4 April 2014

Augmenting the Immersed Boundary Method with Radial Basis Functions (RBFs) for the Modeling of Platelets in Hemodynamic Flows

Varun Shankar [a,1], Grady B. Wright [b], Robert M. Kirby [a] and Aaron L. Fogelson [c]

[a] *School of Computing, Univ. of Utah, Salt Lake City, UT, USA*

[b] *Department of Mathematics, Boise State Univ., Boise, ID, USA*

[c] *Department of Mathematics, Univ. of Utah, Salt Lake City, UT, USA*

Abstract

We present a new computational method by extending the Immersed Boundary (IB) method with a spectrally-accurate geometric model based on Radial Basis Function (RBF) interpolation of the Lagrangian structures. Our specific motivation is the modeling of platelets in hemodynamic flows, though we anticipate that our method will be useful in other applications as well. The efficacy of our new RBF-IB method is shown through a series of numerical experiments. Specifically, we compare our method with the traditional IB method in terms of convergence and accuracy, computational cost, maximum stable time-step size and volume loss. We conclude that the RBF-IB method has significant advantages over the traditional Immersed Boundary method, and is well-suited for modeling of platelets in hemodynamic flows.

Radial Basis Functions, Immersed Boundary Methods, Platelet Modeling

1 Introduction

The Immersed Boundary (IB) Method was introduced by Charles Peskin in the early 1970's to solve the coupled equations of motion of a viscous, incompressible fluid and one or more

Email addresses: shankar@cs.utah.edu (Varun Shankar), wright@math.boisestate.edu (Grady B. Wright), kirby@cs.utah.edu (Robert M. Kirby), fogelson@math.utah.edu (Aaron L. Fogelson).

[1] Corresponding Author

massless, elastic surfaces or objects immersed in the fluid [24]. The IB method was originally developed to model blood flow in the heart and through heart valves [24, 26, 27], but has since been used in a wide variety of other applications, particularly in biofluid dynamics problems where complex geometries and immersed elastic membranes or structures are present and make traditional computational approaches difficult. Examples include swimming of organisms [7, 8], biofilm processes [4], mechanical properties of cells [1], cochlear dynamics [2], and insect flight [20, 21]. In this work, we are motivated by the application of the IB method to platelet aggregation in blood clotting, but expect our method to be useful in other applications as well.

Intravascular blood clots (thrombi) are initiated by damage to the endothelial cell lining of a blood vessel and involve the formation on the damaged surface of clumps of cells intermixed with a fibrous protein gel. The cells involved in this process are platelets, and the subject of this paper is a new approach to modeling platelets in order to simulate their adhesion to the injured vascular wall and cohesion with one another during the formation of a thrombus. The IB method is used to describe the mechanical interactions among a collection of discrete platelets, the background fluid, and the vessel wall. However, we now model our platelets with Radial basis functions (RBFs) in order to achieve more accurate and less costly simulations.

In this introduction, we briefly describe the relevant biology, describe how the IB method has been used in our previous platelet aggregation simulations, and give an overview of how use of the new method changes this description.

1.1 Modeling the mechanics of platelet aggregation

Disruption of the endothelial cell lining exposes collagen and adsorbed von Willebrand factor (vWF) molecules in the subendothelial matrix to the blood. Platelets adhere to both molecules via specific receptor molecules on the platelets' surfaces. In addition to slowing or stopping platelet motion over the subendothelium, this binding triggers intracellular signaling pathways that lead to platelet activation [18, 29].

Our platelet aggregation models [7, 11–13, 33] track the motion and behavior of a collection of individual platelets as they interact with the suspending fluid, one another, and the vessel walls. We also track fluid concentrations of platelet activating chemicals, cell-cell and cell-surface forces, fluid motion, and the local fluid forces on the growing thrombus. In our models, nonactivated platelets are activated by proximity to reactive sites on the injured wall, or through exposure to a sufficiently high concentration of activator in the fluid. Activation enables a platelet to cohere with other activated platelets and to secrete additional activator. The platelets and the secreted chemical move by advection with the fluid and diffusion relative to it. Each platelet is represented as an IB object, *i*.e., as a collection of elastically-linked Lagrangian points that each move at the local fluid velocity. New elastic links are created dynamically to model the adhesion of a platelet to the injured wall or the cohesion of activated platelets to one another. Multiple links can form between a pair of activated

model platelets or between a model platelet and the injured wall, and these links collectively represent the ensemble of molecular bridges binding real platelets to one another or to the damaged vessel. The links exert forces on the surrounding fluid to resist motions which would otherwise separate the linked entities. Links may break if subject to sufficiently high stress. Model variables are fully coupled: the fluid carries the activator and platelets, while the interplatelet forces, potentiated by chemically-induced activation of the platelets, determine the local flow. In this paper, we focus on mechanical interactions, not the activation process, and so we specify the conditions under which a platelet becomes activated and able to cohere with other activated platelets.

1.2 Motivation for the RBF-IB method

We model platelets as closed curves of interconnected IB points in 2D. A platelet's area or volume is determined by the region enclosed by the curve or surface and is preserved because of the incompressibility of the fluid. Inactive platelets are approximately elliptical in 2D models, while activated platelets are approximately circular. Piecewise linear approximations of platelets are currently used in IB methods applied to the simulation of platelet aggregation (*e.g.* [7, 12, 13]).

In previous work [30], we found that interpolation with radial basis functions restricted to the circle (or sphere in 3D) offered accuracy and computational cost comparable to that offered by Fourier-based methods in modeling an infinitely smooth target shape, its normals and tension forces computed on its surface. Furthermore, interpolation with radial basis functions resulted in better convergence (often an order more) than that offered by both Fourier-based methods when the target shape had only one or two underlying derivatives. In general, use of radial basis functions led to a computational cost comparable to that of Fourier-based methods and orders of magnitude lower (for the same accuracy) than that of the standard combination of techniques (piecewise quadratic interpolation and finite differences) used in many IB methods.

We now turn our attention to exploring the consequences of using a parametric RBF geometric model within the full IB method, with platelet aggregation as our target application. We seek to determine if the advantages inherent in the RBF interpolation of static shapes carries over to full-fledged IB simulations, and also if the RBF interpolation can give us benefits that are apparent only in full-fledged IB simulations. In this work, we propose a new immersed boundary algorithm that utilizes the features afforded by our RBF geometric model.

The paper is organized as follows. In Section 2 we briefly discuss the traditional Immersed Boundary Method for simulating fluid-structure interaction. In Section 3 we review the piecewise linear and RBF geometric modeling strategies and review the components necessary for handling immersed elastic structures in the IB method. In Section 4 we provide details of the spatial and temporal discretizations of both versions of the IB method. In Section 5 we

present our comparison of the RBF-IB method with the traditional IB method in terms of convergence, accuracy, area loss and time-step size. We also provide energy estimates for RBF-IB simulations. We then present results from a large platelet aggregation simulation in 2D. Section 6 contains a summary of our findings and a discussion of future research directions.

Notation: We denote vectors with as many components as the spatial dimension in bold. We denote vectors with as many components as the number of data sites ($N_{\rm d}$) or sample sites ($N_{\rm s}$) by underlining. We indicate matrices with ($N_{\rm d}$) or ($N_{\rm s}$) rows and two columns in bold with underlining.

2 Review of the Immersed Boundary Method

To review the IB method, we focus on a simple two-dimensional model problem in which a single fluid-filled closed elastic membrane is immersed in a viscous fluid. The physics of the model problem is that an elastic membrane is under tension and exerts forces on the adjacent fluid. These forces may cause the fluid to move and, correspondingly, cause points on the membrane to move along with the fluid. In the IB method, the fluid is described in the Eulerian frame through a velocity field $\boldsymbol{u}(\boldsymbol{x},t)$ and pressure field $p(\boldsymbol{x},t)$ defined at every point $\boldsymbol{x}$ in the physical domain Ω. The elastic membrane is described in the Lagrangian frame. Let the elastic membrane be parameterized by q, and denote by $\boldsymbol{X}(q,t)$ the spatial coordinates at time t of the membrane point labeled by q. The IB equations are the following coupled equations of motion for the fluid variables $\boldsymbol{u}(\boldsymbol{x},t)$ and $p(\boldsymbol{x},t)$ and the membrane configuration $\boldsymbol{X}(q,t)$.

$$\rho(\boldsymbol{u}_t + \boldsymbol{u}\cdot\nabla\boldsymbol{u}) = -\nabla p + \mu\Delta\boldsymbol{u} + \boldsymbol{f}, \qquad \nabla\cdot\boldsymbol{u} = 0, \tag{1}$$

$$\boldsymbol{F}(q,t) \;=\; \boldsymbol{F}\left(\boldsymbol{X}(q,t), \frac{\partial}{\partial q}\boldsymbol{X}(q,t)\right), \tag{2}$$

$$\boldsymbol{f}(\boldsymbol{x},t) \;=\; \int \boldsymbol{F}(q,t)\;\delta(\boldsymbol{x}-\boldsymbol{X}(q,t))\;dq, \tag{3}$$

$$\frac{\partial \boldsymbol{X}}{\partial t}(q,t) = \int_\Omega \boldsymbol{u}(\boldsymbol{x},t)\;\delta(\boldsymbol{x}-\boldsymbol{X}(q,t))d\boldsymbol{x}. \tag{4}$$

Equations (1) are the Navier Stokes equations which describe the dynamics of a viscous incompressible fluid, of constant density ρ and constant viscosity μ, driven by a force density $\boldsymbol{f}$ which here arises because of the elastic deformation of the immersed membrane. Equation (2) specifies the elastic force (per unit q) at each point of the immersed boundary object. The functional dependence of this force on the state of the boundary is specified appropriately to the material being modeled. Equation (3) defines the fluid force density $\boldsymbol{f}(\boldsymbol{x},t)$ in terms of the immersed boundary elastic force density $\boldsymbol{F}$. Equation (4) specifies that the velocity of each immersed boundary point equals the fluid velocity at the same location, a formulation of the no-slip boundary condition for viscous flows. In the model problem and the platelet applications, we assume that the IB objects are neutrally buoyant; the IB membrane itself

carries no mass and each object's mass is attributed to the fluid in which it sits. For more on the IB method, see [25].

3 Geometric modeling of platelets

In this section, we review the two geometric modeling strategies to be compared in the context of the Immersed Boundary method applied to platelet aggregation. For a full description of these strategies, see [30].

3.1 Piecewise linear model

In the traditional IB method, the surfaces of platelets are represented by a collection of Immersed Boundary points. We henceforth alternatively refer to the IB points in the traditional IB method as *sample sites*, and denote them by $\boldsymbol{X}_{\mathrm{s}}(t) = \boldsymbol{X}(q,t)$ for each discrete $q \in \Gamma$ and at a particular time t. The surface elastic forces of the platelets are spread from these sample sites into the neighboring fluid. Both tension and bending forces are computed using a collection of springs (typically linear springs) between pairs of sample sites. An explicit piecewise linear interpolant of the surface is not formed. If other information (such as normal vectors) is needed at the sample points, an approximation to the surface represented by the sample sites may be formed by a piecewise quadratic interpolation of the sample sites (*e.g.*, [32]). After the incompressible Navier Stokes equations are solved, velocities from the portions of the Eulerian grid surrounding the sample sites are interpolated to the sample sites using a discretization of Equation 4 and used to move the platelets.

3.2 Parametric RBF model

The RBF method is a popular tool for approximating multidimensional scattered data. An excellent overview of the theory and application of this method can be found in the two books by Fasshauer [5] and Wendland [31]. The restriction of the RBF method to interpolation on a circle and/or sphere began to receive considerable attention from a theoretical standpoint in the mid 1990s (see [6, §6] for a discussion). When restricted to these domains, the RBF method is sometimes referred to as the *zonal basis function* (ZBF) or *spherical basis function* (SBF) method [31, Ch. 17]. Several studies have provided error estimates for RBF interpolation on circles and spheres; see, for example, [19, 22]. In [19], it was shown that these interpolants can provide spectral accuracy provided the underlying target function is sufficiently smooth, while in [22], error estimates are given in the case that the target function belongs to a Sobolev space. The RBF method has also been used successfully for numerically solving partial differential equations on the surface of the sphere [9, 10], as well as more general surfaces [15, 28].

Here, we present the RBF model developed in our earlier work [30]. It is based on explicit parametric representations of the objects. Since our target objects are platelets, which in 2D models are nearly elliptical or circular, we choose a polar parameterization. We use our model to define operators necessary for the computation of geometric and mechanical quantities required by the IB method.

