DEPARTMENT OF INFORMATICS

TECHNICAL UNIVERSITY OF MUNICH

Master's Thesis in Informatics

Extending a CFD Lab Course by a preCICE Conjugate Heat Transfer Tutorial

Andreas Reiser

DEPARTMENT OF INFORMATICS

TECHNICAL UNIVERSITY OF MUNICH

Master's Thesis in Informatics

Extending a CFD Lab Course by a preCICE Conjugate Heat Transfer Tutorial

Erweiterung eines CFD Praktikums um eine Lehreinheit zu Wärmeübertrag mittels preCICE

Author:	Andreas Reiser
Supervisor:	Univ.-Prof. Dr. Hans-Joachim Bungartz
Advisor:	Dr. rer. nat. Benjamin Uekermann
Submission Date:	17.09.2018

I confirm that this master's thesis in informatics is my own work and I have documented all sources and material used.

Munich, 17.09.2018 Andreas Reiser

Acknowledgments

First and foremost, I would like to express my deepest gratitude and appreciation to my thesis advisor, Benjamin Uekermann. His continued interest and investment in the topic have made working on this thesis a pleasure. Thank you for always providing much-needed guidance and advice.

Additionally, I would like to thank Gerasimos Chourdakis for always making himself available for questions regarding preCICE and OpenFOAM.

Last but not least, I would like to thank my friends and family for their support.

Abstract

Conjugate heat transfer refers to the coupled analysis of the thermal interactions between fluids and solids. Using preCICE, a free library for black-box, partitioned surface coupling, it is possible to execute such a multi-physics simulation with only single-physics simulation software. For this purpose, each participating solver needs an adapter that connects itself to preCICE, granting the necessary means to exchange required pieces of data and steer the simulation.

This thesis presents the development of an educational concept in form of a lab course which can effectively convey the concept of multi-physics simulations and the usage of preCICE over the course of one semester. An existing CFD lab course is therefore adapted and extended to create a new variant of it that fits the objective. The new version utilizes the Boussinesq approximation as the heat transport model and OpenFOAM as the solid coupling partner. The final implementation is validated with two validation cases where the reference results are obtained using OpenFOAM for the fluid as well as the solid participant.

The new concept is used to teach the CFD lab course in the summer semester of 2018 giving the opportunity to gather and discuss the students' feedback and solutions. Based on this information the educational concept is evaluated and improvements for the future are proposed.

Contents

1 Introduction

Computational Fluid Dynamics and numerical simulation in general has become an important field in research and especially engineering. The applications range from fluid dynamics and structural mechanics to electrodynamics and molecular dynamics. A lot of research has gone into creating a multitude of tools which can simulate individual phenomenon on its own very precisely. Teaching students about these concepts and getting them interested in this field is an integral part of this development.

However, the majority of scenarios are not driven by a single phenomenon alone. Take, for example, the cooling of a CPU. Heat is conducted away from the cores through metal and thermal paste, on to the cooler and spread out with thin metal fins where it is convected away by air flowing through them. Such a scenario refers to conjugate heat transfer (CHT) and requires that we solve the heat-transfer equations for a solid and the mass, momentum and heat-transfer equations for a fluid. Moreover, as the temperature appears in both sets of equations on the interface of the two domains, we have to ensure the continuity of the temperature over it. One approach to solving problems like this is to build and use monolithic tools. An alternative is to couple existing, single physics tools in such a way that they can work together. The coupling library preCICE [Bun+16] provides a "black-box" approach for that exact purpose. Simulating multiple interacting physical effects at the same time produces more accurate simulations and can help us understand their effects more clearly. The concept of coupled simulations is, therefore, a worthwhile and valuable subject to teach students.

The main motivation and ambition of this thesis are to create an educational concept in the form of a lab course which can successfully convey the concept of coupled multi-physics simulations using preCICE. The target audience of the course is second semester Master students studying either Computational Science and Engineering or Computer Science. Some experience in C/C++ programming and a basic understanding of numerical algorithms is required.

There already exists a proven CFD lab course based in parts on the book by [GDN98] that has been taught at the TUM for a number of years. In this course, the students form groups of three and implement both a Navier-Stokes Equations (NSE) solver as well as a solver based on the Lattice-Boltzmann Method (LBM). In later parts, they add functionality to handle arbitrary geometries and parallelization to a solver of their choice.

This goal raises several challenges. For this purpose, we need an appropriate multi-physics coupled phenomenon. We choose CHT our demonstrative example as it has many practical and common applications and is still relatively easy to implement in the already given framework, other than, for example, fluid-structure interaction. First and foremost, we need to select a suitable heat transport model and solid coupling partner. Second, the existing NSE solver has to be adapted and extended in such a way that it can reasonably handle the heat-transfer and coupling requirements needed. The final implementation also has to be validated and compared to reference test cases. Finally, we have to think of new simulation scenarios to be used as exercises and the division of the content to create manageable worksheets.

The solution we propose removes all subject matter related to LBM in order to accommodate the new content in regards to coupling and CHT. The rationale behind this is rather pragmatic: We need space. Furthermore, the educational concept we develop in this thesis is quite modular and meant to be another variant of the already established lab course. We use the Boussinesq Approximation as described in [GDN98] as our heat transport model as well as choosing OpenFOAM to be our solid coupling partner. The lab course is conducted in the summer semester of 2018 at the TUM using the new concept alongside the writing of this thesis. This gives the opportunity to gather the direct anonymous feedback of the students who took the course and their anonymized solutions for the worksheets. We discuss the feedback at the end of the thesis and argue what parts worked well and were well received and what parts did not and need changing. Based on that, we propose potential improvements for the future.

Thesis Structure

The thesis is organized in the following way: Chapter 2 sets the context of this thesis and provides the necessary background information, including an overview of the old lab course, the physics of conjugate heat transfer, and an introduction to preCICE. Chapter 3 deals with making the decision on which heat transport model and coupling partner to use. Chapter 4 contains the description of the implementation and the validation of the final solution. Following that, Chapter 5 concerns itself with the educational considerations. In particular, the structure of the new lab course, new simulations scenarios, and the resources are given to the students. Concluding the thesis, Chapter 6 gathers and discusses the students' feedback and draws conclusions.

2 Background

To put this thesis into context, we lay out the necessary background information in this chapter. First, I explain what was taught in and how the CFD Lab Course was structured thus far. Second, we describe what conjugate heat transfer is by giving a brief introduction on the mechanisms of heat transfer, its governing equations, and associated boundary conditions. Lastly, we outline in more detail the capabilities, usage and the general structure of a preCICE adapter.

2.1 The CFD Lab Course

The CFD Lab Course is a lab course held at the Scientific Computing Chair of the Department of Informatics at TUM. As the name suggests the course deals with Computational Fluid Dynamics and has been repeatedly taught and modified for a number of years. This section describes in more detail how the course was structured so far and how parts of it are implemented.

2.1.1 Previous Structure of the Lab Course

The lab course consists of four worksheets and a project phase afterward. The students work together in teams of three people. Each worksheet has to be handed in after a fixed period of two weeks and is accompanied by one lecture introducing the required theory. Four weeks are planned for the project phase. The worksheets are:

1. **Navier-Stokes Equation Solver:** The students implement a 2D NSE solver using finite differences on a staggered grid.

2. **Lattice Boltzmann Method:** The students implement a 3D LBM solver.

3. **Arbitrary Geometries:** The students choose one of the previously implemented solvers and extend it to handle arbitrary geometries.

4. **Parallelization:** The students again choose one of the solvers from worksheet 1 or 2 and extend it to handle parallel execution via MPI.

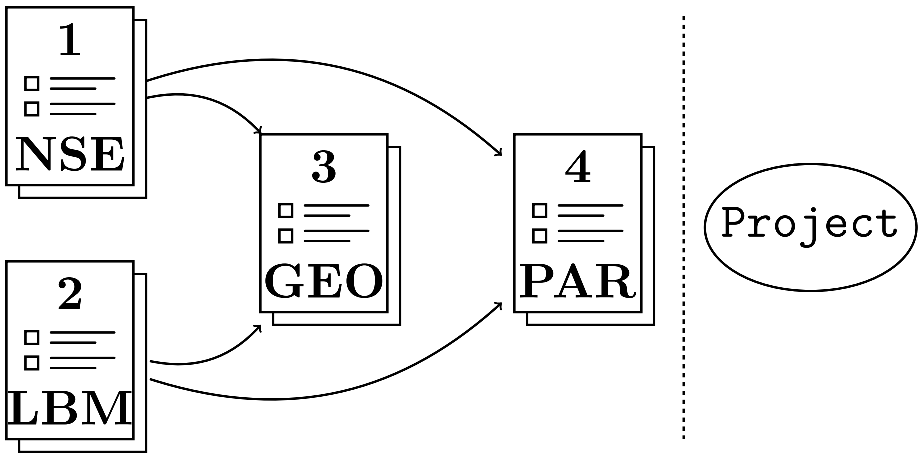


Figure 2.1: The CFD Lab Course Structure. **NSE** - Navier-Stokes solver worksheet, **LBM** - Lattice Boltzmann Method worksheet, **GEO** - Arbitrary Geometries worksheet, **PAR** - Parallelization worksheet.

2.1.2 Implementation of the Navier-Stokes Solver

The implementation of the Navier-Stokes solver is largely based on the instructions in the book [GDN98]. Chapter 3 of the book deals with the basic numerical treatment of the NS equations and the handling of arbitrary geometries.

The domain is discretized using a staggered grid where the different unknown variables are located on different spots on the grid. Pressure values are located at the midpoint of each cell, horizontal velocity values are located on the midpoints of the horizontal cell edges and vertical velocity values on the midpoints of the vertical cell edges.

The *finite difference method* is used for finding the solution. In particular, we use the explict Euler method for the time discretization of the momentum equations. The terms $F^{(n)}$ and $G^{(n)}$ contain the discretized differential expressions of the momentum equations evaluated at time step n. The new velocities in time step $n+1$ are determined in terms of the velocities of the time step n and the pressure of time step $n+1$. To implicitly calculate the pressure, the continuity equation is used to form the pressure-poisson-equation which then has to be solved each time step. An own implementation of the successive over-relaxation (SOR) method is used to iteratively solve the linear system of equations.

All necessary simulation parameters are read at runtime from a text file which enables running different simulations without having to recompile the code. This is not only practical but also necessary later on when two instances of the solver with different configurations have to be run at the same time. Arbitrary geometries are handled via an

`B_O`	`B_W`	`B_S`	`B_N`	`FLUID`

Table 2.1: Original bit field layout.

array of flags which stores additional information about each cell inside the domain. The bitfield indicates if the corresponding cell is either a fluid or obstacle cell and also what kind of cell its four neighbors are. The layout of the bitfield is shown in Table 2.1.

Four new parameters (`wr`, `wl`, `wt`, `wb`) are read, one for each domain boundary, specifying the type of boundary condition along its edge. The configuration of the domain is done via a pgm file which makes for an easy setup. Grayscale pgm stores one integer value per pixel. Each value corresponds to either a fluid- or obstacle-cell. Obstacles inside the domain can only have a no-slip boundary condition.

Algorithm 1: Previous Algorithm

Read parameters from file
Assign initial values to u, v and p
Set $t := 0$, $n := 0$
while $t < t\_end$ **do**
  Select δt
  Set domain boundary values for u and v
  Set obstacle boundary values for u and v
  Set problem specific boundary values (inflow, etc.)
  Compute $F^{(n)}$ and $G^{(n)}$
  Compute right-hand side of the pressure poisson equation
  Set $it := 0$
  while $it < it_{max}$ *and* $r^{it} > eps$ **do**
    Perform an SOR cycle
    Compute residual r^{it}
    $it := it + 1$
  Compute $u^{(n+1)}$ and $v^{(n+1)}$
  Output u, v and p to VTK files if necessary
  $t := t + \delta t$
  $n := n + 1$

2.2 Conjugate Heat Transfer

Conjugate heat transfer refers to the coupled analysis of the thermal interactions between fluids and solids. Conjugate heat transfer analysis matches the temperature and heat flux at the fluid-solid interface and thus eliminates the need for the heat transfer coefficient h, an empirically found constant relating the heat flux to the temperature difference of the fluid and solid. It is not a material property and it depends, among other things, on the geometry. Based on the related matter discussed in the thesis by [Che16], we give a brief summary of the relevant physical models and terminology.

