

PARALLEL ALGORITHMS FOR DIRECT BLOOD FLOW SIMULATIONS

A Thesis
Presented to
The Academic Faculty

by

Abtin Rahimian

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Computational Science and Engineering

Georgia Institute of Technology
May, 2012

PARALLEL ALGORITHMS FOR DIRECT BLOOD FLOW SIMULATIONS

Approved by:

Professor George Biros, Advisor
School of Computational Science and
Engineering
Georgia Institute of Technology

Professor Silas Alben
School of Mathematics
Georgia Institute of Technology

Professor Alberto Fernandez-Nieves
School of Physics
Georgia Institute of Technology

Professor David Hu
School of Mechanical Engineering
Georgia Institute of Technology

Professor Richard Vuduc
School of Computational Science and
Engineering
Georgia Institute of Technology

Date Approved: January 27, 2012

ACKNOWLEDGEMENTS

I am deeply grateful to my advisor Dr. George Biros for his guidance, support, and encouragement throughout the years of my graduate studies. His insight, enthusiasm, and dedication to research are everlasting sources of inspiration for me. I am grateful to Dr. Denis Zorin and Dr. Shravan Veerapaneni for numerous insightful discussions over the course of our fruitful collaborations. I am thankful to Dr. Chaouqi Misbah and Dr. Giovanni Ghigliotti for sharing their insight and a wonderful collaboration. I wish to thank Dr. Ilya Lashuk, Aparna Chandramowlishwaran, Dhairya Malhotra, Logan Moon, Dr. Rahul Sampath, Aashay Shringarpure, Dr. Jeffrey Vetter, and Dr. Richard Vuduc with whom I share the Gordon Bell prize for our work on large scale simulation of blood flow. I would like to thank my thesis committee members, Dr. Silas Alben, Dr. Alberto Fernandez-Nieves, Dr. David Hu, and Dr. Richard Vuduc, for their encouragement and insightful comments.

I am thankful to all members of the Computational Science and Engineering department at Georgia Institute of Technology and the members of the department of Mechanical Engineering and Applied Mechanics at the University of Pennsylvania for providing a great environment for research.

Finally, I would like to express my deepest gratitude to my parents who gave me their unqualified support.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS	iii
LIST OF TABLES	viii
LIST OF FIGURES	ix
SUMMARY	xi
I INTRODUCTION	1
1.1 Background	2
1.2 Numerical Model	5
1.3 Contributions	7
1.4 Outline of The Thesis	12
II DYNAMIC SIMULATION OF LOCALLY INEXTENSIBLE VESICLES SUSPENDED IN AN ARBITRARY TWO-DIMENSIONAL DOMAIN, A BOUNDARY INTEGRAL METHOD	14
2.1 Introduction	14
2.1.1 Contributions	16
2.1.2 Limitations	18
2.1.3 Related work	18
2.1.4 Synopsis of Our Method	19
2.1.5 Nomenclature	20
2.2 Formulation	20
2.2.1 Boundary Integral Formulation	22
2.2.2 Completed Indirect Solution of Stokes Flow in a Confined Geometry	24
2.2.3 Traction Jump Across the Interface	25
2.2.4 Scaling	26
2.2.5 Summary of the Vesicles' Equations of Motion in Two-dimensional Confined Geometries	27
2.3 Numerical Algorithms	27
2.3.1 Time Discretization	27
2.3.2 Spatial Discretization	30
2.3.3 Analysis of the Spectral Properties	31

2.3.4	Computational Cost of the Semi-implicit Scheme for a Single Time-step	32
2.4	Numerical Experiments	33
2.4.1	Single Vesicle	33
2.4.2	Multiple Vesicles	37
2.4.3	Vesicles in Confined Domains	40
2.4.4	Rheology of a Suspension	42
2.5	Conclusions	45
III	VESICLE MIGRATION AND SPATIAL ORGANIZATION DRIVEN BY FLOW LINE CURVATURE	47
3.1	Introduction	47
3.2	Methods	48
3.3	Numerical Experiments	49
3.4	Results	50
IV	A FAST ALGORITHM FOR SIMULATING VESICLE FLOWS IN THREE DIMENSIONS	55
4.1	Introduction	55
4.1.1	Contributions	58
4.1.2	Limitations	60
4.1.3	Outline of the Chapter	61
4.2	Related Work	61
4.3	Formulation	66
4.4	Spatial Scheme	67
4.4.1	Singular Integration	70
4.5	Time Scheme	76
4.5.1	Spectral Analysis on the Unit Sphere	77
4.5.2	Explicit Scheme	81
4.5.3	Semi-implicit Scheme	83
4.5.4	Preconditioners	85
4.6	Reparametrization	86
4.7	Results	91
4.7.1	Derivative Accuracy	91

4.7.2	Accuracy of the Numerical Integration Schemes	92
4.7.3	Reparametrization	93
4.7.4	Stability of the Time-marching Scheme	95
4.7.5	Simulations in Presence of Gravity	96
4.7.6	Multiple Vesicles	97
4.8	Geometric Formulas	99
V	THREE-DIMENSIONAL BOUNDARY INTEGRAL METHOD FOR THE FLOW OF VESICLES WITH VISCOSITY CONTRAST	103
5.1	Introduction	103
5.1.1	Related Work	106
5.1.2	Nomenclature	107
5.2	Mathematical Formulation	108
5.3	Numerical Algorithms	111
5.3.1	Spatial Representation	111
5.3.2	Galerkin Formulation	122
5.3.3	Time Discretization	124
5.3.4	Solution Schemes	125
5.3.5	Preconditioning	131
5.4	Overall Algorithm and Computational Complexity	131
5.5	Conclusions	133
VI	PETASCALE DIRECT NUMERICAL SIMULATION OF BLOOD FLOW ON 200K CORES AND HETEROGENEOUS ARCHITECTURES . .	135
6.1	Introduction	135
6.1.1	Synopsis of Our Approach	137
6.1.2	Contributions	139
6.1.3	Related Work	140
6.2	Formulation and Algorithms for Particulate Flows	141
6.2.1	Overall Algorithm and the Main Computational Kernels	143
6.2.2	Global Interactions: FMM Kernel	144
6.2.3	Local Interactions: RBC Physics Kernels	146
6.3	Scalability Results	151

6.3.1	Single Node Experiments	152
6.3.2	MPI, Strong Scalability Tests on Jaguar	153
6.3.3	MPI, Weak Scalability Tests on Jaguar	154
6.3.4	GPU Weak Scalability Results for FMM on Lincoln	155
6.3.5	Red Blood Cell Distributions and Background flow	156
6.4	Conclusions	158
VII FUTURE RESEARCH DIRECTIONS		159
REFERENCES		161

LIST OF TABLES

1	Composition of cellular phase of human blood	3
2	Index of frequently used symbols and operators in Chapter 2	21
3	Backward difference coefficients	28
4	Stable step size for a second-order semi-implicit scheme	34
5	Stable step size for the first-order explicit scheme	35
6	Number of GMRES iteration averaged over 100 time steps	37
7	The relative error for area and perimeter of a vesicle in shear flow	38
8	Stable step size for a second-order semi-implicit scheme for two vesicles in shear flow	38
9	Relative error for area and volume of a single vesicle in the constricted tube	43
10	Stable step size for a second-order semi-implicit scheme for a vesicle in a constricted tube	43
11	Average number of GMRES iterations for the tension solver	86
12	Average number of GMRES iterations for the position solver in the semi-implicit scheme	86
13	Relative errors in computing smooth and single-layer integrals	93
14	Stable time step for a vesicle in shear flow	95
15	Relative errors for the sedimentation simulation	98
16	Geometric formulas	102
17	Index of frequently used symbols and operators in Chapter 5	108
18	The upsampling rate for differentiation	119
19	Convergence of discrete double-layer integral	121
20	Stable time step for vesicles with viscosity contrast	130
21	The computational cost for a single vesicle	130
22	Index of frequently used symbols and operators in Chapter 6	142
23	Execution time for rotation of the pole	149

LIST OF FIGURES

1	The shape of different blood cells at rest	4
2	Giant vesicles	5
3	Summary of the computational infrastructure for direct numerical simulation of blood flow	8
4	Numerical simulation of vesicles in the Couette apparatus	17
5	Schematic of the two dimensional domain	22
6	The evolution of a single vesicle with reduced area $\Delta = 0.75$ in an unbounded shear flow	35
7	The evolution of a single vesicle with reduced area $\Delta = 0.2$ in an unbounded shear flow	36
8	The sedimentation shape for a vesicle with different viscosity contrasts . . .	36
9	Tank-treading angle as a function of shear rate and viscosity contrast . . .	39
10	Collision of vesicles in shear flow	40
11	The lateral offset of vesicles in shear flow over time	41
12	Evolution of a single vesicle in a constricted tube	42
13	The intrinsic viscosity of the homogeneous fluid vs. viscosity contrast of the suspension for different values of reduced area Δ	45
14	Force distribution on vesicle membrane and local coordinate system	49
15	Trajectory of a vesicle in an unbounded rotational flow	50
16	Normalized inward migration velocity	51
17	Normalized inward migration velocities divided by normal stress difference .	51
18	Spatial organization of vesicles in the Couette apparatus	53
19	Snapshots of twenty vesicles suspended in a simple shear flow	57
20	Aliasing error in calculation of curvature	70
21	Relative errors in computing the principal curvatures	92
22	Effect of reparametrization	94
23	Single vesicle in shear flow	95
24	Terminal shapes of vesicles in a simple shear flow for various shear rates . .	97
25	Snapshots of the sedimentation of ellipsoidal vesicles	98
26	Relative error in the evaluation of nearly-singular integrals	99

27	Two vesicles in shear flow	100
28	Examples of Vesicle flow simulations	101
29	The schematic of the domain	109
30	Aliasing for associated Legendre polynomials	116
31	The Spectrum of the Mean Curvature	118
32	Cross sections of a biconcave vesicle in the Poiseuille flow	120
33	Accuracy of the locally implicit method for two vesicles	127
34	Two vesicles with viscosity contrast	128
35	Summary of the computational infrastructure for direct numerical simulation of blood flow	137
36	An example of global and local interactions	143
37	The typical biconcave RBC shape	148
38	Reparameterization of RBC	151
39	Single node scaling for local interactions	153
40	Single node scaling results for the global interactions	154
41	Strong scaling on Jaguar PF	154
42	Weak Scaling on Jaguar PF	155
43	GPU weak scaling	156
44	Snapshots of the simulation of 40,000 RBCs	157

SUMMARY

Fluid mechanics of blood can be well approximated by a mixture model of a Newtonian fluid and deformable particles representing the red blood cells. Experimental and theoretical evidence suggests that the deformation and rheology of red blood cells is similar to that of phospholipid vesicles. These vesicles have several features that also appear in red blood cells: they are area preserving closed membranes that resist bending. Beyond red blood cells, vesicles can be used to investigate the behavior of cell membranes, intracellular organelles, and viral particles. Given the importance of vesicle flows, in this thesis we focus in efficient numerical methods for such problems: we present computationally scalable algorithms for the simulation of dilute suspension of deformable vesicles in two and three dimensions.

To design efficient methods for vesicle flows, we use the following three observations: (i) we are interested in very low Reynolds number flows; (ii) we assume the interior and exterior fluid to be Newtonian; (iii) we do not consider topological changes and the surface of the vesicles in our scheme is smooth and homeomorphic to a sphere. Given this setup our best option for a numerical scheme is to use a boundary integral formulation. This formulation results in a set of nonlinear integro-differential equations for the vesicle dynamics. The motion of the vesicles is determined by balancing the non-local hydrodynamic forces with the elastic forces due to bending and tension.

We present new schemes for simulating the three-dimensional hydrodynamic interactions of large number of vesicles with viscosity contrast. The algorithms incorporate a stable time-stepping scheme, high-order spatiotemporal discretizations, spectral preconditioners, and a reparametrization scheme capable of resolving extreme mesh distortions in dynamic simulations. The associated linear systems are solved in optimal time using spectral preconditioners. The highlights of our numerical scheme are that (i) the physics of

vesicles is faithfully represented by using nonlinear solid mechanics to capture the deformations of each cell, (ii) the long-range, N-body, hydrodynamic interactions between vesicles are accurately resolved using the fast multipole method (FMM), and (iii) our time stepping scheme is unconditionally stable for the flow of single and multiple vesicles with viscosity contrast and its computational cost-per-simulation-unit-time is comparable to or less than that of an explicit scheme. We report scaling of our algorithms to simulations with millions of vesicles on thousands of computational cores.

CHAPTER I

INTRODUCTION

We consider fast and scalable solvers for the dynamics of particulate flows with deformable particles suspended in viscous incompressible fluids. Such flows are commonly found in biological systems, the most important example of which is blood flow in capillaries. The direct numerical simulation of blood flow in arterioles, venules, and capillaries would significantly broaden our understanding of the microcirculation of blood. This understanding is in turn crucial in thrombosis risk assessment, anti-coagulation therapy, and the design of microfluidic devices. For instance, high-fidelity numerical models can be employed to (i) estimate the rheological properties of the blood, (ii) evaluate the local fluid stresses under different flow regimes, (iii) analyze the formation, growth, and evolution of blood clots under different scenarios, and (iv) facilitate the optimal design of the microfluidic chips. Optimizing microfluidic device is an expensive and time-consuming process involving exhaustive experimentation. We envision computational platforms for chip design and optimization that will mitigate these problems. However, the simulation of complex fluids in microfluidic devices are computationally challenging tasks as they involve large-scale particulate suspensions, arbitrary confined geometries and multi-scale physics. Despite the significance and broad application of high-fidelity numerical tools of microcirculation, until recently, prohibitive computational cost has hampered their development [98].

To address the numerical challenges and to provide scalable and efficient computational tools, we present a fast, petaflop-scalable infrastructure for Stokesian particulate flows in two and three dimensions. Our infrastructure improves the state-of-the-art by several orders of magnitude in term of the number of cells: the previous largest simulation, at the same physical fidelity as ours, resolved the flow of $\mathcal{O}(10,000)$ Red Blood Cells (RBCs). In our scalability studies, we tested our code on a few time steps of the blood flow involving 262 million deformable RBCs, which corresponds to a dynamical system with 90 billion

unknowns in space.

Next, we give a very brief background on the subject of numerical hemodynamics, and then, we outline our model, our contributions, and their limitations.

1.1 Background

The subject of hemodynamics consists of distinct divisions depending on the ultimate physiological phenomenon under consideration. The two major divisions are (i) the flow of blood in large arteries and veins, where the particulate nature of the suspension is a secondary consideration; and (ii) the microcirculation, where the diameter of blood vessels is comparable to that of the blood cells. In this regime, the behavior of individual blood cells, the cell-cell interactions, and cell-wall interactions all play significant roles in defining the characteristic of the flow. Clinical needs in thrombosis risk assessment, anti-coagulation therapy, and stroke research would significantly benefit from an improved understanding of the microcirculation of blood.

In this context, theoretical modeling is used to quantitatively reconcile experimental observations and hypothesized underlying mechanisms [78]. Also, modeling is a valuable tool for rationalizing fragmented understandings under a common quantitative framework. This is especially valuable for large systems where there are many interacting components. An important outcome of theoretical modeling of blood flow is the estimation of unmeasurable parameters, for instance, local fluid stresses. Theoretical models also facilitate the study of pathological cases, impossible to produce or isolate in vivo or in vitro [78].

Nonetheless, the excessive analytical demand and prohibitive computational cost have hampered the theoretical study of microcirculation. Consequently, most of the analytical and computational studies were limited to systems with a few blood cells and have relied on simplifying assumptions such as non-deformable and nearly spherical of blood cells, axisymmetric configurations, and periodic flows. Such assumptions however are incompatible with a wide spectrum of phenomena observed in real flows and thus, limit the capabilities of the model to capture the characteristic cell behaviors such as flipping and tank-treading [16, 69]. During the past decade, elimination of these assumptions from the

models was a major driving force in the research community and resulted in several advances [16, 35, 66, 103, 104, 106, 152, 155, 156].

Theoretical models of blood flow in capillaries typically require three major components: (i) a mathematical description of the cell deformation; (ii) representation of the fluid flow around and within the cell; and (iii) the effect of the vessel wall. The composition of the blood is a deciding factor in the choice of mathematical model and the governing equations for the fluid flow. The blood of a healthy human is mainly composed of erythrocytes, leukocytes, and thrombocytes that are suspended in the plasma. By volume, the RBCs and plasma constitute about 45 and 54.3 percent of the blood, respectively. Plasma, in turn, consists of 92 percent water and 8 percent organic molecules and proteins [57]. Plasma can be modeled as a Newtonian fluid, but due to the existence of blood cells, the blood exhibits viscoelastic properties. Mature RBCs lack a cell nucleus. Their resting shape is a biconcave disk of about $8\text{ }\mu\text{m}$ in diameter and $2\text{ }\mu\text{m}$ in thickness. The cell wall consists of a thin (7 nm) lipid bilayer and the interior of RBCs is a concentrated solution of oxygen-binding protein hemoglobin which behaves as an incompressible Newtonian fluid whose viscosity is higher than that of plasma [69]. In Table 1 the composition of the cellular phase of human blood is given.

Table 1: COMPOSITION OF CELLULAR PHASE OF HUMAN BLOOD. (*reproduced from [57]*).

Cell type	Shape	Diameter (μm)	Number per ml	Mean volume fraction of blood	Number fraction
Erythrocyte (red cell)	Biconcave disc	8 to 8.5	5×10^9	0.45	99.1
Thrombocyte (platelet)	Thin biconcave disc	2 to 4	4×10^8	0.003	0.7
Leucocyte (white cell)	Round or oval	10 to 22	8×10^6	< 0.001	< 0.2

Based on the data given in Table 1 and the previous paragraph, for the *fluid-mechanical* analysis of blood flow, the important characteristics of the microcirculation can be summarized as following:

- The particulate nature of the blood is important in the capillaries where the diameter of the capillary is in the same order of the diameter of RBCs, which means in capillaries

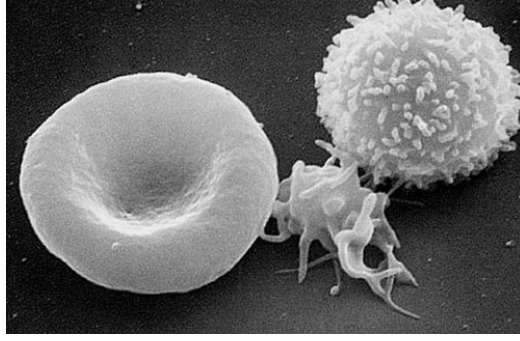


Figure 1: THE SHAPE OF DIFFERENT BLOOD CELLS AT REST. *Scanning electron micrograph of blood cells. From left to right: human erythrocyte, activated thrombocyte, and lymphocyte [source Wikimedia Commons].*

up to about $80\text{ }\mu\text{m}$ in diameter.

- In small capillaries, the Reynolds number, Re , is of the order of 10^{-3} so that inertial effects may be neglected.
- The suspending plasma is a Newtonian fluid.
- Due to their fluid-filled bag-like structure, the RBCs are readily *deformed* in contrast to the relatively rigid vessel walls, platelets, and white blood cells.
- Experimental studies indicate that the approximate ratios between bending, in-plane shear and expansion moduli of the RBC membrane are $1:50:10^6$ [78]. Therefore, for all practical purposes, the membrane of the RBCs is considered non-extensible.
- Since the RBCs lack a nucleus, the dynamics of the RBC are closely captured by a those of a deformable lipid bilayer model [16, 69, 73].

Note that the deformability of RBCs is of major physiological significance because it enables the cell to pass through capillaries with diameters as small as $3\text{ }\mu\text{m}$. This contributes to reducing the bulk viscosity of blood in the larger vessels, and enhances oxygen transport to the tissues by allowing mixing of the hemoglobin.

These conditions lead to a simplified (but still realistic) model in which the RBCs are represented as deformable elastic capsules with bending rigidity that are subject to constant local area and constant enclosing volume constraints. These elastic capsules are also known as vesicles. Moreover, in this model both the plasma and hemoglobin are Newtonian fluids

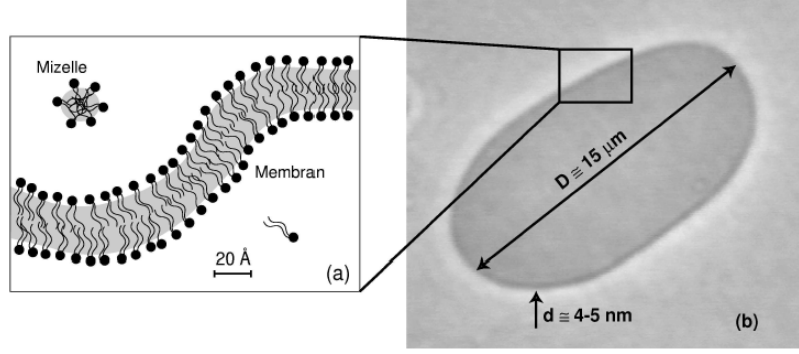


Figure 2: GIANT VESICLES. *Sketch of a lipid bilayer membrane and phase contrast microscopy picture of a giant vesicle, reproduced from [121].*

whose motion is governed by the Stokes equations.

Particulate flows have been studied mostly using physical experiments. But, until recently, analytical and computational analysis of the flow of a large ensemble of deformable particles were respectively hampered by the prohibitive complexity and computational cost of the model. Toward this end, we present a mathematical framework and develop a computational infrastructure to address the following objectives:

- (i) The RBC mechanical model should capture tension and bending forces and enable simulations with highly deformable particles.
- (ii) The discretization scheme should be accurate and stable and if possible, circumvent stiffness.
- (iii) The overall method should have good parallel scalability to allow direct numerical simulations of several microliters of blood.

The first two requirements are easy to be formulated mathematically; but their numerical solution poses significant challenges in terms of accuracy and stability. In the next section, we present a model for direct simulation of blood flow.

1.2 Numerical Model

We model the blood as a particulate suspension of elastic membranes that resist bending and tension and are filled with a Newtonian fluid. This type of particles are generally known as “*vesicles*”, a schematic of which is shown in Figure 2. In the rest of this document we will use the terms vesicle and RBC interchangeably referring to this elastic capsule representation.

The suspending fluid is also assumed to be Newtonian. Results from experiments (with vesicles) and computations that respect the above mentioned assumptions reproduce several phenomena observed in red blood cell flows [16, 69].

The vesicle evolution dynamics is characterized by a competition between membrane elastic energy, surface inextensibility, vanishing in-plane shear resistance, and non-local hydrodynamic interactions. Simulation of vesicles is a challenging nonlinear free boundary value problem, not amenable to analytical solutions in all but a few simple cases; numerical simulations and experiments are the only options for the quantitative characterization of vesicle flows.

The surface of the vesicles in our scheme is very smooth and homeomorphic to a sphere and therefore our best option for a numerical scheme is to use a boundary integral formulation. Boundary integral representations are well suited for the simulation of the flow of deformable particles since they only require the discretization of the boundary of the particles (and the boundary of the domain in case of bounded flows). This formulation tracks the trajectories of Lagrangian points on the membrane of each vesicle. In this formulation, the evolution equation of the interface of the k^{th} vesicle, γ_k , can be formally written as

$$\frac{\partial \mathbf{x}}{\partial t} = \mathbf{v}_{\text{background}}(\mathbf{x}) + \mathbf{v}_{\text{self}}(\mathbf{x}) + \mathbf{v}_{\text{interaction}}(\mathbf{x}), \quad \text{for all } \mathbf{x} \in \gamma_k \ (k = 1, \dots, N), \quad (1)$$

where $\mathbf{v}_{\text{background}}$ represents the imposed flow, analytically given in the case of unbounded flows and imposed by the boundaries in case of confined flows. \mathbf{v}_{self} represents the effect of the current shape of the vesicle on its evolution (for instance, the relaxation of a single vesicle in quiescent fluid has only this term) and the last term $\mathbf{v}_{\text{interaction}}$ represents the interaction of vesicles.

We have developed novel algorithms for the two dimensional flow of in bounded and unbounded domains and the three dimensional flow of vesicles in unbounded domains. The evaluation of the self-interaction term, \mathbf{v}_{self} , requires accurate tracking of the vesicles' interface and involves high-order derivatives of the surface of the vesicles. The high order derivative is a source of ill-conditioning in the resulting linear system of equations. The

self-interaction term also encompasses the singular Stokes integral, which is computationally intensive. For N vesicles, the naive evaluation of the last term, $\mathbf{v}_{\text{interaction}}$, has the computational complexity of $\mathcal{O}(N^2)$ that is intractable when N is large. This evaluation can be accelerated by the fast multipole method (FMM) to a computational complexity of $\mathcal{O}(N)$.

1.3 Contributions

The author had several contributions to the field of numerical simulation of deformable particles in the form of published articles and computer software (Figure 3), which are summarized in this section.

Dynamic simulation of locally inextensible vesicles suspended in an arbitrary two-dimensional domain, a boundary integral method [110]. We considered numerical algorithms for the simulation of hydrodynamics of two-dimensional vesicles suspended in a viscous Stokesian fluid. The main contributions of this paper are (i) the extension of the techniques developed in [74, 99, 135] to vesicle flows in confined geometry and vesicles with *viscosity contrast*, (ii) the numerical investigation of the stability and accuracy of the time-stepping schemes, and (iii) a preliminary validation of our methodology by comparing our numerical results to results in the literature. The presence of viscosity contrast between the vesicle’s interior fluid and the solvent requires a different integral equation formulation that results in non-trivial modifications to the previous numerical scheme [135].

Our scheme is based on Lagrangian tracking of marker particles on the vesicle, semi-implicit time discretization and spectral representation of the interface, together with high-order accurate quadrature rules. High-order accuracy in space is ensured by using a Fourier basis discretization for all functions and computing derivatives in Fourier domain, as well as high-order, Gauss-trapezoidal quadrature rules [2] for discretization of single-layer potentials. In time, we use a semi-implicit marching scheme [5]. This discretization yields a linear system of equations for each time step, which is solved using GMRES. Nearly-singular integrals, which arise when vesicles come close the fixed boundary, are resolved using the

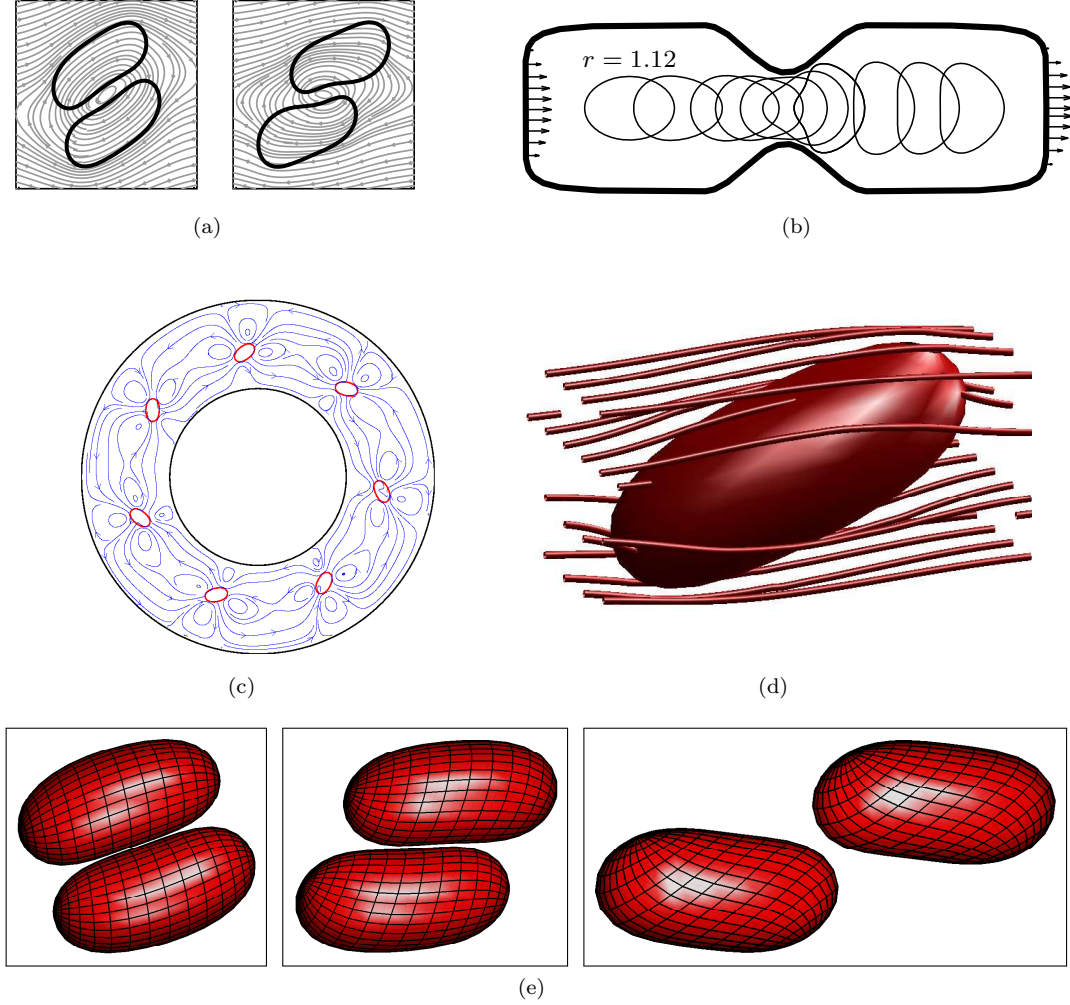


Figure 3: SUMMARY OF THE COMPUTATIONAL INFRASTRUCTURE FOR DIRECT NUMERICAL SIMULATION OF BLOOD FLOW. (a) The close interaction of two vesicles with viscosity contrast in a two dimensional shear flow. The hydrodynamic interaction of vesicles is well resolved by the boundary integral formulation and no extra collision detection algorithm is necessary (Chapter 2). (b) The flow of a vesicle through a constricted tube. The velocity on the fixed boundaries (thick line) is imposed. The vesicles undergo high deformations and interacts closely with the wall (Chapter 2). (c) The spontaneous organization of vesicles in the Couette apparatus. The simulation was started from a random distribution of vesicles. From a random distribution, it takes about 150 non-dimensional time units and about 150,000 semi-implicit steps to reach this steady-state configuration. This kind of simulation is facilitated mainly due to high accuracy in space, semi-implicit time stepping scheme, the low cost of implicit solves per vesicle, and the low cost of the evaluation of the wall effects (Chapter 3). The streamlines show the perturbation field due to the presence of vesicles (compared to the unperturbed velocity field in the Couette machine in the absence of vesicles). (d) The three dimensional flow of a single vesicle in shear flow. The streamlines show the flow field around the vesicle. For the parameters of this problem, after a transition phase, the vesicle undergoes a steady-state tank-treading motion (Chapter 4). (e) The interaction of two vesicles with viscosity contrast in the shear flow. The viscosity contrast of both vesicles is 10 and the reduced volume is 0.85. During this simulation we capture close interaction, tanktreading, and tumbling of vesicles (Chapter 5).

method proposed in [145].

One significant challenge in simulating vesicle dynamics is the numerical stiffness of the governing integro-differential equations. To gain insight on the spectral properties of operators we use a “frozen coefficient” analysis on the unit circle. This analysis allows us to construct a preconditioner for the GMRES solver. Putting everything together, we were able to achieve high accuracy in space and time, while taking large time steps without incurring high computational costs.

Vesicle migration and spatial organization driven by flow line curvature [53].

Using the computational infrastructure described in the previous paragraph, we conducted numerical experiments to investigate the cross-streamline migration of deformable entities. Cross-streamline migration appears in many applications, such as industrial polymer processing [141], DNA sorting [123], drop dynamics [32], and biofluids. We carried out simulations in a Taylor-Couette cell by taking vesicles as a model system for the suspended entities.

Through numerical experiments, we have discovered that vesicles suspended in a flow with curved flow lines migrate towards regions of high flow-line curvature, which are regions of high shear rate. However, typically, deformable particles have the tendency to migrate towards regions of low shear rates [40, 59, 65, 119, 133]. We proposed an explanation for this different behavior and provided quantitative evidence for the case of vesicle flows. Our finding quantitatively demonstrates a direct coupling between a microscopic quantity (migration) and a macroscopic one (normal stress difference).

Furthermore, simulations with multiple vesicles revealed a self-organization, which corresponds to segregation, in a rim closer to the inner cylinder, resulting from a subtle interaction among vesicles. Such segregation effects could have significant impact on rheology of vesicle flows.

A fast algorithm for simulating vesicle flows in three dimensions [136]. We developed a new fast algorithm for the flow of three dimensional deformable particles (with no viscosity contrast) in unbounded geometries. We extended the ideas presented in [135]

to the general case of vesicles in three dimensions. A summary of our contributions is as follows: (i) we developed a high-order spatial discretization for inextensible vesicles based on spherical harmonics, combining and extending a number of previously proposed techniques, including extending the quadrature scheme of [58] to the Stokes kernel; (ii) we proposed a particular linearization for a semi-implicit time-stepping scheme along with a preconditioning scheme; (iii) we analyzed the stiffness of the evolution equations and developed a preconditioner for the linear system of equations in the semi-implicit time scheme; and (iv) we proposed a reparametrization scheme for stabilization of time-stepping, and analyzed its accuracy and stability.

Our method is based on Lagrangian tracking of spectral collocation points placed on the membrane of the vesicle combined with a surface reparametrization scheme. We achieve spectral accuracy in space by using spherical harmonics as basis. For weakly-singular integrals, we use the scheme proposed in [58], which enables high-accuracy simulations with a small (compared to low-order schemes) number of points per vesicle. For the position update in time, we use two variants of a semi-implicit marching scheme first derived for advection-diffusion equations [5] and then applied on integral-equation based fluid-structure interaction problems in [129].

The time-marching scheme requires the solution of a linear system of equations at each time step, which we perform using a Krylov iterative method (GMRES [114]). The problem of poor conditioning is addressed by a preconditioner based on the analytically obtained spectrum of the operators for the special case of a unit sphere. Vesicle-vesicle interactions are carried out using the kernel independent fast multipole method [144].

In all, we were able to achieve high accuracy while using a small number of unknowns per vesicle for the spatial discretization and taking large time steps with a relatively low computational cost per time step.

Three-dimensional boundary integral method for the flow of vesicles with viscosity contrast [111]. We extend the formulation and numerical schemes described in the previous paragraph to the case of three-dimensional vesicle flows with viscosity contrast,

where the viscosity of the fluid enclosed inside each vesicle can differ from the viscosity of the suspending fluid. This generalization poses two new types of difficulty: (i) change in the boundary integral formulation of the solution, in which a double-layer Stokes integral is introduced; and (ii) change of the fluid dynamics inherent to vesicle flows with viscosity contrast. We propose algorithms to deal with these challenges. The main contributions of this article are: (i) treatment of vesicles with viscosity contrast; (ii) proposal and analysis of an implicit time stepping method, which does not have stability constraints for flows of single and multiple vesicles with different viscosity contrast; and (iii) the characterization and treatment of aliasing in differentiation.

As in [136], we present vesicles in spherical harmonics basis and singular integrals are calculated by the scheme proposed in [58]. We show that our proposed semi-implicit method does not have time-step stability constraints for flows with single and multiple vesicles with different viscosity contrast and the computational cost-per-simulation-unit-time is comparable to or less than that of an explicit scheme.

Petascale direct numerical simulation of blood flow on 200K cores and heterogeneous architectures [109]. We developed a fast, petaflop-scalable algorithm for Stokesian particulate flows. We focused on the parallelization and performance analysis for the computation of \mathbf{v}_{self} and $\mathbf{v}_{\text{interaction}}$ introduced in Equation (1). \mathbf{v}_{self} requires nine different computational kernels. $\mathbf{v}_{\text{interaction}}$ uses the FMM, which in turn has five major computational phases (tree construction, three tree-traversals, and the direct interactions). Our key contributions are as follows: (i) We developed a hybrid-parallel implementation of the computational kernels that are used for the computation of \mathbf{v}_{self} and $\mathbf{v}_{\text{interaction}}$. The kernels are multithreaded and work-partitioned between CPU and GPU, which execute concurrently, thereby delivering excellent per-node performance. (ii) The most intensive kernels in our computation have been designed for locality, accuracy, and computational efficiency, capitalizing in particular on highly optimized BLAS3 (GEMM) operations. (iii) We further improved the performance of the SC’09 FMM algorithm [77]. These improvements include explicit SSE vectorization and multithreading via OpenMP, as described in prior work [33].

In this paper, we added simultaneous asynchronous GPU acceleration. (iv) We present single-node analysis for computations of \mathbf{v}_{self} and $\mathbf{v}_{\text{interaction}}$ on AMD, Intel, and NVIDIA platforms; and (v) we present weak and strong scaling results on the Jaguar PF system at Oak Ridge National Laboratory (ORNL).

We achieve 780 TFlop/s of sustained performance on the 196,608 cores of the AMD Istanbul-based Jaguar PF system (4 GFlop/s per core), with $160\times$ speedup on strong scaling when moving from 48 to 24,576 cores ($512\times$); and 75% efficiency for the weak scaling. On other platforms, we demonstrate up to 18 GFlop/s per core of sustained performance on the Intel Nehalem-EP; and up to 350 GFlop/s per NVIDIA Fermi C2050 card (both in single precision).

In our largest simulation, we solved a problem involving 8,000 RBCs per MPI process, on 32,768 MPI processes for a total of 196,608 cores. We discretized using 84 points per RBC. This set of parameters results in a total of 262,144,000 RBCs (50 drops of blood) and 90 billion unknowns per time step.

Software artifacts. The algorithms presented in [110] are implemented in Matlab. The software is utilized by different research groups for the analysis of the flow of vesicles in two dimensions [53, 66], and resulted in significant scientific discoveries [66]. The algorithms for the flow of vesicles in three dimension [111, 136] are also available as a Matlab library. As we mentioned above, the algorithms in [136], were implemented and scaled to thousands of cores in a C++ software library MoBo [109]. We designed MoBo to support parallelism at all levels, including inter-node distributed memory parallelism, intra-node shared memory parallelism, data parallelism (vectorization), and fine-grained multithreading for GPUs. We have implemented and optimized the majority of the computation kernels on both Intel/AMD x86 and NVidia’s Tesla/Fermi platforms for single and double floating point precision.

1.4 Outline of The Thesis

Each of the aforementioned publications is individually presented in a chapter of this thesis. In Chapter 2 we present the two dimensional flow of vesicles with viscosity contrast in

bounded domains. In Chapter 3 we investigate the cross-streamline migration of deformable particles using the tools developed in Chapter 2. In Chapter 4 we outline algorithms for the simulation of vesicles in three dimensions. Chapter 5 extends the algorithms given in Chapter 2 and Chapter 4 to the flow of vesicles with viscosity contrast in unbounded three dimensional domains. In Chapter 6 we report the implementation and scaling of the algorithms given in Chapter 4 to thousands of cores.

Nomenclature. For the sake of clarity, we refrain from defining universal notational conventions for the whole document. Instead, depending on the context and objectives of each chapter, we define the symbols and notations appropriately at the beginning of each chapter.

CHAPTER II

DYNAMIC SIMULATION OF LOCALLY INEXTENSIBLE VESICLES SUSPENDED IN AN ARBITRARY TWO-DIMENSIONAL DOMAIN, A BOUNDARY INTEGRAL METHOD

In this chapter we consider numerical algorithms for the simulation of hydrodynamics of two-dimensional vesicles suspended in a viscous Stokesian fluid. Our method is an extension of the work of Veerapaneni et al. [135], in which a semi-implicit time-marching scheme based on a boundary integral formulation of the Stokes problem for vesicles in an unbounded medium was proposed. In this chapter, we consider two important generalizations:

- (i) confined flows within arbitrary-shaped stationary/moving geometries; and
- (ii) flows in which the interior (to the vesicle) and exterior fluids have “*viscosity contrast*”.

These two problems require solving additional integral equations and cause nontrivial modifications to the previous numerical scheme. Our method does not have severe time-step stability constraints and its computational cost-per-time-step is comparable to that of an explicit scheme. The discretization is pseudo-spectral in space, and multistep BDF in time. We conduct numerical experiments to investigate the stability, accuracy and the computational cost of the algorithm. Overall, our method achieves several orders of magnitude speed-up compared to standard explicit schemes.

As a preliminary validation of our scheme, we study the dependence of the inclination angle of a single vesicle in shear flow on the viscosity contrast and the reduced area of the vesicle, the lateral migration of vesicles in shear flow, the dispersion of two vesicles, and the effective viscosity of a dilute suspension of vesicles.

2.1 Introduction

Vesicles are closed lipid membranes suspended in a viscous medium. The mechanical deformation of vesicles and their interaction with viscous fluids are thought to play an important

role in many biological phenomena [73, 121] and are used experimentally to understand properties of biomembranes [115]. In addition, vesicle mechanics have been used as models for the motion of red and white blood cells [92, 101], whose quantitative description will help in better understanding of blood rheology.

In this chapter, we focus on numerical schemes for continuum models of vesicle dynamics in two dimensions. This is a challenging problem because the motion and shape of the vesicles must be determined dynamically from a balance of interfacial forces with fluid stresses. The shape dynamics of fluid vesicles is governed by the coupling of the flow within the membrane of the vesicle with the hydrodynamics of the surrounding bulk fluid. Following our previous publication on vesicle flows [135], we present a semi-implicit numerical scheme for the simulation of the motion of arbitrarily shaped vesicles that can have a viscosity contrast with the background fluid. We also extend our formulation to handle interior flows and interaction of vesicles with other moving particles with prescribed motion.

Our method is based on an integral equation formulation. In particulate flow problems involving vesicles, the elastic and incompressibility properties of their membranes must be resolved and the numerical schemes must be modified in order to accommodate these properties and to solve the resulting set of equations. Details of the boundary integral formulation for elastic interfaces and incompressible vesicles can be found in the works of Pozrikidis [103, 105].

The overwhelming majority of works on particulate flows uses explicit schemes that pose severe restrictions on the time step. In contrast, semi-implicit methods result in two to three orders of magnitude larger step size that is almost independent of the spacial grid size [135]. In contrast to stencil-based methods (e.g., finite element methods), integral equation formulations avoid discretization of the overall domain and instead discretize only the vesicle boundary and the boundary of the enclosing domain. This is the main reason that integral equations have been used extensively for vesicle, and more generally, particulate and interfacial flow simulations [105].

The rest of this chapter is organized as follows: In the remainder of this section we outline our contributions as well as the limitations of our work and then review the related

work to the study of particulate flows with deformable particles in two dimensions. We finish this section with a summary of the nomenclature used in this chapter. In Section 2.2, we state the problem and its formulation. In Section 2.3, we outline the numerical scheme we use to solve the derived equations. In Section 2.4, we report results from numerical experiments we performed to demonstrate the stability of the proposed time-marching scheme in different flow regimes and geometry configurations. In particular, we investigate the effect of fixed boundaries on the time-stepping numerical stability of our scheme. We conclude in Section 2.4.4 with a discussion of the rheology of dilute suspensions of vesicle flows with viscosity contrast.

2.1.1 Contributions

The boundary integral formulation coupled to the shape dynamics results in an integro-differential equation that is constrained by the local inextensibility. Extending our previous work [135], we use semi-implicit time-stepping, fast summation schemes, and spectral discretization in space. The combination of these approaches for flows with interface singularities is not unique. However, we are unaware of any previous analysis and application of implicit time-stepping schemes combined with fast solvers to vesicles that have a viscosity contrast with the surrounding fluid and are interacting with confined boundaries. These improvements enable the simulation of a large number of interacting vesicles, as described in Section 2.3 and Section 2.4, and depicted in Figure 4. The main contributions of the work presented in this chapter are:

- The extension of the techniques developed in [74, 99, 135] to vesicle flows in confined geometry and vesicles with viscosity contrast.
- The numerical investigation of the stability and accuracy of the time-stepping schemes.
- A preliminary validation of our methodology by comparing our numerical results to results in the literature.

In particular, for validation, we investigate (i) the dependence of vesicles' inclination angle in shear flow on viscosity contrast and reduced area; (ii) the lateral migration of

vesicles in shear flow due to collision; and (iii) the rheology of a dilute suspension of vesicles.

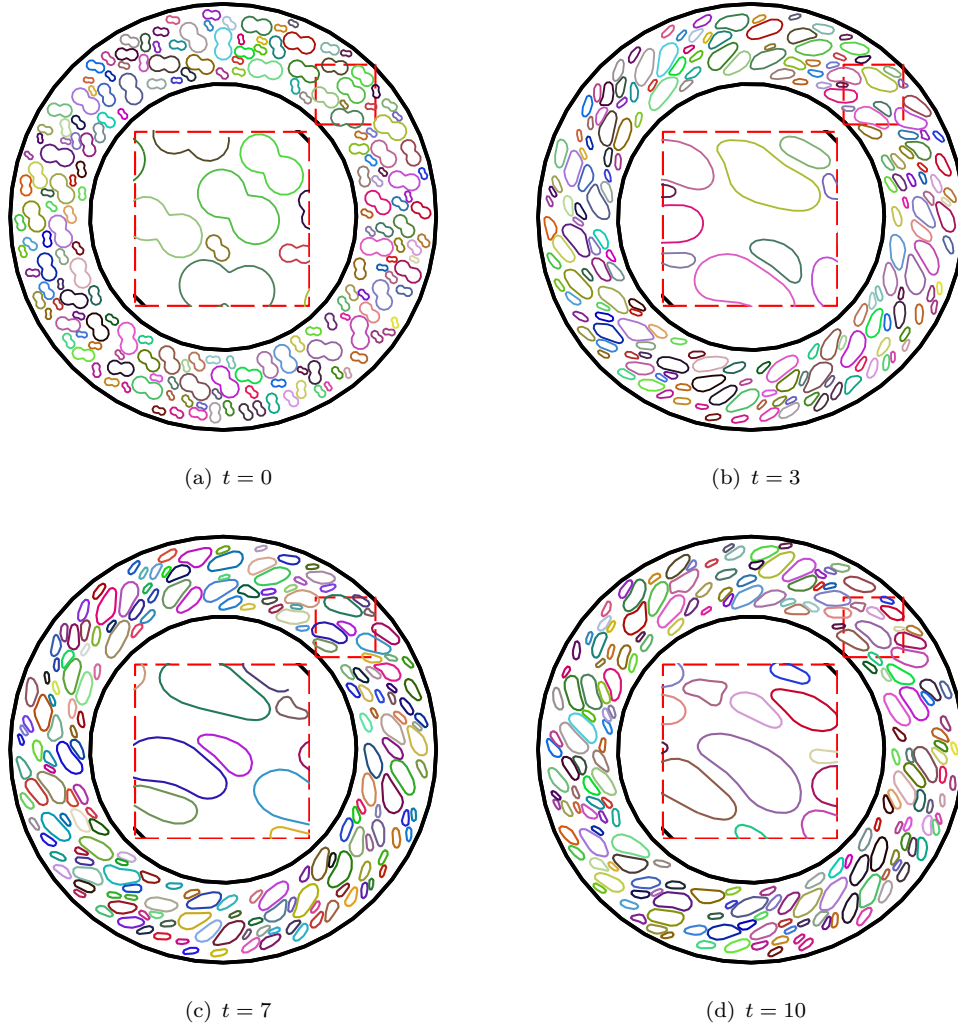


Figure 4: NUMERICAL SIMULATION OF VESICLES IN THE COUETTE APPARATUS. *In this figure, we demonstrate the capabilities of our method; in particular, its ability to resolve complex interactions between multiple vesicles. We simulated the motion of 192 vesicles in a 2D Couette apparatus. The outer boundary is fixed while the inner boundary rotates with a constant angular velocity. In this simulation, we used 64 discretization points per vesicle, 640 points on boundaries, and we took a total of 1000 time steps. The computations were performed using MATLAB. The wall-clock time per time-step is 80s on a Xeon processor, vesicle-vesicle and vesicle-boundary interactions are evaluated on a NVIDIA Tesla Graphics Processing Unit with the total wall-clock time being three seconds per time-step. Four snapshots of the simulation are shown. We zoom in on the region marked by the broken-line square to show the details of the interaction between vesicles. Here, $t \in [0, 10]$ is the nondimensional time. In this time span, the inner cylinder makes approximately one full rotation. The initial configuration was obtained by random distribution of vesicles. Due to bending, the vesicle shapes are quickly smoothed to lower energy configurations. We resolve high curvature regions (subfigures (c) and (d)), conserve vesicle areas and lengths ($\max |A - A_0|/A_0 = 2.18e-2$ and $\max |L - L_0|/L_0 = 4.69e-2$), and compute the hydrodynamic interactions with sufficient accuracy to avoid collisions without employing a collision detection algorithm. Details on the accuracy and complexity of our method are presented in later sections.*

2.1.2 Limitations

The most significant limitation of our method is that the number of Fourier modes used to represent the vesicle membrane and the time step are not chosen adaptively. The former is a minor limitation (in 2D) but the latter is quite significant. Our spectral discretization (which we combine with a special high-order scheme for singular integrals) in space results in discretization errors that are dominated by the time-stepping scheme. In our experiments, 64 to 128 Fourier modes in space are typically sufficient to fully resolve the shapes of the vesicles in the flow regimes we have examined. For more concentrated suspensions, adaptive schemes combined with a posteriori estimates may be necessary.

We solve the discretized system of equations using the Generalized Minimum Residual Method (GMRES) [114] with an appropriate set of preconditioners, which are based on the spectral properties of the operators. Nonetheless, for very small viscosity contrasts $\nu \ll 1$ (See Table 2 for its definition), the spectral properties of the operators change and a generic preconditioner, as we use here, fails to fully compensate for the poor conditioning of the operators.

2.1.3 Related work

Vesicles are used, theoretically and experimentally, to investigate the properties of biological membranes [115], blood cells [92, 101], and drug-carrying capsules [125].

Integral equation methods have been used extensively for the simulation of Stokesian particulate flows, mostly for droplets (with or without viscosity contrast) and bubbles. These methods were introduced by Youngren and Acrivos [146] for a flow past a rigid particle of arbitrary shape. An excellent review of numerical methods for Stokesian flows is done by Pozrikidis [105]. The present work is based on a formulation derived by Rallison and Acrivos [112] for two fluids separated by an interface with surface forces and the work of Power and Miranda [99, 100], who introduced an integral equation formulation of the Stokes problem in two and three dimensional multiply-connected domains with Dirichlet boundary conditions.

In spite of the large body of literature devoted to investigating the rheological properties

of red blood cell and vesicles suspensions, to the best of our knowledge, the work on numerical methods for vesicle flows with viscosity contrast and confined boundaries is rather limited. Freund [50] considers vesicles with no viscosity contrast in a bounded domain. In his work, the boundaries are treated as panels fixed to their location with virtual springs. Zhou and Pozrikidis [149] consider the flow of a periodic suspension of 2D viscous drops between two parallel plane walls, for which an explicit expression of Green’s function is available.

Let us also mention works related to the test-case flows we have used to validate our numerical method. Kraus et al. [73] studied the dynamics of a vesicle and its steady-state inclination angle in the absence of viscosity contrast. Beaucourt et al. [16] tackled the same problem in the presence of viscosity contrast. Kantsler and Steinberg [64] reported results from experimental study of the inclination angle of vesicles and their transition from tank-treading to tumbling. Misbah [88] looked at the theoretical aspects of a vesicle’s inclination angle problem. Loewenberg [81] and Loewenberg and Hinch [83] studied the dispersion of drops in shear flow and Eckstein et al. [47] investigated particle-particle interaction and their lateral migration in an experimental setting. Rheology of (dilute) suspension of vesicles was investigated by Danker et al. [38], Loewenberg [81], Ramanujan and Pozrikidis [113], and Vitkova et al. [137]. Danker et al. [38] investigated the rheological properties of a dilute suspension of vesicles in shear flow analytically.

2.1.4 Synopsis of Our Method

We propose a computational scheme for the evolution of vesicles in a confined domain. We also extend our method [135] to the case where there is a viscosity contrast between the suspending fluid and the internal fluid of the vesicle. Our scheme is based on Lagrangian tracking of marker particles on the vesicle, semi-implicit time discretization and spectral representation of the interface, together with high-order accurate quadrature rules. These choices result in a spectrally accurate method in space and second-order accurate method in time.

High-order accuracy in space is ensured by using a Fourier basis discretization for

all functions and computing derivatives in Fourier domain, as well as high-order, Gauss-trapezoidal quadrature rules [2] for discretization of single-layer potentials. In time, we use a semi-implicit marching scheme [5]. This discretization yields a linear system of equations for each time step, which is solved using GMRES. One significant challenge in simulating vesicle dynamics is the numerical stiffness of the governing integro-differential equations. To gain insight on the spectral properties of operators we use a “frozen coefficient” analysis on the unit circle. This analysis allows us to construct a preconditioner for the GMRES solver. Putting everything together, we were able to achieve high accuracy in space and time, while taking large time steps without incurring high computational costs. Our formulation for confined boundaries is based on the method in [99]. Finally, we resolve nearly-singular integrals, which arise when vesicles come close the fixed boundary, using the method proposed in [145].

2.1.5 Nomenclature

Throughout this chapter, lower case letters will refer to scalars, and lowercase bold letters will refer to vectors. We use \otimes for the tensor product of two vectors and $|\cdot|$ to denote the measure of its argument (e.g., the Euclidean norm of a vector or the area of a domain). We denote the jump across interfaces by $\llbracket u \rrbracket := u^+ - u^-$, where $u^\pm(\mathbf{x}) := \lim_{h \downarrow 0} u(\mathbf{x} \pm h\mathbf{n})$, \mathbf{n} denoting the outward normal to the boundary. We denote the convolution of an integral kernel K with density $\boldsymbol{\eta}$ by $\mathcal{K}[\mathbf{y}, \boldsymbol{\eta}](\mathbf{x}) := \int_{\Gamma} K(\mathbf{x}, \mathbf{y}) \boldsymbol{\eta}(\mathbf{y}) ds(\mathbf{y})$ where the product inside the integral should be interpreted as a tensor operation when K is a tensor and as a dot product when K is a vector. In Table 2, we list symbols and operators frequently used in this chapter.

2.2 Formulation

The general formulation for the flow of vesicles in an unbounded domain was presented in the introduction. Here we extend that formulation to the case of bounded flows. Consider a suspension of vesicles in an ambient Newtonian fluid. Let Ω be the domain of interest, an open bounded subset of \mathbb{R}^2 that can be multiply connected and whose boundary consists of $M + 1$ infinitely differentiable curves $\Gamma_0, \dots, \Gamma_M$, among which, Γ_0 denotes the enclosing

Table 2: INDEX OF FREQUENTLY USED SYMBOLS AND OPERATORS IN CHAPTER 2. *The equation number or the section of definition is referred to inside the parenthesis.*

Symbol/Operator	Definition
Γ	Boundary of domain Ω
Δ	Reduced Area (Section 2.2.4)
Ω	Domain of interest
γ_p	Boundary of p^{th} vesicle
ζ	Traction jump across interface (Section 2.2.3)
η	Double-layer density over Γ (Section 2.2.2)
μ	Fluid viscosity
ν_p	Viscosity contrast μ_p/μ_0
σ	Tension, or stress tensor
χ	Shear rate
ω_p	Domain enclosed by γ_p
$\mathcal{B}(\mathbf{y})\mathbf{x}$	Bending operator (Equation (16))
$\mathcal{B}[\eta, \Xi, \Lambda](\mathbf{x})$	Hydrodynamic operator due to fixed boundaries with density η Rotlet strength Ξ , and Stokeslet strength Λ , evaluated at point \mathbf{x} (Section 2.2.2)
$\mathcal{E}[\mathbf{y}, \mathbf{u}, \zeta](\mathbf{x})$	Hydrodynamic operator due to current configuration of vesicle \mathbf{y} , velocity field \mathbf{u} , and traction jump ζ evaluated at point \mathbf{x} (Equation (4))
$\mathcal{P}(\mathbf{x})$	Surface divergence operator (Equation (2c))
$\mathcal{T}(\mathbf{y}, \mathbf{x})\sigma$	Tension operator (Equation (16))
T	Simulation time horizon
p	Pressure
\mathbf{u}	Velocity

boundary of the domain. The ambient fluid has viscosity μ_0 . The vesicles are evolving under the influence of an imposed velocity field. Let γ_p ($p = 1, \dots, N$) denote the boundary of the p^{th} vesicle, ω_p denote the domain enclosed by γ_p , and μ_p denote the viscosity of the fluid inside that vesicle (see Figure 5). Let $\Gamma := \bigcup_k \Gamma_k$ and $\gamma := \bigcup_p \gamma_p$. We use \mathbf{x} or \mathbf{y} to denote both a typical point in the domain Ω and a point on the interface γ . When $\mathbf{x} \in \gamma$, we define $\dot{\mathbf{x}}$ and \mathbf{x}_s as the interfacial velocity and the derivative of the \mathbf{x} with respect to arc length.

