Fast, High-Order Accurate Integral Equation Methods and Application to PDE-Constrained Optimization

by

Hai Zhu

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Applied and Interdisciplinary Mathematics)
in The University of Michigan
2021

Doctoral Committee:

Associate Professor Eric Johnsen, Co-Chair
Associate Professor Shravan Veerapaneni, Co-Chair
Professor Silas Alben
Professor Robert Krasny

Hai Zhu

hszhu@umich.edu

ORCID iD: 0000-0002-8407-8643

To my family.

ACKNOWLEDGEMENTS

My life at University of Michigan as a graduate student over the course of the past six years has been truly invaluable. First and foremost, I would like to deeply thank my advisor, Prof. Shravan Veerapaneni, for his constant encouragement and guidance, not only on research, but also on career and life. Without his advice and support, this thesis would not be possible, and I would never be where I am today. I am especially grateful for his generosity with his time and wisdom, his persistence to aim for delivering high-quality work, and the freedom he allowed in the research projects that I truly enjoy. It was an honor for me to have him as my advisor. I am looking forward to work with him on the journey next to make a larger impact.

I would also like to thank Prof. Eric Johnsen, Prof. Silas Alben, and Prof. Robert Krasny for serving on my committee and providing valuable comments to my thesis. My gratitude to them for sharing their knowledge and availability on this thesis. In addition to my thesis committee, I want to extend my heartfelt thanks to Prof. Alex Barnett, Prof. Marc Bonnet, Dr. Bowei(Bobbie) Wu, Dr. Hanliang Guo, and Dr. Ruowen Liu. My first paper was done with Bobbie supervised Prof. Barnett and Prof. Veerapaneni. As a beginner, I was fortunate to be brought into a project that addresses the close evaluation issue in simulating Stokes flow, through which I sharpened my skills and laid a solid foundation in building integral equation methods-based solvers. Later in applying fluid solvers, I learned the importance of using adjoint-based sensitivity analysis to solve Stokes fluid optimization problems from Prof. Bonnet, Hanliang, and Ruowen. I am particularly appreciative to Hanliang, who

has been an excellent mentor and a great collaborator to me. It has been a very enriching experience to learn from him over the past few years.

I am so grateful to the support of the Mathematics department, and the selfless help I got from many incredible staff therein, especially Teresa, who has always greeted me with great patience and generous care. The same gratitude also goes to my wonderful office mates: Alexander Zaitzeff and Ryan Sandberg, to whom I won't forget their kindness and will always appreciate the priesthood blessings they gave me for healing and comforting; Tianchen(Eric) Zhao, who is both my office mate and a member of our research group, to all the musics we have listened together in the office.

I have been very fortunate to work in a friendly research community, with many excellent individuals including former and current members from our research group.

I would like to thank all my friends and family for their support. Especially, I thank my parents for their love and encouragement. Last but not least, I thank my fiancee Xuan for her unconditional support for these many years, especially during my most difficult time and the COVID-19 pandemic. We share all the joys and challenges together through this journey.

TABLE OF CONTENTS

CHAPTER

LIST OF TABLES

<u>**Table**</u>

LIST OF FIGURES

Figure

ABSTRACT

Over the last several decades, the development of fast, high-order accurate, and robust integral equation methods for computational physics has gained increasing attention. Using integral equation formulation as a global statement in contrast to a local partial differential equation (PDE) formulation offers several unique advantages. For homogeneous PDEs, the boundary integral equation (BIE) formulation allows accurate handling of complex and moving geometries, and it only requires a mesh on the boundary, which is much easier to generate as a result of the dimension reduction. With the acceleration of fast algorithms like the Fast Multipole Method (FMM), the computational complexity can be reduced to $\mathcal{O}(N)$, where N is the number of degrees of freedom on the boundary. Using standard potential theory decomposition, inhomogeneous PDEs can be solved by evaluating a volume potential over the inhomogeneous source domain, followed by a solution of the homogeneous part.

Despite the advantages of BIE methods in easy meshing, near-optimal efficiency, and well conditioning, the accurate evaluation of nearly singular integrals is a classical problem that needs to be addressed to enable simulations for practical applications. In the first half of this thesis, we develop a series of product integration schemes to solve this close evaluation problem. The use of differential forms provides a dimensional-agnostic way of integrating the nearly singular kernels against polynomial basis functions analytically. So the problem of singular integration gets reduced to a matter of source function approximation. In 2D, this procedure has been traditionally portrayed by building a connection to complex Cauchy integral, then supplemented by a complex monomial approximation. In 3D, the closed differential form requirement leads to the design of a new function approximation scheme based on harmonic polynomials and quaternion algebra. Under a similar framework, we develop a high-order accurate product integration scheme for evaluating singular and nearly singular volume integral equations (VIE) in complex domains using regular Cartesian grids discretization. A high-order accurate source term approximation scheme matching smooth volume integrals on irregular cut cells is developed, which requires no function extension.

BIE methods have been widely used for studying Stokes flows, incompressible flows at low Reynolds' number, in both biological systems and microfluidics. In the second half of this thesis, we employ the BIE methods to simulate and optimize Stokes fluid-structure interactions. In 2D, a hybrid computational method is presented for simulating cilia-generated fluid mixing as well as the cilia-particle hydrodynamics. The method is based on a BIE formulation for confining geometries and rigid particles, and the method of regularized Stokeslets for the cilia. In 3D, we use the time-independent envelop model for arbitrary axisymmetric microswimmers to minimize the power loss while maintaining a target swimming speed. This is a quadratic optimization problem in terms of the slip velocity due to the linearity of Stokes flow. Under specified reduced volume constraint, we find prolate spheroids to be the most efficient micro-swimmer among various families of shapes we considered. We then derive an adjoint-based formulation for computing power loss sensitivities in terms of a time-dependent slip profile by introducing an auxiliary time-periodic function, and find that the optimal swimmer displays one or multiple traveling waves, reminiscent of the typical metachronal waves observed in ciliated microswimmers.

CHAPTER I

Introduction

1.1 Motivation and Problem Statement

The goal of this thesis is to develop fast and accurate integral equation methods-based solvers for elliptic PDEs, including the Laplace, Stokes, and Poisson equation in complex geometry, with a focus on recent developments in singular and nearly-singular layer and volume potential evaluation. The applications we consider in this thesis include fluid mixing and several PDE-constrained fluid optimization problems for microswimmers. These studies have implications for understanding natural systems, as well as designing artificial ones for accomplishing various physical tasks.

In the following, we introduce the PDEs which we are interested in solving. We consider a domain Ω in $\mathbb{R}^d$, where $d = 2$ or 3, with a smooth boundary $\Gamma = \partial\Omega$.

The Laplace Dirichlet boundary value problem (BVP) in two and three dimensions is given by the following equations:

$$\Delta u(\boldsymbol{r}) = 0 \quad \text{in} \quad \Omega, \quad u(\boldsymbol{r}) = g(\boldsymbol{r}) \quad \text{on} \quad \Gamma. \tag{1.1}$$

This equation models processes that can be termed as potential problems. The solutions of Laplace equation are the harmonic functions, which arise naturally in many problems in physics, including electrostatics and fluid dynamics.

One generalization of the Laplace equation is the Poisson's equation, which describes the potential field in the presence an inhomogeneous source density distribution $f(\boldsymbol{r})$ in Ω. The potential field $u(\boldsymbol{r})$ is a solution to the Poisson BVP, if it satisfies

$$-\Delta u(\boldsymbol{r}) = f(\boldsymbol{r}) \quad \text{in} \quad \Omega, \quad u(\boldsymbol{r}) = g(\boldsymbol{r}) \quad \text{on} \quad \Gamma. \tag{1.2}$$

This is a critical task in many areas of computational physics and we do not have to emphasize its importance.

We also consider the following BVP for a steady-state Stokes flow in Ω

$$-\mu\Delta \boldsymbol{u}(\boldsymbol{r}) + \nabla p(\boldsymbol{r}) = 0 \quad \text{and} \quad \nabla \cdot \boldsymbol{u}(\boldsymbol{r}) = 0 \quad \text{in} \quad \Omega, \tag{1.3a}$$

$$\boldsymbol{u}(\boldsymbol{r}) = \boldsymbol{g}(\boldsymbol{r}) \quad \text{on} \quad \Gamma. \tag{1.3b}$$

Here vector field $\boldsymbol{u}$ is the fluid velocity, p is the pressure in the fluid, μ is the coefficient of viscosity of the fluid, and $\boldsymbol{g}$ indicates the specified velocity field on Γ. The first equation models the balance of momentum, when the inertial convective forces are negligible relative to viscous forces, and the second equation models the conservation of mass by enforcing the incompressibility on the fluid.

The solution to all these problems are subject to appropriate conditions if Ω contains ∞. Fast and accurate solvers for these equations serve as templates for solving more complex equations, such as the incompressible Navier-Stokes equations and nonlinear elliptic PDEs.

1.2 Integral Equation Formulation

In this section, we discuss the integral equation formulation for homogeneous and inhomogeneous elliptic PDEs. One of the distinguished feature of BIE methods comparing to finite difference and finite element methods is that the only unknowns in the formulation lie on the domain boundary, which results in a dimension reduction in terms of the linear system that needs to be solved.

The free-space Green's function for the Laplace equation is given by

$$G(\boldsymbol{r}) = \begin{cases} -\frac{1}{2\pi}\log|\boldsymbol{r}|, & \text{2D,} \\ \frac{1}{4\pi|\boldsymbol{r}|}, & \text{3D.} \end{cases} \tag{1.4}$$

where $\boldsymbol{r}$ is a vector in $\mathbb{R}^d$. And its partial derivative along some surface normal vector $\boldsymbol{n}$ is:

$$\frac{\partial}{\partial \boldsymbol{n}} G(\boldsymbol{r}) = \begin{cases} -\frac{1}{2\pi}\frac{\boldsymbol{r}\cdot\boldsymbol{n}}{|\boldsymbol{r}|^2}, & \text{2D,} \\ -\frac{1}{4\pi}\frac{\boldsymbol{r}\cdot\boldsymbol{n}}{|\boldsymbol{r}|^3}, & \text{3D.} \end{cases} \tag{1.5}$$

which represents a normally-oriented dipole. Using BIE formulation, the solution *ansatz* $u(\boldsymbol{r}')$ at some target point $\boldsymbol{r}'$ is typically given by Laplace single- (SLP) or double-layer (DLP) potentials, defined by

$$u(\boldsymbol{r}') = \mathcal{S}[\tau](\boldsymbol{r}') := \begin{cases} -\frac{1}{2\pi}\int_\Gamma \left(\log\frac{1}{|\boldsymbol{r}'-\boldsymbol{r}|}\right)\tau(\boldsymbol{r})\mathrm{d}s_{\boldsymbol{r}}, & \text{2D,} \\ -\frac{1}{4\pi}\int_\Gamma \frac{1}{|\boldsymbol{r}'-\boldsymbol{r}|}\tau(\boldsymbol{r})\mathrm{d}S_{\boldsymbol{r}}, & \text{3D.} \end{cases} \tag{1.6}$$

$$u(\boldsymbol{r}') = \mathcal{D}[\tau](\boldsymbol{r}') := \begin{cases} \frac{1}{2\pi}\int_\Gamma \frac{(\boldsymbol{r}'-\boldsymbol{r})\cdot\boldsymbol{n}_{\boldsymbol{r}}}{|\boldsymbol{r}'-\boldsymbol{r}|^2}\tau(\boldsymbol{r})\mathrm{d}s_{\boldsymbol{r}}, & \text{2D,} \\ \frac{1}{4\pi}\int_\Gamma \frac{(\boldsymbol{r}'-\boldsymbol{r})\cdot\boldsymbol{n}_{\boldsymbol{r}}}{|\boldsymbol{r}'-\boldsymbol{r}|^3}\tau(\boldsymbol{r})\mathrm{d}S_{\boldsymbol{r}}, & \text{3D.} \end{cases} \tag{1.7}$$

where $\boldsymbol{r}' \in \Omega$ is the target, and $\boldsymbol{r} \in \Gamma$ is the source. τ is a real, scalar, unknown density function that needs to be solved on Γ. Since the above convolution kernels in (1.6) and (1.7) are fundamental solutions to the Laplace equation, all that is left to ensure that the *ansatz* solves the BVP is to enforce the boundary condition $u = g$ in (1.1). To solve the unknown density τ on Γ, we let $\boldsymbol{r}'$ tend to Γ to obtain a BIE from (1.6) or (1.7). The integral in (1.6) is weakly singular, and the DLP integral in (1.7) must be taken in the principal value sense, therefore incurs a standard jump relation. A detailed discussion of the significance and properties of these potentials will be deferred until later Chapters. The resulting BIE is then

$$\mathcal{S}[\tau](\boldsymbol{r}') = g(\boldsymbol{r}'), \quad \forall \quad \boldsymbol{r}' \in \Gamma, \tag{1.8}$$

or

$$\pm\frac{1}{2}\tau(\boldsymbol{r'}) + \mathcal{D}[\tau](\boldsymbol{r'}) = g(\boldsymbol{r'}), \quad \forall \quad \boldsymbol{r'} \in \Gamma, \tag{1.9}$$

The limiting values of the DLP formulation satisfy the +1/2 jump relation for the exterior BVP, and $-1/2$ for the interior BVP. Once we obtain the unknown density function τ on Γ, by solving (1.8) or (1.9), we can evaluate the potential field u at any point in Ω by using (1.6) or (1.7) respectively.

Similarly, we can use the free-space Green's function for the Stokes equation, given by

$$\boldsymbol{G}(\boldsymbol{r}) = \begin{cases} \frac{1}{4\pi\mu}\left(I\log\frac{1}{|\boldsymbol{r}|} + \frac{\boldsymbol{r}\otimes\boldsymbol{r}}{|\boldsymbol{r}|^2}\right), & \text{2D,} \\ \frac{1}{8\pi\mu}\left(I\frac{1}{|\boldsymbol{r}|} + \frac{\boldsymbol{r}\otimes\boldsymbol{r}}{|\boldsymbol{r}|^3}\right), & \text{3D.} \end{cases} \tag{1.10}$$

and the double layer kernel, given by

$$\boldsymbol{K}(\boldsymbol{r}) = \begin{cases} \frac{1}{\pi}\frac{\boldsymbol{r}\cdot\boldsymbol{n}}{|\boldsymbol{r}|^2}\frac{\boldsymbol{r}\otimes\boldsymbol{r}}{|\boldsymbol{r}|^2}, & \text{2D,} \\ \frac{3}{4\pi}\frac{\boldsymbol{r}\cdot\boldsymbol{n}}{|\boldsymbol{r}|^2}\frac{\boldsymbol{r}\otimes\boldsymbol{r}}{|\boldsymbol{r}|^3}, & \text{3D.} \end{cases} \tag{1.11}$$

to write down the Stokes single- or double-layer potentials *ansatz*:

$$\mathbf{S}[\boldsymbol{\sigma}](\boldsymbol{r'}) = \begin{cases} \frac{1}{4\pi\mu}\int_\Gamma\left(I\log\frac{1}{|\boldsymbol{r'}-\boldsymbol{r}|} + \frac{(\boldsymbol{r'}-\boldsymbol{r})\otimes(\boldsymbol{r'}-\boldsymbol{r})}{|\boldsymbol{r'}-\boldsymbol{r}|^2}\right)\boldsymbol{\sigma}(\boldsymbol{r})\,\mathrm{d}s_{\boldsymbol{r}}, & \text{2D,} \\ \frac{1}{8\pi\mu}\int_\Gamma\left(I\frac{1}{|\boldsymbol{r'}-\boldsymbol{r}|} + \frac{(\boldsymbol{r'}-\boldsymbol{r})\otimes(\boldsymbol{r'}-\boldsymbol{r})}{|\boldsymbol{r'}-\boldsymbol{r}|^3}\right)\boldsymbol{\sigma}(\boldsymbol{r})\,\mathrm{d}S_{\boldsymbol{r}}, & \text{3D.} \end{cases} \tag{1.12}$$

$$\mathbf{D}[\boldsymbol{\sigma}](\boldsymbol{r'}) = \begin{cases} \frac{1}{\pi}\int_\Gamma\frac{(\boldsymbol{r'}-\boldsymbol{r})\cdot\boldsymbol{n}_{\boldsymbol{r}}}{|\boldsymbol{r'}-\boldsymbol{r}|^2}\frac{(\boldsymbol{r'}-\boldsymbol{r})\otimes(\boldsymbol{r'}-\boldsymbol{r})}{|\boldsymbol{r'}-\boldsymbol{r}|^2}\boldsymbol{\sigma}(\boldsymbol{r})\,\mathrm{d}s_{\boldsymbol{r}}, & \text{2D,} \\ \frac{3}{4\pi}\int_\Gamma\frac{(\boldsymbol{r'}-\boldsymbol{r})\cdot\boldsymbol{n}_{\boldsymbol{r}}}{|\boldsymbol{r'}-\boldsymbol{r}|^2}\frac{(\boldsymbol{r'}-\boldsymbol{r})\otimes(\boldsymbol{r'}-\boldsymbol{r})}{|\boldsymbol{r'}-\boldsymbol{r}|^3}\boldsymbol{\sigma}(\boldsymbol{r})\,\mathrm{d}S_{\boldsymbol{r}}, & \text{3D.} \end{cases} \tag{1.13}$$

where $\boldsymbol{\sigma}$ is an unknown vector density function that can be solved following the same procedure as in the case of Laplace.

For Poisson's equation, one could solve (1.2) by direct evaluation of a particular solution in the form of a volume integral

$$u^P(\boldsymbol{r'}) = \mathcal{V}[f]\,(\boldsymbol{r'}) = -\frac{1}{2\pi}\int_\Omega \log\left(|\boldsymbol{r'}-\boldsymbol{r}|\right)\,f(\boldsymbol{r})\,\mathrm{d}A, \tag{1.14}$$

then followed by solving a homogeneous Laplace problem with corrected boundary data

$$-\Delta u^H = 0 \quad \text{in} \quad \Omega, \quad u^H = g - u^P \quad \text{on} \quad \Gamma. \tag{1.15}$$

Throughout this thesis, we choose to use the single plus double layer formulation to represent the solution whenever possible, since it leads to a second-kind Fredholm equation, and is able to represent both interior and exterior problem. For axisymmetric microswimmer simulations, we stick with Stokes SLP formulation for simplicity reasons. As for the Laplace volume potential, we also restrict our discussion to the implementation of evaluating $\int_\Omega G(\boldsymbol{r})\mathrm{d}A$. The extensions to high order derivatives are briefly discussed in Chapter VIII.

1.3 Contribution and Thesis Overview

One of the major problems plaguing integral equation methods-based solvers is that the resulting solutions become inaccurate close to the interfaces. Naive smooth quadrature does not work well as the integrals become nearly singular. The first half of this thesis, including Chapter II to IV, is about developing product integration schemes for layer and volume potentials to eliminate the error arising from these singular and nearly-singular kernel integrations. We also benchmark the accuracy and computational efficiency of our proposed schemes on a variety of typical problems therein.

The second half of this thesis, consisting of Chapter V to VII, focuses on understanding hydrodynamic efficiency in microstructures, in which case flows are governed by Stokes equation. These low-Reynolds locomotions have long been studied and analyzed to provide insights for designing artificial micro-swimmers, and to understand pathology of ciliary dysfunction. Analytical *ansatz* can be limited only to ideal conditions, which could lead to design choices departing significantly from optimal in experiments. We have successfully applied our solvers to several studies, including simulating cilia-driven mixing and transport in complicated microstructures in two dimensions, finding optimal slip velocities profile and optimal ciliary locomotion of axisymmetric micro-swimmers with arbitrary shapes.

Finally, in Chapter VIII, we comment on some of the observations and directions that we are currently pursuing.

CHAPTER II

Product Integration Scheme for 2D Singular Layer Potentials

Preamble. In this chapter, we present a fast, high-order accurate and adaptive boundary integral scheme for solving the Stokes equations in complex—possibly nonsmooth—geometries in two dimensions. We apply the panel-based quadratures of Helsing and coworkers [80, 77, 138] to evaluate to high accuracy the weakly-singular, hyper-singular, and super-singular integrals arising in the Nyström discretization, and also the near-singular integrals needed for flow and traction evaluation close to boundaries. The resulting linear system is solved iteratively via calls to a Stokes fast multipole method. We include an automatic algorithm to "panelize" a given geometry, and choose a panel order, which will efficiently approximate the density (and hence solution) to a user-prescribed tolerance. We show that this adaptive panel refinement procedure works well in practice even in the case of complex geometries with large number of corners, or close-to-touching smooth curves. In one example, for instance, a model 2D vascular network with 378 corners required less than 200K discretization points to obtain a 9-digit solution accuracy. This is joint work with Bowei Wu, Alex Barnett and Shravan Veerapaneni, and is published in [211].

2.1 Introduction

Since the pioneering work of Youngren and Acrivos [96, 97], boundary integral equation (BIE) methods have been widely used for studying various particulate Stokes flow systems including drops, bubbles, capsules, vesicles, red blood cells and swimmers (e.g., see recent works [178, 159, 29] and references therein). BIE methods exploit the linearity of Stokes equations and offer several advantages such as reduction in dimensionality, exact satisfaction of far-field boundary conditions, and ease of handling moving geometries. Development of fast algorithms such as the fast multipole methods (FMMs) [62, 213, 205, 190, 54, 126], Ewald summation methods [165, 119, 109, 3], and variants tailored to Stokes equations [168, 217, 208, 207] further extended their scope for solving problems in physically-realistic parameter regimes. Consequently, they are being used to investigate problems in a wide range of physical scales, from microhydrodynamics of isolated particles to large-scale flows generated by suspensions of particles (e.g., see [158, 136, 212]).

Despite their success, many numerical challenges still remain open for BIEs as applied to particulate Stokes flow problems. Prominent among them is the accurate handling of hydrodynamic interactions between surfaces that are almost in contact. For example, in dense suspension flows, the constituent particles often approach very close to each other or to the walls of the enclosing geometries (e.g., see Figure 2.1). To prevent artificial instabilities in this setting, numerical methods often require adaptive spatial discretizations, accurate nearly-singular integral evaluation schemes (noting that while this issue is specific to BIEs, it manifests itself in other forms for other numerical methods) and accurate time-stepping schemes. The primary focus of this chapter is addressing the first two issues for two-dimensional problems.

In this work we apply specialized panel quadrature schemes that can accurately evaluate layer potentials defined on a smooth open curve, and for target points arbitrarily close to, or on, the curve. This helps the efficient tackling of dense particulate flows constrained in large multiply-connected domains such as in Figure 2.1. In addition, we formulate a set of rules for refining (or coarsening) the panels used to represent the boundaries, so that a

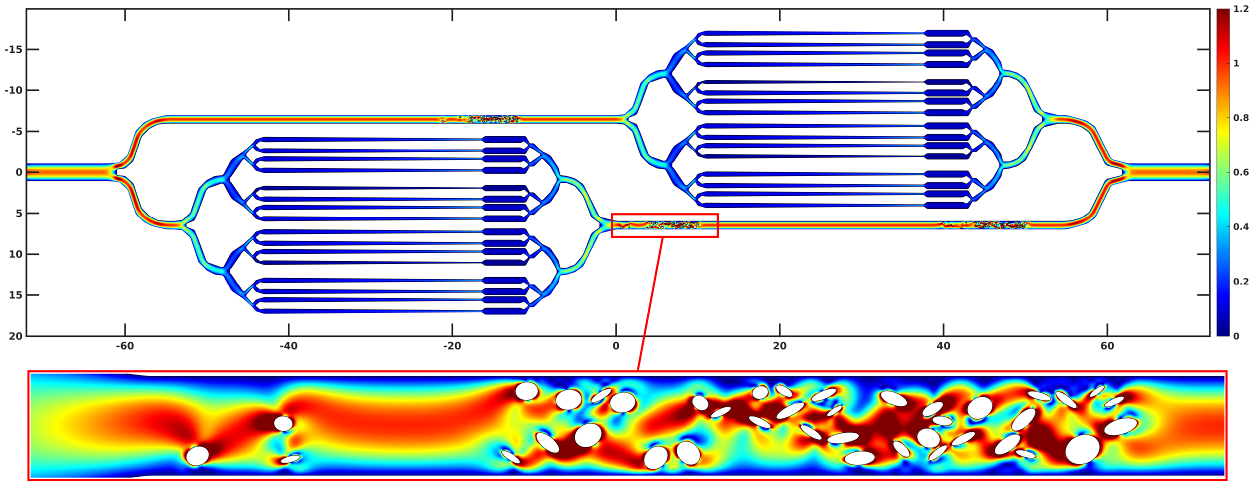

Figure 2.1: Snapshot from a simulation of bacterial supension flow in a microfluidic chip geometry, which is inspired from the design proposed in [89]. A squirmer model [91] is used for modeling the bacteria, which treats them as rigid bodies with a prescribed slip at the fluid-structure interface. Thereby, we solve the Stokes equations with a no-slip boundary condition on the microfluidic chip geometry, a prescribed tangential velocity on the squirmer boundaries and an imposed parabolic flow profile at the inlet and outlet. We used $730,080$ discretization points for the chip boundary, resulting in $1,460,160$ degrees of freedom, and 128 discretization points at each of the 120 squirmers. GMRES took about 10 hours to reach a relative residual of 5.6×10^{-8}, using an 8-core 3.6 GHz Intel Core i7 processor with 128 GB of RAM. Color indicates the magnitude of fluid velocity. The estimated PDE relative L_2-norm error is 2×10^{-5}.

user-specified error tolerance can be achieved automatically. One of the advantages of this adaptive panel refinement procedure is that it can handle geometries with corners (as in Figure 2.2) or nearly self-touching geometries, using the same quadrature schemes for both weakly- and nearly-singular integrals.

Our work is closely related to two recent efforts, that of Barnett et al. [14] and Ojala–Tornberg [139], both of which, in turn, are extensions to Stokes potentials of two distinct quadrature schemes originating with the work of Helsing–Ojala [80]. In [14], three of the co-authors developed high-order global quadrature schemes based on the periodic trapezoidal rule (PTR) for the evaluation of Laplace and Stokes layer potentials defined on a smooth closed planar curve, which achieved spectral accuracy for particles arbitrarily close to each other. Nevertheless, a key limitation of these close evaluation schemes is that they only work for closed curves and have uniform resolution on any part of a curve. Thereby, while they are well-suited for moving rigid and deformable particles immersed in Stokes flow, new machinery is needed for the constraining complex geometries such as those in Figures 2.1 and 2.2.

On the other hand, Ojala–Tornberg [139] developed a panel quadrature scheme, as is done in this work. They applied high-order quadratures for logarithmic, Cauchy, and hypersingular contour integrals pioneered by Helsing and Ojala in the context of Laplace boundary-value problems [80, 77, 138]. The main distinction from our work is that [139] uses the BIE framework of Kropinski [108], converting Stokes equations into a biharmonic equation, then tailors it to a specific application (droplet hydrodynamics), whereas our scheme directly tackles all the Stokes layer potentials using physical variables (single-layer, double-layer and their associated pressure and traction kernels). Thereby, it can be integrated with existing BIE methods developed for various physical problems (colloids, drops, vesicles, squirmers and other suspensions) and flow conditions (imposed flow, pressure-, electrically- or magnetically-driven flows, etc) more naturally. Our physical variable formulation also applies quadratures for the above three contour integrals plus the super-singular integral; the resulting panel quadratures are a subset of those developed recently for the (modified) biharmonic case [79, Sec. 6]. Note that resolving nearly-singular integrals, and corner problems, is an active area of research owing to its importance in solving several other linear elliptic partial differential equations (PDEs), such as Laplace, Helmholtz and biharmonic equations, via BIEs; other recent works that considered two-dimensional problems include [103, 15, 78, 30, 157, 4, 149].

The predominant class of algorithms for adaptive meshing found in the Stokes BIE literature are the *reparameterization* schemes (also called the *resampling* techniques) that are dedicated to resolving boundary mesh distortions arising in deformable particle flow simulations (see [199] for a review on this topic). The primary focus of this work, on the other hand, is to determine an optimal distribution of boundary panels on a given domain, and a choice of panel order p, so that a user-prescribed tolerance is met when computing the solution. Prior work in this area has mostly been restricted to low-order boundary element methods (BEMs); see [102] for a review. In the last 20 years, high-order *hp*-variants of BEM have been tested for Laplace and Helmholtz problems on polygons (for example [85, 11]). However, we are not aware of any Nyström *hp*-BIEs for the Stokes equations capable of handling arbitrary complex geometries. To handle corners, the RCIP method developed by Helsing [82] compresses the degrees of freedom used in a graded mesh, and has been recently applied to the biharmonic case [79]. The recent research of Rachh–Serkh [156] exploits a power-law basis resulting from analysis of the wedge problem to solve Stokes corner problems via a BIE. Here, we take a more pedestrian approach to handling corner singularities via the use of graded meshes. A prototypical example is shown in Figure 2.2.

This chapter is organized as follows. In Section 2.2, we define the multiply-connected viscous flow boundary value problem, give the integral equation formulation, and formulate the Stokes boundary integral operators in terms of Laplace and complex contour integrals. In Section 2.3, we review panel quadrature rules for evaluating the Stokes layer potentials, including key details about branch cuts and efficient filling of matrix blocks. We summarize our algorithm for adaptively discretizing a given geometry in Section 2.4. We consider several test cases and present results on the performance of our algorithms in Section 2.5, including a demonstration of how the adaptive scheme can handle nearly-touching smooth curves efficiently. We conclude in Section 2.6.

2.2 Mathematical preliminaries

In this section, we first state the PDE formulation for multiply-connected interior (as in Figure 2.2) or exterior flow problems, reformulate them as a BIE and then represent the resulting layer potentials in terms of contour integrals and Laplace potentials.

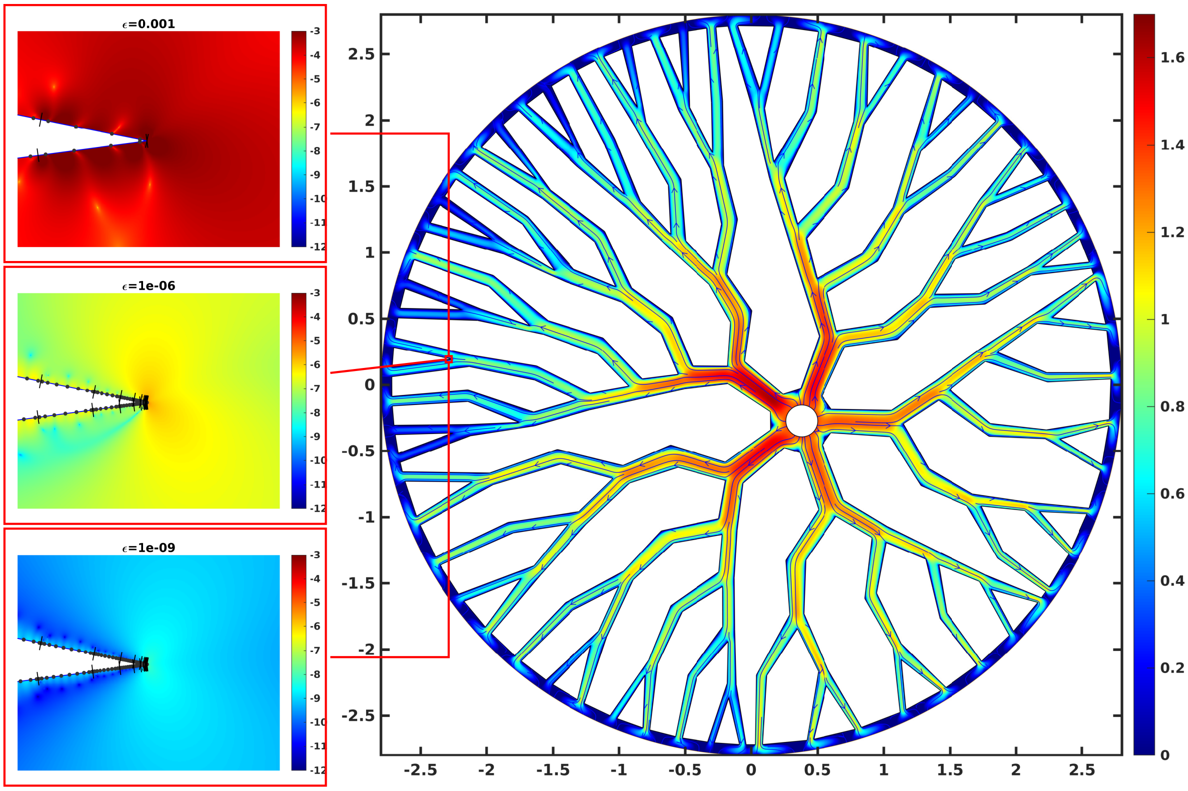

Figure 2.2: Solution of the Stokes equation in a nonsmooth circular vascular network with Dirichlet boundary condition. We apply no-slip boundary condition at all branch walls, and it is driven by a uniform flow from inner to outer circle. Color here indicates log of the magnitude of fluid velocity. We used automatically generated panels for both smooth boundaries and 378 corners, resulting in $356,580$ degrees of freedom. GMRES took about 1 hour to reach a relative residual of 7.61×10^{-11} on an 8-core 4.0 GHz Intel Core i7 desktop. The PDE solution has a relative L_2-norm error of 1×10^{-9}. Three high-resolution $\log_{10}$ error plots that correspond to different user-requested tolerance ϵ near the same reentrant corner are shown on the left; here the short normal lines show panel endpoints, and the black dots quadrature nodes.

2.2.1 Boundary value problem and its integral equation formulation

The fluid domain Ω in Figure 2.2 is a multiply-connected region bounded by N_Γ closed curves, $\{\Gamma_i, i = 1, \ldots, N_\Gamma\}$. Without loss of generality, let Γ_1 be the all-enclosing boundary—i.e., the outer circle in Figure 2.2—and let $\Gamma = \cup_{i=1}^{N_\Gamma} \Gamma_i$. Denoting the fluid viscosity by μ, the velocity by u and the pressure by p, the governing boundary-value problem, in the vanishing Reynolds number limit, is

$$-\mu \Delta \boldsymbol{u} + \nabla p = 0 \quad \text{and} \quad \nabla \cdot \boldsymbol{u} = 0 \quad \text{in} \quad \Omega, \tag{2.1a}$$

$$\boldsymbol{u} = \boldsymbol{g} \quad \text{on} \quad \Gamma. \tag{2.1b}$$

The Dirichlet data $\boldsymbol{g}$ must satisfy the consistency condition $\int_\Gamma \boldsymbol{g}\cdot\boldsymbol{n}\mathrm{d}s = 0$, where $\boldsymbol{n}$ is the normal to Γ. For example, in Figure 2.2, the flow is driven by an outward flow condition at the inner circle and an inward flow condition at the outer circle. Their magnitude is chosen such that the consistency condition is respected. On the rest of the curves, a no-slip ($\boldsymbol{g} = \boldsymbol{0}$) boundary condition is enforced.

While there are many approaches for reformulating the Dirichlet problem (2.1) as a BIE [152, 88], for simplicity, we use an indirect, combined-field BIE formulation that leads to a well-conditioned and non-rank-deficient linear system. There are alternative indirect formulations, combined-field BIE formulation works well in practice for both interior and exterior problems. We make the following *ansatz*:

$$\boldsymbol{u}(\boldsymbol{r}') = \sum_{i=1}^{N_\Gamma} \mathbf{S}_i[\boldsymbol{\sigma}](\boldsymbol{r}') + \mathbf{D}_i[\boldsymbol{\sigma}](\boldsymbol{r}') := \mathbf{S}[\boldsymbol{\sigma}](\boldsymbol{r}') + \mathbf{D}[\boldsymbol{\sigma}](\boldsymbol{r}')\ , \qquad \boldsymbol{r}' \in \Omega, \tag{2.2}$$

where $\boldsymbol{\sigma}$ is an unknown vector "density" function to be determined, $\mathbf{S}_i$ is the velocity Stokes single layer potential (SLP) and $\mathbf{D}_i$ is the double layer potential (DLP), defined by

$$\mathbf{S}_i[\boldsymbol{\sigma}](\boldsymbol{r}') = \frac{1}{4\pi\mu}\int_{\Gamma_i}\left(I\log\frac{1}{|\boldsymbol{r}'-\boldsymbol{r}|} + \frac{(\boldsymbol{r}'-\boldsymbol{r})\otimes(\boldsymbol{r}'-\boldsymbol{r})}{|\boldsymbol{r}'-\boldsymbol{r}|^2}\right)\boldsymbol{\sigma}(\boldsymbol{r})\,\mathrm{d}s_{\boldsymbol{r}}, \qquad \boldsymbol{r}'\in\Omega, \tag{2.3}$$

$$\mathbf{D}_i[\boldsymbol{\sigma}](\boldsymbol{r}') = \frac{1}{\pi}\int_{\Gamma_i}\left(\frac{(\boldsymbol{r}'-\boldsymbol{r})\cdot\boldsymbol{n}_{\boldsymbol{r}}}{|\boldsymbol{r}'-\boldsymbol{r}|^2}\frac{(\boldsymbol{r}'-\boldsymbol{r})\otimes(\boldsymbol{r}'-\boldsymbol{r})}{|\boldsymbol{r}'-\boldsymbol{r}|^2}\right)\boldsymbol{\sigma}(\boldsymbol{r})\,\mathrm{d}s_{\boldsymbol{r}}, \qquad \boldsymbol{r}'\in\Omega. \tag{2.4}$$

Here, I is the 2-by-2 identity tensor, $\boldsymbol{r} = [x, y]^T$, $\boldsymbol{\sigma} = [\sigma_1, \sigma_2]^T$, and $\mathrm{d}s_{\boldsymbol{r}}$ is the arc length element on Γ. As (2.2) indicates, we use $\mathbf{S}$ or $\mathbf{D}$ to denote the sum of layer potentials due to all source curves. When combined with their associated pressure kernels (given in the Appendix), the convolution kernels above are fundamental solutions to the Stokes equations (2.1a); therefore, all that is left to ensure that the *ansatz* (2.2) solves the boundary-value problem is to enforce the velocity boundary condition (2.1b).

We introduce the operator block notations $\mathbb{S}_{ij}\boldsymbol{\sigma} = \mathbf{S}_j[\boldsymbol{\sigma}](\Gamma_i)$ and $\mathbb{D}_{ij}\boldsymbol{\sigma} = \mathbf{D}_j[\boldsymbol{\sigma}](\Gamma_i)$, i.e., the layer velocity potentials due to a source curve Γ_j, with target restricted to the curve Γ_i. When $i = j$, for the SLP case this integral is improper, and for the DLP the integral must be taken in the principal value sense. By taking the limit as $\boldsymbol{r}'$ approaches Γ (from the interior or exterior for Γ_1, and exterior for the rest of the curves), the standard jump relations for the single and double layer potentials [152, 88] give the interior case of the following $N_\Gamma \times N_\Gamma$ BIE system:

$$\left[\begin{pmatrix} I/2 & 0 & \dots \\ 0 & I/2 & \dots \\ \vdots & \vdots & \ddots \end{pmatrix} + \begin{pmatrix} \eta(\mathbb{S}_{11}+\mathbb{D}_{11}) & \mathbb{S}_{12}+\mathbb{D}_{12} & \dots \\ \eta(\mathbb{S}_{21}+\mathbb{D}_{21}) & \mathbb{S}_{22}+\mathbb{D}_{22} & \dots \\ \vdots & \vdots & \ddots \end{pmatrix}\right] \begin{pmatrix} \eta\boldsymbol{\sigma}(\Gamma_1) \\ \boldsymbol{\sigma}(\Gamma_2) \\ \vdots \end{pmatrix} = \begin{pmatrix} \boldsymbol{f}(\Gamma_1) \\ \boldsymbol{f}(\Gamma_2) \\ \vdots \end{pmatrix}, \tag{2.5}$$

Here $\eta = -1$ for the exterior case, and $\eta = 1$ for the interior case. The $I/2$ blocks resulting from the DLP mean that (2.5) is of Fredholm second kind. The admixture of SLP and DLP in (2.2) insures that (2.5) there is no nullspace induced by each closed curve $i = 2, \dots, N_\Gamma$ [76, Thm. 2.1] [152, p.128]. For the case $\eta = +1$ this is also true for Γ_1, so that there is no nullspace; in the case $\eta = -1$ where Γ_1 encloses the entire geometry, it still introduces a nullity of 1, associated with overall pressure changes. Note that the negation of the density for the outer curve (first block column) results in all positive identity blocks. Once we obtain the unknown density function $\boldsymbol{\sigma}$ on Γ, by solving (2.5), we can evaluate the velocity at any point in the fluid domain by using 2.2.

We also consider the *exterior* boundary-value problem (see, e.g., Figure 2.6), which can be thought of taking the above outer curve Γ_1 to infinity, removing it from the problem. The fluid domain Ω becomes the entire plane minus the closed interiors of the other curves; these curves we may relabel as $\Gamma = \cup_{i=1}^{N_\Gamma} \Gamma_i$. There is also now a given imposed background flow $\boldsymbol{u}_\infty(\boldsymbol{r}')$, which in applications is commonly a uniform, shear, or extensional flow. The decay condition $\boldsymbol{u}(\boldsymbol{r}') - \boldsymbol{u}_\infty(\boldsymbol{r}') = \boldsymbol{\Lambda} \log|\boldsymbol{r}'| + O(1)$, for some $\boldsymbol{\Lambda} \in \mathbb{R}^2$, must be appended to (2.5) to give a well-posed BVP [88]. The representation (2.2) of the physical velocity is now

$$\boldsymbol{u}(\boldsymbol{r}') = \boldsymbol{u}_\infty(\boldsymbol{r}') + (\mathbf{D}[\boldsymbol{\sigma}] + \mathbf{S}[\boldsymbol{\sigma}])(\boldsymbol{r}') \ . \tag{2.6}$$

The resulting BIE is given as the second case of (2.5). In the simplest case of physical no-slip boundary conditions, the data in (2.5) is now $\boldsymbol{f} = -\boldsymbol{u}_\infty|_\Gamma$, which one may check cancels the velocity on all boundaries.

The rest of this section rewrites the Stokes potentials $\mathbf{S}\boldsymbol{\sigma}$ and $\mathbf{D}\boldsymbol{\sigma}$ in terms of four fundamental complex contour integrals. Laplace layer potentials are also used as a convenient intermediate step. The motivation is to interface to the close-evaluation schemes of Helsing and co-workers [80, 77, 138, 79], which will be used both for on-curve evaluation (discretization of (2.5)) and near-to-curve evaluation of the representation (2.2).

2.2.2 Fundamental contour integrals

Let τ be a given, possibly complex, scalar density function on Γ. We associate $\mathbb{R}^2$ with the complex plane $\mathbb{C}$. Let n_z be the outward-pointing unit vector at $z \in \Gamma$, expressed as a complex number. Given a target point $z' \in \Omega$, we define the potentials

$$I_L = I_L[\tau](z') := \int_\Gamma \log|z'-z|\tau(z)|\mathrm{d}z| = \int_\Gamma \log|z'-z|\frac{\tau(z)}{in_z}\mathrm{d}z\ , \quad \text{(real logarithmic)} \tag{2.7}$$

$$I_C = I_C[\tau](z') := \int_\Gamma \frac{\tau(z)}{z-z'}\mathrm{d}z\ , \quad \text{(Cauchy)} \tag{2.8}$$

$$I_H = I_H[\tau](z') := \int_\Gamma \frac{\tau(z)}{(z-z')^2}\mathrm{d}z\ , \quad \text{(Hadamard)} \tag{2.9}$$

$$I_S = I_S[\tau](z') := \int_\Gamma \frac{\tau(z)}{(z-z')^3}\mathrm{d}z\ . \quad \text{(supersingular)} \tag{2.10}$$

Note that, as functions of target point z', I_C, I_H and I_S are holomorphic functions in Ω, that $(\mathrm{d}/\mathrm{d}z')I_C[\tau](z') = I_H[\tau](z')$, and $(\mathrm{d}/\mathrm{d}z')I_H[\tau](z') = 2I_S[\tau](z')$. In contrast, I_L is not generally holomorphic; yet, for τ real, I_L is the real part of a holomorphic function. The Stokes single- and double-layer traction kernels can be written in terms of Hadamard I_H and supersingular I_S integrals respectively.

2.2.3 Laplace layer potentials

Now, let τ be a real, scalar, density function on Γ. The Laplace single- and double-layer potentials are defined, respectively, by

$$\mathcal{S}[\tau](\boldsymbol{r}') := \frac{1}{2\pi}\int_\Gamma \left(\log \frac{1}{|\boldsymbol{r}'-\boldsymbol{r}|}\right)\tau(\boldsymbol{r})\mathrm{d}s_{\boldsymbol{r}}, \tag{2.11}$$

$$\mathcal{D}[\tau](\boldsymbol{r}') := \frac{1}{2\pi}\int_\Gamma \left(\frac{\partial}{\partial \boldsymbol{n}_{\boldsymbol{r}}}\log \frac{1}{|\boldsymbol{r}'-\boldsymbol{r}|}\right)\tau(\boldsymbol{r})\mathrm{d}s_{\boldsymbol{r}} = \frac{1}{2\pi}\int_\Gamma \left(\frac{(\boldsymbol{r}'-\boldsymbol{r})\cdot\boldsymbol{n}_{\boldsymbol{r}}}{|\boldsymbol{r}'-\boldsymbol{r}|^2}\right)\tau(\boldsymbol{r})\mathrm{d}s_{\boldsymbol{r}}, \tag{2.12}$$

where $\boldsymbol{r}' \in \Omega$. In terms of the contour integrals (2.7) and (2.8), using $\boldsymbol{r} = [x,y]^T$, $\boldsymbol{n}_{\boldsymbol{r}} = [n_1,n_2]^T$, $z = x+iy$, $n_z = n_1 + in_2$, $\mathrm{d}s_{\boldsymbol{r}} = |\mathrm{d}z| = \mathrm{d}z/in_z$, and the restriction to τ real, we can rewrite the SLP and DLP as

$$\mathcal{S}[\tau](\boldsymbol{r}') = \frac{-1}{2\pi}I_L[\tau](z') \quad \text{and} \quad \mathcal{D}[\tau](\boldsymbol{r}') = \mathrm{Re}\,\frac{i}{2\pi}I_C[\tau](z'), \quad z' \in \Omega. \tag{2.13}$$

We will also need the gradients and Hessians of Laplace layer potentials. The gradient of the SLP has components

$$\frac{\partial}{\partial x'}\mathcal{S}[\tau](\boldsymbol{r}') = \frac{1}{2\pi}\mathrm{Re}\left(I_C[\tau/in_z]\right)(z'), \qquad \frac{\partial}{\partial y'}\mathcal{S}[\tau](\boldsymbol{r}') = \frac{-1}{2\pi}\mathrm{Im}\left(I_C[\tau/in_z]\right)(z')\ . \tag{2.14}$$

The gradient of the DLP requires the Hadamard kernel, and has components

$$\frac{\partial}{\partial x'}\mathcal{D}[\tau](\boldsymbol{r}') = \frac{-1}{2\pi}\mathrm{Im}\, I_H[\tau](z'), \qquad \frac{\partial}{\partial y'}\mathcal{D}[\tau](\boldsymbol{r}') = \frac{-1}{2\pi}\mathrm{Re}\, I_H[\tau](z')\ . \tag{2.15}$$

The Hessians, which also require I_S, are discussed in Appendix A.1.

2.2.4 Stokes velocity layer potentials

As in [14], we rewrite the Stokes SLP in terms of Laplace potentials, and the DLP in terms of Laplace and Cauchy potentials. Using the identity

$$\frac{(\boldsymbol{r}'-\boldsymbol{r})\otimes(\boldsymbol{r}'-\boldsymbol{r})}{|\boldsymbol{r}'-\boldsymbol{r}|^2}\boldsymbol{\sigma} = \frac{(\boldsymbol{r}'-\boldsymbol{r})}{|\boldsymbol{r}'-\boldsymbol{r}|^2}((\boldsymbol{r}'-\boldsymbol{r})\cdot\boldsymbol{\sigma}) = ((\boldsymbol{r}'-\boldsymbol{r})\cdot\boldsymbol{\sigma})\nabla_{\boldsymbol{r}'}\log|\boldsymbol{r}'-\boldsymbol{r}|,$$

we can rewrite the Stokes SLP in terms of the Laplace SLP (2.11) as

$$
\begin{aligned}
\mathbf{S}[\boldsymbol{\sigma}](\boldsymbol{r}') \; &= \frac{1}{4\pi\mu}\Big\{ \int_\Gamma \left(\log \frac{1}{|\boldsymbol{r}'-\boldsymbol{r}|}\right) \boldsymbol{\sigma} \mathrm{d}s_{\boldsymbol{r}} + \nabla \int_\Gamma \left(\log \frac{1}{|\boldsymbol{r}'-\boldsymbol{r}|}\right) (\boldsymbol{r}\cdot\boldsymbol{\sigma}) \mathrm{d}s_{\boldsymbol{r}} \\
&\qquad - x'\nabla \int_\Gamma \left(\log \frac{1}{|\boldsymbol{r}'-\boldsymbol{r}|}\right) \sigma_1 \mathrm{d}s_{\boldsymbol{r}} - y'\nabla \int_\Gamma \left(\log \frac{1}{|\boldsymbol{r}'-\boldsymbol{r}|}\right) \sigma_2 \mathrm{d}s_{\boldsymbol{r}} \Big\} \; ,
\end{aligned}
\tag{2.16}
$$

where $\nabla = \nabla_{\boldsymbol{r}'}$ is assumed from now on. Therefore, three Laplace potentials (and their first derivatives) with real scalar density functions $\boldsymbol{r}\cdot\boldsymbol{\sigma}$, σ_1, and σ_2 need to be computed to evaluate the Stokes SLP. Similarly, using the identity

$$
\nabla \left(\frac{(\boldsymbol{r}'-\boldsymbol{r})\cdot \boldsymbol{n}_{\boldsymbol{r}}}{|\boldsymbol{r}'-\boldsymbol{r}|^2} \right) = \frac{\boldsymbol{n}_{\boldsymbol{r}}}{|\boldsymbol{r}'-\boldsymbol{r}|^2} - 2((\boldsymbol{r}'-\boldsymbol{r})\cdot\boldsymbol{n}_{\boldsymbol{r}})\frac{\boldsymbol{r}'-\boldsymbol{r}}{|\boldsymbol{r}'-\boldsymbol{r}|^4} \; ,
\tag{2.17}
$$

the Stokes DLP can be written as

$$
\begin{aligned}
\mathbf{D}[\boldsymbol{\sigma}](\boldsymbol{r}') \; &= \frac{1}{2\pi} \int_\Gamma \frac{\boldsymbol{n}_{\boldsymbol{r}}}{|\boldsymbol{r}'-\boldsymbol{r}|^2}((\boldsymbol{r}'-\boldsymbol{r})\cdot\boldsymbol{\sigma})\mathrm{d}s_{\boldsymbol{r}} + \frac{1}{2\pi}\nabla \int_\Gamma (\boldsymbol{r}'-\boldsymbol{r})\cdot \frac{\boldsymbol{n}_{\boldsymbol{r}}}{|\boldsymbol{r}'-\boldsymbol{r}|^2}(\boldsymbol{r}\cdot\boldsymbol{\sigma})\mathrm{d}s_{\boldsymbol{r}} \\
&\qquad - \frac{1}{2\pi} x \nabla \int_\Gamma \frac{(\boldsymbol{r}'-\boldsymbol{r})\cdot\boldsymbol{n}_{\boldsymbol{r}}}{|\boldsymbol{r}'-\boldsymbol{r}|^2}\sigma_1 \mathrm{d}s_{\boldsymbol{r}} - \frac{1}{2\pi} y \nabla \int_\Gamma \frac{(\boldsymbol{r}'-\boldsymbol{r})\cdot\boldsymbol{n}_{\boldsymbol{r}}}{|\boldsymbol{r}'-\boldsymbol{r}|^2}\sigma_2 \mathrm{d}s_{\boldsymbol{r}} \; .
\end{aligned}
\tag{2.18}
$$

While the last three terms are gradients of Laplace DLPs, the first term requires a Cauchy integral. More concisely, we can write (2.16) and (2.18) as

(2.19)

$$
\mathbf{S}[\boldsymbol{\sigma}](\boldsymbol{r}') = \frac{1}{2\mu}\Big((\mathcal{S}[\sigma_1], \mathcal{S}[\sigma_2]) + \nabla\mathcal{S}[\boldsymbol{r}\cdot\boldsymbol{\sigma}] - x\nabla\mathcal{S}[\sigma_1] - y\nabla\mathcal{S}[\sigma_2]\Big)(\boldsymbol{r}'), \qquad \boldsymbol{r}' \in \Omega,
$$

(2.20)

$$
\mathbf{D}[\boldsymbol{\sigma}](\boldsymbol{r}') = \Big(\frac{1}{2\pi}\mathrm{Re}\left(I_C[\tau_1], I_C[\tau_2]\right) + \nabla\mathcal{D}[\boldsymbol{r}\cdot\boldsymbol{\sigma}] - x\nabla\mathcal{D}[\sigma_1] - y\nabla\mathcal{D}[\sigma_2]\Big)(\boldsymbol{r}'), \qquad \boldsymbol{r}' \in \Omega,
$$

where, as above, a pair $(\cdot,\cdot)$ indicates two vector components, and a short calculation verifies that the complex scalar density functions τ_1 and τ_2

$$
\tau_1 = (\sigma_1 + i\sigma_2)\frac{\mathrm{Re}\, n_z}{n_z}, \qquad \tau_2 = (\sigma_1 + i\sigma_2)\frac{\mathrm{Im}\, n_z}{n_z},
\tag{2.21}
$$

where n_z is $\boldsymbol{n}_{\boldsymbol{r}} = [n_1, n_2]^T$ interpreted as a complex number, makes the identity hold (i.e. the first term in (2.20) equals the first term in (2.18)).

In summary, the procedure discussed in this section enables us to express the velocity field anywhere in the fluid domain, represented by (2.2), in terms of fundamental contour integrals. Similar formulae exist for the fluid pressure and hydrodynamic stresses, which are commonly needed in several applications; we present these in Appendix A.1.

2.3 Nyström discretization and evaluation of layer potentials

2.3.1 Overview: discretization and the plain Nyström formula

Firstly, we need to specify a numerical approximation of the density function σ. For simplicity, consider the case of an exterior BVP on a single closed curve Γ parameterized by $Z : [0, 2\pi) \to \mathbb{R}^2($ or $\mathbb{C})$, such that $\Gamma = Z([0, 2\pi))$. The Stokes BIE (2.5) is then

$$(I/2 + \mathbb{D} + \mathbb{S})\boldsymbol{\sigma} = \boldsymbol{g}, \tag{2.22}$$

where $\boldsymbol{\sigma}$ and $\boldsymbol{g}$ are 2-component vector functions on Γ.

Given the user requested tolerance ϵ, the boundary is split into n_Λ disjoint panels Λ_i, $i = 1, \ldots, n_\Lambda$, using the adaptive algorithm to be described in Section 2.4. Each panel will have p nodes, giving $N = pn_\Lambda$ nodes in total, hence $2N$ scalar density unknowns. The ith panel is $\Lambda_i = Z([a_{i-1}, a_i])$, where a_i, $i = 0, \ldots, n_\Lambda$ are the parameter breakpoints of all panels (where $a_{n_\Lambda} \equiv a_0 (\mathrm{mod}\ 2\pi)$). Let the p-point Gauss–Legendre nodes and weights on the parameter interval $[a_{i-1}, a_i]$ be $t_j^{(i)}$ and $w_j^{(i)}$ respectively, for $j = 1, \ldots, p$. Then, for smooth vector functions $\boldsymbol{g}$ on Γ, the quadrature rule for $\int_\Gamma \boldsymbol{g}(\boldsymbol{r}) \mathrm{d}s_{\boldsymbol{r}}$

$$\int_0^{2\pi} \boldsymbol{g}(Z(t)) \left|\frac{\mathrm{d}}{\mathrm{d}t} Z(t)\right| dt \approx \sum_{i=1}^{n_\Lambda} \sum_{j=1}^{p} \boldsymbol{g}(\boldsymbol{r}_j^{(i)}) \left|\frac{\mathrm{d}}{\mathrm{d}t} \boldsymbol{r}_j^{(i)}\right| w_j^{(i)} =: \sum_{\ell=1}^{N} \boldsymbol{g}(\boldsymbol{r}_\ell) \left|\frac{\mathrm{d}}{\mathrm{d}t} \boldsymbol{r}_\ell\right| w_\ell \tag{2.23}$$

holds to high-order accuracy. In the last formula above $\{\boldsymbol{r}_\ell\}_{\ell=1}^N$ denotes the entire set of nodes on Γ, $t_\ell = Z^{-1}(\boldsymbol{r}_\ell)$ their preimage in parametric space $[0, 2\pi]$, and w_ℓ their corresponding weights.

The Nyström method [106, Sec. 12.2] is then used to approximately solve the BIE (2.22). Broadly speaking, this involves substituting the quadrature rule (2.23) for the integral implicit in the BIE, then enforcing the equation at the quadrature nodes themselves. The result is the $2N$-by-$2N$ linear system,

$$\boldsymbol{A}\boldsymbol{\Sigma} = \mathfrak{g} \tag{2.24}$$

where $\mathfrak{g} := \{\boldsymbol{g}(\boldsymbol{r}_\ell)\}_{\ell=1}^{N}$ is the vector of 2-component values of the right-hand side $\boldsymbol{g}$ at the N nodes, and $\boldsymbol{\Sigma} := \{\boldsymbol{\sigma}_\ell\}_{\ell=1}^{N}$ is the vector of 2-component densities at the N nodes. $\boldsymbol{A} = \{A_{ij}\}_{i,j=1,\ldots,n_\Lambda}$ is a $n_\Lambda \times n_\Lambda$ block matrix, where each block A_{ij} is a $2p \times 2p$ submatrix that represents the interaction from source panel Λ_j to target panel Λ_i. For targets that are "far" (in a sense discussed below) from a given source panel, the formula for filling corresponding elements of $\boldsymbol{A}$ is simple. Letting the index be ℓ' for such a target node, and ℓ for a source lying in such a panel, using the smooth rule (2.23) gives the matrix element

$$A_{\ell',\ell} \;=\; \left(\boldsymbol{G}(\boldsymbol{r}_{\ell'} - \boldsymbol{r}_\ell) + \boldsymbol{K}(\boldsymbol{r}_{\ell'} - \boldsymbol{r}_\ell)\right) \left|\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{r}_\ell\right| w_\ell \;, \tag{2.25}$$

(Nyström rule for $y_{\ell'}$ "far" from panel containing y_ℓ) where $\boldsymbol{G}$ and $\boldsymbol{K}$ are the kernels appearing in (2.3)–(2.4). Recall that each element in (2.25) is a 2×2 tensor, since $\boldsymbol{G}$ and $\boldsymbol{K}$ are. This defines the plain (smooth) rule for matrix elements (note that we need not include the diagonal $I/2$ from (2.22) since $\ell' = \ell$ is never a "far" interaction).

Sections 2.3.2–2.3.4 will be devoted to defining "close" vs "far" and explaining how "close" matrix elements are filled. Assuming for now that this has been done, the result is a dense matrix $\boldsymbol{A}$ that is well-conditioned independent of N, because the underlying integral equation is of Fredholm second kind. Then an iterative solver for (2.24), such as GMRES, often converges rapidly. The result is the vector $\boldsymbol{\Sigma}$ approximating the density at the nodes. Assuming that (2.24) has been solved exactly, there is still a discretization error, whose convergence rate is known to inherit that of the quadrature scheme applied to the kernel [106, Sec. 12.2]. Given this, the density function may be evaluated at any point $\boldsymbol{r} \in \Gamma$ using either the Nyström interpolant (which is global and hence inconvenient), or the local pth-order Lagrange interpolant from just the points on the panel in which $\boldsymbol{r}$ lies. When needed, we will use the latter. The generalization of the above Nyström method to multiple closed curves is clear.

Remark 2.3.1. When the problem size is large, the matrix-vector multiplication in (2.24), which requires $O(N^2)$ time, can be rapidly computed using a fast multipole method (FMM) in only $O(N)$ time. This is because all but $O(N)$ of the matrix elements involve the plain rule (2.25), for which applying the matrix is equivalent to evaluation of a potential with weighted source strengths. In our large examples (Figures 2.1 and 2.2), we use an OpenMP-parallelized Stokes FMM code due to Rachh, built upon the Goursat representation of the biharmonic kernel [154, 59].

Finally, once $\boldsymbol{\Sigma}$ has been solved for, the evaluation of the velocity $\boldsymbol{u}(\boldsymbol{r}')$ at arbitrary targets $\boldsymbol{r}' \in \Omega$ is possible, by approximating the representation (2.2) or (2.6), as appropriate. By linearity, this breaks into a sum of contributions from each source panel on each curve, which may then be handled separately. Thus a given target $\boldsymbol{r}' \in \Omega$ may, again, fall "far" or "close" to a given source panel (denoted by $\Lambda = Z([a, b])$). If it is "far" (according to the same criterion as for matrix elements), then a simple plain rule is used. Letting $\boldsymbol{u}_\Lambda$ be the contribution to u from source panel Λ, this rule arises, as with (2.25), simply by substituting (2.23) into the representation, giving the evaluation rule for x "far" from Λ

$$\begin{aligned} \boldsymbol{u}_\Lambda(\boldsymbol{r}') &= \int_\Lambda \left(\boldsymbol{G}(\boldsymbol{r}' - \boldsymbol{r}) + \boldsymbol{K}(\boldsymbol{r}' - \boldsymbol{r})\right) \boldsymbol{\sigma}(\boldsymbol{r}) \mathrm{d}s_{\boldsymbol{r}} \\ &\approx \sum_{j=1}^{p} \left(\boldsymbol{G}(\boldsymbol{r}' - \boldsymbol{r}_j) + \boldsymbol{K}(\boldsymbol{r}' - \boldsymbol{r}_j)\right) \left(\left| \frac{\mathrm{d}}{\mathrm{d}t} \boldsymbol{r}_j \right| w_j \boldsymbol{\sigma}_j \right) , \end{aligned} \tag{2.26}$$

where $\boldsymbol{r}_j := Z(t_j)$ are the nodes, and $\boldsymbol{\sigma}_j := \boldsymbol{\sigma}(t_j)$ the 2-component density values, belonging to Λ. For large N, the FMM is ideal for the task of evaluating $\boldsymbol{u}$ at many targets, using this plain rule. An identical quadrature rule may be applied to the representations in the Appendix to evaluate pressure and traction at $\boldsymbol{r}'$.

2.3.2 Close-evaluation and self-evaluation corrections

If a target (on-surface node or off-surface evaluation point) is close enough to a source panel so that the error in using (2.25) exceeds the user tolerance, a *close-evaluation* formula is needed. A special case is when the target is a node on the source panel itself, which we call *self-evaluation*, and for which we use the same formulae. We quantify “close” and “far” as follows. Given a panel $\Lambda_j = Z([a,b])$, a target point $\boldsymbol{r}'$ is *close* to Λ_j if it lies inside the ellipse

$$|\boldsymbol{r}' - Z(a)| + |\boldsymbol{r}' - Z(b)| = C\, S, \tag{2.27}$$

otherwise $\boldsymbol{r}'$ is *far* from Λ_j. A panel Λ_i is close to Λ_j if any point $\boldsymbol{r}' \in \Lambda_i$ lies inside the ellipse (2.27), otherwise Λ_i is far from Λ_j. (Of course, Λ_j is close to itself.) In (2.27), S is the arc length of Λ_j and $C > 1$ is a constant. For the numerical examples in Section 2.5 we have picked $C = 2.5$, which is large enough to include all of both neighboring panels of Λ_j most of the time.

The rationale for the above heuristic is based upon the accuracy of the plain rule (2.26) (and its corresponding matrix element rule (2.25)). Examining (2.23), if the integrand $\boldsymbol{f}(\boldsymbol{r}_\ell)\left|\frac{d}{dt}\boldsymbol{r}_\ell\right|$ is analytic with respect to t within a Bernstein ellipse (for the parameter domain $[a_{j-1}, a_j]$ for this panel) of size parameter $\varrho > 1$, then the error convergence for the Gauss–Legendre rule for this panel is $O(\varrho^{-2p})$, i.e. exponential in the panel degree p [191, Theorem 19.3]. Since in our case $\boldsymbol{f}(\boldsymbol{r}) = (\boldsymbol{G}(\boldsymbol{r}' - \boldsymbol{r}) + \boldsymbol{K}(\boldsymbol{r}' - \boldsymbol{r}))\,\boldsymbol{\sigma}(t)$, and $\boldsymbol{G}(\boldsymbol{r}' - \boldsymbol{r}) + \boldsymbol{K}(\boldsymbol{r}' - \boldsymbol{r})$ is analytic for $\boldsymbol{r}' \neq \boldsymbol{r}$, for such analyticity of the integrand, $\boldsymbol{r}'$ must be outside the *image* of this Bernstein ellipse under Z. In the case where the panel is approximately flat, this image is approximated by the ellipse with foci $Z(a)$ and $Z(b)$, which gives the above geometric “far” criterion. The choice of C is empirically made to achieve an exponential error convergence rate in p no worse than that due to the next-neighboring panels discussed in Stage 1 of Section 2.4, in the case of panel shapes produced by the procedure in that section. Note that $\boldsymbol{\sigma}(t)$ must also be assumed to be analytic in the ellipse; we expect this to hold again because the panels will be sufficiently refined. For a more detailed error analysis of the plain panel rule using the Bernstein ellipse, see [4, Sect. 3.1] and [2, Sec. 2.1].

So far we have presented (only in the "far" case) formulae for both filling Nyström matrix elements (2.25), and for evaluation of the resulting velocity potential (2.26). We now make the point that, in both the "far" and "close" cases, these are essentially the same task.

Remark 2.3.2 (Matrix-filling is potential evaluation)*.* The matrix element formula (2.25) is just a special case of the evaluation formula (2.26) for the on-surface target $\boldsymbol{r}' = \boldsymbol{r}_\ell$, and with a Kronecker-delta density $\sigma_j = \delta_{j,\ell}$. I.e., one can compute (the "far" contributions to) $\boldsymbol{A\sigma}$ from $\boldsymbol{\sigma}$, as needed for each iteration in the solution of (2.24), simply via the evaluation formula (2.26) with targets $\boldsymbol{r}'$ as the set of nodes $\{\boldsymbol{r}_\ell\}$. This will also apply for the special "close" formulae presented below; note that the diagonal blocks A_{ii} when computed using these quadratures will *implicitly* include the $I/2$ jump relation appearing in (2.22). Thus from now on we present only formulae for evaluation; the corresponding matrix element formulae are easy to extract (see Section 2.3.4).

Finally, to accelerate the computation, the close- and self-evaluations will be precomputed as matrices (see Section 2.3.4) which include subtracting the incorrect entries using the plain rule (2.25). The resulting "correction matrix" blocks are assembled and stored as a $2N$-by-$2N$ sparse matrix. The entire application of $\boldsymbol{A}$ to the density vector is then performed using the FMM with the plain rule (2.25), plus the action of this sparse matrix to replace the "close" interactions with their accurate values. This matrix-vector multiplication is used to solve the whole linear system iteratively via GMRES. We do this for our large-scale examples in Section 2.5.

2.3.3 Close-evaluation of potentials

Since we will perform all Nyström matrix filling using the same formulae as for close-evaluation of potentials (on- or off-surface), we now review formulae for the close-evaluation task. As before, we consider a single target point $\boldsymbol{r}' \in \Omega$ which is "close" to the single source panel $\Lambda = Z([a,b])$, on which a density $\boldsymbol{\sigma}_j$ is known at the nodes $j = 1, \ldots, p$. Recall that the p-point Gauss–Legendre nodes and weights for the parameter interval $[a,b]$ are t_j and w_j.

We make use of special panel quadratures developed for the Laplace double-layer case by Helsing–Ojala [80, Sec. 5], for the logarithmic and hypersingular kernels by Helsing [77], and its simple generalization to supersingular (one of those used by Helsing–Jiang [79, Sec. 6]). They use high-order polynomial interpolation in the complex plane to approximate the density function, then apply a recurrence to exactly evaluate the near-singular integral for each monomial basis function. In Section 2.2 we showed that all the needed Stokes potentials may be written in terms of $I_L[\tau](z')$, $I_C[\tau](z')$, $I_H[\tau](z')$ and $I_S[\tau](z')$ from (2.7)–(2.10), involving scalar Cauchy densities τ derived from the given Stokes density $\boldsymbol{\sigma}$. Thus in the following subsections we review close-evaluation of each contour integral in turn. We also describe in more detail than in the original literature a recipe to handle branch cuts.

It turns out that monomial approximation is most stable when assuming that $Z(a) = 1$ and $Z(b) = -1$, i.e. the panel endpoints are ± 1 in the complex plane [80]. Thus we start by making this assumption, then in Sec. 2.3.3 review how to correct the results for a panel with general endpoints.

Recall that $\boldsymbol{\sigma}$, hence the derived scalar functions τ needed in the contour integrands, is available only at the p nodes of Λ. In order to improve the accuracy of the complex approximation for curved panels, firstly an upsampling is performed to $m > p$ "fine" nodes, using Lagrange interpolation in the parameter $t \in [a,b]$ from the p nodes to the m fine nodes. We find that $m = 2p$ is beneficial without incurring significant extra cost. Let $\tilde{\tau}_j$, $j = 1, \ldots, m$ denote the fine density values, $\tilde{\boldsymbol{r}}_j = Z(\tilde{t}_j) \in \Lambda$ be the fine nodes, and $\tilde{z}_j$ be complex representation of $\tilde{\boldsymbol{r}}_j$, where $\tilde{t}_j$ and $\tilde{w}_j$ are the m-point Gauss–Legendre nodes and weights respectively for $[a, b]$. The following schemes now will use only the fine values and nodes.

