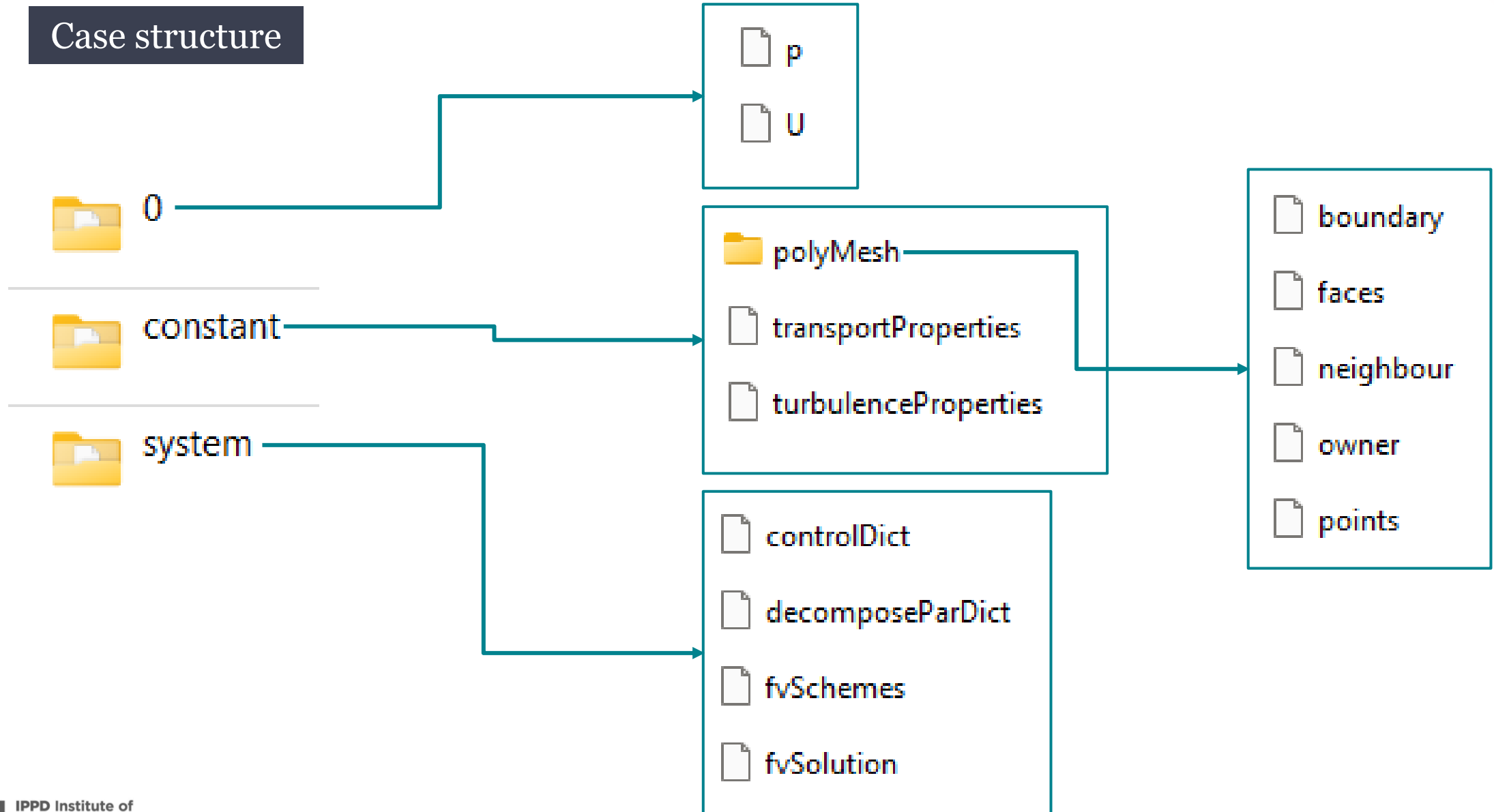
# OpenFOAM 1<sup>st</sup> case study

1. Open WSL
2. Load OpenFOAM variables
3. `>> run`
4. `>> wget https://github.com/Computational-Rheology/PPS39_OFCourse/raw/main/2-CaseFiles/CaseFiles.zip`
5. `>> unzip CaseFiles.zip`
6. `>> cd case 11`
7. `>> code .`
8. Check the case structure with the instructor, in folders *0*, *constant* and *system*