When the Reynolds number is based on the vesicle size, the effect of inertial forces is insignificant and the fluid flow is governed by the Stokes equation,

$$\mu \Delta \mathbf{u} - \nabla p = \mathbf{0}, \quad \text{div } \mathbf{u} = 0, \quad \text{in } \Omega \setminus \gamma, \quad (2a)$$

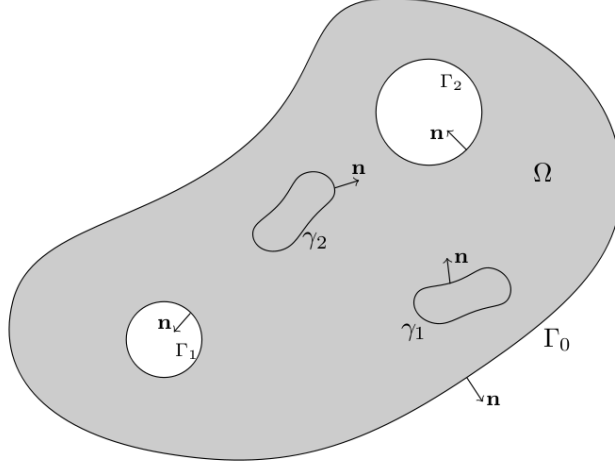


Figure 5: SCHEMATIC OF THE TWO DIMENSIONAL DOMAIN. A typical domain of interest Ω (shaded area) with boundary Γ_i ($i = 0, \dots, M$). The boundary of each vesicle is denoted by γ_j ($j = 1, \dots, N$). The enclosed domain by γ_j is denoted by ω_j .

where $\mathbf{u}(\mathbf{x}, t)$ is the velocity field, $p(\mathbf{x}, t)$ is the pressure field, and μ is the viscosity of fluid. We supplement Equation (2a) with the velocity no-slip condition on γ_p , and with velocity Dirichlet boundary condition on Γ as

$$\mathbf{u}(\mathbf{x}, t) = \dot{\mathbf{x}}(t) \quad \text{when } \mathbf{x} \in \gamma, \text{ and } \quad \mathbf{u}(\mathbf{x}, t) = \mathbf{U}(\mathbf{x}, t) \quad \text{when } \mathbf{x} \in \Gamma, \quad (2b)$$

where $\dot{\mathbf{x}}$, as we mentioned earlier, is the velocity of the points on the vesicle membrane. The assumption that the velocity field is continuous across the vesicle interface, $[[\mathbf{u}]] = 0$, is implicit in our expression of no-slip condition. Henceforth, we drop the explicit time dependence of the variables. Finally, to enforce the local inextensibility of the vesicles we require that the surface divergence of the velocity field vanishes. That is

$$\mathbf{x}_s \cdot \mathbf{u}_s = 0 \quad \text{for } \mathbf{x} \in \gamma. \quad (2c)$$

Here \mathbf{x}_s is the derivative of position of the points on γ with respect to arc length, i.e. the tangent vector. For notational convenience, when $\mathbf{x} \in \gamma$ and \mathbf{u} is defined on γ we define $\mathcal{P}(\mathbf{x})\mathbf{u} := \mathbf{x}_s \cdot \mathbf{u}_s$; then the inextensibility condition can be written as $\mathcal{P}(\mathbf{x})\mathbf{u} = 0$ for $\mathbf{x} \in \gamma$.

2.2.1 Boundary Integral Formulation

Due to the continuity of the velocity field across the interfaces, we can follow the standard approach of potential theory [99, 102, 105], and reformulate Equations (2a) and (2b) using

layer potentials. It follows that the velocity at a point \mathbf{x} is formally given by

$$\alpha \mathbf{u}(\mathbf{x}) = \mathcal{E}[\mathbf{y}, \mathbf{u}, \boldsymbol{\zeta}](\mathbf{x}) + \mathcal{B}[\boldsymbol{\eta}, \Xi, \Lambda](\mathbf{x}), \quad \mathbf{x} \in \Omega, \quad (3a)$$

where

$$\alpha = \begin{cases} 1 & \mathbf{x} \in \Omega \setminus \cup_p \omega_p, \\ \nu_p & \mathbf{x} \in \omega_p, \\ (1 + \nu_p)/2 & \mathbf{x} \in \gamma_p, \end{cases}$$

subject to the inextensibility constraint,

$$\mathcal{P}(\mathbf{x})\mathbf{u} = 0, \quad \mathbf{x} \in \gamma, \quad (3b)$$

where $\nu_p := \mu_p/\mu_0$ is the contrast between the viscosity of the fluid enclosed by p^{th} vesicle and that of the background medium, and $\boldsymbol{\zeta}$ denotes the traction jump across the interface.

As it is explained in [102, 105] we have

$$\mathcal{E}[\mathbf{y}, \mathbf{u}, \boldsymbol{\zeta}](\mathbf{x}) := \sum_{q=1}^N \mathcal{S}_q[\mathbf{y}, \boldsymbol{\zeta}](\mathbf{x}) + \mathcal{D}_q[\mathbf{y}, \mathbf{u}](\mathbf{x}). \quad (4)$$

Operators \mathcal{S}_q and \mathcal{D}_q are the single- and double-layer hydrodynamic potentials for Stokes flow evaluated on the q^{th} interface, defined as

$$\mathcal{S}_q[\mathbf{y}, \boldsymbol{\zeta}](\mathbf{x}) = \frac{1}{4\pi\mu_0} \int_{\gamma_q} \left(-\log \rho I + \frac{\mathbf{r} \otimes \mathbf{r}}{\rho^2} \right) \boldsymbol{\zeta} ds(\mathbf{y}), \quad (5)$$

$$\mathcal{D}_q[\mathbf{y}, \mathbf{u}](\mathbf{x}) = \frac{1 - \nu_q}{\pi} \int_{\gamma_q} \frac{\mathbf{r} \cdot \mathbf{n}}{\rho^2} \frac{\mathbf{r} \otimes \mathbf{r}}{\rho^2} \mathbf{u} ds(\mathbf{y}), \quad (6)$$

in which $\mathbf{r} := \mathbf{x} - \mathbf{y}$ and $\rho := |\mathbf{r}|$. \mathcal{B} is the completed double-layer operator for confined Stokes flow with density $\boldsymbol{\eta}(\mathbf{y})$, defined as

$$\mathcal{B}[\boldsymbol{\eta}, \Xi, \Lambda](\mathbf{x}) := \mathcal{D}[\mathbf{y}, \boldsymbol{\eta}](\mathbf{x}) + \mathcal{N}_0[\mathbf{y}, \boldsymbol{\eta}](\mathbf{x}) + \sum_{k=1}^M R(\mathbf{x}, \mathbf{c}_k) \xi_k + \sum_{k=1}^M S(\mathbf{x}, \mathbf{c}_k) \boldsymbol{\lambda}_k, \quad (7)$$

where $\Lambda = \{\boldsymbol{\lambda}_1, \dots, \boldsymbol{\lambda}_M\}$ and $\Xi = \{\xi_1, \dots, \xi_M\}$. In Section 2.2.2, we define \mathcal{D} , \mathcal{N}_0 , R , S , Ξ , and Λ and outline the derivation of \mathcal{B} that is based on [99]. Taking the limit of Equation (3a) to the boundary Γ we obtain an equation for the double-layer density $\boldsymbol{\eta}$

$$\alpha \mathbf{U}(\mathbf{x}) = -\frac{1}{2} \boldsymbol{\eta}(\mathbf{x}) + \mathcal{E}[\mathbf{y}, \mathbf{u}, \boldsymbol{\zeta}](\mathbf{x}) + \mathcal{B}[\boldsymbol{\eta}, \Xi, \Lambda](\mathbf{x}) \quad \mathbf{x} \in \Gamma. \quad (8)$$

Operator \mathcal{E} represents the velocity induced by the evolution of the vesicle and operator \mathcal{B} corresponds to the velocity due to the imposed flow via the boundary Γ . Note that if we replace \mathcal{B} with a far field \mathbf{u}^∞ in Equation (3a), we obtain the formulation for the case of an unbounded flow. Also, note that we need the traction jump $\boldsymbol{\zeta}$ to evaluate \mathcal{E} . We give details on how to calculate the traction jump in Section 2.2.3.

2.2.2 Completed Indirect Solution of Stokes Flow in a Confined Geometry

Consider the domain Ω given in Figure 5 in the absence of vesicles. The solution of the Stokes equation, Equation (2a), in this domain can be written as a double-layer integral with density $\boldsymbol{\eta}$ [99, 102]. The velocity and pressure at a point $\mathbf{x} \in \Omega$ can be written as

$$\mathbf{u}(\mathbf{x}) = \mathcal{D}[\mathbf{y}, \boldsymbol{\eta}](\mathbf{x}) = \frac{1}{\pi} \int_{\Gamma} \frac{\mathbf{r} \cdot \mathbf{n}}{\rho^2} \frac{\mathbf{r} \otimes \mathbf{r}}{\rho^2} \boldsymbol{\eta} ds, \quad (9)$$

$$p(\mathbf{x}) = \mathcal{K}[\mathbf{y}, \boldsymbol{\eta}](\mathbf{x}) = -\frac{\mu}{\pi} \int_{\Gamma} \frac{\mathbf{n} \cdot \boldsymbol{\eta}}{\rho^2} - 2 \frac{\mathbf{r} \cdot \mathbf{n}}{\rho^2} \frac{\mathbf{r} \cdot \boldsymbol{\eta}}{\rho^2} ds. \quad (10)$$

As it is nicely explained in [68], Equation (9) cannot represent general flow fields. In particular, it cannot represent the flow due to rigid body motions. To compensate for this deficiency, following [99], we add *Stokeslet* and *Rotlet* terms for each Γ_k ($1 \leq k \leq M$) to Equation (9). The Stokeslet is the Green's function for the Stokes equation, which is the same as the single-layer kernel S given in Equation (6). The Rotlet is the antisymmetric component of the Stokes doublet defined by

$$R(\mathbf{x}, \mathbf{y})(\boldsymbol{\xi}) := \frac{\boldsymbol{\xi} \cdot \mathbf{r}^\perp}{\mu \rho^2}, \quad (11)$$

for any strength $\boldsymbol{\xi}$. For a vector $\mathbf{r} = (r_1, r_2)$, we define $\mathbf{r}^\perp =: (r_2, -r_1)$. Both the Stokeslet and Rotlet are centered at an interior point \mathbf{c}_k of the domain enclosed by the boundary component Γ_k . For convenience, we choose $\boldsymbol{\lambda}$, the strength vector for each Stokeslet, and $\boldsymbol{\xi}$ to depend linearly on the unknown density $\boldsymbol{\eta}$ in the following manner

$$\boldsymbol{\lambda}_{k,1} := \frac{1}{2\pi} \int_{\Gamma_k} \boldsymbol{\phi}_1(\mathbf{y}) \cdot \boldsymbol{\eta}(\mathbf{y}) ds(\mathbf{y}), \quad \boldsymbol{\lambda}_{k,2} := \frac{1}{2\pi} \int_{\Gamma_k} \boldsymbol{\phi}_2(\mathbf{y}) \cdot \boldsymbol{\eta}(\mathbf{y}) ds(\mathbf{y}), \quad (12)$$

and

$$\boldsymbol{\xi}_k := \frac{1}{2\pi} \int_{\Gamma_k} \boldsymbol{\phi}_3(\mathbf{y}) \cdot \boldsymbol{\eta}(\mathbf{y}) ds(\mathbf{y}) \quad (13)$$

where $\phi_1 = \delta_{1i}, \phi_2 = \delta_{2i}$ ($i = 1, 2$) are the rigid body translations in the plane and ϕ_3 is the rigid body rotation, that is $\phi_3(\mathbf{x}) = \mathbf{x}^\perp$.

Since the flow is confined by the contour Γ_0 , as noted in [68] and [102], conservation of mass ($\text{div } \mathbf{u} = 0$) implies that the velocity field defined by Equation (9) satisfies the Stokes equation only when $\boldsymbol{\eta}$ satisfies $\int_{\Gamma_0} \boldsymbol{\eta} \cdot \mathbf{n} ds = 0$. To enforce this orthogonality condition, we follow [68] and add an additional operator $\mathcal{N}_0[\mathbf{y}, \boldsymbol{\eta}](\mathbf{x}) = \int_{\Gamma_0} N_0(\mathbf{x}, \mathbf{y}) \boldsymbol{\eta}(\mathbf{y}) ds(\mathbf{y})$ with kernel $N_0(\mathbf{x}, \mathbf{y}) = \mathbf{n}(\mathbf{x}) \otimes \mathbf{n}(\mathbf{y})$ to the right-hand side of Equation (9). Hence, the velocity for $\mathbf{x} \in \Omega$ can be represented by

$$\mathbf{u}(\mathbf{x}) = \mathcal{D}[\mathbf{y}, \boldsymbol{\eta}](\mathbf{x}) + \mathcal{N}_0[\mathbf{y}, \boldsymbol{\eta}](\mathbf{x}) + \sum_{k=1}^M R(\mathbf{x}, \mathbf{c}_k) \xi_k + \sum_{k=1}^M S(\mathbf{x}, \mathbf{c}_k) \boldsymbol{\lambda}_k =: \mathcal{B}[\boldsymbol{\eta}, \Xi, \Lambda](\mathbf{x}), \quad (14)$$

where $\Xi = \{\xi_1, \dots, \xi_M\}$, $\Lambda = \{\boldsymbol{\lambda}_1, \dots, \boldsymbol{\lambda}_M\}$. In this way, we obtain a system of Fredholm integral equations of the second kind. Taking the limit of Equation (14) to points \mathbf{x} on the boundary of the domain, we obtain an equation for $\boldsymbol{\eta}$ supplemented with Equation (12) and Equation (13) for calculation of $\boldsymbol{\lambda}_k$ and ξ_k

$$\mathbf{U}(\mathbf{x}) = -\frac{1}{2} \boldsymbol{\eta}(\mathbf{x}) + \mathcal{B}[\boldsymbol{\eta}, \Xi, \Lambda](\mathbf{x}) \quad \mathbf{x} \in \Gamma, \quad (15a)$$

$$\boldsymbol{\lambda}_k = \frac{1}{2\pi} \int_{\Gamma_k} \boldsymbol{\eta}(\mathbf{y}) ds(\mathbf{y}), \quad \xi_k = \frac{1}{2\pi} \int_{\Gamma_k} \phi_3(\mathbf{y}) \cdot \boldsymbol{\eta}(\mathbf{y}) ds(\mathbf{y}). \quad (15b)$$

Along with $\boldsymbol{\eta}$, we need the jumps at the boundaries to be able to evaluate the pressure and stress. We have

$$[[\mathbf{u}]] = \boldsymbol{\eta}; \quad [[p]] = -2\mu(\boldsymbol{\eta}_s \cdot \mathbf{t}); \quad [[\sigma \mathbf{n}]] = 0,$$

in which \mathbf{t} is the tangent vector to the boundary, $\boldsymbol{\eta}_s$ denotes the derivative of density with respect to the arc length, and σ is the stress tensor on the boundary.

2.2.3 Traction Jump Across the Interface

In the absence of gravity,¹ the traction jump across the interface depends exclusively on the material properties and the configuration of the interface. The traction jump across the interface balances the forces caused by bending and tension [105]. We can write $\boldsymbol{\zeta} = \mathbf{f}_b + \mathbf{f}_\sigma$,

¹We can include gravity by adding $(\Delta\rho)(\mathbf{g} \cdot \mathbf{x})\mathbf{n}$ to the traction jump, where $\Delta\rho = \rho_{\text{out}} - \rho_{\text{in}}$ is the density difference.

where \mathbf{f}_b denotes the bending force and \mathbf{f}_σ is the force due to tension. Let κ_b denote the bending modulus of the vesicle, σ the tension, and κ the local curvature. Then, the elastic energy of the membrane is given by $\varepsilon_p(\kappa, \sigma) = \int_{\gamma_p} \left(\frac{1}{2} \kappa_b \kappa^2 + \sigma \right) ds$. The forces are obtained by taking the gradient of the membrane energy. See [105, 135] for details on the derivation of the expressions for these forces. Accordingly, forces can be written as $\mathbf{f}_b = -\kappa_b \mathbf{x}_{ssss}$ and $\mathbf{f}_\sigma = (\sigma \mathbf{x}_s)_s$, where the subscript s denotes differentiation with respect to arc length. For convenience, we introduce the following notation

$$\begin{aligned}
\mathcal{B}(\mathbf{y})\mathbf{x} &= -\kappa_b \mathcal{S}[\mathbf{y}, \mathbf{x}_{ssss}](\mathbf{x}), \\
\mathcal{T}(\mathbf{y}, \mathbf{x})\sigma &= \mathcal{S}[\mathbf{y}, (\sigma \mathbf{x}_s)_s](\mathbf{x}), \\
\mathcal{L}(\mathbf{x}) &= \mathcal{P}(\mathbf{x})\mathcal{T}(\mathbf{x}, \mathbf{x}), \\
\mathcal{M}(\mathbf{x}) &= \mathcal{T}(\mathbf{x}, \mathbf{x})\mathcal{L}^{-1}(\mathbf{x})\mathcal{P}(\mathbf{x}), \\
\mathcal{D}(\mathbf{y})\mathbf{u} &= \mathcal{D}[\mathbf{y}, \mathbf{u}](\mathbf{x}),
\end{aligned} \tag{16}$$

and $\mathcal{T}(\mathbf{x}) := \mathcal{T}(\mathbf{x}, \mathbf{x})$.

2.2.4 Scaling

Let \bar{r} , and \bar{t} denote the characteristic length and time and let L be the perimeter of the vesicle. We define $\bar{r} := L/2\pi$ as the radius of the circle with perimeter L , and the time scale $\bar{t} := \mu_0 \bar{r}^3 / \kappa_b$. We define the characteristic tension as $\bar{\sigma} := \kappa_b / \bar{r}$. For the velocity scale, we consider two cases: an unbounded shear flow and a confined flow. *Shear Flow*: We assume that $\mathbf{u}^\infty = \chi[x_2, 0]^T$ where χ is the shear rate. The characteristic shear rate is then $\bar{\chi} := 1/\bar{t}$ and the characteristic velocity is $\bar{r}\bar{\chi}$. *Confined Flow*: In this case, the characteristic velocity may be defined by \bar{U} as the velocity defined on the boundary.

A parameter that determines the behavior of vesicles significantly is the *reduced area*, which is the contrast of vesicle's area to that of a circle with the same perimeter, $\Delta := A/(\pi \bar{r}^2)$.

2.2.5 Summary of the Vesicles' Equations of Motion in Two-dimensional Confined Geometries

Incorporating the notation introduced in Equation (16), for a Lagrangian point \mathbf{x} we can write the governing equations as

Vesicle evolution:

$$\alpha \mathbf{u}(\mathbf{x}) = \sum_{q=1}^N \mathcal{B}_q(\mathbf{y})\mathbf{x} + \mathcal{T}_q(\mathbf{y})\sigma + \mathcal{D}_q(\mathbf{y})\mathbf{u} + \mathcal{B}[\boldsymbol{\eta}, \Xi, \Lambda](\mathbf{x}), \quad \mathbf{x} \in \gamma, \quad (17a)$$

Inextensibility constraint:

$$\mathcal{P}(\mathbf{x})\mathbf{u} = 0, \quad \mathbf{x} \in \gamma, \quad (17b)$$

Fixed boundaries:

$$\alpha \mathbf{U}(\mathbf{x}) = -\frac{1}{2}\boldsymbol{\eta}(\mathbf{x}) + \mathcal{E}[\mathbf{y}, \mathbf{u}, \boldsymbol{\zeta}](\mathbf{x}) + \mathcal{B}[\boldsymbol{\eta}, \Xi, \Lambda](\mathbf{x}), \quad \mathbf{x} \in \Gamma. \quad (17c)$$

These equations are solvable for the interfacial velocity and tension, as well as double-layer density $\boldsymbol{\eta}$ on the fixed boundaries.

2.3 Numerical Algorithms

We use a multistep time-marching scheme. In Section 2.3.1, we give the details of our time-stepping schemes and in Section 2.3.2, we outline our approach to spatial discretization. We adopt a Lagrangian formulation, which simplifies the implementation of the high-order multistep schemes.

2.3.1 Time Discretization

We use backward difference formula to advance in time. We can write a generic form of a backward difference formula as $\frac{d\mathbf{x}}{dt} \approx \frac{\beta \mathbf{x}^{n+1} - \mathbf{x}^o}{\Delta t}$, in which \mathbf{x}^o is a linear combination previous time steps and β depends on the order. The values of \mathbf{x}^o and β are given in Table 3 for different orders of accuracy. Then, a semi-implicit formulation of equation set (17) can be

Table 3: BACKWARD DIFFERENCE COEFFICIENTS. *The Backward difference coefficients for p^{th} order accurate backward difference method.*

p	β	\mathbf{x}^o	\mathbf{x}^e
1	1	\mathbf{x}^n	\mathbf{x}^n
2	3/2	$2\mathbf{x}^n - \frac{1}{2}\mathbf{x}^{n-1}$	$2\mathbf{x}^n - \mathbf{x}^{n-1}$
3	11/6	$3\mathbf{x}^n - \frac{3}{2}\mathbf{x}^{n-1} + \frac{1}{3}\mathbf{x}^{n-2}$	$3\mathbf{x}^n - 3\mathbf{x}^{n-1} + \mathbf{x}^{n-2}$
4	25/12	$4\mathbf{x}^n - 3\mathbf{x}^{n-1} + \frac{4}{3}\mathbf{x}^{n-2} - \frac{1}{4}\mathbf{x}^{n-3}$	$4\mathbf{x}^n - 6\mathbf{x}^{n-1} + 4\mathbf{x}^{n-2} - \mathbf{x}^{n-3}$

written as

$$\begin{aligned} \frac{\alpha}{\Delta t} (\beta \mathbf{x}^{n+1} - \mathbf{x}^o) &= \mathcal{B}_p(\mathbf{y}^e) \mathbf{x}^{n+1} + \mathcal{T}_p(\mathbf{y}^e) \sigma^{n+1} + \mathcal{D}_p(\mathbf{y}^e) \mathbf{u}^{n+1} + \mathcal{B}[\boldsymbol{\eta}^n, \Xi^n, \Lambda^n](\mathbf{x}^e) \\ &+ \sum_{\substack{q=1 \\ q \neq p}}^N \{ \mathcal{B}_q(\mathbf{y}^e) \mathbf{x}^n + \mathcal{T}_q(\mathbf{y}^e) \sigma^n + \mathcal{D}_q(\mathbf{y}^e) \mathbf{u}^n \}, \quad \mathbf{x} \in \gamma_p \end{aligned} \quad (18a)$$

$$\beta \mathcal{P}(\mathbf{y}^e) \mathbf{x}^{n+1} = \mathcal{P}(\mathbf{y}^e) \mathbf{x}^o =: g, \quad (18b)$$

$$\begin{aligned} \alpha \mathbf{U}(\mathbf{x}) &= -\frac{1}{2} \boldsymbol{\eta}^{n+1}(\mathbf{x}) + \mathcal{E}[\mathbf{y}^{n+1}, \mathbf{u}^{n+1}, \boldsymbol{\zeta}^{n+1}](\mathbf{x}) \\ &+ \mathcal{B}[\boldsymbol{\eta}^{n+1}, \Xi^{n+1}, \Lambda^{n+1}](\mathbf{x}), \quad \mathbf{x} \in \Gamma, \end{aligned} \quad (18c)$$

where, \mathbf{y}^e is the interfacial position obtained by extrapolation from previous locations (see Table 3). We will use g to denote the right-hand side of Equation (18b). Γ is fixed or has a prescribed motion; \mathbf{U} may depend on time. We would like to make the following remarks regarding our scheme:

- The vesicle-boundary interactions are treated explicitly.
- The vesicle-vesicle interactions are also explicit.
- For each vesicle, its new position and tension are computed semi-implicitly by solving a *linear* system of equations.

To advance in time, first we need to solve the coupled system of equations (18a) and (18b) to calculate the new position of the vesicle \mathbf{x}^{n+1} and tension σ^{n+1} . Then, we solve Equation (18c) to calculate $\boldsymbol{\eta}^{n+1}$. We explore two different schemes to solve (18a) and (18b).

Semi-implicit scheme. Let

$$E_{\text{far}}^n := \sum_{\substack{q=1 \\ q \neq p}}^N \{ \mathcal{B}_q(\mathbf{y}^e) \mathbf{x}^n + \mathcal{T}_q(\mathbf{y}^e) \sigma^n + \mathcal{D}_q(\mathbf{y}^e) \mathbf{u}^n \} + \mathcal{B}[\boldsymbol{\eta}^n, \Xi^n, \Lambda^n](\mathbf{x}^e), \quad \mathbf{x} \in \gamma_p \quad (19)$$

Then, we rewrite Equation (18a) as

$$\frac{\alpha}{\Delta t} (\beta \mathbf{x}^{n+1} - \mathbf{x}^o) = \mathcal{B}_p(\mathbf{y}^e) \mathbf{x}^{n+1} + \mathcal{T}_p(\mathbf{y}^e) \sigma^{n+1} + \mathcal{D}_p(\mathbf{y}^e) \mathbf{u}^{n+1} + E_{\text{far}}^n. \quad (20)$$

Upon rearranging, we obtain

$$\left[\alpha \beta I - \beta \mathcal{D}_p - (\Delta t) \mathcal{B}_p \right] \mathbf{x}^{n+1} - (\Delta t) \mathcal{T}_p \sigma^{n+1} = \mathbf{q}, \quad (21)$$

where $\mathbf{q} := (\alpha I - \mathcal{D}_p) \mathbf{x}^o + E_{\text{far}}^n$. (For brevity, we have dropped the notational dependence of the operators on \mathbf{y}^e .) Notice that

$$\alpha \beta \mathbf{x}^{n+1} = (\beta \mathcal{D}_p + (\Delta t) \mathcal{B}_p) \mathbf{x}^{n+1} + (\Delta t) \mathcal{T}_p \sigma^{n+1} + \mathbf{q}.$$

Substituting into Equation (18b), we get an equation for tension as a function of the vesicle's configuration,

$$(\Delta t) \mathcal{L}_p \sigma^{n+1} = \alpha g - \mathcal{P} \mathbf{q} - \mathcal{P} (\beta \mathcal{D}_p + (\Delta t) \mathcal{B}_p) \mathbf{x}^{n+1}. \quad (22)$$

Substituting this in Equation (21), the equation for the new position is

$$\{ \alpha \beta I + (\mathcal{M}_p - I) (\beta \mathcal{D}_p + (\Delta t) \mathcal{B}_p) \} \mathbf{x}^{n+1} = \mathbf{q} + \mathcal{T}_p \mathcal{L}_p^{-1} (\alpha g - \mathcal{P} \mathbf{q}). \quad (23)$$

Explicit scheme. Let E_{far}^n be defined as before. Another approach is to first calculate \mathbf{x}^{n+1} explicitly and then compute the corresponding σ^{n+1} . Therefore, Equation (21) can be written as

$$\alpha \beta \mathbf{x}^{n+1} = (\Delta t) \mathcal{T}_p \sigma^n + \mathbf{q} + (\beta \mathcal{D}_p + \Delta t \mathcal{B}_p) \mathbf{x}^e. \quad (24)$$

Accordingly, the equation for tension becomes

$$(\Delta t) \mathcal{L}_p \sigma^{n+1} = \alpha g - \mathcal{P} \mathbf{q} - \mathcal{P} (\beta \mathcal{D}_p + \Delta t \mathcal{B}_p) \mathbf{x}^{n+1}. \quad (25)$$

In Algorithms 2.3.1 and 2.3.2, we give the pseudocode for our numerical schemes.

Algorithm 2.3.1 SEMI-IMPLICIT TIME MARCHING.

Require: $E_{\text{far}}^n, g, \mathbf{q}$

```

// Solving Equation (23) for the vesicle
for  $p = 1$  to  $N$  do
     $\sigma_{\text{exp}} \leftarrow \mathcal{L}_p \sigma_{\text{exp}}^{n+1} = \alpha g - \mathcal{P} \mathbf{q}$  // Calculating the tension part of the RHS
     $\mathbf{r} \leftarrow \mathbf{q} + \mathcal{T}_p \sigma_{\text{exp}}^{n+1}$ 
     $\mathbf{x}^{n+1} \leftarrow$  use GMRES to solve Equation (23) with right-hand side  $\mathbf{r}$  and  $\mathbf{x}^e$  as initial guess.
     $\sigma_{\text{imp}} \leftarrow \mathcal{L}_p \sigma_{\text{imp}}^{n+1} = -\mathcal{P} (\beta \mathcal{D}_p + (\Delta t) \mathcal{B}_p) \mathbf{x}^{n+1}$  // Solving for  $\sigma$  in Equation (22) with known position  $\mathbf{x}^{n+1}$ 
     $\sigma^{n+1} \leftarrow (\sigma_{\text{exp}}^{n+1} + \sigma_{\text{imp}}^{n+1}) / (\Delta t)$ 
end for
 $\mathbf{u}^{n+1} \leftarrow \frac{1}{\Delta t} (\beta \mathbf{x}^{n+1} - \mathbf{x}^o)$ 
 $\boldsymbol{\zeta}^{n+1} \leftarrow \kappa_b \mathbf{x}_{ssss}^{n+1} + (\sigma^{n+1} \mathbf{x}_s^{n+1})_s$ 
// Solving Equation (18c) for double-layer density  $\boldsymbol{\eta}$ 
 $\mathbf{r} \leftarrow \alpha \mathbf{U} - \mathcal{E}[\mathbf{x}^{n+1}, \mathbf{u}^{n+1}, \boldsymbol{\zeta}^{n+1}]$ 
 $\boldsymbol{\eta}^{n+1} \leftarrow (-I/2 + \mathcal{B})^{-1} \mathbf{r}$  // The inverse can be precomputed
return  $\mathbf{x}^{n+1}, \sigma^{n+1}, \boldsymbol{\eta}^{n+1}$ 
```

Algorithm 2.3.2 EXPLICIT TIME MARCHING.

Require: $E_{\text{far}}^n, g, \mathbf{q}$

```

for  $p = 1$  to  $N$  do
     $\mathbf{r} \leftarrow \mathbf{q} + (\Delta t) \mathcal{T}_p \sigma^n$ 
     $\mathbf{x}^{n+1} \leftarrow$  update position using Equation (24)
     $\sigma^{n+1} \leftarrow$  solve for tension using Equation (25)
end for
 $\mathbf{u}^{n+1} \leftarrow \frac{1}{\Delta t} (\beta \mathbf{x}^{n+1} - \mathbf{x}^o)$ 
 $\boldsymbol{\zeta}^{n+1} \leftarrow \kappa_b \mathbf{x}_{ssss}^{n+1} + (\sigma^{n+1} \mathbf{x}_s^{n+1})_s$ 
// Solving Equation (18c) for double-layer density  $\boldsymbol{\eta}$ 
 $\mathbf{r} \leftarrow \alpha \mathbf{U} - \mathcal{E}[\mathbf{x}^{n+1}, \mathbf{u}^{n+1}, \boldsymbol{\zeta}^{n+1}]$ 
 $\boldsymbol{\eta}^{n+1} \leftarrow (-I/2 + \mathcal{B})^{-1} \mathbf{r}$ 
return  $\mathbf{x}^{n+1}, \sigma^{n+1}, \boldsymbol{\eta}^{n+1}$ 
```

2.3.2 Spatial Discretization

Let $\theta \in (0, 2\pi]$ be a parametrization of the interface γ_p and $\theta_k = 2k\pi/n$ ($k = 1, \dots, n$) be n uniformly distributed discretization points. We have

$$\mathbf{x}(\theta) = \sum_{k=-n/2+1}^{n/2} \hat{\mathbf{x}}(k) e^{-ik\theta}.$$

This enables us to use FFT to calculate $\hat{\mathbf{x}}$ and derivatives of \mathbf{x} with spectral accuracy, since we have assumed that γ and Γ belong to C^∞ .

Quadrature Rule. The single-layer potential \mathcal{S} has a logarithmic singularity. For its integration, we use the hybrid Gauss-trapezoidal quadrature rule given in Table 8 of [2], designed to handle this kind of singularity. Let $\mathbf{y}_k = \mathbf{y}(\theta_k)$, then

$$\mathcal{S}[\mathbf{y}, \boldsymbol{\zeta}](\mathbf{x}) \approx \sum_{k=1}^{n+m} w_k S(\mathbf{x}, \mathbf{y}_k) \boldsymbol{\zeta}(\mathbf{y}_k) |\mathbf{y}_{\theta,k}|,$$

where n is the number of nodes, m is number of quadrature nodes, w_k the quadrature weights, and \mathbf{y}_θ is the Jacobian. The number m is determined by the desired order of convergence for the integral.

The double-layer potential has no singularity in two dimensions and for $\mathbf{x}, \mathbf{y} \in \gamma$, $\lim_{\mathbf{x} \rightarrow \mathbf{y}} D(\mathbf{x}, \mathbf{y}) = \kappa \mathbf{t} \otimes \mathbf{t} / 2\pi$. Thus, a composite trapezoidal rule will give spectral accuracy since the integrands are periodic and smooth. Therefore,

$$\mathcal{D}[\mathbf{y}, \mathbf{u}](\mathbf{x}) \approx \frac{2\pi}{n} \sum_{k=1}^n D(\mathbf{x}, \mathbf{y}_k) \mathbf{u}(\mathbf{y}_k) |\mathbf{y}_{\theta,k}|.$$

2.3.3 Analysis of the Spectral Properties

We use spectral analysis of the operators defined on the unit circle to characterize the stiffness of the underlying problem. If we use n Lagrangian points to represent the interface, as it is explained in detail in [135], the condition number of the single-layer potential operator scales as $\mathcal{O}(n)$ and the condition number of the bending operator scales as $\mathcal{O}(n^3)$. On the other hand, the double-layer potential operator alters only the first three frequencies of the integrand.

The viscosity contrast ν appears in the coefficient $\alpha = \frac{1+\nu q}{2}$ and the operator \mathcal{D} , given in Equation (6). Now, consider three cases where $\nu \gg 1$, $\nu \approx 1$, and $\nu \ll 1$: (i) $\nu \gg 1$: The vesicle acts more like a rigid body and in Equation (23), $\frac{1}{\alpha} \mathcal{D}$ remains finite but $\frac{1}{\alpha} \mathcal{B}$ tends to zero and thus the double-layer operator dominates. Therefore, the condition number of the whole operator is bounded. (ii) $\nu \approx 1$: The effect of double-layer potential is minimal and the bending operator dominates. (iii) $\nu \ll 1$: The double-layer potential and bending

terms are comparable to each other. In the last two cases, the condition number of the operator grows in cubic rate thereby requiring preconditioning for the iterative solvers. We explore this in greater detail in our experiments in Section 2.4.1. The existence of viscosity contrast has no effect on the tension operator and thus its spectral properties are the same as those explored in [135].

In our time-stepping scheme, we require the solution of systems with the operators \mathcal{L} , and $I + (\mathcal{M} - I)(\mathcal{D} + (\Delta t)\mathcal{B})$. Based on our spectral analysis here and in [135], the condition number of these system behave as $\mathcal{O}(n)$ and $\mathcal{O}(n^3)$, where n is the number of modes in space. In [135], we proposed a set of low-cost preconditioners for the position and tension solver. Since the tension operator is exactly the same for all cases of viscosity contrast, what we had for the case of $\nu = 1$ carries over to the general case. On the other hand, because the double-layer operator has a bounded condition number, the main source of ill-conditioning for the position solver (at least at moderate values of viscosity contrast) is the bending operator and the preconditioner proposed in [135] applies.

2.3.4 Computational Cost of the Semi-implicit Scheme for a Single Time-step

Assume that we have N vesicles, with n discretization points per vesicle interface and m discretization points on the boundary Γ (in the case of bounded flow, of course). The semi-implicit scheme involves “inversion” of three operators: the prescribed-motion boundary double-layer, the inextensibility operator, and the position-update operator. And it also requires evaluation of single- and double-layers, and differentiations on the vesicle boundary.

The semi-implicit algorithm has two main components: one is the evaluation of the effect of boundaries on a vesicle’s Lagrangian points and the other is the solution of Equation (23) for the new position and tension. There are two facts that lead to a fast algorithm: 1. The double-layer operator is well-conditioned and thus, GMRES converges to the solution in mesh-independent manner; and 2. the boundary-vesicle and vesicle-vesicle interactions can be accelerated by the fast multipole method. Thus, using GMRES, the calculation of the density over Γ requires $\mathcal{O}(m)$ work per time step.² The evaluation the double-layer at

²Alternatively, when the boundary with prescribed motion is fixed, one could use a fast scheme to precompute the action of its inverse to a vector [86].

each Lagrangian point on the vesicle interface (evaluation of \mathcal{B}) is $\mathcal{O}(m)$ and, using FMM, $\mathcal{O}(Nn + m)$ for all vesicles.³ Furthermore, we use FFT to calculate the derivatives, thus the evaluation of traction jump on each vesicle requires $\mathcal{O}(n \log n)$ per time step. When we solve for the positions using Equation (23), at each GMRES iteration of position, we solve for tension, which takes $\mathcal{O}(Nn \log n)$ time and thus, each GMRES iteration requires $\mathcal{O}(Nn \log n)$ work. However, in Section 2.4.1, we demonstrate numerically that the number of iterations is nearly independent of the problem size (Table 6). Hence, the overall cost of updating the positions and the tensions for all vesicles is $\mathcal{O}(Nn \log n + m)$.

2.4 Numerical Experiments

In this section, we present results on the convergence, stability, and algorithmic complexity of the proposed methods, which we have implemented in MATLAB. We perform the following tests:

- We consider a single vesicle in Section 2.4.1. Our goal is to demonstrate the stability and accuracy of our scheme as a function of the parameters, specifically viscosity contrast ν . We also report the dependence of tank-treading inclination angle on a vesicle's viscosity contrast and shear rate.
- We consider the dispersion of vesicles due to pairwise interaction and collision in shear flow is studied in Section 2.4.2.
- We consider the effect of fixed boundaries in Section 2.4.3 and their effect on the overall accuracy of our method.
- We consider the rheological properties of dilute suspensions of vesicles in Section 2.4.4.

2.4.1 Single Vesicle

We consider the case of a single vesicle with viscosity contrast ν suspended in an unbounded shear flow. In our experiment, we consider ν to be 0.01, 0.1, 1, 10, and 100. When the

³Notice, that when the number of vesicles is high, one needs to implement a fast summation scheme for the Rotlet and Stokeslet terms. The Kernel Independent Fast Multipole Method can be used for this purpose [144].

Table 4: STABLE STEP SIZE FOR A SECOND-ORDER SEMI-IMPLICIT SCHEME. *The initial vesicle configuration is the same as the one given in Figure 6. When $\chi = 0$, the vesicle relaxes to minimum energy equilibrium shape. Since there is no input energy from the ambient fluid flow, all step sizes are permissible and do not lead to (numerical) instability. Comparing with Table 5, we see three orders of magnitude speedup.*

$\Delta = .4$										
n	$\nu = .01$		$\nu = .1$		$\nu = 1$		$\nu = 10$		$\nu = 100$	
	$\chi = 1$	10	1	10	1	10	1	10	1	10
32	4e-1	2e-2	2e-1	1e-2	2e-1	7e-2	7e-1	9e-2	7e-1	7e-2
64	4e-1	1e-2	2e-1	1e-2	3e-1	5e-2	7e-1	2e-2	7e-1	5e-2
128	9e-2	3e-3	9e-2	6e-3	2e-1	6e-2	7e-1	9e-3	7e-1	3e-2
256	1e-2	1e-3	4e-2	2e-3	2e-1	6e-2	3e-1	3e-3	3e-1	2e-2
$\Delta = .75$										
32	7e-1	3e-2	7e-1	5e-2	7e-1	7e-2	7e-1	7e-2	7e-1	7e-2
64	7e-1	3e-2	7e-1	5e-2	7e-1	7e-2	7e-1	7e-2	7e-1	7e-2
128	7e-1	5e-3	7e-1	5e-2	7e-1	7e-2	7e-1	7e-2	7e-1	7e-2
256	2e-2	2e-3	3e-2	3e-3	7e-1	7e-2	7e-1	6e-3	4e-1	3e-2

viscosity contrast is low, based on experiment and theory [64, 88], the vesicle undergoes a tank-treading motion at an equilibrium angle ϕ_ν . When the viscosity contrast is high, the vesicle tumbles. We first investigate the stability and convergence properties of the proposed numerical schemes. In Table 4, we report the largest uniform step-size required to maintain numerical stability for different shear rates and viscosity contrasts for a BDF method of order two for the semi-implicit time marching scheme. (We used a simple bisection method to find these time steps.) The time horizon is chosen such that the vesicle reaches steady state or the desired dynamical phenomenon (i.e., tank-treading or tumbling is observed). The reduced area Δ is set to .4 and .75. Comparing these results with the results of the explicit scheme given in Table 5, we see three orders of magnitude speedup for most cases.

In Figure 6 and Figure 7, we show the configuration of the vesicle for different values of viscosity contrasts ν and reduced area Δ . In Figure 8, we show the sedimentation shape of the vesicle under the influence of gravity.

In Table 6, we report the number of GMRES iterations (corresponding to $\Delta = .75$ only). Observe that the number of iterations for the tension solver is almost mesh independent and viscosity contrast has no effect on the tension solver, as expected. The effect of the

Table 5: STABLE STEP SIZE FOR THE FIRST-ORDER EXPLICIT SCHEME. *Simulation parameters and the initial configuration of the vesicle are the same as the simulation shown in Figure 6.*

n	$\nu = .1$		$\nu = 1$		$\nu = 10$	
	$\chi = 1$	10	1	10	1	10
32	3e-4	2e-4	2e-3	1e-3	3e-3	1e-3
64	2e-5	2e-5	9e-5	1e-4	2e-4	2e-4
128	2e-6	3e-6	3e-6	5e-6	1e-5	2e-5
256	3e-7	2e-7	6e-7	3e-7	2e-6	1e-6

high condition number (caused by bending) becomes very pronounced in cases where the mesh size equals 128 and 256 but our preconditioner compensates for that well.

Due to inextensibility and incompressibility, the length and area of vesicles should be preserved. In Table 7, we report the relative error in area and length of a single vesicle in shear flow.

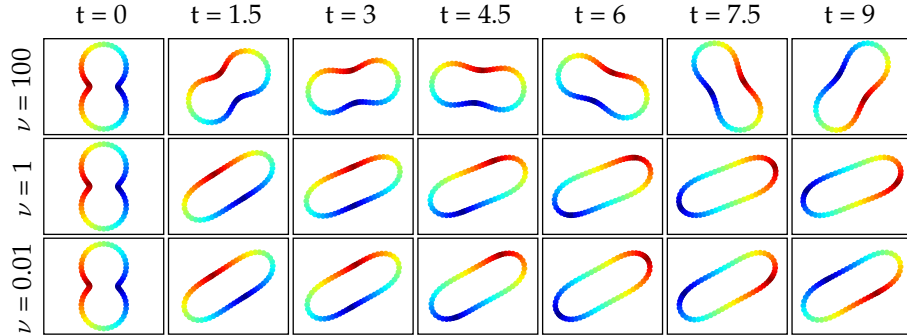


Figure 6: THE EVOLUTION OF A SINGLE VESICLE WITH REDUCED AREA $\Delta = 0.75$ IN AN UNBOUNDED SHEAR FLOW. *The viscosity contrast $\nu = .01, 1$, and 100 , $\chi = 1$, $\kappa_b = 1$, and time horizon $T = 9$. Lagrangian points on the vesicle membrane is colored for visual purposes only and has no other significance. When $\nu < \nu_c$ we see that the vesicle reaches an equilibrium and then undergoes tank-treading motion (see Figure 9 for further analysis). When ν is large the vesicle tumbles.*

The inclination angle of vesicles in tank-treading motion is of physical interest since it can be compared with experimental results. Since we consider only two-dimensional vesicle flows, this comparison is qualitative. Based on theory, this inclination angle depends only on the reduced area Δ and the viscosity contrast ν . For any fixed Δ , there exists a ν_c such that for viscosity contrasts larger than that, the vesicle starts to tumble. For $\nu < \nu_c$, there

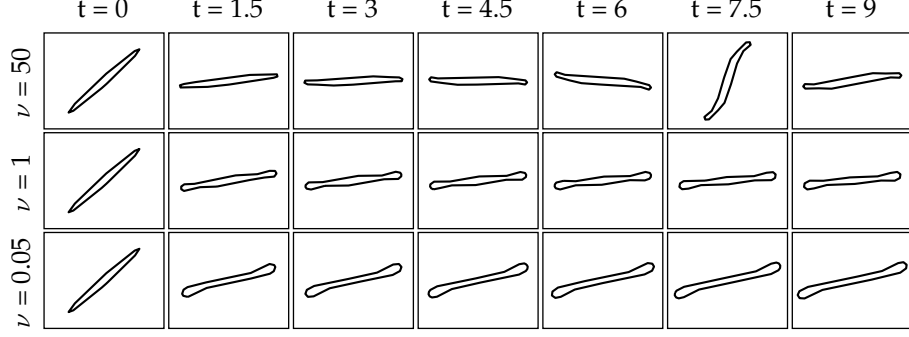


Figure 7: THE EVOLUTION OF A SINGLE VESICLE WITH REDUCED AREA $\Delta = 0.2$ IN AN UNBOUNDED SHEAR FLOW. *The rest of the parameters are the same as the simulation shown in Figure 6.*

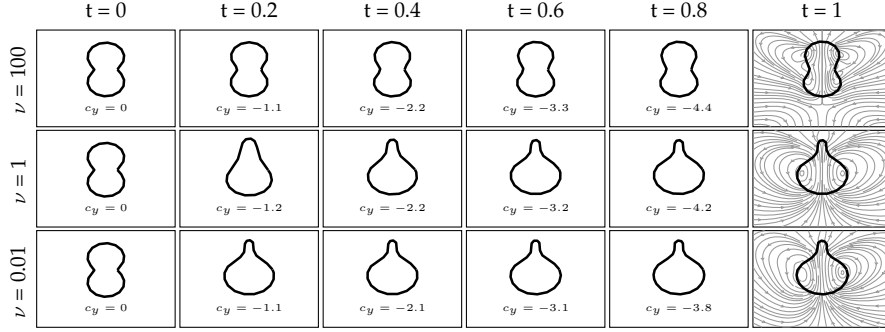


Figure 8: THE SEDIMENTATION SHAPE FOR A VESICLE WITH DIFFERENT VISCOSITY CONTRASTS. *The viscosity contrast $\nu = .01, 1$, and 100 , $\kappa_b = 1$, reduced area $\Delta = .75$, $\Delta\rho = 1$, $\mathbf{g} = -40\mathbf{e}_y$, and time horizon $T = 1$.*

exists an angle β_ν in which the vesicle tank treads. We investigate the dependence of the inclination angle β_ν (defined below) of a vesicle with reduced area $\Delta = .75$. We stopped the simulation when the rate of change in the inclination angle was less than one percent of its value, i.e. $\left| \frac{1}{\beta} \frac{d\beta}{dt} \right| \leq .01$. The alignment of the vesicle with the flow increases as the viscosity contrast is increased. A viscosity difference of $\nu_c \approx 4.1$ is the fold bifurcation point. Vesicles with larger viscosity difference undergo tumbling. As we further increase the viscosity contrast, the frequency of tumbling increases. However, we did not investigate the effect of parameters on the frequency of tank-treading and tumbling motions. Our stopping criteria explain the slight decrease in the dependence of β_ν versus the shear rate χ .

We define the inclination angle as the angle between the principal axis corresponding to the smallest principal moment of inertia with the x_1 -axis. We calculate the moment of

Table 6: NUMBER OF GMRES ITERATION AVERAGED OVER 100 TIME STEPS. *The time horizon $T = 2/\chi$ (and 10 when $\chi = 0$), $\kappa = 1$, and $\Delta = .75$. GMRES tolerance is set to 10^{-8} for the position solver and 10^{-12} for tension solver. The extrapolated position is used as starting point for the GMRES. The preconditioners are the Fourier spectrum of the involved operator on the unit circle [135].*

n	Position solver without preconditioner														
	$\nu = .01$			$\nu = .1$			$\nu = 1$			$\nu = 10$			$\nu = 100$		
	$\chi = 0$	1	10	0	1	10	0	1	10	0	1	10	0	1	10
32	13	19	13	11	18	10	8	14	7	6	10	7	5	8	8
64	17	35	21	17	34	20	19	29	15	14	20	11	7	12	8
128	41	82	49	43	81	46	47	72	34	34	48	23	16	27	13
256	103	194	122	108	193	116	119	178	84	85	120	55	40	64	31
n	Position solver with bending spectrum as preconditioner														
	$\nu = .01$			$\nu = .1$			$\nu = 1$			$\nu = 10$			$\nu = 100$		
	$\chi = 0$	1	10	0	1	10	0	1	10	0	1	10	0	1	10
32	14	23	21	11	23	16	11	18	12	8	10	8	4	8	6
64	11	25	25	11	25	22	13	20	15	9	11	8	4	8	7
128	13	28	26	13	28	23	14	22	17	10	12	9	4	10	7
256	14	32	29	14	32	25	15	25	18	10	13	9	4	10	8
n	Preconditioned tension solver														
	$\nu = .01$			$\nu = .1$			$\nu = 1$			$\nu = 10$			$\nu = 100$		
	$\chi = 0$	1	10	0	1	10	0	1	10	0	1	10	0	1	10
32	16	17	16	16	17	15	15	16	13	10	13	12	10	12	12
64	18	19	19	18	19	19	18	18	19	16	17	15	12	14	14
128	21	21	21	21	21	21	21	21	21	19	20	18	15	17	17
256	22	22	22	22	22	22	22	22	22	20	21	20	17	20	20

inertia tensor J by

$$J = \int_{\omega} (|\mathbf{r}|^2 I - \mathbf{r} \otimes \mathbf{r}) \, d\mathbf{x} = \frac{1}{4} \int_{\gamma} \mathbf{r} \cdot \mathbf{n} (|\mathbf{r}|^2 I - \mathbf{r} \otimes \mathbf{r}) \, ds,$$

where $\mathbf{r} = \mathbf{x} - \mathbf{c}$ is the distance of point \mathbf{x} from the centroid \mathbf{c} . The principal axes of inertia are the eigenvectors of J .

2.4.2 Multiple Vesicles

First, let us briefly investigate the numerical properties of our scheme in the case of multiple vesicles. In Table 8, we report the stable time-step in the presence of multiple vesicles. The vesicle-flow parameters used in Table 8 remain the same as those in Table 4. For most viscosity contrasts, the stable time-step size is similar to that of a single vesicle. However, for $\nu = 100$, we observe an order of magnitude smaller time-steps. This time-step growth was expected, if we consider the fact that we use the viscosity of the suspending fluid for

Table 7: THE RELATIVE ERROR FOR AREA AND PERIMETER OF A VESICLE IN SHEAR FLOW. *The error in area $e_A^n := |A - A_0|/A_0$ and length $e_L^n := |L - L_0|/L_0$ in the evolution of a single vesicle in an unbounded shear flow with shear rate, χ , of 10 and 250. Time marching scheme is chosen to be second-order. Due to inextensibility and incompressibility, the area and length should be preserved. For these tests, $\kappa_b = 1$, $\Delta = .75$, time horizon $T = 1/\chi$, and time step is set to $t_s = T/n$. At low shear rates we observed an erratic convergence behavior also noticed in [74, 129, 135].*

n	$\chi = 10, \nu = .04$				$\chi = 10, \nu = 1$			
	e_A^n	(e_A^n/e_A^{n-1})	e_L^n	(e_L^n/e_L^{n-1})	e_A^n	(e_A^n/e_A^{n-1})	e_L^n	(e_L^n/e_L^{n-1})
32	2.49e-2		2.32e-5		3.66e-4		1.97e-4	
64	6.26e-3 (3.98)		3.24e-4 (0.07)		8.70e-5 (4.20)		1.31e-4 (1.49)	
128	1.56e-3 (3.99)		1.31e-4 (2.46)		2.11e-5 (4.11)		5.60e-5 (2.35)	
256	3.92e-4 (3.99)		4.78e-5 (2.74)		5.23e-6 (4.04)		1.83e-5 (3.04)	
n	$\chi = 10, \nu = 25$				$\chi = 250, \nu = .04$			
	e_A^n	(e_A^n/e_A^{n-1})	e_L^n	(e_L^n/e_L^{n-1})	e_A^n	(e_A^n/e_A^{n-1})	e_L^n	(e_L^n/e_L^{n-1})
32	3.66e-4		1.97e-4		8.79e-04		2.46e-02	
64	8.70e-5 (4.20)		1.31e-4 (1.49)		1.56e-04 (5.62)		6.16e-03 (3.99)	
128	2.11e-5 (4.11)		5.60e-5 (2.35)		2.31e-05 (6.76)		1.50e-03 (4.09)	
256	5.23e-6 (4.04)		1.83e-5 (3.04)		3.69e-06 (6.26)		3.72e-04 (4.05)	
n	$\chi = 250, \nu = 1$				$\chi = 250, \nu = 25$			
	e_A^n	(e_A^n/e_A^{n-1})	e_L^n	(e_L^n/e_L^{n-1})	e_A^n	(e_A^n/e_A^{n-1})	e_L^n	(e_L^n/e_L^{n-1})
32	2.97e-06		1.75e-04		2.39e-04		1.73e-04	
64	4.50e-05 (0.07)		3.16e-05 (5.56)		1.15e-05 (20.7)		3.13e-05 (5.54)	
128	7.69e-06 (5.85)		8.56e-06 (3.69)		3.02e-06 (3.81)		6.59e-06 (4.75)	
256	1.36e-06 (5.66)		2.23e-06 (3.83)		6.73e-07 (4.50)		1.50e-06 (4.39)	

Table 8: STABLE STEP SIZE FOR A SECOND-ORDER SEMI-IMPLICIT SCHEME FOR TWO VESICLES IN SHEAR FLOW. *Both vesicles have reduced area $\Delta = .75$. Snapshots of the flow with $\nu = 1$ are shown in the plot.*

n	$\nu = .01$		$\nu = .1$		$\nu = 1$		$\nu = 10$		$\nu = 100$	
	$\chi = 1$	10	1	10	1	10	1	10	1	10
32	7e-1	3e-2	7e-1	5e-2	7e-1	7e-2	4e-1	5e-2	1e-3	5e-4
64	7e-1	3e-2	7e-1	5e-2	7e-1	7e-2	4e-1	4e-2	7e-3	1e-3
128	5e-2	5e-3	1e-1	1e-2	7e-1	7e-2	3e-1	3e-2	1e-2	1e-3
256	1e-2	1e-3	2e-2	2e-3	7e-1	7e-2	2e-1	9e-4	1e-2	1e-3

scaling and thus when $\nu = 100$, vesicles behave as rigid bodies, which thus, require smaller time steps in order to resolve their dynamics.

