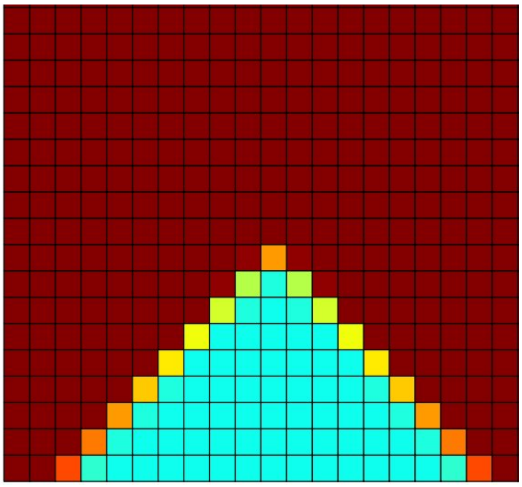
RESEARCH REPORT

VTT-R-00446-11



Introduction to PORFLO and user instructions 2010

Authors:	Eveliina Takasuo, Ville Hovi, Mikko Ilvonen
Confidentiality:	Public

Report's title Introduction to PORFLO and user instructions 2010	
Customer, contact person, address SAFIR2010 National Research Programme on Nuclear Power Plant Safety	Order reference 7/2007 SAF
Project name Tridimensional core transient analysis methods PORFLO-koulutus ja sovellus partikkelikeon jäähdytettävyyteen	Project number/Short name 41435/TRICOT_2010 43216/PORFLO-koulutus
Author(s) Eveliina Takasuo, Ville Hovi, Mikko Ilvonen	Pages 25 p. / 2 appendices
Keywords PORFLO code, thermal hydraulics, porous medium, two-phase flow, user instructions	Report identification code VTT-R-00446-11

Summary

PORFLO is a 3D two-phase flow simulation code which utilizes the porous medium approach in the modelling of complex geometries. The code has been developed specifically for thermal hydraulic safety analysis of nuclear power plants. The code has recently been under development at VTT for the following applications: core debris coolability, secondary side of steam generators, flows across fuel bundles and the EPR reactor downcomer, lower plenum and core. In the future, the aim is to connect PORFLO to the family of system codes and reactor core simulators, particularly TRAB-3D-SMABRE, that has been developed by VTT. A brief review of the general solution methods utilized in PORFLO and the current status of the code is presented. User instructions and guidelines on how to set up, run and post-process PORFLO simulations are also given.

Confidentiality	Public

Espoo 24.1.2011

Written by	Reviewed by	Accepted by
Eveliina Takasuo Research Scientist	Elina Syrjälahti Research Scientist, TRICOT Project Manager	Timo Vanttola Technology Manager

VTT's contact address
PO Box. 1000, 02044 VTT, Finland

Distribution (customer and VTT)
SAFIR2010 and SAFIR2014 Reference Group 3
VTT: Timo Vanttola, Eija Karita Puska, Vesa Suolanen, Jarto Niemi, Pasi Inkinen

Preface

The PORFLO code development and modelling work has been conducted mainly within the TRICOT project in the frame of the SAFIR2010 Research Programme on Nuclear Power Plant Safety (http://virtual.vtt.fi/virtual/safir2010). In 2010, significant support to the SAFIR2010 work has been provided by a self-financed project which aims for training a new code user and extending the PORFLO applications to the coolability of porous core debris beds.

The Authors

Contents

1 Introduction

PORFLO is a two-phase flow simulation code developed at VTT for solving 3D problems in nuclear power plant thermal hydraulics. Applications of the code thus far include the modelling of the secondary side of the Loviisa VVER-440 steam generators, BWR fuel bundles in the frame of the BFBT calculation benchmark, isolation condensers of the passive safety systems of BWRs, particle bed coolability and the flows in the EPR reactor core. The code is presently under active development at VTT.

An important objective in the future is to further develop the PORFLO modelling of flows in the EPR reactor downcomer, lower plenum and core. Especially, the effect of lateral cross-flow between the fuel elements would be an interesting research issue from the safety point of view because such flows tend to even out the temperature differences between the fuel assemblies. This issue was investigated in the frame of the SAFIR2010 research programme within the TRICOT project, and will be continued in the SAFIR2014 project KOURA. In the future, modelling of the behavior of passive safety systems is foreseen to be an important topic because of new types of nuclear power plants, and the continuation of the modelling of heat exchangers in these systems would also be of interest.

Another long-term goal of the code development is coupling the code to the thermal hydraulics system code SMABRE and, eventually, to the neutronics models (TRAB-3D) in nuclear reactor core analysis. The possibility to parallelize the code has also been investigated but not yet put to practice in production runs.

Another specific application of PORFLO is the heat transfer and two-phase flow in a core debris bed that generates decay heat - a problem specific to severe accident analysis. Predicting the dryout heat flux (coolability) in a porous debris bed was one of the first applications of the PORFLO predecessor PILEXP code (Hyytinen, 2001). The re-introduction of this problem to be solved with the up-to-date PORFLO version was done within the SAFIR2010/HYBCIS2 project. A new laboratory-scale test facility, COOLOCE (Coolability of Cone) using which the coolability of a heap-like (conical) debris bed may be investigated, was assembled in the project in 2010 (Takasuo et al. 2010).

Traditionally, the dryout behavior of porous particle beds consisting of core debris has been evaluated by 1D models or simple correlations. However, in realistic accident scenarios the particle bed may have a highly irregular structure or a heap-like geometry. In these cases, the simplified approaches might not be adequate due to complex multi-dimensional flow configurations allowed by the debris bed geometry. Also, multi-dimensional flooding is present in the conical test-set up which means that a full 3D model is required for detailed analyses of the problem.

PORFLO solves the Navier-Stokes equations for liquid and gas phases in three dimensions. In contrast to many other CFD codes, PORFLO utilizes porous medium approach instead of body-fitting the mesh to the modelled geometry. In this approach, the exact locations of the walls of solid structures are not known.

The advantage of the approach is that it allows the modelling of complex systems such as tube bundles without the heavy computational effort that would be necessary with a detailed model of the flow channels. The inherent drawback of this approximation is that the exact flow field within the pores or channels of the geometry is not known, and the physical phenomena including heat transfer and turbulence are averaged over a selected volume which contains both fluid and solid phases.

This report describes the code development status in 2010 and gives general user instructions for setting up and running PORFLO simulations in Linux environment. Instructions for post-processing are also presented and examples of using the multi-purpose StarNode visualization tool developed by VTT are given.

2 PORFLO basics

PORFLO solves the six conservation equations of mass, momentum and energy for the liquid and gas phases. Also, a 5-equation model in which the liquid and gas phases are separated by the drift-flux model is available but this has been replaced by the six-equation model in recent applications. The continuity equations for the liquid (Eq. 1) and gas (Eq. 2) are

$$\frac{\partial\left[\varepsilon\left(1-\alpha\right)\rho_\mathrm{l}\right]}{\partial t}+\nabla\cdot\left[\varepsilon\left(1-\alpha\right)\rho_\mathrm{l}\mathbf{u}_\mathrm{l}\right]=-\gamma \tag{1}$$

$$\frac{\partial\left(\varepsilon\alpha\rho_\mathrm{g}\right)}{\partial t}+\nabla\cdot\left(\varepsilon\alpha\rho_\mathrm{g}\mathbf{u}_\mathrm{g}\right)=+\gamma \tag{2}$$

where t is time [s], ε is porosity [-], $\mathbf{u}_\mathrm{g}$ is vapor velocity [m/s], $\mathbf{u}_\mathrm{l}$ is liquid velocity [m/s], α is void fraction [-], ρ_g is vapor density [kg/m^3], ρ_l is liquid density [kg/m^3] and γ is evaporation / condensation rate per unit volume [kg/m^{3}s]. The momentum equations for the liquid (Eq. 3) and gas (Eq. 4) phases are

$$\begin{aligned}&\frac{\partial\left[\varepsilon\left(1-\alpha\right)\rho_\mathrm{l}\mathbf{u}_\mathrm{l}\right]}{\partial t}+\nabla\cdot\left\{\left[\varepsilon\left(1-\alpha\right)\rho_\mathrm{l}\mathbf{u}_\mathrm{l}\right]\otimes\mathbf{u}_\mathrm{l}\right\}=-\varepsilon\left(1-\alpha\right)\nabla p+\varepsilon\left(1-\alpha\right)\nabla\cdot\mathbf{T}\\&+\varepsilon\left(1-\alpha\right)\rho_\mathrm{l}\mathbf{g}+\mathbf{F}_\mathrm{PC}+\mathbf{F}_\mathrm{IF}+\mathbf{F}\end{aligned} \tag{3}$$

$$\frac{\partial\left(\varepsilon\alpha\rho_\mathrm{g}\mathbf{u}_\mathrm{g}\right)}{\partial t}+\nabla\cdot\left[\left(\varepsilon\alpha\rho_\mathrm{g}\mathbf{u}_\mathrm{g}\right)\otimes\mathbf{u}_\mathrm{g}\right]=-(\varepsilon\alpha)\nabla p+(\varepsilon\alpha)\nabla\cdot\mathbf{T}+\varepsilon\alpha\rho_\mathrm{g}\mathbf{g}-\mathbf{F}_\mathrm{PC}-\mathbf{F}_\mathrm{IF}+\mathbf{F} \tag{4}$$

where p is pressure [Pa], $\mathbf{T}$ is the surface stress tensor [N/m^2], $\mathbf{F}_\mathrm{IF}$ is the interfacial friction force [N/m^3], $\mathbf{F}_\mathrm{PC}$ is the momentum transfer due to phase change [N/m^3] and $\mathbf{F}$ [N/m^3] is used to describe other forces in the system such as drag between the fluids and solid particles in the case of particle beds. The energy equations have the following form:

$$\frac{\partial\left[\varepsilon\left(1-\alpha\right)\rho_{\mathrm{l}}h_{\mathrm{l}}\right]}{\partial t}+\nabla\cdot\left[\varepsilon\left(1-\alpha\right)\rho_{\mathrm{l}}h_{\mathrm{l}}\mathbf{u}_{\mathrm{l}}\right]=Q_{l} \tag{5}$$

$$\frac{\partial\left(\varepsilon\alpha\rho_{\mathrm{g}}h_{\mathrm{g}}\right)}{\partial t}+\nabla\cdot\left(\varepsilon\alpha\rho_{\mathrm{g}}h_{\mathrm{g}}\mathbf{u}_{\mathrm{g}}\right)=Q_{g} \tag{6}$$

where h_{g} is vapor enthalpy [J/kg] and h_{l} is liquid enthalpy [J/kg]. The terms Q on the right hand sides represent the heat fluxes to and from the respective phase in the control volume. These depend on heat transfer correlations which are application-dependent. In addition to the fluid-phase temperatures, conduction in solid phases can be solved. This is used in modeling the COOLOCE experiments in which the effectiveness of heat transfer from the heating elements to the solid particles is estimated, i.e. in this case two different types of solids are treated separately.

