NURESAFE WP1.1
TESTBED FOR INTEGRATED COUPLING AND UNCERTAINTY QUANTIFICATION METHODS

SUBCHANFLOW sensitivity analysis with URANIE

J. Jimenez, N. Trost, V. Sanchez

Presented by J. Jimenez

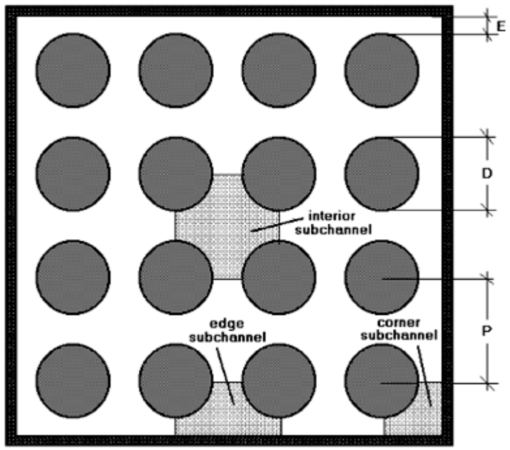
victor.sanchez@kit.edu
or
javier.jimenez@kit.edu

Outline

- **Summary from the last meeting in HZDR**
  - SUBCHANFLOW code
  - URANIE software
  - BFBT benchmark used for the analysis
  - URANIE measurement scripts
  - Results for BFBT single phase pressure drop analysis
  - Results for BFBT two phase pressure drop analysis
- **New scripts for transient analysis**
- **Application to the O2 -1999 Feedwater transient**
- **Conclusion and Outlook**

SUBCHANFLOW – a KIT TH sub-channel code

- Sub-channel fuel assembly simulations
- Fuel assembly or “channel wise” whole core simulations
- Steady state and transient solution
- Available fluids: water, lead, lead-bismuth sodium, helium, air
- Works with SI units
- Flexible geometry definition (square, hexagonal)
- Completely programmed in Fortran 95
- Can be used with WINDOWS or LINUX
- Dynamic memory management
- Modular structure, keyword and table oriented input



SUBCHANFLOW Features

SUBCHANFLOW
KIT Subchannel Code

HZDR Dresden
- Simulation of single and two phase flow tests

RWTH Aachen
- INBK: Simulation of gas cooled reactors for transmutation (AGATE Project)

TÜV Nord
- LWR applications of SUBCHANFLOW (Testing Phase)

National

Use of SUBCHANFLOW at KIT
- EU THINS Project (Lead/Bismuth)
- EU CDT Project (Lead)
- Multi-physics coupling with
  - MCNP5, SERPENT
  - DYN3D-SP3 (transport)
  - COBAYA3 (Diffusion)
- International Benchmarks e.g. BFBT, PSBT
- Code improvement and validation:
  - (Internships, Bachelor, MSc, PhD)

EU JRC IE Petten
- Gen-IV reactors

UPM/ Nuclear Engineering Chair
- EU ESFR Project
- Coupling with COBAYA3

DNC / Delft
- EU HPMC Project
- SCF coupling with time-dependent Monte Carlo

CIEMAT/ Nuclear Safety Department
- SCF / MCNPX Coupling

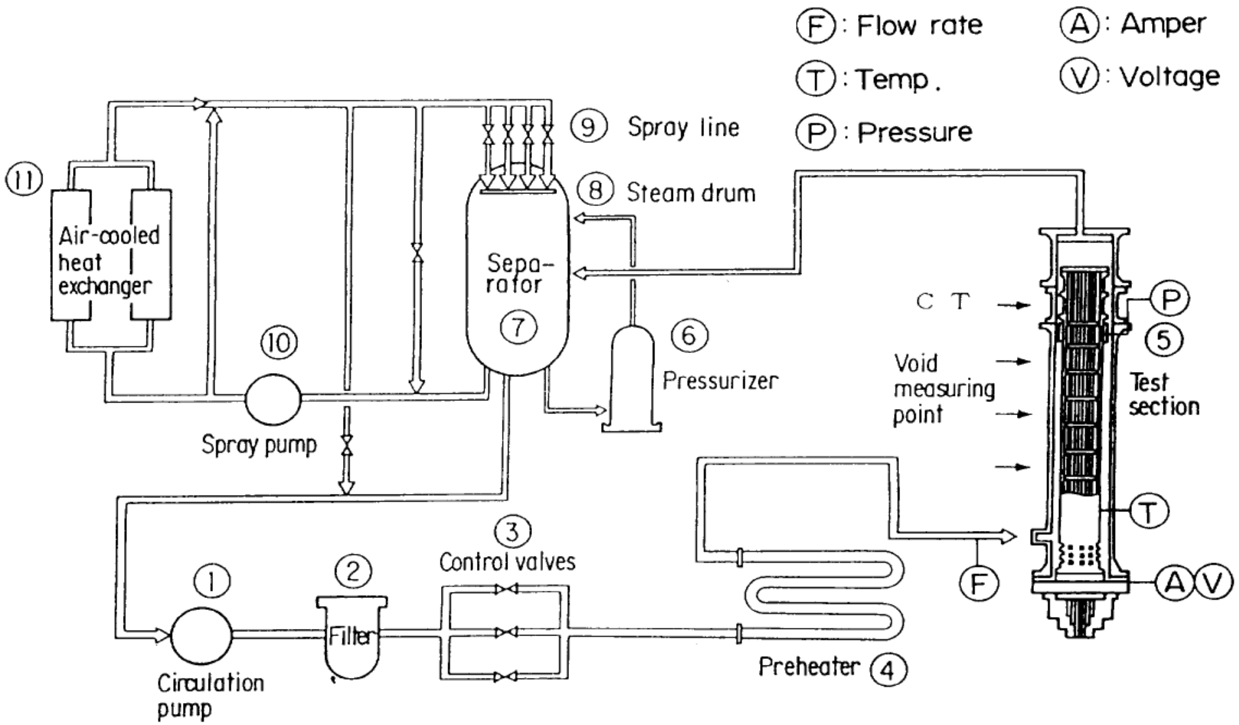
International

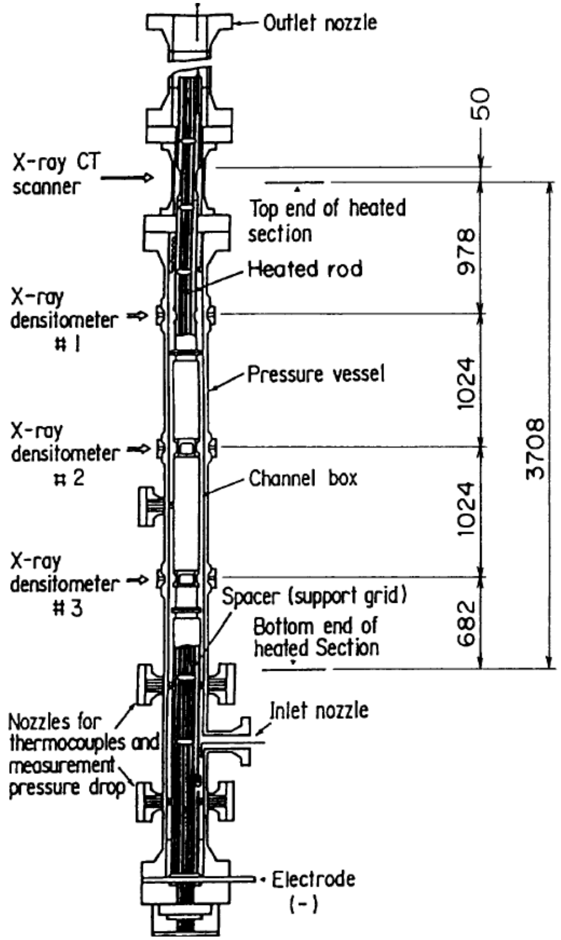
URANIE Software

- URANIE is a software dedicated to uncertainty and optimization. It allows to perform studies on uncertainty propagation, sensitivity analysis or model calibration in an integrated environment, based on ROOT, a software developed at CERN for particle physics data analysis. As a result, URANIE benefits from the numerous features of ROOT, among which:
  - a C++ interpreter (CINT)
  - a Python interface (PyROOT)
  - access to SQL databases
  - many advanced data visualization features
- URANIE training course attended 2-4th April 2013 in Saclay.

BFBT benchmark Phase II, Exercise 0

- **NUPEC BFBT experimental facility**





- **Steady state pressure drop measurements where used for the analysis**

Sensitivity study parameters

- BFBT single phase P70001 – P70012 cases for pressure drop analyses
- BFBT two phase P60001 – P60012 cases for pressure drop analyses
- 100 SUBCHANFLOW simulation runs per test scenario using URANIE
- Normal and Uniform law for boundary parameter distribution, see Table below

No.	Parameter	Range	Distribution
1	Outlet pressure	± 1.0 %	Normal
2	Mass flow rate	± 1.0 %	Normal
3	Inlet temperature	± 1.5 K	Uniform
4	Power (only P600xx)	± 1.5 %	Normal

Script for SUBCHANFLOW run

```
void BFBT_P6x(Int_t nS = 100) {

  TDataServer *tds = new TDataServer("tdsSCF", "BFBT_P6x");

  TNormalDistribution *tnp = new TNormalDistribution("OutletPressure", 7.16e6, 71600.0);  // +-1%
  TNormalDistribution *tnf = new TNormalDistribution("MassFlowRate", 5.61, 0.0561);       // +-1%
  TUniformDistribution *tut = new TUniformDistribution("InletTemperature", 276.3, 279.3); // +-1.5K
  TNormalDistribution *tnl = new TNormalDistribution("Power", 1.951e6, 29265.0);          // +-1.5%

  tnp->setBounds(7.0884e6, 7.2316e6);
  tnf->setBounds(5.5539, 5.6661);
  tnl->setBounds(1.921735e6, 1.980265e6);

  tds->addAttribute(tnp);
  tds->addAttribute(tnf);
  tds->addAttribute(tut);
  tds->addAttribute(tnl);

  TString sFileName = TString("input.txt");
...
```

Specify the number of simulation runs

Create DataServer to hold all informations

Specify the distribution for each parameter

Set upper and lower bound to avoid non-physical values

Fill up the DataServer

Specify the input file of SUBCHANFLOW to be parsed by URANIE and replaced by the sampled random data

