Code_Saturne documentation

*Code_Saturne* version 5.0 tutorial:
Shear Driven Cavity Flow

contact: saturne-support@edf.fr
# TABLE OF CONTENTS

## I  Introduction 3

1  Introduction .............................................. 4  
1.1  Tutorial Components .................................. 4  
1.2  Tutorial Structure .................................. 4  

## II  Setting up 5

1  Setting up .............................................. 6  
1.1  What you will learn .................................. 6  
1.2  Using SALOME _CFD_ together for End-to-End Simulations .......... 6  
1.3  Installing SALOME _CFD_ or Code_Saturne and SALOME separately .......... 6  
1.4  Getting Started with End-to-End SALOME/ Code_Saturne CFD Simulations 7  

## III  Shear Driven Cavity Flow Case with CFDStudy 11

1  General description .................................... 12  
1.1  What You Will Learn .................................. 12  
1.2  Case Description .................................... 12  
1.3  Creating the Code_Saturne case ............................ 13  
2  Creating the Computational Domain within SALOME ................. 13  
2.1  Geometry ............................................. 14  
2.2  Meshing ............................................. 20  
3  Simulating within SALOME ................................ 24  
3.1  Setting up the CFD Simulation ............................. 24  
3.2  Running and Analysing the Simulation ....................... 34  
3.3  Post-processing the Results ............................. 35  
4  References ............................................... 47  
5  Reference Data from [4] .................................. 47
Part I

Introduction
1 Introduction

1.1 Tutorial Components

This tutorial makes use of:

- The SALOME [1] platform for geometry generation, meshing, and post-processing
- Code_Saturne [2], [3] for CFD calculations
- Reference [4] for comparison with published results

To work through this tutorial you will need a computer on which these two software applications are already available or on which you have permission to install them.

1.2 Tutorial Structure

This tutorial is made of two complementary sections:

- Section II describes all the procedures required to get going with setting up CFD simulations using SALOME and Code_Saturne, from code download to case creation.
- Section III illustrates setting up, running, and analysing a CFD simulation entirely with SALOME and Code_Saturne, using the laminar, Shear Driven Cavity as an example. This case is simple but contains enough physics to make it interesting and relevant to practical problems.

If you are already familiar with setting up CFD simulations with Code_Saturne and SALOME, you may skip Section II and go directly to Section III.
Part II

Setting up
1 Setting up

The first part of the tutorial is designed to explain the preliminary steps to setup an end-to-end CFD simulation using Code_Saturne and SALOME and provide an introduction to these software applications.

1.1 What you will learn

In the first part of the tutorial, you will learn:

- How Code_Saturne and SALOME may be combined in SALOME _CFD for end-to-end CFD analyses, from CAD generation to post-processing of CFD results
- How to access and download Code_Saturne and SALOME
- How to install SALOME and Code_Saturne within the CFDStudy module of SALOME
- How to set up a Code_Saturne case with CFDStudy

1.2 Using SALOME _CFD together for End-to-End Simulations

Code_Saturne is a general, 3D CFD solver which can be applied to a large variety of problems involving fluid flow and heat transfer. The Code_Saturne Graphical User Interface (GUI) can be used on its own to define CFD cases, and to run them when a mesh is already available. It has also been integrated as a module called CFDStudy in the SALOME open-source host platform, where it links with other modules to provide a single, user-friendly, graphical environment for end-to-end CFD analyses:

- **GEOM**: CAD module
  - Geometry and computational domain definition
- **SMESH**: Meshing module
  - Surface and volume meshing of the computational domain and defining boundaries
- **CFDStudy**: CFD module
  - Set up and run Code_Saturne CFD calculations
- **Paravis**: Post-processing module
  - Visualisation and analysis with Paraview [5]

The use of the different modules to set-up and run Code_Saturne studies is described throughout the different tutorials, starting with how to initiate a study with CFDStudy, as explained in 1.4.

1.3 Installing SALOME _CFD or Code_Saturne and SALOME separately

The quickest way to get started is to retrieve a SALOME _CFD installer from Code_Saturne website: [www.code-saturne.org](http://www.code-saturne.org). Then install SALOME _CFD by executing the installer in a terminal.

Alternatively, each application, Code_Saturne and SALOME, can be installed separately with an option allowing to integrate Code_Saturne inside Salome. Access to the applications is provided through their respective, dedicated internet sites. Install utilities, installation instructions, environment variables setup for Linux installations, and code documentation including user manuals can also be downloaded from the sites.

To integrate the Code_Saturne and the CFDStudy module in SALOME, SALOME must be installed first, followed by Code_Saturne:
1. **SALOME**

SALOME can be downloaded from the website: [www.salome-platform.org/](http://www.salome-platform.org/)

Complete versions come with the source and binary files and interactive installation procedures.

2. **Code_Saturne** and CFDStudy module

   **Code_Saturne** can be downloaded from the website: [www.code-saturne.org](http://www.code-saturne.org)

   The code is packaged with all its libraries and the installer, which can also be downloaded from the same site, makes the installation process fully automatic. However, external, required packages, such as Python, are not part of the **Code_Saturne** package and must be acquired from their respective publishers and installed separately. As part of the installation process, the **Code_Saturne** installer will check for their availability and issue notifications if they are not found.

   To enable the CFDStudy module, the `--with-salome` flag must be added as a parameter to the `configure` command. The flag must point to the path of your SALOME application.

   For example: `--with-salome=/opt/salome_8.1.0`.

   Additional information can be found in the installation guide of **Code_Saturne** available on the **Code_Saturne** website.

### 1.4 Getting Started with End-to-End SALOME/ Code_Saturne CFD Simulations

The process of creating the required **Code_Saturne** directory and file structures is described first, followed by an example for the Shear Driven Cavity Flow case.

#### File Structure

**Code_Saturne** makes use of a pre-set directory structure to access and save input and output files. Simulations are organised in terms of studies and cases. Conceptually, a study contains a series of cases which all rely on the same geometry. The cases represent different instances of simulations for this common geometry, for example for different operating conditions. The pre-set **Code_Saturne** directory structure can be easily created using the CFDStudy module within SALOME, as described below.

To set up a study and a case, first open the combined interface of **Code_Saturne** and SALOME by typing the command:

```bash
$ code_saturne salome
```

in your terminal. This will start the SALOME platform with the CFDStudy module enabled.

Then, activate CFDStudy, either by selecting it in the module chooser in SALOME’s top tool bar or by clicking on its icon (Figure II.1).

![Figure II.1: Opening CFDStudy module](image)

The module’s activation screen will open as shown in Figure II.2.
In the module’s activation screen, select [New] to create a new study. Then enable CFDSTUDY toolbar, if it does not appear, in [View] Toolbars CFDSTUDY. In the toolbar or in the CFDSTUDY menu, click on Choose an existing CFD study or Create. The study and case creation panel will then pop up (Figure II.3).

