Immersed boundary simulations of cell-cell interactions in whole blood

Andrew Kassen[a,*], Aaron Barrett[a], Varun Shankar[b], Aaron L. Fogelson[a,c]

[a] *Department of Mathematics, University of Utah, Salt Lake City, UT 84112, USA*
[b] *School of Computing, University of Utah, Salt Lake City, UT 84112, USA*
[c] *Department of Bioengineering, University of Utah, Salt Lake City, UT 84112, USA*

Abstract

We present a new method for the geometric reconstruction of elastic surfaces simulated by the immersed boundary method with the goal of simulating the motion and interactions of cells in whole blood. Our method uses parameter-free radial basis functions for high-order meshless parametric reconstruction of point clouds and the elastic force computations required by the immersed boundary method. This numerical framework allows us to consider the effect of endothelial geometry and red blood cell motion on the motion of platelets. We find red blood cells to be crucial for understanding the motion of platelets, to the point that the geometry of the vessel wall has a negligible effect in the presence of RBCs. We describe certain interactions that force the platelets to remain near the endothelium for extended periods, including a novel platelet motion that can be seen only in 3-dimensional simulations that we term "unicycling." We also observe red blood cell-mediated interactions between platelets and the endothelium for which the platelet has reduced speed. We suggest that these behaviors serve as mechanisms that allow platelets to better maintain vascular integrity.

Keywords: Whole blood, Endothelium, Immersed boundary, RBFs

1. Introduction

Blood is a complex mixture of cellular and fluid-phase components, composed most notably of red blood cells (RBCs) and platelets suspended in plasma. RBCs are primarily the basis for transporting oxygen throughout the body. Platelets, meanwhile, play a key role in the maintenance of the vasculature. Blood flows under pressure through vessels, which vary in diameter between a few centimeters in the aorta to a few microns in the capillaries. A single layer of endothelial cells, called the endothelium, lines healthy blood vessels. Interactions between platelets and RBCs, and between platelets and the endothelium, are important for a platelet's function, but models of these interactions typically treat them in isolation.

At rest, RBCs are biconcave disk-shaped cells approximately 8 µm in diameter and 2.5 µm in thickness. The volume fraction occupied by RBCs, or *hematocrit*, ranges from approximately 36% to 45% in healthy humans. In

[*] Corresponding author

Email addresses: kassen@math.utah.edu (Andrew Kassen), barrett@math.utah.edu (Aaron Barrett), shankar@cs.utah.edu (Varun Shankar), fogelson@math.utah.edu (Aaron L. Fogelson)

order to deliver oxygen throughout the body, RBCs must be extremely flexible, as some vessels are smaller than the cell itself. Due to the wide variety of shapes exhibited by RBCs, their mechanical properties were studied intensively during the 1970s and 80s. Canham [1] theorized that the biconcave disk shape minimizes bending energy. Skalak *et al.* [2] devised a purpose-built constitutive law to describe the tension of RBC membranes. Under the assumption of a viscoelastic response, Evans & Hochmuth [3] estimated the membrane viscosity. Mohandas & Evans [4] gave estimates of the shear, bulk, and bending moduli, which have guided RBC models ever since [5, 6].

Platelets in their inactive state are ellipsoidal disks, approximately 3–4 μm by 1 μm in size. They are much more rigid than RBCs due to their actin and microtubule-based cytoskeletons. Platelets are also much less numerous, with 10–20 RBCs per platelet. Less is known about the mechanical properties of platelets. Models range from perfectly rigid ellipsoids [7] to systems of springs with [8, 9] or without [10] a preferred curvature. One study estimates the shear modulus and viscosity for platelets [11], but models tend to use a higher shear modulus than estimated and neglect viscous effects altogether.

Because they are deformable, RBCs tend to move towards the center of a blood vessel, and in doing so may encounter platelets, but the relative size and deformability of the RBC means a platelet is ejected from the RBC's path, ultimately pushing the platelet into an RBC-free layer along the vessel wall. This process, called margination, affects platelets and leukocytes (white blood cells) alike, and is the focus of many studies [8, 12–24]. From their marginated positions, platelets survey the vessel wall for injury. Injury sites expose proteins, *e.g.*, collagen and von Willebrand factor (vWF), to which platelets can bind and become activated. Platelet contact with the injured wall is the essential first step. This, in turn, leads the platelet to bind to the injury site and release its own chemical signals to recruit further platelets, which eventually results in the formation of a thrombus. All of this occurs in flowing blood, which sweeps these chemical signals downstream. While mechanisms for platelet activation have been proposed for low and pathologically high shear rates, the case of physiological shear rates is undecided [25]. For these reasons, we focus our efforts on studying the near-wall region of the vessel, where we expect shear-like flow.

Models of platelet motion over a thrombus indicate that there are stagnation zones immediately upstream and downstream of the thrombus, where the fluid velocity is very slow, even when the thrombus protrudes only a few microns from the vessel wall [7, 9]. Platelets that enter these regions may spend an extended period of time near the thrombus. The portion of an endothelial cell containing its nucleus also protrudes into the vessel approximately 1 μm. Moreover, these endothelial bumps are roughly periodic. If the endothelium creates a stagnation zone, the trailing zone from one protrusion might lead into the leading zone of the subsequent protrusion. This may allow for the sequestration of platelets or chemical signals. However, typical models of platelet-wall interaction model the endothelium as a flat surface [10, 23].

The goal of this article is therefore to conduct 3D simulations of whole blood, incorporating red blood cell, platelet, and endothelium interactions. Our model treats platelets and red blood cells as discrete elastic objects immersed in and interacting with blood (which is modeled as an incompressible Newtonian fluid). We use this model to compare the flow of whole blood across bumpy and flat walls and characterize the behavior and interactions of platelets with the wall and RBCs. To simulate this model, we develop a cohesive numerical framework comprising a fluid-structure interaction method, a meshless parametric modeling technique for reconstructing cell surfaces from point clouds and computing elastic force densities on these surfaces, and a meshless quadrature scheme that enables numerical integration of force densities on these surfaces as well.

We use the immersed boundary (IB) method for fluid-structure interaction. Originally developed by Peskin to study the flow of blood around heart valves [26], it has since been used to simulate, among numerous other applications, vibrations in the inner ear [27], the opening of a porous parachute [28], and sperm motility [29], and has generated numerous related methods. The IB method remains popular for modeling fluid-structure interaction because of its simplicity and ease of use, and involves maintaining an Eulerian description of the fluid and a purely Lagrangian description of all immersed elastic structures.

For parametric modeling and force density computations on these immersed elastic structures, we utilize meshless interpolation based on radial basis functions (RBFs), which have been used for generating differentiation matrices for the solution of PDEs [30], surface reconstruction [31–34], and in the context of regularized Stokeslets to represent interfaces and approximate their geometries [35]. More relevantly, RBFs have been used in the context of the IB method to reconstruct platelet surfaces from point clouds and to compute Lagrangian force densities in 2D simulations [36], where the authors find that the RBF-IB method generates smoother Eulerian forces with a smaller set of points than traditional IB methods. The smoother forces result in fewer iterations in the multigrid solver used for the fluid equations. A similar work [37] shows that smooth representations recover Lagrangian forces better than piecewise linear methods with far fewer points. However, the RBF-IB method has yet to be applied to surface reconstruction and force density calculation in 3D simulations. Further, the 2D version of the RBF-IB method presented in [36] required tuning in the RBF representation to achieve stability. In this work, we present the first extension of the RBF-IB method to the simulation of whole blood in 3D geometries. Due to the use of the meshless high-order accurate RBF-based representation, we are able to represent the constituent cells within whole blood as point clouds with relatively small cardinality. Further, we eliminate the aforementioned tuning parameter using a recently-developed parameter-free RBF representation [34].

The remainder of this paper is organized as follows. We begin with an overview of the IB method in Section 2. We then describe our method for solving the incompressible Navier-Stokes equations in Section 3. We describe the elastic models used for each type of cell in Section 4 and Section 5 details our methods for discretizing the cells using RBFs. Our results are presented in Section 6. Finally, we discuss the implications of our findings in Section 7.

2. The immersed boundary method

2.1. Overview

Consider a rectangular domain, $\Omega \subset \mathbb{R}^3$, which contains one or more deformable structures and is otherwise filled with an incompressible, Newtonian fluid with constant density, ρ, and viscosity, μ. The IB method treats these structures as an extension of the fluid. The motion of any particle in Ω is therefore governed by the incompressible Navier-Stokes equations,

$$\rho\left(\frac{\partial \boldsymbol{u}}{\partial t} + \nabla\cdot(\boldsymbol{u}\otimes\boldsymbol{u})\right) = \mu\Delta\boldsymbol{u} - \nabla p + \boldsymbol{f}, \tag{1}$$

$$\nabla\cdot\boldsymbol{u} = 0, \tag{2}$$

where, for $\boldsymbol{x} = (x, y, z) \in \Omega$, $\boldsymbol{u} = \boldsymbol{u}(\boldsymbol{x}, t) = (u, v, w)$ is the fluid velocity, $p = p(\boldsymbol{x}, t)$ is the pressure, and $\boldsymbol{f} = \boldsymbol{f}(\boldsymbol{x}, t)$ is an external force density. Here and throughout this paper, we use bold italic symbols to indicate vectors in $\mathbb{R}^3$. Treating the entire domain as a fluid allows us to discretize Ω independently of the immersed structures with a fixed Eulerian grid of spacing h. The discretization of (1) and (2) is discussed in detail in Section 3.

Let $\boldsymbol{X} = \boldsymbol{X}(\theta, \varphi, t)$, for surface coordinates (θ, φ) in $\Theta \subset \mathbb{R}^2$, be a parametrization for the immersed boundary Γ. In the IB method, a Lagrangian representation is used to track immersed boundaries. Nevertheless, in continuum form, $\boldsymbol{X}$ satisfies a no-slip condition and moves with the background fluid. The velocity of Γ can therefore be represented as a convolution of the local fluid velocity against the Dirac delta function $\delta(\boldsymbol{x})$, *i.e.*,

$$\frac{\partial \boldsymbol{X}}{\partial t} = \boldsymbol{U}(\boldsymbol{X}, t) = \int\limits_{\Omega} \boldsymbol{u}(\boldsymbol{x}, t)\delta(\boldsymbol{x} - \boldsymbol{X})\,\mathrm{d}\boldsymbol{x}, \tag{3}$$

where $\boldsymbol{U}$ is the restriction of $\boldsymbol{u}$ to Γ. The process of transferring the fluid velocity to the immersed structure is termed “interpolation.” As a boundary deforms, it generates a force density $\boldsymbol{F} = \boldsymbol{F}(\boldsymbol{X}, t)$, which it imparts onto the fluid as $\boldsymbol{f}$ in (1). Similar to (3), $\boldsymbol{F}$ is transferred to the fluid at $\boldsymbol{x}$ via

$$\boldsymbol{f}(\boldsymbol{x}, t) = \int\limits_{\Gamma} \boldsymbol{F}(\boldsymbol{X}, t)\delta(\boldsymbol{x} - \boldsymbol{X})\,\mathrm{d}\boldsymbol{X}. \tag{4}$$

This transfer of forces is called “spreading.” The structure Γ is typically represented in Lagrangian form, usually as a discrete set of points. Consequently, (3) must be discretized to update those points based on their velocities. Similarly, (4) must be discretized to both compute $\boldsymbol{F}(\boldsymbol{X}, t)$ and to approximate the integral with $\boldsymbol{F}(\boldsymbol{X}, t)$ over Γ.

Finally, the Dirac delta $\delta(\boldsymbol{x})$ is typically replaced by a smoothed, compactly-supported, h-dependent analog, typically referred to as the “discrete delta function” $\delta_h(\boldsymbol{x})$. Replacing δ with δ_h brings with it the need to modify the spread and interpolation operators near the boundaries so that the support of $\delta_h(\boldsymbol{x} - \boldsymbol{X})$ from extending outside of Ω. However, we avoid doing so by initializing Lagrangian points sufficiently far from the boundaries and ensuring that they do not come too close to the boundary throughout a simulation.

In the classical version of the IB method [38], velocities are interpolated to the same Lagrangian points that forces are spread from. However, there exist several IB methods which instead use different sets of points for each operation instead. For instance, Griffith & Luo [39] use a finite element representation for the structure, and consequently spread forces from (Lagrangian) quadrature points, interpolate velocities to quadrature points, and then project these using the finite element basis to the individual element nodes (which are also Lagrangian points). The RBF-IB method [36] used in this work does something similar. In Section 2.2, we discuss the discretization of (3) and (4) in the context of the RBF-IB method.

2.2. *The RBF-IB method*

The RBF-IB method obtains its name from the fact that it uses meshless interpolation with RBFs to parametrize the structure Γ as $\boldsymbol{X}(\theta, \varphi, t)$. However, it also distinguishes itself from the classical IB method by using two sets of Lagrangian points to represent the structure. One set of points, which we term "movement points" and label $\boldsymbol{X}^{\text{D}}_k$, $k = 1, \ldots, n_d$, are used to move the structure in the discrete analog to (3). Thus, the fully discrete version of (3) in the RBF-IB method reads

$$\frac{\partial \boldsymbol{X}^{\text{D}}_k}{\partial t} = \boldsymbol{U}(\boldsymbol{X}^{\text{D}}_k, t) \approx h^3 \sum_i \boldsymbol{u}(\boldsymbol{x}_i, t)\delta_h(\boldsymbol{x}_i - \boldsymbol{X}^{\text{D}}_k), \quad k = 1, \ldots, n_d, \tag{5}$$

where h is the spacing of the background Eulerian grid and i enumerates Eulerian grid points. In this work, we select the movement points to be initially approximately $2h$ apart in $\mathbb{R}^3$ at the start of a simulation. The RBF-IB method also uses a second set of points, which we term "spreading points" and label $\boldsymbol{X}^{\text{S}}_j$, $j = 1, \ldots, n_s$. These points are used to discretize (4) in order to spread forces to the background Eulerian grid. The fully discrete version of (4) in the RBF-IB method for an Eulerian grid point $\boldsymbol{x}_i$ can therefore be written as

$$\boldsymbol{f}(\boldsymbol{x}_i, t) \approx \sum_{j=1}^{n_s} W_j \boldsymbol{F}(\boldsymbol{X}^{\text{S}}_j)\delta_h(\boldsymbol{x}_i - \boldsymbol{X}^{\text{S}}_j), \tag{6}$$

where W_j, $j = 1, \ldots, n_s$ is a set of quadrature weights for integrating quantities at the spreading points $\boldsymbol{X}^{\text{S}}$; the spreading points can therefore be thought of as a set of quadrature points.

In general, the spreading points are selected to be approximately h apart in $\mathbb{R}^3$ at the start of a simulation; this spacing corresponds to spacing used for the single set of Lagrangian points (IB points) used in traditional IB simulations. Thus, in the RBF-IB method, $n_s > n_d$. In this sense, the RBF-IB method is a generalization of the standard IB method. The use of movement points and spreading points partially decouples the error in representing the structure Γ from the error in integrating forces. The spacing of the movement points $\boldsymbol{X}^{\text{D}}$ is chosen so that the reconstruction of Γ from $\boldsymbol{X}^{\text{D}}$ and the calculation of $\boldsymbol{F}(\boldsymbol{X}, t)$ by the RBF geometric model is sufficiently accurate. On the other hand, the spacing of the spreading points $\boldsymbol{X}^{\text{S}}$ is chosen so that the numerical integration of $\boldsymbol{F}(\boldsymbol{X}, t)$ is sufficiently accurate, to prevent leaks, and to ensure that spread forces overlap sufficiently on the Eulerian grid. In this work, we identify a set of fixed parametric locations $\Theta^{\text{D}} = \{(\theta_k, \varphi_k)\}$, $k = 1, \ldots, n_d$ with the movement points $\boldsymbol{X}^{\text{D}} = \boldsymbol{X}(\Theta^{\text{D}}, t)$, and call them "data sites." Analogously, we also maintain a set of fixed parametric points $\Theta^{\text{S}} = \{(\theta_j, \varphi_j)\}$, $j = 1, \ldots, n_s$, that we identify with the spreading points $\boldsymbol{X}^{\text{S}}$; these points are called "sample sites." In general, we need only explicitly track the movement points $\boldsymbol{X}^{\text{D}}$, reconstruct Γ from $\boldsymbol{X}^{\text{D}}$ and the Θ^{D} using our RBF geometric model in order to compute $\boldsymbol{F}(\boldsymbol{X}, t)$, and regenerate the spreading points $\boldsymbol{X}^{\text{S}}$ at the end of every step using the Θ^{S} and the RBF geometric model.