Close evaluation of the Cauchy potential

This method first appeared in [80, Sec. 4–5]. One approximates τ on the panel Λ in the complex variable by the degree $m-1$ polynomial

$$\tau(z) \approx \sum_{k=1}^{m} a_k z^{k-1}, \qquad z \in \Lambda \ . \tag{2.28}$$

The vector of complex coefficients $\mathbf{a} := \{a_k\}_{k=1}^{m}$ is conveniently found by using a dense direct solve of the square Vandermonde system

$$V\mathbf{a} = \tilde{\boldsymbol{\tau}} \ , \tag{2.29}$$

with elements $V_{jk} = \tilde{z}_j^{k-1}$, $j, k = 1, \ldots, m$, and right hand side $\tilde{\boldsymbol{\tau}} := \{\tilde{\tau}_j\}_{j=1}^{m}$. It is known that, for any arrangement of nodes $\tilde{z}_j$ other than those very close to the roots of unity, the condition number of V grows exponentially with m [145]. However, as discussed in [80, App. A], at least for $m < 50$, despite the extreme ill-conditioning, *backward stability* of the linear solver insures that the resulting polynomial matches the values at the nodes to close to machine precision. For this we use MATLAB `mldivide` which employs standard partially-pivoted Gaussian elimination.

The remaining step is to compute the contour integral of each monomial,

$$p_k = p_k(z') := \int_{-1}^{1} \frac{z^{k-1}}{z - z'} \mathrm{d}z, \quad k = 1, \ldots, m \ , \tag{2.30}$$

which are, recalling (2.8), then combined using (2.28) to get

$$I_C[\tau](z') \ \approx \ \sum_{k=1}^{m} a_k p_k \ . \tag{2.31}$$

By design, since the monomials are with respect to z in the complex plane (as opposed to, say, the parameter t), Cauchy's theorem implies that each p_k is independent of the curve Λ and depends only on the end-points. Specifically, for $k = 1$ we may integrate analytically by deforming Λ to be the curve connecting -1 and 1, so

$$p_1 := \int_{-1}^{1} \frac{1}{z - z'} \mathrm{d}z = \log(1 - z') - \log(-1 - z') \pm 2\pi i \mathcal{N}_{z'} \tag{2.32}$$

where $\mathcal{N}_{z'} \in \mathbb{Z}$ is an integer winding number that depends on the choice of branch cut of the log function. For the standard cut on the negative real axis then $\mathcal{N}_{z'} = 0$ when x is outside the domain enclosed by the oriented curve composed of Λ traversed forwards plus $[-1, 1]$ traversed backwards, $\mathcal{N}_{z'} = +1$ (-1) when z' is inside a region enclosed counterclockwise (clockwise) [80]. However, since it is inconvenient and error-prone to decide $\mathcal{N}_x$ for points on or very close to Λ and $[-1, 1]$, we prefer to combine the two logs then effectively rotate its branch cut by a phase $\phi \in \mathbb{R}$, by using

$$p_1 = i\phi + \log \frac{1 - z'}{e^{i\phi}(-1 - z')} , \tag{2.33}$$

where $\phi = -\pi/4$ when the upwards normal of the panel points into Ω (as for an interior curve), or $\phi = \pi/4$ otherwise. This has the effect of pushing the branch cut "behind" the panel (away from Ω; see Fig. 2.3), with the cut meeting ± 1 at an angle ϕ from the real axis. The potential is correct in the closure of Ω, including on the panel itself, without any topological tests needed (hence the unified handling of close and self evaluations in Sec. 2.3.2). This can fail if a panel is very curved (such a panel would be inaccurate anyway), or if a piece of Ω approaches close to the back side of the panel (which can be prevented by adaptive refinement as in Section 2.4). In practice we find that this is robust. However, we will mention one special situation where (2.33) could fail and therefore careful adjustment of the branch cut is critical; see Remark 2.3.3 below.

The following 2-term recurrence is easy to check by adding and subtracting $z'z^{k-1}$ from the numerator of the formula (2.30) for p_{k+1}:

$$p_{k+1} = z' p_k + (1 - (-1)^k)/k \ . \tag{2.34}$$

For $|z'| < 1.1$ we find that the recurrence is sufficiently stable to use upwards from the value p_1 computed by (2.33), to get $p_2, \ldots, p_m$. However, for targets outside this close disc, especially for larger m, the upwards direction is unstable. Thus, here instead we use numerical quadrature on (2.30) to get

$$p_m \approx \sum_{j=1}^{m} \frac{z_j^{m-1}}{z_j - z'} \frac{\mathrm{d}}{\mathrm{d}t} Z(\tilde{t}_j) \tilde{w}_j \ , \tag{2.35}$$

then apply (2.34) downwards to compute $p_{m-1}, \ldots, p_2$, and as before use p_1 from (2.33). Outside of the disc, no branch cut issues arise.

Remark 2.3.3 (branch cuts at corners)*.* When a panel Λ is directly touching a corner, directly applying (2.33) can fail no matter how much the panels are refined. In Figure 2.3a, the panel on the opposite side of the corner, Λ', is always behind Λ and lying across the branch cut associated to Λ. Consequently, the close evaluation from Λ to Λ' results in completely wrong values, also leading to ill-conditioning of the whole system (2.24). Instead, one can simply change the sign of ϕ in (2.33) to flip the branch cut to accommodate the targets on Λ' (Figure 2.3b). In practice, we find that this is robust for corners of arbitrary angles.

Close evaluation of the logarithmic potential

This method first appeared in [77, Sec. 2.3]. For τ real, we can write the final form in (2.7) as

$$I_L[\tau](z') \ = \ \mathrm{Re} \int_{\Lambda} \log(z - z') \frac{\tau(z)}{i n_z} \mathrm{d}z, \tag{2.36}$$

which shows that the quantity to approximate on Λ as a complex polynomial is $\tau(z)/i n_z$. Thus we find the coefficients in

$$\frac{\tau(z)}{i n_z} \ \approx \ \sum_{k=1}^{m} b_k z^{k-1}, \qquad z \in \Lambda \ , \tag{2.37}$$

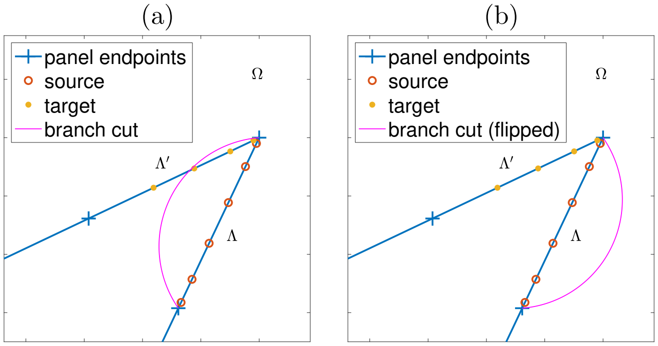

Figure 2.3: Special handling of close evaluation branch cut when the panel is touching a reentrant corner. (a) The target panel Λ' is crossing the branch cut of the source panel Λ defined by (2.33), resulting in wrong close evaluation values. (b) Changing the sign of ϕ in (2.33) flips the branch cut to the other side; close evaluation at the targets are now correct.

by solving (2.29) but with modified right hand side $\{\tilde{\tau}_j / in_{\tilde{z}_j}\}_{j=1}^{m}$. Defining

$$q_k \ := \int_{-1}^{1} z^{k-1} \log(z-z')\mathrm{d}z, \quad k=1,\dots,m \tag{2.38}$$

then (2.37) gives

$$\int_{-1}^{1} \log(z-z') \frac{\tau(z)}{in_z}\mathrm{d}z \ \approx \ \sum_{k=1}^{m} b_k q_k \ , \tag{2.39}$$

whose real part is $I_L[\tau](z')$. Each q_k is computed from p_k of (2.30) as evaluated in Sec. 2.3.3, via a formula easily proven by integration by parts, $q_k = \frac{-p_{k+1}+\log(1-z')-(-1)^k\log(1+z')}{k}$, therefore

$$q_k = \begin{cases} \left(-p_{k+1} + i\phi + \log\frac{1-z'}{e^{i\phi}(-1-z')}\right)/k \ , & k \text{ even} \ , \\ \left(-p_{k+1} + \log[(1-z')(-1-z')]\right)/k \ , & k \text{ odd} \ , \end{cases} \tag{2.40}$$

where the latter form is that used in the code, needed to match the branch cut rotation used for p_k in (2.33). The final evaluation of I_L is then via

$$I_L[\tau](z') \ \approx \ \mathrm{Re}\sum_{k=1}^{m} b_k q_k \ . \tag{2.41}$$

Close evaluation of the Hadamard and supersingular potentials

We apply formulae from [77, Sec. 2.2] (Hadamard) and [79, Sec. 6] (supersingular). The double-layer Stokes velocity requires gradients of Laplace potentials (2.15), which require

$I_H[\tau](z')$. Also, the traction of the Stokes single-layer (A.1.5) involves I_H applied to $\tau(z)/in_z$, and the traction of the Stokes double-layer further involves I_S (Appendix A.1).

Using the complex monomial expansion (2.28), we have

$$I_H[\tau](z') \approx \sum_{k=1}^{m} a_k r_k, \qquad I_S[\tau](z') \approx \sum_{k=1}^{m} a_k s_k, \tag{2.42}$$

where

$$r_k := \int_{-1}^{1} \frac{z^{k-1}}{(z-z')^2}\mathrm{d}z, \qquad s_k := \int_{-1}^{1} \frac{z^{k-1}}{(z-z')^3}\mathrm{d}z, \qquad k = 1,\ldots,m\ . \tag{2.43}$$

The following formulae can be shown by integration by parts, and enable r_k and s_k to be found,

$$r_k = (k-1)p_{k-1} + \frac{(-1)^{k-1}}{-1-z'} - \frac{1}{1-z'}\ , \qquad s_k = \frac{k-1}{2} r_{k-1} + \frac{(-1)^{k-1}}{2(-1-z')^2} - \frac{1}{2(1-z')^2}\ , \tag{2.44}$$

using p_k from (2.30) as computed in Sec. 2.3.3, and $p_0 = 0$.

Review of transforming for general panel endpoints

To apply close-evaluation methods in the above three sections to a general panel $\Lambda = Z([a,b])$, define the complex scale factor $\alpha_0 := (Z(b) - Z(a))/2$ and origin $z_0 = (Z(b) + Z(a))/2$. Then the affine map

$$z = s(\tilde{z}) := \frac{\tilde{z} - z_0}{\alpha_0}$$

takes $\tilde{z}$ to its scaled version z. Likewise, the factor $\frac{\mathrm{d}}{\mathrm{d}t}Z(\tilde{t}_j)$ in (2.35) is replaced by $\frac{\mathrm{d}}{\mathrm{d}t}Z(\tilde{t}_j)/\alpha_0$. Following Sec. 2.3.3 using these scaled target and fine nodes, no change in the result I_C is needed. However, the value of I_H computed in Sec. 2.3.3 must afterwards be divided by α_0, and the value of I_S divided by α_0^2. The value of I_L computed in Sec. 2.3.3 must be multiplied by $|\alpha_0|$, and then have $|\frac{\mathrm{d}}{\mathrm{d}t}Z(\tilde{t}_j)w_j/\alpha_0| \log|\alpha_0|$ subtracted.

2.3.4 Computation of close-evaluation matrix blocks

The above described how to evaluate $I_L[\tau](z')$, $I_C[\tau](z')$, $I_H[\tau](z')$ and $I_S[\tau](z')$ given known samples $\tilde{\tau}_j$ at a panel's fine nodes. In practice it is useful to instead precompute a matrix block A which takes any density values at a panel's original p nodes y_j to a set of n target values of the contour integral. We review a result contained in [80, Eq. (51)] (where a faster numerical method was also given in the Cauchy case). Consider the case of the Cauchy kernel, and let A denote this n-by-p matrix. Let L be the m-by-p Lagrange interpolation matrix from the nodes t_j to fine nodes $\tilde{t}_j$, which need be filled once and for all. Let P be the n-by-m matrix with entries $P_{ik} = p_k(z_i')$, given by (2.30), where $\{z_i'\}_{i=1}^n$ is the set of desired targets. In exact arithmetic one has

$$A = PV^{-1}L \ .$$

However, since V is very ill-conditioned, filling V^{-1} and using it to multiply to the right is numerically unstable. Instead an *adjoint method* is used: one first solves the matrix equation $V^\top X = P^\top$, where $\top$ indicates non-conjugate transpose, then forms the product

$$A = X^\top L \ .$$

The matrix solve is done in a backward stable fashion via MATLAB's `mldivide`; see [80, App. A]. A further advantage of the adjoint approach is that if n is small, the solve is faster than computing V^{-1}.

The formulae for the logarithmic, Hadamard and supersingular kernels are analogous.

2.4 Adaptive panel refinement

In order to solve a BIE to high accuracy, it is necessary to set up panels such that the given complex geometry is correctly resolved. In this section, we describe a procedure that adaptively refine the panels based solely on the geometric properties. Specifically, our refinement algorithms take into account the accuracy of geometric representations (including arc length and curvature), the location of corners, and the distance between boundary components. It necessarily has some ad-hoc aspects, yet we find it quite robust in practice.

Suppose that for a user-prescribed tolerance ϵ, the goal is to find a partition $\Gamma = \bigcup_{i=1}^{n_\Lambda} \Lambda_i$ such that the error, ε, of evaluating boundary integral operators such as (2.22) satisfies $\varepsilon \lessapprox \epsilon$. To this end, we describe our adaptive refinement scheme which proceeds with three stages. In what follows, we again assume that the panel under consideration $\Lambda = Z([a, b])$ is rescaled such that its two endpoints are ± 1.

Stage 1: Choice of overall p. Given tolerance ϵ, the goal is to determine a number of quadrature points, p, applied to all panels, such that the relative quadrature error on any panel is $O(\epsilon)$. As mentioned above, the p-point Gauss–Legendre quadrature on $[-1, 1]$ has $O(\varrho^{-2p})$ error if the integrand can be analytically extended to a Bernstein ellipse of parameter $\varrho > 1$, where the semi-major axis of this ellipse is $(\varrho + \varrho^{-1})/2$ [191, Thm. 19.3]. Therefore, making $\varrho^{-2p} \leq \epsilon$ we obtain the first term of the right-hand side in

$$p \geq \left\lceil \frac{\log_{10}(1/\epsilon)}{2 \log_{10} \varrho} \right\rceil + c, \tag{2.45}$$

where the second term accounts for unknown prefactors. Empirically we set $c = 1$.

To determine ϱ, we require that the Bernstein ϱ-ellipse of each panel encloses both its immediate neighboring panels. This insures that, when applying the smooth quadrature rule (2.23) between the nearest non-neighboring ("far") panels that do not touch a corner, the integrand continues to a function analytic inside the ϱ-ellipse, so, by the above discussion, the relative error is no worse than ϵ. (This will not apply to panels touching a corner, but they are small enough to have negligible contributions.) Stages 2–3 below will place an upper bound of λ on the ratio of the lengths (with respect to parameter) of neighboring panels. Combining these two relations gives

$$\frac{\varrho + \varrho^{-1}}{2} = 1 + \frac{2}{\lambda}. \tag{2.46}$$

One then solves (2.46) for ϱ and substitutes it into (2.45) to obtain a lower bound for p.

For example, $\lambda \leq 3$ holds in our examples, so $\varrho = 3$, and therefore we have as sufficient the simple rule $p = \lceil \log_{10}(1/\epsilon) + 1 \rceil$.

Stage 2: Local geometric refinement. In this stage, panels are split based on local geometric properties:

1. *Corner refinement.* Panels near a corner are refined geometrically so that each panel is a factor λ shorter in parameter than its neighbor (see lines 8–11 of Algorithm 1).

   To each corner is associated a factor $\lambda \geq 2$. A rule of thumb is to use $\lambda = 2$ for sharper corners (e.g. whose angle θ is close to 0 or 2π) which are harder to resolve, and use $\lambda > 2$ for "flatter" corners (e.g. θ closer to π) to reduce the number of panels without affecting the overall achieved accuracy. In practice, we use $\lambda = 3$ for corners $\pi/2 \leq \theta \leq 3\pi/2$; for a problem with many flat corners, this can reduce the total number of unknowns by a factor of about $2/3$ (or about $\log_3 2$).

   Near a corner, refinement stops when the panels touching the corner are shorter than ϵ^α, where α is an empirical power parameter chosen for each corner. Recent theoretical results for the plain double-layer formulation for the Stokes Dirichlet BVP state that the density is a constant plus a bounded singular function with positive power for any corner angle in $(0, 2\pi)$ [156]. For our $\mathbf{D} + \mathbf{S}$ formulation we observe a density behavior consistent with this. This suggests choosing $\alpha = 1$ for any corner angle, since the constant term will dominate the error. In fact, for small (non-reentrant) angles we are able to reduce α somewhat without loss of accuracy, hence do so, to reduce the number of panels.

2. *Curved panel refinement.* Panels away from any corners are refined based on how well the smooth geometric properties are represented by the interpolants on their p Legendre nodes. We measure the accuracy of geometric representations by the interpolation errors

of a set of test functions on a set of test points. First, we define the set of test functions $G = \{g_1, g_2, g_3, \ldots\}$ to be approximated on the panel Λ. The following list of functions are included in G whenever the necessary derivatives are available:

- $g_1(t) = Z(t)$, which resolves the geometry representation.
- $g_2(t) = |\frac{\mathrm{d}}{\mathrm{d}t} Z(t)|$, which resolves arc length, recalling that arc length is

$$S = \int_\Lambda \mathrm{d}s = \int_a^b \left| \frac{\mathrm{d}}{\mathrm{d}t} Z(t) \right| \mathrm{d}t$$

- $g_3(t) = \frac{\left|\mathrm{Im}\,(\frac{\mathrm{d}^2}{\mathrm{d}t^2} Z(t) / \frac{\mathrm{d}}{\mathrm{d}t} Z(t))\right|^2}{\left|\frac{\mathrm{d}}{\mathrm{d}t} Z(t)\right|}$, which resolves bending, since bending energy is

$$E = \int_\Lambda \kappa^2 \mathrm{d}s = \int_a^b \left| \frac{\frac{\mathrm{d}}{\mathrm{d}t} Z(t) \times \frac{\mathrm{d}^2}{\mathrm{d}t^2} Z(t)}{\left|\frac{\mathrm{d}}{\mathrm{d}t} Z(t)\right|^3} \right|^2 \left| \frac{\mathrm{d}}{\mathrm{d}t} Z(t) \right| \mathrm{d}t = \int_a^b \frac{\left|\mathrm{Im}\,(\frac{\mathrm{d}^2}{\mathrm{d}t^2} Z(t) / \frac{\mathrm{d}}{\mathrm{d}t} Z(t))\right|^2}{\left|\frac{\mathrm{d}}{\mathrm{d}t} Z(t)\right|} \mathrm{d}t$$

Next, we define the test points to be the m equally spaced points on $[a, b]$, denoted $\tilde{t}_j^\Lambda, j = 1, \ldots, m$, and let $t_j^\Lambda, j = 1, \ldots, p$ be the Legendre nodes. Then for each $i = 1, 2, 3$, the relative error of approximating g_i is

$$\varepsilon_i = \frac{\|\tilde{\mathbf{g}}_i - \mathbf{M} \cdot \mathbf{g}_i\|}{\|\mathbf{g}_i\|},$$

where $\tilde{\mathbf{g}}_i := (g_i(\tilde{t}_1^\Lambda), \ldots, g_i(\tilde{t}_m^\Lambda))$, $\mathbf{g}_i := (g_i(t_1^\Lambda), \ldots, g_i(t_p^\Lambda))$, and $\mathbf{M}$ is the $m \times p$ interpolation matrix from the Legendre nodes to the test points. The panels are refined until $\max_i \varepsilon_i < \epsilon^\beta$, where $\beta > 0$ is another power parameter, with default value $\beta = 1$.

The corner and curved panel refinement rules are applied to all panels recursively. The complete procedure is summarized in Algorithm 1.

Stage 3: Global closeness refinement. At this final stage, panels are further refined if they are (relatively) too close to any non-neighboring panels. Specifically, let Λ^{left} and Λ^{right} be the two immediate neighboring panels of Λ, and define $\Gamma^{\text{far}} := \Gamma \setminus (\Lambda^{\text{left}} \cup \Lambda \cup \Lambda^{\text{right}})$ as all non-neighboring panels of Λ. Then the panel Λ is refined if $d(\Lambda, \Gamma^{\text{far}})$, its distance

Algorithm 1 Local geometric refinement

```
Require: The current panel $\Lambda = Z([a,b])$; tolerance $\epsilon$; corner information $C = \{t^c_j, \lambda_j, \alpha_j\}_{j=1}^k$; test function(s) $G = \{g_1, g_2, g_3, \ldots\}$; $\beta$ is the tolerance exponent for the test functions (default value 1).
function REFINE($Z([a,b]), \epsilon, C, G, \beta$)
    Panel parametric length $L = b - a$
    Panel arc length $S = \int_a^b |\frac{\mathrm{d}}{\mathrm{d}t} Z|$
    if geometry has corners then
        Let $t^c_i$ be the corner closest to $[a,b]$
        if $L < \epsilon^{\alpha_i}$ or $S < \epsilon^{\alpha_i}$ then
            return $\{a, b\}$                              ▷ panel length reached lower limit, do not split
        if $t^c_i$ is close enough to the panel $[a,b]$ then
            if $a < t^c_i < b$ then $s = t^c_i$                          ▷ split right at the corner
            else if $t^c_i < a$ then $s = a + L/\lambda_i$                ▷ split towards the corner
            else if $t^c_i > b$ then $s = b - L/\lambda_i$                ▷ split towards the corner
    if split point $s$ is not defined then
        $\mathbf{g}_i = g_i$ values at quadrature points
        $\tilde{\mathbf{g}}_i = g_i$ values at test points
        $\mathbf{M}$ = interpolation matrix from quadrature points to test points
        $\varepsilon = \max_{g_i \in G} \|\tilde{\mathbf{g}}_i - \mathbf{M} \cdot \mathbf{g}_i\| / \|\mathbf{g}_i\|$   ▷ interpolation error of test function(s)
        if $\varepsilon > \epsilon^\beta$ then
            $s = (a+b)/2$                                              ▷ split in half
    if split point $s$ is defined then
        $\mathbf{t}^1$ = REFINE($Z([a,s]), \epsilon, C, G, \beta$)
        $\mathbf{t}^2$ = REFINE($Z([s,b]), \epsilon, C, G, \beta$)
        return $\mathbf{t}^1 \cup \mathbf{t}^2$                       ▷ recursively refine panel
    return $\{a, b\}$                                                  ▷ do not split
```

from Γ^{far}, is shorter than its arc length by a factor of 3 (see Lines 4–10 of Algorithm 2); see Remark 2.5.1 for an alternative, less restrictive, refinement criterion.

In practice, the distance $d(\Lambda, \Gamma^{\text{far}})$ can be approximated by $\min_{i,j} |\boldsymbol{r}_i - \boldsymbol{r}_j|$, where the minimum is searched among all pairs of nodes $\boldsymbol{r}_i \in \Lambda$ and $\boldsymbol{r}_j \in \Gamma^{\text{far}}$. A kd-tree algorithm [184] is used to accelerate this process for our large examples in Section 2.5, in which case the elliptical close neighborhood (2.27) is also replaced by $\bigcup_{i=1}^p B(Z_i, C\,S)$, the union of disks around each node on Λ, for convenience.

The above refinement process is applied to each panel from the output of the previous stage and repeats until no further splitting. The algorithm for this stage is summarized in Algorithm 2. We note that since our algorithm is panel-based, it is agnostic as to whether two touching panels belong to the same boundary component or not. Hence this algorithm

Algorithm 2 Global closeness refinement

```
Require: The refined panels $\Gamma = \bigcup_k \Lambda_k$ from Stage 2 (local geometric refinement).
function CloseRefine($\Gamma$)
    Initialize the output set $\mathbf{t} = \emptyset$, which will contain the final panel endpoints
    Initialize the set of new endpoints $\mathbf{t}^{\mathrm{new}} = \{$endpoints of $\Gamma = \bigcup_k \Lambda_k\}$
    while $\mathbf{t}^{\mathrm{new}} \neq \emptyset$ do                          ▷ repeat until no further splitting
        $\mathbf{t} = \mathbf{t} \cup \mathbf{t}^{\mathrm{new}}$
        $\mathbf{t}^{\mathrm{new}} = \emptyset$
        Update panels $\Gamma = \bigcup_k \Lambda_k$ based on $\mathbf{t}$          ▷ ready for a new round of refinement
        for each panel $Z([a,b]) \subset \Gamma$ do
            Locate $\Lambda^{\mathrm{left}}$ and $\Lambda^{\mathrm{right}}$, the two immediate neighboring panels of $Z([a,b])$
            Define $\Gamma^{\mathrm{far}} = \Gamma \setminus (\Lambda^{\mathrm{left}} \cup Z([a,b]) \cup \Lambda^{\mathrm{right}})$
            Compute the distance $d = d(Z([a,b]), \Gamma^{\mathrm{far}})$
            Calculate $S =$ arc length of $Z([a,b])$
            if $\frac{1}{3} S > d$ then
                $\mathbf{t}^{\mathrm{new}} = \mathbf{t}^{\mathrm{new}} \cup \{\frac{a+b}{2}\}$                  ▷ split in half
    return final panel endpoints $\mathbf{t}$
```

handles two situations simultaneously: the case of close-touching between different boundary components, as well as the case of "self-touching" where a boundary component is almost touching itself.

Remark 2.4.1 (Expected convergence rate with corners)*.* The above three stages involve two quantities—the panel order p, and a resulting number of panels per corner—both of which grow linearly with $\log 1/\epsilon$. However, N is the product of these two quantities, thus, in the presence of corners, one expects asymptotically $N = O(\log^2 1/\epsilon)$ as $\epsilon \to 0$. In other words, the error converges *root exponentially* in N, i.e. as $O(e^{-c\sqrt{N}})$. This matches the theoretical convergence rate for *hp*-BEM on polygons by Heuer–Stephan [85]. This rate has also recently been observed and proven for a geometrically graded "method of fundamental solutions" approach to polygons by Gopal–Trefethen [57].

2.5 Numerical results and discussion

In what follows, numerical examples will be presented to test the overall solution scheme presented so far. In each example, we solve a Dirichlet problem in the domain exterior to the given geometries. The BIE formulation of the problem on Γ is $(\frac{1}{2} + \mathbb{S} + \mathbb{D})[\boldsymbol{\sigma}] = \boldsymbol{g}$, as described in Section 2.2.1.

The solution procedure is to first adaptively refine the representation of the geometry using our refinement scheme (Section 2.4), then the BIE is discretized using the plain and special quadratures (Section 2.3) and solved for the density $\boldsymbol{\sigma}$, and finally the solution $\boldsymbol{u} = \boldsymbol{u}_\infty + (\mathbf{S} + \mathbf{D})[\boldsymbol{\sigma}]$ is evaluated everywhere in the exterior domain (again making use of special quadratures). In our examples the evaluation grid spacing will be $\Delta x = 0.02$.

We mention that our solution scheme has been tested on boundary value problems with boundary data extracted from an analytically known smooth flow $\boldsymbol{u}$, and, as expected, achieves superalgebraic convergence. However, in the presence of corners, such smooth test problems do *not* involve the corner singularities that generically arise in physical problems. For this reason, we only present results on physical flows such as imposed uniform or linear shear flows. In all the examples, the exact solution is not known analytically; therefore, we use the finest grid solution as the reference solution.

Example 1. Smooth domain. This example tests our scheme on a linear shear flow around a smooth starfish-shaped island defined by function $Z(t) = (1 + 0.3\cos 5t)\,[\cos t, \sin t]^T$, with no-slip boundary condition $\boldsymbol{u}|_\Gamma = 0$ and $\boldsymbol{u}_\infty(x, y) = [-y, 0]^T$ as $|\boldsymbol{r}| \to \infty$. We chose $\beta = 0.8$ (Line 17 of Algorithm 1), in order to reduce N. In addition to the velocity field, we have also investigated the convergence of the pressure field and the traction field in the $(1, 2)$-direction, both of which are obtained using our close evaluation scheme (Section 2.3.3 and Appendix A.1). Our scheme achieved accuracies that match the requested tolerance (Figure 2.4b). All the solution fields converge super-algebraically with respect to the problem size (Figure 2.4c).

Example 2. Domain with corners. The smooth geometry in Example 1 is now replaced with a non-convex polygon. Figure 2.5 shows a linear shear flow around a “shuriken” domain

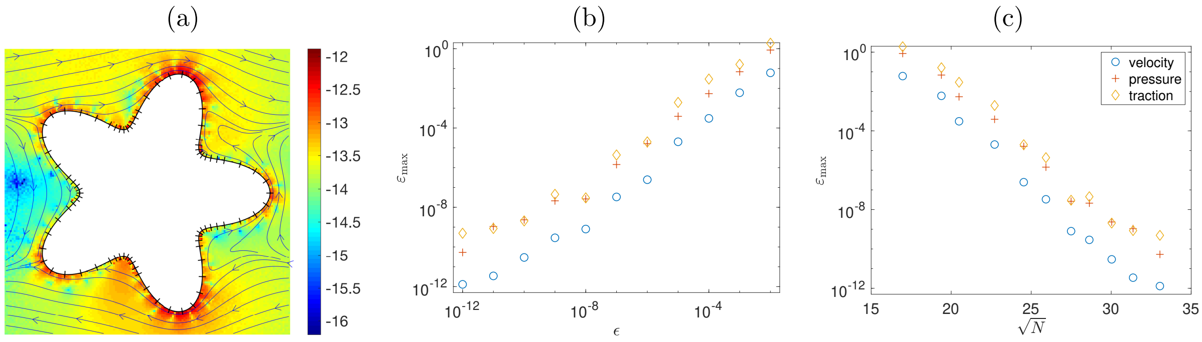

Figure 2.4: (a) Linear shear flow past a starfish-shaped island. Streamlines of the flow and panel endpoints (small segments) are shown. Color represents estimated $\log_{10}$ error of the velocity computed under tolerance $\epsilon = 10^{-12}$ (resulted in $2N = 2184$ degrees of freedom). (b) Convergence of the maximum relative error $\varepsilon_{\max}$ versus requested tolerance ϵ, where the traction field is computed along the $(1, 2)$-direction. (c) Convergence of the maximum error $\varepsilon_{\max}$ versus the square root of the number of nodes.

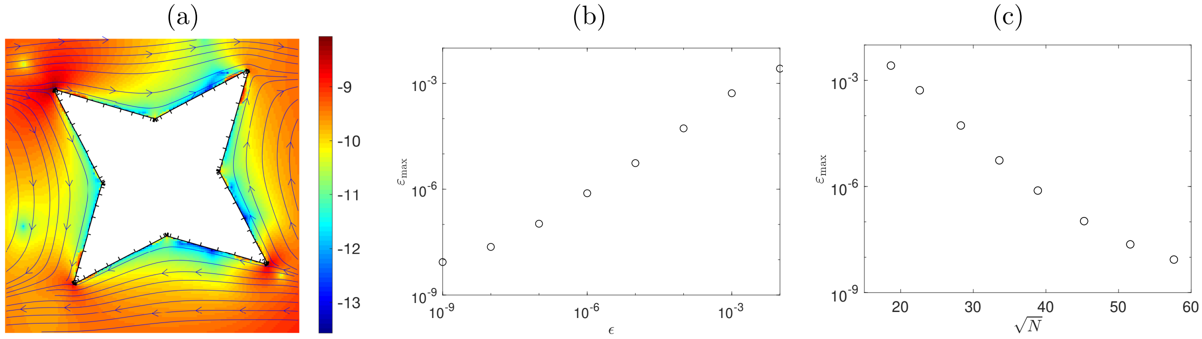

Figure 2.5: (a) Linear shear flow around a shuriken-shaped island with 8 corners. Streamlines of the flow and panel endpoints (small segments) are shown. Color represents the $\log_{10}$ error of the velocity computed under tolerance $\epsilon = 10^{-10}$ (resulted in $2N = 6640$ degrees of freedom). (b) Convergence of the maximum error $\varepsilon_{\max}$ versus requested tolerance ϵ. (c) Convergence of the maximum error $\varepsilon_{\max}$ versus the square root of the number of nodes; root-exponential convergence would result in a straight line.

with eight corners, the outer four of which are reentrant (with respect to Ω) corners of angle $\theta = 1.74\pi$. With $\alpha = 0.5$ for the flatter corners and $\alpha = 1.1$ for the sharper ones, our scheme achieved accuracies that match the requested tolerance (Figure 2.5b). Note that the convergence with respect to problem size is super-algebraic (Figure 2.5c), and consistent with root-exponential convergence, as expected for problems with corner singularities (see Remark 2.4.1). We used the numerical solution obtained using $\epsilon = 10^{-10}$ as the reference solution.

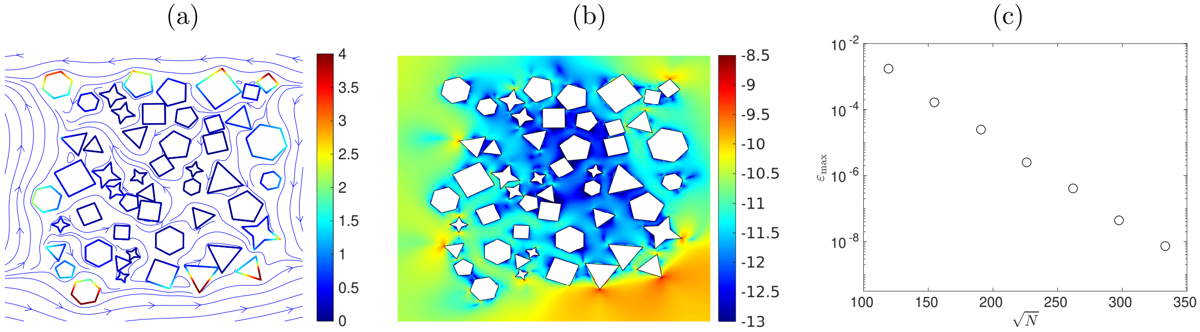

Figure 2.6: (a) Streamlines of a shear flow past 50 randomly generated polygonal islands with a total number of 253 corners. Color on the polygon boundaries indicate the magnitude of density σ. (b) $\log_{10}$ of absolute error of the velocity, computed using $2N = 222140$ degrees of freedom. Error is measured on a 1000×1000 grid (spacing $\Delta x \approx 8.3 \times 10^{-3}$) by comparing to the solution obtained with $\epsilon = 10^{-10}$. (c) Convergence of the maximum error $\varepsilon_{\max}$ versus the square root of the number of nodes.

Example 3. Multiple polygonal islands. This example models a porous media flow through a collection of non-smooth, non-convex and closely packed boundaries: we set up 50 polygonal islands with a total number of 253 corners (Figure 2.6). The computational domain has width ≈ 8 and the closest distance between the polygons is about 10^{-2}. The background flow is the same as in the previous examples. With $\alpha = 0.75$ and $\lambda = 2$ for all corners, the convergence (Figure 2.6c) is similar to the single-polygon island example (Figure 2.5b–c), achieving more than 8 digits using approximately 800 degrees of freedom per corner. This demonstrates the robustness of our adaptive scheme, that is, the performance is as good for a more complex example as for a simple one.

Example 4. Artificial vascular network. We now turn to the example shown in Figure 2.2. We construct an artificial vascular network (with 378 corners) that mimics those observed in an eye of a zebra fish [6]. The flow in this network is driven by a uniform influx from the circular wall at the center and a uniform outflux at the outer circular wall, such that the overall volume is conserved; all other boundaries have a no-slip condition. We solve the BIE for this problem using GMRES with a block diagonal preconditioner consisting of the diagonal panel-wise blocks of the BIE system itself (i.e., the self-evaluation blocks for each panel). The FMM is used for applying the matrix and for final flow evaluations; see Remark 2.3.1. The sparse correction matrix (see Section 2.3.2) is applied via MATLAB's single-threaded built-in matrix-vector multiplication; its rows have been precomputed as described in Section 2.3.4. All computations are done on an 8-core 4.0 GHz Intel Core i7 desktop.

For various tolerances ϵ, Table 2.1 shows: the relative L_2-error ε_{L_2}, the relative maximum error in velocity $\varepsilon_{\max}$, the total number of panels used n_Λ, the number of degrees of freedom $2N$, memory (RAM) usage, the number of GMRES iterations and time used, the setup time for precomputing the close-correction matrices, and the percentage time for applying Stokes FMM during GMRES. Several observations are in order:

1) Both ε_{L_2} and $\varepsilon_{\max}$ decay super-algebraically with the number of degrees of freedom; this data is plotted in Figure 2.7a. The closeness between ε_{L_2} and $\varepsilon_{\max}$ shows that our scheme has achieved high accuracies near the sharper reentrant corners (hard) that are similar to those near the smooth edges (easy). This error analysis remains valid even in the zoomed-in high-resolution error plots in Figure 2.2. Furthermore, the convergence performance of this example is the same as the previous two examples—we achieved more than 8 digits, with a ratio $\frac{\text{degrees of freedom}}{\#\text{corners}} \approx 943$, which is similar to the ratios in Example 2 (830) and Example 3 (878). This once again demonstrates the robustness of our overall scheme to problem complexity.

2) The number of GMRES iterations increases only because we have requested smaller tolerance: each additional digit needs about 100 more iterations. The GMRES convergence *rate* is stable, which demonstrates that our second kind BIE formulation is well-conditioned even in the presence of corners.

3) The fact that the Stokes FMM time is the main cost shows that our algorithm has achieved

ϵ	ε_{L_2}	$\varepsilon_{\max}$	n_Λ	$2N$	RAM (gb)	GMRES iteration	GMRES time(s)	setup time(s)	% FMM
1e-03	4.34e-04	5.43e-03	6549	52392	2.3	796	248	48	78.50
1e-04	2.20e-05	4.58e-04	8281	82810	2.9	919	458	63	77.86
1e-05	3.55e-06	7.21e-05	10301	123612	3.7	1091	759	84	75.92
1e-06	1.26e-06	7.15e-06	12061	168854	5.0	1282	1197	106	72.69
1e-07	2.53e-07	1.51e-06	14079	225264	6.9	1390	1670	135	70.91
1e-08	6.01e-09	1.58e-07	15839	285102	9.0	1501	2433	164	68.77
1e-09	1.44e-09	5.34e-08	17829	356580	12	1597	3195	204	66.16

Table 2.1: Results and statistics of solving the BVP in the vascular network in Figure 2.2 for various tolerance ϵ. Errors $\varepsilon_{\max}$ and ε_{L_2} are measured on a 2160×2160 grid (spacing $\approx 2.5 \times 10^{-3}$) by comparing to the solution obtained at $\epsilon = 10^{-10}$. CPU time and RAM used are measured using [12].

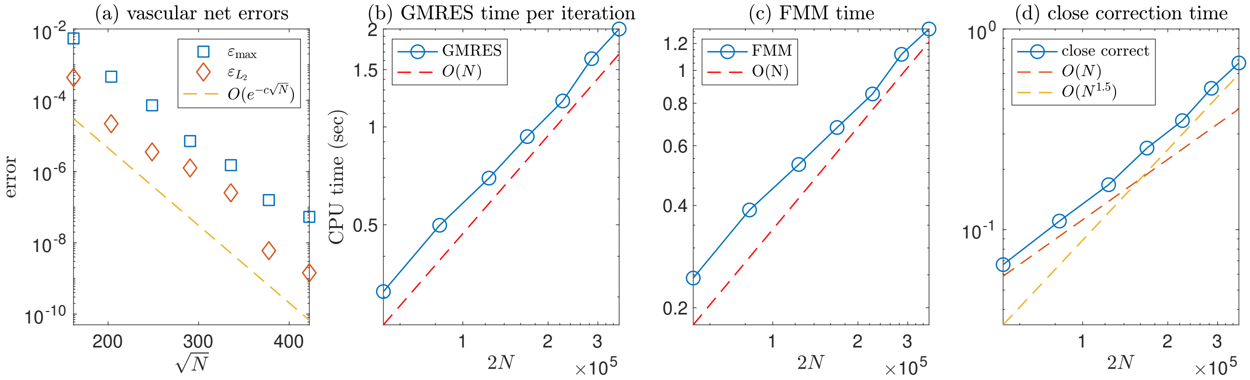

Figure 2.7: Error and timing of solving the vascular network BVP. (a) Convergence of errors in log-linear scale. (b) Log-log scale plot of the total CPU time per GMRES iteration, which consists of the FMM time, shown in (c), and the close correction time, shown in (d).

close to optimal efficiency for this discretization. The slight decrease of the percentage FMM times at smaller ϵ is due to the fact that the FMM time grows only linearly with N, while the close evaluation matrix-vector multiplication time grows like $O(N^{3/2})$. The latter estimate is obtained as follows. The number of matrix-vector multiplications grows like $O(n_\Lambda) = O(\log 1/\epsilon) = O(p)$, where each matrix-vector product takes $O(p^2)$ time. Note that $N = n_\Lambda \times p = O(p^2)$, so the total close evaluation time grows as $O(p^3) = O(N^{3/2})$. (See Figure 2.7b–d.)

Example 5. Uniform versus adaptive for close-to-touching curves. Finally, to demonstrate the advantage of using an adaptive scheme over a uniform discretization, let us consider a uniform flow past two close-to-touching disks (Figure 2.8). The background flow is a constant

$\boldsymbol{u}_\infty = [1,0]^T$, the separation is $d = 10^{-6}$, and the radii 1 and 0.1. For the uniform-resolution scheme we use a global periodic trapezoid grid on each circle, where, in order to have similar node spacings, the larger circle has 9 times as many points as the smaller one. Here, global close-evaluation is done using the spectrally accurate quadrature from [14]. On the other hand, the adaptive quadrature uses a grid that is determined by our adaptive refinement scheme of Section 2.4, with one modification that much improves the scaling in the number of refined panels (see Remark 2.5.1). We observe that, for more than 4 accurate digits, the number of unknowns required by the adaptive scheme is much less than that of the uniform-resolution scheme (Figure 2.8). The smoothness of the density function discussed in the remark below suggests that, at fixed ϵ, the uniform scheme (and also the original refinement scheme) needs $N = O(1/\sqrt{d})$ unknowns, whereas the modified adaptive scheme needs only $N = O(\log(1/d))$. It is hard to imagine improving upon the latter by much, so it is what we recommend for closely-interacting curves. (See also Examples 4 and 6 in [16] for a "globally adaptive" variant.)

Remark 2.5.1 (Refinement for close-to-touching smooth curves)*.* For viscous flow in the region between two smooth curves separated by a small distance d, asymptotic analysis gives that the width of the "bump" in fluid force scales as $O(\sqrt{d})$ [167]. By dimensional analysis, if the sum of the two curvatures of the surfaces near the contact point is κ, then the width in fact scales as $O(\sqrt{d/\kappa})$. Assuming that this also applies to the density σ, this suggests a looser criterion for refinement: panels should be refined only when they are longer than this width scale. This allows panels to come much closer than their length, without being refined. In the case of close smooth curves, the test in line 13 of Algorithm 2 can thus be modified to $(c'\sqrt{\kappa}S)^2 > d$. We find that the constant $c' = 0.7$ achieves the requested tolerance. We derive the scaling of N as follows. Setting $\kappa = 1$ for simplicity, a generic local model of the separation is $h(t) \approx d + t^2$ as a function of parameter t, and

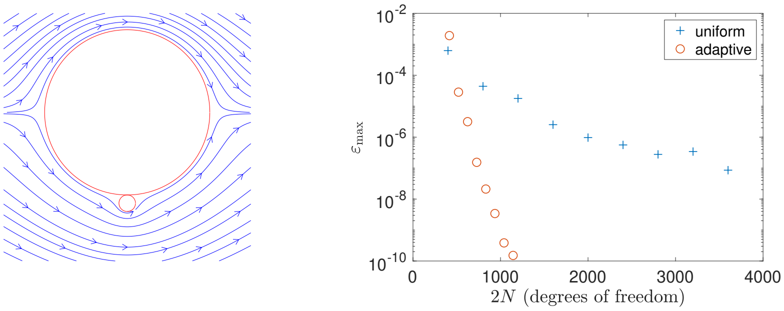

Figure 2.8: Convergence of a uniform flow past two touching disks that are $d = 10^{-6}$ apart, and whose radii are 1 and 0.1. The required number of unknowns in the adaptive scheme is much less than the global scheme with uniform resolution; see Example 5.

$n_\Lambda = O\big(\int_{-1}^{1} dt/S(h(t))\big)$ where $S(h)$ is the local panel size as a function of separation. The original refinement scheme, $S(h) = O(h)$, thus gives $n_\Lambda = O(d^{-1/2})$, whereas the modified $S(h) = O(\sqrt{h})$ gives $n_\Lambda = O(\log(1/d))$.

2.6 Conclusions

Applying complex contour quadratures of Helsing and coworkers, we have presented a set of panel quadrature rules for accurate evaluation of single- and double-layer Stokes velocity potentials, and their associated pressures and tractions. They can be used for targets that are either on or off the boundary, and can be located arbitrarily close to it. In addition, we presented an adaptive panel refinement procedure that sets the length of panels on the boundary (“h-adaptivity”), and the overall degree of approximation p (“p-adaptivity”), required to achieve a user-prescribed tolerance, purely using *geometric* information, even in the case of close-touching smooth curves. This contrasts the approach of adapting based on local *density* information, which would (at least for an iterative solver) demand multiple BIE solutions. We demonstrated via numerical experiments that our algorithm achieves super-algebraic convergence even for complex geometries with corners, and that the CPU time grows linearly with problem size and is dominated by the cost of FMMs for large-scale problems. Applications of our work include providing design tools for rapid prototyping of microfluidic chips (for cell sorting, mixing or other manipulations e.g., [98]), shape optimization (e.g., [24]) and simulating cellular-level blood flow in microvasculature.

We envision building a fast 2D particulate flow software library by utilizing the algorithms developed in this work for the fixed complex geometry components (such as microfluidic chips or vascular networks) and our global close evaluation schemes developed in [14] for the moving rigid or deformable particles (such as colloids, drops or vesicles). Incorporating corner singularity representations such as the RCIP [82, 83], or analytic expansion quadratures [156], are expected to significantly reduce the number of unknowns, increasing the performance. Note that, by contrast, it is easy to check that adapting p on a per-panel basis (i.e. full hp-adaptivity [85, 11]) is expected only to reduce the unknowns associated with corners by a factor of at most two. Another key ingredient would be a *fast direct solver* for solving the BIEs on the fixed geometries, similar to that developed in [127], wherein the boundary integral operators were compressed by exploiting their low-rank structures, inverted as a precomputation step, and applied at an optimal $O(N)$ cost at every time-step of the particulate flow simulation as particles move through the fixed geometry. One open research question in this context is: *Can we update the compressed representations as the boundary panels are refined (or coarsened) without rebuilding them?* A similar question was recently investigated in [215], where the authors report a 3× speedup when locally perturbing the geometry. We plan to explore their approach and report its performance in the context of our adaptive panel refinement procedure.

CHAPTER III

Product Integration Scheme for 3D Singular Layer Potentials

Preamble. In this chapter, we present a new approach for solving the close evaluation problem in three dimensions. The approach introduced here converts these nearly-singular integrals on a patch of the boundary to a set of non-singular line integrals on the patch boundary using the Stokes theorem on manifolds. A function approximation scheme based on harmonic polynomials is designed to express the integrand in a form that is suitable for applying the Stokes theorem. As long as the data—the boundary and the density function—is given in a high-order format, the double-layer potential and its derivatives can be evaluated with high-order accuracy using this scheme both on and off the boundary. In particular, we present numerical results demonstrating seventh-order convergence on a smooth, warped torus example achieving 10-digit accuracy in evaluating double layer potential at targets that are arbitrarily close to the boundary. The preprint is ready for review in [216].

3.1 Introduction

In this chapter, we describe a high-order accurate numerical algorithm for evaluating the double-layer potential (DLP) for Laplace equation given by

$$\mathcal{D}[\tau](\boldsymbol{r}') = \int_{\mathcal{M}} \frac{\partial G(\boldsymbol{r}' - \boldsymbol{r})}{\partial \boldsymbol{n}_{\boldsymbol{r}}} \tau(\boldsymbol{r}) \mathrm{d}S_{\boldsymbol{r}} \tag{3.1}$$

where $G(\boldsymbol{r}' - \boldsymbol{r}) = 1/4\pi|\boldsymbol{r}' - \boldsymbol{r}|$ is the Green's function for the Laplace equation, $\tau(\boldsymbol{r})$ is a density function, $\mathcal{M}$ is a closed two-dimensional manifold in $\mathbb{R}^3$, which is equivalent to boundary notation Γ in other chapters, and $\boldsymbol{n_r}$ is its normal. Layer potentials such as the DLP satisfy the underlying partial differential equation (PDE) by construction and are often employed in mathematical analysis and numerical solution of PDEs [107]. Fast and accurate numerical schemes for (3.1) are fundamentally important owing to the ubiquity of Laplace equation in sciences and engineering. Moreover, they serve as templates for other linear elliptic PDE solvers via potential theory.

In practical applications, one needs to evaluate (3.1) both on and off the surface $\mathcal{M}$. If the target $\boldsymbol{r}'$ is located far off the surface, a smooth quadrature rule designed for the given surface representation can be applied efficiently. However, the integrand in (3.1) becomes weakly-singular for on-surface targets and nearly-singular for targets located close to the surface. In both cases, specialized quadrature rules are necessary to achieve desired order of accuracy. While the subject of developing high-order rules for weakly-singular integrals is a classical one, nearly-singular integration is an active area of research. For two-dimensional problems (where $\mathcal{M}$ is a curve on the plane), significant progress has been made on accurate evaluation schemes for nearly-singular integrals, some recent works include [155, 30, 157, 4, 211] (also see references therein). In contrast, fewer number of works exist for high-order close evaluation in the case of three-dimensional problems, owing to the complexity of handling a stronger kernel singularity over high-order surface meshes.

Synopsis of the new approach. Consider a subdomain $D \subset \mathcal{M}$. A surface integral on D can be converted into a line integral on ∂D using the Stokes theorem on manifolds as long as the integrand is an exact form [180]. Clearly, this condition is not necessarily satisfied in the case of DLP (3.1) for an arbitrary τ. The main idea here is that we can construct basis functions for approximating τ in D, which when multiplied by the kernel in (3.1) are exact

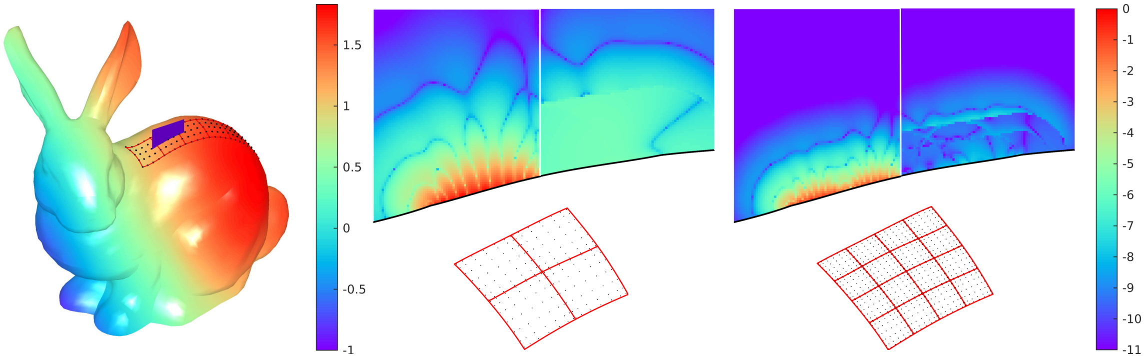

Figure 3.1: One of the key advantages of the close evaluation scheme developed in this chapter is its ease of handling arbitrary meshes. Here, we demonstrate its performance on the Stanford bunny triangulation data [192]. We used the interactive sketch-based quadrangulation method of [185] to create high-quality quad remeshings locally as shown on the top of the bunny. We evaluate the DLP at targets that are located arbitrarily close to the surface as shown on top in blue color. The surface is colored by the density function τ, which was set as $\tau(x, y, z) = e^{xy} - 1 + x + \sin(x^4 + 1/2y^3) + y - 1/2y^2 + 1/5y^6 + z$. (Middle and right) Given this setup, we demonstrate the performance of the new scheme by considering one of the quads, successively refining it two-fold and visualizing the errors due to direct evaluation of DLP via high-order smooth quadrature rule (left half) and the new close evaluation scheme (right half). We note that while the errors stagnate in a band close to the surface in the case of smooth quadrature, the new scheme achieves uniform accuracy upto 10-digits. More details on this experiment are provided in Section 3.5.

forms. Thereby, when the target $\boldsymbol{r}'$ is close to D, we can apply this procedure to convert a nearly-singular surface integral on D to a non-singular line integral on ∂D (assuming $\boldsymbol{r}'$ is not close to ∂D). In this chapter, we construct such basis functions using harmonic polynomials and quaternion algebra. The scheme is relatively insensitive to the underlying high-order surface discretization. Once the density function is expressed in our basis on D (e.g., *via* collocation), the layer potential evaluation is carried out in a similar fashion as a product integration scheme, with the caveat that some smooth line integrals on ∂D need to computed numerically in addition.

Related work. Here, we restrict our discussion to closely related recent works; a more extensive literature survey on singular and near-singular integration schemes can be found in [74, 149, 133]. In the first class of methods, the issue of close evaluation is overcome by exploiting the smoothness of DLP away from $\mathcal{M}$. In the quadrature-by-expansion (`QBX`) scheme, originally proposed in [15, 103], the DLP is approximated at centers away from $\mathcal{M}$ using high-order local expansions, which are valid at points closer to or on $\mathcal{M}$. Extension of `QBX` to three-dimensional problems was recently explored in [171, 203, 204]. A related algorithm is the `hedgehog` scheme of [133], which in turn is an extension of the earlier work by Ying et al. [214]. Similar to `QBX`, `hedgehog` exploits the smoothness of (3.1) away from the boundary and evaluates it at carefully chosen "check" points along a line passing through the target located close to $\mathcal{M}$ and extrapolates the solution to the target.

Another popular class of methods are those based on singularity subtraction, wherein, the kernel in (3.1) is split into a singular part and a smooth part, with the action of the former treated analytically. While low-order variants are often used in practice for three-dimensional problems, high-order extension was presented in [81] for toroidal geometries. Recently, an alternative strategy, termed as harmonic density interpolation (`HDI`), is presented in [149] which focuses on the density instead of the kernel. It regularizes the kernel singularities by splitting the density into two parts: one whose convolution with the kernel can be treated analytically and the other whose derivatives vanish to prescribed order as the target $\boldsymbol{r}'$ approaches the source $\boldsymbol{r}$. Lastly, regularized kernel methods for 3D close evaluation were also developed recently in [18, 189]; high-order accuracy is achieved by introducing correction terms to control the regularization error.

Our approach shares many of the desirable features of `QBX` and `hedgehog` schemes including, prominently, the ease of integration with fast algorithms such as the fast multipole method (FMM) [64] since it doesn't modify the kernel and affects the local part evaluation only. On the other hand, the fact that all the computational variables stay on the manifold $\mathcal{M}$ in our scheme offers further advantages such as avoiding the need for optimizing auxiliary parameters like local expansion centers or check points, which may be challenging in situations such as nearly self-touching geometries. Although both `HDI` and our scheme employ harmonic polynomials for approximating the density function, their usage is fundamentally different in both schemes. In [149], harmonic polynomial approximations are sought which cancel the kernel singularity to high-order as $\boldsymbol{r}' \to \boldsymbol{r}$; it is unclear if such approximations can be constructed to arbitrarily high-order in three-dimensions ([149] demonstrates third-order convergence). In contrast, our scheme only requires smoothness of the density. Lastly, a key advantage of our approach is that it works on any user supplied meshes without the need for geometry processing; high-order convergence is guaranteed as long as the boundary and the density function are specified in a high-order format (an example is shown in Figure 3.1).

In our view, our work is most closely related to the work of Helsing-Ojala [80], which developed a panel-based close evaluation scheme in two dimensions by approximating the density using monomial basis and evaluating their product with the nearly-singular kernels via recurrences. This approach has been shown to offer rapid and accurate solution of several elliptic problems [80, 14, 211]. The quaternionic harmonic polynomial approximation scheme of the density, introduced in this work, can be viewed as a 3D analogue of their complex monomial approximation scheme. Similarly, we also employ recurrences to evaluate the product of nearly-singular kernels and polynomial basis functions. We note that the differential geometry framework presented here is applicable both for two- and three-dimensional problems, thereby, is a unifying approach.

Limitations. In this work, we restrict our attention to the Laplace layer potentials only. While our close evaluation scheme can be extended to other linear elliptic PDE kernels, it is by no means trivial: the density approximation scheme needs to be tailored for each individual kernel. We note, however, that there are alternative approaches to directly apply our scheme to other PDE problems—e.g., Stokes potentials can be expressed as a linear combination of Laplace potentials and their derivatives [190]. Lastly, the method described in this chapter cannot be applied directly to globally parameterized surfaces (e.g., spherical harmonic representations). One remedy is to maintain an auxiliary adaptive surface mesh just for the purposes of close evaluation.

The remainder of this chapter is organized as follows. In Section 3.2, we review some preliminaries on exterior calculus and describe the 3D close evaluation problem using the language of exterior calculus. In Section 3.3, the key ideas of the product integration scheme are outlined, followed by a presentation of our quaternionic approximation. Then in Section 3.4, we present the overall implementation of our close evaluation scheme. We demonstrate the performance of our algorithm on a variety of examples in Section 3.5, followed by conclusions and discussion on future directions in Section 3.6.

3.2 Mathematical preliminaries

The use of exterior calculus greatly simplifies the presentation of our numerical algorithms even though, strictly speaking, is not required for their development. In this section, we review some basic concepts but refer the reader to [9, 180] (or other standard textbooks) for a more thorough introduction to this subject.

3.2.1 Exterior algebra

If V is a vector space over $\mathbb{R}$, we will denote by $Alt^k(V)$ the space of alternating k-linear maps $V \times \cdots \times V \to \mathbb{R}$. We refer to such maps as alternating algebraic k-forms. A k-linear map $\omega \in Alt^k(V)$ is called alternating if

$$\omega(\boldsymbol{v}_1, \cdots, \boldsymbol{v}_i, \cdots, \boldsymbol{v}_j, \cdots, \boldsymbol{v}_k) = -\omega(\boldsymbol{v}_1, \cdots, \boldsymbol{v}_j, \cdots, \boldsymbol{v}_i, \cdots, \boldsymbol{v}_k). \tag{3.2}$$

Thus, an algebraic k-form on V assigns to a k-tuple $(\boldsymbol{v}_1, \cdots, \boldsymbol{v}_k)$ of elements of V a real number $\omega(\boldsymbol{v}_1, \cdots, \boldsymbol{v}_k)$, with the mapping linear in each argument, and reversing sign when any two arguments are interchanged.

Given $\omega \in Alt^k(V)$ and $\eta \in Alt^l(V)$, simple tensor product of ω and η is usually not an alternating algebraic $(k+l)$-form. We instead employ the *exterior product* or *wedge product*

$\omega \wedge \eta \in Alt^{k+l}$, defined by

$$\begin{aligned}(3.3)\qquad &(\omega \wedge \eta)(\boldsymbol{v}_1, \cdots, \boldsymbol{v}_{k+l}) \\ &= \sum_{\sigma \in Sh_{k,l}} \operatorname{sgn}(\sigma)\omega(\boldsymbol{v}_{\sigma(1)}, \cdots, \boldsymbol{v}_{\sigma(k)})\eta(\boldsymbol{v}_{\sigma(k+1)}, \cdots, \boldsymbol{v}_{\sigma(k+l)}), \quad \boldsymbol{v}_i \in V,\end{aligned}$$

where $Sh_{k,l}$ is the subset of (k, l) permutations of the set $\{1, 2, \cdots, k + l\}$ such that each element $\sigma \in Sh_{k,l}$ satisfies $\sigma(1) < \sigma(2) < \cdots < \sigma(k)$, and $\sigma(k + 1) < \sigma(k + 2) < \cdots < \cdots < \sigma(k + l)$. The exterior product is both bilinear and associative.

Throughout this chapter, V will be $\mathbb{R}^3$, and k will be 1 or 2. In $\mathbb{R}^3$, the canonical basis $\boldsymbol{e}_1, \boldsymbol{e}_2$ and $\boldsymbol{e}_3$ gives rise to a natural dual basis of $Alt^1(\mathbb{R}^3)$, the space of covectors. This dual basis will often be denoted by $\mathrm{d}x, \mathrm{d}y$ and $\mathrm{d}z$. These basis elements are linear maps (not to be confused with infinitely small change in the variable). For example, $\mathrm{d}x(\boldsymbol{v}) = \boldsymbol{e}_1 \cdot \boldsymbol{v} = v_1$. The wedge product is also an operation connecting the various $Alt^k(V)$ spaces. For example, the basis for $Alt^2(\mathbb{R}^3)$ can be written using wedge products of $Alt^1(\mathbb{R}^3)$ basis elements as $\mathrm{d}x \wedge \mathrm{d}y, \mathrm{d}y \wedge \mathrm{d}z$ and $\mathrm{d}z \wedge \mathrm{d}x$.

3.2.2 Exterior calculus on manifolds

At each point $\boldsymbol{r}$ of a sufficiently smooth manifold $\mathcal{M}$ of dimension n, the tangent space $T_{\boldsymbol{r}}\mathcal{M}$ is a vector space of dimension n (in our case, $n = 2$). We could think of this as a local coordinate system. If the selection of a vector $\boldsymbol{v}(\boldsymbol{r})$ at $\boldsymbol{r}$ is made in each $T_{\boldsymbol{r}}\mathcal{M}$, we obtain a vector field.

Applying the exterior algebra construction to the tangent spaces, we obtain the exterior forms bundle $(\boldsymbol{r}, \eta)$ with $\boldsymbol{r} \in \mathcal{M}$, $\eta \in Alt^k(T_{\boldsymbol{r}}\mathcal{M})$. A differential k-form is a map ω which associates to each $\boldsymbol{r} \in \mathcal{M}$ an element $\omega_{\boldsymbol{r}} \in Alt^k(T_{\boldsymbol{r}}\mathcal{M})$. If the map $\boldsymbol{r} \in \mathcal{M} \to \omega_{\boldsymbol{r}}(\boldsymbol{v}_1(\boldsymbol{r}), \cdots, \boldsymbol{v}_k(\boldsymbol{r})) \in \mathbb{R}$ is smooth whenever the $\boldsymbol{v}_i$'s are smooth vector fields, then we say that ω is a smooth differential k-form. We denote by $\Lambda^k(\mathcal{M})$ the space of all smooth differential k-forms on $\mathcal{M}$. As $\boldsymbol{r}$ moves around smoothly on $\mathcal{M}$, ω provides a smoothly varying algebraic k-form at each tangent space $T_{\boldsymbol{r}}\mathcal{M}$.

The exterior product of differential forms can be defined pointwise from exterior product of algebraic forms

$$(\omega \wedge \eta)_{\boldsymbol{r}} = \omega_{\boldsymbol{r}} \wedge \eta_{\boldsymbol{r}}. \tag{3.4}$$

If D is an oriented submanifold of $\mathcal{M}$, and ω is a continuous k-form, then the integral $\int_D \omega$ is well-defined.

The exterior derivative d is a linear operator that maps $\Lambda^k(\mathcal{M})$ into $\Lambda^{k+1}(\mathcal{M})$ for each $k \geq 0$. We give a formula for the case $\mathcal{M}$ is a domain in $\mathbb{R}^n$. For given $\omega \in \Lambda^k(\mathcal{M})$ and vectors $\boldsymbol{v}_1, \cdots, \boldsymbol{v}_k$, we obtain a smooth mapping $\mathcal{M} \to \mathbb{R}$ given by $\boldsymbol{r} \to \omega_{\boldsymbol{r}}(\boldsymbol{v}_1, \cdots, \boldsymbol{v}_k)$. We then define

$$\mathrm{d}\omega_{\boldsymbol{r}}(\boldsymbol{v}_1, \cdots, \boldsymbol{v}_k) = \sum_{j=1}^{k+1} (-1)^{j+1} \partial_{\boldsymbol{v}_j} \omega_{\boldsymbol{r}}(\boldsymbol{v}_1, \cdots, \hat{\boldsymbol{v}}_j, \cdots, \boldsymbol{v}_{k+1}), \tag{3.5}$$

where the hat is used to indicate a suppressed argument. If $\omega \in \Lambda^k(\mathcal{M})$ and $\eta \in \Lambda^l(\mathcal{M})$, then

$$\mathrm{d}(\omega \wedge \eta) = \mathrm{d}\omega \wedge \eta + (-1)^k \omega \wedge \mathrm{d}\eta. \tag{3.6}$$

3.2.3 Integral equation formulation

Consider the following interior Dirichlet problem for the Laplace equation in a three-dimensional domain Ω bounded by $\mathcal{M}$,

$$\Delta u = 0 \quad \text{in} \quad \Omega, \qquad u = g \quad \text{on} \quad \mathcal{M}. \tag{3.7}$$

We can employ an indirect integral equation formulation [107] for solving this problem, wherein, we set $u(\boldsymbol{r}') = \mathcal{D}[\tau](\boldsymbol{r}')$, the double-layer potential as defined in (3.1). This ansatz satisfies the Laplace equation by construction and enforcing the boundary condition yields the following boundary integral equation for the unknown τ:

$$-\frac{1}{2}\tau(\boldsymbol{r}') + \mathcal{D}[\tau](\boldsymbol{r}') = g(\boldsymbol{r}'), \quad \forall \quad \boldsymbol{r}' \in \mathcal{M}, \tag{3.8}$$

where the evaluation of $\mathcal{D}$ on $\mathcal{M}$ is performed in the principal value sense. Solving this BIE for τ, one can evaluate the solution u at any target in the domain by using (3.1). Similarly, other Laplace boundary value problems can be recast as BIEs using potential theory (e.g., see [107]).

Now, let's express the DLP evaluation as integration of differential forms. On the manifold $\mathcal{M}$, we have [180, Thm. 5-6]:

$$n_1 \mathrm{d}S_{\boldsymbol{r}} = \mathrm{d}y \wedge \mathrm{d}z, \quad n_2 \mathrm{d}S_{\boldsymbol{r}} = \mathrm{d}z \wedge \mathrm{d}x, \quad n_3 \mathrm{d}S_{\boldsymbol{r}} = \mathrm{d}x \wedge \mathrm{d}y. \tag{3.9}$$

Therefore, the DLP (3.1) can be written as

$$\begin{aligned} \mathcal{D}[\tau](\boldsymbol{r}') &= \int_{\mathcal{M}} \frac{(\boldsymbol{r}' - \boldsymbol{r}) \cdot \boldsymbol{n}_{\boldsymbol{r}}}{4\pi|\boldsymbol{r}' - \boldsymbol{r}|^3} \tau(\boldsymbol{r}) \mathrm{d}S_{\boldsymbol{r}} \\ &= \int_{\mathcal{M}} \frac{(x' - x)\tau(\boldsymbol{r})}{4\pi|\boldsymbol{r}' - \boldsymbol{r}|^3} \mathrm{d}y \wedge \mathrm{d}z + \frac{(y' - y)\tau(\boldsymbol{r})}{4\pi|\boldsymbol{r}' - \boldsymbol{r}|^3} \mathrm{d}z \wedge \mathrm{d}x + \frac{(z' - z)\tau(\boldsymbol{r})}{4\pi|\boldsymbol{r}' - \boldsymbol{r}|^3} \mathrm{d}x \wedge \mathrm{d}y, \end{aligned} \tag{3.10}$$

where $\boldsymbol{r}' = (x', y', z')$ and $\boldsymbol{r} = (x, y, z)$.

3.3 Density approximation and exact form construction

In this section, we systematically introduce the key ideas required to develop our numerical scheme. We provide the necessary analytical and algebraic background employed in Section 3.4. We briefly review Stokes theorem and Poincaré's Lemma to illustrate our basic ideas in Section 3.3.1, and introduce a quaternionic approximation scheme in Section 3.3.2.

3.3.1 Stokes theorem and Poincaré's lemma

We will be relying on the Stokes theorem to evaluate (3.10) when $\boldsymbol{r}'$ is close to $\mathcal{M}$. Using exterior calculus, one can summarize the Stokes theorem on a patch D in an elegant way [180]:

Theorem 3.3.1. *(Stokes theorem) If D is a compact oriented 2-manifold, for any smooth 1-form ω defined on D, the following holds,*

$$\int_D \mathrm{d}\omega = \int_{\partial D} \omega. \tag{3.11}$$

Remark 3.3.1. The advantage of using Stokes theorem to reduce a surface integral of 2-form $\mathrm{d}\omega$ on D to a line integral of 1-form ω is essentially two-fold. One is that we have localized the work involved in evaluating layer potential on part of the integration surface. The other comes from the benefit of dimensionality reduction. Essentially, this eliminates singularity that populates the two dimensional manifold to only boundaries of its panel discretization, which has a measure zero. This further helps in accurate evaluation of layer potentials when targets are extremely close to or on the surface.

The key idea is to use Stokes theorem to evaluate the double-layer potential when a target $\boldsymbol{r}'$ is close to D. But Stokes theorem does not help with finding a suitable ω such that

$$\frac{(\boldsymbol{r}' - \boldsymbol{r}) \cdot \boldsymbol{n}_{\boldsymbol{r}}}{4\pi|\boldsymbol{r}' - \boldsymbol{r}|^3} \tau(\boldsymbol{r}) \mathrm{d}S_{\boldsymbol{r}} \approx \mathrm{d}\omega. \tag{3.12}$$

To address the challenges in systematically finding ω, we introduce one additional tool in differential geometry, Poincaré's lemma [180, Thm. 4-11]: for every differential form on an open star-shaped subset D of $\mathbb{R}^n$, suppose $\mathrm{d}\alpha = 0$ for $\alpha \in \Lambda^k(D)$, then locally there is some $\omega \in \Lambda^{k-1}(D)$ such that $\mathrm{d}\omega = \alpha$. The proof of the lemma considers a $k-$form,

$$\alpha = \sum_{i_1 < \cdots < i_k} g_{i_1, \cdots, i_k} \mathrm{d}x^{i_1} \wedge \cdots \wedge \mathrm{d}x^{i_k}, \tag{3.13}$$

and shows that $(k-1)$-form ω defined by

$$\omega = \sum_{i_1 < \cdots < i_k} \sum_{l=1}^{k} (-1)^{(l-1)} \left(\int_0^1 t^{k-1} g_{i_1, \cdots, i_k}(t\boldsymbol{x}) dt \right) x^{i_l} \, \mathrm{d}x^{i_1} \wedge \cdots \wedge \mathrm{d}\hat{x}^{i_l} \wedge \cdots \wedge \mathrm{d}x^{i_k} \tag{3.14}$$

satisfies $\mathrm{d}\omega = \alpha$ given $\mathrm{d}\alpha = 0$.

