HEMODYNAMIC INVESTIGATION OF ARTERIES USING THE LATTICE BOLTZMANN METHOD

by

GÁBOR ZÁVODSZKY

A THESIS SUBMITTED FOR THE DEGREE OF DOCTOR OF PHILOSOPHY.

Department of Hydrodynamic Systems
http://www.hds.bme.hu/
January 2015

E-MAIL: zavodszy@bme.hds.hu

WEBSITE: http://www.zavodszy.com/
Alulírott Závodszy Gábor kijelentem, hogy ezt a doktori értekezést magam készítettem és abban csak a megadott forrásokat használtam fel. Minden olyan részt, amelyet szó szerint, vagy azonos tartalomban, de átfogalmazva más forrásból átvettem, egyértelműen, a forrás megadásával megjelöltem.

Budapest, 2015. január 5.

Závodszy Gábor
ACKNOWLEDGEMENTS

I would like to express my gratitude to my supervisor, György Paál for establishing a great working environment and providing precisely the required help every time I ran out of ideas. His flexibility and open-minded attitude were often pivotal for moving my work forward.

I have received a lot of help from Orsolya Pock, who, besides being patient enough to correct several of my grammatical mistakes, also held together our living environment and managed most of the daily duties single-handedly, henceforth I could concentrate on my work. For her help I will be forever grateful.

It is the most probable that the completion of this work would not have happened without the friendly and helpful atmosphere created by the colleagues at the Department of Hydrodynamic Systems. I would like to express my gratitude towards my external advisor, István Csabai as well, who was quite influential on my larger-scale scientific world-view. I am also grateful for the support of the HPC-EUROPA2 program for making the validations on the supercomputer of the Erlangen Regional Computing Center (RRZE) possible, and for the computer scientist team there led by Thomas Zeiser for their very friendly and supportive attitude.

A large part of this work was supported by the Öveges scholarship founded by the Healthcare division of General Electric Hungary.
ABSTRACT

This work summarises my investigations about a human pathologic state concerning the circulation of the brain. The cerebral aneurysm is a vessel malformation that manifests itself as a lesion on the vessel wall alongside the Circle of Willis, which is responsible for carrying about 80% of the blood supply of the brain. The formation and development of this disease is in close correlation with the local hemodynamic factors. Therefore, I first select the most appropriate numerical tools for calculating the flow conditions inside the arteries. With these tools I simulate the emerging velocity field inside the geometry of a real aneurysm with a very high numerical resolution. I validate the results using available experimental results on the same geometry. As a next step, I apply the tested numerical methods on two further problems emerging within the context of aneurysms. In two dimensions I propose a simple numerical model for the haemostasis process in an arteriole that can reproduce the proper platelet density distribution and is able to recover a qualitatively correct thrombus shape compared to the experimental results available in literature. Furthermore, I investigate the properties of particle transport inside the validated flow field of the real aneurysm geometry which turns out to exhibit chaotic behaviour. Using the free-energy formalism I measure several key quantities of the emerging chaotic structure.

ÖSSZEFoglalás

A jelen munka egy, az agyi keringést érintő betegség körüli kutatásaimat foglalja össze. Az agyi aneurysma egy eredetileg kóros deformáció, ami az agy vérérellátásának 80%-át szállító Willis kör mentén jelentkezik az érfalon lévő kiöblösödések formájában. A betegség kialakulása és fejlődése szoros kapcsolatban van a helyi véráramlás tényezőivel. Emiatt először a megfelelő numerikus eszközöket választom ki, amelyek segítségével az artériákban fel lépő áramlás számítható. Ezekkel a szimulációs eszközökkel kiszámítom a kialakuló sebességteret igen nagy numerikus felbontással egy valós aneurysma geometriában. Az numerikus eredményeket ugyan ezen a geometrián végzett mérések eredményeihez hasonlítom. Ezután a kipróbált számítási módszereket két további problémára alkalmazom, amelyek az aneurysmát kapcsán
## CONTENTS

I  **Introduction**  
   1  **FOREWORD**  
      3  

2  **THE ARTERIAL CIRCULATION**  
   2.1  Intracranial aneurysms  
   2.2  Clinical treatment of intracranial aneurysms  
   2.3  Risk estimation of an intracranial aneurysm rupture event  
   2.4  The governing macroscopic equations  
      2.4.1  The mathematical notation  
      2.4.2  The general form of conservation expressions  
      2.4.3  Incompressible isothermal approximation  
      2.4.4  Dimensionless formulation  

3  **OVERVIEW OF THE APPLIED CFD METHODS**  
   3.1  The modelling choices for materials  
   3.2  The assumption of laminar flow  
   3.3  Numerical modelling methods  
   3.4  The Lattice Boltzmann Method  
   3.5  The construction of the lattice Boltzmann method  
      3.5.1  Starting from molecular dynamics  
      3.5.2  The Liouville equation  
      3.5.3  The Boltzmann equation  
      3.5.4  Connection with macroscopic quantities  

II  **LBM models and implementations for haemodynamic investigations**  
   29  

4  **LATTICE BOLTZMANN MODELS**  
   4.1  The BGK approximation  
   4.2  Further discretisations  
   4.3  Incompressible BGK model  
   4.4  Multiple relaxation time model  
   4.5  Entropic lattice Boltzmann model  
   4.6  Regularised model  

5  **FURTHER DETAILS AND SOLVER IMPLEMENTATIONS**  
   37
8.2  Methods .................................................. 87
  8.2.1  Flow field .............................................. 87
  8.2.2  Free-energy function ............................... 88
  8.2.3  Computation of chaotic properties ................. 89
8.3  Results .................................................. 92
  8.3.1  Qualitative results ................................... 92
  8.3.2  Quantitative results ................................. 96
8.4  Conclusions ............................................. 98
  8.4.1  Thesis 5 ................................................ 98

9  FINAL REMARKS ........................................... 99

BIBLIOGRAPHY .................................................. 103
Part I

Introduction
Modern research topics often incorporate fragments from different domains of science. Indeed, it is a natural process as our technology progresses that the areas of various scientific disciplines begin to overlap. As a consequence, the solutions of the emerging problems often require proficiency in several areas. Medical research topics are typically good examples since the investigations in this field often involve biological, physical, and biochemical processes, the combination of which is usually solved by means of computer science. The main subject of the investigations presented in the current work is a pathological human medical condition of the brain arteries, which is known as cerebral aneurysm.

The discussion itself is composed of three main parts each building upon the previous ones. Part I introduces the details of the medical disease forming the central question of my investigation and summarises our level of understanding of this problem according to the current literature. Since this medical state is in close connection with the fluid dynamics of the blood flow, a significant portion of the research done in this topic makes use of numerical flow analyses. Therefore, in this part I review some of the major steps from the research projects accomplished in this direction before overviewing the theoretical bases for these investigations. Then I briefly summarise two fundamentally different numerical schemes capable of producing answers for a number of questions emerged. One of them is the finite-volume method, which is more widely known among engineers and in the industry, and can be regarded as the more mature one. The other one is a younger technique that evolved to the level of practical usefulness in the mid-nineties. It is an analogical description of fluid mechanics recovering the same Navier-Stokes equations, yet it is based on a smaller length-scale description of the fluid. I discuss the latter one in more detail as I chose that method as the instrument for my investigations.

Since the aim is to resolve arterial flows, in Part II I explain the numerical schemes found to be suitable for this purpose in more detail as well as I cover some of the important differences among them. The three separate solver implementations I created are also discussed here. As far as I am concerned, the actual implementation and usage of a numerical scheme can be an important factor in the consideration of the validity of the results, therefore I discuss these solver versions in detail. Afterwards, I summarise the validation work which I carried out in a large German supercomputer centre to testify
the validity of the created implementations for the study of the previously introduced medical problem.

Finally, in Part III I make use of these validated solver codes to experiment with two different arising aspects of the topic. For each of them I present a further overview of the literature that is tailored more specifically towards the exact topic of these experiments. One is about a smaller-scale phenomenon of the circulatory system, where I simulate the behaviour of a few considered components of the blood in two dimensions. As a first step, I introduce a virtual force to recover the correct margination effect of the platelets, then I use the calculated platelet distribution and the macroscopic behaviour of the blood flow to simulate a blood clotting process in a small artery (an arteriole). Though the simulated process is not exactly the same as the one aneurysms undergo during thrombotisation, the influencing factors are similar and the numerical model is general enough, therefore I expect it to be easily extendible for larger-scale aneurysm thrombotisation processes in three dimensions in the future. The second experiment is about the fractal properties of the validated three-dimensional flow field emerging in a real aneurysm geometry. The chaotic behaviour of the blood flow seems important since it can influence particle level biochemical processes as well as it seems to encompass information about the geometrical properties of the malformation in a robust way.
The human circulatory system is usually defined as the composition of the closed cardiovascular system which is responsible for distributing blood inside our body and the lymphatic system which circulates lymph in the sub-arteriola vessel regime to provide oxygen and nutrients to the tissues and to dispatch carbon dioxide and wastes. The cardiovascular system is organised into two separate circulatory loops: the pulmonary circulation transports deoxygenated blood from the right ventricle of the heart to the lungs from where the newly oxygenated blood is pumped to the left atrium of the heart; the systemic circulation which carries the highly oxygenated blood from the left ventricle to the tissues of the body and then back to the right atrium of the heart. This circulation happens through blood vessels. These tubular structures contain a hollow area commonly referred to as lumen in medical practice. The vessel wall which resides around the lumen varies in structure depending on the vessel type. Capillaries are vessels with the smallest diameter found throughout almost every tissue of the human body. Due to their size, they can carry blood very close to cells in order to exchange oxygen, nutrients, and waste products. The wall of capillaries consists only of a thin cellular layer called the endothelium that is technically the minimum amount of structure possible between the blood and the tissues. Veins and venules are larger return vessels which carry the oxygen-sparse blood towards the right chambers of the heart. They are subject to nearly constant flow rates and contain valves to prevent blood flowing backwards that could (in theory) return waste materials to the tissues. Venules are tighter veins branching off from the ends of veins and connecting to the capillary level of blood vessels. Arteries and arterioles share plenty of similarities with veins and venules in their wall structures. An overview of the layers of the wall structure is shown in Fig. 2.1.

Their inner endothelial layer is surrounded by elastic layers containing fibrin and collagen threads to provide optimal wall resilience. The smooth muscle layer between the elastic layers helps the vessel expand or contract to regulate the flow of blood inside the lumen. Arterioles which branch off from arteries often have thinner wall structure as they are further away from the direct pumping activity of the heart, thus they are facing much lower blood pressures than arteries. Finally, arterioles connect to the other end of the capillaries completing the circulation.
The most probable cause of death in modern societies is usually the consequence of either some form of cancer or the degradation or extenuation of the cardiovascular system. In 2004, ischemic heart diseases together with cerebrovascular diseases were by far the leading cause of deaths worldwide (Mathers et al., 2009). One well-known form of the latter is called stroke. This name does not denote a single malformation but rather a collection of diseases that in the end lead to the same disorder: loss of brain function due to the disturbance in cerebral blood supply. These diseases can be further categorized into subgroups. The insufficiency of blood supply can be of ischemic type (approx. 83% of the cases) when a clot appears inside the lumen of the vessel blocking the flow. This clot usually forms either as a local narrowing of the vessel (stenosis), or it can be originated from a ruptured stenosis, or clotted blood from further upstream (embolism). The other type (accounting for approx. 17% of the cases) is the hemorrhagic stroke. It mainly manifests in two forms. One possibility is a vessel malformation where pathologic connections between the arterial and venous circulation exist, thus defying the proper transport of oxygen and other metabolic ingredients which in turn leads to weakened vessel walls and in several cases to the rupture of these vessels. The other possible manifestation form is an intracranial aneurysm (IA) (see Fig. 2.2 for an example).

These pathologic lesions are usually found at the bifurcation points or after highly curved sections along the Circle of Willis (Weir, 2002; Wiebers et al., 2004), which is the main blood supply of the brain (it is responsible for car-
rying around 80% of the total blood volume circulating in the brain). It is a moderately frequent disease that affects about 3 – 5% of the total human population (Rinkel et al., 1998; Vlak et al., 2011). The initiation and formation processes of such aneurysms are still not well-understood and are heavily researched areas which gives a rather good ground for a lot of scientific and medical debates. What is known according to several previous studies is that beside genetic factors (Hazama et al., 1986), these processes involve a great number of components including hemodynamics and the biomechanics of the wall (Gobin et al., 1994; Kerber et al., 1999; Kulcsár et al., 2011; Suzuki and Ohara, 1978). Kondo et al. (1997) showed that the initiation often requires increased flow conditions and elevated blood pressure (hypertension). It is hypothesised that the elevated wall shear stress along the vessel surface can be linked to the endothelial damage often encountered in the proximity of the aneurysmal locations (Stehbens, 1989) and (Steiger, 1989). This pathologic state of the vessel wall starts a cascade of processes including the remodelling of fibrin and elastin threads, apoptosis of endothelial and smooth muscle cells, and other inflammatory events (Sho et al., 2001). This leads to thinner and less flexible wall structure that goes through an inflation period that is driven besides the obvious inner pressure by structural remodelling. A broader overview of the current theories of the concerned pathogenesis is provided by Sforza et al. (2009).
2.2 CLINICAL TREATMENT OF INTRACRANIAL ANEURYSMS

The rupture of an intracranial aneurysm is fortunately a relatively rare event. Several people who carry an IA can live their lives through without ever having to worry about it. The reason for this is also a frightening fact about the disease: it does not usually produce symptoms until the rupture. Though after the rupture the implications are rather severe as the carried risk of mortality and morbidity rate is very high (Winn et al., 2002). Those lucky occasions when an aneurysm is discovered before the rupture are usually in conjunction with the medical imaging records of some different disease. Even when they are discovered in time, they make the medical practitioners face a hard decision about the treatment. Although the consequences of a possible rupture are severe, the medical intervention itself is far from being risk-free. Up to the mid-nineties, the most common technique to treat a cerebral aneurysm was to apply open brain surgery during which a metallic clip was put around the neck of the aneurysm sac (see Fig. 2.3a). A later, more sophisticated method for rupture prevention is based on the idea that even though we do not fully understand the initiation and formation of intracranial aneurysms, the rupture itself must require the mechanical load exerted by the blood flow inside the sac. Thus, if we can somehow slow the blood flow down inside the sac, we can reduce the possibility of a rupture. The solution (that is still widely applied today) is an endovascular treatment.

Figure 2.3: The most often employed current cerebral aneurysm treatment methods. (Source: several edited images from http://www.wikipedia.org)

A catheter is driven up inside the vessels of the arterial system started from the largest artery in the thigh, the femoral artery. When it reaches the aneurysmal neck, a metallic coil is inserted into the bulge through the catheter. As the coil starts to fill up the available space, the flow velocity decreases inside the sac (see Fig. 2.3b). Another technique is to use a stent graft similar to those utilised during coronary stenosis endovascular operations. While the
involved medical equipment might look very similar (both have a tubular shape made of a densely woven metallic net), their purpose is entirely different. For stenosis treatment, the stent is used from inside the vessel to support the wall and to prevent any further local narrowing whereas for the intracranial aneurysms it is used to cover the neck of the aneurysm (see Fig. 2.3c) to create an artificial hydrodynamic resistance and to hamper the communication between the parent artery and the aneurysm bulge. In some cases the latter two methods are used in conjunction in order to improve efficiency.

2.3 RISK ESTIMATION OF AN INTRACRANIAL ANEURYSM RUPTURE EVENT

The above-mentioned treatment methods usually carry some inherent risks (mainly in the form of complications) for the patient (Wiebers, 2003). Alas, the physician has to decide if the risk of the aneurysmal rupture is higher than the risk of the intervention. As we do not fully understand the initiation and formation processes, the fate of an aneurysm is rather hard to predict. The question of the intervention is usually decided by relying on the previous experience of the medical practitioner and some classifications of the aneurysmal geometries. The most common classification was based on the mere size of the aneurysm (Rinkel et al., 1998) or some size ratio like the diameter of the aneurysmal neck compared to the largest diameter of the bulb.

Later, strong indications have been found (Cebral, Castro, Burgess, et al., 2005; Lasheras, 2007; Piccinelli et al., 2009; Steinman et al., 2003) that the properties of the local blood flow have a significant impact on the aneurysmal behaviour. Thus, a newer, more refined characterisation method was developed to correlate the rupture risk (among other features of the pathogenesis) to the local wall shear stress (Boussel et al., 2008; Sforza et al., 2009; Shojima et al., 2004) or the wall shear stress combined with the wall shear stress gradient (Kulcsár et al., 2011). In this way, some of the properties of the emerging blood flow are incorporated in the decision, not just the bare geometry. Further investigations suggested the importance of one particular property of the wall shear stress, namely its periodic change in time quantified by the oscillating shear index (Kawaguchi et al., 2012). The most recent works bring a new concept called energy loss (Y. Qian et al., 2011; Takao et al., 2012) being defined as the difference in total energy (the sum of pressure and kinetic energy) of the flow inside the concerned vessel section between the aneurysmal and the reconstructed pre-aneurysmal states. These are just the major contenders for the improvement of the risk assessment, for a broader overview see the work of Zanaty et al. (2014).

Though each of these classification methods shows some level of correlation with the ruptured or unruptured state of the examined aneurysms, the
results are still controversial (Takao et al., 2012) and none of them seems to show a high enough statistical validity to give a basis for a robust and unequivocal decision making process. Consequently, all the information points towards the fact that further improvements of the treatment methods and further developments of the risk estimation will incorporate attributes not only of the geometry, but also of the evolved flow field inside the aneurysm. Indications for this can also be found in the work of Cebral, Castro, Burgess, et al. (2005). It brings into attention the techniques which can accurately model the emerging flow field inside a real aneurysm geometry.

2.4 THE GOVERNING MACROSCOPIC EQUATIONS

The toolset of Computational Fluid Dynamics (CFD) is capable of providing both qualitative and quantitative predictions of fluid flows by means of mathematical and numerical modelling. In general it consists of the following ingredients:

THE MATHEMATICAL DESCRIPTION usually refers to a set of partial differential equations (PDEs) which describes the evolution of the observable physical quantities in space and time. These equations are usually based on phenomenological or theoretical considerations. When we consider a set of mathematical descriptions of a physical process acting on different scales to be well-understood, it usually refers to the fact that we can also describe the underlying smaller-scale processes and in the limit of the larger scale the two descriptions match.

A NUMERICAL METHOD is used to discretise the equations, usually both in space and time. This results in a new set of discrete equations that provides an approximation for the original problem. The problem might require further treatment, but eventually, a set of algebraic equations are produced that are ready to be implemented in some computational environment.

COMPUTATIONAL TOOLS form the executive layer in the workflow, incorporating both the implementations of the algebraic equations and various other pre- and post-processing utilities.

2.4.1 The mathematical notation

In this thesis, I use two formalism for mathematical notations: the vector notation and the index notation. In vector notation the vector quantities are denoted by using a bold-faced font and an arrow above them (e.g., \( \vec{a} \)), while tensors are denoted by bold faced font and a special sign above their character
In the index notation the component of vectors and tensors are expressed as Greek letter indices (e.g., \( a_\alpha \) and \( T_{\alpha\beta} \)). With the double indices I follow the convention that whenever an index is repeated twice within the same term, a summation over the possible values of that index is assumed. In Table 2.1, I briefly summarise the mathematical operations I employ most frequently within the bounds of this work.

<table>
<thead>
<tr>
<th>Mathematical operation</th>
<th>Vector notation</th>
<th>Index notation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scalar product</td>
<td>( \lambda = \mathbf{a} \cdot \mathbf{b} )</td>
<td>( \lambda = a_\alpha b_\alpha )</td>
</tr>
<tr>
<td>Dyadic vector product</td>
<td>( \mathbf{A} = \mathbf{a} \mathbf{b} )</td>
<td>( A_{\alpha\beta} = a_\alpha b_\beta )</td>
</tr>
<tr>
<td>Gradient</td>
<td>( \mathbf{a} = \nabla \lambda )</td>
<td>( a_\alpha = \partial_\alpha \lambda )</td>
</tr>
<tr>
<td>Divergence of a vector</td>
<td>( \lambda = \nabla \cdot \mathbf{a} )</td>
<td>( \lambda = \partial_\alpha a_\alpha )</td>
</tr>
<tr>
<td>Divergence of a (second-order) tensor</td>
<td>( \mathbf{a} = \nabla \cdot \mathbf{T} )</td>
<td>( a_\alpha = \partial_\beta T_{\alpha\beta} )</td>
</tr>
<tr>
<td>Tensor contraction</td>
<td>( \lambda = \mathbf{A} : \mathbf{B} )</td>
<td>( \lambda = A_{\alpha\beta} B_{\alpha\beta} )</td>
</tr>
</tbody>
</table>

Table 2.1: Summary of the mathematical notations employed throughout the following sections.