We represent a platelet surface at any time t parametrically by

$$\boldsymbol{X}(\lambda, t) = (X(\lambda, t), Y(\lambda, t)) \tag{5}$$

where $0 \leq \lambda \leq 2\pi$ is the parametric variable and $\boldsymbol{X}(0, t) = \boldsymbol{X}(2\pi, t)$. We explicitly track a finite set of N_{d} points $\boldsymbol{X}_1^{\mathrm{d}}(t), \ldots, \boldsymbol{X}_2^{\mathrm{d}}(t)$, which we refer to as *data sites*. Here $\boldsymbol{X}_j^{\mathrm{d}}(t) := \boldsymbol{X}(\lambda_j^{\mathrm{d}}, t)$, $j = 1, \ldots, N_{\mathrm{d}}$, and we refer to the parametric coordinates $\lambda_1^{\mathrm{d}}, \ldots, \lambda_{N_{\mathrm{d}}}^{\mathrm{d}}$ as the *data site nodes* (or simply *nodes*). We construct each component of $\boldsymbol{X}$ by using a smooth RBF interpolant of the data sites in parameter space as discussed in detail below. We also make use of derivatives of the interpolant at the data sites and we use the interpolant and its derivatives at another set of prescribed *sample points* or *sample sites*, which correspond to N_{s} parameter values: $\lambda_1^{\mathrm{s}}, ..., \lambda_{N_{\mathrm{s}}}^{\mathrm{s}}$.

We first explain how to construct an RBF interpolant to the X component of $\boldsymbol{X}$ using the data $(\lambda_1^{\mathrm{d}}, X_1^{\mathrm{d}}(t)), ..., (\lambda_{N_{\mathrm{d}}}^{\mathrm{d}}, X_{N_{\mathrm{d}}}^{\mathrm{d}}(t))$; the construction of the Y component follows in a similar manner. Let $\phi(r)$ be a scalar-valued radial kernel, whose choice we discuss below. Define $X(\lambda, t)$ by

$$X(\lambda, t) = \sum_{k=1}^{N_d} c_k^X \phi\left(\sqrt{2 - 2\cos(\lambda - \lambda_k^{\mathrm{d}})}\right). \tag{6}$$

Note that the square root term in Equation (6) is the Euclidean distance between the points on the unit circle whose angular coordinates are λ and λ_k^{d}. For this paper, we use the multi-quadric (MQ) radial kernel function, given by

$$\text{MQ:} \quad \phi(r) = \sqrt{1 + (\varepsilon r)^2}, \tag{7}$$

where ε is called the shape parameter. To have $X(\lambda, t)$ interpolate the given data, we require that the coefficients $c_k^X, k = 1, ..., N_{\mathrm{d}}$ satisfy the following system of equations:

$$\underbrace{\begin{bmatrix} \phi(r_{1,1}) & \cdots & \phi(r_{1,N_{\mathrm{d}}}) \\ \phi(r_{2,1}) & \cdots & \phi(r_{2,N_{\mathrm{d}}}) \\ \vdots & \ddots & \vdots \\ \phi(r_{N_{\mathrm{d}},1}) & \cdots & \phi(r_{N_{\mathrm{d}},N_{\mathrm{d}}}) \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} c_1^X \\ c_2^X \\ \vdots \\ c_{N_{\mathrm{d}}}^X \end{bmatrix}}_{\underline{c}_{\mathrm{d}}^X} = \underbrace{\begin{bmatrix} X_1^{\mathrm{d}}(t) \\ X_2^{\mathrm{d}}(t) \\ \vdots \\ X_{N_{\mathrm{d}}}^{\mathrm{d}}(t) \end{bmatrix}}_{\underline{X}_{\mathrm{d}}(t)}, \tag{8}$$

where $r_{j,k} = \sqrt{2 - 2\cos(\lambda_j^{\mathrm{d}} - \lambda_k^{\mathrm{d}})}$. Since $r_{j,k} = r_{k,j}$, the matrix A in this system is symmetric. More importantly, for the MQ kernels, A is non-singular, with the global support and infinite

smoothness of $\phi(r)$ lending itself to spectral accuracy and convergence on smooth problems [5, 31].

In our application, we want to be able to evaluate $X(\lambda, t)$ at sample sites corresponding to parameter values $\lambda_1^{\mathrm{s}}, ..., \lambda_{N_{\mathrm{s}}}^{\mathrm{s}}$, that stay fixed over time. While we could use Equation (6) to do this, it is much more convenient from a notational and computational perspective to construct an *evaluation matrix* that combines the linear operations of constructing the interpolant to $\underline{\boldsymbol{X}}_{\mathrm{d}}(t) = [\underline{X}_{\mathrm{d}}(t),\ \underline{Y}_{\mathrm{d}}(t)]$, for any t, and evaluating it at $\lambda_1^{\mathrm{s}}, ..., \lambda_{N_{\mathrm{s}}}^{\mathrm{s}}$. The evaluation matrix can be constructed by first noting that Equation (6) can be written as

$$X(\lambda, t) = \underbrace{\left[\phi\left(\sqrt{2 - 2\cos(\lambda - \lambda_1^{\mathrm{d}})}\right) \cdots \phi\left(\sqrt{2 - 2\cos(\lambda - \lambda_{N_{\mathrm{d}}}^{\mathrm{d}})}\right)\right]}_{\underline{b}(\lambda)^T} \underline{c}_{\mathrm{d}}^X. \tag{9}$$

Since $\underline{c}_{\mathrm{d}}^X = A^{-1}\underline{X}_{\mathrm{d}}(t)$, we can write Equation (6) as $X(\lambda, t) = \underline{b}(\lambda)^T A^{-1}\underline{X}_{\mathrm{d}}(t)$. The evaluation of $X(\lambda, t)$ at $\lambda_1^{\mathrm{s}}, ..., \lambda_{N_{\mathrm{s}}}^{\mathrm{s}}$ can then be obtained as follows:

$$\underbrace{\begin{bmatrix} X(\lambda_1^{\mathrm{s}}, t) \\ \vdots \\ X(\lambda_{N_{\mathrm{s}}}^{\mathrm{s}}, t) \end{bmatrix}}_{\underline{X}_{\mathrm{s}}(t)} = \underbrace{\begin{bmatrix} \underline{b}(\lambda_1^{\mathrm{s}})^T \\ \vdots \\ \underline{b}(\lambda_{N_{\mathrm{s}}}^{\mathrm{s}})^T \end{bmatrix}}_{B} A^{-1}\underline{X}_{\mathrm{d}}(t) = \underbrace{BA^{-1}}_{\mathcal{E}_{\mathrm{s}}}\underline{X}_{\mathrm{d}}(t). \tag{10}$$

So, given the data sites $\underline{X}_{\mathrm{d}}(t)$ at any time t, we can interpolate their coordinates with an RBF expansion *and* evaluate the interpolant at the sample site nodes $\lambda_1^{\mathrm{s}}, ..., \lambda_{N_{\mathrm{s}}}^{\mathrm{s}}$ to get $\underline{X}_{\mathrm{s}}(t)$ by the matrix-vector product $\mathcal{E}_{\mathrm{s}}\underline{X}_{\mathrm{d}}(t)$. In fact, this same procedure can be used to give values at sample site nodes for any quantity whose values we have at data site nodes and which we represent using an RBF expansion (*e.g.*, $\underline{Y}_{\mathrm{d}}(t) = [Y_1^{\mathrm{d}}(t) \cdots Y_{N_{\mathrm{d}}}^{\mathrm{d}}(t)]^T$). Furthermore, the evaluation matrix $\mathcal{E}_{\mathrm{s}}$ can be precomputed once at $t = 0$ and used for all subsequent times.

We also need to compute geometric quantities such as tangent vectors, and mechanical quantities such as forces at data sites and/or sample sites. These quantities require computing derivatives with respect to λ of the platelet surface coordinates $(X(\lambda, t), Y(\lambda, t))$. We use the RBF-based representation of the surface to compute these derivatives, and we will express derivatives of the RBF interpolant in matrix-vector form. Toward this end, we use similar notation to Equation (9) and define the vector

$$\begin{aligned} \underline{b}_{\lambda}^n(\tilde{\lambda}) &:= \frac{\partial^n}{\partial \lambda^n}\underline{b}(\lambda)\bigg|_{\lambda=\tilde{\lambda}} \\ &= \left[\frac{\partial^n}{\partial \lambda^n}\phi\left(\sqrt{2 - 2\cos(\lambda - \lambda_1^{\mathrm{d}})}\right)\bigg|_{\lambda=\tilde{\lambda}} \cdots \frac{\partial^n}{\partial \lambda^n}\phi\left(\sqrt{2 - 2\cos(\lambda - \lambda_{N_{\mathrm{d}}}^{\mathrm{d}})}\right)\bigg|_{\lambda=\tilde{\lambda}}\right]^T, \end{aligned}$$

for any $0 \leq \tilde{\lambda} \leq 2\pi$. Just as $\underline{b}(\tilde{\lambda})^T A^{-1}\underline{X}_{\mathrm{d}}(t)$ gives the value of $X(\tilde{\lambda}, t)$, we can use $\underline{b}_{\lambda}^n(\tilde{\lambda})$ to

obtain the n^{th} derivative of $X(\lambda, t)$ with respect to λ as

$$\left.\frac{\partial^n}{\partial \lambda^n} X(\lambda, t)\right|_{\lambda=\tilde{\lambda}} = \underline{b}_\lambda^n(\tilde{\lambda})^T A^{-1} \underline{X}_{\text{d}}(t).$$

The evaluation of the n^{th} derivative of $X(\lambda, t)$ at the data site nodes $\lambda_1^{\text{d}}, \ldots, \lambda_{N_{\text{d}}}^{\text{d}}$ can then be obtained as follows:

$$\begin{bmatrix} \left.\frac{\partial^n}{\partial \lambda^n} X(\lambda, t)\right|_{\lambda=\lambda_1^{\text{d}}} \\ \vdots \\ \left.\frac{\partial^n}{\partial \lambda^n} X(\lambda, t)\right|_{\lambda=\lambda_{N_{\text{d}}}^{\text{d}}} \end{bmatrix} = \underbrace{\begin{bmatrix} \underline{b}_\lambda^n(\lambda_1^{\text{d}})^T \\ \vdots \\ \underline{b}_\lambda^n(\lambda_{N_{\text{d}}}^{\text{d}})^T \end{bmatrix}}_{B_{\lambda^{\text{d}}}^n} A^{-1} \underline{X}_{\text{d}}(t) = \underbrace{B_{\lambda^{\text{d}}}^n A^{-1}}_{\mathcal{D}_{\lambda^{\text{d}}}^n} \underline{X}_{\text{d}}(t). \tag{11}$$

In a similar manner, the evaluation of the n^{th} derivative of $X(\lambda, t)$ at the sample site nodes $\lambda_1^{\text{s}}, \ldots, \lambda_{N_{\text{d}}}^{\text{s}}$ can be obtained by

$$\begin{bmatrix} \left.\frac{\partial^n}{\partial \lambda^n} X(\lambda, t)\right|_{\lambda=\lambda_1^{\text{s}}} \\ \vdots \\ \left.\frac{\partial^n}{\partial \lambda^n} X(\lambda, t)\right|_{\lambda=\lambda_{N_{\text{s}}}^{\text{s}}} \end{bmatrix} = \underbrace{\begin{bmatrix} \underline{b}_\lambda^n(\lambda_1^{\text{s}})^T \\ \vdots \\ \underline{b}_\lambda^n(\lambda_{N_{\text{s}}}^{\text{s}})^T \end{bmatrix}}_{B_{\lambda^{\text{s}}}^n} A^{-1} \underline{X}_{\text{d}}(t) = \underbrace{B_{\lambda^{\text{s}}}^n A^{-1}}_{\mathcal{D}_{\lambda^{\text{s}}}^n} \underline{X}_{\text{d}}(t). \tag{12}$$

For given data sites $\underline{X}_{\text{d}}(t)$ at any time t, we can interpolate these values with an RBF expansion *and* evaluate the n^{th} derivative of the interpolant at the data site nodes by the matrix-vector product $\mathcal{D}_{\lambda^{\text{d}}}^n \underline{X}_{\text{d}}(t)$ and at the sample site nodes by $\mathcal{D}_{\lambda^{\text{s}}}^n \underline{X}_{\text{d}}(t)$. We refer to the $N_{\text{d}} \times N_{\text{d}}$ matrices $\mathcal{D}_{\lambda^{\text{d}}}^n$ and the $N_{\text{s}} \times N_{\text{d}}$ matrices $\mathcal{D}_{\lambda^{\text{s}}}^n$ as *RBF differentiation matrices*.