Based on this result, we can accomplish the 2-to-1 form conversion as written in (3.12). This construction process can be viewed as finding the vector potential whose curl is a given vector field.

A simplified version of the Poincaré's Lemma relevant to our setting can be summarized as follows.

Lemma 3.3.2. *(2-to-1 form conversion) Consider a compact oriented* 2*-dimensional manifold* D *in* $\mathbb{R}^3$*. Let*

$$\alpha = g_1(\boldsymbol{r})\mathrm{d}y \wedge \mathrm{d}z + g_2(\boldsymbol{r})\mathrm{d}z \wedge \mathrm{d}x + g_3(\boldsymbol{r})\mathrm{d}x \wedge \mathrm{d}y, \tag{3.15}$$

be a differential 2*-form on* D*. If* $\mathrm{d}\alpha = 0$ *(i.e.,* $\nabla \cdot (g_1, g_2, g_3) = 0$*), then*

$$\begin{aligned}\omega = {}&\left(\int_0^1 (tzg_2(t\boldsymbol{r}) - tyg_3(t\boldsymbol{r}))\,dt\right)\mathrm{d}x + \left(\int_0^1 (txg_3(t\boldsymbol{r}) - tzg_1(t\boldsymbol{r}))\,dt\right)\mathrm{d}y \\ &+ \left(\int_0^1 (tyg_1(t\boldsymbol{r}) - txg_2(t\boldsymbol{r}))\,dt\right)\mathrm{d}z\end{aligned} \tag{3.16}$$

satisfies $\mathrm{d}\omega = \alpha$*.*

Proof. See Appendix B.2 □

Consequently, it is possible to convert a surface integral into a line integral as long as the vector $\boldsymbol{g}$ is divergence-free. For example, if τ is a scalar in (3.12), it is clear that the above Lemma applies since

$$\nabla \cdot \frac{\boldsymbol{r}' - \boldsymbol{r}}{|\boldsymbol{r}' - \boldsymbol{r}|^3} = \nabla \cdot \nabla \frac{1}{|\boldsymbol{r}' - \boldsymbol{r}|} = -\delta(\boldsymbol{r}', \boldsymbol{r}),$$

where $\delta(\boldsymbol{r}', \boldsymbol{r})$ is the Dirac δ function. Therefore $\nabla \cdot \frac{\boldsymbol{r}'-\boldsymbol{r}}{|\boldsymbol{r}'-\boldsymbol{r}|^3} = 0$ for $\boldsymbol{r} \in D$, $\boldsymbol{r}' \in D^c$. In the general case, our goal is to find a high-order approximation scheme for τ which makes the vector $(\boldsymbol{r}' - \boldsymbol{r})\tau(\boldsymbol{r})/|\boldsymbol{r}' - \boldsymbol{r}|^3$ divergence-free. Clearly, standard polynomial approximation schemes (e.g., tensor-product monic polynomials) won't yield the desired result. In the next subsection, we present an approximation scheme based on harmonic polynomials and quaternionic representations that accomplishes this task.

The key insight that motivates our approach is summarized in Lemma 3.3.3, but first let's review some preliminaries on quaternions. Let $\boldsymbol{i}, \boldsymbol{j}$ and $\boldsymbol{k}$ be the standard quaternion units, that is, they satisfy the identities

$$\boldsymbol{i}^2 = \boldsymbol{j}^2 = \boldsymbol{k}^2 = \boldsymbol{ijk} = -1, \quad \boldsymbol{ij} = \boldsymbol{k} = -\boldsymbol{ji}, \quad \boldsymbol{jk} = \boldsymbol{i} = -\boldsymbol{kj}, \quad \boldsymbol{ki} = \boldsymbol{j} = -\boldsymbol{ik}. \tag{3.17}$$

A quaternionic function g is comprised of a scalar part g_0 and a vector part $\boldsymbol{g} = (g_1, g_2, g_3)$, and written as

$$g(\boldsymbol{r}) = g_0(\boldsymbol{r}) + g_1(\boldsymbol{r})\boldsymbol{i} + g_2(\boldsymbol{r})\boldsymbol{j} + g_3(\boldsymbol{r})\boldsymbol{k}.$$

Alternatively, one can write the quaternion in the pair form as $g = (g_0, \boldsymbol{g})$. For any given vector $\boldsymbol{g}$, we can define a quaternion g as above and if not specified, $g_0 = 0$ is assumed by default. Using (3.17), we can easily verify that the product of two quaternions g and h to be

$$gh = (g_0 h_0 - \boldsymbol{g} \cdot \boldsymbol{h},\ g_0 \boldsymbol{h} + h_0 \boldsymbol{g} + \boldsymbol{g} \times \boldsymbol{h}).$$

With these preliminaries, we can now state and prove the following lemma that motivates our density approximation scheme.

Lemma 3.3.3. *Let $\boldsymbol{g} = \nabla\psi$ and $\boldsymbol{f} = \nabla\phi$, where ϕ and ψ are some harmonic functions. Then each component of the quaternionic 2-form $g n_r f \mathrm{d}S_{\boldsymbol{r}}$ is exact.*

Proof. Using the identities (3.17) and after some algebra, we obtain

$$g n_r f = \left(-\left(\boldsymbol{g} \times \boldsymbol{n_r}\right) \cdot \boldsymbol{f}, -\left(\boldsymbol{g} \cdot \boldsymbol{n_r}\right) \boldsymbol{f} + \left(\boldsymbol{g} \times \boldsymbol{n_r}\right) \times \boldsymbol{f}\right)$$

The scalar part of $g\, n_r\, f\, \mathrm{d}S_{\boldsymbol{r}}$ then becomes

$$-\left(\boldsymbol{g} \times \boldsymbol{n_r}\right) \cdot \boldsymbol{f}\, \mathrm{d}S_{\boldsymbol{r}} = -\left(\boldsymbol{f} \times \boldsymbol{g}\right) \cdot \boldsymbol{n_r} \mathrm{d}S_{\boldsymbol{r}}$$

Invoking the fact that both $\boldsymbol{g}$ and $\boldsymbol{f}$ are gradients of harmonic functions, we get

$$\nabla \cdot (\boldsymbol{f} \times \boldsymbol{g}) = \boldsymbol{g} \cdot (\nabla \times \nabla\phi) - \boldsymbol{f} \cdot (\nabla \times \nabla\psi) = 0,$$

thereby, confirming that the scalar part is an exact form (using Lemma 3.3.2). Similarly, the first component of its vector part is given by

$$(n_1\,(\boldsymbol{g}\cdot\boldsymbol{f}) - g_1\,(\boldsymbol{f}\cdot\boldsymbol{n_r}) - f_1\,(\boldsymbol{g}\cdot\boldsymbol{n_r}))\,\mathrm{d}S_{\boldsymbol{r}} = ((\boldsymbol{g}\cdot\boldsymbol{f})\,\boldsymbol{e}_1 - g_1\boldsymbol{f} - f_1\boldsymbol{g})\cdot\boldsymbol{n_r}\mathrm{d}S_{\boldsymbol{r}}$$

Therefore, this term is also exact since

$$\nabla\cdot((\boldsymbol{g}\cdot\boldsymbol{f})\,\boldsymbol{e}_1 - g_1\boldsymbol{f} - f_1\boldsymbol{g}) = \frac{\partial}{\partial x}\,(\boldsymbol{g}\cdot\boldsymbol{f}) - \nabla g_1\cdot\boldsymbol{f} - g_1\Delta\phi - \nabla f_1\cdot\boldsymbol{g} - f_1\Delta\psi = 0.$$

□

The following corollary applies this Lemma to help verify the divergence-free condition in Lemma 3.3.2 to convert the double-layer integral into a 1-form integral.

Corollary 3.3.4. *Let α be a quaternionic differential 2-form on D given by*

$$\alpha = \alpha_0 + \alpha_1\boldsymbol{i} + \alpha_2\boldsymbol{j} + \alpha_3\boldsymbol{k} = \frac{(r'-r)n_r}{|r'-r|^3}f(\boldsymbol{r})\mathrm{d}S_{\boldsymbol{r}}. \tag{3.18}$$

If the vector $\boldsymbol{f}(\boldsymbol{r})$ is the gradient of a harmonic function, then $d\alpha_i = 0$, $i = 0,1,2,3$.

Proof. This result directly follows from Lemma 3.3.3 by setting $\boldsymbol{g}(\boldsymbol{r}) = \nabla\frac{1}{|\boldsymbol{r}'-\boldsymbol{r}|}$ and noting that $|r'-r| = |\boldsymbol{r}'-\boldsymbol{r}|$. □

3.3.2 Approximation scheme using harmonic polynomials and quaternionic representation

Our goal is to express the density τ in terms of some basis functions that allow us to apply Corollary 3.3.4 to convert the DLP (3.10) to a 1-form using Lemma 3.3.2. From Corollary 3.3.4, it is clear that the elements of such a basis set essentially must be in quaternionic form and their vector components must be gradients of some harmonic functions (notice that in Corollary 3.3.4 we need to expand the quaternionic form $\frac{(r'-r)n_r}{|r'-r|^3}f(\boldsymbol{r})\mathrm{d}S_{\boldsymbol{r}}$ before projecting the surface elements). To construct such a basis set, we turn to *harmonic polynomials* (we refer the reader to [51] for a review on this topic).

While there are $2p+1$ independent harmonic polynomials of degree p, we will chose p of them for our construction. The only requirement is that their gradients must be linearly independent, for example, the following set of harmonic polynomials for upto degree 7:

$$\begin{aligned}
\mathcal{P}_1 =&\{z\},\ \mathcal{P}_2 = \{x^2 - z^2,\ y^2 - z^2\},\ \mathcal{P}_3 = \{x^3 - 3xz^2,\ y^3 - 3yz^2,\ xyz\},\\
\mathcal{P}_4 =&\{x^4 - 6x^2z^2 + z^4,\ y^4 - 6y^2z^2 + z^4, 3x^2yz - yz^3,\ 3xy^2z - xz^3\},\\
\mathcal{P}_5 =&\{x^5 - 10x^3z^2 + 5xz^4,\ y^5 - 10y^3z^2 + 5yz^4,\ x^4y - 6x^2yz^2 + yz^4,\\
&xy^4 - 6xy^2z^2 + xz^4,\ -15x^2y^2z + 5x^2z^3 + 5y^2z^3 - z^5\},\\
\mathcal{P}_6 =&\{x^6 - 15x^4z^2 + 15x^2z^4 - z^6,\ y^6 - 15y^4z^2 + 15y^2z^4 - z^6,\\
&x^5y - 10x^3yz^2 + 5xyz^4,\ xy^5 - 10xy^3z^2 + 5xyz^4\\
&5x^4yz - 10x^2y^3z + y^5z,\ 5xy^4z - 10x^3y^2z + x^5z\},\\
\mathcal{P}_7 =&\{x^7 - 7xz^6 + 35x^3z^4 - 21x^5z^2,\ y^7 - 7yz^6 + 35y^3z^4 - 21y^5z^2,\\
&x^6y - 15x^4yz^2 + 15x^2yz^4 - z^6y,\ xy^6 - 15xy^4z^2 + 15xy^2z^4 - xz^6,\\
&3x^5y^2 - 3x^5z^2 - 30x^3y^2z^2 + 10x^3z^4 + 15xy^2z^4 - 3xz^6,\\
&3x^2y^5 - 3y^5z^2 - 30x^2y^3z^2 + 15x^2yz^4 + 10y^3z^4 - 3yz^6,\\
&(3x^5y - 10x^3y^3 + 3xy^5)z\}.
\end{aligned} \tag{3.19}$$

Then, we assign the gradients of each of these harmonic polynomials, expressed in quaternionic form, as the elements of the required basis set. Denoting this set by $\{f^{(k,1)}, \cdots, f^{(k,k)}\}$, $k = 1, \cdots, p$, we can easily derive them from (3.19) as

$$\begin{aligned}
&\nabla\mathcal{P}_1 = \{f^{(1,1)} = \boldsymbol{k}\}, \quad \nabla\mathcal{P}_2 = \{f^{(2,1)} = x\boldsymbol{i} - z\boldsymbol{k},\ f^{(2,2)} = y\boldsymbol{j} - z\boldsymbol{k}\},\\
&\nabla\mathcal{P}_3 = \{\left(x^2 - z^2\right)\boldsymbol{i} - 2xz\boldsymbol{k},\ \left(y^2 - z^2\right)\boldsymbol{j} - 2yz\boldsymbol{k},\ yz\boldsymbol{i} + xz\boldsymbol{j} + xy\boldsymbol{k}\},
\end{aligned} \tag{3.20}$$

and so on. Therefore, in total, there are $p(p+1)/2$ quaternionic functions in this basis set. Moreover, the set $\nabla\mathcal{P}_p$ is composed of homogenous, quaternionic polynomials of degree $(p-1)$. Thereby, a p^{th} order convergent scheme is obtained when the set $\{\nabla\mathcal{P}_k,\ k = 1, \ldots, p\}$ is used for approximating smooth quaternionic functions.

Now, consider a triangular patch $D \subset \mathcal{M}$, as illustrated in Fig. 3.2(b). On this patch, we use the basis functions $\{f^{(k,l)}(\boldsymbol{r})\}$ to approximate the density function as

$$\tau(\boldsymbol{r}) + 0\boldsymbol{i} + 0\boldsymbol{j} + 0\boldsymbol{k} \approx \sum_{k=1}^{p} \sum_{l=1}^{k} f^{(k,l)}(\boldsymbol{r})\, c^{(k,l)}, \tag{3.21}$$

where the unknown coefficients $c^{(k,l)}$ are also quaternions, that is,

$$c^{(k,l)} = c_0^{(k,l)} + c_1^{(k,l)}\boldsymbol{i} + c_2^{(k,l)}\boldsymbol{j} + c_3^{(k,l)}\boldsymbol{k}.$$

Therefore, there are $4 \cdot \frac{p(p+1)}{2}$ unknowns that need to be determined. They can be obtained, for instance, by applying a standard collocation scheme. For a p-node composite tensor product Gauss-Legendre quadrature in Fig. 3.2(c), there are $(p^2 + p)/2$ quadrature nodes on the pre-image of each triangular patch D, denoted as $\{\boldsymbol{r}^{(k,l)} | 1 \leq k \leq l \leq p\}$. Enforcing (3.21) at these quadrature nodes will generate the required $4 \cdot \frac{p(p+1)}{2}$ number of equations. Empirically, we found that the resulting square linear systems are invertible in general.

Remark 3.3.2. In our implementation, we perform a change of coordinates in each patch so that $\boldsymbol{r}^{(1,1)}$ becomes the origin (as illustrated in Fig. 3.2(c)). In this case, $c^{(1,1)} = \tau(\boldsymbol{r}^{(1,1)})$ is known, and there are only $(p^2 + p)/2 - 1$ quaternionic unknowns (and corresponding equations).

We are now ready to substitute the density approximated as in (3.21) into the DLP (3.10). However, the double-layer kernel needs to be written in quaternionic form to take advantage of Corollary 3.3.3. Following lemma summarizes the result.

Lemma 3.3.5. *Let r and n_r be the source location and the normal on D respectively, in quaternionic form:*

$$r = 0 + x\boldsymbol{i} + y\boldsymbol{j} + z\boldsymbol{k}, \ n_r = 0 + n_1\boldsymbol{i} + n_2\boldsymbol{j} + n_3\boldsymbol{k}. \tag{3.22}$$

If the density is approximated as in (3.21)*, then the scalar part of*

$$-\int_D \frac{(r' - r)n_r}{4\pi|r' - r|^3} \left(\sum_{k=1}^{p} \sum_{l=1}^{k} f^{(k,l)}(\boldsymbol{r}) c^{(k,l)} \right) \mathrm{d}S_{\boldsymbol{r}}, \tag{3.23}$$

is a p^{th} order convergent scheme to the DLP defined on D.

Proof. The proof follows from the following two observations: (i) owing to (3.21), only the scalar part of $\sum_{k=1}^{p}\sum_{l=1}^{k} f^{(k,l)}c^{(k,l)}$ remains and (ii) the scalar part of quaternion product rn_r is given by $-\boldsymbol{r}\cdot\boldsymbol{n_r}$. $\square$

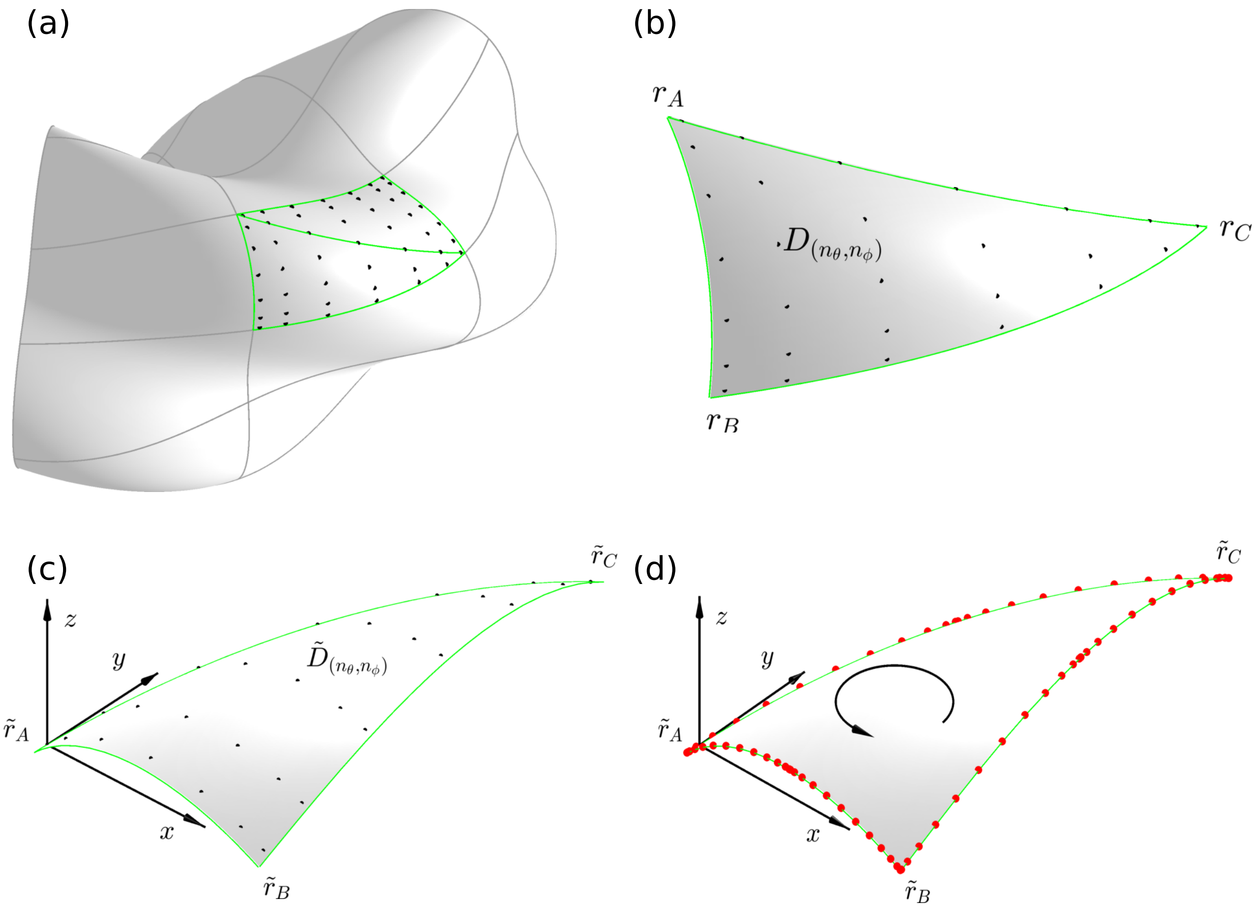

Figure 3.2: Schematic of our proposed product integration scheme for Laplace double-layer potential. (a) is part of a parameterized “cruller” surface. (b) is one of the triangular sub panel, denoted by $D_{(n_\theta,n_\phi)}$, from the rectangular panel in (a). (c) is the transformed triangular patch $\tilde{D}_{(n_\theta,n_\phi)}$. (d) shows the quadrature nodes (red) on integration contour $\partial\tilde{D}_{(n_\theta,n_\phi)}$, the boundary of the transformed triangular patch.

Finally, to construct basis functions upto an arbitrary order beyond $p = 7$, we can exploit the fact that restricting harmonic polynomials to a unit sphere yields the standard spherical harmonics [51]. Similar to (3.19), we can chose p of the $2p+1$ solid spherical harmonics of degree p. For example, let (ρ,θ,ϕ) be the coordinates of a point in spherical coordinates, then, we can set

$$\mathcal{P}_p = \{\rho^p \cos(k\phi) P_p^k(\cos\theta),\ k = 1, \cdots, p\}, \tag{3.24}$$

where $P_p^k(\cos\theta) = (-1)^k \sin^k\theta \frac{d^k}{d(\cos\theta)^k}\left(P_p(\cos\theta)\right)$ are the associated Legendre polynomials. When these functions are expressed in terms of the Cartesian coordinates, $x = \rho\sin\theta\cos\phi, y = \rho\sin\theta\sin\phi$ and $z = \rho\cos\theta$, we get a set of homogeneous harmonic polynomials of degree p. For example, consider the case of $p = 8$, $k = 2$, we get the following after using basic trigonometric identities:

$$\begin{aligned} \rho^8\cos(2\phi)P_8^2(\cos\theta) =& \frac{315}{16}\rho^8\cos(2\phi)\sin^2\theta\left(143\cos^6\theta - 143\cos^4\theta + 33\cos^2\theta - 1\right) \\ =& \frac{315}{16}\left(x^2 - y^2\right)\left(143z^6 - 143z^4(x^2+y^2+z^2)\right. \\ & \left. +33z^2(x^2+y^2+z^2)^2 - (x^2+y^2+z^2)^3\right). \end{aligned} \tag{3.25}$$

The convergence results of the approximation using this basis upto order 10 are shown in Fig. 3.3.

3.4 Numerical scheme

We now have all the tools necessary to build a high-order accurate close evaluation scheme. Here, we will restrict our discussion to toroidal geometries parameterized by an infinitely-differentiable, doubly-periodic function $\boldsymbol{r}(\theta, \phi) = (x(\theta, \phi), y(\theta, \phi), z(\theta, \phi))$, where $(\theta, \phi) \in [0, 2\pi)^2$, with the understanding that the scheme can be generalized to other topologies and problem settings (since it inherently works at the level of local patches). Such a surface can be covered by the images of disjoint union of uniform rectangular patches in parameter space as shown in Fig. 3.2(a). When target $\boldsymbol{r}'$ is far from $\mathcal{M}$, we simply use the standard Nyström discretization for evaluation of (3.1) based on a composite tensor-product Gauss-Legendre quadrature (e.g., see [13]).

For both weakly-singular and nearly-singular integrals, our numerical scheme essentially remains the same. Therefore, from here on, we do not distinguish whether the target is on or off the surface. A high-level schematic description of the scheme is given in Fig. 3.2(b-d). In a patch D where the double-layer potential is singular, we reduce the 2-form integration to 1-form integration on ∂D. The steps involved in this conversion are described next.

3.4.1 Close evaluation scheme for Laplace double-layer potentials

The computation proceeds in two stages. In the first stage, which is *target-independent*, the density function is expressed in terms of the quaternionic basis functions $\{\nabla \mathcal{P}_k\}$. In the second stage, which is *target-dependent*, a sequence of steps are outlined that accomplish 2-form to 1-form conversion of the DLP.

Stage 1: Precomputation Given the spatial discretization parameters, n_θ and n_ϕ, we cover $\mathcal{M}$ with n_θ-by-n_ϕ rectangular patches, with a composite p-by-p Gauss-Legendre quadrature on each patch as shown in Fig. 3.2(a). On a standard patch, we further divide it into two triangular patches as shown in Fig. 3.2(a). For each triangular patch D, we first identify the set of all close targets. A coordinate transform is applied on each such target $\boldsymbol{r}'$ and the sources in D such that $\boldsymbol{r}^{(1,1)}$ becomes the origin (Fig. 3.2(c)). Points after transformation are denoted as $\tilde{\boldsymbol{r}}'$, and $\tilde{\boldsymbol{r}}^{(k,l)}$, $1 \le l \le k \le p$.

Enforcing (3.21) in the transformed coordinate system at the rest of the quadrature nodes, we obtain the following system of vector equations:

$$\sum_{k=2}^{p}\sum_{l=1}^{k} A[f^{(k,l)}](\tilde{\boldsymbol{r}}^{(i,j)})C^{(k,l)} = U^{(i,j)}, \quad 1 \le j \le i,\ 1 < i \le p, \tag{3.26}$$

where the operator $A[\cdot]$ acting on a quaternionic function f, the unknown coefficient vector C and the right-hand-side vector U are given by

$$A[f] = \begin{pmatrix} 0 & -f_1 & -f_2 & -f_3 \\ f_1 & 0 & -f_3 & f_2 \\ f_2 & f_3 & 0 & -f_1 \\ f_3 & -f_2 & f_1 & 0 \end{pmatrix},\ C^{(k,l)} = \begin{pmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{pmatrix}^{(k,l)},\ \text{and}\quad U^{(i,j)} = \begin{pmatrix} \tau(\boldsymbol{r}^{(i,j)}) - \tau(\boldsymbol{r}^{(1,1)}) \\ 0 \\ 0 \\ 0 \end{pmatrix}. \tag{3.27}$$

Assembling the vector equations (3.26) for all (i, j) yields a square linear system of size $4 \cdot \left(\frac{p(p+1)}{2} - 1\right)$, for which we simply apply a direct solver. Therefore, this stage incurs a computational cost of $\mathcal{O}(p^6)$ per triangular patch D.

Stage 2: 2-to-1 form conversion and contour integration Once the quaternionic coefficients $c^{(k,l)}$ for approximating the density are found in Stage 1, the next step is to substitute them in (3.23) and proceed with converting this 2-form to quaternionic differential 1-form $\omega^{(k,l)}$ using Lemma 3.3.2. After the coordinate transformation, we can rewrite (3.23) as (we omit $\tilde{}$ on $\tilde{\boldsymbol{r}}$):

(3.28)
$$\sum_{k=2}^{p}\sum_{l=1}^{k}\left(\frac{1}{4\pi}\int_{\tilde{D}}\alpha^{(k,l)}\right)c^{(k,l)} + \frac{1}{4\pi}\int_{\tilde{D}}\alpha^{(1,1)}\tau(\boldsymbol{r}^{(1,1)}), \quad \text{where} \quad \alpha^{(k,l)} = -\frac{(r'-r)n_r}{|r'-r|^3}\,f^{(k,l)}\mathrm{d}S_{\boldsymbol{r}}.$$

The quaternionic components of $\alpha^{(k,l)}$ can be simplified to the following after carrying out the product of quaternions on the right hand side:

$$\alpha_i^{(k,l)} = \frac{1}{|\boldsymbol{r}'-\boldsymbol{r}|^3}\boldsymbol{q}_i^{(k,l)} \cdot \boldsymbol{n}_{\boldsymbol{r}}\,\mathrm{d}S_{\boldsymbol{r}}, \ k = 2, \cdots, p, \ 1 \le l \le k, \tag{3.29}$$

where $i = 0, 1, 2, 3$ corresponds is the index of the quaternion and each of the vectors $\boldsymbol{q}_i^{(k,l)}$ are k^{th} degree polynomials in $\boldsymbol{r}$, which can be derived using quaternion product as

$$\begin{aligned}
&\boldsymbol{q}_0^{(k,l)}(\boldsymbol{r}',\boldsymbol{r}) = \boldsymbol{f}^{(k,l)}(\boldsymbol{r}) \times (\boldsymbol{r}'-\boldsymbol{r}),\\
&\boldsymbol{q}_1^{(k,l)}(\boldsymbol{r}',\boldsymbol{r}) = -\big((\boldsymbol{r}'-\boldsymbol{r})\cdot\boldsymbol{f}^{(k,l)}(\boldsymbol{r})\big)\,\boldsymbol{e}_1 + (x'-x)\,\boldsymbol{f}^{(k,l)}(\boldsymbol{r}) + f_1^{(k,l)}(\boldsymbol{r})\,(\boldsymbol{r}'-\boldsymbol{r}),\\
&\boldsymbol{q}_2^{(k,l)}(\boldsymbol{r}',\boldsymbol{r}) = -\big((\boldsymbol{r}'-\boldsymbol{r})\cdot\boldsymbol{f}^{(k,l)}(\boldsymbol{r})\big)\,\boldsymbol{e}_2 + (y'-y)\,\boldsymbol{f}^{(k,l)}(\boldsymbol{r}) + f_2^{(k,l)}(\boldsymbol{r})\,(\boldsymbol{r}'-\boldsymbol{r}),\\
&\boldsymbol{q}_3^{(k,l)}(\boldsymbol{r}',\boldsymbol{r}) = -\big((\boldsymbol{r}'-\boldsymbol{r})\cdot\boldsymbol{f}^{(k,l)}(\boldsymbol{r})\big)\,\boldsymbol{e}_3 + (z'-z)\,\boldsymbol{f}^{(k,l)}(\boldsymbol{r}) + f_3^{(k,l)}(\boldsymbol{r})\,(\boldsymbol{r}'-\boldsymbol{r}).
\end{aligned} \tag{3.30}$$

The advantage of our density approximation scheme is now apparent: from (3.29), it is clear that $\alpha_i^{(k,l)}$ is an exact form by construction and we can apply Lemma 3.3.2 to convert it into a 1-form $\omega_i^{(k,l)}$ as follows:

$$\begin{aligned}\omega_i^{(k,l)} &= \left(\int_0^1 \frac{tzq_{i,2}^{(k,l)}(\boldsymbol{r}', t\boldsymbol{r}) - tyq_{i,3}^{(k,l)}(\boldsymbol{r}', t\boldsymbol{r})}{|t\boldsymbol{r}-\boldsymbol{r}'|^3}\mathrm{d}t\right)\mathrm{d}x \\ &+ \left(\int_0^1 \frac{txq_{i,3}^{(k,l)}(\boldsymbol{r}', t\boldsymbol{r}) - tzq_{i,1}^{(k,l)}(\boldsymbol{r}', t\boldsymbol{r})}{|t\boldsymbol{r}-\boldsymbol{r}'|^3}\mathrm{d}t\right)\mathrm{d}y \\ &+ \left(\int_0^1 \frac{tyq_{i,1}^{(k,l)}(\boldsymbol{r}', t\boldsymbol{r}) - txq_{i,2}^{(k,l)}(\boldsymbol{r}', t\boldsymbol{r})}{|t\boldsymbol{r}-\boldsymbol{r}'|^3}\mathrm{d}t\right)\mathrm{d}z\end{aligned} \tag{3.31}$$

Notice that the numerators of each of the integrands in the 1-form above are polynomials of degree $(k+1)$ in the variable t. Defining $M_k(\boldsymbol{r}', \boldsymbol{r}) = \int_0^1 \frac{t^k}{|t\boldsymbol{r}-\boldsymbol{r}'|^3}\mathrm{d}t$ and using (3.30), we can separate out the terms that depend on t and rewrite $\omega_i^{(k,l)}$ as

$$\begin{aligned}\omega_i^{(k,l)} = &\left(v_{i,1}^{(k,l)}(\boldsymbol{r})M_{k+1} + w_{i,1}^{(k,l)}(\boldsymbol{r}', \boldsymbol{r})M_k\right)\mathrm{d}x + \left(v_{i,2}^{(k,l)}(\boldsymbol{r})M_{k+1} + w_{i,2}^{(k,l)}(\boldsymbol{r}', \boldsymbol{r})M_k\right)\mathrm{d}y \\ &+ \left(v_{i,3}^{(k,l)}(\boldsymbol{r})M_{k+1} + w_{i,3}^{(k,l)}(\boldsymbol{r}', \boldsymbol{r})M_k\right)\mathrm{d}z,\end{aligned} \tag{3.32}$$

where, for brevity, we omitted $(\boldsymbol{r}', \boldsymbol{r})$ dependency on M_k. Here, $v_{i,j}^{(k,l)}(\boldsymbol{r})$ are $(k+1)^{\text{th}}$ degree polynomials in $\boldsymbol{r}$ and $w_{i,j}^{(k,l)}(\boldsymbol{r}', \boldsymbol{r})$ are k^{th} degree polynomials in $\boldsymbol{r}$ and linear in $\boldsymbol{r}'$. The explicit expressions for these can be easily derived from (3.30) and (3.31); for example, in the case of $i = 0$, we have

$$\begin{aligned} v_{0,1}^{(k,l)}(\boldsymbol{r}) &= \left(y^2+z^2\right) f_1^{(k,l)}(\boldsymbol{r}) - xyf_2^{(k,l)}(\boldsymbol{r}) - xzf_3^{(k,l)}(\boldsymbol{r}) \\ w_{0,1}^{(k,l)}(\boldsymbol{r}', \boldsymbol{r}) &= -\left(y'y + z'z\right) f_1^{(k,l)}(\boldsymbol{r}) + x'yf_2^{(k,l)}(\boldsymbol{r}) + x'zf_3^{(k,l)}(\boldsymbol{r}) \\ v_{0,2}^{(k,l)}(\boldsymbol{r}) &= -xyf_1^{(k,l)}(\boldsymbol{r}) + \left(x^2+z^2\right) f_2^{(k,l)}(\boldsymbol{r}) - yzf_3^{(k,l)}(\boldsymbol{r}) \\ w_{0,2}^{(k,l)}(\boldsymbol{r}', \boldsymbol{r}) &= y'xf_1^{(k,l)}(\boldsymbol{r}) - \left(x'x + z'z\right) f_2^{(k,l)}(\boldsymbol{r}) - y'zf_3^{(k,l)}(\boldsymbol{r}) \\ v_{0,3}^{(k,l)}(\boldsymbol{r}) &= -xzf_1^{(k,l)}(\boldsymbol{r}) - yzf_2^{(k,l)}(\boldsymbol{r}) + \left(x^2+y^2\right) f_3^{(k,l)}(\boldsymbol{r}) \\ w_{0,3}^{(k,l)}(\boldsymbol{r}', \boldsymbol{r}) &= z'xf_1^{(k,l)}(\boldsymbol{r}) + z'yf_2^{(k,l)}(\boldsymbol{r}) - \left(x'x + y'y\right) f_3^{(k,l)}(\boldsymbol{r}) \end{aligned} \tag{3.33}$$

Also see Appendix D.4 where we derive explicit formulas for all the quantities $\omega_i^{(k,l)}$, $v_{i,j}^{(k,l)}$ and $w_{i,j}^{(k,l)}$ in the case of a second-order scheme.

Therefore, the kernel singularity in the DLP is now encoded into the moments $\{M_k\}$. For each source $\boldsymbol{r}$ and target $\boldsymbol{r}'$, we evaluate these moments analytically via recurrences given in

Appendix C.3. Now, we can express the complete 1-form ω as a linear combination of each individual 1-forms with coefficients $C^{(k,l)}$ and the constant term corresponding to $k = 1$ as

$$\omega = \tau(\boldsymbol{r}^{(1,1)})\omega_0^{(1,1)} + \sum_{k=2}^{p}\sum_{l=1}^{k} \varOmega^{(k,l)}C^{(k,l)}, \tag{3.34}$$

where $\varOmega^{(k,l)} = [\omega_0^{(k,l)}, \omega_1^{(k,l)}, \omega_2^{(k,l)}, \omega_3^{(k,l)}]$. Then, we evaluate boundary path integral $\int_{\partial\tilde{D}} \omega$ using a high-order smooth quadrature rule (Gauss-Legendre). However, note that the 1-forms (3.32) are still singular if $\boldsymbol{r}'$ approaches $\boldsymbol{r}$ (as can be inferred from the base conditions (C.3.3)). But since the sources now reside on ∂D, this situation can be avoided altogether by simply choosing a larger patch.

The computational complexity of this stage is primarily dictated by the evaluation of the moments $\{M_k\}$. For each target, evaluating the recurrences for these moments take $\mathcal{O}(p)$ time for each of the $\mathcal{O}(p)$ sources on ∂D, bringing the total per-target cost to $\mathcal{O}(p^2)$. Therefore, the computational cost of both stages of the close evaluation per patch D is $\mathcal{O}(p^6 + n_{targ}p^2)$, where n_{targ} is the number of targets that are considered to be close to D.

3.4.2 A generalization to evaluate the single-layer and the gradient of double-layer Laplace potentials

In several applications, one needs to evaluate high-order derivatives of layer potentials either as a post-processing step or, in some cases, as part of representing the solution itself. Lemma 3.3.3 simplifies this task for us since we can exploit the fact that the kernels themselves are harmonic functions. For example, assuming the density is approximated as in (3.21), it is easy to show that the scalar part of the integral

$$\begin{aligned} -\int_{\mathcal{M}} \frac{1}{|r'-r|^3} \Bigg(\left(\delta_{1j} - 3\frac{(x'-x)(r'-r)}{|r'-r|^2}\right) n_r,\ \left(\delta_{2j} - 3\frac{(y'-y)(r'-r)}{|r'-r|^2}\right) n_r, \\ \left(\delta_{3j} - 3\frac{(z'-z)(r'-r)}{|r'-r|^2}\right) n_r \Bigg) \left(\sum_{k=1}^{p}\sum_{l=1}^{k} f^{(k,l)}(\boldsymbol{r})c^{(k,l)} \right) \mathrm{d}S_{\boldsymbol{r}}, \end{aligned} \tag{3.35}$$

is an approximation to $\nabla\mathcal{D}(\tau)(\boldsymbol{r})$. The requirements $\mathrm{d}\alpha = 0$ for converting the directional derivatives of the double-layer potential to 1-form ω are satisfied based on the interchangeability of partial derivatives and exterior derivatives. However, constructing the 1-forms in this case also requires us to evaluate integrals of the form $L_k(\boldsymbol{r}', \boldsymbol{r}) = \int_0^1 \frac{t^k}{|t\boldsymbol{r}-\boldsymbol{r}'|^5} dt$; recurrences to evaluate these are provided in Appendix C.3.

Lastly, we consider the single-layer potential (SLP), given by $\mathcal{S}[\tau](\boldsymbol{r}') = \int_{\mathcal{M}} \frac{1}{|\boldsymbol{r}'-\boldsymbol{r}|}\tau(\boldsymbol{r})\, \mathrm{d}S_{\boldsymbol{r}}$. Since the SLP does not have a 2-form structure like the DLP as in (3.12), we instead write it as

$$\mathcal{S}[\tau](\boldsymbol{r}') = \int_{\mathcal{M}} \frac{(r'-r)\, n_r}{|r'-r|^3} \overline{n}_r \overline{(r'-r)} \tau(\boldsymbol{r})\, \mathrm{d}S_{\boldsymbol{r}}, \tag{3.36}$$

using quaternion algebra. Now, we can treat it the same way as a DLP but with a modified density function. That is, we construct the following approximation,

$$\overline{n}_r \overline{(r'-r)} \tau(\boldsymbol{r}) \approx \sum_{k=1}^{p} \sum_{l=1}^{k} f^{(k,l)}(\boldsymbol{r})\, c^{(k,l)}, \tag{3.37}$$

substitute in (3.36) and follow the close evaluation scheme as in Section 3.4.1. While it appears like this approach requires us to perform the approximation (3.37) independently for each target, since the term $\overline{(r'-r)}$ is separable, we can just form it once per patch with minor bookkeeping.

3.5 Numerical results and discussion

In this section, we present numerical results from a series of tests to validate the accuracy of our close evaluation scheme. We consider three different geometries of toroidal and spherical topologies. First, we conduct a convergence test of the quaternionic approximation algorithm presented in Section 3.3.2. We then test the performance of our DLP close evaluation, first *via* self-convergence on three different geometries, followed by a boundary value problem solve, whose exact solution is known analytically.

3.5.1 Convergence properties of the quaternionic approximation

Consider a local patch D and a smooth function τ defined on it. We can express the coordinate functions of D and τ in terms of the tensor-product monic polynomials:

$$\boldsymbol{r} \approx [x, y, z]^T = [x, y, \sum_{1\leq k+l\leq m} a^D_{k,l} x^k y^l]^T, \quad \tau(\boldsymbol{r}) \approx \sum_{1\leq k'+l'\leq m} a^\tau_{k',l'} x^{k'} y^{l'}, \tag{3.38}$$

where $a^D_{k,l}$ and $a^\tau_{k',l'}$ are the Taylor coefficients of the $z-$component and τ respectively, in terms of x and y.

In this test, we provide empirical evidence that the quaternionic approximation scheme employed in (3.21) is p^{th} order convergent. To do so, we proceed as follows. In the domain $(x, y) \in [-0.5, 0.5]^2$, we generate a sequence of the function pairs (z, τ) by setting $a^D_{k,l} = a^\tau_{k',l'} = 1$ and the rest to zero for each admissible pairs of indices (i.e., such that $1 \leq k + l \leq m$ and $1 \leq k' + l' \leq m$). Two such pairs are plotted in Fig. 3.3 (left). Then, on each of these pairs, we construct the density approximation in quaternionic form (3.21) *via* collocation by solving (3.26). We report the relative L^∞ error between $\tau(\boldsymbol{r})$ and $\sum_{k=1}^{p}\sum_{l=1}^{k} f^{(k,l)}(\boldsymbol{r})\, c^{(k,l)}$ measured on a 250×250 target grid for four different h_D, leg size of right triangular patches, in Fig. 3.3 (middle). We observe that 9-digit accuracy is reached as we reduce h_D. Moreover, from Fig. 3.3 (right), we can see that the expected order of convergence is attained asymptotically.

3.5.2 Laplace DLP evaluation test

Here, we consider the double-layer solution *ansatz* (3.1) for the exterior Laplace equation. We begin with a cruller surface from Ref. [13] which is parameterized by an infinitely differentiable, double 2π-periodic function $\boldsymbol{r}(\theta, \phi) : [0, 2\pi)^2 \to \mathbb{R}^3$, followed by a cushion surface from Ref. [149], which has a global parameterization in spherical coordinate $\boldsymbol{r}(\theta, \phi) : [-\pi/2, \pi/2] \times [0, 2\pi) \to \mathbb{R}^3$. Lastly, to demonstrate the applicability of our approach to arbitrary parameterizations, we consider an input mesh without any analytic description of

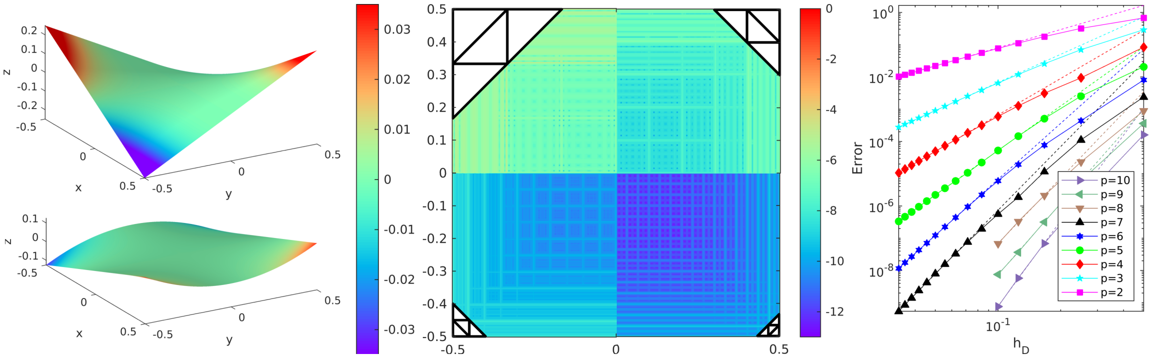

Figure 3.3: *Left:* Two examples of the function pairs (z, τ) constructed using the procedure outlined in Sec 3.5.1. In each case, the $z-$component is plotted with color scaled by the density τ. *Middle:* The density approximation scheme (3.21) is applied to each of these function pairs and the max of the relative L^∞ error is plotted here. The domain in (x, y) is subdivided in the manner shown here, $h_D = 1/6,\ 1/10,\ 1/20,\ 1/30$. The four skeletons on the corners illustrate refinements of the triangular grids at four corresponding quadrants. *Right:* Convergence plot of approximation error using $p = 2, \cdots, 7$. Dashed lines of corresponding colors are plots of expected error. For $p > 7$, the generalized higher-order approximation scheme in (3.24) was employed.

the geometry.

Example 1: "Cruller" geometry. This example demonstrates the handling of complex geometries parameterized by doubly 2π-periodic functions. We consider a smooth, warped torus surface $\mathcal{M}$ in Cartesian coordinates, given by

(3.39)

$$\boldsymbol{r} = \boldsymbol{r}(\theta, \phi) = \left(\left(a + f\left(\theta, \phi\right)\cos\left(\theta\right)\right)\cos\left(\phi\right),\ \left(a + f\left(\theta, \phi\right)\cos\left(\theta\right)\right)\sin\left(\phi\right),\ f\left(\theta, \phi\right)\sin\left(\theta\right)\right)$$

where $f(\theta, \phi) = b + w_c \cos(w_n \phi + w_m \theta)$, $a = 1$ and $b = 1/2$. We test the close evaluation scheme using mean curvature H as density:

$$\tau(\theta, \phi) = H(\theta, \phi) = \frac{1}{2}\left(EN - 2FM + GL\right)/\left(EG - F^2\right) \tag{3.40}$$

where the fundamental forms are given by $E = \boldsymbol{r}_\phi \cdot \boldsymbol{r}_\phi$, $F = \boldsymbol{r}_\phi \cdot \boldsymbol{r}_\theta$, $G = \boldsymbol{r}_\theta \cdot \boldsymbol{r}_\theta$, $L = \boldsymbol{r}_{\phi\phi} \cdot \boldsymbol{n}$, $M = \boldsymbol{r}_{\phi\theta} \cdot \boldsymbol{n}$, $N = \boldsymbol{r}_{\theta\theta} \cdot \boldsymbol{n}$, and the normal $\boldsymbol{n} = \left(\boldsymbol{r}_\phi \times \boldsymbol{r}_\theta\right)/|\boldsymbol{r}_\phi \times \boldsymbol{r}_\theta|$. We discretize $\mathcal{M}$ uniformly with n_θ-by-n_ϕ rectangular patches and conduct a self-convergence study with reference values computed using the close evaluation scheme with $n_\theta = 108, n_\phi = 144$. The results are shown in Table 3.1 and Fig. 3.4. The errors are measured at targets located on the $\phi = \pi/2$ plane

	$p = 3$		4		5		6		7	
$n_\theta \times n_\phi$	max E_{rel}	$\hat{p}$	max E_{rel}	$\hat{p}$	max E_{rel}	$\hat{p}$	max E_{rel}	$\hat{p}$	max E_{rel}	$\hat{p}$
12× 16	3.70e-02		1.25e-02		6.30e-03		3.68e-03		1.32e-03	
36× 48	6.72e-04	3.65	8.27e-05	4.57	6.05e-06	6.32	1.56e-06	7.07	2.51e-07	7.80
60× 80	4.92e-05	5.12	3.32e-06	6.30	3.36e-07	5.66	3.40e-08	7.50	1.35e-09	10.2
84× 112	1.79e-05	3.00	9.67e-07	3.67	4.59e-08	5.91	3.61e-09	6.66	9.37e-11	7.94

Table 3.1: Laplace DLP close evaluation scheme using mean curvature as prescribed density in the exterior of a smooth, warped torus surface parameterized by $(\theta, \phi) \in [0, 2\pi)^2$, with $w_c = 0.065$, $w_m = 3$ and $w_n = 5$. A cross-section on the YZ-plane ($\phi = \pi/2$) is chosen to study the convergence. We report both the maximum relative error and the observed convergence rate ($\hat{p}$) across the same slice as the number of panels are increased.

(as shown in Fig. 3.4, left). We note that the expected order of convergence is observed asymptotically.

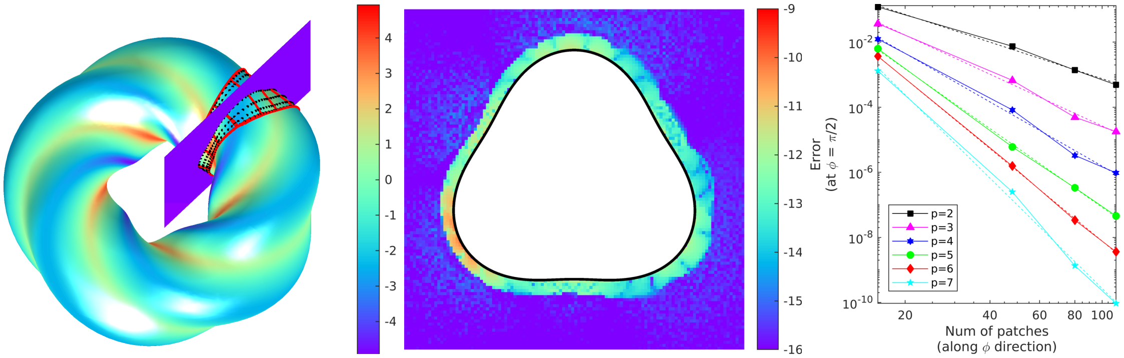

Figure 3.4: Laplace DLP close evaluation scheme on a smooth, warped torus surface, using mean curvature as prescribed density. Left: Surface $\mathcal{M}$, with $w_c = 0.065$, $w_m = 3$ and $w_n = 5$, showing panel divisions (red lines) intersecting YZ-plane and Nyström nodes (black). The color indicates the magnitude of mean curvature. Middle: Cross-section view of the $\log_{10}$ relative error in the exterior of $\mathcal{M}$, in the YZ-plane ($\phi = \pi/2$), with 84×112 patches. Right: Rate of convergence of the relative errors across the same shown slice with respect to number of panels along toroidal direction, for $p = 2, \cdots, 7$.

Example 2: "Cushion" geometry. In this example, we consider the cushion surface $\mathcal{M}$ from Ref. [149] defined by

$$\mathcal{M} = \boldsymbol{r}(\theta, \phi) = \left(f(\theta, \phi)\cos(\theta)\cos(\phi),\ f(\theta, \phi)\sin(\theta)\cos(\phi),\ f(\theta, \phi)\sin(\phi)\right) \tag{3.41}$$

where $f(\theta, \phi) = (4/5 + 1/2\,(\cos(2\theta) - 1)\,(\cos(4\phi) - 1))^{1/2}$. The density τ is set to the mean curvature (3.40). As in [149], we discretize $\mathcal{M}$ using a set of non-overlapping patches and tensor product grids in each patch (Fig. 3.5, left). The DLP is evaluated on two planes, intersecting the surface at $\phi = 3\pi/16$ and $\phi = 27\pi/16$, as shown in 3.5. The relative errors are plotted as the number of patches are increased for the case of $p = 7$. While the experimental setup is slightly different from that in [149], our goal is to showcase that several more digits of accuracy can be obtained using higher order close evaluation schemes. For example, from Fig. 3.5, we observe that around 10-digits of accuracy (or better) can be achieved at targets that are arbitrarily close to the surface on both the planes.

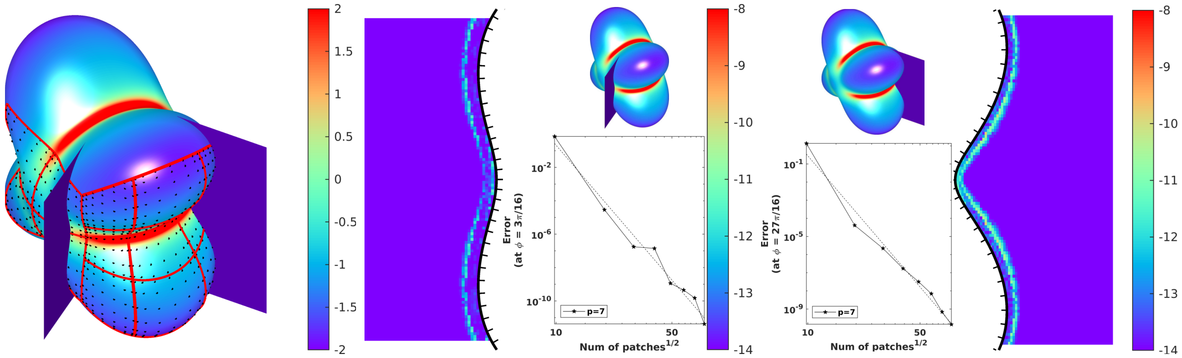

Figure 3.5: Laplace DLP close evaluation on a cushion-shaped geometry. Left: Illustration of the surface discretization with non-overlapping patches. The magnitude of mean curvature is indicated by the color. The solution is evaluated on the shown slices. Middle: Cross-section view on the plane $\phi = 3\pi/16$ of the $\log_{10}$ relative error in the exterior of the cushion. The inset plots the relative error corresponding to $p = 7$ as a function of the number of patches. Max relative error is 4.1314×10^{-12} with a total number of 6144 ($32 \times 32 \times 6$) patches. Current surface discretization is shown by the ticks ($'|'$) along the surface. Right: The $\log_{10}$ relative error in the exterior of the cushion on the plane $\phi = 27\pi/16$ with a total number of 6144 patches. Max relative error is 1.4157×10^{-10}.

Example 3: "Bunny" geometry. Finally, we showcase that when a smooth surface is given without (θ, ϕ) parameterization, a local correction could still be implemented to higher order based on a local set of control points and a high order polynomial interpolation. The bunny geometry is taken from a standard mesh library. We use an interactive high-quality quad remeshing tool developed in [185] to return a collection of patches and nodes within. We then construct a 6^{th} order polynomial approximation to each patch, and generate Gauss-Legendre quadrature nodes for naive evaluation and boundaries of each patch for applying our close evaluation scheme. We use a randomly chosen density function $\tau(x, y, z) = e^{xy} - 1 + x + \sin(x^4 + 1/2y^3) + y - 1/2y^2 + 1/5y^6 + z$ on the bunny surface. The results on one patch are shown in Fig. 3.1 to demonstrate the performance.

3.5.3 Laplace BVP test

For this numerical experiment, we solve Laplace interior boundary value problems (BVP) inside two cruller (3.39) domains, one with higher curvature compared to the other, as shown in Figs. 3.6 and 3.7. The boundary data is generated from a superposition of randomly distributed point sources located exterior to the domain, i.e., we evaluate the function $g(\boldsymbol{r}) = \sum_{j=1}^{N_s} G(\boldsymbol{r} - \boldsymbol{r}_j)h_j$, where $\boldsymbol{r} \in \mathcal{M}$, the sources $\boldsymbol{r}_j$ are located in the exterior and the source strengths h_j are set to some random values. Starting from this boundary data, the BIE (4.9) can be solved for unknown density function τ. We use the quadrature scheme and BIE solver developed in [13] for this purpose. The resulting density τ for both geometries are plotted in Fig. 3.6 (left) and Fig. 3.7 (left). The numerical solution of the BVP is evaluated on the $\phi = \pi/8$ plane using our close evaluation scheme with $p = 7$. It is then compared against the exact solution $u_{\text{exact}}(\boldsymbol{r}') = \sum_{j=1}^{N_s} G(\boldsymbol{r}' - \boldsymbol{r}_j)h_j$, where $\boldsymbol{r}' \in \Omega$. We can make the following observations from Figs. 3.6 and 3.7: (i) the accuracy is uniform throughout the interior (that is, no degradation at targets close to the boundary), (ii) similar to the exterior problem case, order of convergence is consistent with the basis function space used, and (iii) level of accuracy achieved is consistent with the complexity of the geometry.

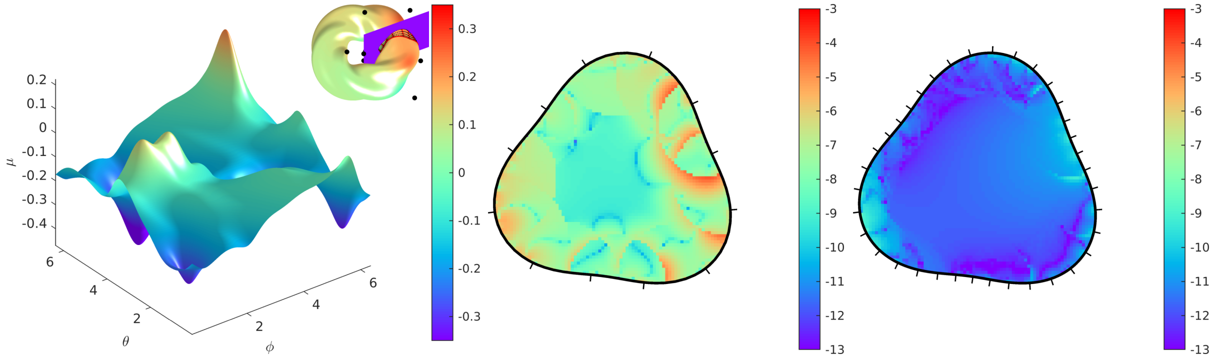

Figure 3.6: Solution of Laplace BVP in the interior of a torus, using an indirect DLP formulation (4.9). Left: Density function plotted as a function of toroidal and poloidal directions. The inset at the upper right corner shows the geometry whose surface color indicates the Dirichlet data due to a few randomly placed sources (black dots) in the exterior. The solution is evaluated on the shown slice. Here the shape parameters in (3.39) were set to $w_c = 0.065$, $w_m = 3$ and $w_n = 5$. Middle: Cross-section view of the $\log_{10}$ relative error on the plane $\phi = \pi/8$. Max relative error is 7.1761×10^{-5} with 12×16 panels. Right: The $\log_{10}$ relative error on the same shown slice with 36×48 panels. Max relative error is 3.7405×10^{-9}.

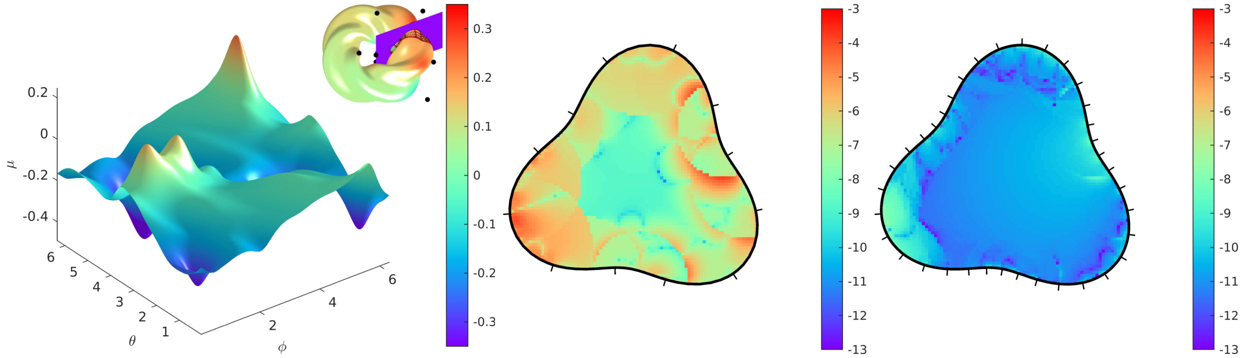

Figure 3.7: Same setup as in Fig. 3.6 but with shape parameters $w_c = 0.1$, $w_m = 3$ and $w_n = 5$ (higher curvature). The max relative error is 1.6098×10^{-4} with 12×16 panels (middle) and with 36×48 panels, it is 1.6667×10^{-8} (right).

3.6 Conclusions

In summary, we presented a high-order technique for evaluating nearly singular integrals for Laplace layer potentials in three dimensions and demonstrated its efficacy on a range of test problems. This scheme has modest requirements: it can work on any user-supplied surface mesh directly to solve the close evaluation problem up to the level of accuracy commensurate with that of the given data. Moreover, to some extent, this scheme is *dimension-agnostic*. It is intriguing to note that if we carry out the same steps for two-dimensional DLPs, we will likely recover the scheme of Helsing-Ojala [80]. There, nearly singular integrals were computed using a quadrature scheme that employs piecewise complex monomial approximation on panels, Cauchy's theorem and recurrence relations. In our case, we can use harmonic polynomials in two dimensions—which are closely related to complex monomials—for density approximation and the DLP can be transformed from a 1-form line integral to 0-form antiderivative evaluation (i.e., using (3.13) and (3.14)).

We plan to extend our work on several fronts. Our immediate next step is to integrate the close evaluation routine with an open-source FMM package (e.g., [63]) and test its performance on large-scale examples. Another natural direction is to consider various other elliptic PDE kernels including Helmholtz, Stokes and Navier kernels. As indicated earlier, such a task is non-trivial since the density approximation likely needs to be modified and new recurrences for 1-forms need to be derived. Lastly, accurate three-dimensional close evaluation schemes open up possibilities to investigate physical phenomena that are otherwise hard to simulate including chain formation and chaotic behavior in vesicle electrohydrodynamics [196, 210], flows through complex geometries [127, 211] and self-assembly of active particles [212, 105]. We plan to generalize these previous works to large-scale three-dimensional flows with arbitrary particle shapes.

CHAPTER IV

Product Integration Scheme for Volume Potentials on Irregular Domains

Preamble. In this chapter, we extend our close evaluation idea for layer potential evaluation to computing volume potentials for Gaussian and Laplace kernel in complex domains. By applying Green's theorem, these smooth or singular domain integrals on a volume mesh is converted to a set of line integrals on the boundary skeleton of the volume mesh. This new approach allows easier solver development in complex domains, without drastic refinement on leaf level boxes near the domain boundary. We address two fundamental difficulties encountered in solving inhomogeneous elliptic PDEs. One is a unified high order evaluation scheme for both singular and nearly-singular volume potential on irregular leaf boxes, provided a high order body force representation is given. Secondly, a novel approximation scheme on irregular boxes is employed, which allows a more stable computation on approximation coefficients without requiring any extension of body force outside irregular domain. We then present various numerical experiments on the Poisson problem in complex geometries.

4.1 Introduction

Here, we describe a high-order accurate numerical scheme for evaluating the volume potentials of the form

$$\mathcal{V}[f]\,(\boldsymbol{r}') = \int_\Omega G(\boldsymbol{r}' - \boldsymbol{r})\, f(\boldsymbol{r})\, \mathrm{d}A, \tag{4.1}$$

where Ω is an irregular two-dimensional domain, f is a given density function and G is the Green's function that satisfies an underlying partial differential equation (PDE). In particular, we will focus on two kernels:

$$G(\boldsymbol{r}) = \begin{cases} -\frac{1}{2\pi}\log|\boldsymbol{r}| & \text{(Poisson kernel)}, \\ \\ e^{-\frac{|\boldsymbol{r}|^2}{\delta}} & \text{(Gaussian kernel)}. \end{cases}$$

Accurate volume potential schemes can be useful for solving non-linear equations using potential theory. One of the main advantages of this approach is in most cases no solve is required. However, they have not gained traction in solving nonlinear PDEs. One of the main reasons is the close evaluation problem. Recent advances in the field overcame the close evaluation problem and there are now several options for two-dimensional [155, 157, 4, 211, 77, 139, 80, 14, 103, 15, 78, 30, 157, 4, 149] and three-dimensional problems [171, 203, 204, 133, 214, 81, 149, 18, 189]. This motivated a resurgence of interest in evaluating volume potentials and applying potential theory to solve nonlinear problems [206, 60, 1].

Existing schemes include: (1) embedded boundary method, FEM + BIE [20]; (2) Smooth function extension and apply FFT: PUX [48], Fourier Continuation [28]; (3) application to Navier-Stokes [1], which generalized [61]. There also exists FMM accelerated adaptive box-codes: Harper-Greengard-Zorin [112], PVFMM[126] in 3D; Ethridge-Greengard [43], Greengard-Lee[61] in 2D.

Product integration schemes separate the issue of singular evaluation with function approximation: once the density is approximated, singular integration with polynomials is computed analytically. For a regular box domain, this can be done for a number of kernels including Laplace and Gaussian kernels. However, a recursive integration on irregular domains is hard to do. In this paper, employing differential geometry tools, we show how to do product integration partially on irregular domains. To to keep things simple and illustrate the ideas, in this paper we only consider regular Cartesian grids.

The structure of this article is as follows. In Section 4.2, we review some preliminaries on exterior calculus and the integration equation formulation. In Section 4.3, we introduce the ideas of the product integration scheme, volume discretization, and a source term approximation scheme that matches inner product in the sense of volume integral. In Section 4.4, we present the overall implementation of the volume integral, its coupling with FMM, and briefly go through the boundary integral solver for the homogeneous solution. In Section 4.5, we conduct numerical experiments on Gauss transform and several Poisson problems, followed by conclusions and discussion on future direction in Section 4.6.

4.2 Mathematical Preliminaries

4.2.1 Exterior calculus

Here, we apply exterior calculus as a key tool, previously used in Chapter IV, to handle volume integral of arbitrary irregular domain discretizations in a unified manner. While the use of exterior calculus was shown to be capable of evaluating nearly-singular surface integrals to high accuracy in our previous work, here we focus more on showcasing its strength in advancing a unified common framework for computing high order volume quadrature rules in arbitrary domain Ω. This also relies on the general Stokes' theorem using differential forms and exterior derivatives,

Theorem 4.2.1. *(Stokes theorem) If Ω is a compact oriented 2-manifold, for any smooth 1-form ω defined on Ω, the following holds,*

$$\int_{\Omega} \mathrm{d}\omega = \int_{\partial\Omega} \omega. \tag{4.2}$$

where ω is a smooth 1-form defined on Ω, and $d(*)$ is the exterior derivative operator.

In our setting, it follows from the general Stokes' theorem that if function M and N are piecewise smooth on Ω and having continuous partial derivatives, then

$$\oint_{\partial\Omega} M(x,y)\,\mathrm{d}y - N(x,y)\,\mathrm{d}x = \int_{\Omega} \left(\frac{\partial M}{\partial x} + \frac{\partial N}{\partial y}\right) \mathrm{d}x\mathrm{d}y \tag{4.3}$$

To go backwards of this equality relation, we use the Poincaré's lemma [180, Thm. 4-11], with also the benefit of easily extending this to higher dimensions. A simplified version of the Poincaré's Lemma relevant to our setting can be summarized as follows:

Lemma 4.2.2. *(2-to-1 form conversion) If* Ω *is a compact oriented* 2*-dimensional manifold in* $\mathbb{R}^2$*, denote a 2-form* $\alpha = g(x,y)\,\mathrm{d}x \wedge \mathrm{d}y$*, where* $g(x,y) = \frac{\partial M}{\partial x} + \frac{\partial N}{\partial y}$*, then* $M(x,y) = x\left(\int_0^1 t\, g(tx,ty)\,\mathrm{d}t\right)$*,* $N(x,y) = y\left(\int_0^1 t\, g(tx,ty)\,\mathrm{d}t\right)$ *in* (4.3)*, i.e.*

$$\int_{\Omega} \alpha = \int_{\Omega} g(x,y)\,\mathrm{d}x \wedge \mathrm{d}y = \oint_{\partial\Omega} x\left(\int_0^1 t\, g(tx,ty)\,\mathrm{d}t\right) \mathrm{d}y - y\left(\int_0^1 t\, g(tx,ty)\,\mathrm{d}t\right) \mathrm{d}x \tag{4.4}$$

Proof. Here $\alpha = g(x,y)\,\mathrm{d}x \wedge \mathrm{d}y$ is the 2-form $\mathrm{d}\omega$ in Thm. 4.2.1. It is straightforward to verify $g(x,y) = \frac{\partial M}{\partial x} + \frac{\partial N}{\partial y}$:

$$\begin{aligned} \frac{\partial M}{\partial x} + \frac{\partial N}{\partial y} &= \int_0^1 2t\, g(tx,ty)\,\mathrm{d}t + x\left(\int_0^1 t^2 \frac{\partial}{\partial x} g(tx,ty)\,\mathrm{d}t\right) + y\left(\int_0^1 t^2 \frac{\partial}{\partial y} g(tx,ty)\,\mathrm{d}t\right) \\ &= \int_0^1 \frac{d}{\mathrm{d}t}\left(t^2\, g(tx,ty)\right)\,\mathrm{d}t = t^2\, g(tx,ty)\Big|_0^1 = g(x,y) \end{aligned} \tag{4.5}$$

□

Similarly, if Ω is a compact oriented 3-dimensional manifold in $\mathbb{R}^3$, $\alpha = g(x,y,z)\,\mathrm{d}x\wedge\mathrm{d}y\wedge\mathrm{d}z$ is a 3-form, where $g(x,y,z) = \frac{\partial M}{\partial x}+\frac{\partial N}{\partial y}+\frac{\partial P}{\partial z}$, then we have $M(x,y,z) = x\left(\int_0^1 t^2\, g(tx,ty,tz)\,\mathrm{d}t\right)$, $N(x,y,z) = y\left(\int_0^1 t^2\, g(tx,ty,tz)\,\mathrm{d}t\right)$, and $P(x,y,z) = y\left(\int_0^1 t^2\, g(tx,ty,tz)\,\mathrm{d}t\right)$, i.e.

$$\begin{aligned} \int_{\Omega} \alpha &= \int_{\Omega} g(x,y,z)\,\mathrm{d}x \wedge \mathrm{d}y \wedge \mathrm{d}z \\ &= \oint_{\partial\Omega} x\left(\int_0^1 t^2\, g(tx,ty,tz)\,\mathrm{d}t\right) \mathrm{d}y \wedge \mathrm{d}z + y\left(\int_0^1 t^2\, g(tx,ty)\,\mathrm{d}t\right) \mathrm{d}z \wedge \mathrm{d}x \\ &\quad + z\left(\int_0^1 t^2\, g(tx,ty)\,\mathrm{d}t\right) \mathrm{d}x \wedge \mathrm{d}y \end{aligned} \tag{4.6}$$

4.2.2 Integral equation formulation

To be more specific, we consider solving well-posed Dirichlet problem for the Poisson's equation in a two-dimensional multiply connected domain Ω,

$$-\Delta u = f \quad \text{in} \quad \Omega, \quad u = g \quad \text{on} \quad \partial\Omega. \tag{4.7}$$

Using the linearity of the differential operator Δ, the Poisson problem is split into a homogeneous part u^H solved with a Laplace single layer boundary integral formulation, and a particular part u^P, where the solution is obtained by evaluating a volume potential over the forcing term. The particular solution

$$u^P(\boldsymbol{r}') = \mathcal{V}[f]\,(\boldsymbol{r}') = -\frac{1}{2\pi}\int_\Omega \log\left(|\boldsymbol{r}' - \boldsymbol{r}|\right)\, f(\boldsymbol{r})\,\mathrm{d}A \tag{4.8}$$

satisfies (4.7) with a different boundary data. The homogeneous solution is then acquired by setting a Laplace single-layer potential solution ansatz $u^H(\boldsymbol{r}') = \mathcal{S}[\mu](\boldsymbol{r}')$ to the following Dirichlet problem with modified boundary condition:

$$-\Delta u^H = 0 \quad \text{in} \quad \Omega, \quad u^H = g - u^P \quad \text{on} \quad \partial\Omega. \tag{4.9}$$

Enforcing the boundary condition yields boundary integral equation $\mathcal{S}[\mu](\boldsymbol{r}') = g(\boldsymbol{r}') - u^P(\boldsymbol{r}')$ for the unknown μ.

