Inaugural-Dissertation

zur
Erlangung der Doktorwürde
der
Naturwissenschaftlich-Mathematischen Gesamtfakultät
der
Ruprecht-Karls-Universität
Heidelberg

vorgelegt von
Diplom-Mathematikerin Bärbel Janssen
aus Aurich

Tag der mündlichen Prüfung: _____

Adaptive Coupling of Finite Element Methods for Simulation of Hydrodynamics and Pollutant Transport in Lakes

Gutachter: Prof. Dr. Dr. h. c. Rolf Rannacher

Dedicated to my father,
who was never able
to read this thesis.

Abstract

Subject of this work is the development of new numerical methods for the solution of problems occurring in hydrodynamics of lakes. The computation of transport processes of pollutants demands to resolve sharp fronts accurately. This requires a high accuracy in certain parts of the domain.

To achieve the necessary accuracy and to keep the computational costs moderate, we do not resolve the whole three dimensional domain. In parts of the domain in which the required accuracy is marginal, a two dimensional solution is sufficient. In those parts of the domain in which a better accuracy is to be achieved, we add a three dimensional correction to the two dimensional solution. This way, we accomplish a more accurate, three dimensional solution in certain parts of the domain with moderate additional expenses. We derive the equations that arise according to the coupling of different dimensions.

As a preconditioner of the coupled system, we apply a block preconditioner. For each of the diffusion dominated blocks, we developed a multilevel preconditioner for continuous finite elements on adaptively refined meshes. The smoothing is only applied locally. By the means of numerical examples we show the efficiency for higher order finite elements.

Zusammenfassung

Gegenstand dieser Arbeit ist die Entwicklung neuer numerischer Methoden zur Lösung von Problemen der Hydrodynamik in Seen. Für die Berechnung von Transportprozessen von Schadstoffen ist es wichtig Fronten scharf aufzulösen. Dies erfordert eine hohe Genauigkeit in bestimmten Bereichen des Gebiets.

Um die erforderliche Genauigkeit zu erreichen und gleichzeitig die Kosten bei der Berechnung moderat zu halten, lösen wir das dreidimensioale Gebiet nicht überall komplett auf. In den Teilen des Gebiet, in denen nur geringe Genauigkeit gefordert wird, genügt eine zweidimensioale Lösung. Für die Bereiche in unserem Gebiet, in denen wir bessere Genauigkeit erzielen wollen, addieren wir zu der zweidimensionalen Lösung eine dreidimensionale Korrektur. Auf diese Weise erreichen wir in gewissen Teilen des Gebiets eine genauere, dreidimensional Lösung bei moderatem Mehraufwand. Die Gleichungen, die durch diese Kopplung entstehen, werden hergeleitet.

Für die Vorkonditionierung des gekoppelten Systems verwenden wir einen Block-Vorkonditionierer. Für die einzelnen Blöcke haben wir einen Mehrgitter-Vorkonditionierer für stetige Finite Elemente auf adaptiv verfeinerten Gittern entwickelt. Dabei geschieht die Glättung nur lokal. Anhand von numerischen Beispielen zeigen wir die Effizienz für Elemente höherer Ordnung.

Contents

1 Introduction **1**

2 Basic notations **7**
2.1 Basic notation 7
2.1.1 Sobolev spaces on a fixed domain 7
2.1.2 Continuous function spaces 8
2.1.3 Sobolev spaces on a time dependent domain 8

3 Equations **9**
3.1 Basic equations describing fluid flows 9
3.1.1 Navier-Stokes equations 9
3.1.2 Boussinesq approximation 11
3.1.3 Variational formulation of the Boussinesq equations 12
3.1.4 Shallow-Water equations 13
3.1.5 Variational formulation of the Shallow-Water equations 14
3.2 Equations for free surface flows 14
3.2.1 Arbitrary Lagrangian Eulerian (ALE) formulation 15
3.2.2 Spacial and temporal derivatives in ALE formulation 15
3.2.3 Construction of the ALE mapping 16
3.2.4 Equations formulated in ALE framework 18

4 Coupling of models in 2D and 3D **19**
4.1 Coupling of 2D and 3D Poisson problem 19
4.1.1 Decomposition of the space W 20
4.1.2 Discussion of the coupling terms 21
4.1.3 Discussion of the boundary term 22
4.1.4 Numerical results 23
4.1.5 Discussion of the numerical results 29
4.2 Coupling of 2D Shallow-Water and 3D Boussinesq model 30
4.2.1 First equation 31
4.2.2 Second equation 32
4.2.3 Discussion of the linear coupling terms 32
4.2.4 Discussion of the non-linear coupling terms 35

5 Discretization **43**
5.1 Discretization in time 43
5.1.1 Preliminaries on time stepping schemes 43
5.1.2 Temporal regularity 48

5.1.3 Temporal discretization of the Boussinesq equations . . . 50
5.1.4 Temporal discretization of the Shallow-Water equations . . . 51
5.1.5 Temporal discretization of the Boussinesq equations in ALE formulation 53
5.2 Issues on meshes . . . 54
5.2.1 Splitting of adaptive meshes into levels . . . 54
5.2.2 Hanging nodes . . . 58
5.3 Discretization in space . . . 59
5.3.1 Galerkin finite element discretization . . . 60
5.4 Stabilization . . . 62
5.5 Coupling discretization in 2D and 3D . . . 63
5.5.1 Connecting meshes in 2D and 3D . . . 63

6 Algorithm **67**
6.1 Basic algorithms . . . 67
6.1.1 Newton's method . . . 67
6.1.2 Solving linear systems . . . 67
6.1.3 Block preconditioning with Schur complements . . . 68
6.2 Multilevel algorithm . . . 69
6.2.1 Splitting of level spaces . . . 71
6.2.2 Prolongation and restriction . . . 72
6.2.3 Local smoothing . . . 73
6.2.4 Overlapping Schwarz smoother . . . 74
6.2.5 Complexity of the algorithm . . . 75
6.2.6 Memory requirements . . . 76
6.2.7 Numerical experiments . . . 76
6.3 Details on the implementation . . . 81
6.3.1 Fully discrete coupled problem . . . 81
6.3.2 Integration of coupled terms on active cells . . . 82
6.3.3 Integration of coupled terms on inactive cells . . . 83

7 Numerical results **85**
7.1 Numerical example . . . 85
7.2 Initial and boundary conditions . . . 87
7.3 Choice of parameters . . . 87
7.4 Discussion of numerical results . . . 94

8 Conclusions **95**
8.1 Conclusion . . . 95
8.2 Outlook . . . 95

Acknowledgments **97**

List of Figures **99**

List of Tables **101**

List of Algorithms **103**

Bibliography **105**

1 Introduction

The prediction of water quality requires a good knowledge of the mechanisms of free surface flows. Although the technical progress allows us to solve more and more complex problems, we have to ensure that this is possible on today's computers.

The purpose of this work is to develop efficient algorithms for simulations of hydrodynamics in lakes. To predict the water quality it is important to use the most complex model to capture sharp fronts in transport processes of pollutants, for example. We restrict ourselves to carry out the simulations on a desktop computer. This means we have to come up with a fast solver on the one hand and a way to achieve the accuracy of the most complex model on the other hand.

One way to reduce the computational costs but to keep the required accuracy at the same time, is to use adaptivity. In this thesis we solve the complex problem in a monolithic formulation. This means the whole system of nonlinear coupled equations has to be solved all at once. This will be the starting point to apply a posteriori error analysis for goal oriented mesh adaptivity.

It is very costly to solve a fully coupled nonlinear time dependent problem in a three dimensional space. We focus also on how to reduce the computational cost in various ways.

Physical phenomena are often localized in specific portions of the domain. For the mathematical modeling of these phenomena it is reasonable to utilize a model that is computationally cheap enough, but sufficiently accurate at the same time. The idea behind this approach is to use the full three dimensional model only in areas of the domain where a better approximation of the solution is necessary to accurately solve for the unknown.

A novel strategy to couple a three dimensional model with a model in two dimensions is introduced. We define the part of the domain that is solved with the more complex model a priori. One can also think of applying an automatic tool to choose the area of the domain where the three dimensional model has to be applied.

There has been work in this direction for a systematic model reduction based on a posteriori error estimates as already realized in the context of reactive flow models [15]. Model coupling techniques using a posteriori error analysis were presented in [14], [46], [47] and [52]. Our approach differs from the techniques presented in those articles since we couple different models in various finite element spaces. The equations that arise due to this kind of coupling are derived.

One main issue of this work is the geometric multilevel preconditioner using continuous finite elements on adaptively refined meshes. The novelty of this algorithm is its applicability to adaptive meshes with hanging nodes as well as higher order finite elements. Due to the local

smoothing of overlapping Schwarz type it has optimal computational complexity. This makes the preconditioner well suited for solving three dimensional complex coupled problems in reasonable time.

In our setting the multilevel preconditioner is used in a block-Schur type preconditioner for inverting diffusion dominant blocks.

The formulation of the fully coupled nonlinear system is presented and the technical details on how to implement this in a finite element code are described.

Simulations for hydrodynamics in lakes

When dealing with hydrodynamics a crucial point is the model used for simulations. The flow and the transport of particles in lakes is described by the general Navier-Stokes equations for the fluid and thermodynamics and a transport equation for the species concentration in the full three dimensional domain.

$$\begin{aligned} \partial_t v + (v\cdot\nabla)v - \frac{1}{\varrho}\nabla\cdot\sigma - \gamma \boldsymbol{g}\varrho(T) &= \boldsymbol{g} + f, \\ \frac{1}{\varrho}\partial_t \varrho + \nabla\cdot v &= 0, \\ \partial_t T + v\cdot\nabla T - \nabla\cdot(\nu_T \nabla T) &= q_T, \\ \partial_t S + v\cdot\nabla S - \nabla\cdot(\nu_S \nabla S) &= q_S, \\ \varrho &= \varrho(T,p). \end{aligned} \tag{1.1}$$

where the stress tensor is given as

$$\sigma = -p\,\mathrm{Id} + \varrho\nu(\nabla v + \nabla v^T).$$

The variables v, p, T, S stand for velocity, pressure, temperature, and species concentration, respectively. The coefficients ν, γ and ν_T denote the kinematic viscosity, volume expansion, and thermal diffusivity of water, respectively. $\boldsymbol{g}$ denotes the negative acceleration of gravity.

A reduced model called Shallow-Water model for v and p can be derived from the first two equations in (1.1). This is vertically averaged and therefore only two dimensional. The equations are given as

$$\begin{aligned} h\partial_t w_{2D} + h(w_{2D}\cdot\nabla)w_{2D} - \nu\nabla\cdot(h\nabla w_{2D}) + gh\nabla h &= F, \\ \partial_t h + w_{2D}\nabla h + h\nabla\cdot w_{2D} &= 0. \end{aligned} \tag{1.2}$$

The unknowns are the vertically averaged velocity $w_{2D} = (w_{2D}^1, w_{2D}^2)$ and the height h. The pressure of the Navier-Stokes equations (1.1) has dropped out and can be expressed in terms of the height:

$$p = \varrho g(h - x_3).$$

The main drawback of two dimensional models is the accuracy. Vertical mixing is not taken into account and due to the fact of pressure depending only on the weight of the column of

water in the Shallow-Water model we will not get satisfying results. To capture sharp fronts in transport processes a resolution in three dimensions is required. This means we have to use the full Navier-Stokes equations (1.1) for a better resolution.

A coordinate transformation form Cartesian coordinates to so called σ coordinates was introduced in [11]. This makes the domain movement in three dimensional free surface flow possible. It is extensively used e.g. in [63]. We follow a different approach as is explained in [29].

In order to solve the Navier-Stokes equations on the time dependent domain we apply a transformation to a fixed reference domain. This technique is called ALE formulation and is well established for solving fluid structure interaction problems, see [31], [56], and [37]. Using this approach we end up with a monolithic formulation of the problem. This means we do not use an operator splitting but rather get a fully coupled system to be solved.

There are several groups that work in this field, e.g. [36], and also commercial codes are available: Delft3D [30] and Telemac [63]. In the former SFB 359 "Reactive Flows, Diffusion and Transport" there was a project in which numerical simulations were done for lake Willersinn [70].

Shortcomings in existing numerical approaches are unsatisfactory accuracy and enormous computing times. Either the mathematical model used is too much simplified or the discretization is too coarse to capture local effects properly. Alternatively some of the existing software based on the full model uses large parallel computers to cope with the huge computational costs.

The idea would be to develop code for supporting limnologists to take their measurements. This is the reason why in this thesis we develop new numerical tools for performing numerical simulations with sufficient accuracy on desktop computers.

Solving discrete problems

The fully coupled problem in 3D has to be solved efficiently. The resulting linear problems are preconditioned by so called block preconditioners introduced by Elman, Silvester, and Wathen for Stokes problems [32].

As part of this block preconditioner we present a multilevel preconditioner introduced in [39] for H^1- and H^{curl}-conforming high order finite element methods. We apply it to the diffusion dominated blocks that occur in the linear block systems.

In this thesis we only describe the application to continuous higher order finite element methods on adaptively refined meshes. We restrict smoothing only to the most refined part of the domain. We note that qualitatively, it is clear from work by Bramble [17], Griebel and Oswald [34], and Xu [73], that local smoothing, if correctly implemented, yields convergence rates independent of the mesh size. Nevertheless, it is a priori not clear whether the constants in these estimates deteriorate in the presence of refinement edges. Furthermore, the estimates are usually not uniform in the polynomial degree of the finite element shape functions. In Section 6.2.7, we provide numerical evidence that these rates are quantitatively not worse than those obtained on uniform meshes and depend only weakly on the polynomial degree.

We discuss the artificial boundary conditions that have to be imposed on the boundary between two regions of the domain with different refinement levels. For the application to H^{curl}-conforming high order finite element methods we refer to [39].

When we consider adaptively refined meshes for multigrid methods, we have to distinguish between meshes with hanging nodes and conforming meshes. The latter can either be generated by red-green refinement [57] or bisection [48, 61], and often appear with simplicial meshes. In that case, the question of dealing with hanging nodes does not arise. Local adaptive multigrid methods for these meshes can be found for instance in [6–8, 58].

On meshes with hanging nodes, several types of optimal multilevel preconditioners have been devised. These are:

1. Local smoothing *inside* the region of local refinement, but not at the interface between refined and coarser region was introduced by Brandt [19] for finite difference methods. In McCormick's monograph [45], this method is discussed for the finite volume element method under the name of *multilevel composite grid scheme*. These methods are used in applications that allow for dynamically changing meshes in the multigrid procedure [65]. We extend these methods to higher order finite element methods in the H^1-conforming case.

2. Local smoothing in the region of local refinement, including the interface between fine and coarse cell and the support of basis functions associated with node values on the interface [9]. In [45], this scheme is applied to finite volume methods as *bordered multilevel scheme*.

3. The method of global coarsening avoids the problem of setting a border to the subdomain for smoothing by introducing a hierarchy of level spaces where each space covers the whole computational domain. By choosing the levels carefully, these methods still maintain optimal complexity. They share the drawbacks of the bordered scheme, but offer advantages in problems with global constraints like the zero mean value property of the pressure in incompressible flow problems. This method has been applied for instance in [10, 12, 41, 42, 60]. In [9], it is compared to the bordered scheme. Global smoothing does not yield optimal complexity [8] of the multilevel algorithm but in this case no artificial boundary has to be created.

The usage of this multilevel preconditioner speeds up the computation, especially when dealing with a three dimensional domain. In these cases the use of a direct solver is no longer feasible.

Coupling models in different dimensions

To reduce the high computational costs we present a novel approach in dealing with fluid flows. The technique we present has been used in solid mechanics, see [54]. The idea is to decompose the solution of the problem in 3D into a sum as

$$u = u_{2D} + u_{3D}, \tag{1.3}$$

where u_{2D} is a solution of the Shallow-Water model and u_{3D} is understood as a correction which is only computed in certain parts of the domain in 3D. This approach allows us to prescribe areas of the 3D domain where a higher accuracy is needed. In those areas where a 2D solution is sufficiently accurate the Shallow-Water model is used.

To be more precise, we do not only split the function u as in (1.3), but the whole space in which we are looking for a solution of our problem described using a semi-linear form $a(u)(\varphi)$. When the original problem reads: Find $u \in V$ such that

$$a(u)(\varphi) = (f, \varphi) \text{ for all } \varphi \in V,$$

the coupled problem reads: Find $u_{2D} \in V_{2D}$ and $u_{3D} \in V_{3D}$ such that

$$a(u_{3D} + u_{2D})(\varphi_{3D} + \varphi_{2D}) = (f, \varphi_{3D} + \varphi_{2D}) \tag{1.4}$$

for all $\varphi_{2D} \in V_{2D}$ and for all $\varphi_{3D} \in V_{3D}$. The subscript indicates that V_{3D} consists of functions on a domain in $\mathbb{R}^3$, whereas functions in V_{2D} are defined on two dimensional domain.

As a first step we apply this coupling to a test case, namely the coupled Laplace problem. We derive the equations in two dimensions as well as in three dimensions. Numerical tests are carried out to show the convergence of this approach by means of the gained numerical results.

In a next step we adopt the same techniques to the generalized Navier-Stokes problem (1.1). The derivation of the coupling terms in this case is a little more crucial due to the nonlinear terms than for the linear Laplace problem.

Implementational aspects

As can be seen in (1.4) the coupling of models in 2D and 3D requires to discretize forms and residuals such as

$$a(u_{2D})(\varphi_{3D}), \quad a(u_{3D})(\varphi_{2D}), \quad (r_{2D}, \varphi_{3D}), \quad (r_{3D}, \varphi_{2D}). \tag{1.5}$$

From a computational point of view we explain how the coupling is realized in a framework of a finite element software library. The concept of the MeshWorker in deal.II offers the possibility to incorporate the coupling between meshes, cells and functions, etc.

We explain the general use of the MeshWorker and based on the concept of the MeshWorker we present how the implementation of the coupling can be done in an elegant way.

For example we explain how the meshes in 2D and 3D are connected and how refinement is realized such that it “matches” between the meshes in different dimensions.

The outline of this thesis is as follows:
In Chapter 2 we introduce the basic notation that will be used throughout this thesis. In Chapter 3 we present the governing equations and derive the equations we will later use.

In Chapter 4 the new technique for coupling of different models is explained for the Laplace problem as well as for the equations describing fluid flows.

In Chapter 5 the discretization of the coupled variational problem is introduced. Principal aspects of the discretization in space and time are explained. Furthermore we describe the coupling of the meshes.

In Chapter 6 the algorithms used for the whole solution process are provided. The multilevel algorithm is explained in detail.

We end this thesis with Chapter 8. The presented results are summarized and possible extensions and future work are pointed out.

2 Basic notations

In this chapter we briefly introduce the notation used throughout this thesis. Furthermore we present the requirements for a discretization of our computational domain Ω.

2.1 Basic notation

In this section we introduce the notation we use in this thesis. Let $\Omega(t) \subset \mathbb{R}^3$ be an open bounded time dependent domain with boundary $\partial\Omega(t)$. Due to the free surface we allow the boundary at the top to be time dependent. In this thesis we restrict ourselves to fixed vertical boundaries and a fixed bottom. The projection of the time dependent domain $\Omega(t)$ onto the horizontal plane is denoted by Ω_{2D}. This two dimensional domain is no longer time dependent since we only allow the surface of $\Omega(t)$ to evolve in time. The boundary of $\Omega(t)$ is assumed to be Lipschitzian for all $t \in I$. The outer unit normal vector is denoted be $n(t)$.

2.1.1 Sobolev spaces on a fixed domain

For a fixed domain in time Ω we denote the space of measurable functions which are Lebesgue-integrable to the p-th power as $L^p(\Omega), 1 \leq p \leq \infty$. For the special case $p = 2$ the space $L^2(\Omega)$ becomes a Hilbert space with the inner product

$$(u, v)_{L^2(\Omega)} := \int_\Omega u(x)v(x)\,\mathrm{d}x.$$

The Sobolev space $W^{m,p}(\Omega), m \in \mathbb{N}, 1 \leq p \leq \infty$, is the space of functions in $L^p(\Omega)$ with distributional derivative of order up to m in $L^p(\Omega)$. For $p = 2, H^m(\Omega) := W^{m,p}(\Omega)$ is a Hilbert space with the inner product

$$(u, v)_{H^m(\Omega)} := \sum_{|\alpha| \leq m} (\partial^\alpha u, \partial^\alpha v)_{L^2(\Omega)}.$$

If it is clear from the context we drop the subscripts and write

$$(u, v) := (u, v)_\Omega := (u, v)_{L^2(\Omega)}, \quad \|u\| := \|u\|_\Omega := \|u\|_{L^2(\Omega)}.$$

For a detailed introduction to Sobolev spaces we refer to [1].

2.1.2 Continuous function spaces

For $k \in \mathbb{N}$ we denote with $C^k(\Omega)$ the space of functions whose derivatives up to order k are continuous on Ω. We set

$$C^\infty(\Omega) = \bigcap_{k \in \mathbb{N}} C^k(\Omega).$$

For simplicity we set $C(\Omega) := C^0(\Omega)$. The space $C^k(\overline{\Omega})$ consists of all functions from $C^k(\Omega)$ whose derivatives up to order k possess continuous extensions onto $\overline{\Omega}$. The space $C_0^\infty(\Omega) \subseteq C^\infty(\Omega)$ is the subspace of functions which are non-zero only in a compact subset of Ω. The closure of $C_0^\infty(\Omega)$ in $W^{m,p}(\Omega)$ is denoted by $W_0^{m,p}(\Omega)$ or $H_0^m(\Omega)$ if $p = 2$. The dual space of any space X is indicated by an asterisk X^*, the dual space of $H_0^m(\Omega)$ is denoted by $H^{-m}(\Omega)$. The corresponding spaces of d-dimensional vector functions are denoted by $L^p(\Omega)^d, W^{m,p}(\Omega)^d$ and $H^m(\Omega)^d$. They are equipped with the usual product norm. The norms and inner products on these spaces are denoted in the same way as for scalar functions.

Let $I := (0, T)$ with $0 < T < \infty$ be a bounded time interval. For any Banach space X and $1 \leq p \leq \infty, L^p(I, X)$ denotes the space of L^p-integrable functions w from I into X. For a detailed derivation of these spaces by means of the Bochner integral, we refer to [72].

2.1.3 Sobolev spaces on a time dependent domain

In the case of a domain varying in time we carry over the definitions of the previous section. The space $C^k(\bar{I}, X), k \in \mathbb{N}$ consists of functions from $\bar{I}$ into X that are k times continuously differentiable on I and whose derivatives $\partial_t^i f(t)$ of order $0 \leq j \leq k$ possess continuous extensions onto $\bar{I}$. For convenience we set $C(\bar{I}, X) := C^0(\bar{I}, X)$.

3 Equations

In this chapter we present the governing equations. For the full model we use the Navier-Stokes equations in three dimensions. After making some assumptions we derive the reduced Shallow-Water model in two dimensions.

3.1 Basic equations describing fluid flows

For the simulation of hydrodynamics in lakes we use the Boussinesq equations derived from well known Navier-Stokes equations. These equations are valid on a three dimensional domain.

The equations are transformed to a fixed computational domain via an ALE transformation. We introduce a height describing the position of the free surface. The height is determined by the Shallow-Water equations.

3.1.1 Navier-Stokes equations

This section introduces the Navier-Stokes equations as well as equations describing the transport of a tracer. Throughout this thesis the temperature T is the only tracer that couples back to the Navier-Stokes equations. For the transport of pollutants we assume that the species that are transported do not couple back to the system. These kind of tracers are called passive tracers, whereas the temperature is an active tracer, because it influences the Navier-Stokes equations.

A derivation of the Navier-Stokes equations can be found in [25]. The equations are valid for a domain $\Omega(t),\ t \in \bar{I} = [0, T]$ with $0 < T < \infty$, varying in time.

The conservation of mass is described by

$$\partial_t \varrho + \nabla \cdot (\varrho v) = 0$$

and called the *continuity equation* where ϱ describes the density and v the velocity of the fluid. The balance of momentum leads to the *momentum equation*

$$\varrho \partial_t v + \varrho (v \cdot \nabla) v - \nabla \cdot \sigma = \varrho f + \varrho \boldsymbol{g} \tag{3.1}$$

with the stress tensor σ and external forces f. For the cases we consider in the thesis the stress tensor is given as

$$\sigma = -p \,\mathrm{Id} + \varrho \nu (\nabla v + \nabla v^T).$$

The pressure is denoted by p. The transport equation which holds for any tracer is

$$\partial_t T + v \cdot \nabla T - \nu_T \Delta T = q_T. \tag{3.2}$$

For the density there holds an equation of state:

$$\varrho = \varrho(T, p). \tag{3.3}$$

These equations are simplified after some assumptions are applied. The following reasoning is taken from [38]. The first step is to treat water as an incompressible fluid. This is based on the assumption that the water density is independent of pressure. The variations are only taken into account through the equation of state (3.3). Then the *continuity equation* becomes

$$\nabla \cdot v = 0, \tag{3.4}$$

which means that the velocity field is solenoidal. We state the weak formulation of the Navier-Stokes equations for constant density:

Problem 3.1. For $f \in L^2(I, V^*)$ and $v_0 \in H$ find $v \in L^2(I, V)$ with

$$\frac{\mathrm{d}}{\mathrm{d}t}(v, \varphi_v) + \nu(\nabla v, \nabla \varphi_v) + ((v \cdot \nabla)v, \varphi_v) = \langle f, \varphi_v \rangle \ \forall \varphi_v \in V,$$
$$v(0) = v_0.$$

The existence of a corresponding pressure is ensured by the following lemma:

Lemma 3.1 (**Inf-Sup Condition**). *For each $f \in H^{-1}(\Omega)^d$ with $\langle f, \varphi \rangle = 0 \ \forall \varphi \in V$ there is a unique $p \in L^2(\Omega)/\mathbb{R}$ such that*

$$f = \nabla p, \quad \textit{i.e.,} \ \langle f, \varphi \rangle = \langle \nabla p, \varphi \rangle = -(p, \nabla \cdot \varphi) \ \forall \varphi \in H_0^1(\Omega)^d.$$

Furthermore the inf-sup stability condition holds:

$$\|p\|_{L^2(\Omega)/\mathbb{R}} \leq \|\nabla p\|_{H^{-1}(\Omega)} = C \sup_{\varphi \in H_0^1(\Omega)^d} \frac{|(p, \nabla \cdot \varphi)|}{\|\nabla \varphi\|}.$$

Proof. A proof of this fundamental property can be found, for instance, in [33] or [64]. □

Under additional assumptions on the data one can show that the weak solution of Problem 3.1 possesses more regularity and that the pressure can be viewed as an almost everywhere defined function rather than only a distribution. The results for the three dimensional case are presented in the following proposition:

Proposition 3.2. *Let Ω be a bounded domain in $\mathbb{R}^3$ with boundary of class C^2 and $v_0 \in V$. For $f \in L^\infty(I, H)$ there exists T', $0 < T' \leq T$ such that Problem 3.1 possesses a unique solution v on $I' := (0, T')$. Moreover,*

$$v \in L^2(I', H^2(\Omega)^3 \cap H_0^1(\Omega)^3) \cap L^\infty(I', V) \ \textit{and} \ \partial_t v \in L^2(I', H).$$

For the corresponding pressure we obtain $p \in L^2(I', H^1(\Omega)^3)$.

Proof. We refer to [64] for a proof of this regularity and uniqueness result. □

Remark 3.1. The requirements concerning the regularity of the boundary $\partial\Omega$ can be weakened. The statement of Proposition 3.2 remains true if Ω is a polygonally bounded and convex domain or its boundary is of class $C^{1,1}$.

For simplicity we will write $I' = I$ in the following.

3.1.2 Boussinesq approximation

As a next step we apply the Boussinesq approximation to the Navier-Stokes equations. In natural water bodies as rivers, lakes and seas, the variations of density are relatively small. Based on this fact the variations of density are taken into account only in those terms of the *momentum equation* (3.1).

The density is assumed to be a sum of a constant average density ϱ_0 and a variable density variance $\Delta\varrho$ in the form

$$\varrho = \varrho_0 + \Delta\varrho.$$

Then it holds

$$\frac{1}{\varrho} = \frac{1}{\varrho_0 + \Delta\varrho} = \frac{1}{\varrho_0(1 + \frac{\Delta\varrho}{\varrho_0})}. \tag{3.5}$$

Taking into account that $\frac{\Delta\varrho}{\varrho_0} \ll 1$ we can linearize (3.5) to

$$\frac{1}{\varrho} \approx \frac{1}{\varrho_0}\left(1 - \frac{\Delta\varrho}{\varrho_0}\right). \tag{3.6}$$

The hydrostatic pressure p_H results from the average density ϱ_0 as

$$p_H = \varrho_0 g(h - x_3),$$

where $h(x_1, x_2, t)$ is the position at the free surface. Thus we can write the pressure as a sum of the hydrostatic pressure and the pressure variation Δp:

$$p = p_H + \Delta p. \tag{3.7}$$

Using what we got for the density ϱ in (3.6) and the pressure in (3.7) we derive

$$\begin{aligned}
\frac{1}{\varrho}\nabla p &= \frac{1}{\varrho_0}\left(1 - \frac{\Delta\varrho}{\varrho_0}\right)\nabla p = \frac{1}{\varrho_0}\nabla p - \frac{\Delta\varrho}{\varrho_0^2}\nabla p \\
&= \frac{1}{\varrho_0}\nabla p - \frac{\Delta\varrho}{\varrho_0^2}\nabla p \\
&= \frac{1}{\varrho_0}\nabla p - \frac{\Delta\varrho}{\varrho_0^2}\nabla p_H - \frac{\Delta\varrho}{\varrho_0^2}\nabla(\Delta p) \\
&= \frac{1}{\varrho_0}\nabla p - \frac{\Delta\varrho}{\varrho_0}\boldsymbol{g} - \frac{\Delta\varrho}{\varrho_0}g\nabla h - \frac{\Delta\varrho}{\varrho_0^2}\nabla(\Delta p)
\end{aligned} \tag{3.8}$$

where $\boldsymbol{g} = (0, 0, -g)$ denotes the gravity. The idea of the Boussinesq approximation is to neglect the last two terms in (3.8) which leads to

$$\frac{1}{\varrho}\nabla p = \frac{1}{\varrho_0}\nabla p - \frac{\Delta\varrho}{\varrho_0}\boldsymbol{g}.$$

The equation for the conservation of momentum (3.1) transforms after applying the Boussinesq approximation into

$$\partial_t v + (v\cdot\nabla)v + \frac{1}{\varrho_0}\nabla p - \nu\Delta v - \frac{\varrho}{\varrho_0}\boldsymbol{g} = f. \tag{3.9}$$

This means we end up with the following set of equations to be solved

$$\begin{aligned} \partial_t v + (v\cdot\nabla)v + \frac{1}{\varrho_0}\nabla p - \nu\Delta v - \frac{\varrho}{\varrho_0}\boldsymbol{g} &= f, \\ \nabla\cdot v &= 0, \\ \partial_t T + v\cdot\nabla T - \nu_T\Delta T &= q_T, \\ \varrho &= \varrho(T), \end{aligned} \tag{3.10}$$

subject to boundary and initial conditions for the velocity

$$v(0) = v_0, \qquad v = 0 \text{ on } \partial\Omega \times [0,T],$$

as well as for the temperature

$$T(0) = T_0, \qquad n\nabla T = 0 \text{ on } \Gamma_v \times [0,T], \qquad T = g_T \text{ on } \Gamma_s \times [0,T].$$

In the case we impose boundary conditions for the velocity on the whole boundary of the domain, the pressure is determined up to a constant. To fix the pressure uniquely we require

$$\int_{\partial\Omega} p \,\mathrm{d}s = 0.$$

From this point we are able to set up a variational formulation including boundary and initial conditions.

3.1.3 Variational formulation of the Boussinesq equations

We take the density to be a function of the temperature $\varrho(T)$ and set $\gamma := \frac{1}{\varrho_0}$. The solution $u := (v, p, T) \in X(t)$ of (3.10) satisfies $v(0) = v_0$ and $T(0) = T_0$ and for almost all $t \in I$ it fulfills

$$\begin{aligned} (\partial_t v, \varphi_v) + ((v\cdot\nabla)v, \varphi_v) + \gamma(\nabla p, \varphi_v) + (\nu\nabla v, \nabla\varphi_v) - \gamma(\varrho(T)\boldsymbol{g}, \varphi_v) + (\nabla\cdot v, \varphi_p) \\ +(\partial_t T, \varphi_T) + (v\cdot\nabla T, \varphi_T) + \nu_T(\nabla T, \nabla\varphi_T) \\ = (f, \varphi_v) + (q_T, \varphi_T) \;\forall \varphi = (\varphi_v, \varphi_p, \varphi_T) \in \widetilde{X}(t), \end{aligned} \tag{3.11}$$

where

$$X(t) := \Big\{ u = (v, p, T) \;\Big|\; v(t) \in H_0^1(\Omega(t))^3, \partial_t v(t) \in L^2(\Omega(t))^3, \\ p(t) \in L^2(\Omega(t))/\mathbb{R}, T(t) \in H_0^1(\Omega(t)), \partial_t T(t) \in L^2(\Omega(t)) \text{ for almost all } t \in I \Big\}, \tag{3.12}$$

and

$$\widetilde{X}(t) := \Big\{ u = (v, p, T) \;\Big|\; v \in H_0^1(\Omega(t))^3, p \in L^2(\Omega(t))/\mathbb{R}, T \in H_0^1(\Omega(t)) \Big\}. \tag{3.13}$$

If we are interested in the transport of tracers which do not couple back to the system (3.11) we have to solve additional equations of type

$$\partial_t S + v \cdot \nabla S - \nu_S \Delta S = q_S \tag{3.14}$$

for each tracer. We call the tracers that do not couple back to the system a passive tracer. The temperature denoted by T e.g. is an active tracer. The variational formulation for the tracer equation (3.14) reads: Find $S(t) \in Y(t)$ such that $S(0) = S_0$ and such that for almost all $t \in I$ it holds

$$(\partial_t S, \varphi_S) + (v \cdot \nabla S, \varphi_S) + \nu_S (\nabla S, \nabla \varphi_S) = (q_S, \varphi_S) \; \forall \varphi_S \in \widetilde{Y}(t),$$

where

$$Y(t) := \Big\{ S \;\Big|\; S(t) \in H_0^1(\Omega(t)), \partial_t S(t) \in L^2(\Omega(t)) \text{ for almost all } t \in I \Big\},$$

and

$$\widetilde{Y}(t) := \Big\{ S \;\Big|\; S \in H_0^1(\Omega(t)) \Big\}.$$

3.1.4 Shallow-Water equations

In this section we present the Shallow-Water equations. By assuming hydrostatic pressure and small velocity in the vertical direction in the Navier-Stokes equations

$$\begin{aligned} \partial_t v + (v \cdot \nabla) v + \gamma \nabla p - \nu \Delta v - \varrho(T) \gamma \boldsymbol{g} &= f \\ \nabla \cdot v &= 0 \end{aligned} \tag{3.15}$$

the Shallow-Water equations (3.16) can be derived by integration from bottom to the free surface. A detailed derivation can be found in [51] or [71]. The Navier-Stokes equations valid on the domain $\Omega(t)$ transform into the Shallow-Water equations which are stated on a two dimensional domain Ω_{2D} which is a projection of the three dimensional domain $\Omega(t)$ onto the horizontal plane. Since only the free surface in 3D is evolving with time the two dimensional domain is not time dependent.

In non-conservative form the Shallow-Water equations read

$$\begin{aligned} h \partial_t v_{2D} + h (v_{2D} \cdot \nabla) v_{2D} - \nu \nabla \cdot (h \nabla v_{2D}) + g h \nabla h &= F, \\ \partial_t h + v_{2D} \nabla h + h \nabla \cdot v_{2D} &= 0. \end{aligned} \tag{3.16}$$

Here, $h = h(x_1, x_2, t)$ denotes the height of the water body. We assume that h is defined pointwise for all $x_1, x_2 \in \Omega_{2D}$ and for all $t \in I$. The two dimensional velocity $v_{2D} = v_{2D}(x_1, x_2, t)$ is the averaged velocity of the Navier-Stokes equations (3.15):

$$v_{2D}^i = \frac{1}{h} \int_0^h v^i \, \mathrm{d}z, \quad i = 1, 2, \tag{3.17}$$

and the pressure can be expressed using the height h as

$$p = \varrho g(h - x_3) + p_A,$$

where p_A denotes the atmospheric pressure which is constant. For later purposes we also state the Shallow-Water equations in conservative form:

$$\begin{aligned} \partial_t(hv_{2D}) + (v_{2D}\cdot\nabla)(hv_{2D}) + h(v_{2D}\cdot\nabla)v_{2D} - \nu\nabla\cdot(h\nabla v_{2D}) + \frac{1}{2}g\nabla h^2 = F, \\ \partial_t h + \nabla\cdot(hv_{2D}) = 0. \end{aligned} \tag{3.18}$$

3.1.5 Variational formulation of the Shallow-Water equations

The variational problem for the Shallow-Water equations reads: Find $u_{2D} = (v_{2D}, h) \in X_{2D}$ such that $v(0) = v_0$ and $h(0) = h_0$ and such that for almost all $t \in I$ it holds

$$\begin{aligned} (h\partial_t v_{2D}, \varphi_{v_{2D}}) + (h(v_{2D}\cdot\nabla)v_{2D}, \varphi_{v_{2D}}) - \nu(h\nabla v_{2D}, \nabla\varphi_{v_{2D}}) + (gh\nabla h, \varphi_{v_{2D}}) \\ +(\partial_t h, \varphi_h) + (v_{2D}\nabla h, \varphi_h) + (h\nabla\cdot v_{2D}, \varphi_h) \\ = (F, \varphi_{v_{2D}}), \quad \forall\varphi = (\varphi_{v_{2D}}, \varphi_h) \in \widetilde{X}_{2D}, \end{aligned} \tag{3.19}$$

where

$$\begin{aligned} X_{2D} := \Big\{ u = (v_{2D}, h) \;\Big|\; v_{2D} \in L^2(I, H^2(\Omega_{2D}) \cap H_0^1(\Omega_{2D})^2), \partial_t v_{2D} \in L^2(I, L^2(\Omega_{2D})^2), \\ h \in L^2(I, H^2(\Omega_{2D}) \cap H_0^1(\Omega_{2D})), \partial_t h \in L^2(I, L^2(\Omega_{2D})) \Big\}, \end{aligned}$$

and

$$\widetilde{X}_{2D} := \Big\{ u = (v_{2D}, h) \;\Big|\; v_{2D} \in H_0^1(\Omega_{2D})^2, h \in H_0^1(\Omega_{2D}) \Big\}.$$

3.2 Equations for free surface flows

With regard to the discretization of the problems stated in Section 3.1, we make use of the concept of the ALE formulation. This approach is often used in fluid-structure interaction problems, see [31], [50], [56] and [37].

3.2.1 Arbitrary Lagrangian Eulerian (ALE) formulation

Let $\mathcal{A}^t$ be a family of mappings which at each time $t \in [0,T]$ associate a point x of a reference configuration Ω to a point x^t on the current domain configuration $\Omega(t)$:

$$\mathcal{A}^t : \Omega \to \Omega(t), \quad x^t(x,t) = \mathcal{A}^t(x).$$

We assume $\mathcal{A}^t$ to be an homeomorphism, that is $\mathcal{A}^t$ is invertible with continuous inverse $(\mathcal{A}^t)^{-1} =: \mathcal{A}$ and $\mathcal{A}^t$ is also two times continuously partially differentiable.

Let $f^t : \Omega(t) \times I \to \mathbb{R}$ be a function defined on the Eulerian frame and $f := f^t \circ \mathcal{A}^t$ the corresponding function on the ALE frame, defined as

$$f : \Omega \times I \to \mathbb{R}, \quad f(x,t) = f^t(\mathcal{A}^t(x),t)$$

With the help of the mapping $\mathcal{A}^t$ functions and operators in $\Omega(t)$ can be rewritten as such in the domain Ω. For this reason we introduce the transformation identities. By F and J we denote the Jacobian matrix of the mapping $\mathcal{A}^t$ and its determinant respectively:

$$F := \nabla \mathcal{A}^t, \quad J := \det F. \tag{3.20}$$

3.2.2 Spacial and temporal derivatives in ALE formulation

Spacial derivatives of f can be obtained by the chain rule:

$$\partial_i f(x,t) = \sum_j \partial_j^t f^t(\mathcal{A}^t(x),t)\partial_i \mathcal{A}_j^t(x,t),$$

where ∂_i^t is a partial derivative on the domain $\Omega(t)$. Thus the gradient of a scalar function can be written as

$$\nabla f = F \nabla^t f,$$

with ∇^t being the gradient on $\Omega(t)$. Applying this identity to each component of a vector field we obtain

$$\nabla v = \nabla^t v F.$$

We present also the transformed temporal derivatives. For a proof of the identities we refer to [31]:

$$\begin{aligned} \partial_t^t v &= \partial_t v_t - (F^{-1}\partial_t \mathcal{A}^t \cdot \nabla)v \\ d_t^t v &= \partial_t v_t + (F^{-1}(v - \partial_t \mathcal{A}^t)\cdot \nabla)v. \end{aligned}$$

This means we have to following identities for the transformation of spacial and temporal derivatives. Here f denotes a scalar function whereas v denotes a vector field:

$$\begin{aligned} \nabla^t f^t &= F^{-T}\nabla f, & \nabla^t v^t &= \nabla v F^{-1}, \\ \partial_t^t f^t &= \partial_t f - (F^{-1}\partial_t \mathcal{A}^t \cdot \nabla)f, & \partial_t^t v^t &= \partial_t v - (F^{-1}\partial_t \mathcal{A}^t \cdot \nabla)v, \\ d_t^t f^t &= \partial_t f + (F^{-1}(v - \partial_t \mathcal{A}^t)\cdot \nabla)f, & d_t^t v^t &= \partial_t v + (F^{-1}(v - \partial_t \mathcal{A}^t)\cdot \nabla)v. \end{aligned} \tag{3.21}$$

With $f(x) = f^t(\mathcal{A}^t(x)) = f^t(x^t) \in L^2(\Omega)$ we transform the volume integral in $\Omega(t)$ to an integral in Ω as

$$\int_{\Omega(t)} f^t(x^t)\,\mathrm{d}x^t = \int_{\Omega} f(x)J\,\mathrm{d}x.$$

Prior to define the ALE mapping we mention the regularity conditions we address to $\mathcal{A}^t$.

