Lattice boltzmann simulations of viscoelastic fluids in biofabrication and adjacent characterization

Von der Universität Bayreuth
zur Erlangung des Grades eines
Doktors der Naturwissenschaften (Dr. rer. nat.)
genehmigte Abhandlung

von

Richard Kellnberger

geboren in Deggendorf

1. Gutachter: Prof. Dr. Stephan Gekle
2. Gutachter: Prof. Dr. Arthur Peeters

Tag der Einreichung: 02.02.2026
Tag des Kolloquiums: 26.02.2026

In memory of the thesis I actually wrote.

Publications

During work on this thesis, two journal articles have been published (see references [1, 2]) with intermediate results. These contain early versions of some of the chapters in this thesis. The thesis is more in depth on these topics, but will not reiterate the experimental sections of the mentioned publications. Therefore, reading the publications is only necessary, if one is strongly interested in the applications of the methods discussed in the present work. The respective chapters reference the relevant publications. For one of the publications, the assets are also provided via GitHub[1]. A third publication is still in preparation at the time of writing this thesis.

[1]On the account “Richardk2n” under the title of reference [1]

Abstract

This thesis provides a rigorous introduction into many concepts important to Lattice Boltzmann simulations for viscoelastic fluids. It does in fact provide the first such simulation, that is able to simulate realistic viscoelastic fluids. Many algorithms necessary to perform the Lattice Boltzmann simulations presented in this thesis are covered explicitly. This includes, the Lattice Boltzmann algorithm, the Immersed Boundary Method and a finite volume algorithm. Validations are provided for the algorithms. Instabilities and their mitigation are explored. The shuffling algorithm necessary for viscoelastic fluids, which was developed in this thesis is presented. Derivations are given for most theoretical concepts used in this thesis. Interpretations and definitions of important quantities are explored solely driven by usability and without regard for previous definitions. An altered derivation of the Reynolds number provides a better way to speed up the simulation using Reynolds scaling. A different definition of the Capillary number allows finding the relevant parameter range more easily. Defining the Weissenberg number for each fluid model allows for better comparisons between the models. Afterwards, parameters are extracted from real data using the theoretical concepts. These are the parameters used throughout the thesis. Applications from atomic force microscopy to bioprinting needles are discussed. The simulations are used to explain and quantify the altered tank-treading frequency, that can be observed in shear-flows. A rectangular Poiseuille flow is investigated to explore the visibility of the elastic effects in different deformation metric. Lookup tables are provided for Real-Time Deformability-Cytometry (RT-DC). RT-DC is also investigated to explore a curious local minimum of the deformation, which is present in current simulations but not in experimental data. Finally, channels used for evaluation of cells at constant extensional rate are explored. According to literature, these should be hyperbolic. However, this thesis demonstrates, that they are not, and provides adequate shapes for different fluids. All tools are provided for maximum reproducibility.

Zusammenfassung

In dieser Arbeit werden viele Konzepte die für Lattice Boltzmann Simulationen von viskoelastischen Fluiden benötigt werden strikt eingeführt. In dieser Arbeit wird die erste derartige Simulation bereitgestellt, die in der Lage ist realistische viskoelastische Fluide zu simulieren. Viele Algorithmen, die zum Durchführen der Lattice Boltzmann Simulationen in dieser Arbeit benötigt werden, werden ausführlich besprochen. Dies beinhaltet, den Lattice Boltzmann Algorithmus selbst, die Immersed Boundary Methode und einen finite Volumen Algorithmus. Validierungen werden für die Algorithmen durchgeführt. Instabilitäten und deren Bekämpfung werden behandelt. Der shuffling Algorithmus, der für das Simulieren von viskoelastischen Fluiden notwendig ist und in dieser Arbeit entwickelt wurde, wird präsentiert. Herleitungen werden für die meisten theoretischen Konzepte dieser Arbeit bereitgestellt. Interpretationen und Definitionen wichtiger Größen werden ausschließlich an der Nützlichkeit ausgerichtet, ohne Rücksicht auf frühere Definitionen zu nehmen. Eine veränderte Herleitung der Reynolds Zahl ermöglicht einen besseren Weg die Simulationen mittels Reynolds Skalierung zu beschleunigen. Eine andere Definition der Kapillarzahl ermöglicht ein einfacheres Auffinden des relevanten Parameterbereichs. Die Weissenberg Zahl für jedes Fluidmodell individuell zu definieren erlaubt einfachere Vergleiche zwischen den Modellen. Später werden Parameter mittels der theoretischen Konzepte aus realen Daten extrahiert. Dies sind die Parameter, welche in der gesamten Arbeit verwendet werden. Anwendung vom Rasterkraftmikroskop bis hin zu Bio-Druck Nadeln werden diskutiert. Simulationen werden verwendet, um die veränderte Rotationsfrequenz von Zellen im Scherfluss zu beschreiben und zu quantifizieren. Ein rechteckiger Poiseuillefluss wird diskutiert, um die Sichtbarkeit elastischer Effekte in unterschiedlichen Deformationsmetriken zu zeigen. Lookup-Tabellen werden für Real-Time Deformability-Cytometry zur Verfügung gestellt. Dieses Experiment wird ebenfalls untersucht, um ein lokales Minimum der Deformation zu erklären, welches zwar in aktuellen Simulationen, nicht aber im Experiment sichtbar ist. Letztlich werden Kanäle mit konstanter Dehnrate untersucht. Diese sind laut der Literatur hyperbolisch. Diese Arbeit demonstriert hingegen, dass dies nicht der Fall ist und stellt adäquate Formen für unterschiedliche Fluide zur Verfügung. Alle verwendeten Programme werden im Namen der maximalen Reproduzierbarkeit bereitgestellt.

Contents

Reading this thesis

This thesis is split into three parts. The methods, the applications and the appendices. A reader familiar with the subjects at hand might be tempted to skip the methods. This is not recommended, as they contain many important definitions, which are not straight forwards and different from how they are typically defined in literature. Also, this part introduces core concepts and simulation techniques. The applications contain the actual use of the methods for typical problems or problems from literature. Both the methods and the applications are designed to be more of a light read. Derivations, further explanations and deeper insight are placed in the appendix. The interested reader should read every appendix, when it is first mentioned. Any subsequent mentions are just to refresh the reader's memory. There are no appendices, that are only linked from other appendices. Reading the methods and applications, following every link to the appendix and returning afterwards, covers the entire appendix. Throughout the thesis, one can find setup boxes. A dummy for such a box can be seen in the following (simulation 1).

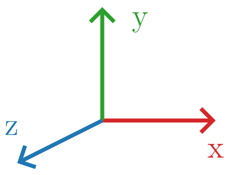
1 Example setup

Box:	$L_x \times L_y \times L_z$
	$L_0 \approx 1\,\mu\text{m}$
Fluid:	Newtonian::water (fluid 1)
BC:	velocity (BC 2)
	$\dot{\gamma} = 1 \times 10^4\,\frac{1}{\text{s}}$

It provides a convenient place, where all important parameters of any given simulation meet.

Terminology and notation

Abbreviations are generally introduced before they are used in this thesis. However, some are incredibly common and warrant an early mention. The two core algorithms of this thesis are the Lattice Boltzmann Method (LBM) and the Immersed Boundary Method (IBM). Their formal introduction will follow later, however they will be referenced a lot already beforehand. It is also necessary to know a few parts of a computer. The simulations in this thesis are run on graphics cards (GPUs). These have a very fast connection to their integrated memory (VRAM). They are connected via the orders of magnitude slower PCIe bus to the central Processor (CPU). A consumer CPU can work on around 16 to 32 work items simultaneously. It has 16 to 32 threads. A GPU can do 10 000. The illustrations in this thesis are all oriented the same way in relation to the coordinate axis. This is done for simplicity. Consequently, the coordinate system is typically omitted. It is always laid out like the following.



It is important to remember the abbreviations and coordinate axis introduced here.

Introduction

Biofabrication is a currently fast-growing field aiming at the manufacturing of tissues for various purposes. This includes the eye-catching use-case of printing replacement organs, but also more realistic topics like the manufacturing of tissues for drug testing. There are multiple manufacturing methods used in the field. One of the more versatile options is microfluidics, which is used to advance the dream or organ printing. Microfluidics refers to experiments using fluid channels on chips on the micrometer scale, typically built using lithography, that achieve a particular purpose. It has been used for manufacturing beforehand and is well-developed. In biofabrication, microfluidic methods are used to deposit living cells and construct artificial tissues in a process similar to 3D printing [3–5]. (Biological) cells exhibit a wide range of interesting behaviors when put into microfluidic flows. This can also be used to study them. Consequently, similar microfluidic channels are used for medical diagnostics [6]. Diseases can be detected through them pathologically altering cell properties, which in turn alters their behavior in microfluidc flows. As working with biological cells has many complicating factors (for example keeping them alive), sometimes polymer-beads are used as a stand-in [2]. This thesis uses the word "cells" for both for simplicity. Due to the data availability, it however mostly refers to cell-mimetic polymer particles. In all microfluidic applications, cells are suspended in a cell carrier fluid. This fluid is often called "bio-ink" in biofabrication. These fluids need to exhibit high viscosity for both fabrication and characterization applications. This is typically archived by dissolving polymers in water. This causes the fluids to be highly viscoelastic and shear-thinning. The requirement for high viscosity differs slightly depending on the field. For medical diagnostics, the deformation, which increases with higher viscosity, is evaluated. Therefore, the viscosity needs to be high enough to make changes in the deformation due to hypothetically present diseases detectable. In biofabrication, the printed construct relies on the viscosity of the fluid to keep it stable until the cells form a network. The higher the viscosity, the more accurate the printed shape is retained. If deformations become to large, the resulting stress is not survivable for a livable cell. Consequently, the cell carrier fluids used in biofabrication need to be strongly shear-thinning. A small viscosity is desired within the printing apparatus and a large one, as soon as the bio-ink is deposited. This means, that for bio-inks, the viscosity ratio is typically very large. The zero-shear viscosity is often several orders of magnitude higher than the pure solvent viscosity [7]. For one of the fluids used in this thesis, this ratio exceeds 10 000. To understand the behavior of viscoelastic fluids in non-trivial geometries, computer simulations are needed. Many techniques exist for the simulation of viscous fluids, however the simulation of viscoelastic fluids remains a formidable numerical challenge [8]. Viscoelastic fluids are especially problematic for high viscosity ratios, which cell carrier fluids exhibit by design. The Lattice Boltzmann method (LBM) is a very popular simulation technique for Newtonian fluids [9]. It is simple and highly parallelizable. The Lattice Boltzmann method has no issues with shear-thinning fluids and has been extended to viscoelastic simulations. This has mostly been done in two dimensions [10–26]. However, simulations in three dimensions also exist [27–38]. Critically, stability suffers for strong shear-thinning fluids with high viscosity ratios. In order to combat stability problems, different approaches exist. Some of the existing techniques include an artificial diffusivity of the polymer stress [27, 31, 33, 36]. Others perform special decompositions of the stress tensor [15]. However, despite these workarounds, high viscosity ratio fluids remain inaccessible. Therefore, realistic cell-carrier fluids fall in the range of inaccessible fluids. This thesis presents a new way to handle viscoelastic fluids. The basic idea is the insight, that stability issues arise due to an imbalance between the polymer stress and the viscous stress handled by the base LBM algorithm. This is handled by redistributing the stress. The resulting algorithm removes the need for artificial diffusivity and the like. This increases the accuracy. Furthermore, the limit for the viscosity ratio to retain stability is fully removed. This enables the simulation of cell carrier fluids for the first time. There is a trend in microfluidics to attribute any unexplained phenomenon to the elastic stress components of the viscoelastic fluid. Armed with the ability to actually simulate the experiments, this thesis provides insight in the relevance of the elastic normal stresses for different scenarios.

Part I.
Methods

1. Basics of fluid dynamics

This thesis explores complicated fluid dynamics simulations of cells (see section 6) in viscoelastic fluids (see section 3.7). This strives far from water flowing in a simple channel. For this reason a solid framework is essential. This section provides a rigorous mathematical basis for the core concepts of dealing with the Navier-Stokes equation. As literature tends to be less strict with its definitions, they might not fully line up with the ones made in this thesis. Therefore, it is recommended, even for experienced readers, to refer to the definitions provided here, rather than to memory, when considering quantities.

1.1. Navier-Stokes equation

The Navier-Stokes equation is the core concept of all fluid dynamics. It describes the time evolution of complex flow fields, considering any external forces with a rather simple form[2]. The Navier-Stokes equation can be written in a few different forms, depending on the use-case and assumptions. In the following, the convective form of the incompressible Navier-Stokes equation is given as equation (1). Any mention of the Navier-Stokes equation throughout this thesis refers to this form. The additional effects, that will be discussed in the present work are expressed through including additional terms in this baseline version of the equation.

$$\rho\frac{\partial\vec{u}}{\partial t} + \rho(\vec{u}\cdot\nabla)\vec{u} = \mu\nabla^2\vec{u} - \nabla p \tag{1}$$

Here ρ is the density, $\vec{u}$ is the velocity, t is the time, μ is the dynamic viscosity and p is the pressure. The Navier-Stokes equation is vastly more commonly seen in literature using the kinematic viscosity ν instead of the dynamic viscosity. The present choice is intentional and will be important later (see section 16). It is critical to avoid the use of ν as this would only introduce significant problems. The terms of the Navier-Stokes equation are called (in order): (time) derivative term, convective term (or non-linearity), dampening term and source term. These terms will be refereed to by these names later. According to the Buckingham π theorem [39–41], the physics of any equation can be fully captured by a few dimensionless numbers. This is done by making the given equation - the Navier-Stokes equation in this case - non-dimensional. What remains are a few new dimensionless factors containing all the physics of the problem. This means, that all parameters leading to the same dimensionless numbers produce the same solution for the given equation. The most important dimensionless number for fluid dynamics is the Reynolds number Re, which shall be discussed and most importantly rigorously defined in the next subsection.

[2]Technically it describes a momentum balance.

1.2. Reynolds number

The Reynolds number Re describes the ratio of inertial and viscous forces. It arises from the dimensionless Navier-Stokes equation. In the following the derivation and mathematical considerations about limits of the Reynolds number will be shown. These lay the foundation for Reynolds-Scaling, discussed in section 16. Afterwards a short description is provided how the Reynolds number is typically calculated in this thesis.

1.2.1. Derivation of the Reynolds number

The Reynolds number arises when one makes the Navier-Stokes equation (eq. (1)) non-dimensional. To remove the dimensions a typical density ρ_t, a typical velocity u_t, a typical viscosity μ_t, a typical length L_t and a typical time T_t are introduced. With these the conversions to non-dimensional numbers, indicated by primes, are specified as follows.

$$\rho = \rho_\mathrm{t}\rho' \tag{2}$$
$$\vec{u} = u_\mathrm{t}\vec{u}' \tag{3}$$
$$\mu = \mu_\mathrm{t}\mu' \tag{4}$$
$$\nabla = \frac{1}{L_\mathrm{t}}\nabla' \tag{5}$$
$$\frac{\partial}{\partial t} = \frac{1}{T_\mathrm{t}}\frac{\partial}{\partial t'} \tag{6}$$
$$p = \frac{\rho_\mathrm{t} u_\mathrm{t} L_\mathrm{t}}{T_\mathrm{t}} p' \tag{7}$$

It shall be noted, that in literature, the subscript for these is typically 0 instead of the t used here. This conflicts with the notation for the lattice conversion factors discussed later (see section 2.3) and is therefore avoided here. While the first four of these conversion are rather straight forward, the latter two warrant more explanation. In literature, the pressure is typically either scaled using a prefactor proportional to u_t^2 (velocity scale) [9] or proportional to $u_\mathrm{t}\mu_\mathrm{t}$ (viscosity scale) [42, 43]. The peculiar choice of a prefactor proportional to $\frac{u_\mathrm{t}}{T_\mathrm{t}}$, as is used here, stems from interpretation of ∇p as a source term. During a step of typical time T_t it creates a typical amount of velocity u_t, thus justifying the use of these factors. Consequently, the time derivative is scaled with the same typical time. In literature, this is typically done with a so-called external timescale provided by some additional effect (e.g. periodic excitation). However, this depends on whether the author in question intends to produce a Reynolds number, a Stokes number or both. In contrast, here, the typical timescale is argued to be always present and will be picked to be the smallest timescale in the system. As any present effect influences the velocity (according to its timescale), the velocity changes with any present timescale. Picking the largest of the timescales underestimates the importance of the time derivative term. Picking the smallest one overestimates it. Therefore, not picking the smallest one could hide important effects (see section 16.6 for the influence of the time derivative term). For now, it shall be kept as a variable in the derivation. Inserting the conversions listed above into the Navier-Stokes equation results in the following.

$$\rho'\frac{\partial\vec{u}'}{\partial t'} + T_\mathrm{t}\frac{u_\mathrm{t}}{L_\mathrm{t}}\rho'(\vec{u}'\cdot\nabla')\vec{u}' = \frac{\mu_\mathrm{t}T_\mathrm{t}}{\rho_\mathrm{t}L_\mathrm{t}^2}\mu'\nabla'^2\vec{u}' - \nabla'p' \tag{8}$$

This system contains two timescales. The advective timescale T_a and the viscous timescale T_v, defined as follows.

$$T_\mathrm{a} = \frac{L_\mathrm{t}}{u_\mathrm{t}} \tag{9}$$

$$T_\mathrm{v} = \frac{\rho_\mathrm{t}L_\mathrm{t}^2}{\mu_\mathrm{t}} \tag{10}$$

With these, the Reynolds number, which is typically defined as follows [9], reduces to a ratio of timescales as follows.

$$Re = \frac{\rho_\mathrm{t}u_\mathrm{t}L_\mathrm{t}}{\mu_\mathrm{t}} = \frac{T_\mathrm{v}}{T_\mathrm{a}} \tag{11}$$

It shall be noted, that many other dimensionless numbers can also be expressed as ratios of timescales (e.g. see sections 2.7 and 3.4). With these timescales and Re, the Navier-Stokes equation reduces to the following.

$$\rho'\frac{\partial\vec{u}'}{\partial t'} + Re\frac{T_\mathrm{t}}{T_\mathrm{v}}\rho'(\vec{u}'\cdot\nabla')\vec{u}' = \frac{T_\mathrm{t}}{T_\mathrm{v}}\mu'\nabla'^2\vec{u}' - \nabla'p' \tag{12}$$

This can further be simplified by introducing the Stokes number St as follows.

$$St = \frac{T_\mathrm{v}}{T_\mathrm{t}} \tag{13}$$

$$\rho'\frac{\partial\vec{u}'}{\partial t'} + \frac{Re}{St}\rho'(\vec{u}'\cdot\nabla')\vec{u}' = \frac{1}{St}\mu'\nabla'^2\vec{u}' - \nabla'p' \tag{14}$$

This equation now has the Stokes number at an unconventional place, but is very versatile due to that. As discussed above, T_t shall be the shorter of the present timescales to capture all important effects. This separates the two cases $Re \leq 1$ and $Re \geq 1$, depending on which timescale is the smaller one. With this the Stokes number is also separated into two regimes as follows.

$$St = \begin{cases} 1 & \text{, for } Re \leq 1 \\ Re & \text{, for } Re \geq 1 \end{cases} \tag{15}$$

And with this, the Navier-Stokes equation can be written as follows.

$$\rho'\frac{\partial\vec{u}'}{\partial t'} + Re\rho'(\vec{u}'\cdot\nabla')\vec{u}' = \mu'\nabla'^2\vec{u}' - \nabla'p', \quad \text{for } Re \leq 1 \tag{16}$$

$$\rho'\frac{\partial\vec{u}'}{\partial t'} + \rho'(\vec{u}'\cdot\nabla')\vec{u}' = \frac{1}{Re}\mu'\nabla'^2\vec{u}' - \nabla'p', \quad \text{for } Re \geq 1 \tag{17}$$

It shall be noted, that the equations are equal for $Re = 1$. They produce the non-stationary (or stationary in steady-state) Stokes equation [44] for $Re \to 0$ by removing the convective term. The Euler equation [45] is produced for $Re \to \infty$ by removing the dampening term. Demonstrating, that this way of making the Navier-Stokes equation dimensionless reproduces the typical approximations for different Reynolds number regimes with a unified scheme. These approximations do not only apply in the limit, but also for $Re \ll 1$ and $Re \gg 1$ irrespective of the concrete value of the Reynolds number as the respective terms still vanish. Therefore, the Reynolds number can be changed without altering the flow. This is the basic idea of Reynolds-Scaling. How this is beneficial to simulation runtime will be discussed in section 16. For the microfluidics discussed in this thesis, typically $Re \ll 1$ holds. The concrete value of the Reynolds number depends on ones choice of the "typical" values, which are not strictly defined. The definitions used in this thesis are listed in the following.

1.2.2. Calculating the Reynolds number

As the typical value of a quantity is not strictly defined, especially if it is not constant across the totality of the simulation a choice needs to be made what to pick. In this thesis the typical density is the density of the fluid, which is roughly constant across the simulation. The typical length is the diameter of the widest part of the channel if no cell is present and the radius of the cell if a cell is present. The typical velocity is the maximum velocity present in the system. When the maximum velocity cannot be determined before the simulation as is the case for complex geometries or complex fluids, it is estimated and corrected with the real value afterwards. For pure shear-flows with a cell, the typical velocity is replaced by the typical shear-rate using the typical length. For a pure shear-flow without a cell this thesis does not provide Reynolds numbers. If the viscosity is not constant it is estimated using an average shear rate from the estimated maximum velocity. The Reynolds numbers is calculated like this if not otherwise mentioned. It shall be noted, that among the valid choices for typical values this leads to a comparatively large Reynolds number. This allows a somewhat looser interpretation of the $Re \ll 1$ condition. This will be discussed further in section 16. While the Reynolds number is arguably the most important dimensionless number, it is by far not the only one. Others will be introduced throughout the thesis. Next, the methods used to solve the Navier-Stokes equation using a computer will be discussed.

2. Lattice-Boltzmann methods

The Lattice-Boltzmann method (LBM) is a standard technique used to simulate fluid flows. It forms the backbone of this thesis and all other algorithms mentioned here tie back into it. The book by Krüger *et al.* [9] provides an excellent overview on the topic and can be used as a starting point for a working implementation. The simulation software developed[3] as part of this thesis (*FluidX3D*, see appendices A and A.1 for details) is originally based on the book (see appendix A.1.1). However, through development of the expanded algorithms of this thesis - in particular the tetrahedral Immersed Boundary Method (IBM; Described in section 6) and viscoelastic fluids (see section 4) - some deviations were required. This is both for stability and because some algorithms - e.g. boundary conditions (see section 13) - could be implemented more accurately due to architectural differences. This opportunity was of course sized. This thesis aims to mention all relevant implementation details here and in the following sections describing the various algorithms used in the present work. It refers to *FluidX3D* source regularly. Also, the full *FluidX3D* source code is available on GitHub (see appendix A) or upon inquiry. This is to allow for easy reproduction of the results presented in this thesis. In the following, the very basics of LBM are mentioned first to introduce the required nomenclature, before expanding to some of the conventions used in the present work. The discussion on the basics of LBM is limited as it is a standard method. For further reading consider Krüger *et al.* [9], but be aware of slight implementation differences.

2.1. Basics of LBM

The equations in this subsection and some of their accompanying text have already been published [1]. Instead of solving the Navier-Stokes equation, LBM solves the Boltzmann equation for fluid particle distributions. This view is somewhat between a continues model of the fluid and a particle model. It assumes groups of fluid, that are large enough individually for a continues description, which behave towards each other like particles. These packets of fluids called populations will shortly be discussed. First, the Boltzmann equation needs to be discretized in several different ways. Like for every simulation algorithm, the time t is discretized into steps of size Δt. Furthermore, the space is discretized into a grid (the eponymous lattice) with a spacing of Δx. Any position $\vec{x}$ can only exist on this grid. From this follows, that velocities must be discrete as well. Within a time-step a particle currently located at one lattice point must reach another. In this only neighbors are considered. There are however multiple choices for these neighborhoods. The discrete velocities $\vec{c}_i$ belonging to a neighborhood are termed “velocity sets”. They are typically denoted as DdQq, with d being the number of dimensions and q being the number of discrete velocities. The velocity sets contain a velocity to remain at the current lattice point and one to travel to any neighbor in a symmetric neighborhood. There are multiple options for 2D and 3D. 2D sets are neither

[3]The software is older, but was massively reworked as part of this thesis. See appendix A.1.1 for details.

officially supported by *FluidX3D*, nor will they be discussed in this thesis[4]. The 3D velocity sets *FluidX3D* officially supports are D3Q15, D3Q19 and D3Q27. These can be seen in figure 1.

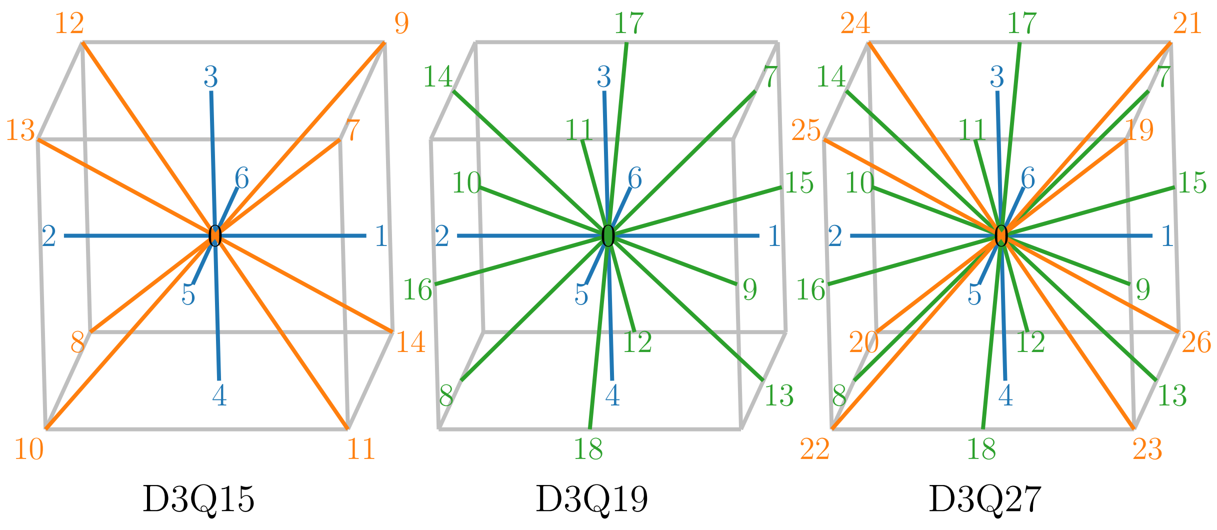


Figure 1: Illustration of the velocity sets implemented in *FluidX3D*. Each line represents the velocity to a neighbor. The 0 velocity remains stationary. The indices are in principle arbitrary, but must be held consistent throughout.

There are smaller velocity sets (D3Q7 and D3Q13), which are too inaccurate to be recommend[4]. Larger ones can theoretically exist, but are costly to compute with little benefit over the ones listed here. To accommodate the velocity sets, the density is split into the aforementioned populations $f_i(\vec{x}, t)$. These represent the fluid packets, which at time t are located on the lattice point $\vec{x}$ and are moving with the velocity $\vec{c}_i$. With these discretizations the Boltzmann equation can be written as follows.

$$f_i(\vec{x} + \vec{c}_i \Delta t, t + \Delta t) = f_i(\vec{x}, t) + \Omega_i(\vec{x}, t) \tag{18}$$

Here $\Omega_i(\vec{x}, t)$ is the collision operator, which redistributes the density among the populations, that meet at a lattice point. This equation describes two basic effects: The movement of the populations along the discrete velocities to a neighboring point (typically called "streaming"). And the exchange of momentum due to the interaction of the populations (called "collision"). These are relevant to the discussion of the concrete implementation (see appendix C). As redistributing the populations is very complex, so is the collision operator - in theory. To avoid this it is approximated using the Bhatnagar-Gross-Krook (BGK) operator [46], which reads as follows.

$$\Omega_i = -\frac{f_i - f_i^{\text{eq}}}{\tau_{\text{r}}} \Delta t \tag{19}$$

[4]They are implemented though.

Here τ_r is the relaxation time and f_i^eq are the equilibrium populations defined as follows.

$$f_i^\mathrm{eq} = w_i \rho \left[1 + \frac{\vec{u} \cdot \vec{c}_i}{c_\mathrm{s}^2} + \frac{(\vec{u} \cdot \vec{c}_i)^2}{2 c_\mathrm{s}^4} - \frac{\vec{u} \cdot \vec{u}}{2 c_\mathrm{s}^2} \right] \tag{20}$$

Here $\vec{u}$ is the local fluid velocity, ρ is the local density, w_i are the lattice weights associated with the chosen velocity set and c_s is the lattice speed of sound defined as follows.

$$c_\mathrm{s}^2 = \frac{1}{3} \frac{\Delta x^2}{\Delta t^2} \tag{21}$$

The complete LBM equation with BGK therefore takes the form of a relaxation towards equilibrium:

$$f_i(\vec{x} + \vec{c}_i \Delta t, t + \Delta t) - f_i(\vec{x}, t) = -\frac{\Delta t}{\tau_\mathrm{r}} [f_i(\vec{x}, t) - f_i^\mathrm{eq}(\vec{x}, t)] \tag{22}$$

This is also known as the single relaxation time scheme (SRT). There are more complex options, like the following two, which can be advantageous for certain parameters - at a cost of runtime. The two relaxation time scheme (TRT), which constructs moments from the populations and relaxes them individually. And the multiple relaxation time scheme (MRT), which provides a relaxation time for each moment (One for each velocity in the set). These are implemented in *FluidX3D*, but not used in this thesis. Appendix B elaborates, why their use is not recommended. Regardless of the relaxation scheme, the populations are initialized with their equilibrium, after which equation (22) can be used to determine their time evolution. To retrieve the physical quantities from the discrete populations, the following equations are used.

$$\rho = \sum_i f_i^\mathrm{eq} = \sum_i f_i \tag{23}$$

$$\rho \vec{u} = \sum_i \vec{c}_i f_i^\mathrm{eq} = \sum_i \vec{c}_i f_i \tag{24}$$

$$\mu = \rho c_\mathrm{s}^2 \left(\tau_\mathrm{r} - \frac{\Delta t}{2} \right) \tag{25}$$

$$p = \rho c_\mathrm{s}^2 \tag{26}$$

This coupling of the pressure to the density notably does not allow setting it independently. Pressure differences can still be accurately portrayed using density differences though - assuming the LBM method is used in its stable range.

2.2. Population offset

The populations have values around $w_i \rho$. This is best seen from the equilibrium populations f_i^eq (see eq. (20)). The populations in the direction of the velocity are larger than $w_i \rho$, while the others are smaller. To reduce numeric errors, the populations are stored with an offset as follows (modified from reference [47] for dimensional accuracy).

$$f_i' = f_i - w_i \rho_0 \tag{27}$$

Here, f_i' are the populations as stored and ρ_0 is the undisturbed equilibrium density. This moves the populates as stored to around 0, making the sign a direct indicator of the alignment to the velocity. This requires an adaptation to the calculation of the density from the populations (eq. (23)) as follows.

$$\rho = \sum_i f_i = \sum_i f_i' + \rho_0 \sum_i w_i = \sum_i f_i' + \rho_0 \tag{28}$$

The lattice weights add up to unity. The equation for the velocities (eq. (24)) retains its shape, as every non-zero lattice velocity has a counterpart in the opposite direction and the same weight.

2.3. Lattice Units

A computer does not understand physical units, therefore those have to be removed. However, just dropping them would lead to - at times - minuscule numbers, that are tedious to handle. With some clever definitions, the complexity of the equations can be reduced significantly, improving simulation runtime. Therefore, the concept of lattice units is introduced. The basic idea is, that any physical unit is represented by a numerical value. These values are conveniently chosen to be approximately the inverse of the typical values of the quantities[5]. This allows some number to become equal to the integer unity. This removes these quantities from the equations. Multiples of these quantities (e.g. sizes) can now be handled as integers instead of floating point numbers. This is easier and faster. These scaled numbers are referred to as the quantities in “lattice units” (LU).

[5]These two quantities are however distinct and should never be confused (see red box)!

Lattice units vs typical values

In literature, the conversion factors used for lattice units are typically denoted using a subscript zero. The typical values used for non-dimensionalization (see section 1.2.1) are also usually denoted using a subscript zero. This leads to confusion in literature and even in lectures. And while the conversions are partially identical to the typical values, this does not hold for all typical values. Therefore, these need to be strictly differentiated. The non-dimensionalization according to the Buckingham π theorem (one scaling factor per quantity) produces a dimensionless number to house the physics. It does not provide conversion factors for the SI units. Scaling with the conversion factors (one scaling factor per unit) on the other hand removes the SI units, but does not provide a dimensionless number. This transformation is mathematically identical to scaling the whole equation. In this thesis, a subscript c is used for conversions and a subscript t for typical values to avoid ambiguity. *FluidX3D* source retains the zero subscript for conversions as it does not contain references to the typical values.

The Navier-Stokes equation without extensions only contains three SI units: kg, m and s. Consequently, three conversion factors are needed. These can be acquired by fixing three (compatible) quantities in LU. Naturally, the lattice constant Δx_{LU} and the time-step Δt_{LU} are chosen to be unity. The mass conversion factor is determined using the density.

$$\Delta x_{\text{LU}} = 1 \tag{29}$$

$$\Delta t_{\text{LU}} = 1 \tag{30}$$

$$\rho_{\text{LU}} = 1 \tag{31}$$

These choices have the side effect of simplifying the LBM equations. The SI value of the lattice constant is determined by the resolution and the desired geometry. For example, if a channel of diameter d_{SI} shall be depicted using a resolution of d_{LU} lattice nodes, the length conversion factor L_{c} reads as follows.

$$L_{\text{c}} = \frac{d_{\text{SI}}}{d_{\text{LU}}} \tag{32}$$

The SI value of the density is determined by the fluid used, meaning the conversion factor for densities ρ_{c} trivially becomes as follows.

$$\rho_{\text{c}} = \rho_{\text{SI}} \tag{33}$$

Similarly, the time conversion could be defined as follows.

$$T_{\text{c}} = \frac{\Delta t_{\text{SI}}}{\Delta t_{\text{LU}}} = \Delta t_{\text{SI}} \tag{34}$$

However, this cannot yet be used for practical applications as the size of the times-step in SI is still unknown. This can be determined using the relaxation time τ_{r}, which can be chosen freely within a certain range. Its ideal value in LU is unity [9], which also simplifies the equations. This is known as instant relaxation. Setting the relaxation time differently will be discussed in more detail later (see 17). With the relaxation time set (to unity), equation (25) gives the following relation.

$$\tau_{\mathrm{r,\ LU}} = 1 \tag{35}$$

$$\mu_{\mathrm{LU}} = \frac{1}{6} \tag{36}$$

As the SI value of the viscosity is known from the fluid, this is the equation actually used to fix the conversion as follows.

$$T_{\mathrm{c}} = \frac{\rho_{\mathrm{c}} L_{\mathrm{c}}^2}{\mu_{\mathrm{c}}} = \rho_{\mathrm{c}} L_{\mathrm{c}}^2 \frac{\mu_{\mathrm{LU}}}{\mu_{\mathrm{SI}}} = \frac{\rho_{\mathrm{c}} L_{\mathrm{c}}^2}{6\mu_{\mathrm{SI}}} \tag{37}$$

This is referred to as the "viscous timescale". A time conversion factor can also be found using the velocity ("velocity scale"). The viscous scale is the one to pick for the simulations in this thesis. Depending on the parameters used the velocity scale ought to be picked. This is to assure, that the typical time is the smallest timescale in the simulation (see section 1.2.1). Using the three base conversions presented here, all other units used in this thesis can be converted to LU. All equations are valid both in SI and in LU. Often one of these is significantly easier to calculate.

2.4. Lattice coordinates

FluidX3D knows two coordinate systems. Aside from specifying the geometry, where lattice coordinates are needed[6], the user is only confronted with the IBM coordinate system (see section 6.2). The LBM algorithm itself, boundary conditions (see section 13) and some part of the IBM algorithm (see section 6) require the use of a coordinate system in tune with the lattice. The lattice coordinates are Cartesian coordinates designed to refer to the position of the lattice nodes in LU. Therefore, the nodes have integer coordinates. The origin is placed at a corner node. The coordinates therefore run from $(0, 0, 0)$ to $(L_x - 1, L_y - 1, L_z - 1)$. With L_x, L_y and L_z being the number of LBM nodes along the respective axis. The size of the simulation volume typically gets denoted as $L_x \times L_y \times L_z$. These coordinates facilitate easy array access. On the GPU side, the arrays are flattened[7] for speed. For that reason, nodes also are assigned a running index n as follows.

$$n = x + (y + zL_y)L_x \tag{38}$$

Here x, y and z are the lattice coordinates of the respective node. Throughout this thesis a lattice node is viewed as representing a $1 \times 1 \times 1$ (in LU) cube centered around its location. This view is advantages for boundary conditions (see section 13) and the associated finite volume algorithm (see section 4). Therefore, in this coordinate system,

[6]Removing this requirement is in the works.
[7]Made to have one running index instead of three.

the simulation volume technically extends from $\left(-\frac{1}{2}, -\frac{1}{2}, -\frac{1}{2}\right)$ to $\left(L_x - \frac{1}{2}, L_y - \frac{1}{2}, L_z - \frac{1}{2}\right)$. Throughout this thesis, L_x, L_y and L_z are typically chosen to be some nice round number plus two. This is to accommodate the boundaries. The size of the structure of interest is the round number.

2.5. Additional contributions

The discussion so far was only in regard to simple Newtonian fluids without any additional effects. In this thesis however, external forces, more complex fluids (see section 3.7) and flows containing cells (see section 6) are discussed. These additional contributions boil down to the inclusion of either an additional forcing term or an additional stress term. Those will be discussed in the following.

2.5.1. Force contributions

Adding an (external) force changes the Navier-Stokes equation (eq. 1) as follows.

$$\rho\frac{\partial \vec{u}}{\partial t} + \rho(\vec{u}\cdot\nabla)\vec{u} = \mu\nabla^2\vec{u} - \nabla p + \vec{f} \tag{39}$$

Where $\vec{f}$ is the new force density. This is a force per volume, that can be used for any body-force (e.g. a pressure gradient). Literature offers many different options to include additional forcing terms. *FluidX3D* uses the algorithm presented by Guo *et al.* [48]. This algorithm is termed Guo forcing. It extends the LBM equation (eq. (22)) by an additional term.

$$f_i(\vec{x} + \vec{c}_i\Delta t, t + \Delta t) - f_i(\vec{x}, t) = -\frac{\Delta t}{\tau_\mathrm{r}}[f_i(\vec{x}, t) - f_i^\mathrm{eq}(\vec{x}, t)] + \Delta t F_i(\vec{x}, t) \tag{40}$$

The forcing term $F_i(\vec{x}, t)$ is defined as follows.

$$F_i(\vec{x}, t) = \left(1 - \frac{1}{2\tau_\mathrm{r}}\right) w_i \left[\frac{\vec{c}_i - \vec{u}(\vec{x}, t)}{c_\mathrm{s}^2} + \frac{\vec{c}_i \cdot \vec{u}(\vec{x}, t)}{c_\mathrm{s}^4}\vec{c}_i\right] \cdot \vec{f}(\vec{x}, t) \tag{41}$$

Note, that the force density $\vec{f}$ gets evaluated at the lattice node locations. This is important later (see section 6). Also, the velocity needs to be corrected and equation (24) becomes the following.

$$\rho\vec{u}(\vec{x}, t) = \sum_i \vec{c}_i f_i(\vec{x}, t) + \frac{\Delta t}{2}\vec{f}(\vec{x}, t - \Delta t) \tag{42}$$

The original publication suppresses the arguments in this extension. While most of those are clear, the time argument of the force in the last equation is open to interpretation. It has been found, that not updating the force before recalculating the velocity is crucial to stability, when used in conjunction with IBM (see section 15.1). The more precise criterion is probably, that the velocity and the force used to correct it must be spaced an uneven amount of time-steps apart. Both using the force from the same time-step and using one two time-steps old lead to considerable instability. This concrete choice made here is however not necessary, Krüger for example does not update the velocity after streaming and rather corrects an old one. This means, in the implementation used by his group, the force is actually more up to date than the velocity. They found this to work if velocity and force are separated by one time-step as Krüger confirmed via email. *ESPResSo* [49, 50] also has a separation of one, but with both the velocities and the force used one time-step older than the ones used by *FluidX3D*. Together, this shows, that this implementation detail matters. It is not essential to be identical to the choice made in this thesis, but the separation of one time-steps is necessary.

2.5.2. Stress contributions

Aside from forces, some extensions like viscoelastic fluids (see section 3.7) provide additional stresses. As is the case for a force, an additional (external) stress can also be included, by adding a term to the Navier-Stokes equation (eq. 1) as follows.

$$\rho\frac{\partial\vec{u}}{\partial t}+\rho(\vec{u}\cdot\nabla)\vec{u}=\mu\nabla^2\vec{u}-\nabla p+\nabla\cdot\underline{\tau}' \tag{43}$$

Where $\underline{\tau}'$ is the stress tensor to be coupled into the LBM. *FluidX3D* uses the algorithm presented by Dzanic *et al.* [15, 19, 28, 32]. This extends the LBM equation (eq. (22)) by an additional term.

$$f_i(\vec{x}+\vec{c}_i\Delta t,t+\Delta t)-f_i(\vec{x},t)=-\frac{\Delta t}{\tau_\mathrm{r}}[f_i(\vec{x},t)-f_i^\mathrm{eq}(\vec{x},t)+B_i(\vec{x},t)] \tag{44}$$

With the stress contributions $B_i(\vec{x},t)$ defined as follows.

$$B_i=w_i\left(\frac{\vec{c}_i\vec{c}_i}{2c_\mathrm{s}^4}-\frac{\mathbb{1}}{2c_\mathrm{s}^2}\right):\underline{\tau}' \tag{45}$$

The colon denotes the Frobenius inner product. It is also possible to transform the stress into a force using a numerical derivative and couple it to the LBM using Guo forcing. However, this was found during the present work to be less stable and sometimes less accurate.

2.6. Standard choices

There are some choices to be made with little to no influence on simulation results. These are kept consistent throughout the thesis and will be mentioned in the following, together with the reasoning for this pick.

2.6.1. Typical velocity sets

Larger velocity sets are more accurate. However, they require more compute, increasing runtime. And they require more VRAM, limiting the maximum size of the system that can be simulated and increasing runtime even more. During testing, it became clear that D3Q27 provides little benefit over D3Q19. D3Q15 is sometimes a little inaccurate. Therefore, all simulations in this thesis are done using D3Q19. The 19 velocities in this set can be found in equation (46) below.

$$\frac{\Delta t}{\Delta x}\vec{c}_i = \left\{\begin{matrix} 0 & 1 & -1 & 0 & 0 & 0 & 0 & 1 & -1 & 1 & -1 & 0 & 0 & 1 & -1 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 & -1 & 0 & 0 & 1 & -1 & 0 & 0 & 1 & -1 & -1 & 1 & 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 1 & -1 & 1 & -1 & 0 & 0 & -1 & 1 & -1 & 1 \end{matrix}\right\} \tag{46}$$

Where each column is a vector corresponding to an index i. The corresponding weights can be found in equation (47) below.

$$w_i = \begin{cases} \frac{1}{3} & \text{, for } i = 0 \\ \frac{1}{18} & \text{, for } 1 \leq i \leq 6 \\ \frac{1}{36} & \text{, otherwise} \end{cases} \tag{47}$$

2.6.2. Relaxation model

As was mentioned before (see appendix B), MRT is often recommended, but slightly detrimental to stability at a steep cost. TRT showed no noticeable difference during testing. Consequently, this thesis uses SRT throughout.

2.6.3. Geometry

LBM has no preferred axis and the geometries vary. However, it has proven advantageous to keep certain trends. In this thesis, when ever possible the flow is along the x-axis. The y-axis is chosen to be the second most interesting one. This means, that for example in pure shear-flow, the velocity (along the x-axis) varies along the y-axis. If the geometry varies, this variation is chosen to be along the y-axis. The geometry is picked to assure the highest amount of translational symmetry along the z-axis. Consequently, flows in this thesis are usually evaluated by cuts parallel to the xy-plane. If the flow is not evaluated along the z-axis, as is done throughout this thesis, this is due to the choice of coordinate system. This makes that axis particularly uninteresting.

2.6.4. Boundary conditions

The default boundary condition in *FluidX3D* is periodic boundaries (see section 13.1.1). If not otherwise mentioned, this thesis implies periodic boundary conditions.

2.7. Mach number

The Mach number Ma is another dimensionless number and defined as follows.

$$Ma = \frac{|\vec{u}|}{c_\mathrm{s}} \tag{48}$$

It can be seen as relating the typical timescale to the viscous timescale. It is important, because it represents a LBM stability criterion, solely related to the LBM itself. However, it is not possible to automatically fulfill it by the standard choices and conversion definitions presented above and needs manual checking. In theory the bulk fluid in LBM is stable given $Ma \leq 1$. For $\tau_\mathrm{r} = 1$ (optimal stability condition [51]) or above, as is generally the case in this thesis, even $Ma \leq 2$ is allowable. With boundary conditions and numerical errors, though, the real stability is considerably worse. For small values of τ_r ($\tau_\mathrm{r} \approx \frac{1}{2}$), the maximal allowable Mach number approaches zero. Aside from a mostly hypothetical discussion of the merits of higher values of τ_r in section 17, this thesis stays at unity. Given the parameter space this thesis operates in, the Mach number is usually rather low. However, it is still checked to assure $Ma \leq 1$, which usually is a sensible stability criterion, when picking the ideal value for τ_r. It is most easily checked in lattice units. If one does not intend to change the relaxation time, the Mach number can be easily changed via the lattice resolution.

2.8. Strain-rate tensor from populations

The strain-rate tensor $\underline{D}$ is defined as follows.

$$\underline{D} = \frac{1}{2}\left[\nabla\vec{u} + (\nabla\vec{u})^\mathrm{T}\right] \tag{49}$$

Note that, $\nabla\vec{u}$ is sloppy notation, that often gets misinterpreted for its transpose. This does not matter here, but for clarity, it is defined as follows.

$$(\nabla\vec{u})_{ij} = \nabla_i u_j = \frac{\partial u_j}{\partial x_i} \tag{50}$$

With this it is quite clear, that the strain-rate tensor could be calculated from the velocity field $\vec{u}$ using finite differences. However, there is another way. For MRT, it can be defined as follows [52].

$$D_{\alpha\beta} = -\frac{1}{2\rho c_\mathrm{s}^2} \sum_i (\vec{c}_i)_\alpha (\vec{c}_i)_\beta \sum_j \left(\underline{M}^{-1}\underline{SM}\right)_{ij} \left[f_j(\vec{x},t) - f_j^\mathrm{eq}(\vec{x},t)\right] \tag{51}$$

This expression contains, the density ρ, the lattice speed of sound c_s, the size of a time-step Δt, the lattice velocities $\vec{c}_i$, the populations f_i and the equilibrium populations f_i^eq. The matrix operator $\underline{M}^{-1}\underline{SM}$ is the MRT relaxation operator. The calculation presented here also works with SRT. MRT is presented as it is the most difficult and general case. Of course, actually calculating this operator would be expensive. Consider the LBM equation for MRT. It is defined as follows [9, 53].

$$f_i(\vec{x}+\vec{c}_i\Delta t, t+\Delta t) - f_i(\vec{x},t) = -\sum_j \big(\underline{M}^{-1}\underline{S}\underline{M}\big)_{ij}\big[f_j(\vec{x},t) - f_j^{\text{eq}}(\vec{x},t)\big]\Delta t \tag{52}$$

Some additional mathematical symbols have been added compared to the source, to make the notation clear. This relation can be used to remove the relaxation operator from the strain-rate tensor equation. The result can be seen in the following.

$$D_{\alpha\beta} = \frac{1}{2\rho c_{\text{s}}^2 \Delta t}\sum_i (\vec{c}_i)_\alpha (\vec{c}_i)_\beta [f_i(\vec{x}+\vec{c}_i\Delta t, t+\Delta t) - f_i(\vec{x},t)] \tag{53}$$

Note, that the two buffers used for LBM contain exactly these populations after a `PUSH` LBM step. This makes this easy to implement. Walls are a bit tricky, as some of the alter populations, which would cause problems in this definition. *FluidX3D* avoids this by stashing the unaltered population in the unused population registers of the walls. Aside from being easier to implement than the fine difference approach, this is also more accurate. The strain-rate tensor from the velocity field is a bit smoother though, which can help with stability of the algorithms using this tensor. *FluidX3D* has a flag to switch between the two options. Generally, the more accurate one is used. With this, the basics of simulating Newtonian fluids are discussed. In the following, more complex fluids, which require the strain-rate tensor to be calculated are presented.

3. Fluid models

Fluids can exhibit a wide range of behaviors. Consequently, a wide range of models exist to describe them. In the following only the models used in this thesis are described with their advantages and disadvantages. Note the change in notation from μ as the dynamic viscosity in section 1 to η for viscosities here. This is done to stress, that while these models do have quantities as parameters, which have the dimension of a viscosity, this is not the viscosity the system currently experiences. For some models, the concept of a viscosity even breaks down and gives way to a generalized stress. It should also be noted, that the models presented here are not limited to the fluids discussed in the present work and can be used to describe many fluids. Aside from Newtonian fluids, this thesis distinguishes two additional categories of fluids: Generalized Newtonian fluids (see subsection 3.5), which exhibit a stress, that can be expressed as a function of the shear-rate[8]. And viscoelastic fluids (see subsection 3.7), which are described by a defining equation, that can be transformed into the standard notation presented in equation (75). This means, they can be written as a time evolution of a generalized proxy quantity for the stress. The analytical solutions that exist, and their associated derivations can be found in appendix F. This section only reproduces the most important results. Aside from the models described below, which are implemented in *FluidX3D*, the software is written in a way that lends itself to easy extension. Given a fluid model fits in one of the categories mentioned above, it can be easily implemented. This process is not yet documented, but can be easily understood by searching for the name of an already implemented model in the entire code-base. Without any extension, the LBM algorithm by default simulates a Newtonian fluid, which shall be briefly described first as a baseline.

3.1. Newtonian fluids

In a Newtonian fluid, the stress $\underline{\sigma}$ is simply proportional to the strain-rate tensor $\underline{D}$, as can be seen in the following.

$$\underline{\sigma} = 2\eta_\mathrm{s}\underline{D} \tag{54}$$

This stress is purely viscous. The strain-rate tensor $\underline{D}$ is defined as follows[9].

$$\underline{D} = \frac{1}{2}\left[\nabla\vec{u} + (\nabla\vec{u})^\mathrm{T}\right] \tag{55}$$

The proportionality **constant** η_s is the dynamic viscosity of the Newtonian fluid. The LBM algorithm inherently describes Newtonian fluids. The dynamic viscosity is set using the relaxation time, which is linked to the viscosity via equation (25). This means, that a Newtonian fluid is always present in the simulations. Later (see section 18.5), a workaround will be discussed, that removes this restriction[10]. However, most models discussed in this thesis have a Newtonian component. All the fluids discussed in this

[8] *FluidX3D* allows dependency on the strain-rate tensor, which is more general.
[9] $\nabla\vec{u}$ is sloppy notation, that often gets misinterpreted for its transpose. This does not matter here.
[10] This is not physical, but required for reproduction of some results in literature.

thesis are manufactured by dissolving polymer in a solvent. It can be argued, that the Newtonian behavior of the solvent is retained for all these fluids. Hence, more complex fluids typically exhibit a Newtonian stress contribution from the solvent they are based on. Consequently, this Newtonian stress term will reappear for the other models. Therefore, the subscript s, which refers to solvent, has already been introduced here. The velocity profile for a Newtonian Poiseuille flow[11] is well known. It reads as follows (see appendix F.1.1).

$$\vec{u}(r) = \frac{G}{2^{j+1}\eta}\left(R^2 - r^2\right)\hat{e}_x \tag{56}$$

Here $G = -\frac{\partial p}{\partial x}$ is the pressure gradient, R is the radius of the pipe and $j = 0$ for 2D and unity for 3D. A typical flow profile can be seen in figure 2.

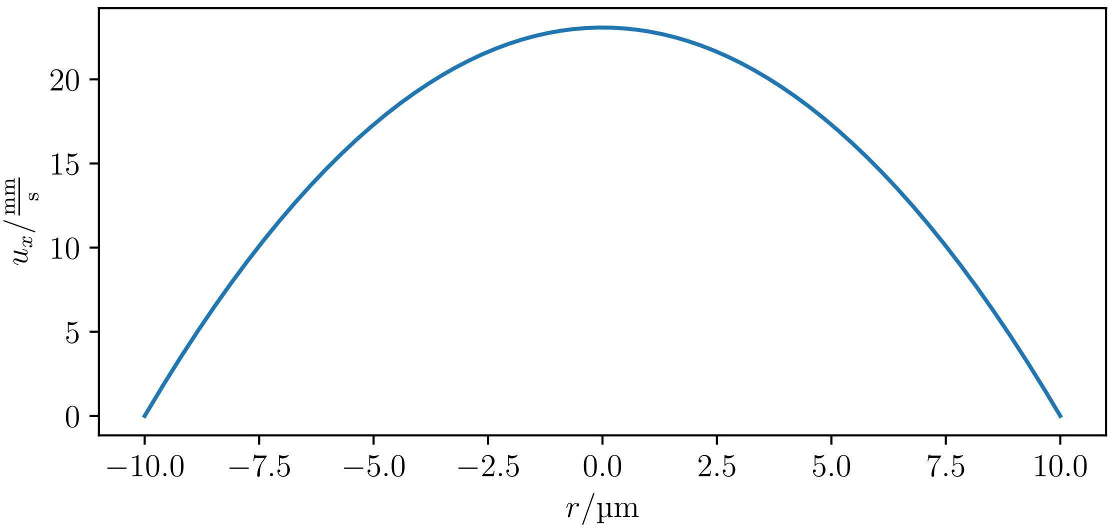


Figure 2: Example of a typical Newtonian Poiseuille flow.

The Weissenberg number Wi (see subsection 3.4) is typically not defined for Newtonian fluids, as they are not shear-thinning. However, the following definition for Wi in the Newtonian case is consistent with the other models.

$$Wi = 0 \tag{57}$$

Examples for Newtonian fluids are water, cytosol or phosphate-buffered saline (PBS). The more complex fluids share some characteristics, which is made clear in this thesis through a common notation[12], which shall be discussed next.

[11] A Poiseuille flow, strictly speaking, is defined for a Newtonian fluid. This thesis uses a broader definition (see appendix P.2)

[12] This was also used to simplify the implementation significantly.

3.2. Common variables

For more general fluids, the viscosity is not a constant. The models for these, which will be discussed in the following do have multiple parameters. They all have a notation, for any given model, that is generally agreed upon in literature. However, these parameters are not distinct from each other. The models share parameters with very similar meaning (but different typical notation). This thesis opts to use a common notation, reflecting this fact instead. The parameters shared among all the models are the solvent viscosity η_s, the polymer viscosity η_p and the (polymer) relaxation time λ. Each model has one or two additional dimensionless parameters. These models can be used to describe a shear-dependent viscosity, which is general concept, that shall be elaborated in the following.

3.3. Shear-dependent viscosity

The relationship between the stress and the strain-rate tensor can become quite complicated in general. Still, in a pure shear-flow, it is typically possible to describe a viscosity, that is dependent on the shear-rate $\dot{\gamma}$. The shear-rate can be calculated from the strain-rate tensor as follows.

$$\dot{\gamma} = \sqrt{2\underline{D} : \underline{D}} \tag{58}$$

The colon denotes the Frobenius inner product. If the viscosity increases with the shear-rate it is known as a shear-thickening fluid. An example would be a suspension of starch in water called “Oobleck”. These are well known from online videos and very popular with the general public (usually they are called “non-Newtonian fluids”[13]). However, as those properties are not desirable for biofabrication (for desirable properties, see section 19.4), they are not discussed here. If the viscosity decreases with the shear-rate the fluids are called shear-thinning. Aside from their importance in biofabrication, they are present in everyday life. Examples include toothpaste, ketchup, lotions and blood. The well known phenomenon of ketchup suddenly exiting the bottle all at once is in fact due to its shear-thinning nature. A plot of a typical shear-thinning curve can be seen in figure 3. The corresponding state the polymers in the solution are in, can be seen in figure 4.

[13]All the fluids here are non-Newtonian, the public just happens to use this term specifically for shear-thickening fluids.

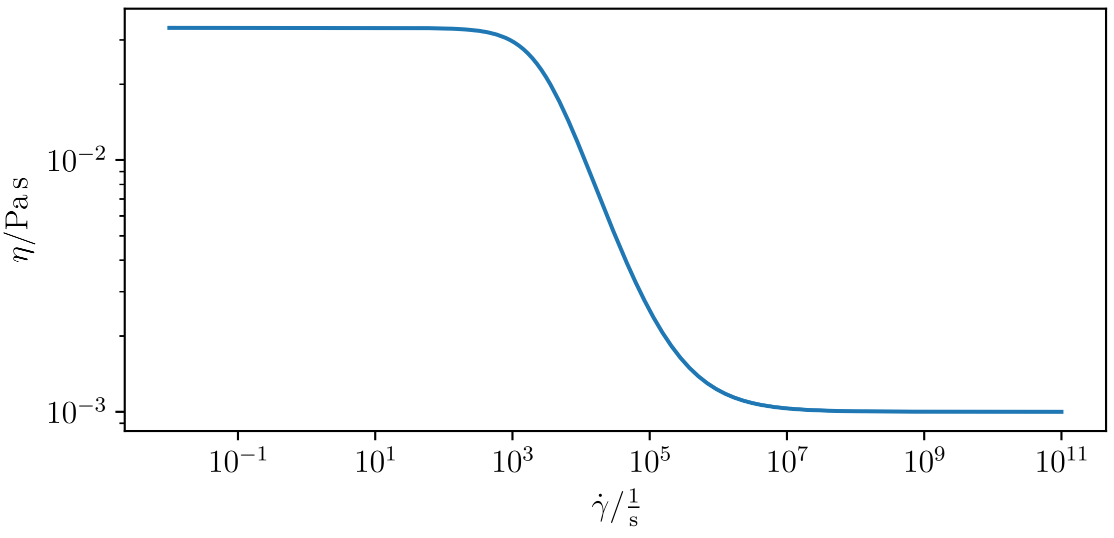


Figure 3: Example of a typical shear-thinning curve. The viscosity diminishes suddenly for a typical shear-rate.

For the polymer solutions discussed in this thesis, the explanation for the shear thinning behavior is as follows. At rest, or for low shear-rates ($\dot{\gamma} < 1 \times 10^3\,\frac{1}{\mathrm{s}}$ in figure 3), the polymers interlock (figure 4 a), leading to high viscosities (described by the polymer viscosity η_p). The viscosity experienced for negligible shear-rates is termed the fluids zero-shear viscosity η_0. If subjected to shear-flow ($\dot{\gamma} \approx 1 \times 10^4\,\frac{1}{\mathrm{s}}$ in figure 3), the polymer chains start to align (figure 4 b), reducing the interlocking and therefore decreasing the viscosity. The shear-rate at which this alignment happens is a material parameter. This is captured by the (polymer) relaxation time λ. More on the curious relationship between the necessary shear-rate for disentanglement and the time required to complete it can be found in the following. For high shear rates ($\dot{\gamma} > 1 \times 10^7\,\frac{1}{\mathrm{s}}$ in figure 3), the polymers are fully aligned (figure 4 c) and due to their typically small concentration barely contribute to the viscosity. The viscosity experienced for very high shear-rates is termed the fluids infinite-shear viscosity η_∞. This drop in viscosity (close) to the solvent viscosity at infinite shear cannot be observed due to experimental constraints and therefore has to be assumed in this thesis[14].

[14]The infinite-shear viscosity may be above the solvent viscosity, but available experimental data does provide an upper bound of at most double the solvent viscosity. Still, this does not matter in the shear-rate regimes discussed in this thesis.

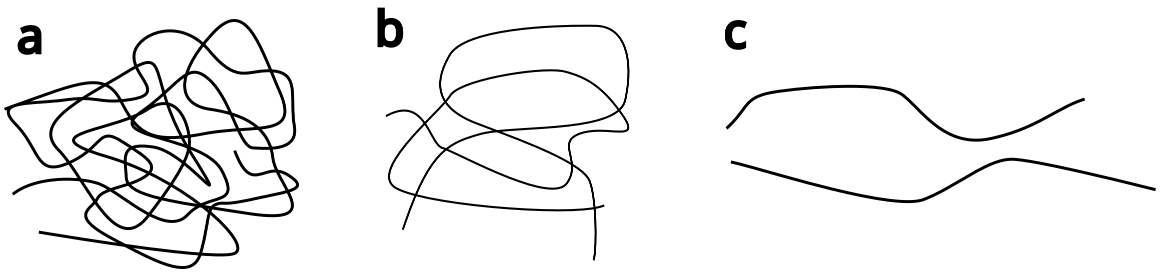


Figure 4: Illustration of the polymers contained in the solution for a) small, b) intermediate and c) large shear-rates.

All the following models are shear-thinning. The state of this shear-rate/relaxation time interplay is captured by the Weissenberg number. It can be used to describe the onset of shear-thinning and will be discussed next.

3.4. Weissenberg number

The Weissenberg number Wi is a dimensionless number often defined as follows.

$$Wi = \lambda\dot{\gamma} \tag{59}$$

It relates the relaxation time λ and the advective timescale represented by the shear-rate. Its value can be viewed as a proxy for how much viscosity has already been lost to shear thinning. $Wi = 1$ is typically viewed as the point, where shear-thinning becomes significant. This can be seen in figure 5.

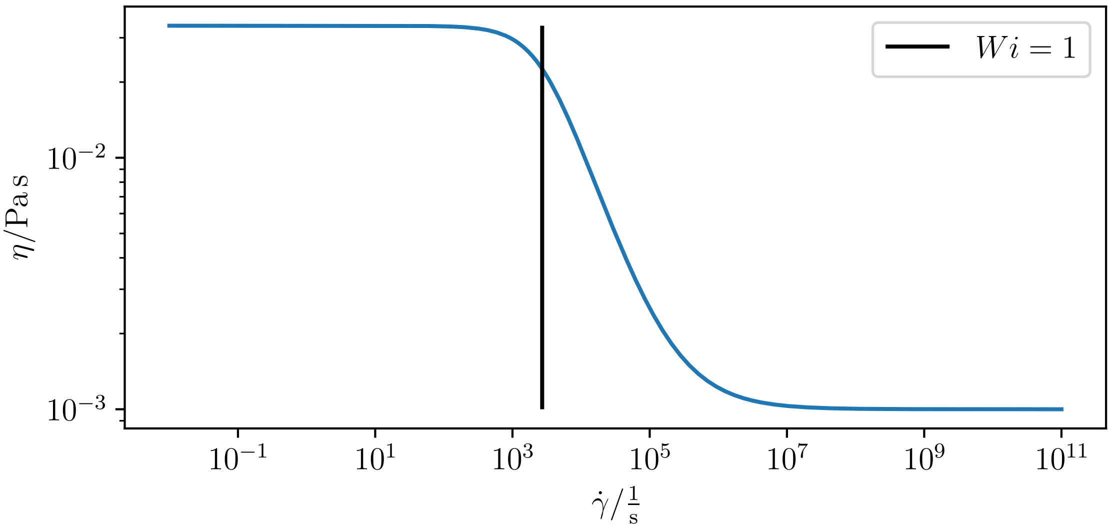


Figure 5: Example of a typical shear-thinning curve. The viscosity diminishes suddenly for a typical shear-rate. This drop is approximately marked by $Wi = 1$

In order to assure, that $Wi = 1$ is always roughly around the onset of shear-thinning, it has to be defined slightly differently for some of the fluid models. This is done by including some of the dimensionless parameters. One can see, that from the technical derivation of Wi (non dimensionalization of the constitutive equations of the models), there is a choice on the exact definition. This means, that inclusion of these non-dimensional factors is a matter of choice. This thesis opts to define the Weissenberg number slightly differently for the viscoelastic models, to ensure it retains the same meaning. For the less complicated fluids (Generalized Newtonian fluids), which will be discussed next, the definition above holds.

3.5. Generalized Newtonian fluids

For Generalized Newtonian fluids, the stress is purely viscous, and a viscosity can be given. The viscosity η depends only on the shear-rate $\dot{\gamma}$, giving the following stress.

$$\underline{\sigma} = 2\eta(\dot{\gamma})\underline{D} \tag{60}$$

Technically *FluidX3D* allows implementing any fluid described by the following equation in this category and is therefore even more general.

$$\underline{\sigma} = \underline{\sigma}(\underline{D}) \tag{61}$$

The following two possible descriptions do however, follow the simpler equation. Both are shear-thinning. It should be noted, that while real-live shear-thinning fluids can be described as Generalized Newtonian fluids, technically, they are usually viscoelastic fluids (see subsection 3.7). Ignoring the elastic components is however often useful, making these models important.

3.5.1. Carreau-Yasuda

The Carreau-Yasuda (CY) model [54, 55] is an empirical shear-thinning model. Its model equation contains the additional parameters a and p aside from the ones already mentioned. The constitutive equation reads as follows.

$$\eta(\dot{\gamma}) = \frac{\eta_\mathrm{p}}{[1+(\lambda\dot{\gamma})^a]^{\frac{p}{a}}} + \eta_\mathrm{s} \tag{62}$$

The example provided previously for shear-thinning fluids (see figure 5) is a CY curve. The CY model describes a Newtonian plateau of $\eta = \eta_\mathrm{p} + \eta_\mathrm{s}$ for $\dot{\gamma} \to 0$. Afterwards, it transitions into a power law described by the power p. The parameter a describes the smoothness of this transition. Finally, another Newtonian plateau with $\eta = \eta_\mathrm{s}$ for $\dot{\gamma} \to \infty$ is reached. The zero-shear viscosity and infinite-shear viscosity are often separated by several orders of magnitude. Through its parameters, CY was designed to describe many shear-thinning fluids. It can be fitted to all the fluids used in this thesis. In the present work, the CY model is primarily used to describe the viscous behavior of the more complicated viscoelastic models. This allows to separate viscous effects from the elastic ones. In cases, where elasticity is found to not matter, it is recommended to use the CY model, as it is significantly cheaper to calculate, than the viscoelastic models.

It should be noted, that this model expects $\dot{\gamma} > 0$, which conflicts with the definition used in the remaining thesis. The absolute value of $\dot{\gamma}$ is taken whenever necessary to accommodate for this. The velocity profile in a Poiseuille flow[15] can be obtained through some numerical steps (see appendix F.1.2). The resulting shear-thinning velocity profile can be seen in figure 6

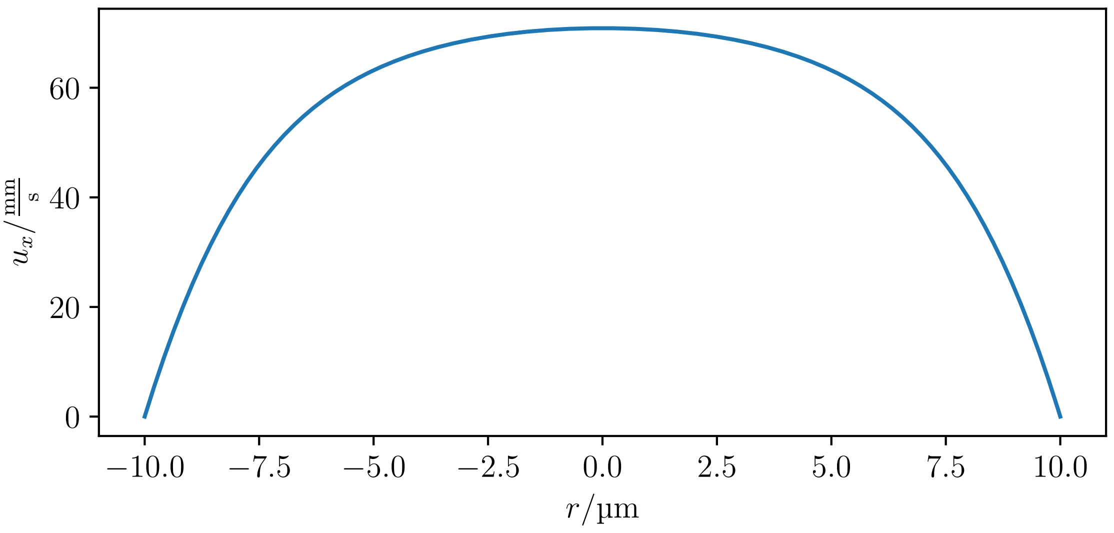


Figure 6: Example of a typical CY Poiseuille flow. The center-line velocity is considerably higher than for a Newtonian fluid with the same zero-shear viscosity (see figure 2).

Instead of the parabolic profile, that was seen for Newtonian fluids (see figure 2), this profile is flatter on top and hints at higher powers. As was previously mentioned, the Weissenberg number is defined as follows.

$$Wi = \lambda\dot{\gamma} \tag{63}$$

In literature sometimes simpler models like the Power law fluid are preferred.

3.5.2. Power Law

The power law fluid is simply described by a power law as follows.

$$\eta = \begin{Bmatrix} \eta_\mathrm{p} & \text{, for } \dot{\gamma} < \dfrac{1}{\lambda} \\ \eta_\mathrm{p}(\lambda\dot{\gamma})^{-P} & \text{, else} \end{Bmatrix} (+\eta_\mathrm{s}) \tag{64}$$

The power la fluid can be obtained from the CY model for $a \to \infty$. It requires a cutoff for small shear-rates as the viscosity would otherwise diverge. In literature, this model

[15] A Poiseuille flow, strictly speaking, is defined for a Newtonian fluid. This thesis uses a broader definition (see appendix P.2)

typically does not include η_s and in this thesis it is consequently always zero. However, it has been included here for consistency with the other models. The cutoff and λ are typically chosen as independent variables. However, there is no need to do this. Choosing the cutoff in relation to λ ensures a sensible behavior for Wi and the zero-shear viscosity. It should be noted, that this model expects $\dot{\gamma} > 0$, which conflicts with the definition used in the remaining thesis. The absolute value of $\dot{\gamma}$ is taken whenever necessary to accommodate for this. The velocity profile in a Poiseuille flow[16] reads as follows, using a few approximations (see appendix F.1.3).

$$\vec{u}(r) = \left(\frac{G\lambda^P}{\eta_\mathrm{p} 2^j}\right)^{\frac{1}{1-P}} \frac{1}{e}(R^e - r^e)\hat{e}_x \tag{65}$$

With the exponent $e = \frac{2-P}{1-P}$. Here, $G = -\frac{\partial p}{\partial x}$ is the pressure gradient, R is the radius of the pipe and $j = 0$ for 2D and unity for 3D. The resulting shear-thinning velocity profile can be seen in figure 7

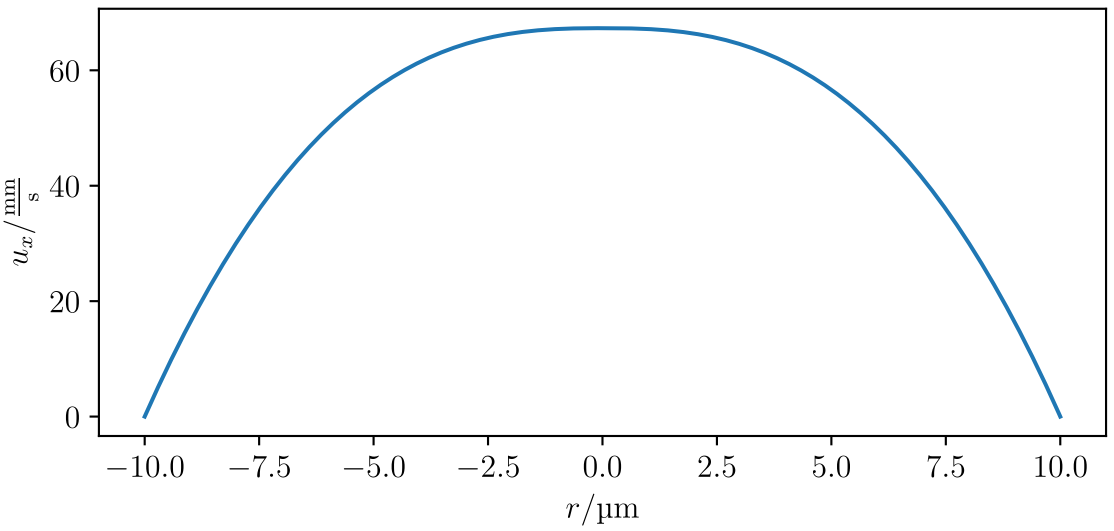


Figure 7: Example of a typical Power Law Poiseuille flow. It is rather similar to the CY equivalent (see figure 6).

As was previously mentioned, the Weissenberg number is defined as follows.

$$Wi = \lambda\dot{\gamma} \tag{66}$$

This concludes the discussion on Generalized Newtonian fluids. Next, the first normal stress difference will be discussed as a stepping stone towards viscoelastic fluids.

[16] A Poiseuille flow, strictly speaking, is defined for a Newtonian fluid. This thesis uses a broader definition (see appendix P.2)

3.6. First normal stress difference

One of the first observations of viscoelasticity in fluids, was during stirring of flamethrower fuel (polymer solution)[56]. Contrary to Newtonian fluids, which would form a depression in the center, due to the centrifugal force, these solutions climb up the stirrer. This is due to normal stresses. For a pure shear-flow, the strain-rate tensor is as follows.

$$\underline{D} = \frac{1}{2}\begin{pmatrix} 0 & \dot{\gamma} & 0 \\ \dot{\gamma} & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \tag{67}$$

The stress $\underline{\sigma}$ of such a flow also only has these two non-vanishing components for a Generalized Newtonian fluid. For a viscoelastic fluid, diagonal components are present[17]. From these, the first normal stress difference N_1 is defined as follows.

$$N_1 = \sigma_{xx} - \sigma_{yy} \tag{68}$$

There are other methods to quantify viscoelasticity. For example the frequency response of the fluid, which is more popular these days. The first normal stress difference is easy to measure and easy to derive for all models presented in the following (see appendix F.2 for derivation). This is why it is used in this thesis to quantify viscoelasticity. The viscoelastic models are presented in the following.

3.7. Viscoelastic models

There is a large amount of viscoelastic models. As Oliveira [57] shows, many of these can be expressed using a common form and thus actually are variants of a common idea. This allows both to easily find new models and implement them. The common notation presented in the following is, however, different to the notation presented by Oliveira, because it was optimized for implementability. All the models presented in this thesis use the upper-convected time derivative which for an arbitrary tensor $\underline{A}$ is defined as follows.

$$\overset{\nabla}{\underline{A}} = \frac{\mathrm{D}\underline{A}}{\mathrm{D}t} - \left((\nabla\vec{u})^{\mathrm{T}} \cdot \underline{A} + \underline{A} \cdot (\nabla\vec{u})\right) \tag{69}$$

This equation contains the material derivative defined as follows.

$$\frac{\mathrm{D}\underline{A}}{\mathrm{D}t} = \frac{\partial \underline{A}}{\partial t} + \vec{u} \cdot \nabla \underline{A} \tag{70}$$

It should be noted, that the transpose operation in the upper-convected time derivative can sometimes be found on the other gradient term due to ambiguity of the $\nabla\vec{u}$ notation. Here it is defiend as follows.

$$(\nabla\vec{u})_{ij} = \nabla_i u_j \tag{71}$$

The transpose can also be found in literature. Usually without mentioning the definition. For these models as they are presented in this thesis, the total stress $\underline{\sigma}$ in the fluid is

[17] Other off-diagonal components may also be present.

composed of a Newtonian solvent part (denoted $\underline{\sigma}_{\mathrm{s}}$) and a viscoelastic polymer stress (denoted $\underline{\tau}$).

$$\begin{aligned} \underline{\sigma} &= \underline{\sigma}_{\mathrm{s}} + \underline{\tau} && (72) \\ &= 2\eta_{\mathrm{s}}\underline{D} + \underline{\tau} && (73) \end{aligned}$$

While this naming strongly suggests a polymer solution, it is not exclusive to it and is also used for different systems. Literature often includes an index for the polymer stress and denotes it $\underline{\tau}_{\mathrm{p}}$ or $\underline{\sigma}_{\mathrm{p}}$. The models as discussed in the following only describe the evolution of the polymer part. The solvent contribution is an additional offset, that is independent of this time evolution. The value and existence of the infinite shear viscosity is up for debate. Physically, the viscosity may not vanish. There are arguments to be made, that the viscosity does stay above the solvent viscosity. However, the required shear-rate range is not accessible experimentally, and small deviations do not affect the simulation. Consequently, a choice was made to just use the solvent viscosity as the infinite shear viscosity in this thesis. This is most likely close enough. The polymer stress also provides viscous stresses. In the following all models currently implemented in *FluidX3D* are described. The notation of the original form mostly follows close to Oliveira [57]. The common notation used for implementation and analysis is presented in the following.

3.7.1. Unified notation

Instead of working with $\underline{\tau}$ directly, the implementation uses the polymer-conformation tensor $\underline{C}$. This tensor is a dimensionless proxy for the polymer stress. Many of the model differences can be absorbed by its definition. The relation for the conversion from $\underline{C}$ to $\underline{\tau}$ is given for each model. This conversion needs to be performed in the simulation whenever the stress is coupled into the LBM algorithm. For Oldroyd-B (see subsection 3.7.2) this does not seem necessary and might even be slightly detrimental to performance. However, such a replacement simplifies the constitutive equation for FENE-P (see subsection 3.7.3) significantly, providing large benefits in terms of implementation complexity, runtime and memory requirements. Doing this replacement for all models allows a unified code base with small adaptions for each specific model. This reduces implementation overhead and makes implementation errors less likely. Consequently, $\underline{C}$ is used as the polymer-conformation tensor for each model, although its relation to the polymer stress and therefore its exact meaning actually changes for the different models listed here. With this all the models presented here can be brought to the common form given below.

$$\frac{\partial \underline{C}}{\partial t} + \vec{u} \cdot \nabla \underline{C} = \left((\nabla \vec{u})^{\mathrm{T}} \cdot \underline{C} + \underline{C} \cdot (\nabla \vec{u}) \right) + \underline{S}_{\mathrm{R}} + 2\underline{D} \tag{74}$$

$$\overset{\nabla}{\underline{C}} = \underline{S}_{\mathrm{R}} + 2\underline{D} \tag{75}$$

Where $\underline{S}_{\mathrm{R}}$ is a model-dependent source term. One can see, that any model difference actually can be put into the definition of the source term $\underline{S}_{\mathrm{R}}$ and the polymer-conformation tensor $\underline{C}$. This is remarkable as they initially seem to have forms, that differ significantly. Consequently, only these terms actually need to be specified to fully describe the models. The following sections only list the results. The derivations can be found in appendix D. In the following, the simple and therefore commonly used Oldroyd-B model is discussed first.

3.7.2. Oldroyd-B

The Oldroyd-B model was originally developed by Oldroyd [58] in the infancy of the field. It is an important historic stepping stone towards the models used in the present work. It can still often be seen in literature. However, it is elastic, but not shear-thinning. Therefore, it is not used in this thesis. It also produces diverging extensional stress in finite extensional flow [57], which necessitated the development of new models. It models the polymers in solution as infinitely extensible dumbbells. This is one of two approaches for theory informed models discussed in this thesis and will also be relevant to other models. The constitutive equation for the Oldroyd-B model reads as follows.

$$\overset{\nabla}{\underline{\tau}} = -\frac{\underline{\tau}}{\lambda} + 2\frac{\eta_\mathrm{p}}{\lambda}\underline{D} \tag{76}$$

One can show, that this is a limit of the PTT model (see subsection 3.7.4) and it is implemented in *FluidX3D* as such. It is also a limit of the FENE-P model (see subsection 3.7.3). In standard notation (see appendix D.1) it is written as follows.

$$\underline{S}_\mathrm{R} = -\frac{\underline{C}}{\lambda} \tag{77}$$

$$\underline{\tau} = \frac{\eta_\mathrm{p}}{\lambda}\underline{C} \tag{78}$$

In a pure shear-flow its viscosity and first normal stress difference can be given as follows (see appendix F.2.1 for derivation).

$$\eta = \eta_\mathrm{p} + \eta_\mathrm{s} \tag{79}$$

$$N_1 = 2\eta_\mathrm{p}\lambda\dot{\gamma}^2 \tag{80}$$

Note, that the elastic forces are proportional to the viscosity η_p. An example plot of the first normal stress difference can be seen in figure 8.

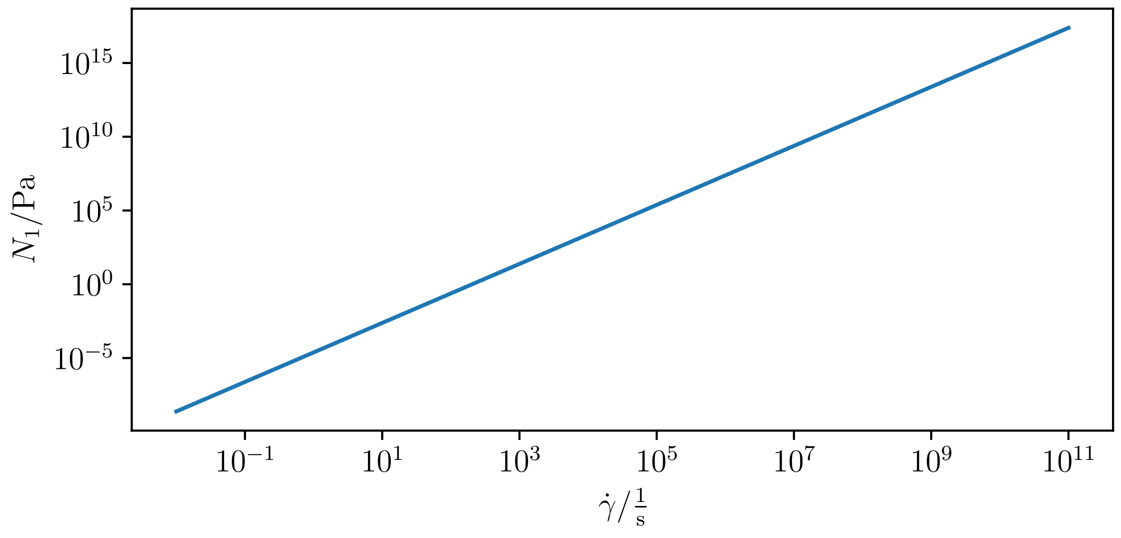


Figure 8: Example of an Oldroyd-B curve for the first normal stress difference. This is a simple power-law.

Given, that the viscosity is constant, the velocity profile in a Poiseuille flow[18] matches the one of a Newtonian fluid as can be seen below.

$$\vec{u}(r) = \frac{G}{2^{j+1}\eta}\left(R^2 - r^2\right)\hat{e}_x \tag{81}$$

Here $G = -\frac{\partial p}{\partial x}$ is the pressure gradient, R is the radius of the pipe and $j = 0$ for 2D and unity for 3D. The fact, that it is not shear-thinning, means, that consequently, the definition of the Weissenberg number must be as follows.

$$Wi = 0 \tag{82}$$

This too, will be shown to be consistent with FENE-P and PTT. In the following FENE-P is discussed first as a logical evolution step of Oldroyd-B.

3.7.3. FENE-P

The assumptions of Oldroyd-B are not all valid and it does not reproduce shear-thinning behavior. A straight-forward idea is to extend it by limiting the polymers to finite extensibility. This is where the name comes from. "FENE" stands for finite extensibility. The "P" stands for Peterlin-closure, which is a statistical approximation [57]. This eventually lead to the FENE-P model, which was originally developed by Bird [59, 60]. There exist many more models in the FENE-family, some are derived from FENE-P, others are more akin to parallel development. FENE-P is the most popular, so this is the one discussed here. However, looking at the data of real fluids developing new

[18]A Poiseuille flow, strictly speaking, is defined for a Newtonian fluid. This thesis uses a broader definition (see appendix P.2)

FENE-type models clearly is interesting. This will be discussed later (see section 19) The constitutive equation of FENE-P reads as follows.

$$\frac{Z}{\lambda}\underline{\tau} + \overset{\nabla}{\underline{\tau}} - \left(\underline{\tau} + \frac{\eta_\mathrm{p}}{\lambda}\mathbb{1}\right)\frac{1}{Z}\frac{\mathrm{d}Z}{\mathrm{d}t} = 2\frac{\eta_\mathrm{p}}{\lambda}\underline{D} \tag{83}$$

$$Z = 1 + \frac{\lambda}{\eta_\mathrm{p}}\frac{\mathrm{Tr}\,\underline{\tau}}{\hat{b}+3} \tag{84}$$

Here $\hat{b}$ is the additional dimensionless model parameter. It relates to the limit of the extension of the polymers and is always positive. Note, that this notation differs from Bird significantly (see appendix D.2). Aside from some scaling, notably, the sign of $\underline{\tau}$ has been defined differently here. In standard notation (see appendix D.2) it is written as follows.

$$\underline{\tau} = \frac{\eta_\mathrm{p}}{\lambda}\left(\frac{\hat{b}\underline{C} + \mathrm{Tr}\,\underline{C}\mathbb{1}}{\hat{b} - \mathrm{Tr}\,\underline{C}}\right) = -\eta_\mathrm{p}\underline{S}_\mathrm{R} \tag{85}$$

$$\underline{S}_\mathrm{R} = -\frac{\hat{b}\underline{C} + \mathrm{Tr}\,\underline{C}\mathbb{1}}{\lambda\left(\hat{b} - \mathrm{Tr}\,\underline{C}\right)} \tag{86}$$

In a pure shear-flow its viscosity and first normal stress difference can be given as follows (see appendix F.2.2 for derivation).

$$\eta(\dot{\gamma}) = \frac{\eta_\mathrm{p}}{\lambda\dot{\gamma}}\sqrt{\frac{2\left(\hat{b}+3\right)}{3}}\sinh\left(\frac{1}{3}\operatorname{arcsinh}\frac{3\lambda\dot{\gamma}}{\sqrt{\frac{2\left(\hat{b}+3\right)}{3}}}\right) + \eta_\mathrm{s} \tag{87}$$

$$N_1 = 2\frac{\eta_\mathrm{p}}{\lambda}\frac{2\left(\hat{b}+3\right)}{3}\sinh^2\left(\frac{1}{3}\operatorname{arcsinh}\frac{3\lambda\dot{\gamma}}{\sqrt{\frac{2\left(\hat{b}+3\right)}{3}}}\right) \tag{88}$$

Example plots of the viscosity and first normal stress difference can be seen in figures 9 and 10.

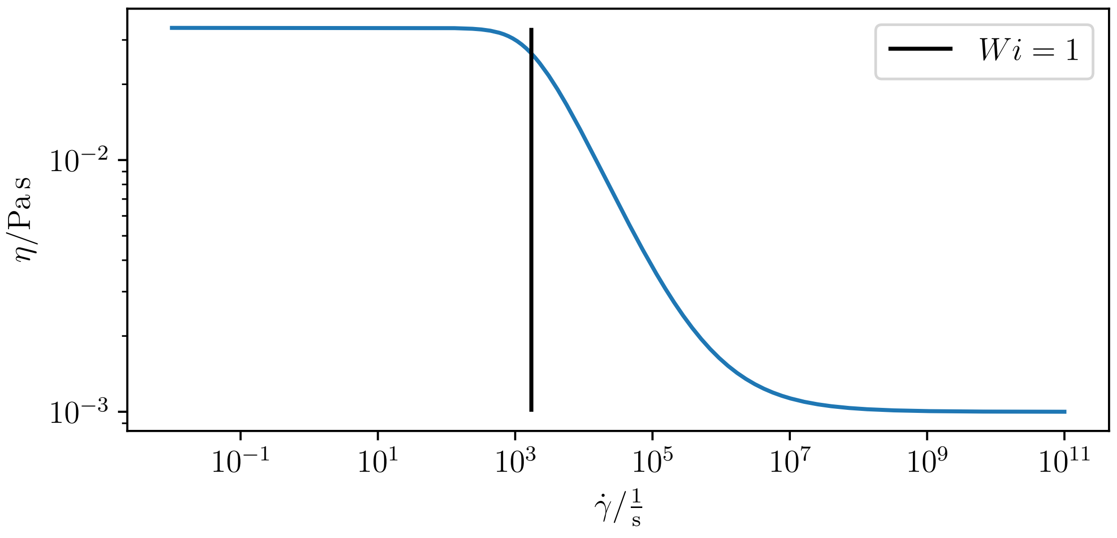


Figure 9: Example of an FENE-P curve for the viscosity. Notice the similarity to the equivalent CY curve in figure 3.

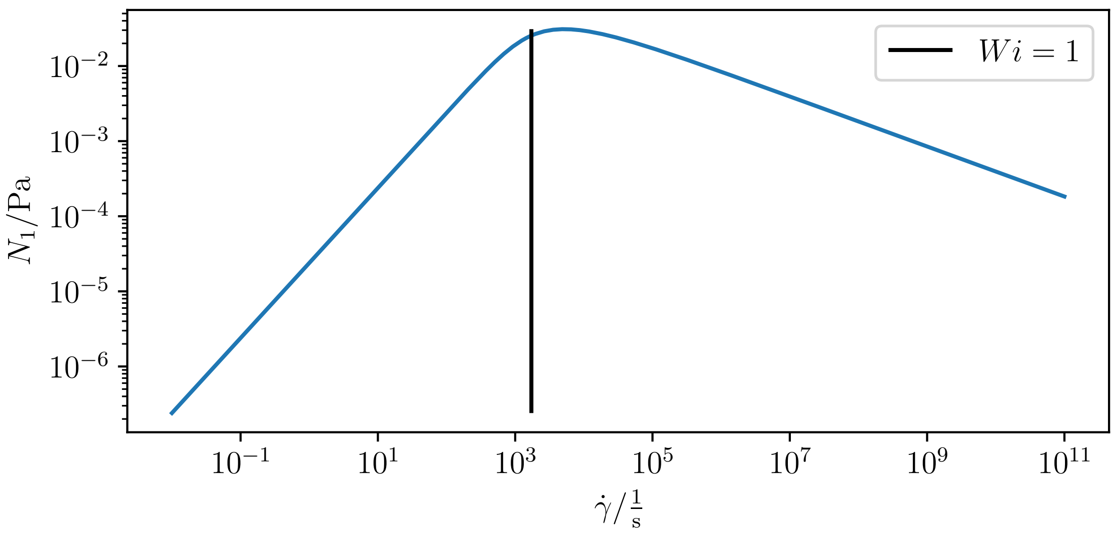


Figure 10: Example of an FENE-P curve for the first normal stress difference. After a maximum is reached, N_1 diminishes again. This is different from PTT (see figure 13). However, this region is not accessible with rheology.

Note, that the elastic forces are proportional to the viscosity η_p, as is the case for Oldroyd-B. This restricts the possibility for allowing N_1 behavior to differ from the behavior of the viscosity. However, this is what happens in some real fluids, yielding unphysical fitting results. Some model development might be beneficial here. In the cases where fitting works, defining the shear-rate $\dot{\gamma}_\mathrm{H}$ at which the viscosity contribution of the polymers is halved as follows can help.

$$\dot{\gamma}_{\mathrm{H}} = \frac{1}{\lambda}\sqrt{2\left(\hat{b}+3\right)} \tag{89}$$

With this definition, the viscosity can be written in a very fitting-friendly way as follows.

$$\eta(\dot{\gamma}) = \eta_{\mathrm{p}}\frac{\dot{\gamma}_{\mathrm{H}}}{\dot{\gamma}}\frac{1}{\sqrt{3}}\sinh\left(\frac{1}{3}\operatorname{arcsinh}\left(3\frac{\dot{\gamma}}{\dot{\gamma}_{\mathrm{H}}}\sqrt{3}\right)\right) + \eta_{\mathrm{s}} \tag{90}$$

The velocity profile in a Poiseuille flow[19] can be obtained by numerically integrating the following equation over r (see appendix F.2.2 for derivation).

$$\dot{\gamma} = -\frac{R_\eta}{\lambda}\sqrt{\frac{2\left(\hat{b}+3\right)}{3}}(R_\eta+1)\sinh\left(\frac{1}{3}\operatorname{arcsinh}\frac{Gr3^{\frac{3}{2}}R_\eta\lambda}{2^j\eta_{\mathrm{p}}\sqrt{2\left(\hat{b}+3\right)}(R_\eta+1)^{\frac{3}{2}}}\right) - \frac{Gr}{2^j\eta_{\mathrm{s}}} \tag{91}$$

Here $G = -\frac{\partial p}{\partial x}$ is the pressure gradient, and $j = 0$ for 2D and unity for 3D. The viscosity ratio R_η is defined as follows.

$$R_\eta := \frac{\eta_{\mathrm{p}}}{\eta_{\mathrm{s}}} \tag{92}$$

In the case of $\eta_{\mathrm{s}} = 0$ an analytical solution is possible and given in the following.

$$\vec{u}(r) = \frac{G}{2^{j+1}\eta_{\mathrm{p}}}\left[\frac{G^2\lambda^2}{2^{2j}\eta_{\mathrm{p}}^2\left(\hat{b}+3\right)}\left(R^4-r^4\right)+\left(R^2-r^2\right)\right]\hat{e}_x \tag{93}$$

Here R is the radius of the pipe. The resulting shear-thinning velocity profile can be seen in figure 11

[19]A Poiseuille flow, strictly speaking, is defined for a Newtonian fluid. This thesis uses a broader definition (see appendix P.2)

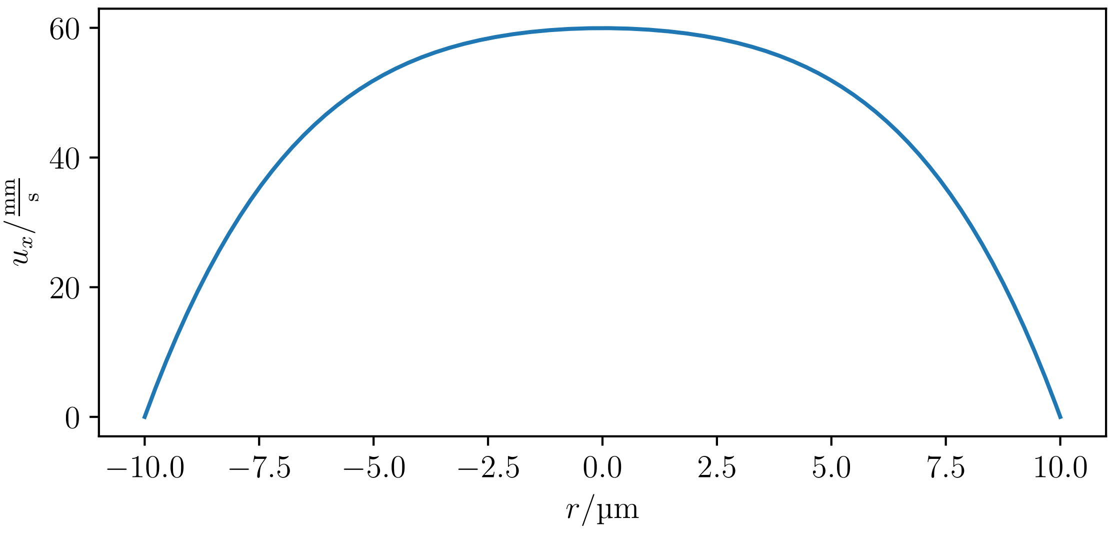


Figure 11: Example of a typical FENE-P Poiseuille flow. It is near identical to the CY equivalent (see figure 6).

Notably, this retrieves the Poiseuille flow solution in the Oldroyd-B limit ($\hat{b} \to \infty$). Given η_s is typically small, this is often a good approximation, especially for low to medium Wi. For FENE-P, the Weissenberg number Wi is defined as follows (see appendix F.2.2 for derivation).

$$Wi = \frac{3}{\sqrt{2\left(\hat{b}+3\right)}}\lambda\dot{\gamma} \tag{94}$$

The FENE-P model limits the infinitely extensible dumbbells to a finite extensibility $\hat{b}$. Consequently, $\hat{b} \to \infty$ reproduces the Oldroyd-B model. Particularly for large shear rates, it is numerically possible, that the polymer exceeds its maximal extensibility. If it is extended beyond the maximum the equations cause it to extend further, making this model unstable for high shear rates. This problem is further discussed in appendix E. The FENE-P model is interesting for its good fit to the experimental fluid data, but the numeric limitations can currently not be overcome. This limits its use in this thesis. Instead, the PTT model, which will be discussed next is primarily used.

3.7.4. PTT

The Phan-Thien-Tanner (PTT) model, is the model mostly used for this thesis (in conjunction with CY). It was originally developed by Phan-Thien and Tanner [61] and soon after slightly extended by Phan-Thien [62] to the exponential form used in this thesis. There is also an even more general form from Ferrás [63], which due to its use of the Gamma function is hard to implement on GPUs. As *FluidX3D* is GPU based, the exponential version is preferred here. The constitutive equation reads as follows.

$$\overset{\nabla}{\underline{\tau}} = -\xi(\underline{\tau}\cdot\underline{D}+\underline{D}\cdot\underline{\tau}) - e^{\frac{\epsilon\lambda}{\eta}\operatorname{Tr}\underline{\tau}}\frac{\underline{\tau}}{\lambda} + 2\frac{\eta}{\lambda}\underline{D} \tag{95}$$

Where ϵ and ξ are additional dimensionless model parameter. This model is based on a polymer network. ϵ describes the extensibility of the network and ξ describes the slip within the network [63]. For our polymer solutions ξ is always (effectively) zero and is set to exactly zero throughout this thesis for simplicity. In standard notation (see appendix D.3) it is written as follows.

$$\underline{S}_\mathrm{R} = -\xi(\underline{C} \cdot \underline{D} + \underline{D} \cdot \underline{C}) - e^{\epsilon \operatorname{Tr} \underline{C}} \frac{\underline{C}}{\lambda} \tag{96}$$

$$\underline{\tau} = \frac{\eta_\mathrm{p}}{\lambda} \underline{C} \tag{97}$$

In a pure shear-flow its viscosity and first normal stress difference can be given as follows (see appendix F.2.3 for derivation).

$$\eta(\dot{\gamma}) = \frac{\eta_\mathrm{p}}{\exp[0.5 W_0(4\epsilon\lambda^2\dot{\gamma}^2)]} + \eta_\mathrm{s} \tag{98}$$

$$N_1 = \frac{\eta_\mathrm{p}}{2\epsilon\lambda} W_0\left(4\epsilon\lambda^2\dot{\gamma}^2\right) \tag{99}$$

Where W_0 is the Lambert W function. Example plots of the viscosity and first normal stress difference can be seen in figures 12 and 13.

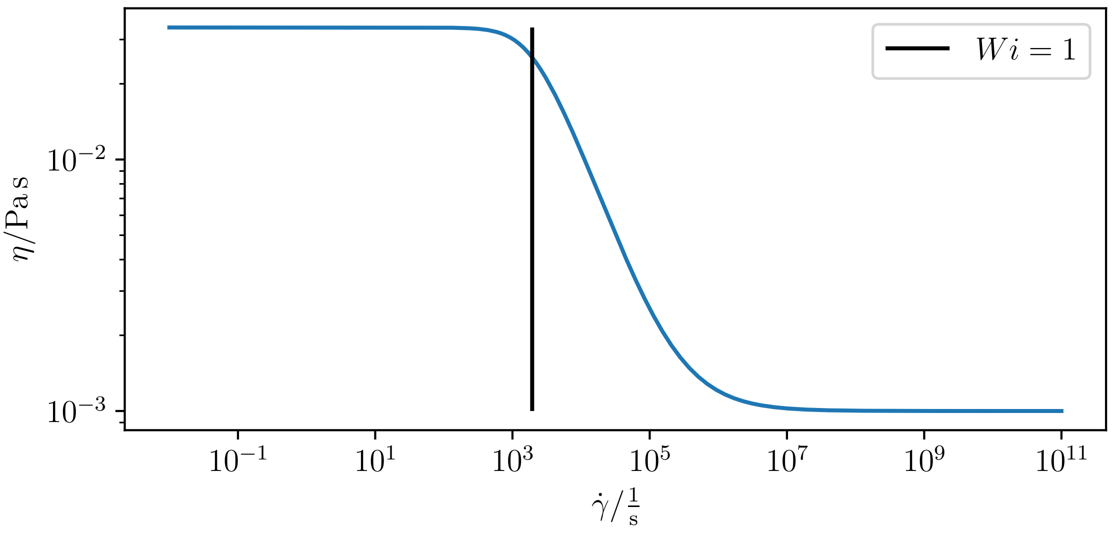


Figure 12: Example of an PTT curve for the viscosity. Notice the similarity to the equivalent CY curve in figure 3.

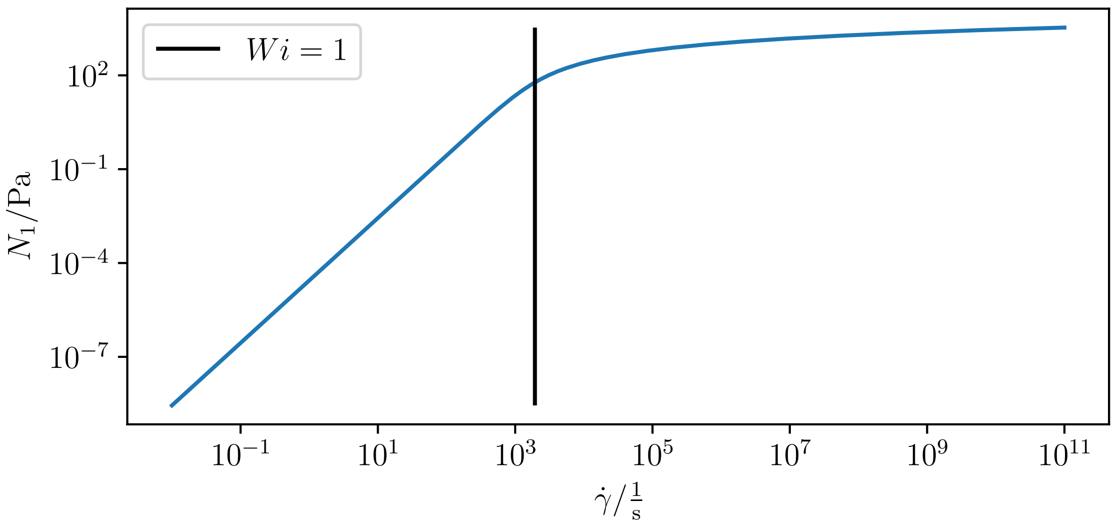


Figure 13: Example of an PTT curve for the first normal stress difference. N_1 rises continually with a power law, but changes its exponent relatively abruptly in the middle. This is different from FENE-P (see figure 10). However, this region is not accessible with rheology.

Note, that the elastic forces are proportional to the viscosity η_p, as is the case for Oldroyd-B. The velocity profile in a Poiseuille flow[20] can be obtained through some numerical steps (see appendix F.2.3). The resulting shear-thinning velocity profile can be seen in figure 14

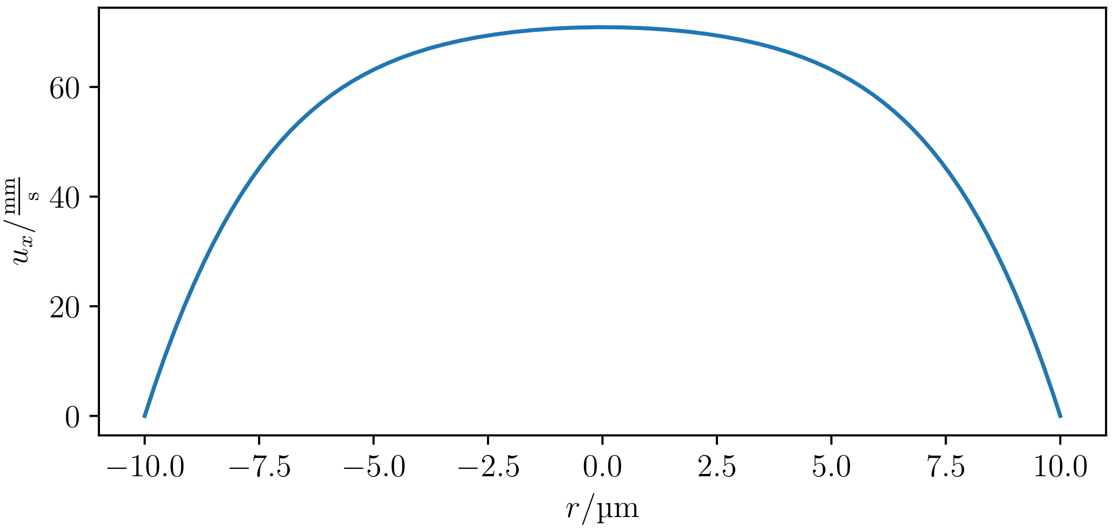


Figure 14: Example of a typical PTT Poiseuille flow. It is near identical to the CY equivalent (see figure 6).

[20] A Poiseuille flow, strictly speaking, is defined for a Newtonian fluid. This thesis uses a broader definition (see appendix P.2)

For PTT, the Weissenberg number Wi is defined as follows (see appendix F.2.3 for derivation).

$$Wi = 2\sqrt{\epsilon}\lambda\dot{\gamma} \tag{100}$$

The PTT model becomes the Oldroyd-B model for $\xi = \epsilon = 0$. Physically this means removing the slip and extensibility limits of PTT, which returns the Oldroyd-B description. With this, the fluid models used in this thesis are defined. Some common occurrences can be observed, which will be discussed in the following.

3.8. Common remarks

All the shear thinning models are capable to produce viscosity curves, that are very similar in the experimentally accessible range. Consequently, they are able to produce near-identical velocity profiles in generalized Poiseuille flow conditions (see appendix P.2). Remarkably, this is despite the equations for the velocity profile being completely different. They do differ significantly in the first normal stress difference. However, this is mostly in ranges not accessible by experiment. The very easy (and cheap to simulate) CY model produces the same velocity profile as PTT in generalized Poiseuille flow. This poses the question; When does the first normal stress difference actually matter, and when can one get away with running CY? Most of the thesis is about answering this questions for different use-cases. All the viscoelastic fluid models are somewhat viable due to their similarity. The choice in the end comes down to mostly implementation related reasons. These do favor PTT and therefore, this is the primary model used in this thesis (aside from CY for comparison). How to make the computer actually calculate these models and how this system interacts with the LBM algorithm will be discussed next.

4. Finite-volume algorithm for polymer stress

The discussion of LBM so far only covered Newtonian fluids with a fixed viscosity, as these are present by default in LBM. Generalized Newtonian fluids (see section 3.5) can be implemented via modification of the relaxation time τ. This has been done by for example by Müller *et al.* [64] in *ESPResSo* [49, 50] before. The present work opts for using the stress coupling introduced in section 2.5.2 instead[21]. This is easier, and the stress coupling is needed for viscoelastic fluids anyway, as will be seen later. The (viscous) stress (see section 3.5) provided by the fluid instantaneously reacts to changes in the strain-rate tensor $\underline{D}$ and is therefore available at any point in space and time. However, some of the fluids used in this thesis are retarded. The stress tensor provided by viscoelastic fluids shows non-trivial spatio-temporal dynamics. As this advection-reaction equation is slightly different for each fluid a general notation has been developed. This equation (eq. 74), reiterated in the following, describes the evolution of the polymer-conformation tensor $\underline{C}$, which is a dimensionless proxy for the polymer stress tensor $\underline{\tau}$.

$$\frac{\partial \underline{C}}{\partial t} + \vec{u} \cdot \nabla \underline{C} = \left((\nabla \vec{u})^{\mathrm{T}} \cdot \underline{C} + \underline{C} \cdot (\nabla \vec{u}) \right) + \underline{S}_{\mathrm{R}} + 2\underline{D} \tag{101}$$

Here $\underline{S}_{\mathrm{R}}$ is a model-dependent source term, $\vec{u}$ is the velocity field and $\underline{D}$ is the strain-rate tensor. Section 3.7.1 shows, that any model difference can be put into the definition of the source term $\underline{S}_{\mathrm{R}}$ and the polymer-conformation tensor $\underline{C}$ for the models used in this thesis. To solve this equation, a finite-volume scheme is employed. It runs in parallel with the LBM solver using an identical time-step. This scheme subdivides the simulation volume in cubes of $1 \times 1 \times 1$ (in LU). These cubes are centered around each lattice node. This choice allows for particularly easy implementation of the boundary conditions (see section 13.1). The value at the lattice node is interpreted as the average value of the polymer-conformation tensor and the strain rate tensor within the respective cube. The advection term (the second term on the left-hand side of equation 101) is handled using a special advection algorithm. Two possible choices are shown in the following sections. The derivatives on the right-hand side of equation 101 are calculated using finite differences. Finally, the source terms are integrated using the Euler method. The polymer-conformation tensor described by this scheme, or rather its associated stress (plus a few offsets described in section 18) are coupled back into the LBM with the stress coupling introduced in section 2.5.2. Notably, *FluidX3D* only executes the finite-volume algorithm for LBM nodes flagged as `TYPE_F` for historic reasons. The only non-trivial step here is the advection scheme, which will be discussed next.

4.1. Advection scheme

Advection can happen from and to any valid flux neighbor. With the grid choice outlined above walls are simply no valid flux neighbors, making these boundaries trivial. More

[21]There are some strings attached. Notably the need for shuffling (see section 18)

complex boundaries may are may not be valid flux neighbors (see section 13.1). Special thought is needed, in regard to deciding to what degree polymer-conformation tensor flux is allowed within cells or across cell boundaries. Handling this is discussed later (see section 4.3). Once the valid neighbors are found an advection scheme is employed. A discretized version of equation (101) could read as follows.

$$\underline{C}(\vec{x}, t + \Delta t) = -\underline{J}(\vec{x}, t)\Delta t + \underline{S}(\vec{x}, t)\Delta t + \underline{C}(\vec{x}, t) \tag{102}$$

Where all the source terms are represented by $\underline{S}$ and $\underline{J}$ is the yet to be determined flux. *FluidX3D* has two avection schemes implemented (Two options how to determine this flux). This work has been done by Fabian Häusl [65]. These two algorithms are the simple scheme and the corner transport upwind scheme (CTU) scheme. This thesis exclusively uses CTU for its superior stability, but simple shall be explained due to it being simple.

4.1.1. Simple

From averaging the constitutive equation over the box surrounding the LBM node and transforming this into a discretized surface integral, one can get the following [65]. The equation has been adjusted to fix the dimensionality.

$$(\underline{J})_{ij}(\vec{x}, t) = \frac{1}{\Delta x^d} \sum_{lk} I_{ijk}\left(\vec{x} + \frac{1}{2}\vec{c}_l \Delta t, t\right)(\hat{n}_l)_k \frac{\Delta x^{d-1}}{A_0} \tag{103}$$

Where d is the number of dimensions. One can see, that the flux through a discrete number of surfaces is calculated and summed. The flux I through a surface is defined as follows.

$$I_{ijk} = c_{ij} u_k \tag{104}$$

Where c_{ij} refers to the components of $\underline{C}$. For this term, the polymer-conformation tensor $\underline{C}$ and the velocity $\vec{u}$ needs to be known at $\vec{x} + \frac{1}{2}\vec{c}_l \Delta t$, aka at the surface of box surrounding the LBM node. This is done by averaging the values at the two nodes involved in the transfer. This flux is projected onto the normal vector in that direction $\hat{n}_i$, which is defined as follows.

$$\hat{n}_i = \frac{\vec{c}_i}{|\vec{c}_i|} \tag{105}$$

This is weighted by the area of the surface associated with this normal vector. Which is defined as follows.

$$\frac{\Delta x^2}{A_0} \tag{106}$$

Where A_0 reads as follows.

$$A_0 = \frac{1}{2d} \sum_i |\vec{c}_i| \frac{\Delta t}{\Delta x} \tag{107}$$

According to Häusl [65] and Kuron *et al.* [37], this works for arbitrary velocity sets. It does not even have to be the same as for LBM. The issue is, that this only accurately reproduces the integral for D2Q5 and D3Q7. If the diagonal velocity sets are associated with surfaces, the box surrounding each lattice node cannot be a cube. Which likely alters is volume, although the equation implies it to stay the same. Also, the surface of this new "box", does not (likely) lie halfway between the nodes. Kuron *et al.*cite Capuani *et al.* [66] for equation (107). However, the given source does not give the equation, but only a result for "D3Q18", that is consistent with the equation. So, this is some kind of arcane knowledge with an unknown origin. From A_0[22] one can reconstruct the surface area of the volume, that was used to approximate the integral. For D3Q27, this results in a total area of $A \approx 4.2$ (in LU). The primitive guess of a potential shape, which can be seen in figure 15, yields $A \approx 3.7$.

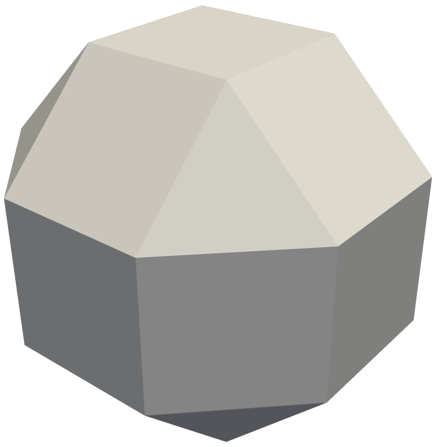


Figure 15: Render of a naive example for how the FV box might actually look when considering edge- and corner neighbors for the simple advection algorithm.

This means, that the shape actually represented by equation (107) is further from a sphere, than the guess above. One can conclude, that in the construction, the faces are likely of different size to account for the different directions being of differing importance (similar to the lattice weights for LBM). However, the numeric integral gets calculated with the average, rendering this thought useless. The authors of the referenced publications note, that using larger velocity sets than D3Q7 does not yield any improvement [37, 65]. Given how many questions remain, it is surprising, that it works at all with larger velocity sets and should be considered "Here be dragons 🐉"-territory. Given, this simple approach is prone to some instabilities [65], this thesis uses the corner transport upwind (CTU) scheme instead. It can reduce these instabilities and will be explained in the following.

4.1.2. CTU

Contrary to the linear (in the velocity) approach of the simple advection algorithm, the corner transport upwind (CTU) scheme is volumetric [37]. The basic idea of CTU is to

[22] A_0 is treated by the sources like an area, and has the dimension of a velocity. It should be neither.

virtually displace the finite volume cell (the Δx^3 cube mentioned before) by $\vec{u}\Delta t$, where $\vec{u}$ is the local velocity and Δt is the size of a time-step. This displacement results in an overlap with the neighboring cells. Taking the overlap volume as a fraction of the cell's volume results in the percentage of the polymer stress currently present within the cell. This percentage is advected to the respective neighbor. Mathematically this is somewhat involved and shown in the following based on the derivation from Häusl [65] and Kuron *et al.* [37]. Consider figure 16 as an illustration to reference during the mathematical explanation.

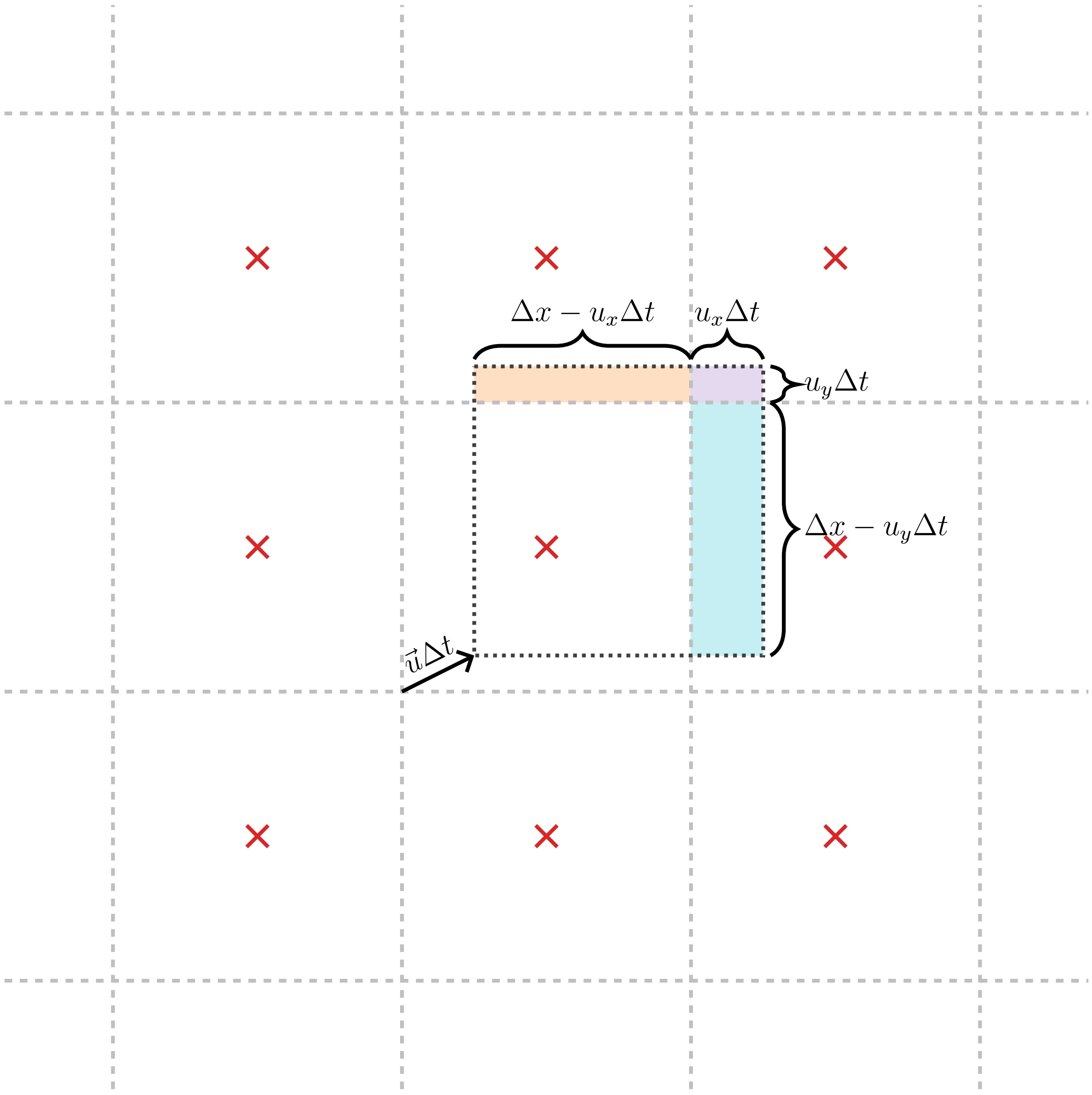


Figure 16: Illustration of the CTU algorithm (in 2D for visibility). Each lattice node has an associated volume, which gets moved by $\vec{u}\Delta t$. This creates overlaps with (in this case three) neighboring volumes. These define the magnitude of the advection.

The overlap volume V_i in direction i is defined as follows.

$$V_i(\vec{x},t) = \alpha_i(\vec{x},t)\prod_k l_{ik}(\vec{x},t) \tag{108}$$

The α decides, whether the overlap is considered. Only the overlap with the neighboring cells is considered. This means, only the outflow is tracked here. α is defined as follows.

$$\alpha_i = \begin{cases} 1 & \text{, if } u_k(\vec{x},t)(\vec{c}_i)_k \geq 0 \,\forall\, k \\ 0 & \text{, else} \end{cases} \tag{109}$$

This can be understood as no component of the velocity may point away from the neighbor currently considered. *FluidX3D* uses a more complicated condition to save on runtime. It is mathematically identical. l_{ik} gives the overlap length in a given direction and is defined as follows.

$$l_{ik}(\vec{x},t) = \begin{cases} \Delta x - u_k(\vec{x},t)\Delta t & \text{, if } (\vec{c}_i)_k = 0 \\ u_k(\vec{x},t)\Delta t & \text{, else} \end{cases} \tag{110}$$

Consult figure 16, to see, how this defines $l_{11}l_{12}$, $l_{31}l_{32}$ and $l_{51}l_{52}$. The outward flux $\underline{J}_{\mathrm{o},i}$ in direction i, is then defined as follows.

$$\underline{J}_{\mathrm{o},i}(\vec{x},t) = \frac{V_i(\vec{x},t)}{\Delta x^d}\underline{C}(\vec{x},t) \tag{111}$$

Here d refers to the number of dimensions. The total flux $\underline{J}$ is found by summing over all directions and by considering the inward flux provided by the neighboring nodes. This is defined as follows.

$$\underline{J}(\vec{x},t) = \sum_i \underline{J}_{\mathrm{o},i}(\vec{x},t) - \underline{J}_{\mathrm{o},i}(\vec{x}-\vec{c}_i\Delta t,t) \tag{112}$$

This scheme is known to prevent checkerboard instabilities [65]. One is technically free to choose the neighborhood used for CTU. Choosing one smaller than D3Q27 does incur an error, but is faster. In this thesis the choice is always a neighborhood consistent with the D3Q27 velocity set. This is done here in contrast to LBM, where a smaller velocity set is used, because larger velocity sets are a lot cheaper in CTU. Furthermore, the viscoelastic fluids are consistently close to instability, so the reward is greater. With this, the advection of the (dimensionless) stress is defined. Next, making the stress tensor more human-readable is discussed.

4.2. Scalar stress

Both the stress tensor $\underline{\tau}$ and its dimensionless proxy, the polymer-conformation tensor are tensors with 6 independent components. For comparison and understandability, it would be desirable to have a scalar measure for stress. Depending on which effect one wishes to discuss, two numbers are popular. The first of these is the pressure p, typically defined as follows.

$$p = \sum_i \frac{1}{3}\tau_{ii} \tag{113}$$

The remaining stress is termed the "deviator of the stress tensor" or the "deviatoric stress tensor" τ_{ij}^{dev}. It is defined as follows.

$$\tau_{ij}^{\text{dev}} = \tau_{ij} - \sum_k \frac{1}{3}\tau_{kk}. \tag{114}$$

It is this deviatoric stress, from which the other popular scalar stress is defined. The von Mises stress τ_{vM} is defined as follows [67].

$$\tau_{\text{vM}} = \sum_{ij} \sqrt{\frac{3}{2}\tau_{ij}^{\text{dev}}\tau_{ij}^{\text{dev}}}. \tag{115}$$

These measures are most commonly used in this thesis to describe the polymer stress of the fluid. They can however be applied to any stress. For examples to cell stress. The cell's interaction with the fluid will be covered next.

4.3. Handling (biological) cells

If biological cells or similar particles are present (These are handled via the Immersed Boundary Method (IBM); see section 6), these also need to be considered by the finite volume algorithm. There are two reasons for this. For one, the fluid on the inside in general behaves different from the fluid outside. This is typically called contrast and is discussed in the following. Furthermore, physically speaking, the polymer cannot pass through the cell membrane and the flux of the polymer-conformation tensor should reflect this. The group of algorithms that aim to handle this, is termed shoveling here and will be discussed first.

4.3.1. Stress shoveling

If flux through the cell wall is fully prevented, a build-up of polymer-conformation tensor at the boundary is observed. This eventually leads to problems. Previous work has been carried out to reduce this issue [65]. This resulted in different kinds of shoveling algorithms, where the polymer-conformation tensor is permitted to flow into the outermost layer of the cell. Subsequently, this stress is shoveled back out, distributing it more evenly in the process. This reduces the build-up of polymer-conformation tensor, but issues persist. These consist of the polymer-conformation tensor building up at different areas or oscillations getting promoted leading to instability. It was found, that letting the polymer-conformation tensor advect freely and just discarding the produced force works most reliably. While this is quite a trivial solution, it even outperforms more involved solutions in terms of accuracy. For this thesis, this approach of handling the polymer-conformation tensor flux was merged with a generalization of the concept of a viscosity contrast. This is discussed in the following.

4.3.2. Viscosity contrast

For Newtonian fluids, the viscosity contrast, meaning the fraction between the viscosity inside and outside the cell, is a well-defined concept. Its relevant effect on deformation and behavior of cells is well studied [68]. For viscoelastic fluids, this is not straight forward. In this thesis, this is handled by using a different parameter-set inside the cell than outside. This allows to reproduce (close to) any behavior inside the cell. This behavior may be entirely different from the fluid outside the cell. With this scheme, the polymer-conformation tensor, that is allowed to flow into the cell does have a valid use here. A notable limit of *FluidX3D* is, that the same fluid model has to be used inside and outside the cell. However, it is often desired to have the inside of the cell act Newtonian. For this purpose, it is notable, that all the models used in this thesis can be made to act like a Newtonian fluid through clever parameter use. Irrespective of if they are Generalized Newtonian or fully shear thinning, a Newtonian behavior can be archived, by setting $\eta_\mathrm{p} = 0$ and putting the desired viscosity into η_s. This sets the first Normal stress difference to 0 for all models, because they all scale with η_p as mentioned in section 3.7. The same section shows the viscosities becoming strictly Newtonian for $\eta_\mathrm{p} = 0$. Retaining all other parameters results in the constitutive equation in its common notation (eq. (75)) being preserved (see section 3.7). This allows the polymer conformation tensor to pass uninterrupted through the cell (as is desired; See subsection 4.3.1), while still providing adequate stress inside. With the fluids covered for now, it is time to consider putting cells into it. This will be discussed next.

5. Elastic models

In the following sections, cell behavior is described for experiments of increasing complexity. Before these, the basic effects describing the physical behavior of a cell are covered. These are its internal viscosity (see section 4.3.2) and its governing elastic equations. The elastic models used in this thesis are covered in this section. They can be split into two categories. The models dealing with surface effects (e.g. the cell membrane), and the models dealing with the bulk volume of the cell. This distinction also exists in *FluidX3D* using the flags `MEMBRANE_FORCES` and `TETRA`. A different definition of the dimensionless Capillary number Ca_K is required depending on category. Models of both categories can (and in some instances should) be combined to most accurately reproduce cell behavior. Before these models are covered, some shared notation to quantify the strain is introduced.

5.1. Strain

First, the deformation has to be quantified. Technically, there are multiple options, and more elaborate models use different strains. This makes a general description of the base concepts quite extensive. This thesis follows the definitions from previous works [69, 70] and only mentions the core results. For a more general approach, comprehensive literature exists [71]. A typical deformation can be seen in figure 17.

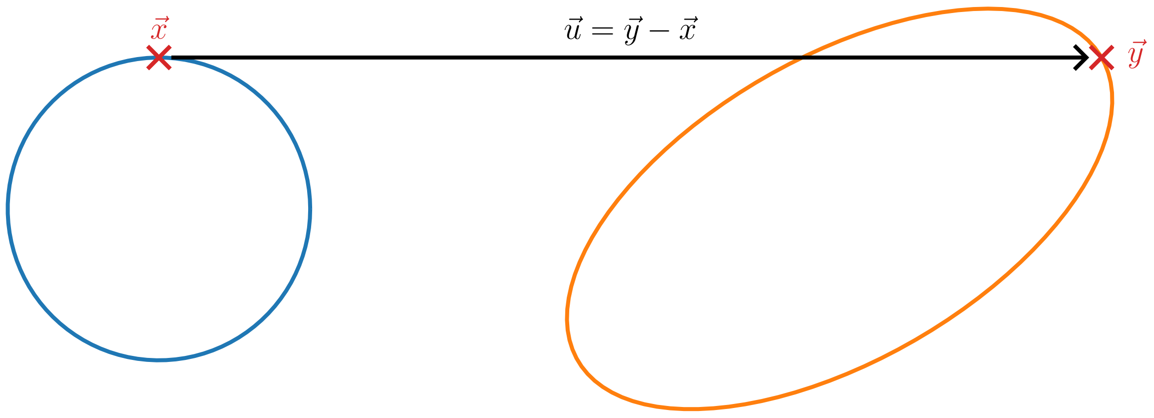


Figure 17: Illustration of the cross-sectional geometry for a cell under typical deformation: The transformation from the initial state, represented by the point $\vec{x}$, to the deformed state, represented by the point $\vec{y}$, can be written in material coordinate as a simple difference, the displacement $\vec{u}$.

Note, that $\vec{u} = \vec{y} - \vec{x}$ characterizes a deformation here. This is the typical notation and is commonly seen in literature. Here it is only mentioned for reference and quickly replaced by the deformation gradient tensor $\underline{F}$. It is defined as follows [71].

$$F_{ij} = \frac{\partial u_i}{\partial x_j} + \delta_{ij} = \frac{\partial y_i}{\partial x_j} \tag{116}$$

This is a simple strain measure, that is used for some linear models[23]. Note, that it is dimensionless, and consequently all quantities derived from it are as well. From the deformation gradient tensor the left Cauchy-Green deformation tensor $\underline{B}$ is calculated as follows.

$$\underline{B} = \underline{F}\underline{F}^{\mathrm{T}} \tag{117}$$

This is the strain measure most of the models in this thesis are based on. The following models are technically defined using the right Cauchy-Green deformation tensor, which has the product flipped. However, this leads to conflicting notations. For the cases discussed here, the stress can be shown to be identical either way. This is, because the only terms containing the Cauchy-Green deformation tensor are its trace and the trace of its square. Factors can be switched below the trace, and therefore, the following holds.

$$\mathrm{Tr}\left(\underline{F}\underline{F}^{\mathrm{T}}\right) = \mathrm{Tr}\left(\underline{F}^{T}\underline{F}\right) \tag{118}$$

$$\mathrm{Tr}\left(\left(\underline{F}\underline{F}^{\mathrm{T}}\right)^2\right) = \mathrm{Tr}\left(\underline{F}\underline{F}^{\mathrm{T}}\underline{F}\underline{F}^{\mathrm{T}}\right) = \mathrm{Tr}\left(\underline{F}^{\mathrm{T}}\underline{F}\underline{F}^{\mathrm{T}}\underline{F}\right) = \mathrm{Tr}\left(\left(\underline{F}^{\mathrm{T}}\underline{F}\right)^2\right) \tag{119}$$

One can see, that this allows to use the right and left Cauchy-Green deformation tensor interchangeably in this thesis. From the strain, scalar invariants are typically calculated and used for the actual stress-strain relationship. These vary depending on model, with one important commonality. The Jacobian determinant J of the deformation gradient tensor, which is defined as follows [69].

$$J = \det(\underline{F}) = \frac{\mathrm{d}V}{\mathrm{d}V_0} \tag{120}$$

Here $\mathrm{d}V$ and $\mathrm{d}V_0$ refer to the volume of an infinitesimal element of the cell, in the deformed and reference state respectively. This means, that J gives the volume change and therefore the following must hold for physical deformations.

$$J > 0 \tag{121}$$

If J is unity, the material is incompressible. With the strain defined, the stresses can be discussed. In the following, the models in relation to the cell surface will be covered first.

5.2. Membrane models

Elastic forces acting on the surface of a cell are well suited to model a cell membrane, giving this category its name. These models are well studied [68], the implementation is taken from previous work [65, 72][24] and the uses in this thesis was limited. Therefore, the coverage here is also limited. The membrane is described using shear forces (Skalak or Neo-Hookean), bending forces and artificial conservation forces for the area and volume. A force describing surface tension was added in this thesis. The artificial forces depend on the discrete implementation and are therefore covered later in section 6.4.1. The models are defined as follows.

[23]In literature, linear elasticity is usually defined using a different tensor, but it can be written with this one as well.

[24]Except for surface tension

5.2.1. Shear stress (Skalak/Neo-Hookean)

The relevant strain invariants for these models are termed I_1 and I_2. They are calculated from the strain defined in subsection 5.1 as follows [73].

$$I_1 = \mathrm{Tr}(\underline{B}) - 2 \tag{122}$$

$$I_2 = J^2 - 1 \tag{123}$$

With this, the energy density U of the Neo-Hookean model is defined as follows [73, 74].

$$U = \frac{\kappa_1}{6}\left(I_1 + \frac{1}{I_2 + 1} - 1\right) \tag{124}$$

While, the energy density of the Skalak model reads as follows [73–76].

$$U = \frac{\kappa_1}{6}\left(\frac{1}{2}I_1^2 + I_1 - I_2\right) + \frac{\kappa_2}{12}I_2^2 \tag{125}$$

Here κ_1 and κ_2 are two moduli. κ_1 relates to shear, while κ_2 relates to area conservation. Consequently, the Neo-Hookean model does not inherently conserve surface area. The notation is largely inconsistent across literature, with the denominators of the fraction containing the moduli varying. These effects can be absorbed into the definition of the moduli and are therefore irrelevant. Care has to be taken to assure consistency though. Next, the membrane's resistance to bending is discussed.

5.2.2. Bending resistance

The lipid bilayer encapsulating cells does also resist bending. The Helfrich model [77, 78] was designed to capture this behavior. The energy E_B stored in the bending reads as follows [79, 80].

$$E_\mathrm{B} = \int_S 2\kappa_\mathrm{B}(H - H_0)^2 \mathrm{d}S \tag{126}$$

Here S is the surface of the cell, κ_B is the bending modulus, and H and H_0 are the mean curvature and its reference. The correct choice for the reference curvature H_0 is up for debate. One could argue for flat being the correct reference, due to the structure of the bilayer. However, one could also assume, that in a cell, the bilayer is stress-free in its equilibrium shape. The preexisting code was modified, in this thesis, to allow for both. Simulations in simple standard geometries revealed no big difference. This concludes the physical membrane forces. Next, the bulk stresses will be explored.