In short, we first evaluate volume potential particular solution $u^P(\boldsymbol{r}')$, and obtain the boundary data $g(\partial\Omega) - u^P(\partial\Omega)$ for the reduced homogeneous problem (4.9). We then obtain the solution u^H to (4.9) using a indirect boundary integral formulation. The solution to the Poisson's equation (4.7) is then the sum of the two solutions, $u(\boldsymbol{r}') = u^H(\boldsymbol{r}') + u^P(\boldsymbol{r}')$ for $\boldsymbol{r}' \in \Omega$. If f and Ω are simple, u may be calculated analytically or using FFT-based method. However, they are usually not as applicable for highly irregular discretizations of a domain.

4.3 Scheme

In the first subsection, we introduce our volume potential evaluation scheme for smooth and singular kernel in general in some complex geometry Ω. In the second subsection, we discuss the volume discretization scheme used in our numerical experiments. In the last subsection, we give details on the approximation scheme on both regular and irregular boxes.

4.3.1 Volume Integral Scheme

Quadrature scheme for Gaussian kernel

For smooth kernel like Gaussian, a volume quadrature scheme depending on $\partial\Omega$ and $\mathbf{0}$, the origin of local coordinate system, can be computed:

$$v(\boldsymbol{r}') = \int_{\Omega} e^{-\frac{|\boldsymbol{r}'-\boldsymbol{r}|^2}{\delta}} f(\boldsymbol{r}) \,\mathrm{d}A = \oint_{\partial\Omega} M_\delta(\boldsymbol{r}', \boldsymbol{r}) \,\mathrm{d}y - N_\delta(\boldsymbol{r}', \boldsymbol{r}) \,\mathrm{d}x, \tag{4.10}$$

where $\Omega \subset \mathbb{R}^2$, $M_\delta(\boldsymbol{r}', \boldsymbol{r}) = x \int_0^1 te^{-\frac{|\boldsymbol{r}'-t\boldsymbol{r}|^2}{\delta}} f(t\boldsymbol{r}) \,\mathrm{d}t$ and $N_\delta(\boldsymbol{r}', \boldsymbol{r}) = y \int_0^1 te^{-\frac{|\boldsymbol{r}'-t\boldsymbol{r}|^2}{\delta}} f(t\boldsymbol{r}) \,\mathrm{d}t$.

Remark 4.3.1. If $\Omega \subset \mathbb{R}^3$, then we have

$$\begin{aligned} v(\boldsymbol{r}') &= \int_{\Omega} e^{-\frac{|\boldsymbol{r}'-\boldsymbol{r}|^2}{\delta}} f(\boldsymbol{r}) \,dV \\ &= \oint_{\partial\Omega} M_\delta(\boldsymbol{r}', \boldsymbol{r}) \,\mathrm{d}y \wedge \mathrm{d}z + N_\delta(\boldsymbol{r}', \boldsymbol{r}) \,\mathrm{d}z \wedge \mathrm{d}x + P_\delta(\boldsymbol{r}', \boldsymbol{r}) \,\mathrm{d}x \wedge \mathrm{d}y, \end{aligned} \tag{4.11}$$

$M_\delta = x \int_0^1 t^2 e^{-\frac{|\boldsymbol{r}'-t\boldsymbol{r}|^2}{\delta}} f(t\boldsymbol{r}) \,\mathrm{d}t$, $N_\delta = y \int_0^1 t^2 e^{-\frac{|\boldsymbol{r}'-t\boldsymbol{r}|^2}{\delta}} f(t\boldsymbol{r}) \,\mathrm{d}t$, $P_\delta = z \int_0^1 t^2 e^{-\frac{|\boldsymbol{r}'-t\boldsymbol{r}|^2}{\delta}} f(t\boldsymbol{r}) \,\mathrm{d}t$.

In our numerical implementation, we use volume quadrature nodes $\{t_\ell \boldsymbol{r}_j\}$, where $\{t_\ell\}_1^p$ is a pth order Gauss-Legendre quadrature nodes on $[0, 1]$, and $\{\boldsymbol{r}_j\}_1^N$ are panel-based pth order Gauss-Legendre quadrature nodes discretization of $\partial\Omega$. The quadrature weight associated with each volume quadrature node $t_\ell \boldsymbol{r}_j$ are given explicitly in Section 4.4.1.

This fixed quadrature is inaccurate when the integration kernel is sharply peaked near the evaluation point. We use analytic integration to resolve this. We denote $I_\alpha(\boldsymbol{r}', \boldsymbol{r}, \delta) = \int_0^1 t^\alpha e^{-\frac{|\boldsymbol{r}'-t\boldsymbol{r}|^2}{\delta}} \mathrm{d}t$

$$I_\alpha(\boldsymbol{r}', \boldsymbol{r}, \delta) = \frac{\boldsymbol{r}' \cdot \boldsymbol{r}}{|\boldsymbol{r}|^2} I_{\alpha-1}(\boldsymbol{r}', \boldsymbol{r}, \delta) + \frac{(\alpha-1)\,\delta}{2|\boldsymbol{r}|^2} I_{\alpha-2}(\boldsymbol{r}', \boldsymbol{r}, \delta) - \frac{\delta}{2|\boldsymbol{r}|^2} e^{-\frac{|\boldsymbol{r}'-t\boldsymbol{r}|^2}{\delta}} \tag{4.12}$$

All the higher order I_α can be computed from I_0 and I_1

$$\begin{aligned} I_0(\boldsymbol{r}', \boldsymbol{r}, \delta) =& \frac{\sqrt{\pi\delta}}{2|\boldsymbol{r}|} e^{\frac{|\boldsymbol{r}'\cdot\boldsymbol{r}|^2}{\delta|\boldsymbol{r}|^2} - \frac{|\boldsymbol{r}'|^2}{\delta}} \left(\operatorname{erf}\left(\frac{|\boldsymbol{r}|}{\sqrt{\delta}} - \frac{\boldsymbol{r}'\cdot\boldsymbol{r}}{\sqrt{\delta}|\boldsymbol{r}|}\right) + \operatorname{erf}\left(\frac{\boldsymbol{r}'\cdot\boldsymbol{r}}{\sqrt{\delta}|\boldsymbol{r}|}\right)\right) \\ I_1(\boldsymbol{r}', \boldsymbol{r}, \delta) =& \frac{\sqrt{\pi\delta}\,(\boldsymbol{r}'\cdot\boldsymbol{r})}{2|\boldsymbol{r}|^3} e^{\frac{|\boldsymbol{r}'\cdot\boldsymbol{r}|^2}{\delta|\boldsymbol{r}|^2} - \frac{|\boldsymbol{r}'|^2}{\delta}} \left(\operatorname{erf}\left(\frac{|\boldsymbol{r}|}{\sqrt{\delta}} - \frac{\boldsymbol{r}'\cdot\boldsymbol{r}}{\sqrt{\delta}|\boldsymbol{r}|}\right) + \operatorname{erf}\left(\frac{\boldsymbol{r}'\cdot\boldsymbol{r}}{\sqrt{\delta}|\boldsymbol{r}|}\right)\right) \\ & - \frac{\delta}{2|\boldsymbol{r}|^2}\left(e^{-\frac{|\boldsymbol{r}'-\boldsymbol{r}|^2}{\delta}} - e^{-\frac{|\boldsymbol{r}'|^2}{\delta}}\right) \end{aligned} \tag{4.13}$$

where $\operatorname{erf}(z) = \frac{2}{\sqrt{\pi}}\int_0^z e^{-t^2}\mathrm{d}t$ is the error function.

Quadrature scheme for Laplace single layer kernel

For singular kernel, analytic expressions of 1-forms for each body force basis function need to be worked out. This is a direct result of kernel singularity, which renders naive Chebyshev quadrature inaccurate when any fixed order nodes $t_i \in [0,1]$ were used in shrinking the boundary $\partial\Omega$. It is inevitable to develop specialized adaptive quadrature nodes depending on the location of target singularity.

$$u(\boldsymbol{r}') = \oint_{\partial\Omega} -y\left(\int_0^1 t\,\log\left(|\boldsymbol{r}' - t\boldsymbol{r}|\right)\,f(t\boldsymbol{r})\,\mathrm{d}t\right)\mathrm{d}x + x\left(\int_0^1 t\,\log\left(|\boldsymbol{r}' - t\boldsymbol{r}|\right)\,f(t\boldsymbol{r})\,\mathrm{d}t\right)\mathrm{d}y \tag{4.14}$$

For basis element $x^m y^n$, we need to compute $L_\alpha(\boldsymbol{r}', \boldsymbol{r}) = 2\int_0^1 t^\alpha \log\left(|\boldsymbol{r}' - t\boldsymbol{r}|\right)\mathrm{d}t$ where $0 \le \alpha \le m+n+1$.

$$\begin{aligned} L_\alpha(\boldsymbol{r}', \boldsymbol{r}) =& \frac{2\alpha\,(\boldsymbol{r}'\cdot\boldsymbol{r})}{(\alpha+1)\,|\boldsymbol{r}|^2} L_{\alpha-1}(\boldsymbol{r}', \boldsymbol{r}) - \frac{(\alpha-1)\,|\boldsymbol{r}'|^2}{(\alpha+1)\,|\boldsymbol{r}|^2} L_{\alpha-2}(\boldsymbol{r}', \boldsymbol{r}) \\ & + \frac{|\boldsymbol{r}' - \boldsymbol{r}|^2}{(\alpha+1)\,|\boldsymbol{r}|^2}\log|\boldsymbol{r}' - \boldsymbol{r}|^2 + \frac{2\,(\boldsymbol{r}'\cdot\boldsymbol{r})}{\alpha\,(\alpha+1)\,|\boldsymbol{r}|^2} - \frac{2}{(\alpha+1)^2} \end{aligned} \tag{4.15}$$

For the case $\boldsymbol{r}' = (0,0)$, no recursion is needed, and this formula reduces to

$$L_\alpha(\boldsymbol{r}', \boldsymbol{r}) = \frac{1}{\alpha+1}\log|\boldsymbol{r}|^2 - \frac{2}{(\alpha+1)^2} \tag{4.16}$$

For the case $\boldsymbol{r}' \neq (0,0)$, to get this recursion start, we need $L_0(\boldsymbol{r}',\boldsymbol{r})$ and $L_1(\boldsymbol{r}',\boldsymbol{r})$:

$$\begin{aligned} L_0(\boldsymbol{r}',\boldsymbol{r}) &= \left(1-\frac{\boldsymbol{r}'\cdot\boldsymbol{r}}{|\boldsymbol{r}|^2}\right)\log|\boldsymbol{r}'-\boldsymbol{r}|^2+\frac{\boldsymbol{r}'\cdot\boldsymbol{r}}{|\boldsymbol{r}|^2}\log|\boldsymbol{r}'|^2 \\ &\quad + 2\frac{|\boldsymbol{r}'\times\boldsymbol{r}|}{|\boldsymbol{r}|^2}\tan^{-1}\left(\frac{|\boldsymbol{r}|^2\,|\boldsymbol{r}'\times\boldsymbol{r}|}{|\boldsymbol{r}'\times\boldsymbol{r}|^2-\left(|\boldsymbol{r}|^2-(\boldsymbol{r}'\cdot\boldsymbol{r})\right)(\boldsymbol{r}'\cdot\boldsymbol{r})}\right)-2 \\ L_1(\boldsymbol{r}',\boldsymbol{r}) &= \frac{\boldsymbol{r}'\cdot\boldsymbol{r}}{|\boldsymbol{r}|^2}L_0(\boldsymbol{r}',\boldsymbol{r})+\frac{|\boldsymbol{r}'-\boldsymbol{r}|^2}{2|\boldsymbol{r}|^2}\log|\boldsymbol{r}'-\boldsymbol{r}|^2-\frac{|\boldsymbol{r}'|^2}{2|\boldsymbol{r}|^2}\log|\boldsymbol{r}'|^2+\frac{\boldsymbol{r}'\cdot\boldsymbol{r}}{|\boldsymbol{r}|^2}-\frac{1}{2} \end{aligned} \tag{4.17}$$

A volume quadrature scheme can be worked out based on these analytic 1−form. Before that, we must attend a few details to justify that $L_\alpha(\boldsymbol{r}',\boldsymbol{r})$ are 1-forms of the proper sort to be used in singular volume potential evaluation. The feasibility of applying Stokes' theorem to singular volume potential are presented in the following lemmas, which is generalized by considering a slight perturbation of the domain to get rid of the singularity. Here, we restrict our discussion to Laplace kernel only.

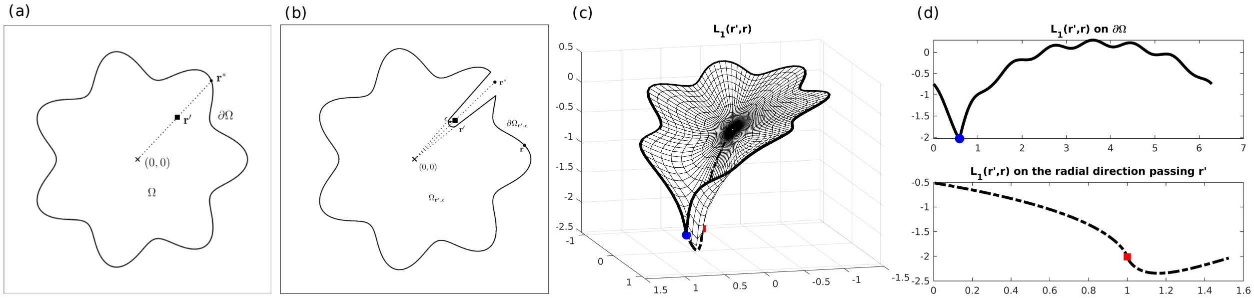

Figure 4.1: (a): a starfish geometry with uniform 2D volume grid; (b): $\Omega_{\boldsymbol{r}',\epsilon}$; (c): $L_1(\boldsymbol{r}',\boldsymbol{r})$ in Ω; (d): $L_1(\boldsymbol{r}',\boldsymbol{r})$ on $\partial\Omega$ and radial direction passing $\boldsymbol{r}'$.

Corollary 4.3.1. *Let Ω be a bounded star-shaped domain with respect to $(0,0)$, $\Omega_{\boldsymbol{r}',\epsilon}$ be a perturbed subset of Ω, such that $\Omega_{\boldsymbol{r}',\epsilon}$ is still star-shaped with respect to $(0,0)$, and for a given target $\boldsymbol{r}' \in \Omega$ and $\boldsymbol{r}' \neq (0,0)$,*

$$\boldsymbol{r}' \notin \Omega_{\boldsymbol{r}',\epsilon}, \quad \int_{\Omega\backslash\Omega_{\boldsymbol{r}',\epsilon}} \mathrm{d}A < \epsilon$$

, as shown in Fig.4.1(a)(b), then

$$u_\epsilon(\boldsymbol{r}') = \int_{\Omega_{\boldsymbol{r}',\epsilon}} \log\left(|\boldsymbol{r}'-\boldsymbol{r}|\right) f(\boldsymbol{r})\,\mathrm{d}A = \oint_{\partial\Omega_{\boldsymbol{r}',\epsilon}} P(\boldsymbol{r}',\boldsymbol{r})\,\mathrm{d}x + Q(\boldsymbol{r}',\boldsymbol{r})\,\mathrm{d}y \tag{4.18}$$

$P(\boldsymbol{r}',\boldsymbol{r}) = -y\left(\int_0^1 t\,\log\left(|\boldsymbol{r}'-t\boldsymbol{r}|\right)\,f(t\boldsymbol{r})\,\mathrm{d}t\right)$, $Q(\boldsymbol{r}',\boldsymbol{r}) = x\left(\int_0^1 t\,\log\left(|\boldsymbol{r}'-t\boldsymbol{r}|\right)\,f(t\boldsymbol{r})\,\mathrm{d}t\right)$.

Proof. Since $\Omega_{\boldsymbol{r}',\epsilon}$ is a star-shaped domain with piecewise smooth, simple closed boundary $\partial\Omega_{\boldsymbol{r}',\epsilon}$, and Laplace single-layer kernel $G(\boldsymbol{r}',\boldsymbol{r})$ is non-singular, $P(\boldsymbol{r}',\boldsymbol{r})$ and $Q(\boldsymbol{r}',\boldsymbol{r})$ are continuously differentiable functions of $\boldsymbol{r}$ on $\Omega_{\boldsymbol{r}',\epsilon}$. It follows Theorem 4.2.1 to conclude $u_\epsilon(\boldsymbol{r}') = \oint_{\partial\Omega_{\boldsymbol{r}',\epsilon}} P(\boldsymbol{r}',\boldsymbol{r})\,\mathrm{d}x + Q(\boldsymbol{r}',\boldsymbol{r})\,\mathrm{d}y$. □

Lemma 4.3.2. *Let* Ω, $\Omega_{\boldsymbol{r}',\epsilon}$ *be the same star-shaped domains as in Corollary 4.3.1,* $u(\boldsymbol{r}') = \int_\Omega \log\left(|\boldsymbol{r}'-\boldsymbol{r}|\right)\, f(\boldsymbol{r})\,\mathrm{d}V$, $u_\epsilon(\boldsymbol{r}') = \int_{\Omega_{\boldsymbol{r}',\epsilon}} \log\left(|\boldsymbol{r}'-\boldsymbol{r}|\right)\, f(\boldsymbol{r})\,\mathrm{d}V$, *where* $f(\boldsymbol{r})$ *is a smooth source function, then*

$$|u(\boldsymbol{r}') - u_\epsilon(\boldsymbol{r}')| = \left|\int_{\Omega\backslash\Omega_{\boldsymbol{r}',\epsilon}} \log\left(|\boldsymbol{r}'-\boldsymbol{r}|\right)\, f(\boldsymbol{r})\,\mathrm{d}A\right| \to 0, \ \text{as } \epsilon \to 0 \tag{4.19}$$

Proof. We subdivide $\Omega\backslash\Omega_{\boldsymbol{r}',\epsilon}$ into an ϵ-ball centered at $\boldsymbol{r}'$ and the rest of it where the Laplace kernel is bounded by $\log|\epsilon|$. If we consider the volume integral using a local polar coordinate system at $\boldsymbol{r}'$ within the ϵ-ball, the integrand function behaves like $\rho\log|\rho|$, where $\rho = |\boldsymbol{r}'-\boldsymbol{r}|$. Since $\rho\log|\rho|$ approached 0 as $\rho \to 0$, and bounded on the ϵ-ball, we conclude the volume integral on this ϵ-ball is $\mathcal{O}(\epsilon)$. On the other hand, the integrand function is bounded by $M\log(\epsilon)$ outside the ϵ-ball, where M depends on source term $f(\boldsymbol{r})$, we have the volume integral is $\mathcal{O}\left(|\epsilon\log(\epsilon)|\right)$. Thus we obtain $|u(\boldsymbol{r}') - u_\epsilon(\boldsymbol{r}')| \to 0$, as $\epsilon \to 0$.

□

Theorem 4.3.3. *Let* Ω *be a bounded star-shaped domain with respect to* $(0,0)$, $f(\boldsymbol{r})$ *be a smooth source function, then*

$$u(\boldsymbol{r}') = \int_\Omega \log\left(|\boldsymbol{r}'-\boldsymbol{r}|\right)\, f(\boldsymbol{r})\,\mathrm{d}A = \oint_{\partial\Omega} P(\boldsymbol{r}',\boldsymbol{r})\,\mathrm{d}x + Q(\boldsymbol{r}',\boldsymbol{r})\,\mathrm{d}y \tag{4.20}$$

$P(\boldsymbol{r}',\boldsymbol{r}) = -y\left(\int_0^1 t\,\log\left(|\boldsymbol{r}'-t\boldsymbol{r}|\right)\, f(t\boldsymbol{r})\,\mathrm{d}t\right)$, $Q(\boldsymbol{r}',\boldsymbol{r}) = x\left(\int_0^1 t\,\log\left(|\boldsymbol{r}'-t\boldsymbol{r}|\right)\, f(t\boldsymbol{r})\,\mathrm{d}t\right)$.

Proof. Take $\epsilon \to 0$ in previous lemma and corollary. □

Lemma 4.3.4. *For any* $\boldsymbol{r}' \in \Omega$, $P(\boldsymbol{r}', \boldsymbol{r})$ *and* $Q(\boldsymbol{r}', \boldsymbol{r})$ *in Theorem 4.3.3 are continuously differentiable on* Ω *except on* $s\boldsymbol{r}'$*, where* $1 \leq s \leq |\boldsymbol{r}^*|/|\boldsymbol{r}'|$*,* $\boldsymbol{r}^*$ *is the intersection of* $s\boldsymbol{r}'$ *and* $\partial\Omega$*, as shown in Fig.4.1(c)(d)*

Proof. Given $\boldsymbol{r}' \in \Omega$, $\boldsymbol{r} \notin \{s\boldsymbol{r}' | 1 \leq s \leq |\boldsymbol{r}^*|/|\boldsymbol{r}'|\}$, we can define star-shaped perturbation $\Omega_{\boldsymbol{r}',\epsilon}$, $P(\boldsymbol{r}', \boldsymbol{r})$ and $Q(\boldsymbol{r}', \boldsymbol{r})$, where the upper limit of ϵ depends on $|\boldsymbol{r}' - \boldsymbol{r}|$, and derivatives of $L_\alpha(\boldsymbol{r}', \boldsymbol{r})$ in equation (4.15) are bounded by $\frac{c}{|\boldsymbol{r}'-\boldsymbol{r}|}$. Therefore by taking $\epsilon \to 0$, we get continuously differentiable limit of $P(\boldsymbol{r}', \boldsymbol{r})$ and $Q(\boldsymbol{r}', \boldsymbol{r})$ with respect to $\boldsymbol{r}$ defined on Ω. $\square$

Remark 4.3.2. It is probably not surprising that we have some remnants as a result of the singularity of the Laplace kernel, but with far less severe conditions. This suggests an adaptive panel distribution along $\partial\Omega$ clustering around $\boldsymbol{r}^*$ to resolve the cusp of 1-forms. This cusp along radial direction $s\boldsymbol{r}'$ can be shown by computing the derivative of $\int_0^1 t \log(|\boldsymbol{r}' - t\boldsymbol{r}|)\, f(t\boldsymbol{r})\, dt$ along along direction $\boldsymbol{r}'_\perp$, perpendicular to $s\boldsymbol{r}'$. This can be seen easily from the following,

$$\begin{aligned}
&\lim_{\epsilon\to 0} \frac{\int_0^1 \log\left((1-st)^2 + \epsilon^2\right)\, \mathrm{d}t - \int_0^1 \log\left((1-st)^2\right)\, \mathrm{d}t}{\epsilon} \\
=&\lim_{\epsilon\to 0} \mathcal{O}(\epsilon) + 2\left(\tan^{-1}\left(\frac{s-1}{\epsilon}\right) + \tan^{-1}\left(\frac{1}{\epsilon}\right)\right) = \pm 2\pi
\end{aligned} \tag{4.21}$$

Here the rest of the integrand, $tf(st\boldsymbol{r}' + \epsilon\boldsymbol{r}'_\perp)$, does not affect the nature of the logarithm part when taking directional derivative.

4.3.2 Volume Mesh

Generate volume mesh $\{\mathcal{B}_k\}$

Fig 4.2(a) illustrate the volume mesh generated from a uniform Cartesian grid. We start from a grid of uniform volume discretization boxes $\{\hat{\mathcal{B}}_k\}$ (grey dash-line mesh) that covers Ω. Then we use $\{\mathcal{B}_k = \hat{\mathcal{B}}_k \cap \Omega\}$ as our volume mesh. Besides the size of $\hat{\mathcal{B}}_k$ and whether the forcing term is well resolved, the quality of representing $\partial\mathcal{B}_k$ will also affect the overall performance of our smooth and singular volume quadrature scheme(Section 4.3.1). Here we rely on an analytical formula of $\partial\Omega$ and root finding to define a parametric representation for $\partial\mathcal{B}_k$ to at least 12 digits.

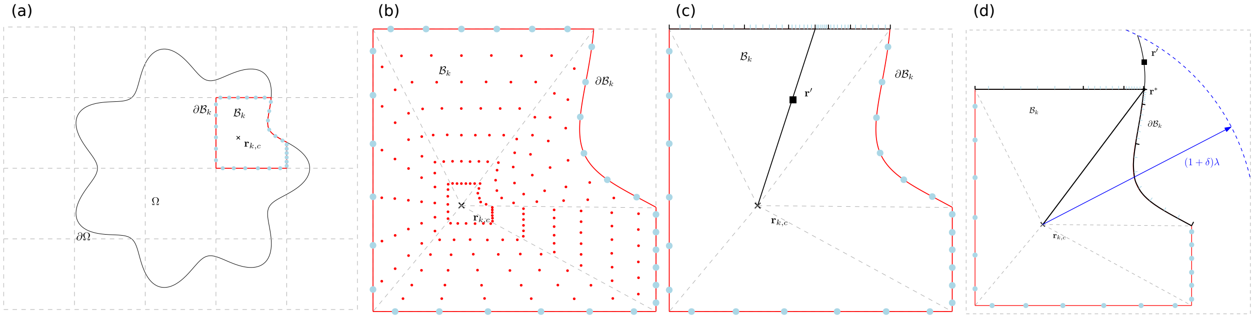

Figure 4.2: (a) volume mesh of a starfish geometry Ω; (b) smooth volume quadrature of an irregular box $\mathcal{B}_k$; (c) boundary panel distribution for singular volume quadrature; (d) boundary panel distribution for nearly-singular volume quadrature

Assume f is given only inside some complex domain, then there are some extra remarks on the quality of volume mesh.

Remark 4.3.3. No additional volume mesh refinement was used in our numerical experiments. While this is generally true to achieve certain accuracy if we keep reducing size of $\hat{\mathcal{B}}_k$ for fixed geometry, irregular boxes $\hat{\mathcal{B}}_k \cap \Omega$ of weird ratio can always happen for problems with moving geometries. A more advanced algorithm to generate volume mesh adaptively is needed. This could be handled by inspecting geometry properties of each irregular box itself, and is independent of Ω^c. In other words, no extension on the forcing term is required, and this usually can be fixed without dramatic mesh refinement.

Remark 4.3.4. Volume quadrature nodes need to be inside the domain. That is $\{t_\ell \boldsymbol{r}_j | \, \boldsymbol{r}_j \in \partial\mathcal{B}_k, \, j \le M; \, t_\ell \text{ G-L on } [0,1], \ell \le p\} \subset \mathcal{B}_k$, and $\cup\mathcal{B}_k = \Omega$. In all of our numerical experiments, we assume Ω has been resolved by $\cup\mathcal{B}_k$, so that if we use centroid of each box $\mathcal{B}_k$ as reference, then $t\partial\mathcal{B}_k \subset \mathcal{B}_k$ for $t \in [0,1]$. This usually can be done with a reasonably refined uniform volume mesh.

Remark 4.3.5. We also avoid the case when $\partial\Omega$ crosses one side of $\partial\hat{\mathcal{B}}_k$ multiple times. This could be an indication of whether the complex geometry Ω has been fully resolved.

Although the quadrature algorithm does not put any extra condition on how $\partial\Omega$ interacts with $\hat{\mathcal{B}}_k$. To keep track of $\partial\hat{\mathcal{B}}_k$, we found it useful to enforce this condition when generating a volume mesh, especially when a parametric description of $\partial\hat{\mathcal{B}}_k$ is needed, or to compute cusp locations beforehand.

Remark 4.3.6. Whether Ω is complex or not, it is almost unavoidable to have irregular box $\mathcal{B}_k = \hat{\mathcal{B}}_k \cap \Omega$ of volume being only a tiny fraction of a regular box $\hat{\mathcal{B}}_k$. In our numerical experiments, this does not affect the accuracy and stability by much.

Boundary discretization of $\partial\mathcal{B}_k$

For smooth kernels like Gaussian, we use Gauss-Legendre panel quadrature to discretize $\partial\mathcal{B}_k$ on irregular box $\mathcal{B}_k$, as shown in Fig 4.2(b). Each side of $\partial\mathcal{B}_k$ has one panel, and $t \in [0, 1]$ is then discretized into one panel, both with panel order $2p$ for better accuracy. (On regular square box, we use standard $p \times p$ product Chebyshev quadrature.)

For Laplace single layer kernel, $L_\alpha(\boldsymbol{r}', \boldsymbol{r})$ will develop a cusp on $\partial\Omega$. Boundary discretization $\{\boldsymbol{r}_j\}$ changes according to the location of target $\boldsymbol{r}'$. Depending on the target point $\boldsymbol{r}'$ location (Fig 4.2(c)), it is required to add additional panel distribution clustering around the intersection point $\boldsymbol{r}^*$ to resolve the cusp. Finding the primitive variable is done by building a high order inverse map from phase angle of $\boldsymbol{r}'$ to parametric space $\partial\mathcal{B}_k^{-1}(\boldsymbol{r}^*)$, which is built on a bisection root finding in parametric space for a order p Gauss-Legendre nodes in phase angle.

4.3.3 Approximation

We first scale all boxes $\mathcal{B}_k$ to be $[-1, 1]^2$, in the case of $\mathcal{B}_k$ is irregular, the scaling factor is chosen based on $\hat{\mathcal{B}}_k$. For an irregular box, we further use the smooth volume quadrature to compute its centroid, and shift $\mathcal{B}_k$ accordingly so that $(0, 0)$ becomes its new centroid for convenience.

On regular box $[-1,1]^2$, we use product of one-dimension pth order Chebyshev nodes to sample $f(\boldsymbol{r})$, and compute its approximation.

$$f(x,y) \approx \sum_{n=0}^{p-1}\sum_{m=0}^{p-1} \hat{c}_{n,m} T_n(x) T_m(y) \tag{4.22}$$

where $T_n(x) = \cos\left(n \arccos\left(x\right)\right)$.

On irregular box $\mathcal{B}_k$, deriving p^2 approximation coefficients $\{\hat{c}_{n,m}\}$ for the product of Chebyshev basis won't follow generalizations of one dimensional slice. Taking advantage of our accurate high order volume quadrature scheme for smooth kernel in Section 4.3.1, we solve a linear system formed by applying volume integral (a bilinear form $a(\cdot,\cdot)$) to Chebyshev polynomials on $\mathcal{B}_k$ to get approximation coefficient. Firstly, we use product of Chebyshev polynomials $T_n(x)T_m(y)$ as basis function on irregular box $\mathcal{B}_k$ of any shape. Then we impose a natural bilinear form for integrable function on $\mathcal{B}_k$

$$a\left(\phi(x,y), \psi(x,y)\right) = \int_{\mathcal{B}_k} \phi(x,y)\psi(x,y)\,\mathrm{d}A \tag{4.23}$$

$a(\phi(x,y),\psi(x,y))$ can be computed accurately using the high order smooth volume quadrature we developed. We then enforce the condition that the difference between $f(\boldsymbol{r})$ and its approximation is orthogonal to all basis functions on $\mathcal{B}_k$:

$$a\left(f(x,y) - \sum_{n=0}^{p-1}\sum_{m=0}^{p-1} \hat{c}_{n,m} T_n(x) T_m(y), T_{n_0}(x) T_{m_0}(y)\right) = 0, \quad n_0, m_0 < p \tag{4.24}$$

The principle is that it should make no difference whether we evaluate volume integral of $f(x,y)$ or its approximation $\sum_{n=0}^{p-1}\sum_{m=0}^{p-1} \hat{c}_{n,m} T_n(x) T_m(y)$, when they are convolved with some polynomial of order less than p.

Since analytic 1-forms $I_\alpha\left(\boldsymbol{r}', \boldsymbol{r}, \delta\right)$ and $L_\alpha(\boldsymbol{r}', \boldsymbol{r})$ are derived for monic polynomials. All coefficients $\{\hat{c}_{n,m}\}$ for Chebyshev polynomials are further mapped to coefficients for product of monic polynomials, denoted as $\{c_{n,m}\}$.

Remark 4.3.7. $T_n(x)T_m(y)$ are no longer orthogonal polynomials on irregular $\mathcal{B}_k$. As a result, we need to solve a linear system to get approximation coefficients. This is not the case on regular boxes. An analogue of Gram-Schmidt process could be worked out if stability issue ever comes up. In our numerical experiments, the approximation scheme works very well to achieve 10 digits accuracy in computing $u(\boldsymbol{r}')$ when coupled with the volume quadrature scheme, and it is robust to various irregular box shapes when order $p \leq 10$.

Remark 4.3.8. Another remark we want to make is that for simplicity reasons, we did not explore the effects of weight functions ($\rho = 1$ for all our numerical experiments). Therefore even in the case of $\mathcal{B}_k$ deforming to regular box, the approximation coefficients wouldn't be exactly the same as coefficients from regular box.

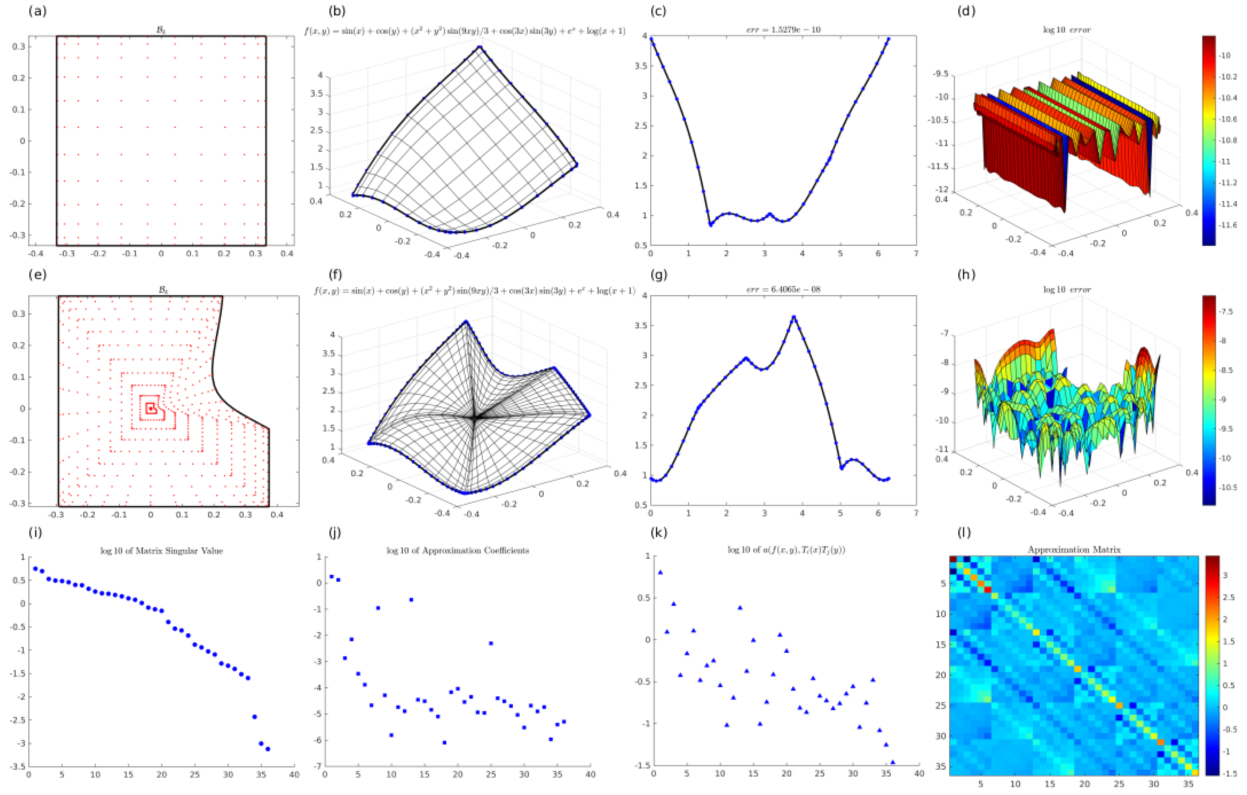

Figure 4.3: Approximation scheme on regular and irregular boxes.

4.4 Numerical Implementation

Our proposed volume potential evaluation scheme combines a hybrid function approximation scheme (Chebyshev on regular box, matching inner product on irregular box) with a hybrid layer potential evaluation scheme (fixed numerical quadrature for far evaluation/smooth kernel, semi-analytical quadrature for nearly-singular and singular kernel/sharply peaked Gaussian). In this section, we summarize the steps to implement our numerical evaluation scheme.

Turning to the structure of implementing the solver for Poisson's equation, Section 4.4.1 demonstrates the volume integral solver to get a particular solution for Poisson's equation (u^P in Section 4.2.2). In Section 4.4.2, we present an efficient and accurate general boundary integral solver for Laplace equation see [80][211] (u^P in Section 4.2.2). We briefly explain the coupling with FMM to enable an $\mathcal{O}(N)$ complexity computational framework in Section 4.4.3, where N is the number of domain discretization nodes.

In Section 4.5, we conduct numerical experiments on both interior and exterior problems. Since the volume integral solver is comprised of summing up layer potential contributions from all boxes $\{\mathcal{B}_k\}$, they fall into the evaluation of either a local regular or an irregular patch for both interior and exterior problems. We restrict our discussion to an interior configuration as shown in Fig.4.2, with the understanding that the volume solver works exactly the same for exterior problems of interest, assuming Ω is bounded.

4.4.1 Formal description of the volume integral solver

In this section, we describe the implementation of the volume integral solver for the particular solution $u^P(\boldsymbol{r}')$ from (4.8).

Stage 1: domain and boundary discretization As discussed in Section 4.3.2, given a two-dimensional domain Ω, and mesh size, we start with a uniform grid of square boxes $\{\hat{\mathcal{B}}_k\}$

containing Ω. Then the intersection sets $\{\mathcal{B}_k\} = \{\hat{\mathcal{B}}_k \cap \Omega\}$ which shares a fraction of $\partial\Omega$ as part of its boundary, as shown in Fig.4.2(a), is our volume mesh.

To apply the contour integral on $\partial\mathcal{B}_k$, we define a parametric representation of $\partial\mathcal{B}_k$ analytically by gluing together sides of regular boxes and the shared piece of $\partial\Omega$ in a counterclockwise way ($\partial\mathcal{B}_k$ is colored in red as illustrated in Fig.4.2(b)).

At this stage, we set up the fixed smooth volume quadrature on irregular box $\mathcal{B}_k$ (on regular box, we use product of pth order Chebyshev quadrature rule). Assume $\{(t_\ell, w_\ell)\}_1^p$ denotes the pth order Gauss-Legendre quadrature nodes and weights on $[0,1]$; $\{(\boldsymbol{r}_j, \mathsf{w}_j, \boldsymbol{\tau}_j)_{\partial\mathcal{B}_k}\}_1^{pN_k}$ denotes the entire set of panel quadrature nodes, weights and the corresponding tangent vectors on $\partial\mathcal{B}_k$, where N_k is the number of sides of $\mathcal{B}_k$, then we have the volume quadrature nodes and weights (We drop the subscript $\partial\mathcal{B}_k$ in all the following notations of $(\boldsymbol{r}_j, \mathsf{w}_j, \boldsymbol{\tau}_j)_{\partial\mathcal{B}_k}$ without ambiguity):

$$\{(t_\ell \boldsymbol{r}_j, t_\ell w_\ell \mathsf{w}_j |\boldsymbol{r}_j \times \boldsymbol{\tau}_j|)\}_{\ell,j} \tag{4.25}$$

Here we assume each box $\mathcal{B}_k$ is rescaled and uses a local coordinate system, with its centroid being the origin, as discussed in the approximation scheme in Section 4.3.3. This is essentially shrinking $\partial\mathcal{B}_k$ according to the centroid to get a smooth volume quadrature.

Stage 2: source approximation To form an approximation of source distribution function $f(\boldsymbol{r})$ on irregular $\mathcal{B}_k$ with no extension outside Ω. We use product of one-dimensional Chebyshev polynomials $\{\eta_\alpha(\boldsymbol{r})\} = \{T_n(x)T_m(y)\}$ as our basis for both regular and irregular boxes, where $\alpha = np + m$.

Using smooth volume quadrature (4.25), and inner product defined by bilinear form (4.23) on $\mathcal{B}_k$, we set up the $p^2 \times p^2$ Gram matrix A on $\mathcal{B}_k$:

$$A_{\alpha\beta} = a\left(\eta_\alpha(\boldsymbol{r}), \eta_\beta(\boldsymbol{r})\right) \approx \sum_{\ell=1}^{p} \sum_{j=1}^{pN_k} \eta_\alpha(t_\ell \boldsymbol{r}_j)\, \eta_\beta(t_\ell \boldsymbol{r}_j) t_\ell w_\ell \mathsf{w}_j |\boldsymbol{r}_j \times \boldsymbol{\tau}_j|, \tag{4.26}$$

and the $p^2 \times 1$ right hand side vector g, where

$$g_\alpha = a\left(\eta_\alpha(\boldsymbol{r}), f(\boldsymbol{r})\right) \approx \sum_{\ell=1}^{p} \sum_{j=1}^{pN_k} \eta_\alpha(t_\ell \boldsymbol{r}_j)\, f(t_\ell \boldsymbol{r}_j) t_\ell w_\ell \mathsf{w}_j |\boldsymbol{r}_j \times \boldsymbol{\tau}_j|.$$

We then get the approximation coefficients $\hat{c}$ by solve this linear system $\hat{c} = A^{-1}g$, s.t. $f(\boldsymbol{r}) \approx \sum_\alpha \hat{c}_\alpha \eta_\alpha(\boldsymbol{r})$. These coefficients $\{\hat{c}_\alpha\}$ are then transformed into coefficients $\{c_\alpha\}$ for monomial basis.

To get coefficients for product of Chebyshev polynomial on a regular box, we sample $p \times p$ product Chebyshev nodes on each square patch, and then apply transformation $\frac{2}{p}\cos\left(\frac{\pi((1{:}p)-1)'(2(p{:}-1{:}1)-1)}{2p}\right)$ on $p \times p$ sampled density $f(\boldsymbol{r})$ from both dimensions.

Stage 3: volume potential evaluation For smooth kernels like Gaussian and far interaction of the Laplace kernel, computing the volume potential is the same as computing inner product in Stage 2 using a smooth volume quadrature. For close interaction of the Laplace kernel or sharply peaked Gaussian, i.e. for target points $\boldsymbol{r}'$ within a $(1+\delta)\lambda$ ball with respect to the centroid of $\mathcal{B}_k$ as shown in Fig 4.2(d), the conversion from volume integral to boundary integral using $L_\alpha(\boldsymbol{r}', \boldsymbol{r})$ and $I_\alpha(\boldsymbol{r}', \boldsymbol{r}, \delta)$ in Section 4.3 is what enables our semi-analytical layer potential evaluation scheme.

For sharply peaked Gaussian, its thin support gets projected to $\partial\mathcal{B}_k$, where 1-form $I_\alpha(\boldsymbol{r}', \boldsymbol{r}, \delta)$ needs to be integrated. For Laplace kernel, the logarithmic singularity gives rise to a cusp along $s\boldsymbol{r}'$ (Lemma 4.3.4), which will intersect $\partial\mathcal{B}_k$ at some point denoted as $\boldsymbol{r}^*$. This means for each new target $\boldsymbol{r}'$, we must rebuild a panel discretization to adapt to this intersection $\boldsymbol{r}^*$. Let $\{(\tilde{\boldsymbol{r}}_j, \tilde{\mathsf{w}}_j, \tilde{\boldsymbol{\tau}}_j)\}_1^{p\tilde{N}_k}$ denotes the new set of panel quadrature and tangent vectors on $\partial\mathcal{B}_k$ associated with target $\boldsymbol{r}'$, as shown in Fig.4.2(c) ($\tilde{N}_k$ equals N_k plus additional panels required to resolve cusp; we drop the subscript $\partial\mathcal{B}_k$). Then we compute 1-forms $L_{n+m+1}(\boldsymbol{r}', \tilde{\boldsymbol{r}}_j)$, associated with monomial basis element $x^n y^m$, using the recursive relation (4.15).

Finally, we compute

$$\begin{aligned}\int_{\mathcal{B}_k} \log\left(|\boldsymbol{r}' - \boldsymbol{r}|^2\right) x^n y^m \,\mathrm{d}A &= \oint_{\partial\mathcal{B}_k} L_{n+m+1}\left(\boldsymbol{r}', \boldsymbol{r}\right)\left(-x^n y^{m+1}\,\mathrm{d}x + x^{n+1} y^m \,\mathrm{d}y\right) \\ &\approx \sum_{j=1}^{p\hat{N}_k} L_{n+m+1}\left(\boldsymbol{r}', \tilde{\boldsymbol{r}}_j\right) \tilde{x}_j^n \tilde{y}_j^m \tilde{\mathsf{w}}_j |\tilde{\boldsymbol{r}}_j \times \tilde{\boldsymbol{\tau}}_j|,\end{aligned} \tag{4.27}$$

and combine them using coefficient c from Stage 2 to get $u^P(\boldsymbol{r}')$. For sharply peaked Gaussian, the same step follows with the flexibility of no need to compute the precise center of its projected support on $\partial\mathcal{B}_k$. Though we are integrating around a collection of adjacent contours, there won't be cancellation on the boundaries of adjacent $\mathcal{B}_k$, unless the density approximation is constructed under the same local coordinate system.

Stage 4: build singular and nearly-singular volume potential evaluation matrix (optional)

By constructing the volume potential evaluation matrix, computing a particular solution for multiple source function $f(\boldsymbol{r})$ within a fixed domain becomes a problem of local sparse matrix vector product, then coupled with FMM for far interaction. On a regular or irregular box $\mathcal{B}_k$, we follow the volume potential evaluation procedure described in previous stage to build the matrix of singular and nearly-singular volume potential contribution from every monic polynomial in the basis for all requested target points close to $\mathcal{B}_k$. Then depending on whether the box is regular or irregular, a matrix representation of the approximation scheme can be worked out, which maps sampled $f(\boldsymbol{r})$ from p^2 Chebyshev nodes or the smooth volume quadrature nodes on irregular box to p^2 approximation coefficients $\{\hat{c}_{n,m}\}$. On regular boxes, the volume potential matrix is formed by the multiplication of the evaluation matrix from monic polynomials to close targets, the map from $\{\hat{c}_{n,m}\}$ to $\{c_{n,m}\}$, and the approximation matrix using Chebyshev polynomials. On irregular boxes, we keep the approximation matrix separated from the other two. This allows forming evaluation matrix of column size p^2 for both regular and irregular boxes. And in our experiments, we discard coefficients with magnitude less than 10^{-9} on irregular boxes.

Similar to the close evaluation scheme we developed for Laplace layer potential in Chapter III, the product integration scheme for two-dimensional volume potential share a similar computational complexity. For each target, $\mathcal{O}(p)$ moments are evaluated via recurrences for $\mathcal{O}(p)$ sources on $\partial\mathcal{B}_k$. Forming and inverting the approximation matrix both cost $\mathcal{O}(p^6)$, which is independent of number of targets. Overall, the computational cost of the scheme is $\mathcal{O}(p^6 + n_{targ}p^2)$, where n_{targ} is the number of targets that are considered to be close to $\mathcal{B}_k$.

4.4.2 Formal description of the boundary integral solver

In this section, we briefly overview the boundary integral solver for the evaluation of homogeneous solution $u^H(\boldsymbol{r}')$ from (4.9). For a more thorough implementation of the product integration rules, we refer the reader to [80][211] and references cited in these papers.

We form a uniform panel distribution of pth order Gauss-Legendre quadrature rule, denoted as $\{(\boldsymbol{r}'_j, w'_j)_{\partial\Omega}\}$, for the homogeneous problem (4.9) when $\partial\Omega$ is smooth, and an adaptive panel distribution when $\partial\Omega$ has corners. Using the volume potential results from previous Stage on $\boldsymbol{r}'_j$, we then enforce the corrected boundary condition for the homogeneous problem:

$$u^H\left(\boldsymbol{r}'_j\right) = g\left(\boldsymbol{r}'_j\right) - u^P\left(\boldsymbol{r}'_j\right) \tag{4.28}$$

We then consider a single layer potential ansatz:

$$u^H(\boldsymbol{r}'_j) = \mathcal{S}[\tau](\boldsymbol{r}'_j) = -\frac{1}{2\pi}\int_{\partial\Omega} \log\left(|\boldsymbol{r}'_j - \boldsymbol{r}|\right)\, \tau(\boldsymbol{r})\, \mathrm{d}s_{\boldsymbol{r}} \tag{4.29}$$

By enforcing the boundary condition $\mathcal{S}[\tau](\boldsymbol{r}'_j) = u^H\left(\boldsymbol{r}'_j\right)$, we can solve for unknown density τ. If $\boldsymbol{r}'_j$ is far away from a source panel, the smooth layer potential evaluation is approximated by the Gauss-Legendre quadrature rule. For target $\boldsymbol{r}'_j$ that fall close to a source panel, the integral kernel is nearly-singular, therefore we adapt a panelwise close evaluation scheme proposed in [80], where the density function μ is approximated by a high-order polynomial interpolation in the complex plane, and the exact integrals of the single layer kernel against these polynomial basis functions are worked out via a two-term recurrence.

4.4.3 Couple singular and nearly-singular volume integral scheme with FMM

In this section, we describe the coupling of our local singular volume potential evaluation scheme with Fast Multipole Method (FMM) [64], and refer the reader to [211] on the integration of singular layer potential evaluation scheme into the FMM framework. This completes the description of the $\mathcal{O}(N)$ fast Poisson solver.

We call target point $\boldsymbol{r}'$ and $\mathcal{B}_k$ "close" if $d\left(\boldsymbol{r}', \boldsymbol{r}_{k,c}\right)$, the distance between $\boldsymbol{r}'$ and the centroid $\boldsymbol{r}_{k,c}$ of $\mathcal{B}_k$, is smaller than some prescribed tolerance $(1+\delta)\lambda$ (we use $\delta = 0.4$ in our experiments), where λ is the maximum distance between vertices of $\mathcal{B}_k$ and $\boldsymbol{r}_{k,c}$. For these target points, the Laplace kernel $G\left(\boldsymbol{r}', \boldsymbol{r}\right)$ becomes singular or nearly-singular, we use the semi-analytic product integration scheme described in Section 4.4.1. As for targets outside this close disc, but within about grid size distance to the center of $\hat{\mathcal{B}}_k$, we use an upsampling of the smooth volume quadrature to improve accuracy, when building the evaluation matrix. For targets well-separated from $\mathcal{B}_k$, both the kernel function and the source density are smooth, therefore the contribution from product Chebyshev quadrature or smooth quadrature (4.25) can be rapidly computing using FMM.

To summarize, a Nyström method is used to approximately compute $u^P(\boldsymbol{r}')$, which results in a matrix-vector multiplication. When the problem size is large, the Nyström matrix is never formed explicitly. Smooth quadrature rule, i.e. product Chebyshev quadrature and (4.25), are accelerated by FMM to compute interaction from all volume source points to all target points. Then a local correction term is added to $u^P(\boldsymbol{r}')$ using a sparse singular or nearly-singular matrix vector multiplication.

4.5 Numerical results and discussion

In this section, we report the numerical accuracy and performance results for a series of tests to validate the overall solution scheme. First, we investigate the convergence of the approximation scheme presented in Section 4.3.3 in the volume on geometries similar to Fig 4.2. We then perform self-convergence test on Gaussian kernel in two and three dimensions. Lastly, we test the performance of the Poisson solver on smooth and polygonal domains, and use FMM for applying the volume potential matrix to extend it to a multiple inclusion setting. We mention that the domain and boundary discretization for computing the volume and layer potentials in our numerical experiments are both uniform, except for boundaries with corner. Although the homogeneous part of the solution using boundary integral formulation achieves the expected order of convergence, we need to inspect the quality of the uniformly generated volume mesh in order to perform a similar convergence test on the particular solution, i.e. the size of box is chosen for better convergence results.

1. This example tests our approximation scheme in Section 4.3.3 on a smooth starfish-shaped domain defined by function $Z(t) = (1 + 0.3\cos 5t)\,[\cos t, \sin t]^T$. We first choose some arbitrary function $f(\boldsymbol{r}) = \sin(3x + 0.1) + \cos(4y + 0.2) + \sin(5\,(x - 0.1)\,(y - 0.2))$ as the test source density. Then we discretize Ω uniformly for different mesh size h and perform the approximation scheme for various order p. We report the relative L^∞ error measured on a 50×500 polar target grid.

   This experiment shows that our approximation scheme is able to achieve $7 - 8$ digit accuracy as we refine the volume mesh. We would like to point out that the approximation scheme is built upon the idea to match inner product in the volume integral sense, therefore a steady pointwise error uniformly inside Ω indicates that on each box $\mathcal{B}_k$, the absolute error of a smooth volume integral will decrease as the area of the box decreases. Also we observe that when using the smooth quadrature to compute $\int_\Omega \mathrm{d}A$, the area of Ω, the accuracy is around $11 - 12$ digits.

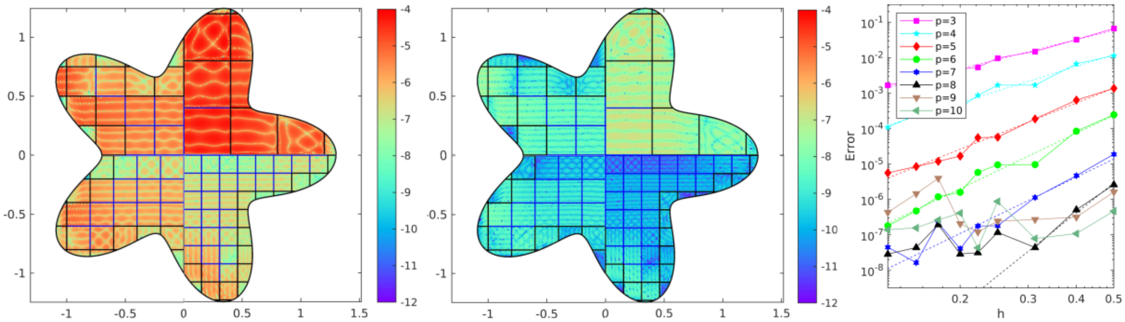

Figure 4.4: Relative L^∞ Error: (Left) $p = 5$; (Middle) $p = 7$; (Right) convergence

2. Here, we first consider the two dimensional Gaussian kernel for $\delta = 1/100$ and $\delta = 1/10000$, $f(\boldsymbol{r}) = \sin x + \cos y + 2\cos\left(\sqrt{2}x\right) y^2 - 2\cos\left(\sqrt{2}x\right) + \sin\left(\sqrt{3}x^2 y\right)\left(3x^4 + 12x^2y^2\right) - 2\sqrt{3}\cos\left(\sqrt{3}x^2y\right) y$. We report the convergence using absolute error for comparison, since $u(\boldsymbol{r}')$ is much smaller for $\delta = 1/10000$. Top three figures in Fig 4.5 shows the result using a numerical quadrature as in Fig 4.2(b) when $\delta = 1/100$. Bottom three figures in Fig 4.5 shows the result using product integration scheme on a boundary panel discretization similar to Fig 4.2(c) but without panels adapting to cusp, when $\delta = 1/10000$.

   The error plot for $\delta = 1/100$ using a smooth volume quadrature behaves as expected. However, the error plot and acceleration of convergence rate in the convergence plot for $\delta = 1/10000$ suggests that for even thinner support of Gaussian, the support of the resulting 1-form won't be resolved with a fixed boundary panel quadrature on $\partial\mathcal{B}_k$, which could be addressed in a cusp-like way as in the singular volume potential case.

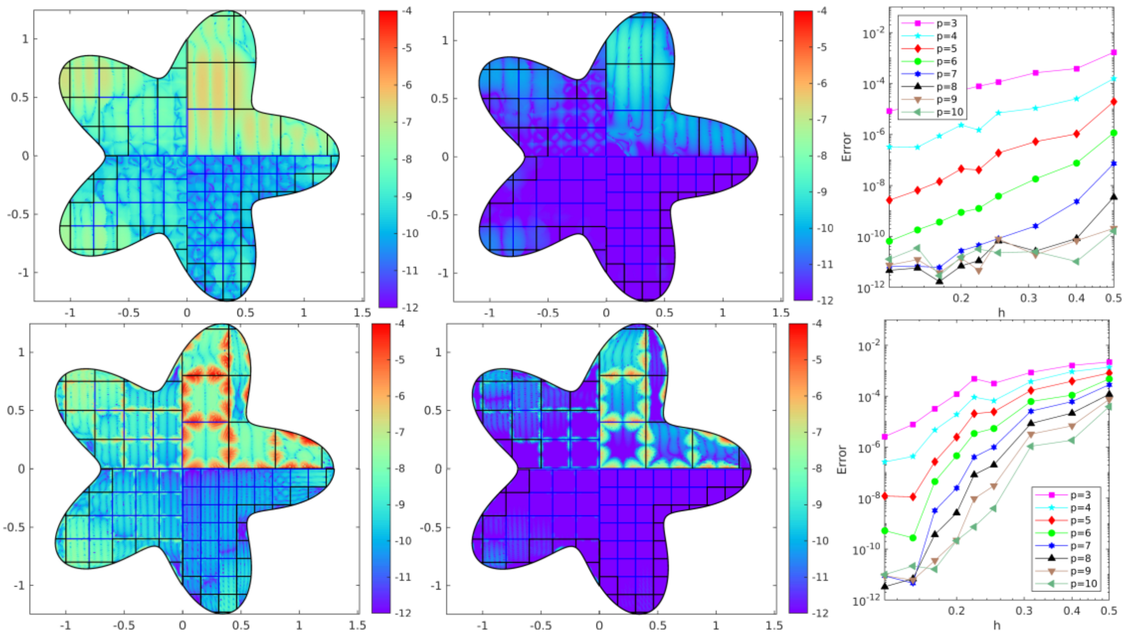

Figure 4.5: (1st row): $\delta = 1/100$ Absolute Error (max abs value 0.1784): (Left) $p = 5$; (Middle) $p = 7$; (Right) convergence. (2nd row): $\delta = 1/10000$ Absolute Error (max abs value 3.2643×10^{-3}): (Left) $p = 5$; (Middle) $p = 7$; (Right) convergence.

Lastly, we briefly test the feasibility and performance of the scheme on three dimensional Gaussian kernel for $\delta = 1/10$. As shown in Fig 4.6, the spheroid is divided into 8 irregular boxes according to their octants. Unlike Laplace kernels, both two dimensional and three dimensional Gaussians share the same kernel structure, which allows the semi-analytic product integration scheme to be easily extended to three dimensions. Here, $f(\boldsymbol{r})$ is taken to be the summation of product of monic polynomials with order less or equal than 4, so that no three dimensional approximation scheme is needed.

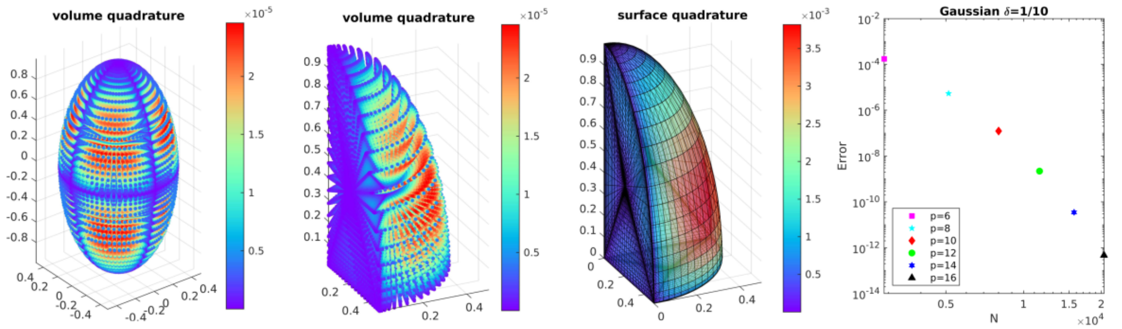

Figure 4.6: Volume quadrature and error for three dimensional Gauss transform

3. We now solve the Poisson equation (4.7) for $u_e(\boldsymbol{r}) = \sin x + \cos y + \cos\left(\sqrt{2}x\right) y^2 + \sin\left(\sqrt{3}x^2 y\right)$. We assume $f(\boldsymbol{r})$ is only known within Ω. The left column in Fig 4.7 shows smooth volume quadrature nodes and weights for different geometries. Comparing to exact solution $u_e(\boldsymbol{r})$, the numerical solution we get from our Poisson solver is about 10 digit accuracy in both smooth and polygonal domains, without using any extension on $f(\boldsymbol{r})$ and drastic refinement on the volume mesh.

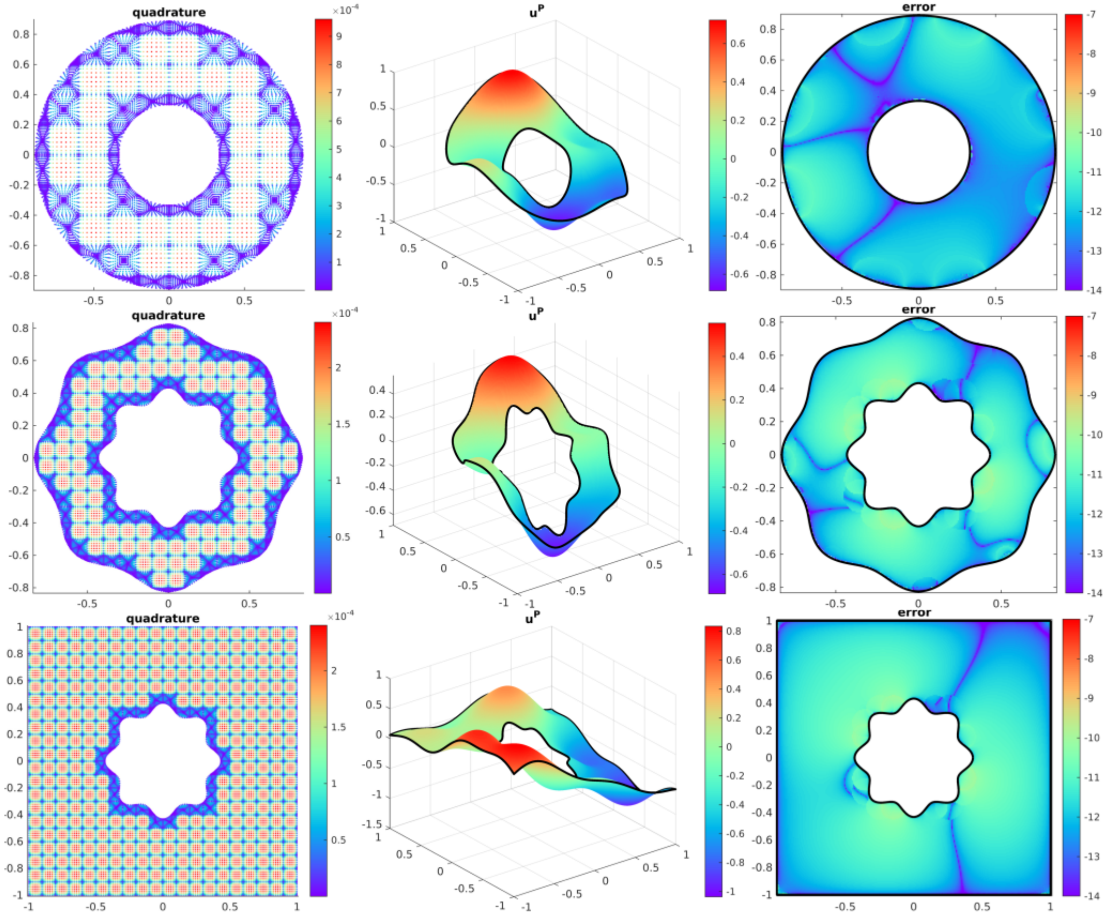

Figure 4.7: p=10 with 250 × 250 resolution (a) box size 0.2; (c) max relative error 2.1074e-11; (d) box size 0.1; (f) max relative error 1.0017e-10; (g) box size 0.2; (i) max relative error 9.9970e-11.

4. Finally, we solve the Poisson equation (4.7) on a Ω that has a boundary of a collection of 94 interior smooth islands plus 1 exterior square. These islands were generated randomly using a limited range of parameters, and then perturbed to ensure the quality of the

volume mesh. Here, we use $u_e(\boldsymbol{r}) = \sin(x) + \cos(y + 0.1) + \sin(xy) + \cos(x)\sin(y)$. And we report the time for applying the correction is less than 1 percent comparing to applying FMM under the current problem settings.

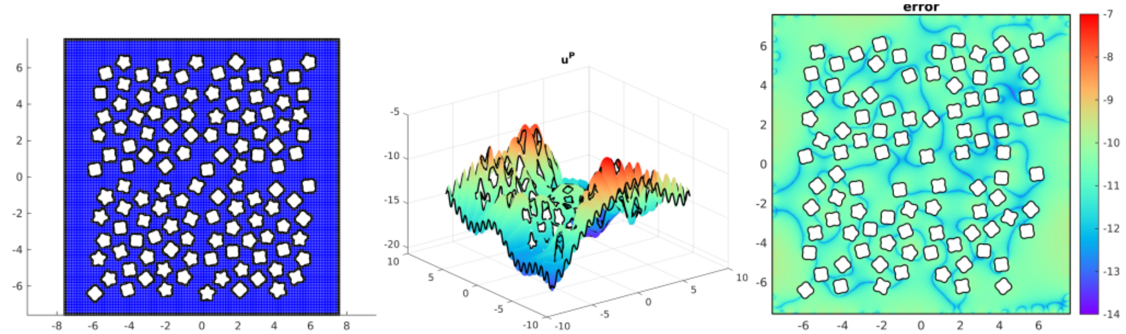

Figure 4.8: 94 islands, 17336 regular boxes, 2896 irregular boxes, and uniform volume mesh size of 0.1.($[-7.6, 7.6]^2$). The number of sources and targets are $N_s = 11718400$, $N_t = 222274$. (c) max relative error 5.5771e-10 (450×450).

4.6 Conclusion

We have demonstrated an accurate and high order product integration scheme for computing the particular solution of Poisson problem via volume potential. The Poisson solver scales linearly with the number of degrees of freedom in the domain. The scheme proceeds by constructing a high order approximation of the forcing function on both regular and irregular boxes, and applying general Stokes' theorem to the singular or nearly singular kernel analytically to convert the domain integral to a line integral on the boundary. We report the performance and error results on a variety of examples. In all numerical experiments, we achieve around 10 digit accuracy. Similarly, we have also considered applying the scheme to Gauss transform in both two and three dimensions.

Clearly, a number of questions arise about the computational scheme outlined here: *The grid size must resolve the geometry; wouldn't it lead to too many boxes? What if non-star shaped irregular boxes are encountered? Wouldn't the irregular boxes be extremely small or have high aspect ratio?* All of these concerns can be addressed by adaptive mesh refinement and some variation of overset grids, implementing which is left to future work. Adaptive mesh refinement, say based on quad tree representations in 2D and octrees in 3D, is sufficient to address all the above questions. Work is currently in progress towards these direction.

CHAPTER V

Simulating Cilia-driven Mixing and Transport in Complex Geometries

Preamble. Cilia and flagella are self-actuated microtubule-based structures that are present on many cell surfaces, ranging from the outer surface of single-cell organisms to the internal epithelial surfaces in larger animals. In this chapter, we develop a hybrid numerical method that employs a boundary integral method for handling the confining geometries and the constituent rigid particles and the method of regularized Stokeslets for handling the cilia. We provide several examples demonstrating the effects of confining geometries on cilia-generated fluid mixing as well as the cilia-particle hydrodynamics. This is joint work with Hanliang Guo and Shravan Veerapaneni, and is published in [211]. My contribution in this study is developing the boundary integral solver.

5.1 Introduction

Cilia are microscopic hair-like structures that protrude from cell surfaces. Motile cilia possess sophisticated internal structures, generally known as the “9+2” structures (see [52] for more details), that enable active periodic ciliary beatings. Being one of the most preserved structures in nature, cilia can be found in almost every phylum in the animal kingdom, from unicellular eukaryotes to invertebrate metazoans and vertebrates. Cilia play significant roles in small animals such as unicellular organisms and invertebrate metazoans, including locomotion, generating water currents for feeding, and more [58, 174, 123, 195]. In vertebrates, cilia are mostly found on the epithelial cell surfaces of internal organs, including the respiratory tract, brain, ear and oviduct [5, 86, 201, 46, 45, 140], as the role in locomotion is replaced by muscles.

The beating of individual cilium usually presents an asymmetric pattern that consists of a straight power stroke and a curly recovery stroke. This asymmetric pattern could break the famous "scallop theorem" ([153]) in viscous fluid at the individual level. Additionally, cilia are usually found in dense groups. Each cell could feature hundreds of cilia in the respiratory tracts [179]. Interestingly, healthy cilia usually do not beat either in-phase or completely randomly. Instead, they beat in an orchestrated wavelike fashion: the so-called metachronal waves. Simply put, the metachronal waves are formed by all cilia performing similar beating patterns, but deforming in time with a small phase difference with respect to their neighbors. The metachronal wave is an effective approach to transport fluid, as it breaks the scallop theorem at the collective level.

The study of ciliary/flagellar propulsion of micro-swimmers dates back to 1950s when G.I. Taylor [186] modeled the flagellum of a sperm cell to an infinite sinusoidal traveling wave and studied analytically the relations between the swimming velocity and the wavenumber, the beating amplitude, and the traveling wave velocity. We refer the reader to [113] and [50] for a more detailed review on this topic. Ciliary transport in the airway systems has also received much attention due to its native relations with some human diseases [47, 188]. Numerous studies have been carried out using various numerical methods, including resistive force theory [49], slender body theory [67, 66, 65], immersed boundary method [38, 121, 95], immersed boundary-lattice Boltzmann method [115, 117, 33], finite element method [131], and the method of regularized stokeslet (MRS) [176, 177, 175, 39, 71, 135].

