TRUST Tutorial V1.9.4

CEA Saclay

Support team: trust@cea.fr

May 29, 2024

Overview

Initialization

To initialize TRUST environment:

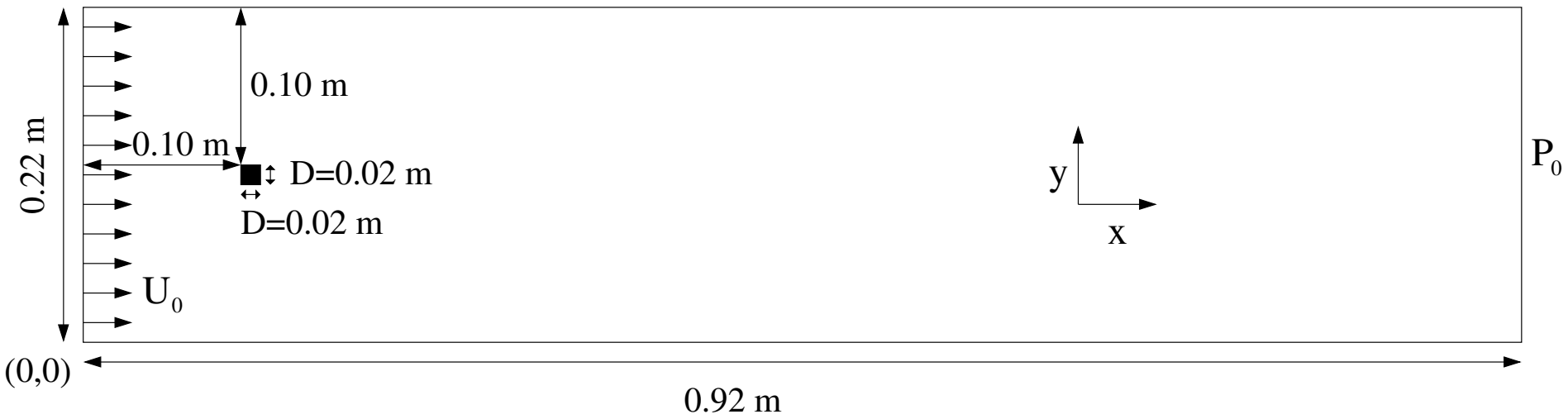
- On CEA Saclay PCs, TRUST version is available with:
  source /home/trust_trio-public/env_TRUST-1.9.4.sh
- On your own computer, download and install the latest version of TRUST in your local folder $MyPathToTRUSTversion (unless this was already performed), then run on your terminal:
  source $MyPathToTRUSTversion/env_TRUST.sh

Text Editor configuration

Several editors (**vim**, **emacs**, **nedit**, **gedit**) can be configured to highlight TRUST keywords when editing data files.

- If you prefer using **nedit**, please do the following:
  - Run **nedit**, and select Preferences → Save Defaults.
  - Then run **trust -config nedit**, the message "nedit.rc updated" should appear.
- If you prefer using **gedit**, run:
  - **trust -config gedit**

Geometry



- Fluid: $\mu = 3.7\,10^{-5} kg.m^{-1}.s^{-1}$, $\rho = 2 kg.m^{-3}$ and $Re = \frac{U_0 H_{inlet} \rho}{\mu} = \frac{1 \times 0.22 \times 2}{3.7\ 10^{-5}} = 11891$
- Boundary conditions:
  - Inlet with uniform velocity: $U_0 = 1 m.s^{-1}$
  - Outlet with constant pressure: $P_0 = 0$
  - Square cylinder: No-slip wall
  - Upper and Lower walls: Symmetry

Create a study

- First, load TRUST environment.
- Open a terminal and create a directory for this tutorial:
  mkdir -p Formation_TRUST/yourname
  cd Formation_TRUST/yourname
- Copy a test case from TRUST's database with:
  trust -copy Obstacle
  cd Obstacle
- Ask for trust script options:
  trust -help
- Ask for help on the options of TRUST's executable:
  trust Obstacle -help_trust
- Run the test case with:
  trust Obstacle

Probes and parameters

- Edit the data file Obstacle.data and set the time step to 0.004s:
  nedit Obstacle.data &
- Replace the keyword "**format lml**" with "**format lata**" inside the post-processing block in order to use VisIt post-processing tool during and/or at the end of calculation.

Visualization during the calculation

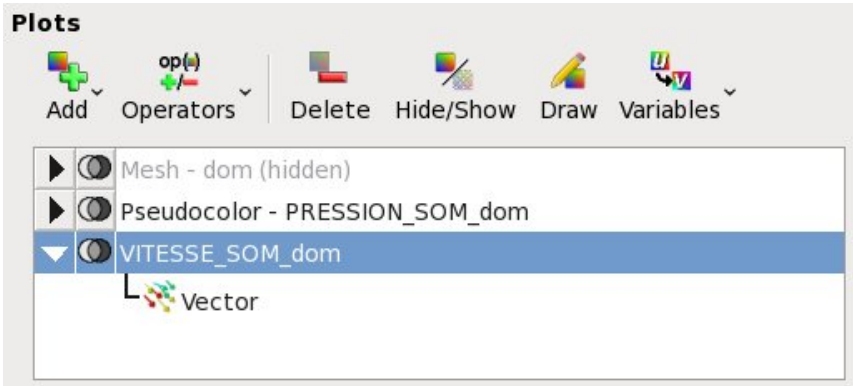
- Launch the "trust -evol" tool with:
  trust -evol Obstacle &
  This tool allows to launch calculation and visualize results
- To run the calculation, click on the button "Start computation!".
- Visualization:
  - Select "PRESSION(X=0.13,Y=0.105)" in the left list and click on "Plot" to draw the evolution of the pressure at the probe location.
  - Check the velocity profile behind the square cylinder by plotting "VITESSE_X(X=0.14,Y=0.115)" and "VITESSE_Y(X=0.14,Y=0.115)".
  - Visualize the equation residuals on the same plot, select "$Ri = max\left|\frac{dV}{dt}\right|$" and "$residu = max\,|Ri|$" using the button "Plot on same" or select the two graphs with "Ctrl" button then "plot".

VisIt

- To quit this tool, close the GUI.
- Once the calculation is finished, visualize the results with VisIt: **visit &** or using "trust -evol" tool: **trust -evol Obstacle &** and click on "Visualisation" on the right menu.
  - First, we are going to configure VisIt: In the menu File → Open file, select Off instead of Smart for File grouping option. For the Filter, specify *.lata to list only the lata files (results). Then save your choices, in the menu Options → Save Settings.
  - In the menu File → Open file, select the Obstacle.lata file.
  - Visualize the mesh in the "Plots" area with "Add → Mesh → dom" then click on the button "Draw". Zoom and move the mesh in the right window. You can un-zoom with right button (View → Reset view) or with a combination of "Ctrl" keypad and left button.
  - Visualize the pressure field (Plots area: "Add → Pseudocolor → PRESSION_SOM_dom" + Draw then select the last time on the Time slider)
  - Suppress or hide the mesh (Select Mesh then click on Delete or Hide/Show).

VisIt

- Visualize the velocity field (Plots area: "Add → Vector → VITESSE_SOM_dom" + Draw). You can change each plot attributes:
  - ◇ click once onto the small arrow "▶" then
  - ◇ double click on the item Vector (cf the figure below). For example, change the number of vectors being plotted (by default 400, set it to 40000 then click the button "Make default" and save definitively this modification with the menu Options → Save Settings). You need to click "Apply" to update. Then click "Dismiss" to close the window.



- Print your visualization (File → Save window): a PNG file is created into your working directory.

VisIt

- Add a second screen with "Windows → Layouts → 1x2",
- Plot a pressure horizontal profile:
  - select the pressure field,
  - on the visualisation, use the right click and select "Mode → Line out",
  - then define your profile with left button,
  - click on the origin point, let the left button pushed, and release at the end point.
  - The profile is shown on the second window.
- You notice that it is necessary to update (button Draw) the right window after adding a new plot or changing an option. It is possible to automatically update by activating "Auto apply" on the top right of the VisIt's GUI.
- You can create create new fields (expression) with "Controls → Expressions → New" by using existing variables and complex functions and visualize it.
- You can animate your visualization and/or create a movie (File → Save movie)
- You can operate calculations on variables with complex queries (Controls → Query),
- You can save a complex session (File → Save session) and reopen it during a next analyze with VisIt (File → Restore session),

Outputs and resuming calculation

  - During a 3D visualization, you will use one of the available Operators (In Plots, "Operators → Slicing → Slice") to create a 2D slice either in a 3D space, or projected to a 2D space.
- For more information on **VisIt**, you can refer to:
  - the **VisIt** website and its manuals: https://wci.llnl.gov/simulation/computer-codes/visit/manuals
  - the **VisIt** user community web site: http://visitusers.org
  - or send an email to the VisIt software users community at: visit-users@elist.ornl.gov
- Edit the different output (*.out) files to read the complete balances (mass, stress, energy, ...) on the whole domain or at the boundaries.
- Now we want to edit the data file in order to resume the calculation. So, open it using "PLOT2D" tool: **trust -evol Obstacle &**.

Outputs and resuming calculation

- Find the last backup time of the previous calculation in the .err file (or in the bottom right file in the "PLOT2D" tool if it is still running).
- Edit your data file with "Edit data", then modify **tinit**, **tmax** values in the object "mon_schema".
- Add in the problem description block just before the last "}":
  reprise binaire Obstacle_pb.sauv
  (The file "Obstacle_pb.sauv" must have been created during the first run.)
- Save and close the window.
- Resume the calculation again with "Start calculation!" button. You can see that values are added to the first probes during the new calculation.

⇒ *Remark:* to resume your calculation, you can also use the keyword **resume_last_time** instead of **reprise** and only change the **tmax** value (cf Reference Manual).

Probes and fields

- Edit the data file Obstacle.data:
  nedit Obstacle.data &
- Add to the post-processing block of Obstacle.data the following elements:
  - A pressure probes segment (22 probes between points (0.01, 0.12) and (0.91, 0.12)).
  - A velocity probes segment (22 probes between points (0.92, 0.00) and (0.92, 0.22)) to plot the velocity profile behind the square cylinder.
  - Change fields post-processing period from 1s to 0.5s.
  - Add the vorticity to the fields to the list of post-processed fields. To find the appropriate keyword, have a look to the Generic Guide:
    trust -doc &

  → You have access to useful resources in the TRUST index. Take few minutes to find test cases containing a particular keyword using the Keywords link:
  trust -index &

Parallel calculation

The goal of this exercise is to introduce parallelism in the data file of the previous exercise.

- Go to the previous study (should be done) and after you had suppressed the **reprise** keyword and set **tinit** to 0 again in the *Obstacle.data* file, create two new files:
  cd Formation_TRUST/yourname/Obstacle
  mkdir PARA1
  cd PARA1
  cp ../Obstacle.data DEC_Obstacle.data
  cp ../Obstacle.data PAR_Obstacle.data
  cp ../Obstacle.geo .

- Edit the first file (*DEC_Obstacle.data*) to create the partition of the mesh.

Parallel calculation

- In this file, uncomment the block around the **Partition** keyword.
  - Here, the partitioning tool **Metis** is used. We cut in **nb_parts** blocks, here in 2.
  - The overlapping width **Larg_joint** between two parts of the partition should be defined according to the numerical scheme higher order, generally the convective scheme. Its value is generally 1 for a second-order scheme, and 2 for third- or fourth-order schemes such as Quick scheme.
  - In VEF, you should use **2** for **Larg_joint** except when partitioning a domain where only the conduction equation will be solved.
  - At least, the keyword **zones_name** is useful to define the name of the files containing the partitioned mesh and to write these files.
  - Notice the presence of the keyword **End** in the "Partition" block: the code will stop reading the data file at this line!
- Run the data file: **trust DEC_Obstacle**
- Check that the partitioned mesh files DOM_0000.Zones and DOM_0001.Zones are generated inside your working directory: **ls *.Zones**

Parallel calculation

- Now, edit the file PAR_Obstacle.data and comment the read of the mesh (using # tags of the 'BEGIN/END MESH' comments).
- Uncomment the **Scatter** keyword which will read the partitioned mesh.
- Now, run a parallel calculation with TRUST:
  trust PAR_Obstacle 2
- Post-processing step is identical in sequential or parallel modes. Probes are written into .son files and fields into .lata files. Run VisIt with:
  visit -o PAR_Obstacle.lata &
- Select the last time step and visualize the blocks (with Plots: Add → Subset → blocks) which represent the parts of the domain partition, then the velocity fields. You can also visualize a field only on a selected part (block) with the menu Control → Subset.
- Visualize probes at the end of the calculation using:
  trust -evol PAR_Obstacle &

Parallel calculation on a cluster

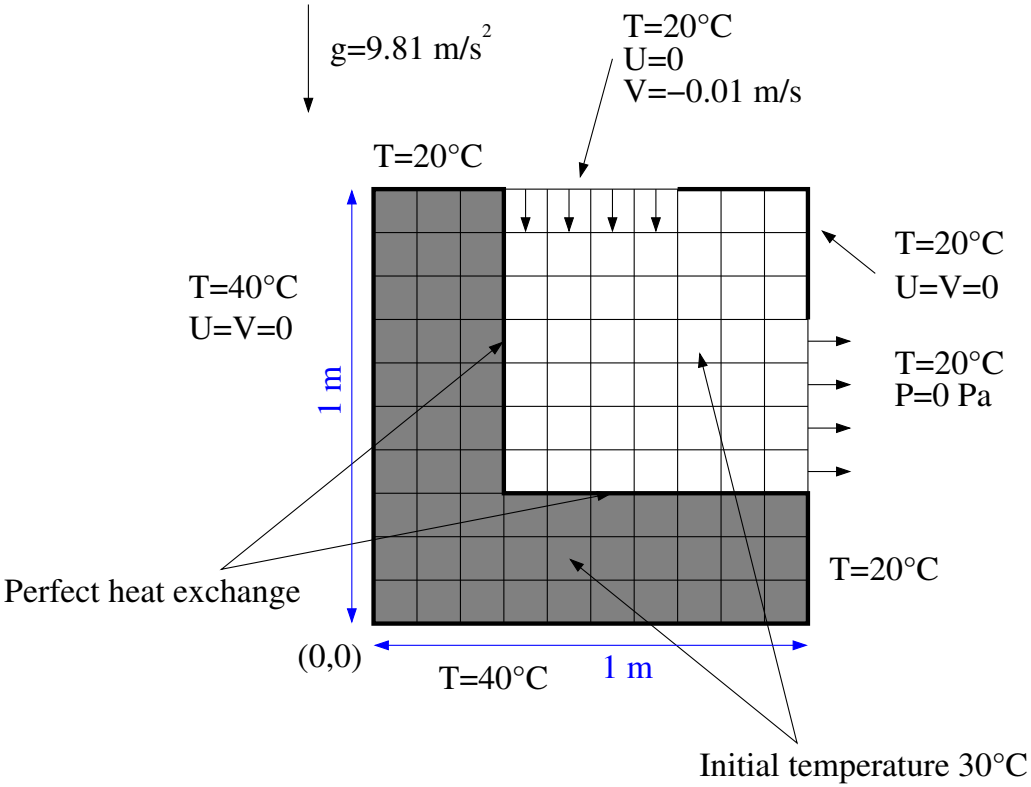
NB: On CEA Clusters, TRUST is already installed and the procedure of launching calculation is described below. Out of CEA, your cluster administrator should install and configure TRUST. In addition, submission files and procedure depend on the cluster itself and could be different from those presented below.

