NTNU

Norwegian University of
Science and Technology

Numerical solution of partial differential equations in time-dependent domains

Øystein Tråsdahl

Master of Science in Physics and Mathematics
Submission date: June 2008
Supervisor: Einar Rønquist, MATH

Norwegian University of Science and Technology
Department of Mathematical Sciences

Problem Description

The governing equations for heat transfer and fluid flow problems in time-dependent domains will be reviewed. In particular, the Arbitrary Lagrangian Eulerian formulation will be used in the formulation and the numerical treatment of such problems. Selected model problems will be solved numerically using a spectral method in space combined with a semi-implicit multi-step method in time.

Assignment given: 24. January 2008
Supervisor: Einar Rønquist, MATH

Abstract

Numerical solution of heat transfer and fluid flow problems in two spatial dimensions is studied. An arbitrary Lagrangian-Eulerian (ALE) formulation of the governing equations is applied to handle time-dependent geometries. A Legendre spectral method is used for the spatial discretization, and the temporal discretization is done with a semi-implicit multi-step method. The Stefan problem, a convection-diffusion boundary value problem modeling phase transition, makes for some interesting model problems. One problem is solved numerically to obtain first, second and third order convergence in time, and another is used to illustrate the difficulties that may arise with distribution of computational grid points in moving boundary problems. Strategies to maintain a favorable grid configuration for some particular geometries are presented. The Navier-Stokes equations are more complex and introduce new challenges not encountered in the convection-diffusion problems. They are studied in detail by considering different simplifications. Some numerical examples in static domains are presented to verify exponential convergence in space and second order convergence in time. A preconditioning technique for the unsteady Stokes problem with Dirichlet boundary conditions is presented and tested numerically. Free surface conditions are then introduced and studied numerically in a model of a droplet. The fluid is modeled first as Stokes flow, then Navier-Stokes flow, and the difference in the models is clearly visible in the numerical results. Finally, an interesting problem with non-constant surface tension is studied numerically.

Contents

Introduction

The spectral or high-order method [9, 11, 28] is closely related to the finite element method in that it uses a Galerkin approach, based on a weak formulation of the problem. It involves seeking a solution to a differential equation in a discrete space of smooth functions, expressing the solution in terms of a truncated series of basis functions. If the discrete space and the basis are well chosen, then the problem can be solved relatively fast and with errors that are sub-dominant the errors caused by the discretization. Hence, the quality of the solution is determined by the quality of the approximation space. The most commonly chosen discrete spaces are those of trigonometric functions or high order polynomials, depending on the problem at hand. In this paper we will use Legendre polynomials for the approximation. If the exact solution and the geometry are analytic functions, we can achieve *exponential convergence*, i.e. the error will decay faster than any algebraic power of the polynomial degree N [23].

The Arbitrary Lagrangian-Eulerian (ALE) formulation provides a powerful framework for studying partial differential equations on time-dependent domains. The approach gathers the best from the Lagrangian and Eulerian frameworks, enabling proper description of the boundary, while preventing mesh distortions. In the ALE formulation we operate with a *domain velocity*, which may differ from the particle velocity. The domain velocity must satisfy the *kinematic condition*, which requires that the normal component equals the normal component of the particle velocity on the boundary of the computational domain.

The ALE formulation finds many applications in solid and fluid mechanics, and is an area of active research [15]. In order to study the ALE framework, it is convenient to start with simpler equations than the Navier-Stokes equations. Here, the convection-diffusion equation will be used as a starting point for studying the framework. It is a simpler equation, but still shows all the aspects of the ALE formulation that we will encounter with the Navier-Stokes equations. An interesting example of a convection-diffusion boundary value problem is the Stefan problem, in which the heat flux across the boundary is used to determine the movement of the domain in the direction of the surface normal vector. The Stefan problem models phase transition in heat transfer problems, typically between solid and liquid state, i.e. melting or freezing problems.

When solving a Stefan problem on a finite or spectral element grid, the Stefan condition determines how the computational grid points on the boundary should move in the direction of the normal vector. The movement in the tangential direction, however, is not subject to any condition, so we are given a certain freedom of choice. There is no algorithm for determining the optimal choice;

we do not even have a way of defining what an optimal choice should imply in this context. But a good goal is to try to retain the regularity of the mapping from the reference domain to the physical domain after the domain is updated. This is done in practice by imposing certain Dirichlet or Neumann boundary conditions on the domain velocity.

Spectral methods have evolved to be applicable to a wide range of fluid dynamics problems. Many of the most frequently recurring fluids in engineering applications (e.g. water) can be modeled as *incompressible Newtonian flow*, i.e. flow with a constant viscosity. This type of flow is properly modeled by the Navier-Stokes equations, which describe conservation of mass and linear momentum for viscous fluid flow. Numerical treatment of the Navier-Stokes equations is not trivial. There are complicating factors which make them a lot more difficult to handle than the convection-diffusion equation. However, the Navier-Stokes equations can be simplified in various ways, for example by assuming steady flow or infinite viscosity. The way we will study the equations in this paper is by starting with only the viscous term, which is the differential operator of highest order, and then gradually expanding the equations, adding new terms. Each new term will introduce new challenges that will be discussed.

The Navier-Stokes equations can be accompanied by Dirichlet or Neumann boundary conditions and solved on a time-independent domain. To study the equations in time-dependent domains, we will add a free surface condition once we arrive at the Stokes problem. The surface tension allows the surface to deform elastically, and the free surface condition requires that there is no shear stress on the surface at equilibrium. The addition of free surface conditions necessitates an ALE formulation, which will be presented.

Due to the complexity of the equations, free surface problems with known exact solutions are difficult to construct, and very few are available. We will study a problem for the which the steady state solution is known, but the evolution from the initial state is unknown. The problem models a drop of liquid with free surface and constant surface tension in an environment free of gravitational forces. The only external force is the ambient pressure, which is constant. For this problem the equilibrium, at which the surface area is minimized, is a circular drop with no particle velocity and constant pressure. This problem will be modeled both as a Stokes problem and as a Navier-Stokes problem. The unsteady Stokes model will make the drop deform slowly towards the steady state, while the unsteady Navier-Stokes model will make it oscillate, behaving more elastically as the viscosity decreases and the inertial effects become more important. The effect of the dimensionless Capillary number on the oscillations will be studied.

Finally, the problem of the liquid drop will be modeled with non-constant surface tension. The surface tension must be everywhere positive, and periodic along the surface. We will study a problem where the surface tension varies linearly with one of the spatial coordinates. Now the steady state is unknown. It seems reasonable to anticipate some sort of vorticity. We will use the steady state solution from the constant surface tension problem as an initial condition and see if numerical simulations lead to a steady state.

This structure of this paper reflects the way the work has been done. The material is divided into chapters, organized by the equation being studied. Each chapter builds on the experience from the previous chapters, and only novel concepts are discussed in detail. The framework for both the spectral method and the ALE formulation is presented in the first chapter, using the convection-diffusion equation as a framework. In the second chapter, mesh update techniques are introduced and studied for the Stefan problem. This chapter also contains a brief discussion of the multi-element approach, as we move from solving problems on a single element to solving problems on multiple elements. This transition is motivated by the need for studying problems on circular domains. The last chapter covers the entire evolution towards the Navier-Stokes equations, from the crude simplification involving only the viscous term, to the full Navier-Stokes equations with free surface boundary conditions.

Numerical examples are included for every new problem or aspect introduced. The examples are all two-dimensional, and most of the problems are just constructed to illustrate matters in the text. In the examples where the solution is known a priori, convergence properties of the numerical methods are studied. All numerical simulations are done in MATLAB, and no extra software packages are used. All code is written from scratch by the author, apart from a few simple functions, which were part of the material in the course MA8502.

The reader is assumed to have a knowledge of the basic ingredients of spectral element methods, such as weak formulation and discretization, although these are briefly covered in the text. A mathematics student on Master's thesis level should be able to follow.

Chapter 1

The convection-diffusion equation

In this chapter we will use the unsteady convection-diffusion equation as a basis for investigating different aspects of numerical treatment of partial differential equations in time-dependent domains. This equation is a convenient starting point since it is linear, and exact solutions are easy to construct in two dimensions. At the same time it is useful, since it properly models many heat flow problems.

The arbitrary Lagrangian-Eulerian (ALE) formulation will be used to handle the challenges of the time-dependent geometry. It is based on the weak form of the problem and results in a formulation that is numerically tractable and allows for accurate interface-tracking. A semi-discrete system will then be derived, using a Legendre spectral method. The resulting system of ordinary differential equations is readily solvable with the conjugate gradient algorithm.

1.1 Strong and weak formulation

The strong form of the unsteady convection-diffusion problem reads:

$$\begin{aligned} \frac{\partial \varphi}{\partial t} - \nabla^2 \varphi + \boldsymbol{U} \cdot \nabla \varphi &= f && \text{in } \Omega \subset \mathbb{R}^2, \\ \varphi &= 0 && \text{on } \partial\Omega, \\ \varphi &= \varphi_0(x, y) && \text{at } t = 0. \end{aligned} \tag{1.1}$$

Here $\boldsymbol{U}$ is a prescribed two-dimensional convection field, and f is a given volumetric heat source. Homogeneous Dirichlet boundary conditions are given for simplicity, but they may be partially or fully exchanged with inhomogeneous Dirichlet boundary conditions or Neumann boundary conditions.

Before stating the weak form, we introduce the spaces

$$\begin{aligned} X &= \{v \in H^1(\Omega) \bigm| v = 0 \text{ on } \partial\Omega\} \equiv H_0^1(\Omega), \\ Y(X) &= \{v \bigm| \forall t \in [0, T],\ v(x, y; t) \in X,\ \int_0^T \|v\|^2_{H^1(\Omega)} \,\mathrm{d}t < \infty\}. \end{aligned}$$

The weak form is attained by multiplying with a test function and integrating. Integration by parts of the diffusion term and application of the boundary conditions yields the following problem: find $\varphi \in Y(X)$ such that

$$\int_\Omega v \frac{\partial \varphi}{\partial t} \,\mathrm{d}\Omega + \int_\Omega \nabla\varphi \cdot \nabla v \,\mathrm{d}\Omega + \int_\Omega v \boldsymbol{U} \cdot \nabla\varphi \,\mathrm{d}\Omega \;=\; \int_\Omega f\, v \,\mathrm{d}\Omega \qquad \forall v \in X \quad (1.2)$$

1.2 ALE framework

The Lagrangian and Eulerian descriptions of a moving computational mesh each have their benefits and problems. The Lagrangian description, in which each mesh node follows a material particle in its motion, is great for tracking free surfaces and interfaces between different materials. Its weakness is its inability to follow large distortions without recourse to frequent re-meshing operations. The Eulerian description uses a fixed computational grid and is therefore able to handle large distortion with relative ease. On the other hand, a fixed grid means a fixed resolution in space, limiting the accuracy in surface or interface descriptions and resolution of flow details.

The Arbitrary Lagrangian-Eulerian (ALE) description is designed to overcome these shortcomings, allowing the mesh nodes to move independently of the fluid particles. It combines features of both the Lagrangian and the Eulerian descriptions, since the mesh nodes may move along with the material particles, or be held fixed. The motion of the computational mesh is described by the *domain velocity*, which we denote $\boldsymbol{w}$. This needs not be that same as the *particle velocity*, here denoted $\boldsymbol{u}$. The only requirement is that the computational grid must follow the surface of the computational domain. The tangential movement of the mesh points on the surface, however, is not subject to any condition, so here we have significant freedom. Typically, we choose a compromise between Lagrangian and Eulerian grid motion in order to keep a favorable grid configuration.

The requirement that the surface of the computational domain follows the material surface results in the so-called *kinematic condition*. It states that the normal components of the particle velocity $\boldsymbol{u}$ and the domain velocity $\boldsymbol{w}$ must be equal,

$$\boldsymbol{u} \cdot \boldsymbol{n} = \boldsymbol{w} \cdot \boldsymbol{n}, \qquad (1.3)$$

as illustrated in Figure 1.1. The surface must move along with the particles that reside on the surface. We have implicitly assumed a certain regularity of the surface, namely that particles on the surface remain on the surface, and that no particles “switch places”.

The starting point for the ALE formulation is the weak form (1.2). The first integral contains a time derivative inside the integral, and is therefore difficult to evaluate numerically. The time derivative can be moved outside the integral using Reynold’s transport theorem [1]. It states that for any property φ of the material (e.g. mass, momentum or energy) in a moving domain,

$$\frac{\mathrm{d}}{\mathrm{d}t} \int_\Omega \varphi \,\mathrm{d}\Omega = \int_\Omega \frac{\partial \varphi}{\partial t} \,\mathrm{d}\Omega + \int_{\partial\Omega} \varphi\, \boldsymbol{u} \cdot \boldsymbol{n} \,\mathrm{d}S, \qquad (1.4)$$

where $\boldsymbol{u}$ is the material velocity and $\boldsymbol{n}$ is the unit normal on the surface $\partial\Omega$. In order to be able to apply Reynold’s transport theorem, we first have to integrate

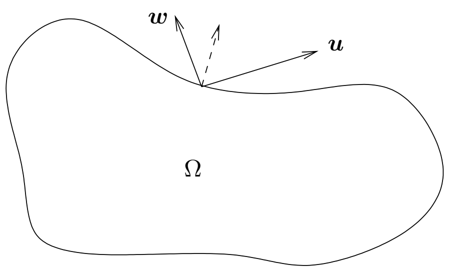


Figure 1.1: *The kinematic condition requires that the normal component of the particle velocity $\boldsymbol{u}$ equals the normal component of the domain velocity $\boldsymbol{w}$ on the surface.*

by parts

$$\begin{aligned}\int_\Omega v\frac{\partial\varphi}{\partial t}\,\mathrm{d}\Omega &= \int_\Omega \frac{\partial(v\varphi)}{\partial t}\,\mathrm{d}\Omega - \int_\Omega \varphi\frac{\partial v}{\partial t}\,\mathrm{d}\Omega \\ &= \frac{\mathrm{d}}{\mathrm{d}t}\int_\Omega v\,\varphi\,\mathrm{d}\Omega - \int_{\partial\Omega} v\varphi\,\boldsymbol{u}\cdot\boldsymbol{n}\,\mathrm{d}S - \int_\Omega \varphi\frac{\partial v}{\partial t}\,\mathrm{d}\Omega. \qquad (1.5)\end{aligned}$$

Now the last term contains time derivative of the test function. This can be replaced by a convective term, due to the fact that the derivative of the test function convected with the domain velocity is zero:

$$\frac{Dv}{Dt} = \frac{\partial v}{\partial t} + \boldsymbol{w}\cdot\nabla v = 0.$$

Hence the last integral can be written

$$\int_\Omega \varphi\frac{\partial v}{\partial t}\,\mathrm{d}\Omega = -\int_\Omega \varphi\,\boldsymbol{w}\cdot\nabla v\,\mathrm{d}\Omega.$$

The second term on the right hand side of (1.5) contains the normal component of the material velocity. We apply the the kinematic condition to replace this with the normal component of the domain velocity. We also apply the divergence theorem, and the term becomes

$$\int_{\partial\Omega} v\,\varphi\,\boldsymbol{u}\cdot\boldsymbol{n}\,\mathrm{d}S = \int_\Omega \nabla\cdot(v\varphi\boldsymbol{w})\,\mathrm{d}\Omega.$$

Now expanding the spatial derivatives of the product in the last integrand, we get three terms. Inserting this and the rest of (1.5) into the weak form (1.2), we arrive at

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t}\int_\Omega v\,\varphi\,\mathrm{d}\Omega - \int_\Omega \varphi\,\boldsymbol{w}\cdot\nabla v\,\mathrm{d}\Omega - \int_\Omega v\,\boldsymbol{w}\cdot\nabla\varphi\,\mathrm{d}\Omega - \int_\Omega v\varphi\,\nabla\cdot\boldsymbol{w}\,\mathrm{d}\Omega \\ + \int_\Omega \varphi\,\boldsymbol{w}\cdot\nabla v\,\mathrm{d}\Omega + \int_\Omega \nabla\varphi\cdot\nabla v\,\mathrm{d}\Omega + \int_\Omega v\,\boldsymbol{U}\cdot\nabla\varphi\,\mathrm{d}\Omega = \int_\Omega f\,v\,\mathrm{d}\Omega.\end{aligned}$$

The second and the fifth term on the left hand side are equal and cancel out. Thus, we arrive at the following ALE formulation, written in abstract form: find

$\varphi \in Y(X)$ such that

$$\frac{\mathrm{d}}{\mathrm{d}t}(v,\varphi) + a(v,\varphi) + c(v,\varphi) - e(v,\varphi) = (v,f) \qquad \forall v \in X, \tag{1.6}$$

where $(\cdot,\cdot)$ is the usual L^2 inner product and

$$\begin{aligned} a(v,\varphi) &= \int_\Omega \nabla\varphi \cdot \nabla v \,\mathrm{d}\Omega, \\ c(v,\varphi) &= \int_\Omega v\,(\boldsymbol{U} - \boldsymbol{w}) \cdot \nabla\varphi \,\mathrm{d}\Omega, \\ e(v,\varphi) &= \int_\Omega v\varphi\nabla \cdot \boldsymbol{w} \,\mathrm{d}\Omega. \end{aligned} \tag{1.7}$$

Comparing the ALE formulation to the weak form (1.2), we see that both the right hand side and the diffusion term remain unchanged. The convection term is modified by the domain velocity, such that the effective convection field now is the difference between the prescribed convection field and the domain velocity. A completely new term has also appeared, an "expansion term" containing the divergence of the domain velocity. Last but not least, the time derivative is no longer a partial derivative, since integration is performed before taking the derivative. This greatly simplifies time integration, and makes the ALE formulation a more suitable starting point for numerical approximation.

1.3 Spatial discretization

The method that will be applied throughout this paper is a Legendre spectral method. The solution is approximated using high order polynomials on a reference domain, and then transformed to the physical domain. The reference domain is the square $\widehat{\Omega} = ((-1,1))^2$, and it is mapped to the physical domain by a one-to-one mapping $\mathcal{F} : \widehat{\Omega} \to \Omega$. A point (x, y) in the physical domain is

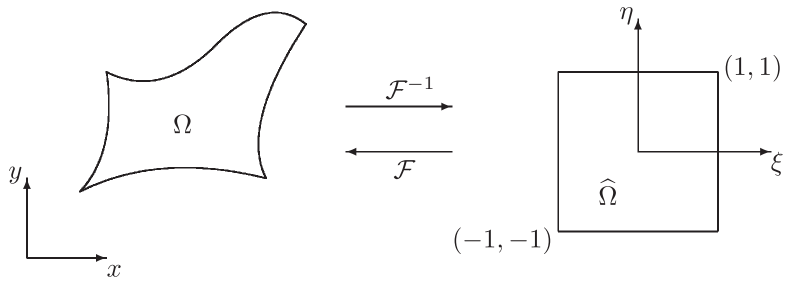


Figure 1.2: *$\mathcal{F}$ is the mapping from the reference domain to the deformed domain.*

uniquely determined by the mapping

$$\begin{aligned} x &= x(\xi,\eta), \\ y &= y(\xi,\eta), \end{aligned}$$

where ξ and η are the coordinates in the reference domain. A function v in the physical domain can be transformed to the reference domain by composition,

$$v(x,y) = (v \circ \mathcal{F})(\xi,\eta) = \hat{v}(\xi,\eta), \tag{1.8}$$

such that $\hat{v}$ is a function in the reference variables. The hat notation will be used to separate functions in the reference variables from functions in the physical variables.

