EDF R&D

Fluid Dynamics, Power Generation and Environment Department
Single Phase Thermal-Hydraulics Group

6, quai Watier
F-78401 Chatou Cedex

Tel: 33 1 30 87 75 40
Fax: 33 1 30 87 79 16

MAY 2020

Code_Saturne documentation

Code_Saturne version 6.0 tutorial: full domain

contact: saturne-support@edf.fr

TABLE OF CONTENTS

Part I

Introduction

1 Introduction

1.1 *Code_Saturne* short presentation

Code_Saturne is a system designed to solve the Navier-Stokes equations in the cases of 2D, 2D axisymmetric or 3D flows. Its main module is designed for the simulation of flows which may be steady or unsteady, laminar or turbulent, incompressible or potentially dilatable, isothermal or not. Scalars and turbulent fluctuations of scalars can be taken into account. The code includes specific modules, referred to as "specific physics", for the treatment of lagrangian particle tracking, semi-transparent radiative transfer, gas, pulverized coal and heavy fuel oil combustion, electricity effects (Joule effect and electric arcs) and compressible flows. *Code_Saturne* relies on a finite volume discretization and allows the use of various mesh types which may be hybrid (containing several kinds of elements) and may have structural non-conformities (hanging nodes).

1.2 About this document

The present document is a tutorial for *Code_Saturne* version 6.0. It presents three simple test cases and guides the future *Code_Saturne* user step by step into the preparation and the computation of the cases.

The test case directories, containing the necessary meshes and data are available in the `examples` directory.

This tutorial focuses on the procedure and the preparation of the *Code_Saturne* computations. For more elements on the structure of the code and the definition of the different variables, it is higly recommended to refer to the user manual.

1.3 *Code_Saturne* copyright informations

Code_Saturne is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version. *Code_Saturne* is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

Part II

Full domain

1 Study description

1.1 Objective

The aim of this case is to tackle the merging of initially separate meshes into a single fluid domain. The questions of mesh joining and hanging nodes will be addressed. The test case will then be used to present more complex calculations, with time dependent variables and Fortran user routines.

1.2 Description of the configuration

The fluid domain is composed of three separate meshes, very roughly representing elements of a nuclear pressurized water reactor vessel:

- the downcomer
- the vessel's bottom
- the lower core plate and core

Figure II.1 represents the complete domain. The flow circulates from the top left horizontal junction to the right vertical outlet.

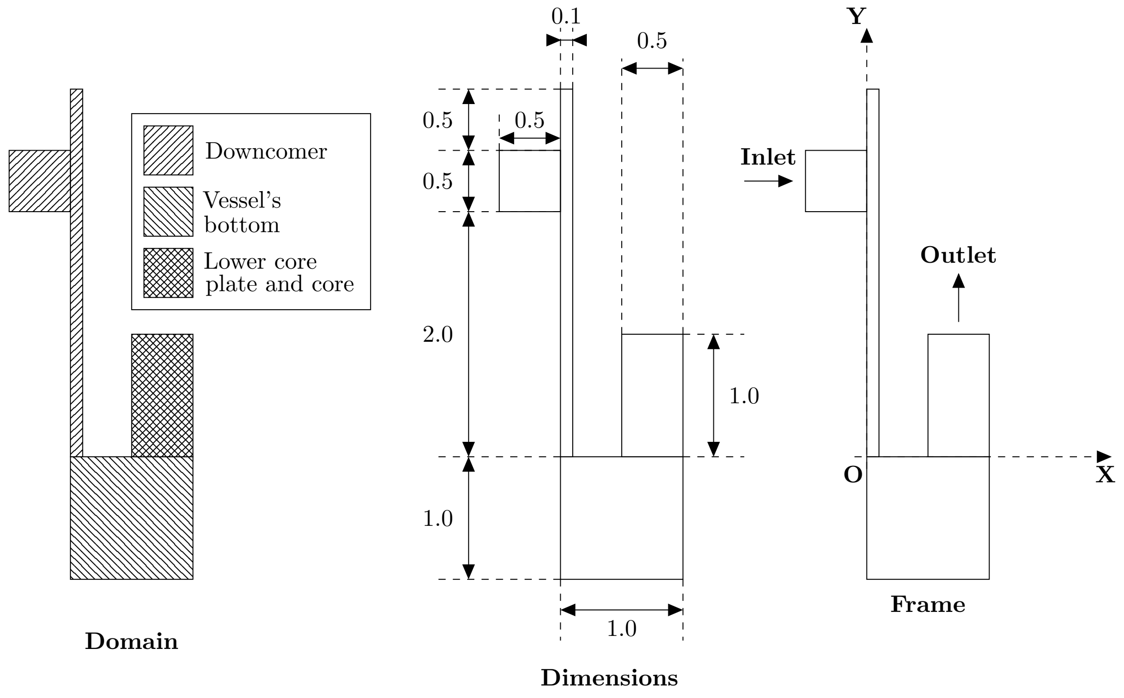


Figure II.1: Geometry of the complete domain

1.3 Characteristics

Characteristics of the geometry and the flow:

Height of downcomer	$H = 3.00\ m$
Thickness of downcomer	$E_d = 0.10\ m$
Diameter of the inlet cold branch	$D_b = 0.50\ m$
Height of vessel's bottom	$H_{fc} = 1.00\ m$
Width of vessel's bottom	$l_{fc} = 1.00\ m$
Height of core above the lower core plate	$H_{pic} = 1.00\ m$
Width of core above the lower core plate	$l_{pic} = 0.50\ m$
Inlet velocity of fluid	$V = 1\ m.s^{-1}$

Table II.1: Characteristics of the geometry and the flow

Physical characteristics of fluid:

The initial water temperature in the domain is equal to 20°C. The inlet temperature of water in the cold branch is 300°C. Water characteristics are considered constant[1] and their values taken at 300°C and $150 \times 10^5\ Pa$, except density which is considered variable in `case2` and `case3`:

- density: $\rho = 725.735\ kg.m^{-3}$
- dynamic viscosity: $\mu = 0.895 \times 10^{-4}\ kg.m^{-1}.s^{-1} = 8.951 \times 10^{-5}\ Pa.s$
- heat capacity: $C_p = 5\,483\ J.kg^{-1}.°\mathrm{C}^{-1}$
- thermal conductivity $= 0.02495\ W.m^{-1}.K^{-1}$

1.4 Mesh characteristics

Figure II.2 shows a global view of the mesh and some details of the joining zones, to show that *Code_Saturne* can deal with hanging nodes. This mesh is composed of 1 650 cells, which is very small compared to those used in real studies. This is a deliberate choice so that tutorial calculations run fast.

Type: block structured mesh

Coordinates system: cartesian, origin on the edge of the main pipe at the outlet level, on the nozzle side (figure II.2)

Mesh generator used: SIMAIL and mesh joining with the Preprocessor of *Code_Saturne* (in order to deal with hanging nodes)

Color definition: see figure II.3

1.5 Summary of the different calculations

Three cases will be studied with this geometry. The following table gives a summary of their different characteristics.

Remark: In this case, you must add three meshes which have to be joined. In order to join the three meshes, you must add a selection criteria in the box **Selection criteria** under the **Preprocessing** sub-folder. In this case, only faces of colors 5, 24 and 32 are liable to be joined (different colors can be entered on a single line, separated by comma).

You can verify the quality of your mesh by running a **Mesh quality criteria only** computation, which you can access through **Execution mode** in the **Mesh** heading.

[1]Which makes temperature a passive scalar ... but it is only for simplification purposes.

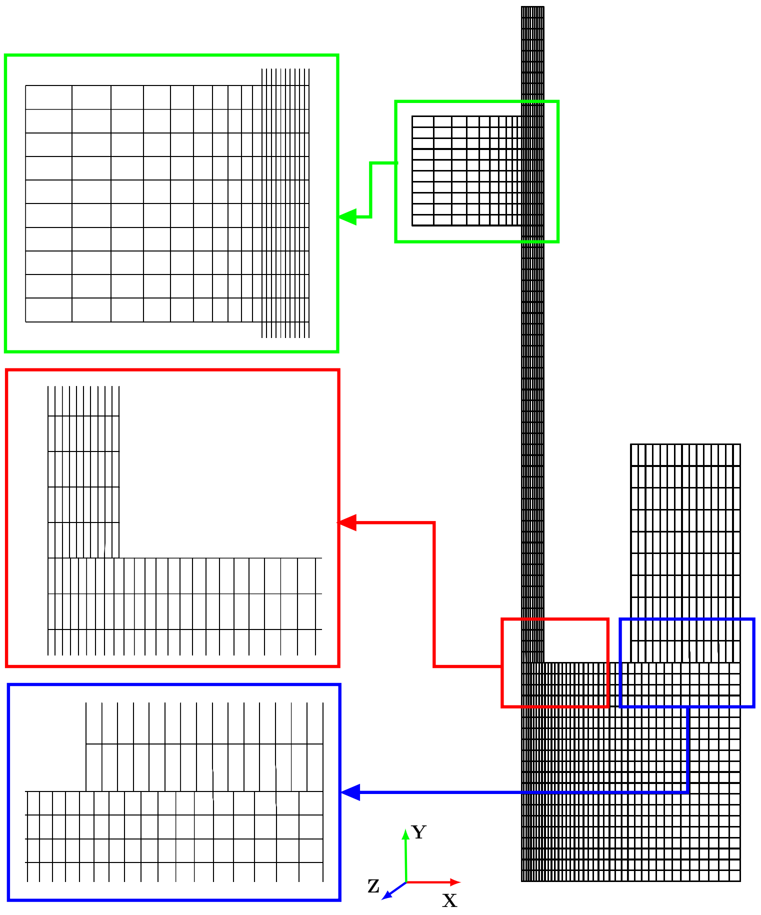


Figure II.2: View of the full domain mesh with zoom on the joining regions

CASE	**Characteristics**
CASE 1	Unsteady flow, additionnal passive scalar, output management
CASE 2	Same as case 1 with time dependent boundary conditions, fluid density depending on the temperature and calculation restart
CASE 3	Same as case 2 with head losses, parallelism and spatial average

Table II.2: Summary of the different calculations

2 CASE 1: Passive scalar with various boundary conditions and output management

2.1 Calculation options

Some options are similar to those of the `simple_junction` tutorial:

→ Turbulence model: $k - \epsilon$

→ Scalar(s): 1 - temperature

→ Physical properties: uniform and constant

The new options are:

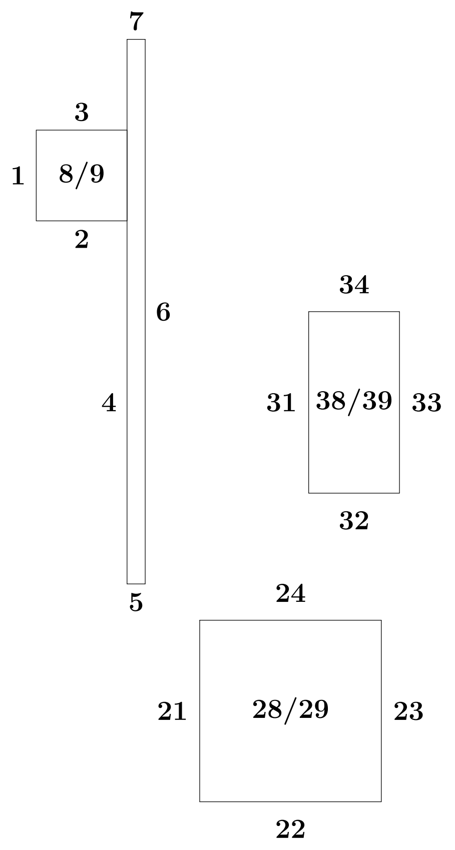


Figure II.3: Colors of the boundary faces

→ Flow type: unsteady flow

→ Time step: uniform and constant

→ Scalar(s): 2 - passive scalar[2], with diffusion coefficient 8.55 ($\times 10^{-5}$ $m^2.s^{-1}$)

→ Management of monitoring points

2.2 Initial and boundary conditions

→ Initialization: 20°C for temperature
10 for the passive scalar

The boundary conditions are defined in the user interface and depend on the boundary zone.

- **Flow inlet**: Dirichlet condition, an inlet velocity of 1 $m.s^{-1}$, an inlet temperature of 300°C and an inlet value of 200 for the passive scalar are imposed
- **Outlet**: default value
- **Walls**: velocity, pressure and thermal scalar: default value
  passive scalar: different conditions depending on the color and geometric parameters

In order to test the ability to specify boundary condition regions in the Graphical Interface, various conditions will be imposed for the passive scalar, as specified in the following table:

[2]It could correspond to a tracer concentration for instance.

Wall	Nature	Value
wall_1	Imposed value (Dirichlet)	0
wall_2	Imposed value (Dirichlet)	5
wall_3	Imposed value (Dirichlet)	0
wall_4	Imposed value (Dirichlet)	25
wall_5	Imposed value (Dirichlet)	320
wall_6	Imposed value (Dirichlet)	40

The **wall_1** to **wall_6** regions are defined as follows, through color references and geometric localization:

Label	Color and geometric parameters
wall_1	24 and $0.1 \leqslant x$ and $x \leqslant 0.5$
wall_2	2 or 3
wall_3	4 or 7 or 21 or 22 or 23
wall_4	6 and $y > 1$
wall_5	6 and $y \leqslant 1$
wall_6	31 or 33

Figure II.3 shows the colors used for boundary conditions and table II.3 defines the correspondance between the colors and the type of boundary condition to use.

Colors	Conditions
1	Inlet
34	Outlet
2 3 4 6 7 21 22 23 24 31 33	Wall
8 9 28 29 38 39	Symmetry

Table II.3: Boundary faces colors and associated references

2.3 Parameters and User routines

All parameters necessary to this study can be defined through the Graphical Interface without using any user Fortran files.

Calculation control parameters	
Pressure-Velocity coupling	SIMPLEC algorithm
Number of iterations	300
Reference time step	0.05
Output period for post-processing files	2

In order to join the separate meshes into a single domain, colors 5, 24 and 32 will have to be joined through the Graphical Interface.

2.4 Output management

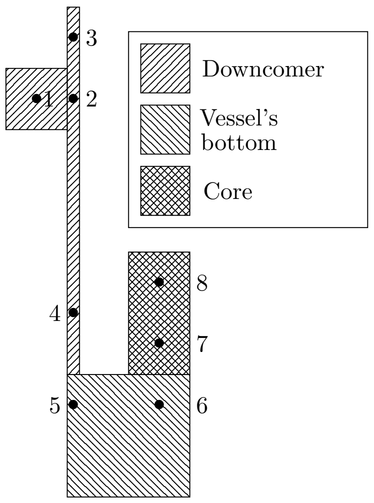
In this case, different aspects of output management will be addressed.

By default, in the Graphical Interface, all variables are set to appear in the listing, the post-processing and the chronological records. This default choice can be modified by the user.

In this case, the **Pressure**, the **Turbulent energy** and the **Dissipation** will be removed from the listing file.

The **Courant number** (CFL) and **Fourier number** will be removed from the post-processing results[3].

Eventually, probes will be defined for chronological records, following the data given in figure II.4. Then the **total pressure** will be deactivated for all probes.



Probe n^o.	x (m)	y (m)	z (m)
1	-0.25	2.25	0
2	0.05	2.25	0
3	0.05	2.75	0
4	0.05	0.5	0
5	0.05	-0.25	0
6	0.75	-0.25	0
7	0.75	0.25	0
8	0.75	0.75	0

Figure II.4: Position and coordinates of probes in the full domain

In addition the domain boundary will be post-processed. This allows to check the boundary conditions, and especially that of the passive scalar.

2.5 Results

Figure II.5 shows the boundary domain colored by the passive scalar boundary conditions. The different regions of boundary conditions defined earlier can be checked.

Figure II.6 presents results obtained at different times of the calculation. They were plotted from the post-processing files, with ParaView.

[3] This can be very useful to save some disk space if some variables are of no interest, as post-processing files can be large.

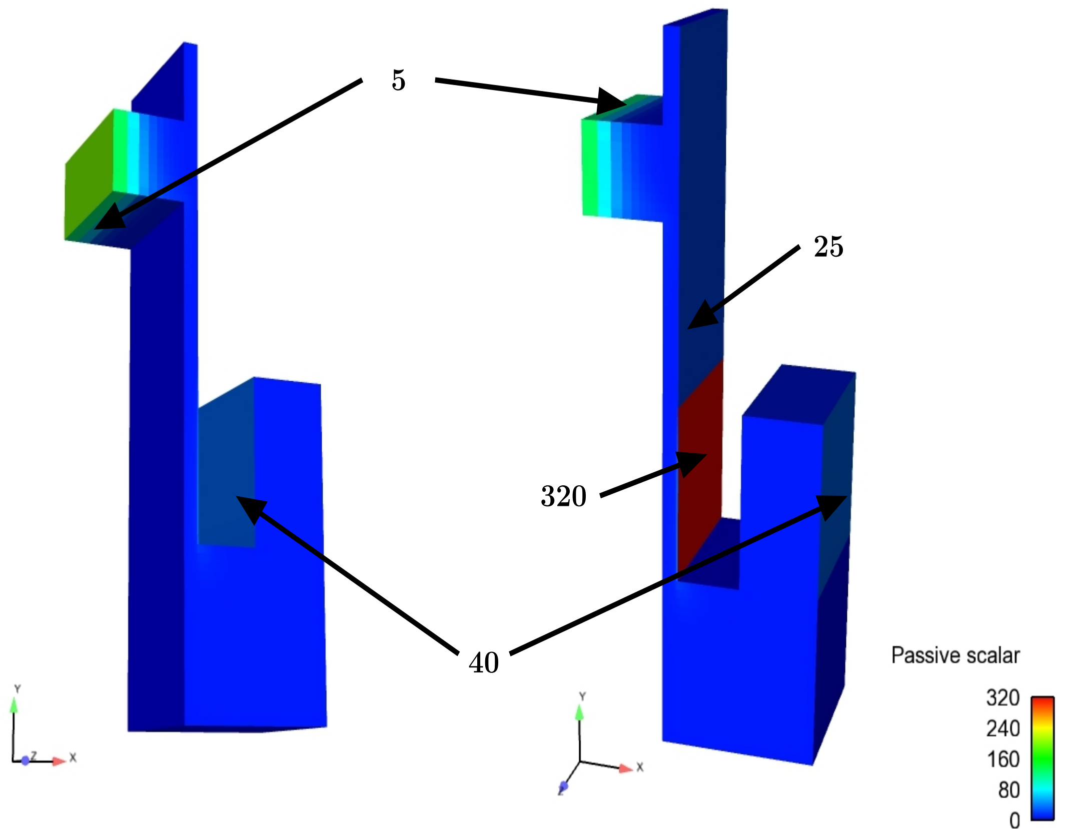


Figure II.5: View of the boundary domain colored by the scalar2 variable - Case 1

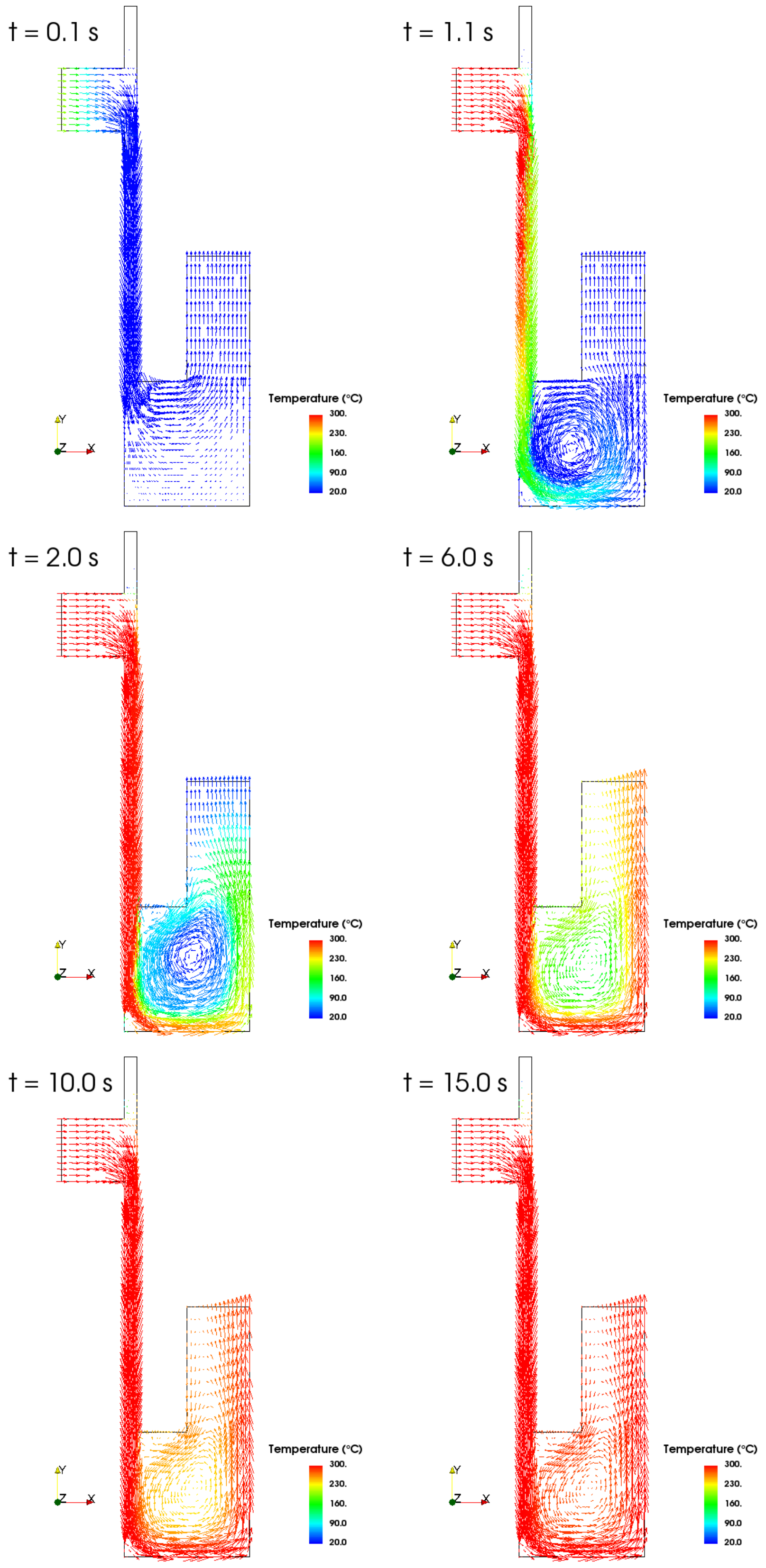


Figure II.6: Water velocity field colored by temperature at different time steps - Case 1

3 CASE 2: Time dependent boundary conditions and variable fluid density

In this case some boundary conditions will be time dependent and some physical characteristics of the fluid will be dependent on the temperature.