In addition to the calculation of heat transfer, closure models are incorporated for the different drag forces between the phases and turbulence for which the standard k-ε model is under development. For more information of the physical models, see Hovi (2007), Hovi and Ilvonen (2010) and Takasuo et al. (2011). A list of the main PORFLO updates in 2010 for the code development “branch” of the particle bed application is presented in Appendix A.

The iterative solution process of the PORFLO six-equation model is presented in Fig. 1. The pressure-velocity solution is done by the SIMPLE algorithm (Semi-Implicit Method for Pressure-Linked Equations) originally developed by Patankar (1980).

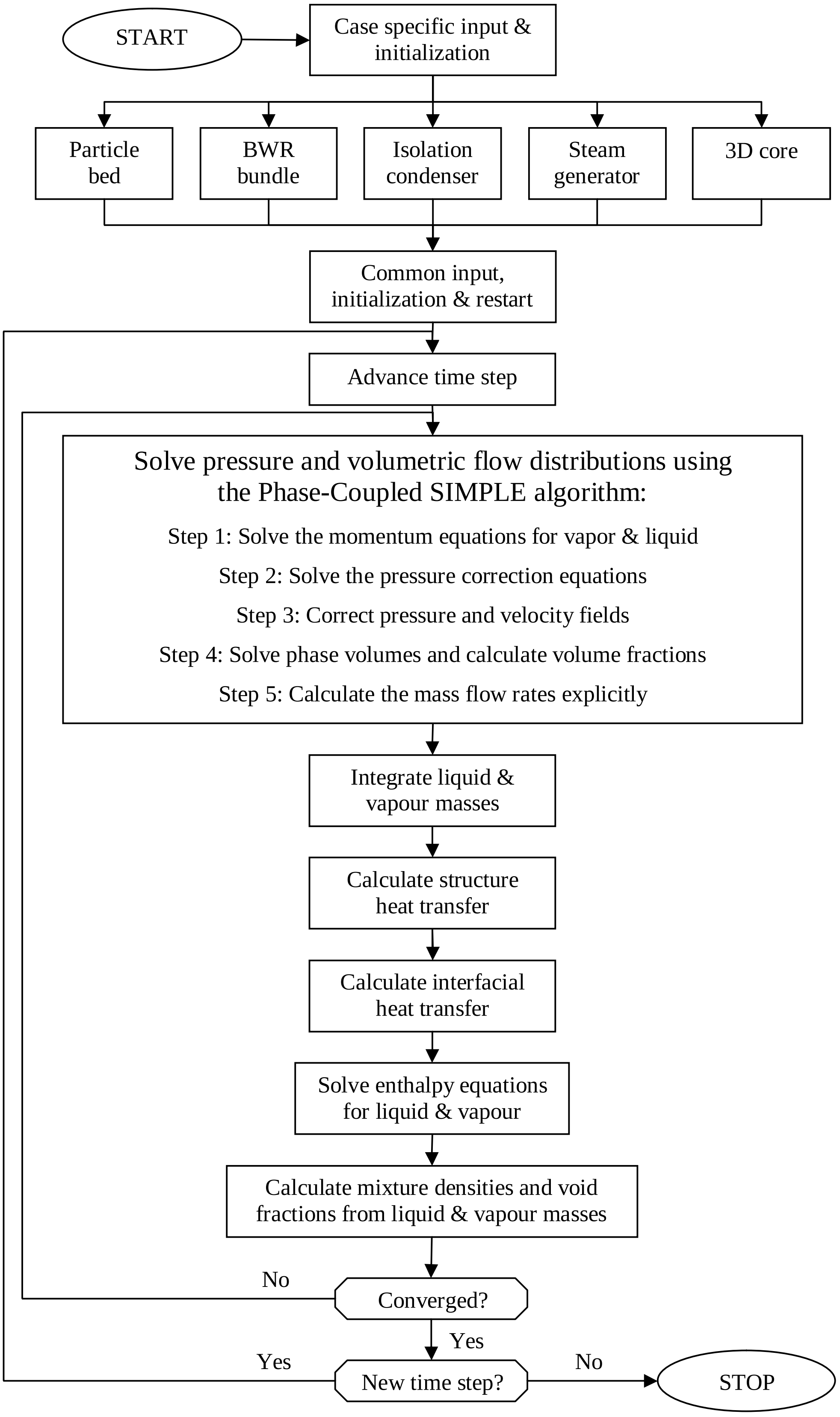

START
Case specific input & initialization
Particle bed
BWR bundle
Isolation condenser
Steam generator
3D core
Common input, initialization & restart
Advance time step
Solve pressure and volumetric flow distributions using the Phase-Coupled SIMPLE algorithm:
Step 1: Solve the momentum equations for vapor & liquid
Step 2: Solve the pressure correction equations
Step 3: Correct pressure and velocity fields
Step 4: Solve phase volumes and calculate volume fractions
Step 5: Calculate the mass flow rates explicitly
Integrate liquid & vapour masses
Calculate structure heat transfer
Calculate interfacial heat transfer
Solve enthalpy equations for liquid & vapour
Calculate mixture densities and void fractions from liquid & vapour masses
No
Converged?
Yes
Yes
New time step?
No
STOP

Fig. 1. The solution procedure of PORFLO.

3 Setting up simulations

PORFLO simulations are set-up by editing the input file (porflo.dat) and modifying the application specific parts of the source code. The code does not include a general-form user interface. Thus, it is almost always necessary to edit the source code to specify the simulation parameters. Naturally, source-editing has to be done also if the user wishes to apply new models and correlations. It should be noticed that, to some extent, programming is also necessary when using commercial codes such as FLUENT in which User-Defined Functions (UDFs) are used to include specific models which are not available as default.

Even though easy readability and comprehensive comments are emphasized, source editing is not always a straightforward process, and the user should be familiar with the basics of programming in Fortran and the architecture of PORFLO. On the other hand, user programming offers a lot of possibilities for modifying the code for solving different kinds of flow problems. The source code consists of the files listed in Table 1. An explanation of each of the modules is given.

Table 1. PORFLO source files.

File name	Contents
Parameters.F90	Dimensioning parameters for meshes and the material properties for the fluid and solid phases
OldRoutines.f	Fixed form source code from previous PORFLO versions
Geometry.F90	The routines for mesh generation and mesh-based initial data (i.e. conversion of geometrical data to a format readable by PORFLO)
LinearSolvers.F90	Solution of the linear systems of equations (Contains several preconditioned linear iterative solution methods, such as CG and BiCGSTAB)
FlowSolution.F90	Coupled solution of pressure and velocity fields: the Phase-Coupled SIMPLE algorithm.
ScalarTransport.F90	Solution of scalar transport equations (a general-purpose scheme)
HeatTransfer.F90	Heat transfer and temperature solution
TestiMain.F90	Main program, input read and initialization, the main iteration loop, output write

The PORFLO code has been applied to specific cases addressing two-phase flow in complex geometries. Because the code has been under continuous development during recent years, it should be noticed that the applications utilize different versions of the code. This report focuses on the code development in 2010 and the latest version that has been set-up for the particle bed simulations in November 2010. However, since the code development has been done with the aim being the ability to model a specific phenomenon or a system, the starting point for the selection of the physical models may be understood through the different applications. Thus far, the code has been utilized in the following applications:

1) Isolation condensers (Miettinen and Karppinen, 2003)
2) BWR fuel bundle and modelling of the NEA/NRC BFBT benchmark (Ilvonen and Hovi, 2009)
3) Horizontal steam generators (research report and a conference publication by Hovi and Ilvonen, 2010)
4) Porous core debris beds (Takasuo et al. 2011)
5) 3D model of the EPR core (modelling work has been started in 2010 but not yet published)

The cases deal with a multitude of flow configurations and heat transfer modes. Thus, each application requires a different selection of models for heat transfer, frictional pressure losses and possibly turbulence. For instance, the frictional pressure drop in the interior of a densely packed particle bed is significantly greater than the pressure drop across steam generator tubes.

The closure models of the momentum conservation equations can be found in the code module named FlowSolution.F90. The friction models for the drag forces between the solid and fluid phases and the interfacial friction are programmed as separate subroutines. The user may modify these subroutines and determine the friction (momentum exchange) coefficients according to the needs of his/her application. For the particle bed application, the selection of the friction models is provided in the input file which is presented in the next section.

The calculation of heat transfer coefficients is found in the module HeatTransfer.F90. The module calculates the heat transfer rates between the three different phases, namely liquid, solid and vapour, and the phase interface (boiling). The possibility to solve heat conduction within solid materials has been included in the particle bed model and the fuel rod bundle model that was used in the BFBT benchmark. The correlations for the different heat fluxes are clearly commented in the code and the user may modify or add his/her own correlations into the module.

In addition, appropriate boundary conditions have to be specified for the simulations. The below options are pre-set in the input file named “porflo.dat” (see the example in Section 3.2).

1) Frictionless wall
2) Velocity inlet
3) Pressure outlet

The inclusion of other boundary types is possible and will be done according to the requirements of future applications and user needs.

3.1 Grid generation

The spatial discretization is based on 3D Cartesian staggered grids. A sub-routine that performs the creation of the grid for the problem geometry has to be included for each application. This routine generates the three-dimensional arrays for porosity, power sources and other node-specific data such as the areas used in heat transfer calculations. The geometry and grid information is handled on the level of the source code (in contrast to many commercial codes such as GAMBIT), so modifications to be made into the grid have to be programmed into the Geometry.F90 module which contains sub-routines for each of the five applications.

The physical boundaries of the modelled geometry are “smoothed” on the Cartesian grid in such a way that in the grid cell, in which the physical boundary is located, porosity of the cell is weighted according to the fractions of different structures in the cell in question. Using this method, even highly complex geometries can be described with relative ease but with the cost of losing information of the exact geometry in the flow solution.

Example grids are presented in Fig. 2 and Fig. 3. The vertical cross-sections in the centre of the 3D grids for the conical particle bed (that of the COOLOCE facility) and for the somewhat simpler cylindrical particle bed are illustrated in Fig. 2 and Fig. 3, respectively. The cell size in the example grids is 3 cm.