2.2.1 Mechanisms of Heat Transfer

Heat transfer mechanisms are the ways by which energy can be transferred between objects. The three main mechanisms are thermal conduction, thermal convection and radiation. Radiation is not relevant for this topic, hence we only explain the first two in more detail.

Heat Conduction

Heat conduction occurs at a molecular scale as heat is transferred by microscopic collisions of particles. Energy is transferred from molecules with higher internal energy to molecules with lower energy. The rate of the heat transfer between two bodies in contact is proportional to their temperature difference and is given by Fourier's law of heat conduction:

$$\mathbf{q} = -k\nabla T \tag{2.1}$$

where

- $\mathbf{q}$ $[W/m^2]$ is the heat flux density.
- k $[W/(m \cdot K)]$ is thermal conductivity of the material and may be temperature dependent.
- ∇T $[K/m]$ is the temperature gradient.

Heat Convection

Heat convection happens in fluids and occurs both because of the microscopic collisions of particles (diffusion) as well as the bulk motion of the fluid (advection). The effect can be described by Newton's law of cooling:

$$\frac{dQ}{dt} = hA(T_s(t) - T_\infty) \tag{2.2}$$

where

- Q $[J]$ is the thermal energy.
- T_s is the temperature of the solid body.
- T_∞ is the environment temperature.
- A $[m^2]$ is contact surface area.
- h $[W/(m^2 \cdot K]$ is the heat transfer coefficient.

More parameters of heat transfer

In addition to the already covered thermal conductivity k and heat transfer coefficient h, we need a few more:

- c_p $[J/kg \cdot K]$ is the specific heat capacity and describes the amount of energy needed to raise the body's temperature by 1 K.
- α $[m^2/s]$ is the thermal diffusivity and describes the thermal inertia of the material.

Another important dimensionless quantity is the Prandtl number. The Prandtl number describes the relative strength of the diffusion of momentum to that of heat. Other than the Reynolds number, the Prandtl number is solely a property of the fluid and not the flow [GDN98]. The Prandtl number is the ratio of the kinematic viscosity ν of the fluid and the thermal diffusivity α.

$$Pr = \frac{\nu}{\alpha} \tag{2.3}$$

Additionally, the conductivity and diffusivity are connected via the density ρ and specific heat capacity c_p:

$$k = \alpha \rho c_p \tag{2.4}$$

2.2.2 Coupling Approach

There exist two approaches to couple heat transfer simulations on their interface: Dirichlet - Neumann coupling and Robin - Robin coupling. The latter is not directly relevant to this thesis and hence we do not discuss it any further.

When using Dirichlet - Neumann coupling, we assume the continuity of the temperature and the heat flux on the fluid-solid interface Γ_{FS}:

$$T_s = T_f \text{ on } \Gamma_{FS} \tag{2.5}$$

$$k_s \frac{\delta T_s}{\delta n} = -k_f \frac{\delta T_f}{\delta n} \text{ on } \Gamma_{FS} \tag{2.6}$$

With this setup, no heat transfer coefficient is needed to calculate the temperature and heat flux distributions at the interface. They are part of the solution to the conjugate problem.

2.3 The Coupling Library preCICE

preCICE (*Precise Code Interaction Coupling Environment*) is a coupling library for partitioned multi-physics simulations [Bun+16]. The philosophy behind preCICE is to reuse existing single-physics simulation software and make them work together to simulate multi-physics phenomena. In such a setup the individual participating solvers are “black-boxes” and preCICE provides different coupling schemes, communication, data-mapping and time interpolation. All the aforementioned components of preCICE are fully configured at run-time via an XML file. preCICE provides TCP/IP socket-based and MPI-based communication methods. Regarding data mapping, a multitude of mapping schemes are available e. g. nearest-neighbor mapping and radial-basis function interpolation. In other words, the solvers connect to preCICE and preCICE provides the necessary means for them to interact with each other. For this purpose, some modifications to the solvers have to be made. The code integrated thereby is generally referred to as the coupling adapter.

Figure 2.2 shows an overview.

Explicit Coupling

For the next examples, we assume that the numerical solver consists of a time-stepping loop which solves a system of partial differential equations. The first step is to extend it with the steering methods preCICE provides:

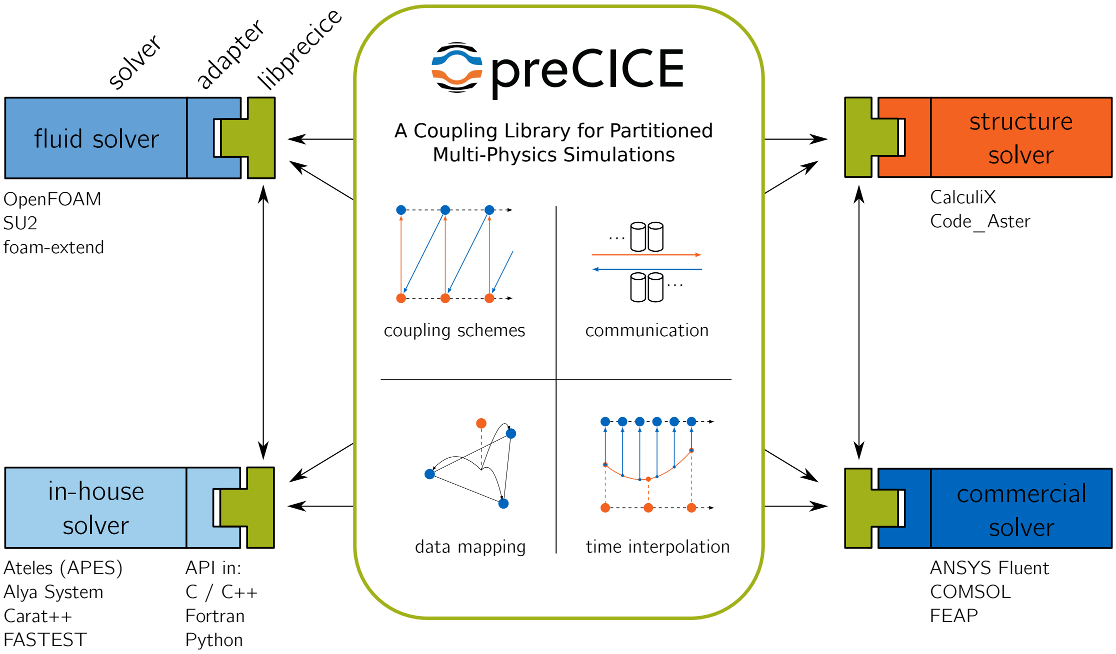


Figure 2.2: preCICE Overview [Bun+16]

`createSolverInterface(config)` creates the solver interface and configures it at run-time using the configuration file.

`initialize()` initializes preCICE and establishes communication channels. It also returns the first maximum time step length.

`advance(computedTimeStepLength)` needs to be called each iteration after the computation with computed time step length. It returns the next maximum time step and applies mapping schemes and facilitates the communication of the coupling data.

`finalize()` frees up resources related to preCICE and closes communication channels.

`couplingOngoing()` return true as long as the maximum simulation time has not been reached yet.

The second step is to actually exchange mesh locations and boundary values. The relevant preCICE methods are:

`setMeshVertices(vertexLocations)` is used to tell preCICE about the location of the coupling mesh.

`writeBlockScalarData()` is used to write coupling data to preCICE.

`readBlockScalarData()` is used to read coupling data from preCICE.

Putting it all together results in the basic structure of an adapted solver, shown in Listing 2.1. In each time step, we first read coupling data and update the boundary values. Second, we solve the time step, extract new boundary values and send them to preCICE. With this kind of setup, explicit coupling schemes can be used.

Listing 2.1: Structure of an adapted solver: Ready for explicit coupling.

```
precice.createSolverInterface("precice_config.xml");
precice.setMeshVertices(...);
precice_dt = precice.initialize();

while precice.couplingOngoing() {
	if precice.isReadDataAvailable() {
		precice.readBlockScalarData(...);
		solver.setCouplingBoundaryValues();
	}

	solver_dt = min(precice_dt, solver.calculateDt());
	solver.solve(solver_dt);

	if precice.isWriteDataRequired(solver_dt) {
		solver.gatherCouplingBoundaryValues();
		precice.writeBlockScalarData(...);
	}

	precice_dt = precice.advance(dt);
}

precice.finalize();
```

Implicit Coupling

When using implicit coupling schemes, preCICE preforms sub-iterations for each time step. To accomplish this the solver has to be able to save the current state of the simulation and restore it, i. e. saving and restoring checkpoints. Depending on the solver, it might not be necessary to store the complete solution fields, sometimes storing only boundary value data can be sufficient. preCICE needs to tell the solver when to save and restore such a checkpoint. For this purpose, preCICE uses actions to signal that a specific operation has to be carried out by the solver. The same actions are used by the solver to confirm that the operation has been successful. Listing 2.2 shows the adapted solver from Listing 2.1 extended with saving and reloading checkpoints in order to handle implicit coupling schemes.

Listing 2.2: Structure of an adapted solver: Ready for implicit coupling.

```
precice.createSolverInterface("precice_config.xml");
precice.setMeshVertices(...);
precice_dt = precice.initialize();

if precice.isActionRequired("write-initial-data") {
        solver.gatherCouplingBoundaryValues();
        precice.writeBlockScalarData(...);
        precice.fulfilledAction("write-initial-data");
}
precice.initializeData();
while precice.couplingOngoing() {
        if precice.isActionRequired("write-iteration-checkpoint") {
                solver.saveCheckpoint();
                precice.fulfilledAction("write-iteration-checkpoint");
        }
        if precice.isReadDataAvailable() {
                precice.readBlockScalarData(...);
                solver.setCouplingBoundaryValues();
        }

        solver_dt = min(precice_dt, solver.calculateDt());
        solver.solve(solver_dt);

        if precice.isWriteDataRequired(solver_dt) {
                solver.gatherCouplingBoundaryValues();
                precice.writeBlockScalarData(...);
        }

        precice_dt = precice.advance(dt);

        if precice.isActionRequired("read-iteration-checkpoint") {
                solver.restoreCheckpoint();
                precice.fulfilledAction("read-iteration-checkpoint");
        }
}
precice.finalize();
```

3 Initial Considerations

As we want to use CHT as our demonstrative example of a coupled simulation setup, we first need to choose an appropriate heat transport model which we can extend the NSE solver by. Afterward, we can think about selecting a suitable solid coupling partner. In this chapter, we discuss the aforementioned points and make a decision regarding both subjects.

3.1 Energy Transport: The Boussinesq Approximation

In this chapter, we focus on the energy transport model we use for our solver: The Boussinesq approximation. We choose this model because its both easy to implement and it is described in the book by [GDN98] on which the NSE solver is already based upon. Hence, the dimensionless formulation and proposed discretization fit nicely into the existing framework. We quickly recapitulate the relevant chapter 9 from the book by [GDN98] and explain what simplifications the Boussinesq approximation makes.

The principle of conservation of energy yields the energy equation.

$$\frac{\delta T}{\delta t} + \vec{u} \cdot \nabla T = \alpha \Delta T + q^{'''} \tag{3.1}$$

with

- α: constant thermal diffusivity.
- $q^{'''}$: a heat source.
- negligible viscous dissipation.

It states that the temperature is not only convected with the flow but also diffuses uniformly in all directions. Changes of the temperature inside a fluid lead to variations in the fluid's density. Heating a fluid causes an increase in its volume and thus lowers the density which in turn causes the fluid to rise. The end effect is temperature dependent buoyancy forces. As further effects on the fluid and flow are difficult to treat, some simplifications have to be made. The Boussinesq approximation states that:

- the density is constant everywhere except in the buoyancy terms,

- all other fluid properties are assumed constant,
- viscous dissipation is negligibly small,
- and that the relation between the density ρ and temperature T is linear.

The first assumption ensures that the continuity equation retains its incompressibility and that density variations only occur in the external force term in the momentum equation. The last assumption implies the coefficient of thermal expansion $\beta = \rho^{-1}\frac{\delta\rho}{\delta T}$ which linearly relates the density and temperature. Of course, these simplifications only make it applicable for small temperature variations.