We now introduce a discrete space of functions which are polynomials in the reference variables:

$$X_N = \{v \in H_0^1(\Omega) \bigm| (v \circ \mathcal{F}) \in \mathbb{P}_N(\widehat{\Omega})\}.$$

The dimension of this space is $(N-1)^2$, since a polynomial of degree N needs $N+1$ parameters to be uniquely determined, but two degrees of freedom are lost in each spatial direction due to the boundary conditions. The discrete space in which we will seek a solution is then

$$Y(X_N) = \{v \mid \forall t \in [0,T],\ v(x,y;t) \in X_N,\ \int_0^T \|v\|^2_{H^1(\Omega)}\,\mathrm{d}t < \infty\}.$$

The discretization is based on the ALE formulation (1.6), and the discrete problem reads: find $\varphi_N \in Y(X_N)$ such that

$$\frac{\mathrm{d}}{\mathrm{d}t}(v,\varphi_N)_N + a_N(v,\varphi_N) + c_N(v,\varphi_N) - e_N(v,\varphi_N) = (v,f)_N \qquad \forall v \in X_N \tag{1.9}$$

The subscript indicates that the integrals in (1.7) and the inner products will be evaluated using Gauss-Lobatto-Legendre (GLL) quadrature. The quadrature is applied to integrals over the reference domain, so integrals over Ω must be transformed to integrals over $\widehat{\Omega}$. Integrating a function $v \in L^2(\Omega)$ over Ω, the transformation is simply

$$\int_\Omega v\,\mathrm{d}\Omega = \int_{\widehat{\Omega}} \hat{v}\,J\,\mathrm{d}\widehat{\Omega},$$

where J is the determinant of the Jacobian of the mapping $\mathcal{F} : \widehat{\Omega} \to \Omega$, given by

$$J = \frac{\partial x}{\partial \xi}\frac{\partial y}{\partial \eta} - \frac{\partial x}{\partial \eta}\frac{\partial y}{\partial \xi}.$$

GLL quadrature with $N+1$ quadrature points is then preformed as a weighted sum in each spatial direction,

$$\int_{\widehat{\Omega}} \hat{v} J\,\mathrm{d}\widehat{\Omega} = \sum_{\alpha=0}^{N}\sum_{\beta=0}^{N} \rho_\alpha \rho_\beta\, \hat{v}(\xi_\alpha,\xi_\beta) J(\xi_\alpha,\xi_\beta),$$

where ξ_α are the quadrature points and ρ_α the associated quadrature weights (both are the same in each spatial directions). With $N+1$ quadrature points, we are able to integrate exactly all polynomials up to degree $2N-1$ in each spatial direction [8].

The domain Ω is approximated using high order polynomials. We will use an *isoparametric* approach, approximating the geometry with the same order polynomials as the numerical solution φ_N. In particular, the computational mesh points are placed at the GLL points ξ_α in each spatial direction in the reference domain, which is then transformed to the physical domain.

We will use a nodal basis for the discrete space X_N. The basis functions $\hat{v}_{ij}(\xi,\eta)$ are products of Lagrangian interpolants through the nodal points in both spatial directions,

$$\hat{v}_{ij}(\xi,\eta) = \ell_i(\xi)\,\ell_j(\eta), \qquad 1 \le i,j \le N-1.$$

Note the range of the indices. Due to the Dirichlet boundary conditions there are no test functions associated with the nodal points on the boundary. The Lagrangian interpolants are polynomials of degree N with the property that

$$\ell_i(\xi_\alpha) = \delta_{i\alpha},$$

where $\delta_{i\alpha}$ is the Kronecker delta. Hence, when the solution $\widehat{\varphi}_N$ (i.e., expressed in the reference variables) is expanded in this basis, the basis coefficients φ_{mn} equal the function values in the GLL points:

$$\begin{aligned}\widehat{\varphi}_N(\xi_\alpha,\xi_\beta) &= \sum_{m=0}^{N}\sum_{m=0}^{N} \varphi_{mn}\,\ell_m(\xi_\alpha)\,\ell_n(\xi_\beta)\\ &= \sum_{m=0}^{N}\sum_{m=0}^{N} \varphi_{mn}\,\delta_{m\alpha}\,\delta_{n\beta}\\ &= \varphi_{\alpha\beta}, \qquad 0 \le \alpha,\beta \le N.\end{aligned}$$

Note that these are also the function values in the corresponding nodal points on the computational grid on the physical domain.

Each of the basis functions is inserted as a test function into the discrete problem (1.9). Consider for example the inner product on the right hand side. Inserting the test function $\hat{v}_{ij}(\xi,\eta)$ and applying the GLL quadrature, we get

$$\begin{aligned}\int_\Omega f\; v_{ij}\;\mathrm{d}\Omega &= \int_{\widehat{\Omega}} \hat{f}\;\hat{v}_{ij}\;J\,\mathrm{d}\widehat{\Omega}\\ &= \int_{-1}^{1}\int_{-1}^{1} \hat{f}(\xi,\eta)\;\ell_i(\xi)\,\ell_j(\eta)\;J(\xi,\eta)\,\mathrm{d}\xi\,\mathrm{d}\eta\\ &= \sum_{\alpha=0}^{N}\sum_{\beta=0}^{N} \rho_\alpha\rho_\beta\;\hat{f}(\xi_\alpha,\xi_\beta)\;\delta_{i\alpha}\delta_{j\beta}\;J(\xi_\alpha,\xi_\beta)\\ &= \rho_i\rho_j\;J(\xi_i,\xi_j)\;\hat{f}(\xi_i,\xi_j), \quad 1 \le i,j \le N-1.\end{aligned}$$

Let $\boldsymbol{f} \in \mathbb{R}^{(N-1)^2}$ be a vector containing the function values $\hat{f}(\xi_i,\xi_j)$ in the internal nodal points, sampled first in the ξ-direction, then the η-direction. We can then evaluate the inner product in all the computational grid points as a matrix-vector product [10, 11]

$$\boldsymbol{w} = \boldsymbol{B}\boldsymbol{f}, \tag{1.10}$$

where $\boldsymbol{B} \in \mathbb{R}^{(N-1)^2\times(N-1)^2}$ is the mass matrix, with entries given by

$$B_{(ij)(mn)} \equiv (\ell_i\ell_j, \ell_m\ell_n)_N = \rho_i\rho_j J(\xi_i,\xi_j)\delta_{im}\delta_{jn}. \tag{1.11}$$

The two-digit index counts all the nodal points, sampled in the same way as in $\boldsymbol{f}$ (first in the ξ-direction, then the η-direction), but now both for rows and columns of the matrix. Note that $\boldsymbol{B}$ is diagonal, due to the fact that the test functions are orthogonal in the discrete inner product, i.e. evaluated using GLL quadrature.

In order to achieve similar matrix-vector products for the other terms in (1.9), we construct a vector $\boldsymbol{\varphi}$ similar to $\boldsymbol{f}$, containing the values φ_{mn} of the solution in the nodal points. Evaluation of the operators can be realized as matrix-vector products similar to (1.10) with matrices $\boldsymbol{A}$, $\boldsymbol{C}$ and $\boldsymbol{E}$. In the same way as for the mass matrix (1.11), the entries of $\boldsymbol{A}$ are given by

$$A_{(ij)(mn)} = a_N(\ell_i \ell_j, \ell_m \ell_n),$$

and similarly for $\boldsymbol{C}$ and $\boldsymbol{E}$. We arrive at a set of ordinary differential equations

$$\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{B}\boldsymbol{\varphi} + \boldsymbol{A}\boldsymbol{\varphi} + \boldsymbol{C}\boldsymbol{\varphi} - \boldsymbol{E}\boldsymbol{\varphi} = \boldsymbol{B}\boldsymbol{f}. \tag{1.12}$$

1.3.1 Implementation issues

Constructing the matrices in (1.12) explicitly is possible, but not computationally efficient. The full mass matrix requires storage for $\mathcal{O}(N^4)$ floating point numbers, and a matrix-vector product requires $\mathcal{O}(N^4)$ floating point operations. Utilization of techniques for storing sparse matrices could save a lot of storage space for the mass matrix, but the other matrices are not diagonal, and they will require full storage. Hence, it will not lower the asymptotic running time and storage requirement.

However, the matrices need not be constructed. Instead of representing $\boldsymbol{f}$ as a vector, we will represent it as a matrix, such that $\boldsymbol{f} \in \mathbb{R}^{(N-1)\times(N-1)}$. This representation much better reflects the geometry of the problem. We also create matrices $\boldsymbol{R}$ and $\boldsymbol{J}$ with entries

$$\begin{aligned} R_{ij} &= \rho_i \rho_j, \\ J_{ij} &= J(\xi_i, \xi_j), \end{aligned}$$

and we can then perform the discrete operator evaluation (1.10) as an element-wise matrix-matrix product

$$W_{ij} = R_{ij} J_{ij} \varphi_{ij}, \qquad 1 < i, j < N-1.$$

This only requires $\mathcal{O}(N^2)$ operations. Evaluation of the other discrete operators can be done in the same way, exploiting the tensor-product feature of our basis functions [21, 26]. However, they will involve some full matrix-matrix products of $N \times N$ matrices, requiring $\mathcal{O}(N^3)$ operations. Still, it is a significant improvement. All in all we are able to evaluate all the operators in (1.12) in $\mathcal{O}(N^3)$ operations, using only $\mathcal{O}(N^2)$ storage locations.

For a more detailed discussion on the topic of computational efficiency in the implementation of spectral methods, see for example [11].

1.4 Temporal discretization

The semi-discrete system (1.12) is a set of first order linear ordinary differential equations. When applying temporal discretization, any time integration method can in principle be used. Explicit schemes are best avoided though, since the time step restriction associated with such schemes may render them too costly.

When constructing an implicit scheme, we want to be able to customize the treatment of each operator according to its properties. Both $\boldsymbol{A}$ and $\boldsymbol{B}$ are symmetric positive definite (SPD), and allow for efficient iterative algorithms such as the Conjugate Gradient (CG) method for solving the algebraic system. Hence, they can be treated implicitly. The matrices $\boldsymbol{C}$ and $\boldsymbol{E}$, on the other hand, are not SPD, and should be treated explicitly. One way to achieve this is through an operator splitting method, as described in [24]. Another possibility is a simple implicit multi-step scheme with modified coefficients to allow for explicit treatment of the terms $\boldsymbol{C\varphi}$ and $\boldsymbol{E\varphi}$ [14]. This is the method we will use in this text. A second order semi-implicit Backward Difference (BD2) scheme for (1.12) results in an algebraic system

$$\begin{aligned}(\boldsymbol{A}^{n+1} + \frac{3}{2\Delta t}\boldsymbol{B}^{n+1})\boldsymbol{\varphi}^{n+1} = \boldsymbol{B}^{n+1}\boldsymbol{f}^{n+1} + \frac{2}{\Delta t}\boldsymbol{B}^{n}\boldsymbol{\varphi}^{n} - \frac{1}{2\Delta t}\boldsymbol{B}^{n-1}\boldsymbol{\varphi}^{n-1}\\ + \sum_{j=0}^{2}\beta_j(\boldsymbol{C}^{n-j} - \boldsymbol{E}^{n-j})\boldsymbol{\varphi}^{n-j}, \quad (1.13)\end{aligned}$$

where the coefficients β_j are chosen so that second order polynomials are integrated exactly. The coefficients (taken from [14]) are

$$\beta_0 = \frac{8}{3}, \quad \beta_1 = -\frac{7}{3}, \quad \beta_2 = \frac{2}{3}. \tag{1.14}$$

The Helmholtz type operator on the left hand side of (1.13) is SPD, since it is a sum of SPD matrices. Hence, the system of equations can be solved efficiently with CG iterations.

1.5 An example with known domain velocity

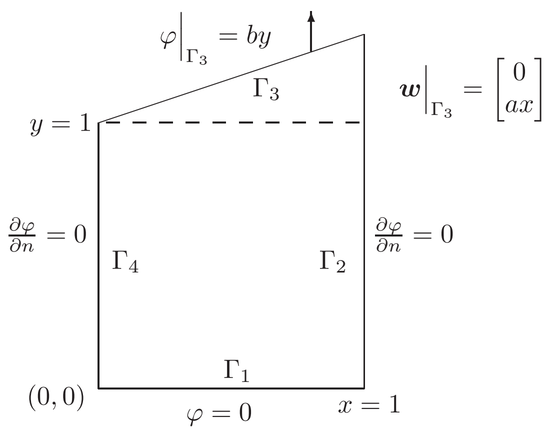
To illustrate the concepts introduced in this chapter, we look at a simple model problem with a known exact solution and known domain velocity. The problem is an unsteady diffusion problem with no volumetric heat sources in Ω:

$$\begin{aligned} \frac{\partial\varphi}{\partial t} &= \nabla^2\varphi && \text{in } \Omega,\\ \varphi &= 0 && \text{on } \Gamma_1,\\ \varphi &= by && \text{on } \Gamma_3,\\ \frac{\partial\varphi}{\partial n} &= 0 && \text{on } \Gamma_2 \,\&\, \Gamma_4. \end{aligned} \tag{1.15}$$

Although there is no convection in the strong formulation of the problem, there will be a non-zero convective term in the ALE formulation, due to the moving domain.

The vertical surfaces of Ω represent adiabatic walls, through which there is no heat flow. At the bottom surface the temperature is held constant, and at the top surface the imposed temperature changes as the domain moves. The problem has the exact solution

$$\varphi(x, y, t) = b\,y,$$



$\varphi\big|_{\Gamma_3} = by$
Γ_3
$\boldsymbol{w}\big|_{\Gamma_3} = \begin{bmatrix} 0 \\ ax \end{bmatrix}$
$y = 1$
$\frac{\partial\varphi}{\partial n} = 0$
Γ_4
Γ_2
$\frac{\partial\varphi}{\partial n} = 0$
Γ_1
$(0,0)$
$\varphi = 0$
$x = 1$

Figure 1.3: Ω *is initially a square. The top boundary* Γ_3 *moves with a velocity* $w_y\big|_{\Gamma_3} = ax$.

where b is a constant. This function is independent of t, but since the computational grid moves as the domain boundary moves, the solution in each grid point will be a time-dependent function.

The horizontal component of the domain velocity is everywhere zero, so the x-component of the grid points do not need to be updated. The domain velocity is zero on the bottom surface Γ_1, while on the top surface Γ_3 it is given by

$$\boldsymbol{w}\big|_{\Gamma_3} = \begin{bmatrix} 0 \\ ax \end{bmatrix}. \tag{1.16}$$

Integrating the second component from $t_0 = 0$ to t, the vertical displacement of Γ_3 is found to be

$$y\big|_{\Gamma_3} = 1 + axt.$$

In order to keep the GLL points correctly distributed in Ω, we interpolate w_y between Γ_1 to Γ_3, using the Gordon-Hall algorithm [13]. This yields a vertical domain velocity

$$w_y(x, y, t) = \frac{axy}{1 + axt}.$$

A linear interpolation is necessary to maintain a GLL distribution of the nodal points in the vertical direction.

Both the solution and the geometry consist of linear functions, and the Jacobian is simply

$$J = \frac{1}{8}\left(2 + at(\xi + 1)\right). \tag{1.17}$$

Hence, all the integrands in (1.6) are polynomials, and there should not be any quadrature error.

The solution φ is a linear function of y, which in turn is a linear function of t. Hence, $\hat{\varphi}$ (i.e., represented in the reference coordinates) is a linear function of t. So is the Jacobian (1.17), so the entries of $\boldsymbol{B\varphi}$ are quadratic functions of t and should be integrated exactly by the second order scheme (1.13).

For the numerical simulations we set both $a = 1$ and $b = 1$. The numerical solution, shown in Figure 1.4, is indeed independent of x and linear in y.

The computational grid is properly shaped in the interior, due to our linear interpolation of the domain velocity.

The Figure also shows the error as a function of the step length Δt in the time integration. The error is measured in the energy norm, which is defined by

$$|||v|||^2 \equiv a(v, v).$$

For convection-diffusion problems, the energy norm coincides with the H^1 semi-norm. We see that the error is independent of Δt, and is everywhere on level with the tolerance level in the CG iterations. Simulations have also been done to confirm that the spatial discretization error is negligible for all N. Thus we conclude that the problem can be solved to machine precision level for any resolution in both space and time.

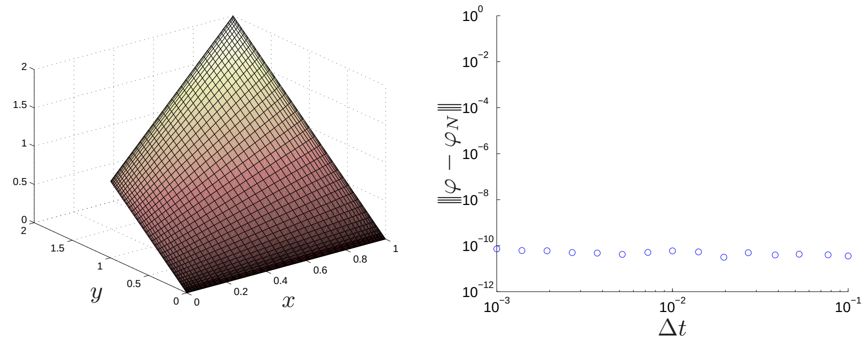


Figure 1.4: *Left: numerical solution of the model problem with $a = 1$ and $b = 1$, using polynomial degree $N = 20$ and $n = 1000$ time steps from time $t_0 = 0$ to time $t = 1$. Right: error in the numerical solution, as a function of the step length Δt in the time integration. The error is on the tolerance level of the CG iterations, since neither the spatial nor the temporal discretization introduce any numerical error.*

Chapter 2

The Stefan problem

An a priori known domain velocity is typically not available. Instead we may have a general law or condition of how the domain should change given a certain solution to the partial differential equation. The Stefan condition is an interface condition providing such a connection in convection-diffusion problems, linking the heat flow across the boundary of the computational domain to the motion of the boundary.

The Stefan problem is a convection-diffusion boundary value problem, used to model phase transition in heat transfer problems. Different domains represent different phases of a medium, in which physical properties such as thermal conductivity may be different. Typically, the phases are solid and liquid, and the interface between the phases is the point at which matter melts or solidifies. The temperature at which this phase transition occurs is always constant for a certain matter at a certain pressure (i.e. $0°$C for water at one atmosphere). Melting is an *endothermic* reaction, meaning heat is absorbed, while freezing is an *exothermic* reaction, where heat is released. The heat needed to melt a unit mass one degree is called the *latent heat* and is denoted L.

The PDE is solved to find the temperature in each phase separately, and then the Stefan condition is used to determine how the interface between the two phases moves. Note that the Stefan condition only provides information about what happens in the direction normal to the surface. Tangentially, we have no information. This means that when moving the computational grid, we have to decide how the grid points are to move tangentially along the surface. A bad choice of tangential velocity may result in a poor distribution of the grid points on the surface, which again results in poor numerical approximation properties. We will consider different ways to remedy this.

But first we will present the Stefan condition and show how it relates the normal movement of the interface to the heat flux across that interface. We will consider two ways of finding the heat flux.

2.1 The Stefan condition

Consider a two-dimensional Stefan problem as shown in Figure 2.1. In the two domains Ω_1 and Ω_2 there are either liquid or solid. Which is which is irrelevant. The interface between the two phases, denoted Γ_3, is where the phase transition occurs. The convection-diffusion equation models the heat flow in each phase of the matter. Boundary conditions may be Neumann or Dirichlet, but on Γ_3

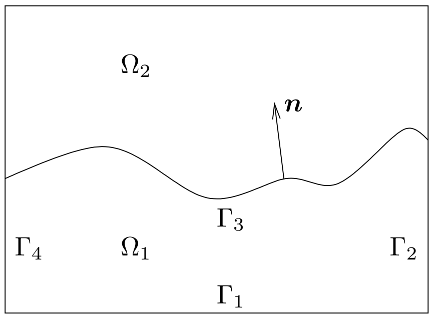


Figure 2.1: *A two-dimensional Stefan problem. The domains Ω_1 and Ω_2 represent two different phases of a matter, and Γ_3 is the interface at which phase transition occurs.*

Dirichlet boundary conditions are natural, since temperature is constant during phase transitions.