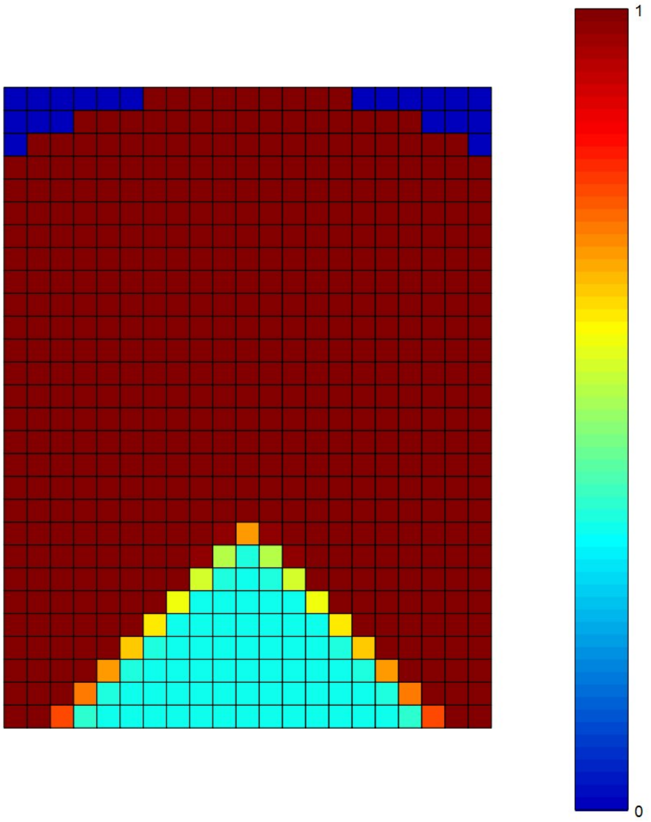


Fig. 2. Cartesian 21x21x28 grid for the COOLOCE particle bed including a large pool volume. The color scale indicates porosity. Porosity in the conical region is approximately 36%.

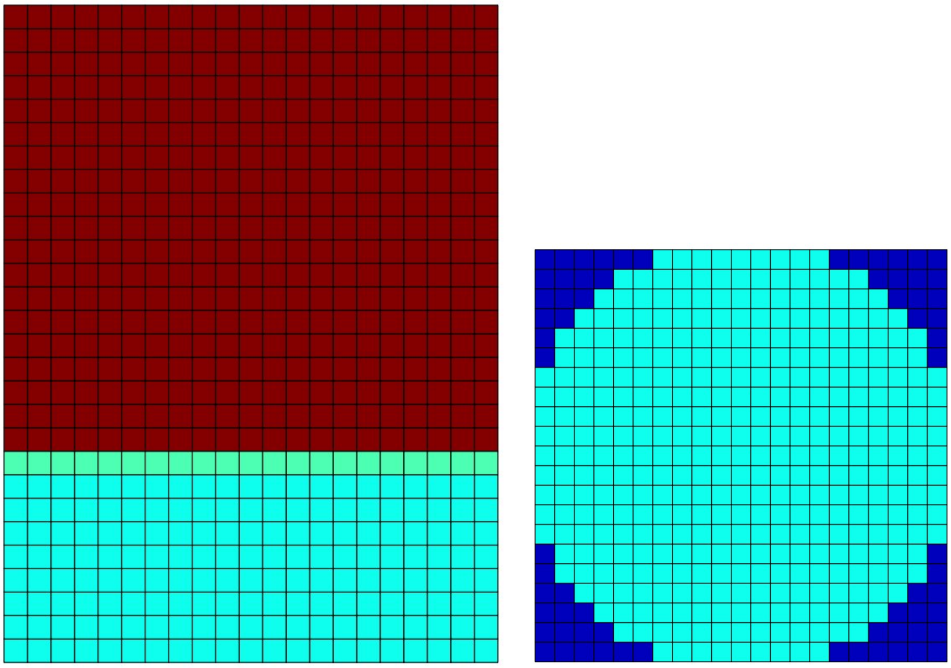


Fig. 3. Cartesian 21x21x28 grid for a cylindrial particle bed: vertical (left) and horizontal (right) cross-sections. The grid consists of a pool region and a particle bed region.

3.2 Example input

The control of simulation time, time step size and the interval of generating output (restart) files are done in the input file named “porflo.dat”, along with the selection of numerical and physical parameters. The input for the porous particle bed case is presented next with explanations of each section given below the sections.

```
########################################################################
#                 PORFLO v2.0 INPUT FILE                               #
########################################################################
#
########################## GENERAL INPUT ##############################
#
# Restart mode: 0 = run from beginning, 1 = restart
    0
#
# Restart interval (number of time-steps):
# The restart / result file is written to a unique file at the end of
# the interval.
  2000
#
# Launch Matlab? 0 = no, 1 = yes
    0
#
# Close the residual file after write? 0 = no, 1 = yes
    1
```

```
#
# Range of residual plotting (the number of iterations plotted)
   200
#
# Gravity:
#  x-comp    y-comp    z-comp
     0.0,      0.0,    -9.81
#
# Fictional loss coefficient, for FREE FLOW only (open pool)
     0.0d+0
#
# Superheat of the heater elements [C]
     5.0
#
######################################################################
```

In the above section, we first define if the calculation is restarted from a previous calculation (the simulation parameters are read from an output file). The restart interval gives the frequency of writing output files in time steps. Residuals of the solution may be monitored during the calculations using Gnuplot, for which the range of iterations to be plotted is given. For this, the option "Close the residual file after write?" should be set to 1.

In addition, the following physical parameters are given: gravity component, "fictional" loss coefficient (this is a friction coefficient given for the free flow of the pool volume to make the calculation numerically stable in cases in which the pool dynamics is not modelled in detail and no wall friction is taken into account) and superheat of the heater elements. The superheat is given in case surface temperature of the solid phase is constant.

```
######################## SOLUTION CONTROL ###########################
#
# Minimum number of SIMPLE iterations per time-step
     5
#
# Maximum number of SIMPLE iterations per time-step
   100
#
# Is explicit Temperature / Heat transfer solution used? 0 = no, 1 = yes
     1
#
# Maximum number of Heat transfer / Temperature iterations
     5
#
# Maximum allowed norm of Temperature change
      1.0d-10
#
# Convergence criteria:
#
#  Sum of P Cor RHS    Max of P Cor RHS
      1.0d-8,            1.0d-9
#
#############################################################################
```

The number of iterations and convergence criteria for the pressure/velocity and heat transfer solutions are given in the section above. Explicit temperature solution may be selected which means that the temperatures of the three phases are solved once in a time-step, otherwise temperatures are solved iteratively within each SIMPLE iteration. Explicit temperature solution may be chosen to improve the convergence and calculation speed of certain cases. The maximum number of temperature iterations (5 iterations in the example) defines the maximum number of heat transfer / temperature iterations, within a loop in which all the heat transfer coefficients are calculated before the simultaneous solution of all the temperatures.

```
######################### SOLUTION METHODS ############################
#
# Index for selecting the friction model for
# FREE FLOW (open pool):
#     1 = Lipinski
#     2 = Reed
#     3 = Hu & Theofanous
#     4 = Tung & Dhir
#     5 = Isachenko (for Horizontal SG)
#
    4
#
# Index for selecting the friction model for
# FLOW THROUGH POROUS MATERIAL:
#     1 = Lipinski
#     2 = Reed
#     3 = Hu & Theofanous
#     4 = Tung & Dhir
#     5 = Isachenko (for Horizontal SG)
#
    4
#

# Index for selecting the InterPhase Friction model for
# FREE FLOW (open pool):
#     0 = No InterPhase Friction
#     1 = Tung & Dhir
#     2 = Isachenko (for Horizontal SG)
#
    1
#
# Index for selecting the InterPhase Friction model for
# FLOW THROUGH POROUS MATERIAL:
#     0 = No InterPhase Friction
#     1 = Tung & Dhir
#     2 = Isachenko (for Horizontal SG)
#
    1

#########################################################################
```

The selection of the physical friction models is given above. First, the friction model for the open pool with no porosity as the default is given (indices 1-5). The first four options are aimed for the particle bed application and the last one for the horizontal steam generator application. It should be noticed that the present code version does not include models verified for open pools. The flows in open pools are approximated using the porous medium and tube bundle friction models (1-5) or, alternatively, the “Fictional loss coefficient” in the General input section may be used.

Next, the friction model between the solid matrix and the fluid phases is given for the porous section of the geometry. The first four options (1-4) are commonly known models that are verified and validated for particle bed simulations and the last one (5) for the horizontal steam generator with flow across tube bundles. For information on the selection of a suitable particle bed friction model, see Takasuo et al. (2010).

The last two options give the friction between the gas and liquid phases. This is separately given for the particle bed region and the open pool region. The interfacial friction may also be “turned off” by selecting zero. For the particle bed application, the Tung and Dhir model is the recommended selection.

```
############################ DISCRETIZATION ################################
#
# Index for selecting the discretization scheme:
#     1 = Downwind scheme
#     2 = Central difference scheme
#     3 = Upwind scheme
#     4 = Second order upwind scheme
#     5 = FROMM
#     6 = SUPERBEE
#     7 = MINMOD
#     8 = OSHER
#     9 = MUSCL
#
# Discretization scheme for Velocities
      3
#
# Discretization scheme for Volume fractions
      3
#
# Discretization scheme for Temperatures
      3
#
##################### UNDER-RELAXATION FACTORS##########################
#
# Under-Relaxation Factor for Pressure
    0.7
#
# Under-Relaxation Factor for Velocities
    0.5
#
# Under-Relaxation Factor for Volume fractions
    1.0
#
# Under-Relaxation Factor for Temperatures
    1.0
###############################################################################
```

The discretization schemes and under-relaxation factors are defined above. The default option is first order upwind. Often, the values that produce the fastest convergence are case-dependent and have to be selected by trial-and-error.

```
#################### ITERATIVE LINEAR SOLUTION #########################
#
#  Minimum     Maximum
# number of   number of   Tolerance
#   iters       iters
#
       3,       1000,     1.0e-8    ! Velocities
#
      10,       10000,     1.0e-8    ! Pressure correction
#
       3,       1000,     1.0e-10   ! Volume fractions
#
       3,       1000,     1.0e-10   ! Temperatures
#
######################################################################
```

The minimum and maximum number of iterations and the convergence criterion for the solutions of velocity, pressure correction, volume fractions of the phases and temperature are given in this section. “Tolerance” indicates the difference in the Euclidean norm of the residuals. The default values are seen above.

```
############################ MONITORS ################################
#
# Number of monitors
     9
#
# Variable indices:
#     1 = Velocity
#     2 = Mass flow rate
#     3 = Pressure
#     4 = Volume fraction
#     5 = Temperature
#
# type  i-index  j-index  k-index  phase   var
    0,      11,      11,       9,     2,     4  ! monitor-1
    0,      11,      11,       8,     2,     4  ! monitor-2
    0,      11,      11,       7,     2,     4  ! monitor-3
    0,      11,      11,       6,     2,     4  ! monitor-4
    0,      11,      11,       5,     2,     4  ! monitor-5
    0,      11,      11,       4,     2,     4  ! monitor-6
    0,      11,      11,       3,     2,     4  ! monitor-7
    0,      11,      11,       2,     2,     4  ! monitor-8
    0,      11,      11,       1,     2,     4  ! monitor-9
#
######################################################################
```

