Detection of Lattice-points inside Triangular Mesh for Variable-viscosity Model of Red Blood Cells

Tibor Poštek, František Kajánek and Mariana Ondrušová
Cell-in-Fluid Biomedical Modelling and Computations Group, Faculty of Management Science and Informatics, University of Žilina, Slovakia

Keywords: Deformable Objects, Lattice-Boltzmann Method, Variable Fluid Viscosity.

Abstract: In this work we introduce the extension of an existing computational model for red blood cells that enables modelling of different viscosity inside and outside the cells. The extension is based on an algorithm for detection of fluid lattice-points inside the cell given the membrane of the cell as a closed triangular mesh. This algorithm enables the setting of variable viscosity in the underlying lattice-Boltzmann method for computation of fluid dynamics.

1 INTRODUCTION

Computational models serve a wide variety of roles, including hypothesis testing, generating new insights, deepening understanding, suggesting and interpreting experiments, tracing chains of causation or performing sensitivity analyses. Computational modelling helps to predict outcomes in circumstances that are difficult to achieve in laboratory (Calder and et al., 2018). Model cannot replace experiment, but it can demonstrate whether or not a proposed mechanism is sufficient to produce an observed phenomenon. Model can be an effective tool in optimization and design (Janacek et al., 2017; Kleineberg et al., 2017).

Computational modelling helps to predict outcomes in circumstances that are difficult to achieve in laboratory (Calder and et al., 2018). It is an effective tool in optimization and design (Janacek et al., 2017; Kleineberg et al., 2017).

Computational modelling of deformable objects immersed in a flow such as biological cells inside microfluidic devices (Bušík et al., 2016; Gusenbauer et al., 2018; Jančigová and Cimrák, 2016; Cimrak, 2016; Reichel et al., 2019), requires capturing various bio-mechanical phenomena. One such example is different fluid viscosities inside and outside the cell. For example, red blood cells (RBC) are composed of cell membrane and the inner fluid called cytoplasm. Cytoplasm has four to five times higher viscosity than plasma surrounding red blood cells. Similar scenario holds for other cells or other environments, for example when various kinds of cells are manipulating in microfluidic devices.

Several computational models of RBC do not consider the difference between inner and outer fluid viscosities (Krueger, 2012; Dupin et al., 2007; Schiller et al., 2018) while others include this possibility (Doddi and Bagchi, 2009; Fedosov et al., 2010). There is an ongoing discussion about the necessity to consider variable viscosity, although it has been shown in (Fedosov et al., 2010), that different inner viscosity lead to different rotational behaviour of red blood cells in a shear flow.

The principal algorithmic task that is needed to be solved is to determine all spatial points in a three-dimensional space that lie within closed surface. In particular, we are interested only in those points with coordinates being integer numbers that are enclosed by a triangulation of a closed surface. The triangulation mesh-points can have non-integer coordinates. This specification comes from the fact that the fluid is discretized by a fixed lattice - therefore fluid nodes have integer coordinates, and the cell membrane is discretized by a triangular network.

This algorithmic task seems to be similar to task solved e.g. in Computational Geometry Algorithms Library project in a highly optimized manner. CGAL's particular package (Loriot et al., 2019) has been designed to handle huge triangular meshes and to manipulate them, for example find intersections of two meshes, or, decide whether a point in 3D space is inside or outside the mesh.

Our case is however different: We need to make such decision for all lattice points in space. There-

Poštek, T., Kajánek, F. and Ondrušová, M.
Detection of Lattice-points inside Triangular Mesh for Variable-viscosity Model of Red Blood Cells.
DOI: 10.5220/0009165102120217
In *Proceedings of the 13th International Joint Conference on Biomedical Engineering Systems and Technologies (BIOSTEC 2020) - Volume 3: BIOINFORMATICS*, pages 212-217
ISBN: 978-989-758-398-8; ISSN: 2184-4305

fore, the use of CGAL is quite limited because running over all lattice points we get cubic computational complexity. We further discuss the topic of complexity in Section 4.2.

In this work we provide a basic study of an extension of existing RBC model to include the possibility of various viscosities inside and outside the cell. In Section 2, we provide description of underlying model for the fluid and the notion of non-constant viscosity. Next in Section 3 we analyze in detail how non-constant viscosity is implemented in the lattice-Boltzmann model of fluid flow. In Section 4 we present our algorithm, with discussion on complexity.