- Login to the CEA cluster "orcus" and initialize the TRUST environment:
  ssh -X yourlogin@orcusloginint1(.intra.cea.fr) or
  ssh -X yourlogin@orcusloginamd1(.intra.cea.fr)
  source /home/trust_trio-public/env_TRUST-1.9.4.sh
- Copy the study Obstacle:
  cd $SCRATCH
  mkdir -p Formation_TRUST/yourname
  cd Formation_TRUST/yourname
  trust -copy Obstacle
  cd Obstacle
- Open Obstacle.data and set the **format** to **lata** in the post-processing block.

Parallel calculation on a cluster

- Partition mesh and create a parallel data file with:
  trust -partition Obstacle
- For clusters, you have to create a submission file:
  trust -create_sub_file PAR_Obstacle 2
- Open the file sub_file and rename the job. Note that we will see only the first eight characters of the job name in the submitted jobs list.
- Submit the job with: **sbatch sub_file**
- Check job status with: **"squeue"** or **"squeue -u yourlogin"**
- To vizualize your results, use TurboVNC as described on Users training presentation.

Heat transfer (2D, VDF/VEF)



Fluid:

$$Pr = \frac{\mu C_p}{\lambda} = 1,$$
$$T_{ref} = 30^o C,$$
$$\mu = 2\ 10^{-3} kg.m^{-1}.s^{-1},$$
$$\rho = 2\ kg.m^{-3},$$
$$\lambda = 1.W.m^{-1}.K^{-1}$$
$$Cp = 500 J.kg^{-1}.K^{-1},$$
$$\beta = 1.10^{-4} K^{-1}$$

Solid:

$$\rho = 1000 kg.m^{-3}$$
$$\lambda = 250 W.m^{-1}.K^{-1}$$
$$Cp = 100 J.kg^{-1}.K^{-1}$$

Heat transfer (2D, VDF/VEF)

- Load TRUST environment as described on page 3
- Create a new study Coupling_VDF by copying the docond study:
  cd Formation_TRUST/yourname
  trust -copy docond
  mv docond Coupling_VDF
  cd Coupling_VDF
- Check the fluid and solid characteristics inside the docond.data file.
- This coupled problem is constituted by 2 domains of calculation with a mesh of 10x10 cells ($\Delta x = \Delta y = 0.1m$) created with 3 blocks.
- Now open your data file with "Plot2D" tool:
  trust -evol docond &
- Click on "Edit data".
- We want to modify the data file to have the 2 domains on a mesh of 40x40 cells ($\Delta x = \Delta y = 0.025m$).

Heat transfer (2D, VDF/VEF)

- Change the number of nodes for each block like this:
  First block (Cavite1): 4 11 $\rightarrow$ 13 41
  Second block (Cavite2): 8 4 $\rightarrow$ 29 13
  Third block (Cavite3): 8 8 $\rightarrow$ 29 29
- Change **"format lml"** to **"format lata"** into the two problems definition
- Click on "Save" and close the window.
- Run the calculation with "Start computation!" and check the evolution.
- Then post-process the temperature field with VisIt tool: "Visualization" button. A natural convection cell appears.
- Change the color tables for the temperature to have the same one on the 2 domains. Close VisIt.
- We are going to change the discretization of the test case: triangulate the domains with the keyword **Trianguler_H** (refer to the Reference Manual).

Heat transfer (2D, VDF/VEF)

- Then give an unstructured aspect to the 2 meshes using the following syntax:
  Transformer name_of_domain x*(1-0.5*y*y) y*(1+0.1*x*y)
- Substitute the discretization VDF (pressure nodes at the element center) to VEFPreP1B (pressure nodes at the element's center and nodes).
- Close the Plot2D tool.
- Run the calculation with:
  trust docond
- Open the IHM:
  trust -evol docond
- Select 'Ri=max_pb1$|dT/dt|$', 'Ri=max_pb2$|dT/dt|$', 'Ri=max_pb2$|dV/dt|$', 'residu=max$|Ri|$' with "Ctrl" button and click on 'Plot on same'.
- To see when convergence is reached, select a probe (for example temperature) and click on 'Plot'.

Heat transfer (2D, VDF/VEF)

- If the calculation is too long, open the docond.stop file, put a 1 instead the 0 and save. The calculation will stop after the current time step and make post-process.
- Post-process the results and compare the CPU performances with VDF discretization: the VEF calculation is running $\approx$ 10 times slower (because more pressure unknowns and shorter time steps). Check the docond.out file to see the time steps for each equation (click on "Edit .out" at the upper right corner of the GUI).
- Accelerate the calculation by impliciting the diffusive term of each equation with **diffusion_implicite** option in the explicit Euler scheme (check again the Generic Guide: **trust -doc &**).
- Run the calculation without any option:
  trust docond
- Now, use a fully implicit scheme (suppress **diffusion_implicite**), by substituting **Scheme_Euler_Explicit** by **Scheme_Euler_implicit** and adding the Implicit solver "**solveur implicite**".

Heat transfer (2D, VDF/VEF)

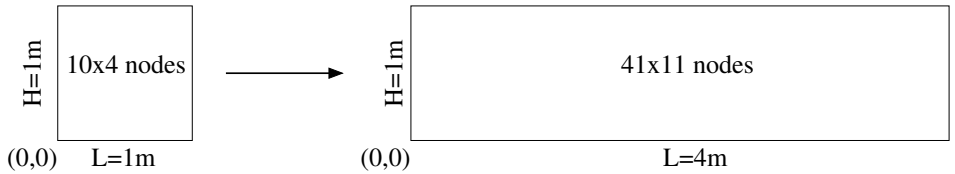
- Have a look at the Reference Manual for the **gmres** options and define, according to the advice given on it, a value for **facsec, facsec_max**.
- Your block will look like:
  Solveur Implicite { solveur gmres { diag seuil 1e-30 nb_it_max 5 impr } seuil_convergence_implicite 0.01 }
- Run the calculation:
  trust -evol docond &

Quasi Compressible flow (2D)

- Open a terminal and Load TRUST environment as described on page 3
- Copy the study **TP_Temp_QC_VEF** (it is a 2D simulation of helium gas flow from left to right between two heated walls) as follows:
  mkdir -p Formation_TRUST/yourname
  cd Formation_TRUST/yourname
  trust -copy TP_Temp_QC_VEF
  cd TP_Temp_QC_VEF
- Open the Generic Guide with (it will be useful to search for keywords in this exercise): **trust -doc &**
- Edit the data file with your favorite editor (**nedit** is recommended because it is configured to recognize the TRUST syntax):
  nedit TP_Temp_QC_VEF.data &
  or
  trust -evol TP_Temp_QC_VEF & and "Edit data" button.

Quasi Compressible flow (2D)

- Edit the data file in order to:
  - Modify the geometry and the mesh:



  - Add several probes (velocity, density, temperature) near the upper right corner of the geometry at location (x,y)=(4,1).
  - Add a probe **"segment"** (with 9 points) between the locations (x,y)=(4,0.05) and (x,y)=(4,0.95) for the temperature field.
  - Write the results on the **lata** format and change the **dt_post** period to 1s.
  - We are looking for the steady state, so suppress tmax keyword and change the **seuil_statio** ε value to 10 ($|dT/dt| < \varepsilon$ and $dt \sim 0.001s$ so $|dT| < 0.01$).
  - Add the keyword **impr** into the pressure solver to print its convergence.
  - If you use Plot2D tool, save and close the editor.
- Run the simulation with the TRUST command:
  trust -evol TP_Temp_QC_VEF &

Quasi Compressible flow (2D)

- Click on "Start computation!".
- Check mass flow rate (absolute and relative values) in the TP_Temp_QC_VEF.out file:
  nedit TP_Temp_QC_VEF.out &
  or look at the upper small window on the right of the PLOT2D tool.
- Once the calculation finishes, visualize the results by running **VisIt**:
  visit -o TP_Temp_QC_VEF.lata &
  or
  "Visualization" button on Plot2D tool.
  - Show the mesh (Plots: "Add → Mesh → dom → Draw").
  - Visualize the temperature field (Select the last Time with the slicer, then Plots: "Add → Pseudo Color → TEMPERATURE_SOM_dom → Draw").
  - Suppress or hide the mesh (Select "Mesh-dom" in the list of plots then "Delete" or "Hide/Show").
  - Visualize the velocity field (Add → Vector → VITESSE_SOM_dom → Draw).

Quasi Compressible flow (2D)

- Select the Zoom mode with the right button of the mouse (Mode → Zoom) then zoom by selecting an area on the plot. To un-zoom push "Ctrl" button and select an area with the left button or with the right button select "View → Reset view".
- Print your visualization (File → Set Save options → File type → Select a type → Save): a file named visit*** is created into your working directory.
- Add a second screen with "Window → Layout → 1x2".
- Plot a horizontal profile of temperature (Select the temperature field and thanks to the right button, select "Mode → Lineout", and define your profile with left button): the profile is shown on the second window.

- Substitute the time scheme by an implicit time scheme (like **scheme_euler_implicit**).
- Use the **implicite** solver and specify **facsec** and **facsec_max** parameters according to the advice given on the Reference Manual (search for the **scheme_euler_implicit** keyword). You can also see the instructions at the end of the Heat transfer VDF/VEF exercise on p.29.

Quasi Compressible flow (2D)

- Run the calculation with this time scheme using the PLOT2D tool or:
  trust TP_Temp_QC_VEF.data 1>TP_Temp_QC_VEF.out 2>TP_Temp_QC_VEF.err
- Edit the file containing information about dt (used time step), dt_stab (stability time step), **facsec** (dt=dt_stab*facsec) and residuals evolution for each equation:
  nedit TP_Temp_QC_VEF.dt_ev &
- If everything is OK, try to enhance the convergence speed of the implicit solver with the value of **seuil_convergence_implicite** keyword (look at the TP_Temp_QC_VEF.out file, if the number of iterations for GMRES is comprised between 3 and 5 then it is enough to converge quickly).
- In order to resume a calculation, you will have to change the **tinit** value within the data file (pick up the last saved time in the **.err** file) and insert into the data file, in the problem definition block, the following keywords:
  reprise binaire TP_Temp_QC_VEF_pb.sauv

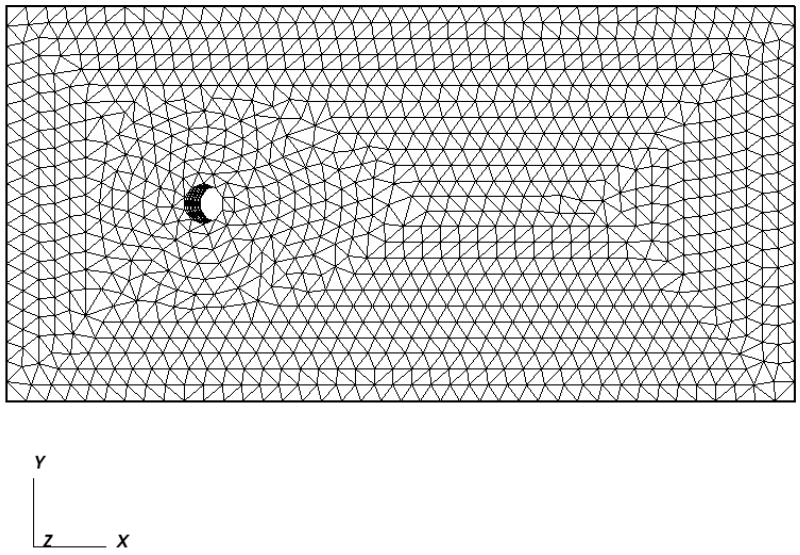
Quasi Compressible flow (2D)

- Then run the calculation with:
  trust TP_Temp_QC_VEF.data 1>TP_Temp_QC_VEF.out 2>TP_Temp_QC_VEF.err
  or
  trust -evol TP_Temp_QC_VEF.data & which automatically creates the .out file

Weakly Compressible

If you are interested in the comparison between Quasi Compressible and Weakly Compressible simulations, see the validation form:
$TRUST_ROOT/Validation/Rapports_automatiques/Verification/Verification_codage/QC_vs_WC

Periodic channel flow (3D)



Fluid: $Re = 2000$, $\rho = 2kg.m^{-3}$, $\mu = 0.01kg.m^{-1}.s^{-1}$, initial velocity $V0 = 1m/s$, periodic boundary condition following the Z-direction.

- Copy the study named **P1toP1Bulle** as explained on page 8. It simulates a 3D incompressible laminar flow ($Re = 2000$) with periodic boundary following the Z-direction only.
- Open the P1toP1Bulle.data file and use **RegroupeBord** keyword to merge Entree and Sortie boundaries into a single one named periox.

Periodic channel flow (3D)

- Modify boundary conditions to apply a periodic boundary on the new boundary.
- Change the velocity initial condition to $U_0 = (1, 0, 0)$.
- Set the option **diffusion_implicite** to 1 into the Euler scheme to implicit the diffusive term in the Navier-Stokes equations.
- You have now a 3D calculation with periodic boundary conditions on X- and Z-directions. Run the calculation for 30 time-steps (keyword **nb_pas_dt_max**).
- Have a look at the P1toP1Bulle_pb_Debit.out file, check the flow rate on the periox boundary. Why does it decrease?
- Add the **Canal_perio** source term in the Navier-Stokes equations of the data file and run again the calculation to check the flow rate evolution on 30 time steps.
- Look at pressure and viscous forces applied on the cylinder inside the .out files.

Periodic channel flow (3D)

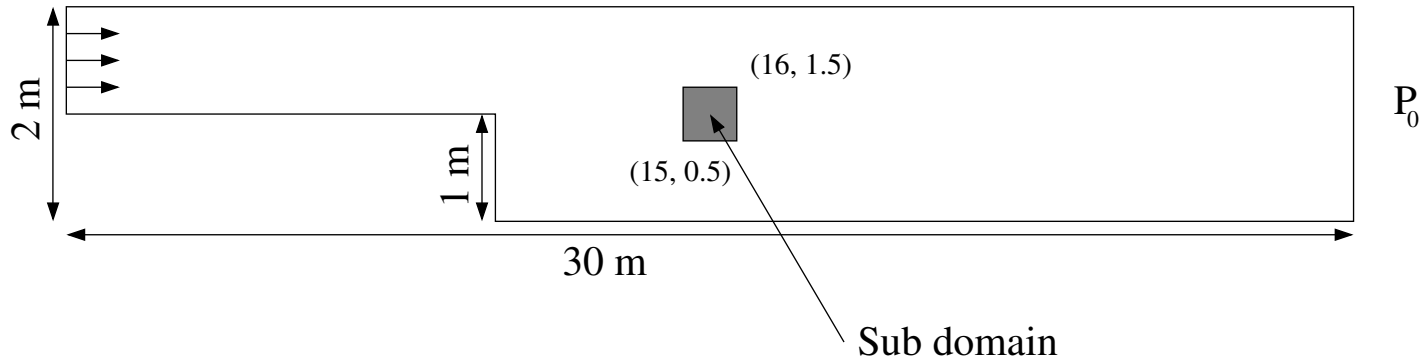
- Now, the calculation domain is a rotating channel according to Z direction with a constant velocity $\Omega = 1 rad/s$.
- Add the **Acceleration** source term in the Navier-Stokes equations. Suppress the **nb_pas_dt_max** keyword and set **tmax** to 100s.
- Add, if you wish, velocity or statistic calculation to the post-processing instructions.
- Run the calculation.
- You can create a uniformly refined mesh using, for instance, the keyword **Raffiner_Anisotrope**.
- Then improve the calculation speed on this mesh, you can use a coarse discretization **P1** (**Read dis** { **P1** }) with less pressure unknowns. On this latter, it runs 3 times faster than on P1Bulle discretization but it is less accurate: 8452 unknowns compared to 49221 unknowns.