5.3. Volumetric models

Aside from very simple cells like red blood cells, cells typically have a cytoskeleton and organelles. The elastic properties of those are captured, by treating the whole cell as an elastic solid. No significant error is incurred by treating the cells as homogeneous [81]. Half of the models in the following have been ported from preexisting software. The notation was however greatly improved in terms of readability compared to previous publications [69, 70]. The rest is known from literature or consists of small modifications. As these models are used throughout the thesis, they will be covered thoroughly. The volumetric models can be nonlinear both in their strain definition and in the stress-strain relationship itself. The models described in the following are ordered by their amount of nonlinearity, from the most linear model, to the most nonlinear. The linear elastic model is fully linear with a simple strain measure. The Modified Saint Venant–Kirchhoff model is linear with the typical strain measure used here. The Saint Venant–Kirchhoff model has a nonlinear strain. The Neo-Hookean model is additionally nonlinear in its stress-strain relationship. The Mooney-Rivlin model considers an additional faster growing term compared to the Neo-Hookean model. These models already cover shear stress and volume conservation. Consequently, no additional artificial forces are required for the bulk. Aside from the invariants covered in a previous subsection (see subsection 5.1), the following additional invariants are used by the volumetric stress models [69, 70].

$$I = \mathrm{Tr}(\underline{B})J^{-\frac{2}{3}} \tag{127}$$

$$K = \frac{1}{2}\left[\mathrm{Tr}(\underline{B})^2 - \mathrm{Tr}\left(\underline{B}^2\right)\right]J^{-\frac{4}{3}} \tag{128}$$

The following models also share the shear modulus G and the bulk modulus κ. Both of which can be expressed using the Young's modulus E, and the Poisson ratio ν as follows.

$$G = \frac{E}{2(1+\nu)} \tag{129}$$

$$\kappa = \frac{E}{3(1-2\nu)} \tag{130}$$

As the Young's modulus and Poisson ratio are more expressive, these are used throughout the thesis. However, the formulation of the models is easier using shear and bulk modulus. Consequently, they are written here using those. The most basic model shall be covered first.

5.3.1. Linear elastic

Linear elastic (LE) stress is typically defined using a strain tensor identical to the strain-rate tensor used for fluids. Here it shall be written using the deformation gradient tensor for consistency. Its energy density U_{LE} reads as follows [71].

$$U_{\mathrm{LE}} = \frac{G}{2}\left[\mathrm{Tr}\left(\underline{F}^2\right) + \mathrm{Tr}\left(\underline{F}^{\mathrm{T}}\underline{F}\right) - \frac{2}{3}\mathrm{Tr}(\underline{F})^2\right] + \frac{\kappa}{2}\left[\mathrm{Tr}(\underline{F}) - 3\right]^2 \tag{131}$$

This is only useful for small deformations and not invariant under all transformations (e.g. rigid body rotation). It is only used in this thesis as a limit to compare the other models to. It should be noted, that for small deformations, the last term can be approximated as follows.

$$\mathrm{Tr}(\underline{F}) - 3 \approx J - 1 \tag{132}$$

This means, that the volume conservation is present, even so it is not obvious. However, the energy does not diverge for $J \to 0$, which means, that a linear elastic cell can be deformed to the point of vanishing with a finite force. This is unphysical and does cause problems in simulations starting at medium deformations. Next, the first production model will be discussed.

5.3.2. Modified Saint Venant–Kirchhoff

The Modified Saint Venant–Kirchhoff model (SVKK) was developed in this thesis in order to produce forces between the Saint Venant–Kirchhoff model (subsection 5.3.3) and the Neo-Hookean model (subsection 5.3.4). Later sections (see section 20.4) will show, that experimental data lies between these two models, motivating the design of an intermediate model. The Saint Venant–Kirchhoff (SVK) model, can be in general be written as follows [82].

$$U_{\mathrm{SVK}} = G\left[\mathrm{Tr}\big(\underline{E}^2\big) - \frac{1}{3}\,\mathrm{Tr}\big(\underline{E}\big)^2\right] + \frac{\kappa}{2}\,\mathrm{Tr}\big(\underline{E}\big)^2 \tag{133}$$

Here $\underline{S}$ is a yet to be determined strain tensor. Fraldi *et al.* [82] demonstrate such a model with two different strains for the 1D case. They use the Green-Lagrange strain for one, which is commonly used in SVK model. Furthermore, the Biot strain is used, which is uncommon in 3D due to some mathematical difficulty it introduces. Here, a scaled Biot strain defined as follows is used.

$$\underline{E} = \frac{1}{\sqrt{2}}(\underline{S} - \mathbb{1}) \tag{134}$$

Where $\underline{S}$ is defined as follows.

$$\underline{S}^2 = \underline{B} \tag{135}$$

Typically, this would also be written using the right Cauchy-Green deformation tensor. Here, this does produce different results and the ones using the left Cauchy-Green deformation tensor have proven more desirable. Solving this equation is the aforementioned mathematical difficulty. The scaling (the modification) is done in order to place the resulting force between the Saint Venant–Kirchhoff and the Neo-Hookean model. Both these models describe the experimental data for hydrogel particles well (see section 20.4), with the Saint Venant–Kirchhoff model drifting slowly below the data and the Neo-Hookean model doing the opposite. Consequently, the Modified Saint Venant–Kirchhoff was created for an optimized description of the data. For small symmetric deformations, the following holds.

$$\underline{S} \approx \underline{F} \tag{136}$$

$$\mathrm{Tr}\big(\underline{E}\big)^2 \approx \frac{1}{4}(J-1)^2 \tag{137}$$

This means, volume conservation also hides in this term. However, the energy does not diverge for $J \to 0$, which means, that a Modified Saint Venant–Kirchhoff cell can be deformed to the point of vanishing with a finite force. An illustration for this can be found in literature [82]. This is unphysical and does cause problems in simulations starting at medium deformations. Next a less linear version of the Saint Venant–Kirchhoff model will be discussed.

5.3.3. Saint Venant–Kirchhoff

The Saint Venant–Kirchhoff (SVK) model in its more typical formulation, in principal uses the same equation as the Modified Saint Venant–Kirchhoff model, which is listed below [82].

$$U_{\mathrm{SVK}} = G\left[\mathrm{Tr}\left(\underline{E}^2\right) - \frac{1}{3}\,\mathrm{Tr}(\underline{E})^2\right] + \frac{\kappa}{2}\,\mathrm{Tr}(\underline{E})^2 \tag{138}$$

However, the definition of the strain $\underline{E}$, given in the following is nonlinear.

$$\underline{E} = \frac{1}{2}(\underline{B} - \mathbb{1}) \tag{139}$$

With the left Cauchy-Green deformation tensor $\underline{B}$, which is similar to a square of the deformation gradient tensor $\underline{F}$. The term corresponding to volume conservation does not transform to the same form as simply as for the other models. It still provides a force towards the equilibrium volume. However, the energy does not diverge for $J \to 0$, which means, that a Saint Venant–Kirchhoff cell can be deformed to the point of vanishing with a finite force. An illustration for this can be found in literature [82]. This is unphysical and does cause problems in simulations starting at medium deformations. In the valid range, the Saint Venant–Kirchhoff model is well suited to describe hydrogel particles (see section 20.4). Next the models capable of significant nonlinearity are covered.

5.3.4. Neo-Hookean

The Neo-Hookean (NH) model is, as the name suggests, inspired by Hookean forces. It is effectively the default model and is treated as such in this thesis. The reason for this is, that the Neo-Hookean model is well suited to describe hydrogel particles (see section 20.4). And it does not have the problem of the possibility of deformation to the point of vanishing with a finite force. Its energy density is listed in the following [69, 70].

$$U_{\mathrm{NH}} = \frac{G}{2}(I - 3) + \frac{\kappa}{2}(J - 1)^2 \tag{140}$$

The Neo-Hookean model can be interpreted as a limit of the Mooney-Rivlin model. It is implemented in *FluidX3D* as a limit of the Mooney-Rivlin model. The Mooney-Rivlin model will be covered next.

5.3.5. Mooney–Rivlin

The Mooney–Rivlin (MR) model is the most complex model in this thesis. It is capable of highly nonlinear behavior and has a parameter to increase these effects. Its energy density is listed in the following [69, 70].

$$U_{\mathrm{MR}} = \frac{G_1}{2}(I-3) + \frac{G_2}{2}(K-3) + \frac{\kappa}{2}(J-1)^2 \tag{141}$$

Note, that the shear modulus G is split here as follows.

$$G = G_1 + G_2 \tag{142}$$

For small deformations it does in fact act like a Neo-Hookean model with this sum as its shear modulus [70]. Furthermore, the two moduli are defined as follows.

$$G_1 = wG \tag{143}$$
$$G_2 = (1-w)G \tag{144}$$

This allows to tune the higher order non-linearity from Neo-Hookean ($w = 1$) to strongly strain hardening towards $w = 0$. Intermediate values of w can be chosen depending on the desired amount of strain hardening. The Mooney–Rivlin model is well suited to describe biological cell (see section 20.4). With the elastic models defined, one can think about the relevant dimensionless numbers. This is covered next.

5.4. Capillary number

The elastic force density $\vec{f}$ can be defined from the principal of virtual work as follows.

$$f_i = -\frac{\partial U}{\partial u_i} \tag{145}$$

$\vec{u}$ is the displacement mentioned in subsection 5.1. It should be noted, that this density is in respect to volume for the volumetric models and in respect to area for the membrane models. Adding an force changes the Navier-Stokes equation (eq. 1) as follows.

$$\rho\frac{\partial \vec{u}}{\partial t} + \rho(\vec{u}\cdot\nabla)\vec{u} = \mu\nabla^2\vec{u} - \nabla p + \vec{f} \tag{146}$$

Importantly, the $\vec{f}$ here is always in respect to the volume, so this needs to be compensated with an additional factor for the membrane models. Performing the non-dimensionalization from section 1.2.1 on this equation results with prefactors on the force term. For the volume forces these are as follows.

$$\frac{U_{\mathrm{t}}}{St\mu_{\mathrm{t}}\dot{\gamma}_{\mathrm{t}}} \tag{147}$$

With the typical energy density U_{t}, the Stokes number St, the typical dynamic viscosity μ_{t} and the typical shear-rate $\dot{\gamma}_{\mathrm{t}}$. For the membrane forces this equation differs by a typical length as expected. This can be seen in the following.

$$\frac{U_{\mathrm{t}}}{St\mu_{\mathrm{t}}u_{\mathrm{t}}} \tag{148}$$

Here u_t is the typical velocity. As noted in section 1.2.1, the Stokes number is unity in the relevant regime. This leaves the question of how to define the typical energy density. It is customary to define the shear modulus G or its membrane equivalent κ_1 respectively as the typical value. This introduces an additional dimensionless number (or two), which are defined as the fractions of the moduli. These typically get ignored, even so they are relevant, particularly for the Mooney-Rivlin model. With these choices, the prefactors become as follows.

$$\frac{G}{\mu_\mathrm{t}\dot{\gamma}_\mathrm{t}} \tag{149}$$

$$\frac{\kappa_1}{\mu_\mathrm{t}u_\mathrm{t}} \tag{150}$$

These are typically interpreted as the inverse of the Capillary number Ca. This results in the following.

$$Ca = \begin{cases} \dfrac{\mu_\mathrm{t}\dot{\gamma}_\mathrm{t}}{G} & \text{, for volumetric forces} \\ \dfrac{\mu_\mathrm{t}u_\mathrm{t}}{\kappa_1} & \text{, for membrane forces} \end{cases} \tag{151}$$

This means that for large Capillary numbers, the force caused by the cell is irrelevant, and it gets freely advected. Consequently, it experiences a large deformation. For small Capillary numbers on the other hand, the term becomes dominant, the cell deforms little, and the flow is forced to conform to the cell. This makes this number useful to judge expected deformation. However, slightly diverting from this typical definition, this thesis defines the Capillary number Ca_K as follows.

$$Ca_\mathrm{K} = \sqrt{\frac{8}{1+\nu}}\sqrt{Ca} \tag{152}$$

This choice is very sensible as is explained in appendix G. The prefactor assures, that as a rule of thumb, a large Capillary number starts around $Ca_\mathrm{K} > 1$. Taking the root of the Capillary number is the conceptually important part. For volumetric forces, the Capillary number simplifies to the following using the Young's modulus E.

$$Ca_\mathrm{K} = 4\sqrt{\frac{\mu_\mathrm{t}\dot{\gamma}_\mathrm{t}}{E}} \tag{153}$$

The Capillary number can be used for scaling. This will be discussed next.

5.5. Scaling of elastic models

Scaling is concept most often used together with dimensionless numbers. Due to their derivation, one can see, that the simulated physics are not altered as long as the dimensionless numbers stay the same. This means, that one would get the same simulation result for volumetric cells, if one would have both the shear-rate $\dot{\gamma}$ and the Young's modulus E. This transformation does not change the Capillary number. This is often useful to make parameters more convenient. Care has to be taken to not accidentally change another dimensionless number in the process. Another type of scaling is the analysis of the behavior of a system in regard to the change of a parameter. In this thesis, the elastic force density $\vec{f}$, which is defined as follows is of interest.

$$f_i = -\frac{\partial U}{\partial u_i} \tag{154}$$

The energy densities listed in this section are proportional to a modulus and otherwise dimensionless. For the volumetric models, the Young's modulus can be factored out, for the membrane models an equivalent modulus exists for each one. This means, that the energy density only reacts linearly to scaling of the Young's modulus (or equivalent) and is otherwise invariant under any scaling. The force density is defined as a derivative of the energy density in respect to the displacement. Considering a cell with scaled size and the same shape, this displacement would scale according to this size scaling factor. If one considers the force a volume for the volumetric models or a surface for the membrane models has to be multiplied onto the force density. This factor would scale with a power of three or two of the size scaling factor respectively. In total, the force increases with the square of the size scaling for volumetric models and linearly for the membrane models. This means, that scaling the size, while preserving the physics would require to scale the Young's modulus or its volumetric counterpart as well. The moduli would need different scaling in order to preserve the forces relative strength. However, in a case of a simple system, the force response of a cell of a given size can easily be determined from the simulation of a cell of a different size. Next, the implementation of the models presented in this section and their use shall be discussed.

6. Immersed Boundary Method

The Immersed Boundary Method (IBM) is a finite element algorithm originally designed to allow the simulation of membranes immersed in a fluid. It has originally been developed independently of the LBM algorithm [83, 84]. Due to the popularity of the method, modern literature on the subject is plentiful [9]. The basic idea is, that on the membrane a no-slip condition must be fulfilled, meaning that the membrane must rest in the fluid. This is archived, by treating the membrane as part of the fluid. However, previous research [70] has demonstrated, that the underlying principles can be extended to more general use-cases. The following discussion will therefore be from a more general standpoint. In this framing, IBM allows the simulation of the behavior of point-particles in a flow. In the following, the discretization of cells into meshes and the coordinate system used for IBM are covered, before covering the algorithm, the relevant forces and some limitations of IBM.

6.1. Meshes

In order to simulate an object like a cell, it needs to be subdivided into point-particles and appropriate inter-particle forces. These forces can sensibly be defined on surface and volume elements. Therefore, it is indicated to subdivide the cell into such elements. In this thesis, the surface is discretized by splitting it into triangles. The vertices of the triangles are the aforementioned point-particles. For cells only consisting of a membrane and some internal fluid, like red blood cells (RBCs), this is enough. For more general cases, the cells volume is subdivided into tetrahedra. The vertices of the tetrahedra are also part of the aforementioned point-particles. This thesis focuses on such volumetric cells. Generating such meshes is done differently depending on the mesh type. The membrane-only meshes, termed RBC meshes in the following are generated using an icosahedron subdivision method [79] to create a sphere. This sphere is subsequently deformed into the typical discocyte shape. The volumetric meshes are generated using gmsh [85]. It shall be noted, that the meshes produced by gmsh will differ slightly between versions and even between computers. In this thesis gmsh is generally instructed to produce a spherical mesh with a fixed radius R and an average distance between the points of 1. With this R is a measure of the mesh resolution and is treated as such in this thesis. The average distance of 1 is treated as being in lattice unit, meaning the mesh has an average spacing equal to the lattice unit. Picking the spacing (exactly) equal to the lattice constant is necessary according to literature [9, 86], but can be interpreted more loosely under some conditions (see appendix H). This thesis generally adheres to this convention (see appendix H for details). A notable exception being the prevention of a certain instability (see section 15). *FluidX3D* ships with four standard meshes, which are used extensively throughout this thesis. The first one is the membrane-only mesh in a discocyte shape. This is useful for modeling RBCs. It can be seen in figure 18.

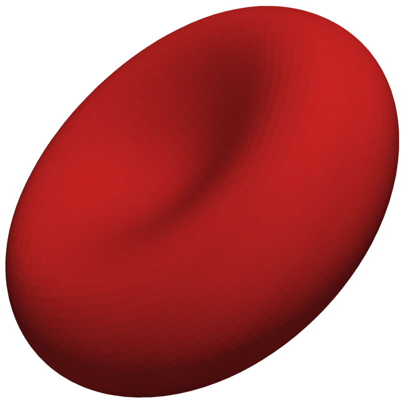


Figure 18: Render of the discocyte shaped, membrane-only mesh used for RBCs.

The other three meshes are a volumetric sphere in three sensible standard resolutions. These can be seen in figure 19.

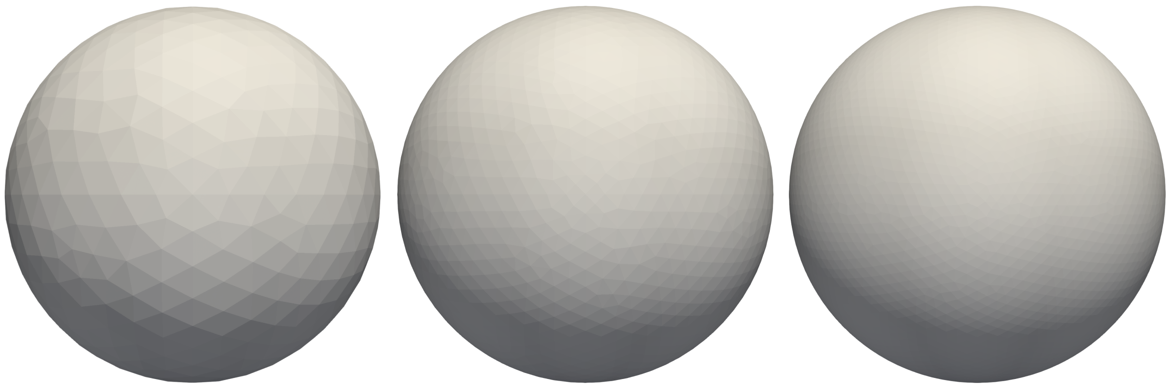


Figure 19: Render of volumetric meshes in the three standard resolutions ($R = 6$, $R = 12$ and $R = 18$).

Note, that the compute burden of the IBM algorithm is small compared to LBM. However, if one picks the average distance of IBM nodes as the lattice constant, the LBM resolution has to increase with increasing IBM resolution. With this in mind, higher resolution meshes are used in simulations with longer runtime. The small ($R = 6$) mesh is prone to error, but useful for quick tests, with small runtime. The medium ($R = 12$) mesh is typically good enough for most applications. The large ($R_1 8$) mesh is used if minimal errors are desired. Simulations using it are often quite intense in terms of compute. Throughout this thesis, meshes with different sizes are used as well from time to time. These are generated using a gmsh configuration file and the following command.

```
gmsh mesh.geo -3 -smooth 100 -optimize_netgen \
-algo hxt -format vtk -o mesh.vtk
```

Those meshes are addressed by their resolution R. As the average distance between points in the mesh is 1 and chosen to be equal to the lattice constant, the coordinates in the mesh are compatible with lattice coordinates. This allows easy loading of the meshes into *FluidX3D*. However, the IBM points do not lie on the lattice nodes, but in between. This means, that their coordinates are not integers. Formally, this necessitates the definition of a second coordinate system. This will be covered next.

6.2. IBM coordinates

The second coordinate system present in *FluidX3D* aside from the LBM coordinate system (see section 2.4) is the IBM coordinate system. The IBM coordinate system is also Cartesian. It is the one the user is usually confronted with. Its purpose is to handle the positions of the IBM mesh, which do not align with the LBM grid. Its origin is centered in the simulation volume. This has proven advantageous as many problems are in some way symmetric. Consider a lattice grid made of $L_x \times L_y \times L_z$ lattice nodes. In LU, the simulation volume in the IBM coordinate system extends from $\left(-\frac{L_x}{2}, -\frac{L_y}{2}, -\frac{L_z}{2}\right)$ to $\left(\frac{L_x}{2}, \frac{L_y}{2}, \frac{L_z}{2}\right)$. Lattice coordinates can therefore be converted to IBM coordinates using the following equation.

$$\vec{x}_{\text{IBM}} = \vec{x}_{\text{LBM}} - \frac{1}{2}\begin{pmatrix} L_x - 1 \\ L_y - 1 \\ L_z - 1 \end{pmatrix} \Delta x \tag{155}$$

Here, Δx is the lattice constant. These coordinates can also easily be converted to SI units and convey meaningful information. Simulation output in *FluidX3D* is always written in this coordinate system. This includes the lattice associate data. The LBM coordinate system is truly only used for some algorithms, that require it and for the definition of geometries. Work is done to also translate the latter to IBM coordinates wherever possible. With the basics covered, the algorithm can be described. This is done next.

6.3. IBM algorithm

The base working principle of the IBM algorithm is, that the actual solving is done by the LBM algorithm. There is no direct solver in the IBM algorithm. The influence of the immersed points is handed of to the fluid solver, and the results are passed back. The IBM algorithm can be broken into the following five steps.

- Force calculation
- Force spreading
- LBM step
- Velocity interpolation

- Advection

Note, that these can be cyclically reordered. *FluidX3D* actually starts with velocity interpolation, in order to separate IBM code from LBM code. These steps in their intuitive order will be discussed in the following. Figure 20 shows an illustration of the principal components of this algorithm.

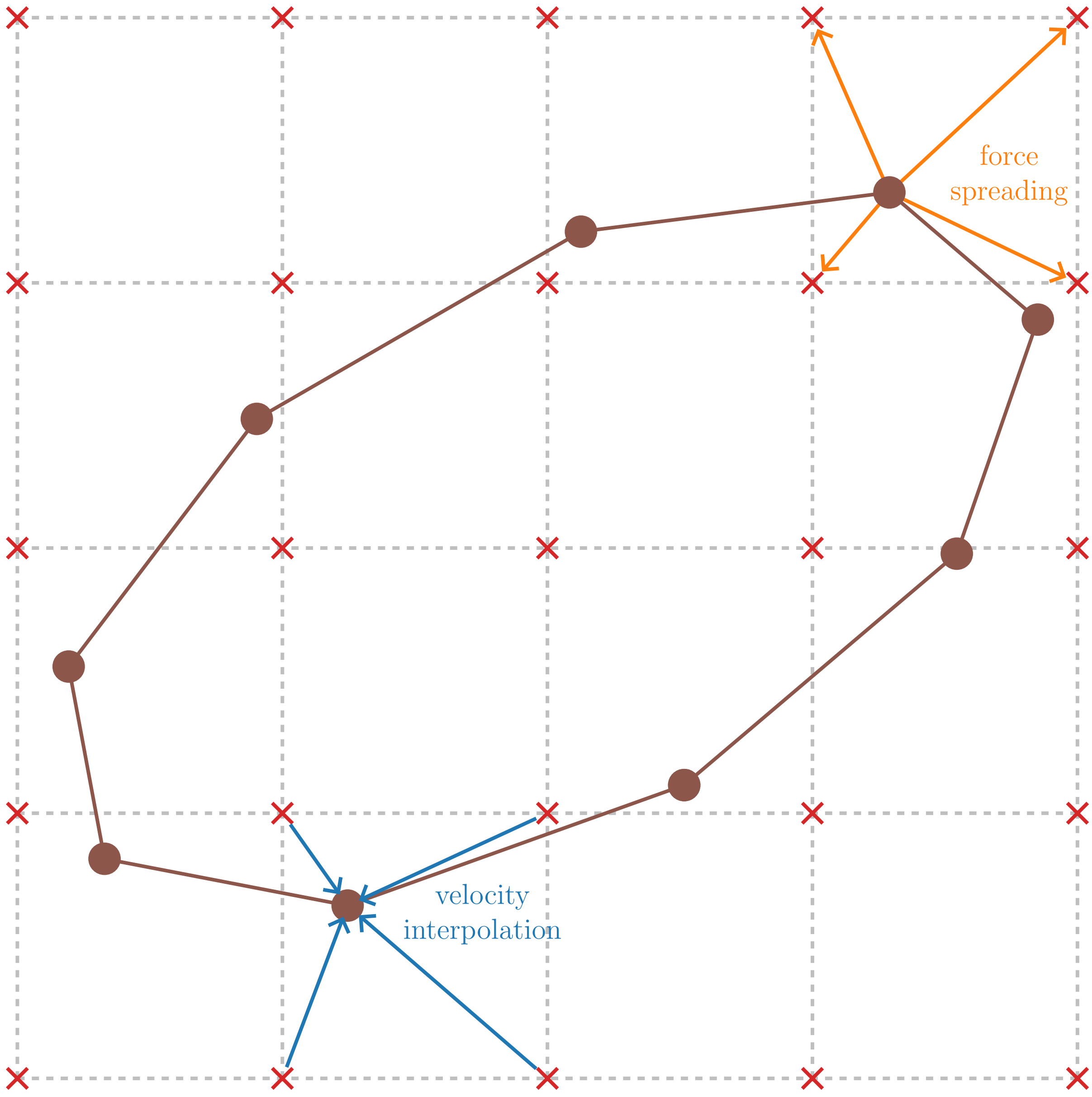


Figure 20: Illustration of the IBM algorithm (in 2D for visibility). A deformed mesh lies in a grid of lattice nodes. The force on any given point of the mesh is distributed to the adjacent lattice nodes. Similarly, the velocity for any given point of the mesh is calculated by interpolating from the adjacent lattice nodes.

6.3.1. Force calculation

The points on the mesh are exposed to different kinds of forces. For one, there are external forces, for example gravity. These in general require the point to be associated with some amount of mass. If the points are part of a membrane-only mesh, this is tricky. However, the only case where such external forces are relevant in this is the compression of volumetric cells (see section 7). In these cases, the force only has to be large and is not required to fulfill any quantitative criteria. The other type of force is internal forces. The discretization into a mesh allows any complicated macroscopic strain to be split into a well-defined deformation of a triangle or tetrahedron. This local strain is simple enough to solve the material models and calculate a local stress. This gives the forces, that if put together reproduce the macroscopic behavior of the cell. The equation for the forces relevant to this thesis can be found in subsection 6.4. It should be noted, that due to these forces being per triangle or tetrahedron respectively, multiple contributions need to be added for any mesh point. As *FluidX3D* runs in parallel on GPUs, this is an issue as it can lead to race conditions. This is handled (see appendix I) with a workaround. Even with the (currently not available) ideal solution, the order of the additions is not deterministic. This causes some noise it the simulation output depending on how the rounding errors compound. With these forces on the immersed points determined, they need to be handed of to the LBM algorithm. This is explained next.

6.3.2. Force spreading

From the previous step, the force on each point of the mesh is known. This needs to be transferred to the fluid. The fluid acts on the LBM nodes. Therefore, the force of a mesh point is distributed among its neighboring lattice points. To do this, first, the position of the IBM point is converted to LBM coordinates as follows.

$$\vec{x}'_{\text{LBM}} = \vec{x}_{\text{IBM}} + \frac{1}{2}\begin{pmatrix} L_x - 1 \\ L_y - 1 \\ L_z - 1 \end{pmatrix}\Delta x + \begin{pmatrix} L_x \\ L_y \\ L_z \end{pmatrix}\Delta x = \vec{x}_{\text{LBM}} + \begin{pmatrix} L_x \\ L_y \\ L_z \end{pmatrix}\Delta x \tag{156}$$

Here, Δx is the lattice constant. Note, that compared to the simple inverse of the transformation described in subsection 6.2, an extra offset is added. The reason for this is, that the simulation volume extends towards the negative x, y and z direction from the LBM node with coordinate $(0, 0, 0)$. The LBM coordinates are used as array indices and therefore are designed to be positive. This extra term assures this. Furthermore, *FluidX3D* always has periodic boundaries (see section 13 for other boundaries) and therefore, this extra term can always be removed using a modulus operation. The next step is a modulus operation as well. The position $\vec{p}$ of the IBM point relative to its neighboring LBM point with the smallest LBM coordinate is determined as follows.

$$\vec{p} = \vec{x}'_{\text{LBM}} \,\%\, \Delta x \tag{157}$$

Note, that $\cdot \,\%\, \cdot$ denotes the modulo operation. Similarly, said neighboring LBM point with the smallest LBM coordinate $\vec{x}_0$ is defined as follows.

$$(\vec{x}_0)_i = (\vec{x}'_{\mathrm{LBM}})_i \,\%\, (L_i \Delta x) \tag{158}$$

With this, the force on the surrounding LBM nodes $\vec{F}_{\mathrm{LBM}}$ can finally be calculated as follows.

$$\vec{F}_{\mathrm{LBM}}\left(\vec{x}_0 + \begin{pmatrix} i \\ j \\ k \end{pmatrix} \Delta x\right) = \frac{1}{\Delta x^3} \prod_m \left\{ \Delta x - \left| \left[\vec{p} - \begin{pmatrix} i \\ j \\ k \end{pmatrix} \Delta x \right]_m \right| \right\} \vec{F}_{\mathrm{IBM}} \tag{159}$$

Here $\vec{F}_{\mathrm{IBM}}$ is the force on the IBM point that needs to be spread. The indices $i, j, k \in \{0, 1\}$ in all their permutations are used to address all 8 neighboring points. While this looks complicated when written as an equation, it is actually quite simple. This can be seen in figure 21.

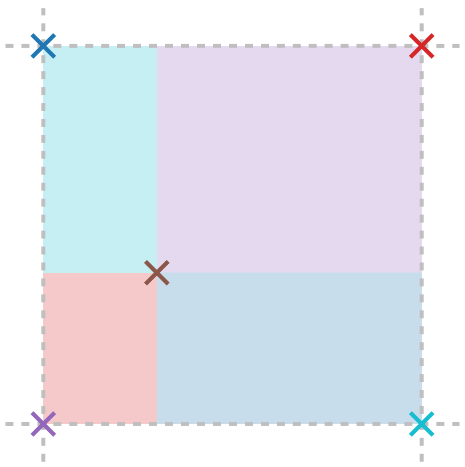


Figure 21: Illustration of the force spreading algorithm (in 2D for visibility). The surface between four latttice nodes is split into four sub-surfaces by the position $\vec{p}$ of the IBM point. Each latttice node receives the part of the force, that is proportional to the area, which is opposite to itself.

The IBM point splits the volume into eight sub-volumes proportional to the share of the force the opposite lattice node receives. The sub-volume is taken as a fraction of the entire volume enclosed by the adjacent lattice nodes. Here, it is also the case, that multiple threads may write to the same lattice node. It is handled in the same way as in the previous subsection (see appendix I). With the forces distributed to the lattice nodes, they can be handled by the LBM algorithm as described in section 2.5.1. Note, that the LBM algorithm technically deals with force densities. The conversion is done, via the interpretation, that the lattice node represents a Δx^3 volume centered around it. The LBM algorithm results in a velocity defined at the lattice nodes. This is used in the next step.

6.3.3. Velocity interpolation

To get the velocity at the IBM point $\vec{u}_{\mathrm{IBM}}$, trilinear interpolation is performed from the velocity at the adjacent LBM nodes $\vec{u}_{\mathrm{LBM}}$. This is described by the following equation.

$$\vec{u}_{\mathrm{IBM}} = \sum_{ijk} \frac{1}{\Delta x^3} \prod_{m} \left\{ \Delta x - \left| \left[\vec{p} - \begin{pmatrix} i \\ j \\ k \end{pmatrix} \Delta x \right]_m \right| \right\} \vec{u}_{\mathrm{LBM}} \left(\vec{x}_0 + \begin{pmatrix} i \\ j \\ k \end{pmatrix} \Delta x \right) \tag{160}$$

Conceptually, this is the same as in the previous section, just the other way round. The velocities at the lattice nodes get weighted with the fraction of the volume opposite to the lattice node. All eight such contributions get summed. This is the fluid velocity at the position of the IBM point. The next step moves the point.

6.3.4. Advection

The base concept of IBM is, that the point has no slip. It moves with the fluid. Consequently, the point is advected with the local fluid velocity determined in the previous step. A simple Euler integration is used for this. The LBM is responsible for the higher order accuracy. With the algorithms sorted, the next section discusses the forces, that are used in this thesis with this algorithm.

6.4. IBM forces

In this thesis, IBM is used to simulate biological cells or hydrogel beads. To do this, like before (section 5), the models discussed should be split in forces modeling a membrane and forces modeling the bulk material of a cell. These forces can be found in literature already. However, they are typically denoted as needlessly complicated strings of scalar equations. Writing them as matrix valued equations is entirely possible and a lot simpler. Also, this speeds up computation considerably due to improved cashing. In a matter of speaking, the flattened equations are so complicated even the compiler is confused. Some derivation is required to gather a useful form. In this thesis, the forces are derived from the principle of virtual work. This means, they are a result of the derivative of the energy in respect to the deformation. Ultimately a definition of the deformation gradient tensor $\underline{F}$ (section 5.1) on the discrete mesh is required for the calculations. In the end, the forces on the vertices will be retrieved as the rows of a force tensor $\underline{M}_{\mathrm{Force}}^{\mathrm{T}}$. Note, that an alternative derivation using stresses exists. The result is identical [68][25]. In the following, membrane forces will be discussed first. Due to their limited importance in this thesis, the discussion is kept short.

6.4.1. Membrane forces

Membrane forces act on triangles or groups of triangles. The ones implemented in *FluidX3D* are shear stress, bending resistance, area and volume conservation and surface

[25]The Mooney-Rivlin code for *FluidX3D* was originally in this form and was later seamlessly transformed into the force form.

tension. They are listed in this order in the following.

6.4.1.1. Shear stress (Skalak/Neo-Hookean)

Due to limited importance for this thesis, the derivation of this model was skipped. The following is just an improved notation for the derivations found in literature [68, 76]. First, consider a triangle as can be seen in figure 22.

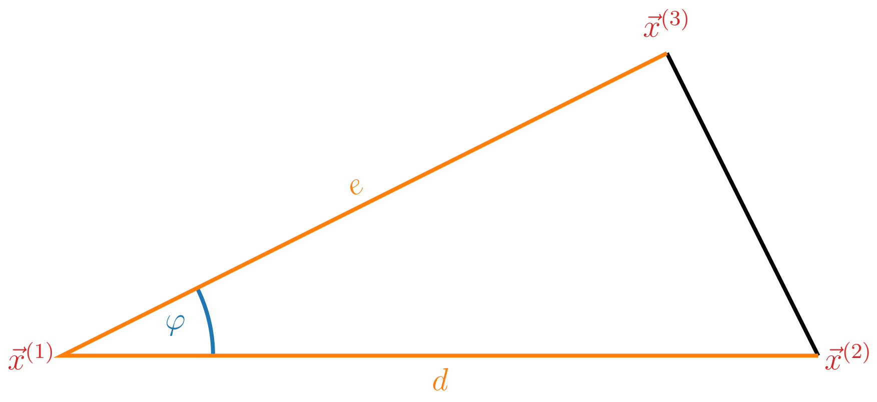


Figure 22: Illustration of a triangle the shear force is calculated on. It has the vertices $\vec{x}^{(i)}$, the sides d and e as well as the angle φ.

The deformation is easiest to describe if one considers the triangle in a coordinate system based on the triangle itself. Consider the reference triangle to be rotated so it lies in the same plane as the deformed triangle. The points $\vec{x}^{(1)}$ of both coincide and the sides d of both are aligned. Deformation should not depend on rotation. Rotating a priori, takes care of this. A local coordinate system is defined with the x-axis parallel to $\vec{x}^{(2)} - \vec{x}^{(1)}$ and the y-axis normal to the x-axis and in the plane of the triangle towards $\vec{x}^{(3)}$. The deformation $\vec{u}^{(i)}$ for the vertex i can then be defined in this coordinate system as follows.

$$\vec{u}^{(i)} = \vec{x}^{(i)} - \vec{x}_0^{(i)} \tag{161}$$

Quantities with an index 0 refer to the undeformed state. This immediately gives the following relations by definition.

$$\vec{u}^{(1)} = \vec{0} \tag{162}$$
$$u_x^{(2)} = 0 \tag{163}$$

The deformation gradient tensor $\underline{F}$ can then be written as a matrix product as follows.

$$\underline{F} = \begin{pmatrix} \vec{u}^{(1)} & \vec{u}^{(2)} & \vec{u}^{(3)} \end{pmatrix} \begin{pmatrix} \vec{a} & \vec{b} \end{pmatrix} + \mathbb{1} \tag{164}$$

Here, $\vec{a}$ and $\vec{b}$ are interpolation weights defined as follows.

$$\vec{a} = -\frac{e_0 \sin(\varphi_0)}{2A_0} \begin{pmatrix} 1 \\ -1 \\ 0 \end{pmatrix} = -\frac{1}{d_0} \begin{pmatrix} 1 \\ -1 \\ 0 \end{pmatrix} \tag{165}$$

$$\vec{b} = \frac{1}{2A_0} \left[e_0 \cos(\varphi_0) \begin{pmatrix} 1 \\ -1 \\ 0 \end{pmatrix} + d_0 \begin{pmatrix} -1 \\ 0 \\ 1 \end{pmatrix} \right] \tag{166}$$

Here A_0 is the area of the triangle. It is defined as follows.

$$A_0 = \frac{1}{2} d_0 e_0 \sin(\varphi_0) \tag{167}$$

This definition is more complex than necessary, given $\underline{F}$ only has three non-vanishing entries, two of which are quite simple. However, this form allows one to see the parallels to the definition of $\underline{F}$ in section 5.1, where the $\mathbb{1}$ could be removed by switching $\vec{u}$ for the new coordinates. This is possible here as well, and one can write the following.

$$\underline{F} = \begin{pmatrix} \vec{x}^{(1)} & \vec{x}^{(2)} & \vec{x}^{(3)} \end{pmatrix} \underline{X}_{\text{inv}} \tag{168}$$

With the matrix $\underline{X}_{\text{inv}}$ purely defined by the reference positions as follows.

$$\underline{X}_{\text{inv}} = \begin{pmatrix} g & h \\ \frac{1}{\left(\vec{x}_0^{(2)}\right)_x} & -\frac{\left(\vec{x}_0^{(3)}\right)_x}{\left(\vec{x}_0^{(2)}\right)_x \left(\vec{x}_0^{(3)}\right)_y} \\ 0 & \frac{1}{\left(\vec{x}_0^{(3)}\right)_y} \end{pmatrix} = \begin{pmatrix} \vec{a} & \vec{b} \end{pmatrix} \tag{169}$$

The parameters g and h are actually arbitrary due to the choice of coordinate system. The reason for the index of $\underline{X}_{\text{inv}}$ is the following relation.

$$\begin{pmatrix} \vec{x}_0^{(1)} & \vec{x}_0^{(2)} & \vec{x}_0^{(3)} \end{pmatrix} \underline{X}_{\text{inv}} = \mathbb{1} \tag{170}$$

It is not a true inverse, because it is not square and the above equation is not commutative. However, this way of defining $\underline{F}$ is a general concept, that will return for the volumetric models. With $\underline{F}$ defined, the force tensor $\underline{M}^{\text{T}}_{\text{Force}}$ for the Skalak model it reads as follows.

$$\underline{M}^{\text{T}}_{\text{Force}} = -\frac{A_0}{3} \underline{X}_{\text{inv}} \left[\kappa_1 (I_1 + 1) \underline{F}^{\text{T}} + (-\kappa_1 + \kappa_2 I_2) J^2 \underline{F}^{-1} \right] \begin{pmatrix} \hat{e}_x^{\text{T}} & \hat{e}_y^{\text{T}} \end{pmatrix} \tag{171}$$

Note the definition from section 5.2.1 reiterated in the following.

$$J = \det(\underline{F}) \tag{172}$$

$$\underline{B} = \underline{F}\,\underline{F}^{\text{T}} \tag{173}$$

$$I_1 = \text{Tr}(\underline{B}) - 2 \tag{174}$$

$$I_2 = J^2 - 1 \tag{175}$$

κ_1 and κ_2 are two moduli. κ_1 relates to shear, while κ_2 relates to area conservation. The forces on the individual vertices are given by the rows of $\underline{M}^{\mathrm{T}}_{\mathrm{Force}}$. The direction vectors $\hat{e}_x$ and $\hat{e}_y$ are defined in the unaltered coordinate system as follows.

$$\hat{e}_x = \hat{N}\big(\vec{x}^{(2)} - \vec{x}^{(1)}\big) \tag{176}$$

$$\hat{e}_z = \hat{N}\big[\big(\vec{x}^{(2)} - \vec{x}^{(1)}\big) \times \big(\vec{x}^{(3)} - \vec{x}^{(1)}\big)\big] \tag{177}$$

$$\hat{e}_y = \hat{e}_z \times \hat{e}_x \tag{178}$$

Where the normalization operator $\hat{N}\cdot$ shall be defined as follows.

$$\hat{N}\vec{a} = \frac{\vec{a}}{|\vec{a}|} \tag{179}$$

The definition of these direction vectors is the transform back in the original coordinate system. The Neo-Hookean is slightly simpler. Its force reads as follows.

$$\underline{M}^{\mathrm{T}}_{\mathrm{Force}} = -\frac{A_0}{3}\kappa_1 \underline{X}_{\mathrm{inv}} \left[\underline{F}^{\mathrm{T}} - \frac{1}{J^2}\underline{F}^{-1}\right]\begin{pmatrix}\hat{e}_x^{\mathrm{T}} & \hat{e}_y^{\mathrm{T}}\end{pmatrix} \tag{180}$$

It is enough to calculate the force on two points, as the sum of the forces has to vanish. With this, the shear models have been brought into a nice and manageable form. Next, the bending forces are discussed.

6.4.1.2. Bending resistance

The force density $\vec{f}$, generated using the principal of virtual work, can be written as follows [79, 80].

$$\vec{f} = 2\kappa_{\mathrm{B}}\big[\Delta_S(H - H_0) + 2(H - H_0)\big(H^2 - K + H_0 H\big)\big]\hat{n} \tag{181}$$

Here κ_{B} is the bending modulus, H and H_0 are the mean curvature and its reference, K is the Gaussian curvature, and $\hat{n}$ is the unit normal of the bent surface. The Laplace-Beltrami operator Δ_S is the Laplace operator in respect to the surface. Calculating the curvature on a discrete mesh is not trivial. There is also no particularly nice way to write it down. For the implementation the algorithm proposed by Meyer *et al.* [87] is employed. Due to the limited importance for this thesis, writing it down here is skipped. It should be noted, that even the most uniform meshes have special points. These might for example be points with a different number of neighbors than the other points. Such points introduce small error terms into the simulation [79]. This force was used in an attempt to combat the instability discussed in section 15.2. For a relevant impact, unreasonably high κ_{B} are required. The instability was addressed effectively using a different approach. Next, area and volume conservation will be discussed.

6.4.1.3. Area/Volume conservation

Physically speaking, both the surface area and the volume of a cell should be conserved. The arguments for this are quite simple. The membrane is made of a fixed number of lipid molecules arranged in a particular pattern (assuming the cell does not have enough time to grow). As neither the arrangement nor the size of the lipid molecule can change, the surface are is constant. Furthermore, the membrane is (mostly) impermeable and its content (mostly water) is incompressible. Consequently, the volume has to stay the same. The Skalak shear models does preserve area, while the Neo-Hookean model does not. Neither of them considers volume conservation. Technically, one of the implementations should assure volume conservation (see section 6), however this is not exact. To assure the cell does accurately conserve area and volume, additional forces are introduced. For the computation, the surface is discretized into triangles. For each surface triangle consisting of the vertices $\vec{x}_i$, the area conservation force on the first point $\vec{F}_1^{\mathrm{A}}$ is defined as follows [88].

$$\vec{F}_1^{\mathrm{A}} = -\frac{\kappa_{\mathrm{A}}}{2}\frac{A - A_0}{A_0}\hat{n} \times (\vec{x}_3 - \vec{x}_2) \tag{182}$$

Here κ_{A} is the area modulus, A and A_0 are the surface area of the cell and its reference respectively and $\hat{n}$ is the unit normal vector. The forces on the other points are given by cyclical permutation. The volume conservation force on the first point $\vec{F}_1^{\mathrm{V}}$ is defined as follows [88].

$$\vec{F}_1^{\mathrm{V}} = -\frac{\kappa_{\mathrm{V}}}{6}\frac{V - V_0}{V_0}(\vec{x}_3 \times \vec{x}_2) \tag{183}$$

Here κ_{V} is the volume modulus and V and V_0 are the volume of the cell and its reference respectively. The forces on the other points are given by cyclical permutation. Note, that the positions here are defined in relation to the center of the cell. Next, the surface tension will be discussed.

6.4.1.4. Surface tension

As previously discussed, the bending force was used to try to combat an instability (see section 15.2). Due to this being unsuccessful, a simple surface tension was implemented. This has no other use in this thesis, but is valid in general and therefore was retained, despite also being unsuccessful. For the computation, the surface is discretized into triangles. For each surface triangle consisting of the vertices $\vec{x}_i$, the surface tension force on the first point $\vec{F}_1^{\mathrm{S}}$ is defined as follows

$$\vec{F}_1^{\mathrm{S}} = -\frac{\kappa_{\mathrm{S}}}{2}\hat{n} \times (\vec{x}_3 - \vec{x}_2) \tag{184}$$

Here κ_{S} is the surface tension modulus, that represents the surface tension. The forces on the other points are given by cyclical permutation. This force aims to minimize the surface are and is the same as the surface conservation force (eq. (182)) for a reference area $A_0 = 0$. In order to archive the desired effect on the cell, values of the modulus are required, which lead to significant shrinkage. This is another opportunity to discuss whether, a cell at rest is stress-free. One could argue, that the actual reference size of the cell is bigger than observed, and it has been squished by surface tension. As the surface tension is dependent on the cell's affinity towards water, this value is unknown. Therefore, picking realistic parameters for such a simulation is not possible. Consequently, this avenue of though was not explored further.

6.4.2. Volumetric forces

For the volumetric forces, the volume is subdivided into tetrahedrons. The points indices are chosen to run from 0 to 3. The whole derivation, which can be found in appendix J deals with the points 1 to 3. The force on the remaining point is calculated using the fact, that the sum of the forces has to vanish. Consequently, the force tensor $\underline{M}^{\mathrm{T}}_{\mathrm{Force}}$ contains three forces in its rows. As alluded to before, the deformation gradient tensor $\underline{F}$ is defined as follows for all volumetric models.

$$\underline{F} = \underline{R}\underline{R}_0^{-1} \tag{185}$$

Here $\underline{R}$ is a tensor containing the current positions in its columns and $\underline{R}_0^{-1}$ is the inverse of the tensor containing the reference positions in its columns. The invariants from section 5.3 are relevant here as well. Some are reiterated in the following.

$$J = \det(\underline{F}) \tag{186}$$

$$\underline{B} = \underline{F}\underline{F}^{\mathrm{T}} \tag{187}$$

The others could be used to shorten some terms, but this has been omitted here. The moduli used in this section vary depending on convenience. The shear modulus G, the bulk modulus κ and Lamé's first parameter λ are used in the model definitions. They can be expressed through the Young's modulus E and the Poisson ratio ν as follows.

$$G = \frac{E}{2(1+\nu)} \tag{188}$$

$$\kappa = \frac{E}{3(1-2\nu)} \tag{189}$$

$$\lambda = \kappa - \frac{2}{3}G = \frac{E\nu}{(1+\nu)(1-2\nu)} \tag{190}$$

Both *FluidX3D* and this thesis use the Young's modulus and Poisson ratio as the parameters for the moduli. Other than that, no new variables are introduced. In the following, only the results of the derivations are listed. As before, the models are ordered from most linear, to leas linear. First, the linear elastic model is discussed.

6.4.2.1. Linear elastic

The linear elastic force reads as follows.

$$\underline{M}^{\mathrm{T}}_{\mathrm{Force}} = -V_0\underline{R}_0^{-1}\left[G\left(\underline{F}+\underline{F}^{\mathrm{T}}\right) + \left(\left(\kappa - \frac{2G}{3}\right)\mathrm{Tr}(\underline{F}) - 3\kappa\right)\mathbb{1}\right] \tag{191}$$

It should be reiterated, that a finite force is sufficient to deform a linear elastic tetrahedron to the point of vanishing. Note, that all the moduli in this equation are proportional to the Young's modulus. This means, that it could be factored out. With this, all the dimensional quantities are in front. The remaining bracket is dimensionless and as such indifferent to scaling. Next, the first production model will be discussed.

6.4.2.2. Modified Saint Venant–Kirchhoff

The modified Saint Venant–Kirchhoff force reads as follows.

$$\underline{M}^{\mathrm{T}}_{\mathrm{Force}} = -V_0 \underline{R}_0^{-1} \left\{ G(\underline{S} - \mathbb{1}) + \frac{\lambda}{2} \operatorname{Tr}(\underline{S} - \mathbb{1})\mathbb{1} \right\} \underline{S}^{-1} \underline{F}^{\mathrm{T}} \tag{192}$$

To find the tensor $\underline{S}$, the computation of a root of a tensor is required. This is discussed in appendix K. The same comments for vanishing deformation and scaling apply as for the linear elastic force. Next a less linear version of the Saint Venant–Kirchhoff model will be discussed.

6.4.2.3. Saint Venant–Kirchhoff

The Saint Venant–Kirchhoff force reads as follows.

$$\underline{M}^{\mathrm{T}}_{\mathrm{Force}} = -V_0 \underline{H} \{ G \underline{B}' + 2\lambda [\operatorname{Tr}(\underline{B}')] \mathbb{1} \} \tag{193}$$

$\underline{B}'$ and $\underline{H}$ are helper variables defined as follows.

$$\underline{B}' = \underline{B} - \mathbb{1} \tag{194}$$

$$\underline{H} = \underline{R}_0^{-1} \underline{F}^{\mathrm{T}} \tag{195}$$

The same comments for vanishing deformation and scaling apply as for the linear elastic force. Next the models capable of significant nonlinearity are covered.

6.4.2.4. Mooney–Rivlin

The Saint Mooney–Rivlin force reads as follows.

$$\begin{aligned} \underline{M}^{\mathrm{T}}_{\mathrm{Force}} = -V_0 J \underline{R}^{-1} \Bigg\{ & G_1 J^{-\frac{5}{3}} \left(\underline{B} - \frac{1}{3} \operatorname{Tr}(\underline{B}) \mathbb{1} \right) \\ & + G_2 J^{-\frac{7}{3}} \left\{ \left[\operatorname{Tr}(\underline{B}) \underline{B} - \underline{B}^2 \right] + \frac{1}{3} \left[\operatorname{Tr}\left(\underline{B}^2\right) - \operatorname{Tr}(\underline{B})^2 \right] \mathbb{1} \right\} \\ & + \kappa (J - 1) \mathbb{1} \} \end{aligned} \tag{196}$$

The shear modulus G is split as follows.

$$G_1 = wG \tag{197}$$

$$G_2 = (1 - w)G \tag{198}$$

This makes the non-linearity tuneable usig the Mooney-Rivlin ratio w. The Neo-Hookean limit is recovered for $w = 1$. For small deformations the Mooney-Rivlin model does act like a Neo-Hookean model with $G = G_1 + G_2$ as its shear modulus [70]. This model does not allow the tetrahedron to collapse. The comments regarding scaling apply as for the linear elastic force. Next some considerations in regard to scaling of these equations are discussed.

6.5. Scaling of elastic forces

As previously mentioned in section 5.5, the forces can be scaled. They are proportional to the Young's modulus for the volumetric forces or its membrane counterpart. Furthermore, the volumetric forces scale with the square of the size and the membrane forces scale linearly with the size. Departing from the abstract notation in the previous section, this can be seen here more clearly in the models' shared prefactor. It should be noted in that regard, that the inverse of a tensor for example $\underline{R}^{-1}$ behaves the same as the inverse of a scalar would. Meaning, in this case, this represents an inverse of the size of the cell. With the elastic behavior of cells thoroughly covered, next, their content is addressed.

6.6. Viscosity contrast

As mentioned previously (see section 4.3.2), the inside of a cell might have a different viscosity than the outside fluid. Especially for membrane-only cells this difference drives most of their behavior [68]. For this to work, the algorithm needs to know which LBM nodes are inside the cell. In *FluidX3D* the algorithms responsible for this functionality are marked by the flag `INOUT`. In principle, a ray-casting algorithm can be used for this. A ray is emitted by each lattice point in an arbitrary direction, and the number of intersections with the membrane are counted. From this one can determine whether a node is inside the cell, depending on whether the count is odd (inside) or even (outside). However, this is expensive. There are optimizations, where one moves through multiple LBM nodes with the same ray. Care has to be taken in the case of periodic boundaries. Furthermore, there is an implementation difficulty, where memory has to be reserved for the expected number of intersections. For optimal speed, ray-casting is only used for the initialization and from then only the required nodes are updated. A quick overview of the algorithm used in *FluidX3D* is given in the following. Publications exist on the implementation of this algorithm [65, 89]. However they do not describe the algorithm as it is actually implemented in its current state. After initial ray-casting only the points close to the surface of the mesh are considered. For each surface triangle, its center $\vec{p}$ and its normal vector $\hat{n}$ are determined. Only the LBM nodes $\vec{x}^{(i)}$ adjacent to $\vec{p}$ are considered, similar to the velocity interpolation and force spreading parts of the IBM algorithm discussed in subsection 6.3. This carries the assumption, that the mesh moves less than a lattice constant per time-step. This is assured in IBM, because violating this is only possible with an unstable Mach number. From this the direction from the triangles center to the LBM node $\vec{d}^{(i)}$ is determined as follows.

$$\vec{d}^{(i)} = \vec{x}^{(i)} - \vec{p} \tag{199}$$

Now one can sort the nodes using a scalar product as follows.

$$\vec{d}^{(i)} \cdot \hat{n} \begin{cases} < 0 & \text{, node } i \text{ inside} \\ \geq 0 & \text{, node } i \text{ outside} \end{cases} \tag{200}$$

This sorting is actually not a grantee, but refers to where the node likely is. Consequently, for some geometries this can sometimes lead to odd results like a single outside node being entirely surrounded by inside nodes. To prevent this usually an additional step is taken, where all nodes surrounded by a given threshold amount of the other type are flipped. This is not done in *FluidX3D* due to race condition concerns. Instead, two passes are made. In the first, only the adjacent lattice node with the smallest lattice coordinates is considered. In the second pass, all neighbors are considered, but the value is only updated if the nodes are outside. It should be noted, that this approach requires a triangle center to be near each lattice node. This means, that the average distance between IBM mesh points (see appendix H) may not be much larger than the lattice constant for this algorithm to work. Lastly, a few limitations of the IBM algorithm shall be covered.

6.7. Limitations

The following limitations are in no particular order. As has been noted above, the implementation deals with 3×3 matrices. The implementation of those relies on undocumented features of the OpenCL compiler. Changes to that code should only be made by qualified personnel. As has been mentioned multiple times already, most of the volumetric models allow the tetrahedron to fully collapse with a finite force. This problem will actually be observable later in the thesis (see section 20). The mesh resolution must be somewhat in tune with the LBM grid. This means, that the increasing the mesh resolution is very expensive in terms of compute even so the IBM algorithm is practically free compared to LBM. Even the medium $R = 12$ mash commonly used in this thesis is noticeably not completely round. Especially in cases of strong deformation a large resolution is required for accurate reproduction of the relevant effects. This is particularly an issue for localized deformations. IBM does not support meshes with higher local resolution well. Consider a very obtuse triangle in a membrane, that is larger than its equilibrium. This triangle should shrink. The surface conservation force will archive this by making the triangle even more obtuse. This can even cause the triangle to vanish because the point comes to coincide with the hypotenuse. The same is possible for tetrahedrons. This behavior is physically correct, but annoying in simulations. This does practically never happen for tetrahedrons and for triangles only in instances of very strong deformation. Also, this does not break the simulation, only the noise, which is present anyway due to rounding errors, is considerably amplified. The surface area and volume used for the respective conservation forces cannot easily be calculated in parallel. Consequently, this is done on the CPU. This necessitates the transfer of the IBM mesh over the PCIe bus for each time-step, which is not ideal for performance. There is a maximum deformation that can sensible be simulated, as the mesh is static and therefore in effect looses resolutions in areas of great stretch. Above this point accuracy suffers. This limit is reached for large deformations. Runtime can be improved by performing the calculations at lower precision [47]. However, especially for higher viscosity fluids IBM requires 64-bit precision (see appendix O). Even so the cells physically are incompressible, this is not possible to archive in simulations. The relevant limit $\nu \to \frac{1}{2}$ of the Poisson ratio would cause diverging forces. $\nu = 0.48$ is used as a reasonable compromise instead. This is the default for this entire thesis. Unless otherwise stated, the Poisson ratio is set to 0.48. IBM forces that are too large, cause LBM populations to become negative. This leads to unphysical behavior which is likely to crash the simulation. The same effect limits the maximum Young's modulus, that can be simulated. With the aforementioned compromise, the cells are mostly incompressible and producing forces of problematic magnitude is extremely unlikely. Due to the forces being handled by the LBM nodes and the IBM mesh having a similar size, the forces from multiple vertices of the same triangle or tetrahedron can be spread to the same lattice node. This can cause the force to partially compensate itself. This is an issue that does appear from time to time and probably contributes to the formation of one of the instabilities (see section 15.2). IBM can promote some instabilities (see section 15). Care needs to be taken to avoid these. This concludes the discussion of IBM. With this the cells are fully described, and they can be subjected to external forces. Next the relevant models for indentation experiments are discussed.

7. Modeling indentation experiments

Indentation experiments using atomic force microscopy (AFM) provide information about a material's elastic properties (e.g. its Young's modulus). The relevant equations will be introduced in this section. Parts of this section have already been published [2]. Another publication on this topic is being prepared as of the time of writing. An illustration of a generalized indentation experiment can be found in figure 23.

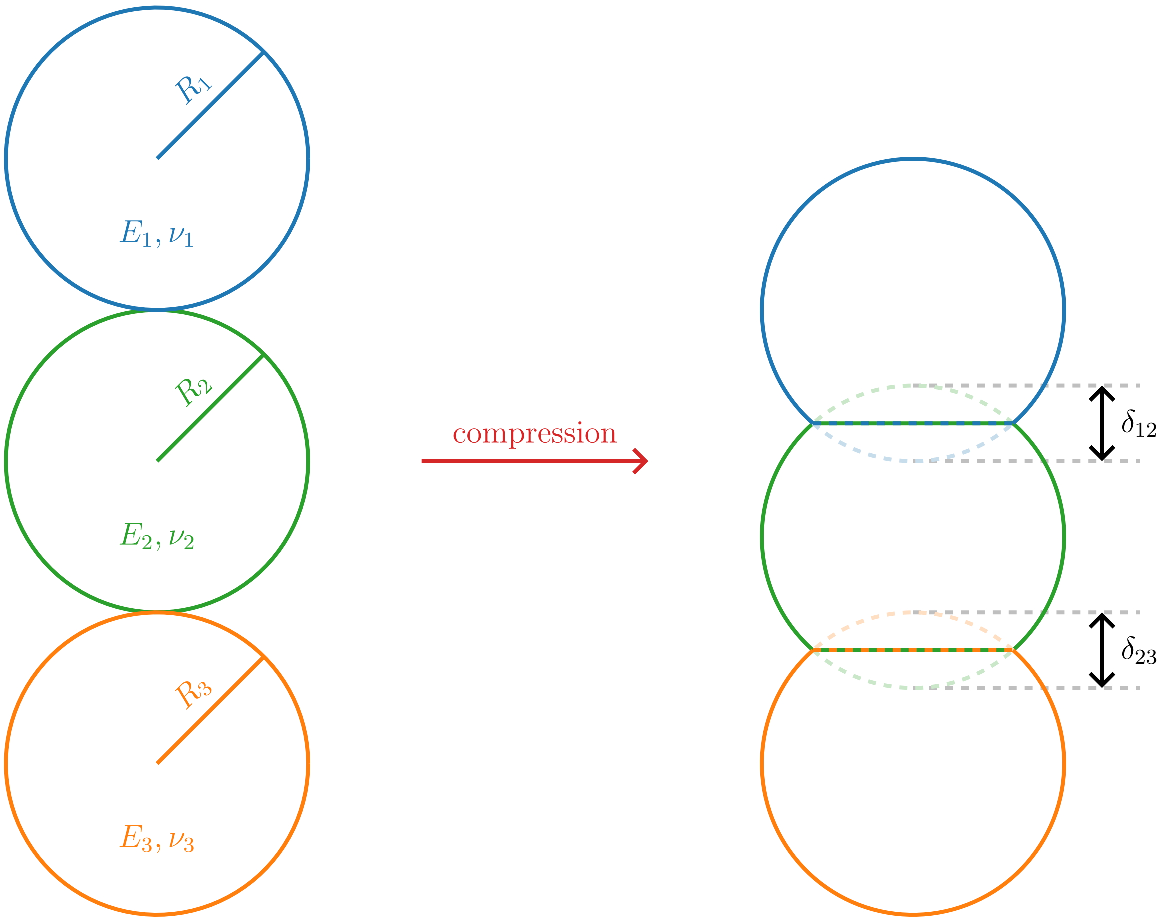


Figure 23: Illustration of a generalized indentation experiment. The cell is placed in between an indenter and a substrate. Each of them is modeled to be an elastic sphere with radius R_i, Young's modulus E_i and Poisson ratio ν_i. While the cell is compressed, indenter and substrate respectively move by δ_{12} and δ_{23} relative to the cell. The shape of the interface between the cell and the indenter or substrate is in general not flat and can vary.

In general, the radii differ significantly between indenter, cell and substrate, which is not displayed in the illustration. The following discussion is split in the importance of considering both contacts, conical indenters and the scaling behavior of models discussed. In the following, the relevant equations will be introduced first.

7.1. Double contact

A contact between two linear elastic spheres can be modeled as follows using the Hertz model [90, 91].

$$F = \frac{4}{3}\sqrt{R_{12}}E_{12}\delta_{12}^{\frac{3}{2}} \tag{201}$$

Here the indices refer to the interface of spere 1 and 2. The interfacial properties R_{12} and E_{12} are derived from the individual sphers' radius and Youngs's modulus as follows.

$$\frac{1}{R_{12}} = \frac{1}{R_1} + \frac{1}{R_2} \tag{202}$$

$$\frac{1}{E_{12}} = \frac{1-\nu_1^2}{E_1} + \frac{1-\nu_2^2}{E_2} \tag{203}$$

This models one of the two contacts in this geometry. It should be noted, that this is an approximation for small deformations δ. Larger deformations will be discussed in section 20. Neglecting the second contact is so common in literature, that some authors feel the need to point out, that the second contact has not been neglected. This is termed double contact model [2]. This thesis will stick with the term “Hertz model”, as the double contact model is nothing more than the correct use of the Hertz model. In order to deal with the second contact on first inverts the single contact model as follows.

$$\delta_{12} = \left(\frac{3}{4}\frac{F}{\sqrt{R_{12}}E_{12}}\right)^{\frac{2}{3}} \tag{204}$$

This can now be combined with the corresponding term for the second contact to the total deformation δ. This can be seen in the following.

$$\delta = \delta_{12} + \delta_{23} = \left(\frac{3}{4}\frac{F}{\sqrt{R_{12}}E_{12}}\right)^{\frac{2}{3}} + \left(\frac{3}{4}\frac{F}{\sqrt{R_{23}}E_{23}}\right)^{\frac{2}{3}} \tag{205}$$

$$= \left(\frac{3}{4}F\right)^{\frac{2}{3}}\left[\left(\frac{1}{\sqrt{R_{12}}E_{12}}\right)^{\frac{2}{3}} + \left(\frac{1}{\sqrt{R_{23}}E_{23}}\right)^{\frac{2}{3}}\right] \tag{206}$$

In a typical AFM experiment, the sample (the cell in this thesis) is significantly softer than both the indenter and the substrate. One can therefore perform the following approximation.

$$E_1 \gg E_2 \tag{207}$$

$$E_3 \gg E_2 \tag{208}$$

Consequently, the following holds.

$$E_{12} \approx \frac{E_2}{1-\nu_2^2} \tag{209}$$

$$E_{23} \approx \frac{E_2}{1-\nu_2^2} \tag{210}$$

Consequently, the Hertz model for two contacts can be written as follows.

$$\delta \approx \left(\frac{3}{4}\frac{1-\nu_2^2}{E_2}F\right)^{\frac{2}{3}}\left[\left(\frac{1}{R_{12}}\right)^{\frac{1}{3}}+\left(\frac{1}{R_{23}}\right)^{\frac{1}{3}}\right] \tag{211}$$

A simple case with great importance is the indenter and the substrate being flat plates. This is done using the following limit.

$$R_1 \to \infty \tag{212}$$
$$R_3 \to \infty \tag{213}$$

Consequently, the following holds in this limit.

$$R_{12} = R_2 \tag{214}$$
$$R_{23} = R_2 \tag{215}$$

With this, the Hertz model for two contacts takes a form similar to the model for a single contact. This can be seen in the following.

$$\delta \approx k\left(\frac{3}{4}\frac{1-\nu_2^2}{E_2\sqrt{R_2}}F\right)^{\frac{2}{3}} \tag{216}$$
$$F \approx \frac{4}{3}\frac{E_2\sqrt{R_2}}{1-\nu_2^2}\left(\frac{\delta}{k}\right)^{\frac{3}{2}} \tag{217}$$

The difference to the single contact is the factor $k = 2$. This is the form used most in this thesis. One can see, that if one were to miss the second contact, the solution would be off by a factor of 2 for this approximation. In arbitrary radii, the discrepancy factor k is as follows.

$$k = \frac{R_{12}^{\frac{1}{3}} + R_{23}^{\frac{1}{3}}}{R_{23}^{\frac{1}{3}}} = 1 + \left(\frac{R_{12}}{R_{23}}\right)^{\frac{1}{3}} \tag{218}$$

Assuming $R_1 \leq R_3$ one can also assert the following.

$$k \leq 2 \tag{219}$$

It should be noted, that the following approximation is valid.

$$R_1 \ll R_2 \tag{220}$$
$$R_1 \ll R_3 \tag{221}$$
$$k \approx 1 \tag{222}$$

This means, that in the case of very small indenters, the second contact may actually be omitted. Realistic spherical indenters are likely not this small. There are however conical indenters with a small tip. These will be discussed next.

7.2. Conical indenter

The force-deformation relations for a hard ($E_1 \gg E_2$) conical indenter are as follows [92].

$$F = \frac{2}{\pi}\frac{E_2}{1-\nu_2^2}\tan(\varphi)\delta_{12}^2 \tag{223}$$

$$\delta_{12} = \sqrt{\frac{\pi}{2}\frac{1-\nu_2^2}{E_2}\frac{F}{\tan(\varphi)}} \tag{224}$$

Here φ is half of the tip angle of this cone. Note, that there is a typo in the original paper. Evidently, this equation does not transform into the solution for a plate indenter for $\varphi \to \frac{\pi}{2}$ as it should. With some math a reasonable approximation can be found though. A discussion on this can be found in appendix L. Given, that δ_{12} diverges for $\varphi \to 0$, the second contact may be omitted for small α. With the forces modeled for the relevant tip shapes a small discussion on their scaling is warranted. This is done in the following.

7.3. Scaling behavior of elastic cells

In sections 5.5 and 6.5, the scaling of the elastic models and the forces they produce has been discussed. The elastic models converge to the linear elastic model in the limit of small deformations [91]. The Hertz model describes linear elastic solids for small deformations. Consequently, the scaling of the models must also be present in the Hertz model. In the hard indenter and substrate limit discussed above, the force is proportional to the Young's modulus. With a simple scaling in size, where the shape is preserved, R_{12}, R_{23}, δ_{12} and δ_{23} are proportional to the size scaling factor. Consequently, the Hertz model scales with the square of the size scaling factor. The equation for a conical indenter has its factors distributed differently, but exhibits the same scaling. This will be used in the following section, where the simulation techniques used for indentation experiments are discussed.

8. Simulations without a fluid

Indentation experiments using atomic force microscopy (AFM) are quasi-static. This means, that the force measures are taken on a relaxed cell. In this case, the viscosity does not contribute. Consequently, the simulation of the fluid can be fully omitted. As the simulation of a fluid using LBM is expensive, when possible, it was avoided. This section will elaborate how this was done. Parts of this section have already been published [2]. Another publication on this topic is being prepared as of the time of writing. The code used for these simulations was named *noFluidX3D* by Sebastian Wohlrab, who was originally tasked to create this for his master's thesis [93]. Details on it can be found in appendix A.2. In the following, the algorithm, the modeling of the indenters, vocabulary surrounding the indentation and some limits are discussed. The base simulation loop of *noFluidX3D* is discussed first.

8.1. Algorithm

The elastic forces are calculated the exact same way, using the same kernels as described in section 6. As mentioned before (see section 6.5), there is no need to simulate using the parameters from the experiment. Using a standard Young's modulus ($E = 1\,\mathrm{kPa}$ in this thesis) and a standard size ($R = 12.5\,\mu\mathrm{m}$ in this thesis) with subsequent scaling to the parameters in the experiment is valid. This is done in this thesis to avoid retuning the simulation relaxation. On each mesh point, an additional force modeling the indenter is added. This is described in the next subsection (see subsection 8.2). Instead of spreading the total force to a fluid as is done in IBM, the integration happens on the mesh points. For this dampened classical dynamics are employed. It should be noted, that due to the dampening, the trajectories of the mesh points are not accurately reproduced. The equilibrium is however unaffected, as the minimal energy the system relaxes to is unaltered. Also, the total force is measured at equilibrium. Therefore, the dampening does have no effect for the quasi-static AFM measurements discussed in this thesis. The dampening is introduced to aid in stability. This is required for large indentations with small indenters. The Velocity-Verlet algorithm [94] is used with the additional dampening parameter ξ. The undampened equations read as follows.

$$\vec{v}(t) = \vec{v}(t - \Delta t) + \frac{1}{2}[\vec{a}(t) + \vec{a}(t - \Delta t)]\Delta t \tag{225}$$

$$\vec{x}(t + \Delta t) = \vec{x}(t) + \vec{v}(t)\Delta t + \frac{1}{2}\vec{a}(t)\Delta t^2 \tag{226}$$

Here $\vec{x}$ is the position, $\vec{v}$ is the velocity, $\vec{a}$ is the acceleration and m is the mass of a given point on the mesh. This describes a step from time t to time $t + \Delta t$. It should be noted, that the mass cannot be determined accurately in these simulations. This only affects the trajectory as the conformation of minimal energy is independent of mass. Therefore, the following choice is made in dimensionless units for simplicity.

$$m = 1 \tag{227}$$

It should be noted, that these dimensionless units are not technically lattice units, as there is no lattice here. However, the same unit conversion as in *FluidX3D* is employed here for compatibility. Therefore, the units here are the same as lattice units. Furthermore, this choice to fix the mass in dimensionless units causes the trajectory to be dependent on the conversion factors. In effect, changing the resolution might affect stability. In order to help with stability, the dampening factor ξ is introduced as follows.

$$\vec{a} = \frac{1}{m}\vec{F} - \frac{\xi}{\Delta t}\vec{v} \tag{228}$$

Here $\vec{F}$ is the force. For the simulations in this thesis, relaxation akin to critical dampening is observed for $\xi = 0.005$. Consequently, this is the value used in this thesis. With this, the Velocity-Verlet equations read as follows.

$$\vec{v}(t) = \left\{\vec{v}(t-\Delta t)\left(1 - \frac{1}{2}\xi\right) + \frac{1}{2m}\left[\vec{F}(t) + \vec{F}(t-\Delta t)\right]\Delta t\right\}\frac{1}{1+\frac{1}{2}\xi} \tag{229}$$

$$\vec{x}(t+\Delta t) = \vec{x}(t) + \vec{v}(t)\left(1 - \frac{1}{2}\xi\right)\Delta t + \frac{1}{2m}\vec{F}(t)\Delta t^2 \tag{230}$$

The sum of the indenter force evaluated at the mesh point locations gives the result of the AFM simulations. The indenter potentials are covered in the following.

8.2. Indenter potentials

Both the indenter and the substrate are implemented as location dependent additional forces on the membrane forces. Technically, this could be written as a potential. The force equation can be seen in the following.

$$\vec{F} = F_0\hat{n}\frac{d}{\Delta x} \tag{231}$$

Here F_0 is a force constant, $\hat{n}$ is the normal vector of the indenter and d is the distance that the given point has traveled **into** the indenter. The distance is scaled using the conversion factor for length, meaning it is used in dimensionless units. The force constant needs to be big enough to effectively repel the mesh points even at large deformation, with the corresponding large restoring forces. It may not be too large, as this would cause instabilities. The concrete value is irrelevant. In dimensionless units $F_0 = 0.1$ has been used throughout this thesis as it yielded satisfactory results. If a point has not yet entered the indenter, it does not experience any force. The force grows with the penetration depth. The force may not abruptly start, as this results in instabilities. As the force grows linearly, the potential grows with the square of the distance. Higher order potentials did not improve simulation results. Giving the finite nature of the force modeling the indenter, the mesh points penetrate the indenter. This is discussed further in the following.

8.3. Nominal and real deformation

The cell existing inside the indenter is of course nonphysical. However, the observed penetration depth for realistic parameters was very small and thus can be neglected. It is slightly dependent on indentation depth, but stays small at all indentations explored in this thesis. This does however pose the question of how to measure the cell's deformation, as the indenter position is not the same as the position of the top of the cell. This thesis opts to distinguish the nominal and the real deformation. The real deformation is defined through the actual position of the cell, while the nominal position is the one given by the position of the indenter. The nominal deformation can be seen as the deformation intended by the simulation. It is near identical to the real deformation and is used throughout this thesis in its stead. The naming difference is purely to acknowledge this approximation. In the following some limitations are discussed.

8.4. Limitations

The use of a mesh requires it to be sufficiently fine. The average distance of the mesh nodes needs to be small compared to the bending radius of the tip of the indenter. For pointed indenters, this is extremely difficult. Locally increasing the mesh resolution is questionable in regard to the $m = 1$ approximation. Increasing the resolution of the entire cell is costly. For indenters, that are not flat horizontal plates, the system is inherently in an unstable state. The cell can slip out to the side to reduce its strain. To prevent this, tangential movements of the top and bottom of the cell are ignored, enforcing their position. The movement normal to the indenter and parallel to its movement are carried out as normal. Given, that no fluid is considered for AFM simulations, dynamic AFM cannot be modeled using this *noFluidX3D*. As was previously mentioned (see section 5.3 and section 6), some models allow for tetrahedrons to collapse under finite load. For a cell with a Young's modulus E of $E = 1\,\mathrm{kPa}$, this is the case beginning at $\left|\vec{F}\right| \approx 25\,\mathrm{nN}$. Experiments do exceed this value. Simulations using the affected models cannot. This concludes the implementation of AFM experiments. Next, the theoretical background of cells in pure shear-flows will be discussed.

9. Roscoe theory for deformed cells in shear-flow

If one puts a spherical cell into a pure-shear-flow, it deforms into an ellipsoid, aligns itself in relation to the flow and exhibits a rotation termed tank-treading. A quantitative description of these effects in steady state was derived by Roscoe [95] for Newtonian fluids. This is an important validation case for Newtonian fluids. For viscoelastic fluids, differences are reported that could not yet be fully explained [96]. To evaluate these, Roscoe theory is also needed as a baseline. The use of the equations is not straight-forward and there is a typo in the original publication. This warrants a short explanation, which is provided in the following. In the following, the parameters, the relevant equations and how to use them to find the geometry and tank-treading frequency for a given parameter set are covered. First all the relevant parameters are introduced.

9.1. Parameters

Consider a spherical cell of radius R, in a shear-flow with a shear-rate of $\frac{\partial u_x}{\partial y} = \dot{\gamma} > 0$, where u_x is the velocity in x direction. Note, that Roscoe calls the shear-rate κ. The choice of the sign of the shear-rate is without loss of generality. This is done in this thesis, to avoid needing to handle the sign in the associated algorithms. The shear modulus of the cell is G, the viscosity of the fluid surrounding the cell is η_0 and the internal fluid has a viscosity of η_1. An illustration of the typical deformation for this situation can be seen in figure 24.

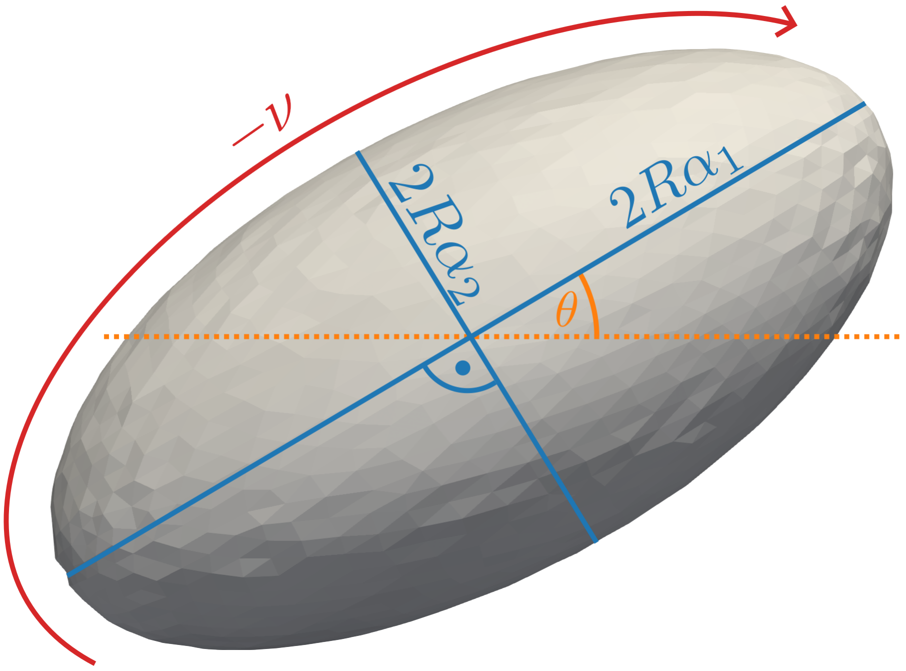


Figure 24: Illustration of a cell with the deformation typical for shear-flow. Many relevant parameters are introduced. The start of the deformation and its steady-state behavior can best be understood looking at the time evolution of the cell mesh.

The cell is scaled with α_i along direction i. The largest of which is α_1, the smallest is α_2 and the third one, which cannot be seen as it is normal to the image plane is typically $\alpha_3 \approx 1$. The angle between the largest axis and the flow direction of the fluid is θ. The cell rotates with the tank-treading frequency $\nu < 0$. The sign of ν is opposite of the one for $\dot{\gamma}$. This means, that for the choice here, the cell rotates clockwise, meaning against the mathematical positive direction. The rotation can also be seen in figure 25, where it takes the form of an oscillation.

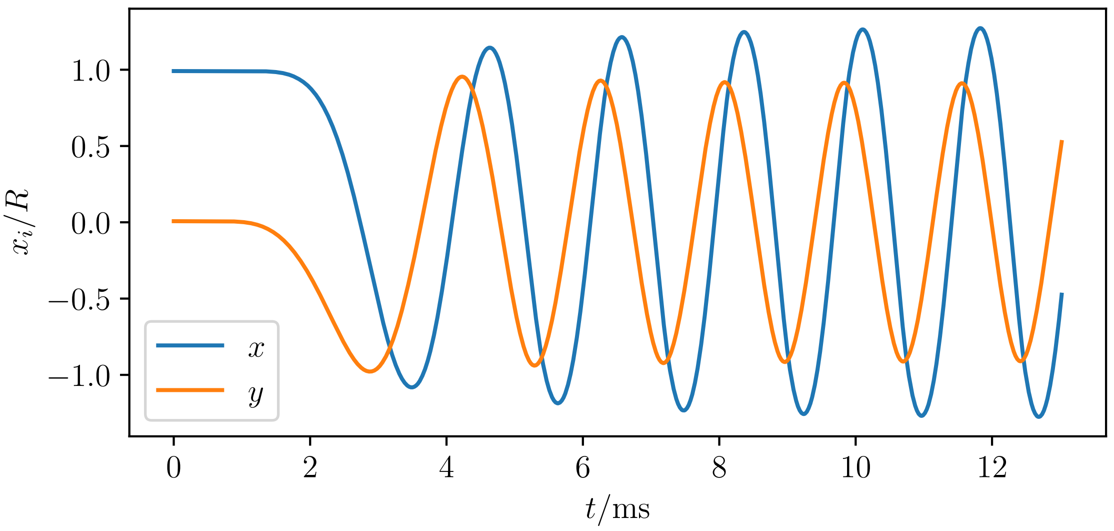


Figure 25: Plot of the x and y coordinates as a fraction of the cell's radius of a point passing through the cell's apex. The coordinates are given in relation to the center of the cell. After the initial startup is completed, the cell slowly continues stretching until its final shape is archived.

This type of plot is what will be used for evaluations. These examples are results of simulation 2.

2 Roscoe example

Box:	$100 \times 500 \times 100$
	$L_0 \approx 625\,\mathrm{nm}$
Fluid:	Newtonian::water (fluid 1)
BC:	velocity (BC 2)
	$\dot{\gamma} = 1 \times 10^4\,\frac{1}{\mathrm{s}}$
Cell:	Young's modulus $E = 1 \times 10^2\,\mathrm{Pa}$
	Radius, Resolution $R = 7.5\,\mathrm{\mu m}, 12$
	Capillary number $Ca_\mathrm{K} \approx 1.3$

Next, the relevant equations are covered.

9.2. Relevant equations

Not all equations in the publications are necessary. Here, only the ones required for determining the observable parameters are listed. The notation is altered, where it was found to be convenient.
Roscoe (28):

$$\prod_i \alpha_i = 1 \tag{232}$$

Roscoe (18):

$$g_i'' = \int_0^\infty \lambda \frac{\alpha_i^2 + \lambda}{\prod_j \left(\alpha_j^2 + \lambda\right)^{\frac{3}{2}}} \mathrm{d}\lambda \tag{233}$$

Roscoe (21):

$$g_i' = \int_0^\infty \frac{\alpha_i^2 + \lambda}{\prod_j \left(\alpha_j^2 + \lambda\right)^{\frac{3}{2}}} \mathrm{d}\lambda \tag{234}$$

The following three equations have been altered by moving a few factors around to make the equations shorter. This is indicated by primes.
Roscoe (39) and Roscoe (40):

$$\left.\begin{matrix} I' \\ J' \end{matrix}\right\} = 4 \frac{g_1'' \pm g_2''}{\sum_{i \neq j} g_i'' g_j''} \tag{235}$$

Roscoe (43):

$$K' = \frac{\alpha_1^2 + \alpha_2^2}{g_3'} \tag{236}$$

The following equation has been reordered for the numerics done later. This is done to avoid accidentally dividing by 0.
Roscoe (78):

$$\frac{J'}{I'}\left(\alpha_1^2 - \alpha_2^2\right) - \left(\alpha_1^2 + \alpha_2^2 - \frac{2}{\alpha_1^2 \alpha_2^2}\right) = 0 \tag{237}$$

Roscoe (62):

$$\sigma = \frac{5\eta_0}{2G} \tag{238}$$

$$\tau = \frac{3\eta_0 + 2\eta_1}{2G} \tag{239}$$

These definitions clash a bit with the other nomenclature in this thesis. Furthermore, mostly only the fraction of them is required. This fraction, defined as follows only depends on the viscosity contrast c.

$$\frac{\tau}{\sigma} = \frac{3\eta_0 + 2\eta_1}{5\eta_0} = \frac{3 + 2c}{5} \tag{240}$$

With c defined as follows.

$$c = \frac{\eta_1}{\eta_0} \tag{241}$$

Some equations do require σ, for these it will be inserted directly to avoid the use of the variable.
Roscoe (80):

$$2\theta = \arccos\left(\frac{\alpha_1^2 - \alpha_2^2}{\alpha_1^2 + \alpha_2^2} \frac{2K' + (c-1)(\alpha_1^2 + \alpha_2^2)^2}{2K' + (c-1)(\alpha_1^2 - \alpha_2^2)^2}\right) \tag{242}$$

The following equation has typo in the original (the sign is wrong). It has been corrected here.
Roscoe (79):

$$\dot{\gamma} = \frac{G}{\eta_0} \frac{\alpha_1^2 - \alpha_2^2}{I' \sin 2\theta} \tag{243}$$

Roscoe (41):

$$\nu = -\frac{\alpha_1^2 + \alpha_2^2}{2\alpha_1\alpha_2} \left(1 - \frac{\alpha_1^2 - \alpha_2^2}{\alpha_1^2 + \alpha_2^2} \cos 2\theta\right) \frac{\dot{\gamma}}{2} \tag{244}$$

With this, all relevant equations are listed. The following discussion covers how to use them to retrieve the expected deformation from a set of parameters.

9.3. Usage

The equation in Roscoe theory are not invertible. They can still be used to find the deformation for a given set of parameters. This is a little more involved. First, it should be noted, that equations (232) – (237) purely deal with geometry and can theoretically be solved universally, without any regard for the parameters. For any pair of α_1 and α_2, α_3 can be calculated using equation (232). With this, the aforementioned equations can be viewed as depending only on α_1 and α_2. In this view, equation (237) can be solved numerically using a root-finding algorithm, to give an α_2 for any given α_1. These can be used together with the concrete parameters in equations (242) and (243) in order to generate the shear-rate corresponding to the given α_1. If the shear-rate matches the desired one, the α_i and the angle θ, together defining the geometry have already been computed. The remaining quantity, namely the tank-treading frequency ν is straight forward to calculate using equation (244). From here, it is a matter of generating an α_1, that is consistent with the intended shear-rate. This is done in this thesis with an efficient parallel algorithm, with an error smaller than 0.1 ‰. For a contrast $c > 2$, there exists a maximum α_1, which can be used to generate reasonable guesses. A decent heuristic approximation can be found in the following.

$$\alpha_{1,\max} \approx \sqrt{\frac{8}{3}\frac{1}{c-1} + 1} \tag{245}$$

A more in depth discussion can be found in appendix M. If there is no maximum deformation, this thesis assumes, that $\alpha_1 < 10$ is a reasonable first guess. Technically, the algorithm is stable until $\alpha_1 \approx 50$, however such large deformations do not happen in the experiments discussed in this thesis and are not relevant. For the example simulation above (simulation 2), the parameters predicted from theory can be seen in table 1.

α_1	α_2	α_3	θ	ν_{tt}
1.37	0.722	1.01	0.485	-4.12 $\times 10^3 \frac{1}{\text{s}}$

Table 1: Roscoe parameters as predicted from theory. Errors suppressed for brevity.

Due to their numeric origin, these values carry errors. These are nine orders of magnitude below the values. The values have been rounded to a sensible accuracy. With this, the Roscoe model is covered. In the following, extraction of the relevant data from simulations is explained.

10. Evaluating Roscoe simulations

There is no additional simulation implementation required to handle Roscoe simulations. However, evaluating them, especially in an automatic way is difficult. Therefore, this section elaborates on the algorithm enabling this. Consider the deformed shape seen in figure 24 from the previous section. There are different approaches to evaluate the deformation. The general approach using the inertia ellipsoid is cumbersome, therefore this thesis opts for a simple approach using only select mesh points. The following algorithm assumes, the cell to be centered on the origin of the coordinate system. Furthermore, in the implementation the coordinates are scaled with the relaxed radius R of the cell, to create a cell with a relaxed radius of 1. One point is needed in the rotational plane (xy-plane in this thesis). It does not have to pass through the apex of the cell, but this is preferable. The meshes in the present work contain points at the surface of the cell in each cardinal direction. This thesis picks the point initially on the x-axis. This is called $\vec{x}$ in the following. α_3 can trivially be determined from the size of the cell in z-direction in relation to R. The point on the z-axis is used for this. This algorithm is a two-step process. First $\vec{x}$ is used to estimate most parameters. Afterwards, the full model for the movement of $\vec{x}$ is fitted to its trajectory (compare figure 25 from the previous section), with the estimates as start values. Said full model of the trajectory is given in the following.

$$\vec{x} = R \begin{pmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \alpha_1 & 0 & 0 \\ 0 & \alpha_2 & 0 \\ 0 & 0 & \alpha_3 \end{pmatrix} \begin{pmatrix} \cos(\nu t + o) \\ \sin(\nu t + o) \\ 0 \end{pmatrix} \tag{246}$$

Here $2\pi > o > 0$ is some phase offset. This equation can be seen as an ordinary rotation model with the tank-treading frequency ν, that is then scaled by the α_i and finally rotated around the z-axis by θ. Executing the matrix multiplications and grouping the trigonometric functions yields the following.

$$\begin{aligned} x_x &= R\alpha_1 \cos(\nu t + o)\cos(\theta) - R\alpha_2 \sin(\nu t + o)\sin(\theta) && (247) \\ &= \frac{R}{2}\alpha_1[\cos(\nu t + o - \theta) + \cos(\nu t + o + \theta)] \\ &\quad - \frac{R}{2}\alpha_2[\cos(\nu t + o - \theta) - \cos(\nu t + o + \theta)] && (248) \\ &= \frac{R}{2}(\alpha_1 - \alpha_2)\cos(\nu t + o - \theta) \\ &\quad + \frac{R}{2}(\alpha_1 + \alpha_2)\cos(\nu t + o + \theta) && (249) \end{aligned}$$

$$x_y = R\alpha_1 \cos(\nu t + o)\sin(\theta) + R\alpha_2 \sin(\nu t + o)\cos(\theta) \tag{250}$$

$$\begin{aligned} &= \frac{R}{2}\alpha_1[-\sin(\nu t + o - \theta) + \sin(\nu t + o + \theta)] \\ &\quad + \frac{R}{2}\alpha_2[\sin(\nu t + o - \theta) + \sin(\nu t + o + \theta)] \end{aligned} \tag{251}$$

$$\begin{aligned} &= -\frac{R}{2}(\alpha_1 - \alpha_2)\sin(\nu t + o - \theta) \\ &\quad + \frac{R}{2}(\alpha_1 + \alpha_2)\sin(\nu t + o + \theta) \end{aligned} \tag{252}$$

$x_z = 0$ is ignored in this section due to its irrelevance. Now, the magnitude of this position is considered as follows.

$$|\vec{x}|^2 = x_x^2 + x_y^2 \tag{253}$$

$$\begin{aligned} &= \frac{R^2}{4}\left\{(\alpha_1 - \alpha_2)^2 + (\alpha_1 + \alpha_2)^2 \right. \\ &\quad \left. + (\alpha_1 - \alpha_2)^2[\cos(2\nu t + 2o) + \cos(2\theta) + \cos(2\nu t + 2o) - \cos(2\theta)]\right\} \end{aligned} \tag{254}$$

$$= \frac{R^2}{4}\left[2\alpha_1^2 + 2\alpha_2^2 + 2\left(\alpha_1^2 - \alpha_2^2\right)\cos(2\nu t + 2o)\right] \tag{255}$$

$$= \frac{R^2}{2}\left(\alpha_1^2 - \alpha_2^2\right)\left[1 + \frac{2\alpha_2^2}{\alpha_1^2 - \alpha_2^2} + \cos(2\nu t + 2o)\right] \tag{256}$$

$$= \frac{R^2}{2}\left(\alpha_1^2 - \alpha_2^2\right)\left[\frac{2\alpha_2^2}{\alpha_1^2 - \alpha_2^2} + 2\cos^2(\nu t + o)\right] \tag{257}$$

$$= R^2\left[\alpha_2^2 + \left(\alpha_1^2 - \alpha_2^2\right)\cos^2(\nu t + o)\right] \tag{258}$$

This can be seen in figure 26 for the example from the previous section (simulation 2).

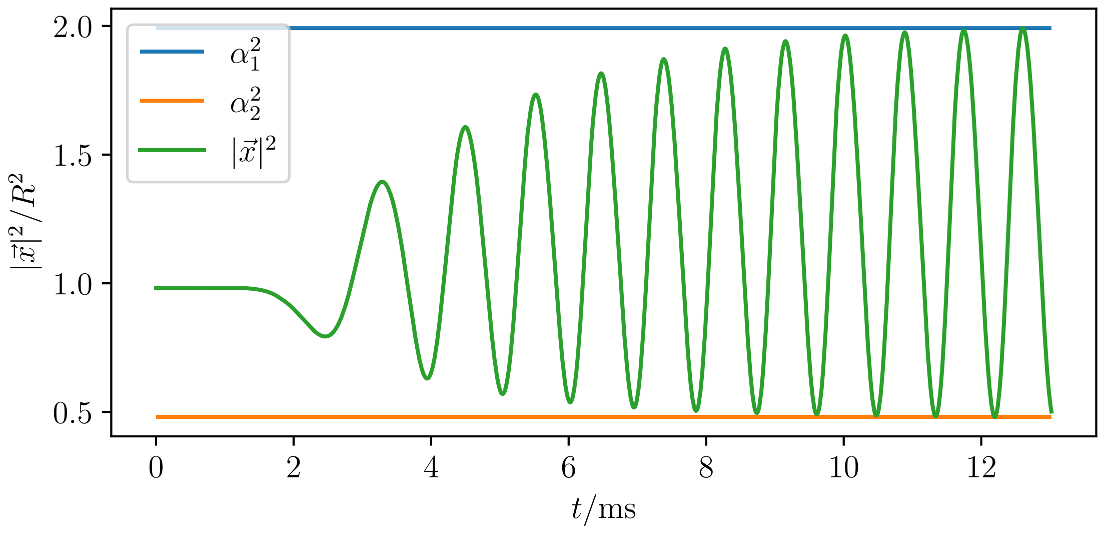


Figure 26: Plot of $|\vec{x}|^2$ as a fraction of the squared radius of the cell for the point originally on the x-axis. The coordinates are given in relation to the center of the cell. After the initial startup is completed, the cell slowly continues stretching until its final shape is archived. The limits give α_1 and α_2.

As can be seen in the equation as well, the oscillation of the squared magnitude is bounded by $R^2\alpha_2^2$ and $R^2\alpha_1^2$ as its maximum and minimum respectively. The global maximum and minimum are easy to compute and are used for this. Next, the magnitude derived above is reordered to find an equation for the cosine as follows.

$$\cos^2(\nu t + o) = \frac{|\vec{x}|^2 - R^2\alpha_2^2}{R^2(\alpha_1^2 - \alpha_2^2)} \tag{259}$$

A fast Fourier transform is performed on the result. Ignoring the peak at the origin, the global maximum and minimum reveal an estimate for ν. Next, the phase offset is estimated. For this, the global maximum of $|\vec{x}|^2$ is used. It is a maximum of a squared cosine, so the following must hold for the time t, the maximum occurs at.

$$-\nu t - o = k\pi \tag{260}$$

With k being an unknown integer. Note, that the negation of the argument of the cosine has been done, because $\nu < 0$ in this thesis. This sign can be absorbed into the definition of k. The offset is then first estimated as follows.

$$o_1 = -\nu t \,\%\, \pi \tag{261}$$

Note, that $\cdot \,\%\, \cdot$ denotes the modulo operation. Finally, a conditional offset is added as follows.

$$o = o_1 + \begin{cases} \pi & \text{, for } o_1 > \dfrac{\pi}{2} \\ 0 & \text{, else} \end{cases} \tag{262}$$

As the cosine is squared, an offset of π in the model equations (eq. (246)) would disappear. This offset is added back here in order to put o as close to a multiple of 2π as possible. Experience shows, that o is typically small. With these estimates known, first a fit is performed with equation (258), to refine them. Afterwards, equation (246) is fitted to retrieve θ. Before this second fitting process starts, the model is evaluated using the start parameters of the fit. If the data is negative at the location of the maximum of the model, another π offset is introduced. With this all parameters are extracted. This algorithm is robust enough to handle the Roscoe simulations in this thesis without requiring manual intervention. For the example simulation above (simulation 2), the values seen in table 2 are retrieved.

α_1	α_2	α_3	θ	$\nu_{\mathrm{tt}}/\frac{1}{\mathrm{s}}$
1.413 ± 0.001	0.702 ± 0.002	1.0109 ± 0.0007	0.508 ± 0.001	-3660 ± 3

Table 2: Roscoe parameters extracted from the simulation.

Note, that only the last 15 % of the simulation output has been used, as the simulation converges slowly. Compared to the expected values from section 9.3, these are off by about 10 %. This is caused by inaccuracies in the simulations, which are discussed in appendix X. The extraction algorithm is contained within the evaluation scripts (see appendix A.3). Clearly, for Roscoe simulations, the deformation is quantified differently, than for the AFM experiments discussed before. An overview about the different deformation measures used in this thesis is given in the following section.

11. Quantifying deformation

The concrete shape of a cell is difficult to convey. Working with it often necessitates expressing this shape through a scalar. This is conversion looses a lot of information, and consequently many different options exist, depending on what the other believes best retains the relevant information. Many of them are rather similar. In the following, only the ones used in this thesis are listed. This large number of deformation measures is not by choice, but because working with publications often necessitates using their deformation measure. The deformation measures, listed in the following, are ordered from simple to complex. First, the very simple indentation is discussed.

11.1. Indentation

The deformation in an AFM experiment is typically quantified through the indentation depth δ. Often, this is expressed as fraction of the cell's diameter. It is a one dimensional measurement, where only the vertical movement of the vertex centered below the indenter tip is considered. This neglects the complicated shapes at the cell-indenter contact. However, this is easily available from the experimental setup from the movement of the stage. Consequently, it is widely used. Indentations from different indenter shapes should not be used together, as they are not comparable. Otherwise, this is fine for the use in AFM. The Hertz model used for modeling AFM experiments accounts for the neglected shape. The equilibrium has a deformation of 0. Typical deformations produce positive values up to unity (relative to the cell's diameter). Deformation against the typical direction can be expressed with a negative sign. This is used in sections 7 and 20 as well as appendices L and W. A slightly more complicated model, considers the extent of the cell along multiple axes. This is covered next.

11.2. Dimensions

Deformation data is typically initially gathered as an image. For simple flows, one can align one axis of the image with the flow direction, with the other one being given, by the physical limitations of the experiment. How to best translate the distortion along the different axes into a single scalar is difficult. Also, whatever method is chosen, one needs to think about error propagation and how the errors might compound or compensate each other. A simpler approach is to just avoid it and list the extent of the cell along each axis separately. This is also normally expressed as a fraction of the cell's diameter. These values change depending on the axes chosen. Therefore, this method is only sensible for cases were the definition of axes is clear by the experimental geometry. Also handling two values is not as easy as dealing with a single one. The equilibrium has a deformation of 1 relative to the diameter. Typical deformations produce positive values. Deformation against the typical direction do as well. This is way of quantifying deformation is used in section 24. In theory, this can be extended to all three axes. This is discussed next.

11.3. Roscoe deformation parameters

One can also specify the deformation as the extent relative to the original size along all three axis. This is what Roscoe theory is based on. The α_i give the ratio with which each axis is stretched. The axes are defined as the axes of the rotational ellipsoid, that is formed by a cell in a shear-flow. Aside from the things listed for measuring along two axes, this way of quantifying the deformation has the issue of observing the third axis. In typical experimental setups, the third axis cannot be seen. If the initial shape of the cell is not known, guesswork is required. The equilibrium has a deformation of 1. Typical deformations produce positive values. Deformation against the typical direction are excluded by definition. This quantification is used in sections 9, 10 and 21 as well as appendix M. If the relaxed size is not known, it can be removed from the deformation by way of relating the dimensions. This is discussed next.

11.4. Aspect ratio

One can determine the Dimensions as listed above and then set them in relation to each other. It could be defined as follows.

$$D_\mathrm{R} = \frac{l}{h} \tag{263}$$

With l being the length along a Poiseuille(-ish) flow and h being the extent normal to it. This removes the need to know the relaxed size of the cell. It reduces the two numbers to a single one. The error of D_R is derived from the errors of l and h as follows.

$$u_D = \sqrt{\left(\frac{\partial D_\mathrm{R}}{\partial l}\right)^2 u_l^2 + \left(\frac{\partial D_\mathrm{R}}{\partial h}\right)^2 u_h^2} \tag{264}$$

$$\frac{u_D}{D_\mathrm{R}} = \sqrt{\left(\frac{u_l}{l}\right)^2 + \left(\frac{u_h}{h}\right)^2} \tag{265}$$

The u quantities denote the uncertainty or error. This uses standard error propagation for independent variables. The relative error of D_R is given by Pythagorean addition of the relative errors of l and h. Meaning, the error propagation is fine and does not introduce issues. The equilibrium has a deformation of 1. Typical deformations produce values above unity. Deformation against the typical direction can be expressed with positive values below unity. This is used in section 23. There exists a more refined approach of the same concept, which is listed next.

11.5. Taylor deformation

The Taylor deformation parameter is defined as follows [97].

$$D = \frac{\alpha_1 - \alpha_2}{\alpha_1 + \alpha_2} \tag{266}$$

Here, α_1 and α_2 refer to the major and minor semi axis of the cell respectively. The values from Roscoe can be used here as well. This deformation parameter removes the need to know the initial radius. The error propagation returns the following.

$$\frac{u_D}{D} = \frac{2\alpha_1\alpha_2}{\alpha_1^2 - \alpha_2^2}\sqrt{\left(\frac{u_{\alpha_1}}{\alpha_1}\right)^2 + \left(\frac{u_{\alpha_2}}{\alpha_2}\right)^2} \tag{267}$$

$$= \frac{2\alpha_1\alpha_2}{(\alpha_1 + \alpha_2)^2 D}\sqrt{\left(\frac{u_{\alpha_1}}{\alpha_1}\right)^2 + \left(\frac{u_{\alpha_2}}{\alpha_2}\right)^2} \tag{268}$$

The equilibrium has a deformation of 0. This causes the error to diverge for small deformations. This means, that this deformation metric is not usable for small deformations. Typical deformations produce positive values below unity. Deformation against the typical direction are excluded by definition. This is used in appendix G. Next, the most complex and error-prone deformation metric used in this thesis is discussed.

11.6. Circularity

The deformation from circularity is defined as follows [98].

$$D_\circ = 1 - \frac{2\sqrt{\pi A}}{L} \tag{269}$$

Here, A is the area of the visible cross-section of the cell and L is its circumference. This deformation parameter removes the need to know the initial radius. The error propagation returns the following.

$$\frac{u_D}{D_\circ} = \frac{1 - D_\circ}{D_\circ}\sqrt{\frac{1}{4}\left(\frac{u_A}{A}\right)^2 + \left(\frac{u_L}{L}\right)^2} \tag{270}$$

The equilibrium has a deformation of 0. This causes the error to diverge for small deformations. This means, that this deformation metric is not usable for small deformations. It must be stressed, that even rather significant deformations are small in this context. Consider an ellipse, that is twice as long as it is wide. This is approximately the deformation seen in the Roscoe example (simulation 2), with $Ca_\mathrm{K} \approx 1.3$. This is a significant deformation. It is not quite “large”, but in the upper range of what one would call “medium”. For the Taylor deformation, $D = \frac{1}{3}$ in this case. This means, that the error gets scaled with a $\frac{4}{3}$ in total. This is not an issue. In contrast, for the definition from circularity $D_\circ \approx 0.018$. This cannot be put as a straight scaling, because the error contribution of A is halved compared to the contribution of L. However, the term containing the deformation, which causes the error scaling is approximately 54 here. This means, that the relative error is increased by nearly two orders of magnitude. Even if the circumference does only carry a 2 % error, this deformation metric is completely unusable. For smaller, and especially truly small deformations, this factor can easily reach 100 or even 1000. Small deformations are important in Poiseuille(-ish) flows, where deformations can be quite subtle. Measuring a circumference from pixelated data is a lot more difficult, than the straight lines required for the other models. Consequently, errors in the one- to two-digit percent range are to be expected. This renders this deformation metric useless for everything but the largest deformations. Typical deformations produce positive values below unity. Deformation do not have a typical direction. This is used in sections 23 and 24. This concludes the discussion on cells for the time being. Next, the ways normal forces from viscoelastic fluids may influence cell deformation, are discussed.

12. Influence of normal forces on cell deformation

There is plenty of literature reporting cell behavior, that seems to slightly conflict with the prediction for Newtonian fluids [98–100]. With the present thesis, it is actually possible to simulate viscoelastic fluids. Before the concrete experiments are covered, it is however worthwhile, to answer the question, what is actually to be expected. The experiments presented in this thesis cannot be analytically solved. What is possible, is to take two standard scenarios, Poiseuille flow (see appendix P.2) and shear-flow and examine what forces would act on an undeformed cell. The Poiseuille flow will be covered first as it is more important.

12.1. Poiseuille flow

For simplicity, a round channel is considered. However, as the flow profile is quite similar towards the center in rectangular channels, this is not much of a restriction. First the stresses are considered in cylindrical coordinates (see appendix N.2). From appendix F, the viscous stress contributions are knows as follows (eq. (467) and eq. (492)).

$$\sigma_{xr} = -G\frac{r}{2^j} = \eta(\dot{\gamma})\dot{\gamma} \tag{271}$$

With the pressure gradient $G = -\frac{\partial p}{\partial x}$, the radius r, and the variable j for switching between 2D ($j = 0$) and 3D ($j = 1$). This is universal. Furthermore, $\eta(\dot{\gamma})$ is the model-dependent viscosity and $\dot{\gamma} < 0$ is the shear-rate. For PTT, which is used in this thesis to describe viscoelastic behavior, the following is known (eq. (629)).

$$c_{xx} = 2c_{xr}^2 \tag{272}$$

Note, that this holds regardless if a solvent stress is included, as it is in this thesis, or not, like it can be found in literature [63]. Converted to the stress, it reads as follows.

$$\tau_{xx} = 2\frac{\lambda}{\eta_\mathrm{p}}\tau_{xr}^2 \tag{273}$$

$$\sigma_{xx} = 2\frac{\lambda}{\eta_\mathrm{p}}(\sigma_{xr} - \eta_\mathrm{s}\dot{\gamma})^2 \tag{274}$$

With this, the only non-zero components of $\underline{\sigma}$ are given. In total, it reads as follows (see appendix N for coordinate transformation).

$$\underline{\sigma} = \begin{pmatrix} \sigma_{xx} & \sigma_{xr}\cos(\varphi) & \sigma_{xr}\sin(\varphi) \\ \sigma_{xr}\cos(\varphi) & 0 & 0 \\ \sigma_{xr}\sin(\varphi) & 0 & 0 \end{pmatrix} \tag{275}$$

The surface of sphere is parameterized as follows (see appendix N.3).

$$\vec{x} = \begin{pmatrix} r_\circ\cos(\theta) \\ r_\circ\sin(\theta)\cos(\varphi) \\ r_\circ\sin(\theta)\sin(\varphi) \end{pmatrix} \tag{276}$$

The force $\vec{F}$ caused by this stress can be expressed as follows.

$$\vec{F} = \int \sigma \hat{n} \mathrm{d}A \approx \sigma \hat{n} \mathrm{d}A \tag{277}$$

$$\vec{f} \approx \sigma \hat{n} \tag{278}$$

Here, $\hat{n}$ is the normal vector corresponding to $\vec{x}$. The integration area is assumed small, this allows the integrand to be approximated as constant. This is divided out to get a force area-density $\vec{f}$. In total this reads as follows.

$$\vec{f} = \begin{pmatrix} \sigma_{xr} \sin(\theta) + \sigma_{xx} \cos(\theta) \\ \sigma_{xr} \cos(\theta) \cos(\varphi) \\ \sigma_{xr} \cos(\theta) \sin(\varphi) \end{pmatrix} \tag{279}$$

This can be split into elastic and viscous contributions along the coordinate axis as follows.

$$\vec{f}_{\text{elastic}} = \begin{pmatrix} \sigma_{xx} \cos(\theta) \\ 0 \\ 0 \end{pmatrix} = \sigma_{xx} \cos(\theta) \hat{e}_x \tag{280}$$

$$\vec{f}_{\text{viscous}} = \begin{pmatrix} \sigma_{xr} \sin(\theta) \\ \sigma_{xr} \cos(\theta) \cos(\varphi) \\ \sigma_{xr} \cos(\theta) \sin(\varphi) \end{pmatrix} = \sigma_{xr} \sin(\theta) \hat{e}_x + \sigma_{xr} \cos(\theta) \hat{e}_r \tag{281}$$

For the interpretation of these equations it is necessary, to realize that the stress components differ by sign as follows.

$$\sigma_{xx} \geq 0 \tag{282}$$

$$\sigma_{xr} \leq 0 \tag{283}$$

Furthermore, the following proportionalities can be assumed.

$$\sigma_{xr} \propto r \tag{284}$$

$$\sigma_{xx} \propto r^2 \tag{285}$$

Note, that the second one is an approximation for neglected η_{s}. It always grows with r^2 as its leading term. With the solvent viscosity included it also has a linear reducing term, that grows in importance towards the outside of the channel. Meaning, in the center, where the cell is, neglecting the solvent viscosity is always acceptable. With this, the forces can be written as follows.

$$\vec{f}_{\text{elastic}} \propto r^2 \cos(\theta) \hat{e}_x \tag{286}$$

$$\vec{f}_{\text{viscous}} \propto -r \sin(\theta) \hat{e}_x - r \cos(\theta) \hat{e}_r \tag{287}$$

The cylindrical radius r can be computed from the radius of the cell R and θ as follows.

$$r = R \sin(\theta) \tag{288}$$

With this, the forces can be written as follows.

$$\vec{f}_{\text{elastic}} \propto \sin^2(\theta)\cos(\theta)\hat{e}_x \tag{289}$$

$$\vec{f}_{\text{viscous}} \propto -\sin^2(\theta)\hat{e}_x - \sin(\theta)\cos(\theta)\hat{e}_r \tag{290}$$

It is important to note, that the viscous contributions as listed here are independent of the model. Meaning, any viscous contribution, even a Newtonian fluid results in a force of this shape. An illustration can be seen in figure 27.

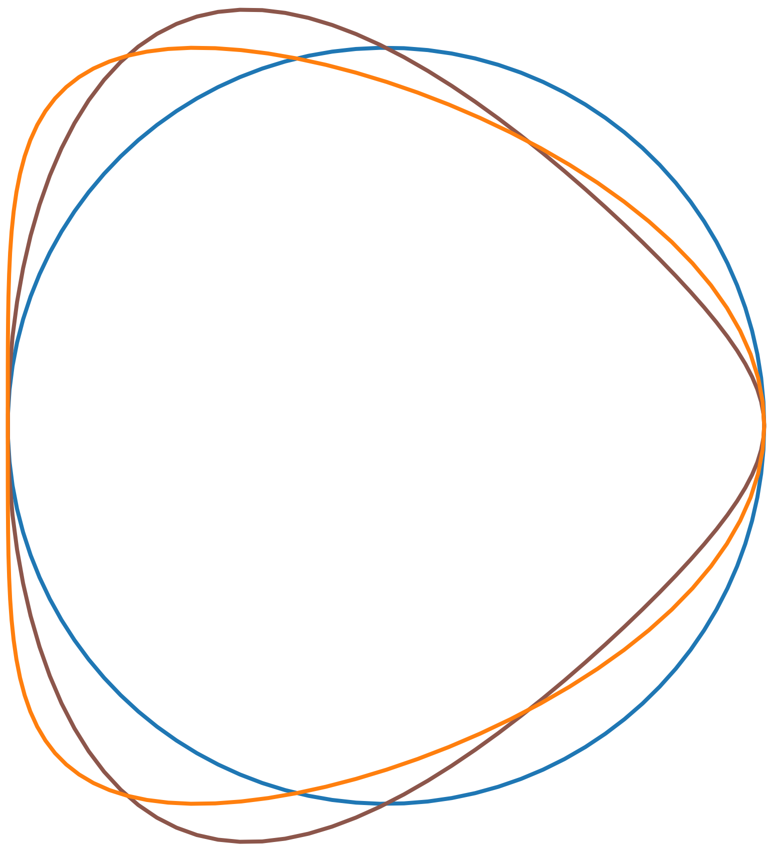


Figure 27: Illustration of how the viscous force in radial direction and the viscous force in x direction influence the shape of a cell in a Poiseuille flow.

This is a crude approximation, generated by adding the force to the undeformed shape. As deformation happens, the force is altered and returning forces are generated. This has been ignored here. Still one can quite clearly see, how the typical bullet shape is formed. It is also apparently possible for the cell to reduce its aspect ratio as defined in section 11 below unity. This means, that the cell extends further into the radial direction, than into x direction. There is only a small window of deformations enabling this though. If viscous force in x direction gets any stronger than displayed here, the cell is elongated in x direction increasing the aspect ratio. An illustration of the effect of the elastic force can be seen in figure 28.

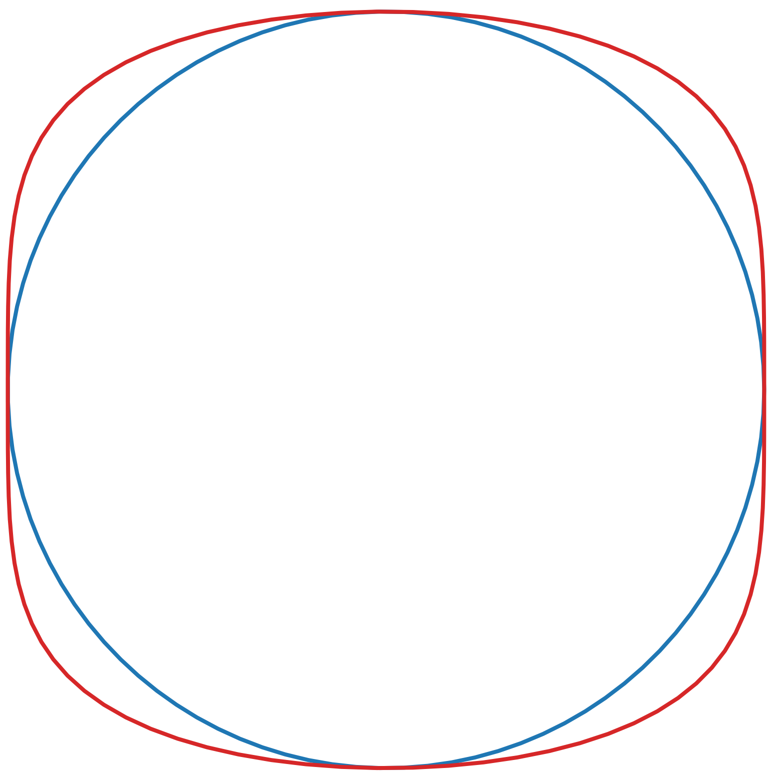


Figure 28: Illustration of how the elastic force influences the shape of a cell in a Poiseuille flow.

It should be noted, that in reality, the cell does of course not become square. The reduction of the stress to 0 at the x-axis does not happen fully and the elastic forces from the cell work towards a round(-ish) shape. What can be seen in simulations and experiment alike is a slight elongation [101]. Conceptually, it is easier, if one goes back an abstraction layer. Ignoring the complex behavior of the stress components and only accounting for its sign, one can easily draw general trends. This can be seen in figure 29.

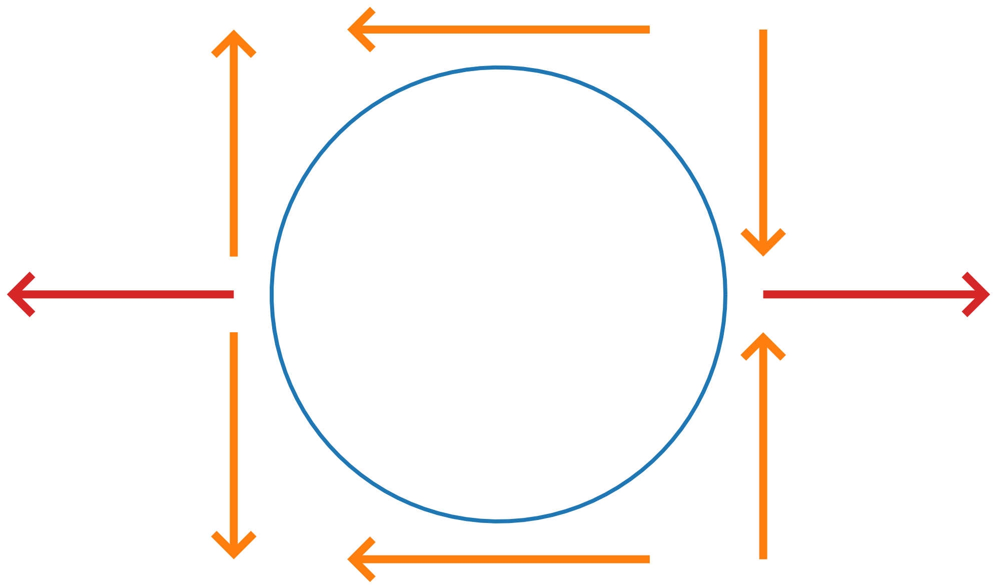


Figure 29: Illustration of how the viscous forces and elastic force influence the shape of a cell in a Poiseuille flow.

This illustration is good enough to predict the viscoelastic effects most easily observed. As mentioned above, the viscous forces do not influence the aspect ratio for small forces and may decrease it for intermediate ones. The elastic forces always increase the aspect ratio. This means, that cells when suspended in a viscoelastic medium should be longer, than in an elastic medium. For large deformations, the difference will not be as noticeable, because the viscous forces also increase the aspect ratio on such cases. So far, a discussion on the relative magnitude of viscous and elastic stress has been omitted. This is because this is dependent on material parameters. In general, for very small shear-rates, the viscous stress dominates. For very large shear-rates, the elastic stress dominates. In the range explored in this thesis, they are of a similar order of magnitude. This concludes the discussion of the Poiseuille flow scenario. The shear-flow will be discussed next.

12.2. Shear-flow

For shear-flow, the assumption of location independent stress is actually accurate. This is because $\dot{\gamma} > 0$ is a constant here. The stress components, listed in the following, can be given in Cartesian coordinates.

$$\sigma_{xy} = \eta(\dot{\gamma})\dot{\gamma} \tag{291}$$

$$\sigma_{xx} = 2\frac{\lambda}{\eta_\mathrm{p}}(\sigma_{xy} - \eta_\mathrm{s}\dot{\gamma})^2 \tag{292}$$

Consequently, the stress tensor can also be written using the Cartesian components. This simplifies it, as can be seen in the following.

$$\underline{\sigma} = \begin{pmatrix} \sigma_{xx} & \sigma_{xy} & 0 \\ \sigma_{xy} & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \tag{293}$$

With the same parameterization for the sphere as before, the following force density can be calculated.

$$\vec{f} = \begin{pmatrix} \sigma_{xy}\sin(\theta)\cos(\varphi) + \sigma_{xx}\cos(\theta) \\ \sigma_{xy}\cos(\theta) \\ 0 \end{pmatrix} \tag{294}$$

In the following discussing, only the $\varphi = 0$ and $\varphi = \pi$ cases will be covered. This means, the cell's intersection with the xy-plane is of interest. No insight is lost through this. The total force can be exasperated into its viscous and elastic components as follows.

$$\vec{f}_{\text{elastic}} = \begin{pmatrix} \sigma_{xx}\cos(\theta) \\ 0 \\ 0 \end{pmatrix} = \sigma_{xx}\cos(\theta)\hat{e}_x \tag{295}$$

$$\vec{f}_{\text{viscous}} = \begin{pmatrix} \sigma_{xy}\sin(\theta)\cos(\varphi) \\ \sigma_{xy}\cos(\theta) \\ 0 \end{pmatrix} = \sigma_{xy}\sin(\theta)\cos(\varphi)\hat{e}_x + \sigma_{xy}\cos(\theta)\hat{e}_y \tag{296}$$

The viscous forces explain the deformation described by Roscoe (see section 9) well. This can be seen in figure 30.

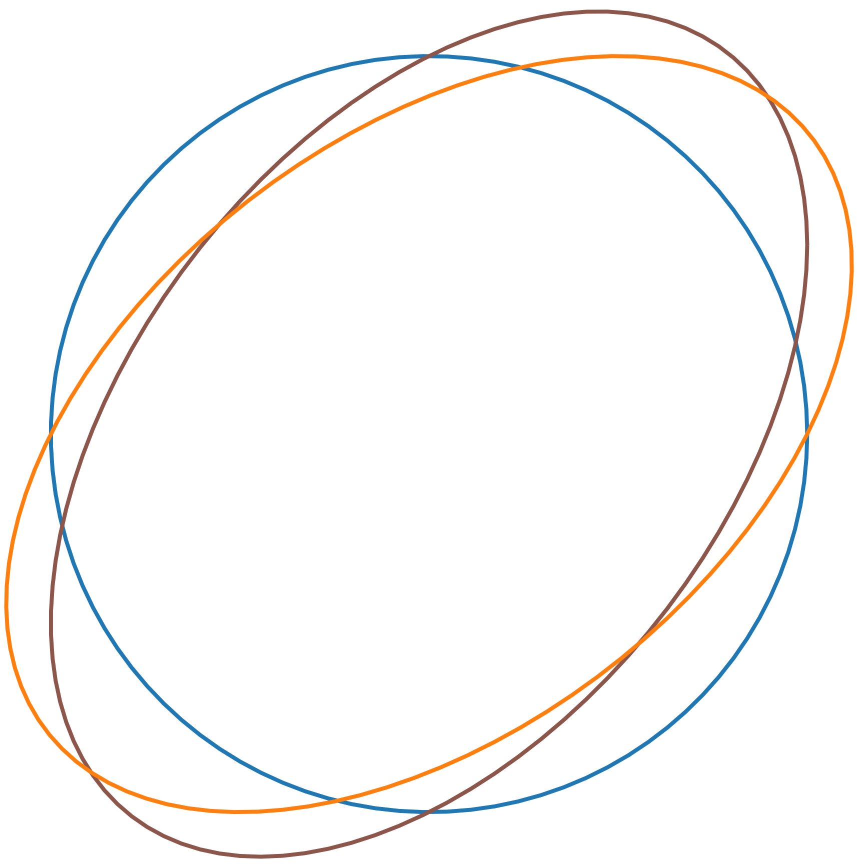


Figure 30: Illustration of how the viscous force in y direction and the viscous force in x direction influence the shape of a cell in a shear-flow.

The same simplified diagram as for the Poiseuille case can be seen in figure 31

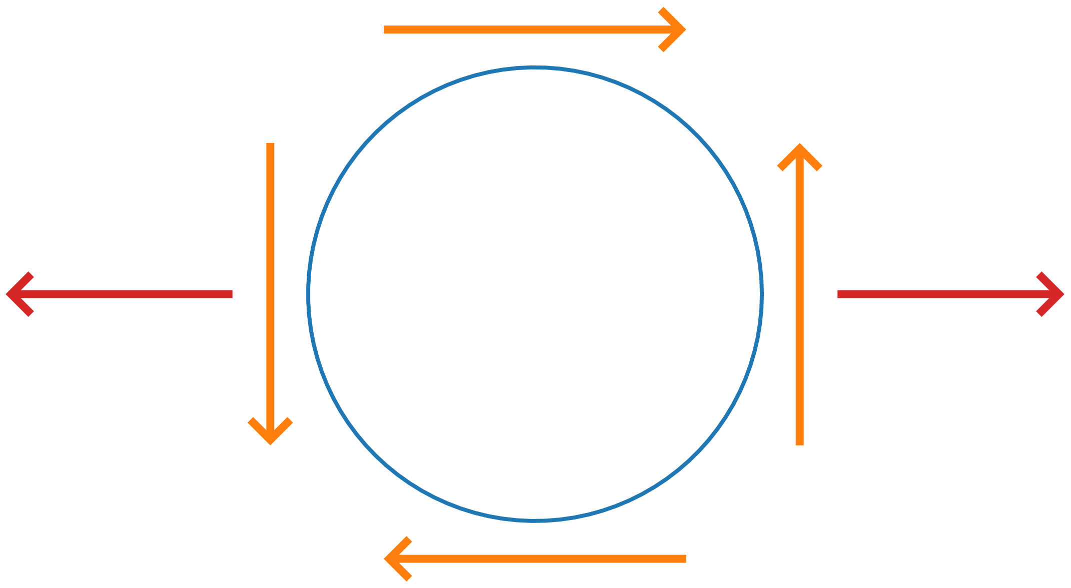


Figure 31: Illustration of how the viscous forces and elastic force influence the shape of a cell in a shear-flow.

The consequences of the normal stress elongating the cell are not straight forward to understand. The influence on the α_i, likely depends on the alignment angle. This angle will be reduced by the normal stress. Given, that there is a component of the force caused by the normal stress, that is pointed against the tank-treading, one would expect its frequency to reduce. For stronger deformations, this effect is reduced. Also, this poses the question of reduced compared to what. The shallower alignment angle by itself leads to slower tank-treading according to Roscoe. In a way this is a matter of opinion and will be explored in the corresponding simulations (see section 21). This concludes the discussion on the expected effect of the inclusion of normal stresses in regard to cell deformation. Next, the last piece of the simulation algorithm, the boundaries, are discussed.

13. Boundaries

The simulation domain is limited in space and time. Therefore, one has to specify how these boundaries should be handled. In the following all boundary conditions implemented in *FluidX3D* are listed. These refer to the handling of the boundaries in space. Afterwards, the relevant boundaries in time (initial conditions) are explored. This is not trivial for the extensions and a discussion whether it is possible to shorten the startup is also warranted. Finally, the different options to drive the flow in the simulation volume using the available boundary conditions are explored. This includes their relevant advantages and drawbacks and recommendations depending on the issue at hand.

13.1. Boundary conditions

Boundary conditions (BCs) handle how the edges of the simulation volume are handled. Given *FluidX3D* combines an LBM solver with a finite volume scheme for the polymer stress, the listed boundary conditions need to be specified for both. Particularly for LBM, much has been written about the vast amount of available boundary conditions already (see for example Krüger *et al.* [9]). *FluidX3D* was originally based on the implementation suggestion from Krüger *et al.* [9]. Due to implementation details most of the boundary conditions where implemented slightly differently. Typically, *FluidX3D* is most of the time more rigorous in regard to handling the density. This is advantageous especially when using Reynolds-Scaling (see section 16). The outflow boundaries presented below were developed from scratch. Due to minute differences having a profound impact at times, all other boundary conditions implemented in *FluidX3D* are also listed in the following and described in detail. In the following sections, each boundary condition is given as an equation, describing how the post-collision populations f_i^* are handled if a boundary is present. In these equation $-i$ refers to the direction opposite to i, $\vec{x}_n$ refers to the current position and $\vec{x}_{j(i)}$ refers to the neighboring node in direction i. This explains how the boundaries behave for LBM. The same boundary conditions also apply to the finite volume algorithm. Their handling however is often up to the interpretation of the user on a simulation to simulation basis. The respective sections contain suggestions. Each boundary condition is accompanied by a small red box listing the most important information to use it. This includes the flag to enable them (set in the `defines.hpp`), the flag to indicate such a boundary (set in the `setup.cpp`) and which values need to be specified on the boundary.

13.1.1. Periodic boundaries

When a problem with translational symmetry (for example a pipe of infinite length), or a problem with a matching inlet and outlet is considered, the boundaries can be set as periodic. This is done by treating the $x = 0$ and the $x = L_x - 1$ planes as adjacent (L_x is the number of LBM nodes along the x-axis, see section 2.4). The same applies to the other axis as well. This is done for any computation involving neighboring nodes (for example streaming or gradient calculations). In terms of streaming, the equation looks like perfectly normal streaming as can be seen below.

$$f_i\left(\vec{x}_{j(i)}\right) = f_i^*(\vec{x}_n) \tag{297}$$

The trick is simply, that the neighborhood function $j(i)$ performs a modulo operation on every index. The finite volume advection scheme uses the same neighborhood function and therefore requires no special treatment at this boundary. For the example above, this means, that the outlet flows directly back into the inlet. This is often rather convenient and allows the simulation domain to be set to $L = 2$ along the periodic axis for problems with translational symmetry. This speeds up computation considerably. Therefore, this is the default boundary condition for *FluidX3D*. With no other boundary condition set, periodic boundaries are always implied. These boundaries carry the caveat, that the pressure drop typically expected along a pipe cannot be reproduced. This means, that they actually do worse for bigger simulation volumes. This is relevant if the pipe shall for example be large enough to contain an IBM cell (see section 6). Also driving the flow (see section 13.3) becomes non-straightforward, if the channel does not have translation symmetry. Periodic boundaries are the only ones, that require special discussion of the IBM algorithm. All other boundaries are handled through the resulting flow field. To handle periodic boundaries, the IBM algorithm performs modulo operations whenever accessing the flow field. This means, that technically the cell leaves the simulation volume, but always accesses information at the corresponding location within the volume. This is done to avoid the difficulties associated with wrapping the cell back into the volume. Particularly, when the cell has partially left the volume a conflict arises between the cell being in the volume and the mesh points' relative positions to each other being maintained correctly. This choice of incrementing the cell position consistently can lead to the advection being of the order of the rounding error for lower precision simulations (see appendix O).

13.1.2. Halfway bounce-back

These boundaries represent solid walls and are consequently referred to as walls in the remainder of the thesis. They are the boundaries most often used. Walls are implemented using the halfway bounce-back algorithm as follows.

$$f_{-i}(\vec{x}_n) = f_i^*(\vec{x}_n), \quad \text{if } \vec{x}_{j(i)} \text{ is a wall} \tag{298}$$

This bounces back the populations within a single time-step, meaning, they only traveled halfway. The result is, that the velocity becomes zero halfway between a fluid node and a neighboring node flagged as a wall. This means, that the actual no-slip boundary condition represented by this scheme is fulfilled halfway between the nodes. With this, the walls line up with the edges of the finite volume domains. This means, that their boundary condition reduces to zero-flux of the polymer-conformation tensor across this wall. This is done by just not executing the advection scheme for those neighbors. Walls are always enabled in *FluidX3D*.

1 Walls

Define to enable	always enabled
Flag for wall nodes	`TYPE_W`
Set on wall	nothing

Strictly speaking, the exact wall location does depend on the value of the LBM relaxation time τ_r. An alternative interpretation if this error could also be a slight slip at the no-slip boundary. This usually does not matter much and is therefore ignored throughout this thesis. Using these to approximate round structures leads to the staircase effect. This leads to the question of how to most accurately depict a circle (see appendix P.4). The artefacts created by this approximation usually are not too bad given the resolution is high enough, but can cause issues particularly in conjunction with viscoelastic fluids (see section 25).

13.1.3. Volume-flow/Velocity boundaries

These boundaries represent the typical Dirichlet boundaries. They enforce a velocity on a wall and are therefore called velocity walls in this thesis. Similar to the walls above, the wall is placed halfway between the fluid node and the node flagged as a wall. The algorithm works as follows.

$$f_{-i}(\vec{x}_n) = f_i^*(\vec{x}_n) - 2w_i\rho(\vec{x}_n)\frac{\vec{c}_i\cdot\vec{u}\big(\vec{x}_{j(i)}\big)}{c_\mathrm{s}^2}, \quad \text{if } \vec{x}_{j(i)} \text{ is a velocity wall} \tag{299}$$

The halfway bounce back condition is a simplification of velocity walls for $\vec{u}\big(\vec{x}_{j(i)}\big) = 0$. Consequently, the same caveats already mentioned there apply here. This boundary condition can be used in two ways. With the velocity parallel to the wall, they can be used for shear-flows. If the velocity is normal to the wall, these conditions can be used to model an inlet or outlet. In this case however, it shall be noted, that the mass flow through such walls is not defined and depends on the pressure in the system. Velocity walls are valid flux-neighbors for the fine-volume advection algorithm (see section 4). Therefore, the polymer-conformation tensor needs to be specified on the wall if these boundaries are used in conjunction with the fine-volume algorithm.

2 Velocity walls

Define to enable	`VELOCITY_WALLS`
Flag for wall nodes	`TYPE_VW`
Set on wall	velocity
	polymer-conformation tensor

13.1.4. Mass-flow

These boundaries are modified velocity walls. They were developed to use them as an inlet or outlet with a defined mass flow. This is done by using a fixed density set on the wall instead of the density on the boundary node (last fluid node before the wall). Therefore, its equation looks as follows.

$$f_{-i}(\vec{x}_n) = f_i^*(\vec{x}_n) - 2w_i\rho\big(\vec{x}_{j(i)}\big)\frac{\vec{c}_i \cdot \vec{u}\big(\vec{x}_{j(i)}\big)}{c_\mathrm{s}^2}, \quad \text{if } \vec{x}_{j(i)} \text{ is a mass-flow wall} \tag{300}$$

This partially solves the conservation of mass problem (see subsection 13.3.3). They are otherwise identical to the velocity walls.

