Numer. Math. Theor. Meth. Appl.
doi: 10.4208/nmtma.2014.1308si

Vol. **7**, No. 4, pp. 413-434
November 2014

Exact Singularity Subtraction from Boundary Integral Equations in Modeling Vesicles and Red Blood Cells

Alexander Farutin and Chaouqi Misbah*

Université Grenoble I/CNRS, Laboratoire Interdisciplinaire de Physique/UMR5588, Grenoble F-38041, France.

Received 15 October 2013; Accepted 14 July 2014

Available online 11 November 2014

Abstract. The study of vesicles, capsules and red blood cells (RBCs) under flow is a field of active research, belonging to the general problematic of fluid/structure interactions. Here, we are interested in modeling vesicles, capsules and RBCs using a boundary integral formulation, and focus on exact singularity subtractions of the kernel of the integral equations in 3D. In order to increase the precision of singular and near-singular integration, we propose here a refinement procedure in the vicinity of the pole of the Green-Oseen kernel. The refinement is performed homogeneously everywhere on the source surface in order to reuse the additional quadrature nodes when calculating boundary integrals in multiple target points. We also introduce a multi-level look-up algorithm in order to select the additional quadrature nodes in vicinity of the pole of the Green-Oseen kernel. The expected convergence rate of the proposed algorithm is of order $\mathcal{O}(1/N^2)$ while the computational complexity is of order $\mathcal{O}(N^2 \ln N)$, where N is the number of degrees of freedom used for surface discretization. Several numerical tests are presented to demonstrate the convergence and the efficiency of the method.

AMS subject classifications: 64N38, 65N80, 74F10, 76Z05

Key words: Stokes flow, fluid structure interaction, boundary integral method, red blood cells, singularity subtraction.

1. Introduction

Blood is a complex fluid that is primarily composed of red blood cells (RBCs), which occupy (in a healthy human body) about 45% of the blood volume. The rest consists of plasma, while the other blood elements (white blood cells, platelets, etc.) take up less than 1% of the total blood volume.

The complex character of blood flow results from an intimate coupling between the shape of RBCs and the fluid dynamics of the ambient plasma, which leads to a rich

*Corresponding author. *Email addresses:* alexandr.farutin@ujf-grenoble.fr (A. Farutin), chaouqi.misbah@ujf-grenoble.fr (C. Misbah)

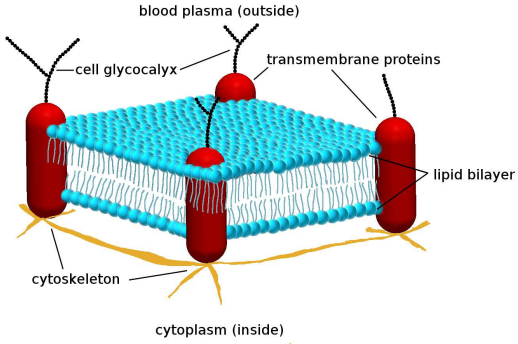

Figure 1: A schematic view of the RBC membrane showing the phospholipid bilayer, the spectrin network (called cytoskeleton) beneath and several membrane and transmembrane proteins.

set of RBC morphologies in the blood circulatory system. Understanding the selection of shapes and dynamics among a large manifold of possibilities, the collective effects, the spatiotemporal organizations is a challenging problem. This type of complexity is a characteristic property of non-equilibrium dissipative systems for which general thermodynamic principles, such as minimization of energy, maximization of entropy, etc., cannot be applied.

Due to the predominance of the concentration of RBCs in blood, blood flow is dictated primarily by RBCs. A RBC is made of a bilayer of phospholipid (see Fig. 1). In addition, several proteins are anchored on this membrane (like ion channels), while beneath the membrane there is a cytoskeleton, a network of proteins, called spectrin, that confers to the cell viscoelastic properties. Healthy human RBC has a biconcave shape at rest. Its size is about few μm. The interior of the RBC is made of an aqueous solution containing hemoglobin (and other species, like ATP-adenosine triphosphate). Hemoglobin is responsible for oxygen transport from lungs towards the microvasculature (arterioles, veinules and capillaries), where gas exchange takes place for tissue metabolism. ATP is, among other functions, responsible for vasodilation. The hemoglobin solution is believed to be a newtonian fluid. It has a viscosity which is of about 5 times that of the plasma (whose viscosity is close to that of water).

In this paper, we shall present modeling of these systems based on the boundary integral(BI) formulation. We propose a method for calculation of BIs in flows of complex geometry. The method combines the singularity subtraction (SS) technique with the refinement in vicinity of the pole of the Green kernels in order to achieve good precision of singular and near-singular integration without excessive increase of computation time with respect to traditional integration techniques.

2. Modeling of blood flow

2.1. Modeling of red blood cells

Due to the complexity of the RBC modeling, especially because of its cytoskeleton, two model systems have been considered as alternative systems serving as a basis for

more elaborate models. The first system is the vesicle model which mimics the bilayer of the RBC, while the second system is the capsule model which mimics the RBC cytoskeleton. A vesicle is a closed membrane which is made of only a phospholipid bilayer (it is, unlike RBC, devoid of any proteins), which is fluid at physiological as well as at room temperatures. It has become now quite standard to fabricate them in the research laboratories, with various sizes (their typical radius ranges from few μm to about 100 μm). Their bending rigidity is about $20k_BT$ (and is close to that of the RBCs; k_B is the Boltzmann constant and T is the ambient temperature). They can be swelled or emptied thanks to osmosis, and thus their reduced volume ν (defined as $\nu = V/(4\pi/3)/(A/4\pi)^{3/2}$, where V and A are the actual vesicle volume and area respectively; for RBC $\nu \simeq 0.6$) can be changed. Their internal content can be varied (by adding polymer solutions in order to act on the internal viscosity, and so on). In particular, for $\nu \simeq 0.6 - 0.65$, vesicles exhibit a biconcave shape [1] which is similar to that of a human RBC. Different types of phospholipids can also be used in order to act, for example, on the bending rigidity. Vesicles share with RBCs an important property, namely, their membrane is locally inextensible.

The other biomimetic model system is represented by capsules. Capsules are quasi-spherical shells made of polymers. They can be obtained thanks to interfacial polymerization of a liquid drop. This leads to approximately spherical particles enclosed by a thin polymerized membrane with mechanical properties that depend on the fabrication process. For biological applications (as carrier of active substances, for example) the typical membranes are synthetic polymers such as poly-L-lysine, alginate, or polyacrylates. Their membrane is extensible, unlike that of vesicles and RBCs, and they are endowed with shear elasticity, mimicking the cytoskeleton of the RBCs. Their sizes can be quite diverse (say from a fraction of μm up to few mm; the later size is quite familiar in everyday life, the pharmaceutical capsule). The mechanics of capsule membrane belongs to the class of nonlinear elasticity [2].

The methods of force calculation both for vesicles and capsules can be found in many recent works, e.g., using finite elements [3], finite difference [4, 5], spectral parametrization [6–8], or Loop subdivision technique [9, 10]. An alternative approach is represented by mesoscopic models, derived from molecular models by coarsegraining [11–13]. Regardless of the model, specifying the positions of material points of the membrane in the 3D space fully determines its shape and the elastic forces exerted by the membrane on the surrounding medium (the fluids inside and outside the RBC in this case).

2.2. Modeling of hydrodynamic interactions

Once the membrane forces known, the hydrodynamics equations must be solved to find the flow inside and outside the RBCs. In general, this requires solving the 3D Navier-Stokes equations with complicated boundary conditions, specified by the shapes of the RBCs, the geometry of the flow and of the walls surrounding it, and by the forces exerted by the RBCs on the fluids. An additional difficulty lies in the fact that the shapes

of the cells change over the course of simulation. Several methods have been developed in the past to solve fluid/structure interactions, including the applications to vesicles and capsules [3–7, 9–21]. (See also reviews [2, 22, 23]). At the scale of individual RBCs or small blood vessels, the Reynolds number is quite small. In this case, the hydrodynamics of blood plasma or the internal solution of RBCs can be adequately described by Stokes equations

$$\eta \Delta \boldsymbol{u}(\boldsymbol{r}) - \boldsymbol{\nabla} p(\boldsymbol{r}) = -\boldsymbol{f}(\boldsymbol{r}), \quad \boldsymbol{\nabla} \cdot \boldsymbol{u}(\boldsymbol{r}) = 0, \tag{2.1}$$

where $\boldsymbol{u}(\boldsymbol{r})$ is the fluid velocity at point $\boldsymbol{r}$, $p(\boldsymbol{r})$ is the pressure, and $\boldsymbol{f}(\boldsymbol{r})$ is the density of the force applied on the liquid. The viscosity η, generally speaking, has different values inside η_{int} and outside η_{ext} the RBCs. If the only applied forces are acting at the membrane of RBCs or at the boundaries surrounding the flow, the powerful tools of fundamental solutions and BI methods can be used to solve the hydrodynamic part of the problem and thus calculate the velocities of material points at the membranes of RBCs. This method has been adopted by several groups for the study of vesicles and capsules [4, 5, 7, 8, 14, 24–26]. The found velocities can be used to update the conformation and to probe macroscopic properties of the flow, such as the effective viscosity [27].