Now let η denote the elevation of the surface in the direction of the surface normal $\boldsymbol{n}$. The *Stefan condition* then states that the normal velocity of the interface is given by

$$L\frac{\mathrm{d}\eta}{\mathrm{d}t} = (\boldsymbol{q_2} - \boldsymbol{q_1}) \cdot \boldsymbol{n} \qquad \text{on } \Gamma_3, \tag{2.1}$$

where L is the latent heat and $\boldsymbol{q_2} - \boldsymbol{q_1}$ is the net heat flux across the surface.

When the temperature φ is known, the heat flux is found from Fourier's law,

$$\boldsymbol{q} = -k\nabla\varphi, \tag{2.2}$$

where k is the thermal conductivity. Assume for simplicity that the temperature $\varphi_2 = 0$ in Ω_2, such that there is no heat flux on Γ_3 from Ω_2. The net heat flow across the interface is then equal to the heat flow from Ω_1,

$$Q = -\int_{\Gamma_3} k_1 \frac{\partial\varphi_1}{\partial n}\,\mathrm{d}s, \tag{2.3}$$

where $\frac{\partial\varphi_1}{\partial n}$ is the rate of temperature change in the direction of the unit normal on Γ_3, and $\mathrm{d}s$ is an infinitesimal surface element. In the following, only φ_1 will be considered, and the subscript will be dropped.

The heat flow Q, as given in (2.3), can be computed numerically, given a numerical solution φ_N of the PDE. Partial derivatives of φ_N and the unit normal $\boldsymbol{n}$ can be computed in the nodal points of the computational mesh, and the integral can be evaluated using for example GLL quadrature. This procedure is quite elaborate, and the accuracy is limited by the accuracy of the numerical solution. In the case of a Legendre spectral method, the limiting factor is the polynomial degree N.

There is a much more convenient method, however, based on the connection between equation (2.3) and Neumann boundary conditions. Consider a steady heat transfer problem

$$\begin{aligned} -\nabla \cdot k\nabla\varphi &= f && \text{in } \Omega, \\ \varphi &= 0 && \text{on } \Gamma_1 \;\&\; \Gamma_3, \\ k\frac{\partial\varphi}{\partial n} &= 0 && \text{on } \Gamma_2 \;\&\; \Gamma_4. \end{aligned} \tag{2.4}$$

This problem is a simplification of the full convection-diffusion problem (1.1) in order to focus on the essential elements of the procedure. Extension to unsteady convection-diffusion is straightforward. We allow here for the thermal conductivity to be a function of x and y to show that it does not affect the method.

The weak formulation of the problem is: find $\varphi \in X = \{v \in H^1(\Omega) \mid v\big|_{\Gamma_1} = v\big|_{\Gamma_3} = 0\}$ such that

$$a(v,\varphi) = (v,f) \qquad \forall v \in X, \tag{2.5}$$

where $(\cdot,\cdot)$ is the L^2 inner product and the diffusion term now includes the conductivity k:

$$a(v,\varphi) = \int_\Omega k\,\nabla\varphi\cdot\nabla v\,\mathrm{d}\Omega.$$

Assume that this problem is solved to find φ. In order to satisfy the Dirichlet boundary conditions, there is heat flux across Γ_1 and Γ_3, but we do not know this heat flux.

Now consider a modification of the problem (2.4), with inhomogeneous Neumann boundary conditions imposed on Γ_3 instead of Dirichlet boundary conditions:

$$\begin{aligned} -\nabla \cdot k\nabla\varphi &= f && \text{in } \Omega, \\ \varphi &= 0 && \text{on } \Gamma_1, \\ k\frac{\partial\varphi}{\partial n} &= g && \text{on } \Gamma_3, \\ k\frac{\partial\varphi}{\partial n} &= 0 && \text{on } \Gamma_2 \;\&\; \Gamma_4. \end{aligned} \tag{2.6}$$

The change of boundary conditions calls for a new solution space. It also results in a new weak formulation, which reads: find $\varphi \in X^* = \{v \in H^1(\Omega) \mid v\big|_{\Gamma_1} = 0\}$ such that

$$a(\varphi,v) = (v,f) + l^S(v) \qquad \forall v \in X^*. \tag{2.7}$$

The new term on the right hand side is a surface integral

$$l^S(v) = \int_{\Gamma_3} k\frac{\partial\varphi}{\partial n}v\,\mathrm{d}s = \int_{\Gamma_3} gv\,\mathrm{d}s \tag{2.8}$$

that appears due to the inhomogeneous Neumann boundary condition on Γ_3. Note that this integral is quite similar to (2.3). In fact, comparing the two, we see that the heat flow through Γ_3 can be written as

$$Q = -l^S(v^*) \tag{2.9}$$

for functions v^* with the property that $v^*\big|_{\Gamma_3} = 1$.

Now the key is to assume that the function g is chosen so that the modified problem has the exact same solution as the original problem. That is, the imposed heat flux across Γ_3 is adjusted so that the temperature is zero on this surface. The solution φ of the original problem is then also a solution of the modified problem. The value of the surface integral term (2.8) can then be computed as the remainder

$$-l^S(v) = (v, f) - a(\varphi, v) = r \qquad \forall v \in X^*. \tag{2.10}$$

This remainder will be zero everywhere except on Γ_1 and Γ_3.

To achieve a numerical approximation of Q, we first discretize the original problem (2.5) with a Legendre spectral method. This yields an algebraic system

$$\boldsymbol{A}\boldsymbol{\varphi} = \boldsymbol{B}\boldsymbol{f}$$

which is solved for $\boldsymbol{\varphi}$. If N is the polynomial degree used in the discretization, then $\boldsymbol{\varphi}, \boldsymbol{f} \in \mathbb{R}^{N^2-1}$ and $\boldsymbol{A}, \boldsymbol{B} \in \mathbb{R}^{(N^2-1)\times(N^2-1)}$.

Discretizing the modified problem (2.7) results in a modified algebraic system

$$\boldsymbol{A}^*\boldsymbol{\varphi}^* = \boldsymbol{B}^*\boldsymbol{f}^* - \boldsymbol{r}^*,$$

where $\boldsymbol{r}^*$ is a vector containing the remainder (2.10) in each computational grid point. Now there are degrees of freedom associated with Γ_3 as well, so $\boldsymbol{\varphi}^*, \boldsymbol{f}^*, \boldsymbol{r}^* \in \mathbb{R}^{N(N+1)}$ and $\boldsymbol{A}^*, \boldsymbol{B}^* \in \mathbb{R}^{N(N+1)\times N(N+1)}$. The solution $\boldsymbol{\varphi}^*$ is found by extending $\boldsymbol{\varphi}$ by zeros in all positions corresponding to grid points on Γ_3 (recall that we assumed homogeneous boundary conditions on this boundary in (2.4)). The remainder in (2.10) is then found by computing

$$\boldsymbol{r}^* = \boldsymbol{B}^*\boldsymbol{f}^* - \boldsymbol{A}^*\boldsymbol{\varphi}^*.$$

Comparing (2.9) and (2.10), we see that we need to choose a $v \in X^*$ such that $v\big|_{\Gamma_3} = 1$. The unity function on Γ_3 is simply the sum of all the test functions associated with degrees of freedom on this surface, and hence the heat flow across the interface is given by

$$Q_N = \sum_{k=0}^{N} r_{kN},$$

i.e. the sum of the remainder in the grid points on Γ_3.

Finally we add a note about practical implementation. The system matrices in the original and the modified problems have different dimensions, and it seems we would have to construct $\boldsymbol{A}^*$ and $\boldsymbol{B}^*$ explicitly, just for the purpose of finding Q. This is not the case. We can actually form $(N+1)^2 \times (N+1)^2$ matrices, as if solving with pure Neumann boundary conditions, and use them to solve both problems. We just have to *mask* the results in the grid points where we have Dirichlet boundary conditions. This can be done by creating a *mask matrix* $\boldsymbol{M}$ with zeros in all the entries corresponding to nodes on Dirichlet boundaries, and ones in all the other entries. Each side of the algebraic system is then multiplied by the mask matrix element-wise in order to ensure zero contribution to the values already imposed in those grid points from the Dirichlet boundary conditions.

2.2 Updating the domain velocity

When the heat flux through the phase transition front is known, the movement of the interface can be computed, using the Stefan condition (2.1) and the kinematic condition (1.3). The Stefan condition gives the speed of propagation of the interface, and the kinematic condition requires that this also be the normal component of the domain velocity. The latter is by definition equal to the temporal derivative of the surface elevation η, i.e.

$$w_n \equiv \frac{\mathrm{d}\eta}{\mathrm{d}t}.$$

Now assume that we have followed the procedure of the last section to find the remainder r_{kN} in each grid point on Γ_3. Consider equation (2.9), which gives the heat flux on the whole interface. By multiplying the integrand with a test function associated with a grid point on Γ_3, we see that the result equals the remainder r_{kN} in that grid point:

$$r_{kN} = -\int_{\Gamma_3} k\frac{\partial \varphi_N}{\partial n} v_{kN}\,\mathrm{d}s.$$

Applying the Stefan condition to the integrand, we get

$$r_{kN} = \int_{\Gamma_3} L w_n\, v_{kN}\,\mathrm{d}s, \qquad k = 1, \ldots, N.$$

The integral is transformed to the reference domain and evaluated using GLL quadrature,

$$r_{kN} = L\sum_{\alpha} \rho_\alpha (w_n)_{\alpha N}\delta_{\alpha k}(J_S)_\alpha = L\rho_k(w_n)_{kN}(J_S)_k,$$

where J_S is the Jacobian along the surface Γ_3. In order to solve the above equation for all k, we have to solve a linear system

$$\boldsymbol{r} = \boldsymbol{B}_S\boldsymbol{w}.$$

Here $\boldsymbol{r}, \boldsymbol{w} \in \mathbb{R}^{N+1}$ are vectors containing the remainder and the normal velocity, respectively, on Γ_3, and $\boldsymbol{B}_S$ is the surface mass matrix. Fortunately, due to the discrete orthogonality of the test functions when using GLL quadrature, $\boldsymbol{B}_S$ is diagonal. Hence, the above equation can easily be solved for $\boldsymbol{w}$ just by dividing $\boldsymbol{r}$ by the diagonal elements of $\boldsymbol{B}_S$.

Obviously we must require $w_n = 0$ on the part of the surface where there is no phase transition. As a result, w_n is known all over the boundary $\partial\Omega$. For now we postpone the problem of choosing a tangential component of $\boldsymbol{w}$, we just assume that it has been done. We then have to extend the domain velocity from the boundary $\partial\Omega$ to the interior of Ω. We typically want it to be as smooth as possible in the interior, but apart from that we have quite a lot of freedom in how to do it. There are several possible approaches. It can be solved as an elasticity problem, modeling the components of $\boldsymbol{w}$ as an elastic material [17, 20, 33]. The boundary values are the values we have found previously in this section.

Another possibility is harmonic extension. It means solving a Poisson problem for the domain velocity:

$$\begin{aligned} \nabla^2 \boldsymbol{w} &= \boldsymbol{0} && \text{in } \Omega, \\ \boldsymbol{w} &= \boldsymbol{w}_\Gamma && \text{on } \partial\Omega. \end{aligned} \tag{2.11}$$

This yields a vector-valued function $\boldsymbol{w}$ in Ω where each component is in H^1, adhering to the known boundary data.

Yet another possibility is modeling the motion of the mesh nodes as a steady Stokes flow [5]. This means solving a steady Stokes problem with proper boundary values. The incompressibility condition will result in a divergence-free domain velocity. Although we will not apply this technique for extending the domain velocity, we will return to both the steady and the unsteady Stokes problems later.

Last but not least, we can use the Gordon-Hall algorithm. This may be the quickest method, and it yields a smooth domain velocity. The Gordon-Hall algorithm is based on a linear interpolation, and it uses the sum of three different interpolations (in $\mathbb{R}^2$) in order to satisfy the boundary conditions. The Gordon-Hall algorithm was first presented in [13], and it is also described in for example [11].

Once the domain velocity has been found all over the computational domain, we update the computational grid by solving

$$\frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}t} = \boldsymbol{w}$$

with an explicit time integration scheme.

2.2.1 A numerical example: the Stefan problem

We illustrate the concepts of this section with an example. The model problem is a very simple Stefan problem: a rectangular domain, zero temperature in one of the phases, constant temperature along the phase transition front and hence a constant domain velocity along the front. Besides, we assume that the temperature stabilizes very rapidly in Ω compared to the dynamics associated with phase transition, enabling us to model it as a steady state problem at each point in time. The problem is then:

$$\begin{aligned} \nabla^2 \varphi &= 0 && \text{in } \Omega, \\ \varphi &= 1 && \text{on } \Gamma_1, \\ \varphi &= 0 && \text{on } \Gamma_3, \\ \frac{\partial \varphi}{\partial n} &= 0 && \text{on } \Gamma_2 \ \& \ \Gamma_4. \end{aligned} \tag{2.12}$$

The upper boundary Γ_3 is the interface at which the phase transition occurs. The temperature is zero on and outside Γ_3, so there is no contribution to heat flux on Γ_3 from outside Ω. The vertical surfaces are adiabatic walls, and at the lower surface the temperature is kept constant, with no phase transition.

The exact solution is a linear function, decreasing from $\varphi = 1$ at Γ_1 to $\varphi = 0$ at Γ_3:

$$\varphi(x, y, t) = 1 - \frac{y}{y_{top}}, \tag{2.13}$$

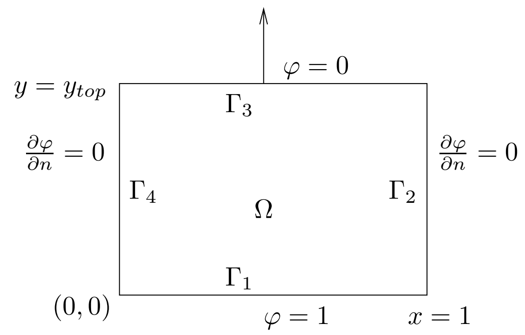


Figure 2.2: *Illustration of the Stefan problem (2.12). The temperature φ is independent of x, so the phase transition front Γ_3 will remain horizontal.*

where $y_{top}(t)$ is the y-value at Γ_3. Due to the geometry of the problem, y_{top} does not depend on x. The velocity of the phase transition can also be found exactly. Applying Fourier's law (2.2) to the heat flux in the Stefan condition (2.1), we find the following equation for y_{top}:

$$L\frac{\mathrm{d}y_{top}}{\mathrm{d}t} = -k\nabla\varphi = -k\frac{\partial\varphi}{\partial y} = \frac{k}{y_{top}}.$$

This differential equation is separable and can easily be solved, integrating from $t_0 = 0$ to t. The exact solution for the vertical displacement of Γ_3 is

$$y_{top}(t) = \sqrt{y_0^2 + \frac{2kt}{L}}, \tag{2.14}$$

where $y_{top}(0) = y_0 = 1$.

The problem is solved numerically to check how the error in the geometry depends on the temporal step length. Discretization is applied directly to the weak formulation of (2.12). There is no ALE formulation now, due to our assumption of rapidly stabilizing temperature. We arrive at a system of algebraic equations

$$\boldsymbol{A}\boldsymbol{\varphi} = \boldsymbol{g},$$

where $\boldsymbol{A}$ is the discrete Laplace operator, and $\boldsymbol{g}$ is a vector representing the known inhomogeneous boundary conditions along Γ_1. The normal component of the domain velocity at the moving surface is found by solving

$$\boldsymbol{B}_S\,\boldsymbol{w}_n = (\boldsymbol{g} - \boldsymbol{A}\,\boldsymbol{\varphi})\big|_{\Gamma_3},$$

where $\boldsymbol{w}_n$ is a vector containing w_n on Γ_3. The domain velocity is then given by $w_x = 0$, $w_y = w_n$. We update the domain by solving

$$\frac{\mathrm{d}y_{top}}{\mathrm{d}t} = w_y$$

with both first, second and third order Adams-Bashforth schemes (AB1, AB2 and AB3), to see three different orders of convergence. Note that while this is actually a scalar equation, it has been implemented as a vector equation to be solved for each grid point on Γ_3. This is done to check that the domain velocity is actually the same along all of Γ_3, and that the surface remains horizontal. It is also a preparation for the more complex model problems to come.

Convergence is verified by measuring the difference between the numerically computed $y_{top,N}$ and the theoretical $y_{top}(t)$ from (2.14), averaged over all the nodal points on Γ_3. Results are displayed in Figure 2.3. The plot is logarithmic in both axes, so the order of temporal convergence can be read out as the slope of the curves. As expected, we see first, second and third order convergence for AB1, AB2 and AB3, respectively.

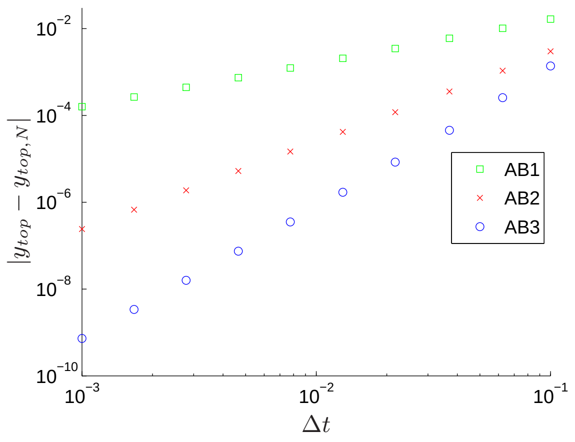


Figure 2.3: *Error in the numerical solution of the model problem, measured as the absolute value of the difference between the theoretical and the numerical values of y_{top}, the vertical position of the upper boundary Γ_3. Time integration is preformed from $t_0 = 0$ to $t = 1$ for different values of the step length Δt. The polynomial degree is simply $N = 1$, to show that this is sufficient for representing both the solution and the geometry exactly in this particular example. We observe first, second and third order convergence from the first, second and third order Adams-Bashforth schemes.*

Note that we need the exact solution of the domain velocity prior to t_0 in order to start the multi-step schemes AB2 and AB3. The exact domain velocity at Γ_3 for $t < t_0$ is found by differentiating (2.14) with respect to time.

2.2.2 Choice of tangential component of the domain velocity

As mentioned in the beginning of this chapter, we have significant freedom in the choice of tangential velocity w_t on the interface. Our target is to preserve the regularity of the mapping from the reference domain to the physical domain. An easy way out is to require $\boldsymbol{w}$ to be normal to the surface [7], i.e. $\boldsymbol{w} \cdot \boldsymbol{t} = 0$, where $\boldsymbol{t}$ is the unit tangential vector. Then we do not have to compute any tangential component, we can just find the unit normal and multiply it by the proper length w_n.

In the previous example, the distribution of the grid points on Γ_3 remained unchanged because the normal vector was the same all over the interface. This may not always be the case. If there is a non-constant curvature on the interface, the grid points may move relative to each other on the boundary, even though the domain velocity is normal to the boundary. An example of such behavior is shown in Figure 2.4. Over time, as the interface moves, this distribution may become increasingly different from the initial distribution, resulting in increasing interpolation errors.

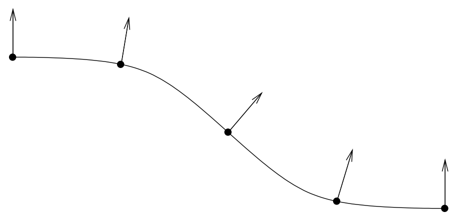


Figure 2.4: *Displacement of grid points on a moving surface. The direction of the unit normal will over time lead to a clustering of grid points in the rightmost part of the curve.*

The error in the distribution of nodal points may be remedied by adding a tangential component to the domain velocity. This does not affect the kinematic condition, and will not change the shape of the surface of the domain. What tangential component to add, however, is not a simple question, and there may be more than one answer. It is a question that can not be answered without considering the geometry at hand, and it may be difficult to find an answer without some a priori knowledge of how the domain evolves over time. This problem is best illustrated with an example.

