Cluster of Excellence 128
"Integrative Production Technology for
High-Wage Countries"

- Research Area A2 -
New Technology Frontiers for Individualized Production

# Simulating Die Swell in the Context of Profile Extrusion

**Diploma Thesis**

By

Lutz Pauli

Supervisors: Dipl.-Ing. Stefanie Elgeti
Prof. M. Behr, Ph.D.

Aachen, May 18, 2011

CHAIR FOR
COMPUTATIONAL
ANALYSIS OF
TECHNICAL
SYSTEMS

RWTH AACHEN
UNIVERSITY

## Affidavit:

I hereby declare that I wrote this thesis on my own and without the use of any other than the cited sources and tools and all explanations that I copied directly or in their sense are marked as such, as well as that the thesis has not been available to any audit authority yet.

## Eidesstattliche Erklärung:

Ich versichere, dass ich die Diplomarbeit selbstständig und ohne Benutzung anderer als der angegebenen Quellen und Hilfsmittel angefertigt habe und alle Ausführungen, die wörtlich oder sinngemäß übernommen wurden, als solche gekennzeichnet sind, sowie dass die Diplomarbeit in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegt wurde.

Aachen, May 18, 2011 Lutz Pauli

# Abstract

Plastics extrusion is a manufacturing process suitable for continuous profiles such as floor skirtings, window frames, and pipes. In industrial applications, extrusion is a large-scale process, financially attractive only for mass products. This is due to the time consuming and costly regulation of the extrusion process and especially, the design of an extrusion die which relies on running-in experiments. The effect of changes in the die geometry on the flow through the die is completely non-intuitive since the flow of plastics melt behaves very nonlinear, e.g. due to viscoelastic effects. This motivates the development of numerical tools capable of simulating the flow through and behind extrusion dies in order to reduce the design costs.

The most important design criteria for an extrusion die are a homogeneous velocity distribution at the outflow and homogeneous die swell. Especially, an accurate numerical representation of die swell profiles turns out to be a challenging task. On the one hand, viscoelastic models have to be considered to be able to predict the amount of die swell realistically. On the other hand, the free surface has to be represented. In this particular application interface tracking presents many advantages. However, the standard mesh deformation of a free surface fails if sharp edges, like corners, are part of the die geometry. This is due to inaccurate normal and mesh displacement computations.

The aim of this study is the implementation of a numerical method which is able to predict die swell. As a basis, two viscoelastic models, the Oldroyd-B and the Giesekus model, are implemented in a Galerkin/Least-Squares finite element formulation with space-time discretization. Both models are tested in the context of two- and three-dimensional die swell simulations. In addition, a NURBS representation of free surfaces is presented to robustly compute three-dimensional free-surface flows.

# Contents

# List of Figures

# List of Tables

# Notation

### Latin Symbols

| | |
|---|---|
| $\mathbf{D}$ | source term of the constitutive equation |
| $\mathbf{g}$ | Dirichlet boundary condition for the velocity |
| $\mathbf{G}$ | Dirichlet boundary condition for the viscoelastic stress |
| $G$ | relaxation modulus |
| $\mathbf{h}$ | Neumann boundary condition |
| $H^1$ | First order Sobolev space |
| $\mathbf{I}$ | Identity tensor |
| $\mathbf{K}$ | Stiffness matrix |
| $\mathbf{N}$ | Nonlinear operator |
| $N_a, N_b$ | finite element shape functions |
| $Re$ | Reynolds number |
| $\mathbf{S}$ | viscoelastic stress tensor weighting function |
| $\mathbf{T}$ | viscoelastic stress tensor |
| $We$ | Weissenberg number |
| $b_i$ | NURBS basis function |
| $b_i'$ | NURBS basis function derivative |
| $\boldsymbol{c}$ | NURBS curve |
| $\mathbf{f}$ | external force vector |
| $h^e$ | element length |
| $p$ | pressure |
| $\boldsymbol{p}$ | NURBS control point |
| $q$ | pressure weighting function |
| $\mathbf{r}$ | residual vector |
| $\boldsymbol{c}$ | NURBS surface |
| $t$ | time |
| $\boldsymbol{t}$ | NURBS knot vector |
| $\mathbf{n}$, $\boldsymbol{n}$ | normal vector |
| $\mathbf{u}$ | velocity vector |
| $\mathbf{v}$ | nodal displacements |
| $w_{i,j}$ | NURBS weight |

## Greek Symbols

| | |
|---|---|
| $\Gamma$ | boundary of computational domain |
| $\Omega$ | computational domain |
| $\boldsymbol{\varepsilon}$ | rate of strain tensor |
| $\boldsymbol{\sigma}$ | total stress tensor |
| $\dot{\gamma}$ | shear rate |
| $\lambda_{\mathrm{mesh}}, \mu_{\mathrm{mesh}}$ | Lamé parameter |
| $\tau,\ \eta$ | NURBS parameter |
| $\tau_{\mathrm{MOM}},\ \tau_{\mathrm{CONT}},\ \tau_{\mathrm{CONS}}$ | GLS stabilization parameters |
| $\lambda$ | relaxation time |
| $\alpha$ | Giesekus mobility factor |
| $\rho$ | density |
| $\mu$ | viscosity |
| $\mu_s$ | solvent viscosity |
| $\mu_p$ | polymer viscosity |

## Calligraphic Symbols

| | |
|---|---|
| $\mathcal{L}$ | differential operator |
| $\mathcal{V}$ | test function space |
| $\mathcal{S}$ | trial function space |

## Superscripts

| | |
|---|---|
| h | discretized variable |

## Subscripts

| | |
|---|---|
| e | element |
| sd | space dimensions |
| tc | tensor components |

## Abbreviations

| | |
|---|---|
| ALE | Arbitrary Lagrangian-Eulerian formulation |
| BC | Boundary Condition |
| CAD | Computer Aided Design |
| DOF | Degrees Of Freedom |
| DSD/SST | Deforming Spatial Domain Stabilized Space-Time Method |
| EMUM | Elastic Mesh Update Method |
| FEM | Finite Element Method |
| GLS | Galerkin/Least-Squares |
| NURBS | Non-Uniform Rational B-Splines |
| UCM | Upper-Convected Maxwell Model |
| XNS | Flow solver |

# 1. Introduction

Plastics engineering involves many different types of processing. Five prominent examples are blow molding, injection molding, thermoforming, foil extrusion, and profile extrusion. This work is focused on profile extrusion which will be further described in the following.

Extrusion is suitable for continuous profiles such as floor skirtings, window frames, and pipes. The machinery is called extrusion line and schematically shown in Fig. 1.1. The single sections of the extrusion line listed in extrusion direction are the extruder, the extrusion die, the calibration, the cooling section, the puller, and the packaging unit. The entire process can be summarized in three main steps: melting, reshaping and cooling of the polymer. Raw plastic material enters the extruder, where it is melted and transported by a large screw. The molten material is subsequently pushed through the extrusion die. In the calibration and cooling section the melt is cooled down to fix its shape.

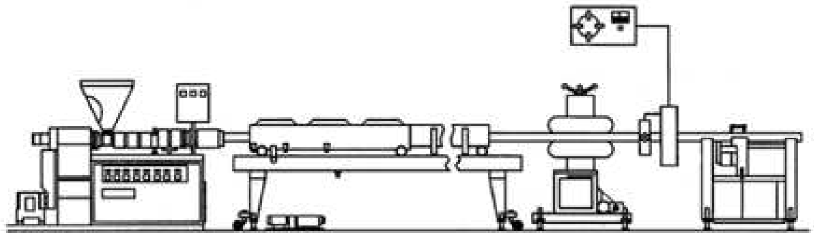


Figure 1.1.: Arrangement of a standard profile extrusion line. From left to right: the extruder, the extrusion die, the calibration, the cooling track, the puller and the packaging unit [Kunz et al. (2004)].

The extrusion die turns out to be a critical part in the entire process. The quality of the final product is highly dependent on the homogeneity of the material distribution of the outflow of the extrusion die. If the extrusion die does not ensure this homogeneity, the extrusion process is strongly disturbed. A further complication arises through the phenomenon of die swell. Liquids swell when they exit a shape-giving structure such as an extrusion die. This is due to the shifting from a fully-developed to a block velocity profile. In case of polymeric liquids this swelling is increased in strength significantly as compared to Newtonian liquids by viscoelastic stresses inside of the flow field. These stresses depend on the reshaping process in the interior of the die. Therefore, even the swelling, which has to be taken into consideration for the final height of the product, can become inhomogeneous, when determining the thickness of the outflow cross-section with respect to the thickness of the profile cross-section.

A common extrusion die can be subdivided into three zones: The inflow section, usually circular or ellipsoidal, a distribution section, where the melt is deformed, and a parallel zone (cf. Fig. 1.2). The distribution section should lead to a homogeneous and wake-free flow field, whereas the parallel zone is used to reconnect potential subdivided flow channels and also, to slightly lessen the swell intensity behind the die.

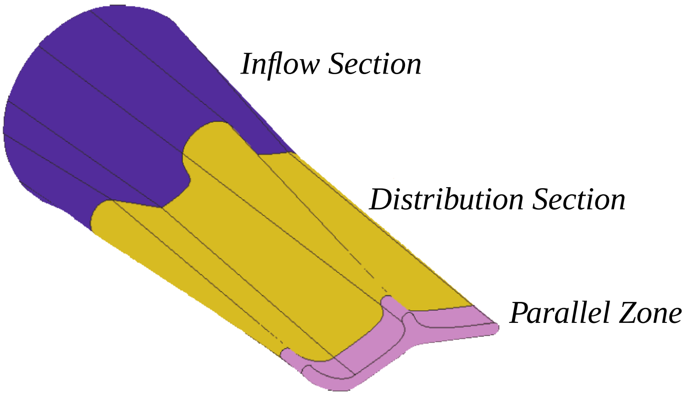


Figure 1.2.: Functional sections of an extrusion die.

The design of an extrusion die that fulfills the condition of homogeneity for the velocity as well as the swell profile is very unintuitive. Several iteratively manufactured dies have to be mounted onto the extruder to test the flow behavior – a costly and time consuming process. This motivates the development of numerical tools able to accurately compute the flow field inside as well as the swell behavior behind the die, to numerically optimize the die geometry before the manufacturing.

The aim of this study is to introduce a solver for three-dimensional viscoelastic flows into the existing numerical tools. In addition, a method is further developed to accurately compute free-surfaces with complex shapes like sharp edges.

## 1.1. Overview

Intention of Chapter 2 is to provide the necessary background about viscoelasticity and the modeling of viscoelastic free-surface flows. In Chapter 3 all the utilized methods are presented. Chapter 4 shows the numerical behavior of the implementation of the viscoelastic models. In Chapter 5 several die swell simulations are studied and evaluated. Finally, in Chapter 6 the new methods are summerized and discussed.

# 2. Background

## 2.1. Viscoelasticity

Polymeric liquids used in plastics extrusion show a flow behavior quite different to most other liquids. On the one hand they show elastic effects comparable to a Hookean solid and on the other hand viscous effects like a Newtonian liquid. This mixture of viscous and elastic effects, referred to as viscoelasticity, causes some interesting phenomena. The most important ones are summarized below:

1. Time dependence of the stresses (relaxation)
2. Shear-thinning viscosity
3. Normal stresses in shear
4. Extensional-thickening viscosity

The time dependence of viscoelastic stresses is illustrated in Fig. 2.1. It shows the stress response over time after a step increase in strain of a Hookean solid, a Newtonian liquid, and a viscoelastic liquid. On the one hand, the stress of a Hookean solid is proportional to the strain and thus, shows the same progression as the strain itself. On the other hand, the stress of a Newtonian liquid is proportional to the derivative of the strain and therefore, stresses only occur as long as the strain is changing. In contrast to the Newtonian liquid, the viscoelastic liquid is able to “remember” its original form and needs some time until it is fully relaxed. Here, the stress is dependent on time, generating a memory effect inside of the liquid.

Shear-thinning viscosity can be explained by imagining a glass bottle filled with tomato sauce, a popular viscoelastic substance. If the bottle is tilted the tomato sauce starts flowing slowly and only close to the wall, where the viscosity already slightly decreases due to the acting shear stress. However, if the bottle is given a shake a very high shear stress occurs, so that the viscosity decreases significantly and the tomato sauce flows very fast.

More dramatic non-Newtonian phenomena are normal stresses in shear flow. Fig. 2.2 shows two tanks, one filled with a Newtonian and the other one with a viscoelastic liquid. Both tanks are rotated around a fixed rod. The inertia forces cause the Newtonian liquid to move away from the rod. In contrast, the viscoelastic liquid starts climbing up the rod due to the developing normal stresses. One can think of the polymeric liquid as containing rubber bands, which when stretched pull back in the direction of the flow. This tension along the circular lines of flow generates a pressure towards the center, pushing the fluid up the rod. This phenomenon is called *Weissenberg effect*, named after Karl Weissenberg [Macosko (1994)].

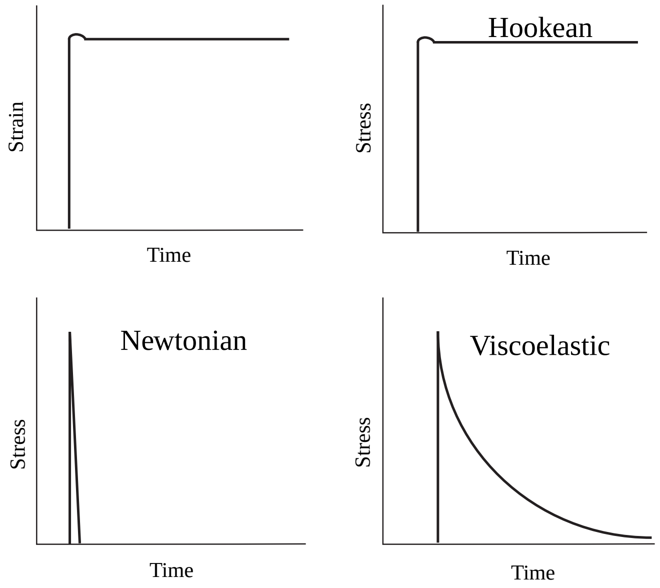


Figure 2.1.: Stress response of a Hookean solid, Newtonian liquid and viscoelastic liquid after a step increase in strain (reproduced from [Macosko (1994)]).

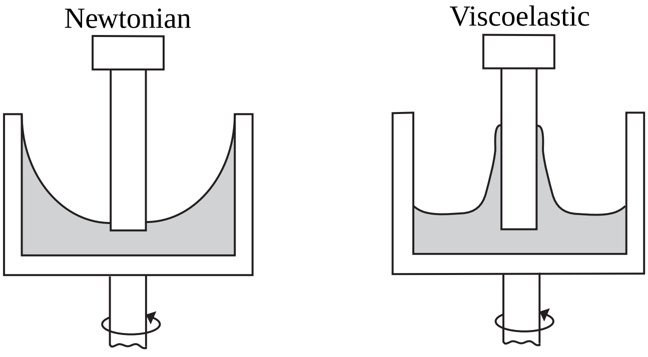


Figure 2.2.: Comparison of a Newtonian and a viscoelastic liquid in a tank rotated around a fixed rod. The viscoelastic liquid starts climbing up the rod due to normal stresses.

The last phenomenon is extensional thickening viscosity. The two tanks can again serve as an illustration. This time the rod is dipped into the liquid and slowly lifted up. The Newtonian liquid will break off very soon. However, some viscoelastic liquids can be pulled up many centimeters as a result of the increased viscosity during the elongation process. Polymeric liquids can show much higher viscosities in extension than in shear.

In plastics extrusion time-dependent, normal stresses cause the polymeric melt to swell stronger than Newtonian liquids after leaving the extrusion die. This die swell has to be considered in the design process of the die. Furthermore, if the die design is assisted by numerical simulations, viscoelastic effects have to be taken into account in the modeling as well.

## 2.2. Mathematical Modeling

### 2.2.1. Governing Equations

The flow of polymeric liquids in extrusion dies is characterized by low velocities, high viscosities and a constant density. In consequence the Reynolds number[1] is much smaller than one, i.e. $\mathrm{Re} << 1$. For such a flow it is sufficient to use the equations of Stokes flow, which assume $\mathrm{Re} \equiv 0$. Considering an incompressible, isothermal and unsteady liquid occupying at an instant $t \in [0, T]$ a bounded region $\Omega_t \subset \mathbb{R}^{n_\mathrm{sd}}$, where $n_\mathrm{sd}$ is the number of space dimensions, the mass and momentum equations of the Stokes flow are defined by

$$\begin{aligned} \rho \left( \frac{\partial \mathbf{u}}{\partial t} - \mathbf{f} \right) - \boldsymbol{\nabla} \cdot \boldsymbol{\sigma} = \mathbf{0} \quad &\text{on } \Omega_t \quad \forall t \in [0, T] \,, \\ \boldsymbol{\nabla} \cdot \mathbf{u} = 0 \quad &\text{on } \Omega_t \quad \forall t \in [0, T] \,, \end{aligned} \tag{2.1}$$

where $\mathbf{u}(\mathbf{x}, t)$ is the velocity vector, $\boldsymbol{\sigma}(\mathbf{x}, t)$ the total stress tensor and $\mathbf{f}(\mathbf{x}, t)$ the vector of external body forces. The closure is obtained with a constitutive equation relating the total stress tensor $\boldsymbol{\sigma}$ to velocity and pressure fields. Following the notation of [Donea and Huerta (2003)] vectors are bold lower-case letters and tensors are bold upper-case and bold Greek letters.

#### 2.2.1.1. Constitutive Equations

Material laws are specified by constitutive equations. In the field of fluid dynamics the simplest one is the Newtonian liquid. It states that the stress is proportional to the rate of strain, where the viscosity is the constant of proportionality. The constitutive equation reads

$$\boldsymbol{\sigma} = -p\mathbf{I} + 2\mu\boldsymbol{\varepsilon}(\mathbf{u}) \,, \tag{2.2}$$

[1] The Reynolds number is a characteristic, dimensionless flow number and defined by $\mathrm{Re} = \frac{\rho u l}{\mu}$, where $\rho$, $u$, $l$ and $\mu$ are characteristic density, velocity, length and viscosity, respectively.

where $p$ is the pressure and $\boldsymbol{\varepsilon}(\mathbf{u}) = \frac{1}{2}\left(\boldsymbol{\nabla}\mathbf{u} + (\boldsymbol{\nabla}\mathbf{u})^T\right)$ the rate of strain tensor.

In Sec. 2.1 it is already shown that the Newtonian constitutive model is not sufficient to describe the flow characteristics of polymeric liquids. An improved representation of the viscoelastic behavior can be obtained by using the Maxwell model, which is able to account for time dependent stress. It is composed of a damper and a spring element connected in a series (cf. Fig. 2.3). The differential form of the model in 1-D can be expressed by

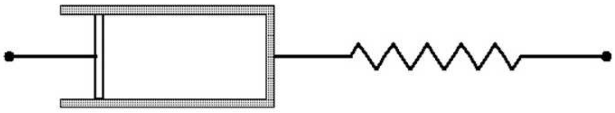


Figure 2.3.: Illustration of the Maxwell model, composed of a damper and a spring element in series connection.

$$T + \lambda \frac{dT}{dt} = \mu\varepsilon\,, \tag{2.3}$$

with $\lambda$ being the so-called relaxation time, $T$ the viscoelastic stress and $\varepsilon$ the strain rate. If the spring element of the Maxwell model has the stiffness $G$ than the viscosity of the damper element applies to $\mu = \lambda G$. The generalization to nonlinear viscoelasticity and higher dimensions is called upper-convected Maxwell model (UCM) and reads

$$\mathbf{T} + \lambda \overset{\triangledown}{\mathbf{T}} = 2\mu\boldsymbol{\varepsilon}(\mathbf{u})\,. \tag{2.4}$$

$\overset{\triangledown}{\mathbf{T}}$ is the upper-convected derivative of the viscoelastic stress tensor $\mathbf{T}$ and defined by

$$\overset{\triangledown}{\mathbf{T}} = \frac{\partial \mathbf{T}}{\partial t} + \mathbf{u}\cdot\boldsymbol{\nabla}\mathbf{T} - \left(\boldsymbol{\nabla}\mathbf{u}\cdot\mathbf{T} + \mathbf{T}\cdot(\boldsymbol{\nabla}\mathbf{u})^T\right)\,. \tag{2.5}$$

It can be thought of a stress rate which is rotation invariant in three dimensions. Therefore, stress is only produced by particle deformations, but not by pure rotations. The constitutive equation of the UCM model is obtained by adding the viscoelastic stress $\mathbf{T}$, which is computed from Eq. (2.4), to the total stress $\boldsymbol{\sigma}$, i.e.

$$\boldsymbol{\sigma} = -p\mathbf{I} + \mathbf{T}\,. \tag{2.6}$$

Therefore, the independent components of the stress tensor $\mathbf{T}$ become additional degrees of freedom[2] (DOF).