Ciliary mixing, on the other hand, has not been an active area of study until the last

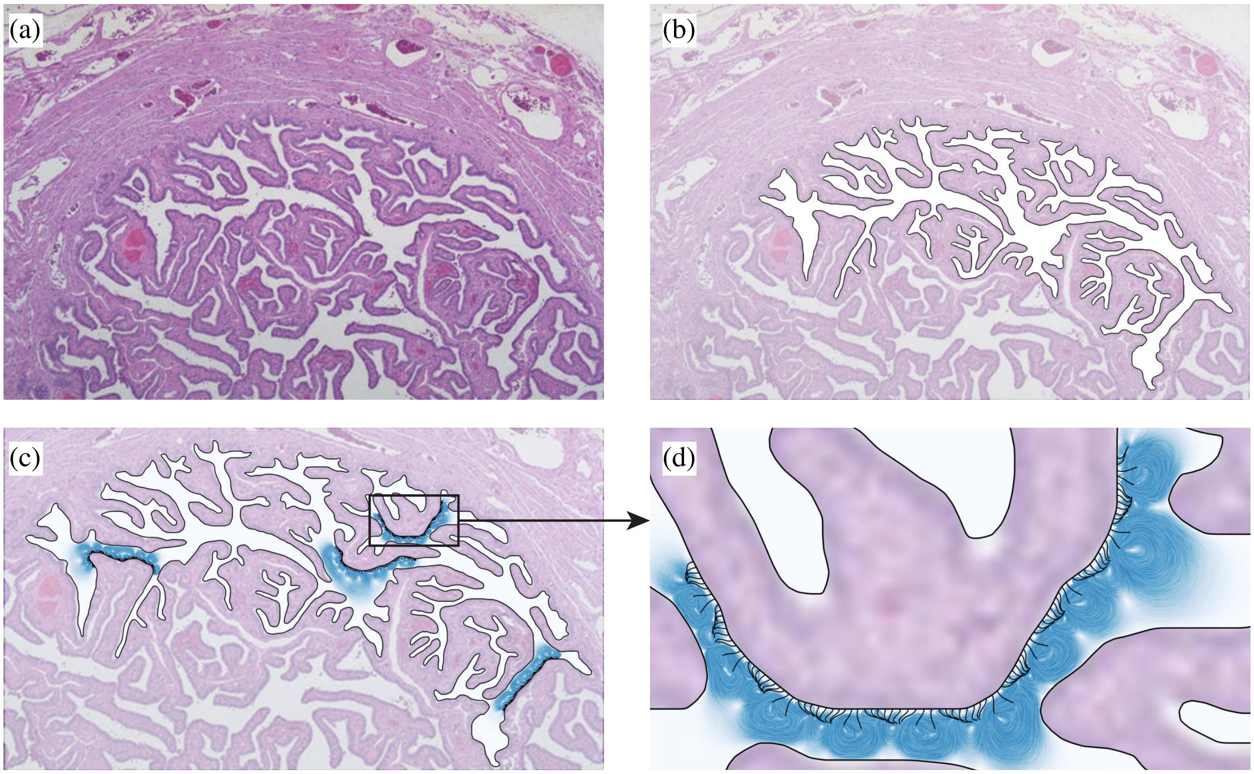

Figure 5.1: An illustration of the current simulation capabilities of the hybrid numerical developed in this chapter. For the purposes of this illustration, we (a) took a generic microscopic image of the cross-section of a Fallopian tube (source: NORM062), (b) extracted a subset of the fluid domain and its boundary highlighted here, (c) seeded four patches of the boundary with around four hundred cilia and solved the governing equations using the numerical method developed in this work and (d) visualized the solution via streamlines in one of the patches. *Note:* This example is for illustrative purposes only and must not be viewed as representative of *in vivo* flows. Physiologically, the major direction of fluid flow in the Fallopian tube is perpendicular to the cross-section as shown here; thereby, a three-dimensional simulation is needed to fully characterize the flows.

decade. Admittedly, mixing in viscous dominant fluid is inherently difficult due to the lack of turbulence [143, 8]. Nevertheless, *in vivo* experiments of zebrafish embryo showed the transition of a unidirectional flow to vortical flows above and below the cilium tip [183], suggesting that the unidirectional transport is far from the sole purpose of the ciliary flow. More recently, [135] showed, using the squid-vibrio symbiotic systems, that long and short cilia that grow on the same ciliated organ serve different functions. Specifically, the long cilia beating with metachronal waves focus on the fluid transport and size-selective functions, while the short cilia with random phase differences enhance the fluid mixing with zero net flow on average. Other works have also been able to show enhanced mixing using artificial cilia (e.g., see [44, 170, 34, 164]). Numerically, [121] studied the fluid mixing generated by a single cilium using the immersed boundary method and found distinct transport region and mixing region higher and lower than the cilium tip, respectively. [39] studied the fluid transport and mixing by a doubly-periodic array of cilia in a half-space bounded by a plane using MRS and its image systems. They systematically vary the phase differences between neighboring cilia and found consistent results in terms of the transport and mixing regions. Their results also showed that metachronal waves enhance not only fluid transport but also mixing. Recent works have also considered transport and mixing of multi-phase fluid in airway systems using the immersed boundary-lattice Boltzmann method [32]. The results are qualitatively similar to that of [39], although tracers in different fluid layers are prevented from mixing due to surface tension effects present at the interface. More recently, [162] developed a large-scale simulation technique making using of the kernel-independent FMM and applied it on dense cilia carpets beating in phase; [181] took a different route, wherein, instead of treating each cilium explicitly, they proposed an elegant coarse-grained model with anisotropic Brinkman equation and solved the cilia-driven transport problem using immersed boundary method.

We note that all of the aforementioned computational works considered simple geometries and idealized boundary conditions such as periodic, free-space or half-space (bounded by a plane wall) flows. While such mathematical simplifications are important for problem tractability, the real environments that cilia beat in are far more complex. For example, [45] showed that the complex flow of cerebrospinal fluid (CBF) in the delicate mice brain ventricles is regulated by the motile cilia; other human organs where cilia play important roles such as the tracheal and the Fallopian tubes present no less complex geometries [19, 114]. Engineering applications such as manufacturing micro-fluidic devices that could transport and/or mix the fluid are also designed to have complex geometries (see, e.g. [100]). To the best of our knowledge, no work to date has been focused on solving the cilia-driven flow in arbitrary complex geometries. Leveraging on recent advances in the boundary integral methods (BIMs) for Stokes flow, in this work, we develop methods for simulating active cilia-driven flow of rigid particles in complex geometries. They are applicable in the regime where the hydrodynamics of the cilia-geometry and cilia-particle interactions are dominated by viscous effects and inertia could be neglected.

Specifically, we use the BIM for solving the Stokes equations inside the confining complex geometry and for evolving the rigid particles and the MRS for simulating the ciliary dynamics. Applying the boundary conditions at the fluid-structure interfaces leads to a set of coupled integro-differential equations at every time-step. We use the recently developed Nyström method in [211] for discretizing the boundary integrals in these equations. This method is both $h-$adaptive and $p-$adaptive; that is, both the size of the boundary panels (h) and the degree of approximation per panel (p) are chosen automatically to achieve a prescribed error tolerance in the solution. Another advantage of this method is that the nearly singular integrals that arise due to the proximity of the walls, cilia and/or rigid particles are computed to high accuracy. A fourth-order explicit Runge-Kutta method is used for evolving both the cilia and the rigid particles.

The current capabilities of this hybrid method are demonstrated by simulating the cilia-driven flow within the planar cross-section of a Fallopian tube as shown in figure 5.1. Note that while the method can handle such arbitrary shapes, for simplicity and ease of analysis, the results section in this chapter only considers flows in relatively classical geometries (e.g., Taylor-Couette channel, see figure 5.2(a)). We emphasize here that such geometries are still difficult to handle using existing methods such as MRS with image systems.

The chapter is organized as follows. We give the problem formulation and describe our numerical solvers in Section 5.2. Analysis of the mixing and transport properties of actuated cilia in complex domains will be presented in Section 5.3, followed by conclusions and future work in Section 5.4.

5.2 Model and methods

In this section, we first describe the problem formulation for the specific case of cilia-driven flow of rigid particles suspended in a Taylor-Couette device. Then, we show how to recast it as a set of mixed boundary integral and discrete equations with unknowns residing on the cilia, particle and wall boundaries only (thus leading to dimensionality reduction). Lastly, we describe a numerical method for discretizing and solving these equations.

5.2.1 Model

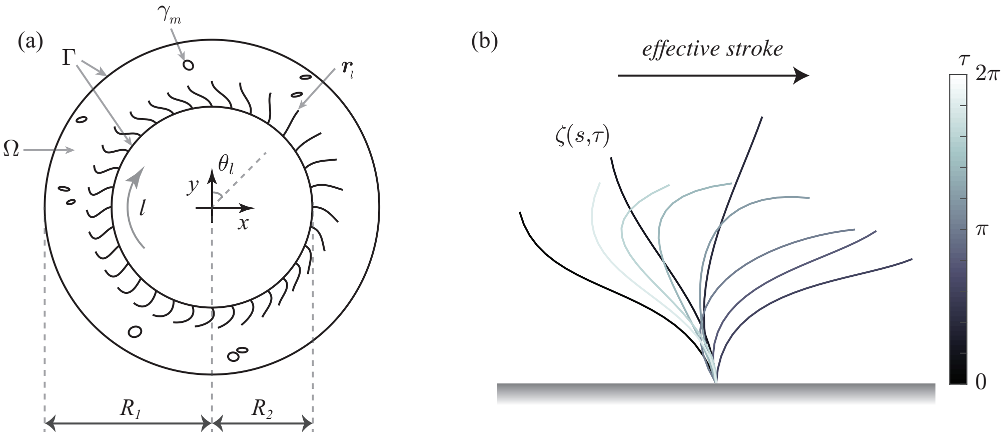

Figure 5.2: Schematic figure. (a) N cilia uniformly distributed at the inner surface of the stationary Taylor-Couette device. N_p particles are freely suspended in the fluid domain Ω bounded by $\{(x,y)|R_2^2 < x^2+y^2 < R_1^2\}$. (b) The snapshots of the beating pattern extracted from [49]. Color-coded by its phase τ.

Consider a thin gap of fluid confined between two stationary concentric circles of radius R_1 and R_2 with $R_1 > R_2$. The fluid domain is denoted by $\Omega = \{(x,y) \mid R_2^2 < x^2+y^2 < R_1^2\}$. N_p rigid particles are immersed in the fluid; N cilia of length ℓ are uniformly distributed on the surface of the inner circle (see figure 5.2(a)). Following [49], the kinematics $\boldsymbol{\zeta}$ of each cilium in its body frame can be approximated by a truncated Fourier series in time τ and a Taylor series in arc-length s. The resulting beating pattern is shown in figure 5.2(b). We apply proper rotations and translations to $\boldsymbol{\zeta}$ to obtain the position of the l-th cilium $\hat{\boldsymbol{r}}$ such that the cilia are uniformly distributed along the inner circle and are oriented perpendicular to the circle. Specifically, the position of the l-th cilium at arc-length s and time t is given by

$$\hat{\boldsymbol{r}}_l(s,t) = \begin{pmatrix} \cos(\theta_l) & \sin(\theta_l) \\ -\sin(\theta_l) & \cos(\theta_l) \end{pmatrix} \boldsymbol{\zeta}(s,\tau_l) + R_2 \begin{pmatrix} \sin(\theta_l) \\ \cos(\theta_l) \end{pmatrix}, \tag{5.1}$$

where $\theta_l = 2\pi(l-1)/N$ and $\tau_l = 2\pi t + (l-1)\Delta\phi$. By construction, the 1st cilium is rooted at $(x,y) = (0,R_2)$ and the index of the cilium increases clockwise. Here $\Delta\phi$ is a constant phase difference between neighboring cilia. Specifically, all cilia beat in synchrony if $\Delta\phi = 0$; $0 < \Delta\phi < \pi$ yields a wave that travels in the opposite direction as the effective stroke (antiplectic waves) and vice versa for $-\pi < \Delta\phi < 0$ (symplectic waves). We note here that the stationary Taylor-Couette device is chosen to construct a "periodic" domain without the periodic kernels.

In the low Reynolds number regime, the fluid dynamics is governed by the non-dimensional incompressible Stokes equation

$$-\nabla p(\boldsymbol{r}) + \nabla^2 \boldsymbol{u}(\boldsymbol{r}) = \boldsymbol{0}, \quad \nabla \cdot \boldsymbol{u} = 0, \quad \forall\, \boldsymbol{r} \in \Omega. \tag{5.2}$$

Here p is the pressure, $\boldsymbol{u}$ is the velocity. Taking advantage of the small aspect ratio of the cilia, we assume that the fluid velocity along the cilia is consistent with the prescribed beating pattern, namely

$$\boldsymbol{u}(s,t) = \frac{\mathrm{d}\hat{\boldsymbol{r}}(s,t)}{\mathrm{d}t}, \quad \text{for } 0 < s \le \ell. \tag{5.3}$$

A no-slip boundary condition is applied on the wall boundaries, that is,

$$\boldsymbol{u}(\boldsymbol{r}) = \boldsymbol{0}, \quad \forall\, \boldsymbol{r} \in \Gamma. \tag{5.4}$$

On the other hand, given the translational velocity $\boldsymbol{U}_m$ and the angular velocity ω_m of the m-th rigid particle $(1 \le m \le N_p)$, a no-slip boundary condition on γ_m implies

$$\boldsymbol{u}(\boldsymbol{r}) = \boldsymbol{U}_m + \omega_m(\boldsymbol{r} - \boldsymbol{r}^c_m)^\perp, \quad \forall\, \boldsymbol{r} \in \gamma_m, \tag{5.5}$$

where $\boldsymbol{r}_m^c$ is the centroid of the particle and the *perp* operator $(\cdot)^\perp$ acts on vectors in $\mathbb{R}^2$ and is defined by $\boldsymbol{r}^\perp = \begin{pmatrix} -y \\ x \end{pmatrix}$.

Since the inertia is negligible in viscous dominant fluid, the particles need to also satisfy the no-net-force and no-net-torque conditions [101]. Particularly, in the absence of external forces and torques, we have

$$\int_{\gamma_m} \boldsymbol{g}(\boldsymbol{r})\,\mathrm{d}s_{\boldsymbol{r}} = \boldsymbol{0} \quad \text{and} \quad \int_{\gamma_m} \boldsymbol{g}(\boldsymbol{r}) \cdot (\boldsymbol{r} - \boldsymbol{r}_m^c)^\perp\,\mathrm{d}s_{\boldsymbol{r}} = 0, \tag{5.6}$$

where $\boldsymbol{g}$ denotes the hydrodynamic traction on the particle boundaries.

Throughout this chapter, we normalize lengths by the typical length of cilia $\ell_c = 20\mu\text{m}$, time by the typical beating period $T_c \approx 1/30\text{s}$, and force by $F_c = \mu\ell_c^2/T_c = 12\text{pN}$, where $\mu = 10^{-3}\text{Pa}\cdot\text{s}$ is the water viscosity.

5.2.2 Cilia-channel interactions

We start by considering the simpler case where there are no rigid particles immersed in the fluid. By virtue of linearity of the Stokes equation, the fluid velocity can be written as

$$\boldsymbol{u}(\boldsymbol{r}') = \boldsymbol{u}^c(\boldsymbol{r}') + \boldsymbol{u}^\Gamma(\boldsymbol{r}'), \quad \forall\ \boldsymbol{r}' \in \Omega, \tag{5.7}$$

where $\boldsymbol{u}^c$ and $\boldsymbol{u}^\Gamma$ represent the disturbance flow due to the cilia and the boundary Γ respectively. We then take a hybrid approach: the governing equations for $\boldsymbol{u}^c$ are solved using MRS whereas those of $\boldsymbol{u}^\Gamma$ are solved using a BIM. The MRS is widely used in flagella-driven flow simulation due to its ease of implementation. The BIM offers an accurate and scalable methodology for Stokes flow simulation within complex geometries. Both approaches follow from our previous work in [71] and [211] respectively.

We use an indirect integral equation formulation, also known as the combined field integral equation formulation [152], for the confining geometry Γ, which begins with an *ansatz* that the velocity is a sum of single and double layer potentials:

$$\boldsymbol{u}^\Gamma(\boldsymbol{r}') = (\mathbf{S}_\Gamma + \mathbf{D}_\Gamma)[\boldsymbol{\sigma}](\boldsymbol{r}'), \quad \forall\ \boldsymbol{r}' \in \Omega, \tag{5.8}$$

where $\boldsymbol{\sigma}$ is an unknown density function, $\mathbf{S}$ and $\mathbf{D}$ are the Stokes single- and double-layer operators respectively, defined as

$$\mathbf{S}_\Gamma[\boldsymbol{\sigma}](\boldsymbol{r}') := \int_\Gamma \boldsymbol{G}(\boldsymbol{r}'-\boldsymbol{r})\boldsymbol{\sigma}(\boldsymbol{r})\mathrm{d}s_{\boldsymbol{r}} \quad \text{and} \quad \mathbf{D}_\Gamma[\boldsymbol{\sigma}](\boldsymbol{r}') := \int_\Gamma \boldsymbol{K}(\boldsymbol{r}'-\boldsymbol{r})\boldsymbol{\sigma}(\boldsymbol{r})\,\mathrm{d}s_{\boldsymbol{r}}. \tag{5.9}$$

The convolution kernels $\boldsymbol{G}$ (Stokeslet) and $\boldsymbol{K}$ (stresslet) are the fundamental solutions to Stokes equations (5.2), given as,

$$\begin{aligned} \boldsymbol{G}(\boldsymbol{r}'-\boldsymbol{r}) &= \frac{1}{4\pi}\left(I\log\frac{1}{|\boldsymbol{r}'-\boldsymbol{r}|} + \frac{(\boldsymbol{r}'-\boldsymbol{r})\otimes(\boldsymbol{r}'-\boldsymbol{r})}{|\boldsymbol{r}'-\boldsymbol{r}|^2}\right), \\ \boldsymbol{K}(\boldsymbol{r}'-\boldsymbol{r}) &= \frac{1}{\pi}\frac{(\boldsymbol{r}'-\boldsymbol{r})\cdot\boldsymbol{n}_{\boldsymbol{r}}}{|\boldsymbol{r}'-\boldsymbol{r}|^2}\frac{(\boldsymbol{r}'-\boldsymbol{r})\otimes(\boldsymbol{r}'-\boldsymbol{r})}{|\boldsymbol{r}'-\boldsymbol{r}|^2} \end{aligned} \tag{5.10}$$

where $\boldsymbol{n}_{\boldsymbol{r}}$ is the outer normal vector of the surface Γ at $\boldsymbol{r}$. By definition, (5.8) satisfies the Stokes equations and what remains is to enforce the no-slip boundary condition (5.4). To simplify the formulation, we reverse the order of the discretization points on the inner circle so that the fluid domain can be considered "interior" to both outer and inner circles. Taking the limit as the target $\boldsymbol{r}'$ approaches Γ from the interior and using standard jump conditions for the layer potentials (e.g., see [111], Chapter 3), we arrive at the following equation:

$$\left(-\frac{1}{2}I + \mathbf{S}_\Gamma + \mathbf{D}_\Gamma\right)[\boldsymbol{\sigma}](\boldsymbol{r}') = -\boldsymbol{u}^c(\boldsymbol{r}'), \quad \forall\ \boldsymbol{r}' \in \Gamma. \tag{5.11}$$

The above is a second-kind integral equation (SKIE) for the unknown density function $\boldsymbol{\sigma}$. The main advantage of SKIEs is that they result in a well-conditioned linear system when discretized. However, this system has a null-space of dimension one in the direction normal to the boundary, which needs to be eliminated via standard techniques (e.g., see [172], we can perturb a rank deficient matrix to a nonsingular one with probability 1). Particularly, we eliminate the null-space by adding the components of the normal vector of each quadrature point in the first column of the resulted discretization matrix. A more popular approach for obtaining SKIE in this context is the completed double-layer formulation of Power and Miranda [151]. We chose the above for simplicity (we will employ the same formulation for particles as well). Note, however, that the right hand side vector $\boldsymbol{u}^c$ in (5.11) is also unknown; we discuss its formulation next.

We discretize each cilium into N_s uniformly placed beads along the arc-length. Specifically, the position of the m-th bead for one cilium is at arclength $s = m\ell/N_s$ with $1 \leq m \leq N_s$. We treat each bead along the cilia as a 2D free-space regularized force. The flow field can then be reconstructed using the method of regularized stokeslet [36]. Assuming the position and strength of the n-th regularized forces are $\boldsymbol{r}_n$ and $\boldsymbol{f}_n$, the flow field generated by such a force distribution is given by

$$\boldsymbol{u}^c(\boldsymbol{r}') = \sum_{n=1}^{NN_s} \widetilde{\boldsymbol{G}}(\boldsymbol{r}' - \boldsymbol{r}_n)\boldsymbol{f}_n, \tag{5.12}$$

where $\widetilde{\boldsymbol{G}}$ is the regularized version of the Green's function $\boldsymbol{G}$ given in (5.10), defined in the Appendix E.5. From here on, a regularized operator will be denoted using the symbol $\widetilde{}$ as above. Since the representation (5.12) can be viewed as a discrete, regularized analogue of the layer potentials (5.9), we will use a similar notation; that is, we let

$$\widetilde{\mathcal{S}}_c[\boldsymbol{f}](\boldsymbol{r}') := \sum_{n=1}^{NN_s} \widetilde{\boldsymbol{G}}(\boldsymbol{r}' - \boldsymbol{r}_n)\boldsymbol{f}_n. \tag{5.13}$$

Since the cilia beating pattern is assumed to be given *a priori*, the force density $\boldsymbol{f}$ is unknown. This will be determined by enforcing the no-slip boundary condition at the cilia-fluid interface (5.3). Together with (5.11), the system of equations for the two unknown densities can then be summarized in the matrix form as

$$\begin{pmatrix} -\frac{1}{2}I + \mathbb{S}_{\Gamma,\Gamma} + \mathbb{D}_{\Gamma,\Gamma} & \widetilde{\mathbb{S}}_{c,\Gamma} \\ \mathbb{S}_{\Gamma,c} + \mathbb{D}_{\Gamma,c} & \widetilde{\mathbb{S}}_{c,c} \end{pmatrix} \begin{pmatrix} \boldsymbol{\sigma} \\ \boldsymbol{f} \end{pmatrix} = \begin{pmatrix} \boldsymbol{0} \\ \frac{\mathrm{d}\boldsymbol{r}}{\mathrm{d}t} \end{pmatrix}. \tag{5.14}$$

Here, with a slight abuse of notation, we denoted the single-layer potential defined on Γ (sources) evaluated at the discrete points on the cilia (targets) by $\mathbb{S}_{\Gamma,c}$. Other operators are defined analogously. Once we solve this matrix equation, we can evaluate the velocity field at any point $\boldsymbol{r}'$ in the fluid domain by using (5.7).

Notice that the operator $\left(-\frac{1}{2}I + \mathbb{S}_{\Gamma,\Gamma} + \mathbb{D}_{\Gamma,\Gamma}\right)$ in the matrix equation (5.14) remains fixed as the cilia beat, since Γ is stationary. Therefore, it is computationally efficient to simply compute its inverse (once discretized) as a precomputation step before time-stepping for the evolution of cilia. In large-scale systems (such as in Figure 5.1), one can accelerate this precomputation, and application of inverse at every time-step, using a low-rank factorization based fast direct solver as done recently for similar problems in [127].

5.2.3 Cilia-channel-particle interactions

We now extend our formulation to include rigid particles suspended in ciliary-driven flow in confining geometries. For notational simplicity, we consider the case where only a single rigid particle with boundary γ is present. In this case, the velocity in the fluid domain can be decomposed into three components due to disturbance flows created by the cilia, the stationary wall and the particle respectively as

$$\boldsymbol{u}(\boldsymbol{r}') = \boldsymbol{u}^c(\boldsymbol{r}') + \boldsymbol{u}^\Gamma(\boldsymbol{r}') + \boldsymbol{u}^\gamma(\boldsymbol{r}'), \quad \forall\, \boldsymbol{r}' \in \Omega. \tag{5.15}$$

Similar to earlier treatment, $\boldsymbol{u}^c$ and $\boldsymbol{u}^\Gamma$ are given by (5.12) and (5.8) respectively. For $\boldsymbol{u}^\gamma$, we use the same ansatz as in (5.8), that is, we write:

$$\boldsymbol{u}^\gamma(\boldsymbol{r}') = (\mathbf{S}_\gamma + \mathbf{D}_\gamma)[\boldsymbol{\sigma}](\boldsymbol{r}'), \quad \forall\, \boldsymbol{r}' \in \Omega. \tag{5.16}$$

The vector density function $\boldsymbol{\sigma}$ defined on γ again is an unknown that needs to be determined by applying the boundary conditions on the particle boundaries. By taking the limit of (5.16) as $\boldsymbol{r}'$ approaches γ from the exterior and applying the rigid body velocity condition (5.5) yields the following BIE,

$$\left(\frac{1}{2}I + \mathbf{S}_\gamma + \mathbf{D}_\gamma\right)[\boldsymbol{\sigma}](\boldsymbol{r}') = -\boldsymbol{u}^c(\boldsymbol{r}') - \boldsymbol{u}^\Gamma(\boldsymbol{r}') + \boldsymbol{U} + \omega(\boldsymbol{r}' - \boldsymbol{r}^c)^\perp, \quad \forall\, \boldsymbol{r}' \in \gamma. \tag{5.17}$$

Again, the above is a SKIE for the unknown $\boldsymbol{\sigma}$ defined on γ. The rigid body translational and rotational velocities $(\boldsymbol{U}, \omega)$ are also unknown *a priori* and need to be solved for by applying the force- and torque-free conditions (5.6) on γ. To do so, we need to evaluate the hydrodynamic traction on γ based on the velocity representation (5.15) which can now be written in its full form as

$$\boldsymbol{u}(\boldsymbol{r}') = \widetilde{\mathbf{S}}_c[\boldsymbol{f}](\boldsymbol{r}') + (\mathbf{S}_\Gamma + \mathbf{D}_\Gamma)[\boldsymbol{\sigma}](\boldsymbol{r}') + (\mathbf{S}_\gamma + \mathbf{D}_\gamma)[\boldsymbol{\sigma}](\boldsymbol{r}'). \tag{5.18}$$

The traction force at γ can be computed using the formula $\boldsymbol{g}(\boldsymbol{r}') = -p(\boldsymbol{r}')\boldsymbol{n}_{\boldsymbol{r}'} + (\nabla \boldsymbol{u}(\boldsymbol{r}') + \nabla \boldsymbol{u}^T(\boldsymbol{r}')) \cdot \boldsymbol{n}_{\boldsymbol{r}'}$, where $\boldsymbol{n}$ is the outward normal to γ. We can avoid computing the derivatives numerically by plugging (5.18) into this formula and evaluating the derivatives of the integral kernels analytically. For notational convenience, we define the velocity vector $\boldsymbol{U}^*$ and the operators $\mathcal{L}$ and $\mathcal{C}$ as

$$\boldsymbol{U}^* = \begin{pmatrix} \boldsymbol{U} \\ \omega \end{pmatrix}, \qquad \mathcal{L}_\gamma \boldsymbol{g} := \begin{pmatrix} \int_\gamma \boldsymbol{g}(\boldsymbol{r}) \mathrm{d}s_{\boldsymbol{r}} \\ \int_\gamma \boldsymbol{g}(\boldsymbol{r}) \cdot (\boldsymbol{r} - \boldsymbol{r}^c)^\perp \mathrm{d}s_{\boldsymbol{r}} \end{pmatrix} \quad \text{and} \quad \mathcal{C}\,\boldsymbol{U}^* := \boldsymbol{U} + \omega(\boldsymbol{r} - \boldsymbol{r}^c)^\perp. \tag{5.19}$$

Since the force- and torque-free conditions (5.6) does not require the pointwise values for the traction, we can utilize the identity that the action of $\mathcal{L}_\gamma$ on the Stokes double-layer potentials in (5.18) produces the zero vector (e.g., see [152, 16]). What remains is to evaluate the traction force due to the single-layer potentials in (5.18). The traction associated to the single layer potential $\mathcal{S}_\gamma[\boldsymbol{\sigma}]$, for example, is given by

$$\mathbf{T}^S_\gamma[\boldsymbol{\sigma}](\boldsymbol{r}') := \boldsymbol{n}_{\boldsymbol{r}'} \int_\gamma \boldsymbol{T}(\boldsymbol{r}' - \boldsymbol{r}) \boldsymbol{\sigma}(\boldsymbol{r}) \mathrm{d}s_{\boldsymbol{r}}, \tag{5.20}$$

where

$$\boldsymbol{T}(\boldsymbol{r}' - \boldsymbol{r}) = -\frac{1}{\pi} \frac{(\boldsymbol{r}' - \boldsymbol{r}) \otimes (\boldsymbol{r}' - \boldsymbol{r}) \otimes (\boldsymbol{r}' - \boldsymbol{r})}{|\boldsymbol{r}' - \boldsymbol{r}|^4}, \tag{5.21}$$

The regularized traction kernel is given in Appendix E.5. Based on these definitions, we can now write the force- and torque-free conditions on the rigid particles as

$$\mathcal{L}_\gamma \left(-\frac{1}{2}\boldsymbol{\sigma}(\boldsymbol{r}') + \mathbf{T}^S_\gamma[\boldsymbol{\sigma}](\boldsymbol{r}') + \mathbf{T}^S_\Gamma[\boldsymbol{\sigma}](\boldsymbol{r}') + \widetilde{\mathbf{T}^S}_c[\boldsymbol{f}](\boldsymbol{r}') \right) = \mathbf{0}, \quad \forall\ \boldsymbol{r}' \in \gamma. \tag{5.22}$$

Therefore, together with (5.17), this equation is sufficient to determine the unknowns $\boldsymbol{\sigma}$ and $\boldsymbol{U}^*$ residing on γ. The coupled system of equations for all the unknowns can now be summarized in the matrix form as

$$\begin{pmatrix} -\frac{1}{2}I + \mathbb{S}_{\Gamma,\Gamma} + \mathbb{D}_{\Gamma,\Gamma} & \mathbb{S}_{\gamma,\Gamma} + \mathbb{D}_{\gamma,\Gamma} & \widetilde{\mathbb{S}}_{c,\Gamma} & 0 \\ \mathbb{S}_{\Gamma,\gamma} + \mathbb{D}_{\Gamma,\gamma} & \frac{1}{2}I + \mathbb{S}_{\gamma,\gamma} + \mathbb{D}_{\gamma,\gamma} & \widetilde{\mathbb{S}}_{c,\gamma} & -\mathcal{C} \\ \mathbb{S}_{\Gamma,c} + \mathbb{D}_{\Gamma,c} & \mathbb{S}_{\gamma,c} + \mathbb{D}_{\gamma,c} & \widetilde{\mathbb{S}}_{c,c} & 0 \\ \mathcal{L}_\gamma\, \mathbb{T}_{\Gamma,\gamma} & \mathcal{L}_\gamma \left(-\frac{1}{2}I + \mathbb{T}_{\gamma,\gamma}\right) & \mathcal{L}_\gamma\, \widetilde{\mathbb{T}}_{c,\gamma} & 0 \end{pmatrix} \begin{pmatrix} \boldsymbol{\sigma}(\Gamma) \\ \boldsymbol{\sigma}(\gamma) \\ \boldsymbol{f} \\ \boldsymbol{U}^* \end{pmatrix} = \begin{pmatrix} \mathbf{0} \\ \mathbf{0} \\ \dfrac{d\boldsymbol{r}}{dt} \\ \mathbf{0} \end{pmatrix}. \tag{5.23}$$

While the above matrix form is helpful in understanding the overall formulation, in practice, we may invert smaller systems depending on the nature of the problem. For example, as discussed earlier, if the number of unknowns on Γ is large compared to others, it would be beneficial to form its inverse as a precomputation step. All the layer potentials in this system matrix lead to $N-$body sums when discretized, thereby, require fast algorithms to accelerate their computation for large problem sizes. Many such algorithms are now well-established; we use the open-source fast multipole method (FMM) implementation of [53]. Note that solving this system gives all the unknowns at a particular time snapshot only; we then have to update the position of cilia using (5.1) and the position of the rigid particle using $\boldsymbol{U}^*$.

Finally, the formulation generalizes to multiple rigid particles in a trivial manner. We let γ denote the union of all the particle boundaries i.e., $\gamma = \bigcup_{m=1}^{N_p} \gamma_m$, where, as before, γ_m is the boundary of the m-th particle. Then, the definition of the boundary integral operators introduced so far hold as is; for example,

$$\mathbf{S}_\gamma[\boldsymbol{\sigma}](\boldsymbol{r}') = \int_\gamma \boldsymbol{G}(\boldsymbol{r}' - \boldsymbol{r})\boldsymbol{\sigma}(\boldsymbol{r})\mathrm{d}s_{\boldsymbol{r}} := \sum_{m=1}^{N_p} \int_{\gamma_m} \boldsymbol{G}(\boldsymbol{r}' - \boldsymbol{r})\boldsymbol{\sigma}(\boldsymbol{r})\mathrm{d}s_{\boldsymbol{r}}. \tag{5.24}$$

The operators and variables in (5.19), on the other hand, must be defined separately for each particle and the system (5.23) needs to be modified accordingly by concatenating the unknowns on all the particle boundaries.

5.2.4 Nyström discretization and close-evaluation of layer potentials

Given a single closed curve Γ parameterized by $Z(\alpha) : [0, 2\pi) \to \mathbb{R}^2$, such that $\Gamma = Z([0, 2\pi))$, we split the curve uniformly into n_Λ disjoint panels Λ_i, $i = 1, \ldots, n_\Lambda$. In each panel, we use p quadrature nodes so that there are $N_\Gamma = pn_\Lambda$ discrete points on the curve. The standard Gauss-Legendre quadrature, with nodes $\{t_i\}_{i=1}^{N_\Gamma}$ and associated weights $\{w_i\}_{i=1}^{N_\Gamma}$, offers high-order accuracy for integrating any smooth function g on Γ,

$$\int_\Gamma g(\boldsymbol{r})\mathrm{d}S_{\boldsymbol{r}} = \int_0^{2\pi} g(Z(\alpha)) \left|\frac{d}{d\alpha} Z(\alpha)\right| d\alpha \;\approx\; \sum_{i=1}^{N_\Gamma} g(Z(\alpha_i)) \left|\frac{d}{d\alpha} Z'(\alpha_i)\right| w_i. \tag{5.25}$$

Now consider the task of computing the velocity $\boldsymbol{u}(\boldsymbol{r'})$ at a target $\boldsymbol{r'} \in \Omega$ by evaluating (5.8). If $\boldsymbol{r'}$ is far away from a source panel Λ, the contribution of Λ to $\boldsymbol{u}$ is obtained by simply using the quadrature rule (5.25) since the integrand is smooth in this case. However if $\boldsymbol{r'}$ is close to Λ, one could expect the integral kernels in (5.9) to be much more rapidly changing functions of $\boldsymbol{r} \in \Gamma$ than $\boldsymbol{\sigma}$. In fact, the error in a fixed smooth quadrature rule grows exponentially to $\mathcal{O}(1)$ as target $\boldsymbol{r'}$ approaches Γ. These inaccuracies may lead to numerical instabilities. Therefore we adapt a local panelwise close evaluation scheme proposed in Chapter II to accurately handle nearly singular hydrodynamic interactions. This is done by first rewriting velocity field $\boldsymbol{u}$ represented by Stokes single or double layer potential in terms of complex contour integrals with different types of singularity and then using a high-order polynomial interpolation in complex plane to approximate the density function $\boldsymbol{\sigma}$. We may integrate analytically the resulting contour integral of each monomial using a two-term recurrence. This specialized panel quadrature scheme provides uniform accuracy for targets arbitrarily close to, or on, the curve.

5.3 Results and discussions

The numerical parameters used in this section are listed in Table 5.1. With these choices of the parameters, we are able to achieve a close to machine-precision accuracy with the

Parameter	Symbol	non-dimensional value
Number of cilia	N	32
Number of regularized stokeslets per cilium	N_s	20
Regularization parameter	ϵ	1/80
Radius of the outer wall	R_1	5
Radius of the inner wall	R_2	3
Quadrature points on the channel walls		≈ 3000
Quadrature points on each particle		128
Panel order		16
Number of waves	N_w	$-10, -9, -8, \cdots, 10$
Phase difference	$\Delta\phi$	$\frac{\pi}{16}N_w$
Time step	Δt	1/200

Table 5.1: List of numerical parameters.

spatial scheme and a fourth order convergence with the temporal scheme. The numerical validations are shown in the Appendix.

5.3.1 Mixing of tracers

We first apply the numerical method to study the mixing of passive tracers. Specifically, we uniformly seed 5000 tracers inside the channel and color them blue or green as shown in figure 5.3(a). We track the tracers for 10 beating cycles and visualize the positions of the passive tracers in figure 5.3(b-d) for three different phase differences between neighboring cilia. We use the number of waves formed by the cilia N_w as a proxy of the phase difference $\Delta\phi$ to make sure that there are always complete waves in the channel. One can convert between the two variables using the relation $\Delta\phi = \frac{\pi}{16}N_w$. The waves travel in the counter-clockwise direction if $N_w > 0$, which corresponds to the antiplectic metachronal waves, and vice versa for $N_w < 0$, which corresponds to the symplectic metachronal waves. Clearly, in the case of cilia beating in synchrony ($N_w = 0$), the tracers are barely mixed - a shear region could be identified between the tips of the cilia and the outer channel wall, consistent with previous numerical results [121, 39]. Note that although all the cilia are beating in synchrony, the asymmetry between the effective and recovery strokes drives the flow over one cycle. The mixing performance becomes much stronger as the phase difference becomes non-zero. Two representative cases are shown in figure 5.3(c)&(d) with $N_w = 1$ and $N_w = -9$. In the case of $N_w = 1$, the mixing region becomes much larger compare to the synchronized case while a small shear region is still observable close to the outer channel wall. On the other hand, the mixing region completely dominates the shear region when $N_w = -9$. Note that the diminished shear region and the mixing region above the ciliary tips have not been reported before in other geometries. The authors speculate that this is due to the narrowness of the channel and the no-slip boundary condition on the channel walls. Additionally, the fact that the fluid domain is closed and there is a lack of "fresh" supply of fluid could also be a reason for the diminished shear region.

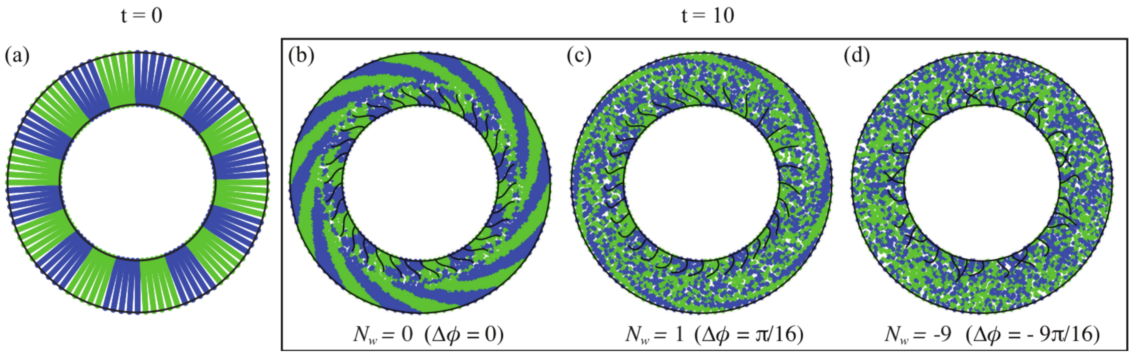

Figure 5.3: Uniformly seeded tracers mixed by cilia with different phase differences after 10 beating cycles. (a) Initial seeding; (b) $N_w = 0$; (c) $N_w = 1$; (d) $N_w = -9$.

To quantify the mixing performance, following [182], we use the shortest distance between particles of different colors as a measure. Let N_t be the total number of tracers of each color, i, j be the indices of the blue and green tracers respectively. The mixing number could be defined as

$$\mathrm{m} = \left(\prod_{i=1}^{N_t} \min_j (|\boldsymbol{r}_i - \boldsymbol{r}_j|)^2 \right)^{1/N_t} . \tag{5.26}$$

Note that the mixing number m is positive by definition; a well mixed state has a mixing number close to 0.

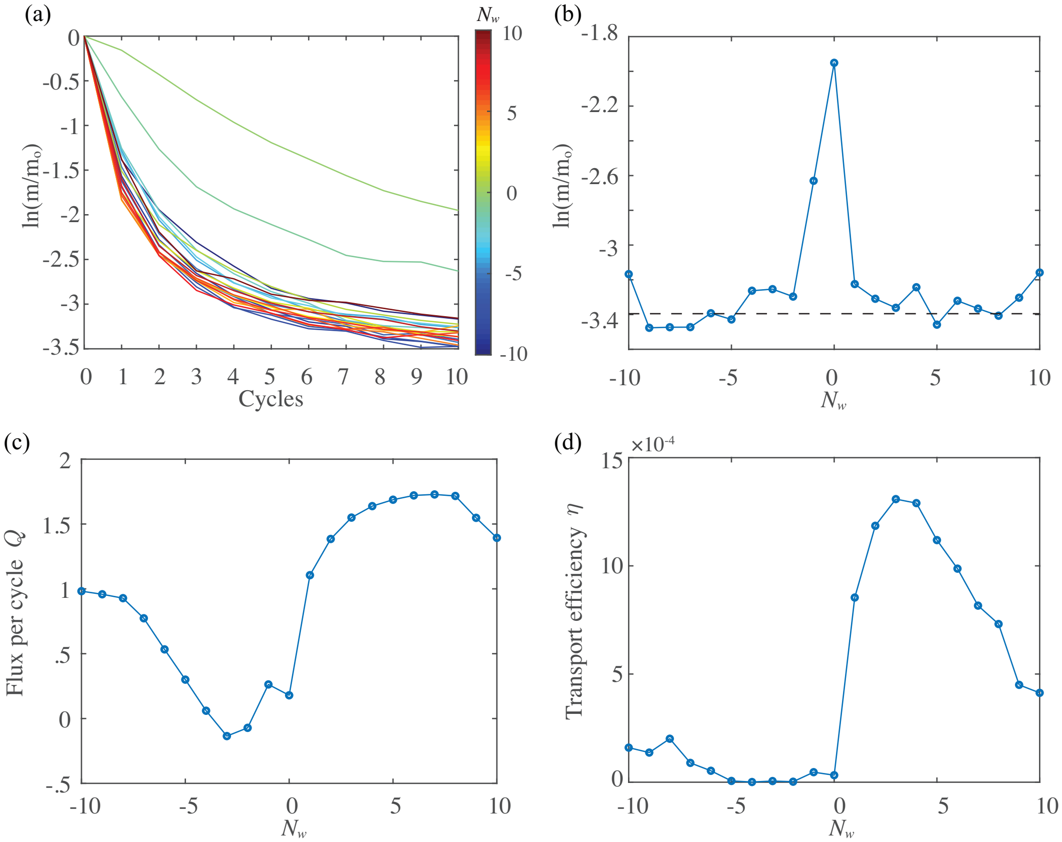

Figure 5.4: Mixing and transport performance. (a) Mixing efficiency as a function of cycles for different number of waves (phase difference). (b) Mixing efficiency after 10 cycles as a function of the number of waves N_w. (c) Total flux per cycle as a function of N_w. (d) Transport efficiency as a function of N_w.

In figure 5.4(a) we show the mixing number normalized by the initial mixing number $\mathrm{m}_o = \mathrm{m}|_{t=0}$ as a function of beating cycles in semi-logarithm axes. A wide range of phase differences are depicted by the different line colors. The mixing numbers decrease fast in the first couple of cycles and start to plateau afterwards. For all phase differences, the mixing number decreases as a function of cycles, indicating that the ciliary beating keeps mixing the fluid. The case that all cilia beat in synchrony ($N_w = 0$) has the worst mixing performance, consistent with the “scallop theorem” [153]. The mixing numbers after 10 beating cycles are shown in figure 5.4(b) as a function of number of waves N_w. The case that yields the best mixing performance is $N_w = -9$, corresponds to a phase difference $\Delta\phi = -9\pi/16$. It is also evident that the mixing performance is relatively robust to the phase difference. In fact, almost all mixing numbers reach as low as $e^{-3.4} \approx 0.03$ after normalization except for the two cases where $N_w = 0$ and -1 ($\Delta\phi = 0$ and $-\pi/16$). Note that this is in contrast to what has been observed previously in [39] for an idealized geometry where they found the mixing performances are sensitive to the phase differences and two clear local extrema were identified.

We continue by examining the transport performance of the ciliary flow in the channel. We quantify the transport by evaluating the total flux Q going through a vertical cross-section $\{(x, y)|x = 0, R_2 < y < R_1\}$ over one beating cycle. To be consistent with the ciliary effective stroke direction, we take the positive x-direction as the positive direction for the flux at this cross-section. By virtue of incompressibility, the flux going through different cross-sections are equal to each other, which we verified in our simulations (results not shown here). The total flux per cycle is shown in figure 5.4(c) as a function of phase differences. Similar to the mixing performance, having all cilia beating in synchrony generates almost no transport due to the scallop theorem. Additionally, antiplectic waves ($N_w > 0$) in general perform better than symplectic waves ($N_w < 0$). The case that generates the largest flux is $N_w = 7$ ($\Delta\phi = 7\pi/16 \approx 0.44\pi$), which is similar to what authors in [39] report, albeit in different geometries. To determine the transport efficiency, we follow the previous works of [142, 42, 40] and [71], and define the dimensionless transport efficiency η as

$$\eta = \mu \ell^{-1} \frac{Q^2}{W}, \tag{5.27}$$

where $W = \frac{1}{N_c} \sum_i^{Nc} \int_0^T \int_0^\ell \max(0, \boldsymbol{q} \cdot \boldsymbol{\alpha}) \mathrm{d}s \mathrm{d}t$ is the power loss over one beating cycle averaged over per cilium, $\boldsymbol{\alpha} = \|\dot{\boldsymbol{t}}(s)\| \frac{\boldsymbol{t} \times \dot{\boldsymbol{t}}}{\|\boldsymbol{t} \times \dot{\boldsymbol{t}}\|}$ is the angular velocity vector, $\boldsymbol{q} = \boldsymbol{t}'' \times \boldsymbol{t} + \boldsymbol{t} \times \int_s^\ell \boldsymbol{f}(\tilde{s}, t) d\tilde{s}$ is the internal moments generated along each cilium, and only positive works are accounted for. The results show that the phase difference that optimizes transport efficiency, $\Delta\phi = 3\pi/16 \approx 0.19\pi$, is smaller compare to that optimizes the total flux, due to the high power loss at larger phase differences (power loss results not shown here). Overall, the transport efficiency in viscous fluid remains small, consistent with previous works.

To further illustrate the effects of geometries on the mixing performance, we study the mixing results of ciliary beating inside a "wavy channel". Specifically, we perturb the outer channel wall such that the boundary can be written as $z = x+\mathrm{i}y = \frac{R_1(1+0.1\cos(5\theta))}{\sqrt{1+0.1^2/2}}\exp(\mathrm{i}\theta)$, $\theta \in [0, 2\pi)$ in complex form. The coefficient in the denominator is to scale the channel such that the fluid domain has the same area compared to the regular circular channels. The coefficient of the cosine term perturbs the radius of the outer boundary by about $\pm 10\%$. In other words, the narrowest and the widest channel widths are about 1.5 and 2.5 unit length. The initial seeding and the tracer positions after 10 beating cycles are shown in figure 5.5(a)&(b) with $N_w = 10$, which yields the best mixing results as shown in figure 5.5(c). When compared to figure 5.4, it is clear that not only the number of waves that yields the best mixing performance changes from -9 to 10, but also the overall mixing performance is negatively affected by the presence of the wavy channel – indicated by the mixing number $\ln(\mathrm{m}/\mathrm{m_o})$ increased from -3.4 to about -2.8 (in other words, $\mathrm{m}/\mathrm{m_o}$ increased from $e^{-3.4} \approx 0.03$ to $e^{-2.8} \approx 0.06$). The effect of the wall perturbation on mixing is even apparent to the eye: in figure 5.5(b), at each of the humps on the outer wall, a shear region could be observed which does not exist in the case of the regular Taylor-Couette geometry.

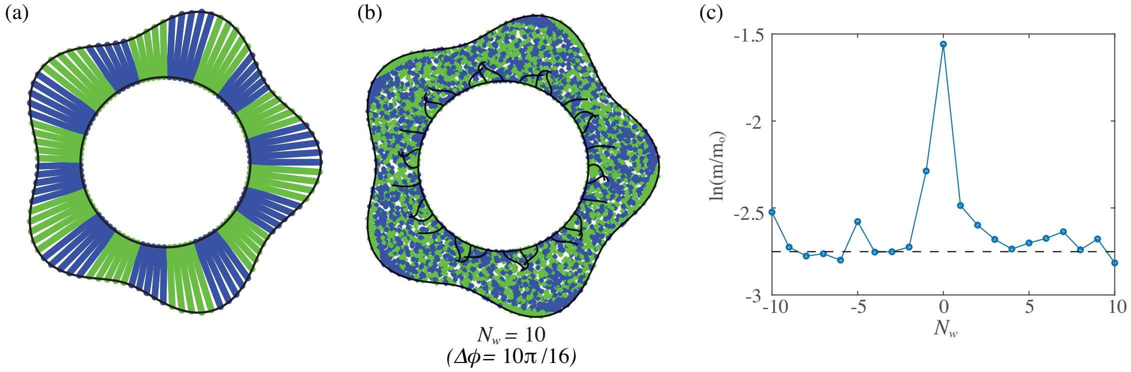

Figure 5.5: Uniformly seeded tracers mixed by cilia in a wavy channel. (a) Initial seeding. (b) Tracers after 10 cycles for $N_w = 10$ ($\Delta\phi = 10\pi/16$). (c) Mixing efficiency after 10 cycles as a function of N_w.

5.3.2 Finite size particles

In this subsection we study the full cilia-channel-particle problem and compare the results with passive tracers, in an effort to showcase the effects of the particle size in such problems. (See Section 5.2.3 on rigid particle discretization with no-slip, zero net force and net torque condition.)

We start by uniformly seeding 20 circular particles of radius r_p inside the channel and trace their centroids within one ciliary beating cycle. With small particle size, as shown in the top row of figure 5.6, the differences between the passive tracers and the finite size particles are hardly visible, as expected. With large particle size, however, the difference becomes much more evident. Specifically, in the first two phase differences ($\Delta\phi = 0, \pi/16$), the motions of the large particles are close to those of the passive tracers, albeit having noticeable shorter distance traveled (dashed curves have shorter lengths compare to solid curves). In the case of large phase difference ($\Delta\phi = -9\pi/16$), the difference between the trajectories of the large particles and the tracers are even more evident.

The radial and azimuthal positions of the tracer and particle initially centered at $(0, 4.5)$ are shown in figure 5.7 (a)&(b). It is clear that while the differences between the tracer trajectories and small particle trajectories are minimal, large particles deviate from the tracer trajectory since the beginning. Particularly, the movements of the tracers and the small particles consist of significant deviations in the radial direction, the large particle experiences limited radial deviation throughout the cycle and move in the azimuthal direction only. The net displacement of the particle over one beating cycle is shown in figure 5.7(c) as a function of N_w. The net displacement of the large particle is almost always smaller than those of the tracers and the small particles except for two special cases where $N_w = -3$ and 5. Specifically, the net displacement of the particle decreases as much as 40% in the case where $N_w = 1$.

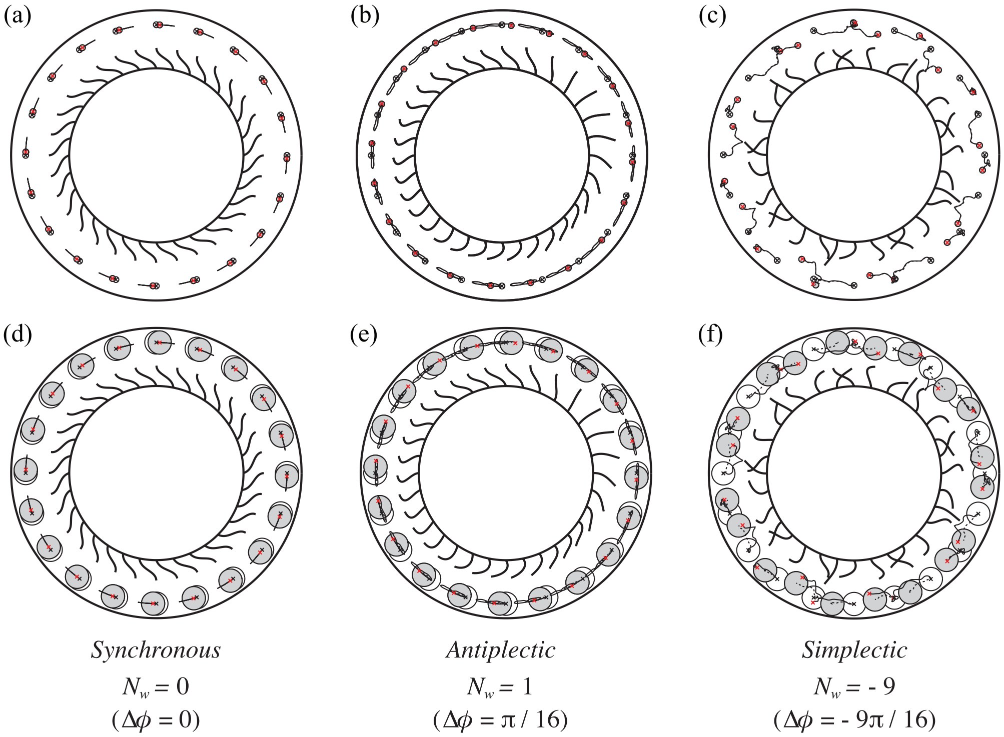

Figure 5.6: Tracers' trajectories compared to rigid particles' trajectories after one beating cycle. Tracers' initial and ending positions are shown in black and red crosses respectively; particles' initial and ending positions are shown in open and solid circles. The trajectories of the tracers and the particle centers are shown in solid and dashed lines respectively. (a)-(c): particle radius is $r_p = 0.1$; (d)-(f): particle radius is $r_p = 0.4$. Left to right: $N_w = 0, 1, -9$.

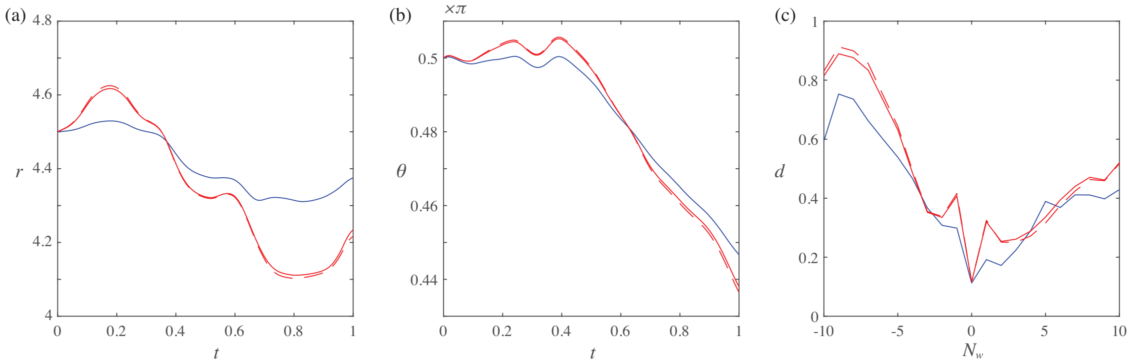

Figure 5.7: Particle displacement over one beating cycle. (a-b) The radial (r) and the angular (θ) positions of the particle (tracer) during one beating cycle with $N_w = -9$ ($\Delta\phi = -9\pi/16$). (c) The total displacement of the particle (tracer) after one beating cycle as a function of N_w. Large ($r_p = 0.4$) and small ($r_p = 0.1$) particle results are shown in blue and red lines respectively; tracer result is shown in red dash lines.

We conclude this section by studying the effects of the particle sizes in a shear flow. In particular, we seed a cluster of 4 particles of radius 0.1 in a square lattice fashion inside the channel. The zoomed-in view is shown in figure 5.8(a). We track the motion of the 4 particles over one beating cycle and measure the angle formed by the bottom 3 particles θ_p as a metric of shear deformation. Passive tracers with the same initial positions are also simulated, with the bottom angle denoted by θ_t. θ_p and θ_t are shown in figure 5.8(b) as functions of N_w. In general, the two angles follow the same trend as we sweep through N_w. A noticeable fact is that θ_p is almost always closer to 90° compare to θ_t, meaning that the finite-size of the particle is resisting shear deformation when they are in close proximity. Lastly, we track the displacement of each particle over one cycle and average across all particles as a measure of net transport. The average displacement as a function of N_w is shown in figure 5.8(c). Interestingly, having a finite-size does not always result in a smaller or a larger displacement. In fact, for most cases where $N_w > 0$ or $N_w \leq -7$, the cluster of finite-size particles moves farther than the passive tracers; whereas the cluster of passive tracers move farther than particles when $-6 \leq N_w < 0$. The differences between the displacements of the particles and the tracers reach 10% in most cases.

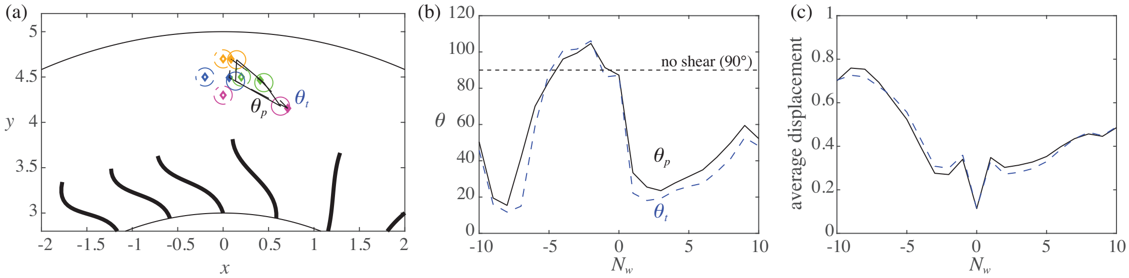

Figure 5.8: Shear deformation for tracers and rigid particles. Four tracers/particles initially seeded as a square lattice translated by cilia driven flow. (a) A zoomed-in view of the Taylor-Couette device with $N_w = 2$ ($\Delta\phi = 2\pi/16$). Initial positions are shown in open diamonds/dashed circles, final positions are shown in closed diamonds/circles. θ_t and θ_p denotes the bottom angle of the deformed lattice after one beating cycle for tracers and particles respectively. (b) θ_t (dashed line) and θ_p (solid line) as functions of N_w. (c) Displacements of the tracers and particles over one cycles, averaged for all tracers (dashed line) and particles (solid line).

5.4 Conclusions and future work

We presented a hybrid numerical method for simulating cilia-driven particulate flows in complex domains. It features a well-conditioned BIE scheme for handling the moving rigid particles and stationary walls and the method of regularized Stokeslets for handling the cilia. We emphasize that, due to the linearity of Stokes flow, each of the computational modules can be replaced with alternative formulations (e.g., a slender-body theory for handling cilia) or software implementations.

We applied this method to showcase several examples with varying degree of complexity. In particular, we systematically studied the mixing of fluid tracers inside a ciliary Taylor-Couette device. The mixing results are qualitatively different from earlier results obtained using ideal geometries. Specifically, we demonstrated a case where the mixing region completely dominated the transport region which hasn't been shown before. We also showed that a slight perturbation in the geometry could lead to a global change in the mixing performance: the mixing number increased by about a factor of 2 with the perturbation of the geometry (from $e^{-3.4} \approx 0.03$ to $e^{-2.8} \approx 0.06$). We believe the strong influence of the geometry on the mixing performance is a clear indication that efficient numerical methods that can resolve complex geometry flows are essential for bringing critical insights into ciliary flows observed in natural and engineering applications. Furthermore, we studied the transport of finite-size particles inside the confining geometry. In particular, we showed that small particles in general behave consistently with fluid tracers (this is a limiting case where the particle size is zero), while large particles impede ciliary-generated movements, as can be expected. Additionally, we demonstrated that finite-size particles in close proximity resist shear deformation.

We are currently working on extending our work on several fronts. First, we will extend our method to two-way coupled systems, specifically to deformable particles interacting with elastic cilia, capitalizing on prior works such as [70] and [198, 160]. Conceptually, our computational scheme can be extended in a straightforward manner to accommodate other two-way coupled models such as [37, 17, 31]. Specifically, in most of the two-way coupled models, the force density along the ciliary centerline, $\boldsymbol{f}$, is a function of configuration or time. Consequently, one can treat $\boldsymbol{f}$ as known and move the related terms in (5.14) to the right-hand-side and solve for $\boldsymbol{\sigma}$ only. The ciliary dynamics could then be computed via the no-slip boundary condition (5.3). We plan to explore these in the near future. Second, we will consider shape optimization problems, such as optimizing the confining geometry for a given ciliary function (e.g., fluid transport, mixing, etc.) using ideas proposed in [24]. Extension of this to work to three-dimensional problems is another natural direction.

CHAPTER VI

Optimal Slip Velocities of Micro-swimmers with Arbitrary Axisymmetric Shapes

Preamble. This chapter presents a computational approach for determining the optimal slip velocities on any given shape of an axisymmetric micro-swimmer suspended in a viscous fluid. The objective is to minimize the power loss to maintain a target swimming speed, or equivalently to maximize the efficiency of the micro-swimmer. Owing to the linearity of the Stokes equations governing the fluid motion, we show that this PDE-constrained optimization problem reduces to a simpler quadratic optimization problem, whose solution is found using a high-order accurate boundary integral method. We consider various families of shapes parameterized by the reduced volume and compute their swimming efficiency. Among those, prolate spheroids were found to be the most efficient micro-swimmer shapes for a given reduced volume. We propose a simple shape-based scalar metric that can determine whether the optimal slip on a given shape makes it a pusher, a puller or a neutral swimmer. This is joint work with Hanliang Guo, Ruowen Liu, Marc Bonnet and Shravan Veerapaneni, and is published in [73]. My contribution in this study is developing the boundary integral solver.

6.1 Introduction

The *squirmer model* [118, 21] is widely adopted by mathematicians and physicists over the past decades to model ciliated micro-swimmers such as *Opalina*, *Volvox* and *Parame-*

cium [113]. On a high level, this continuum model, sometimes referred to as the *envelope model*, effectively tracks the motion of the envelope formed by the tips of the densely-packed cilia, located on the swimmer body, while neglecting the motion below the tips. Individual and collective ciliary motions could be mapped to traveling waves of the envelope on the surface. Assuming no radial displacements of the surface and time-independent tangential velocity led to the simpler *steady* squirmer model [147], wherein, a prescribed slip velocity on the boundary propels the squirmer. While the model was originally designed for spherical shapes, it has since been adapted to more general shapes and has recently been shown to capture realistic collective behavior of suspensions [110].

Shape is also a key parameter in the design of artificial micro-swimmers for promising applications such as targeted drug delivery. In particular, the squirmer model is often employed to study the propulsion of *phoretic particles*, which are micro- to nano-meter sized particles that propel themselves by exploiting the asymmetry of chemical reactions on their surfaces [7, 56]. A classical example is the Janus sphere [87], which consists of inert and catalytic hemispheres. When submerged in a suitable chemical solution, the asymmetry between the chemical reactions on the two hemispheres creates a concentration gradient. The gradient creates an effective steady slip velocity on the surface via osmosis that naturally suits the squirmer model. Besides the classical Janus spheres and bi-metallic nanorods [146], more sophisticated shapes have also been proposed recently, such as two-spheres [194, 144], spherocylinder [193], matchsticks [132] and microstars [173]. Interestingly, [193] showed that special shapes of phoretic particles exhibit novel properties such as ‘edge-following’ when put close to chemically patterned surfaces.

Studying the efficiency of biological micro-swimmers is pivotal to understanding natural systems and designing artificial ones for accomplishing various physical tasks. The mechanical efficiency [118] of the spherical squirmer can be directly computed, as its rate of viscous energy dissipation, or power loss, can be written in terms of the modes of the squirming motion. [128] found the optimal swimming strokes of unsteady spherical squirmers by employing a pseudo-spectral method for solving the Stokes equations that govern the ambient fluid and a numerical optimization procedure. Their approach, however, does not readily generalize to arbitrary shapes. On the other hand, [116] analytically investigated the efficiency of micro-swimmers of prolate spheroids shapes with a time-independent 'treadmilling' slip velocity and found that the optimal efficiency increases unboundedly with the aspect ratio. [202] optimized the steady slip velocity and the shape at the same time, with constraints on its volume and maximum curvature. The work considered power loss inside the squirmer surface, which could be an order of magnitude higher than the outside power loss [99, 94]. However, it assumed that the tangential force on the squirmer surface is linear to its *local* slip velocity, which is not always the case for microswimmers.

In this chapter, we address the following broader questions: *Given an axisymmetric shape of a steady squirmer, what is the slip velocity that maximizes its swimming efficiency?* The optimization problem, being quadratic, is reduced to a linear system of equations solved by a direct method, while forward exterior flow problems are solved using a boundary integral method. Those combined features produce a simple and efficient solution procedure. We introduce the optimization problem and our numerical solver in Section 6.2, present the optimal solution for various shape families, summarize the correlations between the shapes and the optimal slip velocities, and propose a shape-based scalar metric to predict whether the optimized swimmer would be a pusher or a puller in Section 6.3, followed by conclusions and a discussion on future research directions in Section 6.4.

6.2 Problem Formulation and Numerical Solution

6.2.1 Model

Consider an axisymmetric micro-swimmer whose boundary Γ can be obtained by rotating a curve γ about $\boldsymbol{e}_3$ axis as shown in Fig. 6.1(a). Using the arc-length $s \in [0, \ell]$ to parameterize the generating curve, its coordinate functions can be written as $\gamma(s) = (x(s), 0, z(s))$. Here, we restrict our attention to shapes of spherical topology, therefore, all shapes considered satisfy the conditions $x(0) = x(\ell) = 0$ and $x(s) > 0,\ \forall\, s \in (0, \ell)$. We assume that the micro-swimmer is suspended in an unbounded viscous fluid domain. The governing equations for the ambient fluid in the vanishing Reynolds number limit are given by the Stokes equations:

$$-\mu \nabla^2 \boldsymbol{u} + \nabla p = \boldsymbol{0}, \quad \nabla \cdot \boldsymbol{u} = 0, \tag{6.1}$$

where μ is the fluid viscosity, p and $\boldsymbol{u}$ are the pressure and flow field respectively. In the absence of external forces and imposed flow fields, the far-field boundary condition simply is

$$\lim_{\boldsymbol{r} \to \infty} \boldsymbol{u}(\boldsymbol{r}) = \boldsymbol{0}. \tag{6.2}$$

A tangential slip u^{S} defined on γ propels the micro-swimmer forward with a translational velocity U in the $\boldsymbol{e}_3$ direction. Its angular velocity as well as the translational velocities in the $\boldsymbol{e}_1$ and $\boldsymbol{e}_2$ directions are zero by symmetry. Consequently, the boundary condition on γ is given by

$$\boldsymbol{u} = u^{\mathrm{S}} \boldsymbol{\tau} + U \boldsymbol{e}_3, \tag{6.3}$$

where $\boldsymbol{\tau}$ is the unit tangent vector on γ. Note that, in order to avoid singularities, the slip must vanish at the end points:

$$u^{\mathrm{S}}(0) = u^{\mathrm{S}}(\ell) = 0. \tag{6.4}$$

Due to the axisymmetry of Γ, the required no-net-torque condition on the freely-suspended micro-swimmer is automatically satisfied while the no-net-force condition reduces to one

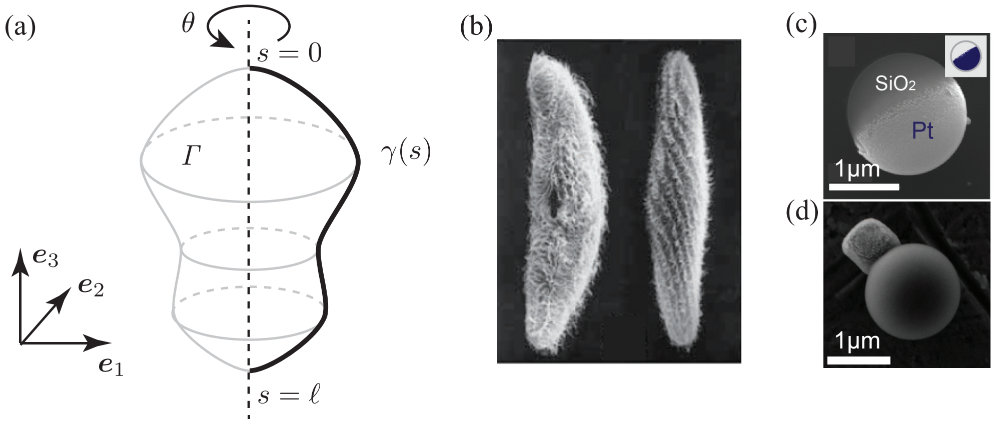

Figure 6.1: (a) Schematic of the micro-swimmer geometry. The shape is assumed to be axisymmetric, obtained by rotating the generating curve γ about the $\boldsymbol{e}_3$ axis. (b) Biological swimmers ([122], Chap 4 Fig 4.6). (c) Scanning electron microscope (SEM) image of a single half-coated Janus particle; inset: the dark-blue shows the location of the Pt cap. [35] (d) SEM image of a phototactic swimmer, which consists of a haematite particle extruded from a colloidal bead. [10]

scalar equation

$$\int_\Gamma \boldsymbol{f}(\boldsymbol{r}) \cdot \boldsymbol{e}_3 \, \mathrm{d}S_{\boldsymbol{r}} = 2\pi \int_\gamma f_3(\boldsymbol{r}) \, x \mathrm{d}s_{\boldsymbol{r}} = 0, \tag{6.5}$$

where $\boldsymbol{f}$ is the active force density on the micro-swimmer surface (negative to fluid traction) and f_3 is its $\boldsymbol{e}_3$ component.