3 Mass-flow walls

Define to enable	`MASSFLOW_WALLS`
Flag for wall nodes	`TYPE_MW`
Set on wall	velocity
	density
	polymer-conformation tensor

13.1.5. Pressure

These boundaries represent a pressure through setting the density as described in equation (26). They are termed pressure walls in this thesis. Similarly to the boundaries above, the wall is placed halfway between the fluid node and the node flagged as a wall. Mathematically, the equation, which can be seen below, should actually contain the velocity on the wall. As the interpolation suggested by Krüger *et al.* [9] is not possible in the algorithm used in this thesis, the local velocity is used. This carries a small error.

$$f_{-i}(\vec{x}_n) = f_i^*(\vec{x}_n) - 2w_i\rho\big(\vec{x}_{j(i)}\big)\left(1 + \frac{\left(\vec{c}_i \cdot \vec{u}(\vec{x}_n)\right)^2}{2c_\mathrm{s}^4} - \frac{\vec{u}(\vec{x}_n)^2}{2c_\mathrm{s}^2}\right), \quad \text{if } \vec{x}_{j(i)} \text{ is a pressure wall} \tag{301}$$

These boundaries are particularly powerful, as they are easy to specify correctly. They are also close to the parameters as they would be specified in the actual experiment. However, particularly when used as an inlet, they excite the line instability inherent to LBM (see section 15.1). This can lead to instability, particularly when using viscoelastic fluids. Pressure walls are valid flux-neighbors for the finite-volume advection algorithm (see section 4). Therefore, the polymer-conformation tensor needs to be specified on the wall if these boundaries are used in conjunction with the fine-volume algorithm.

4 Pressure walls

Define to enable	`PRESSURE_WALLS`
Flag for wall nodes	`TYPE_PW`
Set on wall	density
	polymer-conformation tensor

13.1.6. Outflow

Due to the mass conservation issue (see subsection 13.3.3), it is desirable to have a boundary condition, that is capable of letting the flow leave uninterrupted. To do this, the boundary node should behave as if the flow would extend further. First consider a laminar flow in a channel with translational symmetry. The velocity is constant along the streamlines, while the density drops with a constant slope (due to the pressure drop, see equation 26)). This means, that the populations of a neighboring node can be estimated using the local populations, scaled according to the drop in density ρ (conceptionally equation (20)), which can be found using the opposing neighbor. These estimates are used to replace the populations leaving the node without replacement due to the boundary. Formally streaming is modified as follows.

$$f_{-i}(\vec{x}_n) = \left[2\rho(\vec{x}_n) - \rho\left(\vec{x}_{j(-i)}\right)\right] f^*_{-i}(\vec{x}_n), \quad \text{if } \vec{x}_{j(i)} \text{ is an outflow boundary} \tag{302}$$

The validity of which depends on the following assumption, which notably is stricter than the velocity $\vec{u}$ being constant along streamlines.

$$\vec{u}\left(\vec{x}_{j(i)}\right) = \vec{u}(\vec{x}_n), \quad \text{if } \vec{x}_{j(i)} \text{ is an outflow boundary} \tag{303}$$

For this assumption to hold, the channel has to have constant diameter and shape towards the boundary for long enough so that the flow can equilibrate. This can be done by just artificially adding a straight piece of channel onto the actual simulation volume of interest, should it not be shaped in this way. This assumption also does not hold for the populations not normal to the boundary. However, given the resolution is high enough to make the velocity profile along the boundary linear on the scale of Δx, the density error introduced by this inaccuracy gets compensated by the other populations incurring errors of same magnitude but different sign. Deviations from these assumptions manifest themselves in the form of a slow decrease or increase of the total density in the system. This velocity assumption is also used to calculate velocity gradients needed for the constitutive equation of the polymer stress near the boundary. For the finite volume scheme, advection is allowed into these boundaries, where the flux is discarded, while advection from them is prevented. Given care is taken to include a straight piece of channel with decent resolution before the outlet, these boundaries work well for Newtonian flows. If the flow directly before the boundary reverses to flow away from it, like can happen during the startup of viscoelastic flows, these boundaries can become unstable and are therefore not suited for all viscoelastic flows. Strong forces exciting the line instability (see section 15.1) can also lead to negative velocities.

5 Outflow walls

Define to enable	`OUTFLOW_WALLS`
Flag for wall nodes	`TYPE_OW`
Set on wall	nothing

13.1.7. Free-slip

These boundaries represent a solid wall but allow for tangential velocity components. They are referred to as slip walls in the remainder of the thesis. This is accomplished by half-way bounce-back with retained tangential velocity components. Slip walls are implemented as follows.

$$f_{d_i(i,n)}\left(\vec{x}_{d_n(i,n)}\right) = f_i^*(\vec{x}_n), \quad \text{if } \vec{x}_{j(i)} \text{ is a slip wall} \tag{304}$$

With the non-trivial functions for the destination population d_i and destination node d_n best described by the algorithm used. This algorithm is displayed in the following using pythonic pseudocode.

```
# returns $d_n(i,n)$ and $d_i(i,n)$
def getNI(n, i):
    # Number of non-zero components in $\vec{c}_i$
    type = getVelocityType(i)
    if type == 1:
        # simple bounce-back
        return n, -i
    elif type == 2:
        # Get the cardinal directions composing $\vec{c}_i$
        baseVectors = decompose(i)
        flipped = i
        for v in baseVectors:
            if neighbor(i) is SLIP_WALL:
                # Mirror $i$ on the plane with the normal $v$
                flipped = flip(i, v)
            elif neighbor(i) is WALL:
                # simple bounce-back
                return n, -i
        if flipped == i:
            # simple bounce-back
            return n, -i
        # vector to node where the populations ends up being
        # $(\vec{c}_i + \vec{c}_{\mathit{flipped}})/2$ rounded down
        total = (c(i) + c(flipped))//2
        # Convert the vector to an index
        return getCorrespondingIndex(total), flipped
    elif type == 3:
        pass # Currently not implemented
```

Figure 32: Pythonic pseudocode of the slip wall algorithm.

As can quite easily be seen in figure 32, this scheme is currently not implemented in *FluidX3D* for velocity-sets containing space diagonals. This excludes D3Q15 and D3Q27. Given that this thesis focuses on D3Q19 due to its price to performance ratio, this is fine. The algorithm could also be implemented for space diagonals with similar scheme based on decomposition and flipping but a lot more edge cases would need to be considered. This algorithm as been developed in this thesis in order to replace an older, more complicated and more limited algorithm used in *ESPResSo* [49, 50], which has been used by the group so far (see reference [67]). In regard to the finite volume algorithm, as the no-slip walls this boundary also has zero flux. A short validation can be found in appendix Q. It shows, that these boundaries work very well, as long as the walls are straight. On diagonal walls approximated with steps, they act like no-slip walls.

6 Slip walls

Define to enable	`SLIP_WALLS`
Flag for wall nodes	`TYPE_SW`
Set on wall	nothing

13.2. Initial conditions

The LBM algorithm requires initial values for the populations. These are set as the equilibrium populations associated with the initial density and velocity field (see equation (20)). A discussion is warranted, whether the initialization should be the resting fluid or the fluid near equilibrium. For (Generalized) Newtonian fluids this generally does not matter much as they reach equilibrium quickly. Therefore, these are initialized at rest in this thesis. This means, that the velocity field is initialized to zero and the density to unity (in LU; see section 2.3). Setting the density to unity, means, that the fluid initially is not subject to any pressure gradient, that might be applied. However, any such gradient establishes itself quickly and meanwhile any error incurred due to that is small. This is due to typically deviations from unity due to pressure differences being close to negligible (see equation (26)). For this reason, the density is initialized at unity for all cases, regardless of the flow field. In the case of viscoelastic fluids reaching equilibrium can take a significant amount of time. Furthermore, the viscoelastic algorithm is susceptible to large gradients. Therefore, it would be beneficial to initialize the flow field close to equilibrium. As the startup is not reproduced accurately when using Reynolds-Scaling anyway (see section 16), nothing is lost. Despite the lack of analytical solution existing the flow field can be somewhat closely estimated using the flow of a Newtonian fluid with the viscosity present at average shear expected in the present geometry. The viscoelastic finite-volume algorithm does however also require the polymer conformation tensor to be set. For a fluid at rest all of its components are zero. Its value close to equilibrium cannot easily be estimated. Picking it wrong can lead to instabilities. Initializing a viscoelastic fluid close to equilibrium can lead to difficulty in telling whether it indeed has reached equilibrium as any remaining changes would be small and slow. For this reason, viscoelastic fluids are initialized at rest for validation simulations. For any other simulation it is doable, but considered risky. Therefore, this is an option to speed up simulations, which in this thesis is generally reserved as a last resort. Close to all simulations are initialized at rest without further mention.

13.3. Driving the flow

The flow field observed within the simulation volume is generated by the boundaries. The simple walls are used to enclose the flow in solid walls. All the other boundaries can be used to drive the flow. Pure shear-flow is an important case, which will be discussed first. All other flow fields are some combination of inlets and outlets. The discussion is in regard to a simple circular Poieseuille-flow (pipe), but does also apply to general geometries. An overview of important geometries used in this thesis can be found in appendix P. There are multiple options of combining the boundaries mentioned above to generate similar flows. In the following a few combinations and their advantages and drawbacks are listed.

13.3.1. Shear-flow

Pure shear-flows are important for benchmarking and to isolate physical effects happening due to shear. The flow field of a Poiseuille-flow near its border can be approximated as pure shear. As pure shear-flow is a lot easier (and faster) to simulate this is routinely done. For this the simulation volume is enclosed by two velocity walls at $y = 0$ and $y = L_y - 1$. These are set to a velocity in negative and positive x-direction respectively. The resulting flow field is as follows.

$$\vec{u} = \dot{\gamma} y \hat{e}_x \tag{305}$$

Here y is in IBM coordinates (see section 6.2) and $\dot{\gamma}$ is the shear-rate. This flow is reproduced very accurately and converges quickly with minimal risk of instabilities.

13.3.2. Periodic boundaries and body-force

In systems with translational symmetry, there is a constant pressure gradient along the flow. This can be interpreted as a constant force acting on each lattice node. Therefore, this case can be implemented by picking periodic boundaries and adding a constant force to each lattice node. This is particularly easy and stable. However, most interesting systems do not exhibit translational symmetry. In such cases, this way of driving the flow can still be done. Strictly speaking, it is not valid anymore. However, the resulting flow field is still accurate. In such cases it is no longer possible to relate the flow-rate to the pressure gradient. As experiments typically use the flow-rate as their control parameter, this means, that such simulations need some amount of trial and error to reproduce the experimental conditions. If systems can be made periodic, this is the preferred way to drive the flow.

13.3.3. Mass-flow boundaries

When mass-flow walls are positioned in a way, that the velocity is normal to the wall, they can act as an inlet or outlet. With this, the flow-rate can trivially be reproduced. However, to be fully accurate, the boundary would already have to exhibit the correct velocity profile. This is not possible in general. It is usually good enough, to approximate the profile and add some length of channel after the inlet to allow the proper profile to establish itself. The required distance is dependent on the fluid and has to be determined using trial and error. The most significant issue with this way of driving the flow is however the mass conservation issue. These boundaries fix the flow through the inlets and outlets. If these do not match exactly, the mass in the simulation volume is not conserved. Even errors of the size of the rounding error of the used float (see appendix O) can amount to significant issues due to the large amount of time-steps the typical simulation contains ($\approx 1 \times 10^5$ to 1×10^6). Anything but bit-perfect agreement is unusable for medium to long simulations. Aside from identical inlets and outlets this is practically impossible. Inlets and outlets can often be forced to be identical. For other use-cases other options are explored in the following.

13.3.4. Mass-flow and outflow boundaries

As described above, the outflow boundaries have specifically been developed to solve the mass conservation issue. Using mass-flow walls for the inlets and outflow walls for the outlets, the advantages of the mass-flow walls can be retained. The outflow walls compensate for differences in the shape of inlet and outlet. For Newtonian fluids this is a good and convenient option. The flow near the outlet is reproduced accurately. No extra length of channel needs to be added here. For viscoelastic fluids (the bulk of this thesis), the outflow walls can have bad interactions during startup and therefore this scheme is purely suited for that use-case.

13.3.5. Pressure boundaries

As the velocity does not need to be specified, these boundaries, do not carry the caveat of determining the flow profile. Therefore, they can be used without any extra channel for Newtonian fluids. For viscoelastic fluids, they carry the same issue with respect to the polymer-conformation tensor. They also require the nontrivial translation of flow-rate to pressure difference, for comparisons to experiments. The mass conservation issue is not present and the outlet is more stable than outflow boundaries for viscoelastic fluids. In the case of viscoelastic fluids, pressure wall inlets can however excite the line instability (see section 15.1).

13.3.6. Mass-flow and pressure boundaries

Using mass-flow walls as inlets gives all their advantages, at the trade-off of requiring an additional piece of channel as described above. A pressure wall outlet (with a pressure corresponding to $\rho_{\mathrm{LU}} = 1$) is a stable way to let the flow pass mostly uninterrupted. The average density in the flow field will increase very slightly, but does not present an issue. The resulting flow is accurate near the outlet. For complex viscoelastic flows, this is the way to go.

13.3.7. Iterative approach

Of the options presented above, some technically require information not known at the time of simulation design. This is compensated by allowing some stretch of channel in which the flow field is known to be wrong. When using IBM it is often required for the cell to pass the channel multiple times to reach equilibrium. In such cases any region of incorrect flow field is not acceptable. For these cases the body-force scheme is preferable. However, if this is not possible, the flow is first simulated without the cell and with the region of incorrect velocity. From that, accurate boundary conditions are extracted to be used with the same scheme but without the region of incorrect velocity (or polymer-conformation tensor). This is twice the work and complicates the setup significantly. Therefore, this is only done if it is unavoidable. With this all components of the algorithm are covered. On overview will be provided next.

14. Overview of complete simulation algorithm

In the previous sections, the core algorithms of *FluidX3D* have been covered piece by piece. In this section an overview over the complete algorithm is provided to show how the pieces fit together. The complete algorithm is illustrated in figure 33.

```python
def simulation():
    # Preparation of the geometry and boundaries
    init()

    for i in range(numberSteps//vtkInterval):
        for j in range(vtkInterval):
            # Actual simulation algorithm.
            simulationStep()

        # Check if the fields contain NaN values (error values)
        # or if the cell is stretched too much.
        checkForErrors()
        # Generates fp32 vtk files with all disired fields.
        # This is done in fp32 by default to save disk space
        saveOutput()
```

Figure 33: Pythonic pseudocode of the command order in a complete simulation.

The function `init()` contains all the setup code. This is what is provided in the simulation boxes in this thesis. The checks for errors a writing of output is only done every few thousand LBM steps. This is done for one to limit the data transfer over PCIe, because this is slow and to limit the amount of output file. Files containing information on the whole simulation volume are rather large, even when saved with limited precision. To keep the storage requirements reasonable, saving of output is done only about a hundred to a thousand files per simulation. The commands making up a simulation step can be seen in figure 34.

```
def simulationStep():
    # Sets the force for each lattice point back to $0$.
    # $\vec{F}_{LBM} = \vec{0}$
    resetLatticeForce()
    # Update the IBM poisitions (see section 6.3.4)
    # using the interpolated velocities (see section 6.3.3).
    # Can be argued to belong to the end of the time-step.
    advectIBMPoints()
    # Updates the flags to indicate,
    # which LBM nodes are withing the cell (see section 6.6).
    handleINOUT()
    # Calculate all the forces $\vec{F}_{IBM}$ acting on the IBM mesh
    # (see sections 6.3.1 and 6.4).
    calcIBMForces()
    # Distribute $\vec{F}_{IBM}$ to $\vec{F}_{LBM}$ (see section 6.3.2).
    forceSpreading()
    # Stress shoveling part 1 (see section 4.3.1).
    conserveStressA()
    # Core LBM algorithm.
    # Gather stress contributions (see sections 4 and 18).
    # Perform collsion and streaming (eq. (22)).
    # Streaming adheres to boundaries (see section 13).
    # More details can be found in appendix~C.
    lbmStep()
    # Calculate new velocity and density (eq.(24) and eq. (23)).
    velocityAndDensityUpdate()
    # Additional output if desired (see section 2.8).
    strainRateAndViscousStressTensors()
    # Update the polymer-conformation tensor $\underline{C}$, (see section 4).
    polymerConformationTensorStep()
    # Stress shoveling part 2 (see section 4.3.1).
    conserveStressB()
```

Figure 34: Pythonic pseudocode of the command order in a simulation step.

With this, the core algorithm is covered. Later some advanced concepts will be discussed, which are advantageous for speed or required for stability. These necessitate small modifications to the algorithms presented so far. Next, important instabilities inherent to these algorithms are discussed.

15. Instabilities

There are multiple instabilities, which can be found described in literature [9]. Some instabilities have already been mentioned in passing. For example, LBM in general becomes unstable if the Mach number is too high (see section 2.7). The viscoelastic algorithm uses CTU to avoid checkerboard instabilities (see section 4.1.2). However, it should be mentioned, that this rarely still appears for large stresses around corners. Also, instability happens for large IBM forces (see section 6.7). All of these are well known, and are standard problems handled with standard methods. This section intends to describe instabilities, that were concrete problems to be solved in this thesis. These are the line instability and the jagged cell instability. First, the line instability is covered.

15.1. Line instability

The LBM algorithm allows for the propagation of waves-like disturbances with a wavelength of $2\Delta x$ that expand with the lattice speed of sound. These are not proper physical sound waves and rather a grid artefact. They can pass each other without interacting and whether they get dampened properly depends on implementation details of the LBM. This mode naturally arises from every disturbance of the flow field. As an example, simulation 3 is performed.

3 Line instability example

Box:	$51 \times 51 \times 11$
	$L_0 = 1.25\,\mu\text{m}$
Fluid:	Newtonian::water (fluid 1)
BC:	periodic

This simulation has been initialized with a disturbance in the velocity in x-direction of $u_x = 1\,\frac{\text{m}}{\text{s}}$ at the center of the simulation volume. This is an example for this instability mode naturally arising. After 0.26 ms, the simulation volume exhibits distinct lines as can be seen in figure 35.

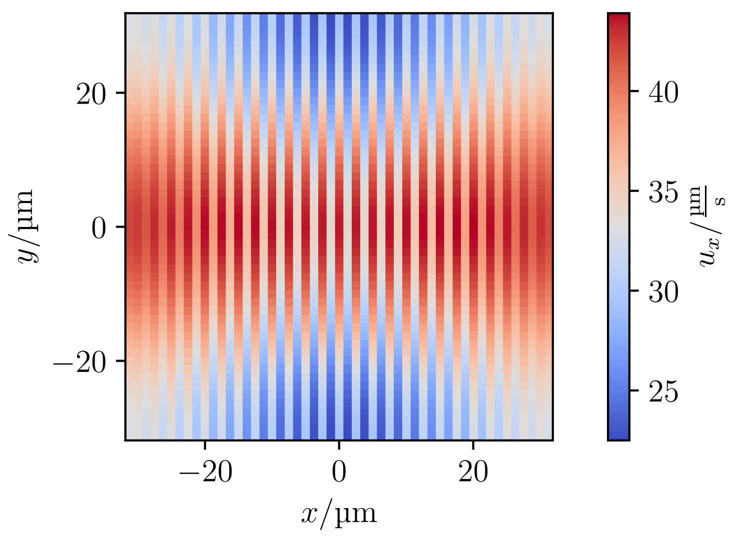


Figure 35: Plot of the velocity in the xy-plane 0.26 ms after a perturbation.

The distinctive line shape is very obvious in the two-dimensional plots typically used to examine flow fields. Examining the flow field along the propagation axis at the same time yields the plot seen in figure 36.

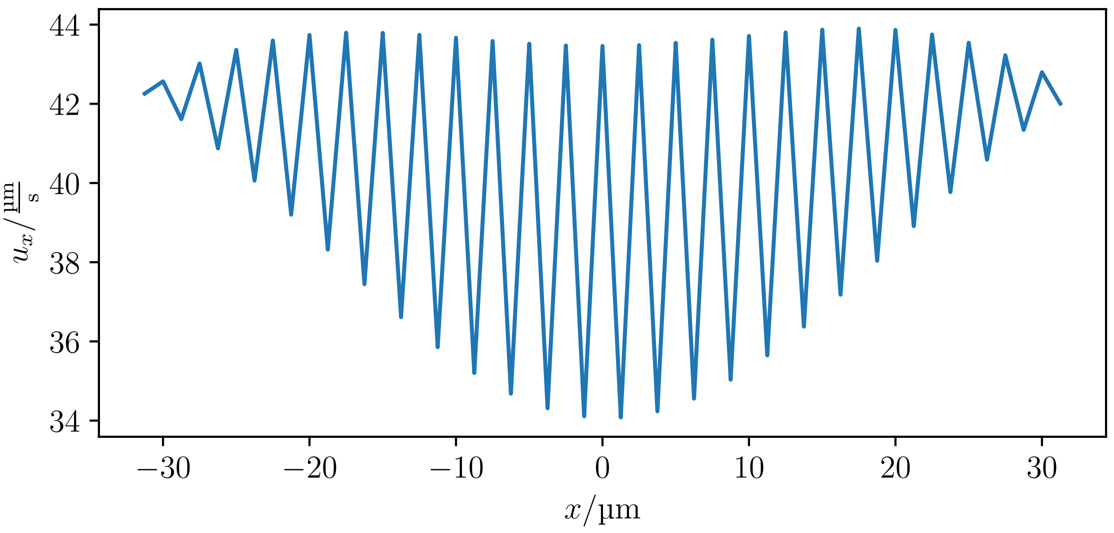


Figure 36: Plot of the velocity along the x-axis 0.26 ms after a perturbation.

This can be interpreted as a wave with a wavelength of $2\Delta x$, where Δx is the lattice constant. The amplitude of this wave gets dampened with time. Furthermore, the velocity actually is negative at the minima early on. The instability behaves similar to a sound-wave traveling with the lattice speed of sound. Notable these wave like structures are not circles but propagate in a line shape. Each frame of the video is a single LBM step. It can be seen clearly, that every affected node switches between high and low velocity with every time-step. This leads to the origin of this instability, which will be discussed next.

15.1.1. Origin

As already stated, the line instability is inherent to LBM. If neighboring lattice nodes have velocities pointing towards each other, these will just pass each other instead of interacting. The fluid viscosity will slowly dampen this instability. Any perturbation will cause this behavior. The explanation is a bit difficult in three dimensions. To avoid this, a D1Q3 lattice is considered instead. An illustration can be seen in figure 37.

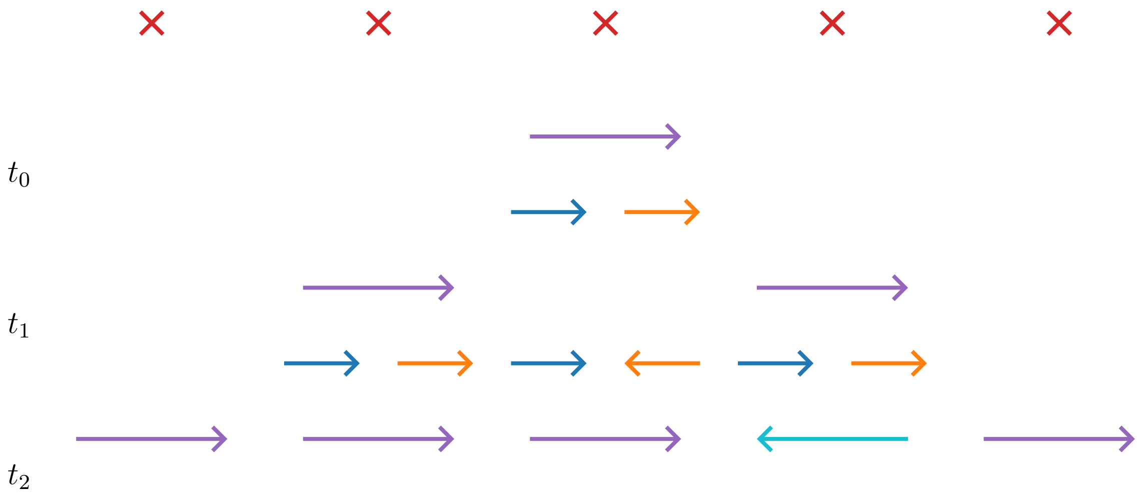


Figure 37: Illustration of the genesis of the line instability from a perturbation. For each lattice node, the direction of the velocity, leftward offset population and rightward offset population is displayed for tree time-steps. Notably for $t = t_2$ at the node neighboring the perturbed node in the direction of the perturbation, the sign of the velocity changes. The instability begins.

The arrows representing the populations should be seen as the direction of the $f_i' \vec{c}_i$ vector. Here $f_i\prime$ are the offset populations in accordance to section 2.2 and $\vec{c}_i$ are the lattice velocities. The movement of the initial perturbation creates a low pressure zone on this lattice node at $t = t_1$. In the population frame, this corresponds to all offset equilibrium populations being negative because $\rho < \rho_0$. In the next time-step, the streaming of the inverted rightward offset population causes the velocity to flip. Note, that the magnitude of the arrows has been omitted deliberately, to reduce complexity. However, the velocity to the left of the node with the initial perturbation does have a velocity lower than its neighboring node at $t = t_2$. So, the instability progresses in both directions. By itself, this is not too bad, as this wave pattern gets dampened by the viscosity rather quickly. However, during the development of *FluidX3D*, this instability was often observed. A closer look revealed, that depending on implementation details, the IBM algorithm can couple to this instability and cause something akin to constructive interference leading to broken simulations. For the discussion of this, consider the oscillatory background to be present. It is always present in IBM and its amplitude is dependent on the magnitude of the forces present in the simulation. Instead of the cell mesh two points connected by a spring are discussed. One needs to separate the discussion far and close to the equilibrium of the spring.

15.1.1.1. Far from equilibrium

The forces generated by the spring dominate and overwrite the oscillatory background. The oscillation should relax. Still, this instability can happen sometimes depending on a crucial implementation detail. Section 2.5.1 already mentioned, that the velocity and the force need to be separated by a time-step. The reason as, that for stronger forces, the IBM points oscillate with an oscillation period of two lattice steps by themselves. The IBM algorithm causes a small time lag between force, velocity and position an any given IBM vertex. This means, that for this oscillation, a stretching force can act, when the spring is stretched or when the spring is compressed. This causes either constructive interference or relaxation. If constructive interference happens, the oscillation of the IBM points promotes the line formation on the velocity field, as those oscillate with the same frequency.

15.1.1.2. Close to equilibrium

When the position approaches the equilibrium, the background oscillations begin to dominate the motion. In this case, it is relevant, whether the velocities and forces caused by the spring are in sync with the background oscillation. If they are, the instability gets amplified, causing the simulation to break. Otherwise, the oscillations get dampened. Whether the movement is in sync is initially random and therefore, in a large mesh, there are always bonds amplifying the instability. What is critical here, is whether the movement stays in sync. This is dependent on an implementation detail. This detail is the distinction of `PULL` vs `PUSH` as discussed in appendix C. The small difference in the stored populations leads to slight differences in the velocities. The main difference being if the force contribution has already propagated at the end of the time-step (`PUSH`) or not (`PULL`). If the movement is out of sync with the background oscillation, it does not matter as both cases show relaxation. This is different for the in-sync case. For springs close to equilibrium, this contribution influences, whether the spring passes its equilibrium during a time-step. For `PULL`, the equilibrium is always passed, and the instability can grow. For `PUSH`, the movement actually shows increased growth for one time-steps, but then does not manage to pass the equilibrium. This means, that the in-sync behavior switches to out of sync behavior and the instability is dampened. With this, the mitigation steps are clear, but they are summarized again in the following.

15.1.2. Mitigation

As already mentioned, the time shift for the force in equation (42) is crucial (see section 2.5.1). Also, in the combination with IBM, it is necessary to use the `PUSH` formulation of LBM. It should be noted, that this actually seems to be a question of dimensionality. A one dimensional mesh fails on a one dimensional LBM grid when using `PULL`. The same holds, when using a 2D mesh on a 2D grid and a 3D mesh on a 3D grid. However, a 1D mesh on a 2D grid or a 2D mesh on a 3D grid works. This is likely because the vertices of the mesh can escape into the additional dimension and break the buildup of the instability this way. As already mentioned, the line instability is inherent to LBM and can never be fully avoided. Particularly for very large forces it can be briefly seen, before disappearing to dampening. Such large forces at the very edge of stability can and should be avoided. This is sometimes costly for viscoelastic simulations. With this, the line instability is covered. Next, an important instability concerning cells is covered.

15.2. Jagged cells

Simulations (for example simulations 4 and 5) with a meshed cell in a Poiseuille flow, develop a surface instability. This is particularly prevalent for high-shear situations. This instability can be seen in figure 38. It develops exponentially over time.

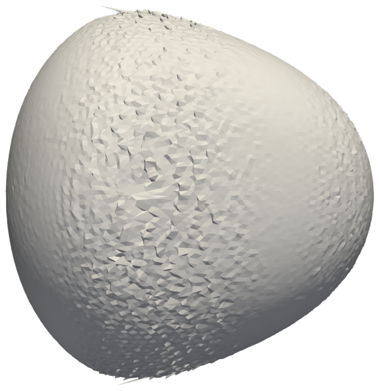


Figure 38: Render of the jagged cell instability.

This in itself causes issues with data evaluation and sparks questions about accuracy. However, these surface deformities can grow even larger (see the very top and bottom of figure 38). Over time, this can cause the simulation to crash. This subsection discusses the origin of this instability and presents a mitigation strategy.

15.2.1. Origin

A cursory glance at figure 38 reveals, that the instability is most prominent at the equator of the cell. Furthermore, it is worsened by increases shear-rate at the equator. The phenomenon can be explained by a numeric failure of the mesh to conform to the shape enforced by the flow. For this discussion, consider a cylindrical coordinate system. The cylinder shall be aligned with the flow direction. In these coordinates, the points on the equator are the ones with the largest radius. Due to the developed Poiseuille flow-profile, these points consequently experience a lesser velocity compared to the rest of the mesh. This means, that they move backwards in relation to the rest of the mesh. Due to the discretization, some points on the equator experience less velocity than the others. This is partially due to them having a slightly larger radial coordinate and partially due to their exact position relative to the LBM mesh. The LBM mesh is square and has to approximate a round(ish) flow profile for Poiseuille geometry. Even with interpolation, the velocity seen on a plane with theoretically equal velocity will not be constant (see appendix P.4). These points, subject to the lowest velocity in the mesh, are the first to experience relative motion in negative flow direction. Notably, this does not influence their radial coordinate, therefore this driving effect is retained. In the following, these points are called the tips of their associated tetrahedrons. Looking back at figure 38 and observing carefully, one can see, that these tips are, as is expected from their origin, all pointing towards negative flow direction. The deformation resulting from the relative motion causes a force pulling the tips back, as well as a force pulling the bases of the associated tetrahedrons towards the tip. The latter is, what is physically supposed to happen, to establish the shape the mesh should take in such a flow. However, if the size of a tetrahedron is similar to the distance between lattice nodes, these forces (partially) cancel out and can have no noticeable affect. In this case, the deformation increases due to the present flow field, and the instability is observed. From this three things follow:

1. This instability can never be fully removed as its root cause is discretization
2. Higher resolution reduces the instability
3. The size of the tetrahedrons relative to the lattice matters

With this mitigation strategies will be discussed in the next subsection.

15.2.2. Mitigation

As higher resolution is associated in an at least cubic increase in runtime, the focus of this discussion is on the size of the tetrahedrons. In this thesis, the resolution of cells is given by their radius R in average tetrahedron side-lengths[26]. According to common wisdom (see appendix H), an average tetrahedron side-length should be about the distance between two lattice nodes. For the following this shall be referred to as the cells native resolution R_{native}. Several simulations (simulation 4) are performed in a square Poiseuille-flow using a cell with different cell resolutions R calculated as follows.

$$R = S_{\text{resolution}} R_{\text{native}} \tag{306}$$

Here $S_{\text{resolution}}$ is a scaling factor. $S_{\text{resolution}} < 1$ produce coarser meshes than the native one, while $S_{\text{resolution}} > 1$ produce finer ones.

[26]The average tetrahedron side-length refers to the value given to gmsh [85]. The actual value in the mesh produced by gmsh will differ slightly.

4 Low resolution jagged cells

Box:	$500 \times 55 \times 55$
	$L_0 = 375\,\text{nm}$
Fluid:	PTT::mc0_49 (fluid 2)
BC:	body-force (BC 13.3.2)
Cell:	Young's modulus $E = 10\,\text{kPa}$
	Radius, Native resolution $R = 7.5\,\mu\text{m}, 20$

It shall be noted, that the meshes produced by gmsh [85] will differ slightly between versions and even between computers. The influence of which can be seen in figures 39 and 40, which have been produced using the same scaling factor. The relative change of the surface are $\frac{\Delta A}{A_0}$ is used as a proxy for the observed instability here.

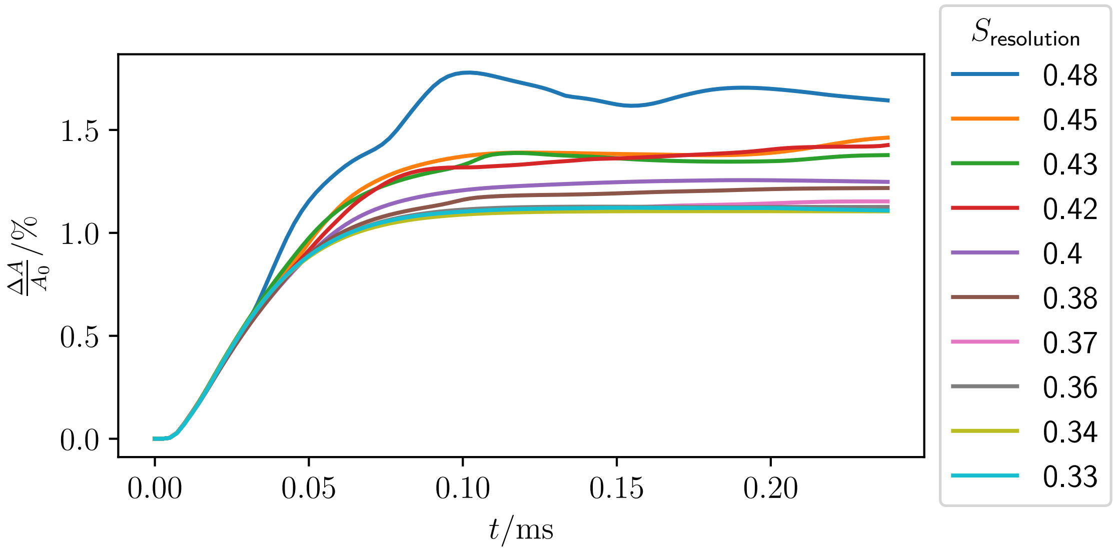


Figure 39: Plot of the surface-area discrepancy as a function of time, for different cell resolutions.

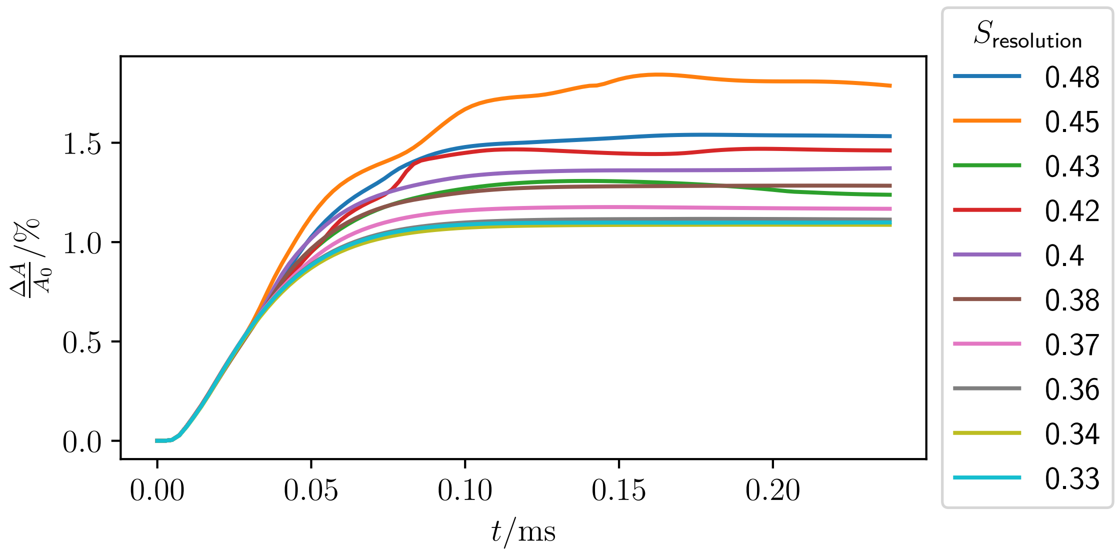


Figure 40: Plot of the surface-area discrepancy as a function of time, for different cell resolutions.

These figures show, that for some resolutions, the area grows initially and then stays at a constant level. This is expected, as even without the instability, the deformation will cause some change in the surface area. These scaling values are therefore considered partially stable. Others vary wildly and are inconsistent for the two slightly different meshes. These will often eventually go towards infinity and crash the simulations. Consequently, these scaling values are considered as unstable. One can see, that all the stable values lie well below unity. So the common choice to have the tetrahedrons similar in size to the lattice constant is not stable. They have to be considerably larger. Observing the curves of the partially stable curves, one can see, that not all of them end with the same value. This is due to the instability being present in all of them. After a certain point it mostly stops growing and is stable. The degree to which it grows differs, leading to the different final values. Figure 41 shows the same output for another simulation (simulation 5), with twice the lattice resolution of the previous one.

5 High resolution jagged cells

Box:	$1000 \times 108 \times 108$
	$L_0 = 187.5\,\text{nm}$
Fluid:	PTT::mc0_49 (fluid 2)
BC:	body-force (BC 13.3.2)
Cell:	Young's modulus $E = 10\,\text{kPa}$
	Radius, Native resolution $R = 7.5\,\mu\text{m}, 40$

Here the same behavior can be observed as described above aside from the partially stable range reaching higher scaling values.

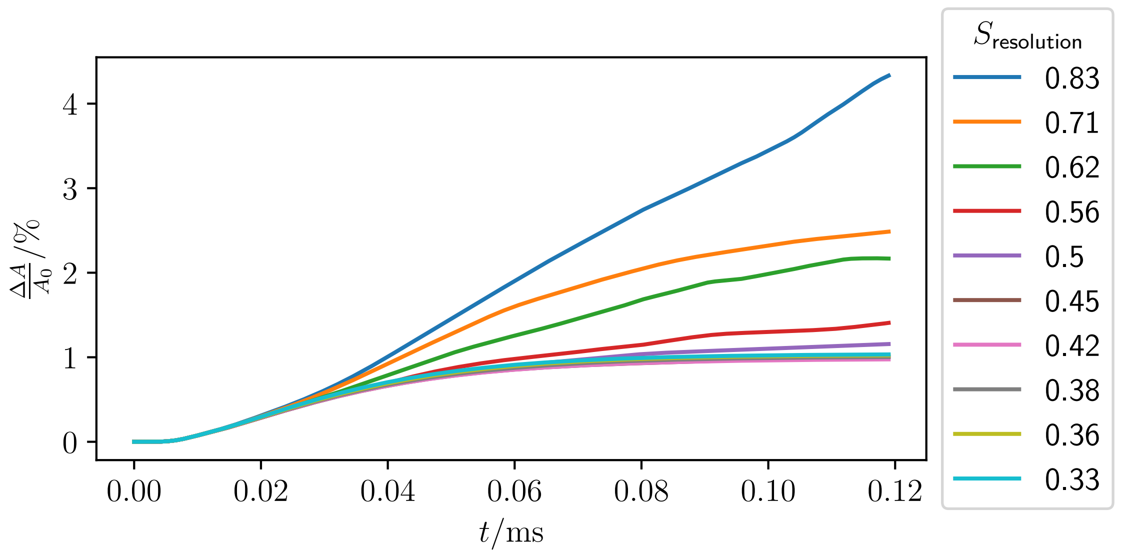


Figure 41: Plot of the surface-area discrepancy as a function of time, for different cell resolutions. Double the lattice resolution compared to figures 39 and 40. Only every second resolution is shown.

In these plots, the final surface area discrepancies of the partially stable curves seem to converge towards a specific value for $S_{\text{resolution}} \to 0$. This trend is shown in figure 42 for both resolutions.

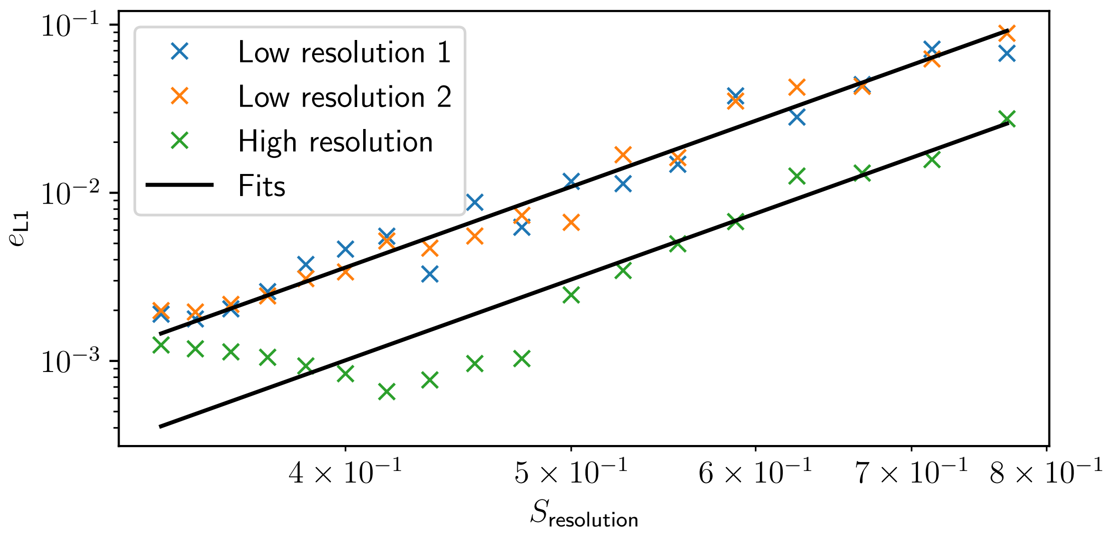


Figure 42: Plot of the $L1$ error due to the jagged instability as a function of the scaling factor, for two different lattice resolutions ($R_{\text{native}} = 20$ and $R_{\text{native}} = 40$).

The best fit for the data is a power-law as follows.

$$\frac{\Delta A}{A_0} = aS^b_{\text{resolution}} + c \tag{307}$$

With the generic parameters a, b and c. The error shown above shall be defined as follows.

$$e_{\mathrm{L1}} = \frac{\Delta A}{A_0} - c \tag{308}$$

The reasoning being, that c represents the change in surface area, which is expected due to the deformation. Only the change exceeding this is attributed to the instability. Fitting the low and high resolution curves independently produces values for b and c, which are the same within the errors. This aligns with the interpretation for c. For b this suggests, that it is a general constant for this type of instability. To honor this interpretation, the data was fitted simultaneously with a common b and c and an a for each resolution. The results can be seen in table 3.

$a_{\text{low res}}$	$a_{\text{high res}}$	b	c
0.34 ± 0.04	0.095 ± 0.010	5.0 ± 0.3	$(0.91 \pm 0.04)\,\%$

Table 3: Optimal fit parameters for a power law fitted to the surface area error caused by the jagged instability.

It should be noted, that the fit does not fit perfectly for the high resolution curve in the region of low scaling. Here the observed area change is close to constant, while the error is expected to go to 0. Therefore, the noise overpowers the signal. This likely causes the $a_{\text{high res}}$ to be slightly larger than it should be. It is roughly a factor 4 smaller than $a_{\text{low res}}$. This could be interpreted as the error scaling with the square of the lattice resolution. This would fit with an area being the basis of this error. However, two data-points is of course too little to state this confidently. Following this interpretation, a linear decrease of the error would require to increase runtime by a power of at least 1.5. While this is not too costly, reducing the resolution linearly and getting a reduction of the runtime by a power of 5 at the cost of absolutely free[27] is of course better. However, there are limits to this scheme. Namely, the reduced-resolution mesh still needs enough resolution to accurately approximate as sphere. This limits the scaling possibility for the low resolution cases in figure 42 and explains why it ends before the low resolution curves have reached their minimum. As simulations are performed with as little resolution as possible one has to use both approaches. Increase the lattice resolution to get a small stability improvement and reduce the cells resolution to the original one for the big improvement. Therefore, in practice there still is a cost associated with avoiding this instability. Consequently, this is only done in this thesis when required. The effect of different scaling factors can be seen in figure 43. The development of the instability is exponential.

[27]Technically the runtime even decreases, but this is negligible.

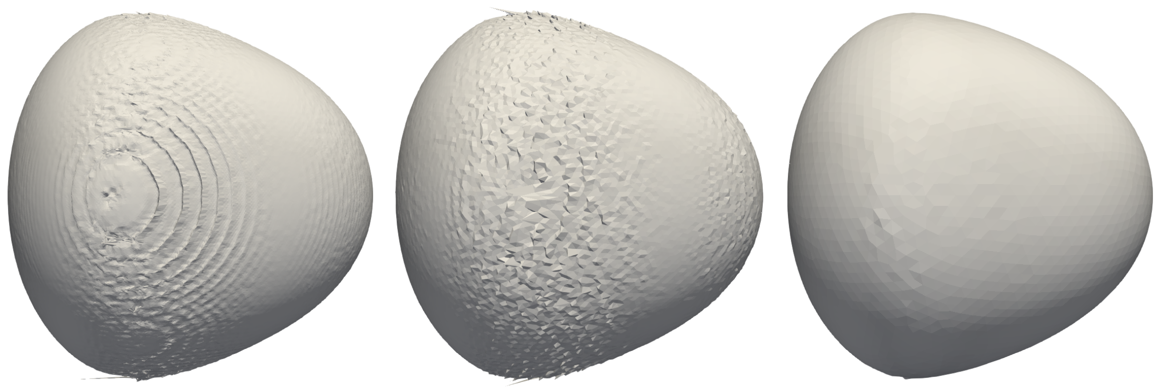


Figure 43: Renders of the jagged cell instability in the large lattice resolution case for $S_{\text{resolution}} = 2.5$, $S_{\text{resolution}} = 1$ and $S_{\text{resolution}} \approx 0.42$.

The cell with increases resolution shows an interesting instability with rings instead of the random peaks on the unscaled cell. These rings differ in their radial coordinate by approximately one lattice-node-distance. This indicates, that the primary reason in this case is related to the lattice-cell interaction. This showcases, that depending on the relative importance of the two promoters of the jagged cells instability, it can look different. The cell with a lower resolution is very smooth, but not perfect. It has small bumps, primarily along the coordinate axes, where the shear-rate in this square channel is the largest. Especially compared to the cell in native resolution, the instability can be considered gone for all practical intents and purposes. Given that the native resolution is typically considered to be correct (see appendix H), the question, whether this scaling is valid arises. This shall be discussed next.

15.2.3. Accuracy

There are few analytically solvable problems for cells in flow. One of them is Roscoe theory (see section 9), which describes the deformation and orientation of an elastic sphere in a pure shear-flow. A simulation (simulation 6) is therefore performed with the scaled mesh as well as with the mesh in native resolution within a pure shear-flow.

6 Roscoe native and scaled cell

Box:	$300 \times 800 \times 300$
	$L_0 = 187.5\,\text{nm}$
Fluid:	Newtonian::water (fluid 1)
BC:	velocity (BC 2)
	$\dot{\gamma} = 1 \times 10^3\,\frac{1}{\text{s}}$
Cell:	Young's modulus $E = 50\,\text{Pa}$
	Radius, Native resolution $R = 7.5\,\mu\text{m}, 40$

The observed shape is analyzed according to Roscoe theory (see section 10). The resulting parameters are listed in table 4.

Data source	α_1	α_2	α_3	θ	$\nu_{\mathrm{tt}}/\frac{1}{\mathrm{s}}$
Scaled simulation	1.071 59	0.927 74	1.000 091	0.717 62	−485.68
Native simulation	1.073 89	0.928 58	0.990 67	0.717 01	−485.45
Theory	1.075 89	0.928 99	1.000 513	0.712 26	−494.66

Table 4: Roscoe parameters determined from the simulation of a scaled and native mesh. Theory rounded to same precision as simulation. Errors suppressed for brevity.

The parameters are rounded according to the determined fit error. This does not always reflect the real error. In this case, the precision is likely lower, than the rounding suggests. The theory can only be numerically solved, so these values also carry an error. This error however is negligible compared to the simulations. One can see, that the parameters only differ by small single-digit percentages at most. In case of the alphas, one should consider their distance from unity. In that case α_1 shows a larger error, than the other parameters. The reason for this is, that a rougher resolution limits how well the "pointiness" of the cell in the direction of its largest axis can be resolved. This numeric smoothing reduces the observed α_1. This can be seen in general for rougher meshes (see appendix X) and is not related to the scaling. Therefore, one can conclude, that the scaling not only increases the stability, but does so while remaining accurate. Contrary to literature, which typically claims native resolution would be necessary (see appendix H), this scaling does not introduce any amount of noticeable error and is a valid technique. Consequently, it will be employed throughout this thesis wherever necessary. Next, the most important tool for speeding up simulations is discussed.

16. Reynolds-Scaling

Reynolds-Scaling is a technique, that allows to decrease simulation runtime significantly for certain simulations. The simulation can be sped up, by increasing the size of the time-step. One way to achieve this is by scaling of the Reynolds number Re. Another way will be discussed in section 17. This section will cover why Reynolds-Scaling is allowed, how it benefits runtime, how it is done and what needs to be considered for more complicated simulations. First, the justification for scaling the Reynolds-number is explained.

16.1. Justification

Scaling the Reynolds number means altering it, which would mean altering a dimensionless number. According to the Buckingham π theorem [39–41], the physics of any equation can be fully captured by the dimensionless numbers. Meaning, altering any dimensionless number would in general mean altering the simulated physics. However, there are some limits in which the altering of the Reynolds number is allowed. Care has to be taken in order to assure, that no other dimensionless number is accidentally altered in this process. Section 1.2.1 already mentioned during the derivation of the Reynolds number, that there is a limit independent of the Reynolds number. Equation (16) is given again in the following to illustrate this.

$$\rho' \frac{\partial \vec{u}'}{\partial t'} + Re\rho'(\vec{u}' \cdot \nabla')\vec{u}' = \mu' \nabla'^2 \vec{u}' - \nabla' p' \tag{309}$$

Here, the primed quantities are the dimensionless versions of the density ρ, velocity $\vec{u}$, dynamic viscosity μ and pressure p. The gradients with respect to time $\frac{\partial}{\partial t}$ and space ∇. Have also been made dimensionless. Evidently, the second term, also called convective term or non-linearity vanishes for $Re \ll 1$. Meaning, the Navier-Stokes equation becomes the non-stationary Stokes equation, which is independent of Re for any sufficiently small Re. Consequently, if one would run the simulation with a scaled Reynolds number Re^*, the flow would remain unaltered given, that the following holds.

$$Re^* \ll 1 \tag{310}$$

The comparison to unity warrants some discussion. The limit is archived if the convective term is small compared to the user terms. This criterion quietly assumes, that all the terms have approximately the same magnitude, with the Reynolds number being the only difference. This is not fully accurate. The nondimensionalization is designed to produce primed values around unity. For the density and the dynamic viscosity, this is trivial as both are (essentially) constants in Newtonian simulations. For the pressure a view considerations need to be made as the nondimensionalization (eq. (7)) repeated in the following is not quite straight forward.

$$p = \frac{\rho_\mathrm{t} u_\mathrm{t} L_\mathrm{t}}{T_\mathrm{t}} p' \tag{311}$$

The typical time T_t is identified as the viscous timescale (eqs. (13) and (15)). With this, the conversion reads as follows.

$$p = \frac{u_\mathrm{t}\mu_\mathrm{t}}{L_\mathrm{t}} p' \tag{312}$$

In illustrative example to better understand this is a pressure driven flow in a pipe. For this a relationship liking all the typical quantities exists (see appendix F.1.1). Inserting the center-line velocity of this flow gives the following conversion.

$$p = \frac{GL_\mathrm{t}}{2^{j+1}} p' = -\Delta p \frac{L_\mathrm{t}}{L2^{j+1}} p' \tag{313}$$

Here, the pressure gradient G, is the difference of pressure along the pipe of length L. The j switches between 2D pipes ($j = 0$) and 3D pipes (j=1). This shows, that aside from a little factor, the typical pressure assumed by the complicated expression above is equivalent to the pressure difference. Consequently, p' is of the order of 1. General flows, do not have such analytical solutions to prove this, but behave similarly. This leaves the velocity to be discussed. In this thesis, the typical velocity is picked as the maximum velocity (see section 1.2.2). Consequently, within most of the simulation volume the dimensionless velocity is significantly below unity. The non-linearity, which is named after its dependence on the square of the velocity, is consequently smaller than the other terms in most of the simulation volume. Furthermore, for a typical flow geometry (see appendix P), the gradient of a velocity is mostly or fully normal to its direction. This reduces the convective term significantly. With these effects together, the non-linearity is considerably smaller than the other terms even with the Reynolds number approaching unity. Therefore, $Re^* \ll 1$ is often interpreted in this thesis as $Re^* < 1$. In general, such a loose interpretation is not advisable. To archive the best results it is recommended to keep $Re^* < 0.01$. However, experience shows, that one can get away with going at least an order of magnitude higher. The discussion in this section and section 1.2.1, which forms the basis of this section was strictly Newtonian so far. However, it extends to Generalized Newtonian fluids (see section 3.5). This requires the additional consideration, that the viscosity and consequently the Reynolds number are strictly speaking location dependent, even in steady-state. Local considerations are difficult, therefore this thesis uses the sufficient condition of enforcing condition for the Reynolds number ($Re^* \ll 1$) everywhere. Determining the maximum value of the Reynolds number for a given simulation can be difficult and at times can only be done after the fact. However, with this decision made, the location dependence has no further influence on the following discussion. Therefore, the following discussion will also suppress this technicality for readability. With the scaling justified, the following discussion covers how this is beneficial to simulation speed.

16.2. Simulation speed

LBM naturally links the size of one time-step to the viscous timescale T_v (see equations (25) and (10)). However, the effects one desires to observe (e.g. advection of a particle) happen on the advective timescale T_a. This results in number of required steps and therefore the runtime T_runtime being proportional to the fraction of these timescales.

$$T_\mathrm{runtime} \propto \frac{T_\mathrm{a}}{T_\mathrm{v}} = \frac{1}{Re} \tag{314}$$

This means by definition (eq. (11)), that the runtime is inversely proportional to the Reynolds number. It follows, that artificially increasing the Reynolds number as much as possible (without violating $Re^* \ll 1$) saves on time and resources. Consequently, it should be used whenever it is valid. How the Reynolds number can be artificially increased is covered next.

16.3. Scaling the Reynolds number

The Reynolds number is scaled, by changing simulation parameters, that influence the Reynolds number. This means, the system being simulated is actually a different one from the one, that is intended to be simulated. It is of course desired to not change the system in a way that would be problematic or confusing. The altered system shall be denoted by an asterisk superscript in the following. There are several parameters influencing the Reynolds number and consequently, there are several options to scale it. The requirement to not alter any other dimensionless number, to not alter the simulated physics, provides additional constraints. Furthermore, the following practical concerns should be considered. Input and output of the simulation will be in the starred system and need to be converted back to the desired system. Depending on which parameter is scaled, keeping the other dimensionless numbers the same requires the scaling of additional quantities. Determining the full extent of the consequences of scaling these quantities can be rather difficult and confusing. To keep things simple, neither the velocity nor the length should be scaled. The reasoning for this is these quantities being the ones, one interacts with the most. Without length or velocity being scaled, the advective timescale remains unaltered. Formally, this is written as follows.

$$T_\mathrm{a}^* = T_\mathrm{a} \tag{315}$$

The Reynolds number shall be scaled by a scaling factor S_{Re} as follows.

$$Re^* = S_{Re} Re \tag{316}$$

As the Reynolds number is defined as a fraction of advective and viscous timescales (see eq. (11)), the viscous timescale must be scaled as follows.

$$T_\mathrm{v}^* = S_{Re} T_\mathrm{v} \tag{317}$$

As the viscous timescale gives the size of stimulation steps (see equations (25) and (10)), this demonstrates the increase in step size. This carries the important caveat, of changing the typical timescale of the time derivative in non-stationary cases. The consequences of this will be explained in more detail later (see subsection 16.6). From the definition of the viscous timescale (eq. (10)) the following holds for the scaling.

$$T_{\mathrm{v}} = S_{Re} T_{\mathrm{v}} \tag{318}$$
$$\frac{\rho^* L^2}{\mu^*} = S_{Re} \frac{\rho L^2}{\mu} \tag{319}$$
$$\frac{\rho^*}{\mu^*} = S_{Re} \frac{\rho}{\mu} \tag{320}$$

Note, that the subscript to indicate typical values has been dropped. This is done, because it is necessary to scale all values consistently and in accordance with the scaling of the typical value. While this looks like a scaling of the kinematic viscosity, this is actually the reason why the kinematic viscosity is avoided in this thesis. If the scaling is defined as scaling the kinematic viscosity, it is unclear whether the density ρ should be scaled, or the dynamic viscosity μ. The density is one of the three base fixed conversions of the LBM (see section 2.3). Therefore, a clear definition is needed. Consequently, a decision must be made here, to either scale the density, or the dynamic viscosity. Choosing either one is valid and works. However, scaling the dynamic viscosity entails the requirement to scale other parameters as well once extensions are considered. For example, the dynamic viscosity appears in the definition of the Capillary number (see eq. (153)). This would carry the requirement to also scale other quantities in the definition of the Capillary number to keep it constant. Extensions will be discussed in a later subsection (see subsection 16.7). Meanwhile, the density is not present in any of the other non-dimensional numbers considered in this thesis. Therefore, in this thesis, the following shall be used for Reynolds-Scaling.

$$L^* = L \tag{321}$$
$$\vec{u}^* = \vec{u} \tag{322}$$
$$\mu^* = \mu \tag{323}$$
$$\rho^* = S_{Re} \rho \tag{324}$$

This means, that all quantities except for the density stay the same. The density is the only value getting scaled. This is also convenient, given that it is rarely looked at. Still, this carries some caveats, which will be explored in the following.

16.4. Caveats

In the following, all relevant points of concern in regard to scaling the density are discussed. They are not sorted in any particular order.

16.4.1. Density specific quantities

Scaling the density entails the requirement of scaling all density specific quantities (like the kinematic viscosity) as well. In this thesis, the use of such quantities is completely avoided, thus trivially fulfilling this requirement without the need to handle it. The kinematic viscosity is by far the most relevant such quantity. This is why avoiding it is stressed here. Others are unlikely to be used.

16.4.2. Pressure from density

LBM provides a relation between the density and the pressure (eq. (26)). Using this on the default density yields an altered value. This might cause concern, however it does not matter as absolute pressures are irrelevant, and the scaling does not affect pressure differences. In fact no specific handling is required for this. Consistently using the conversion factors resulting from the altered density is sufficient even tho this seems counter-intuitive.

16.4.3. Density not unity

In LBM one usually assumes the density to be (close to) unity in lattice units (see section 2.3). The flow or its boundary conditions might create pressure differences, which are represented in LBM by density differences. These are typically small, however the Reynolds-Scaling amplifies them (by the scaling factor). This does not cause issues, given one does not assume the density to be always unity as is suggested in literature [9]. Code (e.g. for the boundary conditions) should use the actual value of the density as is done in *FluidX3D* (see section 13.1) to achieve the best results.

16.4.4. Mach number

Reynolds-Scaling increases the Mach number Ma, which is an important stability criterion in LBM (see section 2.7). The change in the Mach number needs to be observed to assure it does not exceed its stable range due to the scaling. Before continuing to more complex models, a few validations are in order.

16.5. Validation

To demonstrate the efficacy of Reynolds-Scaling a few simulations are performed using *FluidX3D*. All of them are done on pairs. One simulation with the scaling $S_{Re} = 1$, meaning unscaled and the other one with $S_{Re} = 10$. Two different geometries are used. First, the geometry seen in figure 44 will be used. Later, a pure shear-flow will be used to use Roscoe theory (see section 9) for validating simulations with cells.

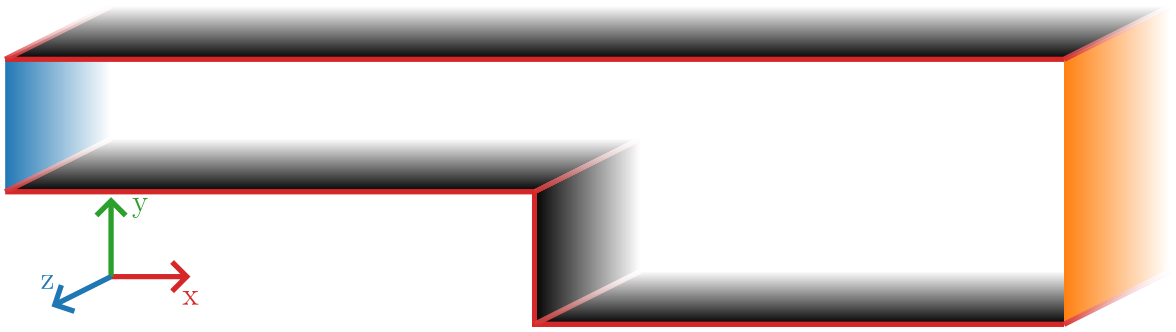


Figure 44: Illustration of the geometry for the validation of Reynolds-Scaling: The flow enters at the inlet and exits at the outlet. In the middle, the channel abruptly expands. It exhibits translational symmetry along the z-axis and is therefore effectively 2D.

The channel gets twice as wide at half its length. This offers a somewhat challenging geometry, with a velocity that is not constant along streamlines. The channel is periodic along the z-axis, making it a 2D channel. It is driven by a pressure difference. The exact dimension and remaining parameters vary and are discussed for each simulation individually. A detailed validation for each part of the section is found in appendix R. Here, only general findings and some errors (according to appendix S) are given. Typically, the velocity along the center-line of the inlet is used for illustration. For a Newtonian fluid (validation in appendix R.1) this can be seen in figure 45.

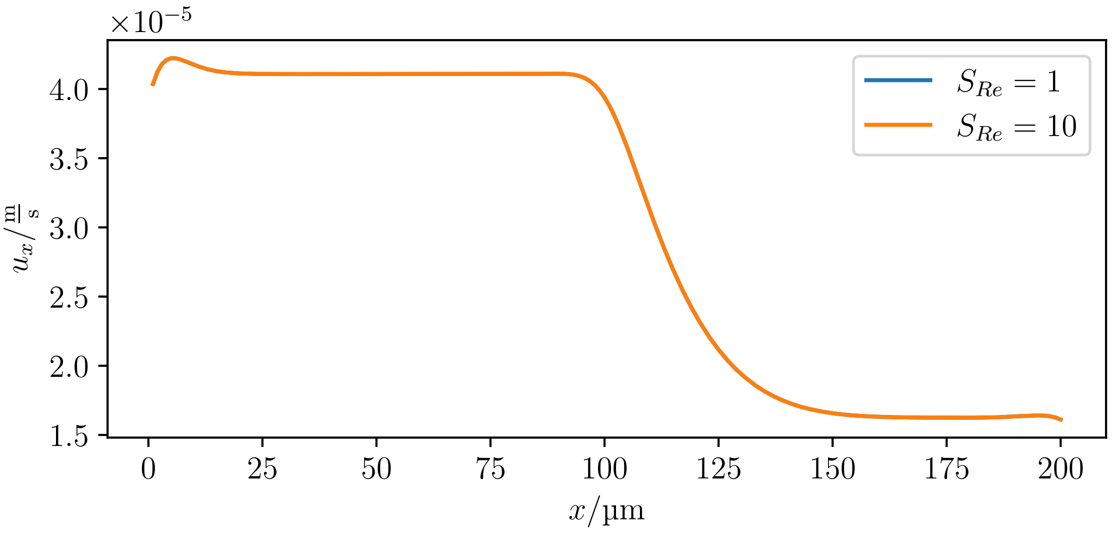


Figure 45: Plot of the x-component of the velocity field along the center-line of the narrow part of the channel. The maximum L1 error of the velocity is $|e_{\text{L1}}| < 2 \times 10^{-4}$.

To the naked eye, this agreement is great. However, $e_{\text{L1}} < 2 \times 10^{-4}$ is a few orders of magnitude above what is achievable using LBM. This shows, that even though Reynolds-

Scaling clearly works well here, it is still an approximation. Next, a few words on time-dependent behavior under Reynolds-Scaling are required.

16.6. Time-dependent behavior

From the arguments above it might seem like Reynolds-Scaling perfectly reproduces all effects, given the Reynolds number is small enough. However, viscous effects in non steady-state systems are actually reproduced at the wrong timescale. The reason is simply, that Reynolds-Scaling by definition (eq. (317)) alters the viscous timescale and therefore effects on that timescale are altered. Whenever the derivative term is relevant, Reynolds-Scaling alters the simulation output. Given the viscous timescale is far enough from other timescales in the system (which for Generalized Newtonian fluids at $Re^* \ll 1$ is always the case), the behavior is slower (for $S_{Re} > 1$), but otherwise identical. The go-to example for such an effect would be the startup of the flow. The change in velocity 3 µm behind the center of the inlet during the startup of the flow form appendix R.1 can be seen in figure 46.

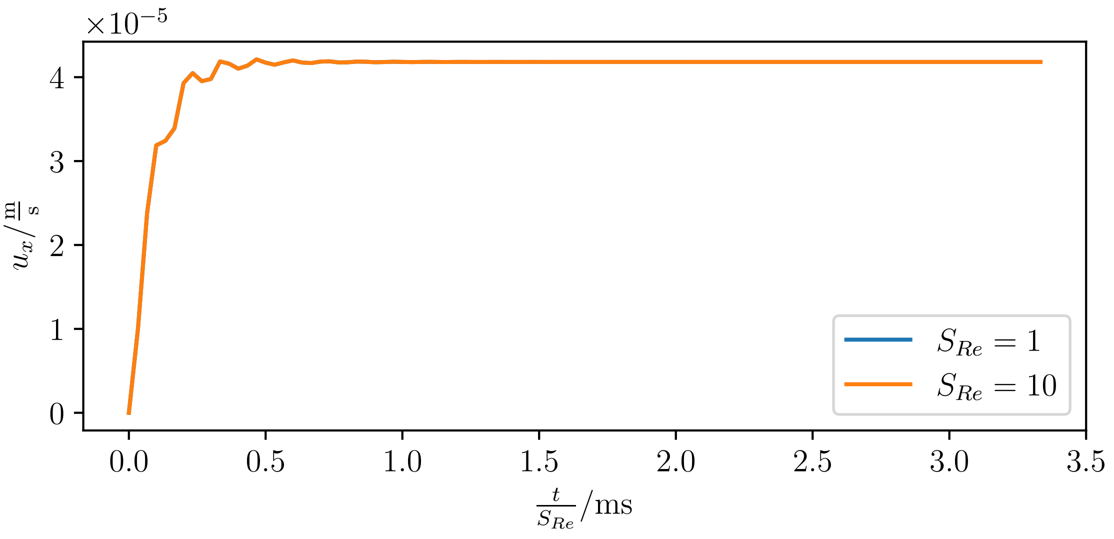


Figure 46: Plot of the x component of the velocity field 3 µm behind the center of the inlet as a function of time.

Here, one can see, that these curves are identical aside from the scaled curve developing slower as discussed above. Even the noise produced by the boundary condition is accurately reproduced. Once the time derivative term disappears (in steady state), the output is near identical, as was already shown above. A shear-thinning Generalized Newtonian fluid shows the same behavior (see appendix R.2) This concludes the coverage of simple LBM cases. There can be additional contributions to the Navier-Stokes equation (see section 2.5). How those are to be treated is covered in the following.

16.7. Additional Navier-Stokes contributions

For viscoelastic fluids (see section 3.7) and cells (see section 6), additional terms are added to the Navier-Stokes equations. This section discusses how these are affected by the Reynolds-Scaling. First, viscoelastic fluids are discussed.

16.7.1. Viscoelastic fluids

Viscoelastic fluids couple their additional stress term into the Navier-Stokes equation. Meaning, the right-hand side of equation (1), gets an additional term as follows.

$$\rho \frac{\partial \vec{u}}{\partial t} + \rho(\vec{u} \cdot \nabla)\vec{u} = \mu \nabla^2 \vec{u} - \nabla p + \nabla \cdot \underline{\tau} \tag{325}$$

The non-dimensionalization of $\underline{\tau}$ is already known from the standard notation of viscoelastic fluids (see appendix D). This is done by identifying η_p as the typical viscosity. For the PTT model (see section 3.7.4), the additional term in the non-dimensional form becomes as follows.

$$\frac{T_\mathrm{t}}{\rho_\mathrm{t} u_\mathrm{t}} \frac{\mu_\mathrm{t}}{\lambda L_\mathrm{t}} \nabla' \cdot \underline{C} = \frac{L_\mathrm{t}}{u_\mathrm{t}} \frac{1}{\lambda} \nabla' \cdot \underline{C} = \frac{2\sqrt{\epsilon}}{Wi} \nabla' \cdot \underline{C} \tag{326}$$

It should be noted, that PTT is used here specifically in order to have a specific definition of Wi. PTT is also the default viscoelastic model in this thesis and therefore of primary importance. For other viscoelastic models, this looks similar. The main difference is the occurrence of additional factors around Wi. These always depend on the additional dimensionless parameter each model carries. It shall be noted, that in this thesis viscosity shuffling is always used for viscoelastic fluids (see section 18). This technically introduces an additional term to the equation. Mathematically, this works out to another dampening term (term containing the viscosity) in the Navier-Stokes equation. As this is inconsequential for the present discussion it was omitted here. The Weissenberg number Wi neither contains the viscous timescale nor the density. Consequently, neither of them affect viscoelastic behavior. However, a new timescale, the relaxation time λ is introduced. This new timescale is not scaled, and consequently has different scaling behavior than the viscous timescale. Also, these timescales do interact. So, in contrast to the Newtonian and Generalized Newtonian case, the startup is not just scaled. The startup for viscoelastic fluids is just plain different. This can be seen in figure 47.

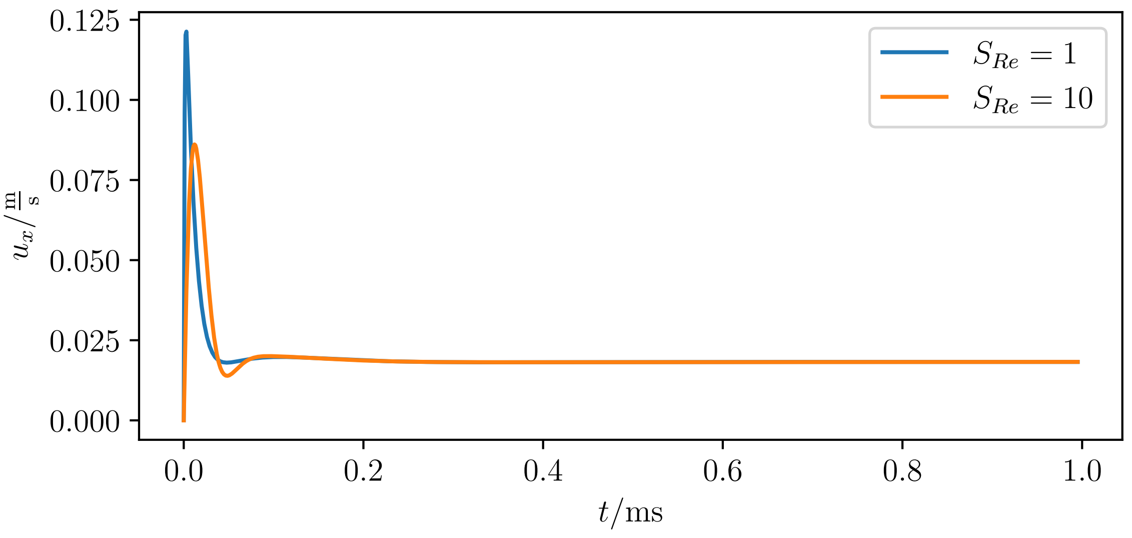


Figure 47: Plot of the x component of the velocity field 3 µm behind the center of the inlet as a function of time.

This means that Reynolds-Scaling may not be used for non-steady-state processes involving viscoelastic fluids. However, it is particularly viscoelastic fluids, where Reynolds-Scaling is the most useful. This example simulation, takes orders of magnitude longer to run than the CY simulation used above for Generalized Newtonian fluids. Still, it is technically not fully in steady-state yet. This is due to the typically large values of the relaxation time λ, that the fluids used in this thesis (see section 19) have. Similar to the runtime for observing advective effects being inversely proportional to the Reynolds number, the runtime required for a viscoelastic fluid to relax is as follows.

$$T_{\mathrm{runtime}} \propto \frac{Wi}{Re} \propto \frac{\lambda}{t_{\mathrm{v}}} \tag{327}$$

This makes especially the interesting cases of $Wi > 1$ even more expensive. Reynolds-Scaling can alleviate this as long as the desired effect is observed in a fluid in steady-state. Steady-state still allows for some movement. Such a case will be discussed next.

16.7.2. Cells

Cells couple their additional force term into the Navier-Stokes equation. Meaning, the right-hand side of equation (1), gets an additional term as follows.

$$\rho \frac{\partial \vec{u}}{\partial t} + \rho(\vec{u} \cdot \nabla)\vec{u} = \mu \nabla^2 \vec{u} - \nabla p + \vec{f} \tag{328}$$

Note, that $\vec{f}$ is a force density, rather than a force. A general non-dimensionalization can be derived from the scaling of the volumetric elastic forces discussed in section 6.5. From this, one can gather, that the force densities are non-dimensional aside from a factor made from the Young's modulus E and the cell radius R. This is listed in the following.

$$\vec{f} = \frac{E}{R}\vec{f'} \tag{329}$$

Consequently, the non-dimensionalization of the additional term reads as follows.

$$\frac{T_\mathrm{t}}{\rho_\mathrm{t} u_\mathrm{t}} \frac{E}{R}\vec{f'} = \frac{L_\mathrm{t}^2}{\mu_\mathrm{t} u_\mathrm{t}} \frac{E}{R}\vec{f'} = \frac{E}{\mu_\mathrm{t}\dot{\gamma}_\mathrm{t}}\vec{f'} = \frac{16}{Ca_\mathrm{K}^2}\vec{f'} \tag{330}$$

This uses the Capillary number according to eq. (153). The Capillary number Ca_K neither contains the viscous timescale nor the density. Consequently, Reynolds-Scaling is expected to work. For this a pure shear-flow (see appendix P.1) is used to validate the Roscoe parameters (see section 9). This is done in appendix R.4. The results can be seen in table 5.

Simulation	α_1	α_2	α_3	θ	$\nu_\mathrm{tt}/\frac{1}{\mathrm{s}}$
$S_{Re} = 1$	1.1873	0.8442	1.006 71	0.5227	−7683
$S_{Re} = 10$	1.1905	0.8425	1.005 58	0.5333	−7577.6

Table 5: Roscoe parameters determined from the simulation with $S_{Re} = 1$ and $S_{Re} = 10$. Values rounded in accordance to errors. Errors suppressed for brevity.

These values do not quite align. However, they are only off by a few percent. Within the typical error range for Roscoe simulations (see appendix X), this can be considered to be identical. So, aside from some startup effects, Generalized Newtonian Roscoe simulations are not affected by Reynolds-Scaling. Putting the cells through a channel with changing geometry is not steady-state. One would expect, that this would show significant disagreement. This is done in appendix R.5. However, the result is, that Reynolds scaling mostly still works in this case. Apparently, it is more robust than expected. Next, the combination of a cell in a viscoelastic fluid is considered.

16.7.3. Cells in viscoelastic fluid

First, the Roscoe simulation is repeated with a viscoelastic fluid. This is done in appendix R.6. The results are presented in table 6.

Simulation	α_1	α_2	α_3	θ	$\nu_\mathrm{tt}/\frac{1}{\mathrm{s}}$
$S_{Re} = 1$	1.2094	0.8526	0.976	0.4325	−5090
$S_{Re} = 10$	1.2075	0.8519	0.9771	0.4393	−5166.3

Table 6: Roscoe parameters determined from the simulation with $S_{Re} = 1$ and $S_{Re} = 10$. Values rounded in accordance to errors. Errors suppressed for brevity. α_1 and α_2 match between scaled and unscaled simulation within their respective errors.

These values do not quite align. However, they are only off by a few percent. Within the typical error range for Roscoe simulations (see appendix X), this can be considered to be identical. So, aside from some startup effects, viscoelastic Roscoe simulations are not affected by Reynolds-Scaling. Interestingly, the values here are not the same as the ones found for a Generalized Newtonian fluid. This is despite the simulation matching in the vast majority of parameters. This is an interesting discussion for a later section (see section 21). For the viscoelastic fluids a cell in Poiseuille geometry (see appendix P.2) is considered before the suddenly widening channel. This is done as an intermediate step between the valid simulations and the suddenly widening channel. Poiseuille flow also violates the conditions for Reynolds-Scaling. However, it is not as dynamic as the suddenly widening channel. The cell in Poiseuille flow is examined in appendix R.7. Here, some disagreement can be seen. The deformation is not fully accurate. This has been expected. However, the error is small enough to justify the use of Reynolds-Scaling for this example as well. Finally, appendix R.8 evaluates the cell in the suddenly widening channel. The result is similar to the same validation for the Generalized Newtonian fluid. The agreement is no longer perfect, but still usable. This leads to a somewhat surprising conclusion, which will be discussed next.

16.8. Reynolds-Scaling Conclusion

Reynolds-Scaling is never exact. Even in the cases where the simulation is in steady-state and the scaling is fully valid, there are tiny discrepancies. However, the more interesting result is actually the cases not in steady-state, which are technically invalid candidates for Reynolds-Scaling. For a cell moving through a Poiseuille-flow (see appendix R.7) or a suddenly changing geometry (see appendices R.5 and R.8), Reynolds-Scaling should not be valid. This is because the time derivative term does not vanish. Time-dependent effects get scaled with Reynolds-Scaling. However, the results show, that even in these cases Reynolds-Scaling yields agreement close enough to call it valid. This robustness is a surprising result. The reason for this is likely, that the velocities in these examples change with the advective timescale, while Reynolds-Scaling acts on the viscous timescale. Meaning, they change slowly. Or in short, the time derivative term is small and therefore, it does not matter much even without vanishing. Reynolds-Scaling only affects results significantly for the startup of simulations, which is particularly long for viscoelastic fluids. In total Reynolds-Scaling is a useful tool, that gets used regularly. However, the limits for the Reynolds and Mach numbers need to be observed. This sometimes prevents the use of this scaling. There is another much simpler scaling technique, which does see some use. This will be discussed next.

17. dt Scaling

With all the difficulties and caveats associated with Reynolds-Scaling, one might look for a more straight forward way to decrease runtime. One such approach, that is noteworthy is *dt* Scaling. Its basic idea is to just make bigger time-steps by making adjustment to the quantities in lattice units (see section 2.3). This works as follows. First the *dt* scaling factor S_{dt} is introduced.

$$\mu^*_{\mathrm{LU}} = S_{\mathrm{dt}}\mu_{\mathrm{LU}} \tag{331}$$

The starred quantities represent the quantities in the scaled system. This changes μ_{LU} from its default value of $\frac{1}{6}$. This has two consequences. First, the time conversion factor changes to the following (see equation (37)).

$$T^*_{\mathrm{c}} = S_{\mathrm{dt}}\frac{\rho_{\mathrm{c}}L^2_{\mathrm{c}}}{\mu_{\mathrm{c}}} = S_{\mathrm{dt}}\rho_{\mathrm{c}}L^2_{\mathrm{c}}\frac{\mu_{\mathrm{LU}}}{\mu_{\mathrm{SI}}} = S_{\mathrm{dt}}\frac{\rho_{\mathrm{c}}L^2_{\mathrm{c}}}{6\mu_{\mathrm{SI}}} \tag{332}$$

This therefore results in a direct scaling of the time-step as follows.

$$\Delta t^*_{\mathrm{SI}} = T^*_{\mathrm{c}}\Delta t_{\mathrm{LU}} = T^*_{\mathrm{c}} = S_{\mathrm{dt}}\Delta t_{\mathrm{SI}} \tag{333}$$

Secondly, the relaxation time changes to the following (see equation(25)).

$$\tau^*_{\mathrm{r,\,LU}} = \frac{\mu^*_{\mathrm{LU}}}{\rho_{\mathrm{LU}}c^2_{\mathrm{s}}} + \frac{\Delta t_{\mathrm{LU}}}{2} = \tau_{\mathrm{r,\,LU}} + \frac{(S_{\mathrm{dt}}-1)\mu_{\mathrm{LU}}}{\rho_{\mathrm{LU}}c^2_{\mathrm{s}}} = \tau_{\mathrm{r,\,LU}} + \frac{1}{2}(S_{\mathrm{dt}}-1) = \frac{1}{2}(S_{\mathrm{dt}}+1) \tag{334}$$

The spatial truncation error of LBM is proportional to $\left(\tau_{\mathrm{r}} - \frac{\Delta t}{2}\right)^2$ [102]. Therefore, *dt* Scaling causes an error proportional to S^2_{dt}. This quickly gets out of hand and effectively limits the maximum possible scaling to around two to maybe three. While this provides a runtime benefit (at the cost of accuracy), the maximum effect is minuscule compared to what Reynolds-Scaling can do. Therefore, this thesis avoids this scaling. Next, the last, and arguably most important algorithm in this thesis is covered.

18. Viscosity shuffling

The algorithm in this section is arguably the most important one in this thesis, because it enables the simulation of realistic viscoelastic fluids. Most of this section and its associated appendices has already been published [1]. This section will explore the motivation for viscosity shuffling, typical fluid parameters, the algorithm itself, the validation of the algorithm and some unexpected use-case. First, the reason why special handling is required will be discussed.

18.1. Necessity for the algorithm

The difficulty in simulating viscoelastic fluids is to a large part due to the parameters of real viscoelastic fluids and not the models themselves. The stress caused by the polymers in a viscoelastic solution can be separated into a viscous stress and a normal stress. Both of these are proportional to the polymer viscosity η_p (see section 3). Depending on model and parameter choice, their magnitude is typically similar. These contributions enter the LBM as a stress (see section 2.5.2). The LBM itself provides the stress contribution from the solvent viscosity η_s. If the stress from the polymer contributions exceeds the solvent stress, the LBM algorithm gets overwhelmed. This is comparable to IBM producing forces, that are too large (see section 6.7). Some populations become negative, leading the simulation to become unstable and likely causing it to crash. With this, the fraction of these viscosities can be seen as a stability criterion. This viscosity ratio R_η shall be defined as follows.

$$R_\eta = \frac{\eta_\mathrm{p}}{\eta_\mathrm{s}} = \frac{\eta_0 - \eta_\infty}{\eta_\infty} \tag{335}$$

For the models used in this thesis, the polymer and solvent viscosity can be related to the zero-shear viscosity η_0 and the infinite-shear viscosity η_∞ (see section 3.3). This allows the determination of this ratio for general fluids without the need to describe them through a model. It shall be noted, that a relation of viscosities is a somewhat common dimensionless number. No standard definition exists, so care must be taken to distinguish how the different viscosity ratios one can find in literature are defined. There is no firm limit for stability as a function of R_η, as this is dependent on the model used. Also, the fraction of the stresses is a function of the Weissenberg number Wi (see section 3.4), making this a stability criterion as well. Most authors only mention the Weissenberg number. Existing LBM algorithms reach relevant Weissenberg numbers, but fail to exceed $R_\eta = 10$ (see appendix T), even with workarounds like artificial diffusivity. Consequently, fluids exceeding this value cannot be modeled using current methods and require a new algorithm. Next, the typical viscosity fractions for real viscoelastic fluids are discussed.

18.2. Typical viscosity ratios

This thesis is interested in biofabrication and associated fields. Such use-cases require the solution to not be toxic. Consequently, the primary solvent used for the fluids (termed "bio-inks") is water. Therefore, the solvent viscosity is typically around $\eta_\mathrm{p} \approx 1\,\mathrm{mPa\,s}$. At the same time a high zero-shear rate is desired (see section 19.4). Consequently, the viscosity ratio is typically high. Furthermore, viscosity behavior exists on an exponential scale. Meaning, for a noticeable difference to occur, one must increase the viscosity by an order of magnitude. Consequently, there are often several orders of magnitude between the zero-shear viscosity and the infinite shear viscosity. The alginate solution (fluid 13) used in the present work as an example for a popular bio-ink has a viscosity ratio of $R_\eta = (48.2 \pm 0.4) \times 10^3$. The lowest concentration of the least viscous shear-thinning fluid used in this thesis (fluid 2) still has a viscosity ration of $R_\eta = 18.7 \pm 0.4$. It should be noted, that this fluid is not designed for biofabrication, but for related characterization. The fluids of this thesis will be described in detail later (see section 19). There are also plenty of examples from literature for popular bio-inks. A few are listed in table 7. All of them have viscosity ratios orders of magnitude above what is possible with existing LBM algorithms.

Author	Material	$\eta_0/\mathrm{Pa\,s}$	$\eta_\infty/\mathrm{Pa\,s}$	R_η	extracted from
Amorim [103]	pre-crosslinked alginate	$> 1 \times 10^4$	$< 3\times10^{-1}$	$> 3 \times 10^4$	Fig. 3b
Pössl [3]	blend	2×10^2	5×10^{-1}	4×10^2	Figure 4
Paxton [7]	Alginate	5×10^4	$< 5 \times 10^1$	$> 1 \times 10^3$	Fig. 4b
Jalaal [104]	pluronic F-127	$> 1 \times 10^3$	$< 1\times10^{-1}$	$> 1 \times 10^4$	FIG. 6. (e)

Table 7: Compilation of literature data on typical bio-inks showing that a very high viscosity ratio R_η is a generic feature of these liquids. Reproduced from [1].

One can easily see, that in order to simulate realistic bio-inks a new algorithm is required. This is presented next.

18.3. Shuffling algorithm

For many realistic viscoelastic fluids, the polymer stress overpowers the viscous stress of the solvent. This is the key difficulty in simulating them. Note, that polymer stress is a term for the additional stress, the viscoelastic fluid produces. Most bio-inks and other viscoelastic fluids are polymer solutions justifying this name. The concepts discussed do not only apply to polymer solution. While the viscous solvent stress is handled efficiently and reliably by the LBM algorithm, the polymer stress enters the LBM equations as an additional source term (see section 2.5.2). The relative magnitude of the two terms is given by the viscosity ratio R_η (see eq. (335)). For many situations the viscosity ratio is of the order of $10^2 - 10^3$ (see table 7) The key idea of the shuffling scheme is to artificially increase the LBM viscous stress while at the same time reducing the polymer stress by the same amount. This holds the total stress constant, but effectively decreases the ratio of the terms. This can be seen by reconsidering eq. (73), which is repeated in the following.

$$\underline{\sigma} = 2\eta_{\mathrm{s}}\underline{D} + \underline{\tau} \tag{336}$$

The total stress $\underline{\sigma}$ is given as the sum of the solvent stress proportional to the strain-rate tensor $\underline{D}$ and the polymer contribution $\underline{\tau}$. Now, one can add a 0 to the right-hand side as follows.

$$\underline{\sigma} = 2\eta_{\mathrm{s}}\underline{D} + \underline{\tau}_{\mathrm{shuffle}} + \underline{\tau} - \underline{\tau}_{\mathrm{shuffle}} \tag{337}$$

Here $\underline{\tau}_{\mathrm{shuffle}}$ is the "shuffled" stress, which gets transferred between the terms. It shall be defined as a viscous stress as follows.

$$\underline{\tau}_{\mathrm{shuffle}} = 2\eta_{\mathrm{shuffle}}\underline{D} \tag{338}$$

Here η_{shuffle} is the shuffled viscosity. Its magnitude will be discussed shortly. First, this definition in inserted into the total stress to write the following.

$$\underline{\sigma} = 2(\eta_{\mathrm{s}} + \eta_{\mathrm{shuffle}})\underline{D} + (\underline{\tau} - 2\eta_{\mathrm{shuffle}}\underline{D}). \tag{339}$$

The viscosity used in the LBM solver is set to $\eta_{\mathrm{s}} + \eta_{\mathrm{shuffle}}$. The polymer stress determined by the constitutive equation (see sections 3.7 and 4) is reduced by the amount $2\eta_{\mathrm{shuffle}}\underline{D}$. This is done for every time-step before $\underline{\tau}' = \underline{\tau} - 2\eta_{\mathrm{shuffle}}\underline{D}$ is coupled into the LBM. With this algorithm, the stress contributions of the LBM and the polymer components are balanced, which vastly improves numerical stability. Critically, this viscosity shuffle algorithm is mathematically exact. Contrary to other workarounds such as adding artificial diffusivity [27, 31, 33, 36] viscosity shuffling does not affect the physics of the simulated system. In terms of stability, one could define an effective viscosity ration R'_η defined from the magnitudes of the shuffled stresses as follows.

$$R'_\eta < \frac{\max(\eta_{\mathrm{p}}, \eta_{\mathrm{shuffle}})}{\eta_{\mathrm{s}} + \eta_{\mathrm{shuffle}}} \tag{340}$$

As the stresses are of course tensors, defining their magnitude is not straight forward. At low Weissenberg numbers, the viscous stress dominates. Therefore, subtracting a viscous stress from the polymer stress reduces it. However, only as long as the shuffle viscosity is smaller than the viscosity contribution from the polymers, which drops with the Weissenberg number. For large Weissenberg numbers, the elastic stress dominates. Consequently, the subtraction does not alter the magnitude of the polymer stress significantly. This is expressed by the maximum function in the equation above. First consider the case of $\eta_{\mathrm{shuffle}} \leq \eta_{\mathrm{p}}$. The equation for R'_η becomes as follows.

$$R'_\eta < \frac{\eta_\mathrm{p}}{\eta_\mathrm{s} + \eta_\mathrm{shuffle}} \in \,]1, R_\eta] \tag{341}$$

Depending on the choice of η_shuffle values down to nearly unity are achievable. This is assuming a large R_η and consequently $\eta_\mathrm{p} + \eta_\mathrm{s} \approx \eta_\mathrm{p}$. Without a large R_η, this algorithm is not required. Now consider the case of $\eta_\mathrm{shuffle} \geq \eta_\mathrm{p}$. The equation for R'_η becomes as follows.

$$R'_\eta < \frac{\eta_\mathrm{shuffle}}{\eta_\mathrm{s} + \eta_\mathrm{shuffle}} \approx 1 \tag{342}$$

As $\eta_\mathrm{shuffle} \geq \eta_\mathrm{p} \gg \eta_\mathrm{s}$, the effective viscosity ratio is equal to unity irrespective of the actual choice of shuffling viscosity. A ratio of unity is well within the stable range of preexisting algorithms. It is even stable enough to not require any artificial diffusivity. This means, that all viscoelastic simulations become stable using this algorithm with a reasonable choice of η_shuffle. The limit is completely removed. For many simulations, one could get away with a smaller shuffling viscosity, but *FluidX3D* sets it as follows by default.

$$\eta_\mathrm{shuffle} = \eta_\mathrm{p} \tag{343}$$

This is done, because this is a reasonable default, for minimizing failed simulations. There are some exotic cases, where a slightly higher shuffling viscosity is required for stability. It is important to note, that the shuffling viscosity needs to be defined in respect to the highest viscosity in the system (consider section 4.3.2). There is a singular trade-off assuming, that the lattice relaxation time is maintained at its optimal value of unity. The viscosity shuffling scheme increases the LBM viscosity and therefore decreases the LBM time-step. Next, this shuffling approach is validated for (semi-)analytically solvable problems with realistic parameters. Furthermore, its accuracy is compared to existing algorithms.

18.4. Validation and accuracy

The validation should consider different shuffling viscosities, different fluids and multiple geometries. All of this is done in appendix U. These are the validations from the paper (see reference [1]). It should be noted, that all these validations have been rerun with a current version of *FluidX3D*. This is to assure, that nothing of relevance has changed since the publication of said paper. Compared to the previous publications, the need to initialize the alginate simulations in a pre-stretched state has been removed. Also, the errors have been reduced considerably. In some case, the errors are halved. The validation demonstrates, that shuffling is valid for a great range of parameters. The largest errors through the validation is 3 % and it is likely not caused by the shuffling itself. Consequently, it can be used without second though. *FluidX3D* uses it by default without requiring user input. In terms of accuracy, it is prudent to not only compare against (semi-)analytical solutions, but to also consider literature. Among the preexisting publication listed in appendix T, Malapsinas *et al.* [27] is the best candidate for comparison. The authors studied an Oldroyd-B fluid. Their simulation geometry was a planar Poiseuille flow. The simulated velocity profile was compared to an exact analytical solution, to retrieve an error. Most parameters are clearly given. The rest can be guessed with reasonable accuracy. This allows for an honest head-to-head comparison. Malapsinas *et al.*define the Weissenberg number Wi in its simple common form as follows.

$$Wi = \lambda\dot{\gamma} \tag{344}$$

This uses the shear-rate $\dot{\gamma}$ and the model dependent relaxation time λ. They define a viscosity fraction R_ν as follows.

$$R_\nu = \frac{\nu_\mathrm{s}}{\nu_\mathrm{p} + \nu_\mathrm{s}} = \frac{1}{R_\eta + 1} \tag{345}$$

This is defined from the kinematic solvent viscosity ν_s and the kinematic polymer viscosity ν_p. This can be directly transformed into the viscosity ratio definition used in this thesis (R_η). Furthermore, Malapsinas *et al.*, also define the L2 error differently than in the present work (see appendix S). Their definition is as follows.

$$E_\mathrm{u} = \frac{1}{u_\mathrm{max}}\sqrt{\frac{1}{N}\sum|u_\mathrm{simulation} - u_\mathrm{theory}|^2} \tag{346}$$

Here, E_u is the L2 error, u_max is the maximum theoretical viscosity in the channel, $u_\mathrm{simulation}$ is the simulated velocity field, u_theory is the analytical solution, and N is the number of data-points. It should be noted, that in eq. (61) of reference [27], there is a typographic mistake. The prefactor of 4 should be an 8 instead. These definitions are used in accordance to Malapsinas *et al.*, to reduce confusion. Given, that Malapsinas *et al.*define all quantities in respect to dimensionless numbers, one quantity must be set to define the scale. In the present work $\eta_\mathrm{s} = 1\,\mathrm{mPa\,s}$ is chosen as a base for this system. Simulation 7 is run with 16 different combinations of parameters.

7 Comparison to Malapsinas *et al.*

Box:	$2 \times N + 2 \times 2$
	$L_0 = \frac{1}{N}20\,\mathrm{\mu m}$
Fluid:	custom Oldroyd-B, $\eta_\mathrm{s} = 1\,\mathrm{mPa\,s}$
BC:	body-force (BC 13.3.2)

The L2 error defined as described by Malapsinas *et al.*as function of lattice resolution can be seen in figure 48. The legend is also defined in accordance with Malapsinas *et al.*.

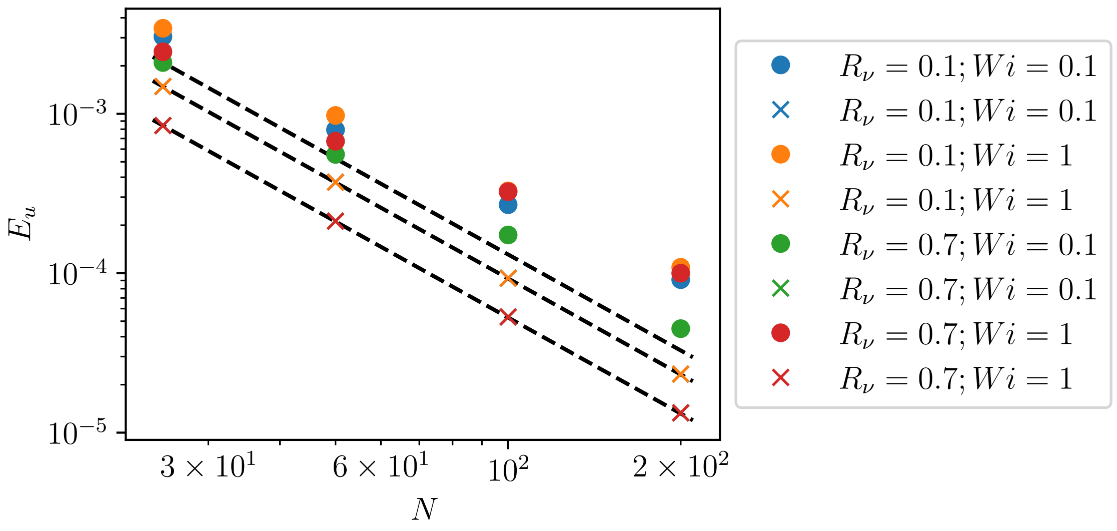


Figure 48: Plot of the L2 error as defined in equation (346) as a function of lattice resolution. The data from Malaspinas *et al.* [27] (dots) is compared to the algorithm from the present work (x). The blue and green crosses are not visible as they overlap exactly with the orange and red crosses, respectively. The dashed lines are power laws with an exponent of -2. Reproduced from [1]

The crosses overlapping in figure 48 demonstrates, that the present algorithm is independent of the Weissenberg number Wi in the studied parameter range. It should be noted, that this thesis does not define a Weissenberg number for Oldroyd-B (see section 3.7.2). It is also quite clear, that the present algorithm has half the error for small lattice resolutions and improves to an order of magnitude in difference for the higher resolutions. The accuracy depends on R_ν. This is acceptable given the error remains small throughout. Finally, it can be seen, that the error of the viscosity shuffling algorithm decreases with the lattice resolution to the power of -2. For small N, Malaspinas *et al.* found a similar decrease in error. At higher N, the reduction is shallower. This difference is likely due to the present work not making use of artificial viscosity. LBM has an error dropping with the square of the resolution. The present thesis is able to fully use the effect. If an artificial viscosity is used, like is commonly suggested in literature, the error has a minimum, that Malaspinas *et al.*is already approaching for $N = 200$. This demonstrates the increased accuracy this algorithm provides compared to existing algorithms, while also allowing to perform simulations in a vastly increased parameter range. A few examples of previously impossible simulations can be seen in appendix V. These two have been simulated anew to demonstrate, that the result has not fundamentally changed since the paper was written. All the topics covered in this appendix will be covered in greater detail in this thesis. This section demonstrated, that the viscosity shuffling viscosity is accurate. Close to every topic in this thesis has been covered in the validation. The two most opposite viscoelastic fluids have been used for validation, to assure a very wide range of parameters works. Viscosity shuffling allows the simulation of previously untreated ground. Furthermore, in cases that are already solvable, it is more accurate than existing solutions. There is only one last feature of viscosity shuffling to cover. It can be used to remove the need for a solvent viscosity.

18.5. Removing the solvent viscosity

In literature, the solvent viscosity is often omitted. There are good reasons for this. For one, for realistic viscoelastic fluids it is a very small contribution to the total viscosity. Also, it often prevents analytical solutions. Consequently, when comparing to such literature, there should be no solvent viscosity in the simulation. With the accuracy achievable with the techniques laid out in this thesis, just picking it small is not good enough. The reason for the necessity of a solvent viscosity is the fact, that the LBM algorithm intrinsically simulates a viscosity. However, with viscosity shuffling, one can set the LBM viscosity to the shuffled viscosity, removing the solvent viscosity altogether. This technically corresponds to $R_\eta \to \infty$, but is stable in *FluidX3D*. Care must be taken, which viscosity is actually currently present in total, to avoid accidentally producing very large Reynolds numbers. With this technique, one could technically accidentally simulate a fluid with no, or even negative viscosity. While this surprisingly does not break the simulation, the output is not very meaningful. This concludes both the discussion of the velocity shuffling algorithm, and the discussion of the methods used in *FluidX3D* and *noFluidX3D* for performing simulations. Next, applications are covered, starting with a description of the fluids used in this thesis.

Part II. Applications

19. Fluids used in this thesis

For a fluid to be suited for biofabrication or the associated characterization, in principle it has to be non-toxic and sufficiently viscous. While the demands on an ideal fluid (see section 19.4) are more complicated, any polymer dissolved in water can technically be used as a bio-ink or cell-carrier fluid. These names are used interchangeably. The fluids used for this thesis are mostly selected by data availability. Alginate solutions are a typical go-to modeling system. Methyl cellulose solutions are useful for characterization. The two cover a wide range of parameters, especially when taking the different concentrations of methyl cellulose into account. Therefore, if something works with these fluids, it can be expected to work with most bio-inks. The same model is often used with different models in literature, depending on application. Consequently, this thesis also fits different models (see appendix F) to the available data. To properly cover the behavior of a viscoelastic fluid, two curves are necessary. One of them is the shear-rate dependent viscosity. For the other, one could use the frequency response, or the first normal stress difference. In rheology, it is apparently more popular to measure the frequency response these days. It is however a lot easier to derive solutions for the first normal stress difference. Consequently, this is used in this thesis. The first normal stress difference N_1 is calculated from the measured normal force using the following equation [105].

$$N_1 = \frac{2F}{\pi R^2} + \frac{3}{20}\rho\omega^2 R^2 \tag{347}$$

Here, F is the measured normal force, R is the radius of the rheometer, ρ is the density of the fluid and ω is the angular velocity of the rheometer. The second term is usually omitted. It only matters at high shear-rate, where data availability is very limited. Furthermore, the available data carries significant error. Therefore, such detailed corrections are not warranted. Consequently, it is also omitted in this thesis. It should be pointed out, that the data quality is limited, because rheology is very error-prone and purely reproducible in general. The fluids are listed in the following. They use the common notation described in section 3.7.1. The fluid models are found in section 3. First, water is introduced as a base-line.

19.1. Water

Water is not really useful as a cell-carrier fluid in biofabrication, because its viscosity is too small. However, it can be used as a baseline. Furthermore, the cytosol within cells behaves like water [106] and in this thesis is typically approximated by water. Also, the other fluids are based on water, or phosphor buffered saline, which is cell-friendly salt water. Water is numbered fluid 1.

1 Newtonian::water	
viscosity	$\eta_s = 1\,\mathrm{mPa\,s}$

Each fluid has such a box for each model, that list the values of all parameters. Next, the methyl cellulose solutions are discussed.

19.2. Methyl cellulose

The methyl cellulose (MC) solution is manufactured by dissolving methyl cellulose in phosphor buffered saline (PBS). The data is from reference [107]. This subsection contains data for three different concentration. These are 0.49 %, 0.59 % and 0.83 %. The fluids are addressed by these percentages. These percentages are by weight. The lower concentrations are both easier to simulate and easier to handle in experiments. Consequently, they are more commonly used in this thesis. The increased concentration mostly scales the viscosity and first normal stress difference. From theory, this should be the only effect. However, the other parameters are also slightly influenced. This fluid is used to fit, the PTT model, the CY model in two different ways and a Power law. Next, the fitting of the PTT model will be discussed.

19.2.1. As a PTT fluid

The relevant equations for η and N_1 can be found in section 3.7.4. It is impossible to determine η_s from a fit, because data cannot be gathered for such high shear-rates. This thesis opts for setting it to the viscosity of water. The fits for η and N_1 can be seen in figures 49 and 50

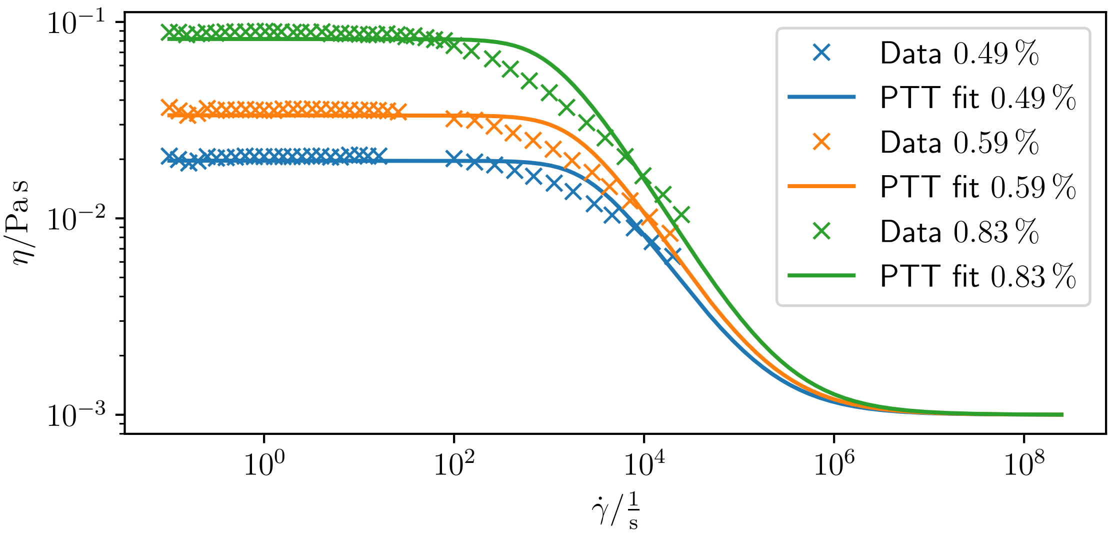


Figure 49: Plot of the viscosity for three methyl cellulose solutions from reference [107]. Lines show a fit of the Phan-Thien and Tanner (PTT) rheological model. Reproduced from [1].

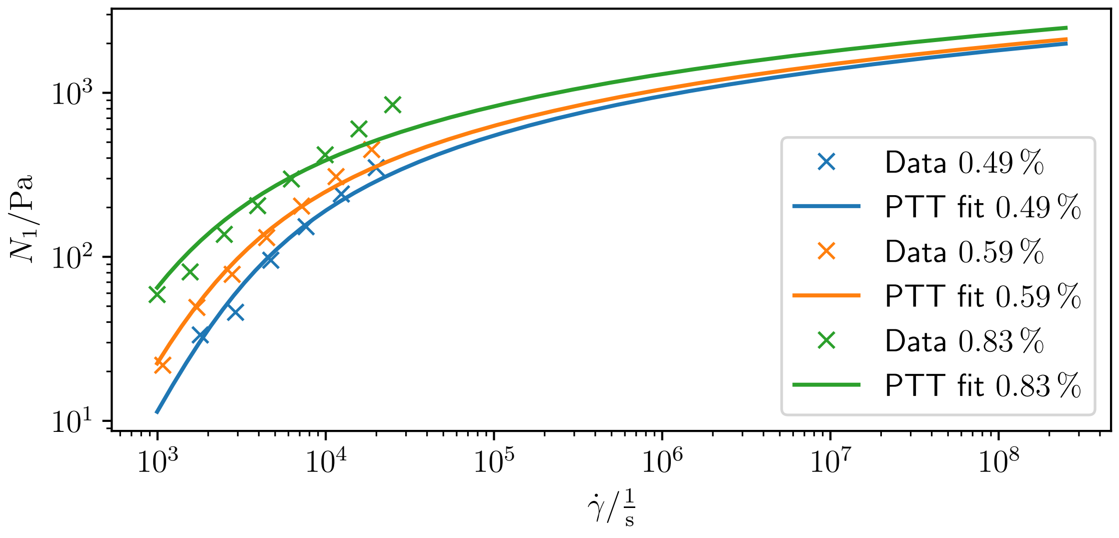


Figure 50: Plot of the first normal stress difference for three methyl cellulose solutions from reference [107]. Lines show a fit of the Phan-Thien and Tanner (PTT) rheological model. Reproduced from [1].

It is important to note, that both fits have to be carried out simultaneously. Fitting the model to the viscosity and the first normal stress difference separately yields two different parameter-sets. One can quite clearly see, that the PTT model does not fully describe the viscosity. This is due to the fact, that the PTT model as a fixed power in the power-law section, that cannot be altered using a parameter. The data falls with a different power and therefore the gap forms. This is not great, but the best agreement, that is available. For example FENE-P fits very nicely to the viscosity, and the first normal stress difference, but not at all to both at the same time. Consequently, this thesis accepts the discrepancy between the data and the PTT model and takes steps to manage it. The fit results are covered by fluids 2 – 4.

2 PTT::mc0_49

viscosity	$\eta_\mathrm{p} = (18.7 \pm 0.4)\,\mathrm{mPa\,s}$
	$\eta_\mathrm{s} = 1\,\mathrm{mPa\,s}$
relaxation time	$\lambda = (0.34 \pm 0.03)\,\mathrm{ms}$
ϵ	$\epsilon = 0.27 \pm 0.03$

3 PTT::mc0_59

viscosity	$\eta_p = (32.5 \pm 0.7)\,\mathrm{mPa\,s}$
	$\eta_s = 1\,\mathrm{mPa\,s}$
relaxation time	$\lambda = (0.43 \pm 0.04)\,\mathrm{ms}$
ϵ	$\epsilon = 0.36 \pm 0.04$

4 PTT::mc0_83

viscosity	$\eta_p = (81 \pm 2)\,\mathrm{mPa\,s}$
	$\eta_s = 1\,\mathrm{mPa\,s}$
relaxation time	$\lambda = (0.71 \pm 0.09)\,\mathrm{ms}$
ϵ	$\epsilon = 0.50 \pm 0.06$

Next, the two CY fits are discussed.

19.2.2. As a CY fluid

The CY model is not viscoelastic, it is only shear-thinning. However, it is a phenomenological model designed to fit to as many different curves as possible. The relevant equation for η can be found in section 3.5.1. The fit can be seen in figure 51.

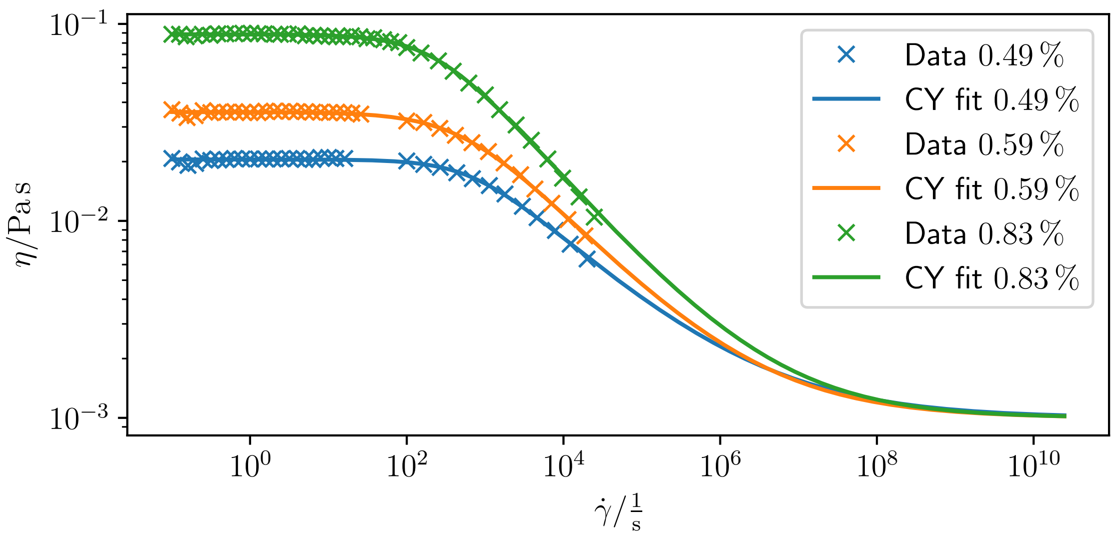


Figure 51: Plot of the viscosity for three methyl cellulose solutions from reference [107]. Lines show a fit of the Carreau-Yasuda (CY) rheological model.

The model by design fits very well. The fits result in fluids 5 – 7

5 CY::mc0_49_D

viscosity	$\eta_p = (19.52 \pm 0.08)\,\mathrm{mPa\,s}$
	$\eta_s = 1\,\mathrm{mPa\,s}$
relaxation time	$\lambda = (1.4 \pm 0.5)\,\mathrm{ms}$
a	$a = 1.1 \pm 0.2$
power	$p = 0.37 \pm 0.05$

6 CY::mc0_59_D

viscosity	$\eta_p = (34.7 \pm 0.1)\,\mathrm{mPa\,s}$
	$\eta_s = 1\,\mathrm{mPa\,s}$
relaxation time	$\lambda = (1.8 \pm 0.6)\,\mathrm{ms}$
a	$a = 0.91 \pm 0.10$
power	$p = 0.43 \pm 0.05$

7 CY::mc0_83_D

viscosity	$\eta_p = (87.6 \pm 0.2)\,\mathrm{mPa\,s}$
	$\eta_s = 1\,\mathrm{mPa\,s}$
relaxation time	$\lambda = (4.2 \pm 0.5)\,\mathrm{ms}$
a	$a = 1.07 \pm 0.05$
power	$p = 0.46 \pm 0.02$

This straight-forward way is however not the only way to fit a CY curve to the data. The curve above is well suited to describe the data. However, if one wants to differentiate the elastic from the viscous effects a curve expressing the same viscosity as the PTT model is needed. To achieve this, the CY model is fitted to the PTT curves of fluids 2 – 4. This can be seen in figure 52.

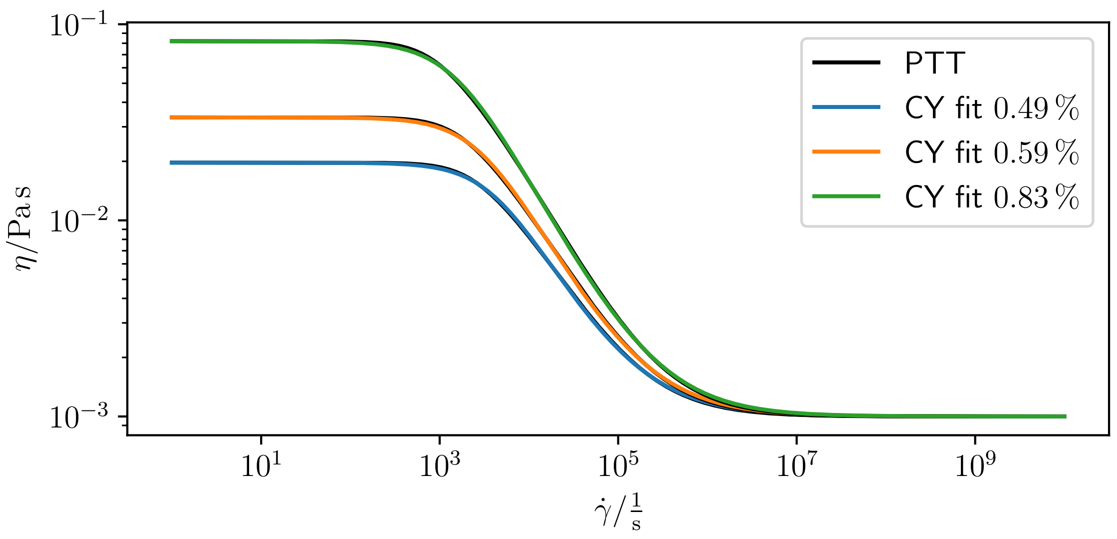


Figure 52: Plot of the viscosity for three PTT models of methyl cellulose solutions from reference [107]. Colored lines show a fit of the Carreau-Yasuda (CY) rheological model.

Clearly, the CY model is also able to describe the viscosity curve of the PTT model with decent precision. This yields fluids 8 – 10.

8 CY::mc0_49

viscosity	$\eta_p = (18.7 \pm 0.4)\,\mathrm{mPa\,s}$
	$\eta_s = 1\,\mathrm{mPa\,s}$
relaxation time	$\lambda = (0.261 \pm 0.001)\,\mathrm{ms}$
a	$a = 1.469 \pm 0.006$
power	$p = 0.837 \pm 0.001$

9 CY::mc0_59

viscosity	$\eta_p = (32.5 \pm 0.7)\,\mathrm{mPa\,s}$
	$\eta_s = 1\,\mathrm{mPa\,s}$
relaxation time	$\lambda = (0.369 \pm 0.002)\,\mathrm{ms}$
a	$a = 1.432 \pm 0.006$
power	$p = 0.847 \pm 0.001$

10 CY::mc0_83

viscosity	$\eta_\mathrm{p} = (81 \pm 2)\,\mathrm{mPa\,s}$
	$\eta_\mathrm{s} = 1\,\mathrm{mPa\,s}$
relaxation time	$\lambda = (0.682 \pm 0.003)\,\mathrm{ms}$
a	$a = 1.374 \pm 0.007$
power	$p = 0.8606 \pm 0.0010$

So to reiterate. The PTT curve and the CY curve, that is generated by fitting to the PTT curve can be used to determine whether the elastic forces are relevant to describe a particular problem. If they are not relevant, the CY curve fitted directly to the data can be used to best approximate the experiment. This is a common pattern in this thesis. Next, the power law fits are mentioned.

19.2.3. AS a power law fluid

The relevant equations can be found in section 3.5.2. Using a pure power law for the viscosity model does not capture the viscosity well for small shear-rates. This can be seen in figure 53.

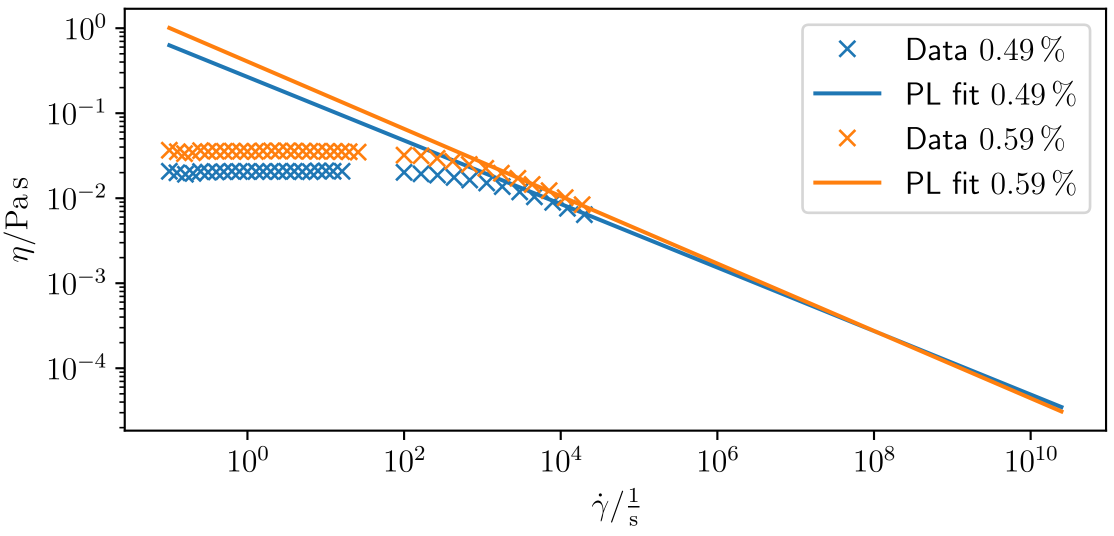


Figure 53: Plot of the viscosity for two methyl cellulose solutions from reference [107]. Lines show a fit of the Power law (PL) rheological model.

The power law fluid is only here, because it gets used by some of the sources this thesis works with [98, 99, 107]. The resulting parameters are from reference [107]. They have been converted to the standard notation used in the present work to produce fluids 11 and 12.

11 PowerLaw::mc0_49

viscosity	$\eta_\mathrm{p} = 1.492\,\mathrm{Pa\,s}$
	$\eta_\mathrm{s} = 0\,\mathrm{Pa\,s}$
relaxation time	$\lambda = 100\,\mathrm{s}$
power	$p = 0.3736$

12 PowerLaw::mc0_59

viscosity	$\eta_\mathrm{p} = 2.514\,\mathrm{Pa\,s}$
	$\eta_\mathrm{s} = 0\,\mathrm{Pa\,s}$
relaxation time	$\lambda = 100\,\mathrm{s}$
power	$p = 0.3961$

With this, the discussion of methyl cellulose solutions is complete. Next, an alginate solution is considered.

19.3. Alginate

The data for the alginate solution is agglomerated from 21 independent measurements. The preparation is as follows [1]. "Alginate (DuPont VIVAPHARM Alginate PH176) [is dissolved] in ultrapure water at a ratio of 40 $\frac{\mathrm{mg}}{\mathrm{mL}}$ [...] To ensure homogeneous dissolution, the material was stored at 37 °C for 24 h and regularly mixed to ensure complete dissolution. The measurements were conducted with a 25 mm plate-plate geometry at a 500 µm gap on a rheometer (Anton paar MCR 702). A solvent trap was used to limit evaporation. All measurements were done at a set temperature of 25 °C as shear rate sweep. [...] A cone-rheometer would be preferred here, but as $N_2 = 0$ for PTT with $\xi = 0$, the difference is negligible." [1] This solution is an important standard material for bioprinting. It is however not particularly well suited for bioprinting and a pain to handle, both in experiment and in simulations. The main reason for this is that its relaxation time is three orders of magnitude larger than the one of MC. It is used, because it was there first and has many medical applications. This means, it is mass-produced and extremely cheap. This fluid is used to fit, the PTT model, the CY model in two different ways and a Power law. Next, the fitting of the PTT model will be discussed.

19.3.1. As a PTT fluid

The relevant equations for η and N_1 can be found in section 3.7.4. The fits for η and N_1, which as before need to be carried out simultaneously, can be seen in figures 54 and 55

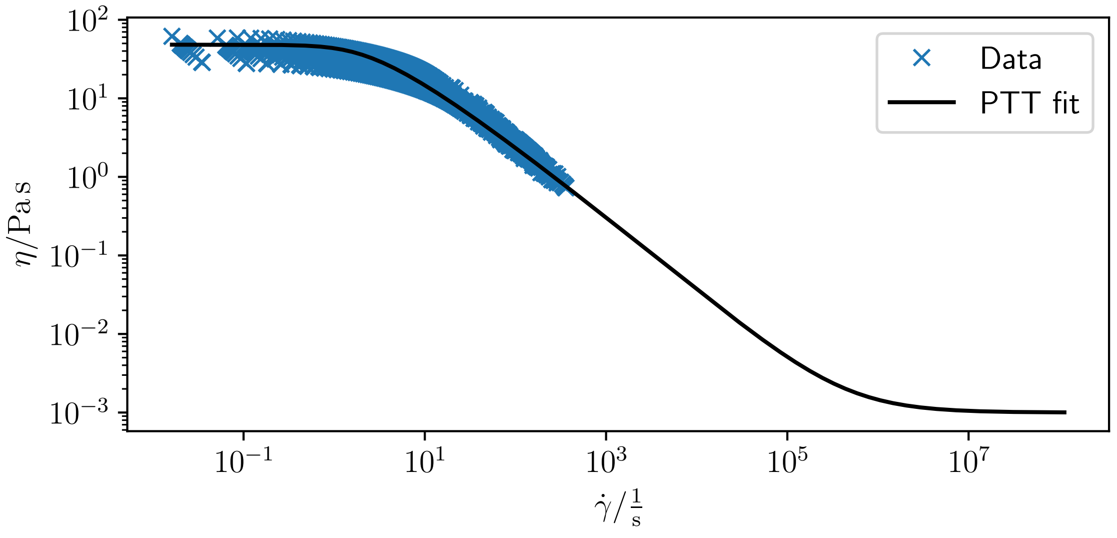


Figure 54: Plot of the viscosity for an alginate solution. The line shows a fit of the Phan-Thien and Tanner (PTT) rheological model. Similar to [1].

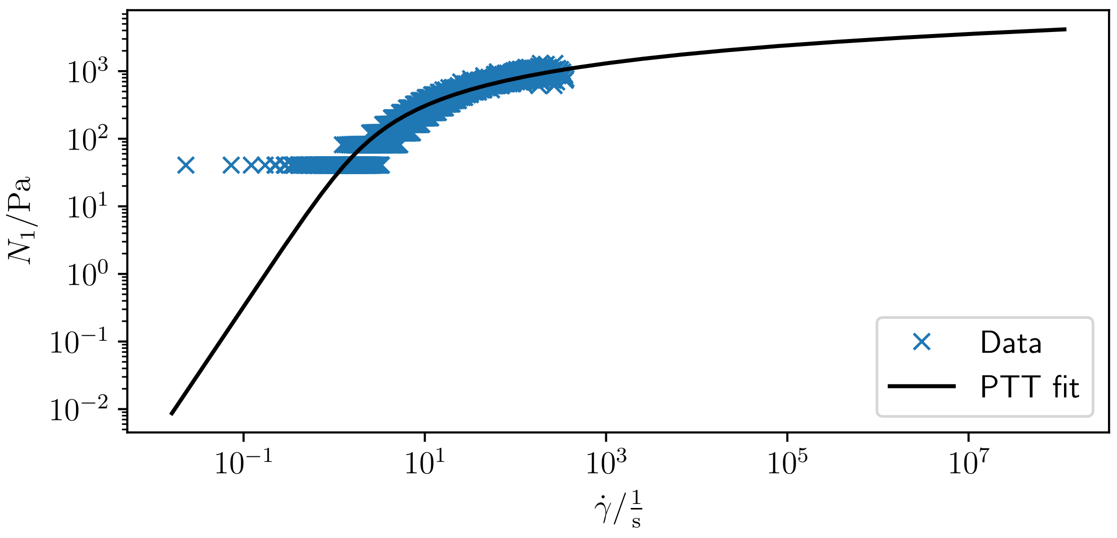


Figure 55: Plot of the first normal stress difference for an alginate solution. The line shows a fit of the Phan-Thien and Tanner (PTT) rheological model. Similar to [1].

The measurements for the first normal stress difference show a step artefact. This is caused by limited resolution of the rheometer. The fit surely is not perfect, but given how large the data cloud is, this does not matter. This fit results in fluid 13

13 PTT::alginate

viscosity	$\eta_p = (48.2 \pm 0.4)\,\mathrm{Pa\,s}$
	$\eta_s = 1\,\mathrm{mPa\,s}$
relaxation time	$\lambda = (343 \pm 4)\,\mathrm{ms}$
ϵ	$\epsilon = 0.545 \pm 0.010$

Next, the CY model is considered in two different ways.

19.3.2. AS a CY fluid

The CY model is not viscoelastic, it is only shear-thinning. However, it is a phenomenological model designed to fit to as many different curves as possible. The relevant equation for η can be found in section 3.5.1. The fit can be seen in figure 56.

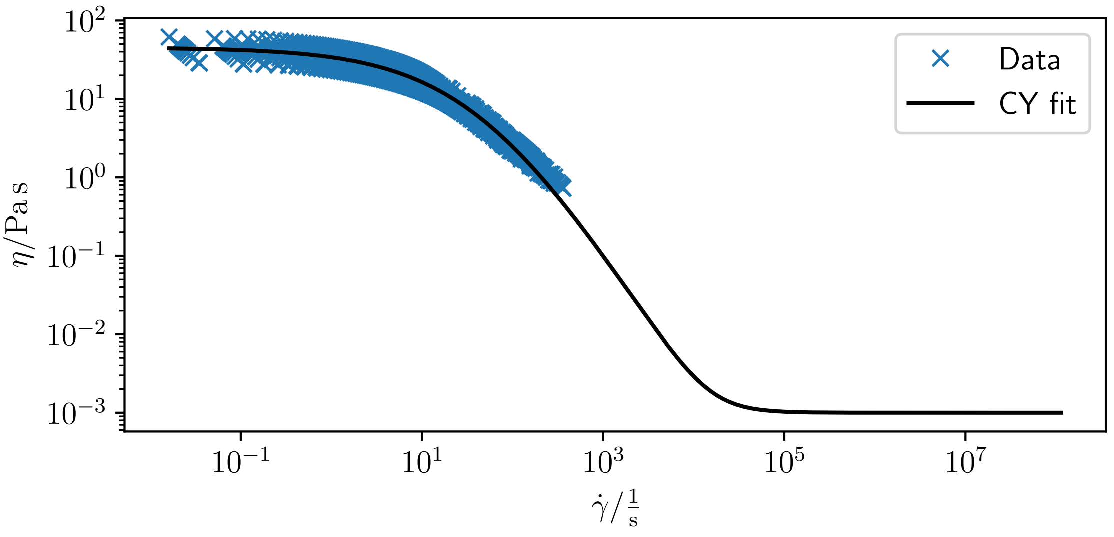


Figure 56: Plot of the viscosity for an alginate solution. The lines shows a fit of the Carreau-Yasuda (CY) rheological model.

The model by design fits very well. The fit results in fluid 14.

14 CY::alginate_D

viscosity	$\eta_p = (45.2 \pm 0.5)\,\mathrm{Pa\,s}$
	$\eta_s = 1\,\mathrm{mPa\,s}$
relaxation time	$\lambda = (21 \pm 9)\,\mathrm{ms}$
a	$a = 0.61 \pm 0.03$
power	$p = 1.9 \pm 0.3$

As before, for the methyl cellulose solutions, a secondary fit is performed. For this, the CY model is fitted to the PTT curve of fluid 13. This can be seen in figure 57.

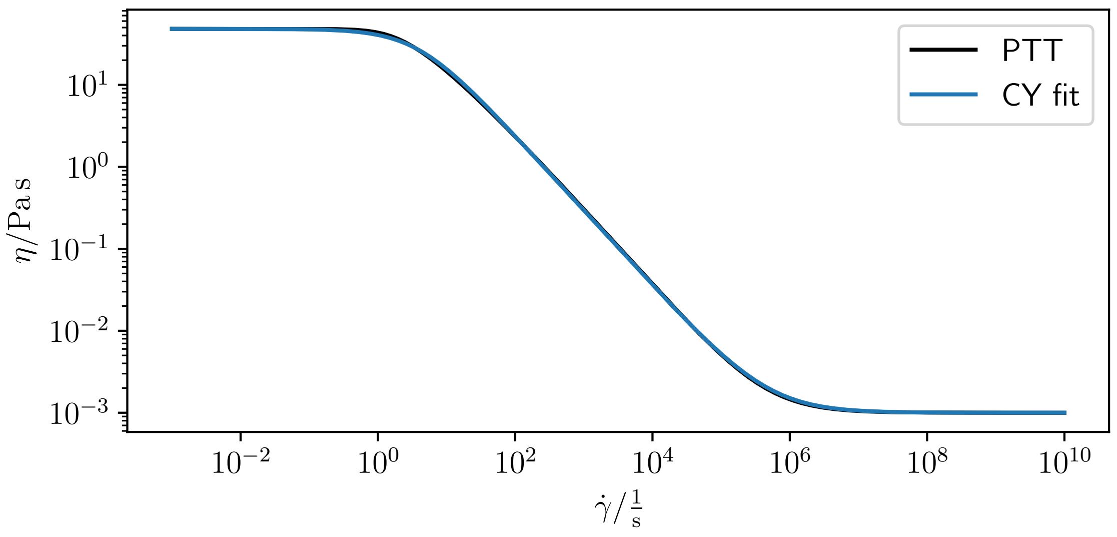


Figure 57: Plot of the viscosity for a PTT model of an alginate solution. The colored line shows a fit of the Carreau-Yasuda (CY) rheological model.

Clearly, the CY model is also able to describe the viscosity curve of the PTT model with decent precision. This yields fluid 15.

15 CY::alginate

viscosity	$\eta_\mathrm{p} = (48.2 \pm 0.4)\,\mathrm{Pa\,s}$
	$\eta_\mathrm{s} = 1\,\mathrm{mPa\,s}$
relaxation time	$\lambda = (260 \pm 1)\,\mathrm{ms}$
a	$a = 1.101 \pm 0.008$
power	$p = 0.9185 \pm 0.0007$

So to reiterate, the two CY models can be used to reproduce experiments as closely as possible and to distinguish elastic from viscous effects. If not necessary, this thesis tries to avoid the alginate solution. Next, a few requirements for an ideal fluid for biofabrication are discussed.

19.4. The ideal fluid for biofabrication

The ideal fluid for biofabrication exhibits very high viscosity directly after exiting the printer Nozzle. In the context of the fluids discussed here, this means, that the relaxation time goes to zero. However, the relaxation time determines the shear-rate at which shear thinning starts. The ideal bio-ink has low viscosity in the printer nozzle. This is most easily achieved though large relaxation times. Evidently, there is an optimum. In this regard, the methyl cellulose solutions are well suited. However, their viscosity is too small to keep a construct stable. In section 3, the relationship between the stiffness of a polymer and its relaxation time was has been hypothesized. This explains some observations made with these fluids. Assuming this is true, one would be able to predict the relaxation time from the chemical formula of a polymer. The relaxation time is probably the most important property of a bio-ink. It gets rarely discussed. However, this would be an interesting pathway to designing useful bio-inks. This concludes the considerations in regard to real fluids. Next, the basics of cell characterization shall be explored.

