Code_Saturne documentation

Code_Saturne version 2.0 tutorial -
Fluid-structure interaction

contact: saturne-support@edf.fr
Synthèse

Ce document décrit la mise en œuvre pratique de calculs d’interaction fluide-structure avec Code_Saturne. Les points principaux abordés sont les suivants :

- comment effectuer un calcul sur un maillage mobile à l’aide du module ALE (Arbitrary Lagrangian-Eulerian);
- comment paramétrer l’algorithme de déformation de maillage et comment définir les conditions limites pour un maillage mobile ;
- comment imposer manuellement les déplacements du maillage ou comment associer un système “masse-ressort” à certaines frontières mobiles du domaine de calcul ;
- comment effectuer des post-traitements basiques pour des calculs de ce type.

Ce didacticiel a été créé pour version 2.0 de Code_Saturne. Des ajustements pourraient s’avérer nécessaires pour des versions plus récentes.
Executive Summary

The present document aims at describing the use of the main features of Code_Saturne available for fluid-structure interaction calculations. The key points addressed here concern:

- performing a calculation on a mobile mesh using the ALE (Arbitrary Lagrangian-Eulerian) framework;
- how to parameter the mesh deformation and properly define the parameters of the boundary conditions for a mobile mesh;
- how to impose the mesh deformations or how to link a solid body in the fluid domain to a mass-spring system (with fluid-structure coupling);
- how to perform some basic post-processing related to such calculations.

The tutorial has been created based on Code_Saturne 2.0. Adjustments might be needed for other versions.
Sommaire / Summary

AVERTISSEMENT / CAUTION 1

Synthèse 2

Executive Summary 3

1 Objectives 5

2 Description of the test case 5

3 Tutorial 5

3.1 How to setup a case involving fluid-structure interactions (GUI) 8
3.1.1 Setting up and running the test case 8
3.1.2 Post-processing 8

3.2 How to setup a case involving fluid-structure interactions with user subroutines 48
3.2.1 Standard parameters definition 48
3.2.2 Boundary conditions 48
3.2.3 Definitions specific to the ALE 48
3.2.4 Mobile mesh boundary conditions 49
3.2.5 Structure definition 49
3.2.6 Mesh viscosity definition 51
3.2.7 Run case script 51

3.3 How to impose the displacement of the structure (GUI) 53
3.4 How to impose the displacement of the structure (user subroutines) 60
3.5 How to impose the velocity of the structure (GUI) 61
3.6 How to impose the velocity of the structure (user subroutines) 68
3.7 How to compute the force acting on the structure (user subroutines) 69
3.8 How to control the convergence of the internal fluid-structure coupling procedure (advanced user) 70
1 Objectives

The present tutorial aims at describing the use of the main features of Code_Saturne available for fluid-structure interaction calculations. The key points addressed here concern:

- performing a calculation on a mobile mesh using the ALE (Arbitrary Lagrangian-Eulerian) framework;
- how to parameter the mesh deformation and properly define the parameters of the boundary conditions for a mobile mesh;
- how to impose the mesh deformations or how to link a solid body in the fluid domain to a mass-spring system (with fluid-structure coupling);
- how to perform some basic post-processing related to such calculations.

The tutorial has been created based on Code_Saturne 2.0. Adjustments might be needed for other versions.

2 Description of the test case

The present test case focuses on the numerical simulation of the transverse response of an elastically mounted cylinder subjected to vortex-induced vibrations (VIV). A sketch of the flow setup is provided in figure 1: the flow is uniform and the mechanical dynamics of the cylinder is modeled by a simple mass-spring system.

The physical parameters are chosen so that the system configuration may be simulated with a reasonable computational cost. The Reynolds number based on the cylinder diameter is taken to be 100 so that the flow is laminar and a 2D calculation may then be able to capture the whole features of the phenomenon. The parameters of the simulation are given throughout the tutorial but a sum-up is provided in table 1.

As shown in figure 2 the calculation domain is 2D with only one cell in spanwise direction.