Generally, the boundary conditions are as follows: no slip at the RBC membrane, jump of the normal component of viscous stress at the membrane is equal to the local force density applied by the RBC, impermeability of the membrane and the velocity filed tends to the imposed flow at infinity (for unbounded flow, as discussed here). In the BI method, the 3D Stokes equations and the boundary conditions at the 2D surfaces of RBCs and external boundaries are converted [28] to an integral equation

$$\begin{aligned}\frac{(\eta_{ext} + \eta^A_{int})}{2} u_i(\boldsymbol{r}) =& \eta_{ext} u_i^{\infty}(\boldsymbol{r}) + \sum_B \int\limits_{\Sigma^B} G_{ij}(\boldsymbol{r}, \boldsymbol{r'}) f_j(\boldsymbol{r'}) d^2 r' \\ &+ \sum_B (\eta_{ext} - \eta^B_{int}) \int\limits_{\Sigma^B} T_{ijk}(\boldsymbol{r}, \boldsymbol{r'}) u_j(\boldsymbol{r'}) n_k(\boldsymbol{r'}) d^2 r',\end{aligned} \tag{2.2}$$

where the indices A and B run over all surfaces (RBCs and vessel walls) present within the considered domain, $\boldsymbol{r}$ is any point on the surface Σ^A, $\boldsymbol{n}(\boldsymbol{r})$ is the outward normal to the surface at the point $\boldsymbol{r}$. It is convenient to look at the two integrals on the right hand side of (2.2) as at linear operators with the single-layer (Stokeslet) kernel $G_{ij}(\boldsymbol{r}, \boldsymbol{r'})$ and the double-layer (stresslet) kernel $T_{ijk}(\boldsymbol{r}, \boldsymbol{r'}) n_k(\boldsymbol{r'})$, respectively. The exact form of the kernels G and T depends on the boundary conditions of the problem. Often the free-space version (no boundary conditions other than the flow imposed at infinity) is considered, for which the kernels $G(\boldsymbol{r}, \boldsymbol{r}')$ and $T(\boldsymbol{r}, \boldsymbol{r}')$ depend only on the difference of the target position $\boldsymbol{r}$ and the source position $\boldsymbol{r}'$

$$G_{ij}(\boldsymbol{r}, \boldsymbol{r}') = \frac{1}{8\pi} \left(\frac{\delta_{ij}}{|\boldsymbol{r} - \boldsymbol{r}'|} + \frac{(\boldsymbol{r} - \boldsymbol{r}')_i (\boldsymbol{r} - \boldsymbol{r}')_j}{|\boldsymbol{r} - \boldsymbol{r}'|^3} \right), \tag{2.3a}$$

$$T_{ijk}(\boldsymbol{r},\boldsymbol{r}') = \frac{3}{4\pi}\frac{(\boldsymbol{r}-\boldsymbol{r}')_i(\boldsymbol{r}-\boldsymbol{r}')_j(\boldsymbol{r}-\boldsymbol{r}')_k}{|\boldsymbol{r}-\boldsymbol{r}'|^5}. \tag{2.3b}$$

The main advantage of the formulation (2.2) is that the velocities of material points at the membranes of RBCs can be calculated without any knowledge about the flow in the bulk of the fluids. Thus the problem of temporal evolution of RBC conformation can be solved with discretizing only the interfaces, which greatly reduces the number of degrees of freedom that must be tracked during the simulation. This simplification comes at a price: Two challenges are added by converting the Stokes equations to BI formulation: (i) computation of BI (2.2) requires calculation of the BI kernels for each pair of discretization points N (implying a complexity $\mathcal{O}(N^2)$) and (ii) the diverging behavior of the BI kernels (2.3) when the distance between the source and the target approaches 0 makes traditional techniques of numerical integration completely inapplicable or insufficiently precise.

Various approaches exist in the literature regarding the singular behavior of the BI kernels: One choice is to use coordinate transformation that introduces a multiplier that cancels the singular behavior of the BI kernel. This transformation can involve mapping a square to the mesh triangle [29] or using polar coordinates on the source surface with the origin of polar coordinates located either at the target point if it belongs to the source surface or at its projection on the target surface if it does not [6]. A similar approach can be used to calculate BIs over a triangular element, regardless if the target point belongs to the element or not [30]. A new technique based on multipole expansions of the singular kernels has been proposed recently [31].

Another method consists in regularizing the behavior of BI kernel by subtracting specially designed exact identities, which have a similarly diverging behavior near the pole of the BI kernel. One way to obtain such an identity is to substitute a linear flow into the BI identity valid for an arbitrary non-singular Stokes flow

$$\begin{aligned}\delta(\boldsymbol{r},\Sigma)u_i^\infty(\boldsymbol{r}) = &\int_\Sigma G_{ij}(\boldsymbol{r},\boldsymbol{r'})\left[\partial'_j u_k^\infty(\boldsymbol{r'}) + \partial'_k u_j^\infty(\boldsymbol{r'}) - \delta_{jk}p^\infty(\boldsymbol{r'})\right] n_k(\boldsymbol{r'})d^2r' \\ &+ \int_\Sigma T_{ijk}(\boldsymbol{r},\boldsymbol{r'})u_j^\infty(\boldsymbol{r'})n_k(\boldsymbol{r'})d^2r',\end{aligned} \tag{2.4}$$

where $\delta(\boldsymbol{r},\Sigma)$ is equal to 0, if $\boldsymbol{r}$ lies outside of Σ, to $1/2$ if $\boldsymbol{r}$ lies on Σ, and to 1 if $\boldsymbol{r}$ lies inside Σ, ∂' denotes derivative with respect to $\boldsymbol{r}'$. The fields $\boldsymbol{u}^\infty(\boldsymbol{r}')$ and $p^\infty(\boldsymbol{r}')$ are an arbitrary velocity-pressure pair satisfying the Stokes equation (2.1) [32, 33]. Another method is to use separate identities for regularization of the normal and tangential components of the Stokeslet kernel G and an identity for regularization of the Stresslet kernel T, as will be detailed below.

Despite these efforts, two problems in BI technique remain not fully resolved: Even after regularization, the BI kernels can remain singular, although the singularity is manifested in their higher derivatives. This singularity limits the convergence rate of non-singular Gaussian quadrature rules. The other problem is related to the near

singular integration, when the target point does not belong to the source surface but is located near it. In this study we will provide a way to reduce the effect of each of these problems, thus improving the precision of BI calculation, without excessive additional computational cost of the algorithm.

3. Problem formulation

Consider a 2D surface Σ representing the membrane of a RBC or a wall surrounding the vessel. The purpose of this study is to propose a method to calculate the integrals of forms

$$\int_\Sigma G_{ij}(\boldsymbol{r},\boldsymbol{r'})f_j(\boldsymbol{r'})d^2r', \tag{3.1}$$

$$\int_\Sigma T_{ijk}(\boldsymbol{r},\boldsymbol{r'})u_j(\boldsymbol{r'})n_k(\boldsymbol{r'})d^2r', \tag{3.2}$$

for a certain set M_t of target points $\boldsymbol{r}$. A known subset of target points M_t lies on the surface Σ, while the other target points do not. In this study we present the integration technique only, i.e., we assume that the positions of material points $\boldsymbol{r}'$, the area elements d^2r', the force distributions $\boldsymbol{f}(\boldsymbol{r}')$, the velocities $\boldsymbol{u}(\boldsymbol{r}')$, and the normals $\boldsymbol{n}(\boldsymbol{r}')$ at any point of the surface Σ can be calculated with sufficient precision and in reasonable time by one or another method.