The matrices $\mathcal{D}_{\lambda^{\text{d}}}^n$ and $\mathcal{D}_{\lambda^{\text{s}}}^n$ can be used to give values at respective data site or sample site nodes of the n^{th} derivative of the RBF interpolant of any quantity whose values we have at the data site nodes (*e.g.*, $\underline{Y}_{\text{d}}(t) = [Y_1^{\text{d}}(t) \cdots Y_{N_{\text{d}}}^{\text{d}}(t)]^T$). These matrices can also be precomputed once at $t = 0$ and used for all subsequent times.

Having defined the operators to compute derivatives of the RBF interpolant, we define the quantity

$$\boldsymbol{\tau} := \frac{\partial}{\partial \lambda} \boldsymbol{X}(\lambda, t) = \left(\frac{\partial}{\partial \lambda} X(\lambda, t), \frac{\partial}{\partial \lambda} Y(\lambda, t) \right) = (\tau_X, \tau_Y). \tag{13}$$

The unit tangent vector to $\boldsymbol{X}(\lambda, t)$ is then given by

$$\hat{\boldsymbol{\tau}} := \frac{\boldsymbol{\tau}}{\|\boldsymbol{\tau}\|} = (\hat{\tau}_X, \hat{\tau}_Y). \tag{14}$$

In our experiments, we assume that the Lagrangian force at a point on a platelet is the sum of a tension force, a bending-resistant force and possibly a force due to a bond between that point and a point on another platelet or the vessel wall. For the tension force, we use the fiber model defined in [25], according to which the elastic tension force density at $\boldsymbol{X}(\lambda_i, t_k)$ is given by

$$\boldsymbol{F}^{\mathrm{T}}(\lambda_i, t_k) = \frac{\partial}{\partial \lambda}(T\hat{\boldsymbol{\tau}})\bigg|_{\lambda_i, t_k}, \tag{15}$$

where $T = k_{\mathrm{t}}(\|\boldsymbol{\tau}\| - l_0)$ is the fiber tension and $k_{\mathrm{t}} > 0$ is constant. We set $l_{0,i} = \|\boldsymbol{\tau}\|\big|_{\lambda_i, t_0}$, where t_0 is the initial time of the simulation. For a bending-resistant force, we use a linear variant of the force defined in [16] and define the elastic force density at $\boldsymbol{X}(\lambda_i, t_k)$ due to how much the platelet surface there is bent to be

$$\boldsymbol{F}^{\mathrm{B}}(\lambda_i, t_k) = -\, k_{\mathrm{b}} \left(\frac{\partial^4 \boldsymbol{X}}{\partial \lambda^4} - \frac{\partial^4 \boldsymbol{X}^0}{\partial \lambda^4} \right)\bigg|_{\lambda_i, t_k}. \tag{16}$$

Here $\boldsymbol{X}^0 = \boldsymbol{X}(\lambda_i, t_0)$ is the initial configuration of the platelet and $k_{\mathrm{b}} > 0$ is constant. Ideally, the constants k_{t} and k_{b} would be chosen to reflect values determined from experiments involving real platelets. In our work, we choose k_{t} and k_{b} that keep the platelets rigid, and scale them as we refine the background Eulerian grid.

We defer discussion of how we compute the forces given by Equations (15) and (16) to the next section (and the Appendix), since the implementation is different for the RBF and piecewise-linear representations of the platelet boundary. However, the force acting on a platelet due to other platelets (and/or walls) is common to both methods. We use the spring force defined in [13]: let $p_1, p_2, ..., p_{N_p}$ be the indices corresponding to the platelets in the domain. Let p_1 and p_2 be the indices of two platelets which are linked at sample sites $\boldsymbol{X}_{p_1}(\lambda^s_{i_1})$ and $\boldsymbol{X}_{p_2}(\lambda^s_{i_2})$. The force at $\boldsymbol{X}_{p_1}(\lambda^s_{i_1})$ is then given by:

$$\boldsymbol{F}^{\mathrm{C}}_{p_1}(\lambda^s_{i_1}, t_k) = K_{\mathrm{C}}(||\boldsymbol{X}_{p_2}(\lambda^s_{i_2}) - \boldsymbol{X}_{p_1}(\lambda^s_{i_1})|| - l_{0,\mathrm{C}}) \frac{\boldsymbol{X}_{p_2}(\lambda^s_{i_2}) - \boldsymbol{X}_{p_1}(\lambda^s_{i_1})}{||\boldsymbol{X}_{p_2}(\lambda^s_{i_2}) - \boldsymbol{X}_{p_1}(\lambda^s_{i_1})||}, \tag{17}$$

where K_{C} and $l_{0,\mathrm{C}}$ are the interplatelet cohesion spring stiffness and the resting length, respectively; we also set $\boldsymbol{F}^{\mathrm{C}}_{p_2}(\lambda^s_{i_2}, t_k) = -\boldsymbol{F}^{\mathrm{C}}_{p_1}(\lambda^s_{i_1}, t_k)$. The formulation for platelet-wall links is similar.

4 Numerical Discretization

In this section we present the implementation details for both IB methods. For each method, we briefly describe the spatial discretization for both the Lagrangian and Eulerian quantities. We then describe the time-stepping scheme for each method.

4.1 The Piecewise-Linear IB method

Traditionally, finite-difference approximations of Equations (15) and (16) are used in conjunction with piecewise linear methods in 2D (*e.g.* [16]). We use a second-order central difference involving triads of sample sites or IB points to discretize the derivatives involved in the computation of both the tension and bending forces (including tangent lengths). It is useful to think of these finite difference approximations to the constitutive model as Hookean springs connecting pairs of IB points. Note that these differences are only second-order assuming a near-uniform sampling. This is one of the sources of error for the IB method.

For the Eulerian spatial discretization, we use a second-order centered finite-difference approximation to the Laplacian on a staggered grid. We discretize the advection term using second-order centered differences, averaging quantities to cell edges or nodes as required. For the approximate δ-function, we use the "cosine" form described by Peskin [25] which ensures that the entire IB force is transmitted to the grid, that the force density on the grid is a continuous function of the IB point locations, and that the communication between grid and IB points is very localized. After each update of the IB point locations, new links are formed and existing ones are broken using the model's rules for these types of events. To prevent leakage, it is common to enforce a restriction that the IB point spacing on the surface be no greater than $0.5h$, where h is the Eulerian grid cell width.

We use the formally second-order Runge-Kutta time-stepping scheme outlined in [3]. This time-stepping scheme appears to show second-order convergence in time for a smooth forcing function, or for an elastic material that fills the whole domain, as demonstrated in [3]. However, as our results will show, this scheme will produce only first-order convergence in time in the presence of a sharp interface between the fluid and the elastic material, as is typical of IB methods. The full scheme is presented in Appendix B.

4.2 The RBF-IB method

In order to construct the operators utilized by our algorithm, we must first choose an appropriate node set. We use N_{d} equally-spaced values on the interval $(0, 2\pi]$ as the data site node set $\{\lambda_k^{\mathrm{d}}\}_{k=1}^{N_{\mathrm{d}}}$. This gives a uniform sampling in the parametric space. More importantly, since our target objects are either circular or elliptical in 2D, these node points correspond to a nearly uniform distribution of data sites on the object. We also use $N_{\mathrm{s}} >> N_{\mathrm{d}}$ equally-spaced points in the interval $(0, 2\pi]$ as the set of sample site nodes $\{\lambda_j^{\mathrm{s}}\}_{j=1}^{N_{\mathrm{s}}}$ since this results in a set of sample sites that are well distributed over the object. As in the piecewise-linear IB method, we make sure to start with enough points that the sample site spacing is never greater than $0.5h$, though the data site spacing can be much greater. In the results section, we explore the ramifications of this choice.

We have formulated our operators to ensure that operations like evaluation of the interpolant

and computing derivatives (and therefore the constitutive model) do not require solving a linear system for any time step of the platelet simulation except the initial step. This is possible because, though the data sites and sample sites move over the course of the simulation, their values in parameter space do not change. For the RBF model of the platelets, the evaluation matrix $\mathcal{E}_{\mathrm{s}}$ in Equation (10) and differentiation matrices $\mathcal{D}^n_{\lambda^{\mathrm{d}}}$ and $\mathcal{D}^n_{\lambda^{\mathrm{s}}}$ in Equations (11) and (12), respectively can be computed using the FFT as discussed in our previous work [30]. This is possible since the data site nodes $\{\lambda^{\mathrm{d}}_k\}_{k=1}^{N_{\mathrm{d}}}$ are equally-spaced, which results in the A matrix defined Equation (8) having a circulant matrix structure. The costs and accuracy of the RBF models are elaborated upon in the discussion of the results. The algorithm to compute forces on platelets using these operators is presented in the Appendix A.

The RBF-IB method uses the same time-stepping scheme and Eulerian discretization as the piecewise linear IB method, with one important difference. When computing the forces at time level $n + 1/2$, we advance the *data sites* to time level $n + 1/2$, generate a set of sample sites at that time level, and compute forces at the sample sites. Similarly, we use the mid-step approximation to the velocity field to advance the *data sites* to time level $n + 1$. We thus generate only a single set of sample sites every time-step, since the sample sites are only needed when the data sites are advanced to time level $n + 1/2$. It is clear that if the number of data sites is fewer than the number of sample sites, this results in improved computational efficiency over the piecewise linear IB method. However, it is important to explore the effect of our changes on the convergence of the algorithm. We explore these questions in the results section. For a more complete description of the RBF-IB time-stepping scheme, see Appendix C.

5 Results

In this section, we first compare the convergence of both methods on a canonical test problem. We also use this test problem to explore the relationship between the number of data sites (N_{d}) and the Eulerian grid spacing (h). We then compare the area loss in an elastic object simulated by each method on the same problem, and discuss the time-step sizes allowed by both methods. We follow with a discussion of the change in energy over time in the RBF-IB method. We then provide timings for platelet simulations and discuss both foreseen and unforeseen results of using the RBF model within the IB method. Finally, we present the results of platelet aggregation simulations conducted using the RBF-IB method.

Description of our standard fluid-structure interaction problem:

We describe a standard fluid-structure interaction problem on which we test both versions of the IB method. This problem is commonly used in the IB literature (*e.g.*, [23]). The problem involves placing an elliptical object with its center of mass at the center of the $[0, 1]^2$ physical domain. The elliptical object has a circle of the same area as its rest configuration, and attempts to attain the rest configuration subject to a combination of tension and bending forces. The physical domain is filled with a fluid that is initially at rest, with periodic

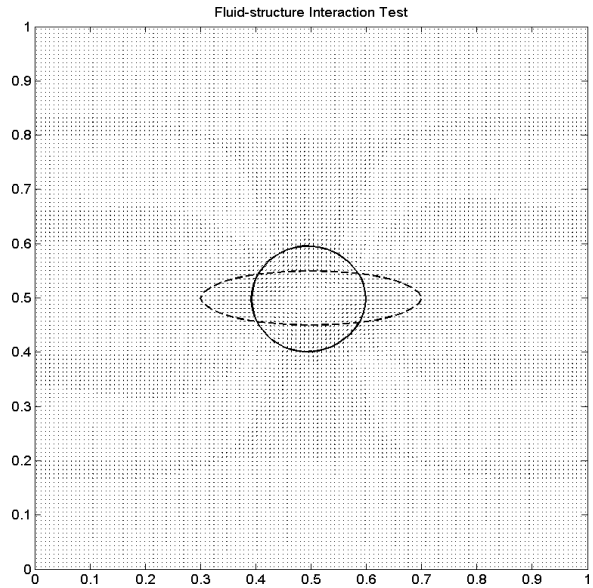

Fig. 1. A visualization of the fluid-structure interaction test. The dashed lines show the initial ellipse, while the filled line indicates the near-circular object at the final time $t = 2.0$. The small arrows indicate the velocity field at the final time. The maximum velocity is very close to zero at this time as the object is almost at rest.

boundary conditions in the x-direction and no-slip Dirichlet boundary conditions in the y-direction. We set the radius of the target circle to be $r = 0.1$ units, with the ellipse initially having a major axis of $a = 2r$ and a minor axis of $b = 0.5r$. This test is visualized in Figure 1.