Proposition 3.3. *Let Ω be a bounded domain with Lipschitzian boundary and let $\mathcal{A}_t$ be invertible in $\overline{\Omega}$. For each $t \in I$ let the following conditions be satisfied*

(i) $\Omega_t = \mathcal{A}^t(\Omega)$ is bounded and $\partial\Omega_t$ is Lipschitzian.

(ii) $\mathcal{A}^t \in W^{1,\infty}(\Omega)$ and $(\mathcal{A}^t)^{-1} \in W^{1,\infty}(\Omega_t)$.

Then, $v^t \in H^1(\Omega_t)$ if and only if $v = v^t \circ \mathcal{A}^t \in H^1(\Omega)$. Moreover, $\left\|v^t\right\|_{H^1(\Omega_t)}$ is equivalent to $\|v\|_{H^1(\Omega)}\ \forall v \in H^1(\Omega)$.

Proof. The proof can be found in [50]. □

Concerning the time regularity of the mapping we assume that the function $x^t(x,t)$ satisfies

$$x^t \in H^1(I, W^{1,\infty}(\Omega)). \tag{3.22}$$

Proposition 3.4. *Under the assumption* (3.22)*, we have that if $v \in H^1(I, H^1(\Omega))$ then, $v^t = v \circ (\mathcal{A}^t)^{-1} \in H^1(I, H^1(\Omega_t))$ and*

$$\partial_t v \in L^2(I, H^1(\Omega_t)).$$

Proof. The proof can be found in [50]. □

3.2.3 Construction of the ALE mapping

The widely used technique, especially in the atmospheric and oceanographic communities, referred to as the sigma transformation [53], is used to construct the ALE mapping. The sigma transformation performs a transformation in the vertical coordinate to follow the free surface. This is shown in Figure 3.1. We define the ALE mapping as

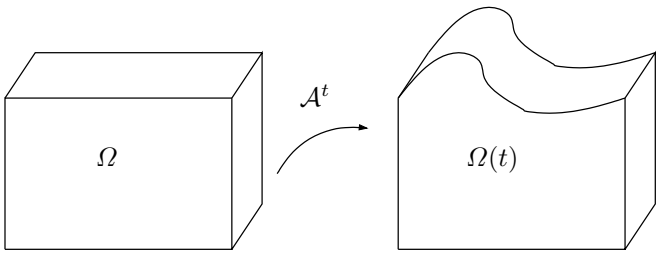


Figure 3.1. Transformation to a fixed domain

$$\mathcal{A}^t(x,t) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & h \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}. \tag{3.23}$$

The function $h(x_1, x_2, t)$ denotes the height from the bottom to the free surface of the domain $\Omega(t)$. It remains to rewrite the equations of the previous section on the fixed reference domain Ω taking the relations (3.21) into account. We use the variational formulation for discretization and therefore we present the rewritten equations in weak formulation.

Using the definitions (3.20) and the relation $J = \det F = h$, we get for the continuity equation (3.4)

$$(\nabla^t \cdot v^t, \varphi_p^t)_{\Omega(t)} = (h\partial_1 v_1 + h\partial_2 v_2 - x_3\partial_1 h\partial_3 v_1 - x_3\partial_2 h\partial_3 v_2 + \partial_3 v_3, \varphi_p)_\Omega. \tag{3.24}$$

We have for the three terms in the momentum equation (3.9)

$$\begin{aligned} (\partial_t^t v^t + v^t \cdot \nabla^t v^t, \varphi_v^t)_{\Omega(t)} &= (h\partial_t v + h(F^{-1}(v - \partial_t \mathcal{A}^t)\cdot\nabla)v), \varphi_v)_\Omega \\ (\sigma^t, \nabla^t \varphi_v^t)_{\Omega(t)} &= (h\sigma F^{-T}, \nabla\varphi_v)_\Omega \\ -\gamma(\varrho(T)^t \boldsymbol{g}, \varphi_v^t)_{\Omega(t)} &= -\gamma(h\varrho(T)\boldsymbol{g}, \varphi_v)_\Omega. \end{aligned} \tag{3.25}$$

The terms in the temperature equation (3.2) transform into

$$\begin{aligned} (\partial_t^t T^t + v^t \cdot \nabla^t T^t, \varphi_T^t)_{\Omega(t)} &= (h\partial_t T + h(F^{-1}(v - \partial_t \mathcal{A}^t)\cdot\nabla)T, \varphi_T)_\Omega \\ \nu_T(\nabla^t T^t, \nabla^t \varphi_T^t)_{\Omega(t)} &= (\nu_T h F^{-1} F^{-T} \nabla T, \nabla\varphi_T)_\Omega. \end{aligned} \tag{3.26}$$

Before we can write the coupled system in ALE formulation we have to require an additional equation for the height h that was introduced with the definition of the ALE mapping in (3.23). One choice is to take the so called *kinematic boundary condition* at the free surface:

$$\partial_t h + v_1^S \partial_1 h + v_2^S \partial_2 h - v_3^S = 0. \tag{3.27}$$

The superscript S indicates that the velocity components are taken at the free surface. Instead one could use the *free surface equation* in conservative form, which is given as

$$\partial_t h + \partial_1(h v_{2D}^1) + \partial_2(h v_{2D}^2) = 0, \tag{3.28}$$

where v_{2D}^i is defined in (3.17). For more information about these equations and their derivation, we refer to [38].

For sake of completeness we present the variational formulation of (3.27) and (3.28). For a given velocity at the surface we require $h \in Y_{2D}$ to satisfy $h(0) = h_0$ and for almost all $t \in I$ it holds

$$(\partial_t h + v_1^S \partial_1 h + v_2^S \partial_2 h - v_3^S, \varphi_h)_{\Omega_{2D}} = 0, \quad \forall \varphi_h \in \widetilde{Y}_{2D}, \tag{3.29}$$

where

$$Y_{2D} := \Big\{ h \;\Big|\; h \in L^2(I, H^2(\Omega_{2D}) \cap H_0^1(\Omega_{2D})), \partial_t h \in L^2(I, L^2(\Omega_{2D})) \Big\},$$

and

$$\widetilde{Y}_{2D} := \Big\{ h \;\Big|\; h \in H_0^1(\Omega_{2D}) \Big\}.$$

With Ω_{2D} we denote the vertical projection of Ω onto the plane $x_3 = 0$. Instead one could require $h \in Y_{2D}$ to satisfy $h(0) = h_0$ and for almost all $t \in I$

$$(\partial_t h + \partial_1(hv_{2D}^1) + \partial_2(hv_{2D}^2), \varphi_h)_{\Omega_{2D}} = 0, \quad \forall \varphi_h \in \widetilde{Y}_{2D}. \tag{3.30}$$

We use one of these equations posed on the free surface to complete the set of equations for the coupled ALE formulation.

3.2.4 Equations formulated in ALE framework

In the following we omit the subscript Ω since we only consider integrals on the fixed domain. We make use of a subscript in those cases where the domain of integration differs from Ω. Taking into account the equations (3.24), (3.25), (3.26) and (3.29), the full coupled system in ALE formulation (3.11) reads: Find $u := (v, p, T, h) \in X$ such that $v(0) = v_0$, $T(0) = T_0$ and $h(0) = h_0$ and such that for almost all $t \in I$ it holds

$$\begin{aligned}
(h\partial_t v + h(F^{-1}(v - \partial_t \mathcal{A}^t)\cdot\nabla)v, \varphi_v) - \gamma(hpIF^{-T}, \nabla\varphi_v) + (\nu h \nabla v F^{-1}F^{-T}, \nabla\varphi_v) \\
-\gamma(\varrho(T)h\boldsymbol{g}, \varphi_v) + (h\partial_1 v_1 + h\partial_2 v_2 - x_3\partial_1 h\partial_3 v_1 - x_3\partial_2 h\partial_3 v_2 + \partial_3 v_3, \varphi_p) \\
+(h\partial_t T, \varphi_T) + (h(F^{-1}(v - \partial_t\mathcal{A}^t)\cdot\nabla)T, \varphi_T) + (\nu_T h F^{-1}F^{-T}\nabla T, \nabla\varphi_T) \\
+(\partial_t h + v_1^S\partial_1 h + v_2^S\partial_2 h - v_3^S, \varphi_h)_{\Omega_{2D}} \\
= (f, \varphi_v) + (q_T, \varphi_T)
\end{aligned} \tag{3.31}$$

for all $\varphi = (\varphi_v, \varphi_p, \varphi_T, \varphi_h) \in \widetilde{X}$, where

$$\begin{aligned}
X := \Big\{ u = (v, p, T) \;\Big|\; & v \in L^2(I, H^2(\Omega)^3 \cap H_0^1(\Omega)^3), \partial_t v \in L^2(I, L^2(\Omega)^3), \\
& p \in L^2(I, L^2(\Omega)/\mathbb{R}), T \in L^2(I, H^2(\Omega) \cap H_0^1(\Omega)), \partial_t T \in L^2(I, L^2(\Omega)), \\
& h \in L^2(I, H^2(\Omega_{2D}) \cap H_0^1(\Omega_{2D})), \partial_t h \in L^2(I, L^2(\Omega_{2D}))\Big\},
\end{aligned}$$

and

$$\widetilde{X} := \Big\{ u = (v, p, T) \;\Big|\; v \in H_0^1(\Omega)^3, p \in L^2(\Omega)/\mathbb{R}, T \in H_0^1(\Omega), h \in H_0^1(\Omega_{2D}) \Big\}.$$

Note that in contrast to the definition of the spaces $X(t)$ and $\widetilde{X}(t)$ in (3.12) and (3.13), respectively, the definition of X and $\widetilde{X}$ is based on a fixed domain Ω whereas the spaces $X(t)$ and $\widetilde{X}(t)$ rest on a moving domain $\Omega(t)$.

Results concerning existence and uniqueness of solutions of free surface flows are discussed in [23] for one and two dimensional domains.

4 Coupling of models in 2D and 3D

The chapter is devoted to the development of coupling models in different dimensions. First we describe how we achieve this coupling for a model problem (Laplace problem). Later on we adapt these ideas to coupling different models for fluid flow.

A way to reduce the model but keep the accuracy at the same time is to couple models in different dimensions. In a first step we introduce the idea of this coupling for a Poisson model problem. The same techniques are then applied in a second step to the models we consider for describing fluid flows, namely the three dimensional Navier-Stokes equations and the reduced version which are the Shallow-Water equations.

4.1 Coupling of 2D and 3D Poisson problem

Let us consider the Poisson problem on a domain $\Omega = (-1,1)^3$ in $\mathbb{R}^3$. With Γ_{top} and Γ_{bottom} we denote the top and bottom boundary of Ω, respectively. The remaining part of the boundary is denoted by Γ. The weak formulation reads: Find $u \in W = \{u \in H^1(\Omega),\ u = 0 \text{ on } \Gamma\}$ such that

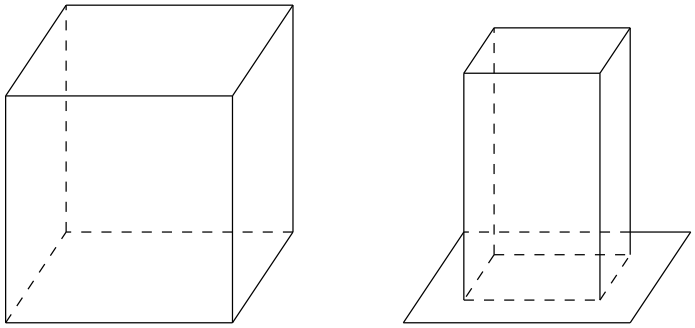
$$(\nabla u, \nabla \varphi) + \langle \alpha, \varphi \rangle_{\Gamma_{\text{top}} \cup \Gamma_{\text{bottom}}} = (f, \varphi) \text{ for all } \varphi \in W, \tag{4.1}$$

with α defined as

$$\alpha(x_1, x_2) = \begin{cases} 10, & \text{if } -\frac{1}{4} \leq x, y \leq \frac{1}{4}, \\ 0, & \text{else.} \end{cases}$$

Since this is very costly to solve in three dimensions we reduce the model to two dimensional model. Only in areas where the accuracy of the two dimensional model is not satisfactory we apply the three dimensional model. As an example let us consider the cube in three dimensions. The original problem (4.1) we want to solve is reformulated on the reduced domain shown in Figure 4.1 on the right.

The idea is to solve the problem on the two dimensional domain extended by a cube in an area defined a priori. The solution is then understood as a solution on the two dimensional domain and the solution on the part in three dimensions is treated as a correction to the solution on the square. Thus we can write the solution $u \in W$ as $u = u_{2D} + u_{3D}$ where u_{2D} is defined on the square and u_{3D} lives on the cube in the middle. Since we understand the part u_{3D} as a correction of the solution on the square domain we impose zero boundary conditions for u_{3D} on the vertical faces of the cube.



(a) Cube in 3D

(b) Cube in 2D with a 3d cube

Figure 4.1. Reduction from a 3D cube to a 2D cube with a 3D cube in the middle.

4.1.1 Decomposition of the space W

This means that we decompose the space of test and trial functions W into a sum of two spaces with such that

$$W = W_{2D} + W_{3D}$$

holds and the weak formulation (4.1) then reads: Find $u = u_{2D} + u_{3D} \in W = W_{2D} + W_{3D}$ such that (4.1) holds. This means we understand a function $v \in W_{2D}$ as a function in W_{3D}:

$$v = v(x_1, x_2) \in W_{2D} \mathrel{\widehat{=}} v(x_1, x_2, x_3) \in W_{3D}. \tag{4.2}$$

If it is clear from the context we write differential operators in three dimensions for functions in W_{2D} by taking the corresponding function in W_{3D}. As an example the gradient is given as

$$\nabla v = \begin{pmatrix} \partial_1 v \\ \partial_2 v \\ 0 \end{pmatrix}.$$

We define a bilinear form $a : W \times W \to \mathbb{R}$ as

$$a(u, \varphi) = (\nabla u, \nabla \varphi) + \langle \alpha, \varphi \rangle_{\Gamma_{\text{top}} \cup \Gamma_{\text{bottom}}} \tag{4.3}$$

and examine it on the decomposed spaces $a : W_{2D} + W_{3D} \times W_{2D} + W_{3D} \to \mathbb{R}$. Inserting the sum for test and trial functions leads to

$$a(u, \varphi) = a(u_{2D} + u_{3D}, \varphi_{2D} + \varphi_{3D}). \tag{4.4}$$

The original problem (4.1) using the bilinear form on the decomposed spaces then reads: Find $u = u_{2D} + u_{3D} \in W = W_{2D} + W_{3D}$ such that

$$a(u_{2D} + u_{3D}, \varphi_{2D} + \varphi_{3D}) = (f, \varphi_{2D} + \varphi_{3D}) \text{ for all } \varphi_{2D} \in W_{2D} \text{ and } \varphi_{3D} \in W_{3D}.$$

We can vary the test functions φ_{2D} and φ_{3D} independently and we arrive at two equations:

$$a(u_{2D} + u_{3D}, \varphi_{3D}) = (f, \varphi_{3D}) \text{ for all } \varphi_{3D} \in W_{3D},$$
$$a(u_{2D} + u_{3D}, \varphi_{2D}) = (f, \varphi_{2D}) \text{ for all } \varphi_{2D} \in W_{2D},$$

or equivalently

$$(\nabla u_{2D} + \nabla u_{3D}, \nabla \varphi_{3D}) + \langle \alpha, \varphi_{3D} \rangle_{\Gamma_{\text{top}} \cup \Gamma_{\text{bottom}}} = (f, \varphi_{3D}) \text{ for all } \varphi_{3D} \in W_{3D},$$
$$(\nabla u_{2D} + \nabla u_{3D}, \nabla \varphi_{2D}) + \langle \alpha, \varphi_{2D} \rangle_{\Gamma_{\text{top}} \cup \Gamma_{\text{bottom}}} = (f, \varphi_{2D}) \text{ for all } \varphi_{2D} \in W_{2D}.$$

This means we have couplings between functions in W_{2D} and W_{3D} which appear in the bilinear form, namely

$$(\nabla u_{2D}, \nabla \varphi_{3D}) \text{ and } (\nabla u_{3D}, \nabla \varphi_{2D}).$$

Furthermore we have to give meaning to the term $\langle \alpha, \varphi_{2D} \rangle_{\Gamma_{\text{top}} \cup \Gamma_{\text{bottom}}}$.

4.1.2 Discussion of the coupling terms

$(\nabla u_{2D}, \nabla \varphi_{3D})_\Omega$:

This term arises in the first equation and is integrated over the part of the domain which is three dimensional. We have to provide the function u_{2D} in the three dimensional domain. The function is extended into the third dimension constantly as explained in (4.2). For the coupling we derive

$$\begin{aligned}(\nabla u_{2D}, \nabla \varphi_{3D})_\Omega &= \int_\Omega \nabla u_{2D} \cdot \nabla \varphi_{3D} \, \mathrm{d}x \\ &= \int_\Omega \partial_1 u_{2D} \partial_1 \varphi_{3D} + \partial_2 u_{2D} \partial_2 \varphi_{3D} \, \mathrm{d}x.\end{aligned}$$

This is an integral on the three dimensional domain Ω. The two dimensional function needs to be transferred to the three dimensional space to perform this integration. In Section 6.3 we comment on the technical details how this is achieved in the implementation.

$(\nabla u_{3D}, \nabla \varphi_{2D})_\Omega$:

To simplify the coupling in the second equation we express the integral in three dimension as an integral in two dimensions:

$$\begin{aligned}(\nabla u_{3D}, \nabla \varphi_{2D})_\Omega &= \int_\Omega \nabla u_{3D} \cdot \nabla \varphi_{2D} \, \mathrm{d}x \\ &= \int_\Omega \partial_1 u_{3D} \partial_1 \varphi_{2D} + \partial_2 u_{3D} \partial_2 \varphi_{2D} \, \mathrm{d}x \\ &= \int_{\Omega_{2D}} \int_0^1 \partial_1 u_{3D} \partial_1 \varphi_{2D} \, \mathrm{d}x_3 \, \mathrm{d}x_1 x_2 + \int_{\Omega_{2D}} \int_0^1 \partial_2 u_{3D} \partial_2 \varphi_{2D} \, \mathrm{d}x_3 \, \mathrm{d}x_1 x_2.\end{aligned} \tag{4.5}$$

Since φ_{2D} is only a function of x_1 and x_2 we can write the integrals as

$$\begin{aligned}\int_{\Omega_{2D}} \int_0^1 \partial_1 u_{3D} \partial_1 \varphi_{2D} \, \mathrm{d}x_3 \, \mathrm{d}x_1 x_2 &= \int_{\Omega_{2D}} \partial_1 \varphi_{2D} \int_0^1 \partial_1 u_{3D} \, \mathrm{d}x_3 \, \mathrm{d}x_1 x_2, \\ \int_{\Omega_{2D}} \int_0^1 \partial_2 u_{3D} \partial_2 \varphi_{2D} \, \mathrm{d}x_3 \, \mathrm{d}x_1 x_2 &= \int_{\Omega_{2D}} \partial_2 \varphi_{2D} \int_0^1 \partial_2 u_{3D} \, \mathrm{d}x_3 \, \mathrm{d}x_1 x_2.\end{aligned} \tag{4.6}$$

We are allowed to switch integration and differentiation and we get

$$\begin{aligned}\int_{\Omega_{2D}} \partial_1\varphi_{2D} \int_0^1 \partial_1 u_{3D}\,\mathrm{d}x_3\,\mathrm{d}x_1x_2 &= \int_{\Omega_{2D}} \partial_1\varphi_{2D}\partial_1 \int_0^1 u_{3D}\,\mathrm{d}x_3\,\mathrm{d}x_1x_2,\\ \int_{\Omega_{2D}} \partial_2\varphi_{2D} \int_0^1 \partial_2 u_{3D}\,\mathrm{d}x_3\,\mathrm{d}x_1x_2 &= \int_{\Omega_{2D}} \partial_2\varphi_{2D}\partial_2 \int_0^1 u_{3D}\,\mathrm{d}x_3\,\mathrm{d}x_1x_2.\end{aligned} \tag{4.7}$$

With the definition

$$u^i_{2D} := \int_0^1 u^i_{3D}\,\mathrm{d}x_3, \quad i = 1,2$$

and (4.5), (4.6), and (4.7) we can write the integral, we started from, as an integral in two dimensions:

$$\begin{aligned}(\nabla u_{3D}, \nabla\varphi_{2D})_\Omega &= \int_{\Omega_{2D}} \partial_1\varphi_{2D}\partial_1 u_{2D}\,\mathrm{d}x_1x_2 + \int_{\Omega_{2D}} \partial_2\varphi_{2D}\partial_2 u_{2D}\,\mathrm{d}x_1x_2\\ &= (\nabla u_{2D}, \nabla\varphi_{2D})_{\Omega_{2D}}.\end{aligned}$$

In the same way we interpret the remaining terms $(\nabla u_{2D}, \nabla\varphi_{2D})_{3D}$ and $(f, \varphi_{2D})_{3D}$ as integrals in two dimensions.

4.1.3 Discussion of the boundary term

$\langle\alpha, \varphi_{2D}\rangle_{\Gamma_{\text{top}}\cup\Gamma_{\text{bottom}}}$:

This term translates into a reaction term in two dimensions:

$$\begin{aligned}\langle\alpha, \varphi_{2D}\rangle_{\Gamma_{\text{top}}\cup\Gamma_{\text{bottom}}} &= \langle\alpha, \varphi_{2D}\rangle_{\Gamma_{\text{top}}} + \langle\alpha, \varphi_{2D}\rangle_{\Gamma_{\text{bottom}}},\\ &= (\alpha, \varphi_{2D})_{\Omega_{2D}} + (\alpha, \varphi_{2D})_{\Omega_{2D}}.\end{aligned}$$

This means we end up with solving the following problem: Find $(u_{3D}, u_{2D}) \in W_{3D} \times W_{2D}$ such that

$$(\nabla u_{3D}, \nabla\varphi_{3D})_\Omega + (\nabla u_{2D}, \nabla\varphi_{3D})_\Omega + \langle\alpha, \varphi_{3D}\rangle_{\Gamma_{\text{top}}\cup\Gamma_{\text{bottom}}} = (f, \varphi_{3D})_\Omega, \tag{4.8}$$

$$2(\nabla u_{2D}, \nabla\varphi_{2D})_{\Omega_{2D}} + 2(\alpha, \varphi_{2D})_{\Omega_{2D}} = (f, \varphi_{2D})_{\Omega_{2D}}, \tag{4.9}$$

for all $(\varphi_{3D}, \varphi_{2D}) \in W_{3D} \times W_{2D}$.

Remark 4.1. A boundary term on the top or the bottom of the three dimensional domain results in a reaction term in the two dimensional equation.

Integrating the three dimensional equation by parts we recover the equation in strong form as well as the boundary conditions

$$\begin{aligned}(\nabla u_{3D} + \nabla u_{2D}, \varphi_{3D}) + \langle\alpha, \varphi_{3D}\rangle_{\Gamma_{\text{top}}\cup\Gamma_{\text{bottom}}} &= (\nabla u, \nabla\varphi_{3D}) + \langle\alpha, \varphi_{3D}\rangle_{\Gamma_{\text{top}}\cup\Gamma_{\text{bottom}}}\\ &= -(\Delta u, \varphi_{3D}) + \langle\partial_n u, \varphi_{3D}\rangle + \langle\alpha, \varphi_{3D}\rangle_{\Gamma_{\text{top}}\cup\Gamma_{\text{bottom}}}.\end{aligned}$$

Therefore we obtain

$$\langle\partial_n u, \varphi_{3D}\rangle_{\Gamma_{\text{top}}\cup\Gamma_{\text{bottom}}} + \langle\alpha, \varphi_{3D}\rangle_{\Gamma_{\text{top}}\cup\Gamma_{\text{bottom}}} = 0,$$

since the test function φ_{3D} vanishes on all the vertical boundaries.

4.1.4 Numerical results

As a test case we consider a problem in three dimensions. This reads: Find $u \in W$ such that

$$(\nabla u, \nabla \varphi) + \langle \alpha, \varphi \rangle_{\Gamma_{\text{top}} \cup \Gamma_{\text{bottom}}} = (f, \varphi) \text{ for all } \varphi \in W. \tag{4.10}$$

on a domain $\Omega = (-1, 1)^3$ subject to homogeneous boundary conditions on the vertical faces. We take the right hand side to be $f \equiv 1$. To show that the coupled method works and yields

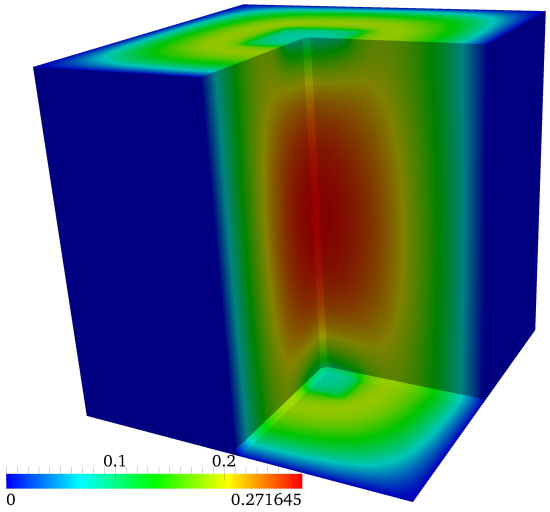


Figure 4.2. Reference solution of three dimensional Poisson problem

results that converge towards the three dimensional solution we compare the L^2 and the H^1 semi-norm as well as point values of the solution. We take a look at the points $p_1 = (0.5, 0.5, 0)$, $p_2 = (0, 0, 0)$, $p_3 = (0.5, 0.5, 0.5)$, $p_4 = (0, 0, 0.5)$, $p_5 = (0.5, 0.5, 1)$, $p_6 = (0, 0, 1)$.

The solution is computed using trilinear finite elements and it is shown in Figure 4.2. In Section 5.3 we explain the finite element spaces more detailed. Results for global refinement are stated in Table 4.1 and Table 4.2. We applied the residual based error estimator from [69] for local refinement. Those results are shown in Table 4.3 and Table 4.4. The corresponding meshes obtained for adaptive refinement are shown in Figure 4.3.

We show three different coupled problems for the solution of (4.10).

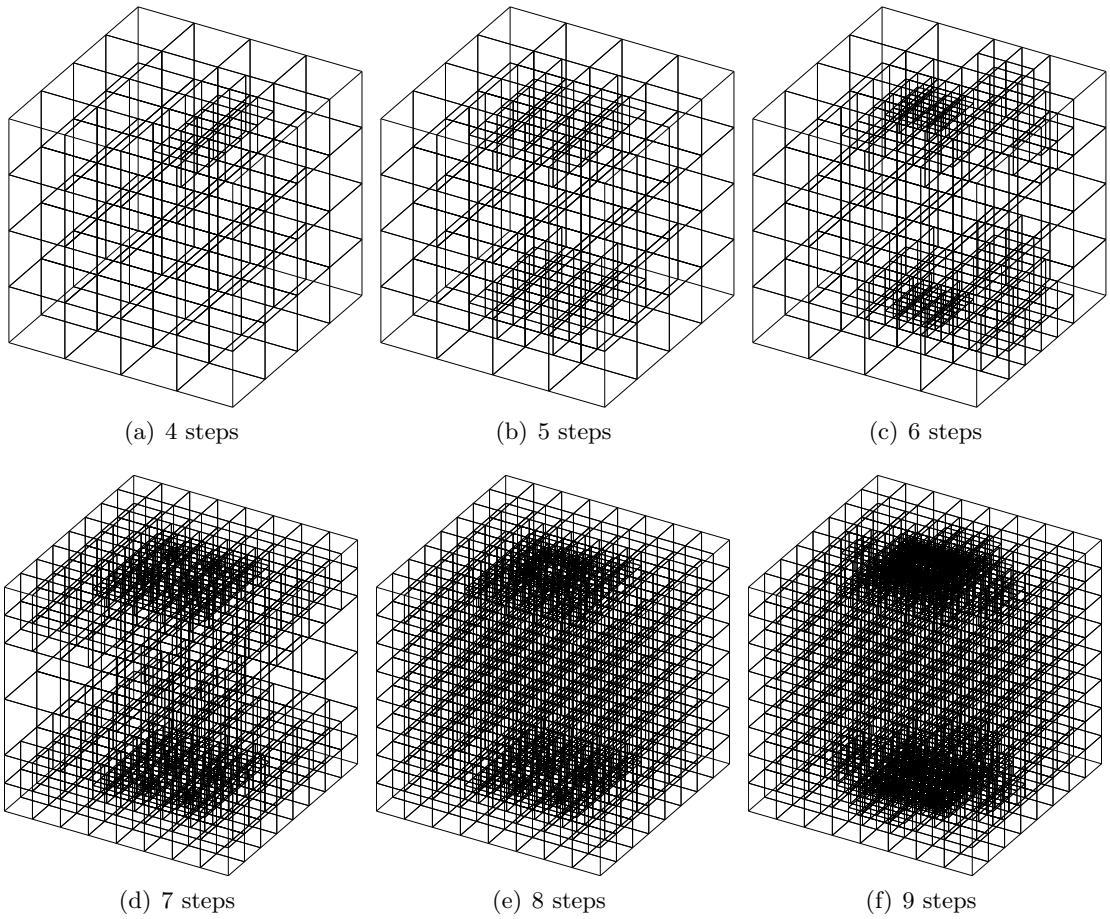
Test 4.1. For the first numerical test we choose the part for the three dimensional domain to be the whole cube. We solve for $u = (u_{2D}, u_{3D)\in W_{2D} \times W_{3D}}$ such that

$$\begin{aligned} (\nabla u_{3D}, \nabla \varphi_{3D}) + (\nabla u_{2D}, \nabla \varphi_{3D}) + \langle \alpha, \varphi_{3D} \rangle_{\Gamma_{\text{top}} \cup \Gamma_{\text{bottom}}} &= (f, \varphi_{3D}), \quad \text{in } \Omega, \\ 2(\nabla u_{2D}, \nabla \varphi_{2D}) + 2(\alpha, \varphi_{2D})_{\Omega_{2D}} &= (f, \varphi_{2D}), \quad \text{in } \Omega_{2D} = (-1, 1)^2, \end{aligned}$$

for all $\varphi = (\varphi_{2D}, \varphi_{3D}) \in W_{2D} \times W_{3D}$. The spaces W_{3D} and W_{2D} are defined as follows

$$W_{3D} = \{u \in H^1(\Omega) \mid u = 0 \text{ on } \Gamma = \partial\Omega \setminus (\Gamma_{\text{top}} \cup \Gamma_{\text{bottom}})\}, \tag{4.11}$$

$$W_{2D} = \{u \in H^1(\Omega_{2D}) \mid u = 0 \text{ on } \partial\Omega_{2D}\}. \tag{4.12}$$



(a) 4 steps (b) 5 steps (c) 6 steps

(d) 7 steps (e) 8 steps (f) 9 steps

Figure 4.3. Sequence of adaptively refined meshes after various refinement steps

Table 4.1. Results in the L^2 norm and H^1 semi norm for the Poisson problem computed in three dimensions on a globally refined mesh

# cells	# DoFs	$\|u\|$	$\|\nabla u\|$
1	27	$3.2503 \cdot 10^{-1}$	$8.0393 \cdot 10^{-1}$
8	125	$3.9638 \cdot 10^{-1}$	$9.4696 \cdot 10^{-1}$
64	729	$3.7560 \cdot 10^{-1}$	$9.2638 \cdot 10^{-1}$
512	4913	$4.0605 \cdot 10^{-1}$	$9.7045 \cdot 10^{-1}$
4096	35937	$4.0668 \cdot 10^{-1}$	$9.7203 \cdot 10^{-1}$
32768	274625	$4.0687 \cdot 10^{-1}$	$9.7251 \cdot 10^{-1}$
262144	2146689	$4.0692 \cdot 10^{-1}$	$9.7265 \cdot 10^{-1}$

Table 4.2. Point values for the Poisson problem computed in three dimensions on a globally refined mesh

# cells	# DoFs	$u(p_1)$	$u(p_2)$	$u(p_3)$	$u(p_4)$	$u(p_5)$	$u(p_6)$
1	27	0.1536	0.2730	0.1258	0.2237	0.0426	0.0758
8	125	0.1705	0.2402	0.1607	0.2512	0.1452	-0.0248
64	729	0.1647	0.2597	0.1557	0.2318	0.1375	0.0634
512	4913	0.1705	0.2713	0.1651	0.2498	0.1564	0.0838
4096	35937	0.1706	0.2716	0.1653	0.2502	0.1568	0.0772
32768	274625	0.1706	0.2716	0.1654	0.2504	0.1569	0.0770
262144	2146689	0.1706	0.2717	0.1654	0.2504	0.1569	0.0770

Table 4.3. Results for the Poisson problem computed in three dimensions on an adaptively refined mesh

# cells	# DoFs	$\|u\|$	$\|\nabla u\|$
1	27	$3.2503 \cdot 10^{-1}$	$8.0393 \cdot 10^{-1}$
8	125	$3.9638 \cdot 10^{-1}$	$9.4696 \cdot 10^{-1}$
29	407	$3.8721 \cdot 10^{-1}$	$9.3508 \cdot 10^{-1}$
50	621	$3.8001 \cdot 10^{-1}$	$9.2880 \cdot 10^{-1}$
78	936	$3.8264 \cdot 10^{-1}$	$9.3639 \cdot 10^{-1}$
120	1469	$4.0621 \cdot 10^{-1}$	$9.6999 \cdot 10^{-1}$
211	2637	$4.0658 \cdot 10^{-1}$	$9.7093 \cdot 10^{-1}$
589	7019	$4.0686 \cdot 10^{-1}$	$9.7187 \cdot 10^{-1}$
848	9089	$4.0677 \cdot 10^{-1}$	$9.7221 \cdot 10^{-1}$
1779	19607	$4.0690 \cdot 10^{-1}$	$9.7252 \cdot 10^{-1}$
3109	32753	$4.0691 \cdot 10^{-1}$	$9.7256 \cdot 10^{-1}$
5153	49335	$4.0690 \cdot 10^{-1}$	$9.7258 \cdot 10^{-1}$
8870	88697	$4.0693 \cdot 10^{-1}$	$9.7266 \cdot 10^{-1}$
19181	200062	$4.0694 \cdot 10^{-1}$	$9.7268 \cdot 10^{-1}$
32859	331139	$4.0694 \cdot 10^{-1}$	$9.7268 \cdot 10^{-1}$
51276	473601	$4.0694 \cdot 10^{-1}$	$9.7268 \cdot 10^{-1}$
94823	924048	$4.0694 \cdot 10^{-1}$	$9.7269 \cdot 10^{-1}$

Table 4.4. Point values for the Poisson problem computed in three dimensions on an adaptively refined mesh

# cells	# DoFs	$u(p_1)$	$u(p_2)$	$u(p_3)$	$u(p_4)$	$u(p_5)$	$u(p_6)$
1	27	0.1536	0.2730	0.1258	0.2237	0.0426	0.0758
8	125	0.1705	0.2402	0.1607	0.2512	0.1452	-0.0248
29	407	0.1662	0.2403	0.1563	0.2401	0.1356	-0.0023
50	621	0.1650	0.2454	0.1559	0.2388	0.1354	-0.0016
78	936	0.1664	0.2622	0.1616	0.2383	0.1533	0.0660
120	1469	0.1704	0.2710	0.1652	0.2490	0.1572	0.0838
211	2637	0.1705	0.2712	0.1653	0.2493	0.1562	0.0772
589	7019	0.1706	0.2711	0.1651	0.2494	0.1569	0.0773
848	9089	0.1706	0.2716	0.1653	0.2503	0.1569	0.0770
1779	19607	0.1706	0.2716	0.1654	0.2504	0.1569	0.0770
3109	32753	0.1706	0.2716	0.1654	0.2504	0.1569	0.0770
5153	49335	0.1706	0.2717	0.1654	0.2504	0.1569	0.0770
8870	88697	0.1706	0.2717	0.1654	0.2504	0.1569	0.0770
19181	200062	0.1706	0.2717	0.1654	0.2504	0.1569	0.0770
32859	331139	0.1706	0.2717	0.1654	0.2504	0.1569	0.0770
51276	473601	0.1706	0.2717	0.1654	0.2504	0.1569	0.0770
94823	924048	0.1706	0.2717	0.1654	0.2504	0.1569	0.0770

Results for Test 4.1 are shown in Table 4.5 and Table 4.6 for global refinement in 3D. According to the refinement of the mesh in three dimensions, the two dimensional mesh is refined. The solution of the coupled problem was obtained using bilinear finite elements in two dimensions and trilinear finite elements in three dimensions.

Table 4.5. Results in the L^2 norm and H^1 semi norm for the coupled Poisson problem on a globally refined mesh obtained for the first test

# cells	# DoFs	$\|u\|$	$\|\nabla u\|$
8	27	$2.3470 \cdot 10^{-1}$	$6.3217 \cdot 10^{-1}$
64	125	$3.7709 \cdot 10^{-1}$	$9.1092 \cdot 10^{-1}$
512	729	$3.9462 \cdot 10^{-1}$	$9.4605 \cdot 10^{-1}$
4096	4913	$4.0322 \cdot 10^{-1}$	$9.6450 \cdot 10^{-1}$
32768	35937	$4.0583 \cdot 10^{-1}$	$9.7020 \cdot 10^{-1}$
262144	274625	$4.0661 \cdot 10^{-1}$	$9.7195 \cdot 10^{-1}$
2097152	2146689	$4.0685 \cdot 10^{-1}$	$9.7247 \cdot 10^{-1}$

For the next coupled solution we choose the three dimensional domain to be a little bit more narrow than in Test 4.1.

Table 4.6. Point values for the coupled Poisson problem on a globally refined mesh obtained for the first test

# cells	# DoFs	$u(p_1)$	$u(p_2)$	$u(p_3)$	$u(p_4)$	$u(p_4)$	$u(p_6)$
8	27	0.0938	0.3750	0.0589	0.2356	0.0240	0.0961
64	125	0.1822	0.2882	0.1745	0.2854	0.1630	0.0478
512	729	0.1724	0.2744	0.1658	0.2535	0.1548	0.0506
4096	4913	0.1709	0.2721	0.1653	0.2507	0.1563	0.0750
32768	35937	0.1707	0.2717	0.1653	0.2504	0.1567	0.0762
262144	274625	0.1706	0.2717	0.1654	0.2504	0.1568	0.0768
2097152	2146689	0.1706	0.2717	0.1654	0.2504	0.1569	0.0770

Test 4.2. We solve for $u = (u_{2D}, u_{3D}) \in W_{2D} \times W_{3D}$ such that

$$\begin{aligned}(\nabla(u_{3D} + u_{2D}), \nabla\varphi_{3D}) + \langle\alpha, \varphi_{3D}\rangle_{\Gamma_{\text{top}}\cup\Gamma_{\text{bottom}}} &= (f, \varphi_{3D}), \quad \text{in } \Omega = \left(-\frac{1}{2}, \frac{1}{2}\right)^2 \times (-1, 1),\\ 2(\nabla u_{2D}, \nabla\varphi_{2D}) + 2(\alpha, \varphi_{2D})_{\Omega_{2D}} &= (f, \varphi_{2D}), \quad \text{in } \Omega_{2D} = (-1, 1)^2,\end{aligned}$$

for all $\varphi = (\varphi_{2D}, \varphi_{3D}) \in W_{2D} \times W_{3D}$. The spaces W_{3D} and W_{2D} are defined in (4.11).

The solution of the coupled problem using bilinear finite elements in two dimensions and trilinear finite elements in three dimensions is shown in Figure 4.4. We used global refinement

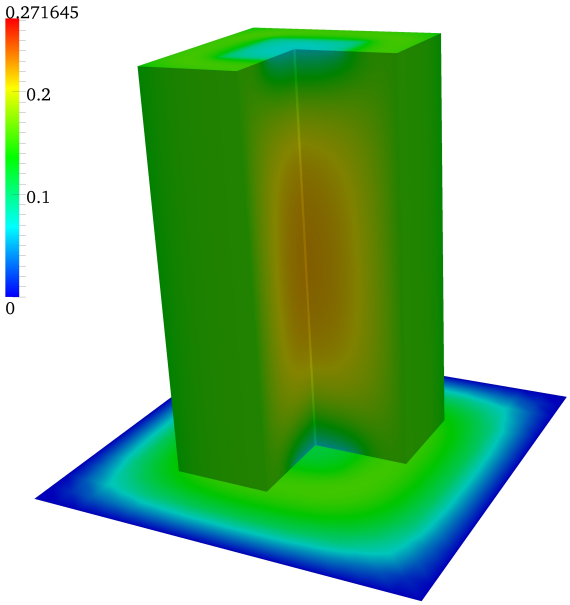


Figure 4.4. Coupled solution of the three dimensional Poisson problem obtained for the second test

of the three dimensional mesh. The mesh in 2D was refined accordingly as described in

Section 5.5.1. The results we obtained for Test 4.2 in the L^2 norm are shown in Table 4.7 and the H^1 semi-norm is presented in Table 4.8. Point values for Test 4.2 are given in Table 4.9.