The dimensionless energy and momentum equation then read:

$$\frac{\delta T}{\delta t} + \vec{u} \cdot \nabla T = \frac{1}{Re \cdot Pr}\Delta T + q^{'''} \tag{3.2}$$

$$\frac{\delta \vec{u}}{\delta t} + (\vec{u} \cdot \nabla)\vec{u} = -\nabla p + \frac{1}{Re}\Delta\vec{u} + (1 - \beta T)\vec{g} \tag{3.3}$$

Pr is the Prandtl number as described in Chapter 2.2

3.2 Choice of Coupling Partner

Having made the decision on the heat transport model, we now need a suitable solid coupling partner. In theory, any solver which is capable of solving the heat transport equation in a solid is a possible candidate. In practice, we need to consider more aspects. In particular, ease of installation and setup, educational viability and overall feasibility in the given time frame. In this chapter, we first present our three top candidates, CalculiX, OpenFOAM and a proprietary python solver. Finally, we consider each one's pros and cons and make the final selection.

3.2.1 CalculiX

CalculiX is a free, open-source finite-element analysis application for three-dimensional structural mechanic's problems and consists of two parts, the solver (CCX) and the pre- and post-processor (CGX). Both are developed by employees of MTU Aero Engines. [DW18] There is a preCICE adapter available. There are ready to install packages of CalculiX for both Linux and Windows but installing the adapter requires building CCX from source which proved to be a considerable challenge.

3.2.2 OpenFOAM

OpenFOAM is a free, open-source software for Computational Fluid Dynamics released under the GNU General Public License. Since 2004 it is published by OpenCFD Ltd., a subsidiary of ESI Group. It is a pure three-dimensional solver and is based on the finite volume method. It comes with a range of solvers and utilities for pre- and post-processing and is readily available on all three major operating systems as well as an Ubuntu package. [Fou]

As OpenFOAM allows its individual solvers to be changed and also has the capability to load libraries at runtime, the preCICE adapter can be built independently from OpenFOAM and does not require the building of OpenFOAM from source. The adapter was first developed by [Che16]. In his master's thesis, [Cho17] then used the previous work and built a general OpenFoam adapter on top of it. The fact that OpenFOAM is a fluid mechanic's solver makes it a non-optimal choice from an educational standpoint for the problem at hand.

3.2.3 Proprietary Python Solver

Besides free open-source solver with available preCICE adapters, there is also the possibility of a proprietary python solver exactly tuned to the problems the students run in the worksheet exercises. It was thought about writing a simple solver solving the Laplace equation on a two-dimensional grid using finite differences. Going with this option means also having to not only validate the fluid solver but also validate the solid solver. The students would not be shown that they can couple their code to a state of the art solver without any problems. This is relevant for the project phase at the end of the course if some students decided that they want to work on a project with preCICE they would have no prior example with an established solver.

3.2.4 Conclusion and Final Selection

None of the three options above are optimal, each one has its own merits and drawbacks. From an educational standpoint, CalculiX is definitely the best one as it is widely used and a true structural mechanic's solver, even though it is a 3D solver only. Its biggest drawbacks are the building procedure and its own system for visualizing the results. Time constraints of the worksheets are a big factor and it is not in our interest to occupy a large portion of the available time to teach building scientific software on Linux.

The biggest advantage of a proprietary python solver is its simplicity to build and run. In addition, it can, by design, be developed as a true 2D solver. Because of time constraints, we opted to not follow this route. The python solver needs to be validated as well and possibly introduces a new source of errors.

Ultimately we choose OpenFOAM to be the solid coupling partner. The biggest drawback is that it is a fluid mechanic's solver and its inherently 3D nature which turns out does not noticeably affect the simulations. Installing OpenFOAM und building its preCICE adapter is also easier in comparison to CalculiX.

4 Implementation of the Reference Solution

At this point in time, we have made the important decisions on what heat transport model and what solid solver to use as coupling partner. This chapter deals with the implementation of the final solution. At first, we look at how the old solver has to be adapted in order to accommodate the heat transport model. Second, we develop and add the preCICE adapter and take a close look at the treatment of the coupling boundaries. Lastly, we validate the finished implementation using two validation cases.

4.1 Adapted Handling of Arbitrary Geometries

Instead of reading the four parameters specifying the boundary condition on the domain boundaries, the adapted solver now uses the pgm file to configure the whole domain, including the domain boundary conditions. Consequently, the bitfield entries need to be extended with additional flags. The updated layout is shown in 4.1.

Not only does this make configuration of the solver more consistent it also adds the capability to have different kinds of boundary values on the same domain boundary. This is especially relevant for coupled simulations where sometimes only a specific section of the boundary is part of the coupling interface.2 In coupled simulations its important that the absolute coordinates of the coupling interfaces of the participants coincide with each other. For this purpose, two new parameters for the x- and y-origin of the domain are added. They are used to adjust the VTK output accordingly and to calculate the positions for the mesh vertices of the coupling interface.

`B_O`	`B_W`	`B_S`	`B_N`	`INFLOW`	`OUTFLOW`	`FREE-SLIP`	`NO-SLIP`	`FLUID`

Table 4.1: Modified flag field layout

4.2 Additions for Heat Transport

Three new parameters are needed for the heat transport, the initial temperature T_I, the coefficient of thermal expansion β and the Prandtl number Pr. An additional data array for the temperature is needed. The updated temperature $T^{(n+1)}$ is computed after setting the boundary values. Temperature boundary conditions are by default adiabatic everywhere. The temperature values also have to be added to the VTK output. As the Equations 3.2 and 3.3 in Chapter 3.1 state, the temperature for the current time step has to be computed before the new `F` and `G` terms can calculated.

Algorithm 2: New Algorithm with Heat Transport

```
Read parameters from file
Assign initial values to u, v, p and T
Set t := 0, n := 0
while t < t_end do
    Select δt
    Set all boundary values for u, v and T
    Set problem specific boundary values (inflow, heated walls, etc.)
    Compute T^(n+1)
    Compute F^(n) and G^(n) adjusted with T^(n+1)
    Compute right-hand side of the pressure poisson equation
    Set it := 0
    while it < it_max and r^it > eps do
        Perform an SOR cycle
        Compute residual r^it
        it := it + 1
    Compute u^(n+1) and v^(n+1)
    Output u, v, p and T to VTK files if necessary
    t := t + δt
    n := n + 1
```

4.3 The preCICE Adapater

At this point, the solver can handle arbitrary geometries as well as heat transport - the necessary prerequisites for finally adding a preCICE adapter in order to simulate conjugate heat transfer phenomena. In this section, we specify the requirements for this new feature along with the necessary additions and changes that have to be made. In the second part, we describe in detail how and why the coupling boundaries are treated in a certain way.

4.3.1 Requirements and Necessary Additions

To start the implementation of the adapter, we first need to extend and make changes to some code infrastructure. We need to read five more strings related to preCICE: the path to the preCICE config file, the participant name, the mesh name, and read and write data names. In addition, we need to extend our bitfield by one more flag indicating a coupling boundary value.

The adapter only needs to be capable of a Dirichlet - Neumann coupling setup, i. e. writing temperature values and reading heat flux values. [Che16] states in her thesis, that Dirichlet - Neumann coupling produces the identical results as the reverse order, it only influences the stability. the Dirichlet - Neumann coupling setup works well for all simulation cases we use as exercises later on and simplifies the configuration and implementation of the solver. The main functions needed for the adapter are the following:

`save_checkpoint()` Saves the velocity fields `U` and `V`, the temperature field `TEMP` and the current simulation time `t`.

`restore_checkpoint()` Restores the velocity fields `U` and `V`, the temperature field texttttTEMP and the current simulation time `t` to the previously saved checkpoint.

`set_vertex_positions()` Calculates the absolute positions of vertices which lay on the coupling interface (s. Figure 4.1).

`write_coupling_data()` Extracts dimensionless temperature values at the coupling interface, transforms them into the dimensional form and sends them to preCICE.

`read_coupling_data()` Reads dimensional heat flux values from preCICE and transforms them into the dimensionless form.

`set_coupling_boundary_values()` Uses the dimensionless heat flux values to set the boundary values at the coupling interface.

The result is Algorithm 3 which can handle explicit as well as implicit coupling. The algorithm closely follows the typical structure of an adapted solver described in Listing

2.2.

Algorithm 3: New Algorithm with preCICE Adapter

Read parameters from file
Assign initial values to u, v, p and T
Set $t := 0$, $n := 0$
Initialize preCICE
`set_vertex_positions()`
while $t < t\_end_{preCICE}$ **do**
    if *action required* **then**
        `save_checkpoint()`
    Select $\delta t := min(\delta t, \delta t_{preCICE})$
    Set all boundary values for u, v and T
    Set problem specific boundary values (inflow, heated walls, etc.)
    `set_coupling_boundary_values()`
    Compute $T^{(n+1)}$
    Compute $F^{(n)}$ and $G^{(n)}$ with $T^{(n+1)}$
    Solve pressure poisson equation iteratively
    Compute $u^{(n+1)}$ and $v^{(n+1)}$
    if *action required* **then**
        `write_coupling_data()`
    Advance preCICE by δt
    if *action required* **then**
        `read_coupling_data()`
    $t := t + \delta t$
    $n := n + 1$
    if *action required* **then**
        `restore_checkpoint()`
    Output u, v, p and T to VTK files if necessary

4.3.2 Dimensional Solver - Non-dimensional Solver Coupling

Special attention has to be given to the temperatures and heat fluxes which are written to and read from preCICE. The lab course solver uses a dimensionless formulation, i. e. the dimensionless temperature T^* and heat flux Q^*. In order to manage a correct coupling T^* has to be first converted to the dimensional temperature T before writing and the dimensional heat flux Q has to be converted to Q^* directly after reading and before setting the boundary value.

$$T^* := \frac{T - T_\infty}{T_\Delta}, \qquad Q^* := \frac{L \cdot Q}{k_s \cdot T_\Delta}, \qquad u^* := \frac{u}{u_\infty}, \qquad \beta^* = \beta \cdot T_\Delta \tag{4.1}$$

- L is the characteristic length scale of the scenario, i. e. 1
- T_Δ is the temperature difference in natural convection setups between the heated and the cooled wall: $T_\Delta := T_H - T_C$.
- T_∞ is the reference or ambient temperature, usually taken as $T_\infty := T_C$ or $T_\infty := (T_H - T_C)/2$
- u_∞ is the reference velocity, usually taken as either the inflow velocity where applicable or set to 1 if the velocity is not known in advance.

4.3.3 Handling of Coupling Boundaries

Temperature values are stored in the center of each cell, for the coupling, however, temperature values at the boundary are needed. The easiest solution is to just use the value from the center of the cell and disregard the error. An example of this concept is shown in Figure 4.1. The temperature value $T_{3,1}$ at position $(2 + \delta x, \delta y)$ is used as if it were located at the vertex position $v_{3,0} = (2 + \delta x, 0)$. We see in the next chapter that this has a negligible impact on the end result of the simulations.

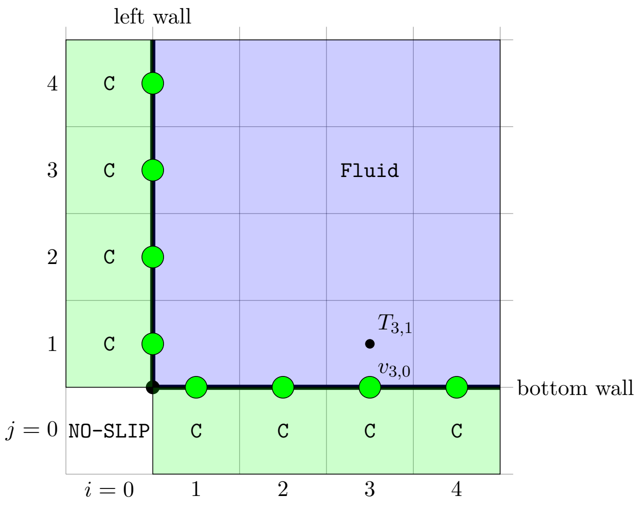


Figure 4.1: Coupling Boundary at Walls

For now, coupling interfaces at the domain boundaries are working fine but the solver is also required to handle coupling interfaces at internal obstacle boundaries. This creates an issue with cells at obstacle corners. Before, there was exactly one vertex associated with one cell which made the implementation easy and straightforward. At obstacle corners, there would be two coupling vertices associated with the corner cell making the calculations for the vertex positions, writing temperature value and setting the boundary values more intricate. Not wanting to add unnecessary complexity, I decided to only allow horizontal coupling interfaces at internal obstacle boundaries as shown in 4.2. This mitigates the negatives while it is still possible to run the 2D heat exchanger simulation shown in 5.2.3.