In this section, monitors that record the history of the selected variables can be set-up. The indices i, j and k refer to the cells (x,y,z) whose values are to be saved in a monitor file “monitors.txt”. In the “phase” index, 1 and 2 denote the liquid and gas phases, respectively. The values of the specified variables are saved each time-step. In the example above, the volume fraction of gas is saved in nine cells at successive layers.

```
############################ ANIMATION #################################
#
# Number of variables saved for animation
# (If this is set zero, do not give any of the information below)
    5
#
# Variable indices:                    Phase indices:
#     1 = Velocity                     0 = All phases
#     2 = Mass flow rate                 1 = phase-1
#     3 = Pressure                      2 = phase-2
#     4 = Volume fraction                  .
#     5 = Temperature                      .
#
#  variable,  phase
          1,      0      ! animation-1
          2,      0      ! animation-2
          3,      0      ! animation-3
          4,      0      ! animation-4
          5,      0      ! animation-5
#
# Animation interval (in seconds):
    1.0d+1
#
# Beginning time for animation (in seconds):
    0.0d+0
#
# End time for animation (in seconds):
    10000.0d+0
#
#######################################################################
```

The specifications for saving the binary data for generating animations of the simulations are given in this section. The selected variables (1-5) and the phase (0-2) are saved for all mesh points. The interval of the animation screens and the beginning and end simulation times for the animation are given. The default name of the animation file is “animation.bin”.

```
######################### INITIAL VALUES ###############################
#
# Beginning of the simulation [s]
  0.0d+0
#
# Length of time-step [s]
  1.0d-2
#
# Total heating power [W]
  1.0d+0
#
# Liquid mass flow rate at inlet [kg/s]
  0.0d+0
#
# System pressure [Pa]
  1.0d+5
#
######################################################################
```

This section presents the initial values and the width of the time-step used in the simulation. If a restart file is read, this part of the input is skipped. The heating power is given as total power of all the heat generating structure of the geometry in watts.

```
############ NEW INPUT VALUES FOR EACH TIME INTERVAL #################
#
# time_max, dtime, total_power,    glin, p_system
#     [s]       [s]                [W]       [kg/s]     [Pa]
    10.000,  1.d-2,     -15.0d+3,   0.00,  1.0d5
    500.00,  1.d-2,      15.0d+3,   0.00,  1.0d5
    510.00,  1.d-2,     -20.0d+3,   0.00,  1.0d5
    1000.00,  1.d-2,     20.0d+3,   0.00,  1.0d5
    1010.00,  1.d-2,    -25.0d+3,   0.00,  1.0d5
    1500.00,  1.d-2,     25.0d+3,   0.00,  1.0d5
    1510.00,  1.d-2,    -30.0d+3,   0.00,  1.0d5
    2000.00,  1.d-2,     30.0d+3,   0.00,  1.0d5
    2010.00,  1.d-2,    -35.0d+3,   0.00,  1.0d5
    2500.00,  1.d-2,     35.0d+3,   0.00,  1.0d5
      -1.0,   1.d-1,     35.0d+3,   0.00,  1.0d5
#
############################################################################
```

In this section, the time-dependent input values are given. These are time-step width “d_time”, heating power “total_power”, inlet boundary mass flow rate “glin” and system pressure “p_system”. The values signed “-“ indicate that the (possible) change of the value is done as a liner increase from the previous value up to the time indicated on the same line, “+” indicates that the change is instantaneous. For restart calculations, it should be noticed that the “time_max” value has already been read at the time of the restart file save, and the time-dependent input for restart calculations should be started from a time value greater than the “time_max” that is set in the restart text file. For restarting from a previous save, the output text file should be named “porflo0.txt”.

4 Running calculations

After the simulation has been set up and the input file (porflo.dat) generated, the source code including the case-specific modifications has to be compiled. The code developers use Intel Fortran Compiler for Linux operating systems and Visual Studio for Windows. Once compiled, calculation is started by executing the output file generated by the compiler (we recommend to name the file “porflo.exe”). Most of the simulations have been run on a Linux cluster that consists of four computer nodes. A typical running time for a particle bed calculation with 21 x 21 x 28 cells (with no parallelization) is 24 h per 500 s of simulation time with a time step of 0.01 s.

It is often useful to save the output of running the simulation into a log file (porflo.log) for e.g. tracking possible errors. The log file contains the output from the initialization of the calculation and mesh generation as well as the numerical data that is printed each time-step and iteration. (Note that in long simulation runs, this can result in very large log files.) An example extract of the log file during one time-step is given below.

```
TIME-STEP:      248115
 ITERATION:           1
 sum_rhs:    1.054984973833832E-009
 max_rhs:    1.039213957942930E-010
 Scaled_VI:  6.462217459434782E-007

Liquid mass flow out [kg/s]:        -0.015
Vapor mass flow out  [kg/s]:         0.015
Possible breakdown, dot2:  3.136358206071463E-065
Possible breakdown, dot2:  3.136332456197489E-065
Possible breakdown, dot2:  4.818010376092232E-061
Eisen_CG_2 converged! iter, dotpo, normr:          62  5.502181000275844E-031
 4.760181068825489E-016

 TIME-STEP:      248115
 ITERATION:           2
 sum_rhs:    1.448735037399436E-009
 max_rhs:    4.790601964600738E-011
 Scaled_VI:  6.581468975922142E-007

Liquid mass flow out [kg/s]:        -0.015
Vapor mass flow out  [kg/s]:         0.015
Possible breakdown, dot2:  3.136337596420553E-065
Possible breakdown, dot2:  3.136303949479932E-065
Possible breakdown, dot2:  4.818010700972901E-061
Eisen_CG_2 converged! iter, dotpo, normr:          62  1.865354666523802E-032
 9.934854680066850E-017

 TIME-STEP:      248115
 ITERATION:           3
 sum_rhs:    3.690748165034168E-010
 max_rhs:    1.203304296482077E-011
 Scaled_VI:  1.908508566455768E-007

Liquid mass flow out [kg/s]:        -0.015
Vapor mass flow out  [kg/s]:         0.015
Possible breakdown, dot2:  3.136328851929896E-065
Possible breakdown, dot2:  3.136306655695237E-065
Possible breakdown, dot2:  4.818010570857190E-061
Eisen_CG_2 converged! iter, dotpo, normr:          59  3.963100837722369E-033
 4.372973551751188E-017

 TIME-STEP:      248115
 ITERATION:           4
 sum_rhs:    2.342841975851242E-010
 max_rhs:    5.817554988088447E-012
 Scaled_VI:  1.160823559634645E-007

Liquid mass flow out [kg/s]:        -0.015
Vapor mass flow out  [kg/s]:         0.015
Possible breakdown, dot2:  3.136324136644908E-065
Possible breakdown, dot2:  3.136313744276104E-065
Possible breakdown, dot2:  4.818010806491714E-061
Eisen_CG_2 converged! iter, dotpo, normr:          62  1.710564400908876E-033
 2.966982940673197E-017

 TIME-STEP:      248115
 ITERATION:           5
 sum_rhs:    1.562905868600654E-010
 max_rhs:    3.575445061548832E-012
 Scaled_VI:  7.486190141817209E-008
```

```
Liquid mass flow out [kg/s]:        -0.015
Vapor mass flow out  [kg/s]:         0.015
 Total liquid mass:   230.429381880713
 Total vapor mass:   1.801825595384579E-002

 Liquid mass inside the particle bed:   66.4658116372507
 Vapor mass inside the particle bed:   5.999543779945474E-003

 Liquid massflow out of the particle bed: -1.547510472827853E-002
 Vapor massflow out of the particle bed:   1.547624379678370E-002


 INPUT VALUES:
time_s       = 2481.16000 dtime_s      =    0.01000
time_min     = 2010.00000 time_max     = 2500.00000
Power [kW]   =   35.00000 glin         =  0.000E+00
p_system     =  0.100E+06 p_system_bar =  0.100E+01

Possible breakdown, dot2:  4.589679784884888E-062
Possible breakdown, dot2:  2.279529682505454E-109
Possible breakdown, dot2:  6.255284805235861E-057
Possible breakdown, dot2:  7.661167708683980E-099
Possible breakdown, dot2:  4.590908788373912E-062
Possible breakdown, dot2:  2.266957398103207E-109
Possible breakdown, dot2:  6.255151368848210E-057
Possible breakdown, dot2:  7.662559230645362E-099
Possible breakdown, dot2:  4.589842714228127E-062
Possible breakdown, dot2:  2.279738633534677E-109
Possible breakdown, dot2:  6.255237963332649E-057
Possible breakdown, dot2:  7.662661631610286E-099
Possible breakdown, dot2:  4.590213923079704E-062
Possible breakdown, dot2:  2.273915384433118E-109
Possible breakdown, dot2:  6.255182751701605E-057
Possible breakdown, dot2:  7.661627775431119E-099
Possible breakdown, dot2:  4.590208071977607E-062
Possible breakdown, dot2:  2.274002521603293E-109
Possible breakdown, dot2:  6.255154050626475E-057
Possible breakdown, dot2:  7.661009617219395E-099
Possible breakdown, dot2:  4.591071760542527E-062
Possible breakdown, dot2:  2.268180854958074E-109
Possible breakdown, dot2:  6.255205928786866E-057
Possible breakdown, dot2:  7.662697143800874E-099
Possible breakdown, dot2:  3.136319245473636E-065
Possible breakdown, dot2:  3.136323485404765E-065
Possible breakdown, dot2:  4.818011341200079E-061
Eisen_CG_2 converged! iter, dotpo, normr:         58  8.741532365392332E-031
 5.541497706632693E-016
```

Gnuplots scripts have been generated that can be used for follow-up of the residuals and heat transfer rates during the simulations. Plotting of the residuals in real time can be used to help detect convergence problems. Examples of residual and heat transfer plots are presented in Fig. 4 and Fig. 5. Heat transfer monitors are especially useful in detecting the contributions of different heat transfer mechanisms to the overall steady-state/transient behavior of the modelled system.