Script for SUBCHANFLOW run

```
...
  tds->getAttribute("OutletPressure")->setFileKey(sFileName, "exit_pressure");
  tds->getAttribute("MassFlowRate")->setFileKey(sFileName, "inlet_flow_rate");
  tds->getAttribute("InletTemperature")->setFileKey(sFileName, "inlet_temperature");
  tds->getAttribute("Power")->setFileKey(sFileName, "total_power");

  TSampling *sampling = new TSampling(tds, "lhs", nS);
  sampling->generateSample();

  TOutputFileKey *fout = new TOutputFileKey("output.txt");

  TAttribute *avgpres = new TAttribute("total_ax_pres_loss_uranie");
  avgpres->setDefaultValue(-200.0);
  fout->addAttribute(avgpres);

  TCode *mycode = new TCode(tds, "SUBCHANFLOW");
  mycode->addOutputFile(fout);

  TLauncher *tlch = new TLauncher(tds, mycode);
  tlch->setSave();
  tlch->setClean();
  tlch->setWorkingDirectory(gSystem->Getenv("PWD") + TString("/tmpUranie/SCF"));
  tlch->setVarDraw("MassFlowRate:total_ax_pres_loss_uranie","","");
...
```

Give URANIE the strings of SUBCHANFLOW input to be changed

Generate the random data

Specify the output file to extract the data for post-processing

Tell URANIE which value has to be extracted from SCF output and initialize it with a non-physical value for easier error detection

Initiate a SUBCHANFLOW simulation run

Script for SUBCHANFLOW post-processing

```
 TCanvas *Canvas = new TCanvas("c1", "Graph",5,64,1270,667);
  c1->Divide(2, 2);
  c1->cd(1);
  tlch->run();
  c1->cd(3);
  tds->draw("OutletPressure:total_ax_pres_loss_uranie");
  c1->cd(2);
  tds->draw("InletTemperature:total_ax_pres_loss_uranie","","");
  c1->cd(4);
  tds->draw("Power:total_ax_pres_loss_uranie","","");

  TCanvas *Canvas2 = new TCanvas("c2", "Graph",5,64,1270,667);
  tds->draw("total_ax_pres_loss_uranie");

  tds->exportData("BFBT_P6x_Sampling.dat");

}
```

Visualize the output of URANIE

Plot a histogram of the output data

Export the output as well as the sampled random numbers into a file for post-processing.

Script for SUBCHANFLOW statistics

```
{
  using namespace URANIE::DataServer;
  using namespace URANIE::Sampler;

  TDataServer *tds = new TDataServer();

  tds->fileDataRead("BFBT_P6x_Sampling.dat");

  tds->addAttribute("OutletPressure", "OutletPressure");
  tds->addAttribute("MassFlowRate", "MassFlowRate");
  tds->addAttribute("InletTemperature", "InletTemperature");
  tds->addAttribute(„Power", „Power");
  tds->addAttribute("AxialPressureLoss", "total_ax_pres_loss_uranie");

  tds->computeStatistic();
  tds->computeCorrelationMatrix()->Print();

  std::cout << "mean: " << tds->getAttribute("AxialPressureLoss")->getMean()    << std::endl;
  std::cout << "min:  "  << tds->getAttribute("AxialPressureLoss")->getMinimum() << std::endl;
  std::cout << "max:  "  << tds->getAttribute("AxialPressureLoss")->getMaximum() << std::endl;
  std::cout << "std:  "  << tds->getAttribute("AxialPressureLoss")->getStd()     << std::endl;
}
```

Read in an URANIE file that contains sampled random data

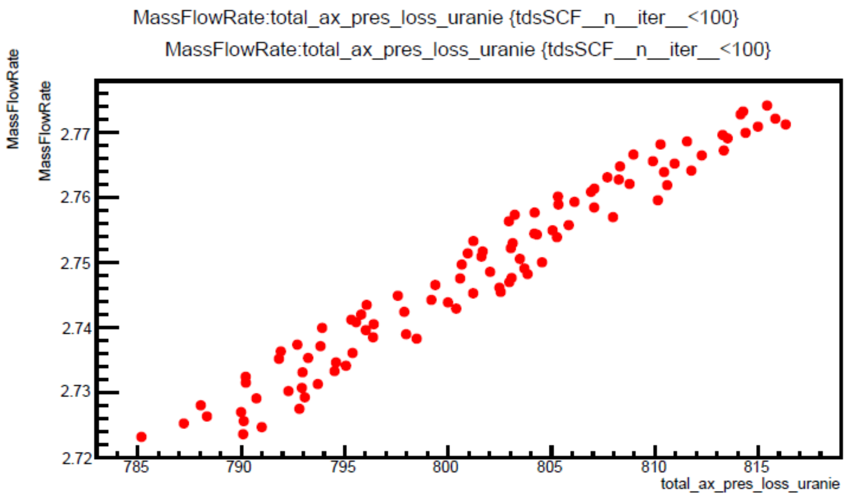
Fill up the DataServer structure

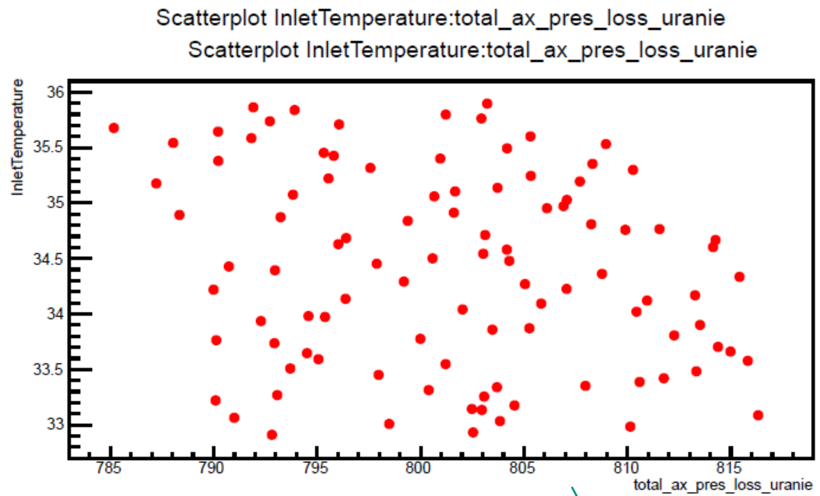
Compute the statistics and print the correlation matrix

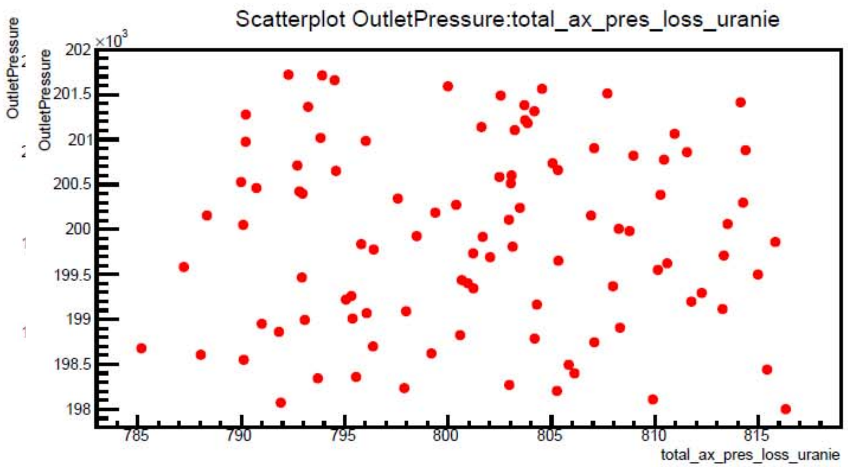
Print mean, min, max, and standard deviation

Visualization for total axial pressure loss

- URANIE visualization output for P70001 single phase BFBT benchmark



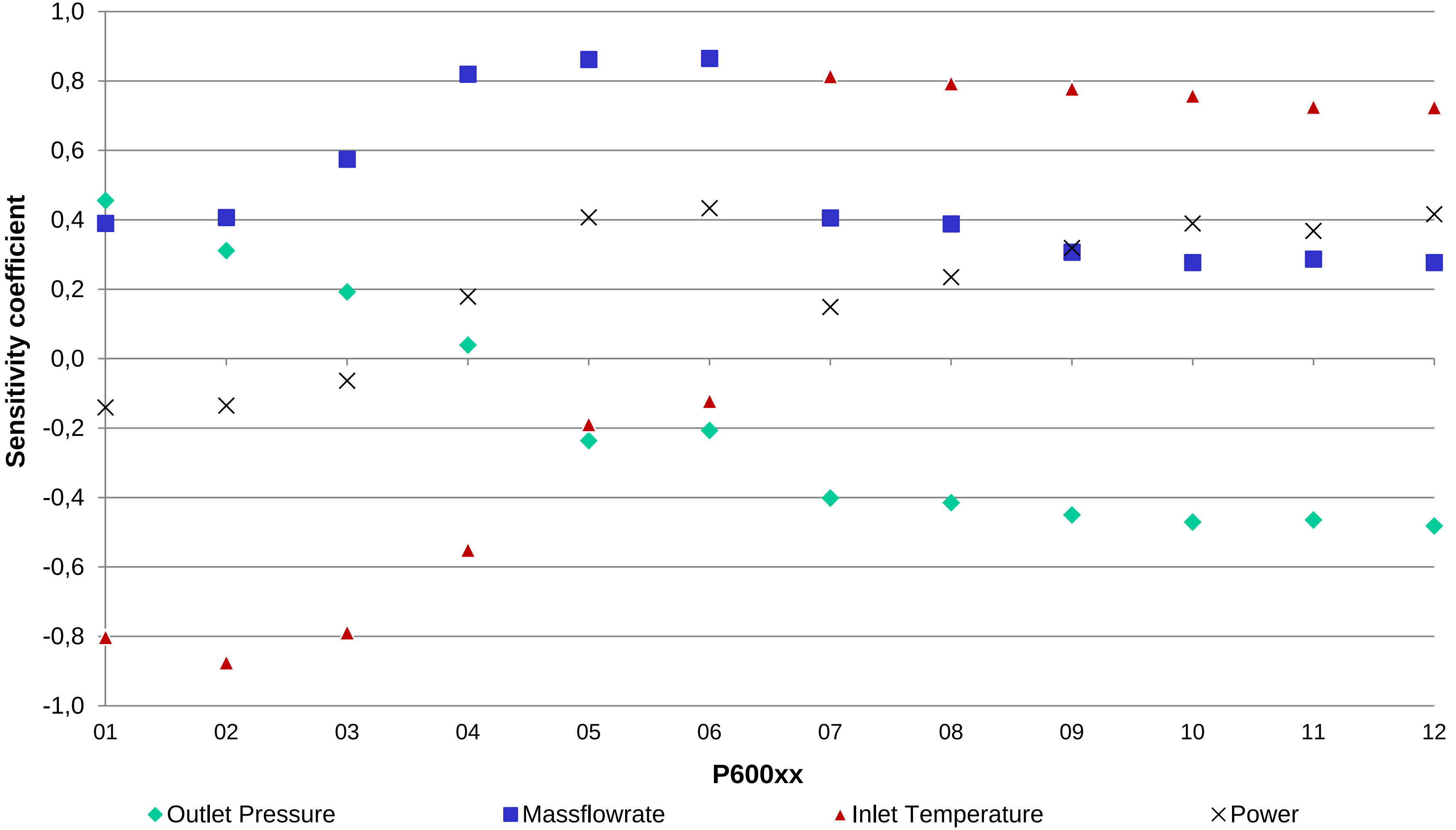




Correlation of massflow rate, inlet temperature and outlet pressure with the total axial pressure loss.
Correlation matrix can be also extracted