2 NON-CONSTANT VISCOSITY IN MODELLING OF DEFORMABLE OBJECTS

We have been using the lattice-Boltzmann method (LBM) coupled with the spring network method for modelling of flow of deformable objects in a fluid (Cimrák et al., 2012; Cimrák et al., 2014). Here, the LBM governs the evolution of the fluid while the spring network is used to characterize the boundary of the immersed object. The LBM and the spring network is coupled by a dissipative coupling penalizing the difference between the velocities of the fluid and the spring network mesh points.

This method has been previously used for simulations of blood cells, having uniform fluid viscosity inside and outside the cell. In reality however, the inner fluid of red blood cells has four times higher viscosity than the surrounding fluid. Original implementation presented in (Cimrák et al., 2012; Cimrák et al., 2014) concerns the case of constant fluid viscosity.

We aim at introducing the possibility to account for variable viscosity. Therefore we will consider a spatial and time dependent kinematic viscosity $\nu_s(\vec{x},t)$ instead of the constant kinematic viscosity. From now on, if we mention viscosity, we mean kinematic viscosity of the fluid. Bearing in mind our field of application being the modelling of blood or tumour cells, we will consider $\nu_s(\vec{x},t)$ to be a multi valued piece-wise constant function for each time instance t. Since multiple cell types could be involved, we will work with possibility to have several different viscosities of the inner fluid of the cells, depending on the type of cells. For example, red blood cells have different viscosity of cytoplasm than white blood cells.

Definition of variable viscosity function will be

$$\nu_s(\vec{r},t) = \begin{cases} \nu_i, & \text{if } \vec{r} \text{ lies inside a cell of type } i \\ \nu_0, & \text{if } \vec{r} \text{ lies outside any cells} \end{cases} \tag{1}$$

where ν_i is the shear viscosity of the inner fluid of the cells of type i, and ν_0 is the shear viscosity of the outer fluid.

To justify the possibility to use variable fluid viscosity in LBM, we present several such cases when lattice-Boltzmann method has been used also for cases of non-constant fluid viscosity. Viscosity of the fluid may change for example with temperature, shear or concentration. Laminar convection of a fluid with a temperature-dependent viscosity in an enclosure filled with a porous medium was studied in (Guo and Zhao, 2005). The authors used lattice-Boltzmann method combined with heat transfer equation.

In (Wang and Ho, 2011) the authors use new lattice Boltzmann approach within the framework of D2Q9 lattice for simulating shear-thinning non-Newtonian blood flows described by the power-law. In (Chao et al., 2007) the authors consider variable viscosity complicating the truncation error analysis and creating additional interaction between the truncation error and the boundary condition error. In (Rakotomalala et al., 1996) the authors address the use of LBM for flows in which the viscosity is space and time dependent, namely for non-Newtonian fluids with shear dependent viscosity and miscible fluids with concentration dependent viscosity.

The case of multiple-relaxation-time lattice-Boltzmann method for incompressible miscible flow with large viscosity ratio and high Péclet number was studied in (Meng and Guo, 2015). Authors in (Krafczyk and Toelke, 2003) include Smagorinsky's algebraic eddy viscosity approach into the multiple-relaxation- time lattice Boltzmann equation for large-eddy simulations of turbulent flows. The inclusion of variable viscosity is enabled by three-dimensional multi-relaxation time lattice-Boltzmann models for multi-phase flow presented in (Premnath and Abraham, 2007).

3 LATTICE-BOLTZMANN METHOD IN THE COMPUTATIONAL MODEL

After we justified the possibility to use variable viscosity in the LBM we looked closely on how to implement the variable viscosity into the LBM. We work with LBM described in (Dunweg and Ladd, 2009; Weik et al., 2019). This model has been implemented in ESPResSo (Arnold et al., 2013) which is a scientific package that we have been using. In ESPResSo, a three-dimensional lattice-Boltzmann model is implemented where velocity space is discretized in 19 velocities (D3Q19 model). The details about this model are available in (Dunweg and Ladd, 2009). The collision step is performed after a transformation of the populations into mode space. This allows implementation of the multiple relaxation time (MRT) lattice-Boltzmann method (Lallemand et al., 2003). It also allows the use different relaxation times for different modes, thus to adjust the bulk and shear viscosity of the fluid separately.

Bulk vs Shear Viscosity. Common notion of viscosity refers mostly to shear viscosity. Shear viscosity is the resistance of the fluid against shearing motion. It is present e.g. in the Couette flow: Two parallel plates are moving in the opposite directions generating shear flow. The layers of liquid in different heights are moving with different velocities, the friction between them arises that generate the resistance against such movement. This resistance is called shear viscosity.