The collision of deformable particles has received substantial attention in the literature.

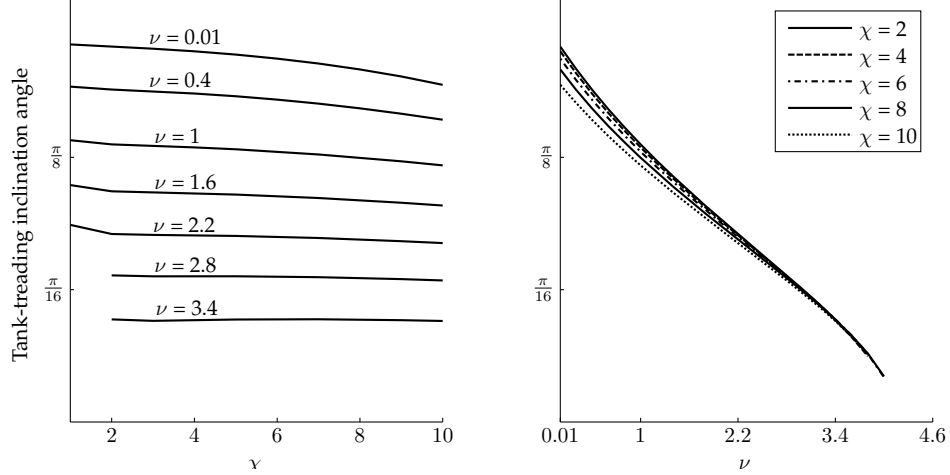


Figure 9: TANK-TREADING ANGLE AS A FUNCTION OF SHEAR RATE AND VISCOSITY CONTRAST. *For small values of viscosity contrast, a vesicle reaches steady-state and then undergoes a motion called tank-treading (see the bottom row of plots in Figure 6). Theory and experiment suggest that the inclination angle of the vesicle during tank-treading depends solely on the viscosity contrast ν and the reduced area Δ ; this dependence is confirmed in our computations. Here, the reduced area $\Delta = .75$. The critical value for viscosity contrast is $\nu_c \approx 4.1$.*

From one aspect, the particle-particle interactions in a suspension will produce irregular motions one of which is the lateral migration of particles. This migration causes dispersion in the suspension. Eckstein et al. [47] discussed the importance of this phenomena and performed experiments involving rigid particles. Loewenberg and Hinch [83] performed a numerical study of the collision of two deformable drops in shear flow.

Here, we investigate the effect of vesicles' viscosity contrast on their lateral migration in an unbounded shear flow. Initially one of the vesicles is located at the origin and the other one at $[-10, .5]^T$. The shear rate $\chi = 2$ and vesicles' bending modulus κ_b is chosen to be 0.1. The relative orientation of vesicles is a factor in the dynamics of collision and a statistical approach is needed to study its effect. Here, to minimize the effect of vesicles' orientation we choose the vesicle be very close to a circle with reduced area $\Delta = .98$. We define the offset between two vesicles to be $\delta := |c_{y,1} - c_{y,2}|$, where $c_{y,i}$ is the y coordinate position of i^{th} vesicle's centroid ($i = 1, 2$).

In Figure 10, snapshots of the interaction between vesicles are shown. Due to the inextensibility of a vesicle's membrane and the incompressibility of its fluid, vesicles maintain their circular shape at all times. In the last column of Figure 10, we plot the streamlines

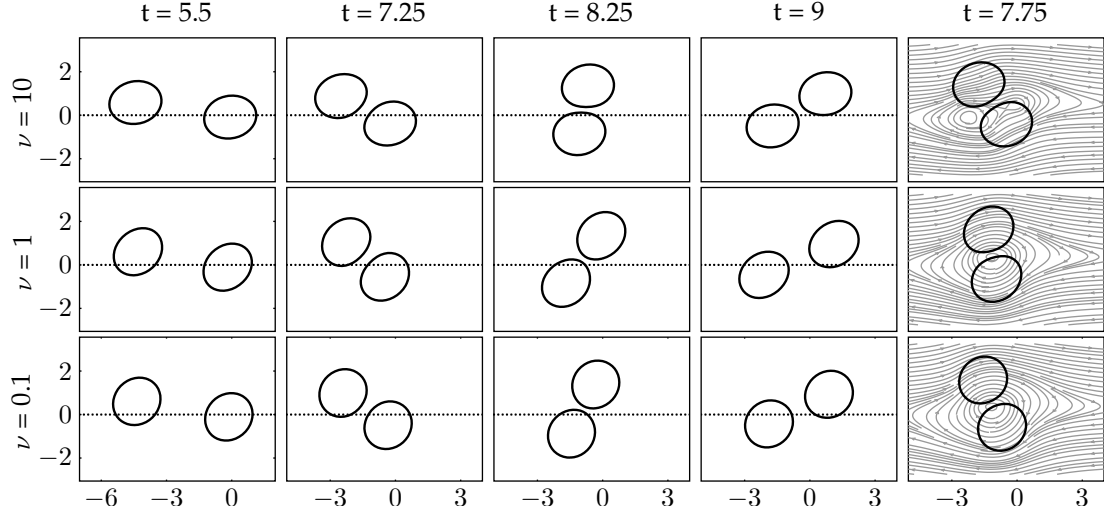


Figure 10: COLLISION OF VESICLES IN SHEAR FLOW. *Snapshots of the position of two vesicles in shear flow. Initially, one of the vesicles is located at the origin and the other one at $[-10, .5]^T$. To minimize the effect of relative orientation of vesicles on the dynamics of collision we choose the vesicles to be two identical ellipsoids with $\Delta = .98$ and $\kappa_b = .1$. In the last column we plot the streamlines at an intermediate time $t = 7.75$. Each row corresponds to a different viscosity contrast ν . The offset between the centroids of the vesicles is plotted in Figure 11.*

at an intermediate time.

In Figure 11, we plot the offset δ versus x for different values of viscosity contrast ν . The qualitative dependence of the offset on position is the same. However, we can see in the inset, the final offset δ_∞ does not monotonically depend on ν . Initially, by increasing the viscosity contrast, the final offset increases. But as ν becomes larger than one, the offset starts to decrease. The decrease in the final offset value when $\nu \gg 1$ is expected because the vesicle behaves increasingly like a rigid particle in which case the Stokes flow is reversible. However, the initial increase comes as a surprise. This trend is in contrast with the monotone decrease of the final offset for bubbles [83].

2.4.3 Vesicles in Confined Domains

The main question in the confined geometry case concerns the numerical stability of our scheme, which treats vesicle-boundary interaction explicitly. To examine the stability of our scheme, we consider the flow of a vesicle in a constricted tube. On the fixed boundaries, we impose velocity boundary conditions that correspond to an unperturbed Poiseuille flow.

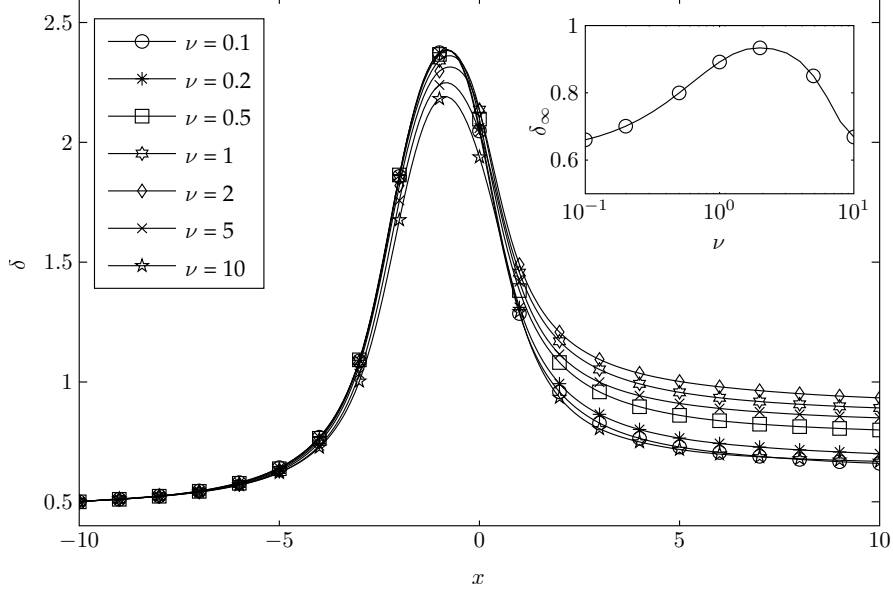


Figure 11: THE LATERAL OFFSET OF VESICLES IN SHEAR FLOW OVER TIME. The offset δ between the centroids of two vesicles in shear flow. A few exemplary snapshots of interacting vesicles are plotted in Figure 10. In the inset, the offset of the vesicle at $x = 10$, denoted by δ_∞ , is plotted versus viscosity contrast ν .

In this experiment, we consider three cases with viscosity contrasts .04, 1 and 25 respectively and $\kappa_b = .5$. There are 400 grid points on the fixed boundary and 128 points on the vesicle. The initial shape of the vesicle is an ellipsoid with reduced area $\Delta = .94$. We increase the size of the vesicle compared to the opening of the channel, and monitor the error. As a measure of the relative size, we set $r := (a + b)/2c$, where a and b are, respectively, major and minor diameter of vesicle and c is the size of the gap in the tube. To have an estimate on the errors in case of unbounded flow (and thus being able to distinguish the effect of the walls), we also simulate the evolution of a single vesicle in unbounded Poiseuille flow. We report the errors in Table 9. Also observe that in case of unbounded Poiseuille flow, the vesicle starts to migrate toward the center line. This phenomenon was studied in detail in [65].

In Table 9 we compare the stable step size corresponding to the second-order time stepping scheme of bounded flow with that of an unbounded Poiseuille flow. For the case of a vesicles with small viscosity contrast ($\nu = .2$), their faster time scale makes them responsive to the sudden outward flow in the divergent part of the tube and therefore for

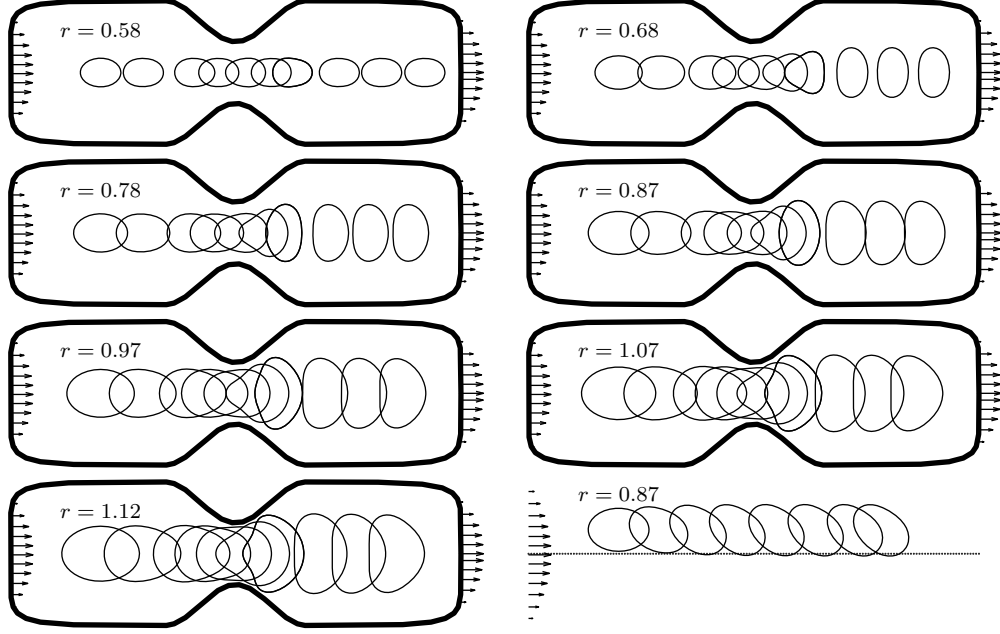


Figure 12: EVOLUTION OF A SINGLE VESICLE IN A CONSTRICTED TUBE. *Each subplot corresponds to a different vesicle size: r denotes the relative size of the vesicle with respect to diameter of the tube (defined in the text). A Poiseuille flow boundary condition on the inlet and outlet of the tube is prescribed (vector plot). In order to investigate, the error incurred due to explicit treatment of the walls, we repeat a “similar” test without boundaries, in which we offset the initial position of vesicle by 1. This experiment is depicted in the bottom-right plot. In all of the experiments, $\nu = 1$ and $\kappa_b = .5$.*

large vesicles, time steps should be chosen such that the vesicle does not cross the boundary of the domain.

2.4.4 Rheology of a Suspension

In this section, we discuss numerical homogenization for suspension rheology. The effective viscosity of a suspension can be defined as the viscosity of a homogeneous Newtonian fluid that has the same energy dissipation per macroscopic volume element. We derive the effective viscosity as the constant of proportionality between rates of energy dissipation with and without the inclusions. This definition coincides with the one derived by averaging the stress over a volume for particulate flows [62]. Here, we give a brief derivation of average stress tensor in a vesicle suspension.

Average Stress in a Suspension of Vesicles. Consider a suspension of vesicles in an ambient fluid with viscosity μ_0 . Each vesicle has a viscosity contrast ν_p . Let Ω denote an

Table 9: RELATIVE ERROR FOR AREA AND VOLUME OF A SINGLE VESICLE IN THE CONSTRICTED TUBE. *In this table we report errors in area $e_A := |A - A_0|/A_0$ and length $e_L := |L - L_0|/L_0$ for the simulation in Figure 12. For values of $r > 1.25$ a vesicle with $\Delta = .94$ is larger than the gap and cannot pass through it (due to inextensibility and incompressibility).*

		Unbounded	r (Contrast of vesicle's representative length to the gap size)					
		$(r = .87)$.68	.78	.87	.97	1.07	1.12
$\nu = .2$	e_A	4.73e-4	1.30e-4	1.17e-4	8.02e-5	3.18e-5	3.57e-4	1.90e-4
	e_L	1.95e-4	2.47e-4	2.41e-4	2.39e-4	2.42e-4	1.93e-4	1.97e-4
$\nu = 1$	e_A	6.65e-4	2.19e-4	3.69e-4	4.17e-4	4.27e-4	4.30e-4	2.98e-3
	e_L	4.72e-4	5.25e-4	8.21e-4	9.33e-4	1.05e-3	1.19e-3	1.23e-3
$\nu = 5$	e_A	2.78e-4	2.02e-6	6.45e-6	1.46e-5	4.35e-5	1.89e-4	2.67e-4
	e_L	1.03e-4	2.98e-4	1.68e-4	1.05e-4	1.43e-4	1.69e-4	1.95e-4

Table 10: STABLE STEP SIZE FOR A SECOND-ORDER SEMI-IMPLICIT SCHEME FOR A VESICLE IN A CONSTRICTED TUBE.

n	Unbounded ($r = .87$)			$r = .87$			$r = 1.12$		
	$\nu = .2$	1	5	.2	1	5	.2	1	5
32	7e-1	7e-1	7e-1	7e-1	7e-1	7e-1	6e-3	7e-1	7e-1
64	7e-1	7e-1	7e-1	7e-1	7e-1	7e-1	1e-2	7e-1	7e-1
128	7e-1	7e-1	7e-1	7e-1	7e-1	7e-1	3e-2	7e-1	7e-1
256	7e-1	7e-1	7e-1	7e-1	7e-1	7e-1	9e-3	7e-1	7e-1

arbitrary volume containing several vesicles; let $\langle \cdot \rangle := \frac{1}{|\Omega|} \int_{\Omega} \cdot d\mathbf{x}$, and let ω^+ denote the infinitesimally enhanced volume containing vesicles. The average stress tensor $\langle \sigma \rangle$ can be broken into two parts as

$$\langle \sigma \rangle = \frac{1}{|\Omega|} \int_{\Omega} \sigma d\mathbf{x} = \frac{1}{|\Omega|} \int_{\Omega \setminus \omega^+} \sigma d\mathbf{x} + \frac{1}{|\Omega|} \int_{\omega^+} \sigma d\mathbf{x}. \quad (26)$$

By definition $\sigma = -pI + 2\mu D$, where D is the strain rate tensor. Substituting into Equation (26), we have

$$\langle \sigma \rangle = \frac{1}{|\Omega|} \int_{\Omega \setminus \omega^+} -pI + 2\mu_0 D d\mathbf{x} + \frac{1}{|\Omega|} \int_{\omega^+} \sigma d\mathbf{x}, \quad (27)$$

$$= \frac{1}{|\Omega|} \int_{\Omega} -pI + 2\mu_0 D d\mathbf{x} + \frac{1}{|\Omega|} \int_{\omega^+} \sigma + pI - 2\mu_0 D d\mathbf{x}, \quad (28)$$

using the fact $p = -\text{tr}(\sigma)/3$ and the divergence theorem for the second integral we obtain

$$= -\langle p \rangle I + 2\mu_0 \langle D \rangle + \frac{1}{|\Omega|} \int_{\partial\omega^+} [\sigma \mathbf{n} \otimes \mathbf{x} - \frac{1}{3}(\sigma \mathbf{n} \cdot \mathbf{x})I - \mu_0(\mathbf{u} \otimes \mathbf{n} + \mathbf{n} \otimes \mathbf{u})] ds. \quad (29)$$

For the fluid inside each vesicle, we have $\text{div } \sigma = 0$. It follows that

$$0 = \int_{\partial\omega_p^-} \sigma \mathbf{n} \otimes \mathbf{x} ds - \int_{\omega_p^-} \sigma d\mathbf{x} = \int_{\partial\omega_p^-} [\sigma \mathbf{n} \otimes \mathbf{x} - \frac{1}{3}(\sigma \mathbf{n} \cdot \mathbf{x})I - \mu_p(\mathbf{u} \otimes \mathbf{n} + \mathbf{n} \otimes \mathbf{u})] ds. \quad (30)$$

Taking the limit of Equation (29) and Equation (30) to γ_p , subtracting the results, and denoting the trace of $-\langle \sigma \rangle/3$ by P , we obtain

$$\langle \sigma \rangle = -PI + 2\mu_0 \langle D \rangle + \frac{1}{|\Omega|} \sum_p \int_{\gamma_p} [\zeta \otimes \mathbf{x} - \mu_0(1 - \nu_p)(\mathbf{u} \otimes \mathbf{n} + \mathbf{n} \otimes \mathbf{u})] ds. \quad (31)$$

For dilute suspensions, the particle-particle interactions are negligible and the effective stress can be written in terms of flow past an isolated or “reference” vesicle [12, 13]. In simple shear flow, the ambient velocity field is given by $\mathbf{u}(x, y) = [\chi y, 0]^T$, where χ is the shear rate. For a suspension of vesicles in such a flow, we define

$$[\mu] := \frac{\mu_{\text{eff}} - \mu_0}{\phi \mu_0} = \frac{1}{\chi \mu_0 T} \int_{T_i}^{T_e} \langle \sigma_{12}^p \rangle dt, \quad (32)$$

in which $\langle \sigma^p \rangle$ is defined above. $[\mu]$ is referred to as the “intrinsic viscosity”, ϕ is the areal contrast of vesicles, and σ^p is the perturbation in the stress due to presence of vesicles. T_i is the point at which we start the measurement and is chosen such that the reference vesicle has reached a steady state (for $\nu < \nu_c$), T_e is the end of simulation, and $T = T_e - T_i$. In our formulation, the traction jump across the interface is given by $\zeta = \mathbf{f}_b + \mathbf{f}_\sigma = -\kappa_b \mathbf{x}_{sss} + (\sigma \mathbf{x}_s)_s$. Using integration by parts, we obtain

$$\int_{\gamma} \zeta \otimes \mathbf{x} ds = - \int_{\gamma} (\kappa_b \kappa^2 \mathbf{n} \otimes \mathbf{n} + \sigma \mathbf{t} \otimes \mathbf{t}) ds. \quad (33)$$

As a validation of our scheme, in Figure 13 we report the intrinsic viscosity for a dilute suspension of vesicles in simple shear flow. The vesicle is an ellipse with reduced area $\Delta = .75$ and bending modulus of 1. When $0 < \nu < \nu_c$, the vesicle tank-treads. Larger values of ν cause the vesicle to align itself with the flow thus resulting in less resistance. On the onset of tumbling, we observe that the intrinsic viscosity increases.

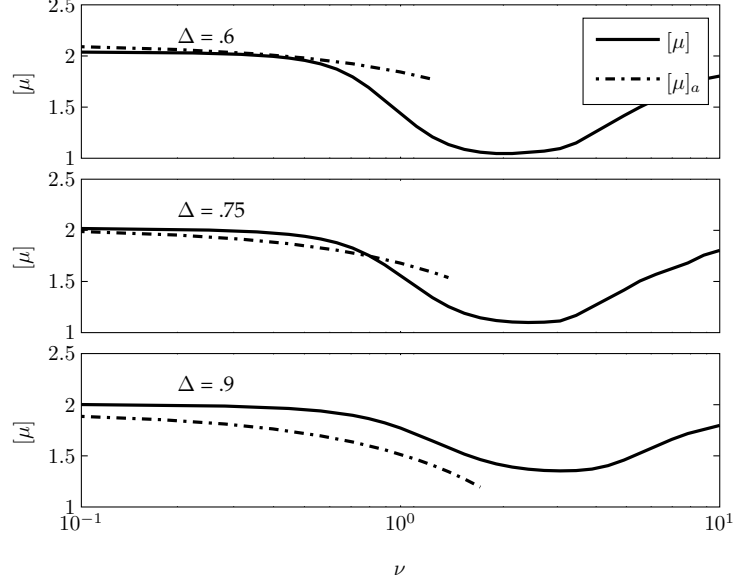


Figure 13: THE INTRINSIC VISCOSITY OF THE HOMOGENEOUS FLUID VS. VISCOSITY CONTRAST OF THE SUSPENSION FOR DIFFERENT VALUES OF REDUCED AREA Δ . As we increase ν , the vesicles align with the flow and the intrinsic viscosity decreases. When $\nu > \nu_c$, tumbling occurs and the reference vesicle does not maintain a fixed orientation. The frequency of this tumbling motion is inversely proportional to ν_c : as we increase ν , vesicles tumble faster, which in turn causes an increase in the intrinsic viscosity $[\mu]$.

Misbah [88] derived an analytical expression for the effective viscosity of a suspension of quasi-spherical vesicles in the tank-treading region. According to [88], we have

$$[\mu]_a := \frac{5}{2} - \frac{(23\nu + 32)\Delta}{16\pi}. \quad (34)$$

The dashed line in Figure 13, corresponds to $[\mu]_a$ obtained from this formula. The qualitative agreement between the analytical result for quasi-spherical vesicles and our numerical simulations for 2D vesicles is good. To our knowledge, there is no analysis for the case of a tumbling motion.

2.5 Conclusions

We proposed a semi-implicit numerical scheme to simulate the motion of inextensible vesicles suspended in bounded or unbounded domains. For several test cases, we have demonstrated, through the use of numerical experiments, that the proposed scheme does not exhibit a mesh-dependent high-order stability constraint on the time-step size. Our scheme exhibits second-order accuracy in time and spectral-accuracy in space. We have presented efficient

low-cost preconditioners to solve the discrete evolution equations by iterative solvers. An additional extension of our work would be to design an algorithm that allows decoupling of the time-step size from the shear rate. We believe, however, that such an algorithm would require the use of nonlinear solvers and contact detection methods that fully couple the vesicle position updates. Such coupling would be more difficult to implement and analyze. Our next step, which we will explore in Chapter 4, is the extension of this scheme to three dimensional flows.

CHAPTER III

VESICLE MIGRATION AND SPATIAL ORGANIZATION DRIVEN BY FLOW LINE CURVATURE

Cross-streamline migration of deformable entities is essential in many problems such as industrial particulate flows, DNA sorting, and blood rheology. Building upon the infrastructure outlined in Chapter 2, through numerical experiment, we have discovered that vesicles suspended in a flow with curved flow lines migrate towards regions of high flow-line curvature, which are regions of high shear rates. The migration velocity of a vesicle is found to be a universal function of the normal stress difference and the flow curvature. This finding quantitatively demonstrates a direct coupling between a microscopic quantity (migration) and a macroscopic one (normal stress difference). Furthermore, simulations with multiple vesicles revealed a self-organization, which corresponds to segregation, in a rim closer to the inner cylinder, resulting from a subtle interaction among vesicles. Such segregation effects could have significant impact on rheology of vesicle flows.

3.1 Introduction

Complex fluids are generally made of rigid or soft particles that are suspended in a Newtonian fluid. Examples of complex fluids include emulsions, polymer solutions, particulate suspensions (many food products belong to this category) and blood. One of the major challenges in understanding the physics of complex fluids is the link between the local microstructure (i.e. spatio-temporal organization of suspended entities) and the macroscopic rheology. Microstructures spontaneously arise in many complex fluids, and may have a dramatic impact on flow properties [36].

A phenomenon that may induce inhomogeneous organization of the suspended entities is lateral or cross-streamline migration. Recall that a single rigid particle immersed in a Newtonian fluid at vanishing Reynolds number Re cannot migrate in the direction transverse to the flow lines [14]. On the contrary, deformable particles have the ability to migrate

cross-streamline, even at $Re = 0$ —if a certain symmetry is broken, for example the centrosymmetry in linear shear flow. Symmetries may be broken due to the presence of walls, gradient in shear rate (e.g., Poiseuille flow), or the presence of flow line curvature (e.g., cylindrical Couette flow).

Cross-streamline migration appears in many applications, such as industrial polymer processing [141], DNA sorting [123], drop dynamics [32], and biofluids. A prominent example of the latter system is blood flow in which cross-streamline migration of erythrocytes may result in ample collapse of blood viscosity, reducing blood flow resistance in microvasculature (Fåhræus-Lindqvist effect).

A common belief is that deformable particles have the tendency to migrate towards regions of low shear rates [40, 59, 65, 119, 133]. In some circumstances, however, the opposite is predicted, for example, the case of drops in a certain range of viscosity contrast between the internal and external fluids [32]. In this chapter, we propose an explanation for these differences and provide quantitative evidence for the case of vesicle flows. Our discussion will allow us to conjecture general principles that can be used to predict lateral migration. The study of multiple vesicles reveals a self organization in a rim.

We carried out simulations in a Taylor-Couette cell by taking vesicles as a model system for the suspended entities. Vesicles constitute a simple model for the description of some features of red blood cell dynamics. We have chosen to simulate the Taylor-Couette system because it is widely used for studying the rheology of complex fluids.

3.2 *Methods*

The numerical simulations are carried out in two dimensions using a boundary integral formulation. The detail of the boundary integral formulation is given in Chapter 2. The main governing equations for the flow of vesicles are Equation (3a) and Equation (3b). The membrane interfacial force, which has contributions from bending energy and local inextensibility, is defined in Section 2.2.3. We used the implementation outlined in Chapter 2 to carry out our numerical experiments. To indicate the accuracy of the simulations let us mention that the vesicle surface is conserved within a relative error of 10^{-6} and *local* contour

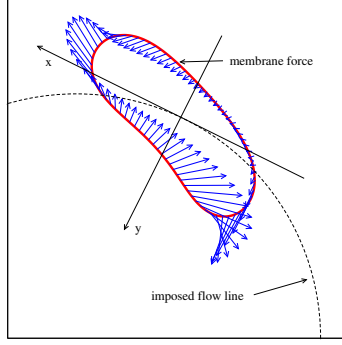


Figure 14: FORCE DISTRIBUTION ON VESICLE MEMBRANE AND LOCAL COORDINATE SYSTEM. The local coordinate system is used for the calculation of N in Equation (35). For this figure, $\Delta = 0.7$ and $\nu = 1$.

length within 10^{-3} .

The normal stress difference is defined as $N = \sigma_{xx} - \sigma_{yy}$, where σ is the stress tensor of the suspension, computed using [12, 71]:

$$N = \frac{1}{A\mu_0\chi} \left[\int_{\gamma} (xf_x - yf_y) ds + 2\mu_0(\nu - 1) \int_{\gamma} (n_x v_x - n_y v_y) ds \right], \quad (35)$$

where A is the vesicle area, μ_0 is the viscosity of suspending fluid, $\chi = -2a/r^2$ is the imposed shear rate, \mathbf{f} denotes the interfacial force, ν denotes the viscosity contrast, \mathbf{n} denotes the normal vector to the surface γ , and \mathbf{v} denotes the velocity. The x, y coordinate system is relative to the instantaneous vesicle position and is defined in Figure 14. The coordinate axes correspond to circumferential and radial directions ($\mathbf{e}_\theta, -\mathbf{e}_r$) and the origin is at the vesicle's center of mass. shear rate. Notice that it depends on the radial position of the vesicle.

3.3 Numerical Experiments

First, we consider a single two-dimensional vesicle immersed in a Newtonian fluid with a velocity field $v_\theta = a/r$, $v_r = 0$ (for a Couette flow, $v_\theta = a/r + br$). This is an unbounded flow. This choice is made in order to exclude any migration due to bounding walls, allowing us to identify the role of curvature in the flow lines. Boundaries are introduced in a second step. Vesicles are initialized at a distance of $10r_0$ from the origin, where $r_0 = \sqrt{A/\pi}$. The length unit is chosen to be r_0 in our simulations. The dimensionless numbers that enter the problem are [67] the reduced vesicle area $\Delta = 2\sqrt{A}/L$, the viscosity contrast ν and the

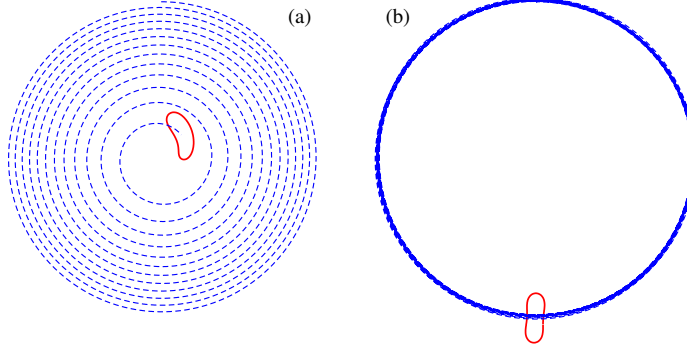


Figure 15: TRAJECTORY OF A VESICLE IN AN UNBOUNDED ROTATIONAL FLOW. *Trajectory and contour of (a) tank-treading vesicle ($\Delta = 0.7$, $\nu = 1$) migrating towards high shear regions and (b) tumbling vesicle ($\Delta = 0.7$, $\nu = 4$) showing no significant radial migration (after 13 revolutions and 8 tumbling periods).*

capillary number $Ca = \mu_0 \chi r_0^3 / \kappa$. In the simulations both Δ and ν are varied. The value of Ca depends on the radial position, while a is fixed to a value $a = -10$ (a weak dependence of vesicle dynamics on this parameter [52] is observed). We have performed three sets of simulations for different $\Delta \in \{0.7; 0.8; 0.9\}$. For every set the range $\nu \in [1, 10]$ is explored, a range that covers both tank-treading and tumbling regimes [16, 52, 69].

3.4 Results

Typical simulation results are shown in Figure 15. Tank-treading vesicles migrate towards the center, while tumbling ones show a negligibly small outward migration. We have found that the migration rate depends on the reduced area and viscosity contrast in a non trivial way.

In Figure 16, we report the migration velocity v^{mig} for different vesicles at fixed initial radial position $r = 10r_0$. Analogous results are obtained for any radial position $3 \leq r/r_0 \leq 10$. In the left panel, the migration velocity is shown as a function of the two independent dimensionless parameters explored in our simulations, namely (Δ, ν) .

The data do not seem to show a simple trend. For example, the lines in Figure 16(a) for migration velocities obtained for different vesicles intersect at some viscosity contrast. This points to the absence of a simple law in this parameter space. We have thus attempted to rationalize these results by evoking basic physical facts that distinguish a simple fluid from a complex one. A particular property of complex fluids is the manifestation of normal

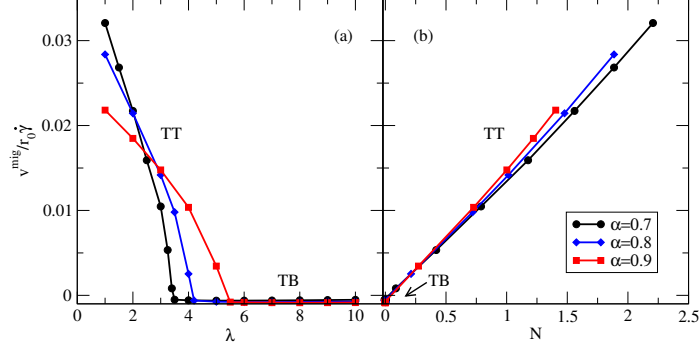


Figure 16: NORMALIZED INWARD MIGRATION VELOCITY. *The Inward migration velocity normalized by $r_0\chi$ as a function of (a) ν and (b) N for different Δ at fixed radial position $r = 10r_0$. Every point corresponds to the (ν, N) pair on the abscissas.*

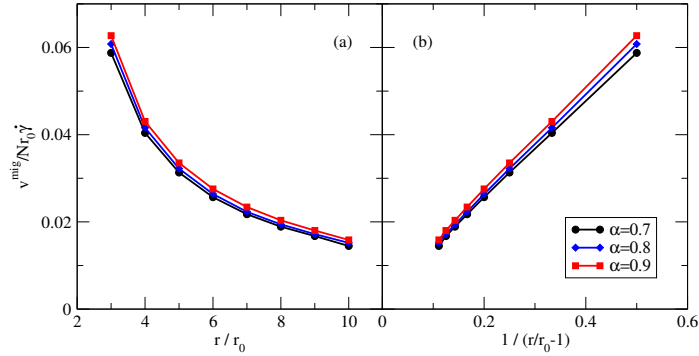


Figure 17: NORMALIZED INWARD MIGRATION VELOCITIES DIVIDED BY NORMAL STRESS DIFFERENCE. *Inward migration velocities divided by $Nr_0\chi$ as a function of (a) r and (b) $1/(r - r_0)$ for different Δ . Every point corresponds to the (ν, N) pair on the abscissas.*

stress difference. We have thus represented the data (Figure 16(b)), in terms of the normal stress difference N measured in the (x, y) coordinate system (see Figure 14 for notations). Interestingly enough, we observe that the data closely collapse on a single master curve, showing that the dynamics does not depend on the control parameters (Δ, ν) independently, but rather on their combination embedded in the function $N(\Delta, \nu)$.

Moreover, Figure 16(b) shows that $v^{mig}/r_0\chi$ is simply proportional to N . This result holds for all the radial positions explored so far, $3 \leq r/r_0 \leq 10$: data collapse is manifested within an error of 10% (or less), and the results are represented with a universal straight line passing through the origin. The small discrepancies are believed to be due to the details of the flow around vesicles with different shapes and orientations.

To gain further insight we have examined the migration velocity as a function of the curvature of the flow. Figure 17(a) shows $v^{mig}/Nr_0\chi$ as a function of the radial position.

This dependence on r is nonlinear. Expressing the results with the help of an appropriate rescaling (Figure 17(b)) reveals that $v^{mig}/N\chi$ is a simple linear function of $\xi := 1/(r/r_0 - 1)$. Note that, approximately, ξ is the flow curvature on the innermost part of the vesicle, which is also the highest curvature among the flow lines passing through the vesicle. This is considered to be due to membrane incompressibility, that propagates stresses along the surface of the vesicle. From the above analysis, we infer the following scaling relation for migration

$$v^{mig} \sim r_0 \dot{\gamma} \xi N \quad (36)$$

This is a key result: a macroscopic measure of N (which may be a very complex function of various parameters) directly leads to the determination of the (microscopic) migration velocity. One might ask why should migration be dictated by normal stress difference at all. To answer this question, one may consider the composite fluid and denote its spatial and temporal averaged stress by $\boldsymbol{\sigma}$ (very much like the definition of the classical stress, as used in Equation (35)). Let us assume stationary, circular motion, enjoying symmetry with respect to the angle θ . Using momentum conservation in polar coordinates one can show that [19]:

$$\frac{\partial \sigma_{rr}}{\partial r} = \frac{1}{r} [-\rho v_\theta^2 + N] \quad (37)$$

where ρ is the fluid density, $N = \sigma_{\theta\theta} - \sigma_{rr}$, $1/r$ is the flow line curvature, and $-\rho v_\theta^2/r$ is the inertia term, which is absent in our case. If $N \neq 0$, a radial stress gradient takes place, resulting in an inward force pushing the fluid towards the center if $N > 0$. No such simple result holds in flows which do not exhibit flow line curvature. This result shows that both N and the flow curvature cause inward motion, in accord with Equation (36). We have performed simulations in a parallel flow having the same velocity profile as the irrotational vortex, that is $v_x(y) = 1/y$ (in Cartesian coordinates (x, y)). Contrary to the Couette flow, in parallel flow vesicles migrate toward regions of low shear rate despite the fact that $N > 0$. This points to the conclusion that inward migration in the Couette set-up is due to the curvature of the flow lines rather than to a shear gradient.

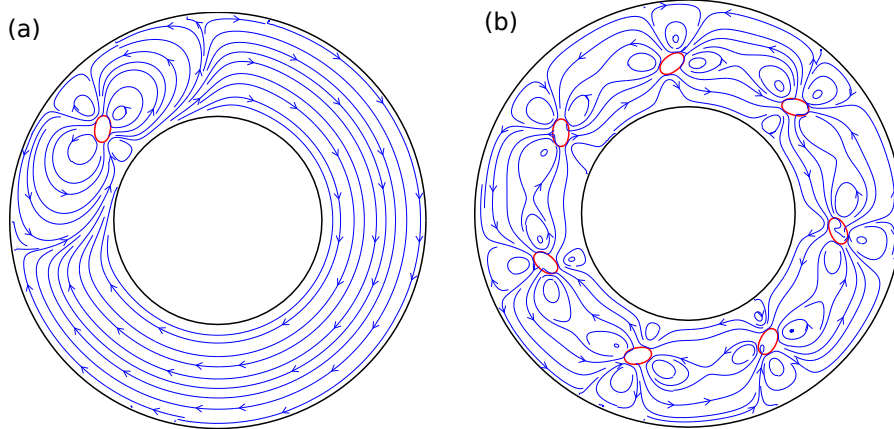


Figure 18: SPATIAL ORGANIZATION OF VESICLES IN THE COUETTE APPARATUS. *Equilibrium configurations in a Taylor-Couette device for (a) a single vesicle and (b) several vesicles (volume fraction $\approx 2\%$). The lines represent the induced flow. Spontaneous organization in (b) is due to inward migration and vortex interaction.*

Tank-treading vesicles show positive normal stresses and they migrate inwards. In the tumbling regime, we have found $N \simeq 0$ (averaged over a tumbling period), and a negligibly small migration. Moreover, N vanishes for tank-treading vesicles when approaching the transition to tumbling or when the shape is close to a sphere [38, 52]. This is consistent with the fact that $v_{mig} \rightarrow 0$ with increasing ν or Δ (Figure 16).

Having identified the basic phenomena for migration of a single vesicle, we are now in a position to address the question of the impact of this feature on the organization of a collection of vesicles. Here we address the case of a sufficiently dilute suspension. In these simulations we have included confining boundary conditions [110, 135], in order to address a realistic situation. Starting from various initial configurations (including a randomly ordered) we have found that the mutual interactions between vesicles lead to a nontrivial spatial organization. We present the results for a volume concentration $\phi \approx 2\%$ in Figure 18 (we have performed simulations with concentrations between $1\% \lesssim \phi \lesssim 6\%$, leading to the same conclusion). After a transient the vesicles exhibit a spatial order: they organize themselves in a rim by keeping the same interdistance. The rim radius is (within numerical uncertainties) very close to the final position of a single vesicle (Figure 18). A single vesicle stops when the inward migration compensates the lift force due to the inner

cylinder. The organization in a rim which has the same radius as that dictated by the final position of a single vesicle is not obvious: indeed, the fact that the vesicles select the same interdistance is a clear indication for their significant mutual interactions, and despite this effect the terminal position does not seem to be affected.

For a better understanding, we have analyzed the behaviour of the flow lines. A single vesicle creates two vortices as shown in Figure 18(a). The size of the vortices is of about a quarter of the circumference. We thus expect vesicles to interact significantly when their number M approaches 4. This is confirmed by our simulations that show disorder for $M < 4$ and order for $M \geq 4$. For $M \geq 4$ vesicles keep order because deviations would cause restoring forces due to vortex interactions. For all explored volume fractions we found persistence of order as shown in Figure 18(b).

Finally, in view of the generality of the arguments presented for migration, it is natural to attempt to extend them to other types of complex fluids. For example, experimental measurements of migration are known for soft entities in Taylor-Couette and cone-plate rheometers: drops [31, 61] and polymers in dilute suspensions [3, 26] either migrate towards the center (cone-plate) or adopt an equilibrium position that lies between the gap centre-line and the inner cylinder (Taylor-Couette), which corresponds to high shear rate regions. Because drops, polymers, and vesicles have quite different properties, their similar behavior with respect to migration supports our conjecture that the basic mechanisms governing migration are independent of the mechanical details of the suspended entity and depend only on the flow curvature and N . An interesting line of research would be to study the evolution of order for higher concentrations, and provide a detailed link between microstructure and rheology.

CHAPTER IV

A FAST ALGORITHM FOR SIMULATING VESICLE FLOWS IN THREE DIMENSIONS

In this chapter, we present algorithms for the simulation of general three-dimensional vesicle flows in unbounded domains, extending our work presented in Chapter 2 for the two dimensional domains. The main components of the algorithm are similar in spirit to the two dimensional case (spectral approximation in space, semi-implicit time-stepping scheme), important new elements need to be introduced for a 3D method. In particular, spatial quantities are discretized using spherical harmonics, and quadrature rules for singular surface integrals need to be adapted to this case; an algorithm for surface reparametrization is needed to ensure sufficient of the time-stepping scheme, and spectral filtering is introduced to maintain reasonable accuracy while minimizing computational costs. To characterize the stability of the scheme and to construct preconditioners for the iterative linear system solvers used in the semi-implicit time-stepping scheme, we perform a spectral analysis of the evolution operator on the unit sphere.

By introducing these algorithmic components, we obtain a time-stepping scheme that, in our numerical experiments, is unconditionally stable. We present results to analyze the cost and convergence rates of the overall scheme. To illustrate the applicability of the new method, we consider a few vesicle-flow interaction problems: a single vesicle in relaxation, sedimentation, shear flows, and many-vesicle flows.

4.1 *Introduction*

As we mentioned in Chapter 1, vesicle-inspired mechanical models can be used to approximate red blood cell mechanics. For example, at equilibrium (i.e., in a quiescent fluid), healthy red blood cells have a biconcave shape that corresponds to a minimal membrane bending energy. Under nonequilibrium conditions, as experienced in a simple shear flow, the best-studied features of red cell dynamics, formation of tank-treading ellipsoids and

tumbling motion, are shared with vesicles [16, 69, 73].

The vesicle evolution dynamics is characterized by a competition between membrane elastic energy, surface inextensibility, vanishing in-plane shear resistance, and non-local hydrodynamic interactions. Simulation of vesicles is a challenging nonlinear free boundary value problem, not amenable to analytical solutions in all but a few simple cases; numerical simulations and experiments are the only options for the quantitative characterization of vesicle flows.

In this chapter, we present an algorithm for the simulation of general three-dimensional vesicle flows, extending our work for two dimensional domains, presented in Chapter 2, and 3D axisymmetric vesicles [134]. To demonstrate the capabilities of our code, we depict a few time-snapshots from a twenty-vesicle simulation in Figure 19.

To establish notation for this chapter, we start with the standard PDE formulation of the problem:

$$\begin{aligned}
-\Delta \mathbf{v} + \nabla p &= \mathbf{0} \text{ in } \mathbb{R}^3 \setminus \gamma && \text{(conservation of momentum in bulk fluid),} \\
\operatorname{div} \mathbf{v} &= 0 \text{ in } \mathbb{R}^3 \setminus \gamma && \text{(conservation of mass),} \\
\operatorname{div}_\gamma \mathbf{v} &= 0 \text{ on } \gamma && \text{(surface inextensibility),} \\
\llbracket -p\mathbf{n} + (\nabla \mathbf{v} + \nabla \mathbf{v}^T)\mathbf{n} \rrbracket &= \mathbf{f}_b + \mathbf{f}_\sigma \text{ on } \gamma && \text{(balance of momentum on the membrane),} \\
\mathbf{v} - \mathbf{v}_\infty &\rightarrow 0, \text{ if } \|\mathbf{x}\| \rightarrow \infty && \text{(far field condition),} \\
\frac{\partial \mathbf{x}}{\partial t} &= \mathbf{v} \text{ on } \gamma && \text{(membrane material point motion).}
\end{aligned} \tag{38}$$

In the first equation, \mathbf{v} is the fluid velocity, p is the pressure, and γ is the vesicle membrane. In the third equation, $\operatorname{div}_\gamma$ is the surface divergence operator. In the fourth equation, $\llbracket q \rrbracket$ denotes the jump of a quantity q across the vesicle membrane, \mathbf{n} is the normal to the vesicle membrane, \mathbf{f}_b is a force due to bending resistance of the vesicle membrane, and \mathbf{f}_σ is a force (tension) that enforces the surface inextensibility constraint. In the momentum balance equation, we assume that the mass of the membrane is negligible. In the last equation, \mathbf{x} is the position of a material point on the membrane. In addition, the last equation encapsulates the fact that $\llbracket \mathbf{v} \rrbracket = 0$ due to conservation of mass (normal direction) and the

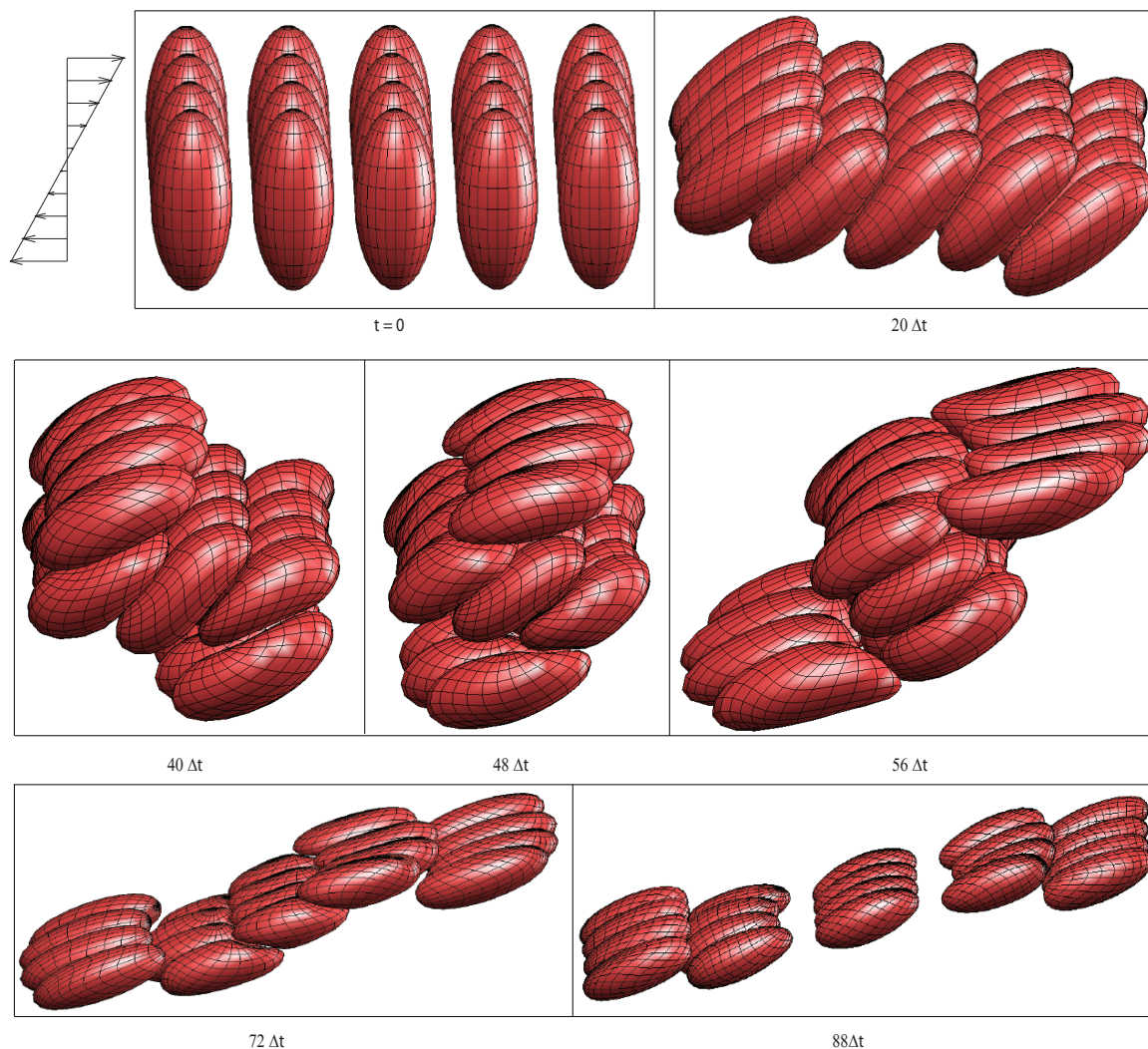


Figure 19: SNAPSHOTS OF TWENTY VESICLES SUSPENDED IN A SIMPLE SHEAR FLOW. *Initially, each vesicle has a non-equilibrium 2-1 ellipsoidal shape and they are arranged in a rectangular lattice. The shear rate is $\chi = 18$. The number of spatial discretization points is 338 ($p = 12$) and the average wall-clock time is 110 seconds per time-step on a 2.33 GHz Xeon workstation with 4GB of RAM.*

non-slip condition (tangent plane).

A mathematically-equivalent formulation is based on an explicit form of the solution of Stokes equation in the free space:

$$\begin{aligned} \mathbf{v} &= \mathbf{v}_\infty + \mathcal{S}[\mathbf{f}_b + \mathbf{f}_\sigma] && \text{on } \gamma, \\ \operatorname{div}_\gamma \mathbf{v} &= 0 && \text{on } \gamma, \\ \frac{\partial \mathbf{x}}{\partial t} &= \mathbf{v} && \text{on } \gamma. \end{aligned} \tag{39}$$

where \mathcal{S} is the single-layer Stokes operator, defined in Section 4.3. The first equation encapsulates five equations of the PDE formulation: conservation of momentum in the bulk and on the membrane, conservation of mass, and the far-field boundary conditions for the fluid.

4.1.1 Contributions

We extend the ideas presented in [135] to the general case of vesicles in three dimensions. The main features of the method of [135] are an integral equation formulation, spectral discretization in space, and a semi-implicit time-stepping scheme. In [134], an extension of [135] to the axisymmetric case is proposed. There are several challenges specific to numerical simulation of non-axisymmetric vesicles flows in 3D:

- (i) What should the spatial discretization scheme be to maximize accuracy, computational efficiency, and numerical stability?
- (ii) Unlike 2D, using a Lagrangian frame of reference for spatial discretization of vesicle boundary results in extreme distortions, leading to numerical instability; how can this instability be controlled?
- (iii) Bending and tension forces in the 3D case have a more complex nonlinear form; can we design a time-stepping scheme that addresses the severe stiffness while having similar per-time-step complexity to the explicit scheme (similar to our schemes for 2D and axisymmetric flows [134, 135])?

A summary of our contributions is as following:

- We develop a high-order spatial discretization for inextensible vesicles based on spherical harmonics (Section 4.4), combining and extending a number of previously proposed techniques (Section 4.2), including extending the quadrature scheme of [58] to the Stokes kernel (Section 4.4.1);
- We propose a particular linearization for a semi-implicit time-stepping scheme along with a preconditioning scheme (Section 4.5.3);
- We analyze the stiffness of Equation (39) (Section 4.5) and use it to precondition the linear solves in the semi-implicit time scheme;
- We propose a reparametrization scheme for stabilization of time-stepping, and analyze its accuracy and stability (Section 4.6);
- We verify the numerical scheme by comparing vesicle dynamics obtained with the proposed method against shapes obtained using our axisymmetric solver , which is based on an entirely different discretization scheme, (Section 4.7);
- We present results on computing the equilibrium shapes of dilute suspension under shear flow, examine a two-vesicle interaction problem, study the sedimentation of a vesicle, and provide an example of a simulation with multiple vesicles (Section 4.7).

Synopsis of our method. Our method is based on Lagrangian tracking of spectral collocation points placed on the membrane of the vesicle combined with a surface reparametrization scheme. We achieve spectral accuracy in space by discretizing using spherical harmonics, which we use for the representation of the membrane and its spatial derivatives, for quadratures, and for anti-aliasing (Section 4.4). For weakly-singular integrals, we use the scheme proposed in [58], which enables high-accuracy simulations with a small (compared to low-order schemes) number of points per vesicle. For the position update in time, we use two variants of a semi-implicit marching scheme first derived for advection-diffusion equations [5] and then applied on integral-equation based fluid-structure interaction problems in [129].

The time-marching scheme requires the solution a linear system of equations at each time step, which we perform using a Krylov iterative method (GMRES [114]). The problem of poor conditioning is addressed by a preconditioner based on the analytically obtained spectrum of the operators in Equation (39) for the special case of a unit sphere. Vesicle-vesicle interactions can be carried out using the kernel independent fast multipole method [144]. In this chapter, we focus on the mathematical formulation and not on performance and parallelization. We report, preliminary results on a high-performance parallel implementation of this method in Chapter 6.

In all, we are able to achieve high accuracy while using a small number of unknowns per vesicle for the spatial discretization and taking large time steps with a relatively low computational cost per time step. Pseudocode that summarizes the scheme can be found in Section 4.5.3.

4.1.2 Limitations

We restrict our attention to suspensions of vesicles in fluids with unbounded domains. We have ignored inertial terms, so the overall method is restricted to low Reynolds numbers. We only consider spherical-topology vesicles and we do not allow for topological changes, which are present in many biophysical phenomena involving vesicles.

In our examples, we assume that the interior and exterior of the vesicles are filled with the same liquid. The algorithm extends to the case of viscosity contrast with the introduction of double layer potentials [104]. Additional convolution with a *stresslet* is required which can be computed using the singular quadrature rule described in this paper. The detailed algorithm for two-dimensional flow was presented in Chapter 2, in which in addition to viscosity contrast we introduce confined flows with Dirichlet boundary conditions. The formulation for the three dimensional case is similar. We will report the extension in Chapter 5.

An important limitation of our scheme is the lack of adaptivity (both in space and time). This lack of adaptivity can cause vesicle-vesicle collisions, which are not possible in the mathematical model we use. Indeed, one can easily construct simulations where without

a significant increase of the surface discretization size, our code fails to resolve inter-vesicle interactions accurately. This is an open problem and we are currently working on addressing this issue. We have also observed a dependence of the stable time step size on the shear rate.

4.1.3 Outline of the Chapter

In the next section, we review the related work on numerical methods for vesicle flows in three dimensions. In Section 4.3, we present the overall integral equation formulation. In Section 4.4 and Section 4.5, we present the spatial and temporal discretization schemes respectively. In Section 4.6, we discuss the reparametrization scheme and we conclude in Section 4.7 with results from numerical experiments.