Table 4.7. Results in the L^2 norm for the coupled Poisson problem on a globally refined mesh obtained for the second test

# cells	# DoFs	$\|u_{2D}\|$	$\|u_{3D}\|$	$\|u\|$
8	27	$2.0577 \cdot 10^{-1}$	$2.6339 \cdot 10^{-1}$	$4.6917 \cdot 10^{-1}$
64	125	$1.5867 \cdot 10^{-1}$	$2.7182 \cdot 10^{-1}$	$4.3048 \cdot 10^{-1}$
512	729	$1.7229 \cdot 10^{-1}$	$2.7373 \cdot 10^{-1}$	$4.4602 \cdot 10^{-1}$
4096	4913	$1.7300 \cdot 10^{-1}$	$2.7470 \cdot 10^{-1}$	$4.4771 \cdot 10^{-1}$
32768	35937	$1.7229 \cdot 10^{-1}$	$2.7507 \cdot 10^{-1}$	$4.4737 \cdot 10^{-1}$
262144	274625	$1.7178 \cdot 10^{-1}$	$2.7520 \cdot 10^{-1}$	$4.4698 \cdot 10^{-1}$
2097152	2146689	$1.7150 \cdot 10^{-1}$	$2.7524 \cdot 10^{-1}$	$4.4674 \cdot 10^{-1}$

Table 4.8. Results in the H^1 semi norm for the coupled Poisson problem on a globally refined mesh obtained for the second test

# cells	# DoFs	$\|\nabla u_{2D}\|$	$\|\nabla u_{3D}\|$	$\|\nabla u\|$
8	27	$6.2818 \cdot 10^{-1}$	$2.3177 \cdot 10^{-1}$	$8.5995 \cdot 10^{-1}$
64	125	$7.4572 \cdot 10^{-1}$	$2.6138 \cdot 10^{-1}$	$1.0071 \cdot 10^{0}$
512	729	$9.4619 \cdot 10^{-1}$	$2.7756 \cdot 10^{-1}$	$1.2238 \cdot 10^{0}$
4096	4913	$1.2292 \cdot 10^{0}$	$2.8235 \cdot 10^{-1}$	$1.5116 \cdot 10^{0}$
32768	35937	$1.6544 \cdot 10^{0}$	$2.8400 \cdot 10^{-1}$	$1.9384 \cdot 10^{0}$
262144	274625	$2.2778 \cdot 10^{0}$	$2.8457 \cdot 10^{-1}$	$2.5624 \cdot 10^{0}$
2097152	2146689	$3.1767 \cdot 10^{0}$	$2.8476 \cdot 10^{-1}$	$3.4614 \cdot 10^{0}$

Table 4.9. Point values for the coupled Poisson problem on a globally refined mesh obtained for the second test

# cells	# DoFs	$u(p_1)$	$u(p_2)$	$u(p_3)$	$u(p_4)$	$u(p_5)$	$u(p_6)$
8	27	0.1597	0.3336	0.1597	0.2016	0.1597	0.0697
64	125	0.1593	0.2513	0.1593	0.2525	0.1593	0.0609
512	729	0.1526	0.2452	0.1526	0.2331	0.1526	0.0702
4096	4913	0.1511	0.2435	0.1511	0.2298	0.1511	0.0719
32768	35937	0.1507	0.2431	0.1507	0.2292	0.1507	0.0725
262144	274625	0.1506	0.2431	0.1506	0.2291	0.1506	0.0726
2097152	2146689	0.1506	0.2431	0.1506	0.2291	0.1506	0.0727

In the last test case we use an even smaller domain in three dimensions.

Test 4.3. We solve for $u = (u_{2D}, u_{3D}) \in W_{2D} \times W_{3D}$ such that

$$(\nabla(u_{3D} + u_{2D}), \nabla\varphi_{3D}) + \langle \alpha, \varphi_{3D} \rangle_{\Gamma_{\text{top}} \cup \Gamma_{\text{bottom}}} = (f, \varphi_{3D}), \quad \text{in } \Omega_{3D} = \left(-\frac{1}{3}, \frac{1}{3}\right)^2 \times (-1, 1),$$
$$2(\nabla u_{2D}, \nabla\varphi_{2D}) + 2(\alpha, \varphi_{2D})_{\Omega_{2D}} = (f, \varphi_{2D}), \quad \text{in } \Omega_{2D} = (-1, 1)^2,$$

for all $\varphi = (\varphi_{2D}, \varphi_{3D}) \in W_{2D} \times W_{3D}$. The definition of the spaces is given in (4.11).

The results we obtained for Test 4.3 in the L^2 norm are displayed in Table 4.10 and the results in the H^1 semi-norm are shown in Table 4.11. Point values are given in Table 4.12 for Test 4.3.

Table 4.10. Results in the L^2 norm for the coupled Poisson problem on a globally refined mesh obtained for the third test

# cells	# DoFs	$\|u_{2D}\|$	$\|u_{3D}\|$	$\|u\|$
1	8	$1.7778 \cdot 10^{-1}$	$1.5085 \cdot 10^{-1}$	$3.2863 \cdot 10^{-1}$
8	27	$2.1068 \cdot 10^{-1}$	$2.1740 \cdot 10^{-1}$	$4.2809 \cdot 10^{-1}$
64	125	$2.1199 \cdot 10^{-1}$	$1.8034 \cdot 10^{-1}$	$3.9234 \cdot 10^{-1}$
512	729	$2.1513 \cdot 10^{-1}$	$1.8053 \cdot 10^{-1}$	$3.9567 \cdot 10^{-1}$
4096	4913	$2.1495 \cdot 10^{-1}$	$1.8088 \cdot 10^{-1}$	$3.9583 \cdot 10^{-1}$
32768	35937	$2.1458 \cdot 10^{-1}$	$1.8100 \cdot 10^{-1}$	$3.9558 \cdot 10^{-1}$
262144	274625	$2.1435 \cdot 10^{-1}$	$1.8104 \cdot 10^{-1}$	$3.9540 \cdot 10^{-1}$
2097152	2146689	$2.1423 \cdot 10^{-1}$	$1.8106 \cdot 10^{-1}$	$3.9529 \cdot 10^{-1}$

Table 4.11. Results in the H^1 semi norm for the coupled Poisson problem on a globally refined mesh obtained for the third test

# cells	# DoFs	$\|\nabla u_{2D}\|$	$\|\nabla u_{3D}\|$	$\|\nabla u\|$
1	8	$4.1312 \cdot 10^{-1}$	$2.3440 \cdot 10^{-16}$	$4.1312 \cdot 10^{-1}$
8	27	$7.0249 \cdot 10^{-1}$	$1.5412 \cdot 10^{-1}$	$8.5661 \cdot 10^{-1}$
64	125	$7.8734 \cdot 10^{-1}$	$2.0658 \cdot 10^{-1}$	$9.9392 \cdot 10^{-1}$
512	729	$9.9631 \cdot 10^{-1}$	$2.1057 \cdot 10^{-1}$	$1.2069 \cdot 10^{0}$
4096	4913	$1.3167 \cdot 10^{0}$	$2.1341 \cdot 10^{-1}$	$1.5301 \cdot 10^{0}$
32768	35937	$1.7931 \cdot 10^{0}$	$2.1442 \cdot 10^{-1}$	$2.0075 \cdot 10^{0}$
262144	274625	$2.4856 \cdot 10^{0}$	$2.1480 \cdot 10^{-1}$	$2.7004 \cdot 10^{0}$
2097152	2146689	$3.4791 \cdot 10^{0}$	$2.1494 \cdot 10^{-1}$	$3.6940 \cdot 10^{0}$

4.1.5 Discussion of the numerical results

For a first example we choose an easy problem with a three dimensional solution. The results we obtained for the Test 4.1 are in really good agreement with the real three dimensional solution, see Table 4.1 and Table 4.2 and Table 4.5 and Table 4.6. While the L^2 norm is still very good as the three dimensional part shrinks, the H^1 semi norm is a little worse, as can be seen in Table 4.8 and Table 4.11.

This means the coupling between different dimensions as we introduced it for the Poisson problem works well and we carry the method over to the three dimensional problem we derived for simulation of fluid flows in Chapter 3. In the following section we derive the coupled method for fluid flows with a free boundary.

Table 4.12. Point values for the coupled Poisson problem on a globally refined mesh obtained for the third test

# cells	# DoFs	$u(p_2)$	$u(p_4)$	$u(p_6)$
1	8	0.1600	0.1600	0.1600
8	27	0.3336	0.2290	0.1245
64	125	0.2225	0.2339	0.0756
512	729	0.2199	0.2170	0.0725
4096	4913	0.2192	0.2138	0.0728
32768	35937	0.2190	0.2131	0.0730
262144	274625	0.2190	0.2130	0.0731
2097152	2146689	0.2189	0.2130	0.0731

4.2 Coupling of 2D Shallow-Water and 3D Boussinesq model

This section describes the coupling of two fluid flow models. We start from the most complex model in three dimensions, the Boussinesq equations. We recall the variational formulation Multiplying with a test-function and integration by parts yields

$$\begin{aligned}(\partial_t v, \varphi_v) + ((v\cdot\nabla)v, \varphi_v) + (\gamma\nabla p, \varphi_v) + (\nu\nabla v, \nabla\varphi_v) - (\varrho\gamma \boldsymbol{g} T, \varphi_v) &= (f, \varphi_v),\\ (\nabla\cdot v, \varphi_p) &= 0,\\ (\partial_t T, \varphi_T) + (v\cdot\nabla T, \varphi_T) + (\nu_T\nabla T, \nabla\varphi_T) &= (q_T, \varphi_T),\end{aligned}$$

where $\boldsymbol{g} = (0, 0, -g)$.

We define a semi-linear form $a : X \times \widetilde{X} \to \mathbb{R}$. Since we end up with the same test and trial space after time discretization, we restrict this discussion to $X = \widetilde{X}$.

$$\begin{aligned}a(u)(\varphi) = (\partial_t v, \varphi_v) + ((v\cdot\nabla)v, \varphi_v) + (\gamma\nabla p, \varphi_v) + (\nu\nabla v, \nabla\varphi_v) + (\nabla\cdot v, \varphi_p)\\ -(\varrho\gamma \boldsymbol{g} T, \varphi_v) + (\partial_t T, \varphi_T) + (v\cdot\nabla T, \varphi_T) + (\nu_T\nabla T, \nabla\varphi_T).\end{aligned} \tag{4.13}$$

The problem then reads: Find $u = (v, p, T) \in X$ such that

$$a(u)(\varphi) = (f, \varphi_v) + (q_T, \varphi_T) =: (F, \varphi), \quad \text{for all } \varphi = (\varphi_v, \varphi_p, \varphi_T) \in X.$$

The explicit definition of the spaces X and $\widetilde{X}$ is given in (3.12) and (3.13), respectively. We apply the splitting of the space X in which we are looking for the solution into a sum:

$$X = X_{2D} + X_{3D}. \tag{4.14}$$

The solution of the coupled problem then reads

$$u = u_{2D} + u_{3D}. \tag{4.15}$$

For the coupled problem we look for a coupled solution $u = u_{2D} + u_{3D}$ such that

$$a(u)(\varphi) = (F, \varphi), \quad \text{for all } \varphi \in X,$$

or equivalently

$$a(u_{2D} + u_{3D})(\varphi_{2D} + \varphi_{3D}) = (F, \varphi_{2D}) + (F, \varphi_{3D}),$$

for all $\varphi_{2D} \in X_{2D}$ and for all $\varphi_{3D} \in X_{3D}$. This means we understand a function $v \in X_{2D}$ as a function in X_{3D} by setting the third component to zero:

$$v = \begin{pmatrix} v_1 \\ v_2 \end{pmatrix} \in X_{2D} \mathrel{\widehat{=}} v = \begin{pmatrix} v_1 \\ v_2 \\ 0 \end{pmatrix} \in X_{3D}.$$

If it is clear from the context we write differential operators in three dimensions for functions in W_{2D} by taking the corresponding function in W_{3D}. As an example the Jacobian matrix is given as

$$\nabla v = \nabla \begin{pmatrix} v_1 \\ v_2 \\ 0 \end{pmatrix} = \begin{pmatrix} \partial_1 v_1 & \partial_2 v_1 & 0 \\ \partial_1 v_2 & \partial_2 v_2 & 0 \\ 0 & 0 & 0 \end{pmatrix}.$$

We understand a vertically averaged velocity and the height in two dimensions as

$$u_{2D} = \frac{1}{h} \int_0^h u_{3D} \ \mathrm{d}x_3, \quad p_{3D} = \varrho g(h - x_3). \tag{4.16}$$

Varying the test functions φ_{2D} and φ_{3D} independently we arrive at two equations:

$$\begin{aligned} a(u_{2D} + u_{3D})(\varphi_{3D}) &= (F, \varphi_{3D}) \text{ for all } \varphi_{3D} \in X_{3D}, \\ a(u_{2D} + u_{3D})(\varphi_{2D}) &= (F, \varphi_{2D}) \text{ for all } \varphi_{2D} \in X_{2D}, \end{aligned}$$

We further investigate the two equations independently.

4.2.1 First equation

We start to consider the first equation

$$a(u_{2D} + u_{3D})(\varphi_{3D}) = (F, \varphi_{3D}) \text{ for all } \varphi_{3D} \in X_{3D}.$$

Since we test only with φ_{3D} we omit the subscript and write φ instead:

$$\begin{aligned} a(u_{2D} + u_{3D})(\varphi) = (\partial_t(v_{3D} + v_{2D}), \varphi_v) + (((v_{3D} + v_{2D}) \cdot \nabla)(v_{3D} + v_{2D}), \varphi_v) \\ + (\gamma \nabla(p_{3D} + h), \varphi_v) + (\nu \nabla(v_{3D} + v_{2D}), \nabla \varphi_v) + (\varrho \gamma \boldsymbol{g}(T_{3D} + T_{2D}), \varphi_v) \\ + (g \nabla h, \varphi_v) + (\nabla \cdot (v_{3D} + v_{2D}), \varphi_p) + (\partial_t(T_{3D} + T_{2D}), \varphi_T) \\ + ((v_{3D} + v_{2D}) \cdot \nabla(T_{3D} + T_{2D}), \varphi_T) + (\nu_T \nabla(T_{3D} + T_{2D}), \nabla \varphi_T). \end{aligned}$$

Here we leave the semi-linear form as it is. The two dimensional functions ensure the correct coupling.

4.2.2 Second equation

It remains to take a look at the second equation

$$a(u_{2D} + u_{3D})(\varphi_{2D}) = (F, \varphi_{2D}) \text{ for all } \varphi_{2D} \in V_{2D}.$$

Since we test only with φ_{2D} we omit the subscript and write φ instead:

$$\begin{aligned} a(u_{2D} + u_{3D})(\varphi) = (\partial_t(v_{3D} + v_{2D}), \varphi_v) + (((v_{3D} + v_{2D})\cdot\nabla)(v_{3D} + v_{2D}), \varphi_v) \\ +(\gamma\nabla(p_{3D} + h), \varphi_v) + (\nu\nabla(v_{3D} + v_{2D}), \nabla\varphi_v) + (\varrho\gamma\boldsymbol{g}(T_{3D} + T_{2D}), \varphi_v) \\ +(g\nabla h, \varphi_v) + (\nabla\cdot(v_{3D} + v_{2D}), \varphi_p) + (\partial_t(T_{3D} + T_{2D}), \varphi_T) \\ +((v_{3D} + v_{2D})\cdot\nabla(T_{3D} + T_{2D}), \varphi_T) + (\nu_T\nabla(T_{3D} + T_{2D}), \nabla\varphi_T). \end{aligned}$$

We are interested to investigate the coupling between the functions in 2D and 3D. In the following section we consider the linear terms contributing to the semi-linear form.

4.2.3 Discussion of the linear coupling terms

We start with the original term as an integral on the three dimensional domain Ω. In order to keep the notation short we write $\mathrm{d}x$ for $\mathrm{d}x_1\,\mathrm{d}x_2\,\mathrm{d}x_3$ and $\mathrm{d}x_{ij}$ for $\mathrm{d}x_i\,\mathrm{d}x_j$ in the following.

$(\partial_t v_{3D}, \varphi_{v_{2D}})_\Omega$:

Changing the order of differentiation and integration and using the definition in (4.16) we get

$$\begin{aligned} (\partial_t v_{3D}, \varphi_{v_{2D}})_\Omega &= \int_\Omega \partial_t v_{3D} \cdot \varphi_{v_{2D}}\,\mathrm{d}x, \\ &= \int_\Omega \partial_t v^1_{3D}\varphi^1_{v_{2D}}\,\mathrm{d}x + \int_\Omega \partial_t v^2_{3D}\varphi^2_{v_{2D}}\,\mathrm{d}x, \\ &= \int_{\Omega_{2D}} \varphi^1_{v_{2D}}\ \partial_t \int_0^h v^1_{3D}\,\mathrm{d}x_3\,\mathrm{d}x_{12} + \int_{\Omega_{2D}} \varphi^2_{v_{2D}}\ \partial_t \int_0^h v^2_{3D}\,\mathrm{d}x_3\,\mathrm{d}x_{12}, \\ &= \int_{\Omega_{2D}} \varphi^1_{v_{2D}}\ \partial_t(h v^1_{2D})\,\mathrm{d}x_{12} + \int_{\Omega_{2D}} \varphi^2_{v_{2D}}\ \partial_t(h v^2_{2D})\,\mathrm{d}x_{12}, \\ &= (\partial_t(h v_{2D}), \varphi_{v_{2D}})_{\Omega_{2D}}. \end{aligned}$$

The same procedure is applied to the next term including the pressure.

$(\gamma\nabla p_{3D}, \varphi_{v_{2D}})_\Omega$:

For the investigation of this term we recall the relation between the pressure and the height:

$$p_{3D} = \varrho g(h - x_3).$$

The gradient of the pressure can be expressed as

$$\nabla p_{3D} = \varrho g \begin{pmatrix} \partial_1 h \\ \partial_2 h \\ -1 \end{pmatrix}.$$

Taking this into account we can rewrite the integral as

$$\begin{aligned}
(\gamma \nabla p_{3D}, \varphi_{v_{2D}})_\Omega &= \gamma \int_\Omega \partial_1 p_{3D} \varphi^1_{v_{2D}} \,\mathrm{d}x + \gamma \int_\Omega \partial_2 p_{3D} \varphi^2_{v_{2D}} \,\mathrm{d}x, \\
&= \gamma \int_{\Omega_{2D}} \int_0^h \varrho g \partial_1 h \varphi^1_{v_{2D}} \,\mathrm{d}x_3 \,\mathrm{d}x_{12} + \gamma \int_{\Omega_{2D}} \int_0^h \varrho g \partial_2 h \varphi^2_{v_{2D}} \,\mathrm{d}x_3 \,\mathrm{d}x_{12}, \\
&= \gamma \int_{\Omega_{2D}} h \varrho g \partial_1 h \varphi^1_{v_{2D}} \,\mathrm{d}x_{12} + \gamma \int_{\Omega_{2D}} h \varrho g \partial_2 h \varphi^2_{v_{2D}} \,\mathrm{d}x_{12}, \\
&= (\gamma \varrho g h \nabla h, \varphi_{v_{2D}})_{\Omega_{2D}}.
\end{aligned}$$

$(\nu \nabla v_{3D}, \nabla \varphi_{v_{2D}})_\Omega$:

$$\begin{aligned}
(\nu \nabla v_{3D}, \nabla \varphi_{v_{2D}})_\Omega &= \int_\Omega \nu \nabla v_{3D} : \nabla \varphi_{v_{2D}} \,\mathrm{d}x, \\
&= \int_\Omega \nu \partial_1 v^1_{3D} \partial_1 \varphi^1_{v_{2D}} \,\mathrm{d}x + \int_\Omega \nu \partial_2 v^1_{3D} \partial_2 \varphi^1_{v_{2D}} \,\mathrm{d}x \\
&+ \int_\Omega \nu \partial_1 v^2_{3D} \partial_1 \varphi^2_{v_{2D}} \,\mathrm{d}x + \int_\Omega \nu \partial_2 v^2_{3D} \partial_2 \varphi^2_{v_{2D}} \,\mathrm{d}x, \\
&= \int_{\Omega_{2D}} \nu \partial_1 \varphi^1_{v_{2D}} \; \partial_1 \int_0^h v^1_{3D} \,\mathrm{d}x_3 \,\mathrm{d}x_{12} + \int_{\Omega_{2D}} \nu \partial_2 \varphi^1_{v_{2D}} \; \partial_2 \int_0^h v^1_{3D} \,\mathrm{d}x_3 \,\mathrm{d}x_{12} \\
&+ \int_{\Omega_{2D}} \nu \partial_1 \varphi^2_{v_{2D}} \; \partial_1 \int_0^h v^2_{3D} \,\mathrm{d}x_3 \,\mathrm{d}x_{12} + \int_{\Omega_{2D}} \nu \partial_2 \varphi^2_{v_{2D}} \; \partial_2 \int_0^h v^2_{3D} \,\mathrm{d}x_3 \,\mathrm{d}x_{12}, \\
&= \int_{\Omega_{2D}} \nu \partial_1 \varphi^1_{v_{2D}} \; \partial_1 (h v^1_{2D}) \,\mathrm{d}x_{12} + \int_{\Omega_{2D}} \nu \partial_2 \varphi^1_{v_{2D}} \; \partial_2 (h v^1_{2D}) \,\mathrm{d}x_{12} \\
&+ \int_{\Omega_{2D}} \nu \partial_1 \varphi^2_{v_{2D}} \; \partial_1 (h v^2_{2D}) \,\mathrm{d}x_{12} + \int_{\Omega_{2D}} \nu \partial_2 \varphi^2_{v_{2D}} \; \partial_2 (h v^2_{2D}) \,\mathrm{d}x_{12}, \\
&= (\nu \nabla (h v_{2D}), \nabla \varphi_{v_{2D}})_{\Omega_{2D}}.
\end{aligned}$$

$(\varrho \gamma g \boldsymbol{T}_{3D}, \varphi_{v_{2D}})_\Omega$:

This term vanishes since the test function is two dimensional:

$$(\varrho \gamma \boldsymbol{g} T_{3D}, \varphi_{v_{2D}})_\Omega = \int_\Omega \varrho \gamma \boldsymbol{g} T_{3D} \varphi_{v_{2D}} \,\mathrm{d}x = -\int_\Omega \varrho \gamma g T^3_{3D} \varphi^3_{v_{2D}} \,\mathrm{d}x = 0.$$

$(\nabla \cdot v_{3D}, \varphi_{p_{2D}})_\Omega$:

Using Leibniz formula

$$\partial_i \int_0^h f \,\mathrm{d}x_3 = \int_0^h \partial_i f + f(x_1, x_2, h)\partial_i h, \quad i = 1, 2.$$

we find

$$\begin{aligned}
(\nabla \cdot v_{3D}, \varphi_{p_{2D}})_\Omega &= \int_\Omega \nabla \cdot v_{3D} \varphi_{p_{2D}} \,\mathrm{d}x = \int_{\Omega_{2D}} \int_0^h \nabla \cdot v_{3D} \varphi_{p_{2D}} \,\mathrm{d}x_3 \,\mathrm{d}x_{12}, \\
&= \int_{\Omega_{2D}} \varphi_{p_{2D}} \partial_1 \int_0^h v^1_{3D} \,\mathrm{d}x_3 - v^1_{3D}(x_1, x_2, h)\partial_1 h \,\mathrm{d}x_{12} \\
&+ \int_{\Omega_{2D}} \varphi_{p_{2D}} \partial_2 \int_0^h v^2_{3D} \,\mathrm{d}x_3 - v^2_{3D}(x_1, x_2, h)\partial_2 h \,\mathrm{d}x_{12} \\
&+ \int_{\Omega_{2D}} \varphi_{p_{2D}} \left(v^3_{3D}(x_1, x_2, h) - v^3_{3D}(x_1, x_2, 0) \right) \mathrm{d}x_{12}.
\end{aligned}$$

The function $\Phi(x_1, x_2, x_3, t) := x_3 - h(x_1, x_2, t)$ describes the surface elevation. For the total time derivative it holds

$$\begin{aligned}
d_t \Phi &= \partial_t \Phi + \nabla \Phi \cdot \partial_t x = \partial_t \Phi + \nabla \Phi \cdot v, \\
&= -\partial_t h - \partial_1 h v^1_{3D} - \partial_2 h v^2_{3D} + v^3_{3D}, \\
&= -\partial_t h + v_{3D} \cdot n = 0.
\end{aligned}$$

Taking the boundary conditions $v(x_1, x_2, h) \cdot n = 0$ at the free surface and $v(x_1, x_2, 0) = 0$ at the bottom into account, it follows that

$$\begin{aligned}
(\nabla \cdot v_{3D}, \varphi_{p_{2D}})_\Omega &= \int_{\Omega_{2D}} \varphi_{p_{2D}} \partial_1 (h v^1_{2D}) \,\mathrm{d}x_{12} + \int_{\Omega_{2D}} \varphi_{p_{2D}} \partial_2 (h v^2_{2D}) \,\mathrm{d}x_{12} \\
&+ \int_{\Omega_{2D}} \varphi_{p_{2D}} \partial_t h \,\mathrm{d}x_{12}, \\
&= (\partial_t h + \nabla \cdot (h v_{2D}), \varphi_{p_{2D}})_{\Omega_{2D}}
\end{aligned}$$

$(\partial_t T_{3D}, \varphi_{T_{2D}})_\Omega$:

For the time derivative of the temperature we can rewrite the three dimensional integral as

$$\begin{aligned}
(\partial_t T_{3D}, \varphi_{T_{2D}})_\Omega &= \int_\Omega \partial_t T_{3D} \cdot \varphi_{T_{2D}} \,\mathrm{d}x = \int_{\Omega_{2D}} \varphi_{T_{2D}} \; \partial_t \int_0^h T_{3D} \,\mathrm{d}x_3 \,\mathrm{d}x_{12}, \\
&= \int_{\Omega_{2D}} \varphi_{T_{2D}} \; \partial_t (h T_{2D}) \,\mathrm{d}x_{12} = (\partial_t (h T_{2D}), \varphi_{T_{2D}})_{\Omega_{2D}}.
\end{aligned}$$

$(\nu_T \nabla T_{3D}, \nabla \varphi_{T_{2D}})_\Omega$:

For the remaining linear term we derive

$$\begin{aligned}
&(\nu_T \nabla T_{3D}, \nabla \varphi_{T_{2D}})_\Omega \\
&\quad = \int_\Omega \nu_T \nabla T_{3D} \cdot \nabla \varphi_{T_{2D}} \,\mathrm{d}x = \int_\Omega \nu_T \partial_1 T_{3D} \partial_1 \varphi_{T_{2D}} \,\mathrm{d}x + \int_\Omega \nu_T \partial_2 T_{3D} \partial_2 \varphi_{T_{2D}} \,\mathrm{d}x, \\
&\quad = \int_{\Omega_{2D}} \nu_T \partial_1 \varphi_{T_{2D}} \; \partial_1 \int_0^h T_{3D} \,\mathrm{d}x_3 \,\mathrm{d}x_{12} + \int_{\Omega_{2D}} \nu_T \partial_2 \varphi_{T_{2D}} \; \partial_2 \int_0^h T_{3D} \,\mathrm{d}x_3 \,\mathrm{d}x_{12}, \\
&\quad = \int_{\Omega_{2D}} \nu_T \partial_1 \varphi_{T_{2D}} \; \partial_1 (h T_{2D}) \,\mathrm{d}x_{12} + \int_{\Omega_{2D}} \nu_T \partial_2 \varphi_{T_{2D}} \; \partial_2 (h T_{2D}) \,\mathrm{d}x_{12}, \\
&\quad = (\nu_T \nabla (h T_{2D}), \nabla \varphi_{T_{2D}})_{\Omega_{2D}}.
\end{aligned}$$

In the following we investigate the non-linear coupling terms that arise in the semi-linear form.

4.2.4 Discussion of the non-linear coupling terms

$(((v_{3D} + v_{2D}) \cdot \nabla)(v_{3D} + v_{2D}), \varphi_{v_{2D}})_\Omega$:

Before we can rewrite these terms as integrals in two dimensions, we write it as a sum of four terms. Each of these terms is treated separately.

$$\begin{aligned}
&(((v_{3D} + v_{2D}) \cdot \nabla)(v_{3D} + v_{2D}), \varphi_{v_{2D}})_\Omega = ((v_{3D} \cdot \nabla) v_{3D}, \varphi_{v_{2D}})_\Omega \\
&+ ((v_{3D} \cdot \nabla) v_{2D}, \varphi_{v_{2D}})_\Omega + ((v_{2D} \cdot \nabla) v_{3D}, \varphi_{v_{2D}})_\Omega + ((v_{2D} \cdot \nabla) v_{2D}, \varphi_{v_{2D}})_\Omega
\end{aligned}$$

For the first term in this sum we derive

$((v_{3D} \cdot \nabla) v_{3D}, \varphi_{v_{2D}})_\Omega$:

$$\begin{aligned}
&((v_{3D} \cdot \nabla) v_{3D}, \varphi_{v_{2D}})_\Omega \\
&= \int_\Omega v^1_{3D} \partial_1 v^1_{3D} \varphi^1_{v_{2D}} \,\mathrm{d}x + \int_\Omega v^2_{3D} \partial_2 v^1_{3D} \varphi^1_{v_{2D}} \,\mathrm{d}x + \int_\Omega v^3_{3D} \partial_3 v^1_{3D} \varphi^1_{v_{2D}} \,\mathrm{d}x \\
&+ \int_\Omega v^1_{3D} \partial_1 v^2_{3D} \varphi^2_{v_{2D}} \,\mathrm{d}x + \int_\Omega v^2_{3D} \partial_2 v^2_{3D} \varphi^2_{v_{2D}} \,\mathrm{d}x + \int_\Omega v^3_{3D} \partial_3 v^2_{3D} \varphi^2_{v_{2D}} \,\mathrm{d}x, \\
&= \int_{\Omega_{2D}} \int_0^h v^1_{3D} \partial_1 v^1_{3D} \varphi^1_{v_{2D}} \,\mathrm{d}x_3 \,\mathrm{d}x_{12} + \int_{\Omega_{2D}} \int_0^h v^2_{3D} \partial_2 v^1_{3D} \varphi^1_{v_{2D}} \,\mathrm{d}x_3 \,\mathrm{d}x_{12} \\
&+ \int_{\Omega_{2D}} \int_0^h v^3_{3D} \partial_3 v^1_{3D} \varphi^1_{v_{2D}} \,\mathrm{d}x_3 \,\mathrm{d}x_{12} + \int_{\Omega_{2D}} \int_0^h v^1_{3D} \partial_1 v^2_{3D} \varphi^2_{v_{2D}} \,\mathrm{d}x_3 \,\mathrm{d}x_{12} \\
&+ \int_{\Omega_{2D}} \int_0^h v^2_{3D} \partial_2 v^2_{3D} \varphi^2_{v_{2D}} \,\mathrm{d}x_3 \,\mathrm{d}x_{12} + \int_{\Omega_{2D}} \int_0^h v^3_{3D} \partial_3 v^2_{3D} \varphi^2_{v_{2D}} \,\mathrm{d}x_3 \,\mathrm{d}x_{12}.
\end{aligned}$$

Further manipulation shows

$$
\begin{aligned}
&((v_{3D}\cdot\nabla)v_{3D},\varphi_{v_{2D}})_\Omega \\
&\qquad = \int_{\Omega_{2D}} \varphi^1_{v_{2D}} \frac{1}{2}\int_0^h \partial_1((v^1_{3D})^2)\,\mathrm{d}x_3\,\mathrm{d}x_{12} \\
&\qquad + \int_{\Omega_{2D}} \varphi^1_{v_{2D}} \int_0^h \partial_2(v^2_{3D}v^1_{3D}) - v^1_{3D}\partial_2 v^2_{3D}\,\mathrm{d}x_3\,\mathrm{d}x_{12} \\
&\qquad + \int_{\Omega_{2D}} \varphi^1_{v_{2D}} \int_0^h \partial_3(v^3_{3D}v^1_{3D}) - v^1_{3D}\partial_3 v^3_{3D}\,\mathrm{d}x_3\,\mathrm{d}x_{12} \\
&\qquad + \int_{\Omega_{2D}} \varphi^2_{v_{2D}} \int_0^h \partial_1(v^1_{3D}v^2_{3D}) - v^2_{3D}\partial_1 v^1_{3D}\,\mathrm{d}x_3\,\mathrm{d}x_{12} \\
&\qquad + \int_{\Omega_{2D}} \varphi^2_{v_{2D}} \frac{1}{2}\int_0^h \partial_2((v^2_{3D})^2)\,\mathrm{d}x_3\,\mathrm{d}x_{12} \\
&\qquad + \int_{\Omega_{2D}} \varphi^2_{v_{2D}} \int_0^h \partial_3(v^3_{3D}v^2_{3D}) - v^2_{3D}\partial_3 v^3_{3D}\,\mathrm{d}x_3\,\mathrm{d}x_{12}.
\end{aligned}
$$

Using the definition of the divergence this results in

$$
\begin{aligned}
&((v_{3D}\cdot\nabla)v_{3D},\varphi_{v_{2D}})_\Omega \\
&\qquad = \int_{\Omega_{2D}} \varphi^1_{v_{2D}} \int_0^h \partial_1((v^1_{3D})^2)\,\mathrm{d}x_3\,\mathrm{d}x_{12} + \int_{\Omega_{2D}} \varphi^1_{v_{2D}} \int_0^h \partial_2(v^2_{3D}v^1_{3D})\,\mathrm{d}x_3\,\mathrm{d}x_{12} \\
&\qquad + \int_{\Omega_{2D}} \varphi^1_{v_{2D}} \int_0^h \partial_3(v^3_{3D}v^1_{3D})\,\mathrm{d}x_3\,\mathrm{d}x_{12} + \int_{\Omega_{2D}} \varphi^2_{v_{2D}} \int_0^h \partial_1(v^1_{3D}v^2_{3D})\,\mathrm{d}x_3\,\mathrm{d}x_{12} \\
&\qquad + \int_{\Omega_{2D}} \varphi^2_{v_{2D}} \int_0^h \partial_2((v^2_{3D})^2)\,\mathrm{d}x_3\,\mathrm{d}x_{12} + \int_{\Omega_{2D}} \varphi^2_{v_{2D}} \int_0^h \partial_3(v^3_{3D}v^2_{3D})\,\mathrm{d}x_3\,\mathrm{d}x_{12} \\
&\qquad - \int_{\Omega_{2D}} \varphi^1_{v_{2D}} \int_0^h v^1_{3D}\nabla\cdot v_{3D}\,\mathrm{d}x_3\,\mathrm{d}x_{12} \\
&\qquad - \int_{\Omega_{2D}} \varphi^2_{v_{2D}} \int_0^h v^2_{3D}\nabla\cdot v_{3D}\,\mathrm{d}x_3\,\mathrm{d}x_{12}.
\end{aligned}
$$

Since $(\nabla\cdot v_{3D},\varphi)_\Omega = 0$ and under the assumption that v^3_{3D} is small and can be neglected we have that

$$
\begin{aligned}
&((v_{3D}\cdot\nabla)v_{3D},\varphi_{v_{2D}})_\Omega \\
&\quad \approx \int_{\Omega_{2D}} \varphi^1_{v_{2D}}\partial_1 \int_0^h (v^1_{3D})^2\,\mathrm{d}x_3\,\mathrm{d}x_{12} + \int_{\Omega_{2D}} \varphi^1_{v_{2D}}\partial_2 \int_0^h v^2_{3D}v^1_{3D}\,\mathrm{d}x_3\,\mathrm{d}x_{12} \\
&\quad + \int_{\Omega_{2D}} \varphi^2_{v_{2D}}\partial_1 \int_0^h v^1_{3D}v^2_{3D}\,\mathrm{d}x_3\,\mathrm{d}x_{12} + \int_{\Omega_{2D}} \varphi^2_{v_{2D}}\partial_2 \int_0^h (v^2_{3D})^2\,\mathrm{d}x_3\,\mathrm{d}x_{12}.
\end{aligned}
$$

We need to approximate this further as

$$
\begin{aligned}
&((v_{3D}\cdot\nabla)v_{3D},\varphi_{v_{2D}})_\Omega \\
&\qquad\approx \int_{\Omega_{2D}} \varphi^1_{v_{2D}}\partial_1(h(v^1_{2D})^2)\,\mathrm{d}x_{12} + \int_{\Omega_{2D}} \varphi^1_{v_{2D}}\partial_2(hv^2_{2D}v^1_{2D})\,\mathrm{d}x_{12} \\
&\qquad+ \int_{\Omega_{2D}} \varphi^2_{v_{2D}}\partial_1(hv^1_{2D}v^2_{2D})\,\mathrm{d}x_{12} + \int_{\Omega_{2D}} \varphi^2_{v_{2D}}\partial_2(h(v^2_{2D})^2)\,\mathrm{d}x_{12}, \\
&\qquad= (\partial_1(h(v^1_{2D})^2),\varphi^1_{v_{2D}})_{\Omega_{2D}} + (\partial_2(hv^2_{2D}v^1_{2D}),\varphi^1_{v_{2D}})_{\Omega_{2D}} \\
&\qquad+ (\partial_1(hv^1_{2D}v^2_{2D}),\varphi^2_{v_{2D}})_{\Omega_{2D}} + (\partial_2(h(v^2_{2D})^2),\varphi^2_{v_{2D}})_{\Omega_{2D}}.
\end{aligned}
$$

Using the product rule we get

$$
\begin{aligned}
&((v_{3D}\cdot\nabla)v_{3D},\varphi_{v_{2D}})_\Omega \\
&= 2(v^1_{2D}\partial_1(hv^1_{2D}),\varphi^1_{v_{2D}})_{\Omega_{2D}} + 2(\partial_2(hv^2_{2D}),\varphi^2_{v_{2D}})_{\Omega_{2D}} + (v^2_{2D}\partial_2(hv^1_{2D}),\varphi^1_{v_{2D}})_{\Omega_{2D}} \\
&+ (v^1_{2D}\partial_2(hv^2_{2D}),\varphi^1_{v_{2D}})_{\Omega_{2D}} + (v^1_{2D}\partial_1(hv^2_{2D}),\varphi^2_{v_{2D}})_{\Omega_{2D}} + (v^2_{2D}\partial_1(hv^1_{2D}),\varphi^2_{v_{2D}})_{\Omega_{2D}}, \\
&= ((v_{2D}\cdot\nabla)(hv_{2D}),\varphi_{v_{2D}})_{\Omega_{2D}} + (\nabla\cdot(hv_{2D})v_{2D},\varphi_{v_{2D}})_{\Omega_{2D}}.
\end{aligned}
$$

The following three non-linear terms are rewritten as integrals in two dimensions in the usual manner.