Bulk viscosity refers to resistance of the fluid against shear-less compression or expansion of the fluid. For incompressible liquids the bulk viscosity does not enter the hydrodynamic equations at all.

Mode Space. The principle of the mode space is that populations are transformed into modes. There are 19 modes, denoted by indexes 0 to 18. Modes 0-3 correspond to the conserved variables, mass density ρ, and momentum density **j**. Moments 4-9 correspond to bulk and shear stresses. These moments also affect the viscosity of the fluid. The higher modes 10-18 represent kinetic modes, they do not enter bulk hydrodynamic equations, but they play a role near boundaries and in thermal fluctuations.

For the implementation of relaxation time, there are two possibilities in ESPResSo: The multiple-relaxation-time or two-relaxation-time (TRT). The TRT model is described in (Chun and Ladd, 2007). These two possibilities have two different sets of eigenvectors and eigenvalues of the collision operator. They are shown in Table 1 in (Chun and Ladd, 2007). We will consider the MRT case. Here, the eigenvalues for the first four modes are zero, for mode 4 we have eigenvalue γ_b corresponding to dynamic bulk viscosity μ_b and for modes 5-9 we have eigenvalue γ_s corresponding to dynamic shear viscosity μ_s. Exact relation between eigenvalues and viscosities are given by Equation 80 and 81 from (Dunweg et al., 2007)

$$\mu_s = \frac{\tau\rho c_s^2}{2}\frac{1+\gamma_s}{1-\gamma_s}$$

$$\mu_b = \frac{\tau\rho c_s^2}{d}\frac{1+\gamma_b}{1-\gamma_b}$$

Here, ρ is the fluid density, τ is the time step and c_s is the speed of sound in the LB lattice, namely

$$c_s = \frac{1}{\sqrt{3}}\frac{b}{\tau}$$

where b is the lattice spacing. So when the γ_s needs to be evaluated for given viscosity, we plug expression for c_s to get

$$\mu_s = \frac{\tau\rho}{2}\frac{b^2}{3\tau^2}\frac{1+\gamma_s}{1-\gamma_s} = \frac{\rho b^2}{6\tau}\frac{1+\gamma_s}{1-\gamma_s}$$

$$\frac{1+\gamma_s}{1-\gamma_s} = \frac{6\tau\mu_s}{\rho b^2}$$

If kinematic viscosity $\nu_s = \mu_s/\rho$ is used, the fluid density ρ vanishes from the expressions

$$\frac{1+\gamma_s}{1-\gamma_s} = \frac{6\tau\nu_s}{b^2}$$

After some computations, one gets

$$\gamma_s = \frac{\frac{6\tau\nu_s}{b^2}-1}{\frac{6\tau\nu_s}{b^2}+1} = 1-\frac{2}{\frac{6\tau\nu_s}{b^2}+1}$$

which is exactly the relation implemented in a `lb.cpp` file of the `lb_reinit_parameters()` function in the ESPResSo core code.

Eventually, we have localized the relation that tells us dependence of γ_s and ν_s and at this place in the code we need to set different viscosities for the inner fluid of the cells and outer fluid.

4 ALGORITHMS FOR DETECTION OF CELL INNER SPACE

Practical question is how to determine whether you are inside or outside the cell. The fluid in LBM is discretized by a fixed perpendicular 3D grid. For ease of explanation, assume this grid has spacing 1. The cell on the other hand is discretized by a triangular mesh that covers its surface. This mesh is not linked to the grid, it can freely deform according to the mechanics of the membrane. So the problem can be stated in the following way:

Find all points in 3D with coordinates being natural numbers that are enclosed inside a given triangular network.

We have developed efficient algorithms that solve this problem and we have described them in detail in (Poštek, 2019). The basic idea of the algorithm is to loop over all horizontal lines parallel to x axis direction with coordinates being natural numbers. For each such line, we note an ”entering” mark whenever we detect crossing with any triangle from the cell’s mesh, meaning we are coming from the outside into the inside of the cell. When the second such mark occurs, this means we are leaving the cell and we mark it as ”exiting” mark.

After such loop we just loop again over horizontal lines and all the grid points that lie between two ”entering” and ”exiting” marks are flagged to be inside the cell. All other grid points are automatically outside the cell.