Despite the advantages mentioned above, the choice of two sets of Lagrangian points potentially raises certain concerns, which we address here. The first major concern relates to the force-spreading and velocity interpolation operations. In traditional IB methods, the corresponding spreading and interpolation operators are formally adjoint, leading to conservation of energy/power [38]. However, in the RBF-IB method, these operators are no longer adjoint, leading to concerns about numerical stability. Nevertheless, in previous work [36], the authors demonstrated that the RBF-IB method dissipated energy in 2D simulations as expected, despite the absence of formal adjointness. In this work, we show similar results for an RBC relaxation problem in a full 3D simulation in Section 6.3. The second major concern relates to the wider $2h$ spacing of the movement points $\boldsymbol{X}^{\mathrm{D}}$. If the points are spaced too widely apart, it is possible to obtain unphysical behavior, with some parts of the structure responding more to the neighboring fluid than others. However, both in [36] and in this work, we observed that starting these points with a spacing of $2h$ was sufficient to reproduce standard results for RBC simulations. It may be possible to alleviate any spacing issues by periodically rearranging the movement points $\boldsymbol{X}^{\mathrm{D}}$, but this approach was not necessary in this work, and so we leave a full exploration of such Lagrangian rearrangement strategies for future work.

On a higher level, (6) illustrates the need for three pieces of information for each immersed structure: the points $\boldsymbol{X}^{\mathrm{s}}_j$ used to evaluate forces on the structure; a force density $\boldsymbol{F}(\boldsymbol{X}^{\mathrm{s}}_j)$ at each of those points; and surface integration or quadrature weights W_j for those points. We detail the RBF-based meshless geometric model used to obtain $\boldsymbol{X}^{\mathrm{s}}_j$, $\boldsymbol{F}(\boldsymbol{X}^{\mathrm{s}}_j)$, and W_j in Section 5, give analytic expressions for energy densities used to derive $\boldsymbol{F}$ in Section 4, and give analytic expressions for the force densities $\boldsymbol{F}$ in AppendixB.

3. Solution of the incompressible Navier-Stokes equations

3.1. Spatial Discretization

To discretize the Navier-Stokes equations (1) and (2), we use a marker-and-cell (MAC) grid [40]: for grid cell center $\boldsymbol{x}_i$, scalar-valued function $s(\boldsymbol{x})$ is discretized at $\boldsymbol{x}_i$, and component $\boldsymbol{e}_a \cdot \boldsymbol{v}$ of vector-valued function $\boldsymbol{v}(\boldsymbol{x})$ at $\boldsymbol{x}_i - {}^1\!/\!{}_2 h\boldsymbol{e}_a$, where $\boldsymbol{e}_a$ is a canonical basis vector. Define the centered difference operator

$$D_a\phi(\boldsymbol{x}) = \frac{\phi(\boldsymbol{x} + {}^1\!/\!{}_2 h\boldsymbol{e}_a) - \phi(\boldsymbol{x} - {}^1\!/\!{}_2 h\boldsymbol{e}_a)}{h}, \tag{7}$$

for which, *e.g.*, D_1 approximates differentiation in the x direction. The discrete divergence, gradient, and Laplacian operators use centered differences, resulting in a 2-point stencil for each discrete first derivative and the standard 7-point discrete Laplacian. We also define the centered average operator

$$A_a\phi(\boldsymbol{x}) = \frac{\phi(\boldsymbol{x} + {}^1\!/\!{}_2 h\boldsymbol{e}_a) + \phi(\boldsymbol{x} - {}^1\!/\!{}_2 h\boldsymbol{e}_a)}{2}. \tag{8}$$

By averaging u in the y direction and v in the x direction, we obtain collocated approximations to u and v at the center of a cell edge. Averaging, e.g., u in the x direction yields an approximation to u at the cell center. We can therefore discretize the components of the advection term $\nabla \cdot (\boldsymbol{u} \otimes \boldsymbol{u})$ by

$$\nabla_h \cdot (\boldsymbol{u} \otimes \boldsymbol{u}) := \begin{bmatrix} D_1[(A_1u)(A_1u)] + D_2[(A_1v)(A_2u)] + D_3[(A_1w)(A_3u)] \\ D_1[(A_2u)(A_1v)] + D_2[(A_2v)(A_2v)] + D_3[(A_2w)(A_3v)] \\ D_1[(A_3u)(A_1w)] + D_2[(A_3v)(A_2w)] + D_3[(A_3w)(A_3w)] \end{bmatrix}. \tag{9}$$

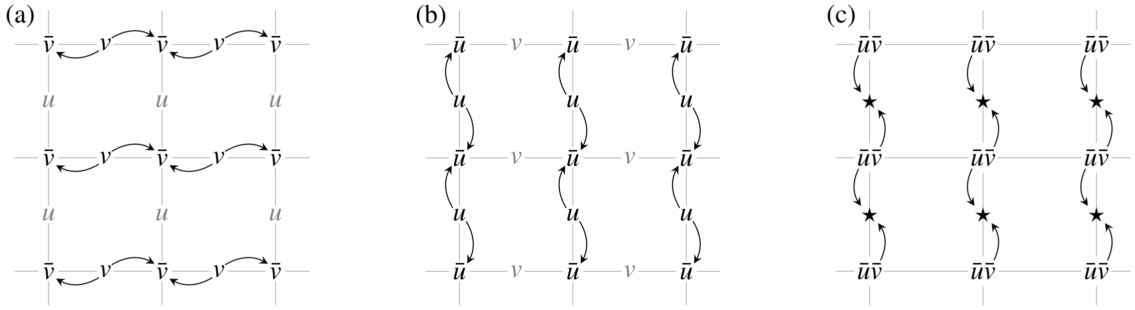


Figure 1: A cross-section illustrating the steps in computing the $D_2[(A_2u)(A_1v)]$ term of the first component of the advection. The horizontal and vertical velocity component discretization locations are marked by u and v, respectively. Arrows emanate from a point contributing to a stencil and point to the center of the stencil. (a) A_1 averages v in the x direction, yielding an approximation $\bar{v}$ at grid vertices (in 3D, centers of cell edges for which x and y are constant). (b) A_2 averages u in the y direction, yielding an approximation $\bar{u}$ at the same points as (a). The quantities A_1v and A_2u are collocated and can be directly multiplied to obtain an approximation of uv at locations marked $\bar{u}\bar{v}$. (c) D_2 approximately differentiates uv in the y direction, yielding the desired quantity at each point marked ★. The approximation of uv is also used to compute $D_1[(A_1v)(A_2u)]$ in the second component of the advection, wherein application of D_1 instead yields approximations collocated with locations marked v in (a).

The symbol $\nabla_h\cdot$ represents the discrete divergence operator. Figure 1 illustrates the steps in computing $D_2[(A_1v)(A_2u)]$, which appears in the first component in (9). Morinishi *et al.* [41] show that this scheme, $Div. - S2$ in their parlance, is conservative under the assumption that $\boldsymbol{u}$ is discretely divergence-free.

3.2. *Temporal Discretization*

To advance the solution, we use either the backward-forward Euler-based scheme [42] or the 2-stage scheme described by Peskin [38], modified to advance structures using the newest velocities. Both schemes utilize a pressure projection (or fractional step) method. The modification makes these schemes formally first-order in time, but allow us to separate the Eulerian update from the Lagrangian update by requiring only information at the beginning of the timestep to evaluate forces and moving the structure at the end of the timestep. For the backward-forward Euler scheme, discretizing (1) to advance time to $t + k$, yields linear solves of Helmholtz type,

$$(I - k\rho^{-1}\mu\Delta_h)\boldsymbol{u}^* = \boldsymbol{u}^n - k\left[\nabla_h \cdot (\boldsymbol{u}^n \otimes \boldsymbol{u}^n) + \rho^{-1}\left(\boldsymbol{f}^{n+1} - \nabla_h p^n\right)\right] \quad \text{in } \Omega, \tag{10}$$

with boundary conditions

$$\boldsymbol{u}^* = \boldsymbol{u}_b^{n+1} + k\nabla_h q^n \quad \text{on } \partial\Omega, \tag{11}$$

where superscripts denote the time step, $\boldsymbol{u}_b$ is velocity boundary data, Δ_h and ∇_h are the discrete Laplacian and gradient, respectively, and q is described below. The force density $\boldsymbol{f}^{n+1}$ is advanced using only data at timestep n. The intermediate velocity field $\boldsymbol{u}^*$ may not be divergence-free. To obtain a velocity field that satisfies (2), we use projection method II (PmII) of Brown, Cortez, and Minion [43]. PmII updates the pressure using

$$p^{n+1} = p^n + (\rho I - k\mu\Delta_h)q^{n+1}, \tag{12}$$

and generates the divergence-free velocity field

$$\boldsymbol{u}^{n+1} = \boldsymbol{u}^* - k\nabla_h q^{n+1} \tag{13}$$

using the pseudo-pressure q^{n+1}, which satisfies

$$\begin{aligned} k\Delta_h q^{n+1} &= \nabla_h \cdot \boldsymbol{u}^* && \text{in } \Omega, \\ \boldsymbol{n} \cdot \nabla_h q^{n+1} &= 0 && \text{on } \partial\Omega. \end{aligned} \tag{14}$$

The velocity update (13) provides the boundary conditions (11) using a lagged value of the pseudo-pressure. The 2-stage RK method consists of a backward-forward Euler step followed by a Crank-Nicolson-midpoint step, which involves only minor modifications to (10). In total, we perform 3 Helmholtz solves and 1 Poisson solve per RK stage.

We employ preconditioned conjugate gradients (PCG) to perform the solves. We use Chebyshev iteration as a preconditioner for the Helmholtz solves and as an error smoothing procedure and direct solver for multigrid (MG) to precondition the Poisson solve. Chebyshev iteration is a generalization of weighted Jacobi iteration which requires only the ability to perform sparse matrix polynomial-vector multiplication. Chebyshev iteration (MG) PCG is therefore parallelized by using a parallel sparse matrix-vector multiplication routine with Horner's method to evaluate the polynomials.

In the case of a triply periodic domain, the linear solves involve symmetric matrices. For Dirichlet or Neumann boundaries, we extrapolate using values at the neighboring grid points and boundary data to fill ghost points. In these situations, the standard discrete second derivative may actually approximate some non-unit multiple of its continuous counterpart near the boundary. To account for this while maintaining symmetry of the Helmholtz matrices, we scale equations involving near-boundary values. For details, see AppendixA. The trade-off is 3 extra diagonal matrix-vector multiplications per RK stage for the ability to use PCG for the linear solves.

4. Cell energy and power models

In this section, we describe the various forms of energy density (energy per area) and power density (power per area) used in our simulations, and give analytic expressions for each. The corresponding force densities are given in AppendixB. We use five kinds of densities: spring energy, damped spring power, tension energy, dissipative power, and Canham-Helfrich bending energy. For constitutive law W, we define the functional

$$\mathcal{E}[\boldsymbol{X}, \boldsymbol{U}] = \int_\Gamma W(\boldsymbol{X}, \boldsymbol{U}, \ldots)\,\mathrm{d}\boldsymbol{X}, \tag{15}$$

where the ellipsis indicates that W may depend on spatial derivatives of $\boldsymbol{X}$ or $\boldsymbol{U}$. The force density associated with W is found by computing the first variation of $\mathcal{E}$,

$$\boldsymbol{F} = -\delta\mathcal{E}, \tag{16}$$

with respect to $\boldsymbol{X}$ for energy densities and to $\boldsymbol{U}$ for power densities. Because our ultimate goal is a three-dimensional simulation, we limit our descriptions to the three-dimensional case. Considerations for the two-dimensional case are treated elsewhere [38, 44].

We begin with Hookean energy and damped spring power density. These have the simplest constitutive laws we use, depend only on surface locations and surface velocities, and take the form

$$W_{\mathrm{Hk}}(\boldsymbol{X}) = \frac{k}{2}\|\boldsymbol{X} - \boldsymbol{X}'\|^2 \quad \text{and} \quad W_{\mathrm{damped}}(\boldsymbol{U}) = \frac{\eta}{2}\|\boldsymbol{U} - \boldsymbol{U}'\|^2, \tag{17}$$

where $\boldsymbol{X}' = \boldsymbol{X}'(\theta, \varphi, t)$ is the tether location for $\boldsymbol{X}(\theta, \varphi, t)$, $\boldsymbol{U}' = \boldsymbol{U}'(\theta, \varphi, t)$ is the prescribed velocity of the tether point, k is the spring constant, and η is the damping constant. Due to the lack of information about the mechanical properties of endothelial cells, we model the endothelium as an approximately rigid, stationary object with $k_{\mathrm{endo}} = 2.5\,\mathrm{dyn/cm}$ and $\eta_{\mathrm{endo}} = 2.5 \times 10^{-7}\,\mathrm{dyn \cdot s/cm}$, chosen to be as large as possible for the chosen spatial and temporal step size with prescribed velocity $\boldsymbol{U}' = \boldsymbol{0}$. We compare different choices for $\boldsymbol{X}'$ in Section 6.6.

Next, we consider the tension energy densities for RBCs and platelets. These penalize stretching and areal dilation of the cell membranes. Let λ_1 and λ_2 be the principal extensions, *i.e.*, the maximal and minimal ratios of stretching relative to a reference configuration. We define the invariants $I_1 = \lambda_1^2 + \lambda_2^2 - 2$ and $I_2 = \lambda_1^2\lambda_2^2 - 1$, which measure relative changes in length and area, respectively, such that $I_1 = I_2 = 0$ correspond to a rigid body motion. We express the tension density in terms of these invariants. Skalak's Law was designed specifically for RBCs [2]:

$$W_{\mathrm{Sk}}(I_1, I_2) = \frac{E}{4}\left(I_1^2 + 2I_1 - 2I_2\right) + \frac{G}{4}I_2^2. \tag{18}$$

E is the shear modulus, and G is the bulk modulus. For RBCs, we follow Fai *et al.* [6] and set $E_{\mathrm{RBC}} = 2.5 \times 10^{-3}\,\mathrm{dyn/cm}$ and $G_{\mathrm{RBC}} = 2.5 \times 10^{-1}\,\mathrm{dyn/cm}$. We use the shape given by Evans & Fung [45] for the reference RBC with radius $R_0 = 3.91\,\mu\mathrm{m}$,

$$\hat{\boldsymbol{X}}_{\mathrm{RBC}}(\theta, \varphi) = R_0 \begin{bmatrix} \cos\theta\cos\varphi \\ \sin\theta\cos\varphi \\ z(\cos\varphi)\sin\varphi \end{bmatrix}, \tag{19}$$

where $(\theta, \varphi) \in (-\pi, \pi] \times [-\pi/2, \pi/2]$ and $z(r) = 0.105 + r^2 - 0.56r^4$. Platelets, on the other hand, do not have a purpose-built constitutive law, but are known to be stiffer than RBCs. We use the neo-Hookean model

$$W_{\mathrm{nH}}(I_1, I_2) = \frac{E}{2}\left(\frac{I_1 + 2}{\sqrt{I_2 + 1}} - 2\right) + \frac{G}{2}\left(\sqrt{I_2 + 1} - 1\right)^2 \tag{20}$$

with $E_{\mathrm{plt}} = 1 \times 10^{-1}\,\mathrm{dyn/cm}$ and $G_{\mathrm{plt}} = 1\,\mathrm{dyn/cm}$, and an ellipsoidal reference configuration [46]

$$\hat{\boldsymbol{X}}_{\mathrm{plt}}(\theta, \varphi) = \begin{bmatrix} 1.55\,\mu\mathrm{m}\cos\theta\cos\varphi \\ 1.55\,\mu\mathrm{m}\sin\theta\cos\varphi \\ 0.5\,\mu\mathrm{m}\sin\varphi \end{bmatrix}. \tag{21}$$

Platelets and RBCs also respond to changes in membrane curvature. Let H be the membrane's mean curvature. The Canham-Helfrich bending energy density takes the form [1]

$$W_{\text{CH}}(H) = 2\kappa(H - H')^2, \tag{22}$$

where κ is the bending modulus in units of energy, and H' is the spontaneous or *preferred* curvature. An RBC generates a relatively weak response to changes in its curvature. Its bending modulus is estimated to be in the range 0.3–4 × 10^{-12} erg [4]. We use a bending modulus of $\kappa_{\text{RBC}} = 2 \times 10^{-12}$ erg and a preferred curvature $H' = 0$ for RBCs. RBCs, therefore, tend to locally flatten their membranes. For platelets, we use a larger bending modulus of $\kappa_{\text{plt}} = 2 \times 10^{-11}$ erg and a preference for its reference curvature. Together with the neo-Hookean tension above, this maintains a fairly rigid platelet.