2.4.2 The general form of conservation expressions

The most common way in the topic of my study is to describe the dynamics of the fluid at a continuum level. In this way, we are neglecting the particle level behaviour, and thus we can gain a reduced set of macroscopic variables that provides sufficient information about the state of the fluid. If we observe some physical phenomenon exhibited by the fluid that cannot be described by our variables, we can extend our model by introducing new variables and PDEs. This model development technique tends to be more on the phenomenological side. A different method of acquiring a suitable description is to understand and describe the microscopic nature of the fluid. Hence, the macroscopic properties of the fluid are just emergent features from the underlying smaller-scale dynamics. This bottom-up approach is more of a peculiarity of the theoretical model development.

From the macroscopic point of view, the state of a simple fluid can be defined using the following variables:

- \( \rho \), the fluid density
- \( \mathbf{u} \), the velocity of the fluid
- \( p \), the pressure
- \( E \), the energy
- \( T \), the temperature
The governing equations describing the development of these quantities can either be derived from the microscopic behaviour of the fluid using the techniques of statistical physics or one can acquire them from the macroscopic interpretation by the means of prescribed conservation laws acting upon these variables. The most commonly used conservation expressions in divergence form are (Anderson et al., 1995):

\[ \frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \vec{u}) = 0, \tag{2.1} \]

\[ \frac{\partial (\rho \vec{u})}{\partial t} + \nabla \cdot (\rho \vec{u} \vec{u} + p \hat{I} - \hat{\tau}) = \rho \vec{g}, \tag{2.2} \]

\[ \frac{\partial E}{\partial t} + \nabla \cdot \left( (\rho E + p) \vec{u} - \kappa \nabla T - (\vec{u} \cdot \hat{\tau}) T \right) = \rho (q + \vec{g} \cdot \vec{u}), \tag{2.3} \]

where \( \vec{g} \) denotes the gravitational acceleration, \( \kappa \) denotes the thermal conductivity, \( q \) is a heat source term, and \( \hat{\tau} \) denotes the deviatoric stress tensor. These three equations stand for the mass, momentum, and energy conservation respectively. The deviatoric stress tensor in Newtonian fluids obeys the following definition:

\[ \hat{\tau} = -\nu' \rho (\nabla \cdot \vec{u}) \hat{I} + 2\nu \hat{S}, \tag{2.4} \]

where the strain rate tensor \( \hat{S} \) is defined as:

\[ \hat{S} = \frac{1}{2} (\nabla \vec{u} + (\nabla \vec{u})^T). \tag{2.5} \]

The constants \( \nu \) and \( \nu' \) denote the kinematic shear and bulk viscosity. Equation 2.2 is commonly referred to as compressible Navier-Stokes equation. Together with the other two mentioned conservation equations they present \( D + 2 \) equations for \( D + 4 \) unknowns in a \( D \)-dimensional space. Thus, the system of equations is not yet closed, we still need some expression that can couple the pressure with the temperature and the temperature with the energy.

### 2.4.3 Incompressible isothermal approximation

In isothermal flows, the effects of Eq. 2.3 are not taken into account. This way, an equation of state that relates the pressure to the density of the fluid is sufficient to close the system of equations. A form of the equation of state for ideal gas is used in the following:

\[ p = c_s^2 \rho, \tag{2.6} \]

where \( c_s \) denotes the speed of sound. This means that the pressure scales linearly with density. For an incompressible fluid, density takes a constant value \( \rho = \rho_0 \) which does not vary in space and time. Incorporating this
additional information, the conservation law for mass can be simplified and the velocity field becomes divergence-free:

\[ \nabla \cdot (\rho \vec{u}) = 0, \]  
\[ \nabla \cdot \vec{u} = 0. \]  

The equation 2.8. is actually widely used as the definition for incompressibility. The momentum equation can also be simplified to:

\[ \partial_t \vec{u} + (\vec{u} \cdot \nabla) \vec{u} = -\frac{1}{\rho_0} \nabla p + \nu \nabla^2 \vec{u}. \]  

This form is often referred to as the incompressible Navier-Stokes (or Navier-Stokes for short). To fulfill the incompressibility criteria (Eq. 2.8) with a sufficient pressure value, one has to take the divergence of Eq. 2.9 and solve it together with Eq. 2.8 for \( p \). This action yields the Poisson equation:

\[ \nabla^2 p = -p_0 (\nabla \vec{u}) : (\nabla \vec{u})^T. \]  

This time-independent equation replaces Eq. 2.8 in the system of equations. During computation, this Poisson equation is evaluated iteratively at every time-step to adjust the value of the pressure in such a way that the velocity conforms the restriction of being divergence-free.

2.4.4 Dimensionless formulation

Leaving the physical units behind is useful for several reasons. One of those being that these physical systems are easier to compare while another reason is that this helps the numerical computation as we only have to implement generic equations acting upon the dimensionless values. For this conversion, we have to choose some general scales that are representative of the currently examined flow. The length scale \( l_0 \) can be the length of an obstacle immersed into the fluid or the diameter of a pipe in which the flow takes place. The \( t_0 \) time scale can be for instance the time it takes for a fluid particle to pass the immersed obstacle. The physical values describing time (\( t \)) and position (\( \vec{r} \)) are replaced by their dimensionless counterparts:

\[ t_d = \frac{t}{t_0} \quad \text{and} \quad \vec{r}_d = \frac{\vec{r}}{l_0}. \]  

Using these dimensionless variables, the same transformation can be carried out for further variables and operators:

\[ \vec{u}_d = \frac{t_0}{l_0} \vec{u}, \quad \rho_d = \frac{\rho}{\rho_0}, \quad p_d = \frac{t_0^2}{\rho_0 l_0^2}, \]  
\[ \partial_{t_d} = \frac{t_0}{l_0} \partial_t, \quad \nabla_d = l_0 \nabla. \]  

(2.12)
Applying these variable changes to Eq. 2.9 and Eq. 2.8, we can gain the dimensionless incompressible Navier-Stokes equations:

\[
\partial_t \vec{u}_d + (\vec{u}_d \cdot \nabla_d) \vec{u}_d = -\nabla_d p_d + \frac{1}{Re} \nabla_d^2 \vec{u}_d, \\
\nabla_d \vec{u}_d = 0,
\]

(2.13)

where \( Re = \frac{l_0^2}{t_0 \nu} \) is the dimensionless Reynolds number. In our modelling approximation two flows within the same geometry (except for a scaling factor) are said to be equivalent if they obey the same Navier-Stokes equation and their Reynolds numbers match.
3

OVERVIEW OF THE APPLIED CFD METHODS

3.1 THE MODELLING CHOICES FOR MATERIALS

The investigation of the role of blood flows has a long-term history and is of special interest for several cardiovascular diseases. For aneurysms, there exists a fine palette of literature about the possible consequences of the different hemodynamic properties, e.g., the work of Gobin et al. (1994); Gonzalez et al. (1992); Shojima et al. (2004). These studies are usually carried out on idealised aneurysm geometries or on aneurysms surgically induced in animals. Although they shed light on several important details of this disease, they fail to give a patient-specific insight, thus are unable to help in the estimation of the rupture risk. Szikora et al. (2008) showed that the exact geometry of the aneurysm has a major impact on the flow field. To circumvent the above-mentioned problem, the usual approach is to run numerical flow simulations using a recorded patient-specific geometry with general velocity boundary conditions (Cebral, Castro, Burgess, et al., 2005; Paál et al., 2007). (The numerical modelling of blood flows is actually unavoidable as the current in-vivo imaging modalities, such as ultrasound, MRI, or CT based methods, are too limited to be used for flow pattern quantification in cerebral vessels. An overview of the experimental and computational methods in cardiovascular fluid mechanics can be found in Taylor and Draney (2004).)

In these numerical computations the blood itself is usually represented as an incompressible, continuous Newtonian fluid, therefore most CFD studies are based on solving some discretised version of the three-dimensional unsteady incompressible Navier-Stokes equations. For aneurysmal flows, however, the non-Newtonian properties of the blood can sometimes become significant as these flows may present some slow flow regions. This important question was recently addressed by Castro et al. (2014) comparing the implications of the Newtonian and the Casson rheological models on wall shear stress in real cerebral aneurysm geometries. Their conclusion is that even though some differences can be found, they are not statistically significant. The other substantial question is the effect of the imposed wall model, whether the use of a rigid wall approximation can be justified. Elastic wall models usually employ a simple linear material characteristic that obeys Hooke’s law. This also raises some questions as the wall behaviour is far from being linear. Furthermore, it is not a “passive” material in the sense that it reacts to various changes in the flow pattern (e.g., with the help of glycocalyx chains) as well as to the changes in the emerging stress distributions. This problem seems enormously difficult
to tackle from theoretical starting points as the biomechanics of the vessel wall is rather complex, with numerous processes involved that can actively respond to blood flow changes inside the lumen. Also, the structure of the vessel wall together with the parameters of the mentioned processes show strong spatial inhomogeneity. Apart from these complexities, an accurately formalised wall model would require patient-specific information about the material properties (e.g., wall thickness, strength, elasticity, support from the surrounding tissues, etc.) all along the given vessel segment, especially for the material of the aneurysmal sac, where the cellular structure undergoes inflammatory processes and apoptosis. In a study of Oubel et al. (2007) a more empirical path was chosen: they recorded the vessel wall motion during a cardiac cycle and imposed it as a time-dependent boundary condition during numerical simulations. The results seem to suggest that even though differences arise compared to the rigid wall case, the main characteristics of the flow patterns (such as the location and the size of the inflow jet and the complexity and stability of the intra-aneurysmal flow pattern) were not significantly altered. More recently, in the work of Józsa and Paál (2014), two abdominal aortic aneurysm geometries were tested with a slightly modified version of the hyperelastic wall model often used in the literature (Khanafer et al., 2009; Leung et al., 2006). They found only minor deviations in the qualitative picture of the flow field between this hyperelastic model and the rigid wall model, if the rigid wall geometry was recorded at the time of the systolic peak, when the enclosed volume of the vessel is the largest. Moreover, for cerebral aneurysms, the typical amount of deformation is much less than for aortic aneurysms. For my work, based on the information outlined above, I chose to work with the rigid wall and Newtonian fluid approximations.

### 3.2 THE ASSUMPTION OF LAMINAR FLOW

Approaching from the aspect of geometrical similarities, it is known that a stationary flow remains laminar up to the Reynolds numbers of around 2300 in straight pipes. The Reynolds number (Re) in the case of pipe flows is commonly defined as \( Re = \frac{\|\mathbf{u}\|L}{\nu} \), where \( \|\mathbf{u}\| \) denotes the magnitude of the mean flow velocity, \( L \) is the diameter of the pipe, and \( \nu \) is the kinematic viscosity of the fluid. For vessels, this definition can be extended by redefining \( \mathbf{u} \) as the component of velocity which is perpendicular to the cross-section of the vessel (which is in turn perpendicular to the centreline). Fully developed turbulence in turn occurs around \( Re = 4000 \). The pulsatile nature of the examined flows may both increase and decrease the threshold for the transition. In particular, flow deceleration typically promotes while acceleration delays the onset of the turbulent spots.
From a clinical point of view, most fluid flows in our body are believed to be laminar in healthy individuals, with the exception of the flow around heart valves and inside some sections of the aorta. This has simple physiological reasons, such as that turbulence would introduce increased friction, thus making the fluid transport less efficient. A cerebral aneurysm is a pathological deviation from the healthy case regarding (among other biological factors) the geometry of the vessel. It has been long known that despite the lower Reynolds numbers occurring in brain arteries (typically from 50 to several hundreds), some aneurysmal malformations can present indications of turbulence (Ferguson, 1970). In these cases, the disturbed flow usually causes “murmur” sounds detectable by phonocatheters or even tinnitus audible by the patient (Austin and Maceri, 1993; Sonmez et al., 2007). Most of the times, however, the presence of the aneurysm is symptomless. Turbulence itself is defined by Hinze (1975) as:

Turbulent fluid motion is an irregular condition of flow in which the various quantities show a random variation with time and space coordinates, so that statistically distinct average values can be discerned.

These flows can present events over various length scales, which are bounded above by the geometric dimension of the flow field and bounded below by the molecular scale of the fluid material. The smallest scale of events that defines the required resolution of a direct numerical simulation (DNS) scales with the Reynolds number as (Canuto et al., 2010):

\[
\frac{L}{\eta} = Re^{(3/4)},
\]

where \( L \) is the macroscopic integral length scale and \( \eta \) denotes the smallest microscopic length scale often referred to as Kolmogorov length scale. The question of turbulence or even transitional flow within the bounds of this dissertation may only arise regarding one geometry discussed in both Chapter 6. and 8. from different aspects. During the simulations I chose to work with a laminar flow model since the numerical grid resolution was high (c.f., Chapter 6.). I hypothesised based on the moderate Reynolds number of the flow that the smallest scale of events is larger than the resolution of the applied numerical grid, thus even if some spots of turbulence arose, its effects were described appropriately. The results of the validation seem to reinforce this hypothesis.

### 3.3 Numerical Modelling Methods

The two main candidates for carrying out these kinds of numerical flow simulations are the finite volume (FVM) and the lattice Boltzmann method (LBM).
These two methods were compared in various works for both accuracy and computational efficiency. They were compared in two dimensions for a steady flow around an octagonal cylinder by Noble, Georgiadis, et al. (1996), and for both steady and unsteady flows around a square cylinder by Breuer et al. (2000). He, Duckwiler, et al. (2009) carried out a three-dimensional flow comparison inside an aneurysm geometry using OpenFOAM as the finite volume solver. Axner et al. (2009) simulated a section of the aorta and a side branch in 3D with both methods and, as in the previous pieces of work, they found a reasonable agreement.

While both methods solve some form of the Navier-Stokes equations (details on this statement will follow in Section 3.4) they are definitely not equal from either a technical or an implementational point of view. One of the difficulties of solving fluid flow problems in vessel geometries often arises from the fact that these cases can incorporate multiple physical scale levels. Bernaschi et al. (2009) implemented a multi-scale fluid flow simulation with the capability of coupling LBM with molecular dynamics. Another problematic point usually encountered is the generation of an appropriate numerical mesh. This step has heavy influence on the accuracy and on the computational resource requirements of the simulation. For highly curved and irregular geometries (such as a vessel section) it tends to be complicated. The LBM method uses a regular cubic-shaped numerical grid usually called voxel space representation. This simplifies the creation of a spatially discretised numerical grid up to the point where it does not actually require any human interaction. Technically it could even be automated. Furthermore, this method carries at least one more computational property that makes it very appealing in today’s computational environments: it is inherently parallel due to being explicit in time and local in space, as Melchionna et al. (2010) demonstrated it with an efficient multi-GPU based implementation of LBM capable of simulating a large-scale cardiovascular flow system. Later Bisson et al. (2012) also demonstrated this multi-scale implementation to exhibit linear scaling when running across a cluster of 32 GPUs. During the course of my work I have chosen to work with the lattice Boltzmann method for my blood flow investigations. To summarise the above-mentioned main three reasons for this decision:

- It is relatively easy to couple it with other physical processes, even with those on different time- or length-scale.
- Numerical grid generation tends to be more simple and less error-prone than with the FVM.
- The method is inherently parallel, thus it is a good fit for many-core computational environments.
For the different computations mentioned in this dissertation I have created three separate LBM solver implementations. Two of them build upon open source components and the third is a completely in-house implementation built from scratch. Their differences and their purposes are outlined in chapter 5.

3.4 **THE LATTICE BOLTZMANN METHOD**

As already stated in the previous section, the dynamics of the fluid motion can be described in several ways depending on the required detail. In my current interest I would like to recover the macroscopic behaviour of the fluid governed by Eq. 2.9. The continuum assumption used in that equation is valid when the macroscopic properties (such as pressure, velocity, density, etc.) are insensitive to the exact position and velocity of the individual particles that make up the fluid. Figure 3.1 shows the different possible description levels. As we approach the lower levels of description (where more details of the microscopic properties of the fluid can be recovered), the computational cost increases steeply. Therefore, it is important to plan carefully with the approximations and restrictions of the individual models. For the completeness of the picture it has to be mentioned that there exists a well-established further level of description for below the scales governed by molecular dynamics, the domain of quantum mechanics where the sub-particle dynamics are described by some form of Schrödinger’s equation.

The level of the required description can be characterised by the dimensionless **Knudsen number** (Kn), which is defined as the ratio of the mean free path of the particles (λ) and the smallest characteristic length in the examined flow (l):

$$Kn = \frac{\lambda}{l}.$$  

(3.1)

It means that the continuum approximation is only valid at low Knudsen number as in that case the average distance travelled between collisions by the particles is insignificant compared to the characteristic lengths. Thus, the events of a single particle have little effect on the macroscopic quantities, which means that the fluid acts as continuum. For high Knudsen numbers the scale of the physical processes that we would like to recover approaches the scale of the particle events. In this regime the laws of molecular dynamics can produce meaningful answers. Between these two extrema the statistical description of the molecular dynamics creates a link. In this intermediate level one still uses the kinetic theory of molecules, but instead of tracking each individual particle, the equations are defined upon their statistical distribution functions. The lattice Boltzmann method follows a rather pragmatic way as I shall briefly outline it in the next section. Its root originates from the
kinetic theory of gases from where the description is averaged over the phase space to gain the statistical Boltzmann equation. The degrees of freedom of the distribution function is then reduced using a set of well chosen approximations to finally gain a discretised version of the Boltzmann equation that is applicable over a wide range of Knudsen numbers including the continuum range.

In my opinion, the lattice Boltzmann method (in contrast with for example the Navier-Stokes equations) is constructed rather than derived using rigorous reduction in the degrees of freedom across multiple scale levels. It is exactly this upside-down approach that makes it so versatile and capable of handling multi-scale physics within a single framework.

3.5 THE CONSTRUCTION OF THE LATTICE BOLTZMANN METHOD

Historically, the lattice Boltzmann method (LBM) evolved out of the lattice gas automata, which is an artificial microscopic model for gases. It was shown later that the LBM can also be recovered from the Boltzmann equation using some standard discretisations. In this section I shall follow the more theoretical path instead of the historical one in the construction of the LBM based on molecular dynamics. This overview progresses in large steps highlight-
ing only the key points. For a more detailed description on the kinetic theory part please see the work of Harris (2004) and for the steps from the Boltzmann equation the work of Guo and Shu (2013).

3.5.1 Starting from molecular dynamics

Molecular dynamics describes the time evolution of a given set of point masses. These particles can lack the information about their orientation and they can be thought of as finite-sized hard spheres with homogeneous mass distribution. Their motion is governed by Newton’s equation, thus they can be characterised by a Hamiltonian \((H)\) or a Lagrangian \((L)\) system as follows (see e.g., in the work of Tadmor and Miller (2011)):

\[
\begin{align*}
\frac{\partial H}{\partial \vec{x}_i} &= -\frac{d\vec{p}_i}{dt}, \\
\frac{\partial H}{\partial \vec{p}_i} &= \frac{d\vec{x}_i}{dt}, \\
\frac{\partial H}{\partial t} &= -\frac{\partial L}{\partial t}. \quad (3.2)
\end{align*}
\]

where \(\vec{p}_i\) denotes the momentum of the \(i\)-th particle. For a system of \(N\) particles this yields \(6N\) first-order partial differential equations. The macroscopic flow can be calculated from the resolved \(3N\) positions and \(3N\) momenta. The advantage here is that these equations are easy to solve. The drawback (at least from the viewpoint of my interests) is that the examined fluid domain would contain particles in the order of the Avogadro number \((\approx 6 \times 10^{23})\). With the available computational resources this would simply be impossible to solve. Naturally, a given macroscopic state of the fluid does not correspond to a single unique microscopic state of the particles. For any given macroscopic state there is an infinite number of corresponding microscopic states that are equivalent from the macroscopic point of view. If we plot all these equivalent states in phase space, then we can express any given microscopic state as a probability density function in this phase space of the particles.