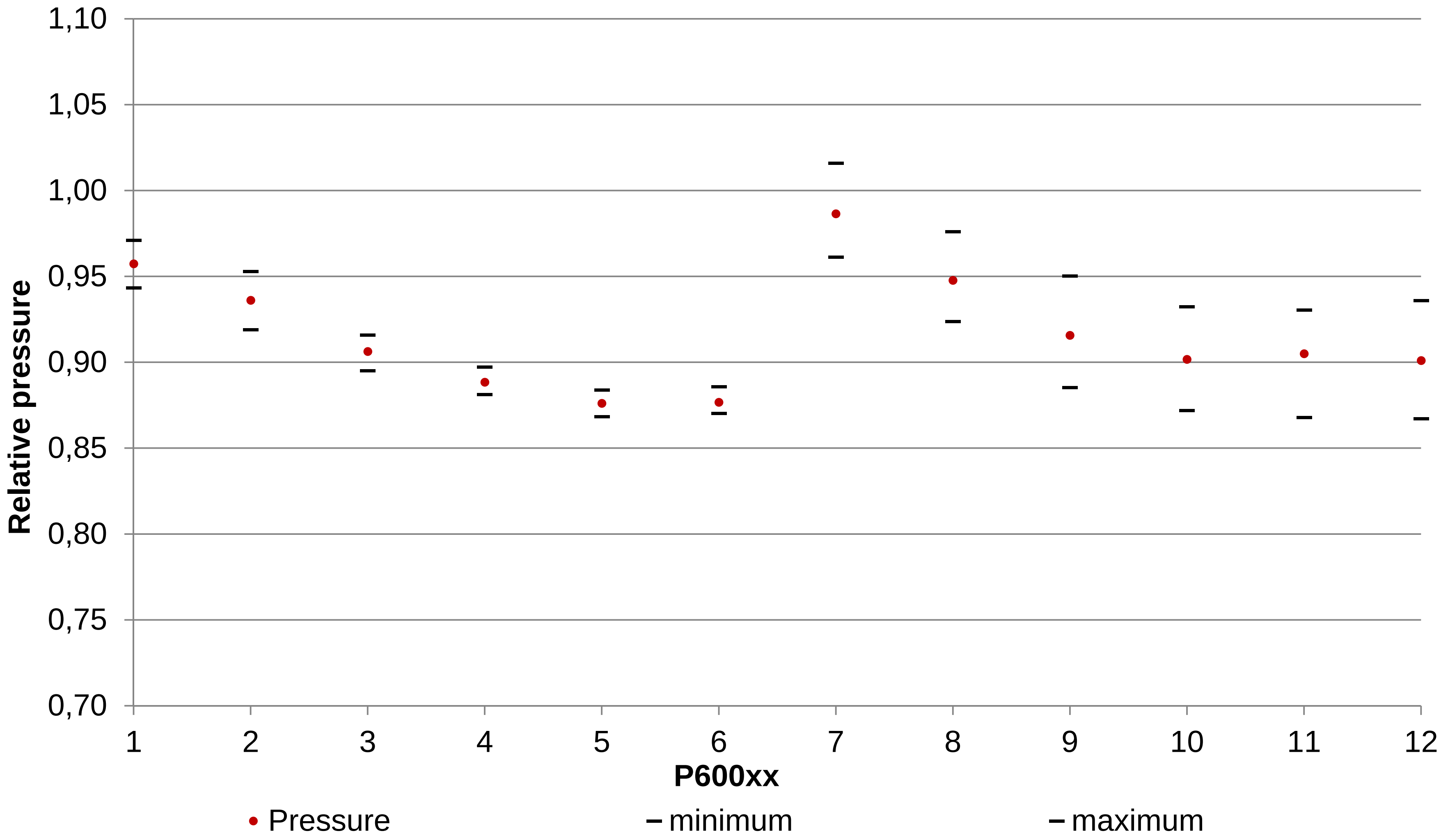
BFBT Two Phase P60001 – P60012

- Sensitivity coefficient for BFBT Two Phase P60001 – P60012 pressure loss analysis



BFBT Two Phase P60001 – P60012

- Relative pressure for BFBT Two Phase P60001 – P60012 pressure loss analysis to the experimental data



New scripts for transient analysis

- **Up to this point it was reported in the last meeting**
- **Study of the pump trip and turbine trip transient scenario of the BFBT benchmark**
- **Range of variation ± 1.0 % only for academic purposes**

No.	Parameter	Range	Distribution
1	Outlet pressure	± 1.0 %	Normal
2	Mass flow rate	± 1.0 %	Normal
3	Inlet temperature	± 1.0 %	Uniform
4	Power	± 1.0 %	Normal
5	Cladding Wall Roughness	± 1.0 %	Normal
6	Spacer grid pressure drop coefficient	± 1.0 %	Normal
7	Flow Area	± 1.0 %	Normal
8	Wetted Perimeter	± 1.0 %	Normal

New scripts for transient analysis

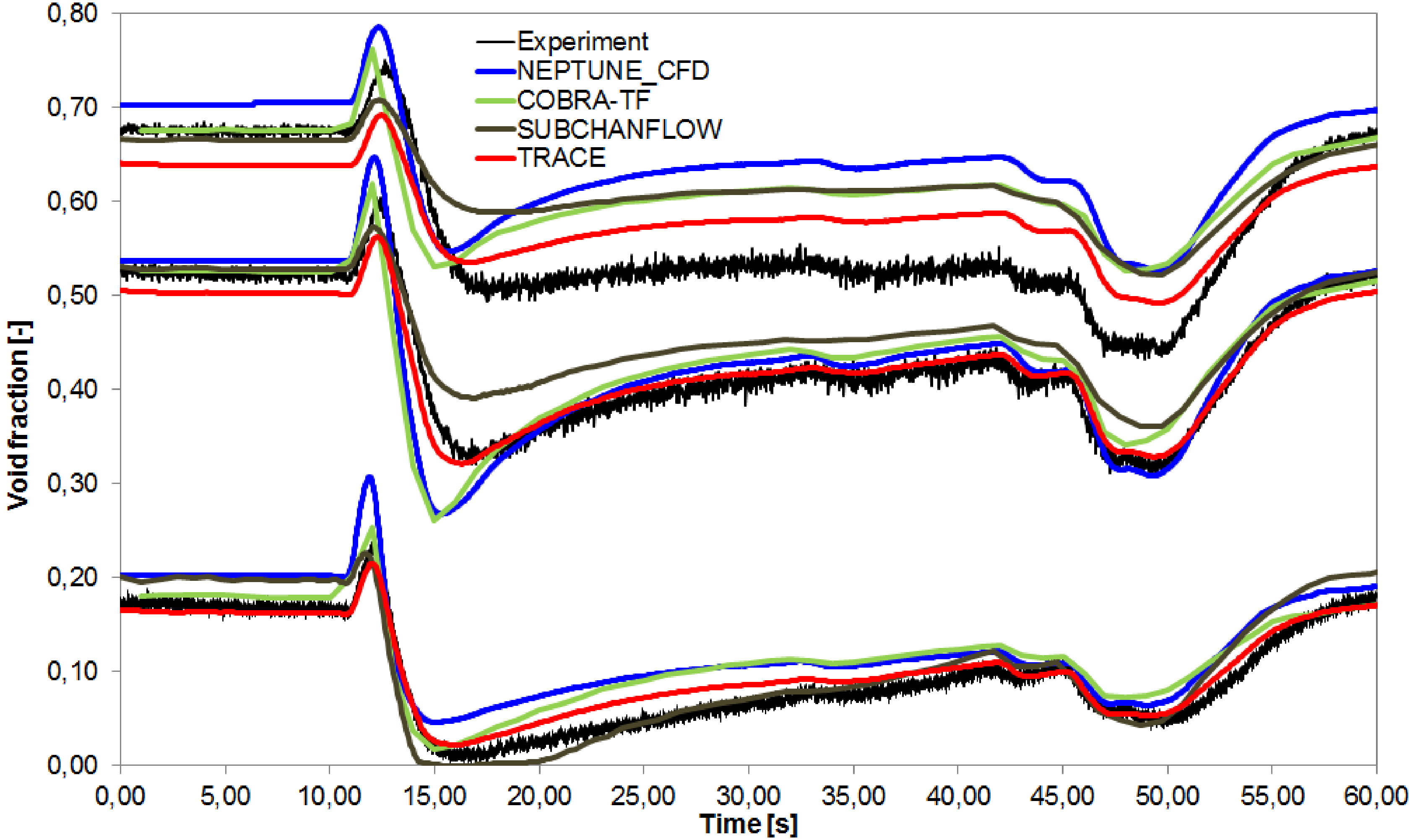
- New scripts for transient analyses and post-processing
- Around 500 lines of URANIE code
- More details can be found in:
  - *D11.22 - Report on FLICA UQ results for BWR ATWS analysis (t0+18)*