We quantify the performance of the micro-swimmer with slip velocity u^{S} by its power loss while maintaining a target swimming speed U. The power loss is defined by

$$P = \int_\Gamma \boldsymbol{f} \cdot \boldsymbol{u} \, \mathrm{d}S_{\boldsymbol{r}} = 2\pi \int_\gamma \boldsymbol{f} \cdot (u^{\mathrm{S}} \boldsymbol{\tau} + U \boldsymbol{e}_3) x \mathrm{d}s_{\boldsymbol{r}}. \tag{6.6}$$

Note that P (different from Stokes pressure integral operator $\mathrm{P}[\boldsymbol{\sigma}]$) can be made arbitrarily small by lowering the swimming speed U. It is therefore necessary to compare the power loss of different swimmers that have the same swimming speed U. We note that a lower P with a fixed shape and swimming speed U corresponds to a higher efficiency, $\eta = C_D U^2/P$, as defined by [118], where C_D is the drag coefficient of the given swimmer.

6.2.2 Boundary integral method for the forward problem

Before stating the optimization problem, we summarize our numerical solution procedure for (6.1) – (6.3). Again, we fix the swimming speed U, referred to from here onwards as the“target swimming speed”, and assume that the tangential slip u^S is given. In general, an arbitrary pair of u^{S} and U does not satisfy the no-net-force condition (6.5). This condition will be treated as a constraint in our optimization problem. Therefore, the goal is to find the active force density $\boldsymbol{f}$ given the velocity on the boundary γ as in (6.3). We use the single-layer potential ansatz [152], which expresses the velocity as a convolution of an unknown density function $\boldsymbol{\sigma}$ with the Green's function for the Stokes equations G, from which the force density can be determined by convolution with the traction kernel T:

$$(6.7) \qquad \boldsymbol{u}(\boldsymbol{r}') = \int_\Gamma \boldsymbol{G}(\boldsymbol{r}' - \boldsymbol{r})\,\boldsymbol{\sigma}(\boldsymbol{r})\,\mathrm{d}S_{\boldsymbol{r}}, \quad \boldsymbol{f}(\boldsymbol{r}') = -\frac{1}{2}\boldsymbol{\sigma}\left(\boldsymbol{r}'\right) + \boldsymbol{n}_{\boldsymbol{r}'}\int_\Gamma \boldsymbol{T}\left(\boldsymbol{r}' - \boldsymbol{r}\right)\boldsymbol{\sigma}\left(\boldsymbol{r}\right)\mathrm{d}S_{\boldsymbol{r}},$$

where $\boldsymbol{n}$ is the unit normal vector pointing into the fluid. We can solve for $\boldsymbol{\sigma}$ by taking the limit of $\boldsymbol{r}' \to \Gamma$ in the above ansatz and substituting in (6.3). The boundary integrals in (6.7) become weakly singular on Γ, requiring specialized quadrature rules. Here, we use the approach of [197] which performs an analytic integration in the $\theta-$direction reducing the integrals to convolutions on the generating curve and applies a high-order quadrature rule designed to handle the $log-$singularity of the resulting kernels. More details on the numerical scheme are provided in Appendix H.8.

6.2.3 Optimization problem and its reformulation

The goal is to find a slip profile $u^{\mathrm{S}*}(s)$ that minimizes the power loss P while maintaining the target swimming speed U of a given axisymmetrical micro-swimmer. Let J be the objective function, here equated to P defined in (6.6), and F be the net force functional:

$$(6.8) \qquad J(u^{\mathrm{S}}) := 2\pi \int_\gamma \boldsymbol{f}(u^{\mathrm{S}}) \cdot (u^{\mathrm{S}}\boldsymbol{\tau} + U\boldsymbol{e}_3)\,x\mathrm{d}s, \quad F(u^{\mathrm{S}}) := 2\pi \int_\gamma \boldsymbol{f}(u^{\mathrm{S}}) \cdot \boldsymbol{e}_3\,x\mathrm{d}s.$$

They are slip velocity functionals as their values are completely determined by u^{S}. The optimization problem can now be stated as follows:

$$u^{\mathrm{S}*} = \operatorname*{arg\,min}_{u^{\mathrm{S}} \in \mathcal{U}} J(u^{\mathrm{S}}) \quad \text{subject to } F(u^{\mathrm{S}}) = 0, \tag{6.9}$$

with $\mathcal{U}$ being the space of the all possible slip velocities satisfying (6.4). Notice that the no-net-force condition (6.5) is added as a constraint here.

By (6.3) and linearity of the Stokes equation (6.1), the forward solution $\boldsymbol{u}$ and the net force F are affine in u^{S} ($\boldsymbol{u}$ is linear in u^{S} if $F = 0$). Consequently, $J(u^{\mathrm{S}})$ is a quadratic functional and (6.9) is inherently a quadratic optimization problem. To make it more explicit, consider a discretized version of the slip optimization problem where u^{S} is sought in the form

$$u^{\mathrm{S}}(\boldsymbol{r}) = \sum_{k=1}^{m} U \xi_k \, u_k^{\mathrm{S}}(s), \tag{6.10}$$

for some set of m basis functions u_k^{S} satisfying (6.4). We adopt a B-spline formulation for these basis functions (see Appendix G.7 for more details). Let $(\boldsymbol{u}_0, p_0, \boldsymbol{f}_0)$ and $(\boldsymbol{u}_k, p_k, \boldsymbol{f}_k)$ (with $1 \leq k \leq m$) denote the solutions of the forward problem (6.1) with $\boldsymbol{u} = \boldsymbol{e}_3$ and $\boldsymbol{u} = u_k^{\mathrm{S}} \boldsymbol{\tau}$ being their boundary conditions on γ, respectively.

The net force $F(u^{\mathrm{S}})$ is then given by $F(u^{\mathrm{S}}) = 2\pi U \mathcal{F}(\boldsymbol{\xi})$, where

$$\mathcal{F}(\boldsymbol{\xi}) := \int_{\gamma} \left(\boldsymbol{f}_0 + \sum_{k=1}^{m} \xi_k \boldsymbol{f}_k \right) \cdot \boldsymbol{e}_3 \, x \mathrm{d}s = F_0 + \boldsymbol{F}^{\mathrm{T}} \boldsymbol{\xi}. \tag{6.11}$$

Here $\boldsymbol{\xi} = (\xi_1, \dots, \xi_m)^{\mathrm{T}}$, $\boldsymbol{F} = (F_1, \dots, F_m)^{\mathrm{T}}$, and $F_k = \int_\gamma \boldsymbol{f}_k \cdot \boldsymbol{e}_3 \, x\mathrm{d}s$ for $k = 0, 1, \cdots, m$.

Similarly, we have $J(u^{\mathrm{S}}) = 2\pi U^2 \mathcal{J}(\boldsymbol{\xi})$, where

$$\mathcal{J}(\boldsymbol{\xi}) := \int_{\gamma} \left(\boldsymbol{f}_0 + \sum_{k=1}^{m} \xi_k \boldsymbol{f}_k \right) \cdot \left(\boldsymbol{e}_3 + \sum_{j=1}^{m} \xi_j u_j^{\mathrm{S}} \boldsymbol{\tau} \right) x \mathrm{d}s = \boldsymbol{\xi}^{\mathrm{T}} \boldsymbol{A} \boldsymbol{\xi} + 2 \boldsymbol{\xi}^{\mathrm{T}} \boldsymbol{F} + F_0. \tag{6.12}$$

The elements of the $m \times m$ matrix $\boldsymbol{A}$ are given by $A_{kj} = \int_\gamma \boldsymbol{f}_k \cdot u_j^{\mathrm{S}} \boldsymbol{\tau} \, x\mathrm{d}s$. We have used the fact that $\int_\gamma \boldsymbol{f}_0 \cdot u_k^{\mathrm{S}} \boldsymbol{\tau} \, x\mathrm{d}s = \int_\gamma \boldsymbol{f}_k \cdot \boldsymbol{e}_3 \, x\mathrm{d}s$ for the linear term by the reciprocal theorem [75]. We note that $\boldsymbol{A}$ is symmetric, also by the reciprocal theorem. Physically speaking, $\boldsymbol{\xi}^{\mathrm{T}} \boldsymbol{A} \boldsymbol{\xi}$ represents the scaled power loss of the swimmer being held still with its slip velocity parameterized by $\boldsymbol{\xi}$, implying that $\boldsymbol{A}$ is positive-definite; $\boldsymbol{\xi}^T \boldsymbol{F}$ is the scaled power loss of the active force along the swimming direction; F_0 is the scaled power loss of towing a rigid body with the same shape as the micro-swimmer at unit speed.

Now, the discretized optimization problem becomes

$$\min_{\boldsymbol{\xi} \in \mathbb{R}^m} \mathcal{J}(\boldsymbol{\xi}) \quad \text{subject to } \mathcal{F}(\boldsymbol{\xi}) = 0. \tag{6.13}$$

Introducing the Lagrangian $L(\boldsymbol{\xi}, \lambda) := \mathcal{J}(\boldsymbol{\xi}) - 2\lambda \mathcal{F}(\boldsymbol{\xi})$, the slip optimization problem is reduced to solving the first-order stationarity equations for L given by

$$\begin{pmatrix} \boldsymbol{A} & -\boldsymbol{F} \\ -\boldsymbol{F}^{\mathrm{T}} & 0 \end{pmatrix} \begin{pmatrix} \boldsymbol{\xi} \\ \lambda \end{pmatrix} = \begin{pmatrix} -\boldsymbol{F} \\ F_0 \end{pmatrix}. \tag{6.14}$$

Note that forming the matrix requires $(m + 1)$ solves of the forward problem (6.1) with appropriate boundary conditions. Since the micro-swimmer is assumed to be rigid, the single layer potential operator as well as the traction operator, required for forming $\boldsymbol{A}$ and $\boldsymbol{F}$, are both fixed for a given shape. Therefore, we only need to form them once.

6.3 Results

We tested the convergence of our numerical solvers rigorously; the boundary discretization for all the numerical examples presented here is chosen so that at least 6-digit solution accuracy is attained (determined *via* self-convergence tests). The optimal slip velocity for a particular prolate spheroid tested against the (truncated) analytical solution given by [116] is shown in Fig. 6.2. Our numerical solution is indistinguishable against the analytical solution at their finer truncation level $L = 10$. Additional validation results can be found in the Appendix H.8.

Here we focus on analysis of the optimal solutions for various micro-swimmer shape families. Let V be the volume enclosed by the swimmer. We normalize lengths by the radius of a

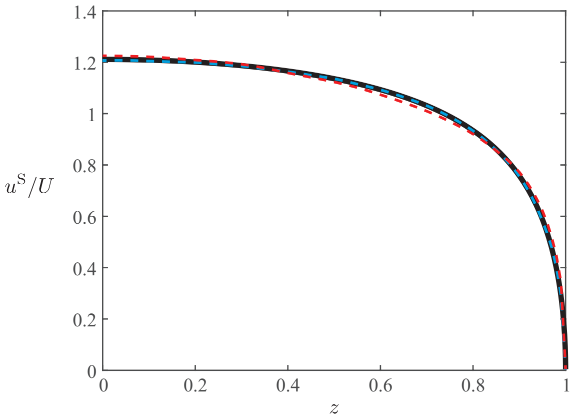

Figure 6.2: Optimal slip velocity compared to [116, Figure 4]. The aspect ratio of the prolate spheroid is $(1+2.5^2)^{1/2}$. Our numerical optimization is depicted in black solid curve, while dash curves represent analytical solutions at different truncation levels $L=4$ (red) and $L=10$ (blue).

sphere of equivalent volume i.e., by $R=(3V/4\pi)^{1/3}$, and velocities by the swimming speed U. A simple calculation shows that, for a micro-swimmer submerged in water of size $R=5\,\mu$m and the speed of one body-length per second, the Reynolds number (Re) $\approx 5\times10^{-5}$; thereby, confirming the validity of the Stokes equation (6.1). We will use the dimensionless *reduced volume*, defined by $\nu=6\sqrt{\pi}V/A^{3/2}$ where A is the surface area of the given shape, to characterize each shape family. The largest possible value of ν, attained by spheres, is $\nu=1$, while for example ν decreases monotonically for spheroids as the aspect-ratio is increased.

We first consider six different micro-swimmer shapes and plot their optimal slip profiles obtained by solving (6.14) in Fig. 6.3. In each case, we also show the flow fields in both the body and lab frames. The optimal slip velocities plotted against the arclength, measured from north pole to south pole, are shown in the insets. In the case of a sphere (Fig. 6.3(a)), we recover the standard result that the optimal profile is a sine curve [128]. The optimal slip velocity of the prolate swimmer, shown in Fig. 6.3(b), 'flattens' the sine curve in the middle while that of the oblate swimmer, shown in Fig. 6.3(c), 'pinches' the sine curve. Additionally, the peak value of the optimal slip velocity is low for the prolate swimmer, and high for the oblate swimmer, compared to the spherical swimmer.

Next, we consider three shapes corresponding to different shape families. In Fig. 6.3(d), we consider the ‘wavy’ configuration obtained by adding high-order axisymmetric modes to the spherical shape. The optimal slip velocity follows the general trend for that of (a), while lower slip velocities are observed at the troughs, qualitatively consistent to those obtained in [202]. The spherocylinder (Fig. 6.3(e)) resembles closely the prolate spheroid of Fig. 6.3(b) with the same aspect ratio, its optimal slip velocity being nearly the same (albeit with a slightly narrower plateau and higher peak slip velocity). Finally, we investigate the optimal slip velocity of the stomatocyte shape (Fig. 6.3(f)), which is the only non-convex shape among those considered here. Similar to that of the oblate swimmer, the general slip velocity is like a pinched sine wave. However, one distinguishing feature is that slip velocity is nearly zero over part of its surface, namely the cup-like region in its posterior.

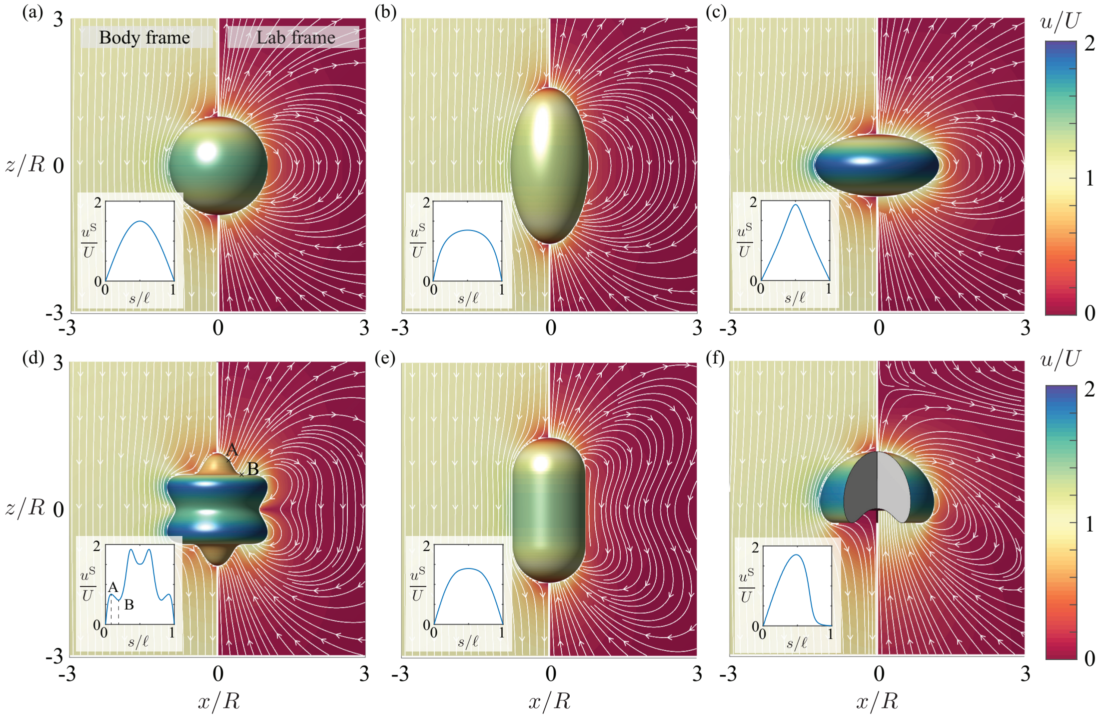

Figure 6.3: Flow fields and the optimal slip velocity for a few swimmers with typical shapes: (a) Sphere, (b) Prolate spheroid, (c) Oblate spheroid, (d) Wavy, (e) Spherocylinder, (f) Stomatocyte. Insets show the optimal slip velocities as functions of arc-length along the generating curve. The optimization is performed using 21 control points on the generating curve for representing the slip velocity. The colormap holds for both the slip velocity and the flow fields.

The optimal slip velocity strongly depends on the local geometry of the micro swimmer.

Generally speaking, the optimal slip velocity is high if the material point is far away from the axis of symmetry. This could be seen most clearly in the cases of spheroids Fig. 6.3(a)-(c). Specifically, the peak value of the optimal slip velocity is the highest for the oblate spheroid and lowest for the prolate spheroid among the three. Intuitively, an object that has a larger radius would endure a higher fluid drag compare to one with a smaller radius when moving in the same speed. Thus extra effort, in the form of slip velocity, would need to be put in to balance the drag. Additionally, the slip velocity is high when the orientation of the generating curve aligns with the swimming direction (axis of symmetry), and low otherwise. This is understandable as the slip velocity is constructed to be tangential to the generating curve, and a slip velocity perpendicular to the swimming direction generates little swimming velocity at the cost of additional power loss. This could be seen most clearly in the wavy shape Fig. 6.3(d). Specifically, comparing the two points A & B marked in the panel, although point B has a larger radius than point A, the slip velocity of point B is lower because the orientation of the generating curve is almost perpendicular to the swimming direction.

Additionally, we note that the optimal slip velocity is proportional to the target swimming speed U due to linearity of the Stokes equations. As a consequence, while the results only showcase micro-swimmers propelling themselves in the positive $\boldsymbol{e}_3$ direction, the optimal solution $u^{\mathrm{S}*}$ for swimming in the opposite direction is merely a change of sign.

Micro-swimmers can be loosely classified as *pushers* that repel fluid from the body along the axis of symmetry, *pullers* that draw fluid to the body along the axis of symmetry, or *neutral* swimmers that do not repel or draw fluid along the axis of symmetry [113]. At first sight, the flow fields for all optimal swimmers studied here seem to be neutral swimmers. A closer look into the stresslet tensor $\boldsymbol{S}_{tr}$, however, reveals a more interesting story. For axisymmetric swimmer whose swimming direction is $\boldsymbol{e}_3$, the stresslet tensor could be simplified to $\boldsymbol{S}_{tr} = S(\boldsymbol{e}_3\boldsymbol{e}_3 - \frac{1}{3}I)$, where I is the identity matrix. The sign of S characterizes whether the swimmer is a pusher ($S < 0$) or a puller ($S > 0$).

It is easy to prove by contradiction that the optimal 'front-back symmetric' swimmers can not be pushers nor pullers: flipping the direction of the slip velocity would make a pusher into a puller of the same shape with an equal (minimal) power loss, contradict to the unique solution guaranteed by the quadratic nature of the problem. However, the contradiction does not apply for 'front-back asymmetric' swimmers as flipping the swimming direction would essentially change the shape of the swimmer. In fact, the optimal 'front-back asymmetric' swimmers are not always neutral. For example, the stomatocyte shown in Fig. 6.3(f) is a puller where the stagnation point in the lab frame's flow field is in front of the micro-swimmer.

Conventionally, pusher and puller particles have been associated with 'tail-actuated' swimmers (e.g. spermatozoa) and 'head-actuated' swimmers (e.g. *Chlamydomonas reinhardtii*) respectively [166]. It is however not immediately clear whether a micro-swimmer should be a pusher (tail-actuated) or a puller (head-actuated) to optimize its efficiency when given an arbitrary shape. Here, capitalizing on our earlier observation on the dependence of local geometry and optimal slip velocity, we propose a shape-based scalar metric $\mathbb{A}$ that can be used to predict whether the optimal swimmer for a given shape is a pusher or puller without the need of optimization. Simply speaking, $\mathbb{A}$ quantifies the relative 'nominal actuation' of the 'head' part and the 'tail' part of the swimmer based solely on the swimmer shape:

$$\mathbb{A} = \log\left(\frac{\int_{\gamma_h} \boldsymbol{\tau}\cdot\boldsymbol{e}_3\, x^2 \mathrm{d}s / \int_{\gamma_h} x \mathrm{d}s}{\int_{\gamma_t} \boldsymbol{\tau}\cdot\boldsymbol{e}_3\, x^2 \mathrm{d}s / \int_{\gamma_t} x \mathrm{d}s}\right), \tag{6.15}$$

where the generating curve γ is divided into two curves $\gamma = \gamma_h \cup \gamma_t$; γ_h represents the generating curve of the head part and γ_t represents the generating curve of the tail part. The numerator and denominator inside the logarithm function are the surface averages of the nominal actuation for the head and tail part respectively. The nominal actuation is stronger if the generating curve aligns with the swimming direction better (larger $\boldsymbol{\tau}\cdot\boldsymbol{e}_3$),

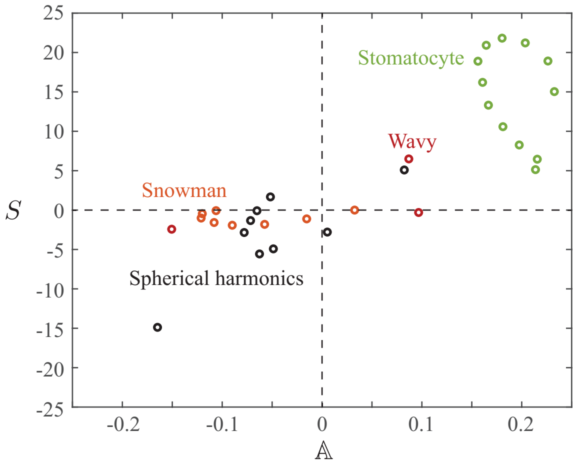

Figure 6.4: $\mathbb{A}$ provides a simple prediction of the swimmer type. Swimmers with $\mathbb{A} < 0$ are predicted to be pushers ($S < 0$), and swimmers with $\mathbb{A} > 0$ are predicted to be pullers ($S > 0$). Swimmers in the first and third quadrants are correctly predicted. Shape families are shown in Fig. 6.6 and the generating curves are given in Appendix I.9.

or if the material point is farther away from the axis of symmetry (larger x). For front-back symmetric shapes, we naturally divide γ in the middle thus $\mathbb{A} \equiv 0$; for front-back asymmetric shapes, we divide γ at the arclength where x is the largest along the generating curve $s^* = \arg\max_{s\in\gamma} x(s)$, or the average s^* if arg max returns more than one s^*. Positive $\mathbb{A}$ corresponds to shapes whose head part actuates stronger than its tail part, which indicates that the micro-swimmer is likely to be a puller; similarly negative $\mathbb{A}$ indicates that the micro-swimmer is likely to be a pusher.

The predictions based on $\mathbb{A}$ for various families of asymmetric shapes are shown in Fig. 6.4. Specifically, most of the shapes are correctly predicted as they lie in the first and the third quadrants; the ones that are misclassified, on the other hand, have close-to-zero $\mathbb{A}$ and S, which means the head and tail are similarly actuated and the optimal swimmers are close to neutral.

Next, we study the optimal active force density $\boldsymbol{f}$ corresponding to the same shapes. Its normal and tangential components are plotted in Fig. 6.5. We note that by the no-net-force condition (6.5), the power loss reduces to $P = 2\pi \int_\gamma \boldsymbol{f} \cdot (u^{\mathrm{S}} \boldsymbol{\tau}) x \mathrm{d}s$, implying that only the

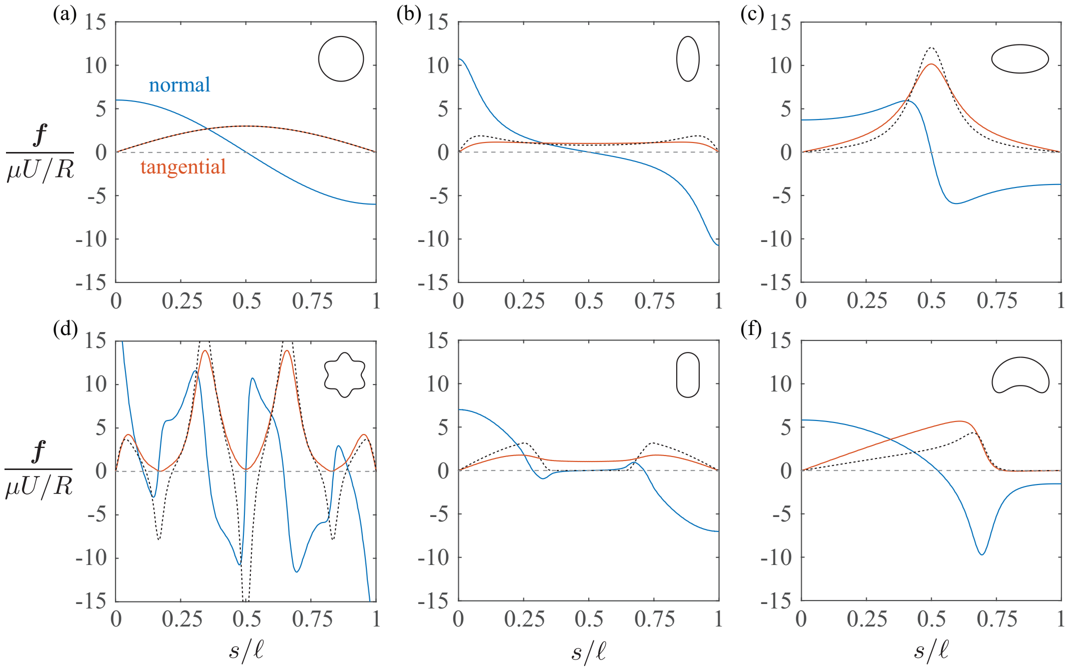

Figure 6.5: Active force density on the swimmer surface as functions of arc-length along the generating curve. Normal and tangential components of the force densities are depicted by blue and orange curves. Scaled optimal slip velocities $2u^{\mathrm{S}*}\kappa R/U$ are shown in dotted curves, where κ is the local curvature of the generating curve. Insets are the shapes of the corresponding swimmers.

tangential component contributes to the power loss. The change in tangential forces as a function of arclength loosely resembles that of the optimal slip velocity, mediated by the local curvature of the generating curve. Qualitatively, a low local curvature suppresses the traction relative to the slip velocity, and a high local curvature amplifies it. Slip velocities scaled by their local curvatures are shown in black dotted curves for a reference.

In Fig. 6.6, we plot the minimal power loss as a function of the reduced volume for various shape families. The power loss is scaled by the minimal power loss of a spherical swimmer with the same volume $J_o = 12\pi\mu R U^2$ with $R = (3V/4\pi)^{1/3}$. The minimal power loss for prolate spheroids monotonically decreases as the shape gets more slender; in contrast, it is well-known that the shape with the minimal fluid drag is one with approximately 2:1 aspect ratio [150]. By slender body theory, the power loss of a prolate spheroids scales as $\sim \mu\alpha^{2/3}U^2$, where α is the aspect ratio (see [116]). On the other hand, the minimal power loss for oblate spheroids grows rapidly as the reduced volume is increased. Shapes of the spherocylinder family behave similarly to the prolate spheroids, and converge to the spherical case when the length of the cylinder reduces to 0, as expected. It is however worth pointing out that spherocylinder costs more power loss than prolate spheroids with the same reduced volume; this relates to the fact that the peak slip velocity for spherocylinder is higher than that of the prolate spheroids (Fig. 6.3 (b)&(e)). The stomatocyte family is constructed by 'pulling' the rim of the shape, effectively making the shape 'taller' and curls deeper and deeper inside. We find that 'taller' shapes require lower power loss for this shape family, which is qualitatively consistent with the spheroid family. Finally, we note that the power loss of the snowman family (two spheres attaching with each other) is quite robust to the relative sizes of the two spheres. The power loss is only about 25% higher than that of a single sphere in the limit case where the two spheres are of the same size.

A few other examples that take more generic shapes are also shown in Fig. 6.6. The optimal slip velocities are colored on their surfaces while their power loss is shown in the form of scatter points. The generating curves of these shapes are formed by spherical harmonics. We note that the optimal performance of shapes that appear similar can be very different. For example, the difference in power loss between examples 6 and 8 is about 150% of the spherical swimmer, or 60% of example 6. This result is a strong indicator that the slip velocity of the artificial swimmer, as well as its shape, must be carefully designed to achieve good performance.

We note that the minimal power loss for all the shape families considered here are bounded from below by the curve for prolate spheroids. However, since the current work does not optimize shape, whether the prolate spheroids are universally optimal remains to be tested.

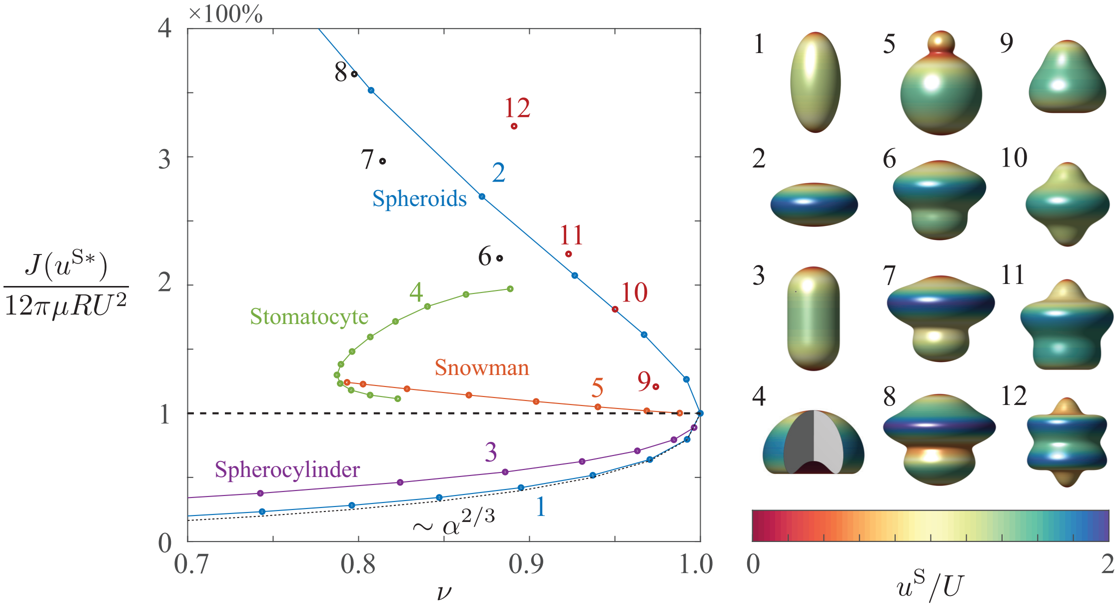

Figure 6.6: Scaled minimal power loss of different shape families, plotted against the reduced volume ν. Example shapes are color-coded by the optimal slip velocity. The dotted line shows the approximation of power loss given by the slender body theory $P \sim \mu\alpha^{2/3}U^2$ [116].

6.4 Conclusions

In this work, we provided a solution procedure for the PDE-constrained optimization problem of finding the optimal slip profile on an axisymmetric micro-swimmer that minimizes the power loss required to maintain a target swimming speed. While it can be extended to other objective functions, we exploited the quadratic nature of the power loss functional in the control parameters to simplify and streamline the solution procedure. In the general case, an adjoint formulation and iterative optimization algorithms can be employed. Regardless of the formulation, however, the use of boundary integral method to solve the Stokes equations greatly reduces the computational cost due to dimensionality reduction. Solving any of the examples presented in this work, for example, required only a few seconds on a standard laptop. Extending our procedure to fully three-dimensional (non-axisymmetric) shapes is straightforward; the key technical challenge is incorporating a high-order boundary integral solver, for which open-source codes are now available (e.g., see [55]).

Based on our numerical results, we came up with a heuristic metric that can classify the optimal swimming pattern for a given shape. It measures relative actuation of the 'head' and the 'tail' of the swimmer and predicts whether the optimal swimmer is head-actuated (puller) or tail-actuated (pusher). This metric could inform the early design of optimal slip for a given shape without the need for carrying out numerical optimization.

The optimization procedure developed in this work can directly be employed in the design pipeline of autophoretic particles. For example, in the case of diffusiophoresis, the computed optimal slip profile for a given shape can be used to formulate the chemical coating pattern of the phoretic particles. We acknowledge that the cost function for such optimization may need to be modified accordingly to reflect the chemical nature of the problem [163]. Another natural extension of this work is to relax the steady slip assumption and consider time-periodic squirming motion as done in [128]. This would be particularly useful for studying the ciliary locomotion of micro-organisms with arbitrary shapes. Furthermore, building on the recent work of [24], we are developing solvers for the shape optimization problem of finding the most efficient micro-swimmer shapes under specified area, volume or other physical constraints.

CHAPTER VII

Optimal Ciliary Locomotion of Axisymmetric Microswimmers

Preamble. Many biological microswimmers locomote by periodically beating the densely-packed cilia on their cell surface in a wave-like fashion. While the swimming mechanisms of ciliated microswimmers have been extensively studied both from the analytical and the numerical point of view, the optimization of the ciliary motion of microswimmers has received limited attention, especially for non-spherical shapes. In this chapter, using an envelope model for the microswimmer, we numerically optimize the ciliary motion of a ciliate with an arbitrary axisymmetric shape. The forward solutions are found using a fast boundary integral method, and the efficiency sensitivities are derived using an adjoint-based method. Our results show that a prolate microswimmer with a 2:1 aspect ratio shares similar optimal ciliary motion as the spherical microswimmer, yet the swimming efficiency can increase two-fold. More interestingly, the optimal ciliary motion of a concave microswimmer can be qualitatively different from that of the spherical microswimmer, and adding a constraint to the cilia length is found to improve, on average, the efficiency for such swimmers. This is joint work with Hanliang Guo, Ruowen Liu, Marc Bonnet and Shravan Veerapaneni. The preprint is ready for review in [72]. My contribution in this study is developing the boundary integral solver.

7.1 Introduction

Many swimming microorganisms propel themselves by periodically beating the active slender appendages on the cell surfaces. These slender appendages are known as cilia or flagella depending on their lengths and distribution density. Eukaryotic flagella, such as the ones in mammalian sperm cells and algae cells, are often found in small numbers, whereas ciliated swimmers such as *Paramecium* and *Opalina* present more than hundreds of cilia densely packed on the cell surfaces [25, 209]. Besides the locomotion function for microswimmers, cilia inside mammals serve various other functions such as mucociliary clearance in the airway systems and transport of egg cells in fallopian tubes (see [169], and reference therein). Cilia are also found to be critical in transporting cerebrospinal fluid in the third ventricle of the mouse brain [45] and in creating active flow environments to recruit symbiotic bacteria in a squid-vibrio system [135].

Owing to the small length scale of cilia, the typical Reynolds number is close to zero. In this regime, inertia is negligible and the dynamics are dominated by the viscous effects. As a result, many effective swimming strategies familiar to our everyday life become futile. For example, waving a rigid tail back-and-forth will not generate any net motion over one period. This is known as the time reversibility, or the 'scallop theorem', which states that a reciprocal motion cannot generate net motion [153]. Microswimmers therefore need to go through non-time-reversible shape changes to overcome and exploit drag [113].

Ciliated microswimmers break the time-reversibility on two levels. On the individual level, each cilium beats in an asymmetric pattern: during the effective stroke, the cilium pushes the fluid perpendicular to the cell surface like a straight rod, and then moves almost parallel to the cell surface in a curly shape during the recovery stroke, in preparation for the next effective stroke. On the collective level, neighboring cilia beat with a small phase difference that produces traveling waves on the cell surface, namely the metachronal wave.

Existing evidence suggests that the optimal ciliated swimmers exploit the asymmetry on the collective level more than that on the individual level [128, 71].

In this chapter, we study the (hydrodynamic) swimming efficiency of ciliated microswimmers of an arbitrary axisymmetric shape. Specifically, the swimming efficiency is understood as the ratio between the 'useful power' against the total power. The useful power could be computed as the power needed to drag a rigid body of the same shape as the swimmer with the swim speed while the total power is the rate of energy dissipation through viscous stresses in the flow to produce this motion [118]. The goal of this chapter is to find the optimal ciliary motion that maximizes the swimming efficiency for an arbitrary axisymmetric microswimmer.

Studies of ciliated microswimmers can be loosely classified into two types of models. One type is known as the sublayer models in which the dynamics of each cilium is explicitly modeled, either theoretically [25, 23] or numerically [67, 68, 69, 142, 42, 40, 71, 94, 141]. The other type is known as the *envelope model* (commonly known as the *squirmer model* if the slip profile is time-independent), which takes advantage of the densely-packing nature of cilia, and traces the continuous envelope formed by the cilia tips. The envelope model has been extensively applied to study the locomotion of both single and multiple swimmers (e.g., see [118, 21, 92, 90, 128, 27, 41, 73, 134]), as well as the nutrient uptake of microswimmers (e.g., [124, 125, 129, 130]). While originally developed for spherical swimmers, the envelope model has been generalized to spheroidal swimmers (e.g., [93, 187]).

In particular, in a seminal work, [128] studied the optimal beating stroke for a spherical swimmer using the envelope model. Specifically, the material points on the envelope were assumed to move tangentially on the surface in a time-periodic fashion, hence the swimmer retains the spherical shape. The flow field, power loss, swimming efficiency as well as their sensitivities, thereby, were computed explicitly using spherical harmonics. Their optimization found that the envelope surface deforms in a wave-like fashion, which significantly breaks the time-symmetry at the organism level similar to the metachronal waves observed in biological microswimmers.

Since most biological microswimmers do not have spherical shapes, there is a need for extending the previous work to more general geometries. Such an extension, however, is hard to carry out using semi-analytical methods. Therefore, in this chapter, we develop a computational framework for optimizing the ciliary motion of a microswimmer with arbitrary axisymmetric shape. We employ the envelope model, wherein, the envelope is restricted to move tangential to the surface so the shape of the microswimmer is unchanged during the beating period. We use a boundary integral method to solve the forward problem and derive an adjoint-based formulation for solving the optimization problem.

This chapter is organized as follows. In Section 7.2, we introduce the optimization problem, derive the sensitivity formulas and discuss our numerical solution procedure. In Section 7.3, we present the optimal unconstrained and constrained solutions for microswimmers of various shape families. Finally, in Section 7.4, we discuss our conclusions and future directions.

7.2 Problem Formulation

7.2.1 Model

Consider an axisymmetric microswimmer whose boundary Γ is obtained by rotating a generating curve γ of length ℓ about $\boldsymbol{e}_3$ axis, as shown in Figure 7.1(a). We adopt the classic envelope model [118] and assume that the ciliary tips undergo time-periodic *tangential* movements along the generating curve. Let $s = \alpha(s_0, t)$ be the ciliary tip's arclength coordinate on the generating curve γ at time t for a cilium rooted at s_0. The tangential slip velocity of

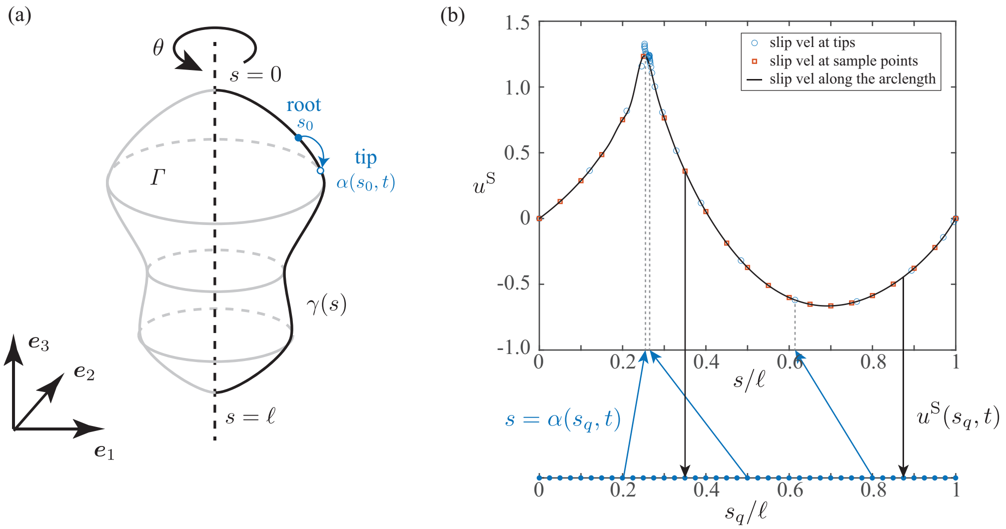

Figure 7.1: (a) Schematic of the microswimmer geometry. The shape is assumed to be axisymmetric, obtained by rotating the generating curve γ about the $\boldsymbol{e}_3$ axis. The tip of the cilium rooted at s_0 at time t is given by $s = \alpha(s_0, t)$. (b) Illustration of the algorithm for computing the slip velocity at the quadrature points $u^S(s_q, t)$. We first compute the "tip" position and the corresponding tip velocities (open blue circles) of cilia rooted at the N_q quadrature points s_q (closed blue circles). We then obtain the slip velocities at sample points uniformly distributed along the generating curve (open red squares) by a cubic interpolation. The slip velocity at any arclength (black curve) are then obtained by a high-order B-spline interpolation from the sample points. We have reduced the number of quadrature and sample points in this figure (compared to values used in the numerical experiments) to avoid visual clutter.

this material point in its body-frame is thus

$$u^S(s, t) = u^S(\alpha(s_0, t), t) = \partial_t \alpha(s_0, t). \tag{7.1}$$

In addition to the time-periodic condition, the ciliary motion α needs to satisfy two more conditions to avoid singularity [128]. First, the slip velocities should vanish at the poles

$$\alpha(0, t) = 0 \quad \text{and} \quad \alpha(\ell, t) = \ell, \quad \forall\ t \in \mathbb{R}^+, \tag{7.2}$$

and second, α should be a monotonic function, that is,

$$\partial_{s_0} \alpha(s_0, t) > 0, \quad \forall\ (s_0, t) \in [0, \ell] \times \mathbb{R}^+. \tag{7.3}$$

The last condition ensures the slip velocity is unique at any arclength s; in other words, crossing of cilia is forbidden. While in reality, cilia do cross, this condition is enforced to ensure validity of the continuum model.

In the viscous-dominated regime, the flow dynamics is described by the incompressible Stokes equations at every instance of time

$$-\mu\nabla^2\boldsymbol{u} + \nabla p = \boldsymbol{0}, \quad \nabla\cdot\boldsymbol{u} = 0, \tag{7.4}$$

where μ is the fluid viscosity, p and $\boldsymbol{u}$ are the fluid pressure and velocity fields respectively. In the absence of external forces and imposed flow field, the far-field boundary condition is simply

$$\lim_{\boldsymbol{r}\to\infty} \boldsymbol{u}(\boldsymbol{r},t) = \boldsymbol{0}. \tag{7.5}$$

The free-swimming microswimmer also needs to satisfy the no-net-force and no-net-torque conditions. Owing to the axisymmetric assumption, the no-net-torque condition is satisfied by construction, and the no-net-force condition is reduced to one scalar equation

$$\int_\Gamma \boldsymbol{f}(\boldsymbol{r},t)\cdot\boldsymbol{e}_3 \mathrm{d}S_{\boldsymbol{r}} = 2\pi\int_\gamma f_3(\boldsymbol{r},t)\, x \mathrm{d}s_{\boldsymbol{r}} = 0, \tag{7.6}$$

where x is the $\boldsymbol{e}_1$ component of $\boldsymbol{r}$, $\boldsymbol{f}$ is the active force density the swimmer applied to the fluid (negative to fluid traction) and f_3 is its $\boldsymbol{e}_3$ component.

Given any ciliary motion $\alpha(s_0,t)$ that satisfies (7.2) & (7.3), there is a unique tangential slip velocity $u^{\mathrm{S}}(s,t)$ defined by (7.1). Such a slip velocity propels the microswimmer at a translational velocity $U(t)$ in the $\boldsymbol{e}_3$ direction, determined by (7.6). Its angular velocity as well as the translational velocities in the $\boldsymbol{e}_1$ and $\boldsymbol{e}_2$ directions are zero by symmetry. Consequently, the boundary condition on γ is given by

$$\boldsymbol{u}(\boldsymbol{r}(s),t) = u^{\mathrm{S}}(s,t)\boldsymbol{\tau}(s) + U(t)\boldsymbol{e}_3, \tag{7.7}$$

where $\boldsymbol{\tau}$ is the unit tangent vector on γ. Thereby, the instantaneous power loss $P(t)$ can be written as

$$\begin{aligned} P(t) &= \int_\Gamma \boldsymbol{f}(\boldsymbol{r},t)\cdot\boldsymbol{u}(\boldsymbol{r},t)\,\mathrm{d}\Gamma \\ &= 2\pi\left[\int_\gamma \boldsymbol{f}(s,t)\cdot\boldsymbol{\tau}(s)u^{\mathrm{S}}(s,t)\,x\,\mathrm{d}s + U(t)\int_\gamma \boldsymbol{f}(s,t)\cdot\boldsymbol{e}_3\,x\,\mathrm{d}s\right]. \end{aligned} \tag{7.8}$$

The second term on the right-hand-side is zero provided that the no-net-force condition (7.6) is satisfied.

Following [118], we quantify the performance of the microswimmer by its swimming efficiency ϵ, defined as

$$\epsilon = \frac{C_D \langle U \rangle^2}{\langle P \rangle}, \tag{7.9}$$

where $P = P(t)$ and $U = U(t)$ are the instantaneous power loss and swim speed, $\langle \cdot \rangle$ denotes the time-average over one period, and C_D is the drag coefficient defined as the total drag force of towing a rigid body of the same shape at a unit speed along $\boldsymbol{e}_3$ direction. The coefficient C_D depends on the given shape γ only; for example, $C_D = 6\pi\mu a$ in the case of a spherical microswimmer with radius a.

In our simulations, we normalize the radius of the microswimmer to unity, and the period of the ciliary motion to 2π. It is worth noting that the swimming efficiency (7.9) is size and period independent, thanks to its dimensionless nature. The Reynolds number of a ciliated microswimmer of radius 100μm and frequency 30Hz submerged in water can be estimated as $\text{Re} \sim 10^{-4}$, confirming the applicability of Stokes equations.

7.2.2 Numerical algorithm for solving the forward problem

Before stating the optimization problem, we summarize our numerical solution procedure for the governing equations (7.4) – (7.7). By the quasi-static nature of the Stokes equation (7.4), the flow field $\boldsymbol{u}(\boldsymbol{r}, t)$ can be solved independently at any given time, and the time-averages can be found using standard numerical integration techniques (e.g., trapezoidal rule). Here we adopt a boundary integral method (BIM) at every time step. A similar BIM implementation was detailed in our recent work [73] which studied the optimization of *time-independent* slip profiles. The main procedures are summarized below.

We use the single-layer potential *ansatz*, which expresses the velocity as a convolution of an unknown density function $\boldsymbol{\sigma}$ with the Green's function for the Stokes equations:

$$\boldsymbol{u}(\boldsymbol{r}') = \frac{1}{8\pi}\int_\Gamma \left(\frac{1}{|\boldsymbol{r}'-\boldsymbol{r}|}I + \frac{(\boldsymbol{r}'-\boldsymbol{r})\otimes(\boldsymbol{r}'-\boldsymbol{r})}{|\boldsymbol{r}'-\boldsymbol{r}|^3}\right)\boldsymbol{\sigma}(\boldsymbol{r})\,\mathrm{d}S_{\boldsymbol{r}}. \tag{7.10}$$

The force density can then be evaluated as a convolution of $\boldsymbol{\sigma}$ with the (negative of) traction kernel:

$$\boldsymbol{f}(\boldsymbol{r}') = \frac{1}{2}\boldsymbol{\sigma}(\boldsymbol{r}') + \frac{3}{4\pi}\int_\Gamma \left(\frac{(\boldsymbol{r}'-\boldsymbol{r})\otimes(\boldsymbol{r}'-\boldsymbol{r})}{|\boldsymbol{r}'-\boldsymbol{r}|^5}\right)((\boldsymbol{r}'-\boldsymbol{r})\cdot\boldsymbol{n}_{\boldsymbol{r}'})\,\boldsymbol{\sigma}(\boldsymbol{r})\,\mathrm{d}S_{\boldsymbol{r}}. \tag{7.11}$$

We convert these weakly singular boundary integrals into convolutions on the generating curve γ by performing an analytic integration in the orthoradial direction, and apply a high-order quadrature rule designed to handle the *log*−singularity of the resulting kernels [197]. The Stokes flow problem defined at any time t by equations (7.4) – (7.7) is then recast as the BIM system for the unknowns $\boldsymbol{\sigma}$ and $U(t)$ obtained by substituting (7.10) in (7.7) and (7.11) in (7.6). The numerical solution method consists in discretizing γ into N_p non-overlapping panels, each panel supporting the nodes of a 10-point Gaussian quadrature rule. The single-layer operator is approximated in Nyström fashion, by collocation at the $N_q = 10N_p$ quadrature nodes, while the values of $\boldsymbol{\sigma}$ are sought at the same quadrature nodes. The resulting BIM system is

$$\begin{pmatrix} \mathbb{S} & -\mathcal{B} \\ \mathcal{C} & 0 \end{pmatrix}\begin{pmatrix} \boldsymbol{\sigma} \\ U(t) \end{pmatrix} = \begin{pmatrix} \boldsymbol{u}^{\mathrm{S}} \\ 0 \end{pmatrix}, \tag{7.12}$$

where the vectors $\boldsymbol{\sigma} = \boldsymbol{\sigma}(s_q, t)$ and $\boldsymbol{u}^{\mathrm{S}} = \boldsymbol{u}^{\mathrm{S}}(s_q, t)$ are the unknown density and the given slip velocity at all quadrature nodes s_q, $\mathbb{S}$ is the axisymmetric single-layer potential operator (which is fixed for a given shape γ), $\mathcal{B}$ is the column vector reproducing $\boldsymbol{e}_3$ at each quadrature node, $\mathcal{C}$ is the row vector such that $\mathcal{C}[\boldsymbol{\sigma}] = \int_\Gamma \boldsymbol{f}(\boldsymbol{r})\cdot\boldsymbol{e}_3\mathrm{d}S$ is the total traction force in the $\boldsymbol{e}_3$ direction.

The algorithm to obtain the slip velocity at the quadrature nodes at a given time $\boldsymbol{u}^{\mathrm{S}}(s_q, t)$ is summarized in Figure 7.1(b). Specifically, we start by computing the corresponding ciliary tip position $s = \alpha(s_q, t)$ and the slip velocity $u^{\mathrm{S}}(s,t)$ from (7.1). These tip positions s can be highly nonuniform, depending on the form of α, which could be difficult for the forward solver. To circumvent this difficulty and to find a smooth representation of the slip velocities on the quadrature points, we first find the slip velocities at N_s sample points uniformly distributed along the generating curve by interpolating $u^{\mathrm{S}}(s,t)$ (we use the routine `PCHIP` in MATLAB); the slip velocities at the quadrature nodes $u^{\mathrm{S}}(s_q,t)$ are then in turn interpolated from the N_s sample points using high-order B-spline bases. An alternative approach could be to follow the position and the slip velocity of each material point. In other words, one can use $\boldsymbol{u}^{\mathrm{S}}(s,t)$ directly on the right-hand-side of (7.12), which will bypass the interpolation steps mentioned above. However, it requires re-assembly of the matrix $\mathbb{S}$ at every time step, significantly increasing the computational cost.

7.2.3 Optimization problem

The goal of this work is to find the optimal ciliary motion for a given arbitrary axisymmetric shape, that is, the ciliary motion $\alpha^\star(s_0, t)$ that maximizes the swimming efficiency ϵ:

$$\alpha^\star = \underset{\alpha \in \mathcal{A}}{\arg\max}\, \epsilon(\alpha), \tag{7.13}$$

where $\mathcal{A}$ is the space of all possible time-periodic ciliary motion satisfying (7.2) & (7.3). It is, however, not easy to define and manipulate finite-dimensional parameterizations of α that remain in that space. To circumvent this difficulty, we follow the ideas in [128] and represent α in terms of a time-periodic function $\psi(x,t)$, such that

$$\alpha(s_0, \psi) = \frac{\ell \int_0^{s_0} [\psi(x,t)]^2 \mathrm{d}x}{\int_0^{\ell} [\psi(x,t)]^2 \mathrm{d}x}, \tag{7.14}$$

where ℓ is the total length of the generating curve γ. Note that α is also (implicitly) a function of time t, through $\psi = \psi(x,t)$. It is easy to verify that α given by (7.14) satisfies the boundary conditions (7.2) and the monotonicity requirement (7.3) for any choice of ψ. Conversely, for any α satisfying (7.2) and (7.3), there is at least one ψ that provides α. As a result, the optimization problem is recast as finding

$$\psi^\star = \arg\max_{\psi} \epsilon(\psi), \tag{7.15}$$

where $\psi(\cdot, t)$ is only required to be square-integrable over $[0, \ell]$ for any t.

We use a quasi-Newton BFGS method [137] to optimize the ciliary motion *via* ψ, which requires repeated evaluations of efficiency sensitivities with respect to perturbations of ψ. The sensitivities of power loss and swim speed are derived using an adjoint-based method, while the efficiency sensitivity is found using the quotient rule thereafter. The adjoint-based method exhibits a great advantage against the traditional finite difference method when finding the sensitivities, as regardless of the dimension of the parameter space, the objective derivatives with respect to all design parameters can here be evaluated on the basis of *one* solve of the forward problem for each given ciliary motion α. The derivations are detailed below.

7.2.4 Sensitivity analysis

We start by finding the sensitivities in terms of the slip profile u^{S}. The sensitivities in terms of the auxiliary unknown ψ will be found subsequently by a change of variable. As the concept of adjoint solution in general rests on duality considerations, we recast the forward flow problem in weak form for the purpose of finding the sought sensitivities of power loss and swim speed, even though the numerical forward solution method used in this work does not directly exploit that weak form. Specifically, we recast the forward problem (7.4) – (7.7) in mixed weak form (see, e.g., [26, Chap. 6]). That is, find $(\boldsymbol{u}, p, \boldsymbol{f}, U) \in \boldsymbol{\mathcal{V}} \times \mathcal{P} \times \boldsymbol{\mathcal{F}} \times \mathbb{R}$, such that

$$
\text{(7.16)}\qquad
\begin{array}{ll}
\text{(a)} & a(\boldsymbol{u},\boldsymbol{v}) - b(\boldsymbol{v},p) - b(\boldsymbol{u},q) - \langle \boldsymbol{f},\boldsymbol{v}\rangle_\Gamma = 0 \quad \forall(\boldsymbol{v},q) \in \boldsymbol{\mathcal{V}} \times \mathcal{P} \\[2ex]
\text{(b)} & \langle \boldsymbol{g},\boldsymbol{e}_3\rangle_\Gamma U + \langle \boldsymbol{g}, u^{\mathrm{S}}\boldsymbol{\tau}\rangle_\Gamma - \langle \boldsymbol{g},\boldsymbol{u}\rangle_\Gamma = 0 \quad \forall \boldsymbol{g} \in \boldsymbol{\mathcal{F}} \\[2ex]
\text{(c)} & \langle \boldsymbol{f},\boldsymbol{e}_3\rangle_\Gamma = 0
\end{array}
$$

where the bilinear forms a and b are defined by

$$
a(\boldsymbol{u},\boldsymbol{v}) := \int_\Omega 2\mu \boldsymbol{D}[\boldsymbol{u}] : \boldsymbol{D}[\boldsymbol{v}]\,\mathrm{d}V, \qquad b(\boldsymbol{v},q) := \int_\Omega q \operatorname{div} \boldsymbol{v}\,\mathrm{d}V, \tag{7.17}
$$

and $\boldsymbol{D}[\boldsymbol{u}] := (\boldsymbol{\nabla u} + \boldsymbol{\nabla}^T\boldsymbol{u})/2$ is the strain rate tensor. $\langle\cdot,\cdot\rangle_\Gamma$ is a short-hand for the inner product on Γ. For example, $\langle \boldsymbol{f},\boldsymbol{v}\rangle_\Gamma = \int_\Gamma \boldsymbol{f}\cdot\boldsymbol{v}\,\mathrm{d}\Gamma$. Similarly, with a slight abuse of notation, the power loss functional could be written as $P(u^{\mathrm{S}}) := \langle \boldsymbol{f}, u^{\mathrm{S}}\boldsymbol{\tau} + U\boldsymbol{e}_3\rangle_\Gamma$, where $U := U(u^{\mathrm{S}})$ is the swim speed functional.

The Dirichlet boundary condition (7.7) is (weakly) enforced explicitly through (7.16 b), rather than being embedded in the velocity solution space $\boldsymbol{\mathcal{V}}$, as this will facilitate the derivation of slip derivative identities; this is in fact our motivation for using the mixed weak form (7.16). Condition (7.16 c) is the no-net-force condition (7.6).

First-order sensitivities of functionals at u^{S} are defined as directional derivatives, by considering perturbations of u^{S} of the form

$$
u^{\mathrm{S}}_\eta = u^{\mathrm{S}} + \eta\nu \tag{7.18}
$$

for some ν in the slip velocity space and $\eta \in \mathbb{R}$. Then, the directional (or Gâteaux) derivative of a functional $J(u^{\mathrm{S}})$ in the direction ν, denoted by $\dot{J}(u^{\mathrm{S}};\nu)$, is defined as

$$
\dot{J}(u^{\mathrm{S}};\nu) = \lim_{\eta\to 0} \frac{1}{\eta}\left(J[u^{\mathrm{S}}_\eta] - J[u^{\mathrm{S}}]\right). \tag{7.19}
$$

For the power loss functional, we obtain (since the derivative of u^{S} in the above sense is ν)

$$
\dot{P}(u^{\mathrm{S}};\nu) = \langle \dot{\boldsymbol{f}}, u^{\mathrm{S}}\boldsymbol{\tau} + U\boldsymbol{e}_3\rangle_\Gamma + \langle \boldsymbol{f}, \nu\boldsymbol{\tau}\rangle_\Gamma + \langle \boldsymbol{f},\boldsymbol{e}_3\rangle_\Gamma \dot{U}, \tag{7.20}
$$

where $\dot{\boldsymbol{f}}$ and $\dot{U}$ are the derivatives of the active force $\boldsymbol{f}$ and swim speed U solving problem (7.16), considered as functionals on the slip velocity u^{S}:

$$\dot{\boldsymbol{f}} = \lim_{\eta\to 0} \frac{1}{\eta}\left(\boldsymbol{f}[u^{\mathrm{S}}_{\eta}] - \boldsymbol{f}[u^{\mathrm{S}}]\right), \qquad \dot{U} = \lim_{\eta\to 0} \frac{1}{\eta}\left(U[u^{\mathrm{S}}_{\eta}] - U[u^{\mathrm{S}}]\right). \tag{7.21}$$

Differentiating the weak formulation (7.16) of the forward problem with respect to u^{S} leads to the weak formulation of the governing problem for the derivatives $(\dot{\boldsymbol{u}}, \dot{\boldsymbol{f}}, \dot{p}, \dot{U})$ of the solution $(\boldsymbol{u}, \boldsymbol{f}, p, U)$

$$\begin{aligned}
&\text{(a)} \quad a(\dot{\boldsymbol{u}}, \boldsymbol{v}) - b(\dot{\boldsymbol{u}}, q) - b(\boldsymbol{v}, \dot{p}) - \langle \dot{\boldsymbol{f}}, \boldsymbol{v}\rangle_{\Gamma} = 0 && \forall(\boldsymbol{v}, q) \in \boldsymbol{\mathcal{V}} \times \mathcal{P}\\
&\text{(b)} \quad \langle \nu\boldsymbol{\tau}, \boldsymbol{g}\rangle_{\Gamma} + \dot{U}\langle \boldsymbol{e}_3, \boldsymbol{g}\rangle_{\Gamma} - \langle \dot{\boldsymbol{u}}, \boldsymbol{g}\rangle_{\Gamma} = 0 && \forall \boldsymbol{g} \in \boldsymbol{\mathcal{F}}\\
&\text{(c)} \quad \langle \dot{\boldsymbol{f}}, \boldsymbol{e}_3\rangle_{\Gamma} = 0
\end{aligned} \tag{7.22}$$

Here we have assumed without loss of generality that the test functions in (7.16) verify $\dot{\boldsymbol{v}} = \boldsymbol{0}$, $\dot{\boldsymbol{g}} = \boldsymbol{0}$, and $\dot{q} = 0$, which is made possible by the absence of boundary constraints in $\boldsymbol{\mathcal{V}}$.

At first glance, evaluating $\dot{P}(u^{\mathrm{S}}; \nu)$ in a given perturbation ν appears to rely on solving the derivative problem (7.22). However, a more effective approach allows to bypass the actual evaluation of $\dot{\boldsymbol{f}}$. Let the adjoint problem be defined by

$$\begin{aligned}
&\text{(a)} \quad a(\hat{\boldsymbol{u}}, \boldsymbol{v}) - b(\hat{\boldsymbol{u}}, q) - b(\boldsymbol{v}, \hat{p}) - \langle \hat{\boldsymbol{f}}, \boldsymbol{v}\rangle_{\Gamma} = 0 && \forall(\boldsymbol{v}, q) \in \boldsymbol{\mathcal{V}} \times \mathcal{P},\\
&\text{(b)} \quad \langle \boldsymbol{e}_3, \boldsymbol{g}\rangle_{\Gamma} - \langle \hat{\boldsymbol{u}}, \boldsymbol{g}\rangle_{\Gamma} = 0 && \forall \boldsymbol{g} \in \boldsymbol{\mathcal{F}},
\end{aligned} \tag{7.23}$$

i.e. $(\hat{\boldsymbol{u}}, \hat{p})$ are the flow variables induced by prescribing a unit velocity $\boldsymbol{e}_3$ on Γ. For later convenience, we let F_0 denote the (nonzero) net force exerted on Γ by the adjoint flow:

$$F_0 := \langle \hat{\boldsymbol{f}}, \boldsymbol{e}_3\rangle_{\Gamma}. \tag{7.24}$$

Problem (7.23) in strong form is defined by equations (7.4) – (7.7) with $U = 1$, $u^{\mathrm{S}} = 0$. In fact, F_0 takes the same value as the drag coefficient C_D in (7.9).

Then, combining the derivative problem (7.22) with the forward problem (7.16) or the adjoint problem (7.23) with appropriate choices of test functions allows to derive expressions of $\dot{P}(u^{\mathrm{S}};\nu)$ and $\dot{U}(u^{\mathrm{S}};\nu)$ which do not involve the forward solution derivatives.

Specifically, set the test functions to $(\boldsymbol{v},q,\boldsymbol{g}) = (\dot{\boldsymbol{u}},\dot{p},\dot{\boldsymbol{f}})$ in equations (7.16a,b) of the forward problem and $(\boldsymbol{v},q,\boldsymbol{g}) = (\boldsymbol{u},p,\boldsymbol{f})$ in equations (7.22a,b) of the derivative problem. Then, the combination $(7.22a) + (7.22b) - (7.16a) - (7.16b)$ is evaluated, to obtain

$$\langle \dot{\boldsymbol{f}}, u^{\mathrm{S}}\boldsymbol{\tau} + U\boldsymbol{e}_3\rangle_\Gamma = \langle \boldsymbol{f}, \nu\boldsymbol{\tau}\rangle_\Gamma + \langle \boldsymbol{f}, \boldsymbol{e}_3\rangle_\Gamma \dot{U}. \tag{7.25}$$

Substituting (7.25) into (7.20), and recalling the no-net-force condition (7.6), we have

$$\boxed{\dot{P}(u^{\mathrm{S}};\nu) = 2\langle \boldsymbol{f}, \nu\boldsymbol{\tau}\rangle_\Gamma = 4\pi \int_\gamma (\boldsymbol{f}\cdot\boldsymbol{\tau})\,\nu x \,\mathrm{d}s.} \tag{7.26}$$

Likewise, setting the test functions to $(\boldsymbol{v},q,\boldsymbol{g}) = (\dot{\boldsymbol{u}},\dot{p},\dot{\boldsymbol{f}})$ in the adjoint problem (7.23) and $(\boldsymbol{v},q,\boldsymbol{g}) = (\hat{\boldsymbol{u}},\hat{p},\hat{\boldsymbol{f}})$ in equations (7.22a,b) of the derivative problem (7.22), then evaluating $(7.22a) + (7.22b) - (7.23a) - (7.23b)$, yields

$$0 = \langle \hat{\boldsymbol{f}}, \nu\boldsymbol{\tau}\rangle_\Gamma + \langle \hat{\boldsymbol{f}}, \dot{U}\boldsymbol{e}_3\rangle_\Gamma - \langle \dot{\boldsymbol{f}}, \boldsymbol{e}_3\rangle_\Gamma = \langle \hat{\boldsymbol{f}}, \nu\boldsymbol{\tau}\rangle_\Gamma + F_0\dot{U}. \tag{7.27}$$

Note that $\langle \dot{\boldsymbol{f}}, \boldsymbol{e}_3\rangle_\Gamma = 0$ according to (7.22c). Rearranging terms in (7.27), we have

$$\boxed{\dot{U}(u^{\mathrm{S}};\nu) = -\frac{1}{F_0}\langle \hat{\boldsymbol{f}}, \nu\boldsymbol{\tau}\rangle_\Gamma = -\frac{2\pi}{F_0}\int_\gamma (\hat{\boldsymbol{f}}\cdot\boldsymbol{\tau})\,\nu x\,\mathrm{d}s.} \tag{7.28}$$

The sensitivity formulas (7.26) & (7.28), however, are not practically applicable in this form to the current optimization problem, because the constraints (7.2) & (7.3) are not easy to enforce on parameterizations of the unknown slip profiles u^{S}. For this reason, we rewrite the quantities of interest as functionals of ψ, and the connection between ψ and α is given by (7.14). Specifically, the slip profile is

$$u^{\mathrm{S}}(s,t) = \partial_t\alpha(s_0,\psi) = \partial_\psi\alpha(s_0,\psi;\dot{\psi}) = \partial_\psi\alpha\left(\beta(s,\psi),\psi;\dot{\psi}\right) = v^{\mathrm{S}}(s,\psi), \tag{7.29}$$

where $\dot{\psi} := \partial_t \psi$, and $\beta(s,\psi)$ is the inverse function of α, i.e., $s_0 = \beta(s,\psi)$. The average power loss and swim speed functionals are written as

$$\langle \mathbb{P} \rangle(\psi) := \langle P \rangle(u^{\mathrm{S}}), \quad \langle \mathbb{U} \rangle(\psi) := \langle U \rangle(u^{\mathrm{S}}) \qquad \text{with } \; u^{\mathrm{S}}(s,t) = v^{\mathrm{S}}(s,\psi). \tag{7.30}$$

On applying the change of variables $s = \alpha(s_0,\psi)$ in the integrals (7.26) & (7.28) and average over one period, we obtain

$$\boxed{\langle \dot{\mathbb{P}} \rangle(\psi;\hat{\psi}) = 2 \int_0^{2\pi} \int_\gamma \boldsymbol{f}(\alpha) \cdot \boldsymbol{\tau}(\alpha)\, x_1(\alpha)\, \dot{v}^{\mathrm{S}}(s,\psi;\hat{\psi})\, \partial_s \alpha \, \mathrm{d}s_0 \, \mathrm{d}t,} \tag{7.31}$$

$$\boxed{\langle \dot{\mathbb{U}} \rangle(\psi;\hat{\psi}) = -\frac{1}{F_0} \int_0^{2\pi} \int_\gamma \hat{\boldsymbol{f}}(\alpha) \cdot \boldsymbol{\tau}(\alpha)\, x_1(\alpha)\, \dot{v}^{\mathrm{S}}(s,\psi;\hat{\psi})\, \partial_s \alpha \, \mathrm{d}s_0 \, \mathrm{d}t,} \tag{7.32}$$

where $\dot{v}^{\mathrm{S}}(s,\psi;\hat{\psi})$ is the directional derivative of u^{S} with respect to ψ and in the direction $\hat{\psi}$. Specifically, we can show that

$$\begin{aligned} \dot{v}^{\mathrm{S}}(s,\psi;\hat{\psi})\, \partial_s \alpha(s_0,\psi) \mathrm{d}s_0 = \Big\{ & \partial_s \alpha(s_0,\psi) \, \Big[\partial^2_\psi \alpha \left(s_0, \psi; \hat{\psi}, \dot{\psi} \right) + \partial_\psi \alpha \left(s_0, \psi; \dot{\hat{\psi}} \right) \Big] \\ & - \partial_{\psi s} \alpha \left(s_0, \psi; \dot{\psi} \right) \; \partial_\psi \alpha \left(s_0, \psi; \hat{\psi} \right) \Big\} \, \mathrm{d}s_0. \end{aligned} \tag{7.33}$$

The derivation and the explicit expression of each term in (7.33) are given in the Appendix. Finally, the efficiency sensitivity in terms of ψ readily follows by the quotient rule

$$\dot{\epsilon}(\psi;\hat{\psi}) = C_D \frac{2 \langle \mathbb{U} \rangle \langle \dot{\mathbb{U}} \rangle \langle \mathbb{P} \rangle - \langle \mathbb{U} \rangle^2 \langle \dot{\mathbb{P}} \rangle}{\langle \mathbb{P} \rangle^2}. \tag{7.34}$$

7.2.5 Constraints on surface displacement

The unconstrained optimization problem (7.15) introduced above has the tendency to converge to unphysical/unrealistic strokes, where each cilium effectively ‘covers’ the entire generating curve. For a more realistic model, we should add a constraint on the length of the cilium. To this end, and again following [128], we replace the initial unconstrained optimization problem (7.15) with the penalized optimization problem

$$\psi^\star = \arg\max_\psi E(\psi), \qquad E(\psi) = \epsilon(\psi) - C(\psi) \tag{7.35}$$

where the (non-negative) penalty term $C(\psi)$, defined as

$$C(\psi) = \int_0^\ell H(A(\psi) - c)\mathrm{d}s_0, \tag{7.36}$$

serves to incorporate the kinematical constraint $A(\psi) \leq c$ in the optimization problem. The functional $A(\psi)$ in (7.36) is a measure of the amplitude of the displacement of individual material points for the stroke (through α), and c is a threshold parameter to bound $A(\psi)$ (a smaller c corresponding to a stricter constraint). H is a smooth non-negative penalty function defined by

$$H(u) = \Lambda_1 \left[1 + \tanh\left(\Lambda_2 u\right)\right] u^2, \tag{7.37}$$

which for large enough Λ_2 approximates $u \mapsto 2\Lambda_1 u^2 Y(u)$ (Y being the Heaviside unit step function). The multiplicative parameter Λ_1 then serves to tune the severity of the penalty incurred by violations of the constraint $A(\psi) \leq c$. We use $\Lambda_1 = 10^4$ and $\Lambda_2 = 10^4$ in our numerical simulations unless otherwise mentioned. The optimization results are not sensitive to the choice of Λ_1 and Λ_2. A small caveat of the penalty function (7.37) is that it has a (small) bump at $\Lambda_2 u \approx -1.109$. This bump would occasionally trap the optimizations into local extrema that have significantly lower efficiencies, depending on the initial guesses. Perturbing Λ_2 for such cases helps to alleviate the problem.