3.5.2 The Liouville equation

The Liouville equation is the governing equation for this N-particle density function. This function:

\[
F_N(\vec{x}_1, \vec{x}_2, \ldots, \vec{x}_N, \vec{p}_1, \vec{p}_2, \ldots, \vec{p}_N, t), \quad (3.3)
\]

represents the probability that the state of the particle system is at a particular point in the phase space at a given \(t\) time. More exactly, it shows the probability that the first particle is located somewhere within the infinitesimal small volume that resides between \(\vec{x}_1\) and \(\vec{x}_1 + d\vec{x}\), and its momentum resides in the volume of \(\vec{p}_1\) to \(\vec{p}_1 + d\vec{p}\), while the second particle has its coordinate value in the volume of \(\vec{x}_2\) to \(\vec{x}_2 + d\vec{x}\), etc. Alas, instead of solving for the motion of the individual particles, the problem is generalised to solving for the
evolution of this N-particle probability density function. The change in $F_N$, or the total derivative at a given point can be expressed as:

$$dF_N = \frac{\partial F_N}{\partial t} dt + \sum_{i=1}^{N} \frac{\partial F_N}{\partial \mathbf{x}_i} d\mathbf{x}_i + \sum_{i=1}^{N} \frac{\partial F_N}{\partial \mathbf{p}_i} d\mathbf{p}_i.$$ (3.4)

During the evolution of the particle system, the value of $F_N$ does not change in time along trajectories in phase space, thus if we write up the total change along the trajectory of the current system, the right hand side becomes zero: $dF_N = 0$. The result is called the Liouville equation:

$$\frac{\partial F_N}{\partial t} + \sum_{i=1}^{N} \left[ \frac{\partial F_N}{\partial \mathbf{x}_i} \frac{d\mathbf{x}_i}{dt} + \frac{\partial F_N}{\partial \mathbf{p}_i} \frac{d\mathbf{p}_i}{dt} \right] = 0.$$ (3.5)

This general equation does not contain any description of the inter-particle forces. They are usually inserted by replacing the $\frac{d\mathbf{p}_i}{dt}$ term with some model of the acting forces.

3.5.3 The Boltzmann equation

Now we have a single partial differential equation describing the evolution of our particle system. Though this simplifies the problem a little, it is still too complex for any technical purposes due to the large number of degrees of freedom. To reduce the complexity, first we need to decouple the N-particle distribution function to an ensemble of 1-particle distribution functions. The single particle distribution function represents the probability for a single particle of being in the volume that resides between $\mathbf{x}_1$ and $\mathbf{x}_1 + d\mathbf{x}$ with its momentum being between $\mathbf{p}_1$ and $\mathbf{p}_1 + d\mathbf{p}$ and it has to include all the possible configurations for the remaining particles. It can be calculated then from the N-particle distribution function as follows:

$$F_1(\mathbf{x}_1, \mathbf{p}_1) = \int F_N d\mathbf{x}_2 d\mathbf{p}_2 \ldots d\mathbf{x}_N d\mathbf{p}_N.$$ (3.6)

The governing equation for $F_1$ can be acquired from the Liouville equation by performing this integration in it. As an example for the detailed steps see the aforementioned work of Harris (2004). After some calculation this integration yields the Bogoliubov-Born-Green-Kirkwood-Yvon hierarchy equation (also called BBGKY hierarchy for short):

$$\frac{\partial F_1}{\partial t} + \frac{\mathbf{p}_1}{m} \cdot \frac{\partial F_1}{\partial \mathbf{x}_1} = (N - 1) \int \frac{\partial \phi_{12}}{\partial \mathbf{x}_1} \cdot \frac{\partial F_2}{\partial \mathbf{p}_1} d\mathbf{x}_2,$$ (3.7)

where $\phi_{12}$ stands for the inter-particle forces. The main difficulty with this equation arises from the fact that it is not closed. The term $F_2$ on the right side stands for the two-particle distribution function, which in turn depends
on the three-particle distribution function that depends on the four-particle one and so on. Hence the name hierarchy equation.

In the case of no interactions between particles, the equation takes the simple form:

$$\frac{\partial F_1}{\partial t} + \frac{\vec{p}_1}{m} \cdot \frac{\partial F_1}{\partial \vec{x}_1} = 0. \quad (3.8)$$

That is, the right hand side of the BBGKY equation is simply zero. A more intuitive explanation for this is that in the absence of particle collisions the rate of change in the number of particles inside the phase space volume of $\delta \vec{x} \delta \vec{c}$ is zero, where $\vec{c}$ denotes the $\frac{\vec{p}}{m}$ particle velocity. In the case of existing collisions this rate is no longer zero. The $\delta n$ change in the number of particles residing in the phase volume of $\delta \vec{x} \delta \vec{c}$ during a $\delta t$ time period can be expressed as:

$$\left(\frac{\partial F_1}{\partial t} + \frac{\vec{p}_1}{m} \cdot \frac{\partial F_1}{\partial \vec{x}_1}\right) \delta \vec{x} \delta \vec{c} \delta t = \delta n. \quad (3.9)$$

From here we need to extract the information of the number of particles leaving the phase volume and the number of particles entering it due to collisions from the two-particle distribution function. The detailed steps can be found for example in the work of Liboff (1991) and some additional information in the book of Guo and Shu (2013). The logic behind those steps is as follows: we will use the assumption of the particles being hard spheres of equal diameter ($\sigma$), and then an interaction cross-section area can be defined based on the particle position and radius. For every possible pair collision for every $\delta t$ time period, based on the interaction cross-section area and the relative velocities of the particles, a small interaction volume for the collision event can be calculated. To handle the collision events that drove out particles from the phase volume similarly to those that bump them in, we will denote the post-collision location and velocity with $\tilde{\vec{x}}, \tilde{\vec{c}}$, respectively, and define the inverse collision that drives the particle pair back from the post-collision phase point to the pre-collision one. With these definitions, Eq. 3.9 takes the following form:

$$\frac{\partial F_1}{\partial t} + \frac{\vec{p}_1}{m} \cdot \frac{\partial F_1}{\partial \vec{x}_1} = \int \left[ F_2(\tilde{\vec{x}}_1, \tilde{\vec{c}}_1, \tilde{\vec{x}}_2, \tilde{\vec{c}}_2) - F_2(\vec{x}_1, \vec{c}_1, \vec{x}_2, \vec{c}_2) \right] d\tilde{\vec{c}}_1 \parallel \tilde{\vec{c}}_1 - \vec{c}_2\parallel \sigma d\Omega, \quad (3.10)$$

where the term $\sigma d\Omega$ stands for the interaction cross-section area, thus $\parallel \vec{c}_1 - \vec{c}_2\parallel d\text{div}d\Omega$ is the interaction volume. This form is no longer a hierarchy equation, though the right hand side still depends on the two-particle distribution function. A few more assumptions need to be made to close the equation. First, we can treat the $F_2$ function as homogeneous over the physical spatial domain where the collision events can occur. In this way, the
two-particle distribution can be expressed simply as the function of velocities:

\[ F_2(\vec{x}_1, \vec{c}_1, \vec{x}_2, \vec{c}_2) = (\vec{c}_1, \vec{c}_2). \] (3.11)

The last step here is to assume *molecular chaos*. That states the uncorrelated nature of the particle trajectories both before and after the collision. With this assumption, the two-particle distribution decouples to the product of two single-particle distributions:

\[ F_2(\vec{c}_1, \vec{c}_2) = F_1(\vec{c}_1)F_1(\vec{c}_2). \] (3.12)

Now if we rewrite Eq. 3.10 in terms of single-particle mass density \( f \) (as an alternative of the distribution function) with the previous assumptions, it will yield the *Boltzmann equation*:

\[ \frac{\partial f}{\partial t} + \vec{c}_1 \cdot \nabla f = \frac{1}{m} \int \sigma \| \vec{c}_1 - \vec{c}_2 \| \left[ f(\tilde{\vec{c}}_1)f(\tilde{\vec{c}}_2) - f(\vec{c}_1)f(\vec{c}_2) \right] d\Omega d\vec{c}_2, \] (3.13)

where \( f = NmF_1 \).

The Boltzmann equation, as a governing equation for the single-particle mass density function, thus can be originated from the BBGKY equation with the outlined steps. The major assumptions made along the construction are:

- Only binary collisions are considered. This restricts the equation (at least in its current form) to dilute gases where multi-particle collisions are rare.

- Particles are represented as hard spheres with a small and uniform radius. Collisions are modelled as instantaneous events with no change in the total kinetic energy of the participating particles. (No interaction with longer range than the local collision was defined).

- The particles move along a rectilinear path (this assumption is used to calculate the interaction cross-section and volume).

- \( F_2 \) is assumed to be homogeneous over the collision space domain. The phase space of the distribution function then can be reduced to include only velocities.

- The trajectories of the colliding particles are uncorrelated, thus \( F_2 \) can be decoupled to the product of two single-particle distributions.

Finally, for the dynamics of a single particle in the system the Boltzmann equation has a degrees of freedom of only 6 instead of 6\( N \).
3.5.4 Connection with macroscopic quantities

The macroscopic properties of the particle ensemble emerge from this description as moments of the local single-particle mass density function. The zeroth moment is simply the integral of \( f(\mathbf{x}, \mathbf{c}, t) \) over the possible velocities. The physical meaning of \( f \) is the expected mass density of particles at \( \mathbf{x} \) moving with velocity \( \mathbf{c} \). Integration of this function over the velocity domain at a given \( \mathbf{x} \) location yields the amount of total mass density located at that spatial point:

\[
\rho = \int f(\mathbf{x}, t) d\mathbf{c}. \tag{3.14}
\]

Along the same logic, further moments can be introduced. The first moment yields the average of the particle velocities, that is, the emerging macroscopic velocity:

\[
\rho \mathbf{u} = \int \mathbf{c} f(\mathbf{x}, t) d\mathbf{c}. \tag{3.15}
\]

The second moment gives the expected total kinetic energy of the particles. As the movement of a single particle with unit mass produces \( \frac{\|\mathbf{c}\|^2}{2} \) kinetic energy, the expression for the energy summed over the velocity space can be written as:

\[
E = \frac{1}{\rho} \int \frac{\|\mathbf{c}\|^2}{2} f d\mathbf{c}. \tag{3.16}
\]

This total energy is composed of the kinetic energy of the bulk fluid motion and the energy of the motion of the individual particles within the flowing fluid. To separate them, a peculiar velocity can be defined: \( \mathbf{c}_0 = \mathbf{c} - \mathbf{u} \). The stress tensor can be formalised using this velocity if we express the momentum balance upon an infinitesimal surface around \( \mathbf{x} \) that moves with the fluid. Since it moves together with the flow, the momentum flux through the surface is generated only by the peculiar velocity:

\[
\hat{P} = \int \mathbf{c}_0 \mathbf{c}_0 f d\mathbf{c}. \tag{3.17}
\]

From this definition it follows that this stress tensor is symmetric ( \( \hat{P}_{\alpha\beta} = \hat{P}_{\beta\alpha} \)). Furthermore, the pressure can be defined using its trace:

\[
p = \frac{1}{3} \int \|\mathbf{c}_0\|^2 f d\mathbf{c}. \tag{3.18}
\]

The heat flux can be defined along the same logic as:

\[
\hat{Q} = \int \frac{\|\mathbf{c}_0\|^2}{2} \mathbf{c}_0 f d\mathbf{c}. \tag{3.19}
\]
The collision integral (technically the right hand side of Eq. 3.13) has several symmetries. Using these symmetry properties and Boltzmann’s $\mathcal{H}$–theorem, it can be shown (Guo and Shu, 2013) that it obeys such laws that the macroscopic quantities derived from it obey the macroscopic conservation laws defined in Section 2.4.

Another emerging property from the theorem is that in case of local equilibrium the solution for $f$ follows the Maxwell-Boltzmann distribution:

$$f_{eq} = \left( \frac{\rho}{(2\pi RT)^{2/3}} \right) e^{-\frac{(\vec{c} - \vec{u})^2}{2RT}},$$

(3.20)

where $T$ is the temperature and $R$ is the universal gas constant. This is the most probable state of the system, thus successive collisions will drive the system towards this distribution. When this distribution is reached, the collision integral becomes zero and in the absence of external forces the system remains in equilibrium.
Part II

LBM models and implementations for haemodynamic investigations
The Boltzmann equation summarised in the previous chapter is an integro-differential equation. Its complexity originates from the collision integral residing on the right hand side in Eq. 3.13. In order to make this equation useful for practical purposes, this collision integral is often modelled with some simpler approximation that is capable of recovering the same macroscopic transport equations.

4.1 THE BGK APPROXIMATION

The theoretical basis of this model for simulating fluid flows was laid by Bhatnagar, Gross, and Krook (BGK) (Bhatnagar et al., 1954) by introducing their collision operator. The basic idea is that for flows near the equilibrium state the mass density distribution function can be approximated as the equilibrium state and the small deviation from the Maxwell-Boltzmann distribution:

\[ f = f^{eq} + \Delta f. \] (4.1)

With the help of this expression the collision integral can be simplified and Eq. 3.13 can be written as:

\[ \frac{\partial f}{\partial t} + \vec{c} \cdot \nabla f = -\omega [f - f^{eq}], \] (4.2)

where \( \omega \) is a collision frequency related to the collision cross-section area and the relative velocity of the colliding particles. The BGK approximation transforms the Boltzmann equation into a simple partial differential equation greatly reducing its complexity. This model is thoroughly reviewed in the work of Chen and Doolen (1998) and Succi (2001) as a numerical method for solving general fluid flow problems. The meaning of the right hand side is expressive: the collisions drive the system towards the equilibrium state with \( \omega \) frequency. At a first glance it might seem to be an over-simplification to introduce a linearised operator to model an obviously non-linear dependence but the dependence of \( f^{eq} \) on density, velocity, and temperature encompasses the non-linearity. Applying the Chapman-Enskog analysis as in Chapman and Cowling (1992) in the limit of long wavelengths (or low frequencies), the above-defined system can be related to the Navier-Stokes equation for
incompressible flows with an ideal equation of state: \( p = \rho c_s^2 \) and a numerical kinematic viscosity in the form of

\[
\nu = c_s^2 \left( \tau - \frac{1}{2} \right),
\]

(4.3)

where \( \tau = \frac{1}{\omega} \) and \( c_s \) is the numerical grid-dependent speed of sound.

### 4.2 Further Discretisations

As written before, the LBM originates from the lattice gas automata in which molecules had less degrees of freedom in the numerical sense. They are constrained to move along lattice sites on the shortest path. For a two-dimensional rectangular numerical grid it means five possible velocities, one pointing towards each side and one for staying in place. Combining this with the boolean state of particles (0 for a free lattice and 1 for an occupied one) presents several problems e.g., the violation of Galilean invariance and the appearance of strong numerical noise. To circumvent these limitations in the LBM, as I have discussed in the previous sections, the lattice states are described with single-particle collision functions instead of boolean numbers and the collision matrix of the lattice gas automata is replaced with some relaxation type process (BGK is an example but there are other ways to model the collision integral).

The proper discretisation is a bit more complicated than for the macroscopic conservation equations as the \( f \) distribution function is a function of not only time and space, but of velocity. The entire velocity space cannot be retained as it would increase the computational requirements to the level of being useless for practical purposes. Alas, it has to be discretised as well to reduce the computational complexity. Thus, similarly to the lattice gas automata, one should select a finite set of velocities and constrain the particles to travel only along those velocity vectors. Naturally, the selection of available velocities is not arbitrary. The lattice and the velocities together need to be symmetrical and contain a sufficient number of velocities, otherwise it is not possible to recover the correct transport equations as Pomeau and Frisch (1986) showed it for lattices used for LGAs.

Throughout the current work, two different numerical grids are used: the \( D_2Q_9 \) for two-dimensional investigations and the \( D_3Q_{19} \) for three-dimensional ones. The naming convention is as follows: the first part states the dimensionality of space and the second one states the number of possible velocities. These two grids are shown in Fig. 4.1.

For the following description I use the two dimensional grid for simplicity but the three-dimensional one carries analogous properties (apart from hav-
Figure 4.1: The two most common lattice Boltzmann numerical grid types with the set of selected velocities for the two-dimensional D2Q9 (left side) and three-dimensional D3Q19 (right side). These were used in this work.

The 9 velocity vectors for the two-dimensional case are:

\[
\vec{c}_i = \begin{cases} 
(0,0), & \text{if } i = 0, \\
(0,1), (1,0), (0,-1), (-1,0), & \text{for } i = 1,3,5,7, \\
(1,1), (-1,1), (-1,-1), (-1,1), & \text{for } i = 2,4,6,8,
\end{cases}
\] (4.4)

where the index of \( \vec{c}_i \) denotes the numbering for the discrete velocities from the set. Using this discretisation, the distribution function can be decoupled into 9 separate distribution functions that are no longer dependent on the velocity. Using the Cartesian coordinate system in which the D2Q9 grid has a uniform rectangular geometry, it is easy to see that the velocities noted in Eq. 4.4 are of different length. Those pointing to the corners of the lattice are longer than those pointing to the median of the lattice sides. To account for this, a weighting factor must be introduced and the relationship for the velocity-discrete distribution functions will take the following form:

\[
f_i(\vec{x},t) = w_i f(\vec{x}, \vec{c}_i, t),
\] (4.5)
where the weighting factor \( w_i \) is determined so that the integral terms in the momentum equations Eq. 3.14, 3.15, and 3.16 can be replaced by equivalent sums:

\[
\rho = \sum_{i=0}^{9} f_i, \quad (4.6)
\]

\[
\rho \vec{u} = \sum_{i=0}^{9} \vec{c}_i f_i, \quad (4.7)
\]

\[
\rho E = \sum_{i=0}^{9} \frac{\| \vec{c}_i \|^2}{2} f_i. \quad (4.8)
\]

We can decouple the expression of the equilibrium distribution (Eq. 3.20) similarly. This equation still contains an exponential function which is numerically expensive to compute, thus is computationally inefficient:

\[
f_{i}^{eq} = w_i \left( \frac{\rho}{(2\pi RT)^{2/3}} \right) e^{\left( \frac{-\|\vec{c}_i-\vec{u}\|^2}{2RT} \right)}. \quad (4.9)
\]

If we expand the parentheses in the argument of the exponential and then replace the resulting terms containing the macroscopic velocity (\( \vec{u} \)) with their Taylor series expansion about \( \vec{u} = 0 \), the following expression can be recovered:

\[
f_{i}^{eq} = W_i \rho \left[ 1 + \frac{\vec{c}_i \cdot \vec{u}}{RT} + \frac{(\vec{c}_i \cdot \vec{u})^2}{(RT)^2} - \frac{\| \vec{u} \|^2}{2RT} \right], \quad (4.10)
\]

where the new \( W_i \) weighting function incorporates all the remaining terms from the expansion that do not contain \( \vec{u} \):

\[
W_i = w_i \left( \frac{1}{(2\pi RT)^{2/3}} \right) e^{\left( \frac{-\|\vec{c}_i\|^2}{2RT} \right)}. \quad (4.11)
\]

After solving for the equality of the discretised momentum equations and the continuous ones, as I mentioned previously, \( W_i \) for the D2Q9 lattice will take the following values:

\[
W_i = \begin{cases} 
\frac{4}{9}, & \text{if } i = 0, \\
\frac{1}{3}, & \text{for } i = 1, 3, 5, 7, \\
\frac{1}{36}, & \text{for } i = 2, 4, 6, 8.
\end{cases} \quad (4.12)
\]

One further implication of the calculation of the lattice weights from the Taylor expansion is the fixed value of \( RT \) at:

\[
RT = \frac{1}{3}. \quad (4.13)
\]
The value of the gas is constant thus fixes the value for temperature as well for D2Q9. Since the temperature is fixed by the ideal-gas law \( p = \rho RT \), it follows that the value of \( \frac{T}{\rho} \) is also constant. Combining this with Eq. 2.6 yields the value for the numerical speed of sound on the D2Q9 lattice:

\[
cs = \sqrt{\frac{p}{\rho}} = \sqrt{\frac{1}{3}}.
\]  

(4.14)

It completes the specification for the ingredients of the discrete Boltzmann equation (DBE):

\[
\frac{\partial f_i}{\partial t} + \mathbf{c}_i \cdot \nabla f_i = -\omega(f_i - f_i^{eq}),
\]

(4.15)

which describes a set of first-order partial differential equations. One for every discrete velocity. The left hand side can be written using the substantial derivative formulation for the single-particle distribution:

\[
\frac{D f_i}{Dt} = -\omega(f_i - f_i^{eq}).
\]

(4.16)

This substantial derivative can be approximated with a first-order finite difference scheme:

\[
\frac{f_i(\mathbf{x} + \mathbf{c}_i \Delta t, t + \Delta t)}{\Delta t} = -\omega(f_i - f_i^{eq}).
\]

(4.17)

The right hand side is only evaluated at time \( t \), thus this is a fully explicit method. It is usually rearranged to isolate the term that contains \( t + \Delta t \):

\[
f_i(\mathbf{x} + \mathbf{c}_i \Delta t, t + \Delta t) = (1 - \omega')f_i + \omega' f_i^{eq},
\]

(4.18)

where \( \omega' = \Delta t \omega \). This form is most usually referred to as lattice Boltzmann equation (LBE). In most practical cases \( \Delta t \) is chosen to be 1 in lattice units. When this equation is implemented on a uniform square (or in 3D cubic) lattice, in which \( \Delta \mathbf{x}_i = \mathbf{c}_i \Delta t \), then there is no need for interpolation and the method is exactly conservative in the terms of macroscopic variables.