In the panel, choose where you want the study to be stored on your computer and enter the names of the study and the first case. Further cases can be added to any study at a later stage. Leave the create MESH directory and create POST directory tick-boxes selected, as this will automatically create the entire directory structure required by Code_Saturne. Press [OK] to create the study and the case. The Code_Saturne directory tree for the study and case will then become visible in the Object Browser, as shown in Figure II.4.
The meaning of the different directories and files is recalled here briefly. For detailed information, please refer to the Code_Saturne user manual [3].

- **DATA directory**
  - **REFERENCE directory**
    - Tabulated thermophysical properties for chemical species and atmospheric properties
    - *cs_user_scripts.py* file
      - Python file in which users can modify Code_Saturne parameters and settings
  - *SaturneGUI* file
    - Shell file pointing to the Code_Saturne GUI executable

- **RESU directory**
  - Initially empty, it will contain the results files and Code_Saturne outputs of each Code_Saturne run, organised in chronological order

- **SCRIPTS directory**
  - Bash file *runcase*, in which the PATH to the Code_Saturne executable and the run command are automatically setup for the case name

- **SRC directory**
  - User coding source files
    - **REFERENCE directory**
      - Templates for all the available user coding files
    - **EXAMPLES directory**
      - Practical examples of implementation of user coding for different purposes

Initially, the **MESH** and **POST** directories are both empty. By default when run from a case directory, Code_Saturne is set up to look for the mesh file in the **MESH** directory. The **POST** directory could contain post-processing routines if required.
Example: Creating the Shear Driven Cavity Flow Study

Follow the steps described in 1.4 to open CFDStudy and create a new study.

Then select the path where the study directory will be created and name the study Driven_Cavity. Tick the box Add new case(s) and enter the case name Reynolds1000 as shown in Figure II.5. The boxes to create the MESH and POST directories have to be ticked.

![Figure II.5: Specifying the Shear Driven Cavity Flow Study and Reynolds1000 Case](image)

The whole directory structure of Code_Saturne is generated and can be visualized in the Object Browser tab (Figure II.6).

![Figure II.6: Shear Driven Cavity Flow Study and Reynolds1000 Case file structure](image)

The case is then ready to be set up, as described in 3.1.
Part III

Shear Driven Cavity Flow Case with CFDStudy
1 General description

In the second part of the tutorial, the preparation, simulation and analysis of the Reynolds1000 case of the DrivenCavity study is described, from the construction of the computational domain and mesh to the preparation, running and post-processing of the CFD simulation. All the following steps in this second part are presented using CFDStudy module. However you could also go through this tutorial using Code_Saturne and SALOME separately.

1.1 What You Will Learn

Through this tutorial, you will learn how to perform an end-to-end CFD simulation using Code_Saturne [2, 3] together with the SALOME [1] platform, from creating the computational domain to analysing the end results and comparing them with available data. Specifically, you will:

- Create a computational domain with SALOME using available shapes and groups
- Create an hexahedral mesh with different mesh refinements in the X, Y and Z directions with SALOME
- Setup a steady-state, viscous, laminar, isothermal, constant properties fluid CFD simulation with non-slip walls, a moving wall, and symmetry planes with the CFDStudy module and using the SALOME mesh
- Control and run the Code_Saturne simulation from the CFDStudy module
- Examine the Code_Saturne output and results files, including data along specified profiles and at monitoring points
- Analyse and visualise the results with SALOME

1.2 Case Description

A viscous fluid is contained in a box, or cavity. All the walls of the cavity are stationary, but for one wall which is sliding in its plane and sets the fluid in motion inside the box through entrainment. This is a well-know academic case for which published results are available [4].

For the purpose of this example, the cavity is rectangular, two-dimensional, and the top wall of the cavity slides from left to right, as illustrated in Figure III.1 below.

Figure III.1: Schematic of the 2D Lid Driven Cavity.
Geometry

The cavity is a square of length $L = 1.0 \text{ m}$.

Fluid Properties

The fluid is given the properties listed in Table III.1 below:

<table>
<thead>
<tr>
<th>Property</th>
<th>Value</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>Density ($\rho$)</td>
<td>1.0</td>
<td>kg/m$^3$</td>
</tr>
<tr>
<td>Viscosity ($\mu$)</td>
<td>$10^{-3}$</td>
<td>Pa.s</td>
</tr>
</tbody>
</table>

Table III.1: Fluid Properties.

Boundary Conditions

The problem is considered to be isothermal and the domain is fully enclosed by impermeable, non-slip walls. This means that, exactly at the surface of the walls, the fluid inside the box attaches to the walls and has exactly the same velocity as the walls. Therefore, the problem is fully defined by specifying the velocity of the walls (Table III.2). The side and bottom walls are stationary. The top wall slides in the positive $X$ direction at a speed of $1.0 \text{ m.s}^{-1}$.

<table>
<thead>
<tr>
<th>Surface</th>
<th>$V_x$ (m/s)</th>
<th>$V_y$ (m/s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$W_l$</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>$W_b$</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>$W_r$</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>$W_t$</td>
<td>1.0</td>
<td>0.0</td>
</tr>
</tbody>
</table>

Table III.2: Wall velocities.

Flow Regime

The flow Reynolds number is evaluated to determine whether the flow is in the laminar or turbulent regime. The Reynolds number is calculated based on the lid velocity, the box dimensions, and the fluid properties.

For this case, the fluid properties and lid velocity have been chosen so that $Re = \frac{\rho U L}{\mu} = 1000$, indicating that the problem is in the laminar flow regime.

1.3 Creating the Code_Saturne case

The DrivenCavity study and Reynolds1000 case are created by following the instructions in section II of this tutorial.

The next paragraphs describe how to setup and run the lid driven cavity simulations for a flow Reynolds number of 1000.

2 Creating the Computational Domain within SALOMÉ

First you need to create the geometry of the cavity which defines the bounds of the computational domain and then mesh this domain. Installation of this tool is described in section II.
Code_Saturne is a 3D code and, therefore, the computational domain will be created as a 3D box, aligned with the X, Y, and Z directions. However, as we are only interested in 2D modelling and the flow motion in the X−Y plane, the box will be designed to be thin and contain only one computational cell in the Z direction.

2.1 Geometry

Once the CFD case structure is created from the CFDSStudy module, the current SALOME file is saved by accessing File > Save. For this tutorial, we choose the file name Cavity (Figure III.2). The file Cavity.hdf will be created in directory Driven_Cavity.

![Figure III.2: Saving the SALOME file](image)

It is advisable to save the file at regular intervals when preparing a case, preferably, before and following all significant modifications. That way your latest work is preserved and, should mistakes happen, it is always possible to exit SALOME and restart it from the latest saved file.

As SALOME is an integrated interface to different capabilities: CAD, meshing, visualisation, the next step is to access the Geometry module of the program in order to start building the shape of the computational domain.