20. Indentation experiments

Indentation experiments using atomic force microscopy (AFM) provide information about a material's elastic properties (e.g. its Young's modulus). In the case of a cell this information can be interpreted in a wide variety of ways. Factors, from the cell's state in its life cycle, to disease, impact its stiffness for example. Accurately determining a cell's mechanical properties can therefore be valuable in the assessment of its biological state. However, once the indentation leaves the limit of negligible deformations, finding a way to accurately model the force response of a cell becomes difficult [2]. This is however required for example to interpret more complex experiments like RT-DC [108] (see section 24). This section improves on current simulation methods and provides an overview over notable limitations. Reasonable guidelines in the evaluation of AFM experiments are also provided. For the underlying theory and simulation basics consult sections 7 and 8.

20.1. Refined simulations

In this subsection possible improvements to previously published works [70] are explored. Only the Neo-Hookean (see section 5.3.4) model is discussed here due to its use in the previous publication. These improvements do however also apply to the other models listed in section 5 and discussed later in section 20.4. The Mooney-Rivlin (see section 5.3.5) model, as a generalization of the Neo-Hookean model is not discussed here to avoid the runtime cost associated with evaluating an additional parameter. For simplicity this subsection only deals with planar indenters. Other indenters and their influence on measurements are discussed in the following subsection (see section 20.2). As will be seen in the following, the existing method significantly overestimates the force particularly in low indentation scenarios. There is no analytical solution for arbitrary indentations of spherical samples. To give an estimated for expected simulation error and to distinguish this from model error, an exact solution is required. Therefore, first a cubic sample is considered.

20.1.1. Cubic Cell

A simple cube (see figure 58) can be built from 5 tetrahedrons.

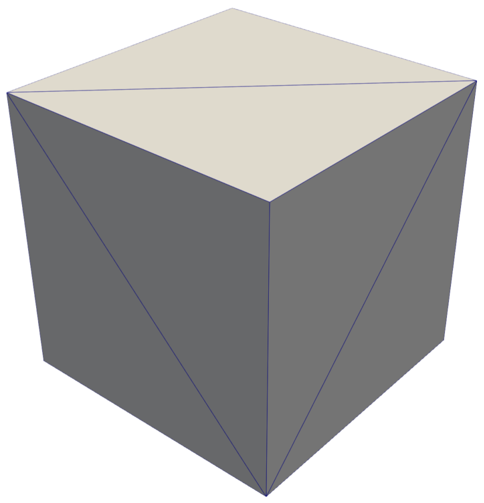


Figure 58: Illustration of the small cube mesh made of 5 tetrahedrons.

The equations describing such a cube can be simply written as follows.

$$E = \frac{\sigma}{\epsilon} \tag{348}$$

$$\sigma = \frac{F}{A} \tag{349}$$

$$\epsilon = \frac{\Delta h}{h_0} \tag{350}$$

With the Young's modulus E, the stress σ, the strain ϵ, the force F, the contact area A, the original height of the cube h_0 and the change in the height Δh. From these a simple equation relating the change of the geometry to the measured force can be found to be as follows.

$$F = EA\frac{\Delta h}{h_0} \tag{351}$$

Notably, this is not an approximation, but an exact solution. Comparing the simulation result to this therefore provides an estimate for the error experienced due to computational reasons. Ideally, this would be done at a Poisson ratio of $\nu = 0.5$, which is not computationally stable and therefore not possible. Typically, this thesis assumes $\nu = 0.48$, however this section sets $\nu = 0.49$ for increased accuracy at the cost of risking instability. An increased accuracy could not be observed for even higher ν. An AFM-Simulation (simulation 8) is performed with an (arbitrary) Young's modulus of $E = 1\,\mathrm{kPa}$ and an initial height of $h_0 = 25\,\mu\mathrm{m}$. This h_0 represents a typical value for cells but is of no special meaning here. All AFM simulations in this thesis are performed with $E = 1\,\mathrm{kPa}$ as the results can be scaled (see sections 5.5 and 6.5). The nominal indentation was set at 2.5 %.

8 Compression of a small cube

Cell:	Young's modulus $E = 1\,\mathrm{kPa}$
	Size, Resolution; $25\,\mu\mathrm{m}, 1$

The simulation measures a force, which together with the observed deformation can be inserted into equation 351 to calculate the Young's modulus. The resulting $E \approx 996\,\mathrm{Pa}$ lies approximately $4.4\,‰$ below the expected value of $1\,\mathrm{kPa}$. Repeating the simulation (simulation 9) using a much finer mesh (approximately 2×10^5 tetrahedrons) generated using gmsh [85], yields a slightly decreased error of $2.6\,‰$.

9 Compression of a large cube

Cell:	Young's modulus $E = 1\,\mathrm{kPa}$
	Size, Resolution; $25\,\mu\mathrm{m}, 36$

This is a first hint of a resolution dependence, which will be relevant for the more complicated spherical geometry (see section 20.1.3). From this, it is clear, that errors in the permille-region should be expected from the simulation. Comparing this to Müller *et al.* [70], which displays considerably larger errors for small indentations, it is clear an improvement is possible. First, one should consider relaxation.

20.1.2. Relaxation

Consider an AFM experiment with a spherical particle and a flat indenter (simulation 10 for a resolution of 100). The AFM simulations are performed by approaching the indenter-potential until the desired nominal deformation is reached. Observing the force F from this point onward (see figure 59) reveals a relaxation. This is expected due to the dampening factor the simulation introduces (see section 8). The lines have different length, due to the higher indentations being reached later, while this point is set as $t = 0$ for all. Subtracting the last measured force in each line from the set and only plotting the deviation $F_{\text{deviation}}$ allows showing the exponential nature of this relaxation (see figure 60).

10 Relaxation of a deformed cell

Cell:	Young's modulus $E = 1\,\mathrm{kPa}$
	Radius, Resolution $R = 12.5\,\mu\mathrm{m}$, variable

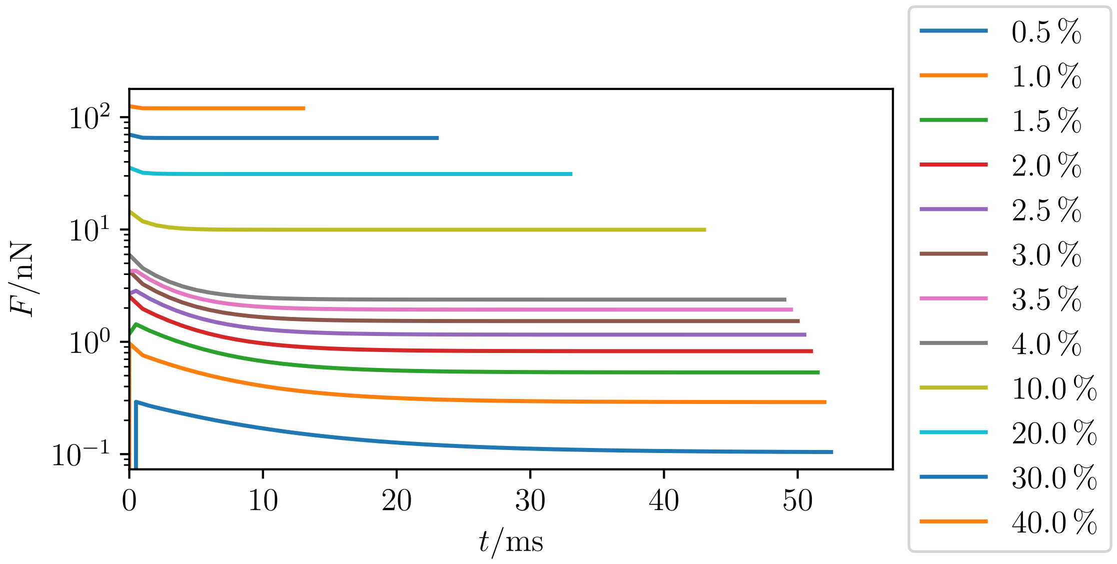


Figure 59: The force after the nominal indentation is reached as a function of time for different indentations. The observed relaxation is dependent on the indentation.

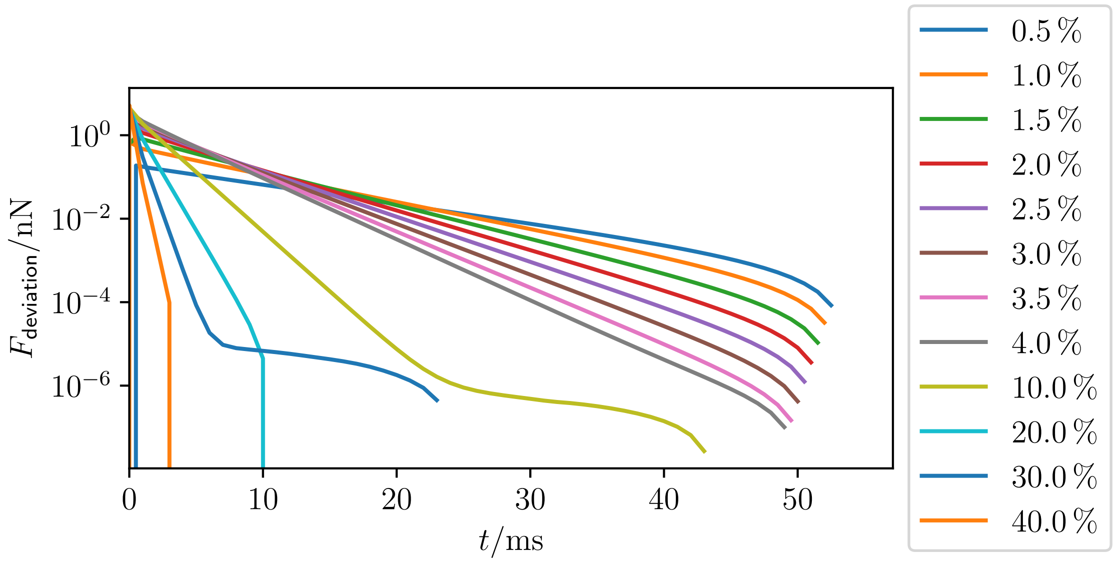


Figure 60: The deviation from the final force after the nominal indentation is reached as a function of time for different indentations. The observed relaxation is dependent on the indentation.

The higher indentations lack the temporal resolution to properly visualize the relaxation and therefore show some artefacts. Nevertheless, the exponential nature of this relaxation is made obvious by the linear portions of each line. Notably, the simulations with higher indentations relax significantly faster. This is unexpected and could lead to neglect of the relaxation if one only checks the relaxation for higher indentations. If one neglects to take into account the relaxation as was done by Müller *et al.* [70] for example, one can increase the deformation continuously while measuring the force. If the relaxation is considered, the simulation requires relaxation for each data-point. This additional computational cost and the result being discrete points rather than a continuous curve motivates the neglect of relaxation. However, the L1 error (see appendix S), defined here as follows, of such a neglect is large especially for small deformations, as can be seen as a function of the nominal deformation δ in figure 61.

$$e_{\text{relaxation}} = \frac{F}{F_{-1}} - 1 \tag{352}$$

Where F_{-1} is the relaxed force (final force in each series). As can be seen in the logarithmic plot, the error follows a roughly exponential trend from around 200 % for small deformations to about 4 % for larger ones. It should be noted, that the error for small deformations displayed here is slightly under-estimated due to the limited run-time of the simulations.

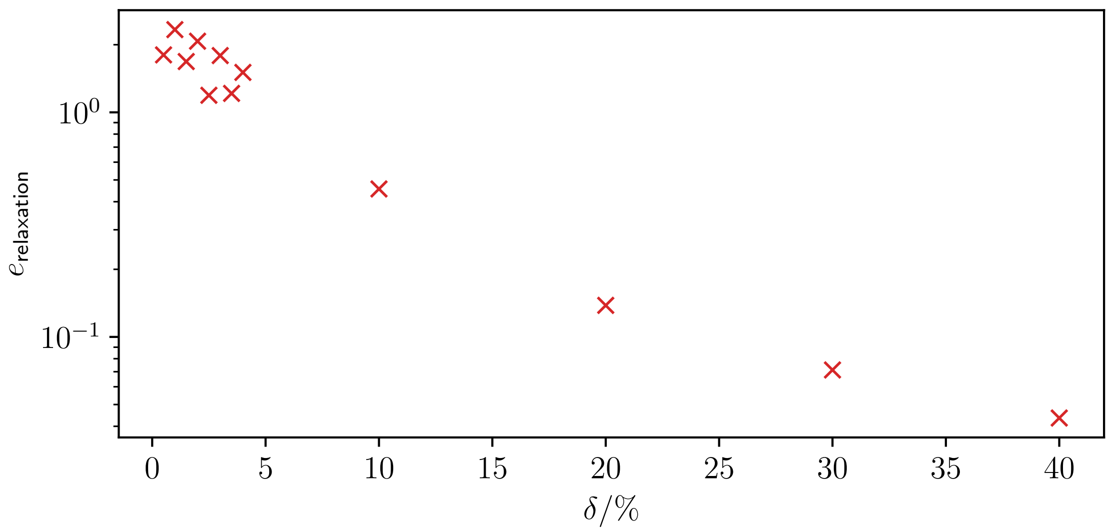


Figure 61: The error caused by neglecting the relaxation as a function of the nominal deformation.

Considering, that it is customary to primarily consider deformations below 10 % for the evaluation of AFM experiments, this error needs to be avoided. Aside from the error due to relaxation, the resolution of the simulated cell also contributes to the error and will be discussed next.

20.1.3. Resolution dependence

Contrary to the cube example from above, the local deformation in the round cells is not constant and equal to the global deformation but a function of the location.

Local deformation refers to the local value of a continuous deformation field defined over the whole cell. Global deformation refers to the deformation measure addressing the entire cell as a whole. The implementation (see section 8) treats the local deformation as constant across one tetrahedron. The validity of this discretization hinges on the resolution of the cell. Therefore, an error is introduced, which depends on the resolution of the cell. In this thesis, the resolution of cells is given by their radius R in average tetrahedron side-lengths[28]. Here resolutions from $R = 5$ to $R = 100$ are considered (simulation 10). Most of the low resolution simulations proved unstable and were discarded. It should be noted, that computational time approximately increases with R^3 and therefore it is important to limit the resolution. The relaxed force appears to be a power-law function of the resolution. The following power-law is fitted to the simulation data for each indentation.

$$F = aR^b + c \tag{353}$$

With the generic parameters a, $b < 0$, c. The parameter c is the force for infinite resolution and is such interpreted as the true force if the resolution error were absent. Dividing the data-sets by their respective c yields the resolution error as follows.

$$e_{\text{resolution}} = \frac{F}{c} - 1 \tag{354}$$

This error can be seen in figure 62.

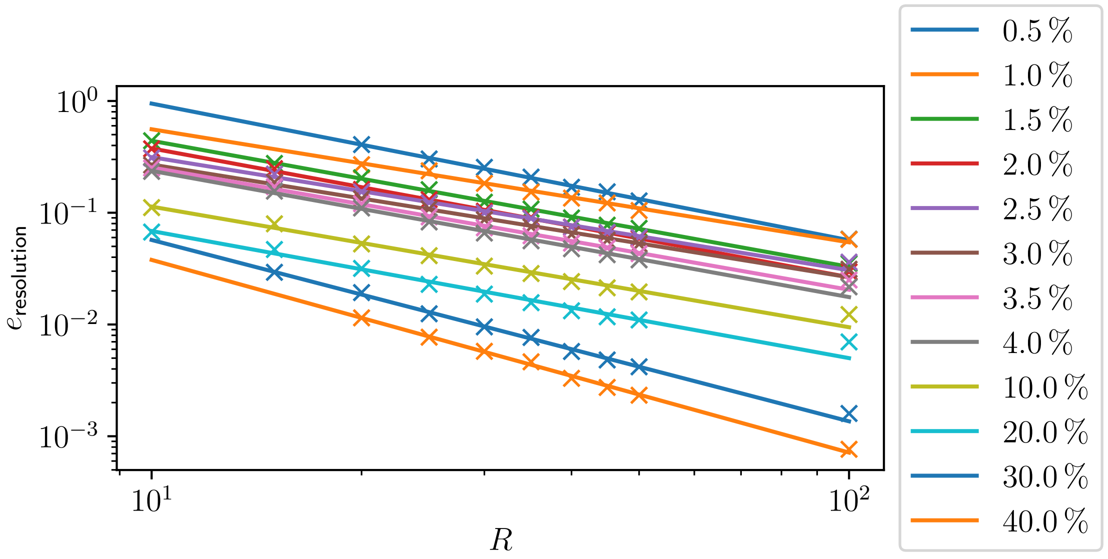


Figure 62: The error as a function of resolution for different indentations. The crosses are simulation data-points, the lines are power-law fits.

The power-law nature can be seen in the double logarithmic plot. The higher indentations showed deviations before, which might be caused by limited time resolution here as well.

[28]The average tetrahedron side-length refers to the value given to gmsh [85]. The actual value in the mesh produced by gmsh will differ slightly.

Otherwise, aside from small deviations, all these lines are parallel. This means, that the error decreases equally quickly with resolution for all indentations. The only difference in the double logarithmic plot is an offset (a scaling factor), resulting in both, the error for small resolutions being larger, and a higher resolution being required to reach an acceptable error. Similarly, counter-intuitive as before, the smaller indentations show the larger errors. This likely caused Müller *et al.* [70] to neglect it. A resolution of $R = 100$ still shows an error of approximately 6 % for smaller indentations and is already very costly in terms of computational time. Therefore, archiving an exact solution by ways of higher resolution is untenable. It is advisable to perform simulations with a few medium resolutions (the small ones tend to be unstable), perform a fit and extrapolate from there. The elimination of these errors can significantly decrease the overall error as can be seen in the following section.

20.1.4. Comparison

For small deformation, the Neo-Hookean model converges with the Hertz model [91]. Therefore, the improvement to the simulation can be seen in the discrepancy between the Hertz model (see section 7, eq. (217)) and the simulation. This discrepancy u_{Hertz} is defined as a L1 error (see appendix S) in the following.

$$u_{\text{Hertz}} = \frac{F_{\text{simulation}}}{F_{\text{Hertz}}} - 1 \tag{355}$$

With the force measured by the simulation $F_{\text{simulation}}$ and the one predicted by the Hertz model F_{Hertz}. This discrepancy can be seen in figure 63 for small indentations δ.

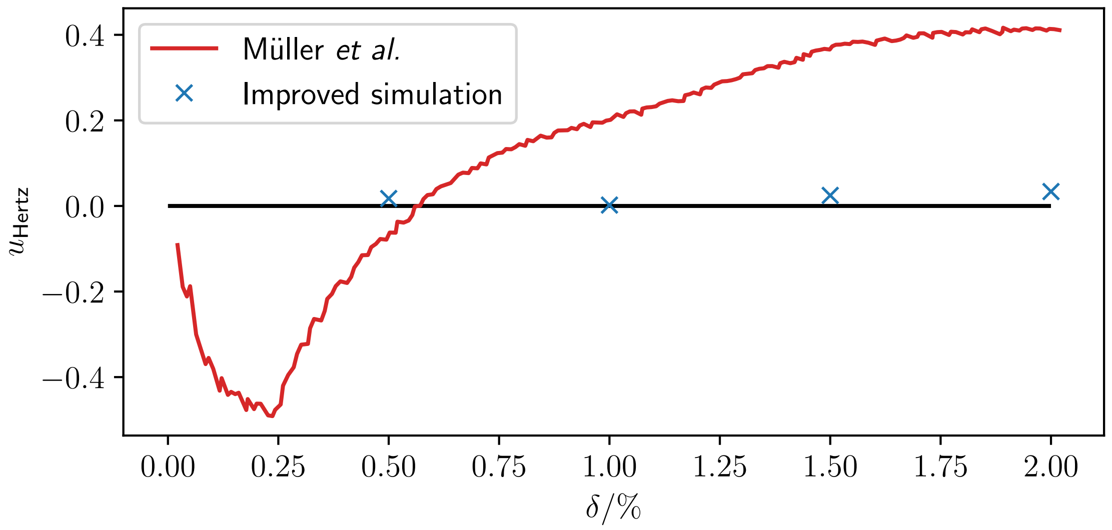


Figure 63: The discrepancy from the Hertz theory as a function of the indentation. Focus on small indentations.

While Müller *et al.* [70] reach 43 % discrepancy, the values determined here lie between 2 ‰ and 3 %. As can be seen in figure 64, relatively speaking, these values approach each other for larger indentations. This is expected as the neglected errors decrease for larger indentations as was shown above.

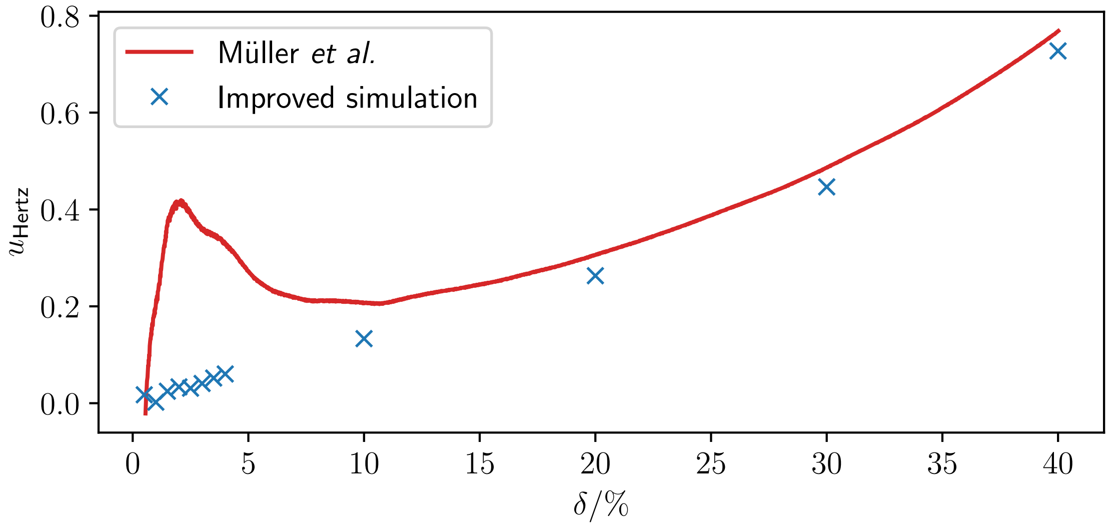


Figure 64: The discrepancy from the Hertz theory as a function of the indentation.

It should be noted, that the Hertz theory only applies in the limited of small deformations. Therefore, a discrepancy is expected and is not necessarily an error. Given the discrepancy hovers around 2 % for the smallest deformations, there likely is a residual error of that magnitude though. A discussion on the bounds of the small deformation limit for hertz theory can be found in section 20.2.

20.1.5. Conclusion

This section demonstrates and realizes significant improvement positional in AFM simulations. These are primarily relevant for small indentations. As small indentations are typically used to evaluate AFM measurements using the Hertz theory, these improvements are critical to ensure comparability in this regime. Forces for larger indentations are improved markedly, but contrary to the values for small indentations not by enough to invalidate conclusions drawn from previous simulations.

20.2. Hertzian range

The Hertz approximation [90] (see section 7) is well known to only be valid for small approximations. This necessitates the definition of a cutoff for what qualifies as small. Due to ambiguity of the material response, experiments cannot do this reliably. In simulations, it is however possible to guarantee pure Neo-Hookean behavior. Therefore, given discretization errors and the like are accounted for, deviations can be attributed to leaving the valid regime of Hertz theory. This allows to clearly establish bounds for this theory as defined by an acceptable error margin. This subsection elaborates on the safeguards employed to assure the separation of the simulation error from the theory error. Furthermore, effective bounds for the usefulness of Hertz theory are found.

20.2.1. Declared limits

Hertz theory carries a few assumptions described in the publication of which the following warrant discussion [90].

1. The contact surface can be described by a parabolic equation.
2. The indentation is small in relation to the particle.
3. The indentation is small in relation to the indenter.
4. The material is linear elastic.

So far only a planar indenter was considered. In general, this thesis discusses a hard spherical indenter compressing a particle. In this case, the contact area is the undeformed surface of the indenter. Small sections of a sphere can be Taylor-approximated as parabolic. So the first assumption is true, given the third holds. A flat indenter (equivalent to a sphere of infinite radius) guarantees the third assumption and allows evaluation of the second one. This will be discussed first. Afterwards, the third assumption shall be discussed. This often gets ignored in literature in order to use small indenters to determine local stiffness [109]. The Neo-Hookean model is only approximately linear elastic for small deformations. This will be discussed in a separate subsection (see section 20.3).

20.2.2. Flat indenter

A flat indenter guarantees both a parabolic contact surface and an indentation small compared to the indenter. This geometry was already used in the previous subsection and the following error discussion is based on these results. The discrepancy from Hertz theory u_{Hertz} shall once more be defined as follows.

$$u_{\mathrm{Hertz}} = \frac{F_{\mathrm{simulation}}}{F_{\mathrm{Hertz}}} - 1 \tag{356}$$

As can be seen in figure 65, it rises with increasing nominal deformation δ.

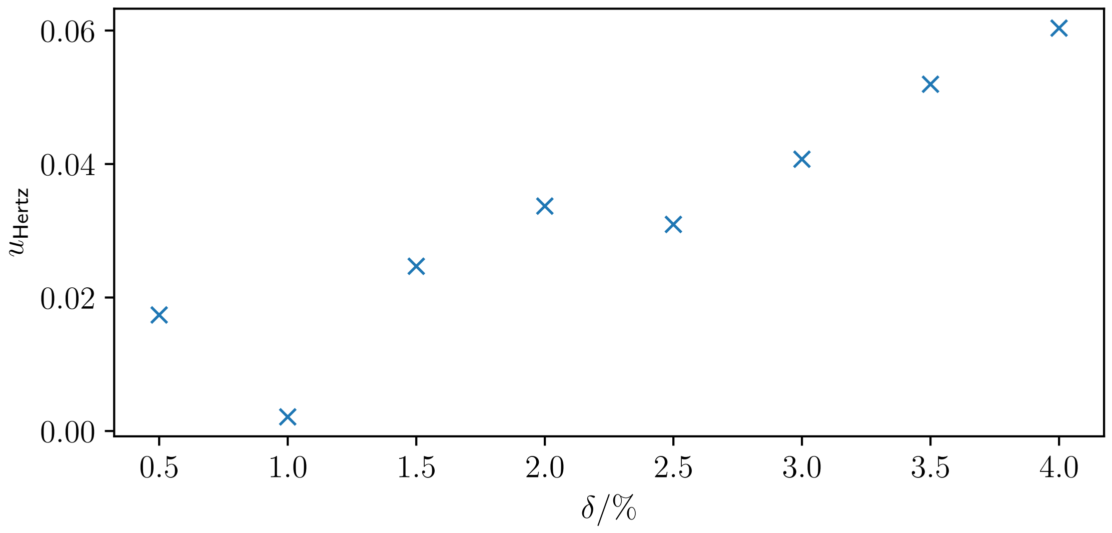


Figure 65: The discrepancy from the Hertz theory as a function of the indentation. Focus on small indentations. Excerpt from figure 64.

The error seems to hover around 2 % for deformations below 2 % and rises roughly linearly above. For larger deformations ($\approx 20\,\%$), the error becomes super linear (see figure 64). It is reasonable to assume, that the remaining error for deformations approaching 0 is a simulation artefact and not physical. Depending on ones error tolerance one should not use Hertz theory for deformations larger than 3.5 %, which yields a roughly 5 % error for Neo-Hookean materials. Linear elastic materials offer a greater range of closeness to the Hertz model (see section 20.3). Given the linear behavior starts at around 2 %, with residual simulation error dominating values below that, this represents the lowest error achievable via these simulations. Therefore, $\delta = 2\,\%$ is used throughout this thesis as a cutoff for the reasonable use of Hertz theory in the case of Neo-Hookean materials (which are the default here). This should limit the error to approximately 3 %.

20.2.3. Spherical indenter

Subsection 20.1.3 already demonstrated the importance of resolution. In case of a spherical indenter, resolution attains additional importance. If the resolution is not sufficiently large, the simulation result is independent of the indenter size. This can be seen in figure 66. Here the indenter acts on only a single node independent of the indenter size for a significant range of sizes.

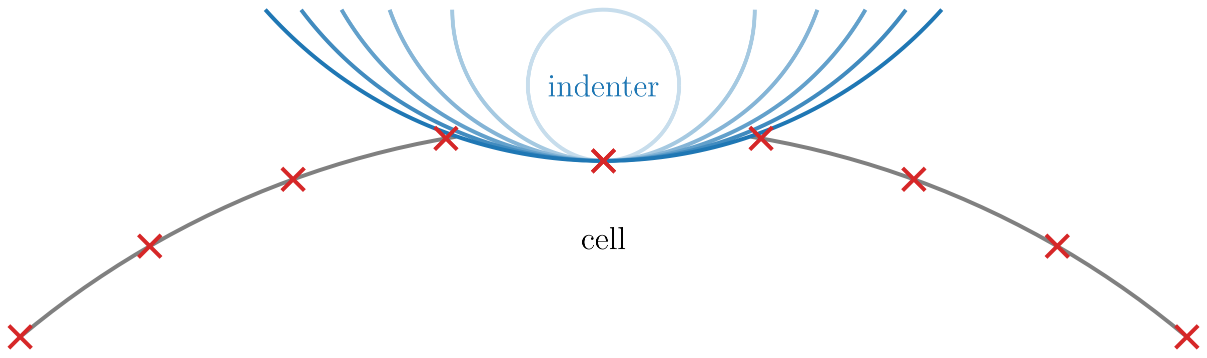


Figure 66: Illustration of the cross-sectional geometry for an AFM experiment with a spherical indenter: Indenters with sizes varying from 8 % to 50 % of the cell size are shown. They indent a $R = 6$ cell shown by its discrete surface nodes by 2 %.

This problem disappears given the resolution is high enough. It is more pronounced for smaller indentations. In the case of the flat indenter before, this effect can safely be ignored, because it only becomes relevant for tiny indentations. Furthermore, even for those cases the introduced error is relatively small. This is a potential cause for the 2 % residual error observed in the subsection before. In case of spherical indenters however, their size (and thus the expected force) can vary significantly without changing the force measured in simulations. If a resolution is insufficient for a given indenter size, the error caused by this behavior would impact the resolution fit procedure described in the previous subsection. To avoid this effect any such data is excluded as described in appendix W. The dataset explored in simulation 11 is shown in table 8. The excluded data-points are shown in figure 67 by the shaded area.

11 Deforming a cell using a spherical indenter

Cell:	Young's modulus	$E = 1\,\mathrm{kPa}$
	Radius, Resolution	$R = 12.5\,\mu\mathrm{m}$, variable

quantity	values
deformation $\delta/\%$	0.5, 1, 1.5, 2
resolution R	10, 20, 30, 40, 50, 100
indenter size $i/\%$	10, 20, 30, 40, 50, 60, 70, 80, 100, 200, 300, 400

Table 8: Parameters explored for determining the error range of spherical indenters. The values are relative to the cell diameter.

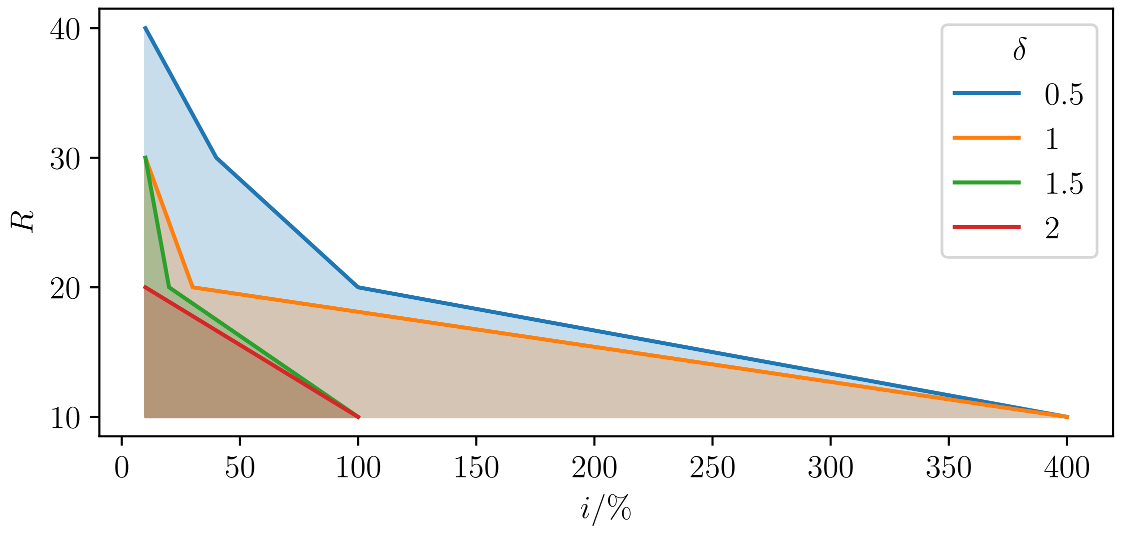


Figure 67: Plot of the data-points with insufficient resolution. The shaded area represents the excluded points.

With this, the edges of the third Hertzian assumption as listed before in section 20.2.1 can be explored. If the indentation is just barely in the range of being small compared to the cell (here $\delta = 2\,\%$ as an example), having an indenter smaller than the cell will violate the third assumption (indentation small compared to the indenter). The influence of this can be seen in figure 68, where the force F as a function of the indenter size R_{indenter} relative to the cell size R_{cell} is shown. The simulation force being higher at the end is expected, as this was also observed by for flat indenters before. It is clear, that the error incurred by using too small of an indenter lowers the force. As the remaining error for the flat indenter increases the force as discussed before, this could be used to cancel both errors.

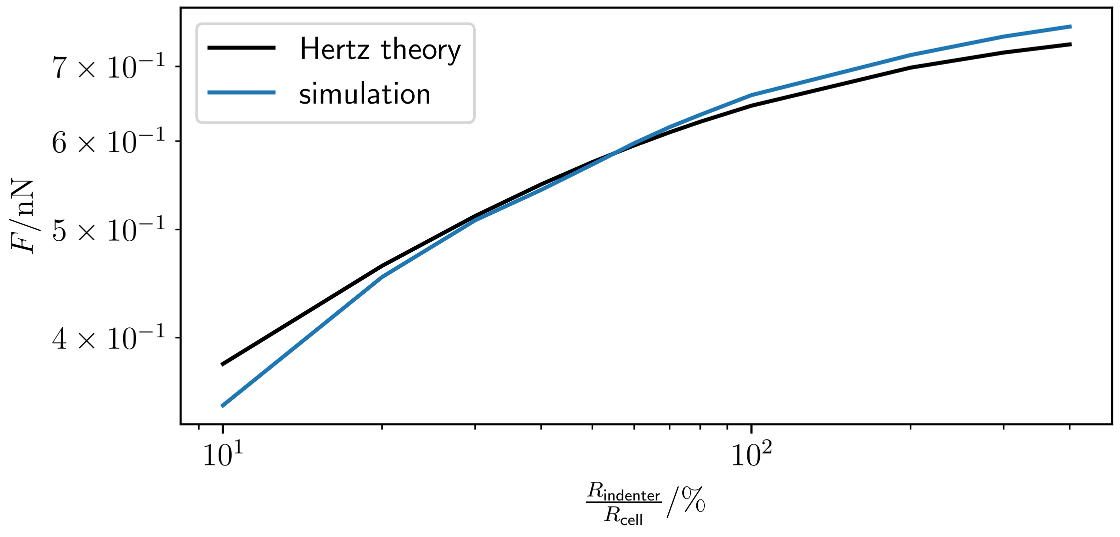


Figure 68: Plot of the simulated force vs the force predicted by Hertz theory for different indenter sizes at 2 % indentation.

The discrepancy from the Hertz model is shown in figure 69. While a significant error can be produced for very small indenters, this is generally not as severe as for increasing indentation. This is due to the indentation error occurring at both the indenter-cell and the cell-substrate contact, while the error discussed here only occurs at the top contact. Using tiny indenters for local resolution is still not advisable, but compared to going beyond low single digit indentation small indenters are of no concern.

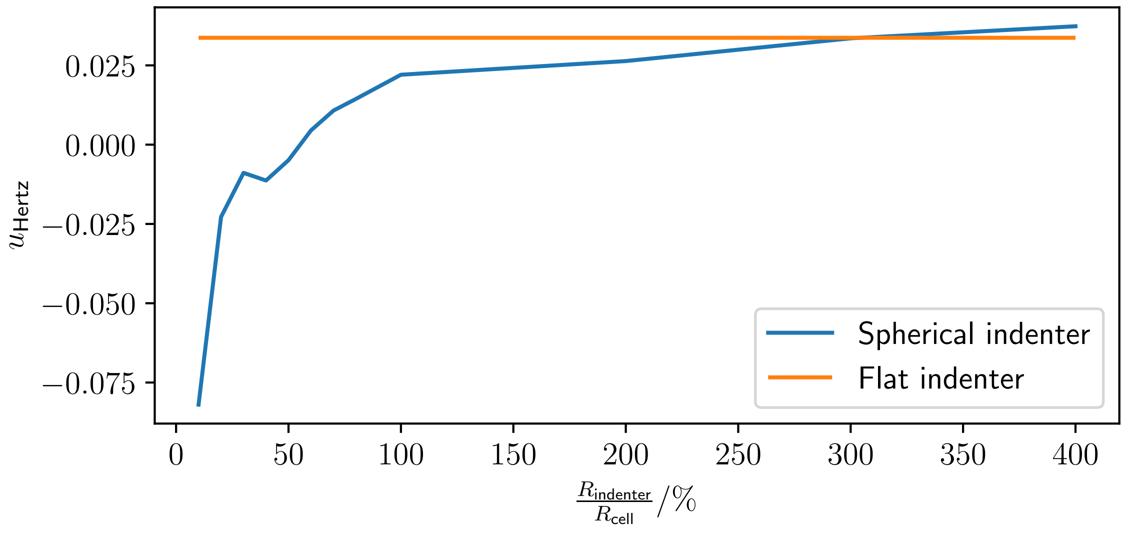


Figure 69: Plot of the discrepancy between simulated force and the force predicted by Hertz theory for different indenter sizes at 2 % indentation.

20.3. Comparison of elastic models

So far only the Neo-Hookean elasticity model has been considered. It and its generalization, the Mooney-Rivlin model will be the primary models used in this thesis, due to them describing biological cells with acceptable accuracy [70, 81] (see section 20.4.7). As discussed in the previous subsections, the Hertz theory is only applicable for the Neo-Hookean model in the case of small deformations. However, polymer particles used for studying cells in a simplified manner [2], act differently. They exhibit more linear behavior, which is closer to the Hertz model than the Neo-Hookean cells. This allows larger ranges to be used for fitting to the Hertz model, reducing the uncertainty. A comparison of a few of these models follows. Notably, the Neo-Hookean, the Saint Venant-Kirchoff (SVK) and a modified Saint Venant-Kirchoff (SVKK) model are considered (see section 5.3). For this, simulation 10 is repeated in the whole parameter space for SVK (simulation 12) and SVKK (simulation 13).

12 Relaxation of a deformed SVK cell

Cell:	Young's modulus $E = 1\,\text{kPa}$
	Radius, Resolution $R = 12.5\,\mu\text{m}$, variable

13 Relaxation of a deformed SVKK cell

Cell:	Young's modulus $E = 1\,\text{kPa}$
	Radius, Resolution $R = 12.5\,\mu\text{m}$, variable

For small indentations δ, these models all roughly follow the Hertz theory (see figure 70).

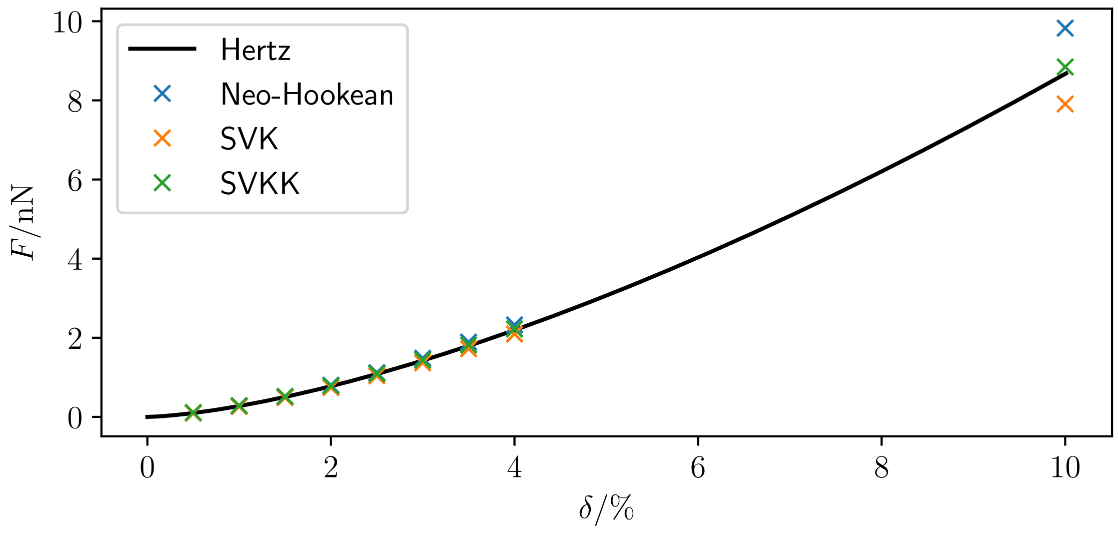


Figure 70: The force from simulations of three elastic models compared to the Hertz theory as a function of the indentation. Focus on small to intermediate indentations.

As the deformation increases a notable diversion can be seen. Plotting the discrepancy from the hertz theory for these models (see figure 71), one can see the following trends. The Neo-Hookean model is consistently above Hertz, with increasing distance for higher deformations. The SVK model is consistently below Hertz, with higher distance for higher deformations. Aside from very small deformations, the SVKK model is between the aforementioned. With this it is very close to the Hertz model even for intermediate deformations. It shall be noted, that the simulation starts breaking down around a deformation of 30 % for SVK and 20 % for SVKK. This is due to the force being produced by those models staying finite, even for vanishing tetrahedrons. This is impossible to simulate using current simulation techniques. For the example illustrated here, this starts happening around 25 nN of force. Finding exact results for this geometry is difficult, but literature has discussed this for uni-axial stress [82]. Note, that SVKK differs from the SVK(E_1) defined by Fraldi *et al.* by a factor of $\frac{1}{\sqrt{2}}$ in the strain definition.

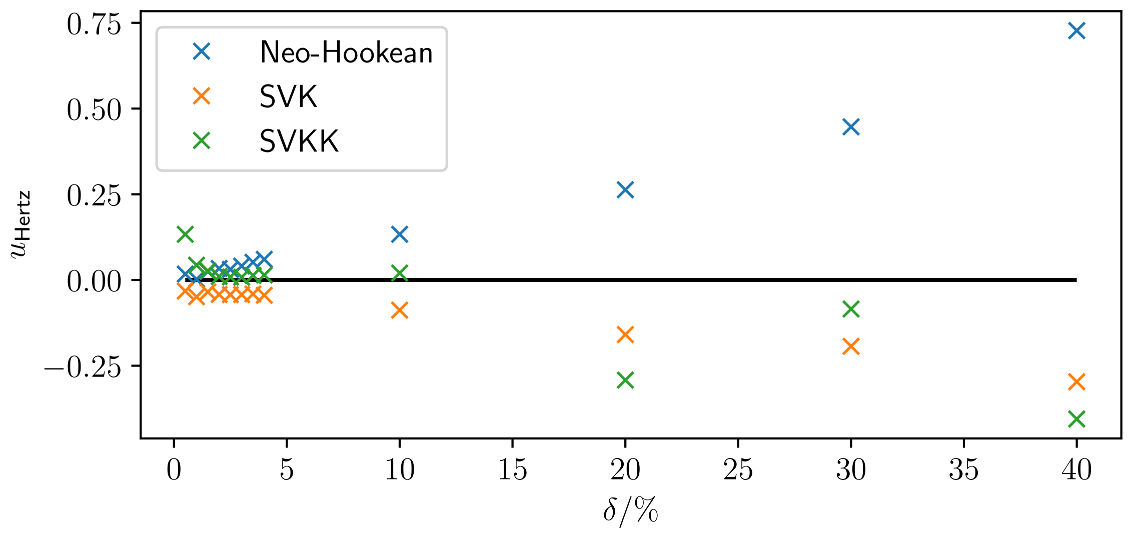


Figure 71: The discrepancy from the Hertz theory as a function of the indentation for three elastic models.

It is reasonable to assume, that if simulation techniques were developed to overcome vanishing tetrahedrons, the SVKK model would stay in between the Neo-Hookean and the SVK model. Such behavior can be seen in the uni-axial case [82]. If stably, the SVKK model would thus likely reproduce data from linear elastic particles (see next subsection) quite well. The models shown here represent various steps between linear elastic and nonlinear elasticity. The SVKK is the most linear model and thus the one closest to Hertz theory. The SVK model has a non-linear strain. The Neo-Hookean model is also non-linear in its stress-strain relationship. The Mooney-Rivlin model (not shown here) considers an additional term compared to the Neo-Hookean model and produces even higher forces. The following subsection will discuss the applicability of these models to measured data from different materials. Some of which behave more linear than others.

20.4. Evaluating experimental data

After establishing a few elastic models, their applicability to real experimental data is explored. First cell-mimetic hydrogel particles are considered. These were manufactured and measured by Steffen Trippmacher. Details on that can be found in earlier publications [2]. Afterwards real cells are considered. For this data from a previous publication by Müller *et al.* [70] is used. This subsection uses data from simulations 10, 12 and 13 and evaluates it. The polymer particles are expected to behave linear elastic to some degree. To distinguish this different indentation regimes are considered. Before that, the data needs some preparation.

20.4.1. Preparation

The data as returned by the experiment gives the force as a function of the position of the indenter. This position (termed separation here) has an arbitrary offset, which needs to be found. The measured force might have a small offset, due to drift. In order to correct both, the following power-law is fitted around the contact point.

$$F' = \begin{cases} a(x+b)^{\frac{3}{2}} + c & ,\text{for } x+b \geq 0 \\ c & ,\text{else} \end{cases} \tag{357}$$

Where F' is the measured force, x is the separation and a, b, c are parameters. It should be noted, that the exponent of 1.5 could be replaced by another parameter. This yields values of around 1.6 to 1.8 for this dataset. As discussed in previous subsections (see sections 20.2 and 20.3), for small deformations, the material models converge to the Hertz theory. Therefore, assuming the exponent of the Hertz theory is sensible. The gathered offsets are used to remove the force offset and to turn the separation into the deformation. This fit can be seen in figure 72. The red box represents a region of uncertainty due to the noise of the data, within which the plots true origin most likely lies. This region of ambiguity spans about $\frac{1}{3}$% deformation, which in case of small indentations is important as will be discussed first.

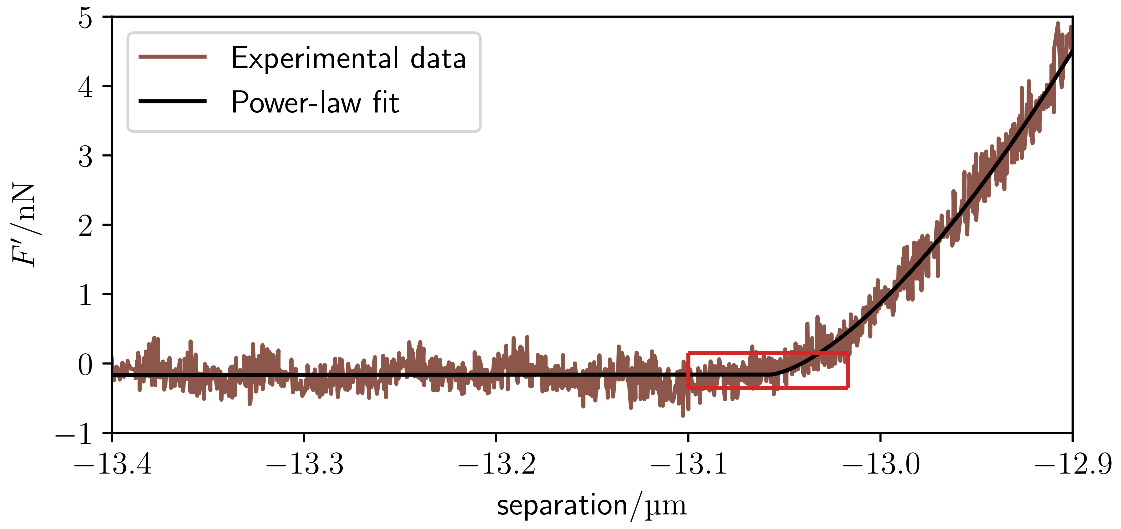


Figure 72: The raw data as measured by Steffen Trippmacher compared to a power-law fit. The red box represents a region of uncertainty for the true origin, caused by measurement noise.

20.4.2. Fitting with simulation data

In the following, the simulation data for the different material models will be used as a model used for fitting. The measured force scales with the Young's module (see sections 5.5 and 6.5). This allows to produce force-deformation curves for any Young's modulus by simulating with a fixed one (1 kPa for this analysis) and scaling it appropriately. This scaling, together with small shifts in the deformations allows fitting the simulation data to the experimental data like a theoretical model. In return one gets the appropriate Young's modulus and offset. This is the process by which the fits in the following subsections are obtained.

20.4.3. Small indentations

As discussed in previous subsections (see sections 20.2 and 20.3), the Hertz model can always be used for the bottom 2 % deformation. Therefore, this range is termed the small-indentation-regime here and will be discussed in the following. For this regime, the exact location of the origin is essential. When fitting the Hertz model, varying it within the red box seen in figure 72 yields Young's moduli ranging from 28 kPa to 39 kPa. To honor this sensitivity, a deformation offset is allowed for all the modules fitted in this section. The Hertz model and the three models discussed previously (Neo-Hookean, SVK and SVKK) are fitted to the data. All the fits find offsets that are very small compared to 1 %. As expected and as can be seen in figure 73, the Hertz model fits very well. The Young's module found from the Hertz fit is $E_{\text{Hertz, }2\,\%} = (34.46 \pm 0.05)\,\text{kPa}$.

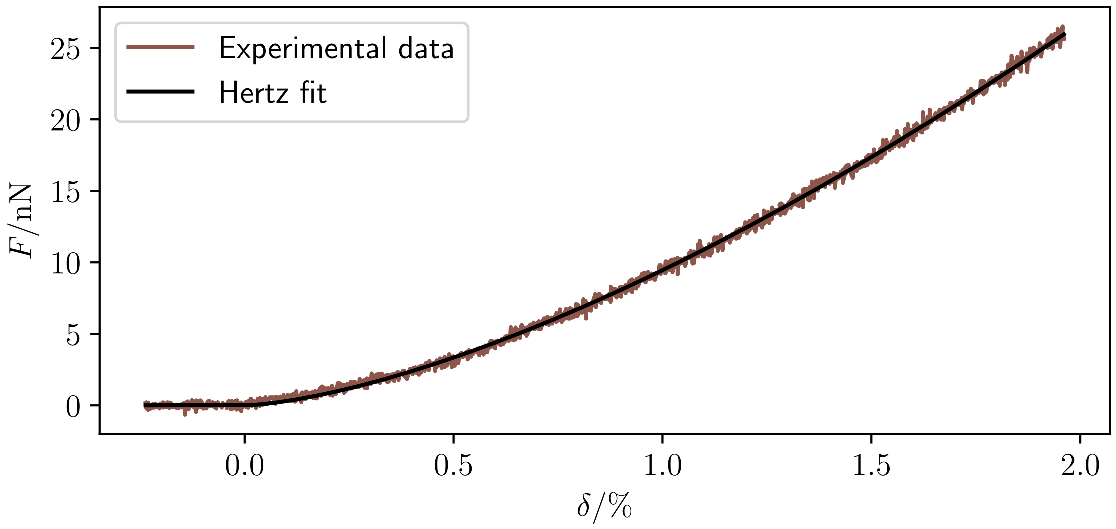


Figure 73: The corrected experimental data compared to a Hertz fit.

Among the numeric elastic models, the Neo-Hookean model fits the best with a Young's modulus of $E_{\text{Neo-Hookean, }2\,\%} = (32.2 \pm 0.3)\,\text{kPa}$. This can be seen in figure 74. As expected though, SVK and SVKK also fit very well in this regime with Young's moduli of $E_{\text{SVK, }2\,\%} = (34.5 \pm 0.2)\,\text{kPa}$ and $E_{\text{SVKK, }2\,\%} = (32.5 \pm 0.4)\,\text{kPa}$

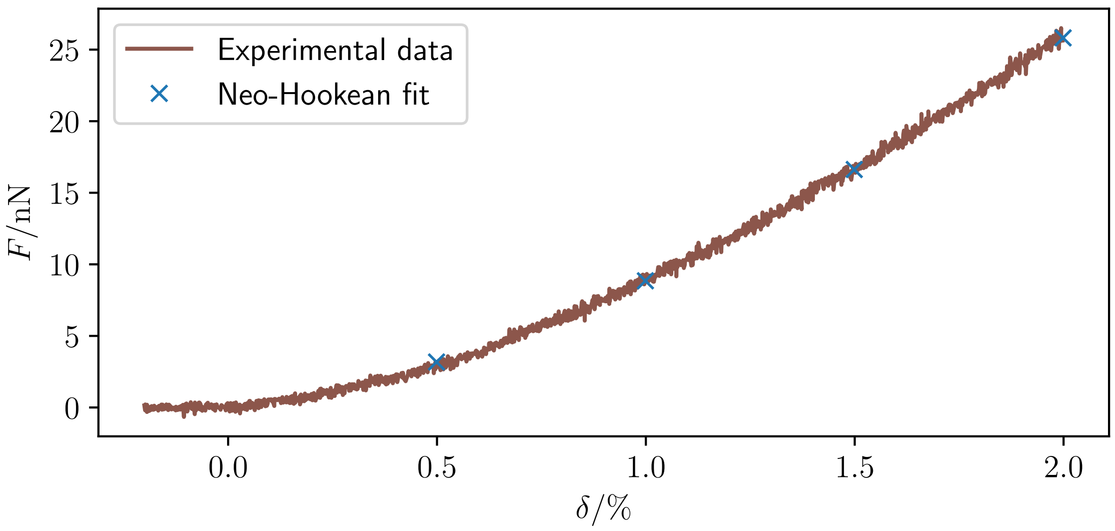


Figure 74: The corrected experimental data compared to a Neo-Hookean fit.

This corresponds to an average of $E_{2\,\%} = (33 \pm 1)\,\mathrm{kPa}$ for all the models. Extrapolating this to higher deformation does however reveal, that the parameters determined for small deformations are not suitable for all deformations. This as can be seen in figures 75 (linear) and 76 (logarithmic). While the most prominent feature is the sharp increase in the force compared to the Hertz model at around $\delta = 40\,\%$, the next section will first discuss the small drop below the Hertz model for medium deformations. This can best be seen in the linear plot.

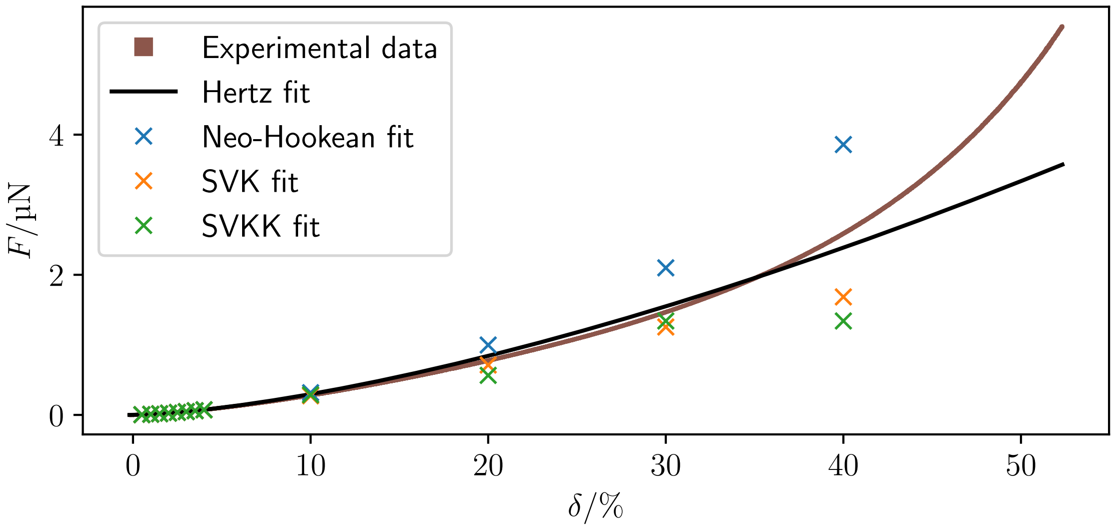


Figure 75: The corrected experimental data compared to the fits for each model found in this section.

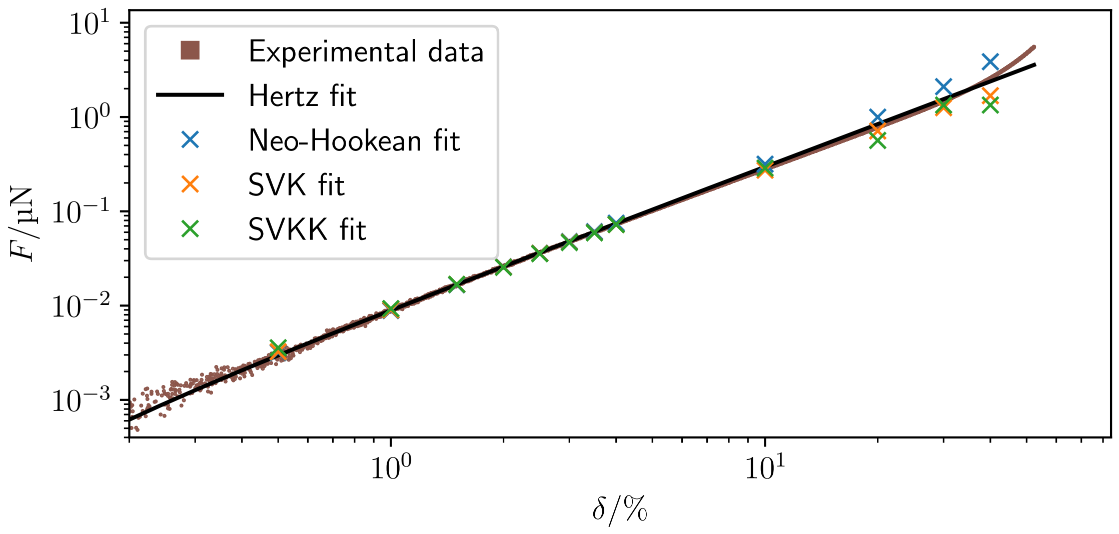


Figure 76: The corrected experimental data compared to the fits for each model found in this section. Logarithmic version.

20.4.4. Medium indentations

Medium indentations refer to deformations from 2 % to 10 % here. To best visualize the, the inverse of the Hertz model is used. This means, that each data-point gets an associated effective Young's module E_{eff}. This keeps the data in a smaller range and thus deviations are easier to spot. The Hertz model itself will produce a constant line in this display. This can be seen in figure 77 The fitting was done with the corrected force data from 2 % to 10 %. There is a decently sized range, where the Hertz model fits well. The determined Young's modulus is $E_{\text{Hertz, 2\,\% to 10\,\%}} = (32.215 \pm 0.007)\,\text{kPa}$ However, it is quite clear, that it does not reproduce the general shape of the data.

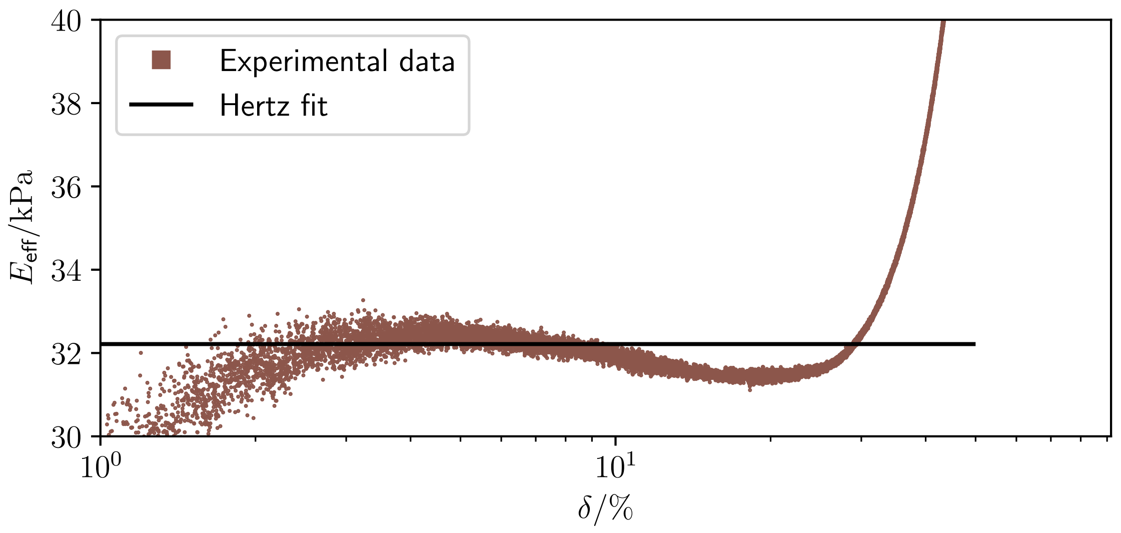


Figure 77: The effective Young's modulus of each point of the corrected experimental data compared to the Hertz fit.

The offsets are also very small compared to 1 % for the fits of medium indentations. It is however possible to see their impact in the smaller deformations. For this one can compare figure 78 to figure 77. It can be seen, that even such minuscule offset significantly influence the bottom 3 %. For anything above, they are largely irrelevant. Contrary to the results for small indentations, Neo-Hookean does not fit very swell for medium indentations as can be seen in figure 78. It rises, while the data falls. The data eventually does rise, but a lot later than Neo-Hookean and a lot sharper. Still, despite these discrepancies, the resulting Young's modulus ($E_{\text{Neo-Hookean, 2\% to 10\%}} = (31.5 \pm 0.7)\,\text{kPa}$) is still close to the Hertz result.

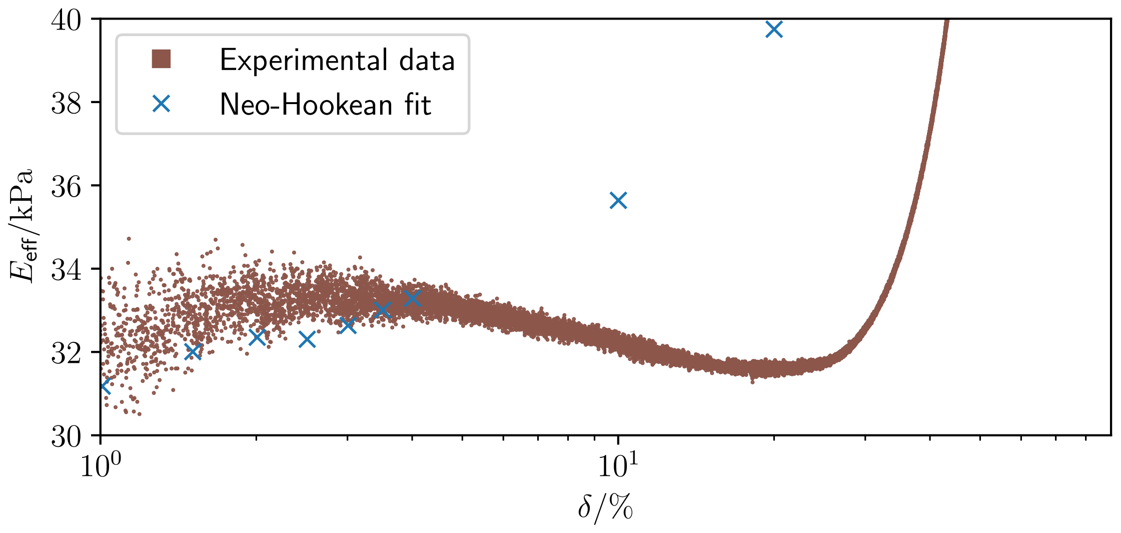


Figure 78: The effective Young's modulus of each point of the corrected experimental data compared to the Neo-Hookean fit.

The best fit in this regime can be achieved using the SVK model as can be seen in figure 79. The resulting Young's modulus is $E_{\mathrm{SVK},\,2\,\%\text{ to }10\,\%} = (35.09 \pm 0.09)\,\mathrm{kPa}$. The SVKK model looks very similar but results in a somewhat different Young's modulus of $E_{\mathrm{SVKK},\,2\,\%\text{ to }10\,\%} = (32.6 \pm 0.2)\,\mathrm{kPa}$.

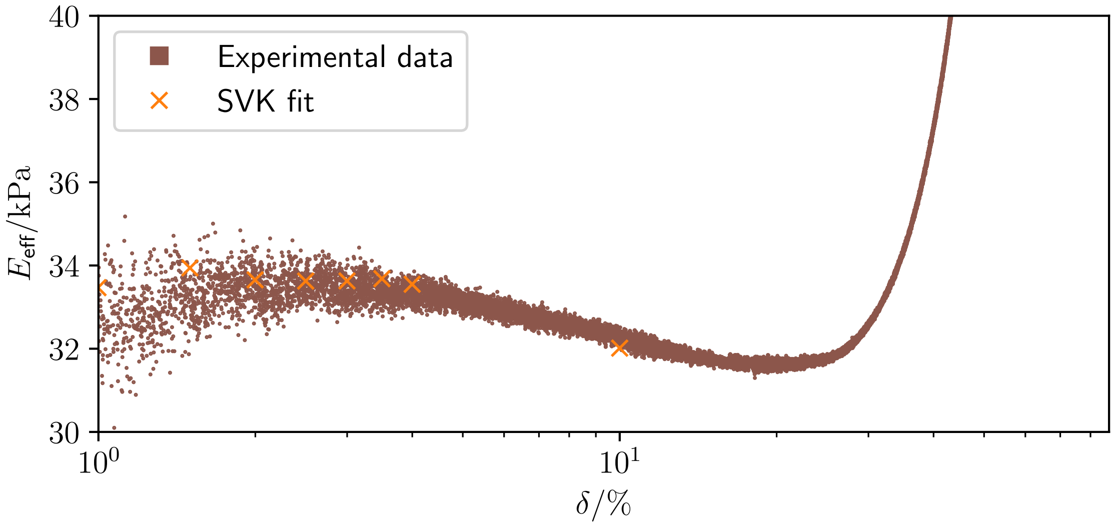


Figure 79: The effective Young's modulus of each point of the corrected experimental data compared to the SVK fit.

The failure of the Neo-Hookean fit in this range compared to SVK and SVKK demonstrates the linear elastic nature of the particles examined here. The average Young's modulus $E_{2\,\%\text{ to }10\,\%} = (33 \pm 2)\,\mathrm{kPa}$ is the same as for the small indentations, indicating, that up until here, the particle has not changed its behavior and is linear elastic. This will change for the large indentations discussed in the following.

20.4.5. Large indentations

For large indentations (here from 30 % to 50 %), SVK and SVKK are not stable. The Neo-Hookean model rises to early as has been shown above. The Hertz model does of course have no force-uptick compared to the Hertz model and is therefore also not suitable here. For this regime the Mooney-Rivlin (MR) model is used instead. It is steeper than the Neo-Hookean model, which is an edge-case of the MR model and can therefore more easily capture the steep rise seen in the data. The resulting parameters are however still problematic. The simulation data was recycled from Müller *et al.* [70] as he already produced MR data with a decent resolution in the parameter w. As was shown previously, this data is not to be trusted for small deformations, but as this subsection deals with large deformations, this is not an issue. Two sets of parameters work reasonably well. For one, with $w = 0.7$, the fit is decent for all deformations as can be seen in figure 81. However, particularly the medium deformations and to a lesser degree the large deformations do not line up very well as can be seen more clearly in figure 80.

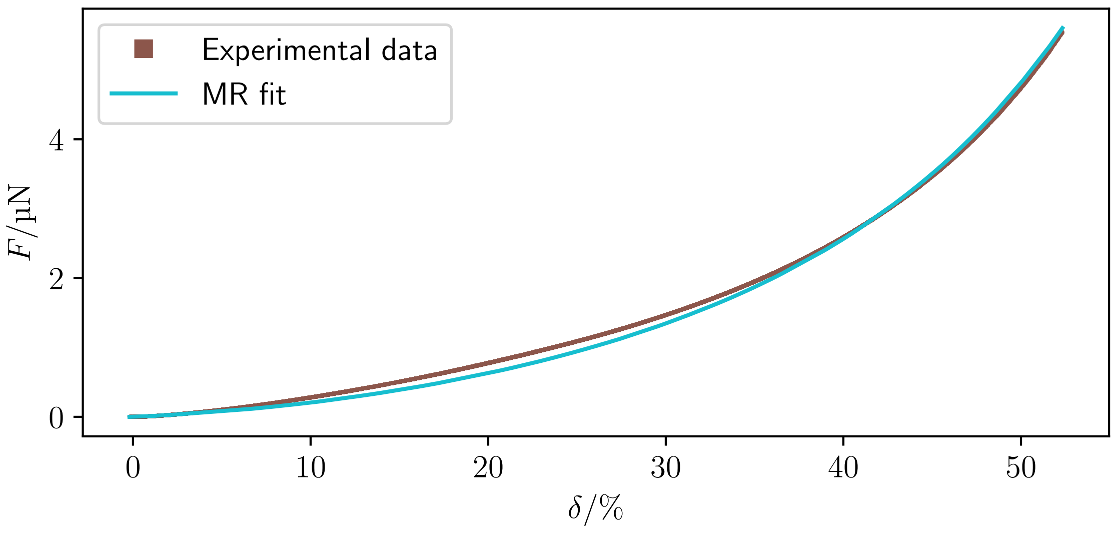


Figure 80: The corrected experimental data compared to a Mooney-Rivlin fit with $w = 0.7$.

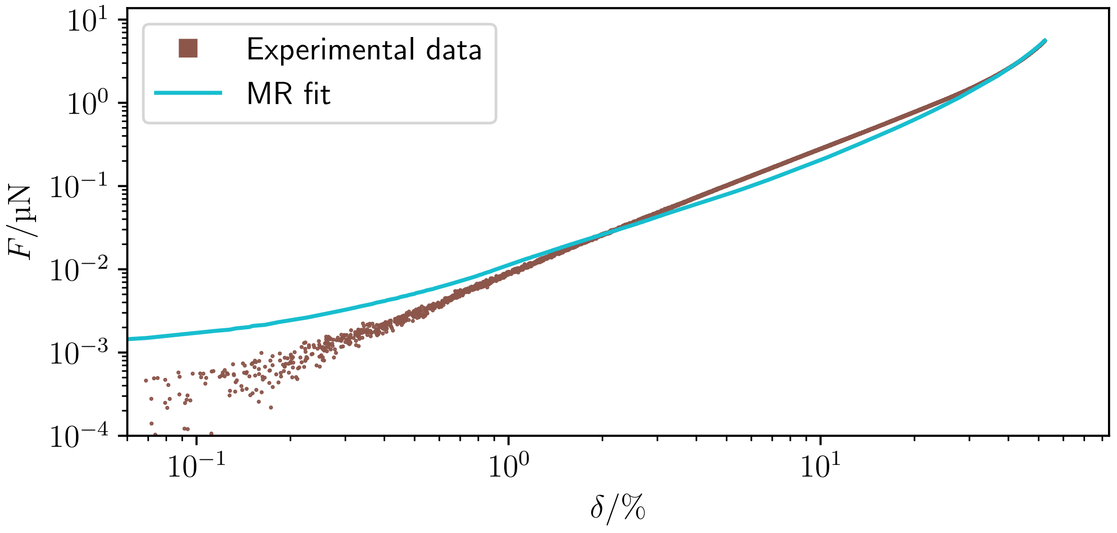


Figure 81: The corrected experimental data compared to a Mooney-Rivlin fit with $w = 0.7$. Logarithmic version.

On the other hand, the $w = 0.9$ is better at medium deformations and fits very well for large deformations as can be seen in figure 82. It however does not reproduce small deformations at all, as is made clear by figure 83. It should be reiterated, that this Mooney-Rivlin data is not accurate for small deformations anyway, so this might be a negligible issue. The resulting Young's moduli are $E_{\text{MR, 0.7, 30\,\% to 50\,\%}} = (18.441 \pm 0.006)\,\text{kPa}$ and $E_{\text{MR, 0.9, 30\,\% to 50\,\%}} = (19.295 \pm 0.003)\,\text{kPa}$. Here offsets of $(0.170 \pm 0.003)\,\%$ for the $w = 0.7$ case and $(0.9016 \pm 0.0004)\,\%$ for the $w = 0.9$ case are found. This is considerably larger than in previous sections. Nearly a full percentage point is tolerable compared to the range of data this subsection is examining, but it demonstrates, that the model does not fully reflect the data. This is also evident by the fact, that the Young's moduli found in this regime are over 40 % smaller than the ones found in the previous regimes. This illustrates, that while simulations are capable of capturing the behavior of these particles in all regimes, they are not able to do it with one model or even a consistent set of parameters. Another possible way to interpret that, is that these particles considerably change their behavior for large deformations.

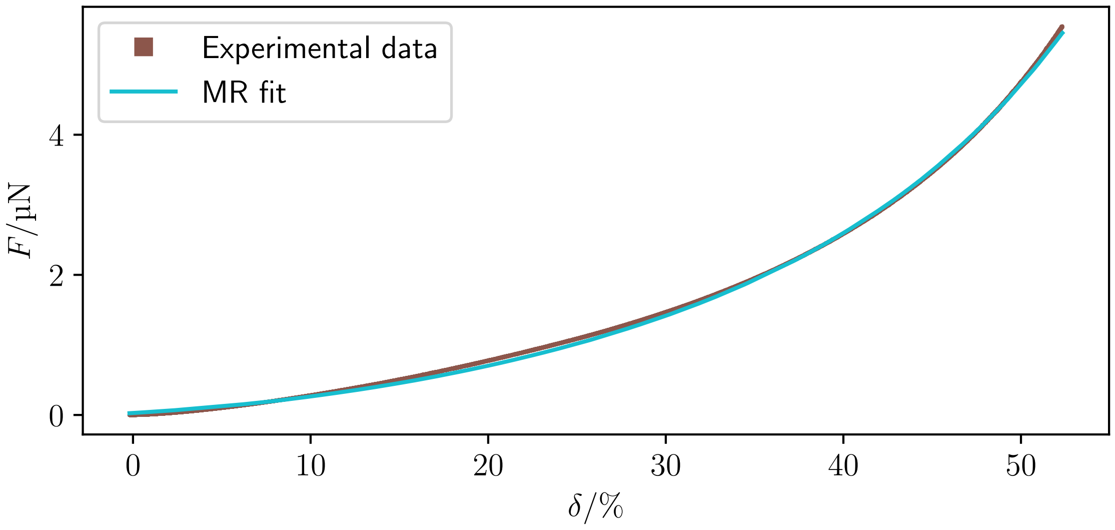


Figure 82: The corrected experimental data compared to a Mooney-Rivlin fit with $w = 0.9$.

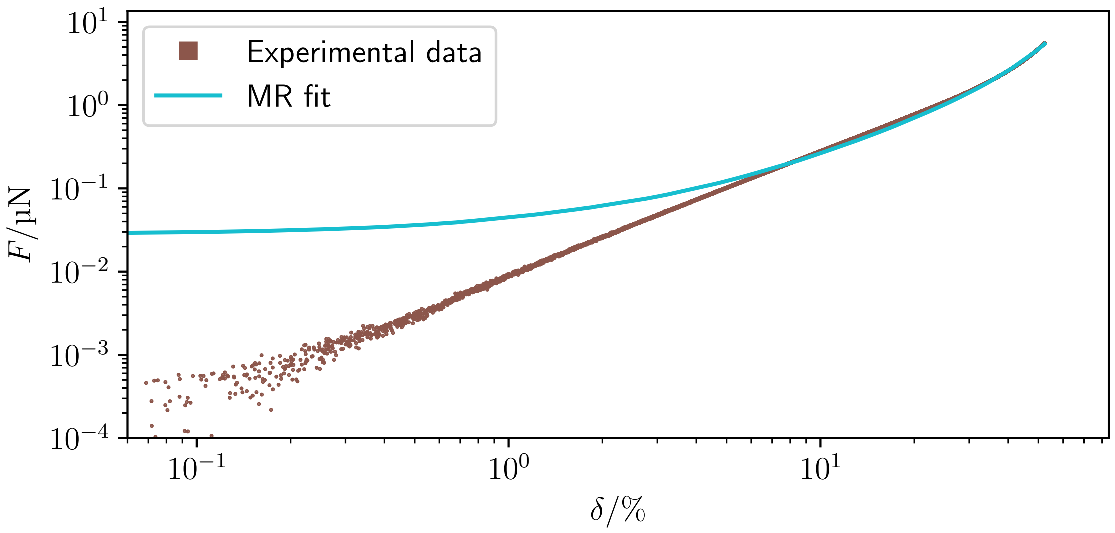


Figure 83: The corrected experimental data compared to a Mooney-Rivlin fit with $w = 0.9$. Logarithmic version.

The Mooney-Rivlin is not only able to reproduce the force these particles exert on the indenter, but also the shape the particles get deformed into. This is discussed next.

20.4.6. Reproducing the deformed shape

Steffen Trippmacher managed to capture a side-view video of the indentation experiment. This allows an attempt to reproduce the observed shape. Parts of this will be published in an upcoming paper. The use of the Mooney-Rivlin model is required to reproduce the forces for high indentation. The fit process described in the previous subsection was used with the new dataset to determine realistic parameters. These were used in simulation 14, to reproduce the shape at three different indentations.

14 Deforming a Mooney-Rivlin cell

Cell:	Young's modulus $E = 1\,\text{kPa}$
	Mooney-Rivlin ratio $w = 0.2$
	Radius, Resolution $R = 12.5\,\mu\text{m}, 50$

The indentations have to be considered separately to allow for relaxation. The video is converted to gray-scale to improve contrast and stabilized using edge detection. An overlay for the outline of the simulated cell is added. The resulting images can be seen in figures 84 – 86.

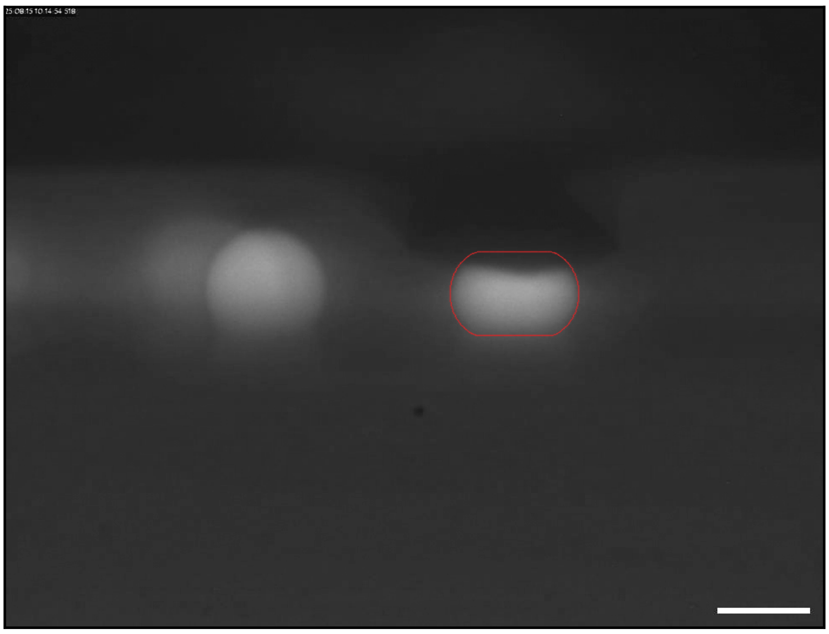


Figure 84: Overlay of the shape as simulated over the side-view video at $\delta = 30\,\%$ indentation.

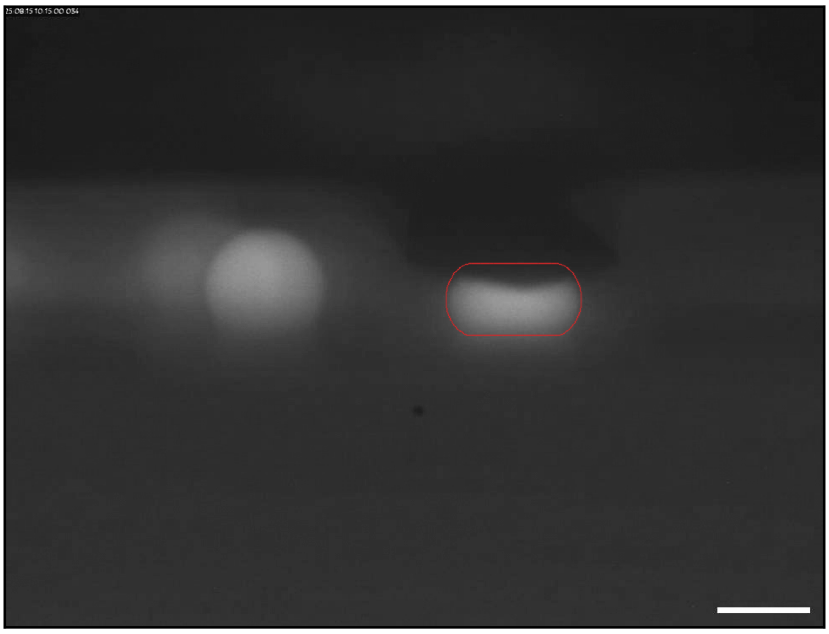


Figure 85: Overlay of the shape as simulated over the side-view video at $\delta = 40\,\%$ indentation.

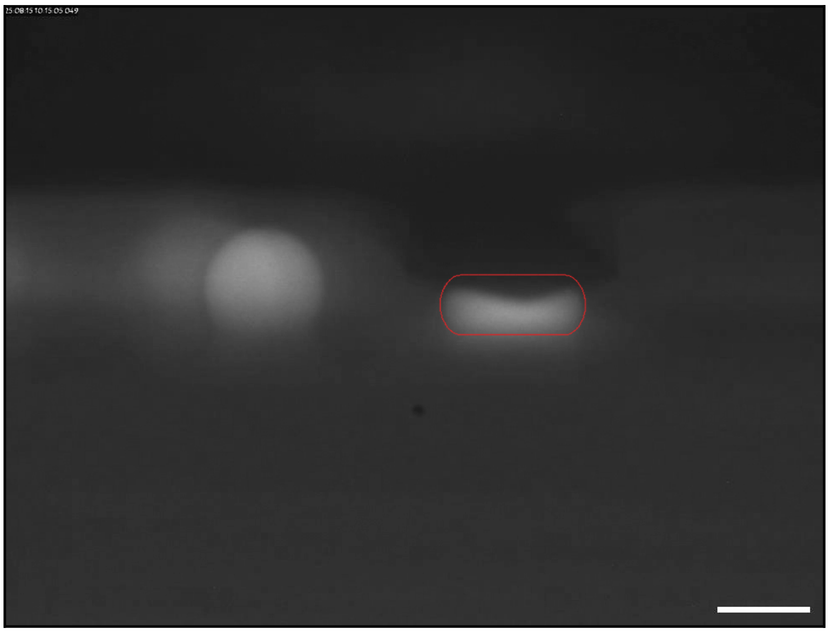


Figure 86: Overlay of the shape as simulated over the side-view video at $\delta = 50\,\%$ indentation.

The shape of the simulation (red lines) are in very good agreement with the experiment. In particular the bulging out of the particle in the horizontal direction is well captured by the simulation. Any error is below a single pixel of the experimental image. Notably, the parameters used for the simulation are determined independent of the video. The excellent agreement between experiment and simulation is achieved with no further fitting parameters. Not even the scaling factor needs adjustment. This highlights, that the simulations fully reproduce the real behavior of the particle in the experiment. More details can be found in the upcoming paper with Steffen Trippmacher. Once this is published, it can likely also be found on the GitHub account "Richardk2n". The changing behavior for different deformation regimes can also be observed for biological cells, which will be discussed next.

20.4.7. Biological cells

So far polymer based hydrogel beads have been considered. These do mimic biological cells to some degree, but show some different behavior. This warrants a separate discussion. The AFM data used in this subsection was taken from Müller *et al.* [70]. It was gathered from REF52 cells using FluidFM®. It is not nearly as smooth as the data gathered by Steffen Trippmacher for the polymer particles. This can be due to measurement method or biological activity. For the analysis here, only the smoothest plots were selected. The indices are the same as in Müller *et al.* [70]. For the visualizations a moving average has been performed to reduce noise. The fits were done using the raw data. Firstly, these exhibit significantly stronger non-linear behavior. This can be seen in figures 87 – 89. In this double logarithmic notation, the force-deformation curve of the linear elastic particles of the prior subsections are straight lines up to high deformations. For the biological cells in comparison stronger non-linearity is observed. This warrants the use of the Mooney-Rivlin model. The fit was limited to deformations from 30 % to 60 %, because the both the MR data and the experimental data is unreliable below. As the simulation data used for this model over-estimates the force for small deformations (see section 20.1), the discrepancy observed here does not necessarily indicate failure of the model. The upper ranges of deformation show good agreement. The found ideal parameters are as listed in table 9 below.

Index	E/Pa	w	offset/%
2	31.129 ± 0.007	0.2	7.390 ± 0.003
3	31.41 ± 0.06	0.05	6.54 ± 0.02
9	22.772 ± 0.009	0.15	9.089 ± 0.004

Table 9: Optimal fit parameters for the Mooney-Rivlin model fitted to the force-deformation curves of biological cells.

The high variance between these values is expected for biological cells. The data show too much noise at low deformations to find the true origin. Only fitting higher deformation does get around this issue. The high offset found by the fit is possible when looking at the raw data. It might in fact still be slightly too small. Over all, the Mooney-Rivlin model shows good agreement and is such suited to simulate biological cells. It should be noted, that the Young's moduli listed here are around $\frac{1}{6}$ of the values found by Müller *et al.* [70]. This is due to the range fitted and due to the vastly different choice in origin. In contrast to the fits presented by Müller *et al.*, the ones found here are nearly indistinguishable from the data when plotted on linear axis. Therefore, the values found here are likely more accurate. This illustrates once more, that the smaller deformation values traditionally evaluated in AFM experiments are not a reliable source to determine the Young's modulus.

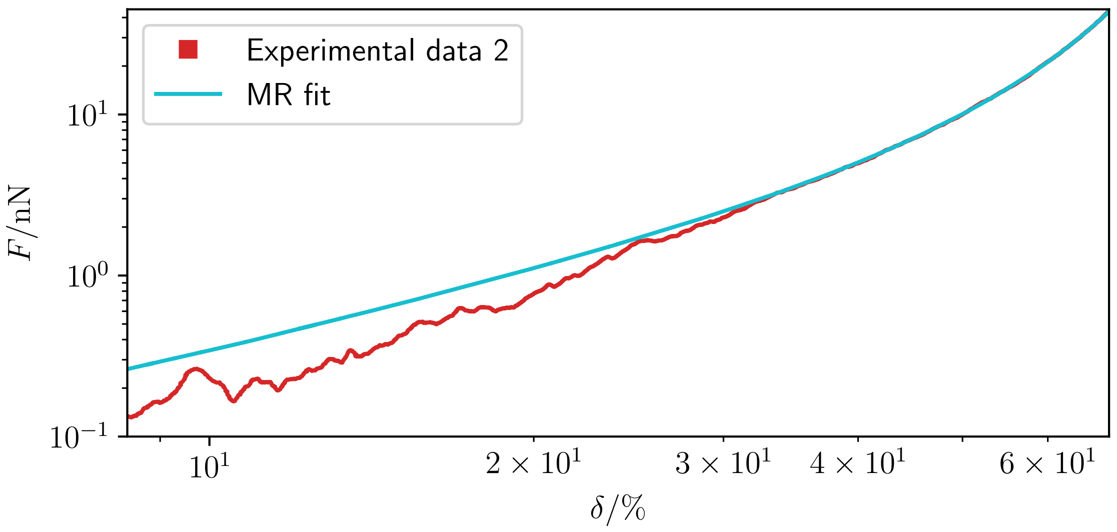


Figure 87: The experimental data for biological cells compared to a Mooney-Rivlin fit with $w = 0.2$. Logarithmic axis.

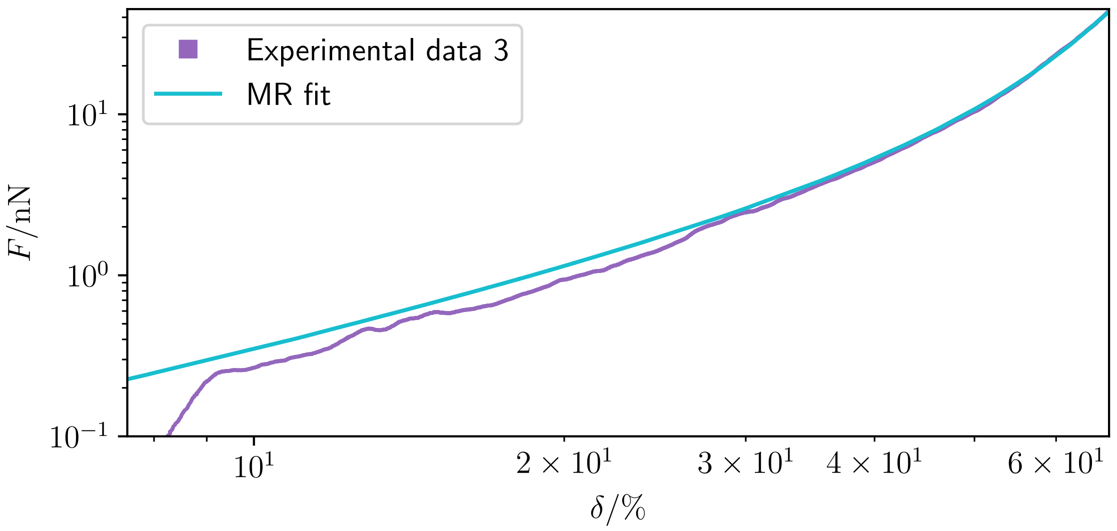


Figure 88: The experimental data for biological cells compared to a Mooney-Rivlin fit with $w = 0.05$. Logarithmic axis.

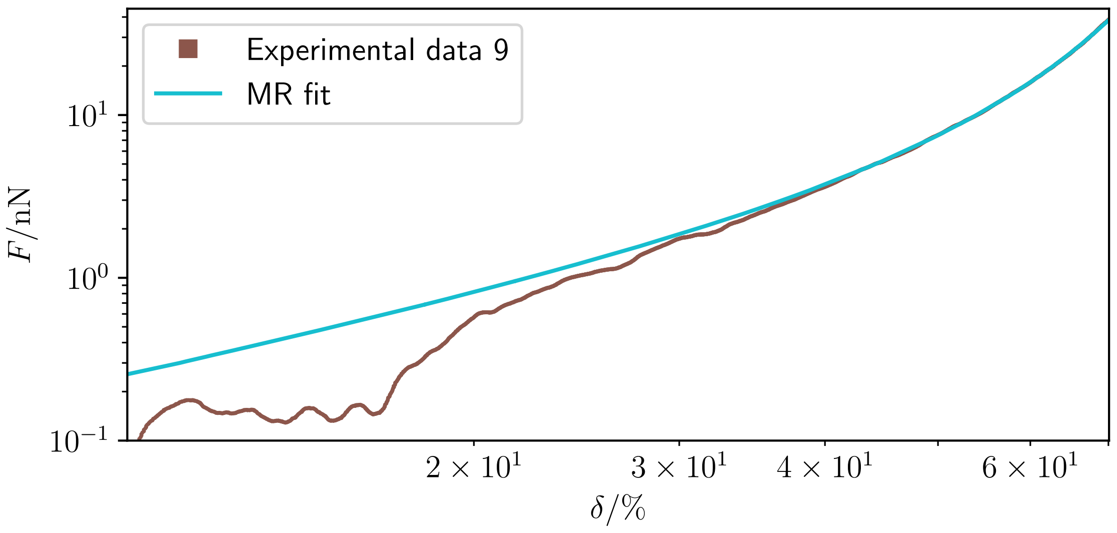


Figure 89: The experimental data for biological cells compared to a Mooney-Rivlin fit with $w = 0.15$. Logarithmic axis.

20.5. Conclusion

Traditionally one is forced to evaluate AFM experiments with the Hertz model, as it is the only analytical solution available. This limits the analysis to the smallest deformations. If the material behaves like the Neo-Hookean model deformations up to 3.5 % can be analyzed while keeping the error limited. For materials conforming to the SVK model, one can go twice as far. For materials conforming to the SVKK model, the Hertz model is even useful up to 10 %. If the data is not captured with outstanding accuracy, noise makes the evaluation of small deformations impossible. Even with good data, the true origin of the force-deformation curve carries enough error to significantly influence fitting results for small deformations, leading to inaccurate Young's moduli. This limits the usable range for Neo-Hookean materials to next to nothing. Furthermore, if the material at hand is not linear elastic, the use of the Hertz model introduces a non-negligible error even for deformations below 2 %. Only evaluating small deformations neglects materials changing their behavior for larger indentations, like the polymer beads. With the refined simulations introduced in this section, all these issues can be avoided, by moving to higher deformations. This approach assures accurate Young's moduli and a detailed description of the cells' behavior across a wide range of deformations. Given that deformations in experiments are typically larger than the valid range of the Hertz model, it is also more appropriate to look for the behavior at higher more realistic deformations. With such a simple deformation scenario finished, the next section will cover a slightly more complicated one. A shear-flow.

21. Roscoe in viscoelatic fluids

Cells placed in viscoelastic shear-flows show behavior not consistent with Roscoe theory (see section 9). A smaller than expected tank-threading has been reported for rigid particles [96] and elastic cells [100]. This effect has also been observed in simulations for rigid spheres [110]. The comparison is always Roscoe theory, which is defined for Newtonian fluids. This means, it is not entirely clear whether this effect is due to the shear-thinning properties or due to the normal stresses as stated in literature. Experimental data has the added issue, that it can only be evaluated from the visible quantities. In Roscoe theory, the Young's modulus can be used to predict a deformation, which in turn predicts the tank-threading frequency. Experiments cannot evaluate all axis and thus cannot measure the full deformation. The tank-treading frequency as predicted by the deformation consequently also carries an error. Also, if a deformation is predicted for a given Young's modulus using Roscoe theory, this is not necessarily the correct one. This is due to the possibility of the viscoelastic fluid changing the relationship between the Young's modulus and the deformation. The algorithms presented in this thesis are able to simulation shear-thinning (CY) and viscoelastic (PTT) fluids with the same viscosity profile. The comparison of simulations using these fluids is used in this section to determine how each contribution influences the adherence to Roscoe theory. In this section, the behavior of a cell in shear-flow is evaluated, for small Capillary numbers, for large Capillary numbers and for the values in between. First, small Capillary numbers, meaning a rigid sphere, are considered.

21.1. Tank-treading of a rigid cell

A rigid sphere placed in a pure shear-flow of a Newtonian fluid exhibits a tank-treading frequency of half the shear rate [95]. Due to limitations of IBM it is not possible to make a rigid sphere within the simulations presented here (see section 6.7). Instead, a hard cell is combined with moderate forces to produce a very small Capillary number of $Ca_\mathrm{K} \approx 0.06$.

Old simulations

The simulations in this subsection are 1.5 a old. Consequently, both the setups and evaluation scripts do not match the standard set in the remaining thesis. It is not recommended to use *FluidX3D* versions this old. The simulation should rather be redone with a newer version to reproduce these results.

Preliminary tests assured, that the tank-treading frequency does not change with the Reynolds number or a scaling of the fluid parameters. Small varying Capillary numbers are considered in simulation 15.

15 Roscoe for varying vanishing Ca_{K} in PTT

Box:	$182 \times 182 \times 182$
	$L_0 = 1.25\,\mu\mathrm{m}$
Fluid:	constructed PTT
BC:	velocity (BC 2)
	$\dot{\gamma} = 1 \times 10^4\,\frac{1}{\mathrm{s}}$
Cell:	Young's modulus variable
	Radius, Resolution $R \approx 7.5\,\mu\mathrm{m}, 6$
	Capillary number variable

The fluid in this simulation was constructed to hold as many dimensionless numbers constant as possible. The Capillary number is held small enough to mimic a rigid cell with negligible deformation. This yields a roughly constant tank-treading frequency. This can be seen in figure 90.

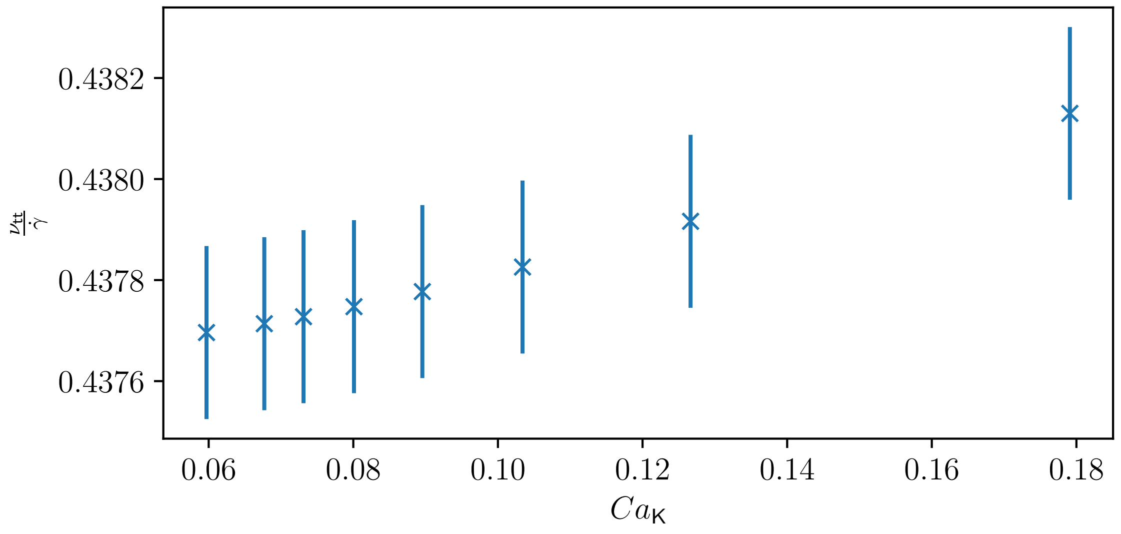


Figure 90: Plot of the tank-treading frequency ν_{tt} as a fraction of the shear-rate $\dot{\gamma}$ as a function of the Capillary number Ca_{K}. Fitting errors are provided.

For a rigid sphere, the tank-treading frequency is expected to be half of the shear-rate. Evidently it is more than 10 % lower than expected. This is consistent for multiple simulations and thus a real effect. The variation for small Capillary numbers is remarkably small. As the elastic forces could be suspected of causing the reduced tank-threading frequency, this is evaluated. For this simulation 16 varies the ϵ parameter of the PTT fluid.

16 Roscoe for varying stress fraction in PTT

Box:	$182 \times 182 \times 182$
	$L_0 = 1.25\,\mu\text{m}$
Fluid:	constructed PTT
BC:	velocity (BC 2)
	$\dot{\gamma} = 1 \times 10^4\,\frac{1}{\text{s}}$
Cell:	Young's modulus $E = 29.6\,\text{kPa}$
	Radius, Resolution $R \approx 7.5\,\mu\text{m}, 6$
	Capillary number variable

One can scale the fluid viscosity parameters simultaneously to not alter the viscosity curve (see section 3.7.4). The resulting curve can be seen in figure 91.

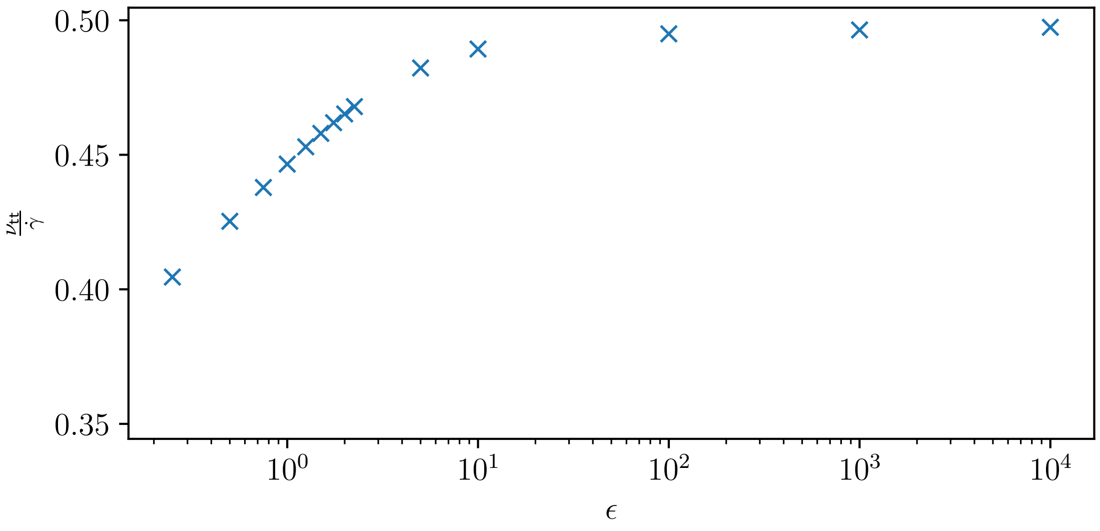


Figure 91: Plot of the tank-treading frequency ν_{tt} as a fraction of the shear-rate $\dot{\gamma}$ as a function of the PTT parameter ϵ.

Small ϵ cause the tank-treading frequency to drop, while it reaches the expected value of half the shear-rate for sufficiently large ϵ. One can see, that this scaling alters the fraction between the elastic normal stress and the viscous shear stress (see section 3.7.4). The tank-treading frequency as a function of this fraction can be seen in figure 92.

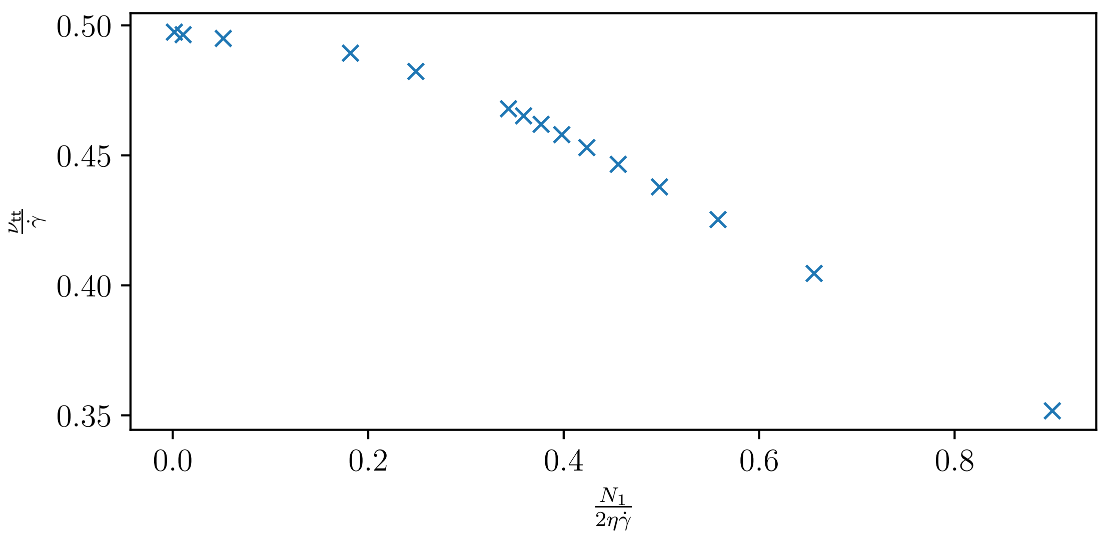


Figure 92: Plot of the tank-treading frequency ν_{tt} as a fraction of the shear-rate $\dot{\gamma}$ as a function of the fraction of normal stress and shear stress.

Clearly, the expected behavior is reproduced for vanishing shear stress. For larger fractions a linear relationship develops. This demonstrates the drop in tank-treading frequency reported in literature. It shows, that this is indeed due to the elastic normal stress. Assuming this linear relationship holds, this could be used in experiments to measure the magnitude of the normal stress. The discussion so far was limited to rigid cells. Next cells with finite deformation shall be discussed.

21.2. Cell behavior at constant Ca_{K}

Holding the Capillary number constant while varying the shear-rate alters the local viscosity and the slope of the local viscosity. Furthermore, the relative amplitude of the elastic and viscous forces are altered. Comparing the extracted Roscoe parameters (see section 10) for the viscous and viscoelastic fluid to the prediction from Roscoe theory (see section 9) reveals the influence of both the shear thinning and the elastic forces. The Capillary number is held constant at $Ca_{\mathrm{K}} = 2$. Consequently, the Young's modulus is calculated as follows (see eq. (153)).

$$E = 4\eta(\dot{\gamma})\dot{\gamma} \tag{358}$$

With this, multiple shear-rates are probed in simulations 17 and 18 for the viscous and viscoelastic fluid respectively.

17 Roscoe for constant Ca_{K} in CY

Box:	$100 \times 500 \times 100$
	$L_0 = 625\,\mathrm{nm}$
Fluid:	CY::mc0_49 (fluid 8)
BC:	velocity (BC 2)
	$\dot{\gamma}$ variable
Cell:	Young's modulus variable
	Radius, Resolution $R \approx 7.5\,\mathrm{\mu m}, 12$
	Capillary number $Ca_{\mathrm{K}} = 2$

18 Roscoe for constant Ca_{K} in PTT

Box:	$100 \times 500 \times 100$
	$L_0 = 625\,\mathrm{nm}$
Fluid:	PTT::mc0_49 (fluid 2)
BC:	velocity (BC 2)
	$\dot{\gamma}$ variable
Cell:	Young's modulus variable
	Radius, Resolution $R \approx 7.5\,\mathrm{\mu m}, 12$
	Capillary number $Ca_{\mathrm{K}} = 2$

The deformation parameters α_1 and α_2, which are typically visible in experiments are shown in figures 93 and 94

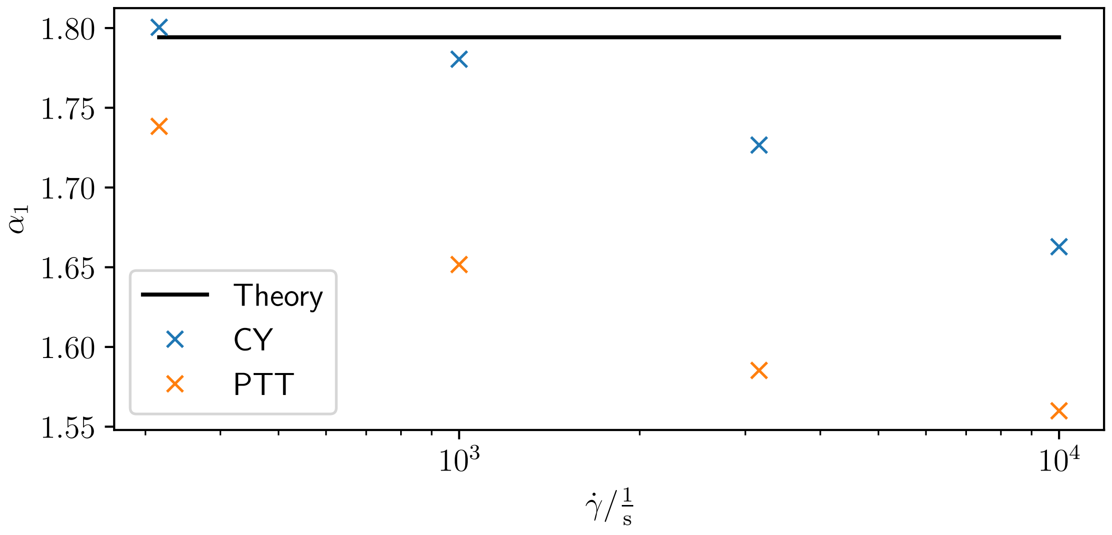


Figure 93: Plot of the Roscoe deformation parameter α_1 as a function of the shear-rate $\dot{\gamma}$. Curves for Roscoe theory, a CY simulation and a PTT simulation are shown.

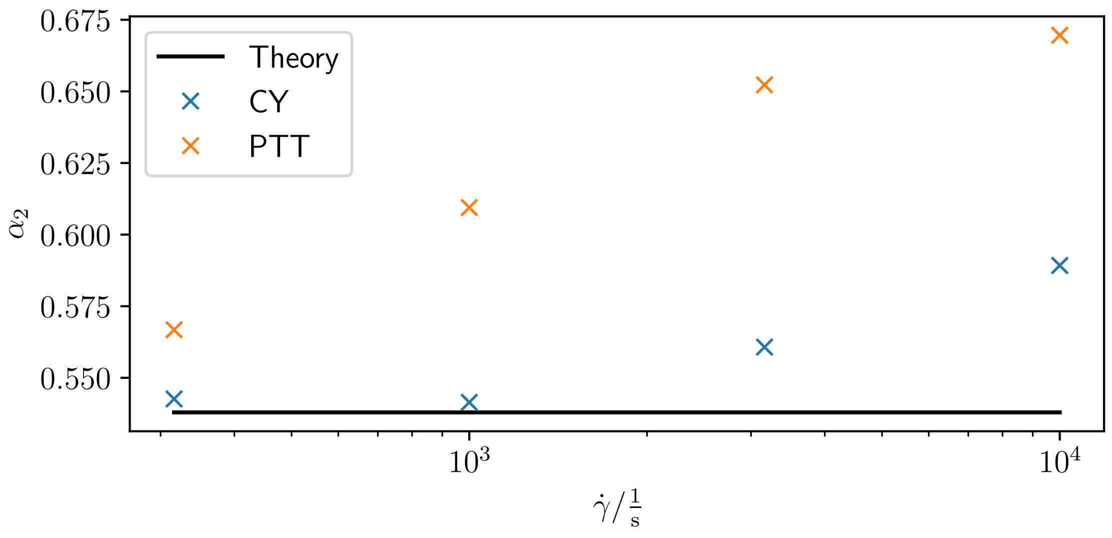


Figure 94: Plot of the Roscoe deformation parameter α_2 as a function of the shear-rate $\dot{\gamma}$. Curves for Roscoe theory, a CY simulation and a PTT simulation are shown.

Both fluids are less deformed than predicted by theory. The discrepancy grows with the shear-rate. Both the viscous and the viscoelastic fluid show roughly the same dependence on the shear-rate. Consequently, this can be interpreted as a shear-thinning effect. The fluids considered here increase their shear-thinning slope in this shear-rate range (see section 19). This also supports this hypothesis. The elastic stress causes an offset, with no significant dependence on the shear-rate. As the magnitude of the elastic stresses in relation to the viscous stresses behaves roughly like the root of a logarithm of the shear-rate (see section 3.7.4), no large change in the influence is to be expected. The significant decrease of the deformation due to the elastic stresses is notable. If one would treat the fluid as Newtonian and apply Roscoe theory, one would significantly underestimate the Capillary number. This deformation can be rationalized using the sketch in section 12.2. The elastic forces increase α_2, forcing α_1 and α_3 to shrink for volume conservation. This explanation is supported by the behavior of α_3 depicted in figure 95.

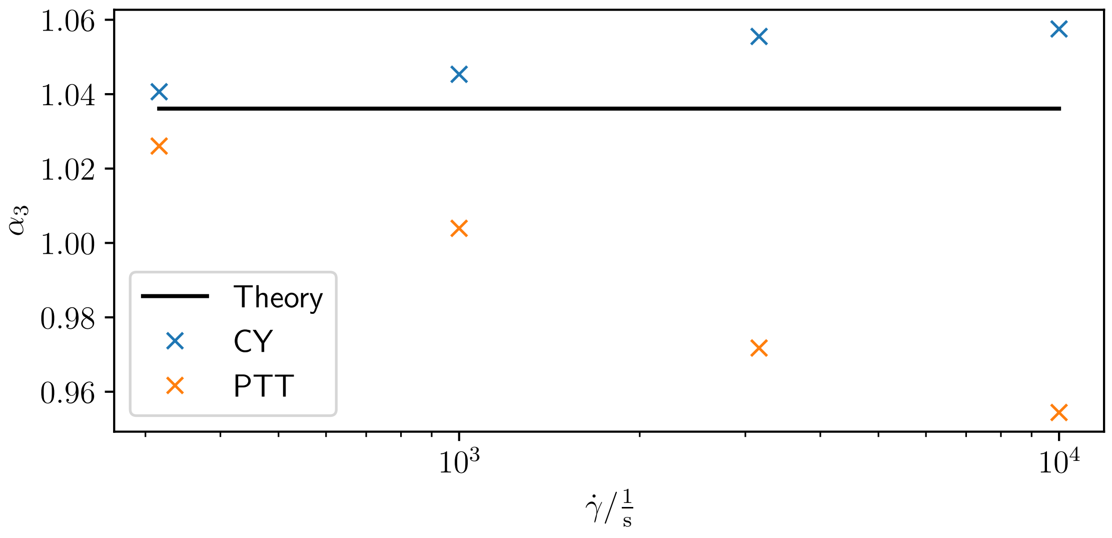


Figure 95: Plot of the Roscoe deformation parameter α_3 as a function of the shear-rate $\dot{\gamma}$. Curves for Roscoe theory, a CY simulation and a PTT simulation are shown.

Here, the behaviors of the viscous fluid and the viscoelastic fluid differ greatly. In Roscoe simulations, α_3 is normally just slightly above unity, even for strongly deformed cells. The shear-thinning causes it to increase a little more. The normal forces on the other hand cause it to drop significantly below unity. This is consistent with the behavior of α_1 and α_2 seen above. Notable, someone doing an experiment typically does not see this axis. This would lead to an overestimation of the cell volume and an underestimation of α_1 and α_2. The cell volume is roughly conserved for both fluids as can be seen in figure 96.

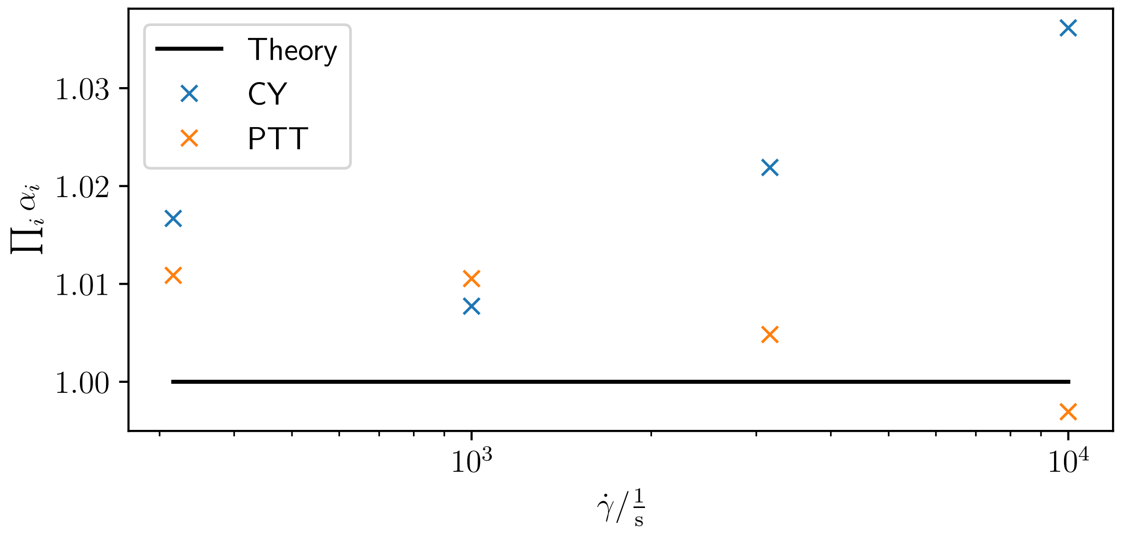


Figure 96: Plot of the cell volume as calculated from the Roscoe deformation parameters as a function of the shear-rate $\dot{\gamma}$. Curves for Roscoe theory, a CY simulation and a PTT simulation are shown.

Some discrepancies are to be expected as these simulations are not fully accurate (see appendix X). The alignment parameter θ also shows a difference as can be seen in figure 97.

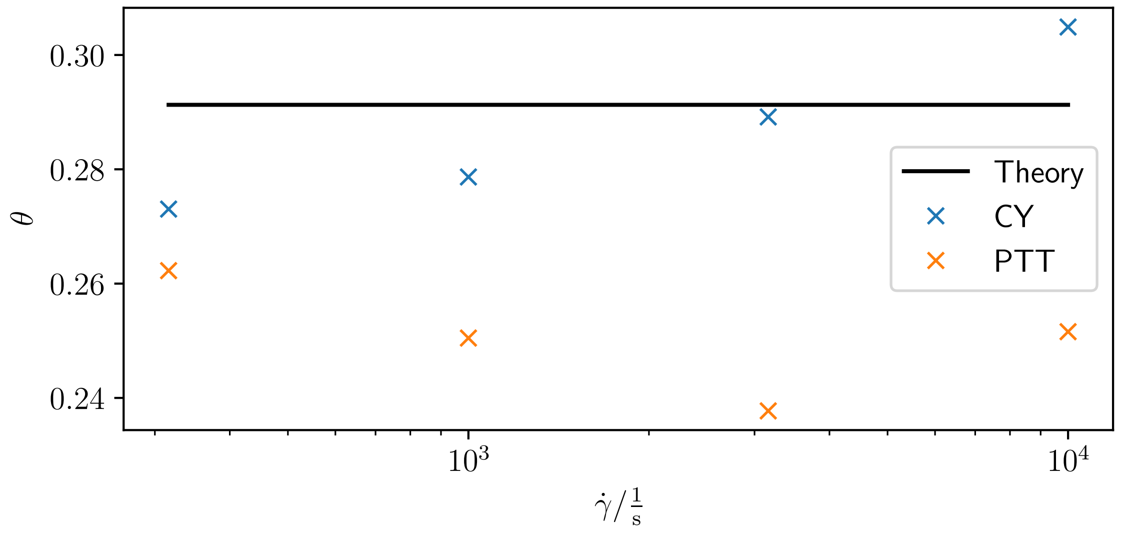


Figure 97: Plot of the Roscoe alignment parameter θ as a function of the shear-rate $\dot{\gamma}$. Curves for Roscoe theory, a CY simulation and a PTT simulation are shown.

The shear thinning fluid stays close to the angle predicted by theory. It does start out below and crosses over to higher values for larger shear-rates. The viscoelastic case shows a roughly constant value at a significantly smaller alignment angle. This means, that for the viscoelastic fluid, the largest cell axis is closer to the velocity direction. This too can be rationalized using the sketch in section 12.2. The elastic force can be separated into a component, which does reduce the alignment angle and one, which does not oppose this. For smaller deformations, Roscoe theory predicts a faster tank-treading. For a smaller alignment angle, slower tank-treading is predicted. The total effect of these two opposing effects can be seen in figure 98.

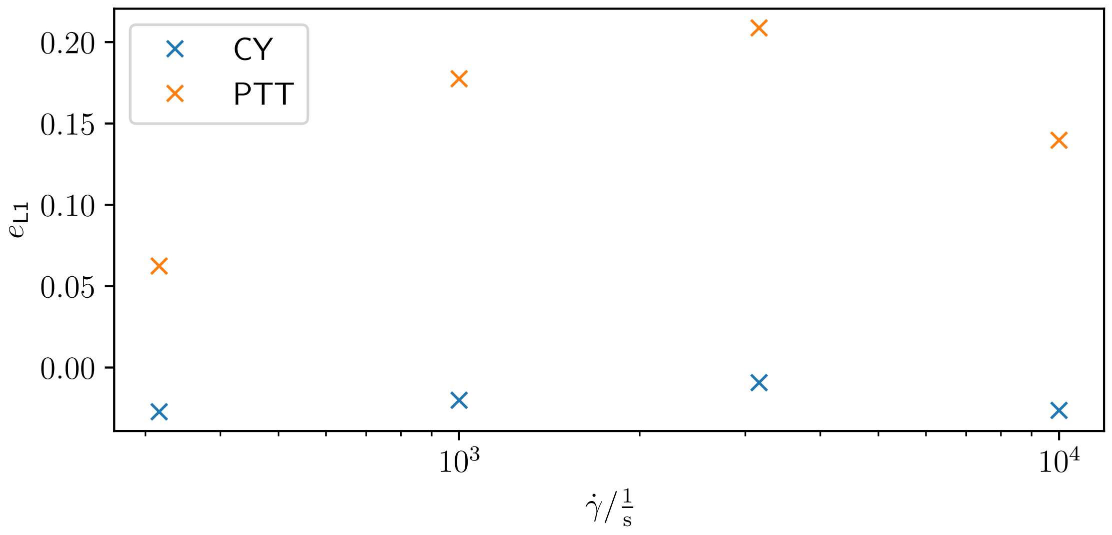


Figure 98: Plot of the L1 error e_{L1} of the tank-treading frequency ν_{tt} as a function of the shear-rate $\dot\gamma$. Curves for Roscoe theory, a CY simulation and a PTT simulation are shown.

Here the L1 error e_{L1} (see appendix S), which is defined as follows has been used for visibility.

$$e_{\mathrm{L1}} = \frac{\nu_{\mathrm{tt}}}{\nu_{\mathrm{Theory}}} - 1 \tag{359}$$

The shear thinning fluid is just slightly below the prediction by Roscoe. For these parameters it could be viewed as accurately reproduced. The viscoelastic fluid causes the tank-treading frequency to increase significantly. No clear trend can be seen, and it might be called roughly constant. This 15 % to 20 % increase in the tank-treading frequency is a bit unexpected given that the literature prompting this investigation reported a lowering of the frequency. Also, rigid cells reproduce this reported drop in the tank-treading frequency. The reason behind these results, which seem to be conflicting will be explored next.

21.3. Cell behavior at varying Ca_{K}

In the previous subsections, the cell behavior was explored for small and large Ca_{K}. Here, the point in between shall be investigated. The Capillary number is varied by adjusting the Young's modulus. Two simulations are performed, simulation 20 with a shear-thinning CY fluid and simulation 20 with a viscoelastic fluid.

19 Roscoe for varying Ca_K in CY

Box:	$100 \times 500 \times 100$
	$L_0 = 625\,\mathrm{nm}$
Fluid:	CY::mc0_49 (fluid 8)
BC:	velocity (BC 2)
	$\dot{\gamma} = 1 \times 10^4\,\frac{1}{\mathrm{s}}$
Cell:	Young's modulus variable
	Radius, Resolution $R \approx 7.5\,\mathrm{\mu m}, 12$
	Capillary number variable

20 Roscoe for varying Ca_K in PTT

Box:	$100 \times 500 \times 100$
	$L_0 = 625\,\mathrm{nm}$
Fluid:	PTT::mc0_49 (fluid 2)
BC:	velocity (BC 2)
	$\dot{\gamma} = 1 \times 10^4\,\frac{1}{\mathrm{s}}$
Cell:	Young's modulus variable
	Radius, Resolution $R \approx 7.5\,\mathrm{\mu m}, 12$
	Capillary number variable

The Roscoe deformation parameters α_1 and α_2 can be seen in figures 99 and 100 respectively.

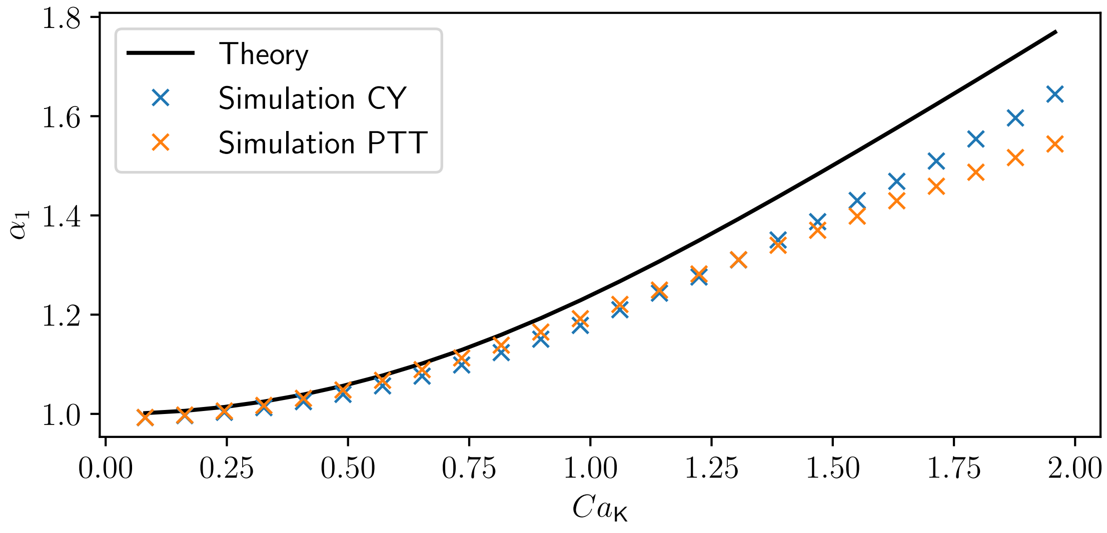


Figure 99: Plot of the Roscoe deformation parameter α_1 as a function of the Capillary number Ca_K. Curves for Roscoe theory, a CY simulation and a PTT simulation are shown.

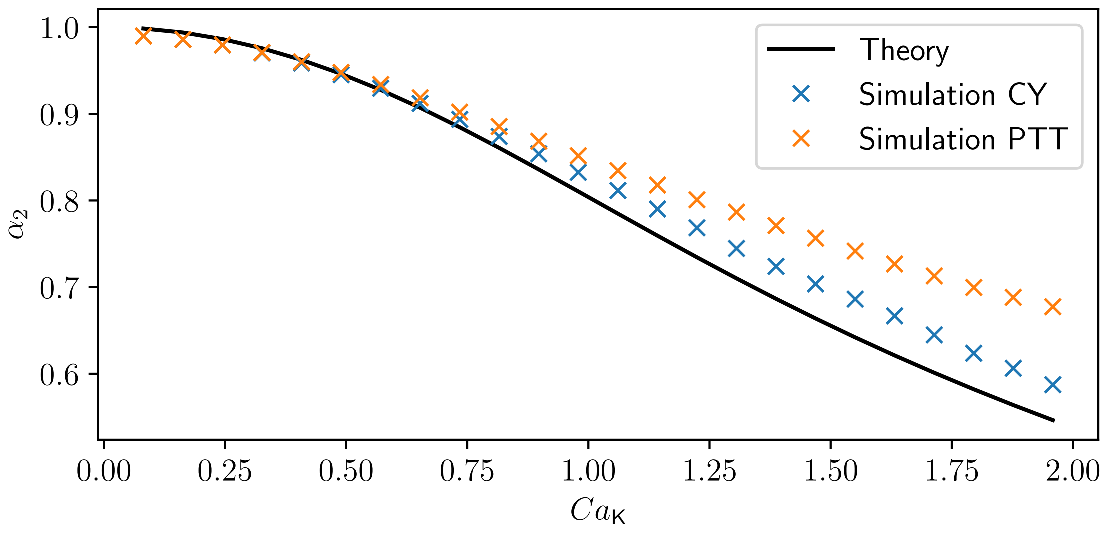


Figure 100: Plot of the Roscoe deformation parameter α_2 as a function of the Capillary number Ca_K. Curves for Roscoe theory, a CY simulation and a PTT simulation are shown.

For small Capillary numbers, both models adhere to Roscoe theory. At times, the viscoelastic curve is even closer to the theory than the shear thinning curve. However, mostly, this is within error. For larger Capillary numbers, both curves start to depart from the theory curve. Both exhibit less deformation than the Newtonian Roscoe theory would predict. The effect is considerably stronger for the viscoelastic curve. It shall be noted, that the discrepancy between the models is larger for α_2. This is consistent with the explanation from before about the normal force primarily elongating α_2. The third deformation parameter α_3 behaves similarly, as can be seen in figure 101.

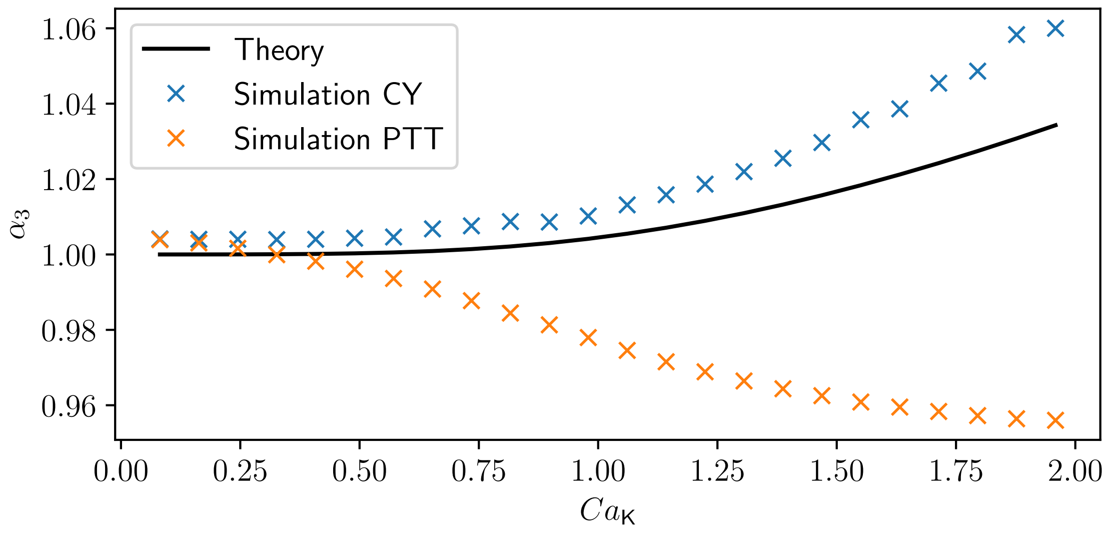


Figure 101: Plot of the Roscoe deformation parameter α_3 as a function of the Capillary number Ca_K. Curves for Roscoe theory, a CY simulation and a PTT simulation are shown.

The shear-thinning fluid stays close to the Roscoe curve, while the viscoelastic curve departs for larger values. Notably, the discrepancy here is in different directions. The elastic force causes the cell to strongly contract along α_3. Unsurprisingly, the volume is mostly conserved in the whole parameter space as can be seen in figure 102.

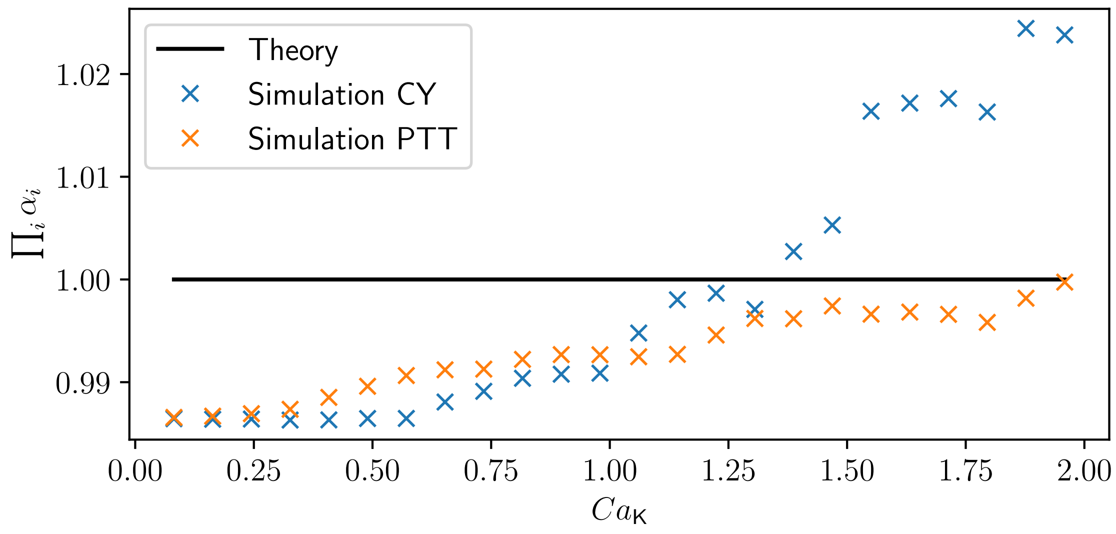


Figure 102: Plot of the cell volume as calculated from the Roscoe deformation parameters as a function of the Capillary number Ca_K. Curves for Roscoe theory, a CY simulation and a PTT simulation are shown.

So far the behavior was mostly in line with the results for a constant finite Capillary number. The alignment angle however shows larger differences. This can be seen in figure 103.