5.1 Convergence studies

In previous work [30], we compared the accuracy and convergence of both RBF and traditional IB geometric modeling strategies for static platelet-like shapes. We now compare the accuracy and convergence of the full RBF-IB and PL-IB methods, both for the velocity field and for the immersed elastic structure.

For the fluid, on each grid with cell width h, we define the quantity $\boldsymbol{u}^{c,h}$, the coarsened discrete velocity field from the 256×256 grid. For each grid point i on a grid with cell width h, we compute the quantity $f_i^h = ||\boldsymbol{u}_i^h - \boldsymbol{u}_i^{c,h}||$. We define the l_2 error in the velocity field as $e_2(h) = \sqrt{\sum_i f_i^h h^2}$, and the l_∞ error in the velocity field to be $e_\infty(h) = \max_i f_i^h$. The convergence rate for errors $e(2h)$ and $e(h)$ is measured as $p_u = \log_2\left(\frac{e(2h)}{e(h)}\right)$.

For the Lagrangian markers (sample sites or IB points), we adopt the following procedure:

(1) Given the number of sample sites N_s and the radius of the target circle r, we define $\lambda = \frac{2\pi}{N_\mathrm{s}}$, the angle subtended at the center of the circle if the points were evenly-spaced.
(2) We define the quantity $C = 2r\sin(0.5\lambda)$, the chord length between any two points in a

set of evenly-spaced points on an ideal circle. We also define C_{exact} to be the ideal chord length for $N_s = 400$.

(3) We compute the actual distances d_i between the sample sites (or IB points) for a simulation computed on the 256×256 grid with $N_s = 400$. We then compute the quantities $s^e_\infty = \max_i |d_i - C_{exact}|$ and $s^e_2 = (1/N_s)\sqrt{\sum_i |d_i - C_{exact}|}$.

(4) We compute $s^{N_s}_\infty$ and $s^{N_s}_2$ for $N_s = 50, 100, 200$. We then define the l_2 error to be $e_2(N_s) = |s^{N_s}_2 - s^e_2|$ and the l_∞ error to be $e_\infty(N_s) = |s^{N_s}_\infty - s^e_\infty|$.

We define the convergence rate for errors $e(N_s)$ and $e(2N_s)$ to be $p_X = \log_2\left(\frac{e(N_s)}{e(2N_s)}\right)$.

Convergence of the fluid solver:

Before testing both IB methods, we verify the convergence of our fluid solver. The RK2 time-stepping scheme used for the IB method in this paper is designed to give second-order convergence for smooth problems. To test this scheme, we specify a smooth initial condition to the Navier-Stokes equations on a $[0, 1]^2$ domain. The initial condition is a simple parabolic velocity profile. The maximum fluid velocity was set to $u_{max} = 5.0$, the fluid density to $\rho = 1.0$ and the non-dimensionalized viscosity to $\mu = 8.0$. We also force the Navier-Stokes equations with a constant forcing function to give us a Poisseuille flow. We impose periodic boundary conditions in the x direction and no-slip boundary conditions in the y direction.

We then simulate on successfully finer grids and compute errors in the velocity field as outlined previously, using the solution computed on a 256×256 grid as our gold standard. We run the simulation to time $t = 1.0$. Our results (see Table 1) show that the fluid solver exhibits second-order convergence in both space and time for a smooth initial condition. We note that for full Immersed Boundary simulations involving thin interfaces, it is unlikely that we will see second-order convergence.

Grid Size	Δt	L_2 error	Order of convergence	L_∞ error	Order of convergence
32×32	0.0050	6.4090e-03		6.4176e-03	
64×64	0.0025	1.5259e-03	2.07	1.5281e-03	2.07
128×128	0.00125	3.0518e-04	2.322	3.0572e-04	2.321

Table 1
Results of a refinement study on the fluid solver. Errors were measured against the solution computed on 256×256 grid with a time-step of $\Delta t = 6.25 \times 10^{-4}$.

Convergence of the PL-IB method:

We now test the convergence of the PL-IB method on the standard fluid-structure interaction problem. We compare the velocity field and IB point positions to those computed for the same test problem on a 256×256 grid with $N_s = 400$ IB points, and compute errors as outlined previously. It is important to note that the time-steps used for IB simulations with explicit time integrators are necessarily smaller than what we could use for a standard fluid solver. We simulate to final time $t = 2$. Our results are shown in Tables 2 and 3. For both the velocity field and the IB points, we see high convergence when refining from a 32×32 grid

Grid Size	N_s	Δt	L_2 error	Order of convergence	L_∞ error	Order of convergence
32×32	50	2×10^{-4}	4.8302e-03		2.6640e-02	
64×64	100	1×10^{-4}	4.4352e-04	3.45	3.5463e-03	2.91
128×128	200	5×10^{-5}	1.1651e-04	1.93	1.6403e-03	1.11

Table 2
Results of a refinement study with the PL-IB method. We show the convergence of the velocity field, with errors measured against the velocity field of a simulation on a 256×256 grid with $N_s = 400$ sample sites and $\Delta t = 2.5 \times 10^{-5}$.

N_s	Grid Size	Δt	L_2 error	Order of convergence	L_∞ error	Order of convergence
50	32×32	2×10^{-4}	3.1545e-05		5.9530e-04	
100	64×64	1×10^{-4}	4.4447e-06	6.15	8.7363e-05	2.76
200	128×128	5×10^{-5}	9.2942e-07	2.26	2.5866e-05	1.76

Table 3
Results of a refinement study with the PL-IB method. We show the convergence in the IB point positions, with errors measured against the IB point positions from a simulation on a 256×256 grid with $N_s = 400$ IB points and $\Delta t = 2.5 \times 10^{-5}$.

to a 64×64 grid, and closer to first-order convergence when refining from the 64×64 grid to the 128×128 grid. Indeed, as was mentioned earlier, one expects first-order convergence on this problem with the IB method. The high-order convergence seen with the first refinement is likely due to the solution being fairly inaccurate on the coarsest grid. For completeness, we note that $s_2^e = 5.0994e-07$ and $s_\infty^e = 4.3042e-05$. Below, we compare these quantities to the corresponding ones computed from the RBF-IB method.

Convergence of the RBF-IB method for $N_d = 25$:

Grid Size	N_s	Δt	L_2 error	Order of convergence	L_∞ error	Order of convergence
32×32	50	2×10^{-4}	1.7666e-04		8.8682e-04	
64×64	100	1×10^{-4}	1.5097e-04	0.23	5.4255e-04	0.71
128×128	200	5×10^{-5}	8.4247e-05	0.84	3.0738e-04	0.82

Table 4
Results of a refinement study with the RBF-IB method with $N_d = 25$ data sites. We show the convergence of the velocity field, with errors measured against the velocity field of a simulation on a 256×256 grid with $N_s = 400$ sample sites, $N_d = 100$ data sites and $\Delta t = 2.5 \times 10^{-5}$.

Here, we test the convergence of the RBF-IB method on the fluid-structure interaction problem described above. Once again, we compare the velocity field and sample site positions to those computed for the same test problem on a 256×256 grid with $N_s = 400$ sample sites and $N_d = 100$ data sites. Tables 4 and 5 show the results obtained with $N_d = 25$ data sites for the fluid and the object respectively.

Table 4 shows that the errors produced in the RBF-IB method (for the same initial con-

N_s	Grid Size	Δt	L_2 error	Order of convergence	L_∞ error	Order of convergence
50	32×32	2×10^{-4}	3.1188e-06		2.3238e-05	
100	64×64	1×10^{-4}	3.5898e-07	3.12	3.0048e-06	2.95
200	128×128	5×10^{-5}	1.5310e-07	1.23	1.7905e-06	0.75

Table 5
Results of a refinement study with the RBF-IB method with $N_d = 25$ data sites. We show the convergence in the sample site positions, with errors measured against the sample site positions of a simulation on a 256×256 grid with $N_s = 400$ sample sites, $N_d = 100$ data sites and $\Delta t = 2.5 \times 10^{-5}$.

figuration and final time as in the PL-IB method) are lower for the same grid resolution. We see that the RBF-IB method for $N_d = 25$ does not offer good convergence rates for the fluid velocity in any norm, despite producing lower errors on each grid level than the PL-IB method. Table 5 shows results for the errors on the object. The errors on the coarsest grid are high, leading to a higher-than-expected convergence rate in both norms when we measure the errors on a 64×64 grid. However, even on the coarsest grid level, the errors are already lower than those shown by the PL-IB method for the same number of sample sites and initial configuration. Further refinement gives us lower errors than the PL-IB method, but with lower convergence rates as well.

We note that the l_2 and l_∞ errors for the structure on the 256×256 grid for $N_d = 25$ data sites are $s_2^e = 4.3694e-08$ and $s_\infty^e = 1.1113e-06$. These are each an order of magnitude lower than those produced by the PL-IB method on the 256×256 grid.

Convergence of the RBF-IB method for $N_d = 50$:

Grid Size	N_s	Δt	L_2 error	Order of convergence	L_∞ error	Order of convergence
32×32	50	2×10^{-4}	5.5617e-03		3.7404e-02	
64×64	100	1×10^{-4}	2.2436e-04	4.63	1.2403e-03	4.91
128×128	200	5×10^{-5}	7.8934e-05	1.51	2.8859e-04	2.10

Table 6
Results of a refinement study with the RBF-IB method with $N_d = 50$ data sites. We show the convergence of the velocity field, with errors measured against the velocity field of a simulation on a 256×256 grid with $N_s = 400$ sample sites, $N_d = 100$ data sites and $\Delta t = 2.5 \times 10^{-5}$.

N_s	Grid Size	Δt	L_2 error	Order of convergence	L_∞ error	Order of convergence
50	32×32	2×10^{-4}	3.6212e-05		5.2794e-04	
100	64×64	1×10^{-4}	3.8824e-07	6.54	7.8472e-06	6.07
200	128×128	5×10^{-5}	1.7236e-07	1.17	2.0636e-06	1.93

Table 7
Results of a refinement study with the RBF-IB method with $N_d = 50$ data sites. We show the convergence in the sample site positions, with errors measured against the sample site positions of a simulation on a 256×256 grid with $N_s = 400$ sample sites, $N_d = 100$ data sites and $\Delta t = 2.5 \times 10^{-5}$.

We repeat the above test problem with $N_\mathrm{d} = 50$ data sites. As before, we compare the velocity field and sample site positions to those computed for the same test problem on a 256×256 grid with $N_\mathrm{s} = 400$ sample sites and $N_\mathrm{d} = 100$ data sites. Tables 6 and 7 show the results for the fluid and the object respectively.

Examining Table 6, we see high convergence when going from the coarsest grid to a finer grid. However, we see that when moving to the finest grid, we have recovered first-order convergence. The errors on the 128×128 grid for $N_\mathrm{d} = 50$ with the RBF-IB method are close to those on the same grid with $N_\mathrm{d} = 25$, and much lower than those produced by the PL-IB method in both norms. Table 7 shows errors similar to those seen in Table 5, albeit with less erratic convergence. Indeed, we recover first-order convergence in the l_2 norm and close to second-order convergence in the l_∞ norm. Of course, the errors are much lower than those seen in Table 3.

We note that using $N_\mathrm{d} = 50$ data sites does not result in significantly better convergence on the structure than $N_\mathrm{d} = 25$. There are two possible explanations. The first is that the function representing the shape of the object is of limited smoothness (as seen in our previous work [30]), with higher values of N_d causing the interpolation error to saturate or even increase. The alternate (and more likely) explanation is that, since our RBFs are parametrized on the circle, $N_\mathrm{d} = 25$ would already have a very high accuracy when the object becomes a circle, considering the spectral accuracy of RBF interpolation on the circle; in such a scenario, using $N_\mathrm{d} = 50$ data sites would only serve to increase the rounding errors in the representation of the structure. However, it is clear that using $N_\mathrm{d} = 50$ results in better convergence in the velocity field.

The l_2 and l_∞ errors for the structure on the 256×256 grid for $N_\mathrm{d} = 50$ data sites are the same as those for $N_\mathrm{d} = 25$ data sites. Once again, these are each an order of magnitude lower than those produced by the PL-IB method on the 256×256 grid.