3 Tutorial

The tutorial itself contains several parts. The first point, entitled “How to setup a case involving fluid-structure interactions with the GUI”, needs to be performed before looking at the other items. The content of the present tutorial is the following:

How to setup a case involving fluid-structure interactions with the GUI ... 8
How to setup a case involving fluid-structure interactions with user subroutines ... 48
How to impose the displacement of the structure (GUI) ... 53
How to impose the displacement of the structure (user subroutines) ... 60
How to impose the velocity of the structure (GUI) ... 61
How to impose the velocity of the structure (user subroutines) ... 68
How to compute the force acting on the structure (user subroutines) ... 69
How to control the convergence of the internal fluid-structure coupling procedure ... 70
Table 1: Simulation parameters of the vortex-induced vibration test case.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$D$</td>
<td>cylinder diameter</td>
<td>0.025 m</td>
</tr>
<tr>
<td>$L$</td>
<td>cylinder spanwise length</td>
<td>0.005 m</td>
</tr>
<tr>
<td>$U_\infty$</td>
<td>inflow velocity</td>
<td>0.004 m.s$^{-1}$</td>
</tr>
<tr>
<td>$\rho$</td>
<td>density</td>
<td>1000 kg.m$^{-3}$</td>
</tr>
<tr>
<td>$\mu$</td>
<td>dynamic viscosity</td>
<td>0.001 Pa.s</td>
</tr>
<tr>
<td>$m$</td>
<td>cylinder mass (kg)</td>
<td>see $m^*$</td>
</tr>
<tr>
<td>$k$</td>
<td>cylinder stiffness (N.m$^{-1}$)</td>
<td>see $k^*$</td>
</tr>
<tr>
<td>$m^*$</td>
<td>normalized mass</td>
<td>3.3</td>
</tr>
<tr>
<td>$k^*$</td>
<td>normalized stiffness</td>
<td>12.02</td>
</tr>
</tbody>
</table>

Figure 1: Sketch of the vortex-induced vibration test case.

Figure 2: Overview of the computational domain.
Figure 3: View of the mesh.
3.1 How to setup a case involving fluid-structure interactions (GUI)

3.1.1 Setting up and running the test case

The first step is to create the Code_Saturne case. The following command creates a study directory, named Tuto_VIV, and a new case, referred to as VIV:

```
code_saturne create -s Tuto_VIV -c VIV
```

It is then necessary to copy the mesh file `mesh_viv.des` into the mesh directory of the study whose path is Tuto_VIV/MESH (the mesh file is located into the tutorial directory Tutorial_Files). Note that the mesh `mesh_viv.des` is rather coarse so that it will lead to short calculations. Another mesh, referred to as `mesh_viv_file.des`, is also available. It will give more accurate results but will require more computational time. One may first start with the coarse mesh to setup the test case and then switch to the fine one.

The graphical user interface (GUI) can now be started. Just go into the DATA directory of the case (with the path Tuto_VIV/VIV/DATA/) and type in the command:

```
./SaturneGUI
```

in order to run the GUI. The next steps and commentaries are provided in figures 4 to 41.

3.1.2 Post-processing

As concern calculations with a mobile mesh, basic flow visualization can be performed. As pointed out during the tutorial (in figure 37), as long as the outputs are based on a deformable mesh one may visualize the mesh deformations.

Some others valuable outputs related to the structure motions are also available in the working directory of the run. These data are copied into the RESU directory of the case, in the HIST directory corresponding to the run:

- go into the history directory of the run which should be Tuto_VIV/VIV/RESU/HIST.010100. The last eight digits corresponds to the date and time at which the calculation has started;
- a series of data files are available on the time history of the acceleration (`str_acceleration_x.dat`), velocity (`str_vitesse_x.dat`), displacement (`str_deplacement_x.dat`) of the structure and on the fluid forces acting on it (`str_force_x.dat`). Note that each direction, x, y and z, are treated in separated files;
- these data can easily be plotted using standard plotting software. The following command shows how to plot the displacement of the structure in the y-direction as a function of time, using the plotting tool `xmgrace`:

```
xmgrace -block str_deplacement_y.dat -bxy 2:3
```

which plots the displacement in the y-direction as a function of time. The figure 42 illustrates the result that should be obtained. Remark that during the initialization period, for $t < 500$ s, the structure does not move. See figure 10 for the definition of the initialization period in the GUI.
Figure 4: VIV test case. First steps of the case opening.
Figure 5: VIV test case. First steps of the case opening.

Once the new data file is created, the interface automatically update a few informations such as the study name, the case name, the case directory and sub-directories.
Figure 6: VIV test case. First steps of the case opening.