Remark: You can copy your `case1` in order to make the `case2`:

```
$ code_saturne create --copy-from case1 case2
```

3.1 Calculation options

The options for this case are the same as in `case1`, except for the variable fluid density:

→ Flow type: unsteady flow

→ Time step: uniform and constant

→ Turbulence model: $k - \epsilon$

→ Scalar(s): 1 - temperature
2 - passive scalar

→ Physical properties: uniform and constant (except density)

→ Management of monitoring points

3.2 Initial and boundary conditions

→ Initialization: 20°C for temperature
10 for the passive scalar

The boundary conditions are defined in the user interface and depend on the boundary zone. The time dependence of the temperature boundary condition implies the use of a Fortran user routine (see below).

- **Flow inlet**: Dirichlet condition, an inlet velocity of 1 $m.s^{-1}$, a time dependent inlet temperature and a value of 200 for the passive scalar are imposed;
- **Outlet**: default value;
- **Walls**: velocity, pressure and thermal scalar: default value
passive scalar: different conditions depending on the color and geometric parameters.

The boundary conditions for the passive scalar are identical as those in `case1`, as specified in the following table:

Wall	Nature	Value
wall_1	Imposed value (Dirichlet)	0
wall_2	Imposed value (Dirichlet)	5
wall_3	Imposed value (Dirichlet)	0
wall_4	Imposed value (Dirichlet)	25
wall_5	Imposed value (Dirichlet)	320
wall_6	Imposed value (Dirichlet)	40

The **wall_1** to **wall_6** regions are defined as follows, through color references and geometric localization:

Figure II.3 shows the colors used for boundary conditions and table II.4 defines the correspondance between the colors and the type of boundary condition to use.

Label	Color and geometric parameters
wall_1	24 and $0.1 \leqslant x$ and $x \leqslant 0.5$
wall_2	2 or 3
wall_3	4 or 7 or 21 or 22 or 23
wall_4	6 and $y > 1$
wall_5	6 and $y \leqslant 1$
wall_6	31 or 33

Colors	Conditions
1	Inlet
34	Outlet
2 3 4 6 7 21 22 23 31 33	Wall
24 for $0.1 \leq x \leq 0.5$	Wall
8 9 28 29 38 39	Symmetry

Table II.4: Boundary faces colors and associated references

3.3 Variable Density

In this case the density is a function of the temperature. The variation law is defined in the Graphical User Interface, although it can also be defined in a Fortran user routine. The expression is:

$$\rho = T(AT + B) + C \tag{II.1}$$

where ρ is the density, T is the temperature, $A = -4.0668\times10^{-3}$, $B = -5.0754\times10^{-2}$ and $C = 1\,000.9$.

In order for the variable density to have an effect on the flow, gravity must be set to a non-zero value. $\underline{g} = -9.81\underline{e}_y$ will be specified in the Graphical Interface.

Remark:
The temperature is **temperature** in the user expression. Don't forget **;** at the end of the expression.

3.4 Parameters

The calculation parameters are identical as those in `case1`.

All the parameters necessary to this study can be defined through the Graphical Interface, except the time dependent boundary conditions that have to be specified in user routines.

In order to join the separate meshes into a single domain, colors 5, 24 and 32 will have to be joined through the Graphical Interface.

3.5 User routine

The routine `cs_user_boundary_conditions.f90` has to be copied from the folder `SRC/REFERENCE` into the folder `SRC` [4].

[4]Only when it appears in the `SRC` directory will it be taken into account by the code.

Parameters of calculation control	
Number of iterations	300
Reference time step	0.05
Output period for post-processing files	2

- **cs_user_boundary_conditions.f90**
  This routine allows to define advanced boundary conditions on the boundary faces.

  Even if `cs_user_boundary_conditions.f90` is used, all boundary conditions have to be defined in the Graphical User Interface (GUI). Only the conditions that differ from this first definition need to appear in `cs_user_boundary_conditions.f90`. The boundary conditions defined in `cs_user_boundary_conditions.f90` will replace those specified in the Graphical Interface.

  In this case, the temperature at entry is supposed variable in time, following the law:

$$\begin{cases} T = 20 + 100t & \text{for } 0 \leqslant t \leqslant 3.8 \\ T = 400 & \text{for } t > 3.8 \end{cases} \tag{II.2}$$

  where T is the temperature in °C and t is the time in seconds (s).

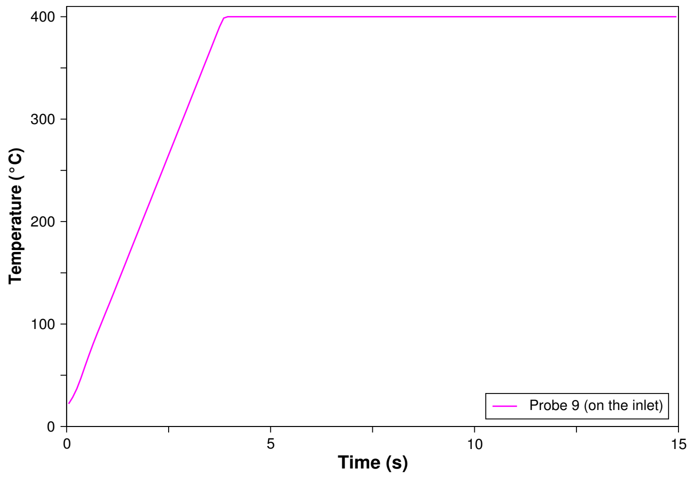


Figure II.7: Time evolution of the temperature at inlet.

Remark:
`ttcabs` is the current physical time. See the example file in the subdirectory 🗀`SRC/EXAMPLES` for the complete `cs_user_boundary_conditions.f90` file.

3.6 Output management

The output management is the same as in `case1`, except that a nineth monitoring point will be added, just at the entry, to monitor the temperature evolution at inlet.

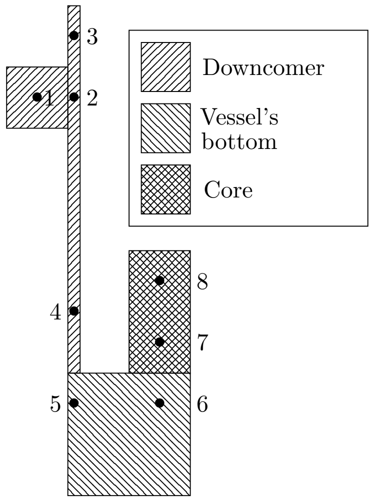
In this case, the **Pressure**, the **Tubulent Energy** and the **Dissipation** will be removed from the listing file.

The **Courant number** (CFL) and **Fourier number** will be removed from the post-processing results[5].

Eventually, probes will be defined for chronological records, following the data given in figure II.8. Then the **total_pressure** will be deactivated from all probes.

In addition the domain boundary will be post-processed. This allows to check the boundary conditions, and especially that of the temperature and passive scalar.

[5]This can be very useful to save some disk space if some variables are of no interest, as post-processing files can be large.



Probe n^o.	x (m)	y (m)	z (m)
1	-0.25	2.25	0
2	0.05	2.25	0
3	0.05	2.75	0
4	0.05	0.50	0
5	0.05	-0.25	0
6	0.75	-0.25	0
7	0.75	0.25	0
8	0.75	0.75	0
9	-0.5	2.25	0

Figure II.8: Position and coordinates of probes in the full domain

3.7 Calculation restart

After the first run, the calculation will be continued for another 400 time steps. The calculation restart is managed through the Graphical Interface.

3.8 Results

Figure II.9 shows the time evolution of temperature recorded on each monitoring probe. Note that the .csv files obtained for each case, were concatenated to plot the evolution of temperature over the entire period.

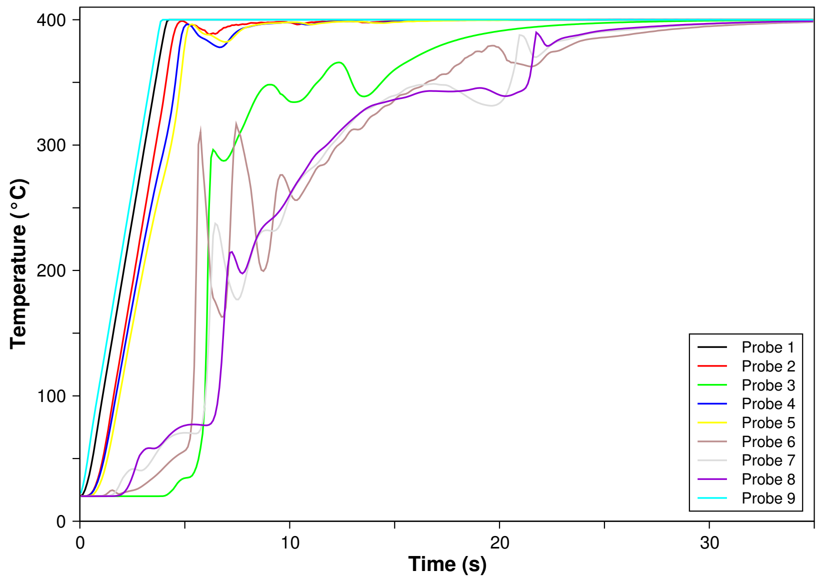


Figure II.9: Time evolution of temperature at monitoring probes - Case 2

Figure II.10 shows the velocity fields colored by temperature in the first run of calculation.

Figure II.11 shows the velocity fields in the second calculation (restart of the first one).

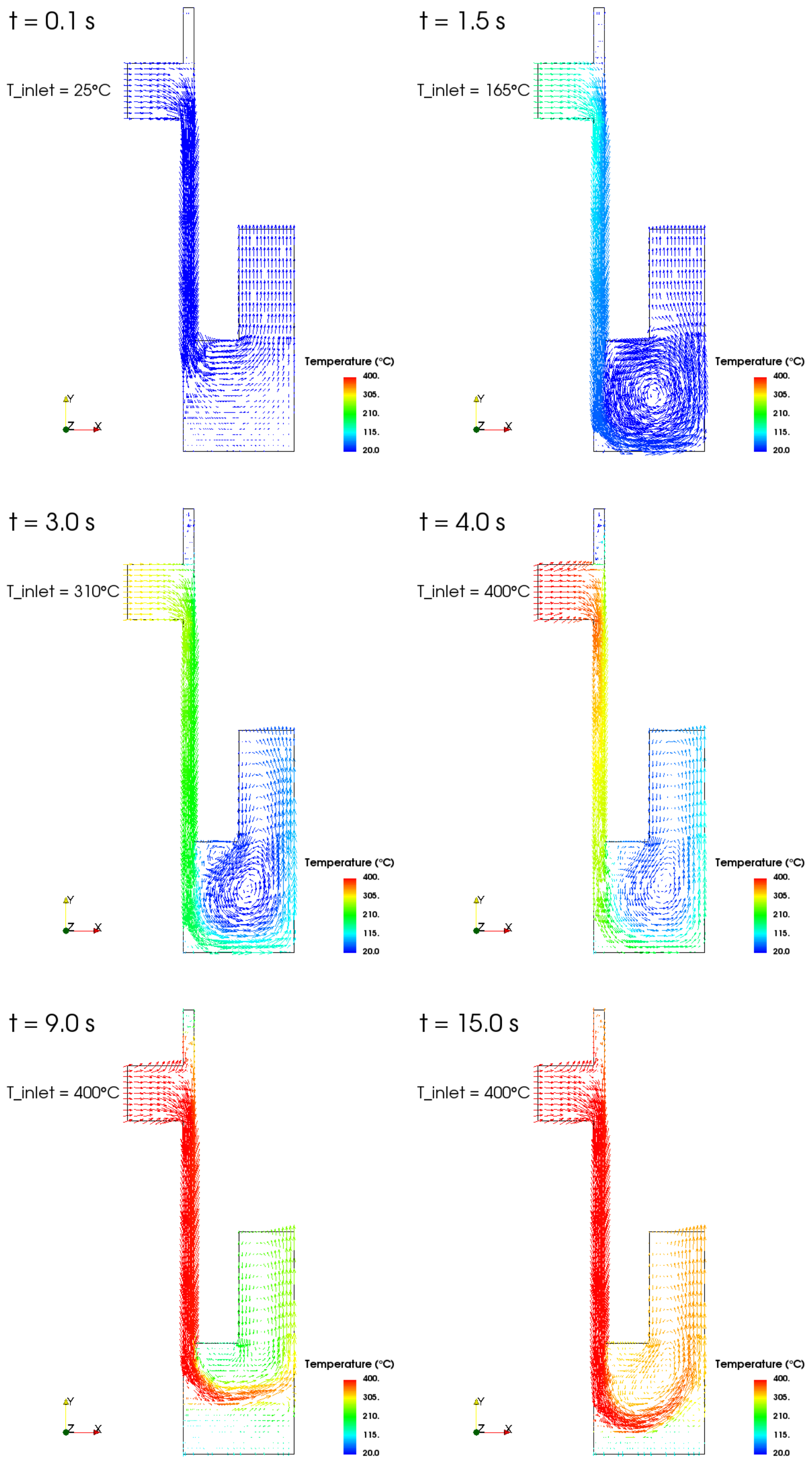


Figure II.10: Water velocity field colored by temperature and inlet temperature value at different time steps (first calculation) - Case 2

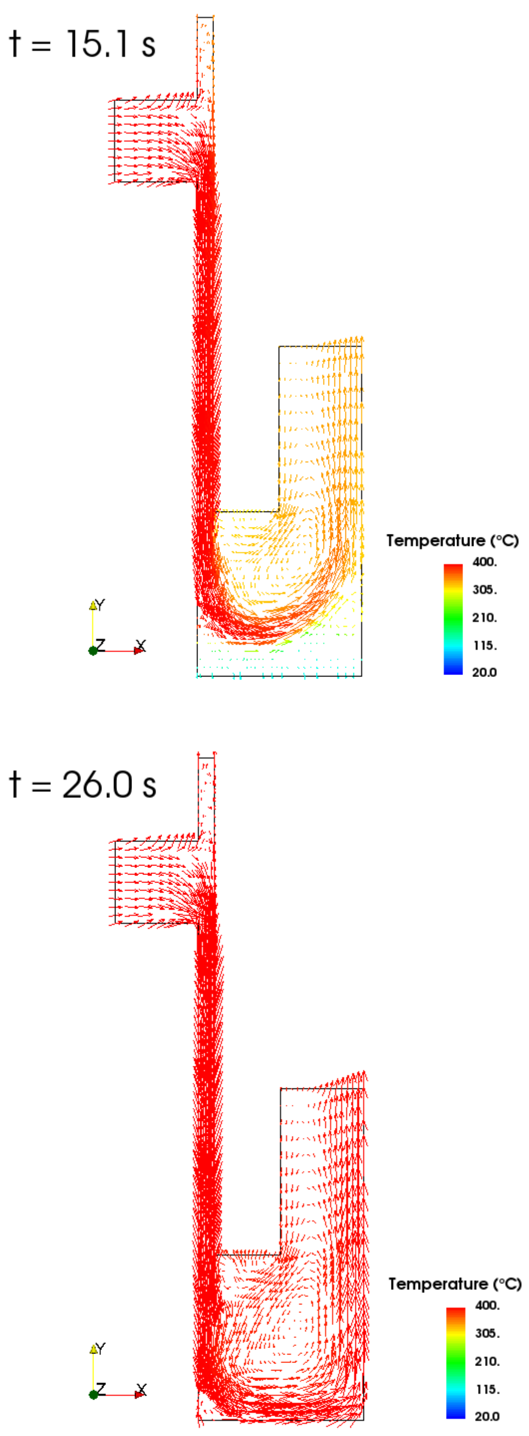
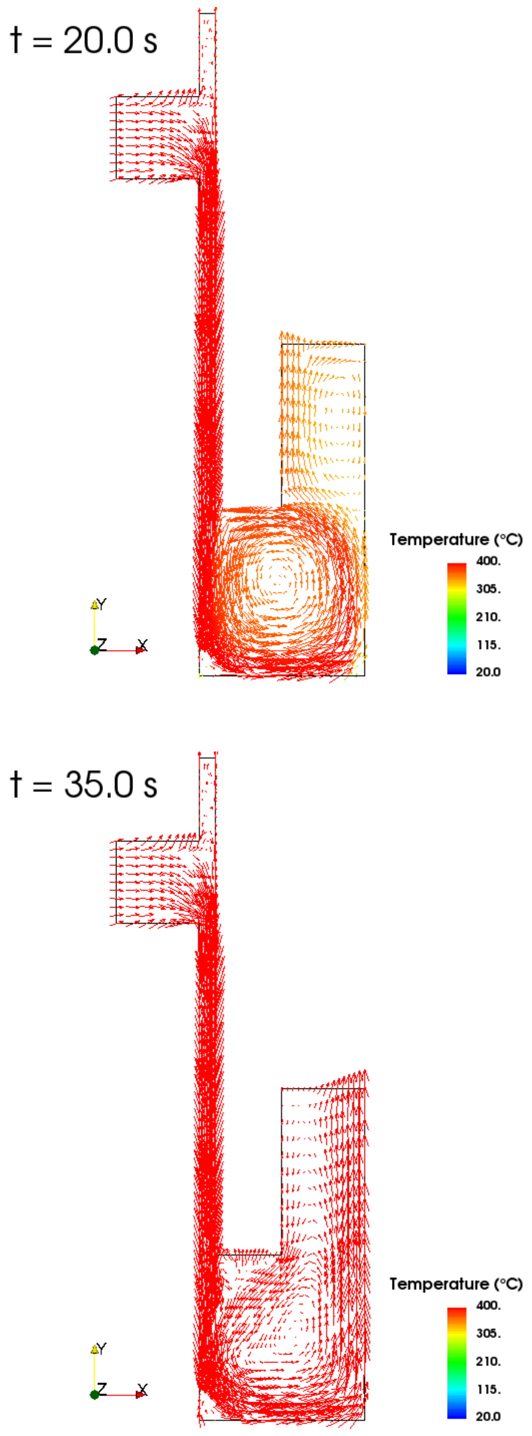


Figure II.11: Water velocity field colored by temperature at different time steps (second calculation) - Case 2



4 CASE 3: Head losses, parallelism and spatial average

This case will be run in parallel on two processors. Head losses will be used to simulate the presence of an obstacle in the flow and the spatial average of the temperature will be calculated at each time step.

4.1 Calculation options

The options for this case are the same as in `case2`:

→ Flow type: unsteady flow

→ Time step: uniform and constant

→ Turbulence model: $k - \epsilon$

→ Scalar(s): 1 - temperature
  2 - passive scalar, with diffusion coefficient 8.55 ($\times 10^{-5}$ $m^2.s^{-1}$)

→ Physical properties: uniform and constant (except density)

→ Management of monitoring points

4.2 Initial and boundary conditions

→ Initialization: 20°C for temperature
  10 for the passive scalar

The boundary conditions are defined in the user interface and depend on the boundary zone.

- **Flow inlet**: Dirichlet condition, an inlet velocity of 1 $m.s^{-1}$ and a time dependent inlet temperature and a value of 200 for the passive scalar are imposed
- **Outlet**: default value
- **Walls**: velocity, pressure and thermal scalar: default value
  passive scalar: different conditions depending on the color and geometric parameters

The boundary conditions for the passive scalar are identical as those in `case2`, as specified in the following table:

Wall	Nature	Value
wall_1	Imposed value (Dirichlet)	0
wall_2	Imposed value (Dirichlet)	5
wall_3	Imposed value (Dirichlet)	0
wall_4	Imposed value (Dirichlet)	25
wall_5	Imposed value (Dirichlet)	320
wall_6	Imposed value (Dirichlet)	40

The **wall_1** to **wall_6** regions are defined as follows, through color references and geometric localization:

Label	Color and geometric parameters
wall_1	24 and $0.1 \leqslant x$ and $x \leqslant 0.5$
wall_2	2 or 3
wall_3	4 or 7 or 21 or 22 or 23
wall_4	6 and $y > 1$
wall_5	6 and $y \leqslant 1$
wall_6	31 or 33

Figure II.3 shows the colors used for boundary conditions and table II.5 defines the correspondance between the colors and the type of boundary condition to use.

Colors	**Conditions**
1	Inlet
34	Outlet
2 3 4 6 7 21 22 23 31 33	Wall
24 for $0.1 \leq x \leq 0.5$	Wall
8 9 28 29 38 39	Symmetry

Table II.5: Boundary faces colors and associated references

4.3 Variable Density

The law for the variable density is identical as that in `case2`.