Convergence of the RBF-IB method for $N_\mathrm{d} = 0.25N_\mathrm{s}$:

Grid Size	N_s	Δt	L_2 error	Order of convergence	L_∞ error	Order of convergence
32×32	50	2×10^{-4}	4.7909e-04		2.4700e-03	
64×64	100	1×10^{-4}	1.5097e-04	1.67	5.4255e-04	2.19
128×128	200	5×10^{-5}	9.0802e-05	0.73	3.3020e-04	0.72

Table 8
Results of a refinement study with the RBF-IB method with $N_\mathrm{d} = 0.25N_\mathrm{s}$ data sites. We show the convergence of the velocity field, with errors measured against the velocity field of a simulation on a 256×256 grid with $N_\mathrm{s} = 400$ sample sites, $N_\mathrm{d} = 100$ data sites and $\Delta t = 2.5 \times 10^{-5}$.

In the PL-IB method, the number of IB points depends on the grid spacing h. Specifically, the number of IB points is chosen so that the distance between any two sample sites is always less that $0.5h$. In all the tests above, we have maintained that relationship for the IB points in the PL-IB method and the sample sites in the RBF-IB method. In the RBF-IB method, we always use fewer data sites than sample sites, *i.e.*, $N_\mathrm{d} < N_\mathrm{s}$, with the choice of N_d being

N_s	Grid Size	Δt	L_2 error	Order of convergence	L_∞ error	Order of convergence
50	32×32	2×10^{-4}	9.7439e-06		7.0459e-05	
100	64×64	1×10^{-4}	3.5898e-07	4.76	3.0048e-06	4.55
200	128×128	5×10^{-5}	1.2694e-07	1.50	1.4362e-06	1.07

Table 9
Results of a refinement study with the RBF-IB method. We show the convergence in the sample site positions, with errors measured against the sample site positions of a simulation on a 256×256 grid with $N_s = 400$ sample sites, $N_d = 100$ data sites and $\Delta t = 2.5 \times 10^{-5}$.

justified by the results in previous work [30]. Furthermore, in the tests above, we fix N_d even as we refine the fluid grid. For $N_d = 25$, this means that as we refine N_s the distance between data sites increases from $0.8h$ to $3.2h$ (at the start of the simulation).

In order to gain intuition on the relationship between N_d and h, we now perform a convergence study (using the same test problem given above) with increasing values of N_d as h is reduced. To accomplish this, we use values of $N_d = 12, 25, 50, 100$ for $N_s = 50, 100, 200, 400$, *i.e.*, we enforce $N_d = 0.25N_s$. We use the solution computed with $N_d = 100$ and $N_s = 400$ on a 256×256 grid as our gold standard, just as we have in all the other tests. Table 8 shows the results for the fluid. Clearly, the errors are higher and the convergence more erratic than for the fixed $N_d = 50$ tests previously presented, but varying N_d certainly seems to give better convergence than fixing it to $N_d = 25$. However, the convergence in the structure is comparable, with lower errors being achieved than both $N_d = 25$ and $N_d = 50$ (and the PL-IB method as well). This can be seen in Table 9. The advantages of varying N_d with N_s are not clear. Using $N_d = 50$ yields the lowest errors in the fluid on the finest grid level, and reasonably low errors on the structure for all grid levels. Given the similarity of the errors achieved with $N_d = 50$ to those achieved with increasing N_d, we choose the simpler strategy of using a fixed value of $N_d = 50$ for our tests, though we present timings with $N_d = 25$ as well.

Effect of the shape parameter ε:

In previous work [30], we found that the RBF shape parameter $\varepsilon > 0$ had to be selected carefully to achieve spectral accuracy in the representation of the elastic structure. In that work, we found that small values of ε were ideal for interpolating smooth target shapes and larger ones for rougher target shapes. In our tests, we found that the errors depended on ε even in the case of fluid-structure interaction, with smaller values of ε giving the lowest values of s_2^e and s_∞^e on the 256×256 grid. However, as we mentioned in our previous work, lower values of ε can make the RBF interpolation matrix more ill-conditioned. While methods (such as RBF-QR and RBF-RA) have been developed to overcome this poor conditioning [14], they are much more expensive than forming and inverting the standard RBF interpolation matrix. We thus choose a small value of $\varepsilon = 1.2$ for all our tests. When using $N_d = 100$, we use $\varepsilon = 2.0$. These are the smallest values we were able to pick without the interpolation matrix becoming ill-conditioned, a strategy consistent with the one used in our previous work [30].

5.2 Area loss and time-step size in the RBF-IB method:

In this section, we study the area loss in the RBF-IB method in a refinement study. We then explore the maximum stable time-step size afforded by each IB method.

N_s	Grid Size	Δt	% area loss ($N_d = 25$)	% area loss ($N_d = 50$)	% area loss ($N_d = N_s/4$)
50	32×32	2×10^{-4}	0.0680	0.3081	0.0450
100	64×64	1×10^{-4}	0.0047	0.0049	0.0047
200	128×128	5×10^{-5}	0.0023	0.0025	0.0025
400	256×256	2.5×10^{-5}	0.0015	0.0015	0.0015

Table 10
Percentage area loss in the RBF-IB method as a function of grid size, the number of sample sites N_s and the time step Δt. The PL-IB method gives area losses similar to the $N_d = 50$ case, except on the coarsest grid, where the percentage area loss is three times that of the RBF-IB method.

The PL-IB method enforces an IB point separation distance of $0.5h$ in order to reduce area loss over the coarse of the simulation. In the RBF-IB method, while the sample site spacing is maintained at $0.5h$, we use a much coarser data site discretization, with the data site separation being almost $3.2h$ in some cases. In addition, we use the same strategy for interpolating velocities that we do in the PL-IB method, *i.e.*, we interpolate velocities to data sites from a 4×4 patch of fluid around each data site. While this can result in signifcant computational savings, it is important to explore the area loss in our discretization. We turn once again to our standard fluid-structure interaction problem. We run that simulation on successively finer grids until time $t = 2$. For both the RBF-IB method and the PL-IB method, we measure the initial area of the object for the same initial configuration of points. We then measure the area at time $t = 2$ and compute the percentage change in area.

In order to get an accurate estimate of the area in both methods, we fit an RBF interpolant to each object's Lagrangian markers (data sites for the RBF-IB method and a subset of the sample sites for the PL-IB method). We then sample that interpolant at a fixed number of points (400 points), and use the trapezoidal rule to compute the area. As was mentioned earlier, we ensure that the initial ellipse has the same area as the target circle by picking its radii to be $a = 2r$ and $b = 0.5r$, where $r = 0.1$ is the radius of the target circle. The exact area is then $\frac{\pi}{100}$. Our approach of sampling each object and computing the area with the trapezoidal rule gives an area estimate that agrees with this value up to 7 digits at $t = 0$. We record the results of our refinement study in Table 10. From the table, it is clear that the area loss for fixed $N_d = 25, 50$ and $N_d = 0.25N_s$ are all close to each other. On the coarsest grid, it appears that smaller values of N_d result in lower area loss. The area losses for $N_d = 50$ match with those given by the PL-IB method (results not shown), except in the case of the coarsest grid, where the PL-IB method gives almost a 1% area loss. The convergence is initial second-order but quickly saturates. This saturation is likely due to two sources of error: the first is the interpolation of velocities to the Lagrangian markers, which does not preserve the divergence-free nature of the fluid velocity; the second is the fact that the time-integration

itself is not specifically designed to preserve area. Nevertheless, it is clear from this study that the RBF-IB method produces similar area losses to the PL-IB method despite using a smaller number of Lagrangian markers to move the structure through the fluid.

Another measure of interest is the maximum stable time-step size afforded by each method. We measure this by increasing the time-step size in small increments and observing the forces produced on the structure in the fluid-structure interaction test. Using a time-step that is too large can result in the forces blowing up and the simulation halting. We immediately note that the PL-IB method allows a maximum time-step size of $\Delta t = 2 \times 10^{-4}$ on the 32×32 grid when $N_{\mathrm{s}} = 50$ IB points are used, and a maximum time-step size of $\Delta t = 10^{-4}$ on the 64×64 grid when $N_{\mathrm{s}} = 100$ IB points are used. We use these values of Δt as the starting point when testing for the time-step sizes allowed by the RBF-IB method, and increase the value of Δt in increments of 10^{-4}. We found that on the 32×32 grid, the RBF-IB method allows us to take time-steps that are $3\times$ larger than the time-steps allowed by the traditional IB method; on the 64×64 grid, the RBF-IB method can use time-steps that are $1.5\times$ larger than the time-steps allowed by the traditional IB method. This pattern holds both when $N_{\mathrm{d}} = 25$ and $N_{\mathrm{d}} = 50$ data sites are used.

In simulations involving platelet-like shapes (ellipses that attempt to maintain their elliptical configuration), we found that the RBF-IB method allows time-step sizes that are $6\times$ larger than those allowed by the PL-IB method on a 32×32 grid, and $3\times$ larger than those allowed by the PL-IB method on a 64×64 grid. This is likely due to the fact that platelet simulations involve smaller deformations than those seen in the standard fluid-structure interaction test.

5.3 Energy Estimates

In this section, we compute energy estimates for the RBF-IB method in the context of our standard fluid-structure interaction problem. We run our simulation out to time $t = 2.0$ on two grid sizes, 32×32 and 64×64, with time-step sizes $\Delta t = 2 \times 10^{-4}$ and $\Delta t = 10^{-4}$ respectively. We use $N_{\mathrm{d}} = 50$ data sites.

In this test, one expects the changes in energy to be mainly due to the deformation of the stiff elastic object. Eventually, the energy of the system must damp out as the elastic object reaches its target configuration. We compute the change in energy to demonstrate that the energy is bounded within the RBF-IB simulation. The energy change in a time-step is computed as the sum of the difference in kinetic energy of the fluid over the time-step and the change in potential energy of the elastic object. This can be written as

$$E = \int_{fluid} \rho \boldsymbol{u}^{n+1} \cdot \boldsymbol{u}^{n+1} - \int_{fluid} \rho \boldsymbol{u}^{n} \cdot \boldsymbol{u}^{n} + \Delta t \int_{\boldsymbol{X}} \boldsymbol{F} \cdot \frac{\partial \boldsymbol{X}}{\partial t} \tag{18}$$

Here, the Lagrangian force $\boldsymbol{F}$ is computed at time level $n+1$ at the sample sites. The $\frac{\partial \boldsymbol{X}}{\partial t}$ term is computed by applying the evaluation matrix $\mathcal{E}_{\mathrm{s}}$ to the velocities obtained at the data

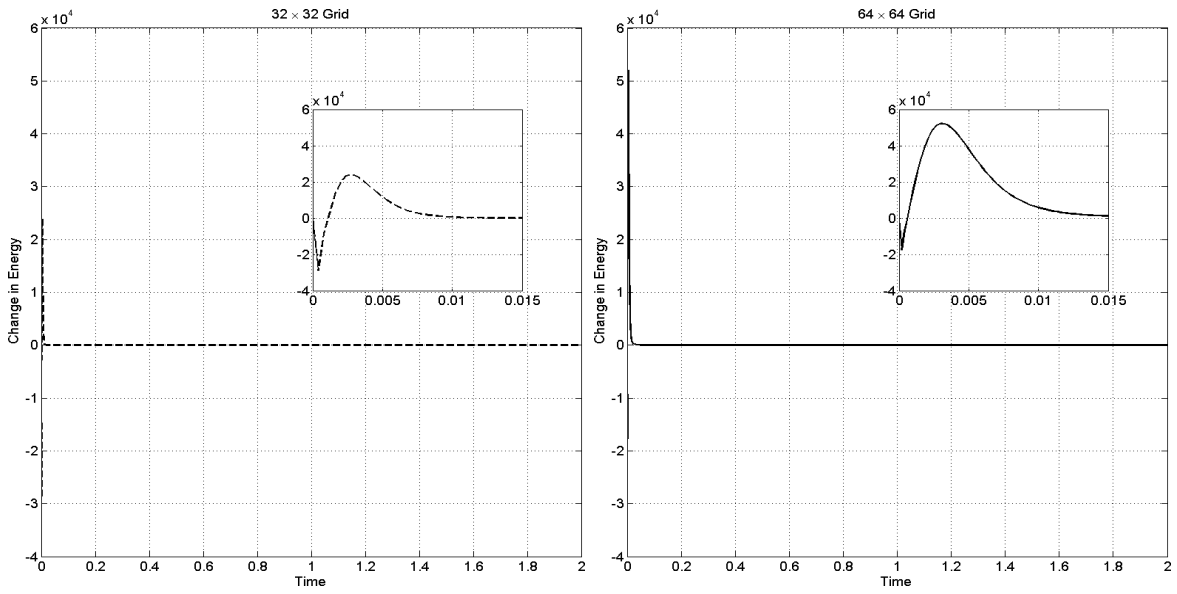

Fig. 2. Change in energy as a function of time in the RBF-IB method. The figure on the top left shows the change in energy over a time-step as a function of time for the standard fluid-structure interaction test on a 32×32 grid. The figure on the top right shows the same quantity on a 64×64 grid. We use $N_{\mathrm{d}} = 50$ data sites for both grid sizes. The inset plots show the initial spikes corresponding to the change from an ellipse to a circle which are difficult to see in the main plots.

sites. This gives us sample site velocities, allowing us to compute dot products with the $\boldsymbol{F}$ terms. We compute the discrete analog of Equation (18) for the fluid-structure interaction problem.