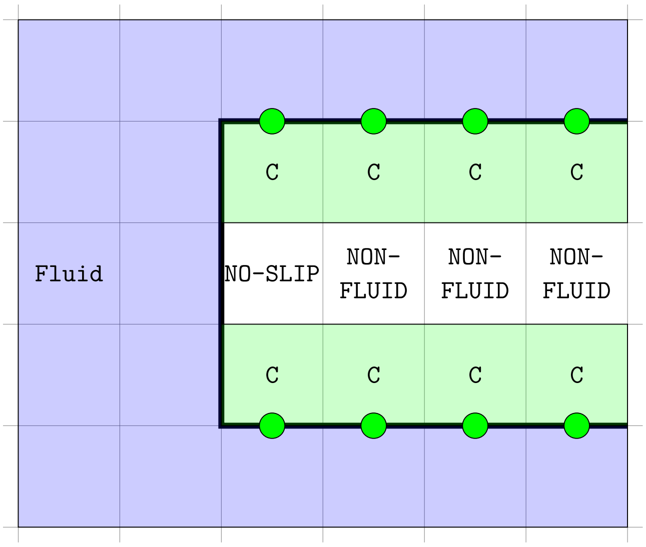


Figure 4.2: Coupling Boundary at Obstacles inside Domain

4.4 Validation of the Implementation

The implementation of the lab course solver is now finished. In this section, I validate the solver using two coupling scenarios covering both forced and natural convection. The reference solutions are obtained by coupling two OpenFOAM solvers, using `laplacianFoam` for the solid and `buoyantPimpleFoam` for the fluid participant. In case of the forced convection scenario, I additionally compare the results to results from the literature and incorporate a mesh study using a number of different mesh resolutions.

4.4.1 Forced Convection: Flow Over a Heated Plate

The flow over a heated plate scenario has already been used to validate the current and previous OpenFOAM adapters [Che16][Cho17].

The setup is described and validated in the literature by Vynnycky et al. [Vyn+98] and shown in more detail in 4.3. The bottom wall of the solid is set to a constant temperature. The fluid inlet is also set to a constant, lower temperature. All other boundaries, except the coupling interface, are adiabatic. Regarding the velocity, the whole top boundary and the bottom boundary up to the leading edge of the solid are set to `slip`, the remaining part to `no-lip`.

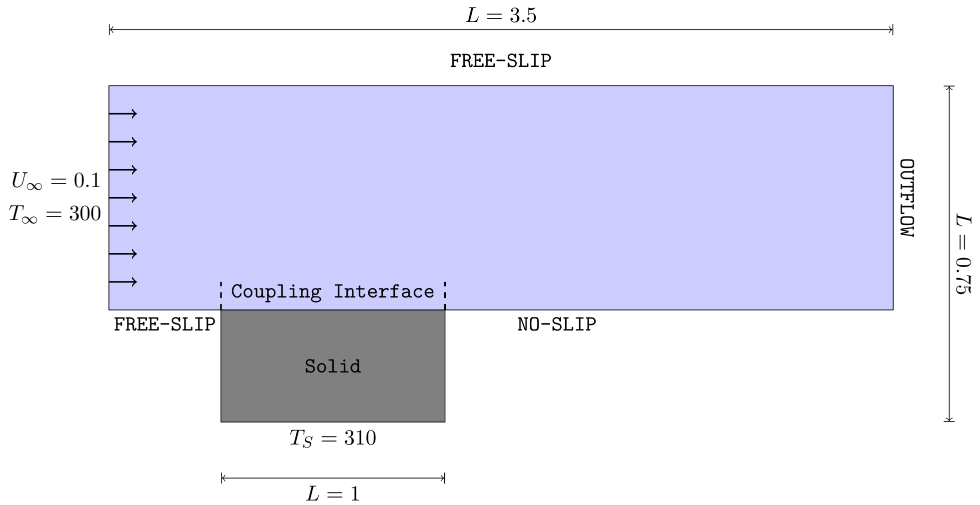


Figure 4.3: Heated flat plate: Geometry and boundary conditions.

Mesh

The mesh resolutions used for the different simulation runs are shown in 4.2. Case `A` is used for both, the OpenFOAM - OpenFOAM reference solution and the Lab Course - OpenFOAM solution. Cases `B`, `C` and `D` are used to investigate how lower resolutions impact the error. As an example, the mesh for case `B` is shown in 4.4. As this is a 2D scenario, the OpenFOAM cases use only one cell in z-direction.

preCICE Configuration

The preCICE configuration is exactly the same for both the reference case as well as the validation case. A serial-explicit coupling scheme with a consistent nearest-neighbor

Case	Fluid	Solid
A	280x80	96x48
B	140x40	48x24
C	105x30	36x18
D	70x20	24x12

Table 4.2: Mesh resolutions

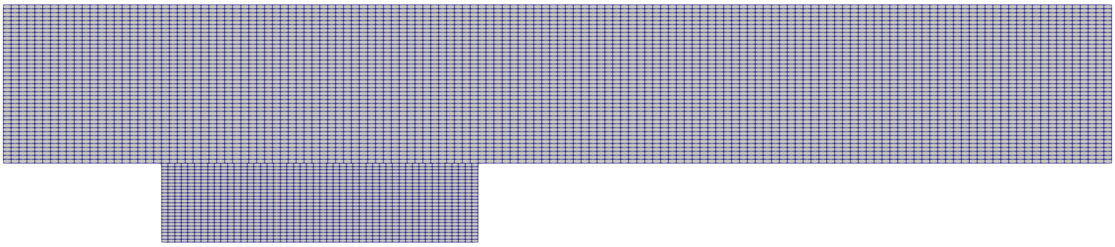


Figure 4.4: Heated flat plate: Mesh B example

mapping for both coupled fields is used. The coupling time step is 0.001, as is the solid solver timestep. This time step leads to subcycling with the Lab Course solver which works well. The maximum simulated time is 20 s.

Material and Flow Properties

I use the first combination of the Reynolds number, the Prandtl number and the conductivity ratio $k = k_s/k_f$ from Vynnycky et al. [Vyn+98] for validation. This refers to $Re = 500$, $Pr = 0.01$ and $k = 1$ and results in the solver parameters shown in Table 4.3.

Results

The dimensionless temperature θ at the interface is plotted against the distance x from the leading edge of the plate.

$$\theta = \frac{T - T_\infty}{T_S - T_\infty} \tag{4.2}$$

Figure 4.5 shows the qualitative comparison of the simulation results. Figure 4.6 shows the dimensionless temperature profile over the coupling interface, comparing it to the OpenFOAM reference case and the literature.

Figure 4.7a shows the results for the dimensionless temperature profile over coupling interface using the different meshes. Figure 4.7b plots the absolute differences in θ in

Parameter	**Symbol**	**Value**
Inlet Velocity	u_{inlet}	0.1
Initial Temperature	T_I	300
Dynamic Viscosity	μ	0.0002
Specific Heat Capacity	c_{pf}	5000
Prandtl Number	Pr	0.01

(a) Fluid parameters for `buoyantPimpleFoam`

Parameter	**Symbol**	**Value**
Inlet Velocity	u_{inlet}	1
Initial Temperature	T_I	30
Reynolds Number	Re	500
Prandtl Number	Pr	0.01

(b) Fluid parameters for the `Lab Course Code`

Parameter	**Symbol**	**Value**
Thermal Conductivity	k_s	100
Density	ρ_s	1
Specific Heat Capacity	c_{ps}	100

(c) Solid parameters for `laplacianFoam`

Table 4.3: Parameters for the heated plate case

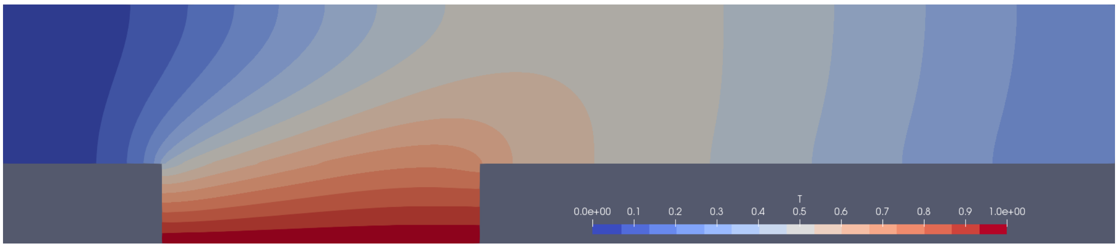
	Mesh A	Mesh B	Mesh C	Mesh D
ℓ^2-norm	0.000705	0.000812	0.001006	0.001616
MSE	0.000048	0.000063	0.000097	0.000251

Table 4.4: Heated flat plate: ℓ^2-norm and mean-squared-error at the coupling interface for different mesh resolutions.

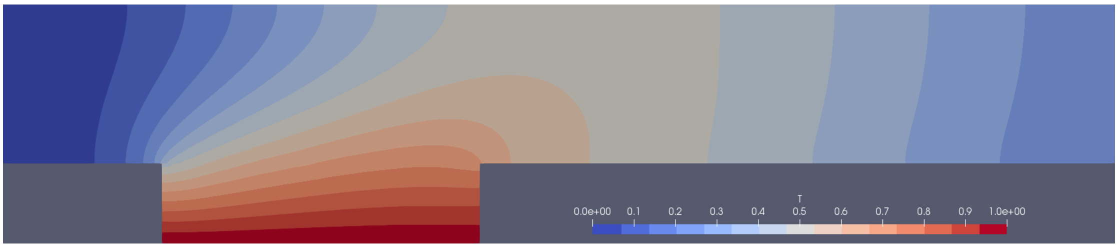
relation to the OpenFOAM reference for each mesh resolution. It shows the large error at the leading and trailing edge of the plate for meshes B, C and D. This is to be expected with the way we set the vertices and handle the coupling boundaries. The quantified errors are listed in Table 4.4. As expected, mesh A results in the smallest error even though the differences in the first third are higher in comparison to mesh B and C.

4.4.2 Natural Convection inside a Cavity with Heat-Conducting Walls

The previous validation case did not take gravity into account. The scenario, natural convection inside a cavity with heat-conduction walls, covers that aspect. The case setup and the used mesh is shown in Figure 4.8. The outside of the left wall of the enclosing is set to a constant temperature $T_H = 310$ while the right side is set to a lower temperature $T_C = 300$. The top and bottom walls of the enclosing are both adiabatic. The initial temperature of both the fluid and the solid is set to 305.



(a) Lab Course - OpenFOAM, Mesh A



(b) OpenFOAM - OpenFOAM, Mesh A

Figure 4.5: Heated flat plate: Comparison of simulation results

preCICE Configuration

The preCICE configuration is exactly the same for both the reference case as well as the validation case. A parallel-implicit coupling scheme with quasi-Newton (IQN-ILS) post-postprocessing and a consistent nearest-neighbor mapping for both coupled fields is used. The coupling time step is 0.01, as is the solid solver timestep. The maximum simulated time is 10 s. An excerpt is shown in

Material and Flow Properties

This case uses the combination $Re = 5000$, $Pr = 0.01$ and $k = 1$. The parameters for the solid participant are identicalto the heated plate case. As there is no reference velocity u_∞ known beforehand, we fix it at 1. This leads to solver parameters shown in Table 4.5. `buoyantPimpleFoam` calculates the coefficient of thermal expansion as $\beta = k/c_p = 0.02$.

Results

The results show the expected behavior. The warmer fluid on the right side moves upwards while the colder fluid on the left side sinks downwards. The comparison of both the x and y velocity components are shown in Figure 4.9. The velocity magnitude, plotted over the horizontal centerline of the fluid part of the cavity is shown in Figure 4.10

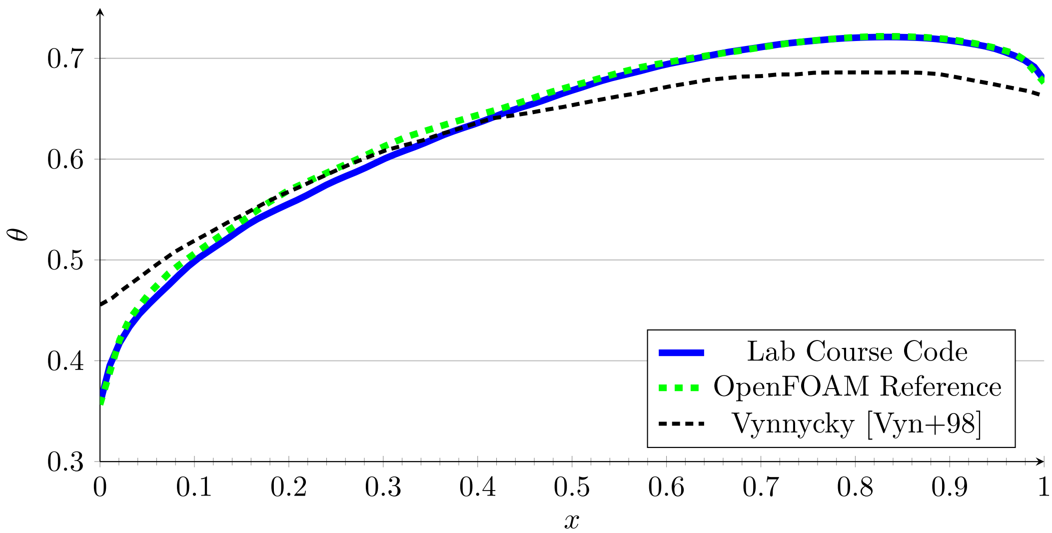


Figure 4.6: Heated flat plate: Dimensionless temperature profile over coupling interface.