In this case the density is a function of temperature, the variation law is defined in the Graphical User Interface although it can also be defined in a Fortran user routine. The expression is:

$$\rho = T(AT + B) + C \tag{II.3}$$

where ρ is the density, T is the temperature, $A = -4.0668 \times 10^{-3}$, $B = -5.0754 \times 10^{-2}$ and $C = 1\,000.9$.

In order for the variable density to have an effect on the flow, gravity must be set to a non-zero value. $\underline{g} = -9.81\underline{e}_y$ will be specified in the Graphical Interface.

4.4 Head losses

To simulate the presence of an obstacle 0.20 (m) large and 0.5 (m) high in the vessel, a zone of head losses will be created in the domain (fig II.12).

The head losses zone is located between the coordinates $x = 0.2$ (m) and $x = 0.4$ (m), and $y = -0.75$ (m) and $y = -0.25$ (m).

The head losses coefficient to apply is $K_{11} = K_{22} = K_{33} = 10^4 = \frac{1}{2}\ \alpha_{11} = \frac{1}{2}\ \alpha_{22} = \frac{1}{2}\ \alpha_{33}$ and is isotropic.

4.5 Parameters

All the parameters necessary to this study can be defined through the Graphical Interface. However, the calculation of the spatial average is defined by a user routine.

Parameters of calculation control	
Number of iterations	900
Reference time step	0.01
Output period for post-processing files	2
The calculation will be run in parallel	2 procs.

In order to join the separate meshes into a single domain, colors 5, 24 and 32 will have to be joined through the Graphical Interface.

Note that the time step has been reduced because of the head losses: the pressure step is more difficult to be solved in presence of head losses.

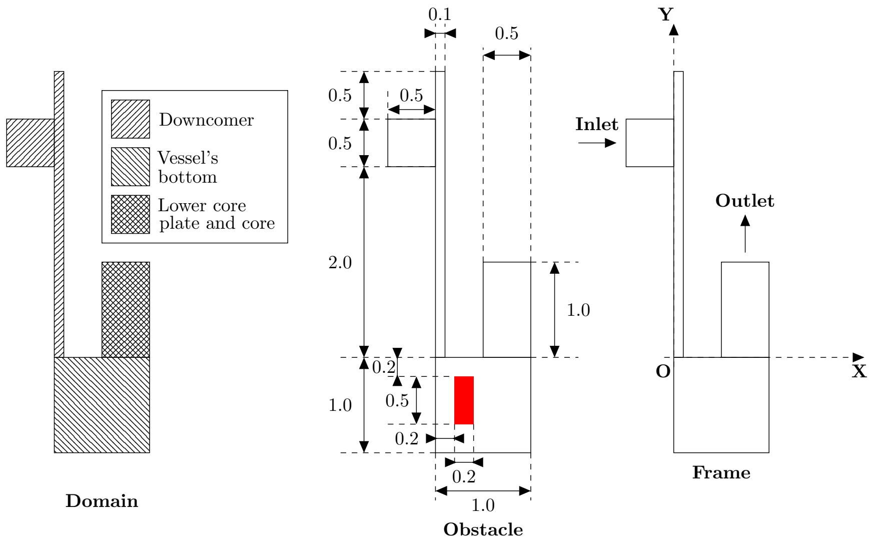


Figure II.12: Full domain geometry with the obstacle

4.6 User routines

The following routines have to be copied from the folder `SRC/REFERENCE/` into the folder `SRC/`[6]: `cs_user_boundary_conditions.f90` and `cs_user_extra_operations.f90`. We can find and copy some basic and specific boundary conditions examples in the folder `SRC/EXAMPLES/` to correctly impose the *Code_Saturne* boundary conditions.

- **cs_user_boundary_conditions.f90**
  This routine allows to define advanced boundary conditions on the boundary faces.

  Even if `cs_user_boundary_conditions.f90` is used, all boundary conditions have to be defined in the Graphical User Interface (GUI). Only the conditions that differ from this first definition need to appear in `cs_user_boundary_conditions.f90`. The boundary conditions defined in `cs_user_boundary_conditions.f90` will replace those specified in the Graphical Interface.

  In this case, the temperature at entry is supposed variable in time, following the law:

$$\begin{cases} T = 20 + 100t & \text{for } 0 \leqslant t \leqslant 3.8 \\ T = 400 & \text{for } t > 3.8 \end{cases} \tag{II.4}$$

  where T is the temperature in °C and t is the time in seconds (s).

- **cs_user_extra_operations.c**
  This routine is called at the end of each time step and has access to the whole set of variables of the code. It is therefore useful for many user-specific post-processing, including the calculation of a spatial average in the present case.

  The spatial average of the temperature will be calculated at each time step and the result wrote in a file named `moy.dat`. The values are saved in order to draw the time evolution of the average temperature.

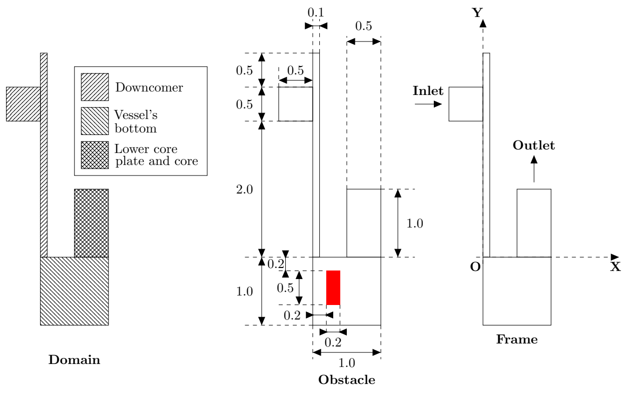
  Beware when calculating the average. Since the calculation is running in parallel, computing the sum of the temperatures on **all the cells** will only yield for each processor to the sum on the cells managed by this processor. In order to obtain the full sum, the parallelism routine `cs_parall_sum` must be used (see example in the `cs_user_extra_operations-scalar_balance.c` routine).

[6]Only when they appear in the SRC directory will they be taken into account by the code.

Remark: `cs_user_extra_operations-xxx.c` are different example routines present in the subdirectory `SRC/EXAMPLES`. They should be removed from the `SRC/` before running the case.

4.7 Output management

The output management is the same as in `case2`.



Probe n^o.	x (m)	y (m)	z (m)
1	-0.25	2.25	0
2	0.05	2.25	0
3	0.05	2.75	0
4	0.05	0.5	0
5	0.05	-0.25	0
6	0.75	-0.25	0
7	0.75	0.25	0
8	0.75	0.75	0
9	-0.5	2.25	0

Figure II.13: Position and coordinates of probes in the full domain

In this case, the **Pressure**, the **Tubulent Energy** and the **Dissipation** will be removed from the listing file.

The **Courant number** (CFL) and **Fourier number** will be removed from the post-processing results[7].

Eventually, probes will be defined for chronological records, following the data given in Figure II.4. Then the **total pressure** will be deactivated from all probes.

In addition the domain boundary will be post-processed. This allows to check the boundary conditions, and especially that of the temperature and passive scalar.

4.8 Results

Figure II.14 shows the evolution of the spatial average of the temperature.

Figure II.15 shows velocity fields colored by temperature. The effect of the head loss modeling the obstacle is clearly visible.

[7]This can be very useful to save some disk space if some variables are of no interest, as post-processing files can be large.

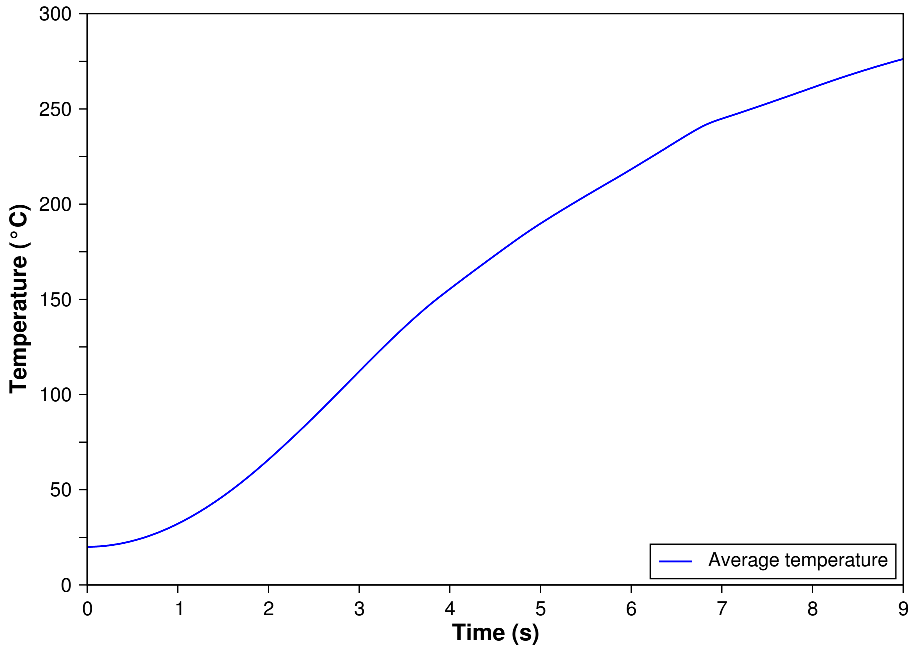


Figure II.14: Evolution of the spatial average of the temperature as a function of time - Case 3

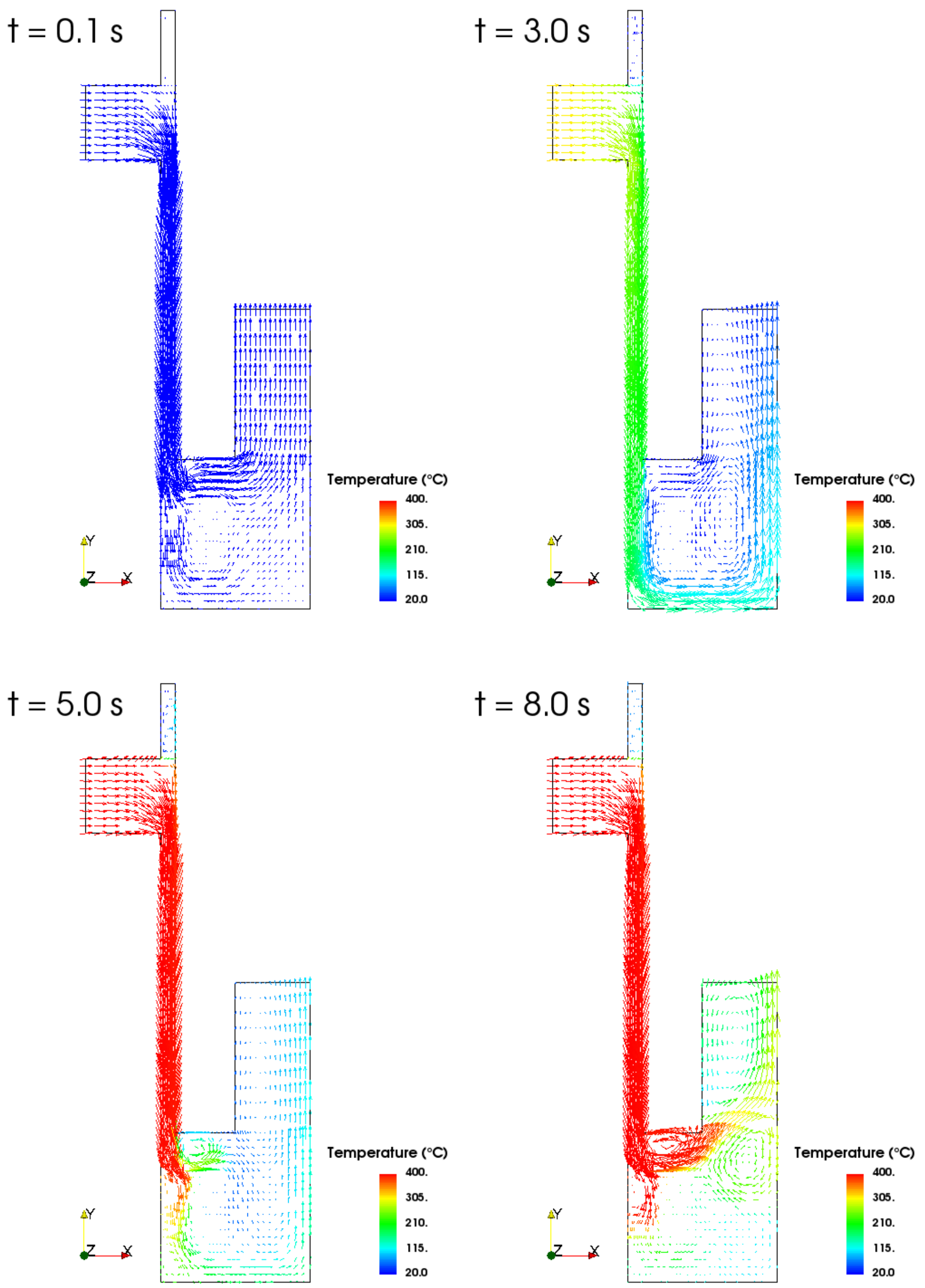


Figure II.15: Water velocity field colored by temperature - Case 3

Part III

Step by step solution

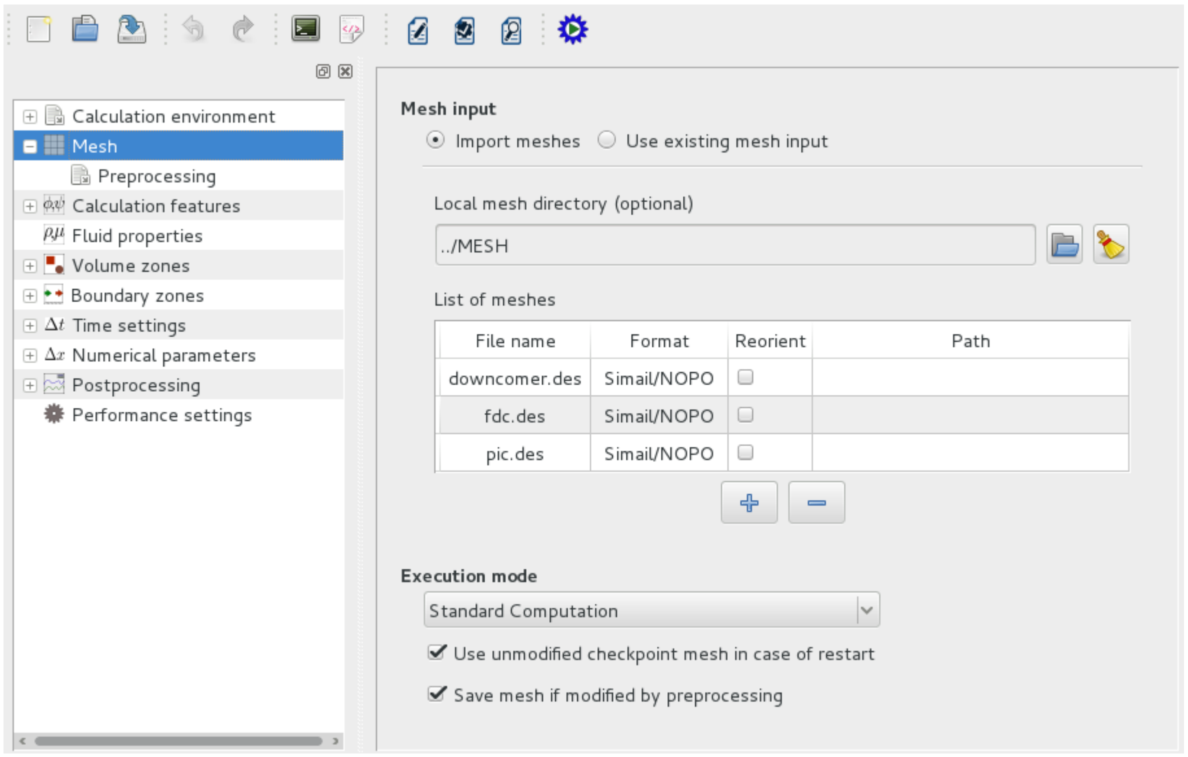
1 Solution for CASE1

This case corresponds to a new study, in which there will be three calculation cases (cases 1, 2 and 3). We can create one case in a single `code_saturne create` command and additional cases can be added later. To test this functionality, first create the study directory, with case subdirectory `case1`, as below:

```
$ code_saturne create -s full_domain -c case1
$ cd full_domain
```

Go to the `DATA` directory in `case1`, open a new case and select the meshes to use.

Click on the heading **Mesh** then add three meshes which have to be joined as shown below.



Mesh input

(● Import meshes) ○ Use existing mesh input

Local mesh directory (optional): ../MESH

List of meshes

File name	Format	Reorient	Path
downcomer.des	Simail/NOPO	☐	
fdc.des	Simail/NOPO	☐	
pic.des	Simail/NOPO	☐	

Execution mode: Standard Computation

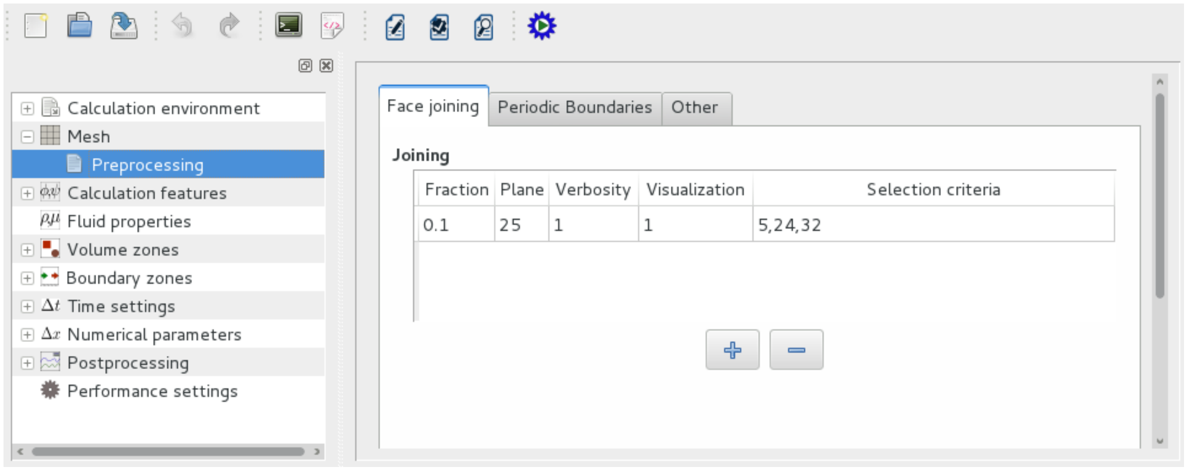
☑ Use unmodified checkpoint mesh in case of restart

☑ Save mesh if modified by preprocessing

Figure III.1: Meshes: list of meshes

In order to join the three meshes, you must add a selection criteria in the box **Selection criteria** of the **Face joining (optional)** item under the **Preprocessing** sub-folder. In this case, only faces of colors 5, 24 and 32 can be joined (different colors can be entered on a single line, separated by comma).

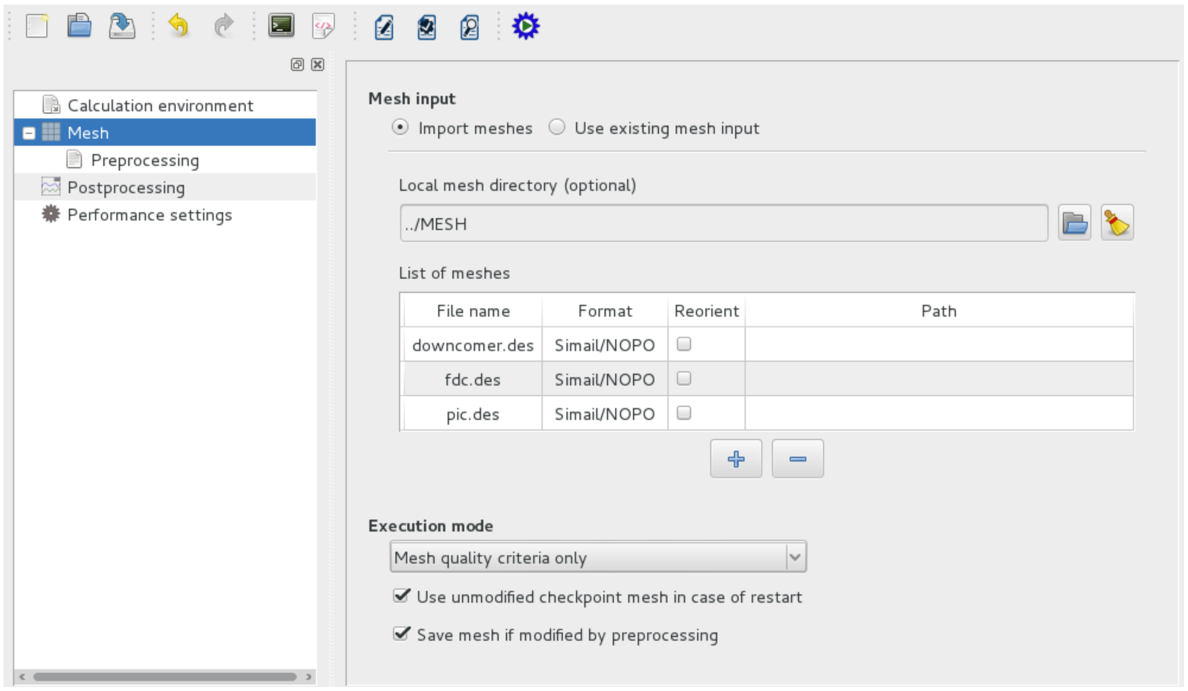
Click on the $\boxed{+}$ icon to enter the list of colors to be joined in the **Face joining (optional)** item.