DRAFT AVAILABLE

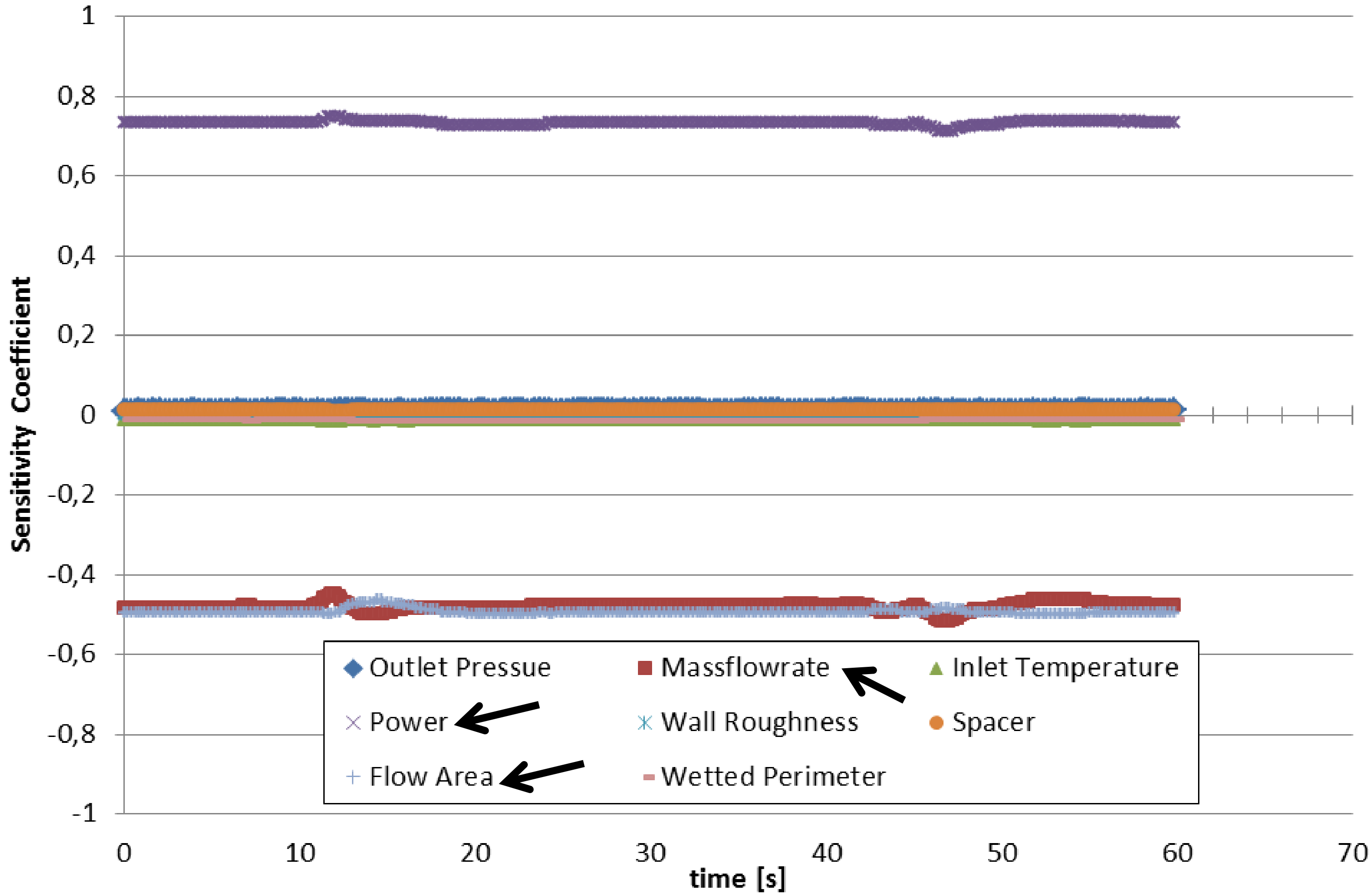
```
for (int i=0; i<timesteps; ++i) {

  std::ostringstream ss;
  ss << i;
  std::string attribname1 = "total_ax_pres_loss_uranie_" + ss.str();
  std::string attribname2 = "exit_void_fraction_uranie_" + ss.str();
  std::string attribname3 = "low_void_fraction_uranie_" + ss.str();
  std::string attribname4 = "mid_void_fraction_uranie_" + ss.str();
  std::string attribname5 = "top_void_fraction_uranie_" + ss.str();

  TAttribute *avgpres = new TAttribute(attribname1);
  avgpres->setDefaultValue(-200.0);
  fout->addAttribute(avgpres);
  TAttribute *avgvoid = new TAttribute(attribname2);
  avgvoid->setDefaultValue(-200.0);
  fout->addAttribute(avgvoid);
  TAttribute *avglow = new TAttribute(attribname3);
  avglow->setDefaultValue(-200.0);
  fout->addAttribute(avglow);
  TAttribute *avgmid = new TAttribute(attribname4);
  avgmid->setDefaultValue(-200.0);
  fout->addAttribute(avgmid);
  TAttribute *avgtop = new TAttribute(attribname5);
  avgtop->setDefaultValue(-200.0);
  fout->addAttribute(avgtop);

}
```

```
void uranie_script(Int_t nS = 100) {

  int timesteps = 300;

  TDataServer *tds = new TDataServer("tdsSCF", "uranie_script");
  TNormalDistribution *tmp = new TNormalDistribution("OutletPressure", 1.0, 0.01);
  tmp->setBounds(0.990000, 1.010000);
  TNormalDistribution *tmf = new TNormalDistribution("MassFlowRate", 1.0, 0.01);
  tmf->setBounds(0.990000, 1.010000);
  TUniformDistribution *tut = new TUniformDistribution("InletTemperature", 0.990000, 1.010000);
  TNormalDistribution *tml = new TNormalDistribution("Power", 1.0, 0.01);
  tml->setBounds(0.990000, 1.010000);
  TNormalDistribution *tmr = new TNormalDistribution("Roughness", 1.0, 0.01);
  tmr->setBounds(0.990000, 1.010000);
  TNormalDistribution *tms = new TNormalDistribution("Spacer", 1.0, 0.01);
  tms->setBounds(0.990000, 1.010000);
  TNormalDistribution *tma = new TNormalDistribution("FlowArea", 1.0, 0.01);
  tma->setBounds(0.990000, 1.010000);
  TNormalDistribution *tmw = new TNormalDistribution("WettedPerimeter", 1.0, 0.01);
  tmw->setBounds(0.990000, 1.010000);

  tds->addAttribute(tmp);
  tds->addAttribute(tmf);
  tds->addAttribute(tut);
  tds->addAttribute(tml);
  tds->addAttribute(tmr);
  tds->addAttribute(tms);
  tds->addAttribute(tma);
  tds->addAttribute(tmw);

  TString sFileName = TString("input.txt");
  tds->getAttribute("OutletPressure")->setFileKey(sFileName, "outlet_pressure");
  tds->getAttribute("MassFlowRate")->setFileKey(sFileName, "mass_flow_rate");
  tds->getAttribute("InletTemperature")->setFileKey(sFileName, "inlet_temperature");
  tds->getAttribute("Power")->setFileKey(sFileName, "power");
  tds->getAttribute("Roughness")->setFileKey(sFileName, "wall_roughness");
  tds->getAttribute("Spacer")->setFileKey(sFileName, "k-spacer");
  tds->getAttribute("FlowArea")->setFileKey(sFileName, "flow_area");
  tds->getAttribute("WettedPerimeter")->setFileKey(sFileName, "wetted_perim");

  TSampling *sampling = new TSampling(tds, "lhs", nS);
  sampling->generateSample();

  TOutputFileKey *fout = new TOutputFileKey("uranie_output.txt");

  for (int i=0; i<timesteps; ++i) {

    std::ostringstream ss;
    ss << i;
    std::string attribname1 = "total_ax_pres_loss_uranie_" + ss.str();
    std::string attribname2 = "exit_void_fraction_uranie_" + ss.str();
    std::string attribname3 = "low_void_fraction_uranie_" + ss.str();
    std::string attribname4 = "mid_void_fraction_uranie_" + ss.str();
    std::string attribname5 = "top_void_fraction_uranie_" + ss.str();

    TAttribute *avgpres = new TAttribute(attribname1);
    avgpres->setDefaultValue(-200.0);
    fout->addAttribute(avgpres);
    TAttribute *avgvoid = new TAttribute(attribname2);
    avgvoid->setDefaultValue(-200.0);
    fout->addAttribute(avgvoid);
    TAttribute *avglow = new TAttribute(attribname3);
    avglow->setDefaultValue(-200.0);
    fout->addAttribute(avglow);
    TAttribute *avgmid = new TAttribute(attribname4);
    avgmid->setDefaultValue(-200.0);
    fout->addAttribute(avgmid);
    TAttribute *avgtop = new TAttribute(attribname5);
    avgtop->setDefaultValue(-200.0);
    fout->addAttribute(avgtop);

  }

  TCode *mycode = new TCode(tds, "/home/trost/SUBCHANFLOW/linux/gfortran-opt-mp/SUBCHANFLOW");
  mycode->addOutputFile(fout);
  mycode->addInputFile("input.txt");

  TLauncher *tlch = new TLauncher(tds, mycode);
  tlch->setSave();
  tlch->setClean();
  tlch->setWorkingDirectory(gSystem->Getenv("PWD") + TString("/tmpUranieSCF"));

  tlch->run();
  tds->exportData("uranie_script_data.dat");

}
```