Periodic channel flow (3D)

- Then restart the calculation with **VEFPreP1B** discretization by reading the velocity field with **Champ_fonc_reprise** keyword in the initial conditions for the velocity:
  vitesse champ_fonc_reprise P1toP1Bulle_pb.xyz pb vitesse last_time
  This will be useful to reach the quasi-stationary regime faster.
- You can also use implicit scheme (change the scheme to **Scheme_Euler_implicit** scheme and use an **Implicite** solver) only if you are looking for the stationary state.
  You can also see the instructions at the end of the Heat transfer VDF/VEF exercise on p. 29.

Constituents and turbulent flow

TrioCFD



Fluid: $\mu = 3.7\,10^{-5} kg.m^{-1}.s^{-1}$, $\rho = 2\,kg.m^{-3}$, $Re = \frac{U_0 H_{inlet} \rho}{\mu} = 54054$

Boundary conditions:

Inlet with imposed velocity: $U_0 = 1 m.s^{-1}$ and constant values of $k = 10^{-2}$ and $\varepsilon = 10^{-3}$ (dimensionless values)

Outlet with constant pressure: $P_0 = 0$ and constant values of $k = 0$ and $\varepsilon = 0$

Top and bottom walls: No-slip wall ($U = 0$) and k standard flux, ε null.

Constituents and turbulent flow

TrioCFD

- Initialize TrioCFD full environment to get access to TRUST&TrioCFD tests.
  - On CEA Saclay computers:
    source /home/trust_trio-public/full_env_TrioCFD-X.Y.Z.sh
  - On your own computer:
    source PathToTrioCFD/full_env_TrioCFD.sh

  echo $exec
  echo $project_directory
- Copy the study named **Marche** (TrioCFD) using: **triocfd -copy Marche** in the directory Formation_TRUST/yourname. It is also possible to use **"trust"** script since both commands have the same options and use the same $exec executable. This test case simulates a 2D incompressible turbulent flow in the above configuration using the k-ε model.
- We will add a source of constituent's diffusion, so copy the **Constituants** (2D incompressible laminar flow) study which uses constituents.

Constituents and turbulent flow

TrioCFD

- Edit your data file in the Marche directory. First, rename the problem in order to add concentration equations (look for the adequate keywords in the TrioCFD Reference Manual).
  triocfd -index then click on **Reference manual**
- Add 3 constituents of equal diffusivities ($\alpha = 1m/s$) after the fluid definition (in the problem block).
- Define the concentration equation into the problem (remember that concentrations will be a vector of 3 components) with correct initial ($C_1 = 0, C_2 = 0, C_3 = 0$) and boundary conditions.
- Use the Schmidt model to close the turbulence model in the concentration equation.
- Change the sources of the Navier-Stokes turbulence model to a **Source_Transport_K_Eps_aniso_concen { C1_eps 1.44 C2_eps 1.92 C3_eps 1. }** to fit with the new concentration equation.

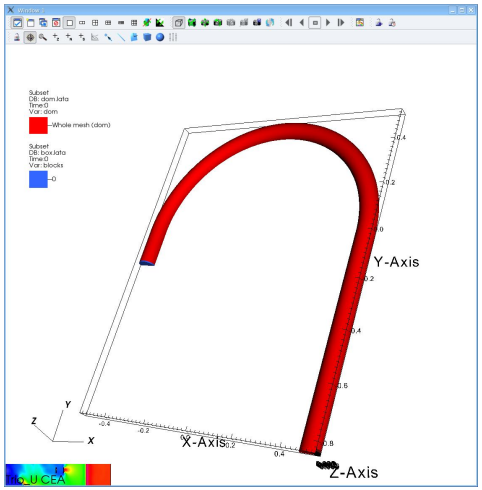
Constituents and turbulent flow

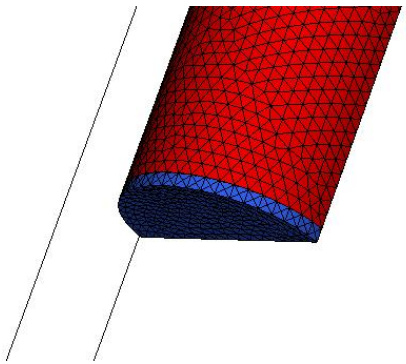
TrioCFD

- Add into the fluid definition, the volume expansion coefficient for the concentration: **beta_co** as a uniform field set to 0.
- You have also to add a gravity field which can be initialized to 0.
- Run the calculation to see if it is ok.
- Define a sub domain (in grey on the previous picture) with the keyword **Sous_Zone** (like in **PCR** data file (TRUST)).
- Add a source term for the second constituent only ($S_2 = 1m^{-3}$) applied on the sub domain thanks to the keyword **Champ_Uniforme_Morceaux**.
- Add **format lata** in the post-processing block.
- Add the keyword **concentration0, concentration1, concentration2** in the **fields** of the post-processing block to write the 3 concentrations into the .lata file.
- Run the calculation and check the results.

Turbulent flow in a curved pipe (3D)

TrioCFD





Goals:

- Use of a RANS or LES model.
- Use of a periodic box to initialize a fully developed turbulent flow.
- Use of the TrioCFD parallel capabilities.

Turbulent flow in a curved pipe (3D)

TrioCFD

- First initialize TrioCFD environment:
  - On CEA Saclay computers:
    source /home/trust_trio-public/full_env_TrioCFD-X.Y.Z.sh
  - On your own computer:
    source PathToTrioCFD/full_env_TrioCFD.sh

  echo $exec
  echo $project_directory

- Create a new directory and copy some data:
  mkdir -p Formation_TRUST/yourname/PeriodicBox
  cd Formation_TRUST/yourname/PeriodicBox
  cp $project_directory/share/Validation/Rapports_automatiques/
  Turbulence/RANS/PeriodicBox/src/* .
  This directory corresponds to an automated validation form. If you want to run it and generate the pdf report, see "Validation form" exercise on page 129, but be aware that this case needs huge computational effort!

Turbulent flow in a curved pipe (3D)

TrioCFD

- There are several files:
  - *BuildMeshes.data*: To build the meshes
  - *PeriodicBoxRANS.data*: To run the flow in the box with RANS model
  - *DomainFlowRANS.data*: To run the flow in the domain with inlet steady conditions from the box domain
  - *PeriodicBoxLES.data*: To run the flow in the box with LES model
  - *DomainFlowLES.data*: To run the flow in the domain with inlet unsteady conditions from the box domain
- First, edit and read the BuildMeshes.data file.
- If you wish to run a RANS simulation, open the PeriodicBoxRANS.data and DomainFlowRANS.data files.
- Or if you wish to run a LES simulation, open the PeriodicBoxLES.data and DomainFlowLES.data files.

Turbulent flow in a curved pipe (3D)

TrioCFD

- Then build the meshes:
  ./prepare
  triocfd BuildMeshes
  Notice that we use **"triocfd"** command lines, but it is also possible to use **"trust"** command because both scripts will use the variable $exec which is the path to the TrioCFD executable.
- Then, set the max number of time steps in the time scheme using **nb_pas_dt_max** to 100 in the files PeriodicBoxRANS and PeriodicBoxLES and run a 2-cores parallel calculation, to initialize the turbulent flow in the box:
  triocfd PeriodicBoxRANS 2
  triocfd PeriodicBoxLES 2
  (The full calculation takes approximately 1h in RANS and 10h in LES.)

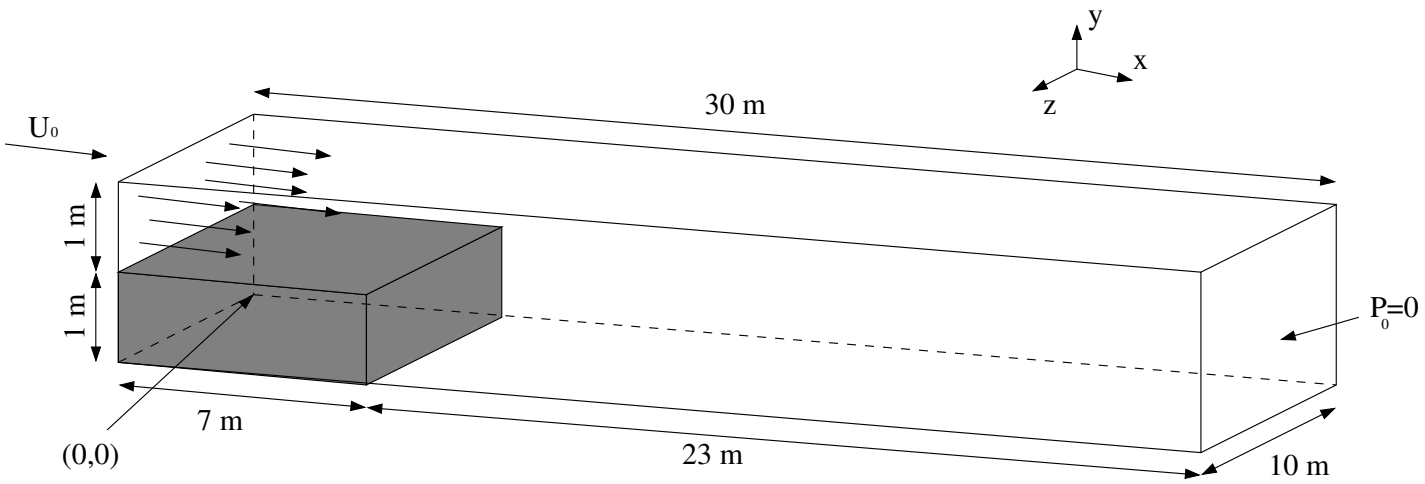
Turbulent flow in a curved pipe (3D)

TrioCFD

- Open the file PeriodicBoxRANS.dt_ev and PeriodicBoxLES.dt_ev to read the last time of the two calculations after 100 time steps.
- Once finished, open the file DomainFlowRANS.data and DomainFlowLES.data and change the maximal number of time steps to 10.
- You can see that these data files are constituted by 2 problems, one for the box and one for the domain.
  - We use the velocity and temperature fields of the last time step of the PeriodicBoxRANS (or LES) calculation as initial conditions for the pb_box problem with the keyword "Champ_fonc_reprise".
  - In addition, the velocity and temperature fields of the pb_box are used as boundary conditions for the pb_dom through the keyword "champ_front_recyclage".
- Run a 6-cores parallel calculation over the domain (it will stop by default after 10 times steps):
  triocfd DomainFlowRANS 6
  triocfd DomainFlowLES 6

Turbulent flow over a backward-facing step

TrioCFD



Meshing: 30x10x10 ($\Delta x = 1m, \Delta y = 0.2m, \Delta z = 1m$)
Fluid: $\mu = 5.10^{-5} kg.m^{-1}.s^{-1}, \rho = 2kg.m^{-3}$
Boundary conditions: with in entry $Re = \frac{U_0 H_{inlet} \rho}{\mu} = \frac{1 \times 1 \times 2}{5.10^{-5}} = 40000$
Inlet: $U_0 = 1m.s^{-1}$
Outlet: $P_0 = 0$

Turbulent flow over a backward-facing step

TrioCFD

- First initialize TrioCFD environment:
  - On CEA Saclay computers:
    source /home/trust_trio-public/env_TrioCFD-X.Y.Z.sh
  - On your own computer:
    source PathToTrioCFD/env_TrioCFD.sh

  echo $exec
  echo $project_directory
- Copy the study named Marche3D: **triocfd -copy Marche3D**
- Edit the data file and:
  - Note that we use a "Pb_Hydraulique_Turbulent" problem with "Navier_Stokes_Turbulent" equations and a "modele_turbulence" model.
  - Modify the fluid characteristics to perform a calculation at $Re = 50000$. For example, impose $\rho = 1 kg.m^{-3}$ and $\mu = 2.10^{-5} kg.m^{-1}.s^{-1}$.

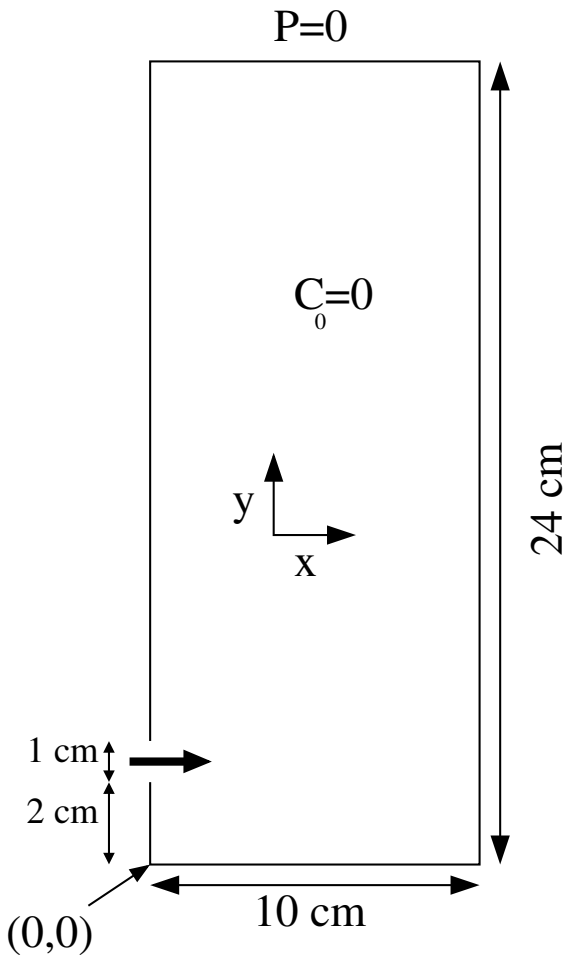
Turbulent flow over a backward-facing step

TrioCFD

- Continue editing the data file
  - Select the sub-grid Smagorinsky turbulence model with standard wall law instead of the "sous_maille" model (LES).
  - Select the Quick convection scheme.
  - Post-process of velocity, pressure, vorticity, turbulent viscosity at the nodes and elements.
- Run the calculation and post-process the main calculated fields.
  triocfd Marche3D
- Notice that we use **"triocfd"** command lines, but it is also possible to use **"trust"** script, since both scripts will use the variable $exec which is the path to the TrioCFD executable.
- Replace the sub-grid model by the standard k_eps model (RANS).
- Run the calculation and post-process of the velocity field to see the differences between the different turbulence models used.