The UCM model is extremely difficult to handle numerically. One reason is the convective character of the stress evolution equation [Baranger and Sandri (1992)]. This difficulty is reduced when

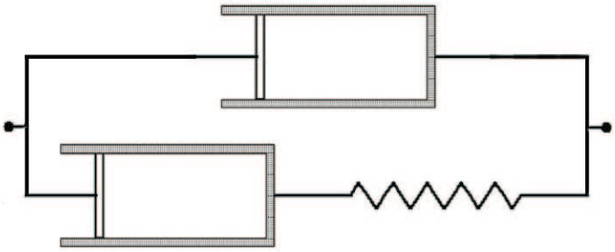


Figure 2.4.: The Oldroyd-B model, composed of a Maxwell and a damper element in a parallel connection.

the model is completed with an additional viscous part. Illustratively, the Maxwell element is complemented by a second damper which is connected in parallel (cf. Fig. 2.4). The resulting model is called Oldroyd-B model.

Combination of Eqs. (2.4), (2.5), and (2.6) leads to the following constitutive equations:

$$\boldsymbol{\sigma} = -p\mathbf{I} + 2\mu_s\boldsymbol{\varepsilon}(\mathbf{u}) + \mathbf{T}\,, \tag{2.7}$$

$$\mathbf{T} + \lambda\overset{\triangledown}{\mathbf{T}} = 2\mu_p\boldsymbol{\varepsilon}(\mathbf{u})\,, \tag{2.8}$$

$$\mu = \mu_s + \mu_p\,. \tag{2.9}$$

The sum of solvent viscosity $\mu_s$ and polymer viscosity $\mu_p$ results in the total viscosity $\mu$. The Oldroyd-B model is able to represent stress relaxation as well as the development of normal stresses in flow direction under constant shear. However, for normal stresses perpendicular to flow direction, shear-thinning and extensional thickening are not predicted.

Several improvements of the UCM/Oldryod-B model exist in the literature [Macosko (1994)]. The one used within this thesis is the Giesekus model. The Giesekus model is obtained by adding a quadratic stress term to Eq. (2.8) with a new parameter $\alpha$, i.e.

$$\mathbf{T} + \lambda\overset{\triangledown}{\mathbf{T}} + \frac{\alpha\lambda}{\mu_p}\mathbf{T}\cdot\mathbf{T} = 2\mu_p\boldsymbol{\varepsilon}(\mathbf{u})\,. \tag{2.10}$$

The additional term is designed to track the interaction between a particular molecule and its surrounding molecules. $\alpha$ signifies the anisotropic mobility induced by flow characteristics. Therefore, it is named mobility factor and can take on values between 0 and 1. The Giesekus model can predict normal stresses in all directions and also shear-thinning viscosity. It is one of the best models for shearing flows and showed good results in die swell simulations [Fink (2008)]. Only extensional thickening cannot be modeled.

As already mentioned, there exist a lot of other models for the simulation of viscoelastic liquids. As examples the Phan-Thien and Tanner (PTT), the FENE-CR and the “Pom-Pom” models should be mentioned here. All three proved to be very useful [Keunings (2000)].

[2] Additional to the velocity components and the pressure

#### 2.2.1.2. Boundary Conditions

To obtain a mathematically closed system for Eqs. (2.1), (2.8) or (2.10) appropriate boundary and initial conditions have to be defined. The boundary of the computational domain $\Omega_t$ is decomposed into two disjoint parts, the Dirichlet and the Neumann boundary, denoted as $(\Gamma_t)_g$ and $(\Gamma_t)_h$ respectively. On the Dirichlet boundary a value for all degrees of freedom, except for the pressure, is directly prescribed:

$$\mathbf{u}(\mathbf{x}, t) = \mathbf{g} \quad \text{on } (\Gamma_t)_{\mathbf{g}} \,, \tag{2.11}$$

$$\mathbf{T}(\mathbf{x}, t) = \mathbf{G} \quad \text{on } (\Gamma_t)_{\mathbf{g}} \,. \tag{2.12}$$

On the Neumann boundary the total stress normal to the boundary is prescribed, i.e.

$$\mathbf{n} \cdot \boldsymbol{\sigma} = \mathbf{h} \quad \text{on } (\Gamma_t)_{\mathbf{h}} \,. \tag{2.13}$$

Note that there is no Neumann boundary for the viscoelastic stress as Eq. (2.8) or (2.10) has no second-order derivative inside. The divergence-free initial condition is given by:

$$\mathbf{u}(\mathbf{x}, 0) = \mathbf{u}_0 \quad \text{on } \Omega_0 \,, \tag{2.14}$$

$$\mathbf{T}(\mathbf{x}, 0) = \mathbf{T}_0 \quad \text{on } \Omega_0 \,. \tag{2.15}$$

### 2.2.2. Stabilized Space-Time Finite Element Methods

The finite element method (FEM) is a very potent technique to solve partial differential equations numerically. Within this thesis only a rough introduction to FEM is given. For more informations on FEM, refer to e.g. [Hughes (2000)] and [Donea and Huerta (2003)].

As a study example for stabilization and space-time formulations, the equations of Stokes flow are used together with the Newtonian constitutive equation and its corresponding boundary and initial conditions without stresses as additional DOF (cf. Eqs. (2.1), (2.2), (2.11), (2.13) and (2.14) respectively). In a first step, the steady case, i.e. $\frac{\partial \mathbf{u}}{\partial t} = \mathbf{0}$ and $\Omega_t = \Omega$, is assumed. Collecting all the necessary equations gives:

$$\boldsymbol{\nabla} \cdot (p\mathbf{I} - 2\mu\boldsymbol{\varepsilon}(\mathbf{u})) = \rho\mathbf{f} \quad \text{on } \Omega \,, \tag{2.16}$$

$$\boldsymbol{\nabla} \cdot \mathbf{u} = 0 \quad \text{on } \Omega \,, \tag{2.17}$$

$$\begin{aligned} \mathbf{u} &= \mathbf{g} \quad \text{on } \Gamma_{\mathbf{g}} \,, \\ \mathbf{n} \cdot \boldsymbol{\sigma} &= \mathbf{h} \quad \text{on } \Gamma_{\mathbf{h}} \,. \end{aligned} \tag{2.18}$$

In the finite element context Eqs. (2.16) and (2.17) are called strong or differential form of the problem.

### 2.2.2.1. Finite Element Discretization

To obtain the weak form, the momentum equation (2.16) is multiplied by the velocity weighting function $\mathbf{w}$ and integrated by parts, thereby generating the natural boundary condition on $\Gamma_h$. Similarly, the continuity equation (2.17) is multiplied by the pressure test function $q$. The result is integrated over the computational domain $\Omega$. Afterwards, this integral equation is discretized using the Galerkin formulation. Therefore, local approximations for both velocity $\mathbf{u}^h$ and pressure $p^h$, as well as their associated weighting functions $\mathbf{w}^h$ and $q^h$ are introduced. $\mathbf{u}^h$ and $p^h$, called trial functions, have to fulfill the Dirichlet boundary conditions and their derivatives have to be square-integrable. Since first order finite elements are used within this work, only the zeroth and first order derivatives of the trial functions have to be square-integrable. A first order Sobolev space, i.e. $H^{1h}$, meets this condition. The weighting functions also have to be in $H^{1h}$, but they have to vanish on the Dirichlet boundary. Summing up, one gets:

$$\begin{aligned}
&\tilde{\mathcal{S}}_{\mathbf{u}}^h = \{\, \mathbf{u}^h \,|\, \mathbf{u}^h \in [H^{1h}(\Omega)]^{n_{\text{sd}}}, \mathbf{u}^h \doteq \mathbf{g}^h \quad \text{on} \quad \Gamma_{\mathbf{g}}^h \,\}\,,\\
&\tilde{\mathcal{V}}_{\mathbf{u}}^h = \{\, \mathbf{w}^h \,|\, \mathbf{w}^h \in [H^{1h}(\Omega)]^{n_{\text{sd}}}, \mathbf{w}^h \doteq \mathbf{0} \quad \text{on} \quad \Gamma_{\mathbf{g}}^h \,\}\,,\\
&\tilde{\mathcal{S}}_p^h = \tilde{\mathcal{V}}_p^h = \{\, p^h \,|\, p^h \in H^{1h}(\Omega) \,\}\,.
\end{aligned} \tag{2.19}$$

With the aid of these function spaces the Galerkin form of the Stokes problem becomes: Given $\mathbf{f}$, $\mathbf{g}$ and $\mathbf{h}$, find $\mathbf{u}^h \in \tilde{\mathcal{S}}_{\mathbf{u}}^h$ and $p^h \in \tilde{\mathcal{S}}_p^h$ such that $\forall \mathbf{w}^h \in \tilde{\mathcal{V}}_{\mathbf{u}}^h$, $\forall q^h \in \tilde{\mathcal{V}}_p^h$:

$$\begin{aligned}
2\mu \int_{\Omega^h} \boldsymbol{\varepsilon}(\mathbf{w}^h) : \boldsymbol{\varepsilon}(\mathbf{u}^h)\, d\Omega - \int_{\Omega^h} \boldsymbol{\nabla} \cdot \mathbf{w}^h p^h \, d\Omega + \int_{\Omega^h} q^h \boldsymbol{\nabla} \cdot \mathbf{u}^h \, d\Omega&\\
= \int_{\Omega^h} \mathbf{w}^h \cdot \rho \mathbf{f} \, d\Omega + \int_{\Gamma_{\mathbf{h}}^h} \mathbf{w}^h \cdot \mathbf{h}^h d\Gamma\,.&
\end{aligned} \tag{2.20}$$

### 2.2.2.2. Stabilization

Unfortunately, Eq. (2.20) does not lead to stable solutions for the chosen first order finite element spaces, because the important LBB compatibility condition is not satisfied [Donea and Huerta (2003)]. One method to circumvent this condition while still using finite elements of first order is Galerkin/Least-Squares stabilization (GLS). Here, an element-by-element weighted least-squares residual of the original differential equation is added to the Galerkin part. With the definition $\mathcal{L}(\mathbf{u}^h) = \boldsymbol{\nabla} p^h - \boldsymbol{\nabla}(2\mu\boldsymbol{\varepsilon}(\mathbf{u}^h))$ the GLS formulation of the Stokes problem (2.16) and (2.17) results in: Given $\mathbf{f}$, $\mathbf{g}$ and $\mathbf{h}$, find $\mathbf{u}^h \in \tilde{\mathcal{S}}_{\mathbf{u}}^h$ and $p^h \in \tilde{\mathcal{S}}_p^h$ such that $\forall \mathbf{w}^h \in \tilde{\mathcal{V}}_{\mathbf{u}}^h$, $\forall q^h \in \tilde{\mathcal{V}}_p^h$:

$$\begin{aligned}
2\mu \int_{\Omega^h} \boldsymbol{\varepsilon}(\mathbf{w}^h) : \boldsymbol{\varepsilon}(\mathbf{u}^h)\, d\Omega - \int_{\Omega^h} \boldsymbol{\nabla} \cdot \mathbf{w}^h p^h \, d\Omega + \int_{\Omega^h} q^h \boldsymbol{\nabla} \cdot \mathbf{u}^h \, d\Omega&\\
+ \sum_{e=1}^{n_{\text{el}}} \int_{\Omega^e} \mathcal{L}(\mathbf{w}^h) \tau (\mathcal{L}(\mathbf{u}^h) - \rho \mathbf{f}) d\Omega&\\
= \int_{\Omega^h} \mathbf{w}^h \cdot \rho \mathbf{f} \, d\Omega + \int_{\Gamma_{\mathbf{h}}^h} \mathbf{w}^h \cdot \mathbf{h}^h d\Gamma\,.&
\end{aligned} \tag{2.21}$$

The stabilization parameter $\tau$ is often defined by [Shakib (1988)]:

$$\tau \approx \frac{1}{\|\mathcal{L}^*\|}\,, \tag{2.22}$$

where $\mathcal{L}^*$ is a linearization of $\mathcal{L}(\mathbf{u}^h)$, e.g. $\mathcal{L}(\mathbf{u}^h) \approx \mathcal{L}^*\mathbf{u}^h$.

#### 2.2.2.3. Space-Time Formulation

So far, only spatial discretization is considered in the model. For unsteady Stokes flow (Eq. (2.1)) also the time needs to be discretized. The space-time formulation uses finite elements for the space as well as for the time discretization.

First, the time interval $[0, T]$ is divided into subintervals $I_n = (t_n, t_{n+1})$, with $t_n$ and $t_{n+1}$ representing an ordered series of time levels $0 = t_0 < t_1 < \cdots < t_N = T$. Setting $\Omega_n = \Omega_{t_n}$ and $\Gamma_n = \Gamma_{t_n}$, the space-time slab $Q_n$ is defined as the domain enclosed by the surfaces $\Omega_n, \Omega_{n+1}$ as well as the surface described by $\Gamma_t$ as $t$ traverses $I_n$, which shall be named $P_n$ (cf. Fig. 2.5). Considering this space-time approach the function spaces in Eq. (2.19) are extended to

$$\begin{aligned}
(\mathcal{S}^h_{\mathbf{u}})_n &= \{\, \mathbf{u}^h \,|\, \mathbf{u}^h \in [H^{1h}(Q_n)]^{n_{\text{sd}}}, \mathbf{u}^h \doteq \mathbf{g}^h \quad \text{on} \quad (P_n)_{\mathbf{g}} \,\}\,,\\
(\mathcal{V}^h_{\mathbf{u}})_n &= \{\, \mathbf{w}^h \,|\, \mathbf{w}^h \in [H^{1h}(Q_n)]^{n_{\text{sd}}}, \mathbf{w}^h \doteq \mathbf{0} \quad \text{on} \quad (P_n)_{\mathbf{g}} \,\}\,,\\
(\mathcal{S}^h_p)_n &= (\mathcal{V}^h_p)_n = \{\, p^h \,|\, p^h \in H^{1h}(Q_n) \,\}\,,
\end{aligned} \tag{2.23}$$

for each space-time slab. Furthermore, the extension of the steady Stokes problem in GLS formulation (2.21) to the unsteady space-time case becomes: Given $(\mathbf{u}^h)^-_n$, $\mathbf{f}$, $\mathbf{g}$ and $\mathbf{h}$, find $\mathbf{u}^h \in (\mathcal{S}^h_{\mathbf{u}})_n$ and

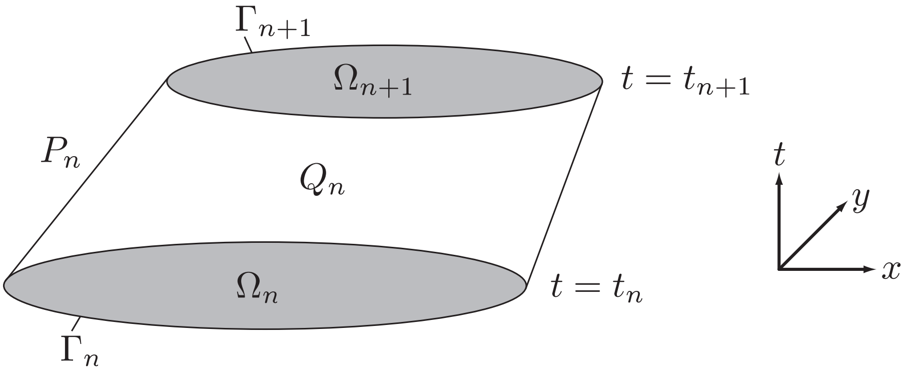


Figure 2.5.: Sketch of a space-time slab (from [Sauerland (2008)]).

$p^h \in (\mathcal{S}_p^h)_n$ such that $\forall \mathbf{w}^h \in (\mathcal{V}_{\mathbf{u}}^h)_n$, $\forall q^h \in (\mathcal{V}_p^h)_n$:

$$\begin{aligned}\int_{Q_n} \mathbf{w}^h \cdot \rho \left( \frac{\partial \mathbf{u}^h}{\partial t} - \mathbf{f} \right) \, dQ - \int_{Q_n} \boldsymbol{\nabla} \cdot \mathbf{w}^h p^h \, dQ \\ +2\mu \int_{Q_n} \boldsymbol{\varepsilon}(\mathbf{w}^h) : \boldsymbol{\varepsilon}(\mathbf{u}^h) \, dQ + \int_{Q_n} q^h \boldsymbol{\nabla} \cdot \mathbf{u}^h \, dQ \\ + \int_{\Omega_n} (\mathbf{w}^h)_n^+ \cdot \rho \left( (\mathbf{u}^h)_n^+ - (\mathbf{u}^h)_n^- \right) \, d\Omega \\ + \sum_{e=1}^{n_{\text{el}}} \int_{Q_n^e} \mathcal{L}(\mathbf{w}^h) \tau (\mathcal{L}(\mathbf{u}^h) - \rho \mathbf{f}) \, dQ \\ = \int_{(P_n)_{\mathbf{h}}} \mathbf{w}^h \cdot \mathbf{h}^h \, dP \,,\end{aligned} \tag{2.24}$$

with $\mathcal{L}(\mathbf{u}^h) = \rho \frac{\partial \mathbf{u}^h}{\partial t} + \boldsymbol{\nabla} p^h - \boldsymbol{\nabla}(2\mu\boldsymbol{\varepsilon}(\mathbf{u}^h))$. In the above equation, the following notation is used:

$$(\mathbf{u}^h)_n^{\pm} = \lim_{\epsilon \to 0} \mathbf{u}(t_n \pm \epsilon) \,, \tag{2.25}$$

$$\int_{Q_n} \ldots dQ = \int_{I_n} \int_{\Omega_t^h} \ldots d\Omega dt \,, \tag{2.26}$$

$$\int_{P_n} \ldots dP = \int_{I_n} \int_{\Gamma_t^h} \ldots d\Gamma dt \,. \tag{2.27}$$

The problem is solved sequentially for each space-time slab, starting with the initial condition:

$$(\mathbf{u}^h)_0^+ = \mathbf{u}_0 \,. \tag{2.28}$$

### 2.2.3. Free-Surface Flows

To predict die swell in an extrusion process, free-surface flows have to be computed. In such simulations the position of the free surface becomes an additional degree of freedom in the equation system. Within this work interface-tracking is used to handle the movable domain. This means that the computational nodes are always fixed at the moving surface and the computational mesh is adjusted to the movement of these nodes.

When dealing with free-surface flows the space-time formulation has advantages because it automatically accounts for the deformation of the computational domain. The domain is allowed to deform within each single space-time slab. Such finite element formulations are called Deformable-Spatial-Domain/Stabilized Space-Time (DSD/SST) [Tezduyar et al. (1992a)] and [Tezduyar et al. (1992b)]. The DSD/SST method makes use of the Arbitrary Lagrangian-Eulerian description, described in the following section.

#### 2.2.3.1. Arbitrary Lagrangian-Eulerian (ALE) Description

The ALE method combines the advantages of both, the Eulerian description which is especially suited for flows with e.g. complex streamlines and the Lagrangian description which is able to track moving interfaces. In the ALE approach the mesh velocities explicitly enter the momentum equation. Therefore, the momentum equation is written on a reference domain. This reference domain can be either transformed to the spatial or the material domain. Thus, three different domains have to be specified. Despite this complexity the ALE approach has proved to be a good description for free-surface flows—see [Hughes et al. (1981); Huerta and Liu (1988)].

#### 2.2.3.2. Elastic Mesh Update Method (EMUM)

In interface-tracking techniques, the set of computational nodes on the free surface remains associated to that surface throughout the entire computation. If the volume mesh is inflexible the elements connected to that surface will be subject to large deformations. Mesh failure and numerical inaccuracies are the consequences. To solve this issue the volume mesh is treated as a fictitious elastic body reacting to the deformation applied to it. Therefore, an elastic model is solved for the node displacement $\mathbf{v}$:

$$\boldsymbol{\sigma}_{\mathrm{mesh}}(\mathbf{v}) = \lambda_{\mathrm{mesh}}\left(\mathrm{tr}\boldsymbol{\varepsilon}_{\mathrm{mesh}}(\mathbf{v})\right)\mathbf{I} + 2\mu_{\mathrm{mesh}}\varepsilon_{\mathrm{mesh}}(\mathbf{v}), \tag{2.29}$$

$$\boldsymbol{\varepsilon}_{\mathrm{mesh}}(\mathbf{v}) = \frac{1}{2}\left(\boldsymbol{\nabla}\mathbf{v} + (\boldsymbol{\nabla}\mathbf{v})^T\right)\,. \tag{2.30}$$

$\lambda_{\mathrm{mesh}}$ and $\mu_{\mathrm{mesh}}$ are the Lamé constants, defining the stiffness of the isotropic, hyperelastic material [Johnson and Tezduyar (1994)]. Each element has different Lamé constants such that larger elements experience also larger deformations. Eq. (2.29) is solved with the classical Galerkin finite element method.

# 3. Methods

This chapter describes the development and implementation of the utilized methods to simulate die swell within this study. All the methods are implemented into the flow solver XNS.

## 3.1. Short Introduction to XNS

XNS is a program to solve flow problems and based on the command line. XNS is able to solve, e.g., advection-diffusion, Stokes and Navier-Stokes equations in two and three dimensions using Galerkin/Least-Squares stabilized finite-element methods with both space-time and semi-discrete time discretization (cf. Sec. 2.2.2 for the former one). It can utilize various parallel machines (Xeon clusters, IBM Blue Gene, etc.) and is able to exploit the common communication interfaces for distributed memory systems (SHMEM and MPI). To set-up simulations an input file (*xns.in*) has to be written, where all the simulation parameters are defined, e.g., mesh, model, material, solver and boundary parameters. The mesh itself is stored in the *MIXD*-format, described in [Behr (2010)].