The results of this test are shown in Figure 2. Both plots show the change in energy of the system for the fluid-structure interaction problem on a 32×32 grid (left) and a 64×64 grid (right). Here, the fluid starts off stationary, so the initial kinetic energy is zero. However, the elliptical elastic object starts off under tension, since its target configuration is a circle. This means that the initial elastic potential energy of the system is high (though negative by convention). As the elastic object attempts to minimize its elastic potential energy, its deformation drives a change in the kinetic energy of the fluid, causing the kinetic energy of the fluid to increase from its initial value of zero to some maximum. However, the elastic object soon attains something close to its reference configuration, causing the kinetic energy of the fluid to drop sharply. The spikes in both the left and right sides of Figure 2 correspond to that rise and fall in kinetic energy and the trending of the potential energy to zero on both grids, and can be seen more clearly in the inset plots. The viscosity of the fluid causes the kinetic energy to eventually damp out almost completely, with minor perturbations due to possible deformations of the elastic object. The energy of the system continues to decrease as the object becomes more and more circular. In fact, our estimates show that the change in energy is negative, indicating that our method is dissipative. The results are similar for $N_{\mathrm{d}} = 25$ (not shown), though using more data sites appears to make our method less dissipative on this particular test problem.

5.4 Timings for Platelet simulations

We now present timings of simulations involving platelet-like shapes. The setup here is different from the standard fluid-structure interaction test. We place ellipses ($r = 0.05$, $a = 2r$, $b = 0.5r$) at the left end of a $[0, 2] \times [0, 1]$ domain that resembles a channel. These ellipses represent platelets, and they attempt to maintain their elliptical shapes, *i.e.*, their configuration at time $t = 0$ is their preferred configuration. We apply a background force that would result in parabolic velocity field in the absence of the platelets, with a density $\rho = 1.0$ and a non-dimensionalized viscosity of $\mu = 8.0$. The field has a maximum velocity value of $u_{max} = 5.0$, with no-slip boundary conditions on the top and bottom of the domain and periodic boundary conditions at the left and right ends. A platelet is removed from the domain if its center of mass crosses the location $x = 1.9$.

Figure 3 shows timings for three grid sizes for each method as a function of the number of platelets (N_{p}) being simulated. The number of sample sites was fixed at $N_{\mathrm{s}} = 100$ for both methods and the number of data sites for the RBF-IB method was set to $N_{\mathrm{d}} = 25$ for one set of tests and then to $N_{\mathrm{d}} = 50$ for the next set. We plot the average time per time-step as a function of the number of platelets; this was computed by running simulations on each grid for 10^5 time-steps, and dividing the total wall-clock time by the number of time-steps. We average this over three runs of each simulation.

While the cost of platelet operations always increases as we increase the number of platelets, the increase in cost is slower (and the absolute cost) for the RBF-IB method due to the RBF representation. For example, for $N_{\mathrm{p}} = 60$, the PL-IB method directly spreads forces from, interpolates to, and moves a total of 6000 IB points (twice per time-step due to the RK2 scheme) while the RBF-IB method with $N_{\mathrm{d}} = 25$ data sites computes forces at 6000 points, and interpolates velocities to (and moves) only 1620 points twice per time-step. If the number of platelets is doubled to $N_{\mathrm{p}} = 120$, the PL-IB method now computes forces at, spreads from, interpolates to, and moves 12000 points twice per time-step, whereas the RBF-IB method computes forces at 12000 points, but interpolates velocities to and moves only 3240 points twice per time-step. The cost of the RBF-IB method shows better than linear scaling with respect to the number of platelets on all the tested grid sizes for these reasons. Furthermore, it is clear that there is not much of a difference in computational cost between using $N_{\mathrm{d}} = 25$ data sites and $N_{\mathrm{d}} = 50$ data sites.

Effect of the RBF representation on the fluid solver

In previous work [30], we showed that using an RBF interpolant for geometric modeling is more computationally efficient (for a given accuracy) than using piecewise quadratics and finite differences. However, that benefit alone does not explain the computational efficiency of the RBF-IB method over the PL-IB method that we see in Figure 3.

To fully understand the speedup seen with the RBF-IB method, it is important to understand how the costs are distributed between the different operations (platelet operations and fluid

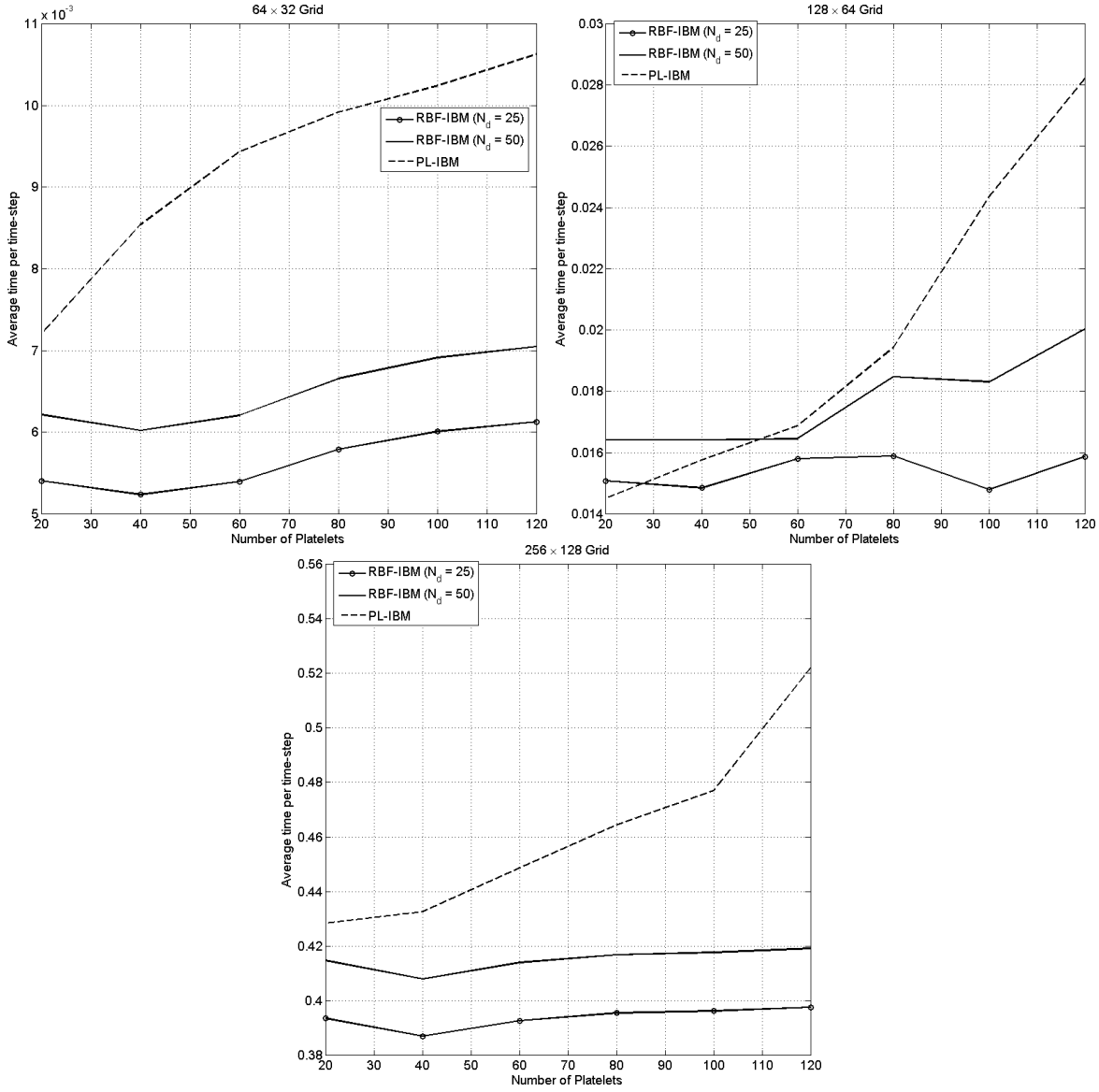

Fig. 3. Average time per time-step for 10^5 time-steps of each simulation method as a function of the number of platelets. In the first row, the figure on the left shows timings on a 64×32 grid and the figure on the right on a 128×64 grid. The figure below shows timings on a 256×128 grid. The time-step was set to $\Delta t = 10^{-4}$ for the figures on the top row, and was set to $\Delta t = 10^{-5}$ for the figure on the bottom.

solves) in both IB methods. We show the results for $N_{\text{p}} = 60$ platelets in Table 11. Clearly, as h is reduced, both IB codes spend more time in the fluid solver than on platelet operations. However, the RBF-IB method clearly spends less time in the fluid solver than the PL-IB method does as we refine the background Eulerian grid.

Indeed, this unexpected result is what gives the RBF-IB method an edge even when the cost of fluid solves dominates the cost of platelet operations. We hypothesize that this may be caused by the RBF representation producing smoother Lagrangian forces than the finite difference model used in the PL-IB method. Our experiments show that the RBF-IB code

needs fewer iterations in the linear solver used in the pressure projection– anywhere from $10 - 30\%$ fewer than the identical fluid solver used in the PL-IB method, depending on the time-step size and the grid resolution, with larger savings on finer grids and smaller time-step sizes.

Grid Size	% time in fluid solver (RBF-IB)	% time in fluid solver (PL-IB)
32×32	33.7	32.1
64×64	56.2	58.0
128×128	65.4	79.3

Table 11
Percentage of time per time-step spent in fluid solver as a function of grid size by both methods for $N_{\mathrm{p}} = 60$ platelets. The percentages for the RBF-IB method are the same for both $N_{\mathrm{d}} = 25$ and $N_{\mathrm{d}} = 50$ data sites, with the total time for the latter being larger. All results use $N_{\mathrm{s}} = 100$ sample sites (or IB points in the PL-IB method) per platelet.

5.5 Platelet Aggregation

We now present the results of a platelet aggregation simulation with the RBF-IB simulation. We used the same boundary conditions, domain size, fluid properties and Poisseuille flow as in the previous subsection, but allow platelets to form links with other platelets and a portion of the chamber wall ($x = 0.4$ to $x = 0.7$) at the sample sites ($N_{\mathrm{s}} = 100$ per platelet). We used $N_{\mathrm{d}} = 50$ data sites per platelet, making the data sites a subset of the sample sites for convenience of visualization, and then visualize the data sites and the links between sample sites. We allowed each platelet to form up to 10 links in total, either with the wall or with a neighbor; we allow links to cross each other for the purpose of simplicity, though this is usually prohibited in a platelet simulation. The simulation was run on a 128×64 grid with a time-step of $\Delta t = 10^{-4}$.

Each platelet is initially an ellipse with radii $a = 0.06$ and $b = 0.015$. We initialize the platelets so that their centers of masses are at locations $(0.5, 0.02)$, $(0.64, 0.02)$, $(0.78, 0.02)$, $(0.55, 0.07)$, $(0.68, 0.07)$, $(0.4, 0.045)$, $(0.23, 0.045)$ and $(0.65, 0.14)$. We chose these locations to ensure that three platelets lay on the wall, with three close enough to bind to the three bound to the wall, and two slightly further away. Each platelet attempts to maintain its initial elliptical shape. We then started the simulation and ran it to time $t = 2.4$. The results are shown in Figure 4. The figure shows both the velocity field and the platelet aggregate for a portion of $[0, 2] \times [0, 1]$ domain, the data sites on each platelet and the links between the sample sites corresponding to those particular data sites on the platelet.