The physically most relevant definition of A would be the actual displacement amplitude of an individual point, i.e., $\Delta s = [\alpha_{\max}(s_0) - \alpha_{\min}(s_0)]/2$. The strong nonlinearity of this measure, however, is not appropriate for the computation of the gradient. Following [128], we measure the displacement by its variance in time:

$$A(\psi) = \langle (\alpha(s_0, \psi) - \langle \alpha \rangle (s_0))^2 \rangle. \tag{7.38}$$

The maximum displacement Δs will be found post-optimization for the optimal ciliary motion $\alpha^\star$ to better illustrate our results in Section 7.3.

Like the initial problem (7.15), the penalized problem (7.35) is solvable using unconstrained optimization methods, and we again adopt a quasi-Newton BFGS algorithm to optimize the ciliary motion. Applying the chain rule to the penalty functional $C(\psi)$, we obtain the derivative of the penalty term in the direction of $\hat{\psi}$ as

$$\dot{C}(\psi;\hat{\psi}) = \int_0^\ell \dot{H}(A(\psi)-c)\dot{A}(\psi;\hat{\psi})\mathrm{d}s_0. \tag{7.39}$$

The derivative of the penalized objective functional $E(\psi)$ is therefore

$$\dot{E}(\psi;\hat{\psi}) = \dot{\epsilon}(\psi;\hat{\psi}) - \dot{C}(\psi;\hat{\psi}), \tag{7.40}$$

where $\dot{\epsilon}$ and $\dot{C}$ are given by equations (7.34) and (7.39), respectively.

7.3 Results and discussion

7.3.1 Parameterization

We parametrize $\psi(s_0,t)$ such that

$$\psi(s_0,t) = \sum_{k=1}^{m} \xi_k(t)B_k(s_0), \tag{7.41}$$

where B_k are the 5th order B-spline basis functions and their coordinates $\xi_k(t)$ are expanded as trigonometric polynomials $\xi_k(t) = a_{0k}/2 + \sum_{j=1}^{n}[a_{jk}\cos jt + b_{jk}\sin jt]$ to ensure time-periodicity. Taken together, we have

$$\psi(s_0,t) = \sum_{k=1}^{m}\left[\frac{a_{0k}}{2} + \sum_{j=1}^{n}(a_{jk}\cos jt + b_{jk}\sin jt)\right]B_k(s_0) \tag{7.42}$$

so that the finite-dimensional optimization problem seeks optimal values for the $m(2n+1)$ coefficients a_{0k}, a_{jk} and b_{jk}. The initial guesses are chosen to be low frequency waves with small wave amplitudes. To obtain such initial waves, the coefficients of the zeroth Fourier mode $a_{0k}/2$ are randomly chosen from a uniform distribution within $[0,1]$, the first Fourier modes a_{1k} and b_{1k} are randomly chosen from a uniform distribution within $[0,0.01]$, and the coefficients for higher order Fourier modes $j>1$ are set to 0. To evaluate the gradient of

$E(\psi)$ with respect to the design parameters a_{0k}, a_{jk} and b_{jk}, we use (7.40) with $\hat{\psi}$ taken as the basis functions of the adopted parameterization (7.42), i.e. $\hat{\psi}(s_0, t) = B_k(s_0)/2$, $\hat{\psi}(s_0, t) = B_k(s_0) \cos jt$ and $\hat{\psi}(s_0, t) = B_k(s_0) \sin jt$, respectively. In terms of parameterization, local minima are multiple in the parameter space, since multiplying optimal parameters by a constant factor yields the same optimum for α.

7.3.2 Spheroidal swimmers

By way of validation, we start with optimizing the ciliary motion of a spherical microswimmer. The efficiency ϵ as a function of iteration number for the unconstrained optimization (7.15) is shown in Figure 7.2(a) in blue. The maximum efficiency is about 35%. The ciliary motion of the optimal spherical microswimmer is shown in Figure 7.2(b). Each curve follows the arclength coordinate of a cilium tip over one period. We observe, similar to the results of [128], clearly distinguished strokes within the beating period. In particular, cilia travel downward '*spread out*' during the effective stroke (corresponding to a stretching of the surface), but travel upward 'bundled' together during the recovery stroke in a shock-like structure (corresponding to a compression of the surface). This type of waveform is known as an *antiplectic metachronal wave* [104, 22]. We note that this optimal ciliary motion produces an efficiency higher than the 23% efficiency obtained numerically by [128, Fig. 11]. This is due to a larger maximum displacement $\Delta s \approx 0.45\ell$ in our optimizations (translated to a maximum angle of 81 degrees vs 53 degrees). Our optimization result aligns well with their results using the analytical *ansatz* [128, Fig. 14]. Additionally, we found that increasing the number of Fourier mode n increases the maximum displacement as well as the efficiency; the optimal ciliary motion of higher n also exhibits a higher slope for the shock-like structures (results not shown here). This is again consistent with their analytical *ansatz*, which shows that the efficiency approaches 50% in the limit of the maximum displacement approaches 90 degrees, and the corresponding 'width' of the shock in this limit is infinitely small. The mean

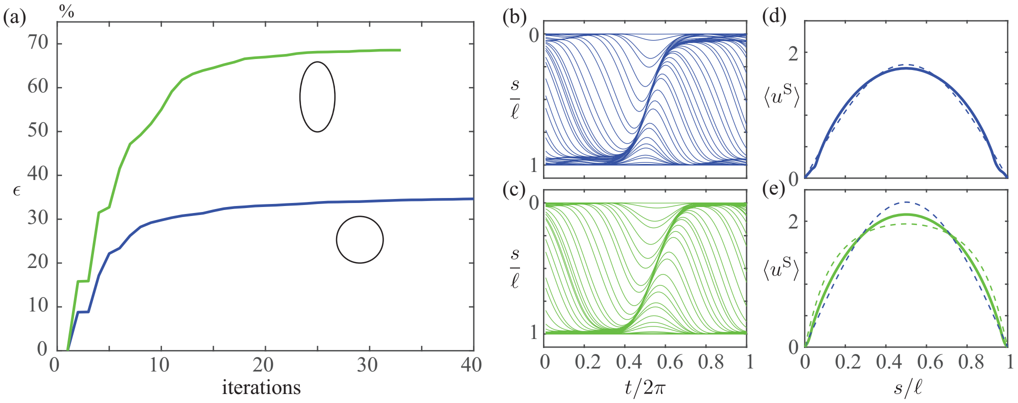

Figure 7.2: Unconstrained optimization history of a spherical swimmer and a prolate swimmer with a 2:1 aspect ratio. The optimal spherical swimmer has an efficiency $\epsilon \approx 35\%$ and swim speed $\langle U \rangle \approx 1.2$. The optimal prolate swimmer has an efficiency $\epsilon \approx 69\%$ and swim speed $\langle U \rangle \approx 1.5$. (a) The efficiency as a function of iterations number. (b) & (c) The ciliary motions of the optimal swimmers. (d) & (e) The time-averaged slip velocities (at Eulerian points) are shown in solid curves. Dashed curves are the time-independent optimal slip velocities of the given shape scaled by the swim speed [73]. Parameters used in the optimization: $m = 25, n = 2$. Number of panels $N_p = 20$, number of sample points $N_s = 80$, number of time steps per period $N_t = 50$. Same below unless otherwise mentioned. Note that the vertical axes of figures (b)&(c) are flipped so that the north pole ($s = 0$) appear on the top of the figure. The corresponding waveforms are known as antiplectic metachronal waves (tips are spread out during the effective stroke and close together during the recovery stroke). The videos of the optimal ciliary motions can be found in the online supplementary material (Movie 1 & 2).

slip velocity of the Eulerian points within each period are almost identical to the optimal *time-independent* slip velocity scaled by the swim speed, as shown in Figure 7.2(d).

The optimal unconstrained prolate spheroidal microswimmer with a 2:1 aspect ratio has an efficiency $\epsilon \approx 69\%$, about twice as high as the spherical microswimmer as shown in Figure 7.2(a). This roughly two-fold increase in efficiency is also observed in the optimal time-independent microswimmers [73]. The optimal ciliary motion is very close to that of the spherical swimmer (Fig. 7.2(b)&(c)) , while the mean slip velocity of the Eulerian points are between the optimal time-independent slip velocity of the same shape and those of a spherical swimmer, as shown in Figure 7.2(e). As a sanity check, swapping the ciliary motions obtained from optimizing the spherical swimmer and the prolate swimmer leads in both cases to lower swimming efficiencies. Specifically, a spherical swimmer with the ciliary motion shown in Figure 7.2(c) has 34% swimming efficiency and a prolate swimmer with the ciliary motion shown in Figure 7.2(b) has 65% swimming efficiency (compared to 35% and

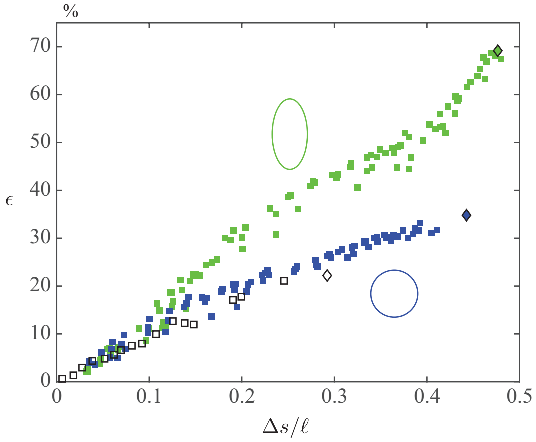

Figure 7.3: Efficiency as a function of maximum displacement of ciliary tips. Blue and green symbols represent spherical and prolate spheroidal swimmers (2:1 aspect ratio) respectively. Diamond symbols are the optimal unconstrained case. Open symbols are optimization results of spherical swimmers taken from [128, Figure 11].

69% using the 'true' optimal ciliary motions, respectively).

We then turn to the case in which the cilia length is constrained by prescribing a bound on the displacement variance (7.38). We control the maximum variance by tuning c in (7.36), and the efficiencies are plotted against the maximum displacement $\Delta s = [\alpha_{\max}(s_0) - \alpha_{\min}(s_0)]/2$ scaled by the total arclength ℓ in Figure 7.3. Three different random initial guesses are used for each c. The unconstrained optimization results for the spherical and prolate spheroidal swimmers are also shown in the figure for reference. Notably, for both the unconstrained swimmers, the length of the cilia is roughly half the total arclength of the generating curve ($\Delta s \approx \ell/2$). In other words, a cilium rooted at the equator would be able to get very close to both poles during the beating cycle. In general, a smaller variance (tighter constraint) leads to a lower efficiency, as expected. The efficiency results of spherical microswimmers closely match those reported by [128]. The efficiencies of the prolate spheroidal microswimmer under constraints are also shown in Figure 7.3. Similar to the spherical microswimmer, the efficiency increases roughly linearly with the scaled cilia length $\Delta s/\ell$, and converges to the kinematically unconstrained optimal microswimmer as the maximum variance c is increased.

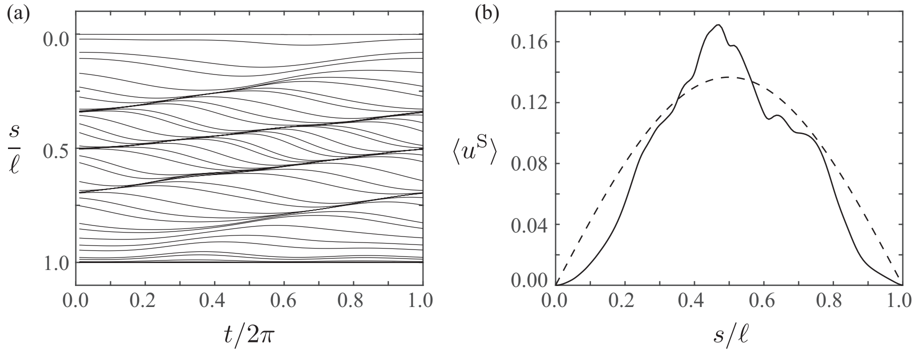

Figure 7.4: Ciliary motion (a) and mean slip velocity (b) for the optimal spherical swimmer with constraint ($\Delta s/\ell \approx 5.0\%$). The efficiency is $\epsilon \approx 6.9\%$, and the swim speed is $\langle U \rangle \approx 0.091$. The swimmer forms multiple waves in the equatorial region, leading to a high slip velocity at $s \approx 0.5\ell$. The motion close to the poles is nearly zero. The dashed curve in (b) is the time-independent optimal slip velocity of the spherical swimmer, scaled by the swim speed. The video of the optimal ciliary motion can be found in the online supplementary material (Movie 3).

It is noteworthy that adding a constraint in the cilia length not only limits the wave amplitudes, but also breaks the single wave with larger amplitude into multiple waves with smaller amplitudes (Fig. 7.4(a)), which resemble the metachronal waves of typical ciliated microswimmers such as *Paramecium*. More interestingly, the mean slip velocity in the constrained case can be qualitatively different from the time-independent optimal slip velocity, as shown in Figure 7.4(b). In particular, the mean slip velocity around the equator is significantly higher than the time-independent slip velocity, while the mean slip velocity near the poles are closer to zero. This can be inferred from the ciliary motions, as the cilia only move slightly near the poles, whereas multiple waves with significant amplitudes travel around the equator within one period.

7.3.3 Non-spheroidal swimmers

We then investigate the effects of shapes on the optimal ciliary motions and the swimming efficiencies. In particular, we examine whether a single wave travelling between north and south poles always maximizes the swimming efficiency, and whether adding a constraint in the cilia length is always detrimental to the swimming efficiency.

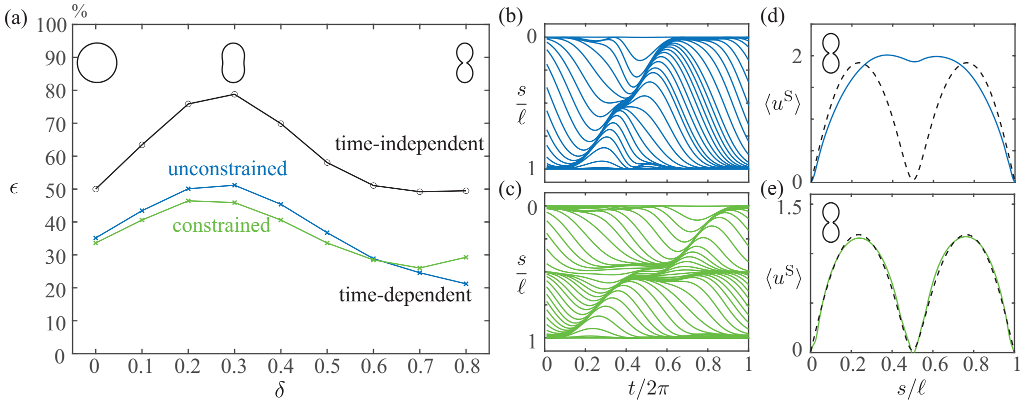

Figure 7.5: Constrained optimizations could lead to more efficient ciliary motions for microswimmers with a thin 'neck' on average. (a): Efficiencies of the microswimmers with various neck widths. The *median* efficiencies of the time-dependent optimizations across 10 randomized initial conditions are shown for each shape in cross symbols '×'. Unconstrained and constrained optimizations ($c = 1$) are depicted in blue and green, respectively. Efficiencies of the microswimmers with time-independent slips are shown, using black circle symbols '$\circ$', as a reference. (b)&(c): Ciliary motions of microswimmers with $\delta = 0.8$ from unconstrained and constrained optimizations from the same initial guess. The swimming efficiencies are 20% and 29%, respectively. (d)&(e): Mean slip velocity corresponding to the ciliary motions in (b)&(c). Blue dashed curves are the optimal time-independent slip velocities scaled by the swim speed. In these simulations, we increase the number of panels $N_p = 40$ to resolve the sharp shape change. The videos of the optimal ciliary motions can be found in the online supplementary material (Movie 4 & 5)

We consider a family of shapes whose generating curves are given by:

$$(x, z) = R(\theta)(\sin\theta, \cos\theta),$$

where $R(\theta) = (1 + \delta\cos 2\theta)$ is a function that makes the radius non-constant, and $\theta \in [0, \pi]$ is the parametric coordinate. For $0 < \delta < 1$, the radius is the smallest at $\theta = \pi/2$, corresponding to a 'neck' around the equator. In the limit $\delta = 0$, the generating curve reduces to a semicircle and the swimmer reduces to the spherical swimmer.

The optimization results are depicted in Figure 7.5 for $0 \leq \delta \leq 0.8$. Some corresponding shapes are shown as insets. The median efficiencies of ten Monte Carlo simulations are plotted for each δ value, and compared against the time-independent efficiencies. For all three cases (constrained, unconstrained, and time-independent), the efficiencies increase as δ increases from 0 to 0.3. This is because increasing δ in this regime makes the shape more elongated. Increasing δ further reduces the efficiencies as the 'neck' at the equator becomes more and more pronounced. Additionally, the unconstrained microswimmers, on average, have better efficiencies than the microswimmers with kinematic-constraints for $0 \leq \delta \leq 0.6$.

Interestingly, unconstrained optimization may result in worse ciliary motions on average when the shape is highly curved, compared to its kinematically-constrained counterpart. Specifically, the constrained microswimmers have higher median efficiencies for $\delta \geq 0.7$. We note that the unconstrained optimizations are likely to be trapped in local optima where the ciliary motion forms a single wave (Fig. 7.5(b)), whereas the constrained optimizations are 'forced' to find the ciliary motion with multiple waves split at the equator (Fig. 7.5(c)), because of the constrained cilia length. Additionally, our numerical results show that a single wave travelling between the north and south poles is not as efficient as two separate waves travelling within each hemisphere for this shape. Figures 7.5(d)&(e) show that the single wave generates a high mean slip velocity at the position where the generating curve bends inward (the equator), whereas the two separate waves generate a mean slip velocity similar to that obtained from the time-independent optimization. In a way, the constraint in cilia length is helping the optimizer to navigate the parameter space.

To better understand the effects of constraints on the highly curved shapes, we present the statistical results of the thin neck microswimmer ($\delta = 0.8$) with various constraints in Figure 7.6. In general, the highest efficiency from the Monte Carlo simulations increases with the constraint for $c \leq 0.8$, similar to the case of spheroidal swimmers (Figure 7.3). Keep increasing c has limited effect on the highest efficiencies, indicating that the constraint is no longer limiting the optimal ciliary motion. The median efficiencies (red horizontal lines), on the other hand, decreases with the constraint if $c \geq 0.8$, consistent with the observation from Figure 7.5. It is worth noting that the constrained optimization is more likely to get stuck in *very* low efficiencies (e.g., the lowest outlier for $c = 0.8$), possibly due to the secondary bump of the penalty function C mentioned earlier.

All data points from the optimization are plotted in Figure 7.6(b) as function of the

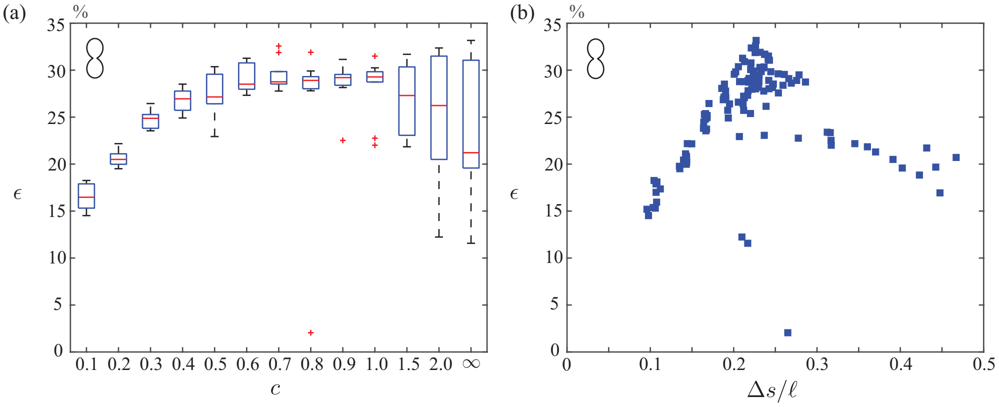

Figure 7.6: Statistical results of thin neck microswimmer of $\delta = 0.8$ with various constraint c for 10 Monte-Carlo simulations. The unconstrained simulation is denoted by $c = \infty$. (a) Efficiencies grouped by the constraint c. For each box, the central mark indicates the median of the 10 random simulations, and the bottom and top edges of the box indicate the 25th and 75th percentiles, respectively. The outliers are denoted by red + symbols. (b) Efficiencies plotted against the maximum displacement $\Delta s/\ell$. The numerical parameter Λ_2 is set to be 10^4 by default. Occasionally the optimization might stop within merely a few iterations, making the ciliary motion stuck in a very inefficient local minimum. Setting Λ_2 to 10^3 for these cases (most of the time) cures the problem.

maximum displacement Δs. The efficiencies grow almost linearly until $\Delta s \approx 0.25\ell$, as in the case of spheroidal swimmers, and decrease for larger Δs. This is another evidence that the optimal ciliary motion for this shape consists of two separate waves traveling within each hemisphere. We want to emphasize that unconstrained optimization can still reach the optimal ciliary motion, as shown in the box of $c = \infty$. However it is more likely to reach the sub-optimal ciliary motion compared to the constrained cases.

7.4 Conclusions and Discussions

In this chapter, we extended the work of [128] and studied the optimal ciliary motion for a microswimmer with arbitrary axisymmetric shape. In particular, the forward problem is solved using a boundary integral method and the sensitivities are derived using an adjoint-based method. The auxiliary function ψ is parameterized using high-order B-spline basis functions in space and a trigonometric polynomial in time. We studied the constrained and unconstrained optimal ciliary motions of microswimmers with a variety of shapes, including spherical, prolate spheroidal, and concave shapes which are narrow around the equator. In all cases, the optimal swimmer displays (one or multiple) traveling waves, reminiscent of the typical metachronal waves observed in ciliated microswimmers. Specifically, for the spherical swimmer with limited cilia length (Fig. 7.4(a)), the ratio between the metachronal wavelength close to the equator and the cilia length could be estimated as $\lambda_{MW}/\Delta s \approx 0.2\ell/0.05\ell = 4$. This ratio lies in the higher end of the data collected in [200, Table 9] for biological ciliates, which reports ratio ranging between 0.5 to 4. Our slightly high ratio estimate may not be surprising after all, as the envelope model prohibits the crossing between neighboring cilia.

We showed that the optimal ciliary motions of prolate microswimmer with a 2:1 aspect ratio are very close to the ones of spherical microswimmer, while the swimming efficiency can increase two-fold. The mean slip velocity of unconstrained microswimmers also tend to follow the optimal time-independent slip velocity, which can be easily computed using our recent work [73].

Most interestingly, we found that constraining the cilia length for some shapes may lead to a better efficiency on average, compared to the unconstrained optimization. It is our conjecture that this counter-intuitive result is because the constraint effectively reduces the size of the parameter space, hence lowering the probability of being trapped in local optima during the optimization. Although the concave shapes studied in Section 7.3.3 are somewhat non-standard, they allows us to gain insights into the effect of local curvature on optimal waveform. Incidentally, these shapes are also observed for ciliates in nature (e.g. during the cell division process).

It is worth pointing out that works on sublayer models (explicitly modeling individual cilia motions) have reported swimming or transport efficiencies in the orders of $0.1 \sim 1\%$ (see, e.g., [40, 94, 141]), *much* lower than the optimal efficiency reported here and others using the envelope models. This large difference can possibly be attributed to the fact that the envelope model we adopted here considers only the energy dissipation *outside* the ciliary layer (into the ambient fluid), while sublayer models in general considers energy dissipation both inside and outside the ciliary layer. Research has shown that the energy dissipation inside the layer could be as high as $90 \sim 95\%$ of the total energy dissipation, due to the large shear rate inside the layer (see, e.g., [99, 94]). We note that it is possible to incorporate energy dissipation *inside* the ciliary layer in the envelope model, as previously done by Vilfan, albeit for a time-independent slip profile. Additionally, the difference could also be due to modeling assumptions on the cilia length and the number of cilia. In particular, the cilia length considered in sublayer models are usually below 1/10 of the body length. [141] showed that the swimming efficiency increases with the cilia length as fast as powers of 3 in the short cilia limit, and the number of cilia also has a significant positive effect on the swimming efficiency (the envelope model assumes a ciliary continuum). Factoring all three factors (energy inside/outside, cilia length, number of cilia) could bridge the gap between the results obtained from these two types of models.

It is without a doubt that maximizing the hydrodynamic swimming efficiency is not the sole objective for biological microswimmers. Other functions such as generating feeding currents [161, 148] and creating flow environment to accelerate mixing for chemical sensing [183, 170, 39, 135] are also important factors to consider as a microswimmer. The effect of such multi-tasking on the ciliary dynamics is not well understood. Nevertheless, our work provides an efficient framework to investigate the hydrodynamically optimal ciliary motions for microswimmers of any axisymmetric shape, and could provide insights into designing artificial microswimmers.

A straightforward extension of our work is to allow more general ciliary motions, e.g., including deformations normal to the surface. Such a swimmer will display time-periodic shape changes and the optimization will require the derivation of shape sensitivities. Additionally, the computational cost would also increase significantly because the matrix in (7.12) needs to be updated at every time step. Our framework is also open to many generalizations and could for example help in accounting for the multiple factors mentioned above, such as mixing for chemical sensing, in the study of optimal ciliary dynamics.

CHAPTER VIII

Concluding remarks

The product integration schemes for singular layer and volume potential developed in this thesis have obvious extensions. In this concluding chapter, we give some brief comments on the possibilities of further improving the schemes, or extending the technologies to more complex problems. Some of which are on going research work.

8.1 Stability Improvement on 2D Product Integration Scheme

In this section, we begin with elaborating on *dimension-agnostic* claim, and how to recover the scheme of Helsing-Ojala [80], as discussed in Section 3.6. Then, we will discuss a new recursion for the product integration scheme using Chebyshev polynomial as basis function. This change makes a difference in both near singular kernel integration and density approximation.

Assume $\tau = 1$, we first rewrite Laplace DLP as differential 1-form

$$\int_\Gamma \frac{(\boldsymbol{r}' - \boldsymbol{r}) \cdot \boldsymbol{n}_{\boldsymbol{r}}}{|\boldsymbol{r}' - \boldsymbol{r}|^2} \mathrm{d}s_{\boldsymbol{r}} = \int_\Gamma \frac{y' - y}{|\boldsymbol{r}' - \boldsymbol{r}|^2} \mathrm{d}x - \frac{x' - x}{|\boldsymbol{r}' - \boldsymbol{r}|^2} \mathrm{d}y \tag{8.1}$$

Then apply Poincaré to accomplish 1-form to 0-form conversion

$$\int_\Gamma \frac{y' - y}{|\boldsymbol{r}' - \boldsymbol{r}|^2} \mathrm{d}x - \frac{x' - x}{|\boldsymbol{r}' - \boldsymbol{r}|^2} \mathrm{d}y = \left[\left(\int_0^1 \frac{y' - ty}{|\boldsymbol{r}' - t\boldsymbol{r}|^2} \mathrm{d}t \right) x - \left(\int_0^1 \frac{x' - tx}{|\boldsymbol{r}' - t\boldsymbol{r}|^2} \mathrm{d}t \right) y \right] \Bigg|_{\Gamma(-1)}^{\Gamma(1)} \tag{8.2}$$

We still use the gradient of harmonic polynomial as our basis function, and borrow the algebra of complex number. It is easy to verify that results similar to Lemma 3.3.3 hold.

It is natural to choose basis $\{z^{k-1} = f^{(k,1)} + if^{(k,2)}, k = 1, 2, \cdots\}$. For high order 1-form to 0-form conversion, we have

$$
\begin{aligned}
p_k =& \int_\Gamma \frac{i\left(\overline{z'-z}\right) n_z}{|z'-z|^2} \left(f^{(k,1)} + 1if^{(k,2)}\right) \mathrm{d}s_{\boldsymbol{r}} \\
=& i\int_\Gamma \left\{ \frac{(y'-y)\, f^{(k,1)} - (x'-x)\, f^{(k,2)}}{|\boldsymbol{r}'-\boldsymbol{r}|^2}\mathrm{d}x - \frac{(x'-x)\, f^{(k,1)} + (y'-y)\, f^{(k,2)}}{|\boldsymbol{r}'-\boldsymbol{r}|^2}\mathrm{d}y \right. \\
&\left. + i\frac{(x'-x)\, f^{(k,1)} + (y'-y)\, f^{(k,2)}}{|\boldsymbol{r}'-\boldsymbol{r}|^2}\mathrm{d}x + i\frac{(y'-y)\, f^{(k,1)} - (x'-x)\, f^{(k,2)}}{|\boldsymbol{r}'-\boldsymbol{r}|^2}\mathrm{d}y \right\} \\
=& 1i\left\{ \left(\int_0^1 \frac{1}{|\boldsymbol{r}'-t\boldsymbol{r}|^2} \left((y'-ty)f^{(k,1)}t^{k-1} - (x'-tx)f^{(k,2)}t^{k-1}\right) \mathrm{d}t \right) x \right. \\
&\left. \left(\int_0^1 \frac{1}{|\boldsymbol{r}'-t\boldsymbol{r}|^2} \left((x'-tx)f^{(k,1)}t^{k-1} + (y'-ty)f^{(k,2)}t^{k-1}\right) \mathrm{d}t \right) y \right\} \Bigg|_{\Gamma(-1)}^{\Gamma(1)} \\
&- \left\{ \left(\int_0^1 \frac{1}{|\boldsymbol{r}'-t\boldsymbol{r}|^2} \left((x'-tx)f^{(k,1)}t^{k-1} + (y'-ty)f^{(k,2)}t^{k-1}\right) \mathrm{d}t \right) x \right. \\
&\left. \left(\int_0^1 \frac{1}{|\boldsymbol{r}'-t\boldsymbol{r}|^2} \left((y'-ty)f^{(k,1)}t^{k-1} - (x'-tx)f^{(k,2)}t^{k-1}\right) \mathrm{d}t \right) y \right\} \Bigg|_{\Gamma(-1)}^{\Gamma(1)}
\end{aligned} \tag{8.3}
$$

If $\Gamma(-1) = -1 + 0 \cdot i$, $\Gamma(1) = 1 + 0 \cdot i$, then $p_{k+1} - (x' + y'i)p_k = \int_0^1 t^{k-1}\mathrm{d}t + \int_0^1 (-t)^{k-1}\mathrm{d}t = \frac{1-(-1)^k}{k}$. We get the recursion for monomial basis function. If instead, we switch to using Chebyshev polynomial $T_k(z)$ as basis function, the resulting recursion can be worked out

$$
\tilde{p}_{k+1} = 2(x' + y'i)\tilde{p}_k - \tilde{p}_{k-1} + \left(\frac{1}{k} - \frac{1}{k-2}\right) - \left(\frac{(-1)^k}{k} - \frac{(-1)^{k-2}}{k-2}\right) \tag{8.4}
$$

We omit the details here. When forward recursion fails for points outside $|z| \le 1$, there is a stable solving process which works similar to the backward recursion in the case of monomial basis function. Various tweaks to the original kernel implementation are needed to adapt to this new formulation. We plan to release the code soon as an improvement to the original 2d close evaluation scheme using monomial as basis function.

8.2 Product Integration Scheme in Axisymmetric Domains

Here we consider an axisymmetric Janus particle suspended in a fluid with viscosity μ in free space Ω_∞. Let $\{\Omega, \Gamma\}$ denote the domain and boundary of the particle respectively.

The chemical concentration C is determined by solving a Laplace Neumann boundary value problem:

$$\nabla^2 C = 0 \quad \text{in} \quad \Omega_\infty, \tag{8.5a}$$

$$\lim_{\|\boldsymbol{r}\|\to\infty} C(\boldsymbol{r}) = 0, \quad \frac{\partial}{\partial \boldsymbol{n}} C = -A(\theta) \quad \text{on} \quad \Gamma, \tag{8.5b}$$

where $\theta \in [0, \pi]$ is the polar angle measured from the south pole to north pole on the generating curve γ.

We follow the standard approach for this problem representing it as a single layer potential defined on Γ to solve for the concentration C. Solutions to this Neumann problems can be obtained through the use of a single layer potential:

$$C(\boldsymbol{r}) = \mathcal{S}[\mu](\boldsymbol{r}), \quad \text{on} \quad \Gamma \tag{8.6}$$

Taking the appropriate limits as $\boldsymbol{r} \to \Gamma$ for the concentration flux, we obtain the following system of boundary integral equations:

$$\left(-\frac{1}{2}I + \mathcal{K}\right)[\mu](\boldsymbol{r}') = -\mathbf{A} \quad \boldsymbol{r} \in \Gamma \tag{8.7}$$

Upon solution of the Neumann problem in (8.7), we then evaluate ∇C and the resulting slip velocities for the given particle configuration.

The problem we address here is how to evaluate these two layer potential accurately for axisymmetric geometry and field, when the kernel is singular or near-singular. We assume symmetry in the azimuthal direction $\varphi \in [0, 2\pi]$. Without loss of generality, we assume that the targets on the surface are located at the cross-section $\varphi = 0$. The positions and normal directions take the following form

$$\boldsymbol{r}' = \begin{pmatrix} \rho' \\ 0 \\ z' \end{pmatrix}, \quad \boldsymbol{n}' = \begin{pmatrix} n'_\rho \\ 0 \\ n'_z \end{pmatrix}, \quad \boldsymbol{r} = \begin{pmatrix} \rho\cos\varphi \\ \rho\sin\varphi \\ z \end{pmatrix} \tag{8.8}$$

The single layer potential can be written as

(8.9)

$$\begin{aligned}
\mathcal{S}[\mu](\boldsymbol{r}') =&\frac{1}{4\pi}\int_0^{2\pi} d\varphi \int_0^{\pi} d\theta \frac{1}{\left(\left(\rho'-\rho\cos\varphi\right)^2+\left(\rho\sin\varphi\right)^2+\left(z'-z\right)^2\right)^{1/2}}\,\mu(\theta)\,\rho\,|r_\theta| \\
=&\frac{1}{4\pi}\int_0^{\pi}\sqrt{\frac{2}{\chi+1}}\text{EllipticK}\left(\frac{2}{\chi+1}\right)\left(2\sqrt{\frac{\rho}{\rho'}}\right)\mu(\theta)|r_\theta|\,d\theta \\
\mathcal{K}[\mu](\boldsymbol{r}') =&-\frac{1}{4\pi}\int_0^{2\pi} d\varphi \int_0^{\pi} d\theta \frac{\left(\rho'-\rho\cos\varphi\right)n'_\rho+\left(z'-z\right)n'_z}{\left(\left(\rho'-\rho\cos\varphi\right)^2+\left(\rho\sin\varphi\right)^2+\left(z'-z\right)^2\right)^{3/2}}\,\mu(\theta)\,\rho\,|r_\theta| \\
=&-\frac{1}{4\pi}\int_0^{\pi}\sqrt{\frac{2}{\chi+1}}\left(\text{EllipticK}\left(\frac{2}{\chi+1}\right)\left(\frac{n'_\rho}{\rho'}\sqrt{\frac{\rho}{\rho'}}\right)\right. \\
&\left.+\text{EllipticE}\left(\frac{2}{\chi+1}\right)\left(2\frac{\left(\rho'-\rho\right)n'_\rho+\left(z-z'\right)n'_z}{|r'-r|^2}\sqrt{\frac{\rho}{\rho'}}-\frac{n'_\rho}{\rho'}\sqrt{\frac{\rho}{\rho'}}\right)\right)\mu(\theta)|r_\theta|\,d\theta
\end{aligned}$$

where $r'=\rho'+iz'$, $n'=n'_\rho+in'_z$, $r=\rho+iz$, and $\chi=1+\frac{|r'-r|^2}{2\rho'\rho}$.

We use the kernel-split idea from [84]. First, we introduce a set of notations:

1. half-integer degree Legendre functions of the second kind $\mathfrak{Q}_{n-\frac{1}{2}}\left(\chi\right)$. We need $\mathfrak{Q}_{-\frac{1}{2}}\left(\chi\right)$ to rewrite EllipticK $\left(\frac{2}{\chi+1}\right)$:

   (8.10) $$\sqrt{\frac{2}{\chi+1}}\text{EllipticK}\left(\frac{2}{\chi+1}\right)=\mathfrak{Q}_{-\frac{1}{2}}\left(\chi\right)$$

2. hypergeometric function ${}_2F_1\left(a,b;c;x\right)=\sum_{k=0}^{\infty}\frac{(a)_k(b)_k}{(c)_k}\frac{x^k}{k!}$. This is related to the coefficient of singular part of EllipticK $\left(\frac{2}{\chi+1}\right)$:

   (8.11) $$\mathfrak{Q}_{-\frac{1}{2}}\left(\chi\right)=-\frac{1}{2}\log\left(\chi-1\right){}_2F_1\left(\frac{1}{2},\frac{1}{2};1;\frac{1-\chi}{2}\right)+\mathfrak{R}\left(\chi\right)$$

   Here $\frac{1-\chi}{2}$ is approximated by

   (8.12) $$\frac{1-\chi}{2}=\frac{|r'-r|^2}{4\rho'\rho}\approx T\left(r',r\right)=\frac{|r'-r|^2}{4\rho'^2}\sum_{k=0}^{3}\left(\frac{\rho'-\rho}{\rho'}\right)^k$$

   And ${}_2F_1\left(a,b;c;x\right)=\sum_{k=0}^{\infty}\frac{(a)_k(b)_k}{(c)_k}\frac{x^k}{k!}$, $|x|<1$, is truncated after four terms

   (8.13) $${}_2F_1\left(a,b;c;x\right)\approx 1+\frac{1}{2^2}x+\frac{1}{2^6}x^2+\frac{1}{3\cdot 2^5}x^3$$

The EllipticE part prevents smooth quadrature from converging for targets near but off-surface, although it has a limiting value. Therefore we use the following relation:

(8.14)

$$
\begin{aligned}
&\sqrt{2(\chi+1)}\text{EllipticE}\left(\frac{2}{\chi+1}\right)\\
=&\chi\sqrt{\frac{2}{\chi+1}}\text{EllipticK}\left(\frac{2}{\chi+1}\right)-\mathfrak{D}_{-\frac{1}{2}}(\chi)\\
=&-\frac{1}{2}\log\left(|r'-r|^2\right)\left(\frac{|r'-r|^2}{4\rho'\rho}-\frac{1}{8}\left(\frac{|r'-r|^2}{4\rho'\rho}\right)^2+\frac{3}{64}\left(\frac{|r'-r|^2}{4\rho'\rho}\right)^3-\frac{25}{1024}\left(\frac{|r'-r|^2}{4\rho'\rho}\right)^4\right)\\
&+\tilde{\mathfrak{R}}(\chi)
\end{aligned}
$$

Then split the kernel into log singularity, Cauchy singularity, and smooth part. This kernel split idea also applies to the singularity in axisymmetric Stokes kernel, which we omit the detailed formula here. Currently, we only have a prototype solver for Stokes single layer potential. For Stokes double layer potential, pressure, and traction kernels, the splitting could involve more complicated cross terms, which requires further derivation.

8.3 Extension to Inhomogeneous Stokes Problems

Here we use two dimensional Stokes kernel as an example to introduce the way of extending our current product integration scheme for volume potentials to other types of singularities. This is for constructing a particular solution to the following BVP for a inhomogeneous Stokes flow

$$-\mu\Delta\boldsymbol{u}(\boldsymbol{r})+\nabla p(\boldsymbol{r})=\boldsymbol{f}(\boldsymbol{r}) \quad \text{and} \quad \nabla\cdot\boldsymbol{u}(\boldsymbol{r})=0 \quad \text{in} \quad \Omega, \tag{8.15a}$$

$$\boldsymbol{u}(\boldsymbol{r})=\boldsymbol{g}(\boldsymbol{r}) \quad \text{on} \quad \Gamma. \tag{8.15b}$$

The particular solution is given by the volume integral of $\boldsymbol{G}(\boldsymbol{r})=-I\log|\boldsymbol{r}|+\frac{\boldsymbol{r}\otimes\boldsymbol{r}}{|\boldsymbol{r}|^2}$ convolved with a vector source function $\boldsymbol{f}(\boldsymbol{r})$ in Ω:

$$\boldsymbol{u}(\boldsymbol{r}')=\int_\Omega \boldsymbol{G}(\boldsymbol{r}'-\boldsymbol{r})\boldsymbol{f}(\boldsymbol{r})\mathrm{d}A \tag{8.16}$$

Integrating $\log|\boldsymbol{r}|$ part follows exactly the routines introduced in Chapter IV for Poisson's equation. Here we rewrite the second terms as partial derivatives of $\log|\boldsymbol{r}|$:

$$\begin{aligned}
&\int_\Omega \frac{(x'-x)\,(x'-x)}{|\boldsymbol{r}'-\boldsymbol{r}|^2} f_1(\boldsymbol{r})\mathrm{d}A \\
\approx &\int_\Omega \frac{(x'-x)\,(x'-x)}{|\boldsymbol{r}'-\boldsymbol{r}|^2} \sum_{i,j} c_{i,j} x^i y^j \mathrm{d}A \\
= &x' \int_\Omega \frac{x'-x}{|\boldsymbol{r}'-\boldsymbol{r}|^2} \sum_{i,j} c_{i,j} x^i y^j \mathrm{d}A - \int_\Omega \frac{x'-x}{|\boldsymbol{r}'-\boldsymbol{r}|^2} \sum_{i,j} c_{i,j} x^{i+1} y^j \mathrm{d}A \\
= &\sum_{i,j} c_{i,j} \left\{ x' \int_\Omega \frac{x'-x}{|\boldsymbol{r}'-\boldsymbol{r}|^2} x^i y^j \mathrm{d}A - \int_\Omega \frac{x'-x}{|\boldsymbol{r}'-\boldsymbol{r}|^2} x^{i+1} y^j \mathrm{d}A \right\} \\
= &\sum_{i,j} c_{i,j} \left\{ x' \frac{\partial}{\partial x'} \int_\Omega \log|\boldsymbol{r}'-\boldsymbol{r}| x^i y^j \mathrm{d}A - \frac{\partial}{\partial x'} \int_\Omega \log|\boldsymbol{r}'-\boldsymbol{r}| x^{i+1} y^j \mathrm{d}A \right\}
\end{aligned} \tag{8.17}$$

To convert $\frac{\partial}{\partial x'}\int_\Omega \log|\boldsymbol{r}'-\boldsymbol{r}| x^i y^j \mathrm{d}A$ and $\frac{\partial}{\partial x'}\int_\Omega \log|\boldsymbol{r}'-\boldsymbol{r}| x^{i+1} y^j \mathrm{d}A$ into 1-forms on $\partial\Omega$, we can using the following recursion.

We present the recursion needed in this case

$$\begin{aligned}
L_{\alpha,x} = &\frac{2\alpha}{\alpha+1}\left[\frac{x}{|\boldsymbol{r}|^2} L_{\alpha-1}(\boldsymbol{r}',\boldsymbol{r}) + \frac{\boldsymbol{r}'\cdot\boldsymbol{r}}{|\boldsymbol{r}|^2} L_{\alpha-1,x}(\boldsymbol{r}',\boldsymbol{r})\right] - \frac{\alpha-1}{\alpha+1}\left[\frac{x'}{|\boldsymbol{r}|^2} L_{\alpha-2}(\boldsymbol{r}',\boldsymbol{r}) + \frac{|\boldsymbol{r}'|^2}{|\boldsymbol{r}|^2} L_{\alpha-2,x}\right] \\
&+ \frac{1}{\alpha+1}\left[\frac{4(x'-x)}{|\boldsymbol{r}|^2}\log|\boldsymbol{r}'-\boldsymbol{r}| + \frac{2(x'-x)}{|\boldsymbol{r}|^2}\right] + \frac{2}{\alpha(\alpha+1)}\frac{x}{|\boldsymbol{r}|^2}
\end{aligned} \tag{8.18}$$

$$\begin{aligned}
L_{\alpha,y} = &\frac{2\alpha}{\alpha+1}\left[\frac{y}{|\boldsymbol{r}|^2} L_{\alpha-1}(\boldsymbol{r}',\boldsymbol{r}) + \frac{\boldsymbol{r}'\cdot\boldsymbol{r}}{|\boldsymbol{r}|^2} L_{\alpha-1,y}(\boldsymbol{r}',\boldsymbol{r})\right] - \frac{\alpha-1}{\alpha+1}\left[\frac{y'}{|\boldsymbol{r}|^2} L_{\alpha-2}(\boldsymbol{r}',\boldsymbol{r}) + \frac{|\boldsymbol{r}'|^2}{|\boldsymbol{r}|^2} L_{\alpha-2,y}\right] \\
&+ \frac{1}{\alpha+1}\left[\frac{4(y'-y)}{|\boldsymbol{r}|^2}\log|\boldsymbol{r}'-\boldsymbol{r}| + \frac{2(y'-y)}{|\boldsymbol{r}|^2}\right] + \frac{2}{\alpha(\alpha+1)}\frac{y}{|\boldsymbol{r}|^2}
\end{aligned} \tag{8.19}$$

To get this recursion started

$$\begin{aligned}
L_{0,x}(\boldsymbol{r}',\boldsymbol{r}) = &-2\frac{x}{|\boldsymbol{r}|^2}\log|\boldsymbol{r}'-\boldsymbol{r}| + 2\left(1-\frac{\boldsymbol{r}'\cdot\boldsymbol{r}}{|\boldsymbol{r}|^2}\right)\frac{x'-x}{|\boldsymbol{r}'-\boldsymbol{r}|^2} + 2\frac{x}{|\boldsymbol{r}|^2}\log|\boldsymbol{r}'| \\
&+ 2\frac{\boldsymbol{r}'\cdot\boldsymbol{r}}{|\boldsymbol{r}|^2}\frac{x'}{|\boldsymbol{r}'|^2} + 2\frac{\mathrm{sign}(\boldsymbol{r}'\times\boldsymbol{r})y}{|\boldsymbol{r}|^2}\left[\tan^{-1}\frac{|\boldsymbol{r}|^2-\boldsymbol{r}'\cdot\boldsymbol{r}}{|\boldsymbol{r}'\times\boldsymbol{r}|} + \tan^{-1}\frac{\boldsymbol{r}'\cdot\boldsymbol{r}}{|\boldsymbol{r}'\times\boldsymbol{r}|}\right] \\
&+ 2\frac{|\boldsymbol{r}'\times\boldsymbol{r}|}{|\boldsymbol{r}|^2}\left[\frac{-x|\boldsymbol{r}'\times\boldsymbol{r}| - (|\boldsymbol{r}|^2-\boldsymbol{r}'\cdot\boldsymbol{r})\,\mathrm{sign}(\boldsymbol{r}'\cdot\boldsymbol{r})y}{|\boldsymbol{r}'\times\boldsymbol{r}|^2 + \left(|\boldsymbol{r}|^2-\boldsymbol{r}'\cdot\boldsymbol{r}\right)^2}\right. \\
&\qquad\left. + \frac{x|\boldsymbol{r}'\times\boldsymbol{r}| - (\boldsymbol{r}'\cdot\boldsymbol{r})\mathrm{sign}(\boldsymbol{r}'\times\boldsymbol{r})y}{|\boldsymbol{r}'\times\boldsymbol{r}|^2 + \left(\boldsymbol{r}'\cdot\boldsymbol{r}\right)^2}\right]
\end{aligned} \tag{8.20}$$

$$L_{1,x}(\boldsymbol{r}', \boldsymbol{r}) = \frac{x}{|\boldsymbol{r}|^2} L_0(\boldsymbol{r}', \boldsymbol{r}) + \frac{\boldsymbol{r}' \cdot \boldsymbol{r}}{|\boldsymbol{r}|^2} L_{0,x}(\boldsymbol{r}', \boldsymbol{r}) + 2\frac{x' - x}{|\boldsymbol{r}|^2} \log|\boldsymbol{r}' - \boldsymbol{r}| - 2\frac{x'}{|\boldsymbol{r}|^2} \log|\boldsymbol{r}'| \tag{8.21}$$

$$\begin{aligned} L_{0,y}(\boldsymbol{r}', \boldsymbol{r}) = &- 2\frac{y}{|\boldsymbol{r}|^2} \log|\boldsymbol{r}' - \boldsymbol{r}| + 2\left(1 - \frac{\boldsymbol{r}' \cdot \boldsymbol{r}}{|\boldsymbol{r}|^2}\right) \frac{y' - y}{|\boldsymbol{r}' - \boldsymbol{r}|^2} + 2\frac{y}{|\boldsymbol{r}|^2} \log|\boldsymbol{r}'| + 2\frac{\boldsymbol{r}' \cdot \boldsymbol{r}}{|\boldsymbol{r}|^2} \frac{y'}{|\boldsymbol{r}'|^2} \\ &+ 2\frac{\operatorname{sign}(\boldsymbol{r}' \times \boldsymbol{r})x}{|\boldsymbol{r}|^2} \left[\tan^{-1} \frac{|\boldsymbol{r}|^2 - \boldsymbol{r}' \cdot \boldsymbol{r}}{|\boldsymbol{r}' \times \boldsymbol{r}|} + \tan^{-1} \frac{\boldsymbol{r}' \cdot \boldsymbol{r}}{|\boldsymbol{r}' \times \boldsymbol{r}|}\right] \\ &+ 2\frac{|\boldsymbol{r}' \times \boldsymbol{r}|}{|\boldsymbol{r}|^2} \left[\frac{-y|\boldsymbol{r}' \times \boldsymbol{r}| - (|\boldsymbol{r}|^2 - \boldsymbol{r}' \cdot \boldsymbol{r}) \operatorname{sign}(\boldsymbol{r}' \cdot \boldsymbol{r})x}{|\boldsymbol{r}' \times \boldsymbol{r}|^2 + (|\boldsymbol{r}|^2 - \boldsymbol{r}' \cdot \boldsymbol{r})^2} \right. \\ &\qquad \left. + \frac{y|\boldsymbol{r}' \times \boldsymbol{r}| - (\boldsymbol{r}' \cdot \boldsymbol{r})\operatorname{sign}(\boldsymbol{r}' \times \boldsymbol{r})x}{|\boldsymbol{r}' \times \boldsymbol{r}|^2 + (\boldsymbol{r}' \cdot \boldsymbol{r})^2}\right] \end{aligned} \tag{8.22}$$

$$L_{1,y}(\boldsymbol{r}', \boldsymbol{r}) = \frac{y}{|\boldsymbol{r}|^2} L_0(\boldsymbol{r}', \boldsymbol{r}) + \frac{\boldsymbol{r}' \cdot \boldsymbol{r}}{|\boldsymbol{r}|^2} L_{0,x}(\boldsymbol{r}', \boldsymbol{r}) + 2\frac{y' - y}{|\boldsymbol{r}|^2} \log|\boldsymbol{r}' - \boldsymbol{r}| - 2\frac{y'}{|\boldsymbol{r}|^2} \log|\boldsymbol{r}'| \tag{8.23}$$

Then the rest, i.e. domain discretization, source term approximation, follows pretty much the discussion from Chapter IV. We have chosen the 2D analytic "body force flow" solution in [20], given by: (there are typos in the originally provided solution $\boldsymbol{u} = 2\{-x^2y, xy^2\}, \quad p = \sin(xy), \quad \boldsymbol{f} = 4\mu\{y(1 + \cos(xy)), -x(1 + \cos(xy))\}$)

$$\boldsymbol{u} = \{-x^2y, xy^2\}, \quad p = 2\mu \sin(xy), \quad \boldsymbol{f} = 2\mu\{y(1 + \cos(xy)), -x(1 - \cos(xy))\} \tag{8.24}$$

This can be generalized to solving the modified Stokes equations with a forcing term:

$$\alpha^2\boldsymbol{u}(\boldsymbol{r}) - \mu\Delta\boldsymbol{u}(\boldsymbol{r}) + \nabla p(\boldsymbol{r}) = \boldsymbol{f}(\boldsymbol{r}) \quad \text{and} \quad \nabla \cdot \boldsymbol{u}(\boldsymbol{r}) = 0 \quad \text{in} \quad \Omega, \tag{8.25a}$$

$$\boldsymbol{u}(\boldsymbol{r}) = \boldsymbol{g}(\boldsymbol{r}) \quad \text{on} \quad \Gamma. \tag{8.25b}$$

In addition to velocity, there is more work to be done to get a robust solver for computing pressure and traction in the presence of inhomogeneous forcing term to get a complete set of solution variables. Finally, we also plan on exploring the possibilities of extending this product integration scheme to a formulation of using space-time fundamental solution for unsteady Stokes flow.

APPENDICES

A.1 Pressure and traction in terms of contour integrals

Here, we give formulae for the traction vector $\mathbf{T}$ induced at a target point with given surface normal, and the associated pressure field p, when the velocity field is represented by a Stokes single or double layer potential. The goal is to write the traction and pressure in terms of the four contour integrals of Sec. 2.2.2, to which close-evaluation methods of Sec. 2.3.3 may then be applied. This enables uniformly accurate force calculations on bodies, or solution of traction BVPs. We use the notation of Sec. 2.2: recall that $\boldsymbol{n}_{\boldsymbol{r}'}$ and $\boldsymbol{n}_{\boldsymbol{r}}$ are the normal vectors at the target $\boldsymbol{r}'$ and source $\boldsymbol{r}$ respectively, $\rho = |\boldsymbol{r}' - \boldsymbol{r}|$, and I denotes the 2×2 identity operator. We also denote $\boldsymbol{d} = \boldsymbol{r}' - \boldsymbol{r}$.

We first consider the single layer potential (2.3). Its traction is

$$\mathbf{T}^S[\boldsymbol{\sigma}](\boldsymbol{r}') = -\frac{1}{\pi}\int_\Gamma \frac{\boldsymbol{d}\cdot\boldsymbol{n}_{\boldsymbol{r}'}}{\rho^2}\,\frac{\boldsymbol{d}\otimes\boldsymbol{d}}{\rho^2}\boldsymbol{\sigma}(\boldsymbol{r})\,\mathrm{d}s_{\boldsymbol{r}}, \tag{A.1.1}$$

which turns out to be the negative of the Stokes DLP (2.4) with $\boldsymbol{n}_{\boldsymbol{r}}$ replaced by $\boldsymbol{n}_{\boldsymbol{r}'}$. While (2.17) is no longer useful in this case, we can instead write the traction as

$$\mathbf{T}^S[\boldsymbol{\sigma}](\boldsymbol{r}') = -\frac{1}{\pi}\int_\Gamma (\boldsymbol{d}\cdot\boldsymbol{\sigma}(\boldsymbol{r}))\frac{\boldsymbol{d}}{\rho^4}(\boldsymbol{d}\cdot\boldsymbol{n}_{\boldsymbol{r}'})\,\mathrm{d}s_{\boldsymbol{r}} \tag{A.1.2}$$

and use the slightly different identity

$$\nabla_{\boldsymbol{r}'}\left(\frac{\boldsymbol{d}\cdot\boldsymbol{\sigma}}{\rho^2}\right) = \frac{\boldsymbol{\sigma}}{\rho^2} - (\boldsymbol{d}\cdot\boldsymbol{\sigma})\frac{2\boldsymbol{d}}{\rho^4} \tag{A.1.3}$$

to write the traction kernel as

$$\begin{aligned}\mathbf{T}^S[\boldsymbol{\sigma}](\boldsymbol{r}') \;&= \frac{-1}{2\pi}\int_\Gamma \frac{\boldsymbol{d}\cdot\boldsymbol{n}_{\boldsymbol{r}'}}{\rho^2}\boldsymbol{\sigma}\,\mathrm{d}s_{\boldsymbol{r}} + \frac{1}{2\pi}(\boldsymbol{r}'\cdot\boldsymbol{n}_{\boldsymbol{r}'})\nabla\int_\Gamma \frac{\boldsymbol{d}\cdot\boldsymbol{\sigma}}{\rho^2}\,\mathrm{d}s_{\boldsymbol{r}}\\ &\quad - \frac{1}{2\pi}n_1'\nabla\int_\Gamma \frac{\boldsymbol{d}\cdot\boldsymbol{\sigma}}{\rho^2}x\,\mathrm{d}s_{\boldsymbol{r}} - \frac{1}{2\pi}n_2'\nabla\int_\Gamma \frac{\boldsymbol{d}\cdot\boldsymbol{\sigma}}{\rho^2}y\,\mathrm{d}s_{\boldsymbol{r}}\ ,\end{aligned} \tag{A.1.4}$$

where $(n_1', n_2') =: \boldsymbol{n}_{\boldsymbol{r}'}$ are the two components of $\boldsymbol{n}_{\boldsymbol{r}'}$. As did in the case of velocity potentials, we can concisely write (A.1.4) as

$$\mathbf{T}^S[\boldsymbol{\sigma}] = \big((\mathcal{S}[\sigma_1])_n + i(\mathcal{S}[\sigma_2])_n\big) + \frac{1}{2\pi}\left(\mathrm{Re}(z'/n_z')\overline{I_H[\sigma/n_z]} - n_1'\overline{I_H[\sigma x/n_z]} - n_2'\overline{I_H[\sigma y/n_z]}\right), \tag{A.1.5}$$

where all the $\mathbb{R}^2$ vectors are now understood as complex numbers in $\mathbb{C}$, the over line in $\overline{I_H[\cdot]}$ denotes the complex conjugate of $I_H[\cdot]$ and the dot product $\boldsymbol{r}'\cdot\boldsymbol{n}_{\boldsymbol{r}'} = \mathrm{Re}(z'/n_z')$ is due to the fact that $1 = |\boldsymbol{n}_{\boldsymbol{r}'}|^2 = n_z'\overline{n_z'}$.

The single layer pressure associated to (2.3) is

$$\mathrm{P}^S[\boldsymbol{\sigma}](\boldsymbol{r}') = \frac{1}{2\pi}\int_\Gamma \frac{\boldsymbol{d}\cdot\boldsymbol{\sigma}}{\rho^2}\,\mathrm{d}s_{\boldsymbol{r}}\ , \tag{A.1.6}$$

which again in the complex plane can be written as

$$\mathrm{P}^S[\boldsymbol{\sigma}] = \mathrm{Re}\frac{i}{2\pi}I_C[\sigma/n_z]. \tag{A.1.7}$$

We now turn to the Stokes double layer potential (2.4). The traction kernel and its associated pressure kernel are given by [120, (5.27)] [16, (3.37)],

(A.1.8)

$$\begin{aligned}\mathbf{T}^D[\boldsymbol{\sigma}](\boldsymbol{r}') =& \frac{\mu}{\pi}\int_\Gamma \Bigg(-8\frac{\boldsymbol{d}\otimes\boldsymbol{d}}{\rho^6}(\boldsymbol{d}\cdot\boldsymbol{n}_{\boldsymbol{r}'})(\boldsymbol{d}\cdot\boldsymbol{n}_{\boldsymbol{r}}) + \frac{\boldsymbol{d}\otimes\boldsymbol{n}_{\boldsymbol{r}'}}{\rho^4}(\boldsymbol{d}\cdot\boldsymbol{n}_{\boldsymbol{r}}) + \frac{\boldsymbol{d}\otimes\boldsymbol{d}}{\rho^4}(\boldsymbol{n}_{\boldsymbol{r}'}\cdot\boldsymbol{n}_{\boldsymbol{r}}) \\ &+ I\frac{1}{\rho^4}(\boldsymbol{d}\cdot\boldsymbol{n}_{\boldsymbol{r}'})(\boldsymbol{d}\cdot\boldsymbol{n}_{\boldsymbol{r}}) + \frac{\boldsymbol{n}_{\boldsymbol{r}}\otimes\boldsymbol{d}}{\rho^4}(\boldsymbol{d}\cdot\boldsymbol{n}_{\boldsymbol{r}'}) + \frac{\boldsymbol{n}_{\boldsymbol{r}'}\otimes\boldsymbol{n}_{\boldsymbol{r}}}{\rho^2}\Bigg)\boldsymbol{\sigma}(\boldsymbol{r})\,\mathrm{d}s_{\boldsymbol{r}} \\ \mathrm{P}^D[\boldsymbol{\sigma}](\boldsymbol{r}') =& \frac{\mu}{\pi}\int_\Gamma\left(-\frac{\boldsymbol{n}_{\boldsymbol{r}}\cdot\boldsymbol{\sigma}(\boldsymbol{r})}{\rho^2} + 2\frac{\boldsymbol{d}\cdot\boldsymbol{\sigma}(\boldsymbol{r})}{\rho^4}(\boldsymbol{r}\cdot\boldsymbol{n}_{\boldsymbol{r}})\right)\mathrm{d}s_{\boldsymbol{r}}.\end{aligned}$$

The corresponding boundary integral operators $\mathbf{T}^D[\boldsymbol{\sigma}](\boldsymbol{r}')$ and $\mathrm{P}^D[\boldsymbol{\sigma}](\boldsymbol{r}')$ are hyper-singular. Following some intricate derivation, we can express this operator in terms of the Laplace double layer potential:

$$\begin{aligned}\frac{1}{\mu}\mathbf{T}^D[\boldsymbol{\sigma}] =& -2\left(\boldsymbol{r}'\cdot\boldsymbol{n}_{\boldsymbol{r}'}\nabla\nabla\mathcal{D}[\boldsymbol{\sigma}] - n_1'\nabla\nabla\mathcal{D}[x\boldsymbol{\sigma}] - n_2'\nabla\nabla[y\boldsymbol{\sigma}]\right) \\ &+ 3I(\boldsymbol{n}_{\boldsymbol{r}'}\cdot\nabla\mathcal{D})[\boldsymbol{\sigma}] - (\boldsymbol{n}_{\boldsymbol{r}'}\otimes\nabla\mathcal{D})[\boldsymbol{\sigma}] - (\nabla\mathcal{D}\otimes\boldsymbol{n}_{\boldsymbol{r}'})[\boldsymbol{\sigma}] \\ &+ \begin{pmatrix}1 & \\ & -1\end{pmatrix}(\boldsymbol{n}_{\boldsymbol{r}'}\cdot\nabla\mathcal{D} + \boldsymbol{n}_{\boldsymbol{r}'}\otimes\nabla\mathcal{D} - \nabla\mathcal{D}\otimes\boldsymbol{n}_{\boldsymbol{r}'})\left[\frac{\bar{n}_z}{n_z}\boldsymbol{\sigma}\right], \\ \mathrm{P}^D[\boldsymbol{\sigma}] =& -2\mu\left(\frac{\partial}{\partial x'}\mathcal{D}[\sigma_1] + \frac{\partial}{\partial y'}\mathcal{D}[\sigma_2]\right) \\ =& \frac{\mu}{\pi}\left(\mathrm{Im}\left(I_H(\sigma_1)\right) + \mathrm{Re}\left(I_H(\sigma_2)\right)\right).\end{aligned} \tag{A.1.9}$$

Here $\frac{\bar{n}_z}{n_z}\boldsymbol{\sigma} = [\frac{\bar{n}_z}{n_z}\sigma_1, \frac{\bar{n}_z}{n_z}\sigma_2]^T$, ∇ is short for $\nabla_{\boldsymbol{r}'}$, and $\nabla\nabla$ is the Hessian tensor. The gradients of Laplace double-layer potentials needed above are expressed in terms of Hadamard integrals using (2.15). The Hessians are given in terms of supersingular integrals as follows:

$$\nabla\nabla\mathcal{D}[\boldsymbol{\sigma}](\boldsymbol{r}') = \begin{pmatrix} \mathrm{Re}\frac{i}{\pi}\left(I_S(\sigma_1)\right)(\boldsymbol{r}') & -\mathrm{Im}\frac{i}{\pi}\left(I_S(\sigma_2)\right)(\boldsymbol{r}') \\ -\mathrm{Im}\frac{i}{\pi}\left(I_S(\sigma_1)\right)(\boldsymbol{r}') & -\mathrm{Re}\frac{i}{\pi}\left(I_S(\sigma_2)\right)(\boldsymbol{r}') \end{pmatrix}. \tag{A.1.10}$$

The close evaluation formulae for these are in Sec. 2.3.3.

To validate the above formulae, we include in Fig. 2.4(b–c) the convergence of the maximum error in pressure and traction for the smooth domain of Example 1 from Sec. 2.5. The convergence rate is very similar to that of velocity albeit a loss of 1–2 digits, which is expected due to the extra derivatives.

B.2 Proof of Lemma 3.3.2

Proof. Denoting $P(\boldsymbol{r}) = \int_0^1 (tzg_2(t\boldsymbol{r}) - tyg_3(t\boldsymbol{r}))\,\mathrm{d}t$, $Q(\boldsymbol{r}) = \int_0^1 (txg_3(t\boldsymbol{r}) - tzg_1(t\boldsymbol{r}))\,\mathrm{d}t$, and $R(\boldsymbol{r}) = \int_0^1 (tyg_1(t\boldsymbol{r}) - txg_2(t\boldsymbol{r}))\,\mathrm{d}t$, the exterior derivative of (3.16), $\omega = P(\boldsymbol{r})\mathrm{d}x + Q(\boldsymbol{r})\mathrm{d}y + R(\boldsymbol{r})\mathrm{d}z$, is

$$\mathrm{d}\omega = \left(\frac{\partial R}{\partial y} - \frac{\partial Q}{\partial z}\right)\mathrm{d}y \wedge \mathrm{d}z + \left(\frac{\partial P}{\partial z} - \frac{\partial R}{\partial x}\right)\mathrm{d}z \wedge \mathrm{d}x + \left(\frac{\partial Q}{\partial x} - \frac{\partial P}{\partial y}\right)\mathrm{d}x \wedge \mathrm{d}y. \tag{B.2.1}$$

The first term, $\frac{\partial R}{\partial y} - \frac{\partial Q}{\partial z}$, can be expanded as

$$\begin{aligned}
&\frac{\partial}{\partial y}\left(\int_0^1 (tyg_1(t\boldsymbol{r}) - txg_2(t\boldsymbol{r}))\,\mathrm{d}t\right) - \frac{\partial}{\partial z}\left(\int_0^1 (txg_3(t\boldsymbol{r}) - tzg_1(t\boldsymbol{r}))\,\mathrm{d}t\right) \\
&= \int_0^1 \big(tg_1(t\boldsymbol{r}) + t^2yg_{1,y}(t\boldsymbol{r}) - t^2xg_{2,y}(t\boldsymbol{r})\big)\,\mathrm{d}t - \int_0^1 \big(t^2xg_{3,z}(t\boldsymbol{r}) - tg_1(t\boldsymbol{r}) - t^2zg_{1,z}(t\boldsymbol{r})\big)\,\mathrm{d}t \\
&= \int_0^1 \big(2tg_1(t\boldsymbol{r}) + t^2yg_{1,y}(t\boldsymbol{r}) + t^2zg_{1,z}(t\boldsymbol{r}) - t^2x\left(g_{2,y}(t\boldsymbol{r}) + g_{3,z}(t\boldsymbol{r})\right)\big)\,\mathrm{d}t \\
&= \int_0^1 \big(2tg_1(t\boldsymbol{r}) + t^2yg_{1,y}(t\boldsymbol{r}) + t^2zg_{1,z}(t\boldsymbol{r}) + t^2xg_{1,x}(t\boldsymbol{r})\big)\,\mathrm{d}t \qquad (\text{since} \quad \nabla\cdot\boldsymbol{g} = 0) \\
&= \int_0^1 \frac{d}{\mathrm{d}t}(t^2g_1(t\boldsymbol{r}))\mathrm{d}t = g_1(\boldsymbol{r}).
\end{aligned}$$

Treating the other two terms, $\frac{\partial P}{\partial z} - \frac{\partial R}{\partial x}$ and $\frac{\partial Q}{\partial x} - \frac{\partial P}{\partial y}$, similarly, we get the result $\mathrm{d}\omega = g_1\mathrm{d}y \wedge \mathrm{d}z + g_2\mathrm{d}z \wedge \mathrm{d}x + g_3\mathrm{d}x \wedge \mathrm{d}y$. □

C.3 Recurrence relations for evaluating moments

Here, we present recurrence relations for evaluating moments of high-order monic polynomials w.r.t certain kernels that arise in 1-form construction. Together with M_k that is required in (3.32), we need L_k and N_k, defined below, for evaluating other Laplace layer potentials:

(C.3.1)

$$L_k(\boldsymbol{r}', \boldsymbol{r}) = \int_0^1 \frac{t^k}{|t\boldsymbol{r} - \boldsymbol{r}'|^5} \mathrm{d}t, \quad M_k(\boldsymbol{r}', \boldsymbol{r}) = \int_0^1 \frac{t^k}{|t\boldsymbol{r} - \boldsymbol{r}'|^3} \mathrm{d}t, \quad N_k(\boldsymbol{r}', \boldsymbol{r}) = \int_0^1 \frac{t^k}{|t\boldsymbol{r} - \boldsymbol{r}'|} \mathrm{d}t$$

Using integration by parts and after some algebra, we can arrive at the following recurrences for evaluating the above moments:

$$\text{(C.3.2)} \qquad \begin{cases} L_k = \frac{2\boldsymbol{r}'\cdot\boldsymbol{r}}{|\boldsymbol{r}|^2} L_{k-1} - \frac{|\boldsymbol{r}'|^2}{|\boldsymbol{r}|^2} L_{k-2} + \frac{1}{|\boldsymbol{r}|^2} M_{k-2} \\ M_k = \frac{\boldsymbol{r}'\cdot\boldsymbol{r}}{|\boldsymbol{r}|^2} M_{k-1} + \frac{k-1}{|\boldsymbol{r}|^2} N_{k-2} - \frac{1}{|\boldsymbol{r}|^2} \frac{t^{k-1}}{|t\boldsymbol{r}-\boldsymbol{r}'|}\Big|_0^1 \\ N_k = \frac{2k-1}{k} \frac{\boldsymbol{r}'\cdot\boldsymbol{r}}{|\boldsymbol{r}|^2} N_{k-1} - \frac{k-1}{k} \frac{|\boldsymbol{r}'|^2}{|\boldsymbol{r}|^2} N_{k-2} + \frac{1}{|\boldsymbol{r}|^2} \frac{t^{k-1}|t\boldsymbol{r}-\boldsymbol{r}'|}{k}\Big|_0^1 \end{cases}$$

The base conditions for these recurrences can also easily be derived as

(C.3.3)

$$N_0 = \frac{1}{|\boldsymbol{r}|} \left(\log\left(|\boldsymbol{r}||\boldsymbol{r} - \boldsymbol{r}'| + |\boldsymbol{r}|^2 - (\boldsymbol{r}' \cdot \boldsymbol{r}) \right) - \log\left(|\boldsymbol{r}'||\boldsymbol{r}| - (\boldsymbol{r}' \cdot \boldsymbol{r}) \right) \right),$$

$$N_1 = \frac{1}{|\boldsymbol{r}|^2} \left(|\boldsymbol{r} - \boldsymbol{r}'| - |\boldsymbol{r}'| \right) + \frac{(\boldsymbol{r}' \cdot \boldsymbol{r})\, N_0}{|\boldsymbol{r}|^2},$$

$$M_0 = \frac{|\boldsymbol{r}|}{|\boldsymbol{r}|^2|\boldsymbol{r}'|^2 - (\boldsymbol{r}' \cdot \boldsymbol{r})^2} \left(\frac{|\boldsymbol{r}|^2 - (\boldsymbol{r}' \cdot \boldsymbol{r})}{|\boldsymbol{r}||\boldsymbol{r} - \boldsymbol{r}'|} + \frac{\boldsymbol{r}' \cdot \boldsymbol{r}}{|\boldsymbol{r}||\boldsymbol{r}'|} \right),$$

$$M_1 = \frac{1}{(\boldsymbol{r}' \cdot \boldsymbol{r})^2 - |\boldsymbol{r}'|^2|\boldsymbol{r}|^2} \left(\frac{-\boldsymbol{r}' \cdot \boldsymbol{r} + |\boldsymbol{r}'|^2}{|\boldsymbol{r} - \boldsymbol{r}'|} - |\boldsymbol{r}'| \right),$$

$$L_0 = \frac{1}{\left(|\boldsymbol{r}'|^2|\boldsymbol{r}|^2 - (\boldsymbol{r}' \cdot \boldsymbol{r})^2 \right)^2} \left(|\boldsymbol{r}|^2 \frac{|\boldsymbol{r}|^2 - \boldsymbol{r}' \cdot \boldsymbol{r}}{|\boldsymbol{r} - \boldsymbol{r}'|} - \frac{1}{3} \frac{(|\boldsymbol{r}| - \boldsymbol{r}' \cdot \boldsymbol{r})^3}{|\boldsymbol{r} - \boldsymbol{r}'|^3} + |\boldsymbol{r}|^2 \frac{\boldsymbol{r}' \cdot \boldsymbol{r}}{|\boldsymbol{r}'|} - \frac{1}{3} \frac{(\boldsymbol{r}' \cdot \boldsymbol{r})^3}{|\boldsymbol{r}'|^3} \right),$$

$$L_1 = -\frac{1}{|\boldsymbol{r}|^2} \left(\frac{1}{3} \frac{1}{|\boldsymbol{r} - \boldsymbol{r}'|^3} - \frac{1}{3} \frac{1}{|\boldsymbol{r}'|^3} \right) + \frac{(\boldsymbol{r}' \cdot \boldsymbol{r})}{|\boldsymbol{r}|^2} L_0.$$

D.4 Second-order approximation scheme for Laplace double-layer potential

Here, we provide further details on the steps outlined in Section 3.4.1 by considering the simpler $p = 2$ case and give explicit formulas for all the intermediate operators and functions.