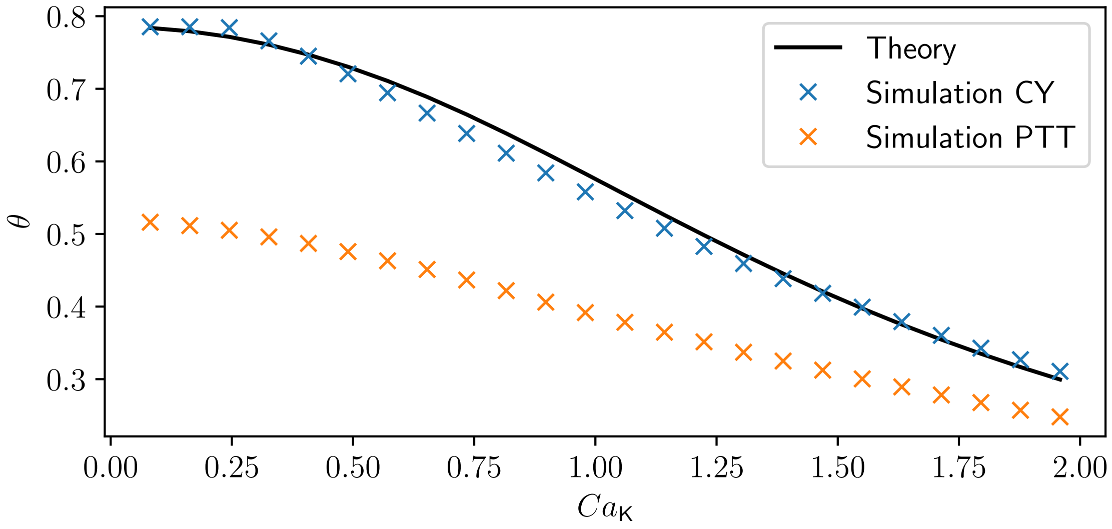


Figure 103: Plot of the Roscoe alignment parameter θ as a function of the Capillary number Ca_K. Curves for Roscoe theory, a CY simulation and a PTT simulation are shown.

The shear-thinning fluid reproduces the Roscoe curve accurately. The viscoelastic curve is however closer to half of the expected value. For small deformation, the CY fluid and Roscoe theory approach an angle of $\theta \approx 45°$. The PTT fluid however tends towards $\theta \approx 30°$. The tilting effect of the elastic force is considerably larger than expected from the previous subsection. This finally leaves the tank-treading frequency left to discuss. It can be seen in figure 104.

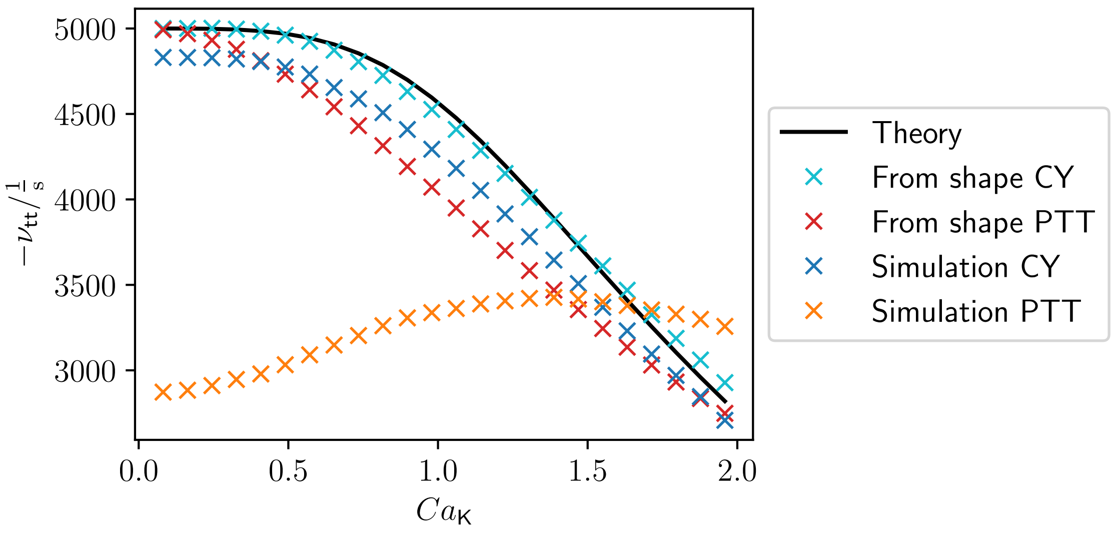


Figure 104: Plot of the tank-treading frequency ν_{tt} as a function of the Capillary number Ca_{K}. Curves for Roscoe theory, a CY simulation and a PTT simulation are shown.

Here two additional curves have been introduced. Roscoe theory allows the calculation of the tank-treading frequency from the visible deformation (see eq. (244)). Namely, the parameters α_1, α_2 and θ are used for this. With the approximation $\alpha_3 \approx 1$, all of these are accessible from a typical experiment. This is used in experimental settings to calculate the predicted tank-treading frequency. If one calculates the tank-treading frequency from the shape using eq. (244), one would get the same prediction as calculating the value based on the viscosities and Young's modulus. The observed value is still lower, but this slowing of the tank-treading frequency due to shear-thinning is not large. For the viscoelastic fluid, the story is quite different. The prediction from the shape is already off by quite a bit. The actually observed value is completely different. For small Capillary numbers, the cell rotates only with about half the expected frequency. However, the tank-treading frequency stays in a narrow band for the viscoelastic fluid. Consequently, it exceeds the prediction for higher Capillary numbers. This reconciles the seemingly conflicting observations from before. A few remarks follow.

21.4. Common remarks

The observed deformations are altered by the presence of elastic forces. The reduction they experience is of moderate size. Consequently, the surface of the cell can still be described using the tools of Roscoe theory (see section 10). However, the tank-treading frequency is altered massively depending on regime. The fact, that the frequency increases or decreases depending on deformation makes this effect hard to track down. The region in which the tank-treading is slowed is $Ca < 0.5$ using the typical definition of the Capillary number. Being below unity, interesting effects are not necessarily expected here. In conclusion, an interesting effect reported in literature has been successfully reproduced here. Also, it has been shown, that the effect reverses for large Capillary numbers. Finding the Young's modulus using Roscoe theory at small Capillary numbers in a viscoelastic fluid is not possible. The simulations shown in this section do however provide a way to relate the base parameters to the tank-treading frequency, even in complex fluids. This section describes how a cell can act within a bioprinting needle. The next section will cover what happens at the end of the needle.

22. Exiting the printer needle

During the bioprinting process, the bio-ink carrying cells is extruded through a thin needle. This needle deposits the cells line by line, in the same way a typical 3D printer works. The resolution of the print is given by the needle size. Assuming the velocity in the needle is constant, the total print time scales with the inverse cube of the needle size. This means, that higher resolutions slow down the printing significantly. Consequently, very high flow velocities are desired within the needle to be able to move the printer faster and speed the process up. This leads to large stresses in the needle, which are harmful to the cells. One of the arguments for using shear-thinning bio-inks is, that the velocity profile within the needle is flattened. This somewhat approximates a plug flow[29], which has a viscous stress equal to 0. However, it has been shown by Müller *et al.* [67], that in these cases, the cell experiences a large amount of stress at the needle exit. The publication by Müller *et al.*considered shear-thinning fluids. Viscoelastic fluids as discussed in this thesis are retarded. This means, that the polymer-conformation tensor does not react instantly to the end of the needle. Some time, and therefore distance, is required for it to adapt to the new flow conditions. One might imagine, that this allows a smoother transition without the sudden shock the cell experiences at the end of the printer needle. To do this, the same geometry as the one used by Müller *et al.*is employed to mimic a printer needle. This consists of a typical 3D Poiseuille channel. After approximately 267 µm, the no slip walls of the needle are replaced by free slip walls mimicking the free flow. With this geometry, two simulations are performed. The first simulation (simulation 21) is only shear-thinning. It provides the purely viscous reference. An average flow velocity of $\overline{v}_x = 5\,\frac{\text{mm}}{\text{s}}$ is picked in accordance with the aforementioned publication. This is set on the inflow. Within a few lattice nodes, the proper profile develops. As a preparation, simulation 59 was run to provide a stable fluid field in equilibrium into which the cell can be placed.

21 CY printer needle

Box:	$902 \times 77 \times 77$
	$L_0 = 1.\overline{3}\,\mu\text{m}$
Fluid:	CY::mc0_49 (fluid 8)
BC:	Inflow: mass-flow (BC 3)
	$\overline{v}_x = 5\,\frac{\text{mm}}{\text{s}}$
	Outflow: pressure (BC 4)
	slip walls (BC 6)
	simulation 59
Cell:	Young's modulus $E = 29.6\,\text{Pa}$
	Radius, Resolution $R = 8\,\mu\text{m}, 6$

To contrast this, simulation 22 is run in the same way with a viscoelastic fluid. The preparation for this is simulation 60. The fluid here matches the shear-thinning fluid in

[29] A flow profile with constant velocity.

its viscosity profile. This highlights the differences caused by the elastic stress and the retardation of the fluid.

22 PTT printer needle

Box:	$902 \times 77 \times 77$
	$L_0 = 1.\overline{3}\,\mu\text{m}$
Fluid:	PTT::mc0_49 (fluid 2)
BC:	Inflow: mass-flow (BC 3)
	$\overline{v}_x = 5\,\frac{\text{mm}}{\text{s}}$
	Outflow: pressure (BC 4)
	slip walls (BC 6)
	simulation 60
Cell:	Young's modulus $E = 29.6\,\text{Pa}$
	Radius, Resolution $R = 8\,\mu\text{m}, 6$

The resulting viscous stress components can be seen in figure 105 for the purely viscous case and in figure 106 for the viscoelastic case.

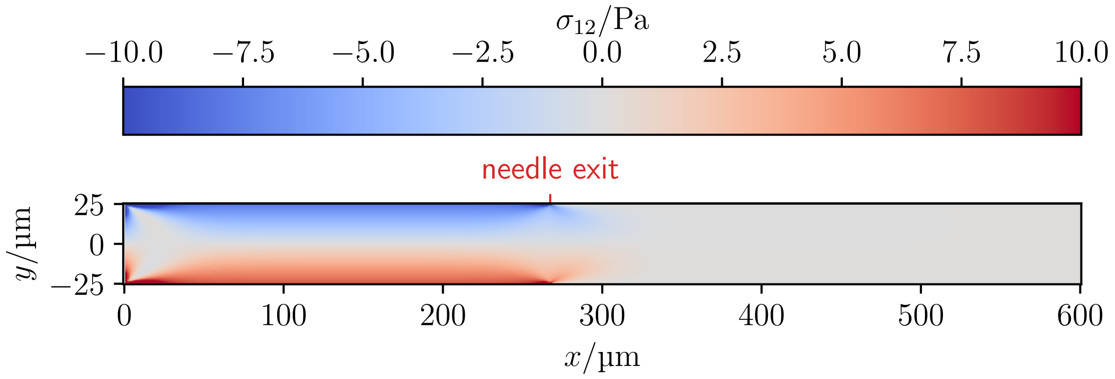


Figure 105: Plot of the cross-section of the viscous stress field around the exit of the printer needle. Here, the fluid is a purely viscous shear-thinning fluid.

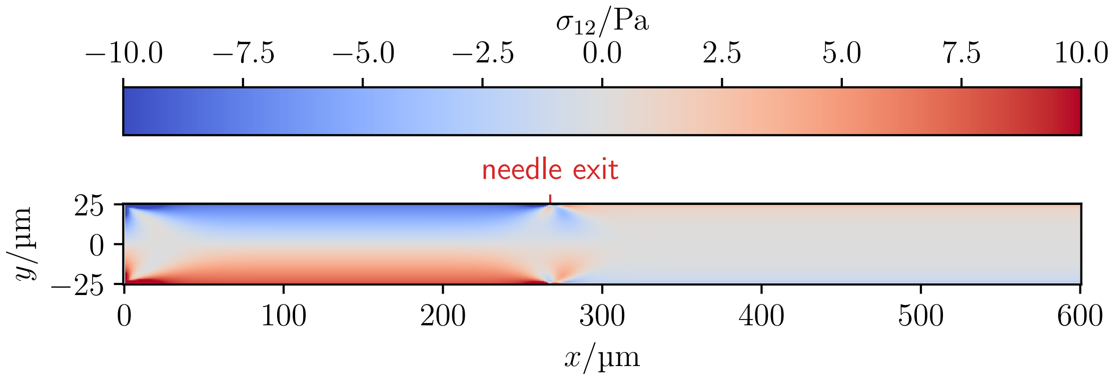


Figure 106: Plot of the cross-section of the viscous stress field around the exit of the printer needle. Here, the fluid is a viscoelastic fluid.

The initial region, where the stress establishes itself is similar. The part inside the needle shows near identical stresses, as it should by design. A remarkable difference at the exit is the region of low stress for the viscoelastic fluid. Shortly before the needle exit, the stress drops to near 0. At the walls, the stress even switches sign. This allows the viscoelastic case to disperse and switch to plug flow quicker than the purely viscous case. This can be seen in figures 107 and 108.

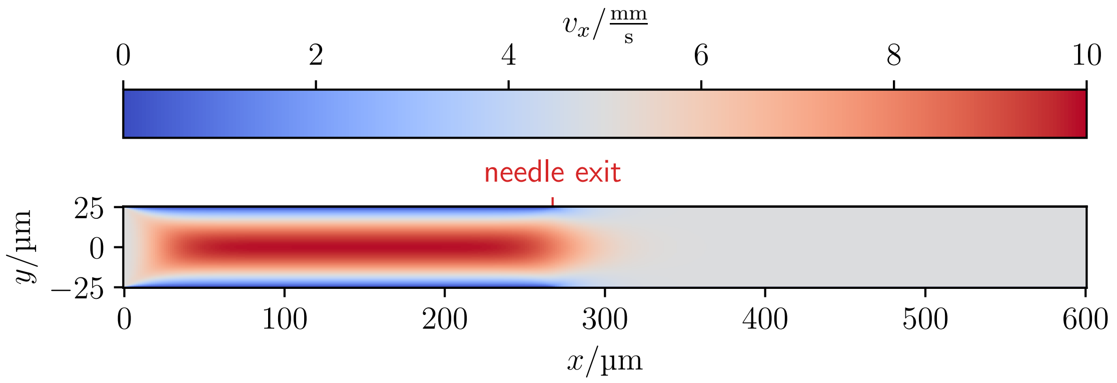


Figure 107: Plot of the cross-section of the velocity field in x-direction around the exit of the printer needle. Here, the fluid is a purely viscous shear-thinning fluid.

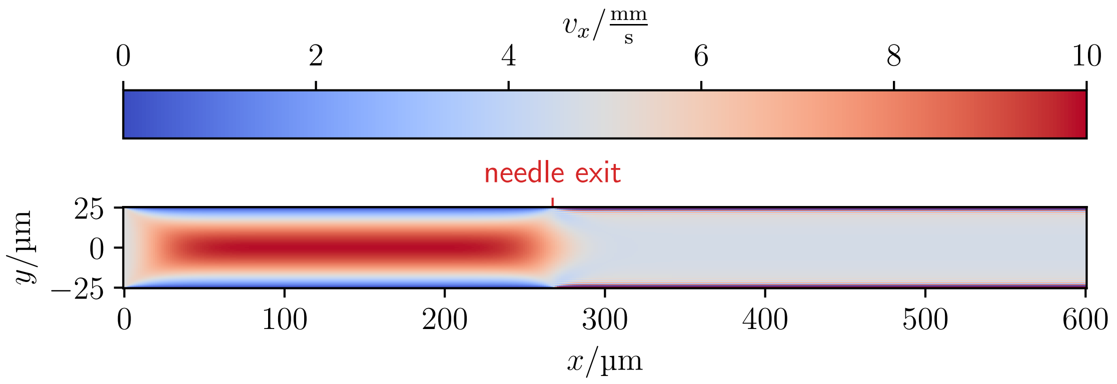


Figure 108: Plot of the cross-section of the velocity field in x-direction around the exit of the printer needle. Here, the fluid is a viscoelastic fluid.

Clearly, the x-component fans out wider in the viscoelastic case. It also reaches the plug flow configuration quicker. There are notably some artefacts around the slip walls in the viscoelastic case. This is due to the polymer-conformation tensor not slipping properly. Given this region is small and far from the cell, it does not matter much here. The same fanning effect can be seen in figures 109 and 110.

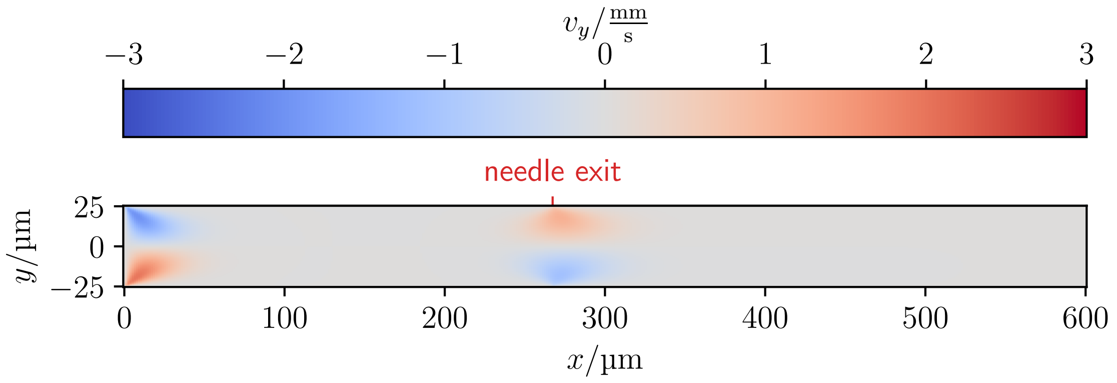


Figure 109: Plot of the cross-section of the velocity field in radial direction around the exit of the printer needle. Here, the fluid is a purely viscous shear-thinning fluid.

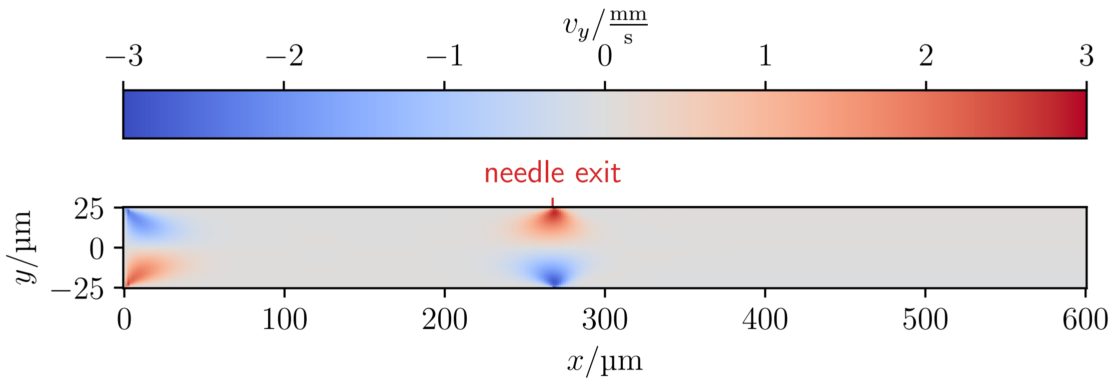


Figure 110: Plot of the cross-section of the velocity field in radial direction around the exit of the printer needle. Here, the fluid is a viscoelastic fluid.

The y component of the velocity is twice as large for the viscoelastic case near the exit of the needle. This is already in stark contrast to the expected behavior. The retardation in combination with the elastic contribution does not preserve the flow for an extended period after the needle exit. It does in fact do the opposite and cause faster adaptation to the plug flow through stronger radial velocity components. Dividing the shear stress by twice the strain-rate tensor yields a viscosity. As mentioned before, this is not really the viscosity. This interpretation of the stress breaks down for viscoelastic fluids. In many areas odd or even unphysical results are produced from this division. However, in the bulk materials away from corners and the center-line, this value can be interpreted as a viscosity. It can be seen in figure 111.

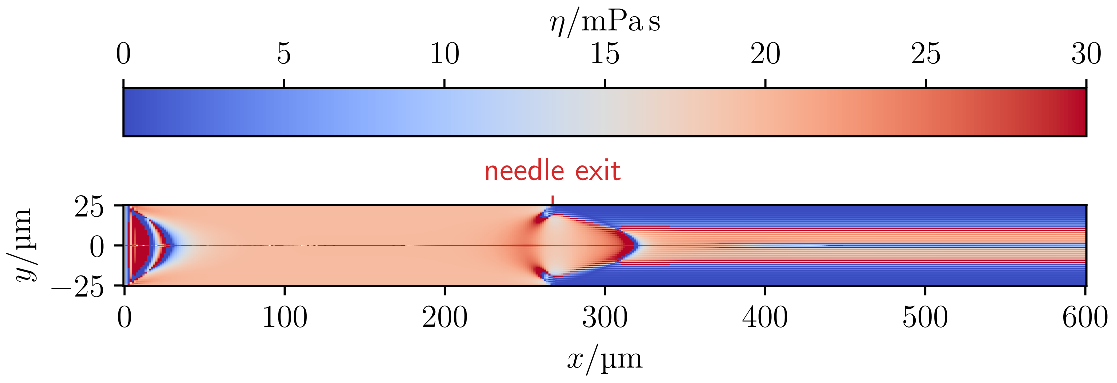


Figure 111: Plot of the cross-section of the viscosity field around the exit of the printer needle. Here, the fluid is a viscoelastic fluid.

The viscosity in the plug flow channel is hard to pin down, as the stress and the strain-rate tensor should both tend to $\underline{0}$. Still, it is quite clear, that the stress from inside the needle persists after the needle exit. At the exit, the viscosity has a local minimum. Shortly before the exit, there is a local maximum of the viscosity. Given that this is where the diverging flow starts, the cell is strongly stretched. This can be seen in figure 112.

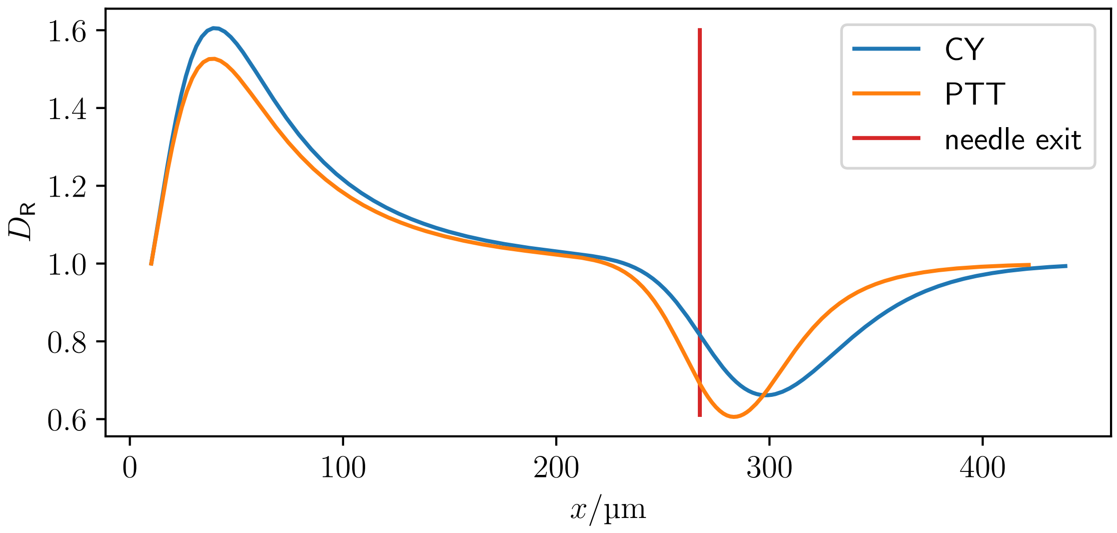


Figure 112: Plot of the deformation defined from the aspect ratio (see section 11) as a function of the position of the cell center. Both the shear-thinning CY fluid and the viscoelastic PTT fluid are shown.

The initial peak is due to startup effects. Notably, the cell gets radially stretched even before the needle exit. In the viscoelastic case the stretching is stronger than in the purely elastic case. However, it also does not persist as long. If this is in total better or worse for cell survivability is hard to tell. However, the elastic contributions one gets due to the typical manufacturing process of a shear-thinning fluid do not aid in reducing the stress at the needle exit. This is unexpected and might be important depending on if cell survivability depends more on the strength or the duration of the deformation. Next cell deformation is considered in a wider parameter range.

23. Viscoelastic lookup tables for RT-DC

A lookup table (LUT) in this context is used to invert the relationship of the deformation D and the Young's modulus E. An experiment, for example an RT-DC experiment (see section 24) produces the shape of a deformed cell. Such an experiment can be used for characterizing the cell. However, describing how a cell deforms under certain conditions is not very expressive. Ideally one would desire to the extract the fundamental parameters of the cell, like the Young's modulus from the deformation. This is what lookup tables are used for. First a reasonable parameter space gets covered with simulations. Given a Young's modulus and a set of conditions, these return a deformation. If one does now see a given deformation in an experiment with known conditions, one can use the tables to infer a Young's modulus. A major problem is the reasonable parameter space and the scalar nature of the deformation. A cell has a great many parameters, not all of them can reasonably be considered. Consequently, many of them are guessed. The base idea of the lookup table only works if the deformation from Young's modulus function is injective. This is not necessarily the case given, that simplification of the deformation to a scalar represents a significant loss of information. Here, multiple deformation measures are considered to explore this issue. In this section a LUT equivalent to Fig. 3b. of reference [98] is produced. Instead of requiring approximations, the present work can run fully viscoelastic simulations. This section also is used to explore the effects of viscoelastic stresses on a cell in Poiseuille flow. Given the comparison to literature, the channel here is square. However, the flow near the center of a square channel and the flow in a circular channel are very similar. Consequently, the results form this Poiseuille-ish flow can be transferred to other similar geometries. In this section, three sets of simulations are performed. The channel is considered with a Newtonian fluid, a CY fluid and a PTT fluid. These are explored in this order in the following sections. First a few common basics are introduced.

23.1. Common geometry

The desired channel is a square channel with a cross-section of $20\,\mu\mathrm{m} \times 20\,\mu\mathrm{m}$. The volume flow in the channel is $Q = 0.04\,\frac{\mu\mathrm{L}}{\mathrm{s}}$. Both area and Young's modulus are sampled with reasonable resolution. For the Young's modulus, this is straight forward. The values range from 0.5 kPa to 6 kPa in accordance to the paper. The area requires extra though. First, the deformation will change the area. This means, the sampled area, and the area in the final LUT will not be the same. The deformation of larger cells is bigger, because the velocity difference between their center and their edge is larger. This can be seen later, but the effect is not large enough to cause issues. Secondly, increasing cell size by increasing the resolution of the cell file would be cost prohibitive. Instead, the box is shrunk and the size of the lattice constant in SI is altered. A cell mesh with a resolution of $R = 16.\overline{6}$ is used for all these simulations. It is scaled by a factor of 2.4 to a size of 40 in LU, in order to prevent the jagged cells instability (see section 15.2). First, the target size of the cell A_{target} is determined from the size index i_{s} as follows.

$$A_{\text{target}} = (i_s + 1)25\,\mu\text{m} \tag{360}$$

From here, the target radius R_{target} is determined as follows.

$$R_{\text{target}} = \sqrt{\frac{A_{\text{target}}}{\pi}} \tag{361}$$

With this, the conversion factor to LU can be defined in two different ways as follows.

$$\frac{R_{\text{target}}}{40} = \frac{20\,\mu\text{m}}{B_{\text{target}}} \tag{362}$$

This defines the target size of the channel B_{target} in LU. This has to be an integer. Consequently, the value gets rounded to B. This rounded value together with the fixed channel size of 20 µm defines the actual conversion value. This requires a slight adjustment to the radius of the cell in LU. The scaling factor absorbs this rounding adjustment. The simulation volume has the dimensions of $10B \times B+2 \times B+2$ lattice nodes. This method is common to the following three simulations. The Newtonian flow is discussed next.

23.2. Newtonian LUT

For the Newtonian case simulation 23 is performed.

23 Newtonian LUT

Box:	variable
Fluid:	Outside: Newtonian $\eta_s = 6\,\text{mPa s}$
	Inside: Newtonian::water (fluid 1)
BC:	Inflow: mass-flow (BC 3)
	$Q = 0.04\,\frac{\mu\text{L}}{\text{s}}$
	Outflow: pressure (BC 4)
Cell:	variable
	Native resolution $16.\overline{6}$

The outside fluid has a viscosity of $\eta_s = 6\,\text{mPa s}$ to conform to reference [98]. The inflow is an analytically found Poiseuille profile (see appendix F.1.1). Two different deformation measures are used to analyze the data. The result for the deformation from circularity (see section 11.6) can be seen in figure 113.

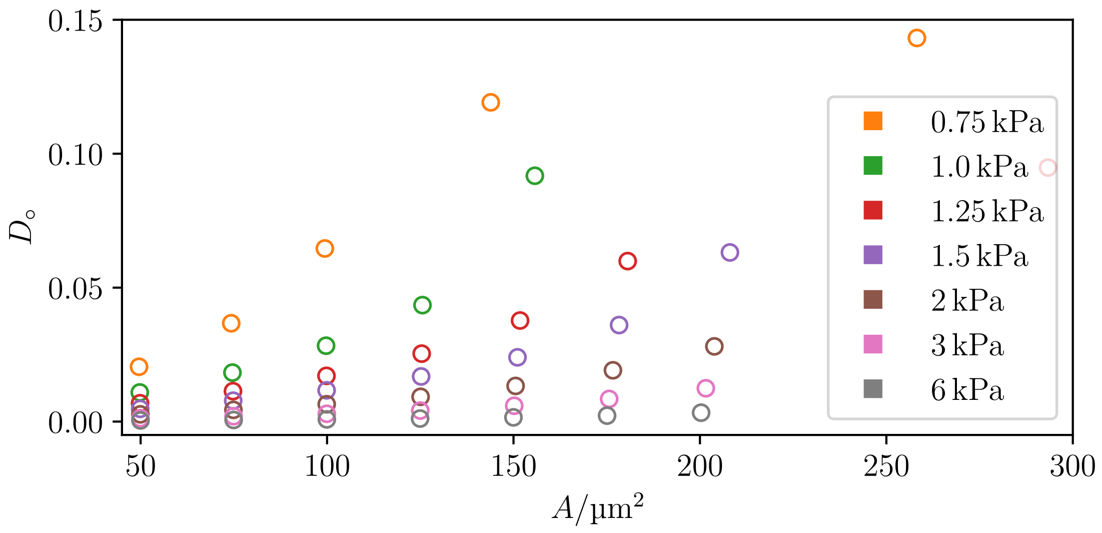


Figure 113: Plot of the deformation calculated from the circularity as a function of the projected area. The cell is suspended in a Newtonian fluid. Different Young's moduli are shown.

The curve for $E = 0.5\,\mathrm{kPa}$ is omitted, because the softer cells show some artefacts. These are so bad in this case, that the results are not usable. One can see, that the areas get progressively larger compared to the target size for increasing deformation. As expected, softer cells show larger deformations. The deformation grows monotonically with the area. Not much can be learned from this. The same simulation output can be analyzed using the deformation from the aspect ratio (see section 11.4). This can be seen in figure 114.

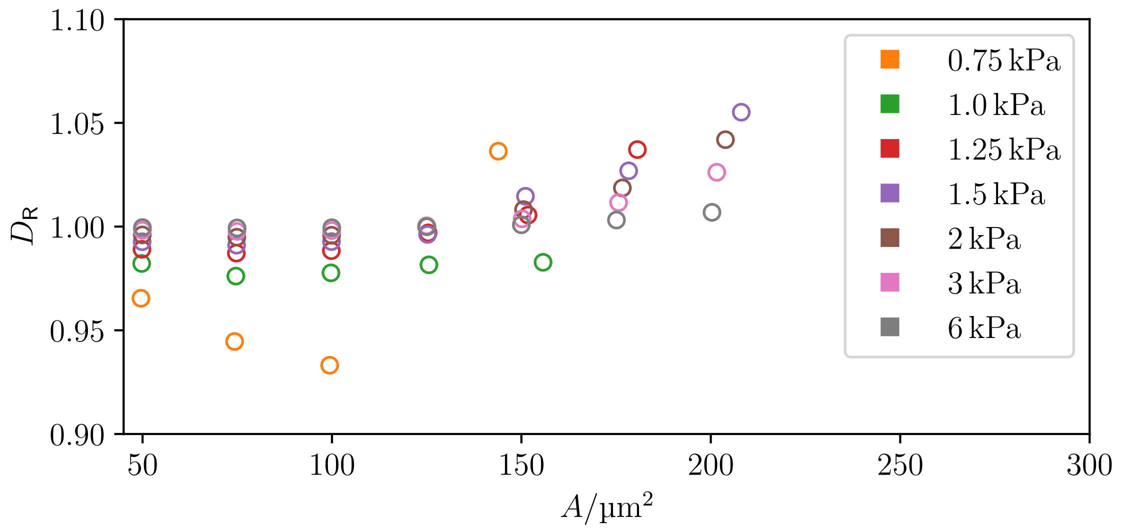


Figure 114: Plot of the deformation calculated from the aspect ratio as a function of the projected area. The cell is suspended in a Newtonian fluid. Different Young's moduli are shown.

Once again, the curve for 0.5 kPa is omitted, because the softer cells show some artefacts. Also, the curve for $E = 0.75\,\text{kPa}$ is only partially present and somewhat questionable. For small areas, the deformations are stacked from the lowest Young's modulus to the highest. For large areas, this trend reverses. An explanation can be made using the diagrams of section 12.1. For the small cells, the force acting in radial direction dominates. This causes the radial extent to increase relative to the extent in x-direction. Consequently, the deformation drops below unity. The effect is stronger for softer cells. For larger cells, the force in x-direction dominates, elongating the cell. The deformation rises above unity. The effect is stronger for softer cells. This shows, that along the size axis, the cell shape changes conceptually around $A = 140\,\mu\text{m}^2$. This is due to the relative relevance of the involved forces changing. It is important to note, that this is not visible for the deformation measure defined from the circularity. On the other hand, the deformation measure defined from the aspect ratio yields the same deformation for vastly different parameters at around $A = 140\,\mu\text{m}^2$. This defeats the purpose of a lookup table. One can clearly see, that the choice of deformation measure is important here. One might want to consider multiple deformation metrics to capture the behavior of the cells more clearly. Next a shear-thinning CY and viscoelastic PTT fluid is considered to show the difference viscoelasticity makes.

23.3. CY vs PTT LUT

Here a CY fluid and a PTT fluid, with the same viscosity as the CY fluid will be used to highlight the effect of the normal stress. There are no analytical or semi-analytical solutions for the flow profile in a square channel with these fluids. Therefore, simulations 61 and 62 are performed to generate boundary conditions, which are approximately equal to $Q \approx 0.04\,\frac{\mu\text{L}}{\text{s}}$. The concrete value is $Q \approx 0.040\,01\,\frac{\mu\text{L}}{\text{s}}$. These simulations provide a velocity field as a boundary condition for simulation 24 and velocity and polymer-conformation tensor boundary conditions for simulation 25.

24 CY LUT

Box:	variable
Fluid:	Outside: CY::mc0_59 (fluid 9)
	Inside: Newtonian::water (fluid 1)
BC:	Inflow: mass-flow (BC 3)
	simulation 61
	$Q \approx 0.04\,\frac{\mu\text{L}}{\text{s}}$
	Outflow: pressure (BC 4)
Cell:	variable
	Native resolution $16.\overline{6}$

25 PTT LUT

Box:	variable
Fluid:	Outside: PTT::mc0_59 (fluid 3)
	Inside: Newtonian::water (fluid 1)
BC:	Inflow: mass-flow (BC 3)
	simulation 62
	$Q \approx 0.04\,\frac{\mu L}{s}$
	Outflow: pressure (BC 4)
Cell:	variable
	Native resolution $16.\overline{6}$

The data is displayed in two different plots in order to improve visibility. The deformation defined from circularity can be seen in figures 115 and 116.

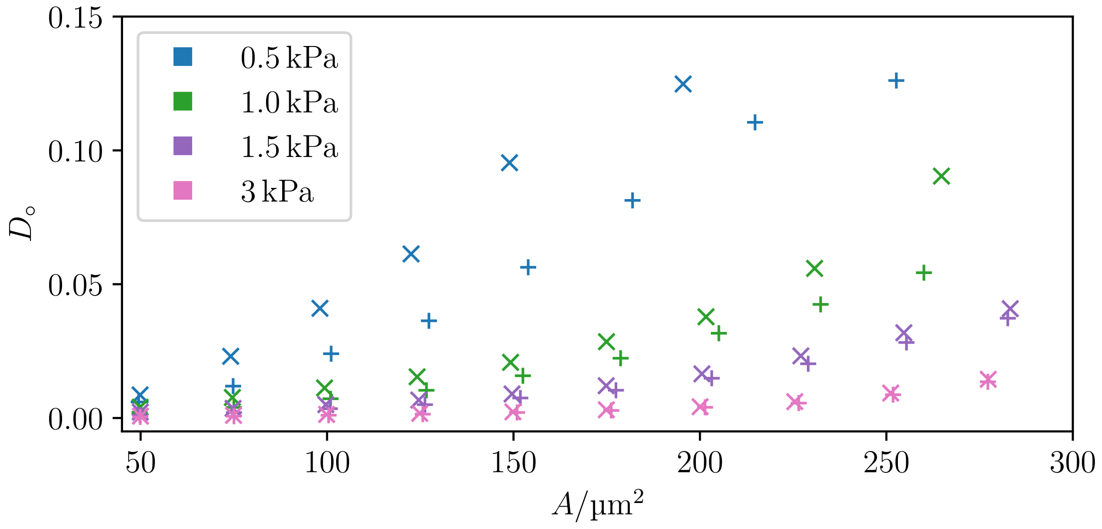


Figure 115: Plot of the deformation calculated from the circularity as a function of the projected area. The cell is suspended in a CY (×) or PTT (+) fluid. Pictured are half of the data points for improved visibility.

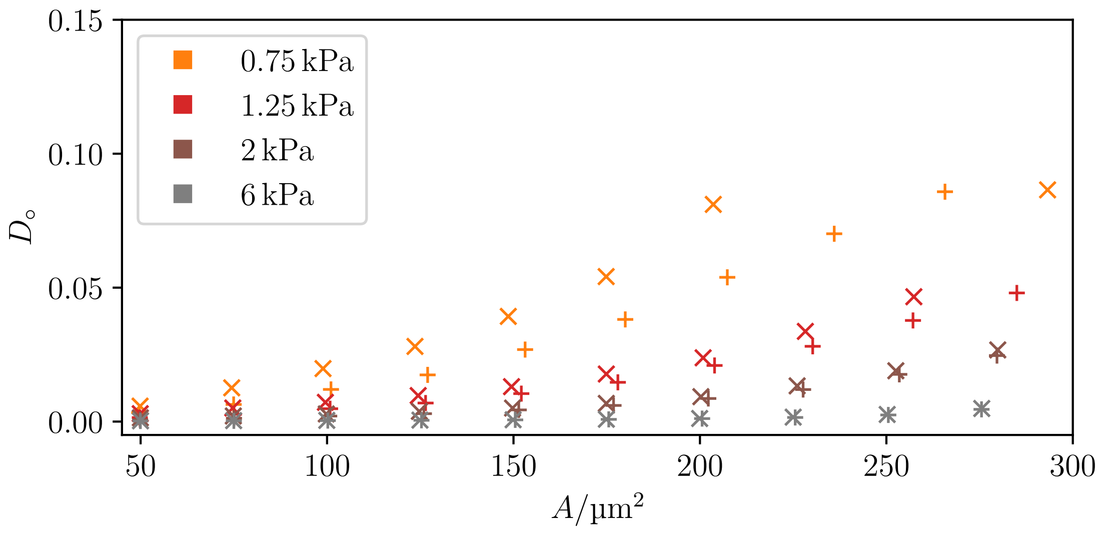


Figure 116: Plot of the deformation calculated from the circularity as a function of the projected area. The cell is suspended in a CY (×) or PTT (+) fluid. Pictured are half of the data points for improved visibility.

The cells have a higher deformation for the CY case. Meaning, the additional elastic force of the PTT fluid decrees the measured deformation. For Young's modli $E \geq 1.25$ kPa, the difference vanishes. Even for smaller Young's moduli, the difference is hard to impossible to spot for smaller cells. Considering the large errors, that are associated with this deformation (see section 11.6), the difference might not be visible in experiments at all. This does however not mean, that there is no relevant deformation. This can be seen in figures 117 and 118.

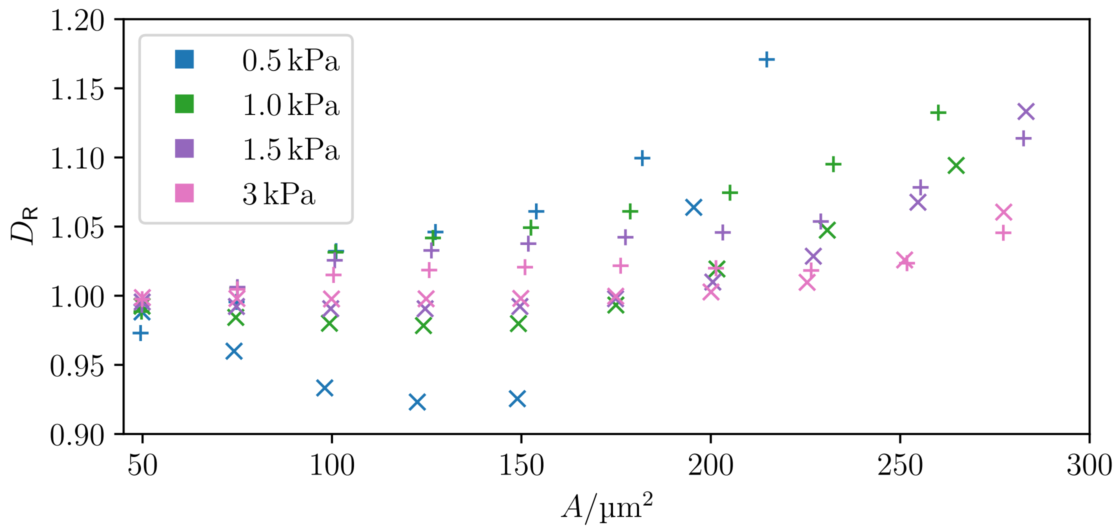


Figure 117: Plot of the deformation calculated from the aspect ratio as a function of the projected area. The cell is suspended in a CY (×) or PTT (+) fluid. Pictured are half of the data points for improved visibility.

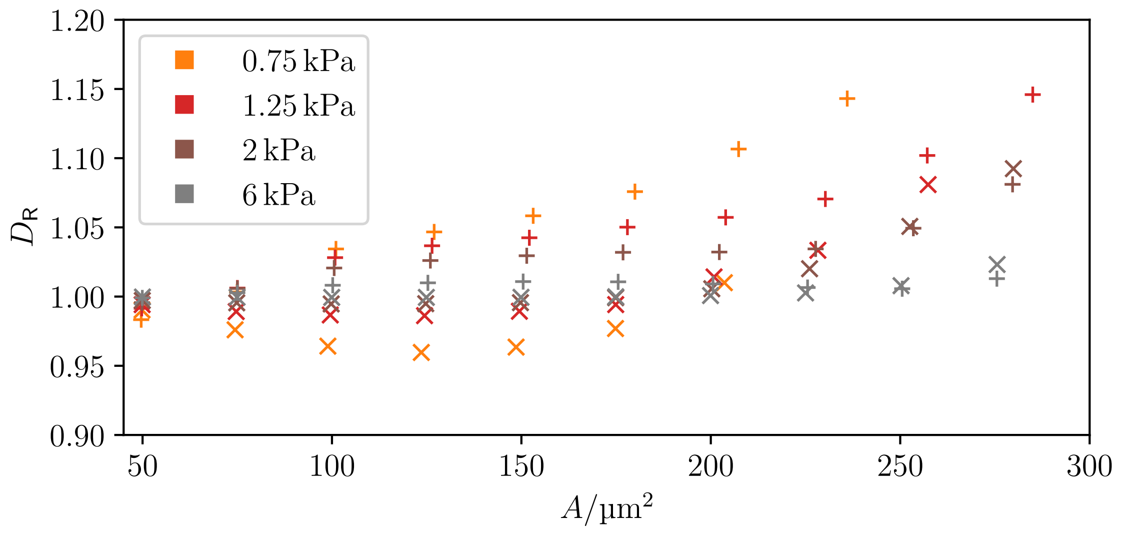


Figure 118: Plot of the deformation calculated from the aspect ratio as a function of the projected area. The cell is suspended in a CY ($\times$) or PTT ($+$) fluid. Pictured are half of the data points for improved visibility.

As with the Newtonian case, the purely viscous fluid causes the deformation to drop below unity for small cells and rise above it for larger cells. The explanation is the same as for the Newtonian case. The CY fluid produces the same forces in the same directions. They do differ in their amplitude, which does locally vary for both cases. However, this is slightly different for a shear-thinning fluid. Consequently, the absolute value of the deformation is different, but the general trends match. The PTT curves are completely different. They start and stay above unity. An explanation can be made using the diagrams of section 12.1. The elastic force elongates the cell for any size of cell. Even for the small cell, the elastic force is strong enough to overpower the slight widening of the cell due to the viscous forces. With this deformation metric, it is particularly easy to see the elastic effects for small cells. Either the deformation is above unity for the viscoelastic fluid or below for the purely viscous fluid. Even the somewhat hard cells with $E = 3\,\text{kPa}$ still show a clear distinction. With this deformation metric, it is the large cells, where the viscoelastic and the purely viscous fluid produce the same deformation. This neatly compliments the deformation from circularity. This leads to some concluding remarks.

23.4. Use of these lookup tables

The viscoelastic equivalent for the LUT seen in Fig. 3b. of reference [98] is provided by figures 115 and 116. However, this section clearly shows, that using a single deformation metric does not capture all relevant behavior. Using both metrics employed here demonstrates the ability to distinguish the deformation caused by viscoelastic and purely viscous fluids in the entire parameter space. Consequently, it should be encouraged to use more than one deformation metric. As a caveat, one should consider, that this section does make assumption, like the inside of the cells having a viscosity equal to water. These cannot be proven and do have an influence on the results. The more deformation metrics one uses the greater the chance to catch an unknown influence. Calculating more metrics is trivial. With this though, an explanation for an effect yet unreproduced in simulations is found in the next section.

24. Real-Time Deformability-Cytometry

Real-Time Deformability-Cytometry (RT-DC) is a new method developed to extract cell properties. This is done by sending the cells through a narrow channel on a microfluidic chip. The geometry of the relevant part of the chip can be seen in appendix P.3. This shape causes a deformation of the cell. This deformation can be used with lookup tables (see section 23) to determine apparent Young's moduli. The great advantage of this technique compared to more established techniques like AFM is the throughput. The fact, that cells are analyzed in a continues stream of cell-carrier fluid, without the requirement to position them by hand greatly increases the speed of data acquisition. RT-DC can theoretically do hundreds of cells per second while AFM takes several minutes per cell. As biological cells vary vastly in their parameters, a statistically significant amount needs to be analyzed to make any kind of statement. Hundreds of cells are required for decent statistics [100]. A detailed description of the method can be found in references [108, 111, 112]. For the purpose of this thesis, it is enough to know, that a channel of the shape described in appendix P.3 is driven to a flow of $Q \approx 0.04\,\frac{\mu\mathrm{L}}{\mathrm{s}}$. This causes a cell traveling through the channel to be deformed. This deformation is analyzed in literature in order to find the apparent Young's modulus of the cell. In this section, the opposite is done. The cell parameters are varied in order to find a deformation, which matches the experiment. This is necessary, as current simulation implementations have difficulty reproducing the observed deformation. This is discussed first.

24.1. Viscous simulations

As discussed in section 23, looking up a single deformation measurement in a lookup table, will yield a result, but not necessarily an accurate one. One way to combat this is to track the deformation over time or rather the position in the channel. This obviously shows more of the cell behavior. Furthermore, the found apparent Young's modulus must be constant across the channel. This might rule out some combination of parameters as they might fit in one spot but not another. The issue with this idea is, that current simulations cannot reproduce the deformation seen in experiments. Experimental data from Reichel [113] and simulation results from Wittwer *et al.* [114] can be seen in figure 119.

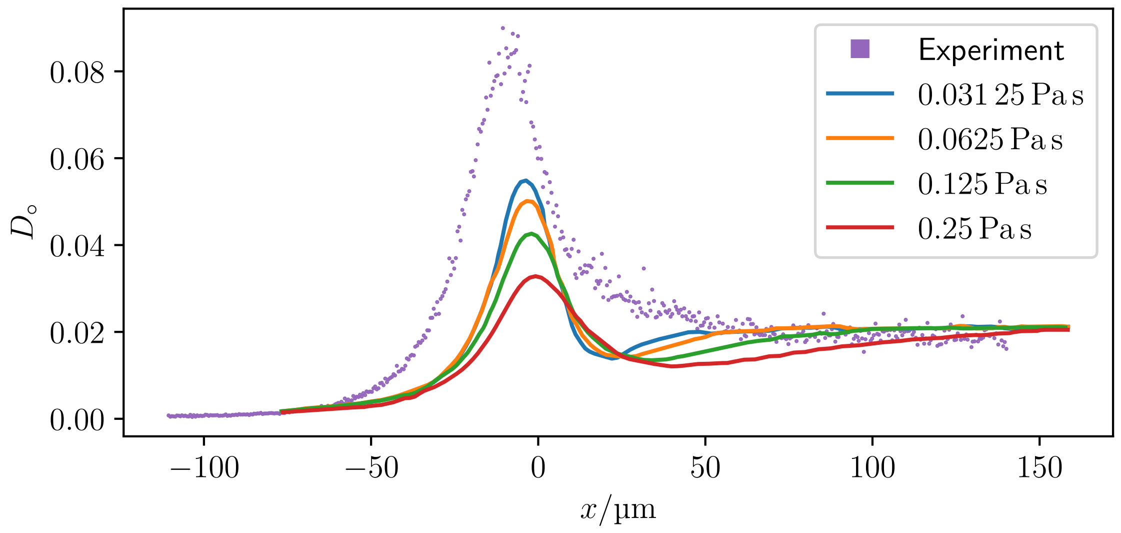


Figure 119: Plot of the deformation defined from circularity as a function of position in the channel. The 0 corresponds to the start of the narrow section. Curves for experimental data and several viscous simulations with varying cell viscosity are shown.

The simulation data is a bit rough, because it is extracted from a plot. The large error-bars on the experimental data have been omitted for visibility. A Young's modulus of $E = 1.5\,\mathrm{kPa}$ has been used for these simulations. Clearly, the major peak does not match. Decreasing the Young's modulus increases the deformation in the whole parameter space [114]. Meaning, the plot dos not match towards the end. Decreasing the cell viscosity could also be used to increase the maximum peak. However, this leads to the local minimum after the maximum peak increasing. This minimum is not present in the experimental data. Altering the parameters in this purely viscous simulations cannot explain the experimental data. The fluid used in this experiment is viscoelastic. How taking this into account in simulations is shown next.

24.2. Viscoelastic simulations

The viscoelatic simulations are performed in three steps. First, simulation 63 is performed to determine a mass-flow boundary condition for the inflow. A pressure gradient is varied until the desired volume flow of $Q \approx 0.04\,\frac{\mu L}{s}$ is archived. The result provides the boundary condition for simulation 64. This simulation has the entire channel but not the cell. It is run until it reaches equilibrium, which takes quite some time. The result can be loaded into a new simulation as a sort of checkpoint. The parameter study is only in regard to cell parameter. This means the fluid field is initially identical for all simulations. Loading it avoids waiting for its equilibration for each simulation. Finally, simulation 26 is performed with a wide range of Young's moduli and cell viscosities.

26 RT-DC parameter study

Box:	$1350 \times 146 \times 50$
	$L_0 = 417\,\mathrm{nm}$
Fluid:	Outside: PTT::mc0_59 (fluid 3)
	Inside: Newtonian variable
BC:	Inflow: mass-flow (BC 3)
	simulation 63
	$Q \approx 0.04\,\frac{\mu\mathrm{L}}{\mathrm{s}}$
	simulation 64
	Outflow: pressure (BC 4)
Cell:	Young's modulus variable
	Radius, Native resolution $R \approx 6.8\,\mu\mathrm{m}, 6$

Instead of the deformation defined form circularity, the length l and height h of the cell are used. Length here is the extent of the cell along the x-axis. Height refers to the extent of the cell along the y-axis. This is part of the raw experimental data provided directly by Felix Reichel. The equilibrium cell diameter $2R$ is subtracted from both to only show the deformation. This is shown in the following.

$$l' = l - 2R \tag{363}$$

$$h' = 2R - h \tag{364}$$

Note, that the sign of h' was switched to make both quantities positive. This helps when comparing them. These values are calculated for the experimental data and the simulations. A comparison can be seen in figure 120.

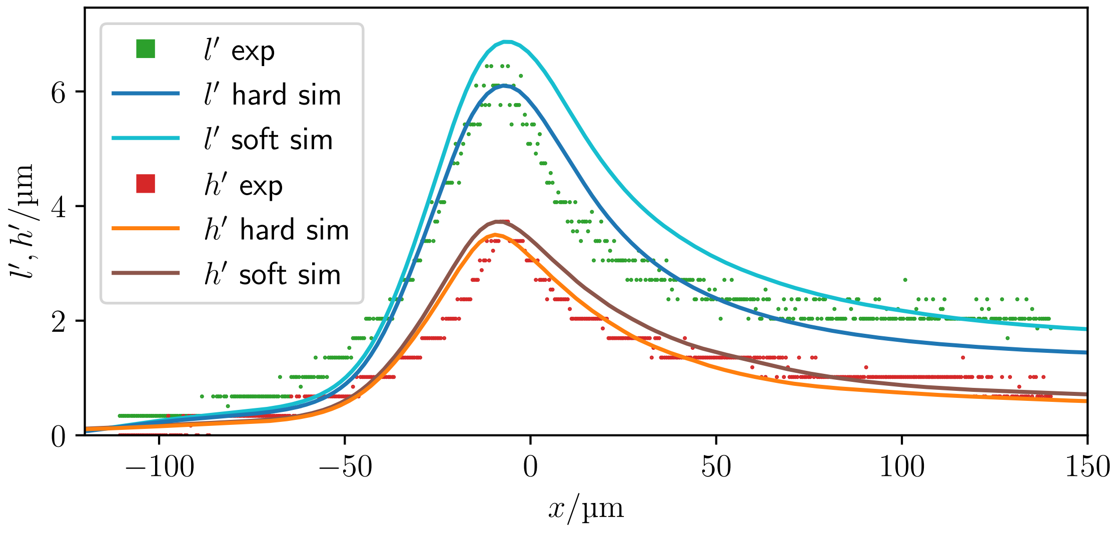


Figure 120: Plot of the deviation from the with and length as a function of the position of the cell center. Data for two different simulations is shown.

Here $x = 0$ marks the start of the narrow portion of the RT-DC channel. Note, that the data was given a small x offset. This is to combat a simulation artefact. Namely, the narrowing of the channel only has an effect on the center once its influence has permeated throughout the fluid. This speed is limited by the Lattice Boltzmann method, causing the simulation curve to lag behind the experimental curve by approximately 25 µm. This shortcoming has been handled by simply shifting the signal and hoping this does not matter. Two simulation outputs have been included to convey the resolution of the parameter space probed by the simulation run. The one termed "hard sim" does fit a bit better. However, interpolating somewhat towards "soft sim" would improve the results. Consequently, the correct parameters likely lie between the two and closer to "hard sim". It should be noted, that when considering the discretization, that these values show, the simulations are not far off. They hardly ever further than one step from the experimental data. This error stems from the fact, that a camera with a sufficiently high frame-rate to cover these experiments is typically severely limited in resolution. The size of a pixel then dictates the error. The internal viscosity for both curves is $\eta_{\text{in}} \approx 68.8\,\text{mPa s}$. The soft curve has a Young's modulus of $E \approx 588\,\text{Pa}$, while the hard curve has a Young's modulus of $E \approx 681\,\text{Pa}$. Clearly, there is plenty of space for optimization between these parameters. Notably, this viscoisity as at the lower end of the viscosities simulated by Wittwer *et al.* [114]. The Youngs's modulus is less than half of the one used for figure 119. The deformation defined from circularity can be seen for these two simulations and the experimental data in figure 121.

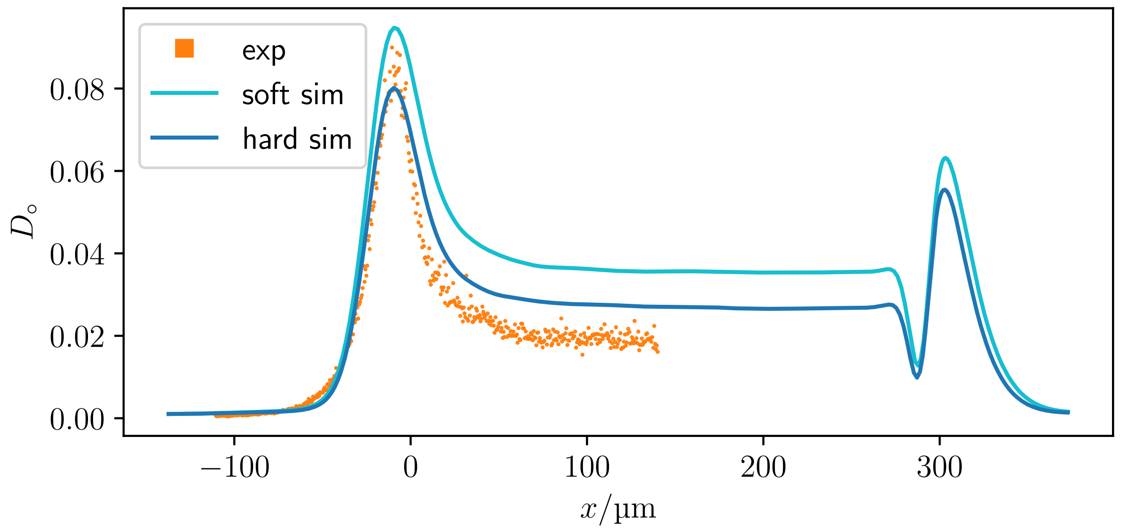


Figure 121: Plot of the deformation defined form circularity as a function of the position of the cell center. Data for two different simulations is shown.

Clearly, the local minimum, that is conflicting with experimental observations is not present here. The hard simulation does fit a bit better. Towards the end of the curve, there is a considerable amount of discrepancy between the experimental curve and the simulation curve. However, as has been mentioned before, this deformation metric carries large errors. Taking these into account, the hard simulation is well within the range of the errors. There are no images for the data discussed here. Here, the channel is filled with PTT::alginate (fluid 13). The walls are hard to see, because the video was processed for easier automatic evaluation. This video was captured by Steffen Trippmacher. The red circle represents a simulation. This is already 2.5 a old, and consequently did not work as well. Still, the agreement was already acceptable. This concludes the discussion on RT-DC channels. Next another type of channel used for characterization will be discussed.

25. Hyperbolic(ish) Channel

Channels, which produce a constant extensional rate are useful for cell deformability research. Such channels can isolate the effect of the extensional rate on the cell and provide insight into the inner workings of cells. Given that a constant extensional rate means a steadily increasing velocity, such experiments are not trivial. Reichel *et al.* [99] attempted the construction of such a channel. The equations describing the flow in such a channel cannot be solved analytically. Consequently, no exact analytical solution can be given for the channel geometry. Reichel *et al.*approximated the equations to acquire an analytical solution. The resulting channels are of constant depth and narrow according to a hyperbolic function. The width of such a channel can be seen in figure 122.

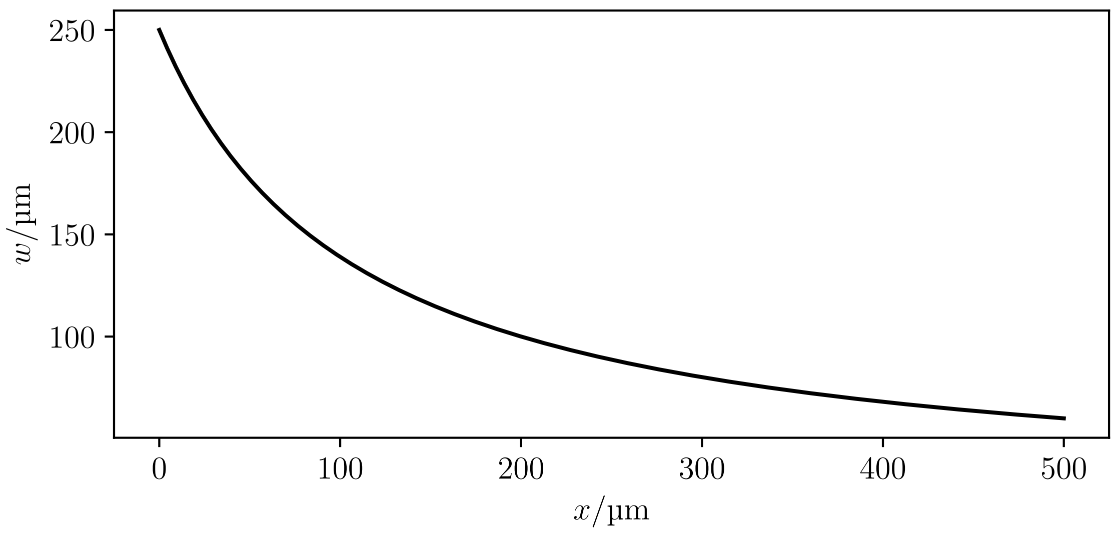


Figure 122: Channel width from Reichel *et al.* [99].

This geometry is still in use as it is the best geometry available for this task. However, the hyperbolic channels do not produce the desired constant extensional rate. This can be seen in Fig. 1 D of reference [99] and in figure 123.

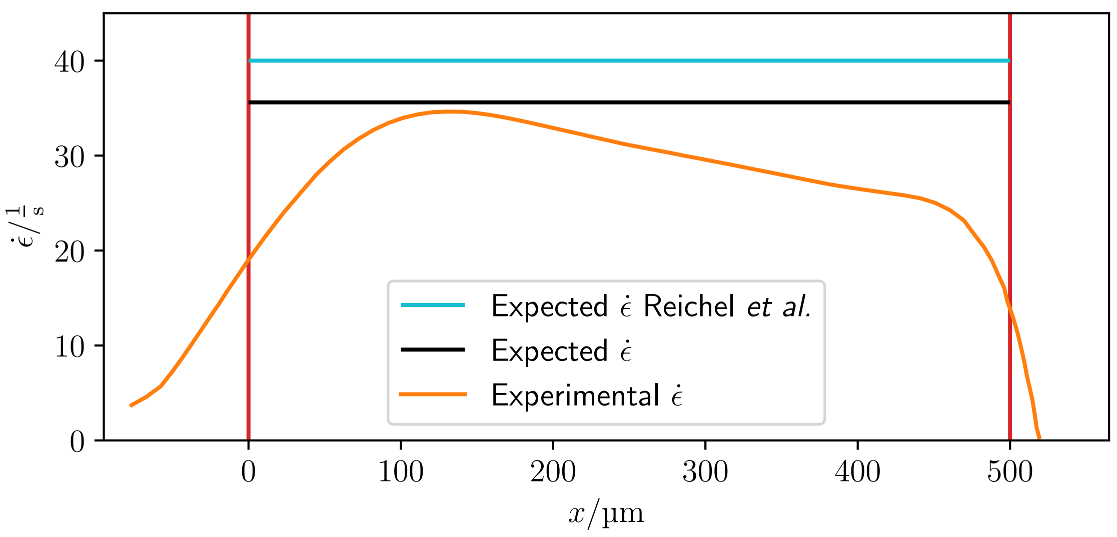


Figure 123: The experimentally determined extensional rate in the channel with the faulty geometry. The varying width channel is between the red lines. The extensional rate expected by Reichel *et al.* [99] and a more sensible expected value (Newtonian fluid in a properly shaped channel) are shown for reference.

The hyperbolic channels actually deployed in research exhibit poor stability of the extensional rate. The extensional rate in the experiment reaches an average of $28.8\,\frac{1}{s}$ between 225 µm and 425 µm, where the extensional rate is most constant. This is approximately 19 % below the value, which should have been expected and therefore in poor agreement. In part, this discrepancy is due to errors in the aforementioned publication (see appendix Z). However, a lot more factors are to be considered for good results. This section explores how such channels would need to be manufactured in order to properly exhibit constant extensional rates. This includes the finding, that such channels are not hyperbolic (even though they are close) and the discussion of the impact of the chosen fluid. The channels developed here have the same length, inlet and outlet size as the ones presented by Reichel *et al.* [99], so that they can be used as a drop-in replacement. Instead of approximation, this thesis opts for numeric solutions. Before these are discussed, first a small elaboration on the reproducibility of the existing experiments using simulations is provided.

25.1. Reproduction of experiments

It should be pointed out, that using the appropriate CY (Carreau-Yasuda fluid model) fluid with the channel developed by Reichel *et al.* [99] (simulation 27) does not fully reproduce the experimental results as can be seen in figure 124.

27 Methyl cellulose solution in the incorrect channel

Box:	$2002 \times 602 \times 74$
	$L_0 \approx 417\,\mathrm{nm}$
Fluid:	CY::mc0_59_D (fluid 6)
BC:	mass-flow (BC 3)
	$Q = 0.02\,\frac{\mu\mathrm{L}}{\mathrm{s}}$
	pressure (BC 4)

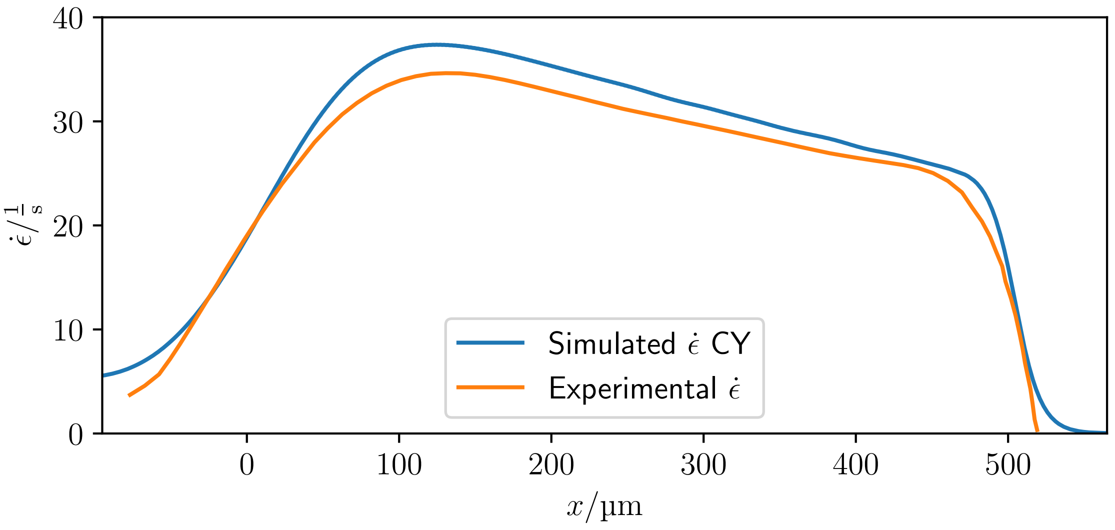


Figure 124: The simulated extensional rate for a CY fluid (CY::mc0_59_D fluid 6) in the channel with the geometry by Reichel *et al.* [99] compared to the experimental results.

This is not a shortcoming of the simulation. The experiment does not probe the flow field in the very center as was done here. It does instead deal with a cell of finite size, which itself influences the flow field. A cell as described by Reichel *et al.* [99] is added in simulation 28.

28 Methyl cellulose solution wit a cell in the incorrect channel

Box:	$2002 \times 602 \times 74$
	$L_0 \approx 417\,\mathrm{nm}$
Fluid:	CY::mc0_59_D (fluid 6)
BC:	mass-flow (BC 3)
	$Q = 0.02\,\frac{\mu\mathrm{L}}{\mathrm{s}}$
	pressure (BC 4)
Cell:	Young's modulus $E = 370\,\mathrm{kPa}$
	Radius, Resolution $R = 7.5\,\mu\mathrm{m}, 18$

The extensional rate is determined from the movement of the cell center. The result can be seen in figure 125.

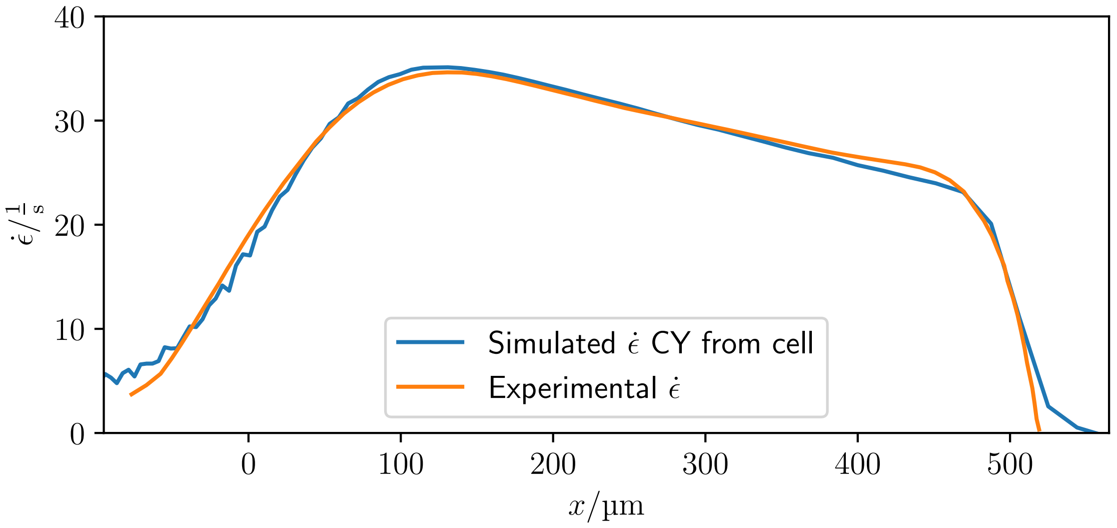


Figure 125: The simulated extensional rate determined from the cell position for a CY fluid (CY::mc0_59_D fluid 6) in the channel with the geometry by Reichel *et al.* [99] compared to the experimental results.

This calculation requires numerical differentiation of higher order. Consequently, it is a little noise around the edges. Otherwise, the agreement is great. This highlights how accurately the experiments are described by these simulations. This means, the simulations presented here can replace experiments and test the channel geometries without the need to manufacture them. However, an experiment can never control parameters as exactly as a simulation. Also, the cell is omitted for the following simulations. Therefore, experiments will not be able to precisely reproduce the simulation results presented in this section. In the following appropriate channels are generated for different fluids. The discussion shall start with the easiest derivation - the Newtonian fluid.

25.2. Newtonian fluids

Such a roughly hyperbolic channel can be viewed as many very short rectangular channels chained together. For every such infinitesimal rectangular channel, a Poiseuille flow can be assumed. This is often considered the solution for channels of infinite length, the actual mathematical condition however is the absence of radial velocity components. This condition is not strictly fulfilled here (particularly not near the inlet), but as the flow is predominantly in x direction, this is a valid approximation. Consider the velocity profile and volume flow rate in a rectangular channel [115]. The form here is modernized and follows Wikipedia [116] instead of the original from Boussinesq [115].

$$u_x(y,z) = \frac{G}{2\mu}y(h-y) - \frac{4Gh^2}{\mu\pi^3}\sum_{n=1}^{\infty}\frac{1}{(2n-1)^3}\frac{\sinh(\beta_n z)+\sinh[\beta_n(l-z)]}{\sinh(\beta_n l)}\sin(\beta_n y) \tag{365}$$

$$Q = \frac{Gh^3 l}{12\mu} - \frac{16Gh^4}{\mu\pi^5}\sum_{n=1}^{\infty}\frac{1}{(2n-1)^5}\frac{\cosh(\beta_n l)-1}{\sinh(\beta_n l)} \tag{366}$$

$$\beta_n = \frac{(2n-1)\pi}{h} \tag{367}$$

With the pressure gradient $G = -\frac{\mathrm{d}p}{\mathrm{d}x}$ and the dimensions w and l of the channel. This formulation defines y and z as $0 \le y \le h$ and $0 \le z \le l$, but it is often preferable to have the coordinates centered. This leads to the following equations (see appendix F.1.1).

$$u_x(y,z) = \frac{G}{2\mu}\left(\frac{h^2}{4}-y^2\right) - \frac{4Gh^2}{\mu\pi^3}\sum_{n=0}^{\infty}\frac{1}{(2n+1)^3}\frac{\cosh(\beta_n z)}{\cosh\left(\beta_n \frac{w}{2}\right)}\cos(\beta_n y)(-1)^n \tag{368}$$

$$Q = \frac{Gh^3 w}{12\mu} - \frac{16Gh^4}{\mu\pi^5}\sum_{n=0}^{\infty}\frac{1}{(2n+1)^5}\tanh\left(\beta_n\frac{w}{2}\right) \tag{369}$$

$$\beta_n = \frac{(2n+1)\pi}{h} \tag{370}$$

With the coordinates y and z now centered as $-\frac{h}{2} \le y \le \frac{h}{2}$ and $-\frac{w}{2} \le z \le \frac{w}{2}$. As it is desired to have the extensional rate be constant in the center, the center-line velocity u_0 is required. It reads as follows.

$$u_0 = \frac{Gh^2}{8\mu} - \frac{4Gh^2}{\mu\pi^3}\sum_{n=0}^{\infty}\frac{1}{(2n+1)^3}\frac{1}{\cosh\left(\beta_n\frac{w}{2}\right)}(-1)^n \tag{371}$$

Both the formulas for the volume flow rate and the center-line velocity are quite cumbersome and therefore approximations exist. Following these leads to major inaccuracies (see appendix Z). Therefore, it is clear, that such an approximation is not sensible and a semi-analytical solution should be used instead. The algorithm used for this is described here using its functions as follows.

```
def QByUFunc(h, w):
```

This calculates the fraction $\frac{Q}{u}$, which notably only depends on h and w as G and μ both cancel out. The sums run up to $n = 99$. Around $n = 2$ would probably be sufficient, but this computation is practically free, so going overboard is irrelevant.

```
def wFunc(QbyU, h):
```

This uses a root-finding algorithm (`scipy.optimize.root_scalar` [117]) on `QByUFunc` to find w for a given $\frac{Q}{u}$ and h. h is practically arbitrary, but this work sticks with $h = H = 30\,\mu\text{m}$ from Reichel *et al.* [99].

```
def findCurve(epsilonDot):
```

This determines w along the whole channel for a desired extensional rate $\dot{\epsilon} = \frac{\partial u}{\partial x}$ as follows. For a given volume flow rate (here $Q = 0.02\,\frac{\mu\text{L}}{\text{s}}$) calculate the center-line velocity at the inlet by dividing the volume flow rate by `QByUFunc(w_c, h)`. Here $w_\text{c} = 250\,\mu\text{m}$ is the desired inlet size. Note, that Reichel *et al.* [99] has inlet and outlet size switched in Fig. 1 (or the equations depending on perspective). With the inlet center-line velocity and an array of sufficiently highly resolved positions from 0 to the length of the channel $L_\text{c} = 500\,\mu\text{m}$, the desired center-line velocity at every point in the channel can be calculated. With Q and u_0 known, their fraction, together with h can be passed to `wFunc`, to determine the profile.

```
def findEpsilonDot():
```

If the size of the channel at the end is fixed due to setup related reasons (here $w_\text{u} = 60\,\mu\text{m}$), the root-finding algorithm can be used again on `findCurve` to find the $\dot{\epsilon}$, that is produced by a channel conforming to this restriction. Here, the result is $\dot{\epsilon} \approx 35.6\,\frac{1}{\text{s}}$. With this, the channel is fully specified. The resulting curve for the width can be seen in figure 126.

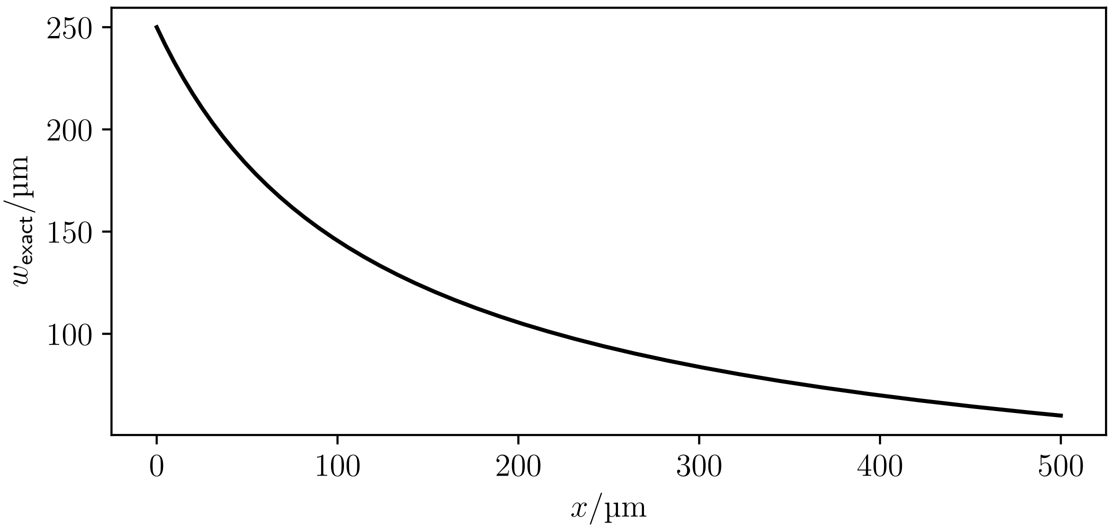


Figure 126: Channel width from the semi-analytical solution.

It shall be noted, that even so this looks rather hyperbolic, it is not quite. Approximating this curve using a hyperbolic function similar as the one suggested by Reichel *et al.* [99] is possible. The following hyperbolic function can be used to approximate the data.

$$w(x) = ah - \left[\frac{1}{L_\mathrm{c}}\left(\frac{1}{ah - w_\mathrm{u}} - \frac{1}{ah - w_\mathrm{c}}\right)x + \frac{1}{ah - w_\mathrm{c}}\right]^{-1} \tag{372}$$

Where a is a parameter, that can be used to optimize the shape. While the approximation is quite decent as can be seen in figure 127, the true shape is actually not hyperbolic. The true shape is closer to hyperbolic near the outlet than near the inlet. Larger a yield a better approximation near the inlet, while smaller values are better at the outlet. This is also the explanation, why Reichel *et al.* [99] only reached the value, they should have expected for $\dot{\epsilon}$ near the inlet, as the $a = 0.63$ resulting from their derivation is too large by a significant margin.

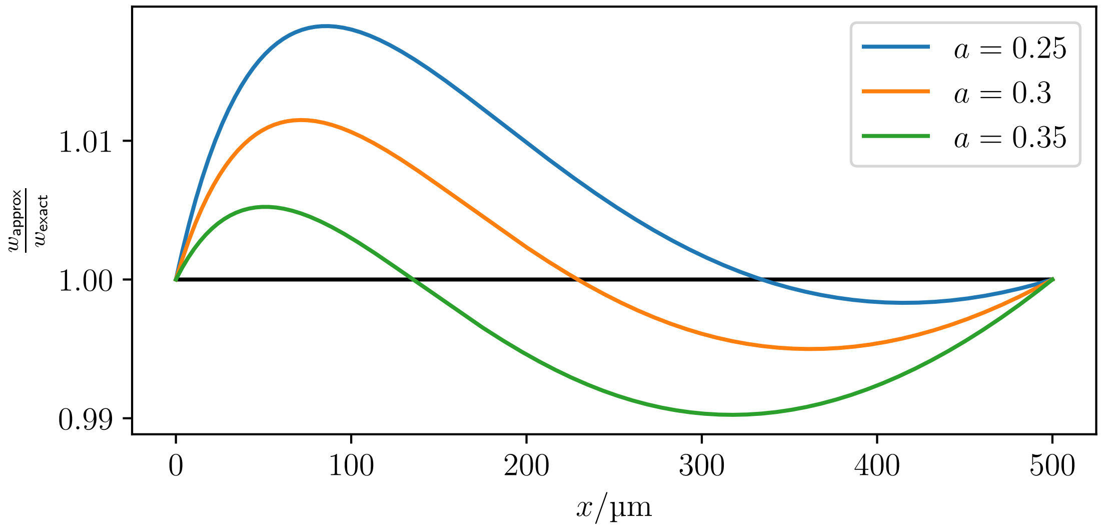


Figure 127: Channel width from a hyperbolic approximation in relation to the semi-analytical solution.

However, the extensional rate is very sensitive to the channel shape and therefore picking a value of around $a = 0.3$ still leads to $\dot{\epsilon}$ clearly not being constant in simulations. It rises at the inlet, reaches a maximum near the center and then falls off towards the outlet. To avoid this, the exact shape as computed by the algorithm outlined above is loaded into the simulation (simulation 29).

29 Water in a constant extensional rate channel

Box:	$2402 \times 802 \times 98$
	$L_0 = 312.5\,\mathrm{nm}$
Fluid:	Newtonian::water (fluid 1)
BC:	mass-flow (BC 3)
	$Q = 0.02\,\frac{\mu\mathrm{L}}{\mathrm{s}}$
	pressure (BC 4)

Figure 128 shows the extensional rate within the channel with the geometry as determined above. The extensional rate shows a slight ripple towards the outlet, which can be attributed to simulation artefacts due to stair-casing (see appendix P.4). The extensional rate in the simulation reaches an average of $35.1\,\frac{1}{\mathrm{s}}$ between 225 µm and 425 µm, where the extensional rate is most constant. This is approximately 1 % below the expected value and therefore in good agreement. It shall be pointed out, that this error is likely not only composed of simulation errors but in part due to the fact, that the simulation reaches equilibrium asymptotically. Therefore, the error could decrease (slightly) for longer simulation durations.

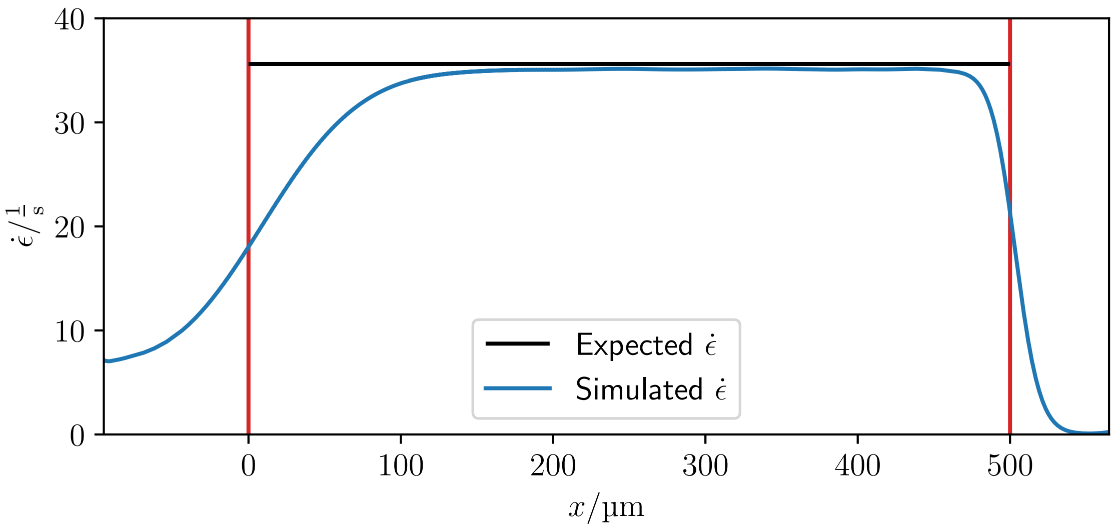


Figure 128: The simulated extensional rate for a Newtonian fluid (water 1) in the channel with the semi-analytically determined geometry. The varying width channel is between the red lines.

Aside from the expected edge effects, it is clear, that the performance near the inlet is considerably worse than near the outlet. The reason for this is likely, that the assumption of negligible radial velocity is least accurate here.

25.3. CY fluids

Shear-thinning fluids, represented here by the Carreau-Yasuda fluid model (CY), do not exhibit a constant extensional rate when placed in the channel constructed for Newtonian fluids (see figure 129 from simulation 30).

30 Methyl cellulose solution in a Newtonian channel

Box:	$2002 \times 602 \times 74$
	$L_0 \approx 417\,\mathrm{nm}$
Fluid:	CY::mc0_59_D (fluid 6)
BC:	mass-flow (BC 3)
	$Q = 0.02\,\frac{\mathrm{\mu L}}{\mathrm{s}}$
	pressure (BC 4)

It should be noted, that the simulations for more complex fluids have lower resolution than for the Newtonian case due to VRAM limitations.

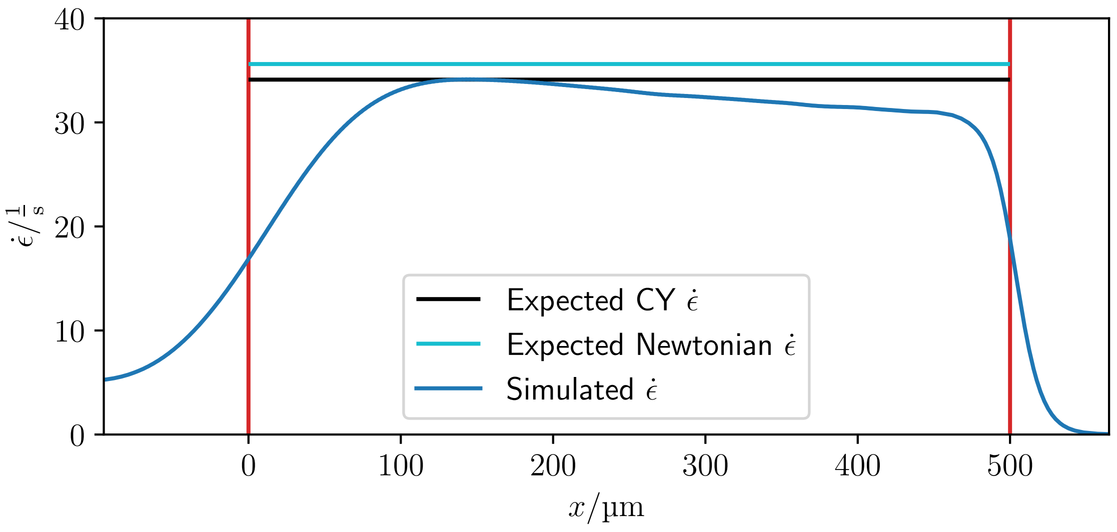


Figure 129: The simulated extensional rate for a shear thinning fluid (CY::mc0_59_D 6) in the channel with the Newtonian geometry. The varying width channel is between the red lines. Both the expected value for a Newtonian fluid and the expected value for this fluid in its correct channel have been included for reference.

The extensional rate in the simulation reaches an average of $32.2\,\frac{1}{\mathrm{s}}$ between $225\,\mathrm{\mu m}$ and $425\,\mathrm{\mu m}$, where the extensional rate is most constant. This is approximately 6 % to 10 % below the expected value, depending on the expectation. This constitutes poor agreement. Therefore, a differently shaped channel is necessary for use with shear-thinning fluids. Here a semi-analytical solution is no longer possible. The viscosity does not cancel out and therefore, the channel needs to be purpose-built for the fluid it is used with. The algorithm used to determine this shape is described using its component functions as before.

```
def run(a, b, G):
```

Run simulation 31 with a channel of size $a \times b$, which is driven by the pressure gradient G. Calculate Q and u_0 from the simulation output and return these values.

31 Find shape

Box:	$1 \times a + 2 \times b + 2$
	$L_0 \approx 1\,\mu\text{m}$
Fluids:	CY::mc0_59 (fluid 9)
	CY::mc0_59_D (fluid 6)
BC:	body-force (BC 13.3.2)

```
def guessStart(a, b, Q):
```

Guess a G, that will likely lead to a volume flow of approximately Q for a channel of size $a \times b$. This is done by calculating an effective radius of an equivalent circular channel as follows.

$$\frac{2}{r} = \sqrt{\frac{1}{a^2} + \frac{1}{b^2}} \tag{373}$$

The maximum velocity is approximated as follows.

$$u = \frac{2Q}{ab} \tag{374}$$

These poor approximations are valid here, as they only serve to provide an initial guess for a fit. From these an average shear-rate $\dot{\gamma}_{\text{avg}}$ can be guessed as follows.

$$\dot{\gamma}_{\text{avg}} = \frac{u}{r} \tag{375}$$

From this, the viscosity present at this shear-rate is calculated for the given fluid. Using this average viscosity, the volume-flow of a Newtonian rectangular channel of size $a \times b$ is calculated assuming $G = 1\,\frac{\text{N}}{\text{m}^3}$. Scaling $G = 1\,\frac{\text{N}}{\text{m}^3}$ by the quotient of the desired Q and this rough estimate yields an estimate for G. Despite the rough approximations made here, this estimate is decent, which is crucial to reduce the mount of (very expensive) steps the fit has to make.

```
def findG(a, b, Q):
```

Finds a G, so that a channel of size $a \times b$ develops a volume flow of Q. An initial guess is provided by `guessStart`, after which a root-finding algorithm (`scipy.optimize.root_scalar` [117]) is used on `run` to find the desired G. As this fit runs a simulation at every step, it is extremely expensive, and a good initial guess is crucial.

```
def doFit():
```

Determines supporting points for the width w of the desired channel as a function of u_0. For each w between w_c and w_u with a resolution of 1 µm (500 points), `findG` is invoked to find the pressure gradient for this w and the globally constant h and Q. Once G is found `run` is invoked to find the associated u_0. The w and its corresponding u_0 are saved.

```
def process():
```

Determines the final shape. Read the result from `doFit`. The expected extensional rate is calculated using the first and last u_0 together with the known length of L_c. Equally-spaced positions are created along the channel. Using the determined $\dot{\epsilon}$ u_0 is determined at these locations. With the data from the previous function as supporting points an interpolation is performed at these u_0 using `np.interp` [118]. With this, the width is recovered as a function of space and can be used for simulations.

32 Methyl cellulose solution in a CY channel

Box:	$2002 \times 602 \times 74$
	$L_0 \approx 417\,\text{nm}$
Fluids:	CY::mc0_59 (fluid 9)
	CY::mc0_59_D (fluid 6)
BC:	mass-flow (BC 3)
	$Q = 0.02\,\frac{\mu\text{L}}{\text{s}}$
	pressure (BC 4)

Simulation 32 is run for two instances of a specific CY fluid and their associated channel. CY::mc0_59_D is designed to most closely resemble the fluid used by Reichel *et al.* [99] (see fluid 6). The simulation output using it can be seen in figure 130. The extensional rate in the simulation reaches an average of $32.9\,\frac{1}{\text{s}}$ between 225 µm and 425 µm, where the extensional rate is most constant. This is approximately 2 % below the expected value and therefore in good agreement.

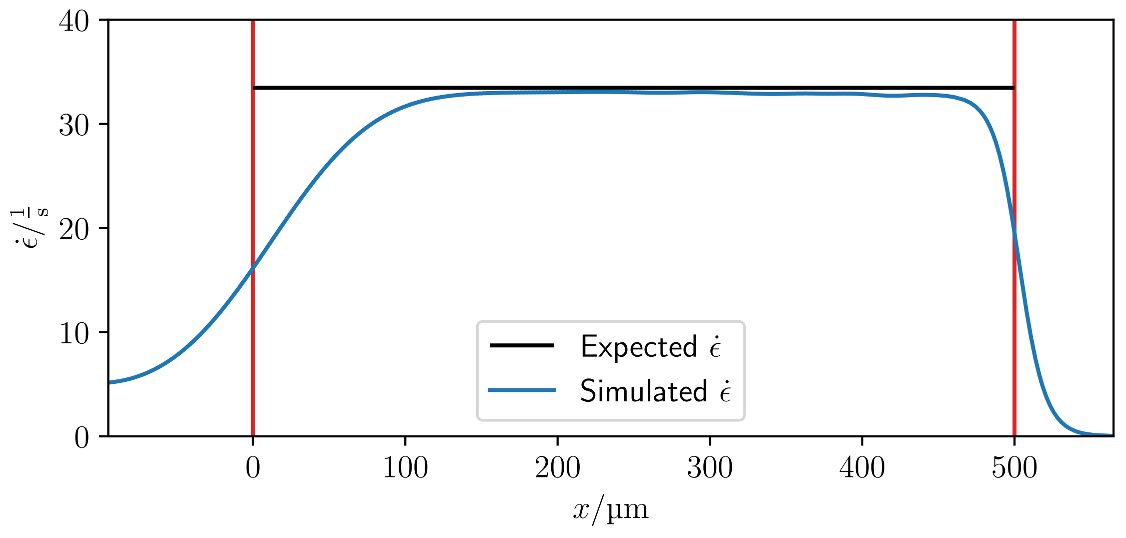


Figure 130: The simulated extensional rate for a shear thinning fluid (CY::mc0_59_D 6) in the channel with the appropriate geometry. The varying width channel is between the red lines.

CY::mc0_59 results from fitting the CY model to the result of fitting the PTT model to the same data. This is done to isolate elastic from viscous effects by comparing these two. The simulation result for this fluid as seen in figure 131 is consequently very similar. The extensional rate in the simulation reaches an average of 33.6 $\frac{1}{\mathrm{s}}$ between 225 µm and 425 µm, where the extensional rate is most constant. This is approximately 1 % below the expected value and therefore in good agreement.

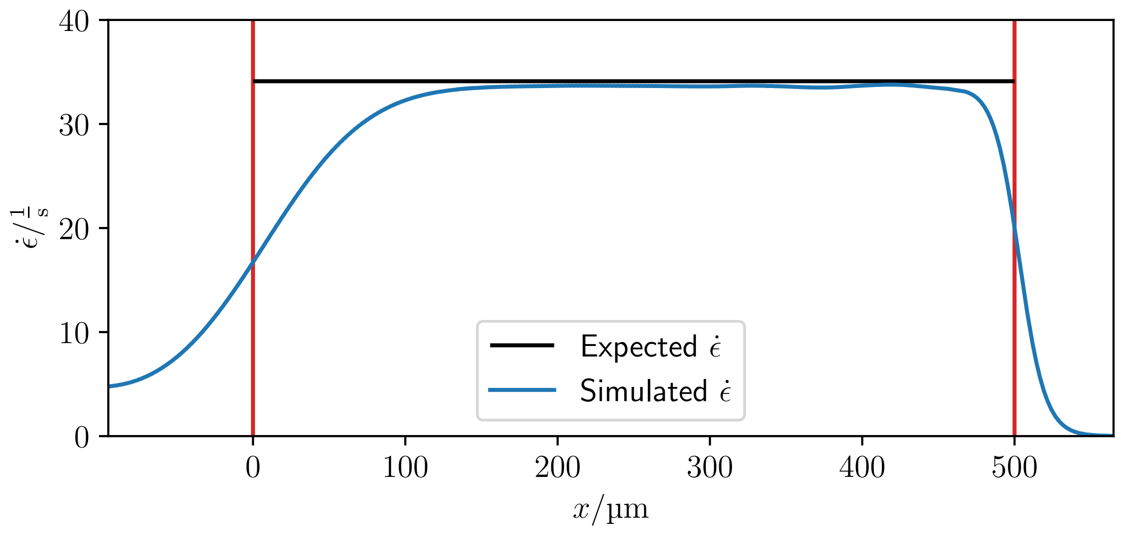


Figure 131: The simulated extensional rate for a shear thinning fluid (CY::mc0_59 9) in the channel with the appropriate geometry. The varying width channel is between the red lines.

For more complicated fluid models, even more effects need considering.

25.4. PTT fluids

The fluids used by Reichel *et al.* [99] exhibit viscoelastic behavior. This is modeled here by using the Phan-Thien and Tanner fluid model (PTT). Compared to the CY model PTT has a first normal stress difference and is retarded. The first normal stress difference only influences the velocity profile around geometry change. The retardation only takes effect as time progresses. Therefore, both effects can only be accounted for when simulating the whole channel. Optimizing the channel shape this way is theoretically possible, but is - at this time - prohibitively expensive. Therefore, the only sensible option is to determine the amount of error cause by ignoring these effects.

33 PTT Methyl cellulose solution in a CY channel

Box:	$2002 \times 602 \times 74$
	$L_0 \approx 417\,\mathrm{nm}$
Fluids:	PTT::mc0_59 (fluid 3)
BC:	mass-flow (BC 3)
	$Q = 0.02\,\frac{\mathrm{\mu L}}{\mathrm{s}}$
	pressure (BC 4)

Simulation 33 took considerably longer to equilibrate than its CY counterpart. The result can be seen in figure 132 and luckily, the effects discussed above seem to not matter much. The extensional rate in the simulation reaches an average of $33.6\,\frac{1}{\mathrm{s}}$ between $225\,\mathrm{\mu m}$ and $425\,\mathrm{\mu m}$, where the extensional rate is most constant. This is approximately $2\,\%$ below the expected value and therefore in good agreement.

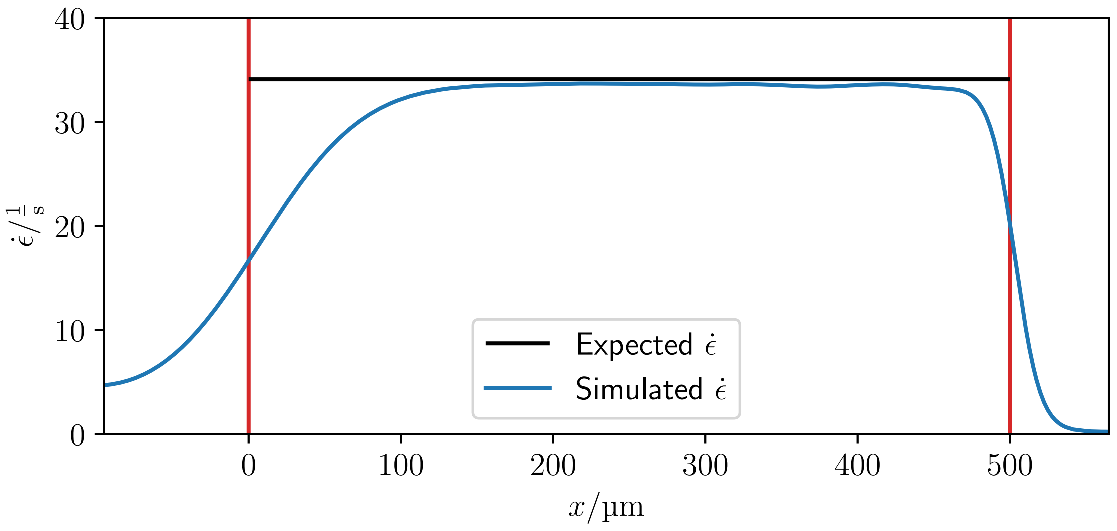


Figure 132: The simulated extensional rate for a PTT fluid (PTT::mc0_59 3) in the channel for the corresponding CY fluid. The varying width channel is between the red lines.

The agreement is only worse than for CY by a fraction of a percent. In general, this depends on the concrete fluid. For the fluid used here, the CY channel is appropriate for PTT. The comparison between simulation 32 and simulation 33 can be seen in figure 133. Aside from a very slight drop in $\dot{\eta}$ along the plateau and $\dot{\eta}$ not returning completely to 0 after the channel no discrepancy can be seen. The second discrepancy is due to the inherent first normal stress difference of this fluid and does not matter here.

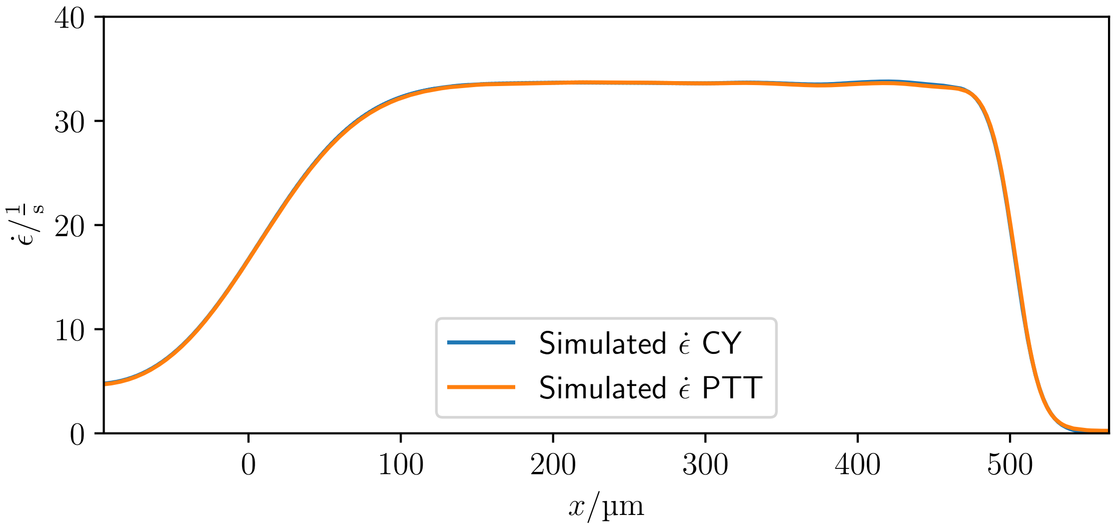


Figure 133: Comparison between the simulated extensional rate for a PTT fluid (PTT::mc0_59 3) and its corresponding CY fluid (CY::mc0_59 9).

25.5. Conclusion

The channels developed in this section are very similar as can be seen in figure 134. It can be seen, that the channel from Reichel *et al.* [99] being most different from the rest. The remaining ones are extremely close, with the CY curve that is also used for PTT and the one for the direct fit being nearly identical. Still, these minuscule differences yield significant differences in $\dot{\epsilon}$. This shows how strict the shape has to be controlled for proper results.

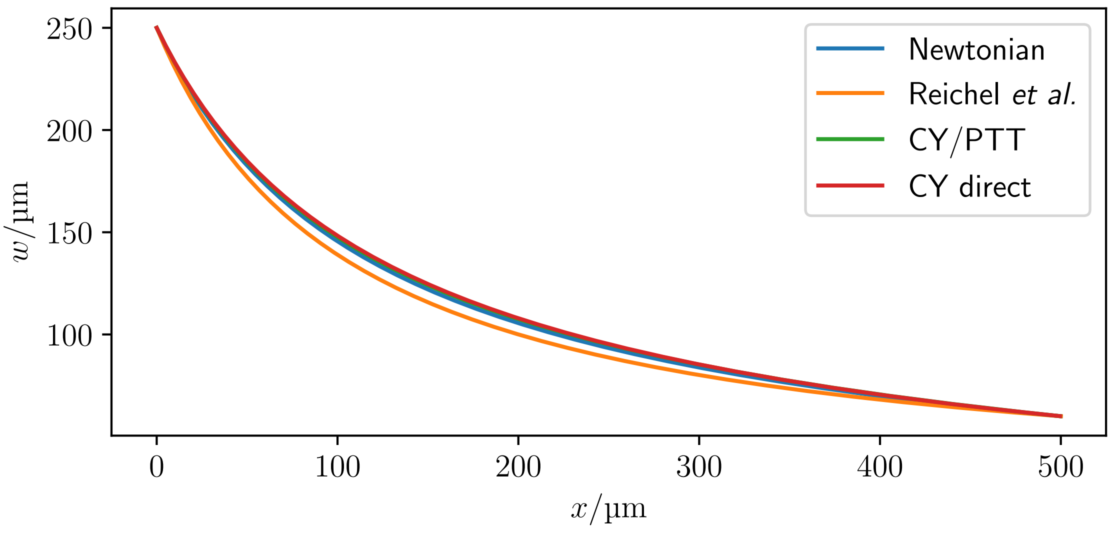


Figure 134: Channel width from the different solutions.

With proper controls in place however, this section shows, that a remarkably constant extensional rate can be archived. As shown above, no general shape can be found for non-Newtonian fluids. Instead, the channel has to be propose-built for the used fluid. As mentioned before, experiments will not be able to fully reproduce the simulations results presented here. However, this section demonstrates, that the state-of-the-art in constant-extensional-rate channel-design can be significantly improved. This concludes the applications discussed in this thesis. Next, some final concluding words are provided.

26. Discussion and conclusions

The very robust simulation frameworks developed in this thesis are applicable to a wide variety of problems. For the first time viscoelastic fluids in a parameter range realistic for bio-inks and other cell carrier fluids can be simulated using a Lattice Boltzmann method. With this many experiments, which could not be reproduced with simulations yet can be analyzed. This thesis starts with a brief introduction into the required theory and simulation techniques (see part I). This covers the derivation of many important quantities and relations. It simplifies existing definitions and alters some to be more powerful. The simulation algorithm is thoroughly validated. After the theory is covered this thesis applies it to several preexisting topics. Section 19 covered the extraction of fluid parameters. Afterwards, the extraction of cell parameters, primarily the Young's modulus was covered in section 20. Here a focus was on explaining cell behavior at higher deformations using higher order models. The simulation technique used for this was introduced to shrink the error significantly. Next, section 21 covered the influence of viscoelastic fluids on cells in a shear-flow. The large discrepancy, that is caused by the elastic forces has been observed in experiments before. Now it is possible to quantify and understand them. This behavior can now also be used to extract cell parameters. Afterwards, a surprising result was found in section 22. The fluid stress persisting for a bit after the exit of a printing needle does not smooth out the forces. And existing cell does instead experience an amplified deformation. This result can be used to rethink bio-ink manufacturing to attempt to reduce the viscoelastic nature of these fluids. A purely shear-thinning fluid would be better in this case. Section 23 explored cells in viscoelastic Poiseuille flows. Here it was shown how the elastic effects can be missed by only using a single suboptimal deformation metric. It highlights which aspect of the deformation are similar to purely viscous fluids, and how differences can be spotted. These thoughts are used in section 24 to reproduce an experiment with a complicated geometry. The deformation seen in an RT-DC channel was reproduced without the prominent local minimum, that can be seen in viscous simulations, but not the experiment. These results can be used to improve disease detection, as they can somewhat accurately reproduce the experiment. Finally, section 25 cautions against using normal stresses as a god of the gaps argument if not all effects can be explained. It uses numerics to design channels, which actually have constant extensional flow. This section demonstrates an accuracy, that can even distinguish between using a cell position to probe a fluid flow and the actual fluid flow. Here simulations can inform decision on how to perform an experiment. This is valuable here, because manufacturing the channels is an involved process. Without the numeric solutions one would need to resort to trial and error. Throughout, this thesis, it is evident, that sometimes, the elastic contributions are not relevant at all. If they are, this thesis can describe them with an accuracy, that actually allows reproducing experiments. With the ability to fully describe experiments, the simulations can be used to tune them. They can be used to predict outcomes for expensive experiments, and they can be used for cell characterization in combination with any experiment. The validations in parts I and III show the remarkable accuracy, these algorithms are capable of. However, part II shows how much accuracy one actually gets, if one does not have enough patience. These numeric methods can be used for a wide variety of topics in the future. A few ideas follow.

27. Outlook

The section on channels of constant extensional rate (section 25) provides concrete shapes that can be used to actually get constant extensional rates. This should be manufactured and tested. It seems likely, that a derivative model of FENE-P could be produced, which better describes the fluid data. *FluidX3D* will be polished and cleaned up. The requirement to recompile it for each setup will be reduced and eventually removed entirely. Additional contributions to the cell surface could be investigated. Interactions between the cells will be an interesting topic to pursue. Clearly, this thesis provides an amazing tool, which can be used for the reproduction and analysis of many different experiments. If forms a solid base from which one can expand, to cover new fluid models, new cell interactions or something completely unheard of. This is where your journey starts.

References

[1] Richard Kellnberger, Tomasz Jüngst, and Stephan Gekle. “Novel lattice Boltzmann method for simulation of strongly shear thinning viscoelastic fluids.” In: *International Journal for Numerical Methods in Fluids* (Oct. 2024). ISSN: 1097-0363.

[2] Nadine Raßmann et al. “Determining the Elastic Modulus of Microgel Particles by Nanoindentation.” In: *ACS Applied Nano Materials* 8.11 (Mar. 2025), 5383–5398. ISSN: 2574-0970.

[3] Axel Pössl, David Hartzke, Thomas M Schmidts, Frank E Runkel, and Peggy Schlupp. “A targeted rheological bioink development guideline and its systematic correlation with printing behavior.” In: *Biofabrication* 13.3 (Apr. 2021), p. 035021. ISSN: 1758-5090.

[4] Gowsihan Poologasundarampillai, Abdelrahman Haweet, Soher Nagi Jayash, George Morgan, James E. Moore, and Alessia Candeo. “Real-time imaging and analysis of cell-hydrogel interplay within an extrusion-bioprinting capillary.” In: *Bioprinting* 23 (2021), e00144. ISSN: 2405-8866.

[5] Cathal O’Connell, Junxiang Ren, Leon Pope, Yifan Zhang, Anushree Mohandas, Romane Blanchard, Serena Duchi, and Carmine Onofrillo. “Correction to: Characterizing Bioinks for Extrusion Bioprinting: Printability and Rheology.” In: *3D Bioprinting* (2022), C1–C1. ISSN: 1940-6029.

[6] Nicole Toepfner et al. “Detection of human disease conditions by single-cell morpho-rheological phenotyping of blood.” In: *eLife* 7 (2018), e29213.

[7] Naomi Paxton, Willi Smolan, Thomas Böck, Ferry Melchels, Jürgen Groll, and Tomasz Jungst. “Proposal to assess printability of bioinks for extrusion-based bioprinting and evaluation of rheological properties governing bioprintability.” In: *Biofabrication* 9.4 (Nov. 2017), p. 044107. ISSN: 1758-5090.

[8] M.A. Alves, P.J. Oliveira, and F.T. Pinho. “Numerical Methods for Viscoelastic Fluid Flows.” In: *Annual Review of Fluid Mechanics* 53.1 (2020), 1–33. ISSN: 0066-4189.

[9] Timm Krüger, Halim Kusumaatmaja, Alexandr Kuzmin, Orest Shardt, Goncalo Silva, and Erlend Magnus Viggen. *The Lattice Boltzmann Method*. Springer. Springer, 2017.

[10] Iaroslav Ispolatov and Martin Grant. “Lattice Boltzmann method for viscoelastic fluids.” In: *Physical Review E* 65.5 (May 2002). ISSN: 1095-3787.

[11] Di Wang, Danielle Tan, and Nhan Phan-Thien. “A lattice Boltzmann method for simulating viscoelastic drops.” In: *Physics of Fluids* 31.7 (July 2019). ISSN: 1089-7666.

[12] Jin Su, Jie Ouyang, Xiaodong Wang, Binxing Yang, and Wen Zhou. “Lattice Boltzmann method for the simulation of viscoelastic fluid flows over a large range of Weissenberg numbers.” In: *Journal of Non-Newtonian Fluid Mechanics* 194 (Apr. 2013), 42–59. ISSN: 0377-0257.

[13] L Giraud, D d'Humières, and P Lallemand. "A lattice Boltzmann model for Jeffreys viscoelastic fluid." In: *Europhysics Letters (EPL)* 42.6 (June 1998), 625–630. ISSN: 1286-4854.

[14] Chiyu Xie, Wenhai Lei, and Moran Wang. "Lattice Boltzmann model for three-phase viscoelastic fluid flow." In: *Physical Review E* 97.2 (Feb. 2018). ISSN: 2470-0053.

[15] V. Dzanic, C.S. From, and E. Sauret. "A hybrid lattice Boltzmann model for simulating viscoelastic instabilities." In: *Computers &amp; Fluids* 235 (Mar. 2022), p. 105280. ISSN: 0045-7930.

[16] Xavier Frank and Huai Z. Li. "Negative wake behind a sphere rising in viscoelastic fluids: A lattice Boltzmann investigation." In: *Physical Review E* 74.5 (Nov. 2006). ISSN: 1550-2376.

[17] Shun Zou, Xue-Feng Yuan, Xuejun Yang, Wei Yi, and Xinhai Xu. "An integrated lattice Boltzmann and finite volume method for the simulation of viscoelastic fluid flows." In: *Journal of Non-Newtonian Fluid Mechanics* 211 (Sept. 2014), 99–113. ISSN: 0377-0257.

[18] Young Ki Lee and Kyung Hyun Ahn. "A novel Lattice Boltzmann method for the dynamics of rigid particles suspended in a viscoelastic medium." In: *Journal of Non-Newtonian Fluid Mechanics* 244 (June 2017), 75–84. ISSN: 0377-0257.

[19] V. Dzanic, C.S. From, and E. Sauret. "Assessment of polymer feedback coupling approaches in simulation of viscoelastic fluids using the lattice Boltzmann method." In: *Computers &amp; Fluids* 246 (Oct. 2022), p. 105629. ISSN: 0045-7930.

[20] M. H. Sedaghat. "A Hybrid Immersed Boundary-Lattice Boltzmann Method for Simulation of Viscoelastic Fluid Flows Interaction with Complex Boundaries." In: *Communications in Computational Physics* 29.5 (June 2021), 1411–1445. ISSN: 1991-7120.

[21] George N. Frantziskonis. "Lattice Boltzmann method for multimode wave propagation in viscoelastic media and in elastic solids." In: *Physical Review E* 83.6 (June 2011). ISSN: 1550-2376.

[22] Shenxu Qin, Maoqiang Jiang, Kuang Ma, Jin Su, and Zhaohui Liu. "Fully resolved simulations of viscoelastic suspensions by an efficient immersed boundary-lattice Boltzmann method." In: *Particuology* 75 (Apr. 2023), 26–49. ISSN: 1674-2001.

[23] Masato YOSHINO, Yasuyuki TORIUMI, and Masahiro ARAI. "Lattice Boltzmann Simulation of Two-Phase Viscoelastic Fluid Flows." In: *Journal of Computational Science and Technology* 2.2 (2008), 330–340. ISSN: 1881-6894.

[24] S. Papenkort and Th. Voigtmann. "Lattice Boltzmann simulations of a viscoelastic shear-thinning fluid." In: *The Journal of Chemical Physics* 143.4 (July 2015). ISSN: 1089-7690.

[25] T. N. Phillips and G. W. Roberts. “Lattice Boltzmann models for non-Newtonian flows.” In: *IMA Journal of Applied Mathematics* 76.5 (Mar. 2011), 790–816. ISSN: 1464-3634.

[26] Junya Onishi, Yu Chen, and Hirotada Ohashi. “A Lattice Boltzmann model for polymeric liquids.” In: *Progress in Computational Fluid Dynamics, An International Journal* 5.1/2 (2005), p. 75. ISSN: 1741-5233.

[27] O. Malaspinas, N. Fiétier, and M. Deville. “Lattice Boltzmann method for the simulation of viscoelastic fluid flows.” In: *Journal of Non-Newtonian Fluid Mechanics* 165.23–24 (Dec. 2010), 1637–1653. ISSN: 0377-0257.

[28] A. Gupta, M. Sbragaglia, and A. Scagliarini. “Hybrid Lattice Boltzmann/Finite Difference simulations of viscoelastic multicomponent flows in confined geometries.” In: *Journal of Computational Physics* 291 (June 2015), 177–197. ISSN: 0021-9991.

[29] Anupam Gupta and Mauro Sbragaglia. “Deformation and break-up of Viscoelastic Droplets Using Lattice Boltzmann Models.” In: *Procedia IUTAM* 15 (2015), 215–227. ISSN: 2210-9838.

[30] Pierre Lallemand, Dominique d’Humières, Li-Shi Luo, and Robert Rubinstein. “Theory of the lattice Boltzmann method: Three-dimensional model for linear viscoelastic fluids.” In: *Physical Review E* 67.2 (Feb. 2003). ISSN: 1095-3787.

[31] Jingtao Ma, Zhen Wang, John Young, Joseph C.S. Lai, Yi Sui, and Fang-Bao Tian. “An immersed boundary-lattice Boltzmann method for fluid-structure interaction problems involving viscoelastic fluids and complex geometries.” In: *Journal of Computational Physics* 415 (Aug. 2020), p. 109487. ISSN: 0021-9991.

[32] J. Onishi, Y. Chen, and H. Ohashi. “Dynamic simulation of multi-component viscoelastic fluids using the lattice Boltzmann method.” In: *Physica A: Statistical Mechanics and its Applications* 362.1 (Mar. 2006), 84–92. ISSN: 0378-4371.

[33] Di Wang, Danielle S. Tan, Boo Cheong Khoo, Zhenyu Ouyang, and Nhan Phan-Thien. “A lattice Boltzmann modeling of viscoelastic drops’ deformation and breakup in simple shear flows.” In: *Physics of Fluids* 32.12 (Dec. 2020). ISSN: 1089-7666.

[34] Paul J. Dellar. “Lattice Boltzmann Formulation for Linear Viscoelastic Fluids Using an Abstract Second Stress.” In: *SIAM Journal on Scientific Computing* 36.6 (Jan. 2014), A2507–A2532. ISSN: 1095-7197.

[35] Jin Su, Jie Ouyang, Xiaodong Wang, and Binxin Yang. “Lattice Boltzmann method coupled with the Oldroyd-B constitutive model for a viscoelastic fluid.” In: *Physical Review E* 88.5 (Nov. 2013). ISSN: 1550-2376.

[36] Di Wang, Danielle S. Tan, Boo Cheong Khoo, Zhenyu Ouyang, and Nhan Phan-Thien. “A lattice Boltzmann modeling of the bubble velocity discontinuity (BVD) in shear-thinning viscoelastic fluids.” In: *Physics of Fluids* 33.3 (Mar. 2021). ISSN: 1089-7666.

[37] Michael Kuron, Cameron Stewart, Joost de Graaf, and Christian Holm. "An extensible lattice Boltzmann method for viscoelastic flows: complex and moving boundaries in Oldroyd-B fluids." In: *The European Physical Journal E* 44.1 (Jan. 2021). ISSN: 1292-895X.

[38] Anupam Gupta and Mauro Sbragaglia. "A lattice Boltzmann study of the effects of viscoelasticity on droplet formation in microfluidic cross-junctions." In: *The European Physical Journal E* 39.1 (Jan. 2016). ISSN: 1292-895X.

[39] E. Buckingham. "On Physically Similar Systems; Illustrations of the Use of Dimensional Equations." In: *Physical Review* 4.4 (Oct. 1914), 345–376. ISSN: 0031-899X.

[40] E. BUCKINGHAM. "The Principle of Similitude." In: *Nature* 96.2406 (Dec. 1915), 396–397. ISSN: 1476-4687.

[41] E. Buckingham. "Model Experiments and the Forms of Empirical Equations." In: *Transactions of the American Society of Mechanical Engineers* 37 (Jan. 1915), 263–292. ISSN: 0097-6822.

[42] G. K. Batchelor. *An Introduction to Fluid Dynamics.* Cambridge University Press, 1967.

[43] Élisabeth Guazzelli and Jeffrey F. Morris. *A Physical Introduction to Suspension Dynamics.* 2011.

[44] George Gabriel Stokes. "On the Theories of the Internal Friction of Fluids in Motion, and of the Equilibrium and Motion of Elastic Solids." In: *Mathematical and Physical Papers vol.1.* Cambridge University Press, 1880, 75–129.

[45] Leonhard Euler. "Principles of the motion of fluids." In: *Physica D: Nonlinear Phenomena* 237.14–17 (Aug. 1757), 1840–1854. ISSN: 0167-2789.

[46] P. L. Bhatnagar, E. P. Gross, and M. Krook. "A Model for Collision Processes in Gases. I. Small Amplitude Processes in Charged and Neutral One-Component Systems." In: *Physical Review* 94.3 (May 1954), 511–525. ISSN: 0031-899X.

[47] Moritz Lehmann, Mathias J. Krause, Giorgio Amati, Marcello Sega, Jens Harting, and Stephan Gekle. "Accuracy and performance of the lattice Boltzmann method with 64-bit, 32-bit, and customized 16-bit number formats." In: *Physical Review E* 106.1 (July 2022). ISSN: 2470-0053.

[48] Zhaoli Guo, Chuguang Zheng, and Baochang Shi. "Discrete lattice effects on the forcing term in the lattice Boltzmann method." In: *Physical Review E* 65.4 (Apr. 2002). ISSN: 1095-3787.

[49] H.J. Limbach, A. Arnold, B.A. Mann, and C. Holm. "ESPResSo—an extensible simulation package for research on soft matter systems." In: *Computer Physics Communications* 174.9 (May 2006), 704–727. ISSN: 0010-4655.

[50] D. Roehm and A. Arnold. "Lattice Boltzmann simulations on GPUs with ESPResSo." In: *The European Physical Journal Special Topics* 210.1 (Aug. 2012), 89–100. ISSN: 1951-6401.

[51] Irina Ginzburg, Dominique d'Humières, and Alexander Kuzmin. "Optimal Stability of Advection-Diffusion Lattice Boltzmann Models with Two Relaxation Times for Positive/Negative Equilibrium." In: *Journal of Statistical Physics* 139.6 (Apr. 2010), 1090–1143. ISSN: 1572-9613.

[52] Zhenhua Chai, Baochang Shi, Zhaoli Guo, and Fumei Rong. "Multiple-relaxation-time lattice Boltzmann model for generalized Newtonian fluid flows." In: *Journal of Non-Newtonian Fluid Mechanics* 166.5–6 (Mar. 2011), 332–342. ISSN: 0377-0257.

[53] Pierre Lallemand and Li-Shi Luo. "Theory of the lattice Boltzmann method: Dispersion, dissipation, isotropy, Galilean invariance, and stability." In: *Physical Review E* 61.6 (June 2000), 6546–6562. ISSN: 1095-3787.

[54] Pierre J Carreau. "Rheological equations from molecular network theories." In: *Transactions of the Society of Rheology* 16.1 (1972), 99–127.

[55] Kenji Yasuda. "Investigation of the analogies between viscometric and linear viscoelastic properties of polystyrene fluids." PhD thesis. 1979.

[56] A.J. Franck. "Normal stresses in shear flow." In: (2017). Ed. by TA Instruments.

[57] P.J. Oliveira. "Alternative derivation of differential constitutive equations of the Oldroyd-B type." In: *Journal of Non-Newtonian Fluid Mechanics* 160.1 (July 2009), 40–46. ISSN: 0377-0257.

[58] JG Oldroyd. "Non-Newtonian effects in steady motion of some idealized elastico-viscous liquids." In: *Proceedings of the Royal Society of London. Series A. Mathematical and Physical Sciences* 245.1241 (1958), 278–297.

[59] R.B. Bird, P.J. Dotson, and N.L. Johnson. "Polymer solution rheology based on a finitely extensible bead—spring chain model." In: *Journal of Non-Newtonian Fluid Mechanics* 7.2–3 (Jan. 1980), 213–235. ISSN: 0377-0257.

[60] Robert Byron Bird, Charles F Curtiss, Robert C Armstrong, and Ole Hassager. *Dynamics of polymeric liquids, volume 2: Kinetic theory*. Wiley, 1987.

[61] Nhan Phan-Thien and Roger I. Tanner. "A new constitutive equation derived from network theory." In: *Journal of Non-Newtonian Fluid Mechanics* 2.4 (July 1977), 353–365. ISSN: 0377-0257.

[62] N. Phan-Thien. "A Nonlinear Network Viscoelastic Model." In: *Journal of Rheology* 22.3 (June 1978), 259–283. ISSN: 1520-8516.

[63] L.L. Ferrás, M.L. Morgado, M. Rebelo, Gareth H. McKinley, and A.M. Afonso. "A generalised Phan–Thien—Tanner model." In: *Journal of Non-Newtonian Fluid Mechanics* 269 (July 2019), 88–99. ISSN: 0377-0257.

[64] Sebastian J. Müller, Elham Mirzahossein, Emil N. Iftekhar, Christian Bächer, Stefan Schrüfer, Dirk W. Schubert, Ben Fabry, and Stephan Gekle. "Flow and hydrodynamic shear stress inside a printing needle during biofabrication." In: *PLOS ONE* 15.7 (July 2020). Ed. by Fang-Bao Tian, e0236371. ISSN: 1932-6203.

[65] Fabian Häusl. “Soft objects in newtonian and non-Newtonian fluids: A computational study of bubbles and capsules in flow.” MA thesis. 2021.

[66] Fabrizio Capuani, Ignacio Pagonabarraga, and Daan Frenkel. “Discrete solution of the electrokinetic equations.” In: *The Journal of Chemical Physics* 121.2 (June 2004), 973–986. ISSN: 1089-7690.

[67] Sebastian J. Müller, Ben Fabry, and Stephan Gekle. “Predicting Cell Stress and Strain during Extrusion Bioprinting.” In: *Physical Review Applied* 19.6 (June 2023). ISSN: 2331-7019.

[68] Katharina Gräßel. “Understanding Transient Red Blood Cell Shapes in Flow Using Numerical Methods.” PhD thesis. Universität Bayreuth, Fakultät für Mathematik, Physik und Informatik, 2024.

[69] Carina Bezold. “Simulation of Stem Cells in 3D-Bioprinters.” MA thesis. 2018.

[70] Sebastian J. Müller, Franziska Weigl, Carina Bezold, Christian Bächer, Krystyna Albrecht, and Stephan Gekle. “A hyperelastic model for simulating cells in flow.” In: *Biomechanics and Modeling in Mechanobiology* 20.2 (Nov. 2020), 509–520. ISSN: 1617-7940.

[71] Allan F. Bower. *Applied Mechanics of Solids.* CRC Press, Oct. 2009. ISBN: 9781439802489.

[72] Moritz Lehmann. *High Performance Free Surface LBM on GPUs.* 2021.

[73] Dominique Barthès-Biesel. “Motion and Deformation of Elastic Capsules and Vesicles in Flow.” In: *Annual Review of Fluid Mechanics* 48.1 (Jan. 2016), 25–52. ISSN: 1545-4479.

[74] DOMINIQUE BARTHÈS-BIESEL, ANNA DIAZ, and EMMANUELLE DHENIN. “Effect of constitutive laws for two-dimensional membranes on flow-induced capsule deformation.” In: *Journal of Fluid Mechanics* 460 (June 2002), 211–222. ISSN: 1469-7645.

[75] R. Skalak, A. Tozeren, R.P. Zarda, and S. Chien. “Strain Energy Function of Red Blood Cell Membranes.” In: *Biophysical Journal* 13.3 (Mar. 1973), 245–264. ISSN: 0006-3495.

[76] T. Krüger, F. Varnik, and D. Raabe. “Efficient and accurate simulations of deformable particles immersed in a fluid using a combined immersed boundary lattice Boltzmann finite element method.” In: *Computers &amp; Mathematics with Applications* 61.12 (June 2011), 3485–3505. ISSN: 0898-1221.

[77] P.B. Canham. “The minimum energy of bending as a possible explanation of the biconcave shape of the human red blood cell.” In: *Journal of Theoretical Biology* 26.1 (Jan. 1970), 61–81. ISSN: 0022-5193.

[78] W Helfrich. “Elastic Properties of Lipid Bilayers: Theory and Possible Experiments.” In: *Zeitschrift für Naturforschung C* 28.11–12 (Dec. 1973), 693–703. ISSN: 0939-5075.

[79] Achim Guckenberger, Marcel P. Schraml, Paul G. Chen, Marc Leonetti, and Stephan Gekle. "On the bending algorithms for soft objects in flows." In: *Computer Physics Communications* 207 (Oct. 2016), 1–23. ISSN: 0010-4655.

[80] Achim Guckenberger and Stephan Gekle. "Theory and algorithms to compute Helfrich bending forces: a review." In: *Journal of Physics: Condensed Matter* 29.20 (Apr. 2017), p. 203001. ISSN: 1361-648X.

[81] Sebastian Wohlrab, Sebastian Mueller, and Stephan Gekle. "Mechanical complexity of living cells can be mapped onto simple homogeneous equivalents." In: *Biomechanics and Modeling in Mechanobiology* 23.3 (Feb. 2024), 1067–1076. ISSN: 1617-7940.

[82] Massimiliano Fraldi, Stefania Palumbo, Angelo Rosario Carotenuto, Arsenio Cutolo, Luca Deseri, and Nicola Pugno. "Buckling soft tensegrities: Fickle elasticity and configurational switching in living cells." In: *Journal of the Mechanics and Physics of Solids* 124 (2019), 299–324.

[83] Charles S Peskin. "Flow patterns around heart valves: A numerical method." In: *Journal of Computational Physics* 10.2 (Oct. 1972), 252–271. ISSN: 0021-9991.

[84] Charles S Peskin. "Numerical analysis of blood flow in the heart." In: *Journal of Computational Physics* 25.3 (Nov. 1977), 220–252. ISSN: 0021-9991.

[85] Christophe Geuzaine and Jean-François Remacle. "Gmsh: A 3-D Finite Element Mesh Generator with built-in Pre- and Post-Processing Facilities." In: *International Journal for Numerical Methods in Engineering* 79.11 (Sept. 2009), 1309–1331. ISSN: 00295981.

[86] Yongguang Cheng, Luoding Zhu, and Chunze Zhang. "Numerical Study of Stability and Accuracy of the Immersed Boundary Method Coupled to the Lattice Boltzmann BGK Model." In: *Communications in Computational Physics* 16.1 (July 2014), 136–168. ISSN: 1991-7120.

[87] Mark Meyer, Mathieu Desbrun, Peter Schröder, and Alan H. Barr. "Discrete Differential-Geometry Operators for Triangulated 2-Manifolds." In: *Visualization and Mathematics III*. Springer Berlin Heidelberg, 2003, 35–57. ISBN: 9783662051054.

[88] Timm Krüger. *Computer Simulation Study of Collective Phenomena in Dense Suspensions of Red Blood Cells under Shear*. Vieweg+Teubner Verlag, 2012. ISBN: 9783834823762.

[89] Moritz Lehmann, Sebastian Johannes Müller, and Stephan Gekle. "Efficient viscosity contrast calculation for blood flow simulations using the lattice Boltzmann method." In: *International Journal for Numerical Methods in Fluids* 92.11 (Apr. 2020), 1463–1477. ISSN: 1097-0363.

[90] Heinrich Hertz. "Ueber die Berührung fester elastischer Körper." In: *crll* 1882.92 (1882), 156–171. ISSN: 1435-5345.

[91] Michael Glaubitz, Nikolay Medvedev, Daniel Pussak, Laura Hartmann, Stephan Schmidt, Christiane A. Helm, and Mihaela Delcea. "A novel contact model for AFM indentation experiments on soft spherical cell-like particles." In: *Soft Matter* 10.35 (June 2014), p. 6732. ISSN: 1744-6848.

[92] Ian N. Sneddon. "The relation between load and penetration in the axisymmetric boussinesq problem for a punch of arbitrary profile." In: *International Journal of Engineering Science* 3.1 (May 1965), 47–57. ISSN: 0020-7225.

[93] Sebastian Wohlrab. "Finite-element analysis of various influences on cell and microparticle response in AFM measurements." MA thesis. 2024.

[94] Daan Frenkel and Berend Smit. Elsevier, 2002. ISBN: 9780122673511.

[95] R. Roscoe. "On the rheology of a suspension of viscoelastic spheres in a viscous liquid." In: *Journal of Fluid Mechanics* 28.02 (May 1967), p. 273. ISSN: 1469-7645.

[96] F. Snijkers, G. D'Avino, P.L. Maffettone, F. Greco, M.A. Hulsen, and J. Vermant. "Effect of viscoelasticity on the rotation of a sphere in shear flow." In: *Journal of Non-Newtonian Fluid Mechanics* 166.7–8 (Apr. 2011), 363–372. ISSN: 0377-0257.

[97] S. RAMANUJAN and C. POZRIKIDIS. "Deformation of liquid capsules enclosed by elastic membranes in simple shear flow: large deformations and the effect of fluid viscosities." In: *Journal of Fluid Mechanics* 361 (Apr. 1998), 117–143. ISSN: 1469-7645.

[98] Lucas Daniel Wittwer, Felix Reichel, Paul Müller, Jochen Guck, and Sebastian Aland. "A new hyperelastic lookup table for RT-DC." In: *Soft Matter* 19.11 (2023), 2064–2073. ISSN: 1744-6848.

[99] Felix Reichel, Ruchi Goswami, Salvatore Girardo, and Jochen Guck. "High-throughput viscoelastic characterization of cells in hyperbolic microchannels." In: *Lab on a Chip* 24.9 (2024), 2440–2453. ISSN: 1473-0189.

[100] Richard Gerum et al. "Viscoelastic properties of suspended cells measured with shear flow deformation cytometry." In: *eLife* 11 (Sept. 2022). ISSN: 2050-084X.

[101] Markus Wittmann, Thomas Zeiser, Georg Hager, and Gerhard Wellein. "Comparison of different propagation steps for lattice Boltzmann methods." In: *Computers &amp; Mathematics with Applications* 65.6 (Mar. 2013), 924–935. ISSN: 0898-1221.

[102] David J Holdych, David R Noble, John G Georgiadis, and Richard O Buckius. "Truncation error analysis of lattice Boltzmann methods." In: *Journal of Computational Physics* 193.2 (Jan. 2004), 595–619. ISSN: 0021-9991.

[103] P.A. Amorim, M.A. d'Ávila, R. Anand, P. Moldenaers, P. Van Puyvelde, and V. Bloemen. "Insights on shear rheology of inks for extrusion-based 3D bioprinting." In: *Bioprinting* 22 (June 2021), e00129. ISSN: 2405-8866.