There are two interesting features in Figure 4. The first is that the fluid flow gets diverted around the growing aggregate, a consequence of the size of the aggregate and the dynamics of the problem that mimics what one would hope to see in a realistic platelet aggregation simulation. The second feature is that some platelets are quite deformed, *e.g.*, the platelet with center of mass approximately at $(0.5, 0.02)$, or its neighbor above and to its left. This

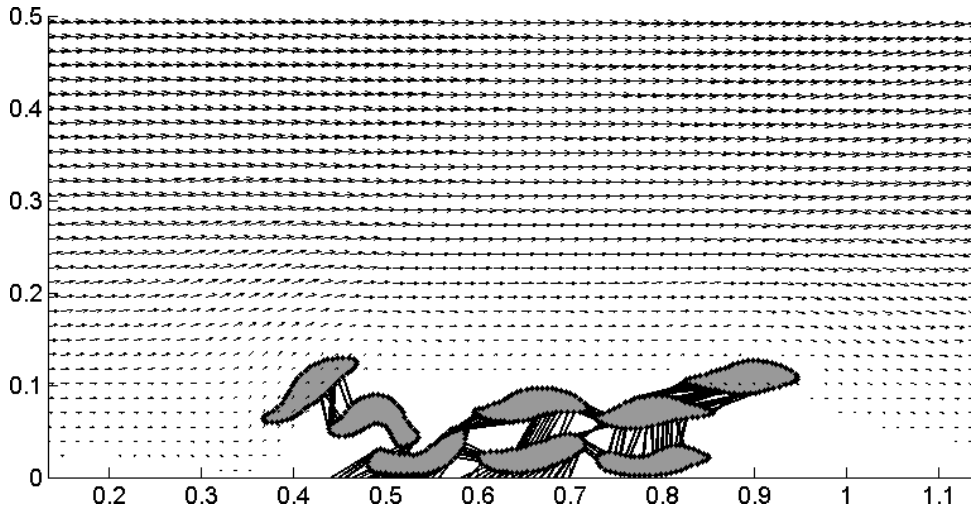

Fig. 4. Results of a platelet aggregation simulations with the RBF-IB method. The figure shows a zoomed-in snapshot of a platelet aggregation simulation achieved with the RBF-IB method with a time-step of $\Delta t = 10^{-4}$. The snapshot was taken at simulation time $t = 2.4$. The simulation was run on a 128×64 grid on a $[0, 2] \times [0, 1]$ domain. The arrows show the magnitude and direction of the velocity field.

is a consequence (and function) of the stiffness of each platelet, the shear rate of the flow and the number of links we allow each platelet to form. Higher platelet stiffness, lower shear rates and/or fewer (or weaker) links would lead to less deformation. The breaking model for inter-platelet and platelet-wall links can also affect the mechanics of aggregation. We note that our RBF model did not run into any instabilities in this simulation even when we ran it out to a time at which all the platelets (except the three closest to the wall) had left the domain.

6 Summary

The IB method, as a numerical methodology for applications involving fluid structure interactions, naturally lends itself to our problem of interest: simulating platelet aggregation during blood clotting. In this application, platelets are modeled as immersed elastic structures whose shapes change dynamically in response to blood flow and chemistry. In previous work [30], we discussed several geometric representations for platelets and compared them to the representation used within the traditional IB method. We concluded that an RBF geometric model for platelets would prove advantageous in several ways.

In this work, we explored the ramifications of using the RBF geometric model within the IB method, and compared the behavior of this new RBF-IB method against that of the traditional IB method. We discussed the issue of selecting an appropriate shape parameter for the RBF-IB method. We then presented a series of convergence studies for measuring errors and convergence both in the velocity field and in the representation of the immersed elastic structure. We went on to compare the computational costs incurred by both methods

in the context of platelet simulations. We then compared the area conservation properties of both methods and also the time-step restrictions on both. We also remarked on the energy properties of our method.

We conclude the following:

- The RBF-IB method facilitates the use of constitutive models within the IB method without having to resort to lower-order approximations;
- The RBF-IB method (especially with $N_{\mathrm{d}} = 50$) produces lower errors in both the velocity field and the immersed elastic structure in convergence studies;
- The RBF-IB is more computationally efficient than the traditional IB method for both $N_{\mathrm{d}} = 25$ and $N_{\mathrm{d}} = 50$, both due to the utilization of a small number of data sites and due to smoother forces being spread into the fluid resulting in a faster convergence from the fluid solver; and
- The RBF-IB allows for larger time-step sizes than those allowed in the traditional IB method for a given grid size.

In previous work [23], a sufficient condition for unconditional stability of an implicit IB method was established. The proof relied on the assumption that the set of points from which IB forces are spread is the same as that to which grid velocities are interpolated to update IB point positions. The RBF-IB method does not meet that condition, and it remains to be seen how this would impact an implict version of our method. Finally, an issue with the RBF-IB method is that it is dependent on the parametrization of the immersed elastic objects. For objects that are not easily parameterized in terms of circles and ellipses, the use of the RBF model as presented in our work (wherein the RBFs are restricted to the circle) may not be ideal. In the future, we thus hope to explore the use of RBFs in a meshfree variational form within the IB method so as to be able to easily evaluate constitutive models on arbitrary shapes.

Acknowledgments: We would like to acknowledge useful discussions concerning this work within the CLOT group at the University of Utah, with Professor Boyce Griffith of New York University and with Professor Robert Guy of the University of California, Davis. The first, third and fourth authors acknowledge funding support under NIGMS grant R01-GM090203. The second author acknowledges funding support under NSF-DMS grant 0540779 and NSF-DMS grant 0934581.

A Algorithm for computing platelet forces with RBFs

We now describe the implementation of the constitutive models outlined in Section 3. We present algorithms for computing platelet forces in 2D.

Notation: In the description of the algorithms below we use standard matrix-vector op-

erations such as multiplication as well as non-standard operations like element-by-element multiplication of matrices and vectors (sometimes called the Hadamard product). We denote this latter operation with the $\circ$ operator. For example, if $\underline{\boldsymbol{J}}$ and $\underline{\boldsymbol{L}}$ are $N_\mathrm{d} \times 2$ matrices and $\underline{R}$ is a vector of length N_d then the i^{th} row of $\underline{\boldsymbol{J}} \circ \underline{\boldsymbol{L}}$ and $\underline{R} \circ \underline{\boldsymbol{J}}$ are given by

$$(\underline{\boldsymbol{J}} \circ \underline{\boldsymbol{L}})_{i,1:2} = [(\underline{\boldsymbol{J}})_{i,1}(\underline{\boldsymbol{L}})_{i,1},\ (\underline{\boldsymbol{J}})_{i,2}(\underline{\boldsymbol{L}})_{i,2}]$$
$$(\underline{R} \circ \underline{\boldsymbol{L}})_{i,1:2} = [(\underline{R})_i(\underline{\boldsymbol{L}})_{i,1},\ (\underline{R})_i(\underline{\boldsymbol{L}})_{i,2}]$$

where $i = 1, \ldots, N_\mathrm{d}$.

We define $\underline{\boldsymbol{\tau}}_\mathrm{d} = \mathcal{D}^1_{\lambda^\mathrm{d}} \underline{\boldsymbol{X}}_\mathrm{d}(t)$, the $N_\mathrm{d} \times 2$ matrix of tangent vectors at the data sites at time t and $\underline{\|\boldsymbol{\tau}_\mathrm{d}\|}$, the N_d vector containing the two-norm of each row of $\underline{\boldsymbol{\tau}}_\mathrm{d}$. The algorithm for computing platelet elasticity is as follows:

(1) Initialization ($t = t_0$): After creating and storing the RBF evaluation matrix as in Equation (10) and differentiation matrices as in Equations (11) and (12), compute for each platelet:
   (a) The rest lengths for the tension force at the data sites: $\underline{l_0} = \underline{\boldsymbol{\tau}}_\mathrm{d} = \mathcal{D}^1_{\lambda^\mathrm{d}} \underline{\boldsymbol{X}}_\mathrm{d}(t_0)$.
   (b) The bending-resistant force term for the platelet's initial configuration at the data sites, $\mathcal{D}^4_{\lambda^\mathrm{s}} \underline{\boldsymbol{X}}_\mathrm{d}(t_0)$.

(2) For each time step ($t = t_k$, $k \geq 1$), compute for each platelet:
   (a) The length of the tangent vectors $\underline{\boldsymbol{\tau}}_\mathrm{d} = \mathcal{D}^1_{\lambda^\mathrm{d}} \underline{\boldsymbol{X}}_\mathrm{d}(t_k)$ at the data sites: $\underline{\|\boldsymbol{\tau}_\mathrm{d}\|}$; and the unit tangents at the data sites: $\underline{\hat{\boldsymbol{\tau}}}_\mathrm{d}$.
   (b) The tension at the data sites, using the constitutive model: $\underline{T}_\mathrm{d} = k_\mathrm{t}(\underline{\|\boldsymbol{\tau}_\mathrm{d}\|} - \underline{l_0})$.
   (c) The tension force at sample sites: $\underline{\boldsymbol{F}}^\mathrm{T}_\mathrm{s} = \mathcal{D}^1_{\lambda^\mathrm{s}} \underline{\boldsymbol{Z}}_\mathrm{d}$, where $\underline{\boldsymbol{Z}}_\mathrm{d} = \underline{T}_\mathrm{d} \circ \underline{\hat{\boldsymbol{\tau}}}_\mathrm{d}$.
   (d) The bending force at sample sites: $\underline{\boldsymbol{F}}^\mathrm{B}_\mathrm{s} = -k_\mathrm{b}\left(\mathcal{D}^4_{\lambda^\mathrm{s}} \underline{\boldsymbol{X}}_\mathrm{d}(t_k) - \mathcal{D}^4_{\lambda^\mathrm{s}} \underline{\boldsymbol{X}}_\mathrm{d}(t_0)\right)$.
   (e) The interplatelet cohesion force from Equation (17) at the sample sites: $\underline{\boldsymbol{F}}^\mathrm{C}_\mathrm{s}$.
   (f) The total Lagrangian force at the sample sites: $\underline{\boldsymbol{F}}_\mathrm{s} = \underline{\boldsymbol{F}}^\mathrm{T}_\mathrm{s} + \underline{\boldsymbol{F}}^\mathrm{B}_\mathrm{s} + \underline{\boldsymbol{F}}^\mathrm{C}_\mathrm{s}$.

B Time-stepping for the PL-IB method

Here, we present the steps of the traditional Immersed Boundary algorithm when used with the RK2 time-stepping scheme from [3].

(1) Advance the structure to time level $t^{n+1/2}$ using the current velocity field on the grid $\boldsymbol{u}^n_g$. This is done by updating each IB point $\boldsymbol{X}_q$ (for each q) using the equation

$$\boldsymbol{X}^{n+1/2}_q = \boldsymbol{X}^n_q + \frac{\Delta t}{2}\boldsymbol{U}^n_q \equiv \boldsymbol{X}^n_q + \frac{\Delta t}{2}\sum_g \boldsymbol{u}^n_g \delta_h(\boldsymbol{x}_g - \boldsymbol{X}^n_q)h^2, \tag{B.1}$$

where h is the fluid grid spacing and δ_h is a discrete approximation to a two-dimensional δ-function. Here, $\boldsymbol{x}_g$ and $\boldsymbol{X}^{n+1/2}_q$ are the coordinates of grid point g and IB point q, respectively

(2) The resultant $\boldsymbol{F}_q^{n+1/2}$ of all of the force contributions that act on an IB point $\boldsymbol{X}_q^{n+1/2}$ is calculated for each q.

(3) These forces are distributed to the Eulerian grid used for the fluid dynamics equations using a discrete version of Equation (3):

$$\boldsymbol{f}_g^{n+1/2} \equiv \boldsymbol{f}^{n+1/2}(\boldsymbol{x}_g) = \sum_q \boldsymbol{F}_q^{n+1/2} \delta_h(\boldsymbol{x}_g - \boldsymbol{X}_q^{n+1/2}) dq. \tag{B.2}$$

Here, $\boldsymbol{F}_q^{n+1/2}$ is the Lagrangian force (per unit q) on the IB point, dq is the increment in parameter q between consecutive discrete sample sites, and δ_h is the same approximate δ-function as used in Equation B.1.