2.2.3 A numerical example: moving grid points

This problem is quite similar to that in example 2.2.1. The difference is the boundary condition on Γ_1 and the addition of a temporal partial derivative in the strong formulation, which will necessitate an ALE formulation in the weak form. The problem is:

$$\begin{aligned} \frac{\partial \varphi}{\partial t} &= \nabla^2 \varphi && \text{in } \Omega, \\ \varphi &= g(x) = 1 + \frac{1}{2}\cos 2\pi x && \text{on } \Gamma_1, \\ \varphi &= 0 && \text{on } \Gamma_3, \\ \frac{\partial \varphi}{\partial n} &= 0 && \text{on } \Gamma_2 \text{ \& } \Gamma_4. \end{aligned} \tag{2.15}$$

Note that the derivative of the imposed temperature on Γ_1 is zero at the end points of that surface, which it needs to be, in order to meet the Neumann conditions on the vertical boundaries. Initially the problem domain is $\Omega = (0,1) \times (0,1/4)$.

The ALE formulation of the problem is the one we derived in equation (1.6). Now, the right hand side is zero, since there is no volumetric heat source. Also, there is no prescribed convection field, so the convection term in (1.7) simplifies to

$$c(v,\varphi) = -\int_\Omega v\, \boldsymbol{w} \cdot \nabla\varphi \,\mathrm{d}\Omega.$$

Spatial discretization is done as always with a Legendre spectral method, resulting in a system of differential equations (1.12). A second order Backward Difference scheme (1.13) is applied for time integration.

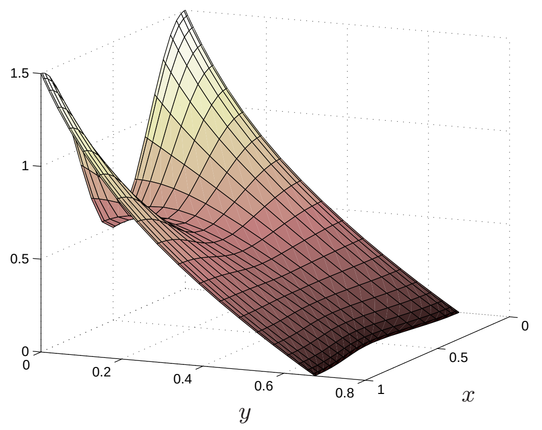


Figure 2.5: *Numerical solution of the model problem at time $t = 1$, using polynomial degree $N = 20$ and a step length $\Delta t = 10^{-3}$. The latent heat is $L = 5$.*

The solution decreases from $g(x)$ at Γ_1 to zero at Γ_3. Since the temperature is higher towards the vertical boundaries of the domain, the rate of phase transition will also be greater here, and hence the domain velocity will be larger. As a consequence the surface Γ_3 will have almost the same cosine shape as $g(x)$. However, as time goes by and the phase transition front moves away from the bottom surface, the contributions from $g(x)$ will even out along Γ_3, and the difference in the domain velocity from the sides to the middle of Γ_3 decreases. Hence, as the domain grows, the curvature of the top boundary decreases.

The horizontal component of the unit normal on Γ_3 is everywhere pointing towards the center of Γ_3, except at the end points, where it is zero. When moving the grid points on the surface in the direction of the surface normal, all the points will move towards the middle of the surface, as shown in Figure 2.6.

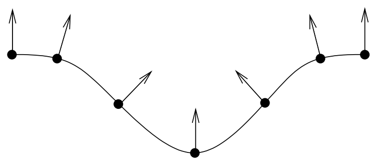


Figure 2.6: *Displacement of the grid points on the interface between the phases. All grid points are moving towards the center of the curve.*

This problem is apparent in the numerical simulation and illustrated in Figure 2.7. Here, the x-component of the grid points on Γ_3 are plotted as a function of the reference coordinate ξ. Initially, the mapping is given by the linear function $x = (\xi + 1)/2$, yielding a GLL distribution in the x-direction. When the grid points are moved with the domain velocity, they all move horizontally towards $x = 0$. Over time the x-component of the nodal points becomes bigger than the GLL distribution to the left of the center point (i.e., for negative ξ), and smaller to the right (ξ positive).

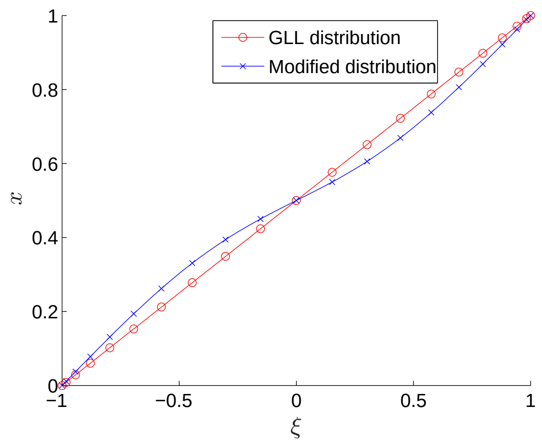


Figure 2.7: *The x-component of the nodal points on Γ_3 as a function of ξ for the solution in Figure 2.5. Initially the points are distributed as $x = (\xi + 1)/2$, i.e. a linear function. As the grid points all move towards the center point, the x-component is larger than the GLL distribution to the left of the center point, and smaller to the right.*

A simple way to avoid this problem is to ignore the x-component of the domain velocity and update just the y-component. After all, the expansion of the domain is in the vertical direction, the vertical boundaries are adiabatic walls. Removing the horizontal component, we are guaranteed that the GLL points will be correctly distributed along the x-axis. The price to pay is that the kinematic condition is no longer satisfied. This makes the method erroneous, and we can not expect the solution to be correct. However, we will avoid collapse.

Considering what has been said about this issue in the previous section, we should be able to do better. We can achieve both goals at the same time, i.e. satisfying the kinematic condition and preserving the GLL distribution, by adding a tangential component to the domain velocity to make the resulting velocity vector vertical all along Γ_3. We do not need to compute the tangential component explicitly. Simple trigonometry leads to a vertical component

$$\widetilde{w}_y = \frac{w_n^2}{w_y},$$

where w_n and w_y are the normal component and y-component of the original $\boldsymbol{w}$, respectively. Since $w_n > w_y$, we expect this approach to yield a faster moving phase transition front than the two other alternatives.

We compare the three methods in a numerical simulation. The system is now simulated much longer in time (until $t = 10$) to make the differences more conspicuous. Figure 2.8 shows the distribution of the grid points on the interface Γ_3 at the final time step. Using a strictly normal $\boldsymbol{w}$ on the interface has caused

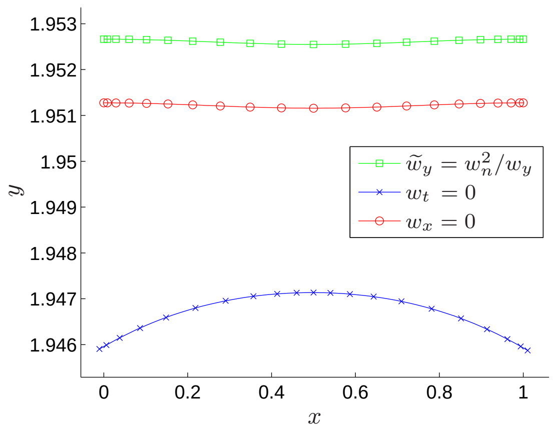


Figure 2.8: *The distribution of the grid points on the top boundary Γ_3 in the final configuration at $t = 10$. The strategy of no tangential component of $\boldsymbol{w}$ on the interface has caused a severe displacement of the grid points and large numerical errors. The shape of the boundary is incorrect, and the numerical solution will only deteriorate further until collapsing. The middle curve represents removal of the x-component from $\boldsymbol{w}$, while the top curve represents addition of a tangential component to satisfy the kinematic condition. The curves have the same shape and the same distribution of the grid points. The upper curve is the solution we expect to be the best.*

a substantial displacement of the grid points and a serious deterioration in the regularity of the mapping from the reference domain. The shape of the interface is obviously incorrect, and the end points of the interface are no longer moving strictly vertically, causing a deformation of the entire domain. The numerical solution is not far from collapse.

The two other approaches have avoided this deterioration. They have retained the cosine-shape of the interface, although the amplitude is now very small, as we predicted. The method which violated the kinematic condition has caused the interface to move shorter than the method which satisfies the kinematic condition, which is also as we expected. The latter is what we consider to be the best solution in this case.

2.3 Multi-element methods

So far all the domains encountered are distorted rectangles. This enables us to solve the differential equations on a single element, representing the domain using smooth transformations from a square reference domain. However, many domains do not fit this description, i.e. their surface can not be divided into four smooth curves. Examples include common, simple domains as triangles or circles. When we try to represent such domains using a single quadrilateral element, we end up with “corners” where the angle between the adjacent curves are 0 or π. The Jacobian of the associated mapping becomes singular, and this can cause seriously deteriorating convergence properties and should be avoided.

The solution is using multi-element methods [23, 27]. The domain Ω is divided into K subdomains

$$\Omega = \bigcup_{k=1}^{K} \Omega_k,$$

where each subdomain has four corners and hence can be mapped to the reference domain. The number of elements and the topology of the elements can be adapted to handle complex geometries. The multi-element approach is highly similar to the finite element method, where the solution is usually approximated by piecewise linear functions. Here we will use high order polynomials in each element, which increases the number of variables to be determined in each element.

In the same way as for the finite element method, we need to *assemble* the contribution from each element across the element boundaries. The solution is required to be smooth across the interface between the elements. When evaluating the operators, care must be taken to ensure that contributions from adjacent elements are properly added up at the right nodal points. This means summing integrals associated with the same test function, evaluated over each subdomain. For example, evaluating the bilinear form $a(\cdot,\cdot)$ from (1.7) yields a sum

$$a(u,v) = \int_{\Omega} \nabla u \cdot \nabla v \,\mathrm{d}\Omega = \sum_{k=1}^{K} \int_{\Omega_k} \nabla u \cdot \nabla v \,\mathrm{d}\Omega_k.$$

First the operator is evaluated in each element, and then contributions are added along surfaces between adjacent elements. We will refer to the summation process as a "direct stiffness summation".

2.3.1 Five-element structure

The multi-element structure that will be used here is a five-element structure, as shown in Figure 2.9. It is one of the simplest structures possible to represent a circle or ellipse.

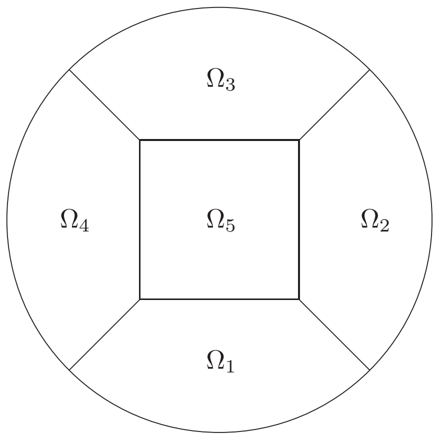


Figure 2.9: *Five-element structure.*

Each of the four elements that are part of the external surface, is rotated so that the circular surface is mapped from the line $\eta = -1$ in the reference domain. This is convenient when it comes to operator evaluation on the external surface, but we must take care when implementing the assembling on the interface between the boundary elements and the center element. In particular, the grid points are distributed in the opposite direction on the interface between elements 3 and 5, and elements 4 and 5. This must be taken into account when constructing the direct stiffness sum.

Each nodal point on the internal surfaces between elements will be represented in the data structure of each adjacent element. Hence, they will be represented more than one place. This is a challenge when it comes to preforming dot products. The product at one point must not be counted more than once. One solution is to make a multiplicity vector (or matrix, depending on the implementation) $\boldsymbol{m}$ consisting of weights, compensating for the number of representations of each point. The weight is 1/2 on internal surfaces with two adjacent elements, and 1/3 on the corners of Ω_5, where three elements meet. Everywhere else the weight is 1. Now, a dot product on Ω is computed as

$$\boldsymbol{v}^T\boldsymbol{w} = \sum_{k=1}^{K}\sum_{j=0}^{N}\sum_{i=0}^{N} v_{ij}^k w_{ij}^k m_{ij}^k$$

where v_{ij}^k is the entry in $\boldsymbol{v}$ corresponding to the GLL point (ξ_i, ξ_j) in subdomain k.

In order to perform the direct stiffness sum, we make a customized function to deal with this particular geometry. It takes one parameter, namely the data structure of values in the computational points in all Ω, and adds the values at grid points that are represented several times in different places in the data structure. We must also take care to store the new value in all the places where the point is represented, in order to avoid inconsistency.

As a side-note it can be added that it is not necessary to sum up the contribution for each operator evaluated. It is possible to first evaluate all the operators involved in the differential equation, and then do the direct stiffness summation. This is computationally more efficient, especially in problems involving many different (discrete) operators.

Even though the topology of the five-element structure is fairly simple, the transition from single-element to multi-element representation is not trivial, and there are many potential sources of error. Therefore, we start by solving a simple Poisson problem with a known analytic solution. This problem will be used to verify exponential convergence in space. Then we will re-implement example 2.2.3 in order to test the multi-element approach on a full ALE formulation. Finally we will present a new ALE example.

2.3.2 Multi-element approach to a Poisson problem

The problem we consider is:

$$\begin{aligned} -\nabla^2\varphi &= 2\pi^2\cos(\pi x)\cos(\pi y) && \text{in } \Omega, \\ \varphi &= \cos(\pi x)\cos(\pi y) && \text{on } \partial\Omega, \end{aligned}$$

where Ω is a circle of radius 1, centered at the origin. The exact solution to this problem is

$$\varphi = \cos(\pi x)\cos(\pi y),$$

i.e. an analytic function. The problem is solved using a five-element approach. The main elements of the numerical procedure that need to be changed from the single-domain approach is the data structure and the evaluation of the discrete Laplace operator. Besides, inner products need to be implemented with the multiplicity matrix, and the direct stiffness sum must be applied after operator evaluations.

Both the numerical solution for $N = 20$ and the error, measured in the energy norm, are shown in Figure 2.10. Exponential convergence is indeed observed. The error stabilizes around 10^{-10}, which is the tolerance in the Conjugate Gradient iterations used to solve the algebraic system.

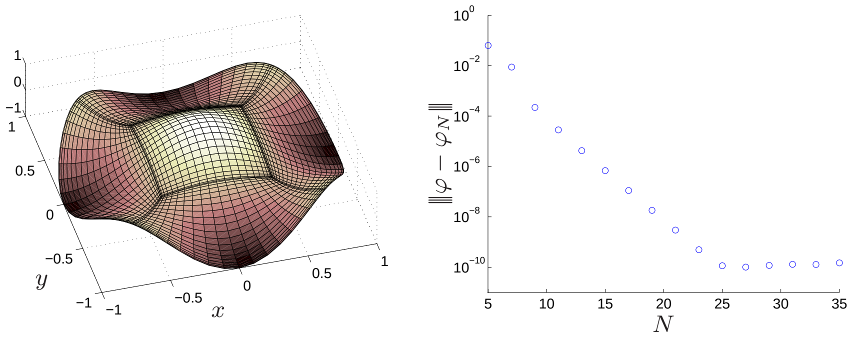


Figure 2.10: *Left: Numerical solution of the model problem, using polynomial degree $N = 20$. The five-element structure can be recognized by a higher density of grid lines near the boundaries between the elements. Right: Error, measured in the energy norm. Convergence is exponential, and the error levels out when reaching the tolerance level in the CG iterations.*

2.3.3 Multi-element approach with full ALE formulation

We solve the model problem from Section 2.2.3, but this time with a multi-element representation of the domain. The problem requires full ALE formulation, which in turn gives several operators to be evaluated over the entire domain. We take care to evaluate each operator over all elements before applying the direct stiffness sum.

Compared to the previous example, we now have to update the computational domain. After finding the heat flux and applying the Stefan condition, we find the normal vector on the external surface. For simplicity, no tangential component is added to the domain velocity on the moving interface. Now, the external surface is distributed on four different elements, and we have to interpolate the domain velocity to the internal grid points. The simplest approach is to use a harmonic extension, as in (2.11). Since we implemented the solution of the Poisson problem with a multi-element approach in the previous example, this is now easily done.

The numerical solution is shown in Figure 2.11. Compared to the solution in Figure 2.5, we see that they are practically identical. This serves as a indication of correct implementation.

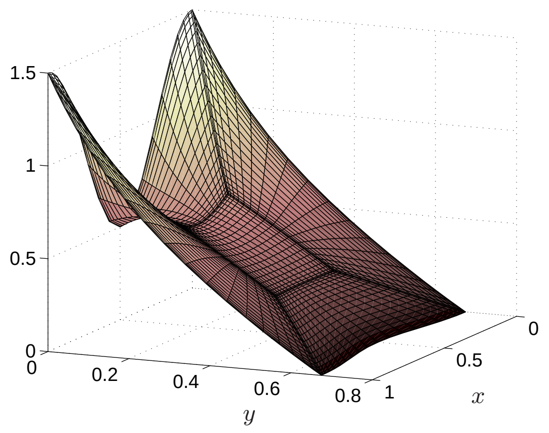


Figure 2.11: *Numerical solution of the model problem, using polynomial degree $N = 20$ and $n = 1000$ time steps from time $t_0 = 0$ to time $t = 1$. The latent heat is $L = 5$. These are the same parameters as those used in the example in Section 2.2.3, and comparing this result to Figure 2.5, we see that the solutions are practically the same. This confirms that our multi-element implementation is correct.*

2.3.4 A Stefan problem on a circular domain

Consider a Stefan problem on a circular two-dimensional domain, where all of the circular boundary is a part of the phase transition front. The initial domain is a circle of radius $r_0 = 1$, and the temperature is in polar coordinates given by

$$\varphi(r, \theta, t) = 1 - r^2.$$

Hence the temperature is zero on the surface, and we assume zero temperature outside Ω, such that the net heat flux on the interface equals the heat flux out from Ω. The Stefan condition applies to all of $\partial\Omega$, and due to the symmetry of the problem, we expect the domain to expand radially, as a growing circle. We also expect the temperature to decrease inside Ω, and since energy is lost in the phase transition, temperature should eventually fall to zero and the domain should stabilize at a certain radius. This radius can be calculated by energy considerations. The internal energy in the initial configuration is

$$E_0 = \int_\Omega \varphi \, \mathrm{d}\Omega = \int_0^{2\pi} \int_0^1 (1 - r^2)\, r \, \mathrm{d}r \, \mathrm{d}\theta = \frac{\pi}{2}.$$

In our two-dimensional model, the energy needed to melt a unit area equals the latent heat L. Hence, the energy needed to expand the domain to a radius r is given by

$$E_{pt} = L \int_0^{2\pi} \int_1^r r \, \mathrm{d}r \, \mathrm{d}\theta = \pi(r^2 - 1),$$

where $L = 1$. Over time, the temperature in Ω approaches zero, and the energy absorbed in the phase transition will approach the initial energy,

$$\pi(r^2 - 1) \rightarrow \frac{\pi}{2}.$$

Hence, as the time $t \rightarrow \infty$, the radius

$$r \rightarrow \sqrt{\frac{3}{2}}.$$

Numerically, we have to run the simulations for a finite amount of time. The order of magnitude of the temperature in Ω can serve as an indicator of how close we should be to the final radius computed above.