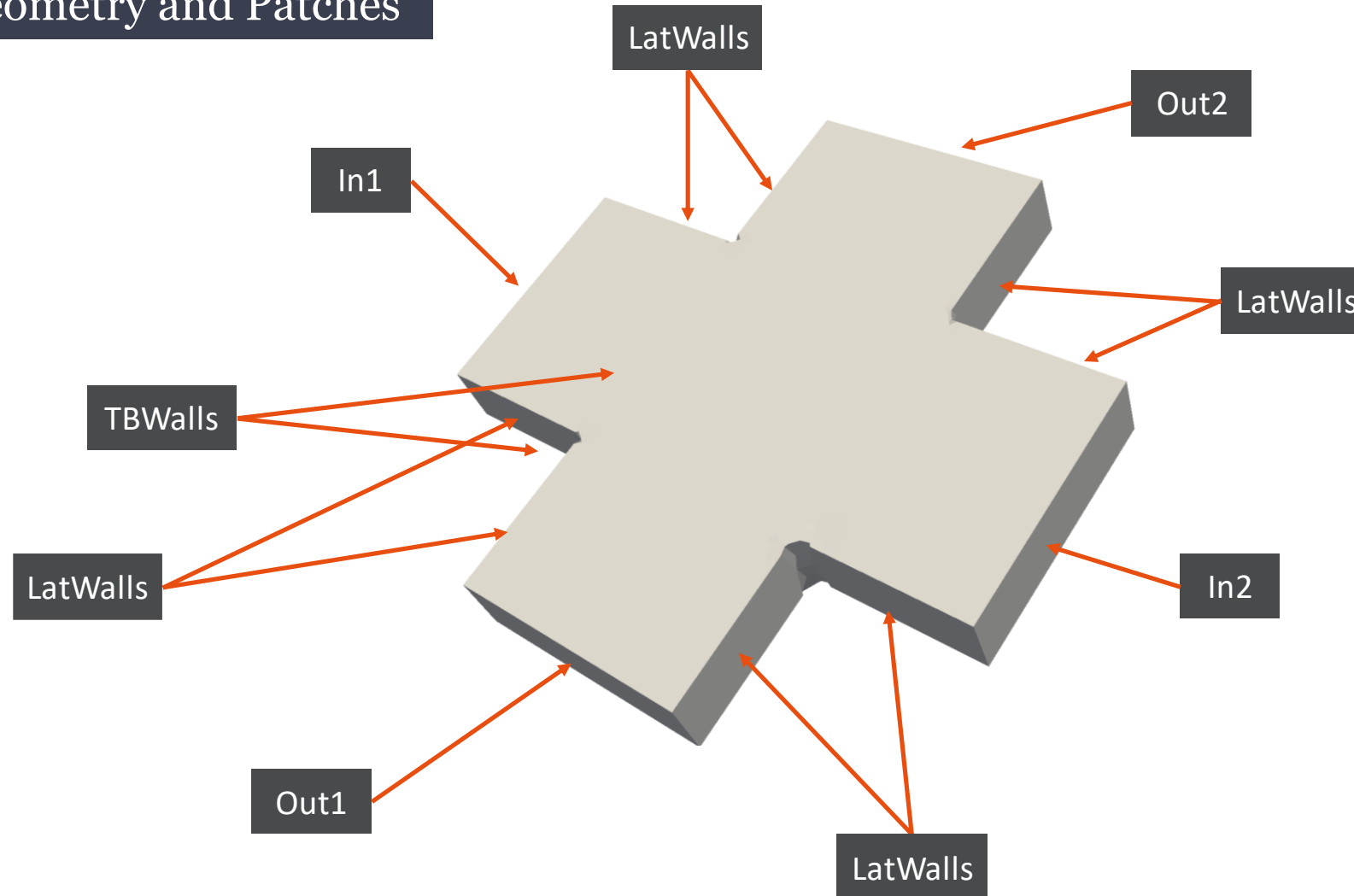
# OpenFOAM 1<sup>st</sup> case study

## Case structure



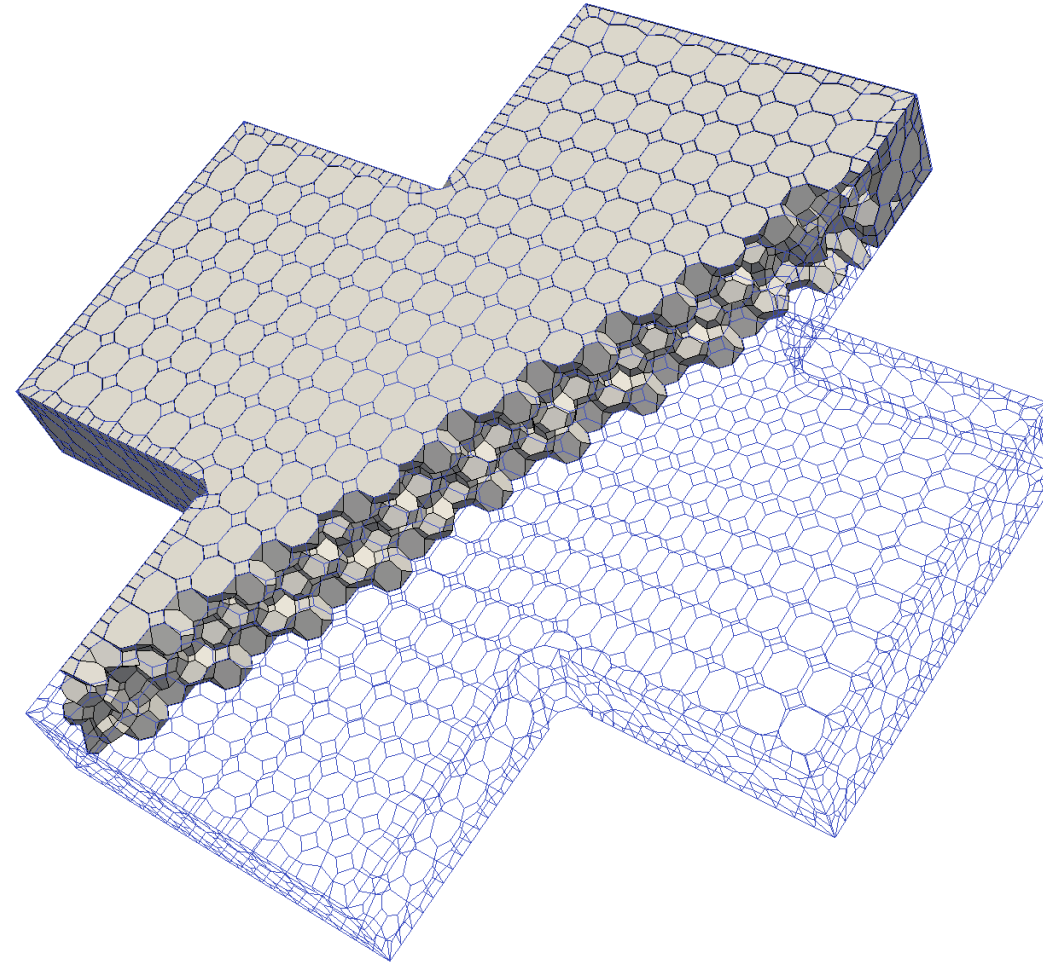
# OpenFOAM 1<sup>st</sup> case study

## Geometry and Patches



# OpenFOAM 1<sup>st</sup> case study

Mesh



## Session P2



# OpenFOAM 1<sup>st</sup> case study

File o/U

.....  
dimensions [0 1 -1 0 0 0 0];

internalField uniform (0 0 0);

boundaryField  
{

In1

{  
  type fixedValue;  
  value uniform (0.01 0 0);  
}

In2

{  
  type fixedValue;  
  value uniform (-0.01 0 0);  
}

Out1

{  
  type zeroGradient;  
}

.....

No.	Property
1	Mass
2	Length
3	Time
4	Temperature
5	Quantity
6	Current
7	Luminous intensity



# OpenFOAM 1<sup>st</sup> case study

File o/p

.....

```
boundaryField
{
```

**In1**

```
{
    type    zeroGradient;
}
```

**In2**

```
{
    type    zeroGradient;
}
```

**Out1**

```
{
    type    fixedValue;
    value    uniform 0;
}
```

.....





## File constant/transportProperties

.....

**transportModel** Newtonian;

**nu** nu [0 2 -1 0 0 0 0] 1e-05;

.....