Tank filling (2D, single-phase flow)

We want to simulate the following flow:



Fluid: Colored water
diffusion $D = 10^{-9} m^2.s^{-1}$,
$\rho = 1000 kg.m^{-3}$, $\mu = 10^{-3} kg.m^{-1}.s^{-1}$

Boundary conditions:
Inlet: Velocity: $(V_x, V_y) = (V(t), 0)$

with $V(t) = \begin{cases} 1 - (y - 0.025/0.005)^2 & , \ t \leq 0.5s \\ 0 & , \ t > 0.5s \end{cases}$

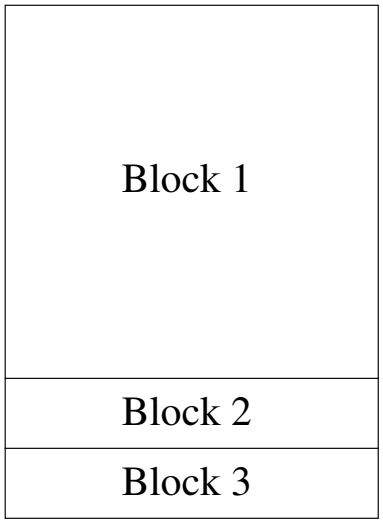
Concentration: $C = \begin{cases} 1 & , \ t \leq 0.5s \\ 0 & , \ t > 0.5s \end{cases}$

Outlet: Pressure $P = 0$
Wall: Velocity $V_x = 0$, $V_y = 0$

Initial conditions: Concentration $C_0 = 0$,
Velocity $V = 0$

Tank filling (2D, single-phase flow)

- Load TRUST environment as described on page 3
- Copy the study named **diagonale**. This test case deals with a 2D flow with Navier-Stokes and the equation for one constituent.
- Edit the data file and modify the fluid characteristics to the previous ones (μ, ρ, D).
- We want to modify the geometry of this problem to the previous picture. So we want to create 3 blocks like:



Tank filling (2D, single-phase flow)

- Create the corresponding mesh with 3 blocks (start with $dx = dy = 0.2cm$ which gives a total nodes number $Nx = 51$ and $Ny = 121$).
  - Create a first block "Block1" whose origin is (0, 0.03), $Nx = 51$, $Ny = 106$ (for $dx = dy = 0.2cm$), $L = 0.1m$, $H = 0.21m$. Name the wall boundaries Left1, Outlet(=Top1) and Right1. (Don't forget the comma between blocks definitions.)
  - Create the second block "Block2" whose origin is (0, 0.02), $Nx = 51$, $Ny = 6$ (for $dx = dy = 0.2cm$), $L = 0.1m$, $H = 0.01m$. Name the wall boundaries Inlet(=Left2) and Right2.
  - Create the third block "Block3" whose origin is (0, 0), $Nx = 51$, $Ny = 11$ (for $dx = dy = 0.2cm$), $L = 0.1m$, $H = 0.02m$. Name the wall boundaries Left3, Bottom3 and Right3.
- Define the boundary wall, using the keyword **"RegroupeBord"**.
- You could also use facteurs and symx, symy keywords to define a refined mesh near the walls.

Tank filling (2D, single-phase flow)

- In the data file, change the values in the time scheme to stop the calculation at 1 second, and modify **dt_min** and **dt_max** values to let TRUST compute time step.
- Change values for the gravity to $-9.81 m.s^{-2}$ following y-axis.
- Note that the **beta_co** keyword may be useful in order to have a Boussinesq coupling between momentum and concentration equations ($\beta C_0 g(C - C_0)$ source term added to the Navier-Stokes equations).
- Change the initial and boundary conditions for Navier-Stokes equations:
  - for the Outlet boundary, you have to impose $P = 0$,
  - for the Wall boundary, you have to impose $V_x = V_y = 0$ with **"paroi_fixe"** keyword.,
  - for the Inlet boundary, you have to impose $(V_x, V_y) = (V(t), 0)$ with $V(t) = \begin{cases} 1 - (y - 0.025/0.005)^2 & , \ t \leq 0.5s \\ 0 & , \ t > 0.5s \end{cases}$. You will use the **Champ_Front_Fonc_txyz** keyword for the velocity, to write something like:
    Champ_Front_Fonc_txyz 2 $(1 - ((y - 0.025)/0.005)^2) * (t < 0.5)$ 0.
    Note: Use $(t[0.5)$ syntax if you prefer $(t <= 0.5)$

Tank filling (2D, single-phase flow)

- Change the initial and boundary conditions for the constituent equation.
  - You will also use **Champ_Front_Fonc_txyz** field for the Inlet boundary condition for concentration.
  - For the Outlet, use the following keywords to insure the external concentration is 0: **Frontiere_ouverte C_ext Champ_front_uniforme 1 0.**
  - For the Wall, the keyword for impermeable boundary condition for concentration is **paroi**.
- Check you have high-order schemes (i.e. **"Quick"** scheme) used in both equations to reduce numerical diffusion.
- Notice you could have suppressed diffusion term in concentration equation rather than using a small diffusion coefficient with:
  Diffusion { **negligeable** }
- Add a concentration probe near the inlet (e.g.: at (0,0.025)).
- Add a velocity segment probe (with 5 points between (0,0.021) and (0,0.029)) at the inlet boundary to see the time evolution of these two quantities (period 0.01s).

Tank filling (2D, single-phase flow)

- Run the study and follow the time evolution with the probes:
  trust -evol diagonale &
  "Start computation!" button and "Plot" or "Plot on same" for probes.
- Check the flow rate in inlet boundary in the diagonale_pb_Debit.out file (plotted on the right of the PLOT2D window). You should find a value near $6.8\ 10^{-3} m^2.s^{-1}$.
- Use VisIt to post-process the results at t=0.2, t=0.4s and t=0.7s. VisIt has some interesting feature for this study. It can give concentration histogram to check the numerical diffusion in the concentration equation: Add $\rightarrow$ Histogram $\rightarrow$ CONCENTRATION_ELEM_dom. The volume of colored water (in m^3) is given by $Vol(t) = 6.66.10^{-3}t$ before $t = 0.5s$ and $Vol(t) = 3.33.10^{-3}$ after.

Tank filling (2D, single-phase flow)

→ VEF

- Copy diagonale.data to diagonale_VEF.data.
- Triangulate your mesh (**trianguler** keyword).
- In this new file, change the discretization (**VEFPreP1B** instead of **VDF**).
- Use **muscl** instead of **quick** scheme.
- And you can switch **GCP** solver by **Cholesky** solver of the Petsc library (direct method which may need large amount of RAM memory) to increase the speed resolution of the pressure linear system:
  GCP { **precond ssor** { **omega 1.5** } **seuil 1.e-6** } → **Petsc Cholesky** { }
- Run the calculation. You must have an error, and TRUST stop the calculation.

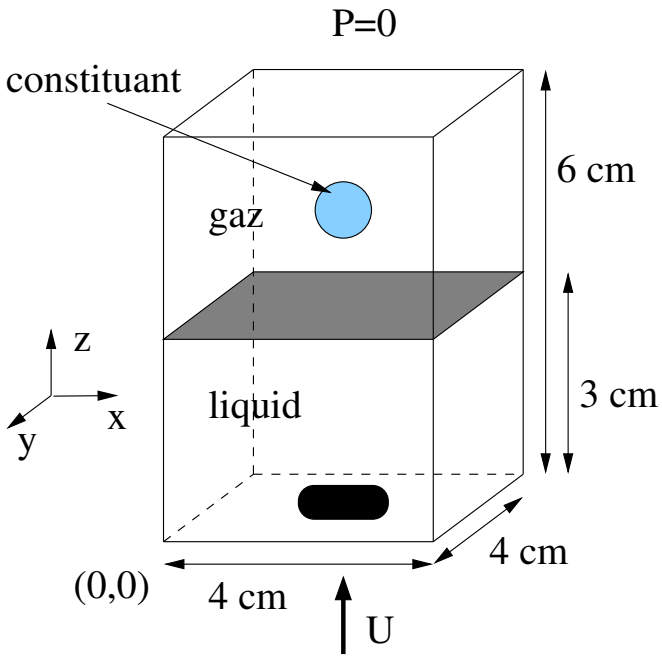
Tank filling (2D, single-phase flow)

→ VEF

- As TRUST indicates, to avoid this problem, you can:
  - change the **trianguler** keyword to **trianguler_h**,
  - or use the **VerifierCoin** keyword. For this, after this first error you must find a "diagonale_VEF.decoupage_som" file in your directory, so you can use it by adding:
    VerifierCoin dom { read_file diagonale_VEF.decoupage_som }
    just after "trianguler dom". This will subdivides inconsistent 2D/3D cells used with VEFPreP1B discretization (cf Reference Manual).
- Run the calculation and compare the results between **VDF/quick** and **VEFPreP1B/muscl** which must take much more time!

Tank filling (3D, two-phase flow)

TrioCFD



Liquid: $\rho = 1000 kg.m^{-3}$,
$\mu = 2,82.10^{-4} kg.m^{-1}.s^{-1}$,
$\sigma = 0.05 N.m^{-1}$, $D = 10^{-6} m^2.s^{-1}$

Gas: $\rho = 100 kg.m^{-3}$,
$\mu = 2,82.10^{-4} kg.m^{-1}.s^{-1}$

Boundary conditions:
Up: Free outlet, Wall: $V = 0$
Down: $V(x, y, z) = (0, 0, 10^{-3} m.s^{-1})$

Initial conditions: $V = 0$,
$C = e^{(-((x-0.02)^2+(y-0.02)^2+(z-0.03)^2)/0.03^2)}$

N.B.: The interface between the air and the gas is a parabolic function.

Tank filling (3D, two-phase flow)

TrioCFD

- First initialize TrioCFD environment:
  - On CEA Saclay computers:
    source /home/trust_trio-public/env_TrioCFD-X.Y.Z.sh
  - On your own computer:
    source PathToTrioCFD/env_TrioCFD.sh
- Copy the study named **FTD_all_VDF**:
  triocfd -copy FTD_all_VDF
- This test case deals with a 3D two-phase flow in a tank with one initial interface between liquid and gas, a droplet, and a rotating solid in the liquid. The Discontinuous Front Tracking method is used with a 3D structured mesh.

Tank filling (3D, two-phase flow)

TrioCFD

- Notice that:
  - 2D Discontinuous Front Tracking method has not been intensively tested yet.
  - the type of the problem: **Probleme_FT_disc_gen** in the data file. This refers to the Discontinuous Front Tracking method.
  - the keyword **modele_turbulence**. Navier-Stokes equations of the Discontinuous Front Tracking problem needs the read of this keyword even if the flow is laminar. In this case, use the **nul** keyword just after **modele_turbulence**. Else, specify the turbulence model to use.

Tank filling (3D, two-phase flow)

TrioCFD

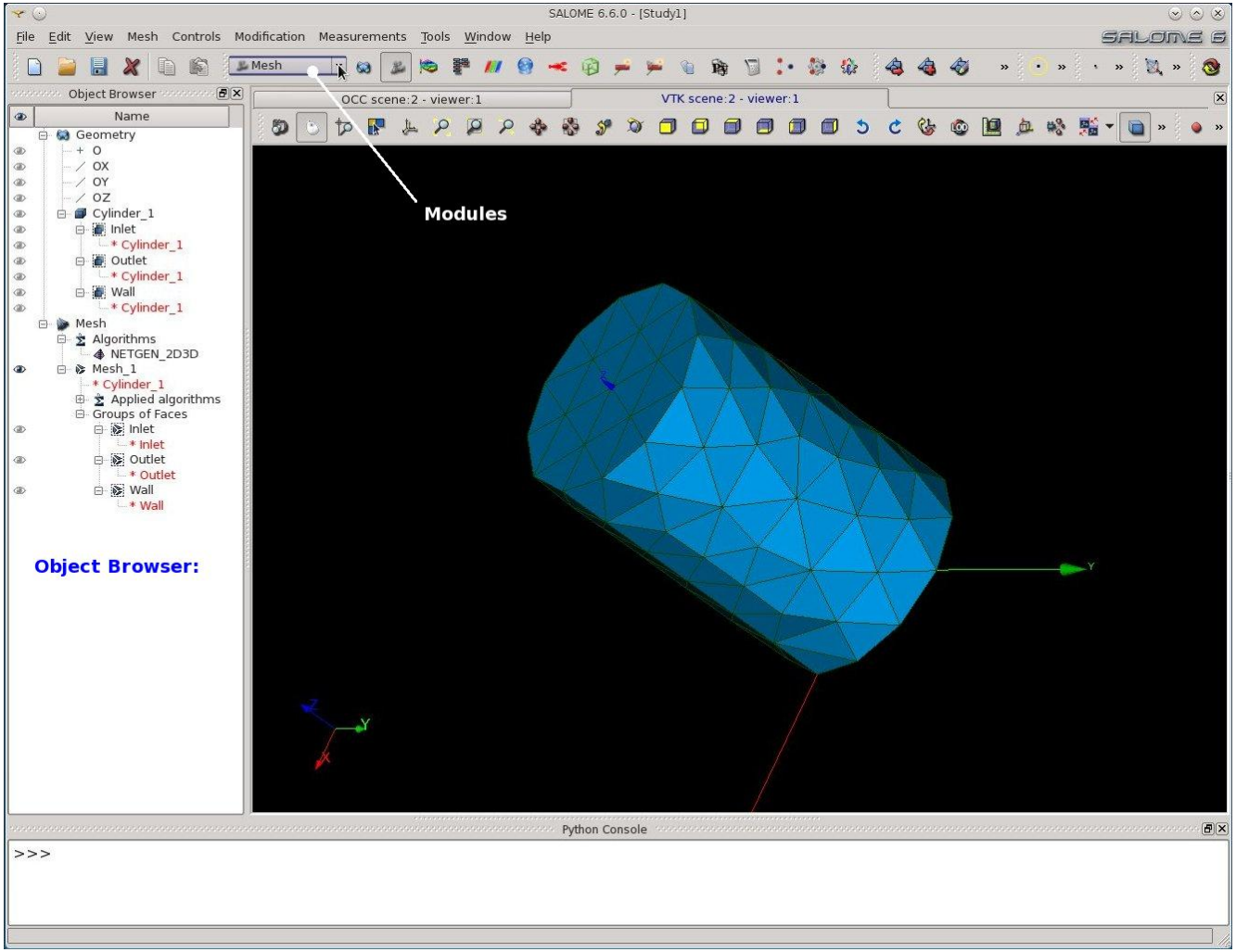
- Increase the height of the tank (from 0.06 to 0.12).
- Add a second drop above the first one, at $z = 0.08$ (keywords **ajout_phase0** could be useful to add other interfaces, cf Reference Manual for **ajout_phase0/ajout_phase1** keywords). It is possible to access to the reference manual by typing **triocfd -index**. Don't forget the comma between the two definition of the drops.
- Change the **dt_post** period of the 3 post-processing blocks (0.05 to 0.01). The first one (add **format lata**) is the classical block for post-processing probes and fields. Here, we want to see the concentration field and the **"indicatrice_interf"** field. Value of this field is 0 for liquid and 1 for gas, so the interface is located at "indicatrice" value 0.5.
- Change the interpolation location of **indicatrice_interf** and the **concentration** fields in the first post-processing block, by adding the keyword **elem** just after the fields: the values in the post-processing tool will be plotted at the center of each element of the mesh.

Tank filling (3D, two-phase flow)

TrioCFD

- The second post-processing block is the new syntax to post-process interfaces moving meshes. You can visualize it with **VisIt**.
- On each interface, you can plot several fields i.e.: curvature with **courbure** keyword and velocity interface with **vitesse** keyword, **pe** field is for debugging purpose, it is useless here, you can suppress it) on several locations (on nodes with **sommets** keyword, on cells with **elements** keyword).
- Run the calculation. Follow the time step evolution by having a look at the **dt_ev** file. It contains on each line the physical time, the time step, security factor and residuals.
- Post-process to visualize the interface and the concentration field.
- You can increase the number of cells to have a finest simulation or also change to VEF discretization.