Finally, we consider dissipative power, which causes the membrane to exhibit a viscoelastic response to strain. It takes the form [47]

$$W_{\text{dissip}}(\dot{\lambda}_1, \dot{\lambda}_2) = \frac{\nu}{2}\left(\frac{\dot{\lambda}_1^2}{\lambda_1^2} + \frac{\dot{\lambda}_2^2}{\lambda_2^2}\right), \tag{23}$$

where ν is the membrane viscosity, and $\dot{\lambda}_i$ is the rate of change of λ_i. We imbue only the RBC with viscoelasticity. We find this effective in eliminating some numerical instabilities, as discussed in Section 6.1. While Evans & Hochmuth suggest a viscosity of approximately 1×10^{-3} dyn · s/cm [3], we find this to be prohibitively expensive in practice, due to time step restrictions, and instead use $\nu_{\text{RBC}} = 2.5 \times 10^{-7}$ dyn · s/cm.

Before discussing the representation of cells, we briefly remark on a detail of discretizing the spread operation (4). As written, this and the energy functional (15) are expressed as integrals over the deformed configuration. One can easily convert between forces computed with respect to the reference and deformed configurations using

$$J\boldsymbol{F}_{\text{def}} = \hat{J}\boldsymbol{F}_{\text{ref}},$$

where $\boldsymbol{F}_{\text{def}}$ is a force computed in the deformed configuration, $\boldsymbol{F}_{\text{ref}}$ is a force computed in the reference configuration, and $J = |\partial\boldsymbol{X}/\partial\boldsymbol{\theta}|$ and $\hat{J} = |\partial\hat{\boldsymbol{X}}/\partial\boldsymbol{\theta}|$ are the Jacobians of the mappings $\boldsymbol{X} : \Theta \to \mathbb{R}^3$ and $\hat{\boldsymbol{X}} : \Theta \to \mathbb{R}^3$, respectively. The spreading operation (4) can therefore be written in the following equivalent ways:

$$\int_\Gamma \delta(\boldsymbol{x} - \boldsymbol{X})\boldsymbol{F}_{\text{def}}\,\mathrm{d}\boldsymbol{X} = \int_{\mathbb{S}^2} \delta(\boldsymbol{x} - \boldsymbol{X}(\boldsymbol{\theta}))\boldsymbol{F}_{\text{def}}J\,\mathrm{d}\boldsymbol{\theta} = \int_{\mathbb{S}^2} \delta(\boldsymbol{x} - \boldsymbol{X}(\boldsymbol{\theta}))\boldsymbol{F}_{\text{ref}}\hat{J}\,\mathrm{d}\boldsymbol{\theta}.$$

The quantities $J\,\mathrm{d}\boldsymbol{\theta}$ and $\hat{J}\,\mathrm{d}\boldsymbol{\theta}$, when discretized (see Section 5.3), are the quadrature weights for the deformed and reference configuration, respectively. Written as an integral over $\mathbb{S}^2$, the integrals for the deformed and reference configurations take a similar form. It is therefore a matter of convenience which configuration is used. For instance, we use reference configurations for tension forces, and the deformed configuration for bending and dissipative forces, based solely on how the literature tends to report those forces. Regardless of how we compute the forces, we will continue to write these integrals (and their discretized versions) with respect to the deformed configuration to keep the exposition simple.

5. Geometry of reconstructed surfaces

From the previous section, we have analytic expressions for the energy and power densities, and consequently the Lagrangian force density, $\boldsymbol{F}$ (see AppendixB). The RBC and platelet force models require first and second derivatives along their surfaces. Additionally, the IB force spreading operation requires quadrature weights for the cell surfaces. This section finishes where we left off by discussing the construction of the necessary discrete linear operators and quadrature weights W_j through the use of RBF-based methods.

5.1. Surface reconstruction with radial basis functions

We now describe our RBF method for reconstructing cell surfaces. RBF interpolation is a meshfree approach to scattered data interpolation where structural information is encoded purely as point-wise distances. This contrasts with, *e.g.*, polynomials, where points must be chosen at grid vertices, or spherical harmonics, for which special node sets are typically used. RBFs have been shown to be a viable approach for representing cells on a par with Fourier methods [33]. They are therefore appealing for representing red blood cells and platelets.

Our goal is to produce a parametric reconstruction of cells. We choose to parametrize both red blood cells and platelets on the 2-sphere, $\mathbb{S}^2$. We use a spherical coordinate mapping to relate Cartesian points on the 2-sphere to their corresponding parametric points. This mapping is given by

$$\chi(\theta, \varphi) = \begin{bmatrix} \cos\theta\cos\varphi \\ \sin\theta\cos\varphi \\ \sin\varphi \end{bmatrix}, \tag{24}$$

$(\theta, \varphi) \in (-\pi, \pi] \times [-\pi/2, \pi/2]$. As mentioned in Section 2.2, let $\Theta^{\text{D}} = \{(\theta_k, \varphi_k)\}$, $k = 1, \ldots, n_d$, be a set of distinct *data sites*, defined by the Bauer spiral [48],

$$\begin{aligned} \varphi_k &= \sin^{-1}(-1 + (2k-1)/n_d), \\ \theta_k &= \operatorname{mod}\left(\sqrt{n_d\pi}\varphi_k + \pi, 2\pi\right) - \pi, \end{aligned} \tag{25}$$

where $\operatorname{mod}(a, b) = a - b\lfloor a/b \rfloor$ is the modulo function. These data sites correspond to the movement points $\boldsymbol{X}^{\text{D}}$ to which the fluid grid velocity is interpolated. We similarly generate a potentially larger set of *sample sites*, Θ^{S}, from which to spread forces by replacing n_d with n_s in the above equation. Computational or memory constraints motivate particular choices of n_d and n_s. We want n_d large enough to faithfully represent the cell's geometry and the number of points, n_s, from which forces are spread sufficiently large to preclude fluid leaking across the cell surface. Let $\chi_i^{\text{D}} = \chi(\theta_i, \varphi_i)$ for each $(\theta_i, \varphi_i) \in \Theta^{\text{D}}$. In this setting, our goal is to construct a parametric mapping from Θ^{D} to the Cartesian locations of points on cell surfaces. For RBC and platelet parametrizations, we identify the point $\boldsymbol{X}(\theta, \varphi, t)$ on Γ with the point $\chi(\theta, \varphi)$ on $\mathbb{S}^2$. Each component of $\boldsymbol{X}$ and consequently each component of a movement point $\boldsymbol{X}_k^{\text{D}}$ is then a function defined on $\mathbb{S}^2$. The problem of surface reconstruction therefore involves approximating each of these functions from their values at Θ^{D} using an RBF interpolant. For the discussion that follows, we use $\psi(\chi) : \mathbb{S}^2 \to \mathbb{R}$ to denote a function that we wish to approximate.

Let $\phi : \mathbb{S}^2 \times \mathbb{S}^2 \to \mathbb{R}$ be a *radial kernel* with the property that $\phi(\chi_i, \chi_j) \equiv \phi(\|\chi_i - \chi_j\|)$. These kernels are sometimes called *spherical basis functions*. In addition to the radial kernels, let $p_k(\chi)$, $k = 1, \ldots, n_p$ denote the first n_p spherical harmonics, which form a natural basis for polynomial approximation on $\mathbb{S}^2$. Then, the RBF interpolant to $\psi(\chi)$ takes the form

$$s(\chi) = \sum_{k=1}^{n_d} c_k \phi(\|\chi - \chi_k^{\text{D}}\|) + \sum_{i=1}^{n_p} d_i p_i(\chi), \tag{26}$$

where c_k and d_i are unknown interpolation coefficients, and the metric can be written parametrically as

$$\|\chi - \chi_k^{\text{D}}\| = \|\chi(\theta, \varphi) - \chi(\theta_k, \varphi_k)\| = \sqrt{2(1 - \cos\varphi \cos\varphi_k \cos(\theta - \theta_k) - \sin\varphi \sin\varphi_k)}. \tag{27}$$

To find c_k and d_i, we enforce that the interpolant (26) exactly interpolate the function $\psi(\chi)$ at each χ_k^{D},

$$s(\chi_k^{\text{D}}) = \psi(\chi_k^{\text{D}}), \qquad k = 1, \ldots, n_d, \tag{28}$$

and that (26) exactly reproduce the first n_p spherical harmonics everywhere on $\mathbb{S}^2$ [30],

$$\sum_{k=1}^{n_d} c_k p_i(\chi_k^{\text{D}}) = 0, \qquad i = 1, \ldots, n_p. \tag{29}$$

We collect (28) and (29) into a dense symmetric block system of the form

$$\begin{bmatrix} \Phi & P \\ P^T & 0 \end{bmatrix} \begin{bmatrix} \vec{c} \\ \vec{d} \end{bmatrix} = \begin{bmatrix} \vec{\psi} \\ \vec{0} \end{bmatrix}, \tag{30}$$

where $\vec{c}$ and $\vec{d}$ are the unknown coefficients, $\vec{\psi}$ is the vector of evaluations of $\psi(\chi^{\text{D}})$, Φ represents the evaluations of ϕ, P represents evaluations of the polynomials p_k, the matrix block 0 is the $n_p \times n_p$ zero matrix, and $\vec{0}$ is a vector of n_p zeros. Because Θ^{D} is fixed, we construct this matrix and compute its factors only once even though $\psi(\chi^{\text{D}})$ changes. The matrix in (30) is invertible for any conditionally positive definite kernel ϕ of order m as long as the data sites Θ^{D} are unisolvent for the first $n_p \geq (m+1)^2$ spherical harmonics, *i.e.*, P is of full rank [30]. A common heuristic choice to ensure this is to set $n_d = 2n_p$ [49].

It now remains to discuss the choice of ϕ. In previous work [36], we chose ϕ to be an infinitely-smooth and positive-definite kernel. While these kernels offer spectral convergence rates, they require tuning a so-called shape parameter [30]. In particular, we instead choose ϕ to be a polyharmonic spline, which is a conditionally-positive definite kernel with finite smoothness. In this work, we set $\phi(r) = r^7$, augmented with fifth-order spherical harmonics to represent RBCs, or just the constant spherical harmonic for platelets. As mentioned previously, we identify the point $\boldsymbol{X}(\theta, \varphi, t)$ on Γ with the point $\chi(\theta, \varphi)$ on $\mathbb{S}^2$. To reconstruct the surface $\boldsymbol{X}$ from the movement points $\boldsymbol{X}^{\text{D}}$, we simply use the fact that each component of a particular movement point $\boldsymbol{X}_k^{\text{D}}$ is one of n_d samples of a function of the form $\psi(\chi)$. For instance, to reconstruct the first component of $\boldsymbol{X}$, we set $\vec{\psi}$ to be the vector of x-coordinates of the movement points $\boldsymbol{X}^{\text{D}}$, and find a set of coefficients corresponding to the x-component of $\boldsymbol{X}$. This process is repeated component-wise for the movement points, and is essentially a component-wise interpolation of the movement points [36].

Clearly, by replacing the quantity $\vec{\psi}$ with samples of any function defined on the sphere, one can also use (26) to interpolate other quantities on the surface Γ, since all such quantities are functions of the form $\psi(\chi)$. In this work, we also use (26) to compute force densities required by the IB method. It is clear from (18)–(23) that computing the force densities requires values of I_1, I_2, and H, among others. These values are derived from the first and second derivatives of $\boldsymbol{X}$, which can be obtained by analytically differentiating the RBF and spherical harmonic bases.

In practice, we do not compute any coefficients in our simulations. For efficiency, it is possible to reformulate the process of interpolation followed by differentiation as a single application of a discrete (nodal) differential operator (or a differentiation matrix). We discuss this in the next section.

5.2. *Discrete linear surface operators*

Let $\mathcal{L}$ be a linear operator. In particular, we are interested in the first- and second-order partial differential operators, $\partial/\partial\theta$, $\partial^2/\partial\theta\partial\varphi$, *etc*. We seek a matrix L such that $L\vec{\psi} \approx \mathcal{L}\psi|_{\Theta}$, where Θ is either Θ^{D} or Θ^{S}. First, note that since $s(\chi) \approx \psi(\chi)$, it is also true that $\mathcal{L}s(\chi) \approx \mathcal{L}\psi(\chi)$. Thus, in the RBF-IB method, it is of interest to be able to apply $\mathcal{L}$ to $s(\chi)$ and evaluate the resulting function efficiently. To do so, we formulate all applications and evaluations of $\mathcal{L}s$ in terms of a matrix-vector multiplication with the quantity $\vec{\psi}$. To see how this is done, first note that if we wish to approximate samples of $\mathcal{L}s$ at a given set of sample sites Θ^{S}, we can write

$$\mathcal{L}s(\chi)\Big|_{\Theta^{\text{S}}} = \sum_{k=1}^{n_d} c_k \mathcal{L}\phi(\|\chi - \chi_k^{\text{D}}\|) + \sum_{i=1}^{n_p} d_i \mathcal{L}p_i(\chi), \tag{31}$$

which can be expressed more compactly in terms of matrix-vector multiplications as

$$\mathcal{L}s(\chi)\Big|_{\Theta^{\text{S}}} = \begin{bmatrix} \mathcal{L}\Phi & \mathcal{L}P \end{bmatrix} \begin{bmatrix} \vec{c} \\ \vec{d} \end{bmatrix}. \tag{32}$$

However, we can use (30) to rewrite this as

$$\mathcal{L}s(\chi)\Big|_{\Theta^{\text{S}}} = \begin{bmatrix} \mathcal{L}\Phi & \mathcal{L}P \end{bmatrix} \begin{bmatrix} \Phi & P \\ P^T & 0 \end{bmatrix}^{-1} \begin{bmatrix} \vec{\psi} \\ \vec{0} \end{bmatrix} = \begin{bmatrix} L & * \end{bmatrix} \begin{bmatrix} \vec{\psi} \\ \vec{0} \end{bmatrix}, \tag{33}$$

where $\mathcal{L}\Phi$ and $\mathcal{L}P$ represent evaluations of $\mathcal{L}\phi$ and $\mathcal{L}p_k$, respectively, and the $n_s \times n_d$ (dense) matrix L is the discrete analog of $\mathcal{L}$. L is similar in spirit to a finite-difference or nodal pseudospectral differentiation matrix, which approximates derivatives of nodal function samples by a matrix-vector multiplication. It is important to note that L is completely independent of the function ψ; in fact, it depends only on the functions ϕ and p_k, the fixed data sites Θ^{D}, and the fixed sample sites Θ^{S}. The block marked by $*$ is multiplied by zeros, and can be discarded. We compute a separate L for each operator $\mathcal{L}$ as a preprocessing step, and simply apply these matrices to any function that needs to be evaluated or differentiated. For instance, setting $\mathcal{L}$ to be the point evaluation operator and setting $\vec{\psi}$ to each component of the movement points $\boldsymbol{X}^{\text{D}}$ allows us to generate each component of the spreading points $\boldsymbol{X}^{\text{S}}$. Similarly, we can obtain samples of parametric derivatives of any function $\psi(\chi)$ at Θ^{S} by replacing $\mathcal{L}$ with that derivative operator.