We assume that the BIs (3.1) and (3.2) have to be calculated for a rather large number of target points: Indeed, even if an accelerated summation technique is used to treat long-range hydrodynamic interactions between RBCs, a significant number of short-range interactions must be taken in a pair-wise fashion.

The proposed algorithm is based on 4 techniques: subtraction of exact identities to regularize the singular kernels, use of additional quadrature nodes to increase the precision of near-singular integration, partitions of unity with cut-off distance to perform smooth splitting of the BI calculation between the main and the additional sets of quadrature nodes, and look-up algorithm to select the quadrature nodes that lie within the cut-off distance from a given target point.

4. Singularity subtraction

A drawback of the BI method is that the kernels in (2.2) are singular when $\boldsymbol{r} \to \boldsymbol{r}'$. This can have an effect on the numerical quality of the solutions. A natural question is whether or not the integral can be regularized in order to increase the numerical precision. It is known that this singularity could be subtracted analytically [34] when the force is normal. This is the case, for example, for fluid-fluid interfaces when the only force is surface tension $2\gamma H\boldsymbol{n}$, where H is the mean curvature (equal to -1 for a

sphere of radius 1) and γ is the surface energy. In this case an *exact* transformation of the BI allows one to subtract the singularity:

$$2\gamma \int_\Sigma G_{ij}(\boldsymbol{r},\boldsymbol{r}')H(\boldsymbol{r}')n_j(\boldsymbol{r}')d^2r' = 2\gamma \int_\Sigma G_{ij}(\boldsymbol{r},\boldsymbol{r}')\left[H(\boldsymbol{r}')-H(\boldsymbol{r})\right]n_j(\boldsymbol{r}')d^2r' + 2\gamma H(\boldsymbol{r}) \int_\Sigma G_{ij}(\boldsymbol{r},\boldsymbol{r}')n_j(\boldsymbol{r}')d^2r', \tag{4.1}$$

where $H(\boldsymbol{r})$ is independent of $\boldsymbol{r}'$. We assume that curvature H varies smoothly along the surface of a droplet, $H(\boldsymbol{r}')-H(\boldsymbol{r}) \propto |\boldsymbol{r}'-\boldsymbol{r}|$ (when $|\boldsymbol{r}'-\boldsymbol{r}|$ is small) and the integrand of the first integral on the right hand side of (4.1) remains bounded for $\boldsymbol{r}' \neq \boldsymbol{r}$. The integral can thus be calculated with the error of order $\mathcal{O}(h^2)$ (where h is the mesh size) by a simple Gaussian quadrature rule. The second integral on the right hand side of (4.1) vanishes exactly owing to the incompressibility condition of the fluids:

$$\int_\Sigma G_{ij}(\boldsymbol{r},\boldsymbol{r}')n_j(\boldsymbol{r}')d^2r' = \int \partial'_j G_{ij}(\boldsymbol{r},\boldsymbol{r}')d^3r' = 0. \tag{4.2}$$

The last integral in (4.2) is performed over the volume enclosed by surface Σ and ∂'_j designates the derivative with respect to r'_j. A remark is in order: actually the integrand of (4.1) is still not fully regular at $\boldsymbol{r} = \boldsymbol{r}'$: In fact, its gradient is divergent. Nevertheless, it can be shown [35] that the numerical error of integration of (4.1) by a simple quadrature rule is $\mathcal{O}(h^2)$.

Vesicles, capsules and RBCs imply not only a normal force but also a tangential one. The tangential force for vesicles arises from membrane incompressibility [4]. For capsules and RBCs, besides membrane incompressibility (which is not often adopted for capsules), the cytoskeleton is modeled by tangential shear elastic forces. It follows thus that the above trick used for droplets can not be used here. It was classically known in literature [16] that regularization is not possible if a tangential component is present in the membrane force. This challenge has been defied only recently [35], as summarized below. The singularity subtraction can still be performed by first noting that the tangential projection operator can be written as a double cross-product with the normal: $(I - \boldsymbol{n}\otimes\boldsymbol{n})\cdot\boldsymbol{f} = -[\boldsymbol{n}\times[\boldsymbol{n}\times\boldsymbol{f}]]$ and to use the identity

$$\int_\Sigma G_{ij}(\boldsymbol{r},\boldsymbol{r}')e_{jkl}n_k(\boldsymbol{r}')d^2r' = -\frac{e_{ijl}}{4\pi}\int_\Sigma \frac{(r_j-r'_j)(r_k-r'_k)n_k(\boldsymbol{r}')}{|\boldsymbol{r}-\boldsymbol{r}'|^3}d^2r'. \tag{4.3}$$

The identity (4.3) can be easily checked by converting the surface integral into a volume integral. This operation reveals the same integrands for both sides of (4.3). Note that a tiny vicinity of the pole yields a vanishing contribution to the surface and to the volume integrals thanks to the weakly singular behavior of the kernel G. Identity (4.3) is less obvious than (4.2): instead of having an integral on the right hand side that evaluates to zero (as in the case of droplet, owing to fluid incompressibility), here the

idea is to convert the integral on the left hand side of (4.3) into a converging one (right hand side). Actually the integral on the right hand side of (4.3) still has an apparent singularity. However, a close inspection shows that the integrand of the right hand side of (4.3) remains bounded because infinitesimal displacements along the surface are perpendicular to the normal: $(\boldsymbol{r}-\boldsymbol{r'})\cdot\boldsymbol{n}(\boldsymbol{r'})=\mathcal{O}(|\boldsymbol{r}-\boldsymbol{r}'|^2)$.

We are now in a position to show how to reduce the order of singularity at the pole of the kernel G for distribution of surface forces with arbitrary normal and tangential components by taking advantage of the two exact identities (4.2) and (4.3). For that purpose we replace the original force by a modified one:

$$\int\limits_{\Sigma} G_{ij}(\boldsymbol{r},\boldsymbol{r}')f_j(\boldsymbol{r}')d^2r' = \int\limits_{\Sigma} G_{ij}(\boldsymbol{r},\boldsymbol{r}')\tilde{f}_j(\boldsymbol{r},\boldsymbol{r}')d^2r' + \frac{[\boldsymbol{n}(\boldsymbol{r})\times\boldsymbol{f}(\boldsymbol{r})]_l e_{ijl}}{4\pi}\int\limits_{\Sigma}\frac{(r_j-r'_j)(r_k-r'_k)n_k(\boldsymbol{r'})}{|\boldsymbol{r}-\boldsymbol{r}'|^3}d^2r', \quad (4.4)$$

where the modified force $\tilde{\boldsymbol{f}}$ is written as

$$\tilde{\boldsymbol{f}}(\boldsymbol{r},\boldsymbol{r'}) = \boldsymbol{f}(\boldsymbol{r'}) - \boldsymbol{n}(\boldsymbol{r'})\left(\boldsymbol{f}(\boldsymbol{r})\cdot\boldsymbol{n}(\boldsymbol{r})\right) + [\boldsymbol{n}(\boldsymbol{r'})\times[\boldsymbol{n}(\boldsymbol{r})\times\boldsymbol{f}(\boldsymbol{r})]]. \quad (4.5)$$

It is easy to see that (i) $\tilde{\boldsymbol{f}}(\boldsymbol{r},\boldsymbol{r}')=\mathcal{O}(|\boldsymbol{r}-\boldsymbol{r}'|)$ for fixed $\boldsymbol{r}$ as $\boldsymbol{r}'$ approaches $\boldsymbol{r}$ and (ii) $(\boldsymbol{r}-\boldsymbol{r'})\cdot\boldsymbol{n}(\boldsymbol{r'})=\mathcal{O}(|\boldsymbol{r}-\boldsymbol{r}'|^2)$ as $\boldsymbol{r}'$ approaches $\boldsymbol{r}$ along the surface Σ. Thus, all the integrands on the right hand side of (4.4) are bounded and continuous as a function of $\boldsymbol{r}'$ for any $\boldsymbol{r}'\neq\boldsymbol{r}$ (provided $\boldsymbol{f}$ is continuous) and therefore can be integrated with decent precision. This completes the basic idea of SS (Singularity Subtraction) technique for arbitrary distribution of forces.