4.2 Related Work

Several of the algorithmic components of the proposed method have appeared in the literature. This work is the extension of our previous works on the dynamics of vesicles in two and three dimensions [110, 134, 135, 136]. Few other closely related papers are the work of Graham and Sloan [58] on singular quadratures for the scalar Helmholtz operator, Zinchenko and Davis [155] on surface reparametrization schemes for drops and deformable particles, and the work of Zhao et al. [148] on boundary integral equation based simulations of red blood cells in shear flow, presenting a spherical harmonic discretization of membranes with bending and shear resistance and explicit time discretization for particulate flows of this type. To our knowledge, there is no prior work on implicit or semi-implicit schemes for locally inextensible vesicles.

We focus our discussion of related work on *three-dimensional numerical methods*; we do not attempt to perform a comprehensive review of experimental and numerical studies of vesicle flows.

Three-dimensional vesicle flows. Stokesian particulate flow problems are solved using a variety of methods, including unstructured finite element methods and Cartesian grids (immersed boundary/interface, fictitious domain, phase-field and level sets). For a brief

review of these methods, in comparison with integral equation formulations, see [20]. Several groups have considered stationary shapes of three-dimensional vesicles using semi-analytic [22, 28, 122], or numerical methods like the phase-field [43, 44] and finite element methods [48, 85]. These approaches are based on a constrained variational approach (i.e., minimizing the bending energy subject to area and volume constraints) and have not been used for resolving fluid-structure interactions.

We restrict our discussion to integral equation formulations, which, for certain classes of problems, offer certain computational advantages compared to stencil-based methodologies (finite differences or finite elements). Pozrikidis [104] reviews the work on boundary integral formulations for particulate flows prior to the year 2000.

The main features of an integral-equation vesicle flow solver are the underlying formulation (direct vs indirect), the discretization of the surface, the quadrature scheme, and the time-stepping scheme. For Lagrangian vesicle surface discretization, a form of adapting the sampling of the surface to the deformed shape (reparametrization) is needed. To our knowledge the majority of particulate flow solvers in three dimensions use explicit schemes with the exception of the work of Dimitrakopoulos [39]. In that work, a Jacobian-free, finite-difference based Newton method was used. Despite its good convergence, the cost per iteration is somewhat higher than our approach as the method requires multiple evaluations for each matrix-vector multiplication and no preconditioning was used. In the remainder of this section we discuss the related work to each of the key algorithmic component in our work.

Integral equation formulation. Several authors [16, 29, 30, 73, 125] have used direct integral equation formulation for simulating vesicle flows. This formulation results in a single-layer potential for flows with no viscosity contrast and a combination of a single-layer and a double-layer for flows with viscosity contrast [73, 104, 125, 150]. A particular feature of vesicle flows is the local surface inextensibility constraint. Kraus et al. [73] and Sukumaran and Seifert [125] enforce the constraint by a penalty formulation. The penalty parameter results in artificial stiffness and loss of accuracy with respect to the constraint.

Imposing the constraint using a Lagrange multiplier removes the stiffness and resolves the inextensibility constraint with spectral accuracy [107, 134, 150].

Surface discretization. In particulate flows, the surface is typically represented by a triangulation. Once such triangulation is available, either collocation or Galerkin schemes can be used to discretize the integral equation [7]. Examples include [73, 106, 113, 125]. High-order B-spline methods were used by Zhou and Pozrikidis [150]. Pozrikidis [105] proposes a boundary element scheme for deformable capsules with bending and in-plane shear resistance based on six-node quadratically curved triangles and a low-order accurate integration method. This discretization requires tracking of 1026 points per surface to get 0.01% error in the total volume of the particle. Spherical harmonics have been used frequently in boundary integral equations [7]. It enables spectrally accurate integration and differentiation. Zhao et al. [148] use a spherical harmonic discretization for deformable capsules with bending and shear resistance, but no inextensibility constraint was imposed.

Singular quadratures. Bruno and Kunyansky [27] proposed a spectral integration scheme for weakly-singular integrals. Ying et al. [145] extended that algorithm to arbitrary-geometry smooth surfaces. Although asymptotically optimal, this scheme is rather expensive as it requires the use of partition of unity functions, for which derivative magnitudes rapidly increase with order and as a consequence, a relatively large number of points is needed for good approximation. This scheme is used by Zhao et al. [148]. To reduce the discretization size without compromising accuracy, we use the scheme proposed by Graham and Sloan [58]. It is only applicable on smooth surfaces of genus zero but it is quite fast. The schemes of Bruno and Kunyansky [27] and Graham and Sloan [58] are compared by Ganesh and Graham [51] and it is shown that the latter scheme is more accurate for small numbers of discretization points.

Simulation of concentrated suspensions of vesicles requires evaluation of nearly-singular integrals. Although we could use the integration scheme proposed by Ying et al. [145], we opted for a simple upsampling-based quadrature for this work.

Reparametrization. In the context of deformable surfaces, the need and methods for maintaining grid quality is ubiquitous. All the resampling methods known to us in this context focus on mesh-based surface representations (primarily piecewise-linear, but also higher-order, e.g. [75]) and we are not aware of any methods designed for spectral discretizations.

Many approaches are based on various types of tangential mesh smoothing, often combined with connectivity adaptation. For simulation of multiple interacting drops, Loewenberg and Hinch [82] proposed a heuristic formula for artificial tangential velocity reducing mesh distortion. Zinchenko et al. [156] observed that the technique of Loewenberg and Hinch [82] leads to instabilities in simulations with gravity induced motions, and constructed a tangential velocity field by global minimization of the sum of squares of the rates of change of the distances between adjacent vertices (tangency of the field is enforced as a constraint), the first instance of the (*passive stabilization*) approach. This method was further developed by introducing different objectives for minimization in Zinchenko et al. [157] and Zinchenko and Davis [151], which adapted the sampling density to curvature and controlled triangle quality. The more complex versions of passive stabilization methods tend to be quite expensive as relatively complex nonlinear energies are minimized using gradient descent. A simpler version of passive stabilization suitable for moderate deformations was used by Zinchenko and Davis [152, 153, 155], including a distance-preserving term, and a triangle “compactness” term penalizing elongated triangles. Later, Zinchenko and Davis [154] supplemented the same simplified energy by node redistribution, aiming to equalize mesh edge lengths. A modification of this method reducing maximal deviation from average is described in [154]. In [37] a damped relaxation with the energy gradient projected to the surface is used. Additionally, nodes are added by triangle quadrisection (also to minimize the energy) and removed by edge collapse.

An advancing-front method for remeshing of quadratic triangular elements (originally proposed by Lo [80]) adapting triangle size to local criteria such as curvature is described in [75] and is extended in a number of subsequent papers to surfaces, in particular in [84]. Tryggvason et al. [131] briefly describe an algorithm for adapting a mesh to an evolving fluid

interface, which uses edge length as a criterion for bisection and edge collapses to remove small elements.

A set of algorithms for anisotropic mesh adaptation, including anisotropic smoothing, applicable to the problem of approximating deforming surfaces, was presented by Jiao et al. [63].

Feng and Klug [48] use subdivision finite elements, and demonstrate that local inextensibility combined with a sufficiently accurate Galerkin method makes it possible to simulate moderate deformations with fluid membranes (without fluid interaction). Ma and Klug [85] mention that local inextensibility combined with large deformations still leads to instability, and describe how viscous stabilization of the mesh can be achieved by minimizing an energy measuring the deviation of edge length from previous values. This approach is suitable for computing equilibrium shapes in the absence of external forces, but not for dynamic simulations.

Aliasing errors. While various nonlinear geometric quantities, such as mean and Gaussian curvatures, can be evaluated accurately pointwise, they may contain high frequencies which cannot be resolved properly with a small number of sampling points, leading to *aliasing errors* [15, 130], i.e., discretization errors that can be resolved by increasing the resolution. As it was experienced by Veerapaneni et al. [135], in two dimensions 64 points were sufficient to resolve the vesicle interactions and aliasing was not an issue. In three dimensions, however, we need to consider aliasing errors for two reasons: first, we would like to enable accurate simulations with very few spectral collocation points (say about 100 points) to allow simulations with very large number of vesicles. In this regime, filtering is required to deal with unresolved frequencies. Second, unlike capsules endowed with strong shear resistance, vesicles are fluid membranes and as a result they experience excessive deformations that lead to significant amplification of aliasing errors. Zhao et al. [148] experimentally show that, typically, an aggressive upsampling of three to four times the current order of spherical harmonics is needed for accurate evaluation of derivatives.

4.3 Formulation

In the rest of the paper we assume that the fluids in the interior and exterior of the vesicle have the *same* dynamic viscosity μ . The case of vesicles with viscosity contrast is formulated in Chapter 5.

Notation. Before we state the overall formulation of the problem, let us set the notation. We consider a single vesicle first:

- With γ we denote the vesicle membrane (or vesicle boundary).
- For a vector field $\mathbf{f}(\mathbf{y})$, $\mathbf{y} \in \gamma$, and an arbitrary point $\mathbf{x} \in \mathbb{R}^3$ we define the single-layer Stokes operator as

$$\mathcal{S}_\gamma[\mathbf{f}](\mathbf{x}) := \int_\gamma G(\mathbf{x}, \mathbf{y}) \mathbf{f}(\mathbf{y}) d\gamma(\mathbf{y}), \quad G(\mathbf{x}, \mathbf{y}) := \frac{1}{8\pi\mu} \left(\frac{1}{\|\mathbf{r}\|} \mathbf{I} + \frac{\mathbf{r} \otimes \mathbf{r}}{\|\mathbf{r}\|^3} \right), \quad \mathbf{r} = \mathbf{x} - \mathbf{y}, \quad (40)$$

where G is the free-space Green's function for the Stokes equation, $\|\cdot\|$ is the Euclidean norm in \mathbb{R}^3 , and \mathbf{I} is the 3×3 identity operator.

- By $\text{div}_\gamma(\mathbf{x})$, $\text{grad}_\gamma(\mathbf{x})$, and $\Delta_\gamma(\mathbf{x})$ we denote the surface divergence, surface gradient, and surface Laplacian (Laplace-Beltrami operator) respectively evaluated on $\mathbf{x} \in \gamma$.
- By $W(\mathbf{x})$, $\mathbf{n}(\mathbf{x})$, $H(\mathbf{x})$ and $K(\mathbf{x})$ we denote the area element, the unit normal to γ , the mean curvature, and the Gaussian curvature defined on a point $\mathbf{x} \in \gamma$. We give explicit formulas in Section 4.8.
- Finally, let us define the bending \mathbf{f}_b and tension \mathbf{f}_σ forces on a point $\mathbf{x} \in \gamma$:

$$\begin{aligned} \mathbf{f}_b(\mathbf{x}) &= -\kappa_B \left(\Delta_\gamma(\mathbf{x}) H(\mathbf{x}) + 2H(\mathbf{x})(H(\mathbf{x})^2 - K(\mathbf{x})) \right) \mathbf{n}(\mathbf{x}), \\ \mathbf{f}_\sigma(\mathbf{x}, \sigma) &= \sigma \Delta_\gamma(\mathbf{x}) \mathbf{x} + \text{grad}_\gamma(\mathbf{x}) \sigma, \end{aligned} \quad (41)$$

where κ_B is the surface's bending modulus. These are obtained by taking the L^2 -gradient of the surface energy $\mathcal{E} = \int_\gamma \frac{1}{2} \kappa_B H^2 + \sigma d\gamma$. See [140] for the derivation.

In the rest of the paper, we will frequently suppress the dependence of div_γ , grad_γ , Δ_γ , H , K , \mathbf{f}_b , \mathbf{f}_σ , \mathbf{n} on the evaluation point, which should be clear from the context. Following [135] the evolution of a point \mathbf{x} on the vesicle's boundary γ is governed by

$$\dot{\mathbf{x}} = \mathbf{v}_\infty(\mathbf{x}) + \mathcal{S}[\mathbf{f}_b + \mathbf{f}_\sigma](\mathbf{x}) \quad (42)$$

$$\text{div}_\gamma(\mathcal{S}[\mathbf{f}_\sigma]) = -\text{div}_\gamma(\mathbf{v}_\infty + \mathcal{S}[\mathbf{f}_b]),$$

where \mathbf{v}_∞ is a specified far-field velocity.

4.4 Spatial Scheme

We use spherical harmonic expansions to represent the surface and the interfacial forces. For force evaluation we need to apply forward and inverse harmonic transforms, as well as differentiate and integrate functions on the surface. In this section, we describe our surface representation, spectral differentiation and quadratures used to integrate functions on the surface, both smooth and singular, with singularity due to multiplication by the Stokes kernel. We conclude with a brief discussion on quadratures for near-singular vesicle-vesicle interactions.

Spherical harmonics. Let γ be a smooth surface of spherical topology and let $\mathbf{x} : U \rightarrow \gamma$ be a parametrization of γ . The domain U is the rectangle $\{(u, v) | u \in [0, \pi], v \in [0, 2\pi)\}$; u parametrizes the latitude and v parametrizes the longitude. A *scalar spherical harmonic* function of degree n and order m is given by [24]

$$Y_n^m(u, v) = \frac{1}{\sqrt{2\pi}} \bar{P}_n^m(\cos u) e^{imv}, \quad \text{where} \quad \bar{P}_n^m(t) := \sqrt{\left(n + \frac{1}{2}\right) \frac{(n-m)!}{(n+m)!}} P_n^m(t). \quad (43)$$

Here, $|m| \leq n$, P_n^m denote the associated Legendre functions of degree n and order m , and \bar{P}_n^m denote the corresponding normalized associated Legendre functions. Spherical harmonics form an orthonormal basis for square-integrable functions defined on the unit sphere: any scalar function $\phi \in L^2(\mathbb{S}^2)$ can be expanded as

$$\phi(u, v) = \sum_{n=0}^{\infty} \sum_{m=-n}^n \hat{\phi}_n^m Y_n^m(u, v), \quad (44)$$

$$\text{where } \hat{\phi}_n^m = \int_0^{2\pi} \int_0^\pi \phi(u, v) \overline{Y_n^m(u, v)} \sin u \, du \, dv. \quad (45)$$

For $\phi \in C^\infty(\mathbb{S}^2)$, the finite-term approximation

$$\phi(u, v) \approx \sum_{n=0}^p \sum_{m=-n}^n \hat{\phi}_n^m Y_n^m(u, v) \quad (46)$$

is spectrally convergent [95].

Surface representation. The vesicle boundary γ is described by a set of spherical harmonic coefficients $\{\hat{\mathbf{x}}_n^m | n = 0, \dots, p; m = -n, \dots, n\}$ so that for all $\mathbf{x} \in \gamma$ we have:

$$\mathbf{x}(u, v) = \sum_{n=0}^p \sum_{m=-n}^n \hat{\mathbf{x}}_n^m Y_n^m(u, v), \quad u \in [0, \pi], \quad v \in [0, 2\pi). \quad (47)$$

Each Cartesian component of the position vector \mathbf{x} is expanded as a real-valued function.

In total, we need $(p+1)^2$ coefficients for each Cartesian component.

Forward and inverse transforms. A standard choice for spectrally accurate integration of a function represented by an order p spherical harmonic expansion is to use the trapezoidal rule for v and Gaussian quadrature rule for u : we choose $2p+2$ equispaced nodes in the v -direction $\{v_k = \frac{\pi k}{p}\}_{k=0}^{2p+1}$ and $p+1$ nodes along the u -direction $\{u_j\}_{j=0}^p$ given by

$$u_j = \cos^{-1}(t_j), \quad \text{where } t_j \in [-1, 1]. \quad (48)$$

Here t_j 's are $(p+1)$ -point Gauss-Legendre quadrature nodes. On this grid a standard convergence estimate holds:

Theorem 4.4.0.1. (Quadrature rule for regular integrals) *For any smooth function f defined on a C^∞ surface γ of spherical topology, the quadrature rule*

$$\int_\gamma f d\gamma = \sum_{j=0}^p \sum_{k=0}^{2p+1} w_{jk} f(u_j, v_k) W(u_j, v_k), \quad \text{where } w_{jk} = \frac{\pi}{p} \frac{\lambda_j}{\sin u_j} \quad (49)$$

is superalgebraically convergent with p . Here, λ_j are the Gaussian quadrature weights and $W(u_j, v_k)$ is the infinitesimal area element of γ .

One can use fast transforms for both longitude (Fast Fourier Transform) and latitude (Fast Legendre Transform [FLT]) so that the spherical harmonic coefficients can be computed in $\mathcal{O}(p^2(\log p)^2 + p^2 \log p)$ (see [89])¹. The pseudocode for the forward spherical harmonic transform is given in Algorithm 4.4.1. A similar algorithm can be used for the inverse transform.

Algorithm 4.4.1 FORWARD SPHERICAL HARMONICS TRANSFORM.

```

for  $j = 0$  to  $p$  do
   $\hat{\mathbf{x}}^m(u_j) = \int_0^{2\pi} \mathbf{x}(u_j, v) e^{-imv} W(u_j, v) dv, \quad \text{for } |m| \leq p$ 
   $\approx \frac{\pi}{p} \sum_{k=1}^{2p} \mathbf{x}(u_j, v_k) e^{-imv_k} W(u_j, v_k)$  // trapezoidal rule and FFT
end for // Can be done in  $\mathcal{O}(p^2 \log p)$ 
for  $m = -p$  to  $p$  do
  for  $n = |m|$  to  $p$  do
     $\hat{\mathbf{x}}_n^m = \int_0^\pi \bar{P}_n^m(u) \hat{\mathbf{x}}^m(u) du$ 
     $\approx \sum_{j=0}^p (\lambda_j / \sin u_j) \bar{P}_n^m(u_j) \hat{\mathbf{x}}^m(u_j)$  // Gaussian quadrature and FLT
  end for
end for // Can be done in  $\mathcal{O}(p^2(\log p)^2)$ 

```

Computing derivatives. We compute derivatives of functions defined by samples on γ by applying the forward spherical harmonic transform and using standard differentiation formulas in the spectral domain:

$$\mathbf{x}_u = \sum_{n=0}^p \sum_{m=-n}^n \hat{\mathbf{x}}_n^m (Y_n^m)_u, \quad \mathbf{x}_v = \sum_{n=0}^p \sum_{m=-n}^n \hat{\mathbf{x}}_n^m (Y_n^m)_v \quad (50)$$

where the derivatives of spherical harmonics are:

$$\frac{\partial^k Y_n^m}{\partial v^k} = (im)^k Y_n^m. \quad (51)$$

$$(Y_n^m)_u = \sqrt{(n-m)(n+m+1)} e^{-iv} Y_n^{m+1} + m \cot u Y_n^m, \quad (52)$$

$$\begin{aligned} (Y_n^m)_{uu} &= (n+m+1) \sqrt{(n-m)(2+m+n)} e^{-2iv} Y_n^{m+2} + \\ &\quad (2m+1) \sqrt{(n-m)(n+m+1)} e^{-iv} \cot u Y_n^{m+1} - m^2 Y_n^m. \end{aligned} \quad (53)$$

For smooth \mathbf{x} this numerical differentiation scheme is spectrally accurate because the only approximation made is the p^{th} -order spherical harmonic transform.

¹In our implementation, we use direct transforms since p is relatively small. As we will see, the main cost of the computation is resolving the weakly singular Stokes single-layer potential.

Computing differential quantities accurately requires *filtering* for anti-aliasing. For instance, the mean curvature, $H(\mathbf{x})$, of a smooth surface is a smooth function which can be computed *at sample locations* from the derivatives of \mathbf{x} (see Section 4.8). However, even if \mathbf{x} is bandlimited and exactly representable with the expansion of order p , this could no longer be true for the mean curvature, and its reconstruction from the values at sample points used for \mathbf{x} suffers from aliasing. To reduce these errors, we upsample \mathbf{x} using spherical harmonics interpolation, perform differentiation, filter out the high-frequency components and then restrict the result to the original grid. The required upsampling order depends on the surface and also the differentiation order and p (for high-enough p , both \mathbf{x} and H are resolved and no filtering is necessary). In our numerical experiments, we found that upsampling to a resolution two times higher than the original grid works well. We demonstrate one such experiment in Figure 20.

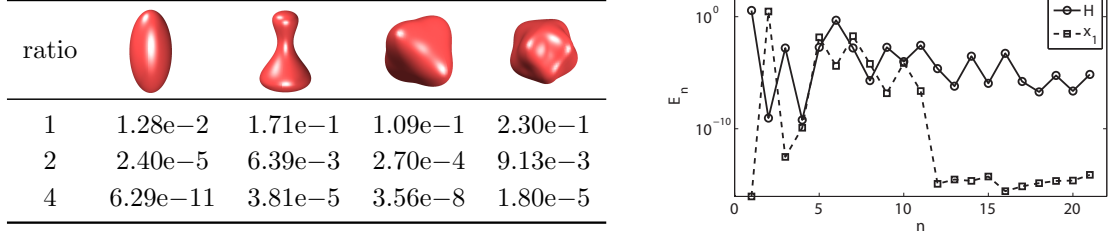


Figure 20: ALIASING ERROR IN CALCULATION OF CURVATURE. In the left table, we report the relative errors $\|H_{q/p} - H^*\|_\infty / \|H^*\|_\infty$ in the computation of the mean curvature. By $H_{q/p}$, we denote the mean curvature obtained by upsampling a p th order representation of the \mathbf{x} to q th order, computing the curvature and then restricting it to the original grid. The reference function H^* is computed analytically. Each row corresponds to a particular ratio q/p and for all shapes we have set $p = 6$. In the right figure, we plot the spectrum $E_n[f] = \sum_{m=-n}^n |\hat{f}_n^m|^2$ of the functions $x_1(u, v)$ and $H(u, v)$ corresponding to the rightmost shape in the table. The bandwidth of the curvature, as expected, is higher than that of the positions justifying the need for upsampling.

4.4.1 Singular Integration

For integrals involving the weakly singular Stokes kernel, we use a superalgebraically convergent integration scheme. There are many techniques that can be used to resolve the singularity accurately. As discussed in the introduction, our main goal in choosing the quadrature scheme is to achieve spectral accuracy while maintaining good computational

efficiency for small values of p ($p \leq 48$). For smooth surfaces that admit global parametrization, there are two main approaches to singular integration: one proposed by Bruno and Kunyansky [27] based on partitions of unity and the other proposed by Graham and Sloan [58] based on rotations and special quadratures. The former scheme has better asymptotic complexity $\mathcal{O}(p^3)$ as opposed to the $\mathcal{O}(p^5)$ of the latter. However, for the values of p that are of interest to us ($p \leq 48$), the scheme by Graham and Sloan performs much better [51]: multiplication by the partition of unity used in the scheme of [27] results in substantial increase in the integrand derivative magnitudes resulting in lower accuracy for a given number of samples. Here, we summarize the Graham-Sloan scheme for convolutions with the Laplace kernel $1/\|\mathbf{r}\|$ and then discuss its extension to the Stokes kernel.

Consider a smooth scalar function $f(\mathbf{x}(u, v)) : \mathbb{S}^2 \rightarrow \mathbb{R}$. We compute the integral

$$\mathcal{A}[f](u_t, v_t) := \int_0^{2\pi} \int_0^\pi \frac{f(u, v)W(u, v)}{\|\mathbf{r}\|} du dv \quad \text{where} \quad \mathbf{r} = \mathbf{x}(u_t, v_t) - \mathbf{x}(u, v) \quad (54)$$

and $W(u, v)$ is the surface area term. First, we rotate the coordinate system $(u, v) \rightarrow (u_R, v_R)$ so that fixed evaluation point (u_t, v_t) becomes $(0, 0)$ (the north pole). The positions are modified to $\mathbf{x}(u_R, v_R)$, the density to $f(u_R, v_R)$ and the surface area term to $W(u_R, v_R)$. In subsequent expressions, we drop the the subscripts from (u_R, v_R) and the coordinate system should be clear from the context. Applying the smooth quadrature rule (Equation (49)), however, may still not be efficient. As an illustration, consider the unit sphere. In this case, $\mathbf{x}_{\mathbb{S}^2}(u, v) = (\sin u \cos v, \sin u \sin v, \cos u)$, $W = \sin u$ and

$$\begin{aligned} \mathcal{A}_{\mathbb{S}^2}[1](0, 0) &= \int_0^{2\pi} \int_0^\pi \frac{\sin u}{\sqrt{2 - 2 \cos u}} du dv = \int_0^{2\pi} \int_0^\pi \cos \frac{u}{2} du dv \\ &= \int_0^{2\pi} dv \int_{-1}^1 dt \frac{1}{\sqrt{1 - t^2}}, \end{aligned} \quad (55)$$

by change of variables $t = \cos u$. The integral has a square-root singularity at $t = \pm 1$, and hence a Gaussian quadrature rule in the t -domain (as applied in (Equation (49))), is not efficient.

The Graham-Sloan scheme makes use of three properties: (1) since γ is diffeomorphic to a sphere, it reduces the problem of computing the harmonic potential on γ to a problem of computing the harmonic potential on \mathbb{S}^2 . Defining $s(u, v) = (W/\|\mathbf{r}\|)_{\mathbb{S}^2} = \cos(u/2)$, we

can write Equation (54) as

$$\mathcal{A}[f](0,0) := \int_0^{2\pi} \int_0^\pi s(u,v) \left(\frac{f(u,v)W(u,v)}{s(u,v)\|\mathbf{r}\|} \right) du dv;$$

(2) Although the function $\frac{fW}{s\|\mathbf{r}\|}$ can be discontinuous at the north pole, it has sufficiently well-behaved spherical harmonic expansions; (3) The harmonic potential on \mathbb{S}^2 can be computed analytically for spherical harmonics [90]:

$$\int_0^{2\pi} \int_0^\pi s(u,v) Y_n^m(u,v) du dv = \frac{4\pi}{2n+1} Y_n^m(0,0). \quad (56)$$

Based on these properties, we can construct modified quadrature weights to compute the harmonic potentials. The result can be summarized as follows:

Theorem 4.4.1.1. (Quadrature rule for singular integrals at poles) [58] *For any smooth boundary \mathbf{x} and for any smooth function f , the quadrature rule for computing the harmonic potential at the north-pole given by*

$$\begin{aligned} \mathcal{A}[f](0,0) &= \sum_{j=0}^p \sum_{k=0}^{2p+1} \frac{w_{jk}^s}{\|\mathbf{x}(0,0) - \mathbf{x}(u_j, v_k)\|} f(u_j, v_k) W(u_j, v_k), \quad \text{with} \\ w_{jk}^s &= w_{jk} \sum_{n=0}^p \frac{4\pi}{\sqrt{2(2n+1)}} \frac{P_n(\cos u_j)}{\cos(u_j/2)} \end{aligned} \quad (57)$$

is superalgebraically convergent² with respect to p .

Proof. Let

$$g(u,v) := \frac{f(u,v)W(u,v)}{s(u,v)\|\mathbf{x}(0,0) - \mathbf{x}(u,v)\|}.$$

In [58] (Lemma 4.6), it is shown that the spherical harmonics expansion of $g(u,v)$ converges

²The proofs outlined in [58] require that the functions need to be oversampled for spectral convergence. However, here we have assumed no oversampling. Empirically we have found that spectral convergence can be observed even without oversampling (eg. Table 13).

sufficiently fast to allow for spectral accuracy of the quadrature rule. Then,

$$\mathcal{A}[f](0,0) = \int_0^{2\pi} \int_0^\pi s(u,v)g(u,v) du dv, \quad (58)$$

$$= \int_0^{2\pi} \int_0^\pi \cos(u/2) \sum_{n=0}^p \sum_{m=-n}^n \hat{g}_n^m Y_n^m(u,v) du dv, \quad (59)$$

$$= \sum_{n=0}^p \sum_{m=-n}^n \hat{g}_n^m \int_0^{2\pi} \int_0^\pi \cos(u/2) Y_n^m(u,v) du dv, \quad (60)$$

$$= \sum_{n=0}^p \sum_{m=-n}^n \hat{g}_n^m \frac{4\pi}{2n+1} Y_n^m(0,0), \quad (61)$$

$$= \sum_{n=0}^p \frac{4\pi}{2n+1} \hat{g}_n^0 \quad (\text{because } Y_n^m(0,0) = 0 \text{ for } |m| > 0), \quad (62)$$

$$= \sum_{n=0}^p \frac{4\pi}{2n+1} \int_0^{2\pi} \int_0^\pi Y_n^0 \frac{W}{\cos(u/2) \|\mathbf{x}(0,0) - \mathbf{x}(u,v)\|} f du dv, \quad (63)$$

$$= \sum_{j=0}^p \sum_{k=0}^{2p+1} \left(w_{jk} \sum_{n=0}^p \frac{4\pi}{2n+1} \sqrt{n + \frac{1}{2}} \frac{P_n(\cos u_j)}{\cos(u_j/2)} \right) \frac{W(u_j, v_k)}{\|\mathbf{x}(0,0) - \mathbf{x}(u_j, v_i)\|} f. \quad (64)$$

The spectral accuracy follows from the convergence rate of approximation Equation (62) for sufficiently large p . ■

Now, let us consider any general point (u, v) other than the poles. Clearly, the quadrature rule Equation (57) is not efficient to compute the harmonic potential at (u, v) . However, we can rotate the coordinate system so that (u, v) becomes the north-pole and then we can apply Equation (57).

Remark 1. *As has been observed by several authors [6, 27, 58], the function $\frac{fW}{\|\mathbf{r}\|}$ in Equation (54) is smooth in the rectangular domain U because the vanishing surface area term W counteracts the singularity due to $1/\|\mathbf{r}\|$. Therefore, applying the Gaussian quadrature along u -direction (as opposed to Theorem 4.4.0.1 which applies Gaussian quadrature in t -direction where $t = \cos u$) and the trapezoidal rule along v -direction yields spectral convergence. However, the errors for a specific p would be slightly higher using this scheme compared to the scheme described in Theorem 4.4.1.1. The reason for this is the clustering of the sampling points near the poles, which results in less accurate approximations for a fixed p .*

A significant part of the overall complexity of Equation (57) is due to the rotation of the coordinate system needed for *every* evaluation point on the vesicle surface. Since we use p^2 evaluation points on the surface, the overall cost of the rotation for *just one* evaluation point is $\mathcal{O}(p^4)$ (mapping p^2 points to p^2 points). (The rotations must be applied for f and the three coordinates of the points.)

However, the $\mathcal{O}(p^4)$ cost per point can be reduced to $\mathcal{O}(p^3)$ per point by rotating in the spherical harmonic representation. Invoking the *addition theorem* [24], one can show that the harmonic coefficients of degree n in the rotated coordinate system depend only on the coefficients of degree n in the original coordinate system.

More precisely, let $\mathbf{F} := \{f(u_j, v_k) | j = 0, \dots, p; k = 0, \dots, 2p + 1\}$. The rotations can be treated sequentially, i.e., we can first rotate the grid in the v -direction in the spherical harmonics space (which corresponds to a simple shift requiring $\mathcal{O}(1)$ operations) and then use a precomputed rotation in the u -direction. Given a target point $(u, 0)$, ($(0, 0)$ in the rotated system), and for each degree n we use a $n \times n$ matrix $R_n(u, 0)$ such that

$$\hat{\mathbf{F}}'_n(u, 0) = R_n(u, 0)\hat{\mathbf{F}}_n.$$

We represent the combined rotations in both u and v more compactly as

$$\hat{\mathbf{F}}'(u, v) = R(u, v)\hat{\mathbf{F}}.$$

Since $\hat{\mathbf{F}}'_n$ can be computed with $\mathcal{O}(p^2)$ work, $\hat{\mathbf{F}}'$ can be computed in $\mathcal{O}(p^3)$ work. The matrices R_n are precomputed either using recurrences or directly by the recently proposed algorithm of Gimbutas and Greengard [54]. Due to symmetry, we only need a $p/2$ set of rotation matrices.

Let P be the forward spherical harmonics transform and P^* the inverse spherical harmonics transform. Algorithm 4.4.2 summarizes the computation of an integral with harmonic potential.

Extension to Stokes kernel. Let us assume the evaluation point to be the north pole. The Stokes kernel Equation (40) has an additional factor

$$K(u, v) = \mathbf{I} + \frac{\mathbf{r} \otimes \mathbf{r}}{\|\mathbf{r}\|^2}, \quad \mathbf{r} = \mathbf{x}(0, 0) - \mathbf{x}(u, v)$$

Algorithm 4.4.2 COMPUTING THE HARMONIC POTENTIAL.

Input: $\mathbf{x} := \{\mathbf{x}(u_j, v_k) \mid j = 0, \dots, p, k = 0, \dots, 2p+1\}$, $\mathbf{F} := \{f(u_j, v_k) \mid j = 0, \dots, p, k = 0, \dots, 2p+1\}$

Output: $\{A[f](u_j, v_k) \mid j = 0, \dots, p, k = 0, \dots, 2p+1\}$

```
 $\hat{\mathbf{X}} = P\mathbf{X}$  and  $\hat{\mathbf{F}} = P\mathbf{F}$  //  $\mathcal{O}(p^2 \log^2 p)$ 
for  $k = 0$  to  $2p+1$  do
  for  $j = 0$  to  $p$  do
     $\hat{\mathbf{x}}' = R(u_j, v_k)\hat{\mathbf{x}}$  //  $R$  is applied to each
                                component of every point in
                                 $\mathbf{x}$ 
                                //  $\mathcal{O}(p^3)$ 
     $\hat{\mathbf{F}}' = R(u_j, v_k)\hat{\mathbf{F}}$ 
    EVALUATE THE HARMONIC POTENTIAL USING EQUATION (57)
     $\mathbf{y}(u_j, v_k) = P^{-1}\hat{\mathbf{x}}'$  and  $\mathbf{Q}(u_j, v_k) = P^{-1}\hat{\mathbf{F}}'$  //  $\mathbf{y}$  and  $\mathbf{Q}$ , used to simplify
                                                                notation
     $A[f](u_j, v_k) = \sum_{n=0}^p \sum_{m=0}^{2p+1} \frac{w_{nm}^s}{\|\mathbf{y}(u_n, v_m) - \mathbf{x}(u_j, v_k)\|} \mathbf{Q}(u_n, v_m) W'(u_n, v_m)$ 
                                                                //  $\mathcal{O}(p^2)$ 
  end for
end for
```

multiplying the Laplace kernel $1/\|\mathbf{r}\|$. At the north-pole, K is continuous but non-smooth. Hence, it is not clear if we can use Equation (57). It turns out that the following two properties of the Stokes kernel suffice to extend the theoretical proofs of [58].

- $K(u, v)$ is *reflectionally symmetric*, that is, $K(u, v) = K(-u, v + \pi)$.
- $K(u, v)$ is an infinitely differentiable function and all its derivatives are 2π -periodic in both u and v .

For more formal descriptions, see Definitions 4.3 and 4.5 of [58]. We use the formula

$$\mathcal{S}[f](0, 0) = \sum_{j=0}^p \sum_{k=0}^{2p+1} w_{jk}^s K(u_j, v_k) f(u_j, v_k) W(u_j, v_k) / \|\mathbf{x}(0, 0) - \mathbf{x}(u_j, v_k)\| \quad (65)$$

analogous to Equation (57) for evaluating the single layer potential with spectral accuracy. At each evaluation point, we rotate the coordinate system so that it is mapped to the north-pole and then use Equation (65). The overall complexity of the algorithm (for a single vesicle) is $\mathcal{O}(p^5)$ work and $\mathcal{O}(p^3)$ storage.

nearly-singular integrals. Vesicle-vesicle hydrodynamic interactions are not singular, so formally the smooth integration rule Equation (49) can be used. However, for close

interactions the accuracy of these rules deteriorates as the integrals become nearly singular. A robust, fourth-order accurate, near-singular integration scheme for smooth surfaces is presented in [145]. However, the constant in the complexity estimate of that scheme is rather high since it involves multiple evaluations and interpolations, and requires finding nearest points on the surface.

For this paper—motivated by our empirical observation that vesicles in the parameter ranges that we have studied tend to be well-separated³ even in relatively concentrated suspensions, we have opted for a simpler scheme that uses upsampling and anti-aliasing. We spectrally upsample⁴ all vesicle surfaces by a factor of two (similar to the medium-distance part of the evaluation in [145]), then we compute the interactions at this higher resolution and then downsample to the original resolution using spectral cutoff. This computation can be done in a per-vesicle adaptive way, but in our implementation we have used uniform upsampling across all vesicles. Let us emphasize that in our simulations, oversampling and filtering are necessary: if they are not invoked one observes numerical blow-up (Figure 26). For more concentrated suspensions, the more expensive scheme of [145] is likely to be required for robustness.

4.5 *Time Scheme*

In this section, we discuss an explicit and a new semi-implicit scheme for the vesicle shape evolution. With very few exceptions (e.g., [39]), explicit schemes have been the method of choice for three-dimensional particle simulations.

Explicit schemes have low computational cost per time step but are known to suffer from stability restrictions on size of the time step. This restriction arises due the numerical stiffness from the high-order spatial derivatives in the bending force. Consequently, the overall computational cost of the simulating in a fixed time horizon tends to be high,

³We consider vesicles well-separated if the minimum inter-vesicle distance is greater than the distance between sample points on the sphere. For example, two spheres in shear flow can come arbitrarily close whereas for droplets the distance tends to stay relatively large [83]. We observe a similar behavior for the case of vesicles; for example, see Figure 27.

⁴Upsampling (or spherical harmonic interpolation) refers to mapping the function to a finer grid by computing the spherical harmonic coefficients (using the given function values at the coarser grid) and then evaluating the spherical harmonic expansion in Equation (46) at the finer grid points. Similarly, downsampling (or spherical harmonic filtering) refers to mapping the function to a coarser grid.

especially with increasing spatial resolution. For vesicles, the inextensibility constraint is typically enforced with an explicit penalty parameter [43, 73] that introduces additional stiffness. In our explicit scheme, the inextensibility constraint is enforced using the tension as a Lagrange multiplier.

Semi-implicit schemes offer a trade-off between the per-time step computational cost and stability requirement on the time-step size. However, they are more difficult to analyze, especially for nonlinear systems. For further discussion on implicit schemes for vesicle simulations, see [135]. Another option is to use a fully implicit scheme. In our previous work on 3D axisymmetric vesicle flows [134], we observed experimentally that a line-search, single-level, Newton scheme in which the Jacobian is approximated using the semi-implicit scheme linearization scheme does not result in computational savings.

An important part of both of our explicit and semi-implicit schemes is the need for reparametrization of the surface to minimize aliasing errors. At every time step, an auxiliary evolution equation

$$\dot{\mathbf{x}} = F(\mathbf{x}) \tag{66}$$

is solved. The choice of the smoothing function F and the time-stepping scheme for Equation (66) is discussed in Section 4.6.

Before we describe the time-stepping schemes, we state three spectral properties of the Stokes, bending, and surface inextensibility operators. We can use these results to (1) characterize the stiffness of the problem, (2) construct preconditioners for the constraint and evolution equations, Section 4.5.4.

4.5.1 Spectral Analysis on the Unit Sphere

To characterize the stiffness of the Equation (42), we consider the case of a spherical vesicle ($\gamma \equiv \mathbb{S}^2$) and we derive the spectrum of \mathcal{S} , $\text{div}_\gamma \mathcal{S}[\mathbf{f}_\sigma]$, and a simplified form of $\mathcal{S}[\mathbf{f}_b]$. To analyze the spectral properties, it is convenient to use *vector spherical harmonics* that form an orthogonal basis for all vectorial functions $\mathbf{f} \in L^2(\mathbb{S}^2)$. They are defined in terms of the

scalar spherical harmonics and their derivatives by

$$\mathbf{V}_n^m(u, v) := \nabla_\gamma Y_n^m(u, v) - (n+1)Y_n^m(u, v)\mathbf{n}(u, v) \quad (67a)$$

$$\mathbf{W}_n^m(u, v) := \nabla_\gamma Y_n^m(u, v) + nY_n^m(u, v)\mathbf{n}(u, v) \quad (67b)$$

$$\mathbf{X}_n^m(u, v) := \mathbf{n}(u, v) \times \nabla_\gamma Y_n^m(u, v) \quad (67c)$$

where \mathbf{n} is the normal to the unit sphere.⁵

For notational convenience, we suppress the (u, v) dependence. Using these definitions, we now derive the spectra of various operators.

Theorem 4.5.1.1. (Stokes operator) *On the unit sphere, the vector spherical harmonic functions \mathbf{V}_n^m and \mathbf{W}_n^m are the eigenfunctions of the Stokes single-layer operator \mathcal{S} and*

$$\mathcal{S}[\mathbf{V}_n^m] = \frac{n}{(2n+1)(2n+3)}\mathbf{V}_n^m, \quad (68a)$$

$$\mathcal{S}[\mathbf{W}_n^m] = \frac{n+1}{(2n-1)(2n+1)}\mathbf{W}_n^m. \quad (68b)$$

Remark 2. *The proofs follow directly from the antenna theorems of [118].*

Proof. The single layer potential satisfies the homogeneous Stokes PDE

$$-\mu\Delta\mathbf{u} + \nabla p = 0, \quad \nabla \cdot \mathbf{u} = 0. \quad (69)$$

inside and outside of \mathbb{S}^2 , combined with velocity continuity and a jump in traction across the interface. We prove the theorem by first finding a solution to Equation (69). We write the fluid velocity \mathbf{u} and the pressure p as

$$\mathbf{u} = f(r)\mathbf{V}_n^m + g(r)\mathbf{W}_n^m \quad (70)$$

$$p = \frac{h(r)}{r}Y_n^m \quad (71)$$

⁵ Various other sets of vector spherical harmonics are used in the literature, tailored for different kinds of problems (e.g., [90, 139]), specifically, for Stokesian flows in [34, 138]. The ones we defined here are first proposed in Hill [60]. They are traces on \mathbb{S}^2 of a corresponding basis set in \mathbb{R}^3 , usually termed as *solid vector spherical harmonics*, defined as [45]

$$\mathcal{V}_n^m(r, u, v) = r\nabla Y_n^m(u, v) - (n+1)Y_n^m(u, v)\mathbf{e}_r,$$

$$\mathcal{W}_n^m(r, u, v) = r\nabla Y_n^m(u, v) + nY_n^m(u, v)\mathbf{e}_r,$$

$$\mathcal{X}_n^m(r, u, v) = \mathbf{e}_r \times (r\nabla Y_n^m(u, v)),$$

where (r, u, v) are the coordinates of a point in the spherical coordinate system with unit vectors $(\mathbf{e}_r, \mathbf{e}_u, \mathbf{e}_v)$.

and substitute in Equation (69). We get the following three ordinary differential equations for f, g , and h :

$$r^2 f_{rr} + 2r f_r - \frac{r}{2n+1} h_r - (n+1)(n+2)f = 0, \quad (72)$$

$$r^2 g_{rr} + 2r g_r + \frac{r}{2n+1} h_r - n(n-1)g + \frac{n}{2n+1} h = 0, \quad (73)$$

$$(n+1)r f_r - n r g_r + (n+1)(n+2)f + n(n-1)g = 0. \quad (74)$$

Solving these equations analytically, we obtain four sets of basic solutions,

	(i)	(ii)	(iii)	(iv)
f	$\frac{1}{r^{n+2}}$	$-\frac{n}{(n+1)r^n}$	0	$\frac{2nr^{n+1}}{(2n+1)(2n+3)(n+1)}$
g	0	$\frac{2}{(2n-1)r^n}$	r^{n-1}	$\frac{r^{n+1}}{2n+1}$
h	0	$-\frac{2n(2n+1)}{(n+1)r^n}$	0	$-2r^{n+1}$

In the exterior region ($r > 1$), only (i) and (ii) are admissible as the other two are unbounded when $r \rightarrow \infty$. In the interior region, only (iii) and (iv) are admissible as the other two are singular at $r = 0$. Therefore, the velocity field \mathbf{u} can be expressed as a linear combination of two solutions in the exterior and two in the interior, leading to four unknown constants. Two of these are determined by enforcing the velocity continuity across \mathbb{S}^2 and the remaining two are determined by the jump in traction. Let us first consider $\mathcal{S}[\mathbf{V}_n^m]$, for which,

$$\llbracket -\mu(\nabla \mathbf{u} + \nabla \mathbf{u}^T) \mathbf{n} + p \mathbf{n} \rrbracket_{\mathbb{S}^2} = \mathbf{V}_n^m. \quad (75)$$

We get two equations by taking inner products with \mathbf{V}_n^m and \mathbf{W}_n^m . Solving these equations, we get a closed form expression for the velocity \mathbf{u} which is same as the single-layer potential and is given by

$$\mathcal{S}[\mathbf{V}_n^m](r, u, v) = \begin{cases} \frac{n}{(2n+1)(2n+3)r^{n+2}} \mathbf{V}_n^m & \text{for } r \geq 1 \\ \frac{nr^{n+1}}{(2n+1)(2n+3)} \mathbf{V}_n^m + \frac{(n+1)(r^{n-1}-r^{n+1})}{2(2n+1)} \mathbf{W}_n^m & \text{for } r \leq 1. \end{cases} \quad (76)$$

The result Equation (68a) is obtained by substituting $r = 1$ in the above equation. Similarly,

equating the traction jump to \mathbf{W}_n^m , we get the following solution,

$$\mathcal{S}[\mathbf{W}_n^m](r, u, v) = \begin{cases} \frac{n}{2(2n+1)} \left(\frac{1}{r^{n+2}} - \frac{1}{r^n} \right) \mathbf{V}_n^m + \frac{n+1}{(2n-1)(2n+1)r^n} \mathbf{W}_n^m & \text{for } r \geq 1 \\ \frac{(n+1)r^{n-1}}{(2n-1)(2n+1)} \mathbf{W}_n^m & \text{for } r \leq 1. \end{cases} \quad (77)$$

and Equation (68b) is obtained by substituting $r = 1$. ■

Theorem 4.5.1.2. (Inextensibility operator) *On the unit sphere, the spherical harmonic functions are eigenfunctions of the inextensibility operator \mathcal{L} defined as*

$$\mathcal{L}\sigma = \operatorname{div}_\gamma \mathcal{S}[\sigma \Delta_\gamma \mathbf{x} + \nabla_\gamma \sigma]. \quad (78)$$

and

$$\mathcal{L}Y_n^m = -\frac{n(n+1)(2n^2+2n-1)}{(2n-1)(2n+1)(2n+3)} Y_n^m. \quad (79)$$

Proof. The proof follows directly from Equation (68). First we write the integrand in terms of the vector harmonics:

$$Y_n^m \Delta_\gamma \mathbf{x} + \nabla_\gamma Y_n^m = \frac{(n+2)\mathbf{V}_n^m + (n-1)\mathbf{W}_n^m}{2n+1}. \quad (80)$$

Then using Theorem 4.5.1.1, we have

$$\mathcal{S}[Y_n^m \Delta_\gamma \mathbf{x} + \nabla_\gamma Y_n^m] = \frac{n(n+2)}{(2n+1)^2(2n+3)} \mathbf{V}_n^m + \frac{(n-1)(n+1)}{(2n-1)(2n+1)^2} \mathbf{W}_n^m \quad (81)$$

The result follows by substituting the identities

$$\nabla_\gamma \cdot \mathbf{V}_n^m = -(n+1)(n+2)Y_n^m, \quad \text{and} \quad \nabla_\gamma \cdot \mathbf{W}_n^m = -n(n+1)Y_n^m \quad (82)$$

in Equation (81). ■

Theorem 4.5.1.3. (Bending operator) *On the unit sphere, the vector spherical harmonic functions \mathbf{V}_n^m and \mathbf{W}_n^m are eigenfunctions of the bending operator defined by $\mathcal{B}[\mathbf{f}] = \mathcal{S}[\Delta_\gamma^2 \mathbf{f}]$ and*

$$\mathcal{B}[\mathbf{V}_n^m] = \frac{n^3(n-1)^2}{(2n+1)(2n+3)} \mathbf{V}_n^m, \quad \mathcal{B}[\mathbf{W}_n^m] = \frac{n(n+1)^2(n+2)^2}{(2n+1)(2n+3)} \mathbf{W}_n^m. \quad (83)$$

Remark 3. Note that we have used a simplified expression ($\mathbf{f}_b = \Delta_\gamma^2 \mathbf{x}$) that retains only the dominant term in the bending force.

Proof. The proof follows from Theorem 4.5.1.1 and the identities $\Delta_\gamma \mathbf{V}_n^m = -n(n-1)\mathbf{V}_n^m$ and $\Delta_\gamma \mathbf{W}_n^m = (n+1)(n+2)\mathbf{W}_n^m$. \blacksquare

From Theorem 4.5.1.1, Theorem 4.5.1.2 and Theorem 4.5.1.3, we conclude that (i) \mathcal{S} is a smoothing operator, (ii) \mathcal{L} and \mathcal{B} are ill-conditioned operators, and (iii) the condition number of \mathcal{L} grows as $\mathcal{O}(n)$ and the condition number of \mathcal{B} grows as $\mathcal{O}(n^3)$. Therefore, we expect severe time-step restrictions for explicit schemes. Next, we discuss such a scheme for the vesicle evolution equations.

4.5.2 Explicit Scheme

We begin our discussion by considering the time-stepping scheme for a single vesicle. Let Δt be a fixed time-step size. Given the positions of the discrete points on the vesicle surface $\{\mathbf{x}^n(u_j, v_k) : 0 \leq j \leq p, 0 \leq k \leq 2p-1\}$ at time $n\Delta t$, the goal is to solve Equation (42) for the new positions at $(n+1)\Delta t$. A first-order, explicit time-stepping scheme for Equation (42) reads

$$\mathcal{L}(\mathbf{x}^n)\sigma^{n+1} = -\operatorname{div}_\gamma(\mathcal{S}[\mathbf{f}_b^n](\mathbf{x}^n)) \quad (84a)$$

$$\frac{1}{\Delta t}(\mathbf{x}^{n+1} - \mathbf{x}^n) = \mathcal{S}[\mathbf{f}_b^n + \mathbf{f}_\sigma^{n+1}](\mathbf{x}^n), \quad (84b)$$

where all the differentiation operators are evaluated at \mathbf{x}^n and $\mathbf{f}_\sigma^{n+1} := \sigma^{n+1} \Delta_\gamma \mathbf{x}^n + \operatorname{grad}_\gamma \sigma^{n+1}$. In the first step, we solve a linear system defined by the discretization of Equation (84a) for the tension σ^{n+1} , given the positions at $n\Delta t$. In the second step, Equation (84b), σ^{n+1} is used for updating the positions. To compute the bending force \mathbf{f}_b^n , we compute the curvatures H^n and K^n of the surface \mathbf{x}^n and substitute in Equation (41). The explicit treatment of the single layer potential means that we evaluate it by the following formula

$$\mathcal{S}[\mathbf{f}_b^n](\mathbf{x}^n) = \int_0^{2\pi} \int_0^\pi G(\mathbf{x}^n(u, v), \mathbf{x}^n(u', v')) \mathbf{f}_b^n(u', v') du' dv', \quad (85)$$

which is computed using the quadrature rule of Theorem 4.4.1.1. For the spatial discretization of Equation (84a) and Equation (84b) we use a collocation method (Fourier-Legendre

quadrature points). Finally, we use a Krylov iterative solver (GMRES [114]) combined with a preconditioner to solve the constraint equation.

This explicit scheme requires the solution of one linear system at each time step and resolves the numerical instability due to tension. However, the bending term is treated explicitly. Based on the spectral analysis of Section 4.5.1, we conjecture that $\Delta t \sim \mathcal{O}(p^{-3})$, which is also corroborated by our numerical tests.

The extension of the scheme given in Equation (84) to multiple vesicle suspensions is done by treating the vesicle-vesicle interactions explicitly. Given the individual vesicle positions $\{\mathbf{x}_k^n\}_{k=1}^K$ at time $n\Delta t$, the update of positions and tensions to $(n+1)\Delta t$ is done in three steps:

STEP 1: EVALUATE BENDING FORCES

for $k = 1$ to K do

 Compute $\mathbf{f}_b^{k,n}, \mathbf{f}_\sigma^{k,n}$

end for

STEP 2: COMPUTE NEW TENSIONS

Compute $\{\sigma_j^{n+1}\}_{j=1}^K$ by solving

$$\mathcal{L}(\mathbf{x}_k^n)\sigma_k^{n+1} = -\operatorname{div}_\gamma \left(\sum_{\substack{j=1 \\ j \neq k}}^K \mathcal{S}[\mathbf{f}_\sigma^{j,n}](\mathbf{x}_k) \right) - \operatorname{div}_\gamma \left(\mathbf{v}_\infty + \sum_{j=1}^K \mathcal{S}[\mathbf{f}_b^{j,n}](\mathbf{x}_k) \right), \quad k = 1, \dots, K$$

STEP 3: EXPLICITLY UPDATE POSITIONS

for $k = 1$ to K do

$$\mathbf{x}_{n+1}^k = \mathbf{x}_n^k + \Delta t \sum_{j=1}^K \mathcal{S}[\mathbf{f}_b^{j,n} + \mathbf{f}_\sigma^{j,n+1}](\mathbf{x}_n^k)$$

SURFACE REPARAMETRIZATION AND TENSION ADVECTION

Solve:

$$\hat{\mathbf{x}}_k^{n+1} = F(\mathbf{x}_k^{n+1}) \quad // \text{ apply tangential smoothing}$$

$$\hat{\sigma}_k^{n+1} + \operatorname{div}_\gamma (\sigma_k^{n+1} F(\mathbf{x}_k^{n+1})) = 0 \quad // \text{ advect the tension}$$

end for

Notice that after the surface reparametrization we need to advect the tension to the new positions so that it is available for the next time step. The advection velocity is defined by the reparametrization of the surface.

Next, we propose a new semi-implicit scheme for which we have observed experimentally that the time step for stability is nearly independent of the spatial discretization size, p .

4.5.3 Semi-implicit Scheme

In a semi-implicit scheme, the linear parts of the stiffest terms are treated implicitly [4]. For example if we have a linear dynamical system of the form $\dot{x} = Ax + Bx$, in which A is the stiff operator, we can discretize it using an implicit scheme for A and an explicit scheme for B . Such schemes have been analyzed in [5]. Our system however, is more complex: it is of the form $\dot{\mathbf{x}} = \mathbf{Q}(\mathbf{x})\mathbf{x}$, where \mathbf{Q} is a nonlinear operator; to our knowledge, there is no analysis of semi-implicit schemes for such dynamical systems. Our semi-implicit approach is, roughly speaking, to discretize it by $\mathbf{x}^{n+1} - \mathbf{x}^n = \Delta t \mathbf{Q}(\mathbf{x}^n)\mathbf{x}^{n+1}$.

Because of its constituent fourth-order spatial derivatives, the bending force is the leading order term that induces stiffness into the evolution equation. Therefore, we look for a linearization of the bending force that results in a stable scheme:

$$\mathcal{L}(\mathbf{x}^n)\sigma^{n+1} = -\operatorname{div}_\gamma \mathcal{S}[\mathbf{f}_b^n](\mathbf{x}^n), \quad (86a)$$

$$\frac{1}{\Delta t}(\mathbf{x}^{n+1} - \mathbf{x}^n) = \mathcal{S}[\mathbf{f}_b^{n+1} + \mathbf{f}_\sigma^{n+1}](\mathbf{x}^n), \quad (86b)$$

where all the differentiation operators are evaluated at \mathbf{x}^n and $\mathbf{f}_\sigma^{n+1} := \sigma^{n+1} \Delta_\gamma \mathbf{x}^n + \operatorname{grad}_\gamma \sigma^{n+1}$. Suppressing the superscripts on explicitly treated terms for notational convenience, the bending force is defined as

$$\mathbf{f}_b^{n+1} = -(\Delta_\gamma H^{n+1} + 2H^{n+1}(H^2 - K))\mathbf{n}, \quad (87)$$

$$\text{and } H^{n+1} = \frac{1}{2W^2} (E\mathbf{x}_{vv}^{n+1} - 2F\mathbf{x}_{uv}^{n+1} + G\mathbf{x}_{uu}^{n+1}) \cdot \mathbf{n}. \quad (88)$$

In the time-stepping scheme, Equation (86), we first solve Equation (86a) for the tension

force starting from the positions at time $n\Delta t$ and then we solve Equation (86b) for the bending force using the linearization defined in Equation (87)⁶.