In the main menu bar, either click on the Geometry icon or choose Geometry from the drop-down module menu as shown in Figure III.3 below.
The **Geometry** menus then become available in the top bar and the **OCC scene viewer** opens in the right hand side, where the geometry objects can be visualised and manipulated. The scene viewer comes with its own set of 3D-viewing menus to control the display and perform operations such as panning in and out, and rotating and moving the object in the scene.

The **Geometry** module of SALOME makes it possible to build bespoke objects from scratch from constituting elements such as vertices, edges and faces, or from pre-set generic shapes.

In this instance, we use the latter, more direct method and create the cubic cavity shape by selecting: `New Entity ➔ Primitives ➔ Box`. Alternatively, use the icon short-cut on the menu bar.

A **Box Construction** pop-up menu is then activated, as shown in Figure III.4 below.

In this menu, choose the second method of Box construction, which makes it possible to build a prismatic box attached to the origin by defining its three dimensions. The default name is changed
to Cavity and the dimensions in the $X$, $Y$, and $Z$ directions are specified according to the problem definition in paragraph 1.2.

The resulting Cavity object is shown in Figure III.5 below, as wireframe (top) and after applying shading (bottom).

![Figure III.5: Wireframe (left) and Shading (right) display modes.](image)

As shown in Figure III.5 (bottom), to change the Display Mode to Shading, right click in the scene viewer to open the drop down menu, then select Display Mode and Shading. Likewise, Change background ... may also be accessed from the same drop-down menu to change the background colour of the OCC viewer to light blue as in the figures.

At this point, the cavity object is complete and could be meshed straight away. However, since the box has been created as a whole entity it would then neither be possible in the mesher to specify different meshing characteristics for the $X$, $Y$ and $Z$ directions nor to identify the 6 external surfaces
as boundaries for the Code_Saturne calculations. Before meshing, it is then necessary to create and identify these elements. In the next step, you will automatically create Edges Groups using the Propagate utility. Faces Groups will then be created by manually selecting them.

With the Cavity object selected in the Object Browser in the left-hand column, in the top menu bar, click on Operations Blocks Propagate as shown in Figure III.6 below. By using the Apply and Close action at the bottom of the menu, you will be able to automatically create three Edges Groups, each of them containing every edges parallel to one direction.

Figure III.6: Using the Propagate utility to automatically create Edges Groups.

One can now rename them as X.Edges, Y.Edges and Z.Edges by right-clicking on each group and then select Rename. The Cavity object in the object browser should now contain three Edges Groups as shown in Figure III.7 below.
The next step is to group faces according to the boundary types listed in Table III.2 above. Right-click on Cavity in the object browser and select Create Group to activate the pop-up menu, as shown in Figure III.8 below.

Alternatively, access the group creation via New Entity Group Create Group. Select the icon as Shape type, name the group, and ensure that the Main Object is Cavity. To be easily identifiable, the groups are named after their boundary types. The first group is created for the external, stationary walls and named Stationary walls. With the default No restriction choice already active in the Main Shape Selection restriction menu, start adding the required faces by...
left-clicking on them in the OCC viewer and then pressing [Add]. One can zoom with the mouse wheel as well as rotate the Cavity object by holding [Ctrl] + Right Click to select the faces more easily. The faces may be added one by one, or together by multiple-selection through left-clicking and holding the [Ctrl] key. Once the three faces are selected, one can check them by pressing the Show only selected button. The Create Group menu should look as shown in Figure III.9.

![Figure III.9: Selected faces of the Stationary_walls face group.](image)

Press [Apply] and repeat these steps for the next two groups: Symmetry_plane and Sliding_wall, pressing [Apply] each time. Conclude the creation of face groups by pressing [Apply and Close]. The face and edges groups are now listed under the Cavity object in the Object Browser (Figure III.10).

![Figure III.10: Cavity with the edges, faces, and groups.](image)
Cavity is now ready to be meshed. Save the file and proceed to Meshing.

## 2.2 Meshing

Switch to the meshing module, which opens the VTK scene viewer as shown in Figure III.11 below. In this view, the background colour has been changed to white similarly to the OCC scene viewer for the Geometry, by right clicking in the viewer and accessing Change background . . .

![Figure III.11: Switching to the Mesh module.](image)

Similarly to the earlier creation of the geometry object Cavity in the Geometry module, a Cavity mesh mesh object is created first by selecting Mesh Create mesh in the main menu bar.

As shown in Figure III.12 below, in the Create mesh pop-up menu, type Cavity_mesh for the Name and select Cavity as the Geometry object by left clicking on it in the Object Browser.

![Figure III.12: Creating the main mesh object.](image)
The main mesh object holds the global characteristics of the Cavity mesh.

For the volume, in the 3D tab choose Hexahedron \((i,j,k)\). All the (external) surfaces of the Cavity will be meshed similarly, therefore a global setup is applied for the entire mesh via this pop-up menu by clicking on the 2D tab and selecting Quadrangle: Mapping. These two selections will result in a block-structured mesh, with an \((i, j, k)\) matrix structure. This requires identical discretisations along all the edges aligned along the same directions. If we wanted the discretisation to be the same along all the directions, for example 15 sub-divisions along the \(X\), \(Y\), and \(Z\) edges, the 1D tab could be selected and this unique discretisation specified. However, here we need to be able, at least, to specify the \(Z\) direction independently of the two others to create a 3D domain which is only one cell deep. Therefore, the edge discretisations will be specified separately using sub-meshes.

The sub-mesh defining the meshing characteristics in the \(X\) direction is specified first.

Right-click on Cavity mesh in the Object Selector and select Create sub-mesh. Alternatively, with Cavity mesh selected (highlighted) in the Object Browser, the sub-meshes can be created from Mesh or the short-cut icon in the main menu bar. This activates the Create sub-mesh pop-up menu in the VTK scene viewer as shown in Figure III.13 below.

![Create sub-mesh](image)

**Figure III.13:** Creating the \(X\) mesh sub-mesh.

In the name field, enter \(X\) mesh. The Cavity mesh should already be selected in the Mesh field. If not, select it by clicking on Cavity mesh in the Object Browser. Next, pick X_edges for the Geometry field which stores the Geometry Objects the sub-mesh will apply to. Selecting an edge group for the Geometry automatically disables the 3D and 2D tabs in the Algorithm menu. Select Wire discretisation in the drop-down Algorithm menu and press on the gear button next to Hypothesis to specify the discretisation in its own pop-up menu. The Number of Segment object is renamed \(X\) discretisation for future reference. Choose 50 segments with an Equidistant distribution and press OK. Returning to the Create sub-mesh menu, press Apply to instantiate the \(X\) mesh sub-mesh. The sub-mesh objects for the other two directions are defined similarly.

Starting with the \(Y\) direction, the sub-mesh is renamed \(Y\) mesh. The same number of segments is applied in the \(Y\) and \(X\) directions. Therefore, for this sub-mesh the previously created \(X\) discretisation Number of Segment Object is reused in the Hypothesis. Complete the specification of \(Y\) mesh and press Apply.