Fraction	Plane	Verbosity	Visualization	Selection criteria
0.1	25	1	1	5,24,32

Figure III.2: Join a mesh

You can now verify the quality of your mesh by running a **Mesh quality criteria only** Computation. To do so, go back to the heading **Mesh** and change the **Execution mode** to **Mesh quality criteria only**. calculation.



Mesh input

(● Import meshes) ○ Use existing mesh input

Local mesh directory (optional)

../MESH

List of meshes

File name	Format	Reorient	Path
downcomer.des	Simail/NOPO	☐	
fdc.des	Simail/NOPO	☐	
pic.des	Simail/NOPO	☐	

Execution mode

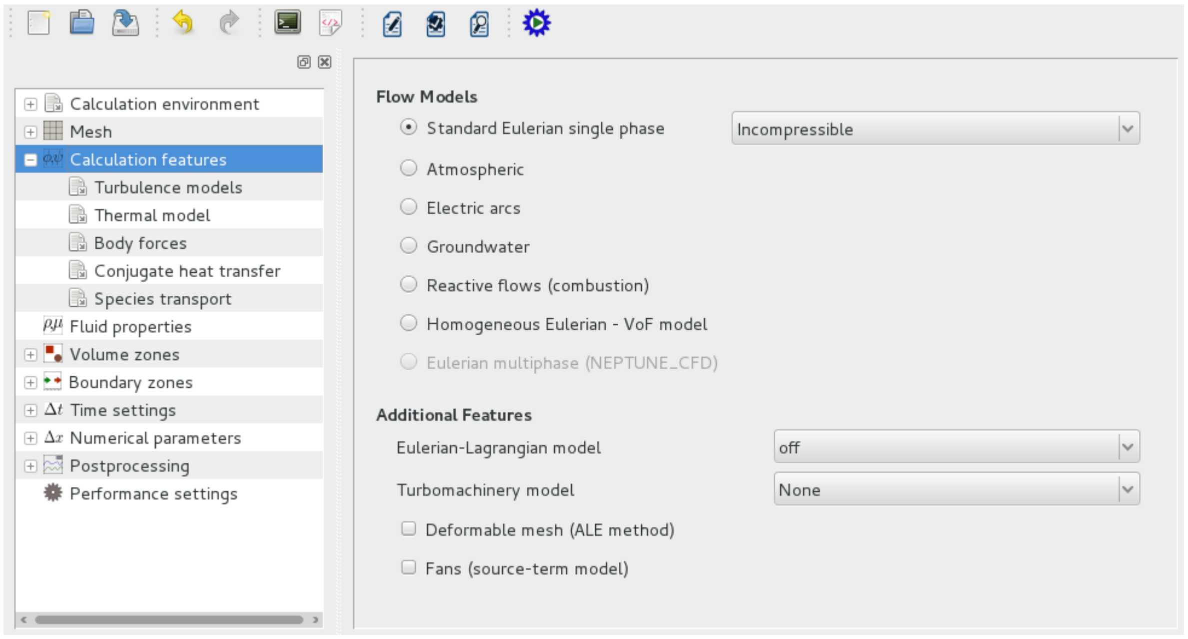
Mesh quality criteria only

✓ Use unmodified checkpoint mesh in case of restart

✓ Save mesh if modified by preprocessing

Figure III.3: Mesh quality criteria

Now, go back to the **Standard Computation** mode and click on the **Calculation features** heading. The heading **Calculation features** is identical to `simple_junction`.



Flow Models

- (●) Standard Eulerian single phase — Incompressible
- ( ) Atmospheric
- ( ) Electric arcs
- ( ) Groundwater
- ( ) Reactive flows (combustion)
- ( ) Homogeneous Eulerian - VoF model
- ( ) Eulerian multiphase (NEPTUNE_CFD)

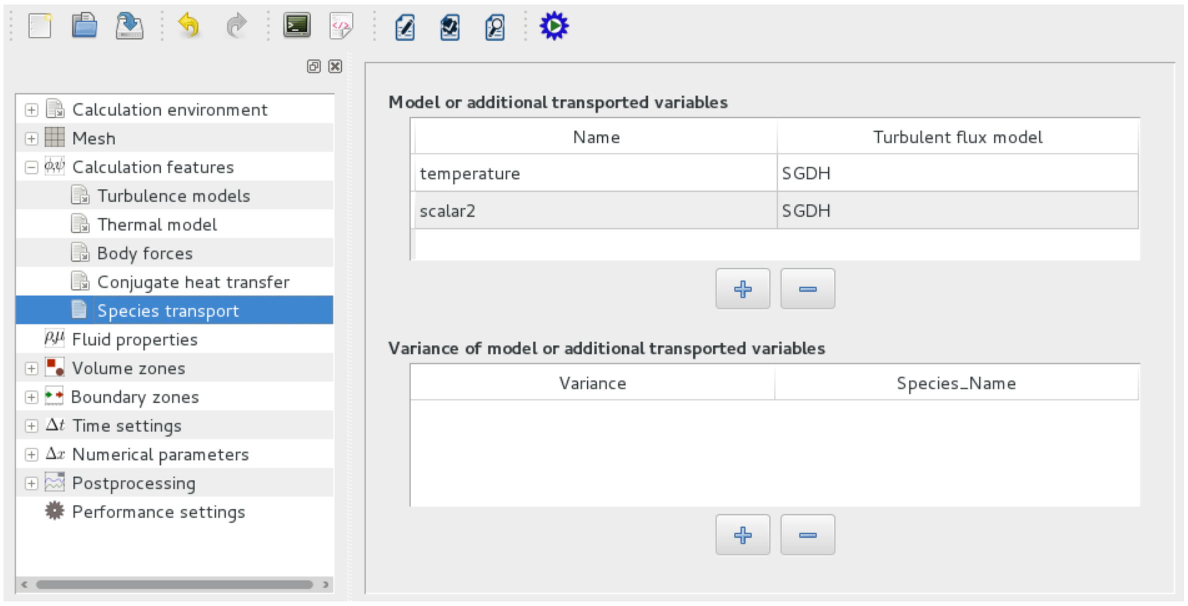
Additional Features

Eulerian-Lagrangian model	off
Turbomachinery model	None

- ☐ Deformable mesh (ALE method)
- ☐ Fans (source-term model)

Figure III.4: Thermophysical models/Calculation features: unsteady flow

To add an additional scalar, click on the **Species transport** item under the **Calculation features** heading.



Model or additional transported variables

Name	Turbulent flux model
temperature	SGDH
scalar2	SGDH

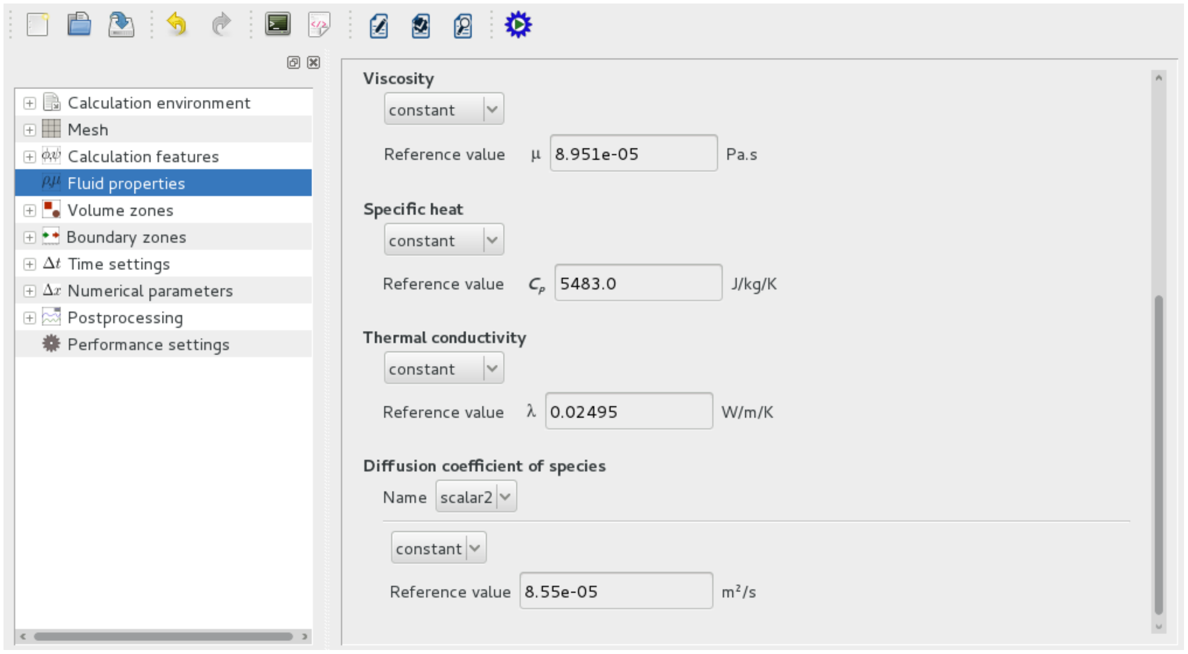
Variance of model or additional transported variables

Variance	Species_Name

Figure III.5: Additional scalar

The heading **Fluid properties** is identical to `simple_junction`, except for the new scalar. Here, wa can specify the diffusion coefficient of this new scalar.

Click on the scalar name to highlight it, then enter the value in the box. In this case, the species diffusion coefficient value is **8.55** ($\times 10^{-5}$ $m^2.s^{-1}$) for the **scalar2** scalar to solve.



Viscosity
constant
Reference value µ 8.951e-05 Pa.s

Specific heat
constant
Reference value C_p 5483.0 J/kg/K

Thermal conductivity
constant
Reference value λ 0.02495 W/m/K

Diffusion coefficient of species
Name scalar2
constant
Reference value 8.55e-05 m²/s

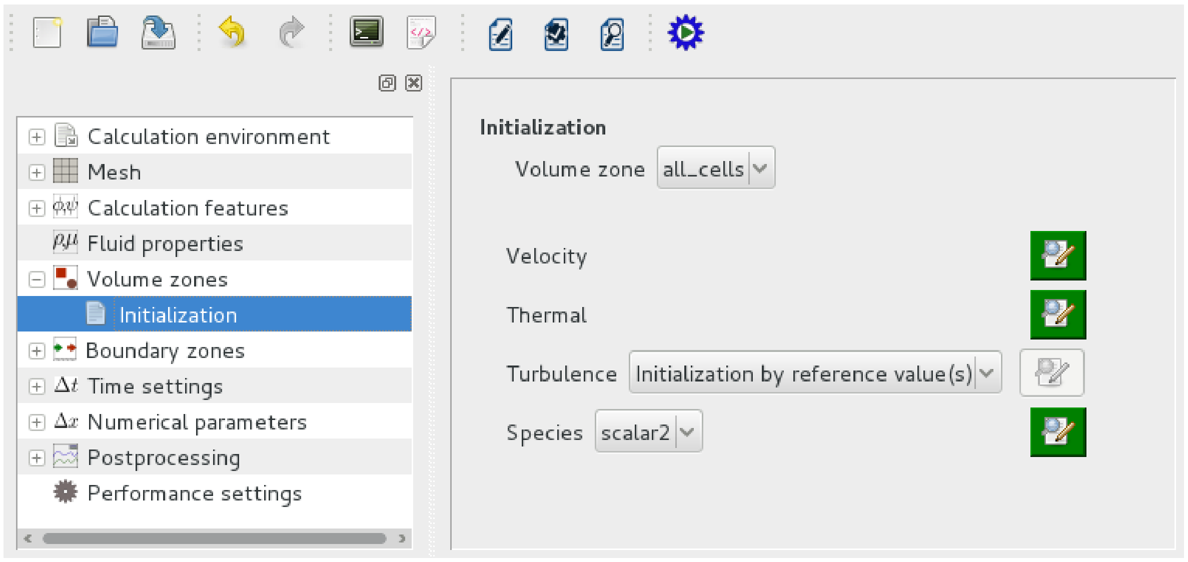
Figure III.6: Fluid properties

Initialization:

To initialize variables at the instant $t = 0$ s, go to the **Initialization** item under the heading **Volume zones**.

Here the velocity, the thermal scalar and the turbulence can be initialized. In this case, the default values to be set are: zero velocity (default), an initial temperature of **20**°C and a turbulence level based on a reference velocity of **1** ($m.s^{-1}$) (default). You must also initialize the `scalar2` species at **10**°C.

Specific zones can be defined with different initializations. In this case, only the default **all_cells** is used.



Initialization

Volume zone all_cells

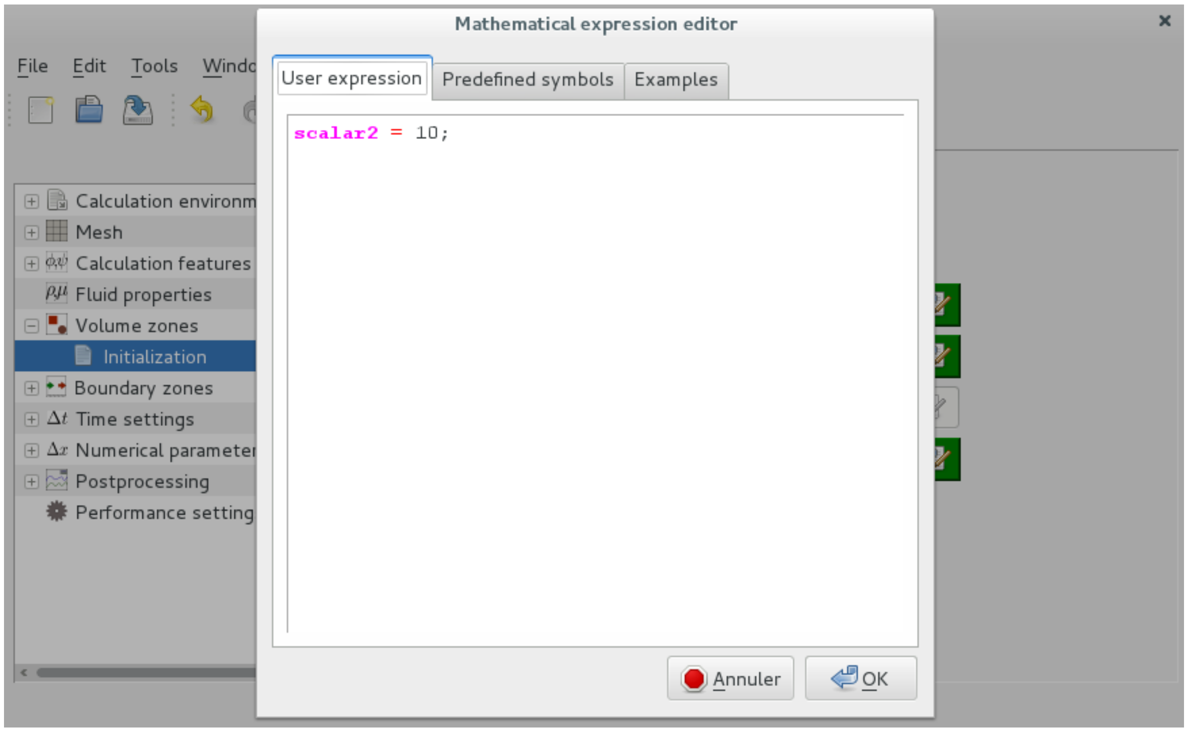
Velocity

Thermal

Turbulence Initialization by reference value(s)

Species scalar2

Figure III.7: Initialization



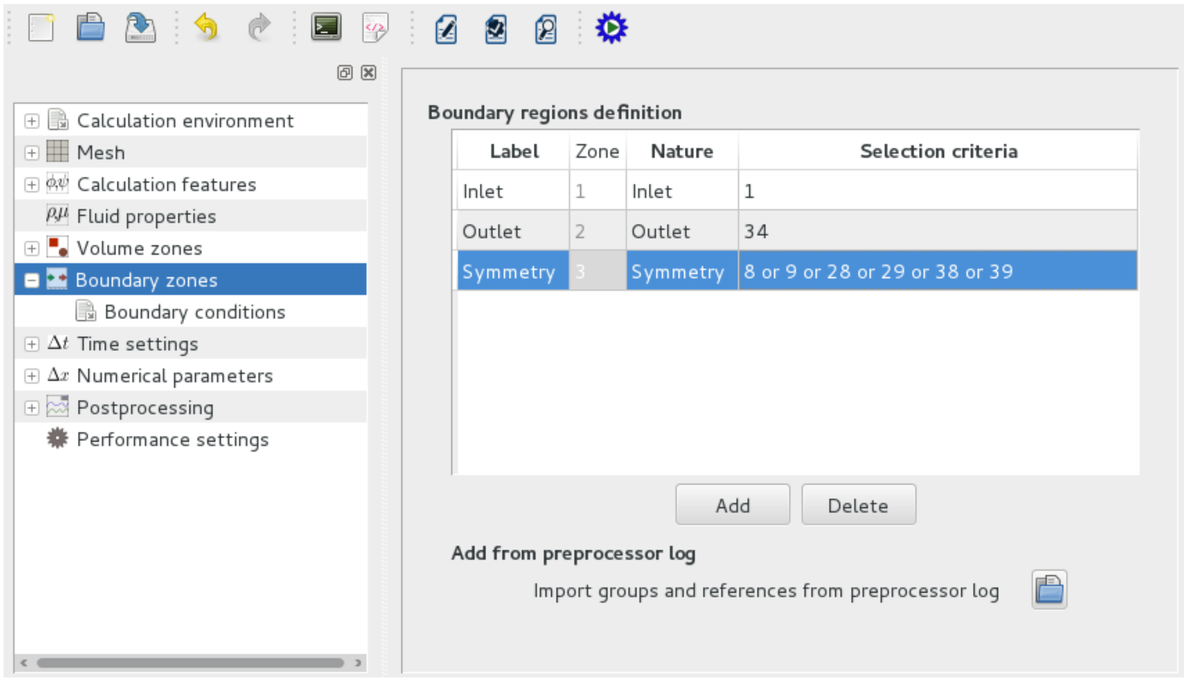
```
scalar2 = 10;
```

Figure III.8: Initialization: species

Create the boundary zones: The procedure is the same as in `simple_junction`, but the colors are different. Note that colors 5 and 32 have completely disappeared in the joining process (they are now internal faces and are not considered as boundaries), while some boundary faces of color 24 remain.

Create the inlet, outlet and symmetry boundary zones with the following colors:

```
- Inlet    :  ''1''
- Outlet   :  ''34''
- Symmetry:  ''8 or 9 or 28 or 29 or 38 or 39''
```



Boundary regions definition

Label	Zone	Nature	Selection criteria
Inlet	1	Inlet	1
Outlet	2	Outlet	34
Symmetry	3	Symmetry	8 or 9 or 28 or 29 or 38 or 39

Add from preprocessor log

Import groups and references from preprocessor log

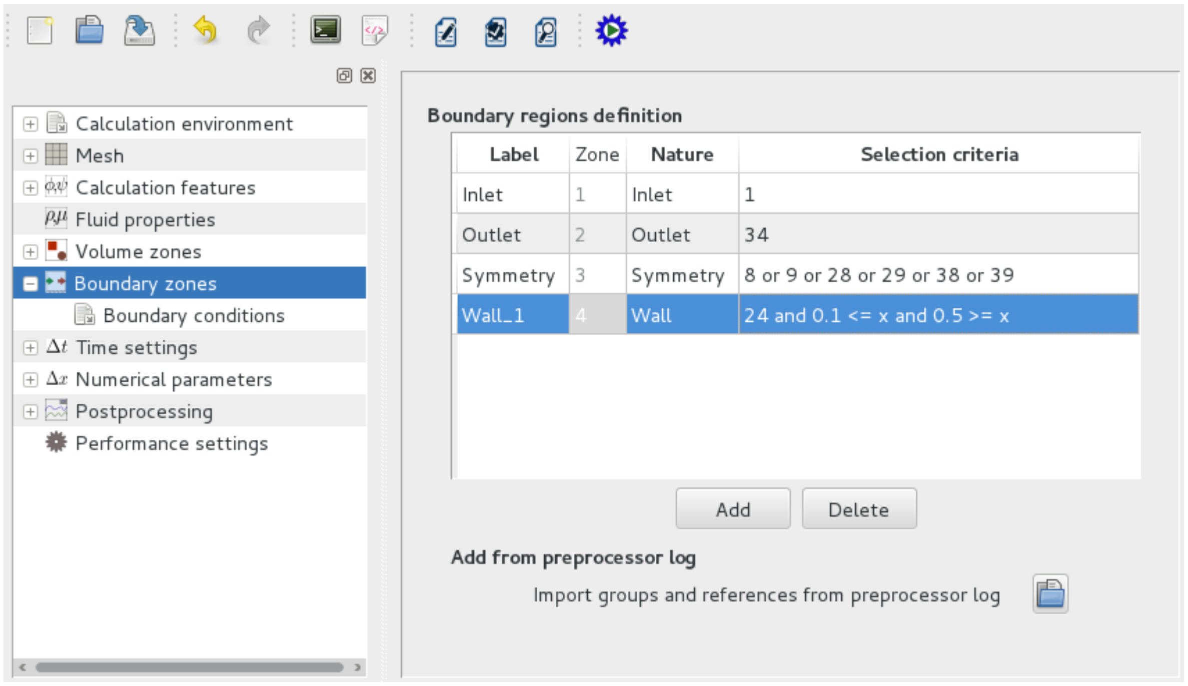
Figure III.9: Creation of the boundary zones

In this case, different conditions are applied for the walls. Separate corresponding wall boundary regions must therefore be created, following the data in the following table.