It is also straightforward to generate versions of L that produce derivatives at Θ^{D} simply by replacing $\mathcal{L}s|_{\Theta^{\text{s}}}$ with $\mathcal{L}s|_{\Theta^{\text{D}}}$ in the above discussion. With these operators in hand, the quantities I_1, I_2, and H in Section 4 are readily discretized, as are the force densities in AppendixB. Application of the dense discrete differential operators is performed in parallel with a parallel implementation of `BLAS`. Lagrangian forces can therefore be computed in parallel with few thread synchronizations.

We now have a method for computing a suitable set of points and for discretizing $\boldsymbol{F}$ for use in (4). To compute a force from a force density, we need to compute the set of quadrature weights W_j, $j = 1, \ldots, n_s$. The following section is devoted to describing our method for computing these weights.

5.3. *RBF-based quadrature*

We now describe how (4) is discretized in the RBF-IB method. We repeat this equation for convenience:

$$\boldsymbol{f}(\boldsymbol{x}, t) = \int_{\Gamma} \boldsymbol{F}(\boldsymbol{X}, t)\delta(\boldsymbol{x} - \boldsymbol{X})\, \mathrm{d}\boldsymbol{X}.$$

First, we note that this integral is over Γ, which is a time-varying structure. As noted in Section 4, while some force models are defined with respect to a reference frame (*e.g.*, tension forces), the general case involves force computations directly on Γ. For this general case, we require *time-varying* quadrature weights W_j at the spreading points $\boldsymbol{X}_j^{\text{s}}$. Our approach to doing so is as follows:

1. First precompute a fixed set of quadrature weights ω_j on $\mathbb{S}^2$.

2. Map these weights to Γ during each time-step.

While $W_j(\theta, \varphi, t)$ is technically a function of time t, we suppress the arguments for brevity. For the reference configuration, quadrature weights follow the same process, but need only be computed once.

5.3.1. *Computing fixed quadrature weights on $\mathbb{S}^2$*

First, we discuss how to compute RBF-based meshless quadrature weights on $\mathbb{S}^2$. Given a function $\psi : \mathbb{S}^2 \to \mathbb{R}$ and a radial kernel $\phi : \mathbb{S}^2 \times \mathbb{S}^2 \to \mathbb{R}$, we now wish to find a set of quadrature weights ω_j such that

$$\int_{\mathbb{S}^2} \psi(\chi)\, \mathrm{d}\chi \approx \sum_{j=1}^{n_s} \omega_j \psi(\chi_j^{\text{s}}). \tag{34}$$

We use a variant of the technique described by Fuselier *et al.* [50]. Choosing $\psi(\chi) = \phi(\|\chi - \chi_i^{\text{s}}\|)$ for each χ_i^{s}, (34) becomes

$$\sum_{j=1}^{n_s} \omega_j \phi(\|\chi_j^{\text{s}} - \chi_i^{\text{s}}\|) \approx \int_{\mathbb{S}^2} \phi(\|\chi - \chi_i^{\text{s}}\|)\, \mathrm{d}\chi := \mathcal{L}\phi|_{\chi_i^{\text{s}}}. \tag{35}$$

However, because

$$\|\chi - \chi_i^{\text{s}}\|^2 = (\chi - \chi_i^{\text{s}}) \cdot (\chi - \chi_i^{\text{s}}) = 2 - 2\chi \cdot \chi_i^{\text{s}} \tag{36}$$

depends only on the angle between the two points, $\mathcal{L}\phi$ is constant over the sphere, and we denote this constant $-I_\phi$. We require further that the ω_j sum to the surface area of $\mathbb{S}^2$, *i.e.*,

$$\sum_{j=1}^{n_s} \omega_j = 4\pi. \tag{37}$$

Treating I_ϕ as an unknown scalar, we rewrite the constraints (35) and (37) in the symmetric block linear system

$$\begin{bmatrix} \Phi & \vec{1} \\ \vec{1}^T & 0 \end{bmatrix} \begin{bmatrix} \vec{\omega} \\ I_\phi \end{bmatrix} = \begin{bmatrix} \vec{0} \\ 4\pi \end{bmatrix}, \tag{38}$$

where $\vec{\omega}$ are the unknown weights, Φ represents the evaluations of ϕ, and $\vec{0}$ and $\vec{1}$ are vectors of n_s zeros and ones, respectively. I_ϕ serves as a Lagrange multiplier that enforces (37), and $-I_\phi$ is a good approximation to $\mathcal{L}\phi$. By choosing $\phi(r) = r$, we guarantee a unique solution. Because n_s is typically larger than n_d, the system (38) can be considerably larger than in (30). Choosing $\phi(r) = r$ results in better conditioning than higher-order polyharmonic splines. We obtain weights that, when used in a quadrature rule, converge at 3rd order in the sample site spacing. It is possible to improve the order of the quadrature weights by increasing the order of the polyharmonic spline RBF at the potential cost of poorer conditioning and either loss of invertibility or requiring knowledge of higher-order moments [50].

5.3.2. Computing quadrature weights on Γ

Once the quadrature weights ω_j, $j = 1, \dots, n_s$ are computed, it is straightforward to compute the corresponding quadrature weights W_j, $j = 1, \dots, n_s$ for integration at the spreading points $\boldsymbol{X}^{\mathrm{s}} \in \Gamma(t)$. Recall that Γ has parametrization $\boldsymbol{X}(\theta, \varphi, t)$ and Jacobian $J(\theta, \varphi, t)$ for a given time t. This implies that the point $\boldsymbol{X}(\theta, \varphi, t)$ directly corresponds to to $\chi(\theta, \varphi)$. Using this fact and the fact that the determinant of the Jacobian for the spherical coordinate mapping (for radius 1) is $\cos\varphi$, we can use a change of variables to express the infinitesimal area $\mathrm{d}\boldsymbol{X}$ on Γ as

$$\mathrm{d}\boldsymbol{X} = J(\theta, \varphi, t)\,\mathrm{d}\boldsymbol{\theta} = J(\theta, \varphi, t) \sec\varphi\,\mathrm{d}\chi, \tag{39}$$

where $\mathrm{d}\boldsymbol{\theta}$ is an infinitesimal area in parameter space. Note that the weights ω_j above are discrete analogs of $\mathrm{d}\chi$ at χ_j^{s}. The discrete analog of $\mathrm{d}\boldsymbol{\theta}$ at the j^{th} sample site,

$$\sigma_j = \sec\varphi_j \omega_j, \tag{40}$$

can therefore be computed at the outset of a simulation. To avoid numerical issues, we require that $\cos\varphi_j \neq 0$ for each sample site. This is true everywhere on $\mathbb{S}^2$ except the poles, $(0, 0, \pm 1)$, which the Bauer spiral (25) conveniently avoids. Finally, the quadrature weight W_j for Γ at the spreading point $\boldsymbol{X}_j^{\mathrm{s}}$ is given by

$$W_j = \sigma_j J_j, \tag{41}$$

where $J_j = J(\theta_j, \varphi_j, t)$. Computing W_j given σ_j and J_j amounts to a single multiplication, which can be done trivially in parallel. This produces a set of (time-varying) weights W_j, $j = 1, \dots, n_s$ that allows us to integrate functions on

$\Gamma(t)$ when their samples are given at $\boldsymbol{X}^s_j$, $j = 1, \ldots, n_s$, as required by (6). Of interest are the tension, bending, and dissipative surface forces, for which the integral over the surface of the cell is expected to vanish. We observed that these quadrature weights recover this property to within machine precision.

6. Results

We have taken a number of departures from the traditional application of the IB method and RBC models. In the following sections, we first motivate some of our modeling choices based on observed numerical instabilities, then establish convergence for our implementation of RBF-IB, demonstrate its energy dissipation properties, demonstrate impermeability of the RBCs, and observe typical RBC behaviors before presenting the results of our whole blood simulations.

6.1. Observations on numerical instabilities

As discussed previously, we model RBCs using more spreading points than movement points with the aim to reduce the memory footprint of the discrete differential operators and to reduce the number of points needed to move the object. However, this choice can potentially create numerical instabilities unless parameters are chosen carefully.

To motivate the parameter choices described in the sections that follow, we now discuss two numerical instabilities encountered in modeling RBCs: intersections between RBCs and membrane wrinkling. These instabilities stem from having a different number of movement and spreading points, and the deformability of the RBCs. In what follows, we use the same number of movement and spreading points to model approximately rigid platelets and the endothelium. For that reason, those cells do not exhibit these instabilities and this section is limited to discussion of RBCs.

6.1.1. Preventing instability from RBC intersections

In the standard IB method, the typical heuristic is that IB points be placed such that neighboring points are approximately within h of each other. As a result, those neighboring IB points will always experience similar fluid velocities due to their proximity and the smoothing properties of the discrete delta function. This is crucial in maintaining distinctness between cells in a simulation, as IB points sufficiently close to one another will remain close together; the IB method will effectively treat them as belonging to the same object.

We use this heuristic to guide our choice of n_s, and choose fewer movement points. With progressively fewer movement points, nearby movement points encounter increasingly different velocities. Additionally, in shear flow, RBCs will elongate in the flow direction, increasing the distance between neighboring movement points in the flow direction. If the shear rate is high enough, the RBCs “tank-tread,” in which the membrane rotates around the approximately fixed shape of the cell interior. Movement points on RBCs that are interacting with one another may pass very close to one another. If nearby movement points belonging to separate RBCs experience different enough velocities, they may gradually move closer to one another until the cells have “fused” from the perspective of the IB method, causing these cells to no longer be able to separate themselves, which in our experience leads to protrusions on both cells and an explosion of forces as they attempt to remain connected. In an attempt to remedy this issue, we limit ourselves to $4n_d \geq n_s$ so that movement points are roughly half as dense as spreading points on the surface of the reference RBC. For the simulations presented below, this appears sufficient to prevent failure by intersection of RBCs.

6.1.2. *Preventing instability from wrinkling*

The other numerical instability we encountered manifested as wrinkles on the surface of one or more RBC as the cells jockey for position. The wrinkles are localized and high-frequency, resulting in surface forces that change direction frequently. The smoothing of the discrete delta function severely damps these forces, or any potential restorative forces. Ultimately, the simulation fails as the wrinkles develop into a large and unphysical protrusions. These protrusions extend into an unoccupied region of the domain, so it does not seem to be caused by the IB machinery. Though the cause of this instability is unclear, we hypothesize that it is observed in the RBF-IB method because our high-order surface representation does not introduce sufficient numerical dissipation; in contrast, simulations of RBCs with the standard IB method have, to the best of our knowledge, only used low-order (piecewise linear) surface representations, which are more likely to introduce additional numerical dissipation.

Regardless, we observed that this wrinkling instability correlates with relatively large deformations happening in a short time, but at different rates for nearby points. To prevent regions from deforming much quicker than their neighbors, we opted to add a very small dissipative force (23) to RBCs. This seems to eliminate the instability, but is atypical of RBC models used in conjunction with the standard IB method. We therefore validate this modified model for the RBF-IB method using commonly reported RBC behaviors in Section 6.4. We leave a full study of dissipation caused by surface representations to future work.

6.2. *Convergence study*

In this section, we perform a series of tests on a single perturbed RBC undergoing relaxation. We simplify the RBC model and use only Skalak's Law (18). We expect the IB method to approximate the fluid velocity at first order for thin shells, as it cannot recover the pressure jump across the interface. We stretch the RBC by a factor of 1.1 in the z direction and compress it in the x direction to maintain its reference volume. We place the cell in the center of a $16\,\mu\text{m} \times 16\,\mu\text{m} \times 16\,\mu\text{m}$ domain with homogeneous Dirichlet boundary conditions in the y direction and periodic boundaries elsewhere. The fluid velocity is initially zero. The cell is then allowed to relax for $180\,\mu\text{s}$.

We use the 3-point kernel $\hat{\delta}_1$ derived by Roma *et al.* [51] for spreading and interpolation and the 2-stage RK method described in Section 2 to advance the fluid velocity. Fluid grids are chosen to have $20r$ grid points per $16\,\mu\text{m}$ in each direction for grid index r from 1 to 6. To compare fluid velocities and cell surface positions on successive grids, we interpolate the fluid velocities to cell centers in a regular grid of 20^3 cells (i.e., at the cell centers of the grid corresponding to $r = 1$) using cubic splines and surface positions to 1000 surface points using an interpolation matrix constructed as described in Section 5.2 for the identity function. Because grids are not refined by a factor of 2, we do not expect to see an improvement by a factor of 2 for 1st order convergence. For convergence of order p, we expect

Table 1: Convergence of $\boldsymbol{u}$ for a sequence of grids. The refinement ratio r, defined as the refinement in h relative to the coarsest grid, determines the simulation parameters: $rh = 0.8\,\mu\text{m}$ and $rk = 180\,\text{ns}$. Differences are computed between grids r and $r+1$. Values of $\boldsymbol{u}$ are sampled at $t = 180\,\mu\text{s}$ at cell centers on the coarsest grid, $r = 1$.

r	L^2 error	order	L^∞ error	order
1	1.74592×10^{-3}		1.59995×10^{-2}	
2	9.92788×10^{-5}		6.52038×10^{-4}	
3	3.65264×10^{-5}	6.85983	3.34322×10^{-4}	1.06075
4	2.31069×10^{-5}	2.87612	2.30732×10^{-4}	1.60689
5	1.65898×10^{-5}	1.25668	2.28193×10^{-4}	0.83030

the ratio of successive differences in the quantity under investigation, $\epsilon_r/\epsilon_{r+1}$, for the grids r and $r+1$, to satisfy

$$\frac{\epsilon_r}{\epsilon_{r+1}} = \left|\frac{((r+1)/r)^p - 1}{((r+2)/(r+1))^{-p} - 1}\right|, \tag{42}$$

which we solve numerically to determine p. We compute the differences in $\boldsymbol{X}$ using discrete versions of the L^2 and L^∞ norms,

$$\|\boldsymbol{X}(\theta,\varphi)\|_2^2 = \int_{\mathbb{S}^2} \boldsymbol{X}(\theta,\varphi)\cdot\boldsymbol{X}(\theta,\varphi)\,\mathrm{d}\boldsymbol{\theta} \quad \text{and} \tag{43}$$

$$\|\boldsymbol{X}(\theta,\varphi)\|_\infty^2 = \max_{(\theta,\varphi)} \boldsymbol{X}(\theta,\varphi)\cdot\boldsymbol{X}(\theta,\varphi), \tag{44}$$

respectively. Equation (43) is discretized over $\mathbb{S}^2$ using the spherical quadrature weights (40). In all cases, we refer to these differences as "errors," though this is not strictly true.

Tables 1 and 2 show L^2 and L^∞ errors in $\boldsymbol{u}$ and $\boldsymbol{X}$ between successive grids. We observe first-order convergence, as expected.

6.3. Energy Estimates

As mentioned in Section 2.2, the RBF-IB method uses different sets of Lagrangian points for spreading forces and updating immersed structures. This is in contrast with traditional IB methods, for which the interpolation and spreading operators are adjoint, and energy is conserved [38]. To show that the RBF-IB method is similarly suitable for simulating the deformation of elastic bodies, we now track the total energy of the combined fluid-RBC system in the relaxation test just described. We aim to establish that the energy from the deformation of the RBC is indeed damped out, as it should be, analogous to a test performed in the 2D version of the RBF-IB method. This manifests

Table 2: Convergence of $\boldsymbol{X}$ for a sequence of grids. The refinement ratio r, defined as the refinement in h relative to the coarsest grid, determines the simulation parameters: $rk = 180\,\text{ns}$, $n_d = 125r^2$, and $n_s = 500r^2$. Differences (errors) are computed between grids r and $r + 1$. Values of $\boldsymbol{X}$ are sampled at $t = 180\,\mu\text{s}$ at 1000 Bauer spiral points.

r	L^2 error	order	L^∞ error	order
1	6.05447×10^{-3}		2.68249×10^{-3}	
2	1.61678×10^{-3}		6.86140×10^{-4}	
3	7.69150×10^{-4}	2.74664	3.30734×10^{-4}	2.86457
4	4.66203×10^{-4}	1.89474	1.86139×10^{-4}	1.84435
5	2.82578×10^{-4}	1.46574	1.18713×10^{-4}	1.82742

as a decrease in total energy over time. This energy is given as

$$E = \int_\Omega \left[\frac{\rho}{2}\boldsymbol{u} \cdot \boldsymbol{u} + \int_\Gamma \delta(\boldsymbol{x} - \boldsymbol{X})W_{\text{Sk}}(\boldsymbol{X}, \ldots)\,\mathrm{d}\boldsymbol{X}\right] \mathrm{d}\boldsymbol{x}. \tag{45}$$

The first term corresponds to the energy of the fluid, and the second to the energy of the structure. Since the relaxation test involves an initial increase in kinetic energy of the fluid followed by a gradual dissipation of the potential energy of the RBC as it relaxes according to the Skalak law, we expect the total energy of the system to decrease over time. We plot a discretized version of the energy (45) as a function of time in Figure 2 for refinement factors $r = 5$ and $r = 6$. Figure 2 clearly shows that our method dissipates energy in the manner expected in this problem. We also observed that the RBF-IB method was stable for representing RBCs and platelets in whole blood simulations.