Salomé to create a 3D VEF mesh: Cylinder



Cylinder

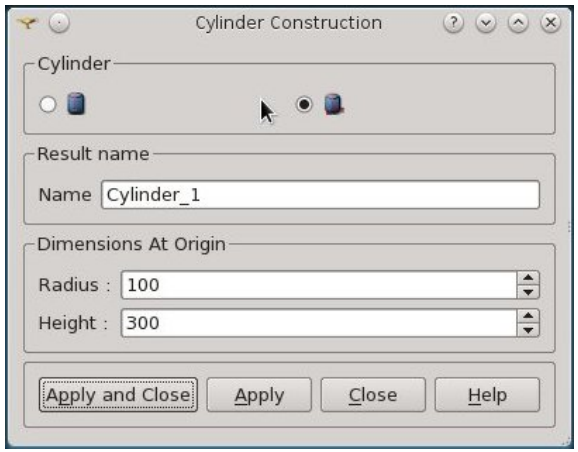
Create a geometry

- Create a new folder:
  ```
  $ mkdir -p Formation_TRUST/yourname/salome/exo1
  $ cd Formation_TRUST/yourname/salome/exo1
  ```
- Launch Salomé (we suppose it is installed in $PathToSalome):
  ```
  $ $PathToSalome/salome &
  ```
- Create a new study: File → New
- Select the Geometry module into the SALOME drop-down menu (contains all the modules).
- Save your study in hdf format (Salome format) frequently.
- Create a first geometry with: New Entity → Primitives → Cylinder

Cylinder

Create a geometry

- Specify Radius $R = 100$ and Height $H = 300$ for the cylinder (the default values). Then Apply and Close.

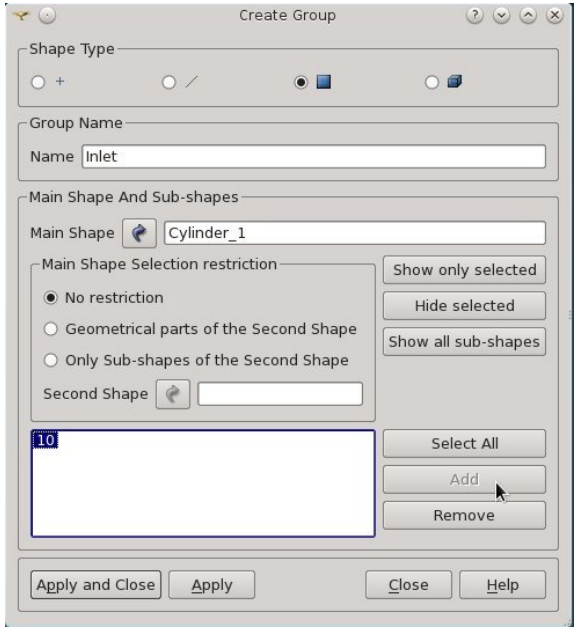


- Rotate, zoom, move the geometry by switching to "Interaction style switch": Mouse icon.
- Create groups for the geometry to define the top, the bottom and the lateral parts of the cylinder: New Entity → Group → Create Group
- Select the good Shape Type (→ □ surface).

Cylinder

Create a geometry

- Give a Group Name for the top: "Inlet".
- Click on the arrow button of the Main Shape field and select the "Cylinder_1" in the "Object browser" or in the visualization window.
- Select the shape defining the top of the cylinder on the visualization window then Add → Apply.
- Select the shape defining the part on the window then Add → Apply.



Cylinder

Create a geometry

- Do the same for the two other parts:
  - For the lateral: "Wall"
  - For the bottom: "Outlet" (you can rotate the cylinder to click on the bottom).
- Close the window once the 3 groups has been created. Check that they appear in the Object Browser (by clicking on the "▷" in front of the "Cylinder_1" object).

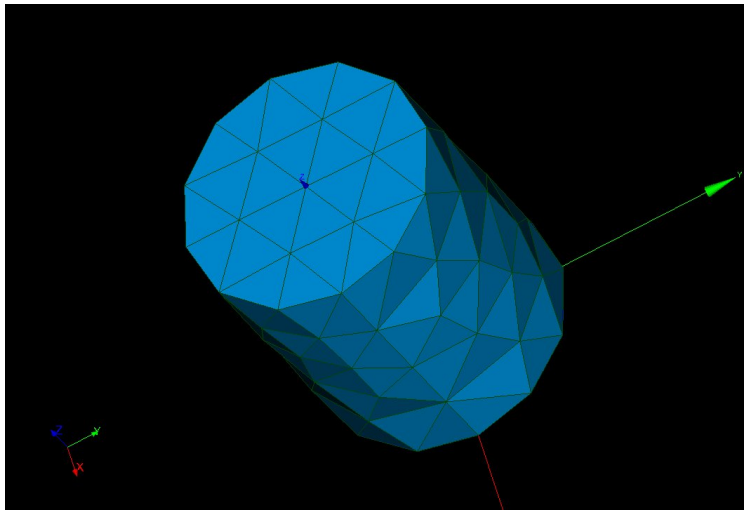
Create a mesh

- Now, switch to the Mesh module in the SALOME drop-down menu.
- Select the Cylinder_1 in the Object Browser and Right Click → 'Show' to visualize the geometry or click on the 'eye' next to the Cylindre_1 object.
- Create a mesh with: Mesh → Create Mesh
- Select the Geometry used for the mesh if not selected by clicking on the Cylinder_1 object in the Object Browser.

Cylinder

Create a mesh

- Choose Netgen 1D-2D-3D algorithm and click on "Apply and Close".
- Select the object Mesh_1 in the Object Browser and Right Click → Compute (or Mesh → Compute).
- A tabular must appear with the number of triangles, quadrangles... Click on "Close".
- Hide the geometry by selecting the Cylinder_1 in the Object Browser and Right Click → Hide (or click on the eye).



Cylinder

Export your mesh in MED format

- Check that the 3 boundaries have automatically been added in the "Group of Faces" of the **Mesh_1** object in the Object Browser.
- Export your mesh with the MED format:
  Select the Mesh_1 object then Right Click → Export → MED file (or File → Export → MED file).

Cylinder

Read your mesh with TRUST

- Now build a data file named dom.data for TRUST:

```
dimension 3
domaine dom
Read_MED { domain dom file Mesh_1.med }
Postraiter_domaine { domaine dom fichier mesh format lata }
```

- Open a new terminal then load TRUST environment as described on page 3
- Run the data file and post-process the mesh with VisIt:
  trust dom
  visit -o mesh.lata

  Warning: The more common error is to forget to define the boundaries with the groups for the mesh (and hence for Geometry). The error in TRUST is printed and detected during the discretization where all the faces of the mesh (in particular the boundary faces) are built.

Cylinder

Refine your mesh and use viscous layers

Goal: Improve the mesh for TRUST near the wall by using viscous layers.

- Create a new mesh named "Refined_mesh" with: Mesh → Create Mesh
- Select the Cylinder_1 geometry in the Object Browser.
- Select the "Netgen 3D" or "MG-Tetra" 3D algorithm.
- Click on the wheel of "Add. Hypothesis" → "Viscous Layers" with:
  - Total thickness: 30
  - Number of layers: 3
  - Stretch factor: 1.1
  - Add to "Faces without layers" the 2 geometry groups "Inlet" and "Outlet" of Cylinder_1 object (select or unselect the mouse icon).
  - Click OK
- Add a 2D algorithm: "Netgen 1D-2D" or "MG-CADSurf".

Cylinder

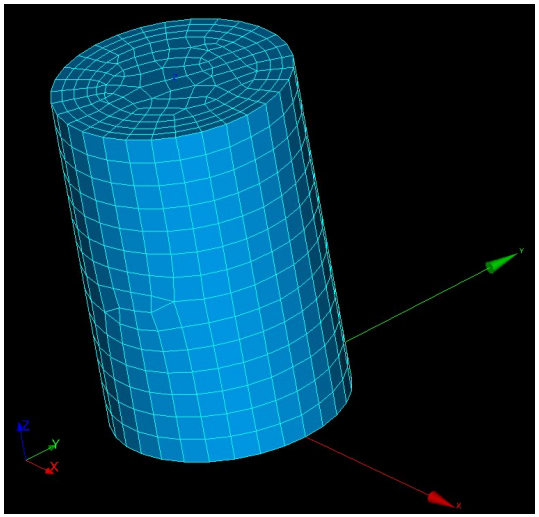
Refine your mesh and use viscous layers

- Click on the wheel of "Hypothesis" → "Netgen 2D parameters" or "MG-CADSurf parameters":
  - For "Netgen 2D parameters":
    - Change "Fineness" from "Moderate" to "Very Fine".
    - Click OK.
  - For "MG-CADSurf parameters", change "User size" to 20. Click OK.
- "Apply and Close" the close mesh window.
- Select the Refined_Mesh object in the Object Browser and Right click → Compute

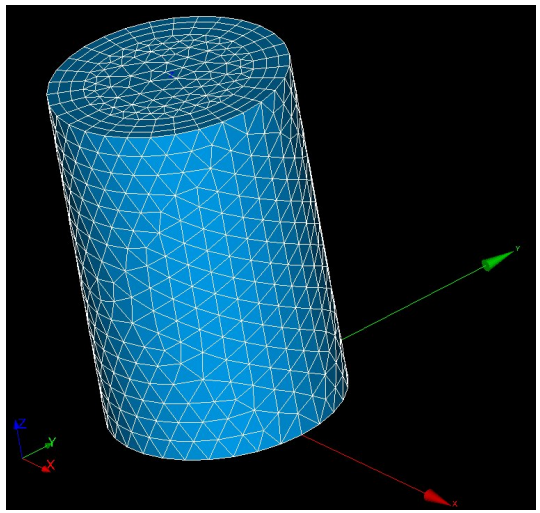
Cylinder

Refine your mesh and use viscous layers

- You should have a refined mesh with a mix of tetra, hexa, pyramid, prism elements for Netgen algorithms, and a mix of tetra and prisms for MG algorithms:



Netgen



MG

- As TRUST accepted only tetras elements, you can quickly tetraedrize:
  - Select the Refined_Mesh in the Object Browser.
  - "Modification" $\rightarrow$ "Split Volumes" and select "Tetrahedron".
  - Don't change the parameters, and click "Apply and Close".

Cylinder

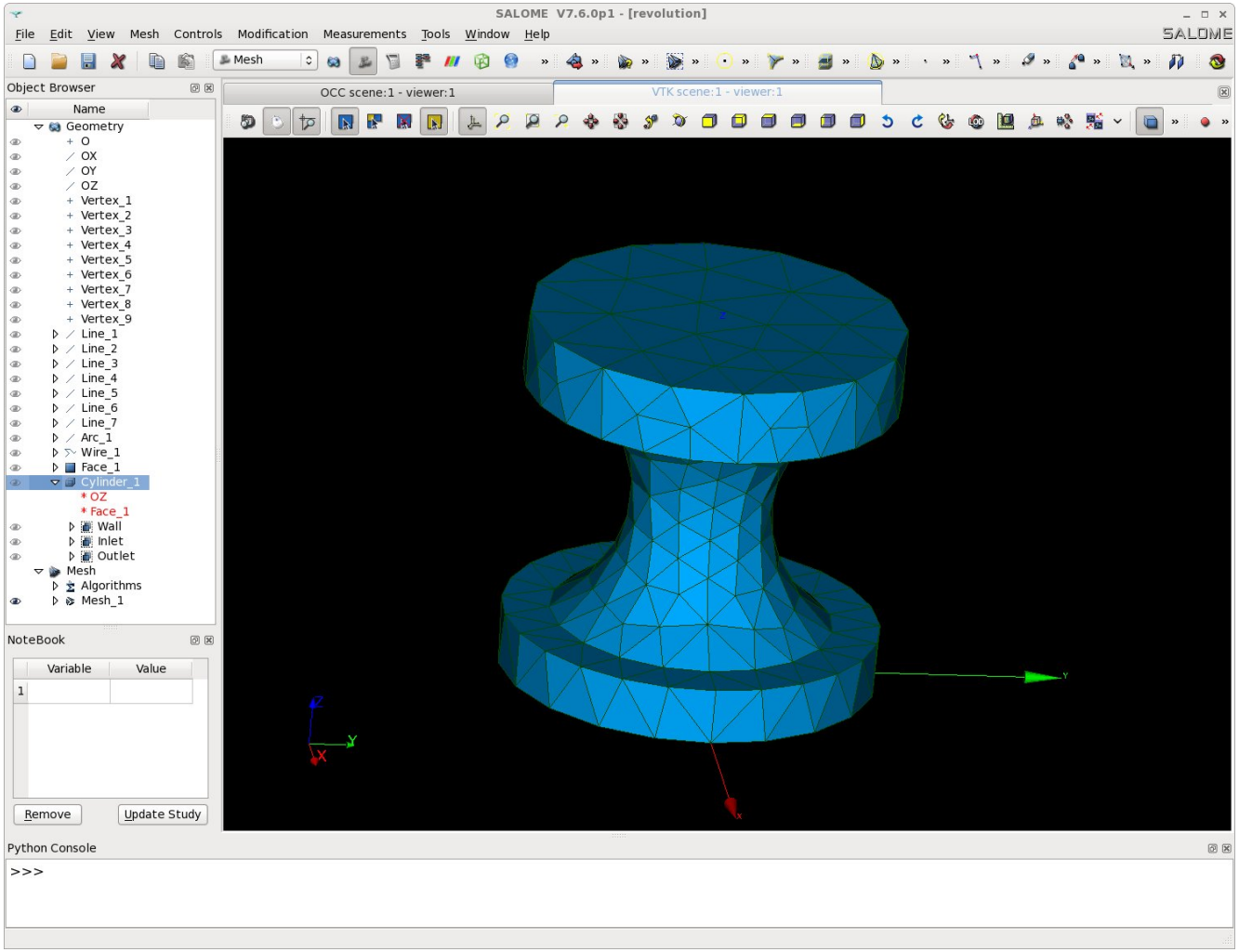
Refine your mesh and use viscous layers

- Check that the 3 boundaries have automatically been added in the "Group of Faces" of the **Refined_Mesh** object in the Object Browser.
- Export the mesh:
  - Select the Refined_mesh, Right click → Export → MED file.
  - Save into a Refined_Mesh.med file.
- Save your work in hdf format ("File" → "Save/Save As..."), and in python format with "File" → "Dump Study..."

Run with TRUST

- Edit your datafile or create a new one to read and visualize your refined mesh.
- **N.B.**: The solutions of the exercise (mesh.py file for the first mesh and prism.py file for the second mesh) are located here: $TRUST_ROOT/doc/TRUST/exercices/salome.

Salomé to create a 3D VEF mesh: Revolution



Salomé to create a 3D VEF mesh: Revolution

Create a geometry

- Create a directory and run Salomé (we suppose it is installed on $PathToSalome):

```
$ mkdir -p Formation_TRUST/yourname/salome/exo2
$ cd Formation_TRUST/yourname/salome/exo2
$ $PathToSalome/salome &
```

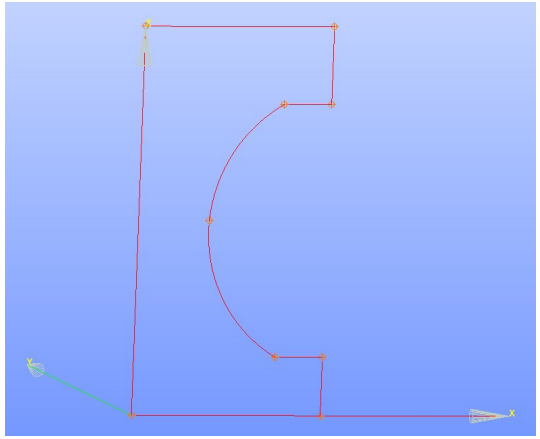
- Create a new study: File → New.
- Select the Geometry module into the SALOME drop-down menu.
- Create points: New Entity → Basic → Point

Vertex_1 (0,0,0)	Vertex_2 (1,0,0)	Vertex_3 (1,0,0.3)
Vertex_4 (0.75,0,0.3)	Vertex_5 (0.375,0,1)	Vertex_6 (0.75,0,1.6)
Vertex_7 (1,0,1.6)	Vertex_8 (1,0,2)	Vertex_9 (0,0,2)

Then "Apply and Close"

Revolution

Create a geometry



- Create edges:
  - New Entity → Basic → Line
    - Line_1 with Vertex_1 and Vertex_2
    - Line_2 with Vertex_2 and Vertex_3
    - Line_3 with Vertex_3 and Vertex_4
    - Line_4 with **Vertex_6** and **Vertex_7**
    - Line_5 with Vertex_7 and Vertex_8
    - Line_6 with Vertex_8 and Vertex_9
    - Line_7 with Vertex_9 and Vertex_1
    - Then Apply and Close.