## 3.2. Stabilized Space-Time Finite Element Formulation of Giesekus and Oldroyd-B Liquids

In [Behr (1992)] a GLS formulation for the steady Oldroyd-B liquid is presented and further studied in, e.g., [Behr et al. (2004); Behr et al. (2005)]. The method was implemented in two space dimensions. In the current work the viscoelastic section of the flow solver is extended by the following functionality:

1. A space-time formulation of the Oldroyd-B model is added to make the method capable for unsteady computations.

2. The two-dimensional implementation of the Oldroyd-B model is extended to three space dimensions.

3. The Giesekus model is implemented for the 2-D as well as the 3-D case.

The equations of Stokes flow from Eq. (2.1) and the constitutive equations (2.7), (2.8) or (2.10) are considered here. The $n_{\mathrm{tc}} = n_{\mathrm{sd}}(n_{\mathrm{sd}} + 1)/2$ independent tensor components of the viscoelastic stress tensor $\mathbf{T}$ are necessarily treated as additional unknowns. Therefore, the finite element function spaces presented in Eq. (2.23) are extended by the appropriate stress spaces, i.e.

$$\begin{aligned}(\mathcal{S}^h_{\mathbf{T}})_n &= \{\, \mathbf{T}^h \,|\, \mathbf{T}^h \in [H^{1h}(Q_n)]^{n_{\mathrm{tc}}}, \mathbf{T}^h \doteq \mathbf{G}^h \quad \text{on} \quad (P_n)_{\mathbf{g}} \,\}\,, \\ (\mathcal{V}^h_{\mathbf{T}})_n &= \{\, \mathbf{S}^h \,|\, \mathbf{S}^h \in [H^{1h}(Q_n)]^{n_{\mathrm{tc}}}, \mathbf{S}^h \doteq \mathbf{0} \quad \text{on} \quad (P_n)_{\mathbf{g}} \,\}\,. \end{aligned} \tag{3.1}$$

Following the explanations and notations from Sec. 2.2.2 and 2.2.3, the DSD/SST finite element formulation of the Giesekus model becomes: Given $(\mathbf{u}^h)^-_n$ and $(\mathbf{T}^h)^-_n$ find $\mathbf{u}^h \in (\mathcal{S}^h_{\mathbf{u}})_n$ and $p^h \in (\mathcal{S}^h_p)_n$ and $\mathbf{T}^h \in (\mathcal{S}^h_{\mathbf{T}})_n$ such that $\forall \mathbf{w}^h \in (\mathcal{V}^h_{\mathbf{u}})_n$, $\forall q^h \in (\mathcal{V}^h_p)_n$, and $\forall \mathbf{S}^h \in (\mathcal{V}^h_{\mathbf{T}})_n$:

$$\begin{aligned}
&\int_{Q_n} \mathbf{w}^h \cdot \rho \left( \frac{\partial \mathbf{u}^h}{\partial t} - \mathbf{f} \right) dQ - \int_{Q_n} \boldsymbol{\nabla} \cdot \mathbf{w}^h p^h \, dQ + \int_{Q_n} \boldsymbol{\varepsilon}(\mathbf{w}^h) : \mathbf{T}^h \, dQ \\
&\quad + 2\mu_s \int_{Q_n} \boldsymbol{\varepsilon}(\mathbf{w}^h) : \boldsymbol{\varepsilon}(\mathbf{u}^h) \, dQ + \int_{Q_n} q^h \boldsymbol{\nabla} \cdot \mathbf{u}^h \, dQ \\
&\quad + \int_{\Omega_n} (\mathbf{w}^h)^+_n \cdot \rho \left( (\mathbf{u}^h)^+_n - (\mathbf{u}^h)^-_n \right) d\Omega + \int_{\Omega_n} (\mathbf{S}^h)^+_n \cdot \left( (\mathbf{T}^h)^+_n - (\mathbf{T}^h)^-_n \right) d\Omega \\
&\quad + \frac{1}{2\mu_p} \int_{Q_n} \mathbf{S}^h : \mathbf{T}^h \, dQ + \frac{\alpha\lambda}{2\mu_p^2} \int_{Q_n} \mathbf{S}^h : \mathbf{T}^h \cdot \mathbf{T}^h dQ + \frac{\lambda}{2\mu_p} \int_{Q_n} \mathbf{S}^h : \overset{\triangledown}{\mathbf{T}}{}^h dQ - \int_{Q_n} \mathbf{S}^h : \boldsymbol{\varepsilon}(\mathbf{u}^h) \, dQ \\
&\quad + \sum_{e=1}^{(n_{el})_n} \int_{Q^e_n} \tau_{\text{MOM}} \frac{1}{\rho} \left[ \rho \frac{\partial \mathbf{w}^h}{\partial t} + \boldsymbol{\nabla} q^h - \boldsymbol{\nabla} \cdot \mathbf{S}^h - \boldsymbol{\nabla}(2\mu_s \boldsymbol{\varepsilon}(\mathbf{w}^h)) \right] \\
&\qquad \cdot \left[ \rho \left( \frac{\partial \mathbf{u}^h}{\partial t} - \mathbf{f} \right) + \boldsymbol{\nabla} p^h - \boldsymbol{\nabla} \cdot \mathbf{T}^h - \boldsymbol{\nabla}(2\mu_s \boldsymbol{\varepsilon}(\mathbf{u}^h)) \right] dQ \\
&\quad + \sum_{e=1}^{(n_{el})_n} \int_{Q^e_n} \tau_{\text{CONS}} 2\mu_p \left[ \frac{1}{2\mu_p} \mathbf{S}^h + \frac{\alpha\lambda}{2\mu_p^2} \mathbf{T}^h \cdot \mathbf{S}^h + \frac{\lambda}{2\mu_p} \overset{\triangledown}{\mathbf{S}}{}^h - \boldsymbol{\varepsilon}(\mathbf{w}^h) \right] \\
&\qquad : \left[ \frac{1}{2\mu_p} \mathbf{T}^h + \frac{\alpha\lambda}{2\mu_p^2} \mathbf{T}^h \cdot \mathbf{T}^h + \frac{\lambda}{2\mu_p} \overset{\triangledown}{\mathbf{T}}{}^h - \boldsymbol{\varepsilon}(\mathbf{u}^h) \right] dQ \\
&\quad + \sum_{e=1}^{(n_{el})_n} \int_{Q^e_n} \tau_{\text{CONT}} \boldsymbol{\nabla} \cdot \mathbf{w}^h \rho \boldsymbol{\nabla} \cdot \mathbf{u}^h \, dQ \\
&\quad = \int_{(P_n)_{\mathbf{h}}} \mathbf{w}^h \cdot \mathbf{h}^h dP \,.
\end{aligned} \tag{3.2}$$

In the case where $\alpha = 0$ the above equation reduces to the DSD/SST finite element formulation of the Oldroyd-B model. For $\lambda = 0$ Eq. (3.2) further reduces to the Newtonian constitutive equation. In the latter case, the components of $\mathbf{T}$ are still additional degrees of freedom.

It should be noted that three least-squares terms enter Eq. (3.2). The first one, with the parameter $\tau_{\text{MOM}}$, stabilizes the momentum equation, the second one uses the parameter $\tau_{\text{CONS}}$ to stabilize the constitutive equation, and the last one stabilizes the continuity equation with the parameter $\tau_{\text{CONT}}$[1]. Two dif-

[1] Since $\tau_{\text{CONT}}$ is used to improve the convergence of nonlinear solvers at high Reynolds numbers, chances are high that this least-squares term can be dropped out of Eq. (3.2) for the Stokes flows considered here. This has not been investigated however.

ferent formulations are used for the stabilization paramter $\tau_{\text{CONS}}$. The first one is taken from [Behr et al. (2005)], originally developed for the Oldroyd-B model, and defined by:

$$\tau_{\text{CONS}_{old}} = \left(1 + \left(\frac{2\lambda\left|\mathbf{u}^h\right|}{h}\right)^2 + \left(\lambda\left|\boldsymbol{\nabla}\mathbf{u}^h\right|\right)^2\right)^{-\frac{1}{2}}, \tag{3.3}$$

where $h$ is a characteristic length of each element. The second one, developed within this work, uses the definition of $\tau$ given in Eq. (2.22) and thus, takes into account the whole source equation, i.e.

$$\tau_{\text{CONS}_{new}} = \left(\left(\frac{2\lambda}{\Delta t}\right)^2 + \left(\frac{2\lambda\left|\mathbf{u}^h\right|}{h}\right)^2 + \frac{1}{6}\left|\mathbf{C}^h\right|^2\right)^{-\frac{1}{2}}, \tag{3.4}$$

$$\mathbf{C}^h = \mathbf{I} - \lambda\left(\boldsymbol{\nabla}\mathbf{u}^h + \mathcal{P}(\boldsymbol{\nabla}\mathbf{u}^h)\right) + \frac{\alpha\lambda}{\mu_p}\mathbf{T}^h\,, \tag{3.5}$$

where $\mathbf{C}^h$ is a linearization of the source term in Eq. (2.10). $\mathcal{P}(\boldsymbol{\nabla}\mathbf{u}^h)$ is a permutation of $\boldsymbol{\nabla}\mathbf{u}^h$, i.e. $\mathcal{P}(\boldsymbol{\nabla}\mathbf{u}^h)\cdot\mathbf{T} = \mathbf{T}\cdot(\boldsymbol{\nabla}\mathbf{u})^T$. In the exact implementation the linearization is evaluated component-by-component. The stabilization parameters $\tau_{\text{MOM}}$ and $\tau_{\text{CONT}}$ follow standard definitions, e.g. given in [Behr and Tezduyar (1994)].

### 3.2.1. Solution Method

Eq. (3.2) has to be solved with numerical algorithms. The solution of the nonlinear system of equations is obtained using a Newton-Raphson algorithm [Dahmen and Reusken (2006)]. In a generic form, the nonlinear equation system can be written as

$$\mathbf{N}\left(\mathbf{u}, p, \mathbf{T}\right) = \mathbf{b}\,, \tag{3.6}$$

where $\mathbf{N}$ is a nonlinear operator and $\mathbf{b}$ the part that is independent of $\mathbf{u}$, $p$ and $\mathbf{T}$. $\mathbf{N}$ is approximated by a Taylor series, such that

$$\begin{gathered}\mathbf{N}\left(\mathbf{u}_i + \Delta\mathbf{u}_i, p_i + \Delta p_i, \mathbf{T}_i + \Delta\mathbf{T}_i\right) = \mathbf{b}\\ \Rightarrow \mathbf{N}\left(\mathbf{u}_i, \mathbf{T}_i\right) + \frac{\partial\mathbf{N}}{\partial\mathbf{T}}\Delta\mathbf{T}_i + \frac{\partial\mathbf{N}}{\partial p}\Delta p_i + \frac{\partial\mathbf{N}}{\partial\mathbf{u}}\Delta\mathbf{u}_i = \mathbf{b}\,.\end{gathered}$$

The Newton-Raphson algorithm follows: Starting from an initial guess $\mathbf{u}_0$, $p_0$, $\mathbf{T}_0$, find $\mathbf{u}_{i+1}$, $p_{i+1}$, $\mathbf{T}_{i+1}$ by solving for $\Delta\mathbf{u}_i$, $\Delta p_i$, $\Delta\mathbf{T}_i$:

$$\left[\begin{array}{c|c|c}\frac{\partial\mathbf{N}}{\partial\mathbf{u}} & \frac{\partial\mathbf{N}}{\partial p} & \frac{\partial\mathbf{N}}{\partial\mathbf{T}}\end{array}\right]\begin{pmatrix}\Delta\mathbf{u}_i\\ \Delta p_i\\ \Delta\mathbf{T}_i\end{pmatrix} = \mathbf{b} - \mathbf{N}\left(\mathbf{u}_i, p_i, \mathbf{T}_i\right) \tag{3.7}$$

$$\Rightarrow \mathbf{K}\begin{pmatrix}\Delta\mathbf{u}_i\\ \Delta p_i\\ \Delta\mathbf{T}_i\end{pmatrix} = \mathbf{r}\,, \tag{3.8}$$

$$\text{with}\quad \mathbf{u}_{i+1} = \mathbf{u}_i + \Delta\mathbf{u}_i\,,\quad p_{i+1} = p_i + \Delta p_i\,,\quad \mathbf{T}_{i+1} = \mathbf{T}_i + \Delta\mathbf{T}_i\,.$$

$\mathbf{r} = \mathbf{b} - \mathbf{N}\left(\mathbf{u}_i, p_i, \mathbf{T}_i\right)$ is called the residual or right hand side (RHS) vector and corresponds to Eq. (3.2). $\mathbf{K}$ is the linearization of the RHS. It is called stiffness or left hand side (LHS) matrix. $\Delta\mathbf{u}_i$, $\Delta p_i$ and $\Delta\mathbf{T}_i$ are the unknown velocity, pressure and stress increments of the current time-step, respectively. Eq. (3.8) is a linear equation system and thus, can be solved by a linear solver. As long as the sparse matrix $\mathbf{K}$ is positive-definite iterative solvers can be used to find a solution and are the most efficient way to solve such big systems. Here, the generalized minimal residual method (GMRES) [Saad and Schultz (1986)] is used, which is a Krylov subspace method. For more informations on this specific GMRES solver, see [Behr (1992)]. Eq. (3.8) is evaluated several times until the residual is smaller than a certain limit.

### 3.2.2. Implementation

The main effort in the implementation process is the formation of element-level matrices and residuals which involves a component-by-component evaluation of Eq. (3.2) and its linearization. Especially in three space dimensions these evaluations become quite complex and time consuming since one has to deal with 10 degrees of freedom. The element-level formation is explained using three example terms, namely:

1. $\dfrac{1}{2\mu_p}\mathbf{S} : \mathbf{T}\,,$
2. $\dfrac{\lambda}{2\mu_p}\mathbf{S} : \left(\mathbf{u}\cdot\boldsymbol{\nabla}\mathbf{T}\right)\,,$
3. $\dfrac{\alpha\lambda}{2\mu_p^2}\mathbf{S} : \left(\mathbf{T}\cdot\mathbf{T}\right)\,.$

All three terms are contained in the Galerkin part of the constitutive law in Eq. (3.2). The element-level matrix which corresponds to the first term can be easily derived since it is fully linear in the stresses. The component-by-component evaluation gives:

$$\begin{aligned}
&\frac{1}{2\mu_p}\mathbf{S} : \mathbf{T} \\
&= \frac{1}{2\mu_p}\begin{bmatrix} S_1 & S_2 & S_3 \\ S_2 & S_4 & S_5 \\ S_3 & S_5 & S_6 \end{bmatrix} : \begin{bmatrix} T_1 & T_2 & T_3 \\ T_2 & T_4 & T_5 \\ T_3 & T_5 & T_6 \end{bmatrix} \\
&\Rightarrow \frac{1}{2\mu_p}\begin{pmatrix} S_1^a \\ S_2^a \\ S_3^a \\ S_4^a \\ S_5^a \\ S_6^a \end{pmatrix}\begin{bmatrix} N_aN_b & 0 & 0 & 0 & 0 & 0 \\ 0 & 2N_aN_b & 0 & 0 & 0 & 0 \\ 0 & 0 & 2N_aN_b & 0 & 0 & 0 \\ 0 & 0 & 0 & N_aN_b & 0 & 0 \\ 0 & 0 & 0 & 0 & 2N_aN_b & 0 \\ 0 & 0 & 0 & 0 & 0 & N_aN_b \end{bmatrix}\begin{pmatrix} T_1^b \\ T_2^b \\ T_3^b \\ T_4^b \\ T_5^b \\ T_6^b \end{pmatrix}\,. \qquad (3.9)
\end{aligned}$$

$N_a$ and $N_b$ are the shape functions of the weighting functions and trial solutions, respectively. The allocation pattern of the element-level matrix becomes:

$$\begin{pmatrix} \mathbf{w}_i \\ q_i \\ \mathbf{S}_i \end{pmatrix} \begin{bmatrix} - & - & - \\ - & - & - \\ - & - & \mathbf{X} \end{bmatrix} \begin{pmatrix} \mathbf{u}_i \\ p_i \\ \mathbf{T}_i \end{pmatrix} .$$

The second term contains both velocities and stresses. Since all DOF have to be considered as variables this term is already nonlinear. Therefore, it must be linearized regarding the stress while keeping the velocity fixed, as well as regarding the velocity while keeping the stress fixed[2], i.e.:

$$\begin{aligned} &\frac{\lambda}{2\mu_p}\mathbf{S} \colon (\mathbf{u}^* \cdot \boldsymbol{\nabla}\mathbf{T}) \\ &= \frac{\lambda}{2\mu_p} \begin{bmatrix} S_1 & S_2 & S_3 \\ S_2 & S_4 & S_5 \\ S_3 & S_5 & S_6 \end{bmatrix} : \begin{bmatrix} \mathbf{u}^* \cdot \boldsymbol{\nabla}T_1 & \mathbf{u}^* \cdot \boldsymbol{\nabla}T_2 & \mathbf{u}^* \cdot \boldsymbol{\nabla}T_3 \\ \mathbf{u}^* \cdot \boldsymbol{\nabla}T_2 & \mathbf{u}^* \cdot \boldsymbol{\nabla}T_4 & \mathbf{u}^* \cdot \boldsymbol{\nabla}T_5 \\ \mathbf{u}^* \cdot \boldsymbol{\nabla}T_3 & \mathbf{u}^* \cdot \boldsymbol{\nabla}T_5 & \mathbf{u}^* \cdot \boldsymbol{\nabla}T_6 \end{bmatrix} \\ &\Rightarrow \frac{\lambda}{2\mu_p} \begin{pmatrix} S_1^a \\ S_2^a \\ S_3^a \\ S_4^a \\ S_5^a \\ S_6^a \end{pmatrix} \begin{bmatrix} N_a\mathbf{u}^* \cdot \boldsymbol{\nabla}N_b & 0 & 0 & 0 & 0 & 0 \\ 0 & 2N_a\mathbf{u}^* \cdot \boldsymbol{\nabla}N_b & 0 & 0 & 0 & 0 \\ 0 & 0 & 2N_a\mathbf{u}^* \cdot \boldsymbol{\nabla}N_b & 0 & 0 & 0 \\ 0 & 0 & 0 & N_a\mathbf{u}^* \cdot \boldsymbol{\nabla}N_b & 0 & 0 \\ 0 & 0 & 0 & 0 & 2N_a\mathbf{u}^* \cdot \boldsymbol{\nabla}N_b & 0 \\ 0 & 0 & 0 & 0 & 0 & N_a\mathbf{u}^* \cdot \boldsymbol{\nabla}N_b \end{bmatrix} \begin{pmatrix} T_1^b \\ T_2^b \\ T_3^b \\ T_4^b \\ T_5^b \\ T_6^b \end{pmatrix} , \end{aligned} \tag{3.10}$$

and

$$\begin{aligned} &\frac{\lambda}{2\mu_p}\mathbf{S} \colon (\mathbf{u} \cdot \boldsymbol{\nabla}\mathbf{T}^*) \\ &\Rightarrow \frac{\lambda}{2\mu_p} \begin{pmatrix} S_1^a \\ S_2^a \\ S_3^a \\ S_4^a \\ S_5^a \\ S_6^a \end{pmatrix} \begin{bmatrix} N_aN_bT^*_{1,x_1} & N_aN_bT^*_{1,x_2} & N_aN_bT^*_{1,x_3} \\ N_aN_bT^*_{2,x_1} & N_aN_bT^*_{2,x_2} & N_aN_bT^*_{2,x_3} \\ N_aN_bT^*_{3,x_1} & N_aN_bT^*_{3,x_2} & N_aN_bT^*_{3,x_3} \\ N_aN_bT^*_{4,x_1} & N_aN_bT^*_{4,x_2} & N_aN_bT^*_{4,x_3} \\ N_aN_bT^*_{5,x_1} & N_aN_bT^*_{5,x_2} & N_aN_bT^*_{5,x_3} \\ N_aN_bT^*_{6,x_1} & N_aN_bT^*_{6,x_2} & N_aN_bT^*_{6,x_3} \end{bmatrix} \begin{pmatrix} u_1^b \\ u_2^b \\ u_3^b \end{pmatrix} , \end{aligned} \tag{3.11}$$

where $T^*_{i,x_j}$ means the derivative of $T^*_i$ against $x_j$. This leads to the following allocation pattern of the element-level matrix:

$$\begin{pmatrix} \mathbf{w}_i \\ q_i \\ \mathbf{S}_i \end{pmatrix} \begin{bmatrix} - & - & - \\ - & - & - \\ \mathbf{X} & - & \mathbf{X} \end{bmatrix} \begin{pmatrix} \mathbf{u}_i \\ p_i \\ \mathbf{T}_i \end{pmatrix} .$$

[2] Fixed DOFs from previous nonlinear iteration are marked by an asterisk ($*$).