Label	Zone	Nature	Selection criteria
Wall_1	4	Wall	24 and $0.1 \leqslant x$ and $x \leqslant 0.5$
Wall_2	5	Wall	2 or 3
Wall_3	6	Wall	4 or 7 or 21 or 22 or 23
Wall_4	7	Wall	6 and y>1
Wall_5	8	Wall	6 and y⩽1
Wall_6	9	Wall	31 or 33

The **Wall_1** region combines color and geometrical criteria. The associated character string to enter in the **Selection criteria** box [1] is as follows:

```
''24 and 0.1 <= x and 0.5 >= x''
```



Boundary regions definition

Label	Zone	Nature	Selection criteria
Inlet	1	Inlet	1
Outlet	2	Outlet	34
Symmetry	3	Symmetry	8 or 9 or 28 or 29 or 38 or 39
Wall_1	4	Wall	24 and 0.1 <= x and 0.5 >= x

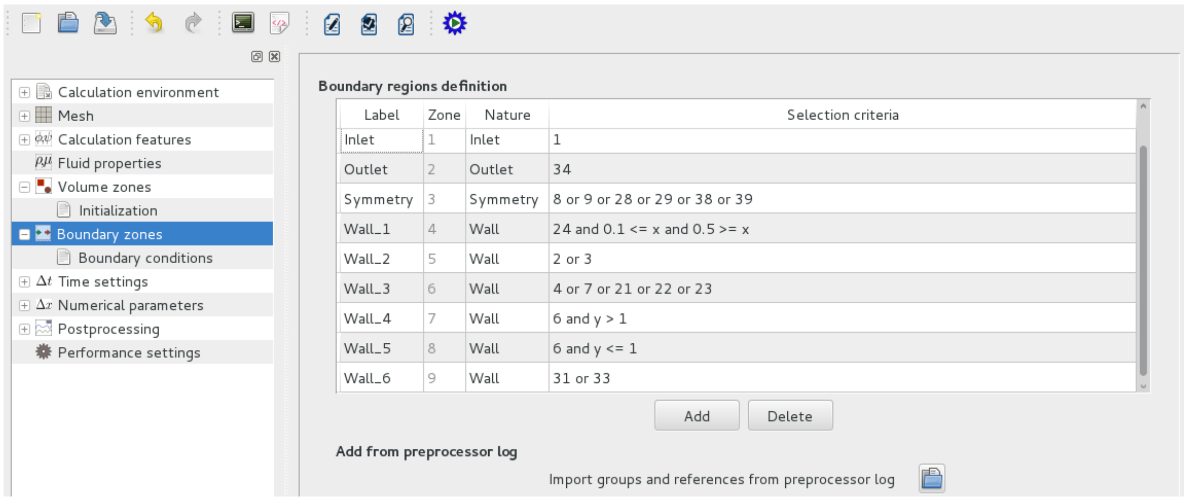
Add from preprocessor log

Import groups and references from preprocessor log

Figure III.10: Creation of a wall boundary region

[1] Note that, due to the joining process, there are in fact no boundary faces of color 24 with x coordinate outside the [0.1;0.5] interval. The geometrical criterium is therefore not necessary. It is presented here to show the capabilities of the face selection module.

Define the other wall boundary zones. The faces of color 6 have to be divided in two separate zones, based on a geometrical criterium on y.



Boundary regions definition

Label	Zone	Nature	Selection criteria
Inlet	1	Inlet	1
Outlet	2	Outlet	34
Symmetry	3	Symmetry	8 or 9 or 28 or 29 or 38 or 39
Wall_1	4	Wall	24 and 0.1 <= x and 0.5 >= x
Wall_2	5	Wall	2 or 3
Wall_3	6	Wall	4 or 7 or 21 or 22 or 23
Wall_4	7	Wall	6 and y > 1
Wall_5	8	Wall	6 and y <= 1
Wall_6	9	Wall	31 or 33

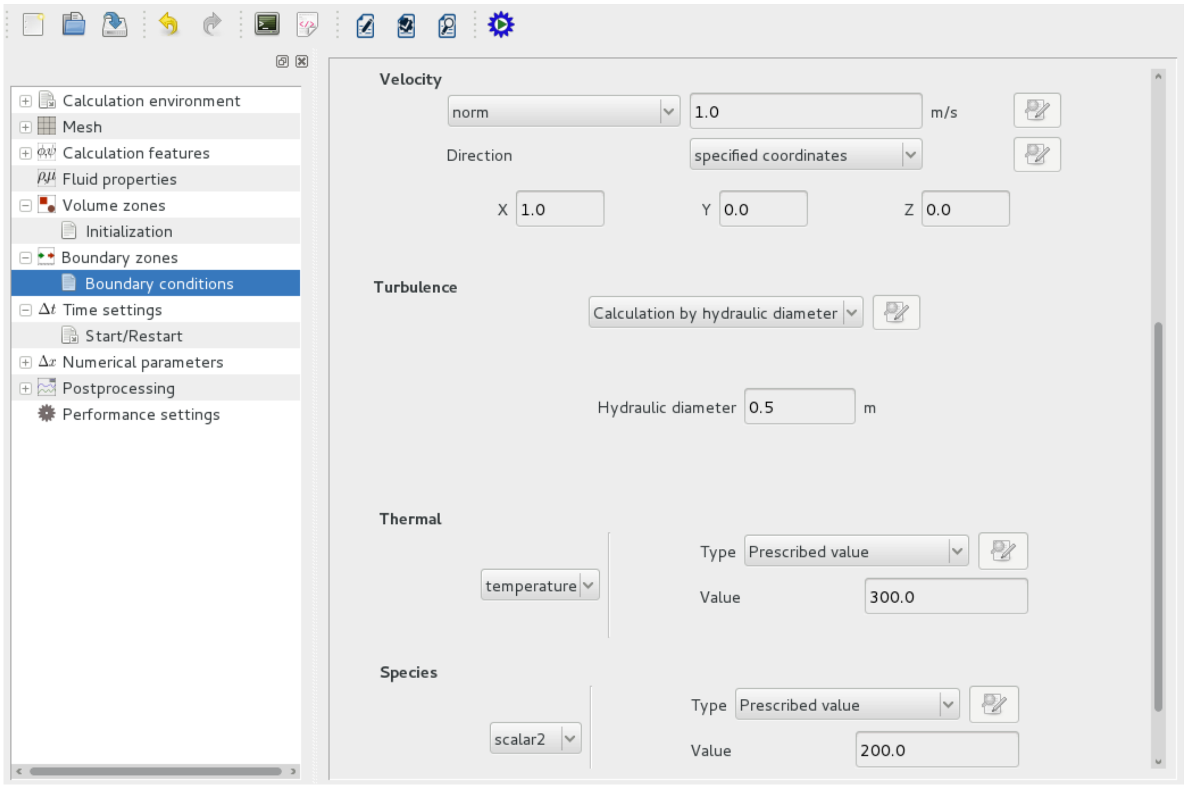
Add from preprocessor log

Import groups and references from preprocessor log

Figure III.11: Creation of wall boundary regions

The dynamic boundary conditions are the same as in `simple_junction` for the inlet, and there are still no sliding walls.

- Inlet:



Velocity
norm 1.0 m/s
Direction specified coordinates
X 1.0 Y 0.0 Z 0.0

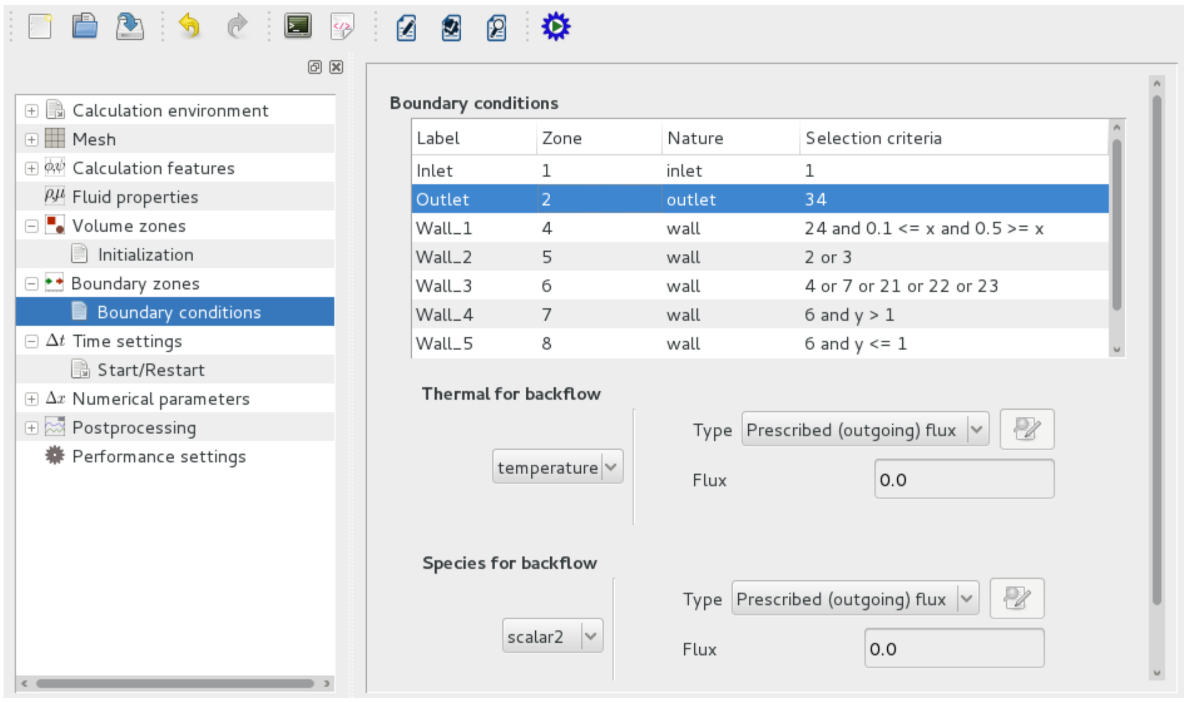
Turbulence
Calculation by hydraulic diameter
Hydraulic diameter 0.5 m

Thermal
temperature
Type Prescribed value
Value 300.0

Species
scalar2
Type Prescribed value
Value 200.0

Figure III.12: Dynamic variables boundary: inlet

– Outlet:



Boundary conditions

Label	Zone	Nature	Selection criteria
Inlet	1	inlet	1
Outlet	2	outlet	34
Wall_1	4	wall	24 and 0.1 <= x and 0.5 >= x
Wall_2	5	wall	2 or 3
Wall_3	6	wall	4 or 7 or 21 or 22 or 23
Wall_4	7	wall	6 and y > 1
Wall_5	8	wall	6 and y <= 1

Thermal for backflow

temperature

Type: Prescribed (outgoing) flux

Flux: 0.0

Species for backflow

scalar2

Type: Prescribed (outgoing) flux

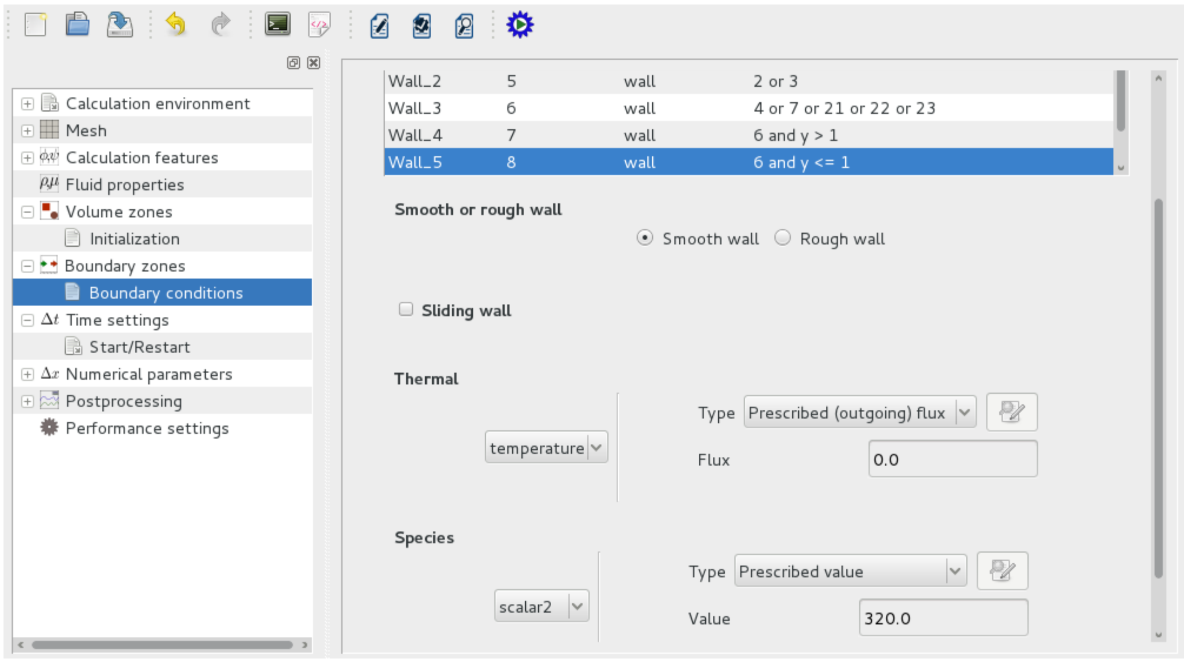
Flux: 0.0

Figure III.13: Dynamic variables boundary: outlet

To configure the scalar boundary conditions on the walls, select individually each wall in the **Boundary conditions** item.

On all the walls, a default homogeneous **Prescribed flux** is set for temperature, and **Prescribed value** is specified for the passive scalar for each wall, named `scalar2`, according to the following table:

Wall	Nature	`Scalar2` value
Wall_1	Prescribed value (Dirichlet)	0
Wall_2	Prescribed value (Dirichlet)	5
Wall_3	Prescribed value (Dirichlet)	0
Wall_4	Prescribed value (Dirichlet)	25
Wall_5	Prescribed value (Dirichlet)	320
Wall_6	Prescribed value (Dirichlet)	40



Wall_2	5	wall	2 or 3
Wall_3	6	wall	4 or 7 or 21 or 22 or 23
Wall_4	7	wall	6 and y > 1
Wall_5	8	wall	6 and y <= 1

Smooth or rough wall
[✓ Smooth wall] Rough wall

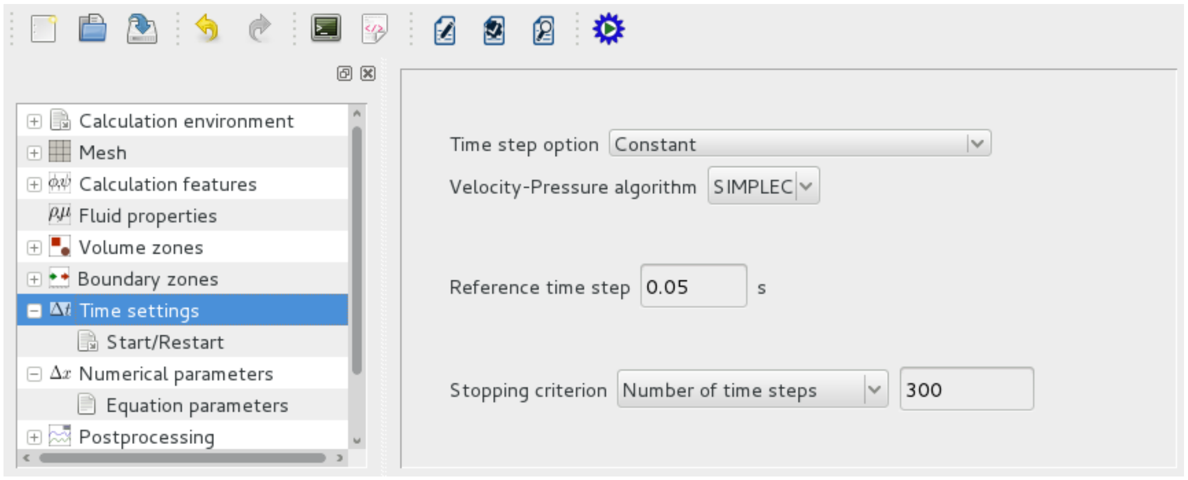
Sliding wall

Thermal
temperature
Type: Prescribed (outgoing) flux
Flux: 0.0

Species
scalar2
Type: Prescribed value
Value: 320.0

Figure III.14: Scalars boundaries: wall_5

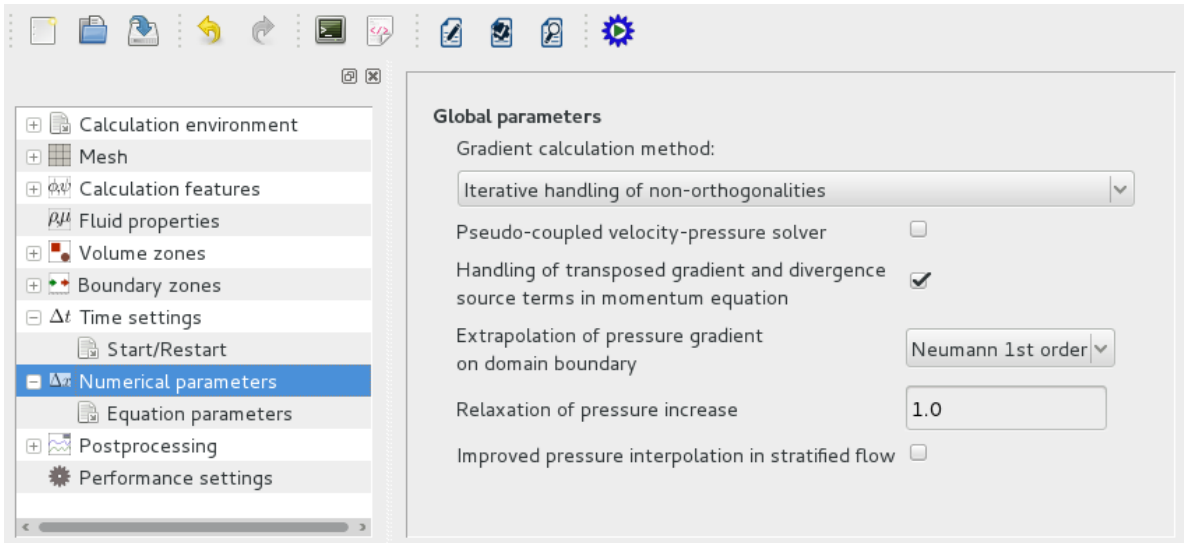
Some calculation parameters need to be defined now. Go to the **Time settings** heading. In our case the time step is **Constant** and the **Velocity-Pressure algorithm** is **SIMPLEC**. Set the number of iterations to **300** and the reference time step to **0.05** (s).



Time step option: Constant
Velocity-Pressure algorithm: SIMPLEC
Reference time step: 0.05 s
Stopping criterion: Number of time steps 300

Figure III.15: Time step setting

Numerical parameters are the same as in `simple_junction`.



Global parameters

Gradient calculation method: Iterative handling of non-orthogonalities

Pseudo-coupled velocity-pressure solver ☐

Handling of transposed gradient and divergence source terms in momentum equation ✓

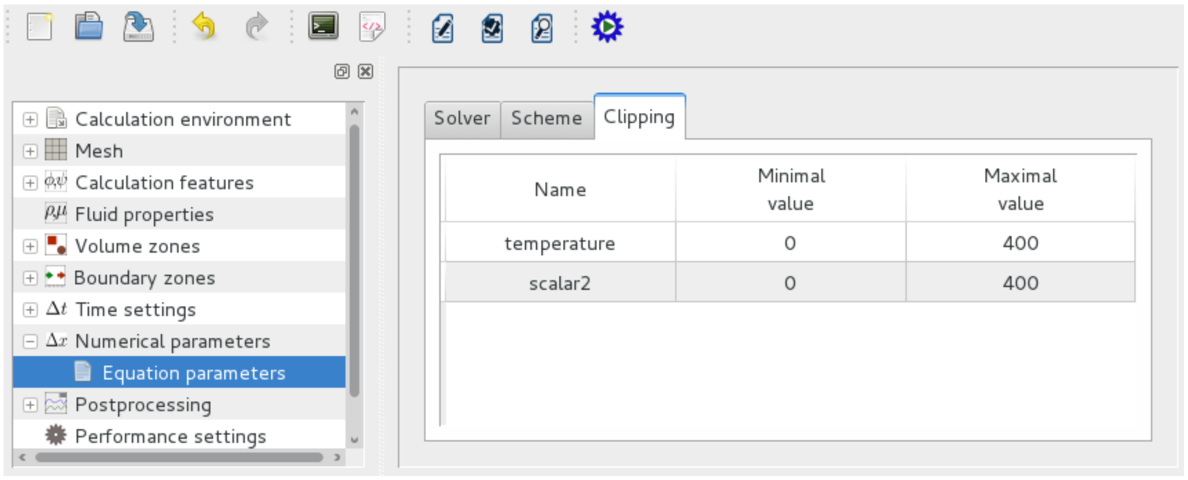
Extrapolation of pressure gradient on domain boundary: Neumann 1st order

Relaxation of pressure increase: 1.0

Improved pressure interpolation in stratified flow ☐

Figure III.16: Numerical parameters

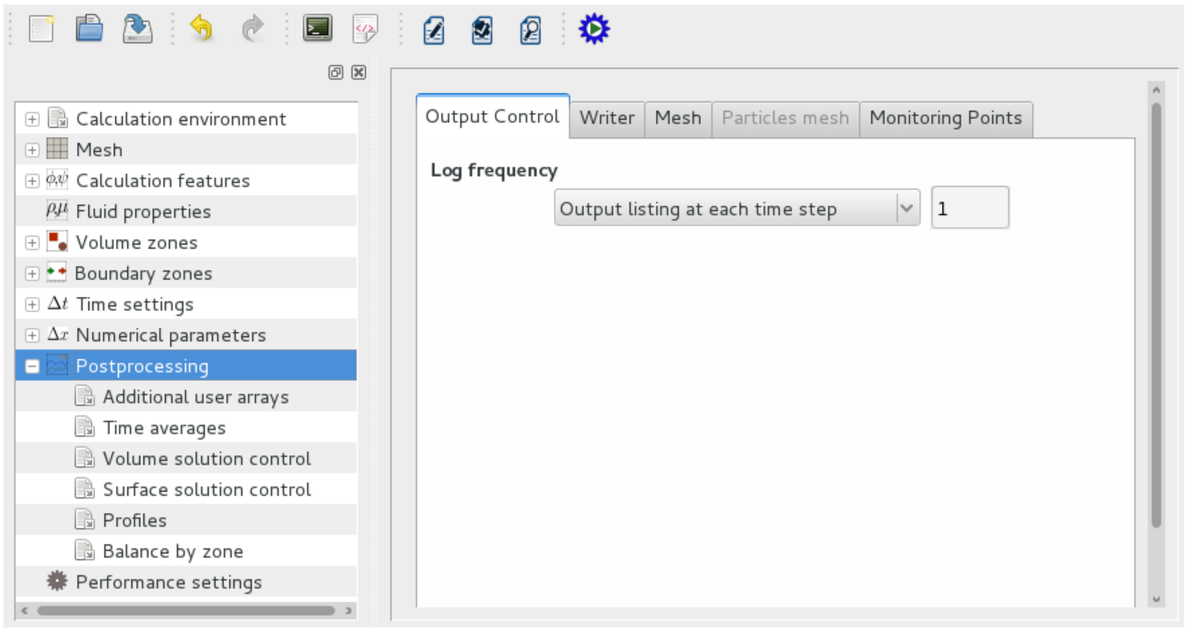
Go to the **Equation parameters** item under the heading **Numerical parameters**. You can define the maximum and minimum value for the `temperature` and for the `scalar2` scalars.