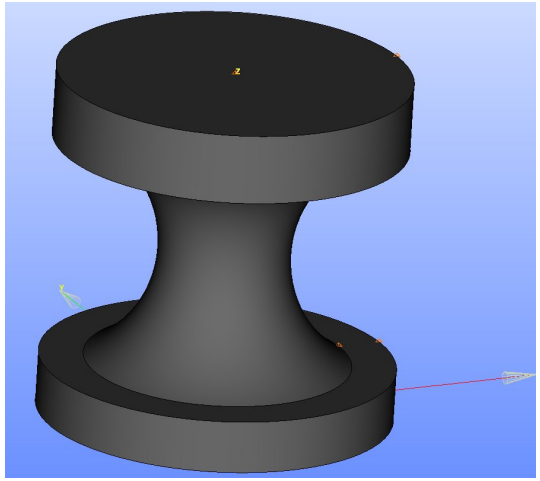
Revolution

Create a geometry

- Create edges:
  - New Entity → Basic → Arc
    - Arc_1 with Vertex_4, Vertex_5 and Vertex_6.
    - Then Apply and Close.
- Create a wire: New Entity → Build → Wire
  - Wire_1 on edges with Line_1,... , Line_7 and Arc_1 (with "Ctrl" button).
  - Then Apply and Close.
- Create a face: New Entity → Build → Face.
  - Face_1 with Wire_1 and "Apply and Close".
- Create a revolution cylinder: New Entity → Generation → Revolution.
  - named Cylinder_1,
  - with Face_1 in Objects,
  - click on the arrow button next "Axis" and select OZ in the Object Browser,
  - set the angle to 360^o and "Apply and Close".

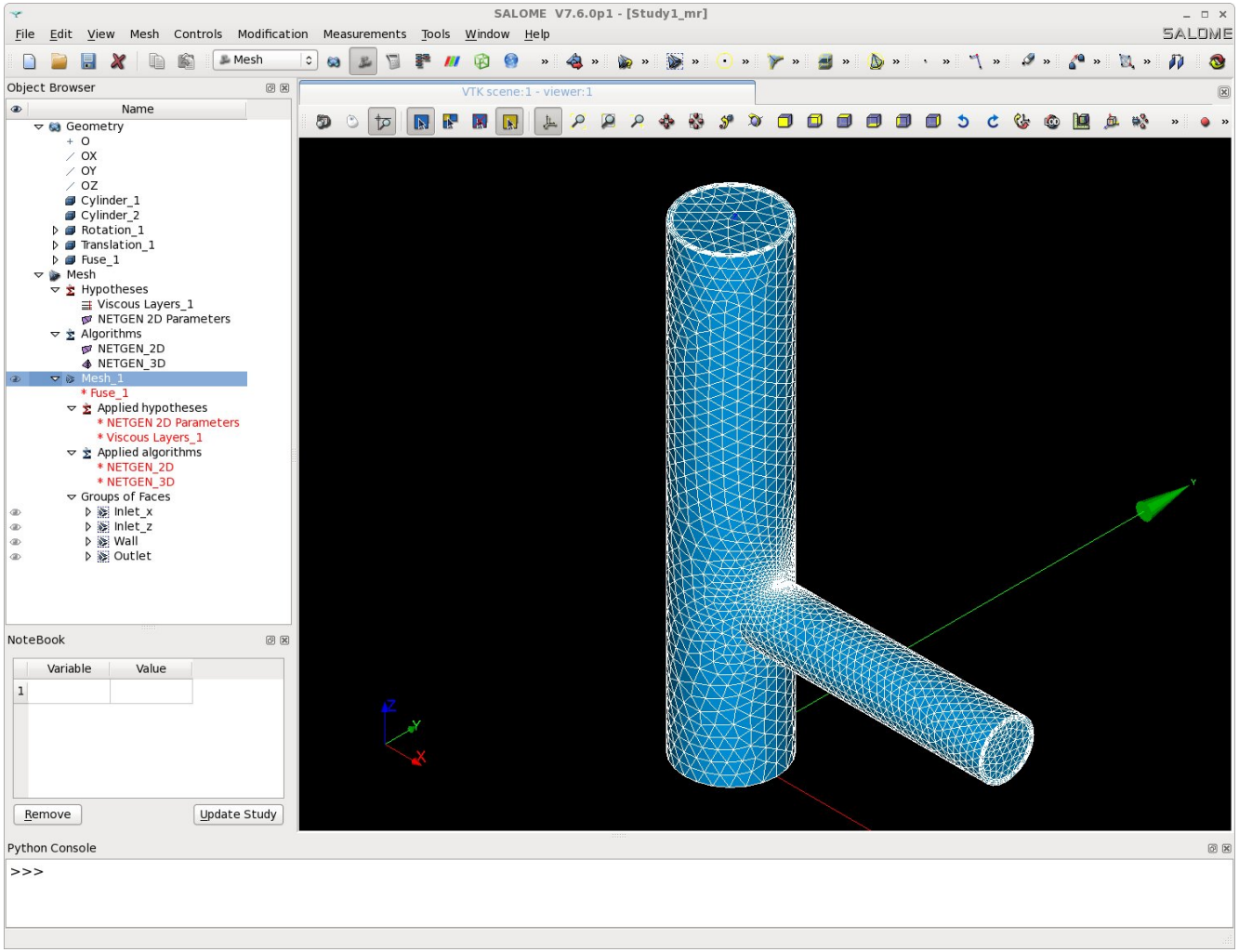
Revolution

Create a geometry



- Create groups for the geometry to define the top, the bottom and the lateral parts of the cylinder: New Entity → Group → Create Group.
- Save your study in hdf format ("File" → "Save/Save As..."), and in python format with "File" → "Dump Study..."
- Now you can create the mesh in the same way than page 101.
- **N.B.**: You can find the solutions of this exercise (revolution.py) in $TRUST_ROOT/doc/TRUST/exercices/salome.

Salomé to create a 3D VEF mesh: T-shape



Salomé to create a 3D VEF mesh: T-shape

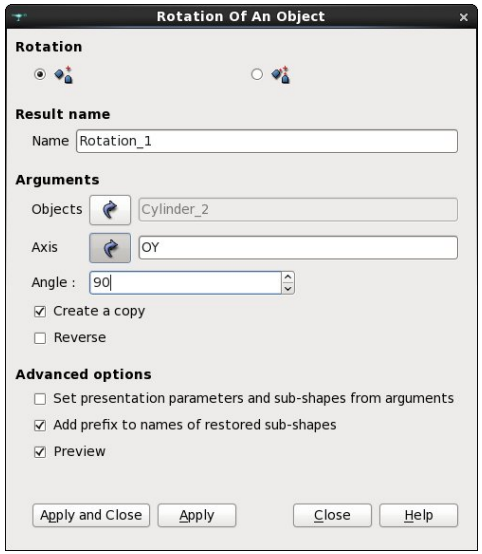
Create a geometry

- Create a directory and run Salomé (we suppose it is installed on $PathToSalome):
  ```
  $ mkdir -p Formation_TRUST/yourname/salome/exo3
  $ cd Formation_TRUST/yourname/salome/exo3
  $ $PathToSalome/salome &
  ```
- Create a new study: File → New.
- Select the Geometry module into the SALOME drop-down menu.
- Create two cylinders: New Entity → Primitives → Cylinders:
  Cylinder_1: radius 0.5, height 5. Then "Apply".
  Cylinder_2: radius 0.3, height 3. Then "Apply and Close".
- Save your study in hdf format (Salome format) frequently.

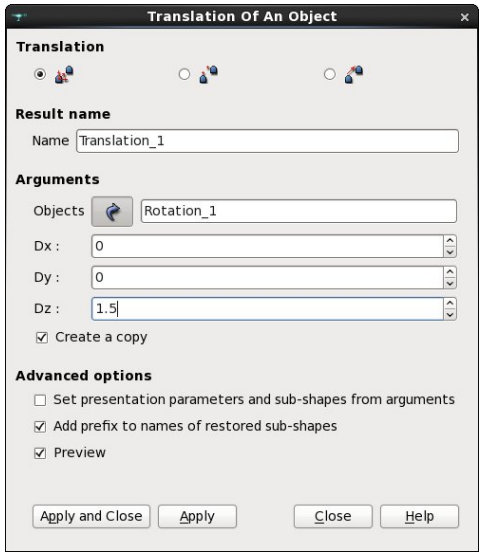
T-shape

Create a geometry

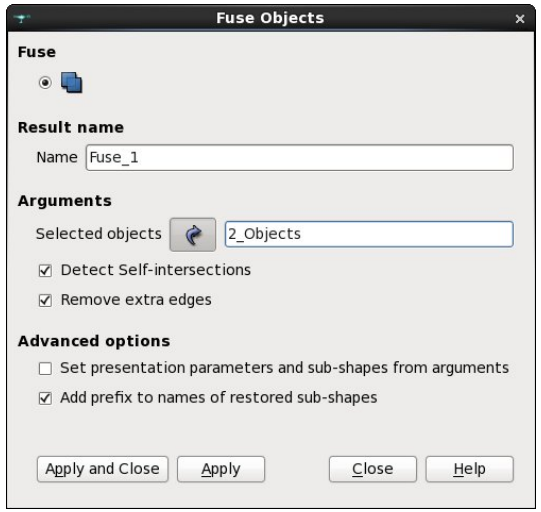
- Rotate Cylinder_2: Operations → Transformation → Rotation
  Name:Rotation_1, Object: Cylinder_2, Axis: 'OY', Angle: 90°
  Then "Apply and Close".
- Translate Rotation_1: Operations → Transformation → Translation
  Name: Translation_1, Object: Rotation_1, Dx=Dy=0, Dz=1.5
  Then "Apply and Close".



Rotation



Translation

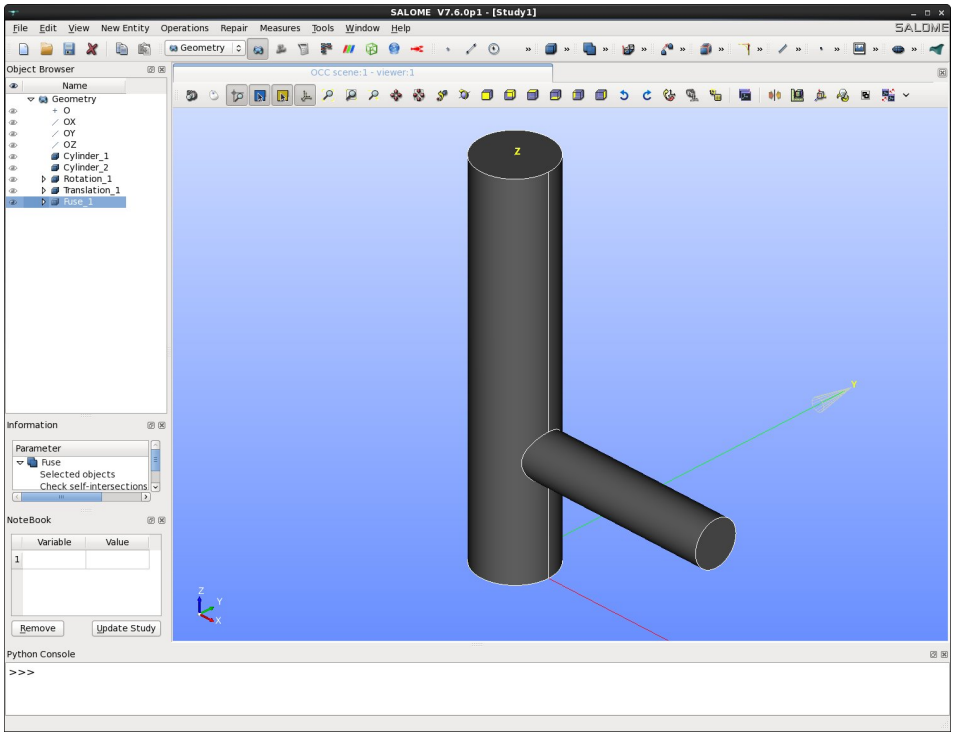


Fuse

T-shape

Create a geometry

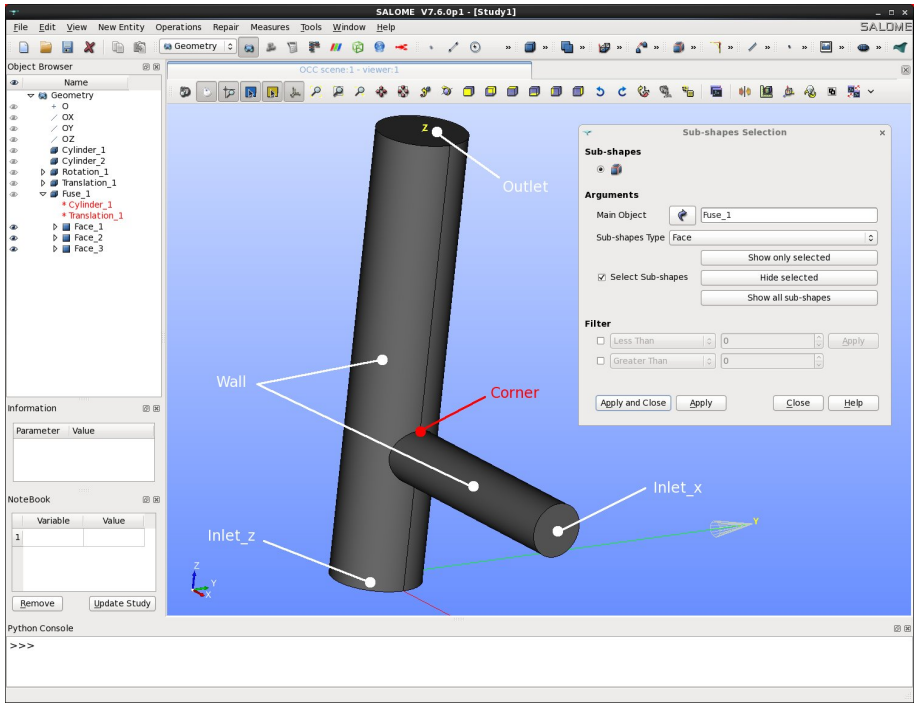
- Fuse Cylinder_1 and Translation_1: Operations → Boolean → Fuse
  Name: Fuse_1, Selected Objects: 2_Objects (use "Ctrl" button to select Cylinder_1 and Translation_1 in the Object Browser).
  Then "Apply and Close".