The LBE formalism shown here is based on the BGK approximation for the collision operator where \( \omega \) is a single scalar value, hence all the moments relax with the same frequency. The suitability of this model for fluid dynamics calculations is demonstrated in the work of Y. H. Qian et al. (1992).

In the next sections I shall briefly describe several variants of the LBM theory where different collision approximations are used and whose results I utilised in the validation work described later in the current part of this dissertation. Though each model is only briefly described, I give the references where the detailed derivations can be found.
4.3 **INCOMPRESSIBLE BGK MODEL**

In the classical BGK approach, the equilibrium term is multiplied by the density (see Eq. 4.10). The incompressible BGK model (IncBGK) modifies this by replacing the \( \rho \) term as shown in He and Luo (1997) with the term \( \rho = \rho_0 + \delta \rho \), where \( \rho_0 \) is a constant density value and \( \delta \rho \) is a small fluctuation around it in the order of \( O(M^2) \), \( M \) denoting the Mach number. If we neglect the terms which are \( O(M^3) \) or higher, the equilibrium distribution function will become:

\[
\begin{align*}
    f_{i}^{eq}(\vec{x},t) &= w_i \left\{ \rho + \rho_0 \left[ 3(\vec{c}_i \cdot \vec{u}) + \frac{9}{2}(\vec{c}_i \cdot \vec{u})^2 - \frac{3}{2} \vec{u}^2 \right] \right\},
\end{align*}
\]

where all the terms containing \( \vec{u} \) are multiplied by the constant \( \rho_0 \) rather than \( \rho \). The velocity is defined to be proportional to momentum, thus this model is computationally cheaper than the simple BGK method as we can avoid the division by \( \rho \) in Equation 4.7. It also reduces the compressibility errors introduced by the quasi-incompressible nature of LBM.

4.4 **MULTIPLE RELAXATION TIME MODEL**

The collision operator acts on the off-equilibrium part of the distribution. In the BGK model, all the lattice node populations relax with the same relaxation time (\( \tau \)). The multiple relaxation time model (MRT) is an extension of this model in a way that every mode can have a different relaxation speed through the usage of a generalised collision matrix. This model was introduced by D’Humières in 1992 and reviewed in D’Humières et al. (2002). If the collision operator matrix can be diagonalised, the physically meaningless ghost modes can be fine-tuned to achieve greater numerical stability. These ghost modes can also be used for modifying the physics of the simulated fluid flow, e.g., for free surface flows (Ginzburg and Steiner, 2003) or thermal flows with improved stability (McNamara et al., 1995). In our simulation, they were tuned for increased stability as suggested by D’Humières using linear stability analysis in their above-mentioned work.

4.5 **ENTROPIC LATTICE BOLTZMANN MODEL**

In this model, we define an entropy function upon the \( f_i \) distribution function as described in Boghosian et al. (2003). Another good description of the entropic lattice Boltzmann model (ELBM) can be found in Ansumali (2004). This model maintains higher numerical stability by respecting the second law of thermodynamics (through Boltzmann’s \( H \)-theorem). Entropy in this model can be expressed as: \( H(f) = \sum_i f_i \ln(f_i/W_i) \). We require the \( f_i^{eq} \) function to
minimise the entropy function under the restriction of local conservation of mass and momentum. The dynamics of the populations can be written as:

\[ f_i(\vec{x} + \vec{c}_i, t + 1) = f_i(\vec{x}, t) + \alpha \beta (f^{eq}_i(\vec{x}, t) - f_i(\vec{x}, t)), \]  

(4.20)

where \( \beta \) is related to the kinematic viscosity and \( \alpha \) can be used for maintaining the entropic balance and its value is defined as the root of the following equation:

\[ H(f + \alpha (f^{eq} - f)) = H(f). \]  

(4.21)

Solving this computationally demanding algebraic equation can greatly increase runtime, even with the usually employed iterative solutions. In my case the Newton-Raphson iteration method was used. Approximating the root of this equation provides an upper limit for the maximum possible change in distribution function populations during the relaxation step while not letting the entropy function value decrease. This prevents the \( f_i \) population values from adopting a negative value, thereby ensuring numerical stability in the absence of boundary conditions.

### 4.6 Regularised Model

We can extend the classical BGK dynamics with a relatively simple pre-collision procedure as shown in Latt and Chopard (2006) which can increase numerical stability at the expense of a little computational overhead. Following the Chapman-Enskog analysis (Chapman and Cowling, 1992), it can be concluded that for the simple BGK model in the hydrodynamic limit only the first three moments of the distribution function change the macroscopic behaviour of the fluid. From this it follows that we can recompute the populations based only on these three moments, as in Latt and Chopard (2005) in such a way that in the \( f_i = f_i^{(0)} + \epsilon f_i^{(1)} + \epsilon^2 f_i^{(2)} + \ldots \) expansion we try to cancel out the \( O(\epsilon^2) \) or higher order terms. If we execute a BGK collision after such a regularisation step, it will have a similar effect to applying a diagonalised linear collision operator, thus this model shows strong similarities with the MRT model.
The medical and physical concepts of the investigation were introduced up to this point. The last missing topic to complete the picture originates from the domain of computer science. In several cases some form of complexity of the emerging physical problems pose high computational demands that the current level of technology is not yet ready to fulfil directly. One of the aims of computer science is to tackle this problem by adapting the solution steps to match our current technologies in such a way that it does not change the fundamental theory of the solution method. Sometimes this can be achieved with the knowledge of the important limiting technological factors and the usage of pure mathematical transformations, other times by the intelligent use of assumptions that are valid for the modelled physical problem. These are details of the numerical solvers that are independent of the bases of the theory but are still parts of the implementation and can influence both the computational efficiency and numerical accuracy. In the following I will outline several such key points that are not strictly parts of the fundamental theory, yet they are required for the realisation of a complete solution.

5.1 Initial Values

The macroscopic description of the state of the fluid requires at least the value for density and the components of velocity at every lattice site. In the formalism of LBM, this is not necessarily adequate as it does not define a distribution function uniquely (namely, it only fixes the first two moments but provides no information for the higher order ones). This might seem non-physical at a first glance. If this method really tends to the same solution as the Navier-Stokes equations, it raises the question why it requires more information for a given state. The answer is simple: the method stores more information than needed for the recovery of the main macroscopic quantities. The computational efficiency of LBM originates from this higher dimensionality of the $f$ function. The increased information density at lattice sites compared to the purely macroscopic description also means that more information is accessible at every spatial location. Further quantities (such as the stress tensor) can be computed from on-site information as opposed to the methods operating with macroscopic description where they would require some form of spatial derivatives, thus information from neighbouring numerical nodes.
A usually employed method to define initial values is to calculate the $f_{\text{eq}}$ equilibrium distribution values from the given macroscopic velocities and densities and use them as initial values for the $f$ values. If the macroscopic velocities are all 0 and the simulation is a steady state one, then this initial value generation is correct. However, if the fluid has some velocity, even if it is a steady flow, the higher order moments are not recovered correctly by the use of the equilibrium distribution alone (as even in steady state the emerging stress is not necessarily zero everywhere in the simulation domain). In the current work, for every simulation I have run I used the following method:

- Calculate the $f_{\text{eq}}$ values from the macroscopic initial values.
- Use them as initial conditions for $f$.
- Run a steady state simulation with the prescribed macroscopic boundary values applied at $t = 0$.
- Run the computation until all the $f$ values converge.
- Use these $f$ values as the initial conditions for further transient simulations.

For a transient simulation, it increases the numerical stability as there will be no relevant initial condition artifacts present at the beginning of the transient phase.

5.2 THE SOLVER ALGORITHM

The main workhorse of the LBM method, the solution of the lattice Boltzmann equation (Eq. 4.18) can be separated into two distinct steps from a numerical point of view: the collision and the streaming steps.

**The Collision** phase is technically the execution of the collision integral for the discretised case. This part presents perfect parallelism as all the computations are local since every needed information is available on-site. This is also the computationally heavier phase, thus parallelism can be very useful here.

**The Streaming** step moves the collided components of the density distribution vector towards the neighbouring cells. The computational performance of this phase is usually memory-bandwidth limited as its main objective is to move values around.

These two steps can be thought of roughly as the components of a decoupled diffusion-advection method. A complete implementation of the LBM
5.2 THE SOLVER ALGORITHM | 39

method, however, might incorporate some additional steps such as the initialisation of the distribution function values and the handling of the boundary conditions. For an overview of the usual steps of an LBM solver please see Fig. 5.1.

![Flowchart diagram](image-url)

Figure 5.1: The typical layout for an LBM solver.
5.3 BOUNDARY CONDITIONS

One of the difficulties arising when working with the lattice Boltzmann method is the construction of appropriate boundary conditions. The reason for this is simple: our working variable is the mass distribution function instead of the macroscopic variables (such as velocity, pressure, density, etc.). Henceforth, we have to prescribe the boundary conditions for \( f \) in such a way that they recover the proper boundary behaviour for the derived macroscopic variables. Since the early 90s, a great amount of work has been devoted to investigate the non-periodic boundary conditions. The first practically useful boundary models were investigated by Ziegler (1993) and Skordos (1993). A simple no-slip rigid boundary wall model was introduced by Inamuro et al. (1995) and several later improvements were presented by Ginzbourg and d’Humières (1996); Maier et al. (1996); Noble, Chen, et al. (1995). Probably the biggest step forward in this topic was the work by Zou and He (1997) where they constructed a Neumann and a Dirichlet boundary condition scheme that is local, simple to implement, and can be second-order accurate under appropriate conditions.

Though my results presented in the current text do not explicitly build on them, more recent further boundary condition schemes can be found in the works of Ansumali and Karlin (2002); Yu et al. (2003); X. Zhang et al. (2002). In general, one of the most appealing aspect of LBM is the possibility to incorporate complex boundaries that can even be coupled with other physical processes (even with ones from different length-scales). The boundary conditions used in the concerned implementations are the bounce back schemes for the rigid no-slip walls and the Zou-He models for velocity and pressure boundaries. Their working mechanics and implementation details are outlined in the next sections.

5.3.1 No-slip boundary condition

In standard macroscopic CFD methods the no-slip wall is often modelled by imposing zero velocity on the boundary nodes. In LBM, more information is required to reconstruct the higher order moments. As we have a mesoscopic method, one possible way is to check what the particles do according to their kinematic description. They technically bounce off from the rigid surface. Indeed, the method adopted for wall boundaries in the presented simulations is called bounce back scheme. Figure 5.2 shows a chosen boundary case.

The density values along velocity 6, 7, 8 move outwards from the flow domain and the values to enter along the velocities 2, 3, 4 are not known for the next streaming step. The idea is to bounce the outgoing distribution values back in their opposite directions without any collision procedure:

\[
f_i'(\mathbf{x}, t + \Delta t) = f_i(\mathbf{x}, t),
\]  

(5.1)
where $\vec{c}_i' = -\vec{c}_i$. That is, the density incoming along velocity 6 is going to be reflected along velocity 2 according to Fig. 5.2. This technique obviously conserves the variables and a free-slip alternative is easy to construct: instead of changing to the opposite directions, densities are reflected specularly along the wall. Which means, that the density incoming along velocity 6 is going to be reflected along velocity 4. If we let the fluid propagate inside the solid lattices where they are reflected back and propagated back to the fluid domain in the next streaming step, then the fluid will behave as if there were a solid wall surface halfway between the last fluid lattice and the solid lattice. This version of the scheme is called half-way bounce-back. For an arbitrary curved surface this method reduces to first-order accuracy but it has been shown that for flows that are aligned in parallel with the wall this scheme can be second-order accurate (He, Zou, et al., 1997). Since we are reflecting back all the incoming densities (thus the momentum as well), the macroscopic velocity (the directional sum of the densities) is zero at the wall. This works very well for stationary walls but it can also be modified for moving walls by accounting for the momentum contribution from the wall movement.

5.3.2 Neumann and Dirichlet boundary conditions

For cases when constraints on the velocity values are required on a boundary site, I apply the von Neumann formalism of the Zou-He (Zou and He, 1997) boundary condition. Let me take the case depicted in Fig. 5.2 again. Now the boundary is a velocity boundary site with parallel inflow of macroscopic velocity $\vec{u}_0 = (u_0, 0)$. The macroscopic density (pressure) needs to be computed. The density values on nodes 0, 1, 2, 3, 4, and 5 are known as they arrive from in-
side the fluid domain. Therefore, we need to solve for $\rho, f_6, f_7, f_8$, and for that we shall need four equations. The expression that connects the distribution function values with the macroscopic density is one (Eq. 4.6). The connection for macroscopic velocity vector is another two (Eq. 4.7). The fourth equation comes from the assumption that the bounce-back condition holds for the off-equilibrium parts in the direction normal to the boundary:

$$f_3 - f_3^{eq} = f_7 - f_7^{eq}. \quad (5.2)$$

If the macroscopic pressure is given on the boundary (Dirichlet case), the same four equations can be used. One further required piece of information is the direction of the inflow velocity, though it is usually taken to be perpendicular to the inlet surface.

5.3.3 Prescribing the correct velocity profile

The prescribed shape of the velocity profile on the inlet can have significant influence on the resulting flow field. To recover the correct shape in fluid simulation in vessel geometries it is common to create an elongated artificial lead-in vessel section in which the accurate velocity profile can develop. For LBM simulations that use direct addressing (for more explanation on this see the next section), this should be avoided as it can increase the prismatic outer boundary of the simulation domain, thus increasing heavily the number of numerical cells to store. To circumvent this problem, in my simulations the inlet velocity profiles are approximated using two components:

- the direction of the inlet velocity vector is chosen so that it is parallel with the beginning section of the vessel centreline and it points from the inlet surface inwards the fluid;
- the length of the vector is calculated using the sum of the Womersley components as explained below.

The inlet function is given as a periodic function that bears the main physiological properties of a volume flow function generated by the heart during a cardiac cycle, therefore it can be expanded using the Fourier series. The periodic function can be then written as a sum of modes as:

$$Q(t) = \sum_{n=0}^{N} q_n e^{i\nu n t}. \quad (5.3)$$

The Womersley formula (Womersley, 1955) describes the velocity distribution inside a rigid, smooth cylindrical pipe flow driven by a periodically pulsating pressure gradient. Combining these, the Womersley velocity profile of every Fourier mode can be calculated and their linear sum produces the answer of
the fluid system for the original driving function as follows (Cebral, Castro, Appanaboyina, et al., 2005):

\[
  u(r,t) = \frac{2q_0}{\pi R^2} \left[ 1 - \left( \frac{r}{R} \right)^2 \right] + \sum_{n=1}^{N} \frac{q_n}{\pi R^2} \left[ 1 - \frac{J_0(\beta_n) J_0(\beta_n)}{\beta_n J_0(\beta_n)} \right] \frac{\alpha n \omega t}{2}, \tag{5.4}
\]

where

\[
  \beta_n = i^\frac{3}{2} \alpha_n = i^\frac{3}{2} R \sqrt{\frac{n\omega}{v}}. \tag{5.5}
\]

In this expression \( u \) is the length of the axial velocity vector at a given \( r \) radius, \( R \) is the radius of the pipe, \( J_0 \) and \( J_1 \) denotes the zeroth and first order Bessel functions, \( v \) is the kinematic viscosity, \( \omega \) is the angular frequency, \( \alpha \) is the Womersley number, and finally \( n \) denotes the ordinal number of the harmonic component from the Fourier expansion.

5.4 PARALLELISM

Computational performance is usually of great interest in every numerical model since it limits the level of attainable detail and complexity. Each modern numerical simulation has to incorporate some level of parallelism and concurrency. The reason for this lies in the modern technological solutions of the devices used for carrying out the computation. According to Moore’s law, the number of transistors in the central processing unit (CPU) of PCs doubles every 18 months. This law held up very well from 1965 till around 2013 but recently a different trend is beginning to form. Currently, the fabrication density of the transistors are defined with the thickness of the wafer, the semiconductor material used as the base layer. In the current technology, its thickness is about 14 nm and Intel predicts to be capable of going down to about 5 nm. At that level, a single transistor consists of about 7 atoms, henceforth the classical electrodynamics does not hold there any more and several quantum mechanical side effects arise. This leads to a slow-down of the increase of the transistor numbers. Another factor, the working frequency of the CPUs also seems to approach its limit with the currently common highest value of about 4 Ghz. Therefore, an alternative way is necessary to keep improving the computational performance.

For numerous applications this way seems to be the execution of the computational task in a parallel manner: even if it is difficult to increase the performance of a single CPU, one can always utilise more in parallel if the formalism of the problem permits so. Ahmdal’s law describes the run-time behaviour of a parallel algorithm.
Ahmedal’s Law  If we have $n$ serial processing units that can work concurrently to each other and $B \in [0; 1]$ denotes the portion of the algorithm that is strictly serial, then the time $T(n)$ it takes the algorithm to finish with the parallel run on $n$ computing units is:

$$T(n) = T(1) \left( B + \frac{1}{n} (1 - B) \right).$$

(5.6)

The theoretical improvement in the execution time $S(n)$ is then written as:

$$S(n) = \frac{T(1)}{T(n)} = \frac{1}{B + \frac{1}{n} (1 - B)}.$$

(5.7)

In practical uses the attainable runtime improvement in parallel applications is usually limited by the portion of strictly serial components. For instance, if for a given problem $B = 0.1$, then even if we have an infinite number of processing units, the reachable maximum speed-up is $S_{\text{max}} = 10$. This is why the LBM algorithm is called memory-bandwidth limited as even if the collision step is extremely parallel, the streaming step (whose run-time relies on memory-bandwidth) contains inherently serial components due to current memory access implementations.

5.5 Implementations

Finally, to conclude this chapter, I shall introduce the three different solver implementations used during the investigation of the problems in the current dissertation. Out of them two are built using open source components while the third one is programmed by myself completely from scratch. Though each of these solvers targets the recovery of the flow field in a given complex geometry, their purpose and their technical realisation slightly differ from each other. In the following I outline their main characteristic features and important implementation details.

5.5.1 2D CPU implementation

This is a two-dimensional implementation that serves as a sandbox for trying new methods and implementations, for this reason it is designed with flexibility as a first priority. It has been developed entirely by myself in C++ and Python and consists of about five thousand lines of source code.

Short Overview  In this context, flexibility means that every numerical cell carries its own complete dynamic description. With the help of this feature changing the geometry during a simulation is possible by simply switching the descriptor of that numerical cell from a fluid node to a solid one only
undergoing bounce-back. Two further implications are the slightly increased memory requirement and the increased parallelism (there is no common data structure that needs to be accessed from separate threads).

**Propagation and Addressing** It uses an AA type propagation just as the other two implementations. In general, Lattice Boltzmann implementations can be distinguished by the exact realisation of the propagation method. For the simplest implementations it is common to use two identical memory spaces to store the values of the distribution function in every spatial point. One of them (memory buffer A) is the active one for reading during one cycle of the algorithm and the computation stores the new values into the other location (memory buffer B). At the end, these two memory locations are simply swapped. Thus, it is easily ensured that no required value is overwritten during the propagation. These kinds of methods are usually called AB type propagations. However, the downside is that the memory requirement is doubled. If the memory is more limited, in-place streaming operation is possible. The most common version is denoted as AA type propagation. A rather complete review of the possible propagation steps can be found in the work of Wittmann et al. (2013) along with their required memory sizes and access characteristics.