(4) With the fluid force density $\boldsymbol{f}_g^{n+1/2}$ now known at each grid point, the fluid velocity is updated taking a half step ($\Delta t/2$) with a discrete Navier-Stokes solver. As in [3], we use a fractional-step projection method. First, a backward Euler discretization of the momentum equations is used. The pressure that enforces discrete incompressibility is determined [17]. This gives us the velocity field $\boldsymbol{u}_g^{n+1/2}$, the mid-step approximation required in an RK2 method.

(5) Using the mid-step fluid velocity $\boldsymbol{u}_g^{n+1/2}$ and the mid-step IB point positions $\boldsymbol{X}_q^{n+1/2}$, update the IB points $\boldsymbol{X}_q^n$ for each q to the time level t^{n+1} using

$$\boldsymbol{X}_q^{n+1} = \boldsymbol{X}_q^n + \Delta t \boldsymbol{U}_q^{n+1/2} \equiv \boldsymbol{X}_q^n + \Delta t \sum_g \boldsymbol{u}_g^{n+1/2} \delta_h(\boldsymbol{x}_g - \boldsymbol{X}_q^{n+1/2}) h^2, \tag{B.3}$$

where δ_h is the same approximate δ-function we have used throughout.

(6) Update the velocity $\boldsymbol{u}_g^n$ to time-level t^{n+1} using the mid-step velocity $\boldsymbol{u}_g^{n+1/2}$ and the force $\boldsymbol{f}_g^{n+1/2}$. The mid-step velocities are advected, while a Crank-Nicolson scheme is used for time-stepping the momentum equations. The pressure projection gives us the discretely-incompressible velocity field $\boldsymbol{u}_g^{n+1}$. Note that this step could have been performed as soon as $\boldsymbol{u}_g^{n+1/2}$ was computed. It is independent of step (5).

C Time-stepping for the RBF-IB method

The RBF-IB method is time-stepped using the same RK2 method as above, with a few changes to incorporate data sites and sample sites.

(1) Advance the structure to time level $t^{n+1/2}$ using the current velocity field $\boldsymbol{u}^n$. This is done by updating the *data sites* $(\boldsymbol{X}_{\mathrm{d}})_j^n$ by a discrete analog of Equation (4)

$$(\boldsymbol{X}_{\mathrm{d}})_j^{n+1/2} = (\boldsymbol{X}_{\mathrm{d}})_j^n + \frac{\Delta t}{2}(\boldsymbol{U}_{\mathrm{d}})_j^n \equiv (\boldsymbol{X}_{\mathrm{d}})_j^n + \frac{\Delta t}{2} \sum_g \boldsymbol{u}_g^n \delta_h(\boldsymbol{x}_g - (\boldsymbol{X}_{\mathrm{d}})_j^n) h^2. \tag{C.1}$$

(2) Generate a new set of sample sites $\underline{\boldsymbol{X}}_{\mathrm{s}}(t_{n+1/2})$ by applying the RBF evaluation operator

to the data sites $\underline{\boldsymbol{X}}_{\rm d}^{n+1/2} := \underline{\boldsymbol{X}}_{\rm d}(t_{n+1/2})$, *i.e.*,

$$\underline{\boldsymbol{X}}_{\rm s}(t_{n+1/2}) = \mathcal{E}_{\rm s}\underline{\boldsymbol{X}}_{\rm d}(t_{\rm new}). \tag{C.2}$$

(3) The total force at the sample sites $\underline{\boldsymbol{F}}_{\rm s}^{n+1/2}$ is calculated using the algorithm from Appendix A.

(4) These forces are distributed to the Eulerian grid used for the fluid dynamics equations using a discrete version of Equation (3):

$$\boldsymbol{f}_g^{n+1/2} \equiv \boldsymbol{f}^{n+1/2}(\boldsymbol{x}_g) = \sum_q \boldsymbol{F}_q^{n+1/2}\delta_h(\boldsymbol{x}_g - (\boldsymbol{X}_{\rm s})_q^{n+1/2})dq. \tag{C.3}$$

Here, $\boldsymbol{x}_g$ and $(\boldsymbol{X}_{\rm s})_q^{n+1/2}$ are the coordinates of grid point g and sample site q, respectively, $\boldsymbol{F}_q^{n+1/2}$ is the Lagrangian force (per unit q) on the sample site, dq is the increment in parameter q between consecutive discrete sample sites, and δ_h is the same approximate δ-function as used in Equation C.1.

(5) With the fluid force density $\boldsymbol{f}_g^{n+1/2}$ now known at each grid point, the fluid velocity is updated taking a half step ($\Delta t/2$) with a discrete Navier-Stokes solver. Again, we use a fractional-step projection method, with a backward Euler discretization of the momentum equation, and a projection to determine the pressure that enforce incompressibility [17]. This gives us the velocity field $\boldsymbol{u}_g^{n+1/2}$, the mid-step approximation required in an RK2 method.

(6) Using the mid-step fluid velocity $\boldsymbol{u}_g^{n+1/2}$ and the mid-step *data site* positions $(\boldsymbol{X}_{\rm d})_j^{n+1/2}$, update the *data sites* $(\boldsymbol{X}_{\rm d})_j^{n+1/2}$ for each j to the time level t^{n+1} using

$$(\boldsymbol{X}_{\rm d})_j^{n+1} = (\boldsymbol{X}_{\rm d})_j^n + \Delta t(\boldsymbol{U}_{\rm d})_j^{n+1/2} \equiv (\boldsymbol{X}_{\rm d})_j^n + \Delta t\sum_g \boldsymbol{u}_g^{n+1/2}\delta_h(\boldsymbol{x}_g - (\boldsymbol{X}_{\rm d})_j^{n+1/2})h^2, \tag{C.4}$$

where δ_h is the same approximate δ-function we have used throughout.

(7) Update the velocity $\boldsymbol{u}_g^n$ to time-level t^{n+1} using the mid-step velocity $\boldsymbol{u}_g^{n+1/2}$ and the force $\boldsymbol{f}_g^{n+1/2}$. The mid-step velocities are advected, while a Crank-Nicolson scheme is used for time-stepping the momentum equations. The pressure projection gives us the discretely-incompressible velocity field $\boldsymbol{u}_g^{n+1}$.

Observe that the data sites are updated twice per time-step in the RK2 scheme, but the sample sites are only generated once. Since the data sites are typically a fraction of the number of IB points from the PL-IB method, the computational cost is significantly lower for the RBF-IB method, even factoring in the interpolation and the sample site generation.

References

[1] 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.

[2] R. P. Beyer, A computational model of the cochlea using the immersed boundary method, Journal of Computational Physics 98 (1992) 145–162.

[3] D. Devendran, C. S. Peskin, An immersed boundary energy-based method for incompressible viscoelasticity, J. Comput. Physics 231 (14) (2012) 4613–4642.

[4] R. Dillon, L. Fauci, A. Fogelson, D. Gaver, Modeling biofilm processes using the Immersed Boundary method, Journal of Computational Physics 129 (1996) 85–108.

[5] G. E. Fasshauer, Meshfree Approximation Methods with MATLAB, Interdisciplinary Mathematical Sciences - Vol. 6, World Scientific Publishers, Singapore, 2007.

[6] G. E. Fasshauer, L. L. Schumaker, Scattered data fitting on the sphere, in: M. Daehlen, T. Lyche, L. L. Schumaker (eds.), Mathematical Methods for Curves and Surface, Vol.2 of the Proceedings of the 4th Int. Conf. on Mathematical Methods for Curves and Surfaces, Lillehammer, Norway, Vanderbilt University Press, Nashville Tennessee, 1998.

[7] L. J. Fauci, A. L. Fogelson, Truncated newton methods and the modeling of complex immersed elastic structures, Communications on Pure and Applied Mathematics 66 (1993) 787–818.

[8] L. J. Fauci, C. S. Peskin, A computational model of aquatic animal locomotion, Journal of Computational Physics 77 (1988) 85–108.

[9] N. Flyer, G. B. Wright, Transport schemes on a sphere using radial basis functions, J. Comp. Phys. 226 (2007) 1059–1084.

[10] N. Flyer, G. B. Wright, A radial basis function method for the shallow water equations on a sphere, Proc. Roy. Soc. A 465 (2009) 1949–1976.

[11] A. Fogelson, A. Kuharsky, H. Yu, Computational modeling of blood clotting: Coagulation and three-dimensional platelet aggregation, in: W. Alt, M. Chaplain, M. Griebel, J. Lenz (eds.), Polymer and Cell Dynamics: Multicsale Modeling and Numerical Simulations, Birkhaeuser-Verlag, Basel, 2003, pp. 145–154.

[12] A. L. Fogelson, A mathematical model and numerical method for studying platelet adhesion and aggregation during blood clotting, Journal of Computational Physics 1 (1984) 111–134.

[13] A. L. Fogelson, R. D. Guy, Immersed-boundary-type models of intravascular platelet aggregation, Computer Methods in Applied Mechanics and Engineering 197 (2008) 2087 – 2104.

[14] B. Fornberg, C. Piret, A stable algorithm for flat radial basis functions on a sphere, SIAM J. Sci. Comp. 30 (2007) 60–80.

[15] E. J. Fuselier, G. B. Wright, A high-order kernel method for diffusion and reaction-diffusion equations on surfaces, Journal of Scientific Computing (2013) 1–31.
URL `http://dx.doi.org/10.1007/s10915-013-9688-x`

[16] B. Griffith, Simulating the fluid dynamics of natural and prosthetic heart valves using the immersed boundary method., Int J Appl Mech. 1 (2009) 137–177.

[17] R. D. Guy, A. L. Fogelson, Stability of approximate projection methods on cell-centered grids, Journal of Computational Physics 203 (2005) 517–538.

[18] S. Jackson, W. Nesbitt, S. Kulkarni, Signaling events underlying thrombus formation, J Thromb Haemost 1 (2003) 1602–1612.

[19] K. Jetter, J. Stöckler, J. D. Ward, Error estimates for scattered data interpolation on spheres, Math. Comput. 68 (226) (1999) 733–747.

[20] L. A. Miller, C. S. Peskin, When vortices stick: an aerodynamic transition in tiny insect flight, The Journal of Experimental Biology 207 (2004) 3073–3088.

[21] L. A. Miller, C. S. Peskin, A computational fluid dynamics of 'clap and fling' in the smallest insects, The Journal of Experimental Biology 208 (2005) 195–212.

[22] F. J. Narcowich, X. Sun, J. D. Ward, H. Wendland, Direct and inverse Sobolev error estimates for scattered data interpolation via spherical basis functions, Found. Comput. Math. 7 (3) (2007) 369–390.

[23] E. P. Newren, A. L. Fogelson, R. D. Guy, R. M. Kirby, Unconditionally stable discretizations of the immersed boundary equations, Journal of Computational Physics 222 (2007) 702–719.

[24] C. S. Peskin, Numerical analysis of blood flow in the heart, Journal of Computational Physics 25 (1977) 220–252.

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

[26] C. S. Peskin, D. M. McQueen, Modeling prosthetic heart valves for numerical analysis of blood flow in the heart, Journal of Computational Physics 37 (1980) 113–132.

[27] C. S. Peskin, D. M. McQueen, A three-dimensional computational method for blood flow in the heart: I. immersed elastic fibers in a viscous incompressible fluid, Journal of Computational Physics 81 (1989) 372–405.

[28] C. Piret, The orthogonal gradients method: A radial basis functions method for solving partial differential equations on arbitrary surfaces, J. Comput. Phys. 231 (20) (2012) 4662–4675.

[29] B. Savage, E. Saldivar, Z. M. Ruggeri, Initiation of platelet adhesion by arrest onto fibrinogen or translocation on von Willebrand factor, Cell 84 (1996) 289–297.

[30] 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 (0) (2013) 58 – 77.
URL http://www.sciencedirect.com/science/article/pii/S0168927412001663

[31] H. Wendland, Scattered data approximation, vol. 17 of Cambridge Monographs on Applied and Computational Mathematics, Cambridge University Press, Cambridge, 2005.

[32] L. Yao, A. L. Fogelson, Simulations of chemical transport and reaction in a suspension of cells i: an augmented forcing point method for the stationary case, International Journal for Numerical Methods in Fluids 69 (11) (2012) 1736–1752.
URL http://dx.doi.org/10.1002/fld.2661

[33] H. Yu, Three dimensional computational modeling and simulation of platelet aggregation on parallel computers, Ph.D. thesis, University of Utah (2000).