T-shape

Create a geometry

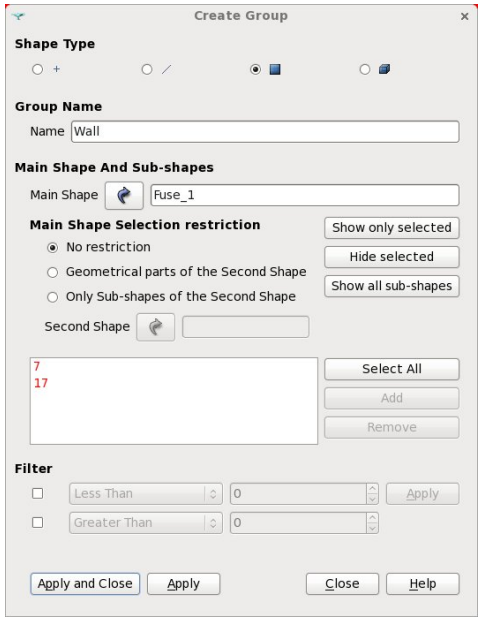
- We are now going to create the boundaries: New Entity → Explode
  - Main Object: Fuse_1, Sub-shape type: Face, select "Select sub-shape" and click on the surface Outlet and "Apply".
  - This will create a face named "Face_1" in the Fuse_1 object (click on the "▶"), rename it "Outlet" (by right-clicking and "rename").



T-shape

Create a geometry

- Do the same for "Inlet_x" and "Inlet_z".
- We are now going to create the boundary Wall: New Entity → Group → Create group:
  Shape Type: surface, Name: Wall, Main Shape: Fuse_1.
  Click on the surface of the Cylinder_1 then "Add", click on the surface of Translation_1 then "Add" and "Apply and Close".



T-shape

Create a geometry

- We are now going to create the point "Corner": New Entity → Explode
  - Main Object: Fuse_1, Sub-shape type: Vertex, select "Select sub-shape" and click on the chosen point and "Apply and Close".
  - This will create a vertex name "Vertex_1", rename it "Corner" (by right-clicking and "rename").

Create a mesh

- Now, switch to the Mesh module in the SALOME drop-down menu.
- Select the Fuse_1 in the Object Browser and Right Click → 'Show' to visualize the geometry or click on the 'eye' next to the Fuse_1 object.
- Create a mesh with: Mesh → Create Mesh.
- Select the Geometry used for the mesh if not selected by clicking on the Fuse_1 object in the Object Browser.

T-shape

Create a mesh

- Choose "Netgen 3D" for 3D algorithm.
- Click on the wheel of "Add. Hypothesis" → "Viscous Layers" and set:
  - Total thickness: 0.05
  - Number of layers: 3
  - Stretch factor: 1.1
  - Extrusion method: Node Offset
  - Add to "Faces with layers (Wall)" the geometry group "Wall" of Fuse_1 object in the Object Browser (select or unselect the mouse icon). Click on "Add".
  - Click "OK".
- Choose "Netgen 1D-2D" for 2D algorithm.
- Click on the wheel of "Hypothesis" → "Netgen 2D parameters" and set for "Arguments" menu:
  - Max. Size: 0.6
  - Min. Size: 0
  - Finess: Custom
  - Growth rate: 0.1

T-shape

Create a mesh

- Nb. segs per Edge: 2
- Nb. segs per Radius: 4
- Select "Limit size by Surface Curvature", "Optimize".
- Unselect "Allow Quadrangles".
- Unselect "Second Order".

- For "Local Size" menu:
  - Select "Corner" object in the Object Browser and click on "On Vertex" in the "Hypothesis Construction" window.
  - Double-click on the value in the table and set it to "0.01".
  - Click "OK".
- For "Advanced" menu:
  - Select "Fuse Coincident Nodes on Edges and Vertices".
- Click on "Apply and Close".
- Select the Mesh_1 object in the Object Browser and Right click $\rightarrow$ Compute.
- You should have a mesh with a mix of tetra and prism elements.

T-shape

Create a mesh

- As TRUST accepted only tetras elements, you can quickly tetraedrize:
  - Select Mesh_1 in the Object Browser.
  - "Modification" → "Split Volumes" and select "Tetrahedron".
  - Don't change the parameters, and click "Apply and Close".
- Check that the 4 boundaries have automatically been added in the "Group of Faces" of the **Mesh_1** object in the Object Browser.
- Export the mesh:
  - Select the Mesh_1, Right click → Export → MED file.
  - Save into a Mesh_1.med file.
- Save your study in hdf format ("File" → "Save/Save As..."), and in python format with "File" → "Dump Study..."
- **N.B.**: You can find the solutions of this exercise (T_shape.py) in $TRUST_ROOT/doc/TRUST/exercices/salome.

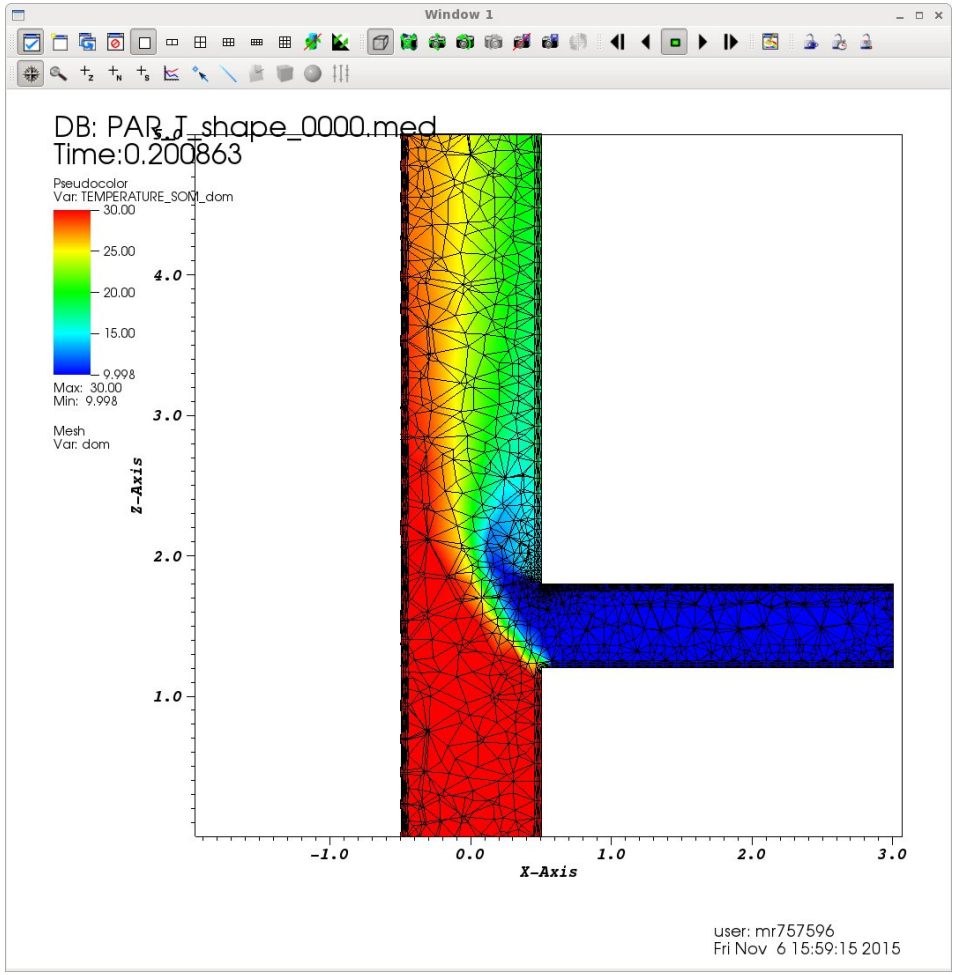
T-shape

Run with TRUST

- Copy the T_shape.data file in your directory:
  cp $TRUST_ROOT/doc/TRUST/exercices/salome/T_shape.data .
- Run it with TRUST:
  trust T_shape
  or in parallel with:
  trust -partition T_shape
  trust PAR_T_shape 4
- You can visualize the results with Visit or Salomé by opening the T_shape_0000.med file for sequential calculation or PAR_T_shape_0000.med for parallel calculation.

T-shape

Visu with Visit



Salomé: Create domains for a TRUST coupled problem

Consider that we want to simulate a coupled problem with TRUST on a complex geometry. Suppose that this latter is drawn by means of Salomé.
The main difficulty araises from the fact that the mesh elements should be connected on the interface between the two domains in order to be correctly read by TRUST.
In this exercise, you will learn how to:

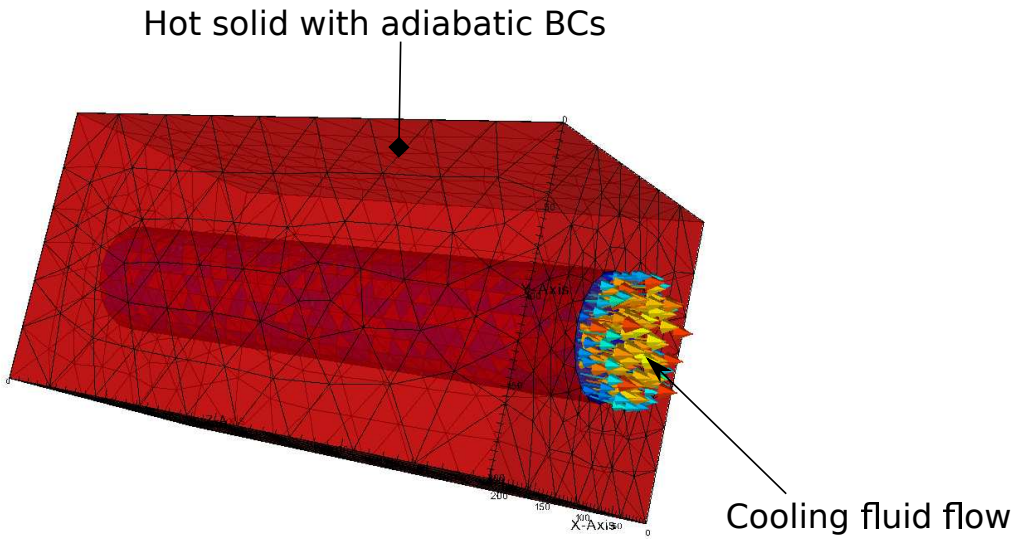
- Create two domains (domain 1 and domain 2) using Salomé
- Get a coherent mesh on the interface between the two domains. We recall that TRUST is able to treat only meshes with connected elements on the interface.
- Mesh both domains and export it into a single MED file.
- Read the MED file from TRUST datafile and simulate a coupled problem.

Note: for simple geometries, the internal TRUST mesher "Mailler" will be largely sufficient (see the exercise 3 for example).

Mesh for coupled problem

Description of the problem

Let us consider the cooling of a solid block by means of a fluid flowing inside circular cross-section channels. The channel is centered in the block of a square cross-section. The outer boundaries of the solid are adiabatic. Below is given a schematic description of the problem.



In order to build meshes using Salomé for such a simulation, we should create two domains: the first domain will represent the solid block and the second domain the fluid.

Mesh for coupled problem

In the Geometry module:

- Create a new folder for this exercise and launch Salomé:
  ```
  $ mkdir -p Formation_TRUST/yourname/salome/exo4
  $ cd Formation_TRUST/yourname/salome/exo4
  $ $PathToSalome/salome &
  ```
- Create a new study: File → New
- Select the Geometry module from drop-down menu of Salomé.
- Save your study in hdf format (Salomé format) frequently.
- Create the first geometry with: New Entity → Primitives → Box. Then, specify dimensions $Dx = 200$, $Dy = 200$ and $Dz = 400$. After that, Apply and Close.
- Create a vertex with: New Entity → Basic → Point. Specify the vertex coordinates $X = 100$, $Y = 100$ and $Z = 0$, then Apply and Close
- Create the second geometry with: New Entity → Primitives → Cylinder. Then, specify the Base Point: Vertex_1 and Vector OZ, Radius $R = 40$ and Height $H = 400$ for the cylinder. After that, Apply and Close.

Mesh for coupled problem

In the Geometry module:

- Perform a cut with: Operations → Boolean → Cut. In the Main Object select: Box_1 and in the Tool Objects select: Cylinder_1 → Apply and Close.
- Create a partition with: Operations → Partition. In Objects select: Cylinder_1 and Cut_1. Then Apply and Close.
- Define 2 groups of volumes, one for each domain with: New entity → Group → Create Group.
  1. Shape Type: Volume. Name: Solid. Main Shape: Partition_1. Select the hollow box then click on Add, after that on Apply.
  2. Shape Type: Volume. Name: Fluid. Main Shape: Partition_1. Select the cylindrical channel then click on Add, after that on Apply and Close.

Mesh for coupled problem

In the Geometry module:

- Define the groups of faces for external boundaries and the interface with: New entity → Group → Create Group.
  1. Shape Type: Surface. Name: Fluid_inlet. Main Shape: Partition_1. Then select the bottom of the cylinder and Add. Click on Apply.
  2. Shape Type: Surface. Name: Fluid_outlet. Main Shape: Partition_1. Then select the top circular boundary of the cylinder then click on Add, then Apply.
  3. Shape Type: Surface. Name: Solid_top. Main Shape: Partition_1. Then select the top of the box then click on Add, then Apply.
  4. Shape Type: Surface. Name: Solid_bottom. Main Shape: Partition_1. Then select the bottom of the box then click on Add, then Apply.
  5. Shape Type: Surface. Name: Solid_lateral_walls. Main Shape: Partition_1. Then select the remaining 4 lateral boundaries of the box then click on Add, then Apply.
  6. Shape Type: Surface. Name: Solid_Fluid_Interface. Main Shape: Partition_1. Then select the top boundary of the box and click on Hide selected, then Click on a lateral boundary and click on Hide selected, then the lateral boundary of the cylinder will be visible. Select it and click on Add then Apply.

Mesh for coupled problem

In the Mesh module:

- Create a mesh based on the Partition_1 with: Mesh → Create Mesh. Let the name be Mesh_1 and in Geometry select Partition_1. In the 3D algorithm, select NETGEN 1D-2D-3D. Click on the wheel of "Hypothesis" then on "NETGEN 3D Parameters". In Arguments, select the fineness "Fine" instead of "Moderate" then click on OK then Apply and Close.
- Right click on Mesh_1, then Compute.
- Check that the 6 boundaries have automatically been added in the "Group of Faces" of the **Mesh_1** object in the Object Browser.
- Check that the 2 volume groups have automatically been added in the "Group of Volumes" of the **Mesh_1** object in the Object Browser.
- Export the mesh in med format (if possible, choose MED 3.2).
- Dump the study and the mesh in a python script with: File → Dump Study. We will need it on the next exercice.

Mesh for coupled problem

Launch the coupled problem datafile:

- Load TRUST environment as described on page 3
- Copy the datafile:
  `$ cp $TRUST_ROOT/doc/TRUST/exercices/salome/Coupled_pb.data .`
- Run the test case using TRUST:
  `$ trust Coupled_pb.data`
- When the computation finishes, visualize results using VisIt:
  `$ visit -o Coupled_pb.lata`
- Draw the temperature profile on both domains and set the min and max on color bar to 300 and 400 respectively. When you visualize time evolution of temperature, you see that the solid is cooled and its temperature decreases. If we increase the time of the simulation, the temperature of the solid will be equal to that of the fluid at the steady state.

Aim of this exercise

Consider you already created a mesh using Salome. You will be able to change mesh and geometry parameters without starting it from scratch if you saved your study in python script.

Salome offers the possibility to save all commands launched from the Graphical User Interface, either in HDF5 format, or to save the study as a python script. If you dump your study in a python script, you can later modify some parameters and run it to build the new mesh, without having to rebuild the geometry nor the mesh in Salome.

In this exercice, you will learn how to do so.