Now that the phase transition front extends over multiple elements, we are faced with a new challenge: how to determine the domain velocity in the joints between the elements. The normal component of the domain velocity, which is found from an integral over the phase transition interface, must now be summed up from contributions over each element. Hence, we need a sort of direct stiffness sum over the external surface. We could use the algorithm used on the entire Ω, discarding the results that are not on $\partial\Omega$. A computationally more economic solution, however, is to design a summation procedure to cater for the points that are in the intersection of the external and the internal surfaces. This is fairly easy and is the path followed in this example.

Then there is the question of how to determine the unit normal in the same "corner points". Usually the normal vector is computed locally in the element in which the surface points reside. Now what about points that reside on the joint between two elements? In order to find a unique $\boldsymbol{w}$, we use information from both adjacent elements. We add the normal vectors computed on each element and normalize the resulting vector. The length of the normal component w_n is the same in both representations of the corner point, due to the direct stiffness sum. Hence, we end up with a unique velocity vector in the corner points that contains information from both adjacent elements.

In this problem we will use the Gordon-Hall algorithm to interpolate the domain velocity to the interior. However, we can not simply interpolate linearly from one boundary to the other. We have to find a way to interpolate from the surface, via the center element, to the other surface. Unless we know exactly the distance from each point on the boundary to the surface of the center element, the interpolation will only be piecewise linear. This is sufficient, though, as long as the domain velocity is continuous across element boundaries [12]. In this case we define the domain velocity on the corners of the center element as half of the domain velocity on the external surface. The number is chosen because this is the ratio of the distance from the origin to these corners and the external surface in the initial configuration. This will make the center element expand with approximately the same velocity as the external surface.

The numerical solution is indeed an expanding circular domain, as expected. The radial symmetry in the geometry is properly reflected in the symmetry of the five elements. Figure 2.12 shows that the solution remains a parabola, but that the height approaches zero. Hence the rate of phase transition decreases and approaches zero. The decreasing expansion rate can be seen in the right plot, which shows the radius of the circle as a function of time. The radius

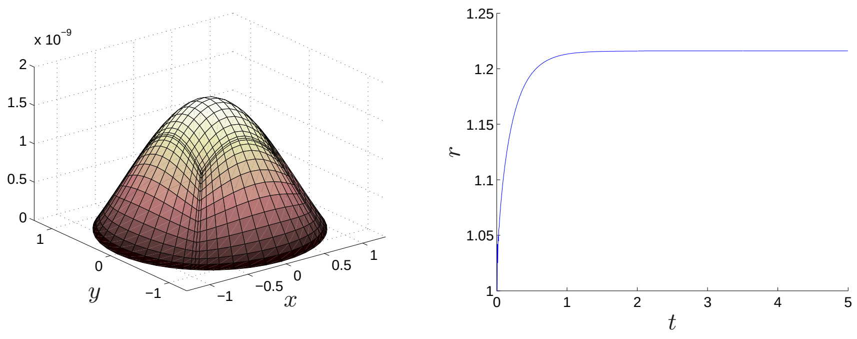


Figure 2.12: *Left: numerical solution of the model problem with $L = 1$, using polynomial degree $N = 14$ and $n = 1000$ time steps from time $t_0 = 0$ to time $t = 5$. The solution φ_N remains a parabola. The peak is of magnitude 10^{-9} and decreasing, since energy is absorbed in the phase transition. Right: radius of the circular domain as a function of time. The radius increases, but stabilizes when the temperature approaches zero.*

increases, quickly in the beginning, then slower and slower as the temperature in Ω decreases.

The radius is calculated as the mean radius of all grid points on the surface, and is found to be $r \approx 1.216$. This is slightly less than the theoretical value $r = \sqrt{3/2} \approx 1.225$. The difference is larger than we might expect, given the magnitude of the temperature. The peak of the temperature is of magnitude 10^{-9}, and the change of radius from $t = 3$ to $t = 5$ is in the fifth decimal. Hence, simulating the system further in time will only result in unnoticeable changes to the radius.

An immediate reaction is that the computation of the radius may be inaccurate. However, this explanation fails as the difference between the maximum and the minimum of the radius in the nodal points on the boundary is about 0.05% at $t = 5$. Changing the way we compute the radius would only change the value in the fourth decimal.

Another explanation could be that the difference is due to numerical errors introduced by either the spatial or the temporal discretization. However, running more simulations with different parameters, we have verified that decreasing the resolution in either time or space (or both) does not change the solution notably. Hence, the resolution is high enough.

What is the explanation then? It is actually quite simple. The semi-implicit scheme (1.13) requires values of φ and $\boldsymbol{w}$ at the last two time steps before t_0 to start the iterations. In this case, since the exact solution is unknown, we have just assumed zero domain velocity prior to t_0. This makes all the convection and expansion terms in the first time step vanish, and in result, the heat flux is not computed correctly in this time step. Energy is lost and can not be recovered, and the final radius is smaller than it should be.

This example shows the value of having model problems where the exact solution is known. Even though we know what the final radius should be, we can not find it numerically with a high precision without some knowledge of what the domain velocity is at the initial configuration.

Chapter 3

The Navier-Stokes equations

We now turn our attention to the world of fluid dynamics. This is one of the most important areas of application for the spectral methods. Fluid flows in time-dependent geometries are encountered in a large number engineering problems, and analytic solutions are hardly available, due to the complexity of the equations. Hence, good numerical approximations are valuable, and the spectral methods have evolved to be able to handle many complex fluid dynamics problems [10].

We will here consider incompressible Newtonian fluid flow, i.e. fluids with constant viscosity. The motion of such fluids is properly described by the Navier-Stokes equations [32]. These equations express conservation of linear momentum and mass, and are in two dimensions given as

$$\begin{aligned}
\rho\left(\frac{\partial u_i}{\partial t} + u_j\frac{\partial u_i}{\partial x_j}\right) - \mu\frac{\partial}{\partial x_j}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right) + \frac{\partial p}{\partial x_i} &= f_i \qquad \text{in } \Omega,\ i = 1,2, \\
\frac{\partial u_j}{\partial x_j} &= 0 \qquad \text{in } \Omega,
\end{aligned} \tag{3.1}$$

where ρ is the fluid density, μ is the dynamic viscosity, u_i is the i-th component of the fluid velocity, and f_i is the i-th component of the body force. The indicial notation is used with the Einstein summation convention, meaning summation over repeated indices. Dirichlet or Neumann boundary conditions are given for either both Cartesian components or for normal and tangential components of the fluid velocity along $\partial\Omega$.

Numerical treatment of the full Navier-Stokes equations is fairly complicated, and the most natural approach to master it is to simplify the equations as much as possible, and then gradually expand them, including more terms. We will start out here by modeling only the viscous term, excluding convection, pressure and unsteady terms. We will then gradually expand the model, via pressure, unsteady terms and convection. In the end we will add a free surface condition, necessitating a moving grid and an ALE formulation.

3.1 The viscous term

The viscous term contains the highest order differential operator in the Navier-Stokes equations. It is a suitable starting point because it is the term that governs both the boundary conditions and the regularity requirement of the weak solution. The simplifications done from the full Navier-Stokes problem is removing pressure from the model, as well the unsteady term and the convection term. The incompressibility condition is then no longer necessary. We arrive at a model problem that is given in strong form as

$$\begin{aligned} -\mu\frac{\partial}{\partial x_j}\left(\frac{\partial u_i}{\partial x_j}+\frac{\partial u_j}{\partial x_i}\right) &= f_i \qquad && \text{in } \Omega, \quad i=1,2, \\ u_j &= 0 && \text{on } \partial\Omega. \end{aligned} \tag{3.2}$$

Homogeneous Dirichlet boundary conditions are assumed for simplicity, and they may be exchanged for any Dirichlet or Neumann boundary conditions. The equations are similar to the Poisson equation, with the important difference that the solution is now a vector, and that the equation for each component of the vector depends on the other component. The solution we are seeking can be written on vector form as

$$\boldsymbol{u} = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}.$$

Similarly, the body force $\boldsymbol{f}$ and test function $\boldsymbol{v}$ are two-dimensional vectors at each point (x, y) in Ω.

The weak formulation is derived by multiplying with test functions and integrating over Ω. The solution space is now a space of two-dimensional vector-valued functions, and so are the test functions. Integration by parts and application of the boundary conditions yields the following weak formulation: find $\boldsymbol{u} \in X = (H_0^1(\Omega))^2$ such that

$$\boldsymbol{a}(\boldsymbol{v},\boldsymbol{u}) = (\boldsymbol{v},\boldsymbol{f}) \qquad \forall \boldsymbol{v} \in X,$$

where $(\cdot,\cdot)$ is an L^2 inner product

$$(\boldsymbol{v},\boldsymbol{f}) = \int_\Omega v_i f_i \,\mathrm{d}\Omega, \qquad i=1,2, \tag{3.3}$$

in each of the components of the vectors, and $\boldsymbol{a} : X \times X \to \mathbb{R}^2$ is the viscous operator, defined by

$$\boldsymbol{a}(\boldsymbol{v},\boldsymbol{u}) = \mu\int_\Omega \frac{\partial v_i}{\partial x_j}\left(\frac{\partial u_i}{\partial x_j}+\frac{\partial u_j}{\partial x_i}\right)\mathrm{d}\Omega, \qquad i=1,2. \tag{3.4}$$

Since there is one equation for each component of the velocity, this operator can be considered vector-valued and is therefore written in vector notation.

The discretization is done with a Legendre spectral element method, where each of the components of $\boldsymbol{u}$ are approximated with Legendre polynomials on each element. The discrete space, in which we seek a solution, is

$$X_N = \{\boldsymbol{v} \in X \mid (v|\Omega_k \circ \mathcal{F} \in (\mathbb{P}_N(\widehat{\Omega}))^2\}.$$

We use a nodal basis of Lagrangian interpolants on a GLL grid in the same way as for the convection-diffusion equation, but now there will be a vector of two test functions associated with each spatial grid point. Hence, a function $\boldsymbol{v} \in X_N$ can on each subdomain Ω_k be expanded in the reference coordinates as

$$\hat{\boldsymbol{v}}^k(\xi,\eta) = \sum_m\sum_n \boldsymbol{v}^k_{mn}\ell_m(\xi)\ell_n(\eta), \qquad k=1,\dots,K,$$

where the coefficients $\boldsymbol{v}_{mn}$ are two-dimensional vectors.

After the spatial discretization, we end up with a system of equations

$$\boldsymbol{A}\boldsymbol{u} = \boldsymbol{B}\boldsymbol{f},$$

where $\boldsymbol{A}$ is the discrete viscous operator and $\boldsymbol{B}$ is the mass matrix. The matrix $\boldsymbol{A}$ is SPD, and hence the system of equations can be solved with the conjugate gradient algorithm.

3.1.1 A numerical example of the pure viscous problem

To illustrate the concepts of the previous section, we solve (3.2) with the following body force:

$$\begin{aligned} f_1 &= \pi^2\mu\left(\frac{3}{4}\cos(\frac{\pi x}{2})\cos(\frac{\pi y}{2}) + \cos(\pi x)\cos(\pi y)\right) \\ f_2 &= -\pi^2\mu\left(\frac{1}{4}\sin(\frac{\pi x}{2})\sin(\frac{\pi y}{2}) + 3\sin(\pi x)\sin(\pi y)\right) \end{aligned}$$

This problem is solved on a domain that coincides with the reference domain, and it has the exact solution

$$\begin{aligned} u_1 &= \cos(\frac{\pi x}{2})\cos(\frac{\pi y}{2}), \\ u_2 &= -\sin(\pi x)\sin(\pi y). \end{aligned}$$

The problem is solved here using a multi-element approach with the five-element structure described in section 2.3.1. This is not necessary in this case since Ω is a square, but it is done in preparation for later problems, when we will encounter circular domains. As for the multi-element approach, there is not much that separates the treatment of fluid problems from that of convection-diffusion problems. The direct stiffness summation depends on the topology of the elements and is therefore the same, except that separate summations must be done for both components of $\boldsymbol{u}$.

The numerical solution for $N = 10$ is shown in Figure 3.1. We see higher density of arrows along the boundaries between the elements, reflecting the five-element structure. The Figure also shows the numerical error. It is measured in the energy norm, given by

$$|\!|\!|\boldsymbol{v}|\!|\!|^2 \equiv \boldsymbol{a}(\boldsymbol{v}, \boldsymbol{v}),$$

where the right hand side is the sum of the contributions from the two integrals of $\boldsymbol{a}(\cdot,\cdot)$. We observe exponential convergence, as is to be expected, considering that the domain in this case is represented exactly, and body force and solution are analytic functions. The error stops decreasing when it reaches the tolerance level of the CG iterations, which is set to 10^{-12}.

3.2 Steady Stokes flow

The Stokes equations are used to model creeping fluid flow, where viscous forces dominate inertial forces [32]. They are a simplification of the Navier-Stokes

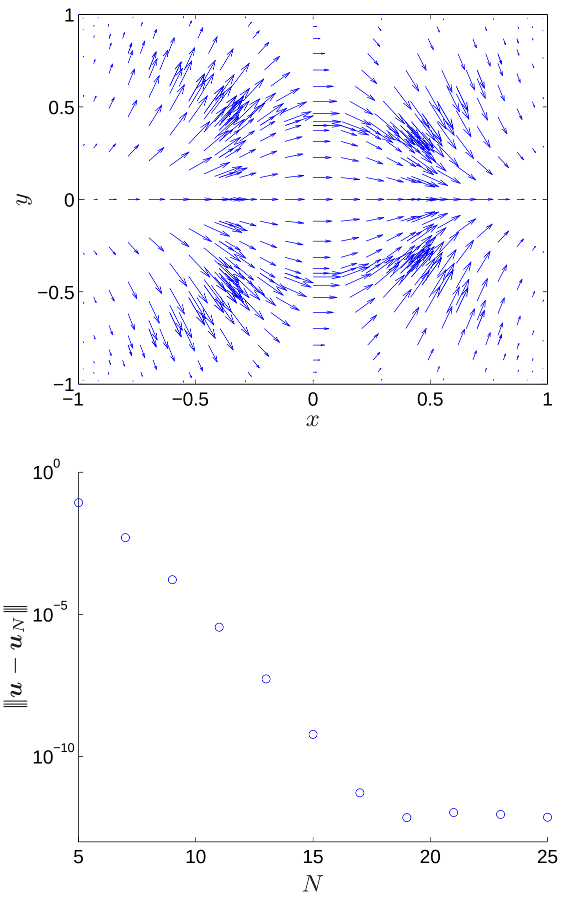


Figure 3.1: *Top: numerical solution of the viscosity model problem, using polynomial degree $N = 10$. The five-element structure gives a higher density of vectors along the boundaries between the elements. Bottom: error, measured in the energy norm, as a function of the polynomial degree N. The exact solution is analytic and convergence is exponential. The error levels out when it reaches the tolerance level of the CG iterations.*

equations in that the convection term is disregarded. For steady Stokes flow the unsteady terms vanish as well, and we end up with the equations

$$\begin{aligned}
-\mu \frac{\partial}{\partial x_j}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right) + \frac{\partial p}{\partial x_i} &= f_i \qquad \text{in } \Omega, \quad i = 1, 2, \\
\frac{\partial u_j}{\partial x_j} &= 0 \qquad \text{in } \Omega, \\
\boldsymbol{u} &= \boldsymbol{g} \qquad \text{on } \partial\Omega,
\end{aligned} \tag{3.5}$$

where we have prescribed inhomogeneous Dirichlet boundary conditions. There are no boundary conditions for the pressure. First, note that with these boundary conditions there is a solvability condition that needs to be satisfied for the problem to have a solution. Conservation of mass and the divergence theorem imply that

$$0 = \int_\Omega \nabla \cdot \boldsymbol{u} \, \mathrm{d}\Omega = \int_{\partial\Omega} \boldsymbol{u} \cdot \boldsymbol{n} \, \mathrm{d}S = \int_{\partial\Omega} \boldsymbol{g} \cdot \boldsymbol{n} \, \mathrm{d}S \tag{3.6}$$

This condition is always satisfied for homogeneous and periodic boundary conditions.

There is also a question of uniqueness that should be addressed. Consider two solutions $(\boldsymbol{u}_1, p_1)$ and $(\boldsymbol{u}_2, p_2)$ of (3.5). Due to linearity of the equations, $(\boldsymbol{w}, q) = (\boldsymbol{u_1} - \boldsymbol{u_2}, p_1 - p_2)$ represents a solution to the problem

$$\begin{aligned} -\mu \frac{\partial}{\partial x_j}\left(\frac{\partial w_i}{\partial x_j} + \frac{\partial w_j}{\partial x_i}\right) + \frac{\partial q}{\partial x_i} &= 0 \qquad \text{in } \Omega, \quad i = 1, 2, \\ \frac{\partial w_j}{\partial x_j} &= 0 \qquad \text{in } \Omega, \\ \boldsymbol{w} &= \boldsymbol{0} \qquad \text{on } \partial\Omega. \end{aligned} \tag{3.7}$$

Multiplying the momentum equations with w_i and integrating over Ω, we get

$$-\mu \int_\Omega w_i \frac{\partial}{\partial x_j}\left(\frac{\partial w_i}{\partial x_j} + \frac{\partial w_j}{\partial x_i}\right) \mathrm{d}\Omega + \int_\Omega w_i \frac{\partial q}{\partial x_i} \, \mathrm{d}\Omega = 0, \qquad i = 1, 2.$$

Integrating by parts and using the homogeneous boundary conditions of (3.7), we get

$$\mu \int_\Omega \frac{\partial w_i}{\partial x_j}\left(\frac{\partial w_i}{\partial x_j} + \frac{\partial w_j}{\partial x_i}\right) \mathrm{d}\Omega + \int_\Omega \frac{\partial w_i}{\partial x_i} q \, \mathrm{d}\Omega = 0, \qquad i = 1, 2.$$

Note that there is no summation over the index i, so there is only one partial derivative in the last integral. Adding the two momentum equations together,

$$\mu \sum_{i=1}^{2} \int_\Omega \left(\frac{\partial w_i}{\partial x_j}\right)^2 + \left(\frac{\partial w_j}{\partial x_j}\right)^2 \mathrm{d}\Omega + \sum_{i=1}^{2} \int_\Omega \frac{\partial w_i}{\partial x_i} q \, \mathrm{d}\Omega = 0,$$

we get a divergence term in both integrals, which vanishes due to the incompressibility condition. The other two terms in the first integral are non-negative and must therefore be everywhere zero for the equation to hold. Hence the gradients are everywhere zero,

$$\nabla w_i = 0 \qquad i = 1, 2,$$

and the solution is a constant in Ω. The homogeneous boundary conditions then imply that $\boldsymbol{w} = \boldsymbol{0}$, and we conclude that

$$\boldsymbol{u}_1 = \boldsymbol{u}_2.$$

Now the momentum equations in (3.7) simplify to

$$\frac{\partial q}{\partial x_i} = 0, \qquad i = 1, 2$$

meaning that q is constant in Ω. But since we have no boundary condition "anchoring" the pressure, it is only determined up to a constant. Hence, for any constant p_0 we may have

$$p_2 = p_1 + p_0.$$

p_0 is called the hydrostatic mode.