In the last of the example terms there is a special case. The stresses occur in nonlinear form. Like in the terms before, for the Galerkin part the exact gradient, i.e. $\frac{\partial(\mathbf{T}\cdot\mathbf{T})}{\partial\mathbf{T}}$, has to be derived component-by-component:

$$
\begin{aligned}
&\frac{\alpha\lambda}{2\mu_p^2}\mathbf{S}\colon \mathbf{T}\cdot\mathbf{T} \\
&= \frac{\alpha\lambda}{2\mu_p^2}\begin{bmatrix} S_1 & S_2 & S_3 \\ S_2 & S_4 & S_5 \\ S_3 & S_5 & S_6 \end{bmatrix} : \begin{bmatrix} T_1^2+T_2^2+T_3^2 & T_2(T_1+T_4)+T_3T_5 & T_3(T_1+T_6)+T_2T_5 \\ T_2(T_1+T_4)+T_3T_5 & T_2^2+T_4^2+T_5^2 & T_3T_2+T_5(T_4+T_6) \\ T_3(T_1+T_6)+T_2T_5 & T_3T_2+T_5(T_4+T_6) & T_3^2+T_5^2+T_6^2 \end{bmatrix} \\
&\Rightarrow \frac{\alpha\lambda}{\mu_p^2}\begin{pmatrix} S_1^a \\ S_2^a \\ S_3^a \\ S_4^a \\ S_5^a \\ S_6^a \end{pmatrix} \begin{bmatrix} N_aN_bT_1^* & N_aN_bT_2^* & N_aN_bT_3^* & 0 & 0 & 0 \\ N_aN_bT_2^* & N_aN_b(T_1^*+T_4^*) & N_aN_bT_5^* & N_aN_bT_2^* & N_aN_bT_3^* & 0 \\ N_aN_bT_3^* & N_aN_bT_5^* & N_aN_b(T_1^*+T_6^*) & 0 & N_aN_bT_2^* & N_aN_bT_3^* \\ 0 & N_aN_bT_2^* & 0 & N_aN_bT_4^* & N_aN_bT_5^* & 0 \\ 0 & N_aN_bT_3^* & N_aN_bT_2^* & N_aN_bT_5^* & N_aN_b(T_4^*+T_6^*) & N_aN_bT_5^* \\ 0 & 0 & N_aN_bT_3^* & 0 & N_aN_bT_5^* & N_aN_bT_6^* \end{bmatrix} \begin{pmatrix} T_1^b \\ T_2^b \\ T_3^b \\ T_4^b \\ T_5^b \\ T_6^b \end{pmatrix} .
\end{aligned}
\tag{3.12}
$$

The allocation pattern of the element-level matrix is the same as for the first example.

It should be noted that to simplify the integration in standard FEM, every element is transformed to a reference element and that the integration itself is performed numerically. Therefore, every term in the element-level matrix and in the residual vector has to be multiplied by the Jacobian of the transformation and the integration weight of the current quadrature point. For this purpose, the algorithm loops over all elements, quadrature points and twice over all element nodes to evaluate the element-level stiffness matrix $\mathbf{K}^e$ and residual vector $\mathbf{r}_a^e$.

### 3.2.3. Boundary Conditions

Setting appropriate boundary conditions for viscoelastic liquids becomes a complicated task. In most cases it is not possible to give exact inflow conditions, e.g., because of shear-thinning effects. However, in case of the Oldroyd-B model it is possible to derive analytic inflow boundary conditions, since shear-thinning is not represented by the model. If a fully-developed flow in $x_1$-direction, defined as a parabolic distribution $u_1(x_2, x_3)$, is assumed, i.e. $u_2 = u_3 = \frac{\partial}{\partial x_1} = 0$, Eq. (2.8) reduces to the additional stress inflow boundary conditions for an Oldroyd-B type liquid:

$$
\begin{aligned}
T_1 &= 2\lambda\mu_p\left[\left(\frac{\partial u_1}{\partial x_2}\right)^2 + \left(\frac{\partial u_1}{\partial x_3}\right)^2\right] , \\
T_2 &= \mu_p\frac{\partial u_1}{\partial x_2} , \\
T_3 &= \mu_p\frac{\partial u_1}{\partial x_3} , \\
T_4 &= T_5 = T_6 = 0 .
\end{aligned}
\tag{3.13}
$$

In Appendix A.1 the whole derivation of the above equations is shown.

Unfortunately, for the Giesekus model such an analytic derivation is not possible. In [Xie and Pasquali (2004)] a method is presented to compute the inflow stresses numerically. In fully developed flows the inertia part of the constitutive equation (cf. Eq. (2.10)) drops out and only source terms are left, i.e.,

$$\mathbf{D} := \mathbf{T} - \lambda \left( \boldsymbol{\nabla}\mathbf{u} \cdot \mathbf{T} + \mathbf{T} \cdot (\boldsymbol{\nabla}\mathbf{u})^T \right) + \frac{\alpha\lambda}{\mu_p} \mathbf{T} \cdot \mathbf{T} - 2\mu_p \boldsymbol{\varepsilon}(\mathbf{u}) = \mathbf{0} \,. \tag{3.14}$$

If the velocity gradient is known, $\mathbf{D}$ depends only on the stress components $\mathbf{T}$. Hence, these components can be determined using a Newton-Raphson algorithm: Guess $\mathbf{T}_0$, find $\mathbf{T}_{i+1}$ such that

$$\frac{\partial \mathbf{D}}{\partial \mathbf{T}} \Delta \mathbf{T}_i = \mathbf{D} \,. \tag{3.15}$$

A pseudo code of the method implemented in XNS is shown in Algorithm 3.1. It should be noted that the solution of Eq. (3.15) is still not the exact inflow boundary condition. Since shear-thinning effects are introduced by the Giesekus model, the exact inflow velocity is unknown and thus, also its gradient in Eq. (3.14). In the test cases considered here the introduced error remains very local to the inflow and does not affect the rest of the solution. However, if the elasticity of the liquid increases, the shear-thinning as well as the introduced error becomes stronger.

**Algorithm 3.1** Pseudo code of the computation of inflow stresses for the Giesekus liquid

```
for $n = 1$ to number of nodes on inflow BC do
    $\mathbf{T}_0 \leftarrow \mathbf{T}_{\text{Oldroyd-B}}$
    while $|\Delta \mathbf{T}_i| \geq \epsilon$ do
        compute $\mathbf{D}$
        compute $\frac{\partial \mathbf{D}}{\partial \mathbf{T}}$
        solve Eq. (3.15) with LR-decomposition
        $\mathbf{T}_{i+1} \leftarrow \mathbf{T}_i + \Delta \mathbf{T}_i$
    end while
end for
```

The outflow boundary is a further challenge compared to Newtonian liquids. An outflow condition where the velocities in tangential direction and the normal stresses are forced to be zero is usually not sufficient in the viscoelastic case (cf. Sec. 4.1). A compatible outflow is needed which can be obtained by an open boundary condition [Papanastasiou et al. (1992)]. So far, the open boundary condition for the viscoelastic case is not implemented in XNS, which should be a future task. However, the first mentioned boundary condition is applicable in free-surface flows, since the traction vanishes for a block velocity profile.

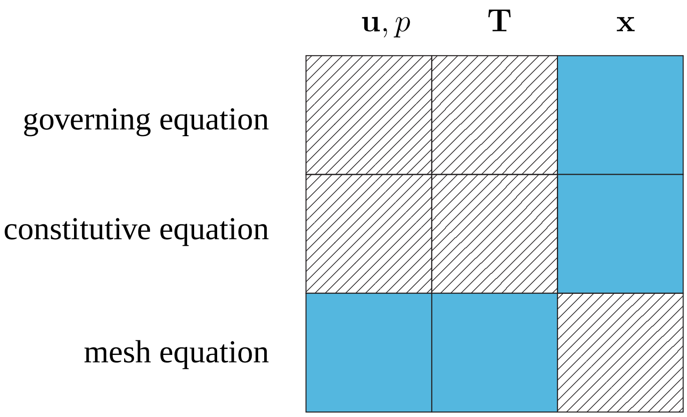


Figure 3.1.: Coupling of the equations as it has been implemented. The shaded regions signify the coupled parts, whereas the blue regions have not been implemented.

### 3.2.4. General Representation of Free Surfaces

If the free surface arising through die swell is considered, the mesh kinematic condition has to be added to the equation system. Since the problem still admits a steady solution, it would be possible to solve the governing equations, the constitutive law and the mesh equations in a fully coupled manner. This coupling would involve an enormous implementation effort. Therefore, the problem is regarded as an unsteady problem, where the steady shape of the free surface is approached over several time-steps. This leaves the option to uncouple the mesh equation from the flow solution. Figure 3.1 illustrates the coupling of the equations as it has been implemented. The shaded regions signify the coupled parts, whereas the blue regions have not been implemented. In DSD/SST, the deformation of the spatial domain is taken into account automatically by writing the variational formulation of the problem over the associated space-time domain. Only the boundary nodes are restricted to the movement of the interface, which is enforced by the following condition:

$$\mathbf{v} \cdot \mathbf{n} = \mathbf{u} \cdot \mathbf{n}, \tag{3.16}$$

with $\mathbf{v}$ being the velocity of the nodes on the boundary and $\mathbf{n}$ denoting the normal vector at the node. This condition leaves several possible options for the choice of $\mathbf{v}$, as the tangent components can be chosen freely. Figure 3.2 illustrates the boundary condition described above and indicates some displacement vectors $\mathbf{v}$ which are possible for the given example of $\mathbf{u}$ and $\mathbf{n}$. In the following, $\mathbf{v}$ is chosen to be equal to the normal component of $\mathbf{u}$. Moving the boundary nodes in this way satisfies the kinematic condition, but at the same time, node clustering due to strong movement in tangential direction is limited. However, the movement is sensitive to the choice of the normal $\mathbf{n}$.

As long as the movement of the boundary nodes stays within a certain range, the EMUM technique described in Sec. 2.2.3.2 is sufficient to account for the mesh motion without the need for remeshing.

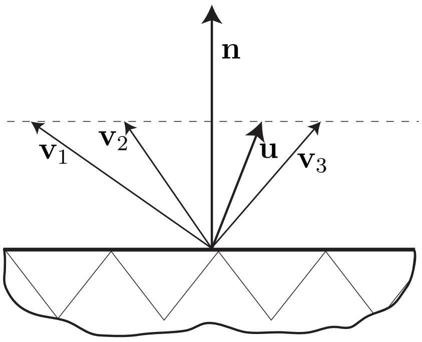


Figure 3.2.: Possible displacement directions satisfying the kinematic boundary condition.

## 3.3. NURBS Representation of Free Surfaces

The standard approach for calculating the movement of free surfaces is to compute the displacement of the nodes directly from the computational mesh using the ALE formulation described in Sec. 2.2.3.1. The direction of the displacement is given by the mesh velocity and the normals at each node (cf. Sec. 3.2.4). Those nodal normals are computed by averaging the normals of the elements sharing that node. Especially for unstructured meshes in three dimensions such a computation of the normals can become very inaccurate. In Chapter 5 examples are shown where these inaccuracies destroy the entire solution.

To circumvent this issue, one solution would be to refine the mesh until the computation of the normals is sufficiently accurate. Such a mesh is much finer as it has to be for the flow computation, leading to unnecessary computational effort. Furthermore, at sharp boundaries, like edges, the computation of the normals can still fail. Therefore, in this work the free surface is approximated by a Nonuniform Rational B-Spline (NURBS) which, as will be shown, can drastically improve its accuracy. In the following section this NURBS representation is described in detail, first, giving a short introduction to NURBS itself.

### 3.3.1. Nonuniform Rational B-Splines (NURBS)

NURBS curves and surfaces are a mathematical representation of 2-D and 3-D geometries. Due to their accuracy, flexibility and efficiency in describing simple 2-D lines, like a circle, as well as complex 3-D shapes, like a car body, NURBS have become a standard in computer aided design (CAD) technologies. Furthermore, NURBS have gained wide acceptance in the CFD community, e.g. in the field of shape optimization, or in the very new isogeometric analysis, where NUBRS are used for the FE discretization, e.g. see [Hughes et al. (2004)].

In general, NURBS are defined by four ingredients: A certain degree, several control points, so-called knot vectors and an evaluation rule. This short introduction focuses on NURBS of degree three and knot vectors in the range $[0, 1]$ only. For a more general and detailed description of NURBS refer to [Piegel and Tiller (1997)].

The underlying structure of NURBS rests upon the basis functions. For a degree equal to three these basis functions are cubic polynomials (cf. Fig. 3.3), defined on a fixed range $\boldsymbol{t} = \{t_0, \dots, t_k\} \in [0, 1]$. $\boldsymbol{t}$ is the knot vector and $t_i \leq t_{i+1}$, $i = 0, \dots, k-1$, its associated knots. The basis functions of degree three are computed in a recursive manner, i.e.

$$b_{i,0}(\tau) = \begin{cases} 1, & \text{if } t_i \leq \tau < t_{i+1} \\ 0, & \text{otherwise} \end{cases} \quad , \tag{3.17}$$

$$b_{i,3}(\tau) = \frac{\tau - t_i}{t_{i+3} - t_i} b_{i,2}(\tau) + \frac{t_{i+4} - \tau}{t_{i+4} - t_{i+1}} b_{i+1,2}(\tau) \, , \tag{3.18}$$

where the half-open interval $\tau \in [t_i, t_{i+1})$ is called the $i$-th knot span. The first derivative of the cubic basis functions can be evaluated as:

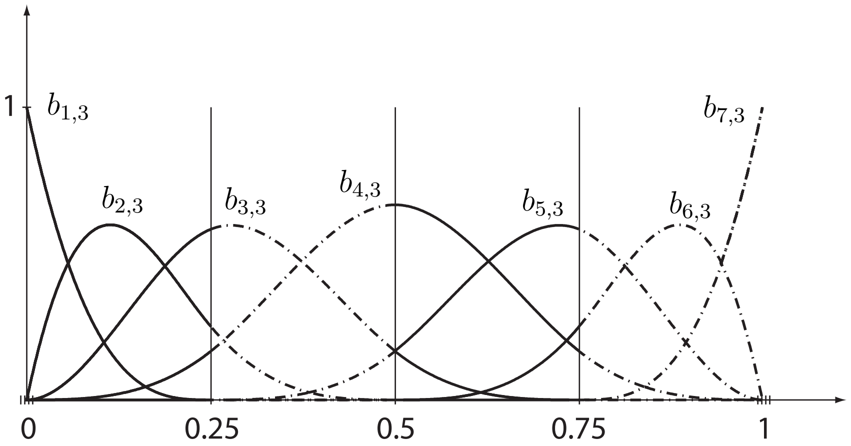


Figure 3.3.: Cubic NURBS basis functions, $\boldsymbol{t} = \{0, 0, 0, 0, 0.25, 0.5, 0.75, 1, 1, 1, 1\}$ (copied and reproduced from [Sauerland (2008)] and [Piegel and Tiller (1997)], respectively).

$$\frac{d\, b_{i,3}(\tau)}{d\, \tau} = b'_{i,3}(\tau) = 3 \left( \frac{b_{i,2}(\tau)}{t_{i+3} - t_i} - \frac{b_{i+1,2}(\tau)}{t_{i+4} - t_{i+1}} \right) \, . \tag{3.19}$$

With the aid of the cubic basis functions $b_{i,3}(\tau)$ from Eq. (3.18) a cubic NURBS curve is defined by:

$$\boldsymbol{c}(\tau) = \frac{\sum_{i=0}^{n} b_{i,3}(\tau) w_i \boldsymbol{p}_i}{\sum_{i=0}^{n} b_{i,3}(\tau) w_i} \qquad 0 \leq u \leq 1 \, , \tag{3.20}$$

where $\boldsymbol{p}_i$ are the control points and $w_i$ the weights of each control point. Cubic NURBS curves have to fulfill some properties. The size $k+1$ of the knot vector $\boldsymbol{t}$ is given by $k = n+4$ if $n-1$ is the number of control points. Therefore, the minimal size of the knot vector is $8$, i.e. $\boldsymbol{t} = \{0, 0, 0, 0, 1, 1, 1, 1\}$

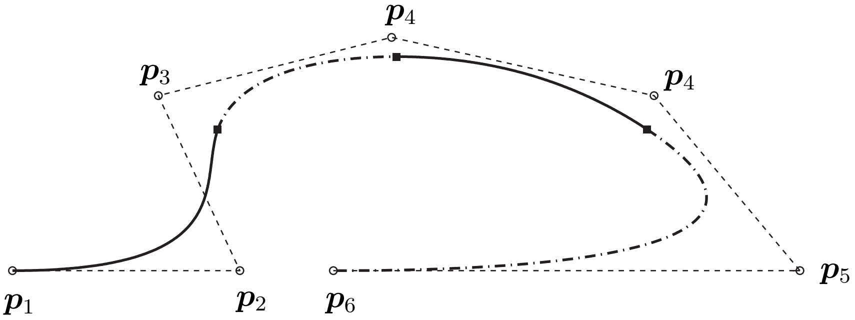


Figure 3.4.: Cubic NURBS curve using the basis functions of Figure 3.3 (copied and reproduced from [Sauerland (2008)] and [Piegel and Tiller (1997)], respectively).

as a NURBS curve must have at least two control points, one at the beginning and one at the end. Fig. 3.4 shows an example of a NURBS curve which uses the basis functions from Fig. 3.3. In this example, the weight of each control point is equal to one. In Fig. 3.4, it is also indicated which part of the spline uses which basis functions. How the behavior of a NURBS curve is influenced by the control point weights is demonstrated in Fig. 3.5.

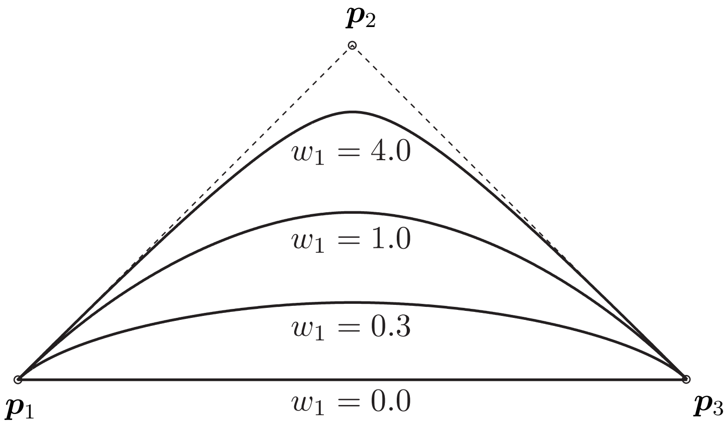


Figure 3.5.: Influence on the NURBS curve when weights are modified (copied and reproduced from [Sauerland (2008)] and [Piegel and Tiller (1997)], respectively).

The extension from NURBS curves to NURBS surfaces is straightforward. The control points $\boldsymbol{p}_{i,j}$ are arranged on a bidirectional net with two knot vectors $\boldsymbol{t}_1$ and $\boldsymbol{t}_2$. A cubic NURBS surface is than defined by:

$$\boldsymbol{s}(\tau,\eta) = \frac{\sum_{i=0}^{n}\sum_{j=0}^{m} b_{i,3}(\tau) b_{j,3}(\eta) w_{i,j} \boldsymbol{p}_{i,j}}{\sum_{i=0}^{n}\sum_{j=0}^{m} b_{i,3}(\tau) b_{j,3}(\eta) w_{i,j}} \qquad 0 \leq \tau,\eta \leq 1\,, \tag{3.21}$$

where $\tau$ and $\eta$ are the local coordinates. The first derivatives of Eq. (3.21) in direction $\tau$ and $\eta$ can be evaluated to:

$$\boldsymbol{s}_\tau = \frac{\sum_i \sum_j b'_{i,3} b_{j,3} w_{i,j} \boldsymbol{p}_{i,j} \cdot \sum_i \sum_j b_{i,3} b_{j,3} w_{i,j} - \sum_i \sum_j b_{i,3} b_{j,3} w_i \boldsymbol{p}_i \cdot \sum_i \sum_j b'_{i,3} b_{j,3} w_i}{\left( \sum_i \sum_j b_{i,3} b_{j,3} w_{i,j} \right)^2} , \qquad (3.22)$$

$$\boldsymbol{s}_\eta = \frac{\sum_i \sum_j b_{i,3} b'_{j,3} w_{i,j} \boldsymbol{p}_{i,j} \cdot \sum_i \sum_j b_{i,3} b_{j,3} w_{i,j} - \sum_i \sum_j b_{i,3} b_{j,3} w_i \boldsymbol{p}_i \cdot \sum_i \sum_j b_{i,3} b'_{j,3} w_i}{\left( \sum_i \sum_j b_{i,3} b_{j,3} w_{i,j} \right)^2} . \qquad (3.23)$$

### 3.3.2. Preprocessing

To incorporate NURBS into the simulation process, the starting point is a CAD model which also serves as a basis for the mesh generation process. For the CAD geometry, the software *Rhinoceros* is used here which is able to create NURBS surfaces. The mesh generation is done using *Pointwise*. The common preprocessing steps to include a NURBS surface into the simulation with XNS are the following:

1. An existing geometry, e.g., an extrusion die, is loaded into Rhinoceros and the swell domain is added to it using a cubic NURBS surface.

2. The number and position of NURBS control points are adjusted to obtain a computationally cheap and at the same time sufficiently accurate approximation of the anticipated swell domain.

3. The geometry and NURBS surface is imported into Pointwise and a mesh is generated.

4. A Glyph-script is used in Pointwise to export the exact local coordinates (i.e. $\tau, \eta$) of the mesh nodes on the NURBS surface.