Right click and expand the treeview.
1. Select «Meshes selection» in the treeview

2. Verify mesh selection
   The list of all the meshes available in the directory MESH appears in this window.
   The mesh needed in the present case is named «mesh_viv.des». Any other mesh should be removed from the list using the «Delete» button.

Figure 7: VIV test case. Meshes selection.
1. Select «Mesh quality criteria» in the treeview

2. Click on «Check mesh»
This action launches Code_Saturne preprocessor. In our case we will use the preprocessor to get the group listing of the mesh in order to enforce the boundary conditions later on.

Figure 8: VIV test case. Mesh quality criteria.
Figure 9: VIV test case. Mesh quality criteria.

Save the preprocessor listing
The preprocessor listing will be used for the enforcement of the boundary conditions in order to get the group listing.
Figure 10: VIV test case. Mobile mesh.

1. Select «Mobile mesh» in the treeview

2. Check «Mobile mesh» to turn on the ALE method

3. Set the number of iterations for initialization to 5000
   Before the mobile mesh feature is turned on, some iterations are needed with a static mesh in order to establish the flow around the cylinder.

4. Set the spatial distribution to «user formula»
   The spatial distribution of the mesh viscosity is going to be defined manually, by the user.
   Once this option is activated the button «Formula for the viscosity of mesh» below should turn green and give access to the formula editor.

5. Click on the «Formula...» button
Figure 11: VIV test case. Mobile mesh.

*Fill in the editor with the following formula for the mesh viscosity*

To avoid any issues with the mesh deformations in the vicinity of the cylinder, the mesh viscosity is set to a very high value ($10^{22}$) close to the cylinder. Away from the cylinder the value falls down to 1.

With this viscosity distribution, mesh cell deformations mainly occur away from the cylinder wall surface.

The other tabs of the editor («Predefined symbols» and «Examples») give some hints and syntax rules on how to write down a correct formula in the editor.
Figure 12: VIV test case. Turbulence models.

1. Select «Turbulence models» in the treeview

2. Set turbulence model to «No model (i.e. laminar flow)»
   We consider the case of a low Reynolds number flow around a cylinder: the flow is indeed laminar.
Figure 13: VIV test case. Fluid properties.

1. Select «Fluid properties» in the treeview

2. Set density to 1000 kg/m³
   Water density.

3. Set viscosity to 0.001 Pa.s
   The value of the viscosity is set with respect to the Reynolds number.
Figure 14: VIV test case. Initialization.

1. Select «Initialization» in the treeview

2. Set axial velocity to 0.004 m/s
1. Select «Definition of boundary regions» in the treeview

2. Click on «Import groups and...»
Instead of manually entering every color of the mesh, the boundary regions can be directly taken from the Preprocessor listing.

Figure 15: VIV test case. Definition of boundary regions.
Figure 16: VIV test case. Definition of boundary regions.
Figure 17: VIV test case. Definition of boundary regions.
Select «Boundary conditions» in the treeview
Once the boundary regions have been defined, the features of the boundary conditions now be given by the user.

Figure 18: VIV test case. Boundary conditions.
1. Select «BC_1»
   This boundary region corresponds to the cylinder wall.

2. Select «Internal coupling»
   This parameter permits to define how the boundary conditions is treated by the ALE.
   Using «Internal coupling», a mass-spring system is attached to the boundary region. The displacement of the mesh for this B.C. will be given by the displacement of the mass-spring system.

Figure 19: VIV test case. Boundary conditions.
1. **Select «BC_2»**
This boundary region corresponds to the upper and lower sides of the mesh.

2. **Select «Fixed boundary»**
The upper and lower sides of the mesh are fixed; frontiers discretized by a fixed mesh.

Figure 20: VIV test case. Boundary conditions.
1. Select «BC_3». This boundary region corresponds to the inlet.

2. Set inlet velocity to 0.004 m/s

3. Select «Fixed boundary». The inlet is a fixed frontier discretized by a fixed mesh.

Figure 21: VIV test case. Boundary conditions.
1. Select «BC_4»
This boundary region corresponds to the outlet.

2. Select «Fixed boundary»
The outlet is a fixed frontier discretized by a fixed mesh.

Figure 22: VIV test case. Boundary conditions.
Figure 23: VIV test case. Boundary conditions.