$((v_{3D}\cdot\nabla)v_{2D},\varphi_{v_{2D}})_\Omega$:

$$
\begin{aligned}
((v_{3D}\cdot\nabla)v_{2D},\varphi_{v_{2D}})_\Omega &= \int_\Omega (v_{3D}\cdot\nabla)v_{2D}\varphi_{v_{2D}}\,\mathrm{d}x, \\
&= \int_\Omega v^1_{3D}\partial_1 v^1_{2D}\varphi^1_{v_{2D}}\,\mathrm{d}x + \int_\Omega \partial_2 v^1_{2D}\varphi^1_{v_{2D}}\,\mathrm{d}x + \int_\Omega \partial_1 v^1_{2D}\varphi^1_{v_{2D}}\,\mathrm{d}x, \\
&+ \int_\Omega \partial_1 v^2_{2D}\varphi^2_{v_{2D}}\,\mathrm{d}x + \int_\Omega \partial_2 v^2_{2D}\varphi^2_{v_{2D}}\,\mathrm{d}x + \int_\Omega \partial_1 v^2_{2D}\varphi^2_{v_{2D}}\,\mathrm{d}x, \\
&= \int_{\Omega_{2D}} \varphi^1_{v_{2D}}\partial_1 v^1_{2D}\int_0^h v^1_{3D}\,\mathrm{d}x_3\,\mathrm{d}x_{12} + \int_{\Omega_{2D}} \varphi^1_{v_{2D}}\partial_2 v^1_{2D}\int_0^h v^2_{3D}\,\mathrm{d}x_3\,\mathrm{d}x_{12} \\
&+ \int_{\Omega_{2D}} \varphi^1_{v_{2D}}\partial_1 v^1_{2D}\int_0^h v^3_{3D}\,\mathrm{d}x_3\,\mathrm{d}x_{12} + \int_{\Omega_{2D}} \varphi^2_{v_{2D}}\partial_1 v^2_{2D}\int_0^h v^1_{3D}\,\mathrm{d}x_3\,\mathrm{d}x_{12} \\
&+ \int_{\Omega_{2D}} \varphi^2_{v_{2D}}\partial_2 v^2_{2D}\int_0^h v^2_{3D}\,\mathrm{d}x_3\,\mathrm{d}x_{12} + \int_{\Omega_{2D}} \varphi^2_{v_{2D}}\partial_1 v^2_{2D}\int_0^h v^3_{3D}\,\mathrm{d}x_3\,\mathrm{d}x_{12}, \\
&= \int_{\Omega_{2D}} \varphi^1_{v_{2D}}\partial_1 v^1_{2D}(hv^1_{2D})\,\mathrm{d}x_{12} + \int_{\Omega_{2D}} \varphi^1_{v_{2D}}\partial_2 v^1_{2D}(hv^2_{2D})\,\mathrm{d}x_{12} \\
&+ \int_{\Omega_{2D}} \varphi^1_{v_{2D}}\partial_1 v^1_{2D}(hv^3_{2D})\,\mathrm{d}x_{12} + \int_{\Omega_{2D}} \varphi^2_{v_{2D}}\partial_1 v^2_{2D}(hv^1_{2D})\,\mathrm{d}x_{12} \\
&+ \int_{\Omega_{2D}} \varphi^2_{v_{2D}}\partial_2 v^2_{2D}(hv^2_{2D})\,\mathrm{d}x_{12} + \int_{\Omega_{2D}} \varphi^2_{v_{2D}}\partial_1 v^2_{2D}(hv^3_{2D})\,\mathrm{d}x_{12}, \\
&= (((hv_{2D})\cdot\nabla)v_{2D},\varphi_{v_{2D}})_{\Omega_{2D}}.
\end{aligned}
$$

$((v_{2D}\cdot\nabla)v_{3D},\varphi_{v_{2D}})_\Omega$:

$$
\begin{aligned}
((v_{2D}\cdot\nabla)v_{3D},\varphi_{v_{2D}})_\Omega &= \int_\Omega (v_{2D}\cdot\nabla)v_{3D}\varphi_{v_{2D}}\,\mathrm{d}x,\\
&= \int_\Omega v^1_{2D}\partial_1 v^1_{3D}\varphi^1_{v_{2D}}\,\mathrm{d}x + \int_\Omega v^2_{2D}\partial_2 v^1_{3D}\varphi^1_{v_{2D}}\,\mathrm{d}x + \int_\Omega v^3_{2D}\partial_1 v^1_{3D}\varphi^1_{v_{2D}}\,\mathrm{d}x\\
&+ \int_\Omega v^1_{2D}\partial_1 v^2_{3D}\varphi^2_{v_{2D}}\,\mathrm{d}x + \int_\Omega v^2_{2D}\partial_2 v^2_{3D}\varphi^2_{v_{2D}}\,\mathrm{d}x + \int_\Omega v^3_{2D}\partial_1 v^2_{3D}\varphi^2_{v_{2D}}\,\mathrm{d}x,\\
&= \int_{\Omega_{2D}} \varphi^1_{v_{2D}} v^1_{2D}\partial_1 \int_0^h v^1_{3D}\,\mathrm{d}x_3\,\mathrm{d}x_{12} + \int_{\Omega_{2D}} \varphi^1_{v_{2D}} v^2_{2D}\partial_2 \int_0^h v^1_{3D}\,\mathrm{d}x_3\,\mathrm{d}x_{12}\\
&+ \int_{\Omega_{2D}} \varphi^1_{v_{2D}} v^3_{2D}\partial_1 \int_0^h v^1_{3D}\,\mathrm{d}x_3\,\mathrm{d}x_{12} + \int_{\Omega_{2D}} \varphi^2_{v_{2D}} v^1_{2D}\partial_1 \int_0^h v^2_{3D}\,\mathrm{d}x_3\,\mathrm{d}x_{12}\\
&+ \int_{\Omega_{2D}} \varphi^2_{v_{2D}} v^2_{2D}\partial_2 \int_0^h v^2_{3D}\,\mathrm{d}x_3\,\mathrm{d}x_{12} + \int_{\Omega_{2D}} \varphi^2_{v_{2D}} v^3_{2D}\partial_1 \int_0^h v^2_{3D}\,\mathrm{d}x_3\,\mathrm{d}x_{12},\\
&= \int_{\Omega_{2D}} \varphi^1_{v_{2D}} v^1_{2D}\partial_1 (h v^1_{2D})\,\mathrm{d}x_{12} + \int_{\Omega_{2D}} \varphi^1_{v_{2D}} v^2_{2D}\partial_2 (h v^1_{2D})\,\mathrm{d}x_{12}\\
&+ \int_{\Omega_{2D}} \varphi^1_{v_{2D}} v^3_{2D}\partial_1 (h v^1_{2D})\,\mathrm{d}x_{12} + \int_{\Omega_{2D}} \varphi^2_{v_{2D}} v^1_{2D}\partial_1 (h v^2_{2D})\,\mathrm{d}x_{12}\\
&+ \int_{\Omega_{2D}} \varphi^2_{v_{2D}} v^2_{2D}\partial_2 (h v^2_{2D})\,\mathrm{d}x_{12} + \int_{\Omega_{2D}} \varphi^2_{v_{2D}} v^3_{2D}\partial_1 (h v^2_{2D})\,\mathrm{d}x_{12},\\
&= ((v_{2D}\cdot\nabla)(h v_{2D}),\varphi_{v_{2D}})_{\Omega_{2D}}.
\end{aligned}
$$

$((v_{2D}\cdot\nabla)v_{2D},\varphi_{v_{2D}})_\Omega$:

$$
\begin{aligned}
((v_{2D}\cdot\nabla)v_{2D},\varphi_{v_{2D}})_\Omega &= \int_\Omega (v_{2D}\cdot\nabla)v_{2D}\varphi_{v_{2D}}\,\mathrm{d}x,\\
&= \int_\Omega v^1_{2D}\partial_1 v^1_{2D}\varphi^1_{v_{2D}}\,\mathrm{d}x + \int_\Omega v^2_{2D}\partial_2 v^1_{2D}\varphi^1_{v_{2D}}\,\mathrm{d}x\\
&+ \int_\Omega v^1_{2D}\partial_1 v^2_{2D}\varphi^2_{v_{2D}}\,\mathrm{d}x + \int_\Omega v^2_{2D}\partial_2 v^2_{2D}\varphi^2_{v_{2D}}\,\mathrm{d}x,\\
&= \int_{\Omega_{2D}} h v^1_{2D}\partial_1 v^1_{2D}\varphi^1_{v_{2D}}\,\mathrm{d}x_{12} + \int_{\Omega_{2D}} h v^2_{2D}\partial_2 v^1_{2D}\varphi^1_{v_{2D}}\,\mathrm{d}x_{12}\\
&+ \int_{\Omega_{2D}} h v^1_{2D}\partial_1 v^2_{2D}\varphi^2_{v_{2D}}\,\mathrm{d}x_{12} + \int_{\Omega_{2D}} h v^2_{2D}\partial_2 v^2_{2D}\varphi^2_{v_{2D}}\,\mathrm{d}x_{12},\\
&= (h(v_{2D}\cdot\nabla)v_{2D},\varphi_{v_{2D}})_{\Omega_{2D}}.
\end{aligned}
$$

$((v_{3D}+v_{2D})\cdot\nabla(T_{3D}+T_{2D}),\varphi_{T_{2D}})_\Omega$:

For the coupling terms in the temperature equation we approach the same way as for the velocity equation and write the coupling as a sum of four terms which are rewritten separately.

$$
\begin{aligned}
((v_{3D}+v_{2D})\cdot\nabla(T_{3D}+T_{2D}),\varphi_{T_{2D}})_\Omega &= (v_{3D}\cdot\nabla T_{3D},\varphi_{T_{2D}})_\Omega\\
+(v_{3D}\cdot\nabla T_{2D},\varphi_{T_{2D}})_\Omega &+ (v_{2D}\cdot\nabla T_{3D},\varphi_{T_{2D}})_\Omega + (v_{2D}\cdot\nabla T_{2D},\varphi_{T_{2D}})_\Omega.
\end{aligned}
$$

$(v_{3D}\cdot\nabla T_{3D}, \varphi_{T_{2D}})_\Omega$:

$$
\begin{aligned}
(v_{3D}\cdot\nabla T_{3D}, \varphi_{T_{2D}})_\Omega &= \int_\Omega v_{3D}\cdot\nabla T_{3D}\varphi_{T_{2D}}\,\mathrm{d}x,\\
&= \int_\Omega v^1_{3D}\partial_1 T_{3D}\varphi_{T_{2D}}\,\mathrm{d}x + \int_\Omega v^2_{3D}\partial_2 T_{3D}\varphi_{T_{2D}}\,\mathrm{d}x + \int_\Omega v^3_{3D}\partial_3 T_{3D}\varphi_{T_{2D}}\,\mathrm{d}x,\\
&= \int_{\Omega_{2D}} \varphi_{T_{2D}} \int_0^h \partial_1(v^1_{3D}T_{3D})\,\mathrm{d}x_3\,\mathrm{d}x_{12} + \int_{\Omega_{2D}} \varphi_{T_{2D}} \int_0^h \partial_2(v^2_{3D}T_{3D})\,\mathrm{d}x_3\,\mathrm{d}x_{12}\\
&+ \int_{\Omega_{2D}} \varphi_{T_{2D}} \int_0^h \partial_3(v^3_{3D}T_{3D})\,\mathrm{d}x_3\,\mathrm{d}x_{12},\\
&= \int_{\Omega_{2D}} \varphi_{T_{2D}}\partial_1 \int_0^h (v^1_{3D}T_{3D})\,\mathrm{d}x_3\,\mathrm{d}x_{12} + \int_{\Omega_{2D}} \varphi_{T_{2D}}\partial_2 \int_0^h (v^2_{3D}T_{3D})\,\mathrm{d}x_3\,\mathrm{d}x_{12}\\
&+ \int_{\Omega_{2D}} \varphi_{T_{2D}}\partial_3 \int_0^h (v^3_{3D}T_{3D})\,\mathrm{d}x_3\,\mathrm{d}x_{12},
\end{aligned}
$$

Here, we need to approximate this as

$$
\begin{aligned}
(v_{3D}\cdot\nabla T_{3D}, \varphi_{T_{2D}})_\Omega &\approx \int_{\Omega_{2D}} \varphi_{T_{2D}}\partial_1(hv^1_{2D}T_{2D})\,\mathrm{d}x_{12} + \int_{\Omega_{2D}} \varphi_{T_{2D}}\partial_2(hv^2_{2D}T_{2D})\,\mathrm{d}x_{12},\\
&= \int_{\Omega_{2D}} \varphi_{T_{2D}}\{v^1_{2D}\partial_1(hT_{2D}) + T_{2D}\partial_1(hv^1_{2D})\}\,\mathrm{d}x_{12}\\
&+ \int_{\Omega_{2D}} \varphi_{T_{2D}}\{v^2_{2D}\partial_2(hT_{2D}) + T_{2D}\partial_2(hv^2_{2D})\}\,\mathrm{d}x_{12},\\
&= (v_{2D}\cdot\nabla(hT_{2D}), \varphi_{T_{2D}})_{\Omega_{2D}} + (T_{2D}\nabla\cdot(hv_{2D}), \varphi_{T_{2D}})_{\Omega_{2D}}.
\end{aligned}
$$

$(v_{3D}\cdot\nabla T_{2D}, \varphi_{T_{2D}})_\Omega$:

$$
\begin{aligned}
(v_{3D}\cdot\nabla T_{2D}, \varphi_{T_{2D}})_\Omega &= \int_\Omega v_{3D}\cdot\nabla T_{2D}\varphi_{T_{2D}}\,\mathrm{d}x,\\
&= \int_\Omega v^1_{3D}\partial_1 T_{2D}\varphi_{T_{2D}}\,\mathrm{d}x + \int_\Omega v^2_{3D}\partial_2 T_{2D}\varphi_{T_{2D}}\,\mathrm{d}x,\\
&= \int_{\Omega_{2D}} \varphi_{T_{2D}}\partial_1 T_{2D} \int_0^h v^1_{3D}\,\mathrm{d}x_3\,\mathrm{d}x_{12} + \int_{\Omega_{2D}} \varphi_{T_{2D}}\partial_2 T_{2D} \int_0^h v^2_{3D}\,\mathrm{d}x_3\,\mathrm{d}x_{12},\\
&= \int_{\Omega_{2D}} \varphi_{T_{2D}}\partial_1 T_{2D}(hv^1_{2D})\,\mathrm{d}x_{12} + \int_{\Omega_{2D}} \varphi_{T_{2D}}\partial_2 T_{2D}(hv^2_{2D})\,\mathrm{d}x_{12},\\
&= ((hv_{2D})\cdot\nabla T_{2D}, \varphi_{T_{2D}})_{\Omega_{2D}}.
\end{aligned}
$$

$(v_{2D}\cdot\nabla T_{3D}, \varphi_{T_{2D}})_\Omega$:

$$
\begin{aligned}
(v_{2D}\cdot\nabla T_{3D}, \varphi_{T_{2D}}T)_\Omega &= \int_\Omega v_{2D}\cdot\nabla T_{3D}\varphi_{T_{2D}}\,\mathrm{d}x,\\
&= \int_\Omega v^1_{2D}\partial_1 T_{3D}\varphi_{T_{2D}}\,\mathrm{d}x + \int_\Omega v^2_{2D}\partial_2 T_{3D}\varphi_{T_{2D}}\,\mathrm{d}x + \int_\Omega v^3_{2D}\partial_3 T_{3D}\varphi_{T_{2D}}\,\mathrm{d}x,\\
&= \int_{\Omega_{2D}} \varphi_{T_{2D}} v^1_{2D}\partial_1 \int_0^h T_{3D}\,\mathrm{d}x_3\,\mathrm{d}x_{12}\\
&+ \int_{\Omega_{2D}} \varphi_{T_{2D}} v^2_{2D}\partial_2 \int_0^h T_{3D}\,\mathrm{d}x_3\,\mathrm{d}x_{12}\\
&+ \int_{\Omega_{2D}} \varphi_{T_{2D}} v^3_{2D}\partial_3 \int_0^h T_{3D}\,\mathrm{d}x_3\,\mathrm{d}x_{12},\\
&= \int_{\Omega_{2D}} \varphi_{T_{2D}} v^1_{2D}\partial_1 (hT_{2D})\,\mathrm{d}x_{12} + \int_{\Omega_{2D}} \varphi_{T_{2D}} v^2_{2D}\partial_2 (hT_{2D})\,\mathrm{d}x_{12}\\
&+ \int_{\Omega_{2D}} \varphi_{T_{2D}} v^3_{2D}\partial_3 (hT_{2D})\,\mathrm{d}x_{12},\\
&= ((v_{2D}\cdot\nabla)(hT_{2D}), \varphi_{T_{2D}})_{\Omega_{2D}}.
\end{aligned}
$$

$(v_{2D}\cdot\nabla T_{2D}, \varphi_{T_{2D}})_\Omega$:

$$
\begin{aligned}
(v_{2D}\cdot\nabla T_{2D}, \varphi_{T_{2D}})_\Omega &= \int_\Omega v_{2D}\cdot\nabla T_{2D}\varphi_{T_{2D}}\,\mathrm{d}x,\\
&= \int_\Omega v^1_{2D}\partial_1 T_{2D}\varphi_{T_{2D}}\,\mathrm{d}x + \int_\Omega v^2_{2D}\partial_2 T_{2D}\varphi_{T_{2D}}\,\mathrm{d}x,\\
&= \int_{\Omega_{2D}} h v^1_{2D}\partial_1 T_{2D}\varphi_{T_{2D}}\,\mathrm{d}x_{12} + \int_{\Omega_{2D}} h v^2_{2D}\partial_2 T_{2D}\varphi_{T_{2D}}\,\mathrm{d}x_{12},\\
&= (hv_{2D}\cdot\nabla T_{2D}, \varphi_{T_{2D}})_{\Omega_{2D}}.
\end{aligned}
$$

To this end, the coupling terms were rewritten as integrals in two dimensions.

Summing up the results

The second equation

$$a(u_{2D} + u_{3D})(\varphi_{2D}) = (F, \varphi_{2D}) \text{ for all } \varphi_{2D} \in V_{2D}$$

is considered. We collect the results we gained in the previous paragraphs. First of all we sum up the terms tested with $\varphi_{v_{2D}}$:

$$\begin{aligned}
a(u_{2D}+u_{3D})(\varphi_{v_{2D}}) &= (\partial_t(v_{3D}+v_{2D}),\varphi_{v_{2D}})_\Omega + (((v_{3D}+v_{2D})\cdot\nabla)(v_{3D}+v_{2D}),\varphi_{v_{2D}})_\Omega \\
&+ (\gamma\nabla(p_{3D}+p_{2D}),\varphi_{v_{2D}})_\Omega + (\nu\nabla(v_{3D}+v_{2D}),\nabla\varphi_{v_{2D}})_\Omega \\
&+ (\varrho\gamma\boldsymbol{g}(T_{3D}+T_{2D}),\varphi_{v_{2D}})_\Omega + (g\nabla h,\varphi_{v_{2D}})_\Omega, \\
&= (\partial_t(hv_{2D}),\varphi_{v_{2D}})_{\Omega_{2D}} + (h\partial_t v_{2D},\varphi_{v_{2D}})_{\Omega_{2D}} + (\nu\nabla(hv_{2D}),\nabla\varphi_{v_{2D}})_{\Omega_{2D}} \\
&+ (h\nu\nabla v_{2D},\nabla\varphi_{v_{2D}})_{\Omega_{2D}} + 2((v_{2D}\cdot\nabla)(hv_{2D}),\varphi_{v_{2D}})_{\Omega_{2D}} \\
&+ (\nabla\cdot(hv_{2D})v_{2D},\varphi_{v_{2D}})_{\Omega_{2D}} + 2(h(v_{2D}\cdot\nabla)v_{2D},\varphi_{v_{2D}})_{\Omega_{2D}} \\
&+ (\varrho\gamma\boldsymbol{g}hT_{2D},\varphi_{v_{2D}})_{\Omega_{2D}} + 2(\gamma\varrho gh\nabla h,\varphi_{v_{2D}})_{\Omega_{2D}}.
\end{aligned} \tag{4.17}$$

The terms tested with $\varphi_{p_{2D}}$ sum up to

$$\begin{aligned}
a(u_{2D}+u_{3D})(\varphi_{p_{2D}}) &= (\nabla\cdot(v_{3D}+v_{2D}),\varphi_{p_{2D}})_\Omega, \\
&= (\partial_t h+\nabla\cdot(hv_{2D}),\varphi_{p_{2D}})_{\Omega_{2D}} + (h\nabla\cdot v_{2D},\varphi_{p_{2D}})_{\Omega_{2D}}, \\
&=: a(u_{2D})(\varphi_{p_{2D}}).
\end{aligned} \tag{4.18}$$

These equations are closely related to the Shallow-Water equations introduced in Section 3.1.4. Collecting the results for the temperature equation, this leads to

$$\begin{aligned}
a(u_{2D}+u_{3D})(\varphi_{T_{2D}}) &= (\partial_t(T_{3D}+T_{2D}),\varphi_T)_\Omega + ((v_{3D}+v_{2D})\cdot\nabla(T_{3D}+T_{2D}),\varphi_T)_\Omega \\
&+ (\nu_T\nabla(T_{3D}+T_{2D}),\nabla\varphi_T)_\Omega, \\
&= (\partial_t(hT_{2D}),\varphi_{T_{2D}})_{\Omega_{2D}} + (h\partial_t T_{2D},\varphi_{T_{2D}})_{\Omega_{2D}} + (\nu_T\nabla(hT_{2D}),\nabla\varphi_{T_{2D}})_{\Omega_{2D}} \\
&+ (\nu_T h\nabla T_{2D},\nabla\varphi_{T_{2D}})_{\Omega_{2D}} + 2(v_{2D}\cdot\nabla(hT_{2D}),\varphi_{T_{2D}})_{\Omega_{2D}} \\
&+ (T_{2D}\nabla\cdot(hv_{2D}),\varphi_{T_{2D}})_{\Omega_{2D}} + 2(hv_{2D}\cdot\nabla T_{2D},\varphi_{T_{2D}})_{\Omega_{2D}}.
\end{aligned} \tag{4.19}$$

Taking into account that

$$a(u_{2D}+u_{3D})(\varphi_{p_{2D}}) = 0$$

holds we can simplify (4.17) to

$$\begin{aligned}
a(u_{2D}+u_{3D})(\varphi_{v_{2D}}) &= 2(\partial_t(hv_{2D}),\varphi_{v_{2D}})_{\Omega_{2D}} + 2(h\nu\nabla v_{2D},\nabla\varphi_{v_{2D}})_{\Omega_{2D}} + (v_{2D}\nabla h,\nabla\varphi_{v_{2D}})_{\Omega_{2D}} \\
&+ 2((v_{2D}\cdot\nabla)(hv_{2D}),\varphi_{v_{2D}})_{\Omega_{2D}} + 2(\nabla\cdot(hv_{2D})v_{2D},\varphi_{v_{2D}})_{\Omega_{2D}} \\
&+ 2(h(v_{2D}\cdot\nabla)v_{2D},\varphi_{v_{2D}})_{\Omega_{2D}} + 2(\gamma\varrho gh\nabla h,\varphi_{v_{2D}})_{\Omega_{2D}}, \\
&=: a(u_{2D})(\varphi_{v_{2D}}).
\end{aligned} \tag{4.20}$$

and (4.19) becomes

$$\begin{aligned}
a(u_{2D}+u_{3D})(\varphi_{T_{2D}}) &= 2(\partial_t(hT_{2D}),\varphi_{T_{2D}})_{2D} + 2(\nu_T h\nabla T_{2D},\nabla\varphi_{T_{2D}})_{2D} \\
&+ (\nu_T T_{2D}\nabla h,\nabla\varphi_{T_{2D}})_{2D} + 2(v_{2D}\cdot\nabla(hT_{2D}),\varphi_{T_{2D}})_{2D} \\
&+ 2(T_{2D}\nabla\cdot(hv_{2D}),\varphi_{T_{2D}})_{2D} + 2(hv_{2D}\cdot\nabla T_{2D},\varphi_{T_{2D}})_{2D}, \\
&+ (T_{2D}h\nabla\cdot v_{2D},\varphi_{T_{2D}})_{2D} \\
&=: a(u_{2D})(\varphi_{T_{2D}}).
\end{aligned} \tag{4.21}$$

This shows that the part of the semi-linear form which is tested by a function in V_{2D} no longer consists of couplings with functions in V_{3D} and the integrals are two dimensional integrals:

$$\begin{aligned} a(u_{2D} + u_{3D})(\varphi_{v_{2D}}) &= a(u_{2D})(\varphi_{v_{2D}}), \\ a(u_{2D} + u_{3D})(\varphi_{p_{2D}}) &= a(u_{2D})(\varphi_{p_{2D}}), \\ a(u_{2D} + u_{3D})(\varphi_{T_{2D}}) &= a(u_{2D})(\varphi_{T_{2D}}). \end{aligned}$$

The definition of these semi-linear forms is given in (4.18), (4.20), and (4.21). From this follows that for the second equation it holds

$$a(u_{2D} + u_{3D})(\varphi_{2D}) = a(u_{2D})(\varphi_{2D}).$$

The fully coupled problem reads: Find $u_{2D} = (v_{2D}, h, T_{2D}) \in X_{2D}$ and $u_{3D} = (v_{3D}, p_{3D}, T_{3D}) \in X_{3D}$ such that

$$\begin{aligned} a(u_{2D} + u_{3D})(\varphi_{3D}) &= (F, \varphi_{3D}) \text{ for all } \varphi_{3D} \in X_{3D}, \\ a(u_{2D})(\varphi_{2D}) &= (F, \varphi_{2D}) \text{ for all } \varphi_{2D} \in X_{2D}. \end{aligned}$$

For the simulations which are described in Chapter 7, we further approximated the two dimensional equation since we used the original Shallow-Water equations given in (3.16).

5 Discretization

The discretization of the equations presented in Chapter 3 will be laid out in this chapter. The finite element method will be used to discretize the equations in space. For temporal discretization we discuss different methods that may be taken. Due to the coupling of 2D and 3D equations in the problem we will focus on the combination of discretization in 2D with equations discretized in 3D. In Chapter 6 details of this discretization in view of algorithms will be discussed.

5.1 Discretization in time

This section is devoted to the discretization of the Navier-Stokes equations as well as the Shallow-Water equations. Since we introduced the ALE formulation in Section 3.2.4 we can restrict ourselves to a fixed domain for all times.

5.1.1 Preliminaries on time stepping schemes

We divide the desired time interval $\bar{I} = [0, T]$ into subintervals such that

$$0 = t_0 < t_1 < \cdots < t_N = T, \quad I_n := (t_{n-1}, t_n], \quad k_n := t_n - t_{n-1},$$

and

$$\bar{I} = \{0\} \cup I_1 \cup I_2 \cup \cdots \cup I_n \cup \cdots \cup I_N.$$

A sketch of such a discretization is given in Figure 5.1. Note that the step size k_n does not necessarily have to be the same on each subinterval.

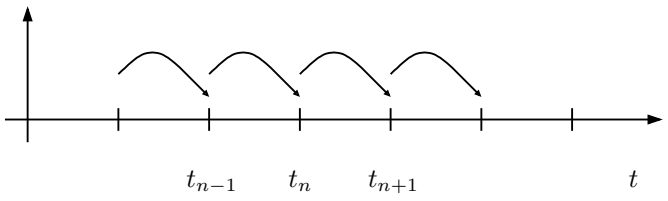


Figure 5.1. Discretization in time from t_{n-1} to t_{n+1}.

Inspired by the work in [55], we analyzed the properties of various time stepping schemes. Based on the following linear test case

$$\partial_t x - Ax = 0, \qquad x(0) = x^0,$$

where A a positive definite and (symmetric) $n \times n$ matrix independent of t, we investigate the behavior as $t \to \infty$ for equidistant step size $k_n = k$. The solution of the system is given as

$$x(t) = e^{At}x^0, \quad x^i(t) = e^{\lambda_i t}x^{0i}.$$

The conclusions can be carried over to an operator which is non-selfadjoint, non-autonomous and weakly non-linear, e.g. the operator related to the Navier-Stokes problem.

Depending on the parameters λ_i the behavior of the solution $x(t)$ for $t \to \infty$ is characterized by the amplification factor $\omega = \omega(z)$. For one-step schemes we have $x^{ni} = \omega(\lambda_i k)^n x^{0i}$, with $x^n = x(t_n)$ being the sequence of values generated by the application of a time stepping scheme.

One-step θ-schemes

With the decomposition

$$A = A_1 + A_2, \qquad A_1 = \theta A \text{ and } A_2 = (1-\theta)A,$$

we derive one-step θ-schemes via

$$\frac{1}{k_n}(x^{n+1} - x^n) - \theta A^{n+1} - (1-\theta)A^n = 0,$$

where $A^i = Ax^i$. For the generated sequence and the amplification factor we achieve

$$x^{ni} = \left(\frac{1 + (1-\theta)k\lambda_i}{1 - \theta k\lambda_i}\right)^n x^{0i}, \quad \omega(z) = \frac{1 + (1-\theta)z}{1 - \theta z}.$$

For the explicit Euler scheme ($\theta = 0$) we have a restriction on the step size $k \leq \frac{1}{\lambda}$ to get stability. Furthermore it strongly amplifies free oscillations since $|\omega(ik)| > 1$. In the case $\theta = 1$ we have a strongly A-stable scheme because of the fact that $\omega(z) \to 0$ as $\operatorname{Re}\lambda \to \infty$. The implicit Euler scheme damps out free oscillations as $|\omega(ik)| < 1$. The implicit and explicit Euler scheme are of first order only.

The choice $\theta = \frac{1}{2}$ yields the second order accurate Crank-Nicolson scheme, but it has only weak damping properties, since $|\omega(z)| \to 1$ as $\operatorname{Re} z \to \infty$. However, free oscillations are well preserved due to the fact that $|\omega(ik)| = 1$.

Fractional-step θ-schemes

For the fractional-step θ-scheme a time step $t_{n-1} \to t_n$ is divided into three sub-steps, see also Figure 5.2:

$$t_{n-1} \to t_{n-1+\theta} \to t_{n-\theta} \to t_n.$$

With the decomposition $A = A_1 + A_2$ the fractional-step θ-scheme reads as follows

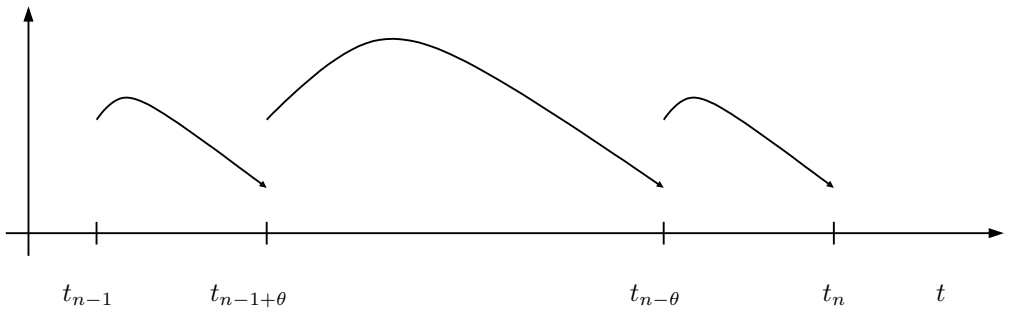


Figure 5.2. Discretization via fractional steps from t_{n-1} to t_n.

$$\begin{aligned} \frac{1}{\theta k_n}(x^{n+\theta} - x^n) - A_1^{n+\theta} - A_2^n &= 0, \\ \frac{1}{\theta' k_n}(x^{n+1-\theta} - x^{n+\theta}) - A_1^{n+\theta} - A_2^{n+1-\theta} &= 0, \\ \frac{1}{\theta k_n}(x^{n+1} - x^{n+1-\theta}) - A_1^{n+1} - A_2^{n+1-\theta} &= 0, \end{aligned} \tag{5.1}$$

where $\theta \in (0, \frac{1}{2}),\ \theta' = 1 - 2\theta$. We take $A_1 = \alpha A,\ A_2 = \beta A$ with $\alpha + \beta = 1,\ 0 < \alpha,\ \beta < 1$ to derive the fractional-step θ-scheme. Now it remains to choose θ, α, and β. For the generated sequence with constant step-size k and the amplification factor we derive

$$x^{ni} = \frac{(1 + \beta\theta k\lambda_i)^{2n}(1 + \alpha\theta' k\lambda_i)^n}{(1 - \alpha\theta k_n\lambda_i)^{2n}(1 - \beta\theta' k_n\lambda_i)^n} x^{0i}, \quad \omega(z) = \frac{(1 + \beta\theta z)^2(1 + \alpha\theta' z)}{(1 - \alpha\theta z)^2(1 - \beta\theta' z)}.$$

The rational function $\omega(\cdot)$ in the neighborhood of $z = 0$ can be expressed as

$$\omega(z) = 1 + z + \frac{z^2}{2}[1 + (\beta - \alpha)(2\theta^2 - 4\theta + 1)] + O(z^3).$$

A comparison to the exponential function yields

$$\alpha = \beta\ (= \frac{1}{2} \text{ from } \alpha + \beta = 1) \quad \text{or} \quad \theta = 1 - 1/\sqrt{2},$$

as necessary conditions for second order accuracy. Due to the fact that

$$\limsup_{\mathrm{Re}\, z \to \infty} |\omega(z)| = \limsup_{\mathrm{Re}\, z \to \infty} \left| \frac{(1 + \beta\theta z)^2(1 + \alpha\theta' z)}{(1 - \alpha\theta z)^2(1 - \beta\theta' z)} \right| = \frac{\beta}{\alpha},$$

we have to choose $\alpha > \beta$ since this is necessary for strong A-stability. If we set $\alpha\theta = \beta\theta' = \beta(1 - 2\theta)$ we get the same matrix in each partial step. This leads to the choice

$$\alpha = \frac{1 - 2\theta}{1 - \theta}, \quad \beta = \frac{\theta}{1 - \theta}, \quad \theta = 1 - 1/\sqrt{2}.$$

We have $|\omega(ik)| < 1$ and $|\omega| \approx 0.99987$ for $k = 0.8$.

Another variant of the fractional-step θ-scheme can be derived if we take $A_1 = A$ and $A_2 = 0$ in (5.1). This leads to

$$\frac{1}{k_n}(x^{n+\theta} - x^n) - \theta A^{n+\theta} = 0,$$
$$\frac{1}{k_n}(x^{n+1} - x^n) - \theta A^{n+1} - (1-\theta)A^{n+\theta} = 0,$$

where $\theta' = 1 - 2\theta$. This scheme has been analyzed for incompressible flow simulations in [67]. For the generated sequence with constant step-size k and the amplification factor we derive

$$x^{ni} = \frac{(1+\theta' k\lambda_i)^n}{(1-\theta k\lambda_i)^{2n}} x^{0i}, \quad \omega(z) = \frac{1+\theta' z}{(1-\theta z)^2}.$$

Again we compare the rational function $\omega(\cdot)$ in the neighborhood of $z = 0$

$$\omega(z) = 1 + z + \frac{z^2}{2}(4\theta - 2\theta^2) + \frac{z^3}{6}(6\theta^2(3-2\theta)) + O(z^4)$$

to the exponential function. A necessary condition for second order accuracy leads to the choice

$$\theta = 1 - 1/\sqrt{2}.$$

It holds $\limsup_{\mathrm{Re}\, z\to\infty} |\omega(z)| = 0$ as well as $|\omega(ik)| < 1$ and $|\omega| \approx 0.9986$ for $k = 0.8$.

To visualize the damping properties of the introduced schemes we consider the following simple test problem

$$x : [0,T] \to \mathbb{R}^4, \qquad x'(t) + Ax(t) = 0, \qquad x(0) = 1.$$

The non-diagonal 4×4-matrix is constructed to have the eigenvalues

$$\lambda_1 = 10^\gamma, \qquad \lambda_2 = i, \qquad \lambda_3 = -i, \qquad \lambda_4 = 1.$$

Thus, the solution contains a *stiff* component for $\gamma \gg 1$ (rapid exponential decay) and periodic components (free oscillations). Given the matrix A as

$$A = \begin{pmatrix} 0 & 10^\gamma & -1 & 0 \\ 0 & 10^\gamma & 0 & 0 \\ 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix},$$

the exact solution is known to be

$$x(t) = \begin{pmatrix} e^{-10^\gamma t} + \sin(t) \\ e^{-10^\gamma t} \\ \cos(t) \\ e^{-t} \end{pmatrix}.$$

Numerical results for free oscillations

In the following, we show the damping behavior by showing the computed solution $x_3(t)$ and the error $e_3(t)$ between the exact and the computed solution. To have a fair comparison between the one-step and the fractional-step schemes we choose the step size accordingly. For the Crank-Nicolson scheme shown in Figure 5.3 the free oscillations are well preserved and the error increases only slightly towards the end time point. Nearly the same behavior is exhibited in Figure 5.4. We note that the error at the end is larger than in the solution computed by

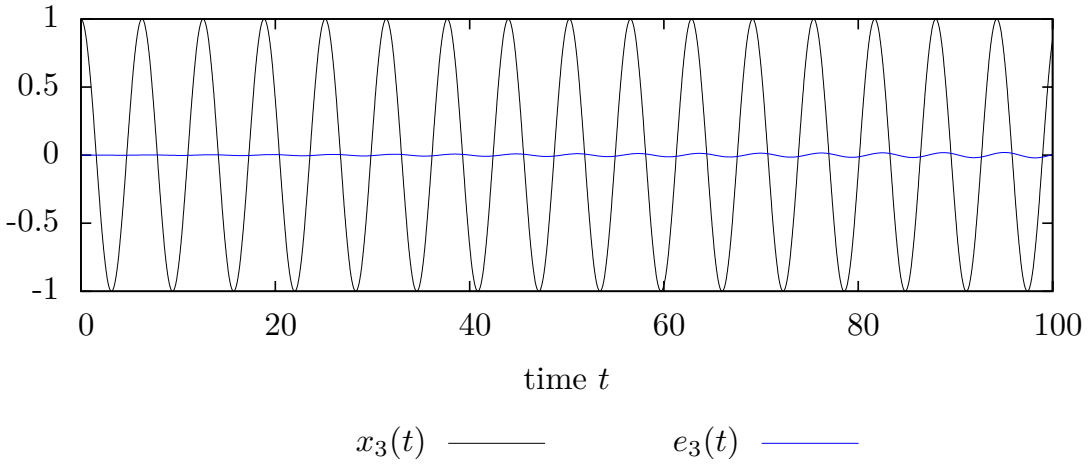


Figure 5.3. Crank-Nicolson with $k_n = 0.042$

the Crank-Nicolson scheme. With a fair choice of the step size in the variant of the fractional

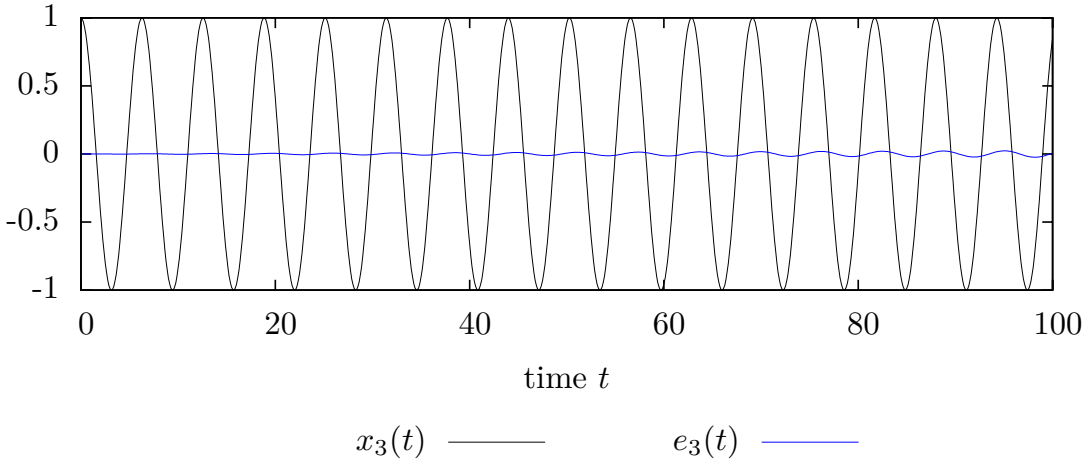


Figure 5.4. Fractional step θ with $k_n = 0.125$

step θ-scheme, we get a behavior as in the case for the original fractional step θ-scheme. This is shown in Figure 5.5.

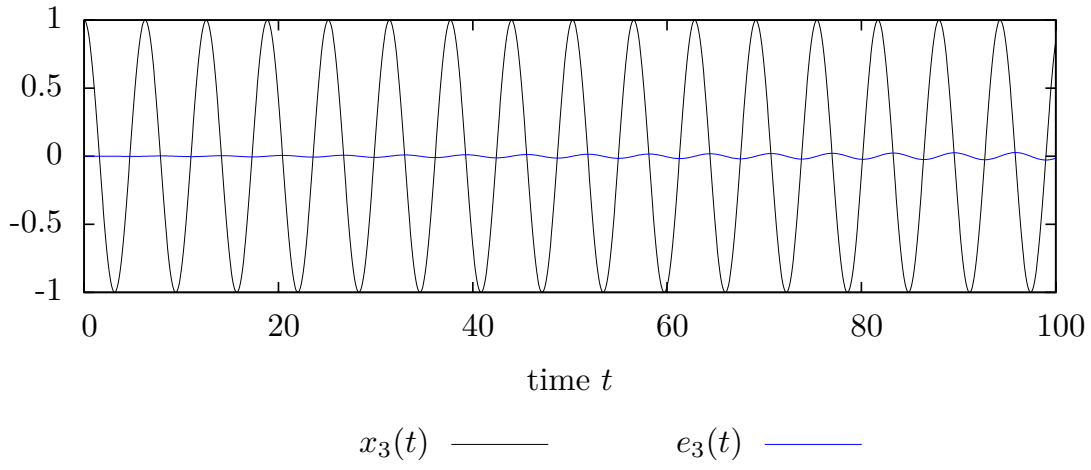


Figure 5.5. Fractional step variant with $k_n = 0.083$

Numerical results for exponential decay

As a next crucial point, we investigate the exponential decay in the first solution component. Again we present results for the computed solution $x_1(t)$ as well as the error $e_1(t)$. In Figure 5.6, we demonstrate the weak damping property of the Crank-Nicolson scheme at the beginning, whereas at the end, the free oscillations are captured again. For the fractional step-θ scheme, Figure 5.7, we also observe an error at the beginning. The variant of the fractional step-θ scheme damps out the exponential decay at the beginning very well. This is shown in Figure 5.8.
This clearly shows that we favor to use one of the fractional step θ-schemes for the numerical simulation in the case of fluid flows.