4.1 Algorithm Details

To find an intersection of all lines along x axis direction with remaining two coordinates natural numbers, we do it in an loop over y-coordinate. For a fixed y-coordinate we need to find all intersections of mesh triangles with plane with the same y-coordinate and parallel to x and z axis, we denote this plane as Y-plane. In Figure 1 you can see the visualization of such intersection.

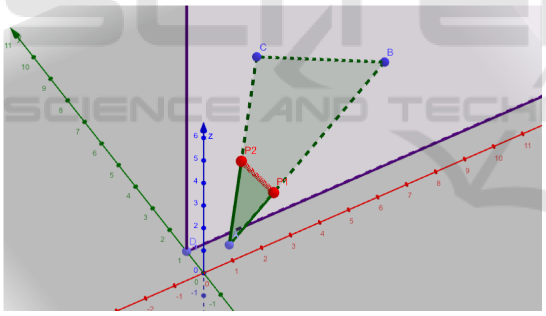

Figure 1: Points P_1 and P_2 characterize the intersection of a triangle with Y-plane.

After identification of segment P_1P_2 that is the intersection of a triangle ABC (one of the mesh triangles) with Y-plane, we need to detect the intersections of all horizontal lines from Y-plane that have z-coordinate a natural number, see Figure 2

After obtaining the ”entering” and ”exiting” marks on each horizontal line with natural y and z coordinates, the loop over all lines will flag the individual grid points as inner or outer grid points, see Figure 3.

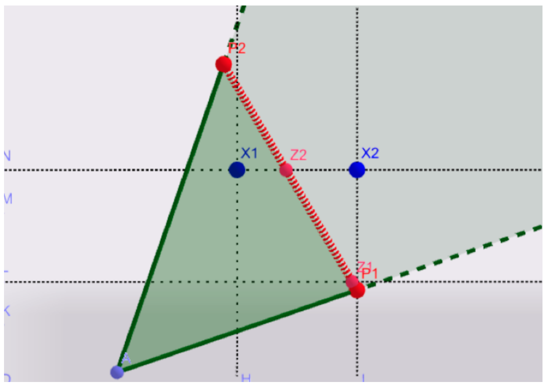

Figure 2: Segment P_1P_2 is the intersection of triangle with plane $y = integer\ number$. Points Z_1, Z_2 are those with integer $z-$coordinate and those are the intersections of horizontal lines with segment P_1 and P_2. Points X_1, X_2 are the points on horizontal lines that have integer $x-$coordinate. These points are either ”entering” or ”exiting” mark for given horizontal line, depending on the orientation of triangle ABC.

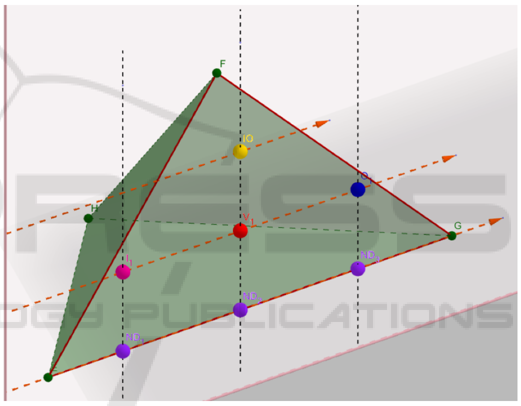

Figure 3: Variants of marking inside LB nodes.

4.2 Computational Efficiency

Consider a cubic computational domain with its edge divided into n points. This leads to n^3 fluid lattice points. In case of a single object (one cell in a simulation), we need to decide about all n^3 points, whether they are inside or outside the object described by a mesh with N triangles. In our algorithm, we first run over N triangles, and we create a list of entering and exiting marks for each horizontal line. Since the number of horizontal lines in n^2, this gives us $O(N \times n^2)$ operations. Afterwards, we run over all lattice points with $O(n^3)$ complexity, and we mark them as inside or outside points according to entering and exiting marks: If they lie between entering and exiting mark, we mark them inside lattice points, otherwise we mark them as outside lattice points. Both operations result in combined $O(N \times n^2) + O(n^3)$ complexity.

To compare our algorithm to the established CGAL computational library: In CGAL, there is an available package Polygon Mesh Processing and its class CGAL::Side_of_triangle_mesh that can decide for a given point in space whether it lies inside or outside the triangular mesh. In order to mark all lattice points as inside or outside points, we need to run the functor of the class CGAL::Side_of_triangle_mesh n^3 times. The complexity of the class is $O(N)$, since it needs to run over all triangles. Therefore, combined complexity of CGAL algorithm is $O(n^3 \times N)$.