Lastly, the sub-mesh for the \(Z\) direction is defined. Named \(Z\) mesh, it is similar to the \(X\) mesh and \(Y\) mesh, but defines the discretisation as 1 segment only. For \(Z\) mesh, define a new Number of Segment object named \(Z\) discretisation and specifying 1 segment. Complete the specification of \(Z\) mesh and press Apply and Close to instantiate \(Z\) mesh and leave the Create sub-mesh menu.
The mesh characteristics of the Cavity are now fully specified, as shown in Figure III.14 below where all the characteristics of Cavity_mesh and its sub-meshes have been listed.

Figure III.14: Complete specification of Cavity_mesh and sub-meshes in the three directions.

To mesh the computational domain, right-click on Cavity_mesh in the Object Browser and select Compute, as shown in Figure III.15 (left) below. Again, with Cavity_mesh selected, Compute can also be accessed via the main menu bar Mesh | Compute or via the short-cut icon . The mesh is created and a pop-up window appears automatically summarising the mesh characteristics (Figure III.15 (right)).

Figure III.15: Computing the mesh (left) and resulting mesh (right).

Before the mesh can be used in Code_Saturne, the boundary surfaces must be identified so that it will be possible to impose boundary conditions in Code_Saturne. To designate these surfaces, we now can call on the groups of faces which were created previously in the Geometry.
Right-click on Cavity_mesh in the Object Selector and select Create Group from Geometry to create containers of mesh elements. The Create Group from Geometry pop-menu is activated. Left click on Stationary_walls, Sliding_wall and Symmetry_planes in the Geometry part of the Object Chooser while holding $\uparrow$. The three groups will be automatically added in the Elements box as shown in Figure III.16 below. Ensure that Cavity_mesh has been selected for the mesh and press [Apply and Close].

Figure III.16: Creating the mesh face groups.

To help visualisation, the colour of these groups of faces can be switched to different colours for each group. Right click on the name of the group in Groups of Faces, select Edit group and choose the colour.

The groups containing the boundary faces are now defined and may be visualised in the VTK scene viewer, together or individually, by selecting the groups of mesh faces in the Object Browser. For example, the Stationary_walls group is shown in Figure III.17 (left), and all the groups are visualised in Figure III.17 (right). Note that the names of the groups which have been selected appear in the top right corner of the viewer, in the colour scheme chosen for the group.

Figure III.17: Visualing the Stationary_walls group on its own (left) and all the groups (right).

The mesh is now ready to be exported in a separate file which can be read in by Code_Saturne.
Before exporting the file, it is important to highlight an important naming convention, as the names of the mesh and its elements will be used as their identifiers in Code_Saturne.

Save the SALOME file and export the mesh file in .med format by selecting from the main menu: File > Export > MED file, as shown in Figure III.18 below. The file should be placed in the MESH directory of the DrivenCavity study, where Code_Saturne will expect the file to be situated by default.

For the file name, choose Cavity_mesh; the .med extension is automatically added. You are now ready to set up the CFD simulation with the CFDStudy module.

3 Simulating within SALOME

3.1 Setting up the CFD Simulation

To set-up the CFD Simulation, select the CFDStudy module by clicking on the CFDStudy button or by selecting CFDStudy in the modules drop-down menu as shown in Figure III.19.

Then, right click on Reynolds1000 in the Object Browser and launch the Code_Saturne GUI inside
the CFDStudy modules by clicking on Launch GUI as shown in Figure III.20 below.

![Figure III.20](image)

Figure III.20: Launching the *Code_Saturne* GUI inside the CFDStudy module.

Once the *Code_Saturne* GUI is open, it is advisable to start by saving your CFD set-up file. In the CFDStudy tool bar, click on the Save CFD Data File button (Figure III.21).

![Figure III.21](image)

Figure III.21: Button bar of CFDStudy module.

By default, the file will be saved in the `DATA` directory. Name the file `Reynolds1000.xml`.

![Figure III.22](image)

Figure III.22: Saving the *Code_Saturne* case.

You can now proceed with setting up the case, in the top down order of the folders in the left hand column of the CFDStudy module.

Press on Calculation Environment to open the folder and select Meshes Selection in order to specify the mesh file which will be used for these calculations. In the Meshes tab, for Mesh import, Import meshes is already selected by default. The Local mesh directory should already be pointing to the `MESH` directory for the study, where the mesh file had been exported in paragraph 2.

Initially, the List of meshes table is empty. Click on the + sign button below the table and a popup menu will show the content of the local mesh directory (here `./MESH`) from which the mesh file can be selected. Select the Cavity mesh.med file previously exported from SALOME at the end of the meshing process, and press Open. The file name will then be added to the list of meshes (Figure III.23).
Figure III.23: Selecting the mesh file for the calculations.

No further input is necessary as the faces do not require reorientation, joining, or sub-dividing, and there are no periodic boundaries to be concerned with.

Next, move to Thermophysical Models to specify the flow physics for the calculations.

In Calculation features, change unsteady flow to steady flow in the choice of algorithms at the top. Leave all the other default values unchanged: single-phase flow is active and the combustion and electrical models are all inactive (Figure III.24).

Figure III.24: Selecting the flow physics.

Continue setting up the Thermophysical Models.

By default Mobile mesh (ALE method) is not selected. As the mesh in our case is stationary, Mobile mesh does not need visiting and changing. However, Turbulence models needs to be changed to No model (i.e. laminar flow), which is the correct flow regime for the flow Reynolds number that we have chosen in paragraph 1.2 (Figure III.25).

Figure III.25: Selecting the laminar model.

Thermal model is inactive by default. As our case is isothermal, it does not need to be visited and changed.

Save the file and, as there are no additional scalars, skip the Species transport category and move to the Physical Properties folder where the thermal and physical characteristics of the fluid of interest for the calculations will be specified.
In Reference values, the pressure reference value of 101325.0 \( \text{Pa} \) is set by default and does not need to be changed. The other default values for velocity and length can also be left unchanged. However, the Fluid Properties must be modified for our fluid. Click on Fluid Properties in the left hand panel and, as per Figure III.26 below, alter the values of Density and Viscosity to the chosen values which are listed in Table III.1 above.

![Figure III.26: Specifying the fluid properties.](image)

The default value for Specific heat does not need to be altered as it is not a parameter in these isothermal calculations. Likewise, gravity is not a parameter of this problem (constant density), therefore the Gravity category does not need to be visited and changed.

Click on the Volume Conditions folder. Sub-volumes of the computational domain (Volume regions definition) have not been defined as separate regions for initialisation and Coriolis source terms are not taken into account in this tutorial. Therefore, the Volume Conditions category must be changed only for the Initialization.

In this sub-folder, press on the icon next to Velocity to bring up the Mathematical Expression Editor and change the initial first component of the velocity to 0.1 \( \text{m/s} \) as shown in Figure III.27 below.

![Figure III.27: Specifying the initial values.](image)

To define both the boundaries and the boundary conditions, under the Boundary Conditions folder, select Definition of Boundary Regions.