The double-layer kernel is regularized thanks to an identity

$$\int\limits_{\Sigma} T_{ijk}(\boldsymbol{r},\boldsymbol{r'})n_k(\boldsymbol{r'})d^2r' = \frac{1}{2}\delta_{ij} \quad (4.6)$$

when $\boldsymbol{r}_0\in\Sigma$. This allows us to regularize the double-layer kernel

$$\int\limits_{\Sigma} T_{ijk}(\boldsymbol{r},\boldsymbol{r'})u_j(\boldsymbol{r'})n_k(\boldsymbol{r'})d^2r' = \frac{1}{2}u_i(\boldsymbol{r}) + \int\limits_{\Sigma} T_{ijk}(\boldsymbol{r},\boldsymbol{r'})[u_j(\boldsymbol{r'})-u_j(\boldsymbol{r})]n_k(\boldsymbol{r'})d^2r' \quad (4.7)$$

as discussed, for example, in [28].

5. Near-singular integration

As already discussed, the SS does not fully regularize the singular Green kernel: While the modified kernel is bounded and continuous everywhere but at the pole, its gradient diverges in vicinity of the pole. Therefore, the numerical error of integration of (4.4) by a non-adaptive quadrature is of order $\mathcal{O}(h^2)$, as noted in [35]. In general,

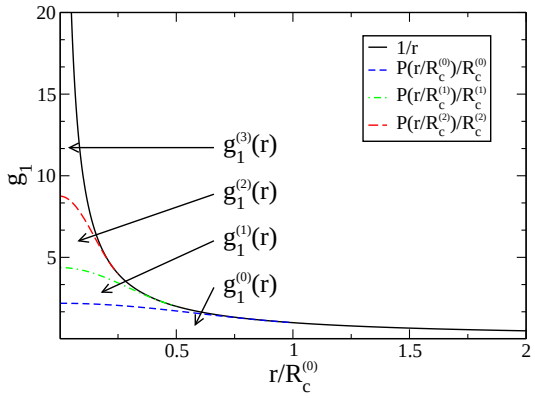

Figure 2: Representing $g_1(r) = 1/r$ as a sum of several smooth contributions and a singular contribution with very small support. The growth rate of each contribution is compatible with the refinement of the mesh on which it is calculated.

it is possible to regularize the singular kernels to any degree of smoothness by more complicated exact identities, which would improve the order of convergence of the numerical integration of (3.1) and (3.2). An alternative approach is to use additional quadrature nodes in vicinity of the pole of Green kernels. The main motivation here is that the number of target points $N_t = |M_t|$ is large, so that if we reuse some data calculated for one target point to calculate the BIs for other target points, the computational complexity of the algorithm will not increase significantly even if a rather large number of additional quadrature nodes is used in vicinity of the pole. The second motivation is that the singularity subtraction does not improve the precision of BI calculation when the source and the target points belong to different surfaces, or to the same surface but the distance between them is much smaller if calculated in the 3D space than if calculated along the membrane of the RBC (like between two concavities of the equilibrium shape of the RBC). A properly set up refinement technique would greatly increase the precision of numerical integration of singular kernels in these cases.

The basic strategy of reusing the quadrature nodes has been presented in [35] for a second-order BI method. Here we propose an adaptation of this technique to higher-order calculations. In the following presentation, we assume that the surface is represented by a mesh of high-order triangular elements, although this technique can be also used for the other methods of high-order representation of surfaces. We call $M_s^{(0)}(\Sigma)$ the quadrature nodes used on the main mesh of surface Σ. The additional quadrature nodes are obtained by constructing a series of N_r refined meshes: each refined mesh is obtained from a previous one (the first from the original mesh) by a refinement procedure, subdividing each triangle of the mesh into 4 smaller triangles by putting new vertices in the middles of the edges (as shown in Fig. 3-left). The set of quadrature nodes on a mesh obtained by p refinements from the main mesh will be called $M_s^{(p)}(\Sigma)$.

Figure 3: Left: Refinement scheme: each triangle (thick black points and lines), is subdivided into 4 smaller triangles (thin red points and lines). Right: Quadrature rules on each triangle. The quadrature weights in units of the area of the triangle are shown for each node.

The singular multipliers of the Green kernels are then partitioned in a sum of N_r+1 contributions,

$$\frac{1}{r} = \sum_{p=0}^{N_r} g_1^{(p)}(r), \tag{5.1}$$

$$\frac{1}{r^3} = \sum_{p=0}^{N_r} g_3^{(p)}(r), \tag{5.2}$$

$$\frac{1}{r^5} = \sum_{p=0}^{N_r} g_5^{(p)}(r), \tag{5.3}$$

where $g_q^{(p)}(r)$ for $p < N_r$ are sufficiently smooth to allow integration by Gaussian quadrature. In addition, $g_q^{(p)}(r)$ are equal to zero for $r > R_c^{(p)}$ and $N_r \geq p > 0$, where $R_c^{(p)} = 2^{1-p}R_c$ is the cut-off distance for calculation on the p-th refined mesh and R_c is the cut-off parameter that should be chosen as a compromise between the precision of near-singular integration and the computational cost of integration on the refined meshes. The calculation of the BI for a given point $\boldsymbol{r}_0$ then goes as follows: All singular multipliers in (4.4) are replaced by (5.1)-(5.3) and the integration of the terms containing $g_q^{(p)}$ is performed on the main mesh for $p = 0$ and on the p-th refined mesh for $p > 0$. Naming $N_s = |M_s^{(0)}(\Sigma)|$ the number of quadrature nodes in the main mesh, we observe that the number of quadrature nodes in the p-th refined mesh is roughly $4^p N_s$. However, since only those vertices of the refined mesh $M_s^{(p)}(\Sigma)$ that lie within the distance $R_c^{(p)} = 2^{1-p}R_c$ from a given target point $\boldsymbol{r}$ contribute to the numerical estimate of the BI at $\boldsymbol{r}$, we conclude that the integration time of the contributions proportional to $g_q^{(p)}$ using the quadrature nodes $M_s^{(p)}(\Sigma)$ is roughly independent of p for $p > 0$, provided the source points that lie within the cut-off distance from the target point $\boldsymbol{r}$ are chosen in advance. In this case, proper time that is spent on BI calculation is proportional to N_r, and the main limiting factor on the number of the refined meshes is the time required to precalculate the coordinates, forces, normals, and area weights in the quadrature nodes $M_s^{(N_r)}(\Sigma)$.

6. Look-up algorithm

The selection of quadrature nodes on the refined meshes that lie within the cut-off distance from a given target point is performed using a multi-level look-up algorithm. The basic version of the look-up algorithm was presented in detail in [35] for a single refined mesh. Here we provide a multi-level look-up algorithm that can be employed for any N_r. For each refined mesh $M_s^{(p)}$, we distribute the vertices between disjoint look-up sets $L_\alpha^{(p)}$. The sets are indexed by vertices α from the coarser mesh $M_s^{(p-1)}$. We also introduce a combined look-up set, $\tilde{L}_\alpha^{(p)}$, such that $\tilde{L}_\alpha^{(N_r)} = L_\alpha^{(N_r)}$ and

$$\tilde{L}_\alpha^{(p)} = \bigcup_{\beta \in L_\alpha^{(p)}} \tilde{L}_\beta^{(p+1)} \tag{6.1}$$

for $p < N_r$.

For each p, the points are distributed between the look-up sets in such a way that gives the minimal value for the look-up radius $L_{\max}^{(p)}$. The look-up radius is defined as the maximum distance between the index vertex of a combined look-up set and a member vertex of the combined look-up set for all vertices and all combined look-up sets

$$L_{\max}^{(p)} = \max_{\alpha \in M_s^{(p-1)}} \max_{\beta \in \tilde{L}_\alpha^{(p)}} |\boldsymbol{r}^\alpha - \boldsymbol{r}^\beta|. \tag{6.2}$$

In general, finding the correct look-up set for a given vertex requires finding the nearest vertex from the coarser mesh. In practice, for a vertex α, we find the nearest vertex belonging to the parent triangles of the ones to which the vertex α belongs. The latter task is much simpler and the increase of the look-up radii introduced by this simplification is hardly noticeable. This construction defines a look-up tree for each vertex of the main mesh. The advantage of the look-up algorithm is that it will allow us to avoid scanning some parts of these trees when searching for the vertices lying within the cut-off distance from a given target point.