6.4. RBC tumbling and tank-treading

Few RBC models include viscoelastic forces. Fedosov *et al.* [52] use a particle-based method to simulate the fluid and cells, where particles representing the RBC membrane experience drag and random forces from relative motion between RBC particles. Gounley & Peng [53] use the IB machinery to spread membrane viscosity to the fluid, thereby modifying the fluid stress term. Our intent in adding a viscoelastic response is to aid in the numerical stability of the discrete RBCs. We wish to verify that the extended model, with dissipative force, retains the ability to tumble and tank-tread, which is commonly used to validate other RBC models [6, 54–56]. To that end, we place a single RBC with $n_d = 625$ and $n_s = 2500$ in the same domain as the previous section, now discretized to have $h = 0.4\,\mu\text{m}$ and with moving top and bottom walls. In the interest of reducing simulation time, we now use the backward-forward Euler timestepping scheme with a time step of $k = 0.1\,\mu\text{s}$. Here, the IB interaction operations use the 4-point B-spline, $\hat{\delta}_1(r) = B_3(r)$, which was first considered by Lee [57]. It is similar in shape to the Roma kernel but has better

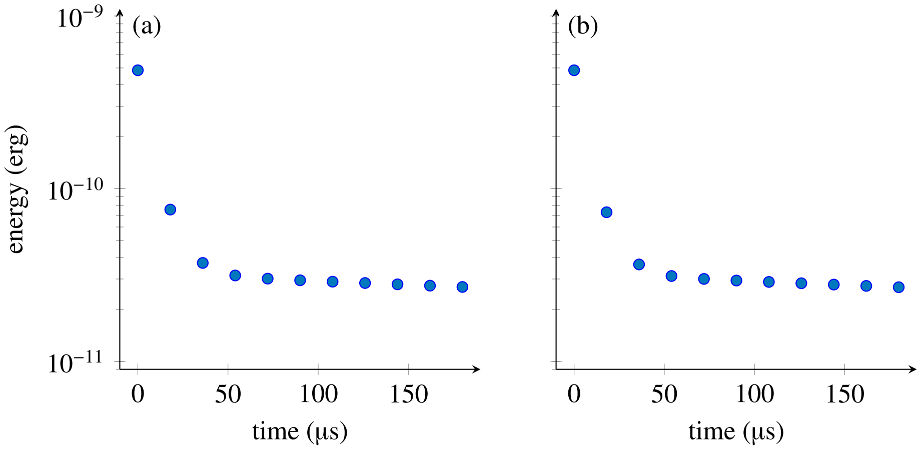


Figure 2: Energy (45) as a function of time for the relaxing RBC test of Section 6.2 with refinement factors (a) $r = 5$ and (b) $r = 6$. The refinement factor determines the simulation parameters: spacestep $rh = 0.8\,\mu\mathrm{m}$, timestep $rk = 180\,\mathrm{ns}$, $n_d = 125r^2$, and $n_s = 500r^2$.

smoothness properties. To recover the tumbling motions, the top wall has a fixed velocity of $\boldsymbol{u}_b = 400\,\mu\mathrm{m\,s}^{-1}$ and the bottom wall $-\boldsymbol{u}_b$. This generates a shear rate of $\dot{\gamma} = 50\,\mathrm{s}^{-1}$ in the absence of cells. For tank-treading experiments, we use $\boldsymbol{u}_b = 8\,\mathrm{mm\,s}^{-1}$ to generate a shear rate of $\dot{\gamma} = 1000\,\mathrm{s}^{-1}$. These values are chosen outside the transitional region between tumbling and tank-treading for the elastic parameters used for the RBC [58]. The velocity field is initially steady for flow without cells. We rotate the cell 1 radian about the x-axis from a horizontally aligned orientation and place it at the center of the domain. The RBC exhibits tumbling for $\dot{\gamma} = 50\,\mathrm{s}^{-1}$ and tank-treading for $\dot{\gamma} = 1000\,\mathrm{s}^{-1}$. Figure 3 shows one period of each.

6.5. Collision tests

With whole blood simulation as the ultimate goal, we must ensure that the method can effectively capture cell-cell interactions. The RBF-IB method has been applied to flow around multiple platelets in an aggregate in 2D simulations, but those cells were kept apart by a network of springs [36]. To study the interaction between cells, we devise a series of tests in which we force two RBCs to collide. The aim is to verify that the cells remain distinct. Using too few data sites could allow the cells to come too close to one another or even to interpenetrate, as the regularized delta function δ_h then causes them to be treated as a single unit. Cells that "fuse" in this manner are problematic, generally causing the simulation to end (due to stability-related time-step restrictions) when the cells attempt to separate.

We use the same physical domain as in the previous two sections, now with $h = 0.2\,\mu\mathrm{m}$, and place two RBCs therein, each with $n_d = 2500$ and $n_s = 10000$. We use the 2-stage RK method with time step $k = 50\,\mathrm{ns}$. To interpolate velocities and spread forces, we use the 4-point cosine $\hat{\delta}_1$ [38]. The cells are placed with cell centers on the line $x = z$, $y = 8\,\mu\mathrm{m}$. They are initially separated by a gap of $4h = 0.8\,\mu\mathrm{m}$ between their convex hulls, *i.e.*, ignoring the concavities. Inspired by Crowl & Fogelson [13], we add the fictitious force density

$$\boldsymbol{F}_{\mathrm{fict}} = \pm 0.1\,\mathrm{dyn/cm} \cdot (\boldsymbol{e}_1 + \boldsymbol{e}_3)/\sqrt{2}, \tag{46}$$

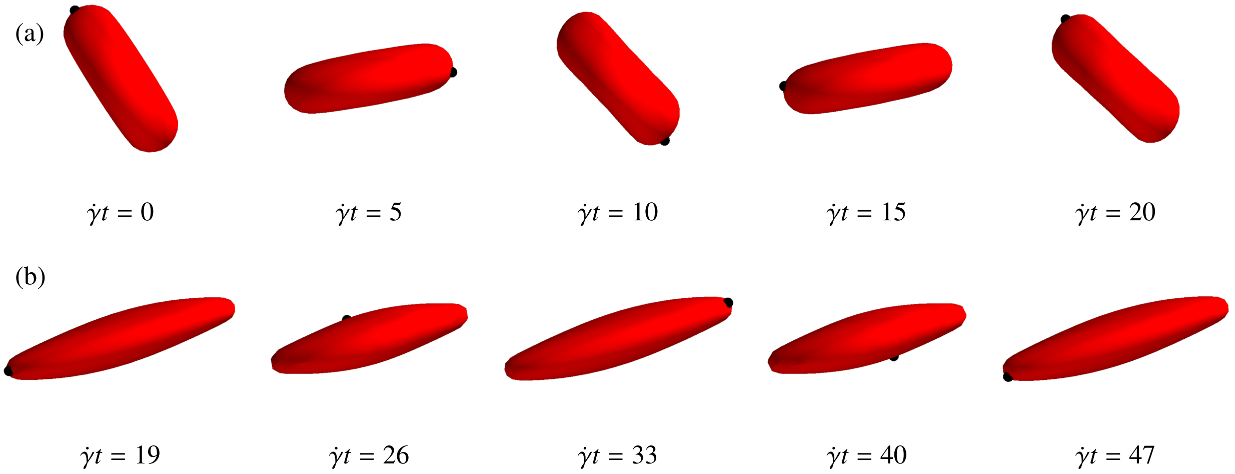


Figure 3: Our model RBC exhibits (a) a tumbling behavior under low shear ($\dot{\gamma} = 50\text{s}^{-1}$) conditions and (b) tank-treading under high shear ($\dot{\gamma} = 1000\text{s}^{-1}$) conditions. The black dot is a reference point with fixed surface coordinates.

to each cell, where the sign is chosen so the force points into the gap, to draw the cells together. Success in these tests implies that this configuration of data and sample sites, the spatial resolution, and the time step are acceptable for whole blood simulations.

Initial conditions and configurations after a short time are illustrated in Figure 4, where we view them from above the $x = z$ plane. In each case, the cells move slightly closer together and then undergo considerable deformation. The data sites are initially approximately $2h$ apart from each other. No problems seem to arise from this, and in some cases the cells eventually attempt to slide past one another. We also deduce that the IB method with the cosine kernel can resolve interactions at a distance of h to $2h$. We consider cells passing within this threshold to be in contact. Throughout the simulation, the cells remain distinct, and the simulations end due to extreme forces triggering the stopping condition [59]

$$k > \frac{1}{4}\sqrt{\frac{h\rho}{\|\boldsymbol{f}\|_\infty}}. \tag{47}$$

For the remainder of this section, we consider only this arrangement of data and sample sites and this grid resolution.

6.6. Whole blood

In what follows, we consider a $16\,\mu\text{m} \times 12\,\mu\text{m} \times 16\,\mu\text{m}$ domain with periodic boundaries in the x and z directions and with Dirichlet boundary conditions in the y direction. The fluid velocity is initially zero except at the top boundary, where it moves at $12\,\text{mm}\,\text{s}^{-1}$. In the absence of cells, the flow tends toward steady Couette flow with a shear rate of $\dot{\gamma} = 1000\,\text{s}^{-1}$. This serves as our model near-wall region of a blood vessel.

For whole blood simulations, we return to the 4-point B-spline, B_3, as the IB kernel. Because some RBC configurations generate large forces, the stopping condition of the previous section limits us to small timesteps. In the interest

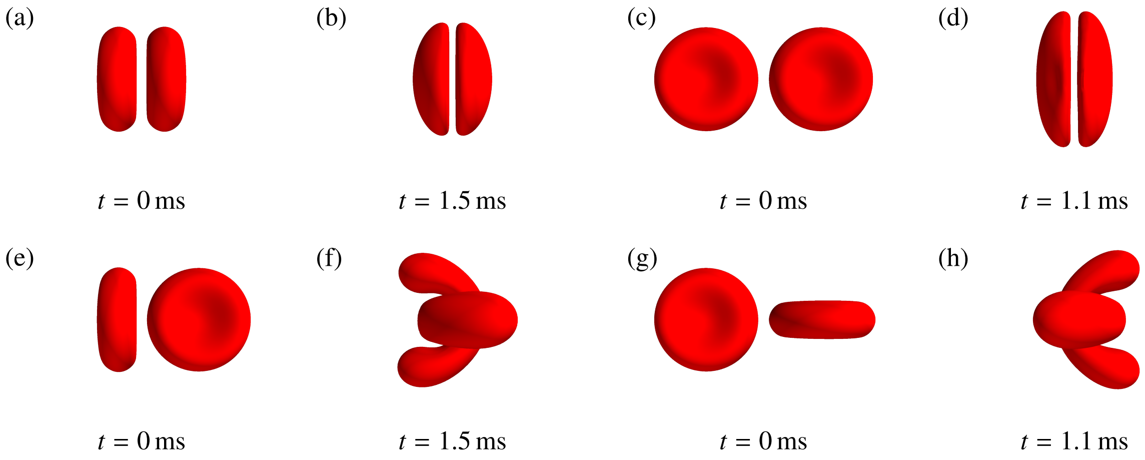


Figure 4: Collision tests between two RBCs. A fictitious force is added to the RBCs to draw them together. (a) and (b) The RBCs are initially aligned with concavities facing one another. By 1.5 ms, the cells take on a hemispherical shape. The concavities in the gap are maintained. Shortly thereafter, asymmetries in the setup lead to the cells sliding past one other. (c) and (d) The RBCs are initially aligned with their edges facing one another. By 1.1 ms, the cells take on a hemispherical shape. The remnants of the concavity can be seen on the left cell in (d). Shortly thereafter, these cells also slide past one another. (e) and (f) The RBCs are initially aligned with the edge of one facing a concavity of the other. The cells wrap around each other by 1.5 ms, taking on a bulbous banana shape. (g) and (h) The RBCs are initially aligned with their edges facing one another with one of the cells rotated about the axis $\boldsymbol{e}_1 + \boldsymbol{e}_3$ by $\pi/2$. By 1.1 ms, the cells wrap around each other, again taking on the bulbous banana shape.

of reducing simulation time, we use the backward-forward Euler scheme with $k = 50\,\mathrm{ns}$. In tests, we observe qualitative agreement between this scheme and the two-stage RK scheme. We have already settled on an RBC discretization in the previous section. We use the same spiral method to discretize the platelet, but with 900 data and sample sites. Because the platelet is considerably smaller than the RBC, 900 is a sufficient number of sample sites and using the same number of data sites is not too computationally taxing. This aligns more closely with traditional IB methods. We also find that the Bauer spiral places points more densely along the edge of the platelet, which is helpful in resolving the large curvatures there. We parametrize the surface of the endothelium over $(\theta, \varphi) \in [0, 2\pi)^2$ with reference shape

$$\hat{\boldsymbol{X}}_{\text{endo}}(\theta, \varphi) = \begin{bmatrix} 16\,\mu\mathrm{m} \cdot (\theta/2\pi) \\ y(\theta, \varphi) \\ 16\,\mu\mathrm{m} \cdot (\varphi/2\pi) \end{bmatrix}, \tag{48}$$

where $y(\theta, \varphi)$ depends on the shape under study. The endothelium is discretized using 16000 points, defined by the spiral

$$\begin{aligned} \varphi_i &= 2\pi(i-1)/N, \\ \theta_i &= \operatorname{mod}\left(\left\lceil \sqrt{N} \right\rceil \varphi_i, 2\pi\right). \end{aligned} \tag{49}$$

We consider two shapes for the endothelium. The first, $y = 1\,\mu\mathrm{m}$, emulates the flat wall typically used in near-wall simulations of RBCs or platelets. The other attempts to recreate the elongated endothelial cell shape typical of exposure to high-shear conditions,

$$y(\theta, \varphi) = 0.75\,\mu\text{m} + 1\,\mu\text{m} \cdot \cos^2(\theta - \varphi)\sin^2(\varphi/2). \tag{50}$$

The bumps have a prominence of 1 μm. The endothelial surface is raised by 0.75 μm to avoid it interacting with the domain boundary. The positions of the surface are chosen to maintain a fixed hematocrit of approximately 34% for both endothelial shapes.

As a preliminary validation of the platelet model and to establish baseline platelet motion, we consider two platelets along a flat wall. They are placed parallel to the wall at distances of 0.3 μm and 0.5 μm. The domain does not contain any RBCs. At a distance of 0.3 μm, the platelet is expected to "wobble," a motion in which the platelet tilts slightly upward and downward, periodically [60]. On the other hand, the platelet initially 0.5 μm from the wall should tumble end-over-end. These behaviors are illustrated in Figure 5. We observe wobbling at a frequency of approximately $10\,\text{s}^{-1}$ and tumbling with a frequency of approximately $30\,\text{s}^{-1}$. These figures are in reasonable agreement with other studies [60]. We also note that the edge of the tumbling platelet remains pointed towards the wall for only 3–4 ms.

6.6.1. Initialization

We assume that the platelets have already been marginated by the RBCs. We think of the domain as having three layers with the endothelium at the bottom, RBCs on top, and platelets in between. We begin by settling the endothelium and RBCs before placing platelets in the space between the RBCs and the endothelium.