[104] Maziyar Jalaal, Graeme Cottrell, Neil Balmforth, and Boris Stoeber. "On the rheology of Pluronic F127 aqueous solutions." In: *Journal of Rheology* 61.1 (Jan. 2017), 139–146. ISSN: 1520-8516.

[105] W. M. Kulicke, G. Kiss, and Roger S. Porter. “Inertial normal-force corrections in rotational rheometry.” In: *Rheologica Acta* 16.5 (Sept. 1977), 568–572. ISSN: 1435-1528.

[106] N. Periasamy, H. P. Kao, K. Fushimi, and A. S. Verkman. “Organic osmolytes increase cytoplasmic viscosity in kidney cells.” In: *American Journal of Physiology-Cell Physiology* 263.4 (Oct. 1992), C901–C907. ISSN: 1522-1563.

[107] Beyza Büyükurgancı, Santanu Kumar Basu, Markus Neuner, Jochen Guck, Andreas Wierschem, and Felix Reichel. “Shear rheology of methyl cellulose based solutions for cell mechanical measurements at high shear rates.” In: *Soft Matter* 19.9 (2023), 1739–1748. ISSN: 1744-6848.

[108] Oliver Otto et al. “Real-time deformability cytometry: on-the-fly cell mechanical phenotyping.” In: *Nature Methods* 12.3 (Feb. 2015), 199–202. ISSN: 1548-7105.

[109] M Heuberger, G Dietler, and L Schlapbach. “Mapping the local Young’s modulus by analysis of the elastic deformations occurring in atomic force microscopy.” In: *Nanotechnology* 6.1 (Jan. 1995), 12–23. ISSN: 1361-6528.

[110] Gaetano D’Avino, Martien A. Hulsen, Frank Snijkers, Jan Vermant, Francesco Greco, and Pier Luca Maffettone. “Rotation of a sphere in a viscoelastic liquid subjected to shear flow. Part I: Simulation results.” In: *Journal of Rheology* 52.6 (Nov. 2008), 1331–1346. ISSN: 1520-8516.

[111] Marta Urbanska, Philipp Rosendahl, Martin Kräter, and Jochen Guck. “High-throughput single-cell mechanical phenotyping with real-time deformability cytometry.” In: *Microfluidics in Cell Biology Part B: Microfluidics in Single Cells.* Elsevier, 2018, 175–198. ISBN: 9780128142820.

[112] Bob Fregin, Fabian Czerwinski, Doreen Biedenweg, Salvatore Girardo, Stefan Gross, Konstanze Aurich, and Oliver Otto. “High-throughput single-cell rheology in complex samples by dynamic real-time deformability cytometry.” In: *Nature Communications* 10.1 (2019), 1 – 11.

[113] Felix Reichel. “High-Throughput Viscoelastic Characterization of Cells in Microchannels.” PhD thesis. 2024.

[114] Lucas Daniel Wittwer, Felix Reichel, and Sebastian Aland. “Numerical simulation of deformability cytometry: Transport of a biological cell through a microfluidic channel.” In: *Modeling of Mass Transport Processes in Biological Media.* Elsevier, 2022, 33–56. ISBN: 9780443157653.

[115] J. Boussinesq. “Mémoire sur l’influence des frottements dans les mouvements réguliers des fluides.” In: *Journal de mathématiques pures et appliquées 2e série* 13 (Nov. 1868), 377–424.

[116] Wikipedia contributors. *Hagen–Poiseuille equation — Wikipedia, The Free Encyclopedia.* [Online; accessed 15-March-2025]. 2024.

[117] Pauli Virtanen et al. “SciPy 1.0: fundamental algorithms for scientific computing in Python.” In: *Nature Methods* 17.3 (Feb. 2020), 261–272. ISSN: 1548-7105.

[118] Charles R. Harris et al. "Array programming with NumPy." In: *Nature* 585.7825 (Sept. 2020), 357–362.

[119] Dominique d'Humieres, Irina Ginzburg, Manfred Krafczyk, Pierre Lallemand, and Li-Shi Luo. *Multiple-relaxation-time Lattice Boltzmann Models in 3D.* Tech. rep. 2002.

[120] Mark J. Mawson and Alistair J. Revell. "Memory transfer optimization for a lattice Boltzmann solver on Kepler architecture nVidia GPUs." In: *Computer Physics Communications* 185.10 (Oct. 2014), 2566–2574. ISSN: 0010-4655.

[121] Karel Rektorys. *Survey of applicable mathematics.* Vol. 280. Springer, 2013.

[122] Paulo J Oliveira and Fernando T Pinho. "Analytical solution for fully developed channel and pipe flow of Phan-Thien–Tanner fluids." In: *Journal of Fluid Mechanics* 387 (1999), 271–280.

[123] H. L. Armstrong. "The Oscillating Spring and Weight—An Experiment Often Misinterpreted." In: *American Journal of Physics* 37.4 (Apr. 1969), 447–449. ISSN: 1943-2909.

[124] Jun-ichi Ueda and Yoshiro Sadamoto. "A Measurement of the Effective Mass of Coil Springs." In: *Journal of the Physical Society of Japan* 66.2 (Feb. 1997), 367–368. ISSN: 1347-4073.

[125] Sebastian Johannes Müller. *From theory to application-3D bioprinting of cells.* Universitaet Bayreuth (Germany), 2023.

[126] Ben Ashbaugh. *cl_ext_float_atomics.* Tech. rep. 2022.

[127] Nicholas J. Higham. "Computing the Polar Decomposition—with Applications." In: *SIAM Journal on Scientific and Statistical Computing* 7.4 (Oct. 1986), 1160–1174. ISSN: 2168-3417.

[128] "IEEE Standard for Floating-Point Arithmetic." In: *IEEE Std 754-2019 (Revision of IEEE 754-2008)* (2019), 1–84.

[129] Henrik Bruus. "Governing Equations in Microfluidics." In: *Microscale Acoustofluidics.* The Royal Society of Chemistry, Dec. 2014, 1–28. ISBN: 9781849737067.

Part III.
Appendices

A. Software

This thesis primarily uses three pieces of software. *FluidX3D*, *noFluidX3D* and the python package used for evaluation. Details on these are provided here.

A.1. FluidX3D

FluidX3D is the main simulation software used in this thesis. It is written in C++ and OpenCL C. The full *FluidX3D* source code is available on the GitHub account "Richardk2n" or upon inquiry. In the following, its history, versioning scheme and basic usage are covered.

A.1.1. History

FluidX3D is originally based on the book by Krüger *et al.* [9]. It was developed under the supervision of Prof. Dr. Stephan Gekle by Moritz Lehmann and Fabian Häusl. Partially separate branches were maintained by the two developers. When Moritz Lehmann finishes his doctorate, the published a partial rewrite on GitHub. The original source code remained with the group of Prof. Dr. Stephan Gekle. For the last four years (duration of this thesis), the role of maintainer went to the author of the present work. *FluidX3D* was subsequently majorly overhauled. The branches, which were mostly different by the datatypes used were brought back together, with variable datatypes. A new build system was introduced, dependency handling was implemented, the code was organized, the C++ standard updated and the usability improved. Both unit tests and integration tests were added. Contributions to the OpenCL libraries were made to allow splitting of the kernels. Some unused code was removed. Several new features were developed (See the changelog in the repository). Others were ported. Several core systems have been completely rewritten. Many optimizations have been removed to improve speed. Timm Krüger was asked for implementation details, however no access to his code was granted. Furthermore, the source code of another LBM simulation software (*ESPResSo* [49, 50]) was used as a reference for the implementation of *FluidX3D*. The amount of usability improvements is hard to overstate. A great focus was placed on extensibility. As it stands now, little code remains from four years ago and the software as a whole has completely changed. At this point, *FluidX3D* in the version provided in this thesis and used by the group of Prof. Dr. Stephan Gekle can be considered to be part of this thesis. The previous authors deserve some credit though, as they are the ones, who got it started.

A.1.2. Versioning

The version of *FluidX3D* is separated into major, minor and patch version. It does not quite follow semantic versioning, as the absence of breaking changes can never be guaranteed. The majority of simulations in this thesis were performed using *FluidX3D* version `v0.4.3`. The current setups enforce a compatible *FluidX3D* version. It should however be noted, that not every hotfix is assigned a new version. Consequently, `v0.4.3` should not be interpreted as the commit with this tag, but includes the hotfixes shortly after. *FluidX3D* is continually validated with unit tests and integration tests to assure validity of the simulations even between version changes. Its current code quality can be viewed as an early beta stage. Extended documentation is available in the repository.

A.1.3. Basic usage

In order to reproduce any simulation in this thesis, perform the following steps. First obtain a copy of the repository. Assure you have a current version of *cmake* and *gcc* installed. The following assumes, that the copy of the repository is placed in a folder called `fluidx3d`. Open a POSIX-complaint shell at the storage location of the folder and type the following.

```
cd fluidx3d
mkdir build
cd build
cmake ..
```

This enters the code directory and creates a folder for the compilation process. This folder is entered too, and the build environment is configured. Next, place the setup files from the corresponding simulation box into `fluidx3d/src/setup` overwriting any present files. Afterwards, enter the following commands.

```
cmake --build . -- -j8
cmake --install .
cd ../install
```

This compiles the software using 8 CPU cores. You should use more if you have them available. Afterwards, the compiled binary is installed to the `fluidx3d/src/install` folder. This folder is entered. From here *FluidX3D* can be started using the following command.

```
./FluidX3D
```

Configuration files are passed as command-line arguments. More advanced usage is beyond the scope of this thesis. For further insight read the `README.md` file in the repository. When in doubt create an issue and ask. Note, that the *FluidX3D* setups used in this thesis require a GPU with fp64 support. This is notably lacking from integrated GPUs. *FluidX3D* is developed for the use with Linux, it may work with Windows. Some of the setups require more VRAM than is available on a typical GPU.

A.2. noFluidX3D

The code used for the AFM simulations was named *noFluidX3D* by Sebastian Wohlrab, who was originally tasked to create this for his master's thesis [93]. Compared to *FluidX3D*, it does not simulate the fluid, explaining its name. He was directed to reuse as much of *FluidX3D* code as possible. This was done via addressing the same OpenCL kernels using Python. Still *noFluidX3D* contains more models. Only the parts of the code used in the present work and the associated publications have been thoroughly validated. Those parts of the code have been made portable and have been partially rewritten to comply with best practices. The rest of the code remains unmaintained. Therefore, the code is in no state, that would allow for publication. Still, the *noFluidX3D* source is available on the GitHub account "Richardk2n" or upon reasonable request. It should not be used for any use case outside the simulations in this thesis and the aforementioned publications. *noFluidX3D* is working towards the same versioning scheme as *FluidX3D*. It installs like any other python package. It can be used by executing the setup files within a python environment, which has *noFluidX3D* installed.

A.3. Evaluation scripts

A python package, which is also called fluidx3d was created to contain all scripts and functionality used for the evaluation of *FluidX3D* and *noFluidX3D* simulations. It currently has no versioning scheme. Some functionality only works with current simulation output. It installs like any other python package. The basic usage can be gathered from the evaluation scripts. The code is available on the GitHub account "Richardk2n" under the name "fluidx3d-python".

B. TRT and MRT relaxation

The TRT and MRT relaxation schemes for LBM (see section 2) are more complicated schemes, with the aim to improve on SRT. Their concrete implementation shall be omitted here, as it was not done as part of this thesis. For more detail on this, consider reading reference [72]. TRT offers one additional parameter for tuning, while MRT offers many. For TRT, it can be shown, that specific choices for the tuning parameter minimize specific parts of the error. This can be particularly useful to improve accuracy in low Reynolds number simulations, as the ones performed in this thesis. MRT has many “magic” parameters, with hard to determine effects. It has been shown to improve accuracy in high Reynolds number simulations [119]. The issue with tuning parameters is, that they are not useful in unknown territory. They can be tuned to analytically solvable problems, to improve accuracy. This looks good, but in the simulations discussed in this thesis, the problems and their solutions are unknown, so there is no way of validating the tuning. The unknown benefit (or even detriment) comes with an increased simulation cost. This is therefore obviously not done in this thesis. It should be recommended to stick with SRT. Notably, in much of literature MRT is treated as some kind of magic bullet, to the point of reviewers even demanding its use. For the low Reynolds number simulations discussed in the present works, it is not expected to offer any benefit [9, 15, 52, 119]. Testing for this thesis showed a slightly worsened stability.

C. LBM algorithm

The concrete implementation of the LBM algorithm is of concern to stability. The LBM can be easily parallelized as is done in to vastly shorten runtime. Legacy code typically runs on large CPU clusters, *FluidX3D* utilizes GPUs. In both cases some care needs to be taken to avoid race conditions. Every thread only knows about the values it modifies and may not touch any values another thread may modify. The LBM equation (eq. (22)) can be separated into its two steps (collision and streaming). This is purely conceptual, but important for the discussion of the simulation, as these two steps cannot be done in place simultaneously. With these steps separated, the equations read as follows.

$$f_i(\vec{x},t)^* = f_i(\vec{x},t) + \Omega_i(\vec{x},t) \tag{376}$$

$$f_i(\vec{x}+\vec{c}_i\Delta t, t+\Delta t) = f_i(\vec{x},t)^* \tag{377}$$

Here $f_i(\vec{x},t)^*$ refers to the post-collision populations. The collision step (eq. (376)) is straight forward and of no concern as it only operates on the local populations. However, the streaming (eq. (377)) can be interpreted two different ways. Either it reads the local populations, and writes to the neighboring, or it reads from the neighboring populations and writes the local ones. In both cases it modifies populations, that might still need to be read by other concurrently executing threads. To avoid this issue a second buffer is used to do the writing. After every LBM step, all threads a synchronized, meaning the execution waits for all nodes to be calculated. Then the buffers are switched and only after that the computation of the next step is started. It is of course beneficial to bundle the computation of the streaming and collision steps. Given the unproblematic nature of the collision step this is possible - with a constraint. If one decides to write to the neighboring populations (`PUSH`), the collision needs to be calculated before the streaming as the local populations are not known after the streaming to the thread doing the streaming. If one decides to read from the neighboring populations (`PULL`), the collision needs to be calculated after the streaming as the local populations are overridden by the streaming. This establishes a clear order for the LBM, depending on this pick. The two options are illustrated by the pythonic pseudocode in figure 135 below.

```
def lbmStepPUSH():
    doCollide()
    doStreamingPUSH()
    synchronize()

def lbmStepPULL():
    doStreamingPULL()
    doCollide()
    synchronize()
```

Figure 135: Pythonic pseudocode of the command order in `PUSH` vs `PULL`.

For pure LBM this is completely fine, as either is just alternating between stream and collide. In literature, it is generally argued, that `PULL` is better as it is faster [9, 47, 72, 120]. However, none of these actually demonstrates this, opting for arguing about synthetic benchmarks instead. Numbers from LBM benchmarks can be found for CPUs [101]. These show slightly slower performance for `PULL`. From the experience gathered in this thesis, it is irrelevant performance wise. If there is a difference, it is slower than run to run variance. However, this choice actually changes what gets stored in the aforementioned buffers. `PULL` stores $f_i(\vec{x}, t)^*$, while `PUSH` stores $f_i(\vec{x}, t)$. If one now wishes to output the velocity after every time-step it needs to be calculated from the stored populations. Meaning this output would actually vary (slightly) depending on which scheme is chosen. This does not matter for pure LBM as the velocity at this point does not influence the simulation. However, extensions need the velocity and are therefore influenced by this choice. During work on this thesis, it was found, that some cases of IBM interact badly with `PULL` (see section 15.1) and therefore this thesis uses `PUSH`. As recalculating the velocity at the right time in the correct way has proven to be important, the LBM algorithm used in *FluidX3D* is presented in the following using pythonic pseudocode with an emphasis on this detail.

```
1  def lbmStep(nodeIndex, oldPopulations, newPopulations, density,
2              velocity, forceOnLatticePoint, stressAtLatticePoint):
3      localPops = oldPopulations[nodeIndex]
4      # according to equations (23) and (24)
5      localDensity, localVelocity = calculateDensityVelocity(localPops)
6      # according to equation (41)
7      forcingTerm = calculateForcing(localVelocity, forceOnLatticePoint)
8      # according to equation (42)
9      localVelocity += 0.5/localDensity*forceOnLatticePoint
10     # according to equation (20)
11     feq = calculateEquilibrium(localDensity, localVelocity)
12     # according to equation (45)
13     feq -= calculateStressContribution(stressAtLatticePoint)
14     # collision (tau is the relaxation time $\tau_r$)
15     postCollisionPopulations = ((1-1/tau)*localPops
16                                 + 1/tau*feq
17                                 + (1-0.5/tau)*forcingTerm)
18     # Handles boundary conditions as desribed in section 13
19     streamToNeighborsHandlingBoundaries(postCollisionPopulations,
20                                         newPopulations)
21     synchronize()
22
23     # Crucial update from the new pops (same as above)
24     localPops = newPopulations[nodeIndex]
25     localDensity, localVelocity = calculateDensityVelocity(localPops)
26     localVelocity += 0.5/localDensity*forceOnLatticePoint
27     density[nodeIndex] = localDensity
28     velocity[nodeIndex] = localVelocity
29     synchronize()
30
31     # Advect and calculate new force from IBM (see section 6),
32     # viscoelastic fluids (see section 4) ...
33     calculateExtentions()
```

Figure 136: Pythonic pseudocode of the complete LBM algorithm.

The recalculation of the velocity (lines 24 to 28 in figure 136) might seem like a waste of resources, but if the extensions are used, it is required to maintain stability.

D. Unified viscoelastic fluid model notation

In the following, the constitutive equations for the implemented models shall be brought to the unified notation (equation (75)). For this the following relations are useful. The upper-convected derivative is a linear operator.

$$\overset{\triangledown}{(k\underline{A})} = k\overset{\triangledown}{\underline{A}} \tag{378}$$

$$\overset{\triangledown}{(\underline{A}+\underline{B})} = \overset{\triangledown}{\underline{A}} + \overset{\triangledown}{\underline{B}} \tag{379}$$

For any arbitrary constant k and tensors $\underline{A}$ and $\underline{B}$. The upper-convected time derivative of the unit tensor relates to the strain-rate tensor.

$$\overset{\triangledown}{\mathbb{1}} = -2\underline{D} \tag{380}$$

D.1. Oldroyd-B

Dividing the constitutive equation (equation (76)) by the constant $\frac{\eta_\mathrm{p}}{\lambda}$ directly yields the unified notation as follows.

$$\overset{\triangledown}{\left(\frac{\lambda}{\eta_\mathrm{p}}\underline{\tau}\right)} = -\frac{\lambda}{\eta_\mathrm{p}}\frac{\underline{\tau}}{\lambda} + 2\underline{D} \tag{381}$$

One can identify the polymer-conformation tensor and the remaining source term as follows.

$$\underline{\tau} = \frac{\eta_\mathrm{p}}{\lambda}\underline{C} \tag{382}$$

$$\underline{S}_\mathrm{R} = -\frac{\underline{C}}{\lambda} \tag{383}$$

D.2. FENE-P

Bringing FENE-P to standard notation is considerably more work. First start from the original constitutive equation. For this the notation from Oliveira *et al.* [57] is used. This differs from the original from Bird [59, 60] by the sign of $\underline{\tau}$.

$$Z'\underline{\tau} + \lambda'\overset{\triangledown}{\underline{\tau}} - \lambda'\left(\underline{\tau} + (1-\epsilon b)\frac{\eta'_\mathrm{p}}{\lambda'}\mathbb{1}\right)\frac{\mathrm{d}\ln Z'}{\mathrm{d}t} = 2(1-\epsilon b)\eta'_\mathrm{p}\underline{D} \tag{384}$$

$$Z' = 1 + \frac{3}{b}\left[(1-\epsilon b) + \frac{\mathrm{Tr}\,\underline{\tau}}{3\frac{\eta'_\mathrm{p}}{\lambda'}}\right] \tag{385}$$

Here, the relation $nk_\mathrm{B}\theta = \frac{\eta'_\mathrm{p}}{\lambda'}$ has already been used. The primes on η'_p, λ' and Z' are added, because a slightly different definition of these variables is more consistent with the other models presented in this thesis. Therefore, they are primed here, and the consistent definition found later will not have primes. ϵ was originally set to 0 by Bird *et* *al.*, but shortly afterwards revised to $\epsilon = \frac{2}{b(b+2)}$. The original version can still often be found in literature. Therefore, both cases will be treated in the following. To do this one can define $\hat{b}$ as follows.

$$\hat{b} = \begin{cases} b & \text{, for } \epsilon = 0 \\ b+2 & \text{, for } \epsilon = \dfrac{2}{b(b+2)} \end{cases} \tag{386}$$

This allows to simplify the terms containing ϵ as follows.

$$(1 - \epsilon b) = \frac{b}{\hat{b}} \tag{387}$$

This yields the following.

$$Z' \underline{\tau} + \lambda' \overset{\nabla}{\underline{\tau}} - \lambda' \left(\underline{\tau} + \frac{b}{\hat{b}} \frac{\eta'_\mathrm{p}}{\lambda'} \mathbb{1} \right) \frac{\mathrm{d} \ln Z'}{\mathrm{d}t} = 2 \frac{b}{\hat{b}} \eta'_\mathrm{p} \underline{D} \tag{388}$$

$$Z' = 1 + \frac{3}{b} \left[\frac{b}{\hat{b}} + \frac{\mathrm{Tr}\, \underline{\tau}}{3 \frac{\eta'_\mathrm{p}}{\lambda'}} \right] \tag{389}$$

The most problematic term, both in terms of implementation and in terms of unified notation is the logarithm derivative. To remove this, first define some dimensionless stress $\underline{\tau}'$ to simplify the term as follows.

$$\frac{\eta'_\mathrm{p}}{\lambda'} \underline{\tau}' = \underline{\tau} + \frac{b}{\hat{b}} \frac{\eta'_\mathrm{p}}{\lambda'} \mathbb{1} \tag{390}$$

$$\underline{\tau} = \frac{\eta'_\mathrm{p}}{\lambda'} \left(\underline{\tau}' - \frac{b}{\hat{b}} \mathbb{1} \right) \tag{391}$$

This definition is similar to the polymer feedback stress introduced by Gupta *et al.* [28]. Inserting this definition into the constitutive equation yields the following.

$$Z' \frac{\eta'_\mathrm{p}}{\lambda'} \left(\underline{\tau}' - \frac{b}{\hat{b}} \mathbb{1} \right) + \lambda' \frac{\eta'_\mathrm{p}}{\lambda'} \overset{\nabla}{\left(\underline{\tau}' - \frac{b}{\hat{b}} \mathbb{1} \right)} - \lambda' \frac{\eta'_\mathrm{p}}{\lambda'} \underline{\tau}' \frac{\mathrm{d} \ln Z'}{\mathrm{d}t} = 2 \frac{b}{\hat{b}} \eta'_\mathrm{p} \underline{D} \tag{392}$$

$$\frac{Z'}{\lambda'} \left(\underline{\tau}' - \frac{b}{\hat{b}} \mathbb{1} \right) + \overset{\nabla}{\left(\underline{\tau}' - \frac{b}{\hat{b}} \mathbb{1} \right)} - \underline{\tau}' \frac{\mathrm{d} \ln Z'}{\mathrm{d}t} = 2 \frac{b}{\hat{b}} \underline{D} \tag{393}$$

$$\frac{Z'}{\lambda'} \left(\underline{\tau}' - \frac{b}{\hat{b}} \mathbb{1} \right) + \overset{\nabla}{\underline{\tau}'} - \underline{\tau}' \frac{\mathrm{d} \ln Z'}{\mathrm{d}t} = \underline{0} \tag{394}$$

Furthermore, Z' becomes as follows.

$$Z' = 1 + \frac{3}{b} \left[\frac{b}{\hat{b}} + \frac{\mathrm{Tr}\left(\underline{\tau}' - \frac{b}{\hat{b}} \mathbb{1} \right)}{3} \right] \tag{395}$$

$$Z' = 1 + \frac{3}{b} \left[\frac{\mathrm{Tr}\, \underline{\tau}'}{3} \right] \tag{396}$$

$$Z' = 1 + \frac{\mathrm{Tr}\, \underline{\tau}'}{b} \tag{397}$$

From the chain rule follows, the simplification below.

$$\frac{\mathrm{d}\ln Z'}{\mathrm{d}t} = \frac{1}{Z'}\frac{\mathrm{d}Z'}{\mathrm{d}t} \tag{398}$$

This allows the simplification of the constitutive equation to the following.

$$\frac{Z'}{\lambda'}\left(\underline{\tau}' - \frac{b}{\hat{b}}\mathbb{1}\right) + \overset{\triangledown}{\underline{\tau}'} - \underline{\tau}'\frac{1}{Z'}\frac{\mathrm{d}Z'}{\mathrm{d}t} = \underline{0} \tag{399}$$

$$\overset{\triangledown}{\underline{\tau}'} - \underline{\tau}'\frac{1}{Z'}\frac{\mathrm{d}Z'}{\mathrm{d}t} = -\frac{Z'}{\lambda'}\left(\underline{\tau}' - \frac{b}{\hat{b}}\mathbb{1}\right) \tag{400}$$

Assuming $Z' \neq 0$.

$$\lambda'\left(\frac{1}{Z'}\overset{\triangledown}{\underline{\tau}'} - \underline{\tau}'\frac{1}{Z'^2}\frac{\mathrm{d}Z'}{\mathrm{d}t}\right) = -\underline{\tau}' + \frac{b}{\hat{b}}\mathbb{1} \tag{401}$$

Now consider a scaled stress as follows.

$$\underline{\tau}'_{\mathrm{S}} = \frac{\underline{\tau}'}{Z'} \tag{402}$$

This yields the following.

$$\overset{\triangledown}{\underline{\tau}'_{\mathrm{S}}} = \frac{\mathrm{D}\underline{\tau}'_{\mathrm{S}}}{\mathrm{D}t} - \left((\nabla\vec{u})^T \cdot \underline{\tau}'_{\mathrm{S}} + \underline{\tau}'_{\mathrm{S}} \cdot (\nabla\vec{u})\right) \tag{403}$$

$$\overset{\triangledown}{\underline{\tau}'_{\mathrm{S}}} = \frac{\mathrm{D}\frac{\underline{\tau}'}{Z'}}{\mathrm{D}t} - \left((\nabla\vec{u})^T \cdot \frac{\underline{\tau}'}{Z'} + \frac{\underline{\tau}'}{Z'} \cdot (\nabla\vec{u})\right) \tag{404}$$

$$\overset{\triangledown}{\underline{\tau}'_{\mathrm{S}}} = \frac{1}{Z'}\frac{\mathrm{D}\underline{\tau}'}{\mathrm{D}t} - \frac{\underline{\tau}'}{Z'^2}\frac{\mathrm{d}Z'}{\mathrm{d}t} - \frac{1}{Z'}\left((\nabla\vec{u})^T \cdot \underline{\tau}' + \underline{\tau}' \cdot (\nabla\vec{u})\right) \tag{405}$$

$$\overset{\triangledown}{\underline{\tau}'_{\mathrm{S}}} = \frac{1}{Z'}\left[\frac{\mathrm{d}\underline{\tau}'}{\mathrm{d}t} - \left((\nabla\vec{u})^T \cdot \underline{\tau}' + \underline{\tau}' \cdot (\nabla\vec{u})\right)\right] - \frac{\underline{\tau}'}{Z'^2}\frac{\mathrm{d}Z'}{\mathrm{d}t} \tag{406}$$

$$\overset{\triangledown}{\underline{\tau}'_{\mathrm{S}}} = \frac{1}{Z'}\overset{\triangledown}{\underline{\tau}'} - \frac{\underline{\tau}'}{Z'^2}\frac{\mathrm{d}Z'}{\mathrm{d}t} \tag{407}$$

This replaces the bracket in the constitutive equation (eq. (401)) as follows.

$$\lambda'\overset{\triangledown}{\underline{\tau}'_{\mathrm{S}}} = -Z'\underline{\tau}'_{\mathrm{S}} + \frac{b}{\hat{b}}\mathbb{1} \tag{408}$$

$$Z' = \frac{b}{b - \operatorname{Tr}\underline{\tau}'_{\mathrm{S}}} \tag{409}$$

At this point, the non-trivial time derivative of Z' is completely gone. However, in the process the stress got scaled and incurred an offset. The implementation requires, that the polymer-conformation tensor must be $\underline{0}$ for $\underline{\tau} = \underline{0}$. So another variable is introduced as follows.

$$\underline{C}' = \frac{\underline{\tau}'_{\mathrm{S}}}{\tau_0} \tag{410}$$

With the following definition of τ_0.

$$\tau_0 \mathbb{1} = \underline{\tau}'_{\mathrm{S}}(\underline{\tau} = \underline{0}) \tag{411}$$

$$= \frac{b}{\hat{b}} \mathbb{1} \frac{\hat{b}}{\hat{b} + 3} \tag{412}$$

$$= \frac{b}{\hat{b} + 3} \mathbb{1} \tag{413}$$

Yielding the following equation for the constitutive equation.

$$\lambda' \frac{b}{\hat{b} + 3} \overset{\triangledown}{\underline{C}'} = -Z' \frac{b}{\hat{b} + 3} \underline{C}' + \frac{b}{\hat{b}} \mathbb{1} \tag{414}$$

$$\lambda' \frac{\hat{b}}{\hat{b} + 3} \overset{\triangledown}{\underline{C}'} = -Z' \frac{\hat{b}}{\hat{b} + 3} \underline{C}' + \mathbb{1} \tag{415}$$

And the following.

$$Z' = \frac{b}{b - \frac{b}{\hat{b}+3} \operatorname{Tr} \underline{C}'} \tag{416}$$

$$= \frac{\hat{b} + 3}{\hat{b} + 3 - \operatorname{Tr} \underline{C}'} \tag{417}$$

Defining $\lambda = \lambda' \frac{\hat{b}}{\hat{b}+3}$ and inserting Z' yields the following.

$$\lambda \overset{\triangledown}{\underline{C}'} = -\frac{\hat{b}}{\hat{b} + 3 - \operatorname{Tr} \underline{C}'} \underline{C}' + \mathbb{1} \tag{418}$$

Now, the polymer conformation tensor can finally be defined by removing the $\mathbb{1}$ offset as follows.

$$\underline{C} = \underline{C}' - \mathbb{1} \tag{419}$$

With this, the constitutive equation becomes as follows.

$$\lambda \left(\overset{\triangledown}{\underline{C}} - 2\underline{D} \right) = -\frac{\hat{b}}{\hat{b} - \operatorname{Tr} \underline{C}} (\underline{C} + \mathbb{1}) + \mathbb{1} \tag{420}$$

$$\lambda \left(\overset{\triangledown}{\underline{C}} - 2\underline{D} \right) = -\frac{\hat{b}\underline{C} + \operatorname{Tr} \underline{C} \mathbb{1}}{\hat{b} - \operatorname{Tr} \underline{C}} \tag{421}$$

$$\overset{\triangledown}{\underline{C}} = -\frac{\hat{b}\underline{C} + \operatorname{Tr} \underline{C} \mathbb{1}}{\lambda \left(\hat{b} - \operatorname{Tr} \underline{C} \right)} + 2\underline{D} \tag{422}$$

Finally, the constitutive equation in the unified notation is found. According to the derivation Z' may not be zero. Considering b should be positive from the physical interpretation of the variable, Z' cannot be zero as can be seen from the following.

$$Z' = \frac{\hat{b} + 3}{\hat{b} - \operatorname{Tr} \underline{C}} =: \frac{\hat{b} + 3}{\hat{b}} Z \tag{423}$$

Here a scaled Z, defined as follows has been introduced.

$$Z = \frac{\hat{b}}{\hat{b} - \operatorname{Tr} \underline{C}} \tag{424}$$

In total, the polymer-conformation tensor and the remaining source term can be identified as follows.

$$\underline{\tau} = \frac{\eta_\mathrm{p}'}{\lambda} \frac{b}{\hat{b}+3} \left(\frac{\hat{b}\underline{C} + \operatorname{Tr} \underline{C} \mathbb{1}}{\hat{b} - \operatorname{Tr} \underline{C}} \right) \tag{425}$$

$$= \frac{\eta_\mathrm{p}}{\lambda} \left(\frac{\hat{b}\underline{C} + \operatorname{Tr} \underline{C} \mathbb{1}}{\hat{b} - \operatorname{Tr} \underline{C}} \right) = -\eta_\mathrm{p} \underline{S}_\mathrm{R} \tag{426}$$

$$\underline{S}_\mathrm{R} = -\frac{\hat{b}\underline{C} + \operatorname{Tr} \underline{C} \mathbb{1}}{\lambda \left(\hat{b} - \operatorname{Tr} \underline{C} \right)} \tag{427}$$

Here the relation for η_p has been introduced as follows.

$$\eta_\mathrm{p} = \eta_\mathrm{p}' \frac{b}{\hat{b}+3} \tag{428}$$

The unified notation reveals a stability condition, that is hidden by the original notation. The trace of the polymer conformation tensor may not reach $\hat{b}$, which relates to the extensibility being finite in this model. This causes issues discussed in appendix E. The original version of the constitutive equations can also be simply written using the new variables as follows.

$$\frac{Z}{\lambda} \underline{\tau} + \overset{\triangledown}{\underline{\tau}} - \left(\underline{\tau} + \frac{\eta_\mathrm{p}}{\lambda} \mathbb{1} \right) \frac{1}{Z} \frac{\mathrm{d}Z}{\mathrm{d}t} = 2 \frac{\eta_\mathrm{p}}{\lambda} \underline{D} \tag{429}$$

$$Z = 1 + \frac{\lambda}{\eta_\mathrm{p}} \frac{\operatorname{Tr} \underline{\tau}}{\hat{b}+3} \tag{430}$$

The logarithm derivative has been replaced for neatness' sake. The appearance of $\hat{b}$ has notably been contained to a singular location. The usefulness of these new variables does however more clearly reveal itself in the shear-flow solutions (see appendix F.2.2). The simulation algorithm also requires the conversion from $\underline{\tau}$ to $\underline{C}$, which is non-trivial for FENE-P. To do this one starts from the trace of equation (426) as follows.

$$\operatorname{Tr}\underline{\tau} = \frac{\eta_\mathrm{p}}{\lambda}\left(\frac{\hat{b}\operatorname{Tr}\underline{C} + 3\operatorname{Tr}\underline{C}}{\hat{b} - \operatorname{Tr}\underline{C}}\right) \tag{431}$$

$$\operatorname{Tr}\underline{\tau} = \frac{\eta_\mathrm{p}}{\lambda}\left(\frac{\hat{b}+3}{\hat{b} - \operatorname{Tr}\underline{C}}\right)\operatorname{Tr}\underline{C} \tag{432}$$

$$\hat{b}\operatorname{Tr}\underline{\tau} - \operatorname{Tr}\underline{C}\operatorname{Tr}\underline{\tau} = \frac{\eta_\mathrm{p}}{\lambda}\left(\hat{b}+3\right)\operatorname{Tr}\underline{C} \tag{433}$$

$$\hat{b}\operatorname{Tr}\underline{\tau} = \left[\frac{\eta_\mathrm{p}}{\lambda}\left(\hat{b}+3\right) + \operatorname{Tr}\underline{\tau}\right]\operatorname{Tr}\underline{C} \tag{434}$$

$$\frac{\hat{b}\operatorname{Tr}\underline{\tau}}{\frac{\eta_\mathrm{p}}{\lambda}\left(\hat{b}+3\right) + \operatorname{Tr}\underline{\tau}} = \operatorname{Tr}\underline{C} \tag{435}$$

This equation for $\operatorname{Tr}\underline{C}$ is reinserted into equation (426) as follows.

$$\underline{\tau} = \frac{\eta_\mathrm{p}}{\lambda}\left[\frac{\hat{b}\underline{C} + \frac{\hat{b}\operatorname{Tr}\underline{\tau}}{\frac{\eta_\mathrm{p}}{\lambda}\left(\hat{b}+3\right)+\operatorname{Tr}\underline{\tau}}\mathbb{1}}{\hat{b} - \frac{\hat{b}\operatorname{Tr}\underline{\tau}}{\frac{\eta_\mathrm{p}}{\lambda}\left(\hat{b}+3\right)+\operatorname{Tr}\underline{\tau}}}\right] \tag{436}$$

$$\underline{\tau} = \frac{\eta_\mathrm{p}}{\lambda}\left\{\frac{\left[\frac{\eta_\mathrm{p}}{\lambda}\left(\hat{b}+3\right) + \operatorname{Tr}\underline{\tau}\right]\underline{C} + \operatorname{Tr}\underline{\tau}\mathbb{1}}{\frac{\eta_\mathrm{p}}{\lambda}\left(\hat{b}+3\right) + \operatorname{Tr}\underline{\tau} - \operatorname{Tr}\underline{\tau}}\right\} \tag{437}$$

$$\underline{\tau}\frac{\eta_\mathrm{p}}{\lambda}\left(\hat{b}+3\right) = \frac{\eta_\mathrm{p}}{\lambda}\left\{\left[\frac{\eta_\mathrm{p}}{\lambda}\left(\hat{b}+3\right) + \operatorname{Tr}\underline{\tau}\right]\underline{C} + \operatorname{Tr}\underline{\tau}\mathbb{1}\right\} \tag{438}$$

$$\underline{\tau}\left(\hat{b}+3\right) = \left[\frac{\eta_\mathrm{p}}{\lambda}\left(\hat{b}+3\right) + \operatorname{Tr}\underline{\tau}\right]\underline{C} + \operatorname{Tr}\underline{\tau}\mathbb{1} \tag{439}$$

$$\frac{\underline{\tau}\left(\hat{b}+3\right) - \operatorname{Tr}\underline{\tau}\mathbb{1}}{\frac{\eta_\mathrm{p}}{\lambda}\left(\hat{b}+3\right) + \operatorname{Tr}\underline{\tau}} = \underline{C} \tag{440}$$

This non-trivial relation means, that using $\underline{C}$ for analytical solutions is not as convenient as for the other models. It can be seen, that Oldroyd-B is reproduced for $\hat{b} \to \infty$.

D.3. PTT

Dividing the constitutive equation (equation (95)) by the constant $\frac{\eta_\mathrm{p}}{\lambda}$ directly yields the unified notation as follows.

$$\overset{\nabla}{\left(\frac{\lambda}{\eta_\mathrm{p}}\underline{\tau}\right)} = -\frac{\lambda}{\eta_\mathrm{p}}\xi(\underline{\tau}\cdot\underline{D} + \underline{D}\cdot\underline{\tau}) - e^{\frac{\epsilon\lambda}{\eta_\mathrm{p}}\operatorname{Tr}\underline{\tau}}\frac{\lambda}{\eta_\mathrm{p}}\frac{\underline{\tau}}{\lambda} + 2\underline{D} \tag{441}$$

One can identify the polymer-conformation tensor and the remaining source term as follows.

$$\underline{\tau} = \frac{\eta_{\mathrm{p}}}{\lambda}\underline{C} \tag{442}$$

$$\underline{S}_{\mathrm{R}} = -\xi(\underline{C}\cdot\underline{D} + \underline{D}\cdot\underline{C}) - e^{\epsilon\,\mathrm{Tr}\,\underline{C}}\frac{\underline{C}}{\lambda} \tag{443}$$

It can be seen, that Oldroyd-B is reproduced for $\xi = \epsilon = 0$. It should be noted, that this thesis only deals with the case $\xi = 0$. This is done, because $\xi \neq 0$ prevents analytical solutions to be found for the viscosity (see appendix F.2.3). Semi-analytic approximation did yield $\xi \ll 1$ for the fluids discussed in this thesis. So the approximation $\xi = 0$ is reasonable for the fluids discussed here.

E. Issues arising from the usage of FENE-P

FENE-P is popular, but proved difficult to use when the Weissenberg number approaches or even exceeds unity. The following will elaborate on the reason behind this. Consider the standard notation for viscoelastic fluids in this thesis (eq. (75)).

$$\overset{\nabla}{\underline{C}} = \underline{S}_{\mathrm{R}} + 2\underline{D} \tag{444}$$

$$\frac{\partial \underline{C}}{\partial t} + \vec{u} \cdot \nabla \underline{C} = \left((\nabla \vec{u})^{\mathrm{T}} \cdot \underline{C} + \underline{C} \cdot (\nabla \vec{u}) \right) + \underline{S}_{\mathrm{R}} + 2\underline{D} \tag{445}$$

Now consider a pure shear-flow as follows.

$$\vec{u} = \dot{\gamma} y \hat{e}_x \tag{446}$$

This gives the following for the constitutive equation.

$$\frac{\partial \underline{C}}{\partial t} + \vec{u} \cdot \nabla \underline{C} = \dot{\gamma} \begin{pmatrix} 2c_{12} & c_{22} & c_{23} \\ c_{22} & 0 & 0 \\ c_{32} & 0 & 0 \end{pmatrix} + \underline{S}_{\mathrm{R}} + 2\underline{D} \tag{447}$$

Translational symmetry along x.

$$\frac{\partial \underline{C}}{\partial t} = \dot{\gamma} \begin{pmatrix} 2c_{12} & c_{22} & c_{23} \\ c_{22} & 0 & 0 \\ c_{32} & 0 & 0 \end{pmatrix} + \underline{S}_{\mathrm{R}} + 2\underline{D} \tag{448}$$

In steady state, the 22 component yields the following.

$$c_{22} = -\frac{\operatorname{Tr} \underline{C}}{\hat{b}} \tag{449}$$

And so does the 33 component. Therefore, the following must hold.

$$c_{11} = a + 2\frac{\operatorname{Tr} \underline{C}}{\hat{b}} \tag{450}$$

In steady state, the 11 component yields the following equation.

$$c_{12} = \frac{\hat{b} c_{11} + \operatorname{Tr} \underline{C}}{\lambda \left(\hat{b} - \operatorname{Tr} \underline{C} \right)} \tag{451}$$

The 12 component yields the following.

$$c_{12} = \frac{\lambda \left(\hat{b} - \operatorname{Tr} \underline{C} \right)}{\hat{b} \dot{\gamma} (c_{22} + 1)} \tag{452}$$

Combining equations (451) and (452) and inserting equation (450) yields the following.

$$\frac{\frac{\operatorname{Tr}\underline{C}}{\hat{b}}}{\left(1-\frac{\operatorname{Tr}\underline{C}}{\hat{b}}\right)^3} = \frac{4}{27}Wi^2 \tag{453}$$

Solving this akin to appendix F.2.2 using reference [121] yields the following.

$$\operatorname{Tr}\underline{C} = \hat{b} - \hat{b}\frac{4Wi^2+9}{2Wi^2}\sinh\left(\frac{1}{3}\operatorname{arcsinh}\left(\frac{1}{Wi^2}\left(\frac{9Wi^2}{4Wi^2+9}\right)^{\frac{3}{2}}\right)\right) \tag{454}$$

From this, it becomes quite clear, that while $\hat{b}$ gives the maximum extensibility, $\operatorname{Tr}\underline{C}$ gives the current extension. With increasing Wi, $\operatorname{Tr}\underline{C}$ approaches $\hat{b}$ in steady-state. Even for moderate Wi (for example $Wi = 5$), the difference is already small ($\frac{\operatorname{Tr}\underline{C}}{\hat{b}} \approx 92\,\%$). For large Wi, the values are near identical. Now consider the non steady-state 22 component (for example) of the constitutive equation. It reads as follows.

$$\frac{\partial c_{22}}{\partial t} = -\frac{\hat{b}c_{22} + \operatorname{Tr}\underline{C}}{\lambda\left(\hat{b} - \operatorname{Tr}\underline{C}\right)} \tag{455}$$

One can see, that this rate diverges if $\hat{b}$ and $\operatorname{Tr}\underline{C}$ approach one another. If $\operatorname{Tr}\underline{C}$ passes $\hat{b}$, this equation turns into exponential growth. This means, that for increasing Wi, $\operatorname{Tr}\underline{C}$ has to approach $\hat{b}$ closer and closer. This has to be done very carefully, because otherwise the simulation breaks. Meanwhile, the approach rate diverges. This makes FENE-P incredibly hard to simulate above small Wi. There exist approaches to combat this issue, for example the decomposition proposed by Dzanic [15]. However, this replaces a diverging rate of change for $\operatorname{Tr}\underline{C}$ with a diverging force, which is also impossible to handle with LBM. This is why FENE-P is not used much in this thesis. Also, these equations highlight, why it is of critical importance for the fit parameter $\hat{b}$ to be positive. Notably, for some of the fluids used in this thesis, it is not.

F. (Semi-)analytical solutions for shear- and Poiseuille flows of fluid models with solvent viscosity

From a simulation standpoint it is convenient to have the velocity profiles for 2D and 3D Poiseuille flow[30]. For some this is possible analytically, some require semi-analytical solutions. To determine the fluid parameters from rheology, the equation for the steady state viscosity in pure shear-flow needs to be known. For the viscoelastic fluids another equation is required for determining the elastic effects. There are multiple options. This thesis uses the first normal stress difference N_1 as it is also steady state, easy to understand and can be measured simultaneously with the viscosity in a rheometer. Solutions for small oscillatory shearing (which would be another option, that can be used instead of N_1) doe exist, but are not presented here. These derivations are a generalized form of the one found in a previous publication (see reference [1]). Several additional results have been added. The basic idea follows Oliveira [122]. The shear-flow as well as 2D and 3D Poiseuille flow case can be solved partially simultaneously. To facilitate this, the coordinate system for the derivations is chosen as follows. The flow direction dictates the x-axis. The center of the channel is picked as the origin. This is consistent with the typical axis orientation in this thesis (see section 2.6) and can be assumed without loss of generality. In three dimensions cylindrical coordinates defined as follows are used (see appendix N.2 for properties).

$$x = x \tag{456}$$

$$y = r\cos\varphi \tag{457}$$

$$z = r\sin\varphi \tag{458}$$

This choice is unusual. It fulfills two purposes. First, it aligns with the typical definition of the coordinate axis in this thesis. Second it allows for the two-dimensional case to emerge as a limit of the three-dimensional case. In the 3D case, one can limit φ to $\varphi \in \{0, \pi\}$. This results in the 2D case. One can see, that for two dimensions, the velocity gradient is aligned along the y-axis. Consequently, in the 2D case $r = |y|$ is defined. Note, that in 2D this requires the sign to be tracked explicitly as follows.

$$\frac{\partial}{\partial r} = \operatorname{sgn}(y)\frac{\partial}{\partial y} \tag{459}$$

The velocity is unidirectional for shear-flow as well as Poiseuille flow. This is dictated by symmetry. Consequently, the following relation holds.

$$\vec{u} = u_x(r)\hat{e}_x \tag{460}$$

The shear-rate $\dot{\gamma}$ is defined as follows.

$$\dot{\gamma}(r) = \frac{\partial u_x}{\partial r} \leq 0 \tag{461}$$

[30]A Poiseuille flow, strictly speaking, is defined for a Newtonian fluid. This thesis uses a broader definition (see appendix P.2)

It is radius dependent for Poiseuille flow and constant for pure shear-flow. Note, that with this, the shear-rate is always negative. To retrieve the velocity profile, some equations, that are independent of the model are needed. These shall be derived first. To derive the velocity problem, one needs to solve the Navier-Stokes equation. Without gravity, it reads as follows.

$$\rho\frac{\mathrm{D}\vec{u}}{\mathrm{D}t} = -\nabla p + \nabla\cdot\underline{\sigma} \tag{462}$$

For a two-dimensional Poieseuille flow in steady state, this reduces to the following.

$$\frac{\partial\sigma_{xr}}{\partial r} = \frac{\partial p}{\partial x} \tag{463}$$

The three-dimensional case is more involved. First consider the total stress as follows.

$$\underline{\sigma} = \underline{\tau} + 2\eta_\mathrm{s}\underline{D} = \underline{\tau} + \eta_\mathrm{s}(\hat{e}_x\otimes\hat{e}_r + \hat{e}_r\otimes\hat{e}_x) \tag{464}$$

As seen above only the σ_{xx}, σ_{xr}, σ_{rx} components of $\underline{\sigma}$ do not vanish. Furthermore, x and φ derivatives of $\underline{\sigma}$ vanish due to symmetry. The following terms of the divergence remain.

$$\nabla\cdot\sigma = \frac{\partial\sigma_{xr}}{\partial r}\hat{e}_x + \frac{\sigma_{xr}}{r}\hat{e}_x \tag{465}$$

Notably, as a result, the reduced Cauchy momentum equation is altered compared to the 2D case.

$$\frac{\partial\sigma_{xr}}{\partial r} + \frac{\sigma_{xr}}{r} = \frac{\partial p}{\partial x} \tag{466}$$

However, the additional term as a convenient form. As a result, both cases are solved by the following.

$$\sigma_{xr} = \frac{\partial p}{\partial x}\frac{r}{2^j} \tag{467}$$

The dimensionality of the problem is handled by j. It is 0 for the two-dimensional flow and 1 in the three-dimensional case. This will be used in most derivations below. In the following the required equations are found for each model.

F.1. Generalized Newtonian fluids

The viscosity curves for Generalized Newtonian fluids are given by their definition. They do not exhibit normal stresses. Consequently, only the velocity profile needs to be discussed here. For completeness' sake, the Newtonian fluid shall be discussed first.

F.1.1. Newtonian

Newtonian solutions are well known. The velocity profile in a pipe is given by the following.

$$\vec{u}(r) = \frac{G}{2^{j+1}\eta}\left(R^2 - r^2\right)\hat{e}_x \tag{468}$$

Here $G = -\frac{\partial p}{\partial x}$ is the pressure gradient, R is the radius of the pipe and $j = 0$ for 2D and unity for 3D. The other models do reproduce this result in certain limits. For Newtonian fluids an analytical solution exists for square channels. The following transforms the solutions found in literature to centered coordinates. This derivation uses a modernized notation from Wikipedia [116] instead of the original from Boussinesq [115]. First, consider the equation for the velocity, where the old coordinates have been primed. Also l has been renamed to w for improved clarity.

$$u_x(y', z') = \frac{G}{2\mu} y'(h - y') - \frac{4Gh^2}{\mu\pi^3} \sum_{n=0}^{\infty} \frac{1}{(2n+1)^3} \frac{\sinh(\beta_n z') + \sinh[\beta_n(w - z')]}{\sinh(\beta_n w)} \sin(\beta_n y') \tag{469}$$

Replace the coordinates as follows.

$$y' = y + \frac{h}{2} \tag{470}$$
$$z' = z + \frac{w}{2} \tag{471}$$

This yields the following.

$$u_x(y, z) = \frac{G}{2\mu}\left(y + \frac{h}{2}\right)\left(h - y - \frac{h}{2}\right) \tag{472}$$
$$- \frac{4Gh^2}{\mu\pi^3} \sum_{n=0}^{\infty} \frac{1}{(2n+1)^3} \frac{\sinh\left[\beta_n\left(z + \frac{w}{2}\right)\right] + \sinh\left[\beta_n\left(w - z - \frac{w}{2}\right)\right]}{\sinh(\beta_n w)} \sin\left[\beta_n\left(y + \frac{h}{2}\right)\right] \tag{473}$$
$$= \frac{G}{2\mu}\left(\frac{h^2}{4} - y^2\right) \tag{474}$$
$$- \frac{4Gh^2}{\mu\pi^3} \sum_{n=0}^{\infty} \frac{1}{(2n+1)^3} \frac{\sinh\left[\beta_n\left(z + \frac{w}{2}\right)\right] + \sinh\left[\beta_n\left(\frac{w}{2} - z\right)\right]}{\sinh(\beta_n w)} \sin\left[\beta_n\left(y + \frac{h}{2}\right)\right] \tag{475}$$

Use $\sinh(-x) = -\sinh x$.

$$= \frac{G}{2\mu}\left(\frac{h^2}{4} - y^2\right) \tag{476}$$
$$- \frac{4Gh^2}{\mu\pi^3} \sum_{n=0}^{\infty} \frac{1}{(2n+1)^3} \frac{\sinh\left[\beta_n\left(z + \frac{w}{2}\right)\right] - \sinh\left[\beta_n\left(z - \frac{w}{2}\right)\right]}{\sinh(\beta_n w)} \sin\left[\beta_n\left(y + \frac{h}{2}\right)\right] \tag{477}$$

Use $\sinh x - \sinh y = 2\sinh\frac{x-y}{2}\cosh\frac{x+y}{2}$.

$$= \frac{G}{2\mu}\left(\frac{h^2}{4} - y^2\right) - \frac{4Gh^2}{\mu\pi^3} \sum_{n=0}^{\infty} \frac{1}{(2n+1)^3} \frac{2\sinh\left(\beta_n\frac{w}{2}\right)\cosh(\beta_n z)}{\sinh(\beta_n w)} \sin\left[\beta_n\left(y + \frac{h}{2}\right)\right] \tag{478}$$

Use $\sinh(2x) = 2\sinh x \cosh x$.

$$= \frac{G}{2\mu}\left(\frac{h^2}{4} - y^2\right) - \frac{4Gh^2}{\mu\pi^3}\sum_{n=0}^{\infty}\frac{1}{(2n+1)^3}\frac{\cosh(\beta_n z)}{\cosh\left(\beta_n \frac{w}{2}\right)}\sin\left[\beta_n\left(y + \frac{h}{2}\right)\right] \tag{479}$$

Use $\sin(x + y) = \sin x \cos y + \cos x \sin y$.

$$= \frac{G}{2\mu}\left(\frac{h^2}{4} - y^2\right) \tag{480}$$

$$- \frac{4Gh^2}{\mu\pi^3}\sum_{n=0}^{\infty}\frac{1}{(2n+1)^3}\frac{\cosh(\beta_n z)}{\cosh\left(\beta_n \frac{w}{2}\right)}\left[\sin(\beta_n y)\cos\left(\beta_n\frac{h}{2}\right) + \cos(\beta_n y)\sin\left(\beta_n\frac{h}{2}\right)\right] \tag{481}$$

Use $\cos\left(\beta_n \frac{h}{2}\right) = \cos\left[\frac{(2n+1)\pi}{h}\frac{h}{2}\right] = \cos\left[(2n+1)\frac{\pi}{2}\right] = -\sin\left(2n\frac{\pi}{2}\right) = -\sin(n\pi) = 0$

$$= \frac{G}{2\mu}\left(\frac{h^2}{4} - y^2\right) - \frac{4Gh^2}{\mu\pi^3}\sum_{n=0}^{\infty}\frac{1}{(2n+1)^3}\frac{\cosh(\beta_n z)}{\cosh\left(\beta_n \frac{w}{2}\right)}\cos(\beta_n y)\sin\left(\beta_n\frac{h}{2}\right) \tag{482}$$

Use $\sin\left(\beta_n \frac{h}{2}\right) = \sin\left[\frac{(2n+1)\pi}{h}\frac{h}{2}\right] = \sin\left[(2n+1)\frac{\pi}{2}\right] = \cos\left(2n\frac{\pi}{2}\right) = \cos(n\pi) = (-1)^n$

$$= \frac{G}{2\mu}\left(\frac{h^2}{4} - y^2\right) - \frac{4Gh^2}{\mu\pi^3}\sum_{n=0}^{\infty}\frac{1}{(2n+1)^3}\frac{\cosh(\beta_n z)}{\cosh\left(\beta_n \frac{w}{2}\right)}\cos(\beta_n y)(-1)^n \tag{483}$$

Similarly, the volume flow rate can be brought to a slightly simpler form as follows.

$$Q = \frac{Gh^3 w}{12\mu} - \frac{16Gh^4}{\mu\pi^5}\sum_{n=0}^{\infty}\frac{1}{(2n+1)^5}\frac{\cosh(\beta_n w) - 1}{\sinh(\beta_n w)} \tag{484}$$

Use $\cosh(2x) = 2\sinh^2 x + 1$.

$$= \frac{Gh^3 w}{12\mu} - \frac{16Gh^4}{\mu\pi^5}\sum_{n=0}^{\infty}\frac{1}{(2n+1)^5}\frac{2\sinh^2\left(\beta_n \frac{w}{2}\right)}{\sinh(\beta_n w)} \tag{485}$$

Use $\sinh(2x) = 2\sinh x \cosh x$.

$$= \frac{Gh^3 w}{12\mu} - \frac{16Gh^4}{\mu\pi^5}\sum_{n=0}^{\infty}\frac{1}{(2n+1)^5}\frac{\sinh\left(\beta_n \frac{w}{2}\right)}{\cosh\left(\beta_n \frac{w}{2}\right)} \tag{486}$$

$$= \frac{Gh^3 w}{12\mu} - \frac{16Gh^4}{\mu\pi^5}\sum_{n=0}^{\infty}\frac{1}{(2n+1)^5}\tanh\left(\beta_n\frac{w}{2}\right) \tag{487}$$

Together, these equations now read as follows.

$$u_x(y,z) = \frac{G}{2\mu}\left(\frac{h^2}{4} - y^2\right) - \frac{4Gh^2}{\mu\pi^3}\sum_{n=0}^{\infty}\frac{1}{(2n+1)^3}\frac{\cosh(\beta_n z)}{\cosh\left(\beta_n \frac{w}{2}\right)}\cos(\beta_n y)(-1)^n \tag{488}$$

$$Q = \frac{Gh^3 w}{12\mu} - \frac{16Gh^4}{\mu\pi^5}\sum_{n=0}^{\infty}\frac{1}{(2n+1)^5}\tanh\left(\beta_n\frac{w}{2}\right) \tag{489}$$

$$\beta_n = \frac{(2n+1)\pi}{h} \tag{490}$$

The sum converges quickly. Given the Newtonian model is not shear-thinning, its Weissenberg number is technically always 0.

$$Wi = 0 \tag{491}$$

F.1.2. CY

There is no analytical solution for the velocity profile, but there are approaches in literature with fully mathematical descriptions [64]. Here a semi-analytical approach will be chosen. The velocity profile derivation starts from equation (467). For CY, the fluid stress is known (eq. (62)), which is inserted into the aforementioned equation to give the following.

$$\eta(\dot\gamma)\dot\gamma = \partial_x p \frac{r}{2^j} \tag{492}$$

One can now use a root-finding algorithm with $r = R$, to retrieve $\dot\gamma$ at the edge of the channel. $\dot\gamma = 0$ at the center from the channel. With that, the range of $\dot\gamma$ is known. From here a large amount of values for $\dot\gamma$ are generated in this interval. They are inserted into equation (492) to retrieve the radii, at which the respective strain-rate is present. This dataset is ideal for numeric integration, because the $\dot\gamma$ or evenly spaced with small steps by design. From there, the velocity profile is retrieved. The Weissenberg number is simply defined as follows.

$$Wi = \lambda\dot\gamma \tag{493}$$

F.1.3. Power law

If a solvent viscosity is present, the same algorithm as was used for CY can be used. In the case where $\eta_s = 0$, an analytical solution exists for the velocity profile. This use-case is popular, and power law fluids are only used in this thesis in relation to literature. Therefore, it is sensible for this derivation to follow the use-case form literature. To derive the velocity profile, start from equation (467) and insert the fluid stress (eq. (64)). This reads as follows.

$$\partial_x p \frac{r}{2^j} = \eta(\dot\gamma)\dot\gamma \tag{494}$$

$$= \frac{\eta_\mathrm{p}}{(\lambda|\dot\gamma|)^P}\dot\gamma \tag{495}$$

Here P is the power model parameter of the power law. Note, that the viscosity cutoff has been dropped to make the solution easier[31]. The absolute value of $\dot{\gamma}$ has to be used here due to the conflicting definition of the sign between the model definition and the rest of the thesis. With $G = -\frac{\partial p}{\partial x}$ one can write the following.

$$G\frac{r}{2^j} = \frac{\eta_\mathrm{p}}{(\lambda|\dot{\gamma}|)^P}|\dot{\gamma}| \tag{496}$$

$$= \frac{\eta_\mathrm{p}}{\lambda^P}|\dot{\gamma}|^{1-P} \tag{497}$$

$$\left(\frac{G\lambda^P}{\eta_\mathrm{p}2^j}r\right)^{\frac{1}{1-P}} = |\dot{\gamma}| \tag{498}$$

$$-\left(\frac{G\lambda^P}{\eta_\mathrm{p}2^j}r\right)^{\frac{1}{1-P}} = \dot{\gamma} \tag{499}$$

Note, that P is typically smaller than unity. Now, the velocity profile can be recovered as follows.

$$\vec{u}(r) = \int_R^r \dot{\gamma}\mathrm{d}r\hat{e}_x \tag{500}$$

$$= \int_r^R \left(\frac{G\lambda^P}{\eta_\mathrm{p}2^j}r\right)^{\frac{1}{1-P}} \mathrm{d}r\hat{e}_x \tag{501}$$

$$= \left(\frac{G\lambda^P}{\eta_\mathrm{p}2^j}\right)^{\frac{1}{1-P}} \int_r^R r^{\frac{1}{1-P}}\mathrm{d}r\hat{e}_x \tag{502}$$

$$= \left(\frac{G\lambda^P}{\eta_\mathrm{p}2^j}\right)^{\frac{1}{1-P}} \frac{1}{e}(R^e - r^e)\hat{e}_x \tag{503}$$

With the exponent $e = \frac{2-P}{1-P}$. One can see, that $P = 0$ describes the Newtonian limit. With the Generalized Newtonian solutions covered, next the viscoelastic models shall be discussed. The Weissenberg number must be unity at the viscosity cutoff. The definition of the model used in this thesis (eq. (64)) makes this equivalent to the following definition.

$$Wi = \lambda\dot{\gamma} \tag{504}$$

F.2. Viscoelastic fluids

The shear-flow derivation for viscoelastic fluids share a common starting point. For this, the following is considered. Due to symmetry, the polymer-conformation tensor can only vary radially. Therefore, the following holds.

$$\vec{u}\cdot\nabla\underline{C} = u_x\frac{\partial}{\partial x}\underline{C} = \underline{0} \tag{505}$$

[31]This solution is given for compatibility with literature, not for its mathematical rigor.

The velocity gradient tensor can be expressed as follows.

$$\nabla\vec{u} = \frac{\partial u_x}{\partial r}\hat{e}_r \otimes \hat{e}_x = \dot{\gamma}\hat{e}_r \otimes \hat{e}_x \tag{506}$$

The strain-rate tensor consequently becomes as follows.

$$\underline{D} = \frac{\dot{\gamma}}{2}(\hat{e}_r \otimes \hat{e}_x + \hat{e}_x \otimes \hat{e}_r) \tag{507}$$

Consider the general formula for $\underline{C}$ as follows.

$$\underline{C} = c_{ij}\hat{e}_i \otimes \hat{e}_j \tag{508}$$

It can be used together with the velocity gradient tensor to find the non-vanishing terms of the upper convected derivative. This is done in steady state in the following.

$$\begin{aligned}(\nabla\vec{u})^{\mathrm{T}} \cdot \underline{C} = \dot{\gamma}\hat{e}_x \otimes \hat{e}_r(&c_{xx}\hat{e}_x \otimes \hat{e}_x + c_{xr}\hat{e}_x \otimes \hat{e}_r + c_{x\varphi}\hat{e}_x \otimes \hat{e}_\varphi \\ &+ c_{rx}\hat{e}_r \otimes \hat{e}_x + c_{rr}\hat{e}_r \otimes \hat{e}_r + c_{r\varphi}\hat{e}_r \otimes \hat{e}_\varphi \\ &+ c_{\varphi x}\hat{e}_\varphi \otimes \hat{e}_x + c_{\varphi r}\hat{e}_\varphi \otimes \hat{e}_r + c_{\varphi\varphi}\hat{e}_\varphi \otimes \hat{e}_\varphi)\end{aligned} \tag{509}$$

With the orthonormality

$$\hat{e}_a \otimes \hat{e}_b\hat{e}_c \otimes \hat{e}_d = \hat{e}_a\hat{e}_b^{\mathrm{T}}\hat{e}_c\hat{e}_d^{\mathrm{T}} = \hat{e}_a(\hat{e}_b \cdot \hat{e}_c)\hat{e}_d^{\mathrm{T}} = \hat{e}_a(\delta_{bc})\hat{e}_d^{\mathrm{T}} = \delta_{bc}\hat{e}_a \otimes \hat{e}_d \tag{510}$$

this can be reduced to the following.

$$(\nabla\vec{u})^{\mathrm{T}} \cdot \underline{C} = \dot{\gamma}(c_{rx}\hat{e}_x \otimes \hat{e}_x + c_{rr}\hat{e}_x \otimes \hat{e}_r + c_{r\varphi}\hat{e}_x \otimes \hat{e}_\varphi) \tag{511}$$

Due to the symmetry of $\underline{C}$, the transpose of this is the second non-vanishing contribution in the upper convected derivative. $\underline{C}$ is symmetric, because $\underline{\tau}$ is (see appendix D). Together with the strain-rate tensor, the complete standard notation of the constitutive equation (see equation (75)) in steady state ($\frac{\partial}{\partial t}\cdot = 0$) reads as follows.

$$\begin{aligned}\underline{0} = &\dot{\gamma}[2c_{rx}\hat{e}_x \otimes \hat{e}_x + c_{rr}(\hat{e}_x \otimes \hat{e}_r + \hat{e}_r \otimes \hat{e}_x) + c_{r\varphi}(\hat{e}_x \otimes \hat{e}_\varphi + \hat{e}_\varphi \otimes \hat{e}_x)] \\ &+ \underline{S}_{\mathrm{R}} + \dot{\gamma}(\hat{e}_r \otimes \hat{e}_x + \hat{e}_x \otimes \hat{e}_r)\end{aligned} \tag{512}$$

Further simplifications depend on the model and will be presented in the following.

F.2.1. Oldroyd-B

For Oldroyd-B the remaining source term is as follows (see appendix D.1).

$$\underline{S}_{\mathrm{R}} = -\frac{\underline{C}}{\lambda} \tag{513}$$

Inserting this into equation (512) yields the following.

$$\begin{aligned}\underline{0} = &[2c_{rx}\hat{e}_x \otimes \hat{e}_x + c_{rr}(\hat{e}_x \otimes \hat{e}_r + \hat{e}_r \otimes \hat{e}_x) + c_{r\varphi}(\hat{e}_x \otimes \hat{e}_\varphi + \hat{e}_\varphi \otimes \hat{e}_x)] \\ &- \frac{\underline{C}}{\lambda\dot{\gamma}} + (\hat{e}_r \otimes \hat{e}_x + \hat{e}_x \otimes \hat{e}_r)\end{aligned} \tag{514}$$

This equation is theoretically separated by components. Simply speaking, this means, that if $\hat{e}_i \otimes \hat{e}_j$ does not appear in this equation $c_{ij} = 0$. This allows some terms to be removed. The remaining ones are as follows.

$$\underline{0} = 2c_{rx}\hat{e}_x \otimes \hat{e}_x - \frac{\underline{C}}{\lambda\dot{\gamma}} + \hat{e}_r \otimes \hat{e}_x + \hat{e}_x \otimes \hat{e}_r \tag{515}$$

The symmetry of $\underline{C}$ also holds in cylindrical coordinates, meaning $c_{rx} = c_{xr}$. With this, the equation reads as follows.

$$\frac{\underline{C}}{\lambda\dot{\gamma}} = 2c_{xr}\hat{e}_x \otimes \hat{e}_x + \hat{e}_r \otimes \hat{e}_x + \hat{e}_x \otimes \hat{e}_r \tag{516}$$

Note, that in the 2D case $c_{xr} = c_{xy}\,\mathrm{sgn}(y) \leq 0$. This equation can be separated by components. This yields three equations. Due to symmetry, two carry the same meaning. The two unique equations are listed below.

$$\frac{c_{xx}}{\lambda\dot{\gamma}} = 2c_{xr} \tag{517}$$

$$\frac{c_{xr}}{\lambda\dot{\gamma}} = 1 \tag{518}$$

Converting to τ using equation (382) yields the following.

$$\tau_{xx} = 2\eta_\mathrm{p}\lambda\dot{\gamma}^2 \tag{519}$$

$$\tau_{xr} = \eta_\mathrm{p}\dot{\gamma} \tag{520}$$

Converting this to the total stress $\underline{\sigma}$ requires the addition of the solvent stress defined as follows.

$$\underline{\sigma}_\mathrm{s} = \eta_\mathrm{s}\dot{\gamma}(\hat{e}_r \otimes \hat{e}_x + \hat{e}_x \otimes \hat{e}_r) \tag{521}$$

This yields the following.

$$\sigma_{xx} = 2\eta_\mathrm{p}\lambda\dot{\gamma}^2 \tag{522}$$

$$\sigma_{xr} = (\eta_\mathrm{p} + \eta_\mathrm{s})\dot{\gamma} \tag{523}$$

As already mentioned above the remaining components of $\underline{\tau}$ or $\underline{\sigma}$ respectively are zero. This yields the first normal stress difference N_1 defined as follows.

$$N_1 = \sigma_{xx} - \left\{\begin{matrix} \sigma_{yy} & \text{, in 2D} \\ \sigma_{rr} & \text{, in 3D} \end{matrix}\right\} = \tau_{xx} = 2\eta_\mathrm{p}\lambda\dot{\gamma}^2 \tag{524}$$

One can see, that the normal stress (as is the case with other elastic effects) is also scaled using the so-called polymer viscosity η_p. This is common among viscoelastic models. The second normal stress difference vanishes due to all contributing components being zero. This can be seen in the following definition.

$$N_2 = \left\{\begin{matrix} \sigma_{yy} - \sigma_{zz} = \tau_{yy} - \tau_{zz} & \text{, in 2D} \\ \sigma_{rr} - \sigma_{\varphi\varphi} = \tau_{rr} - \tau_{\varphi\varphi} & \text{, in 3D} \end{matrix}\right\} = 0 \tag{525}$$

The viscosity defined as follows can similarly be found to be trivial in this model.

$$\eta = \frac{\sigma_{xr}}{\dot{\gamma}} \tag{526}$$

$$= \frac{\tau_{xr} + \eta_{\mathrm{s}}\dot{\gamma}}{\dot{\gamma}} \tag{527}$$

$$= \eta_{\mathrm{p}} + \eta_{\mathrm{s}} \tag{528}$$

With $\sigma_{xr} = \sigma_{xy}\,\mathrm{sgn}(y) \leq 0$ and $\tau_{xr} = \tau_{xy}\,\mathrm{sgn}(y) \leq 0$ in the 2D case analogous to c_{xr}. One can see, that as mentioned before (see section 3.7.2), the Oldroyd-B model is not shear thinning and rather has a constant viscosity. With this, the relevant solutions for shear-flow are covered. It should be noted, that these equations also hold for Poiseuille flow. In the case for Oldroyd-B the Newtonian solution for the flow profile can be used due to the constant viscosity. It reads as follows (see appendix F.1.1).

$$\vec{u}(r) = \frac{G}{2^{j+1}\eta}\left(R^2 - r^2\right)\hat{e}_x \tag{529}$$

Here $G = -\frac{\partial p}{\partial x}$ is the pressure gradient, R is the radius of the pipe and $j = 0$ for 2D and unity for 3D. The fact, that the Oldroyd-B model is not shear-thinning, means, that consequently, the definition of the Weissenberg number must be as follows.

$$Wi = 0 \tag{530}$$

F.2.2. FENE-P

In the following, the solutions for shear rate and Poiseuille flow are found for the FENE-P model.

F.2.2.1. Shear solutions

For FENE-P the remaining source term is as follows (see appendix D.2).

$$\underline{S}_{\mathrm{R}} = -\frac{\underline{\tau}}{\eta_{\mathrm{p}}} \tag{531}$$

Inserting this into equation (512) yields the following.

$$\begin{aligned}\underline{0} = {} & [2c_{rx}\hat{e}_x \otimes \hat{e}_x + c_{rr}(\hat{e}_x \otimes \hat{e}_r + \hat{e}_r \otimes \hat{e}_x) + c_{r\varphi}(\hat{e}_x \otimes \hat{e}_\varphi + \hat{e}_\varphi \otimes \hat{e}_x)] \\ & - \frac{\underline{\tau}}{\eta_{\mathrm{p}}\dot{\gamma}} + (\hat{e}_r \otimes \hat{e}_x + \hat{e}_x \otimes \hat{e}_r)\end{aligned} \tag{532}$$

This equation is theoretically separated by components. Most of the resulting equations only have a term corresponding to the $\underline{\tau}$ term. This is the case if $\hat{e}_i \otimes \hat{e}_j$ does not appear in the above equation. In that case $\tau_{ij} = 0$. This allows some equations to be omitted.

The symmetry of $\underline{C}$ and $\underline{\tau}$ also holds in cylindrical coordinates, meaning $c_{rx} = c_{xr}$. Allowing the duplicate equations to be removed. The remaining ones are as follows.

$$\frac{\tau_{xx}}{\eta_\mathrm{p}\dot{\gamma}} = 2c_{xr} \tag{533}$$

$$\frac{\tau_{xr}}{\eta_\mathrm{p}\dot{\gamma}} = c_{rr} + 1 \tag{534}$$

$$\frac{\tau_{x\varphi}}{\eta_\mathrm{p}\dot{\gamma}} = c_{r\varphi} \tag{535}$$

Notably $\mathrm{Tr}\,\underline{\tau} = \tau_{xx}$ and with equation (440), the following relations can be made.

$$c_{xr} = \frac{\tau_{xr}\left(\hat{b}+3\right)}{\frac{\eta_\mathrm{p}}{\lambda}\left(\hat{b}+3\right) + \tau_{xx}} \tag{536}$$

$$c_{rr} = -\frac{\tau_{xx}}{\frac{\eta_\mathrm{p}}{\lambda}\left(\hat{b}+3\right) + \tau_{xx}} \tag{537}$$

$$c_{r\varphi} = 0 \tag{538}$$

Inserting this back into the previous equations the following is retrieved.

$$\frac{\tau_{xx}}{\eta_\mathrm{p}\dot{\gamma}} = 2\frac{\tau_{xr}\left(\hat{b}+3\right)}{\frac{\eta_\mathrm{p}}{\lambda}\left(\hat{b}+3\right) + \tau_{xx}} =: 2\frac{\tau_{xr}B}{DB + \tau_{xx}} \tag{539}$$

$$\frac{\tau_{xr}}{\eta_\mathrm{p}\dot{\gamma}} = -\frac{\tau_{xx}}{\frac{\eta_\mathrm{p}}{\lambda}\left(\hat{b}+3\right) + \tau_{xx}} + 1 = \frac{\frac{\eta_\mathrm{p}}{\lambda}\left(\hat{b}+3\right)}{\frac{\eta_\mathrm{p}}{\lambda}\left(\hat{b}+3\right) + \tau_{xx}} = \frac{DB}{DB + \tau_{xx}} \tag{540}$$

$$\frac{\tau_{x\varphi}}{\eta_\mathrm{p}\dot{\gamma}} = 0 \tag{541}$$

Here the variables D and B have been introduced in order to simplify the following math. Dividing the non-zero equation yields the following.

$$\frac{\tau_{xx}}{\tau_{xr}} = \frac{2}{D}\tau_{xr} \tag{542}$$

$$\tau_{xx} = \frac{2}{D}\tau_{xr}^2 \tag{543}$$

The same fraction can also be formed by equation (539) alone as follows.

$$\frac{\tau_{xx}}{\tau_{xr}} = 2\frac{DB\lambda\dot{\gamma}}{DB + \tau_{xx}} \tag{544}$$

Relating the fractions an expression for τ_{xx} can be found.

$$\frac{2}{D}\tau_{xr} = 2\frac{DB\lambda\dot{\gamma}}{DB+\tau_{xx}} \tag{545}$$

$$\tau_{xx} = \frac{DB\lambda\dot{\gamma}}{\frac{1}{D}\tau_{xr}} - B \tag{546}$$

$$\tau_{xx} = DB\left(\frac{D\lambda\dot{\gamma}}{\tau_{xr}} - 1\right) \tag{547}$$

Insertion into equation (543) yields the following.

$$DB\left(\frac{D\lambda\dot{\gamma}}{\tau_{xr}} - 1\right) = \frac{2}{D}\tau_{xr}^2 \tag{548}$$

$$\frac{D\lambda\dot{\gamma}}{\tau_{xr}} = \frac{2}{D^2B}\tau_{xr}^2 + 1 \tag{549}$$

$$0 = \frac{2}{D^2B}\tau_{xr}^3 + \tau_{xr} - D\lambda\dot{\gamma} \tag{550}$$

$$0 = \tau_{xr}^3 + \frac{D^2B}{2}\tau_{xr} - \frac{D^3B}{2}\lambda\dot{\gamma} \tag{551}$$

$$0 = \tau_{xr}^3 + 3p\tau_{xr} + 2q \tag{552}$$

With $p = \frac{D^2B}{6} > 0$ and $q = -\frac{D^3B}{4}\lambda\dot{\gamma} > 0$. The single real solution [60, 121] of this equation can be written as follows.

$$\tau_{xr} = -2\sqrt{p}\sinh\left(\frac{1}{3}\operatorname{arcsinh}\frac{q}{p^{\frac{3}{2}}}\right) \tag{553}$$

$$= -D\sqrt{\frac{2B}{3}}\sinh\left(\frac{1}{3}\operatorname{arcsinh}\frac{-3\lambda\dot{\gamma}}{\sqrt{\frac{2B}{3}}}\right) \tag{554}$$

This can also be inserted into equation (543) as follows.

$$\tau_{xx} = 2D\frac{2B}{3}\sinh^2\left(\frac{1}{3}\operatorname{arcsinh}\frac{-3\lambda\dot{\gamma}}{\sqrt{\frac{2B}{3}}}\right) \tag{555}$$

Converting this to the total stress σ requires the addition of the solvent stress defined as follows.

$$\underline{\sigma}_{\mathrm{s}} = \eta_{\mathrm{s}}\dot{\gamma}(\hat{e}_r \otimes \hat{e}_x + \hat{e}_x \otimes \hat{e}_r) \tag{556}$$

This yields the following.

$$\sigma_{xx} = 2D\frac{2B}{3}\sinh^2\left(\frac{1}{3}\operatorname{arcsinh}\frac{3\lambda\dot{\gamma}}{\sqrt{\frac{2B}{3}}}\right) \tag{557}$$

$$\frac{\sigma_{xr}}{\dot{\gamma}} = \frac{D}{\dot{\gamma}}\sqrt{\frac{2B}{3}}\sinh\left(\frac{1}{3}\operatorname{arcsinh}\frac{3\lambda\dot{\gamma}}{\sqrt{\frac{2B}{3}}}\right) + \eta_{\mathrm{s}} \tag{558}$$

As already mentioned above the remaining components of $\underline{\tau}$ or $\underline{\sigma}$ respectively are zero. This yields the first normal stress difference N_1 defined as follows.

$$N_1 = \sigma_{xx} - \begin{Bmatrix} \sigma_{yy} & \text{, in 2D} \\ \sigma_{rr} & \text{, in 3D} \end{Bmatrix} = \tau_{xx} = 2\frac{\eta_\mathrm{p}}{\lambda}\frac{2\left(\hat{b}+3\right)}{3}\sinh^2\left(\frac{1}{3}\operatorname{arcsinh}\frac{3\lambda\dot{\gamma}}{\sqrt{\frac{2\left(\hat{b}+3\right)}{3}}}\right) \tag{559}$$

One can see, that the normal stress (as is the case with other elastic effects) is also scaled using the so-called polymer viscosity η_p. This is common among viscoelastic models. The second normal stress difference N_2 vanishes due to all contribution components being zero. This can be seen in its following definition.

$$N_2 = \begin{Bmatrix} \sigma_{yy} - \sigma_{zz} = \tau_{yy} - \tau_{zz} & \text{, in 2D} \\ \sigma_{rr} - \sigma_{\varphi\varphi} = \tau_{rr} - \tau_{\varphi\varphi} & \text{, in 3D} \end{Bmatrix} = 0 \tag{560}$$

The viscosity defined as follows can similarly be found for this model.

$$\begin{aligned} \eta &= \frac{\sigma_{xr}}{\dot{\gamma}} &(561)\\ &= \frac{\tau_{xr} + \eta_\mathrm{s}\dot{\gamma}}{\dot{\gamma}} &(562)\\ &= \eta(\dot{\gamma}) = \frac{\eta_\mathrm{p}}{\lambda\dot{\gamma}}\sqrt{\frac{2\left(\hat{b}+3\right)}{3}}\sinh\left(\frac{1}{3}\operatorname{arcsinh}\frac{3\lambda\dot{\gamma}}{\sqrt{\frac{2\left(\hat{b}+3\right)}{3}}}\right) + \eta_\mathrm{s} &(563) \end{aligned}$$

With $\sigma_{xr} = \sigma_{xy}\operatorname{sgn}(y) \leq 0$ and $\tau_{xr} = \tau_{xy}\operatorname{sgn}(y) \leq 0$ in the 2D case analogous to c_{xr}. Of notable importance is, that the zero-shear viscosity is as follows.

$$\eta_0 = \lim_{\dot{\gamma}\to 0}\eta(\dot{\gamma}) = \eta_\mathrm{p} + \eta_\mathrm{s} \tag{564}$$

This shows, that η_p has the same role as the for the other models here. This is not the case for the original version of the model. Hence, the re-scaling of the variables is sensible. Of notable interest for fitting is, that the shear-rate at which the viscosity contribution of the polymers is halved $\dot{\gamma}_\mathrm{H}$ can be defined as follows.

$$\dot{\gamma}_\mathrm{H} = \frac{1}{\lambda}\sqrt{2\left(\hat{b}+3\right)} \tag{565}$$

Which allows the viscosity to be written in a very fitting-friendly way as follows.

$$\eta(\dot{\gamma}) = \eta_\mathrm{p}\frac{\dot{\gamma}_\mathrm{H}}{\dot{\gamma}}\frac{1}{\sqrt{3}}\sinh\left(\frac{1}{3}\operatorname{arcsinh}\left(3\frac{\dot{\gamma}}{\dot{\gamma}_\mathrm{H}}\sqrt{3}\right)\right) + \eta_\mathrm{s} \tag{566}$$

With this, the relevant shear-flow solutions are covered. Note, that these equations are also valid for Poiseuille flow.

F.2.2.2. Velocity profile

The velocity profile derivation starts from equation (467). In total, the stress experienced by the fluid is the sum of the polymer stress (the stress provided by the constitutive equation) and the solvent stress. This can be written as follows.

$$\tau_{xr} = \partial_x p \frac{r}{2^j} - \eta_s \dot{\gamma} \tag{567}$$

Inserting into equation (543) yields the following.

$$\tau_{xr} = \partial_x p \frac{r}{2^j} - \eta_s \dot{\gamma} =: X - \eta_s \dot{\gamma} \tag{568}$$

Here, the variable $X < 0$ has been introduced to reduce the mathematical burden. This can now be inserted into equation (551).

$$0 = (X - \eta_s\dot{\gamma})^3 + \frac{D^2B}{2}(X - \eta_s\dot{\gamma}) - \frac{D^3B}{2}\lambda\dot{\gamma} \tag{569}$$

$$0 = -(\eta_s\dot{\gamma})^3 + 3(\eta_s\dot{\gamma})^2X - 3\eta_s\dot{\gamma}X^2 + X^3 + \left(\frac{D^2B}{2}X - \frac{D^2B}{2}\eta_s\dot{\gamma}\right) - \frac{D^3B}{2}\lambda\dot{\gamma} \tag{570}$$

$$0 = -\eta_s^3\dot{\gamma}^3 + 3\eta_s^2\dot{\gamma}^2X - 3\eta_s\dot{\gamma}X^2 - \frac{D^2B}{2}\eta_s\dot{\gamma} - \frac{D^3B}{2}\lambda\dot{\gamma} + \left(X^2 + \frac{D^2B}{2}\right)X \tag{571}$$

Define $g := -\dot{\gamma} \geq 0$, to hide the sign.

$$0 = \eta_s^3g^3 + 3\eta_s^2g^2X + 3\eta_sgX^2 + \frac{D^2B}{2}\eta_sg + \frac{D^3B}{2}\lambda g + \left(X^2 + \frac{D^2B}{2}\right)X \tag{572}$$

Group by power of g.

$$0 = \eta_s^3g^3 + 3\eta_s^2Xg^2 + \left(3\eta_sX^2 + \frac{D^2B}{2}\eta_s + \frac{D^3B}{2}\lambda\right)g + \left(X^2 + \frac{D^2B}{2}\right)X \tag{573}$$

The solution depends on whether $\eta_s = 0$. The general case will be solved first. This is a cubic equation of the form:

$$0 = ag^3 + bg^2 + cg + d \tag{574}$$

This is a standard problem, that can be solved using literature [121]. First one performs the following substitution.

$$g =: y - \frac{b}{3a} = y - \frac{3\eta_s^2X}{3\eta_s^3} = y - \frac{X}{\eta_s} \tag{575}$$

This results in an equation of the following form.

$$0 = y^3 + 3py + 2q = 0 \tag{576}$$

Where p and q are defined as follows.

$$3p := \frac{3ac-b^2}{3a^2} = \frac{\left(3\eta_{\mathrm{s}}X^2 + \frac{D^2B}{2}\eta_{\mathrm{s}} + \frac{D^3B}{2}\lambda\right) - 3\eta_{\mathrm{s}}X^2}{\eta_{\mathrm{s}}^3} \tag{577}$$

$$= \frac{\frac{D^2B}{2}(R_\eta + 1)}{\eta_{\mathrm{s}}^2} \tag{578}$$

$$= \frac{D^2B}{2\eta_{\mathrm{s}}^2}(R_\eta + 1) \tag{579}$$

$$> 0 \tag{580}$$

$$2q := \frac{2b^3}{27a^3} - \frac{bc}{3a^2} + \frac{d}{a} \tag{581}$$

$$= 2\frac{3^3X^3}{27\eta_{\mathrm{s}}^3} - \frac{X\left(3\eta_{\mathrm{s}}X^2 + \frac{D^2B}{2}\eta_{\mathrm{s}} + \frac{D^3B}{2}\lambda\right)}{\eta_{\mathrm{s}}^4} + \frac{\left(X^2 + \frac{D^2B}{2}\right)X}{\eta_{\mathrm{s}}^3} \tag{582}$$

$$= 2\frac{X^3}{\eta_{\mathrm{s}}^3} - \frac{X\left(3X^2 + \frac{D^2B}{2} + \frac{D^2B}{2}R_\eta\right)}{\eta_{\mathrm{s}}^3} + \frac{\left(X^2 + \frac{D^2B}{2}\right)X}{\eta_{\mathrm{s}}^3} \tag{583}$$

$$= 2\frac{X^3}{\eta_{\mathrm{s}}^3} - \frac{X\left(2X^2 + \frac{D^2B}{2}R_\eta\right)}{\eta_{\mathrm{s}}^3} \tag{584}$$

$$= X\frac{D^2B}{2\eta_{\mathrm{s}}^3}R_\eta \tag{585}$$

$$= \frac{X}{\eta_{\mathrm{s}}}\frac{D^2B}{2\eta_{\mathrm{s}}^2}R_\eta \tag{586}$$

$$< 0 \tag{587}$$

Here the viscosity ratio has been defined.

$$R_\eta := \frac{\eta_{\mathrm{p}}}{\eta_{\mathrm{s}}} \tag{588}$$

From these, the discriminant D' can be calculated as follows.

$$D' = -p^3 - q^2 = -\frac{D^6B^3}{6^3\eta_{\mathrm{s}}^6}(R_\eta + 1)^3 - \frac{X^2}{\eta_{\mathrm{s}}^2}\frac{D^4B^2}{4^2\eta_{\mathrm{s}}^4}R_\eta^2 \tag{589}$$

$$= -\frac{D^4B^2}{\eta_{\mathrm{s}}^6}\left(\frac{D^2B}{6^3}(R_\eta + 1)^3 + \frac{X^2}{4^2}R_\eta^2\right) \qquad < 0 \tag{590}$$

$D' < 0$ means, that the equation only has a single real root. This root is as follows.

$$y = 2\sqrt{p}\sinh\left(\frac{1}{3}\operatorname{arcsinh}\frac{q}{-p^{\frac{3}{2}}}\right) \tag{591}$$

$$= 2\sqrt{\frac{D^2B}{6\eta_s^2}(R_\eta+1)}\sinh\left(\frac{1}{3}\operatorname{arcsinh}\frac{\frac{X}{\eta_s}\frac{D^2B}{4\eta_s^2}R_\eta}{-\frac{D^3B^{\frac{3}{2}}}{6^{\frac{3}{2}}\eta_s^3}(R_\eta+1)^{\frac{3}{2}}}\right) \tag{592}$$

$$= \frac{D}{\eta_s}\sqrt{\frac{2B}{3}(R_\eta+1)}\sinh\left(\frac{1}{3}\operatorname{arcsinh}\frac{X\frac{1}{4}R_\eta}{-\frac{DB^{\frac{1}{2}}}{6^{\frac{3}{2}}}(R_\eta+1)^{\frac{3}{2}}}\right) \tag{593}$$

$$= \frac{D}{\eta_s}\sqrt{\frac{2B}{3}(R_\eta+1)}\sinh\left(\frac{1}{3}\operatorname{arcsinh}\frac{-X3^{\frac{3}{2}}R_\eta}{D\sqrt{2B}(R_\eta+1)^{\frac{3}{2}}}\right) \tag{594}$$

Reverting the substitutions of $\dot\gamma$ gives the solution of the shear rate.

$$g = \frac{D}{\eta_s}\sqrt{\frac{2B}{3}(R_\eta+1)}\sinh\left(\frac{1}{3}\operatorname{arcsinh}\frac{-X3^{\frac{3}{2}}R_\eta}{D\sqrt{2B}(R_\eta+1)^{\frac{3}{2}}}\right) - \frac{X}{\eta_s} \tag{595}$$

$$\dot\gamma = -\frac{D}{\eta_s}\sqrt{\frac{2B}{3}(R_\eta+1)}\sinh\left(\frac{1}{3}\operatorname{arcsinh}\frac{-X3^{\frac{3}{2}}R_\eta}{D\sqrt{2B}(R_\eta+1)^{\frac{3}{2}}}\right) + \frac{X}{\eta_s} \tag{596}$$

Numerical integration yields the velocity profile. The remaining $\eta_s = 0$ case is considerably easier. Most terms disappear.

$$0 = \eta_s^3g^3 + 3\eta_s^2Xg^2 + \left(3\eta_sX^2 + \frac{D^2B}{2}\eta_s + \frac{D^3B}{2}\lambda\right)g + \left(X^2 + \frac{D^2B}{2}\right)X \tag{597}$$

$$0 = \frac{D^3B}{2}\lambda g + \left(X^2 + \frac{D^2B}{2}\right)X \tag{598}$$

The equation can be trivially realigned.

$$\dot\gamma = \left(\frac{2}{D^2B}X^2 + 1\right)\frac{X}{D\lambda} \tag{599}$$

$$= \left[\frac{2}{D^2B}\left(\partial_x p\frac{r}{2^j}\right)^2 + 1\right]\frac{\partial_x p\frac{r}{2^j}}{D\lambda} \tag{600}$$

$$= \frac{\frac{\partial_x p}{2^j}}{D\lambda}\left[\frac{2}{D^2B}\left(\frac{\partial_x p}{2^j}\right)^2 r^3 + r\right] \tag{601}$$

Define $G = -\partial_x p$.

$$= -\frac{G}{2^j\eta_p}\left(\frac{G^2}{D^2B2^{2j-1}}r^3 + r\right) \tag{602}$$

This is neat enough to integrate analytically.

$$\vec{u}(r) = \int_R^r \dot{\gamma} \mathrm{d}r \hat{e}_x \tag{603}$$

$$= \frac{G}{2^j \eta_\mathrm{p}} \left(\frac{G^2}{D^2 B 2^{2j+1}} r^4 + \frac{1}{2} r^2 \right) \Bigg|_r^R \hat{e}_x \tag{604}$$

$$= \frac{G}{2^{j+1} \eta_\mathrm{p}} \left[\frac{G^2}{D^2 B 2^{2j}} \left(R^4 - r^4 \right) + \left(R^2 - r^2 \right) \right] \hat{e}_x \tag{605}$$

Notably, this retrieves the Poiseuille flow solution in the Oldroyd-B limit ($B \to \infty$).

F.2.2.3. Weissenberg number

The relevant term, that is present in all solutions for this model is the argument of the arcsinh of the viscosity or first normal stress difference. Setting this to $\sqrt{3}$ approximately produces a 21 % drop in viscosity. This is very close to the simple definition for the CY model. The $\sqrt{3}$ is completely arbitrary, and the drop could be made to resemble any nice number. The important part is the dependency on $\hat{b}$ Therefore, it is sensible to define the Weissenberg number as follows for FENE-P.

$$Wi = 3 \frac{\dot{\gamma}}{\dot{\gamma}_\mathrm{H}} = \frac{3}{\sqrt{2\left(\hat{b} + 3\right)}} \lambda \dot{\gamma} \tag{606}$$

This would allow for a significantly shorter and simpler notation of the previous solution. However, this is not done here to avoid confusion.

F.2.2.4. Relation to Oldroyd-B

The FENE-P model becomes the Oldroyd-B model for $b \to \infty$. Physically speaking, this describes the finite extensible dumbbells (FENE-P) to become infinitely extensible (Oldroyd-B). Forming this limit for the equations above is at times difficult, but provides the expected solution. This also shows the consistency of the definition of the Weissenberg number and highlights the necessity to include b in its definition.

F.2.3. PTT

In the following, the solutions for shear rate and Poiseuille flow are found for the PTT model. Here only the $\xi = 0$ case is discussed.

F.2.3.1. Shear solutions

For PTT the remaining source term is as follows (see appendix D.3).

$$\underline{S}_\mathrm{R} = -\exp(\epsilon \operatorname{Tr} \underline{C}) \frac{\underline{C}}{\lambda} \tag{607}$$

Inserting this into equation (512) yields the following.

$$\begin{aligned}\underline{0} = {} & [2c_{rx}\hat{e}_x \otimes \hat{e}_x + c_{rr}(\hat{e}_x \otimes \hat{e}_r + \hat{e}_r \otimes \hat{e}_x) + c_{r\varphi}(\hat{e}_x \otimes \hat{e}_\varphi + \hat{e}_\varphi \otimes \hat{e}_x)] \\ & - \exp(\epsilon \operatorname{Tr} \underline{C}) \frac{\underline{C}}{\lambda\dot{\gamma}} + (\hat{e}_r \otimes \hat{e}_x + \hat{e}_x \otimes \hat{e}_r)\end{aligned} \tag{608}$$

This equation is theoretically separated by components. Simply speaking, this means, that if $\hat{e}_i \otimes \hat{e}_j$ does not appear in this equation $c_{ij} = 0$. This allows some terms to be removed. The remaining ones are as follows.

$$\underline{0} = 2c_{rx}\hat{e}_x \otimes \hat{e}_x - \exp(\epsilon \operatorname{Tr} \underline{C}) \frac{\underline{C}}{\lambda\dot{\gamma}} + \hat{e}_r \otimes \hat{e}_x + \hat{e}_x \otimes \hat{e}_r \tag{609}$$

The symmetry of $\underline{C}$ also holds in cylindrical coordinates, meaning $c_{rx} = c_{xr}$. With this, the equation reads as follows.

$$\exp(\epsilon \operatorname{Tr} \underline{C}) \frac{\underline{C}}{\lambda\dot{\gamma}} = 2c_{xr}\hat{e}_x \otimes \hat{e}_x + \hat{e}_r \otimes \hat{e}_x + \hat{e}_x \otimes \hat{e}_r \tag{610}$$

Note, that in the 2D case $c_{xr} = c_{xy} \operatorname{sgn}(y) \leq 0$. This equation can be separated by components. This yields three equations. Due to symmetry, two carry the same meaning. The two unique equations are listed below.

$$\exp(\epsilon \operatorname{Tr} \underline{C}) \frac{c_{xx}}{\lambda\dot{\gamma}} = 2c_{xr} \tag{611}$$

$$\exp(\epsilon \operatorname{Tr} \underline{C}) \frac{c_{xr}}{\lambda\dot{\gamma}} = 1 \tag{612}$$

The trace can be found to be as follows (see appendix N.2).

$$\operatorname{Tr} \underline{C} = c_{xx} + c_{rr} + c_{\varphi\varphi} = c_{xx} \tag{613}$$

Eliminating c_{xr} yields the following.

$$\exp(2\epsilon c_{xx}) \frac{c_{xx}}{\lambda^2\dot{\gamma}^2} = 2 \tag{614}$$

$$\exp(2\epsilon c_{xx}) 2\epsilon c_{xx} = 4\epsilon\lambda^2\dot{\gamma}^2 \tag{615}$$

This is solved by the Lambert W function as follows.

$$2\epsilon c_{xx} = W_0\left(4\epsilon\lambda^2\dot{\gamma}^2\right) \tag{616}$$

Therefore the other component simplifies to the following.

$$\frac{c_{xr}}{\lambda\dot{\gamma}} = \exp\left[-0.5 W_0\left(4\epsilon\lambda^2\dot{\gamma}^2\right)\right] \tag{617}$$

Converting to τ using equation (97) yields the following.

$$2\epsilon\tau_{xx} = \frac{\eta_\mathrm{p}}{\lambda} W_0\left(4\epsilon\lambda^2\dot{\gamma}^2\right) \tag{618}$$

$$\frac{\tau_{xr}}{\dot{\gamma}} = \eta_\mathrm{p} \exp\left[-0.5 W_0\left(4\epsilon\lambda^2\dot{\gamma}^2\right)\right] \tag{619}$$

Converting this to the total stress σ requires the addition of the solvent stress defined as follows.

$$\underline{\sigma}_\mathrm{s} = \eta_\mathrm{s}\dot{\gamma}(\hat{e}_r \otimes \hat{e}_x + \hat{e}_x \otimes \hat{e}_r) \tag{620}$$

This yields the following.

$$\sigma_{xx} = \frac{\eta_\mathrm{p}}{2\epsilon\lambda} W_0\big(4\epsilon\lambda^2\dot{\gamma}^2\big) \tag{621}$$

$$\frac{\sigma_{xr}}{\dot{\gamma}} = \eta_\mathrm{p} \exp\big[-0.5W_0\big(4\epsilon\lambda^2\dot{\gamma}^2\big)\big] + \eta_\mathrm{s} \tag{622}$$

As already mentioned above the remaining components of $\underline{\tau}$ or $\underline{\sigma}$ respectively are zero. This yields the first normal stress difference N_1 defined as follows.

$$N_1 = \sigma_{xx} - \begin{Bmatrix} \sigma_{yy} & \text{, in 2D} \\ \sigma_{rr} & \text{, in 3D} \end{Bmatrix} = \tau_{xx} = \frac{\eta_\mathrm{p}}{2\epsilon\lambda} W_0\big(4\epsilon\lambda^2\dot{\gamma}^2\big) \tag{623}$$

One can see, that the normal stress (as is the case with other elastic effects) is also scaled using the so-called polymer viscosity η_p. This is common among viscoelastic models. The second normal stress difference N_2 vanishes due to all contributing components being zero. This can be seen in its following definition.

$$N_2 = \begin{Bmatrix} \sigma_{yy} - \sigma_{zz} = \tau_{yy} - \tau_{zz} & \text{, in 2D} \\ \sigma_{rr} - \sigma_{\varphi\varphi} = \tau_{rr} - \tau_{\varphi\varphi} & \text{, in 3D} \end{Bmatrix} = 0 \tag{624}$$

The viscosity defined as follows can similarly be found for this model.

$$\eta = \frac{\sigma_{xr}}{\dot{\gamma}} \tag{625}$$

$$= \frac{\tau_{xr} + \eta_\mathrm{s}\dot{\gamma}}{\dot{\gamma}} \tag{626}$$

$$= \eta(\dot{\gamma}) = \frac{\eta_\mathrm{p}}{\exp[0.5W_0(4\epsilon\lambda^2\dot{\gamma}^2)]} + \eta_\mathrm{s} \tag{627}$$

With $\sigma_{xr} = \sigma_{xy}\,\mathrm{sgn}(y) \leq 0$ and $\tau_{xr} = \tau_{xy}\,\mathrm{sgn}(y) \leq 0$ in the 2D case analogous to c_{xr}. With this, the relevant shear-flow solutions have been covered. Note, that these equations are also valid for Poiseuille flow.

F.2.3.2. Velocity profile

The velocity profile derivation starts from equation (467). In total, the stress experienced by the fluid is the sum of the polymer stress (the stress provided by the constitutive equation) and the solvent stress. This can be written as follows.

$$\tau_{xr} = \partial_x p \frac{r}{2^j} - \eta_\mathrm{s}\dot{\gamma} \tag{628}$$

Dividing equation (611) by equation (612) leads to the following relation.

$$c_{xx} = 2c_{xr}^2 \tag{629}$$

Together with the previous equation and converting to the polymer-conformation tensor the following is found.

$$c_{xx} = 2\frac{\lambda^2}{\eta_\mathrm{p}^2}\left(\partial_x p\frac{r}{2^j} - \eta_\mathrm{s}\dot{\gamma}\right)^2 \tag{630}$$

The following is recovered by inserting this back into equation (612).

$$\dot{\gamma} = \exp\left(\frac{2\epsilon\lambda^2}{\eta_\mathrm{p}^2}\left[\partial_x p\frac{r}{2^j} - \eta_\mathrm{s}\dot{\gamma}\right]^2\right)\frac{1}{\eta_\mathrm{p}}\left(\partial_x p\frac{r}{2^j} - \eta_\mathrm{s}\dot{\gamma}\right) \tag{631}$$

The inclusion of η_s hinders the integration. As a result, this is only solvable semi-analytically.

$$1 = \exp\left(\frac{2\epsilon\lambda^2}{\eta_\mathrm{p}^2}\left[\partial_x p\frac{r}{2^j} - \eta_\mathrm{s}\dot{\gamma}\right]^2\right)\frac{1}{\eta_\mathrm{p}}\left(\partial_x p\frac{r}{2^j\dot{\gamma}} - \eta_\mathrm{s}\right) \tag{632}$$

$$0 = \frac{2\epsilon\lambda^2}{\eta_\mathrm{p}^2}\left[\partial_x p\frac{r}{2^j} - \eta_\mathrm{s}\dot{\gamma}\right]^2 - \ln\eta_\mathrm{p} + \ln\left(\partial_x p\frac{r}{2^j\dot{\gamma}} - \eta_\mathrm{s}\right) \tag{633}$$

$$0 = 2\epsilon\lambda^2\dot{\gamma}^2\left[\partial_x p\frac{r}{2^j\dot{\gamma}\eta_\mathrm{p}} - \frac{\eta_\mathrm{s}}{\eta_\mathrm{p}}\right]^2 + \ln\left(\partial_x p\frac{r}{2^j\dot{\gamma}\eta_\mathrm{p}} - \frac{\eta_\mathrm{s}}{\eta_\mathrm{p}}\right) \tag{634}$$

This has the following form.

$$0 = ab^2 + \ln b \tag{635}$$

This can be solved using the Lambert W function as follows.

$$b = \sqrt{\frac{W_0(2a)}{2a}} \tag{636}$$

Resolving for r yields the following.

$$r = \frac{2^j\eta_\mathrm{s}}{\partial_x p}\dot{\gamma} - \frac{\eta_\mathrm{p}2^{j-1}}{\lambda\sqrt{\epsilon}\partial_x p}\sqrt{W_0(4\epsilon\lambda^2\dot{\gamma}^2)} \tag{637}$$

One can interpret this by considering the two terms separately. The first term is the solution for a poiseuille flow with viscosity η_s. The second term applies a nonuniform scaling of the profile along the r-axis towards higher radii. To make this equation easier to discuss, the variables $c < 0$ and $d > 0$ are introduced.

$$r = c\dot{\gamma} - R_\eta\frac{c}{d}\sqrt{W_0(d^2\dot{\gamma}^2)} \tag{638}$$

$$= \left(1 - R_\eta\frac{\sqrt{W_0(d^2\dot{\gamma}^2)}}{d\dot{\gamma}}\right)c\dot{\gamma} \tag{639}$$

To recover the final solution, this needs to be inverted. This cannot be done analytically and therefore numerics are required. First, small steps are generated in the interval from

0 to $\frac{R}{c}$. Here R is the radius of the channel or pipe. This range contains all possible values of $\dot{\gamma}$ within the channel or pipe. Evaluating yields the radii at which the respective shear-rate is present. Values above the channel radius are discarded. Then the axis is flipped. Finally, numeric integration using `scipy.integrate.cumulative_trapezoid` [117] is performed. Here, the small stepping for $\dot{\gamma}$, which is the y-axis in this integration improves accuracy. It leads to the steps along the x-axis of the integration being spaced in a way, that produces small steps along the y-axis. The result can be made very accurate through picking a small stepping initially. Linear interpolation is used on the resulting $\vec{u}$ in order to get a value at each required r.

F.2.3.3. Weissenberg number

The relevant term, that is present in all solutions for this model is the Lambert W term as follows.

$$W_0\left(4\epsilon\lambda^2\dot{\gamma}^2\right) \tag{640}$$

For small arguments, the Lambert W function behaves linearly, while for large arguments it can be approximated by a logarithm. If the argument is near unity, both approximations are similarly wrong. This is therefore a sensible place, to term the location at which shear-thinning starts. Therefore, it is sensible to define the Weissenberg number as follows for PTT.

$$Wi = 2\sqrt{\epsilon}\lambda\dot{\gamma} \tag{641}$$

This would allow for a significantly shorter and simpler notation of the previous solution. However, this is not done here to avoid confusion.

F.2.3.4. Relation to Oldroyd-B

The PTT model becomes the Oldroyd-B model for $\xi = \epsilon = 0$. This can easily be seen for all solutions above, when using a linear approximation for the Lambert W function (which is valid for small arguments). Physically speaking, this describes the Network from which PTT is derived not existing. This is consistent with the free dumbbell model, from which Oldroyd-B is derived. This also shows the consistency of the definition of the Weissenberg number and highlights the necessity to include ϵ in its definition.

G. Root of the Capillary number

This section covers, why the root of the typical definition of the Capillary number should be used instead. This discussion will only deal with the definition in respect to the volumetric forces for brevity. It is also valid for membrane forces. The typical definition of the Capillary number Ca is as follows.

$$Ca = \frac{\mu_\mathrm{t}\dot{\gamma}_\mathrm{t}}{G} \tag{642}$$

This uses the typical dynamic viscosity μ_t, the typical shear-rate $\dot{\gamma}_\mathrm{t}$ and the shear modulus G. In section 1.2.1, the concept of non-dimensional numbers being the fraction of the typical timescales of the involved processes has been introduced. Applying the definitions for the timescales from that section to the Capillary number, the following is retrieved.

$$Ca = \frac{\rho_\mathrm{t} L_\mathrm{t}^2}{G} \frac{1}{T_\mathrm{v} T_\mathrm{a}} \tag{643}$$

Now consider an elastic rod. Its spring constant k is given as follows.

$$k = \frac{EA}{L_0} = \frac{EV}{L_0^2} \tag{644}$$

Here, E is the Young's modulus, A is the rods cross-section, V is its volume and L_0 is its length. The oscillation period T_E of a real spring is defined as follows [123, 124].

$$T_\mathrm{E} = 2\pi\sqrt{\frac{m}{k}\frac{4}{\pi^2}} = 4\sqrt{\frac{m}{k}} \tag{645}$$

Here, m is the mass of the rod. The $\frac{4}{\pi^2}$ factor is due to a real spring (like is discussed here) providing its own mass as the mass, that oscillates, gives an effective mass that is reduced compared to its actual mass. This is known as Rayleigh's value Together, one can see, that the oscillation period depends on the Young's modulus as follows.

$$T_\mathrm{E} = 4\sqrt{m\frac{L_0^2}{EV}} = 4L_0\sqrt{\frac{\rho}{E}} \tag{646}$$

$$\frac{T_\mathrm{E}^2}{16} = L_0^2\frac{\rho}{E} \tag{647}$$

From this it becomes clear, that the Young's modulus encodes the square of a typical time. Inserting this back into the definition of the Capillary number (treating L_0 and ρ as typical values) yields the following.

$$Ca = 2(1+\nu)\frac{\rho_\mathrm{t} L_\mathrm{t}^2}{E}\frac{1}{T_\mathrm{v} T_\mathrm{a}} \tag{648}$$

$$= 2(1+\nu)\frac{T_\mathrm{E}^2}{16}\frac{1}{T_\mathrm{v} T_\mathrm{a}} \tag{649}$$

$$= \frac{1+\nu}{8}\frac{T_\mathrm{E}^2}{T_\mathrm{v} T_\mathrm{a}} \tag{650}$$

Here, the Poisson ratio ν has been used to convert between the moduli. The result makes it clear, that the Capillary number, as it is typically defined, does actually represent a fraction of squared timescales. Consequently, in order to keep the rule of fractions of timescales, the root of the Capillary number needs to be taken. This results in the following.

$$Ca' = \sqrt{Ca} = \sqrt{\frac{1+\nu}{8}\frac{T_{\mathrm{E}}}{\sqrt{T_{\mathrm{v}}T_{\mathrm{a}}}}} \tag{651}$$

This relates the timescale provided by the Young's modulus with the geometric means of the advective and viscous timescales. Finally, the prefactor is removed in order to assure, that unity is the typical value, at which the meaning of the dimensionless number changes. This yields the following.

$$Ca_{\mathrm{K}} = \sqrt{\frac{8}{1+\nu}}\sqrt{Ca} = \sqrt{\frac{8}{1+\nu}\frac{\mu_{\mathrm{t}}\dot{\gamma}_{\mathrm{t}}}{G}} \tag{652}$$

With the new Capillary number Ca_{K}, which is used in this thesis. The prefector causes a reduction of ≈ 0.43 for an incompressible solid. It can be ignored for simplicity as the root is the conceptually important part. However, if it is not compensated, the typical value for Ca' at which it can be considered large (or of relevant size) is not at unity, but noticeably below. To better illustrate, the improvement, that this definition provides, a literature comparison is helpful. Consider figure 8a) of reference [70]. This has been reproduced in figure 137.

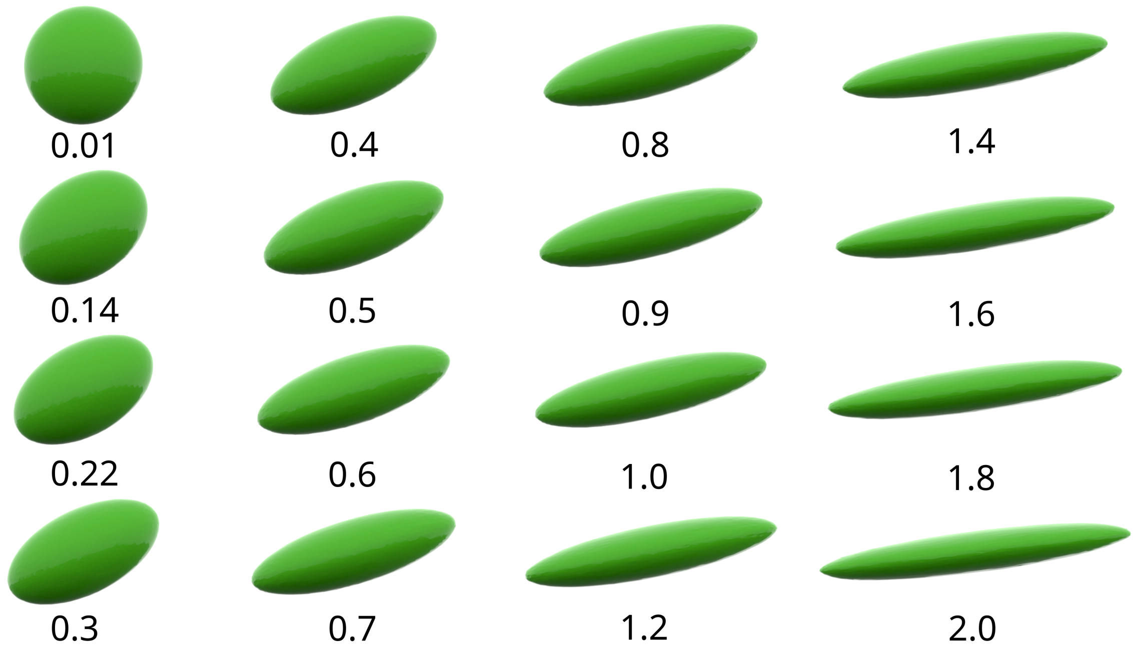


Figure 137: Image of simulation output for deformed cells at various values of Ca. Reproduced from [70].

One can see, that for $Ca = 1$, the cell is already considerably deformed. Furthermore, consider the difference between $Ca = 0.01$ and $Ca = 0.6$, to the difference between $Ca = 1.4$ and $Ca = 2.0$. The first shows large changes, while the second remains largely the same. This shows, that the deformation depends on Ca in a nontrivial way. If Ca ought to serve as a proxy for this, the relationship should be simpler. The same image, but using Ca_K instead, can be seen in figure 138.

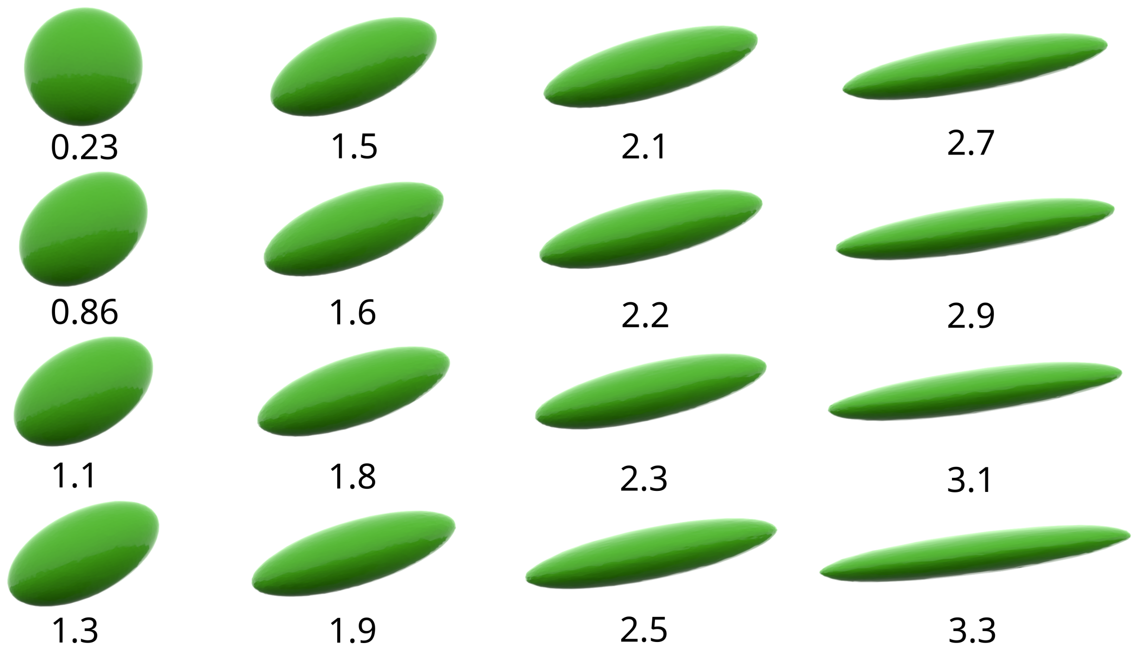


Figure 138: Image of simulation output for deformed cells at various values of Ca_K. Altered from [70].

It is immediately clear, that this scale improves the aforementioned points. $Ca_\mathrm{K} = 1$ is closer to the onset of noticeable deformation. Early deformation happens at a similar rate (in respect to Ca_K) to late deformation. This can be expressed more quantitatively, once a deformation metric is introduced. For this, consider figure 3.5a) from reference [125]. This has been reproduced in figure 139. Note, that the Roscoe theory (see section 9) can be used to predict deformation of cells in pure shear-flows.

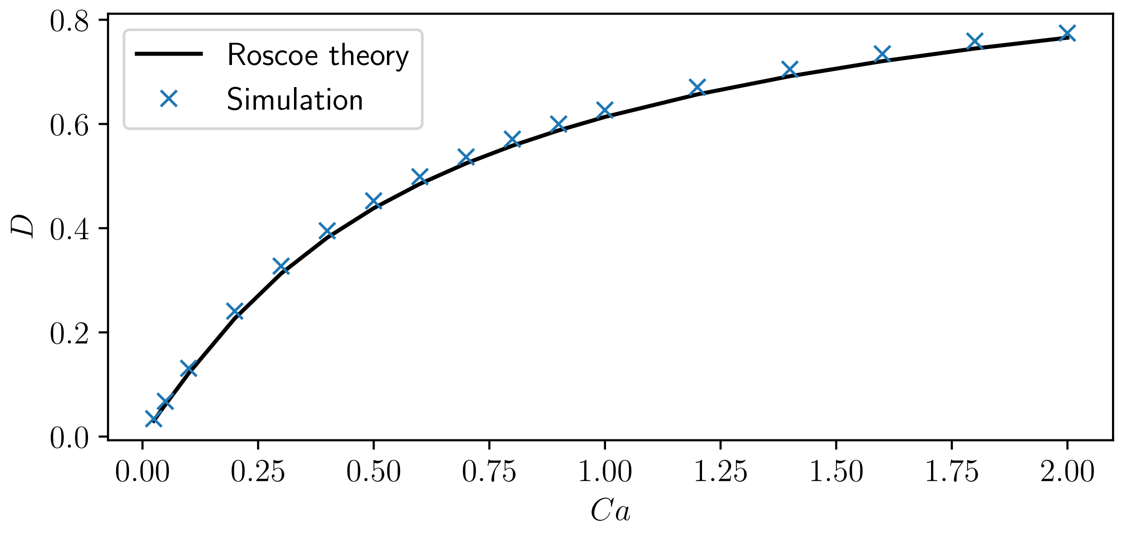


Figure 139: Plot of the Taylor deformation D as a function of Ca for both simulation and theory. Reproduced from [125].

One can immediately see, that the dependency of the Taylor deformation D (see section 11.5 for definition) on Ca seems to be a square root. Consequently, the transformation of the plot to a D-Ca_K diagram is promising. The result can be seen in figure 140.

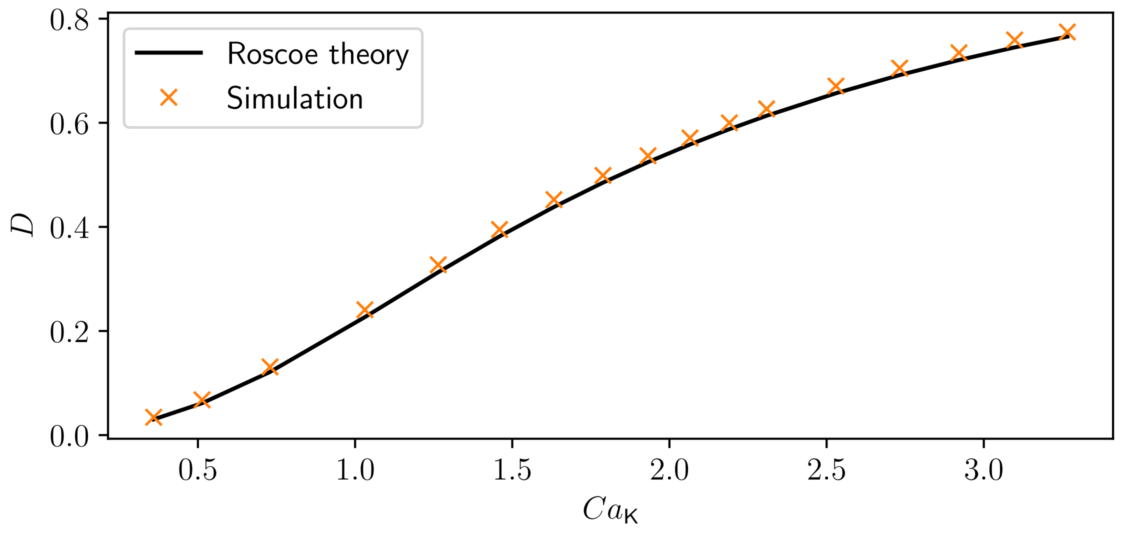


Figure 140: Plot of the Taylor deformation D as a function of Ca_K for both simulation and theory. Adapted from [125].

While the curve is not fully linear, it is close to it. Consequently, Ca_K is significantly better at conveying expected deformation than Ca. This is ultimately the reason why it is the preferred version of this dimensionless number in this thesis.

H. Point distance in IBM meshes

According to literature [9, 86], the distance between two IBM points must be equal to the lattice constant. Some authors even require half the lattice constant or below. This can be seen as a good rule of thumb, given some constraints are fulfilled. Finer meshes are mostly a waste of resources, and can even cause issues if points come to close causing unphysical deformations. One should keep in mind, that the mesh gets deformed during the simulations. This means, that the ideal sharp error minima demonstrated by Krüger *et al.* [9] aren't possible anyway. Furthermore, with a fine mesh, it is quite easy for the deformation to cause points to approach too closely. To understand, why larger gaps are not desirable, one must consider the roots of IBM. Traditionally, the membrane-only cells were the focus. Physically, these cells conserve their volume, as their filling is an incompressible fluid. The models describing the membrane do however not contain a volume-conservation force. So in order to accurately model volume conservation, the simulation needs to accurately reproduce the pressure inside the membrane. Having gaps between IBM points, that are larger, than a lattice constant, would mean, that there are LBM nodes, that theoretically intersect the membrane, but do not have any IBM points nearby. These are not considered by the IBM algorithm and consequently, they do not experience any forces from it. This allows for a flow through the membrane. With this, the pressure escapes and the membrane does not conserve volume. In order to avoid this, the meshes are typically designed with a spacing approximately equal to the lattice constant. For membrane-only cells *FluidX3D* has an additional volume-conservation force (see section 6.4.1.3). Due to the incompressibility of the fluid, the actual magnitude of this force does not matter, it just has to be large. With this the volume conservation is assured even with a less refined mesh. The elastic models for volumetric cells are inherently volume conserving (see section 5.3). Furthermore, volumetric cells have a lot more points, making it a lot less likely for no IBM point to be near an LBM node even for rougher meshes. This means, that this rule of picking the mesh distance equal to the lattice constant does not matter for *FluidX3D*. It is used as a rule of thumb. Deviation from it is required to prevent an instability (see section 15.2).

I. Race conditions and atomics

A race condition refers to two threads of execution doing modifying the same variable at the same time and interfering with each other. In the case of the IBM algorithm, multiple threads try to add to a value stored at the same address. This is an issue due to the fact how a computer works. It has a working register (the accumulator) and can only perform operations on the accumulator. Consequently, the basic instructions a computer performs for the task of adding to a memory address as follows.

- Read the current value from the address into the accumulator.
- Add the desired value to the accumulator.
- Write the value in the accumulator back to the memory address.

If two threads perform this action on the same memory address at the same time, the exact timing between them determines the outcome. If the timing, which is essentially random, lines up badly, one of the additions is lost. To prevent this, computers have instructions for atomic operations. Atomic operations are operations, that take place as one block within a single clock cycle, without the possibility of another thread interfering. In 2022 an OpenCL extension was published to allow for floating point atomics [126], however there is still now widespread adoption. Consequently, integer atomics have to be used for this floating point problem. For this, the atomic compare and exchange instruction `atom_cmpxchg` is used. It reads the value at the memory address (first argument), compares it to the second argument, and if they match writes the third argument to the memory address. The read value is returned. This can be used to construct an atomic addition function as can be seen in figure 141.

```
void INL atomicAddDouble(volatile global double* targetAddress,
                         const double valueToAdd) {
    union {
        ulong  asLong;
        double asDouble;
    } currentValue, lastValue, targetValue;
    currentValue.asDouble = *targetAddress;
    do {
        targetValue.asDouble =
            (lastValue.asDouble=currentValue.asDouble)+valueToAdd;
        currentValue.asLong =
            atom_cmpxchg((volatile global ulong*)targetAddress,
                                lastValue.asLong, targetValue.asLong);
    } while(currentValue.asLong!=lastValue.asLong);
}
```

Figure 141: C++ code of an atomic addition function for floating point values.

This boils down to, reading the value, adding the desired amount and saving it, given that the value at the address has not changed in the meantime. This loops until it works. Some datatype reinterpretation is required to trick the integer function to works with floating point numbers. Given a limited amount of threads use this function, it works reasonably well, albeit with quite some time wasted looping. With too many threads, this can in theory lock up and never finish. For the present IBM code it works. The order of the additions is random, introducing some noise into the simulation output, due to the rounding error compounding differently. This is not an issue, but must be considered in the definition of benchmark simulations. Changing this function to proper floating point atomics as soon as they are available is strongly advised.

J. Derivation of elastic forces

The forces caused by the elastic forces depend on the chosen geometry. The bulk forces act on tetrahedron. Hence, the respective indices (typically the top indices in this section) run from 0 to 3. This derivation for the simple cases is based on reference [69], which in turn is based on reference [71]. The Mooney–Rivlin derivation is based on reference [70]. However, while the reference is scalar, care is taken in this thesis to stay as close to matrix notation as possible. This vastly reduces complexity. The elastic force $\vec{F}$ can be defined from the principal of virtual work as follows.

$$F_I = -\frac{\partial W}{\partial u_I} = -V_0 \frac{\partial U}{\partial u_I} \tag{653}$$

Here, W is the work retrieved, by scaling the energy densities U from section 5.3, by the reference volume of the tetrahedron V_0. $\vec{u}$ is the displacement mentioned in section 5.1. It is defined from the deformed positions $\vec{y}$ and reference positions $\vec{y}$ as follows.

$$\vec{u} = \vec{y} - \vec{x} \tag{654}$$

The displacement is defined across the whole tetrahedron using interpolation from the known values at the vertices $\vec{u}^{(a)}$.

$$\vec{u} = \left(1 - \sum_{a=1}^{3} \xi_a\right) \vec{u}^{(0)} + \sum_{a=1}^{3} \xi_a \vec{u}^{(a)} \tag{655}$$

The ξ_a form a dimensionless coordinate system inside the tetrahedron. The reference node is given the index 0 here contrary to the 4 from references [69, 70], because, the meshing algorithm employed in this thesis (gmsh [85]), generates right-handed tetrahedrons in respect to the first node. This choice assures, that the normals are easy to compute and have a defined direction. Deriving the displacement in respect to the reference coordinates introduces the base tensors required for the derivation as follows.

$$\frac{\partial u_I}{\partial x_J} = \underbrace{\frac{\partial u_I}{\partial \xi_K}}_{U_{IK}} \underbrace{\frac{\partial \xi_K}{\partial x_J}}_{R^{-1}_{0,KJ}} \tag{656}$$

The tensor $\underline{R}_0$ is termed $\underline{J}$ or $\underline{B}^{-1}$ in references [69, 70], both of which are conflicting and less expressive. From these definitions, one can write $\underline{U}$ ($\underline{A}$ in references [69, 70]) as follows.

$$\underline{U} = \begin{pmatrix} u_1^{(1)} - u_1^{(0)} & u_1^{(2)} - u_1^{(0)} & u_1^{(3)} - u_1^{(0)} \\ u_2^{(1)} - u_2^{(0)} & u_2^{(2)} - u_2^{(0)} & u_2^{(3)} - u_2^{(0)} \\ u_3^{(1)} - u_3^{(0)} & u_3^{(2)} - u_3^{(0)} & u_3^{(3)} - u_3^{(0)} \end{pmatrix} \tag{657}$$

$\underline{U}$ contains the displacements. Furthermore $\underline{R}_0$ can be defined as follows.

$$R_{0,IJ} = \frac{\partial x_I}{\partial \xi_J} \tag{658}$$

Using the same interpolation for x as was used for $\vec{u}$, the full matrix can explicitly be given as follows.

$$\underline{R}_0 = \begin{pmatrix} x_1^{(1)} - x_1^{(0)} & x_1^{(2)} - x_1^{(0)} & x_1^{(3)} - x_1^{(0)} \\ x_2^{(1)} - x_2^{(0)} & x_2^{(2)} - x_2^{(0)} & x_2^{(3)} - x_2^{(0)} \\ x_3^{(1)} - x_3^{(0)} & x_3^{(2)} - x_3^{(0)} & x_3^{(3)} - x_3^{(0)} \end{pmatrix} \tag{659}$$

$\underline{R}_0$ contains the reference edge vectors. Another matrix $\underline{R}$ shall be introduced here, defined as follows.

$$\underline{R} = \underline{U} + \underline{R_0} \tag{660}$$

One can see, that $\underline{R}$ contains the deformed edge vectors. It should also be noted, that the deformation gradient tensor $\underline{F}$ can be expressed using these tensors as follows.

$$\underline{F} = \underline{UR_0^{-1}} + \mathbb{1} = \underline{RR_0^{-1}} \tag{661}$$

The second equivalence is what actually gets used in *FluidX3D*. From the definitions above, one can see the following.

$$\frac{\partial U_{LM}}{\partial u_I^{(a)}} = \begin{cases} \delta_{LI}\delta_{Ma} & , \forall a \neq 0 \\ -\delta_{LI} & , a = 0 \end{cases} \tag{662}$$

$$\frac{\partial R_{0,LM}^{-1}}{\partial u_I^{(a)}} = 0 \tag{663}$$

The case $a = 0$ is avoided and handled differently at the end. Furthermore, the following derivatives of $\vec{F}$ can be given.

$$\frac{\partial F_{KJ}}{\partial U_{LM}} = \frac{\partial}{\partial U_{LM}} \left(\underline{UR_0^{-1}} + \mathbb{1}\right)_{KJ} \tag{664}$$

$$= \frac{\partial}{\partial U_{LM}} \sum_k U_{Kk} R_{0,kJ}^{-1} \tag{665}$$

$$= \sum_k \delta_{LK}\delta_{Mk} R_{0,kJ}^{-1} \tag{666}$$

$$= \delta_{LK} R_{0,MJ}^{-1} \tag{667}$$

$$\frac{\partial F_{KJ}}{\partial R_{0,LM}^{-1}} = \frac{\partial}{\partial R_{0,LM}^{-1}} \left(\underline{UR_0^{-1}} + \mathbb{1}\right)_{KJ} \tag{668}$$

$$= \frac{\partial}{\partial R_{0,LM}^{-1}} \sum_k U_{Kk} R_{0,kJ}^{-1} \tag{669}$$

$$= \sum_k U_{Kk}\delta_{Lk}\delta_{MJ} \tag{670}$$

$$= U_{KL}\delta_{MJ} \tag{671}$$

With this, the force on a given point on the tetrahedron $\vec{F}^{(a)}$ can be computed using the chain-rule as follows.

$$F_I^{(a)} = -V_0 \frac{\partial U}{\partial u_I^{(a)}} = -V_0 \sum_{K,J} \frac{\partial U}{\partial F_{KJ}} \left(\sum_{L,M} \frac{\partial F_{KJ}}{\partial U_{LM}} \frac{\partial U_{LM}}{\partial u_I^{(a)}} + \sum_{L,M} \frac{\partial F_{KJ}}{\partial R^{-1}_{0,LM}} \frac{\partial R^{-1}_{0,LM}}{\partial u_I^{(a)}} \right) \tag{672}$$

From here on, a gets treated as $a \in \{1, 2, 3\}$, and $\vec{F}^{(0)}$, gets calculated from the force balance as follows.

$$\vec{F}^{(0)} = -\sum_a \vec{F}^{(a)} \tag{673}$$

Inserting the precomputed derivatives, the force simplifies to the following.

$$F_I^{(a)} = -V_0 \sum_{K,J} \frac{\partial U}{\partial F_{KJ}} \left(\sum_{L,M} \delta_{LK} R^{-1}_{0,MJ} \delta_{LI} \delta_{Ma} \right) \tag{674}$$

$$= -V_0 \sum_{K,J} \frac{\partial U}{\partial F_{KJ}} \left(\sum_{L} R^{-1}_{0,aJ} \delta_{LI} \delta_{IK} \right) \tag{675}$$

$$= -V_0 \sum_{K,J} \frac{\partial U}{\partial F_{KJ}} R^{-1}_{0,aJ} \delta_{IK} \tag{676}$$

$$= -V_0 \sum_{J} \frac{\partial U}{\partial F_{IJ}} R^{-1}_{0,aJ} \tag{677}$$

With this, the individual models can be calculated using a single derivative. It will be shown, that all models can be written to return their forces as the rows of a force tensor $\underline{M}^{\mathrm{T}}_{\mathrm{Force}}$, that results from a matrix valued equation of reasonable size. Note, that the forces could also be calculated via the stress instead of via the derivative. These formulations are identical for the chosen discretization. In the following, the models are listed in the same order as in section 5.3, starting with the most linear, going towards the most nonlinear.