1. Select «BC_5»
This boundary regions front side of the domain.

2. Select «Sliding boundary»
The front side is a fixed frontier but the mesh may be deformable on this boundary region.
Figure 24: VIV test case. Boundary conditions.

1. Select «BC_6»
   This boundary regions back side of the domain.

2. Select «Sliding boundary»
The back side is a fixed frontier but the mesh may be deformable on this boundary region.
Select «Fluid structure interaction» in the treeview.

This interface permits to link the cylinder to a mass-spring system. Since in the boundary conditions tab we have defined a wall B.C., with the option «internal coupling», one structure, linked to BC_1, is visible in the structure definition window.

The parameters of the structure now need to be defined by the user.

Structure linked to BC_1.

Figure 25: VIV test case. Fluid structure interaction.
Figure 26: VIV test case. Fluid structure interaction.

Set the sub-iterations number to 20 and the precision to $10^{-3}$

These parameters control the iterative process used to converge the solution between the fluid and the structure.

Increasing the maximum number of sub-iterations or decreasing the precision increase the computational cost but in the other hand it may prevent the emergence of non-physical solutions.
Figure 27: VIV test case. Fluid structure interaction.

1. Select the structure #1

2. Set all these values to zero

   The cylinder is initially at rest: X, Y, Z equal 0 for the initial position.

   The equilibrium position of the cylinder is the origin: X, Y, Z equal 0 for the position of equilibrium.

   The cylinder has no initial velocity: Vx, Vy and Vz are equal to zero.
Figure 28: VIV test case. Fluid structure interaction.

Click on the «Mass matrix» button in order to define the mass matrix using the formula editor.
Fill in the editor with the following formula for the mass matrix.

There is no motion of the cylinder in the x and z direction. The mass is nonetheless still defined for those directions, but the force acting on the cylinder following x and z will be set to zero later on.

Figure 29: VIV test case. Fluid structure interaction.
Figure 30: VIV test case. Fluid structure interaction.

Click on the «Damping matrix» button in order to define the damping matrix using the formula editor.
Figure 31: VIV test case. Fluid structure interaction.
Figure 32: VIV test case. Fluid structure interaction.

Click on the "Stiffness matrix" button in order to define the stiffness matrix using the formula editor.
Figure 33: VIV test case. Fluid structure interaction.

Fill in the editor with the following formula for the stiffness matrix:
Figure 34: VIV test case. Fluid structure interaction.

Click on the «Force applied to the structure» button
Figure 35: VIV test case. Fluid structure interaction.

Fill in the editor with the following formula for the fluid forces:

To ensure there is no motion of the cylinder in the x and z direction, the fluid forces are set to zero for those directions.
Figure 36: VIV test case. Time step.

1. Select «Time step» in the treeview

2. Set time step to 0.1 s and number of iterations to 15,000
1. Select «Output control» in the treeview

2. Choose «Post-processing every...» and set the period to 500 iterations

3. Choose «deformable (with displacement)»
   This option must be activated to see the mesh deformation during the post-processing.

Figure 37: VIV test case. Output control.
1. Select «Prepare batch...» in the treeview

2. Set the number of processors you want to use

Figure 38: VIV test case. Prepare batch calculation.
Figure 39: VIV test case. Prepare batch calculation.
Figure 40: VIV test case. Prepare batch calculation.

Enter the filename and save it.
Figure 41: VIV test case. Prepare batch calculation.

Click on «Code_Saturne batch...» to run the calculation.
Figure 42: VIV test case. Time history of the structure displacement in the $y$ direction.
3.2 How to setup a case involving fluid-structure interactions with user subroutines

In order to fully define the present test case the following source files are required and must be copied in the SRC directory:

- **usini1.f90**: definition of the standard calculation parameters;
- **usclim.f90**: definition of the nature of the boundary conditions;
- **usalin.f90**: definition of the specific parameters of the ALE (to turn it on and provide the parameters of the internal fluid-structure coupling);
- **usalc1.f90**: definition of the behavior of the boundary regions when the mesh is mobile (fixed b.c., sliding b.c.,...);
- **usstru.f90**: to link a mass-spring structure to a boundary condition (“internal” fluid-structure coupling);
- **usvima.f90**: definition of the mesh viscosity.