We simplify notations and introduce the critical distances $d_c^{(p)}$, defined with the following recurrence relation: $d_c^{(N_r)} = R_c^{(N_r)} + L_{\max}^{(N_r)}$ and $d_c^{(p)} = \max(R_c^{(p)}, d_c^{(p+1)}) + L_{\max}^{(p)}$ for $p < N_r$. Under normal conditions, $d_c^{(p)} = R_c^{(p)} + L_{\max}^{(p)}$ because $L_{\max}^{(p)}$ is the size of the mesh triangles on the mesh $M_s^{(p)}$, while the precision limitations to the integration of the splitting function $g_q^{(p)}$ dictate that $r_c^{(p)}$ be at least several times greater than the size of the mesh triangles on the mesh $M_s^{(p)}$. For a given p, we will call a given source vertex $\alpha \in M_s^{(p)}(\Sigma)$ well separated from a given target point $\boldsymbol{r}_0$ if $|\boldsymbol{r}_0 - \boldsymbol{r}^\alpha| > d_c^{(p)}$.

The idea behind the look-up algorithm is that if for given $p < N_r$ a target point $\boldsymbol{r}_0$ and a vertex $\alpha \in M_s^{(p)}(\Sigma)$ are well separated, then there is no contribution to the BI estimate at $\boldsymbol{r}_0$ from all vertices in the combined look-up set $\tilde{L}_\alpha^{(p+1)}$ considering only contributions from the multipliers $g_q^{(p')}(r)$ with $p' > p$. Indeed, considering a vertex $\beta \in L_\alpha^{(p+1)}$, we can write

$$|\boldsymbol{r}_0 - \boldsymbol{r}^\beta| > |\boldsymbol{r}_0 - \boldsymbol{r}^\alpha| - |\boldsymbol{r}^\alpha - \boldsymbol{r}^\beta| > d_c^{(0)} - L_{\max}^{(0)} = \max(R_c^{(0)}, d_c^{(1)}). \tag{6.3}$$

Inequality (6.3) means that (i) vertex β does not contribute to the BI part calculated on the mesh $M_s^{(1)}(\Sigma)$ and (ii) vertex $\beta \in M_s^{(p+1)}(\Sigma)$ is well separated from $\boldsymbol{r}_0$.

The algorithm itself goes as follows: For a given target point $\boldsymbol{r}_0$, we construct the look-up lists $S^{(p)}(\boldsymbol{r}_0)$ for all $0 < p \leq N_r$. The look-up list $S^{(p)}(\boldsymbol{r}_0)$ contains all vertices belonging to the look-up sets $L_\alpha^{(p)}$, such that α is not well separated from $\boldsymbol{r}_0$. The advantage here is that we need to check only the vertices in $S^{(p)}(\boldsymbol{r}_0)$ when calculating the contribution of the mesh $M_s^{(p)}(\Sigma)$ to the BI estimate at the point $\boldsymbol{r}_0$. From the programming point of view, it is convenient to endow the look-up sets with a list structure. This way the look-up lists can be simply assembled by concatenation.

First, we scan all vertices on the main mesh in order to find all vertices that are not well separated from $\boldsymbol{r}_0$ and to assemble the look-up list $S^{(1)}(\boldsymbol{r}_0)$. Now if we already have the look-up list $S^{(p)}(\boldsymbol{r}_0)$, we inspect all vertices in it and select those that are not well separated from $\boldsymbol{r}_0$. The look-up list $S^{(p+1)}$ is then constructed by concatenation of the look-up sets indexed by the selected points. We found it convenient to combine the assembly of the look-up lists with the actual computation of the contribution of the refined meshes to the BI estimate. This way we could also reuse the results of the intermediate calculations (such as the distance between the target point and the source point).

7. Numerical tests

Here we provide a numerical test of the presented technique. For simplicity, we choose a sphere with center $(0,0,0)$ and radius 1 as the surface Σ. A sample distribution of forces and velocities

$$\boldsymbol{F}(\boldsymbol{r}) = \boldsymbol{V}(\boldsymbol{r}) = (r_x^4 - 6r_x^2r_y^2 + 6r_y^2r_z^2 - r_z^4, -4r_x^3r_y + 12r_xr_yr_z^2, 12r_xr_y^2r_z - 4r_xr_z^3) \quad (7.1)$$

is used. The BI for the distribution (7.1) is calculated analytically as

$$\int G(A\boldsymbol{r}_0, \boldsymbol{r}) \cdot \boldsymbol{F}(\boldsymbol{r}) d^2r = \left[\frac{2}{33A^5} - \frac{5}{22}\left(1 - \frac{1}{A^2}\right)\right] F(\boldsymbol{r}_0) + \frac{1}{2A^5}\left(1 - \frac{1}{A^2}\right) \boldsymbol{r}_0(\boldsymbol{F}(\boldsymbol{r}_0) \cdot \boldsymbol{r}_0) \quad (7.2)$$

for the single layer kernel. Here $A \geq 1$ and $r_0 = 1$ and we have simplified the result by reexpressing it through the definition (7.1) (in fact, the distribution (7.1) is one of the vector spherical harmonics, which are the eigenvectors of the Green operators (2.3) on a sphere). The integral of the double-layer kernel is calculated to read

$$\int [K(\boldsymbol{r}_0, \boldsymbol{r}) \cdot \boldsymbol{V}(\boldsymbol{r})] \cdot \boldsymbol{n}(\boldsymbol{r}) d^2r = -\frac{1}{66} F(\boldsymbol{r}_0). \quad (7.3)$$

Here we only provide the result when $\boldsymbol{r}_0$ lies on the source sphere ($r_0 = 1$).

The main mesh is obtained from a regular icosahedron by several refinement procedures described above (and shown in Fig. 3-left). The new vertices of the main mesh

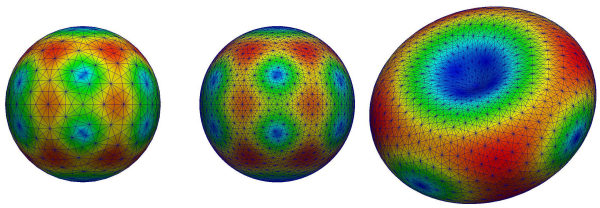

Figure 4: Sample meshes used to test the BI technique. From left to right: (i) sphere, 2 refinements ($N_s = 962$), (ii) sphere, 3 refinements ($N_s = 3842$), (iii) biconcave shape, 3 refinements ($N_s = 3842$). Color by the magnitude of the sample force in all cases [Eq. (7.1) for the first two meshes and Eq. (8.1) for the third one].

generated on each refinement step are projected on the unity sphere. The quadrature rules on the sphere are obtained from quadrature rules on mesh triangles by the projection from the center of the sphere

$$\boldsymbol{r} \to \boldsymbol{r}' = \boldsymbol{r}/r. \tag{7.4}$$

The area dilatation during this projection is $d^2r'/d^2r = \boldsymbol{r} \cdot \boldsymbol{n}_\Delta / r^3$, where $\boldsymbol{n}_\Delta$ is the outward normal to the triangle Δ. It must be noted here that the projection (7.4) is a continuous bijection, infinitely smooth inside each mesh triangle, which allows us to obtain high-order quadrature rules on a sphere from the classical high-order quadrature rules on triangles. In this study we are going to use 4th-order 7-point quadrature rule on each mesh triangle, based on the vertices (weighted as $A_\Delta/20$ each), edge midpoints (weighted as $2A_\Delta/15$ each) and the center of mass (weighted as $9A_\Delta/20$), where A_Δ is the area of a given mesh triangle (as shown in Fig. 3-right). When generating the refined meshes, all refinements are performed before the projection on the sphere.