- **Comparison of results from several codes with the measured data**

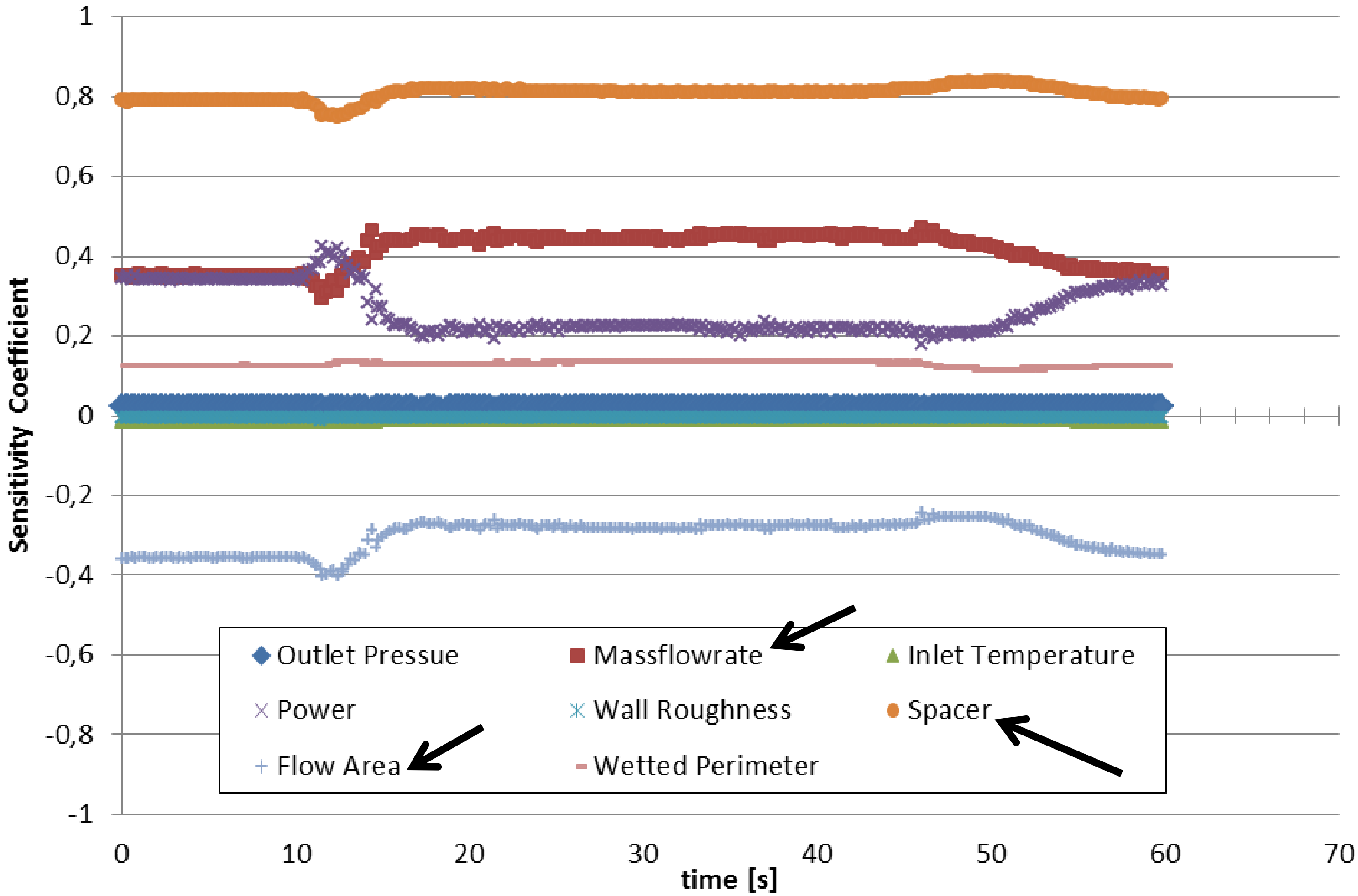


- **Sensitivity coefficient of the void fraction**



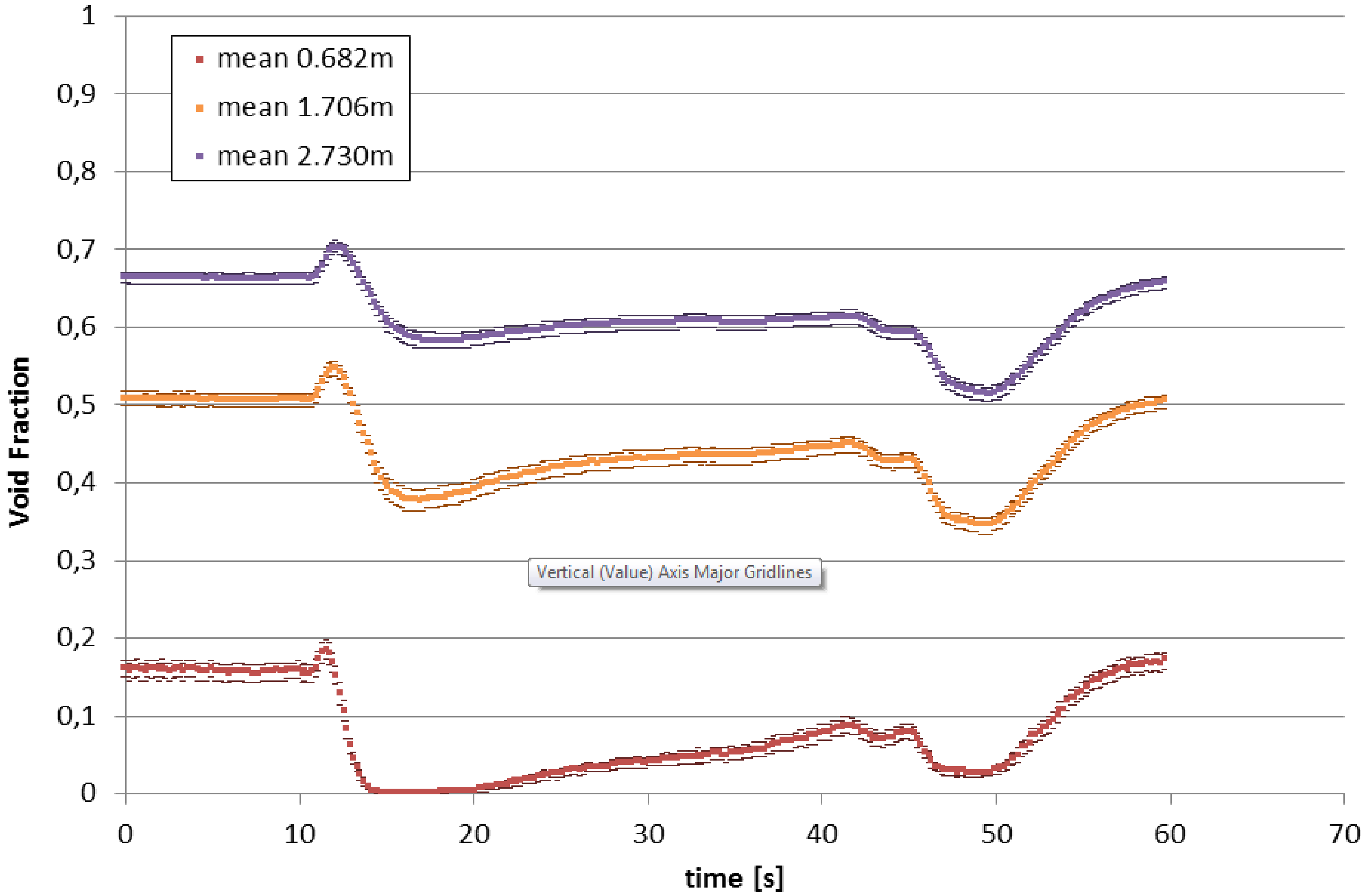
BFBT 4102 - Turbine trip results

- **Sensitivity coefficient of the axial pressure drop**

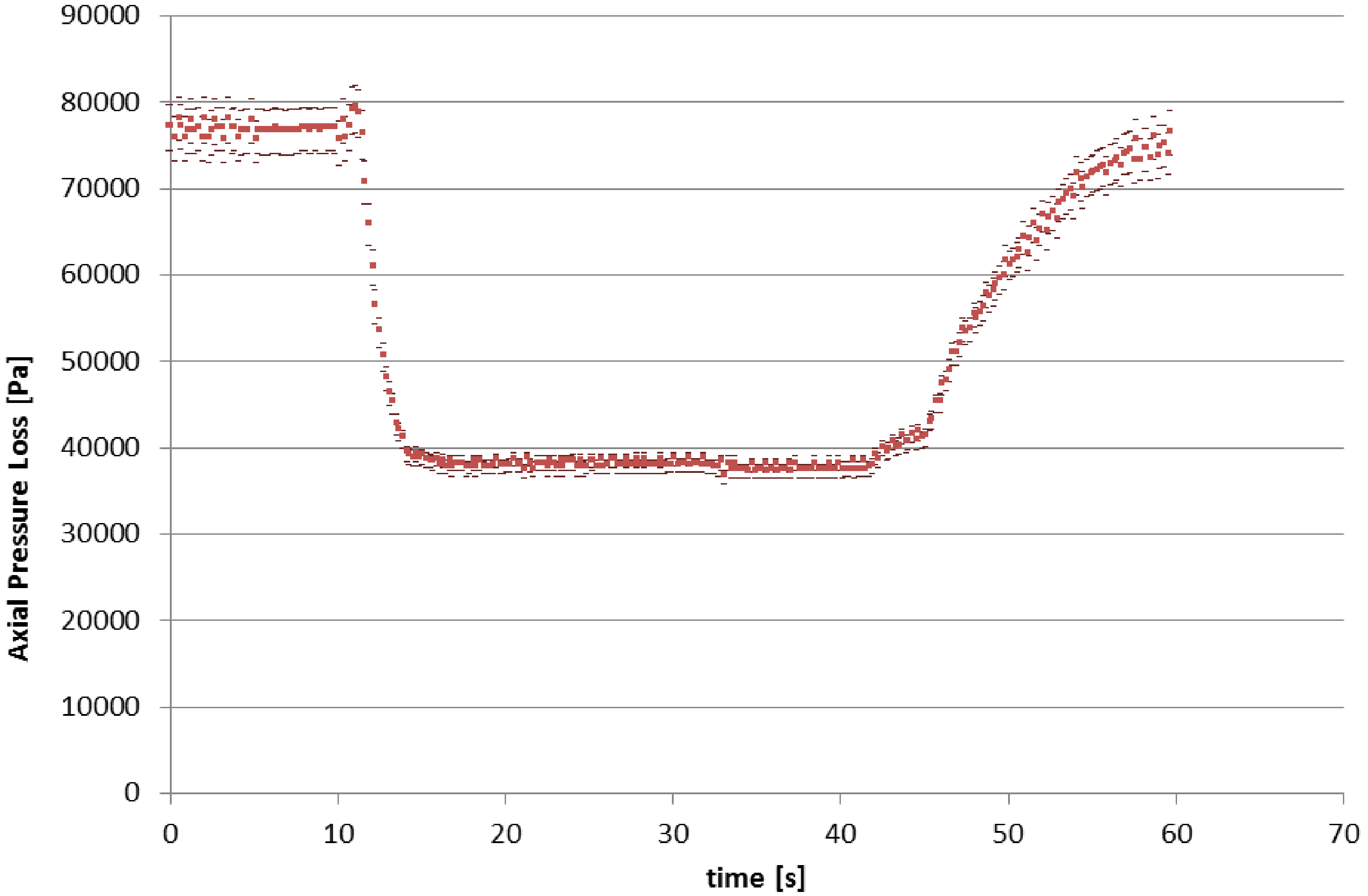


BFBT 4102 - Turbine trip results

- **Void fraction results at the three different elevations.**

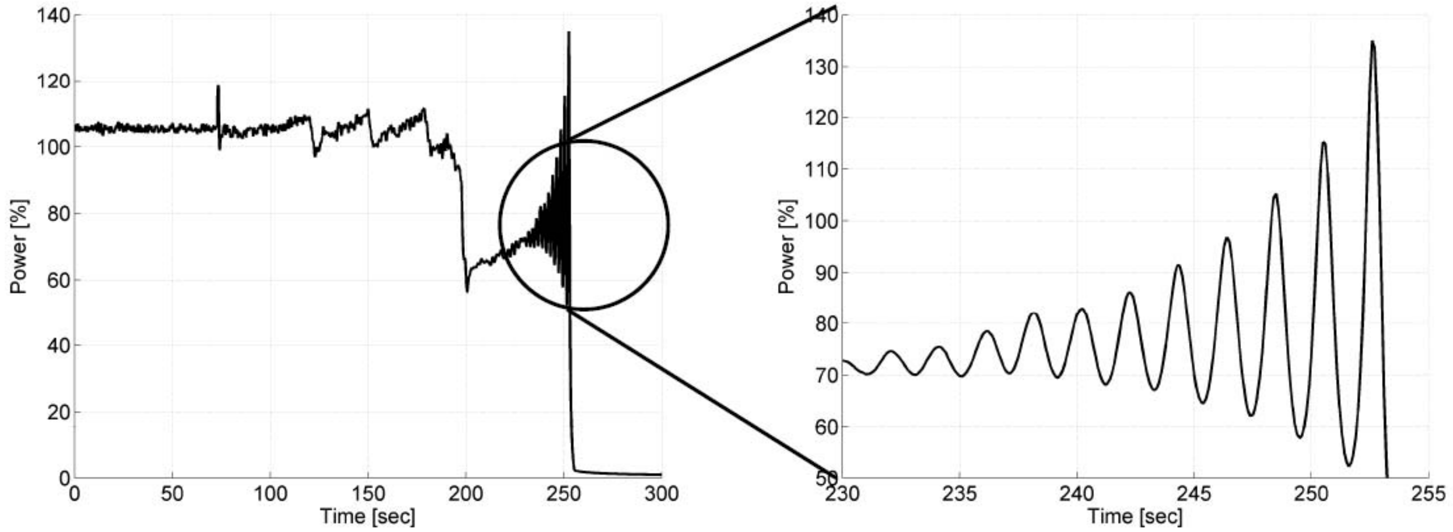


- **Total axial pressure drop**



Application to the O2 -1999 FW transient

- **Power oscillation during the event (feedwater transient)**