Parameter	**Symbol**	**Value**
Initial Temperature	T_I	$305as$
Dynamic Viscosity	μ	0.0002
Specific Heat Capacity	c_{pf}	5000
Prandtl Number	Pr	0.01
Gravity	G_y	-9.81

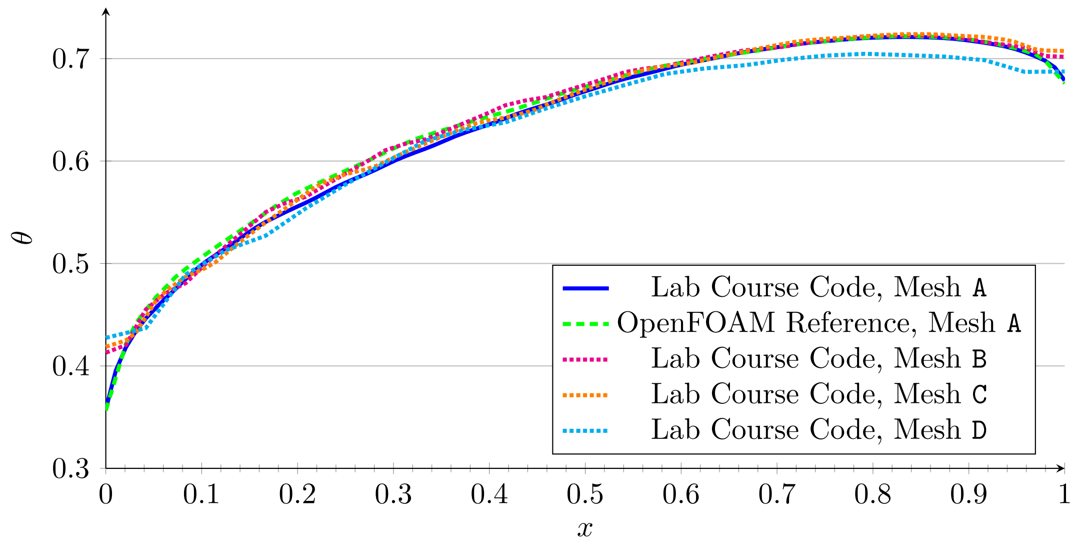
(a) Fluid parameters for `buoyantPimpleFoam`

Parameter	**Symbol**	**Value**
Initial Temperature	T_I	30.5
Reynolds Number	Re	5000
Prandtl Number	Pr	0.01
Thermal expansion	β	0.2
Gravity	G_y	-9.81

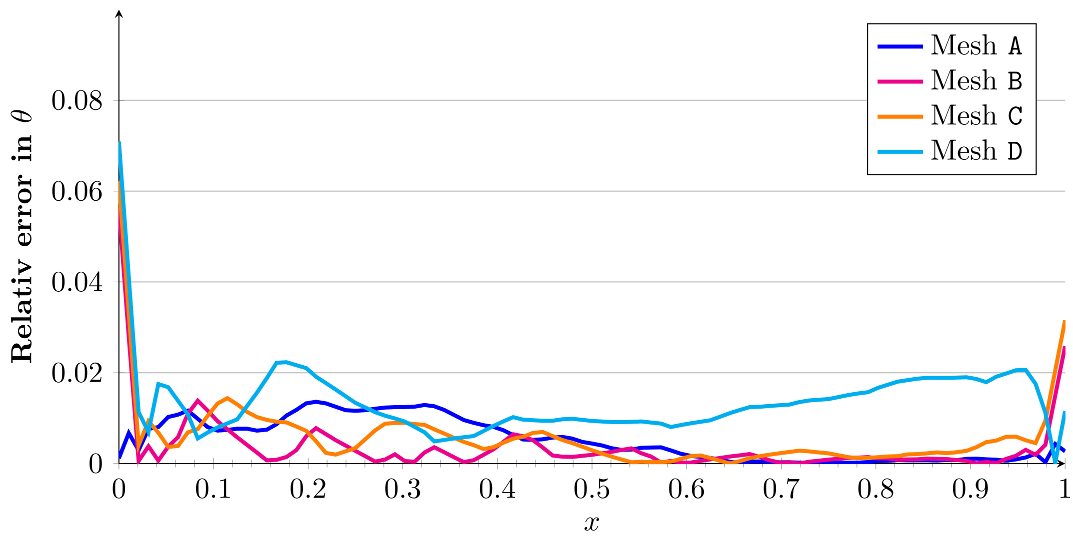
(b) Fluid parameters for the `Lab Course Code`

Table 4.5: Parameters for the natural convection case

The results of the temperature distribution shown in Figure 4.11a and 4.11b also show the expected behaviour. The temperature profile, plotted over the horizontal centerline of the whole cavity, is also nearly identical to the reference simulation. Analogously to the previous case, the non-dimensional temperature T_θ is used. One can see a small discontinuity at the interface locations located at $x = 0.15$ and $x = 0.9$ in the Lab Course Code - OpenFOAM case. This is the consequence of having the temperature values lay at the center point of each cell while using them as if they were laying directly on the coupling interface.

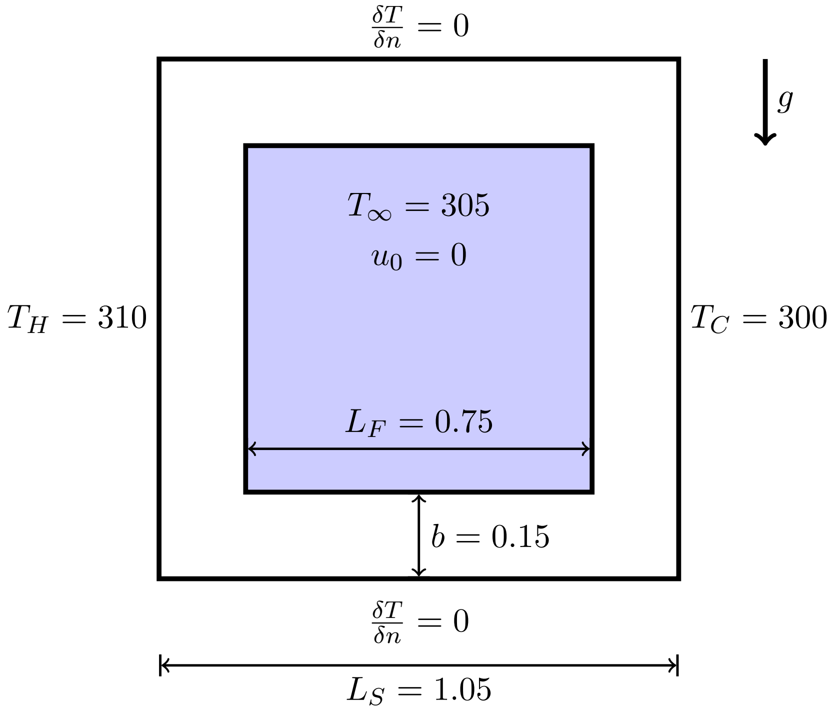


(a) Dimensionless temperature profile over coupling interface.

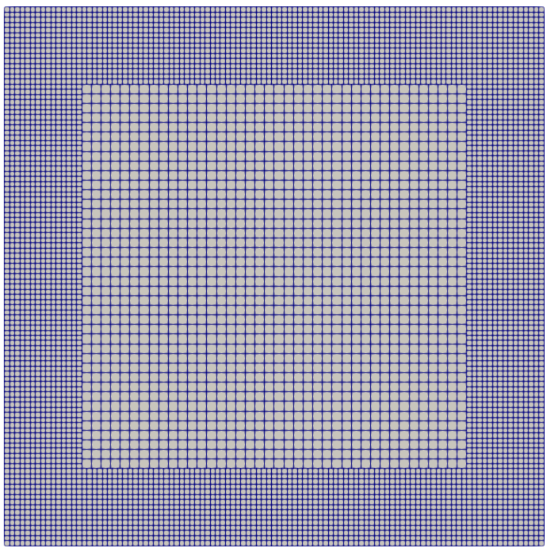


(b) Differences of the dimensionless temperature. The error is the absolute distance to the OpenFOAM reference solution.

Figure 4.7: Heated flat plate: Comparison of different mesh resolutions.

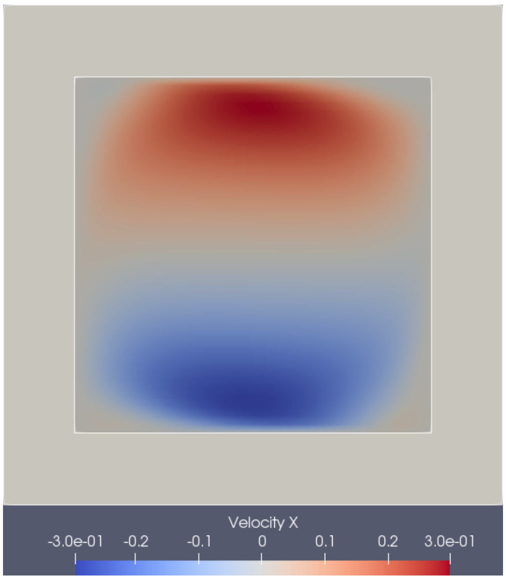


(a) Geometry and Boundary Conditions

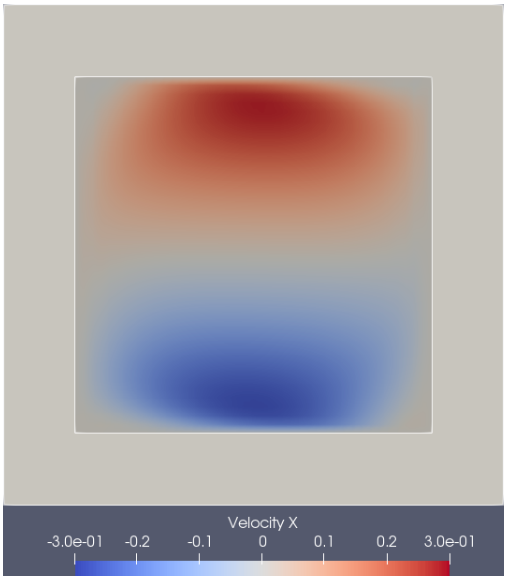


(b) Mesh: 40x40 (Fluid), 105x105 (Solid)

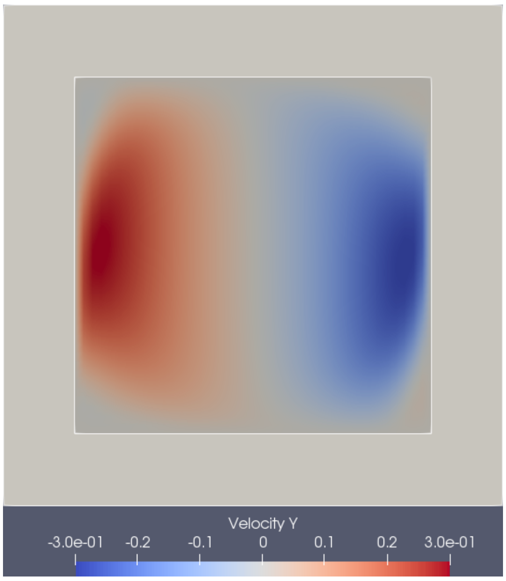
Figure 4.8: Natural Convection inside Cavity