Consistently with the fourth order of the quadrature rule, we choose the partition functions $g_q^{(p)}$ to be 3 times continuously differentiable. The exact expressions are given using polynomials

$$P_q(r) = \sum_{l=0}^{l_{\max}} a_l^{(q)} r^{2l} \tag{7.5}$$

such that

$$\left.\frac{d^l P_q(r)}{dr^l}\right|_{r=1} = \left.\frac{d^l}{dr^l}\left(\frac{1}{r^q}\right)\right|_{r=1} \tag{7.6}$$

for all $l \le l_{\max}$ (as shown in Fig. 2). In our case, $l_{\max} = 3$ and

$$P_1(r) = \frac{35}{16} - \frac{35}{16}r^2 + \frac{21}{16}r^4 - \frac{5}{16}r^6, \tag{7.7}$$

$$P_3(r) = \frac{105}{16} - \frac{189}{16}r^2 + \frac{135}{16}r^4 - \frac{35}{16}r^6, \tag{7.8}$$

$$P_5(r) = \frac{231}{16} - \frac{495}{16}r^2 + \frac{385}{16}r^4 - \frac{105}{16}r^6. \tag{7.9}$$

As seen from (7.5) or Fig. 2, a function equal to $P_q(r)$ for $r < 1$ and equal to r^{-q} otherwise is $l_{\max}$ times continuously differentiable.

Using these notations, we write the functions $g_q^{(p)}(r)$ as

$$g_q^{(0)}(r) = \begin{cases} r^{-q} & \text{if } r \geq R_c^{(1)}, \\ \left(R_c^{(1)}\right)^{-q} P_q(r/R_c^{(1)}) & \text{if } r < R_c^{(1)}, \end{cases} \tag{7.10}$$

$$g_q^{(N_r)}(r) = \begin{cases} 0 & \text{if } r \geq R_c^{(N_r)}, \\ r^{-q} - (R_c^{(N_r)})^{-q} P_q(r/R_c^{(N_r)}) & \text{if } r < R_c^{(N_r)}, \end{cases} \tag{7.11}$$

$$g_q^{(p)}(r) = \begin{cases} 0 & \text{if } r \geq R_c^{(p)}, \\ r^{-q} - \left(R_c^{(p-1)}\right)^{-q} P_q(r/R_c^{(p-1)}) & \text{if } R_c^{(p)} > r \geq R_c^{(p+1)}, \\ \left(R_c^{(p+1)}\right)^{-q} P_q(r/R_c^{(p+1)}) - \left(R_c^{(p)}\right)^{-q} P_q(r/R_c^{(p)}) & \text{if } r < R_c^{(p+1)}. \end{cases} \tag{7.12}$$

8. Results

We first consider the case when $A = 1$, which means that the target point of BI calculation lies on the sphere where the sample forces are distributed. Fig. 5 shows the convergence of the method depending on the number of the points in the main mesh $N_s^{(0)}$ and on the number of refined meshes used for near-singular integration. For a fixed number of refined meshes, the second-order convergence is observed. However, increasing the number of refined meshes by 1 divides the numerical error by 4, provided enough points are used in the original mesh. The explanation is that there are 3 sources of numerical error: The far-field part of the BI is calculated by a fourth-order quadrature rule, whose numerical error is decreased by a factor of 16 upon each refinement of the main mesh. For the near-singular and singular parts of the BI, the numerical error is decreased by a factor of 4 upon refinement of the main mesh or upon increasing the number of refined meshes by 1. The numerical error due to splitting of the BI calculation between meshes decreases by a factor of 16 upon refinement of the main mesh or upon doubling the cut-off parameter. This part gives a noticeable contribution to the overall numerical error only if the cut-off parameter is very small.

In this test, we used the cut-off parameter equal to 0.5 (of the radius of the sample sphere). Figs. 6 and 7 show how the numerical error behaves if the cut-off parameter is increased or reduced. As can be seen, for $r_c = 0.5$, the numerical error due to splitting is negligible for all reasonable values of the number of points in the main mesh. Smaller values of r_c can be used for very refined main meshes, although it does not reduce significantly the computation times.

Next we test how the introduction of refined meshes allows us to increase the precision when calculating the hydrodynamic interactions between two surfaces. We put the target points at the distance $A - 1$ from the sphere on which the source forces

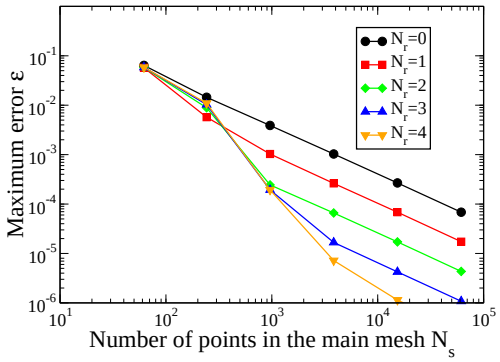

Figure 5: Numerical integration for the single-layer kernel. Maximum numerical error ε as a function of the number of points in the main mesh N_s for several numbers of refined meshes N_r. The cut-off parameter $R_c = 0.5$.

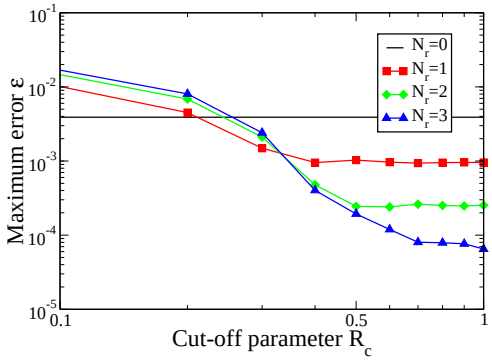

Figure 6: Numerical integration for the single-layer kernel. Maximum numerical error ε as a function of the cut-off parameter R_c for several numbers of refined meshes. Number of points in the main mesh $N_s = 962$. The result for $N_r = 0$ does not depend on R_c and is given as a horizontal line.

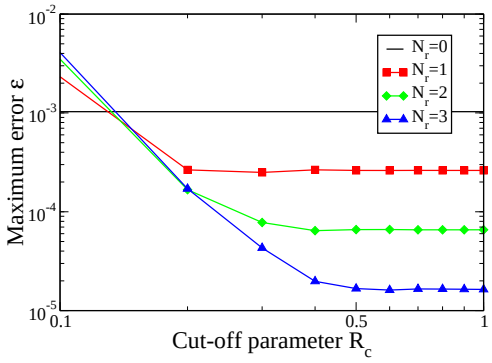

Figure 7: Numerical integration for the single-layer kernel. Maximum numerical error ε as a function of the cut-off parameter R_c for several numbers of refined meshes. Number of points in the main mesh $N_s = 3842$. The result for $N_r = 0$ does not depend on R_c and is given as a horizontal line.

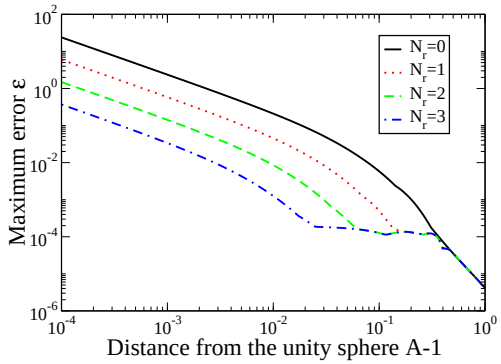

Figure 8: Numerical integration for the single-layer kernel. Maximum numerical error ε as a function of the distance of the target from the source sphere $A-1$ for several numbers of refined meshes. Cut-off parameter $R_c = 0.5$, number of points in the main mesh $N_s = 962$.

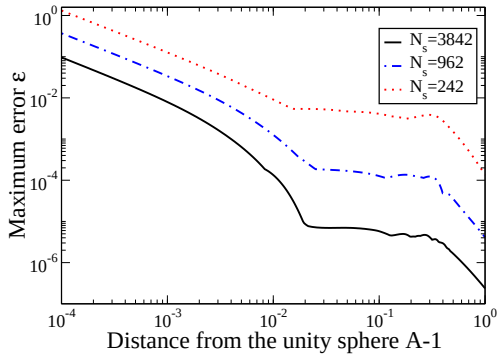

Figure 9: Numerical integration for the single-layer kernel. Maximum numerical error ε as a function of the distance of the target from the source sphere $A-1$ for several numbers of points on the original mesh. Cut-off parameter $R_c = 0.5$, 3 refined meshes ($N_r = 3$).

are distributed and investigate the discrepancy between the numerical results and the theoretical predictions as A approaches 1. Figs. 8 and 9 present the results. We can see, that adding one more refined mesh allows to increase the distance at which the BI calculation starts to diverge by a factor of 2 or 3.

A quick overview of the convergence of the numerical integration for the double-layer kernel is presented in Fig. 10.