5. With the aid of a MATLAB script, binary files are created that store all the NUBRS information necessary for XNS input, i.e. the knot vectors $\boldsymbol{t}_1$ and $\boldsymbol{t}_2$, the control points $\boldsymbol{p}_{i,j}$, the weights $w_{i,j}$ (all obtained from Rhinoceros) and the local coordinates $\tau$ and $\eta$ (obtained from Pointwise).

It should be highlighted that step 4 is a new approach to obtain the local coordinates of the mesh nodes. The method used before was to fit these coordinates by an optimization algorithm. This can become inaccurate. In contrast, the export from Pointwise leads to exact coordinates.

### 3.3.3. Surface Fitting and Normal Calculation in XNS

The NURBS calculations in XNS can be switched on by adding the keyword *"nurbsfit"* to the *xns.in* file. Then, XNS reads in the files containing the NURBS information (cf. Sec. 3.3.2) and distributes it to the processors, i.e. every processor stores the general spline informations, but governs only those local coordinates that correspond to its mesh portion. Using *"nurbsfit"* XNS is able to compute the (almost) exact normals of the free surface which are given as:

$$\boldsymbol{n}(\tau,\eta) = \frac{\boldsymbol{s}_\tau \times \boldsymbol{s}_\eta}{\left\|\boldsymbol{s}_\tau \times \boldsymbol{s}_\eta\right\|_2}\,, \tag{3.24}$$

where $\boldsymbol{s}_\tau$ and $\boldsymbol{s}_\eta$ are taken from Eqs. (3.22) and (3.23), respectively. This requires that the computation of the NURBS surface $\boldsymbol{s}$ itself, the basis functions $b_{i,3}$ and its derivatives $b'_{i,3}$ (cf. Eqs. (3.21), (3.18) and (3.19) respectively) are also implemented in XNS.

The displacement of the free surface is calculated by the accurate normals from Eq. (3.24) and the new positions of the grid points are saved temporary. To keep the NURBS surface aligned with the new grid points the control points of the NURBS are fitted with a steepest-decent algorithm (cf. Algorithm 3.2). Afterwards, the displacements of the nodes of the free surface are calculated again, now using the NURBS surface before and after the fitting. These displacements are chosen to finally move the mesh which leads to a very smooth surface structure. The entire method is also described in [Elgeti et al. (2011)].

**Algorithm 3.2** Fit routine for the NURBS surface

```
Given: The free surface defined as a set of grid points, $\boldsymbol{x}_j$, $0 \le j \le k$,
          and the NURBS surface $\boldsymbol{s}(\tau,\eta)$
Fit $\boldsymbol{p}$ as follows:
$\vartheta = 1.0$
while $\vartheta > \epsilon$ do
   Compute objective function $f = \sum_{j=0}^{k} \left\|\boldsymbol{s}(\tau_j,\eta_j) - \boldsymbol{x}_j\right\|_2^2$ .
   Compute $\nabla f$ with finite differences.
   for $i = 1$ to $n$ do
      for $j = 1$ to $m$ do
         $\boldsymbol{p}^*_{i,j} = \boldsymbol{p}_{i,j} - \vartheta \nabla f$
      end for
   end for
   if $f(\boldsymbol{p}^*_{i,j}) < f(\boldsymbol{p}_{i,j})$ then
      $\boldsymbol{p}_{i,j} = \boldsymbol{p}^*_{i,j}$
   end if
   $\vartheta = \frac{1}{2}\vartheta$
end while
```

# 4. Numerical Behavior

If viscoelasticity is introduced into the system of flow equations, one has to deal with several numerical difficulties. One big issue is the commonly know High Weissenberg Number Problem (HWNP), see e.g. [Owens and Phillips (2002)], where beyond a certain value of the Weissenberg number[1] standard solver fail in convergence. The GLS stabilization of Eq. (3.2) is only able to partially help here. In [Behr et al. (2004)] it is shown that in the two-dimensional case problems can be solved to Weissenberg numbers of around two.

This chapter is used to describe additional issues arising when dealing with three-dimensional problems and free-surface flows. Furthermore, the convergence behavior of the two different implementations of $\tau_{\text{CONS}}$ is compared. However, first, a channel flow is studied to show that the developed code leads to reasonable results.

## 4.1. Flow in a Channel

In the following, the flow of high-density polyethylene (HDPE) through a 3-D channel, discretized by a structured grid with $40 \times 30 \times 20$ elements, is considered as a test case. The 2-D outline of the channel is given in Fig. 4.1. The liquid flows in $x$-direction and has its inflow on the left and its outflow on the right. The top surface is a no-slip boundary, whereas symmetry conditions are applied to the bottom and the lateral surfaces. The material parameters of HDPE are taken from [Fink (2008)] and listed in Table 4.1. As inflow boundary condition, a parabolic velocity profile with a maximum velocity of $1.5\,\mathrm{cm/s}$ is prescribed. Both the Oldroyd-B and the Giesekus model are considered, using the inflow boundary conditions of either Eq. (3.13) or the solution of Eq. (3.15).

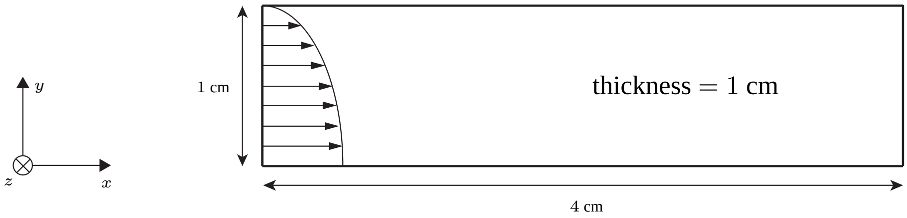


Figure 4.1.: Two-dimensional outline of the 3-D channel.

For such a flow the highest stresses occur close to the no-slip wall, whereas all the stresses should vanish at the bottom surface. Fig. 4.2 shows the computed results at the middle of the channel

[1] The Weissenberg number is a dimensionless flow number, defined by $We = \lambda\dot{\gamma}$, where $\dot{\gamma}$ is a characteristic shear rate. It is used to compare the elastic forces to the viscous effects.

| Name | Identifier | Value | Unit |
|---|---|---|---|
| Density | $\rho$ | $1$ | $\mathrm{g/cm^3}$ |
| Polymer viscosity | $\mu_p$ | $1.8 \times 10^4$ | $\mathrm{Pa \cdot s}$ |
| Solvent viscosity | $\mu_s$ | $2 \times 10^3$ | $\mathrm{Pa \cdot s}$ |
| Relaxation time | $\lambda$ | $0.3$ | s |
| Mobility factor | $\alpha$ | $0.6$ | - |

Table 4.1.: Material parameters for high-density polyethylene (HDPE) at $230°\mathrm{C}$, taken from [Fink (2008)].

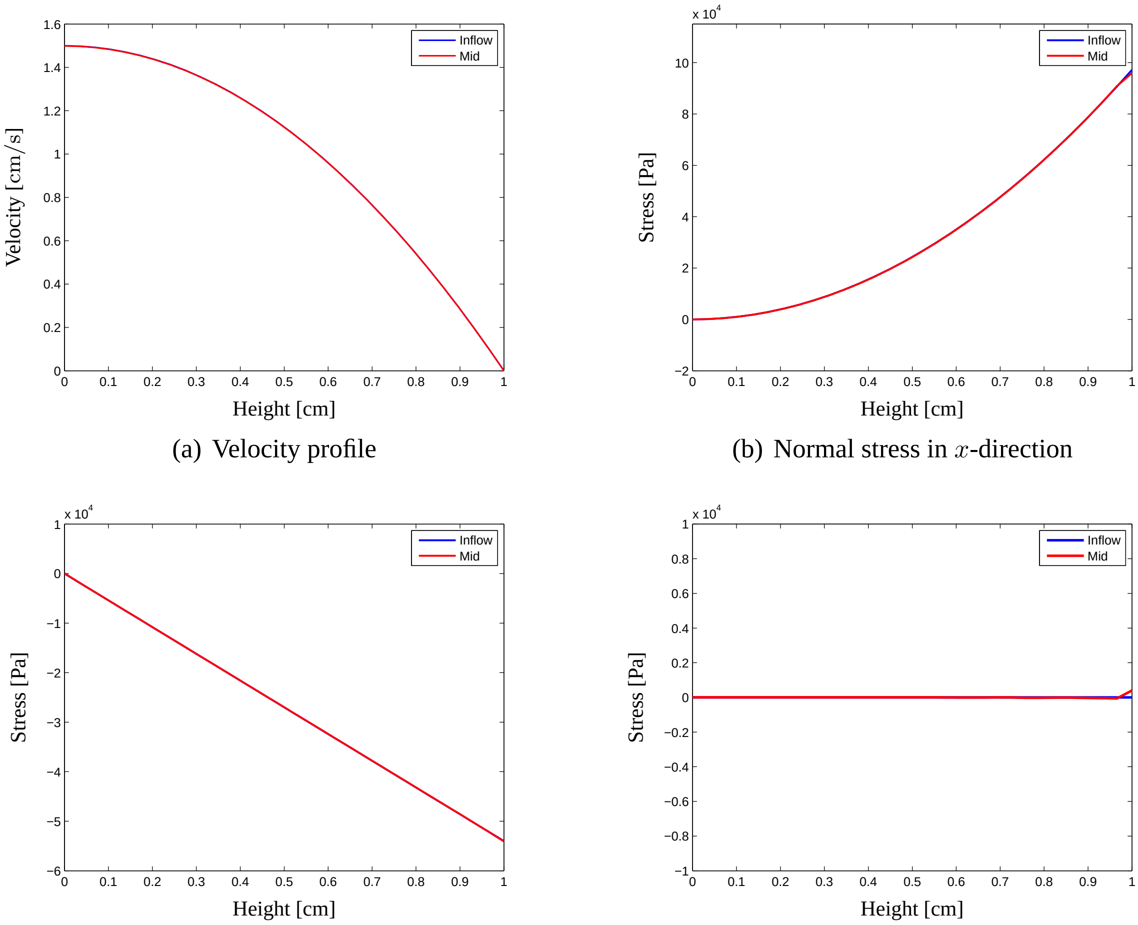
($x = 2\,\mathrm{cm}$, $z = 0.5\,\mathrm{cm}$) for the Oldroyd-B model, i.e., $\alpha = 0$, compared to the inflow boundary condition, i.e., the analytical solution. Since no shear-thinning is introduced by this model, an excellent agreement between the velocities as well as the viscoelastic stresses can be seen. Only close to the no-slip boundary an error occurs. Especially for the normal stress in $y$-direction this error can be seen clearly since it should be zero in the Oldroyd-B case.

In Fig. 4.3, the results of the same computation, this time using the Giesekus model, are shown. Now, clear shear-thinning effects can be observed in the velocity profile. Therefore, also the stresses do not match the inflow conditions. Since no analytical solution can be derived for the Giesekus model, the solution computed at the inflow, as described in Sec. 3.2.3, is used as a reference solution. However, if the Newtonian solvent becomes the dominant part in the equation system, the shear-thinning can be reduced and a better agreement between the inflow boundary and the computed values is obtained. Fig. 4.4 shows the results if solvent and polymer viscosity are interchanged, i.e. $\mu_s = 1.8 \times 10^4\,\mathrm{Pa \cdot s}$ and $\mu_p = 2 \times 10^3\,\mathrm{Pa \cdot s}$. It can be clearly seen that the error between the inflow velocity and the computed velocity, coming from the shear-thinning, is in the same order of magnitude as the error between inflow stresses and computed stresses. Unfortunately, the viscoelastic shear stress becomes very unstable close to the boundaries for this particular case. Further development is necessary to improve the no-slip and symmetry boundary conditions for the viscoelastic regime in the future.

In Fig. 4.5 the Oldroyd-B model is considered once again to compare the inflow with the outflow. Strong deviations can be observed which shows that not only the no-slip and symmetry boundary conditions, but also the outflow boundary conditions needs to be adjusted. As already mentioned in Sec. 3.2.3, an open boundary condition is necessary to deal with the traction of the viscoelastic stresses. So far, the implementation of the open boundary condition existing in XNS is not able to handle this.

## 4.2. Convergence and Stability

As already mentioned at the beginning of this chapter, it is much more complicated to get converged solutions for viscoelastic liquids compared to those of Newtonian type. A common way to approach flow problems with high Weissenberg numbers is to make use of several restarts while gradually in-



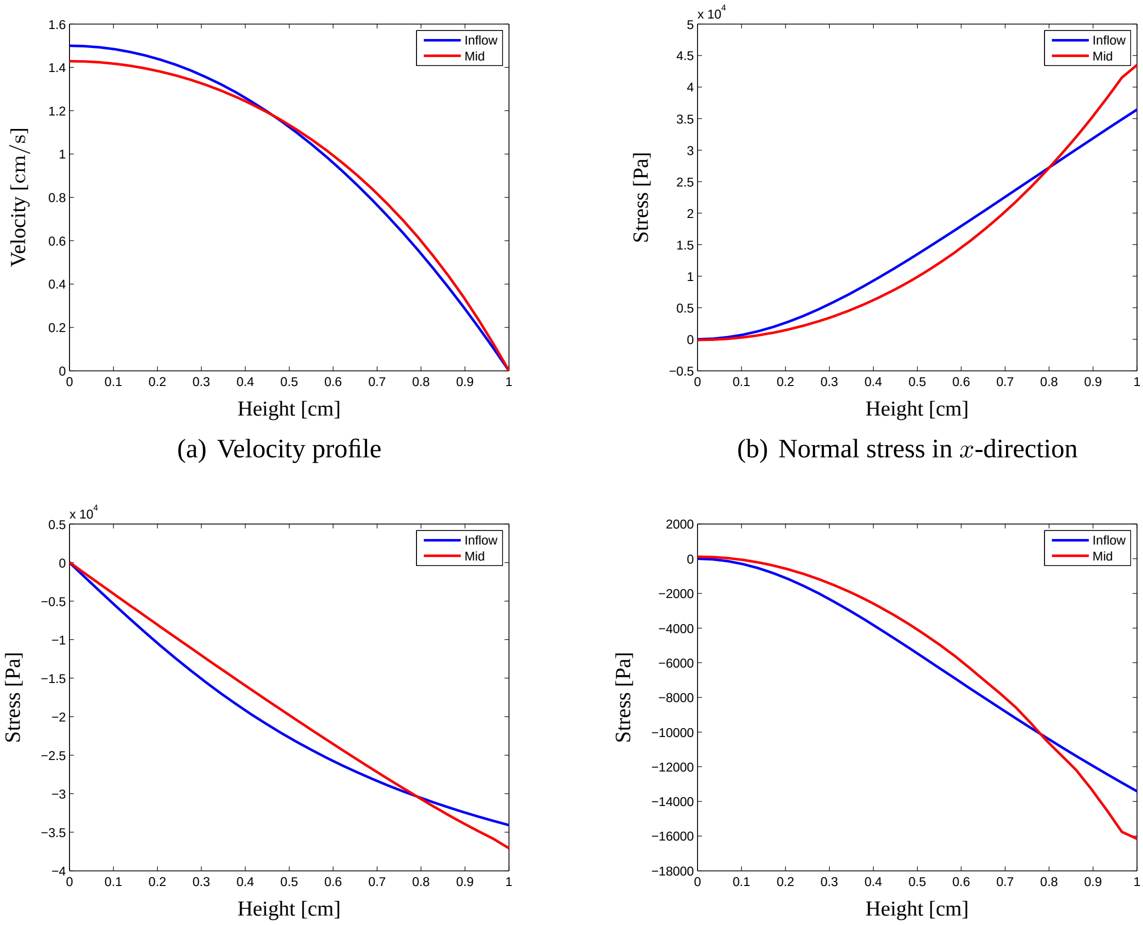
(a) Velocity profile

(b) Normal stress in $x$-direction

(c) Shear stress

(d) Normal stress in $y$-direction

Figure 4.2.: Flow of an Oldroyd-B liquid in a 3-D channel. Comparison of velocity and viscoelastic stresses at the inflow and in the middle of the channel.



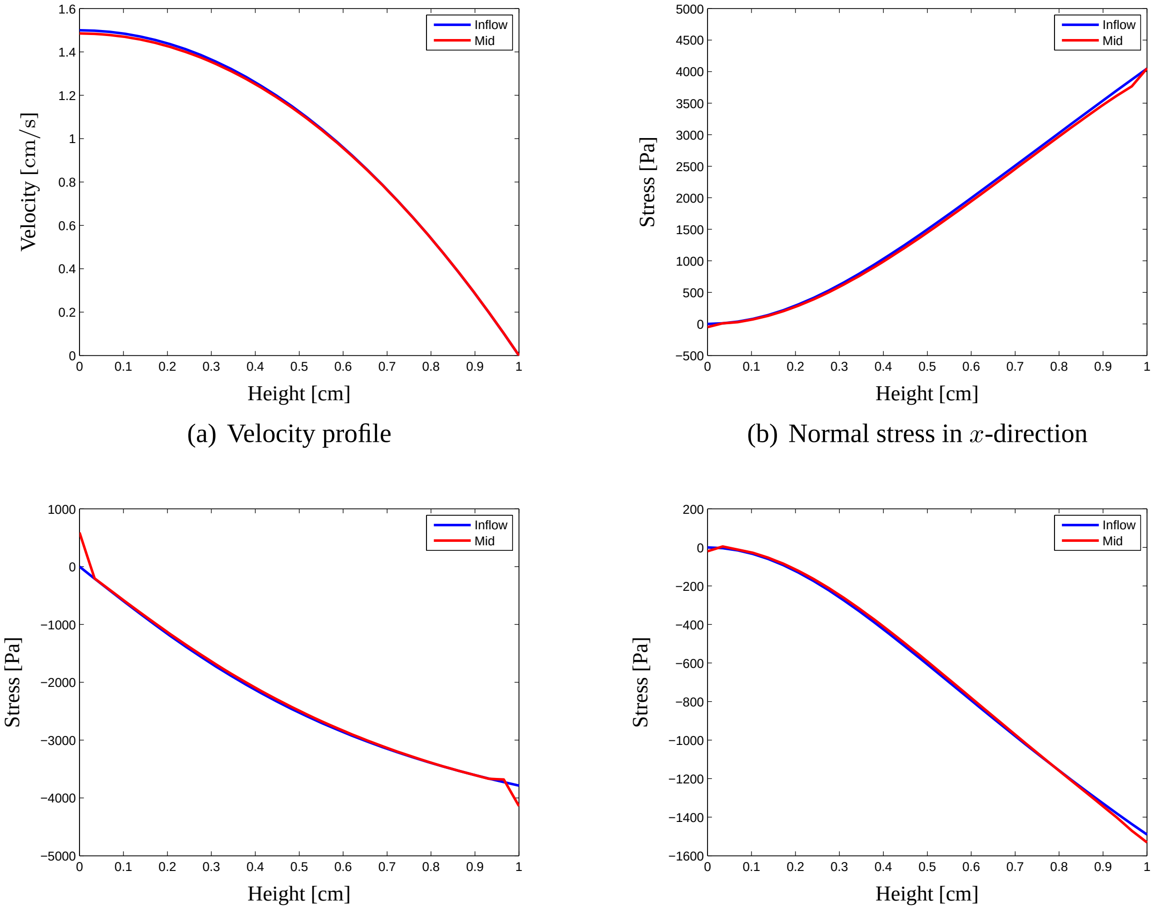
(a) Velocity profile

(b) Normal stress in $x$-direction

(c) Shear stress

(d) Normal stress in $y$-direction

Figure 4.3.: Flow of a Giesekus liquid in a 3-D channel. Comparison of velocity and viscoelastic stresses at the inflow and in the middle of the channel.



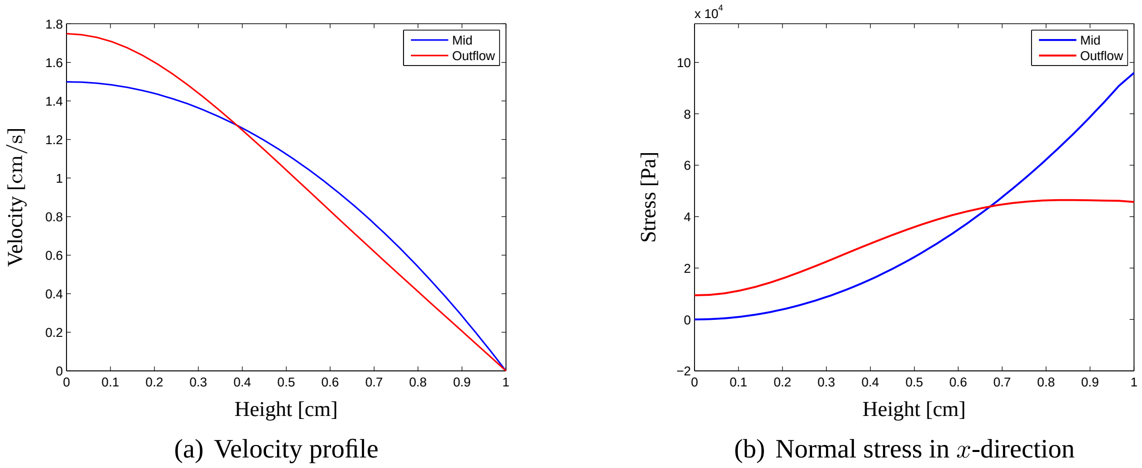
(a) Velocity profile

(b) Normal stress in $x$-direction

(c) Shear stress

(d) Normal stress in $y$-direction

Figure 4.4.: Flow of a Giesekus liquid in a 3-D channel with high solvent viscosity. Comparison of velocity and viscoelastic stresses at the inflow and in the middle of the channel.