In addition to the endothelium, we place 2 rows of 4 RBCs, each in their reference configuration, in the domain with the RBCs' centers of mass on the plane $y = 6\,\mu\text{m}$. Because the domain is not wide enough to accommodate two reference RBCs alongside one another, the cells are staggered by 2 μm. These locations are then randomly translated and rotated while maintaining a distance of at least $2h$ between cells.

Before placing any platelets in the domain, we allow the flow to develop with only the endothelium and RBCs. We allow the initialization to continue until at least 17 ms, which is approximately when the first RBC overtakes its neighbor. From here, we choose a series of times, sampled from a Poisson distribution to be approximately 3 ms apart, at which to begin simulations with platelets.

The RBCs for each of the chosen starting configurations are considerably and unpredictably deformed and have left a space of a few microns above the endothelium in which we place platelets. To find reasonable starting orientations for the platelets, we randomly choose points on the endothelium and one on each platelet surface. The process is illustrated in Figure 6. Each of the simulations in the upcoming sections contains two platelets, so we choose two points on the endothelium that are at least 3.9 μm, a platelet diameter plus $4h$, apart. The resulting platelets are spaced far enough apart to not intersect. We compute normal vectors on the surfaces of the endothelium and platelets at these points. We orient the platelets so that the normal emanating from the platelet opposes the normal at the corresponding point on the endothelium. The platelet is then placed so its chosen surface point is separated from the endothelium

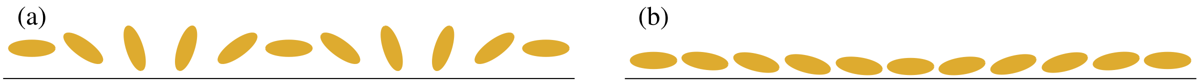


Figure 5: Snapshots of different behaviors of a rigid platelet in shear flow. The behavior of the platelet (yellow ellipse) depends on its initial distance from the wall (black line). (a) The platelet is far enough from the wall to rotate, or tumble, freely without coming into contact with the wall. At a shear rate of $\dot{\gamma} = 1000\,\mathrm{s}^{-1}$, one complete rotation takes approximately 0.03s. (b) The platelet is too near to the wall to rotate freely, and instead exhibits "wobbling", in which the platelet follows a sinusoidal trajectory as it travels over the wall, periodically dipping downward before returning to its original distance from the wall. This motion has been exaggerated here to make the motion more obvious. At a shear rate of $\dot{\gamma} = 1000\,\mathrm{s}^{-1}$, one complete wobble takes approximately 0.1s.

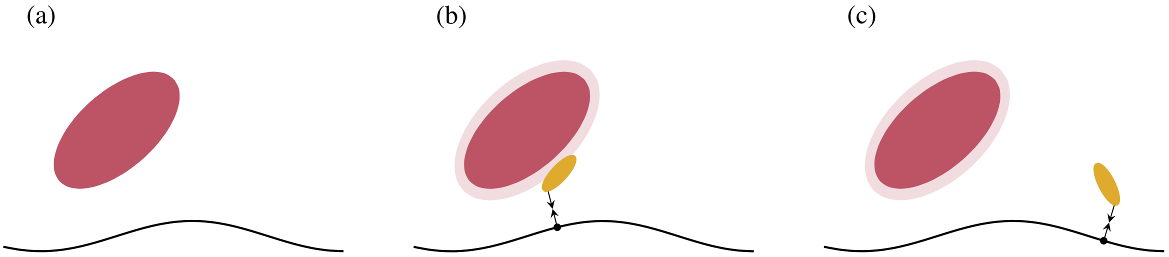


Figure 6: The initialization process. (a) The domain is initialized with RBCs (red ellipse) and the endothelium (black curve). The flow is allowed to develop for at least 17 ms and is stopped. (b) A random location is chosen on the endothelium (black dot), and the outward normal at this point is computed. A point on the platelet (yellow ellipse) is also randomly chosen and the surface normal computed at that point. The platelet is then rotated and translated so the two normals (black arrows) oppose each other and the platelet is between 0.3 µm and 1 µm removed from the endothelium. Here, the potential location is rejected because it is too close to the nearby RBC, as indicated by the pale red ellipse around the RBC. (c) The same process as (b) is repeated. This time, the location is accepted because the platelet is sufficiently removed from any other cells.

point by a random distance between 0.3 µm and 1 µm. If the generated orientation does not pass within 0.4 µm of an RBC, the platelet is accepted. Otherwise, we try again with a different platelet point. This algorithm typically succeeds within 2 attempts.

This initialization process is performed once for each endothelial configuration and we take the first four acceptable initial configurations for each. In the following section, we present behaviors found in these simulations. As a point of comparison, we also consider the same initial configurations for the bumpy wall with the RBCs removed from the domain.

6.6.2. Characterization of flow and cell behaviors

In this section, we catalog the differences in the flow between whole blood along a bumpy and flat wall, and between flow along a bumpy wall with and without RBCs. We aim to compare the interactions platelets have with RBCs and the endothelium for these test cases.

Flow profiles are shown in Figure 7. The most notable difference among the three flow profiles is the nearly Couette flow when RBCs are absent. The only distinction between this and Couette flow with $\dot{\gamma} = 1000\,\mathrm{s}^{-1}$ is the smoother transition at the wall due to the bumps, as can be seen in the inset. This is also the distinguishing feature between the profiles corresponding to bumpy and flat walls in the presence of RBCs. The smooth transition from

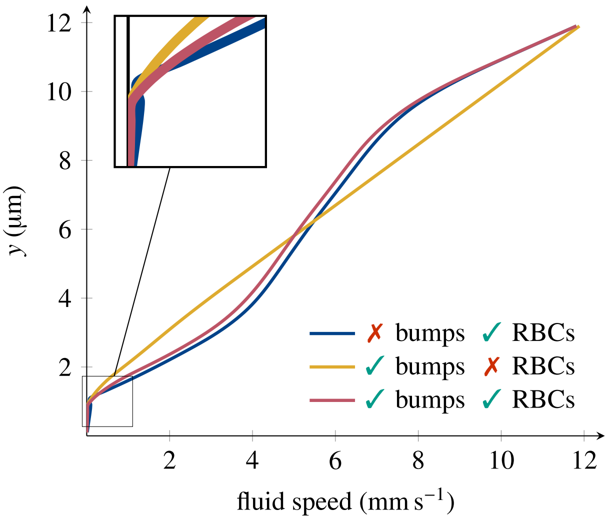
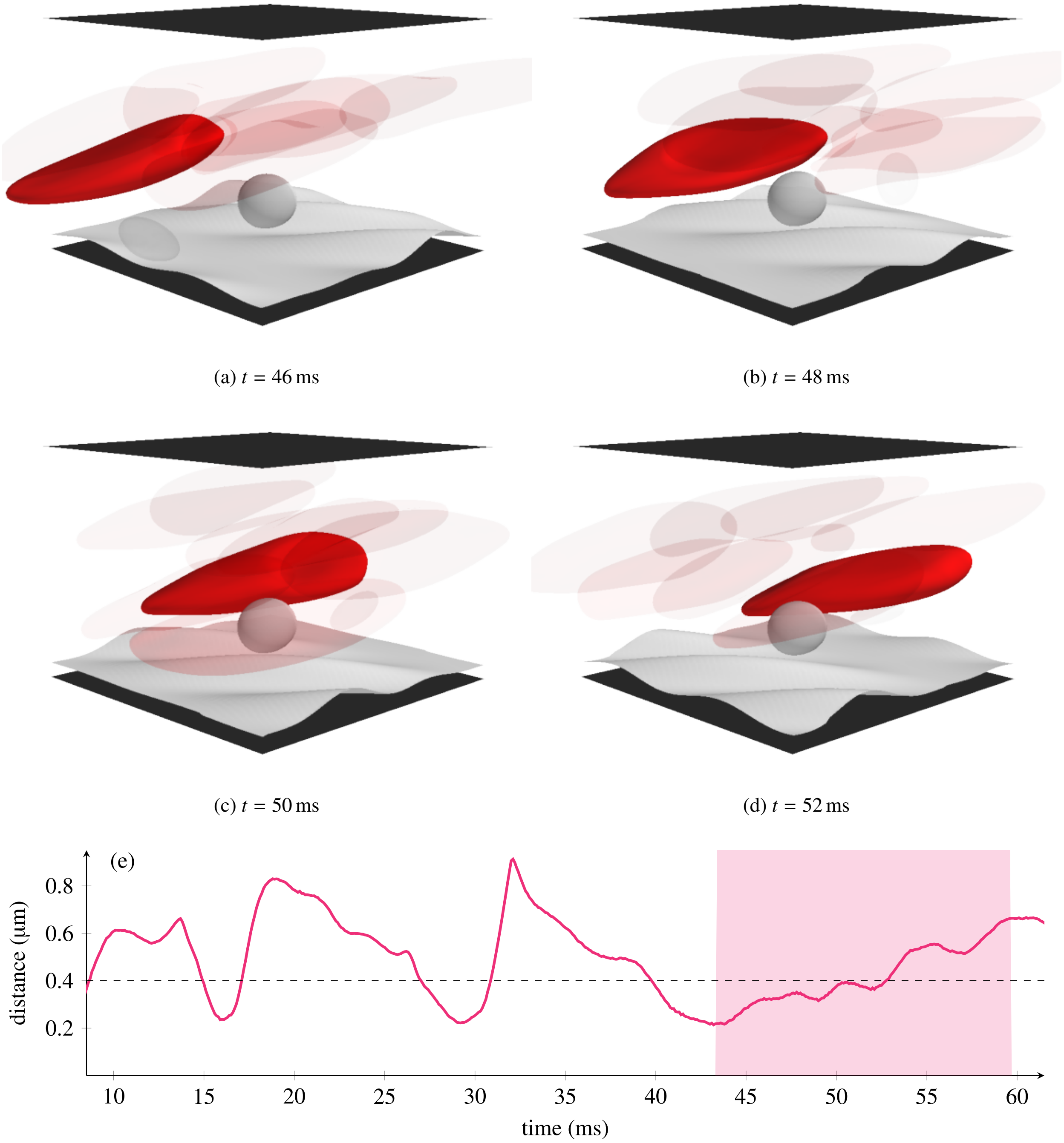


Figure 7: Time- and space-averaged fluid speed profiles for each of the test cases. The inclusion of RBCs (red and blue curves) causes the region inhabited by platelets, 1–4 µm, to experience a higher shear rate than it would without RBCs (yellow curve). Inset: Close up of the region around the endothelium. The transition across the endothelium has the sharpest change in speed for simulations without a bumpy endothelium.

the bumps results in marginally slower flow speeds throughout the domain, compared to the flat wall. The inclusion of RBCs causes the bends in the red and blue curves around $y = 3$ µm and $y = 9$ µm. Platelets located between $y = 1$ µm and $y = 3$ µm experience higher shear rates in simulations featuring RBCs than in those without RBCs. The region near the upper boundary also experiences an increased shear rate. In pressure-driven flow through a tube, we expect parabolic flow. The bend near $y = 9$ µm is therefore nonphysical and arises from satisfying boundary conditions at the top boundary. However, the increased shear rate in the region between $y = 9$ µm and 12 µm seems to be useful in deterring RBCs from approaching the upper boundary. This region acts as another RBC-free layer where the RBCs excurse only infrequently. An exclusionary region of just 1–2 µm along the top boundary increases the effective hematocrit to 37–41%. Furthermore, the reduced shear rate in the region containing RBCs results in slower tank-treading compared to a dilute suspension of RBCs, with one period now lasting approximately 40 ms.

RBCs are effective at preventing the platelets from moving too far from the endothelium. The furthest observed distance from the endothelium any platelet takes is just under 1.5 µm. Likewise, RBCs infrequently enter the RBC-free layer, with some notable exceptions, discussed below. We do not observe any platelet wobbling. Instead, platelets transiently follow the curve of the bumpy walls, tilt down into the valleys between bumps, and tumble. Nothing suggests that bumps in the surface of the endothelium alone can sequester platelets, nor do we directly observe stagnation zones.

Bumpy endothelium simulations without RBCs mimic those with a flat wall; platelets move away from the wall to a point where they are free to tumble. Unsurprisingly, we observe platelet tumbling for both flat and bumpy walls with RBCs as well. In Stokes-like flow, a rigid platelet would tumble faster in flow with a higher shear rate. We might therefore expect the platelet to tumble faster with RBCs. However, RBCs can significantly disturb the fluid around a


(a) t = 46 ms

(b) t = 48 ms

(c) t = 50 ms

(d) t = 52 ms

Figure 8: (a)–(d) Snapshots of a platelet rolling on its edge ("unicycling") with RBCs, one translucent, flanking either side. The camera tracks the opaque platelet. The platelet's motion is indicated by the endothelium moving from right to left. (e) The distance between the platelet and the endothelium. The shaded region indicates that the orientation of the platelet's short axis is within 45° of the vorticity direction. The black dashed line indicates $2h$ and is the maximum distance that might be considered contact with the endothelium. See AppendixC for a video corresponding to this simulation.

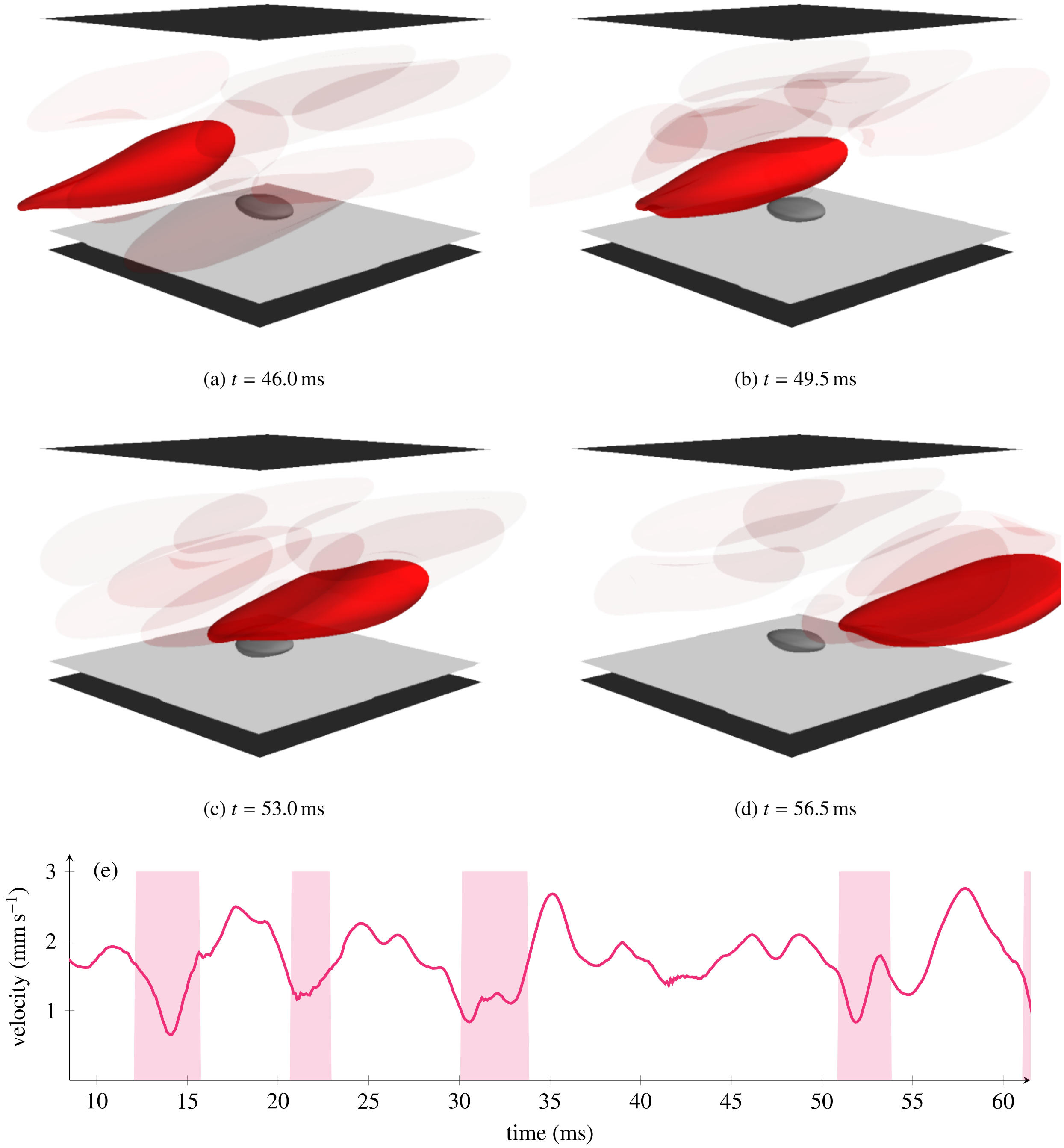
platelet, speeding up its motion, slowing it down, or preventing a tumble altogether.