# OpenFOAM 1<sup>st</sup> case study

## File system/controlDict

```
.....  
application simpleFoam;  
  
startFrom      startTime;  
  
startTime      0;  
  
stopAt         endTime;  
  
endTime        1000;  
  
deltaT         1;  
  
writeControl    timeStep;  
  
writeInterval   500;  
  
purgeWrite      0;  
  
.....
```



# OpenFOAM 1<sup>st</sup> case study

## File system/fvSchemes

### ddtSchemes

```
{  
  default      steadyState;  
}
```

### gradSchemes

```
{  
  default      Gauss linear;  
}
```

### divSchemes

```
{  
  default      none;  
  div(phi,U)    bounded Gauss linearUpwind grad(U);  
  ...  
  div((nuEff*dev2(T(grad(U))))) Gauss linear;  
  div(nonlinearStress) Gauss linear;  
}
```

### laplacianSchemes

```
{  
  default      Gauss linear corrected;  
}  
....
```



# OpenFOAM 1<sup>st</sup> case study

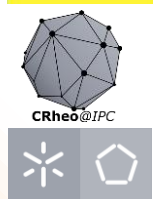
## File system/fvSolution

```
....  
solvers  
{  
  p  
  {  
    solver      GAMG;  
    tolerance    1e-07;  
    relTol      0.1;  
    smoother     GaussSeidel;  
  }  
....  
SIMPLE  
{  
  nNonOrthogonalCorrectors 2;  
  consistent    yes;  
  
  residualControl  
  {  
    p          1e-6;  
    U          1e-6;  
    "(k|epsilon|omega|f|v2)" 1e-6;  
  }  
}  
....
```



# OpenFOAM 1<sup>st</sup> case study

1. `>> simpleFoam #run` the solver and output screen
2. `>> rm -rf 112` #remove the lastTime results folder
3. `>> simpleFoam >log.simpleFoam #run` the solver and output to log.simpleFoam file
4. `>> rm -rf 112` #remove the lastTime results folder
5. `>> simpleFoam >log.simpleFoam & #run` the solver in background and output to log.simpleFoam file
6. Check the contents of the logfile with the instructor



# OpenFOAM 1<sup>st</sup> case study

1. >> touch x.foam
2. >> explorer.exe .
3. Open *x.foam* file with paraview
4. Visualize the results
  - a) Show/hide mesh
  - b) FV results or interpolated
  - c) Rescale color map
  - d) Show/hide interior cells and patches
  - e) Stream tracer (source line and point cloud)
  - f) Contour, clip, slice, threshold and glyph
5. Adapt the velocity boundary conditions
6. Visualize the results in paraview





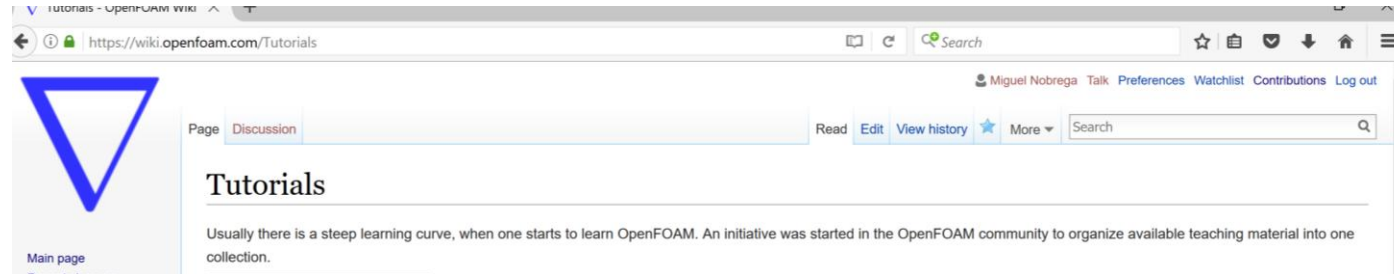
# OpenFOAM 1<sup>st</sup> case study

1. Check the contents of the file system/decomposeParDict with the instructor
2. The banana trick
3. `>> decomposePar`
4. Use the scotch method to decompose
5. `>> mpirun -np 4 simpleFoam -parallel > log.simpleFoamP`
6. `>> simpleFoam > log.simpleFoam`
7. Compare the execution time of both runs



# Where to get more information?

**OpenFOAM Wiki** - [wiki.openfoam.com](https://wiki.openfoam.com)



**OpenFOAM Journal**

[journal.openfoam.com](https://journal.openfoam.com) / [youtube @openfoamjournal6606](https://www.youtube.com/@openfoamjournal6606)

**FOAM@Iberia 2023 – November 2-3, 2023 – Guimarães, Portugal**

<https://2023.foam-iberia.eu/>

**FOAM@Iberia 2024 – October 3-4, 2024 – Ferrol, Spain**

[www.foam-Iberia.eu](http://www.foam-Iberia.eu)

**OpenFOAM Workshop 2024 – Jun 25-28, 2024 – Beijing, China**

[www.openfoamworkshop.org](http://www.openfoamworkshop.org)



IPPD Institute of  
Polymer Processing and  
Digital Transformation