3.2.1 Standard parameters definition

The standard calculation parameters are defined in the user source file **usini1.f90**. Nonetheless, these variables will not be described here since they fall outside the scope of a tutorial on fluid-structure interaction in Code_Saturne. However, as concern the outputs, one should not forget to set the keyword **ichrmd** to 1:

\[
ichrmd = 1
\]

in order to obtain output data on a deformable mesh. This feature makes possible to observe the mesh motions with a visualization software (Ensight or Paraview for instance).

The final source file is provided in the tutorial directory **Tutorial_Files/User_Sources**.

3.2.2 Boundary conditions

In a similar manner, even though a mobile mesh is used, the setup of the boundary conditions does not require any specific treatment in the file **usclim.f90**. This source file will not be described here.

The final source file is provided in the tutorial directory **Tutorial_Files/User_Sources**.

3.2.3 Definitions specific to the ALE

The features of the mobile mesh procedure, as well as those of the fluid-structure internal coupling algorithm, are defined in the user source file **usalin.f90**. For the present case the various parameters of the ALE should be set to the following values:

\[
iale = 1 \quad ! \text{turn on the ALE} \\
nalinf = 5000 \quad ! \text{number of iterations for fluid initialization} \\
nalimx = 20 \quad ! \text{number of sub-cycling iteration for fluid-structure interaction} \\
epalim = 1.d-8 \quad ! \text{relative precision of sub-cycling fluid-structure coupling} \\
iorutm = 0 \quad ! \text{isotropic mesh viscosity}
\]

As pointed out earlier in section 3.1 the key parameters are:
• **iale**: equals 0 when the mobile mesh feature is deactivated and 1 when it is activated;

• **nalinf**: is the number of iteration before the fluid-structure coupling is actually performed. Some iterations might indeed be necessary in order to have a well established flow before allowing any motion of the structure;

• **nalimx** and **epalim**: control the convergence of the iterative resolution of the fluid-structure coupling. These parameters should be chosen carefully to avoid any unphysical behavior of the solution (see for instance section 3.8);

• **iortvm**: defined whether the mesh viscosity is orthotropic (the mesh viscosity is a vector) or isotropic (the mesh viscosity is the same in all three directions).

The final source file is provided in the tutorial directory *Tutorial_Files/User_Sources*.

### 3.2.4 Mobile mesh boundary conditions

Since the mesh is mobile one can define if a boundary region is fixed or mobile. These definitions can be done in the source file *usalcl.f90*. For each face *ifac* of the mesh, the keyword **ialtyb(ifac)** can be set to **igliss** if the mesh is sliding, or to **ibfixe** if the mesh is fixed. In our present case, only boundary regions 10 and 11, which corresponds to the front and back frontiers of the mesh, are sliding:

```fortran
! --- For boundary faces of color 10 and 11 assign a sliding boundary call getfbr('10 and 11',nlelt,lstelt)
!==========
   do ilelt = 1, nlelt
      ifac = lstelt(ilelt)
      ialtyb(ifac) = igliss
   enddo
! --- prescribe elsewhere a fixed boundary call getfbr('not (10 or 11)',nlelt,lstelt)
!==========
   do ilelt = 1, nlelt
      ifac = lstelt(ilelt)
      ialtyb(ifac) = ibfixe
   enddo
```

The final source file is provided in the tutorial directory *Tutorial_Files/User_Sources*.

### 3.2.5 Structure definition

The mechanical properties of the structure and the definition of the forces acting on it can be provided by the user in the file *usstru.f90*.

The first subroutine **usstru** is called at the beginning of the calculation. It first permits to link a structure to a boundary region. For this structure a few parameters may be defined:

• **xstr0(idim,istructure)**: initial position of the structure (*idim* = 1, 2, 3 is the indice of the dimension and *istructure* is the number of the structure);

• **vstr0(idim,istructure)**: initial velocity of the structure;

• **xstreq(idim,istructure)**: equilibrium position of the structure;
• `aexxst`, `bexxst` and `cfopre`: advanced users may also change some parameters of the internal fluid-structure coupling procedure.