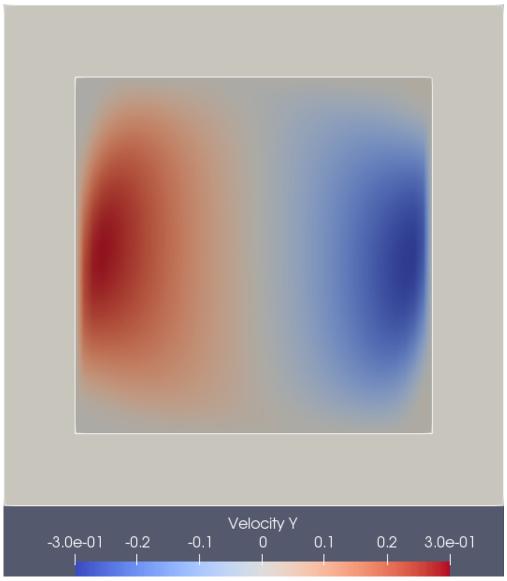
(a) Lab Course - OpenFOAM



(b) OpenFOAM - OpenFOAM



(c) Lab Course - OpenFOAM



(d) OpenFOAM - OpenFOAM

Figure 4.9: Natural Convection inside Cavity: Velocity Comparison

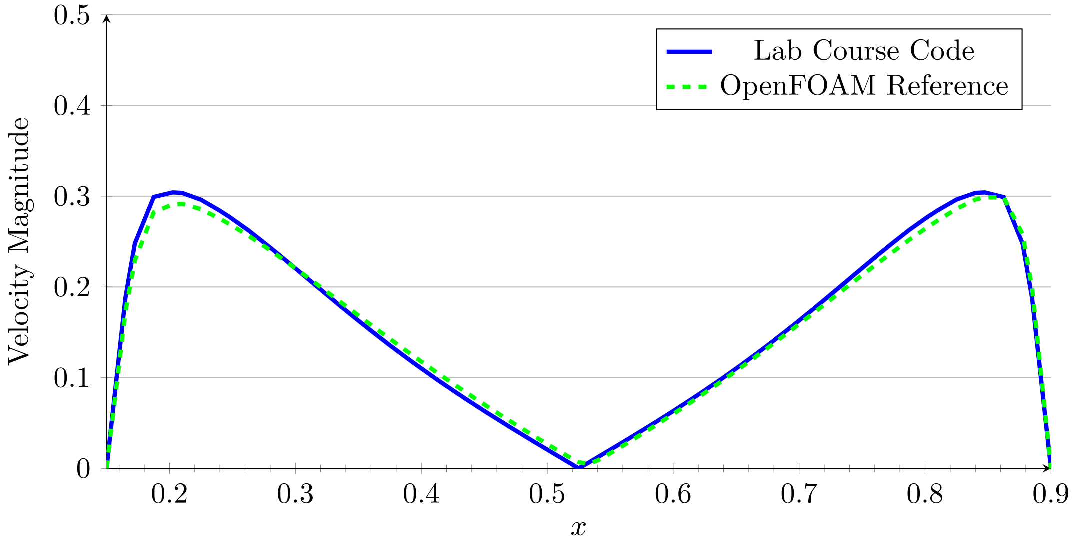
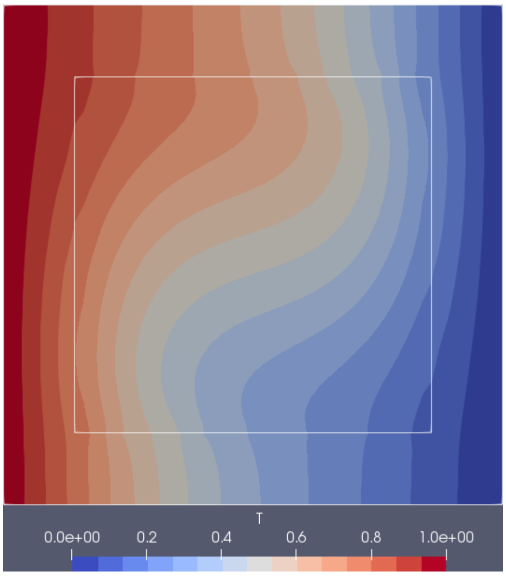
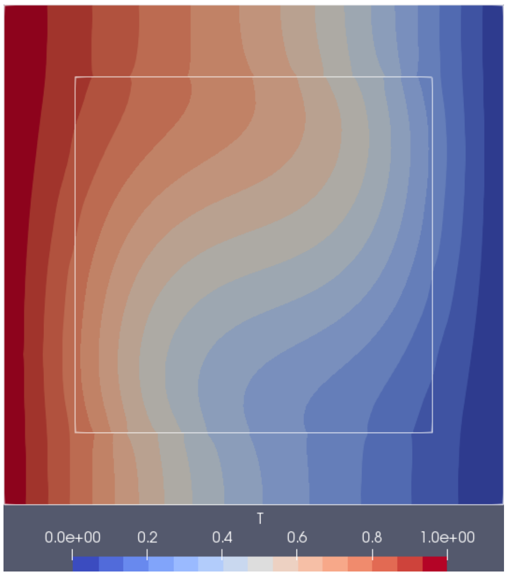


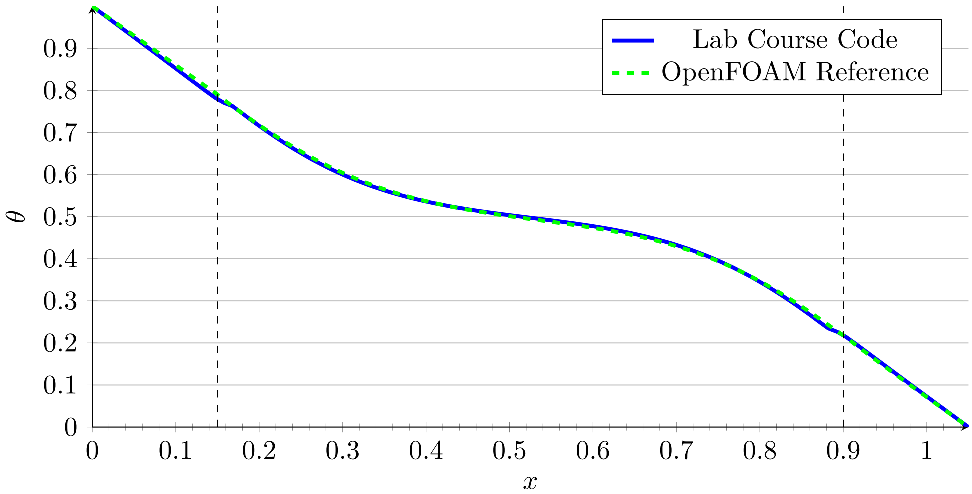
Figure 4.10: Natural Convection inside Cavity: Velocity Magnitude Profile



(a) Lab Course - OpenFOAM



(b) OpenFOAM - OpenFOAM



(c) Temperature profile

Figure 4.11: Natural Convection inside Cavity: Dimensionless Temperature

5 Educational Considerations

It is now clear what the finished solver looks like in the end and that OpenFOAM is being used as the solid coupling partner. In this chapter, we first explain the new lab course structure and why it is chosen that way. Secondly, we present new coupling and non-coupling scenarios which the students simulate with their implementation. At last, we discuss which and what kind of resources the students are given and how much assistance there is in the new and changed worksheets.

5.1 New Lab Course Structure

We now need to decide how to distribute the new content on to the four worksheets and keep each one of it still manageable. The new content can be divided into two modules, the heat transport model (Chapter 4.1) and all matter regarding coupling with preCICE (Chapter 4.3. The latter is relatively self-contained and is thus put onto its own worksheet that got freed up by removing LBM. This leaves the decision on where the implementation of the heat transport model is placed.

Thematically, it fits everywhere except the parallelization worksheet. To justify our decision of putting it on the worksheet together with the arbitrary geometries, we first explain why we did not put it in one of the other two. The first worksheet already requires the students to understand quite a bit of theory regarding fluid dynamics. In addition, they have to figure out logistics with their new team members and get familiar with the provided code framework. The implementation of the heat transport model also requires a more complex handling of the boundary conditions than what is present in the solver from worksheet 1. The worksheet featuring coupling and preCICE also requires the students to understand new theory as well as requiring them to build and install scientific software. The latter can be a major time sink, especially for students with no prior experience in that matter. Given those points, only the arbitrary geometries worksheet remains. The arbitrary geometries worksheet does not require much more theory but the implementation can be quite frustrating and time-consuming as it requires a lot of index manipulations. Nevertheless, it is the best place for the heat transport. The implementation of the heat transport model is in and of itself rather self-contained and can be worked on in parallel to the other tasks of the worksheet. Besides, the students already work on handling the boundary conditions for the velocities and pressure and thus adding temperature boundary values fits right in.

The new arrangement of the worksheets is depicted in Figure 5.1. Worksheet 1 & 3 stay as is. Worksheet 4 contains all subject matter regarding coupling and preCICE. Worksheet 2 now incorporates arbitrary geometries as well as heat transport.

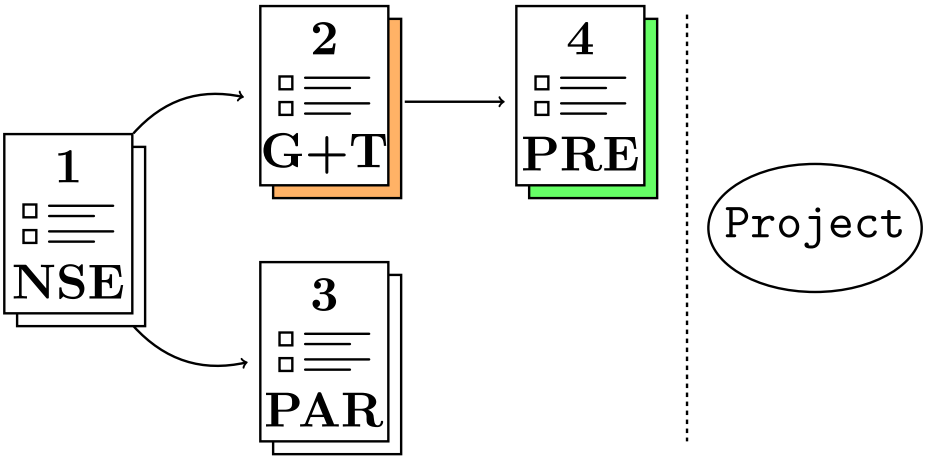


Figure 5.1: The new Lab Course Structure. **NSE** - Navier-Stokes solver worksheet, **G+T** - Arbitrary Geometries and Heat Transport worksheet, **PAR** - Parallelization worksheet, **PRE** - preCICE Coupling worksheet.

5.2 Additional Scenarios

The distribution of the content to the worksheets has now been decided. New flow scenarios for the students to test their code with are still missing. In this chapter, the setup and expected results of three new scenarios are shown. The first two are both directly carried over from [GDN98]. They showcase nicely that even a simple solver is capable of producing interesting phenomena. All three cases presented here are in addition to both validation cases discussed in the previous chapter.

5.2.1 Fluid Trap

The setup is shown in 5.2. Two vertical walls are placed inside the fluid domain. One is connected to the bottom wall, the other one to the top wall. The sum of the heights of the internal walls is greater than the container height. The purpose of this example is to have another natural convection scenario which also makes use of internal geometries. This gives students the opportunity to investigate the flow by experimenting with the height and placement of the walls and changing the hot and cold sides. This is a purely natural convection scenario without coupling and thus is used in worksheet two.

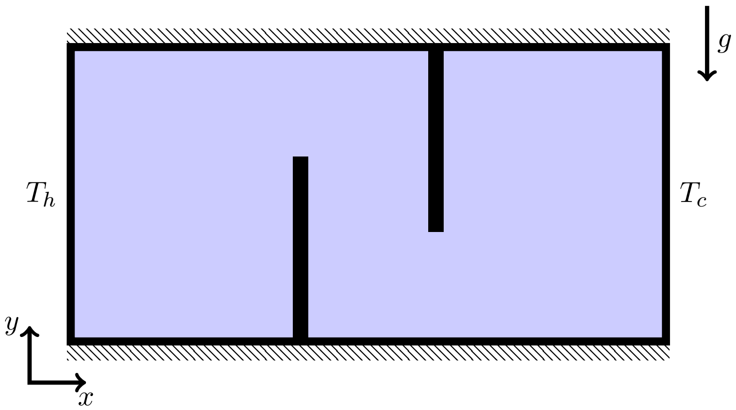


Figure 5.2: Fluid Trap: Scenario setup

The expected results for two different setups are shown in 5.3. The heated and cooled walls are switched between the two.