Name	Minimal value	Maximal value
temperature	0	400
scalar2	0	400

Figure III.17: Clipping

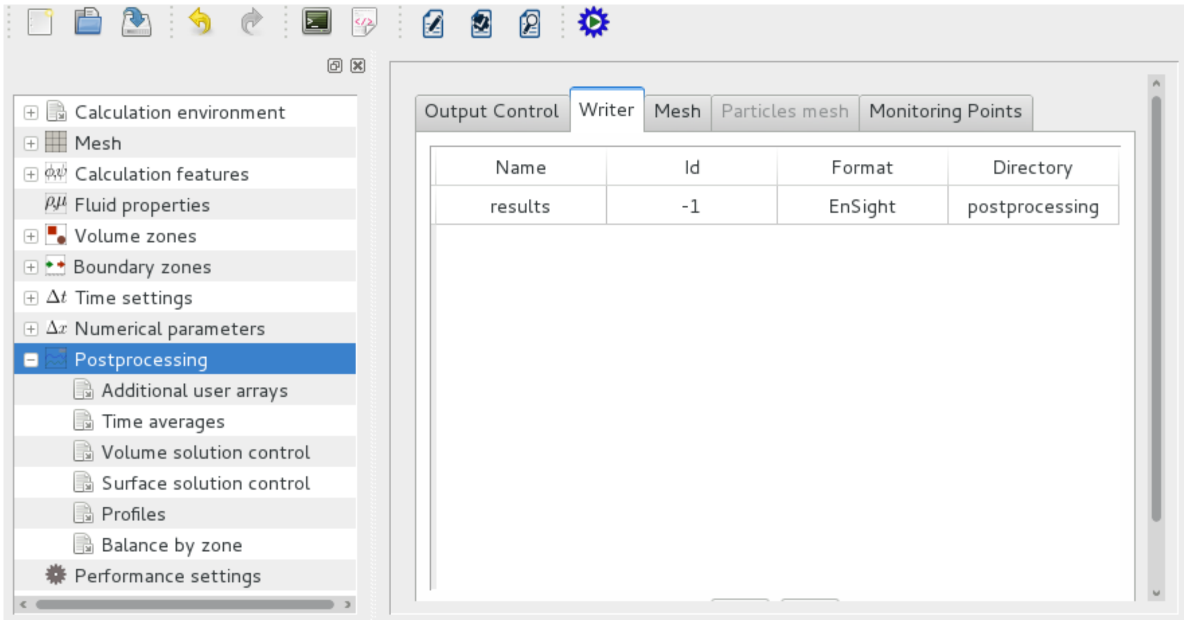
Go to the **Postprocessing** heading to set the output parameters. In the **Output Control** tab, keep the default value for the output listing frequency.



Log frequency: Output listing at each time step, 1

Figure III.18: Output control: log frequency

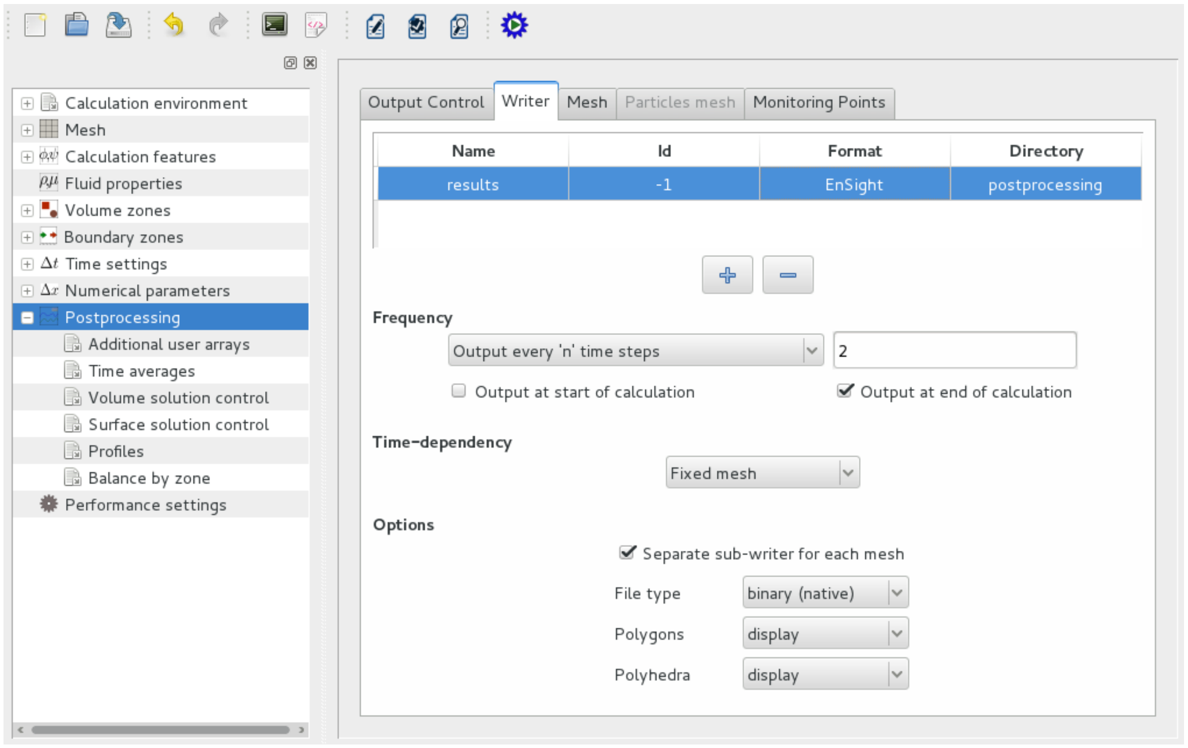
For the post-processing, go to the **Writer** item and click on **results**.



Name	Id	Format	Directory
results	-1	EnSight	postprocessing

Figure III.19: Output control: writer

Now you can select the third option in the **Frequency** (**output every 'n' time steps**) item and set the value of **n** to **2**.



Name	Id	Format	Directory
results	-1	EnSight	postprocessing

Frequency

Output every 'n' time steps: 2

☐ Output at start of calculation ☑ Output at end of calculation

Time-dependency

Fixed mesh

Options

☑ Separate sub-writer for each mesh

File type: binary (native)

Polygons: display

Polyhedra: display

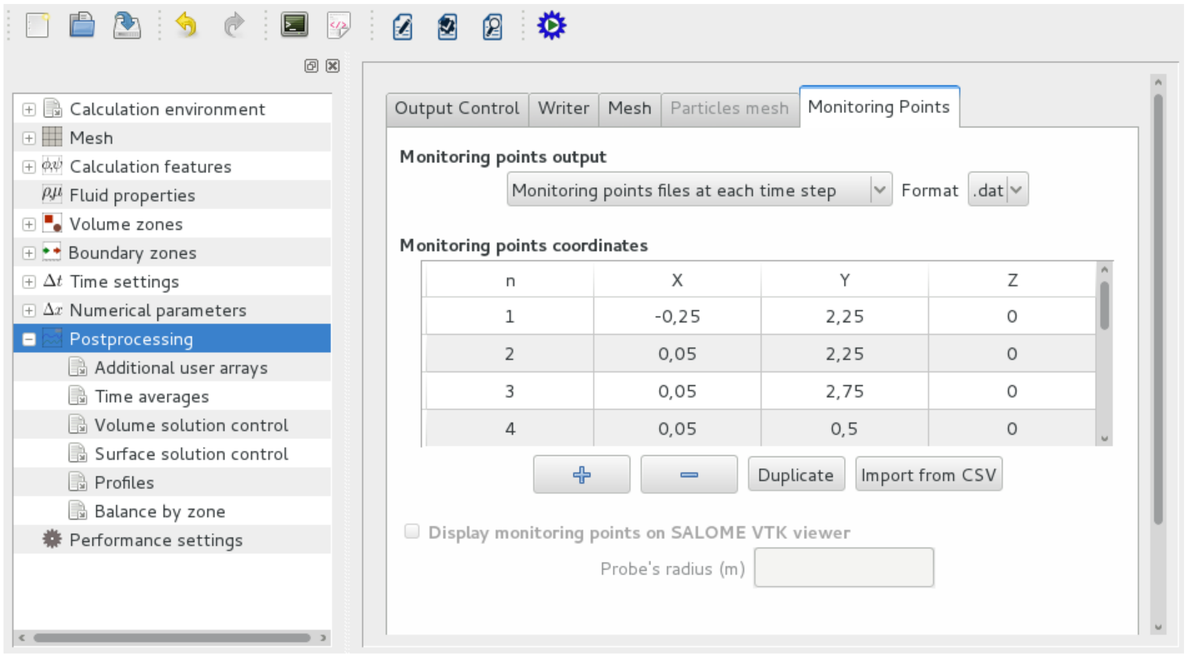
Figure III.20: Output control: results

You can also choose the format. In our case, we choose the EnSight format.

In this case, chronological records on specified monitoring probes are needed. To define the probes, click on the **Monitoring Points** tab. Click on + and enter the coordinates of the monitoring points you want to define.

Repeat the procedure for the other probes. The coordinates of the probes are indicated in the following table (the z coordinate is always 0):

Probe n^o.	**x (m)**	**y (m)**	**z (m)**
1	-0.25	2.25	0.0
2	0.05	2.25	0.0
3	0.05	2.75	0.0
4	0.05	0.50	0.0
5	0.05	-0.25	0.0
6	0.75	-0.25	0.0
7	0.75	0.25	0.0
8	0.75	0.75	0.0



Monitoring points output

Monitoring points files at each time step — Format .dat

Monitoring points coordinates

n	X	Y	Z
1	-0,25	2,25	0
2	0,05	2,25	0
3	0,05	2,75	0
4	0,05	0,5	0

Figure III.21: Output control: monitoring points

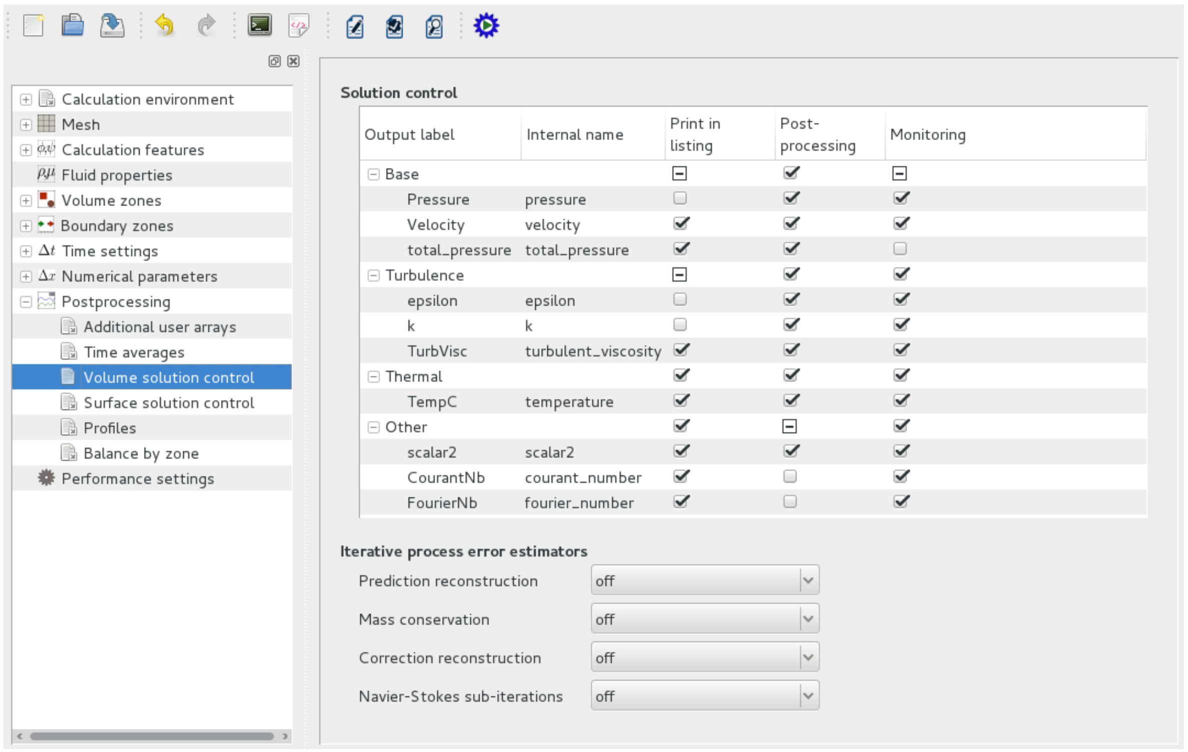
Remember to save the `xml` file regularly.

Go to the **Volume solution control** item to define which variables will appear in the listing, the post-processing and the chronological records.

Uncheck the boxes in front of the **Pressure**, **k** and **epsilon** variables, in the **Print in listing** column. Information on these three variables will not appear in the output listing anymore.

Uncheck the boxes in front of the **CourantNb** and **FourierNb** variables in the **Post-processing** column. These variables will be removed from the post-processing results.

Uncheck the box in front of the **total_pressure** variable in the **Monitoring** column. No chronological record will be created for this variable.



Solution control

Output label	Internal name	Print in listing	Post-processing	Monitoring
Base		⊟	✓	⊟
Pressure	pressure	☐	✓	✓
Velocity	velocity	✓	✓	✓
total_pressure	total_pressure	✓	✓	☐
Turbulence		⊟	✓	✓
epsilon	epsilon	☐	✓	✓
k	k	☐	✓	✓
TurbVisc	turbulent_viscosity	✓	✓	✓
Thermal		✓	✓	✓
TempC	temperature	✓	✓	✓
Other		✓	⊟	✓
scalar2	scalar2	✓	✓	✓
CourantNb	courant_number	✓	☐	✓
FourierNb	fourier_number	✓	☐	✓

Iterative process error estimators

Prediction reconstruction	off
Mass conservation	off
Correction reconstruction	off
Navier-Stokes sub-iterations	off

Figure III.22: Solution control: output configuration

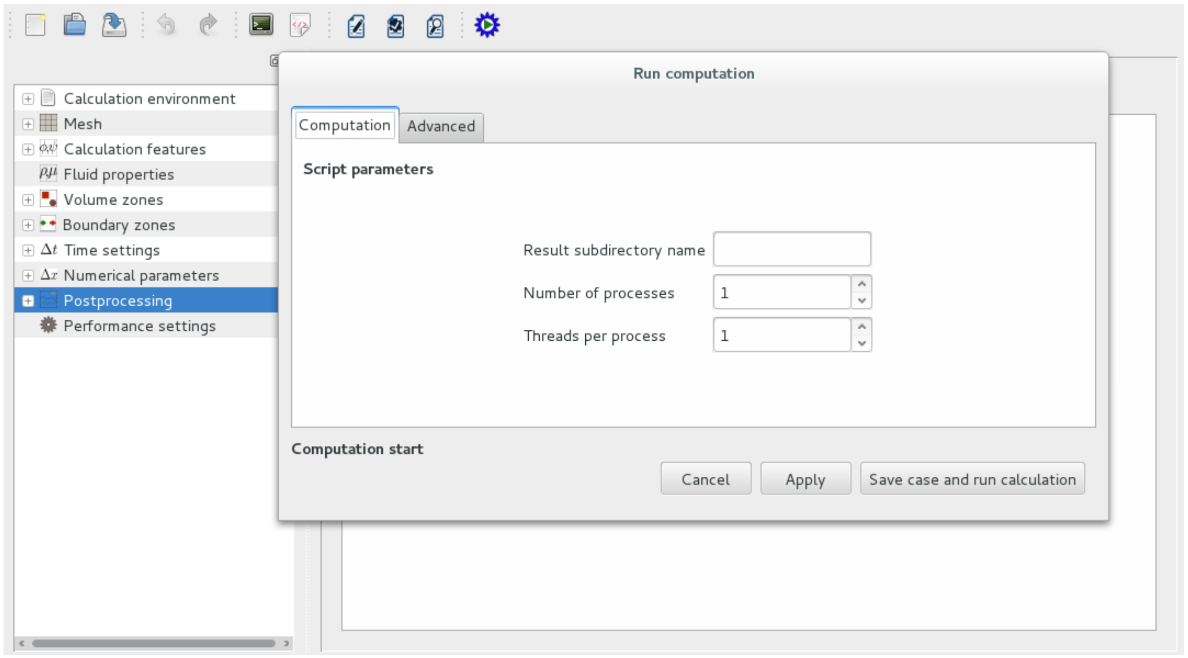
After completing the interface, before running the calculation, some user routines need to be modified to post-process the additional transported scalars on the domain boundary (only boundary temperature post-processing can be dealt with in the GUI).

Go to the folder 🗀`SRC/REFERENCE` and copy `cs_user_parameters.c` in the 🗀`SRC` directory.