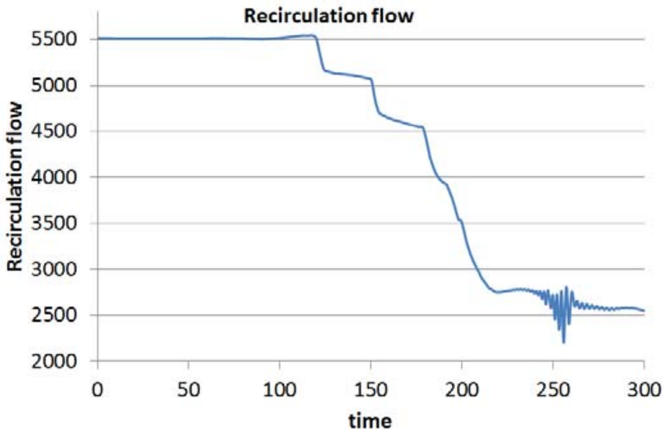


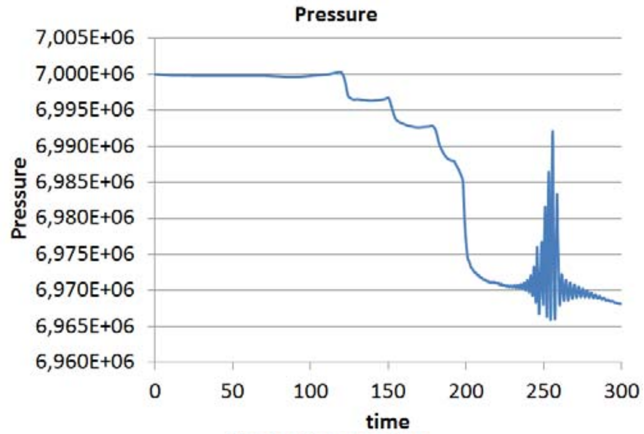
Oskarshamn-2 February 25, 1999 feedwater transient

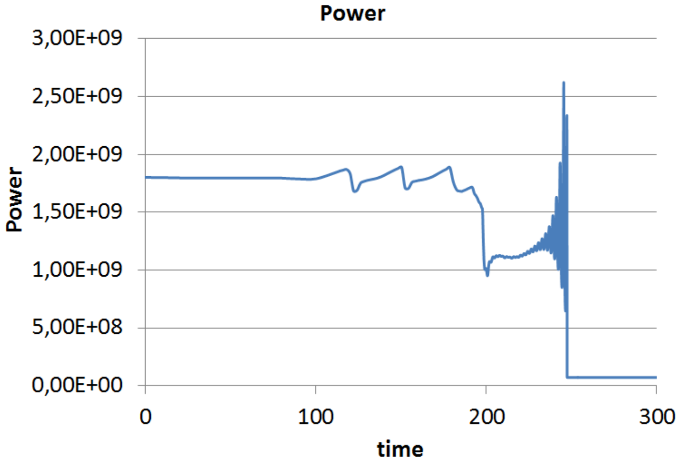
- **Boundary conditions taken from TRACE/PARCS calculation (KIT model with 444 channels)**
- **Modeling the O2 core with SCF and 444 channels**

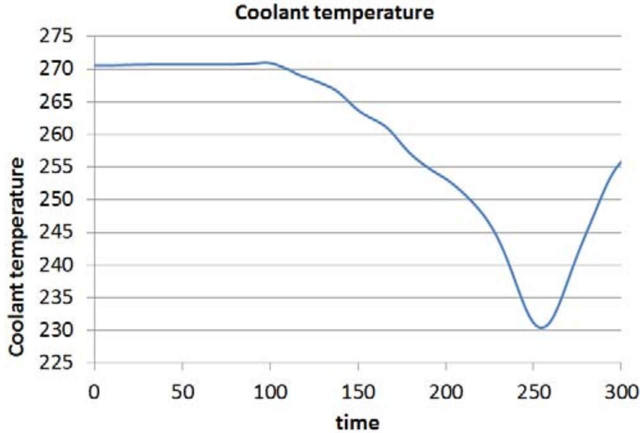
Boundary Conditions applied

- The next boundary conditions were introduced into SCF for the simulation of the oscillations.
- They have been extracted from the TRACE5p3 results
  - Power, inlet temperature, pressure, mass flow rate.