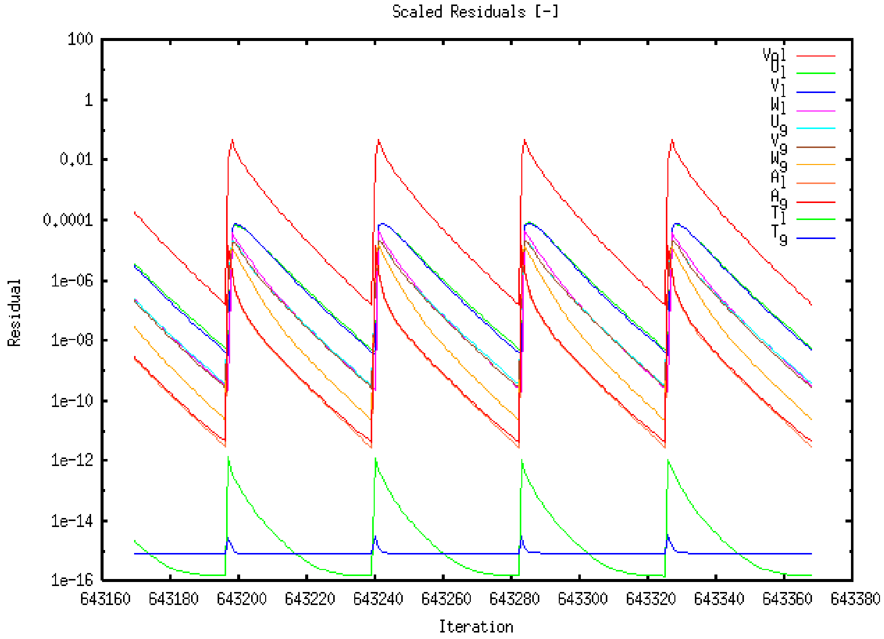


Fig. 4. Convergence of the calculation is monitored by plotting the residuals of the variables each time step.

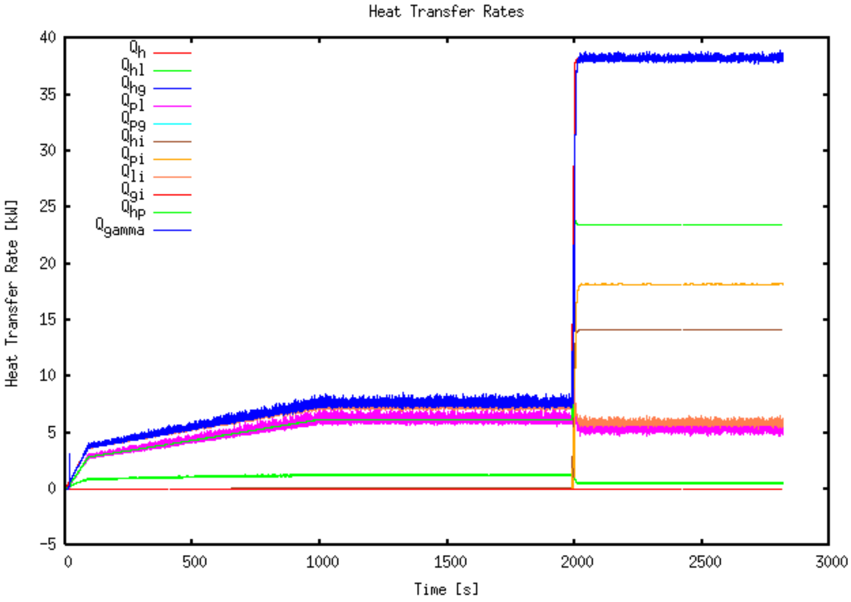


Fig. 5. Example of time evolution of the heat transfer rates in porous particle bed (a system which consists of heaters, solid particles, liquid and vapor that exchange heat).

In the particle bed simulations in which the aim is to detect dryout within the particle bed, it is useful to observe the void fraction by plotting the saturation profiles within the particle bed during the simulations. Saturation is the fraction of liquid within the pores, $s_l = 1.0 - \alpha$. The profiles may be generated by a simple routine that reads the void fraction history from the monitor file (monitos.txt) at selected time step intervals. Then, the void fraction values can be converted to values of saturation and plotted as a function of the height of the geometry. An example is presented in Fig. 6.

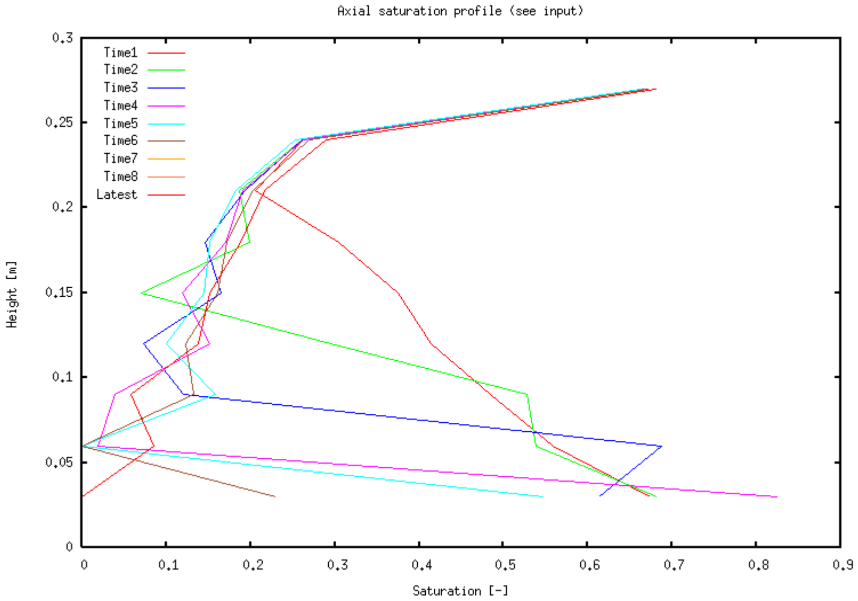


Fig. 6. Saturation profile in the simulation of a cylindrical particle bed.

5 Post-processing

The main output of the code is written in text files as described in the previous Section (for example porflo1234.txt) that contain the three-dimensional arrays of the simulation variables. In addition to this, the code prints several other output files. The output files are listed below:

1) Values of simulation variables (porflo.txt)
2) Log file (porflo.log)
3) User-defined monitor file (monitors.txt)
4) Heat transfer monitor file (heat.txt)
5) Mass flux monitor file (masses.txt)
6) Residual monitor file (residuals.txt)
7) Animation file (animation.bin)

The mass flux monitor file contains the mass flow rates across horizontal cross-sections of the modelled geometry. The heat transfer monitor file contains the total heat fluxes of the different heat transfer mechanisms present in the modeled system (example of which was plotted in Fig. 5). For post-processing purposes, however, it is more convenient to examine these variables through the "porflo.txt" output file. There are presently two options for post-processing and flow field visualization that directly utilize the main output file:

1) Matlab
2) StarNode

Matlab can be used for 2D plotting by simple script files (PlotFromRestart.m and ResidualPlot.m). However, more comprehensive visualization options are provided by the StarNode visualization tool. StarNode is based on the open-source graphics rendering engine Ogre3D (http://www.ogre3d.org/) and the QT user interface toolkit (http://qt.nokia.com/). Presently, StarNode is capable of rendering hundreds of thousands of nodes (Ilvonen et al. 2010). Various data sets can be visualized by the code, and 2D cross sections created from the 3D data. Vector data may also be plotted. Examples of creating plots and animation can be found in Appendix B. More information is found in the StarNode User's Manual (Inkinen, 2010).

Examples of 2D and 3D graphs generated using StarNode are shown in Fig. 7. The graphs represent the volumetrically heated conical COOLOCE particle bed inside a vessel that contains liquid and steam (the red volume in the top section of the grid). Velocity vectors of the gas phase in the same simulation are given in Fig. 8.

The time evolutions of different variables or their profiles at fixed simulation times, e.g. with respect to the height of the geometry, may be examined by using the "monitors.txt" data file. Time evolutions may be directly plotted and the other data may be sorted from the file according to the user's needs, similarly to the example in Fig. 6.

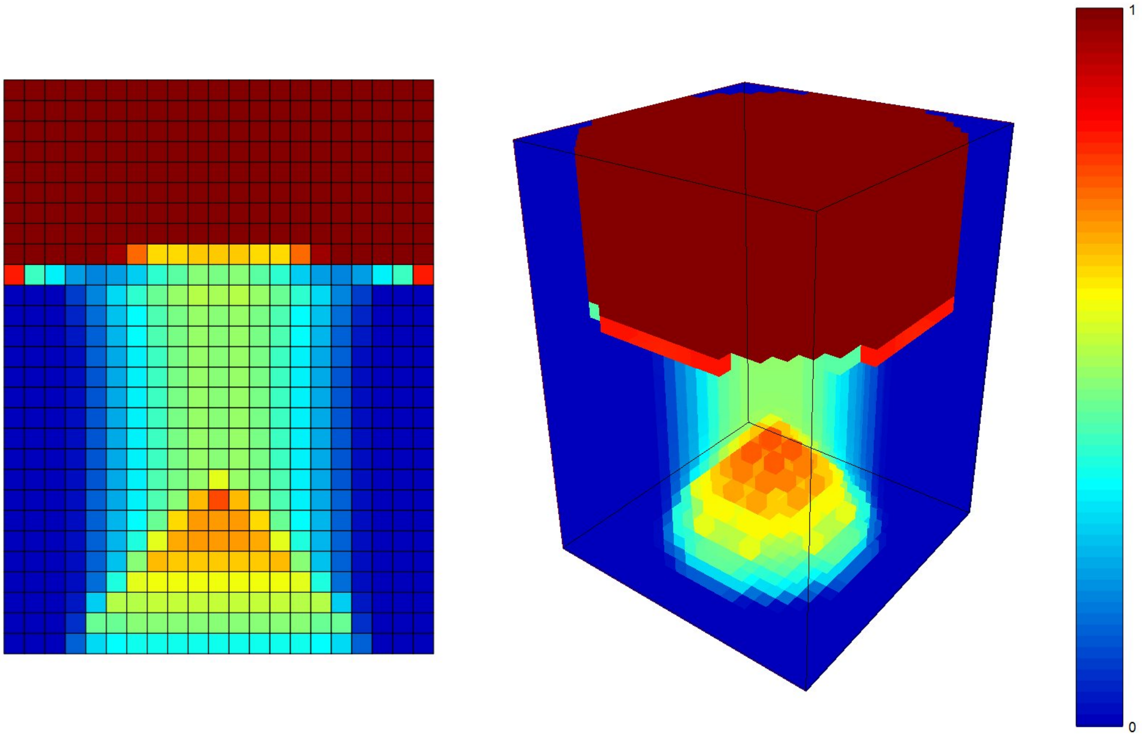


Fig. 7. Visualization by StarNode: 2D cross-section of void fraction in a particle bed simulation (left) and 3D maximum intensity projection of the void fraction in the same simulation (right).