I address the memory locations of the lattices directly. It means that the whole prismatic shaped domain is stored in memory during the computation, even those lattices that are not participating in the simulation. The advantage is the lower memory-bandwidth requirement as I can directly access values. The drawback is the larger amount of memory required as the vessel geometries often take up less than 60% of the prismatic computational volume. However, for two dimensions the memory requirement is not really an issue. Indirect addressing can reduce the required memory for the price that it also increases the computation time since for every lattice access one needs to dereference an additional pointer.

**Parallelism** This implementation exploits shared memory parallelism through OpenMPI, which means that the work of the collision computation can be divided between CPU cores in a single machine where each core has access to the same memory areas.

**Data Storage** As an input format I use simple image files where the colour of every pixel encodes the dynamics to be used on that site (depending on whether it is a fluid or a solid node, or some porous material, etc.). In this way, new simulations are easy to set up (or rather to paint). As an output

1 http://www.open-mpi.org/
format, all three implementations utilise the HDF5 \(^2\) compressed binary file format.

5.5.2 3D CPU implementation

**Short overview** This implementation is built upon the open-source Palabos\(^3\) software library that offers scalable data structures and several algorithms of LBM, such as the pre-written collision algorithms acting upon these data structures. The most attracting feature is the distributed realisation of those structures allowing to divide both the computational work and data storage among several computers connected together with a high bandwidth network. When the CFD application does not have any special requirements, developing a fluid solver with the help of this library is technically reduced to piping together given algorithm blocks and writing parallel functors to transform the data to the intended formats in C++.

**Propagation and addressing** It uses an AA type propagation and direct node addressing. The storage of unused domain parts is avoided by the usage of a multi-block lattice. That is, the whole simulation domain is divided into smaller prismatic volumes that can be thought of as separate LBM simulations and these block lattices can communicate with each other on the boundaries. Thus, neglecting an unused region is easy: its block lattice is simply excluded.

**Parallelism** Parallelism here is also facilitated by this multi-block structure. Since each block stands on its own as a simulation, each can be executed on a different machine. The only factor to pay attention to is the balance of computation and communication between the execution units of the blocks. The higher the number of blocks is, the more need for the communication between the blocks arises.

**Data storage** For file output at the end of the process it uses HDF5 format but for check-pointing it uses a parallel binary format since the different parts of the simulation domain reside on different computational nodes.

5.5.3 3D GPU implementation

In Section 5.2 I mentioned that the algorithm of the LBM presents two main phases of which one is memory-bandwidth while the other is computation performance bounded. Thus, to achieve high overall performance, both the

---

\(^2\) http://www.hdfgroup.org/HDF5/

\(^3\) http://www.palabos.org/
memory-bandwidth and the raw floating point computational power need to be as high as possible.

**Short Overview**  This implementation is built upon the open-source Sailfish library\(^4\), that is (similarly to Palabos), a collection of pre-made LBM algorithms. It is written using meta-programming. In short, this means that the Python language and one of its template modules is used for writing an application, which in turn at run-time writes and executes an optimised graphics processing unit (GPU) program (in my case in CUDA language). The resulting GPU implementation of the LBM has excellent run-time characteristics. In Chapter 6 I elaborate on the exact run-time values. The reason for the fast execution resides in the capabilities of the GPU hardware. Figure 5.3 shows the raw computation and memory-bandwidth performances for recent GPU cards in comparison with recent CPUs.

These good results are implications of the structure of the GPU hardware. Figure 5.4 shows a structural overview. It is immediately noticeable that the GPU contains a larger proportion of arithmetic units and in return offers less control logic units than the CPU. It can be said that the GPU is a lot faster in mechanical number crunching, and it is less agile than the CPU when it comes to logical branches in the code. Given that the problem fits the special bounds of the GPU hardware (and LBM does have such a formalism), very high execution performance can be achieved.

Based on the theoretical peak value of the memory-bandwidth, a performance speed-up of about ten times can be expected. In reality, the performance gap of the realised CPU and GPU implementations can even be wider because of the fact that the GPU exhibits multiple levels of parallelism. The actual worker units are organised into blocks, which is actually a two-dimensional layout and the GPU itself contains several of these blocks. Though each worker accesses the global video memory with the bandwidth noted in Fig. 5.3, the workers in the same block can communicate through a cache-like lower level of memory. The bandwidth of this shared memory can be two orders of magnitude larger than that of the global memory.

**Propagation and Addressing**  For the simulation I use indirect addressing and AA type propagation since the video memory is more limited than the memory of the host computer in which the video card is installed. For the same memory-conservative logic I use AA type propagation.

**Parallelism**  This implementation exploits the multiple level of parallelism available on modern GPU cards. It can also divide the work similarly to the multi-block technique of Palabos, though it is less refined here. These blocks

---

4 [http://sailfish.us.edu.pl/](http://sailfish.us.edu.pl/)
Figure 5.3: Top: The comparison of raw theoretical peak performance between several CPU and GPU products. 1 GFLOP ≈ 10^9 floating point operations. Bottom: The comparison of the available theoretical memory-bandwidth of several CPU and GPU products. (The CPU values noted here stand for server products and are significantly higher than those available to desktop computers. Image source: CUDA Documentation 6.5)
can be located on different video cards inside different hosting machines in which case the boundary data have to be moved to the memory of the host computer and have to be communicated between these host nodes using the ZeroMQ library\(^6\).

**Data Storage** No parallel checkpointing is available in this implementation, therefore all the data across the nodes must be transferred to the host memory of the master node before saving the state of the simulation. Unlike with the previous implementation, one implication is that the full size of the simulated model must fit in the memory of the master node.

Technically this solver is expected to yield the same results as the previously presented solver implementation. The fundamental difference between them lies in those implementation details which do not interfere with the results of the computation but do change the way that the numbers are handled, thus making the whole computation fit a particular hardware and its capabilities better. As an outcome, the results presented in chapter 6 hint a nearly two orders of magnitude reduction in the run-time characteristics. Though this investigation will require further work, such a performance improvement would enable new applications of CFD on desktop computers that were previously accessible only on supercomputers. As a side note, it is no coincidence that more commercial CFD solvers based on the lattice Boltzmann method appeared in the industry recently\(^7\), some of which are utilising GPU techniques to improve the performance.

---

\(^6\) http://zeromq.org/

\(^7\) PowerFlow - http://www.exa.com/,
XFlow - http://www.xflowcfd.com/,
ultraFluidX - http://www.fluidyna.com/content/ultrafluidx
Validation

Validation is a key point of every numerical computation that is expected to reflect any property of real physical processes. Even though there are numerous successful blood flow simulations known in the literature, there still exists a need to verify complex 3D fluid flow simulation results not only by means of grid convergence or comparisons to the results of other solvers, but by comparing them to the results of experiments carried out in the same complex geometry. To target this need, in the current chapter I investigate the accuracy of several lattice Boltzmann-based 3D unsteady fluid flow simulations in a real intracranial aneurysm geometry by comparing their results to both experimental results and to those of a well-known finite volume solver\textsuperscript{1}. The main LBM implementation I used for the validation was the one based on the Palabos library presented in section 5.5.2 since that implementation is the most suitable for high resolution investigations in a supercomputing environment. Also, it has the most complete set of collision operators already implemented. After I chose the most accurate modelling scheme from several runs, I also compared the result of that scheme using the GPU implementation presented in section 5.5.3. I should mention that the website of Palabos also showcases a comparison for a similar vessel computation between a simulation implemented using Palabos and another one carried out using CFX\textsuperscript{2} where grid convergence is tested against a Palabos simulation run on a very high resolution numerical grid without any measurement comparisons. My simulation results, beside the standard grid convergence, are also compared to the experimental particle image velocimetry (PIV) and laser Doppler anemometry (LDA) results that were carried out by Ugron et al. (2012) on a silicon phantom with the very same geometry. Both of these techniques are laser-optical measurement methods. While the LDA measures one velocity component in a single spatial point over an extended period of time, the PIV measures two components of an instantaneous velocity field in a complete plane, defined by pulsed laser light sheets. The accuracy of the LDA is higher but its usage for unsteady flows is more time-consuming. For the detailed description of both techniques see for instance the work of Z. Zhang et al. (1990). In the cited experimental measurement the dominant velocity component was measured along two different lines placed at hydrodynamically important locations using the LDA method (see Fig. 6.1). I also used these results to verify the accuracy of PIV results. The PIV method measured all

\textsuperscript{1} Ansys CFX v14.0 - www.ansys.com
\textsuperscript{2} http://www.palabos.org/academia/usecases/30-cfxbench
three components of the velocity in the whole fluid domain using two different setups. Furthermore, apart from the comparisons of simulations with measurements, I also carried out comparisons between the run-times and the memory consumptions of the different simulations.

6.1 Simulation Setup

The aneurysm geometry I used in this investigation has one inlet and three differently sized outlets and is the same as that in Ugron et al. (2012). In the simplified geometry shown in Figure 6.1, the LDA experimental measurement location is marked by a red dotted line while the green plane shows the position of the cross-section for the later comparison in Section 6.2.

![Figure 6.1: The simple outline of the geometry. Openings are numbered in the order of their diameters: 1 - inlet; 2, 3, 4 - outlets; 5 - plugged opening. The two different locations of the LDA measurements are marked with red dashed lines: a, b, while the cross-section plane for the comparison in Section 6.2 is marked by a green plane. The blue dots on the (a) LDA line inside the aneurysmal sac show the locations for time-dependent comparisons: one near the centre of the aneurysmal vortex (I) and one closer to the wall (II).](image-url)
The time-dependent flow rate values for the inlet and the two larger outlets from the PIV measurements are shown in Figure 6.2. These flow rate values follow the curve of an artificial inlet pulse wave signal that bears the properties of a real cardiac pressure wave on real aneurysm geometry. The cycle length of a typical heart-beat cycle is approximately one second. In the PIV and LDA measurements it was lengthened to 4.74 s due to technical considerations, which also means that the value of the Womersley number is about half of what can be measured in a real human artery of similar size. In my simulations this cycle length is used in order to remain consistent with the measurements. These flow rate values, or from the numerical point of view, velocity boundary conditions were prescribed at the given openings. On the last and smallest outlet, a constant pressure boundary condition was prescribed to avoid the simulation to be over-determined. The vessel wall was defined to be a rigid no-slip boundary in all of my simulations, since the experimental PIV and LDA measurements were also carried out in a rigid-wall model. I implemented it as a half-way bounce-back wall in the LBM simulations. The fluid was set to be a Newtonian fluid to remain consistent with the measurements. I also applied the same physical properties ($\mu = 10.17 \text{ mPas}$ dynamical viscosity and $\rho = 1140 \text{ kg m}^{-3}$ density) as in the experiments. At the inlet, the LBM simulations had a parabolic velocity profile that was set to be parallel with the starting section of the inlet tube. I have to point out that the more accurately precomputed velocity profile described in Section 5.3.3 was not applied during the validation simulations, for the lead-in section of the vessel was long enough for the LBM method to develop the proper velocity profile by the time it reached the aneurysmal neck. The CFX simulation also had a parabolic inlet velocity profile but these velocity vectors were perpendicular to the inlet boundary, and to the $Y-Z$ plane as well. The inlet section in the CFX simulation was elongated by a ten-diameter long straight pipe section and the inlet velocity profiles were prescribed at the beginning of this attached pipe. This way the appropriate Womersley inlet profile could develop along the attached section before reaching the region of interest. This difference was motivated by performance considerations as the finite volume simulations took place on an unstructured grid where the elongated tube did not increase the element count of the numerical grid significantly. As for the LBM simulations on the other hand, it was more convenient to pre-calculate the precise velocity directions in order not to increase the size of the simulation domain, which would in turn significantly increase the memory requirement of the pre-processing phase.

The size of the simulated domain was $90.3 \text{ mm} \times 111 \text{ mm} \times 91 \text{ mm}$. For the finite volume simulation the above-mentioned body-fitted unstructured volume mesh was used as in Ugron et al. (2012) while the lattice Boltzmann simulations were all run on simple Cartesian grids. The flow rate at the inlet starts from a value well above zero as shown in Figure 6.2. To start from
I simulated a stationary state beforehand with the flow-rates prescribed in the \( t = 0 \) instant. The transient phase then was initialised from these values. The run-time of this stationary phase is also included in all of the LBM run-time measurements. In all LBM simulations described in this chapter the magnitude of the maximum lattice velocity was set to \( u_{\text{lattice, max}} = 0.1 \) while the relaxation time was set to \( \tau = 0.506 \) (\( \Omega = 1.97628 \)). Though it approaches 0.5, this relaxation time yields stable simulations using double precision arithmetic during the computations. The numerical viscosity in lattice units then is \( \nu_{\text{LBM}} = 0.02 \).

### 6.1.1 Preparation of the geometry

Beside the PIV and the LDA experimental results I also possess the geometry in the form of a surface mesh. From this mesh voxelised geometries can be generated with an arbitrary resolution. I used the voxeliser routines of the CVMLCPP \(^3\) library for this purpose. The output of the voxelisation process is the three-dimensional boolean array \( \text{Fluids}(x, y, z) \) with \textit{true} values for cells inside the vessel walls and \textit{false} values for the remaining outside ones. From

---

\(^3\) http://tech.unige.ch/cvmlcpp/
this information one can generate wall cells quickly with simple boolean algebra:

\[
WallsFluids(x, y, z) := \left( 1 \bigvee_{i=-1}^{1} 1 \bigvee_{j=-1}^{1} 1 \bigvee_{k=-1}^{1} \text{Fluids}(x+i, y+j, z+k) \right),
\]

\[
Walls(x, y, z) := WallsFluids(x, y, z) \bigwedge \neg\text{Fluids}(x, y, z).
\]

Now the boolean array \( Walls(x, y, z) \) holds true values for cells neighbouring a fluid cell. Every other cell with \((x, y, z)\) coordinates that is not true either in \( Fluids(x, y, z) \) or in \( Walls(x, y, z) \) will not take part in the simulation. This way the wall generation can be implemented very efficiently for computers, even in a parallel fashion. For the boundary conditions on openings I have developed a small tool that takes the Cartesian grid geometry and some labelling information regarding the type of the boundary (e.g., Neumann or Dirichlet type pressure or velocity boundary). This labelling can be done in various segmentation utilities. In my case, the 3DSlicer was used for labelling the boundary voxels. Then the proper boundary conditions were generated on these labelled cells, ensuring that the in-flow velocity profile is parabolic and the vectors are parallel with the lead-in section of the vessel. As the vessel geometry only sparsely fills the simulation domain (depending on the voxel resolution it means a 12 ± 3% fill ratio), the whole domain can be partitioned into an ensemble of smaller cubic sub-domains leaving out those that did not contain any wall or fluid cells. The multi-block data-structure of Palabos is capable of that. This step can largely decrease the memory usage of the LBM simulations.

6.1.2 The finite volume simulation setup

I used Ansys CFX as a finite volume solver. The numerical grid for these simulations consisted of 811,000 elements. The same boundary conditions and fluid properties as with the LBM simulations were prescribed. Afterwards, I started the solver with the following settings: double precision number representation and high resolution (second order) advection scheme with a laminar fluid model.

6.2 COMPARISONS OF EXPERIMENTAL AND SIMULATION RESULTS

The PIV results were available for the whole simulation domain (Ugron et al., 2012). Figure 6.3 shows the characteristic cross-section of the bulge (denoted

3D Slicer 4.0.1 - http://www.slicer.org/
in Figure 6.1 by the green plane) at $t = 1.56 \text{s}$ at the systolic peak, where the maximum flow rates were achieved. The centre point and the orientation of the plane are chosen so that both the approximate centre of the aneurysmal sac and the centreline of the parent vessel in front of the sac lie in the plane. This reveals much of the characteristics of the emerging flow-field. The sawtooth-like edges of the single cross-section images show the voxelised nature of the computational domain. The coloring of the images is smoothed by the visualiser 5 using linear interpolation between all the nearest neighbour cells. This and the lower spatial resolution are the reasons why the incoming jet seems to have some discontinuity in the PIV result image.

![Figure 6.3](image_url)

**Figure 6.3:** Velocity magnitude contours in the different simulations at the given cross-section of the aneurysmal sac at the systolic peak.

I tested each of the above-introduced LBM models with the grid resolution denoted as medium resolution in Section 6.2.1, which has a reduced computational demand compared to the high resolution model. As for the numerical representation, in single precision only the MRT and the ELBM methods proved to be stable while the IncBGK and RLBM models diverged around the high flow rate instant. For this reason, I use the results of the double precision simulations to compare the LBM methods. The numerical stability can usually be improved by lowering the Mach number while maintaining a constant Reynolds number though that would in return increase the computational costs. According to Figure 6.3, the MRT and the ELBM show a better qualitative agreement with the PIV measurements than the other two models regarding near-wall velocities that are of increased importance in clinical

practice, for example due to wall shear stress calculations. The ELBM results, however, show a strongly decreasing velocity profile as we approach the centre of the aneurysmal sac. This is likely to be caused by the method how ELBM forces numerical stability upon the system by limiting the maximum change of the distribution function during the relaxation process. This can be thought of as a dampening process that actually takes out some kinetic energy from the system, causing slightly lower observable velocity magnitudes. A quantitative comparison is shown in Figure 6.4, where results obtained by all the applied LBM models are shown for the flow inside the aneurysmal sac.

![Figure 6.4: Velocity components in the aneurysmal sac according to the different LBM models.](image)

For further comparisons I selected the MRT model as the ELBM, which also produces qualitatively convincing results, underestimated the velocities in several regions of the fluid domain as it can also be observed in Figure 6.3 near the central region of the cross-section plane. The main reason for the difference between all these models might be that the dynamics of the fluid close to the bulge is rather complex and might encompass events from more than one length or time scale, thus those models allowing only for a single relaxation time without smoothing out the smaller scale dynamics are farther from the converged grid at the given resolution. By smoothing here I refer to the way the ELBM method suppresses the small flow instabilities by limiting the maximal temporal entropy change, however, this technically also means that some energy is taken out of the system, which explains the decreased velocity magnitudes for this method.
6.2.1 Grid convergence of the MRT model

For the grid convergence study I use three numerical meshes with different spatial resolutions, which is summarised in Table 6.1.

<table>
<thead>
<tr>
<th>Physical</th>
<th>Low</th>
<th>Medium</th>
<th>High</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>0.0903 m</td>
<td>301</td>
<td>450</td>
</tr>
<tr>
<td>Y</td>
<td>0.111 m</td>
<td>372</td>
<td>555</td>
</tr>
<tr>
<td>Z</td>
<td>0.091 m</td>
<td>302</td>
<td>458</td>
</tr>
<tr>
<td>len.</td>
<td>(3 \times 10^{-4} m)</td>
<td>(2 \times 10^{-4} m)</td>
<td>(1.7 \times 10^{-4} m)</td>
</tr>
</tbody>
</table>

Table 6.1: The first data column shows the physical extent of the simulation domain along the Cartesian axes. The following columns specify the number of lattices along the same axes and the physical length of a lattice side for each of the three resolutions.

The simulation results with the three different grid resolutions are shown in Figure 6.5 and Figure 6.6 at a high and a low flow rate instant. It should be noted that the difference of the results between the medium and the high resolution cases is significantly lower than between the medium and low resolution cases. Even considering that the change in resolution is also smaller between the high and the medium resolution cases, it can be expected that the higher resolution case is not far from reaching grid convergence. The largest deviation in the results is observable at the neck of the aneurysmal sac during the high flow rate phases near the systolic peak (see Fig. 6.5). The aneurysmal neck is where the vortex flow meets the main vessel current. This suggests an unstable velocity profile with strong spatial fluctuations.

With the multi-block structure used in all the Palabos based LBM simulations (see Chapter 5) 18 \( \times \) 18 \( \times \) 18 cell sized cubic sub-domains were defined. Those not containing any fluid or wall cells were simply omitted during the simulations. Only the cells of the remaining sub-domains were allocated and kept in memory. Due to the irregular geometrical shape of the vessel, some of the allocated cells were still unused. For pre-processing the total number of cells was used. Table 6.2 shows the number of used and allocated cells for each resolution.