3.2.1 Weak form

The relevant solution spaces for the Stokes problem (3.5), now with $\boldsymbol{g} = \boldsymbol{0}$ for simplicity, are

$$X = \{\boldsymbol{v} \in (H^1(\Omega))^2 \mid \boldsymbol{v}|_{\partial\Omega} = \boldsymbol{0}\} = (H_0^1(\Omega))^2$$
$$Y = \{q \in L^2(\Omega) \mid \int_\Omega q \, \mathrm{d}\Omega = 0\} = L_0^2(\Omega)$$

The pressure is not required to be continuous, which is due to the fact that no partial derivatives of the pressure are present in the weak formulation of the problem. The weak formulation is derived in the same way as before, and states: find $\boldsymbol{u} \in X$ and $p \in Y$ such that

$$\begin{aligned} \boldsymbol{a}(\boldsymbol{v}, \boldsymbol{u}) - \boldsymbol{d}(\boldsymbol{v}, p) &= (\boldsymbol{v}, \boldsymbol{f}) \qquad && \forall \boldsymbol{v} \in X, \\ -\boldsymbol{d}(\boldsymbol{u}, q) &= \boldsymbol{0} && \forall q \in Y. \end{aligned} \tag{3.8}$$

Here $\boldsymbol{a} : X \times X \to \mathbb{R}^2$ is the viscous operator, as defined in (3.4), and $\boldsymbol{d} : X \times Y \to \mathbb{R}^2$ is a bilinear form

$$\boldsymbol{d}(\boldsymbol{v}, q) = \int_\Omega q \frac{\partial v_i}{\partial x_i} \, \mathrm{d}\Omega, \qquad i = 1, 2. \tag{3.9}$$

3.2.2 Discretization

We will apply a Legendre spectral element method to solve (3.8). Both pressure and velocity will be approximated by polynomials, but we will *not* use equal order approximations. This will lead to a non-trivial null space for $\boldsymbol{d}(\cdot, \cdot)$, consisting of *spurious pressure modes*, which will be hidden by $\boldsymbol{d}(\cdot, \cdot)$ in the equations [3].

Instead we will apply the so-called $\mathbb{P}_N - \mathbb{P}_{N-2}$ method [25]. This involves the discrete spaces

$$\begin{aligned} X_N &= \{v \in X \mid v\big|_{\Omega_k} \circ \mathcal{F}_k \in (\mathbb{P}_N(\widehat{\Omega}))^2\} \\ Y_N &= \{q \in Y \mid q\big|_{\Omega_k} \circ \mathcal{F}_k \in (\mathbb{P}_{N-2}(\widehat{\Omega}))^2\} \end{aligned} \tag{3.10}$$

where $\widehat{\Omega} = ((-1, 1))^2$ is the reference domain. These spaces can be proved to be compatible [4]; they guarantee a unique solution and no spurious pressure modes. This will not be proved here.

The pressure will be a sum of Lagrangian interpolants in the same way as the velocity is. Functions $q \in Y_N$ can be expanded in a nodal basis on the reference domain associated with each spectral element,

$$\hat{q}^k(\xi, \eta) = \sum_{m=1}^{N-1} \sum_{n=1}^{N-1} q_{mn}^k \tilde{\ell}_m(\xi) \tilde{\ell}_n(\eta), \qquad k = 1, \ldots, K,$$

where $\tilde{\ell}_i(\xi) \in \mathbb{P}_{N-2}(\widehat{\Omega})$. The tilde is used to show that these basis functions are different from the polynomials $\ell_i(\xi)$ in the basis for X_N.

The order of the polynomials is different, and this affects the nodal points. If there are $N + 1$ nodal points in each spatial direction for the velocity to be represented in, there are only $N - 1$ for the pressure. One possible choice of points could be the internal GLL points. A better choice, however, are the Gauss-Legendre (GL) points corresponding to a polynomial degree $N-2$. These points are not the same as the internal GLL points, so the grids will be staggered. We use ξ_i to denote the GLL points and ζ_i to denote the GL points. Note that

$$\ell_i(\xi_j) = \delta_{ij} \qquad \text{and} \qquad \tilde{\ell}_i(\zeta_j) = \delta_{ij}.$$

The use of GL points makes use of GL quadrature natural, which is what we will do for the gradient and divergence terms. The result is an algebraic system of equations

$$\begin{aligned} \boldsymbol{A}\boldsymbol{u} - \boldsymbol{D}^T\boldsymbol{p} &= \boldsymbol{B}\boldsymbol{f} \\ -\boldsymbol{D}\boldsymbol{u} \qquad\quad &= \boldsymbol{0} \end{aligned} \tag{3.11}$$

where $\boldsymbol{A}$ represents the discrete viscous operator, $\boldsymbol{D}^T$ the discrete gradient operator, and $\boldsymbol{D}$ the discrete divergence operator. For more details, see for example [11].

3.2.3 The Uzawa algorithm

The system matrix in the discrete Stokes problem (3.11) can be written on block matrix form as

$$\begin{bmatrix} \boldsymbol{A} & -\boldsymbol{D}^T \\ -\boldsymbol{D} & \boldsymbol{0} \end{bmatrix}. \tag{3.12}$$

The matrix $\boldsymbol{A}$ is both symmetric and positive definite, and this system matrix is clearly symmetric. However, it is no longer positive definite, and the eigenvalues are now both positive and negative. The algebraic system does no longer correspond to a minimization problem, it now represents a *symmetric saddle problem* [31]. The conjugate gradient algorithm can no longer be applied. However, the *Uzawa algorithm* [2, 6] will be used to transform the algebraic system (3.11) to a form that allows use of the CG algorithm. Using block Gaussian elimination we arrive at system

$$\begin{bmatrix} \boldsymbol{A} & -\boldsymbol{D}^T \\ \boldsymbol{0} & \boldsymbol{D}\boldsymbol{A}^{-1}\boldsymbol{D}^T \end{bmatrix} \begin{bmatrix} \boldsymbol{u} \\ \boldsymbol{p} \end{bmatrix} = \begin{bmatrix} \boldsymbol{B}\boldsymbol{f} \\ -\boldsymbol{D}\boldsymbol{A}^{-1}\boldsymbol{B}\boldsymbol{f} \end{bmatrix}, \tag{3.13}$$

which is triangular and can be solved. First, consider the second equation. The matrix $\boldsymbol{D}\boldsymbol{A}^{-1}\boldsymbol{D}^T$ is symmetric positive semi-definite, and the system can be solved for $\boldsymbol{p}$ using CG iterations. However, this requires operator evaluations

$$\boldsymbol{w} = \boldsymbol{D}\boldsymbol{A}^{-1}\boldsymbol{D}^T\boldsymbol{p}, \tag{3.14}$$

which must be done in three steps:

1. $\boldsymbol{w}_1 = \boldsymbol{D}^T\boldsymbol{p}$
2. $\boldsymbol{w}_2 = \boldsymbol{A}^{-1}\boldsymbol{w}_1$, which is found by solving $\boldsymbol{A}\boldsymbol{w}_2 = \boldsymbol{w}_1$. $\boldsymbol{A}$ is SPD, so the system can be solved with the CG algorithm.
3. $\boldsymbol{w} = \boldsymbol{D}\,\boldsymbol{w}_2$

The use of the CG algorithm inside the CG iterations gives rise to the name *nested CG iterations*. Once this system is solved for $\boldsymbol{p}$, we solve the system

$$\boldsymbol{A}\boldsymbol{u} = \boldsymbol{D}^T\boldsymbol{p} + \boldsymbol{B}\boldsymbol{f}$$

to find $\boldsymbol{u}$, again with CG iterations.

3.2.4 A numerical example: steady Stokes flow

We test the algorithm presented in the previous section on a simple model problem. The problem has an analytic solution

$$\begin{aligned} u_1 &= \sin(\pi x)\sin(\pi y), \\ u_2 &= \cos(\pi x)\cos(\pi y), \\ p &= \sin(\pi x)\cos(\pi y), \end{aligned}$$

and the problem domain is simply $\Omega = ((-1,1))^2$. We will use a multi-element approach, of the same reasons as before.

This problem does not have homogeneous boundary conditions. The system to solve for the pressure then becomes

$$\boldsymbol{D}\boldsymbol{A}^{-1}\boldsymbol{D}^T\boldsymbol{p} = -\boldsymbol{D}\boldsymbol{A}^{-1}\left(\boldsymbol{B}\boldsymbol{f} - \boldsymbol{A}\boldsymbol{g}\right) - \boldsymbol{D}\boldsymbol{g}, \tag{3.15}$$

where $\boldsymbol{g}$ is a matrix with the boundary values in the entries corresponding to grid nodes on the surface, and zeros in the other nodes.

Figure 3.2 shows the error in both pressure and velocity. The error in the velocity is measured in the energy norm, while the error in the pressure is measured in the L^2-norm. These are natural norms for the respective function spaces where we find $\boldsymbol{u}$ and p. Convergence is exponential as expected, until it reaches the tolerance level of the CG iterations.

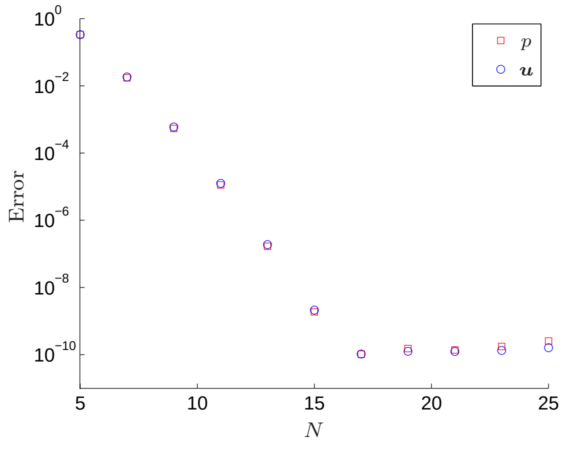


Figure 3.2: *Error in the numerical solution of the model problem as a function of the polynomial degree N, measured in the energy norm. The exact solution is analytic and convergence is exponential. The error stabilizes on the tolerance level of the CG iterations (10^{-12}).*

The algebraic system that is solved to find the pressure, can with success be preconditioned. To see how, observe that the continuous equivalence to the pressure-operator (3.14) is $\nabla\cdot(\nabla^2)^{-1}\nabla$. Heuristically [22], this is similar to the identity operator I,

$$\nabla\cdot(\nabla^2)^{-1}\nabla \sim I.$$

The discrete version of the similarity above is

$$\boldsymbol{D}\boldsymbol{A}^{-1}\boldsymbol{D}^T \sim \widetilde{\boldsymbol{B}}.$$

Hence, we can use the mass matrix $\widetilde{\boldsymbol{B}}$ as a preconditioning matrix. The tilde is used to indicate that this is the mass matrix on the GL grid, where the pressure is computed. This is necessary for the matrix dimensions to be correct. As the Lagrangian interpolants $\tilde{\ell}_i(\zeta)$ in the GL points are orthogonal in the discrete inner product, $\widetilde{\boldsymbol{B}}$ is diagonal and hence easily invertible. Preconditioning involves matrix-vector products with the inverse $\widetilde{\boldsymbol{B}}^{-1}$, which can be done element-wise as

$$\{\widetilde{\boldsymbol{B}}^{-1}\boldsymbol{v}\}_{ij} = \frac{v_{ij}}{\omega_i \omega_j \widetilde{J}_{ij}},$$

where ω_i is the GL weight corresponding to the GL point ζ_i, and $\widetilde{J}_{ij}$ is the Jacobian computed in the GL grid point (ζ_i, ζ_j). As this operation does not require any matrix-matrix products, it is done in $\mathcal{O}(N^2)$ operations. Since operator evaluation in the CG iterations requires $\mathcal{O}(N^3)$ operations, preconditioning does not add any significant computation time to the algorithm.

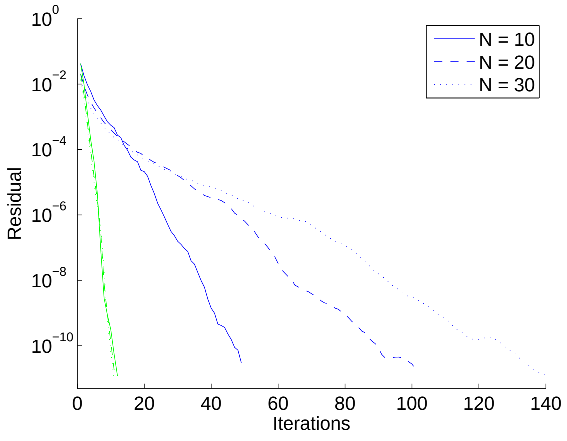


Figure 3.3: *Residual during CG iterations for different polynomial degrees, i.e. problem sizes. With preconditioning, the number of iterations needed for convergence does not depend on the problem size.*

Figure 3.3 shows the residual during the CG iterations as a function of the iteration number. We see that the convergence properties of the unconditioned system depend on the problem size, i.e. the dimension of the matrices in the system solved. For the preconditioned system, the number of iterations needed to reach the desired residual is seemingly independent of the problem size. Convergence in the CG iterations depends on the condition number of the system matrix [30], and preconditioning with $\widetilde{\boldsymbol{B}}$ keeps the condition number constant. For a polynomial degree $N = 10$, preconditioning gives 4 times less iterations, while for $N = 30$, it gives 14 times less iterations. Considering that the computation time for one iteration is approximately the same with or without preconditioning, this ratio also applies to the CPU time in the numerical simulations.

3.3 Unsteady Stokes flow

The next step towards the Navier-Stokes equations is adding the partial derivative of the velocity with respect to time, resulting in an unsteady Stokes problem. The strong form of the problem is

$$\begin{aligned} \rho\frac{\partial u_i}{\partial t} - \mu\frac{\partial}{\partial x_j}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right) + \frac{\partial p}{\partial x_i} &= f_i \qquad \text{in } \Omega, \quad i = 1, 2, \\ \frac{\partial u_j}{\partial x_j} &= 0 \qquad \text{in } \Omega, \end{aligned} \tag{3.16}$$

with boundary conditions

$$\begin{aligned} \boldsymbol{u} &= \boldsymbol{g} && \text{on } \partial\Omega, \\ \boldsymbol{u} &= \boldsymbol{u_0}(x, y) && \text{at } t = 0. \end{aligned} \tag{3.17}$$

For simplicity we set $\rho = 1$. The weak formulation is the same as that of the steady problem (3.8), apart from an added term with the time-derivative. Since the geometry and the test functions are not time-dependent, this term simplifies to a normal derivative outside the integral,

$$\int_\Omega \boldsymbol{v}\frac{\partial \boldsymbol{u}}{\partial t}\,\mathrm{d}\Omega = \frac{\mathrm{d}}{\mathrm{d}t}\int_\Omega \boldsymbol{v}\boldsymbol{u}\,\mathrm{d}\Omega,$$

after integration by parts. Applying the same $\mathbb{P}_N - \mathbb{P}_{N-2}$ method for spatial discretization as we did for the steady problem, we arrive at a semi-discrete system of ordinary differential equations:

$$\begin{aligned} \boldsymbol{B}\frac{\mathrm{d}\boldsymbol{u}}{\mathrm{d}t} + \boldsymbol{A}\boldsymbol{u} - \boldsymbol{D}^T\boldsymbol{p} &= \boldsymbol{B}\boldsymbol{f}, \\ -\boldsymbol{D}\boldsymbol{u} &= \boldsymbol{0}. \end{aligned} \tag{3.18}$$

Temporal discretization of the first equation is done with a second order backward differentiation scheme

$$\boldsymbol{B}\frac{3\boldsymbol{u}^{n+1} - 4\boldsymbol{u}^n + \boldsymbol{u}^{n-1}}{2\Delta t} + \boldsymbol{A}\boldsymbol{u}^{n+1} - \boldsymbol{D}^T\boldsymbol{p}^{n+1} = \boldsymbol{B}\boldsymbol{f}^{n+1}.$$

Rearranging the equation in order to solve it for $\boldsymbol{u}^{n+1}$, we obtain

$$(\boldsymbol{A} + \frac{3}{2\Delta t}\boldsymbol{B})\boldsymbol{u}^{n+1} = \boldsymbol{D}^T\boldsymbol{p}^{n+1} + \boldsymbol{B}\left(\boldsymbol{f}^{n+1} + \frac{2}{\Delta t}\boldsymbol{u}^n - \frac{1}{2\Delta t}\boldsymbol{u}^{n-1}\right). \tag{3.19}$$

The Helmholtz like operator is SPD, since both $\boldsymbol{A}$ and $\boldsymbol{B}$ are SPD. Hence this system can be solved with conjugate gradient iterations. However, this requires $\boldsymbol{p}^{n+1}$ to be known. We find the pressure in the same way as before, with the block Gaussian elimination of the Uzawa algorithm. The resulting system

$$\boldsymbol{D}(\boldsymbol{A}+\frac{3}{2\Delta t}\boldsymbol{B})^{-1}\boldsymbol{D}^T\boldsymbol{p}^{n+1} = -\boldsymbol{D}(\boldsymbol{A}+\frac{3}{2\Delta t}\boldsymbol{B})^{-1}\boldsymbol{B}\left(\boldsymbol{f}^{n+1} + \frac{2}{\Delta t}\boldsymbol{u}^n - \frac{1}{2\Delta t}\boldsymbol{u}^{n-1}\right)$$

can be solved for $\boldsymbol{p}^{n+1}$ with a nested CG algorithm, as shown in the previous section. We can then solve (3.19) for $\boldsymbol{u}^{n+1}$.

3.3.1 A numerical example: unsteady Stokes flow

We solve a problem with exact solution

$$\begin{aligned} u_1 &= \sin(\pi x)\sin(\pi y)(1-e^{-t}), \\ u_2 &= \cos(\pi x)\cos(\pi y)(1-e^{-t}), \\ p &= \sin(\pi x)\cos(\pi y)(1-e^{-t}), \end{aligned}$$

on a circular domain of radius 1 with center in the origin. Inhomogeneous boundary conditions are prescribed in accordance with the exact solution. Both the fluid density ρ and the dynamic viscosity μ are equal to unity.

The method applied is the one described in the previous section, with a second order temporal discretization scheme. The inhomogeneous boundary conditions result in modifications of the method similar to (3.15).

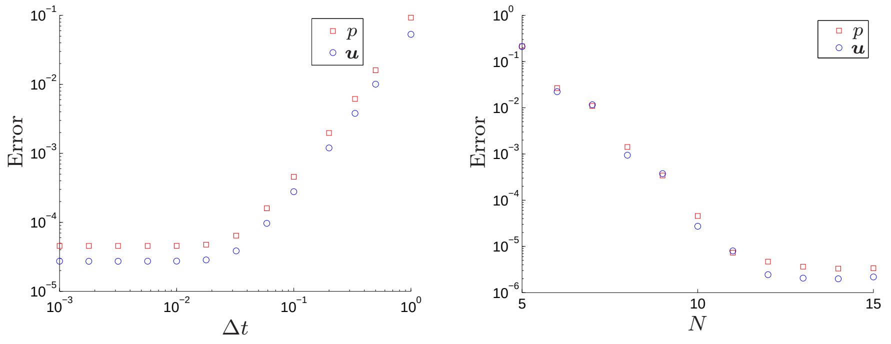


Figure 3.4: *Error in the numerical solution at time $t = 1$. The velocity is measured in the energy norm, the pressure in the L^2-norm. Left: error as a function of the step length Δt for $N = 10$. Convergence is second order for both pressure and velocity until $\Delta t \approx 10^{-2}$, where the spatial discretization error is no longer subdominant the temporal discretization error. Right: error as a function of the polynomial degree N, with a step length $\Delta t = 10^{-3}$. The temporal discretization error is subdominant the spatial discretization error for $N < 12$.*

Second order convergence in time is verified for both pressure and velocity in Figure 3.4. The temporal discretization error dominates the spatial discretization error until $\Delta t \approx 10^{-2}$, where the spatial discretization error becomes dominant. The same Figure also shows exponential convergence in space. Here we have the opposite case: the spatial discretization error dominates the temporal discretization error until $N \approx 12$.