5.1.2 Temporal regularity

For discretization with finite differences we have to make sure that the regularity is sufficiently high. We are concerned with discretization in time of the space

$$Z := \Big\{ u = (v, p, T) \;\Big|\; v \in L^2(I, H_0^1(\Omega)^3), \partial_t v \in L^2(I, L^2(\Omega)^3), \\ p \in L^2(I, L^2(\Omega)/\mathbb{R}), T \in L^2(I, H_0^1(\Omega)), \partial_t T \in L^2(I, L^2(\Omega)) \Big\}.$$

For convenience of the reader we recall the definition of the test space

$$\widetilde{Z} := \Big\{ u = (v, p, T) \;\Big|\; v \in H_0^1(\Omega)^3, p \in L^2(\Omega)/\mathbb{R}, T \in H_0^1(\Omega) \Big\}. \tag{5.2}$$

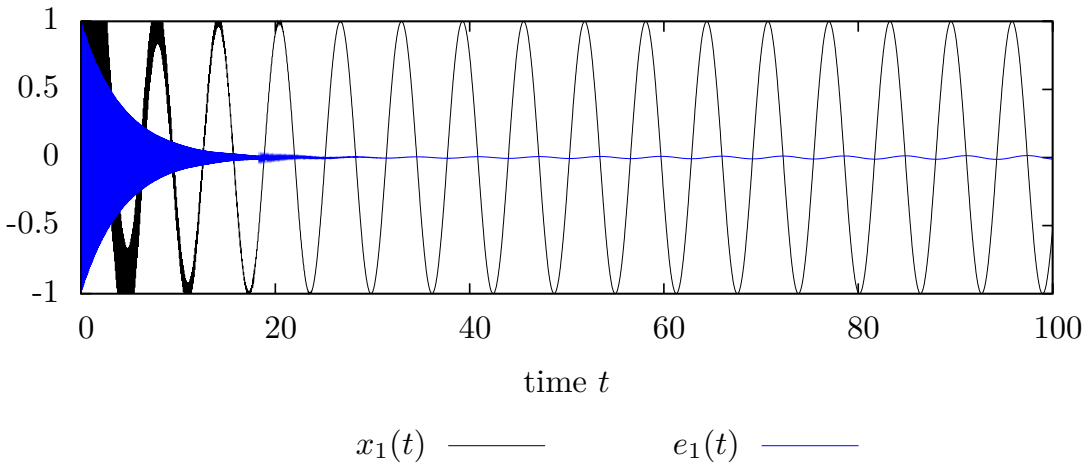


Figure 5.6. Crank-Nicolson with $k_n = 0.042$

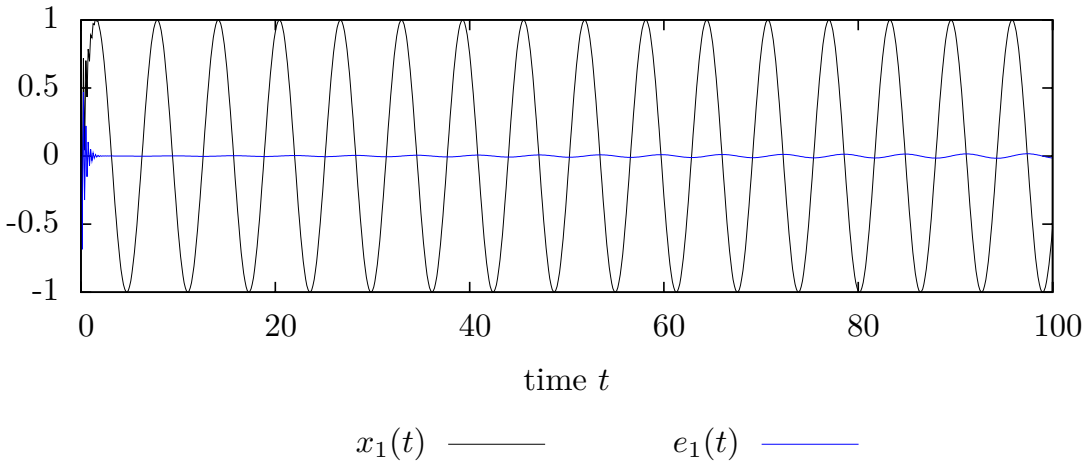


Figure 5.7. Fractional step θ with $k_n = 0.125$

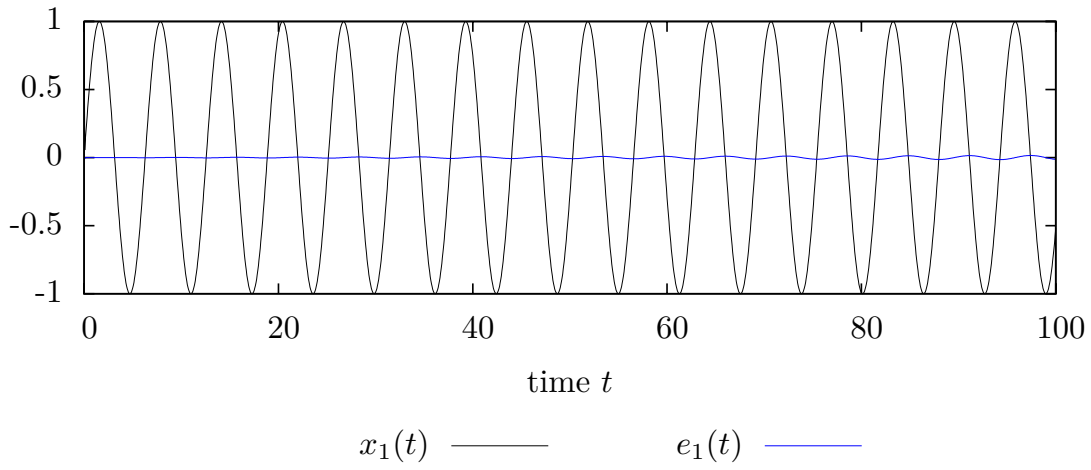


Figure 5.8. Fractional step variant with $k_n = 0.083$

Assuming the data admits a higher regularity of the solution, the following proposition ensures that we are allowed to take point values and point values of gradients in time:

Proposition 5.1. *For $w \in L^2(I, H_0^1(\Omega))$ with $\partial_t w \in L^2(I, H^{-1}(\Omega))$ it holds*

$$w \in C(\bar{I}, L^2(\Omega)).$$

Furthermore for $w \in L^2(I, H^2(\Omega) \cap H_0^1(\Omega))$ with $\partial_t w \in L^2(I, L^2(\Omega))$ it holds

$$w \in C(\bar{I}, H_0^1(\Omega)).$$

Proof. The proof can be found in [28]. □

5.1.3 Temporal discretization of the Boussinesq equations

First of all we discretize the non-linear equations derived in Section 3.2.4 resulting from the Navier-Stokes equations. For time discretization we apply either a one step- or a fractional-step θ-scheme. Based on the variational formulation (3.11) we introduce a semi-linear form $A\colon Z \times \widetilde{Z} \to \mathbb{R}$. This semi-linear form is defined as

$$\begin{aligned} A(u)(\varphi) :=& (\partial_t v, \varphi_v) + ((v\cdot\nabla)v, \varphi_v) + \gamma(\nabla p, \varphi_v) \\ &+ (\nu\nabla v, \nabla\varphi_v) - \gamma(\boldsymbol{g}\varrho T, \varphi_v) + (\nabla\cdot v, \varphi_p) \\ &+ (\partial_t T, \varphi_T) + (v\cdot\nabla T, \varphi_T) + \nu_T(\nabla T, \nabla\varphi_T). \end{aligned}$$

For discretization in time we split this into a sum of semi-linear forms and introduce a discrete form $A_t\colon \widetilde{Z} \times \widetilde{Z} \to \mathbb{R}$ as

$$\begin{aligned} A_t(u)(\varphi) &= a_t(u,\varphi) + a(u)(\varphi) + b(u,\varphi) + c(u,\varphi) \\ a_t(u,\varphi) &:= (v,\varphi_v) + (T,\varphi_T) \\ a(u)(\varphi) &:= ((v\cdot\nabla)v,\varphi_v) + (\nu\nabla v,\nabla\varphi_v) - \gamma(\boldsymbol{g}\varrho T,\varphi_v) + (v\cdot\nabla T,\varphi_T) + \nu_T(\nabla T,\nabla\varphi_T) \\ b(u,\varphi) &:= \gamma(\nabla p,\varphi_v) \\ c(u,\varphi) &:= (\nabla\cdot v,\varphi_p) \end{aligned} \tag{5.3}$$

with $a_t\colon \widetilde{Z}\times\widetilde{Z}\to\mathbb{R}$, $a\colon \widetilde{Z}\times\widetilde{Z}\to\mathbb{R}$, $b\colon \widetilde{Z}\times\widetilde{Z}\to\mathbb{R}$ and $c\colon \widetilde{Z}\times\widetilde{Z}\to\mathbb{R}$. In this case the form $a_t(\cdot,\cdot)$, $b(\cdot,\cdot)$, and $c(\cdot,\cdot)$ are linear in both arguments.

Given the previous solution $u^{k-1} = (v^{k-1}, p^{k-1}, T^{k-1})$ we are concerned with solving the following problem in each sub-step: Find $u = (v,p,T) := u^k = (v^k,p^k,T^k) \in \widetilde{Z}$ such that

$$\begin{aligned} &\frac{1}{k_n}a_t(u,\varphi) + \theta_1 a(u)(\varphi) + \theta_p b(u,\varphi) + c(u,\varphi) \\ &= \frac{1}{k_n}a_t(u^{k-1})(\varphi) - \theta_2 a(u^{k-1})(\varphi) \\ &+ \theta_3\{(f^{k-1},\varphi_v) + (q_T^{k-1},\varphi_T)\} + \theta_4\{(f^k,\varphi_v) + (q_T^k,\varphi_T)\} \end{aligned} \tag{5.4}$$

holds for all $\varphi = (\varphi_v,\varphi_p,\varphi_T) \in \widetilde{Z}$. For the special choice given in Table 5.1 we obtain the one-step θ-schemes. In the following Table 5.2 we present two variants of the fractional-step

Table 5.1. Choice of parameters to obtain one-step θ-schemes.

	θ_1	θ_2	θ_3	θ_4	θ_p	t_{k-1}	t_k
explicit Euler scheme	0	1	1	0	1	t_{n-1}	t_n
implicit Euler scheme	1	0	0	1	1	t_{n-1}	t_n
Crank-Nicolson scheme	0.5	0.5	0.5	0.5	1	t_{n-1}	t_n

θ-scheme. Both are of second order, see [49]. The step-size k_n is divided into three sub-steps. Let

$$\theta = 1 - \frac{\sqrt{2}}{2}, \quad \theta' = 1 - 2\theta, \quad \alpha = \frac{\theta'}{1-\theta}, \quad \beta = 1-\alpha,$$

be the choice of the parameters introduced in Table 5.2. Solving three sub-steps of (5.4) with this choice of parameters gives the fractional step θ-scheme.

5.1.4 Temporal discretization of the Shallow-Water equations

We apply the same time-stepping schemes to the nonlinear Shallow-Water equations (3.16). For this reason we recall the spaces X_{2D} and $\widetilde{X}_{2D}$

$$\begin{aligned} X_{2D} := \Big\{ u = (v_{2D},h) \;\Big|\; & v_{2D} \in L^2(I,H_0^1(\Omega_{2D})^2), \partial_t v_{2D} \in L^2(I,L^2(\Omega_{2D})^2), \\ & h \in L^2(I,H_0^1(\Omega_{2D})), \partial_t h \in L^2(I,L^2(\Omega_{2D}))\Big\}, \end{aligned} \tag{5.5}$$

Table 5.2. Choice of parameters to obtain fractional-step θ-schemes.

	θ_1	θ_2	θ_3	θ_4	θ_p	t_{k-1}	t_k
FS0	$\alpha\theta$	$\beta\theta$	$\beta\theta$	$\alpha\theta$	θ	t_{n-1}	$t_{n-1}+\theta k_n$
	$\beta\theta'$	$\alpha\theta'$	$\alpha\theta'$	$\beta\theta'$	θ'	$t_{n-1}+\theta k_n$	$t_n-\theta k_n$
	$\alpha\theta$	$\beta\theta$	$\beta\theta$	$\alpha\theta$	θ	$t_n-\theta k_n$	t_n
FS1	$\alpha\theta$	$\beta\theta$	θ	0	θ	t_{n-1}	$t_{n-1}+\theta k_n$
	$\beta\theta'$	$\alpha\theta'$	0	θ'	θ'	$t_{n-1}+\theta k_n$	$t_n-\theta k_n$
	$\alpha\theta$	$\beta\theta$	θ	0	θ	$t_n-\theta k_n$	t_n

and

$$\widetilde{X}_{2D} := \Big\{ u=(v_{2D},h) \;\Big|\; v_{2D}\in H_0^1(\Omega_{2D})^2, h\in H_0^1(\Omega_{2D}) \Big\}.$$

Based on the the weak formulation (3.19) we introduce a semi-linear form A on $X_{2D}\times\widetilde{X}_{2D}$.

$$\begin{aligned} A(u_{2D})(\varphi_{2D}) :=& (h\partial_t v_{2D},\varphi_{v_{2D}}) + (h(v_{2D}\cdot\nabla)v_{2D},\varphi_{v_{2D}}) - \nu(h\nabla v_{2D},\nabla\varphi_{v_{2D}}) \\ &+ (gh\nabla h,\varphi_{v_{2D}}) + (\partial_t h,\varphi_h) + (v_{2D}\nabla h,\varphi_h) + (h\nabla\cdot v_{2D},\varphi_h). \end{aligned}$$

To apply the above presented time stepping schemes we decompose the corresponding semi-linear form and introduce a time discrete form $A_t\colon \widetilde{X}_{2D}\times\widetilde{X}_{2D}$ as

$$\begin{aligned} A_t(u_{2D})(\varphi_{2D}) &= a_t(u_{2D})(\varphi_{2D}) + a(u_{2D})(\varphi_{2D}) \\ a_t(u_{2D})(\varphi_{2D}) &:= (hv_{2D},\varphi_{v_{2D}}) + (h,\varphi_h) \\ a(u_{2D})(\varphi_{2D}) &:= (h(v_{2D}\cdot\nabla)v_{2D},\varphi_{v_{2D}}) - \nu(h\nabla v_{2D},\nabla\varphi_{v_{2D}}) \\ &\quad + (gh\nabla h,\varphi_{v_{2D}}) + (v_{2D}\nabla h,\varphi_h) + (h\nabla\cdot v_{2D},\varphi_h), \end{aligned} \tag{5.6}$$

where $a_t\colon \widetilde{X}_{2D}\times\widetilde{X}_{2D}\to\mathbb{R}$ and $a\colon \widetilde{X}_{2D}\times\widetilde{X}_{2D}\to\mathbb{R}$. Since nonlinear terms also occur in the form $a_t(u)(\varphi)$ we adjust the time discretization to obtain second order of convergence in the case of the Crank-Nicolson scheme and the fractional-step θ-schemes.

Given the previous solution $u_{2D}^{k-1}=(v_{2D}^{k-1},h^{k-1})$ we are concerned with solving the following problem in each sub-step: Find $u_{2D}=(v_{2D},h):=u_{2D}^k=(v_{2D}^k,h^k)\in\widetilde{X}_{2D}$ such that

$$\begin{aligned} &\frac{1}{k_n}\Big\{(h^{k-\frac12}(v_{2D}-v_{2D}^{k-1}),\varphi_{v_{2D}}) + (h,\varphi_h)\Big\} + \theta_1 a(u)(\varphi) = \frac{1}{k_n}(h^{k-1},\varphi_h) - \theta_2 a(u^{k-1})(\varphi) \\ &+ \theta_3(F^{k-1},\varphi_{v_{2D}}) + \theta_4(F^k,\varphi_{v_{2D}}), \quad \forall\varphi=(\varphi_{v_{2D}},\varphi_h)\in\widetilde{X}_{2D}, \end{aligned} \tag{5.7}$$

where $h^{k-\frac12}=\frac{h^k-h^{k-1}}{2}$. The terms $b(\cdot,\cdot)$ and $c(\cdot,\cdot)$ in (5.4) do not occur because of the absence of pressure.

5.1.5 Temporal discretization of the Boussinesq equations in ALE formulation

For the time discretization of the Boussinesq equations in ALE formulation we recall the definition of the spaces

$$X := \Big\{ u = (v,p,T) \;\Big|\; v \in L^2(I, H_0^1(\Omega)^3), \partial_t v \in L^2(I, L^2(\Omega)^3), p \in L^2(I, L^2(\Omega)/\mathbb{R}), \\ T \in L^2(I, H_0^1(\Omega)), \partial_t T \in L^2(I, L^2(\Omega)), h \in L^2(I, H_0^1(\Omega_{2D})), \partial_t h \in L^2(I, L^2(\Omega_{2D})) \Big\},$$

and

$$\widetilde{X} := \Big\{ u = (v,p,T) \;\Big|\; v \in H_0^1(\Omega)^3, p \in L^2(\Omega)/\mathbb{R}, T \in H_0^1(\Omega), h \in H_0^1(\Omega_{2D}) \Big\}. \tag{5.8}$$

Based on the variational formulation in the ALE framework (3.31) we apply the discretization in time. We introduce a semi-linear form $A\colon X \times \widetilde{X} \to \mathbb{R}$ defined as

$$\begin{aligned} A(u)(\varphi) :=& (h\partial_t v, \varphi_v) + (h(F^{-1}(v - \partial_t \mathcal{A}^t)\cdot\nabla)v, \varphi_v) - \gamma(hpIF^{-T}, \nabla\varphi_v) \\ &+ (\nu h \nabla v F^{-1} F^{-T}, \nabla\varphi_v) - \gamma(\varrho h \boldsymbol{g} T, \varphi_v) \\ &+ (h\partial_1 v_1 + h\partial_2 v_2 - x_3\partial_1 h \partial_3 v_1 - x_3 \partial_2 h \partial_3 v_2 + \partial_3 v_3, \varphi_p) \\ &+ (h\partial_t T, \varphi_T) + (h(F^{-1}(v - \partial_t \mathcal{A}^t)\cdot\nabla)T, \varphi_T) + (\nu_T h F^{-1} F^{-T} \nabla T, \nabla\varphi_T) \\ &+ (\partial_t h + v_1^S \partial_1 h + v_2^S \partial_2 h - v_3^S, \varphi_h)_{\Omega_{2D}}. \end{aligned}$$

Again we split the terms into sums to apply the time discretization (5.4). A new form $A_t\colon \widetilde{X} \times \widetilde{X} \to \mathbb{R}$ for the time discretization is introduced:

$$\begin{aligned} A_t(u)(\varphi) =& a_t(u)(\varphi) + a(u)(\varphi) + b(u)(\varphi) + c(u)(\varphi), \\ a_t(u)(\varphi) :=& (hv, \varphi_v) + (h, \varphi_T) + (h, \varphi_h)_{\Omega_{2D}} \\ &- (h(F^{-1}\partial_t \mathcal{A}^t \cdot\nabla)T, \varphi_T) - (h(F^{-1}\partial_t \mathcal{A}^t\cdot\nabla)v, \varphi_v), \\ a(u)(\varphi) :=& (h(F^{-1}v\cdot\nabla)v, \varphi_v) + (\nu h \nabla v F^{-1} F^{-T}, \nabla\varphi_v) - \gamma(\varrho h \boldsymbol{g} T, \varphi_v) \\ &+ (h(F^{-1}v\cdot\nabla)T, \varphi_T) + (\nu_T h F^{-1} F^{-T}\nabla T, \nabla\varphi_T) \\ &+ (v_1^S \partial_1 h + v_2^S \partial_2 h - v_3^S, \varphi_h)_{\Omega_{2D}}, \\ b(u)(\varphi) :=& -\gamma(hpIF^{-T}, \nabla\varphi_v), \\ c(u)(\varphi) :=& (h\partial_1 v_1 + h\partial_2 v_2 - x_3\partial_1 h\partial_3 v_1 - x_3\partial_2 h\partial_3 v_2 + \partial_3 v_3, \varphi_p). \end{aligned} \tag{5.9}$$

All the introduced forms are nonlinear and map from $\widetilde{X} \times \widetilde{X}$ into the real numbers. In contrast to the time discretization of the Boussinesq equations without the ALE transformation, nonlinear terms also occur in the forms $a_t(\cdot)(\cdot), b(\cdot)(\cdot)$ and $c(\cdot)(\cdot)$. So we follow the same way for time discretization as in the previous section.

Given the previous solution $u^{k-1} = (v^{k-1}, p^{k-1}, T^{k-1}, h^{k-1})$ we are concerned with solving the following problem in each sub-step: Find $u = (v,p,T,h) := u^k = (v^k, p^k, T^k, h^k) \in \widetilde{X}$ such that

$$\begin{aligned} &\frac{1}{k_n}\Big\{(h^{k-\frac{1}{2}}(v - v^{k-1}), \varphi_v) + (h^{k-\frac{1}{2}}(T - T^k), \varphi_T) + (h, \varphi_h)_{\Omega_{2D}}\Big\} \\ &+ \theta_1 a(u)(\varphi) + \theta_p b(u)(\varphi) + c(u)(\varphi) = \frac{1}{k_n}(h^{k-1}, \varphi_h)_{\Omega_{2D}} - \theta_2 a(u^{k-1})(\varphi) \\ &+ \theta_3\{(f^{k-1}, \varphi_v) + (q_T^{k-1}, \varphi_T)\} + \theta_4\{(f^k, \varphi_v) + (q_T^k, \varphi_T)\} \end{aligned} \tag{5.10}$$

holds for all $\varphi = (\varphi_v, \varphi_p, \varphi_T) \in \widetilde{X}$, where $h^{k-\frac{1}{2}} = \frac{h^k - h^{k-1}}{2}$.

5.2 Issues on meshes

So far, we have only considered semi-discretization in time. Prior to discretize via the finite element method in space we discuss the requirements for meshes in two and three space dimensions. In regard to the multilevel method that will be explained in detail in Section 6.2 we need to emphasize also on the hierarchy of meshes.

Due to the ALE formulation which was introduced in Section 3.2.4 we assume the computational domain $\Omega \subseteq \mathbb{R}^3$ to be polygonal. This domain is partitioned into open cells T. If the domain is three dimensional we use hexahedron. In the case that the domain is two dimensional the cells are quadrilaterals. All cells together form the mesh $\mathcal{T}_h = \{T\}$ of the domain, where the parameter h is given as a cell-wise constant function $h\Big|_T = h_T := \operatorname{diam}(T)$ with the diameter h_T of a cell. The symbol h also denotes the maximum cell diameter, that is

$$h := \max_{T \in \mathcal{T}_h} h_T.$$

Following the standard literature [16], [21] or [26] we define the regularity of a mesh.

Definition 5.1. A mesh $\mathcal{T}_h = \{T\}$ is called regular if the following conditions are fulfilled:

(i) $\overline{\Omega} = \bigcup_{T \in \mathcal{T}_h} \overline{T}$.

(ii) $T_1 \cap T_2 = \emptyset$ for all cells $T_1, T_2 \in \mathcal{T}_h$ with $T_1 \neq T_2$.

(iii) Any face of any cell $T_1 \in \mathcal{T}_h$ is either a subset of the boundary $\partial\Omega$ or a face of another cell $T_2 \in \mathcal{T}_h$.

By one step of refinement we obtain a new mesh $\mathcal{T}_h$ from the mesh $\mathcal{T}_{2h}$. We will often write $\mathcal{T}_\ell$ instead of $\mathcal{T}_{2^\ell h}$ with ℓ indicating the number of refinements.

For our purposes we have to weaken the last condition since we explicitly want to allow for adaptive refinement. In order to incorporate adaptive meshes we introduce so called hanging nodes. More details about hanging nodes will be discussed later in Section 5.2.2.

5.2.1 Splitting of adaptive meshes into levels

The meshes considered in this thesis are obtained from a quasi-uniform, conforming coarse mesh $\mathcal{T}_0$ by consecutively refining mesh cells. In order to achieve high resolution in regions where required, refinement may be restricted to these areas of the domain; examples for controlling this refinement can be found in the rich literature on adaptive mesh refinement, see e.g. [5, 24, 69].

We introduce the notion of an *active cell* of the hierarchy $\{\mathcal{T}_\ell\}$ (see e.g. [2]). These are the cells which are not refined further in any of the triangulations $\mathcal{T}_\ell$. We give an example in Figure 5.9.

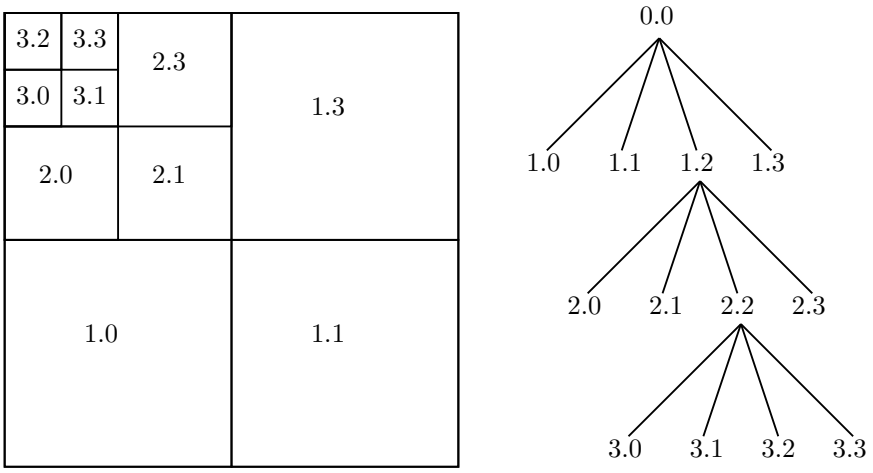


Figure 5.9. Example of active cells in a 2D mesh and corresponding tree graph.

The shape functions of all active cells constitute the space V_L on the finest level. We can also consider the hierarchy of meshes as a tree graph where the grid cells correspond to nodes of the graph and "refinement" corresponds to edges. Then, the cells of $\mathcal{T}_0$ correspond to the roots of the tree (or rather forest) and active cells are the leaves (shaded in Figure 5.10).

The *level* ℓ_T of the cell T in the triangulation hierarchy $\{\mathcal{T}_\ell\}$ is defined recursively as follows: if a cell belongs to the coarse mesh $\mathcal{T}_0$, its level is zero. Otherwise, it was obtained by refinement of another grid cell T_p and we set $\ell_T = \ell_{T_p} + 1$. Since there is no notion of coarsening in a single hierarchy $\{\mathcal{T}_\ell\}$, this level is uniquely defined. The level ℓ_F of a face is defined to be the highest level of the adjacent mesh cells.

We remark that a triangulation $\mathcal{T}_\ell$ does not consist of cells on level ℓ only, but covers the whole domain Ω. Therefore, a single grid cell T with level ℓ_T can belong to several meshes $\mathcal{T}_\ell, \mathcal{T}_{\ell+1}, \ldots$ as shown in Figure 5.10. For instance, the shaded cells on the intermediate level

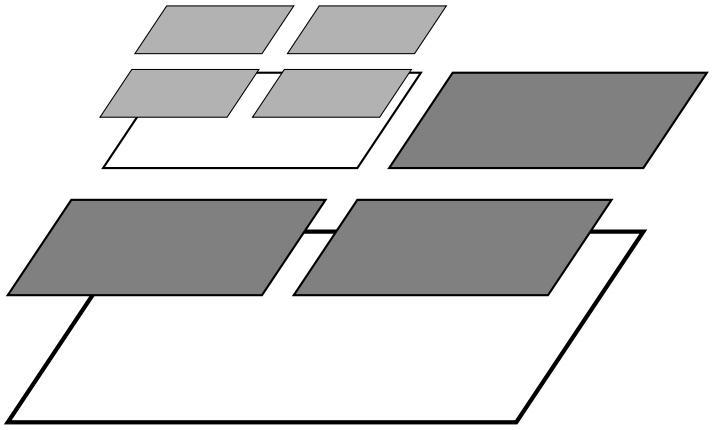


Figure 5.10. A hierarchy of three meshes with local refinement (active cells shaded).

(cell level 1) belong to the meshes $\mathcal{T}_1$ and $\mathcal{T}_2$, the white cell on that level to $\mathcal{T}_1$ only.

Each triangulation $\mathcal{T}_\ell$ will be partitioned into the set of cells strictly on level ℓ

$$\mathcal{T}_\ell^S = \Big\{ T \in \mathcal{T}_\ell \;\Big|\; \ell_T = \ell \Big\},$$

and the set of cells on lower levels

$$\mathcal{T}_\ell^L = \Big\{ T \in \mathcal{T}_\ell \;\Big|\; \ell_T < \ell \Big\}.$$

This partitioning is explained in Figure 5.11, where $\mathcal{T}_2^S$ is shaded and cells in $\mathcal{T}_2^L$ are white. We remark that cells in $\mathcal{T}_\ell^L$ may contain grid cells of several lower levels. Obviously, there

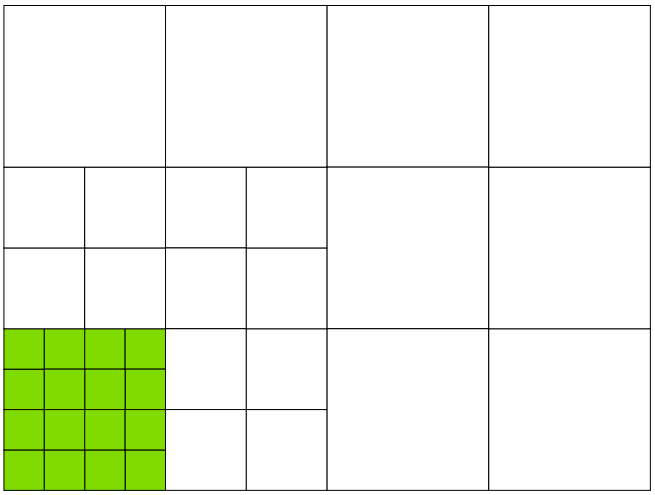


Figure 5.11. Splitting of $\mathcal{T}_\ell$ into $\mathcal{T}_\ell^S$ (shaded cells) and $\mathcal{T}_\ell^L$ (white cells).

holds

$$\mathcal{T}_\ell^S \cup \mathcal{T}_\ell^L = \mathcal{T}_\ell, \qquad \mathcal{T}_\ell^S \cap \mathcal{T}_\ell^L = \emptyset.$$

By $\mathbb{F}_\ell$, we denote the set of all faces of cells in $\mathcal{T}_\ell$. In particular, the set of interior faces $\mathbb{F}_\ell^i$ is the set of faces $F_{ij} = \overline{T}_i \cap \overline{T}_j$, where T_i and T_j are two cells of $\mathcal{T}_\ell$. We call the faces between the sets $\mathcal{T}_\ell^S$ and $\mathcal{T}_\ell^L$ the refinement edge $\mathbb{F}_l^E$ between levels ℓ and $\ell - 1$, that is

$$\mathbb{F}_\ell^E = \left\{ F \in \mathbb{F}_\ell^i \;\middle|\; F \cap \bigcup_{\mathcal{T}_\ell^S} \overline{T} = F \text{ and } F \cap \bigcup_{\mathcal{T}_\ell^L} \overline{T} = F \right\}$$

Mostly for technical reasons, we limit the jump of cell levels ℓ_T across the refinement edge. To this end, we introduce the following two notions:

Definition 5.2. A mesh is **one-irregular**, if the levels of all active cells sharing a face differ by a maximum of one.

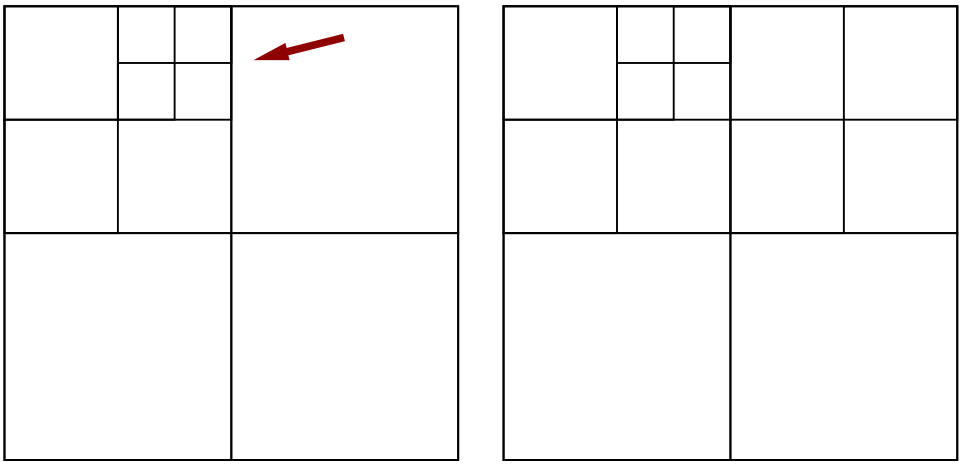


Figure 5.12. Violated one-Irregularity on the left and resolved one-irregular mesh on the right.

In Figure 5.12 we indicate with an arrow where the shown mesh is not one-irregular. Further refinement of the mesh leads to a one-irregular mesh as can be seen on the right mesh in Figure 5.12.

Definition 5.3. A mesh is **v-one-irregular**, if the levels of all active cells sharing a vertex or a face differ by a maximum of one.

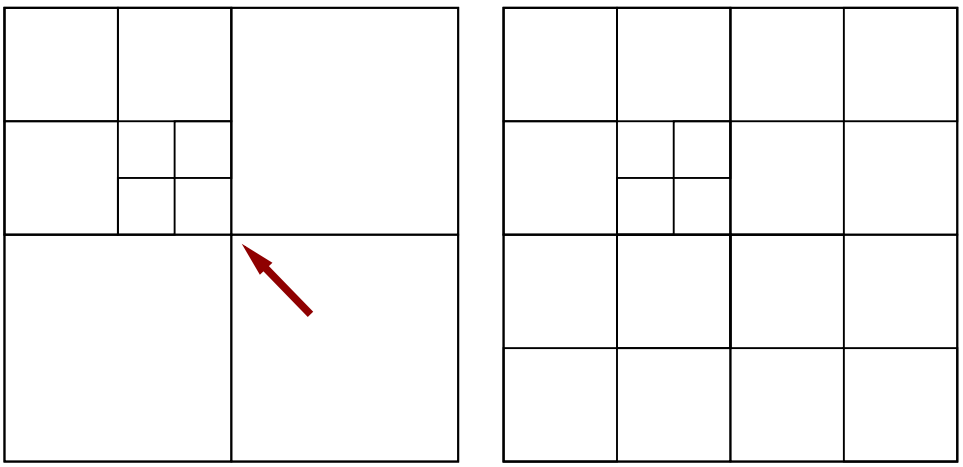


Figure 5.13. Violated v-one-Irregularity on the left and resolved v-one-irregular mesh on the right.

An arrow in Figure 5.13 indicates the vertex where the shown mesh is not v-one-irregular. This can be resolved by refining cells. The v-one-irregular mesh obtained is shown on the left in Figure 5.13.

The notion of one-irregular meshes is ubiquitous in the literature on adaptive refinement with hanging nodes. The additional requirement of being v-one-irregular only acts at corners of the

refinement edge. While one-irregularity was the only condition on the mesh for discontinuous Galerkin methods [44], it is not sufficient, if there are degrees of freedom located on vertices. In this case, we require the mesh to be v-one-irregular. We point out that, while these conditions are not really necessary for the analysis or for the implementation (see e.g.[62]), they are convenient, because they ensure that refinement edges on different levels are separated. In our experience, they are not harmful, since (a) they will not cause global spread of refinement and (b) they are consistent with a uniform approximation quality in most cases.

5.2.2 Hanging nodes

We treat refinement edges by introducing *hanging nodes* on the refined side, not by introducing additional refinement on the coarse side. These correspond to nominal degrees of freedom in

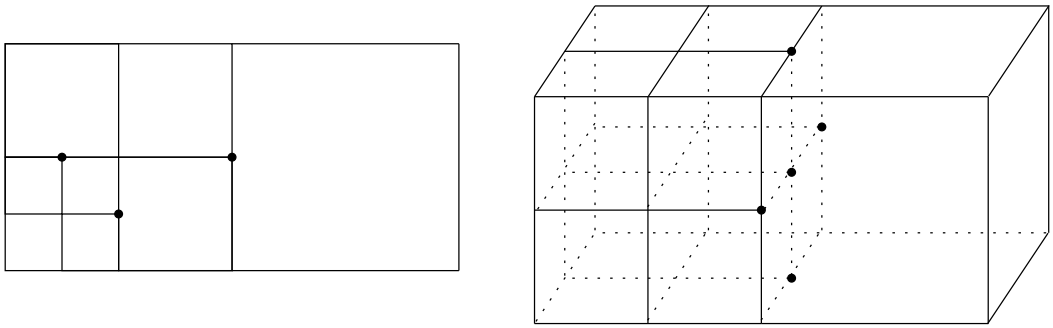


Figure 5.14. Two- and three-dimensional meshes with hanging nodes.

the discretization, but are constrained through the requirement that finite element functions must be conforming across the refinement edge. One option to deal with those degrees of freedom would be eliminating them completely from the linear system. We do not use this technique, since it involves a numbering of degrees of freedom which is not easily represented in the mesh anymore. Instead, we follow [2, 3, 43] in keeping the degrees of freedom in the linear system.

If hanging nodes are used, additional equations are needed to deal with them. These equations are obtained from the continuity condition of the finite element space. This condition essentially states that along the refinement edge, the trace of a function on the refined part of the mesh must be equal to the one on the coarse part. In general, it is the trace operator that ensures conformity of the finite element space.

Thus, we obtain a small linear system of equations, which allows us to eliminate some of the degrees of freedom on the refinement edge [2]. After doing so, a convention has to be found how to represent a function by a coefficient vector. In the implementation in deal.II, two forms are used:

condensed: all degrees of freedom corresponding to hanging nodes are always zero. This is the natural representation for linear forms, since there is no basis function in the finite element space.

distributed: the coefficients on the refined side are set such that the functions on both sides coincide. This is the natural representation for finite element functions, since they are conforming.

As an example, for shape functions linear on edges in two dimensions, the distributed form assigns the mean value of the two neighbors to the hanging node. In order to convert from the distributed form to the condensed form, half of the value in the hanging node is added to its neighbors and its value itself is set to zero. The condensed form is used for the vectors in a linear solver, so as not to spoil the residual by artificial values. Then, whenever multiplication with the system matrix, which was built cell-wise and does not know about the hanging node, is needed, a conversion to distributed and back is needed.

We point out that the concept of hanging nodes is not restricted to linear and bilinear finite elements. Details of its application to higher order elements can be found in [2], its application to local *hp*-refinement in [4]. It has been implemented in the deal.II library [3].

5.3 Discretization in space

The temporal discretized systems that we derived in Section 5.1 are further discretized in space. For spatial discretization of the temporal discrete systems we use the finite element method.

Through semi-discretization in time we obtained spaces $\widetilde{Z}$ in (5.2), $\widetilde{X}_{2D}$ in (5.5) and $\widetilde{X}$ in (5.8). They all contain continuous spaces as $L^2(\Omega)/\mathbb{R}$ and $H_0^1(\Omega)$, for example. To this end we introduce finite dimensional subspaces $V_h^s \subseteq H^1(\Omega)$ of piecewise polynomial functions up to order s.

Following [21] and [26] we define continuous H^1-conforming finite element spaces V_h^s by

$$V_h^s := \left\{ \varphi_h \in C(\overline{\Omega}) \;\middle|\; \varphi_h\Big|_T \in \mathcal{Q}_s(T) \;\forall\, T \in \mathcal{T}_h \right\} \subseteq H^1(\Omega).$$

Here, $\mathcal{Q}_s(T)$ denotes the space of polynomial-like functions on $T \in \mathcal{T}_h$. To give a more precise definition of $\mathcal{Q}_s(T)$, we introduce the space $\hat{\mathcal{Q}}_s(\hat{T})$ of tensor product polynomials up to order s on the reference cell $\hat{T} = (0,1)^d$ given as

$$\hat{\mathcal{Q}}_s(\hat{T}) := \operatorname{span}\left\{ \prod_{i=1}^{d} \hat{x}_i^{\alpha_i} \;\middle|\; \alpha_i \in \{0, 1, \ldots, s\} \right\}.$$

The space $\mathcal{Q}_s(T)$ is obtained by transformations $\kappa_T \colon \hat{T} \to T$, see Figure 5.15, by

$$\mathcal{Q}_s(T) : \left\{ \varphi_h \colon T \to \mathbb{R} \;\middle|\; \varphi_h \circ \kappa_T \in \hat{\mathcal{Q}}_s(\hat{T}) \right\}.$$

For ensuring approximation properties of finite element spaces, additional conditions on the geometry of the cells are required. We state two classical assumptions in this context, namely the so-called *uniformity* and the weaker *quasi-uniformity*, see, for example, [16]:

Definition 5.4 (**Quasi-Uniformity**)**.** A family of meshes $\{\mathcal{T}_h \mid h \downarrow 0\}$ is called *quasi-uniform* if there is a constant K such that the following two conditions are fulfilled:

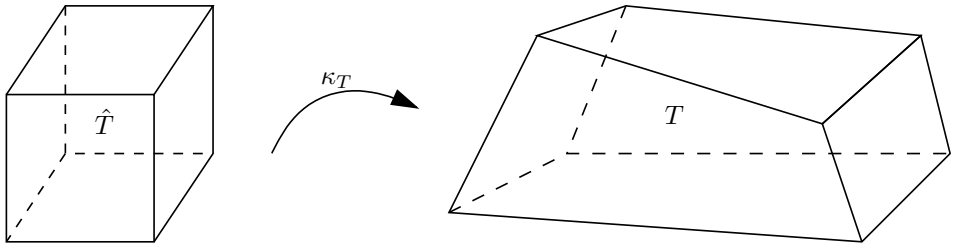


Figure 5.15. Transformation κ_T from the reference cell $\hat{T}$ to a computational cell T

(i) For each transformation $\kappa_T \colon \hat{T} \to T$ it holds

$$\frac{\sup\left\{\|\nabla\kappa_T(\hat{x})x\| \;\middle|\; \hat{x}\in\hat{T}, \|x\|=1\right\}}{\inf\left\{\|\nabla\kappa_T(\hat{x})x\| \;\middle|\; \hat{x}\in\hat{T}, \|x\|=1\right\}} \leq K \;\forall\, T \in \bigcup_h \mathcal{T}_h.$$

(ii) With the diameter ϱ_T of the biggest ball inscribed into the cell T there holds

$$\frac{h_T}{\varrho_T} \leq K \;\forall\, T \in \bigcup_h \mathcal{T}_h.$$

Definition 5.5 (Uniformity). A quasi-uniform family of meshes $\{\mathcal{T}_h \mid h \downarrow 0\}$ is called *uniform* if there is a constant K such that

$$\frac{h}{\varrho_T} \leq K \;\forall\, T \in \bigcup_h \mathcal{T}_h.$$

For the meshes we use throughout this thesis we assume Quasi-Uniformity. With these preliminaries we are able to formulate the fully (in space and time) discretization of the semi-discretized problems presented in Section 5.1. Due to the formulation via semi-linear forms for each problem considered we just have to replace each function by its discrete counterpart.

5.3.1 Galerkin finite element discretization

We begin with formulating the discretization in space of the Boussinesq equations.

Boussinesq equations

Based on the time-discretized problem (5.4) we formulate the fully discrete problem:
Find $u = (v, p, T) := u_h = (v_h, p_h, T_h) \in \widetilde{Z}_h$ such that

$$\begin{aligned}
&\frac{1}{k_n}a_t(u,\varphi) + \theta_1 a(u)(\varphi) + \theta_p b(u,\varphi) + c(u,\varphi)\\
&= \frac{1}{k_n}a_t(u_h^{k-1})(\varphi) - \theta_2 a(u_h^{k-1})(\varphi)\\
&+ \theta_3\{(f^{k-1},\varphi_v) + (q_T^{k-1},\varphi_T)\} + \theta_4\{(f^k,\varphi_v) + (q_T^k,\varphi_T)
\end{aligned}$$

holds for all $\varphi = (\varphi_v, \varphi_p, \varphi_T) := \varphi_h = (\varphi_{h,v}, \varphi_{h,p}, \varphi_{h,T}) \in \widetilde{Z}_h$, where

$$\widetilde{Z}_h := \Big\{ u_h = (v_h, p_h, T_h) \;\Big|\; v_h \in (V_h^q \cap H_0^1(\Omega))^3, p \in V_h^r \cap L^2(\Omega)/\mathbb{R}, T \in V_h^s \cap H_0^1(\Omega) \Big\}.$$

The definition of the bilinear and semi-linear forms are given in (5.3).