For the present test case the source code for this subroutine should read:

```fortran
! --- Assign faces of boundary region 1 to a structure #1
call getfbr('1',nlelt,lstelt)
do ilelt = 1, nlelt
   ifac = lstelt(ilelt)
   idfstr(ifac) = 1
enddo

! --- Some structure parameters
xstr0(2,1) = 0.d0
xstreq(2,1) = 0.d0
vstr0(3,2) = 0.d0

! --- Here one can modify the values of the prediction coefficients for
!     displacements anf fluid forces in internal FSI coupled algorithm.
aexxst = 0.5d0
bexxst = 0.0d0
cfopre = 2.d0

! --- Activation of structural history output
ihistr = 1
```

Then, in subroutine `usstr2` which is called at each time step the mass, stiffness and damping of the structure, as well as the forces acting on the structure can be defined:

- `xmstru(i,j,istr)`, `xcstru(i,j,istr)` and `xkstru(i,j,istr)` correspond to the mass, damping and stiffness matrices of the structure number `istr`;
- `forstr(idim,istr)` is the force acting on the structure. By default this vector is filled in with the fluid-forces but the user can modify these values if needed.

As concern the case treated in this tutorial the source code of the subroutine is given by:

```fortran
! --- Define matrices
istr = 1

cyl_m = 5.16e-3

cyl_k = 4.81e-4

xmstru(1,1,istr) = 1.
xmstru(2,2,istr) = cyl_m
xmstru(3,3,istr) = 1.

xcstru(1,1,istr) = 0.
xcstru(2,2,istr) = 0.
xcstru(3,3,istr) = 0.

xkstru(1,1,istr) = 1.
xkstru(2,2,istr) = cyl_k
xkstru(3,3,istr) = 1.
```
! --- Define forces acting on the structure
! --- only the force in the lift direction (2)
! --- is kept. The others are set to zero
istr = 1
forstr(1,istr) = 0.
forstr(3,istr) = 0.

! --- Structural time step
istr = 1
dtstr(istr) = dtcel(1)

The final source file is provided in the tutorial directory Tutorial_Files/User_Sources.

3.2.6 Mesh viscosity definition

As shown in figure 11, the mesh viscosity is set to a very high value close to the cylinder whereas it is equal to 1 away from the surface of the cylinder. This can be done using the source file usvima.f90 with the following code: in the declarations add the line,

    double precision rad, xr2, xcen, ycen

and the subroutine body should contain:

    if (ntcabs.eq.0) then
        rad = (0.025)**2
        xcen = 0.d0
        ycen = 0.d0
        do iel = 1, ncel
            xr2 = (xyzcen(1,iel)-xcen)**2 + (xyzcen(2,iel)-ycen)**2
            if (xr2.lt.rad) viscmx(iel) = 1.d10
        enddo
    endif

The aim here is to loop over all the cells (do iel = 1, ncel) and apply a given mesh viscosity (viscmx(iel)) according to the location of the center of the cell (xyzcen(1,iel) for x location and xyzcen(2,iel) for y location).

The viscosity is isotropic so that only viscmx(iel) needs to be defined. In the general case the three viscosity components, viscmx(iel), viscmy(iel) and viscmz(iel) have to be provided by the user.

The final source file is provided in the tutorial directory Tutorial_Files/User_Sources.

3.2.7 Runcase script

No special step need to be performed for the runcase when the mobile mesh features is turned on. Simply define the mesh used,

    MESH='mesh_viv.des'

and the number of processors required:
NUMBER_OF_PROCESSORS=3

The final script file is provided in the tutorial directory Tutorial_Files/User_Sources.
3.3 How to impose the displacement of the structure (GUI)

Instead of using the internal coupling to obtain the motions of the structure one may want to impose the displacement of the cylinder. The steps required in the GUI to perform such an operation are explained in figures 43 to 48. Note that the steps described in section 3.1 also need to be performed to setup the calculation.
1. Select «Boundary conditions»

2. Select «BC 1»
   This boundary region corresponds to the cylinder surface.

3. Select «Fixed displacement»
   The displacement of the mesh region is going to be given by the user. The formula editor will help us doing this.

4. Open the formula editor

Figure 43: Imposed displacement case. Boundary conditions.
Figure 44: Imposed displacement case. Boundary conditions.

**Fill in the editor with the following formula for the mesh displacement**

Here, the mesh displacement in the x and z direction is set to zero. The cylinder only undergoes a transverse motion following the y-axis.