3.4 Free surface Stokes flow

A free surface is a surface that is subject to neither normal nor tangential stress. Surface tension is a property of the surface of a liquid that makes it behave like an elastic sheet, and it is often neglected in large scale models. Here, we will assume that surface tension forces *are* significant, and we will consider the surface between two immiscible incompressible fluids, one of which is viscous and one inviscid. Free surface conditions can then be written as [19]

$$\begin{aligned} n_i\sigma_{ij}n_j &= \gamma\kappa - p_0 \qquad && \text{on } \partial\Omega, \\ t_i\sigma_{ij}n_j &= \frac{\mathrm{d}\gamma}{\mathrm{d}s} && \text{on } \partial\Omega, \end{aligned} \tag{3.20}$$

where $\frac{\mathrm{d}\gamma}{\mathrm{d}s}$ is the derivative of the surface tension along the free surface, κ is the local curvature, p_0 is the external pressure and n_i and t_i are the i-th components of the unit normal and tangential vectors, respectively. The stress tensor σ_{ij} is for a Newtonian fluid given by [1]

$$\sigma_{ij} = -p\delta_{ij} + \mu\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right), \qquad i,j = 1,2.$$

The unsteady Stokes equations can be written in stress tensor form, using indicial notation, as

$$\begin{aligned} \rho\frac{\partial u_i}{\partial t} - \frac{\partial \sigma_{ij}}{\partial x_j} &= f_i \qquad && \text{in } \Omega, \quad i = 1,2, \\ \frac{\partial u_j}{\partial x_j} &= 0 && \text{in } \Omega. \end{aligned} \tag{3.21}$$

The relevant function spaces for the weak form are:

$$X = \{v \mid \forall t \in [0,T],\ v(x,y;t) \in H^1(\Omega),\ \int_0^T \|v\|^2_{H^1(\Omega)}\,\mathrm{d}t < \infty\}$$

$$Y = \{q \mid \forall t \in [0,T],\ q(x,y;t) \in L^2(\Omega),\ \int_0^T \|v\|^2_{L^2(\Omega)}\,\mathrm{d}t < \infty\}$$

In deriving the weak formulation, we multiply by test functions and integrate over Ω. The weak formulation in indicial notation reads: find $u_i \in X, i = 1,2$ and $p \in Y$ such that

$$\begin{aligned} \rho\int_\Omega v_i\frac{\partial u_i}{\partial t}\,\mathrm{d}\Omega + \int_\Omega \frac{\partial v_i}{\partial x_j}\sigma_{ij}\,\mathrm{d}\Omega &= \int_\Omega v_i f_i\,\mathrm{d}\Omega + \int_{\partial\Omega} v_i\sigma_{ij}n_j\,\mathrm{d}s \quad && \forall v_i \in X, \\ \int_\Omega q\frac{\partial u_j}{\partial x_j}\,\mathrm{d}\Omega &= 0 && \forall q \in Y. \end{aligned} \tag{3.22}$$

3.4.1 ALE formulation

Now we derive the ALE formulation of the free surface Stokes problem (3.22). The procedure is very similar to the derivation of the ALE formulation of the convection-diffusion problem. Applying integration by parts and Reynold's transport theorem, the first term in (3.22) becomes

$$\int_\Omega v_i\frac{\partial u_i}{\partial t}\,\mathrm{d}\Omega = \frac{\mathrm{d}}{\mathrm{d}t}\int_\Omega v_i\,u_i\,\mathrm{d}\Omega - \int_{\partial\Omega} v_i u_i u_j n_j\,\mathrm{d}s - \int_\Omega u_i\frac{\partial v_i}{\partial t}\,\mathrm{d}\Omega.$$

Using the kinematic condition and the divergence theorem, the second term on the right hand side can be written

$$\int_{\partial\Omega} v_i u_i u_j n_j\,\mathrm{d}s = \int_\Omega \frac{\partial v_i}{\partial x_j}u_i w_j + v_i\frac{\partial u_i}{\partial x_j}w_j + v_i u_i\frac{\partial w_j}{\partial x_j}\,\mathrm{d}\Omega.$$

Inserting this into the weak formulation of the Stokes problem, and integrating the stress tensor term by parts, we arrive at

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t}\int_\Omega v_i\, u_i \,\mathrm{d}\Omega - \int_\Omega \frac{\partial v_i}{\partial x_j} u_i w_j + v_i \frac{\partial u_i}{\partial x_j} w_j + v_i u_i \frac{\partial w_j}{\partial x_j}\,\mathrm{d}\Omega - \int_\Omega u_i \frac{\partial v_i}{\partial t}\,\mathrm{d}\Omega \\ - \int_\Omega \frac{\partial v_i}{\partial x_j}\sigma_{ij}\,\mathrm{d}\Omega = \int_\Omega v_i f_i \,\mathrm{d}\Omega + \int_{\partial\Omega} v_i \sigma_{ij} n_j \,\mathrm{d}s. \quad (3.23)\end{aligned}$$

The derivative of the test functions following the domain velocity is zero,

$$\frac{Dv_i}{Dt} = \frac{\partial v_i}{\partial t} + w_j \frac{\partial v_i}{\partial x_j} = 0,$$

which makes two of the terms on the left hand side vanish, and we end up with the following ALE formulation (returning to vector notation): find $\boldsymbol{u} \in (X)^2$ and $p \in Y$ such that

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t}(\boldsymbol{v},\boldsymbol{u}) + \boldsymbol{a}(\boldsymbol{v},\boldsymbol{u}) + \boldsymbol{c}(\boldsymbol{v},\boldsymbol{u}) - \boldsymbol{d}(\boldsymbol{v},p) - \boldsymbol{e}(\boldsymbol{v},\boldsymbol{u}) \\ = (\boldsymbol{v}, f) + I_\gamma(\boldsymbol{v}) \quad \forall v \in (X)^2, \qquad (3.24)\end{aligned}$$

where $\boldsymbol{a}(\cdot,\cdot)$ is the viscous term (3.4), $\boldsymbol{d}(\cdot,\cdot)$ as defined in (3.9) denotes both the gradient and the divergence terms, and $I_\gamma(\boldsymbol{v})$ denotes the surface integral. The ALE formulation introduces two new operators, very similar to the operators (1.7) in the ALE formulation of the convection-diffusion problem. Here they consist of two integrals, one for each component of $\boldsymbol{u}$:

$$\begin{aligned}\boldsymbol{c}(\boldsymbol{v},\boldsymbol{u}) &= -\int_\Omega v_i w_j \frac{\partial u_i}{\partial x_j}\,\mathrm{d}\Omega, \qquad & i = 1,2, \\ \boldsymbol{e}(\boldsymbol{v},\boldsymbol{u}) &= \int_\Omega v_i u_i \frac{\partial w_j}{\partial x_j}\,\mathrm{d}\Omega, \qquad & i = 1,2.\end{aligned}$$

The ALE formulation of the linear momentum equations is of course accompanied by the incompressibility equation, which is the same as in the weak form.

3.4.2 Surface integral

Let us for simplicity assume an ambient pressure $p_0 = 0$. The total surface tension forces on the boundary are given by

$$\sigma_{ij} n_j = \gamma \kappa n_i + \frac{\mathrm{d}\gamma}{\mathrm{d}s} t_i, \qquad i = 1,2.$$

This, together with the definition of the curvature,

$$\kappa\, n_i = \frac{\mathrm{d}t_i}{\mathrm{d}s},$$

enables us to transform the surface integral to a convenient form:

$$\begin{aligned}
\int_{\partial\Omega} v_i \sigma_{ij} n_j \,\mathrm{d}s &= \int_{\partial\Omega} v_i \left(\gamma \kappa n_i + \frac{\mathrm{d}\gamma}{\mathrm{d}s} t_i \right) \mathrm{d}s \\
&= \int_{\partial\Omega} v_i \left(\gamma \frac{\mathrm{d}t_i}{\mathrm{d}s} + \frac{\mathrm{d}\gamma}{\mathrm{d}s} t_i \right) \mathrm{d}s \\
&= \int_{\partial\Omega} v_i \frac{\mathrm{d}(\gamma t_i)}{\mathrm{d}s} \,\mathrm{d}s \\
&= [\gamma v_i t_i]_a^b - \int_{\partial\Omega} \gamma \frac{\mathrm{d}v_i}{\mathrm{d}s} t_i \,\mathrm{d}s \\
&= - \int_{\partial\Omega} \gamma \frac{\mathrm{d}v_i}{\mathrm{d}s} \frac{\mathrm{d}x_i}{\mathrm{d}s} \,\mathrm{d}s,
\end{aligned}$$

where a and b denote the start and end of the free surface. Here the free surface is a closed surface, so this term vanishes. The last integral includes all surface tension contributions, both tangential and normal.

3.4.3 Discretization

Discretization is merely a matter of combining the techniques presented so far. Spatial discretization is done with the same Legendre spectral element method as before, with the same discrete spaces (3.10). The resulting system of ordinary differential equations is

$$\begin{aligned}
\boldsymbol{B}\frac{\mathrm{d}\boldsymbol{u}}{\mathrm{d}t} + \boldsymbol{A}\boldsymbol{u} + \boldsymbol{C}\boldsymbol{u} - \boldsymbol{E}\boldsymbol{u} - \boldsymbol{D}^T\boldsymbol{p} &= \boldsymbol{B}\boldsymbol{f} + \boldsymbol{g}, \\
-\boldsymbol{D}\boldsymbol{u} &= \boldsymbol{0},
\end{aligned}$$

where $\boldsymbol{g}$ is a vector containing the value of the surface integral $I_\gamma(\boldsymbol{v})$ for each test function.

For temporal discretization we apply the same semi-implicit BD2 scheme as we did for the convection-diffusion equation in (1.13). The discrete system then becomes

$$\begin{aligned}
\frac{3\,\boldsymbol{B}^{n+1}\boldsymbol{u}^{n+1} - 4\,\boldsymbol{B}^{n}\boldsymbol{u}^{n} + \boldsymbol{B}^{n-1}\boldsymbol{u}^{n-1}}{2\Delta t} &+ \boldsymbol{A}^{n+1}\boldsymbol{u}^{n+1} - (\boldsymbol{D}^T)^{n+1}\boldsymbol{p}^{n+1} \\
&= \boldsymbol{B}^{n+1}\boldsymbol{f}^{n+1} + \boldsymbol{g}^{n+1} + \sum_{j=0}^{2} \beta_j (\boldsymbol{C}^{n-j} - \boldsymbol{E}^{n-j})\boldsymbol{u}^{n-j},
\end{aligned}$$

where the coefficients β_j are the same as before (1.14). The block Gaussian elimination of the Uzawa algorithm is now applied, and we end up with the following system to solve for the pressure:

$$\begin{aligned}
&\boldsymbol{D}^{n+1}(\boldsymbol{A}^{n+1} + \frac{3}{2\Delta t}\boldsymbol{B}^{n+1})^{-1}(\boldsymbol{D}^T)^{n+1}\boldsymbol{p}^{n+1} = \\
&\quad - \boldsymbol{D}^{n+1}(\boldsymbol{A}^{n+1} + \frac{3}{2\Delta t}\boldsymbol{B}^{n+1})^{-1}\Big(\boldsymbol{B}^{n+1}\boldsymbol{f}^{n+1} + \boldsymbol{g}^{n+1} + \frac{2}{\Delta t}\boldsymbol{B}^{n}\boldsymbol{u}^{n} \\
&\qquad - \frac{1}{2\Delta t}\boldsymbol{B}^{n-1}\boldsymbol{u}^{n-1} + \sum_{j=0}^{2}\beta_j(\boldsymbol{C}^{n-j} - \boldsymbol{E}^{n-j})\boldsymbol{u}^{n-j}\Big). \qquad (3.25)
\end{aligned}$$

The equation may seem huge and complex, but the only addition from the unsteady Stokes flow in the previous section are elements on the right hand side, which are treated explicitly. The system can be solved using nested CG iterations, as before.

3.4.4 A droplet: free surface Stokes flow

It is hard to construct free surface problems with exact solutions, but we can come up with an example where we easily can see if the solution is physically reasonable.

Consider a two-dimensional free surface Stokes problem with constant surface tension γ all over the surface $\partial\Omega$. This problem models a drop of fluid with a low Reynolds number. The natural equilibrium, in which surface tension is minimized, is a circular surface. In this state the fluid velocity will be zero, and the pressure will be constant all over Ω with a value

$$p = \frac{\gamma}{r}, \tag{3.26}$$

where r is the radius of the circle.

We let the initial domain be elliptical with major semi-axis $a = \sqrt{2}$ and minor semi-axis $b = 1/\sqrt{2}$. There are no body forces and no gravity, so the only force at work in the initial configuration is the surface tension force. Figure 3.5 shows the numerical solution for $\boldsymbol{u}$ at the initial configuration. Observe how the velocity field points the surface towards a more circular shape. The domain must neither shrink nor grow, since the incompressibility condition necessitates a constant area.

Figure 3.6 shows the domain at a few selected time levels. The transition towards a circular shape is fast in the beginning, then slower as we approach the steady state. In the final configuration, the domain is close to a circle with radius $r = 1$.

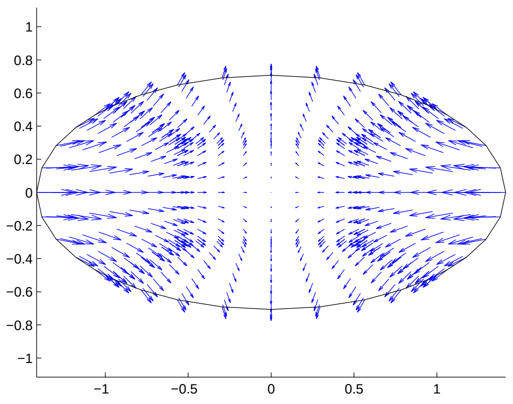


Figure 3.5: *Velocity field $\boldsymbol{u}$ computed at the initial configuration with $N = 10$ and μ, ρ and γ all set to unity. The field points the droplet towards a circular shape, in a way so that the area of Ω remains constant.*

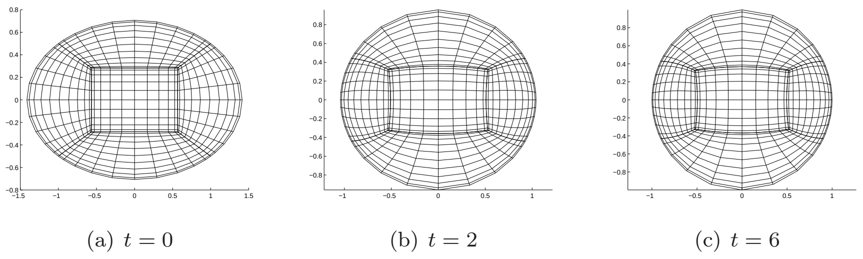


Figure 3.6: *Material domain in a free surface problem at selected time levels. The polynomial degree is $N = 10$, and the step length is $\Delta t = 0.03$. The initial domain is elliptical, but surface tension forces work towards an equilibrium, which is a circular domain.*

Figure 3.7 shows the numerical solutions for $\boldsymbol{u}$ and p at $t = 6$. At this point it is easy to see that the system is close to equilibrium. The domain is almost circular, and the velocity is close to zero, making the velocity vectors microscopic in the Figure. The pressure is almost constant, and deviations are in the fourth decimal. It approaches the constant $p = 1$, which is in agreement with (3.26), since $r \approx 1$ and the surface tension is set to $\gamma = 1$.

A closer look at Figure 3.6 reveals a problem with the GLL distribution of nodal points along the surface. This does not come as a surprise, considering our experience with the same issue in Section 2.2.3. The problem here is that all the nodal points on the surface tend to move toward $y = 0$. This is caused by the geometry of the problem: the domain velocity is largest at the beginning of the iterations, when the eccentricity of the ellipse is at its largest. Hence, the nodal points move most rapidly when the normal vector deviates the most from the radial vector, and they move toward the x-axis.

We recall that the solution to this problem was to add a tangential component to the domain velocity on the boundary, not violating the kinematic

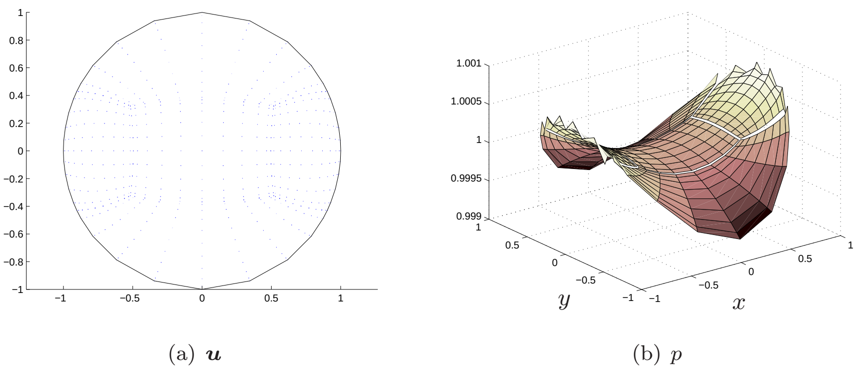


Figure 3.7: *Numerical solution for velocity $\boldsymbol{u}$ and pressure p at time $t = 6$. Simulations are done with $N = 10$ and $\Delta t = 0.03$. We are very close to equilibrium, with close to zero velocity and almost constant pressure.*

condition. Here, it is natural to add a component to make the domain velocity point toward the origin. After all, in the steady state, the surface is circular and the normal vector is everywhere coinciding with the radial vector.

Simulation reveals that this modification has positive effects. A plot of the major axis as a function of time shows no visible difference between the methods with and without the added tangential component. Hence, we conclude that the difference in the numerical solution with the two methods is minimal. However, a plot of the y-component of the nodal points on the part of the surface belonging to the element Ω_2 (see Figure 2.9) reveals a difference in the GLL distribution. That is, none of the methods perfectly preserve the GLL distribution, but Figure 3.8 shows that the method with a radial domain velocity retains a better distribution than the method with a strictly normal domain velocity on the boundary.

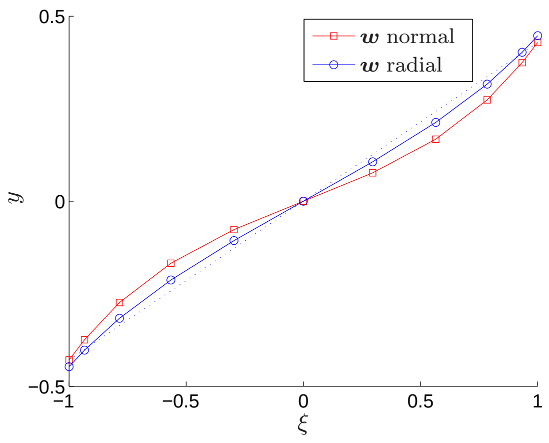


Figure 3.8: *Distribution of the nodal points along the part of the surface that belongs to the element Ω_2 at time level $t = 3$. The shape of the ellipse forces the points toward the line $y = 0$, and the resulting deviation from the GLL distribution can be seen as deviation from the straight line (dotted) in the plot. When adding a tangential component to $\boldsymbol{w}$ to make it radial, we achieve a better distribution of the nodal points.*

3.5 Free surface Navier-Stokes flow

We are now ready to simulate the full Navier-Stokes equations (3.1) with free surface conditions (3.20). There will be very little new to add here, as most of the theory and algorithms necessary already have been covered. In the strong form, the only difference from the unsteady Stokes problem (3.16) is the convection term

$$u_j \frac{\partial u_i}{\partial x_j}, \qquad i = 1, 2,$$

which will account for inertial forces. This term results in a modified convective term

$$\boldsymbol{c}(\boldsymbol{v}, \boldsymbol{u}) = \int_\Omega v_i (u_j - w_j) \frac{\partial u_i}{\partial x_j} \, \mathrm{d}\Omega, \qquad i = 1, 2,$$

in the ALE formulation (3.24), where the difference is the introduction of the particle velocity u_j in the convection field $u_j - w_j$. The rest of the terms in the ALE formulation are the same as before.

All aspects of the discretization are the same as before, both in space and time. Hence, the resulting algebraic system is the same as that for the free surface Stokes flow (3.25), only the discrete convection operator $\boldsymbol{C}$ is slightly different. The algebraic system will be solved using the Uzawa algorithm and nested CG iterations as before.