In this case `cs_user_parameters.c` is used to add boundary values for all scalars as follows :

```c
void
cs_user_parameters(cs_domain_t *domain)
{
  /* Add boundary values for all scalars */
  {
    int n_fields = cs_field_n_fields();
    for (int f_id = 0; f_id < n_fields; f_id++) {
      cs_field_t *f = cs_field_by_id(f_id);
      if (f->type & CS_FIELD_VARIABLE)
        cs_parameters_add_boundary_values(f);
    }
  }
}
```

To prepare the launch script and, on certain architectures, launch the calculation, click on the icon in the menu bar and a new window will appear as shown below:



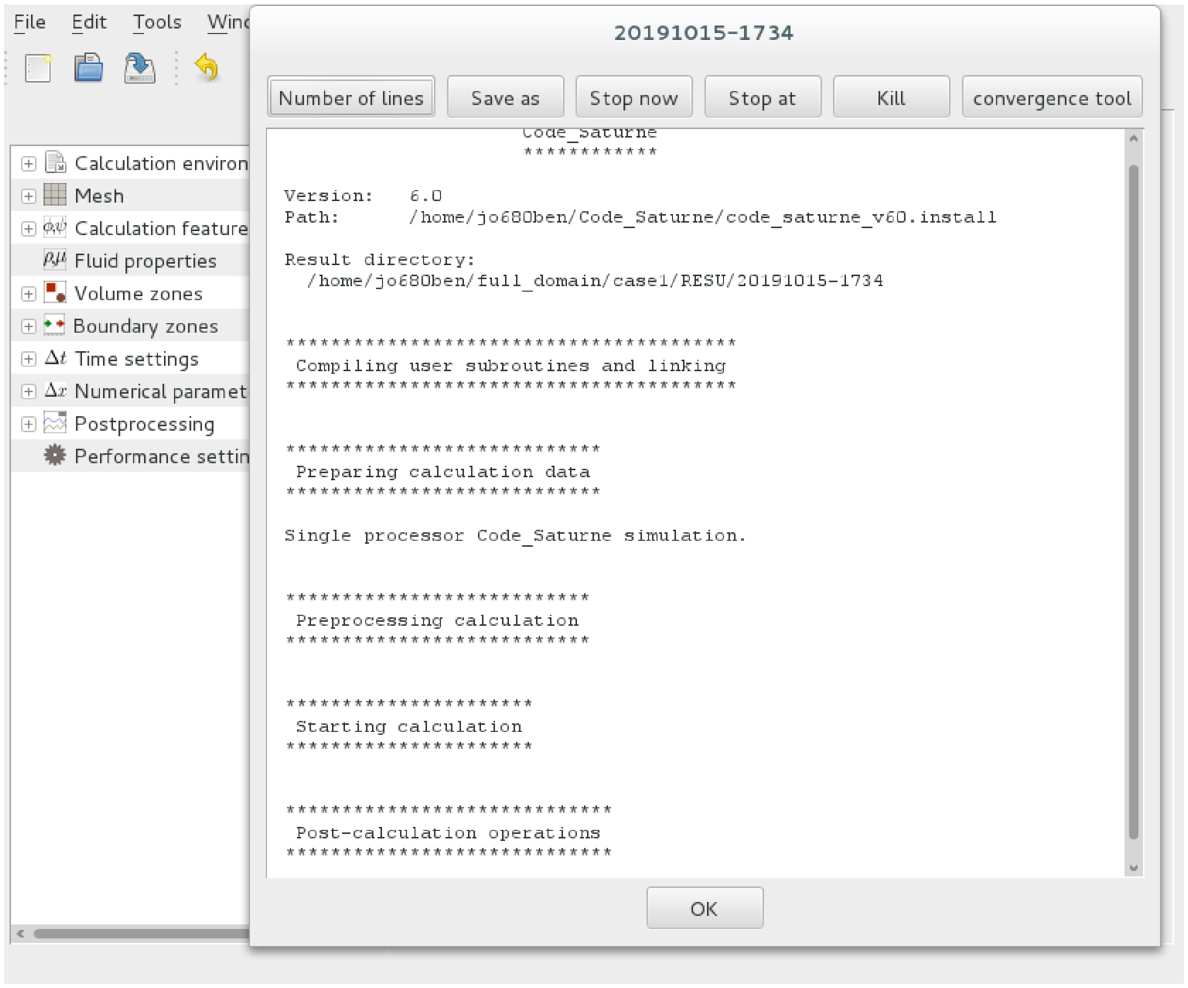
Script parameters

Result subdirectory name
Number of processes 1
Threads per process 1

Computation start

Figure III.23: Prepare batch calculation: computer selection

Run the calculation by clicking on the **Save case and run calculation** button:



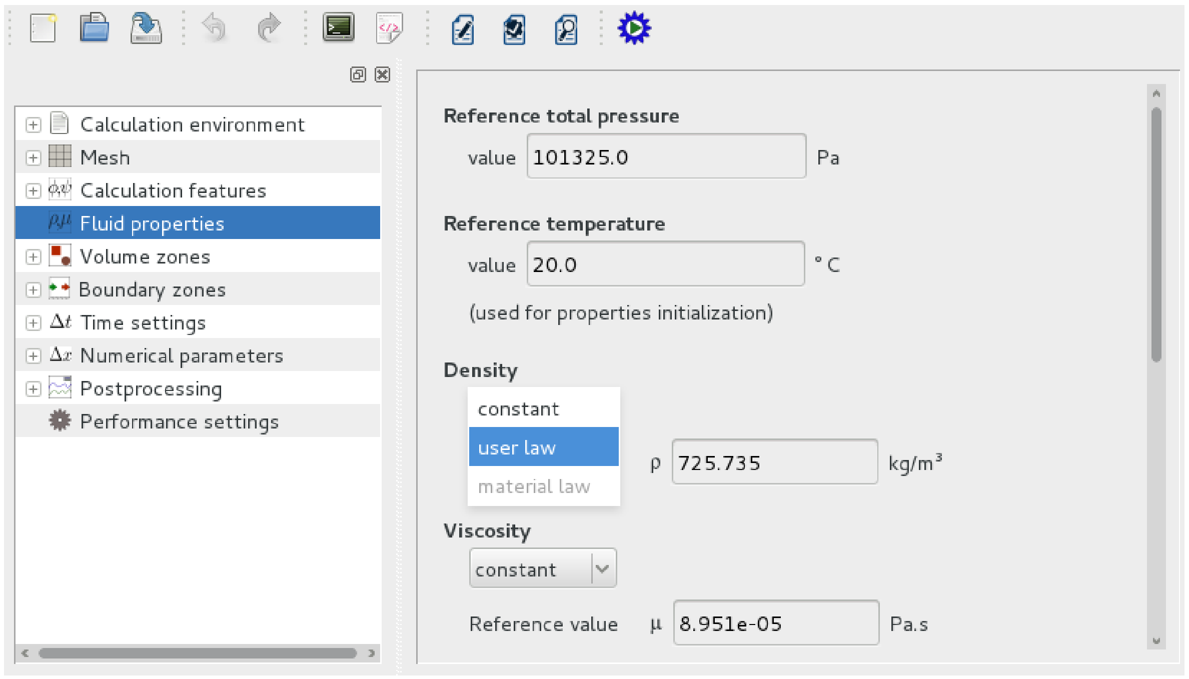
```
                       Code_Saturne
                       ************

Version:   6.0
Path:      /home/jo680ben/Code_Saturne/code_saturne_v60.install

Result directory:
  /home/jo680ben/full_domain/case1/RESU/20191015-1734


 ****************************************
  Compiling user subroutines and linking
 ****************************************


 ****************************
  Preparing calculation data
 ****************************

Single processor Code_Saturne simulation.


 ***************************
  Preprocessing calculation
 ***************************


 **********************
  Starting calculation
 **********************


 *****************************
  Post-calculation operations
 *****************************
```

Figure III.24: Run

2 Solution for CASE2

Only a few elements are different from `case1`.

In this case the density becomes variable. Go to the **Fluid properties** heading and change the nature of the density from **constant** to **user law**.



Reference total pressure

value 101325.0 Pa

Reference temperature

value 20.0 °C

(used for properties initialization)

Density

user law ρ 725.735 kg/m³

Viscosity

constant

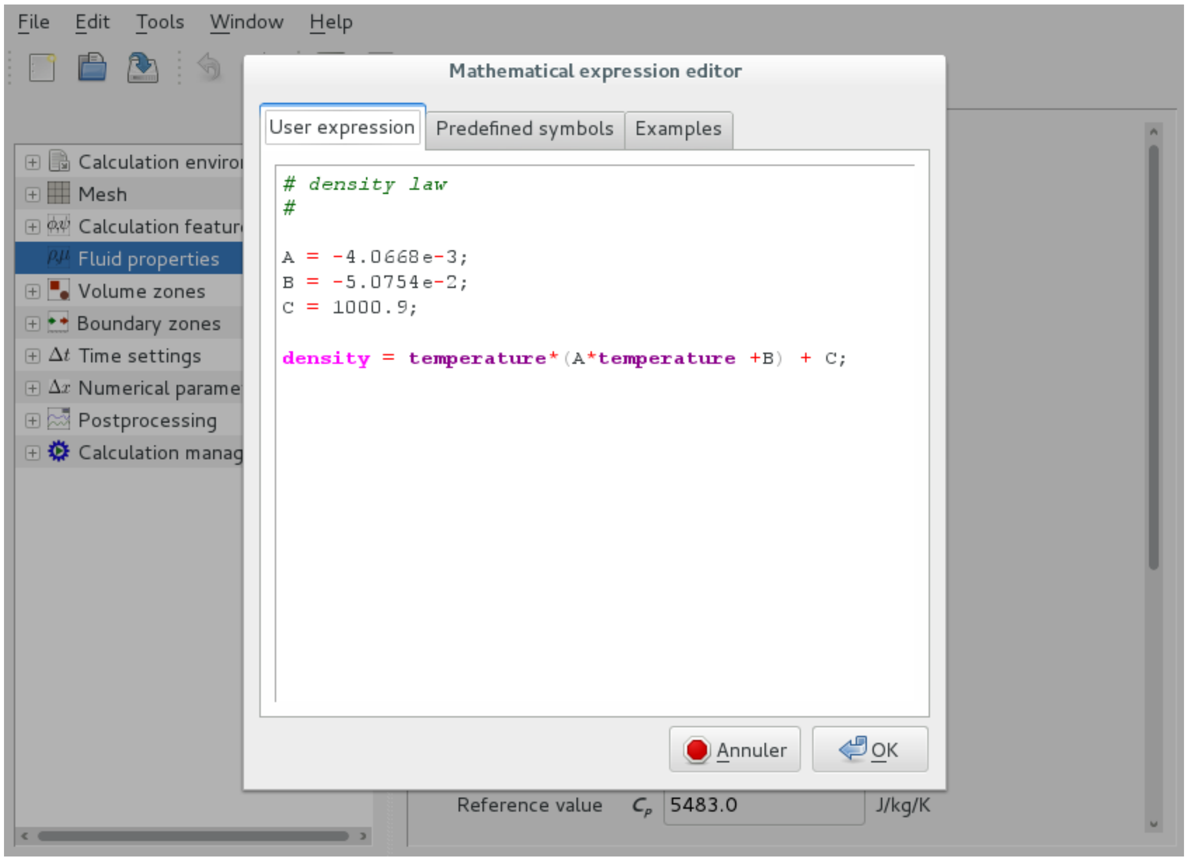
Reference value μ 8.951e-05 Pa.s

Figure III.25: Fluid properties: variable density

The user law of the density is defined as follows in the *Code_Saturne* (GUI):

```
density = temperature * ( -4.0668E-03*temperature - 5.0754E-02 ) + 1000.9;
```

Click on the highlighted icon and define the user law in the window that pops up. Follow the format used in the **Examples** tab.



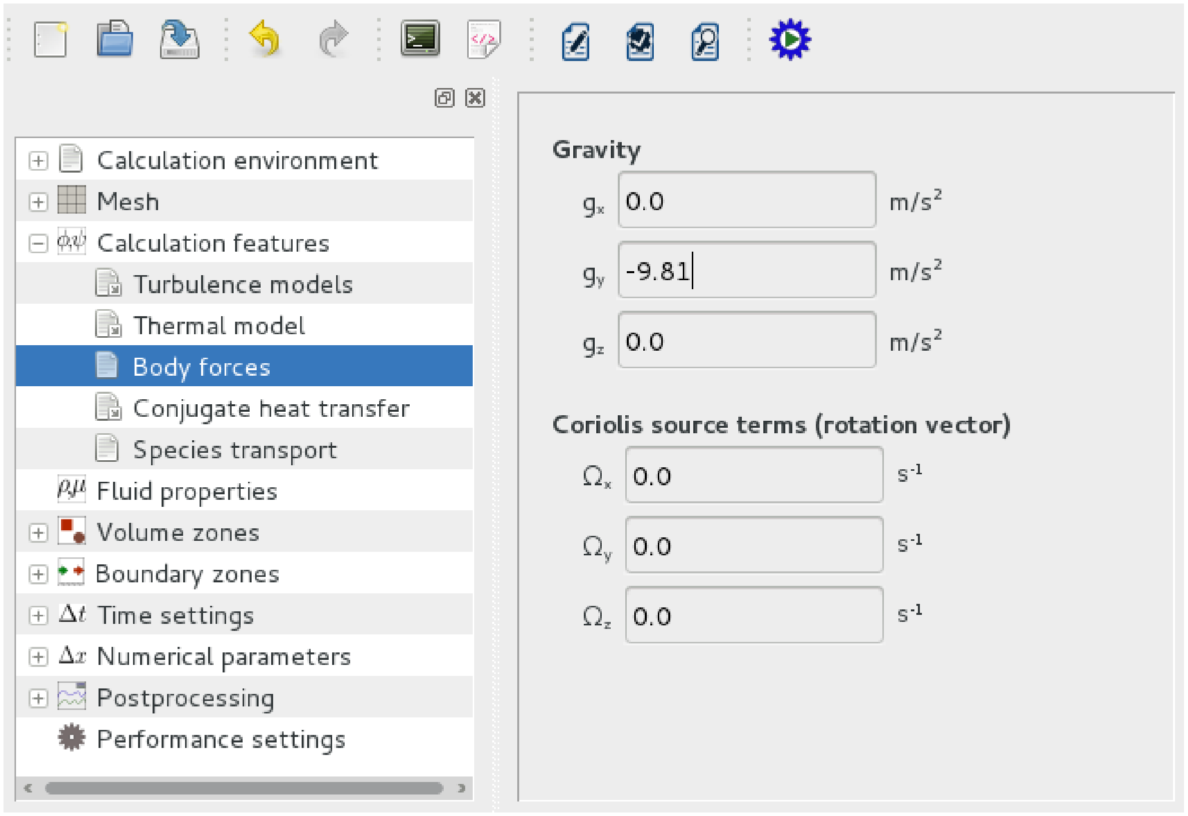
```
# density law
#

A = -4.0668e-3;
B = -5.0754e-2;
C = 1000.9;

density = temperature*(A*temperature +B) + C;
```

Reference value C_p 5483.0 J/kg/K

Figure III.26: Fluid properties/Variable density: user expression

As the density is variable, the influence of gravity has to be considered. Go to **Body forces** item under **Calculation features** heading and set the value of each component of the gravity vector.

```
g_x = 0.0 g_y = -9.81 g_z = 0.0
```



Gravity

g_x	0.0	m/s^2
g_y	-9.81	m/s^2
g_z	0.0	m/s^2

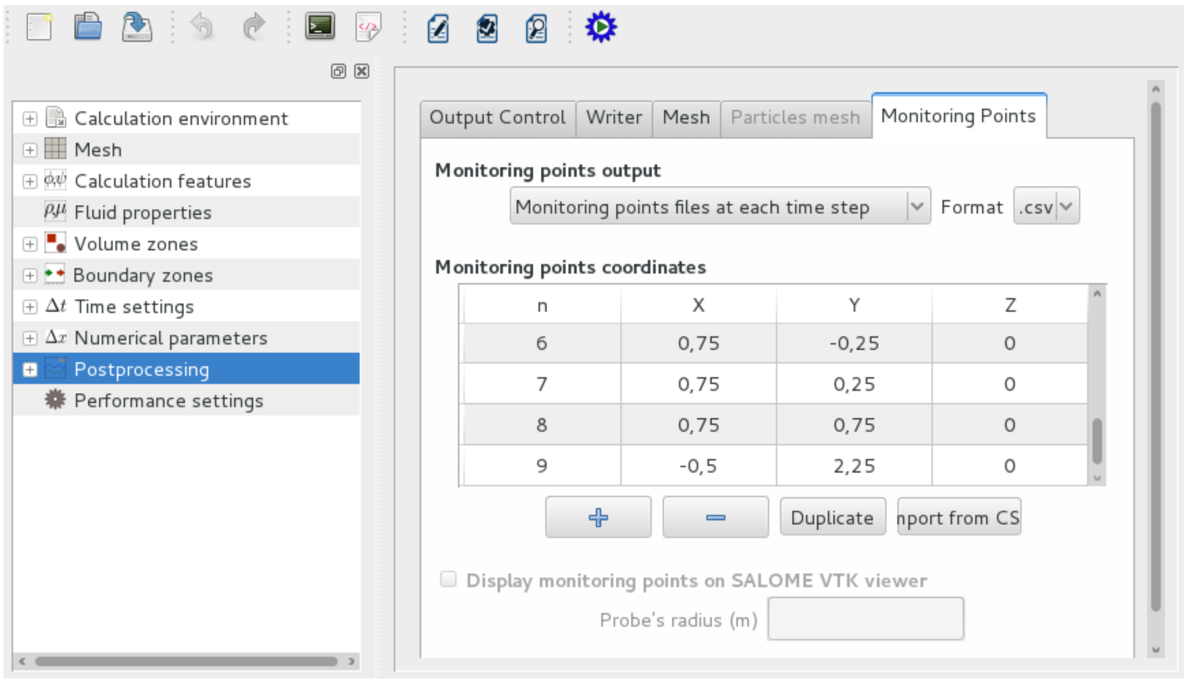
Coriolis source terms (rotation vector)

Ω_x	0.0	s^{-1}
Ω_y	0.0	s^{-1}
Ω_z	0.0	s^{-1}

Figure III.27: Fluid properties: gravity

Add a monitoring point close to the entry boundary condition in the **Monitoring Points** tab under the **Postprocessing** heading.

Probe	x (m)	y (m)	z (m)
9	-0.5	2.25	0.0



Monitoring points output

Monitoring points files at each time step   Format .csv

Monitoring points coordinates

n	X	Y	Z
6	0,75	-0,25	0
7	0,75	0,25	0
8	0,75	0,75	0
9	-0,5	2,25	0

Figure III.28: New monitoring probe

After completing the interface, before running the calculation, some Fortran user routines need to be modified.

Go to the folder 🗀 `SRC/REFERENCE` and copy `cs_user_boundary_conditions.f90` in the 🗀 `SRC` directory.

• **cs_user_boundary_conditions.f90**:

In this case, `cs_user_boundary_conditions.f90` is used to specify the time dependent boundary condition for the temperature. Refer to the comments in the routine or to the *Code_Saturne* user manual for more information on this routine.

In our case, you need to identify the boundary faces of color **1**.

The command

```
call getfbr('1',nlelt,lstelt)
```

will return an integer `nlelt`, corresponding to the number of boundary faces of color 1, and an integer array `lstelt` containing the list of the `nlelt` boundary faces of color 1.

Remark: Note that the string **1 can be more complex and combine different colors, group references or geometrical criteria**, with the same syntax as in the Graphical Interface.

For each boundary face `ifac` in the list, the Dirichlet value is given in the multi-dimension array `rcodcl` as follows:

```
if (ttcabs.lt.3.8d0) then
 do ielt = 1, nlelt
    ifac = lstelt(ielt)
    rcodcl(ifac,isca(2),1) = 20.d0 + 100.d0*ttcabs
  enddo
else
  do ielt = 1, nlelt
    ifac = lstelt(ielt)
    rcodcl(ifac,isca(2),1) = 400.d0
  enddo
endif
```

`isca(2)` refers to the first scalar and `ttcabs` is the current physical time.

See the example `cs_user_boundary_conditions-base.f90` file in the subdirectory 🗀 `SRC/EXAMPLES` to complet correctly your boundary conditions for this `case2`.

Remark: Note that, although the inlet boundary conditions for temperature are specified in the `cs_user_boundary_conditions.f90` file, it is necessary to specify them also in the Graphical Interface.

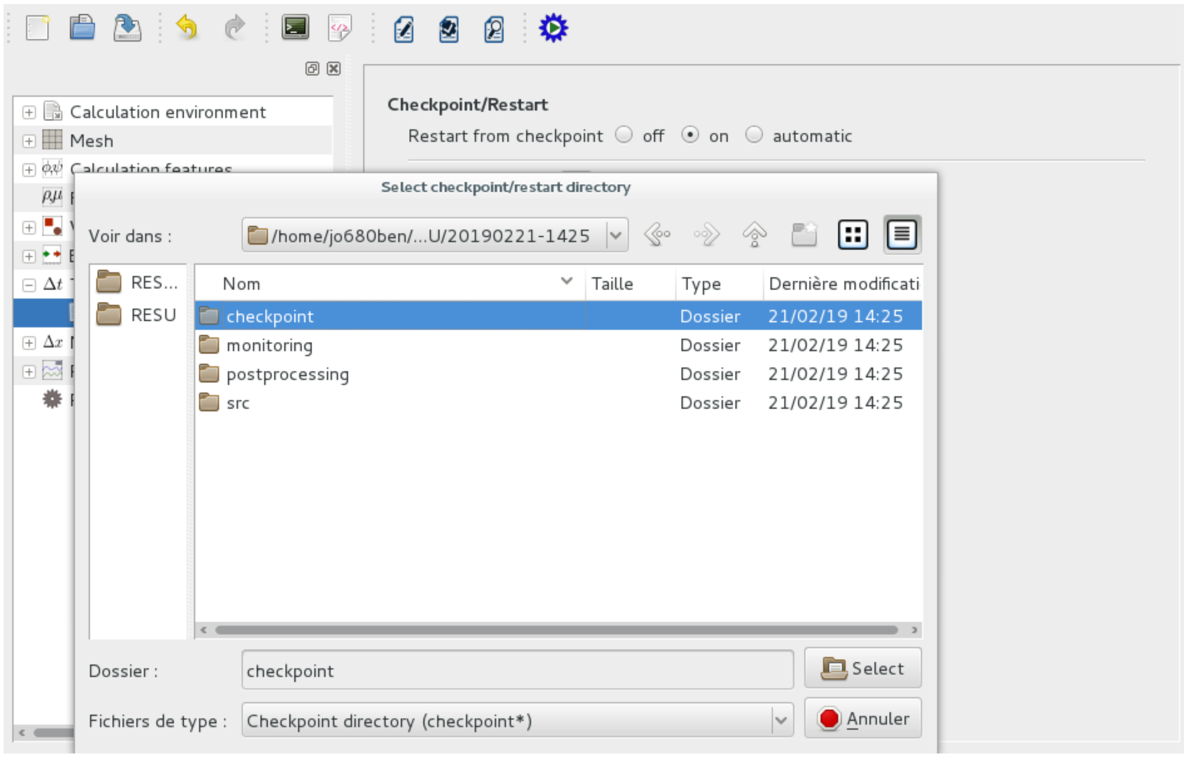
The value given in the Interface can be anything, it will be overwritten by the Fortran routine.

After updating the Fortran file, run the calculation as explained in `case1`.

When a calculation is finished, *Code_Saturne* stores all the necessary elements to continue the computation in another execution, with total continuity. These elements are stored in several files, grouped in a `yyyymmdd-hhmm/checkpoint` subdirectory, in the `RESU` directory.

In this case, after the first calculation is finished, a second calculation will be run, starting from the results of the first one.

Go directly on the **Start/Restart** item under the heading **Time settings**. Activate the **Checkpoint/Restart** by clicking the On box.