5.2.2 Rayleigh-Bénard Cells

The simulation setup is shown in 5.4. In this setup, the lateral walls are insulated and both horizontal walls are fixed at different temperatures, cold on top and hot at the bottom. In contrast to natural convection setups with lateral walls at different temperatures where even very small temperature differences lead to motion of the fluid, in this case, the difference has to exceed a critical value any flow starts. Moreover, the lateral walls carrying no-slip conditions introduce three-dimensional effects. The two-dimensional approximation is therefore only valid if the lateral walls are far enough apart for that their influence can be neglected. [GDN98]

The scenario is characterized by the formation of an arrangement of `Rayleigh-Bénard cells`, areas of rising and descending fluid. The number of developed cells is dependent on the used mesh resolution, giving an opportunity to let the students experiment with different setups. This is a purely natural convection scenario without coupling and thus is used in worksheet two. The expected result is shown in Figure 5.5.

5.2.3 2D Heat Exchanger

As heat exchanger scenarios are a prime example for conjugate heat transfer we also wanted to provide a similar setup for the students. All previous coupling examples where between two participants only. This setup makes use of three participants, the two fluids and the horizontal bars, the solid. The two setups shown in 5.6b and 5.6a are overlapping exactly aligned at the for horizontal bars. The red fluid flows from right to left, splits up into two channels where heat is exchanged with the solid and then joins together again at the outflow. The expected result is a hotter outflow than inflow temperature for `Fluid 1` and the opposite for `Fluid 2`. The expected results are shown in Figure 5.7.

5.3 Guidelines and Resources for Students

We now know what the final implementation looks like and what simulation scenarios we use. We also already decided on the distribution of the new content i. e. the new structure of the lab course. As the final task, we have to assess how much and what kind of resources we give the students. In this chapter, we present the applicable resources and discuss what we decided on giving the students. The goal is to provide as much resources for educationally non-relevant parts while only giving just enough resources such that the worksheets can be solved in time without much trouble.

The main types of resources are:

- Configuration files: Be it for preCICE, simulation scenarios or complete OpenFOAM cases.
- Implementation and coding guidelines: New needed functions including their declarations and placement and call site of said functions; Hints on implementation details.
- Expected results for simulations.
- Instructions on building and installing the necessary software.

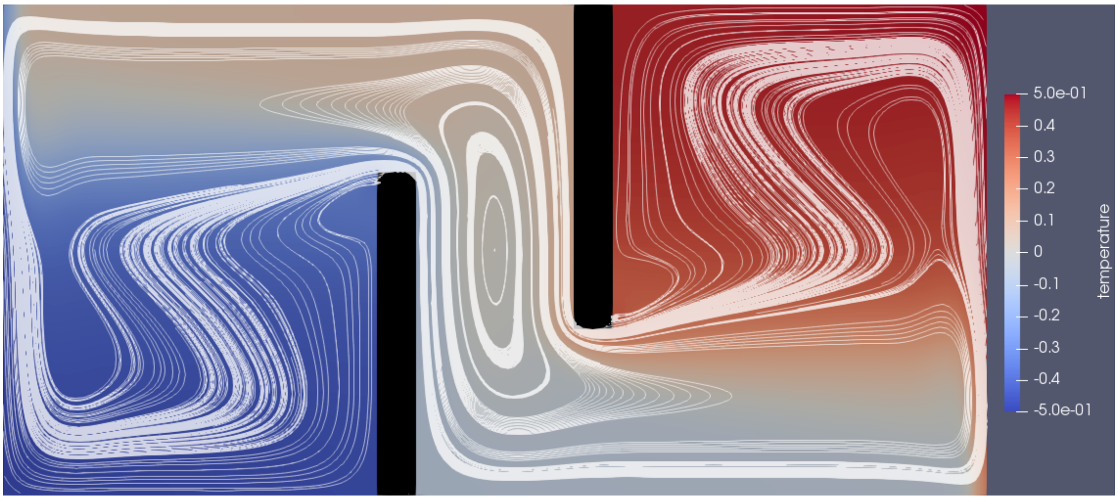
From an educational standpoint, configuration files are the least significant resource. It is easy to justify giving the students all necessary configuration files as there is not much to gain from letting the students write them themselves. Saying that we also want to teach the students about the usage of preCICE. This entails configuring preCICE. For this purpose, we include two configuration files, one for an explicit two-way coupling setup and one for an explicit three-way coupling setup. Thus the students have a frame of reference and can build on top of it to adapt it to an implicit coupling scheme. Examples for that are given on the preCICE wiki.

OpenFOAM cases, in particular, take a long time to set up especially when students are not already familiar with it. Hence we provide all necessary OpenFOAM cases.

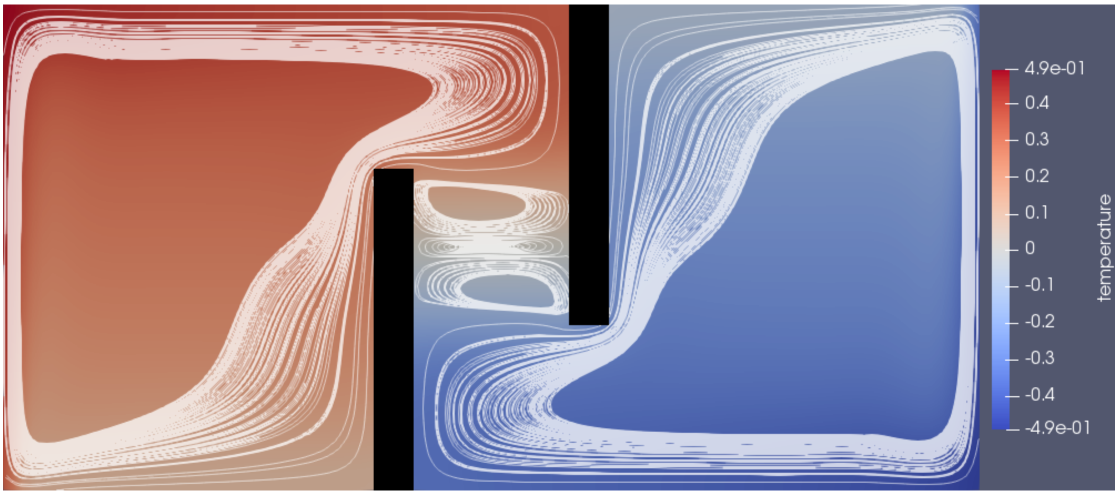
Implementation and coding guidelines are a vital part of ensuring that the students can get started on development without them getting stuck on figuring out the code framework for a prolonged period of time. In addition, giving those guidelines leads to more streamlined solutions making the correction process a lot easier for the instructors. Therefore, we make sure to give the students the necessary function declarations, some hints on where they are needed to be called and where to place their implementation. On the other hand, there is no best solution on how to implement the configuration of the simulation domain, hence we like to leave the details open to the students and only provide a few possible approaches.

We do not provide any expected results for the simulation scenarios. Reason for that is, we want the students to get a feeling for the physics of CFD. They should learn to see when the behavior of the simulations is not physical. If anything they can do internet research themselves to find out what the results should look like. On the other hand, without the expected results the students cannot be a hundred percent certain that what they did is correct.

Instructions on building and installing preCICE, OpenFOAM and the OpenFOAM adapter are provided online on the preCICE wiki. These instructions can never be elaborate enough. Providing a virtual image with everything preinstalled would be another, and actually, the preferred option. The focus of the course is not installing scientific software.



(a) The right wall is heated, the left wall is cooled. Heat can still be freely exchanged between left and right side.



(b) The left wall is heated, the right wall is cooled. Heat exchange only happens in the center at a slow rate.

Figure 5.3: Fluid Trap: Expected results

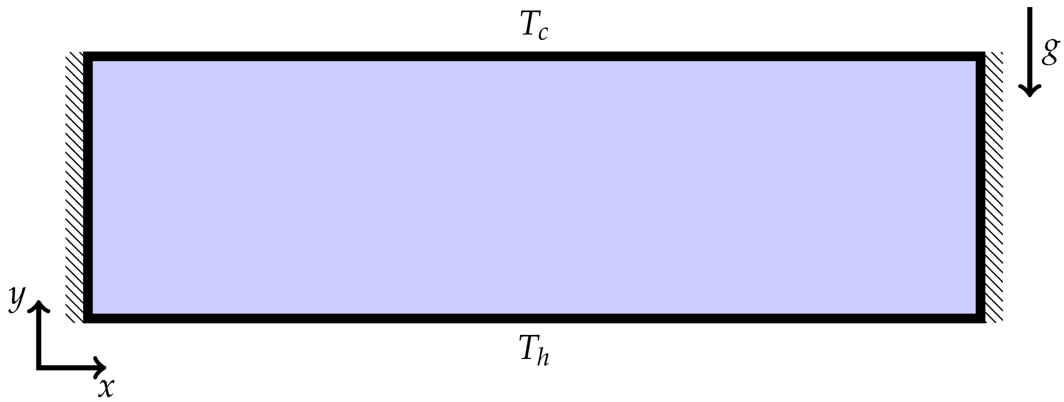
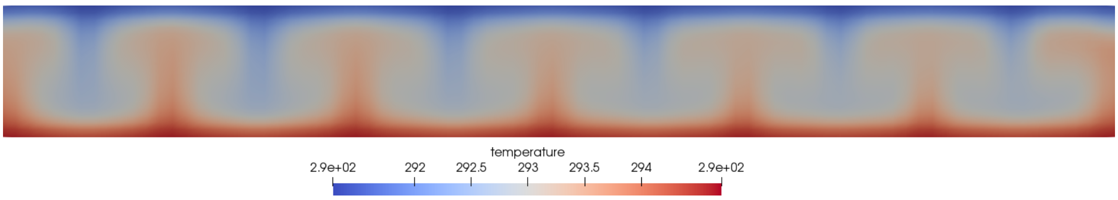
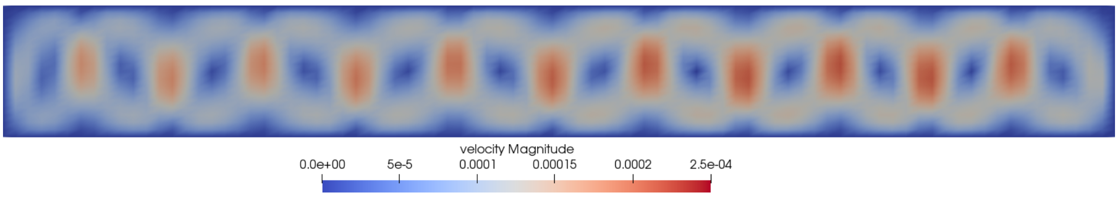