(a) Velocity profile

(b) Normal stress in $x$-direction

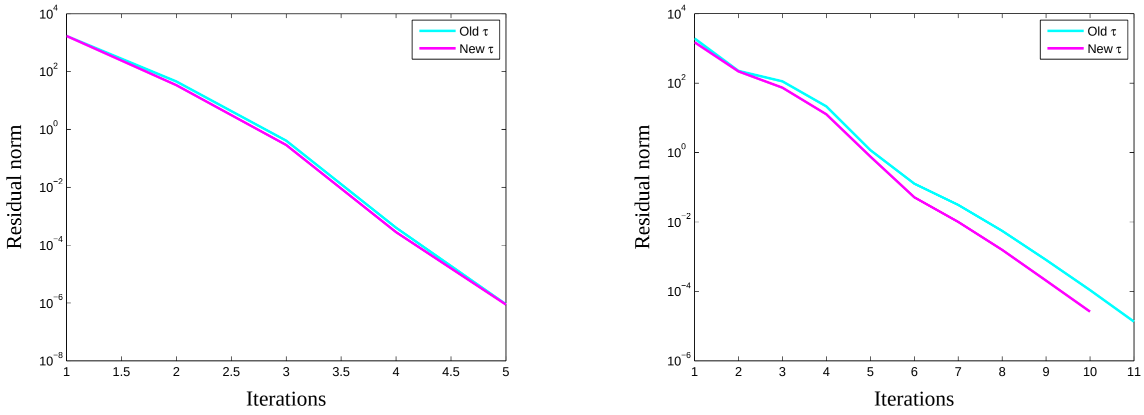
Figure 4.5.: Flow of an Oldroyd-B liquid in a 3-D channel. Comparison of velocity and viscoelastic stress at the inflow and the outflow of the channel.

creasing either the relaxation time $\lambda$ or the velocity $\mathbf{u}$. If special materials like HDPE are used it is preferable to keep the relaxation time fixed, which is a material parameter, and only increase the velocity. However, in most simulations shown in this thesis restarts were not necessary.

First, the convergence of the two different implementations of the stabilization parameter $\tau_{\text{CONS}}$ is studied. In Fig. 4.6 the different implementations, i.e. Eqs. (3.3) and (3.5), are compared to each other using the channel flow from Sec. 4.1, but this time in two-dimensional version. The abscissa shows the number of Newton-Raphson iterations and the ordinate the normalized residual of the equation system. The GMRES solver for the linear equation system terminates after 50 Krylov iterations and 5 restarts.

If the Oldroyd-B model is used, the two stabilization parameters behave more or less the same. For the Giesekus liquid the difference is also not high, but the new parameter converges with one iteration less. However, if three-dimensional simulations with slightly more complicated flow fields are considered, the behavior changes. In Fig. 4.7 the two parameters are compared for a 3-D flow in a channel with a rearrangement of the velocity field from a fully-developed to a block profile, i.e. in the front section of the channel the top surface is a no-slip boundary and in the back section it is changed to a slip boundary. The settings of the GMRES solver stay the same as before. Fig. 4.7 shows clearly that the new implementation of $\tau_{\text{CONS}}$ leads to much better convergence for both, the Oldroyd-B and the Giesekus model.

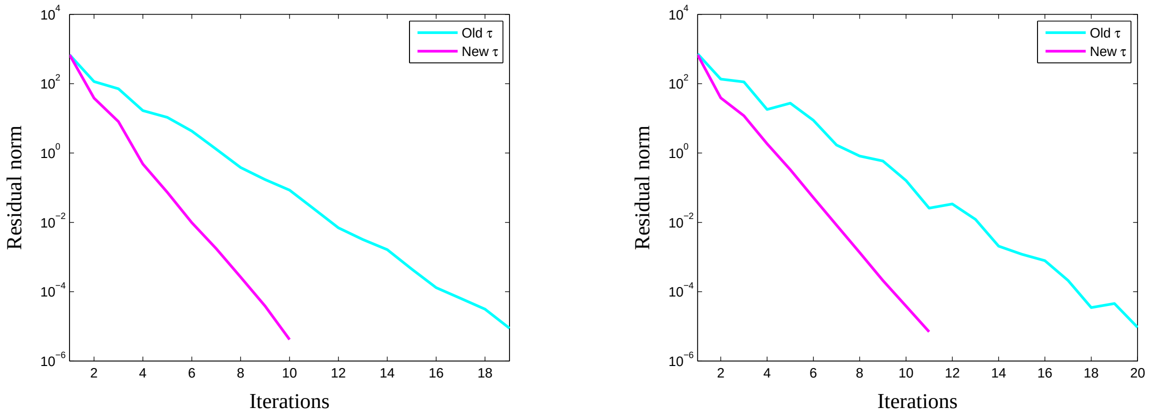
In the case of free-surface flows, the stability of the viscoelastic solver decreases dramatically. Where steady simulations converge for Weissenberg numbers in the order of $1$, the die swell simulations considered here only converge for Weissenberg numbers of around $0.01$. The reason for this



(a) Oldroyd-B liquid

(b) Giesekus liquid

Figure 4.6.: Convergence of the old and the new implementation of $\tau_{\text{const}}$ (cf. Eqs. (3.3) and (3.5)) in the channel test case (2-D).



(a) Oldroyd-B liquid

(b) Giesekus liquid

Figure 4.7.: Convergence of the old and the new implementation of $\tau_{\text{const}}$ (cf. Eqs. (3.3) and (3.5)) in a 3-D channel with rearrangement of the velocity from a fully-developed to a block profile.

bad numerical behavior seems to be that for steady computations it is possible to use several restarts to increase convergence. However, in the author's experience restarts lead to divergent solutions for free-surface simulations. Due to this behavior, only very small flow capacities can be used to compute die swell.

Another big issue arises in 3-D simulations. If unstructured meshes are used, the convergence is suppressed by strong node-to-node oscillations in the viscoelastic stresses. For structured meshes the effect reduces significantly. However, another form of oscillations can still remain visible in the solution. In a flow through a simple cylinder the effect is already visible. The cylinder has a radius of $1$ and a width of $4$. Inflow and outflow are normal to the $x$-direction and the outer surface of the cylinder is modeled as a no-slip wall. The material parameters are listed in Tab. 4.2. The relaxation time $\lambda$ of zero indicates that a Newtonian liquid is computed with the viscoelastic solver. For such a flow, the analytic solution is known and given as the inflow boundary condition as well as the initial condition. The analytic solution is computed with Eq. 3.13 and reads:

| **Name** | **Identifier** | **Value** |
|---|---|---|
| Density | $\rho$ | $1$ |
| Polymer viscosity | $\mu_p$ | $0.41$ |
| Solvent viscosity | $\mu_s$ | $0.59$ |
| Relaxation time | $\lambda$ | $0.0$ |
| Mobility factor | $\alpha$ | $0.0$ |

Table 4.2.: Material parameters for the cylinder.

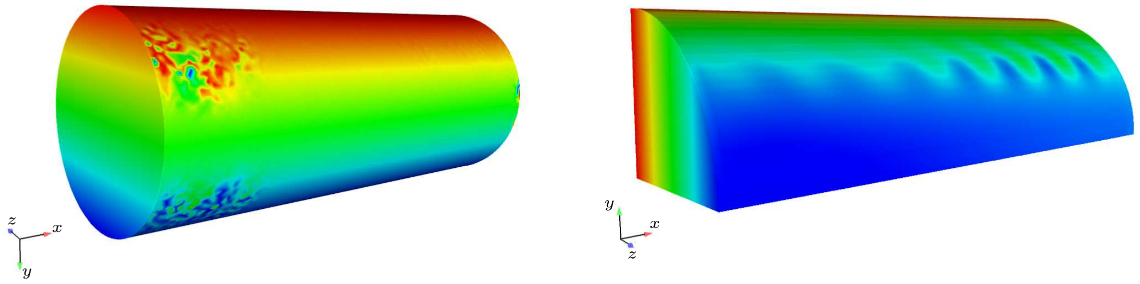
$$u(y,z) = -1.5(y^2 + z^2 - 1)\,, \tag{4.1}$$

$$T_{xy}(y) = -1.23y\,, \tag{4.2}$$

$$T_{xz}(z) = -1.23z\,, \tag{4.3}$$

where $u$, $T_{xy}$ and $T_{xz}$ are the velocity and the shear-stresses in $x$-direction, respectively. All the other velocity and stress components are zero. The flow is computed on an unstructured and a structured mesh, where in case of the structured mesh a quarter cylinder with two symmetry planes is used. The computed results are depicted in Fig. 4.8.

The example shows that the GLS stabilization is not able to smooth out the solution, even so the analytic solution of velocity and stresses is known and given as an initial condition. Other stabilization techniques like discontinuity capturing or discrete maximum principle could be tested in some future codes, but it is not guaranteed that this solves the issue. As long as these difficulties remain in the solver, it is not possible to compute 3-D extrusion dies with viscoelastic flows, since unstructured meshes are indispensable even for simple die geometries.



(a) Unstructured Mesh

(b) Structured Mesh

Figure 4.8.: Shear stress in a flow through a cylinder for an unstructured mesh (component $T_{xy}$) and a structured mesh (component $T_{xz}$). Several oscillations occur even if the relaxation time is zero.

# 5. Results

In this chapter several die swell simulations are presented. In the first part, viscoelastic die swell is studied for a contracting channel and compared to the Newtonian case. The second part is concerned with the simulation of complex swell geometries using Newtonian liquids.

## 5.1. Newtonian and Viscoelastic Die Swell for a Contracting Channel

### 5.1.1. The Two-Dimensional Case

The aim of this section is to study the behavior of the three different constitutive equations discussed in Sec. 2.2.1.1, namely the Newtonian, the Oldroyd-B and the Giesekus liquids, in the context of die swell. For this purpose, a simple 2-D contracting channel is used, which is the basis for all the following profiles. A schematic picture of the channel is shown in Fig. 5.1. Behind the inflow on the left the liquid is contracted from $2\,\mathrm{cm}$ to $0.25\,\mathrm{cm}$. After a short parallel zone, it reaches the channel exit. To be able to compute the swelling of the liquid behind the exit, the computational domain is extended by a further section, the swell area. The geometry is discretized by diagonalized structured elements. Computations have been performed on three meshes of different refinement level. For each refinement the number of computational nodes on the boundaries is doubled. The coarse mesh has $2196$ elements, the medium mesh consists of $9386$ elements and the fine mesh of $38766$ elements. As an example, the coarse mesh is shown in Fig. 5.2.

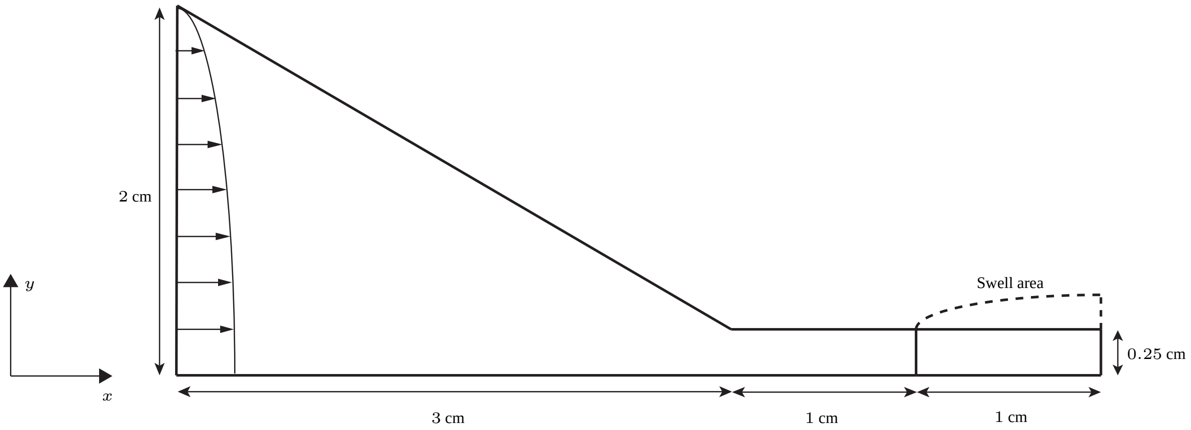


Figure 5.1.: Outline of the 2-D contracting channel.

Inflow and outflow of the channel are on the left and right, respectively. At the top wall a no-slip condition is applied and the bottom boundary is a symmetry line. Regarding the flow boundary condi-

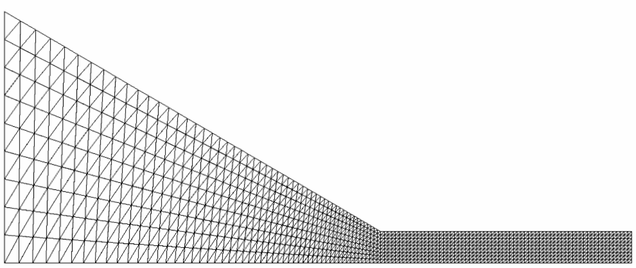
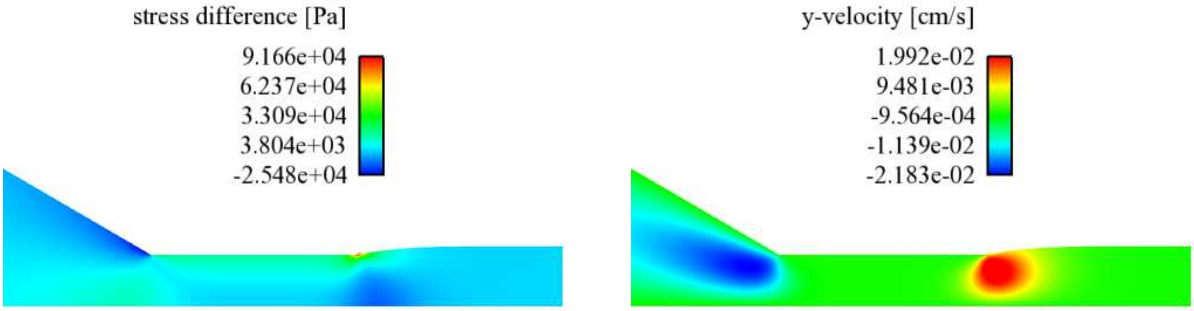


Figure 5.2.: The coarse mesh of the contracting channel with $2196$ elements.

tion at the free-surface, the usual stress-free boundary condition is imposed, whereas the deformation is governed by the mesh kinematic condition as described in Sec. 3.2.4. The free surface is determined by computing an unsteady system until a steady solution is reached. For the convergence of unsteady systems it is important to have a good initial condition. Therefore, in a first step, a steady solution of the channel is computed, while keeping the free surface fixed with a slip boundary condition for the flow. This solution is then used as an initial condition for the free-surface simulation. Once again HDPE from Tab. 4.1 is considered for the material parameters. As inflow boundary condition a parabolic velocity profile with a maximum velocity of $0.02\,\mathrm{cm/s}$ is imposed. With a time step of $0.05\,\mathrm{s}$ the steady state is reached after approximately $270$ iterations for all three models. All the results are computed with the viscoelastic solver, i.e. $\lambda = \alpha = 0$ for the Newtonian liquid and $\alpha = 0$ for the Oldroyd-B liquid.



(a) First normal stress difference

(b) Velocity in $y$-direction

Figure 5.3.: Sample solution of the Giesekus model on the medium mesh.

As an example, the solution for the Giesekus liquid on the medium mesh is depicted in Fig. 5.3. In the left picture, the first normal stress difference[1] is shown. At the transition from the die exit to the free surface a singularity occurs due to the incompatibility of the flow boundary conditions. This

[1] The first normal stress difference is defined by $N_1 = T_{xx} - T_{yy}$, where $T_{xx}$ and $T_{yy}$ is the normal stress in $x$- and $y$-direction, respectively. $N_1$ is a common measure in rheology, e.g. see [Macosko (1994)].

causes the first element of the free surface to swell more than the following one and a small kink develops. The picture on the right shows the transverse velocity, which is responsible for the swell effect. In Appendix A.3, further results, e.g. for the other models, are presented and can be compared to each other.

| Mesh | Newtonian | Oldroyd-B | Giesekus |
|---|---|---|---|
| Coarse | 1.181 | 1.240 | 1.197 |
| Medium | 1.181 | 1.220 | 1.169 |
| Fine | 1.183 | – | 1.153 |

Table 5.1.: Swell ratios for the contracting channel.

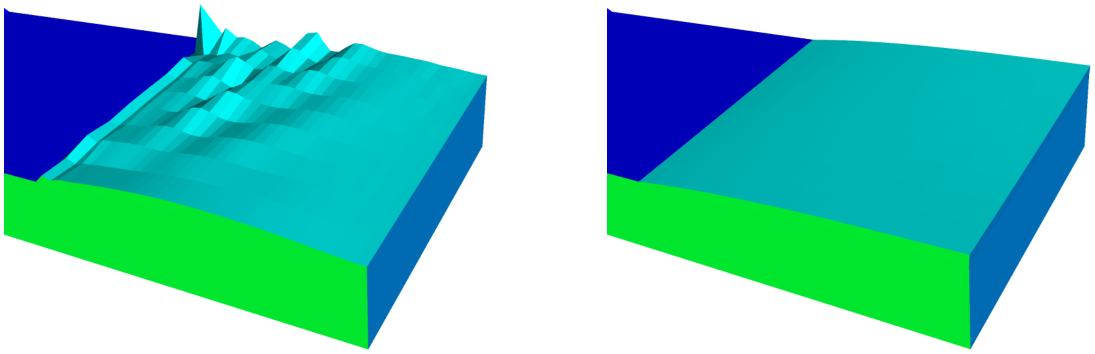
A common measure for the strength of swell is the ratio between the height of the swell profile and the height of the die exit, called swell ratio. The computed swell ratios for the different models and meshes are listed in Tab. 5.1. For the Oldroyd-B model no converged solution could be obtained for the fine mesh, which is why no swell ratio could be computed. In the Newtonian case, good results are already obtained from the coarse mesh. However, the Oldroyd-B and Giesekus models overestimate the swell ratio on the coarse meshes. Compared to each other, the Oldroyd-B model predicts the highest swell profile. For the coarse mesh the Giesekus model has a higher swell ratio then in the Newtonian case, but for refined meshes it swells less than the Newtonian liquid. In general, this is not conformant with the theory, which predicts that the swell ratio is much higher for viscoelastic fluids. A very probable explanation for this result are the very low velocities which occur in this test case (recall that convergence could not be obtained for high Weissenberg numbers, i.e., high velocities). Thus, the relaxation effects are almost negligible. Contrariwise, the Giesekus model represents shear-thinning effects, leading to a fully-developed profile which is already closer to a block-profile than in the Newtonian case. The reduced need for restructuring of the velocity-profile causes slightly reduced die swell.

### 5.1.2. Extension to Three Dimensions

The channel test case from the previous section is now extended to 3-D while keeping the proportions (cf. Fig. 5.1) and boundary conditions the same. The side walls of the 3-D channel have a distance of $1\,\mathrm{cm}$ apart and are modeled as symmetry boundary conditions. This means that physically the same problem is solved, but the 3-D solver has to be used. The simulations are performed on a structured hexahedral mesh for the Newtonian and the Giesekus model, using the grid size of the coarse mesh from the 2-D case. All the parameters stay the same as before.

Since the swell area is now a two-dimensional surface, the difficulties regarding the normal computation as described in Sec. 3.3 occur. In the general approach, the nodal normals on the free surface are computed by averaging the normals of the elements sharing each node. If the initial surface is smooth, like here, this usually works fine for the nodes in the interior of the surface. However, inaccuracies occur at the edges, where only two elements per node are available to average the normals.

Fig. 5.4(a) shows that these inaccuracies cause the free-surface simulation to fail for the chosen mesh refinement. Local inaccuracies are amplified and emanate to surrounding elements, enhancing the development of spiky surfaces even further.

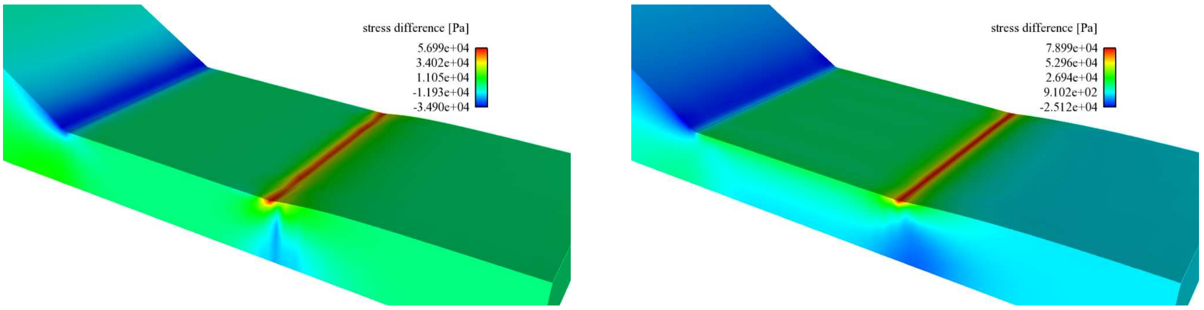


(a) General approach

(b) NURBS approximation

Figure 5.4.: Free surface of the channel with the general and the NURBS approach.