<table>
<thead>
<tr>
<th></th>
<th>Low</th>
<th>Medium</th>
<th>High</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total cells</td>
<td>33.85M</td>
<td>114.39M</td>
<td>186.48M</td>
</tr>
<tr>
<td>Allocated cells</td>
<td>3.77M</td>
<td>9.33M</td>
<td>14.05M</td>
</tr>
<tr>
<td>Used cells</td>
<td>1.57M</td>
<td>5.09M</td>
<td>8.17M</td>
</tr>
</tbody>
</table>

Table 6.2: The total number of cells in the simulation domain, the number of cells allocated during the simulations using the multi-block structure, and the number of cells that actually contained fluid or wall in the three grids with different resolutions.
Figure 6.5: Velocity components sampled at a high flow-rate instant near the systolic peak (at \( t = 1.32 \text{s} \)) along the two measurement lines. In the upper image: at the neck of the aneurysm (denoted by (b) in Figure 6.1); and in the lower image: inside the sac (denoted by (a) in Figure 6.1).

The ratio of allocated cells to used cells could be decreased further using smaller sub-domain size, but as sub-domains have to exchange information on boundaries, it would also increase the communication demand between them.
Figure 6.6: Velocity components sampled at a low flow-rate instant after the systolic peak (at $t = 2.64$ s) along the same two measurement lines. In the upper image: at the neck of the aneurysm (denoted by (b) in Figure 6.1); and in the lower image: inside the sac (denoted by (a) in Figure 6.1).

6.2.2 Qualitative simulation results

The qualitative appearance of the emerging flow field is displayed in Figure 6.7. One can take notice of the complex nature of the flow in this figure in this particular geometry: the streamlines take a twist in every highly curved section of the vessel. During the cycle the overall qualitative picture of the
flow does not change significantly but as the magnitude of these twists increases with velocity, the axis of rotation inside the aneurysm sac tilts slightly towards the centreline of the vessel.

Figure 6.7: Emerging velocity field visualised by streamlines at the instant $t = 1.32$ s

6.2.3 Comparison with measurement results

The LDA measurement results are available along the red dashed lines marked in Figure 6.1. These lines are parallel with the $y$ axis for technical reasons. The line denoted by (a) is located inside the sac and runs nearly through the centre of the vortex up to the proximity of the wall. The other line (b) runs across the neck of the bulge. The LDA measurement velocity components use the usual notation of the $\vec{u}(x, y, z) = (u, v, w)$ velocity vector. The same notation is used for every velocity component. (For the coordinate system see Figure 6.1) The comparisons are shown in Figure 6.8 for a high flow rate instant and in Figure 6.9 for a low flow rate one. In both cases I use the results of the high resolution LBM grid. A good agreement is present between the experimental and the LBM simulation results, especially with the LDA results, which are certainly more accurate than the PIV results. The largest deviation can be observed, as it could be expected, at the neck of the aneurysm, where the flow profile is the most unstable.

In Figure 6.10 I show the time-dependent velocity components measured at the two points given in Figure 6.1. The symmetric error bars on the PIV data points simply reflect the positioning uncertainty and the real measurement error including other sources can be larger than this. For these time-dependent
Figure 6.8: Velocity components: $U$ in the top image along measurement line (b) and $W$ in the bottom image along the measurement line (a), both at the instant $t = 1.32\, \text{s}$.

Comparisons I used the medium resolution LBM results, which follow the PIV curves well within the error bars. The CFX curve shows some strong cut-off at the higher velocity regions (i.e., around the systolic peak). The changing of the laminar fluid model to the Baseline (BSL) Reynolds Stress model yields
better results and the improvement is most significant around the systolic peak. The overall agreement of the MRT LBM simulation with the LDA and PIV measurements can be considered good, especially inside the aneurysmal sac, which is actually my main objective.

Figure 6.9: Velocity components: U in the top image along measurement line (b) and W in the bottom image along the measurement line (a), both at the instant \( t = 2.64 \) s.
Due to the heavily curved nature of the geometry and the relatively high flow velocity, the simulation is not far from numerical instability at the highest flow rate instant. The two methods that seemed to handle small spurious oscillations better are the MRT and the ELBM methods. The MRT method, however, achieved this by fine tuning the ghost modes to increase stability (as pointed out in Section 4.4), without dampening the resulting velocity magnitudes.
6.3 THE SIMULATION ENVIRONMENTS

6.3.1 Environment for the LBM simulations

As mentioned before, I implemented the CPU LBM simulations used until this point on top of the Palabos library which uses the MPI\(^6\) interface for parallel communication. After I selected the most appropriate collision model (MRT) using this implementation, I also carried out the simulation with the third implementation (see Chapter 5) that uses the Sailfish library for GPUs using the CUDA language\(^7\) for performance comparison. The CPU based simulations were executed in a parallel MPI environment. This medium-sized cluster consisted of 64 computing nodes each containing two Xeon 5650 “Westmere” chips (12 cores + SMT) running at 2.66 GHz with 12 MB Shared Cache per chip and 24 GB of RAM (DDR3-1333). The nodes were connected by Infini-band interconnect fabric with 40 GBit/s bandwidth per link and per direction. The GPU version of the simulation was executed in a single node environment that contained two Xeon 5650 “Westmere” chips (12 cores + SMT) running at 2.66 GHz with 12 MB Shared Cache per chip, 48 GB of RAM (DDR3-1333) with two Tesla C2070 cards, and two Tesla C2075 cards.

The total memory requirement for a Lattice Boltzmann simulation can be approximated easily as:

\[
M_{\text{bytes}} = (\text{AccessPattern} \times Q_f + \text{ScalarArrays}) \times \text{Precision},
\]

where the value of AccessPattern is 1 for AA type access and 2 for AB type access. \(Q_f\) denotes the number of degrees of freedom of the distribution function (equals 19 in all my three dimensional simulations) while ScalarArrays counts for the number of used scalar arrays (typically equals 5 in a 3D LBM simulation: 3 arrays to store macroscopic velocity components, one for storing density, and one to describe the geometry). In my case this means that 192 bytes of data is stored per lattice in the CPU based LBM simulations. Table 6.3 shows the memory usage and run-time of the different grid resolutions.

It is known that the Lattice Boltzmann method is a memory bandwidth bound computation. An estimation of the upper limit for the performance can be given knowing the amount of allocated memory (that has to be read and written at least once per time iteration) and the memory bandwidth of the computational node. The cluster I used had 33.6 GB/s memory bandwidth per computational node, which transforms to 94 million lattice updates per second per computational node, or 7.8 million lattice updates per computational core. Naturally, the real performance is usually lower due to inter-node communication latency and other computational costs. In my case, the higher the grid resolution was, the closer the performance was to the theoretical limit.

\(^6\) The OpenMPI v.1.5.3 was used with the gcc 4.7.0 compiler
\(^7\) CUDA v5.0 - http://www.nvidia.com/object/cuda_home_new.html
Table 6.3: Memory requirement of the different resolution simulations. The allocated memory is the amount required during computations (note that the actual memory usage can be somewhat higher due to other structures used for the domain decomposition for example) while the total amount was only required during pre-processing. The required time to complete the simulations is also shown.

<table>
<thead>
<tr>
<th>Resolution</th>
<th>Alloc. memory</th>
<th>Total memory</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low</td>
<td>0.69 GB</td>
<td>6.2 GB</td>
</tr>
<tr>
<td>Medium</td>
<td>1.78 GB</td>
<td>21 GB</td>
</tr>
<tr>
<td>High</td>
<td>3.26 GB</td>
<td>34.1 GB</td>
</tr>
</tbody>
</table>

The larger numerical grid means that more sub-domains are located on the same computational node, thereby reducing the amount of inter-node communication. This implies that during my simulations I have not reached the point where the communication overweights the additional computational benefit of an added node.

6.3.2 The GPU run-time results

GPU cards use higher bandwidth on-board memory chips. The four Tesla cards used in this measurement all had a nominal 148 GB/s memory bandwidth. During the measurements the error-correcting code (ECC) was enabled, which means that an additional 12.5% memory is used for the ECC bits and that some portion (around 20%) of the theoretical memory bandwidth is occupied with the ECC operations. Turning off this ECC was not an option in the supercomputing environment. I tested the low resolution grid with a simpler domain decomposition: the whole simulation domain was divided into two parts along every axis resulting in 8 sub-domains. The sub-domains were computed in four separate processes (one process per GPU) connected by ZeroMQ\(^8\) IPC connection. For performance considerations single precision floating point arithmetic was used with the D3Q19 MRT Lattice Boltzmann model. The single point precision in this case only caused slight differences compared to the CPU implementation results. The simulation completed in 42 minutes 21 seconds. That run-time is actually comparable with the run-time of the CPU version. It should be noted, however, that the CPU version is run on a cluster machine (64 nodes × 12 cores = 768 cores) while the GPU version used only four Tesla cards in a single node.

---

8 [http://www.zeromq.org/](http://www.zeromq.org/)
6.3.3 Scaling of the LBM implementations

I tested the scaling of the CPU and GPU LBM implementations on the low resolution geometry using the MRT approach. The CPU version used a double precision numerical representation while the GPU implementation used a single precision one.

Figure 6.11: The upper part of this figure shows the scaling of the MPI parallel CPU implementation up to 768 cores while the lower part shows the scaling of the GPU implementation on four Tesla cards.
Both implementations exhibit nearly linear scalability, as shown in Figure 6.11. At this geometry resolution the upper end of the CPU scaling curve approaches saturation as the computational benefit coming from the increased number of cores cannot compensate the communication surplus effectively. In the GPU part of the figure, there is a small fallback in performance scaling when using three Tesla cards. It has an obvious reason: to avoid changing the computational conditions, the same domain subdivision was used, but laying out eight sub-domains to three GPU cards causes some imbalance in the work distribution.

6.3.4 Environment for the finite-volume simulation

I run The CFX simulations on a desktop–class machine with an Intel Core i7-2600 chip that has 4 cores running at 3.4 GHz with 8 GB of RAM, with a nominal 22 GB/s memory bandwidth. The unstructured numerical grid consisted of 0.81 million tetrahedra. Using this grid the memory consumption for this simulation was approximately 0.92 GB. The run used MPI local parallelisation with the MetIS partitioning algorithm. On this machine the simulation took 15 hours 03 minutes 14 seconds to complete. The comparison of the results with the LBM simulations might not seem fair as even the low resolution model used nearly twice as many numerical elements, though not body-fitted ones. In a real-world application the higher resolution of the LBM models would not cause any memory problem during the simulations as the amount of required memory even at a few million numerical cells is well within the range that even an average desktop computer can handle (see Table 6.3 for allocated memory for the different resolutions) but naturally, the resolution of the finite volume mesh could be increased as well.

6.4 Concluding the validation

In the geometry of this real intracranial aneurysm model I found that the lattice Boltzmann method is capable of simulating 3D transient flows with an accuracy matching a well-tested commercial finite volume solver. I tested several versions of the LBM and according to these runs, the multiple relaxation time (MRT) model proved to be the most accurate at the medium grid resolution. The LBM simulation results in general are in good agreement with the experimental results and the time-dependent comparisons showed an improved accuracy compared to the laminar finite volume solver. I have to note that this also raises the question whether the flow inside these complex geometries always remains in the laminar flow regime, especially near the systolic peak. The better results of the BSL model seem to point to a different direction. The resolution of the medium LBM grid might have been
large enough to simulate the flow directly without the help of a turbulence model. As for the preparations of the computations, one advantage of the lattice Boltzmann method is the lack of need for a tedious numerical mesh generation which significantly simplifies the pre-processing stage of the simulations. Finally, I would like to mention that using the capability of the developed tools, both the LBM implementations and other utilities I have developed for pre- and post-processing, it might be possible to completely automate the simulation pipeline and to accelerate the simulations by at least an order of magnitude by implementing the code for a GPU environment. The newest generation of Tesla cards (K40 at the time of writing) have increased the memory bandwidth even further, promising even better simulation run-times. Additionally, a more sophisticated domain decomposition would also decrease the GPU version run-time significantly while also reducing the consumption of valuable video memory. From a medical point of view, this set of programs might carry the potential to create an accurate on-line numerical simulation tool to aid medical decision-making.

6.4.1 Thesis 1

I tested four different models of the lattice Boltzmann method: the Regularised model, the Entropic model, the Incompressible BGK model, and the Multiple Relaxation Time model. I demonstrated that from these the Multiple Relaxation Time collision model reproduced the transient flow field inside a real aneurysm geometry the most accurately compared to experimental results. Thus, it is the most suitable for simulating blood flows in aneurysm-like geometries. [T3, T6]

6.4.2 Thesis 2

I demonstrated that implementing the lattice Boltzmann method-based transient hemodynamic blood flow computation on a suitable GPU hardware with the appropriately chosen boundary condition implementations can deliver at least the theoretically expected one order of magnitude performance increase compared to its CPU implementation. This performance increase is sustainable while scaling to multiple computational units (multiple nodes in case of CPU implementation and multiple graphics cards in case of the GPU implementation). [T1, T3, T6]
Part III

Applications
HAEMOSTASIS MODELLING IN TWO DIMENSIONS

7.1 INTRODUCTION

The thrombosis process in cerebral aneurysms would be very useful to predict from an available flow field. When using a flow diverter (stent) during the treatment sessions, the aim is to slow the currents down inside the bulge to the point when the blood begins to form a clot. This clot would then grow to occlude the aneurysmal sac, henceforth closing it out completely from the circulation. The prediction of the occlusion process using numerical computational tools poses some difficult problems. For one, the time scale of the occlusion is hard to tackle with simulations as they can span over several days. Apart from this problem, the biochemical processes during the formation of thrombi are rather complicated. Thus, computing the whole process for days’ length of time does not seem feasible. To circumvent these, Ouared and Chopard (2005) presented a qualitative approximation using a simple model with strong assumptions about the time scale. As a first step towards understanding, instead of trying to reduce the time-scale I chose to investigate the haemostasis process on a smaller length scale, inside an arteriole where the duration of the formation can be measured in seconds. The cascade process of the haemostasis is a vital self-defence process exhibited by our body. This is a natural response of the cardiovascular system that aims at preventing undesirable blood loss in case of vascular injury. The blood coagulation is an important stage during the haemostasis and its highly complicated cascade process is driven by several factors that can act either as a promoter or an inhibitor. The main steps include the aggregation and the binding of the platelets that are dependent on the local flow properties among other factors. Simulating this aggregation phenomenon is also a challenging task as one has to make several appropriately chosen approximations. Anand et al. (2003) reviewed some of the interacting mechanisms regarding clot formation in flowing blood and even though they deal with a narrower collection of participating processes than it is known in the current literature, from the view of a numerical simulation they are still numerous. Nesbitt et al. (2009) provided some more insight into the thrombus formation by emphasising the strong coupling between the biochemical processes and the local hemodynamic properties. According to their experimental work, platelet aggregation is influenced by emerging shear forces as platelet adhesion favours low-shear zones. Several other authors recorded the ongoing process of thrombus formation: for example Celi et al. (2003) recorded the thrombus formation in the
micro-circulation of a living mouse using widefield video-microscopy. Similar measurements were carried out using different reagents to highlight the concentration of the main components of the formation (Falati et al., 2002; Furie and Furie, 2007). While the number of high quality recording of this phenomenon is increasing, the numerical simulations are still in a rather early stage. There are several pieces of work aiming at simulating thrombus formation numerically but most of them cover only portions of the whole process (Alenitsyn et al., 2008; Harrison et al., 2007; Tamagawa et al., 2009; Wang and King, 2012). They try to solve the complex problem by approaching it either from the biochemical side and compute the reactions and the densities of the reagents but fail to couple these equations with the transient (time-varying) flow field, or from the kinetic side, which usually lacks the appropriate handling of platelet concentrations. The platelet concentration profile inside the blood vessels is an important factor, which is far from being constant in either venous or arterial vessels (H. Zhao and Shaqfeh, 2010).

A lot of numerical models were developed in the past decade in order to calculate the platelet concentration profile (Jordan et al., 2004; Tokarev et al., 2011; R. Zhao et al., 2007). Their main findings are that the marginating property of platelets is mainly caused by the finite size effect of both the smaller platelets and the significantly larger red blood cells (RBC). Technically, the rolling motion of the larger RBCs pushes the platelets to the sides of the flow channel, thus creating a highly uneven platelet distribution across the diameter of the vessels. This drift force acting upon platelets in blood flow seems to be proportional to the average local flow velocity. To prove this interaction between platelets and RBCs, Mountrakis et al. (2013) simulated particle collisions in blood flow using the immersed boundary method.

7.2 METHODS OF THE SIMULATION

Validating the simulations of biological processes is often not trivial. The three-dimensional medical images are typically available in larger vessels only, since the resolution of the employed modalities is usually not sufficient to record the details of a sub-arterial scale coagulation process. Furthermore, for arteries or for aneurysms, as mentioned in the previous section, the clotting process can span from hours to days. Henceforth, I chose to carry out the simulation in two dimensions inside an arteriole as a first step, with the intention to extend the investigation to three-dimensional cases in the future. For this investigation, the coagulation process spreads to several seconds and video-microscopy images are available in the literature to use as a validation. With this choice, the computational costs are obviously reduced as well. During my simulations the blood coagulation mechanism altered the geometry in time, which was made possible by using my first LBM implementation (see
Chapter 5). I employed the MRT collision operator for the lattice Boltzmann model again, which, for this case provided increased numerical stability compared to the perhaps more widely used BGK model. The platelets immersed in the blood flow were simulated as a passive scalar concentration field. The passive attribute means that although the platelet distribution is driven by the pulsating blood flow, it has no influence back on it. This approximation is supported by the fact that the usual volumetric ratio of the platelets in blood is under 1%. The advection-diffusion of these platelets was handled by the LBM method in the following way: when the velocity is prescribed by an already given velocity field (like in this case by the blood flow), the equilibrium function (from the original function) of the scalar field reduces to Eq. 7.1, where \( \rho \) is the platelet concentration that can be computed analogously to fluid density (Eq. 4.6):

\[
f_{i}^{eq}(\vec{x}, t) = w_{i} \rho[1 + 3 * \vec{e}_{i}\vec{u}].
\]  

(7.1)

The diffusion coefficient \( D \) is then analogous to the numerical viscosity (Eq. 4.3):

\[
D = \frac{1}{3}(\tau - \frac{1}{2}).
\]  

(7.2)

To acquire a proper platelet density profile, the interaction with the RBCs must be taken into account. The platelet-drifting is mainly caused by the rolling motion of the RBCs, which is in turn caused by the emerging shear forces in the flowing fluid. To simulate this marginating effect, I prescribed a virtual force field \( \vec{F}_M \) that acts upon the density of the platelets. This force actually accounts for the finite size effects of the platelets and RBCs. The drift force acting on real platelets always drives platelets out of the main flow, towards the sides of the vessel. It is a required behaviour as this causes increased platelet concentration in the vicinity of vessel walls where any injury can possibly occur.

- This \( \vec{F}_M \) force is perpendicular to the inclined plane of the maximum shear stress at a given location and points towards that side of the plane along where the velocity gradient is negative.

- Its magnitude is proportional to the magnitude of the emerging shear stress acting in the aforementioned plane. I note here that this proportionality ratio definitely depends on the material properties and the relative sizes of the two particle types (i.e., the platelets and the RBC cells), and that it will require a proper parameter study later. In the current work, it takes the value of unity.

I used the forcing term proposed by Guo, Zheng, et al. (2002) to apply this force upon the passive scalar field of the platelets.
7.3 FAST NUMERICAL CALCULATION OF LOCAL SHEAR STRESS PROPERTIES

In contrast to the finite volume simulation, the lattice Boltzmann formalism allows the local computation of the shear stress tensor. The key idea is, following the Chapman-Enskog analysis (Chapman and Cowling, 1992), that the lattice Boltzmann equation can be expanded into a Taylor series and the distribution function can be written as a power series of a small parameter ($\epsilon$), often identified as the Knudsen number:

$$f_i = f_i^{(0)} + \epsilon f_i^{(1)} + O(\epsilon^2), \quad (7.3)$$

where $f_i^{(0)} = f_i^{eq}$ denotes the equilibrium distribution while the higher order members form the off-equilibrium part of the distribution (also called non-equilibrium part). The shear rate tensor has the following components:

$$\dot{\mathbf{S}}_{\alpha\beta} = \frac{1}{2} (\partial_\alpha u_\beta + \partial_\beta u_\alpha). \quad (7.4)$$

This tensor can be related to the second-order moment of the $f_i^{(1)}$ part of the distribution function as shown in the work of Krüger et al. (2009):

$$\hat{\Pi}^{(1)} = -\frac{2c_s^2 \rho}{\omega} \dot{\mathbf{S}}, \quad (7.5)$$

where the tensor $\hat{\Pi}^{(1)}$ denotes this second-order moment. The difficulty here is that the $\hat{\Pi}^{(1)}$ tensor depends on the value of $f_i^{(1)}$ which is not computable as a standalone quantity. Instead, I use the full non-equilibrium distribution that according to Eq. 7.3 can be regarded as a first-order accurate approximation of $f_i^{(1)}$. It is easy to calculate it with the use of the equilibrium part $f_i^{eq}$ as $f_i^{neq} = f_i - f_i^{eq}$. This leads to the following expression, which is computable locally at every lattice site:

$$\mathbf{\hat{S}} = -\frac{\omega}{2c_s^2 \rho} \hat{\Pi}^{(1)} = -\frac{\omega}{2c_s^2 \rho} \sum_i (f_i - f_i^{eq}) \mathbf{\bar{c}}_i \mathbf{\bar{c}}_i. \quad (7.6)$$

For the virtual force I need the absolute value of the emerging maximum shear stress as well as its inclined plane. The required pieces of information can be computed from $\mathbf{\hat{S}}$. Generally, LBM simulations use a huge number of numeric cells as they are relatively cheap to compute, at least compared to the typical computational cost of a finite-volume numerical cell during one time-step. Thus, it would make sense to aim for a numerical method that allows the cheapest possible computation of the desired shear properties.