J.1. Linear elastic

The energy density for the linear elastic model is defined as follows.

$$U_{\mathrm{LE}} = \frac{G}{2} \left[\mathrm{Tr}\left(\underline{F}^2\right) + \mathrm{Tr}\left(\underline{F}^{\mathrm{T}} \underline{F}\right) - \frac{2}{3} \mathrm{Tr}\left(\underline{F}\right)^2 \right] + \frac{\kappa}{2} [\mathrm{Tr}(\underline{F}) - 3]^2 \tag{678}$$

The derivative reads as follows.

$$\begin{aligned}\frac{\partial U}{\partial F_{IJ}} = \frac{\partial}{\partial F_{IJ}}\Bigg\{\frac{G}{2}\left[\left(\sum_{i,k} F_{ik}F_{ki}\right) + \left(\sum_{i,k} F_{ki}F_{ki}\right) - \frac{2}{3}\left(\sum_i F_{ii}\right)^2\right] \\ +\frac{\kappa}{2}\left[\left(\sum_i F_{ii}\right) - 3\right]^2\Bigg\}\end{aligned} \tag{679}$$

$$\begin{aligned}= \Bigg\{\frac{G}{2}\left[\left(\sum_{i,k}\delta_{Ii}\delta_{Jk}F_{ki} + \delta_{Ik}\delta_{Ji}F_{ik}\right) + \left(\sum_{i,k} 2\delta_{Ik}\delta_{Ji}F_{ki}\right) - \frac{4}{3}\left(\sum_i F_{ii}\right)\left(\sum_i \delta_{Ii}\delta_{Ji}\right)\right] \\ +\kappa\left[\left(\sum_i F_{ii}\right) - 3\right]\left(\sum_i \delta_{Ii}\delta_{Ji}\right)\Bigg\}\end{aligned} \tag{680}$$

$$\begin{aligned}= \Bigg\{\frac{G}{2}\left[\left(\sum_i \delta_{Ii}F_{Ji} + \delta_{Ji}F_{iI}\right) + \left(\sum_i 2\delta_{Ji}F_{Ii}\right) - \frac{4}{3}\left(\sum_i F_{ii}\right)\delta_{IJ}\right] \\ +\kappa\left[\left(\sum_i F_{ii}\right) - 3\right]\delta_{IJ}\Bigg\}\end{aligned} \tag{681}$$

$$= \frac{G}{2}\left[(F_{JI} + F_{JI}) + (2F_{IJ}) - \frac{4}{3}\operatorname{Tr}(\underline{F})\delta_{IJ}\right] + \kappa[\operatorname{Tr}(\underline{F}) - 3]\delta_{IJ} \tag{682}$$

$$= \frac{G}{2}\left[2F_{JI} + 2F_{IJ} - \frac{4}{3}\operatorname{Tr}(\underline{F})\delta_{IJ}\right] + \kappa[\operatorname{Tr}(\underline{F}) - 3]\delta_{IJ} \tag{683}$$

$$= G\left[F_{JI} + F_{IJ} - \frac{2}{3}\operatorname{Tr}(\underline{F})\delta_{IJ}\right] + \kappa[\operatorname{Tr}(\underline{F}) - 3]\delta_{IJ} \tag{684}$$

Inserting this back into the force equation (eq. (677)) yields the following.

$$F_I^{(a)} = -V_0\sum_J\left\{G\left[F_{JI} + F_{IJ} - \frac{2}{3}\operatorname{Tr}(\underline{F})\delta_{IJ}\right] + \kappa[\operatorname{Tr}(\underline{F}) - 3]\delta_{IJ}\right\}R^{-1}_{0,aJ} \tag{685}$$

$$= -V_0\left\{G\left[\sum_J R^{-1}_{0,aJ}\left(\underline{F} + \underline{F}^{\mathrm{T}}\right)_{JI} - \frac{2}{3}\operatorname{Tr}(\underline{F})R^{-1}_{0,aI}\right] + \kappa[\operatorname{Tr}(\underline{F}) - 3]R^{-1}_{0,aI}\right\} \tag{686}$$

$$= -V_0\left\{G\left[\underline{R}_0^{-1}\left(\underline{F} + \underline{F}^{\mathrm{T}}\right) - \frac{2}{3}\operatorname{Tr}(\underline{F})\underline{R}_0^{-1}\right] + \kappa[\operatorname{Tr}(\underline{F}) - 3]\underline{R}_0^{-1}\right\}_{aI} \tag{687}$$

$$= -V_0\left\{\underline{R}_0^{-1}\left[G\left(\underline{F} + \underline{F}^{\mathrm{T}}\right) + \left(\kappa[\operatorname{Tr}(\underline{F}) - 3] - \frac{2G}{3}\operatorname{Tr}(\underline{F})\right)\mathbb{1}\right]\right\}_{aI} \tag{688}$$

$$= -V_0\left\{\underline{R}_0^{-1}\left[G\left(\underline{F} + \underline{F}^{\mathrm{T}}\right) + \left(\left(\kappa - \frac{2G}{3}\right)\operatorname{Tr}(\underline{F}) - 3\kappa\right)\mathbb{1}\right]\right\}_{aI} \tag{689}$$

$$\underline{M}^{\mathrm{T}}_{\mathrm{Force}} = -V_0\underline{R}_0^{-1}\left[G\left(\underline{F} + \underline{F}^{\mathrm{T}}\right) + \left(\left(\kappa - \frac{2G}{3}\right)\operatorname{Tr}(\underline{F}) - 3\kappa\right)\mathbb{1}\right] \tag{690}$$

The resulting equation is rather simple. It consists solely of the base tensors mentioned above. Only a single matrix product and inversion needs to be calculated for this, making it implementation friendly.

J.2. Modified Saint Venant–Kirchhoff

The energy density for the Saint Venant–Kirchhoff model is defined as follows.

$$U_{\text{SVK}} = G\,\mathrm{Tr}\big(\underline{E}^2\big) + \frac{\lambda}{2}\big[\mathrm{Tr}(\underline{E})^2\big] \tag{691}$$

Here $\lambda = \kappa - \frac{2G}{3}$ is Lamé's first parameter. The definition of the strain $\underline{E}$ shall be kept with a variable c for now. This can be seen in the following.

$$\underline{E} = c(\underline{S} - \mathbb{1}) \tag{692}$$

In literature $c = 1$ [82], but it is sensible to alter it. This modification will prove useful later. Consider the definition of $\underline{S}$.

$$\underline{S}^2 = \underline{F}^{\mathrm{T}}\underline{F} \tag{693}$$

From this, one can write the following.

$$\frac{\partial}{\partial F_{IJ}}\underline{S}^2 = \frac{\partial \underline{S}}{\partial F_{IJ}}\underline{S} + \underline{S}\frac{\partial \underline{S}}{\partial F_{IJ}} \tag{694}$$

Use the symmetry of $\underline{S}$.

$$\frac{\partial}{\partial F_{IJ}}\underline{S}^2 = 2\underline{S}\frac{\partial \underline{S}}{\partial F_{IJ}} \tag{695}$$

$$\frac{1}{2}\underline{S}^{-1}\frac{\partial}{\partial F_{IJ}}\underline{S}^2 = \frac{\partial \underline{S}}{\partial F_{IJ}} \tag{696}$$

$$\frac{1}{2}\underline{S}^{-1}\frac{\partial \underline{F}^{\mathrm{T}}\underline{F}}{\partial F_{IJ}} = \frac{\partial \underline{S}}{\partial F_{IJ}} \tag{697}$$

This allows to write the derivative of $\underline{S}$ based on the derivative of $\underline{F}$ as follows.

$$\frac{\partial\big(\underline{F}^{\mathrm{T}}\underline{F}\big)_{ij}}{\partial F_{IJ}} = \frac{\partial}{\partial F_{IJ}}\left(\sum_k F_{ki}F_{kj}\right) \tag{698}$$

$$= \sum_k \delta_{Ik}\delta_{Ji}F_{kj} + \delta_{Ik}\delta_{Jj}F_{ki} \tag{699}$$

$$= \delta_{Ji}F_{Ij} + \delta_{Jj}F_{Ii} \tag{700}$$

$$\begin{aligned}
\frac{\partial S_{kj}}{\partial F_{IJ}} &= \frac{1}{2}\left(\underline{S}^{-1}\frac{\partial \underline{F}^{\mathrm{T}}\underline{F}}{\partial F_{IJ}}\right)_{kj} && (701)\\
&= \frac{1}{2}\sum_i S^{-1}_{ki}(\delta_{Ji}F_{Ij} + \delta_{Jj}F_{Ii}) && (702)\\
&= \frac{1}{2}\left(\sum_i S^{-1}_{ki}\delta_{Ji}F_{Ij} + \sum_i S^{-1}_{ki}\delta_{Jj}F_{Ii}\right) && (703)\\
&= \frac{1}{2}\left(S^{-1}_{kJ}F_{Ij} + \sum_i S^{-1}_{ki}F_{Ii}\delta_{Jj}\right) && (704)\\
&= \frac{1}{2}\left[S^{-1}_{kJ}F_{Ij} + \left(\underline{S}^{-1}\underline{F}^{\mathrm{T}}\right)_{kI}\delta_{Jj}\right] && (705)
\end{aligned}$$

With this, the following can be written for the derivative of the energy density.

$$\begin{aligned}
\frac{\partial U}{\partial F_{IJ}} &= \frac{\partial U}{\partial F_{IJ}}\left\{Gc^2\,\mathrm{Tr}\left(\underline{S}^2 - 2\underline{S} + \mathbb{1}\right) + \frac{\lambda}{2}c^2\left[\mathrm{Tr}(\underline{S} - \mathbb{1})^2\right]\right\} && (706)\\
&= c^2\frac{\partial U}{\partial F_{IJ}}\left\{G\left(\sum_{i,k} S^2_{ik} - 2\sum_i S_{ii} + 3\right) + \frac{\lambda}{2}\left(\sum_i S_{ii} - 3\right)^2\right\} && (707)
\end{aligned}$$

$\underline{S}$ is symmetric.

$$\begin{aligned}
&= c^2\left\{G\left(\sum_{i,k} 2S_{ik}\frac{\partial S_{ik}}{\partial F_{IJ}} - 2\sum_i \frac{\partial S_{ii}}{\partial F_{IJ}}\right) + \lambda\left(\sum_i S_{ii} - 3\right)\sum_i \frac{\partial S_{ii}}{\partial F_{IJ}}\right\} && (708)\\
&= c^2\left\{G\left(\sum_{i,k} 2S_{ik}\frac{1}{2}\left[S^{-1}_{iJ}F_{Ik} + \left(\underline{S}^{-1}\underline{F}^{\mathrm{T}}\right)_{iI}\delta_{Jk}\right] - 2\sum_i \frac{\partial S_{ii}}{\partial F_{IJ}}\right)\right. \\
&\qquad \left. +\lambda\,\mathrm{Tr}(\underline{S} - \mathbb{1})\sum_i \frac{1}{2}\left[S^{-1}_{iJ}F_{Ii} + \left(\underline{S}^{-1}\underline{F}^{\mathrm{T}}\right)_{iI}\delta_{Ji}\right]\right\} && (709)\\
&= c^2\left\{G\left(\sum_{i,k} S_{ki}\left[S^{-1}_{iJ}F_{Ik} + \left(\underline{S}^{-1}\underline{F}^{\mathrm{T}}\right)_{iI}\delta_{Jk}\right] - 2\sum_i \frac{\partial S_{ii}}{\partial F_{IJ}}\right)\right. \\
&\qquad \left. +\lambda\,\mathrm{Tr}(\underline{S} - \mathbb{1})\frac{1}{2}\left[\left(\underline{F}\,\underline{S}^{-1}\right)_{IJ} + \left(\underline{S}^{-1}\underline{F}^{\mathrm{T}}\right)_{JI}\right]\right\} && (710)
\end{aligned}$$

$\underline{S}$ is symmetric.

$$\begin{aligned}
&= c^2\left\{G\left(\sum_k \left[\delta_{kJ}F_{Ik} + \left(\underline{F}^{\mathrm{T}}\right)_{kI}\delta_{Jk}\right] - 2\sum_i \frac{\partial S_{ii}}{\partial F_{IJ}}\right)\right. \\
&\qquad \left. +\lambda\,\mathrm{Tr}(\underline{S} - \mathbb{1})\left(\underline{F}\,\underline{S}^{-1}\right)_{IJ}\right\} && (711)\\
&= c^2\left\{G\left(\left[F_{IJ} + \left(\underline{F}^{\mathrm{T}}\right)_{JI}\right] - 2\left(\underline{F}\,\underline{S}^{-1}\right)_{IJ}\right) + \lambda\,\mathrm{Tr}(\underline{S} - \mathbb{1})\left(\underline{F}\,\underline{S}^{-1}\right)_{IJ}\right\} && (712)\\
&= c^2\left\{2G\left(\underline{F} - \underline{F}\,\underline{S}^{-1}\right) + \lambda\,\mathrm{Tr}(\underline{S} - \mathbb{1})\underline{F}\,\underline{S}^{-1}\right\}_{IJ} && (713)
\end{aligned}$$

Inserting this back into the force equation (eq. (677)) yields the following.

$$F_I^{(a)} = -V_0 \sum_J c^2 \left\{2G\left(\underline{F} - \underline{F}\underline{S}^{-1}\right) + \lambda \operatorname{Tr}(\underline{S} - \mathbb{1})\underline{F}\underline{S}^{-1}\right\}_{IJ} R_{0,aJ}^{-1} \tag{714}$$

$\underline{S}$ is symmetric.

$$\begin{aligned}
&= -V_0 c^2 \sum_J R_{0,aJ}^{-1} \left\{2G\left(\underline{F}^{\mathrm{T}} - \underline{S}^{-1}\underline{F}^{\mathrm{T}}\right) + \lambda \operatorname{Tr}(\underline{S} - \mathbb{1})\underline{S}^{-1}\underline{F}^{\mathrm{T}}\right\}_{JI} && (715)\\
\underline{M}_{\mathrm{Force}}^{\mathrm{T}} &= -V_0 c^2 \underline{R}_0^{-1} \left\{2G\left(\underline{F}^{\mathrm{T}} - \underline{S}^{-1}\underline{F}^{\mathrm{T}}\right) + \lambda \operatorname{Tr}(\underline{S} - \mathbb{1})\underline{S}^{-1}\underline{F}^{\mathrm{T}}\right\} && (716)\\
&= -V_0 c^2 \underline{R}_0^{-1} \left\{2G\left(\mathbb{1} - \underline{S}^{-1}\right) + \lambda \operatorname{Tr}(\underline{S} - \mathbb{1})\underline{S}^{-1}\right\}\underline{F}^{\mathrm{T}} && (717)\\
&= -V_0 2c^2 \underline{R}_0^{-1} \left\{G(\underline{S} - \mathbb{1}) + \frac{\lambda}{2}\operatorname{Tr}(\underline{S} - \mathbb{1})\mathbb{1}\right\}\underline{S}^{-1}\underline{F}^{\mathrm{T}} && (718)
\end{aligned}$$

Aside from the root-finding algorithm need to actually compute $\underline{S}$, this equation is not too difficult. The root-finding algorithm (see appendix K) is likely the reason this is not popular. This thesis chooses $c = \frac{1}{\sqrt{2}}$ in order to place the force produced between the Neo-Hookean and the Saint Venant–Kirchhoff model. This is done as both the Hertz model and the experimental data lies in this range. Furthermore, for small deformations, this gives the shear term of the force a similar form as the corresponding term of the Saint Venant–Kirchhoff model.

J.3. Saint Venant–Kirchhoff

The energy density for the Saint Venant–Kirchhoff model is defined as follows (inserting all the tensors).

$$U_{\mathrm{SVK}} = \frac{G}{4}\operatorname{Tr}\left(\left(\underline{F}\underline{F}^{\mathrm{T}} - \mathbb{1}\right)^2\right) + \frac{\lambda}{2}\left[\operatorname{Tr}\left(\underline{F}\underline{F}^{\mathrm{T}} - \mathbb{1}\right)^2\right] \tag{719}$$

Here $\lambda = \kappa - \frac{2G}{3}$ is Lamé's first parameter. Remove the higher order terms.

$$\begin{aligned}
U_{\mathrm{SVK}} &= \frac{G}{4}\operatorname{Tr}\left(\left(\underline{F}\underline{F}^{\mathrm{T}} - \mathbb{1}\right)^2\right) + \frac{\lambda}{2}\left[\operatorname{Tr}\left(\underline{F}\underline{F}^{\mathrm{T}} - \mathbb{1}\right)^2\right] && (720)\\
&= \frac{G}{4}\operatorname{Tr}\left(\underline{F}\underline{F}^{\mathrm{T}}\underline{F}\underline{F}^{\mathrm{T}} - 2\underline{F}\underline{F}^{\mathrm{T}} + \mathbb{1}\right) + \frac{\lambda}{2}\left[\operatorname{Tr}\left(\underline{F}\underline{F}^{\mathrm{T}}\right) - 3\right]^2 && (721)\\
&= \frac{G}{4}\left[\operatorname{Tr}\left(\underline{F}\underline{F}^{\mathrm{T}}\underline{F}\underline{F}^{\mathrm{T}}\right) - 2\operatorname{Tr}\left(\underline{F}\underline{F}^{\mathrm{T}}\right) + 3\right] + \frac{\lambda}{2}\left[\operatorname{Tr}\left(\underline{F}\underline{F}^{\mathrm{T}}\right) - 3\right]^2 && (722)
\end{aligned}$$

The derivative of the traces read as follows.

$$\frac{\partial \operatorname{Tr}\left(\underline{F}\underline{F}^{\mathrm{T}}\right)}{\partial F_{IJ}} = 2F_{IJ} \tag{723}$$

This is known from the linear elastic model. The other one is more involved as can be seen in the following.

$$\frac{\partial\,\mathrm{Tr}\big(\underline{F}\,\underline{F}^{\mathrm{T}}\underline{F}\,\underline{F}^{\mathrm{T}}\big)}{\partial F_{IJ}} = \frac{\partial}{\partial F_{IJ}}\left\{\sum_i \big(\underline{F}\,\underline{F}^{\mathrm{T}}\underline{F}\,\underline{F}^{\mathrm{T}}\big)_{ii}\right\} \tag{724}$$

$$= \frac{\partial}{\partial F_{IJ}}\left\{\sum_{i,k} \big(\underline{F}\,\underline{F}^{\mathrm{T}}\big)^2_{ik}\right\} \tag{725}$$

$$= \frac{\partial}{\partial F_{IJ}}\left\{\sum_{i,k} \left(\sum_l F_{il}F_{kl}\right)^2\right\} \tag{726}$$

$$= 2\sum_{i,k}\left(\sum_l F_{il}F_{kl}\right)\left(\sum_l \delta_{Ii}\delta_{Jl}F_{kl} + \delta_{Ik}\delta_{Jl}F_{il}\right) \tag{727}$$

$$= 2\sum_{i,k}\left(\sum_l F_{il}F_{kl}\right)\left(\delta_{Ii}F_{kJ} + \delta_{Ik}F_{iJ}\right) \tag{728}$$

$$= 2\sum_{i,k}\big(\underline{F}\,\underline{F}^{\mathrm{T}}\big)_{ik}\left(\delta_{Ii}F_{kJ} + \delta_{Ik}F_{iJ}\right) \tag{729}$$

$$= 2\left(\sum_k \big(\underline{F}\,\underline{F}^{\mathrm{T}}\big)_{Ik}F_{kJ} + \sum_i \big(\underline{F}\,\underline{F}^{\mathrm{T}}\big)_{iI}F_{iJ}\right) \tag{730}$$

$$= 2\Big(\big(\underline{F}\,\underline{F}^{\mathrm{T}}\underline{F}\big)_{IJ} + \big(\underline{F}^{\mathrm{T}}\underline{F}\,\underline{F}^{\mathrm{T}}\big)_{JI}\Big) \tag{731}$$

$$= 4\big(\underline{F}\,\underline{F}^{\mathrm{T}}\underline{F}\big)_{IJ} \tag{732}$$

Using this, the complete derivative reads as follows.

$$\frac{\partial U}{\partial F_{IJ}} = \frac{G}{4}\left[4\big(\underline{F}\,\underline{F}^{\mathrm{T}}\underline{F}\big)_{IJ} - 4F_{IJ}\right] + \lambda\left[\mathrm{Tr}\big(\underline{F}\,\underline{F}^{\mathrm{T}}\big) - 3\right]2F_{IJ} \tag{733}$$

$$= G\left[\big(\underline{F}\,\underline{F}^{\mathrm{T}}\underline{F}\big)_{IJ} - F_{IJ}\right] + 2\lambda\left[\mathrm{Tr}\big(\underline{F}\,\underline{F}^{\mathrm{T}}\big) - 3\right]F_{IJ} \tag{734}$$

Inserting this back into the force equation (eq. (677)) yields the following.

$$F_I^{(a)} = -V_0\sum_J\left\{G\left[\big(\underline{F}\,\underline{F}^{\mathrm{T}}\underline{F}\big)_{IJ} - F_{IJ}\right] + 2\lambda\left[\mathrm{Tr}\big(\underline{F}\,\underline{F}^{\mathrm{T}}\big) - 3\right]F_{IJ}\right\}R^{-1}_{0,aJ} \tag{735}$$

$$= -V_0\left\{G\left[\big(\underline{R}_0^{-1}\underline{F}^{\mathrm{T}}\underline{F}\,\underline{F}^{\mathrm{T}}\big) - \underline{R}_0^{-1}\underline{F}^{\mathrm{T}}\right] + 2\lambda\left[\mathrm{Tr}\big(\underline{F}\,\underline{F}^{\mathrm{T}}\big) - 3\right]\underline{R}_0^{-1}\underline{F}^{\mathrm{T}}\right\}_{aI} \tag{736}$$

$$\underline{M}^{\mathrm{T}}_{\mathrm{Force}} = -V_0\underline{R}_0^{-1}\underline{F}^{\mathrm{T}}\left\{G\left[\big(\underline{F}\,\underline{F}^{\mathrm{T}}\big) - \mathbb{1}\right] + 2\lambda\left[\mathrm{Tr}\big(\underline{F}\,\underline{F}^{\mathrm{T}} - \mathbb{1}\big)\right]\mathbb{1}\right\} \tag{737}$$

$$= -V_0\underline{H}\left\{G\underline{B}' + 2\lambda[\mathrm{Tr}(\underline{B}')]\mathbb{1}\right\} \tag{738}$$

With the definition of the helper variables $\underline{H}$ and $\underline{B}'$, this too is not too difficult.

J.4. Mooney–Rivlin

Note, that the Neo-Hookean model can be retrieved from Mooney–Rivlin through a limit, so there is no need to handle it separately. The energy density for the Mooney–Rivlin model is defined as follows.

$$U_{\mathrm{MR}} = \frac{G_1}{2}(I-3) + \frac{G_2}{2}(K-3) + \frac{\kappa}{2}(J-1)^2 \tag{739}$$

It is sensible to start from the derivatives of the invariants. First consider J.

$$\begin{aligned}
\frac{\partial J}{\partial F_{IJ}} &= \frac{\partial}{\partial F_{IJ}} \sum_{ijk} \epsilon_{ijk} F_{1i} F_{2j} F_{3k} &(740)\\
&= \sum_{ijk} \epsilon_{ijk} [\delta_{I1}\delta_{Ji} F_{2j} F_{3k} + F_{1i}(\delta_{I2}\delta_{Jj} F_{3k} + \delta_{I3}\delta_{Jk} F_{2j})] &(741)\\
&= \sum_{jk} \epsilon_{Jjk}\delta_{I1} F_{2j} F_{3k} + \sum_{ik} \epsilon_{iJk} F_{1i}\delta_{I2} F_{3k} + \sum_{ij} \epsilon_{ijJ} F_{1i}\delta_{I3} F_{2j} &(742)\\
&= \sum_{jk} \epsilon_{Jjk}\delta_{I1} F_{2j} F_{3k} + \sum_{ik} \epsilon_{Jki} F_{1i}\delta_{I2} F_{3k} + \sum_{ij} \epsilon_{Jij} F_{1i}\delta_{I3} F_{2j} &(743)\\
&= \delta_{I1} \sum_{jk} \epsilon_{Jjk} F_{2j} F_{3k} + \delta_{I2} \sum_{ik} \epsilon_{Jki} F_{3k} F_{1i} + \delta_{I3} \sum_{ij} \epsilon_{Jij} F_{1i} F_{2j} &(744)
\end{aligned}$$

Define the column vectors of $\underline{F}^T$ as $\vec{v}^{(c)}$, where c is the column index. With this, one can see the following.

$$\frac{\partial J}{\partial F_{IJ}} = \delta_{I1}\big(\vec{v}^{(2)} \times \vec{v}^{(3)}\big)_J + \delta_{I2}\big(\vec{v}^{(3)} \times \vec{v}^{(1)}\big)_J + \delta_{I3}\big(\vec{v}^{(1)} \times \vec{v}^{(2)}\big)_J \tag{745}$$

Clearly, these cross products from the rows of a matrix. This is actually part of a definition of the inverse. Consequently, one can write the following.

$$\frac{\partial J}{\partial F_{IJ}} = J F^{-1}_{JI} \tag{746}$$

With this, the next invariant can be handled.

$$\frac{\partial I}{\partial F_{IJ}} = 2F_{IJ} J^{-\frac{2}{3}} - \frac{2}{3}\operatorname{Tr}(\underline{B}) J^{-\frac{2}{3}} F^{-1}_{JI} \tag{747}$$

And the last is also mostly known at this point.

$$\frac{\partial K}{\partial F_{IJ}} = 2\big[\operatorname{Tr}(\underline{B}) F_{IJ} - \big(\underline{F}\,\underline{F}^{\mathrm{T}}\underline{F}\big)_{IJ}\big] J^{-\frac{4}{3}} - \frac{2}{3}\big[\operatorname{Tr}(\underline{B})^2 - \operatorname{Tr}\big(\underline{B}^2\big)\big] J^{-\frac{4}{3}} F^{-1}_{JI} \tag{748}$$

With this, the complete derivative of the energy density reads as follows.

$$\begin{aligned}
\frac{\partial U}{\partial F_{IJ}} &= G_1 J^{-\frac{2}{3}} \left(F_{IJ} - \frac{1}{3}\operatorname{Tr}(\underline{B}) F^{-1}_{JI} \right) \\
&+ G_2 J^{-\frac{4}{3}} \left\{ \big[\operatorname{Tr}(\underline{B}) F_{IJ} - \big(\underline{F}\,\underline{F}^{\mathrm{T}}\underline{F}\big)_{IJ}\big] - \frac{1}{3}\big[\operatorname{Tr}(\underline{B})^2 - \operatorname{Tr}\big(\underline{B}^2\big)\big] F^{-1}_{JI} \right\} \\
&+ \kappa (J-1) J F^{-1}_{JI}
\end{aligned} \tag{749}$$

Inserting this back into the force equation (eq. (677)) yields the following.

$$\begin{aligned}\underline{M}^{\mathrm{T}}_{\text{Force}} &= -V_0 G_1 J^{-\frac{2}{3}}\left(\underline{R}_0^{-1}\underline{F}^{\mathrm{T}} - \frac{1}{3}\operatorname{Tr}(\underline{B})\underline{R}_0^{-1}\underline{F}^{-1}\right)\\ &\quad - V_0 G_2 J^{-\frac{4}{3}}\left\{\left[\operatorname{Tr}(\underline{B})\underline{R}_0^{-1}\underline{F}^{\mathrm{T}} - \left(\underline{R}_0^{-1}\underline{F}^{\mathrm{T}}\underline{F}\underline{F}^{\mathrm{T}}\right)\right] - \frac{1}{3}\left[\operatorname{Tr}(\underline{B})^2 - \operatorname{Tr}\left(\underline{B}^2\right)\right]\underline{R}_0^{-1}\underline{F}^{-1}\right\}\\ &\quad - V_0\kappa(J-1)J\underline{R}_0^{-1}\underline{F}^{-1}\end{aligned} \tag{750}$$

$$\begin{aligned} &= -V_0\underline{R}^{-1}G_1 J^{-\frac{2}{3}}\left(\underline{R}\underline{R}_0^{-1}\underline{F}^{\mathrm{T}} - \frac{1}{3}\operatorname{Tr}(\underline{B})\underline{R}\underline{R}_0^{-1}\underline{F}^{-1}\right)\\ &\quad - V_0\underline{R}^{-1}G_2 J^{-\frac{4}{3}}\left\{\left[\operatorname{Tr}(\underline{B})\underline{R}\underline{R}_0^{-1}\underline{F}^{\mathrm{T}} - \underline{R}\underline{R}_0^{-1}\underline{F}^{\mathrm{T}}\underline{B}\right] - \frac{1}{3}\left[\operatorname{Tr}(\underline{B})^2 - \operatorname{Tr}\left(\underline{B}^2\right)\right]\underline{R}\underline{R}_0^{-1}\underline{F}^{-1}\right\}\\ &\quad - V_0\underline{R}^{-1}\kappa(J-1)J\underline{R}\underline{R}_0^{-1}\underline{F}^{-1}\end{aligned} \tag{751}$$

$$\begin{aligned} &= -V_0\underline{R}^{-1}G_1 J^{-\frac{2}{3}}\left(\underline{F}\underline{F}^{\mathrm{T}} - \frac{1}{3}\operatorname{Tr}(\underline{B})\underline{F}\underline{F}^{-1}\right)\\ &\quad - V_0\underline{R}^{-1}G_2 J^{-\frac{4}{3}}\left\{\left[\operatorname{Tr}(\underline{B})\underline{F}\underline{F}^{\mathrm{T}} - \underline{F}\underline{F}^{\mathrm{T}}\underline{B}\right] - \frac{1}{3}\left[\operatorname{Tr}(\underline{B})^2 - \operatorname{Tr}\left(\underline{B}^2\right)\right]\underline{F}\underline{F}^{-1}\right\}\\ &\quad - V_0\underline{R}^{-1}\kappa(J-1)J\underline{F}\underline{F}^{-1}\end{aligned} \tag{752}$$

$$\begin{aligned} &= -V_0\underline{R}^{-1}G_1 J^{-\frac{2}{3}}\left(\underline{B} - \frac{1}{3}\operatorname{Tr}(\underline{B})\mathbb{1}\right)\\ &\quad - V_0\underline{R}^{-1}G_2 J^{-\frac{4}{3}}\left\{\left[\operatorname{Tr}(\underline{B})\underline{B} - \underline{B}^2\right] - \frac{1}{3}\left[\operatorname{Tr}(\underline{B})^2 - \operatorname{Tr}\left(\underline{B}^2\right)\right]\mathbb{1}\right\}\\ &\quad - V_0\underline{R}^{-1}\kappa(J-1)J\mathbb{1}\end{aligned} \tag{753}$$

$$\begin{aligned} &= -V_0 J\underline{R}^{-1}\left\{G_1 J^{-\frac{5}{3}}\left(\underline{B} - \frac{1}{3}\operatorname{Tr}(\underline{B})\mathbb{1}\right)\right.\\ &\qquad + G_2 J^{-\frac{7}{3}}\left\{\left[\operatorname{Tr}(\underline{B})\underline{B} - \underline{B}^2\right] + \frac{1}{3}\left[\operatorname{Tr}\left(\underline{B}^2\right) - \operatorname{Tr}(\underline{B})^2\right]\mathbb{1}\right\}\\ &\qquad \left. + \kappa(J-1)\mathbb{1}\right\}\end{aligned} \tag{754}$$

While this is not quite as nice, it is still usable. The Neo-Hookean limit is recovered for $G_1 = G$ and $G_2 = 0$. This concludes the force solutions for the elastic models.

K. Root of a tensor

The Modified Saint Venant–Kirchhoff model requires the solution to the following equation in regard to $\underline{S}$.

$$\underline{S}^2 = \underline{F}^{\mathrm{T}}\underline{F} \tag{755}$$

For this first consider polar decomposition. Any complex square matrix $\underline{A}$ can be factorized as follows [127].

$$\underline{A} = \underline{U}\,\underline{P} \tag{756}$$

Here $\underline{U}$ is a unitary matrix, meaning the following holds.

$$\underline{U}^* = \underline{U}^{-1} \tag{757}$$

The star denotes the complex conjugate transpose. In the following this is switched to the transpose, as only real matrices are considered here. In this decomposition $\underline{P}$ can be written to be the root of $\underline{A}$. For the concrete problem, the following holds.

$$\underline{S}^2 = \underline{U}\,\underline{S} \tag{758}$$

$$\underline{U}^{\mathrm{T}}\underline{S}^2 = \underline{S} \tag{759}$$

This means, $\underline{S}$ can easily be determined, given that $\underline{U}$ is known. An iterative approximation algorithm exists to determine $\underline{U}$. It reads as follows [127].

$$\underline{U}_{k+1} = \frac{1}{2}\left[\underline{U}_k + \left(\underline{U}_k^{\mathrm{T}}\right)^{-1}\right] \tag{760}$$

One starts the iteration with $\underline{U}_0 = \underline{A} = \underline{S}^2$. For the purposes of *FluidX3D* five iterations seem to be sufficient. Ten are performed to assure proper convergence even for extreme cases. This algorithm is surprisingly cheap in terms of runtime. Inserting the $\underline{U}$ found through this algorithm into equation (759) yields the desired root $\underline{S}$.

L. Plate limit for conical indenter

Taking the plate limit ($\varphi \to \frac{\pi}{2}$) of the conical model listed in the following (see 7.2) yields a diverging force for all cases.

$$F = \frac{2}{\pi} \frac{E_2}{1-\nu_2^2} \tan(\varphi) \delta_{12}^2 \tag{761}$$

$$\delta_{12} = \sqrt{\frac{\pi}{2} \frac{1-\nu_2^2}{E_2} \frac{F}{\tan(\varphi)}} \tag{762}$$

The variables are the force F, the Young's modulus of the cell E_2, the Poisson ratio of the cell ν_2, half of the tip angle of the indenter cone φ and the indentation δ_{12}. A diverging force for all cases is not a valid limit. Sneddon [92] also lists a solution depending on the contact radius a instead of the indentation. This reads as follows.

$$F = \frac{\pi}{2} \frac{E_2}{1-\nu_2^2} \frac{a^2}{\tan(\varphi)} \tag{763}$$

This is transformed into the form depending on the indentation δ_{12} using the following relation.

$$\delta_{12} = \frac{\pi}{2} \frac{a}{\tan(\varphi)} \tag{764}$$

One could however opt for only doing this transformation halfway and eliminating the angle. This results in the following.

$$F = \frac{E_2}{1-\nu_2^2} a \delta_{12} \tag{765}$$

Assuming the circularity of the cell is not altered significantly by the deformation with a flat indenter, the following relation holds.

$$(R_2 - \delta_{12})^2 + a^2 = R_2^2 \tag{766}$$

$$a = \sqrt{2R_2\delta_{12}} \sqrt{1 - \frac{\delta_{12}}{2R_2}} \tag{767}$$

Here, R_2 is the radius of the cell. In the case of small deformations, the following holds.

$$\delta_{12} \ll 2R_2 \tag{768}$$

Consequently, the model can be approximated in this case as follows.

$$F \approx \sqrt{2} \frac{E_2\sqrt{R_2}}{1-\nu_2^2} \delta_{12}^{\frac{3}{2}} \tag{769}$$

$$\delta_{12} \approx \left(\frac{1}{\sqrt{2}} \frac{1-\nu_2^2}{E_2\sqrt{R_2}} F \right)^{\frac{2}{3}} \tag{770}$$

This takes the same form as the single contact plate solution. The only difference being the $\frac{4}{3}$ prefactor being replaced by a $\sqrt{2}$. These values are close. Without the necessity for the approximation of a, the values would be even closer. This means, that the form containing the angle is not ideal, but the model does converge towards the plate single contact.

M. Maximum deformation in Roscoe theory

The maximum deformation is a direct consequence of Roscoe's equation (80). Equation (242), as it is numbered in this thesis, is reiterated in the following.

$$2\theta = \arccos\left(\frac{\alpha_1^2-\alpha_2^2}{\alpha_1^2+\alpha_2^2}\frac{2K'+(c-1)(\alpha_1^2+\alpha_2^2)^2}{2K'+(c-1)(\alpha_1^2-\alpha_2^2)^2}\right) \tag{771}$$

For the definitions see section 9 and more specific section 9.2. For this to be valid, the argument has be limited to unity, meaning the following has to hold.

$$\frac{\alpha_1^2-\alpha_2^2}{\alpha_1^2+\alpha_2^2}\frac{2K'+(c-1)(\alpha_1^2+\alpha_2^2)^2}{2K'+(c-1)(\alpha_1^2-\alpha_2^2)^2} \leq 1 \tag{772}$$

Assume the denominator to be positive (guaranteed for $c \geq 1$).

$$\frac{\alpha_1^2-\alpha_2^2}{\alpha_1^2+\alpha_2^2}\left[2K'+(c-1)\left(\alpha_1^2+\alpha_2^2\right)^2\right] \leq 2K'+(c-1)\left(\alpha_1^2-\alpha_2^2\right)^2 \tag{773}$$

$$2K'\frac{\alpha_1^2-\alpha_2^2}{\alpha_1^2+\alpha_2^2}+(c-1)\left(\alpha_1^2-\alpha_2^2\right)\left(\alpha_1^2+\alpha_2^2\right) \leq 2K'+(c-1)\left(\alpha_1^2-\alpha_2^2\right)^2 \tag{774}$$

$$\begin{aligned}&2K'\left(\frac{\alpha_1^2-\alpha_2^2}{\alpha_1^2+\alpha_2^2}-1\right)+\\ &(c-1)\left(\alpha_1^2-\alpha_2^2\right)\left[\left(\alpha_1^2+\alpha_2^2\right)-\left(\alpha_1^2-\alpha_2^2\right)\right] \leq 0\end{aligned} \tag{775}$$

$$(c-1)\left(\alpha_1^2-\alpha_2^2\right)2\alpha_2^2 \leq 2K'\frac{2\alpha_2^2}{\alpha_1^2+\alpha_2^2} \tag{776}$$

$$(c-1)\left(\alpha_1^2-\alpha_2^2\right) \leq 2\frac{K'}{\alpha_1^2+\alpha_2^2} \tag{777}$$

Insert equation (236).

$$(c-1)\left(\alpha_1^2-\alpha_2^2\right) \leq \frac{2}{g_3'} \tag{778}$$

Here an approximation for g_3' is required. From Roscoe's equation (21), equation (234) is reiterated in the following.

$$g_3' = \int_0^\infty \frac{\alpha_3^2+\lambda}{\prod_j\left(\alpha_j^2+\lambda\right)^{\frac{3}{2}}}\mathrm{d}\lambda \tag{779}$$

This is not analytically solvable. First consider, that the deformation is handled near exclusively as a balance of α_1 and α_2. Consequently, the following can be assumed to be true.

$$\alpha_3 \approx 1 \tag{780}$$

$$\alpha_2 \approx \frac{1}{\alpha_1} \tag{781}$$

With this, the integrand of equation (779) reads as follows.

$$f(\lambda) = \frac{1+\lambda}{\left[(\alpha_1^2+\lambda)\left(\frac{1}{\alpha_1^2}+\lambda\right)(1+\lambda)\right]^{\frac{3}{2}}} \tag{782}$$

The integrand can be seen in figure 142 for a selection of reasonable α_1.

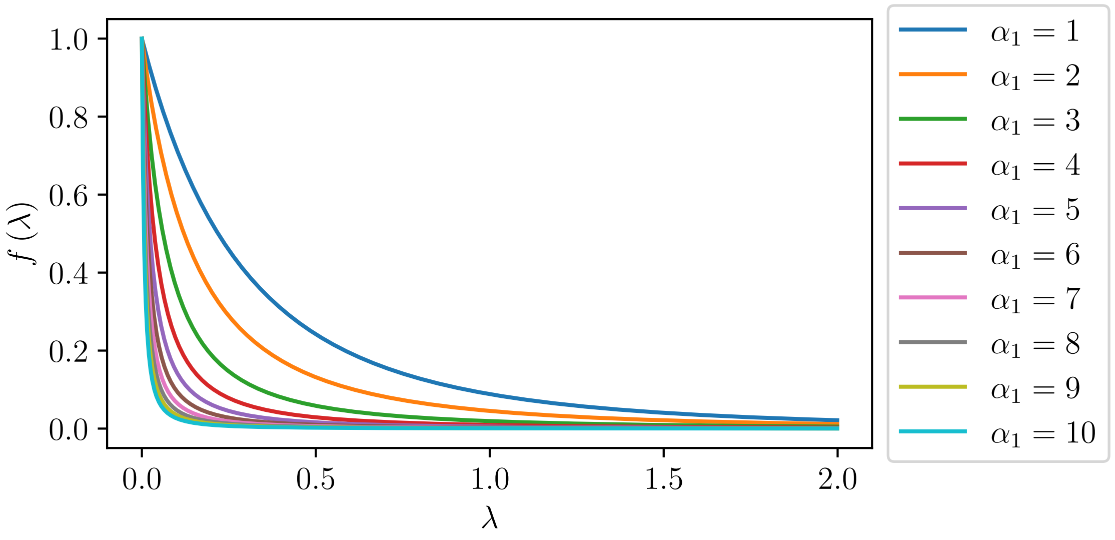


Figure 142: Plot of integrand of equation (779) for a selection of reasonable α_1.

Notably, even tho, the integral technically integrates to infinity, actually only small λ contribute. This effect is even more pronounced for larger α_1. Consequently, an approximation may assume small λ. As this appendix searches for a maximal α_1, the limit of larger α_1 is more interesting. Consequently, the following approximation is made.

$$\lambda \ll \alpha_1 \tag{783}$$

$$f(\lambda) \approx f'(\lambda) = \frac{1+\lambda}{\left[(\alpha_1^2)\left(\frac{1}{\alpha_1^2}+\lambda\right)(1+\lambda)\right]^{\frac{3}{2}}} \tag{784}$$

The approximated integrand can be seen in figure 143 for a selection of reasonable α_1.

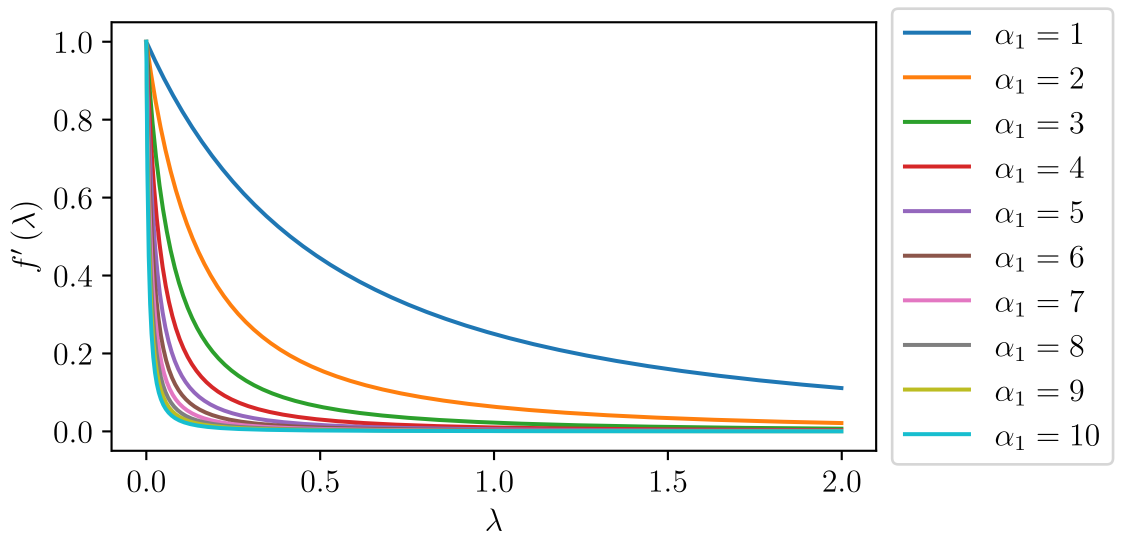


Figure 143: Plot of the approximated integrand of equation (779) for a selection of reasonable α_1.

Notably, this approximation works well for $\alpha_1 = 3$ and is near indistinguishable above. This shows, that the approximation works even for α_1, which might not quite qualify as “large”. Also, this small change makes the integral analytically solvable. The result is listed in the following.

$$g_3' \approx \frac{2}{\alpha_1(\alpha_1 + 1)} \approx \frac{2}{\alpha_1(\alpha_1 + \alpha_2)} \tag{785}$$

Here another approximation was made. This is unorthodox, as dropping the 1 altogether would be more typical for large α_1. However, as the following holds, the approximation using α_2 is just as valid and more useful.

$$\alpha_1(\alpha_1 + 1) \geq \alpha_1(\alpha_1 + \alpha_2) > \alpha^2 \tag{786}$$

The approximated g_3' can be inserted into equation (778), resulting in the following.

$$(c-1)\left(\alpha_1^2 - \alpha_2^2\right) \leq \alpha_1(\alpha_1 + \alpha_2) \tag{787}$$
$$(c-1)(\alpha_1 - \alpha_2) \leq \alpha_1 \tag{788}$$
$$(c-2)\alpha_1 \leq (c-1)\alpha_2 \tag{789}$$

Using $\alpha_2 \approx \frac{1}{\alpha_1}$.

$$(c-2)\alpha_1^2 \leq (c-1) \tag{790}$$

For $1 \leq c < 2$, this is trivially true. Consider $c > 2$.

$$\alpha_1^2 \leq \frac{c-1}{c-2} = \frac{1}{c-2} + 1 \tag{791}$$

For large α_1, the square root can be used to approximate the maximum possible α_1. At intermediary values to small values, it is sensible to use an equation with the same general shape, but allow for slightly different numeric constants. The resulting heuristic equation is given in the following.

$$\alpha_{1,\mathrm{max}} \approx \sqrt{\frac{8}{3}\frac{1}{c-1}+1} \tag{792}$$

The Roscoe equation can be solved numerically. This is compared to the approximation and heuristic solution in figures 144 and 145.

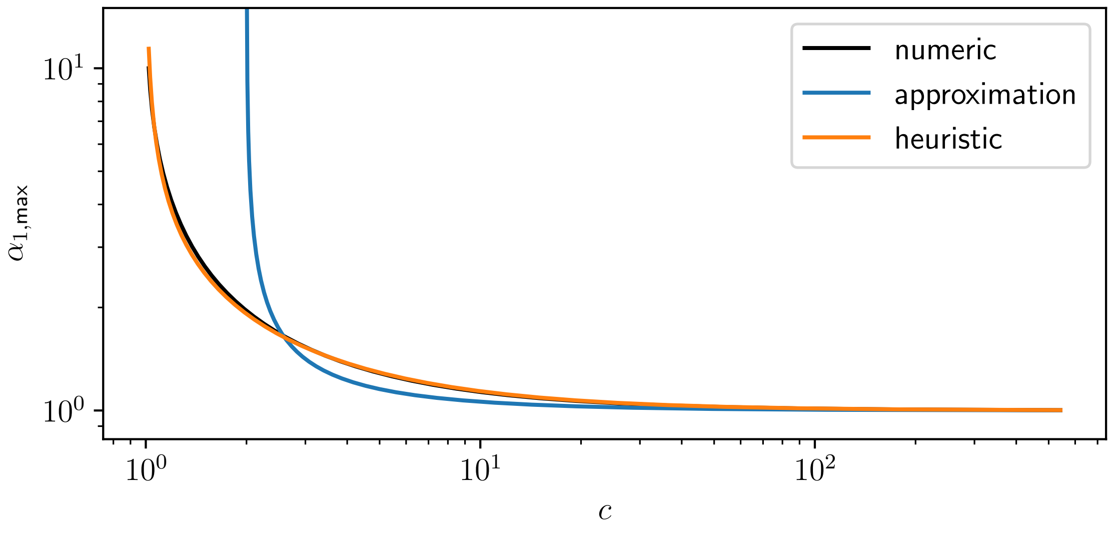


Figure 144: Plot of the maximum α_1 as a function of the contrast c. The numerically exact solution is compared to the approximate solution derived above (eq. (791)), and a heuristic solution (eq. (792)), based on the approximation.

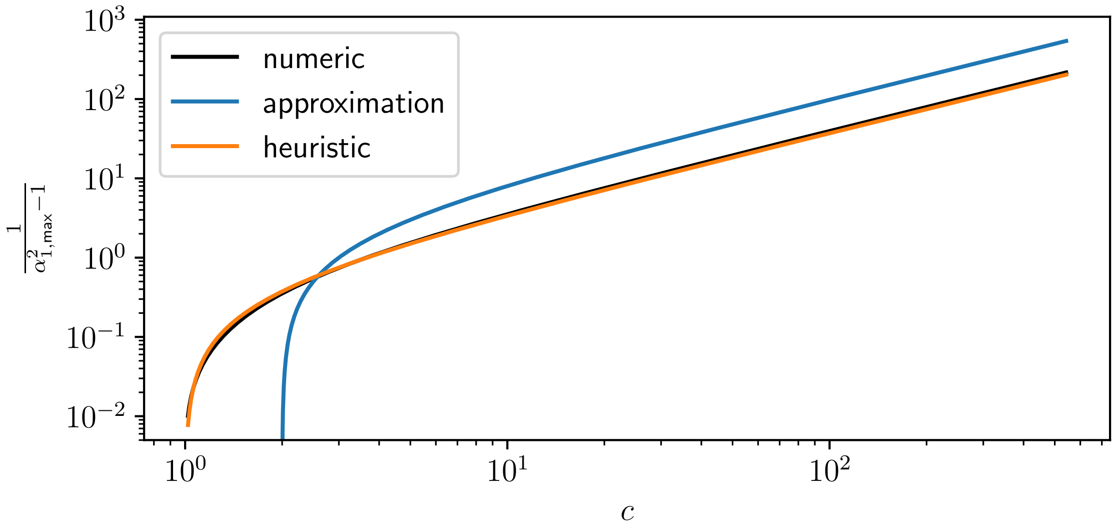


Figure 145: Plot of a simple function of the maximum α_1 as a function of the contrast c. This provides better visibility at higher c. The numerically exact solution is compared to the approximate solution derived above (eq. (791)), and a heuristic solution (eq. (792)), based on the approximation.

This highlights, that approximation in order to find an analytical solution, does not always yield the best results. The heuristic solution is based on the approximate solution. It is used as a “good enough” guess for the maximum α_1, in order to reduce the required compute to solve the Roscoe equations.

N. Curvilinear coordinate systems

This thesis always defines the x-axis to be aligned with the flow direction. The allows for higher symmetry if the coordinate system is aligned to the x-axis. In typical definitions, the z-axis is the axis of highest symmetry. As these definitions arise from another by cyclically switching the components, this is altered definition does not carry any major implications. Only the translation from and to Cartesian coordinates is slightly altered. These will be listed in this section for cylindrical coordinates and for spherical coordinates. First, some common notation is introduced.

N.1. Common notation

For any given parameterization of position-vector $\vec{x}$, the coordinate unit vectors $\hat{e}_i$ can be defined as follows.

$$\hat{e}_i = \hat{N}\frac{\partial \vec{x}}{\partial x_i} \tag{793}$$

Here, x_i refers to the coordinates corresponding to the unit vectors $\hat{e}_i$. Furthermore, the normalization operator $\hat{N}\cdot$ has been used. It shall be defined as follows for any given vector $\vec{b}$.

$$\hat{N}\vec{b} = \frac{\vec{b}}{\left|\vec{b}\right|} \tag{794}$$

The divergence of a tensor $\underline{A}$ with the entries a_{ij} can be written in curvilinear coordinates as follows.

$$\nabla \cdot \underline{A} = \sum_{i,j} \frac{\partial a_{ij}}{\partial x_i}\hat{e}_j \tag{795}$$

The gradient of a vector $\vec{b}$ can be expressed as follows.

$$\nabla \vec{b} = \sum_{i,j} \frac{\partial b_j}{\partial x_i}\hat{e}_i \otimes \hat{e}_j \tag{796}$$

Here $\otimes$ refers to the dyadic product. A tensor can in general be expressed as follows.

$$\underline{A} = a_{ij}\hat{e}_i \otimes \hat{e}_j \tag{797}$$

For convenience, $\hat{e}_i \otimes \hat{e}_j$ is given for cylindrical coordinates. Note, that flipping the vector transposes the result. The cylindrical coordinates are covered next.

N.2. Cylindrical coordinates

The parameterization used in this thesis for systems of cylindrical systems is given in the following.

$$\vec{x} = \begin{pmatrix} x \\ r\cos(\varphi) \\ r\sin(\varphi) \end{pmatrix} \tag{798}$$

The coordinate x retains its meaning, while the new coordinates for the radius r and angle φ are introduced. This yields the new unit vectors listed in the following.

$$\hat{e}_x = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \tag{799}$$

$$\hat{e}_r = \begin{pmatrix} 0 \\ \cos(\varphi) \\ \sin(\varphi) \end{pmatrix} \tag{800}$$

$$\hat{e}_\varphi = \begin{pmatrix} 0 \\ -\sin(\varphi) \\ \cos(\varphi) \end{pmatrix} \tag{801}$$

These unit vectors can also be expressed through the Cartesian unit vectors. This is given in the following.

$$\hat{e}_x = \hat{e}_x \tag{802}$$

$$\hat{e}_r = \cos(\varphi)\hat{e}_y + \sin(\varphi)\hat{e}_z \tag{803}$$

$$\hat{e}_\varphi = -\sin(\varphi)\hat{e}_y + \cos(\varphi)\hat{e}_z \tag{804}$$

From this, the inverse transform can also be given as follows.

$$\hat{e}_x = \hat{e}_x \tag{805}$$

$$\hat{e}_y = \cos(\varphi)\hat{e}_r - \sin(\varphi)\hat{e}_\varphi \tag{806}$$

$$\hat{e}_z = \sin(\varphi)\hat{e}_r + \cos(\varphi)\hat{e}_\varphi \tag{807}$$

For expressing matrices through their components, the following products are required.

$$\hat{e}_x \otimes \hat{e}_x = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \tag{808}$$

$$\hat{e}_x \otimes \hat{e}_r = \begin{pmatrix} 0 & \cos(\varphi) & \sin(\varphi) \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \tag{809}$$

$$\hat{e}_x \otimes \hat{e}_\varphi = \begin{pmatrix} 0 & -\sin(\varphi) & \cos(\varphi) \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \tag{810}$$

$$\hat{e}_r \otimes \hat{e}_r = \begin{pmatrix} 0 & 0 & 0 \\ 0 & \cos^2(\varphi) & \sin(\varphi)\cos(\varphi) \\ 0 & \sin(\varphi)\cos(\varphi) & \sin^2(\varphi) \end{pmatrix} \tag{811}$$

$$\hat{e}_r \otimes \hat{e}_\varphi = \begin{pmatrix} 0 & 0 & 0 \\ 0 & -\sin(\varphi)\cos(\varphi) & \cos^2(\varphi) \\ 0 & -\sin^2(\varphi) & \sin(\varphi)\cos(\varphi) \end{pmatrix} \tag{812}$$

$$\hat{e}_\varphi \otimes \hat{e}_\varphi = \begin{pmatrix} 0 & 0 & 0 \\ 0 & \sin^2(\varphi) & -\sin(\varphi)\cos(\varphi) \\ 0 & -\sin(\varphi)\cos(\varphi) & \cos^2(\varphi) \end{pmatrix} \tag{813}$$

The components of a matrix $\underline{A}$, can then simply be read as follows.

$$a_{ij} = \left(\sum_{k,l} a_{kl} \hat{e}_k \otimes \hat{e}_l \right)_{ij} \tag{814}$$

This avoids the necessity to actually defined the metric tensor. For this coordinate system, the following is retrieved.

$$a_{xx} = a_{xx} \tag{815}$$

$$a_{yy} = a_{rr}\cos^2(\varphi) - (a_{r\varphi} + a_{\varphi r})\sin(\varphi)\cos(\varphi) + a_{\varphi\varphi}\sin^2(\varphi) \tag{816}$$

$$a_{zz} = a_{rr}\sin^2(\varphi) + (a_{r\varphi} + a_{\varphi r})\sin(\varphi)\cos(\varphi) + a_{\varphi\varphi}\cos^2(\varphi) \tag{817}$$

Using this, one can trivially see, that the trace retains its shape in this coordinate system. This is given in the following.

$$\operatorname{Tr}\underline{A} = a_{xx} + a_{yy} + a_{zz} = a_{xx} + a_{rr} + a_{\varphi\varphi} \tag{818}$$

With this all relations of these cylindrical coordinates, which are used in this thesis have been given. Next, the spherical coordinates are explored.

N.3. Spherical coordinates

Spherical coordinates do not get used as intensely in this thesis. Consequently, only a few relations are given. The following parameterization was chosen for spherical coordinates.

$$\vec{x} = \begin{pmatrix} r_\circ \cos(\theta) \\ r_\circ \sin(\theta)\cos(\varphi) \\ r_\circ \sin(\theta)\sin(\varphi) \end{pmatrix} \tag{819}$$

The unit vectors are as follows.

$$\hat{e}_r = \begin{pmatrix} \cos(\theta) \\ \sin(\theta)\cos(\varphi) \\ \sin(\theta)\sin(\varphi) \end{pmatrix} \tag{820}$$

$$\hat{e}_\theta = \begin{pmatrix} -\sin(\theta) \\ \cos(\theta)\cos(\varphi) \\ \cos(\theta)\sin(\varphi) \end{pmatrix} \tag{821}$$

$$\hat{e}_\varphi = \begin{pmatrix} 0 \\ -\sin(\varphi) \\ \cos(\varphi) \end{pmatrix} \tag{822}$$

The surface normal element reads as follows.

$$\mathrm{d}\vec{A} = \frac{\partial \vec{x}}{\partial \theta} \times \frac{\partial \vec{x}}{\partial \varphi} = r_\circ^2 \sin(\theta)\hat{e}_\theta \times \hat{e}_\varphi = r_\circ^2 \sin(\theta)\hat{n} \tag{823}$$

O. Floating-point precision

A computer has to store numbers in a limited amount of bits. Consequently, only a limited amount of numbers can be stored. Real numbers are stored as floating point numbers in order to cover a large range of potential values. Their definition and a few comments on how this affects *FluidX3D* are given in the following.

O.1. Definition

Real numbers are stored as floating point numbers. These are standardized by IEEE 754 [128] and a few associated standards. Floating-point values v can be expressed through the following equation.

$$v = sm2^e \tag{824}$$

Here, s is the sign (-1 or 1), $1 \leq m < 2$ is mantissa and e is the exponent. s is expressed by a single bit. m and e are encoded as normal binary numbers. m is scaled by definition to be between 1 and 2. e has a fixed offset in its definition to make its value range symmetric to 0. This exponential notation allows describing a large range of numbers. However, it is not continuous. Floating-point numbers, or "floats" for short, are discrete and separated by the distance represented by an increment of the least significant bit of the mantissa. This discretization is relative to the value of the number due to the exponential notation. Its depends on the number of bits allotted to each part of the float. The standard defines a few sizes discussed in the following.

O.2. Naming

A float traditionally has 32 bits in total. Higher precision is archived with more bits, and consequently there exists the double precision float ("double" for short) with 64 bits. These names are not guaranteed. Contrary to integers, where the definitions are actually implementation dependent, float and double can be assumed to actually have 32 and 64 bits respectively. Still, this thesis opts for the names "fp32" and "fp64" to be more specific. There are more named precision like "half" (fp16), but these are not relevant to this thesis. Next, a discussion on the actual precision meaning discretization error of the floats is discussed.

O.3. Precision

For fp32, the exponent has 8 bits, the mantissa has 23 bits. The relative distance between values, the discretization error ϵ is calculated as follows.

$$\epsilon_{\text{fp32}} = 2^{-(23+1)} \approx 6 \times 10^{-8} \tag{825}$$

For fp64, the exponent has 11 bits, the mantissa has 52 bits. Consequently, the corresponding epsilon reads as follows.

$$\epsilon_{\text{fp64}} = 2^{-(52+1)} \approx 1 \times 10^{-16} \tag{826}$$

From here, one needs to discuss, if this is enough.

O.4. Precision requirements

If one adds a term, that is smaller than ϵ, the addition is null and void. In the case of IBM simulations, the position is continuously incremented. This means, that this point of additions getting ignored is reached after at most 6×10^{-8} steps. Not every point is moved with the same velocity and if one wants to resolve these differences, the number of steps drops accordingly. Higher viscosities require smaller, and therefore more, steps. Millions of steps are not extraordinary. Consequently, fp32 is often not quite enough and in general IBM requires fp64. Some GPUs, notably from Intel, do not have fp64 capability. For the rest, there is a trade-off. Namely, a significant reduction in speed.

O.5. FLOPS

It is expected for a GPU to take longer to compute a floating-point operation when using fp64 compared to fp32. How much the floating-point operations per second (FLOPS) are reduced for fp64 depends on the vendor. Some modern cards have a fp64 architecture and run fp32 and fp64 at the same speed. Most commonly fp64 comes with a factor 4 fewer FLOPS. Nvidia cards below the data-center price-range are made to have a factor 64 performance penalty. This renders these cards mostly useless for such tasks. In general, *FluidX3D* is programmed to be able to switch between fp32 and fp64 for different parts of the calculations, to allow to only use fp64 were strictly necessary. Finally, a small discussion of the difference between precision and error is warranted.

O.6. Precision vs error

Every computation can in principle introduce up to one ϵ of error. In reality, they do mostly compensate each other by random chance. Otherwise, making millions of LBM steps would be impossible. However, one must distinguish multiplication from addition. Scaling operations like multiplication and division do not significantly introduce relevant error. From standard error propagation, one can show, that the relative error is given by Pythagorean addition of the relative errors of the factors. This is also valid for float arithmetic. If the values have largely different relative errors, only the smaller one is retained without much of an increase. In an absolute worst case scenario of two identical errors, the error is amplified by $\sqrt{2}$. Scaling by known quantities is always free and is a useful tool to make equations more manageable. Addition of two quantities with different signs or more specifically subtraction is an issue. The absolute error is given by Pythagorean addition. The relative error depends on the result. If the result is significantly smaller in magnitude than the terms, that got summed, the relative error is increased accordingly. Subtracting values close to each other can magnify errors massively and should be avoided if possible. With all these effects together in *FluidX3D* and *noFluidX3D* one can see some noise in the data. This is because of the non-deterministic addition in IBM yielding slightly different errors each time-step. The amplitude of this noise is about 100ϵ. This is the real discretization error carried by the simulations.

P. Geometry

There are a few standard geometries, that get used multiple times in this thesis. They are listed here.

P.1. Shear-flow

In a shear-flow the velocity $\vec{u}$ is described by the following equation.

$$\vec{u} = \dot{\gamma} y \hat{e}_x \tag{827}$$

Here $\dot{\gamma}$ is the shear-rate. This thesis always orients the shear-flows with the velocity point in x-direction and the velocity changing in y-direction. The shear-rate is typically defined positive for pure shear-flows in this thesis.

P.2. Generalized Poiseuille flow

(Generalized) Poiseuille flows come in many shapes. Traditionally, a Poiseuille flow is flow of a Newtonian fluid driven through a cylindrical pipe by a pressure gradient. As other fluids behave similarly under such conditions, there is no need to limit the name to Newtonian fluids. This 3D Poiseuille flow has a 2D equivalent. In the 2D case the pipe becomes a channel flow. Only the top and bottom are fixed. This typically changes the equations by a factor of 2. There are also solutions for rectangular and elliptical pipes. These solutions are implemented in *FluidX3D* and some of them are shown in this thesis (see appendix F.1.1). All of the above geometries are termed Poiseuille flows by this thesis. If the fluid flows in a channel with a symmetric but varying cross-section (see sections 24 and 25), the equations for the Poiseuille flow are valid locally. This thesis calls this a Poiseuille-ish flow.

P.3. RT-DC

A render of the channel used to simulate RT-DC experiments in this thesis can be seen in figure 146.

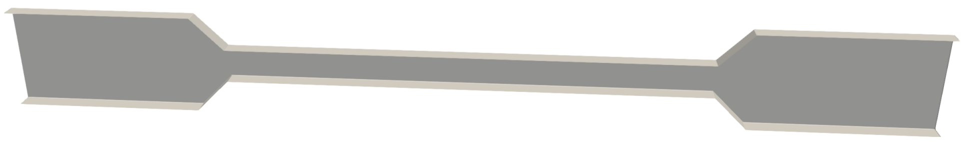


Figure 146: Illustration of the shape of an RT-DC channel. A cross-section is shown. It is slightly tilted for better visibility.

It should be noted, that this image is made from simulation output. In the real channel, the diagonal walls continue outwards. This is prohibitively expensive in simulations. Consequently, only a small diagonal segment is retained. This is enough to reproduce many effects.

P.4. Squaring a circle

The rectilinear lattice cannot form a perfect circle. The edges need to be approximated using stair-casing. This can cause some disturbances, as the corners offer an opportunity for the fluid to misbehave. Even for resolution approaching infinity, a circle cannot be formed, because circumference is not consistent with its area. A trick can be employed here to reduce the impact of this effect. When flagging lattice as either walls or fluid nodes, one would typically check if they are outside the desired radius or not. However, one does actually have some wiggle room here. Altering the actual cutoff by less than a lattice node does not change the size of the channel along the cardinal directions. However, along the other directions some diagonal node may be included or removed from the fluid volume in this way. In a way, this allows to modify the cross-section, without altering the radius. This altered cross-section behaves consistent with an effective radius. Tuning the cross-section carefully allows for more accurate simulations. Determining how much improvement is possible this way is en exercise left to the reader.

Q. Validating free-slip boundaries

Free-slip boundaries as currently implemented in *FluidX3D* carry some caveats which will be explored in the following validations. There are also cases shown, where they work very well.

Q.1. Straight wall

A 2D channel is build using two planes of free-slip boundaries (at $x = 0$ and $x = L_x - 1$). The flow in simulation 34 is driven using a mass-flow boundary at the inlet and a pressure boundary at the outlet.

34 Water in a straight free-slip channel

Box:	$100 \times 20 \times 2$
	$L_0 = 1\,\mu\mathrm{m}$
Fluid:	Newtonian::water (fluid 1)
BC:	mass-flow (BC 3)
	$v_x = 1\,\frac{\mathrm{mm}}{\mathrm{s}}$
	pressure (BC 4)

The flow field can be seen in figure 147.

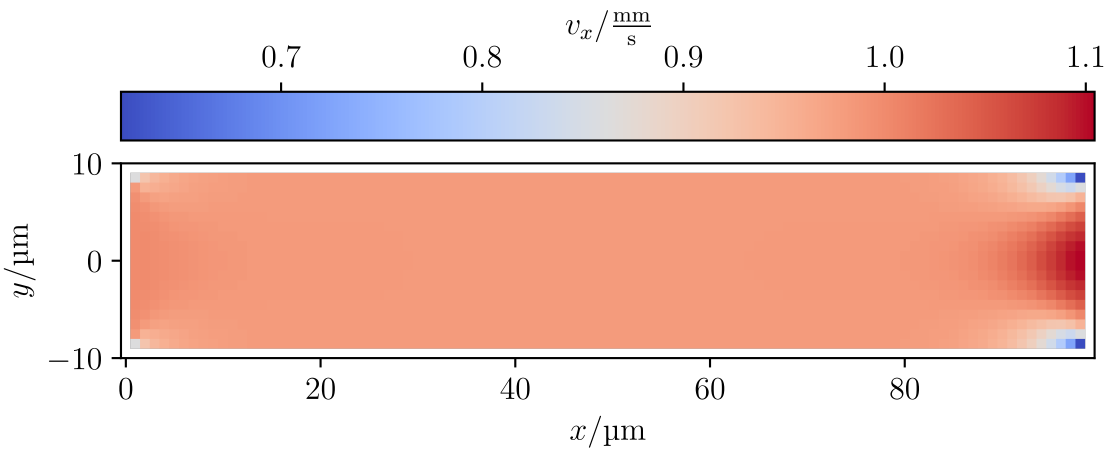


Figure 147: x-component of the velocity in a 2D channel of free-slip walls driven by a fixed mass-flow.

Around the center, the flow field is remarkably homogeneous with deviations of around $3\epsilon_{\mathrm{fp32}}$ (see appendix O). This means a near-perfect plug flow is created and the free-slip boundaries evidently work well. However, at the edges (particularly the outlet) an artefact is quite evident. The reason for these is, that the free-slip boundaries are difficult to implement near other types of non-trivial boundaries. If the free-slip walls cause the populations to be reflected into an e.g. pressure wall, strictly speaking, the boundary algorithm for that wall would need to be called. This however cannot be clearly determined, as those require information from the source node of the population, which technically speaking due to the reflection would be a wall, which cannot provide this information. To avoid this conundrum, free-slip walls are implemented in *FluidX3D* to behave like no-slip walls in case the population cannot be easily reflected. These causes velocity-minima to appear near them.

Q.2. Diagonal

A geometry similar to the previous section is used here. The main difference being that simulation 35 is rotated by 45°. This introduces stair-casing at the channel edge.

35 Water in a diagonal free-slip channel

Box:	$100 \times 100 \times 2$
	$L_0 = 1\,\mu\text{m}$
Fluid:	Newtonian::water (fluid 1)
BC:	mass-flow (BC 3)
	$\lvert\vec{v}\rvert = \sqrt{2}\,\frac{\text{mm}}{\text{s}}$
	pressure (BC 4)

As is immediately obvious from the flow field depicted in figure 148, this acts like surface roughness and slows down the flow near the walls despite the walls technically being free-slip walls.

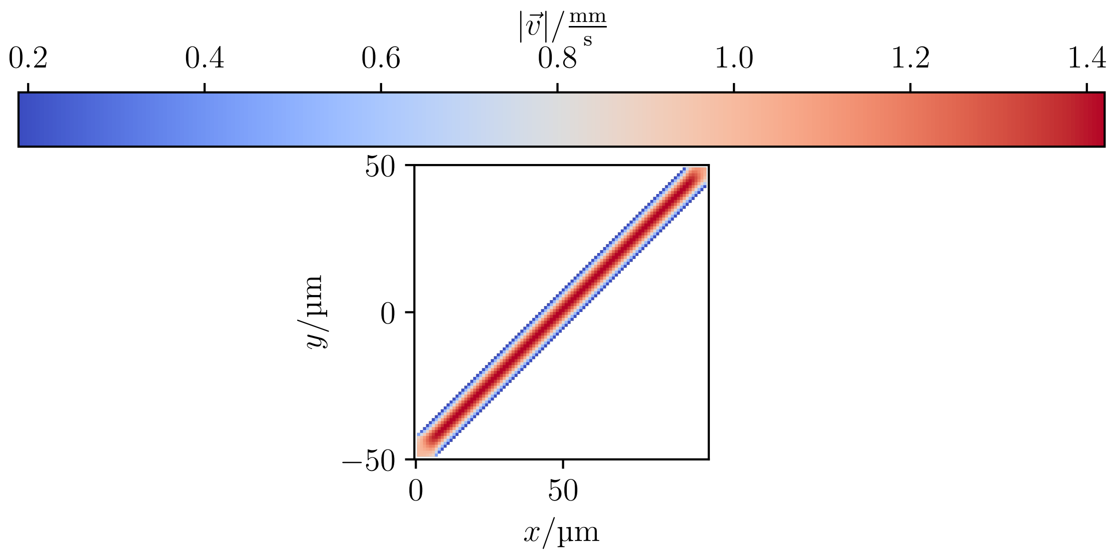


Figure 148: Magnitude of the velocity in a diagonal 2D channel of free-slip walls driven by a fixed mass-flow.

Rotating the resulting flow field by 45° aligns the velocity and brings the plot in a more easily readable form. The axis along which the fluid flows is addresses as l, white the other one is termed r. There result can be seen in figure 149.

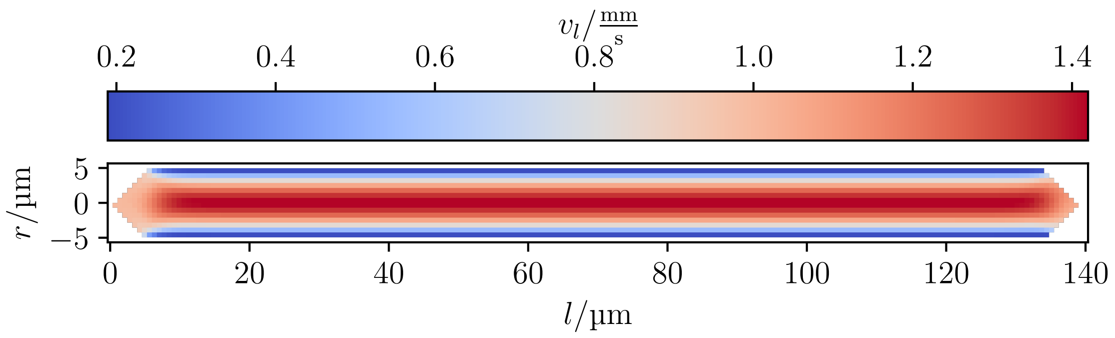


Figure 149: l-component of the velocity in a diagonal 2D channel of free-slip walls driven by a fixed mass-flow rotated by 45° to align with the axis.

Aside from the inlet and outlet this looks remarkably like a Poiseuille-flow and as one can see from figure 150 it is. The stair-casing at the edge acts like a surface roughness, that turns the free-slip boundaries into effectively no-slip walls. Therefore, the free-slip boundaries are not accurate if the flow has components parallel to surfaces approximated by stair-casing.

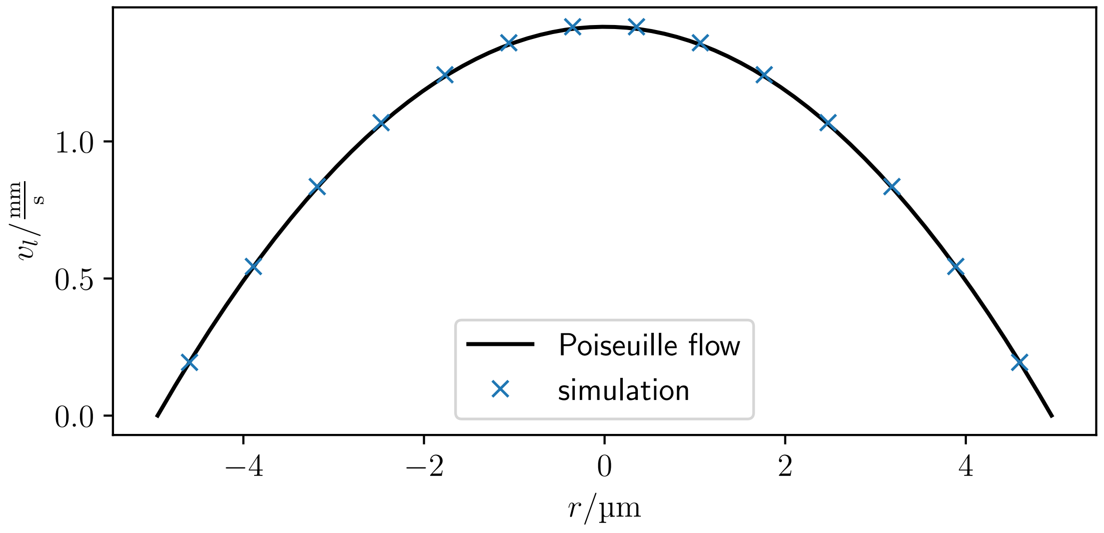


Figure 150: l-component of the velocity in the rotated channel from figure 149 cut along the r-axis in the center of the channel. For comparison a Poiseuille-flow with the same maximum velocity.

Q.3. Nozzle

Free-slip boundaries can be used to approximate free surfaces (see for example reference [67]). Therefore, the switch from no-slip to free-slip boundaries can be used to model the end of a nozzle. In simulation 36 a cylindrical channel has been used to model this.

36 Water in a partial free-slip cylindrical nozzle

Box:	$202 \times 22 \times 22$
	$L_0 = 1\,\mu\mathrm{m}$
Fluid:	Newtonian::water (fluid 1)
BC:	mass-flow (BC 3)
	$v_x = 1\,\frac{\mathrm{mm}}{\mathrm{s}}$
	pressure (BC 4)

Given that the previous section has shown, that free-slip boundaries do not accurately reproduce when stair-casing is involved, this is of particular interest. In this channel the circular walls obviously need approximation. However, in flow direction, the wall is straight. Still, some populations do travel in the problematic direction. But as can be seen in figure 151 this is not an issue and the flow is produced as expected. The plug flow is remarkably homogeneous with deviations of around $3\epsilon_{\mathrm{fp32}}$ (see appendix O) as is the case for straight walls. Note the flow spreading out in figure 152 as is expected at the end of a nozzle.

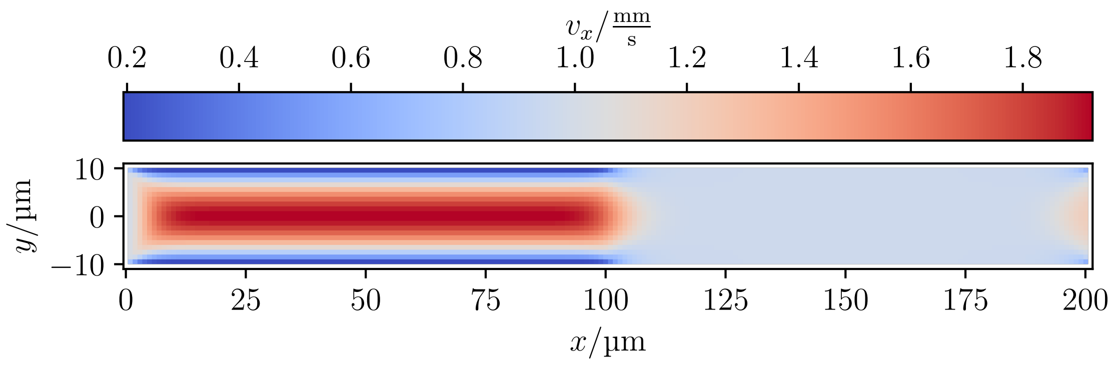


Figure 151: x-component of the velocity in a 3D channel of no-slip walls, which change to free-slip walls in the center (in respect to x) driven by a fixed mass-flow. This approximates the end of a printer nozzle.

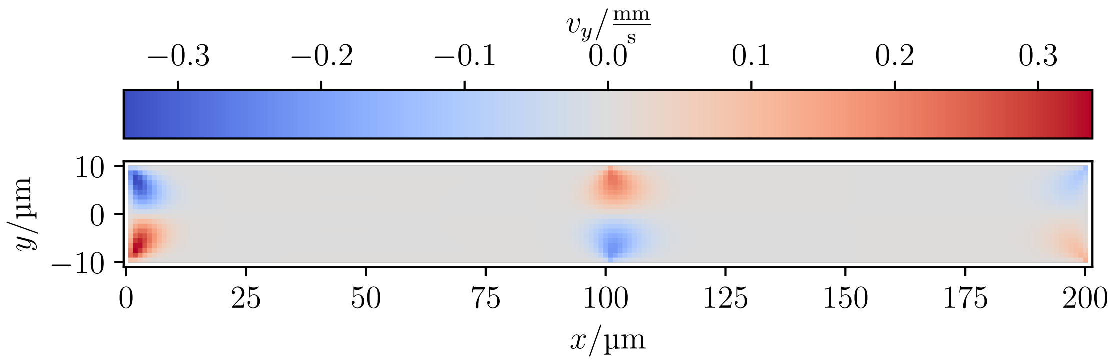


Figure 152: y-component of the velocity in a 3D channel of no-slip walls, which change to free-slip walls in the center (in respect to x) driven by a fixed mass-flow. This approximates the end of a printer nozzle.

R. Validation of Reynolds-Scaling

A validation simulation is provided for each of the cases listed in section 16.

R.1. Newtonian fluid

The example fluid for a Newtonian fluid is of course water (fluid 1). Using the analytical solution for 2D channels (see appendix F.1.1) an estimate for a required pressure difference to archive a certain velocity can be given. This is done by assuming the large width for the whole channel. Due to the actually present geometry the real velocity will however be quite a bit smaller than intended. For this simulation the velocity target is $0.1\,\frac{\text{mm}}{\text{s}}$ and consequently, the pressure difference is set to 64 mPa. Assuming the target velocity is achieved, it follows that $Re < 0.005$ and $Ma \approx 3 \times 10^{-5}$. These are both in an acceptable range, therefore scaling the Reynolds number by 10 is fine. If the velocity is smaller as expected, the condition is still fulfilled. The simulation (simulation 37) is performed with both $S_{Re} = 1$ and $S_{Re} = 10$.

37 Newtonian validation — Reynolds-Scaling

Box:	$202 \times 52 \times 2$
	$L_0 = 1\,\mu\text{m}$
Fluid:	Newtonian::water (fluid 1)
BC:	pressure (BC 4)
	$\Delta p = 64\,\text{mPa}$

The resulting velocity field can be seen in figures 153 and 154 for $S_{Re} = 1$ and $S_{Re} = 10$ respectively.

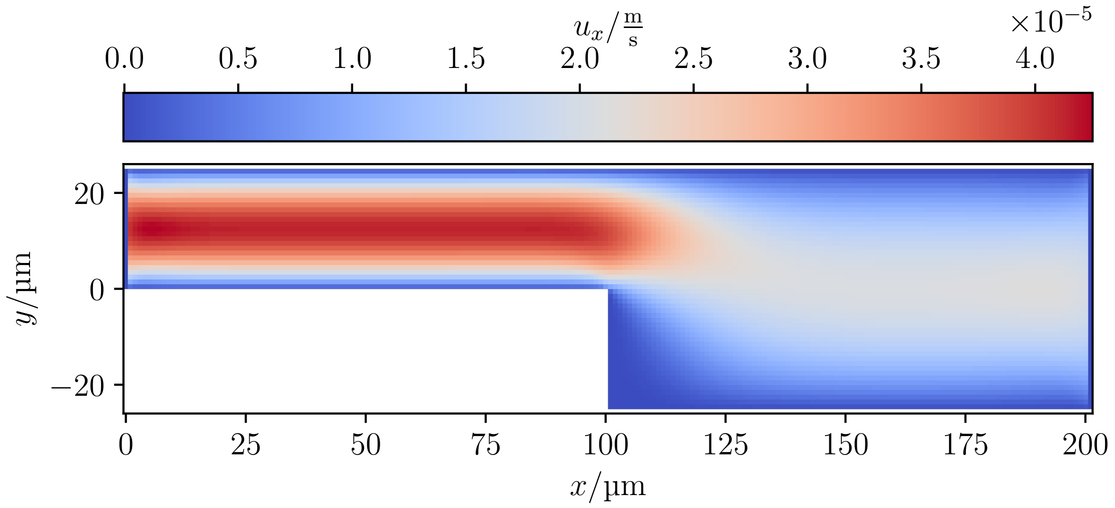


Figure 153: Plot of the x component of the velocity field for $S_{Re} = 1$.

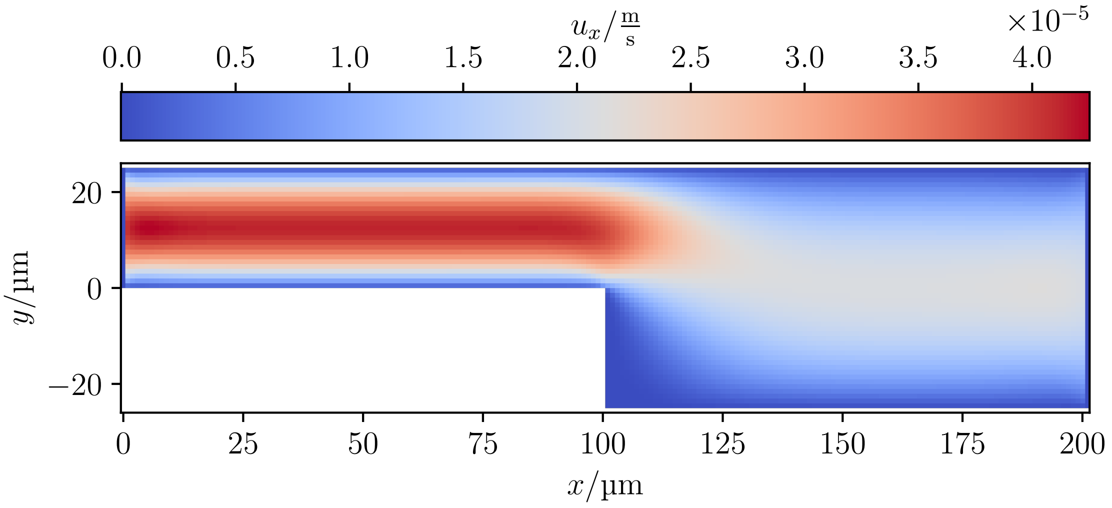


Figure 154: Plot of the x component of the velocity field for $S_{Re} = 10$.

One can see, that the maximum velocity is just a little less than half of the estimated value. This is due to the fact, that the pressure gradient over the wider section is smaller than assumed. This is due to the higher resistance of the narrower section causing a larger pressure drop. The reduced pressure drop in the wider section, can be seen in figures 155 and 156.

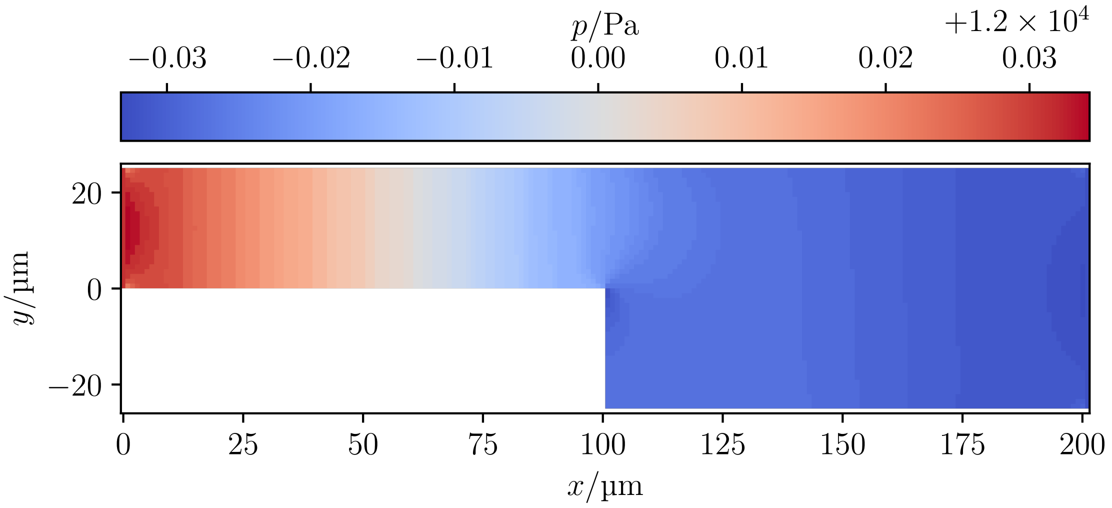


Figure 155: Plot of the pressure field for $S_{Re} = 1$.

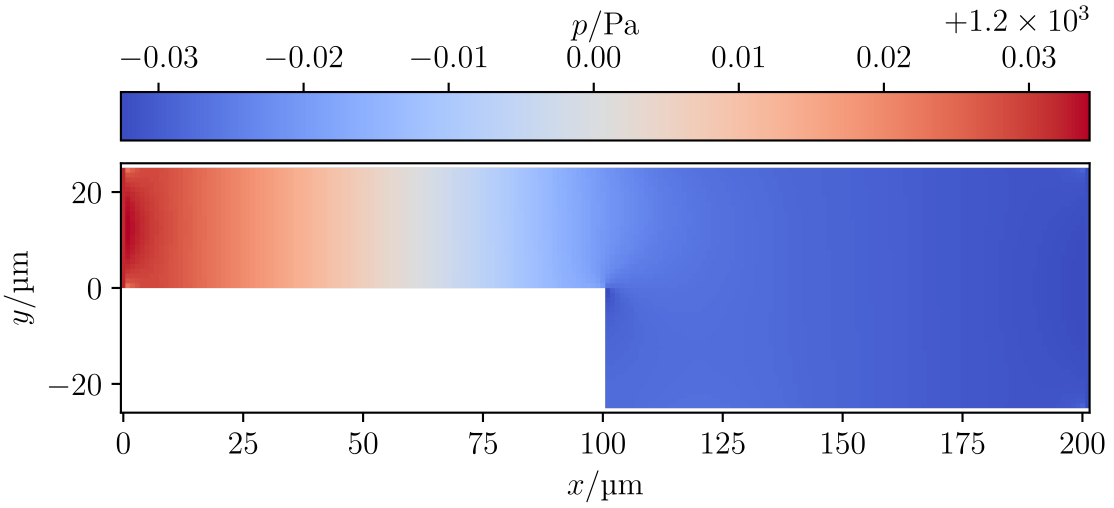


Figure 156: Plot of the pressure field for $S_{Re} = 10$.

These figures also show the change in the absolute value of the pressure, while the pressure difference stays the same. This was discussed in section 16.4.2. It should be stressed, that this is really an absolute difference, that stays the same, not a relative one. This is unintuitive, but by design. The color banding in figure 155 is due to the discretization error caused by saving the data in reduced precision (see section 14). This is an example of 32 bits not being sufficient and the ϵ becoming visible (see appendix O). In total, these results demonstrate the accuracy and validity of Reynolds-Scaling for a Newtonian fluid.

R.2. Generalized Newtonian fluid

Here a Carreau-Yasuda (CY) fluid (see section 3.5.1) is used as an example. The shear rate should be high enough ($Wi > 1$) for the shear-thinning behavior to show (see section 3.5.1). Still, the Reynolds number may not be too high. To archive this the channel was scaled down by a factor of 10 for simulation 38 compared to the Newtonian case.

38 Carreau-Yasuda Reynolds-Scaling validation

Box:	$202 \times 52 \times 2$
	$L_0 = 0.1\,\mu\mathrm{m}$
Fluid:	CY::mc0_59 (fluid 9)
BC:	pressure (BC 4)
	$\Delta p = 2.144\,\mathrm{kPa}$

The pressure difference is set to 2.144 kPa. Using the zero-shear viscosity, this should result in a velocity of roughly $10\,\frac{\mathrm{mm}}{\mathrm{s}}$. The actual value will be a little higher due to the shear-thinning nature of CY. With the expected velocity the dimensionless numbers are estimated to be $Re \approx 0.0015$ and $Ma \approx 9 \times 10^{-6}$. Therefore, Reynolds-Scaling is applicable. Even with a higher velocity as expected, the Reynolds number is unlikely to become to high to allow for scaling. Like the Newtonian case, the Generalized Newtonian fluid also shows the scaled startup behavior with Reynolds-Scaling. This can be seen in figure 157.

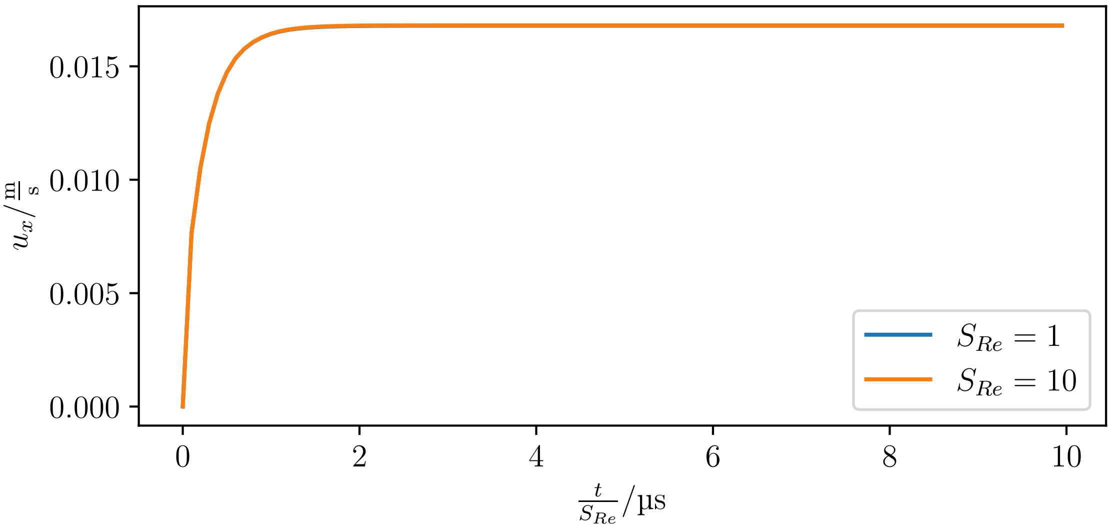


Figure 157: Plot of the x component of the velocity field 3 µm behind the center of the inlet as a function of scaled time.

This figure also shows, that the actual maximum velocity is higher than the estimate - as was expected. With the Reynolds number adjusted accordingly, it is however still small enough to justify Reynolds-Scaling. This shows the difficulty of evaluating whether Reynolds-Scaling is valid before running the simulation for complex fluids.

R.3. Viscoelastic fluid

The CY fluid from appendix R.2 was designed to have the same shear-thinning behavior as corresponding PTT fluid. Consequently, the same simulation can be performed with this fluid, without altering the Reynolds or Mach number. This is referred to as simulation 39.

39 PTT validation — Reynolds-Scaling

Box:	$202 \times 52 \times 2$
	$L_0 = 0.1\,\mu\text{m}$
Fluid:	PTT::mc0_59 (fluid 3)
BC:	pressure (BC 4)
	$\Delta p = 2.144\,\text{kPa}$

The steady-state value is close to identical for the scaled and unscaled simulations as was expected. This can be seen in figure 158.

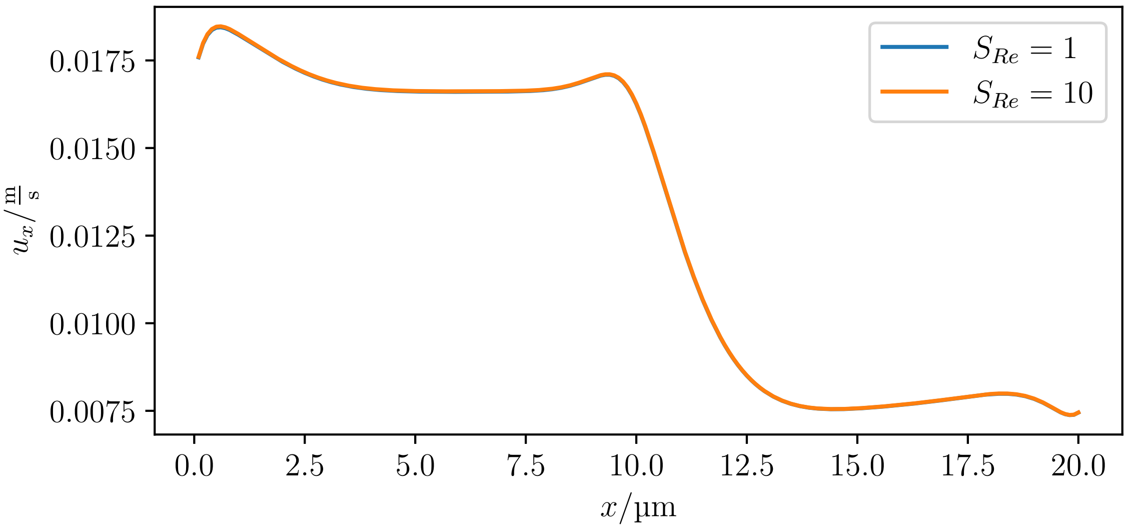


Figure 158: Plot of the x component of the velocity field along the center-line of the narrow part of the channel. The maximum L1 error is $|e_{\text{L1}}| < 1.5 \times 10^{-3}$ (see appendix S).

The error is worse than for the Newtonian simulation. It shall be noted, that due to the fact, that viscoelastic fluids are retarded, the startup takes considerably longer. Consequently, viscoelastic simulations are extremely slow to reach equilibrium. This simulation has likely not fully relaxed yet. This might drive the error higher than it actually is.

R.4. Generalized Newtonian with a cell in shear

Two different simulations are considered for cells in a Generalized Newtonian fluid. First, simulation 40 is performed. It represents the standard Roscoe problem.

40 Carreau-Yasuda Reynolds-Scaling with a cell in shear

Box:	$200 \times 202 \times 200$
	$L_0 = 0.\overline{1}\,\mu\mathrm{m}$
Fluid:	CY::mc0_59 (fluid 9)
BC:	velocity (BC 2)
	$\dot{\gamma} = 18 \times 10^3\,\frac{1}{\mathrm{s}}$
Cell:	Young's modulus $E = 2.072\,\mathrm{kPa}$
	Radius, Resolution $R = 2\,\mu\mathrm{m}, 18$
	Capillary number $Ca_{\mathrm{K}} \approx 1$

The x component of the point typically used for evaluation (see section 10) can be seen in figure 159.

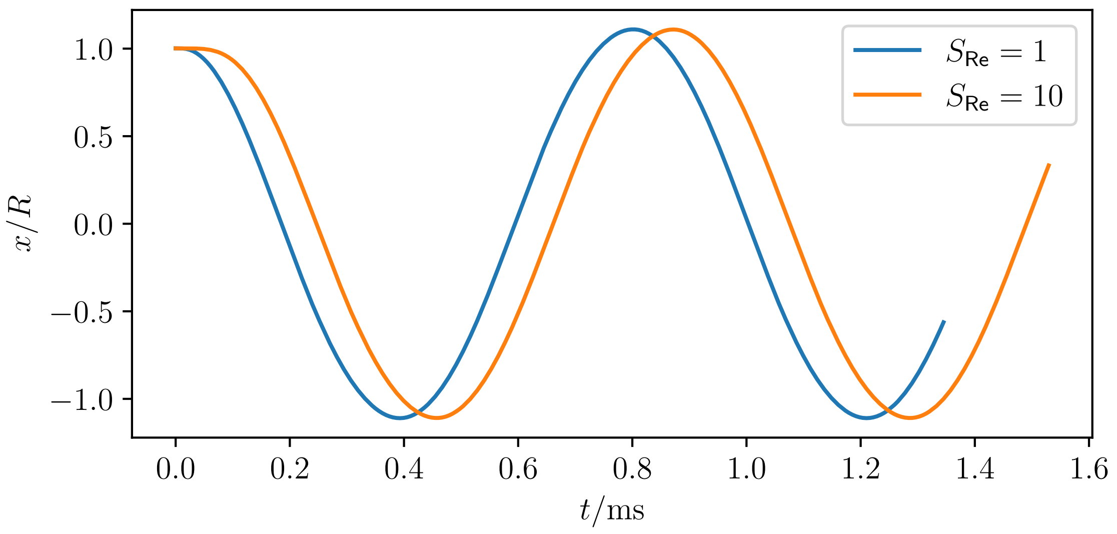


Figure 159: Plot of the normalized x component of the surface point of the cell, which is originally on the x-axis as a function of time. The cell with scaling takes longer to start and has a slightly smaller tank-treading frequency.

The unscaled data is slightly shorter because the simulation has been killed for exceeding the standard amount of allotted time. Obviously, there is an offset due to the extended startup time. The offset is approximately $70\,\mu\mathrm{s}$. It has been removed in figure 160 by shifting of the data.

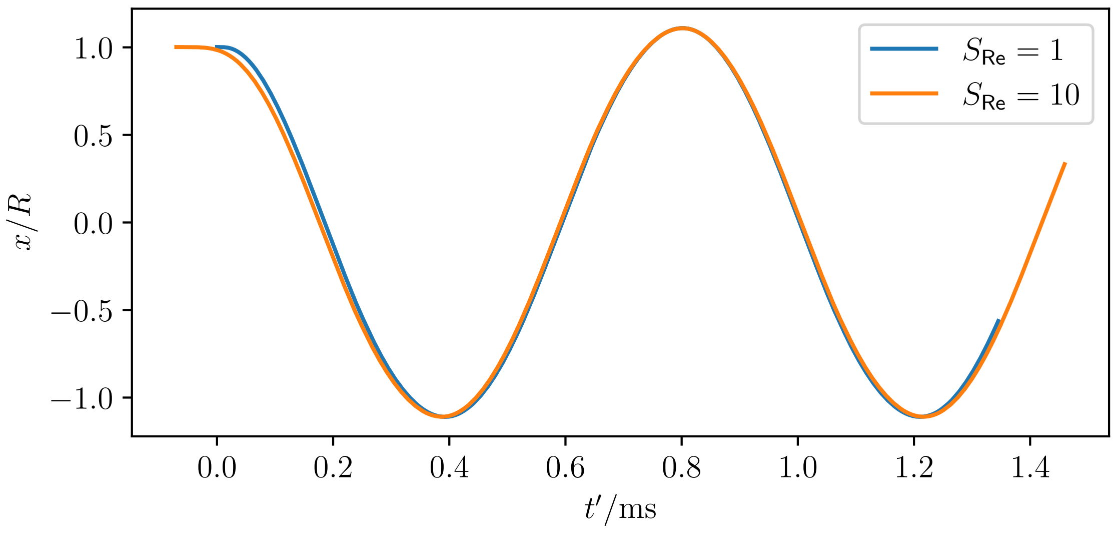


Figure 160: Plot of the normalized x component of the surface point of the cell, which is originally on the x-axis as a function of time. The cell with scaling takes longer to start and has a slightly smaller tank-treading frequency.

This makes it clear, that the tank-treading frequency is ever so slightly smaller in the scaled case. This difference is approximately 1.4 %. Within the typical error range for Roscoe simulations (see appendix X), this can be considered to be identical. The scaled simulation also produces a slightly larger alignment angle. According to Roscoe theory (see section 9), this should be associated with a higher frequency. This minuscule difference in adherence to the theory can actually be seen in figures 161 and 162.

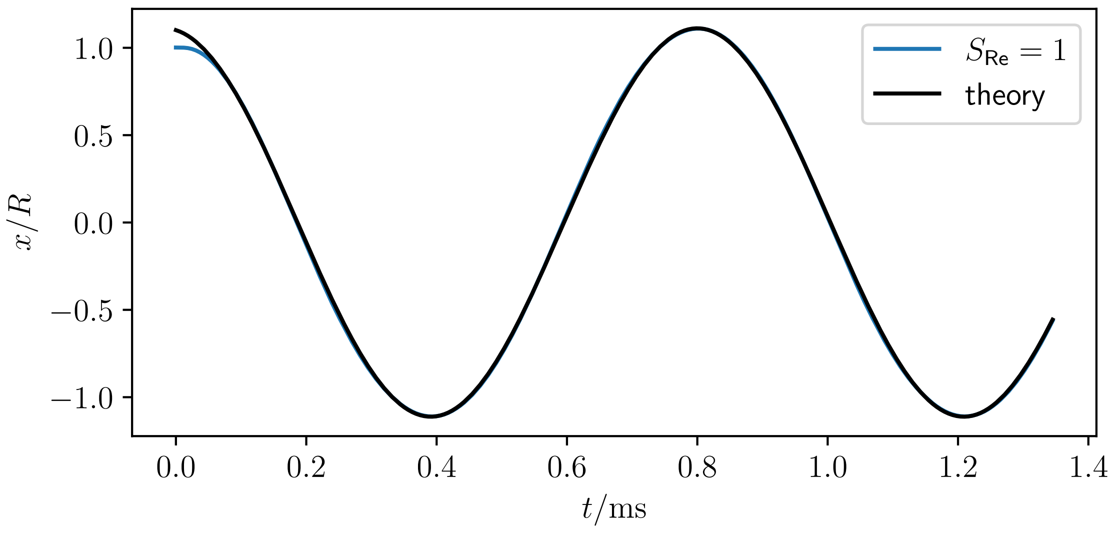


Figure 161: Plot of the normalized x component of the surface point of the cell, which is originally on the x-axis as a function of time. Aside from the startup, the match to theory is near perfect.

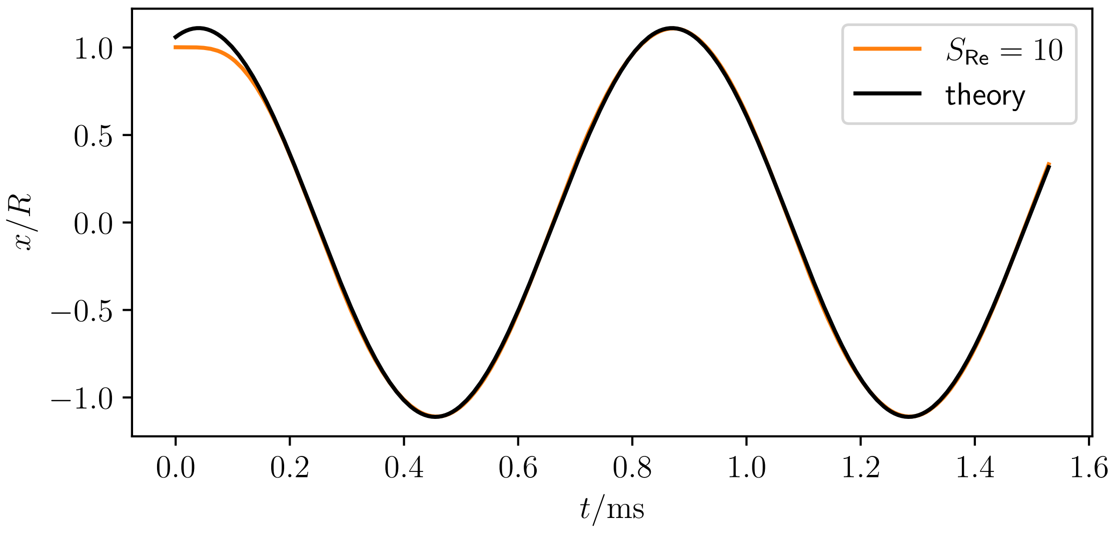


Figure 162: Plot of the normalized x component of the surface point of the cell, which is originally on the x-axis as a function of time. Aside from the startup, the match to theory is near perfect.

Looking extremely closely, one can see, that the scaled line is slightly higher than predicted by theory shortly after the maximum. Such tiny inaccuracies are irrelevant here, but can become an issue in a less forgiving flow.

R.5. Generalized Newtonian with a cell with geometric change

The cell from appendix R.4 is put through the suddenly expanding geometry from appendix R.2. Simulation 41 should produce locally changing velocities in the wake of the cell. These should be reproduced at the wrong timescale.

41 Carreau-Yasuda Reynolds-Scaling with a cell at changing geometry

Box:	$1002 \times 102 \times 52$
	$L_0 = 0.1\,\mu\mathrm{m}$
Fluid:	CY::mc0_59 (fluid 9)
BC:	pressure (BC 4)
	$\Delta p = 12\,\mathrm{kPa}$
Cell:	Young's modulus $E = 2.072\,\mathrm{kPa}$
	Radius, Resolution $R = 0.6\,\mu\mathrm{m}, 6$

As before, the curves are slightly shifted to account for the startup delay. In this case, the shift is approximately 17 µs. The position of the cell can be expressed using its center $\vec{x}_{\mathrm{COM}}$. This is calculated from the N mesh points $\vec{x}_i$ as follows.

$$\vec{x}_{\mathrm{COM}} = \frac{1}{N} \sum_{i}^{N} \vec{x}_i \tag{828}$$

The x and y components of this point can be seen in figures 163 and 164.

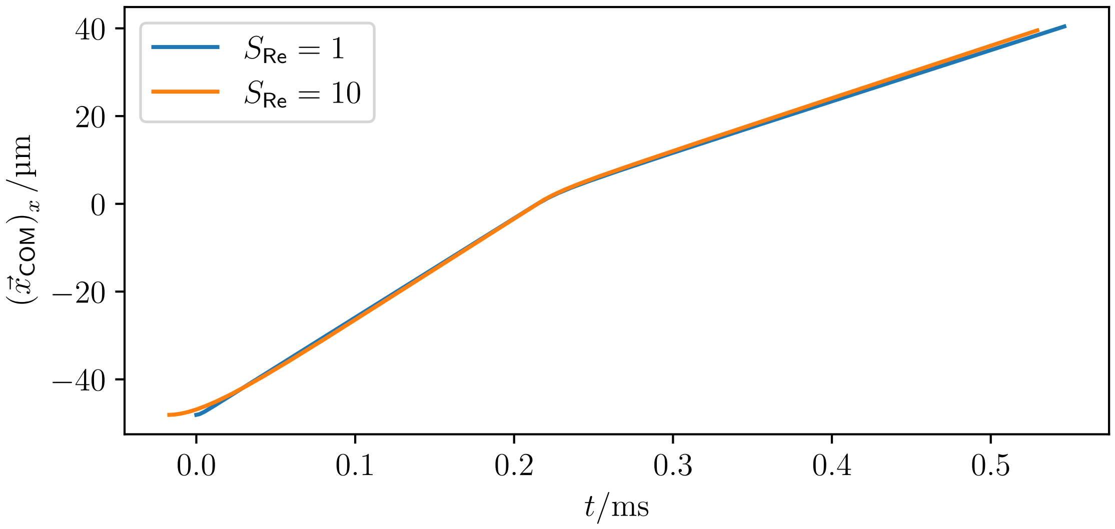


Figure 163: Plot of the x component of the cell center as a function of time. Aside from the startup, the agreement is remarkable.

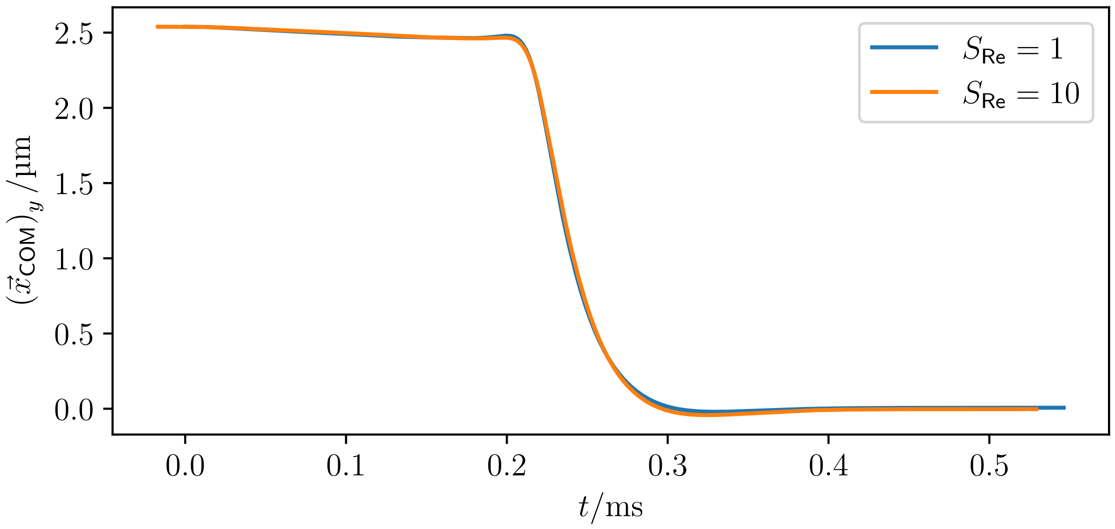


Figure 164: Plot of the y component of the cell center as a function of time. Aside from the startup, the agreement is remarkable.

At the end the unscaled simulation is slightly above the center-line (corresponds to a value of 0), while the scaled one is slightly below. The cell is slightly closer to the center in the scaled simulation. Consequently, it is a little faster, which can be seen in the x component. The observed agreement is remarkable, given, that scaling should be invalid here by construction. This simulation is not in steady-state and should therefore show discrepancies. The agreement of the aspect ratio D_{R} (see section 11) is worse. This can be seen in figure 165.

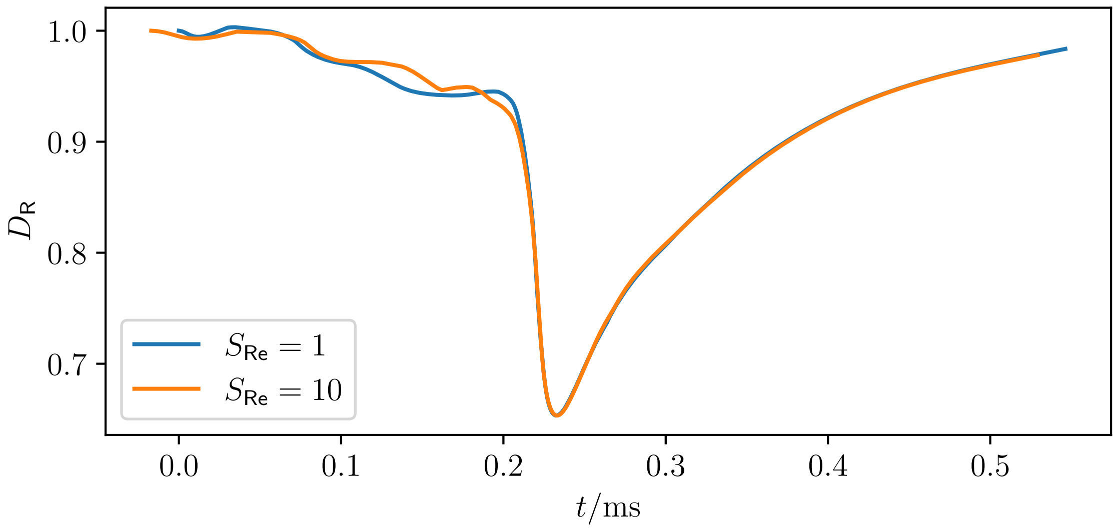


Figure 165: Plot of the aspect ratio of the cell as a function of time. Before the channel widens, the agreement is poor.

The discrepancy before the widening of the channel is likely caused by a small surface instability (see section 15.2). There is also a small discrepancy in the velocity. In total, Reynolds-Scaling is surprisingly valid for this simulation.

R.6. Viscoelastic Reynolds-Scaling with a cell in shear

As before in appendix R.4, a Roscoe simulation is performed. Simulation 42 has the same parameters as aforementioned Generalized Newtonian simulation, but switches the fluid to a viscoelastic one.

42 Viscoelastic Reynolds-Scaling with a cell in shear

Box:	$200 \times 202 \times 200$
	$L_0 = 0.\overline{1}\,\mu\mathrm{m}$
Fluid:	PTT::mc0_59 (fluid 3)
BC:	velocity (BC 2)
	$\dot{\gamma} = 18 \times 10^3\,\frac{1}{\mathrm{s}}$
Cell:	Young's modulus $E = 2.072\,\mathrm{kPa}$
	Radius, Resolution $R = 2\,\mu\mathrm{m}, 18$
	Capillary number $Ca_\mathrm{K} \approx 1$

The x component of the point typically used for evaluation (see section 10) can be seen in figure 166.

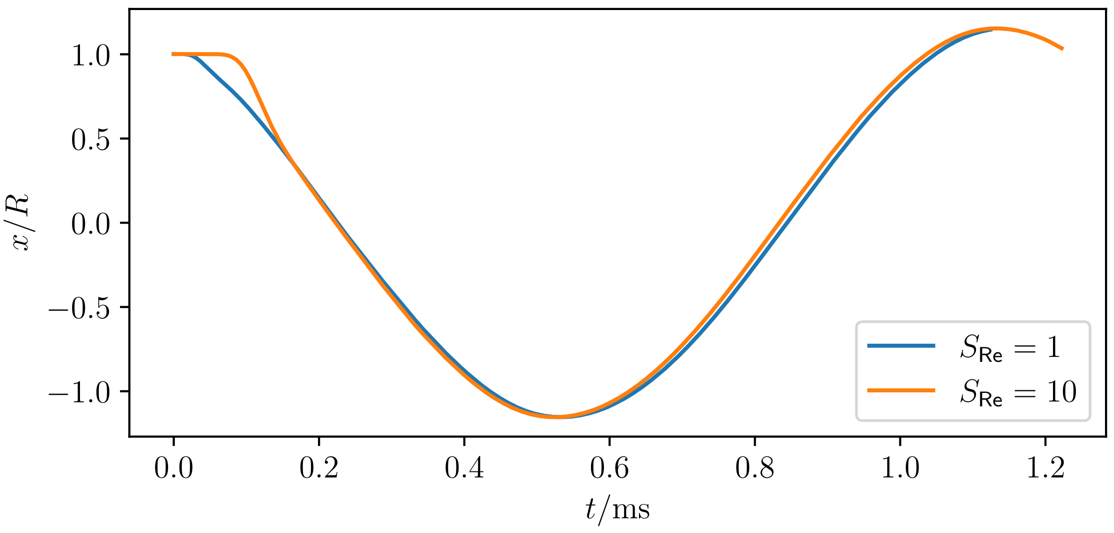


Figure 166: Plot of the normalized x component of the surface point of the cell, which is originally on the x-axis as a function of time. The cell with scaling takes longer to start.

Obviously, there is an offset due to the extended startup time. The offset is approximately $10\,\mu\mathrm{s}$. It has been removed in figure 167 by shifting of the data.

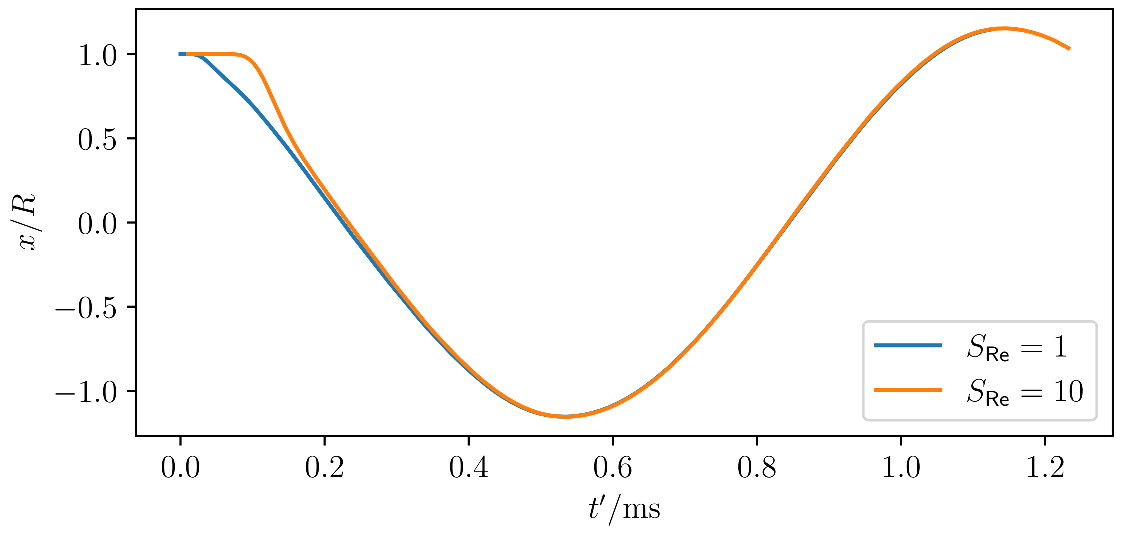


Figure 167: Plot of the normalized x component of the surface point of the cell, which is originally on the x-axis as a function of time. The cell with scaling takes longer to start.

The agreement between the scaled and unscaled simulation is near perfect. Even the tank-treading frequency matches within the limits of human perception. Interestingly, the tank-treading frequency is significantly smaller than for the CY case. This is an interesting topic for a later discussion (see section 21). It is still possible to express these curves using Roscoe theory, as can be seen in figures 168 and 169.

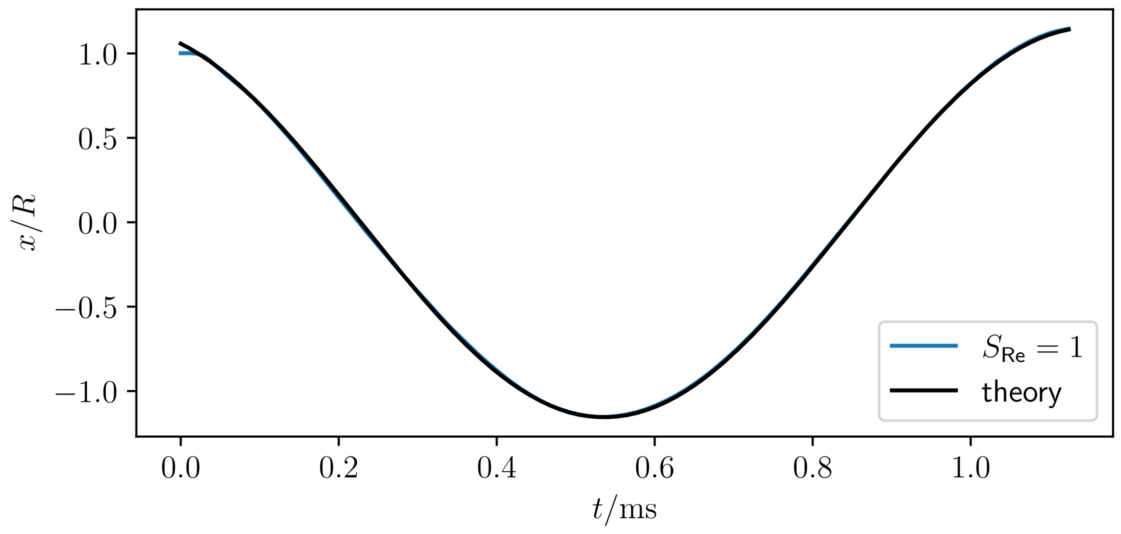


Figure 168: Plot of the normalized x component of the surface point of the cell, which is originally on the x-axis as a function of time. Aside from the startup, the match to theory is near perfect.

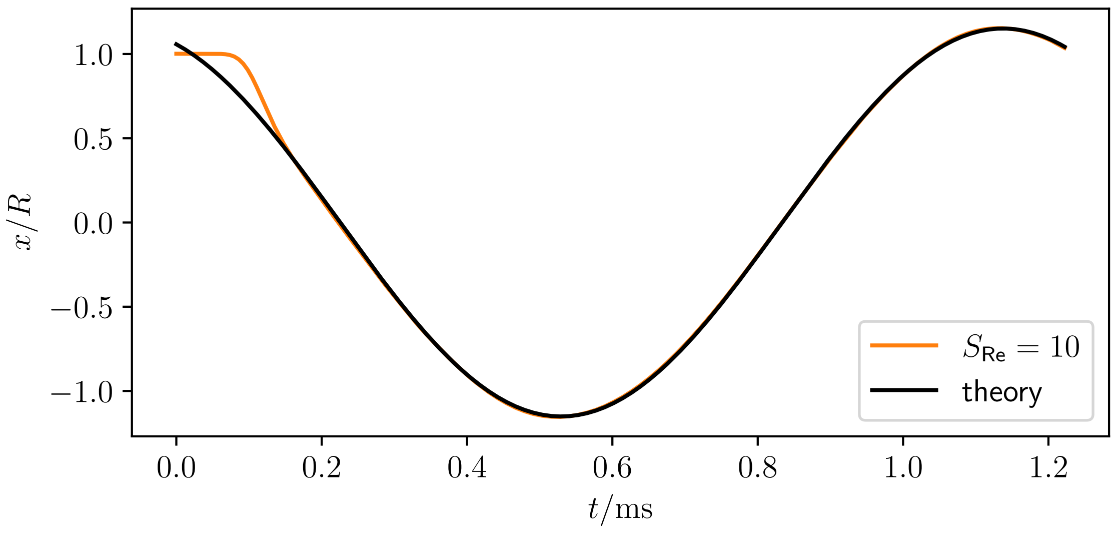


Figure 169: Plot of the normalized x component of the surface point of the cell, which is originally on the x-axis as a function of time. Aside from the startup, the match to theory is near perfect.

One can see, that the additional timescale did not influence the simulation. In total, Reynolds scaling is also valid for viscoelastic Roscoe simulations as was expected.

R.7. Viscoelastic Reynolds-Scaling with a cell in Poiseuille flow

The Poiseuille geometry (see appendix P.2) does violate the criteria for Reynolds-Scaling. However, it is not as dynamic as the suddenly widening channel. Therefore, simulation 43 is performed as an intermediate step.

43 Viscoelastic Reynolds-Scaling with a cell in Poiseuille flow

Box:	$1002 \times 52 \times 52$
	$L_0 = 0.1\,\mu\text{m}$
Fluid:	PTT::mc0_59 (fluid 3)
BC:	pressure (BC 4)
	$\Delta p = 18\,\text{kPa}$
Cell:	Young's modulus $E = 2.072\,\text{kPa}$
	Radius, Resolution $R = 0.6\,\mu\text{m}, 6$

The x and y components of the position of the cell $\vec{x}_{\text{COM}}$ are shown in figures 170 and 171.

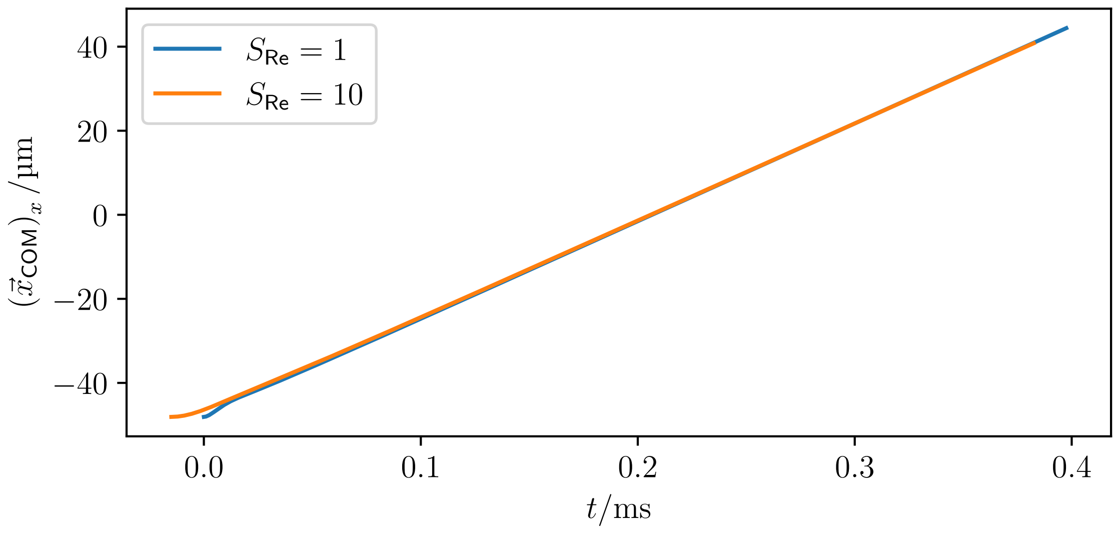


Figure 170: Plot of the x component of the cell center, as a function of time. Aside from the startup, agreement is remarkable.

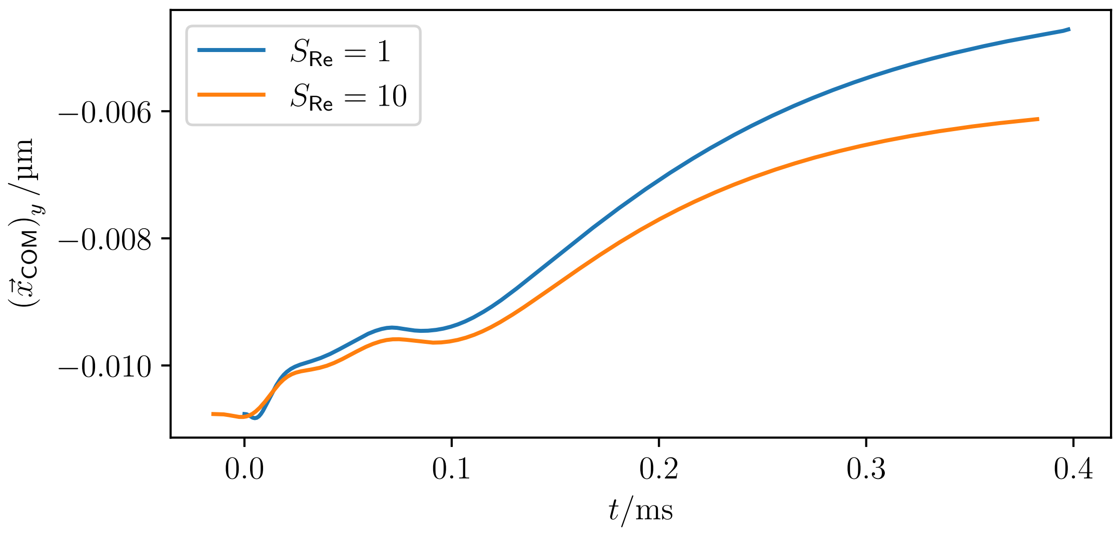


Figure 171: Plot of the y component of the cell center, as a function of time. These curves differ noticeably.

Due to the delayed startup, the curves are offset by approximately 15 µs. Aside from the startup, the x component shows near perfect agreement. Even the velocity is reproduced. However, it should be noted, that the curves only fully align after half the simulation time has passed. This illustrates, how viscoelatic simulations produce extended startup times. The curves for the y component differ. However, both still are approximately 0 as they should, so this does not matter. The agreement of th aspect ratio D_{R} (see section 11) can be seen in figure 172.

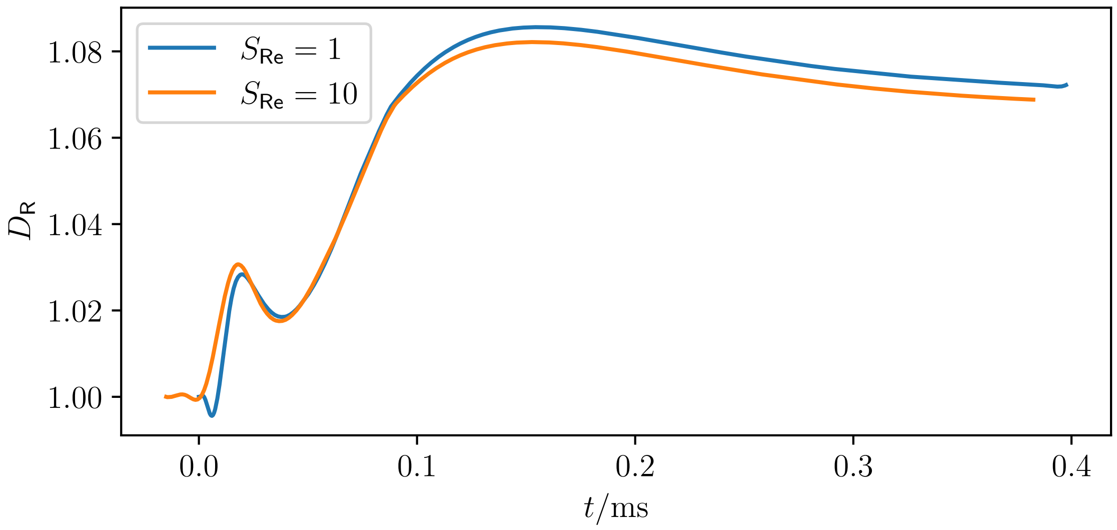


Figure 172: Plot of the aspect ratio of the cell as a function of time. Even after the startup, some discrepancy remains.

To the naked eye, the cells look identical. The observed difference of the deformation is most likely due to time-dependent behavior. This does not get reproduced accurately with Reynolds-Scaling. This demonstrates, that Reynolds-Scaling is technically not valid for this simulation. However, the observed difference is rather small. It is most definitely suited for qualitative analysis. Furthermore, this error is not too big to preclude quantitative analysis.

R.8. Viscoelastic Reynolds-Scaling with a cell with geometric change

As before in appendix R.5, a simulation with the suddenly widening channel is performed. Simulation 44 has the same parameters as aforementioned Generalized Newtonian simulation, but switches the fluid to a viscoelastic one.

44 Viscoelastic Reynolds-Scaling with a cell at changing geometry

Box:	$1002 \times 102 \times 52$
	$L_0 = 0.1\,\mu\mathrm{m}$
Fluid:	PTT::mc0_59 (fluid 3)
BC:	pressure (BC 4)
	$\Delta p = 12\,\mathrm{kPa}$
Cell:	Young's modulus $E = 2.072\,\mathrm{kPa}$
	Radius, Resolution $R = 0.6\,\mu\mathrm{m}, 6$

The x and y components of the position of the cell $\vec{x}_{\mathrm{COM}}$ are shown in figures 173 and 174.

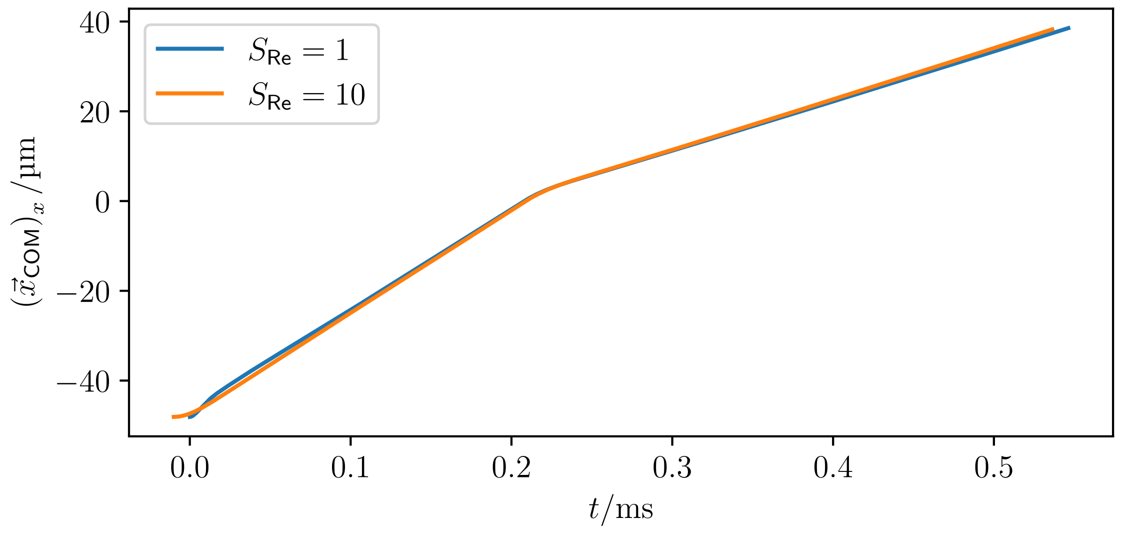


Figure 173: Plot of the x component of the cell center as a function of time. Aside from the startup, the agreement is remarkable.