Figure 5.4: Rayleigh-Benard Cells: Scenario setup

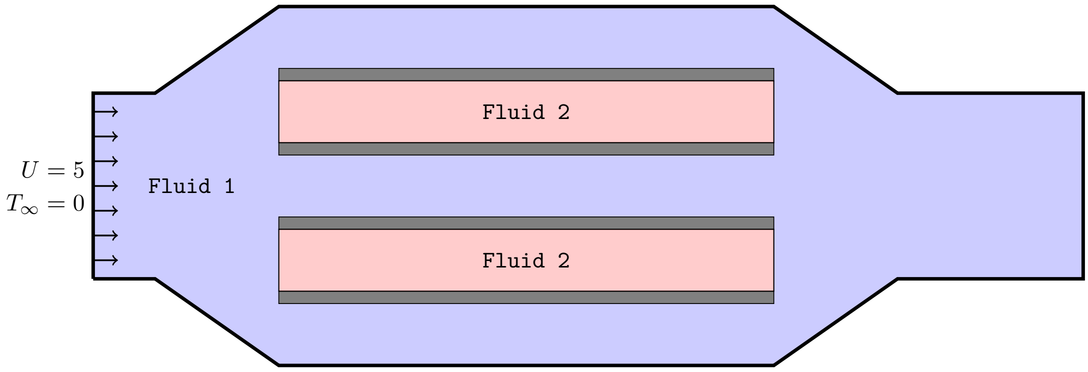


(a) Temperature field

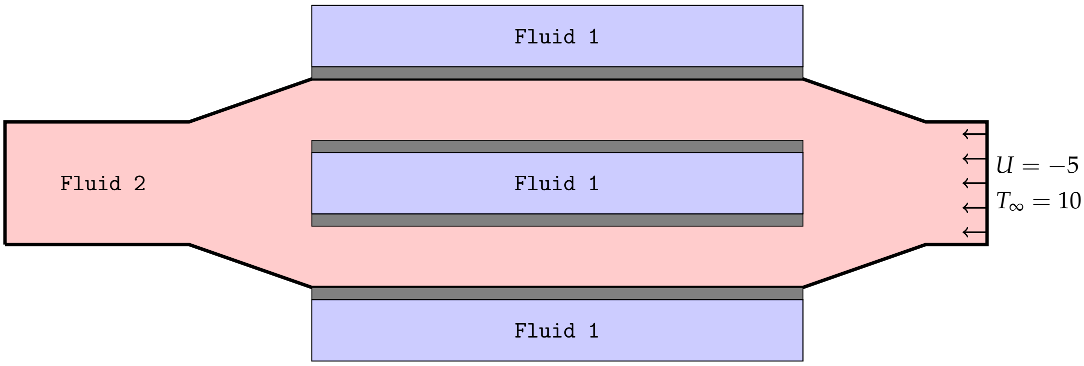


(b) Velocity field

Figure 5.5: Rayleigh-Bénard Cells: Expected results

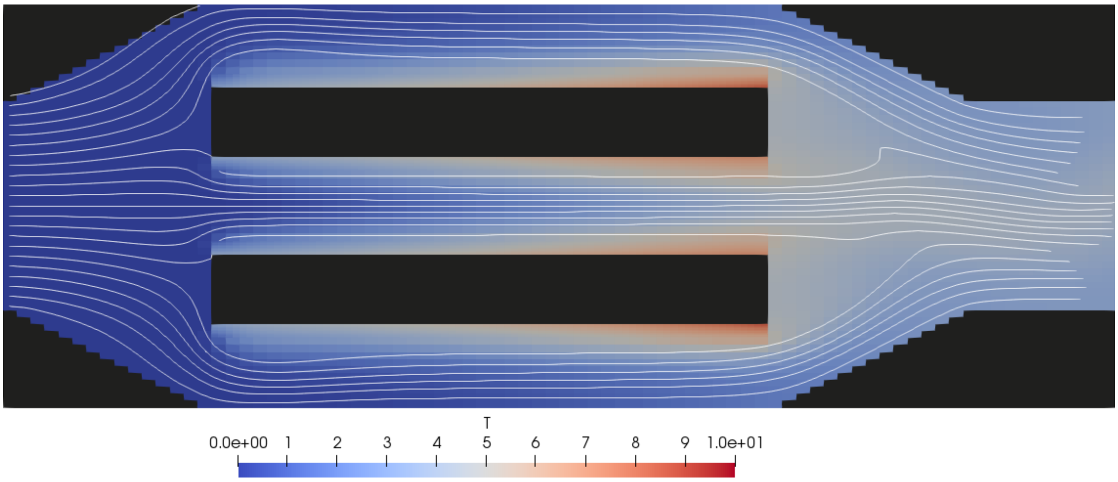


(a) Fluid 1

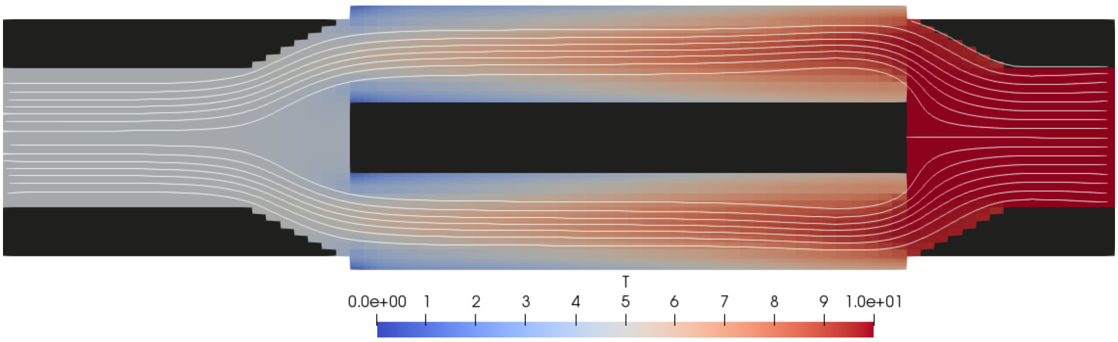


(b) Fluid 2

Figure 5.6: 2D Heat Exchanger: Scenario setup



(a) Fluid 1



(b) Fluid 2

Figure 5.7: 2D Heat Exchanger: Expected Result

6 Evaluation

The new format developed in this thesis was used to teach the CFD lab course at TUM in the summer semester of 2018. Sixteen students divided into teams of three took the course, with the majority of students being enrolled in the Masters Program Computational Science and Engineering and the rest either doing their Master's or Bachelor's in Informatics.

My thesis advisor Benjamin Uekermann kindly provided the anonymous course survey and the anonymized solutions of the students for me to evaluate. The chapter's goal is to assess whether or not the course was successful in conveying the concept of coupled simulations to the students. For this purpose, I first gather and name the main points of the survey and evaluate the student's solutions. Finishing the chapter I discuss the findings and suggest potential improvements for the future.

6.1 Summary of the Student's Feedback

The standardized survey used for evaluating courses at TUM consists of two parts, closed and open questions. The former containing general questions about the course which are scored on a fixed scale. The latter part giving the students space for individual feedback on what they liked and what they think could or should be improved. In this section, I focus mostly on the individual feedback as this seems to be of more significance.

The most notable points made are:

- The overall workload is too much.
- Expected results are not clear enough.
- Order of worksheets is not optimal.
- Installation of the necessary software for worksheet 4 is too time intensive.

On a more positive note, the students also like the following aspects:

- The challenge that the lab course provides.
- The relevance to real applications.
- The multi-disciplinary nature of the lab course.

6.2 Overview of the Student's Solutions

In this part I only consider the solutions for the relevant worksheets, i. e. worksheet two and four. All in all the code structure of each team is fine for both worksheets and it largely sticks to the given instructions.

Worksheet 2

The instructions for the implementation of the configuration and initializing the arbitrary geometries were rather open-ended. The reason for that is that there is no one best solution for this kind of task. As a result, there were a number of different solutions some of which are more flexible than the proposed sample solution.

The end results for both the Rayleigh Bénard Cells scenario and the Natural Convection scenario were correct for all teams. On the other hand, the Fluid Trap scenario was only successfully solved by three out of the five teams.

Worksheet 4

Three teams managed to solve all exercises correctly. One of which also experimented with different parameters and setups for the 2D heat exchanger. Four teams produced kind of correct solutions. The inconsistencies and mistakes can largely be attributed to small off-by-one index errors. Sadly, one team did not get a single correct result.

Group Projects

No team chose to utilize coupled simulations and preCICE in their final project.

6.3 Discussion and Potential Improvements

The handed in solutions are, for the most part, solved quite well. Not all teams managed to get to the correct solution but about half did and one even had time to conduct some extra experiments. This shows us that the difficulty of the new worksheets is adequate.

The preCICE worksheet is, for no particular reason, the last worksheet of the lab course. In hindsight, this does not fit in the natural ordering of the lab course. Going from heat transfer to parallelization back to coupling/heat transfer feels out of order. The easy solution is to just swap the 3rd and 4th worksheet.

Another complaint is, that we do not show the students what the expected result of each simulation is. This, in hindsight, also adds an unnecessary confusion for the students as they can not be sure if they have gotten the correct results yet. An easy solution for the future is to just add pictures of the expected results to the worksheets.

The building and installation of preCICE and OpenFOAM and its adapter was also a problem for a number of students even though there are detailed instructions on the preCICE wiki. It may be worth in the future to consider providing a virtual machine image with the necessary software already preinstalled.

Even though no team chose to use coupling in their projects, we still think it is a worthwhile subject to teach. For the students, using coupling in the projects means that they have to set up OpenFOAM cases themselves which can be quite daunting if there is no previous experience. To mitigate this hurdle, the worksheets or lectures could contain a small introduction to OpenFOAM that explains the process of creating the geometry and setting up the simulation case.

7 Conclusion

The objective of this thesis was to create an educational concept in form of a lab course for the purpose of conveying the method of multi-physics simulations and the usage of the coupling library preCICE over the course of a semester. As there is already a proven CFD lab course in place, though obviously without coupling, it made for an excellent foundation. The previous version of the lab course focused on two different fundamental methods for CFD, the Navier-Stokes Equations (NSE) using finite differences, largely based on the book by [GDN98], and the Lattice-Boltzmann Method (LBM). The need for a different variant of the CFD lab course also arose in part from the fact that there is currently no active research regarding LBM at the Scientific Computing Chair at the TUM. The new variant of the lab course is consequently based on the NSE solver, removing everything concerning LBM and thus freeing up space to use for coupling with preCICE.

Given this information, there had a multitude of decisions to be made. Particularly choosing a demonstrative example where the coupling can be applied. There are a number of very interesting multi-physics phenomena, most notably Fluid-Structure Interaction (FSI), but given the setting, the time frame restriction and overall feasibility, this is not suitable as an example. This lead to choosing conjugate heat transfer (CHT) as it is both feasible to implement and has a wide variety of practical applications.

After settling on this decision there were a number of initial considerations to be made. Most importantly which solver is used as the solid coupling partner and what heat transport model is employed. The book by [GDN98] contains an extensive chapter about energy transport using the Boussinesq approximation. As the lab course code is still based on the code framework given in the book, we decided that the Boussinesq Approximation is our heat transport model of choice. After thinking about different alternatives on which solid coupling partner to use and carefully considering the pros and cons of each one, we finally arrived at OpenFOAM.

We developed a solver capable of being the fluid participant in a coupled CHT simulation always having the scope of the lab course in mind. The implementation was validated using two validation cases both of which have been used by [Che16] and [Cho17] while developing the preCICE OpenFOAM adapter. In addition to those two cases, three more cases were either developed or adapted from [GDN98] to be used as exercise cases for the students. We used this content and divided it up into manageable worksheets. A key contributing factor to this is also the type and the amount of resources which is given to the students.

The CFD lab course, using the new educational concept, was conducted alongside the creation of this thesis in the summer semester of 2018 at the TUM. After completion, we had access to the direct anonymous feedback of the students who took the course and their anonymized solutions for the worksheets.

Gathering the main points from the feedback and examining the solutions we come to the conclusion that the created educational concept generally works, even though no team chose to use coupling in their projects. With a few outliers, all worksheets were adequately solved by the students and it was made known that they liked the practicality of the content. However, the students also noted that adding reference solutions or expected result of the exercises would have helped a lot while working on the worksheets. There were also some complaints about the time investment needed to complete the lab course which we would justify with that it is not a problem inherent to the new variant. The CFD lab course has been labor and time intensive. Using OpenFOAM as the coupling partner of choice is also not the optimal solution and should be thought about again in the future. We think incorporating the potential improvements and thus eliminating some of the downfalls in the future would lead to a really robust and worthwhile educational concept.

List of Figures

List of Tables

Bibliography

[Bun+16] H.-J. Bungartz, F. Lindner, B. Gatzhammer, M. Mehl, K. Scheufele, A. Shukaev, and B. Uekermann. "preCICE – A fully parallel library for multi-physics surface coupling." In: *Computers and Fluids* 141 (2016). Advances in Fluid-Structure Interaction, pp. 250–258.

[Che16] L. Cheung Yau. "Conjugate heat transfer with the multiphysics coupling library preCICE." MA thesis. Institut für Informatik, Technische Universität München, 2016.

[Cho17] G. Chourdakis. "A general OpenFOAM adapter for the coupling library preCICE." MA thesis. Institut für Informatik, Technische Universität München, 2017.

[DW18] G. Dhondt and K. Wittig. *CALCULIX. A Free Software Three-Dimensional Structural Finite Element Program.* Apr. 29, 2018. URL: `http://www.calculix.de` (visited on 09/10/2018).

[Fou] T. O. Foundation. *OpenFOAM. The open source CFD toolbox.* URL: `https://www.openfoam.com` (visited on 09/10/2018).

[GDN98] M. Griebel, T. Dornseifer, and T. Neunhoeffer. *Numerical Simulation in Fluid Dynamics: A Practical Introduction.* Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 1998.

[Vyn+98] M. Vynnycky, S. Kimura, K. Kanev, and I. Pop. "Forced convection heat transfer from a flat plate: the conjugate problem." In: *International Journal of Heat and Mass Transfer* 41.1 (1998), pp. 45–59.