Shallow-Water equations

For the Shallow-Water equation (5.7) the finite element discretization reads:
Find $u_{2D} = (v_{2D}, h) := u_{h,2D} = (v_{h,2D}, h_h) \in \widetilde{X}_{h,2D}$ such that

$$\begin{aligned}
&\frac{1}{k_n}\{(h^{k-\frac{1}{2}}(v_{2D} - v_{2D}^{k-1}), \varphi_{v_{2D}}) + (h, \varphi_h) + \theta_1 a(u_{2D})(\varphi) \\
&\quad = \frac{1}{k_n}(h^{k-1}, \varphi_h) - \theta_2 a(u_{2D}^{k-1})(\varphi_h) + \theta_3(F^{k-1}, \varphi_{v_{2D}}) + \theta_4(F^k, \varphi_{v_{2D}}).
\end{aligned}$$

holds for all $\varphi = (\varphi_{v_{2D}}, \varphi_h) := \varphi_h = (\varphi_{h,v_{2D}}, \varphi_{h,h}) \in \widetilde{X}_{h,2D}$ where

$$\widetilde{X}_{h,2D} := \Big\{ u_h = (v_{h,2D}, h_h) \;\Big|\; v_{h,2D} \in V_h^s \cap H_0^1(\Omega_{2D})^2, h \in V_h^r \cap H_0^1(\Omega_{2D}) \Big\}.$$

For the definition of the semi-linear form $a(\cdot)(\cdot)$ we recall (5.6).

Boussinesq equations in ALE formulation

We are concerned with solving the following problem:
Find $u = (v, p, T, h) := u_h = (v_h, p_h, T_h, h_h) \in \widetilde{X}_h$ such that

$$\begin{aligned}
&\frac{1}{k_n}\{(h^{k-\frac{1}{2}}(v - v^{k-1}), \varphi_v) + (h^{k-\frac{1}{2}}(T - T^k), \varphi_T\Big) + (h, \varphi_h)_{\Omega_{2D}}\} \\
&+ \theta_1 a(u)(\varphi) + \theta_p b(u, \varphi) + c(u, \varphi) = \frac{1}{k_n}(h^{k-1}, \varphi_h)_{\Omega_{2D}} - \theta_2 a(u^{k-1})(\varphi) \\
&+ \theta_3\{(f^{k-1}, \varphi_v) + (q_T^{k-1}, \varphi_T)\} + \theta_4\{(f^k, \varphi_v) + (q_{T_k}, \varphi_{h,T})\},
\end{aligned} \tag{5.11}$$

holds for all $\varphi = (\varphi_v, \varphi_p, \varphi_T, \varphi_h) := \varphi_h = (\varphi_{h,v}, \varphi_{h,p}, \varphi_{h,T}, \varphi_{h,h}) \in \widetilde{X}_h$, where

$$\begin{aligned}
\widetilde{X}_h := \Big\{ u_h = (v_h, p_h, T_h) \;\Big|\; & v_h \in (V_h^q \cap H_0^1(\Omega))^3, p \in V_h^r \cap L^2(\Omega)/\mathbb{R}, \\
& T \in V_h^s \cap H_0^1(\Omega), h \in V_h^t \cap H_0^1(\Omega_{2D}) \Big\}.
\end{aligned}$$

The forms occurring in (5.11) are defined in (5.9). For the discretization of the pair (v, p) we use an $\inf - \sup$ stable Taylor-Hood element. We choose $v_h \in (V_h^2 \cap H_0^1(\Omega))^3$ and $p_h \in V_h^1 \cap L^2(\Omega)/\mathbb{R}$. The remaining functions are discretized with bi- or trilinear finite element functions.

5.4 Stabilization

Since the momentum equation as well as the transport equations have mainly hyperbolic character with small or vanishing diffusion, we apply stabilization when using standard finite element methods as developed in [22].

Based on the fully discretized formulation we add stabilization terms. For the momentum equation the additional terms are given as

$$s(u)(\varphi_v) = \sum_{T\in\mathcal{T}_h} (\partial_t v + (v\cdot\nabla)v + \gamma\nabla p - \nu\Delta v - \gamma \boldsymbol{g}\varrho T - f, \delta_T(v\cdot\nabla)\varphi_v)_T.$$

The term $\delta_T((v\cdot\nabla)v, (v\cdot\nabla)\varphi_v)_T$ is a streamline diffusion term. All the other terms are added to preserve consistency. This means that these additional terms vanish if we insert the continuous solution $u = (v, p)$.

In the case of interpolation pairs $V_h^q \times V_h^r$ with $q = r + 1$ the parameter δ_T is given cell-wise as

$$\delta_T = \frac{h_T^2}{q^2\nu + 1}.$$

This includes $\inf - \sup$ stable Taylor-Hood pairs with $r = q - 1$ for spatial discretization of velocity and pressure. More details on how to choose the parameter δ_T can be found in [13].

The stabilization terms have to be transformed via the ALE mapping. We only take the streamline diffusion term into account as all the other terms were added for consistency. This term then becomes

$$s(u)(\varphi) = \sum_{T\in\mathcal{T}_h} ((v\cdot\nabla)v, \delta_T(v\cdot\nabla)\varphi_v)_T = \sum_{T\in\mathcal{T}_h} \delta_T(h(F^{-1}v\cdot\nabla)v, (F^{-1}v\cdot\nabla)\varphi_v)_T.$$

The terms to be added for stabilization of transport equations are given as

$$s(S)(\varphi_S) = \sum_{T\in\mathcal{T}_h} (\partial_t S + (v\cdot\nabla)S - \nu_S\Delta S - q_S, \delta_T(v\cdot\nabla)\varphi_S)_T.$$

Again we drop the terms that occur due to consistency and we transform the remaining streamline diffusion term via the ALE mapping:

$$s(S)(\varphi_S) = \sum_{T\in\mathcal{T}_h} ((v\cdot\nabla)S, \delta_T(v\cdot\nabla)\varphi_S)_T = \sum_{T\in\mathcal{T}_h} (h(F^{-1}v\cdot\nabla)S, \delta_T(F^{-1}v\cdot\nabla)\varphi_S).$$

The fully discretized and stabilized problem reads:
Find $u = (v, p, T, h) := u_h^k = (v_h^k, p_h^k, T_h^k, h_h^k) \in \widetilde{X}_h$ such that

$$\begin{aligned}
&\frac{1}{k_n}\{(h_h^{k-\frac{1}{2}}(v - v_h^{k-1}), \varphi_v) + (h_h^{k-\frac{1}{2}}(T - T_h^{k-1}), \varphi_T\Big) + (h, \varphi_h)_{\Omega_{2D}}\} \\
&+ \theta_1 a(u)(\varphi) + s(u)(\varphi_v) + s(T)(\varphi_T) + \theta_p b(u, \varphi) + c(u, \varphi) \\
&= \frac{1}{k_n}(h_h^{k-1}, \varphi_h)_{\Omega_{2D}} - \theta_2 a(u_h^{k-1})(\varphi) \\
&+ \theta_3\{(f^{k-1}, \varphi_v) + (q_T^{k-1}, \varphi_T)\} + \theta_4\{(f^k, \varphi_v) + (q_T^k, \varphi_T)\},
\end{aligned} \tag{5.12}$$

holds for all $\varphi = (\varphi_v, \varphi_p, \varphi_T, \varphi_h) = (\varphi_{v,h}, \varphi_{p,h}, \varphi_{T,h}, \varphi_{h,h}) \in \widetilde{X}_h$.

One could also think of stabilizing the two dimensional equations. Since it is more important to use stabilization in three dimensions, we leave the two dimensional equations as they are. Of course, the terms of the stabilization need to be applied to the coupled solution $u = u_{3D} + u_{2D}$ as explained in Section 4.2.

5.5 Coupling discretization in 2D and 3D

In Chapter 4 we developed the coupled model. In this section we present the ideas how to realize the coupling for the discretization.

5.5.1 Connecting meshes in 2D and 3D

First of all we discretize the equation in 2D. The whole domain in 2D is triangulated with quadrilaterals. In those parts of the 2D domain where a better accuracy is required we add a triangulation in 3D. A main issue is the connection between these meshes. Functions that are needed are

(i) Identify the corresponding 2D cell to each cell in the 3D triangulation.

(ii) If a cell in 3D has to be refined the matching 2D cell should be refined accordingly.

In Figure 5.16 we show a 2D mesh. On the shaded (light blue) cells we placed two 3D cells. The 3D cells are refined into the lower left corner as shown in Figure 5.17. We present the resulting 2D meshes in Figure 5.18 and the combination of the two dimensional and three dimensional meshes is presented in Figure 5.19.

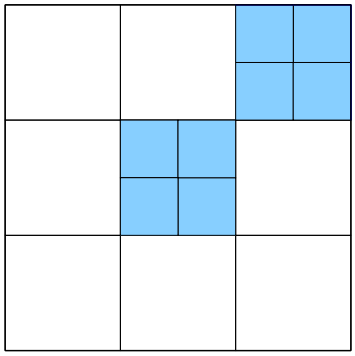


Figure 5.16. Position for two 3D cells on a 2D mesh are colored

Furthermore we have to be able to integrate a 2D finite element function in 3D. This means we have to realize couplings between two dimensional and three dimensional finite element functions as

$$(u_{2D}, \varphi_{3D}), \quad \text{and} \quad (\varphi_{2D}, \varphi_{3D}).$$

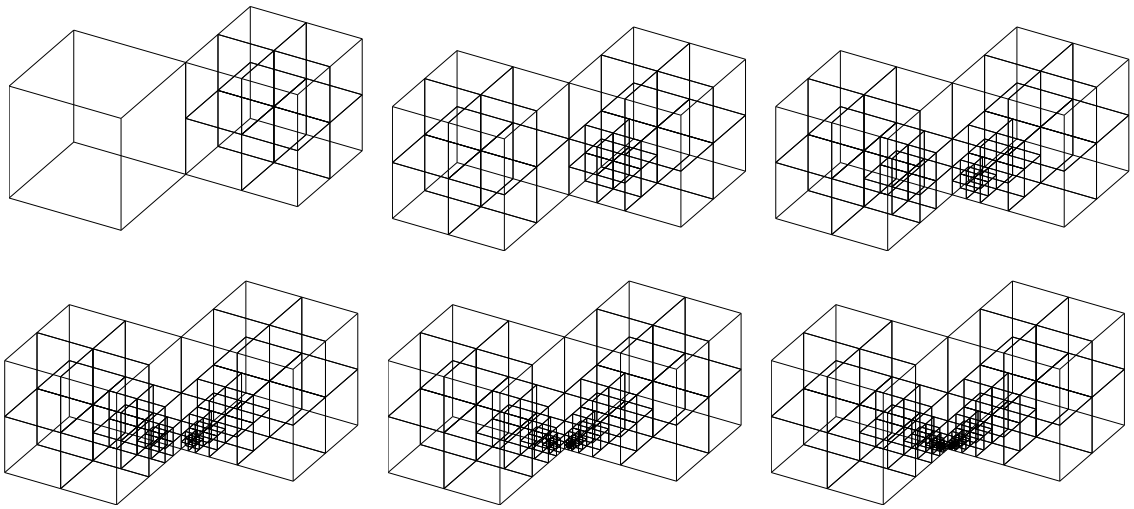


Figure 5.17. Sequence of meshes in 3D refined into a corner

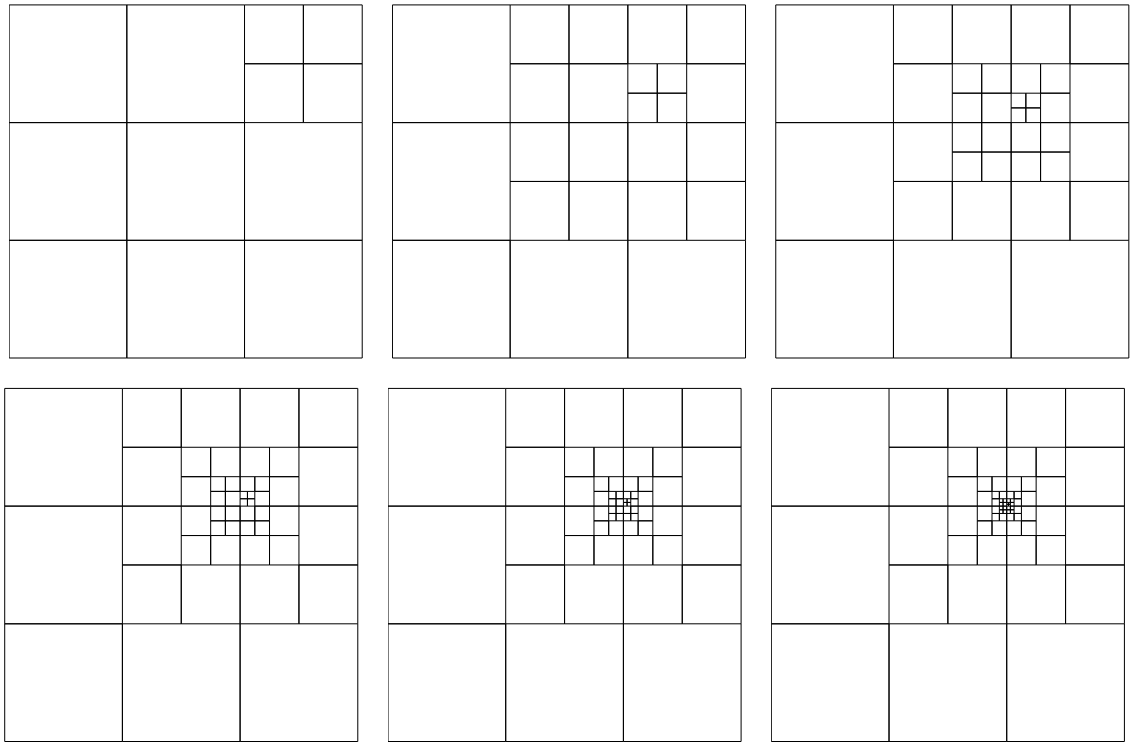


Figure 5.18. Sequence of meshes in 2D refined according to 3D mesh

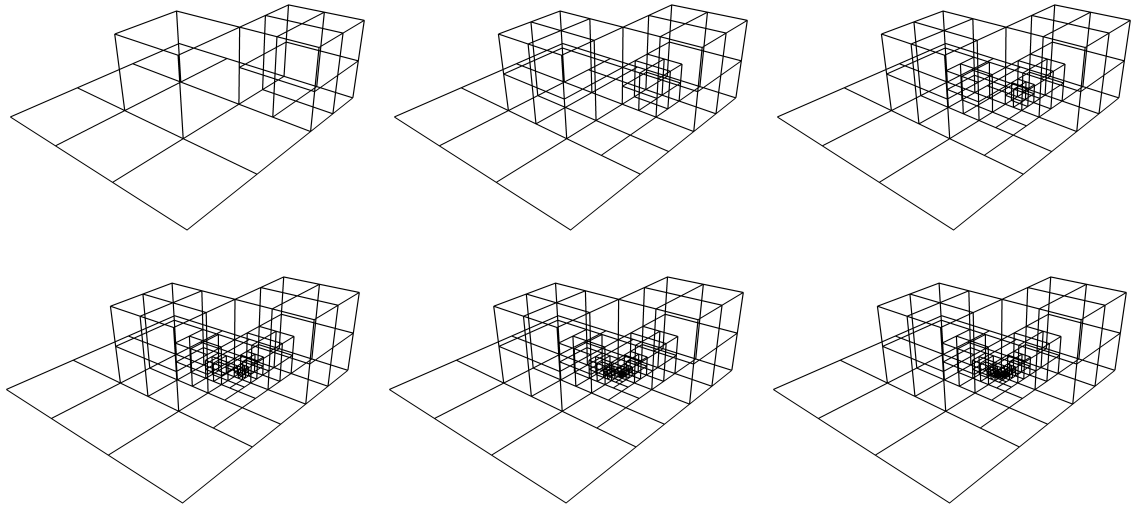


Figure 5.19. Sequence of coupled meshes

These integrals appear in residuals as well as in the semi-linear forms. In the finite element code we used, these kind of couplings was not yet taken into account. We had to come up with new kinds of tools which are

(i) sparsity patterns for coupling finite element functions of different space dimension,

(ii) assembling routines for matrices and residuals for coupling finite element functions of different space dimension,

(iii) matrices that deal with constraints such as boundary conditions and hanging nodes for functions in 2D and 3D at the same time.

All realizations of the mentioned items were implemented in the software library deal.II. In a later Section 6.3 we will comment on the new functionality in detail. To this end we need to mention how residuals and matrices are built between cells and functions in 2D and 3D.

To derive a priori error estimates for the coupled approach, it is necessary to control the error in the two and the three dimensional part. Work on a priori error estimates for free surface flows in one and two dimensions can be found in [23]. The results therein rely on a splitting approach.

6 Algorithm

This chapter describes the algorithms used to solve the discretized problems in Chapter 5. All the occurring problems are nonlinear. We use Newton's method and solve linear systems in each Newton step starting from an initial guess. Depending on the discretized problem we have to deal with a saddle point problem.

This is preconditioned by a block-Schur preconditioner as explained for flow problems in [32]. The saddle point structure is resolved by the application of this preconditioner.

6.1 Basic algorithms

The algorithms explained in this section are either part of the software library deal.II[3] or implemented in the framework of this software package. Let us assume a coupled systems which is fully discretized in space and time as derived in Section 5.1. Then it remains to treat the non-linearities.

6.1.1 Newton's method

Starting from a nonlinear system of equations given as

$$a(U)(\Phi) = (F, \Phi) \quad \forall\, \Phi \in \mathcal{W}. \tag{6.1}$$

we apply a Newton like method. Given an initial guess U_0, we compute updates $\delta U \in \mathcal{W}$ such that

$$a'(U)(\delta U, \Phi) = (F, \Phi) - a(U)(\Phi) \quad \forall \Phi \in \mathcal{W}. \tag{6.2}$$

The next iterate U_{n+1} of Newton's method is then defined by $U_{n+1} = U_n + \lambda \delta U$, where λ is a damping parameter to apply a line search. The directional derivatives are defined as

$$a'(U)(\delta U, \Phi) := \lim_{\varepsilon \to 0} \frac{1}{\varepsilon} \Big\{ a(U + \varepsilon \delta U)(\Phi) - a(U)(\Phi) \Big\} = \frac{\mathrm{d}}{\mathrm{d}\varepsilon} a(U + \varepsilon \delta U)(\Phi) \Big|_{\varepsilon = 0}.$$

6.1.2 Solving linear systems

By Newton's method we obtain linear systems in each step. These systems are solved by Krylov-space solvers like GMRES or Bicgstab. Usually we use the preconditioned versions of these algorithms. As a preconditioner we apply a geometric multilevel directly Section 6.2 or we use block preconditioners. The multilevel preconditioner is then used as a part of the block preconditioner. This will be explained in detail in the following section.

6.1.3 Block preconditioning with Schur complements

After linearization of the non-linear system we have to solve a system of linear equations in each step of the Newton iteration. The block preconditioners we use. are presented on the level of algebraic systems.

As described in Section 4.2 we solve a system for the unknown

$$U = (v_{3D}, p, T_{3D}, v_{2D}, h, T_{2D}).$$

In Table 6.1 we show the couplings between test and trial functions of a fully coupled system.

Table 6.1. Couplings arising from the fully coupled and transformed problem.

	v_{3D}	p	T_{3D}	v_{2D}	h	T_{2D}
φ^v_{3D}	*	*	*	*	*	*
φ^p_{3D}	*	0	0	0	0	0
φ^T_{3D}	*	0	*	*	*	*
φ^v_{2D}	0	0	0	*	*	0
φ^p_{2D}	0	0	0	*	*	0
φ^T_{2D}	0	0	0	0	*	*

This results in a fully discretized linear algebraic problem which reads:

$$\begin{pmatrix} A_{11} & A_{12} & A_{13} & A_{14} & A_{15} & A_{16} \\ A_{21} & 0 & 0 & 0 & 0 & 0 \\ A_{31} & 0 & A_{33} & A_{34} & A_{35} & A_{36} \\ 0 & 0 & 0 & A_{44} & A_{45} & 0 \\ 0 & 0 & 0 & A_{54} & A_{55} & 0 \\ 0 & 0 & 0 & 0 & A_{65} & A_{66} \end{pmatrix} \begin{pmatrix} \delta v_{3D} \\ \delta p \\ \delta T_{3D} \\ \delta v_{2D} \\ \delta h \\ \delta T_{2D} \end{pmatrix} = \begin{pmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \\ f_5 \\ f_6 \end{pmatrix}. \tag{6.3}$$

Following the ideas laid out in [40] we present a block preconditioner for (6.3). To solve system (6.3) we precondition by a matrix P^{-1} and arrive at

$$P^{-1}Ax = P^{-1}b,$$

with $x = (\delta v_{3D}, \delta p, \delta T_{3D}, \delta v_{2D}, \delta h, \delta T_{2D})$ and $b = (f_1, f_2, f_3, f_4, f_5, f_6)$. If we find appropriate entries for P^{-1} such that the condition number of $P^{-1}A$ is moderate, then the whole systems will converge in a few iterations. The derived preconditioner should be close to the inverse of the block system in (6.3), but simple to compute as well. So we neglect some of the off diagonal blocks. We can write the system as

$$\begin{pmatrix} A_{3D,3D} & A_{3D,2D} \\ 0 & A_{2D,2D} \end{pmatrix},$$

and therefore we can approximate the inverse by

$$\begin{pmatrix} A_{3D,3D} & A_{3D,2D} \\ 0 & A_{2D,2D} \end{pmatrix}^{-1} \approx \begin{pmatrix} A_{3D,3D} & 0 \\ 0 & A_{2D,2D} \end{pmatrix}^{-1} = \begin{pmatrix} A_{3D,3D}^{-1} & 0 \\ 0 & A_{2D,2D}^{-1} \end{pmatrix}.$$

This means we have to find approximates for

$$A_{3D,3D}^{-1} = \begin{pmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & 0 & 0 \\ A_{31} & 0 & A_{33} \end{pmatrix}^{-1}, \qquad \text{and} \qquad A_{2D,2D}^{-1} = \begin{pmatrix} A_{44} & A_{45} & 0 \\ A_{54} & A_{55} & 0 \\ 0 & A_{65} & A_{66} \end{pmatrix}^{-1}.$$

We propose to use

$$P^{-1} = \begin{pmatrix} A_{11}^{-1} & 0 & 0 & & & \\ \Sigma^{-1} A_{21} A_{11}^{-1} & -\Sigma^{-1} & 0 & & & \\ 0 & 0 & A_{33}^{-1} & & & \\ & & & A_{44}^{-1} & 0 & 0 \\ & & & 0 & A_{55}^{-1} & 0 \\ & & & 0 & 0 & A_{66}^{-1} \end{pmatrix}$$

as a preconditioner. The matrix Σ is called Schur complement and is defined as

$$\Sigma := A_{21} A_{11}^{-1} A_{12}.$$

For a detailed analysis of the Schur complement approach we refer to [66]. To approximate the inverse of diagonal blocks, we perform a multilevel cycle.

6.2 Multilevel algorithm

To be able to solve three dimensional problem in a reasonable time we apply a multilevel as a preconditioner when solving linear systems with a Krylov-space solver.

In weak formulation, problems we are concerned to solve read find $u \in V$ such that for all $\varphi \in V$ holds:

$$\begin{aligned} a(u, \varphi) &= (f, \varphi), \\ a(u, \varphi) &= (\nabla u, \nabla \varphi). \end{aligned} \tag{6.4}$$

We approximate solutions to this problem by a conforming finite element method, for instance conforming finite elements with tensor product shape functions of degree k (referred to as $\mathcal{Q}_k$). To this end, we introduce a hierarchy of quadrilateral/hexahedral meshes $\{\mathcal{T}_\ell\}$, $0 \leq \ell \leq L$, obtained from a coarse mesh $\mathcal{T}_0$ by consecutive, possibly local, refinement. Let us point out here that, while the process of generating such a hierarchy may involve refinement and coarsening of cells, the final hierarchy can be understood as a quadtree/octree graph only oriented from coarse to fine cells. See Figure 5.10 for an example of such a hierarchy.

In order to compute the approximation of u_ℓ on the mesh $\mathcal{T}_\ell$, we introduce a basis of finite element functions on $\mathcal{T}_\ell$ to generate a "discrete" linear system (see e.g. [20, 27]). Restricting

the bilinear form $a(\cdot,\cdot)$ and the linear form in (6.4) to the finite element spaces V_ℓ of dimension n_ℓ, we obtain linear systems denoted as

$$A_\ell u_\ell = f_\ell. \tag{6.5}$$

Here, u_ℓ and f_ℓ are vectors in $\mathbb{R}^{n_\ell}$ and A_ℓ is a quadratic matrix of dimension n_ℓ. We will refer to the system on the finest level L as the "global" system in order to distinguish it from the level problems introduced later.

In order to solve this system, typically a Krylov-space solver is employed. In a Krylov-space method, new iterates are formed from residual r^k of the current iteration step k. While these methods use orthogonalization techniques to obtain minimization properties of the next iterate and speed up performance (see e.g. [59]), they slow down on fine meshes. To understand this, it is sufficient to consider the Richardson iteration

$$u_\ell^{k+1} = u_\ell^k + \omega r_\ell^k \tag{6.6}$$

as a prototype, denoting that the operation (6.6) is also a part of the cg-method. Here, $r_\ell^k = A_\ell e_\ell^k = f_\ell - A_\ell u_\ell^k$ is the residual of step k and $e_\ell^k = u_\ell - u_\ell^k$ is the error between true and iterative solution in step k. While the addition in (6.6) looks straight forward from the point of view of linear algebra, it is all but this in the function space setting of elliptic problems. There, the function u^k is in the space $H^1(\Omega)$, while the residual is in the space $H^{-1}(\Omega)$ of bounded linear forms on $H^1(\Omega)$ (subject to boundary conditions). The norms of both spaces have very different scaling behavior if applied to oscillating functions. In a nutshell, this is the reason why iterative solvers do not have uniform convergence properties with respect to the refinement level ℓ and slow down considerably on finer meshes.

Remark 6.1. For convenience, we remark that a discrete element of the dualspace is understood as a vector in the sense $l_i = (l, v_i)$ for all $v_i \in V_h$.

In order to speed up convergence, a preconditioner is introduced, which maps the residual back into a function, transforming for instance the Richardson iteration (6.6) into its preconditioned form

$$u_\ell^{k+1} = u_\ell^k + \omega P_\ell^{-1} r_\ell^k. \tag{6.7}$$

For the purpose of this work, P is the multigrid preconditioner studied extensively in the literature (see, e.g., [17, 35, 73]) and discussed here on locally refined meshes. Given smoothers $S_\ell^{(i)}$ and embedding operators $R_\ell^T : V_\ell \to V_{\ell+1}$, the action of P_ℓ^{-1} on a residual vector d_ℓ can be recursively denoted by

We consider two variants of the V-cycle:

classical V-cycle m_ℓ is fixed to the same number independent of the level ℓ; for elliptic problems, typically one or two steps are enough.

variable V-cycle The numbers m_ℓ grow geometrically when the level ℓ decreases. Namely, for $\ell = 1, \dots, L$ exist $1 < \beta_0 \le \beta_1$ such that

$$\beta_0 m_\ell \le m_{\ell-1} \le \beta_1 m_\ell. \tag{6.8}$$

Algorithm 6.1. Multilevel (V-Cycle)

Let $P_0 = A_0$. Set $x^{(0)} = 0$ and compute $P_\ell^{-1} d_\ell$ by the following steps:

1. (Pre-smoothing) Compute $x^{(m_\ell)}$ iteratively by
$$x^{(i)} = x^{(i-1)} + S_\ell^{(i)}(d_\ell - A_\ell x^{(i-1)}), \quad i = 1, \dots, m_\ell.$$
2. (Coarse grid correction) Let
$$y^{(0)} = x^{(m_\ell)} + R_{\ell-1}^T P_{\ell-1}^{-1} R_{\ell-1}(d_\ell - A_\ell x^{(m_\ell)}).$$
3. (Post-smoothing) Compute $y^{(m_\ell)}$ iteratively by
$$y^{(i)} = y^{(i-1)} + S_\ell^{(m_\ell+i)}(d_\ell - A_\ell y^{(i-1)}), \quad i = 1, \dots, m_\ell.$$
4. Set $P_\ell^{-1} d_\ell = y^{(m_\ell)}$.

A typical choice is $\beta_0 = \beta_1 = 2$, which results in doubling the number of smoothing steps when reducing the level and leads to a complexity comparable to the W-cycle. This method is known to yield a uniform preconditioner for any number of smoothing steps on the finest level [17].

In the following subsections, we will first fix a smoother and then recast Algorithm 6.1 in the context of adaptively refined meshes and local smoothing.

Accordingly, the residual is in condensed form and the vector must be returned in the same representation. With this information, we can start rewriting Algorithm 6.1 for locally refined meshes. We will first have to determine, what we consider a "level" in such a case and then determine the subspaces for smoothing.

Following [17, 45], we perform the multilevel method on the complete level spaces V_ℓ, but we restrict the smoother to the part that is really refined to level ℓ, namely the subspace of functions with support covered by $\mathcal{T}_\ell^S$.

6.2.1 Splitting of level spaces

Partitioning of the spaces V_ℓ into subspaces follows the splitting of $\mathcal{T}_\ell$:

$$V_\ell = V_\ell^S \oplus V_\ell^E \oplus V_\ell^L, \tag{6.9}$$

where V_ℓ^S and V_ℓ^L are the functions in V_ℓ with support in $\mathcal{T}_\ell^S$ and $\mathcal{T}_\ell^L$, respectively. V_ℓ^E is the remainder of V_ℓ. It consists of the functions with support in $\mathcal{T}_\ell^S$ **and** $\mathcal{T}_\ell^L$ and is spanned by the basis functions corresponding to node functionals on the refinement edge (see Figure 6.1). Note that in the case of discontinuous Galerkin methods, V_ℓ^E is empty. A basis for the other subspaces is obtained by restricting the definition of the basis of V_ℓ to the subsets of the triangulation. We will assume that the basis is ordered in a way that functions in V_ℓ^S are before those in V_ℓ^E and those before all in V_ℓ^L. Then, a function $u \in V_\ell$ is represented by

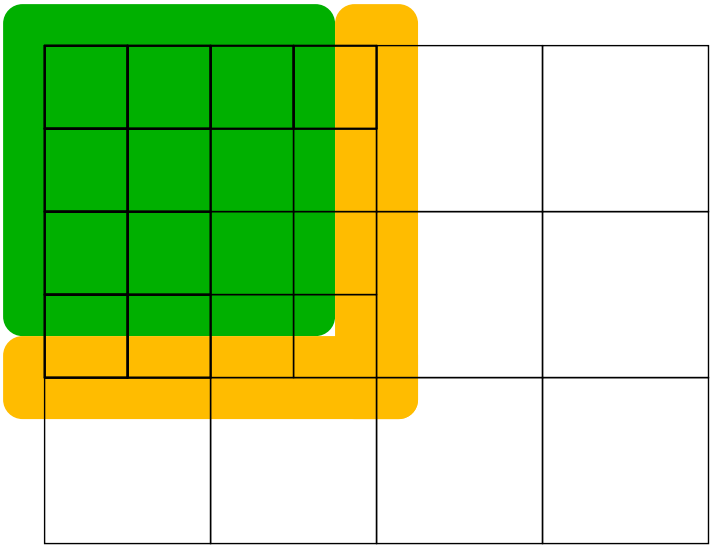


Figure 6.1. The subspaces V_ℓ^S (green), V_ℓ^E (yellow), and V_ℓ^L (white).

a coefficient vector u_ℓ of the form $(u_\ell^S, u_\ell^E, u_\ell^L)^T$. Using the splitting of spaces, (6.5) can be partitioned into the system

$$\begin{pmatrix} A_\ell^{SS} & A_\ell^{SE} & \\ A_\ell^{ES} & A_\ell^{EE} & A_\ell^{EL} \\ & A_\ell^{LE} & A_\ell^{LL} \end{pmatrix} \begin{pmatrix} u_\ell^S \\ u_\ell^E \\ u_\ell^L \end{pmatrix} = \begin{pmatrix} f_\ell^S \\ f_\ell^E \\ f_\ell^L \end{pmatrix}, \tag{6.10}$$

where $u_\ell^S \in V_\ell^S$ and $u_\ell^L \in V_\ell^L$ and $u_\ell^E \in V_\ell^E$. The parts of the right hand side belong to the according dual spaces. The matrices A_ℓ^{SS}, A_ℓ^{LL} and A_ℓ^{EE} are the result of restricting the bilinear form $a(\cdot,\cdot)$ to the spaces V_ℓ^S, V_ℓ^L and V_ℓ^E, respectively. In particular, A_ℓ^{SS} corresponds to a matrix assembled for the interior degrees of freedom of the fine cells with homogeneous Dirichlet boundary conditions on the refinement edge. The matrices A_ℓ^{SE} and A_ℓ^{ES} consist of coupling terms from V_ℓ^S to V_ℓ^E and vice versa. Since we expect the degrees of freedom of V_ℓ^E to be a small number compared to the whole mesh, these matrices only have few nonzero entries and can be stored efficiently.

Due to the elimination of hanging nodes on the refinement edge, all actual degrees of freedom in V_ℓ^E are coarse level degrees of freedom. The other ones are those set to zero by condensing hanging nodes. Therefore, in the actual implementation, the splitting (6.9) reduces to the two spaces V_ℓ^S and V_ℓ^L only, where V_ℓ^E has been added to the latter. Nevertheless, for the sake of presentation, we will continue using a splitting in three spaces.

6.2.2 Prolongation and restriction

The prolongation of a coarse grid vector to the next finer mesh in finite element context is usually the embedding operator. It will be used here as well, but we have to study its action on the different subspaces. It amounts to representing a coarse grid function by fine grid basis

functions. This operation is usually performed with a stencil computed by interpolation on each coarse mesh cell. Since the result of this operation is a coarse grid function, values in hanging nodes will automatically conform to the coarse side of the face. Nevertheless, our goal is representing the vector in condensed form. Therefore, an additional condense operation is necessary. Thus, the structure of the prolongation operator $R_{\ell-1}^T : V_{\ell-1} \to V_\ell$ is:

- Identity for functions in $V_{\ell-1}$, which are in V_ℓ^L as well.
- Standard embedding from $V_{\ell-1}$ into V_ℓ^S taking boundary conditions into account
- For those functions in $V_{\ell-1}$, which require functions in V_ℓ^S and V_ℓ^E for their representation in V_ℓ, we use the standard embedding as well, taking into account that the basis functions in V_ℓ^E after condensation of hanging nodes have non-standard shapes and that there are no basis functions for the node functionals on the refinement edge which do not belong to the coarse grid. To make sure that the function is still in condensed form we incorporate boundary conditions to the restriction matrix.

The restriction operator $R_{\ell-1} : V_\ell \to V_{\ell-1}$ is chosen as the transpose of the prolongation operator in order to preserve symmetry of the method.

6.2.3 Local smoothing

In order to bound the overall complexity of the algorithm linearly by the number of degrees of freedom, we restrict the smoothing method to the subspace V_ℓ^S, that is, to functions with support inside the region $\mathcal{T}_\ell^S$. Since the fine grid degrees of freedom on the refinement edge are eliminated from the global system as "hanging nodes" (see e.g. [43]), they do not contribute to the fine level and can be smoothened on the coarser level. Smoothing on V_ℓ^L will be performed on a coarser level as well. We follow the concept for local smoothing in [44]. Differing from the discontinuous Galerkin methods discussed there, the matrix A_ℓ^{SS} does not originally exist in our data structures, since the actual level space includes V_ℓ^S and all fine grid degrees of freedom on the refinement edge. We denote this level space by $\tilde{V}$. While $\tilde{V}$ does not appear in the analysis of the method, it is the space that actually shows up in the implementation. Instead of A_ℓ^{SS}, we generate the matrix

$$\tilde{A}_\ell = \begin{pmatrix} A_\ell^{SS} & 0 \\ 0 & I \end{pmatrix}, \tag{6.11}$$

which corresponds to the fine level matrix after eliminating "boundary values" on the refinement edge. Here, I is the identity on the space of all fine grid basis functions (including the hanging nodes) on the refinement edge. Then, the smoother in the local version of the algorithm is given as

$$\tilde{S}_\ell^{(i)} = \begin{pmatrix} S_{\ell;S}^{(i)} & 0 \\ 0 & I \end{pmatrix}, \qquad S_\ell^{(i)} = \begin{pmatrix} S_{\ell;S}^{(i)} & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix},$$

where $S_{\ell;S}^{(i)}$ is the restriction of the smoother, for instance the Gauss-Seidel method, to the space V_ℓ^S. Entering the definitions of grid transfer and smoothing operators for locally refined grids into Algorithm 6.1 and simplifying yields

Algorithm 6.2. Multilevel with local smoothing

Let $P_0 = A_0$ and $x^{(0)} = 0$. Then, the action of the operator P_ℓ^{-1} on a vector d_ℓ is defined by:

1. (Pre-smoothing) On the subspace V_ℓ^S only, compute $\tilde{x}^{(m_\ell)}$ iteratively by

$$\tilde{x}^{(i)} = \tilde{x}^{(i-1)} + \tilde{S}_\ell^{(i)}(\tilde{d}_\ell - \tilde{A}_\ell \tilde{x}^{(i-1)}), \quad i = 1, \dots, m_\ell,$$

   with $\tilde{d}_\ell = (d_\ell^S, 0)^T$. Let $x^{(m_\ell)} = (x_S^{(m_\ell)}, 0, 0)^T$ with $x_S^{(m_\ell)}$ the restriction of $\tilde{x}^{(m_\ell)}$ to V_ℓ^S. Since, due to the form of $\tilde{S}_\ell^{(i)}$ and $\tilde{A}_\ell$, the boundary values of $\tilde{x}^{(m_\ell)}$ are equal to zero, we have $x^{(m_\ell)} = (\tilde{x}^{(m_\ell)}, 0)^T$.

2. (Coarse grid correction) Let

$$y^{(0)} = x^{(m_\ell)} + R_{\ell-1}^T P_{\ell-1}^{-1}\big(R_{\ell-1}^S(d_\ell^S - A_\ell^{SS} x_S^{(m_\ell)}) + d_\ell^E - A_\ell^{ES} x_S^{(m_\ell)}\big).$$

3. (Post-smoothing) Compute $y^{(m_\ell)}$ iteratively by

$$\tilde{y}^{(i)} = \tilde{y}^{(i-1)} + \tilde{S}_\ell^{(m_\ell+i)}(\tilde{g}_\ell - \tilde{A}_\ell \tilde{y}^{(i-1)}), \quad i = 1, \dots, m_\ell.$$

   where $\tilde{g}_\ell = (d_\ell^S, 0)^T - (A_\ell^{SE} y_E^{(0)}, 0)^T$

4. Set $P_\ell^{-1} d_\ell = (y_S^{(m_\ell)}, y_E^{(0)}, y_L^{(0)})$.

6.2.4 Overlapping Schwarz smoother

It is well known, that standard Jacobi and Gauss-Seidel smoothers deteriorate dramatically, when the polynomial degree of the finite element discretization is increased. Therefore, we are looking for a smoother, which is nearly as simple, but overcomes this problem. This smoother can be found by using a multiplicative Schwarz method with subspaces related to cells or patches of cells. Let $\{V_{\ell,k}\}$ with $k = 1, \dots, N_\ell$ be such a set of subspaces in V_ℓ^S which will be specified in detail below. Then, instead of defining the action of the operator S_ℓ, the following algorithm describes directly how to obtain $x^{(i)}$ from $x^{(i-1)}$ in the pre-smoothing step of Algorithm 6.2 (for details on the relation to S_ℓ, see for instance [73]): Here, $A_{\ell,k}$ is the projection of the matrix A_ℓ^{SS} onto the subspace $V_{\ell,k}$. Additionally, we define the symmetric version of Algorithm 6.3 as the modification where the loop $k = 1, \dots, N_\ell$ is followed by a loop $k = N_\ell, \dots, 1$. The post-smoothing is done accordingly.

It remains to specify the subspaces $V_{\ell,k}$. In the elliptic case, the inversion of cell matrices has proven very successful for discontinuous Galerkin methods [44]. There, a block Gauss-Seidel smoother based on inverting cell matrices yields preconditioners with very weak dependence on the polynomial degree. With continuous elements, an overlapping smoother is more natural. Thus, for a cell T_k on level ℓ let $V_{\ell,k}$ be the subspace of functions in V_ℓ^S which are not identically zero on T_k. Computational results for this smoother are reported in Section 6.2.7.

Algorithm 6.3. Multiplicative Schwarz smoother

One step of the multiplicative Schwarz smoother with right hand side d_ℓ is defined by

1. Let $y^{(0)} = \tilde{x}^{(i-1)}$.
2. For each $k = 1, \dots, N_\ell$, compute $\tilde{y}^{(k)} \in V_{\ell,k}$ as the solution of

$$A_{\ell,k}\tilde{y}^{(k)} = P_k \left(d_\ell - A_\ell^{SS} y^{(k-1)} \right),$$

   where P_k is the ℓ^2-projection from V_ℓ to $V_{\ell,k}$, and let

$$y^{(k)} = y^{(k-1)} + \tilde{y}^{(k)}.$$

3. Let $\tilde{x}^{(i)} = y^{(k)}$ for the last value of k.

6.2.5 Complexity of the algorithm

We show, that the number of operations of Algorithm 6.2 grows linearly with the number of degrees of freedom in the hierarchy, if the classical V-cycle is chosen. To this end, we have to assume that the complexity of the smoother $S_{\ell;S}^{(i)}$ is linear with respect to the dimension of the space it acts on, which holds for standard relaxation methods as well as incomplete LU and Choleski factorization with limited fill-in. It holds for the smoother outlined in Section 6.2.4.