On the left side of the screen move from Driven_Cavity.Reynolds1000.Reynolds1000.xml Browser
to Object Browser. Then in the Mesh part click on the group of faces desired and on the right side of your screen press on [Add from Salome]. The group added by default is set with the boundary condition Wall as shown in Figure III.28.

![Figure III.28: Importing groups of faces from SALOME.](image)

In case there are a lot of groups, it is quicker to do a Mesh quality criteria calculation. Select the Preprocessing Mode as shown in Figure III.29 and go to Calculation management, then select Mesh quality criteria in the Run Type category.

![Figure III.29: Preprocessing and Calculation modes.](image)

This can be used to analyse the mesh. For visualisation of the mesh quality criteria, export the files in the specified output format. To do so, go to Calculation control and select Output control. Then, in the Writer tab, select the desired output format for the mesh quality criteria. As shown in Figure III.30 below, the EnSight format is preselected by default. Doing a Mesh quality criteria also enables Code_Saturne to detect the different groups of faces in the mesh.
Go back to Calculation Mode and to the Boundary Conditions tab, select Definition of Boundary Regions and then press Import groups and references from Preprocessor listing. The Select a Code_Saturne Preprocessor listing pop-up frame will then appear, which allows you to select the preprocessor.log file to use to load the boundary surfaces of the computational domain. You will then be able to apply specific boundary types to each of the surfaces or groups of surfaces. This file is in a directory called preprocess20161020-0817 for instance. In the pop-up frame, select the file preprocessor.log as shown in Figure III.31.

All the boundary faces defined in the mesh are now loaded and are set by default with the Wall boundary condition.

The first boundary group in our problem contains the side walls, which are considered stationary and non-slip. In the first row of the list of boundary regions, double click the cell in the Selection area column in order to identify the boundary region in the mesh file. With the cursor in the cell, type in the exact name of the boundary in the mesh file, i.e., Stationary_walls. If the preprocessor.log
The file has been used, the Selection area is already completed.

The second set of boundaries are the symmetry planes which are used to enforce the 2D nature of the problem. In the second row, double click on Wall in the Nature column to activate the drop down menu of boundary types. Choose Symmetry and release the menu. In the Selection criteria cell, input the name of the boundary in the mesh file, i.e., Symmetry planes, if no preprocessor.log file has been used.

In the third row of the table, as for the two other boundary conditions, input the name of the sliding wall boundary in the mesh file, i.e., Sliding wall if no preprocessor.log file has been used.

Note that it is better to press enter after typing in the name of the region to ensure that it is properly recorded by the GUI.

The boundaries and their global types are now defined, as shown in Figure III.32 below.

![Figure III.32: Boundary definition.](image)

Having defined their type, the exact characteristics of the boundaries must be further specified. Click on the Boundary conditions sub-folder which presents the list of boundaries which can be further specified (Figure III.33).

![Figure III.33: Specifying the sliding wall.](image)
The symmetry boundaries are fully defined and do not need further specification. Therefore, they do not appear in the list. However, Wall boundaries can each be further defined as smooth or rough, and as Sliding wall. By default, the wall surface is smooth and this parameter does not need to be changed. The Stationary_walls boundary is fixed and fully specified. However, click on the Sliding_wall boundary and activate the Sliding wall selection. Fields then appear for the U, V, and W velocity components of the wall. By default, these velocities are null. Click in the U field and enter 1.0, as shown in Figure III.33.

The mesh and physics of the problem have now been set up. Now, parameters related to the calculation can be specified.

In Numerical Parameters, leave the settings unchanged in the Global parameters sub-folder and move to the Equation parameters. In the Scheme tab, apply the Centered scheme and disable the Slope test for the velocity (Figure III.34). The former will provide second-order accuracy in space and the latter is unnecessary for a smooth flow.

![Figure III.34: Specifying the flux schemes.](image)

To set the duration of the run and the time step, move down the selection tree to the Pseudo-Time step sub-folder. Change the Number of iterations to 400 and increase the Maximal CFL number to 8.0, leaving the other parameters unchanged (Figure III.35).

![Figure III.35: Selecting the number of iteration and CFL number.](image)

Move to the Calculation control folder. Time averages are not of interest for these simulations and the sub-folder does not need to be visited. However, we want to keep track of the solution at different monitoring points to see how it evolves during the calculations. Aside from the solution residuals and the minimum and maximum values of the flow variables which Code_Saturne outputs during the calculations, tracking the solution at significant monitoring points is a very important mean of gaining
confidence in the convergence of the calculations and judging whether calculations have been run for a sufficiently large number of iterations. This is explained further in paragraph 3.2 below.

Click on the **Output control** sub-folder panel. The first three tabs, **Output control**, **Writer**, and **Mesh** are set by default to the correct values for this case and do not need to be changed. The **Log frequency** in the **Output control** tab is set to print the calculations diagnostics such as the residuals at each time step. In the **Writer** tab, the format of the results file which will be used for post-processing is already set to EnSight, which is compatible with the SALOME module ParaVis which will be used to post-process the results after the run. The file will be located in the postprocessing sub-directory of Reynolds1000/RESU/runDateAndTime, where runDateAndTime corresponds to the time at which the run was started. Clicking on the results row brings up the additional information about the **Frequency**, **Time-dependency** and **Options**. As already set by default, the results file will only be written at the end of the run and on the **Fixed mesh** used for this case. The **Options** relate to the specific details of the file format. The **Mesh** tab is already set to output the calculations data in all the fluids cells and at all boundary faces to the results file.

In the **Monitoring Points** tab, change the **Monitoring points output** file format to .csv so that the files may be read in ParaView. Using the [+ ] button, add four monitoring points located at (0.25, 0.5, 0.0), (0.5, 0.25, 0.0), (0.5, 0.75, 0.0) and (0.75, 0.5, 0.0) (Figure III.36).

![Figure III.36: Defining the monitoring points.](image)

A visualization of the monitoring points in the VTK Viewer in the middle of the screen, as shown in Figure III.36 above, is available if **Display monitoring points on SALOME VTK viewer** box is ticked.

![Figure III.37: Selection of output variables on the surfaces of the domain.](image)
Click on the **Surface solution control** sub-folder and disable **Post-processing** for *Yplus* and *Stress* as they are not relevant to these simulations (Figure III.37).

Lastly, we want to output one-dimensional profiles of variables along straight lines at the end of the calculations. Click on **Profiles** and add two profiles which go through the centre of the Cavity by clicking on the [Add] button. The first one for the *X* velocity along the *Y* axis and the second one for the *Y* velocity along the *X* axis. In turn, specify all the fields listed below the table of profiles, starting with **Filename** and finishing with the variables which are to be stored on output.