The advantage of this scheme over the explicit scheme discussed in Section 4.5.2 is that the time-step restriction is overcome with only a modest increase in computational cost. Fully implicit schemes, on the other hand, require solution of nonlinear equations at every time-step making them potentially more expensive. We update $\{\mathbf{x}_n^k, \sigma_n^k\}_{k=1}^K$ as follows:

INTERFACIAL FORCES

for $k = 1$ to K do

 Compute $\mathbf{f}_b^{k,n}$ and $\mathbf{f}_\sigma^{k,n}$

end for

INTERACTION FORCES

for $k = 1$ to K do

$F_k^I = 0$

 for $j \in \{(1, \dots, K) \setminus k\}$ do

$F_k^I = F_k^I + \mathcal{S}[\mathbf{f}_b^{j,n} + \mathbf{f}_\sigma^{j,n}](\mathbf{x}_k)$

 end for

end for

UPDATE POSITIONS AND TENSIONS

for $k = 1$ to K do

 Solve :

$$\mathcal{L}(\mathbf{x}_k^n) \sigma_k^{n+1} = -\operatorname{div}_\gamma(\mathcal{S}[\mathbf{f}_b^{k,n}] + F_k^I) \quad // \text{ Tension}$$

⁶Since the tension force is linear in σ we can easily treat both the tension and bending forces implicitly. In 2D [135] such kind of schemes have superior stability properties at low shear rates. We have implemented this scheme in 3D and noticed (via numerical experiments) no improvements over Equation (86). Compared to Equation (86), these schemes have higher computational cost because of the need to solve coupled linear system of equations and hence are less desirable.

$$\mathbf{x}_k^{n+1} - \Delta t \mathcal{S}[\mathbf{f}_b^{k,n+1}] = \mathbf{x}_k^n + \Delta t \left(\mathcal{S}[\mathbf{f}_\sigma^{k,n}] + F_k^I \right) \quad // \text{ Position}$$

SURFACE REPARAMETRIZATION AND TENSION ADVECTION

Solve:

$$\mathring{\mathbf{x}}^{n+1}_k = F(\mathbf{x}_k^{n+1}) \quad // \text{ Tangential smoothing}$$

$$\mathring{\sigma}_k^{n+1} + \operatorname{div}_\gamma \left(\sigma_k^{n+1} F(\mathbf{x}_k^{n+1}) \right) = 0 \quad // \text{ Advect tension}$$

end for

High-order variants of the explicit Equation (84) and the semi-implicit Equation (86) schemes are readily obtained using the backward difference formula [4].

Both the explicit and the semi-implicit schemes suffer from mesh distortion in longer time simulations. In 2D, the local inextensibility constraint prevents the mesh distortion to a large extent. Unfortunately, this is not the case in 3D. In Section 4.6, we propose a new reparametrization scheme that preserves the quality of the mesh in a dynamic simulation. But first, let us discuss how the spectral analysis of Section 4.5.1 can be used for preconditioning.

4.5.4 Preconditioners

Tension solver. Given the configuration of the vesicle, the inextensibility operator \mathcal{L} , defined in Equation (78), is solved to get the corresponding tension. Depending on the time marching scheme, the right hand side would be some smooth function, denoted here by b . Therefore, the equation for the tension is $\mathcal{L}\sigma = b$. A discrete analogue of this equation is ill-conditioned because by Theorem 4.5.1.2, the condition number of \mathcal{L} on \mathbb{S}^2 grows as $\mathcal{O}(p)$.

Following our method in [135], we propose using $\mathcal{P}^{-1}\mathcal{L}^{-1}(\mathbb{S}^2)\mathcal{P}$ as a low cost preconditioner for the solving the inextensibility constraint on general surfaces, where \mathcal{P} is the projection operator that maps ϕ to $\hat{\phi}$ for any scalar function ϕ . The action of $\mathcal{L}(\mathbb{S}^2)$ on spherical harmonics is given in Equation (79). In Table 11 we list the number of required GMRES iterations with and without using the preconditioner. We note that the gain is significant only for large values of p .

Table 11: AVERAGE NUMBER OF GMRES ITERATIONS FOR THE TENSION SOLVER. *The average is taken over five time steps. GMRES tolerance is $1e-8$ for all cases.*

p	8	16	24	32	48
Preconditioned	3.5	6.8	10	14	20
Non-preconditioned	3.6	8.8	14	20	39

Position solver. Combining equations Equation (86), Equation (87), and Equation (88) one can see that the linear equation which is solved for the new position \mathbf{x}^{n+1} is in the form $(I - \Delta t \mathcal{B})\mathbf{x}^{n+1} = \mathbf{b}$, where \mathbf{b} is some known vector and \mathcal{B} is the bending operator. According to the spectral analysis in Section 4.5.1 and specifically equation set Equation (83), the bending operator’s condition number grows cubically in the spherical harmonics’ order. To subdue its ill-conditioning, we devise a diagonal operator in the spherical harmonic space. The preconditioner is defined as $\mathcal{P}^{-1}\Lambda\mathcal{P}$ where \mathcal{P} the projection operator, and $\Lambda := \text{diag}\{(1 - \Delta t n^3)^{-1}\}$. In Table 12 we report the number of GMRES iterations for the position solver with and without applying the preconditioner. We note that while the performance of the preconditioner is extremely well for low shear rates, it deteriorates for higher shear rates.

Table 12: AVERAGE NUMBER OF GMRES ITERATIONS FOR THE POSITION SOLVER IN THE SEMI-IMPLICIT SCHEME. *The step sizes are the largest stable time steps from Table 14 and the average is taken over ten time steps. The GMRES tolerance is set to 10^{-6} .*

p	Preconditioned			Non-preconditioned		
	$\chi = 0$	15	150	0	15	150
12	2.7	12	17	12	5	3
16	2.7	12	17	21	12	3.3
24	3.6	13	19	36	23	7.4
32	3.6	13	17	55	37	9.2
48	3.6	13	17	98	45	9.4

4.6 Reparametrization

When membrane shear elasticity is present, as for example in Neo-Hookean constitutive laws (e.g., [148]), there is a built-in mechanism that prevents material points on the membrane from clustering together. The local surface inextensibility of vesicles prevents the extreme deformations that can be observed in drops. Nevertheless, the lack of in-plane

shear resistance causes significant distortions of the point distribution, which in turn introduces unresolvable high-frequency components, excessive aliasing errors, and numerical instability (for example see Figure 22). To address these errors and enable long-time accurate simulations using a small number of spherical harmonics coefficients, we reparametrize the surface at each time step through a redistribution of points that seeks to minimize the high-frequency component of the spherical harmonics expansion of the surface parametrization. *Unlike previous work, this anti-aliasing is nonlinear: it is not done by a linear spectral projection but by solving a nonlinear variational problem.*

We define the surface γ as the image of an embedding of the sphere $\mathbf{x}(\mathbf{q}) : \mathbb{S}^2 \rightarrow \mathbb{R}^3$ where \mathbf{q} is a point on \mathbb{S}^2 . We assume that $\mathbf{x}(\mathbf{q}) \in \mathcal{X}$, a space of sufficiently smooth functions on \mathbb{S}^2 . The choice of space depends on the energy; for the energy we consider below any function in L^2 can be used. Many different embeddings correspond to the same surface, and our goal is to choose one that minimizes a quality measure $E(\mathbf{x}) : \mathcal{X} \rightarrow \mathbb{R}$. To characterize all embeddings corresponding to γ , we use its implicit representation, a smooth function $F : \mathbb{R}^3 \rightarrow \mathbb{R}$, such that $F(\gamma) = 0$ and ∇F does not vanish at \mathbf{x} on γ . Then for any parametrization \mathbf{x} , $F(\mathbf{x}(\mathbf{q})) = 0$, for all $q \in \mathbb{S}^2$. The unit normal to γ can be computed as $\mathbf{n} = \nabla F / \|\nabla F\|$. Suppose we want to choose a new parametrization $\mathbf{y}(\mathbf{q})$ of the same surface γ , so that we minimize the energy $E(\mathbf{y})$ subject to the constraint that it is defining the same surface, i.e., $F(\mathbf{y}) = 0$ for all \mathbf{q} .⁷ Using this notation our problem can be formulated as a constrained optimization problem:

$$\begin{aligned} & \min_{\mathbf{y} \in \mathcal{X}} E(\mathbf{y}(\mathbf{q})) \\ & \text{subject to} \quad F(\mathbf{y}(\mathbf{q})) = 0, \quad \forall q \in \mathbb{S}^2, \end{aligned} \tag{89}$$

that is, find a parametrization $\mathbf{y}(\mathbf{q})$ of γ minimizing the quality measure E .

In general, this may be a highly nonlinear problem requiring a computationally expensive method. At the same time, it is generally unnecessary to obtain a precise solution. We present a simple inexpensive method that yields a sufficiently accurate approximation at a

⁷Note that that while it is possible to find a multiple covering of a surface that has lower energy and still satisfies $F(\mathbf{y}) = 0$, in general any such folded surface cannot be obtained by continuous deformation of \mathbf{x} (a non-folded surface) while maintaining $F(\mathbf{y}) = 0$.

low cost.

By introducing a Lagrangian $E(\mathbf{y}) + \int_{\mathbb{S}^2} \mu F(\mathbf{y})$ and taking variations with respect to \mathbf{y} and the Lagrange multiplier $\mu := \mu(\mathbf{q})$, with μ restricted to a space of sufficiently smooth functions on \mathbb{S}^2 , we obtain the (strong form of the) first-order optimality conditions

$$\nabla_{\mathbf{y}} E(\mathbf{y}(\mathbf{q})) + \mu(\mathbf{q}) \nabla_{\mathbf{y}} F(\mathbf{y}(\mathbf{q})) = 0, \quad \text{for all } \mathbf{q} \in \mathbb{S}^2 \quad (90a)$$

and

$$F(\mathbf{y}(\mathbf{q})) = 0, \quad \text{for all } \mathbf{q} \in \mathbb{S}^2. \quad (90b)$$

Due to the special form of these equations, it is possible to eliminate μ by taking the inner product of Equation (90a) with ∇F , so that

$$\nabla E \cdot \nabla F + \mu \|\nabla F\|^2 = 0, \quad \text{or} \quad \mu = -\nabla E \cdot \nabla F / \|\nabla F\|^2,$$

where we drop the subscript \mathbf{y} to simplify the notation. Substituting μ into Equation (90), we obtain

$$(I - \mathbf{n}(\mathbf{y}) \otimes \mathbf{n}(\mathbf{y})) \nabla E(\mathbf{y}) = 0 \quad \text{and} \quad F(\mathbf{y}) = 0. \quad (91)$$

We solve Equation (91) using pseudo-transient continuation [70] by introducing a parameter τ . That is, we solve

$$\mathbf{y}_\tau + (I - \mathbf{n}(\mathbf{y}) \otimes \mathbf{n}(\mathbf{y})) \nabla E(\mathbf{y}) = 0, \quad \mathbf{y}(0) = \mathbf{x}, \quad \text{and} \quad F(\mathbf{y}) = 0, \quad (92)$$

where $\mathbf{y}_\tau := \frac{\partial \mathbf{y}}{\partial \tau}$. This evolution of \mathbf{y} cannot increase the energy, since we are moving in the constraint manifold-projected steepest descent direction.⁸ Our parametrization optimization can be viewed as a discretization of this flow using an explicit scheme:

$$\mathbf{y}_{k+1} = \mathbf{y}_k + \Delta\tau (I - \mathbf{n}_k \otimes \mathbf{n}_k) \nabla E(\mathbf{y}_k). \quad (93)$$

⁸This can be seen as follows. Let $\delta\mathbf{y}$, be the change of \mathbf{y} along the projected steepest-descent direction scaled by some $\zeta > 0$. That is,

$$\delta\mathbf{y} = -\zeta (I - \mathbf{n} \otimes \mathbf{n}) \nabla E = -\zeta \nabla E + \zeta (\nabla E \cdot \mathbf{n}) \mathbf{n}, \quad \zeta > 0.$$

Then the change in the energy

$$\delta E = E(\mathbf{y} + \delta\mathbf{y}) - E(\mathbf{y}) = -\zeta (\|\nabla E\|_2^2 - \|\nabla E \cdot \mathbf{n}\|_2^2).$$

Since $\nabla E = (\nabla E \cdot \mathbf{n}) \mathbf{n} + \nabla E^{\text{tangent}}$ at every point and $\nabla E^{\text{tangent}} \cdot \mathbf{n} = 0$, it follows that $\delta E = -\zeta \|\nabla E^{\text{tangent}}\|_2^2$ is non-positive. Therefore, we are guaranteed to reach a stationary point.

Notice that we do not explicitly impose the constraint $F = 0$. In the continuous case, it is satisfied for all values of τ if the initial value \mathbf{x} satisfies $F(\mathbf{x}) = 0$. Indeed, by substituting Equation (93) into $F(y_{k+1})$ one can show that the former is zero to first-order approximation and $F \rightarrow 0$ for all τ with $\Delta\tau \rightarrow 0$. Of course, one needs to control the size of $\Delta\tau$ to avoid excessive discretization errors.

Choosing the surface parametrization quality metric E . We define a discrete filter by the attenuation factors $a_{n,m}$ for (n, m) -th harmonic. Let $\mathbf{y} = \sum_{n,m} \langle Y_n^m, \mathbf{y} \rangle Y_n^m$. Then we define the energy as

$$E(\mathbf{y}) := \sum_{n,m} a_{n,m}^2 \langle Y_n^m, \mathbf{y} \rangle^2. \quad (94)$$

The variation of E with respect to \mathbf{y} yields

$$\nabla E = \sum_{n,m} a_{n,m}^2 \langle Y_n^m, \mathbf{y} \rangle Y_n^m.$$

Since we want E to penalize the high frequencies in order to minimize aliasing errors, $a_{n,m}$ should be small for low frequencies and should grow for high frequencies. In the special case of perfect low-pass filter ($a_{n,m} = 0$ for $n < n_{cutoff}$ and $a_{n,m} = 1$ for high),

$$\nabla E = \sum_{n > n_{cutoff}, m} \langle Y_n^m, \mathbf{y} \rangle Y_n^m, \quad (95)$$

We penalize the high frequency part of \mathbf{y} .⁹

Integrating the solver with vesicle simulations. In the context of vesicle simulations at the end of each time step, we perform reparametrization to improve the quality of the surface representation, using the scheme given in Equation (93) with the quality measure E given by Equation (95). Since the objective of reparametrization is to maximize the decay of spherical harmonic coefficients, we choose n_0 in Equation (95) to be $p/3$ where p is the order of truncated spherical harmonics series of the surface.

⁹We also experimented with filters associated with the inverse spectrum of the Laplace Beltrami operator on the sphere. In the spectral domain, a_i is proportional to the frequency, leading, as expected, to attenuation with coefficients proportional to the frequency squared. There is no significant difference between the two reparametrization schemes, but the anti-aliasing properties of the scheme that we use produced slightly better results.

In Algorithm 4.6.1, we give the pseudocode for the reparametrization step. In the algorithm, ∇E is given by Equation (95). Notice, that in our implementation we use upsampling (by a factor of two). That is we first upsample \mathbf{x} , then we pseudo-time march to obtain the new points \mathbf{y} , and finally we downsample to the original resolution of \mathbf{x} . This upsampling significantly improves the quality and effectiveness of the reparametrization.

Algorithm 4.6.1 EXPLICIT REPARAMETRIZATION.

Require: x
 choose $\Delta\tau$
 $\mathbf{y}_0 = \mathbf{x}$
 $\mathbf{g}_0 = -(I - \mathbf{n}(\mathbf{y}_0) \otimes \mathbf{n}(\mathbf{y}_0)) \nabla E(\mathbf{y}_0)$
while $\|\mathbf{y} - \mathbf{y}_0\| > \rho_y$ and $\|\mathbf{g}\| > \rho_g \|\mathbf{g}_0\|$ **do**
 $\mathbf{g} = -(I - \mathbf{n}(\mathbf{y}) \otimes \mathbf{n}(\mathbf{y})) \nabla E(\mathbf{y})$ // projected gradient
 $\mathbf{y}^+ \leftarrow \mathbf{y} + \Delta\tau \mathbf{g}$
 $\mathbf{y} \leftarrow \mathbf{y}^+$
end while

Algorithm 4.6.1 corresponds to a sequence of steepest descent steps for the constrained minimization problem, Equation (89). One could use a line search approach for nonlinear programming (e.g., with an ℓ_2 merit function [91]) but this requires access to F and is more complex to implement. Instead, in our approach, the parameter $\Delta\tau$ (which corresponds to the line search step-length) can be chosen using curvature information. That is given \mathbf{x} so that $F(\mathbf{x}) = 0$ and a perturbation in the steepest descent direction $\Delta\tau \mathbf{g}$, then

$$F(\mathbf{x} + \Delta\tau \mathbf{g}) \approx F(\mathbf{x}) + \Delta\tau \nabla F(\mathbf{x}) \cdot \mathbf{g} + \frac{\Delta\tau^2}{2} \mathbf{z} \cdot \nabla \nabla F(\mathbf{x}) \mathbf{z} = \frac{\Delta\tau^2}{2} \mathbf{g} \cdot \nabla \nabla F(\mathbf{x}) \mathbf{g}.$$

The Hessian $\nabla \nabla F(\mathbf{x})$ can be computed using the parametric form of the surface. Then we can choose $\Delta\tau$ so that $F(\mathbf{x} + \Delta\tau \mathbf{g}) < \rho_F \|\mathbf{x}\|$. (Notice that the trace of the Hessian is equal to the mean curvature, so roughly-speaking the pseudo-time step is inversely proportional to mean curvature.)

One additional question is how to choose ρ_y and ρ_g . Parameter ρ_y ensures that if the change in \mathbf{y} becomes small the algorithm terminates. (We choose $\rho_y = 10^{-2} \|\mathbf{x}^{n+1} - \mathbf{x}^n\|$, where \mathbf{x}^{n+1} and \mathbf{x}^n are the new and old positions of the vesicle evolution.) ρ_g ensures that if the gradient becomes too small the algorithm terminates. We demonstrate the effect of reparametrization on the stability of the time marching scheme in the the next section,

along with numerical experiments that examine different aspects of our method.

4.7 Results

In this section, we study the accuracy, stability and complexity of our computational scheme through numerical experiments. In particular, we test (1) the accuracy of the high-order derivatives calculation (curvature) (Section 4.4), (2) the accuracy of the smooth, the weakly-singular, and the nearly-singular quadratures (Section 4.4.1), (3) the need for and effects of reparametrization (Section 4.6); and (4) the time-marching stability properties of the explicit and semi-implicit schemes (Section 4.5). In addition to verifying our method, we also present results on the relaxation shapes of dilute suspension under shear flow, examine a two-vesicle interaction problem, study the sedimentation of a vesicle, and provide an example of a simulation with multiple vesicles.

First, we define the length and time scales for the different flow regimes we consider in our experiments. For a vesicle of area A and volume V suspended in a linear shear flow $\mathbf{v}_\infty = \dot{\gamma}(x_3, 0, 0)$, the non-dimensional parameters are given by

$$\begin{aligned} \text{length scale:} & R_0 = \sqrt{\frac{A}{4\pi}}, \\ \text{time scale:} & \tau = \frac{\mu R_0^3}{\kappa_B}, \\ \text{reduced volume:} & \nu = \frac{6\sqrt{\pi}V}{A^{3/2}}, \\ \text{shear rate:} & \chi = \dot{\gamma}\tau. \end{aligned}$$

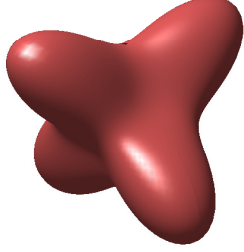
Quiescent flows are characterized by the nondimensional parameter τ , and simple shear flows are characterized by τ and χ .

4.7.1 Derivative Accuracy

Consider a vesicle's surface defined by

$$\mathbf{x}(u, v) = \begin{bmatrix} \rho(u, v) \sin u \cos v \\ \rho(u, v) \sin u \sin v \\ \rho(u, v) \cos u \end{bmatrix}, \quad \rho(u, v) = 1 + e^{-3Re(Y_3^2(u, v))}, \quad u \in [0, \pi], v \in [0, 2\pi]. \quad (96)$$

As we have discussed, we use Equation (50) for differentiation. In Figure 21, we report the relative errors in computing the Gaussian curvature K , and the mean curvature H of this surface.



$M(p)$	H	K
162(8)	$2.44e-1$	$2.21e-1$
578(16)	$3.09e-3$	$1.68e-3$
1250(24)	$1.78e-6$	$1.36e-6$
2178(32)	$4.25e-10$	$2.94e-10$
3362(40)	$2.05e-11$	$8.40e-11$
4802(48)	$3.29e-11$	$1.27e-10$

Figure 21: RELATIVE ERRORS IN COMPUTING THE PRINCIPAL CURVATURES. *The mean and Gaussian curvatures, H and K , are computed numerically on the shape shown. Here M is the number of spatial discretization points and p is the order of corresponding spherical harmonic approximation. The exact values of H and K are computed analytically.*

The errors decay super-algebraically with p . However, for higher values of p , the round-off errors dominate and the relative errors start to grow as $\mathcal{O}(p^2\epsilon)$ where ϵ is the machine precision. This behavior is typical for spectral methods [41].¹⁰ In the axisymmetric case [134], we advocated differentiating non-bandlimited functions as frugally as possible. This approach abates the round-off error growth to some extent. Another promising approach is proposed recently in [23] based on expressing high-order derivatives as Cauchy integrals over circular contours.

4.7.2 Accuracy of the Numerical Integration Schemes

Our singular integral evaluation scheme outlined in Section 4.4 is spectrally accurate. Other popular spectral method is the *floating partition of unity* scheme [27]. Because of the steep gradients in the partition of unity functions, this scheme loses a few digits of accuracy. Our scheme, on the other hand, computes singular integrals as accurately as smooth integrals. See [51] for a comparison of those two schemes.

We report the convergence results for the smooth and singular integral computation

¹⁰In the general case, the error grows as $\mathcal{O}(p^k\epsilon)$ when computing a k^{th} order derivative.

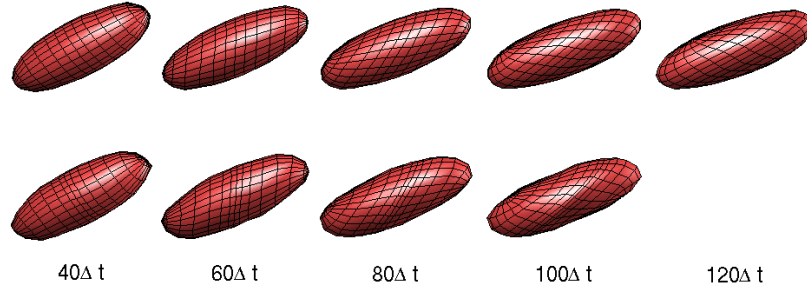
Table 13: RELATIVE ERRORS IN COMPUTING SMOOTH AND SINGLE-LAYER INTEGRALS. *The relative error in (i) smooth integrals: the area, $A = \int_{\gamma} d\gamma = \int_U W du dv$, and the volume, $V = \frac{1}{3} \int_U (\mathbf{x} \cdot \mathbf{n}) W du dv$, (ii) singular integrals: harmonic and Stokes potentials of unit function defined on the shape shown in Figure 21. The reference values are computed using a finer discretization.*

p	Smooth integrals		Singular integrals	
	Area	Volume	Harmonic	Stokes
8	$7.19e-3$	$1.67e-3$	$1.15e-2$	$9.90e-3$
16	$1.42e-6$	$7.53e-8$	$2.50e-4$	$2.96e-4$
24	$6.79e-7$	$2.65e-13$	$2.01e-5$	$2.00e-5$
32	$2.33e-8$	$3.21e-15$	$3.25e-7$	$2.42e-7$

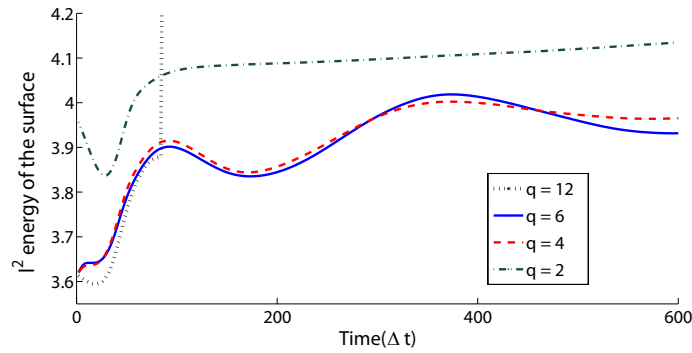
schemes in Table 13. We can observe that the singular integrals are computed with nearly the same order of accuracy as the smooth integrals (compare, for instance, the harmonic potential and area computations). Consequently, using very few spatial discretization points, we are able capture the essential vesicle dynamics. We show one such example in Figure 28.

4.7.3 Reparametrization

To verify the effectiveness of the reparametrization, let us consider a single vesicle in a shear flow. Let the bending modulus of the vesicle $\kappa = 1e-2$, the shear rate $\chi = 15$, and the time horizon $T = 0.6$. In Figure 22(a) we show the configuration of the vesicle at different time steps for two test cases. The top vesicle snapshots correspond to the case when we perform reparametrization, and the bottom set of snapshots is without reparametrization. As we can see in Figure 22(b), the energy of the surface remains bounded when we use reparametrization. In its absence, as soon as the vesicle starts tank-treading, the energy blows up. Note that the need for maintaining the grid quality is not limited to the case when the vesicle undergoes tank-treading motion. As reported in [156], even in the case of axisymmetric gravity-induced motions of deformable drops, the Lagrangian points cluster at the tail of a drop. Moreover, a simple elimination of the tangential component of the velocity, that is, updating the positions using the normal velocity $(\mathbf{u} \cdot \mathbf{n})\mathbf{n}$, suffers from similar mesh degradations [156].



(a) Vesicles configuration



(b) l^2 energy vs. time

Figure 22: EFFECT OF REPARAMETRIZATION. (a) Here we demonstrate the significance of using reparametrization for the representation of the surface. In the simulation depicted on the top row of subfigure (a), we perform surface reparametrization; in the bottom simulation we do not reparametrize. In both cases, we have taken $p = 12$, shear rate $\chi = 15$, $\Delta t = 3e-3$, and we have used the semi-implicit time stepping scheme described in Section 4.5.3. We observe that without reparametrization, at $t \approx 60\Delta t$, the surface becomes highly distorted and the method diverges. Experimentally, the blow-up starts as soon as the vesicle begins to tank-tread. When we do use reparametrization, at the end of simulation, the relative error in area and volume of the vesicle are respectively $3.49e-2$ and $3.86e-2$. (b) Here we plot the l^2 norm of the spherical harmonics coefficients of the surface (for the case of $p = 12$) vs. time for different filtering frequencies, which are denoted by q . The trend of all plots for $q > 6$ is similar to that of $q = 12$ and, for the sake of clarity, those cases are omitted from the plot.

4.7.4 Stability of the Time-marching Scheme

We compare the stability constraints of the explicit and the semi-implicit schemes in simulating the motion of a vesicle suspended in simple shear flow (Figure 23). The initial shape of the vesicle is given by Equation (96) in which we take $\rho(u, v) = 1 + Y_2^0(u, v)$. We computed the stable time-step size Δt using the bisection method: starting from an arbitrary large time step we reduce it by half until the simulation is stable within a predefined time horizon. For this experiment, we have chosen the time horizon long enough so that the vesicle reaches its equilibrium shape in shear flow. The stable time steps are summarized in Table 14 as a function of the shear rate and the spatial resolution p . The explicit

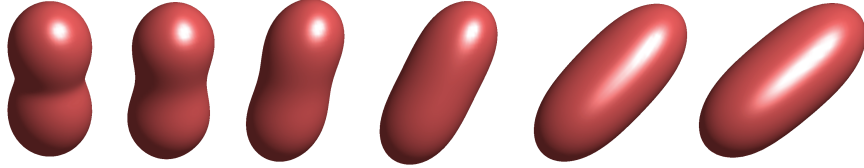


Figure 23: SINGLE VESICLE IN SHEAR FLOW. *Relaxation shape of a vesicle suspended in simple shear flow. Due to the bending term the vesicle quickly relaxes to a shape that has small curvature, but then it gets stretched due to the effect of the shear flow. At the final equilibrium shape the vesicle tank treads. We have used this experiment to test the time-stepping schemes. The corresponding results are reported in Table 14. On a 2.66 GHz Intel Nehalem with 12 GB of memory, the average timing of a time-step for $\chi = 15$ and different spherical harmonics order is as following (p /singular Stokes time/total timing): 8/0.06/0.29, 12/0.44/2.61, 16/1.41/3.47, 24/21.73/22.58. The reported times are in seconds.*

Table 14: STABLE TIME STEP FOR A VESICLE IN SHEAR FLOW. *Here we report stable time-step sizes for the explicit and semi-implicit schemes for a vesicle in simple shear flow. We observe that the implicit-scheme requires a time step whose size is almost independent of the spatial resolution.*

p	Explicit			Semi-implicit		
	$\chi = 0$	15	150	0	15	150
12	$1.04e-2$	$7.81e-3$	$9.76e-4$	$5.00e-1$	$1.02e-2$	$9.77e-4$
16	$4.67e-3$	$1.59e-3$	$3.98e-4$	$3.53e-1$	$1.02e-2$	$4.51e-4$
24	$1.27e-3$	$1.17e-3$	$1.99e-4$	$2.04e-1$	$1.02e-2$	$9.56e-4$
32	$2.05e-4$	$1.70e-4$	$1.25e-4$	$1.49e-1$	$8.15e-3$	$6.79e-4$
48	$1.00e-4$	$1.00e-4$	$5.00e-5$	$1.49e-1$	$8.15e-3$	$6.79e-4$

scheme has two shortcomings: (i) for a fixed p , Δt decreases as χ is increased, (ii) for a

fixed χ and low shear rates, Δt decreases dramatically as p is increased. We can explain these constraints as follows. Embedding the tension force and the constraint equation in a projection operator \mathcal{P} , we can write the non-dimensionalised evolution equation as

$$\dot{\mathbf{x}} = \mathcal{P} \left(\chi \begin{bmatrix} x_3 \\ 0 \\ 0 \end{bmatrix} + \mathcal{S}[-(\Delta_S H + 2H(H^2 - K))\mathbf{n}] \right), \quad (97)$$

where \mathcal{P} acts on a surface velocity field and eliminates the extensible component. We can easily verify that \mathcal{P} is a well-conditioned operator using the spectral analysis of Section 4.3 and it does not contribute to the stiffness. Only the second term within brackets in Equation (97) induces high-order stiffness. Now consider two extreme cases. When $\chi = 0$, the evolution equation is stiff and for this reason we observe a stringent restriction on the time-step. On the other hand, when χ is high, the first term dominates and it attenuates the stiffness arising from the second term. For this reason, the CFL appears to be milder for $\chi = 150$.

The semi-implicit scheme suffers only from the first constraint and its stable time-step size is inversely proportional to χ . But it does not suffer from CFL constraint. Hence, it allows much larger time steps, particularly for low shear rates, compared to the explicit scheme. We have performed additional simulations, reported in Figure 28, Figure 27, and Figure 19 in which we have observed the spatial-resolution independence of the time step. In Figure 24 we summarize the steady-state shapes of vesicles in shear flow for different reduced volumes and shear rates.

4.7.5 Simulations in Presence of Gravity

When there is a difference in the enclosed fluid density (ρ^{in}) and the ambient fluid density (ρ^{out}), the governing equations in the presence of gravity are given by

$$\dot{\mathbf{x}} = \mathcal{S}[\mathbf{f}_b + \mathbf{f}_\sigma + \mathbf{f}_g], \quad (98)$$

$$\text{div}_\gamma(\mathcal{S}[\mathbf{f}_\sigma]) = -\text{div}_\gamma(\mathcal{S}[\mathbf{f}_b + \mathbf{f}_g]), \quad (99)$$

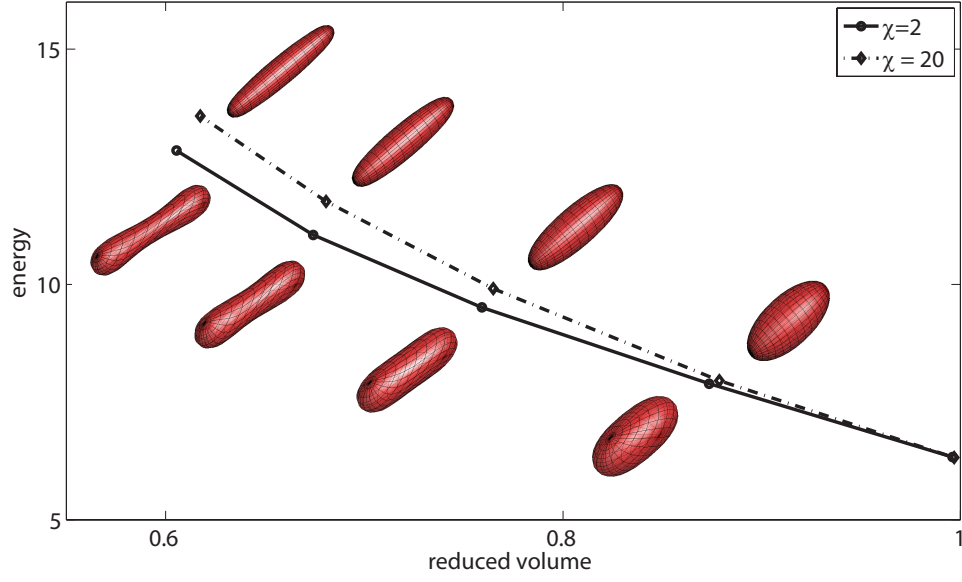


Figure 24: TERMINAL SHAPES OF VESICLES IN A SIMPLE SHEAR FLOW FOR VARIOUS SHEAR RATES. *Contrary to the general belief [73], we find that the terminal shapes depend on the shear rate, especially, for vesicles of low reduced volumes. For low shear rates (equivalently, stiffer membranes), they resemble equilibrium shapes in quiescent flows and consequently possesses lower surface energy compared to the ones at high shear rates.*

where

$$f_g = (\rho^{\text{in}} - \rho^{\text{out}})(\mathbf{g} \cdot \mathbf{x})\mathbf{n} \quad (100)$$

and \mathbf{g} is the gravitational acceleration. The non-dimensional parameter $\hat{g} = \frac{(\rho^{\text{in}} - \rho^{\text{out}})\|\mathbf{g}\| R_0^4}{\kappa_B}$ characterizes gravitational flows. When \hat{g} is less than a threshold value \hat{g}^* , the vesicle reaches its equilibrium shape and translates as a rigid body with constant velocity. When $\hat{g} > \hat{g}^*$, the vesicle reaches a constant average velocity after an initial transient period but the terminal shapes are not unique: they depend in the initial shape. We show three examples in Figure 25. As a further validation of our code, we compare the sedimentation shape that we computed using this code, with the sedimentation shapes computed using our axisymmetric code [134]. We report area, volume, and position errors in Table 15.

4.7.6 Multiple Vesicles

The main additional component required for simulating vesicle-vesicle interactions is the near singular evaluation scheme presented in Section 4.4. Because we have not incorporated

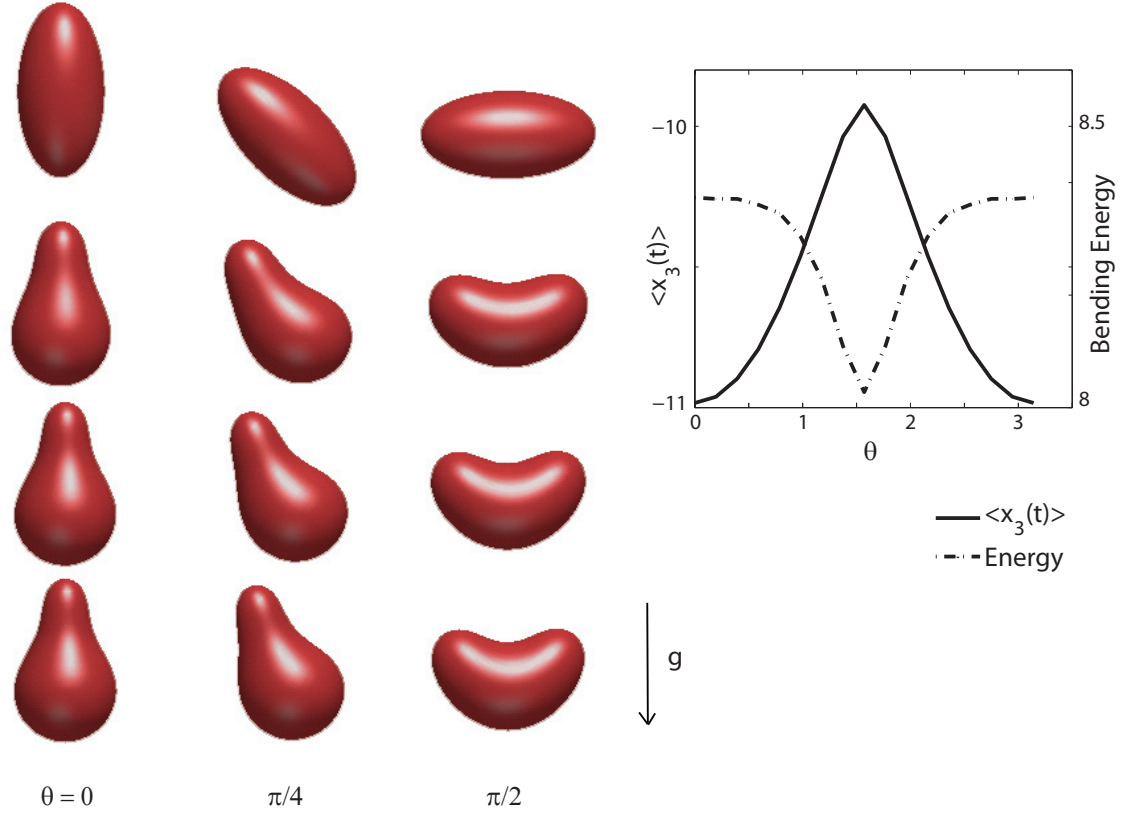


Figure 25: SNAPSHOTs OF THE SEDIMENTATION OF ELLIPSOIDAL VESICLES. The initial shape of the vesicle is given by $(\mathbf{x}(u, v) = (\sin u \cos v, \sin u \sin v, 2 \cos u))$. In all of the three cases, the initial vesicle shape is the same but their orientations (θ) with respect to the axis of gravitational force vary. Unlike the equilibrium shapes in the absence of gravity, the terminal shapes depend both on the reduced volume of the vesicle and on θ . In the right image, we plot the average height of the vesicle at a specific instant $t > 0$, denoted by $\langle x_3(t) \rangle$, and also the bending energy at that instant with respect to initial orientation θ . We can conclude from this plot that the vesicle with $\theta = 0$ (gourd shape) is the most efficient in sedimentation and the one with $\theta = \pi/2$ (ellipsoidal-cap shape) is least efficient. The bending energy, on the other hand, is least for the latter.

Table 15: RELATIVE ERRORS FOR THE SEDIMENTATION SIMULATION. Relative errors in the surface area (A) and volume (V) at the end of the simulation shown in the first column of Figure 25. The first-order semi-implicit time marching presented in Section 4.5 is used for this simulation. We also report the max-norm errors in positions $\mathbf{x}(T)$ using the solution $\mathbf{x}^{\text{axi}}(T)$ of the axisymmetric solver [134] as reference. M is the number of spatial discretization points, p is the order of spherical harmonic approximation, and the time-step size Δt is set to $\mathcal{O}\left(\frac{T}{p}\right)$. The axisymmetric solution is obtained using 64 spatial discretization points and a smaller time-step size.

$M(p)$	$\frac{ A - A_T }{A}$	$\frac{ V - V_T }{V}$	$\frac{\max\ \mathbf{x}(T) - \mathbf{x}^{\text{axi}}(T)\ }{\max\ \mathbf{x}(T)\ }$
162(8)	3.1e-2	1.1e-1	1.5e-1
338(12)	4.8e-4	2.4e-4	3.2e-2
578(16)	6.1e-5	1.0e-4	1.8e-3

collision detection, near singular integral evaluation not only effects the overall accuracy but also the stability of the time-stepping scheme. Although more sophisticated approaches for nearly-singular evaluations exist, we have used a simpler scheme based on upsampling. We estimate the accuracy of using upsampling for nearly-singular integration (Figure 26). Let us emphasize that for a provably accurate scheme for nearly-singular integrals one needs to use the method discussed in [145].

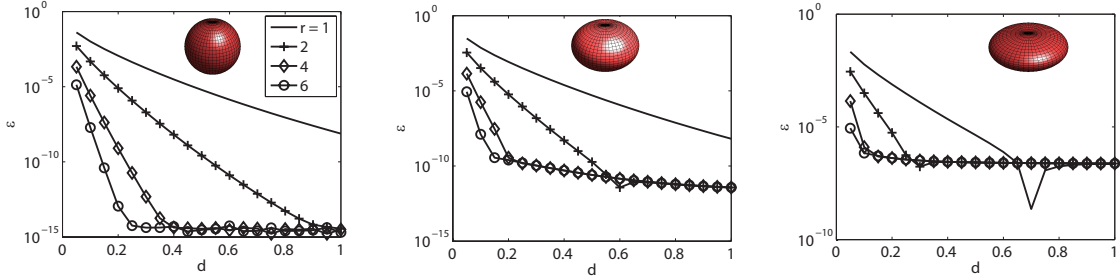


Figure 26: RELATIVE ERROR IN THE EVALUATION OF NEARLY-SINGULAR INTEGRALS. *Relative errors ϵ in computing nearly-singular integrals at an evaluation point whose distance from the vesicle is d using the scheme discussed in Section 4.4 with an oversampling ratio r . For distant points, oversampling improves the accuracy by only a few digits and eventually as we move away from the vesicle (eg., $d > 0.6$ in the rightmost figure) there is no advantage of oversampling. On the other hand, for points closer to the vesicle, oversampling improves the accuracy significantly. Improving accuracy for close interactions is important because it has a direct effect on the overall stability of the numerical simulation. For instance, without any oversampling, the simulation in Figure 27 breaks down when $t > 70\Delta t$ because of numerical instabilities.*

Next, we present two simulations with multiple vesicles. In the first simulation, presented in Figure 27, we consider two vesicles suspended in a simple shear flow. Their initial vertical separation δ gets magnified to Δ . In dilute suspensions, this sort of pairwise interaction is a commonplace and exhibits similar behavior. In Figure 28, we show two simulations that were performed using very few discretization points per vesicle.

4.8 Geometric Formulas

In Table 16, we summarize the formulas for the first fundamental form coefficients E, F and G , the second fundamental form coefficients L, M and N , the unit normal to the surface \mathbf{n} , the Gaussian curvature H , mean curvature K , the Laplace-Beltrami operator Δ_γ , the surface gradient ∇_γ of a scalar ϕ , the surface divergence $\nabla_\gamma \cdot$ of a vector \mathbf{f} , and the surface

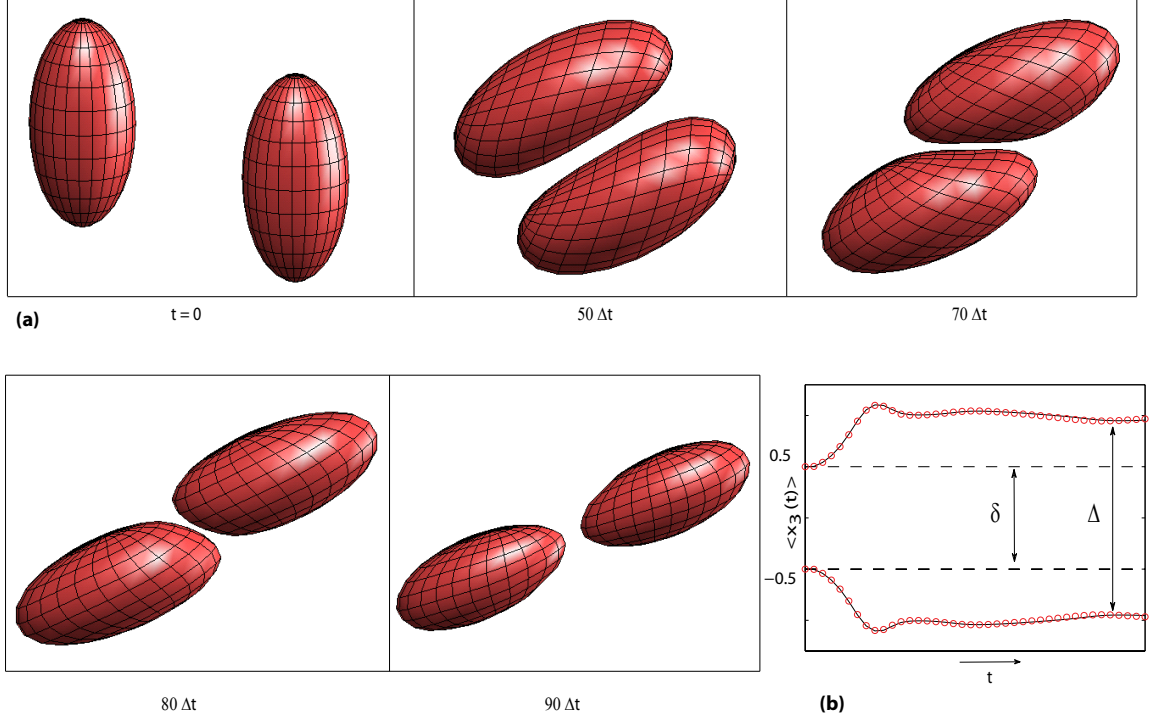


Figure 27: TWO VESICLES IN SHEAR FLOW. (a) Snapshots of two vesicles suspended in a simple shear flow. On a 2.66 GHz Intel Nehalem workstation with 12GB of RAM, the average timing of a semi-implicit step for different spherical harmonics order is as following (p /interaction/singular Stokes time/total timing): 8/0.27/0.39/4.80, 12/0.53/0.86/5.09, 16/0.98/2.73/7.34, and 24/2.80/37.13/42.64. The reported times are in seconds. The interaction between vesicles is evaluated directly. (b) Evolution of the average height ($\langle x_3(t) \rangle$), or equivalently the height of center of mass, of the two vesicles. To show convergence of the numerical scheme, we plot the result obtained using two different discretizations: a finer discretization ($p = 20$) marked by the solid line and a coarser discretization ($p = 12$) marked by “o”. The dashed lines represent the path each vesicle would have followed in the absence of the other. A consequence of this pairwise interaction is that the initial vertical separation δ of the center of masses gets magnified to Δ . This phenomenon is well studied for suspension of drops and elastic capsules [25, 76].

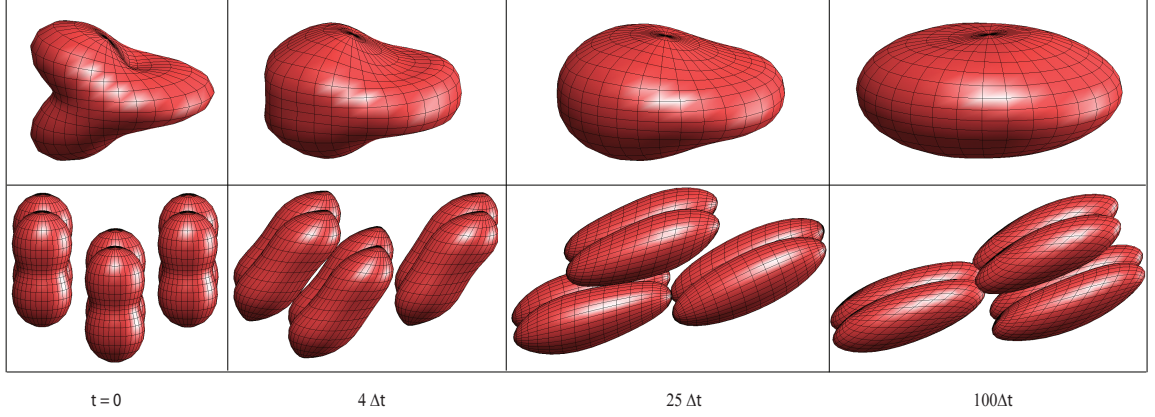


Figure 28: EXAMPLES OF VESICLE FLOW SIMULATIONS. *Top row: snapshots of a freely suspended vesicle relaxing to equilibrium. Bottom row: Multiple vesicle suspension in shear flow. In this example, we used a modest 98 ($p = 6$) spatial discretization points per vesicle. The change in volume and surface area due to numerical errors was less than 1%. Evidently, while finer details are missed (see for example the simulation in Figure 19), our method captures the essential dynamics with very few discretization points. On a 2.66 GHz Intel Nehalem with 12 GB of memory, the average timing of a time-step for different spherical harmonics order is as following (p /interaction/singular Stokes time/total timing): 8/1.07/1.04/11.91, 12/2.30/2.52/14.28, 16/4.58/8.20/22.40, 24/13.98/110.17/130.12. The reported times are in seconds.*

energy \mathcal{E} . The bending and tension forces given in Equation (41) are obtained by taking the L^2 -gradient of the surface energy, that is, the total force $\mathbf{f}_b + \mathbf{f}_\sigma = -\frac{\delta \mathcal{E}}{\delta \mathbf{x}}$. See [140] for a detailed derivation.

Table 16: GEOMETRIC FORMULAS.

Symbol	Definition	Symbol	Definition
E	$\mathbf{x}_u \cdot \mathbf{x}_u$	N	$\mathbf{x}_{vv} \cdot \mathbf{n}$
F	$\mathbf{x}_u \cdot \mathbf{x}_v$	H	$\frac{1}{2} \frac{EN - 2FM + GL}{W^2}$
G	$\mathbf{x}_v \cdot \mathbf{x}_v$	K	$\frac{LN - M^2}{W^2}$
W	$\sqrt{EG - F^2}$	$\Delta_\gamma \phi$	$\frac{1}{W} \left(\frac{E\phi_v - F\phi_u}{W} \right)_v + \frac{1}{W} \left(\frac{G\phi_u - F\phi_v}{W} \right)_u$
\mathbf{n}	$\frac{\mathbf{x}_u \times \mathbf{x}_v}{W}$	$\Delta_\gamma \mathbf{x}$	$(\Delta_\gamma x_1, \Delta_\gamma x_2, \Delta_\gamma x_3)$
L	$\mathbf{x}_{uu} \cdot \mathbf{n}$	$\nabla_\gamma \cdot \mathbf{f}$	$\frac{G\mathbf{f}_u - F\mathbf{f}_v}{W^2} \mathbf{x}_u + \frac{E\mathbf{f}_v - F\mathbf{f}_u}{W^2} \mathbf{x}_v$
M	$\mathbf{x}_{uv} \cdot \mathbf{n}$	$\nabla_\gamma \phi$	$\frac{G\mathbf{x}_u - F\mathbf{x}_v}{W^2} \phi_u + \frac{E\mathbf{x}_v - F\mathbf{x}_u}{W^2} \phi_v$

CHAPTER V

THREE-DIMENSIONAL BOUNDARY INTEGRAL METHOD FOR THE FLOW OF VESICLES WITH VISCOSITY CONTRAST

In this chapter, we consider numerical algorithms for the simulation of dynamics of three-dimensional vesicles suspended in a viscous Stokesian fluid. Here we extend the algorithms of Chapter 4 to flows with viscosity contrast. This generalization poses two new types of difficulty: (i) change in the boundary integral formulation of the solution, in which a double-layer Stokes integral is introduced, and (ii) change of the fluid dynamics inherent to vesicle flows with viscosity contrast. We propose algorithms to deal with these challenges.

Furthermore, we formalize different time stepping algorithms under the same framework, in which the accuracy of the solution dictates the choice of the method (and the time step size). We show that a semi-implicit method does not have time-step stability constraints for flows with single and multiple vesicles with different viscosity contrast and the computational cost-per-simulation-unit-time is comparable to or less than that of an explicit scheme. Some of the components of the algorithm were already present in the previous chapter, but careful treatment of details, such as aliasing, enabled us to achieve high accuracy with very low spectral resolution. The discretization is spectral in space, and first order in time. We conduct numerical experiments to investigate the stability, accuracy, and the computational cost of the algorithm. Overall, our method achieves several orders of magnitude speed-up compared to the standard explicit schemes.

5.1 Introduction

In this chapter, we present an algorithm for the simulation of general three-dimensional vesicle flows with viscosity contrast, where the viscosity of the fluid enclosed inside each vesicle can differ from the viscosity of the suspending fluid. This work is an extension of Chapter 4. The main contributions of this chapter are:

- **Treatment of vesicles with viscosity contrast.** The flow of vesicles with viscosity contrast poses two types of challenges. One is the addition of a double-layer Stokes integral with the velocity as the density to the formulation. We use the algorithm proposed by Graham and Sloan [58] to evaluate the singular double-layer integrals. The second challenge in this type of flow is the change of dynamics. As the viscosity contrast of vesicles increases, they behave more like rigid bodies. This behaviour causes the vesicles to get very close to each other under certain flow conditions. Due to this proximity, locally implicit schemes, in which the interaction of vesicles is treated explicitly, fails to capture the dynamics and poses strict constraint on the required time step.
- **Globally implicit solver.** As we mentioned in the previous item, locally implicit schemes pose a restriction on the required time step for the flow of vesicles with large viscosity contrast. To remove this limitation, we propose an implicit scheme, in which the interaction of vesicles is treated implicitly. We show that the stable time step for this scheme is orders of magnitude larger than the explicit scheme and its computational cost (per unit time) is superior to that of the explicit or locally implicit schemes. For example, Pozrikidis [108] reports that a simulation¹, which captures the periodic dynamics of vesicles such as tumbling, could involve up to 4000 time steps and may require nearly one week of CPU time on a dedicated 2.4 GHz processor operating Linux, to achieve an error tolerance of 0.1% in volume. Using our implicit method, on a similar platform, we are able to capture the tumbling of a vesicle in shear flow with about one hundred time steps in about ten minutes of CPU time.
- **Characterization and treatment of aliasing in differentiation.** In numerical simulations, aliasing is a source of pollution to low frequencies from the high frequency components. In order to facilitate simulations with a small spherical harmonics' truncation order, we use two algorithms to keep aliasing under control. One is the upsampling of functions in differentiation, and the other is reparametrization of the surface.

¹ an explicit boundary element method for vesicle with viscosity contrast

We demonstrate that, to maintain a tolerance in the aliasing error, the upsampling rate for differentiation depend on the shape of vesicles and a constant upsampling rate introduces large errors in certain vesicle configurations. Our reparametrization scheme is the same as the one we proposed in Chapter 4.

Moreover, we use a Galerkin formulation for the boundary integral solution. In Chapter 4, we used pseudo-spectral method, which has the same accuracy as the Galerkin method. Nevertheless, because of the features of spherical harmonics, one needs twice as many variables in the pseudo-spectral method (the grid points) as in Galerkin method (spherical harmonic coefficients).

Four of the time stepping schemes for the vesicle simulation, all can be represented under the same framework, in which the solution to a linear system of equations is approximated numerically. Representing the solution schemes under this unified framework, clarifies the difference of the methods and the superiority of the implicit approach. We report numerical experiment regarding the stability of these methods.

Synopsis of our method. Our method is based on Galerkin formulation corresponding to Lagrangian tracking of spectral collocation points placed on the membrane of the vesicle. We represent the vesicles in the spherical harmonic basis. For weakly-singular integrals, we use the scheme proposed by Graham and Sloan [58], which enables high-accuracy simulations with a small (compared to low-order schemes) number of spectral coefficients per vesicle. For the position update in time, we present an explicit and variants of semi-implicit marching scheme first derived for advection-diffusion equations [5] and then applied to integral-equation based fluid-structure interaction problems in [129].