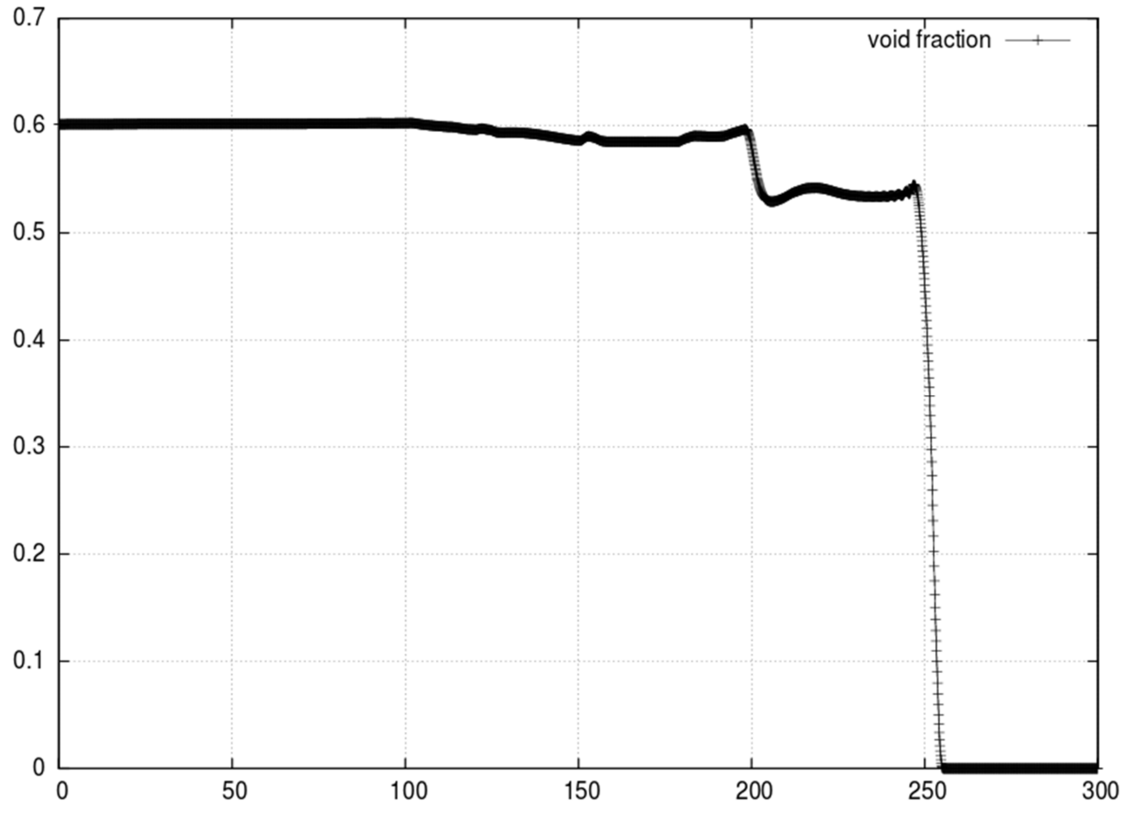






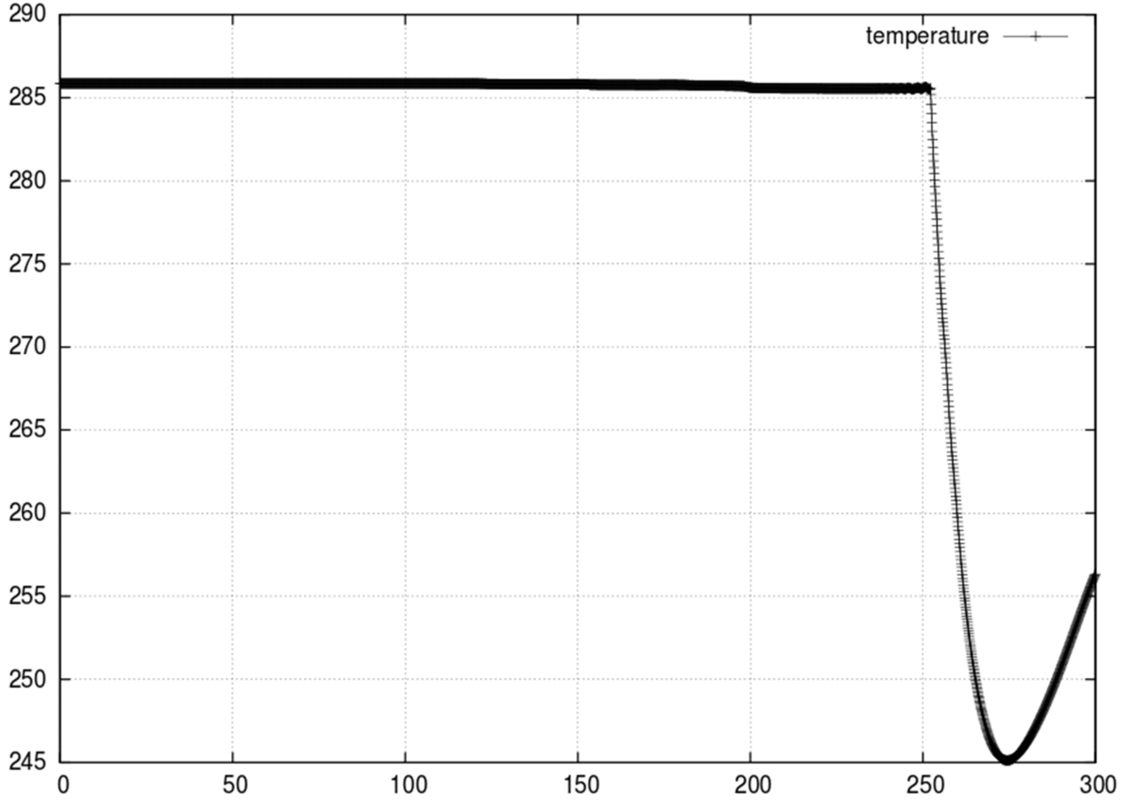
Results from SCF for the full transient

- **Void (bundle average at outlet)**

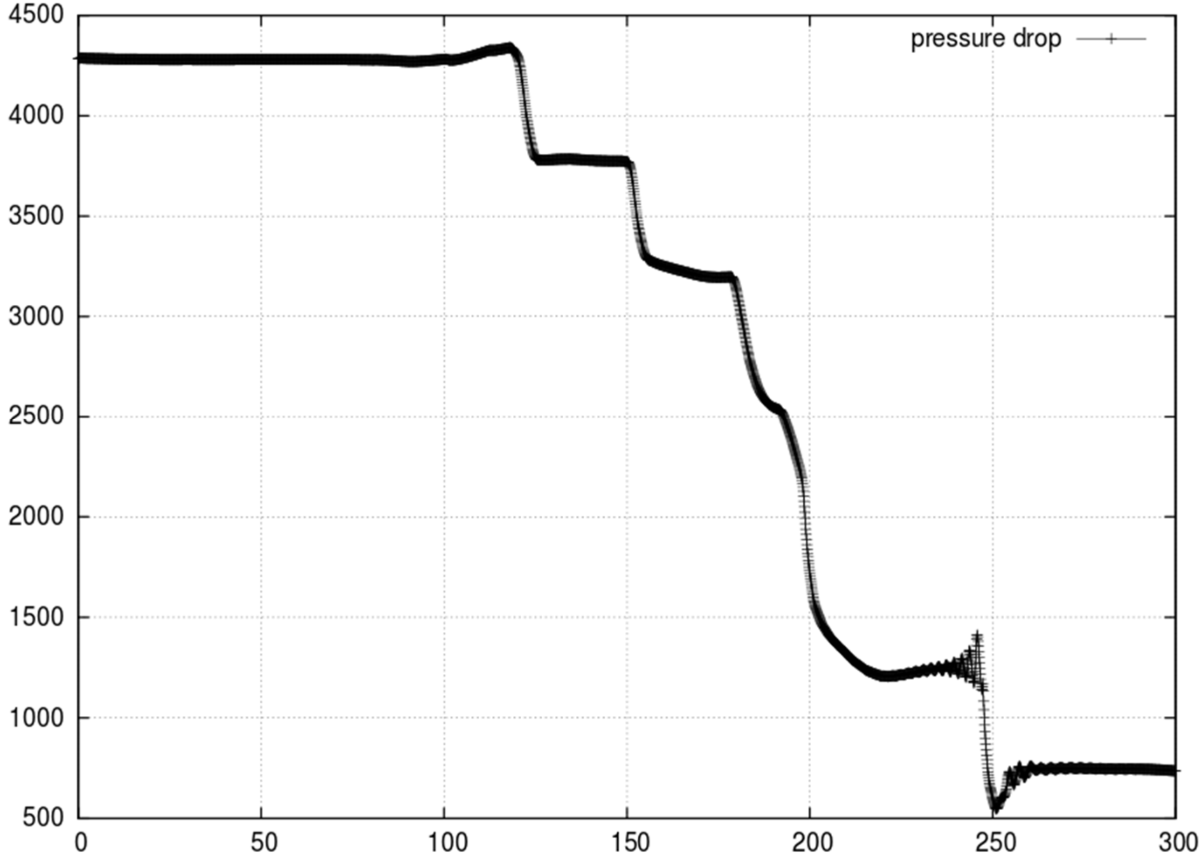


Results from SCF for the full transient

- **Outlet Temperature in Celsius for bundle average**



- **Pressure drop in the core for bundle average**

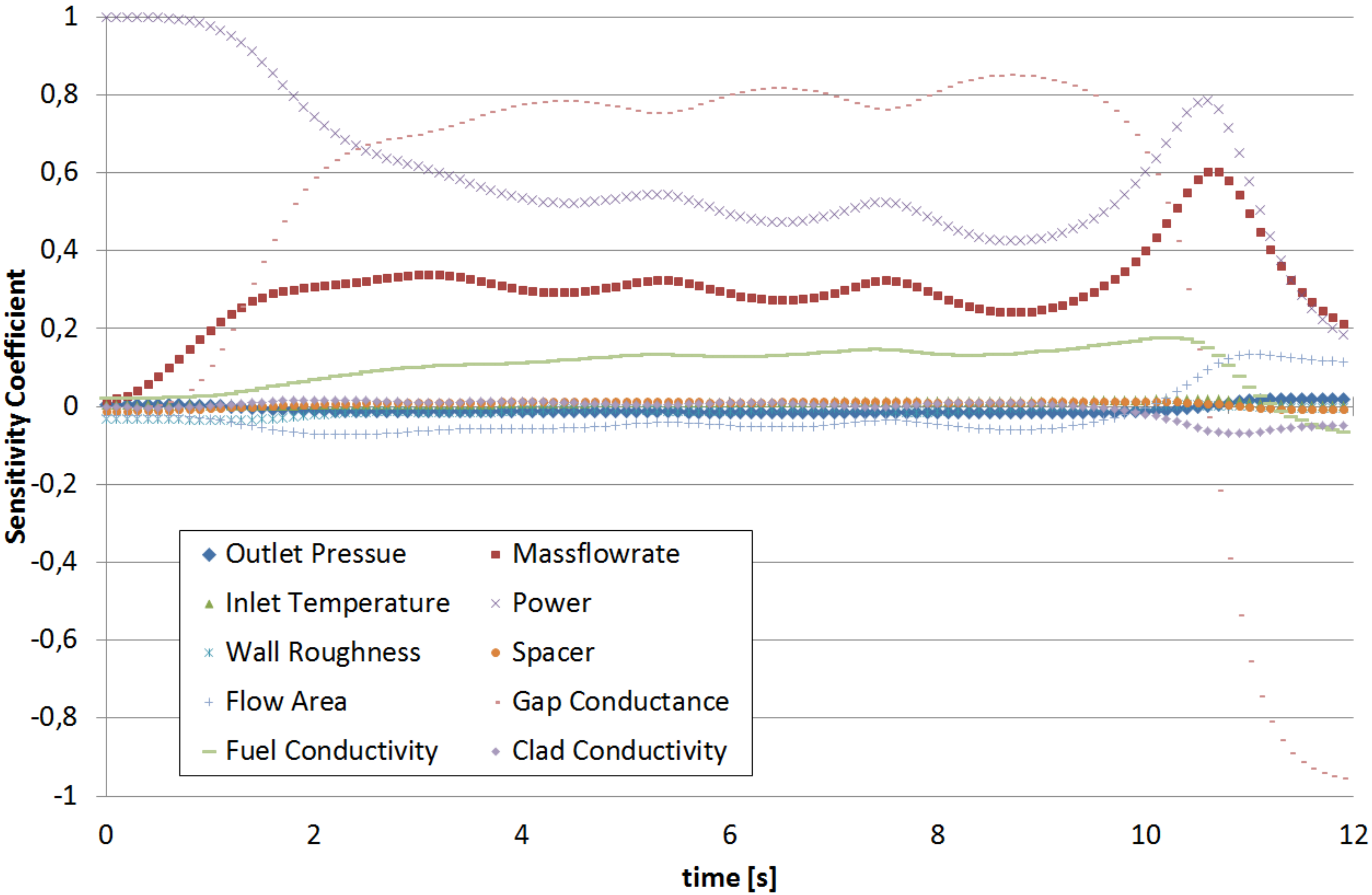


Sensitivity study

- **Zooming and running of a small portion of the transient (12s), when the oscillations start.**
- **Sensitivity analysis with parameters taken from the NURESAFE benchmark specifications (D13.11)**

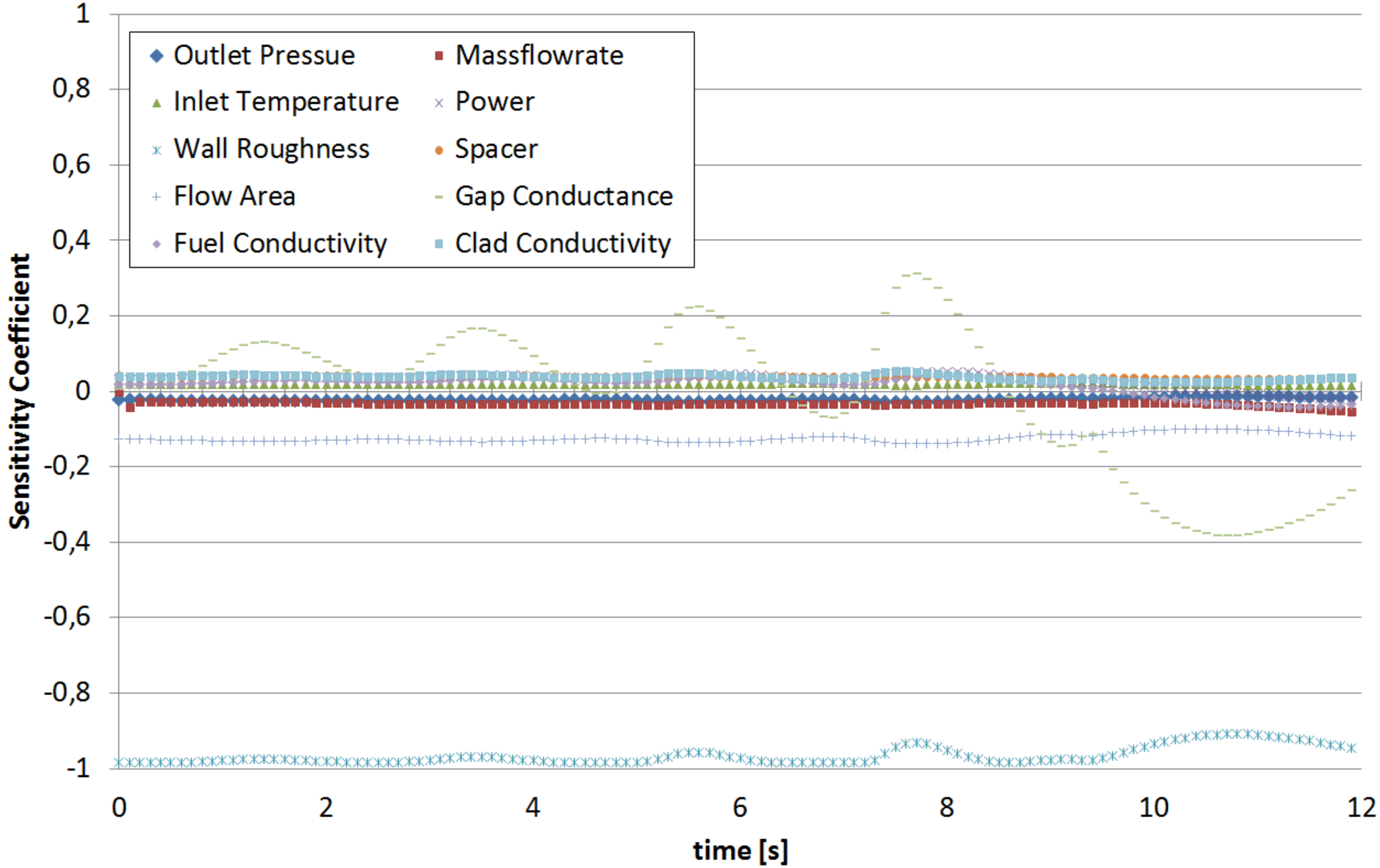
No.	Parameter	Range	Distribution
1	Outlet pressure	± 0.5 %	Uniform
2	Mass flow rate	± 0.5 %	Uniform
3	Inlet temperature	± 2.0 %	Normal
4	Power	± 0.75 %	Normal
5	Cladding Wall Roughness	± 30.0 %	Normal
6	Spacer grid pressure drop coefficient	± 5.0 %	Uniform
7	Gap Conductance	± 35.0 %	Uniform
8	Fuel Conductivity	± 10.0 %	Uniform
9	Cladding Conductivity	± 6.25 %	Uniform