For the *X* velocity profile, choose **XVel_YaxisCenterLine** for **Filename**, and **.csv** for **Format** so that the profiles may be read in the ParaVis post-processing module of SALOME. The **Output frequency** must be set to **at the end of the calculation**. To define the line, press on the **Mathematical expression editor** button adjacent to **Line Definition**. The line is defined by the following equation:

\[
\begin{align*}
  x &= 0.5 \\
  y &= s \\
  z &= 0.0
\end{align*}
\]

By definition, *s* varies between 0 and 1. For the **Number of points** enter 50 to account for the 50 cells across the domain’s height. Finally, click on **Velocity[0]** and use the [[ ] button to add it in the list of variables, and press [Add] to store the profile in the list (Figure III.38).

![Figure III.38: Specifying the 1D output profiles.](image)

Repeat the procedure for the *Y* velocity profiles, this time entering **YVel_XaxisCenterLine** for **Filename**, and the following equation for the line definition:

\[
\begin{align*}
  x &= s \\
  y &= 0.5 \\
  z &= 0.0
\end{align*}
\]

Select the **Velocity[1]** variable instead of **Velocity[0]**. The **Reynolds1000** case CFD simulation is now ready to run.

In the folder **Calculation management**, go directly to the **Prepare batch calculation** sub-folder. By default, the calculation-restart is disabled in **Start/Rerstart** and so this sub-folder does not need
to be visited. In the Prepare batch calculation sub-folder panel (Figure III.39), select the script file which is by default runcase by clicking on the folder button . A pop-up window opens up, select the runcase file and click on Open. For the Run type, select Standard (calculations without mesh or partitions import or pre-processing) and choose 1 processor for the Number of processes. Leave the Run id and Threads per task to their default value.

Figure III.39: Batch calculation settings.

Press the Start calculation button to run Code_Saturne.

3.2 Running and Analysing the Simulation

Upon firing the Code_Saturne run from the GUI, confirmation that Code_Saturne is running, “Code_Saturne is running”, appears in the window from which the GUI was started. This is followed by further messages indicating what stage the calculation is in, from “Preparing calculation data” to “Saving calculation results”.

Wait for the calculations to complete and enter the Reynolds1000/RESU directory or open its contents via a browser to inspect its contents. Explanations of the meaning and purpose of the different output files and directories resulting from a run are available in the Code_Saturne Users Guide [3] and are not repeated here. Instead, we highlight individual items which relate to this specific run and how the output information should always be used in order to analyse a calculation.

The RESU directory now contains a new directory named after the date and time at which the calculation was started, expressed on a 24 hours clock in the format “YearMonthDay-HourMinutes”.

In this latter directory, notice in particular:

- The profile files XVel_YaxisCenterLine and YVel_XaxisCenterLine, written in .csv format,
- The listing file,
- The monitoring and postprocessing directories.

With your text editor, open one of the profile files, to inspect its structure. The requested variables listed in column format as a function of the \((x, y, z)\) coordinates of the points along the profile line defined in the GUI.
Open the listing file to check that the calculation has converged towards stable values. The minimum and maximum values of the solution variables and the solution residuals for these variables are listed in summary tables at each iteration. By comparing the initial and final values in the Drift column for each variable, and their evolution throughout the calculation, check that the residuals have decreased by at least two orders of magnitude.

** INFORMATION ON CONVERGENCE  
----------------------------------------
<table>
<thead>
<tr>
<th>Variable</th>
<th>Rhs norm</th>
<th>N_iter</th>
<th>Norm. residual</th>
<th>Drift</th>
<th>Time residual</th>
</tr>
</thead>
<tbody>
<tr>
<td>c Velocity</td>
<td>0.24601E-03</td>
<td>24</td>
<td>0.46251E-02</td>
<td>0.10142E+00</td>
<td>0.25130E+02</td>
</tr>
<tr>
<td>c Velocity[X]</td>
<td>0.10057E+00</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>c Velocity[Y]</td>
<td>0.85128E-03</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>c Velocity[Z]</td>
<td>0.82487E-29</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>c Pressure</td>
<td>0.18599E-03</td>
<td>22</td>
<td>0.58361E-08</td>
<td>0.10000E+01</td>
<td>0.90909E+01</td>
</tr>
</tbody>
</table>

Figure III.40: Code_Saturne Convergence history from the listing file, after 1 iteration.

** INFORMATION ON CONVERGENCE  
----------------------------------------
<table>
<thead>
<tr>
<th>Variable</th>
<th>Rhs norm</th>
<th>N_iter</th>
<th>Norm. residual</th>
<th>Drift</th>
<th>Time residual</th>
</tr>
</thead>
<tbody>
<tr>
<td>c Velocity</td>
<td>0.28275E-03</td>
<td>18</td>
<td>0.10307E-07</td>
<td>0.67436E-15</td>
<td>0.13487E-06</td>
</tr>
<tr>
<td>c Velocity[X]</td>
<td>0.31555E-15</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>c Velocity[Y]</td>
<td>0.35882E-15</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>c Velocity[Z]</td>
<td>0.55548E-44</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>c Pressure</td>
<td>0.39566E-10</td>
<td>0</td>
<td>0.67068E-07</td>
<td>0.67068E-07</td>
<td>0.12750E-15</td>
</tr>
</tbody>
</table>

Figure III.41: Code_Saturne Convergence history from the listing file, after 400 iterations.

Inspect one of the monitoring files in the monitoring directory. The value of each variable is recorded in separate files at all the monitoring points defined in the GUI and as a function of iteration. Finally, verify that the RESULTS.case file containing all the mesh and output variables information for post-processing has been output in the postprocessing directory.

Having validated the calculation itself, you can now proceed to examining and post-processing the results by returning to the SALOME platform.

### 3.3 Post-processing the Results

In SALOME, select ParaViS from the drop-down module selector in the top menu bar. The name of the module will add itself to the Object Browser list and the ParaView-specific panels and menus will be activated, including a new ParaView scene viewer window.

Before loading the run data in ParaView, modify the default colour schemes. To visualise scenes on a white background, which is advantageous for printing, change the default settings for ParaView by clicking on Preferences in the top menu. For ParaViS, in the ParaView Settings tab, change the colours to black for the foreground and text, white for the background, and grey for the surfaces,
as shown in Figure III.42 below. Press \texttt{Apply} to enforce the new settings and \texttt{OK} to validate your selection when you are satisfied with the changes.

![Figure III.42: Setting the colour preferences for ParaView.](image)

The data to post-process can now be imported in ParaView.