The time-marching scheme requires the solution to a linear system of equations at each time step, which we perform using a Krylov iterative method (GMRES [114]). The problem of poor conditioning is addressed by a preconditioner based on the analytically obtained spectrum of the operators for the special case of a unit sphere Chapter 4. Vesicle-vesicle interactions can be carried out using the kernel independent fast multipole method [143].

In all, we are able to achieve high accuracy while using a small number of unknowns per

vesicle and taking large time steps with a relatively low computational cost per simulation time unit.

Limitations. We restrict our attention to suspensions of vesicles in unbounded domains. We have ignored inertial terms, so the overall method is restricted to low Reynolds numbers. Only vesicles with spherical topology are considered and topological changes, which are present in many biophysical phenomena involving vesicles, are not allowed. We extend our method to vesicles with viscosity contrast, but our method does not extend to the limiting cases of bubbles ($\lambda = 0$) and solid particles ($\lambda = \infty$). An important limitation of our scheme is the lack of adaptivity (both in space and time). This lack of adaptivity manifests itself in the evaluation of nearly-singular integrals and can cause vesicle-vesicle collisions when the viscosity contrast is high. Indeed, one can easily construct simulations with high viscosity contrast where our algorithms fail to resolve inter-vesicle interactions accurately. This is an open problem and we are currently working on addressing this issue.

5.1.1 Related Work

This work is an extension of [136] and we refer the reader to Chapter 4 for a review of the related work to three-dimensional simulation of vesicles. The work of Graham and Sloan [58] on singular quadratures for the scalar Helmholtz operator, the work of Zinchenko and Davis [155] on surface reparametrization schemes for drops and deformable particles, and the work of Zhao et al. [148] on boundary integral equation based simulations of red blood cells in shear flow² were very influential to this chapter.

In spite of the large body of literature devoted to the investigation of rheological properties of red blood cell and vesicles suspensions, to the best of our knowledge, the work on numerical methods for vesicle flows with viscosity contrast is rather limited. Pozrikidis [104] reviews the work on boundary integral formulations for particulate flows prior to the year 2000. Bagchi and Kalluri [9, 10] consider the flow of capsules with viscosity contrast in shear flow based on a front tracking method. Nonetheless, the capsules lack any resistance

²presenting a spherical harmonic discretization of membranes with bending and shear resistance and explicit time discretization

to bending. Biben and Misbah [18] outline an advected-field method for the vesicle, but they use a radial based representation, which limits the applicability of their method to vesicles with very high reduced area. Pozrikidis [108] present a boundary integral formulation and explicit time stepping scheme, but the method suffers from large grid distortion for low and moderate viscosity contrasts.

To our knowledge the majority of particulate flow solvers in three dimensions use explicit schemes with the exception of the work of Dimitrakopoulos [39], in which a Jacobian-free, finite-difference based Newton method was used. In that paper, Dimitrakopoulos considers the flow of a single droplet (no surface inextensibility constraint) in which the bending force is proportional to the mean curvature. To our knowledge, there is no prior work on implicit schemes for locally inextensible vesicles with viscosity contrast in three dimensions.

5.1.2 Nomenclature

Throughout this chapter, lowercase letters refer to (infinite dimensional) scalar or vector fields. Finite dimensional values are denoted by boldface letters and, when necessary, the spherical harmonics' truncation order is used as a superscript. Convolution kernels are referred to by uppercase calligraphic script and the discretized version of a convolution kernel is referred to by the same letter in uppercase bold. As a general rule, uppercase bold letters denote finite dimensional linear operators. The truncation order of the spherical harmonics expansion is generally denoted by p , and the spherical harmonics coefficients are distinguished by a hat symbol “ $\hat{\cdot}$ ”. We let N denote the number of vesicles. The interface between the i^{th} vesicle and the surrounding fluid is denoted by γ_i ($i = 1, \dots, N$) and the union of all these interfaces is defined as $\gamma := \cup_i \gamma_i$. The outward normal vector to the membrane of vesicles (pointing toward the suspending fluid) is denoted by n . The jump of a variable across an interface is denoted by $\llbracket \cdot \rrbracket$. The surface Laplacian, divergence, and gradient operators are formally denoted by Δ_γ , div_γ , and grad_γ , respectively. In Table 17 we list symbols we use frequently in this chapter.

Outline of the chapter. In Section 5.2, we present the overall integral equation formulation. In Section 5.3, the spatial and temporal discretization schemes are presented

Table 17: INDEX OF FREQUENTLY USED SYMBOLS AND OPERATORS IN CHAPTER 5.

Symbol	Definition
$\ \cdot\ _{l_2^p}^p$	Discrete l^2 on a p -grid (Equation (119))
Δ_γ	Surface Laplacian
α_i	$(1 + \lambda_i)/2$
γ_i	The boundary of the i^{th} vesicle
γ	$\cup_i \gamma_i$
λ_i	The viscosity contrast
μ	Viscosity of the ambient fluid
μ_i	Viscosity of the fluid inside i^{th} vesicle
σ	Tension
χ	Shear rate
ω_i	The domain enclosed by γ_i
ω	$\cup_i \omega_i$
\mathcal{D}_i	The double-layer Stokes operator over i^{th} surface (Equation (103))
$E_n(\cdot)$	Modular l^2 norm (Equation (118))
H	Mean curvature
K	Gaussian curvature
N	Total number of vesicles
\mathbb{S}^2	The unit sphere
\mathcal{S}_i	The single-layer Stokes operator over i^{th} surface (Equation (102))
W	Area element
Y_{nm}	Spherical harmonic of degree n and order m (Equation (106))
div_γ	Surface divergence (Equation (125))
f_σ	Tensile force
f_b	Bending force
grad_γ	Surface gradient (Equation (124))
n	Outward normal vector
p	Order of truncated spherical harmonics expansion
u	Velocity
u^∞	The background velocity field
x	The parametrization of the surface

respectively. In the latter part of that section we present the result of our numerical experiments. The chapter is concluded with a summary of the overall algorithm and computational complexity of different steps in Section 5.4.

5.2 Mathematical Formulation

In this section we formally express the problem statement and give its boundary integral formulation. The detailed derivation of this formulation can be found in [102]. The schematic of a typical domain is given in Figure 29.

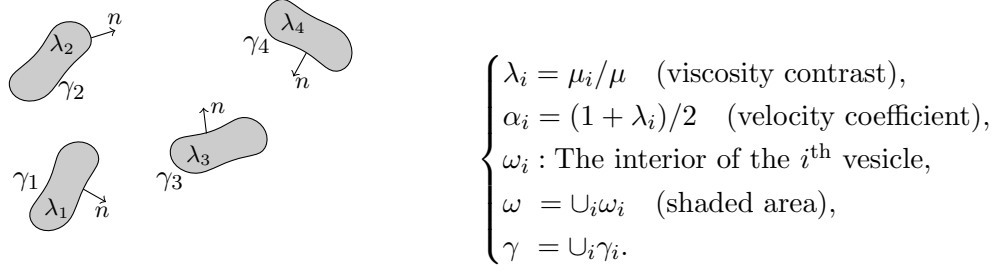


Figure 29: THE SCHEMATIC OF THE DOMAIN.

The differential formulation of the problem is the same as the one given in Chapter 4 which we will not repeat here. The boundary integral formulation is different because of the viscosity contrast of vesicles. Due to the continuity of the velocity field across the interfaces, we can follow the standard approach of potential theory [102, 104] and reformulate Equation Set (38) as an integro-differential equation on the membrane of vesicles. It follows that for all $x \in \gamma_i$ ($i = 1, \dots, N$) we have

$$u(x) = \frac{1}{\alpha_i} \left(u^\infty(x) + \sum_{j=1}^N \mathcal{S}_j[f_b + f_\sigma](x) + \mathcal{D}_j[u](x) \right), \quad (101a)$$

$$\operatorname{div}_{\gamma_i} u(x) = 0, \quad (101b)$$

$$\frac{\partial x}{\partial t} = u(x). \quad (101c)$$

where $\alpha_i = (1 + \lambda_i)/2$, and $\mathcal{S}_j[f_b + f_\sigma](x)$ and $\mathcal{D}_j[u](x)$ denote the single-layer and double-layer convolution integrals over the i^{th} surface with the interfacial force and velocity as respective densities, evaluated at point x . The single-layer Stokes integral over the i^{th} surface, evaluated at point x is defined as

$$\mathcal{S}_i[f](x) := \int_{\gamma_i} S(x, y) f(y) d\gamma(y), \quad S(x, y) = \frac{1}{8\pi\mu} \frac{1}{\|r\|} \left(I + \frac{r \otimes r}{\|r\|^2} \right), \quad (102)$$

where $r := x - y$, I is the identity operator, \otimes denotes the tensor product, and $\|\cdot\|$ is the Euclidean norm. The free-space double-layer integral over the i^{th} surface, evaluated at point x is defined as

$$\mathcal{D}_i[u](x) := \int_{\gamma_i} D_i(x, y) u(y) d\gamma(y), \quad D_i(x, y) = -\frac{3(1 - \lambda_i)}{4\pi} \frac{(r \cdot n)(r \otimes r)}{\|r\|^5}. \quad (103)$$

The subscript i for the double-layer kernel D_i is to emphasize its dependence on the normal

to surface $n(y)$ and the viscosity contrast of the i^{th} vesicle λ_i . Given the initial distribution of vesicles, the Equation Set (101) may be used to solve for their evolution over time.

Galerkin formulation. Using the spherical harmonic functions Y_{nm} (defined in Equation (106)) as the basis set for $L^2(\mathbb{S}^2)$, one can represent the position and tension in this basis set

$$x = \sum_{n=0}^{\infty} \sum_{m=-n}^n \hat{x}_{nm} Y_{nm} \quad \text{and} \quad \sigma = \sum_{n=0}^{\infty} \sum_{m=-n}^n \hat{\sigma}_{nm} Y_{nm}. \quad (104)$$

Letting (\cdot, \cdot) denote the inner product in this space — where the inner product of vector fields is calculated element-wise — the Galerkin method seeks the solution to Equation Set (101) by

$$\alpha_i(u, Y_{nm}) = (u^\infty, Y_{nm}) + \sum_{j=1}^N (\mathcal{S}_j[f_b + f_\sigma], Y_{nm}) + (\mathcal{D}_j[u], Y_{nm}), \quad (105a)$$

$$(\text{div}_{\gamma_i} u, Y_{nm}) = 0, \quad \text{for all } i = 1, \dots, N, \quad (105b)$$

$$\left(\frac{\partial x}{\partial t}, Y_{nm} \right) = (u, Y_{nm}) \quad (105c)$$

for all $n = 1, 2, \dots$ and $|m| \leq n$. The numerical solution of this set of equations involves:

- Accurate computation of interfacial forces f_b and f_σ , which in turn requires the accurate computation of high-order derivatives on the surfaces of vesicles.
- Fast and accurate quadrature over the sphere \mathbb{S}^2 .
- Numerical evaluation of the singular integrals for single- and double-layer Stokes such that the double integral in Equation (105a) is numerically accurate.
- Fast evaluation of the the summation in Equation (105a) to facilitate simulation of a large ensemble of vesicles.

In Section 5.3, we first outline our approach to perform computation over the surface of vesicles using the spherical harmonics. Then, we outline a time stepping method to update the position of vesicles and solve the evolution equation.

5.3 Numerical Algorithms

There are many components required to build the proper numerical machinery to solve the set of integro-differential equation given in the previous section. One needs a fast and accurate method to represent the surface of vesicles in space, perform differentiation on surfaces, and evaluate singular integral over the surfaces. Moreover, in order to evolve the vesicles in time, a method needs to be devised to avoid the stiffness in the evolution equation due to high order derivatives. In this section, we outline a fully discrete Galerkin method in which we represent the surfaces in the spectral basis of spherical harmonics, perform singular integrals using the properties of this representation and evolve the vesicles using a first order semi-implicit time stepping scheme.

5.3.1 Spatial Representation

We assume that the boundary of each vesicle is globally parametrized by a smooth map from \mathbb{S}^2 to \mathbb{R}^3 . This assumption limits the application of this method to the smooth sphere-like (genus zero) surfaces. Nonetheless a large group of biological cells and particles, most importantly red blood cells, has genus zero. The advantage of this assumption is that it allows us to use the spherical harmonics expansions [24, 136].

Spherical harmonics basis. Using the spherical polar coordinates, a point on \mathbb{S}^2 is represented by $(\sin \theta \cos \phi, \sin \theta \sin \phi, \cos \theta)$, where $(\theta, \phi) \in U$ and $U := [0, \pi] \times [0, 2\pi)$. The spherical harmonic function of degree n ($n = 0, 1, 2, \dots$) and order m ($|m| \leq n$) evaluated at point (θ, ϕ) is defined by

$$Y_{nm}(\theta, \phi) = \frac{1}{\sqrt{2\pi}} P_{nm}(\cos \theta) e^{im\phi}, \quad (106)$$

where P_{nm} are the normalized associated Legendre polynomials of degree n and order m . General properties of the spherical harmonics can be found in [24].

Quadrature and discrete inner product over \mathbb{S}^2 . Given an integrable function f over the sphere, its integral can be calculated as

$$\int_{\mathbb{S}^2} f \, ds = \int_0^\pi \int_0^{2\pi} f(\theta, \phi) \sin \theta \, d\phi \, d\theta = \int_{-1}^1 \int_0^{2\pi} f(\cos^{-1} z, \phi) \, d\phi \, dz, \quad (107)$$

where we used the change of variable $z = \cos \theta$.

In this chapter, we use a tensor product quadrature rule to numerically evaluate smooth integrals over the sphere. Given some truncation order p for the spherical harmonics expansion, one needs to numerically calculate the spherical harmonics coefficients for $n = 1, \dots, p$ and $|m| \leq n$. Requiring that these coefficients are evaluated exactly for p -bandlimited functions, our quadrature rule needs to be exact for polynomials of degree $2p$ in the latitude direction and trigonometric polynomials of degree $2p$ in the longitude direction. To this end, we use a $(p+1)$ -point Gauss-Legendre rule in the z direction, and $(2p+2)$ -point uniform trapezoidal rule in the ϕ direction. The Gauss-Legendre nodes $\{z_i\}_{i=0}^p$ are the zeros of the Legendre polynomial of degree $p+1$ and $\{\nu_i\}_{i=0}^p$ are the corresponding weights. We let $\theta_i := \cos^{-1} z_i$ ($i = 0, \dots, p$). The longitudinal nodes are uniform $\{\phi_j = \frac{\pi j}{p+1}\}_{j=0}^{2p+1}$ and the weights are $\{\mu_j = \frac{\pi}{p+1}\}_{j=0}^{2p+1}$. *In the text we refer to this grid as a p -grid.* We denote the discrete samples of the function f on a p -grid by $\mathbf{f}^p := \{f(\theta_i, \phi_j)\}_{i,j}$. It follows that

$$\int_{-1}^1 \int_0^{2\pi} f(\cos^{-1} z, \phi) d\phi dz \approx \sum_{i=0}^p \sum_{j=0}^{2p+1} \nu_i \mu_j f(\theta_i, \phi_j) =: \mathbf{q}^p \cdot \mathbf{f}^p, \quad (108)$$

where the second equality is the definition of the discrete integral operator $\mathbf{q}^p \in \mathbb{R}^{(p+1)(2p+2)}$, and the dot denotes the inner product in $\mathbb{R}^{(p+1)(2p+2)}$ space. Moreover, we let \mathbf{Q}^p denote the matrix with the entries of \mathbf{q}^p on its diagonal, i.e. $\mathbf{Q}^p = \text{diag}(\mathbf{q}^p)$.

We use the usual inner product in $L^2(\mathbb{S}^2)$ denoted by $(f, g) = \int_{\mathbb{S}^2} f \bar{g} ds$, where the overbar denotes the complex conjugate. Based on the integration scheme above, we define a discrete inner product $(\cdot, \cdot)^p$ over the sphere as

$$(f, g)^p := \bar{\mathbf{g}}^p \mathbf{Q}^p \mathbf{f}^p. \quad (109)$$

Spectral representations. Since the spherical harmonics are the eigenfunctions of the Laplace-Beltrami operator on the sphere, they form a basis for $L^2(\mathbb{S}^2)$ and the functions in $L^2(\mathbb{S}^2)$ can be approximated by spherical harmonics expansion, for instance, given a

square-integrable function $f : \mathbb{S}^2 \rightarrow \mathbb{R}$ we can write

$$f(\theta, \phi) = \sum_{n=0}^{\infty} \sum_{m=-n}^n \hat{f}_{nm} Y_{nm}(\theta, \phi), \quad \text{with} \quad (110)$$

$$\hat{f}_{nm} = (f, Y_{nm}) = \int_{\mathbb{S}^2} f \bar{Y}_{nm} ds. \quad (111)$$

Moreover, when the function f is smooth on the sphere, $f \in C^\infty(\mathbb{S}^2)$, the sequence of finite sums converges spectrally to f [95]. In the remainder of this chapter we let p denote the degree of truncated spherical harmonic expansion, that is

$$f(\theta, \phi) \approx \sum_{n=0}^p \sum_{m=-n}^n \hat{f}_{nm} Y_{nm}(\theta, \phi). \quad (112)$$

Given a discrete p -grid and the sample of the function on this grid \mathbf{f}^p , the discrete inner product operator, defined in Equation (109), can be used to evaluate the spherical harmonic coefficients given in Equation (112). We define a discrete spherical harmonic projection $\mathbf{Y}_{\text{Proj}}^p$ such that

$$\hat{\mathbf{f}}^p = \mathbf{Y}_{\text{Proj}}^p \mathbf{f}^p = \bar{\mathbf{Y}}^p \mathbf{Q}^p \mathbf{f}^p, \quad (113)$$

where $\hat{\mathbf{f}}^p = \{\hat{f}_{nm}\}_{n,m}$ is the array of spherical harmonic coefficients, and \mathbf{Y}^p is the matrix with discrete value of spherical harmonics in its columns. Each column of \mathbf{Y}^p is the function Y_{nm} (for some n and m) evaluated on the grid points. Now, we can simply define the inverse spherical harmonics transfer as the multiplication of the coefficient with the discrete basis set

$$\mathbf{f}^p = \mathbf{Y}^p \hat{\mathbf{f}}^p. \quad (114)$$

Due to our requirement on the quadrature rules to be exact for polynomials of degree less than or equal to $2p$ in the latitude and the longitude direction, we have

$$(Y_{nm}, Y_{n'm'})^p = \delta_{nn'} \delta_{mm'}, \quad \text{for all } 0 \leq n, n' \leq p \text{ and } |m| < n, |m'| < n',$$

where δ_{ij} denotes the Kronecker delta. Note that in matrix notation, this translates to

$$\mathbf{Y}_{\text{Proj}}^p \mathbf{Y}^p = \mathbf{Y}^p \mathbf{Y}_{\text{Proj}}^p = \mathbf{I}.$$

However, this discrete orthogonality does not always hold when n or n' is larger than p and there may be cases when $(Y_{nm}, Y_{n'm'})^p \neq 0$ for $n \leq p$ and $n' > p$ [24]. This phenomenon is due to the *aliasing effect*, which we will discuss in a later part of this section.

Representation of the surface. Using the spherical coordinates with domain $U = \{(\theta, \phi) : \theta \in [0, \pi], \phi \in [0, 2\pi)\}$, we define a parametrization of the surface γ by $x : U \rightarrow \mathbb{R}^3$ such that $x(U) = \gamma$. Based on our general assumption that the surfaces are smooth and have genus zero, each Cartesian component of $x = (x_1, x_2, x_3)$ is a smooth function over the sphere and we can write the truncated spherical harmonic expansion of the vector field x as

$$x(\theta, \phi) \approx \sum_{n=0}^p \sum_{m=-n}^n \hat{x}_{nm} Y_{nm}(\theta, \phi), \quad (115)$$

where the spherical harmonic mapping is performed on each component of x , i.e. $\hat{x}_{nm} := ((\hat{x}_1)_{nm}, (\hat{x}_2)_{nm}, (\hat{x}_3)_{nm})$.

Integration over the surface follows from the change of variable given in Equation (107) and the discrete integration formula, Equation (108). Hence, for a scalar field $f(x)$ on the surface γ we have

$$\int_{\gamma} f(x) d\gamma(x) = \int_U f(\theta, \phi) W(\theta, \phi) \sin \theta d\theta d\phi \approx \mathbf{f}^p \cdot \mathbf{Q}^p \mathbf{W}^p, \quad (116)$$

where $W = \sqrt{EG - F^2}$ is the area element of the surface γ (with E , F , and G denoting the coefficients of the first fundamental form of γ) Chapter 4.

Norms. We define different measures of norm in the spherical harmonic domain (or equivalently on the unit sphere) as following:

$$l^2: \quad \|f\|_{l^2} := \left(\sum_{n=0}^{\infty} \sum_{m=-n}^n |\hat{f}_{nm}|^2 \right)^{1/2} \quad (117)$$

$$\text{Modular } l^2: \quad E_n(f) := \left(\sum_{m=-n}^n |\hat{f}_{nm}|^2 \right)^{1/2}, \quad (118)$$

$$\text{Discrete } l^2 \text{ on a } p\text{-grid:} \quad \|\mathbf{f}\|_{l^2}^p := \left(\sum_{n=0}^p \sum_{m=-n}^n |\hat{f}_{nm}^p|^2 \right)^{1/2}. \quad (119)$$

Note that for the l^2 norm on the p -grid, not only the series is truncated but also the coefficients are calculated numerically on that grid using Equation (113).

When we compare the spectral coefficient, it is implied that the series is properly truncated to match the discrete series. For example

$$\|\mathbf{f}^p - f\|_{l^2}^p = \left(\sum_{n=0}^p \sum_{m=-n}^n |\hat{f}_{nm}^p - \hat{f}_{nm}|^2 \right)^{1/2}. \quad (120)$$

The same convention holds when we compare series with different truncation orders p and q ($p \leq q$), where we only compare coefficients with order less than or equal to p :

$$\|\mathbf{f}^p - \mathbf{f}^q\|_{l^2}^p = \left(\sum_{n=0}^p \sum_{m=-n}^n |\hat{f}_{nm}^p - \hat{f}_{nm}^q|^2 \right)^{1/2}. \quad (121)$$

For a vector field $u = (u_1, u_2, u_3)$ and any norm $\|\cdot\|$ given above, we define $\|u\| := (\|u_1\|^2 + \|u_2\|^2 + \|u_3\|^2)^{1/2}$.

Interpolation via spherical harmonics. Given the discrete samples of the function \mathbf{f}^p on a p -grid, one can use Equation (113) and Equation (114) to interpolate f on the nodal points of another q -grid, where q may be larger or smaller than p . The interpolation is performed first by mapping to spherical harmonics domain, followed by padding with zeros or truncation of the array of spherical harmonic coefficients $\hat{\mathbf{f}}^p$, and mapping back to the physical space:

$$\hat{f}_{nm}^q = \begin{cases} \hat{f}_{nm}^p & n \leq p \\ 0 & n > p \end{cases}, \quad \text{for } n = 0, \dots, q \text{ and } |m| \leq n. \quad (122a)$$

Formally, we denote this interpolation process by

$$\mathbf{f}^q = \mathbf{I}_p^q \mathbf{f}^p. \quad (122b)$$

Differentiation on \mathbb{S}^2 and γ . Differentiation of functions defined on the sphere is performed using their spherical harmonic expansion, for example given

$$f(\theta, \phi) = \sum_{n=0}^p \sum_{m=-n}^n \hat{f}_{nm} Y_{nm}(\theta, \phi),$$

one can calculate

$$f_\theta = \sum_{n=0}^p \sum_{m=-n}^n \hat{f}_{nm} (Y_{nm})_\theta, \quad f_\phi = \sum_{n=0}^p \sum_{m=-n}^n \hat{f}_{nm} (Y_{nm})_\phi. \quad (123)$$

For a p -bandlimited function, this approach is exact and for a smooth function the only approximation is in the truncation error of the spherical harmonics expansion.

Given a scalar function $f(x)$ on the surface γ with parametrization $x : U \rightarrow \mathbb{R}^3$, one can calculate the surface gradient of f by

$$\text{grad}_\gamma f = \left(\frac{Gx_\theta - Fx_\phi}{W^2} \right) f_\theta + \left(\frac{Ex_\phi - Fx_\theta}{W^2} \right) f_\phi. \quad (124)$$

Similarly, the surface divergence of a vector field $u(x)$ is given by

$$\text{div}_\gamma u = \left(\frac{Gx_\theta - Fx_\phi}{W^2} \right) \cdot u_\theta + \left(\frac{Ex_\phi - Fx_\theta}{W^2} \right) \cdot u_\phi, \quad (125)$$

and the surface Laplacian is defined as $\Delta_\gamma f = \text{div}_\gamma \text{grad}_\gamma f$.

Aliasing and its remedy. Given a discrete grid (on the real line, the sphere, or any other domain), certain distinct functions, when sampled on this grid, are indistinguishable, i.e. the sampled values agree although the functions are different. The simplest example of this phenomenon is the set of trigonometric polynomials on the real line: on a uniform grid with spacing δ with $t_j = j\delta$, $e^{ik_1 t}$ and $e^{ik_2 t}$ are identical on the sample points when $k_1 - k_2$ is an integer multiple of $\frac{2\pi}{\delta}$ [130]. This phenomenon also occurs for the associated Legendre polynomials on the Gauss-Legendre grid as it is shown in the example given in Figure 30.

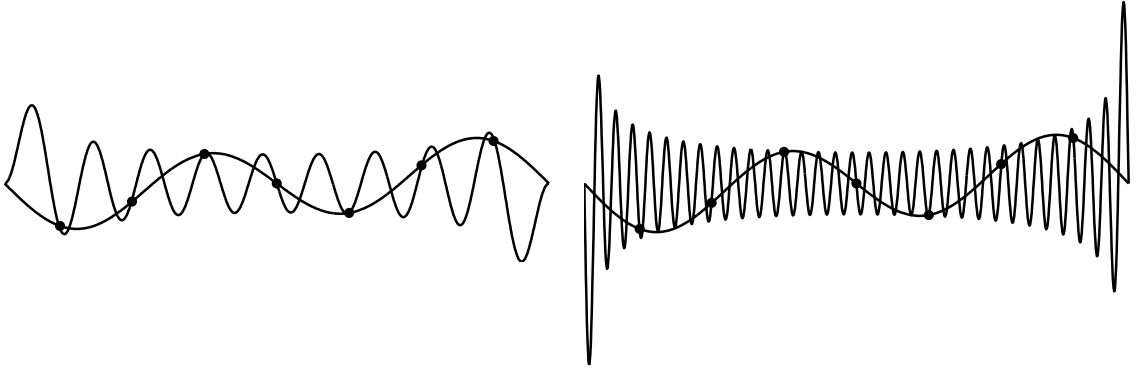


Figure 30: ALIASING FOR ASSOCIATED LEGENDRE POLYNOMIALS. *Aliasing effect for the associated Legendre polynomials on a 7-point Gauss-Legendre grid on $[-1, 1]$. In both of the plots, the less oscillatory functions is $P_{4,1}(z)$. In the left panel, $P_{19,-2}(z)$ is aliased to $P_{4,1}(z)$ and in the right panel $P_{64,1}(z)$ is aliased to $P_{4,1}(z)$. On the sample points, which are denoted by a dot, the value of the high frequency function coincides with that of the low frequency function. (These instances of aliasing are found numerically.)*

In case of spherical harmonics, one high frequency function may alias to many different lower frequency functions. Suppose f is a smooth function over the sphere with the spherical harmonic coefficients \hat{f}_{nm} , i.e. $f = \sum_{n=0}^{\infty} \sum_{m=-n}^n \hat{f}_{nm} Y_{nm}$. Given the samples of f over

a p -grid, one can use the discrete inner product $(\cdot, \cdot)^p$ to calculate the spherical harmonic coefficients $\hat{f}_{n'm'}^p$ ($n' = 0, \dots, p, |m'| < n'$) numerically

$$\begin{aligned}\hat{f}_{n'm'}^p &= (f, Y_{n'm'})^p = \sum_{n=0}^{\infty} \sum_{m=-n}^n \hat{f}_{nm} (Y_{nm}, Y_{n'm'})^p \\ &= \hat{f}_{n'm'} + \sum_{n>p} \sum_{m=-n}^n e_{nm}(n', m') \hat{f}_{nm},\end{aligned}\tag{126}$$

where $|e_{nm}| \leq 1$ by Cauchy-Schwarz inequality. This gives us a crude upper bound on the estimation error as $|\hat{f}_{n'm'}^p - \hat{f}_{n'm'}| \leq \sum_{n>p} \sum_{m=-n}^n |\hat{f}_{nm}|$, that goes to zero as $p \rightarrow \infty$, due to smoothness of f . Therefore, the smoothness of the function f (the rate of decay of \hat{f}_{nm}) and the grid resolution p control the estimation error.

In a spectral setting, aliasing is the source of pollution to lower frequencies and will cause the simulation to diverge. To control the aliasing, when computing the spectral coefficients of a function f on a p -grid, \hat{f}^p , for a given tolerance ϵ , we require that

$$\frac{\|\hat{f}^p - f\|_{l^2}^p}{\|f\|_{l^2}^p} \leq \epsilon.\tag{127}$$

Since in a numerical setting, the exact value of the function is not known, we use the values that are computed on a fine-resolution grid as a surrogate for the exact values. The choice of ϵ depends on the desired accuracy in differentiation. In the context of time stepping methods, we chose this error to be the same order of the time stepping error. For instance, for the Euler method, we choose $\epsilon \leq \delta t^2$, where δt is the time step.

Nonlinear manipulations of functions in the physical space are the main sources of aliasing. Given a band-limited function over the sphere, its derivative may not be band-limited in the spherical harmonic basis and the spectral coefficients may decay much slower. The best example for this, is the mean curvature:

$$H = \frac{EN - 2FM + GL}{2W^2},$$

where E, F , and G are first fundamental form coefficients, L, M , and N are the second fundamental form coefficients, and W is the area element. Given a p -grid samples of a surface x , each term in the expression for mean curvature can be calculated very accurately. Nonetheless, as it is shown in the example given in Figure 31, the mean curvature usually

have more oscillatory components compared to the position x and decay much slower in the spectral domain.

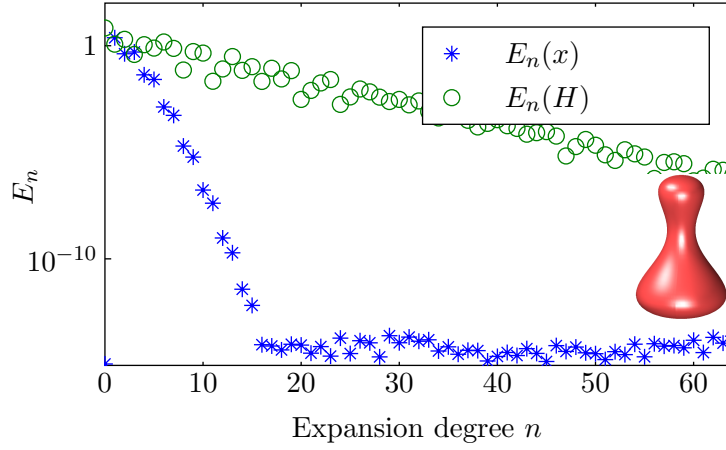
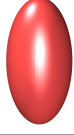

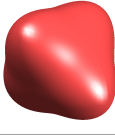




Figure 31: THE SPECTRUM OF THE MEAN CURVATURE. *The modular l^2 norm of the position vector x and that of the mean curvature H for the shape plotted in the inset. The abscissa is the spherical harmonics' degree n and the ordinate is the modular l^2 energy defined in Equation (118). The position vector is band-limited with bandwidth of 16. The bandwidth of H on the other hand, is much wider than that of the position and its spectral coefficients decay much slower. For the data in this figure, the calculation is performed on a 64-grid. On a 16-grid for instance, the shape is represented very accurately. Nonetheless, if the discrete data on this grid are used to evaluate H , through aliasing, all the high frequency components (here $n > 16$) pollute the low frequency coefficients and lead to inaccurate evaluation of \hat{H}_{nm}^{16} .*

Given the discrete samples of the surface x on a p -grid, to avoid the aliasing, one needs to perform the calculation of the mean curvature (or any function that involves differentiation and nonlinear manipulation) on a finer grid q and then restrict the result to the coarse p -grid through Equation (122b). By doing this, one controls the magnitude of the error term in Equation (126) to satisfy Equation (127). Due the highly nonlinear nature of the functions involved in our calculation, the widely used “3/2 upsampling” rate proposed by Orszag [94, 95] for equations with quadratically nonlinear terms does not extend to our case. Moreover, when calculating the curvatures on a p -grid, the required resolution of the fine grid q (or the upsampling rate $v = q/p$) depends on the rate of decay of H , which is not known a priori and depends on the shape. For example, in Table 18 we give the upsampling rate v required to calculate \hat{H}^8 with different values of tolerance, according to Equation (127). In Algorithm 5.3.1 we give the pseudocode to compute the required upsampling rate v to meet a tolerance ϵ . Although, we calculate the upsampling rate for the mean curvature, we use this rate for all differentiations on the surface. During a

Table 18: THE UPSAMPLING RATE FOR DIFFERENTIATION. *The required upsampling rate v in the calculation of the spectral coefficients of the mean curvature H such that $\|\mathbf{H}^{vp} - H\|_{l_2}^p \leq \epsilon \|\mathbf{H}\|_{l_2}^p$. We used a fine resolution $q = 80$ as the surrogate for the exact values.*

ϵ (Tolerance)					
$1e-1$	2	2	1	2	2
$1e-5$	3	4	4	4	4
$1e-9$	4	6	7	7	7

simulation, we update the upsampling rate sporadically as the shape changes over time.

Algorithm 5.3.1 UPSAMPLING RATE CALCULATION.

Require: x, ϵ, p, p_{\max}

$q \leftarrow p$ and $e \leftarrow 2\epsilon$

Calculate \mathbf{H}^p using \mathbf{D}^p as differentiation matrix

while $e > \epsilon$ and $q \leq p_{\max}$ **do**

$q \leftarrow q + p$

 Calculate \mathbf{H}_p^q using \mathbf{D}_p^q as differentiation matrix

 // Defined in Equation (128)

$e \leftarrow \frac{\|\mathbf{H}^p - \mathbf{H}_p^q\|_{l_2}^p}{\|\mathbf{H}_p^q\|_{l_2}^p}$

 // Defined in Equation (127)

$\mathbf{H}^p \leftarrow \mathbf{H}_p^q$

end while

return $v \leftarrow q/p$

Letting \mathbf{D}^q denote a typical differentiation matrix on a q -grid, and using the interpolation operator given in Equation (122b), we define the discrete differentiation with dealiasing as

$$\mathbf{D}_p^q \mathbf{h}^p := (\mathbf{I}_q^p \mathbf{D}^q \mathbf{I}_p^q) \mathbf{h}^p \quad (128)$$

Note, that we adapt this method to keep the representation frequency p as small as possible without losing the accuracy in differentiation. As we will see, the computation complexity of Stokes singular integral is $\mathcal{O}(p^5)$ that gives us enough incentive to try to keep p as small as possible.

In Figure 32 we show the evolution of a single vesicle in Poiseuille flow. This type of flow is challenging because of the kind of deformation it induces on the vesicle. Nonetheless, by treating the aliasing, we can capture the deformation of a vesicle in this flow by using only $p = 8$.

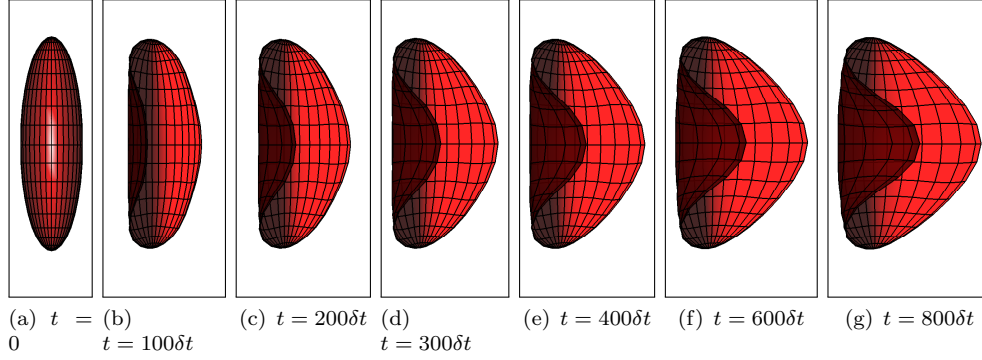


Figure 32: CROSS SECTIONS OF A BICONCAVE VESICLE IN THE POISEUILLE FLOW. *The cross sectional plots showing the evolution of a single vesicle with an initial biconcave shape in a parabolic flow. The time stepping method is implicit, the reduced area is 0.65, and the time step δt is $5e-2$. For this simulation, where $p = 8$, when $t \leq 300\delta t$ the upsampling rate is 4 and when $t > 300\delta t$ the upsampling rate is 5.*

Another important method to control the aliasing and the grid quality is the reparametrization algorithm proposed in Chapter 4.

Singular integrals over the surface γ . In Chapter 4, we extensively explained the quadrature rule to evaluate singular integrals over the sphere, which was adopted from the work of Graham and Sloan [58]. On a surface γ , the double-layer kernel is defined as

$$D(x, y) = -\frac{3}{4\pi} \frac{1}{\|r\|} \left(\frac{r \cdot n}{\|r\|^2} \frac{r \otimes r}{\|r\|^2} \right), \quad \text{with } r = x - y.$$

The first term in the parenthesis satisfies the smoothness requirements outlined in [58]. The second term is similar to the Stokes kernel. In Chapter 4, we argued that since $(r \otimes r)/\|r\|^2$ is reflectionally symmetric³ and smooth over the parameter space of the sphere $U = \{(\theta, \phi) : \theta \in [0, \pi], \phi \in [0, 2\pi)\}$, the method of Graham and Sloan can be extended to the Stokes integral. By the same argument, the double-layer integral can be accurately evaluated using the singular quadratures given in Chapter 4. On the surface γ with a p -grid discretization $\mathbf{x}^p = \{x_{ij}\}$ and density $\mathbf{f}^p = \{f_{ij}\}$, in which $x_{ij} = x(\theta_i, \phi_j)$ and $f_{ij} = f(\theta_i, \phi_j)$, we denote the discrete singular single- and double-layer operators (for all target point \mathbf{x}^p) by $\mathbf{S}^p \mathbf{f}^p$

³A function f is said to be reflectionally symmetric when $f(\theta, \phi) = f(-\theta, \phi + \pi)$ for all $\theta, \phi \in \mathbb{R}$ [58].





and $\mathbf{D}^p \mathbf{f}^p$, such that

$$\mathcal{S}[f](x_{ij}) = \int_{\gamma} S(x_{ij}, y) f(y) d\gamma \approx (\mathbf{S}^p \mathbf{f}^p)_{ij}, \quad (129a)$$

$$\mathcal{D}[f](x_{ij}) = \int_{\gamma} D(x_{ij}, y) f(y) d\gamma \approx (\mathbf{D}^p \mathbf{f}^p)_{ij}, \quad (129b)$$

for all $0 \leq i \leq p$ and $0 \leq j \leq 2p + 2$. The general algorithm to evaluate the double-layer integral is the same as the one for the Stokes integral and we refer to Chapter 4 for the details. Here, we report convergence results for the numerical integration of the double-layer integral over different shapes in Table 19.

Table 19: CONVERGENCE OF DISCRETE DOUBLE-LAYER INTEGRAL. *The relative error $\|\mathbf{D}^p \mathbf{f}^p - \mathbf{D}^q \mathbf{f}^q\|_{l_2}^p / \|\mathbf{D}^q \mathbf{f}^q\|_{l_2}^p$ in the computation of the double-layer integral over the depicted surfaces for different spherical harmonic order p . The discrete double-layer operator is defined in Equation (129b) and the discrete norm is defined in Equation (119). The error is calculated with respect to a fine discretization with spherical harmonics order of $q = 64$. On each surface, the density is chosen to be the same as the position vector ($f = x$).*

p				
8	6.29e-3	2.53e-1	8.11e-2	9.43e-2
12	6.39e-4	6.06e-2	2.61e-2	2.59e-2
16	6.56e-5	1.49e-2	9.91e-3	5.38e-3
24	6.96e-7	3.02e-3	9.28e-4	4.30e-4
32	7.19e-9	3.38e-4	9.83e-5	4.64e-5

In a Galerkin setup, numerical evaluation of a double integral is involved in the Equation Set (105), namely the singular integrals followed by the projection to the basis

$$(\mathcal{S}[f], Y_{nm}) \approx (\mathbf{S}^q \mathbf{f}^q, Y_{nm})^p \quad \text{and} \quad (\mathcal{D}[u], Y_{nm}) \approx (\mathbf{D}^q \mathbf{u}^q, Y_{nm})^p. \quad (130)$$

The result presented in [58, mainly Theorem 4.1] requires that $q > p(1 + a)$ for some $a > 0$. However, it does not give any formula to estimate a . In our numerical experiments and convergence analysis, we noticed that when we solve for position and tension with $q = p$, the coefficients with highest order, i.e. $\hat{\mathbf{x}}_{pm}$ and $\hat{\boldsymbol{\sigma}}_{pm}$ for $|m| \leq p$, are calculated with large error. Nonetheless, this was fixed by choosing $q = p + 1$. Combining all these computational steps (i.e. upsampling to q , performing singular integral, and mapping back to p) together,

we can write

$$\widehat{\mathbf{S}}^p \widehat{\mathbf{f}}^p := (\mathbf{I}_q^p \mathbf{Y}_{\text{Proj}}^q \mathbf{S}^q \mathbf{Y}^q \mathbf{I}_p^q) \widehat{\mathbf{f}}^p, \quad (131)$$

for the single-layer Stokes operator in spherical harmonic domain.⁴ The same can also be written for the double-layer Stokes, which we denote by $\widehat{\mathbf{D}}^p$.

Nearly-singular integrals. Nearly-singular integrals are a class of integral that are artifacts of numerical discretization. In the context of our work, let h denote the discretization spacing for a p -grid samples of a given surface γ :

$$h := \max_{x \in \mathbf{x}^p} \min_{\substack{y \in \mathbf{x}^p \\ y \neq x}} |x - y|.$$

Now, let's consider the single-layer integral for a target point $x \in \mathbb{R}^3$ with density f on surface γ , $\mathcal{S}[f](x) = \frac{1}{8\pi\mu} \int_{\gamma} S(x, y) f(y) d\gamma(y)$. When $x \in \gamma$ the integral is singular and can be numerically integrated using the algorithm mentioned above; when $x \notin \gamma$ then $d = \text{dist}(x, \gamma) > 0$ and the integral is not singular anymore. Nevertheless, when $d \leq \sqrt{h}$ then the Stokes kernel becomes oscillatory and the derivative of $S(x, y)$ with respect to y cannot be bounded uniformly [145]. In this work, nearly-singular integrals arise when computing the interaction between vesicles. When computing the interaction, we try to avoid the occurrence of nearly-singular integrals by uniformly upsampling the grid points to a finer frequency $q \approx p^{3/2}$. The choice of this upsampling frequency is based on the argument given on [145] in order to permit $d \approx h$. There are situations where the vesicles get too close and our method breaks down (an example of this is when the viscosity contrast of vesicles is large, say $\mathcal{O}(10)$). We are currently working to incorporate algorithms to treat the nearly singular integrals.

5.3.2 Galerkin Formulation

Using the machinery developed in the first part of this section, we use a Galerkin formulation to solve for the evolution of vesicles—solve Equation Set (105)—in the spectral domain.

⁴In practice, the computationally efficient way to do this is to keep the frequencies p and q the same, but filter the coefficients with highest degree.

In Chapter 4, we used pseudo-spectral method, which has the same spatial accuracy as the Galerkin method. Nevertheless, because of the features of spherical harmonics, one needs twice as many variables in the pseudo-spectral method (the grid points) as in Galerkin method (spherical harmonic coefficients). For this reason, we opted for the Galerkin method.

Linear interfacial forces. Following our approach in Chapter 4, given the configuration of the vesicle γ , a new configuration defined by x^+ , and a new tension σ^+ we define the linear versions (with respect to x^+ or σ^+) of mean curvature, bending force, and tensile force:

$$H(x^+; \gamma) = \frac{1}{2W^2} \left(Ex_{\phi\phi}^+ - 2Fx_{\theta\phi}^+ + Gx_{\theta\theta}^+ \right) \cdot n, \quad (132a)$$

$$f_b(x^+; \gamma) = -\kappa_b \left[\Delta_\gamma H^+ + 2H^+(H^2 - K) \right] n, \quad (132b)$$

$$f_\sigma(\sigma^+; \gamma) = \sigma^+ \Delta_\gamma x + \text{grad}_\gamma \sigma^+, \quad (132c)$$

where the $H^+ = H(x^+; \gamma)$ and the terms with no superscript are evaluated over γ . The discrete spectral versions of these operator can be written as $\hat{\mathbf{f}}_b = \mathbf{Y}_{\text{Proj}} \mathbf{f}_b$ and $\hat{\mathbf{f}}_\sigma = \mathbf{Y}_{\text{Proj}} \mathbf{f}_\sigma$. Combining these expressions with the spectral stokes operator Equation (131), we define two new linear operators $\hat{\mathbf{B}}$ and $\hat{\mathbf{T}}$ such that

$$\hat{\mathbf{B}} \hat{\mathbf{x}}^+ = \hat{\mathbf{S}} \hat{\mathbf{f}}_b, \quad (133)$$

$$\hat{\mathbf{T}} \hat{\boldsymbol{\sigma}}^+ = \hat{\mathbf{S}} \hat{\mathbf{f}}_\sigma. \quad (134)$$

Moreover, we define the discrete, spectral, divergence operator as

$$\hat{\mathbf{P}} \hat{\mathbf{u}} = \mathbf{Y}_{\text{Proj}} \text{div}_\gamma (\mathbf{Y} \hat{\mathbf{u}}). \quad (135)$$

When we have multiple vesicles, our notation for the bending and tension operators needs to be more specific. We use the double subscript $\hat{\mathbf{B}}_{ij}, \hat{\mathbf{T}}_{ij}, \hat{\mathbf{D}}_{ij}$ ($i, j = 1, \dots, N$) that implies the integral is performed on the j^{th} surface and the target points are the grid points of the i^{th} surface. For the surface divergence, we append the single subscript $\hat{\mathbf{P}}_i$ to denote that the divergence is taken on the i^{th} surface. For the bending, tensile, and double-layer operators, when $i = j$ the integral is evaluated using the singular quadratures and when $i \neq j$ the integral is a smooth integral.

System of ODEs for a suspension of N vesicles. Substituting these discrete operators in the Equation Set (105), we get a system of ODEs for the discrete tension and position of the i^{th} vesicle ($i = 1, \dots, N$):

$$\alpha_i \hat{\mathbf{u}}_i = \hat{\mathbf{u}}^\infty(\mathbf{x}_i) + \sum_{j=1}^N \hat{\mathbf{B}}_{ij} \hat{\mathbf{x}}_j + \hat{\mathbf{T}}_{ij} \hat{\boldsymbol{\sigma}}_j + \hat{\mathbf{D}}_{ij} \hat{\mathbf{u}}_j, \quad (136a)$$

$$\hat{\mathbf{P}}_i \hat{\mathbf{u}}_i = 0, \quad (136b)$$

$$\frac{d\hat{\mathbf{x}}_i}{dt} = \hat{\mathbf{u}}_i. \quad (136c)$$

5.3.3 Time Discretization

We use a first order time discretization to approximate the time derivative of position, where we write $d\mathbf{x}/dt \approx (\mathbf{x}^+ - \mathbf{x})/\delta t$ or equivalently

$$\frac{d\hat{\mathbf{x}}_i}{dt} \approx \frac{\hat{\mathbf{x}}_i^+ - \hat{\mathbf{x}}_i}{\delta t}. \quad (137)$$

Substituting this in Equation (136c), using the result in Equation (136a) and substituting in Equation (136b), we get a system of equations for the new position and tension

$$\alpha_i \hat{\mathbf{x}}_i^+ - \sum_{j=1}^N (\delta t \hat{\mathbf{B}}_{ij} + \hat{\mathbf{D}}_{ij}) \hat{\mathbf{x}}_j^+ - \delta t \sum_{j=1}^N \hat{\mathbf{T}}_{ij} \hat{\boldsymbol{\sigma}}_j^+ = \mathbf{r}_i, \quad (138a)$$

$$- \sum_{j=1}^N (\delta t \hat{\mathbf{P}}_i \hat{\mathbf{B}}_{ij} + \hat{\mathbf{P}}_i \hat{\mathbf{D}}_{ij}) \hat{\mathbf{x}}_j^+ - \delta t \sum_{j=1}^N \hat{\mathbf{L}}_{ij} \hat{\boldsymbol{\sigma}}_j^+ = \mathbf{s}_i, \quad (138b)$$

where $\mathbf{r}_i := \alpha_i \hat{\mathbf{x}}_i + \delta t \hat{\mathbf{u}}^\infty - \sum_{j=1}^N \hat{\mathbf{D}}_{ij} \hat{\mathbf{x}}_j$ and $\mathbf{s}_i := \delta t \hat{\mathbf{P}}_i \hat{\mathbf{u}}^\infty - \sum_{j=1}^N \hat{\mathbf{P}}_i \hat{\mathbf{D}}_{ij} \hat{\mathbf{x}}_j$ for all $i = 1, \dots, N$, and $\hat{\mathbf{L}}_{ij} := \hat{\mathbf{P}}_i \hat{\mathbf{T}}_{ij}$. Instead of Equation (136b) we use Equation (138b) — which is derived by applying $\hat{\mathbf{P}}_i$ to Equation (136a) — because we know an analytical preconditioner for $\hat{\mathbf{L}}_{ii}$, derived in Chapter 4. This system of equations can be written in the matrix format as

$$\mathbf{A}_{ii} \begin{bmatrix} \hat{\mathbf{x}}_i^+ \\ \hat{\boldsymbol{\sigma}}_i^+ \end{bmatrix} + \sum_{\substack{j=1 \\ j \neq i}}^N \mathbf{A}_{ij} \begin{bmatrix} \hat{\mathbf{x}}_j^+ \\ \hat{\boldsymbol{\sigma}}_j^+ \end{bmatrix} = \begin{bmatrix} \mathbf{r}_i \\ \mathbf{s}_i \end{bmatrix}, \quad i = 1, \dots, N \quad (139)$$

where

$$\mathbf{A}_{ii} := \begin{bmatrix} \alpha_i \mathbf{I} - \delta t \hat{\mathbf{B}}_{ii} - \hat{\mathbf{D}}_{ii} & -\delta t \hat{\mathbf{T}}_{ii} \\ -\delta t \hat{\mathbf{P}}_i \hat{\mathbf{B}}_{ii} - \hat{\mathbf{P}}_i \hat{\mathbf{D}}_{ii} & -\delta t \hat{\mathbf{L}}_{ii} \end{bmatrix} \quad \text{and} \quad \mathbf{A}_{ij} := \begin{bmatrix} -\delta t \hat{\mathbf{B}}_{ij} - \hat{\mathbf{D}}_{ij} & -\hat{\mathbf{T}}_{ij} \\ -\delta t \hat{\mathbf{P}}_i \hat{\mathbf{B}}_{ij} - \hat{\mathbf{P}}_i \hat{\mathbf{D}}_{ij} & -\delta t \hat{\mathbf{L}}_{ij} \end{bmatrix}.$$

(140)

To further simplify the notation, let's write $\mathbf{A} = \mathbf{A}_d + \mathbf{A}_o$ where \mathbf{A}_d is the block diagonal matrix with \mathbf{A}_{ii} on its diagonal, and $\mathbf{A}_o = \mathbf{A} - \mathbf{A}_d$, and write $\mathbf{y}_i = [\hat{\mathbf{x}}_i, \hat{\boldsymbol{\sigma}}_i]$, $\mathbf{y}_i^+ = [\hat{\mathbf{x}}_i^+, \hat{\boldsymbol{\sigma}}_i^+]$, $\mathbf{b}_i = [\mathbf{r}_i, \mathbf{s}_i]$, and finally, $\mathbf{y} = [\mathbf{y}_1, \dots, \mathbf{y}_N]$, $\mathbf{y}^+ = [\mathbf{y}_1^+, \dots, \mathbf{y}_N^+]$, and $\mathbf{b} = [\mathbf{b}_1, \dots, \mathbf{b}_N]$. Now the systems of equation that we are trying to solve can be written succinctly as

$$(\mathbf{A}_d + \mathbf{A}_o)\mathbf{y}^+ = \mathbf{b}. \quad (141)$$

The operators \mathbf{A}_d encompasses the self interaction of vesicles, which itself comprises evaluation of interfacial forces, surface differentiation, upsampling for anti-aliasing, and singular integration. The operator \mathbf{A}_o is for interaction between vesicles and involves the N -body interaction of vesicles and nearly-singular integration.

5.3.4 Solution Schemes

Given the system of equation for the evolution of vesicles, Equation (141), and the current position and tension, there are multiple ways with different levels of accuracy one can use to approximate the solution to this equation. On the one hand, one can perform a single Jacobi iteration on this system and then approximate the solution of each diagonal block with an explicit step. On the other hand, for the highest accuracy, one can solve the whole system of equations using an Krylov space iterative method like GMRES. Here, we itemize some of these schemes in the order of accuracy of solution.

- **Explicit:** Given the current position and tension, we can use them as an initial guess for a single block-Jacobi iteration on the evolution matrix

$$\mathbf{A}_d\mathbf{y}^+ = \mathbf{b} - \mathbf{A}_o\mathbf{y},$$

in which the diagonal blocks $\mathbf{A}_{ii}\mathbf{y}_i^+ = \mathbf{b}_i - (\mathbf{A}_o\mathbf{y})_i$ can be solved individually. To solve each diagonal block, we can first solve for tension $\hat{\boldsymbol{\sigma}}_i^+$

$$-\delta t \hat{\mathbf{L}}_{ii} \hat{\boldsymbol{\sigma}}_i^+ = \delta t \hat{\mathbf{P}}_i \hat{\mathbf{u}}^\infty + \delta t \hat{\mathbf{P}}_i \hat{\mathbf{B}}_{ii} \hat{\mathbf{x}}_i - \sum_{\substack{j=1 \\ j \neq i}}^{\infty} \hat{\mathbf{P}}_i \hat{\mathbf{D}}_{ij} \hat{\mathbf{x}}_j. \quad (142)$$

Then, we can *explicitly* update the position

$$\widehat{\mathbf{x}}_i^+ = \frac{1}{\alpha_i} \left[(\delta t \widehat{\mathbf{B}}_{ii} + \widehat{\mathbf{D}}_{ii}) \widehat{\mathbf{x}}_i + \delta t \widehat{\mathbf{T}}_{ii} \widehat{\boldsymbol{\sigma}}_i^+ + \mathbf{r}_i \right].$$

Apart from the low accuracy of this approach in estimating the solution of Equation (139), there is another major shortcoming in this approach. The fact that we used the explicit position to calculate tension, caused the surface velocity term to be erroneously zero. Therefore, in presence of viscosity contrast, this method is incapable of solving the evolution of vesicles correctly.

- **Lagged tension:** Following the same approach as above, we can calculate the new tension $\widehat{\boldsymbol{\sigma}}_i^+$, but instead of an explicit evaluation of the new position, we can use the first equation to solve for position, i.e

$$(\alpha_i \mathbf{I} - \delta t \widehat{\mathbf{B}}_{ii} - \widehat{\mathbf{D}}_{ii}) \widehat{\mathbf{x}}_i^+ = \delta t \widehat{\mathbf{T}}_{ii} \widehat{\boldsymbol{\sigma}}_i^+ + \mathbf{r}_i. \quad (143)$$

We used a variation of this approach for vesicles with no viscosity contrast in Chapter 4. This approach, will increase the accuracy for the new position, but has the same limitation in the presence of viscosity contrast.