The solution is drastically improved if a NURBS approximation of the surface is used. A NURBS with only $20$ control points suffices in this case to improve the solution. As shown in Fig. 5.4(b), the surface is perfectly smooth. Even the kink originating from the stress singularity which appears in the 2-D simulations as well as in Fig. 5.4(a), disappears. For the Giesekus liquid, a swell ratio of $1.153$ is obtained which is the same value as for the fine mesh in the 2-D case (cf. Tab. 5.1). The swell ratio for the Newtonian simulation is $1.155$, thus, clearly less than in the 2-D computations. However, one can observe from Fig. 5.5, where the first normal stress difference of both models is depicted, that the Newtonian model shows slight instabilities, whereas the Giesekus model leads to a sharp stress profile. This might be the reason for the inaccurate solution in the Newtonian simulation.



(a) Newtonian

(b) Giesekus

Figure 5.5.: First normal stress differences of the 3-D contracting channel.

This test case shows that, in principle, the numerical tools are also able to perform viscoelastic die swell simulations for 3-D geometries. However, a lot of problems remain in the 3-D case. Apart from the stability issues, 3-D viscoelastic simulations are computationally very intensive. Since $10$ degrees of freedom have to be solved for at each node, the computing time increases dramatically for finer or bigger meshes.

## 5.2. Newtonian Die Swell for Complex Geometries

So far, only very simple geometries were considered for die swell simulations. This section is concerned with more complex 3-D dies. On the one hand, a more realistic circular inflow section is used and on the other hand, swell profiles with sharp edges are taken into account. The geometries are still simplified versions of real extrusion dies to save computational effort. Nevertheless, all the necessary characteristics are represented. Due to the numerical problems described in Sec. 4.2, it is not possible to solve the flow through these geometries with a viscoelastic model.

### 5.2.1. Die Swell Behind a Slit Profile

Once again, the contracting channel from Fig. 5.1 is used as the basis to construct the 3-D geometry. The inflow surface is a quarter circle, whereas the die exit is a slit of $3\,\mathrm{cm}$ length with a convex corner at the rearmost wall (cf. Fig. 5.6). The die exit is followed by the swell area becoming the free surface of the flow. Two symmetry boundaries are assumed, one at the lateral surface in the front and one at the bottom surface. The section colored in dark blue is the wall, where a no-slip condition is imposed. The material parameters are the same as in Sec. 5.1.1 for the Newtonian case, but the stresses are not computed as additional degrees of freedom. The inflow velocity is a parabolic profile with a maximum of $0.6\,\mathrm{cm/s}$. The time step is $0.1\,\mathrm{s}$.

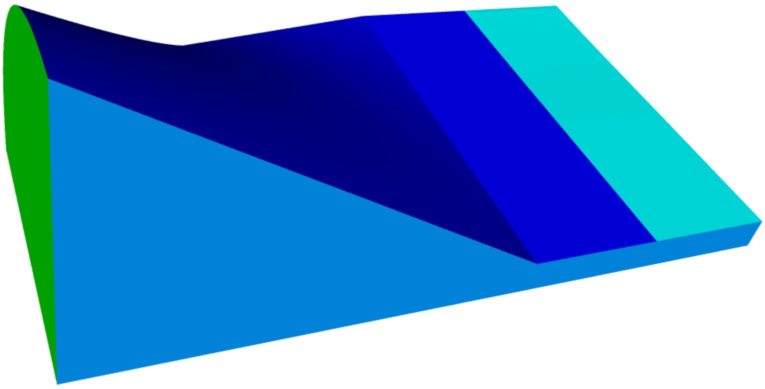


Figure 5.6.: Geometry of the slit profile. Only one quarter of the full geometry is used for the simulation.

During the time steps towards a steady solution, solely the shape of the free surface and the degrees of freedom close to the free surface change. Within the die, the velocity field stays the same. However, the majority of finite elements is situated in the interior of the die. Therefore, it is useful to reduce the complexity of the mesh by removing a certain part of the die after the initial solution has been computed. For this purpose, an intermediate surface is inserted inside of the die during the computation of the steady initial condition and the velocity field on this surface is saved. Afterwards, the upstream part of the die is removed (cf. 5.7). The intermediate surface then becomes the new inflow of the geometry, and the saved velocity field is applied as a Dirichlet boundary condition. This procedure reduces the complexity of the mesh significantly, which translates directly into reduced computational effort.

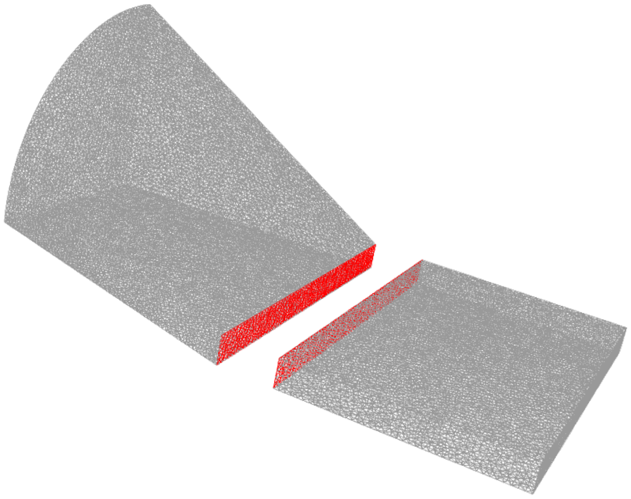
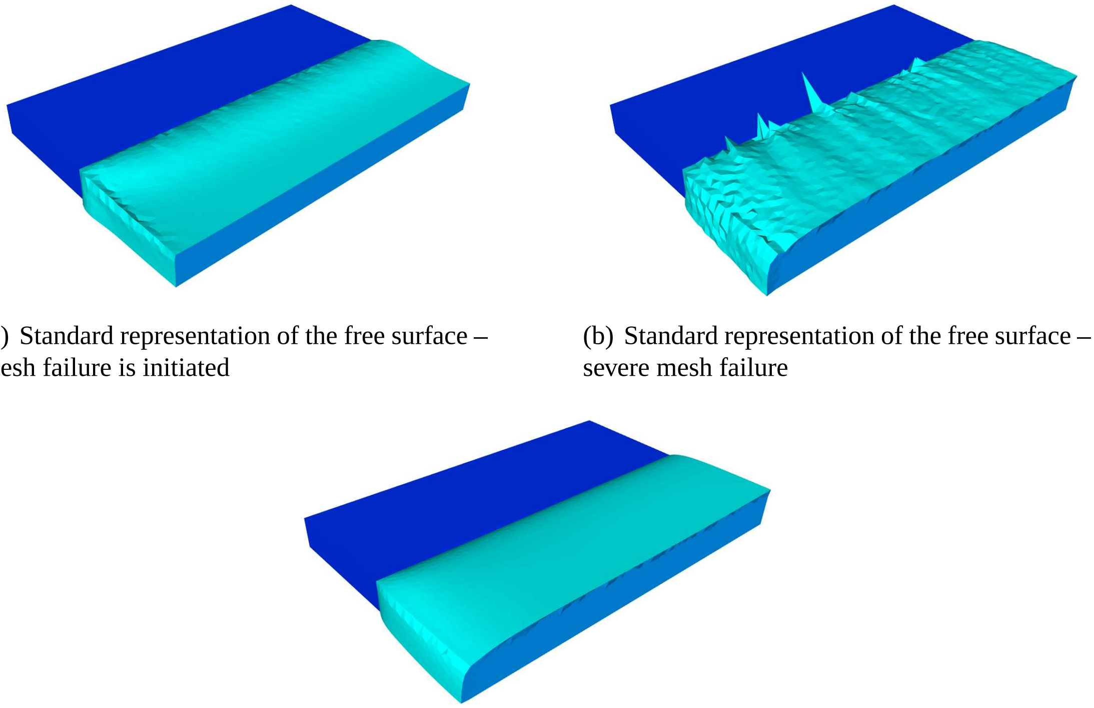


Figure 5.7.: Reduction of the mesh complexity. The upstream part of the die is removed and only the downstream part is considered in the computation of the free-surface flow.

Now, the free surface is computed using both the general and the NURBS approach (cf. Sec. 3.2.4 and Sec. 3.3). In the general approach it is not possible to compute accurate normals at the convex corner. Fig. 5.8(a) illustrates how tangling starts close to the sharp edge. If such an effect is initiated, it is enhanced as the simulation goes on until the mesh suffers from very poor quality, represented in Fig. 5.8(b). This issue cannot be solved by simply refining the mesh. As long as the convex edge remains present, it will always lead to mesh failure.

In comparison, computing the swell surface with an underlying NURBS surface leads to results of much higher quality. The free surface, shown in Fig. 5.8(c), is almost perfectly smooth. The disadvantage of the NURBS approach is that it requires an additional preprocessing step, since the NURBS surface has to be constructed and adjusted by hand with CAD tools. In this case, it means drawing the NURBS in *Rhinoceros* and setting appropriate control points by hand, such that the geometry is accurately represented with as few control points as possible to save computational time.



(a) Standard representation of the free surface – mesh failure is initiated

(b) Standard representation of the free surface – severe mesh failure

(c) NURBS approximation

Figure 5.8.: Swell profile of the slit with and without NURBS approximation. If no NURBS is used the free surface cannot be represented due to mesh failure.

Here, a $10 \times 6$ control point grid is used, shown in Fig. 5.9. The significantly improved results justify the extra effort, however.

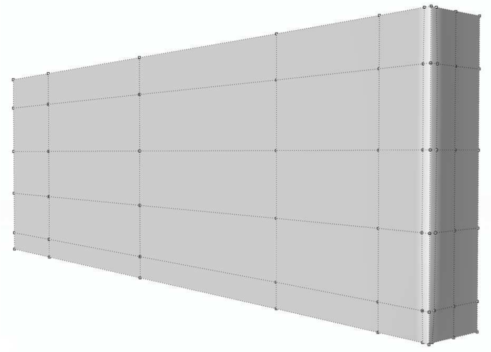


Figure 5.9.: Control point distribution of the undeformed NURBS surface.

### 5.2.2. Die Swell Behind an U-Profile

In this example, the extrusion of an U-profile is considered. The geometry of the inflow section remains the same as for the last profile, but the outflow section is modified. Again, we make use of the symmetry properties of the profile, simulating only one half of the full die. A picture of the geometry is shown in Fig. 5.10. The dimensions of the half "U" are $2\,\mathrm{cm}$ in length, $1.5\,\mathrm{cm}$ in height and $0.5\,\mathrm{cm}$ in thickness. In this case, additional complexity is introduced as compared to the slit profile of the previous section by the concave corner. The applied boundary conditions correspond

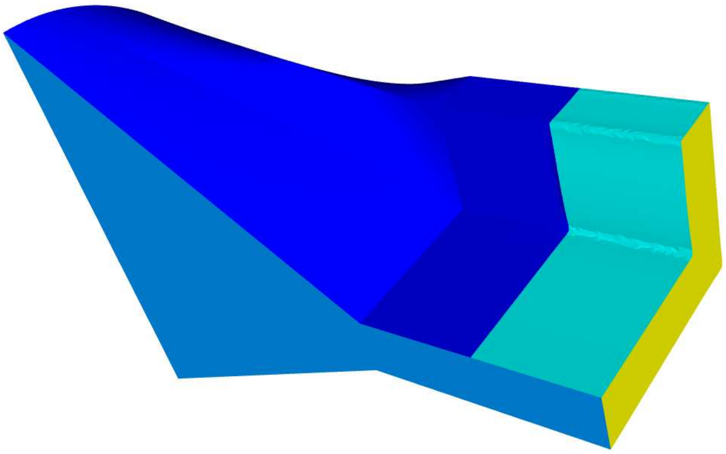
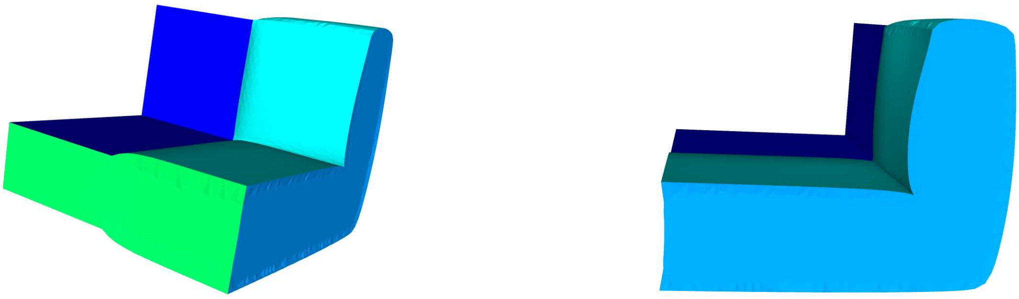


Figure 5.10.: Geometry of the U-profile. One half of the full geometry is used for the simulation.

to the ones of the previous section. Note however, that this profile contains only one instead of two symmetry planes. Furthermore, the material and flow parameters remain identical to the slit test case. Only the time step is reduced to $0.01\,\mathrm{s}$.

For the present test case, the attempt to determine the free-surface shape using the standard mesh deformation approach leads to results of even poorer quality than in the previous section. The mesh fails after already $10$ time steps and only very small deformations due to the concave edge. Contrariwise, the NURBS approach is able to handle the concave corner. Once again, a very smooth swell profile is obtained. In Fig. 5.11 the solution is shown from two different perspectives. To construct the NURBS surface a grid of $19 \times 6$ control points is used like shown in Fig. 5.12. With such a number of control points the fitting routine (cf. Algorithm 3.2) becomes a clear bottleneck in the overall computation. Using identical machines and settings, without NURBS $25\,\mathrm{time\,steps/hour}$ are computed and with NURBS only $7\,\mathrm{time\,steps/hour}$. Taking into account that this mesh is still comparatively small for a realistic extrusion die, further improvements of the fitting routine, such as more efficient optimization algorithms and maybe even possibilities for parallelization, have to be considered.



(a) Side view

(b) Frontal view

Figure 5.11.: Final shape of the free surface for the U-profile with NURBS approximation.

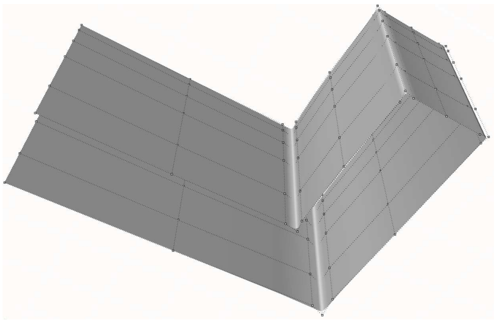


Figure 5.12.: Control point distribution of the undeformed NURBS surface.

In Fig. 5.11, one can observe that the strength of the swelling is quite different in the different sections of the profile. This is due to the fact that the flow velocity varies along the outflow. As already mentioned at the beginning of this thesis, it is the aim of the engineer to design a die with a homogeneous velocity distribution as well as a homogeneous swell profile. In [Elgeti (2011)], automatic shape optimization is used in combination with die swell simulations to perform this task without the need of several experiments and costly prototypes.

# 6. Summary and Discussion

The aim of this study was the implementation of a stabilized space-time finite element formulation of Stokes equations in conjunction with the Oldroyd-B and the Giesekus model. The implementation of the two models was then applied to die swell simulations based on viscoelastic liquids. Furthermore, a NURBS approximation technique was studied for the representation of the free-surface in 3-D die swell computations.

The implementation has been carried out in a 2-D and a 3-D version and has been validated in Sec. 4.1. A new method was implemented for the handling of viscoelastic stresses at the inflow boundary. Whereas analytical stresses can be derived for the Oldroyd-B model, the inflow stresses for the Giesekus model have to be computed numerically. Furthermore, a new stabilization parameter for the constitutive equation was developed which can improve the convergence for both models (cf. Sec. 4.2).

Even though, a lot of time was spent on new implementations within this thesis, several open issues and construction zones remain in the method.

The boundary conditions used here are still not ideal. Since viscoelastic flows are not traction-free the outflow boundary can behave very poorly (cf. Sec. 4.1). An open boundary condition that considers the influence of the viscoelastic stresses has to be implemented. To capture nonlinear effects like shear-thinning the inflow needs to be modeled as an open boundary condition as well. Even the wall boundary conditions (slip and no-slip) revealed some inaccuracies in Sec. 4.1 which should be addressed in future works.

Issues with more dramatic consequences are the strong oscillations in the viscoelastic stresses appearing in 3-D simulations, especially, for unstructured meshes. Due to these instabilities it is, so far, not possible to compute viscoelastic flows through complex geometries. The cause of the oscillations is only speculative. Maybe one reason is the behavior of the wall boundary condition, but it might be more likely that the stabilization of the constitutive equation is not sufficient here. In 3-D meshes the linear elements are not able to accurately cover curved boundary surfaces. The stresses seem to react much more sensitively to these discretization errors than the velocities or the pressure.

The other immense problem of the viscoelastic formulation is the loss of stability of the linear solver, also described in Sec. 4.2. Due to this behavior only very small fluxes can be used when computing die swell. This necessarily results in low relaxation effects. Therefore, the difference between the swell ratios of a viscoelastic and a Newtonian liquid is not as pronounced as it should be, which is also reflected in the results in Sec. 5.1. This shows, unfortunately, that the viscoelastic solver is not very useful for die swell simulations without increased stability—even for 2-D computations. Furthermore, it highlights the need for a systematic stability analysis of the viscoelastic models.

The NURBS representation of free surfaces proved to be a very useful technique to robustly compute die swell profiles. First of all, it was possible to represent smooth and accurate shapes for rather coarse meshes (cf. Sec. 5.1.2). In addition, favorable results could be obtained for complex profiles with sharp convex and concave edges (cf. Sec. 5.2). For such cases the general approach would fail even for very fine meshes. The disadvantages of this technique are a higher effort during the preprocessing, restrictions on possible free-surface motion, and increased computational time for the fitting of the NURBS surface. The latter is especially dominant, if many control points have to be used. The benefits of the NURBS representation, however, encourage to accept these disadvantages.

All in all, the results of this thesis show that the implementation of the viscoelastic models is able to handle die swell computations for simple test cases. However, a lot of work is still necessary to increase its stability for more realistic computations. Because of the NURBS representation, it is possible to determine the shape of the free surface for complex geometries. Up to now, this has only been performed with the Newtonian constitutive model.

## 6.1. Outlook

Future research should focus on the stability issues of the viscoelastic solver as it is the main problem in the simulation process. A lot of research was done in the past concerning this topic and it is still an active field. The main reason for the loss of stability seems to be the loss of positive definiteness of the stress tensor. Many people in the non-Newtonian community use the conformation tensor, in this case defined by $\mathbf{M} = \mathbf{I} + \frac{\lambda}{\mu_p}\mathbf{T}$, instead of the stress tensor $\mathbf{T}$ to solve the constitutive equation. Hulsen showed for several models that the conformation tensor (or the configuration tensor) is always positive definite if it is positive definite initially [Hulsen (1990)]. However, numerical methods can violate this condition. Two numerical methods, known to the author, have been developed to sustain the positive definiteness of the conformation tensor. The first one uses a logarithmic scaling of the conformation tensor, see [Kupferman and Fattal (2004)], and the second one is based on generalized Riccati equations [Xu and Lee (2006)]. Both methods have been successfully applied in two-dimensional viscoelastic flows at high Weissnberg numbers, see e.g. [Hulsen et al. (2005); Becker and Capatina (2010)], but with a different finite element discretization than GLS. A GLS formulation for the conformation tensor was developed in [Coronado et al. (2006)]. Using this formulation in combination with one of the two methods sustaining the positive definiteness of the conformation tensor might lead to better results. However, the extension to three dimensions will most likely lead to further complications.

Regarding the NURBS representation of free surfaces, some further work is necessary to increase the efficiency of the fitting routine. There exist faster optimization algorithms than a steepest-descent method. Furthermore, a parallelization of the method has not been addressed so far.

# 7. Acknowledgments

I wish to express my sincere gratitude to my supervisors Stefanie Elgeti and Prof. Marek Behr from the Chair for Computational Analysis of Technical Systems at RWTH Aachen University for the valuable guidance and assistance in all stages of this study. Furthermore, I would like to thank Dr. Farzin Shakib for several very useful discussions during my internship at ACUSIM Software.

# Bibliography

J. Baranger and D. Sandri. Finite element method for the approximation of viscoelastic fluid flow with a differential constitutive law. *Computational Fluid Dynamics*, 2:1021–1025, 1992.

Roland Becker and Daniela Capatina. Finite element discretization of the Giesekus model for polymer flows. *Numerical Mathematics and Advanced Applications*, 2:135–143, 2010.

M. Behr. *Stabilized Finite Element Methods for Incompressible Flows with Emphasis on Moving Boundaries and Interfaces*. PhD thesis, University of Minnesota, Department of Aerospace Engineering and Mechanics, 1992.

M. Behr. MIXD format. Technical report, Chair for Computational Analysis of Technical Systems - Aachen, `http://www.cats.rwth-aachen.de/software/formats/mixd`, 2010.