First, you are going to load the monitoring point data to validate that a stable, steady-state solution has been obtained. From the top menu bar, select \texttt{File} \rightarrow \texttt{Open ParaView File}. In the \texttt{Open File} pop panel, navigate to the \texttt{monitoring} sub-directory and select three files. The multiple selection is performed by holding the \texttt{Ctrl} key down as you select the files. Select \texttt{probes\_Pressure.csv}, \texttt{probes\_Velocity[X].csv}, and \texttt{probes\_Velocity[Y].csv}. Close the panel by clicking \texttt{OK}. The three sets of data are now displayed under their file name in the \texttt{Pipeline Browser}. For each file, press \texttt{Apply} to load the data. By default, the data is visualised in tabular form in the ParaView scene viewer. Close the View by clicking the cross button \texttt{x} at the top, right hand corner of the view. Then, click on the \texttt{button to replace the data view by a Create View menu (Figure III.43).}
Click on the \texttt{Line Chart View} button and prepare the line plots, starting with the pressure data. Make the data visible for that file by clicking on the eye symbol \(\varnothing\) next to the file name in the Pipeline Browser. Go to the Display tab and set Attribute Type to Row Data. In the Series Parameters category, unselect the iteration variable. The variables 1, 2, 3, and 4, representing the pressure at each iteration at the four monitoring points you specified should be selected and the graphs of their evolution as a function of the number of iterations (time steps) should now be displayed in the view window. For clarity, change the legend for each variable. In the Series Parameters category, left-click on the row for variable 1 to select it. Double left-click on the name of that variable, in the Legend Name column to edit it. Change the name of each variable to Pressure (1), Pressure (2), Pressure (3) and Pressure (4) (Figure III.44 and III.45).
In the View window’s tool bar Change the **Bottom Axis** title to **Iteration**, and the **Left Axis** title to **Pressure (Pa)** (Figure III.46). The graph shows that the pressure at the four monitoring points becomes constant after about 100 iterations (Figure III.46).
Repeat the same operation for the remaining velocity files. For both files, change the left axis title to **Velocity (m/s)**. For the X velocity, change the variable names to **X Vel (i)**, where \( i = 1, 2, 3, \) and 4. Keep the default colours for each line so that they are consistent for each monitoring point. For the Y velocity, change the variable names to **Y Vel (i)**, where \( i = 1, 2, 3, \) and 4. The graphs also confirm that the velocity components at the four monitoring points settle to a stable, constant value after about 100 iterations, indicating that the calculations are converged (Figure III.47).

Figure III.46: Pressure at the four monitoring points as a function of iteration.

Figure III.47: Velocity components \((X \text{ (left)}, Y \text{ (right)})\) at the four monitoring points as a function of iteration.
You may now post-process the results from the calculations. From the top menu by selecting **File** > **Open ParaView File**, or by right-clicking on the **builtin** object in the Pipeline Browser panel and selecting **Open**, access the pop-up **Open File** panel and choose the **RESULTS.case** file from your run. The **RESULTS.case** object will be added to the **Pipeline Browser** and its contents displayed in the **Object Inspector** panel underneath. Press **Apply** to load the data in ParaViS.

Next, to extract the computed data in terms of fluid domain and boundary data, in the top menu bar click on **Filters** > **Alphabetical** > **Extract Block**. The new object **ExtractBlock1** now appears in the **Pipeline Browser**. With the object highlighted, move to the **Block Indices** panel of the **Properties** tab. In the data tree, under **Root**, select **Fluid domain** to visualise the data calculated at the cell centres inside the cavity walls, as shown in Figure III.48.

![Figure III.48: Extract the Fluid domain block in ParaViS.](image)

Left click on the **Display** tab of the **Object Inspector**. In the category **Coloring**, two drop-down menus allow you to choose the variable to visualise. Choose **Velocity** in the first one and **Magnitude** in the second one, as shown in Figure III.49.

![Figure III.49: Velocity magnitude variable chosen to color the fluid domain.](image)

Contour plots of the velocity magnitude are then displayed in the **ParaView scene viewer**. By default, the colour scale is set to the **Cool to Warm** (RGB) colour scheme which is inadequate.

![Figure III.50: Color map editor.](image)

To modify the visualisation colour scheme, click on the **Choose Preset** button (see Figure III.50) and select **Blue to Red Rainbow** (HSV) in the list of **Preset Color Scales**. Press **Apply** and then **Close**.
Press the \text{Save current display settings values as default} button to save the changes. The contour plot of velocity magnitude is now updated for the new colour scale (Figure III.52).

The contour plot indicates that there is a zone of higher velocity flow defined by the green and red zones surrounded by lower and no-velocity regions in blue. Consistent with the chosen boundary conditions, the maximum velocity is equal to 1.0 m/s at the top, sliding wall and decreases to zero at the other, non-slip walls.

With the HSV colour scale, the contour plot is now clear, with the different levels clearly differentiable, but the image looks tessellated. As \text{Code_Saturne} outputs data at cell centres in the \textit{results} file, in ParaView each mesh cell is painted with a pixel of colour corresponding to the exact value of the variable in the cell. Whilst this cell-data visualisation mode is correct to examine exact values at cell centres, it can yield ragged images unrepresentative of the solution’s higher-order spatial accuracy and cannot be used in ParaView to generate vectors and streamline plots. Instead, to produce smoother
images, and vector and streamline plots, the cell-data can be interpolated to cell vertices (corners) using the ParaView \texttt{CellDatatoPointData} filter.

Having clicked on \texttt{ExtractBlock1} in the Pipeline Browser to select it, in the top menu bar click on \texttt{Filters} \textit{Alphabetical} \texttt{CellDatatoPointData}. The new object, which now stores the \texttt{RESULTS.case} data interpolated to cell vertices, is added in the Pipeline Browser (Figure III.53).

![Figure III.53: \texttt{CellDatatoPointData} filter added to the Pipeline Browser.](image)

With \texttt{CellDatatoPointData} selected in the Pipeline Browser, click \texttt{Apply} in the \texttt{Properties} tab of the \texttt{Object Inspector} to bring up the smoothed-out contour plot in the \texttt{ParaView} scene viewer.

![Figure III.54: \texttt{ParaView} short-cuts.](image)

Picking the different variables to map, either from the \texttt{Display} tab in the Object Inspector for the \texttt{CellDatatoPointData} object or directly from the drop-down list in the menu bar for contour plots at the top of the \texttt{SALOME} window (see Figure III.54), create contour plots for the \textit{X} and \textit{Y} velocity components, as shown in Figure III.55 below. The plots use a \texttt{Surface} mode of \texttt{Representation}, which can be selected either from the \texttt{Display} tab and \texttt{Style} category, or directly from the drop-down list in the contour plot menu bar (see Figure III.54).

![Figure III.55: Contour plots of the \textit{X} (left) and \textit{Y} (right) velocity components.](image)

As expected, the maximum velocity in the \textit{X} direction occurs at the top wall, where it nears 1.0 \textit{m/s} (Figure III.55 (left)). The locations of the blue (flow in the negative direction) and red (flow in the positive direction) areas in both plots (Figure III.55) indicate that the flow is entrained by the top wall in a clockwise circular motion inside the box. These plots are useful to verify overall velocity magnitudes and patterns but, to get a better visualisation of the flow, one can create 2D glyphs which show fluid particles trajectories.

To create the 2D glyphs, select \texttt{CellDatatoPointData} in the Pipeline Browser and click on \texttt{Filters} \textit{Common} \texttt{Glyph} in the top menu. The \texttt{Glyph1} object is then added under \texttt{CellDatatoPointData}
in the Pipeline Browser. Keep the CellDatatoPointData object selected and visible ( icon in bold next to the object). In the Object Inspector, select the Properties tab. In the Glyph Source category, choose 2D Glyph in the Glyph Type drop-down list. In the Scaling category, choose vector in the Scale Mode drop-down list (see Figure III.56).