Checkpoint/Restart

Restart from checkpoint ( ) off (•) on ( ) automatic

Voir dans : /home/jo680ben/...U/20190221-1425

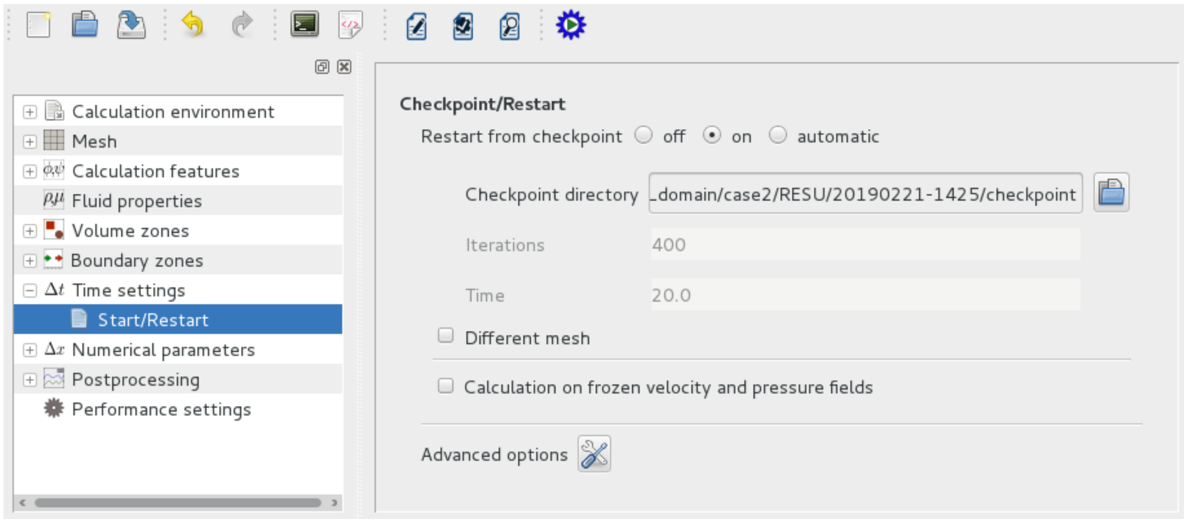
Nom	Taille	Type	Dernière modificati
checkpoint		Dossier	21/02/19 14:25
monitoring		Dossier	21/02/19 14:25
postprocessing		Dossier	21/02/19 14:25
src		Dossier	21/02/19 14:25

Dossier : checkpoint

Fichiers de type : Checkpoint directory (checkpoint*)

Figure III.29: Start / Restart

A window opens, with the architecture of the study sub-directories. In the 🗀`RESU` folder, click on the folder 🗀`yyyymmdd-hhmm/checkpoint` (where `yyyymmdd-hhmm` corresponds to the reference of the first calculation results). Then click on Validate.



Checkpoint/Restart

Restart from checkpoint ○ off [✓ ● on] ○ automatic

Checkpoint directory _domain/case2/RESU/20190221-1425/checkpoint

Iterations 400

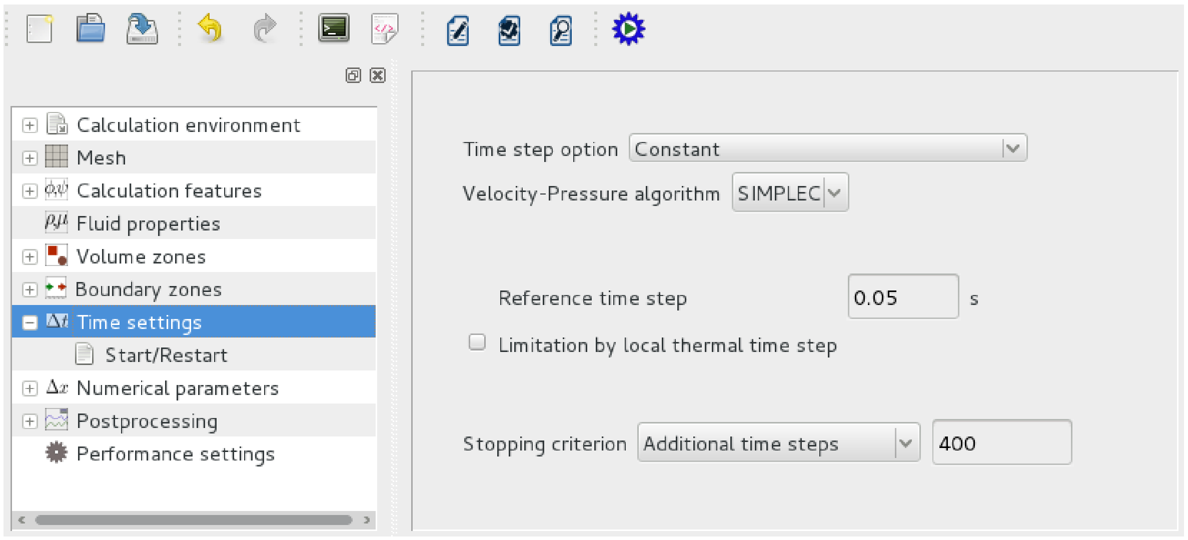
Time 20.0

☐ Different mesh

☐ Calculation on frozen velocity and pressure fields

Figure III.30: Start/Restart: selection of the restart directory

Go to the **Time settings** heading and change the **Stopping criterion** from **Number of time steps** to **Additional time steps**. Set it to 400 which means another 400 iterations (the total number of iterations is 700 iterations).



Time step option: Constant
Velocity-Pressure algorithm: SIMPLEC
Reference time step: 0.05 s
Limitation by local thermal time step: ☐
Stopping criterion: Additional time steps 400

Figure III.31: Time step

Eventually, run the calculation.

3 Solution for CASE3

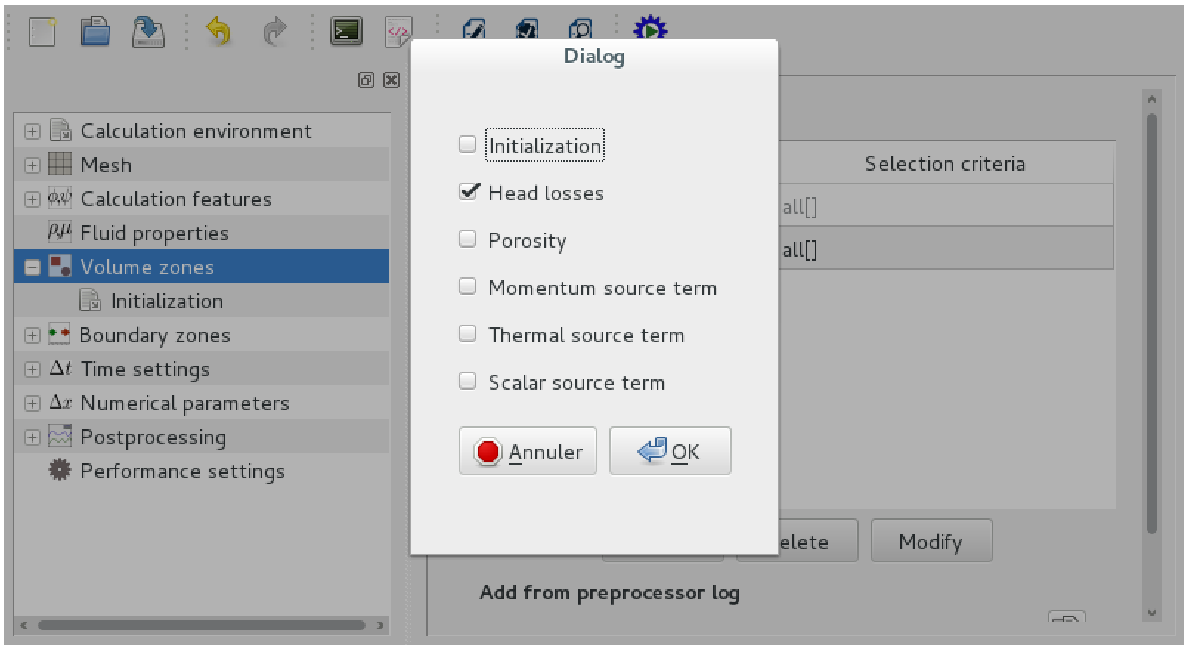
This case is similar to `case2`, with the following differences:

- **Step 1**: define head losses in the fluid domain,
- **Step 2**: compute the spatial average of temperature scalar,
- **Step 3**: parallel computation on 2 processors,
- **Step 4**: dealing with a user results file.

• **Step 1-1**: Define the head losses in the Graphical User Interface (GUI)

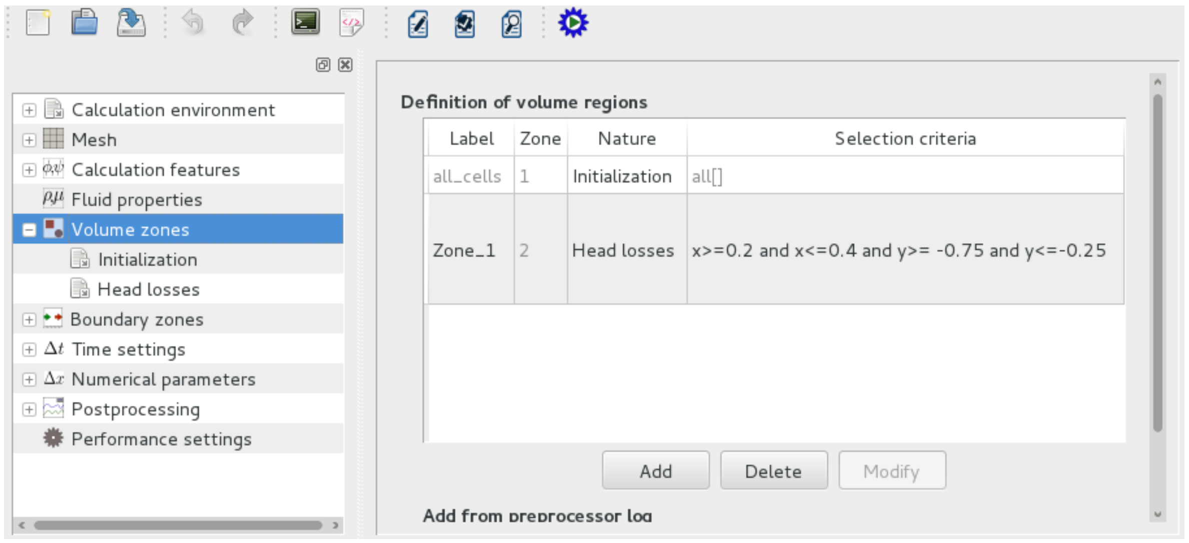
Go to **Volume zones** folder. Click on Add, unselect **Initialization** and select **Head losses** in the Dialog window that pops up (see Figure III.32). In the box named **Label**, name the head losses region.
Define the limits of the head losses region in **Selection criteria**. The associated character string to enter is as below:

```
''x >= 0.2 and x <= 0.4 and y >= -0.75 and y <= -0.25''
```



Initialization [ ]
Head losses [✓]
Porosity [ ]
Momentum source term [ ]
Thermal source term [ ]
Scalar source term [ ]

Figure III.32: Creation of the head losses region



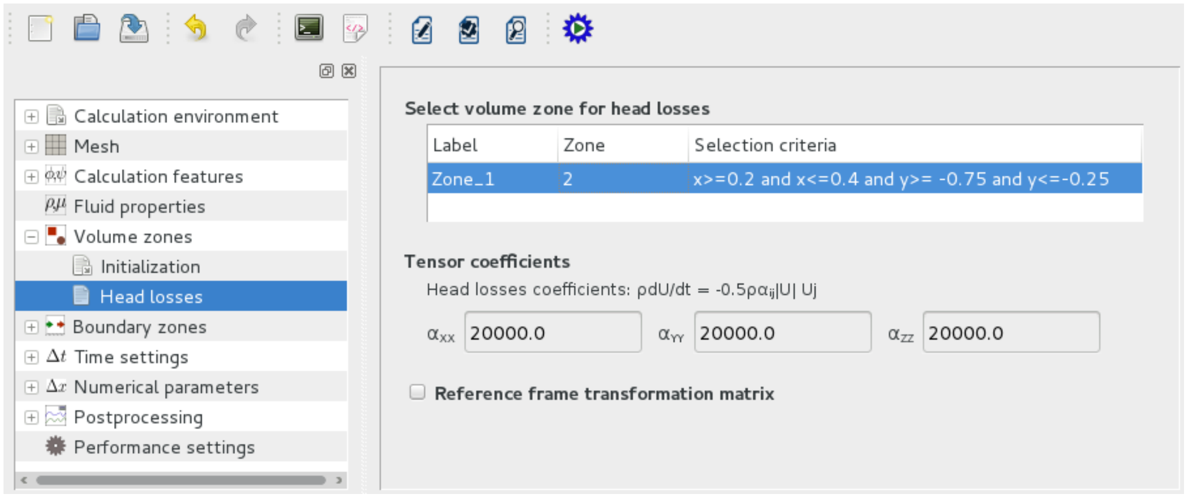
Definition of volume regions

Label	Zone	Nature	Selection criteria
all_cells	1	Initialization	all[]
Zone_1	2	Head losses	x>=0.2 and x<=0.4 and y>= -0.75 and y<=-0.25

Figure III.33: **Selection criteria** of the head losses region

- **Step 1-2**: Specify the head losses coefficients α_{ii}

To specify the head losses coefficients go to the **Head losses** item and select the name of the head losses volume region. In this example, the coefficient is isotropic so that we use the same value for each α_{ii}. Please note that $\alpha_{ii} = 2 \times K_{ii}$, therefore if $K_{ii} = 10^4$, $\alpha_{ii} = 2 \times 10^4$.



Select volume zone for head losses

Label	Zone	Selection criteria
Zone_1	2	x>=0.2 and x<=0.4 and y>= -0.75 and y<=-0.25

Tensor coefficients

Head losses coefficients: ρdU/dt = -0.5ρα_{ij}|U| Uj

α_{xx} 20000.0 α_{yy} 20000.0 α_{zz} 20000.0

☐ Reference frame transformation matrix

Figure III.34: Head losses coefficients

- **Step 2**: Compute the spatial average of **temperature**

The computation of the spatial average must be done in the `cs_user_extra_operations.c` routine.

The following code computes the spatial average temperature and writes it in a file called `moy.dat`.

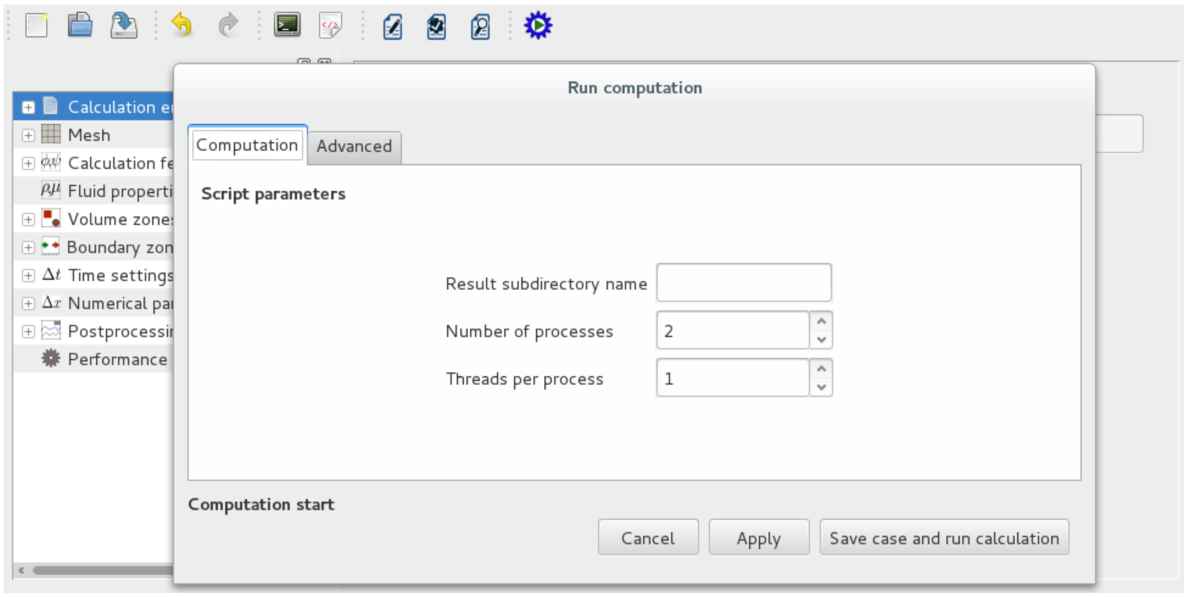
```
/* Global declaration of the moy.dat file that will be filled with Tavg */
static FILE *file = NULL;

void
cs_user_extra_operations(cs_domain_t      *domain)
```

```
{
  /* Variables declaration */
  /* Get pointers to the mesh and mesh quantities structures */
  const cs_mesh_t *m= cs_glob_mesh;
  const cs_mesh_quantities_t *fvq = cs_glob_mesh_quantities;

  /* Number of cells */
  const int n_cells = m->n_cells;

  /* Cell volumes */
  const cs_real_t *cell_vol = fvq->cell_vol;

  /* Get the temperature field */
  const cs_field_t *temp = cs_field_by_name_try("temperature");

  /* Cell volumes sum, Temp * volumes sum and Tavg */
  cs_real_t voltot = 0., temptot =0., Tavg =0.;

  /* Compute the sum of the cell volumes */
  for (int ii = 0 ; ii < n_cells ; ii++)
    voltot += cell_vol[ii];

  /* Compute the sum T*vol */
  for (int ii = 0 ; ii < n_cells ; ii++)
    temptot += temp->val[ii]*cell_vol[ii];

  /* Parallel sums */
  cs_parall_sum(1, CS_DOUBLE, &voltot);
  cs_parall_sum(1, CS_DOUBLE, &temptot);

  /* Compute Tavg */
  Tavg = temptot / voltot ;

  /* Open the file moy.dat at the first iteration
     and write the first comment line only on the
     first processor (0 in parallel, -1 in serial) */
  if (cs_glob_time_step->nt_cur == 1 && cs_glob_rank_id <= 0) {
    file = fopen("moy.dat", "a");
    fprintf(file, "#Time (s)   Average Temperature (C) \n");
  }

  /* Print the average temperature at the current time step
     on the first processor only */
  if (cs_glob_rank_id <= 0)
    fprintf(file, "%.6f  %.6f\n",cs_glob_time_step->t_cur, Tavg);

  /* Close the file moy.dat at the last iteration
     on the first processor only */
  if (cs_glob_time_step->nt_cur == cs_glob_time_step->nt_max
      && cs_glob_rank_id <= 0)
    fclose(file);
}
```

• **Step 3**: Choose a computation with 2 processors

This modification will be done in the **Calculation management** item.
To run the calculation on two processors, simply change the number of processors indicator to 2. The launch script will automatically deal with the rest.

Do not forget to set the right **Reference time step** and **Number of iterations** under the heading **Time settings**.



Script parameters

Result subdirectory name

Number of processes: 2

Threads per process: 1

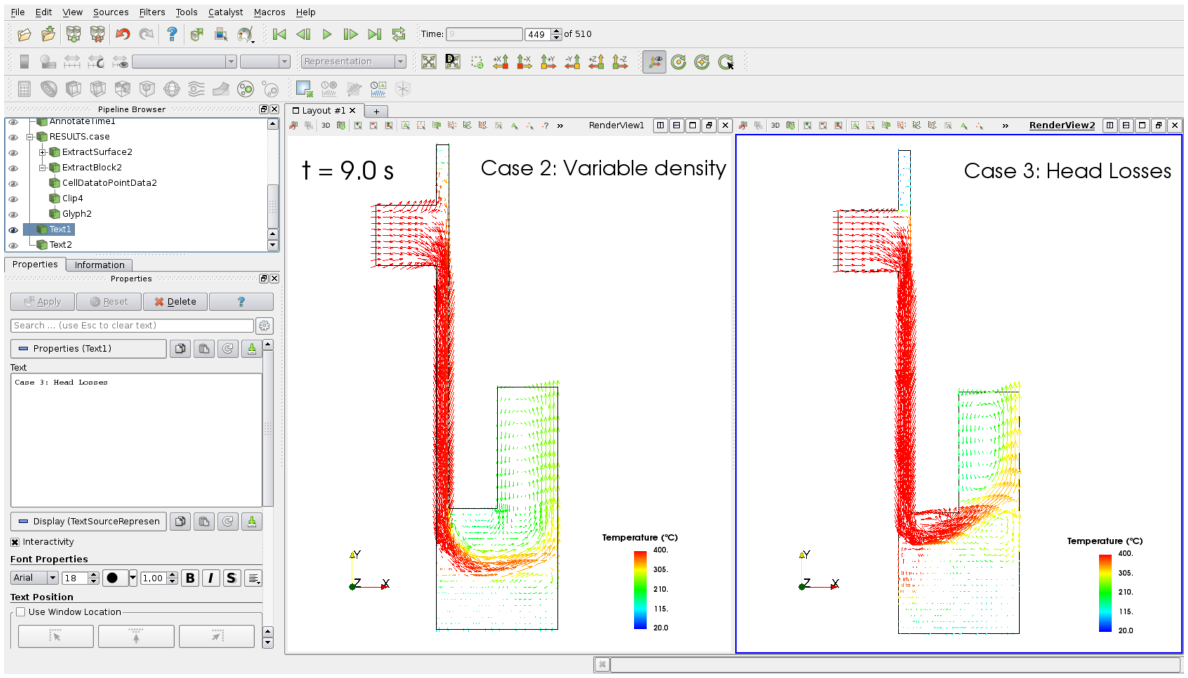
Computation start

Figure III.35: Number of processors

• **Step 4**: Dealing with a user results file `moy.dat`

The new user file `moy.dat` created by `cs_user_extra_operations.c` will be written directly in the 🗀`yyyymmdd-hhmm` results subdirectory created at the end of the computation in the 🗀`RESU` directory.

Remark 3.1 : We do not have to specify the name of the new user file in the Graphical User Interface (GUI), like in previous *Code_Saturne* versions. The name of the new user file had to be identified in the launch script in order to be automatically copied in the 🗀`RESU` directory; this is not requested anymore.



Properties (Text1)

Text

```
Case 3: Head Losses
```

Font Properties: Arial, 18, 1.00

Figure III.36: User results files