We observe platelets rolling in the flow direction along their edge. Because the platelet in this arrangement is aligned vertically, part of the edge stays in near-contact with the endothelium while the opposite edge extends into the region occupied by RBCs. Contact with RBCs is frequent. These contacts can have a destabilizing effect, but may also prolong the rolling. Figure 8(a)–(d) consists of a series of snapshots illustrating the behavior. The platelet in this case is flanked by two RBCs, so it does not have the space to topple over until the RBCs pass a few milliseconds later.

While Figure 8 shows this phenomenon on a bumpy wall, it can also occur above a flat wall. We first observed this motion with a flat wall, and there it lasted over 40 ms. The motion was maintained, in part, by an RBC that rode along the top of the platelet, partially enveloping that platelet. We say that a platelet rolling on its edge is *unicycling*. Figure 8(e) illustrates that the platelet spends more time in contact or near-contact with the endothelium while unicycling compared to the tumbles near $t = 16$ ms and $t = 29$ ms. Though RBCs seem to control the duration of the unicycling, they are not strictly necessary for unicycling to occur. In tests with a bumpy wall without RBCs, unicycling is initiated when a platelet rolls sideways, relative to the flow direction, off of a bump. Without RBCs, the platelet maintains the vertical alignment for a majority of the simulation thereafter. However, without frequent interaction with RBCs, the platelet in these simulations move away from the wall. Moreover, while we have not observed it directly, we expect that a lone platelet traveling over a flat wall would also exhibit unicycling, given the right initial orientation.

We notice that in simulations with a bumpy endothelium, platelets collide with the bumps. This tends to occur while the platelet is tumbling, and the edge of the platelet makes contact with the endothelium. This interaction is characterized by deformations that flatten the edge of the platelet and a ~5% relative change in the aspect ratio of the major axes of the platelet. However, the collision need not occur along the edge of the platelet, nor, indeed, against a bump in the endothelium. Somewhat surprisingly, collisions with a flat wall occur at roughly the same frequency, suggesting that RBCs mediate this behavior. A clear case of this is illustrated in Figure 9(a)–(d). We also note that the few milliseconds preceding the unicycling in Figure 8 correspond to a collision with the wall, showing that this is yet another trigger for unicycling to occur.

Because the platelet comes into contact with the endothelium, or nearly does so, the platelet slows along the area of contact. Figure 9(e) shows the correlation between the relative change in aspect ratio and the reduction in minimum platelet surface velocity. Though the aspect ratio of the platelet changes somewhat while normally tumbling, changes of 5% or greater seem to always correspond to interactions with the endothelium. Collisions with the RBC, for example, result primarily in deformation of the RBC and deflection of the platelet, which is otherwise relatively unperturbed.



(a) t = 46.0 ms

(b) t = 49.5 ms

(c) t = 53.0 ms

(d) t = 56.5 ms

Figure 9: (a)–(d) Snapshots of RBC-mediated collision between a platelet and the endothelium. (a) The platelet attempts to tumble. (b) and (c) An RBC comes into proximity with the platelet, deflects to avoid the platelet, and pushes the platelet into the endothelium, thereby preventing the platelet from tumbling. (d) The platelet is free to tumble again. (e) The minimum velocity on the surface of the platelet. The shaded region indicates that the relative change in aspect ratio of the major axes exceeds 4%. See Appendix AppendixC for a video corresponding to this simulation.

7. Discussion

In this article, we developed a coherent numerical framework based on the RBF-IB method for whole blood simulation. We have shown that a continuous energy RBF-based model RBC incorporating dissipative forces exhibits the traditional tumbling and tank-treading behaviors. We simulated the flow of whole blood involving RBCs and platelets over a model endothelium. We considered both flat and bumpy endothelial shapes and flow with and without RBCs along a bumpy wall.

The most prominent result of the whole blood simulations is that simulations involving platelets but neglecting the influence of RBCs cannot capture the true nature of platelet motion. In fact, platelets near the endothelium experience augmented shear rates due to RBCs and RBCs act to confine platelets to the cell-free layer. Simulations without RBCs fail to capture the more irregular aspects of platelet motion. In fact, our simulations show that interaction with RBCs can disturb otherwise regular wobbling motions exhibited by platelets, and also appear to delay tumbling. Our results also demonstrate that certain platelet behaviors can only be observed by considering numerous starting configurations in developed flows involving RBCs.

Another prominent finding is that the effect of RBCs generally overwhelms the effects of the wall topography. Figure 7 shows that the flat wall yields slightly faster fluid velocities, but the flow profiles for flat and bumpy walls are qualitatively the same, except for the immediate vicinity of the wall. The result is a region of space between the bumps with a velocity gradient. Platelets following the shape of the bumpy wall crest the bump, dip into the region of lower velocity, and tumble. This feature is absent from the flat wall, but tumbling is not an extraordinary behavior. In either case, we observe "unicycling," a unique behavior in which the platelet rolls in the flow direction along its edge. Unicycling can be stabilized by RBCs which flank the platelet, or by an RBC that partially encapsulates the platelet while passing over it. Conversely, it can also be destabilized by an RBC passing on one side. The endothelial protrusions are also sufficient to orient a platelet while unicycling, but without RBCs to confine the platelet near the wall, the platelet does not roll along the wall for long. However, we find that unicycling itself seems to be stable. Unicycling also highlights the need for 3D simulations—it is a behavior that cannot be captured by or predicted from a 2D simulation. We also observe platelet-endothelial interactions for both endothelial shapes. These interactions are typically caused by RBCs driving the platelet into the endothelium. The collisions are characterized by significant deformation to the platelet and a reduction in its speed. From a qualitative standpoint, the endothelial shape alone has minimal impact on the motion of the platelets, meaning that for modeling flow of whole blood above a healthy blood vessel, a flat wall suffices. However, obtaining sub-micron level resolution images at sufficient temporal resolution at realistic flow rates to validate our findings is currently impossible *in vivo*.

It is also reasonable to consider an alternative interpretation of the flat wall: a model exposed subendothelium. Near contact with the subendothelium increases the chance of platelet activation. Unicycling keeps an edge of the platelet near the wall, without hindering mobility. We see that the vertical alignment is often maintained much longer than wall contacts from tumbling, and we do not observe wobbling in the presence of RBCs. We propose unicycling as an effective means by which platelets survey the vasculature for injury. Of course, the platelet can only distinguish a healthy vessel from an injury by encountering the necessary chemical signals. Until then, the platelets unicycle around bumps along the healthy endothelium as well. This also implies that unicycling indirectly assists in platelet activation for these shear rates.

We also consider yet another alternative interpretation of the bumpy wall. Because the bumps are approximately the same size as a platelet, we can consider this to be a rough model of a subendothelium with a few deposited platelets. Under this interpretation, we view platelet-wall contact as interactions between an unactivated platelet and the subendothelium, when contact occurs in a valley, or between an unactivated platelet and an activated platelet adhering to the subendothelium, when contact occurs on a bump. These interactions correlate with a reduction in velocity at the contact zone and the platelet membrane becomes flattened at the point of contact. We suggest that the decreased velocity may be sufficient to allow bonds to form between the platelets or between the platelet and subendothelium. By flattening, the platelet exposes more surface area at the point of contact, so that the activation signals are more likely to reach the platelet. The observed velocity reduction seems insufficient for this, but the resolution of our simulations is also unlikely to allow cells to pass within bonding distance of one another. Overcoming this limitation we leave as a future direction.

AppendixA. Boundary error correction for staggered grids

The marker-and-cell (MAC) grid [40] is a popular method for fluid simulations. Components of vector-valued quantities are discretized at the center of the corresponding cell faces and scalar-valued quantities at the cell center. This staggering is a distinguishing feature of the MAC grid. Staggering avoids the checkerboard instability that arises from using collocated grids [61]. However, in domains with non-periodic boundaries, this means that some vector components will encounter situations where satisfying boundary conditions with linear ghost value extrapolation leads to numerical error. This appendix explores these errors and provides a resolution that maintains compatibility with the conjugate gradients method.

For a rectangular domain with Dirichlet boundary conditions along its top and bottom and periodic boundaries elsewhere, consider a region of the domain adjacent to a boundary. The horizontal component of the fluid velocity, u, is discretized at locations staggered $h/2$ vertically above the bottom of a grid cell, or below the top of a grid cell, as seen in Figure 1(b). We have boundary data for u along the boundary at the same x and z coordinates as the grid points. Using a linear interpolant to fill a value at the ghost cell $h/2$ below the bottom boundary yields

$$u_{i,-1,k} = 2\gamma_{i,k} - u_{i,0,k}, \tag{A.1}$$

where $u_{i,-1,k}$ is the value at the ghost cell, $\gamma_{i,k}$ is the boundary datum, and $u_{i,1,k}$ is the value at the grid point within the domain. Using the standard 7-point Laplacian then yields

$$
\begin{aligned}
(\Delta_h \vec{u})_{i,0,k} &= (D_{xx}\vec{u})_{i,0,k} + h^{-2}\left[u_{i,-1,k} - 2u_{i,0,k} + u_{i,1,k}\right] + (D_{zz}\vec{u})_{i,0,k} \\
&= (D_{xx}\vec{u})_{i,0,k} + h^{-2}\left[2\gamma_{i,k} - 3u_{i,0,k} + u_{i,1,k}\right] + (D_{zz}\vec{u})_{i,0,k},
\end{aligned}
\tag{A.2}
$$

where $\vec{u}$ is the vector of $u_{i,j,k}$ values and D_{xx} and D_{zz} are the standard 3-point discrete second derivative operators with respect to x and z, respectively. Because $D_{xx}\vec{u}$ and $D_{zz}\vec{u}$ only involve values at points within the domain, they are known to approximate their continuous counterparts to second order. We therefore focus on the remaining term. Replacing $u_{i,j,k}$ with $u(hi, h(j+0.5), h(k+0.5))$ and Taylor expanding about $(hi, 0.5h, h(k+0.5))$ yields

$$
\begin{aligned}
h^{-2}\left[2\gamma_{i,k} - 3u_{i,0,k} + u_{i,1,k}\right] = h^{-2}\Bigg[2\left(u - \frac{h}{2}u_y + \frac{h^2}{8}u_{yy} - \frac{h^3}{48}u_{yyy}\right) - 3u \\
+ \left(u + hu_y + \frac{h^2}{2}u_{yy} - \frac{h^3}{6}u_{yyy}\right) + O(h^4)\Bigg] \\
= \frac{3}{4}u_{yy} - \frac{5h}{24}u_{yyy} + O(h^2),
\end{aligned}
\tag{A.3}
$$

i.e., the leading coefficient is $3/4$ where we expect 1, meaning we obtain a 0^{th}-order approximation to u_{yy} near the boundary. The case is the same for the upper boundary.

Correcting this while maintaining symmetry of the Δ_h operator is not as easy as using a second order interpolant for the ghost cell value or scaling the stencil for D_{yy} near the boundary by $4/3$. Either of these options changes the weight of $u_{i,1,k}$, which destroys symmetry. Instead, we scale any equation involving a ghost cell by $3/4$, excluding the bracketed terms in (A.2). We define the modified identity operator $\tilde{I}$, which has a value of $3/4$ on the diagonal for rows corresponding to near-boundary equations and 1 elsewhere on the diagonal. We define the modified discrete Laplacian

$$
\tilde{\Delta}_h = \tilde{I}D_{xx} + D_{yy} + \tilde{I}D_{zz},
\tag{A.4}
$$

where, away from the boundary, D_{yy} is the standard 3-point discrete second derivative with respect to y, and defined according to (A.2) otherwise. The resulting D_{yy} is symmetric. In addition to replacing the discrete Laplacian Δ_h with $\tilde{\Delta}_h$, a timestepping scheme using this correction must replace instances of the identity matrix, many of which are not written explicity, with the modified identity $\tilde{I}$. A stage of such a timestepping scheme takes the form

$$
(\tilde{I} - \alpha k\tilde{\Delta}_h)\vec{u}^n = (\tilde{I} + (1-\alpha)k\tilde{\Delta}_h)\vec{u}^{n-1} + B_h\vec{\gamma} + \tilde{I}\vec{f},
$$

where α is either 0.5 or 1, depending on the scheme and stage, and B_h is an appropriate operator that modifies equations near the boundary with boundary data. This corrected scheme involves modified operators, but requires only an additional diagonal multiplication compared to the uncorrected scheme. The Helmholtz operator on the left-hand side remains symmetric positive definite after correction, and is therefore suitable for solution via conjugate gradients. With this correction, the fluid solver converges at second order in space and time for sufficiently smooth initial conditions and driving forces to the *correct* steady state.

AppendixB. Force models

In this section, we list the force densities associated with each of the constitutive laws given in Section 4. For simplicity, we adopt the Einstein summation notation. A Greek letter featuring as a subscript and superscript within a term, *e.g.*, $a_\alpha b^\alpha$, indicates summation over $\{1, 2\}$ for that letter. We also adopt the comma notation for partial differentiation, where subscripts following a comma indicate partial differentiation with respect to the corresponding coordinates, *e.g.*, $\phi_{,\alpha\beta} = \partial^2\phi/\partial\theta^\alpha\partial\theta^\beta$. The surface coordinates θ^1 and θ^2 correspond to θ and φ of Sections 4 and 5. The exact correspondence is unimportant, as long as it is used consitently. Subscripts indicate a covariant tensor which are expressed in terms of a chosen basis, and expressing the quanitity in terms of another basis requires only application of the change-of-basis matrix. Superscripts indicate a contravariant tensor, which can be re-expressed in another basis by applying the inverse of the change-of-basis matrix. It may be helpful to think of covariant and contravariant indices as enumerating elements in coordinate and coefficient space, respectively.

We begin with Hookean and damped spring forces, whose linearized force densities together take the form

$$\boldsymbol{F}_{\text{spring}} = -k(\boldsymbol{X} - \boldsymbol{X}') - \eta(\boldsymbol{U} - \boldsymbol{U}'). \tag{B.1}$$

It is common practice to treat each spring individually, so that the quadrature weight W_j is absorbed into the coefficients: k has units of force per length and η units of force-time per length. Implementation of these forces requires no geometric information outside of positions and velocities. These are only used for the endothelium.

Next, we consider tension, which generates forces based on stretching or compressing of the elastic surface. This is somewhat more complicated than the Hookean spring case. First, we define the *metric tensor*, $g_{\alpha\beta} = \boldsymbol{X}_{,\alpha} \cdot \boldsymbol{X}_{,\beta}$ (which encodes local information about distance and area) and its inverse, $g^{\alpha\beta}$ (sometimes called the dual metric). We similarly define the metric tensor for the reference configuration, $\hat{g}_{\alpha\beta}$, and its dual. This allows us to write the Green-Lagrange strain tensor as

$$\epsilon_\alpha^\beta = \frac{1}{2}\left(g_{\alpha\mu}\hat{g}^{\mu\beta} - \delta_\alpha^\beta\right), \tag{B.2}$$

The invariants of this tensor encode information about relative changes in lengths and areas, and therefore can be used to write a tension force density. The invariants can be expressed as

$$I_1 = 2\epsilon_\mu^\mu, \tag{B.3}$$

$$I_2 = 4\epsilon + I_1, \tag{B.4}$$

where ϵ_μ^μ and ϵ are the trace and determinant, respectively, of ϵ_α^β. For Skalak's Law (18) and neo-Hookean tension (20), we first define the second Piola-Kirchhoff stress tensor using the invariants I_1 and I_2 as

$$s^{\alpha\beta} = 2\frac{\partial W}{\partial I_1}\hat{g}^{\alpha\beta} + 2I_2\frac{\partial W}{\partial I_2}g^{\alpha\beta}, \tag{B.5}$$

where $W(I_1, I_2)$ is the tension energy density function (see Section 4). This in turn allows us to define the tension force density [37]

$$\boldsymbol{F}_{\text{tension}} = \frac{1}{\sqrt{\hat{g}}}\left(\sqrt{\hat{g}}s^{\alpha\beta}\boldsymbol{X}_{,\beta}\right)_{,\alpha}, \tag{B.6}$$

where $\boldsymbol{X}_{,\beta}$ refer to the tangent vectors on the surface. Because the tension force density is expressed in relation to the reference configuration, the force is computed by multiplying by quadrature weights for the reference configuration, which do not change over the course of a simulation.