We have performed a test for a non-spherical surface. In this geometry, no simple analytical results similar to (7.2) and (7.3) exist. Nevertheless, there is an analytical expression which we can use. Namely, we can calculate numerically Eq. (2.4) for a chosen sample flow. Substituting a linear flow into this equation gives an identity that is used in an alternative way to perform the singularity subtraction [32, 33]. Here we are going to use the identities (4.4) and (4.7) in order to regularize the singular integrals in (2.4). We have chosen the flow

$$u^{\infty}(\boldsymbol{r}) = (r_y r_z, r_z r_x, r_x r_y), \tag{8.1}$$

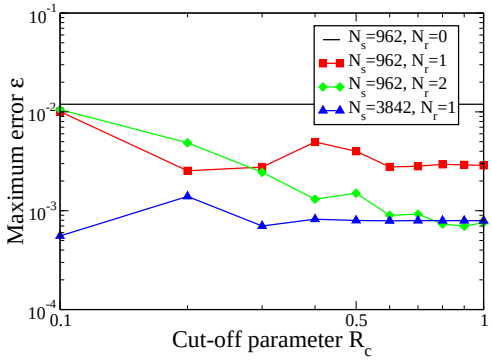

Figure 10: Numerical integration for the double-layer kernel. Maximum numerical error ε as a function of the distance of the cut-off parameter R_c for chosen combinations of number of points on the main mesh N_s and number of refined meshes N_r. The result for $N_r = 0$ does not depend on R_c and is given as a horizontal line.

which satisfies the Stokes equation (2.1) with pressure equal to 0 everywhere. Using the stress tensor corresponding to the flow (8.1), we calculate the force distribution $\boldsymbol{f}^{\infty}(\boldsymbol{r})$

$$\boldsymbol{f}^{\infty}(\boldsymbol{r}) = 2(r_z n_y(\boldsymbol{r}) + r_y n_z(\boldsymbol{r}), r_x n_z(\boldsymbol{r}) + r_z n_x(\boldsymbol{r}), r_y n_x(\boldsymbol{r}) + r_x n_y(\boldsymbol{r})), \tag{8.2}$$

which we substitute into (2.2). The non-spherical surface is parametrized by a map form the unity sphere,

$$\boldsymbol{r}'(\boldsymbol{r}) = \boldsymbol{r}(1 - a r_z^2), \tag{8.3}$$

where a is a parameter, which we set to 0.75. We apply the map (8.3) to spherical meshes that were used above thus obtaining a mesh on the non-spherical surface. The resulting surface has a biconcave shape, reminiscent of the equilibrium shape of RBCs (rightmost shape in Fig. 4). A sample mesh for the shape (8.3) is shown in Fig. 4. The definition (8.3) allows us to calculate the normal at every point analytically and also gives us the dilatation d^2r'/d^2r, which should multiply the quadrature weights on the original spherical mesh.

The results of the calculation of (2.4) for flow (8.1) and shape (8.3) with $a = 0.75$ are given in Fig. 11. Here we plot the difference of left- and right hand sides of (2.4) calculated numerically for different numbers of N_s and N_r. As is evident from Fig. 11, the precision is similar to that observed for the spherical shape.

Finally, we provide some performance tests in order to demonstrate the advantage of using the refined meshes and the look-up algorithm. In all cases, we employ the singularity subtraction and take the vertices of the refined mesh as the target points. We then investigate the effect of using the refined meshes and the look-up algorithm on the performance and the precision of the algorithm. We will compare the results of 3 different approaches: (i) No refined meshes are used. This is the fastest method, although it is not very precise. The complexity scales as $\mathcal{O}(N_t^2)$ and the numerical error is of order $\mathcal{O}(1/N_t)$ (ii) the whole BI is calculated using the most refined mesh. We add

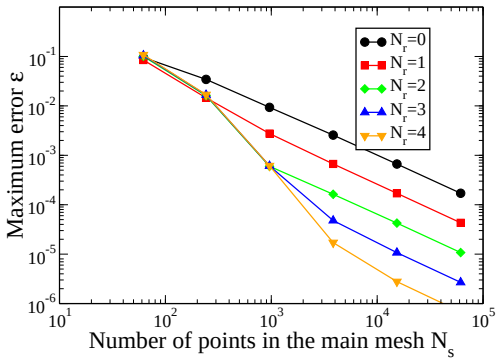

Figure 11: Numerical integration of Eq. (2.4) with the sample flow (8.1) on for the surface given by map (8.3) from the unity sphere for several numbers of N_s and N_r. $a = 0.75$. Cut-off distance is set to 0.5.

one refined mesh for each refinement of the main mesh ($N_r = 1$ for $N_s = N_t = 242$, $N_r = 2$ for $N_s = N_t = 962$, and so on). This method is precise but quite costly in terms of computational time. The theoretical complexity scales as $\mathcal{O}(N_t^3)$, while the precision is of order $\mathcal{O}(1/N_t^2)$. (iii) The method proposed in the present study. We use the same number of refined meshes as in the method (ii), but now the refined meshes are used only for the integration in the vicinity of the pole of the Green kernel and the look-up algorithm is used to select the points that lie within the cut-off distance from a given target point. It is the purpose of this study to demonstrate that the method (iii) provides the best precision for a given computational time. Our expectation is that with a proper choice of the cut-off distance R_c, the numerical error will scale as $\mathcal{O}(1/N_t^2)$ similarly to the method (ii). This estimate is supported by the convergence tests above. Regarding the computational cost, we expect that the total running times will scale as $\mathcal{O}(N_r N_t^2)$, which is equivalent to $\mathcal{O}(N_t^2 ln N_t)$ in our case.

The results of the comparison are given in Fig. 12. We plot the maximum numerical error ε as a function of the computational time T. The time T here is total time, including mesh generation, shape and force distribution, area weight calculation and the integration itself. The calculations were performed using a single core of a modern CPU. The guides for the eye are added in order to simplify the visual analysis of the results. First, we can see that the methods (i) and (ii) agree well with the expected asymptotic scalings. Second, we see that the method (iii) is almost as precise as the method (ii) while being much faster for all cases except for $N_t = 242$, in which case they are mostly equal. We conclude that the method (iii) is indeed the most advantageous one.

We have also isolated the times required for the BI calculation alone, i.e., the cost of mesh generation and of calculation of forces and area wights was excluded. The purpose of this test is to examine whether the theoretical prediction for the computational cost of the look-up algorithm as a function of the number of refined meshes can be reproduced in practice. The results are given in Fig. 13. The time spent on integration

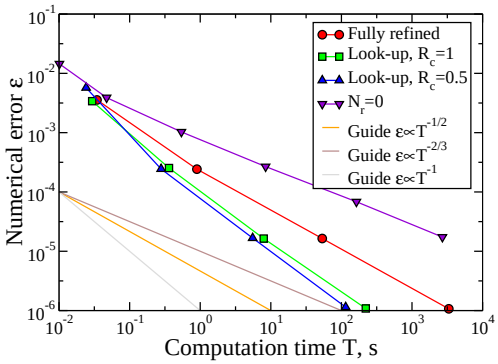

Figure 12: Numerical error as a function of the computational cost for several numbers of mesh vertices. The points correspond (in order of increase of computational times and decrease of the numerical error to $N_s = 242$, $N_r = 1$; $N_s = 962$, $N_r = 2$; $N_s = 3842$, $N_r = 3$; and $N_s = 15362$, $N_r = 4$. For the method without the refined mesh, $N_r = 0$ and two additional points are given, $N_s = 61442$ and $N_s = 245762$. Note that a noticeable scattering of computational cost is observed for $T < 0.1$.

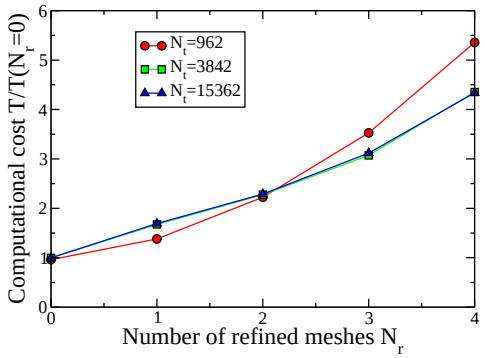

Figure 13: Relative computational cost of BI calculation (excluding the time for mesh generation and calculation of forces and area weights) as a function of the number of refined meshes. The running times with look-up algorithm T is normalized by the running time is now refined meshes are used $T(N_r = 0)$. The linear increase of the relative computational time with the number of the refined meshes is observed fairly well. The cut-off distance is chosen as $R_c = 0.5$.

is normalized by the running time for the method that does not use refined meshes ($N_r = 0$). As can be seen, the observed dependence is indeed close to a linear one.