The technique of Mohr’s Circle is well-known for engineers (for a reference see (Beer and Johnston Jr, 2012)). It is helpful for the visualisation of the
relationships between normal and shear stresses acting on various inclined planes at a given point of a stressed body. This technique can be generalised to any symmetric real valued $2 \times 2$ matrix in two dimensions or $3 \times 3$ matrix in three dimensions. Using this technique, one can extract the direction of the principal stresses and the maximum shear stresses acting on inclined planes. For Newtonian fluids the viscous stress tensor can be related to the shear rate tensor as:

$$\dot{\sigma} = 2\mu \dot{S}, \quad (7.7)$$

where $\mu$ denotes the dynamic viscosity. In two dimension the components of this tensor can be written as:

$$\dot{\sigma} = \begin{bmatrix} \sigma_x & \tau_{xy} \\ \tau_{xy} & \sigma_y \end{bmatrix}. \quad (7.8)$$

Using these notations the magnitude of the maximum shear stress equals the radius of Mohr’s Circle:

$$\tau_{max} = \sqrt{\frac{1}{2}(\sigma_x - \sigma_y)^2 + \tau_{xy}^2}, \quad (7.9)$$

while the orientation of the principal planes (which are $90^\circ$ apart) can be recovered as:

$$\tan 2\theta_p = \frac{2\tau_{xy}}{\sigma_x - \sigma_y}. \quad (7.10)$$

Therefore, using these pieces of information, the whole computation of the virtual force $\vec{F}_M$ can be reduced to a few algebraic operations and a trigonometric function per lattice site.

### 7.4 Platelet Distribution

Applying the $F_M$ virtual drift force, I simulated two simple channel flows, one venule and one arteriola, with Reynolds numbers that are typical in these smaller vessels. The vessel wall was taken as a smooth, no-slip, rigid wall implemented with full-way bounce-back dynamics. The results of the simulations are compared to the experimental results of Woldhuis et al. (1992).

The first simple channel flow simulation used constant velocity inlet and constant pressure outlet boundaries to conform the physiological conditions of a venule. The Reynolds number was set to one, which is a typical value in venules. The result of the simulation is summarised below in Fig. 7.1.

The platelet concentration shows a rather good qualitative agreement with the experimental results. One property of my results, however, is not intentional. The platelet concentration towards the vessel walls increases proportionally with the emerging shear stress between neighbouring fluid elements.
Figure 7.1: Normalised platelet concentration profile in a venule. The continuous red curve shows the results of the stationary simulation plotted over the experimental results of Woldhuis et al. (1992) (Re=1)

In the very close vicinity of the wall the density decreases again. The reason for this is that the platelets actually rolling along the vessel wall experience a lift force that gently drives them away from the wall. In my simulations this concentration decrease is present for a completely different technical reason: at the walls the passive scalar field that represents the platelet concentration obeys the full-way bounce-back dynamics. Therefore, the platelets next to the wall, experiencing the largest margination force, are bounced back away from the walls. Since this effect is dependent on the lattice size, the accuracy of the resolved concentration profile next to the vessel boundaries is going to require a grid-size effect study in the future but the appropriate effect is likely to be achievable by bouncing back only a fine-tuned portion of the concentration.

I tested this numerical model with pulsatile flow as well, in order to reinforce the statement of its qualitative validity inside arterioles. For this computation, the simulation needed a time-varying inlet velocity boundary condition. Figure 7.2 shows the inlet flow rate profile that mimics the behaviour of a real cardiac pressure wave (it has the same shape as the one I used with the 3D validation in Chapter 6 but it is normalised to a flow rate realistically possible inside arterioles).

The Reynolds number was defined with the maximum of the possible inlet velocity, thus it was set to ten, which is a usual physiological value for arterioles. The results of the simulation are also compared to the experimental results of the same authors in Fig. 7.3. Again, a good qualitative agreement
is found. The biggest difference between the simulation and experimental results can be found very close to the vessel walls for the above mentioned reasons but at the given resolution and Reynolds number it is still within the error bars.

The coagulation of blood is modelled simply by changing the type of a numerical element from fluid cell to solid cell. For the thrombus formation in

Figure 7.2: Normalised artificial flow rate curve that mimics the behaviour of a real flow rate curve during a cardiac cycle.

Figure 7.3: Time-averaged normalised platelet concentration profile in an arteriole. The continuous red curve shows the results of a transient simulation plotted over the experimental results of Woldhuis et al. (1992) (Re=10)
my simulation, coagulation is only possible at sites right next to vessel walls or next to rigid blood cells that have already come to stasis. All the fluid LBM cells next to a wall cell were tracked as these cells had the ability to turn into a solid cell themselves. When coagulation at a lattice site happens, its fluid cell neighbours are added to the list of near-wall cells and from that moment on they are possible locations for further coagulation. Though platelet activation in reality is a rather complex cascade process (with several processes not fully understood), I only simulated the concentration of one reagent, the adenosine diphosphate (ADP) which is in my simulation not coupled with the blood flow. This means that the ADP concentration is not directly influenced by the local flow properties. When a platelet is activated (or in our simulation when coagulation of a numerical lattice happens), it releases some ADP that can initiate the activation of other platelets. In the simulations the ADP concentration decreases exponentially in space, consequently, its release can be thought of as a local ADP concentration increase. This diffusion process, however, has in my approximation no temporal dynamics, it is totally instantaneous.

Coagulation at a numeric lattice site in my model therefore is dependent on the relation of three variables:

\[
P_{\text{coag}} = \frac{\rho_{\text{platelet}} \cdot \rho_{\text{ADP}}}{\tau_{\text{max}}},
\]  

(7.11)

where \(P_{\text{coag}}\) is a probability in the sense that its value decides whether a fluid cell should come to stasis or not at any given time, based on the current local platelet concentration \(\rho_{\text{platelet}}\), the local ADP concentration \(\rho_{\text{ADP}}\), and the local maximum shear component of the stress tensor denoted by \(\tau_{\text{max}}\). The threshold level of \(P_{\text{coag}}\) that a numerical lattice has to reach for coagulation is an empirical parameter of the model for now. A later work should explore the deeper relationship among these parameters. For a numerical fluid cell to come to stasis, these three parameters have to remain in the coagulation zone for a \(t_{\text{window}}\) time. This time-window was chosen to be \(t_{\text{window}} = 20\) ms because this is the usual time-frame for ADP to activate a platelet. This also means that a newly registered near-wall lattice cannot turn into a solid cell sooner than 20 ms.

### 7.5 Preliminary Results

In Figure 7.4 I show a visualised snapshot of a coagulation simulation after six heartbeat cycles. The formation of the thrombus starts at an injury site. In in-vivo systems in case of vascular injury, a significant amount of ADP is released into the bloodstream from the injured vessel wall tissue. The injury site in this case was simulated by increasing the ADP concentration artificially
Figure 7.4: Typical result of a thrombus formation simulation at the instant of the end of a heartbeat cycle. The flow direction is from the left to the right. The upper image shows the emerged geometry of the thrombus, with the place of vessel injury (yellow box at the base of the thrombus). The middle image shows the current normalised velocity magnitudes (Reynolds number equals 10). The lower image shows the normalised platelet concentration at the same instant.

at a small amount of numerical lattices along the vessel wall. A large enough concentration induces a thrombus formation process starting from these lattices.

There are a few qualitative features of the calculated thrombus that should be mentioned. First, as the thrombus begins to block the flow channel, the velocity increases and that leads to larger emerging shear stresses that inhibit the process of coagulation. This results in the relatively flat top of the thrombus. It is also noticeable that the thrombus growth is asymmetric to the site of injury; its growth is slower on the side facing the incoming bloodstream. This is again the result of the different shear stresses.

The simulation results were compared to two thrombus geometries of the same vascular injury recorded at different times with video-microscopy by Nesbitt et al. (2009) in Figure 7.5. These experimental results originate from a laser-induced injury inside the arteriole of a living mouse. The exact form of the thrombus naturally depends on much more components that are taken into account in the current numerical simulation. Still, some qualitative features such as the effects of shear stresses can be studied with it.
Figure 7.5: Geometry of the thrombus after A.) three and after B.) six heartbeat cycles. The yellow dotted outline shows the geometry of the thrombus recorded with video-microscopy by Nesbitt et al. (2009).

7.6 CONCLUSIONS

Haemostasis is a heavily studied topic that displays deep complexity. The general aim of a numerical simulation is to reproduce certain features of a phenomenon using the smallest necessary state space. In this chapter I presented a simple numerical model for haemostasis that incorporates different contributing disciplines in a computationally effective way. It is evident that this model has several shortcomings, for example in a real thrombus the outer layer is often not solid and can dissolve easily in the flow and subside again later whereas in my model the thrombus is handled as a solid material unable to dissolve. Furthermore, the effects of several other components of the coagulation cascade were not taken into account either. Still, according to the comparison, these key components coupled with the blood flow seem to be sufficient to reproduce the major characteristics of an injury induced thrombus forming in a shear-constrained environment, inside an arteriole. Finally, I would like to mention that due to the strongly simplified nature of the model, it might be general enough to be applicable for simulating other sediments in blood flows, such as the deposition of fat particles on clinical devices.

7.6.1 Thesis 3

I proposed a virtual force which aims at artificially reproducing the highly non-uniform distribution of the platelets inside blood vessels using the viscous stress tensor of the velocity field. Applying this virtual force upon a scalar density field representation of the platelets yields a realistic margination behaviour in an arteriole section for both stationary and pulsatile flows in two dimensions. This force in a lattice Boltzmann simulation can be calculated locally at every lattice site by using the non-equilibrium part of the density distribution function, therefore in parallel computations this method can be regarded as an efficient solution. [T2, T5]
7.6.2 Thesis 4

I developed a simple model for the simulation of the haemostasis process which uses only the properties of the blood flow along with two variables, namely the local concentration of the platelets and of an inhibitor, the adenosine diphosphate out of the more than two dozen factors currently known to influence the real biological cascade process. I simulated a case of induced vessel injury in two dimensions and this model was capable of producing qualitatively good results compared to experimental video-microscopy images of a real thrombus formation. [T2, T5]
8

CHAOTIC PROPERTIES OF AN ANEURYSMAL FLOW

8.1 INTRODUCTION

Fluid flows that incorporate several biological and chemical processes can be found throughout the different disciplines of science incorporating numerous length-scales (such as plankton blooming (Abraham, 1998; Martin, 2003; Schelin, Károlyi, A. P. de Moura, N. A. Booth, et al., 2010) or ozone hole formation (Eckhardt, 1988; Solomon, 1999)), through chemistry (Epstein, 1995; Menzinger and Jankowski, 1986; Metcalfe and Ottino, 1994) and combustion (Kiss et al., 2003; William, 1985), to microfluidics (Müller et al., 2004; Stroock et al., 2002). In the current chapter, the studied flow originates from a high resolution simulation carried out on the geometry of a medical recording (CT angiography) of a cerebral aneurysm. I showed the validity of this computed flow field in Chapter 6.

It has been shown (Tél, A. de Moura, et al., 2005) that mixing in fluids has an important influence on the biological or chemical activity that takes place in the fluid. Probably the simplest approach to explore the properties of mixing in a fluid flow is to consider a passive scalar field, that is, tracer particles of negligible size and mass that take on the local velocity of the fluid instantaneously without inertia effects. The equation of motion of such particles is therefore \( \dot{\vec{r}}(t) = \vec{v}(\vec{r}, t) \), where \( \vec{r}(t) \) is the position of the particle at time \( t \) and \( \vec{v}(\vec{r}, t) \) is the velocity field of the fluid. Even for laminar time periodic flows, the solution of this equation is typically chaotic (Aref, 1984; Ottino, 1990; Zsugyel et al., 2014), the particle trajectories deviating from the streamlines causing a strong but non-turbulent mixing. This kind of complex behaviour in fluid flows is called chaotic advection (Aref, 1984). I should reinforce here that I am not stating that the flow is chaotic but rather the particle trajectories inside the flow are so.

Generally, open flows constitute a large class of fluid flows where fluid is continuously transported into a certain region of observation and out of it. In these flows, a part of the advected particles can get trapped for a long time within the region of observation. The majority of the particles leave the region of observation in a short time but those that remain for a long time accumulate along a filamentary fractal pattern (Károlyi and Tél, 1997; Tél, A. de Moura, et al., 2005) representing the unstable manifold of a chaotic set, the latter consisting of unstable (saddle type) periodic and non-periodic particle trajectories enclosed in the region of observation. The particle paths forming the chaotic set might seem difficult to acquire (as being an unstable manifold,
one would need to be infinitely close to a point of the set to find it) but they are actually accessible from special initial conditions that form its stable manifold. Trajectories initiated from the stable manifold approach the chaotic set and get trapped in the region of observation.

The fractal dimension $D_0$ of the unstable manifold gives information on the geometry of the patterns traced out by the advected particles while the information dimension $D_1$ also characterises the probability distribution (or relative density) of particles along the fractal unstable manifold (Té l and Gruiz, 2006). To make the definition of $D_0$ and $D_1$ more clear with the help of one possible fractal dimension measurement, let us consider the box-counting method. That is, if we have a chaotic set embedded in a two-dimensional surface, we can draw a square with the side length of $\epsilon$ around it. If we decrease $\epsilon$, the number of squares ($N(\epsilon)$) needed to cover the whole set will increase. The fractal dimension describes this scaling behaviour:

$$D_0 = \lim_{\epsilon \to 0} \frac{\log N(\epsilon)}{\log \frac{1}{\epsilon}}. \quad (8.1)$$

This method, however, does not take into account that the chaotic set might present different densities at different locations. Henceforth, information dimension uses weighted squares based on the density of the set in the covered area:

$$D_1 = \lim_{\epsilon \to 0} -\langle \log p_\epsilon \rangle \frac{1}{\log \frac{1}{\epsilon}}, \quad (8.2)$$

where $p_\epsilon$ is used as weighting factor and it denotes the possibility for a given box that a given point of the chaotic set will fall underneath it. Naturally, even if I described this example using squares and a two-dimensional "host" space, it is easily generalisable to higher-dimensional spaces using higher-dimensional boxes for counting.

The appearance of filamentary fractal patterns is the result of the stretching and folding action of the mixing fluid generated by the strong sensitivity on the initial conditions, the main characteristics of chaos. This strong sensitivity on initial conditions is characterised by the average Lyapunov exponent $\lambda$ describing the typical exponential rate of separation of initially close particles:

$$d(t) = d(0)e^{\lambda t}$$

where $d(t)$ is the distance between the initially close particles at time $t$ (Tétel and Gruiz, 2006). In open flows, another important number is the escape rate $\kappa$ which gives the rate of exponential decay of the number of particles still in the region of observation at time $t$: $n(t) = n(0)e^{-\kappa t}$. These quantities are not independent of each other, the relation $D_0 = D - \kappa / \lambda$ holds between them where $D$ is the dimension of space in which the quantities are measured (Tétel and Gruiz, 2006).

Chaotic advection can also have important consequences in case of blood flows (Schelin, Károlyi, De Moura, et al., 2009; Schelin, Károlyi, A. P. de
Moura, N. A. Booth, et al., 2010; Schelin, Károlyi, A. P. de Moura, N. Booth, et al., 2012). For instance, platelets are transported in the blood while they are biochemically active. It has been shown by Károlyi and Tél (2005); Tél, A. de Moura, et al. (2005); Toroczkai et al. (1998) that the fractality of the pattern traced out by active particles dramatically modifies the reaction equations describing their activity. Productivity, which is proportional to the concentration of the chemical constituents in a well-mixed medium, turns out to depend on the information dimension of the distribution of the constituents in an open flow (Károlyi and Tél, 2005; Tél, A. de Moura, et al., 2005). This dependence is of singular type, productivity is proportional to $c^{-\alpha}$ where $c$ is the average concentration of a species and $\alpha = (D_1 - 1)/(2 - D_1)$ (Toroczkai et al., 1998). In the referred work, a two-dimensional idealisation was assumed, hence $1 < D_1 < 2$ and $\alpha > 0$. This implies that, in contrast to a well-mixed medium, the production is proportional to (nearly) the reciprocal of the concentration rather than the concentration itself. The reason behind the resulting advantage of rarity principle is that the fewer particles trace out the finer and finer details of the filamentary fractal structure of the unstable manifold, the longer the surface becomes between the chemical constituents. Naturally, this principle is only valid as long as fractal patterns are traced out by the constituent. A good example is platelet activation which has been speculated by Schelin, Károlyi, De Moura, et al. (2009) to follow similar rules. Hence, it can be of great interest to obtain information on the fractal and information dimension of biochemically active particle patterns in blood flows in general, particularly for the cases of aneurysmal flows and flows involving thrombus formations.

8.2 METHODS

8.2.1 Flow field

Real blood vessels usually have highly curved and irregular shapes. The layout of the vessel section used throughout the current chapter was obtained from CT (X-ray computed tomography) angiography. The flow field of the blood was obtained from an experimentally validated CFD simulation Závodszky and Paál (2013) that I presented in Chapter 6. As I mentioned there, the simulation used a very fine numerical grid in order to handle the aforementioned highly curved and irregular nature of the geometry. The obtained time-varying flow field represents the realistic blood flow during a single cardiac cycle. Figure 8.1 shows the layout of the vessel geometry again and also gives an impression of the sensitive dependence of the particle trajectories and initial conditions by showing the trajectories of three tracers initiated very close to each other (with under $10^{-4}$ mm initial distance among them).
During the CFD computation, the vessel wall was considered to be a rigid, no-slip surface. On the inlet and on the two larger outlets, time-dependent volumetric flow rates were prescribed that matched a realistic curve during a full cardiac period. This function is the same as the one used in refs. (Ugron et al., 2012; Závodszky and Paál, 2013) and it is also shown in Fig. 6.2. The third, smaller outlet was defined as a constant pressure boundary to avoid over-constraining the model. The blood was approximated as a Newtonian fluid. For a more detailed description of the simulation parameters and the boundary conditions see chapter 6.

![Figure 8.1: The layout of the aneurysm and the investigated segment. The inlet can be found at the bottom of the figure while the other numbered openings are outlets. The coloured paths are simulated tracer trajectories starting from the cross-section at the inlet section of the vessel.](image)

### 8.2.2 Free-energy function

One possible technique to measure important chaos characteristics in a relatively simple way is to take a large number of starting positions along a line segment of length $L$ which intersects the stable manifold of the chaotic set, from where the unstable set is accessible. Then if I measure the residence time for each initial condition, that is, the time it takes for the particles to pass
through the investigated flow domain, the number $N(t)$ of intervals along $L$ with residence time longer than $t$ can be counted.