- **Locally implicit:** To remove the limitation for cases with viscosity contrast, one can fully solve each of the diagonal blocks $\mathbf{A}_{ii} \mathbf{y}_i^+ = \mathbf{b}_i - (\mathbf{A}_o \mathbf{y})_i$. Letting \mathbf{y}^* denote the exact solution to Equation (139), i.e. $(\mathbf{A}_d + \mathbf{A}_o) \mathbf{y}^* = \mathbf{b}$. The error in the new position and tension is

$$\|\mathbf{y}^+ - \mathbf{y}^*\| = \|\mathbf{A}_d^{-1} \mathbf{A}_o (\mathbf{y} - \mathbf{y}^*)\| \leq \delta t \|\mathbf{A}_d^{-1} \mathbf{A}_o\| \|\mathbf{u}\|,$$

where \mathbf{u} denotes the velocity field. Hence, the magnitude of the error is controlled by $\delta t \|\mathbf{A}_d^{-1} \mathbf{A}_o\|$. Since the operator matrix \mathbf{A} depends on the configuration of vesicles, with fixed δt , one cannot guarantee a uniform error bound for the new tension and position over the course of a simulation. In Figure 33 we plot the relative error of one Jacobi iteration with respect to the exact solution of the linear system for the given configuration of two vesicles as a function of their distance. As it is shown in the plot, when the vesicles get very close, a situation that may arise when the viscosity

contrast is high in shear flow, doing a single Jacobi iteration introduces error with a very large magnitude, which in turn causes the simulation to diverge. In order to have a robust solver for the evolution of vesicles that is not sensitive to the configuration of vesicles, instead of having a fixed number of iteration for the solver, one needs to fix the error bound and solve the system of equations for the evolution of all vesicles with that prescribed accuracy.

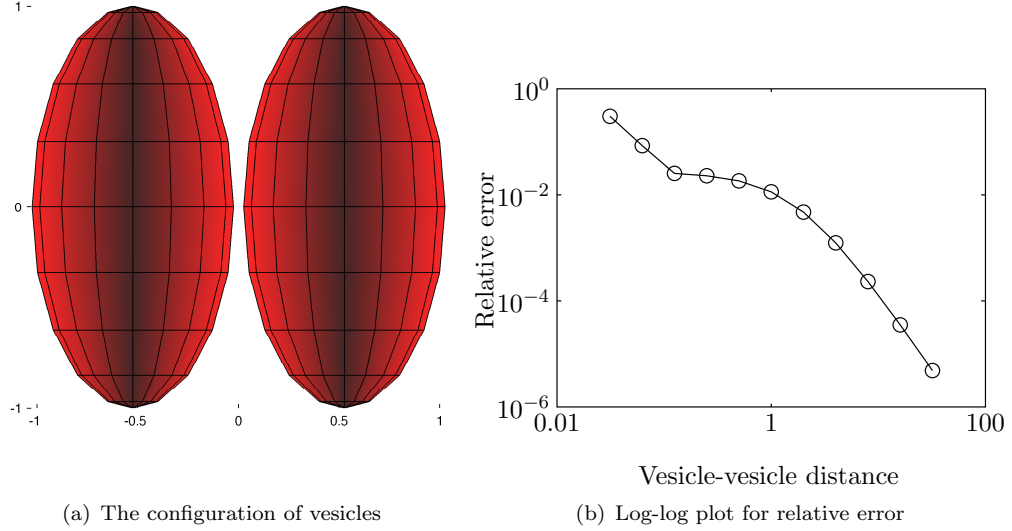


Figure 33: ACCURACY OF THE LOCALLY IMPLICIT METHOD FOR TWO VESICLES. *The relative error when the solution of Equation (139) is approximated by the result of a single Jacobi iteration. The configuration of vesicles is shown in the left subfigure. In the right subfigure, the ordinate show the relative error $\|\mathbf{y}^+ - \mathbf{y}^*\|/\|\mathbf{y}^*\|$ and the abscissa is the minimum distance between the surfaces of the two vesicles (the gap size in the left subfigure). The reference solution \mathbf{y}^* is calculated by an iterative method.*

- **Globally implicit:** Following the argument above, the most accurate way to evaluate the new position and tension is to solve Equation (139) to a prescribed tolerance. This can be done by taking multiple Jacobi iterations — but for the same reason given above the convergence rate can be very slow — or by solving the system of equation using a Krylov subspace method such as GMRES. In this method, we solve Equation (139) to a prescribed tolerance ϵ such that $\|\mathbf{A}\mathbf{y}^+ - \mathbf{b}\| \leq \epsilon\|\mathbf{b}\|$.

In Figure 34 we demonstrate the capabilities of the fully implicit scheme. The plots in the figure show the interaction of two vesicles in shear flow. The viscosity contrast of both vesicles is 10 and the reduced volume is 0.85. Since vesicles with such a

high viscosity contrast have a tendency to get very close in the shear flow, all other methods outlined above fail to capture their dynamics and diverge. This simulation is very interesting because with only $p = 8$ and in only 330 discrete time steps, we can capture very rich dynamics that involve close interaction of vesicles, tank-treading, and tumbling. Tank-treading and tumbling of vesicles are distinct features of vesicle dynamics, observed extensively in experiments [64, 120].

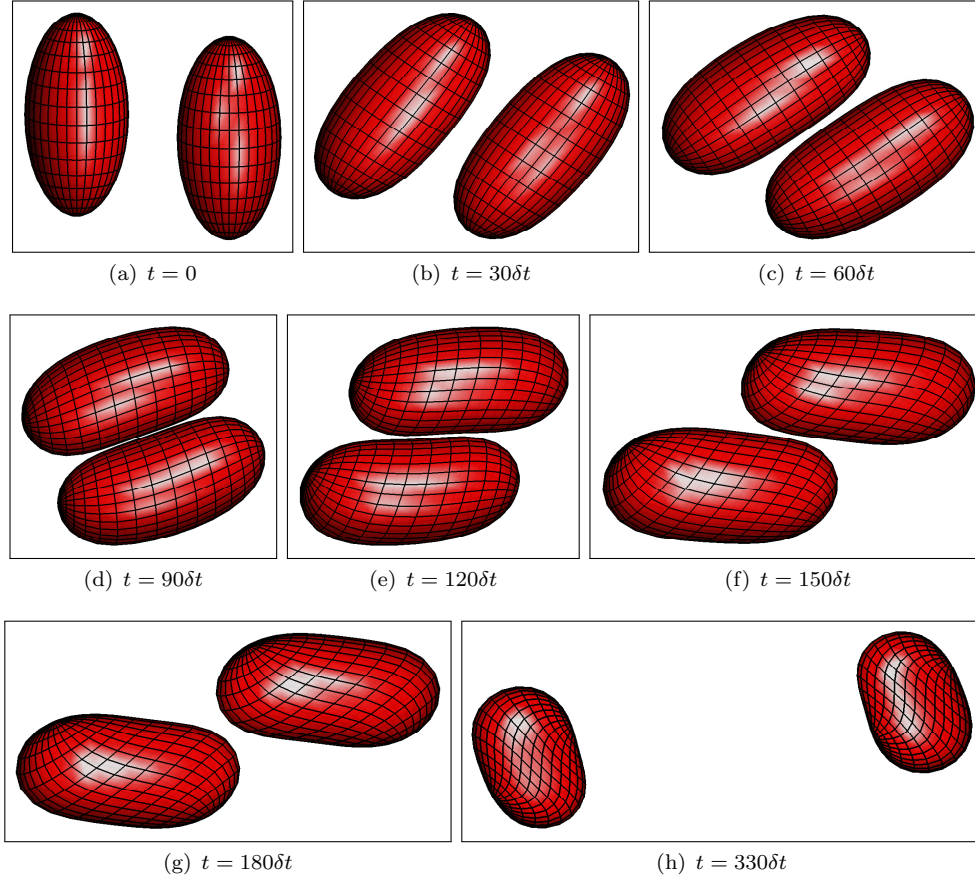


Figure 34: TWO VESICLES WITH VISCOSITY CONTRAST. *The plots show the interaction of two vesicles in the shear flow. The viscosity contrast of both vesicles is 10 and the reduced volume is 0.85. The time step δt is the same as the stable time step given in Table 20. For this simulation $p = 8$. Here we observe several characteristic vesicle dynamics, which are all observed experimentally. Namely, tanktreading (notable by following the grid lines over each surface) and tumbling of vesicles (subfigures g and h).*

To demonstrate the capabilities (and limitations) of the methods outlined above, we investigate the stable time step for these methods in case of one and two vesicles in shear flow in Table 20. Second to sixth columns in the table show the stable time step for a

single vesicle in shear flow. The explicit method shows stability constraint and as we refine the spatial resolution, the stable time step decreases. The lagged tension and implicit⁵ methods show no dependence on the spatial resolution. For a single vesicle, the difference between lagged tension and fully implicit methods is that we solve for tension and position simultaneously in the latter method. The last column in the table shows the stable time step for the interaction of two vesicles. An example of this type of simulation is shown in Figure 34. We failed to find any stable time step for both of the Jacobi based methods. For the globally implicit case, however, the time step is the same as for one vesicle. The Jacobi based methods have another limitation that they cannot capture the dynamics of vesicles with viscosity contrast. This is because, the same position vector is used as the operand for both of the double-layer operators and they cancel each other. As we point out in the footnote of the table, when the viscosity contrast is high ($\lambda = 10$) the vesicles act more like rigid bodies and get very close to each other. In these situations, our simple upsampling approach for the nearly-singular integrals fails. To correctly handle these kinds of situations where the minimum distance of vesicles is less than the spatial grid size, a collision detection algorithm needs to be used.

In the simulation of vesicles, the most expensive computation is the singular integration and with a given accuracy, the choice of the time stepping method depends on which of the solution schemes has the least number of singular integration per unit time of simulation and is therefore faster. Since the tension is the Lagrange multiplier that enforces the local inextensibility of the vesicles, even in the explicit case, at each time step we need to solve for tension. In Table 21 we report the computational cost (number of singular integral per unit time of simulation) for two of the methods. It is evident from the table that in addition to higher accuracy, the implicit case is superior to the explicit case considering the cost per time unit.

⁵For one vesicle, locally and globally implicit cases are equivalent.

Table 20: STABLE TIME STEP FOR VESICLES WITH VISCOSITY CONTRAST. *The entries in the table are the non-dimensional stable time step multiplied by the nondimensional shear rate, i.e $\chi \delta t_{\text{stable}}$. For all schemes we calculated the stable time step for $\chi = 1, 10$, and 100. In the explicit case, these values varied slightly for different shear rates and we report the largest observed number. In other cases, no dependence on χ was observed. For all simulations, the bending modulus was set to 0.01, the time horizon was fixed to $T = 10\tau/\chi$ and was chosen such that one tanktreading or tumbling revolution was observed, the aliasing tolerance was $1e-4$, and the reparametrization tolerance was $1e-4$. In the case of two vesicles, we were not able to successfully capture the close interaction of vesicles in the explicit or lagged tension approaches. In both of these cases, the vesicles collide and the time stepping diverges. For a single vesicle, locally and globally implicit schemes are equivalent and for two vesicles, the locally implicit scheme also diverges. It is only with the globally implicit case that we can capture the dynamics of vesicles with high viscosity contrast. In the globally implicit case we observe much larger stable time step.*

λ	Explicit					
	$p = 8$	12	16	24	32	Two vesicles ($p = 16$)
1	$2.7e-2$	$6.7e-3$	$1.7e-3$	$8.4e-4$	$2.1e-4$	—
1	Lagged tension					
	$5.4e-2$	$5.4e-2$	$5.4e-2$	$5.4e-2$	$5.4e-2$	—
1	Globally implicit					
	$2.1e-1$	$2.1e-1$	$2.1e-1$	$2.1e-1$	$2.1e-1$	$2.1e-1$
	4	$2.1e-1$	$2.1e-1$	$2.1e-1$	$2.1e-1$	$2.1e-1$
	10	$2.1e-1$	$2.1e-1$	$2.1e-1$	$2.1e-1$	$2.1e-1$ †

† The dynamics of vesicles with high viscosity contrast are such that the vesicles may get very close to each other (similar to rigid bodies in Stokes flow). To simulate such flows, nearly-singular integrals need to be properly handled (as explained in this section). This stable time step is therefore not universal for all configurations.

Table 21: THE COMPUTATIONAL COST FOR A SINGLE VESICLE. *For each method, we report the average number of singular integrals per non-dimensional time unit of simulation. For the implicit case, inside the parenthesis, we report the average number of GMRES iterations per time step. We considered the flow of a single vesicle for this table. The vesicle has reduced area of 0.85. For each case, the time step is chosen from the Table 20. The time horizon is chosen such that tank-treading or tumbling of the vesicle is observed.*

λ	Explicit				
	$p = 8$	12	16	24	32
1	37	149	588	1190	4762
1	Implicit				
	203(42)	258(54)	268(56)	283(59)	290(60)
	133(27)	138(28)	134(28)	136(28)	137(28)
	88(18)	82(17)	82(17)	84(17)	84(17)

5.3.5 Preconditioning

Each of the solution schemes that we outlined in Section 5.3.4 involves solving linear systems of equations of different sizes. For the explicit scheme, we solve for the tension of each vesicle, for single Jacobi we solve for position and tension simultaneously, and in the implicit method, we solve the whole system of equation at once. As we discussed in Chapter 4 all these linear systems are ill conditioned. The condition number of tension operator \mathbf{L} , Equation (142), grows as $\mathcal{O}(p)$, and the condition number of the position operator, Equation (143), grows as $\mathcal{O}(p^3)$. To improve the convergence rate of our iterative solvers, here we use the same set of preconditioners that we proposed in Chapter 4. The tension operator is known analytically over \mathbb{S}^2 :

$$\mathbf{L}_{\mathbb{S}^2} = \text{diag} \left\{ -\frac{n(n+1)(2n^2+2n-1)}{(2n-1)(2n+1)(2n+3)} \right\}_{n=0,\dots,p}^{m=-n,\dots,n},$$

and we use its inverse as the preconditioner for the tension operator. For the position operator, we use a heuristic preconditioner that asymptotically matches the spectrum of bending operator, i.e. $\mathbf{M}_1 := \text{diag}\{(1 - \Delta t n^3)^{-1}\}$. For the locally implicit scheme we use $\mathbf{M}_2 = \text{diag}\{\mathbf{M}_1, \mathbf{L}_{\mathbb{S}^2}^{-1}\}$ as the preconditioner; and for the globally implicit method involving N vesicles, we use a block diagonal matrix with N of \mathbf{M}_2 matrices on its diagonal.

5.4 Overall Algorithm and Computational Complexity

In this section we will give a summary of computational steps involved in the simulation of a suspension of vesicles. We assume that there are N vesicles in the suspension and all of these vesicles are represented with spherical harmonics order of p . As we discussed in Section 5.3 the upsampling rate for differentiation depends on the shape of each vesicle. Nevertheless, for the sake of clarity, we assume that the same upsampling frequency q is shared by all vesicles. The generalization to the case in which each vesicle has a different spherical harmonics order and different upsampling rate is easy. The summary of the computational steps as well as their computational complexity is as follows:

- **Spherical harmonic transform:** There are several algorithms for the forward and inverse spherical harmonic transform [24]. The complexity of these algorithm ranges

from $\mathcal{O}(p^4)$ for direct computation to $\mathcal{O}(p^2 \log^2 p)$ [89]. As it is stated in [89], the threshold where the (asymptotically) fast method becomes faster than the direct method is $p = 128$. In the range of frequencies we use, direct Legendre transform in the latitude and FFT in the longitude direction with the complexity of $\mathcal{O}(p^3)$ has the fastest running time. We let $C_{\text{SHT}}(p)$ denote the complexity of spherical harmonics transform for truncation order of p .

- **Differentiation:** Given the spherical harmonic coefficients of a function, differentiation is trivial in a Galerkin setup and a computational cost is only incurred when a mapping to the real space is performed. Since the surface differentials have geometric terms involved, mapping to the real space is required for them. Our dealiasing algorithm also includes a step where the surface is upsampled to some higher frequency q . Therefore the complexity of differentiation over each surface is $C_{\text{SHT}}(q)$.
- **Singular integrals:** The single- and double- layer integrals over each vesicle, namely Equation (129a) and Equation (129b), can be efficiently computed using the algorithm given in [51, 58, 136] with complexity $C_{\text{Sing}}(p) = \mathcal{O}(p^5)$.
- **Vesicle-vesicle interaction:** The vesicle-vesicle interaction involves the evaluation of the non-singular Stokes integrals, which may involve evaluation of nearly-singular integrals. Our current approach to avoid nearly-singular integrals is to upsample the surfaces to some finer grid. Afterwards, we use Fast Multipole Method [142, 143, 144] to evaluate the Stokes integrals. The complexity of FMM is linear with respect to the number of discretization points, $C_{\text{FMM}}(N, p) = \mathcal{O}(Np^2)$. Therefore, in a Galerkin context, the complexity of vesicle interaction is $C_{\text{Inter}}(N, p) = C_{\text{FMM}}(N, p) + NC_{\text{SHT}}(p)$, where the second term is due to the upsampling operation.
- **Evolution operator:** The evolution operator, which is given in Equation (141), involves the evaluation of singular integrals over all surfaces (\mathbf{A}_d) and the interaction of vesicles (\mathbf{A}_o). The computation complexity of \mathbf{A}_d is $NC_{\text{Sing}}(p)$ and the complexity of \mathbf{A}_o is $C_{\text{Inter}}(N, p)$.

- **Reparametrization:** The reparametrization algorithm is given in Algorithm 4.6.1. Each reparametrization step includes differentiation (to compute the normal vector) and spherical harmonic transform for filtering. This implies that each step has the complexity of $C_{\text{SHT}}(q)$. In all of our simulations, we never experienced a situation where more than a limited number of reparametrization steps (say more than 10) were needed. Thus, we assume $C_{\text{Reparam}}(q) = C_{\text{SHT}}(q)$.
- **Upsampling rate calibration:** The algorithm to compute the upsampling rate is given in Algorithm 5.3.1. The body of the algorithm inside the while loop has $\mathcal{O}(q^3)$ complexity, and the loop is evaluated for v times (where v is the final upsampling rate). Therefore, the calibration step has complexity $\mathcal{O}(p^3 v^4)$. Because the shape of each vesicle does not dramatically change in a few steps, we only perform the calibration occasionally and in practice this step has no extra cost compared to differentiation.

In Algorithm 5.4.1 we give the general simulation steps for a suspension of N vesicles.

5.5 Conclusions

We proposed numerical schemes to simulate the motion of inextensible vesicles suspended in unbounded domains based on Galerkin method. We have demonstrated, through numerical experiments, that the implicit scheme does not exhibit a mesh-dependent high-order stability constraint on the time-step size and the cost per time step is superior to that of the explicit scheme. Our schemes exhibits first-order accuracy in time and spectral-accuracy in space. We have presented efficient low-cost preconditioners to solve the discrete evolution equations by iterative solvers. An extension of our work would be to design an algorithm to handle nearly-singular integrals in the context of vesicle flows.

Algorithm 5.4.1 MAIN STEPS FOR SIMULATION OF VESICLE FLOWS.

- (a) From the right hand side for the evolution equation (Equation (141)) using the current position of vesicles \mathbf{x} .
 $\text{// } \mathcal{O}(NC_{\text{SHT}}(q) + \mathbb{I}_{\lambda \neq 1}(NC_{\text{Sing}}(p) + C_{\text{Inter}}(p)))$, where $\mathbb{I}_{\lambda \neq 1}$ is the indicator function, signifying the fact that the double-layer terms are only present when there is viscosity contrast.
- (b) Approximate the solution to the evolution equation by either of the following methods:
- (i) an explicit step,
 $\text{// } \mathcal{O}(C_{\text{Inter}}(p) + NKC_{\text{Sing}}(p))$, where K is the average number of iterations for tension solver, Equation (142).
 - (ii) lagged tension method,
 $\text{// } \mathcal{O}(C_{\text{Inter}}(p) + N(K_p + K_t)C_{\text{Sing}}(p))$, where K_p and K_t are, respectively, the average number of iterations for tension and position solvers, Equation (142) and Equation (143).
 - (iii) a locally implicit step,
 $\text{// } \mathcal{O}(C_{\text{Inter}}(p) + KNC_{\text{Sing}}(p))$, where K is the average number of iteration for the diagonal blocks.
 - (iv) a globally implicit step.
 $\text{// } \mathcal{O}(K(C_{\text{Inter}}(p) + NC_{\text{Sing}}(p)))$, where K is the average number of iteration for the linear solver for Equation (141).
- (c) Reparametrize the surfaces based on Equation (93).
 $\text{// } \mathcal{O}(NC_{\text{SHT}}(q))$
- (d) Calibrate differentiation upsampling rate using Algorithm 5.3.1.
 $\text{// } \mathcal{O}(NC_{\text{SHT}}(q))$
-

CHAPTER VI

PETASCALE DIRECT NUMERICAL SIMULATION OF BLOOD FLOW ON 200K CORES AND HETEROGENEOUS ARCHITECTURES

In this chapter, we present a fast, petaflop-scalable algorithm for Stokesian particulate flows. Our goal is the direct simulation of blood, which we model as a mixture of a Stokesian fluid (plasma) and red blood cells (RBCs). Directly simulating blood is a challenging multiscale, multiphysics problem. We report simulations with up to 200 million *deformable* RBCs. The largest simulation amounts to 90 billion unknowns in space. In terms of the number of cells, we improve the state-of-the art by several orders of magnitude: the previous largest simulation, at the same physical fidelity as ours, resolved the flow of $\mathcal{O}(1,000 - 10,000)$ RBCs.

The new method has been implemented in the software library MoBo (for “Moving Boundaries”). We designed MoBo to support parallelism at all levels, including inter-node distributed memory parallelism, intra-node shared memory parallelism, data parallelism (vectorization), and fine-grained multithreading for GPUs. We have implemented and optimized the majority of the computation kernels on both Intel/AMD x86 and NVidia’s Tesla/Fermi platforms for single and double floating point precision.

Overall, the code has scaled on 256 CPU-GPUs on the Teragrid’s Lincoln cluster and on 200,000 AMD cores of the Oak Ridge National Laboratory’s Jaguar PF system. In our largest simulation, we have achieved 0.7 Petaflops/s of sustained performance on Jaguar.

6.1 Introduction

As we mentioned in Chapter 1, clinical needs in thrombosis risk assessment, anti-coagulation therapy, and stroke research would significantly benefit from an improved understanding of the microcirculation of blood. Toward this end, we present a new computational infrastructure, MoBo, that enables the *direct numerical simulation of several microliters of*

blood at new levels of physical fidelity (Figure 35). MoBo consists of two key algorithmic components: (1) scalable integral equation solvers for Stokesian flows with dynamic interfaces; and (2) scalable fast multipole algorithms. In terms of size alone, MoBo’s overall simulation capability represents an advance that is orders of magnitude beyond what was achieved in prior work in blood flow simulation.

We use an algorithmically optimal semi-implicit scheme. Unlike explicit time-stepping schemes that require only near-neighbor communication (but are algorithmically suboptimal for our problem), our solver requires global communication at every time step. Consequently, our solver is much more challenging to scale than an explicit solver. Nevertheless, our results demonstrate that it is possible to successfully scale implicit solvers to hundreds of thousands of cores.

Challenges in direct numerical simulation of blood. Just one microliter of blood of a healthy individual contains approximately four million RBCs. The surrounding plasma, which is a viscous fluid, mechanically couples every RBC to all other RBCs. Furthermore, RBCs are highly deformable (it is the deformability of RBCs which determines the rheological properties of blood). The large number of cells and their complex local and global interactions pose significant challenges in designing tools for high-fidelity scalable numerical simulations of blood.

Due to these difficulties, multiphase blood flow simulations have been restricted to relatively small number of RBCs. For example, the largest simulations today have scaled to 1,200 cells [155] (using boundary integral equations, like us) and 14,000 cells [35] (using lattice Boltzmann methods). The latter work, which is based on an explicit time-stepping scheme, scaled up to 64K cores but requires an excessive number of time steps due to the non-physical stiffness introduced by the numerical scheme. Other methods that model RBCs as rigid bodies have scaled to an even large number of RBCs; but these are crude approximations of the blood flow. Deformable models of RBCs (Figure 35) are critical for

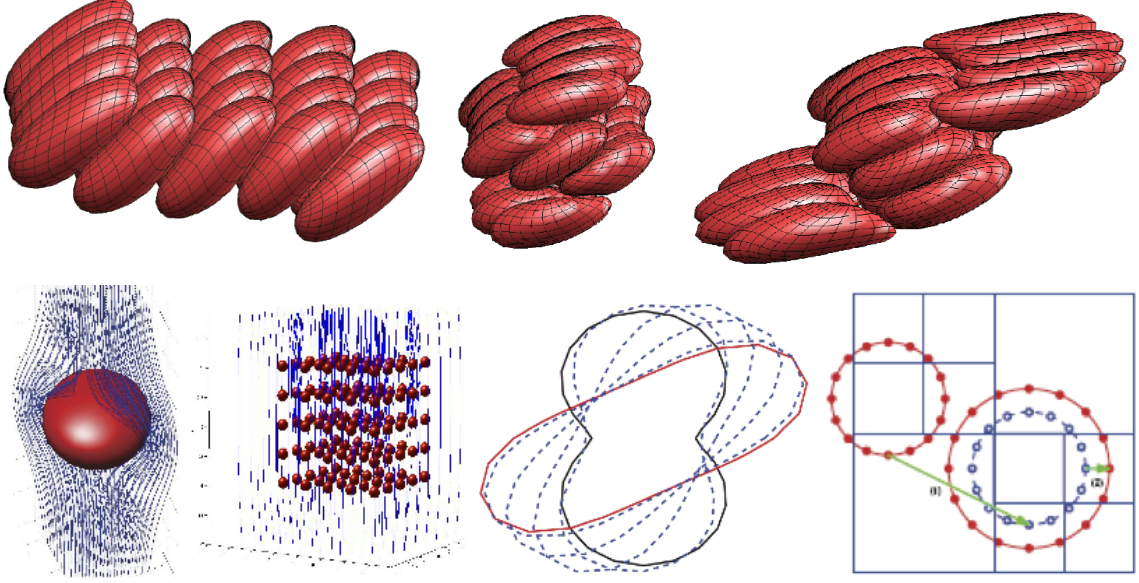


Figure 35: SUMMARY OF THE COMPUTATIONAL INFRASTRUCTURE FOR DIRECT NUMERICAL SIMULATION OF BLOOD FLOW. *In the top row, we depict a few snapshots from the flow of twenty RBCs that are immersed in plasma. At every time step, a Stokes problem must be solved in the exterior and interior of the RBCs. We have developed computational tools for this problem. The main algorithmic components include: (a) spectral RBC shape representations and quadratures for singular integrals on these shapes; (b) accurate modeling of the hydrodynamic interactions between many-RBCs; (c) nonlinear solvers for the mechanics of RBC deformations; and (d,e) parallel, kernel-independent, tree-based, fast summation methods. The advantage of boundary integral methods is that only the RBC boundary is discretized and no discretization of the space between RBCs is necessary. This is crucial for reducing the number of degrees of freedom and eliminates the need for difficult-to-parallelize 3D unstructured mesh generation. Our tools enable parallel and highly accurate simulations of microcirculation phenomena of blood flow. We have achieved the direct numerical simulation of about fifty microliters of blood flow (One can think of the volume of a single blood drop as being roughly equivalent to one microliter).*

accurate blood flow simulations.¹

6.1.1 Synopsis of Our Approach

We model RBCs as deformable viscous sacs with an inextensible, massless membrane that can sustain bending and tension forces. The surrounding plasma is modeled as a Stokesian fluid (we neglect inertial terms). There are several challenges in simulating such a system:

- The evolution of the RBCs requires solving the Stokes equations in the plasma—a very complex geometric region that changes at every time step.

¹For example, a 65% volume-fraction suspension of rigid spheres cannot flow; blood flows even when the volume-fraction of RBCs in the plasma reaches 95% [8].

- Computing the bending and tension forces requires accurate geometric description of the shape of the RBCs. Furthermore, these forces introduce significant numerical stiffness.

To address these challenges, we use

- an integro-differential formulation in which we couple a boundary integral formulation for the Stokes equations (plasma) with the RBC’s membrane elasticity;
- a semi-implicit time-stepping scheme that removes the stiffness due to interfacial forces;
- spherical harmonics representations for the shape and the deformation of RBCs;
- the fast multipole method to accelerate the long-range hydrodynamic interactions between cells and plasma; and
- distributed and shared memory parallelism, SIMD parallelism (vectorization), and fine-grained multithreading via GPGPU acceleration, to expose maximum concurrency.

MOBo employs Fourier and Legendre transforms, adaptive fast multipole methods, Galerkin projections, multi-step time marching, fast spherical harmonics rotations, spectral quadratures for smooth and weakly singular integrals, preconditioned Krylov linear solvers, and dense linear algebra.

Our overall formulation can be outlined as follows. We use a spherical harmonics representation for the boundary of every RBC. This choice is mathematically equivalent to tracking a number of points on the surface of the RBC. In our simulations, we typically track either 84 or 312 points. The motion of each such point \mathbf{x} is governed by

$$\begin{aligned}\frac{\partial \mathbf{x}}{\partial t} &= \mathbf{v}(\mathbf{x}), \\ \mathbf{v}(\mathbf{x}) &= \mathbf{v}_{\text{self}}(x) + \mathbf{v}_{\text{interaction}}(x) + \mathbf{v}_{\text{background}}(x).\end{aligned}\tag{144}$$

Here, \mathbf{v} is the velocity of the point, which we decompose into three components: local, global, and background velocities. Roughly speaking, the “*local*” velocity, \mathbf{v}_{self} , accounts

for the interactions between the specific point in the RBC under consideration and all of the other points within the same RBC. The “*global*” velocity, $\mathbf{v}_{\text{interaction}}$, accounts for all of the interactions occurring across all of the RBCs in the simulation. The “*background*” velocity, $\mathbf{v}_{\text{background}}$, is the imposed flow field. This work builds on our previous work on massively parallel tree-data structures [117, 127], parallel and kernel independent fast multipole methods [33, 77, 143], and fast solvers for particulate flows (Chapter 4).

6.1.2 Contributions

In Chapter 2 and Chapter 4, we presented the details of the formulation and the numerical algorithms that are required to compute \mathbf{v}_{self} and $\mathbf{v}_{\text{interaction}}$. Here, we focus on the parallelization and performance analysis for the computation of \mathbf{v}_{self} and $\mathbf{v}_{\text{interaction}}$. \mathbf{v}_{self} requires nine different kernels. $\mathbf{v}_{\text{interaction}}$ uses the FMM, which in turn has five major computational phases (tree construction, three tree-traversals, and the direct interactions). Our key contributions are:

- We present a hybrid-parallel implementation of nine computational kernels that MOBo uses for the computation of \mathbf{v}_{self} and $\mathbf{v}_{\text{interaction}}$. The kernels are multithreaded and work-partitioned between CPU and GPU, which execute concurrently, thereby delivering excellent per-node performance.
- The most intensive kernels in our computation have been designed for locality, accuracy, and computational efficiency, capitalizing in particular on highly optimized BLAS3 (GEMM) operations.
- We further improve the performance of the SC’09 FMM algorithm [77]. These improvements include explicit SSE vectorization and multithreading via OpenMP, as described in prior work [33]. In this paper, we add simultaneous asynchronous GPU acceleration.
- We present single-node analysis for computations of \mathbf{v}_{self} and $\mathbf{v}_{\text{interaction}}$ on AMD, Intel, and NVIDIA platforms.

- We present weak and strong scaling results on the Jaguar PF system at Oak Ridge National Laboratory (ORNL).

Performance Highlights. We achieve 780 TFlop/s of sustained performance on the 196,608 cores of the AMD Istanbul-based Jaguar PF system (4 GFlop/s per core), with $160\times$ speedup on strong scaling when moving from 48 to 24,576 cores ($512\times$); and 75% efficiency for the weak scaling. On other platforms, we demonstrate up to 18 GFlop/s per core of sustained performance on the Intel Nehalem-EP; and up to 350 GFlop/s per NVIDIA Fermi C2050 card (both in single precision).

In our largest simulation, we solved a problem involving 8,000 RBCs per MPI process, on 32,768 MPI processes for a total of 196,608 cores. We discretized using 84 points per RBC. This set of parameters results in a total of 262,144,000 RBCs (50 drops of blood) and 90 billion unknowns per time step. (We have four unknowns per point: the three coordinates and a scalar tension.)

Limitations. Despite its capabilities, our method has several limitations. First, Stokes formulation and MoBo are restricted to very low Reynolds numbers and, therefore, cannot accurately be used to simulate high-Reynolds blood flow (e.g., flow in large arteries). Second, the discretization of the RBCs is not adaptive: all of the RBCs are approximated using the same number of points. Third, the current version of MoBo does not support confined boundaries. The modifications of the method for the confined boundary case has been presented in [110] but we have not yet parallelized the method. Fourth, the memory requirements of the method grow with the cube of the number of points per RBC.

6.1.3 Related Work

In spite of advances in understanding the complex behavior of particulate flows [104], only recently have algorithmic advances allowed accurate 3D simulations of Stokes flows with hundreds of deformable particles using boundary integrals. Attempts to parallelize integral equation solvers have been restricted to low-accuracy discretizations, spatially uniform particle distributions, and have not scaled to large numbers of cores. The main challenges

are the parallelization of the hydrodynamic interactions, the stiffness of the RBC deformations, and end-to-end scalability and performance for all of the algorithmic components of a method.

Impressive simulations based on *fictitious/immersed boundary* methods have been reported in [1, 8, 49, 56, 79, 132]. However, there is limited work in efficiently parallelizing these methods and no scaling on thousand-core machines have been achieved [55, 87, 124, 147]. In blood rheology simulations, there are at least two examples of simulations with large numbers of particles: one which models a $50\mu\text{m}^2 \times 500\mu\text{m}$ -capillary blood flow with 300 thousand rigid particles [97], and another in which a *dissipative particle dynamics* method was used to model a few thousand deformable RBCs [46]. *Lattice-Boltzmann* based methods for particle simulations have been used for blood flow but are limited to rigid RBCs [126]. An exception is the lattice-Boltzmann approach in [35], that allows for deformability of RBCs. However, lattice Boltzmann methods are low-order accurate in space and require small time steps due to numerical stiffness [93]. Lastly, another class of methods are based on *moving-mesh finite-element methods* [128]. Such methods are difficult to parallelize in particular, for the case of large 3D deformations due to the need for unstructured mesh generation [21].

A different class of methods based on *boundary integral equation* formulations is *ideal for blood flow since it only requires discretization of the RBC membrane*, which is more scalable than discretizing the volume occupied by the plasma [17, 72, 104, 106, 148, 151]. The aforementioned successful simulation of 1200 deformable drops [155] used a boundary integral formulation. But, that implementation was sequential and required 120 CPU hours to complete. Overall, limited work exists in parallelizing such methods. One exception is the parallel Stokes solver of Thien, et al. [96]; their calculations, however, were performed using a suboptimal $\mathcal{O}(N^2)$ algorithm and not a fast multipole method.

6.2 Formulation and Algorithms for Particulate Flows

Notation. Before we proceed with describing the kernels of MoBo, let us introduce some notation. (We use MATLAB’s notation for linear algebra operations.) We use upper-case

letters for matrices and lower-case letters for vectors. We use upper-case bold-face letters for discretizations integral and differential operators, and lower-case bold-face letters to denote vectors and points in \mathbb{R}^3 .

Table 22: INDEX OF FREQUENTLY USED SYMBOLS AND OPERATORS IN CHAPTER 6.

Symbol	Definition
m	Number of points used to discretize the RBC surface
q	Spherical harmonics expansion order
n	Total number of Red Blood Cells
p	Number of processors
\mathbf{v}	Velocity
\mathbf{F}	Discrete Fourier Transform operator
\mathbf{P}_k	k^{th} -order Discrete Legendre transform
$\mathbf{S}, \mathbf{S}^{-1}$	Forward/inverse spherical harmonics transform

The mathematical formulation of the governing equation is given in detail in Chapter 4. Formally, the evolution of a point on the membrane of a RBC reads as

$$\frac{d\mathbf{x}}{dt} = \mathbf{v}(\mathbf{x}), \quad \text{for } \mathbf{x} \in \gamma_k, \text{ and all } k, \quad (145)$$

where \mathbf{x} denotes a point on γ_k . As mentioned in the introduction, the velocity $\mathbf{v}(\mathbf{x})$ can be decomposed into three components:

$$\mathbf{v}(\mathbf{x}) = \mathbf{v}_{\text{self}}(\mathbf{x}) + \mathbf{v}_{\text{interaction}}(\mathbf{x}) + \mathbf{v}_{\text{background}}(\mathbf{x}) \quad (146)$$

The first term, \mathbf{v}_{self} , at a point \mathbf{x} on γ_k (the membrane of the k^{th} RBC) depends only on the shape of γ_k . The second term depends on the shapes of all of the RBCs in the simulation and requires an N-body calculation. The third term, $\mathbf{v}_{\text{background}}$ is known analytically and depends on the numerical experiment (e.g., imposed shear flow). The precise expressions for \mathbf{v}_{self} is given in Chapter 4, as the Stokes integral with the the surface forces as density. The global velocity is easy to state:

$$\mathbf{v}_{\text{interaction}}(\mathbf{x}_k) = \sum_i \frac{1}{\rho_{ki}} \left(\mathbf{d}_i + \frac{(\mathbf{r}_{ki} \cdot \mathbf{d}_i) \mathbf{r}_{ki}}{\rho_{ki}^2} \right), \quad (147)$$

where $\mathbf{r}_{ki} = \mathbf{x}_k - \mathbf{x}_i$, $\rho_{ki} = |\mathbf{r}_{ki}|$, \mathbf{d}_i is the given density at point \mathbf{x}_i , $\mathbf{r}_{ki} \cdot \mathbf{d}_i$ denotes the geometric dot product between these two vectors, and $|\cdot|$ denotes the vector norm in \mathbb{R}^3 .

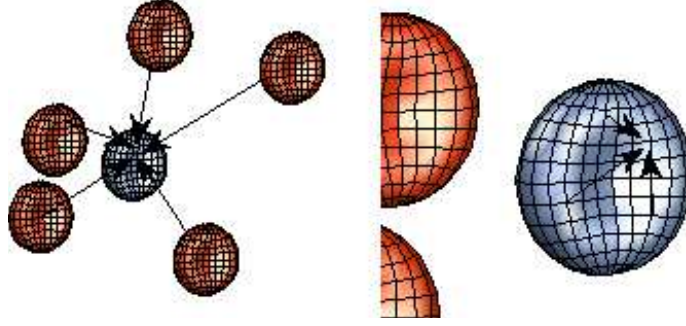


Figure 36: AN EXAMPLE OF GLOBAL AND LOCAL INTERACTIONS. *The interaction ($\mathbf{v}_{\text{interaction}}$) between points on the surface of the blue membrane and points on the surfaces of the other red membranes is **global** (left figure). The interaction (\mathbf{v}_{self}) between points on the surface of the same (blue) membrane is an example of **local** interactions (right figure).*

Representation of the surface. Let $U = \{(\theta, \phi) : \theta \in (0, \pi), \phi \in (0, 2\pi)\}$. Then, we denote a parametrization of the surface by $\mathbf{x} : U \rightarrow \mathbb{R}^3$. θ parametrizes the latitude and ϕ parametrizes the longitude. In this way, \mathbf{x} can be represented in the spherical harmonics basis with spectral accuracy (Chapter 4). This spectral representation enables fast and accurate computation of high-order derivatives of \mathbf{x} with respect to θ and ϕ . Such derivatives are required for the calculation of \mathbf{v}_{self} . (For example, \mathbf{v}_{self} depends on the surface Laplacian of the mean curvature of the membrane, the normal vector, and other geometric quantities.)

6.2.1 Overall Algorithm and the Main Computational Kernels

Due to space limitations, we describe a first-order explicit Euler time-marching scheme. (That is explicit on the RBC membrane mechanics. The Stokes equations for the plasma and the interior of the cells are solved *always implicitly*.) The actual time-stepping scheme used in these simulations is a multistep, semi-implicit scheme and is described in detail in Chapter 4.²

Given n RBCs, each one being represented by its surface γ_k a set of points on this surface \mathbf{x} , the algorithm proceeds as follows:

1. Compute $\mathbf{v}_{\text{self}}(\mathbf{x}, \gamma_k)$, for all $\mathbf{x} \in \gamma_k$, $k = 1, \dots, n$.
2. Compute $\mathbf{v}_{\text{interaction}}$ using FMM;

²The linear solves in the semi-implicit are done with a Krylov method, which requires a matrix-vector operation. The latter has been implemented using the computational kernels described in this section.

3. Evaluate $\mathbf{v}_{\text{background}}$ analytically;
4. Update the position using $\mathbf{x}_{\text{new}} = \mathbf{x} + \Delta t(\mathbf{v}_{\text{self}}(\mathbf{x}) + \mathbf{v}_{\text{interaction}}(\mathbf{x}) + \mathbf{v}_{\text{background}}(\mathbf{x}))$,

where Δt is the time-step size.

For the semi-implicit scheme described in Chapter 4, the first step is embarrassingly parallel across k (RBCs). For each cell, the complexity of computing \mathbf{v}_{self} for all \mathbf{x} on its surface is (q^6) . The second step involves a communication-heavy, all-to-all, N-body calculation with the Stokes kernel. The complexity of the Stokes kernel was analyzed in [77]. We briefly summarize the main components of FMM in the next subsection.

6.2.2 Global Interactions: FMM Kernel

The FMM consists of two main steps, the octree construction phase and the evaluation phase. The evaluation phase consists of the computation of the far-field approximated interactions and the exact near-field direct interaction. Specifically, the evaluation phase involves the following substeps:

1. a bottom-up (post-order) tree-traversal to compute the multipole-moments approximation;
2. an arbitrary-order traversal to translate multipole-moments to local approximations;
3. a top-down (pre-order) tree traversal to accumulate all far-field interactions to the target points, the so-called “VXW-list”-calculation in FMM jargon; and
4. a near-neighbor exchange at the leafs to compute the near-range interactions, the so-called “U-list”-calculation in FMM jargon.

In parallelizing FMM, the main challenges are the tree construction, and the communication involved in step (2), which in principle is local, but involves “fat” neighbors that can cause communication imbalance for highly nonuniform trees. One can prove, under assumptions on the distribution of RBCs, that the complexity of the tree-construction phase is $\mathcal{O}(\frac{n}{p} \log \frac{n}{p}) + \mathcal{O}(p \log^2 p)$ and of the evaluation phase is $\mathcal{O}(\frac{n}{p}) + \mathcal{O}(\sqrt{p})$, where p is the number of MPI processes.

In [77], a novel tree-construction algorithm, a novel hypercube-based reduction for the V-list calculation, and a hybrid GPU-MPI implementation for the U-list and V-list calculations were proposed. However, that approach left the multi-core CPUs that were driving the GPU spinning idle. In , we introduced a set of optimizations that can further accelerate the CPU computations (OpenMP acceleration and SSE vectorization).

In Section 6.3, we report results from a different multithreading strategy. We employ a hybrid OpenMP-MPI-CUDA scheme in which we compute the dense interactions in parallel with the far-field interactions. The CPU sockets are responsible for the far-field computations (V-list) and the GPUs are responsible for the direct interactions (U-list). In addition, we have introduced several optimizations that are specific to our Kernel Independent FMM [143].

We use Streaming SIMD Extensions (SSE) technology, available in a number of modern CPUs, to speed up the floating point computations. In a nutshell, using SSE allows to perform basic arithmetic operations on small vectors of floating point numbers. Specifically, vectors can consist of either four single-precision floating point numbers or two double-precision floating point numbers (in any case vectors are 128 bit long). Arithmetic operations on different vector entries are performed by the CPU in parallel, thus speeding up the computation (in an ideal scenario) by a factor of four or two, depending on the precision. We use SSE to accelerate the particle-to-particle interactions (specifically, the evaluation of the Stokes kernel). We replicate the data associated with each target point (e.g., x -coordinate) into four entries of an SSE vector, and load another SSE vector with x -coordinates of four different source points. Then we apply SSE vector subtraction to evaluate 4 differences in parallel, then we square these four differences in parallel and so forth. Eventually we obtain four potentials in an SSE register and sum them up. We use this approach for the source-to-upward equivalent densities. This work is explained in further detail in [33].

Finally, we use point-to-point interactions without precomputation in many parts of the algorithm to improve float-to-memory access ratios and improved overall performance.

Repartitioning of Red Blood Cells. Partitioning RBCs among MPI processes is done using the FMM underlying data structure. This is necessary because after a few time steps the partitioning of RBCs does not match the optimal partitioning for the FMM and this results to excessive communication. The parallel FMM code that we are using [142], requires particles to be Morton-sorted not only locally on each MPI task but also across MPI tasks. That is, in order to apply FMM for computations, we first have to redistribute (“scatter”) the points between tasks so that points become Morton-sorted, then evaluate the potentials and finally scatter the potentials back to the original layout of the points. When using multiple (say, tens of thousands) MPI tasks, the cost of these two scatters can become prohibitive, unless special measures are taken. We periodically re-distribute the RBCs between MPI tasks, so that the overall distribution of points is “close” to being Morton sorted, and thus, the scatters are relatively inexpensive. Specifically, we use the partitioning of the space between MPI tasks produced by the last call to FMM. For each RBC, we determine preliminary “target MPI task” for each point of the RBC. Then we decide the final MPI task for the RBC by the voting procedure. For the actual data transfer we employ an `MPI_Alltoallv()` call.

6.2.3 Local Interactions: RBC Physics Kernels

The computation of \mathbf{v}_{self} consists of several kernels. In the following, we discuss nine kernels in which the majority of the computation takes place:

The spherical harmonics transform kernel. The spherical harmonics transform may be expressed in terms of matrix operations [24]. For a spherical harmonic expansion of order q , there are $2q$ points in the east-west direction and $q + 1$ points on the north-south direction. Let X , Y , and Z denote matrices, each of size $2q \times (q + 1)$, that hold the x -, y -, and z -coordinate components of the grid points, respectively. The points are stored in a “latitude-major order”. Then, the k^{th} order spherical harmonic coefficients of X is given by

$$\hat{X}_k = \mathbf{P}_k \mathbf{W}(\mathbf{F}X)_k^T, \quad k = 0, \dots, 2q, \quad (148)$$

where $\mathbf{F} \in \mathbb{R}^{2q \times 2q}$ denotes the discrete Fourier transform, $\mathbf{W} \in \mathbb{R}^{(q+1) \times (q+1)}$ is a diagonal matrix holding the Gaussian quadrature weights, $\mathbf{P}_k \in \mathbb{R}^{k \times (q+1)}$ is the k^{th} order associated Legendre transform and $\hat{X}_k \in \mathbb{R}^{k \times 1}$ is the vector of spherical harmonic coefficients of the k^{th} order. The same formula is also true for Y and Z . Considering the fact that q is rather small compared to the number of RBCs, it is best to perform both Fourier and Legendre transforms as matrix multiplications [42]. This also motivates the data structure for our implementation.

The inverse of spherical harmonics transform is given by

$$X = \mathbf{F}^T [\mathbf{P}_1^T \hat{X}_1 \dots \mathbf{P}_{2q}^T \hat{X}_{2q}]^T. \quad (149)$$

Hereinafter, we formally denote the forward and inverse spherical transforms by \mathbf{S} and \mathbf{S}^{-1} . When we have n surfaces, the complexity of the forward and inverse spherical harmonics transform is $\mathcal{O}(nq^3)$ and the depth is $\log q$.

In order to accelerate spherical harmonics transform, we represent the transform as a sequence of multiplications of *real* matrices, that is we use real DFT and not the complex one; we use BLAS or CUDA-BLAS for all the matrix multiplications; and we use column-wise (Fortran-style) storage for all the matrices.

The input data corresponding to different RBCs is packed into a $2q \times (q+1)n$ matrix. Each row of this matrix corresponds to a particular latitude across all RBC. Each column corresponds to a particular longitude on a particular RBC. Columns corresponding to the same RBC are grouped together (columns related to the first RBC are followed by the columns related to the second RBC, etc.)

We start the transform by multiplying the input matrix from the left by the DFT matrix, thus applying DFT to each column independently. Then we transpose the resulting matrix. Note that now each column of the transposed matrix corresponds to a particular frequency and one of the two possible functions (sine or cosine). We then treat each column of the transposed matrix as a $(q+1) \times n$ matrix (stored column-wise). Rows of this matrix correspond to different longitudes across all RBCs, and columns correspond to different RBCs. We then multiply this matrix from the left by an appropriate Legendre transform

matrix, as given in Equation (148).

Note that in case of GPU computations (CUDA-BLAS), we perform both matrix multiplications and the transpose on GPU.

All the data is stored in a one dimensional array and depending on the size parameter passed to the BLAS kernels, its content can be interpreted as matrices of different sizes. Using MATLAB's notation, we store the coordinates array for the k^{th} RBC as $C_k = [X(:)^T Y(:)^T Z(:)^T]$, and for all RBCs as $C = [C_1 \dots C_n]$. Using this structure, data can be streamed to BLAS subroutines for the calculation of spherical harmonics transforms.

Pole rotation kernel for weakly-singular quadratures. Given the surface C_k ($k = 1, \dots, n$) and a target point (x, y, z) on the same surface, there exists a linear transformation $\mathbf{R} \in \mathbb{R}^{m \times m}$ such that the pole for the surface $\bar{C} = \mathbf{R}C$ is located at (x, y, z) . Note that the transformation \mathbf{R} depends on the parametrization of the surface and the target point, but it is independent of the geometry of the surface. An example of this transformation is given in Figure 37. Let $\mathbf{R}_1, \dots, \mathbf{R}_{q+1}$ be the transformations with the target point as the grid points on the $\phi = 0$ meridian, then the transformation for other points on the θ_k latitude is a permutation of \mathbf{R}_k [54].

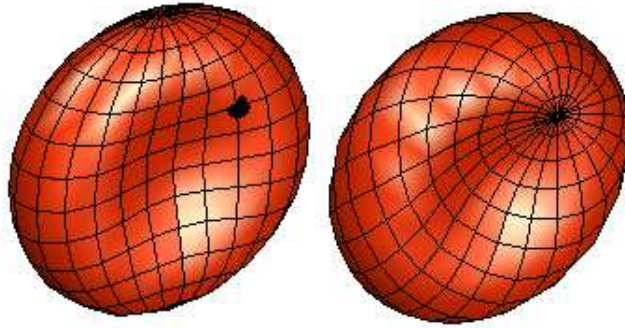


Figure 37: THE TYPICAL BICONCAVE RBC SHAPE. *The plot on the right is the same surface as in the one in the left, but the pole is moved to the point marked by the circle in the left figure.*

In our simulation, we need to perform this rotation for all of the points on the surface of a RBC. For our method, the complexity of a single rotation of the pole is $\mathcal{O}(q^4)$ and the memory requirement is $\mathcal{O}(q^5)$. As we mentioned in Chapter 4, there is another algorithm for the rotation of the pole that is based on the spherical harmonics expansion of the surface

[54]. This algorithm reduces the complexity to $\mathcal{O}(q^5)$ and the memory requirements to $\mathcal{O}(q^4)$. That algorithm has more expensive logic and it is preferable only for small number of RBCs. Table 23 summarizes the result of our comparison between the two algorithms.

Table 23: EXECUTION TIME FOR ROTATION OF THE POLE. *The execution time (ms) on a Tesla GPU to move the pole to all points on the surface for different algorithms*

n	Direct (cublas)		Rotation via spherical harmonics	
	$q = 6$	12	6	12
8	21.25	169.17	0.84	29.15
64	23.48	258.38	6.15	225.85
512	57.75	1360.07	47.85	1795.29
1024	98.02	2597.93	96.20	3589.00

The kernel for the weakly-singular integrals. The computation of $\mathbf{v}_{\text{self}}(\mathbf{x})$, $\mathbf{x} \in \gamma_k$ can be written as

$$\mathbf{v}_{\text{self}}(\mathbf{x}) = \int_{\gamma_k} G(\mathbf{x}, \mathbf{y}) \mathbf{f}(\mathbf{y}) d\mathbf{y},$$

where G is the Green's function for the Stokes equations. This integral has a weak-singularity for $\mathbf{y} = \mathbf{x}$ for all \mathbf{x} in γ_k except the two poles. One method to evaluate the Stokes' integral at a point on the surface is to move the pole to that target point and evaluate the integral for that particular point (Chapter 4). In Algorithm 6.2.1, we outline the evaluation of Stokes integral. For n surfaces, the work is $\mathcal{O}(nq^6)$ and the depth is $\mathcal{O}(\log q)$.

Algorithm 6.2.1 Evaluation of singular Stokes integral. $\mathcal{O}(q^6)$
 D is the input density and U is the evaluated potential.

```

for  $k = 0$  to  $q + 1$  do
  for  $l = 0$  to  $2q$  do
     $\mathbf{R} \leftarrow$  Permute  $\mathbf{R}_k$  for the target  $\phi_l$ 
     $\bar{C} = \mathbf{R}C, \bar{D} = \mathbf{R}D,$ 
    Evaluate  $U_{kl}$  (by direct Stokes kernel)
  end for
end for

```

After the multiplications by the rotation matrices \mathbf{R} , the most costly kernel in our simulation is the Stokes evaluation kernel. For the CPU code, this kernel is accelerated using SSE instructions.

The kernel for surface differentiation. The differentiation with respect to the ϕ parameter is a straightforward calculation using the DFT. But, differentiation with respect to θ needs extra care. With an abuse of notation, let $\mathbf{H}_k = d\mathbf{P}_k/d\theta$ and $\mathbf{W}_k = d^2\mathbf{P}_k/d\theta^2$. Then, to differentiate a function $G \in \mathbb{R}^{2q \times (q+1)}$ on the surface with respect to the parameter θ we have

- (i) $\hat{G} = \mathbf{S}G$,
- (ii) $dG/d\theta = \mathbf{F}^T[\mathbf{H}_1^T \hat{G}_1 \dots \mathbf{H}_{2q}^T \hat{G}_{2q}]^T$,
- (iii) $d^2G/d\theta^2 = \mathbf{F}^T[\mathbf{W}_1^T \hat{G}_1 \dots \mathbf{W}_{2q}^T \hat{G}_{2q}]^T$.

The complexity of these steps is the same as spherical harmonics transform and is $\mathcal{O}(q^3)$. With different matrices, the same kernel is used to evaluate the inverse spherical harmonics and all the derivatives.

Other computation kernels. We have implemented several other kernels that are required for computations on the surfaces, for instance, the computation of geometric properties of the RBC. These kernels include the geometric cross ($\mathbf{a} \times \mathbf{b}$), and dot ($\mathbf{a} \cdot \mathbf{b}$, $\mathbf{a}, \mathbf{b} \in \mathbb{R}^3$) products, matrix transpose, and scaling of vectors. They have been implemented on top of our specific data format for multiple RBCs and have been optimized on both CPUs and GPUs.

The kernel for the FMM correction. The FMM algorithm indiscriminately calculates all the pairwise interactions between the source and target points. When we use FMM to compute $\mathbf{v}_{\text{interaction}}$, we also compute the interaction between the points that belong to the same RBC. But, these interactions need to be evaluated as *local* interactions. Therefore, we need calculate and subtract these erroneous terms from $\mathbf{v}_{\text{interaction}}$. The direct Stokes kernel, is used to evaluate the correction.