9. Discussion

The proposed method of BI calculation can be used for fourth-order methods: Computational complexity of a non-adaptive BI method in real space is of order $\mathcal{O}(N^2)$, where N is the number of points used for surface discretization. Using singularity subtraction, such a method would have numerical error of order $\mathcal{O}(1/N)$. This error comes from the divergence of the gradients of the regularized Green kernels and can-

not be eliminated by using high-order quadrature rules. However, because the time spent on BI calculation grows quadratically with N, we can interpolate the surface and the forces in $\sim N^2$ points at a fixed fraction of the time spent on BI calculation. Using N^2 points for near-singular integration instead of N reduces the numerical error to $\mathcal{O}(1/N^2)$, while the computation complexity of the BI computation presented above is $\mathcal{O}(N^2)\log_4 N$.

Besides the good precision/complexity relation, another advantage of the proposed technique is its versatility: The refinement is used regardless whether the target point belongs to the source surface or not and can also handle efficiently the situations when a surface is close to self-contact. This comes at a price: The method requires much more programming effort than the simplest pairwise integration.

10. Conclusion

In this study, we have presented a numerical technique for calculation of singular BIs with good precision. Other challenges exist in numerical simulation of deformable objects, such as surface interpolation, calculations of curvature, or surface reparametrization. In this sense, this algorithm represents an important step towards efficient simulation of blood flow by BI method but does not solve the problem completely. Nevertheless, there are some problems to which the proposed algorithm can be applied without further development. Most notably, the boundaries surrounding the flow (such as walls of a vessel or of a microfluidic circuit) can be often considered rigid and have a more or less simple geometry. Another possible application is simulation of different beads-and-springs models, in which several rigid spheres are subject to a potential that depends only on their relative positions. The behavior of such systems under flow provides an insight into the dynamics of generic elastic objects under flow [36, 37].

The BI method is powerful and precise. However, its nonlocal character causes the numerical cost to scale as N^2 (N being the number of discretization points) and a resort to FMM (Fast Multipole Method) is necessary, and should be implemented in the future if one wants to handle large collection of vesicles and capsules.

Finally, the BI method is limited to Stokes flow only. If inertia effects are to be taken into account, then one has to resort to other methods, like level set [38–41] phase-field [42–45] or immersed boundary methods [21, 46] that know nowadays a high activity.

Acknowledgments We are grateful to CNES and ESA for a financial support.

References

[1] U. Seifert, Advances in Physics **46**, 13 (1997).
[2] D. Barthès-Biesel, C.R. Physique **10**, 764 (2009).

[3] J. Walter, A.-V. Salsac, D. Barthès-Biesel, and P. Le Tallec, International Journal for Numerical Methods in Engineering **83**, 829 (2010).
[4] T. Biben, A. Farutin, and C. Misbah, Phys. Rev. E **83**, 031921 (2011).
[5] G. Boedec, M. Leonetti, and M. Jaeger, Journal of Computational Physics **230**, 1020 (2011).
[6] H. Zhao, A. H. Isfahani, L. N. Olson, and J. B. Freund, Journal of Computational Physics **229**, 3726 (2010).
[7] S. K. Veerapaneni, A. Rahimian, G. Biros, and D. Zorin, Journal of Computational Physics **230**, 5610 (2011).
[8] H. Zhao and E. S. G. Shaqfeh, Journal of Fluid Mechanics **674**, 578 (2011).
[9] H. Zhao, A. P. Spann, and E. S. G. Shaqfeh, Physics of Fluids **23**, 121901 (pages 12) (2011).
[10] A. P. Spann, H. Zhao, and E. S. G. Shaqfeh, Physics of Fluids (1994-present) **26**, 031902 (2014).
[11] H. Noguchi and G. Gompper, Proceedings of the National Academy of Sciences of the United States of America **102**, 14159 (2005).
[12] I. V. Pivkin and G. E. Karniadakis, Phys. Rev. Lett. **101**, 118105 (2008).
[13] D. A. Fedosov, B. Caswell, and G. E. Karniadakis, Biophysical Journal **98**, 2215 (2010).
[14] M. Kraus, W. Wintz, U. Seifert, and R. Lipowsky, Phys. Rev. Lett. **77**, 3685 (1996).
[15] I. Cantat and C. Misbah, Phys. Rev. Lett. **83**, 235 (1999a).
[16] C. Pozrikidis, Journal of Computational Physics **169**, 250 (2001).
[17] W. R. Dodson and P. Dimitrakopoulos, Phys. Rev. Lett. **101**, 208102 (2008).
[18] P. Bagchi and R. M. Kalluri, Physical Review E **80**, 016307 (2009).
[19] J. Clausen and C. Aidun, Phys. Fluid **23**, 123302 (2010).
[20] W. R. Dodson and P. Dimitrakopoulos, Phys. Rev. E **85**, 021922 (2012).
[21] Y. Kim and M.-C. Lai, Journal of Computational Physics **229**, 4840 (2010).
[22] P. Vlahovska, T. Podgorski, and C. Misbah, C. R. Physique **10**, 775 (2009).
[23] P. Vlahovska, D. Barthès-Biesel, and C. Misbah, C.R. Physique **14**, 451 (2013).
[24] I. Cantat and C. Misbah, Phys. Rev. Lett. **83**, 880 (1999b).
[25] E. Lac, A. Morel, and D. Barthès-Biesel, J. Fluid Mech. **573**, 149 (2007).
[26] S. K. Veerapaneni, D. Gueyffier, D. Zorin, and G. Biros, J. Comp. Phys. **228**, 2334 (2009).
[27] G. K. Batchelor, Journal of Fluid Mechanics **41**, 545 (1970).
[28] C. Pozrikidis, *Boundary Integral and Singularity Methods for Linearized Viscous Flow* (Cambridge University Press, 1992).
[29] M. G. Duffy, SIAM Journal on Numerical Analysis **19**, 1260 (1982).
[30] M. Khayat and D. Wilton, Antennas and Propagation, IEEE Transactions on **53**, 3180 (2005).
[31] A. Klöckner, A. Barnett, L. Greengard, and M. O'Neil, Journal of Computational Physics **252**, 332 (2013).
[32] E. Klaseboer, Q. Sun, and D. Y. C. Chan, Journal of Fluid Mechanics **696**, 468 (2012).
[33] L. Heltai, M. Arroyo, and A. DeSimone, Computer Methods in Applied Mechanics and Engineering **268**, 514 (2014).
[34] M. Loewenberg and E. J. Hinch, Journal of Fluid Mechanics **321**, 395 (1996).
[35] A. Farutin, T. Biben, and C. Misbah, http://hal.archives-ouvertes.fr/hal-00841996 **0**, 000000 (2012).
[36] A. M. Slowicka, M. L. Ekiel-Jezewska, K. Sadlej, and E. Wajnryb, J. Chem. Phys. **136**, 044904 (2012).
[37] A. M. Slowicka, E. Wajnryb, and M. L. Ekiel-Jezewska, Eur. Phys. J. E **36**, 31 (2013).

[38] E. Maitre, C. Misbah, P. Peyla, and A. Raoult, Physica D **243**, 1146 (2012).
[39] A. Laadhari, P. Saramito, and C. Misbah, Phys. Fluid **24**, 031901 (2012).
[40] V. Doyeux, Y. Guyot, V. Chabannes, C. Prud'Homme, and M. Ismail, Journal of Computational and Applied Mathematics **246**, 251 (2013).
[41] D. Salac and M. Miksis, J. Comp. Phys. **230**, 8192 (2011).
[42] T. Biben and C. Misbah, Phys. Rev. E **67**, 031908 (2003).
[43] Q. Du, C. Liu, and X. Wang, J. Comp. Phys. **198**, 450 (2004).
[44] Q. Du, C. Liu, and X. Wang, J. Comp. Phys. **212**, 757 (2006).
[45] T. Biben, K. Kassner, and C. Misbah, Phys. Rev. E **72**, 041921 (2005).
[46] Y. Kim and M.-C. Lai, Phys. Rev. E **86**, 066321 (2012).