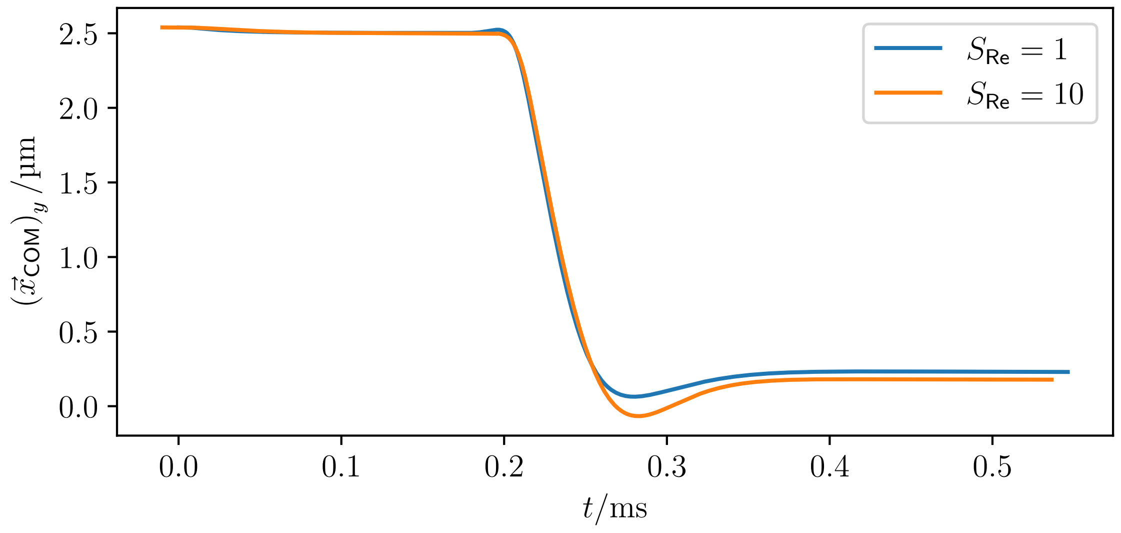


Figure 174: Plot of the y component cell center as a function of time. Aside from the startup, the agreement is remarkable.

Due to the startup, the curves are offset by approximately 10 µm. The agreement between the scaled and unscaled simulation is near perfect for the x component. The plot of the y component shows, that the cell is a bit closer to the center (corresponds to 0) in the scaled simulation. Consequently, it is a little faster, which can be seen in the x component. The observed agreement is remarkable, given, that scaling should be invalid here by construction. This simulation is not in steady-state and should therefore show discrepancies. Only the y component shows any real discrepancy and most of it is only present briefly. The agreement of the aspect ratio D_R (see section 11) is worse. This can be seen in figure 175.

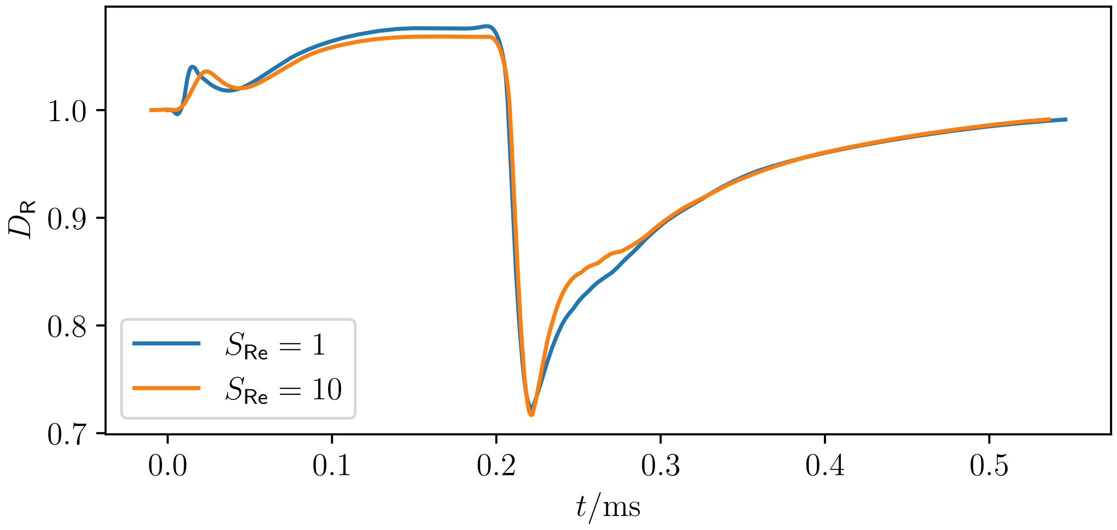


Figure 175: Plot of the aspect ratio of the cell as a function of time. The agreement is suboptimal before the channel widens and poor for a brief period shortly afterwards.

By optical inspection no reason for the discrepancy can be found. Both the scaled and unscaled cell look perfectly normal. Therefore, this is a real effect from the fact, that scaling is technically invalid in this simulation, due to it not being in stead-state. The discrepancy here is worse than for Generalized Newtonian fluids. Still, this result is usable. Reynolds-Scaling is still surprisingly valid in this scenario.

S. Error definitions

Error definitions can be used to nudge the error in a desired direction. To avoid any such effects, this thesis uses two standard error definitions. The L1 error and the L2 error. First, the L1 error is discussed.

S.1. L1 error

The L1 error is denoted e_{L1}. For a quantity a and is reference value a_{ref}, it is defined as follows.

$$e_{\mathrm{L1}} = \frac{a - a_{\mathrm{ref}}}{a_{\mathrm{ref}}} \tag{829}$$

Note, that the reference value is normally the theoretical prediction irrespective of which value is bigger. This error is signed. If the quantity is smaller than its reference, the error is negative. Otherwise, it is positive. If one desires to find the maximum L1 error, it is sensible to use the absolute value of the errors instead. The error is a percentage. It acts on individual values. If one inputs a dataset of N points, one retrieves N errors. The L2 error can be used to get a single value for the whole dataset.

S.2. L2 error

The L2 error is denoted e_{L2}. For a set of quantities a_i and their reference values $a_{i,\mathrm{ref}}$, it is defined as follows.

$$e_{\mathrm{L2}} = \sqrt{\frac{\sum_i (a_i - a_{i,\mathrm{ref}})^2}{\sum_i a_{i,\mathrm{ref}}^2}} \tag{830}$$

This can be viewed as an average of the individual L1 errors. Consequently, this also is a percentage. In contrast to the L1 error, it is not signed. It provides a single error value for a whole dataset. Similar to Pythagorean addition smaller contributions have less impact than larger ones. This concludes the discussion in the errors voluntarily used in this thesis. Comparison to literature often requires the use of the respective error definition of the publication one wishes to analyze.

T. Existing viscoelastic LBM algorithms

This appendix is mostly identical to it's previously published version [1]. Some extension for viscoelastic fluids have existed since the infancy of LBM. This thesis is interested in three-dimensional problems. Consequently, only preexisting 3D algorithms are covered here. Initially, the explicit modeling of a viscoelastic constitutive equation was avoided by the existing methods. Instead, the LB collision operator was modified. This simple change enables the description of a basic viscoelastic fluid obeying Jeffery's model [30]. Another option, with the same result is the use of second locally defined stress tensor [34]. Onishi *et al.* [26, 32] presented more elaborate schemes. A non-shear-thinning Oldroyd-B model was considered. In this, the polymer dumbbells were modeled with a microscopic Fokker-Planck equation. In addition to the Navier-Stokes solver, a secondary parallel LBM solver was employed by Wang *et al.* [33] to model the advection of the polymer stress. Malaspinas *et al.* [27] and Ma *et al.* [31] made use of the same approach for the Oldroyd-B and FENE-P models. Finite-difference [28, 29, 38] or finite-volume [37] schemes were introduced later as a solver for the viscoelastic stress. Those were coupled to the Navier-Stokes LBM in a hybrid algorithm. For these, the LBM handles the solvent and the finite-difference or finite-volume algorithm handles the polymer stress tensor. Sometimes the polymer stress tensor is replaced by its dimensional version, the polymer conformation tensor. From the base idea, this is what is done in *FluidX3D* as well. Numerical instability is commonly observed in literature. To combat this, the inclusion of an artificial diffusivity was proposed by some authors [27, 31, 33, 36]. A small error is introduced this way, as the constitutive equation and a potentially existing analytical solution do not include any diffusivity. However, these authors also archive the largest stable R_η. Other approaches include a special decomposition of polymer conformation tensor or the stress tensor [15]. This prevents unphysical deformation of the polymer solution, but at the cost of potentially diverging forces (see appendix E). The existing methods mentioned so far reach reasonable Weissenberg numbers with maximums from $Wi = 10$ [27] to $Wi = 10$ [28, 31, 38]. The viscosity ratio R_η as defined in eq. (335) is another stability criterion. This is often overlooked also much literature does mention it in passing. $R_\eta = 1$ is an upper bound for many existing methods [28, 29, 32, 33, 35]. Others report being limited below $R_\eta = 10$ [27, 31, 37]. Stability issues are given as the reason for this by Malaspinas *et al.*. Meaning, existing methods cannot reach above $R_\eta = 10$, even with artificial diffusivity and similar workarounds. A list of the publications mentioned here, is provided in table 10. The table contains their maximum archived Wi and R_η, as well as how their R_η was estimated.

Author	max(Wi)	max(R_η)	R_η estimation
Malaspinas [27]	10	9	In 4.3: $R_\nu = 0.1$ & equation (38)
Gupta [28]	100	0.7	In IV.B: $\frac{\eta_\mathrm{p}}{\eta_\mathrm{A}+\eta_\mathrm{p}} = 0.4$
Ma [31]	100	9	In 4.1: $\beta = \frac{\mu_\mathrm{s}}{\mu_\mathrm{s}+\mu_\mathrm{p}} = 0.1$
Onishi [32]	?	1	In 3.1: $\beta = \frac{\mu_\mathrm{p}}{\mu_\mathrm{s}+\mu_\mathrm{p}}$ & Table 1: $\beta = 0.5$
Wang [33]	"up to O(1)"	5	In II: $\eta_\mathrm{p} = c\eta_\mathrm{s}$ & FIG. 9: $c = 5$
Su [35]	10	1	In II: $\beta = \frac{\eta_\mathrm{s}}{\eta_\mathrm{s}+\eta_\mathrm{p}}$ & In IV.A: $\beta = 0.5$
Gupta [29]	?	0.7	In 2: $\frac{\eta_\mathrm{p}}{\eta_\mathrm{d}} = 0.4$ & $\eta_\mathrm{d} = \eta_\mathrm{A} + \eta_\mathrm{p}$
Kuron [37]	1	9	In 4.1: $\beta = 0.9$ & equations (14) & (15)
Gupta [38]	80	0.7	In 3: $\frac{\eta_\mathrm{p}}{\eta_\mathrm{c,d}+\eta_\mathrm{p}} = 0.4$
Onishi [26]	1000	1	In 3.2: $\frac{\eta_\mathrm{p}}{\eta_\mathrm{s}} = 1$

Table 10: Covered ranges of Wi and R_η for existing 3D LBM algorithms. Reproduced from [1]

U. Validation of shuffling algorithm

The validation consists of three standard geometries, shear-flow (see appendix P.1), 2D Poiseuille flow and 3D poiseuille flow (see appendix P.2). These are covered in this order in the following. This appendix reproduces the validations from [1]. It should be noted, that all these validations have been rerun with a current version of *FluidX3D*. This is to assure, that nothing of relevance has changed since the publication of the paper (see reference [1]). First, the simplest case, the shear-flow is discussed.

U.1. Shear-flow

The first validation scenario is a shear-flow. Simulation 45 is to designed to determine whether the simulation error is dependent on the shuffling fraction.

45 Shear shuffling fraction dependency

Box:	$2 \times 43 \times 2$
	$L_0 \approx 0.49\,\mu\text{m}$
Fluid:	PTT::alginate (fluid 13)
BC:	velocity (BC 2)

The result can be seen in figure 176.

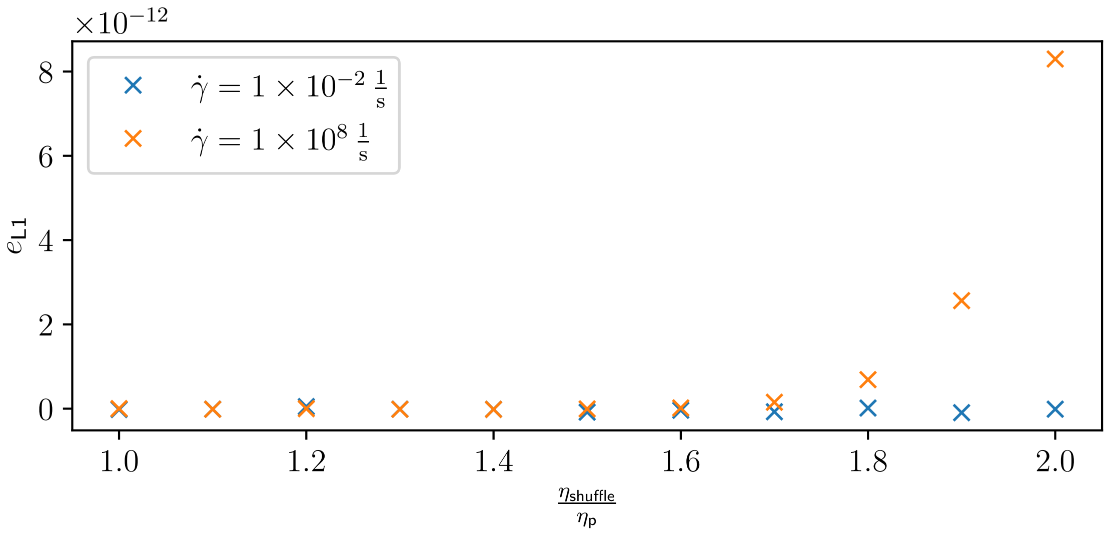


Figure 176: Plot of the L1 error (see appendix S) as a function of the shuffle viscosity.

Initially it seems quite clear, that there is a dependence. Zooming in yields figure 177.

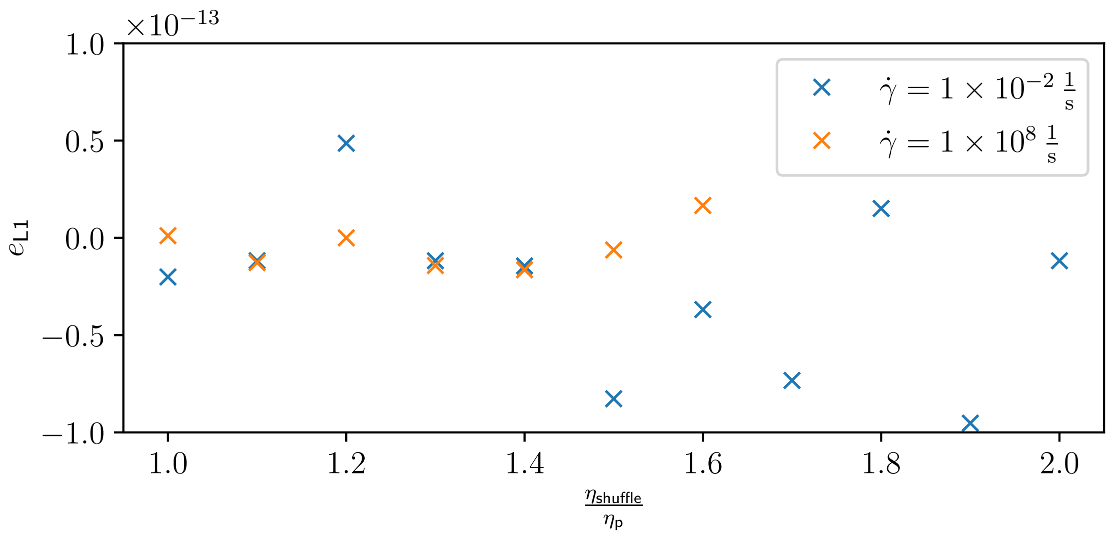


Figure 177: Plot of the L1 error (see appendix S) as a function of the shuffle viscosity. Limited to small errors.

Here it is quite clear, that the $\dot{\gamma} = 1\times10^{-2}\,\frac{1}{\mathrm{s}}$ point have random errors around zero. Most of the $\dot{\gamma} = 1\times10^{8}\,\frac{1}{\mathrm{s}}$ points do as well. To clear rising trend is due to relaxation. The errors here are miniscule. The lowest of them are only about $10\epsilon_{\mathrm{fp64}}$ (see appendix O). Normally simulation output in *FluidX3D* is written in fp32. For this it had to be changed to fp64, because the errors are so small. Relaxation is an asymptotic process. This process normally ends, when the magnitude of the noise is reached. This is so low here, that the relaxation takes extremely long. Additionally, higher shear-rates cannot accommodate as much Reynolds-Scaling as lower ones. This leads to these simulations ending before they are fully relaxed. The error is actually independent of the shuffling viscosity as it should be. The default choice of the $\eta_{\mathrm{shuffle}} = \eta_{\mathrm{p}}$ is therefore inconsequential. Next, the reproduction of the correct viscosity is examined in shear-flow. The simulations are compared to the available analytical solutions (see appendix F.2.3). The examined shear-rates are logarithmically spaced over the whole range of behaviors. Points are placed from the zero-shear Newtonian plateau, along the power-law region to the infinite-shear Newtonian plateau. In simulation 46, the shear-thinning fluid with the smallest R_η in this thesis is examined.

46 Methyl cellulose shuffling viscosity reproduction

Box:	$2 \times 43 \times 2$
	$L_0 \approx 0.49\,\mu\mathrm{m}$
Fluid:	PTT::mc0_49 (fluid 2)
BC:	velocity (BC 2)

The viscosity determined from the polymer-conformation tensor is compared to the analytical solution in figure 178. The polymer-conformation tensor is sampled at the center of the simulation volume and converted to the total fluid stress to retrieve the viscosity.

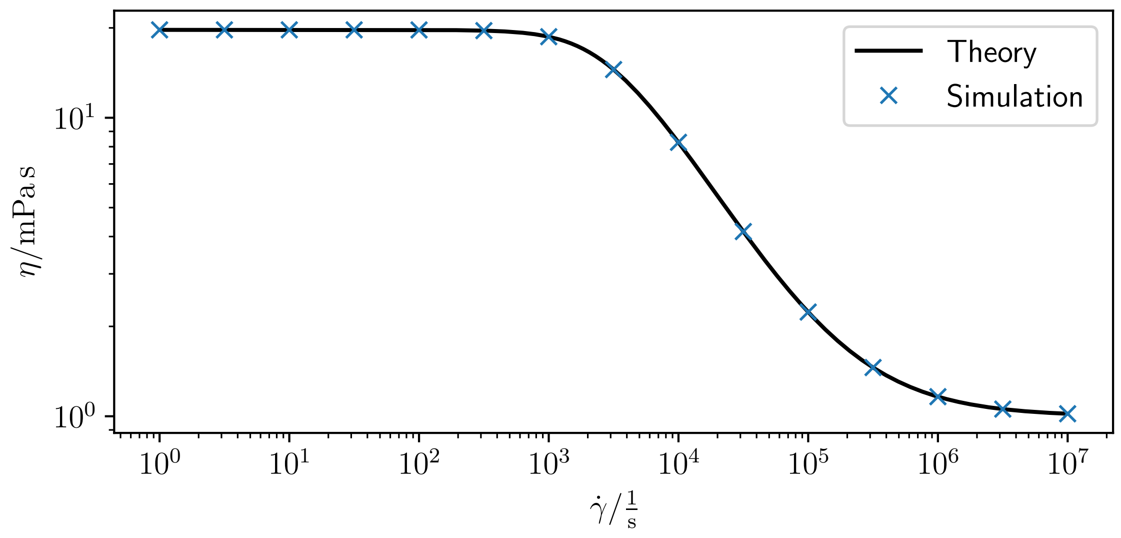


Figure 178: Plot of the viscosity of fluid 2 as determined from simulation and theory as a function of the shear-rate. New data, but otherwise similar to [1]

The agreement is excellent. The maximum L1 error for this curve is $|e_{\mathrm{L1}}| < 2.6 \times 10^{-11}$. The same process is repeated in simulation 47 for the fluid with the highest R_η in this thesis.

47 Alginate shuffling viscosity reproduction

Box:	$2 \times 43 \times 2$
	$L_0 \approx 0.49\,\mu\mathrm{m}$
Fluid:	PTT::alginate (fluid 13)
BC:	velocity (BC 2)

The result can be seen in figure 179.

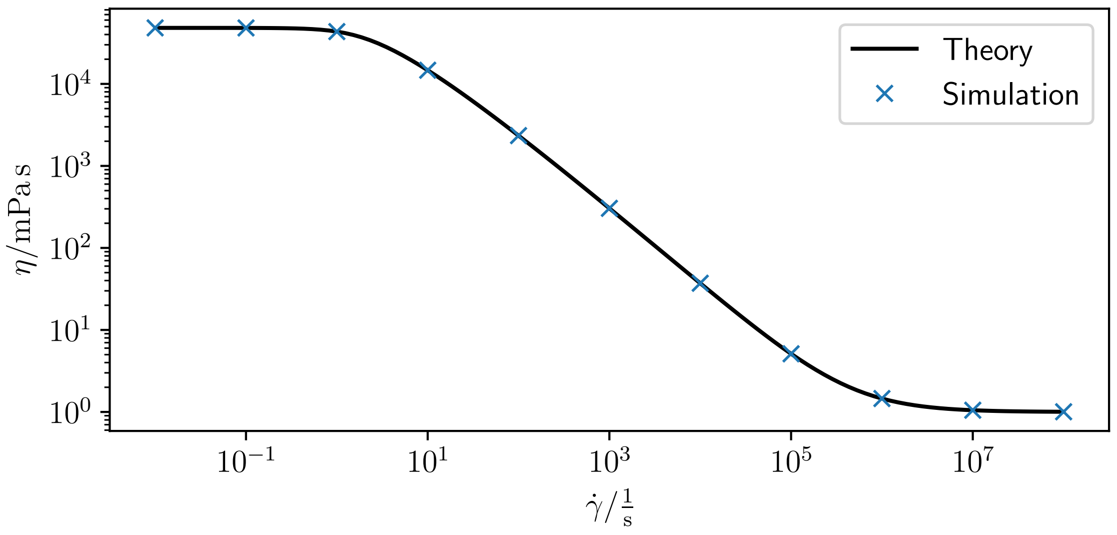


Figure 179: Plot of the viscosity of fluid 13 as determined from simulation and theory as a function of the shear-rate.

As before, the agreement is remarkable, with a maximum L1 error of $\max(|e_{\mathrm{L1}}|) \approx 1.2 \times 10^{-12}$. This sufficiently demonstrates the impressive accuracy of the shuffling algorithm in shear-flow. Next, it shall be examined in 2D Poiseuille flow.

U.2. Planar Poiseuille flow

To validate the accuracy of the viscosity shuffling algorithm in Poiseuille flow, the simulated velocity field is compared to a semi-analytical solution. Note, that with the solvent contributions this thesis considers, no analytical solutions can be found. The semi-analytical solutions are available in appendix F.2.3. As for the shear-flow, first the dependency on the shuffle viscosity is considered. Simulation 48 tests a few stable configurations.

48 Poiseuille shuffling fraction dependency

Box:	$2 \times 43 \times 2$
	$L_0 \approx 0.49\,\mu\mathrm{m}$
Fluid:	PTT::mc0_49 (fluid 2)
BC:	body-force (BC 13.3.2)

The resulting curves can be found in figure 180.

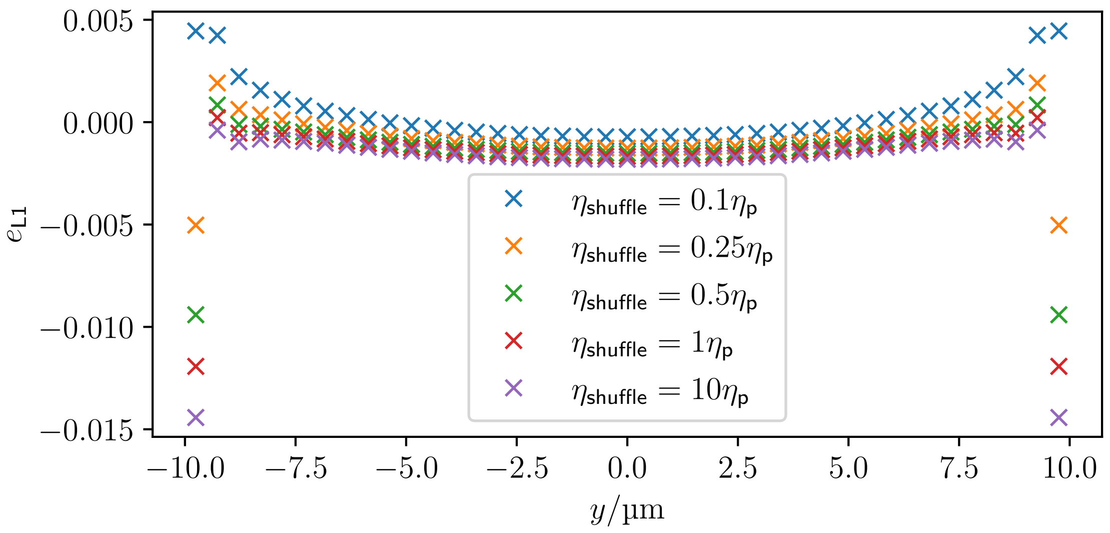


Figure 180: Plot of the L1 error of the velocity in a planar Poiseuille flow of methyl cellulose solution as a function of the y coordinate. The results for different shuffle viscosities are shown. New data, but otherwise similar to [1]

Notably, a dependence on the shuffle fraction is found. This primarily concerns the borders. Small shuffling viscosities overestimate the velocity near the borders. Large ones underestimate them. However, the actual difference (aside from the nodes next to the boundaries) is small. It is irrelevant, if the L1 error is positive or negative 1 ‰. Consequently, this dependence can be ignored. The default choice of $\eta_{\mathrm{shuffle}} = \eta_{\mathrm{p}}$ is not the most accurate here. This demonstrates, that one might be able to produce more accurate numbers using shuffling, if one is willing to tune it. However, a smaller shuffling viscosity risks the stability of the simulation. Next, an error depending on driving pressure gradient is found using simulation 49.

49 Methyl cellulose 2D shuffling velocity error study

Box:	$2 \times 43 \times 2$
	$L_0 \approx 0.49\,\mu\mathrm{m}$
Fluid:	PTT::mc0_49 (fluid 2)
BC:	body-force (BC 13.3.2)

The L2 error (see appendix S) can be seen in figure 181.

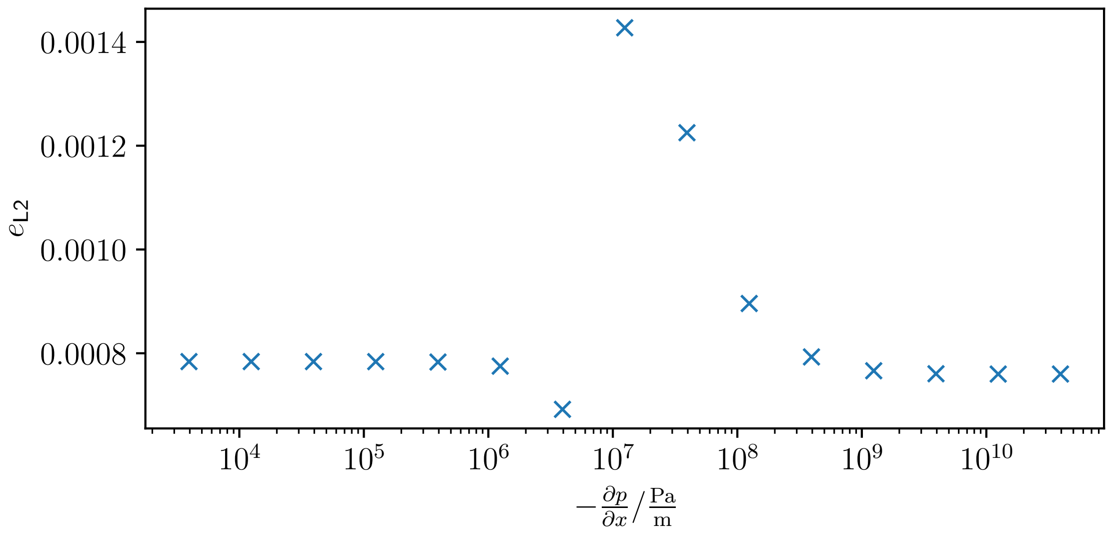


Figure 181: Plot of the L2 error of the velocity for a planar Poiseuille flow of methyl cellulose solution as a function of the pressure gradient. New data, but otherwise similar to [1]

The error profile clearly exhibits a peak. The average shear-rate across the channel, at the location of the peak is consistent with the maximum slope in the viscosity profile as seen in figure 178. For most data-points, distributed over many orders of magnitude, the error is completely independent of the pressure gradient. It only differs around the peak, where the simulation is most reactive to small changes in the pressure gradient. The worst velocity profile has an L2 error of $e_{\mathrm{L2}} \approx 1.4\,‰$. It is shown in figure 182.

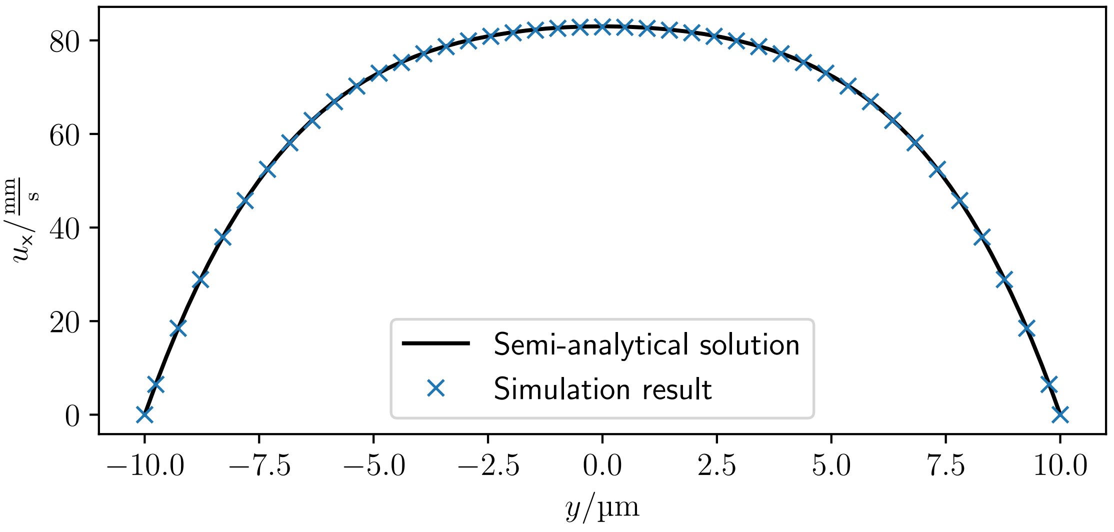


Figure 182: Plot of the velocity in a planar Poiseuille flow of methyl cellulose solution as a function of the y coordinate. Case with the worst agreement. New data, but otherwise similar to [1]

The agreement is clearly great even tho this is the worst case. The corresponding L1 error (see appendix S) can be seen in figure 183.

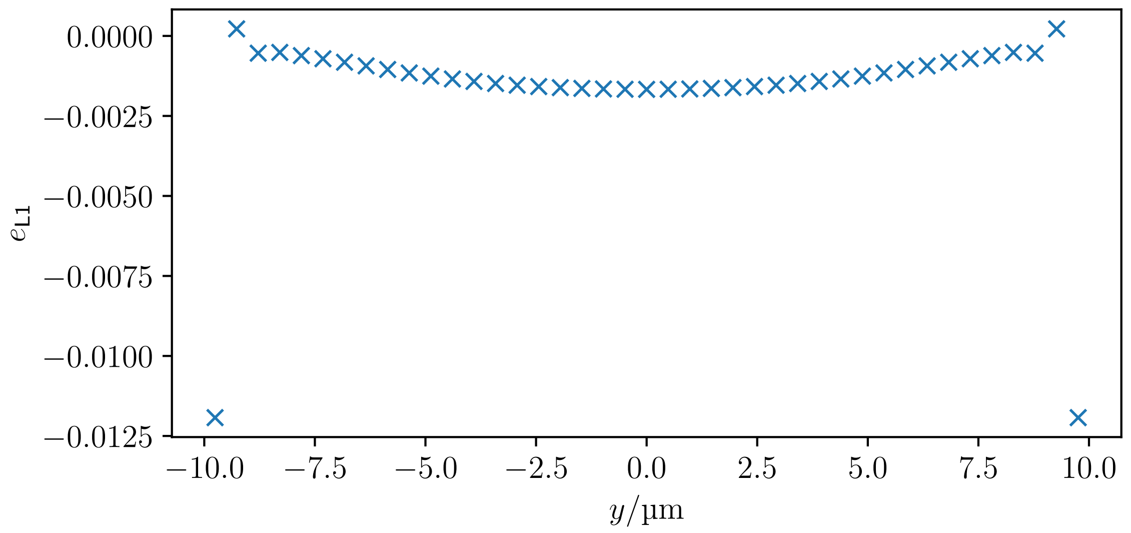


Figure 183: Plot of the L1 error of the velocity in a planar Poiseuille flow of methyl cellulose solution as a function of the y coordinate. Case with the worst agreement. New data, but otherwise similar to [1]

The error is concentrated at the boundaries. Away from the boundaries, the L1 error stays at around $-1\,‰$. This is by far not as accurate as the shear simulations, but still perfectly usable. A better result could be archived with higher resolution. The same geometry is tested with alginate in simulation 50.

50 Alginate 2D shuffling velocity error

Box:	$2 \times 43 \times 2$
	$L_0 \approx 0.49\,\mu m$
Fluid:	PTT::alginate (fluid 13)
BC:	body-force (BC 13.3.2)

The corresponding velocity profile can be found in figure 184.

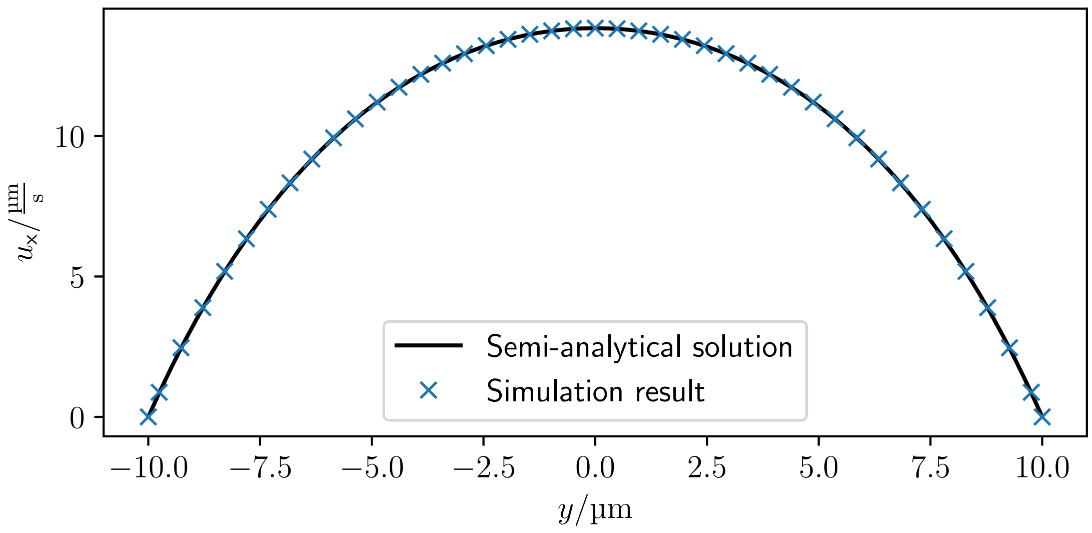


Figure 184: Plot of the velocity in a planar Poiseuille flow of alginate solution as a function of the y coordinate. New data, but otherwise similar to [1]

The agreement is great. The L2 error is $e_{L2} \approx 0.9\,‰$, which is similar to the values seen for the methyl cellulose solution. This means, that the error in this geometry is independent of the fluid parameters. The L1 error can be seen in figure 185.

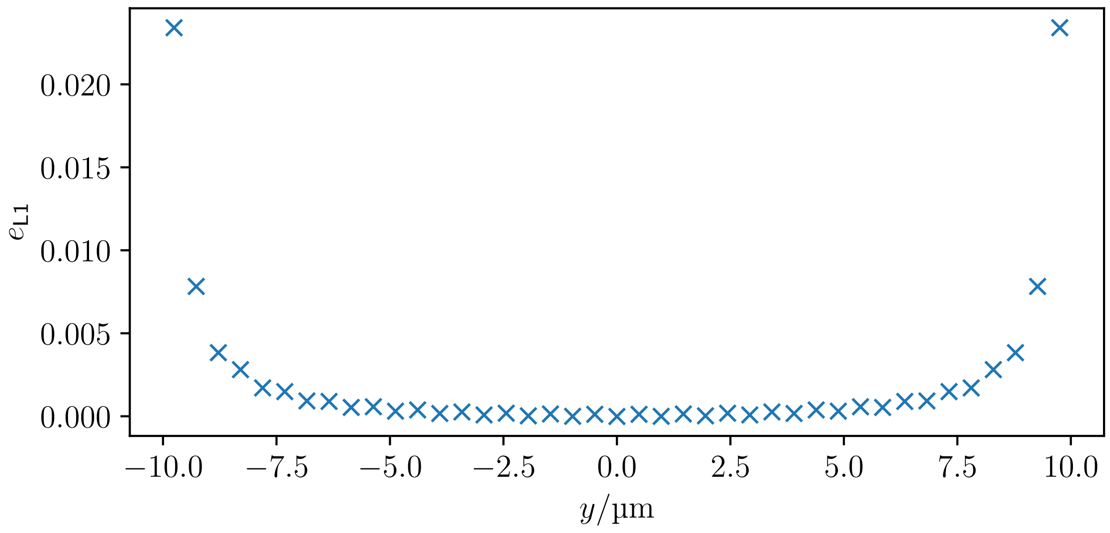


Figure 185: Plot of the L1 error of the velocity in a planar Poiseuille flow of alginate solution as a function of the y coordinate. New data, but otherwise similar to [1]

Similar to the methyl cellulose case, the error is concentrated at the boundaries, and around $1\,‰$ elsewhere. The time evolution of these profiles can be seen in the appendix of the publication [1]. It is not reproduced here, because the output resolution is not sufficient to produce clean curves. These curves demonstrate the oscillatory nature of the startup of viscoelastic fluids. In total, shuffling is valid for 2D Poiseuille flow. The error is numerically relevant, but imperceptible for humans. Next, 3D Poiseuille flow shall be considered.

U.3. Cylindrical Poiseuille flow

The same simulations are performed as for the planar case, but in a cylindrical channel. For the cylindrical geometry, discretization errors become relevant at the border. Mapping the rounded channel walls onto the rectilinear LBM grid causes the so-called "staircase effect" [47] (see appendix P.4). This makes this situation numerically slightly more complex. The same materials are used as above. Consequently, twice the pressure gradient is required to archive similar velocities. This relation appears in many analytical solutions for pressure driven flows, independent of the fluid (see appendix F). First, the parameter study for different driving pressure gradients is performed (simulation 51).

51 Methyl cellulose 3D shuffling velocity error study

Box:	$2 \times 43 \times 43$
	$L_0 \approx 0.49\,\mu\mathrm{m}$
Fluid:	PTT::mc0_49 (fluid 2)
BC:	body-force (BC 13.3.2)

The L2 error (see appendix S) can be seen in figure 186.

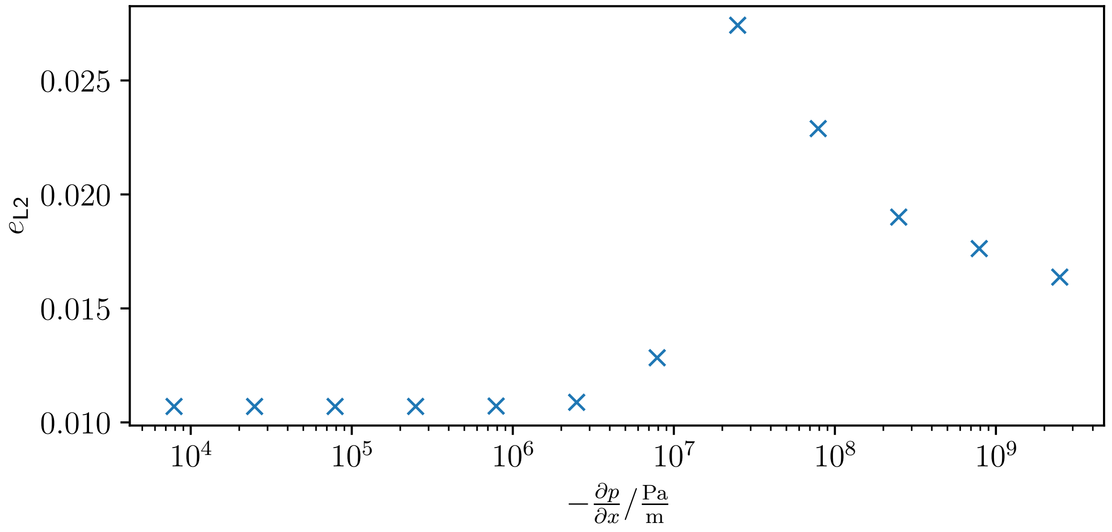


Figure 186: Plot of the L2 error of the velocity for a cylindrical Poiseuille flow of methyl cellulose solution as a function of the pressure gradient. New data, but otherwise similar to [1]

The error profile clearly exhibits a peak. The average shear-rate across the channel, at the location of the peak is consistent with the maximum slope in the viscosity profile as seen in figure 178. For most data-points, distributed over many orders of magnitude, the error is completely independent of the pressure gradient. It only differs around the peak, where the simulation is most reactive to small changes in the pressure gradient. The worst velocity profile has an L2 error of $e_{\mathrm{L2}} \approx 2.7\,\%$. It is shown in figure 187.

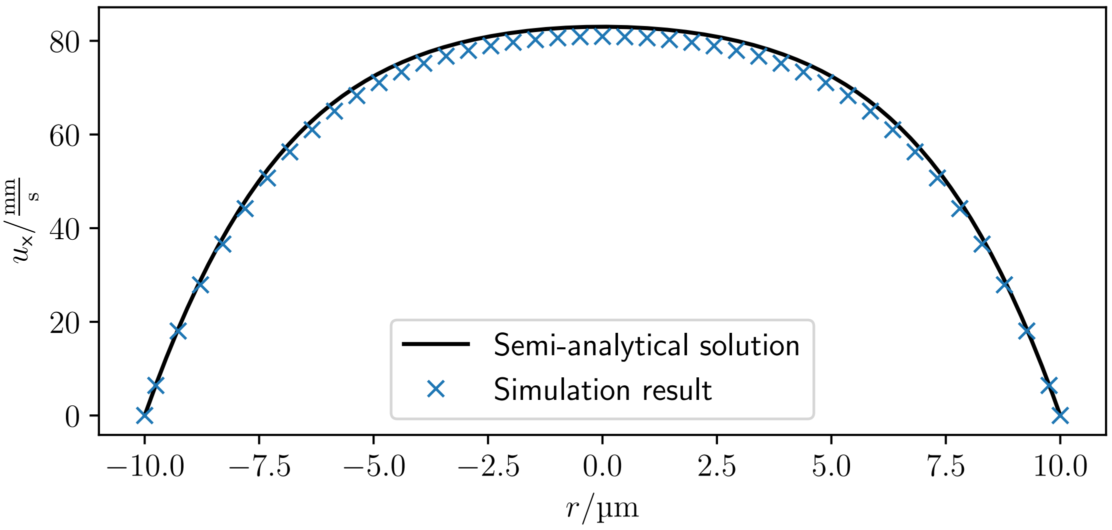


Figure 187: Plot of the velocity in a cylindrical Poiseuille flow of methyl cellulose solution as a function of the y coordinate. Case with the worst agreement. New data, but otherwise similar to [1]

The agreement is still tolerable, but noticeable worse than the 2D case. The corresponding L1 error (see appendix S) can be seen in figure 188.

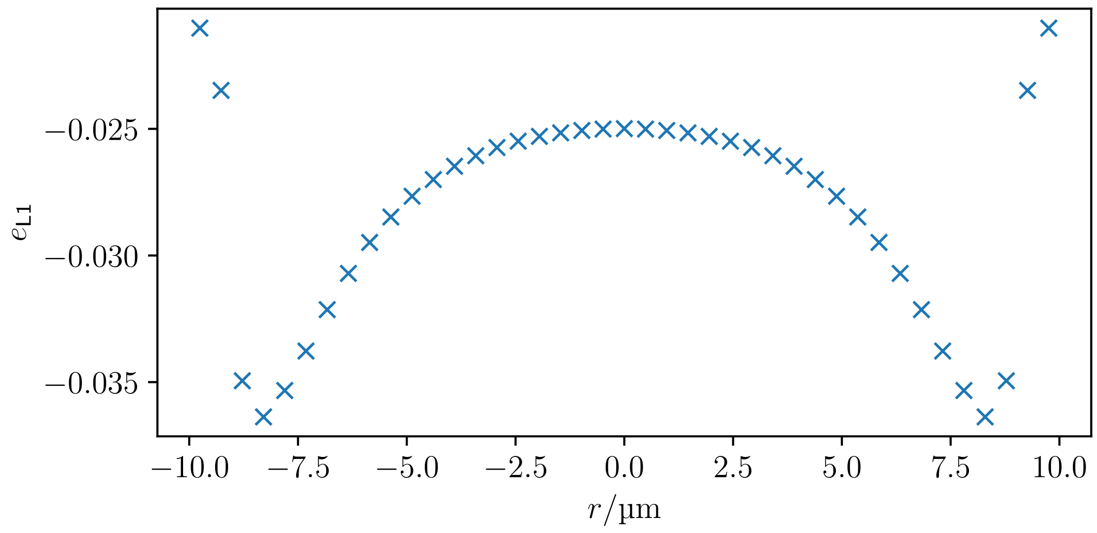


Figure 188: Plot of the L1 error of the velocity in a cylindrical Poiseuille flow of methyl cellulose solution as a function of the y coordinate. Case with the worst agreement. New data, but otherwise similar to [1]

The error is relevant throughout the channel. The L1 error takes values of around 3 %. This is an order of magnitude worse than the 2D case. Given, that this is the worst case, one can accept it. A better result could be archived with higher resolution. Given, that the sign of the error has changed compared to the paper, it is possible, that this simulation is still slowly oscillating at the end. This is a phenomenon viscoelastic fluids commonly exhibit. The time until the oscillation has fully sized can be extremely long. Consequently, many simulations are not in full equilibrium, because this would be prohibitively expensive. The same geometry is tested with alginate in simulation 52.

52 Alginate 3D shuffling velocity error

Box:	$2 \times 43 \times 43$
	$L_0 \approx 0.49$ µm
Fluid:	PTT::alginate (fluid 13)
BC:	body-force (BC 13.3.2)

The corresponding velocity profile can be found in figure 189.

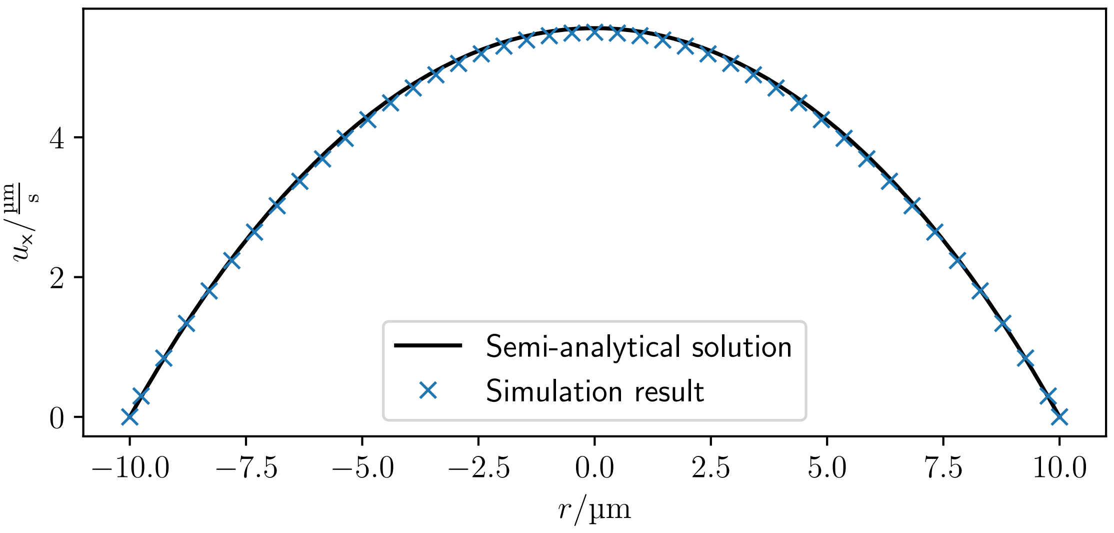


Figure 189: Plot of the velocity in a cylindrical Poiseuille flow of alginate solution as a function of the y coordinate. New data, but otherwise similar to [1]

The agreement is good. The L2 error is $e_{L2} \approx 1.2\,\%$, which is a bit better than the values seen for the methyl cellulose solution. This means, that the error in this geometry has at most a minor dependence on the fluid parameters. The L1 error can be seen in figure 190.

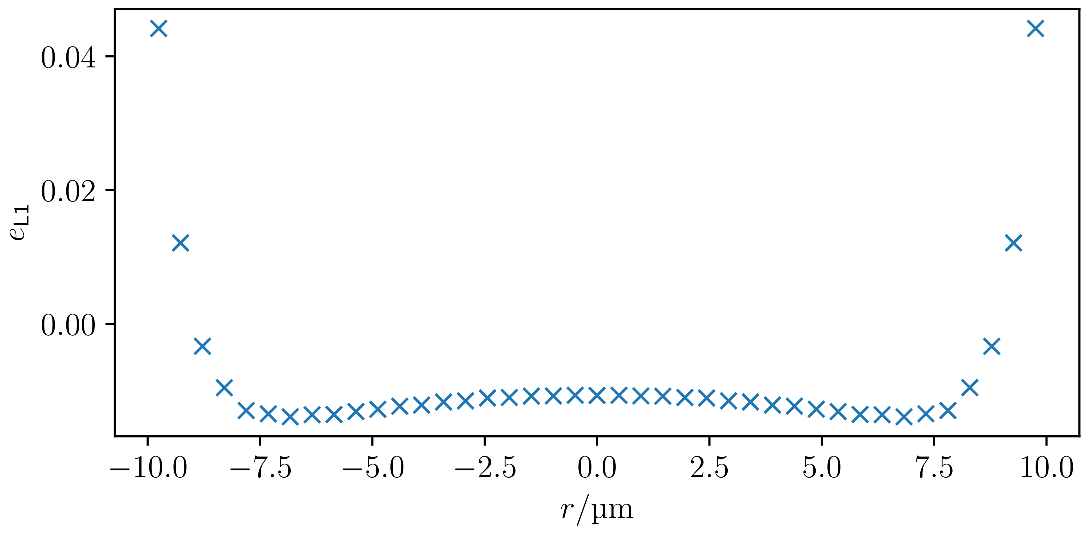


Figure 190: Plot of the L1 error of the velocity in a cylindrical Poiseuille flow of alginate solution as a function of the y coordinate. New data, but otherwise similar to [1]

Similar to the 2D case, the error is concentrated at the boundaries. The error stays around $-1\,\%$ elsewhere, which is about an order of magnitude worse than the 2D case. The time evolution of these profiles can be seen in the appendix of the publication [1]. It is not reproduced here, because the output resolution is not sufficient to produce clean curves. These curves demonstrate the oscillatory nature of the startup of viscoelastic fluids. In total, shuffling is valid for 3D Poiseuille flow. The error is at times larger than desired. It is likely, that this is in part due to the oscillatory nature of viscoelastic fluids. The other major contribution is the staircase effect (see appendix P.4), which causes the 3D values to be markedly worse than the 2D ones. In total, the viscosity shuffling algorithm clearly produces workable results.

V. Example applications

Three typical experimentally relevant situations are presented to illustrate the use of the viscosity shuffling algorithm. These are a printing nozzle, an RT-DC channel (see appendix P.3) and a cell in shear-flow. First, the nozzle is covered.

V.1. Nozzle

This application stems from bioprinting where the bio-ink is pushed through a conical nozzle [3–5]. For this simulation 53 is prepared.

53 Shuffle example nozzle

Box:	$947 \times 191 \times 191$
	$L_0 \approx 19\,\mu\text{m}$
Dimensions:	$R_{\text{inlet}} = 1.8\,\text{mm}$
	$R_{\text{outlet}} = 200\,\mu\text{m}$
	$L = 18\,\text{mm}$
Fluid:	PTT::mc0_49 (fluid 2)
BC:	mass-flow (BC 3)
	$Q \approx 5\,\frac{\mu\text{L}}{\text{s}}$

Due to the difference in the radii of the inlet and outlet, the system cannot be modeled with periodic boundary conditions. An iterative approach is chosen (see section 13.3.7), where first the velocity profiles for the inlet and outlet are determined using simulations with pressure gradients. These are loaded into the final simulation and scaled by the fraction of their mass-flow Q. The mass conservation issue (see section 13.3.3) demands binary identical flows on inlet and outlet. The scaled velocities might still be off by a few ϵ_{fp64} (see appendix O), but this approximation is stable long enough. This scaling causes the flow-rate Q to be a little different from the initially planned value. Here, it works out to $Q \approx 5.09\,\frac{\mu\text{L}}{\text{s}}$ after the scaling. The resulting flow field can be seen in figure 191.

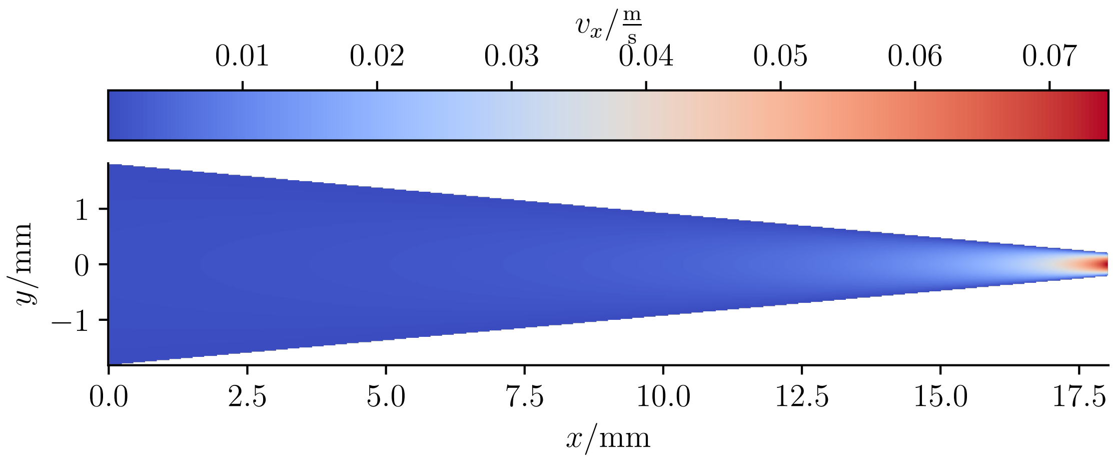


Figure 191: Cross-section of the velocity field for a methyl cellulose solution in a conical nozzle as used for bioprinting. New data, but otherwise similar to [1]

The largest stresses are present close to the wall and at the nozzle exit. This can be seen in figure 192.

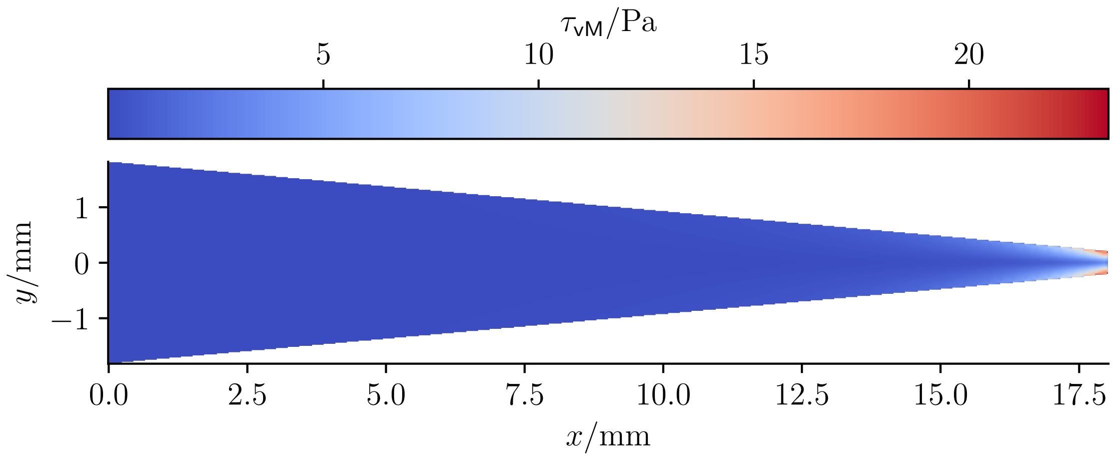


Figure 192: Cross-section of the von Mises stress (see section 4.2) for a methyl cellulose solution in a conical nozzle as used for bioprinting. New data, but otherwise similar to [1]

This means that, within the nozzle, cells traveling in the vicinity of the center-line are subject to less stress, than cell near the walls. This is the case even for fast flows. Earlier results in Newtonian fluids [67] did show the same behavior. The viscous stress component of this flow can be seen in figure 193.

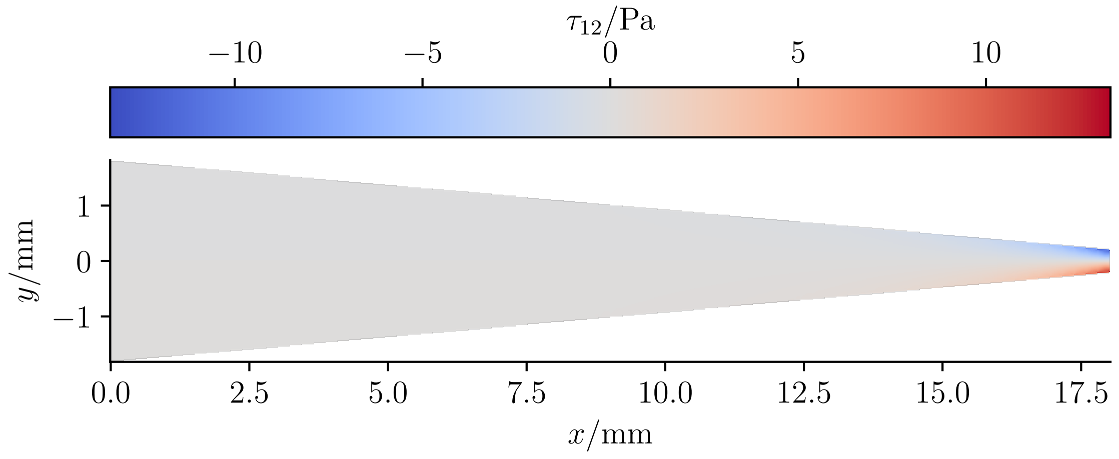


Figure 193: Cross-section of the viscous stress for a methyl cellulose solution in a conical nozzle as used for bioprinting. New data, but otherwise similar to [1]

Dividing the viscous stress by twice the corresponding component of the strain-rate tensor (see section 3) yields a viscosity. The solvent viscosity is added to retrieve the total viscosity. This can be seen in figure 193.

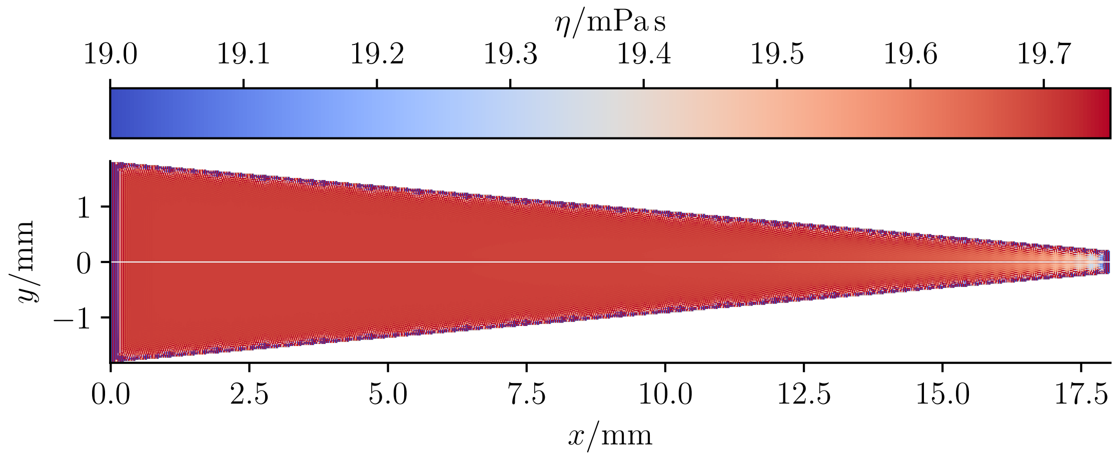


Figure 194: Cross-section of the calculated "viscosity" in a Nozzle. New data, but otherwise similar to [1]

This diverges along the center-line, and thus the center-line is excluded. There is significant stress buildup along the walls. The plot has been limited to make the shear-thinning visible. The values range from approximately 11 mPa s to approximately 42 mPa s Shear-thinning is most noticeable towards the center of the channel, despite the shear rate being highest at the edges. This is an interesting result. The bad interaction with the walls is an issue of the finite volume algorithm (see section 4) and not a problem specific to shuffling. This concludes the nozzle example. Next, the RT-DC channel is considered.

V.2. RT-DC

The RT-DC geometry (see appendix P.3) has a narrow observation channel with a square cross-section [6, 112]. The inflow (not modeled here) opens up to additional filtering structures and ultimately a large reservior, which provided the cells, that are squeezed through the observation restriction. The observation channel has a size of 40 µm × 40 µm. This is done in simulation 54.

54 Shuffle example RT-DC

Box:	$405 \times 92 \times 32$
	$L_0 = 1.\bar{3}\,\mu\text{m}$
Fluid:	PTT::mc0_49 (fluid 2)
BC:	body-force (BC 13.3.2)
	$-\frac{\partial p}{\partial x} = 1 \times 10^7\,\frac{\text{Pa}}{\text{m}}$

The resulting flow field can be seen in figure 195.

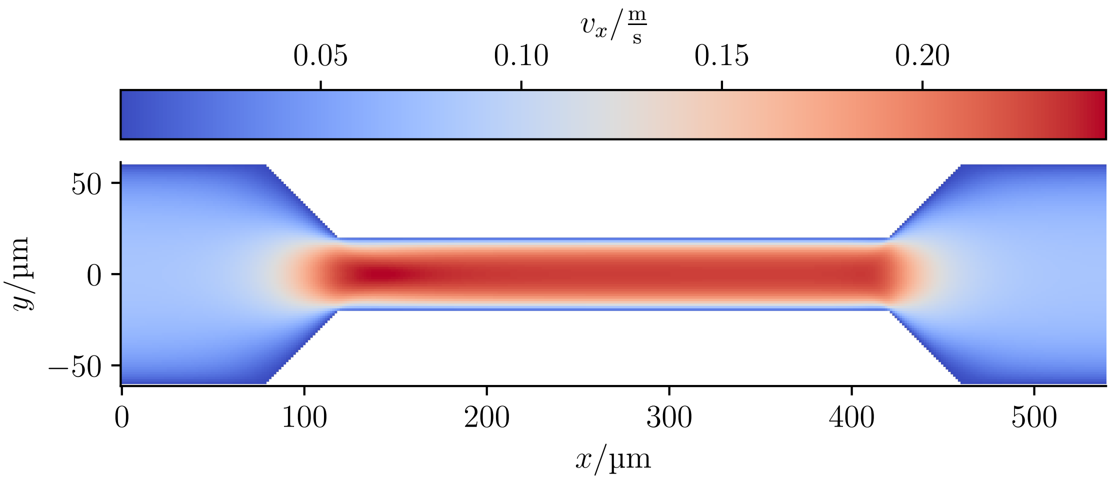


Figure 195: Cross-section of the velocity field for a methyl cellulose solution in a typical RT-DC geometry. New data, but otherwise similar to [1]

There is a slight velocity maximum shortly behind the start of the smaller inner channel. It is barely visible, but represents a relevant difference to simple Newtonian flows. The highest stresses arise in direct vicinity to the wall. This can be seen in figure 196.

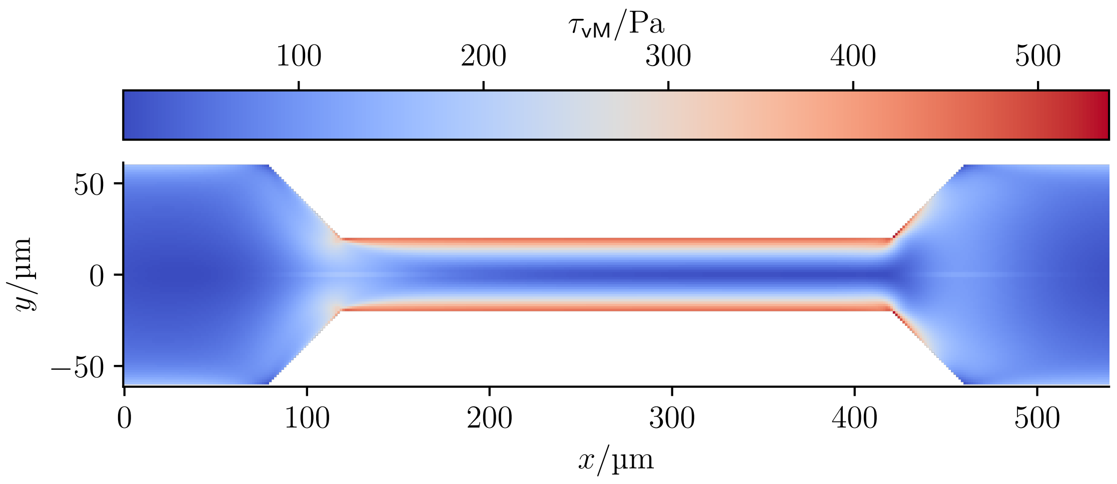


Figure 196: Cross-section of the von Mises stress (see section 4.2) for a methyl cellulose solution in a typical RT-DC geometry. New data, but otherwise similar to [1]

Interestingly, the high stress region extends further on the outflow side. This is a pattern, that is also present in the viscous stress component seen in figure 197.

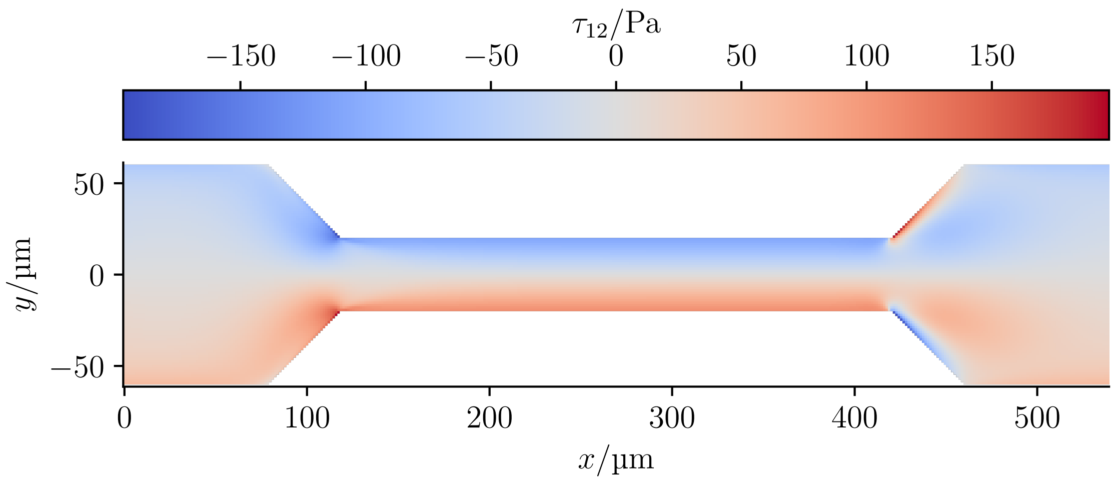


Figure 197: Cross-section of the viscous stress for a methyl cellulose solution in a typical RT-DC geometry. New data, but otherwise similar to [1]

Compared to a fluid exhibiting purely viscous behavior, there are three qualitative differences. There is a region of low stress, directly after the inlet. The polymer conformation tensor reacts with a delay. This means, that shortly after entering the restriction, the strain-rate tensor changes rapidly, but the polymers have not reacted yet. Depending on the speed of the advection, this region varies in size. The same mechanics apply at the exit of the channel. The restriction is gone, but the large deformation and therefore the large stress persists. This heightened stress decays according to the relaxation time. From the definition of the viscous stress one can define the viscosity as the ratio of the stress and twice the strain-rate tensor. Performing this division here would generally speaking yield a region of very low and a region of very high viscosity for the inlet and outlet receptively. These “viscosities” can have values, which cannot be retrieved in steady-state. Meaning they are outside the curve generated from rheology. Along the diagonal walls after the outlet of the restriction, this effect is particularly strong. If one performs the division described above here, one retrieves negative “viscosities”. The simulation volume contains several points at which, the strain-rate vanishes. Therefore, the “viscositie” diverges here. To display it, the results have to be clamped. They can be seen in figure 198.

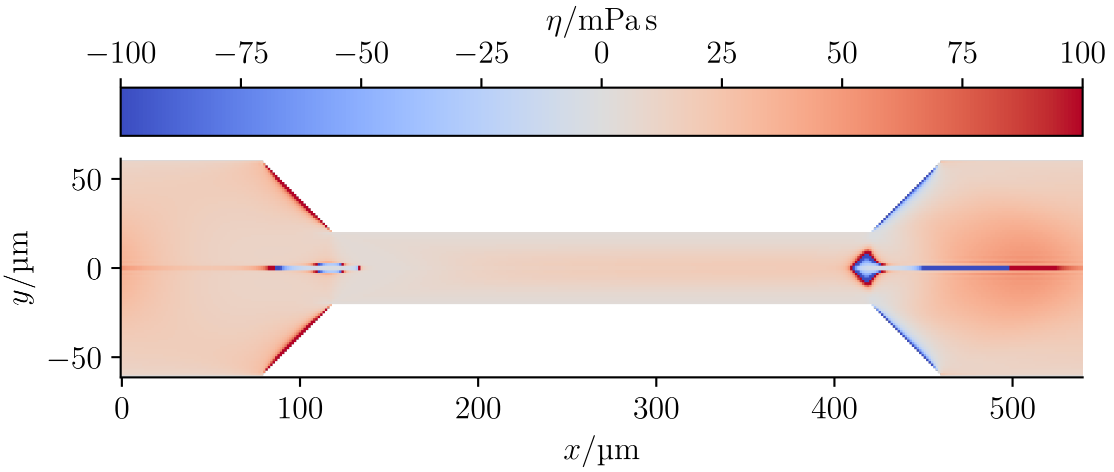


Figure 198: Cross-section of the calculated “Viscosity” in an RT-DC channel. There are some very large and unphysical values present, which have to be clamped. New data, but otherwise similar to [1]

Interpreting this fraction as the viscosity is not possible for a viscoelastic fluid in complex geometry. This is signified by the non-physical values, particularly the negative ones near the walls. This odd behavior also signifies the regions of interest. The importance of viscoelastic simulations is also highlighted here. The regions of interest, which exhibit this odd behavior cannot be reproduced by purely viscous models. Here viscoelastic simulations are required despite the high computational cost. Given the placement of these regions, RT-DC always requires viscoelastic simulations. With this, the RT-DC example concludes. Lastly, the cell in shear is covered.

V.3. Cell in shear-flow

The broad applicability of the viscosity shuffling approach is demonstrated by the third example. Here, the behavior in presence of other LBM extensions such as the immersed-boundary method is explored. A homogeneously elastic particle is added. This is a simplified, but reasonably accurate model for the mechanics of a living, biological cell [81]. For simplicity, the viscoelastic fluid inside and outside the cell is considered to be identical. Simulation 55 converges to a tank-treading ellipsoid.

55 Shuffle example cell in shear

Box:	$100 \times 102 \times 100$
	$L_0 = 1\,\mu\text{m}$
Fluid:	PTT::mc0_49 (fluid 2)
BC:	velocity (BC 2)
	$\dot{\gamma} = 400\,\frac{1}{\text{s}}$
Cell:	Young's modulus $E = 100\,\text{Pa}$
	Radius, Resolution $R = 6\,\mu\text{m}, 6$
	Capillary number $Ca_\text{K} \approx 1.1$

The simulation volume has to be significantly larger than the cell to properly reproduce the pure shear-flow boundary conditions. Otherwise, the walls influence the cell. Here a resolution of 100 lattice points is chosen. The deformed cell can be seen in figure 199.

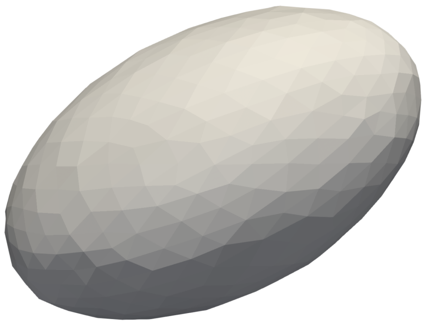


Figure 199: Render of the shape of the deformed cell in viscoelastic shear-flow. New data, but otherwise similar to [1]

This is in qualitative agreement with earlier simulations [70]. It can be evaluated using Roscoe theory (see section 10). This returns the values presented in table 11.

Data source	α_1	α_2	α_3	θ	$\nu_\text{tt}/\frac{1}{\text{s}}$
Simulation	1.300 00	0.7770	1.0107	0.4794	−176.918
Theory	1.295 11	0.7671	1.0066	0.5347	−175.389

Table 11: Roscoe parameters determined from the simulation and corresponding theory. Values rounded in accordance to errors. Errors suppressed for brevity.

While these values do not quite match within the respective errors they are rather close. They are only separated by a small single digit percentage. A notable exception is the alignment angle θ, which is off by more than 10 %. Roscoe theory is for Newtonian fluids. For viscoelastic fluids, a reduction in this angle is expected (see section 12.2). The von Mises stress is shown in figure 200.

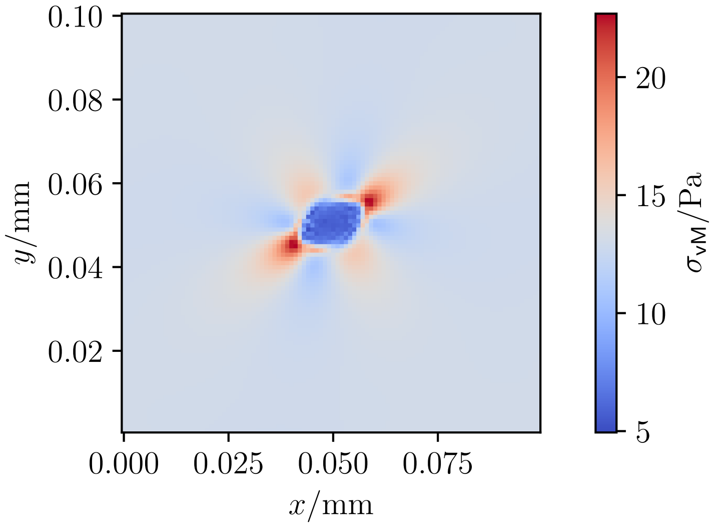


Figure 200: Cross-section of the von Mises stress of the fluid surrounding a cell in viscoelastic shear-flow. New data, but otherwise similar to [1]

The influence of the cell is present even several cell sizes from its location. Along each of its semi-axis a region of high stress clearly emerges from the cell. Along the largest axis, these regions exhibit the largest stress. With this, the example of previously impossible simulations are concluded. All the topics shown here will be covered more in depth later in this thesis.

W. Exclusion of small spherical indenters in AFM

As explained in section 20.1.3, the force should follow a power law in respect to R. Due to the lack of sufficient resolution for small indenters as discussed in section 20.2.3, this is not always the case for spherical indenters. This can be clearly seen in figures 201 – 204. Values not conforming to the power-law trend according to visual inspection of the following figures are discarded in the subsequent analysis. This means, if a curve is constant or a curve for a lower R approaches by a noticeable margin or even crosses a curve for a higher R, the offending values are discarded. Each figure for each deformation δ is acompanied by a table beneath listing the discarded values for this deformation. The discarded range is visualized in figure 67.

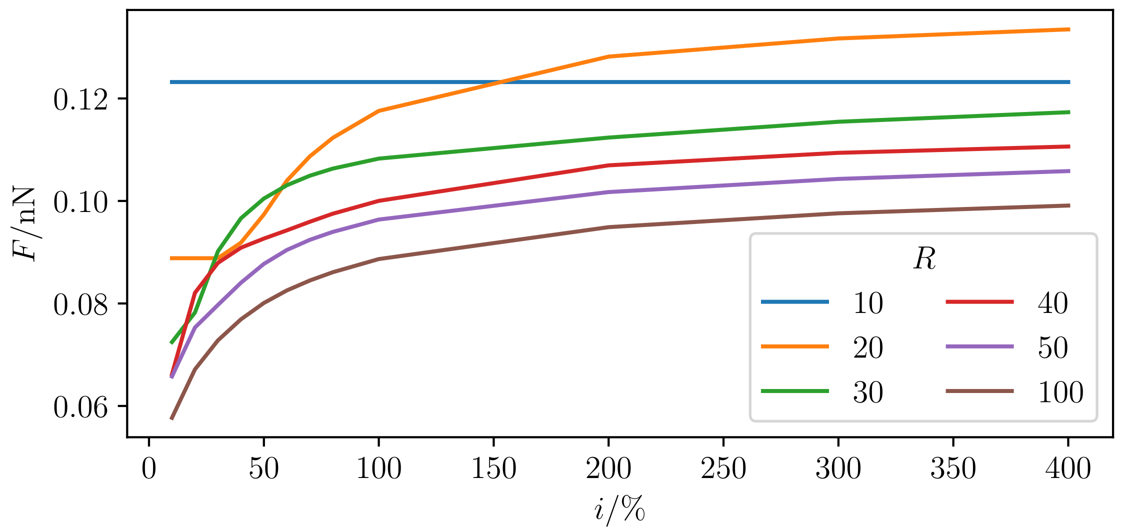


Figure 201: The force measured in the simulation depending on resolution R and indenter i for deformation $\delta = 0.5\,\%$.

R	discarded $i/\%$
10	10, 20, 30, 40, 50, 60, 70, 80, 100, 200, 300, 400
20	10, 20, 30, 40, 50, 60, 70, 80, 100
30	10, 20, 30, 40
40	10

Table 12: Discarded data-points for $\delta = 0.5\,\%$.

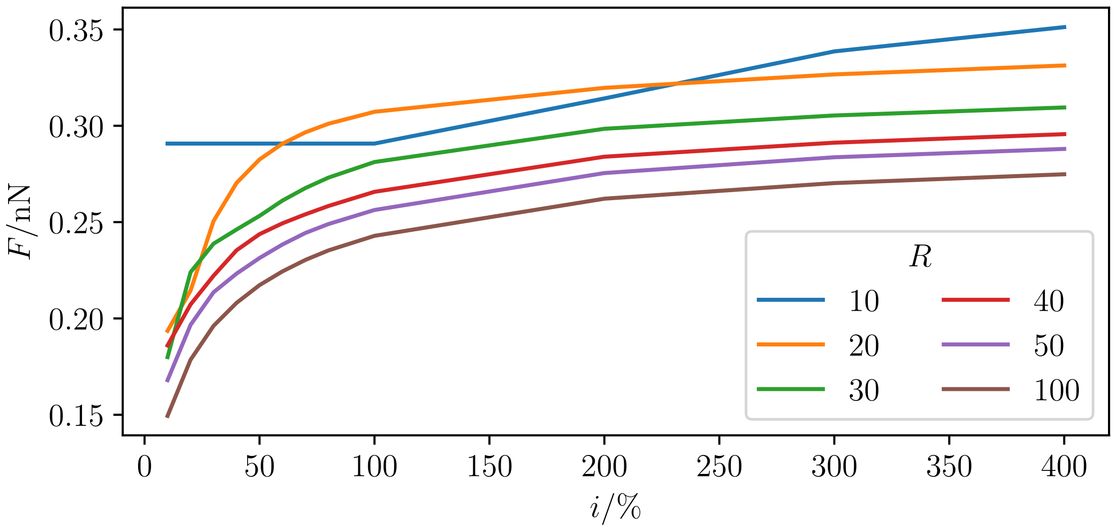


Figure 202: The force measured in the simulation depending on resolution R and indenter i for deformation $\delta = 1\,\%$.

R	discarded i/%
10	10, 20, 30, 40, 50, 60, 70, 80, 100, 200, 300, 400
20	10, 20, 30
30	10

Table 13: Discarded data-points for $\delta = 1\,\%$.

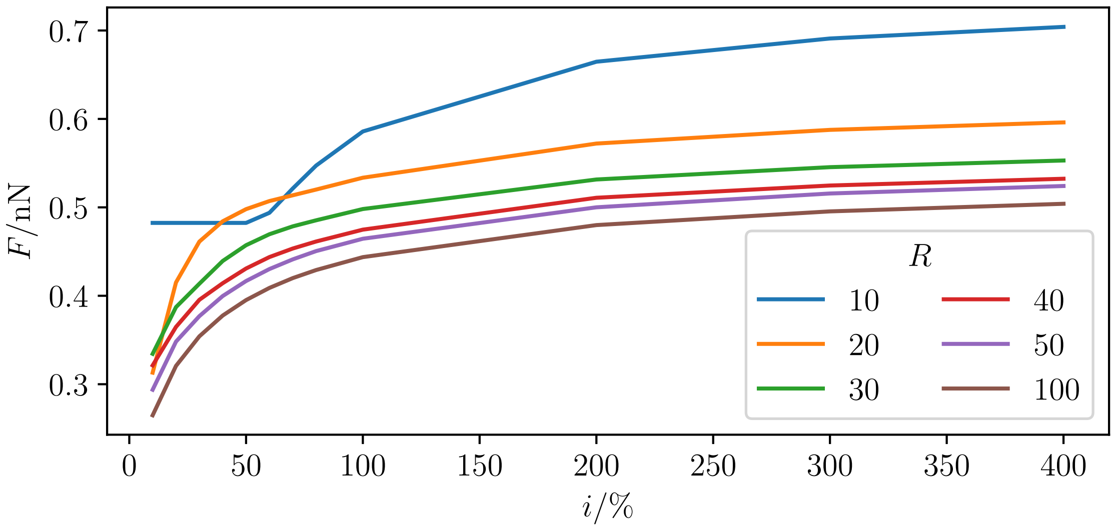


Figure 203: The force measured in the simulation depending on resolution R and indenter i for deformation $\delta = 1.5\,\%$.

R	discarded $i/\%$
10	10, 20, 30, 40, 50, 60, 70, 80, 100
20	10, 20
30	10

Table 14: Discarded data-points for $\delta = 1.5\,\%$.

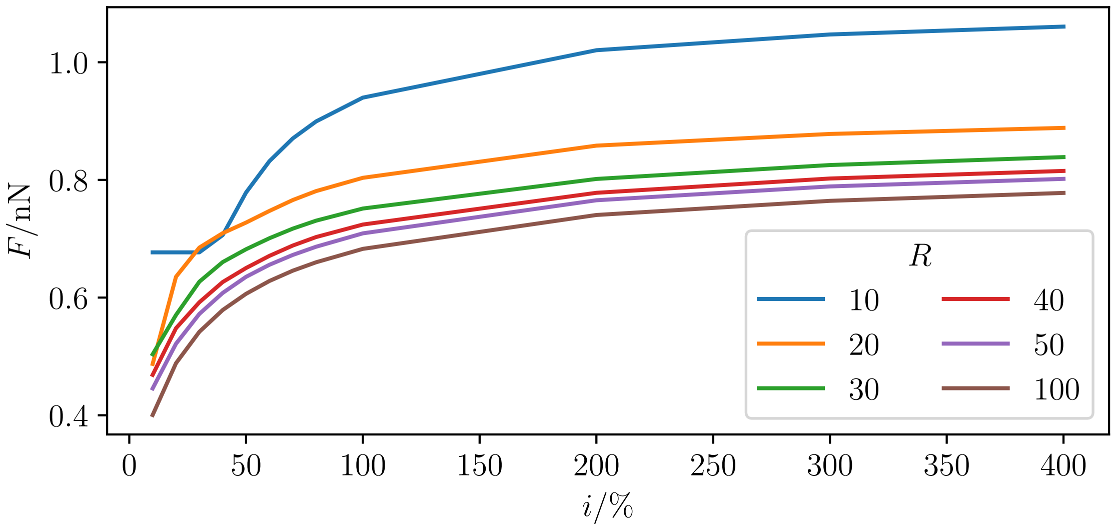


Figure 204: The force measured in the simulation depending on resolution R and indenter i for deformation $\delta = 2\,\%$.

R	discarded $i/\%$
10	10, 20, 30, 40, 50, 60, 70, 80, 100
20	10

Table 15: Discarded data-points for $\delta = 2\,\%$.

X. Accuracy of Roscoe simulations

The Roscoe simulations in this thesis did at times show some unexplained discrepancies. Here some possible reasons are evaluated. This also allows gauging how large the error in such simulation typically is. For this, the Capillary number, the resolution of the cell and the size of the mesh are considered.

X.1. Depending on the Capillary number

The Newtonian simulation 56 is performed with varying Capillary numbers.

56 Roscoe for varying Ca_{K} in water

Box:	$100 \times 500 \times 100$
	$L_0 = 625\,\mathrm{nm}$
Fluid:	Newtonian::water (fluid 1)
BC:	velocity (BC 2)
	$\dot{\gamma} = 1 \times 10^4\,\frac{1}{\mathrm{s}}$
Cell:	Young's modulus variable
	Radius, Resolution $R \approx 7.5\,\mathrm{\mu m}, 12$
	Capillary number variable

The Roscoe deformation parameters α_1 and α_2 show small errors for small Capillary numbers. For Capillary numbers above unity, the error quickly grows to noticeable size. This can be seen in figures 205 and 206.

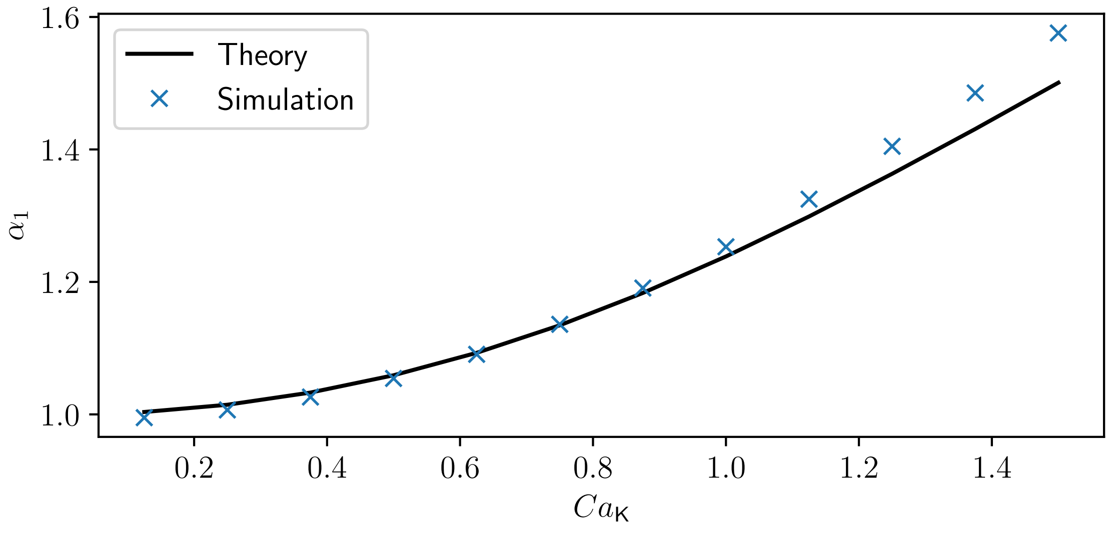


Figure 205: Plot of the Roscoe deformation parameter α_1 as a function of the Capillary number Ca_{K}. Curves for Roscoe theory and a Newtonian simulation are shown.

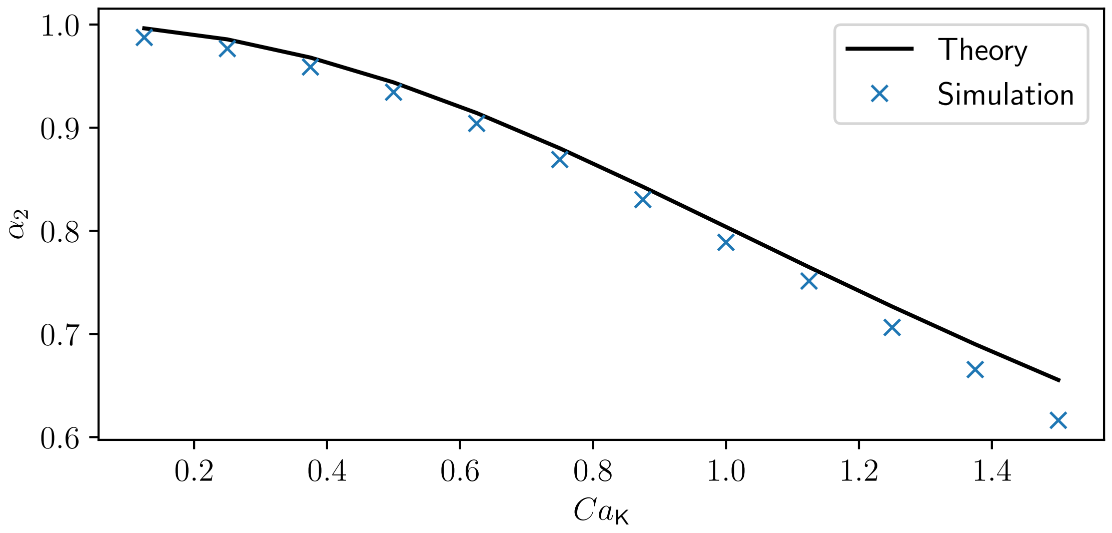


Figure 206: Plot of the Roscoe deformation parameter α_1 as a function of the Capillary number Ca_K. Curves for Roscoe theory and a Newtonian simulation are shown.

The fact, that these cells a more deformed than expected is interesting. A decreed deformation could be explained through discretization on the surface. The Roscoe deformation parameter α_3 shows errors for most capillary number as can be seen in figure 207.

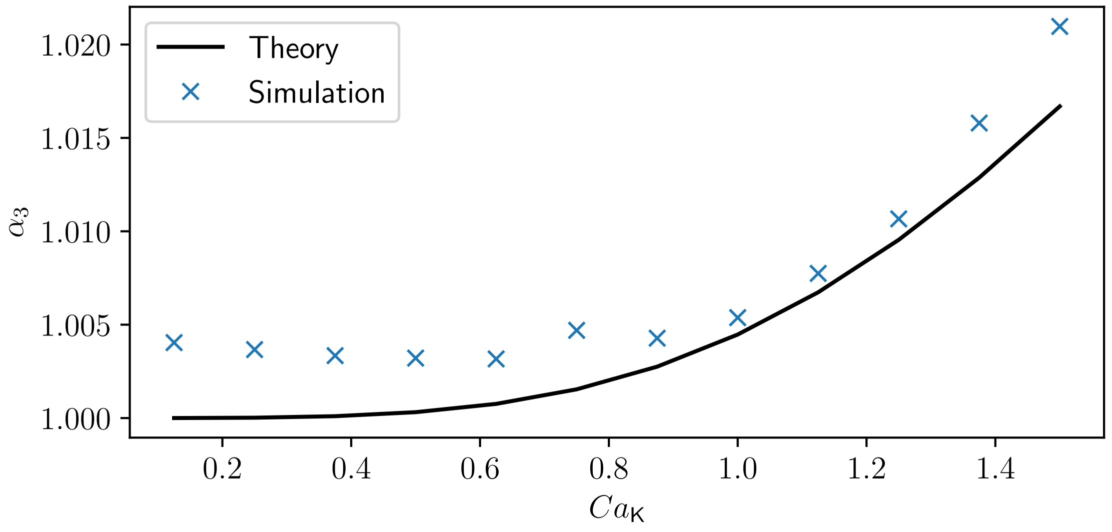


Figure 207: Plot of the Roscoe deformation parameter α_3 as a function of the Capillary number Ca_K. Curves for Roscoe theory and a Newtonian simulation are shown.

However, given, that $\alpha_3 \approx 1$ holds with or without the error, this is not relevant. A similar argument can be made for the volume seen in figure 208.

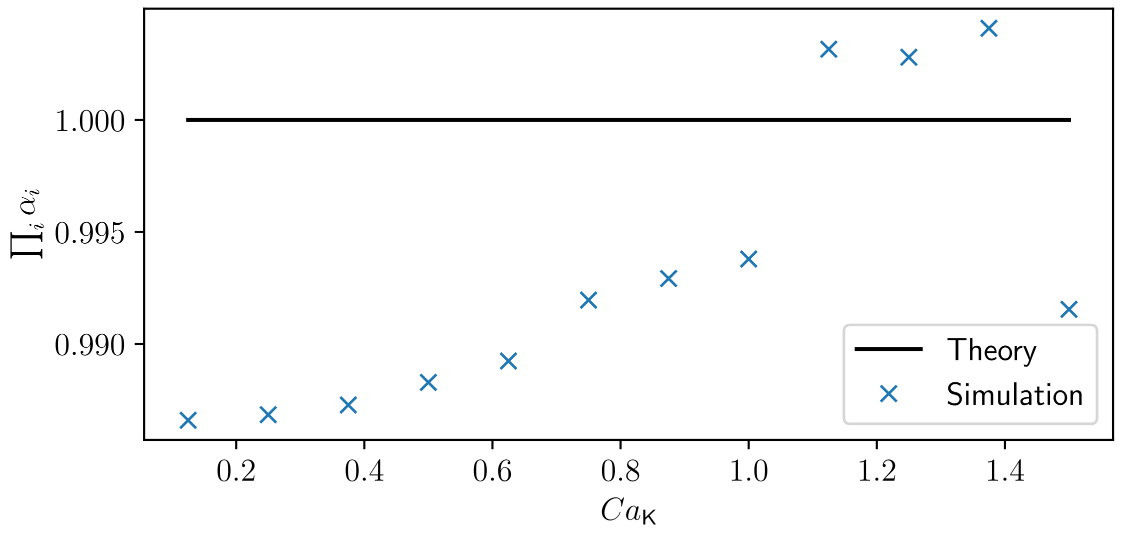


Figure 208: Plot of the cell volume as calculated from the Roscoe deformation parameters as a function of the Capillary number Ca_K. Curves for Roscoe theory and a Newtonian simulation are shown.

The alignment angle also shows noticeable errors as can be seen in figure 209.

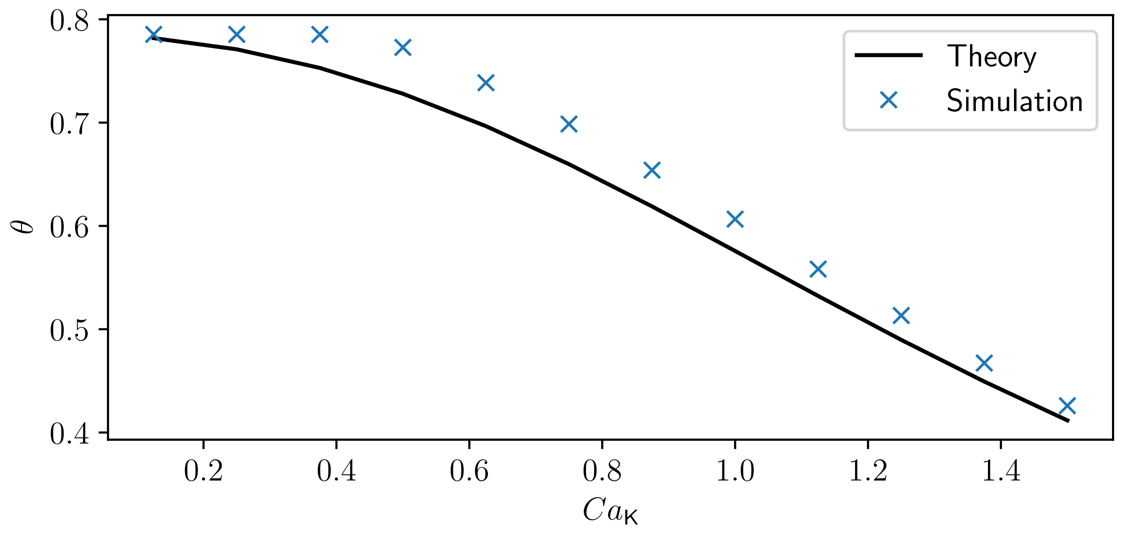


Figure 209: Plot of the Roscoe alignment parameter θ as a function of the Capillary number Ca_K. Curves for Roscoe theory and a Newtonian simulation are shown.

No clear trend can be seen in respect to the capillary number. The biggest discrepancy can be seen for the tank-treading frequency. This can be seen in figure 210.

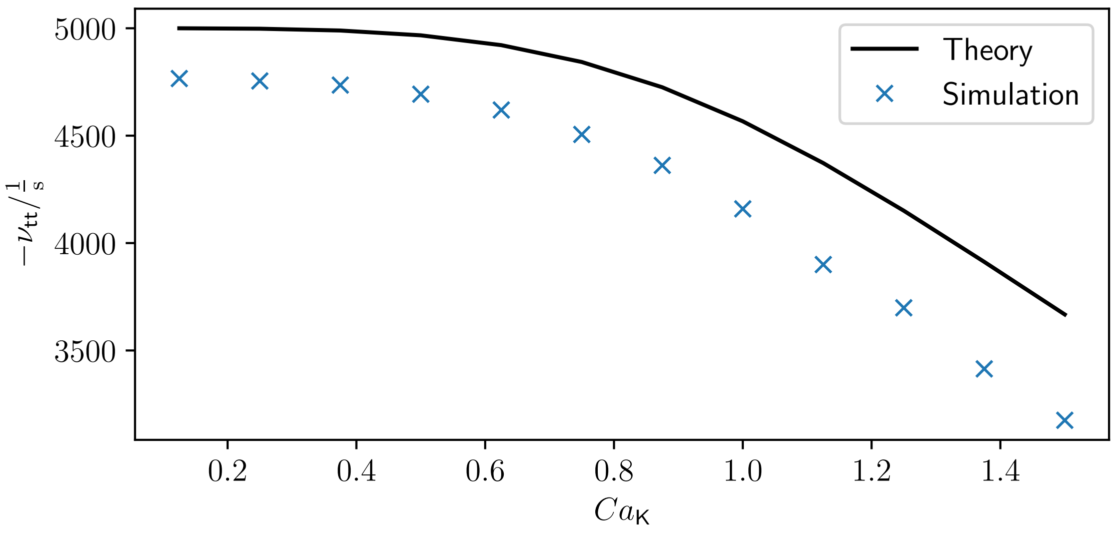


Figure 210: Plot of the tank-treading frequency ν_{tt} as a function of the Capillary number Ca_K. Curves for Roscoe theory and a Newtonian simulation are shown.

The discrepancy is significant and grows slightly with the Capillary number. In general, the dependence on the Capillary number is not very strong if present at all. This is a strong indication, that it is not the root cause of the observed inaccuracies. Next, the cell resolution as another typical error source is investigated.

X.2. Influence of the cell resolution

The Newtonian simulation 57 is performed with varying cell resolutions.

57 Roscoe for varying resolutions in water

Box:	variable
Fluid:	Newtonian::water (fluid 1)
BC:	velocity (BC 2)
	$\dot{\gamma} = 5 \times 10^3 \, \frac{1}{s}$
Cell:	Young's modulus $E = 20\,\mathrm{Pa}$
	Radius, Resolution $R \approx 7.5\,\mu\mathrm{m}$, variable
	Capillary number $Ca_K = 2$

The Capillary number being held constant means, that the expected deformation stays constant for the varying resolution. This is actually somewhat accurate for the Roscoe deformation parameters α_1 and α_2. The error, that they display is roughly constant. This can be seen in figures 211 and 212.

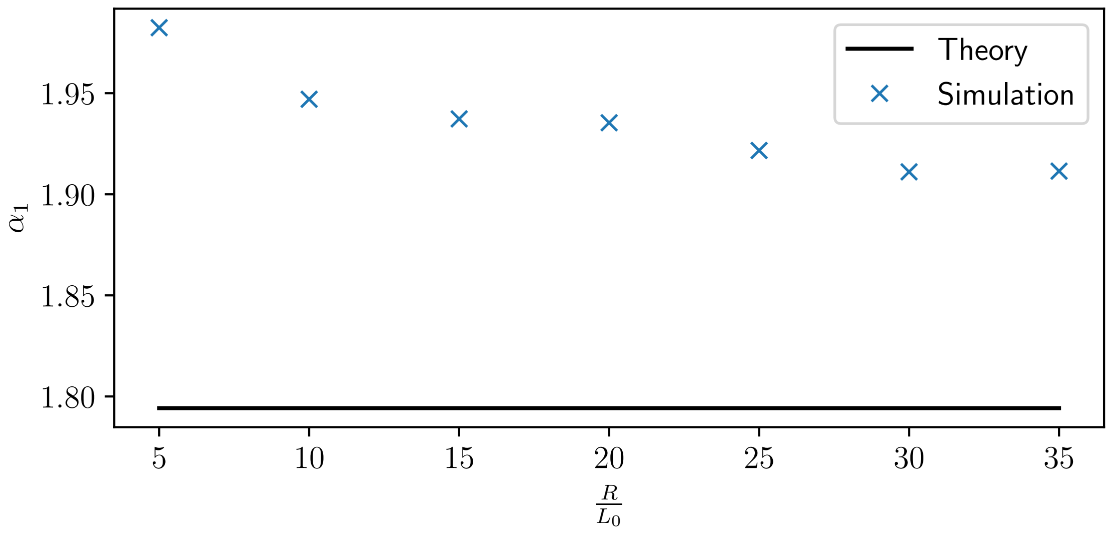


Figure 211: Plot of the Roscoe deformation parameter α_1 as a function of the cell resolution $\frac{R}{L_0}$. Curves for Roscoe theory and a Newtonian simulation are shown.

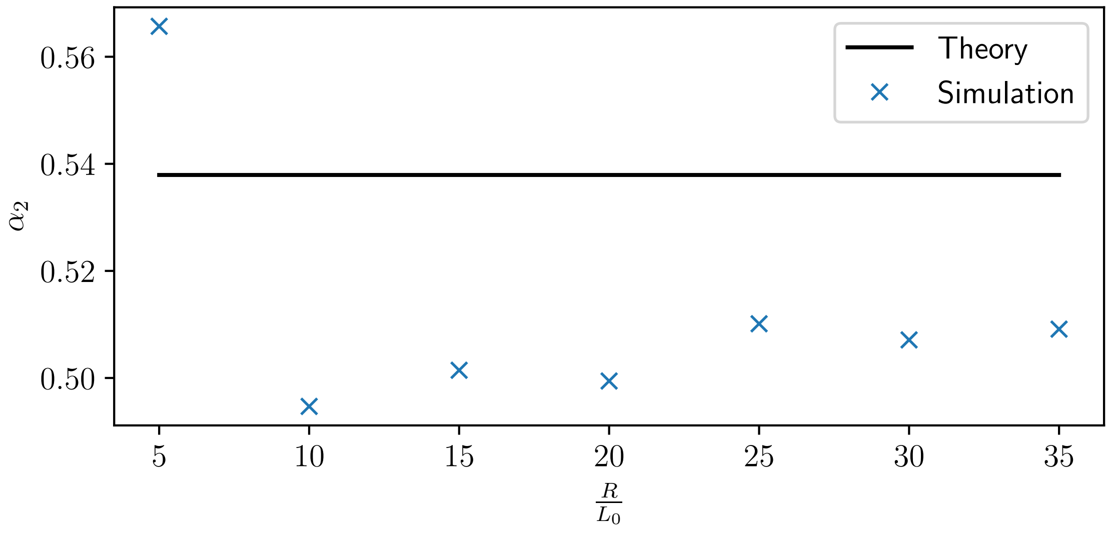


Figure 212: Plot of the Roscoe deformation parameter α_1 as a function of the cell resolution $\frac{R}{L_0}$. Curves for Roscoe theory and a Newtonian simulation are shown.

The Roscoe deformation parameter α_3 can be seen in figure 207.

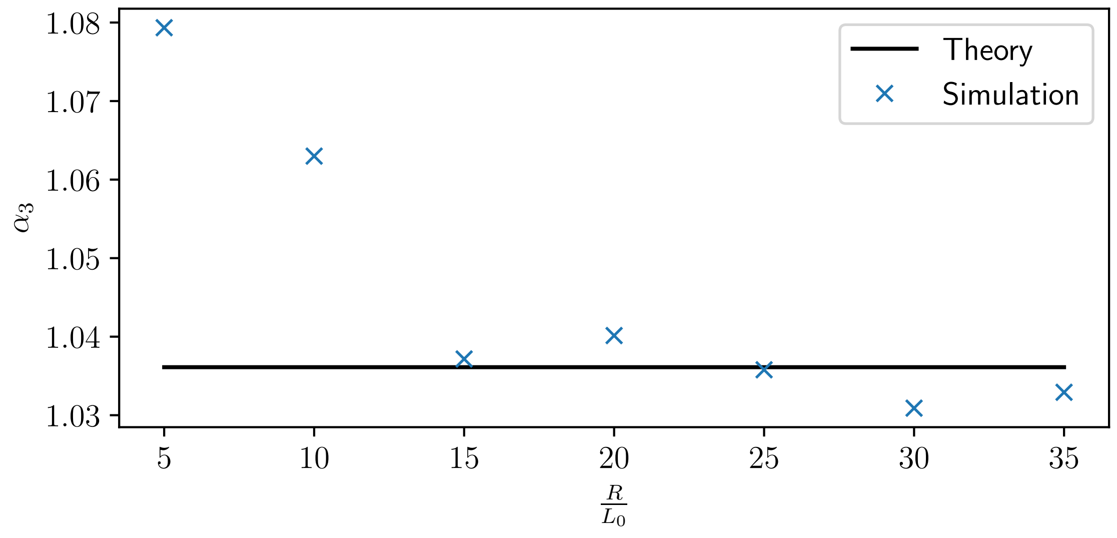


Figure 213: Plot of the Roscoe deformation parameter α_3 as a function of the cell resolution $\frac{R}{L_0}$. Curves for Roscoe theory and a Newtonian simulation are shown.

It does exhibit considerable errors for very small resolutions. Starting with medium resolutions any error is negligible. A similar behavior can be observed for the volume seen in figure 208.

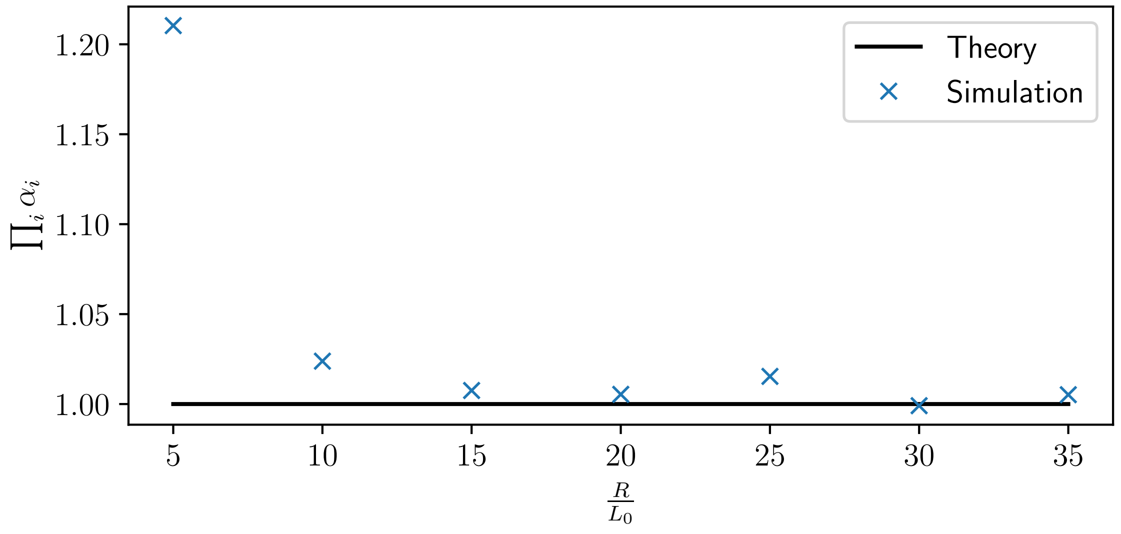


Figure 214: Plot of the cell volume as calculated from the Roscoe deformation parameters as a function of the cell resolution $\frac{R}{L_0}$. Curves for Roscoe theory and a Newtonian simulation are shown.

The alignment angle also shows a dependence of the error on resolution as can be seen in figure 209.

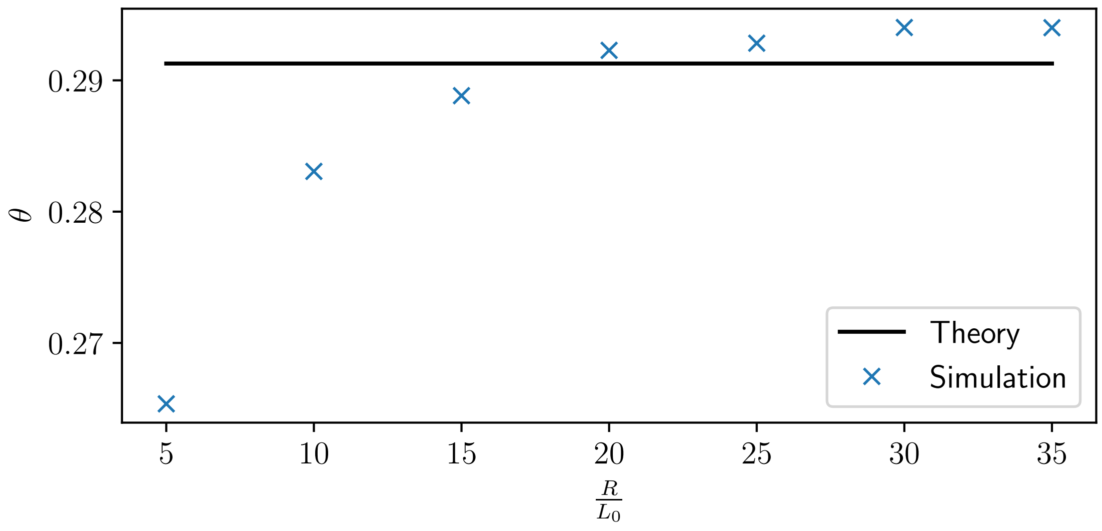


Figure 215: Plot of the Roscoe alignment parameter θ as a function of the cell resolution $\frac{R}{L_0}$. Curves for Roscoe theory and a Newtonian simulation are shown.

For small resolutions a considerable error is observed. Starting with medium resolution, the error changes sign and stays small. The biggest discrepancy can be seen for the tank-treading frequency. This can be seen in figure 210.

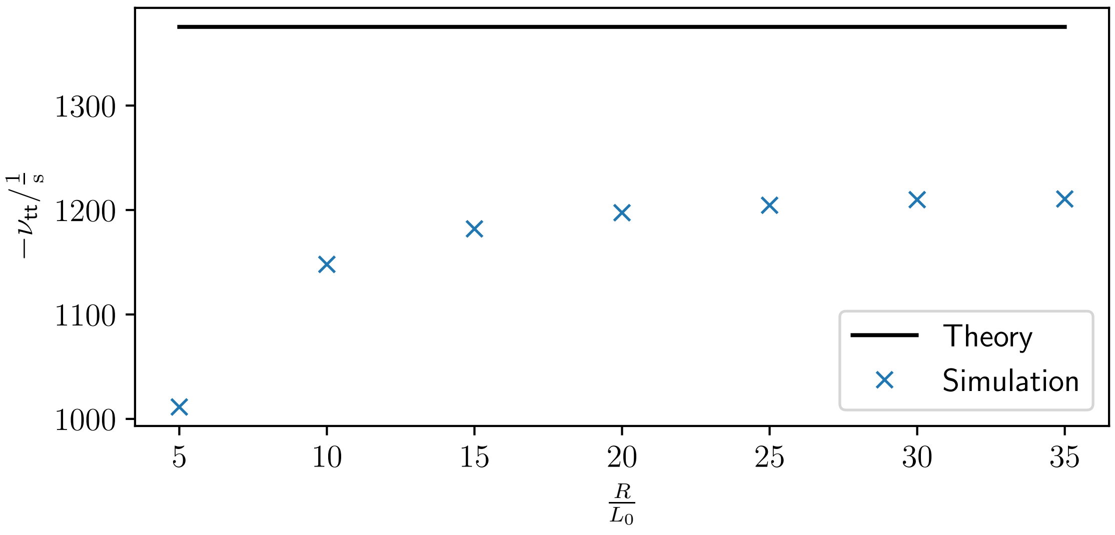


Figure 216: Plot of the tank-treading frequency ν_{tt} as a function of the cell resolution $\frac{R}{L_0}$. Curves for Roscoe theory and a Newtonian simulation are shown.

This shows a particularly bad error for small resolutions and otherwise a constant discrepancy. In total, it is quite evident, that resolutions below 10 to 15 are too small and introduce significant errors. Exceeding this, the error does not depend on resolution and does not vanish even for high resolutions. Consequently, one must conclude, that this too is not the source of the error. The Roscoe extraction fit often does not fit perfectly throughout the whole range of simulated times. This indicates, that the simulations are still changing and not fully in equilibrium. This is explored in the following.

X.3. Larger simulated time

The simulations seem to be in equilibrium, but this is not necessarily the case.

Old simulations

The simulations in this subsection are 3.25 a old. Consequently, both the setups and evaluation scripts do not match the standard set in the remaining thesis. It is not recommended to use *FluidX3D* versions this old. The simulation should rather be redone with a newer version to reproduce these results.

Simulation 58 is quite similar to simulation 56. The main difference being, that this simulation ran for several orders of magnitude longer. The Young's modulus is varied to achieve a varying Capillary number.

58 Roscoe for varying Ca_K in a Newtonian fluid

Box:	$180 \times 180 \times 176$
	$L_0 \approx 217\,\mathrm{nm}$
Fluid:	custom Newtonian $\eta_\mathrm{s} = 5\,\mathrm{Pa\,s}$
BC:	velocity (BC 2)
	$\dot\gamma = 100\,\frac{1}{\mathrm{s}}$
Cell:	Young's modulus variable
	Radius, Resolution $R \approx 3.91\,\mathrm{\mu m}, 18$
	Capillary number variable

With this, the deformations adhere to Roscoe theory as can be seen in figure 217.

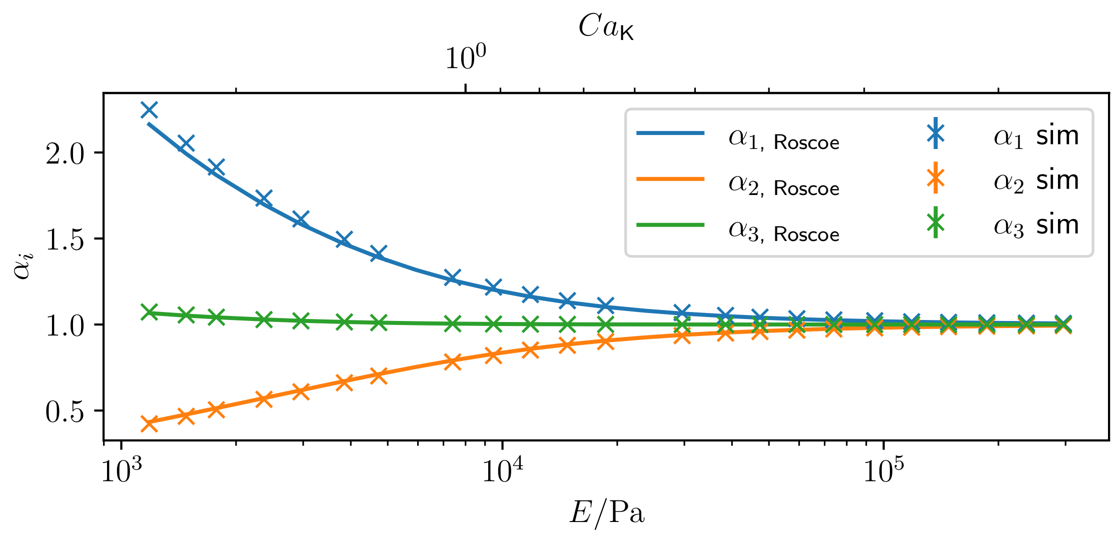


Figure 217: Plot of the Roscoe deformation parameters α_I as a function of the Young's modulus E. Curves for Roscoe theory and a Newtonian simulation are shown.

Only for large deformations, some discrepancy can be seen. It is small and now points towards a less deformed state as one would expect from a mesh with finite resolution. Consequently, the volume is conserved as can be seen in figure 218.

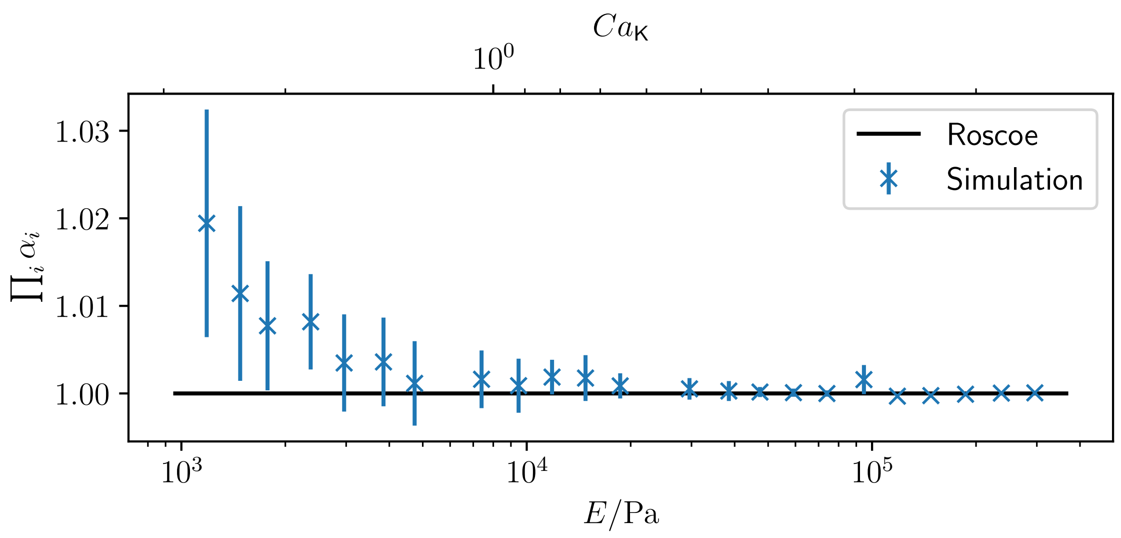


Figure 218: Plot of the cell volume as calculated from the Roscoe deformation parameters as a function of the Young's modulus E. Curves for Roscoe theory and a Newtonian simulation are shown.

The error-bars here, are the error from the fit. Similarly, the alignment angle agrees with Roscoe theory, even for large deformations. This can be seen in figure 219.

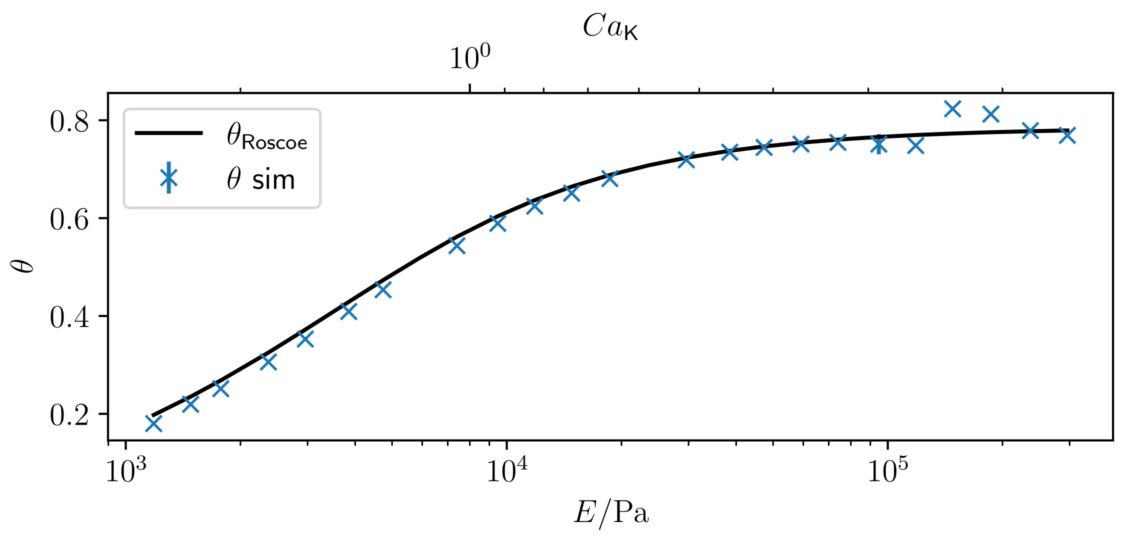


Figure 219: Plot of the Roscoe alignment parameter θ as a function of the Young's modulus E. Curves for Roscoe theory and a Newtonian simulation are shown.

Finally, the tank-treading frequency can be seen in figure 220.

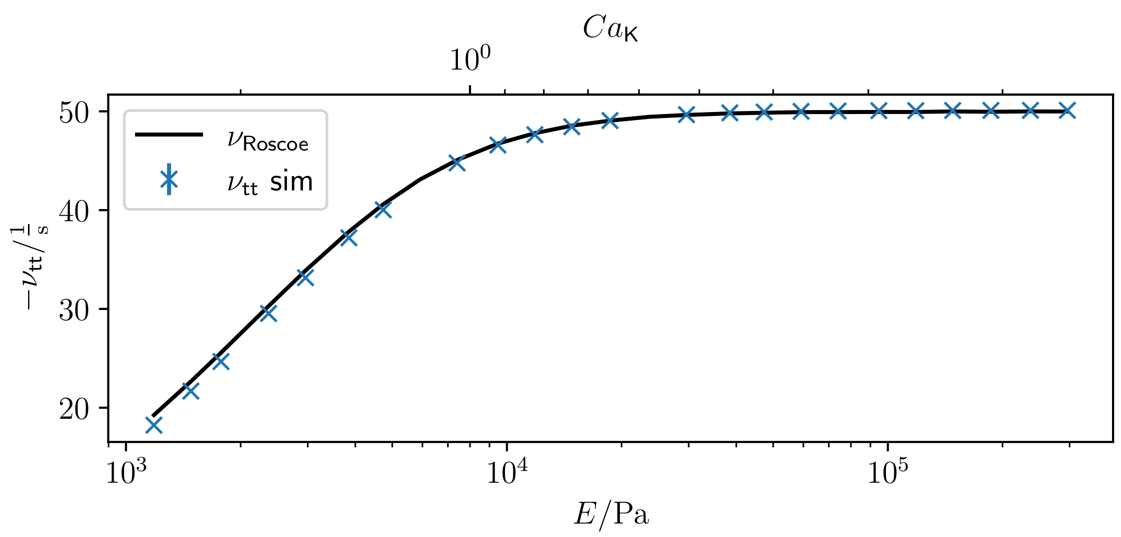


Figure 220: Plot of the tank-treading frequency ν_{tt} as a function of the Young's modulus E. Curves for Roscoe theory and a Newtonian simulation are shown.

For large deformations, there is a small discrepancy, but largely, it clearly is described by Roscoe theory. With this, a few closing remarks are made.

X.4. Closing remarks

Roscoe simulations take a very long time until they reach full equilibrium. Particularly, the tank-treading frequency takes time to establish itself. If one is willing to wait, *FluidX3D* is capable of producing accurate results in such cases. However, if one ends a seemingly stable simulation, errors in the single digit percentage range are to be expected. An improvement can be made, by only fitting the data at the end of the simulation. Using as little data as possible increases the fitting error slightly, but the parameters are reproduced more accurately in this region.

Y. Assorted support simulations

Some simulations require previous simulations to prepare parameters (see section 13.3.7). These are not very interesting, but still of some importance. They are listed here without further explanation.

59 CY printer needle preparation

Box:	$902 \times 77 \times 77$
	$L_0 = 1.\overline{3}\,\mu\text{m}$
Fluid:	CY::mc0_49 (fluid 8)
BC:	Inflow: mass-flow (BC 3)
	$\overline{v}_x = 5\,\frac{\text{mm}}{\text{s}}$
	Outflow: pressure (BC 4)
	slip walls (BC 6)

60 PTT printer needle preparation

Box:	$902 \times 77 \times 77$
	$L_0 = 1.\overline{3}\,\mu\text{m}$
Fluid:	PTT::mc0_49 (fluid 2)
BC:	Inflow: mass-flow (BC 3)
	$\overline{v}_x = 5\,\frac{\text{mm}}{\text{s}}$
	Outflow: pressure (BC 4)
	slip walls (BC 6)

61 CY LUT preparation

Box:	variable
Fluid:	CY::mc0_59 (fluid 9)
BC:	body-force (BC 13.3.2)
	$-\frac{\partial p}{\partial x} = 3.596 \times 10^7\,\frac{\text{Pa}}{\text{m}}$

62 PTT LUT preparation

Box:	variable
Fluid:	PTT::mc0_59 (fluid 3)
BC:	body-force (BC 13.3.2)
	$-\frac{\partial p}{\partial x} = 3.596 \times 10^7\,\frac{\text{Pa}}{\text{m}}$

63 RT-DC preparation boundary condition

Box:	$2 \times 146 \times 50$
	$L_0 = 417\,\mathrm{nm}$
Fluid:	PTT::mc0_59 (fluid 3)
BC:	body-force (BC 13.3.2)
	$-\frac{\partial p}{\partial x} = 1.5166 \times 10^7\,\frac{\mathrm{Pa}}{\mathrm{m}}$

64 RT-DC preparation initial condition

Box:	$1350 \times 146 \times 50$
	$L_0 = 417\,\mathrm{nm}$
Fluid:	PTT::mc0_59 (fluid 3)
BC:	Inflow: mass-flow (BC 3)
	simulation 63
	$Q \approx 0.04\,\frac{\mu\mathrm{L}}{\mathrm{s}}$
	Outflow: pressure (BC 4)

Z. Correcting the hyperbolic approximation

The discussion of the channels with constant extensional rate (section 25) in this thesis is based on the experiments published by Reichel *et al.* [99]. This publication contains valuable data, but is lacking in terms of theory. As some of the errors influence the reproduction of the experiments, they shall be discussed in the following. The equations for the center-line velocity u_0 and volume flow Q in a rectangular Poiseuille flow (see appendix F.1.1) are rather cumbersome as can be seen below.

$$u_0 = \frac{Gh^2}{8\mu} - \frac{4Gh^2}{\mu\pi^3}\sum_{n=0}^{\infty}\frac{1}{(2n+1)^3}\frac{1}{\cosh\left(\beta_n\frac{w}{2}\right)}(-1)^n \tag{831}$$

$$Q = \frac{Gh^3w}{12\mu} - \frac{16Gh^4}{\mu\pi^5}\sum_{n=0}^{\infty}\frac{1}{(2n+1)^5}\tanh\left(\beta_n\frac{w}{2}\right) \tag{832}$$

$$\beta_n = \frac{(2n+1)\pi}{h} \tag{833}$$

With the pressure gradient $G = -\frac{\mathrm{d}p}{\mathrm{d}x}$ and the dimensions w and h of the channel. This complexity drives the desire to approximate these equations. For $w \gg h$ a sensible approximation for the volume flow rate is as follows [129].

$$Q \approx \left(1 - 0.63\frac{h}{w}\right)\frac{Gh^3w}{12\mu} \tag{834}$$

This is found by sending $\frac{h}{w} \to 0$ and such $\tanh\left(\beta_n\frac{w}{2}\right) \to 1$. As the infinite sum drops quickly, only the first term is considered. The given relation follows with $12\frac{16}{\pi^5} \approx 0.63$. This is a somewhat sensible approach, as in the worst case ($w = h$), this approximation is around 12 % below the correct volume flow rate. Doing the same to the center-line velocity as was done by Reichel *et al.* [99] does however lead to the complete neglect of the sum and returns the result of a planar Poiseuille flow. This is up to 70 % larger, than the correct solution. The fraction $\frac{u_0}{Q}$, given by Reichel *et al.* [99] equation (14) as follows, reaches up to double its correct value in due to this approximation.

$$\frac{u_0}{Q} \approx \frac{3}{2}\frac{1}{h(w - 0.63h)} \tag{835}$$

This equation is central to the derivation of the shape. Therefore, it is clear, that such an approximation is not sensible. This error is the most relevant contribution to the insufficient consistency of the extensional rate observed in the experiments. It shall be noted, that the sums converge quickly and only taking the first summand would be a reasonable approximation. However, the hyperbolic functions would still introduce prohibitive complexity. It is therefore best to stick with a semi-analytical solution as this thesis discusses in section 25. Still, for comparisons' sake, the hyperbolic approximation resulting from this needs to be derived. The steps required are outlined in the SI of the aforementioned publication ((S2) – (S5)). As all of them are wrong, the derivation is listed correctly in the following. Start, with the derivation of the center-line velocity.

$$\dot{\epsilon} = \frac{\partial u_0}{\partial x} = \frac{\mathrm{d}u_0}{\mathrm{d}x} = -\frac{3Q}{2h}\frac{1}{(w(x)-0.63h)^2}\frac{\mathrm{d}w(x)}{\mathrm{d}x} \tag{836}$$

Split the differential.

$$-\frac{2h\dot{\epsilon}}{3Q}\mathrm{d}x = \frac{1}{(w(x)-0.63h)^2}\mathrm{d}w \tag{837}$$

Substitute the constants $A = -\frac{2h\dot{\epsilon}}{3Q}$ and integrate both sides.

$$Ax + C = -\frac{1}{w(x)-0.63h} \tag{838}$$

Solve for w.

$$w(x) = 0.63h - \frac{1}{Ax+C} \tag{839}$$

Now, the constants need to be determined. For this, insert the following.

$$w(0) = w_c \tag{840}$$

$$w(L_c) = w_u \tag{841}$$

The paper is inconsistent, which velocity is the inlet velocity and which is the outlet velocity. This choice conflicts with Fig.1 and the naming of the constants, but is more consistent with the math presented in the paper. Therefore, the values of these constants are $w_c = 250\,\mu\mathrm{m}$ and $w_u = 60\,\mu\mathrm{m}$. With these boundary conditions inserted, the full equation reads as follows.

$$w(x) = 0.63h - \left(Ax + \frac{1}{0.63h - w_c}\right)^{-1} \tag{842}$$

With A as follows.

$$A = \frac{1}{L_c}\left(\frac{1}{0.63h - w_u} - \frac{1}{0.63h - w_c}\right) = -\frac{2h\dot{\epsilon}}{3Q} < 0 \tag{843}$$

These equations provide corrected version for equations (15) and (16) from Reichel *et al.* [99]. Given the incorrect assumptions used at the start of the derivation, the shape produced by these equations, does of course not provide an adequate channel shape. Also, the prediction for the extensional rate offered by the relation for A is significantly off as well. Approximating the completely correct curve as presented in section 25 using a hyperbolic function similar as the one suggested by Reichel *et al.* [99] is however possible. The following hyperbolic function, which is a slight generalization, can be used to approximate the data.

$$w(x) = ah - \left[\frac{1}{L_\mathrm{c}}\left(\frac{1}{ah - w_\mathrm{u}} - \frac{1}{ah - w_\mathrm{c}}\right)x + \frac{1}{ah - w_\mathrm{c}}\right]^{-1} \tag{844}$$

Here a is a parameter, that can be used to optimize the shape. The solution presented in the discussed paper is recovered for $a = 0.63$. Its ideal value for a Newtonian fluid is (depending on which region is optimized) $a \approx 0.3$.

Eidesstattliche Versicherung

Hiermit versichere ich an Eides statt, dass ich die vorliegende Arbeit selbstständig verfasst und keine anderen als die von mir angegebenen Quellen und Hilfsmittel verwendet habe.

Weiterhin erkläre ich, dass ich die Hilfe von gewerblichen Promotionsberatern bzw. –vermittlern oder ähnlichen Dienstleistern weder bisher in Anspruch genommen habe, noch künftig in Anspruch nehmen werde.

Zusätzlich erkläre ich hiermit, dass ich keinerlei frühere Promotionsversuche unternommen habe.

Ort, Datum

Richard Kellnberger