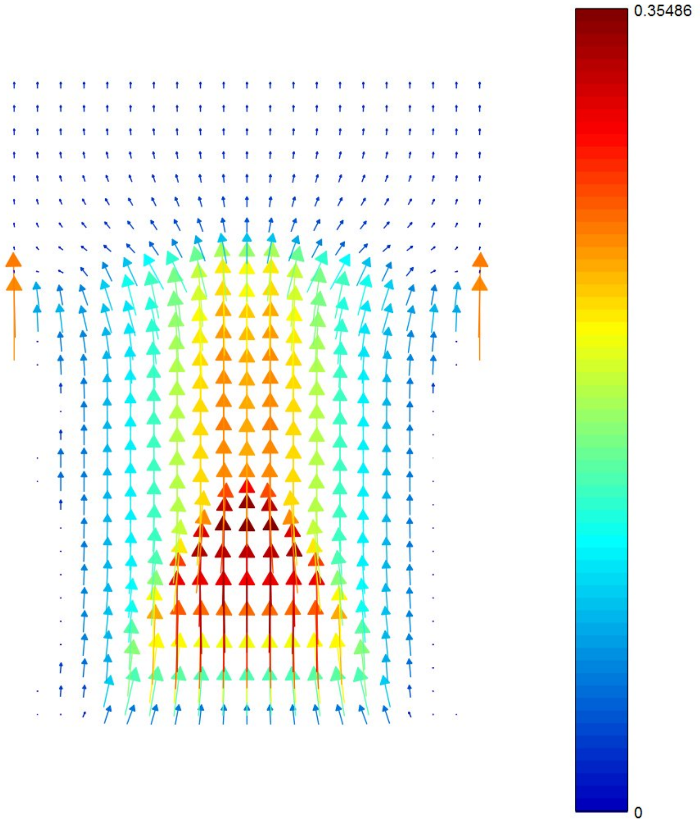


Fig. 8. StarNode vector image of vapor velocity in a particle bed simulation.

6 Concluding remarks

PORFLO is a 3D two-phase flow simulation code which utilizes the porous medium approach in the modelling of flows in complex geometries. The code is applied in the thermal hydraulic safety analysis of nuclear power plants. In recent years, the code has been under active development at VTT for the following applications: core debris coolability, secondary side of steam generators, flows across fuel bundles and the EPR reactor core. Extensions of the PORFLO applications to e.g. modelling of passive safety systems are planned in the future. Another long-term goal is to connect the code to the family of system codes and reactor core simulators (TRAB-3D-SMABRE) that have been developed by VTT.

Review of the solution methods and structure of the PORFLO code version of 2010 were presented, and general guidelines on how to set up and run PORFLO simulations were given. It was emphasized that the process of setting up simulations is application-specific and editing of the source code is often necessary. While this requires some skill in programming and understanding of the source code, it allows significant flexibility in modifying of the code to suit different applications and cases.

Examples of post-processing the simulation data by using the StarNode visualization tool were given. The presented input, output and visualization examples were based on the code version modified for the core debris coolability application in 2010.

References

Gnuplot homepage. http://www.gnuplot.info/ (July 20th, 2010).

Hovi, V. 2008. Calculations of Boiling Two-Phase Flow Using a Porous Media Model. Master's Thesis. Lappeenranta University of Technology. 112 p.

Hovi, V., Ilvonen, M. 2010. PORFLO Simulations of Loviisa Horizontal Steam Generator. VTT Research Report, VTT-R-01406-10, Espoo. 34 p.

Hovi V., Ilvonen, M. 2010. 3D PORFLO simulations of Loviisa steam generator. AER Symposium, September 20-24, 2010, Hanasaari, Espoo, Finland.

Hyytinen, K. 2001. Partikkelikeon jäähdytyskokeiden analysointi. Master's Thesis. Tampere University of Technology, 2001. 67 p.

Ilvonen, M., Hovi, V. 2009. The porous medium model PORFLO for 3D two-phase flow and its application to BWR fuel bundle simulations. Eija Karita Puska (Ed.). SAFIR2010 – The Finnish Programme on Nuclear Power Plant Safety 2007-2010, Interim Report. VTT Research Notes 2466, Helsinki, 2009. ISBN 978-951-38-7266-3. Pp. 131-141.

Ilvonen, M., Hovi, V., Inkinen, P. 2010. PORFLO development, applications and plans in 2008-2009. VTT Research Report, VTT-R-01414-10, Espoo. 28 p.

Inkinen, P. 2010. StarNode v0.4.0. User's Manual.

Miettinen, J., Karppinen, I. 2003. A Porous Media Calculation Applications for the Isolation Condenser Heat Transfer and Circulation. The 10th International Topical Meeting on Nuclear Reactor Thermal Hydraulics (NURETH-10). Seoul, Korea, October 5-9, 2003.

OGRE – Open Source 3D Graphics Engine. http://www.ogre3d.org/ (July 20th, 2010).

Patankar, S.V. 1980. Numerical Heat Transfer and Fluid Flow. New York, Hemisphere Publishing Corporation. 197 p. ISBN 0-89116-552-3.

Qt – Cross-platform application and UI framework. http://qt.nokia.com/ (July 20th, 2010).

Takasuo, E., Hovi, V., Ilvonen, M. 2011. PORFLO modelling of the coolability of porous particle beds. VTT Research Report, VTT-R-09376-10, Espoo, 2011. 41 p.

Takasuo, E., Kinnunen, T., Pankakoski, P.H., Holmström, S. 2010. Description of the COOLOCE test facility – Conical particle bed. VTT Research Report VTT-R-08956-10. Espoo, 2010. 18 p.

APPENDIX A. List of PORFLO updates for the porous particle bed application in 2010

1. The particle bed geometry routine for the generation of the COOLOCE test facility model has been implemented (3/2010).

2. The heat transfer solution routine has been modified so that the heating power may be given as an input value instead of surface temperatures (4/2010).

3. A selection of heat transfer models for the particle bed application has been included (4/2010). The models utilized in the MEWA (WABE) code (Bürger et al. 2006) were used as a starting point for the model implementation.

4. A selection of particle bed friction models has been included (4/2010). The models are based on well-known approaches for modelling porous media.

5. The particle bed geometry routine has been modified for investigations of cylindrical geometries (6/2010).

6. Possibilities for writing time dependent output (monitors) of selected variables during the simulations have been included as an option in the input file (6/2010).

7. Heat transfer models for pure conduction that include the calculation of effective thermal conductivity of porous medium have been implemented in order to examine the heat transfer inside and between different types of solids (8/2010).

8. A possibility for explicit temperature solution which improves the convergence of some simulation cases has been included (8/2010).

9. Several additional output options such as the heat transfer rates between the phases have been included (8/2010).

10. The possibility to create animations of the simulation output by using the StarNode visualization tool has been included (8/2010).

11. The possibility for a flexible selection of friction models for different regions of the computational grid has been included in the input file (9/2010)

Reference

Bürger, M., Buck, M., Schmidt, W., Widmann, W. 2006. Validation and application of the WABE code: Investigations of constitutive laws and 2D effects on debris coolability. Nuclear Engineering and Design 236 (2006), p. 2164-2188.

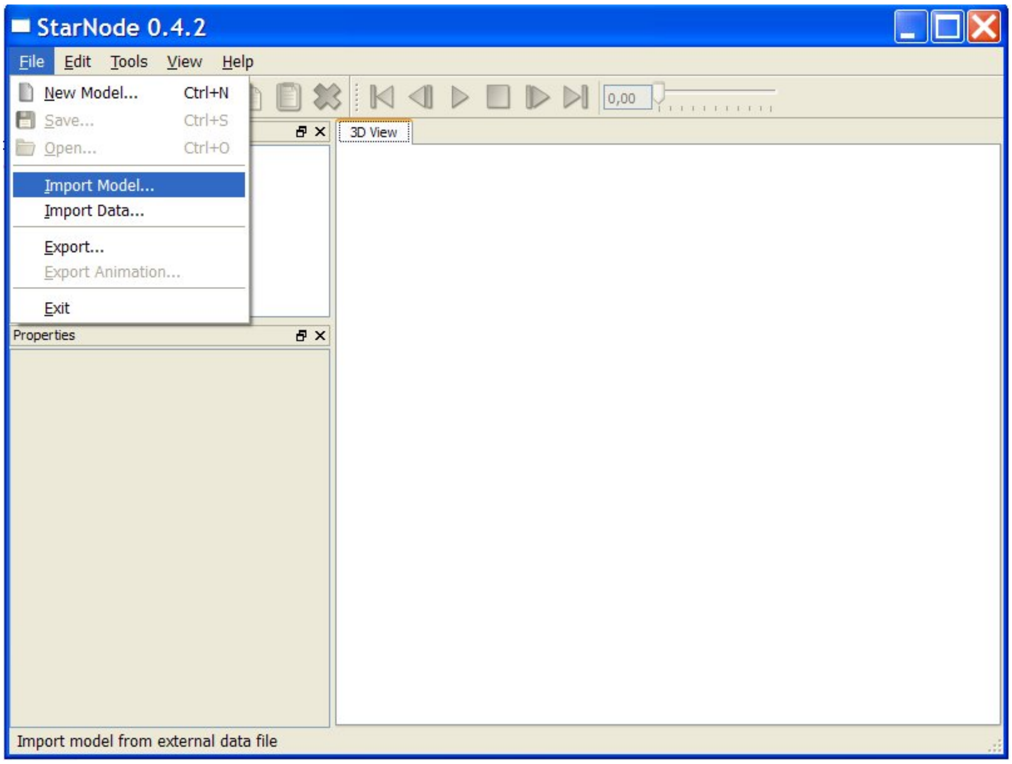
APPENDIX B. Examples of creating plots and animations in StarNode

The following gives step-by-step instructions on how to generate 2D plots, 3D plots and animations in StarNode. The data of a PORFLO particle bed simulation is used to demonstrate the visualization process. A Cartesian grid with the dimensions of 21 x 21 x 28 computational cells is used in the example simulation data. Note that application-specific choices may have to be made depending on the type of the data and the simulation code that has been used to generate it.

Creating a 2D Plot

1. Start the StarNode software.

2. In the menu, select **File → Import Model …** and select the appropriate file (in this case “porflo.txt”). Select the format of the data file (PORFLO Restart File) and click **OK** to read the data file.