Our models also contain terms to penalize bending. First, given tangent vectors $\boldsymbol{X}_{,1}$ and $\boldsymbol{X}_{,2}$, the unit normal vector to the surface is given as

$$\boldsymbol{n} = \frac{1}{\sqrt{g}}(\boldsymbol{X}_{,1} \times \boldsymbol{X}_{,2}). \tag{B.7}$$

This in turn allows us to define the symmetric tensor $b_{\alpha\beta} = \boldsymbol{n} \cdot \boldsymbol{X}_{,\alpha\beta}$, which contains the coefficients of the second fundamental form, and the *shape tensor* $K^{\beta}_{\alpha} = b_{\alpha\mu}g^{\mu\beta}$. The principal curvatures are the eigenvalues of the shape tensor. More importantly, the trace of this tensor is twice the mean curvature $2H = K^{\mu}_{\mu}$, and its determinant K is the Gaussian curvature. We use these formulae to compute H and K for use in a standard expression for the Canham-Helfrich force density [62] to obtain the following bending force density:

$$\boldsymbol{F}_{\text{CH}} = -4\kappa\left(\Delta(H - H') + 2(H - H')(H^2 - K + HH')\right)\boldsymbol{n}, \tag{B.8}$$

where Δ is the Laplace-Beltrami operator. We can compute H and K using the formulas above, but

$$\Delta H = \frac{1}{\sqrt{g}}\left(\sqrt{g}g^{\alpha\beta}H_{,\beta}\right)_{,\alpha} \tag{B.9}$$

requires up to fourth derivatives of $\boldsymbol{X}$. In numerical simulations, we compute H pointwise and apply the discrete Laplace-Beltrami operator to obtain ΔH.

Finally, we consider dissipative forces, which cause the membrane to exhibit a viscoelastic response to strain. With surface velocity $\boldsymbol{U}$, the metric tensor changes in time according to

$$\dot{g}_{\alpha\beta} = \boldsymbol{U}_{,\alpha} \cdot \boldsymbol{X}_{,\beta} + \boldsymbol{X}_{,\alpha} \cdot \boldsymbol{U}_{,\beta}. \tag{B.10}$$

The dissipative force density takes the form [47]

$$\boldsymbol{F}_{\text{dissip}} = \frac{\nu}{\sqrt{g}}\left(\sqrt{g}g^{\alpha\mu}\dot{g}_{\mu\lambda}g^{\lambda\beta}\boldsymbol{X}_{,\beta}\right)_{,\alpha}, \tag{B.11}$$

where ν is the membrane viscosity.

In general, for all force calculations, it is possible to rewrite (B.6), (B.11), and (B.9) in terms of first and second derivatives with respect to parametric variables, thereby ensuring that we require only discrete first and second derivative operators to compute a wide variety of forces.

AppendixC. Supplementary materials

Videos for Figures 8 and 9 can be found in the online supplement.

Acknowledgments

The authors thank Dr. Robert M. Kirby for access to computing resources at SCI Institute. A. T. K., A. B., and A. L. F. acknowledge support for this project under NHLBIgrant 1U01HL143336 and NSF grants DMS-1716898and DMS-1521748. V. S. acknowledges support for this project under NSF grants CISE CCF 1714844 and DMS-1521748.

[1] P. B. Canham, The minimum energy of bending as a possible explanation of the biconcave shape of the human red blood cell, Journal of Theoretical Biology 26 (1) (1970) 61–81.

[2] R. Skalak, A. Tozeren, R. P. Zarda, S. Chien, Strain Energy Function of Red Blood Cell Membranes, Biophysical Journal 13 (1973) 245–264.

[3] E. A. Evans, R. M. Hochmuth, Membrane viscoelasticity, Biophysical Journal 16 (1976) 1–11.

[4] N. Mohandas, E. A. Evans, Mechanical Properties of the Red Cell Membrane in Relation to Molecular Structure and Genetic Defects, Annual Review of Fluid Mechanics 23 (1994) 787–818.

[5] C. Pozrikidis, Numerical Simulation of the Flow-Induced Deformation of Red Blood Cells, Annals of Biomedical Engineering 31 (2003) 1194–1205.

[6] T. G. Fai, B. E. Griffith, Y. Mori, C. S. Peskin, Immersed Boundary Method for Variable Viscosity and Variable Density Problems Using Fast Constant-Coefficient Linear Solvers I: Numerical Method and Results, SIAM Journal on Scientific Computing 35 (5) (2013) B1132–B1161.

[7] W. Wang, T. G. Diacovo, J. Chen, J. B. Freund, M. R. King, Simulation of Platelet, Thrombus and Erythrocyte Hydrodynamic Interactions in a 3D Arteriole with In Vivo Comparison, PLoS ONE 8 (10) (2013) e76949.

[8] L. C. Erickson, A. L. Fogelson, Computational model of whole blood exhibiting lateral platelet motion induced by red blood cells, International Journal for Numerical Methods in Biomedical Engineering 26 (3-4) (2010) 471–487.

[9] T. Skorczewski, L. C. Erickson, A. L. Fogelson, Platelet Motion near a Vessel Wall or Thrombus Surface in Two-Dimensional Whole Blood Simulations, Biophysical Journal 104 (8) (2013) 1764–1772.

[10] Z. Wu, Z. Xu, O. V. Kim, M. S. Alber, Three-dimensional multi-scale model of deformable platelets adhesion to vessel wall in blood flow, Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences 372 (2014) 20130380.

[11] J. H. Haga, A. J. Beaudoin, J. G. White, J. Strony, Quantification of the Passive Mechanical Properties of the Resting Platelet, Annals of Biomedical Engineering 26 (1998) 268–277.

[12] J. B. Freund, Leukocyte margination in a model microvessel, Physics of Fluids 19 (2007) 023301.

[13] L. Crowl, A. L. Fogelson, Analysis of mechanisms for platelet near-wall excess under arterial blood flow conditions, Journal of Fluid Mechanics 676 (2011) 348–375.

[14] H. Zhao, E. S. G. Shaqfeh, Shear-induced platelet margination in a microchannel, Physical Review E 83 (6) (2011) 061924.

[15] A. Kumar, M. D. Graham, Segregation by membrane rigidity in flowing binary suspensions of elastic capsules, Physical Review E 84 (6) (2011) 066316.

[16] H. Zhao, E. S. G. Shaqfeh, V. Narsimhan, Shear-induced particle migration and margination in a cellular suspension, Physics of Fluids 24 (1) (2012) 011902–011921.

[17] D. A. Fedosov, J. Fornleitner, G. Gompper, Margination of White Blood Cells in Microcapillary Flow, Physical Review Letters 108 (2012) 028104.

[18] A. Kumar, M. D. Graham, Mechanism of Margination in Confined Flows of Blood and Other Multicomponent Suspensions, Physical Review Letters 109 (10) (2012) 108102.

[19] D. A. Fedosov, H. Noguchi, G. Gompper, Multiscale Modeling of Blood Flow: From Single Cells to Blood Rheology, Biomechanics and Modeling in Mechanobiology 13 (2) (2013) 293–312.

[20] K. Müller, D. A. Fedosov, G. Gompper, Margination of micro- and nano-particles in blood flow and its effect on drug delivery, Scientific Reports 4 (2014) 4871.

[21] D. A. Fedosov, G. Gompper, White blood cell margination in microcirculation, Soft Matter 10 (2014) 2961–2970.

[22] K. Vahidkhah, S. L. Diamond, P. Bagchi, Platelet Dynamics in Three-Dimensional Simulation of Whole Blood, Biophysical Journal 106 (11) (2014) 2529–2540.

[23] K. Vahidkhah, P. Bagchi, Microparticle shape effects on margination, near-wall dynamics and adhesion in a three-dimensional simulation of red blood cell suspension, Soft Matter 11 (11) (2015) 2097–2109.

[24] M. Mehrabadi, D. N. Ku, C. K. Aidun, Effects of shear rate, confinement, and particle parameters on margination in blood flow, Physical Review E 93 (2) (2016) 455–11.

[25] A. L. Fogelson, K. B. Neeves, Fluid Mechanics of Blood Clot Formation, Annual Review of Fluid Mechanics 47 (1) (2015) 377–403.

[26] C. S. Peskin, Flow patterns around heart valves: a digital computer method for solving the equations of motion, Ph.D. thesis, Yeshiva University, New York (1972).

[27] R. P. Beyer Jr, A Computational Model of the Cochlea Using the Immersed Boundary Method, Journal of Computational Physics 98 (1990) 145–162.

[28] Y. Kim, C. S. Peskin, 2-D Parachute Simulation by the Immersed Boundary Method, SIAM Journal on Scientific Computing 28 (6) (2006) 2294–2312.

[29] R. Dillon, J. Zhuo, Using the immersed boundary method to model complex fluids-struct interaction in sperm motility, Discrete and Continuous Dynamical Systems Series B 15 (2) (2011) 343–355.

[30] G. E. Fasshauer, Meshfree Approximation Methods with MATLAB, World Scientific, 2007.

[31] R. L. Hardy, Multiquadric equations of topography and other irregular surfaces, Journal of Geophysical Research 76 (8) (1971) 1905–1915.

[32] J. C. Carr, R. K. Beatson, J. B. Cherrie, T. J. Mitchell, W. R. Fright, B. C. McCallum, T. R. Evans, Reconstruction and representation of 3D objects with radial basis functions, in: ACM SIGGRAPH, 2001, pp. 1–10.

[33] V. Shankar, G. B. Wright, A. L. Fogelson, R. M. Kirby, A study of different modeling choices for simulating platelets within the immersed boundary method, Applied Numerical Mathematics 63 (C) (2013) 58–77.

[34] V. Shankar, R. M. Kirby, A. L. Fogelson, Robust node generation for mesh-free discretizations on irregular domains and surfaces, SIAM Journal on Scientific Computing 40 (4) (2018) A2584–A2608.

[35] V. Shankar, S. D. Olson, Radial basis function (RBF)-based parametric models for closed and open curves within the method of regularized stokeslets, International Journal for Numerical Methods in Fluids 79 (6) (2015) 269–289.

[36] V. Shankar, G. B. Wright, R. M. Kirby, A. L. Fogelson, Augmenting the immersed boundary method with Radial Basis Functions (RBFs) for the modeling of platelets in hemodynamic flows, International Journal for Numerical Methods in Fluids 79 (10) (2015) 536–557.

[37] O. Maxian, A. T. Kassen, W. Strychalski, A continuous energy-based immersed boundary method for elastic shells, Journal of Computational Physics 371 (2018) 333–362.

[38] C. S. Peskin, The immersed boundary method, Acta Numerica 11 (2002) 479–517.

[39] B. E. Griffith, X. Luo, Hybrid finite difference/finite element immersed boundary method, International Journal for Numerical Methods in Biomedical Engineering 33 (2017) e2888.

[40] F. H. Harlow, J. E. Welch, Numerical Calculation of Time-Dependent Viscous Incompressible Flow of Fluid with Free Surface, Physics of Fluids 8 (12) (1965) 2182–2189.

[41] Y. Morinishi, T. S. Lund, O. V. Vasilyev, P. Moin, Fully Conservative Higher Order Finite Difference Schemes for Incompressible Flow, Journal of Computational Physics 143 (1998) 90–124.

[42] U. M. Ascher, S. J. Ruuth, R. J. Spiteri, Implicit-explicit Runge-Kutta methods for time-dependent partial differential equations, Applied Numerical Mathematics 25 (1997) 151–167.

[43] D. L. Brown, R. Cortez, M. L. Minion, Accurate Projection Methods for the Incompressible Navier–Stokes Equations, Journal of Computational Physics 168 (2) (2001) 464–499.

[44] L. C. Erickson, Blood Flow Dynamics: a Lattice Boltzmann Immersed Boundary Approach, Ph.D. thesis, University of Utah, Salt Lake City (Dec. 2010).

[45] E. A. Evans, Y.-C. Fung, Improved Measurements of Erythrocyte Geometry, Microvascular Research 4 (1972) 335–347.

[46] M. M. Frojmovic, J. G. Milton, Human Platelet Size, Shape, and Related Functions in Health and Disease, Physiological Reviews 62 (1) (1982) 185–261.

[47] P. Rangamani, A. Agrawal, K. K. Mandadapu, G. Oster, D. J. Steigmann, Interaction between surface shape and intra-surface viscous flow on lipid membranes, Biomechanics and Modeling in Mechanobiology 12 (4) (2012) 833–845.

[48] R. Bauer, Distribution of Points on a Sphere with Application to Star Catalogs, Journal of Guidance, Control, and Dynamics 23 (1) (2000) 130–137.

[49] V. Shankar, G. B. Wright, Mesh-free semi-lagrangian methods for transport on a sphere using radial basis functions, J. Comput. Phys. 366 (C) (2018) 170–190.

[50] E. J. Fuselier, T. Hangelbroek, F. J. Narcowich, J. D. Ward, G. B. Wright, Kernel based quadrature on spheres and other homogeneous spaces, Numerische Mathematik 127 (1) (2013) 57–92.

[51] A. M. Roma, C. S. Peskin, M. J. Berger, An adaptive version of the immersed boundary method, Journal of Computational Physics 153 (1999) 509–534.

[52] D. A. Fedosov, B. Caswell, G. E. Karniadakis, A Multiscale Red Blood Cell Model with Accurate Mechanics, Rheology, and Dynamics, Biophysical Journal 98 (10) (2010) 2215–2225.

[53] J. Gounley, Y. Peng, Computational Modeling of Membrane Viscosity of Red Blood Cells, Communications in Computational Physics 17 (4) (2015) 1073–1087.

[54] A. Z. K. Yazdani, R. M. Kalluri, P. Bagchi, Tank-treading and tumbling frequencies of capsules and red blood cells, Physical Review E 83 (4) (2011) 046305.

[55] T. Omori, T. Ishikawa, D. Barthès-Biesel, A. V. Salsac, Y. Imai, T. Yamaguchi, Tension of red blood cell membrane in simple shear flow, Physical Review E 86 (5) (2012) 056321–056329.

[56] Y.-Q. Xu, F.-B. Tian, Y.-L. Deng, An Efficient Red Blood Cell Model in the Frame of IB-LBM and Its Application, International Journal of Biomathematics 6 (1) (2013) 1250061.

[57] J. H. Lee, Simulating *in vitro* Models of Cardiovascular Fluid-Structure Interaction: Methods, Models, and Applications, Ph.D. thesis, University of North Carolina, Chapel Hill (2020).

[58] T. Krüger, M. Gross, D. Raabe, F. Varnik, Crossover from tumbling to tank-treading-like motion in dense simulated suspensions of red blood cells, Soft Matter 9 (37) (2013) 9008–9015.

[59] G. Agresar, J. J. Linderman, G. Tryggvason, K. G. Powell, An adaptive, Cartesian, front-tracking method for the motion, deformation and adhesion of circulating cells, Journal of Computational Physics 143 (1998) 346–380.

[60] N. A. Mody, M. R. King, Three-dimensional simulations of a platelet-shaped spheroid near a wall in shear flow, Physics of Fluids 17 (11) (2005) 113302.

[61] P. Wesseling, Principles of Computational Fluid Dynamics, Vol. 29 of Springer Series in Computational Mathematics, Springer Berlin Heidelberg, Berlin, Heidelberg, 2001.

[62] O.-Y. Zhong-can, W. Helfrich, Bending energy of vesicle membranes: General expressions for the first, second, and third variation of the shape energy and applications to spheres and cylinders, Physical Review A 39 (10) (1989) 5280–5288.