Copy the python script

In your terminal

- Create a new directory
  ```
  $ mkdir -p Formation_TRUST/yourname/salome/exo5
  $ cd Formation_TRUST/yourname/salome/exo5
  ```
- Copy the python script you generated in the previous exercise (see page 108) and the data file
  ```
  $ cp ../exo4/Mesh_1.py .
  $ cp $TRUST_ROOT/doc/TRUST/exercices/salome/Coupled_pb.data .
  ```
  N.B.: If you have not permformed the previous exercise, you can copy the python script as follows:
  ```
  $ path=$TRUST_ROOT/doc/TRUST/exercices/salome
  $ cp $path/Coupled_pb.py Mesh_1.py
  ```

Edit geometry and meshing parameters

Edit the Mesh_1.py script in a text editor

- At the end of the `Mesh_1.py` script, add the line:
  `Mesh_1.ExportMED("Mesh_1.med",0)`
  which allows to export the generated mesh on MED format.
- Change some parameters:
  1. Change the height of the box and the cylinder: 400 → 300
  2. Change the radius of the cylinder: 40 → 70
  3. Change the cell's MaxSize in NETGEN_3D_Parameters_1: 48.9898 → 9.
  4. Change the cell's MinSize in NETGEN_3D_Parameters_1: 6.97246 → 2.
- Save and close

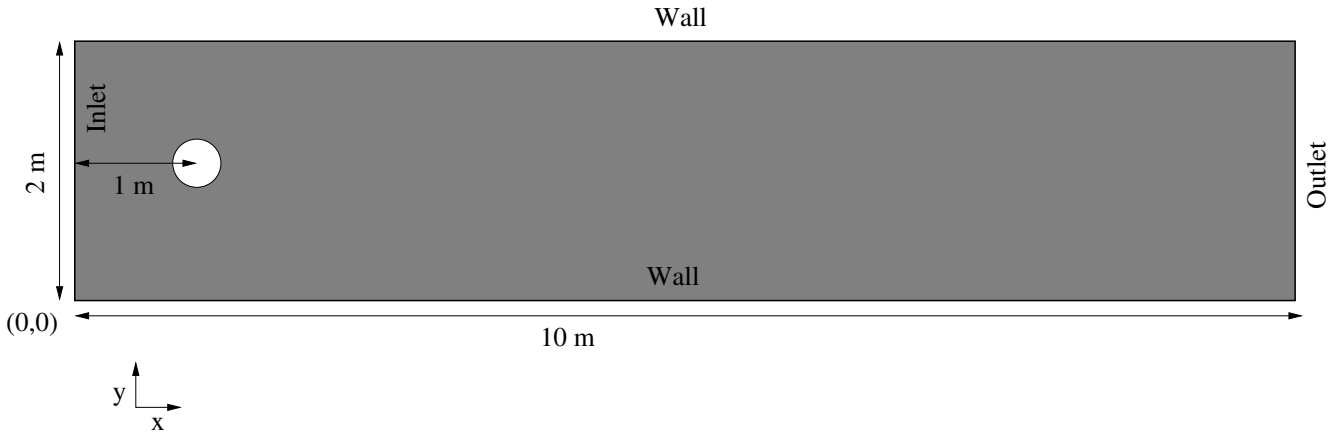
Generate the mesh and vizualize it

In your terminal

- Run your python script using the `-t` option of Salome:
  `$ $PathToSalome/salome -t Mesh_1.py`
- You should have now `Mesh_1.med` generated in your folder
  You should see that the box is smaller in the `z` direction, the cylinder is thicker, and the mesh is finer.
- You can now run the calculation on the new mesh:
  `$ trust Coupled_pb`

Gmsh to create a 2D VEF mesh

Geometry which will be created, based on a TrioCFD validation test case geometry



- Create a directory and copy an example:
  mkdir -p Formation_TRUST/yourname/gmsh
  cd Formation_TRUST/yourname/gmsh
- Load TrioCFD environment as described on page 58.
- Then copy file.geo as follows:
  dir=$project_directory/share/Validation/Rapports_automatiques/Turbulence
  cp $dir/LES/Drag/src/shape.geo file.geo
  nedit file.geo &

  gmsh file.geo &

Gmsh to create a 2D VEF mesh

- First configure gmsh to show points, lines, and surface numbers of the geometry. In menu Tools → Options → Geometry → Visibility, select Lines, Surfaces, Point labels, Line labels, Surface labels, Volume labels close this window.
- Save definitively your choices with File → Save Options As default.
- Now, look at the file.geo file, you can see the definition of parameters, points, lines... You can see the position of the points and lines with theirs numbers in gmsh.
- Modify the file to suppress the obstacle. We want to keep only 4 points and 4 lines, so you have to suppress the points 5,6,7,8 and the lines 1,3,4,5. For this, you have to:
  - comment (with "//") the definition of the points 5,6,7 and 8.
  - comment the definition of the lines 1,3,4,5. Note that the "Circle" is a line so "Circle(1)=Line(1)".
  - modify the line(2), it will now links points 1 and 2.
  - comment the "Physical Line" named "Shape" which use the line(1) (= Circle(1)) and the line(3).

Gmsh to create a 2D VEF mesh

- suppress the numbers 4 and 5 in the "Physical Line" "Axis". It refers to the lines 4 and 5 which does not exist anymore.
- suppress the numbers 1,3,4 and 5 in the "Line Loop(1)".
- set H to 2 and L to 10. (You can comment D, E, param and X definitions.)

- Press "Reload" in the gmsh GUI → Geometry to update the geometry visualization.
- Now we will add the circle. Note that you can only create circle arcs with angle strictly smaller than π!
  - create the middle of the circle like "Point(10)={1,1,0,lc};" where the triplet "1,1,0" are the coordinates of the point and "lc" the thickness of the cells next this point.
  - create 4 points around this point, which will correspond to the 4 arc of the circle:
    Point(11)={1.25,1,0,lc2};
    Point(12)={1,1.25,0,lc2};
    Point(13)={0.75,1,0,lc2};
    Point(14)={1,0.75,0,lc2};

Gmsh to create a 2D VEF mesh

- define the 4 arcs of the circle with this points:
  Circle(10)={11,10,12};
  Circle(11)={12,10,13};
  Circle(12)={13,10,14};
  Circle(13)={14,10,11}; "
- create a "Physical Line" for the circle:
  "Physical Line("Circle") = {10,11,12,13};"
  This name will be used in your TRUST data file as the name of your boundaries.
- create a line loop for the circle just after the first line loop:
  Line Loop(2) = {10,11,12,13};
- Add the number of this line loop in the "Plane Surface(1)":
  Plane Surface(1) = {1,2};

- Suppress the Physical lines "Axis" and "Top" and create a physical line named "Wall" which regroups the top and the bottom of this geometry (lines 2 and 7).

Gmsh to create a 3D VEF mesh

- Select "Mesh" in the drop-down menu of gmsh and mesh in 2D.
- Export it to a MED file: "File" $\rightarrow$ "Save As..." and name the file file.med. (Keep the default options.) You can verify your mesh by opening it with gmsh: **gmsh file.med &**.
- Build a TRUST data file with the **Postraiter_domaine** keyword, to read the mesh (like in the Salome exercise page 83). Visualize the mesh with VisIt.
- Then we will try to create a 3D mesh, by using the Extrusions feature of Gmsh. See more about extrusions in http://geuz.org/gmsh/doc/texinfo/gmsh.html.
- Save your initial file in a new one named file3D.geo.
- Comment your "Physical lines", they will not be used here.
- Add the line "Extrude {0,0,1} { Surface{1} ; }" just before the definition of the physical surface.

Gmsh to create a 3D VEF mesh

- Comment the line "Physical Surface("domain") = {1};" in 3D we will have a "Physical Volume" which will be define at the end of the .geo file.
- Define the "Physical Surface" which will be the boundaries of your geometry with the number of the surfaces which can be read on the geometry plotted by gmsh:
  Physical Surface("Inlet") = {38};
  Physical Surface("Outlet") = {30};
  Physical Surface("Wall") = {1,26,34,55};
  Physical Surface("Obstacle") = {42,46,50,54};
- Define you physical volume, you can see its number in yellow in the window:
  Physical Volume("dom") = {1};
- Select the "Mesh" tool in the drop-down menu pf gmsh and mesh in 3D your geometry. It takes a few minutes, to reduce this time, increase the size of your cells by changing the values of lc.

Gmsh to create a 3D VEF mesh

- You can use the "Optimize 3D" algorithm to optimize your mesh.
- Export your mesh to a MED file. Run gmsh again on this exported MED file to check everything is defined:
  gmsh file.med &
- Now, use your mesh in a TRUST calculation, for example:
  - copy the data file of the first exercise into a Obstacle_VEF.data file,
  - read the MED file:
    Read_MED { domain dom file file.med }
    Notice that by default with Gmsh, the mesh name is the name of the file!
  - change the discretization type,
  - be careful to the choice of the convection scheme for your VEF calculation,
  - and run the simulation on the unstructured mesh.

Validation form

The prefered route to build a validation form in TRUST and its baltiks is to build a Jupyter notebook.

See an example of notebook in
$TRUST_ROOT/Validation/Rapports_automatiques/Verification/SampleFormJupyter/
or run on terminal:
Run_fiche -doc

Validation form

- First copy the validation form named Source_canal_perio:
  mkdir -p Formation_TRUST/yourname/validation
  cd Formation_TRUST/yourname/validation
  VERIF=$TRUST_ROOT/Validation/Rapports_automatiques/Verification
  cp -r $VERIF/Verification_codage/Source_canal_perio .
  cd Source_canal_perio
- Display Run_fiche script options:
  Run_fiche -help
- Build the PDF report with:
  Run_fiche -export_pdf
  then, open the report with:
  evince build/rapport.pdf

Validation form

- Now, we are going to change the validation form (Examples are given in SampleFormJupyter Validation form):
  Run_fiche &
  - Add the mesh plot in the report. For this, at end of the notebook, add a new Markdown cell for a title:
    ##Additional information
    ###Mesh visualization
  - Add a new code cell to plot mesh
    fig=visit.Show("./std.lata", "Mesh", "dom", plotmesh=True,title="Mesh")
    fig.plot()

Validation form

- Add the evolution of residuals in the report in log scale (see .dt_ev file). For this, introduce a new Jupyter text cell and write:
  ## Residual plot
  then, on another cell plot residual using:
  Graph = plot.Graph("Residual plot")
  Graph.addResidu("std.dt_ev",label="Residu")
  Graph.scale(yscale='log')

Validation form

- Visualize the pressure field at the last time: complete the section "Additional information" with a new cell "Visualizing fields"
- Then try to find how to display that field on **SampleFormJupyter** available with **Run_fiche -doc**

Validation form

- Now, we are going to extract the number of cells and the simulation final time from three .err files and write it in .dat files. Extraction script (extraction.sh) is already available in the src directory of this validation form (you can have a look at it).
  - from your validation form, run this script using "executeScript" from run module of trustutils (see SampleFormJupyter for help).
  - Add a table to display the results of .dat files: complete the chapter "Additional information" by introducing new cells.
    Tip: look on the next slide for an example of table plot.

Validation form

Here is an example of a Jupyter cell for displaying a table.

```
from trustutils import plot

tableau=plot.Table(["$$\| P \|_{\infty}$$","$$\| \overrightarrow{v} \|_{\infty}$$"])
data=plot.loadText("droite_erreurs.txt")
tableau.addLigne([data],"droite")
data=plot.loadText("gauche_erreurs.txt")
tableau.addLigne([data],"gauche")
display(tableau)
print ([data])
```

	$\|P\|_\infty$	$\|\vec{v}\|_\infty$
droite	1.455192e-11	9.135146e-11
gauche	7.275958e-12	5.718615e-11

```
[array([7.27595761e-12, 5.71861465e-11])]
```

Validation form

- Now we are going to add a fourth test case: "debit4"
  - "debit4" corresponds to "std" test case with zero initial velocity and imposed flow rate to $2m^3/s$ on "periox" boundary.
  - Add the test case using "substitute" and "addCase"
- We are going to rerun the validation form.
  - Re-build the whole validation form by clicking on the icone corresponding to the ̈restart of the Jupyter kernel and run of the whole notebook:̈
  - build the pdf report using:
    Run_fiche -export_pdf
  - NB: You can add the results of this test case to your "visualization" and to the "table".

Unix Quick Reference

File Commands

ls	Directory listing
ls -al	Formatted listing with hidden files
ls -lt	Sorting the Formatted listing by time modification
cd dir	Change directory to dir
cd	Change to home directory
pwd	Show current working directory
mkdir dir	Creating a directory dir
cat >file	Places the standard input into the file
more file	Output the contents of the file
head file	Output the first 10 lines of the file
tail file	Output the last 10 lines of the file
tail -f file	Output the contents of file as it grows,starting with the last 10 lines
touch file	Create or update file
rm file	Deleting the file
rm -r dir	Deleting the directory

Unix Quick Reference

File Commands

rm -f file	Force to remove the file
rm -rf dir	Force to remove the directory dir
cp file1 file2	Copy the contents of file1 to file2
cp -r dir1 dir2	Copy dir1 to dir2;create dir2 if not present
mv file1 file2	Rename or move file1 to file2,if file2 is an existing directory
ln -s file link	Create symbolic link link to file

Unix Quick Reference

Process management

ps	To display the currently working processes
top	Display all running process
kill pid	Kill the process with given pid
killall proc	Kill all the process named proc
pkill pattern	Will kill all processes matching the pattern
bg	List stopped or background jobs,resume a stopped job in the background
fg	Brings the most recent job to foreground
fg n	Brings job n to the foreground

File permission

chmod octal file	Change the permission of file to octal, which can be found separately for user, group, world by adding: 4-read(r) 2-write(w) 1-execute(x)

Unix Quick Reference

Searching

grep pattern file	Search for pattern in file
grep -r pattern dir	Search recursively for pattern in dir
command | grep	Search pattern in the output of a command pattern
locate file	Find all instances of file
find . -name filename	Searches in the current directory (represented by a period) and below it, for files and directories with names starting with filename
pgrep pattern	Searches for all the named processes , that matches with the pattern and, by default, returns their ID

Unix Quick Reference

System Info

date	Show the current date and time
cal	Show this month's calender
uptime	Show current uptime
w	Display who is on line
whoami	Who you are logged in as
finger user	Display information about user
uname -a	Show kernel information
cat /proc/cpuinfo	Cpu information
cat proc/meminfo	Memory information
man command	Show the manual for command
df	Show the disk usage
du	Show directory space usage
free	Show memory and swap usage
whereis app	Show possible locations of app
which app	Show which applications will be run by default

Unix Quick Reference

Compression

tar cf file.tar file	Create tar named file.tar containing file
tar xf file.tar	Extract the files from file.tar
tar cf file.tar file	Create tar named file.tar containing file
tar xf file.tar	Extract the files from file.tar
tar czf file.tar.gz files	Create a tar with Gzip compression
tar xzf file.tar.gz	Extract a tar using Gzip
tar cjf file.tar.bz2	Create tar with Bzip2 compression
tar xjf file.tar.bz2	Extract a tar using Bzip2
gzip file	Compresses file and renames it to file.gz
gzip -d file.gz	Decompresses file.gz back to file

Unix Quick Reference

Network

ping host	Ping host and output results
whois domain	Get whois information for domains
dig domain	Get DNS information for domain
dig -x host	Reverse lookup host
wget file	Download file
wget -c file	Continue a stopped download

Unix Quick Reference

Shortcuts

"Ctrl"+c	Halts the current command
"Ctrl"+z	Stops the current command, resume with fg in the foreground or bg in the background
"Ctrl"+d	Logout the current session, similar to exit
"Ctrl"+w	Erases one word in the current line
"Ctrl"+u	Erases the whole line
"Ctrl"+r	Type to bring up a recent command
!!	Repeats the last command
exit	Logout the current session

Index I

Index II

Index III

Index IV

End

Tutorial solutions:
$TRUST_ROOT/doc/TRUST/exercices/Tutorial_solutions.pdf