![Figure III.56: Glyph type and Scale Mode.](image)

In the Object Inspector, select the Display tab. In the Coloring category, choose Solid Color and set this color to black by clicking on the Edit color map button (see Figure III.57). Press OK to finish.

![Figure III.57: Glyph coloring.](image)

Conclude by pressing the Apply button. The result should look like Figure III.58.
To make it possible to create and locate the streamlines with regard to the Cavity, you are going to create a combined image showing the streamlines superimposed on top of the mesh.

First, create the streamlines. With CellDatatoPointData selected in the Pipeline Browser, click on Filters > Common > Stream Tracer in the top menu. The StreamTracer object is then added under CellDatatoPointData in the Pipeline Browser. Keep the CellDatatoPointData object selected and visible (icon in bold next to the object). In the Object Inspector, select the Display tab. In the Coloring category, choose Solid Color in the Color by drop-down list. In the Style category, change the Representation to Wireframe and decrease the Opacity to 0.1 (Figure III.59).

The mesh lines displayed in the ParaView scene Viewer should be painted in faint black colour.

Next, select the StreamTracer1 object in the Pipeline Browser. In the Object Inspector, select the Properties tab and modify the default settings for Seeds. Select Point Source for the Seed Type, and modify the X and Y coordinates of the seed point to 0.15 and 0.05, respectively. Request 60,000 points (Number of Points) and a radius of 1.0 (Radius), as shown in Figure III.60 below. Press Apply to validate your changes.
In the Display tab, choose **Velocity** and **Magnitude** from the **Color by** drop-down lists of category **Color** and visualise the streamlines coloured by velocity magnitude, on top of the computational mesh (Figure III.61).

In addition to the general circular motion which had been inferred from the contour plots of velocity components, the streamlines now reveal two recirculation pockets in the bottom corners of the Cavity.

To further compare the calculated data with the available data from [4], create line plots of the X and Y velocities along the Y-axis and X-axis profiles which were set in the **Code_Saturne** GUI. Using the menu **File > Open ParaView File**, import both the `XVel_YaxisCentreLine.csv` and the `YVel_XaxisCentreLine.csv` files. To read in the data and ensure that the velocities are correctly interpreted as real numbers by ParaView, in the Properties tab, define the **Field Delimiter Characters** as ` ' , `: a blank space followed by a comma. Also, select **Merge Consecutive Delimiters** (Figure III.62).
Go to the Information tab and verify that all the data arrays are read in as **double**. The steps to create and label the graphs are similar to those already detailed for the graphs of the monitoring points data and they are not repeated here.

For direct comparisons with the plots from [4], for the graphs of \( X \) velocity, plot the \( Y \) coordinate as a function of \( \text{Velocity}[0] \). For the graphs of \( Y \) velocity, plot \( \text{Velocity}[1] \) as a function of the \( X \) coordinate. The reference data provided in [4] for the fine mesh results (129x129) is listed in Section 5. The data can be copied to a .csv file which is then imported in ParaView at the same time as the \( \text{XVel}\_\text{YaxisCentreLine}\_\text{csv} \) and the \( \text{YVel}\_\text{XaxisCentreLine}\_\text{csv} \) files. Set the plots so that the \textit{Code_Saturne} and the reference results are visualised in the same view, and display the reference points with a marker rather than a line. Each data set can then be compared directly with the reference results (Figure III.63).

![Figure III.62: Specifying the velocity profiles.](image)

![Figure III.63: Comparison between \textit{Code_Saturne} (solid lines) and reference [4] results (circles). \( Y \) coordinate versus \( X \) velocity (left) and \( Y \) velocity versus \( X \) coordinate (right).](image)
Overall, good agreement is obtained with the reference results [4], even though the results with Code_Saturne were obtained on a coarser mesh. Running on a finer mesh would make it possible to capture the velocity extremas with increased accuracy.

4 References

[1] www.salome-platform.org


5 Reference Data from [4]

To make it easier to replicate the comparative plots of the Code_Saturne and reference results, the data from [4] (Tables I and II) at Re = 1000 is reproduced in Figure III.3 below. The reference results listed in [4] were obtained on a fine, 129x129 mesh.

<table>
<thead>
<tr>
<th>Y (m)</th>
<th>U_x (m/s)</th>
<th>X (m)</th>
<th>U_y (m/s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.00000</td>
<td>1.00000</td>
<td>1.00000</td>
<td>0.00000</td>
</tr>
<tr>
<td>0.97660</td>
<td>0.65928</td>
<td>0.96880</td>
<td>-0.21388</td>
</tr>
<tr>
<td>0.96880</td>
<td>0.57492</td>
<td>0.96090</td>
<td>-0.27669</td>
</tr>
<tr>
<td>0.96090</td>
<td>0.51117</td>
<td>0.95310</td>
<td>-0.33714</td>
</tr>
<tr>
<td>0.95310</td>
<td>0.46604</td>
<td>0.94530</td>
<td>-0.39188</td>
</tr>
<tr>
<td>0.85160</td>
<td>0.33304</td>
<td>0.90630</td>
<td>-0.51550</td>
</tr>
<tr>
<td>0.73440</td>
<td>0.18719</td>
<td>0.85940</td>
<td>-0.42665</td>
</tr>
<tr>
<td>0.61720</td>
<td>0.05702</td>
<td>0.80470</td>
<td>-0.31966</td>
</tr>
<tr>
<td>0.50000</td>
<td>-0.06080</td>
<td>0.50000</td>
<td>0.02526</td>
</tr>
<tr>
<td>0.45310</td>
<td>-0.10648</td>
<td>0.23440</td>
<td>0.32235</td>
</tr>
<tr>
<td>0.28130</td>
<td>-0.27805</td>
<td>0.22660</td>
<td>0.33075</td>
</tr>
<tr>
<td>0.17190</td>
<td>-0.38289</td>
<td>0.15630</td>
<td>0.37095</td>
</tr>
<tr>
<td>0.10160</td>
<td>-0.29730</td>
<td>0.09380</td>
<td>0.32627</td>
</tr>
<tr>
<td>0.07030</td>
<td>-0.22220</td>
<td>0.07810</td>
<td>0.30353</td>
</tr>
<tr>
<td>0.06250</td>
<td>-0.20196</td>
<td>0.07030</td>
<td>0.29012</td>
</tr>
<tr>
<td>0.05470</td>
<td>-0.18109</td>
<td>0.06250</td>
<td>0.27485</td>
</tr>
<tr>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00000</td>
</tr>
</tbody>
</table>

Table III.3: X velocity versus Y at the vertical mid-plane and Y velocity versus X at the horizontal mid-plane. Data reproduced from [4]. Re = 1000. 129x129 mesh.