3.5.1 A droplet: free surface Navier-Stokes flow

We solve the same free surface problem as in Section 3.4.4, modeling a drop of a liquid. Using the Navier-Stokes equations, the fluid is now laminar, and convective effects are taken into account. The addition of convection results in inertial forces, making the fluid less viscous. The inertial forces will counteract change in velocity, making the drop deform in a damped oscillation, slowly stabilizing towards a steady state. The equilibrium is still a circular shape and constant pressure, since the surface tension is constant all along the surface. It is only the transition towards the steady state which is different.

We now have to consider the the balance between surface tension and convective forces, which is governed by the dimensionless *Capillary number*

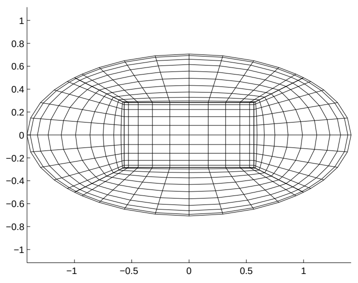
$$Ca = \frac{\mu^2}{\rho \gamma L},$$

where μ is the dynamic viscosity, ρ the fluid density, γ the surface tension and L a characteristic length, here set to unity. The Capillary number is a measure of viscous forces relative to surface tension forces.

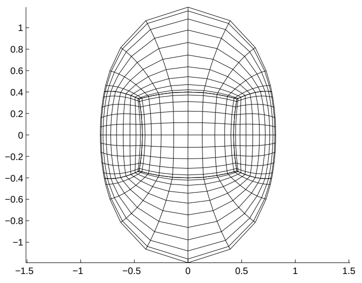
Some numerical testing has been done to find the right Capillary number for oscillations to show. Nice results are found for $Ca = 0.02$. A few grid configurations at selected time levels are shown in Figure 3.9. The time levels are chosen when there is maximum deformation, to make the damped oscillations more conspicuous. The symmetry of the problem makes the domain elliptic at all times, with no asymmetric deformations.

In order to quantify the oscillations, we follow the computational grid point on the boundary $\partial\Omega$ which also lies on the positive x-axis. The x-component of this point will also be the minor or major semi-axis of the ellipse, depending on where we are in the oscillations. Tracking this point during the time integration reveals the nature of the damped oscillations. Figure 3.10 shows the oscillation for three different Capillary numbers. We see that a decreasing Capillary number yields larger oscillations. For $Ca = 1$ the fluid is highly viscous and there are no oscillations.

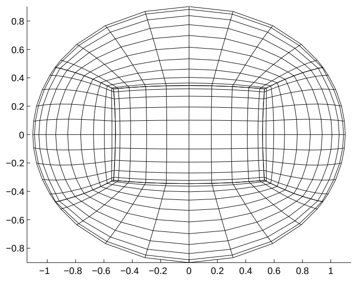
In the current model, fluids with different values of μ, ρ and γ, but with the same Capillary number, should display the same behavior. Numerical tests have been conducted to confirm this. Different values of the parameters have been used, but chosen such that the Capillary number remains constant. Both for $Ca = 1$ and $Ca = 0.02$, multiplying or dividing all the parameters by 10 did not change the oscillating curves seen in Figure 3.10.



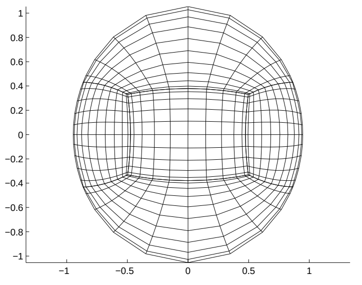
(a) $t = 0$



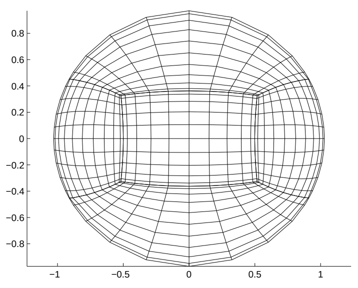
(b) $t = 1.38$



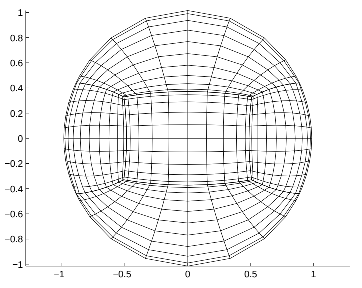
(c) $t = 2.75$



(d) $t = 4.09$



(e) $t = 5.43$



(f) $t = 6.77$

Figure 3.9: *Material domain in a free surface problem at selected time levels, when the deformation is at a local maximum. The polynomial degree is* $N = 10$*, and the step length is* $\Delta t = 10^{-2}$*. The introduction of convection forces the domain beyond the equilibrium and makes the drop behave elastic. The Capillary number is* $Ca = 0.02$*.*

3.5.2 A droplet with non-constant surface tension

The last two examples both had a constant surface tension. This is what makes the circular domain with zero velocity and constant pressure a steady state. It is interesting to see what the steady state will be like under conditions of non-constant surface tension. In the weak form, the surface integral includes only the surface tension itself, no partial derivatives. This makes it very easy to exchange a constant surface tension with a function that depends on geometric

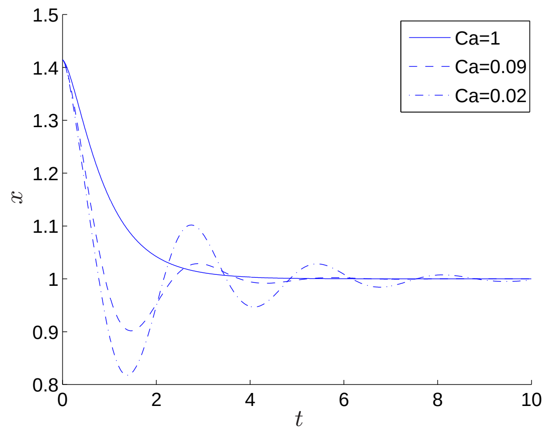


Figure 3.10: *The major semi-axis plotted against time shows the oscillating behavior for three different Capillary numbers. Surface tension causes the oscillations, while convective forces cause the damping. Hence, the smaller the Capillary number, the larger the oscillations. Regardless of the Capillary number, the major semi-axis stabilizes on 1 in the equilibrium state.*

factors:

$$\int_\Omega v_i \sigma_{ij} n_j \, \mathrm{d}s = -\int_{\partial\Omega} \gamma(s) \frac{\mathrm{d}v_i}{\mathrm{d}s} \frac{\mathrm{d}x_i}{\mathrm{d}s} \, \mathrm{d}s.$$

We must take care so that the surface tension is always positive and periodic along the surface. A function that satisfies these requirements on a closed surface is

$$\gamma(x, y) = \gamma_0 + \beta x,$$

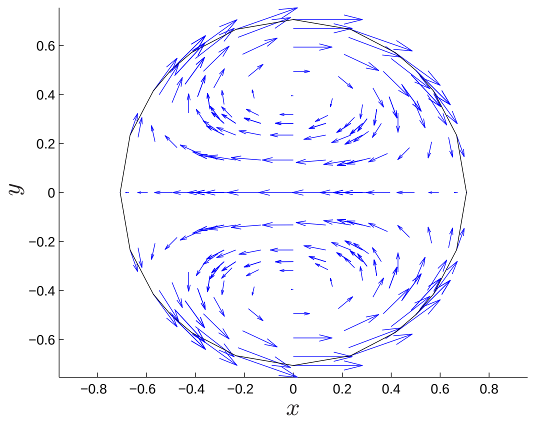
where γ_0 and β are constants.

We let the initial configuration be a circular domain, with zero velocity and constant pressure. With no body forces, the surface tension is what will initiate any movement. We want the surface tension forces and the viscous forces to balance, so that there is some convection to make it realistic, but not more than preventing us from reaching a potential steady state in reasonable time. The balance is achieved by letting $\mu = 1$, $\rho = 1$ and $\gamma_0 = 1$, so that the average of the surface tension is 1.

Several numerical tests have been conducted in order to see if the system approaches a steady state. The results are the same: the system moves towards an equilibrium in approximately the same speed as in the previous examples. In the steady state, the pressure is a linear function in x, and the velocity field creates two vortices. The vortices are symmetric about the x-axis, and they rotate in opposite directions (Figure 3.11). The domain is slightly deformed from the circular shape, but not so much that it is visible in the Figure. Changes in the radius occur in the fourth decimal.

This state is verified to be a steady state by running simulations for different amounts of time, with different spatial and temporal discretization parameters. Solutions are inspected and found to be almost identical.

Test have also been run with different values of the parameter β. Not surprisingly, a smaller β yields a smaller pressure gradient and smaller fluid velocities in the steady state. As $\beta \to 0$, we approach the the same equilibrium as in the previous examples.



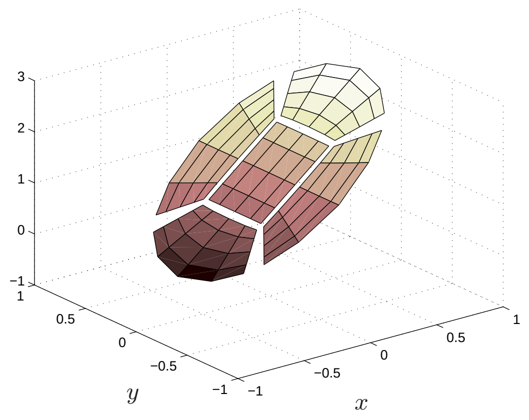


Figure 3.11: *The surface tension depends linearly on the geometry as* $\gamma = 1 + 0.8x$*. The other parameters in the problem are* $\mu = 1$*,* $\rho = 1$*,* $N = 6$ *and* $\Delta t = 0.04$*, and time integration is done from* $t_0 = 0$ *to* $t = 2$*. The time span is shorter than in the previous numerical example, but it is sufficient to show qualitatively what the steady state is. The initial configuration is a circular domain, and the surface tension results in a velocity field with a positive* x*-component all over the surface. This creates two counter-rotating vortices inside* Ω*. The bottom Figure shows that the pressure is a linear function in* x*.*

Conclusion and further work

A Legendre spectral method with an arbitrary Lagrangian-Eulerian formulation has been used to solve numerous heat transfer and fluid dynamics problems on time-dependent domains. In particular, the ALE formulation is derived for both the convection-diffusion equation and the Navier-Stokes equations, and a Legendre spectral discretization is used, first on a single element, then later with a multi-element approach. Important aspects of the numerical method are presented, including weak formulation, discretization and solution of the resulting algebraic system. Numerical examples accompany every new element as they are introduced, verifying the excellent convergence properties for spectral methods on smooth problems.

The convection-diffusion equation is used as a starting point for discussing the ALE formulation, as it is a relatively simple, linear equation, and its physical interpretation is intuitive. After the ALE formulation is derived, we discuss the Stefan problem and how the Stefan condition predicts the movement of the boundary of the problem domain, given the heat flux across the boundary. A useful technique is presented, in which the connection between Dirichlet and Neumann boundary conditions is utilized for determining this heat flux without having to compute normal vectors and temperature gradients. The method has proved highly efficient and is easily implemented numerically. Armed with this technique, we show an example of a simplified Stefan problem where the precision with which we are able to track the phase transition front, is determined by the order of the time integration scheme. First, second and third order convergence is shown.

The Stefan condition only gives information about what happens in the normal direction on the interface between the phases. Tangentially, we are free to add a component to the domain velocity; it does not violate the interface condition. The tangential component is added in order to keep a favorable grid configuration, preserving a certain regularity in the mapping between the reference domain and the physical domain. Since the tangential component is not subject to any condition, it is treated in a rather ad hoc fashion, based on whatever qualitative information we may have about the geometry. Strategies for choosing a tangential component are briefly discussed, and a numerical example is presented, in which the different strategies are tested, and their effect on the solution is studied.

In order to be able to handle domains that are not quadrilateral, we present the multi-element approach for dividing the domain into quadrilateral subdomains. The method is tested on a Poisson problem, and exponential convergence in space is verified. We also solve a Stefan problem with a full ALE formulation, previously solved on a single element, now with a multi-element approach.

This is done to verify that the implementation of the multi-element approach is correct for such problems in time-dependent domains. The correctness of the method is verified qualitatively by comparing the solutions.

We then turn to fluid dynamics and the Navier-Stokes equations. They are explored by studying various simplifications, each adding a new level of complexity to the problem. For each problem, new issues that arise, such as potential solvability or uniqueness issues, spatial and temporal discretization and solution of the resulting algebraic system of equations, are discussed, and numerical examples are presented. For the simplest problem, involving only the viscous operator, exponential convergence in space is verified. Then the steady Stokes problem is considered, and we introduce the Uzawa algorithm to solve the algebraic system. Here, a preconditioning technique is studied, which makes the number of Conjugate Gradient iterations independent of the problem size. Unsteady Stokes problems are also studied, and here we solve a model problem to second order convergence in time and exponential convergence in space.

Finally, free surface conditions for fluid dynamic problems are introduced. These conditions yield time-dependent geometries and necessitate an ALE formulation. This formulation is derived. A droplet with constant surface tension is modeled both as a Stokes problem and a Navier-Stokes problem, and the transition towards a steady state is studied. For the Navier-Stokes model an oscillating motion is observed, and the effect of the non-dimensional Capillary number on the oscillations is studied. An example of a droplet with non-constant surface tension is also studied, and creation of counter-rotating vortices in the steady state is verified numerically.

An important problem with time-dependent domains is the distribution of the computational points when the domain is deformed. If the mapping from the reference domain to the new, deformed domain is not sufficiently regular, the polynomial approximation will become less accurate, and over time, the solution may even collapse. This is studied in this paper, but methods presented are ad hoc, related to specific geometries. This is indeed still an unsolved problem, and further work is needed. No general approach for preserving the regularity of the mapping from the reference domain to the physical domain has been found. Also, a quantitative study of how convergence depends on the regularity of this mapping would be useful.

It would also be helpful to have more model problems with exact solutions, so that the accuracy in the numerical solution on a deformed domain could be studied quantitatively in more detail, and for different problems. For the Stefan problem we have one numerical example in this paper where the exact solution is known, and convergence properties are verified. For the free surface Navier-Stokes problem we have none. The lack of exact solutions is a relevant issue for both the Stefan problem and the fluid dynamics problems.

Bibliography

[1] R. Aris *Vectors, Tensors and the Basic Equations of Fluid Mechanics*, Dover (1989)

[2] K. Arrow, L. Hurwicz, H. Uzawa *Studies in Nonlinear Programming*, Stanford University Press, Standford, CA (1958)

[3] C. Bernardi, Y. Maday *Spectral Methods* in: P.G. Ciarlet, J.L. Lions (eds.) *Handbook of Numerical Analysis. Volume V: Techniques of Scientific Computing*, 209–485, Elsevier (1997)

[4] C. Bernardi, Y. Maday *Uniform inf-sup conditions for the spectral discretization of the Stokes problem*, Math. Methods Appl. Sci., 9:395–414 (1999)

[5] N. Bodard, R. Bouffanais, M.O. Deville *Solution of moving-boundary problems by the spectral element method*, Applied Numerical Mathematics, 58(7):968–984 (2008)

[6] J.H. Bramble, J.E. Pasciak, A.T. Vassilev *Analysis of the inexact Uzawa algorithm for saddle point problems*, SIAM J. Numer. Anal. 34:1072–1092 (1997)

[7] R. Bouffanais, M.O. Deville *Mesh Update Techniques for Free-Surface Flow Solvers Using Spectral Element Method*, Journal of Scientific Computing, 27(1–3):137–149 (2006)

[8] R.L. Burdesn, J.D. Faires *Numerical Analysis*, 7th ed., Brooks/Cole (2001)

[9] C. Canuto, M.Y. Hussaini, A. Quarteroni, T.A. Zang *Spectral Methods in Fluid Dynamics*, Springer (1988)

[10] C. Canuto, M.Y. Hussaini, A. Quarteroni, T.A. Zang *Spectral Methods: Fundamentals in Single Domains*, Springer (2006)

[11] M.O. Deville, P.F. Fischer, E.H. Mund *High-Order Methods for Incompressible Fluid Flow*, Cambridge (2002)

[12] L. Formaggia, F. Nobile *A stability analysis for the arbitrary Lagrangian Eulerian formulation with finite elements*, East-West J. Numer. Math., 7(2):105–131 (1999)

[13] W.J. Gordon, C.A. Hall *Construction of curvilinear co-ordinate systems and applications to mesh generation*, International Journal for Numerical Methods in Engineering, 7:461–477 (1973)

[14] L. Ho, T. Patera *A Legendre spectral element method for simulation of unsteady incompressible viscous free-surface flows*, Comp. Methods Appl. Mech. Eng. 80(11):355–366, (1990)

[15] A. Huerta, A. Rodríguez-Ferran (eds.) *The Arbitrary Lagrangian-Eulerian Formulation*, Comp. Methods Appl. Mech. Eng. 193(39–41):4073–4456, (2004)

[16] A. Iserles *A First Course in the Numerical Analysis of Differential Equations*, Cambridge (1996)

[17] A. Johnson and T. Tezduyar *Mesh update strategies in parallel finite element computation of flow problems with moving boundaries and interfaces*, Comput. Methods Appl. Mech. Eng., 119:73–94 (1994)

[18] C. Johnson *Numerical solution of partial differential equations by the finite element method*, Studentlitteratur (1987)

[19] L.D. Landau, E.M. Lifshitz *Fluid Mechanics*, Course of Theoretical Physics, Volume 6, Butterworth-Heinemann (1987)

[20] D.R. Lynch *Unified approach to simulation on deforming elements with application to phase change problems*, Journal of Computational Physics, 47(3):387–411 (1982)

[21] R.E. Lynch, J.R. Rice, D.H. Thomas *Direct solution of partial differential equations by tensor product methods*, Numer. Math., 6:185–199 (1964)

[22] Y. Maday, D. Meiron, A.T. Patera, E.M. Rønquist *Analysis of Iterative Methods for the Steady and Unsteady Stokes Problem: Application to Spectral Element Discretizations* SIAM Journal on Scientific Computating, 14(2):310–337 (1993)

[23] Y. Maday, A.T. Patera *Spectral element methods for the Navier-Stokes equations* in: A.K. Noor and J.T. Odden (eds.) *State-of-the-Art Surveys in Computational Mechanics*, ASME, New York, 71–143 (1989)

[24] Y. Maday, A.T. Patera, E.M. Rønquist *An Operator-Integration-Factor Splitting Method for Time-Dependent Problems: Application to Incompressible Fluid Flow*, Journal of Scientific Computing, 5(4):263–292 (1990)

[25] Y. Maday, A.T. Patera, E.M. Rønquist *The* $\mathbb{P}_N \times \mathbb{P}_{N-2}$ *method for the approximation of the Stokes problem*, Technical Report 92009, Department of Mechanical Engineering, MIT, Cambridge, MA (1992)

[26] S.A. Orszag *Spectral methods for problems in complex geometry*, Journal of Computational Physics, 37:70–92 (1980)

[27] A.T. Patera *A spectral element method for fluid dynamics*, Journal of Computational Physics, 54:468–488 (1984)

[28] R. Peyret *Spectral Methods for Incompressible Viscous Flow*, Springer (2002)

[29] B. Ramaswamy, M. Kawahara *Arbitrary Lagrangian-Eulerian finite element method for unsteady convective incompressible viscous free surface flow*, International Journal for Numerical Methods in Fluids, 7(10):1053–1074 (1987)

[30] Y. Saad *Iterative Methods for Sparse Linear Systems*, SIAM (2003)

[31] G. Strang *Introduction to Applied Mathematics*, Wellesley-Cambridge Press (1986)

[32] F.M. White *Fluid Mechanics*, 6th ed., McGraw-Hill (2008)

[33] Z. Xu, M. Accorsi *Finite element mesh update methods for fluid-structure interaction simulations*, Finite Elements in Analysis and Design, 40(9–10):1259–1269 (2004)