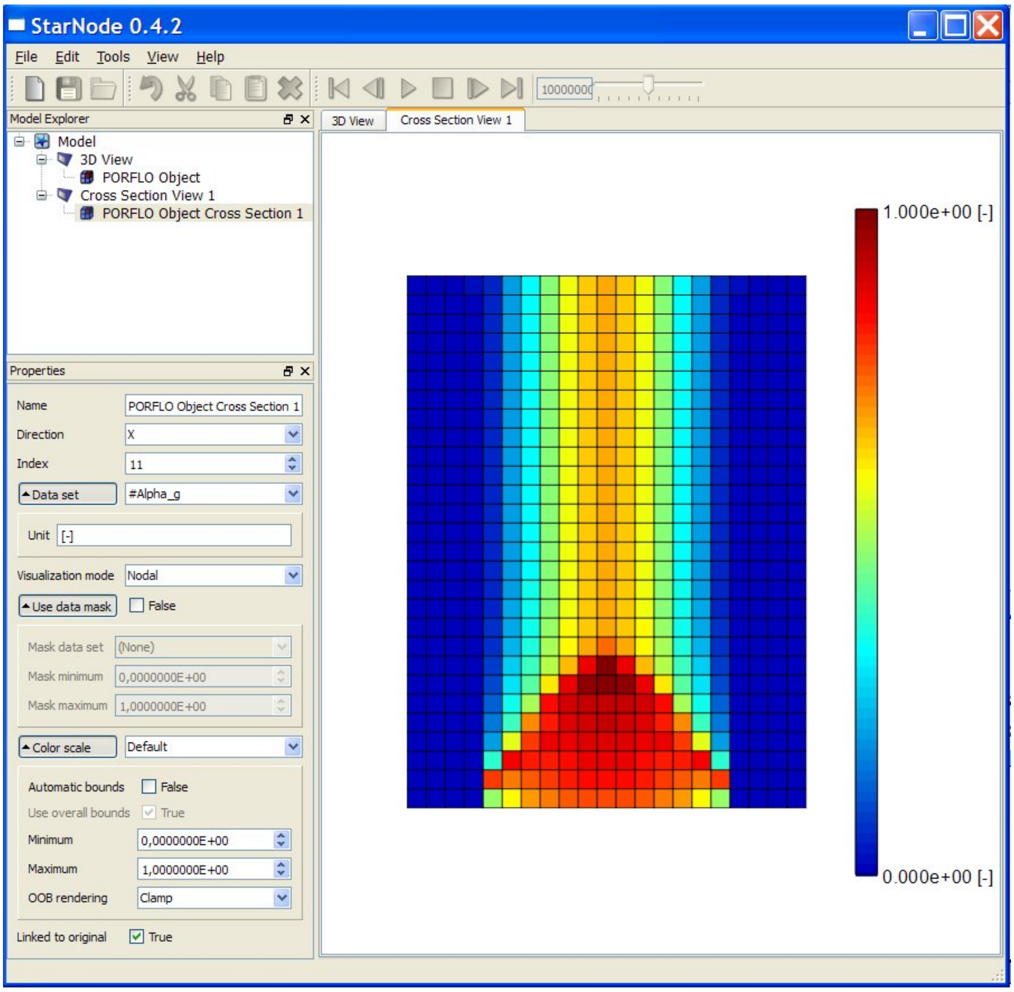


Import model from external data file

3. In the **Model Explorer** window on the left, select PORFLO Object, click the right mouse button and select **Send To 2D View**. A 2D cross section of the data is displayed. In the **Model Explorer**, click **PORFLO Object Cross-Section 1** to access the 2D Plot options. These are shown in the **Properties** window below the **Model Explorer**.

4. The 2D Plot can be zoomed in and out by the mouse wheel.

5. In the **Direction** drop-down menu, select the direction axis (X,Y,Z) for the cross-section that you wish to visualize. In the **Index** menu, select the index of the grid cells in the direction to be visualized. In this case, we choose the central plane in the X direction whose index is 11 and get a vertical cross-section of the modeled geometry. In the **Data Set** menu, select the variable to plot. Here, we take the void fraction (Alpha_g) as the examined variable.



Name: PORFLO Object Cross Section 1
Direction: X
Index: 11
Data set: #Alpha_g
Unit: [-]
Visualization mode: Nodal
Use data mask: ☐ False
Mask data set: (None)
Mask minimum: 0,0000000E+00
Mask maximum: 1,0000000E+00
Color scale: Default
Automatic bounds: ☐ False
Use overall bounds: ☑ True
Minimum: 0,0000000E+00
Maximum: 1,0000000E+00
OOB rendering: Clamp
Linked to original: ☑ True

6. The **Visualization mode** menu presents options for interpolation and drawing contour lines. Here, we do not wish to interpolate the simulation data and select the **Nodal** mode. The **Use data mask** button provides options for masking the data values based on selected minimum and maximum values.

7. The **Color scale** button provides options for changing the default color scale e.g. to grayscale, and for setting overall (global) bounds for the color scale. Here, we take an overall bound from 0.0 to 1.0 for the void fraction by selecting the **Use overall bounds** option and setting the **Minimum** to 0.0 and **Maximum** to 1.0 as has been done in the figure above. The global scale option is especially handy in creating animations because now the color scale is not automatically adjusted between the different screens.

8. You can switch between 2D and 3D views by clicking the **3D View** and **Cross Section 1 View** tabs on the top of the current view.

Creating a 3D Plot

1. Start the StarNode software.

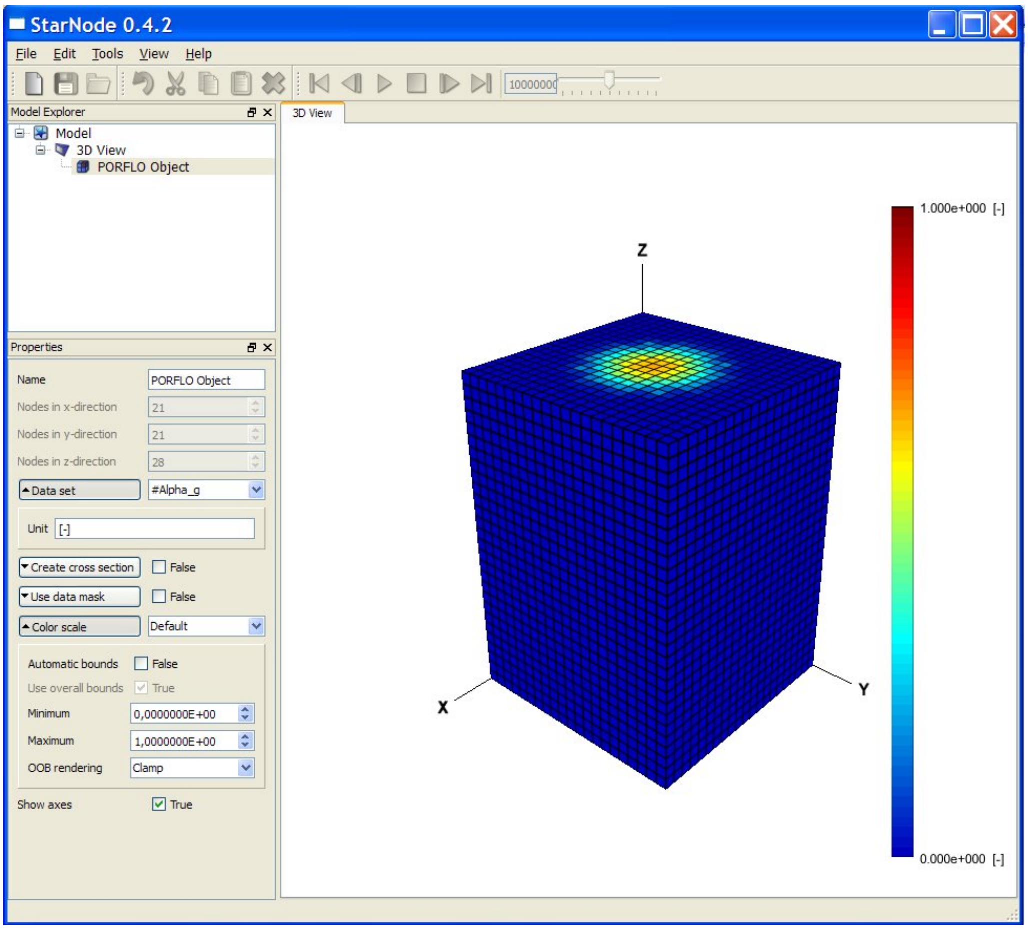
2. In the program menu, select **File** → **Import Model …** and select the appropriate file (in this case “porflo.txt”). Select the format of the data file (PORFLO Restart File) and click **OK** to read the data file.

3. To access the options of the **Properties** window for the 3D Plot, select (left-click) **PORFLO Object** in the **Model Explorer** window on the left.

4. The 3D View can be zoomed in and out by the mouse wheel and rotated by dragging the mouse while pressing the right mouse button. Rotation can also be done by the arrow keys.

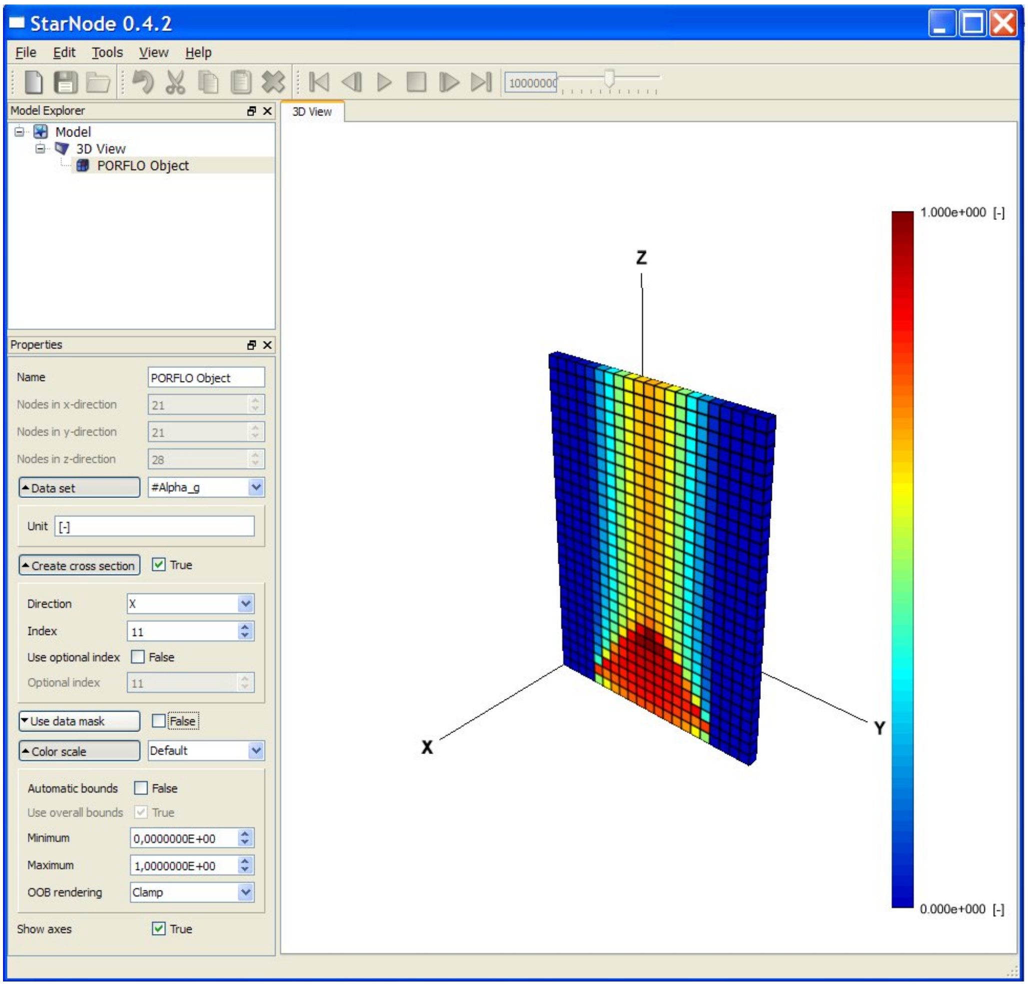
5. Click the **Data set** button and select the variable to be plotted from the drop-down menu. Here, we take the void fraction (Alpha_g) as the example variable.