The kernel for surface reparametrization. In a typical simulation, the RBCs go thorough great distortion and the quality of the grid on their surface diminishes very fast. In Figure 38, we give an example of such distortion. In Chapter 4, we proposed a *tangential correction* algorithm to compensate for the distortions and maintain the grid quality. The

reparametrization of the surface, involves calculation of the normal vector to the surface, mapping the surface to the spherical harmonics domain, filtering the high frequencies, and restricting the correction to the tangential direction on the surface (to keep the shape of the RBC intact). The complexity of this kernel is $\mathcal{O}(nq^3)$.

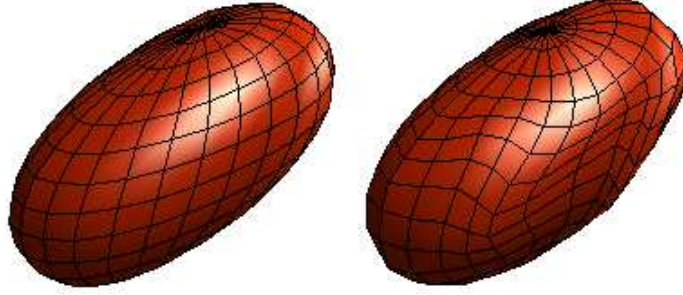


Figure 38: REPARAMETERIZATION OF RBC. *The comparison between the quality of the grid for two simulation with and without the reparametrization, i.e., the redistribution of points on surface of an RBC in order to improve the numerical stability of the time-stepping scheme. Without such reparametrization, the distribution of points on the surface becomes distorted as we march in time and the accuracy of the simulation is quickly lost.*

6.3 Scalability Results

In this section, we describe the results from numerical experiments we conducted to investigate the performance characteristics of MoBo and the parallel scalability of our implementation across different architectures. Below, we summarize the different aspects of our numerical tests.

Platforms and architectures. The large scale weak and strong scalability results have been obtained on the *Jaguar PF* platform at the National Center for Computational Sciences (UT/ORNL). Jaguar is a Cray XT5 system with 224,256 cores (2.6 GHz hex-core AMD Opteron, 2GB/core) and a 3D-torus topology. Jaguar was ranked first in the top-500 list of supercomputers (www.top500.org) as of July of 2010. The GPU scalability results have been obtained on TeraGrid’s *Lincoln* at the National Center for Supercomputing Applications (UIUC/NSF), a Dell cluster with NVIDIA Tesla S1070 accelerators, 1536 cores (Intel Harpertown/2.33 Ghz dual-socket quad-core 2GB/core), 384 GPUs (4GB/GPU), and InfiniBand (SDR) interconnect. The results on Fermi were obtained on a single node AMD machine at ORNL. The Nehalem tests were performed in an in-house 8-node cluster, with

16 sockets and one NVIDIA T10P-based GPU per socket. In all of the experiments on Jaguar, we use one MPI process per socket and six threads per socket. Both \mathbf{v}_{self} and $\mathbf{v}_{\text{interaction}}$ calculations have been multithreaded using OpenMP. Also, in all of our GPU experiments, we use one MPI process per socket.

Implementations and libraries. The code is written in C++ and the accelerator modules in CUDA. We use the PETSc [11] for profiling and certain parts of communication, and the DENDRO [116] package for the tree construction and repartitioning of the RBCs. The $\mathbf{v}_{\text{interaction}}$ module was implemented using our Kernel Independent Fast Multipole Method [77]. All of the kernels required for the calculation of \mathbf{v}_{self} were implemented from scratch. We used the native CRAY libsci and MKL BLAS libraries on the Jaguar and the Intel boxes respectively.

6.3.1 Single Node Experiments

To assess the performance of our code on a single node, we performed various tests for the \mathbf{v}_{self} and $\mathbf{v}_{\text{interaction}}$ calculations. The results are reported in Figure 39 for the \mathbf{v}_{self} evaluation, and Figure 40 for the $\mathbf{v}_{\text{interaction}}$ evaluation. Overall, we observe little difference between CPUs and GPUs although for higher resolutions, GPUs seem to outperform the x86 architectures. Recall that $\mathbf{v}_{\text{interaction}}$ and \mathbf{v}_{self} utilize both CPU and GPUs. For example, the performance of \mathbf{v}_{self} on a dual socket, dual GPU node exceeds 800 GFlops/s for $m = 12$. From Figure 40, we observe that the GPU accelerated version of FMM is roughly three times faster than the CPU-only thus, delivering a combined 60–70 GFlops/s per node for $\mathbf{v}_{\text{interaction}}$. The only data transfers between host and device is for the FMM evaluation in which the host collects the information from all RBCs and then invokes FMM. This is somewhat suboptimal. We are working on having both GPU and CPU versions for all phases of the FMM.

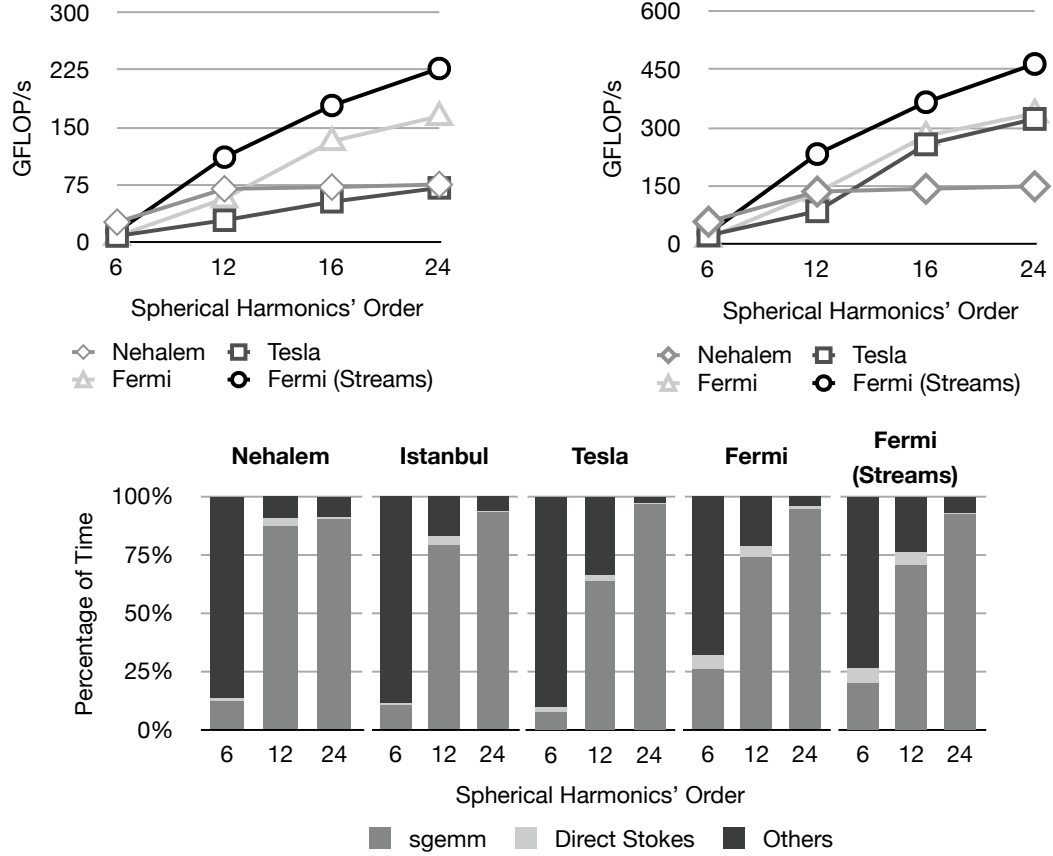


Figure 39: SINGLE NODE SCALING FOR LOCAL INTERACTIONS. *The first figure shows the sustained single and double precision FLOPS per second of the local kernel, on a Nehalem with eight OpenMP threads, a Tesla, and a Fermi. The second figure is the work share of the major components of the local kernel. For this figure, we used 8 OpenMP threads on Nehalem and 12 OpenMP threads on Istanbul.*

6.3.2 MPI, Strong Scalability Tests on Jaguar

The results are reported in Figure 41. The problem size is 300,000 RBCs with 84 points per RBC, which corresponds to 100,000,000 unknowns. The strong scalability results demonstrate excellent speed up resulting in an unprecedented five seconds per time-step on 24,576 cores.

We get excellent speed up and we require less than 10 seconds per time step for 300,000 RBCs. The efficiency, of course, is reduced for the largest processor count as the memory traffic dominates the computations.

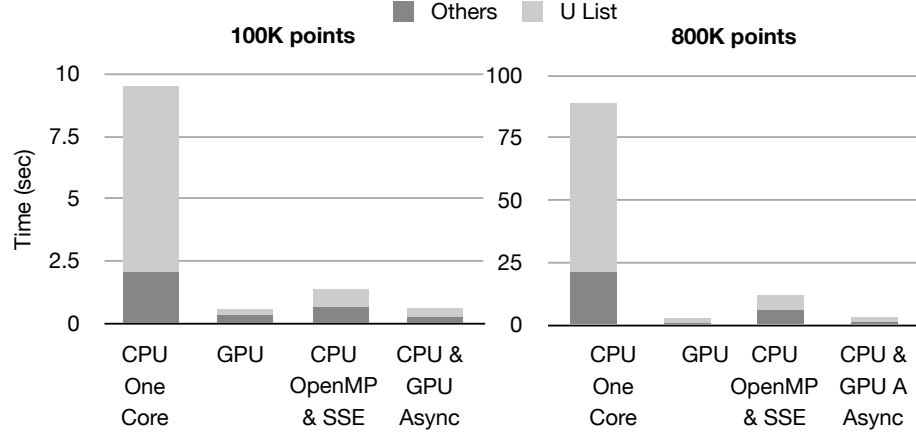


Figure 40: SINGLE NODE SCALING RESULTS FOR THE GLOBAL INTERACTIONS. *In this figure we present results for the FMM code for various hybrid architecture setups: (a) One thread CPU only without SSE, (b) GPU only for both direct and near evaluations, (c) Four threads on CPU with SSE, (d) Four threads on the CPU with SSE for the V-list and asynchronous evaluation of U-list on the GPU. On both Istanbul and Nehalem architectures we observe 1.2 ($m=6$)–1.7($m=12$) GFlops/s per core, for the overall FMM evaluation phase.*

p	48	384	3072	24576
Time (sec)	899.8	116.7	16.7	4.9
Efficiency	1.00	0.96	0.84	0.35

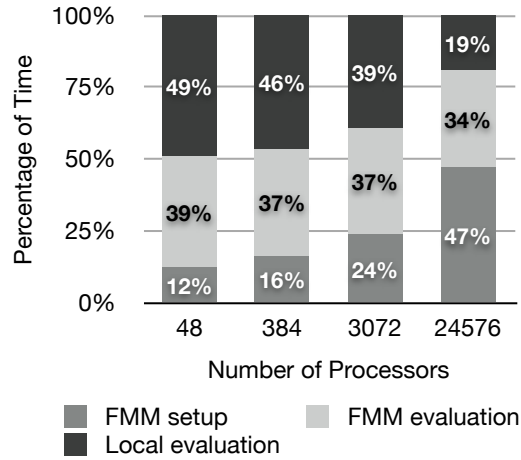


Figure 41: STRONG SCALING ON JAGUAR PF. *The strong scalability result for 262,144 vesicles, and total number of 22M grid points. There are 6 cores (and 6 OpenMP threads) per MPI process. The finest level of the octree is nine and the coarsest is three.*

6.3.3 MPI, Weak Scalability Tests on Jaguar

The results are reported in Figure 42. The problem size (number of RBCs) per core is kept fixed to 8000 RBCs, again with 84 points per RBC for the line-distribution on the Poiseuille

flow. We can observe the the calculation of \mathbf{v}_{self} remains almost constant, whereas the cost of the tree-setup and $\mathbf{v}_{\text{interaction}}$ increase. This is due to several reasons. As we increase the problem size, the tree gets deeper and the cost per core increases. Also for such non-uniform trees it is difficult to load balance for all phases of FMM. The solution to these scaling problems is to employ the hypercube-like tree-broadcast algorithm we developed in [77] for all of the phases of FMM. (Currently it is used only in the post-order communication phase of the evaluation phase.) Finally, the setup is not multithreaded; we are currently working on this, and we expect a factor of four on more speed-ups in this calculation. Despite, the need for further optimizations, we achieve good utilization of the resources: the \mathbf{v}_{self} phase sustains over 18 GFlops/s per core (single precision) and the $\mathbf{v}_{\text{interaction}}$ phase sustains over 1.2 GFlops/per core (double precision). Overall, the code exceeds 0.7 Petaflops of sustained performance.

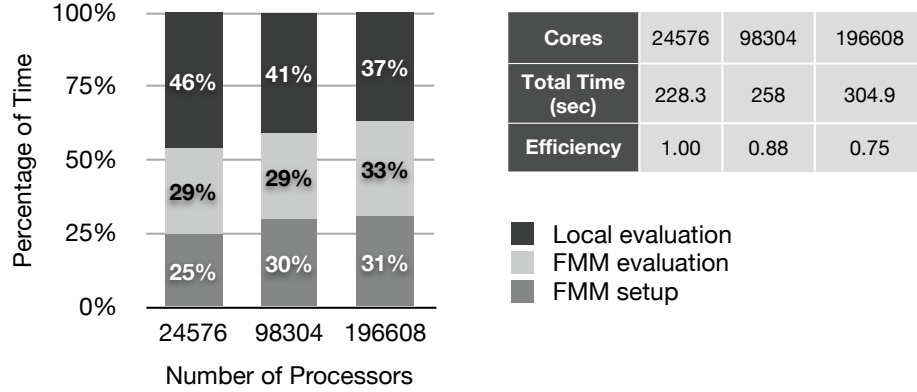


Figure 42: WEAK SCALING ON JAGUAR PF. *The weak scalability of the simulation to 196,608 cores. We have chosen a line distribution of 8,000 RBCs with 84 points per RBC. We use one MPI process per socket and all of the six OpenMP threads in each MPI process. The finest to coarsest octree levels range from 24 to 4. In the largest, simulation, there are 200 million red blood cells and 90 billion unknowns. These results represent the average timings of four explicit Euler time steps.*

6.3.4 GPU Weak Scalability Results for FMM on Lincoln

We report these results in Figure 43. We only report results for the uniform distribution using 1M points per GPU. We use one socket per MPI process and one GPU per socket. In this experiment we use one core per socket. The results on GPUs are excellent on up to 256 processes/GPUs. We get over a 25X per core consistently and we were able to evaluate

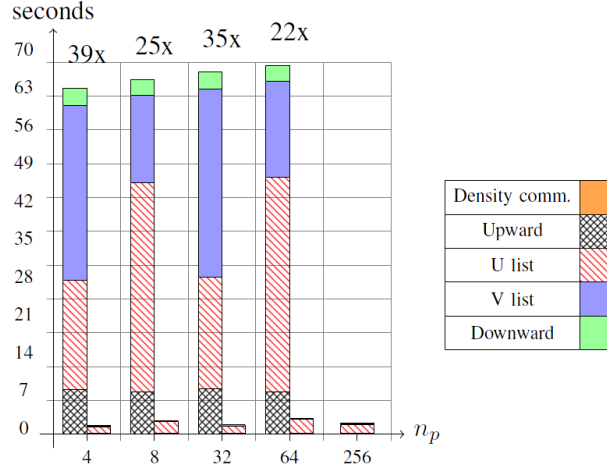


Figure 43: GPU WEAK SCALING. Here we compare CPU-only with GPU/CPU configuration on up to 256 processes. For the largest run the total evaluation on 256 million points takes 2.2 secs. Throughout the computation, we maintain a 25X speed-up over the single-core CPU version with only one thread per socket. When multithreading and vectorization is enabled, the differences become less pronounced as we can see in FFigure 40. For the GPU runs, we use a shallower tree by allowing a higher number of points per box. In this way, we favor dense interactions over far-field computations. The former has a favorable computation/memory communication ratio and performs favorably on a GPU. In this examples, we used roughly 400 points per box for the GPU runs, and 100 points per box for the CPU runs. Both numbers were optimized for their respective architectures. We were able to maintain a 1.8-3 secs / evaluation for the GPU-based implementation. (This figure is reproduced from [77].)

a 256-million particle sum in 2.3 seconds for a total of approximately 8 TFlops/s.

6.3.5 Red Blood Cell Distributions and Background flow

We test a line-like distribution of cells (Figure 44) on a Poiseuille background flow.³ The results of having a line of cells exposed to such a Poiseuille flow are easy to informally “verify” visually. Also, such a flow results in a highly non-uniform distributions of points as the simulation horizon increases.

³More precisely, this is a “pseudo-Poiseuille flow”, since we do not impose confinement boundary conditions around the cells. Rather, we impose a free-space velocity that corresponds to a Poiseuille flow. Roughly speaking, such background flow corresponds to an unperturbed laminar flow in a blood vessel.

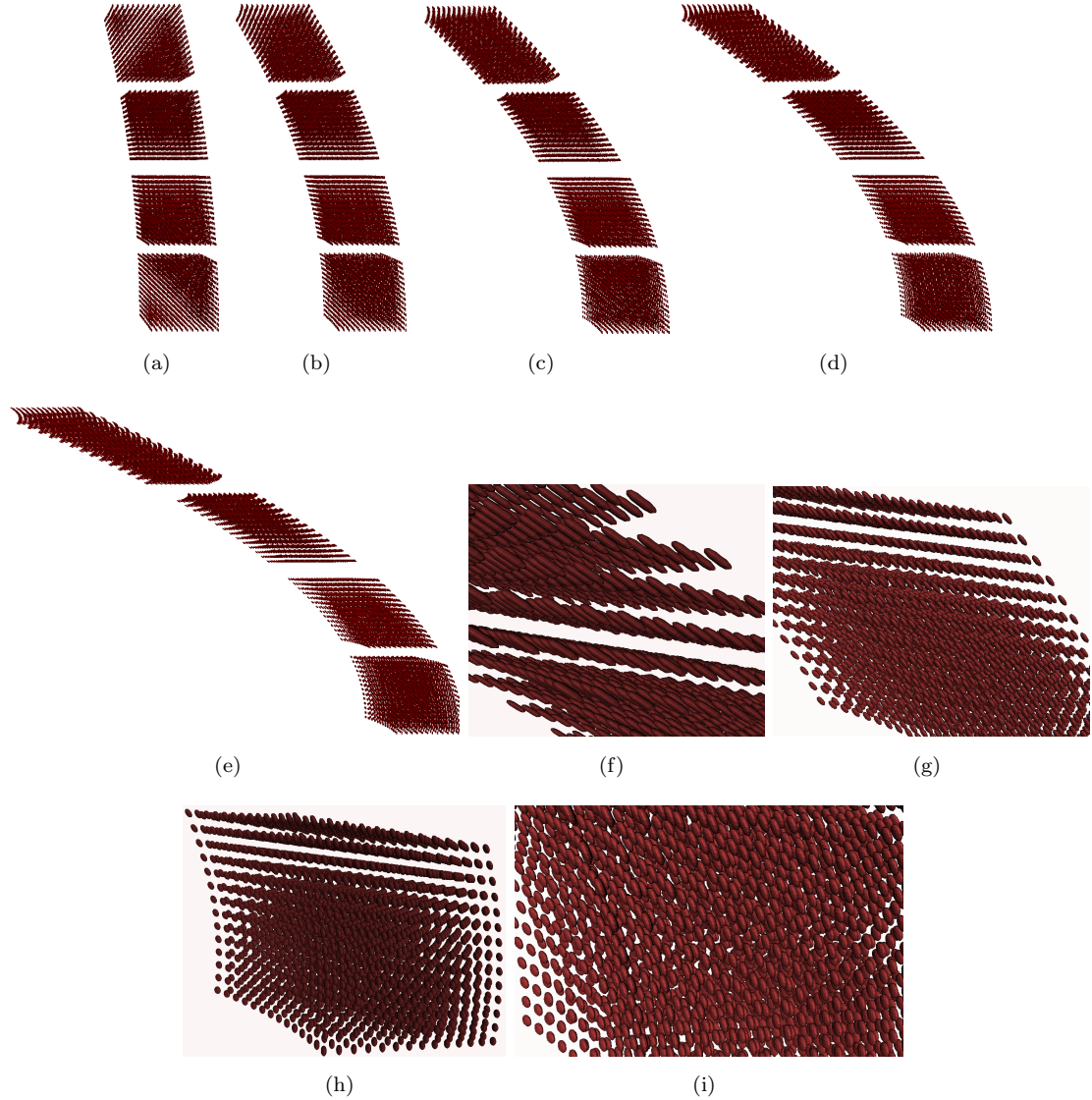


Figure 44: SNAPSHOTS OF THE SIMULATION OF 40,000 RBCs. *In this figure, we present results from a 40,000-RBC simulation with 84 points per RBC for a total 15,000,000 unknowns. In the top row (a–c), we can observe the alignment of the cells with the background Poiseuille flow as we advance in time. We can verify the need for non-uniform solvers. Every time step of this simulation, requires a Stokes solve at the extraordinarily complicated domain defined by the exterior and interior of the RBCs. In addition, the interfacial forces at each RBC are computed by inverting an integro-differential operator that involves fourth order derivatives. In the bottom row, we zoom in on different regions of the flow snapshot (c). We can observe the different deformations of the cells in different regions of the flow. For example, (f) and (g) depict cells from the upper tip (c) in which the shear rate is higher and the cells experience larger deformations. Subfigures (h) and (i) depict cells from the bottom left of (c), which is near the center of the Poiseuille flow and thus, the cells experience smaller deformations. The visualization was performed on a single workstation using ParaView (www.paraview.org).*

6.4 Conclusions

We have presented MoBo, a framework that enables large-scale direct simulations of blood microcirculation. MoBo exposes and exploits concurrency at all stages of a complex multi-physics, multiscale problem and uses several parallel programming paradigms. We showed that we can efficiently scale the different parts of the method and we observe good scalability across different architectures.

For the computation of \mathbf{v}_{self} , on a single node, we get roughly near peak performance for GEMM on both CPUs and GPUs. Our algorithmic choices were targeted to an extensive use of GEMM routines—without compromising overall work optimality. We are able to deliver spectral accuracy while using only a small number of degrees of freedom per RBC (e.g., compare to the 1000s of degrees of freedom per RBC for Lattice Boltzmann methods). For the global interaction, we have achieved 1 GFlop/s *per CPU core* for the overall FMM, which is quite remarkable given the complexity of the algorithm.

In our largest calculation on 196,608 cores, we achieved 0.7 petaflops for the multi-physics/multiscale problem of direct numerical simulation of blood flow. Let us emphasize that these results represent the worst case scenario with respect performance, as we use a very small number of points per cell. If we use an $m = 12$ spherical harmonics approximation for the RBCs the percentage of time spent in the \mathbf{v}_{self} part of the calculation will further increase.

Taken together, MoBo opens the way for blood flow simulations of unprecedented fidelity. It enables the simulation of microfluidic and nanofluidic devices.

CHAPTER VII

FUTURE RESEARCH DIRECTIONS

In Chapter 2 we presented a semi-implicit numerical scheme for the simulation of inextensible vesicles suspended in bounded or unbounded domains. Through numerical experiments, we have demonstrated that the proposed scheme does not exhibit stability constraint on the time-step size. The most significant limitation of this scheme is that the number of Fourier modes used to represent the vesicle membrane and the time step are not chosen adaptively. Our spectral discretization (which we combine with a special high-order scheme for singular integrals) in space results in discretization errors that are dominated by the time-stepping scheme. For concentrated suspensions, adaptive schemes combined with a posteriori estimates may be necessary. Furthermore, we solve the discretized system of equations using the Generalized Minimum Residual Method (GMRES) with an appropriate set of preconditioners, which are based on the spectral properties of the operators. Nonetheless, for very small viscosity contrasts, the spectral properties of the operators change and a generic preconditioner, as we used here, fails to fully compensate for the poor conditioning of the operators. An additional extension of this work would be to design an algorithm that allows decoupling of the time-step size from the shear rate. We believe, however, that such an algorithm would require the use of nonlinear solvers and contact detection methods that fully couple the vesicle position updates. Such coupling would be more difficult to implement and analyze.

In the study of vesicle migration in Chapter 3, several interesting questions remain unanswered. An interesting line of research would be to study the spontaneous organization of vesicles for higher concentrations, and providing a detailed link between microstructure and rheology. There are two dynamical factor contributing to the self organization of vesicles: (1) the existence of a stable orbit (where the inward migration force balances the wall lift force) and (2) the interaction of vesicles within such an orbit, which results in

the uniform distribution of vesicles. In Chapter 3 we demonstrated the coupling between deformability and the inward migration. The effect of deformability of particles on the second step is yet to be investigated.

For the flow of vesicles in three dimensions (Chapters 4, 5, and 6), we restricted our attention to suspensions of vesicles in unbounded domains. The inertial terms were ignored and therefore the overall method is restricted to low Reynolds numbers. We only considered spherical-topology vesicles and we did not allow for topological changes, which are present in many biophysical phenomena involving vesicles. Similar to the two dimensional case, an important limitation of our scheme is the lack of adaptivity (both in space and time). This lack of adaptivity can cause vesicle-vesicle collisions, which are not possible in the mathematical model we use. Indeed, one can easily construct simulations where, without a significant increase of the surface discretization size, our code fails to resolve inter-vesicle interactions accurately. One other important research direction is the development of efficient algorithms for the evaluation of nearly singular integrals, which arise in the close interaction of vesicles.

REFERENCES

- [1] G. Agresar, J. Linderman, G. Tryggvason, and K. G. Powell. An adaptive, cartesian, front-tracking method for the motion, deformation and adhesion of circulating cells. *Journal Of Computational Physics*, 143(2):346–380, 1998.
- [2] B. K. Alpert. Hybrid Gauss-trapezoidal quadrature rules. *SIAM Journal on Scientific Computing*, 20(5):1551–1584, 1999.
- [3] M. V. Apostolakis, V. G. Mavrantzas, and A. N. Beris. Stress gradient-induced migration effects in the taylorcouette flow of a dilute polymer solution. *Journal of Non-Newtonian Fluid Mechanics*, 102(2):409–445, 2002.
- [4] U. M. Ascher and L. R. Petzold. *Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 1998.
- [5] U. M. Ascher, S. J. Ruuth, and B. T. R. Wetton. Implicit-explicit methods for time-dependent partial differential equations. *SIAM Journal on Numerical Analysis*, 32(3):797–823, 1995.
- [6] K. E. Atkinson. The numerical solution of laplace’s equation in three dimensions. *SIAM Journal on Numerical Analysis*, 19(2):263–274, 1982.
- [7] K. E. Atkinson. *The Numerical Solution of Integral Equations of the Second Kind*. Cambridge University Press, 1997.
- [8] P. Bagchi. Mesoscale simulation of blood flow in small vessels. *Biophysical Journal*, 92(6):1858–1877, 2007.
- [9] P. Bagchi and R. M. Kalluri. Dynamics of nonspherical capsules in shear flow. *Physical Review E*, 80(1), 2009.
- [10] P. Bagchi and R. M. Kalluri. Dynamic rheology of a dilute suspension of elastic capsules: effect of capsule tank-treading, swinging and tumbling. *Journal of Fluid Mechanics*, 669:498–526, 2011.
- [11] S. Balay, K. Buschelman, W. D. Gropp, D. Kaushik, M. Knepley, L. C. McInnes, B. F. Smith, and H. Zhang. PETSc home page, 2001. <http://www.mcs.anl.gov/petsc>.
- [12] G. K. Batchelor. *An introduction in fluid dynamics*. Cambridge University Press, 1970.
- [13] G. K. Batchelor. The stress system in a suspension of force-free particles. *Journal of Fluid Mechanics*, 41:545–570, 1970.
- [14] G. K. Batchelor and J. T. Green. The hydrodynamic interaction of two small freely-moving spheres in a linear flow field. *Journal of Fluid Mechanics*, 56(02):375–400, 1972.
- [15] J. T. Beale, T. Y. Hou, and J. Lowengrub. Convergence of a boundary integral method for water waves. *SIAM Journal on Numerical Analysis*, 33(5):1797–1843, 1996.

- [16] J. Beaucourt, F. Rioual, T. Séon, T. Biben, and C. Misbah. Steady to unsteady dynamics of a vesicle in a flow. *Physical Review E*, 69(1):011906, Jan 2004.
- [17] L. E. Becker and M. J. Shelley. Instability of elastic filaments in shear flow yields first-normal-stress differences. *Physical Review Letters*, 8719(19), 2001.
- [18] T. Biben and C. Misbah. An advected-field method for deformable entities under flow. *European Physical Journal B*, 29(2):311–316, 2002.
- [19] R. B. Bird, R. C. Armstrong, and O. Hassager. *Dynamics of Polymeric Liquids, Vol. 1, Fluid Dynamics*. Wiley, New York, second edition, 1987.
- [20] G. Biros, L. Ying, and D. Zorin. A fast solver for the Stokes equations with distributed forces in complex geometries. *Journal of Computational Physics*, 193(1):317–348, 2003.
- [21] G. Biros, L. Ying, and D. Zorin. A fast solver for the Stokes equations with distributed forces in complex geometries. *Journal of Computational Physics*, 194(1):317–348, 2004.
- [22] M. I. G. Bloor and M. J. Wilson. Method for efficient shape parametrization of fluid membranes and vesicles. *Physical Review E*, 61(4):4218–4229, 2000.
- [23] F. Bornemann. Accuracy and stability of computing high-order derivatives of analytic functions by cauchy integrals. *Foundations of Computational Mathematics*, 11:1–63, 2011.
- [24] J. P. Boyd. *Chebyshev and Fourier Spectral Methods*. 1999.
- [25] G. Breyiannis and C. Pozrikidis. Simple shear flow of suspensions of elastic capsule. *Theoretical and Computational Fluid Dynamics*, 13(5):327–347, February 2000.
- [26] P. O. Brunn. Nonuniform concentration profiles of dilute macromolecular solutions in rotational viscometric flows. *The Journal of Chemical Physics*, 80(7):3420–3426, 1984.
- [27] O. P. Bruno and L. A. Kunyansky. A fast, high-order algorithm for the solution of surface scattering problems: basic implementation, tests, and applications. *Journal of Computational Physics*, 169(1):80–110, 2001.
- [28] F. Campelo and A. Hernandez-Machado. Dynamic model and stationary shapes of fluid vesicles. *The European Physical Journal E: Soft Matter and Biological Physics*, 20:37, 2006.
- [29] I. Cantat and C. Misbah. Lift force and dynamical unbinding of adhering vesicles under shear flow. *Physical Review Letters*, 83(4):880–883, Jul 1999.
- [30] I. Cantat, K. Kassner, and C. Misbah. Vesicles in haptotaxis with hydrodynamical dissipation. *The European Physical Journal E: Soft Matter and Biological Physics*, 10:175–189, 2003.
- [31] P. C.-H. Chan and L. Leal. An experimental study of drop migration in shear flow between concentric cylinders. *International Journal of Multiphase Flow*, 7(1):83–99, 1981.

- [32] P. C.-H. Chan and L. G. Leal. The motion of a deformable drop in a second-order fluid. *Journal of Fluid Mechanics*, 92(01):131–170, 1979.
- [33] A. Chandramowlishwaran, S. Williams, L. Olikar, I. Lashuk, G. Biros, and R. Vuduc. Optimizing and tuning the fast multipole method for state-of-the-art multicore architectures. In *Proceedings of IPDPS*, Atlanta, GA, 2010. IEEE Computer Society.
- [34] B. Cichocki, R. B. Jones, R. Kutteh, and E. Wajnryb. Friction and mobility for colloidal spheres in stokes flow near a boundary: The multipole method and applications. *The Journal of Chemical Physics*, 112(5):2548–2561, 2000.
- [35] J. Clausen, D. Reasor Jr, and C. Aidun. Parallel performance of a lattice-boltzmann/finite element cellular blood flow solver on the ibm blue gene/p architecture. *Computer Physics Communications*, 2010.
- [36] P. Coussot, J. S. Raynaud, F. Bertrand, P. Moucheron, J. P. Guilbaud, H. T. Huynh, S. Jarny, and D. Lesueur. Coexistence of liquid and solid phases in flowing soft-glassy materials. *Physical Review Letters*, 88(21):218301, May 2002.
- [37] V. Cristini, J. Blawdziewicz, and M. Loewenberg. An adaptive mesh algorithm for evolving surfaces: simulations of drop breakup and coalescence. *Journal of Computational Physics*, 168(2):445–463, 2001.
- [38] G. Danker, C. Verdier, and C. Misbah. Rheology and dynamics of vesicle suspension in comparison with droplet emulsion. *Journal of Non-Newtonian Fluid Mechanics*, 152(1-3):156–167, 2008. 4th International workshop on Nonequilibrium Thermodynamics and Complex Fluids.
- [39] P. Dimitrakopoulos. Interfacial dynamics in Stokes flow via a three-dimensional fully-implicit interfacial spectral boundary element algorithm. *Journal of Computational Physics*, 225(1):408–426, 2007.
- [40] S. K. Doddi and P. Bagchi. Effect of inertia on the hydrodynamic interaction between two liquid capsules in simple shear flow. *International Journal of Multiphase Flow*, 34(4):375–392, 2008.
- [41] W. S. Don and A. Solomonoff. Accuracy enhancement for higher derivatives using Chebyshev collocation and a mapping technique. *SIAM Journal on Scientific Computing*, 18(4):1040–1055, 1997.
- [42] J. B. Drake, P. Worley, and E. D’Azevedo. Algorithm 888: Spherical harmonic transform algorithms. *ACM Trans. Math. Softw.*, 35(3):1–23, 2008.
- [43] Q. Du, C. Liu, and X. Wang. A phase field approach in the numerical study of the elastic bending energy for vesicle membranes. *Journal of Computational Physics*, 198(2):450–468, 2004.
- [44] Q. Du, C. Liu, and X. Wang. Simulating the deformation of vesicle membranes under elastic bending energy in three dimensions. *Journal of Computational Physics*, 212(2):757–777, 2006.

- [45] G. Dumas and A. Leonard. A divergence-free spectral expansions method for three-dimensional flows in spherical-gap geometries. *Journal of Computational Physics*, 111(2):205–219, 1994.
- [46] W. Dzwinel, K. Boryczko, and D. A. Yuen. A discrete-particle model of blood dynamics in capillary vessels. *Journal Of Colloid And Interface Science*, 258(1):163–173, 2003.
- [47] E. C. Eckstein, D. G. Bailey, and A. H. Shapiro. Self-diffusion of particles in shear flow of a suspension. *Journal of Fluid Mechanics*, 79:191–208, 1977.
- [48] F. Feng and W. S. Klug. Finite element modeling of lipid bilayer membranes. *Journal of Computational Physics*, 220(1):394–408, 2006.
- [49] A. L. Fogelson and R. D. Guy. Platelet-wall interactions in continuum models of platelet thrombosis: formulation and numerical solution. *Mathematical Medicine And Biology-A Journal Of the IMA*, 21(4):293–334, 2004.
- [50] J. Freund. Leukocyte margination in a model microvessel. *Physics of Fluids*, 19:023301, 2007.
- [51] M. Ganesh and I. G. Graham. A high-order algorithm for obstacle scattering in three dimensions. *Journal of Computational Physics*, 198(1):211–242, 2004.
- [52] G. Ghigliotti, T. Biben, and C. Misbah. Rheology of a dilute two-dimensional suspension of vesicles. *Journal of Fluid Mechanics*, 653:489–518, Apr. 2010.
- [53] G. Ghigliotti, A. Rahimian, G. Biro, and C. Misbah. Vesicle migration and spatial organization driven by flow line curvature. *Physical Review Letters*, 106(2):028101, Jan 2011.
- [54] Z. Gimbutas and L. Greengard. Short note: A fast and stable method for rotating spherical harmonic expansions. *J. Comput. Phys.*, 228(16):5621–5627, 2009.
- [55] E. Givberg and K. Yelick. Distributed immersed boundary simulation in titanium. *SIAM Journal On Scientific Computing*, 28(4):1361–1378, 2006.
- [56] R. Glowinski, T. Pan, T. Hesla, D. Joseph, and J. P riaux. A fictitious domain approach to the direct numerical simulation of incompressible viscous flow past moving rigid bodies: Application to particulate flow. *Journal of Computational Physics*, 169:363–426, 2001.
- [57] H. L. Goldsmith and R. Skalak. Hemodynamics. *Annual Reviews of Fluid Mechanics*, 7:213–247, 1975.
- [58] I. G. Graham and I. H. Sloan. Fully discrete spectral boundary integral methods for helmholtz problems on smooth closed surfaces in \mathbb{R}^3 . *Numerische Mathematik*, 92, 2002.
- [59] J. A. Hanna and P. M. Vlahovska. Surfactant-induced migration of a spherical drop in stokes flow. *Physics of Fluids*, 22(1):013102, Jan. 2010.
- [60] E. L. Hill. The theory of vector spherical harmonics. *American Journal of Physics*, 22(4):211–214, 1954.

- [61] K. Hollingsworth and M. Johns. Droplet migration in emulsion systems measured using mr methods. *Journal of Colloid and Interface Science*, 296(2):700–709, 2006.
- [62] D. J. Jeffrey and A. Acrivos. The rheological properties of suspensions of rigid particles. *AIChE Journal*, 22(3):417–432, 1976.
- [63] X. Jiao, A. Colombi, X. Ni, and J. Hart. Anisotropic mesh adaptation for evolving triangulated surfaces. In *Proceedings of the 15th International Meshing Roundtable*, volume 51, page 173. Springer Verlag, 2006.
- [64] V. Kantsler and V. Steinberg. Orientation and dynamics of a vesicle in tank-treading motion in shear flow. *Physical Review Letters*, 95(25), 2005.
- [65] B. Kaoui, G. H. Ristow, I. Cantat, C. Misbah, and W. Zimmermann. Lateral migration of a two-dimensional vesicle in unbounded poiseuille flow. *Physical Review E*, 77(2, Part 1), 2008.
- [66] B. Kaoui, G. Biros, and C. Misbah. Why do red blood cells have asymmetric shapes even in a symmetric flow? *Physical Review Letters*, 103(18):188101, 2009.
- [67] B. Kaoui, A. Farutin, and C. Misbah. Vesicles under simple shear flow: Elucidating the role of relevant control parameters. *Phys. Rev. E*, 80:061905, Dec 2009.
- [68] S. J. Karrila and S. Kim. Integral equations of the second kind for stokes flow: Direct solution for physical variables and removal of inherent accuracy limitations. *Chemical engineering communications*, 82:123–161, 1989.
- [69] S. R. Keller and R. Skalak. Motion of a tank-treading ellipsoidal particle in a shear flow. *Journal of Fluid Mechanics*, 120:27–47, 1982.
- [70] C. T. Kelley and D. E. Keyes. Convergence analysis of pseudo-transient continuation. *SIAM Journal on Numerical Analysis*, 35:508–523, 1998.
- [71] M. Kennedy, C. Pozrikidis, and R. Skalak. Motion and deformation of liquid drops, and the rheology of dilute emulsions in simple shear flow. *Computers & Fluids*, 23(2): 251–278, 1994.
- [72] M. R. King and D. A. Hammer. Multiparticle adhesive dynamics: Hydrodynamic recruitment of rolling leukocytes. *PNAS*, 98(26):14919–14924, 2001.
- [73] M. Kraus, W. Wintz, U. Seifert, and R. Lipowsky. Fluid vesicles in shear flow. *Physical Review Letters*, 77(17):3685–3688, 1996.
- [74] M. C. A. Kropinski. An efficient numerical method for studying interfacial motion in two-dimensional creeping flows. *Journal of Computational Physics*, 171(2):479–508, 2001.
- [75] S. Kwak and C. Pozrikidis. Adaptive triangulation of evolving, closed, or open surfaces by the advancing-front method. *Journal of Computational Physics*, 145(1):61–88, 1998.
- [76] E. Lac, A. Morel, and D. Barth-Biesel. Hydrodynamic interaction between two identical capsules in simple shear flow. *Journal of Fluid Mechanics*, 573(-1):149–169, 2007.

- [77] I. Lashuk, A. Chandramowlishwaran, H. Langston, T. Nguyen, R. Sampath, A. Shringarpure, R. Vuduc, D. L. Ying, and G. Biros. A massively parallel adaptive fast-multipole method on heterogeneous architectures. In *SC '09: Proceedings of the 2009 ACM/IEEE conference on Supercomputing*, pages 1–12, Piscataway, NJ, USA, 2009. IEEE Press.
- [78] J. Lee and N. P. Smith. Theoretical modeling in hemodynamics of microcirculation. *Microcirculation*, 15:699–714, 2008.
- [79] Y. L. Liu and W. K. Liu. Rheology of red blood cell aggregation by computer simulation. *Journal Of Computational Physics*, 220(1):139–154, 2006.
- [80] S. H. Lo. A new mesh generation scheme for arbitrary planar domains. *International Journal for Numerical Methods in Engineering*, 21(8):1403–1426, 1985.
- [81] M. Loewenberg. Numerical simulation of concentrated emulsion flows. *Journal of Fluids Engineering-Transactions of the ASME*, 120(4):824–832, 1998.
- [82] M. Loewenberg and E. Hinch. Numerical simulations of concentrated emulsions. *Journal of Fluid Mechanics*, 321:395–419, 1996.
- [83] M. Loewenberg and E. J. Hinch. Collision of two deformable drops in shear flow. *Journal of Fluid Mechanics*, 338:299–315, 1997.
- [84] R. Lohner. Regridding surface triangulations. *Journal of Computational Physics*, 126(1):1–10, 1996.
- [85] L. Ma and W. S. Klug. Viscous regularization and r-adaptive remeshing for finite element analysis of lipid membrane mechanics. *Journal of Computational Physics*, 227(11):5816–5835, 2008.
- [86] P. Martinsson and V. Rokhlin. A fast direct solver for boundary integral equations in two dimensions. *Journal of Computational Physics*, 205(1):1–23, 2005.
- [87] D. M. McQueen and C. S. Peskin. Shared-memory parallel vector implementation of the immersed boundary method for the computation of blood flow in the beating mammalian heart. *Journal Of Supercomputing*, 11(3):213–236, 1997.
- [88] C. Misbah. Vacillating breathing and tumbling of vesicles under shear flow. *Physical Review Letters*, 96(2), 2006.
- [89] M. J. Mohlenkamp. A fast transform for spherical harmonics. *Journal of Fourier analysis and applications*, 5(2):159–184, 1999.
- [90] J.-C. Nédélec. *Acoustic and Electromagnetic Equations*. Springer-Verlag, New York, 2000.
- [91] J. Nocedal and S. J. Wright. *Numerical optimization*. Springer series in operations research. Second edition, 2006.
- [92] H. Noguchi and D. G. Gompper. Shape transitions of fluid vesicles and red blood cells in capillary flows. *Proceedings Of The National Academy Of Sciences Of The United States Of America*, 102:14159–14164, 2005.

- [93] R. Nourgaliev, T. Dinh, T. Theofanous, and D. Joseph. The lattice boltzmann equation method: theoretical interpretation, numerics and implications. *International Journal of Multiphase Flow*, 29(1):117–169, 2003.
- [94] S. A. Orszag. Numerical simulation of incompressible flows within simple boundaries: accuracy. *Journal of Fluid Mechanics*, 49:75–112, 1971.
- [95] S. A. Orszag. Fourier series on spheres. *Monthly Weather Review*, 102:56–75, 1974.
- [96] N. Phan-Thien, K. Y. Lee, and D. Tullock. Large scale simulation of suspensions with PVM. In *Proceedings of SC97*, The SCxy Conference series, San Jose, CA, November 1997. ACM/IEEE.
- [97] I. V. Pivkin, P. D. Richardson, and G. Karniadakis. Blood flow velocity effects and role of activation delay time on growth and form of platelet thrombi. *Proceedings Of The National Academy Of Sciences Of The United States Of America*, 103(46):17164–17169, 2006.
- [98] A. S. Popel and P. C. Johnson. Microcirculation and hemorheology. *Annual Review of Fluid Mechanics*, 37(1):43–69, 2005.
- [99] H. Power. The completed double layer boundary integral equation method for two-dimensional stokes flow. *IMA Journal of Applied Mathematics*, 51(2):123–145, 1993.
- [100] H. Power and G. Miranda. Second kind integral equation formulation of stokes’ flows past a particle of arbitrary shape. *SIAM Journal of Applied Mathematics*, 47(4):689–698, 1987.
- [101] C. Pozrikidis. The axisymmetric deformation of a red blood cell in uniaxial straining stokes flow. *Journal of Fluid Mechanics*, 216:231–254, 1990.
- [102] C. Pozrikidis. *Boundary Integral and Singularity Methods for Linearized Viscous Flow*. Cambridge University Press, Cambridge, 1992.
- [103] C. Pozrikidis. Finite deformation of liquid capsules enclosed by elastic membranes in simple shear flow. *Journal of Fluid Mechanics*, 297:123–152, 1995.
- [104] C. Pozrikidis. Interfacial dynamics for Stokes flow. *Journal of Computational Physics*, 169:250–301, 2001.
- [105] C. Pozrikidis. Effect of membrane bending stiffness on the deformation of capsules in simple shear flow. *Journal of Fluid Mechanics*, 440:269–291, 2001.
- [106] C. Pozrikidis. Numerical simulation of the flow-induced deformation of red blood cells. *Annals of Biomedical Engineering*, 31(10):1194–1205, 2003.
- [107] C. Pozrikidis. Axisymmetric motion of a file of red blood cells through capillaries. *Physics of Fluids*, 17(3):14, 2005.
- [108] C. Pozrikidis. A spectral collocation method with triangular boundary elements. *Engineering Analysis with Boundary Elements*, 30(4):315 – 324, 2006.

- [109] A. Rahimian, I. Lashuk, S. Veerapaneni, A. Chandramowlishwaran, D. Malhotra, L. Moon, R. Sampath, A. Shringarpure, J. Vetter, R. Vuduc, D. Zorin, and G. Biros. Petascale direct numerical simulation of blood flow on 200K cores and heterogeneous architectures. In *Proceedings of the 2010 ACM/IEEE International Conference for High Performance Computing, Networking, Storage and Analysis*, SC '10, pages 1–11, Washington, DC, USA, 2010. IEEE Computer Society.
- [110] A. Rahimian, S. K. Veerapaneni, and G. Biros. Dynamic simulation of locally inextensible vesicles suspended in an arbitrary two-dimensional domain, a boundary integral method. *Journal of Computational Physics*, 229(18):6466–6484, 2010.
- [111] A. Rahimian, S. K. Veerapaneni, D. Zorin, and G. Biros. Three-dimensional boundary integral method for the flow of vesicles with viscosity contrast. 2012. In preparation.
- [112] J. Rallison and A. Acrivos. A numerical study of the deformation and burst of a viscous drop in an extensional flow. *Journal of Fluid Mechanics*, 89:191–200, 1978.
- [113] S. Ramanujan and C. Pozrikidis. Deformation of liquid capsules enclosed by elastic membranes in simple shear flow: large deformations and the effect of fluid viscosities. *Journal of Fluid Mechanics*, 361:117–143, 1998.
- [114] Y. Saad. *Iterative Methods for Sparse Linear Systems*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, second edition, 2003.
- [115] E. Sackmann. Supported membranes: Scientific and practical applications. *Science*, 271:43–48, 1996.
- [116] R. Sampath, S. S. Adavani, H. Sundar, I. Lashuk, and G. Biros. DENDRO home page, 2008. URL <http://www.cc.gatech.edu/csela>.
- [117] R. S. Sampath, S. S. Adavani, H. Sundar, I. Lashuk, and G. Biros. Dendro: parallel algorithms for multigrid and AMR methods on 2:1 balanced octrees. In *SC '08: Proceedings of the 2008 ACM/IEEE conference on Supercomputing*, pages 1–12, Piscataway, NJ, USA, 2008. IEEE Press.
- [118] R. Schmitz and B. U. Felderhof. Creeping flow about a spherical particle. *Physica A: Statistical and Theoretical Physics*, 113(1-2):90–102, 1982.
- [119] T. Secomb, B. Styp-Rekowska, and A. Pries. Two-dimensional simulation of red blood cell deformation and lateral migration in microvessels. *Annals of Biomedical Engineering*, 35:755–765, 2007.
- [120] U. Seifert. Configurations of fluid membranes and vesicles. *Advances in Physics*, 46: 13–137, 1997.
- [121] U. Seifert. *Fluid Vesicles*. Institute of Solid State Research, Forschungszentrum Jülich, 2004.
- [122] U. Seifert, K. Berndl, and R. Lipowsky. Shape transformations of vesicles: Phase diagram for spontaneous- curvature and bilayer-coupling models. *Physical Review A*, 44(2):1182–1202, 1991.

- [123] R. H. Shafer. Radial migration of dna molecules in cylindrical flow: Ii. the non-draining model and possible application to fractionation. *Biophysical Chemistry*, 2(2):185–188, 1974.
- [124] K. M. Singh and J. J. R. Williams. A parallel fictitious domain multigrid preconditioner for the solution of poisson’s equation in complex geometries. *Computer Methods In Applied Mechanics And Engineering*, 194(45-47):4845–4860, 2005.
- [125] S. Sukumaran and U. Seifert. Influence of shear flow on vesicles near a wall: A numerical study. *Physical Review E*, 64(1), 2001.
- [126] C. H. Sun and L. L. Munn. Particulate nature of blood determines macroscopic rheology: A 2-d lattice boltzmann analysis. *Biophysical Journal*, 88(3):1635–1645, 2005.
- [127] H. Sundar, R. Sampath, C. Davatzikos, and G. Biros. Low-constant parallel algorithms for finite element simulations using linear octrees. In *Proceedings of SC2007*, The SCxy Conference series in high performance networking and computing, Reno, Nevada, 2007. ACM/IEEE.
- [128] T. E. Tezduyar and A. Sameh. Parallel finite element computations in fluid mechanics. *Computer Methods In Applied Mechanics And Engineering*, 195(13-16):1872–1884, 2006.
- [129] A. Tornberg and M. J. Shelley. Simulating the dynamics and interactions of flexible fibers in stokes flows. *Journal of Computational Physics*, 196(1):8–40, 2004.
- [130] L. N. Trefethen. *Spectral Methods in Matlab*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2000.
- [131] G. Tryggvason, B. Bunner, A. Esmaeeli, D. Juric, N. Al-Rawahi, W. Tauber, J. Han, S. Nas, and Y. Jan. A front-tracking method for the computations of multiphase flow. *Journal of Computational Physics*, 169(2):708–759, 2001.
- [132] G. Tryggvason, B. Bunner, A. Esmaeeli, and N. Al-Rawahi. Computations of multiphase flows. In *Advances In Applied Mechanics*, volume 39 of *Advances In Applied Mechanics*, pages 81–120. Academic Press Inc, San Diego, 2003.
- [133] O. B. Usta, J. E. Butler, and A. J. C. Ladd. Transverse migration of a confined polymer driven by an external force. *Physical Review Letters*, 98(9):098301, Feb 2007.
- [134] S. K. Veerapaneni, D. Gueyffier, G. Biros, and D. Zorin. A numerical method for simulating the dynamics of 3D axisymmetric vesicles suspended in viscous flows. *Journal of Computational Physics*, 228(19):7233–7249, 2009.
- [135] S. K. Veerapaneni, D. Gueyffier, D. Zorin, and G. Biros. A boundary integral method for simulating the dynamics of inextensible vesicles suspended in a viscous fluid in 2D. *Journal of Computational Physics*, 228(7):2334–2353, 2009.
- [136] S. K. Veerapaneni, A. Rahimian, G. Biros, and D. Zorin. A fast algorithm for simulating vesicle flows in three dimensions. *Journal of Computational Physics*, 230(14):5610–5634, 2011.

- [137] V. Vitkova, M. Mader, B. Polack, C. Misbah, and T. Podgorski. Micro-macro link in rheology of erythrocyte and vesicle suspensions. *Biophysical Journal*, 95(6):L33–L35, 2008.
- [138] P. M. Vlahovska, M. Loewenberg, and J. Blawdziewicz. Deformation of a surfactant-covered drop in a linear flow. *Physics of Fluids*, 17(10), 2005.
- [139] E. J. Weinberg. Monopole vector spherical harmonics. *Physical Review D*, D49:1086–1092, 1994.
- [140] J. L. Weiner. On a problem of Chen, Willmore, et al. *Indiana University Mathematics Journal*, 27:19–35, 1978.
- [141] S. Wu. Order-disorder transitions in the extrusion of fiber-filled poly (ethylene terephthalate) and blends. *Polymer Engineering & Science*, 19(9):638–650, 1979.
- [142] L. Ying, G. Biros, H. Langston, and D. Zorin. KIFMM3D: The kernel-independent fast multipole (FMM) 3D code. GPL license.
- [143] L. Ying, G. Biros, D. Zorin, and H. Langston. A new parallel kernel-independent fast multiple algorithm. In *Proceedings of SC03*, The SCxy Conference series, Phoenix, Arizona, 2003. ACM/IEEE.
- [144] L. Ying, G. Biros, and D. Zorin. A kernel-independent adaptive fast multipole method in two and three dimensions. *Journal of Computational Physics*, 196(2):591–626, 2004.
- [145] L. Ying, G. Biros, and D. Zorin. A high-order 3d boundary integral equation solver for elliptic pdes in smooth domains. *Journal of Computational Physics*, 219(1):247–275, 2006.
- [146] G. K. Youngren and A. Acrivos. Stokes flow past a particle of arbitrary shape: a numerical method of solution. *Journal of Fluid Mechanics*, 69:377–403, 1975.
- [147] L. T. Zhang, G. J. Wagner, and W. K. Liu. A parallelized meshfree method with boundary enrichment for large-scale cfd. *Journal Of Computational Physics*, 176(2):483–506, 2002.
- [148] H. Zhao, A. Isfahani, L. Olson, and J. Freund. A spectral boundary integral method for flowing blood cells. *Journal of Computational Physics*, 2010.
- [149] H. Zhou and C. Pozrikidis. The flow of suspensions in channels: Single files of drops. *Physics of Fluids A: Fluid Dynamics*, 5(2):311–324, 1993.
- [150] H. Zhou and C. Pozrikidis. Deformation of liquid capsules with incompressible interfaces in simple shear flow. *Journal of Fluid Mechanics*, 283:175–200, 1995.
- [151] A. Zinchenko and R. Davis. An efficient algorithm for hydrodynamical interaction of many deformable drops. *Journal of Computational Physics*, 157(2):539–587, 2000.
- [152] A. Zinchenko and R. Davis. Shear flow of highly concentrated emulsions of deformable drops by numerical simulations. *Journal of Fluid Mechanics*, 455:21–61, 2002.
- [153] A. Zinchenko and R. Davis. A boundary-integral study of a drop squeezing through interparticle constrictions. *Journal of Fluid Mechanics*, 564:227–266, 2006.

- [154] A. Zinchenko and R. Davis. Algorithm for direct numerical simulation of emulsion flow through a granular material. *Journal of Computational Physics*, 227(16):7841–7888, 2008.
- [155] A. Zinchenko and R. H. Davis. Large-scale simulations of concentrated emulsion flows. *Philosophical Transactions Of The Royal Society Of London Series A-Mathematical Physical And Engineering Sciences*, 361(1806):813–845, 2003.
- [156] A. Zinchenko, M. Rother, and R. Davis. A novel boundary-integral algorithm for viscous interaction of deformable drops. *Physics of Fluids*, 9:1493, 1997.
- [157] A. Zinchenko, M. Rother, and R. Davis. Cusping, capture, and breakup of interacting drops by a curvatureless boundary-integral algorithm. *Journal of Fluid Mechanics*, 391:249–292, 1999.