To wrap up, our algorithm with $O(n^3 + N \times n^2)$ complexity is more efficient than CGAL algorithm with $O(N \times n^3)$ complexity.

Speed-up

The presented algorithm has complexity $O(n^3 + N \times n^2)$, assuming the computational domain is a cube with n discretization points on its edge. This might still be too demanding for computations and if this algorithm is executed in each time step, the computations will be dramatically slowed down.

To spare time, we propose to run this algorithm in the beginning of the simulation to detect the cells' interiors. Afterwards, once in a hundred time steps, the algorithm can be executed again. Meanwhile, in each time step a more sophisticated *update algorithm* will be executed. The update algorithm will not run over all horizontal lines and over all mesh points on the lines, but it will only reflag the nearest grid points near the mesh triangles. The update algorithm will loop over all triangles in the mesh - which makes it $O(n^2 \times N)$ complex - and after detecting the points in the triangle with y and z coordinates natural numbers, one can reflag the neighbouring grid points with all coordinates natural numbers. This update algorithm will be useful only in scenarios when during one time step, the triangles from the mesh do not jump over more than one grid point from the fluid discretization. But in most simulations, the cell deforms and moves slow enough so that it does not skip several fluid grid points in one time step.

5 CONCLUSIONS AND OUTLOOK

The presented algorithm enables the use of various viscosity in an existing implementation of blood cell modelling in ESPResSo. There must be several steps done before claiming this algorithm is correct and efficient and before it can be used by the scientific community:

- Computational analysis of computational costs of the algorithm. One must be sure that the introduction of this algorithm to LBM method will not slow the computations down rapidly.
- The whole implementation must be tested against experimental and analytical data. The work of (Feng and Michaelides, 2001) for example includes tables of drag coefficients of viscous spheres in a uniform flow. This could be simulated and the computational drag coefficient compared to those from (Feng and Michaelides, 2001).

ACKNOWLEDGEMENTS

This work was supported by the Slovak Research and Development Agency (contract number APVV-15-0751) and by the Ministry of Education, Science, Research and Sport of the Slovak Republic (contract number VEGA 1/0643/17).

REFERENCES

Arnold, A., Lenz, O., Kesselheim, S., Weeber, R., Fahrenberger, F., Roehm, D., Košovan, P., and Holm, C. (2013). ESPResSo 3.1 - molecular dynamics software for coarse–grained models. In Griebel, M. and Schweitzer, M., editors, *Meshfree Methods for Partial Differential Equations VI, Lecture Notes in Computational Science and Engineering*, volume 89, pages 1–23.

Bušík, M., Jančigová, I., Tóthová, R., and Cimrák, I. (2016). Simulation study of rare cell trajectories and capture rate in periodic obstacle arrays. *Journal of Computational Science*, 17(2):370–376.

Calder, M. and et al. (2018). Computational modelling for decision-making: where, why, what, who and how. *R Soc Open Sci*, 5(6).

Chao, J., Mei, R., and Shyy, W. (2007). Error assessment of lattice boltzmann equation method for variable viscosity flows. *International journal for numerical methods in fluids*, 53:1457–1471.

Chun, B. and Ladd, A. (2007). Interpolated boundary condition for lattice boltzmann simulations of flows in narrow gaps. *Physical Review E*, 75:066705.

Cimrak, I. (2016). Collision rates for rare cell capture in periodic obstacle arrays strongly depend on density of cell suspension. *Computer Methods in Biomechanics and Biomedical Engineering*, 19(14):1525–1530.

Cimrák, I., Gusenbauer, M., and Jančigová, I. (2014). An ESPResSo implementation of elastic objects immersed in a fluid. *Computer Physics Communications*, 185(3):900–907.

Cimrák, I., Gusenbauer, M., and Schrefl, T. (2012). Modelling and simulation of processes in microfluidic devices for biomedical applications. *Computers and Mathematics with Applications*, 64(3):278–288.

Doddi, S. and Bagchi, P. (2009). Three-dimensional computational modeling of multiple deformable cells flowing in microvessels. *Phys. Rev. E*, 79(4):046318.

Dunweg, B. and Ladd, A. J. C. (2009). Lattice-Boltzmann simulations of soft matter systems. *Advances in Polymer Science*, 221:89–166.

Dunweg, B., Schiller, U., and Ladd, A. J. C. (2007). Statistical mechanics of the fluctuating lattice boltzmann equation. *Physical Review E??*, 76:036704.