First, we note that the pre- and post-smoothing steps operate on V_ℓ^S only, thus in the whole recursive cycle, every degree of freedom will be touched only once by these operations. The exception from this are the degrees of freedom in V_ℓ^E, which are part of the space $\tilde{V}_\ell$, thus involved in smoothing on level ℓ, although not actually smoothened themselves. Additionally, they will be smoothened on level $\ell - 1$. Due to the restriction to one-irregular meshes, they will be in $V_{\ell-1}^S$ and not be smoothened on any coarser mesh. Thus, they might be operated on at most twice per smoothing. Figure 6.2 shows, that the number of faces with hanging nodes is not necessarily of lower order than the degrees of freedom, but bounded by them. This concludes that the contribution of the two smoothing steps is linear with respect to the total number of degrees of freedom in the hierarchy $\{T_\ell\}$.

The situation in Figure 6.2 is extreme and representing the worst case. In actual adaptive refinement cycles, it is much more likely, that the refinement edges form a small subset of the total set of edges. In that case, the additional work due to multiplication with the matrices A^{SE} and A^{ES} becomes negligible.

It remains to study complexity of the grid transfer. According to step 2 of Algorithm 6.2, this transfer consists of three parts: first, the restriction of the initial residual d_L onto all lower level meshes, which is of the order of the total degrees of freedom in the hierarchy. Second, we need to restrict the local residuals after local smoothing. This involves only the result of $A_\ell^{SS} x_S$, which, summed up over all levels, gives again the number of degrees of freedom in the hierarchy. The same holds for the prolongation operator R_ℓ^T, such that we can conclude, that the whole intergrid transfer is of the order of the number of degrees of freedom in the mesh hierarchy.

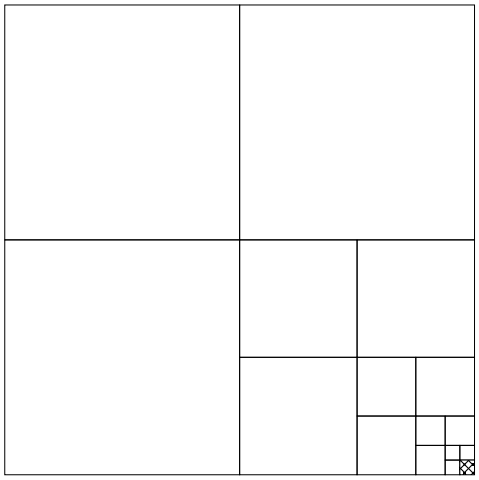


Figure 6.2. A mesh hierarchy with the same number of degrees of freedom on each level. Half of the interior edges bear hanging nodes.

The complexity analysis of standard multigrid methods without local refinement continues by noting that the number of degrees of freedom increases geometrically by factors of four and eight from one level to the next in two and three dimensions, respectively. Thus, it is concluded that the complexity of the V-cycle, variable V-cycle, and W-cycle is bounded linearly by the number of degrees of freedom on the finest level. Such an argument is invalid here. In particular in hierarchies obtained through adaptive iterations before saturation is achieved, this condition is usually violated. An extreme example is the hierarchy in Figure 6.2, which exhibits the same number of degrees of freedom on each level. Thus, we conclude that on general hierarchies we can only show optimal complexity for the V-cycle with respect to the total number of degrees of freedom in the hierarchy.

6.2.6 Memory requirements

The situation for memory requirements is similar to the computational complexity. A standard multigrid method without hanging nodes only involves the matrices A_ℓ^{SS}, since the triangulation subsets $\mathcal{T}_\ell^L$ and $\mathbb{F}_\ell^E$ are empty. The only additional matrices stored in the local method are A^{SE} and A^{ES}. In the worst case example of Figure 6.2, these matrices are of about the same size as A_ℓ^{SS}. Under the more reasonable assumption, that $\mathbb{F}_\ell^E$ is a small subset of the set of faces, these matrices can be stored in a compact way only involving degrees of freedom in V_ℓ^E, thus with negligible memory overhead.

6.2.7 Numerical experiments

Elliptic problems

We test our algorithm with the following model problem: Let $\Omega = (-1, 1)^d$, choose $f \equiv 1$ in (6.4). Starting with a single grid cell $\mathcal{T}_0 = \{\bar{\Omega}\}$, we apply three (artificial) refinement

strategies:

1. Refinement of each grid cell (global refinement) in each step for comparison.

2. Refinement of all cells in the positive quadrant/octant. Figure 6.3 shows the resulting grids with finest cells on levels 2, 3 and 6. The grids are made v-one-irregular (see Definition 5.3) to accommodate for the multilevel method.

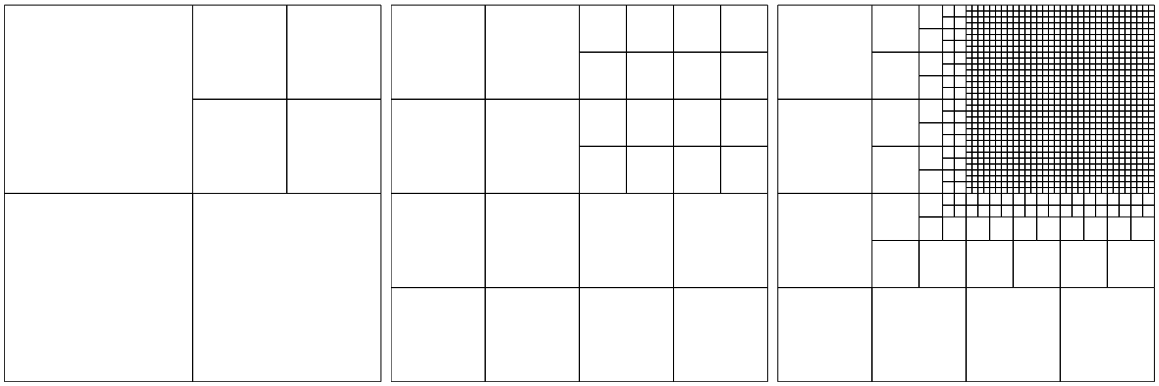


Figure 6.3. Refinement of the positive quadrant, levels 2, 3 and 6, with v-one-irregular closure.

3. Refinement of all cells intersecting the circle of radius $1/(4\pi)$. The resulting grids on levels 3, 4 and 9 are shown in Figure 6.4. Again, the refinement is smoothened to assure that the grid is v-one-irregular.

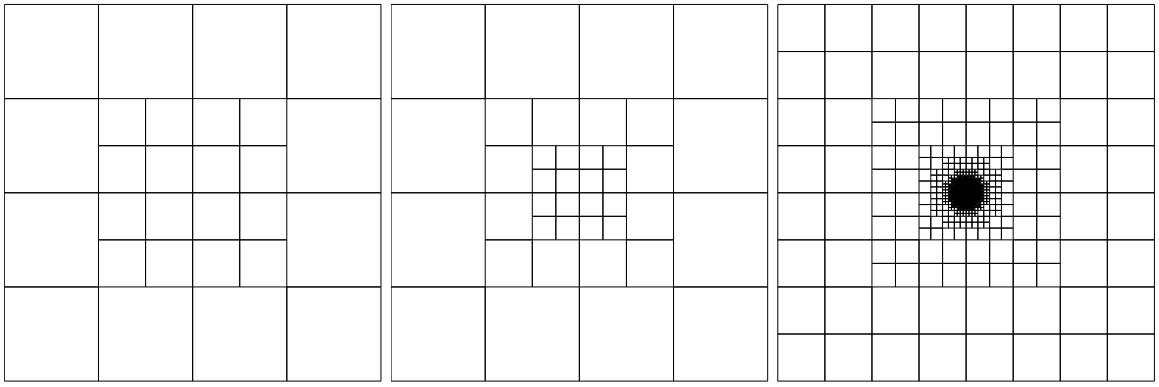


Figure 6.4. Refinement of a circle, levels 3, 4 and 9, with v-one-irregular closure.

The linear systems resulting from the weak formulation (6.4) on these meshes are solved by the conjugate gradient (cg) method with the multilevel preconditioner developed above. First, we use tensor product polynomials of degree one to three on each cell, denoted by $\mathcal{Q}_1$ to $\mathcal{Q}_3$. The start vector is $u^{(0)} = 0$ on each mesh. The values displayed in the tables are the number of steps n_{10} needed to reduce the norm of the residual r by a factor of 10^{10} and the average

logarithmic convergence rate according to Varga [68]

$$\bar{r} := \frac{1}{n} \log_{10} \frac{|r_0|}{|r_n|},$$

where $|r_n|$ is the Euclidean norm of the residual vector r_n after the n-th cg step. Note that while $\bar{r}$ is approximately $10/n_{10}$, it is not rounded to a single digit and thus a finer measure of convergence speed.

In Table 6.2, we report results for the classical V-cycle with one symmetric pre- and post-smoothing steps on each level for the different refinement cases; global refinement is included

Table 6.2. Iteration steps and convergence rates for the preconditioned cg method. One symmetric pre- and post-smoothing step on each level with $\mathcal{Q}_1$-elements in two dimensions.

	global		quadrant		circle	
L	n_{10}	$\bar{r}$	n_{10}	$\bar{r}$	n_{10}	$\bar{r}$
2	1	16.00	1	16.00	1	16.00
3	3	4.83	1	15.99	3	4.83
4	4	2.84	4	2.56	5	2.29
5	5	2.25	6	1.69	6	1.83
6	5	2.15	7	1.61	6	1.76
7	5	2.12	7	1.60	7	1.44
8	5	2.08	7	1.60	7	1.61
9	6	2.06	7	1.60	7	1.57
10	6	2.02	7	1.60	7	1.59
11	6	1.96	7	1.60	7	1.59
12	6	1.92	7	1.60	7	1.59

as a benchmark. As can be seen, the number of steps is independent of the refinement level in all three cases. Moreover, on locally refined meshes the method performs only slightly worse. When we turn to the variable V-cycle (where the smoother is applied in the symmetric fashion suggested in [18]), Table 6.3 shows that this is actually reverted if we only use one smoothing step on the finest level and choose $\beta_0 = \beta_1 = 2$ in (6.8). We compared the same methods to higher order polynomials and obtained the same results. In Table 6.4 we report convergence rates for refinement of a circle and polynomial spaces up to $\mathcal{Q}_9$. They are independent of the refinement level or of the existence of hanging nodes for increasing polynomial degree.

In three dimensions, we compute the same test cases on the cube $(-1, 1)^3$ for examples of the resulting locally refined meshes we refer to Figure 6.5. Convergence rates of the preconditioned conjugate gradient method for this case are shown in Tables 6.5 and 6.6. Again, we see that the convergence rates are independent of the refinement level and nearly independent of the presence of hanging nodes on all meshes. Table 6.7 show thats the convergence rates in three dimensions as well are independent of the refinement level or of the existence of hanging nodes for increasing polynomial degree.

Table 6.3. Iteration steps and convergence rates for the preconditioned cg method for $\mathcal{Q}_1$-elements. Variable smoothing with 1 pre- and post-smoothing step on the finest level.

	global		quadrant		circle	
L	n_{10}	$\bar{r}$	n_{10}	$\bar{r}$	n_{10}	$\bar{r}$
2	1	16.00	1	16.00	1	16.00
3	4	2.53	1	15.99	4	2.53
4	9	1.15	6	1.96	6	1.68
5	9	1.16	8	1.39	6	1.78
6	8	1.28	7	1.46	6	1.82
7	8	1.33	7	1.51	7	1.48
8	8	1.35	7	1.56	7	1.58
9	7	1.43	7	1.59	7	1.57
10	7	1.44	7	1.62	6	1.69
11	7	1.44	7	1.64	6	1.73

Table 6.4. Convergence rates for the preconditioned cg method. Refinement into a circle in 2d. Variable smoothing with 1 pre- and post-smoothing step on the finest level.

L	$\mathcal{Q}_1$	$\mathcal{Q}_2$	$\mathcal{Q}_3$	$\mathcal{Q}_4$	$\mathcal{Q}_5$	$\mathcal{Q}_6$	$\mathcal{Q}_7$	$\mathcal{Q}_8$	$\mathcal{Q}_9$
2	16.0	1.9	1.9	1.6	1.7	1.5	1.5	1.5	1.5
3	2.5	1.3	1.3	1.5	1.5	1.5	1.4	1.5	1.4
4	1.7	1.4	1.5	1.5	1.5	1.5	1.4	1.5	1.5
5	1.8	1.5	1.6	1.6	1.6	1.6	1.5	1.5	1.5
6	1.8	1.7	1.7	1.7	1.6	1.6	1.6	1.6	1.5
7	1.5	1.5	1.6	1.5	1.5	1.5	1.5	1.5	1.4
8	1.6	1.6	1.6	1.6	1.6	1.6	1.6	1.6	1.5
9	1.6	1.6	1.6	1.7	1.6	1.6	1.6	1.6	1.5
10	1.7	1.7	1.7	1.7	1.7	1.7	1.6	1.6	1.6
11	1.7	1.7	1.7	1.8	1.7	1.7	1.7	1.7	1.6
12	1.7	1.7	1.7	1.7	1.7	1.8	1.7	1.7	1.6

Table 6.5. Performance of the preconditioned cg method in three dimensions. One symmetric pre- and post-smoothing step on each level.

	$\mathcal{Q}_1$-elements						$\mathcal{Q}_2$-elements					
	global		octant		ball		global		octant		ball	
L	n_{10}	$\bar{r}$	n_{10}	$\bar{r}$	n_{10}	$\bar{r}$	n_{10}	$\bar{r}$	n_{10}	$\bar{r}$	n_{10}	$\bar{r}$
2	1	16.00	1	16.00	1	16.00	3	4.76	3	4.76	3	4.76
3	2	5.38	1	16.85	2	5.38	4	2.99	5	2.70	4	2.99
4	4	3.13	5	2.28	5	2.08	5	2.51	6	1.72	6	1.73
5	5	2.28	7	1.60	6	1.94	5	2.35	7	1.58	6	1.70
6	5	2.15	7	1.58	6	1.89	5	2.28	7	1.57	6	1.70
7	5	2.08	7	1.58	8	1.37	5	2.27	7	1.57	8	1.35
8	5	2.03	7	1.58	7	1.52			7	1.57	7	1.50

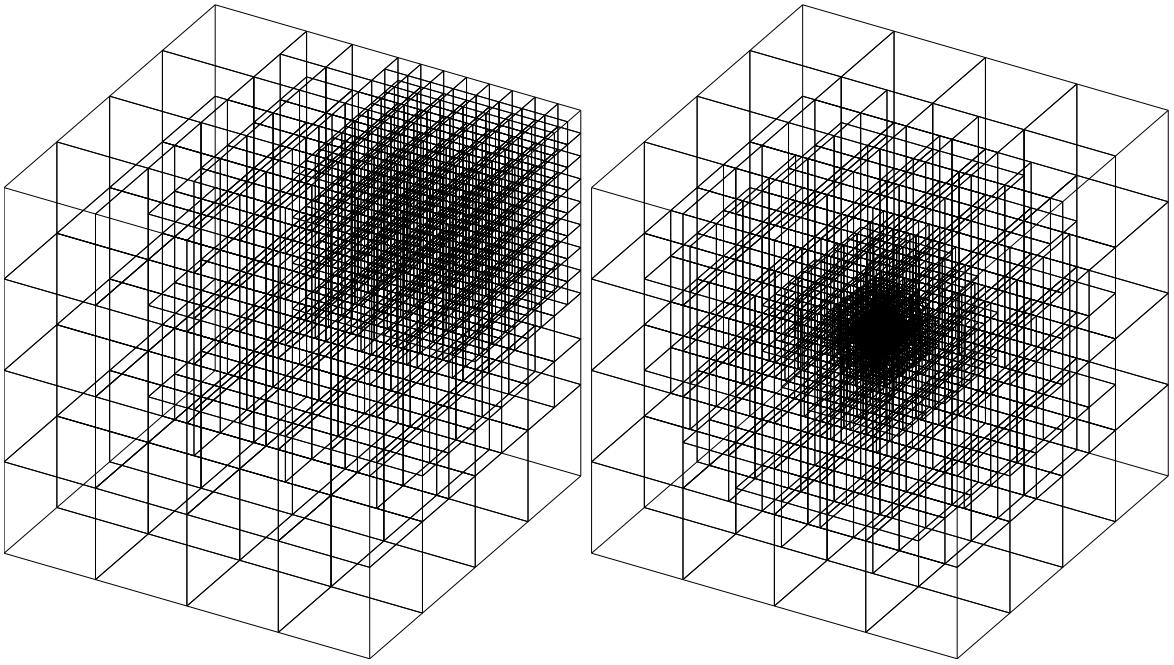


Figure 6.5. Locally refined meshes in three dimensions. First octant refined to level 4 (left) and ball refined to level 6 (right).

Table 6.6. Performance of the preconditioned cg method in three dimensions. Variable block smoothing with 1 pre- and post-smoothing step on the finest level.

	$\mathcal{Q}_1$-elements						$\mathcal{Q}_2$-elements					
	global		octant		ball		global		octant		ball	
L	n_{10}	$\bar{r}$	n_{10}	$\bar{r}$	n_{10}	$\bar{r}$	n_{10}	$\bar{r}$	n_{10}	$\bar{r}$	n_{10}	$\bar{r}$
2	1	16.00	1	16.00	1	16.00	5	2.39	5	2.39	5	2.39
3	4	2.99	1	16.85	4	2.99	7	1.57	5	2.17	7	1.57
4	8	1.32	6	1.94	6	1.84	7	1.46	7	1.58	7	1.55
5	8	1.30	7	1.50	6	1.89	7	1.44	7	1.52	6	1.70
6	8	1.32	7	1.51	6	1.94	7	1.51	7	1.56	6	1.76
7	7	1.44	7	1.57	7	1.46	7	1.55	7	1.60	8	1.41
8	7	1.47	7	1.62	7	1.57			6	1.69	7	1.55

Table 6.7. Convergence rates for the preconditioned cg method. Refinement into a ball in 3d. Variable block smoothing with 1 pre- and post-smoothing step on the finest level.

L	$\mathcal{Q}_1$	$\mathcal{Q}_2$	$\mathcal{Q}_3$	$\mathcal{Q}_4$	$\mathcal{Q}_5$	$\mathcal{Q}_6$	$\mathcal{Q}_7$	$\mathcal{Q}_8$
2	16.0	2.4	2.0	1.8	1.7	1.5	1.5	1.43
3	3.0	1.6	1.5	1.4	1.6	1.5	1.4	1.44
4	1.8	1.6	1.5	1.5	1.5	1.5	1.4	1.44
5	1.9	1.7	1.6	1.6	1.6	1.6	1.5	1.52
6	1.9	1.8	1.7	1.7	1.7	1.7	1.5	1.58
7	1.5	1.4	1.4	1.4	1.4	1.4	1.4	
8	1.6	1.6	1.6	1.6	1.6	1.7	1.6	
9	1.4	1.4	1.5	1.5	1.4	1.5		
10	1.5	1.4	1.5	1.5				
11	1.4	1.4						

6.3 Details on the implementation

Due to the coupling of finite element functions in different dimension in the same equation we had to modify the functionality of deal.II. We use the concept of the MeshWorker that is part of deal.II. In the following section we describe how the MeshWorker is used and that the environment that the MeshWorker creates makes it easy to adapt it to the case we have to deal with.

6.3.1 Fully discrete coupled problem

We explain the details on the coupling of meshes in two dimensions and three dimensions and the resulting coupling of the finite element functions by considering an example. Let us take a look at the Poisson problem (4.1) again: Find $u \in W$ such that

$$(\nabla u, \nabla \varphi) = (f, \varphi) \text{ for all } \varphi \in W.$$

We recall the coupled discrete model which reads: Find $(u_{h,3D}, u_{h,2D}) \in W_{h,3D} \times W_{h,2D}$ such that

$$\begin{aligned}(\nabla u_{3D}, \nabla \varphi_{3D})_\Omega + (\nabla u_{2D}, \nabla \varphi_{3D})_\Omega &= (f, \varphi_{3D})_\Omega,\\ 2(\nabla u_{2D}, \nabla \varphi_{2D})_{\Omega_{2D}} &= (f, \varphi_{2D})_{\Omega_{2D}},\end{aligned}$$

for all $(\varphi_{3D}, \varphi_{2D}) \in W_{3D} \times W_{2D}$. We introduce a bilinear form

$$\begin{aligned}a(u, \varphi) =&(\nabla u_{3D}, \nabla \varphi_{3D})_\Omega + (\nabla u_{2D}, \nabla \varphi_{3D})_\Omega\\ &+ 2(\nabla u_{2D}, \nabla \varphi_{2D})_{\Omega_{2D}}.\end{aligned}$$

Using this bilinear form we have to solve the problem

$$a(u, \varphi) = (f, \varphi_{3D})_\Omega + (f, \varphi_{2D})_{\Omega_{2D}}.$$

Since we also need to be able to solve nonlinear problems we apply a step of defect correction. This would be one step in the Newton algorithm. Given a starting value we solve for δu such that

$$a'(u, \delta u, \varphi) = (f, \varphi_{3D})_\Omega + (f, \varphi_{2D})_{\Omega_{2D}} - a(u, \varphi)$$

holds. This means after discretization with finite elements we have to solve the following problem: Find $(\delta u_{h,3D}, \delta u_{h,2D}) \in W_{h,3D} \times W_{h,2D}$ such that

$$\begin{aligned} &(\nabla \delta u_{h,3D}, \nabla \varphi_{h,3D})_\Omega + (\nabla \delta u_{h,2D}, \nabla \varphi_{h,3D})_\Omega + 2(\nabla \delta u_{h,2D}, \nabla \varphi_{h,2D})_{\Omega_{2D}} \\ &= (f, \varphi_{h,3D})_\Omega + (f, \varphi_{h,2D})_{\Omega_{2D}} - (\nabla u_{h,3D}, \nabla \varphi_{h,3D})_\Omega \\ &- (\nabla u_{h,2D}, \nabla \varphi_{h,3D})_\Omega - 2(\nabla u_{h,2D}, \nabla \varphi_{h,2D})_{\Omega_{2D}}, \end{aligned} \tag{6.12}$$

for all $(\varphi_{h,3D}, \varphi_{h,2D}) \in W_{h,3D} \times W_{h,2D}$.

Let us assume that a part of the coupled mesh looks like Figure 6.6.

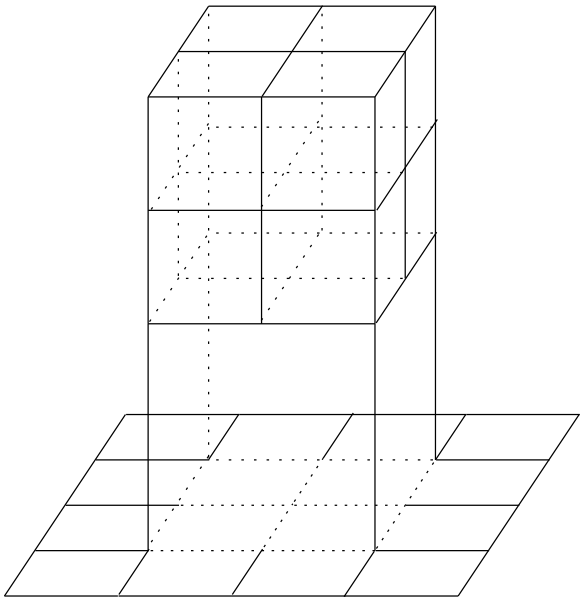


Figure 6.6. Part of a coupled mesh

6.3.2 Integration of coupled terms on active cells

The full discrete system (6.12) has to be integrated. As in every finite element program we perform loops over all cells in the given mesh. In the finite element software we use, this is done by a framework called MeshWorker.

Based on a set of cells in a mesh and an inner integration loop, the MeshWorker assembles the terms. The local contributions on each cell are automatically sorted and added into a global matrix. Various classes provide quadrature rules and finite element functions. The MeshWorker updates all the required information on a cell of a mesh, before the integration is accomplished.

For our purpose we perform the integration based on the three dimensional mesh. This means we loop over all cells in the triangulation of the 3D mesh. Additionally to the basic functionality of the MeshWorker, we need to be able to provide all the information of corresponding two dimensional cell as well. In Figure 6.7 we duplicated the two dimensional mesh to show the correspondence between the three dimensional and two dimensional cells.

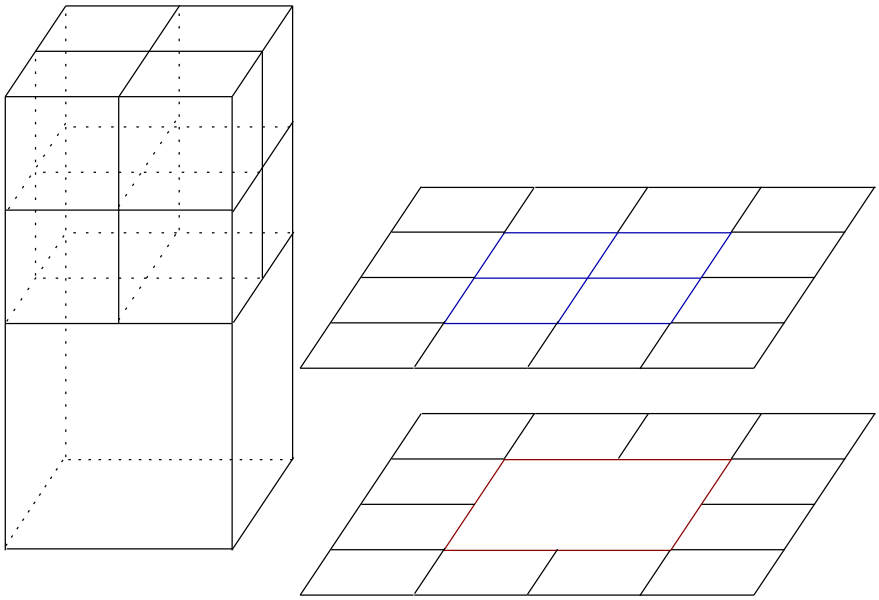


Figure 6.7. Corresponding cells

This figure also exhibits that we need to make sure that the two dimensional mesh is as refined as the three dimensional mesh. Otherwise we would not be able to provide the information for the integration loop. With the corresponding active 2D cell to a cell in the three dimensional mesh we identify the quadrature points on them. The values or gradients of solution variables are extended constantly into the third dimension. This way, we are able to compute both residuals and matrix contributions as

$$(\nabla \delta u_{h,2D}, \nabla \varphi_{h,3D})_T, \quad \text{and} \quad (\nabla u_{h,2D}, \nabla \varphi_{h,3D})_T.$$

This situation is given in the eight cells on the top of the three dimensional mesh in Figure 6.7. A different situation applies in the lower 3D cell. The corresponding cell in the two dimensional mesh is not an active cell.

6.3.3 Integration of coupled terms on inactive cells

The integration of coupled terms on inactive cells is due to local refinement in three dimensions. This way, it happens that an active cell in 3D has a corresponding cell in two dimensions which is not active this means that the 2D cell is further refined. This causes problems in the implementation.

Let us project this situation to a two dimensional sketch, see Figure 6.8. The crosses refer to

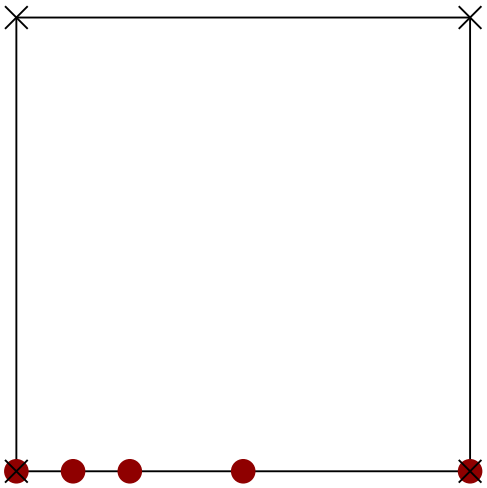


Figure 6.8. Coupling of 3D dofs (×) of an active cell to 2D dofs (•) of a possibly inactive cell

degrees of freedom for the three dimensional cell, whereas the dots refer to degrees of freedom in two dimensions.

In the coupling terms we have to take into account all couplings between two dimensional and three dimensional degrees of freedom in order not to loose information. This requires the knowledge of all children of the corresponding two dimensional cell.

A way to incorporate all these kind of couplings would be to perform the integration on a two dimensional cell and multiply this with the correct three dimensional integral. This is possible due to the tensor product structure of the quadrature rule and the finite element functions. This has not been incorporated yet and is subject to future research.

7 Numerical results

In this chapter we present numerical simulations for the coupled problem for fluid flows.

7.1 Numerical example

We consider the fully coupled problem which reads: Find $u_{2D} = (v_{2D}, h, T_{2D}) \in X_{2D}$ and $u_{3D} = (v_{3D}, p_{3D}, T_{3D}) \in X_{3D}$ such that

$$\begin{aligned} a(u_{2D} + u_{3D})(\varphi_{3D}) &= (F, \varphi_{3D}) \text{ for all } \varphi_{3D} \in X_{3D}, \\ a(u_{2D})(\varphi_{2D}) &= (F, \varphi_{2D}) \text{ for all } \varphi_{2D} \in X_{2D}. \end{aligned}$$

The definition of the semi-linear forms is given in (4.18), (4.20), and (4.21). This is discretized in time and space as explained in Chapter 5. The stabilized form of the three dimensional equation is given in (5.11).

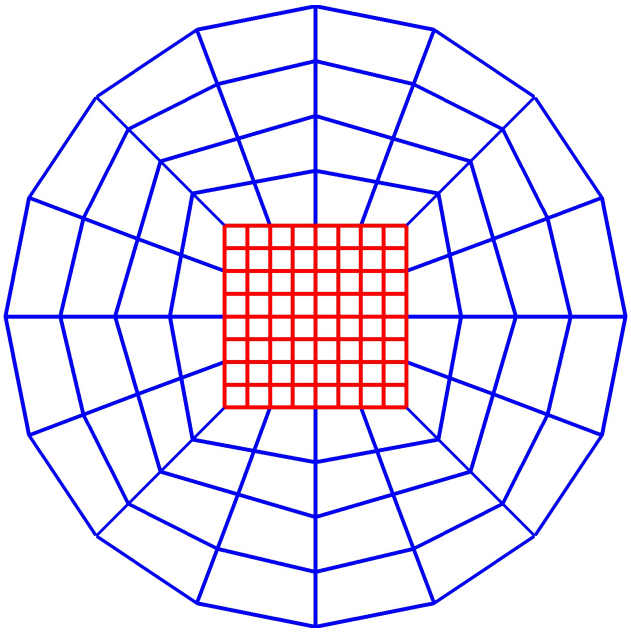


Figure 7.1. Mesh in 2D (blue and red) and mesh in 3D (red)

The two dimensional domain is a circle with center $(0, 0)$ and radius $r = 4$, while the three dimensional cube has the coordinates $(-a, -a, 0)$ with $a = \frac{d}{1+\sqrt{2}}$, $d = \frac{r}{\sqrt{2}}$ at the lower left

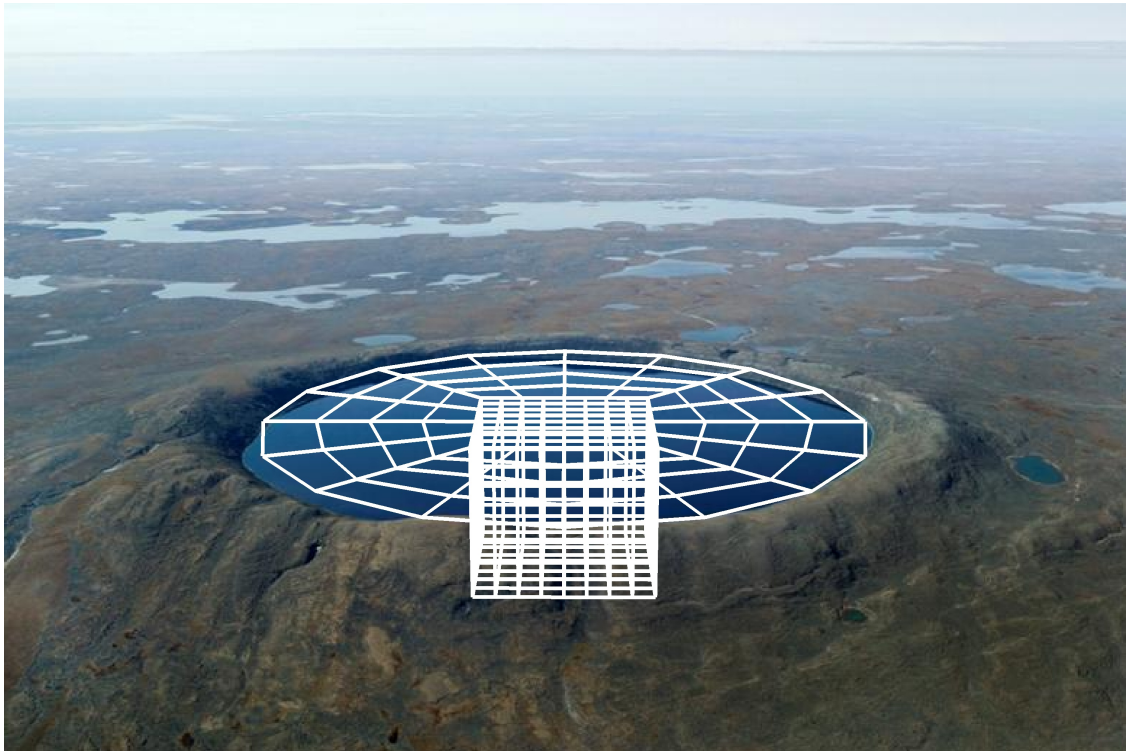


Figure 7.2. Coupled mesh in 2D and 3D

corner and $(a, a, 2)$ at the top right corner. A two dimensional sketch is shown in Figure 7.1 and the full coupled mesh is displayed in Figure 7.2.

7.2 Initial and boundary conditions

For the initial condition we impose for $x \in \Omega_{2D}$

$$
\begin{aligned}
h(0,x) &= 1, \\
v_{2D}^2(t,x) &= \begin{cases} 0.01(x_2-0.5)(x_2+0.5), & \text{if} -0.5 \le x_1 \le 0.5,\ x_2 = 1, \\ 0, & \text{else}, \end{cases} \\
T_{2D}(0,x) &= \begin{cases} 22+2x_2, & \text{if} -0.5 \le x_1 \le 0.5,\ x_2 \le 0, \\ 20, & \text{else}, \end{cases}
\end{aligned}
$$

for $x \in \partial\Omega_{2D}$ we require for $t \in (0,T]$ that $v_{2D}^1(t,x) = 0$, and

$$
\begin{aligned}
v_{2D}^2(t,x) &= \begin{cases} 0.01(x_2-0.5)(x_2+0.5), & \text{if} -0.5 \le x_1 \le 0.5,\ x_2 \le 0, \\ 0, & \text{else}, \end{cases} \\
T_{2D}(t,x) &= \begin{cases} 22+2x_2, & \text{if} -0.5 \le x_1 \le 0.5,\ x_2 \le 0, \\ 15, & \text{else}. \end{cases} \\
S_{2D}(t,x) &= \begin{cases} 1, & \text{if} -0.5 \le x_1 \le 0.5,\ x_2 \le 0, \\ 0, & \text{else}. \end{cases}
\end{aligned}
$$

For the three dimensional correction we impose homogeneous boundary conditions at vertical walls and at the bottom for the velocities v_{3D} and the three dimensional temperature and tracer T_{3D} and S_{3D}. In the case of natural boundary conditions on the top and homogeneous Dirichlet conditions at the bottom we have a reaction term in the two dimensional equation. But here $\alpha = 0$ so the reaction term vanishes.

7.3 Choice of parameters

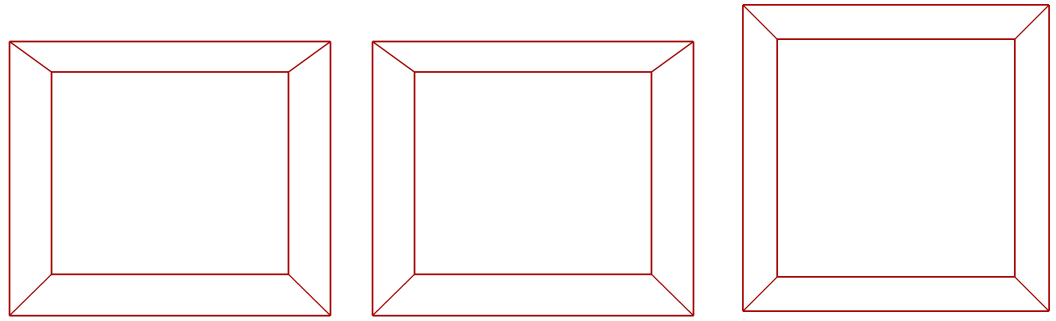
The viscosities are $\nu = 0.001$ and $\nu_T = 0.0001$ and $\nu_S = 0$. The right hand side is zero. The other parameters are $\varrho = 1000$ and $\gamma = 0.00001$. For the time interval we choose $I = [0,T]$ with $T = 20$ and we set the time discretization parameter to $k = 0.01$.

In Figure 7.3 and Figure 7.4 we present streamlines of the correction velocity and the coupled velocity, respectively. At the end, we show the transport and diffusion of a tracer in Figure 7.6 and the temperature in Figure 7.7. We used a fractional-step θ-scheme for the time discretization. For the finite element discretization of the velocity in three dimensions we used tri-quadratic polynomials. All the other unknowns were discretized using bilinear and trilinear finite elements in two or three space dimensions, respectively.

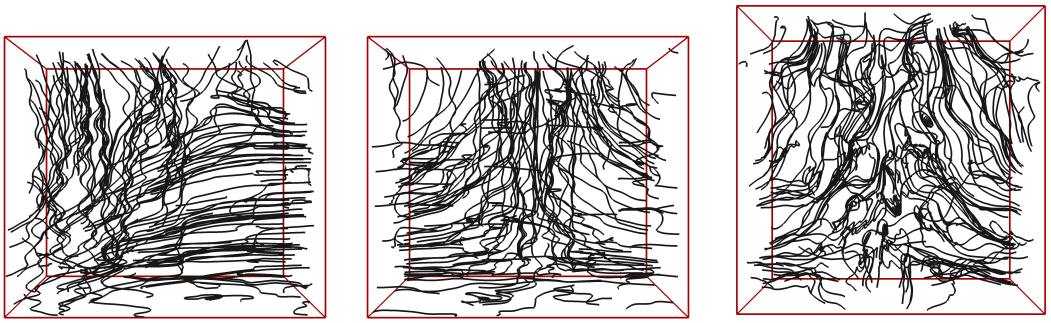
In order to get an impression of the three dimensional part of the simulation, we displayed the cube on top of the two dimensional part. This is only for presentation purposes. The three dimensional part is added to simulate a deep water reservoir and really is below the surface.

Shallow parts of the domain do not need to be resolved in three dimensions. We leave them to the two dimensional part only. It suffices to use the two dimensional model on the shallow part of the lake.

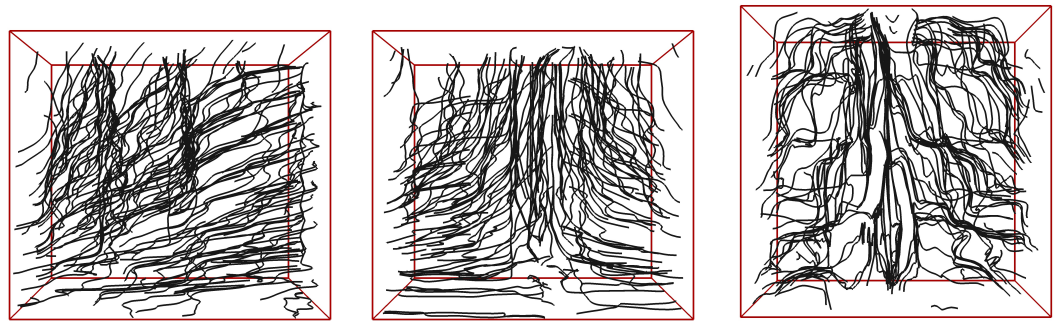
In Figure 7.8 we show the vorticity $\|\nabla \times v(t)\|_\omega$ computed on a patch of cells ω. This patch consists of all cells at the bottom with a center (c_1, c_2, c_3) that satisfies $|c_1| \leq 0.75$, and $|c_2| \leq 0.75$. For the area that covers about 11% of the three dimensional domain we choose the setting described in Section 7.1. We compare the results with a 100% coverage of the three dimensional domain and a pure two dimensional computation which was extended constantly into the third dimension.