Let $\ell_i(t)$ denote the length of these intervals along $L$ with $i = 1, 2, \ldots, N(t)$. I explain this notation in more details with the help of Fig. 8.2a and Fig. 8.3. For a large enough $t$ the free energy function $F(\beta)$ can be defined as in Tél (1988)

$$e^{-\beta F(\beta)t} \propto \sum_{i=1}^{N(t)} (\ell_i(t))^\beta. \quad (8.3)$$

The important quantities describing chaotic advection can then be derived from this function, for instance,

$$\kappa = \beta F(\beta)\big|_{\beta=1}, \quad (8.4)$$

$$\lambda = \left. \frac{d\beta F(\beta)}{d\beta} \right|_{\beta=1}, \quad (8.5)$$

$$0 = \beta F(\beta)\big|_{\beta=D_0}, \quad (8.6)$$

$$D_1 = \left. \frac{dF(\beta)/d\beta}{d\beta F(\beta)/d\beta} \right|_{\beta=1}. \quad (8.7)$$

8.2.3 Computation of chaotic properties

The accuracy of the chaotic properties computed through the free-energy function formalism depends on the accuracy of the calculated particle trajectories and the accuracy of the numerical evaluation of the $\beta F(\beta)$ function, which in turn depends on the number of computed trajectories initiated along a given line segment. Thus, it seems natural to elaborate on the accuracy of these two computations. The time-dependent velocity field I described previously is used for calculating the paths of passive tracer particles of negligible size and mass. Based on the mean flow velocity (approximately 0.3 m/s), the tracers starting from the inlet and following the centreline should leave the geometry through one of the outlets in about 1.1 s. For numerous particles it takes more time to leave as they can be slower than the average portions of the flow and the pathlines rarely follow the centreline. For this reason, I usually run the computations for up to 20 heart beat cycles (equal to 20 s). The computation of the paths requires high numerical precision during the integration as the flow field has several regions where the velocity vector quickly changes from point
Figure 8.2: (a) Residence time as a function of the initial position along a line section of length $L$. (b) The distribution of residence times of particles initiated from the plane shown in Fig. 8.1. The computation is run for up to 20 s, the peak at 20 s shows the number of particles still inside the domain.
Figure 8.3: The number $N(t)$ of intervals of length $\ell_i(t)$ at three different residence time levels within the enlarged area of the small red rectangle indicated in Fig. 8.2(a).
to point. These regions can make adjacent particles diverge quickly, therefore a small inaccuracy in their position can produce rather different trajectories. Typical unstable regions are the highly curved portions and the branching points but the most vigorous one is around the mouth of the aneurysmal sac. The effect of these unstable regions is observable in Fig. 8.1 as these are the typical points where the three trajectories shown in the figure separate. As a result of the sensitivity to initial conditions, a very small time-step has to be used during the integration process. For the integration I chose the 4th order Runge-Kutta method, which is a fourth order explicit method. Due to the very small time-steps stability is not an issue for this explicit method and it is considerably easier to implement compared to an adaptive method. The derivative function (the local velocity of the tracer particles) was evaluated using linear interpolation among the eight nearest grid points enclosing the actual point in space and also between the two neighbouring snapshots in time.

I test the convergence by measuring how the information dimension $D_1$ depends on the integration time step (Fig. 8.4) and on the number of trajectories initiated along a line segment to compute the free energy function $F(\beta)$ (Fig. 8.5). To compute Fig. 8.4, the density of the traced trajectories is set to a high value: $8 \times 10^5$ particles on a 3 mm line segment, which means the distance between the starting positions of the particles is $3.75 \times 10^{-9}$ m. A high value in this context means the highest sensible achievable value using the computational resources available to me. For the evaluation of the $\beta F(\beta)$ function with different number of trajectories in Fig. 8.5, I set the number of time divisions between two snapshots to 600. I used the same test line segment gradually increasing the number of particles launched along the line segment in each run. Figure 8.5 summarises the results of those runs.

The number of time divisions is set to 500 during the computations; that is, the time length of $\frac{1}{36}$ s is further divided into 500 equal intervals resulting in a time step of $5.5 \times 10^{-5}$ s for the integration. The finally applied particle starting point density leads to $5 \times 10^5$ particles placed equidistantly along the 3 mm long test segment, implying a distance of $6 \times 10^{-9}$ m between the starting positions of these particles.

### 8.3 Results

#### 8.3.1 Qualitative results

First, to gain some qualitative overview I compile a qualitative image of the sensitivity to the initial position of the trajectories. Starting from a planar cross-section of the inlet section, whose position is shown in Fig. 8.1, four million randomly placed trajectory paths are computed. In Figure 8.6 the starting
Figure 8.4: The information dimension along the test line segment denoted 'b' in Figure 8.6 as a function of the number of the time divisions between two flow field snapshots.

Figure 8.5: The information dimension along the test line segment denoted 'b' in Figure 8.6 as a function of the number of trajectories initiated along the line segment.
position of each particle on the inlet cross-section is coloured according to the outlet through which the particle leaves the domain. The image shows a wild variation of the final positions of even those particles that started close to each other.

Figure 8.6: The cross-section of the inlet vessel from where the tracers were initiated. The colours represent the different outlets through which the particles (started from the given position) leave the domain. The corresponding outlet positions are shown in Fig. 8.1 with the same numbering. Colour 4 accounts for the particles still inside the domain after 20 s. The white line segments denote the locations for the computations of $D_1$.

Strong dependence of the outlet through which particles leave on the initial position, as shown in Fig. 8.6, is expected to imply similar variations in the residence times. Figure 8.2b shows the distribution of the residence times for the same particles followed in Fig. 8.6. One can see that a great number of particles leave the domain of the investigated vessel section quite rapidly. There are, however, particles that spend more than 20 s, i.e., 20 heartbeats in the domain. Note that I have filtered out from the shown distribution the particles stuck at the vessel wall during their advection. Hence, the particles that spend a very long time in the domain must be of those that are trapped in the
vicinity of the chaotic set. This set, though it consists of a zero measure set of unstable trapped orbits, is accessible from the direction of its stable manifold (Tél and Gruiz, 2006). The particles initially close to the stable manifold of the chaotic set can get close to the orbits of the chaotic set, and can follow them for long time before reaching any of the outlets. Physically, these are particles that “hesitate” for a long time which outlet to take eventually. The trajectories exactly on the stable manifold form the boundary between the basins of attraction of the outlets (Fig. 8.6). Initial conditions exactly on the boundary, on the stable manifold of the chaotic set, never leave the investigated domain (Tél and Gruiz, 2006).

In order to obtain information on the spatial distribution of the initial positions leading to long residence times, I measure the residence time for each initial point used in Fig. 8.6. Then, using colour coding in Fig. 8.7, I plot the residence time on the initial position.

![Figure 8.7](image)

**Figure 8.7:** Using colour coding, the residence times of the particles are indicated at their starting positions on the inlet cross-section. Though the calculation of residence times are done up to 20 cardiac cycles, these residence time values on the colour scale are cut at the upper limit of 6 s (6 cardiac cycles) to provide a better visual representation.
A portion of the trajectories with high residence times start from right beside the vessel wall. This is not surprising because blood velocity is low here due to the no-slip boundary condition on the vessel wall. Inside the vessel, however, we find starting points from where the tracer trajectories approach the vicinity of the chaotic set. At these points the residence times are much higher than 1.1 s, the time the particles would need to pass the investigated vessel segment, were they travelling at the average blood speed along the centreline of the blood vessel. The initial positions for particles with long residence times trace out intricate, filamentary patterns in Fig. 8.7. Such patterns have been observed previously in other fluid flows (Jung et al., 1993; Károlyi and Tél, 1997), where they turned out to be fractal patterns generated by the transient chaotic motion of the particles.

8.3.2 Quantitative results

I use the free-energy formalism to extract chaotic properties of mixing in the blood flow of the investigated vessel segment containing the aneurysm. I chose several line segments in the plane of the inlet cross-section shown in Figure 8.6. The locations of these line segments were selected so that they lie in the $x \in [0.056, 0.061], y \in [0.048, 0.052]$ region. Indeed, this part of the cross-section seems to be the richest in intricate patterns. Three of the selected line segments are parallel to the $x$ axis while the other three are perpendicular to it. The extracted quantities are shown in Table 8.1.

<table>
<thead>
<tr>
<th>line</th>
<th>$D_1$</th>
<th>Lyapunov ($\lambda$)</th>
<th>Escape rate ($\kappa$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>0.623</td>
<td>3.3841</td>
<td>1.3011</td>
</tr>
<tr>
<td>b</td>
<td>0.6025</td>
<td>4.4728</td>
<td>1.2156</td>
</tr>
<tr>
<td>c</td>
<td>0.6427</td>
<td>3.3882</td>
<td>0.7889</td>
</tr>
<tr>
<td>d</td>
<td>0.6446</td>
<td>2.2799</td>
<td>0.9486</td>
</tr>
<tr>
<td>e</td>
<td>0.6098</td>
<td>2.0528</td>
<td>0.8935</td>
</tr>
<tr>
<td>f</td>
<td>0.6143</td>
<td>2.6301</td>
<td>0.9918</td>
</tr>
</tbody>
</table>

Table 8.1: Quantities measured from the $\beta F(\beta)$ function along six different line segments indicated in Fig. 8.6.

Throughout the computation, the information dimension $D_1$ of the intersection of the stable manifold with the chosen line segment proved to be the most stable quantity. Based on these six measurements, its averaged value is $\overline{D_1} = 0.623$ with a standard deviation of $\sigma = 0.017$. The other quantities show much more spatial variability. Hence, I propose that from all the computed quantities the information dimension $D_1$ is the most robust parameter to describe mixing and chaotic advection in such diseased blood vessel segments.

There are several other points of instability in the flow apart from the close surroundings of the aneurysm where the flow profile changes rapidly, such
as the bifurcation points or high curvature sections of the blood vessel. As particles can also get trapped there, I also measured the residence time of particles starting from the same initial plane, but only until they reached the distal neck of the aneurysm. This way, the effect of further flow instabilities downstream of the aneurysm can be excluded from the residence time and the fractal characteristics.

The resulting residence time plot is shown in Fig. 8.8, which has the same main features as Fig. 8.7. I also measured the information dimension average $D'_{1}$ in this plot and it turned out to be within the standard deviation $\sigma$ from the previous average $D_1$ ($D'_{1} = 0.607$ and $\sigma' = 0.009$). Finally, yet another measurement downstream of the aneurysm neck was carried out but the residence time does not show such variation as that in Fig. 8.7, no fractal patterns emerged. This strongly suggests that the chaotic structure is mainly generated by the presence of the aneurysm and it is robust enough to suppresses the effect of other smaller instabilities along the inspected geometry. Naturally, these suppositions will have to be reinforced in the future by carrying out similar analyses on further aneurysmal geometries.

Figure 8.8: Residence time of the particles until they pass the neck of the aneurysm is indicated with colour coding on their initial position. The colour scale is the same as that in Fig. 8.7.
8.4 Conclusions

The aneurysm as a pathologic change in the vessel geometry causes a major disturbance in the flow field, creating vortices and possibly other instabilities. These instabilities modify the paths of some of the particles residing in the flow, resulting in an anomalous increase in their residence time. The modified paths in this case bear the characteristics of chaotic dynamics. Several quantities related to chaos, most importantly the information dimension, can be measured using the methods shown in this chapter. Though further measurements on different geometries will be needed to justify it, the results upon this geometry support the idea that the chaotic behavior inflicted by the aneurysm sac is quite dominant. In the current case the flow disturbances other than the aneurysm sac itself are not relevant in the measurement of $D_1$. This might make regular clinical measurements feasible as the actual nearby geometry and flow disturbance seems not to modify the trapping effect of the aneurysm. Several other interesting questions arise that require further investigation. It is possible that this chaotic behavior or even some of its statistical description (e.g., $D_1$) could be linked to the formation or rupture process of the aneurysms. For example, the increased residence times, efficient mixing, and exponential outflow of the particles in the diseased blood vessel segment can have direct effects on the biochemical reactions taking place in blood (Schelin, Károlyi, De Moura, et al., 2009; Schelin, Károlyi, A. P. de Moura, N. A. Booth, et al., 2010; Schelin, Károlyi, A. P. de Moura, N. Booth, et al., 2012). It would also be meaningful to collect statistical information about the relation of pathological vessel malformations and the fractal properties they enforce upon the flow. At the end, the fractal properties carry information on the aneurysm geometry in such a robust way that it seems useful to utilise them as an indicator during the medical decision-making and as a descriptor in the aneurysm classification process.

8.4.1 Thesis 5

In the validated flow field of a real aneurysm geometry I examined the trajectories of the massless tracer particles released into the flow. I found that these trajectories exhibit chaotic properties which are known to have several important biological and biochemical implications in blood flows. I measured the major characteristic quantities of the chaotic structure, such as the information dimension, the Lyapunov exponent, and the escape rate. In the examined case the information dimension proved to be the most robust measure which is definitely in strong connection with the geometrical properties of the aneurysm. [T4, T7]
Though each chapter describing the results of one of my experiments carry a concluding section focused on that specific topic, I think a few additional notes connecting these topics are in order. The results of the investigations presented in the previous two chapters open several possibilities for further research. In my opinion the study of a phenomenon can be considered successful if it provides some sensible answers and its value is ameliorated if it leads to further questions. The topics presented in this third part can be regarded as foundations for further investigations rather than closed chapters. For instance, the calculation of the platelet margination effect could be generalised to arbitrary suspensions that contain more than one particle size. It would be interesting to explore in greater detail the dependence of the force driving the particles out from the flow from the relative size and shape of the participating particles. As for the fractal dimensions of the particle trajectories in the aneurysmal geometry, it seems to be robust enough to utilise it as a possible descriptor of the geometrical properties. If it proves to be similar for further geometries (this is actually under investigation as of writing), it might also be used for statistically correlating the fractal dimension yielded by the different geometries with other medical attributes.


Abraham, E. R.

Alenitsyn, A., Kondratyev, A., Mikhailova, I., and Siddique, I.

Anand, M., Rajagopal, K., and Rajagopal, K.

Anderson, J. D. et al.

Ansumali, S.
2004 Minimal kinetic modeling of hydrodynamics, PhD thesis.

Ansumali, S. and Karlin, I. V.

Aref, H.

Austin, J. and Maceri, D.

Axner, L., Hoekstra, A. G., Jeays, A., Lawford, P., Hose, R., and Sloot, P. M.

Beer, F. P. and Johnston Jr, E. R.
Bernaschi, M., Melchionna, S., Succi, S., Fyta, M., Kaxiras, E., and Sircar, J. K.

Bhatnagar, P. L., Gross, E. P., and Krook, M.

Bisson, M., Bernaschi, M., Melchionna, S., Succi, S., and Kaxiras, E.
2012 “Multiscale Hemodynamics Using GPU Clusters”, *Communications in Computer Physics*, vol. 11, pp. 48-64.

Boghosian, B., Love, P., Coveney, P., Karlin, I., Succi, S., and Yepez, J.


Breuer, M., Bernsdorf, J., Zeiser, T., and Durst, F.

Canuto, C., Hussaini, M., Quarteroni, A., and Zang, T.
2010 *Spectral methods fundamentals in single domains*, Springer.

Castro, M. A., Olivares, M. C. A., Putman, C. M., and Cebral, J. R.

Cebral, J. R., Castro, M. A., Burgess, J. E., Pergolizzi, R. S., Sheridan, M. J., and Putman, C. M.

Celi, A., Merrill-Skoloff, G., Gross, P., Falati, S., Sim, D., Flaumenhaft, R., Furie, B., and Furie, B.

Chapman, S. and Cowling, T. G.

Chen, S. and Doolen, G. D.

D’Humières, D., Ginzburg, I., Krafczyk, M., Lallemand, P., and Luo, L.-S.

Eckhardt, B.

Epstein, I. R.

Falati, S., Gross, P., Merrill-Skoloff, G., Furie, B. C., and Furie, B.

Ferguson, G. G.

Furie, B. and Furie, B.
Ginzbourg, I. and d’Humières, D.

Ginzburg, I. and Steiner, K.

Gobin, Y., Counord, J., Flaud, P., and Duffaux, J.

Gonzalez, C. F., Cho, Y. I., Ortega, H. V., and Moret, J.

Guo, Z. and Shu, C.

Guo, Z., Zheng, C., and Shi, B.

Harris, S.

Harrison, S., Smith, S., Bernsdorf, J., Hose, D., and Lawford, P.


He, X. and Luo, L.-S.
He, X., Zou, Q., Luo, L.-S., and Dembo, M.

He, X., Duckwiler, G., and Valentino, D. J.

Hinze, J.

Inamuro, T., Yoshino, M., and Ogino, F.


Józsa, T. I. and Paál, G.

Jung, C., Tél, T., and Ziemniak, E.

Károlyi, G. and Tél, T.
1997 “Chaotic tracer scattering and fractal basin boundaries in a blinking vortex-sink system”, *Physics Reports*, vol. 290, 1, pp. 125-147.

Károlyi, G. and Tél, T.
2005 “Chemical transients in closed chaotic flows: the role of effective dimensions”, *Physical review letters*, vol. 95, 26, p. 264501.

Kerber, C. W., Imbesi, S. G., and Knox, K.

Khanafer, K. M., Bull, J. L., and Berguer, R.

Kiss, I., Merkin, J., and Neufeld, Z.

Kondo, S., Hashimoto, N., Kikuchi, H., Hazama, F., Nagata, I., and Kataoka, H.
1997 “Cerebral aneurysms arising at nonbranching sites an experimental study”, *Stroke*, vol. 28, 2, pp. 398-404.

Krüger, T., Varnik, F., and Raabe, D.

Kulcsár, Z., Ugron, A., Marosfői, M., Berentei, Z., Paal, G., and Szikora, I.

Lasheras, J. C.

Latt, J. and Chopard, B.


Noble, D. R., Chen, S., Georgiadis, J. G., and Buckius, R. O.

Noble, D. R., Georgiadis, J. G., and Buckius, R. O.

Ottino, J.

Ouared, R. and Chopard, B.

Oubel, E., De Craene, M., Putman, C. M., Cebral, J. R., and Frangi, A. F.
2007 “Analysis of intracranial aneurysm wall motion and its effects on hemodynamic patterns”, *Medical Imaging*, vol. 6511, 65112A.

Paál, G., Ugron, Á., Szikora, I., and Bojtár, I.

Piccinelli, M., Veneziani, A., Steinman, D. a., Remuzzi, A., and Antiga, L.

Pomeau, B. H. Y. and Frisch, U.

Qian, Y. H., D’Humières, D., and Lallemand, P.
Qian, Y., Takao, H., Umezu, M., and Murayama, Y.

Rinkel, G. J., Djibuti, M., Algra, A., and Van Gijn, J.

Schelin, A. B., Károlyi, G., De Moura, A., Booth, N., and Grebogi, C.

Schelin, A. B., Károlyi, G., de Moura, A. P., Booth, N. A., and Grebogi, C.

Schelin, A. B., Károlyi, G., de Moura, A. P., Booth, N., and Grebogi, C.

Sforza, D. M., Putman, C. M., and Cebral, J. R.


Shojima, M., Oshima, M., Takagi, K., Torii, R., Hayakawa, M., Katada, K., Morita, A., and Kirino, T.

Skordos, P.

Solomon, S.

Sonmez, G., Basekim, C. C., Ozturk, E., Gungor, A., and Kizilkaya, E.
Stehbens, W. E.

Steiger, H.-J.


Succi, S.

Suzuki, J. and Ohara, H.


Tadmor, E. B. and Miller, R. E.

Takao, H., Murayama, Y., Otsuka, S., Qian, Y., Mohamed, A., Masuda, S., Yamamoto, M., and Abe, T.
Tamagawa, M., Kaneda, H., Hiramoto, M., and Nagahama, S.

Taylor, C. A. and Draney, M. T.

Tél, T.

Tél, T., de Moura, A., Grebogi, C., and Károlyi, G.

Tél, T. and Gruiz, M.

Tokarev, A., Butylin, A., Ermakova, E., Shnol, E., Panasenko, G., and Ataullakhanov, F.

Toroczkai, Z., Károlyi, G., Pénét, Á., Tél, T., and Grebogi, C.

Ugron, Á., Farinas, M.-I., Kiss, L., and Paál, G.

Vlak, M. H., Algra, A., Brandenburg, R., and Rinkel, G. J.

Wang, W. and King, M. R.

Weir, B.
Wiebers, D. O.


William, F.
1985 *Combustion Theory: The fundamental theory of chemically reacting flow system.*


Wittmann, M., Zeiser, T., Hager, G., and Wellein, G.

Woldhuis, B., Tangelder, G., Slaaf, D., and Reneman, R. S.

Womersley, J. R.

Yu, D., Mei, R., Luo, L.-S., and Shyy, W.

Zanaty, M., Chalouhi, N., Tjoumakaris, S. I., Fernando Gonzalez, L., Rosenwasser, R. H., and Jabbour, P. M.