Dupin, M. M., Halliday, I., Care, C. M., and Alboul, L. (2007). Modeling the flow of dense suspensions of deformable particles in three dimensions. *Physical Review E, Statistical, nonlinear, and soft matter physics*, 75:066707.

Fedosov, D. A., Caswell, B., and Karniadakis, G. E. (2010). A multiscale red blood cell model with accurate mechanics, rheology and dynamics. *Biophysical Journal*, 98(10):2215–2225.

Feng, Z. G. and Michaelides, E. E. (2001). Drag coefficients of viscous spheres at intermediate and high reynolds numbers. *Journal of Fluids Engineering*, 123:841–849.

Guo, Z. and Zhao, T. (2005). Lattice boltzmann simulation of natural convection with temperature-dependent viscosity in a porous cavity. *Progress in Computational Fluid Dynamics*, 5(1):110–117.

Gusenbauer, M., Tóthová, R., Mazza, G., Brandl, M., Schrefl, T., Jančigová, I., and Cimrák, I. (2018). Cell damage index as computational indicator for blood cell activation and damage. *Artificial Organs*, 42(7):746–755.

Janacek, J., Kohani, M., Koniorczyk, M., and Marton, P. (2017). Optimization of periodic crew schedules with application of column generation method. *Transportation Research Part C: Emerging Technologies*, 83:165 – 178.

Jančigová, I. and Cimrák, I. (2016). Non-uniform force allocation for area preservation in spring network models. *International Journal for Numerical Methods in Biomedical Engineering*, 32(10):e02757–n/a.

Kleineberg, K.-K., Buzna, L., Papadopoulos, F., Boguñá, M., and Serrano, M. A. (2017). Geometric correlations mitigate the extreme vulnerability of multiplex networks against targeted attacks. *Phys. Rev. Lett.*, 118:218301.

Krafczyk, M. and Toelke, J. (2003). Large-edd simulations with multiple-relaxation time lbe model. *International Journal of Modern Physics B*, 17(1):33–39.

Krueger, T. (2012). *Computer Simulation Study of Collective Phenomena in Dense Suspensions of Red Blood Cells under Shear*. PhD thesis, Max-Planck Institut.

Lallemand, P., d'Humieres, D., Luo, L., and Rubinstein, R. (2003). Theory of the lattice boltzmann method: three-dimensional model for linear viscoelastic fluids. *Phys. Rev. E*, 67:021203.

Loriot, S., Rouxel-Labbé, M., Tournois, J., and Yaz, I. O. (2019). Polygon mesh processing. In *CGAL User and Reference Manual*. CGAL Editorial Board, 5.0 edition.

Meng, X. and Guo, Z. (2015). Multiple-relaxation-time lattice boltzmann model for incompressible miscible flow with large viscosity ratio and high péclet number. *PHYSICAL REVIEW E*, 92:043305.

Poštek, T. (2019). Fluid dynamics model of cells with variable inner and outer fluid viscosity. In *Mathematics in science and technologies [print] : proceedings of the MIST conference 2019*, pages 65–72.

Premnath, K. and Abraham, J. (2007). Three-dimensional multi-relaxation time (mrt) lattice-boltzmann models for multiphase flow. *Journal of Computational Physics*, 224:539–559.

Rakotomalala, N., Salin, D., and Watzky, P. (1996). Simulations of viscous flows of complex fluids with a bhatnagar, gross, and krook lattice gas. *Physics of Fluids*, 8:3200–3202.

Reichel, F., Mauer, J., Nawaz, A. A., Gompper, G., Guck, J., and Fedosov, D. A. (2019). High-throughput microfluidic characterization of erythrocyte shapes and mechanical variability. *Biophysical Journal*, 117(1):14 – 24.

Schiller, U. D., Krüger, T., and Henrich, O. (2018). Mesoscopic modelling and simulation of soft matter. *Soft Matter*, 14:9–26.

Wang, C. and Ho, J. (2011). A lattice boltzmann approach for the non-newtonian effect in the blood flow. *Computers and Mathematics with Applications*, 62:75–86.

Weik, F., Weeber, R., Szuttor, K., Breitsprecher, K., de Graaf, J., Kuron, M., Landsgesell, J., Menke, H., Sean, D., and Holm, C. (2019). Espresso 4.0 – an extensible software package for simulating soft matter systems. *The European Physical Journal Special Topics*, 227(14):1789–1816.