(a) $t = 0$



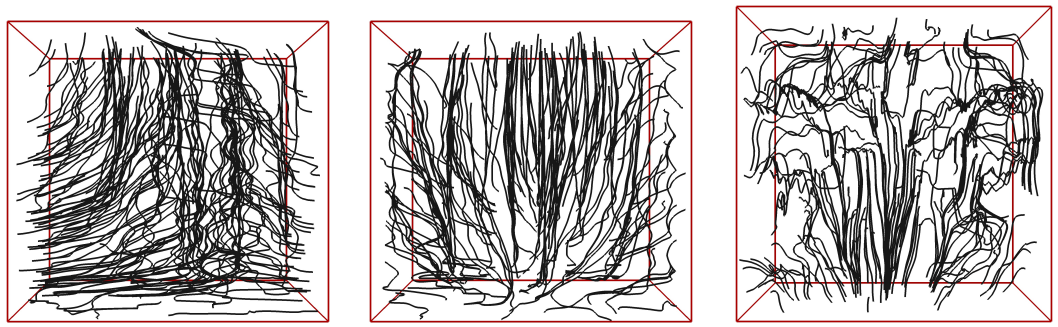
(b) $t = 3$



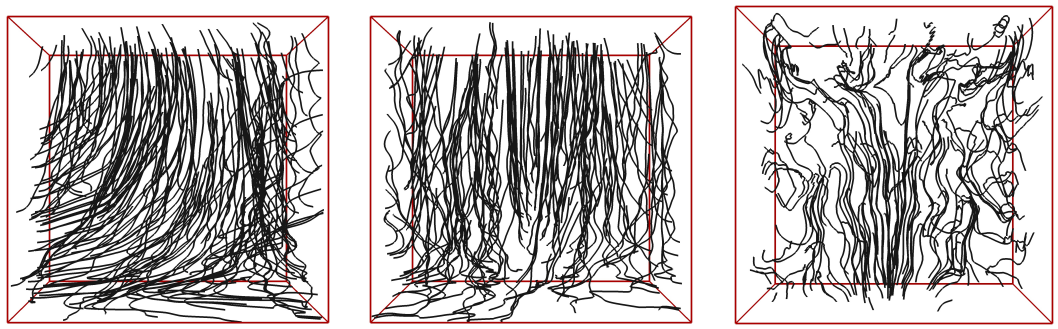
(c) $t = 6$

(d) $t = 9$

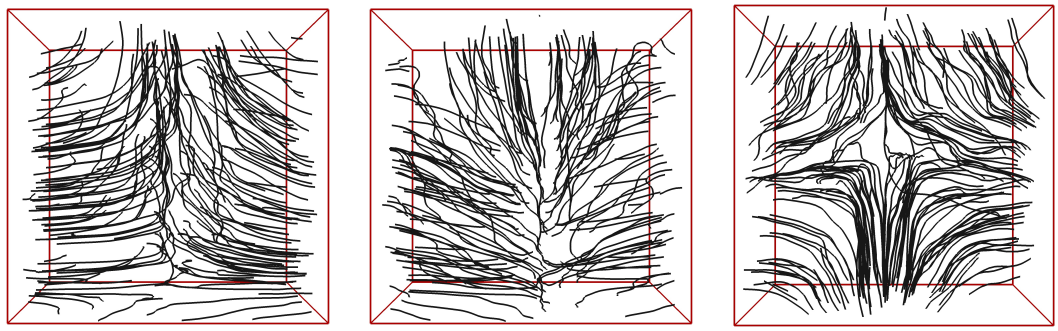
Figure 7.3. Streamlines in x, y, and z-direction at different time points



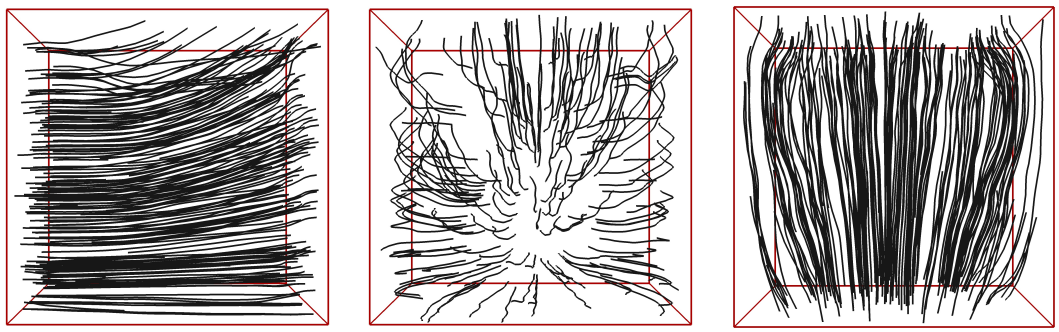
(a) $t = 12$



(b) $t = 15$

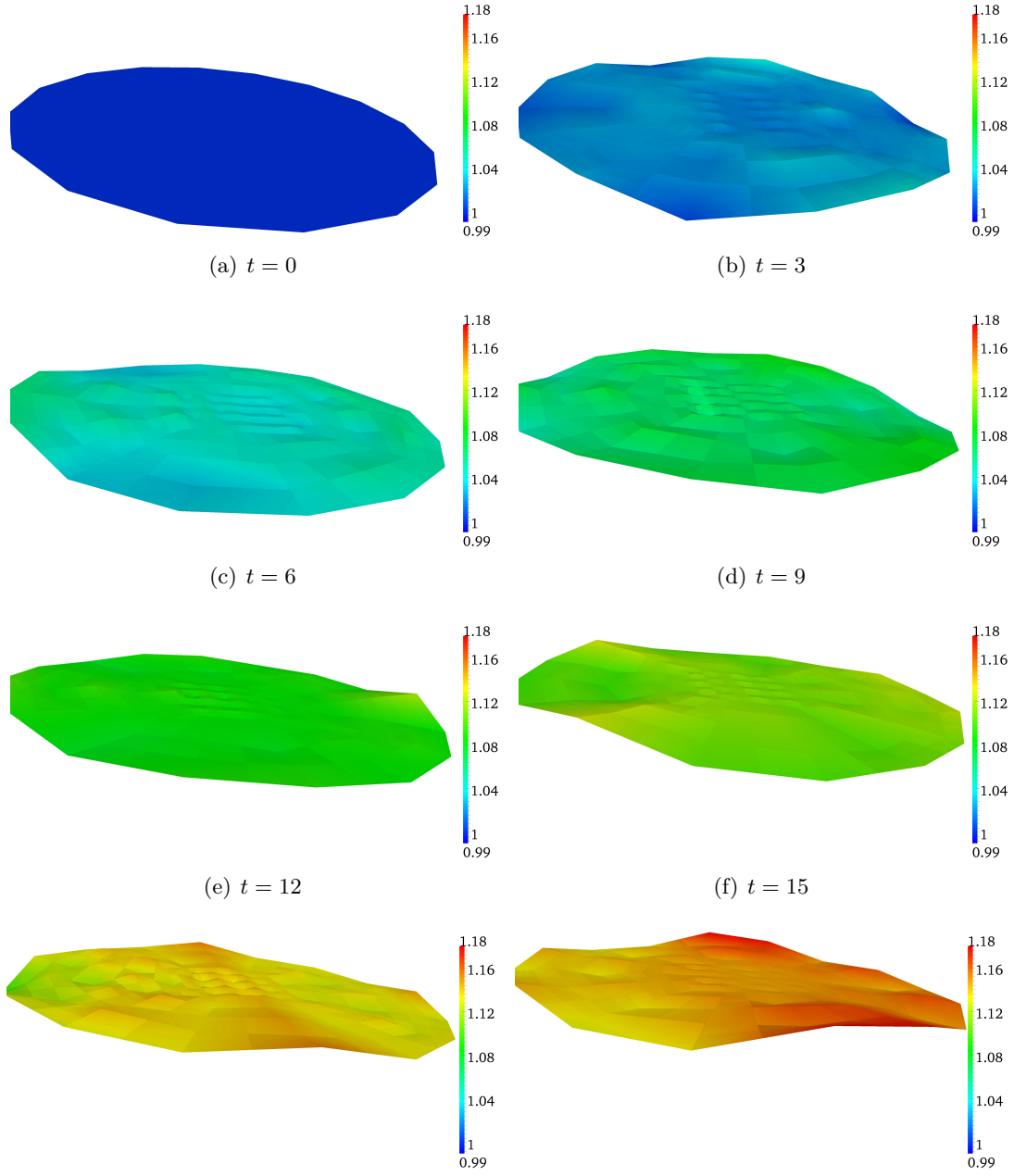


(c) $t = 18$



(d) $t = 20$

Figure 7.4. Streamlines in x, y, and z-direction at different time points, ctd.



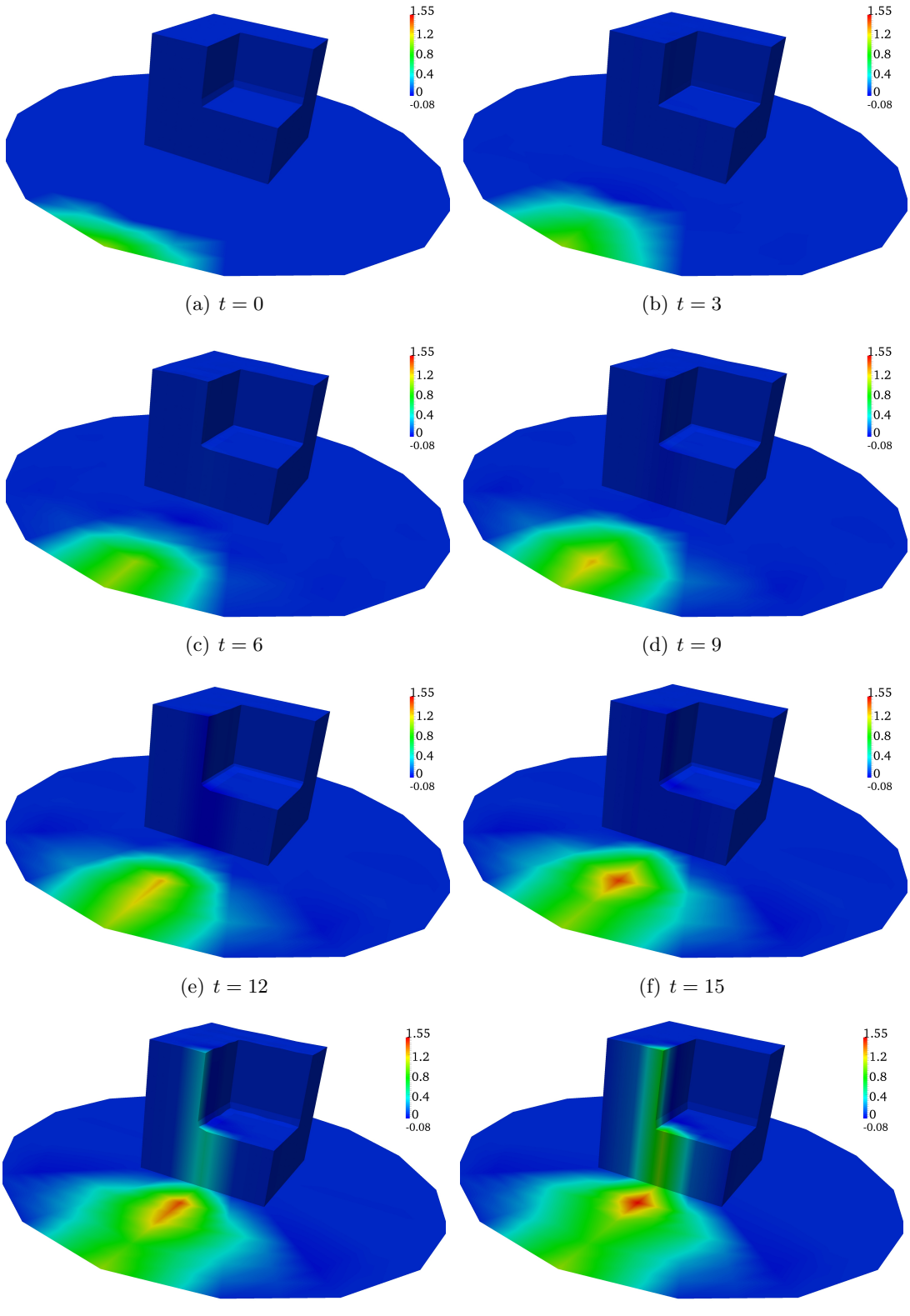
(a) $t = 0$ (b) $t = 3$

(c) $t = 6$ (d) $t = 9$

(e) $t = 12$ (f) $t = 15$

(g) $t = 18$ (h) $t = 20$

Figure 7.5. Height at different time points



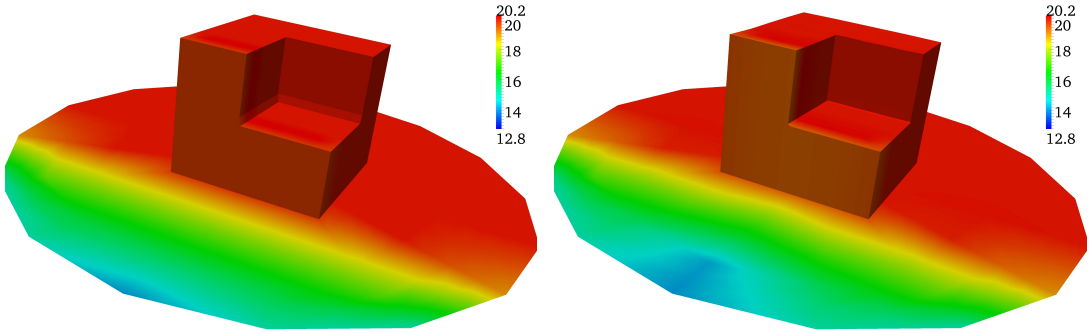
(a) $t = 0$ (b) $t = 3$

(c) $t = 6$ (d) $t = 9$

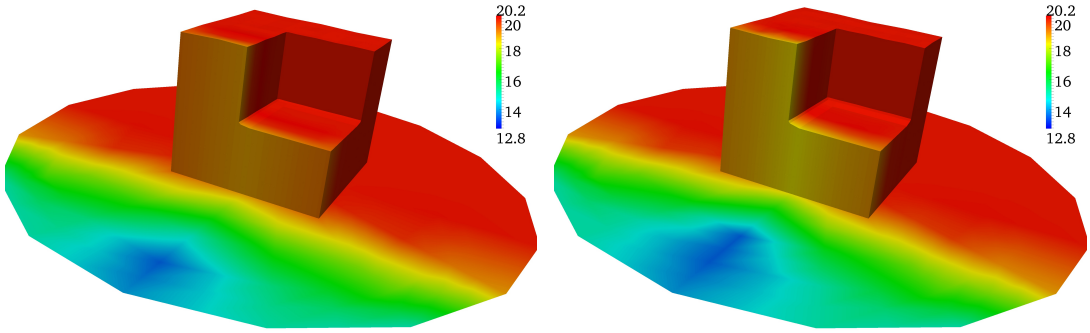
(e) $t = 12$ (f) $t = 15$

(g) $t = 18$ (h) $t = 20$

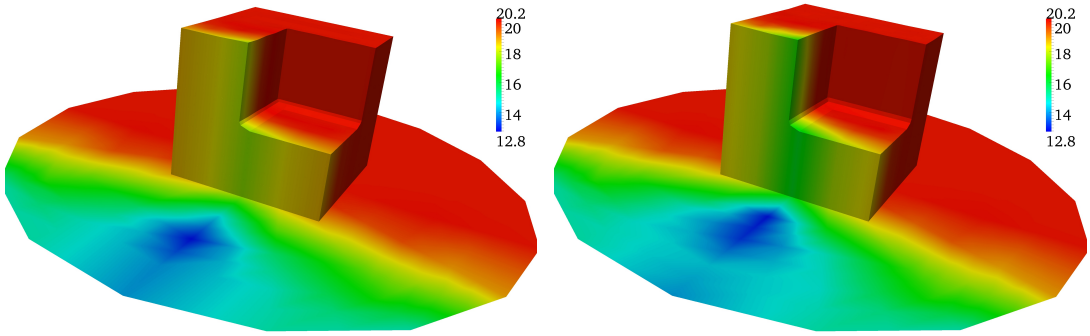
Figure 7.6. Tracer at different time points



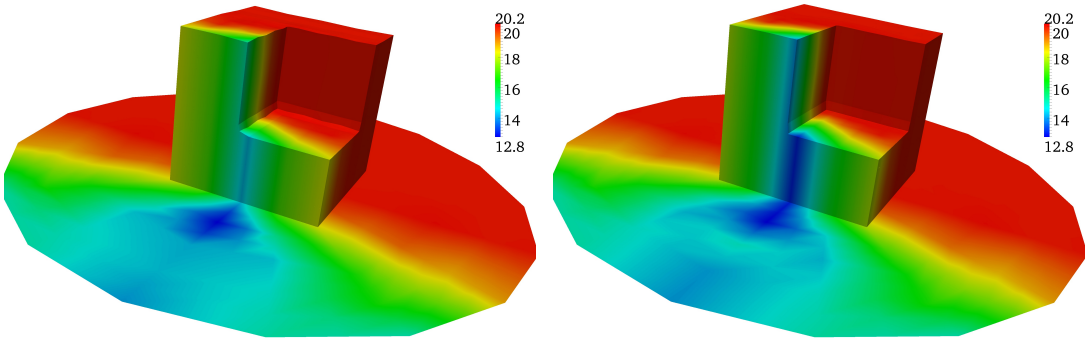
(a) $t = 0$ (b) $t = 3$



(c) $t = 6$ (d) $t = 9$



(e) $t = 12$ (f) $t = 15$



(g) $t = 18$ (h) $t = 20$

Figure 7.7. Temperature at different time points

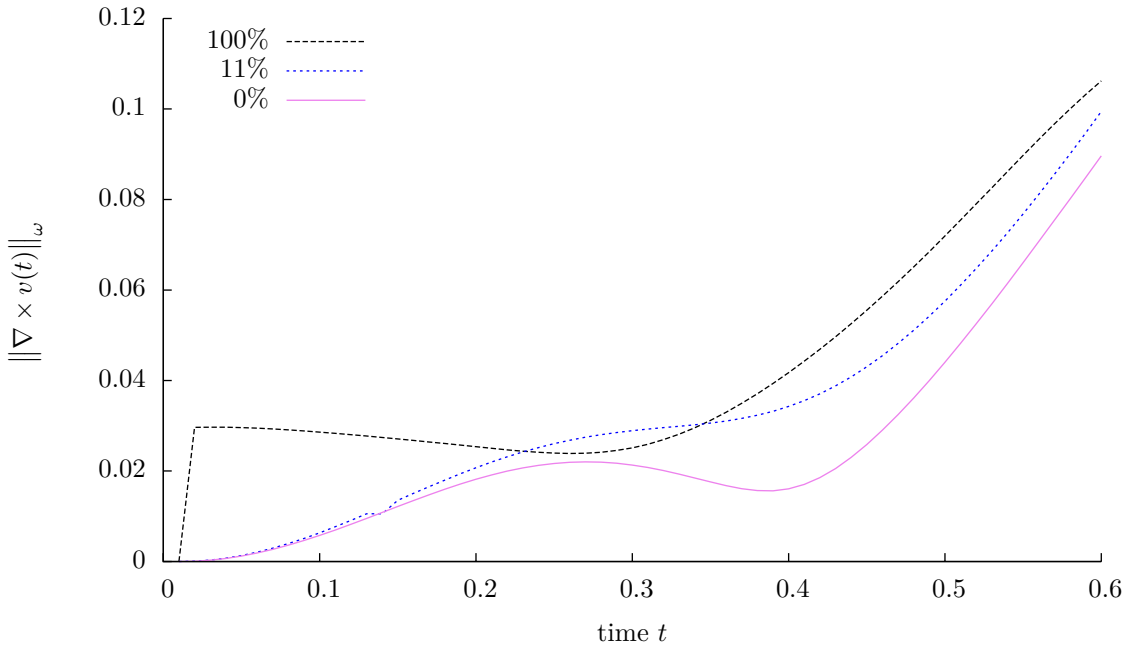


Figure 7.8. Vorticity for various three dimensional covered areas

7.4 Discussion of numerical results

The numerical results we showed in the previous section exhibit that the tracer is transported with a sharp front. Due to the coupling we have a really three dimensional velocity in the inner cube. Also in the temperature and tracer we have a real three dimensional behavior which can not be seen well since the correction in 3D is quite small.

Especially in the Figure 7.8, we clearly see that we get better results if we cover already 11% of the three dimensional area than we get if we only use a two dimensional approximation.

This means the coupling meets the demands we required. For more involved problems the three dimensionality would be more prominent.

An important possibility for further research is to determine the region of three dimensionality adaptively.

8 Conclusions

In this chapter we come to conclusions and give possible directions for future work.

8.1 Conclusion

In this work we developed a method to couple models in different dimensions for simulation of fluid flows in hydrodynamics. These were coupled in such a way that not the whole domain has to be meshed in three dimensions. It is sufficient to solve a cheaper model in two dimensions and to add a three dimensional correction to this two dimensional solution on parts of the domain where a better accuracy is required.

This is a very big step to reduce the computational costs for three dimensional flow problems. In the case that the area where a higher regularity is required, is known, a lot of computation time can be saved by applying this coupling approach.

Likewise, by the development of the multilevel solver for continuous finite element methods, we also save computation time. The direct solvers that are really fast for problems in two dimensions slow down considerably in three dimensions. Therefore it was unavoidable to design a good preconditioner for the fully coupled problem.

8.2 Outlook

The multilevel preconditioner could be extended to different finite elements. So far it can be applied to higher order continuous finite elements and to Nédélec elements.

To this end, we build the multilevel matrices without the coupling terms. If the coupling could be taken into account also in the multilevel matrices, the convergence of the linear solver would be faster.

Still, for the coupling it is not yet possible to adaptively refine the three dimensional mesh. This would be very important to incorporate since this would also reduce the computation time.

As far as the coupling of models in concerned, the area of the three dimensional extension has to be determined beforehand. With some modifications in the code, it should be possible to choose the cells for an extension to three dimensions. Therefore one would need an a posteriori error estimator which estimates the coupling error. Due to the monolithic formulation of the fully coupled problem this should be feasible.

Acknowledgments

First of all I would like to express my gratitude to my supervisor Rolf Rannacher for giving me the opportunity to work on this interesting subject and for the enduring support during the last years.

This work has been supported by the German Research Foundation (DFG) through International Graduiertenkolleg 710 “Complex processes: Modeling, Simulation and Optimization”. As a member of this Graduiertenkolleg and later as a member of the “Heidelberg Graduate School of Mathematical and Computational Methods for the Sciences” I was given the possibility to attend several conferences and to stay abroad in Warsaw and College Station.

My special thanks are addressed to Guido Kanschat for his enduring support, for countless fruitful discussions, as well as for many valuable suggestions concerning my work. Besides, I want to thank him for the productive and agreeable collaboration in software development and paper writing.

Further I would like to thank my office mates Helke and Thomas and the whole Numerical Analysis Group at the University of Heidelberg for always open doors and a nice atmosphere making my time at work enjoyable and unforgettable.

My gratitude goes to the developers of deal.II, who helped me with my programming questions, in particular Guido, Wolfgang, Timo, and Martin.

Vielen lieben Dank möchte ich an dieser Stelle meiner Familie und Stefan sagen, die immer für mich da waren und mich unterstützt haben.

I would like to thank all my friends who encouraged me working and also thanks to all my friends who kept me away from working. I am grateful to all those of you who believed in me even at those times when I did not.

Many thanks to Adrian, Agnieszka, Anna, Anna-Maria, Christian, Cockie, Dominik, Eva, Helke, Jevgeni, Julia, Kathrin, Matthias, Michael, Michael, Tom, Thomas, Thomas, Uli, and Winni.

List of Figures

3.1 Transformation to a fixed domain 16

4.1 Reduction from a 3D cube to a 2D cube with a 3D cube in the middle. 20
4.2 Reference solution of three dimensional Poisson problem 23
4.3 Sequence of adaptively refined meshes after various refinement steps 24
4.4 Coupled solution of the three dimensional Poisson problem obtained for the second test 27

5.1 Discretization in time from t_{n-1} to t_{n+1}. 43
5.2 Discretization via fractional steps from t_{n-1} to t_n. 45
5.3 Crank-Nicolson with $k_n = 0.042$ 47
5.4 Fractional step θ with $k_n = 0.125$ 47
5.5 Fractional step variant with $k_n = 0.083$ 48
5.6 Crank-Nicolson with $k_n = 0.042$ 49
5.7 Fractional step θ with $k_n = 0.125$ 49
5.8 Fractional step variant with $k_n = 0.083$ 50
5.9 Example of active cells in a 2D mesh and corresponding tree graph. 55
5.10 A hierarchy of three meshes with local refinement (active cells shaded). 55
5.11 Splitting of $\mathcal{T}_\ell$ into $\mathcal{T}_\ell^S$ (shaded cells) and $\mathcal{T}_\ell^L$ (white cells). 56
5.12 Violated one-Irregularity on the left and resolved one-irregular mesh on the right. 57
5.13 Violated v-one-Irregularity on the left and resolved v-one-irregular mesh on the right. 57
5.14 Two- and three-dimensional meshes with hanging nodes. 58
5.15 Transformation κ_T from the reference cell $\hat{T}$ to a computational cell T 60
5.16 Position for two 3D cells on a 2D mesh are colored 63
5.17 Sequence of meshes in 3D refined into a corner 64
5.18 Sequence of meshes in 2D refined according to 3D mesh 64
5.19 Sequence of coupled meshes 65

6.1 The subspaces V_ℓ^S (green), V_ℓ^E (yellow), and V_ℓ^L (white). 72
6.2 A mesh hierarchy with the same number of degrees of freedom on each level. Half of the interior edges bear hanging nodes. 76
6.3 Refinement of the positive quadrant, levels 2, 3 and 6, with v-one-irregular closure. 77
6.4 Refinement of a circle, levels 3, 4 and 9, with v-one-irregular closure. 77
6.5 Locally refined meshes in three dimensions. First octant refined to level 4 (left) and ball refined to level 6 (right). 80
6.6 Part of a coupled mesh 82
6.7 Corresponding cells 83

6.8 Coupling of 3D dofs (×) of an active cell to 2D dofs (•) of a possibly inactive cell 84

7.1 Mesh in 2D (blue and red) and mesh in 3D (red) 85
7.2 Coupled mesh in 2D and 3D 86
7.3 Streamlines in x, y, and z-direction at different time points 89
7.4 Streamlines in x, y, and z-direction at different time points, ctd. 90
7.5 Height at different time points 91
7.6 Tracer at different time points 92
7.7 Temperature at different time points 93
7.8 Vorticity for various three dimensional covered areas 94

List of Tables

4.1 Results in the L^2 norm and H^1 semi norm for the Poisson problem computed in three dimensions on a globally refined mesh . . . 24
4.2 Point values for the Poisson problem computed in three dimensions on a globally refined mesh . . . 25
4.3 Results for the Poisson problem computed in three dimensions on an adaptively refined mesh . . . 25
4.4 Point values for the Poisson problem computed in three dimensions on an adaptively refined mesh . . . 26
4.5 Results in the L^2 norm and H^1 semi norm for the coupled Poisson problem on a globally refined mesh obtained for the first test . . . 26
4.6 Point values for the coupled Poisson problem on a globally refined mesh obtained for the first test . . . 27
4.7 Results in the L^2 norm for the coupled Poisson problem on a globally refined mesh obtained for the second test . . . 28
4.8 Results in the H^1 semi norm for the coupled Poisson problem on a globally refined mesh obtained for the second test . . . 28
4.9 Point values for the coupled Poisson problem on a globally refined mesh obtained for the second test . . . 28
4.10 Results in the L^2 norm for the coupled Poisson problem on a globally refined mesh obtained for the third test . . . 29
4.11 Results in the H^1 semi norm for the coupled Poisson problem on a globally refined mesh obtained for the third test . . . 29
4.12 Point values for the coupled Poisson problem on a globally refined mesh obtained for the third test . . . 30

5.1 Choice of parameters to obtain one-step θ-schemes. . . . 51
5.2 Choice of parameters to obtain fractional-step θ-schemes. . . . 52

6.1 Couplings arising from the fully coupled and transformed problem. . . . 68
6.2 Iteration steps and convergence rates for the preconditioned cg method. One symmetric pre- and post-smoothing step on each level with $\mathcal{Q}_1$-elements in two dimensions. . . . 78
6.3 Iteration steps and convergence rates for the preconditioned cg method for $\mathcal{Q}_1$-elements. Variable smoothing with 1 pre- and post-smoothing step on the finest level. . . . 79
6.4 Convergence rates for the preconditioned cg method. Refinement into a circle in 2d. Variable smoothing with 1 pre- and post-smoothing step on the finest level. 79

6.5 Performance of the preconditioned cg method in three dimensions. One symmetric pre- and post-smoothing step on each level. 79
6.6 Performance of the preconditioned cg method in three dimensions. Variable block smoothing with 1 pre- and post-smoothing step on the finest level. 80
6.7 Convergence rates for the preconditioned cg method. Refinement into a ball in 3d. Variable block smoothing with 1 pre- and post-smoothing step on the finest level. 81

List of Algorithms

6.1 Multilevel (V-Cycle) 71
6.2 Multilevel with local smoothing 74
6.3 Multiplicative Schwarz smoother 75

Bibliography

[1] R. A. Adams. *Sobolev spaces*. Pure and applied mathematics ; v. 140. Academic Press, Amsterdam ; Boston, 2nd edition, 2003.

[2] W. Bangerth, R. Hartmann, and G. Kanschat. deal.II — a general purpose object oriented finite element library. *ACM Trans. Math. Softw.* 33(4), 2007. doi: 10.1145/1268776.1268779.

[3] W. Bangerth, R. Hartmann, and G. Kanschat. `deal.II` *Differential Equations Analysis Library, Technical Reference*, 6th edition, 2010. URL `http://www.dealii.org`. First edition 1999.

[4] W. Bangerth and O. Kayser-Herold. Data structures and requirements for *hp* finite element software. *ACM Trans. Math. Softw.* 36(1), pp. 4/1–4/31, 2009.

[5] W. Bangerth and R. Rannacher. *Adaptive Finite Element Methods for Solving Differential Equations*. Birkhäuser, Basel, 2003.

[6] R. E. Bank. *PLTMG: a software package for solving elliptic partial differential equations*. SIAM, Philadelphia, 1998. Users' guide 8.0.

[7] P. Bastian. Load balancing for adaptive multigrid methods. *SIAM J. on Sci. Comput.* 19(4), pp. 1303–1321, 1998. doi:10.1137/S1064827596297562.

[8] P. Bastian and C. Wieners. Multigrid methods on adaptively refined grids. *Computing in Science and Engg.* 8(6), pp. 44–54, 2006. ISSN 1521-9615. doi:http://dx.doi.org/10.1109/MCSE.2006.116.

[9] R. Becker and M. Braack. Multigrid techniques for finite elements on locally refined meshes. *Numer. Linear Algebra Appl.* 7, pp. 363–379, 2000. Special Issue.

[10] M. Berger, M. Aftosmis, and G. Adomavicius. Parallel multigrid on Cartesian meshes with complex geometry. In *Parallel computational fluid dynamics (Trondheim, 2000)*, pp. 283–290. North-Holland, Amsterdam, 2001.

[11] A. F. Blumberg and G. L. Mellor. A description of a three-dimensional coastal ocean circulation model. In *Three-Dimensional Coastal Ocean Models*, edited by N. Heaps, pp. 1–16. American Geophys. Union, 1987.

[12] F. Boyer, C. Lapuerta, S. Minjeaud, and B. Piar. A local adaptive refinement method with multigrid preconditionning illustrated by multiphase flows simulations. Technical report, INRIA a CCSD electronic archive server based on P.A.O.L [http://hal.inria.fr/oai/oai.php] (France), 2008. URL `http://hal.archives-ouvertes.fr/hal-00307186/en/`.

[13] M. Braack, E. Burman, V. John, and G. Lube. Stabilized finite element methods for the generalized oseen problem. *Computer Methods in Applied Mechanics and Engineering* 196(4-6), pp. 853 – 866, 2007.

[14] M. Braack and A. Ern. A posteriori control of modeling errors and discretization errors. *Multiscale Model. Simul.* 1(2), pp. 221–238, 2003.

[15] M. Braack and A. Ern. Adaptive computation of reactive flows with local mesh refinement and model adaptation. In *Proc. ENUMATH 2003, the 5th European conference on numerical mathematics and advanced applications, Prague, Czech Republic*, edited by M. Feistauer, V. Dolejsí, P. Knobloch, and K. Najzar, pp. 159–168. Springer, 2004.

[16] D. Braess. *Finite Elemente.* Springer-Verlag Berlin Heidelberg, Berlin, Heidelberg, vierte, überarbeitete und erweiterte edition, 2007.

[17] J. H. Bramble. *Multigrid Methods.* Number 294 in Pitman research notes in mathematics series. Longman Scientific, 1993.

[18] J. H. Bramble and J. E. Pasciak. The analysis of smoothers for multigrid algorithms. *Math. Comput.* 58(198), pp. 467–488, 1992.

[19] A. Brandt. Multi-level apative solutions to boundary-value problems. *Math. Comput.* 31(138), pp. 333–390, 1977.

[20] S. C. Brenner and L. R. Scott. *The Mathematical Theory of Finite Element Methods.* Springer, 2nd edition, 2002.

[21] S. C. Brenner and L. R. Scott. *The mathematical theory of finite element methods.* Number 15 in Texts in applied mathematics ; 15 ; Texts in applied mathematics. Springer, New York, NY, 3rd edition, 2008.

[22] A. N. Brooks and T. J. Hughes. Streamline upwind/petrov-galerkin formulations for convection dominated flows with particular emphasis on the incompressible navier-stokes equations. *Computer Methods in Applied Mechanics and Engineering* 32(1-3), pp. 199 – 259, 1982.

[23] A. Caboussat. *Analysis and Numerical Simulation of Free Surface Flows.* Ph.D. thesis, Universität Lausanne, 2003.

[24] J. M. Cascon, C. Kreuzer, R. H. Nochetto, and K. G. Siebert. Quasi-optimal convergence rate for an adaptive finite element method. *SIAM J. Numer. Anal.* 46(5), pp. 2524–2550, 2008. ISSN 0036-1429. doi:10.1137/07069047X.

[25] A. Chorin and J. Marsden. *A mathematical introduction to fluid mechanics.* Texts in applied mathematics. Springer-Verlag, 1990.

[26] P. G. Ciarlet. *The finite element method for elliptic problems.* Studies in mathematics and its applications ; v. 4. North-Holland Pub. Co ; sole distributors for the U.S.A. and Canada, Elsevier North-Holland, Amsterdam ; New York ; New York, 1978.

[27] P. G. Ciarlet. *The Finite Element Method for Elliptic Problems.* North-Holland, 1978.

[28] R. Dautray and J.-L. Lions. Evolution problems i. In *Mathematical Analysis and Numerical Methods for Science and Technology*. Springer-Verlag, 1992.

[29] A. Decoene and J. F. Gerbeau. Sigma transformation and ale formulation for three-dimensional free surface flows. *International Journal for Numerical Methods in Fluids* 59, pp. 357–386, 2009.

[30] Delft3D. `http://delftsoftware.wldelft.nl/`.

[31] T. Dunne. *Adaptive Finite Element Approximation of Fluid-Structure Interaction Based on Eulerian and Arbitrary Lagrangian-Eulerian Variational Formulations*. Ph.D. thesis, Universität Heidelberg, 2007.

[32] H. Elman, D. Silvester, and A. Wathen. *Finite elements and fast iterative solvers: with applications in incompressible fluid dynamics*. Numerical mathematics and scientific computation. Oxford University Press, 2005.

[33] V. Girault and P.-A. Raviart. *Finite element methods for Navier-Stokes equations*. Number 5 in Springer series in computational mathematics ; 5 ; Springer series in computational mathematics. Springer, Berlin ; Heidelberg [u.a.], 1986.

[34] M. Griebel and P. Oswald. On the abstract theory of additive and multiplicative schwarz algorithms. *Numer. Math.* 70, pp. 163–180, 1995.

[35] W. Hackbusch. *Multi-grid Methods and Applications*. Springer, 1985.

[36] J.-M. Hervouet. *Hydrodynamics of free surface flows, modelling with the finite element method*. Wiley, 2007.

[37] J. Hron and S. Turek. A monolithic fem/multigrid solver for ALE formulation of fluid structure interaction with application in biomechanics. In *Fluid-Structure Interaction - Modelling, Simulation, Optimization*, edited by H. Bungartz and M. Schäfer, number 53 in Lecture Notes in Computational Science and Engineering, pp. 146–170. Springer, 2006. ISBN 3-540-34595-7.

[38] J. A. Jankowski. *A non-hydrostatic model for free surface flows*. Ph.D. thesis, Universität Hannover, 1999.

[39] B. Janssen and G. Kanschat. Adaptive multilevel methods with local smoothing for H^1- and H^{curl}-conforming high order finite element methods. *SIAM J. Sci. Comput* 33(4), 2011.

[40] B. Janssen and T. Wick. Block preconditioning with schur complements for monolithic fluid-structure interactions. In *V. European Conference on Computational Fluid Dynamics ECCOMAS CFD 2010*, edited by J. Pereira and A. Sequeira. 2010.

[41] A. C. Jones. *A Projected Multigrid Method for the Solution of Nonlinear Finite Element Problems on Adaptively Refined Grids*. Dissertation, The University of Leeds, 2005.

[42] A. C. Jones and P. K. Jimack. An adaptive multigrid tool for elliptic and parabolic systems. *Int. J. Numer. Meth. Fluids* 47, pp. 1123–1128, 2005.

[43] G. Kanschat. *Parallel and Adaptive Galerkin Methods for Radiative Transfer Problems*. Dissertation, Universität Heidelberg, 1996. URL `http://archiv.ub.uni-heidelberg.de/volltextserver/volltexte/2006/6331/`. Preprint SFB 359, 1996-29.

[44] G. Kanschat. Multi-level methods for discontinuous Galerkin FEM on locally refined meshes. *Comput. & Struct.* 82(28), pp. 2437–2445, 2004. doi:10.1016/j.compstruc.2004.04.015.

[45] S. F. McCormick. *Multilevel Adaptive Methods for Partial Differential Equations*, volume 6 of *Frontiers in Applied Mathematics*. SIAM, Philadelphia, 1989.

[46] E. Miglio, S. Perotto, and F. Saleri. Model coupling techniques for free-surface flow problems: Part i. *Nonlinear Analysis* 63(5–7), pp. e1885–e1896, 2005.

[47] E. Miglio, S. Perotto, and F. Saleri. Model coupling techniques for free-surface flow problems: Part ii. *Nonlinear Analysis* 63(5–7), pp. e1897–e1908, 2005.

[48] W. F. Mitchell. Parallel adaptive multilevel methods with full domain partitions. 2003.

[49] S. Müller-Urbaniak. *Eine Analyse des Zwischenschritt-θ-Verfahrens zur Lösung der instationären Navier-Stokes-Gleichungen*. Ph.D. thesis, Universität Heidelberg, 1994.

[50] F. Nobile. *Numerical approximation of fluid-structure interaction problems with application to haemodynamics*. Ph.D. thesis, Lausanne, 2001. URL `http://library.epfl.ch/theses/?nr=2458`.

[51] J. Pedlosky. *Geophysical fluid dynamics*. Springer study edition. Springer-Verlag, 1987.

[52] S. Perotto. Adpative modeling for free-surface flows. *M2AN Math. Model. Numer. Anal.* 40(3), pp. 469–499, 2006.

[53] N. A. Phillips. A coordinate system having some special advantages for numerical forecasting. *Journal of Meteorology* 14, pp. 184–185, 1957.

[54] E. Rank, H. Bröker, A. Düster, R. Krause, and M. Rücker. The p-version of the Finite Element Method for Structural Problems. In *Error-controlled Adaptive Finite Elements in Solid Mechanics*, edited by E. Stein, pp. 263–308. Wiley, 2003.

[55] R. Rannacher. Numerical analysis of nonstationary fluid flow (a survey). Technical Report 492, Institut für Angewandte Mathematik, Universität Heidelberg, 1988.

[56] T. Richter and T. Wick. Finite elements for fluid-structure interaction in ALE and fully Eulerian coordinates. *Computer Methods in Applied Mechanics and Engineering* 199(41–44), pp. 2633 – 2642, 2010.

[57] M.-C. Rivara. Design and data structure of fully adaptive, multigrid, finite-element software. *ACM Trans. Math. Softw.* 10(3), pp. 242–264, 1984. ISSN 0098-3500. doi: http://doi.acm.org/10.1145/1271.1274.

[58] U. Rüde. *Mathematical and Computational Techniques for Multilevel Adaptive Methods*, volume 13 of *Frontiers in Applied Mathematics*. SIAM, Philadelphia, 1993.

[59] Y. Saad. *Iterative Methods for Sparse Linear Systems*. Oxford University Press, 2nd edition, 2000.

[60] R. S. Sampath and G. Biros. A parallel geometric multigrid method for finite elements on octree meshes. *SIAM J. Sci. Comput.* 32(3), pp. 1361–1392, 2010. doi:10.1137/090747774.

[61] A. Schmidt and K. Siebert. *Design of Adaptive Finite Element Software - The Finite Element Toolbox ALBERTA*, volume 42 of *Lecture Notes in Computational Science and Engineering*. Springer, Heidelberg, 2005.

[62] P. Šolín, J. Červený, and I. Doležel. Arbitrary-level hanging nodes and automatic adaptivity in the *hp*-FEM. *Math. Comput. Simulation* 77(1), pp. 117–132, 2008. ISSN 0378-4754. doi:10.1016/j.matcom.2007.02.011.

[63] Telemac. `http://www.telemacsystem.com/`.

[64] R. Temam. *Navier-Stokes equations*. AMS Chelsea Publ., Providence, RI, reprinted with corr. edition, 2001.

[65] U. Trottenberg, C. W. Oosterlee, and A. Schüller. *Multigrid*. Academic Press, London, 2001.

[66] S. Turek. *Efficient Solvers for Incompressible Flow Problems: An Algorithmic and Computational Approach*. Springer, Berlin, 1999.

[67] S. Turek, L. Rivkind, J. Hron, and R. Glowinski. Numerical analysis of a new time–stepping theta–scheme for incompressible flow simulations. Technical report, Fakultät für Mathematik, TU Dortmund, 2005. Ergebnisberichte des Instituts für Angewandte Mathematik, Nummer 282.

[68] R. S. Varga. *Matrix Iterative Analysis*. Series in Computational Mathematics. Springer, 2nd edition, 1999.

[69] R. Verfürth. *A Review of A Posteriori Error Estimation and Adaptive Mesh-Refinement Techniques*. John Wiley/Teubner, 1996.

[70] C. von Rohden, A. Hauser, K. Wunderle, J. Ilmberger, G. Wittum, and K. Roth. Lake dynamics: Observation and high-resolution numerical simulation. In *Reactive Flows, Diffusion and Transport, from experiments via mathematical modeling to numerical simulation and optimzation*, edited by W. Jäger, R. Rannacher, and J. Warnatz, pp. 559–581. Springer, 2006.

[71] C. Vreugdenhil. *Numerical methods for shallow-water flow*. Water science and technology library. Kluwer Academic Publishers, 1994.

[72] J. Wloka. *Partielle Differentialgleichungen*. Mathematische Leitfäden. Teubner, Stuttgart, 1982.

[73] J. Xu. Iterative methods by space decomposition and subspace correction. *SIAM Review* 34(4), pp. 581–613, 1992.