- **Sensitivity of the void fraction**



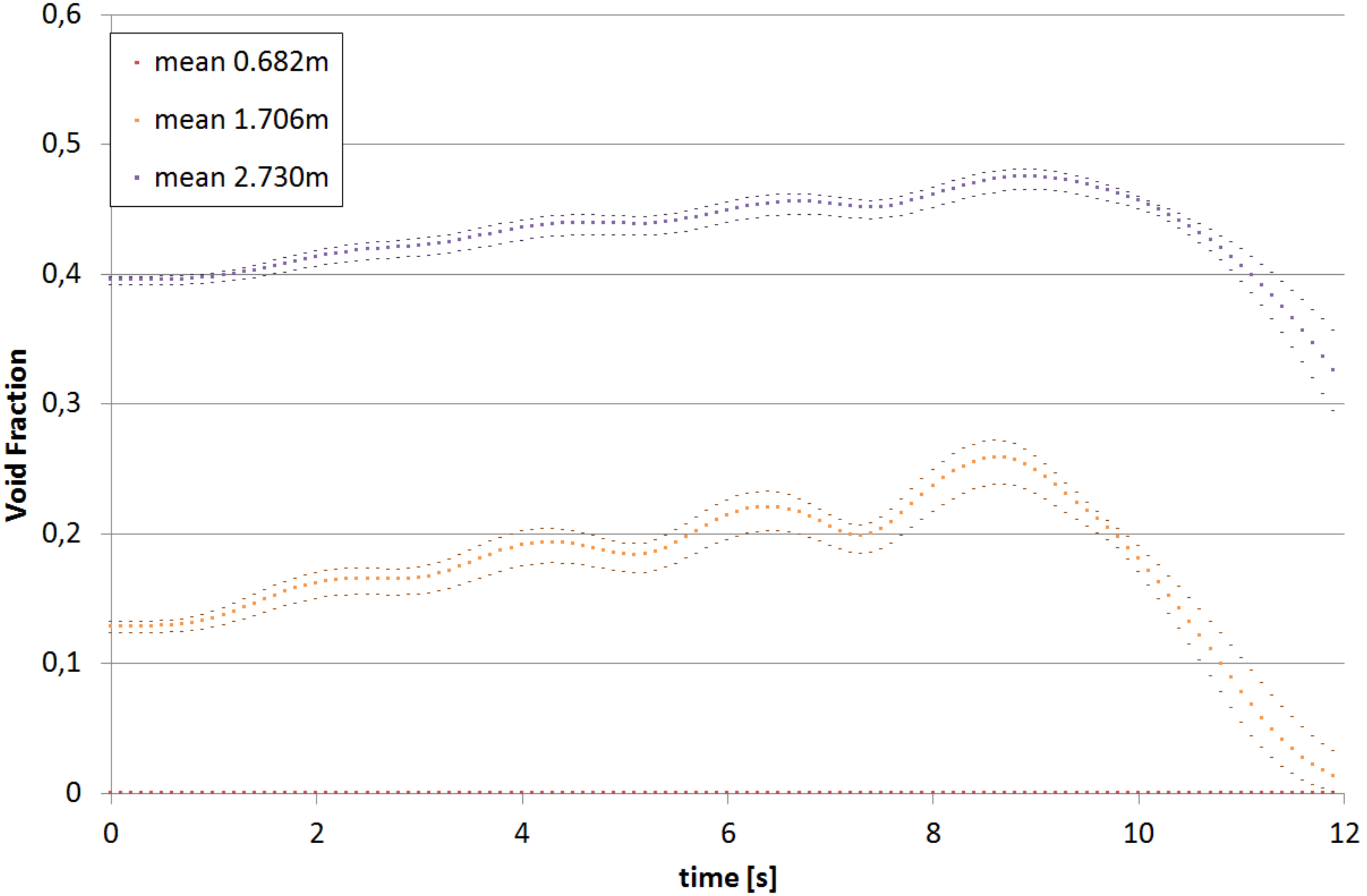
Results in the zooming area

- **Sensitivity of the axial pressure drop**

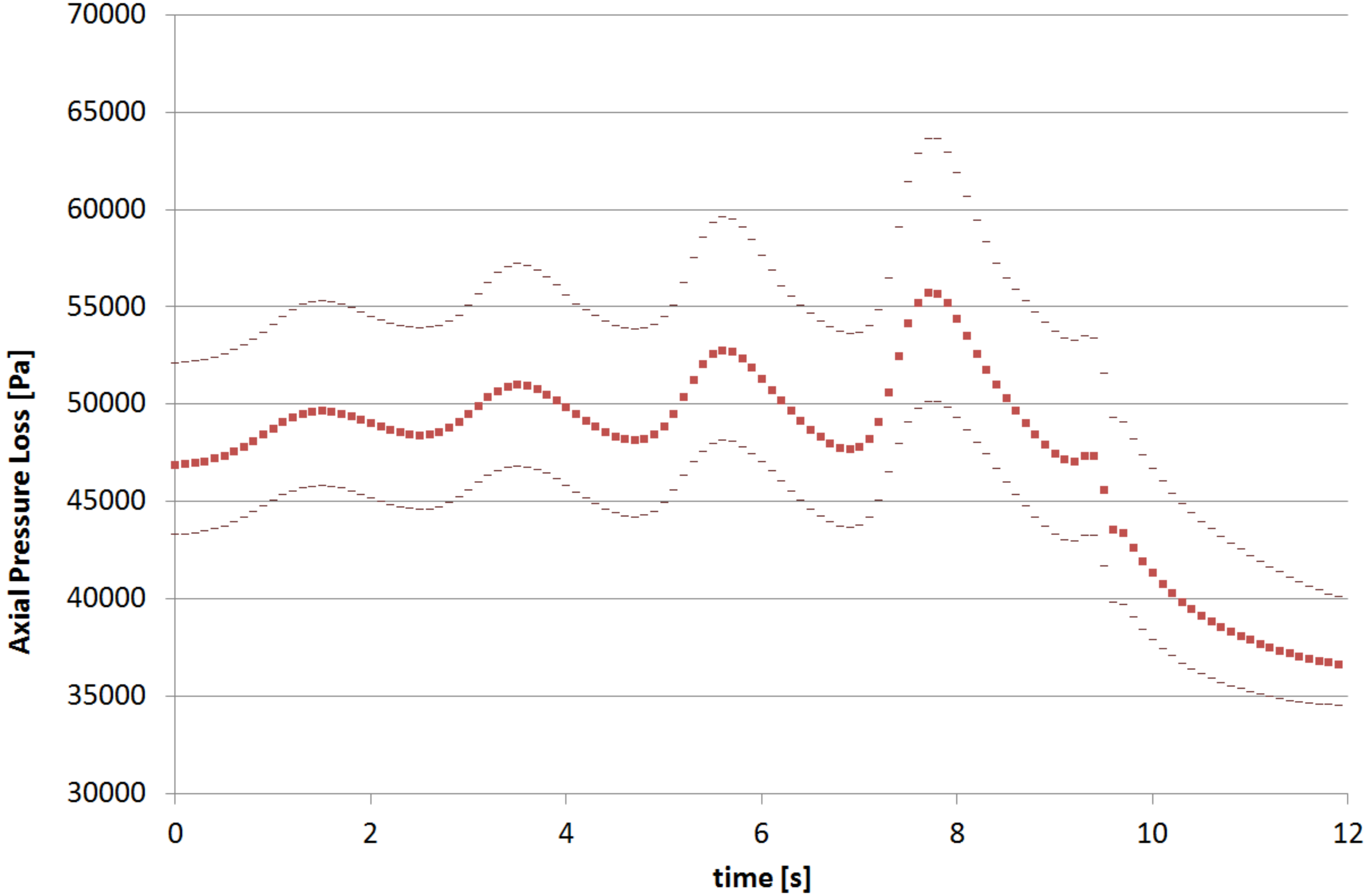


Results in the zooming area

- **Void fraction results**



Axial pressure drop results



Conclusions and Outlook

- **During the first 12 months of the project, investigations on the use of URANIE platform for sensitivity analyses have been conducted.**
- **As a proof of principle, studies have been conducted using the SUBCHANFLOW code (similar to FLICA4).**
- **In June 2013 we reported in Dresden the scripts to analyze steady state scenarios.**
- **Now the scripts have been extended also to transient simulations.**
- **The scripts can be extrapolated to any code with input text file: CTF, FLICA4, DYN3D, COBAYA3, etc, ...**

Conclusions and Outlook

- **Work for the next 6 months: Finalizing D11.22 for (t0+18)**
  - The FLICA4 input deck for O2 has been prepared at KIT for comparison studies
  - FLICA4 will not be used for the coupled computation analyses within WP1.3, only CTF
  - Therefore, to us it is a non-sense to use FLICA4 within WP1.1 if it is not going to be applied within WP1.3
- **Hence we will propose to change the title and content of D11.22 to Report on CTF UQ results for BWR ATWS analysis.**

DISCUSSION

THANKS FOR YOUR ATTENTION