The motion is harmonic with an amplitude equal to d/d (d is the cylinder diameter) and the period is set to 100 time steps.
Figure 45: Imposed displacement case. Mobile mesh.

1. Select «Mobile mesh»

2. Set the number of iterations for initialization to 0
   The flow initialization is skipped in the present example to spare some computational time.
Figure 46: Imposed displacement case. Time step.

1. Select «Time step» in the treeview

2. Set time step to 0.1 s and number of iterations to 500
   500 iterations correspond to 5 periods of the cylinder transverse periodic motion.
1. Select «Output control» in the treeview

2. Choose «Post-processing every...» and set the period to 10 iterations
The output period must be chosen small enough to ensure that the cylinder motions will be visible during the post-processing. Here we will have 10 snapshots of the flow field per period.

Figure 47: Imposed displacement case. Output control
Figure 48: Imposed displacement case. Prepare batch calculation.

1. Save the calculation data file

2. Click on «Code_Saturne batch...» to run the calculation
3.4 How to impose the displacement of the structure (user subroutines)

Copy the user-source usalcl.f90 in the source directory:

```bash
cp Tuto_VIV/VIV/SRC/REFERENCE/base/usalcl.f90 Tuto_VIV/VIV/SRC/.
```

Edit the source file and add in the “local variables” declaration zone the following line to add a variable for the displacement of the mesh:

```plaintext
double precision delta
```

Then, in the body of the subroutine, add the following source code in order to impose to each node a given displacement:

```plaintext
! displacement amplitude (the diameter is 0.025)
delta = (0.025d0/4.d0)*sin(2.d0*pi*ttcabs/100.d0/dtref)

! get the cell faces corresponding to color 1
call getfbr('1',nlelt,lstelt)

! loop over these faces and impose them some displacement
do ilelt = 1, nlelt
   ifac = lstelt(ilelt)
   do ii = ipnfbr(ifac), ipnfbr(ifac+1)-1
      inod = nodfbr(ii)
      if (impale(inod).eq.0) then
         depale(inod,1) = 0.d0 ! displacement /x
         depale(inod,2) = delta ! displacement /y
         depale(inod,3) = 0.d0 ! displacement /z
         impale(inod) = 1
      endif
   enddo
enddo
```

Note that the displacement is defined at the nodes, whereas the mesh velocity is defined at the cell faces (see section 3.6). Finally run Code_Saturne using the GUI.
3.5 How to impose the velocity of the structure (GUI)

The velocity of the cylinder may also be imposed. The steps required in the GUI to perform such an operation are explained in figures 49 to 54. Note that the steps described in section 3.1 also need to be performed to setup the calculation.
1. Select «Boundary conditions»

2. Select «BC. 1»
   This boundary region corresponds to the cylinder surface.

3. Select «Fixed velocity»
   The velocity of the mesh region is going to be given by the user. The formula editor will help us doing this.

4. Open the formula editor

Figure 49: Imposed velocity case. Boundary conditions.
Figure 50: Imposed velocity case. Boundary conditions.

**Fill in the editor with the following formula for the mesh velocity**

Here, the mesh velocity in the x and z direction is set to zero. The cylinder only undergoes a transverse motion following the y-axis. The motion is harmonic with an amplitude equal to \( d/4 \) (\( d \) is the cylinder diameter) and the period is set to 100 time steps.
Figure 51: Imposed velocity case. Time step.

1. Select «Time step» in the treeview

2. Set time step to 0.1 s and number of iterations to 500

500 iterations correspond to 5 periods of the cylinder transverse periodic motion.
Figure 52: Imposed velocity case. Mobile mesh.

1. Select «Mobile mesh»

2. Set the number of iterations for initialization to 0
   The flow initialization is skipped in the present example to spare some computational time.
Figure 53: Imposed velocity case. Output control.

1. Select «Output control» in the treeview

2. Choose «Post-processing every...» and set the period to 10 iterations

The output period must be chosen small enough to ensure that the cylinder motions will be visible during the post-processing. Here we will have 10 snapshots of the flow field per period.
Figure 54: Imposed velocity case. Prepare batch calculation.