6. As in the 2D Plot example, we select overall bounds for the color scale by clicking the **Color scale**, selecting the **Use overall bounds** and giving 0.0 for the **Minimum** limit and 1.0 for the **Maximum** limit. The axes may be turned on and off from the **Show axes** selection. The resulting 3D view is shown in the figure below.



Name	PORFLO Object
Nodes in x-direction	21
Nodes in y-direction	21
Nodes in z-direction	28
Data set	#Alpha_g
Unit	[-]
Create cross section	□ False
Use data mask	□ False
Color scale	Default
Automatic bounds	□ False
Use overall bounds	☑ True
Minimum	0,0000000E+00
Maximum	1,0000000E+00
OOB rendering	Clamp
Show axes	☑ True

7. The **Create cross section** button provides the possibility to view cross sections of the nodal data in 3D. A “slice” of the geometry is shown in the 3D View according to the selected axis in the **Direction** menu and the index of the grid cells in the **Index** menu. In the example seen in the figure below, we use index 11 to see the central plane of the geometry.



Name	PORFLO Object
Nodes in x-direction	21
Nodes in y-direction	21
Nodes in z-direction	28
Data set	#Alpha_g
Unit	[-]
Create cross section	☑ True
Direction	X
Index	11
Use optional index	☐ False
Optional index	11
Use data mask	☐ False
Color scale	Default
Automatic bounds	☐ False
Use overall bounds	☑ True
Minimum	0,0000000E+00
Maximum	1,0000000E+00
OOB rendering	Clamp
Show axes	☑ True

8. The **Use data mask** button provides options for masking the data values based on selected minimum and maximum values.

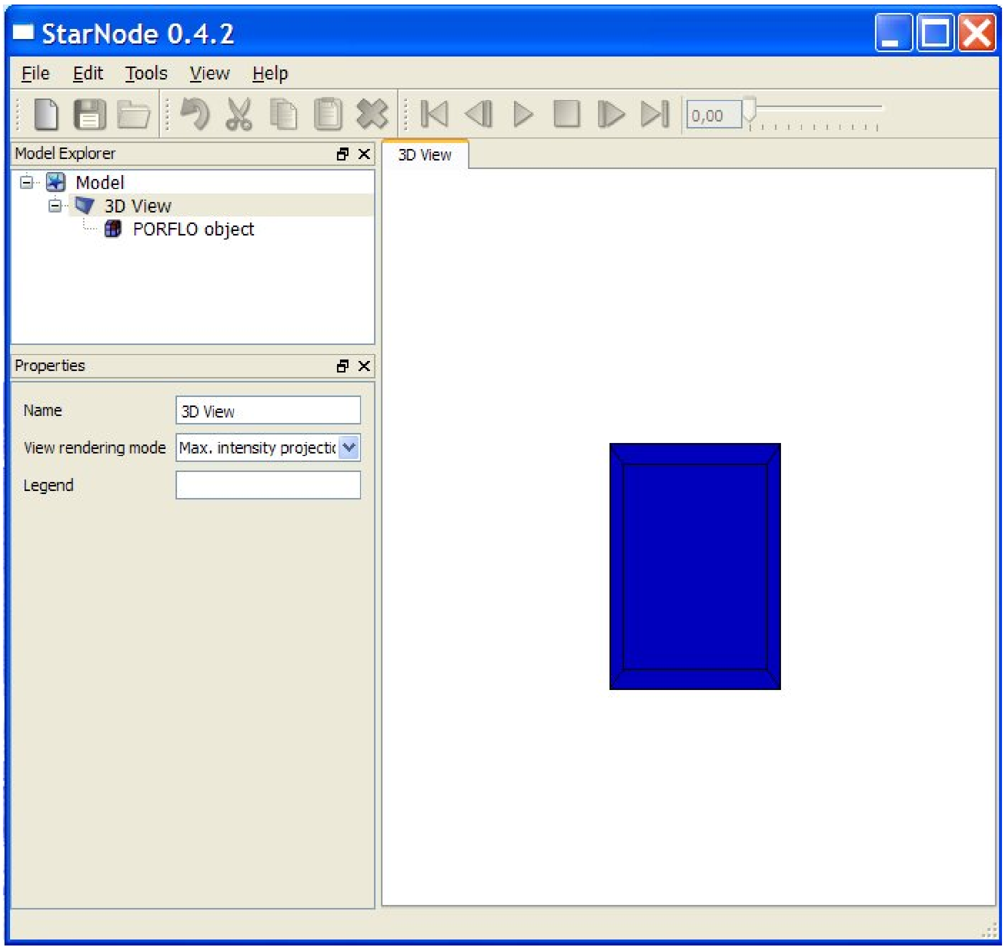
Creating an animation

1. Start the StarNode software.

2. In the program menu, select **File** → **Import Model …** and select the appropriate binary file (in this case “animation.bin”). Select the format of the data file (PORFLO Binary Data) and click **OK** to read the file.

3. Animations can be viewed in both 2D and 3D. The **Model Explorer** window and the **Properties** window are similar as in creating 2D and 3D Plots (see the previous sections). In this example, we create an animation that shows the void fraction as the maximum intensity projection of the variable.

4. Left-click **3D View** in the **Model Explorer**. In the **View rendering mode** drop-down menu, select **Maximum intensity projection**.



Name: 3D View

View rendering mode: Max. intensity projectic

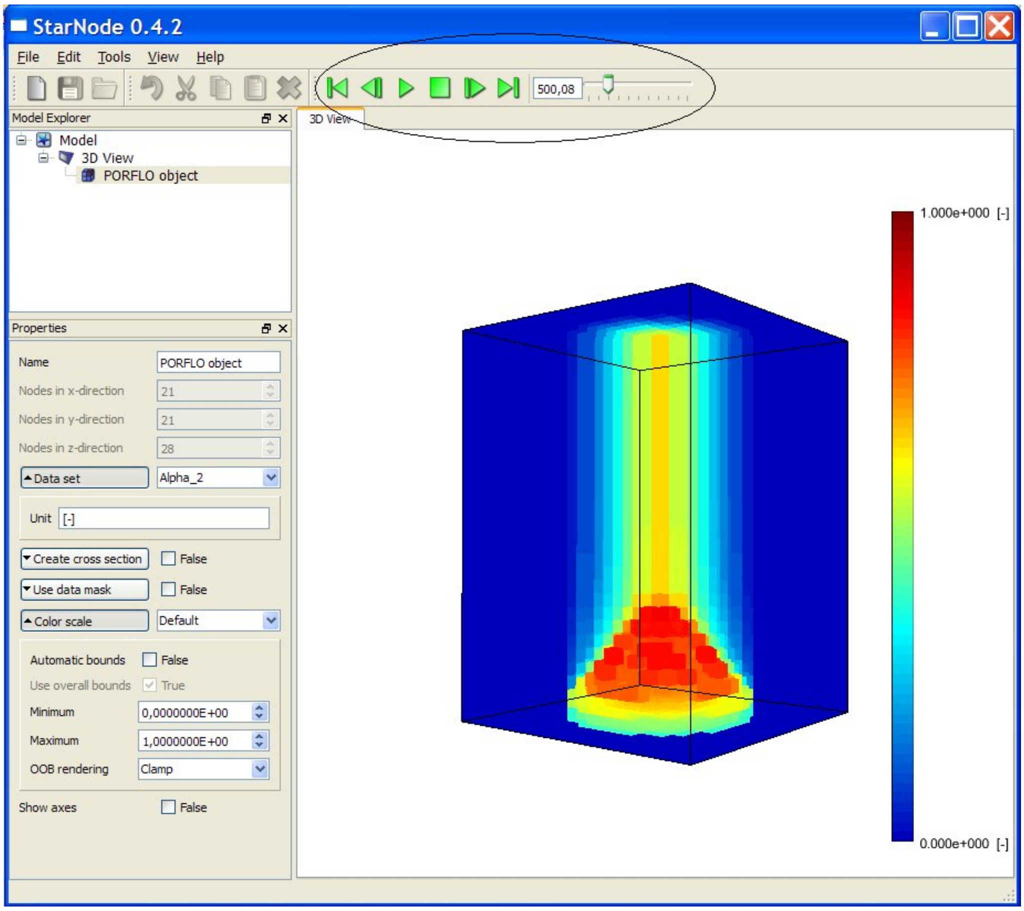
Legend:

5. Left-click the **PORFLO object** in the **Model Explorer**, left-click the **Data set** in the **Properties** window and select the void fraction in the drop-down menu (Alpha_g).

6. Select the global bounds for the color scale by clicking the **Color scale** button and enabling the **Use overall bounds** selection. Give 0.0 for the **Minimum** limit and 1.0 for the **Maximum** limit. The axes may be turned on and off from the **Show axes** selection.

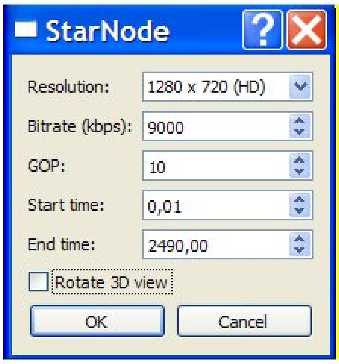
7. If necessary, zoom the view by the mouse wheel and rotate by dragging the mouse while holding the right button.

8. The animation control is found in the toolbar above the 3D View. Use the control buttons to play, stop, rewind and forward animations.



Name: PORFLO object
Nodes in x-direction: 21
Nodes in y-direction: 21
Nodes in z-direction: 28
Data set: Alpha_2
Unit: [-]
Create cross section: □ False
Use data mask: □ False
Color scale: Default
Automatic bounds: □ False
Use overall bounds: ☑ True
Minimum: 0,0000000E+00
Maximum: 1,0000000E+00
OOB rendering: Clamp
Show axes: □ False

9. Exporting animations to MPEG files is done by selecting **File → Export animation…** Type the file name, click **Save** and select the output resolution, bitrate, GOP and the start and end time of the selection to be exported. Click **OK.**



Resolution: 1280 x 720 (HD)
Bitrate (kbps): 9000
GOP: 10
Start time: 0,01
End time: 2490,00
□ Rotate 3D view