M. Behr and T.E. Tezduyar. Finite element solution strategies for large-scale flow simulations. *Computer Methods in Applied Mechanics and Engineering*, 112:3–24, 1994.

M. Behr, D. Arora, O. Coronado-Matutti, and M. Pasquali. Stabilized finite element methods of GLS type for Oldroyd-B viscoelastic fluid. In *European Congress on Computational Methods in Applied Sciences and Engineering*, 2004.

M. Behr, D. Arora, O. Coronado, and M. Pasquali. GLS-type finite element methods for viscoelastic fluid flow simulation. In K.-J. Bathe, editor, *Proceedings of the Third MIT Conference on Computational Fluid and Solid Dynamics*, pages 586–589, Cambridge, Massachussetts, 2005. Elsevier.

O. Coronado, D. Arora, M. Behr, and Matteo Pasquali. Four-field Galerkin/least-squares formulation for viscoelastic fluids. *Journal of Non-Newtonian Fluid Mechanics*, 140:132–144, 2006.

W. Dahmen and A. Reusken. *Numerik fuer Ingenieure und Naturwissenschaftler*. Springer, 2006.

J. Donea and A. Huerta. *Finite Element Methods for Flow Problems*. John Wiley & Sons, New York, USA, 2003.

S. Elgeti. *Free-Surface Flows in Shape Optimization of Extrusion Dies*. PhD thesis, RWTH Aachen University, 2011.

S. Elgeti, H. Sauerland, L. Pauli, and M. Behr. On the usage of nurbs as interface representation in free-surface flows. *International Journal for Numerical Methods in Fluids*, 2011.

B.-K. Fink. *Simulative und experimentelle Untersuchungen zum Aufschwellverhalten von Kunststoffschmelzen hinter Extrusionswerkzeugen*. PhD thesis, RWTH Aachen University, 2008.

A. Huerta and W.K. Liu. Viscous flow with large free surface motion. *Computer Methods in Applied Mechanics and Engineering*, 69:277–324, 1988.

T. J. R. Hughes. *The Finite Element Method*. Dover Publications Inc., 2000.

T.J.R. Hughes, W.K. Liu, and T.K. Zimmermann. Lagrangian-Eulerian finite element formulation for incompressible viscous flows. *Computer Methods in Applied Mechanics and Engineering*, 29: 329–349, 1981.

T.J.R. Hughes, J.A. Cottrell, and Y. Bazilevs. Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement. *Computer Methods in Applied Mechanics and Engineering*, 194:4135–4195, 2004.

M. Hulsen. A sufficient condition for a positive definite configuration tensor in differential models. *Journal of Non-Newtonian Fluid Mechanics*, 38:93–100, 1990.

M. Hulsen, R. Fattal, and R. Kupferman. Flow of viscoelastic fluids past a cylinder at high Weissenberg number: Stabilized simulations using matrix logarithms. *Journal of Non-Newtonian Fluid Mechanics*, 127:27–39, 2005.

A.A. Johnson and T.E. Tezduyar. Mesh update strategies in parallel finite element computations of flow problems with moving boundaries and interfaces. *Computer Methods in Applied Mechanics and Engineering*, 119:73–94, 1994.

R. Keunings. A survey of computational rheology. *XIIIth International Congress on Rheology, Cambridge, UK*, 2000.

J. Kunz, W. Michaeli, N. Herrlich, and W. Land. *WEKA Praxishandbuch PLUS Kunststoffpraxis: Konstruktion Band 3*. WEKA MEDIA GmbH, Kissing, 2004.

R. Kupferman and R. Fattal. Constitutive laws for the matrix-logarithm of the conformation tensor. *Journal of Non-Newtonian Fluid Mechanics*, 123:281–285, 2004.

C.W. Macosko. *Rheology: Principles, Measurements, and Applications*. Wiley-VCH, New York, USA, 1994.

R.G. Owens and T.N. Phillips. *Computational Rheology*. Imperial College Press, 2002.

T. C. Papanastasiou, N. Malamataris, and K. Ellwood. A new outflow boundary condition. *International Journal for Numerical Methods in Fluids*, 14:587–608, 1992.

L. Piegel and W. Tiller. *The NURBS Book*. Springer, Berlin, Germany, 1997.

Y. Saad and M. Schultz. GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM Journal of Scientific and Statistical Computing*, 7:856–869, 1986.

H. Sauerland. Spline approximation of the phase interface in order to determine its curvature in the context of the simulation of multiphase flows with the finite element method. Master's thesis, RWTH Aachen University, 2008.

F. Shakib. *Finite Element Analysis of the Compressible Euler and Navier-Stokes Equations*. PhD thesis, Stanford University, 1988.

T.E. Tezduyar, M. Behr, and J. Liou. A new strategy for finite element computations involving moving boundaries and interfaces – the deforming-spatial-domain/space-time procedure: I. The concept and the preliminary tests. *Computer Methods in Applied Mechanics and Engineering*, 94(3):339–351, 1992a.

T.E. Tezduyar, M. Behr, S. Mittal, and J. Liou. A new strategy for finite element computations involving moving boundaries and interfaces – the deforming-spatial-domain/space-time procedure: II. Computation of free-surface flows, two-liquid flows, and flows with drifting cylinders. *Computer Methods in Applied Mechanics and Engineering*, 94(3):353–371, 1992b.

X. Xie and M. Pasquali. A new, convenient way of imposing open-flow boundary conditions in two- and three-dimensional viscoelastic flows. *Journal of Non-Newtonian Fluid Mechanics*, 122: 159–176, 2004.

J. Xu and Y. Lee. New formulations, positivity preserving discretizations and stability analysis for non-Newtonian flow models. *Computer Methods in Applied Mechanics and Engineering*, 195: 1180–1206, 2006.

# A. Appendix

## A.1. Derivation of the Inflow Boundary Conditions for an Oldroyd-B Type Liquid

If assuming a fully-developed flow in $x_1$-direction, defined as a parabolic distribution $u_1(x_2, x_3)$, i.e., $u_2 = u_3 = \frac{\partial}{\partial x_1} = 0$, Eq. (2.8) reduces to:

$$\begin{aligned}
&\begin{bmatrix} T_1 & T_2 & T_3 \\ & T_4 & T_5 \\ sym & & T_6 \end{bmatrix} + \lambda \begin{Bmatrix} u_1 \\ u_2 \\ u_3 \end{Bmatrix} \begin{bmatrix} \frac{\partial T_1}{\partial x_1} & \frac{\partial T_2}{\partial x_2} & \frac{\partial T_3}{\partial x_3} \\ \frac{\partial T_2}{\partial x_1} & \frac{\partial T_4}{\partial x_2} & \frac{\partial T_5}{\partial x_3} \\ \frac{\partial T_3}{\partial x_1} & \frac{\partial T_5}{\partial x_2} & \frac{\partial T_6}{\partial x_3} \end{bmatrix} \\
&- \lambda \left( \begin{bmatrix} \frac{\partial u_1}{\partial x_1} & \frac{\partial u_1}{\partial x_2} & \frac{\partial u_1}{\partial x_3} \\ \frac{\partial u_2}{\partial x_1} & \frac{\partial u_2}{\partial x_2} & \frac{\partial u_2}{\partial x_3} \\ \frac{\partial u_3}{\partial x_1} & \frac{\partial u_3}{\partial x_2} & \frac{\partial u_3}{\partial x_3} \end{bmatrix} \begin{bmatrix} T_1 & T_2 & T_3 \\ & T_4 & T_5 \\ sym & & T_6 \end{bmatrix} + \begin{bmatrix} T_1 & T_2 & T_3 \\ & T_4 & T_5 \\ sym & & T_6 \end{bmatrix} \begin{bmatrix} \frac{\partial u_1}{\partial x_1} & \frac{\partial u_2}{\partial x_1} & \frac{\partial u_3}{\partial x_1} \\ \frac{\partial u_1}{\partial x_2} & \frac{\partial u_2}{\partial x_2} & \frac{\partial u_3}{\partial x_2} \\ \frac{\partial u_1}{\partial x_3} & \frac{\partial u_2}{\partial x_3} & \frac{\partial u_3}{\partial x_3} \end{bmatrix} \right) \\
&- 2\mu_p \begin{bmatrix} \frac{\partial u_1}{\partial x_1} & \frac{1}{2}(\frac{\partial u_1}{\partial x_2} + \frac{\partial u_2}{\partial x_1}) & \frac{1}{2}(\frac{\partial u_1}{\partial x_3} + \frac{\partial u_3}{\partial x_1}) \\ & \frac{\partial u_2}{\partial x_2} & \frac{1}{2}(\frac{\partial u_2}{\partial x_3} + \frac{\partial u_3}{\partial x_2}) \\ sym & & \frac{\partial u_3}{\partial x_3} \end{bmatrix} = \mathbf{0} \qquad \text{(A.1)} \\
\Rightarrow \quad &\begin{bmatrix} T_1 & T_2 & T_3 \\ & T_4 & T_5 \\ sym & & T_6 \end{bmatrix} + \lambda \begin{Bmatrix} u_1 \\ 0 \\ 0 \end{Bmatrix} \begin{bmatrix} 0 & \frac{\partial T_2}{\partial x_2} & \frac{\partial T_3}{\partial x_3} \\ 0 & \frac{\partial T_4}{\partial x_2} & \frac{\partial T_5}{\partial x_3} \\ 0 & \frac{\partial T_5}{\partial x_2} & \frac{\partial T_6}{\partial x_3} \end{bmatrix} \\
&- \lambda \left( \begin{bmatrix} 0 & \frac{\partial u_1}{\partial x_2} & \frac{\partial u_1}{\partial x_3} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} T_1 & T_2 & T_3 \\ & T_4 & T_5 \\ sym & & T_6 \end{bmatrix} + \begin{bmatrix} T_1 & T_2 & T_3 \\ & T_4 & T_5 \\ sym & & T_6 \end{bmatrix} \begin{bmatrix} 0 & 0 & 0 \\ \frac{\partial u_1}{\partial x_2} & 0 & 0 \\ \frac{\partial u_1}{\partial x_3} & 0 & 0 \end{bmatrix} \right) \\
&- 2\mu_1 \begin{bmatrix} 0 & \frac{1}{2}\frac{\partial u_1}{\partial x_2} & \frac{1}{2}\frac{\partial u_1}{\partial x_3} \\ & 0 & 0 \\ sym & & 0 \end{bmatrix} = \mathbf{0} \\
\Rightarrow \quad &\begin{bmatrix} T_1 & T_2 & T_3 \\ & T_4 & T_5 \\ sym & & T_6 \end{bmatrix} - \lambda \begin{bmatrix} 2(\frac{\partial u_1}{\partial x_2} T_2 + \frac{\partial u_1}{\partial x_3} T_3) & & sym \\ \frac{\partial u_1}{\partial x_2} T_4 + \frac{\partial u_1}{\partial x_3} T_5 & 0 & \\ \frac{\partial u_1}{\partial x_2} T_5 + \frac{\partial u_1}{\partial x_3} T_6 & 0 & 0 \end{bmatrix} - \mu_p \begin{bmatrix} 0 & \frac{\partial u_1}{\partial x_2} & \frac{\partial u_1}{\partial x_3} \\ & 0 & 0 \\ sym & & 0 \end{bmatrix} = \mathbf{0} \\
\Rightarrow \quad &\begin{Bmatrix} T_1 \\ T_2 \\ T_3 \end{Bmatrix} = \begin{Bmatrix} 2\lambda \left( \frac{\partial u_1}{\partial x_2} T_2 + \frac{\partial u_1}{\partial x_3} T_3 \right) \\ \mu_p \frac{\partial u_1}{\partial x_2} \\ \mu_p \frac{\partial u_1}{\partial x_3} \end{Bmatrix} . \qquad \text{(A.2)}
\end{aligned}$$

## A.2. Listing of Keywords for the *xns.in*-File

### General

| Example | Description |
|---|---|
| `ndf` $10$ | Degress of freedom |
| `tup` | Viscoelasticity is switched on |
| `nurbsfit` | Compute with NURBS approximation |
| `tau_constitutive source` | Stabilization parameter $\tau_{\text{CONS}}$ |
| `tau_constitutive_factor` $1.0$ | Factor of $\tau_{\text{CONS}}$ |
| `material` $1$ `viscosity` $180000$ | Polymer viscosity $\mu_p$ |
| `material` $1$ `reltime` $0.3$ | Relaxation time $\lambda$ |
| `material` $1$ `mobility` $0.6$ | Mobility factor $\alpha$ |
| `material` $1$ `solvent_viscosity` $20000$ | Solvent viscosity $\mu_s$ |

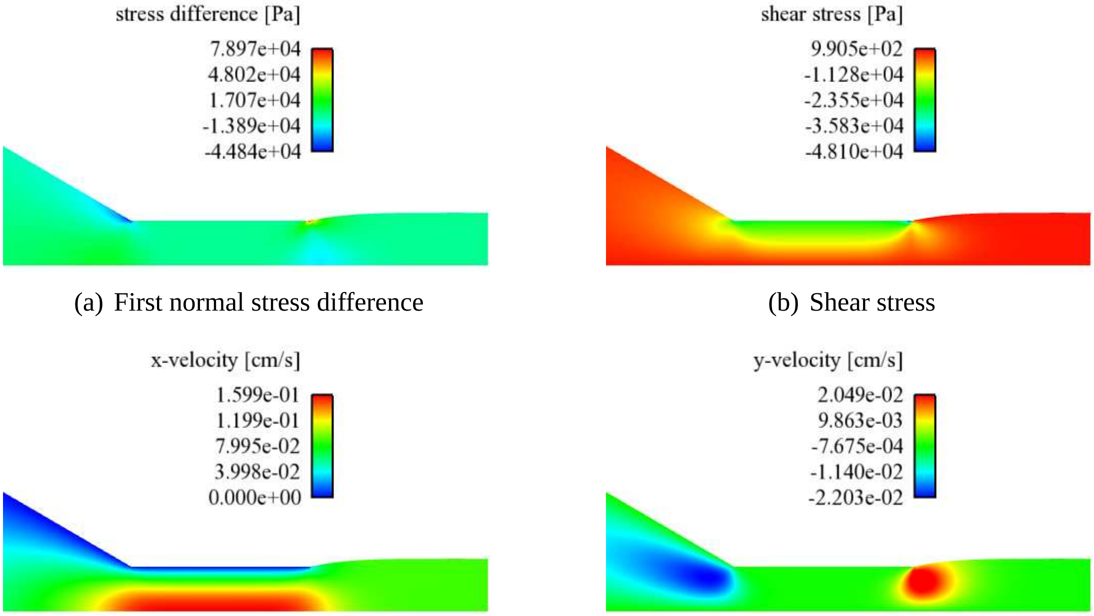
### Inflow Boundary Conditions for a Oldroyd-B Liquid

| Example | Description |
|---|---|
| `rngdset` $1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 0$ | All DOFs are prescribed, except the pressure |
| `rngdexp` $1\ 7\ \ -0.005*(y*y-4)$ | Parabolic velocity profile, for DOF 7 |
| `rngdexp` $1\ 1\ \ 2*0.3*180000*(0.005*2*y)$^2 | BC for the normal stress (cf. Eq.(3.13)) |
| `rngdexp` $1\ 2\ \ -180000*0.005*2*y$ | BC for the shear stress |

### Inflow Boundary Conditions for a Giesekus Liquid

| Example | Description |
|---|---|
| `rngdset` $1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 0$ | All DOFs are prescribed, except the pressure |
| `rngdexp` $1\ 7\ \ -0.005*(y*y-4)$ | Parabolic velocity profile, for DOF 7 $(u(y,z))$ |
| `rngdexp` $1\ 1\ \ 180000$ | Polymer viscosity $\mu_p$ |
| `rngdexp` $1\ 2\ \ 0.3$ | Relaxation time $\lambda$ |
| `rngdexp` $1\ 3\ \ 0.6$ | Mobility factor $\alpha$ |
| `rngdexp` $1\ 4\ \ -0.005*2*y$ | $\frac{\partial u(y,z)}{\partial y}$ |
| `rngdexp` $1\ 5\ \ 0.0$ | $\frac{\partial u(y,z)}{\partial z}$ |
| `rngdexp` $1\ 6\ \ 1$ | Flow direction ($x \rightarrow 1$, $y \rightarrow 2$, $z \rightarrow 3$) |

## A.3. Die Swell in a Contracting Channel – Further Plots



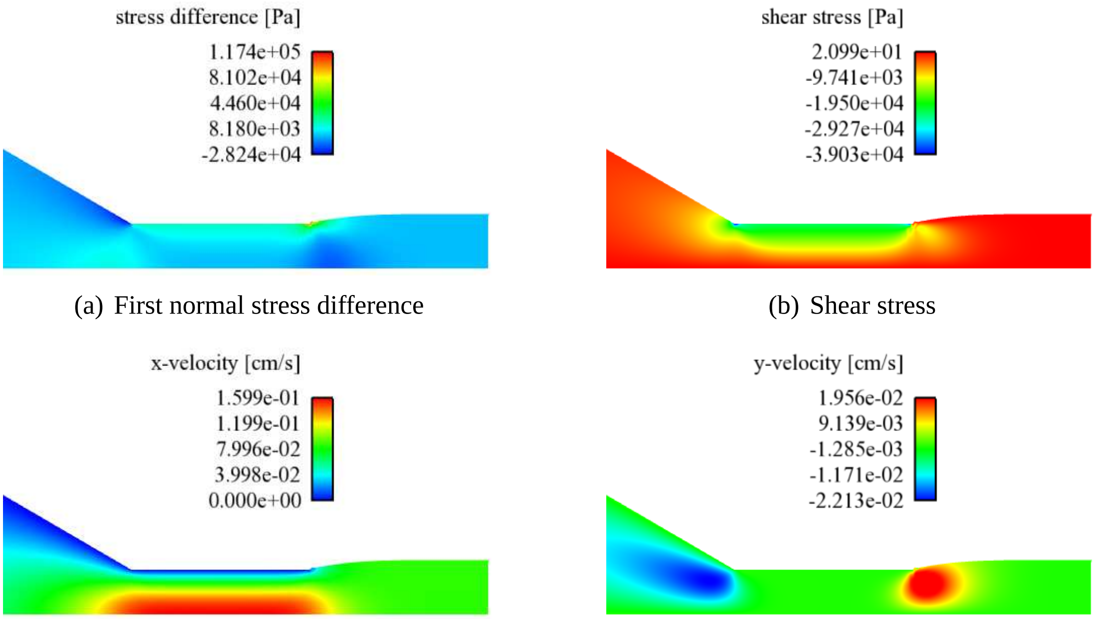
(a) First normal stress difference

(b) Shear stress

(c) Velocity in $x$-direction

(d) Velocity in $y$-direction

Figure A.1.: Results for the Newtonian model on the medium mesh.



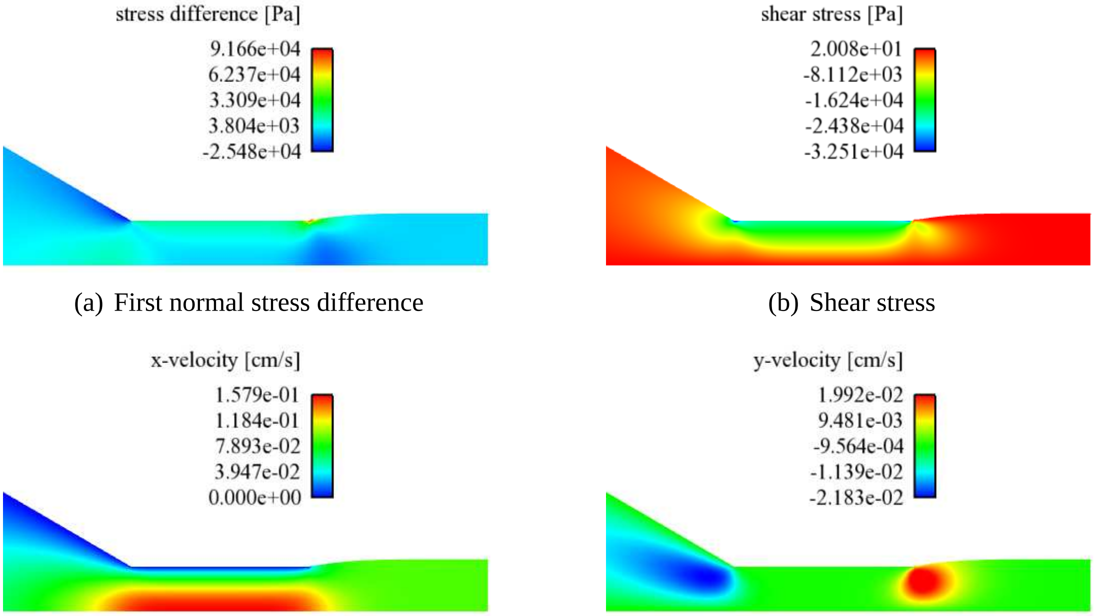
(a) First normal stress difference

(b) Shear stress

(c) Velocity in $x$-direction

(d) Velocity in $y$-direction

Figure A.2.: Results for the Oldroyd-B model on the medium mesh.



(a) First normal stress difference

(b) Shear stress

(c) Velocity in $x$-direction

(d) Velocity in $y$-direction

Figure A.3.: Results for the Giesekus model on the medium mesh.