1. Save the calculation data file

2. Click on «Code_Saturne batch...» to run the calculation
3.6 How to impose the velocity of the structure (user subroutines)

Copy the user-source `usalcl.f90` in the source directory:

```
cp Tuto_VIV/VIV/SRC/REFERENCE/base/usalcl.f90 Tuto_VIV/VIV/SRC/.
```

Edit the source file and add in the “local variables” declaration zone the following line to add a variable for the velocity of the mesh:

```
double precision deltav
```

Then, in the body of the subroutine, add the following source code in order to impose to each a given displacement:

```
! velocity magnitude (the diameter is 0.025)
deltav = (0.025/4.)*(2.*3.141596d0/100.d0/dtref)*cos(2.*3.141596d0*ttcabs/100.d0/dtref)

! get the cell faces corresponding to color 1
call getfbr('1',nlelt,lstelt)
!==========
! loop over these faces and impose them some velocity
do ilelt = 1, nlelt
   ifac = lstelt(ilelt)
   iel = ifabor(ifac)
   ialtyb(ifac) = ivimpo
   rcodcl(ifac,iuma,1) = 0.d0
   rcodcl(ifac,ivma,1) = deltav
   rcodcl(ifac,iwma,1) = 0.d0
endo
```

Note that velocity is defined at the cell faces, whereas the displacement is defined at the nodes (see section 3.4). Finally run `Code_Saturne` using the GUI.
3.7 How to compute the force acting on the structure (user subroutines)

As pointed out in section 3.1, some outputs specific to fluid-structure interactions, such as the forces acting on the structure, are available in the RESULTS.

However it should be noted that these quantities are calculated only when the fluid-structure coupling is activated (as shown in figure 25). When the displacement is imposed, the forces acting on the structure has to be computed by the user if needed.

To do so, copy the user-source usproj.f90 in the source directory. Edit the source file and add in the “local variables” declaration zone the following lines:

```fortran
double precision xfor(3)
```

Then, in the body of the subroutine, add the following source code:

```fortran
! set the force components to zero
do ii = 1, ndim
   xfor(ii) = 0.d0
enddo

! get the cells corresponding to the cylinder surface
call getfbr('1', nlelt, lstelt)
!==========

! loop over the cells to integrate the force
! over the structure surface
do ilelt = 1, nlelt
   ifac = lstelt(ilelt)
   ! update the force
   do ii = 1, ndim
      xfor(ii) = xfor(ii) + ra(iforbr + (ifac-1)*ndim + ii-1)
   enddo
enddo

! if the calculation is parallel, add the data from the
! other processes
if (irangp.ge.0) then
   call parrsm(ndim,xfor)
endif
```

We eventually get a variable xfor containing the three components of the force acting on the cylinder.
3.8 How to control the convergence of the internal fluid-structure coupling procedure (advanced user)

As shown in figure 26, the number of sub-iterations and the precision for the fluid-structure are some parameters that need to be adjusted to obtain a relevant solution. In order to control whether the iterative procedure has fully converged, it may be valuable to print out the number of sub-iterations that have indeed been performed.

To do so, get the source file `strdep.f90` in the source themselves of `Code_Saturne`. This file is not available in the `SRC/REFERENCE` directory of the case but may be found in the installation directory of `Code_Saturne`. Edit the `strdep.f90` and add at the end of the convergence test labelled “5.TEST DE CONVERGENCE” the following lines:

```fortran
if (irangp==1) then
    if (icv==1) then
        open(unit=impusr(1),file='cv.dat',position='append')
        write(impusr(1),*) ntcabs, italim
        close(impusr(1))
    endif
endif
endif
```

in order to create a data file named `cv.dat` containing the time history of the convergence of the coupling iterative process (`impusr(1)` is the unit id corresponding to the user file). The following `Code_Saturne` variables are used:

- `icv`: when equal to 1, indicates that the algorithm has indeed converged (else equal to 0);
- `ntcabs`: current time-step;
- `italim`: sub-iteration number.

The first column of the file contains the iteration number and second column corresponds to the number of sub-iteration required for the fluid-structure coupling to converge. Make sure this number is always lower than the maximum number of sub-iteration defined in the GUI (see figure 26), otherwise unwanted unphysical behaviors may occur.

To allow the file `cv.dat` to be copied in the result directory at the end of the calculation, one also need to add its name in the `runcase` script located in the `SCRIPTS` directory of the case.