Stage 1: Precomputation Recall that the first step is the change of coordinates wherein $\boldsymbol{r}^{(1,1)}$ becomes the origin and rest of the quadrature nodes are transformed accordingly. In the case of $p = 2$, there are eight unknowns, four elements each of the vectors $C^{(2,1)}$ and $C^{(2,2)}$ (defined in (3.27)). We can explicitly write the two vector equations obtained by applying (3.26) as

$$\begin{aligned} A[f^{(2,1)}](\tilde{\boldsymbol{r}}^{(2,1)})C^{(2,1)} + A[f^{(2,2)}](\tilde{\boldsymbol{r}}^{(2,1)})C^{(2,2)} &= U^{(2,1)}, \\ A[f^{(2,1)}](\tilde{\boldsymbol{r}}^{(2,2)})C^{(2,1)} + A[f^{(2,2)}](\tilde{\boldsymbol{r}}^{(2,2)})C^{(2,2)} &= U^{(2,2)}. \end{aligned} \tag{D.4.1}$$

From (3.27), we can expand the matrix operators as

$$A[f^{(2,1)}](\boldsymbol{r}) = \begin{pmatrix} 0 & -x & 0 & z \\ x & 0 & z & 0 \\ 0 & -z & 0 & -x \\ -z & 0 & x & 0 \end{pmatrix} \quad \text{and} \quad A[f^{(2,2)}](\boldsymbol{r}) = \begin{pmatrix} 0 & 0 & -y & z \\ 0 & 0 & z & y \\ y & -z & 0 & 0 \\ -z & -y & 0 & 0 \end{pmatrix}. \tag{D.4.2}$$

Stage 2: 2-to-1 form conversion and contour integration The 1-form ω for linear case can be carried out relatively easy. We only have two quaternionic 2-forms (we omit $\tilde{}$ in $\tilde{\boldsymbol{r}}$):

$$\begin{cases} \alpha_0^{(2,1)} = \frac{(y'-y)z}{|\boldsymbol{r}'-\boldsymbol{r}|^3}\mathrm{d}y \wedge \mathrm{d}z + \frac{-(z'-z)x-(x'-x)z}{|\boldsymbol{r}'-\boldsymbol{r}|^3}\mathrm{d}z \wedge \mathrm{d}x + \frac{(y'-y)x}{|\boldsymbol{r}'-\boldsymbol{r}|^3}\mathrm{d}x \wedge \mathrm{d}y \\ \alpha_1^{(2,1)} = \frac{(x'-x)x+(z'-z)z}{|\boldsymbol{r}'-\boldsymbol{r}|^3}\mathrm{d}y \wedge \mathrm{d}z + \frac{(y'-y)x}{|\boldsymbol{r}'-\boldsymbol{r}|^3}\mathrm{d}z \wedge \mathrm{d}x + \frac{(z'-z)x-(x'-x)z}{|\boldsymbol{r}'-\boldsymbol{r}|^3}\mathrm{d}x \wedge \mathrm{d}y \\ \alpha_2^{(2,1)} = \frac{(y'-y)x}{|\boldsymbol{r}'-\boldsymbol{r}|^3}\mathrm{d}y \wedge \mathrm{d}z + \frac{(z'-z)z-(x'-x)x}{|\boldsymbol{r}'-\boldsymbol{r}|^3}\mathrm{d}z \wedge \mathrm{d}x + \frac{-(y'-y)z}{|\boldsymbol{r}'-\boldsymbol{r}|^3}\mathrm{d}x \wedge \mathrm{d}y \\ \alpha_3^{(2,1)} = \frac{-(x'-x)z+(z'-z)x}{|\boldsymbol{r}'-\boldsymbol{r}|^3}\mathrm{d}y \wedge \mathrm{d}z + \frac{-(y'-y)z}{|\boldsymbol{r}'-\boldsymbol{r}|^3}\mathrm{d}z \wedge \mathrm{d}x + \frac{-(z'-z)z-(x'-x)x}{|\boldsymbol{r}'-\boldsymbol{r}|^3}\mathrm{d}x \wedge \mathrm{d}y \end{cases} \tag{D.4.3}$$

$$\text{(D.4.4)}\quad \begin{cases} \alpha_0^{(2,2)} = & \frac{(y'-y)z+(z'-z)y}{|\boldsymbol{r}'-\boldsymbol{r}|^3}\mathrm{d}y\wedge\mathrm{d}z + \frac{-(x'-x)z}{|\boldsymbol{r}'-\boldsymbol{r}|^3}\mathrm{d}z\wedge\mathrm{d}x + \frac{-(x'-x)y}{|\boldsymbol{r}'-\boldsymbol{r}|^3}\mathrm{d}x\wedge\mathrm{d}y \\ \alpha_1^{(2,2)} = & \frac{(z'-z)z-(y'-y)y}{|\boldsymbol{r}'-\boldsymbol{r}|^3}\mathrm{d}y\wedge\mathrm{d}z + \frac{(x'-x)y}{|\boldsymbol{r}'-\boldsymbol{r}|^3}\mathrm{d}z\wedge\mathrm{d}x + \frac{-(x'-x)z}{|\boldsymbol{r}'-\boldsymbol{r}|^3}\mathrm{d}x\wedge\mathrm{d}y \\ \alpha_2^{(2,2)} = & \frac{(x'-x)y}{|\boldsymbol{r}'-\boldsymbol{r}|^3}\mathrm{d}y\wedge\mathrm{d}z + \frac{(y'-y)y+(z'-z)z}{|\boldsymbol{r}'-\boldsymbol{r}|^3}\mathrm{d}z\wedge\mathrm{d}x + \frac{(z'-z)y-(y'-y)z}{|\boldsymbol{r}'-\boldsymbol{r}|^3}\mathrm{d}x\wedge\mathrm{d}y \\ \alpha_3^{(2,2)} = & \frac{-(x'-x)z}{|\boldsymbol{r}'-\boldsymbol{r}|^3}\mathrm{d}y\wedge\mathrm{d}z + \frac{-(y'-y)z+(z'-z)y}{|\boldsymbol{r}'-\boldsymbol{r}|^3}\mathrm{d}z\wedge\mathrm{d}x + \frac{-(z'-z)z-(y'-y)y}{|\boldsymbol{r}'-\boldsymbol{r}|^3}\mathrm{d}x\wedge\mathrm{d}y \end{cases}$$

where superscript of α denotes which basis the differential 2-form corresponds to, and subscript corresponds to index of its quaternion pair form. Corresponding 1-forms are then given by,

$$\text{(D.4.5)}\quad \begin{cases} \omega_0^{(2,1)} = & ((xy^2+2xz^2)M_3 - (y'xy+z'xz+x'z^2)M_2)\,\mathrm{d}x \\ & +((yz^2-x^2y)M_3 + (y'x^2-y'z^2)M_2)\,\mathrm{d}y \\ & +((-2x^2z-y^2z)M_3 + (z'x^2+x'xz+y'yz))\,\mathrm{d}z \\ \omega_1^{(2,1)} = & (-xyzM_3 - (z'xy-y'xz-x'yz)M_2)\,\mathrm{d}x \\ & +((x^2z+z^3)M_3 + (z'x^2-2x'xz-z'z^2)M_2)\,\mathrm{d}y \\ & +(-yz^2M_3 + (-y'x^2+x'xy+z'yz)M_2)\,\mathrm{d}z \\ \omega_2^{(2,1)} = & ((x^2z-y^2z-z^3)M_3 + (-x'xz+y'yz+z'z^2)M_2)\,\mathrm{d}x \\ & +(2xyzM_3 - 2y'xzM_2)\,\mathrm{d}y \\ & +((-x^3-xy^2+xz^2)M_3 + (x'x^2+y'xy-z'xz))\,\mathrm{d}z \\ \omega_3^{(2,1)} = & (-x^2yM_3 + (x'xy+cyz-y'z^2)M_2)\,\mathrm{d}x \\ & +((x^3+xz^2)M_3 - (x'x^2+2z'xz-x'z^2)M_2)\,\mathrm{d}y \\ & +(-xyzM_3 + (z'xy+y'xz-x'yz)M_2)\,\mathrm{d}z \end{cases}$$

$$\text{(D.4.6)}\qquad \left\{\begin{aligned}
\omega_0^{(2,2)} = {} & ((xz^2 - xy^2)M_3 + (x'y^2 - x'z^2)M_2)\,\mathrm{d}x \\
& + ((-x^2y - 2yz^2)M_3 + (x'xy + z'yz + y'z^2)M_2)\,\mathrm{d}y \\
& + ((x^2z + 2y^2z)M_3 + (-z'y^2 - x'xz - y'yz))\,\mathrm{d}z \\
\omega_1^{(2,2)} = {} & (-2xyzM_3 + 2x'yzM_2)\,\mathrm{d}x \\
& + ((x^2 - y^2z + z^3)M_3 + (-x'xz + y'yz - z'z^2)M_2)\,\mathrm{d}y \\
& + ((x^2y + y^3 - yz^2)M_3 + (-x'xy - y'y^2 + z'yz)M_2)\,\mathrm{d}z \\
\omega_2^{(2,2)} = {} & ((-y^2z - z^3)M_3 - (z'y^2 - 2y'yz - z'z^2)M_2)\,\mathrm{d}x \\
& + (xyzM_3 + (z'xy - y'xz - x'yz)M_2)\,\mathrm{d}y \\
& + (xz^2M_3 + (-y'xy + x'y^2 - z'xz))\,\mathrm{d}z \\
\omega_3^{(2,2)} = {} & ((-y^3 - yz^2)M_3 + (y'y^2 + 2z'yz - y'z^2)M_2)\,\mathrm{d}x \\
& + (xy^2M_3 - (y'xy + z'xz - x'z^2)M_2)\,\mathrm{d}y \\
& + (xyzM_3 + (-z'xy + y'xz - x'yz)M_2)\,\mathrm{d}z
\end{aligned}\right.$$

Now we have expression for the complete 1-form ω,

$$\text{(D.4.7)}\qquad \omega = \tau\left(\boldsymbol{r}^{(1,1)}\right)\omega_0^{(1,1)} + \mathit{\Omega}^{(2,1)}C^{(2,1)} + \mathit{\Omega}^{(2,2)}C^{(2,2)},$$

where $\mathit{\Omega}^{(k,l)} = [\omega_0^{(k,l)}, \omega_1^{(k,l)}, \omega_2^{(k,l)}, \omega_3^{(k,l)}]$. Lastly, the contour integral $\int_{\partial\tilde{D}} \omega$ is evaluated on each transformed patch.

E.5 Regularized forces

The following is the formula for regularized Stokeslet:

$$\text{(E.5.1)}\quad \widetilde{G}_{ij}(\boldsymbol{r}' - \boldsymbol{r}) = \frac{1}{4\pi}\left(\delta_{ij}\left(\log\frac{1}{d_\epsilon + \epsilon} - \frac{\epsilon\,(d_\epsilon + 2\epsilon)}{d_\epsilon\,(d_\epsilon + \epsilon)}\right) + (x_i - y_i)(x_j - y_j)\frac{d_\epsilon + 2\epsilon}{d_\epsilon(d_\epsilon + \epsilon)^2}\right),$$

where $d_\epsilon = \sqrt{|\boldsymbol{r'} - \boldsymbol{r}|^2 + \epsilon^2}$, ϵ is the regularization parameter, and δ is Kronecker delta. Its associated regularized pressure kernel is

$$\widetilde{P}_j(\boldsymbol{r'} - \boldsymbol{r}) = \frac{\left(r'_j - r_j\right)}{2\pi} \frac{d_\epsilon^2 + \epsilon^2 + \epsilon d_\epsilon}{d_\epsilon^3\left(d_\epsilon + \epsilon\right)}. \tag{E.5.2}$$

To the best of our knowledge, the traction kernel $\widetilde{T}_{ijk}(\boldsymbol{r'}, \boldsymbol{r})$ associated with the regularized force has not been given explicitly for the 2D regularized force used in [36]. After some lengthy but straightforward derivation following $\widetilde{T}_{ijk} = -\delta_{ij}\widetilde{P}_k + \left(\widetilde{G}_{ik,j} + \widetilde{G}_{jk,i}\right)$, we obtain the formula

$$\begin{aligned} \widetilde{T}_{ijk}(\boldsymbol{r'} - \boldsymbol{r}) = &- \frac{(r'_i - r_i)(r'_j - r_j)(r'_k - r_k)}{\pi} \frac{d_\epsilon^2 + 3\epsilon d_\epsilon + \epsilon^2}{d_\epsilon^3 (d_\epsilon + \epsilon)^3} \\ &- \left[\delta_{ij}(r'_k - r_k) + \delta_{ik}(r'_j - r_j) + \delta_{kj}(r'_i - r_i)\right] \frac{\epsilon^2(2d_\epsilon + \epsilon)}{2\pi(d_\epsilon + \epsilon)^2 d_\epsilon^3}. \end{aligned} \tag{E.5.3}$$

It is easy to see that $\widetilde{T}_{ijk}$ converges to the singular traction kernel T_{ijk} in the limit $\epsilon \to 0$; the correction term induced by the regularization appears at $\mathcal{O}(\epsilon^2)$ and higher orders of ϵ.

F.6 Numerical Validation

To validate our boundary integral method, we construct a boundary value problem and test the algorithm against the exact solution. Specifically, we place 50 stokeslets with random strengths inside the inner channel boundary and one stokeslet with random strength at the center of each of the 10 particles as shown in figure F.6.1(a). The flow field $\boldsymbol{u}_{exa}(\Omega)$ created by these stokeslets can be found by evaluating directly using the free-space Green's function. To obtain the numerical solution, we set the rigid body velocity vector $\boldsymbol{U}^*$ to be zero and treat the flow field on the channel walls and the particle surfaces, given by $\boldsymbol{u}_{exa}(\partial\Omega)$, as the boundary conditions on $\partial\Omega$ where $\partial\Omega \equiv \Gamma \cup \gamma$. Symbolically, one can think of $\boldsymbol{u}_{exa}$ as $\boldsymbol{u}^c$ and substitute it into (7.12) to solve for the corresponding density function $\boldsymbol{\mu}$. The numerical solution $\boldsymbol{u}_{num}(\Omega) = \boldsymbol{u}^\Gamma + \boldsymbol{u}^\gamma$ is then found by substituting $\boldsymbol{\mu}$ into (5.8) and (5.16).

The logarithm of absolute error between $\boldsymbol{u}_{exa}$ and $\boldsymbol{u}_{num}$ is shown in figure F.6.1(b) with about 4000 Gauss-Legendre quadrature points on Γ and γ in total. It is noticeable that the algorithm has at least a 14-digit accuracy for most of the locations, and 12-digit accuracy is achieved even close to the particles. The l_∞-norm of the error as a function of number of quadrature points is shown in figure F.6.1(c).

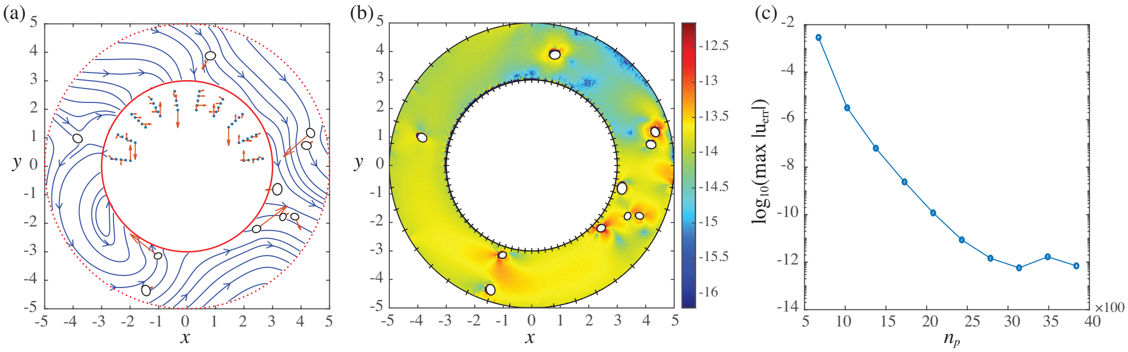

Figure F.6.1: Spatial validation. (a) Flow field generated by 60 stokeslets (red arrows) shown as streamlines. (b) The absolute error between the exact solution and the numerical solution with a total of about 4000 Gaussian quadrature points, color-code represents $\log_{10}(|\boldsymbol{u}_{exa} - \boldsymbol{u}_{num}|)$. (c) The l_∞-norm of the flow field shown as a function of the number of quadrature points.

Next, we place $N = 32$ cilia with phase difference $\Delta\phi = 2\pi/N = \pi/16$ as in (5.1) and use a standard Runge-Kutta 4th order (RK4) scheme to march forward in time. Due to the lack of an exact solution in this case, we test the self convergence rate with respect to Δt. We monitor the motion of a rigid particle of radius $r_p = 0.4$ initially centered at $(0, 4.5)$ for a full cycle $t \in [0, 1]$ and for $\Delta t = \{0.04, 0.02, 0.01\}$. The particle is discretized using 128 quadrature points. At the final time $T = 1$, we measure the following quantities in Table F.6.1:

$$\begin{aligned}
\mathcal{E}_x(T, \Delta t) &= -\log_2 |x_c^{\Delta t}(T) - x_c^{\Delta t/2}(T)| \\
\mathcal{E}_y(T, \Delta t) &= -\log_2 |y_c^{\Delta t}(T) - y_c^{\Delta t/2}(T)| \\
\mathcal{E}_\Theta(T, \Delta t) &= -\log_2 |\Theta^{\Delta t}(T) - \Theta^{\Delta t/2}(T)|
\end{aligned} \tag{F.6.1}$$

where $\Theta(t) = \int_0^t \omega \mathrm{d}t$ is the orientation of the particle. The convergence rate for a passive tracer is also reported in Table F.6.1, with $\mathcal{E}_x$ and $\mathcal{E}_y$ only.

Particle	$\Delta t = 0.04$	$\Delta t = 0.02$	$\Delta t = 0.01$
$\mathcal{E}_x(T, \Delta t)$	13.9948	17.0434	21.6549
$\mathcal{E}_y(T, \Delta t)$	14.6581	17.9964	22.6611
$\mathcal{E}_\Theta(T, \Delta t)$	13.8062	17.6920	21.6093

Tracer	$\Delta t = 0.04$	$\Delta t = 0.02$	$\Delta t = 0.01$
$\mathcal{E}_x(T, \Delta t)$	12.4986	16.5527	22.1399
$\mathcal{E}_y(T, \Delta t)$	12.0234	16.2576	21.3125

Table F.6.1: (Left) Results on the performance of RK4 method applied to evolving the cilia inside a Taylor-Couette device. (Right) Error terms for the particle center at final time $T = 1$.

G.7 Parameter space

We parametrize the slip velocity using a piecewise B-spline approximation. The slip velocity $u^{\mathrm{S}}(t)$ is determined by $(M+1)$ *control points*, $u^{\mathrm{S}}(t_i) = \varphi_i$ for $i = 0, \cdots, M$, and is interpolated by B-spline basis functions between the control points. Here $t \in [0, \pi]$ is a reparametrization of the arc-length s. In theory, we only need to assign control points for t_i between 0 and π to generate an admissible slip velocity by symmetry. In practice, however, we assign control points in the full period $t_i \in [0, 2\pi]$ and impose periodic boundary conditions to determine the spline coefficients, as detailed below.

Let $M = 2N + 2$, where N is the number of *free* control points between 0 and π. Let all control points be equally spaced, we have $t_i = 2\pi i/M$, $i = 0, \cdots, M$. To make sure the slip velocity is axisymmetric, we assign *ghost* control points $\varphi_i = -\varphi_{M-i}$ for $N+1 < i < 2N+2$ and enforce zero conditions at the poles $\varphi_i = 0$, for $i = 0, N+1, 2N+2$.

The general B-spline formulation of order 5 is given by

$$u^{\mathrm{S}}(t) := \sum_{k=-5}^{M-1} \xi_k B_k(t), \quad t \in [0, 2\pi], \tag{G.7.1}$$

where $B_k(t) = B^*_{k,5}(\frac{M}{2\pi}t)$ is a modified k-th B-spline basis function, and $B^*_{k,p}$ is the standard

k-th B-spline basis function of degree p, given by recurrence

$$B^*_{k,0}(t) = \begin{cases} 1, & k \le t < k+1 \\ 0, & \text{otherwise} \end{cases} \tag{G.7.2}$$

$$B^*_{k,p}(t) = \frac{t-k}{p} B^*_{k,p-1}(t) + \frac{p+k+1-t}{p} B^*_{k+1,p-1}(t). \tag{G.7.3}$$

In order to obtain the $(M+5)$ B-spline coefficients ξ_k from the $(M+1)$ control points φ_i, we need four more equations to close the system. Specifically, we use the periodic boundary conditions of the derivatives

$$\frac{\mathrm{d}^n u^{\mathrm{S}}}{\mathrm{d}t^n}(0) = \frac{\mathrm{d}^n u^{\mathrm{S}}}{\mathrm{d}t^n}(2\pi), \quad n = 1, 2, 3, 4. \tag{G.7.4}$$

These system of equations uniquely determine the B-spline coefficient ξ_k from the control points φ_i. The slip velocity $u^{\mathrm{S}}(t)$ along the generating curve could then be found by substituting ξ_k into (G.7.1).

H.8 Numerical validation

The Green's function G and the traction kernel T used in the ansatz (6.7) are defined by

$$\boldsymbol{G}\left(\boldsymbol{r}' - \boldsymbol{r}\right) = \frac{1}{8\pi\mu}\left(I\frac{1}{|\boldsymbol{r}' - \boldsymbol{r}|} + \frac{(\boldsymbol{r}' - \boldsymbol{r}) \otimes (\boldsymbol{r}' - \boldsymbol{r})}{|\boldsymbol{r}' - \boldsymbol{r}|^3}\right), \tag{H.8.1}$$

$$\boldsymbol{n}_{\boldsymbol{r}'}\boldsymbol{T}\left(\boldsymbol{r}' - \boldsymbol{r}\right) = -\frac{3}{4\pi}\frac{(\boldsymbol{r}' - \boldsymbol{r}) \otimes (\boldsymbol{r}' - \boldsymbol{r})}{|\boldsymbol{r}' - \boldsymbol{r}|^5}\left(\boldsymbol{r}' - \boldsymbol{r}\right) \cdot \boldsymbol{n}_{\boldsymbol{r}'}. \tag{H.8.2}$$

Due to the rotational symmetry of Γ, we can transform the layer potentials (6.7) into convolutions on the generating curve γ by integrating analytically in the θ-direction. The integral

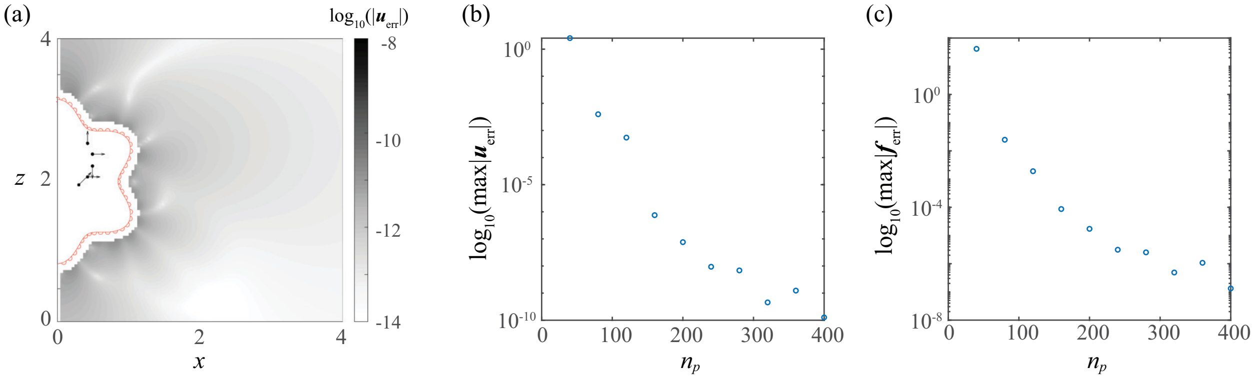

Figure H.8.1: (a) The absolute error between the exact solution and the numerical solution with a total of 400 Gaussian quadrature points; color-code represents $\log_{10}(|\boldsymbol{u}_{exa} - \boldsymbol{u}_{num}|)$. (b) The L_∞-norm of the error in the flow field shown as a function of the number of quadrature points. (c) The L_∞-norm of the traction error shown as a function of the number of quadrature points.

kernels take the following form ([197]):

$$\boldsymbol{G}_\gamma(\boldsymbol{r}' - \boldsymbol{r}) = \frac{1}{8\pi\mu}\int_0^{2\pi} \begin{pmatrix} \frac{\cos\theta}{|\boldsymbol{r}'-\boldsymbol{r}|} + \frac{(x\cos\theta - x')(x - x'\cos\theta)}{|\boldsymbol{r}'-\boldsymbol{r}|^3} & \frac{(x\cos\theta - x')(z-z')}{|\boldsymbol{r}'-\boldsymbol{r}|^3} \\ \frac{(x - x'\cos v)(z-z')}{|\boldsymbol{r}'-\boldsymbol{r}|^3} & \frac{1}{|\boldsymbol{r}'-\boldsymbol{r}|} + \frac{(z-z')^2}{|\boldsymbol{r}'-\boldsymbol{r}|^3} \end{pmatrix} d\theta,$$

$$\boldsymbol{n}_{\boldsymbol{r}'}\boldsymbol{T}_\gamma(\boldsymbol{r}' - \boldsymbol{r}) = -\frac{3}{4\pi}\int_0^{2\pi} \begin{pmatrix} \frac{(x\cos\theta - x')(x - x'\cos\theta)}{|\boldsymbol{r}'-\boldsymbol{r}|^5} & \frac{(x\cos\theta - x')(z-z')}{|\boldsymbol{r}'-\boldsymbol{r}|^5} \\ \frac{(x - x'\cos\theta)(z-z')}{|\boldsymbol{r}'-\boldsymbol{r}|^5} & \frac{(z-z')^2}{|\boldsymbol{r}'-\boldsymbol{r}|^5} \end{pmatrix} \left(n_1\left(x\cos\theta - x'\right) + n_3\left(z - z'\right)\right) d\theta. \tag{H.8.3}$$

The velocity and traction can then be transformed as: $\boldsymbol{u}\left(\boldsymbol{r}'\right) = \int_\gamma \boldsymbol{G}_\gamma\left(\boldsymbol{r}' - \boldsymbol{r}\right)\boldsymbol{\sigma}\left(\boldsymbol{r}\right) x\, \mathrm{d}s_{\boldsymbol{r}}$, $\boldsymbol{f}\left(\boldsymbol{r}'\right) = -\frac{1}{2}\boldsymbol{\sigma}\left(\boldsymbol{r}'\right) + \boldsymbol{n}_{\boldsymbol{r}'}\int_\gamma \boldsymbol{T}_\gamma\left(\boldsymbol{r}', \boldsymbol{r}\right)\boldsymbol{\sigma}\left(\boldsymbol{r}\right) x\, \mathrm{d}s_{\boldsymbol{r}}$. The analytic solution of the integrals (H.8.3) can be found in [197] and [152].

To validate our boundary integral method, we construct a boundary value problem and test the algorithm against the exact solution. As is standard practice, we consider the flow field generated by a set of axisymmetric Stokeslets and the corresponding traction:

$$\boldsymbol{u}_{exa}(\boldsymbol{r}') = \sum_{k=1}^{N} \boldsymbol{G}_\gamma(\boldsymbol{r}' - \boldsymbol{r}_k)\boldsymbol{\sigma}_k x_k, \quad \boldsymbol{f}_{exa}(\gamma) = \boldsymbol{n}_\gamma \sum_{k=1}^{N} \boldsymbol{T}_\gamma(\gamma - \boldsymbol{r}_k)\boldsymbol{\sigma}_k(k) x_k, \tag{H.8.4}$$

where $\{\boldsymbol{r}_k\}$ and $\{\boldsymbol{\sigma}_k\}$ are the location and strength of the k-th Stokeslet. We randomly choose 5 Stokeslets whose locations and strengths are given in Fig. H.8.1(a) by the black arrows and substitute them into (H.8.4) as our reference case.

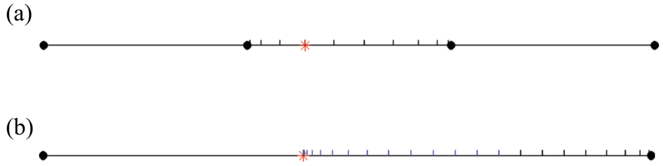

Figure H.8.2: (a) Example of a panel with 10-point Gaussian nodes, and its neighbor panels. The red asterisk is the target. (b) Three panels in (a) are combined into one big panel. The big panel is further divided into two panels by the desired target. Blue grid is a 16th-order Alpert quadrature rule. And black grid is an 8-point smooth quadrature rule.

To obtain the numerical solution, we first evaluate the reference flow field on the generating curve $\boldsymbol{u}_{exa}(\gamma)$, then treat $\boldsymbol{u}_{exa}(\gamma)$ as the boundary condition to obtain the density vector $\boldsymbol{\sigma}$. The generating curve γ is discretized into non-overlapping panels $\gamma = \sum_{p=1}^{N_p} \Lambda_p$. Then on each panel, we place the nodes of a 10-point Gaussian quadrature. The integral operator can then be approximated by the standard Nyström matrix at these collocation points. The logarithmic singularity is resolved with Alpert quadrature using node locations off the Gauss-Legendre grid [74], as illustrated in Fig. H.8.2(a) &(b). Integrals of $\boldsymbol{G}_\gamma(\boldsymbol{r'} - \boldsymbol{r})$ and $\boldsymbol{T}_\gamma(\boldsymbol{r'} - \boldsymbol{r})$ at the desired target, endpoints of two panels in Fig. H.8.2(b), are approximated using correction nodes. Note that two end panels need to be further split adaptively corresponding to north and south poles, until the first and last Gaussian nodes have adjacent neighbors. We subsequently use the density vector $\boldsymbol{\sigma}$ to evaluate the numerical solution $\boldsymbol{u}_{num}(\boldsymbol{r'})$ outside the microswimmer's surface. The traction on the generating curve is evaluated from the same density vector $\boldsymbol{\sigma}$ using the traction kernel $\boldsymbol{f}_{num}(\gamma) = -\frac{1}{2}\boldsymbol{\sigma}(\gamma) + \boldsymbol{n}_\gamma \int_\gamma \boldsymbol{T}_\gamma(\gamma - \boldsymbol{r})\boldsymbol{\sigma}(\boldsymbol{r})\, x \, \mathrm{d}s_{\boldsymbol{r}}$.

The absolute error of the numerical solution $\boldsymbol{u}_{num}$ is shown in Fig. H.8.1(a). As can be observed from Fig. H.8.1(b) &(c), our forward solver achieves 10-digit accuracy in the flow field and 6-digit accuracy for traction with 400 quadrature points on the generating curve. For all the test cases presented in Section 7.3, 600 Gauss-Legendre quadrature points were used.

As a further validation of our numerical scheme, we computed the fluid drag of a family

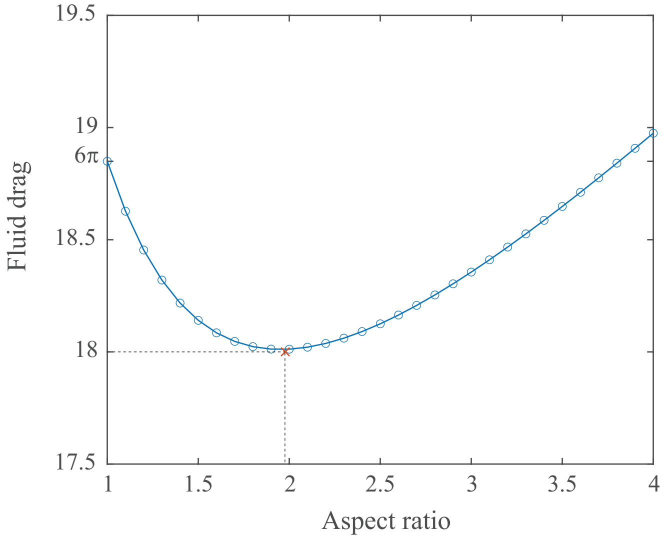

Figure H.8.3: Fluid drag of towing a prolate spheroid with unit speed. All spheroids are of the same volume as the unit sphere. The red cross denotes the fluid drag of the optimal profile that minimizes the fluid drag given by [150].

of prolate and oblate spheroids. The shape that yields the minimal fluid drag is a prolate spheroid with a roughly 2 : 1 aspect ratio (Fig. H.8.3), consistent with the optimal shape obtained previously in [150].

I.9 Generating curves of the shapes used in the chapter

Here, for reproducibility purposes, we list equations of all the generating curves used in this paper. In all cases below, $i = \sqrt{-1}$, $t \in [0, \pi]$ is the polar angle, the equations are defined on the complex plane and the axis of symmetry is the imaginary axis.

- *Spheroids:* $z = \alpha^{-1/3} \sin(t) + i\alpha^{2/3} \cos(t)$, α is the aspect ratio.
- *Wavy shapes:* $z = (1 + 0.15 \cos(kt) \exp(i(\pi/2 - t)))$, $k \in \{3, 4, 5, 6\}$ is the order of the perturbation.
- *Stomatocyte:* $z = (1.5 + \cos t)(\sin(\lambda\pi \sin t) + i \cos(\lambda\pi \sin t)) - 0.5i$, $\lambda \in [0.4, 0.95]$ controls the vertical 'stretchiness' of the shape.
- *Harmonics:* $z = \rho(t) \sin t - i\rho(t) \cos t$, where $\rho(t) = 1 + rY_n^m(t, 0)$, where $Y_n^m(\theta, \varphi)$

is the spherical harmonics of degree n and order m, evaluated at the colatitude θ and longitude φ.

- *Spherocylinder* shapes were generated by simply attaching semi-spherical caps to a cylinder with the same radius and subsequently smoothing using B-splines upto order 5.
- *Snowman* shapes were generated by two spheres of different radii glued together with the centroid distance set to 90% of the sum of the radii, followed by smoothing.

J.10 Derivations of sensitivities

In this Appendix, we include the detail derivations that lead to (7.33) and the explicit expressions of the terms therein.

Recall that the power loss and the swim speed can be written as functionals of ψ, as shown in (7.30). The sensitivities of $\langle \mathbb{P} \rangle$ and $\langle \mathbb{U} \rangle$ can thus be formulated by considering perturbed versions of ψ as in

$$\psi_\eta(x,t) = \psi(x,t) + \eta\hat{\psi}(x,t), \qquad \text{i.e.} \quad \psi_\eta = \psi + \eta\hat{\psi}, \tag{J.10.1}$$

so that the perturbed location s_η at time t of the material particle initially located at s_0 is given by

$$s_\eta = \alpha(s_0, \psi_\eta), \tag{J.10.2}$$

the functional α being unchanged. Similar to (7.29), the perturbed slip velocity $u^{\mathrm{S}}_\eta(s,t)$ satisfies

$$u^{\mathrm{S}}_\eta(s,t) = \partial_\psi \alpha\left(\beta(s,\psi_\eta), \psi_\eta; \dot{\psi}_\eta\right) = \upsilon^{\mathrm{S}}(s,\psi_\eta), \tag{J.10.3}$$

where β, the inverse function of α, is also unchanged.

Notice that u^{S} and u^{S}_{η} given by (7.29) and (J.10.3) are evaluated at the same time t and current location s (the latter being thus reached from different initial positions $\beta(s,\psi)$ and $\beta(s,\psi_{\eta})$). This allows us to define the directional derivative $\dot{\upsilon}^{\mathrm{S}}(s,\psi;\hat{\psi})$ of u^{S} with respect to ψ in the direction $\hat{\psi}$, as a total derivative with respect to η:

$$\dot{\upsilon}^{\mathrm{S}}(s,\psi;\hat{\psi}) := \lim_{\eta\to 0}\frac{1}{\eta}\left[u^{\mathrm{S}}_{\eta}(s,t) - u^{\mathrm{S}}(s,t)\right] = \frac{\mathrm{d}}{\mathrm{d}\eta}\partial_{\psi}\alpha\left(\beta(s,\psi_{\eta}),\psi_{\eta};\dot{\psi}_{\eta}\right)\Big|_{\eta=0} \tag{J.10.4}$$

Carrying out the above differentiation in a straightforward way, we find

$$\begin{aligned}\dot{\upsilon}^{\mathrm{S}}(s,\psi;\hat{\psi}) = {} & \partial_{\psi s}\alpha\left(\beta(s,\psi),\psi;\dot{\psi}\right)\ \partial_{\psi}\beta\left(s,\psi;\hat{\psi}\right) \\ & + \partial_{\psi\psi}\alpha\left(\beta(s,\psi),\psi;\dot{\psi}\,,\,\hat{\psi}\right) + \partial_{\psi}\alpha\left(\beta(s,\psi),\psi;\dot{\hat{\psi}}\right).\end{aligned} \tag{J.10.5}$$

Moreover, for any ψ, the functions α and β are linked through

$$s = \alpha\left(\beta(s,\psi),\psi\right) \tag{J.10.6}$$

which, upon taking the directional derivative in the direction $\hat{\psi}$ and using the chain rule, yields

$$0 = \partial_{s}\alpha\left(\beta(s,\psi),\psi\right)\partial_{\psi}\beta\left(s,\psi;\hat{\psi}\right) + \partial_{\psi}\alpha\left(\beta(s,\psi),\psi;\hat{\psi}\right). \tag{J.10.7}$$

The above equality allows us to eliminate $\partial_{\psi}\beta$ from (J.10.5), to obtain

$$\begin{aligned}\dot{\upsilon}^{\mathrm{S}}(s,\psi;\hat{\psi}) = {} & -\partial_{\psi s}\alpha\left(\beta(s,\psi),\psi;\dot{\psi}\right)\ \frac{\partial_{\psi}\alpha\left(\beta(s,\psi),\psi;\hat{\psi}\right)}{\partial_{s}\alpha\left(\beta(s,\psi),\psi\right)} \\ & + \partial_{\psi\psi}\alpha\left(\beta(s,\psi),\psi\,;\,\dot{\psi}\,,\,\hat{\psi}\right) + \partial_{\psi}\alpha\left(\beta(s,\psi),\psi;\dot{\hat{\psi}}\right).\end{aligned} \tag{J.10.8}$$

In practice, the slip velocity derivative $\dot{\upsilon}^{\mathrm{S}}$ given by (J.10.8) is more conveniently expressed in the initial arclength variable $s_0 = \beta(s,\psi)$. Moreover, in the event that $\psi(s_0,t) = 0$ for some s_0 and t, $\dot{\upsilon}^{\mathrm{S}}$ given by (J.10.8) blows up since $\partial_{s}\alpha(\beta(s,\psi),\psi) = 0$ in this case, whereas $\dot{\upsilon}^{\mathrm{S}}\mathrm{d}s$ remains finite if expressed in terms of s_0 (since $\mathrm{d}s = \partial_{s}\alpha(s_0,\psi)\mathrm{d}s_0$). Upon effecting the change of variable $s = \alpha(s_0, \psi)$ in the integrals (7.26) and (7.28), we obtain

$$\langle\dot{\mathbb{P}}\rangle(\psi;\hat{\psi}) = 4\pi \left\langle \int_\gamma R(\alpha(s_0,\psi))\, \boldsymbol{f}(\alpha(s_0,\psi),t) \cdot \boldsymbol{\tau}(\alpha(s_0,\psi))\, \dot{\upsilon}^{\mathrm{S}}(s,\psi;\hat{\psi})\, \partial_s\alpha(s_0,\psi)\mathrm{d}s_0 \right\rangle \tag{J.10.9}$$

$$\langle\dot{\mathbb{U}}\rangle(\psi;\hat{\psi}) = \frac{-2\pi}{F_0} \left\langle \int_\gamma R(\alpha(s_0,\psi))\, \hat{\boldsymbol{f}}(\alpha(s_0,\psi)) \cdot \boldsymbol{\tau}(\alpha(s_0,\psi))\, \dot{\upsilon}^{\mathrm{S}}(s,\psi;\hat{\psi})\, \partial_s\alpha(s_0,\psi)\mathrm{d}s_0 \right\rangle \tag{J.10.10}$$

where, thanks to (J.10.8), we have used

$$\begin{aligned}
\dot{\upsilon}^{\mathrm{S}}(s,\psi;\hat{\psi})\mathrm{d}s &= \dot{\upsilon}^{\mathrm{S}}(s,\psi;\hat{\psi})\, \partial_s\alpha(s_0,\psi)\mathrm{d}s_0 \\
&= \Big\{ \partial_s\alpha(s_0,\psi)\, \Big[\partial^2_\psi\alpha\left(s_0,\psi;\hat{\psi},\dot{\psi}\right) + \partial_\psi\alpha\left(s_0,\psi;\dot{\hat{\psi}}\right)\Big] \\
&\qquad - \partial_{\psi s}\alpha\left(s_0,\psi;\dot{\psi}\right)\, \partial_\psi\alpha\left(s_0,\psi;\hat{\psi}\right) \Big\}\mathrm{d}s_0.
\end{aligned} \tag{J.10.11}$$

This completes our derivation of (7.33).

For the ciliary motion (7.14) used here, introducing the shorthand notation $I(f,g;s) := \int_0^s f(x)g(x)h$, we have

$$\alpha(s_0,\psi) = \frac{\ell I(\psi,\psi;s_0)}{I(\psi,\psi;\ell)} \tag{J.10.12}$$

$$\partial_s\alpha(s_0,\psi) = \frac{\ell\psi^2(s_0)}{I(\psi,\psi;\ell)} \tag{J.10.13}$$

$$\partial_\psi\alpha\left(s_0,\psi;\hat{\psi}\right) = \frac{2\ell I(\psi,\hat{\psi};s_0)}{I(\psi,\psi;\ell)} - 2\alpha(s_0,\psi)\frac{I(\psi,\hat{\psi};\ell)}{I(\psi,\psi;\ell)} \tag{J.10.14}$$

$$\partial_{s\psi}\alpha\left(s_0,\psi;\dot{\psi}\right) = \frac{2\ell\psi(s_0)\dot{\psi}(s_0)}{I(\psi,\psi;\ell)} - 2\ell\frac{I(\psi,\dot{\psi};\ell)\,\psi^2(s_0)}{\left(I(\psi,\psi;\ell)\right)^2} \tag{J.10.15}$$

$$\begin{aligned}
\partial^2_\psi\alpha\left(s_0,\psi\,;\,\hat{\psi},\dot{\psi}\right) &= \frac{2\ell I(\hat{\psi},\dot{\psi};s_0)}{I(\psi,\psi;\ell)} - 2\alpha(s_0,\psi)\frac{I(\hat{\psi},\dot{\psi};\ell)}{I(\psi,\psi;\ell)} \\
&\quad - \frac{2I(\psi,\hat{\psi};\ell)}{I(\psi,\psi;\ell)}\partial_\psi\alpha\left(s_0,\psi;\dot{\psi}\right) - \frac{2I(\psi,\dot{\psi};\ell)}{I(\psi,\psi;\ell)}\partial_\psi\alpha\left(s_0,\psi;\hat{\psi}\right).
\end{aligned} \tag{J.10.16}$$

K.11 Initial coefficient sensitivity

In our optimizations, the initial guesses are chosen to be low-frequency waves with small wave amplitudes. This is obtained by choosing the coefficients of the first Fourier modes from a uniform distribution within $[0, 0.01]$ (to restrict the initial wave amplitudes), and setting the coefficients of the higher modes to 0 (to discourage high-frequency waves).

Restricting our attention to low-frequency waves effectively sets a time scale in our problem. That is, it helps us to focus on the ciliary motion within *one* beating cycle which is given by the base mode. While cilia beating twice as fast (beating two cycles in the same time span) could double the swim speed, the efficiency would remain unchanged because of the simultaneous increase of the power loss.

Due to the high-dimensional nature of the problem (hundreds of degrees of freedom), many local optima exist. As shown in Figure K.11.1(a), a large initial range of the Fourier coefficient (e.g., $[0, 1]$) increases the risk of the optimizer getting stuck close to an unsuitable local optimum. For example, an initial waveform as shown in Figure K.11.1(c) can only be optimized to a waveform shown in Figure K.11.1(e), which has a swimming efficiency as low as 2%. On the other hand, the initial wave with small amplitudes (as shown in Figure K.11.1(b)) could almost always be optimized to the waveform with swimming efficiency $\epsilon \approx 35\%$.

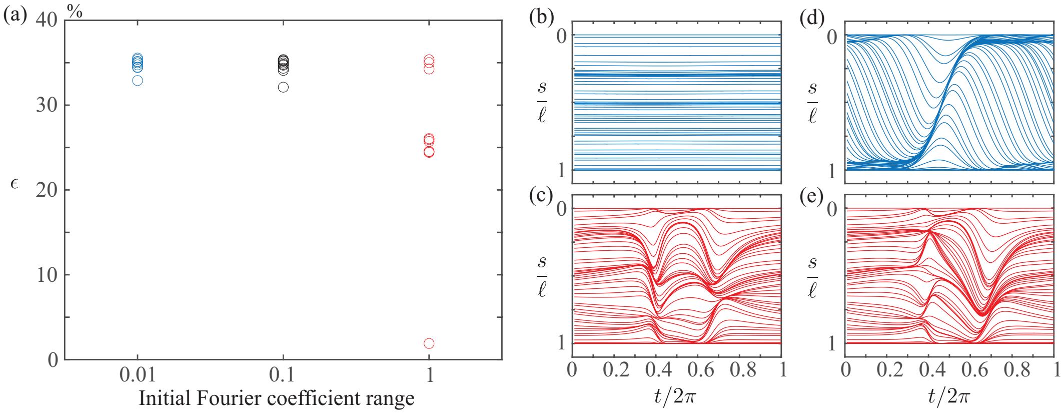

Figure K.11.1: Sensitivity to the initial Fourier coefficient. (a) Optimized efficiencies for the unconstrained spherical swimmer with the initial first Fourier mode chosen from $[0, 0.01]$, $[0, 0.1]$, $[0, 1]$ respectively. (b)&(d) The initial and final waveforms of the case where the range is $[0, 0.01]$. (c)&(e) The initial and final waveforms of the case where the range is $[0, 1]$.

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] Ludvig af Klinteberg, Travis Askham, and Mary Catherine Kropinski. A fast integral equation method for the two-dimensional Navier-Stokes equations. *Journal of Computational Physics*, 409:109353, 2020.

[2] Ludvig af Klinteberg and Alex H Barnett. Accurate quadrature of nearly singular line integrals in two and three dimensions by singularity swapping. *BIT Numerical Mathematics*, 61(1):83–118, 2021.

[3] Ludvig af Klinteberg, Davoud Saffar Shamshirgar, and Anna-Karin Tornberg. Fast Ewald summation for free-space Stokes potentials. *Research in the Mathematical Sciences*, 4(1):1, 2017.

[4] Ludvig af Klinteberg and Anna-Karin Tornberg. Adaptive quadrature by expansion for layer potential evaluation in two dimensions. *SIAM J. Sci. Comput.*, 40(3):A1225—A1249, 2018.

[5] Björn A Afzelius. A human syndrome caused by immotile cilia. *Science*, 193(4250):317–319, 1976.

[6] Yolanda Alvarez, Maria L Cederlund, David C Cottell, Brent R Bill, Stephen C Ekker, Jesus Torres-Vazquez, Brant M Weinstein, David R Hyde, Thomas S Vihtelic, and Breandan N Kennedy. Genetic determinants of hyaloid and retinal vasculature in zebrafish. *BMC developmental biology*, 7(1):114, 2007.

[7] John L Anderson. Colloid transport by interfacial forces. *Annual review of fluid mechanics*, 21(1):61–99, 1989.

[8] Hassan Aref. Chaotic advection of fluid particles. *Philosophical Transactions of the Royal Society of London. Series A: Physical and Engineering Sciences*, 333(1631):273–288, 1990.

[9] Douglas Arnold, Richard Falk, and Ragnar Winther. Finite element exterior calculus, homological techniques, and applications. *Acta numerica*, 15:1–155, 2006.

[10] Antoine Aubret and Jérémie Palacci. Diffusiophoretic design of self-spinning microgears from colloidal microswimmers. *Soft matter*, 14(47):9577–9588, 2018.

[11] Markus Bantle and Stefan Funken. Efficient and accurate implementation of *hp*-BEM for the Laplace operator in 2D. *Applied Numerical Mathematics*, 95:51 – 61, 2015. Fourth Chilean Workshop on Numerical Analysis of Partial Differential Equations (WONAPDE 2013).

[12] Alex Barnett. `memorygraph`: a MATLAB/octave unix tool to record true memory and CPU usage vs time. https://github.com/ahbarnett/memorygraph, 2018.

[13] Alex Barnett, Leslie Greengard, and Thomas Hagstrom. High-order discretization of a stable time-domain integral equation for 3D acoustic scattering. *Journal of Computational Physics*, 402:109047, 2020.

[14] Alex Barnett, Bowei Wu, and Shravan Veerapaneni. Spectrally accurate quadratures for evaluation of layer potentials close to the boundary for the 2D Stokes and Laplace equations. *SIAM Journal on Scientific Computing*, 37(4):B519–B542, 2015.

[15] Alex H Barnett. Evaluation of layer potentials close to the boundary for Laplace and Helmholtz problems on analytic planar domains. *SIAM Journal on Scientific Computing*, 36(2):A427–A451, 2014.

[16] Alex H Barnett, Gary R Marple, Shravan Veerapaneni, and Lin Zhao. A unified integral equation scheme for doubly periodic Laplace and Stokes boundary value problems in two dimensions. *Communications on Pure and Applied Mathematics*, 71(11):2334–2380, 2018.

[17] PV Bayly and SK Dutcher. Steady dynein forces induce flutter instability and propagating waves in mathematical models of flagella. *Journal of The Royal Society Interface*, 13(123):20160523, 2016.

[18] J Thomas Beale, Wenjun Ying, and Jason R Wilson. A simple method for computing singular or nearly singular integrals on closed surfaces. *Communications in Computational Physics*, 20(3):733–753, 2016.

[19] Saskia Bermbach, Karina Weinhold, Thomas Roeder, Frank Petersen, Christian Kugler, Torsten Goldmann, Jan Rupp, and Peter König. Mechanisms of cilia-driven transport in the airways in the absence of mucus. *American journal of respiratory cell and molecular biology*, 51(1):56–67, 2014.

[20] George Biros, Lexing Ying, and Denis Zorin. A fast solver for the Stokes equations with distributed forces in complex geometries. *Journal of Computational Physics*, 193(1):317–348, 2004.

[21] John R Blake. A spherical envelope approach to ciliary propulsion. *Journal of Fluid Mechanics*, 46(01):199–208, 1971.

[22] John R Blake. A model for the micro-structure in ciliated organisms. *Journal of Fluid Mechanics*, 55(01):1–23, 1972.

[23] John R Blake and Michael A Sleigh. Mechanics of ciliary locomotion. *Biological Reviews*, 49(1):85–125, 1974.

[24] Marc Bonnet, Ruowen Liu, and Shravan Veerapaneni. Shape optimization of Stokesian peristaltic pumps using boundary integral methods. *Advances in Computational Mathematics*, 46(2):1–24, 2020.

[25] Christopher Brennen and Howard Winet. Fluid mechanics of propulsion by cilia and flagella. *Annual Review of Fluid Mechanics*, 9(1):339–398, 1977.

[26] Franco Brezzi and Michel Fortin. *Mixed and hybrid finite element methods*. Springer-Verlag, 1991.

[27] Douglas R Brumley, Marco Polin, Timothy J Pedley, and Raymond E Goldstein. Metachronal waves in the flagellar beating of volvox and their hydrodynamic origin. *Journal of the Royal Society Interface*, 12(108):20141358, 2015.

[28] Oscar P Bruno and Jagabandhu Paul. Two-dimensional fourier continuation and applications. *arXiv preprint arXiv:2010.03901*, 2020.

[29] Spencer H Bryngelson and Jonathan B Freund. Global stability of flowing red blood cell trains. *Physical Review Fluids*, 3(7):073101, 2018.

[30] Camille Carvalho, Shilpa Khatri, and Arnold D Kim. Asymptotic analysis for close evaluation of layer potentials. *Journal of Computational Physics*, 355:327–341, 2018.

[31] Brato Chakrabarti and David Saintillan. Spontaneous oscillations, beating patterns, and hydrodynamics of active microfilaments. *Physical Review Fluids*, 4(4):043102, 2019.

[32] Sylvain Chateau, Umberto d'Ortona, Sébastien Poncet, and Julien Favier. Transport and mixing induced by beating cilia in human airways. *Frontiers in physiology*, 9:161, 2018.

[33] Robin Chatelin and Philippe Poncet. A parametric study of mucociliary transport by numerical simulations of 3d non-homogeneous mucus. *Journal of Biomechanics*, 2016.

[34] Chia-Yuan Chen, Chia-Yun Chen, Cheng-Yi Lin, and Ya-Ting Hu. Magnetically actuated artificial cilia for optimum mixing performance in microfluidics. *Lab on a Chip*, 13(14):2834–2839, 2013.

[35] Udit Choudhury, Arthur V Straube, Peer Fischer, John G Gibbs, and Felix Höfling. Active colloidal propulsion over a crystalline surface. *New Journal of Physics*, 19(12):125010, 2017.

[36] Ricardo Cortez. The method of regularized Stokeslets. *SIAM Journal on Scientific Computing*, 23(4):1204–1225, 2001.

[37] Gabriele De Canio, Eric Lauga, and Raymond E Goldstein. Spontaneous oscillations of elastic filaments induced by molecular motors. *Journal of The Royal Society Interface*, 14(136):20170491, 2017.

[38] Robert H Dillon, Lisa J Fauci, Charlotte Omoto, and Xingzhou Yang. Fluid dynamic models of flagellar and ciliary beating. *Annals of the New York Academy of Sciences*, 1101(1):494–505, 2007.

[39] Yang Ding, Janna C Nawroth, Margaret J McFall-Ngai, and Eva Kanso. Mixing and transport by ciliary carpets: a numerical study. *Journal of Fluid Mechanics*, 743:124–140, 2014.

[40] Jens Elgeti and Gerhard Gompper. Emergence of metachronal waves in cilia arrays. *Proceedings of the National Academy of Sciences*, 110(12):4470–4475, 2013.

[41] Jens Elgeti, Roland G Winkler, and Gerhard Gompper. Physics of microswimmers - single particle motion and collective behavior: a review. *Reports on progress in physics*, 78(5):056601, 2015.

[42] Christophe Eloy and Eric Lauga. Kinematics of the most efficient cilium. *Physical Review Letters*, 109(3):038101, 2012.

[43] Frank Ethridge and Leslie Greengard. A new fast-multipole accelerated Poisson solver in two dimensions. *SIAM Journal on Scientific Computing*, 23(3):741–760, 2001.

[44] Francis Fahrni, Menno WJ Prins, and Leo J van IJzendoorn. Micro-fluidic actuation using magnetic artificial cilia. *Lab on a Chip*, 9(23):3413–3421, 2009.

[45] Regina Faubel, Christian Westendorf, Eberhard Bodenschatz, and Gregor Eichele. Cilia-based flow network in the brain ventricles. *Science*, 353(6295):176–178, 2016.

[46] Lisa J Fauci and Robert Dillon. Biofluidmechanics of reproduction. *Annu. Rev. Fluid Mech.*, 38:371–394, 2006.

[47] Manfred Fliegauf, Thomas Benzing, and Heymut Omran. When cilia go bad: cilia defects and ciliopathies. *Nature reviews Molecular cell biology*, 8(11):880–893, 2007.

[48] Fredrik Fryklund, Erik Lehto, and Anna-Karin Tornberg. Partition of unity extension of functions on complex domains. *Journal of Computational Physics*, 375:57–79, 2018.

[49] Glenn R Fulford and John R Blake. Muco-ciliary transport in the lung. *Journal of Theoretical Biology*, 121(4):381–402, 1986.

[50] Eamonn A Gaffney, Hermes Gadêlha, David J Smith, John R Blake, and Jackson C Kirkman-Brown. Mammalian sperm motility: observation and theory. *Annual Review of Fluid Mechanics*, 43:501–528, 2011.

[51] Jean Gallier. Notes on spherical harmonics and linear representations of Lie groups. *preprint*, 2009.

[52] IR Gibbons. Cilia and flagella of eukaryotes. *J Cell Biol*, 91(3):107s–124s, 1981.

[53] Zydrunas Gimbutas and Leslie Greengard. FMMLIB2D, Fortran libraries for fast multipole methods in two dimensions, 2012. `http://www.cims.nyu.edu/cmcl/fmm2dlib/fmm2dlib.html`.

[54] Zydrunas Gimbutas and Leslie Greengard. Computational software: Simple FMM libraries for electrostatics, slow viscous flow, and frequency-domain wave propagation. *Communications in Computational Physics*, 18(2):516–528, 2015.

[55] Zydrunas Gimbutas and Shravan Veerapaneni. A fast algorithm for spherical grid rotations and its application to singular quadrature. *SIAM Journal on Scientific Computing*, 35(6):A2738–A2751, 2013.

[56] Ramin Golestanian, Tanniemola B Liverpool, and Armand Ajdari. Designing phoretic micro-and nano-swimmers. *New Journal of Physics*, 9(5):126, 2007.

[57] Abinand Gopal and Lloyd N. Trefethen. Solving Laplace problems with corner singularities via rational functions. *SIAM Journal on Numerical Analysis*, 57(5):2074–2094, 2019.

[58] James Gray. *Ciliary Movement*. Cambridge University Press, London, 1928.

[59] A Greenbaum, L Greengard, and A Mayo. On the numerical solution of the biharmonic equation in the plane. *Physica D*, 60(1–4):216–225, 1992.

[60] Leslie Greengard and Shidong Jiang. A new mixed potential representation for unsteady, incompressible flow. *SIAM Review*, 61(4):733–755, 2019.

[61] Leslie Greengard and Mary Catherine Kropinski. An integral equation approach to the incompressible Navier–Stokes equations in two dimensions. *SIAM Journal on Scientific Computing*, 20(1):318–336, 1998.

[62] Leslie Greengard, Mary Catherine Kropinski, and Anita Mayo. Integral equation methods for Stokes flow and isotropic elasticity in the plane. *Journal of Computational Physics*, 125(2):403–414, 1996.

[63] Leslie Greengard, Michael O'Neil, Manas Rachh, and Felipe Vico. Fast multipole methods for the evaluation of layer potentials with locally-corrected quadratures. *Journal of Computational Physics: X*, page 100092, 2021.

[64] Leslie Greengard and Vladimir Rokhlin. A fast algorithm for particle simulations. *Journal of Computational Physics*, 73(2):325–348, 1987.

[65] Shay Gueron and Konstantin Levit-Gurevich. Energetic considerations of ciliary beating and the advantage of metachronal coordination. *Proceedings of the National Academy of Sciences*, 96(22):12240–12245, 1999.

[66] Shay Gueron, Konstantin Levit-Gurevich, Nadav Liron, and Jacob J Blum. Cilia internal mechanism and metachronal coordination as the result of hydrodynamical coupling. *Proceedings of the National Academy of Sciences*, 94(12):6001–6006, 1997.

[67] Shay Gueron and Nadav Liron. Ciliary motion modeling, and dynamic multicilia interactions. *Biophysical journal*, 63(4):1045, 1992.

[68] Shay Gueron and Nadav Liron. Simulations of three-dimensional ciliary beats and cilia interactions. *Biophysical journal*, 65(1):499, 1993.

[69] Boris Guirao and Jean-Francois Joanny. Spontaneous creation of macroscopic flow and metachronal waves in an array of cilia. *Biophysical Journal*, 92(6):1900–1917, March 2007.

[70] Hanliang Guo, Lisa Fauci, Michael J Shelley, and Eva Kanso. Bistability in the synchronization of actuated microfilaments. *Journal of Fluid Mechanics*, 836:304–323, 2018.

[71] Hanliang Guo, Janna C Nawroth, Yang Ding, and Eva Kanso. Cilia beating patterns are not hydrodynamically optimal. *Physics of Fluids*, 26(9):091901, 2014.

[72] Hanliang Guo, Hai Zhu, Ruowen Liu, Marc Bonnet, and Shravan Veerapaneni. Optimal ciliary locomotion of axisymmetric microswimmers. *arXiv preprint arXiv:2103.15642*, 2021.

[73] Hanliang Guo, Hai Zhu, Ruowen Liu, Marc Bonnet, and Shravan Veerapaneni. Optimal slip velocities of micro-swimmers with arbitrary axisymmetric shapes. *Journal of Fluid Mechanics*, 910, 2021.

[74] Sijia Hao, Alex Barnett, Per-Gunnar Martinsson, and P Young. High-order accurate methods for Nyström discretization of integral equations on smooth curves in the plane. *Advances in Computational Mathematics*, 40(1):245–272, 2014.

[75] John Happel and Howard Brenner. *Low Reynolds number hydrodynamics with special applications to particulate media*. Noordhoff, 1973.

[76] F.-K. Hebeker. Efficient boundary element methods for three-dimensional exterior viscous flows. *Numer. Methods Partial Differential Equations*, 2:273–297, 1986.

[77] J Helsing. Integral equation methods for elliptic problems with boundary conditions of mixed type. *J. Comput. Phys.*, 228:8892–8907, 2009.

[78] J Helsing and A Holst. Variants of an explicit kernel-split panel-based Nyström discretization scheme for Helmholtz boundary value problems. *Adv. Comput. Math.*, 41(3):691–708, 2015.

[79] J Helsing and S Jiang. On integral equation methods for the first Dirichlet problem of the biharmonic and modified biharmonic equations in nonsmooth domains. *SIAM J. Sci. Comput.*, 40(4):A2609–A2630, 2018.

[80] J. Helsing and R. Ojala. On the evaluation of layer potentials close to their sources. *Journal of Computational Physics*, 227:2899–2921, 2008.

[81] Johan Helsing. A higher-order singularity subtraction technique for the discretization of singular integral operators on curved surfaces. *arXiv preprint arXiv:1301.7276*, 2013.

[82] Johan Helsing. Solving integral equations on piecewise smooth boundaries using the RCIP method: a tutorial. *Abstract and Applied Analysis*, 2013:Article ID 938167, 2013.

[83] Johan Helsing and Shidong Jiang. On integral equation methods for the first Dirichlet problem of the biharmonic and modified biharmonic equations in nonsmooth domains. *SIAM Journal on Scientific Computing*, 40(4):A2609–A2630, 2018.

[84] Johan Helsing and Anders Karlsson. An explicit kernel-split panel-based nyström scheme for integral equations on axially symmetric surfaces. *Journal of Computational Physics*, 272:686–703, 2014.

[85] N Heuer and E P Stephan. The *hp*-version of the boundary element method on polygons. *J. Integral Equ. Appl.*, 8(2):173–212, 1996.

[86] Rolla B Hill. Bloom and fawcett: A textbook of histology. *JAMA*, 256(10):1366–1367, 1986.

[87] Jonathan R Howse, Richard AL Jones, Anthony J Ryan, Tim Gough, Reza Vafabakhsh, and Ramin Golestanian. Self-motile colloidal particles: from directed propulsion to random walk. *Physical review letters*, 99(4):048102, 2007.

[88] G. C. Hsiao and W. L. Wendland. *Boundary integral equations*, volume 164 of *Applied Mathematical Sciences*. Springer, 2008.

[89] S Elizabeth Hulme, Sergey S Shevkoplyas, Javier Apfeld, Walter Fontana, and George M Whitesides. A microfabricated array of clamps for immobilizing and imaging c. elegans. *Lab on a Chip*, 7(11):1515–1523, 2007.

[90] Takuji Ishikawa and Timothy J Pedley. Coherent structures in monolayers of swimming particles. *Physical review letters*, 100(8):088103, 2008.

[91] Takuji Ishikawa and TJ Pedley. The rheology of a semi-dilute suspension of swimming model micro-organisms. *Journal of Fluid Mechanics*, 588:399–435, 2007.

[92] Takuji Ishikawa, MP Simmonds, and Timothy J Pedley. Hydrodynamic interaction of two swimming model micro-organisms. *Journal of Fluid Mechanics*, 568:119–160, 2006.

[93] Kenta Ishimoto and Eamonn A Gaffney. Squirmer dynamics near a boundary. *Physical Review E*, 88(6):062702, 2013.

[94] Hiroaki Ito, Toshihiro Omori, and Takuji Ishikawa. Swimming mediated by ciliary beating: comparison with a squirmer model. *Journal of Fluid Mechanics*, 874:774–796, 2019.

[95] P G Jayathilake, D V Le, Zhijun Tan, H P Lee, and B C Khoo. A numerical study of muco-ciliary transport under the condition of diseased cilia. *Computer Methods in Biomechanics and Biomedical Engineering*, 18(9):944–951, 2015.

[96] Youngren G. K. and A. Acrivos. Stokes flow past a particle of arbitrary shape: a numerical method of solution. *Journal of Fluid Mechanics*, 69:377–403, May 1975.

[97] Youngren G. K. and A. Acrivos. On the shape of a gas bubble in a viscous extensional flow. *Journal of Fluid Mechanics*, 76:433–442, August 1976.

[98] Gökberk Kabacaoğlu and George Biros. Optimal design of deterministic lateral displacement device for viscosity-contrast-based cell sorting. *Physical Review Fluids*, 3(12):124201, 2018.

[99] Stuart R Keller and Theodore Y Wu. A porous prolate-spheroidal model for ciliated micro-organisms. *Journal of Fluid Mechanics*, 80(2):259–278, 1977.

[100] Syed N Khaderi, CB Craus, Jeanette Hussong, N Schorr, J Belardi, J Westerweel, O Prucker, J Rühe, JMJ Den Toonder, and PR Onck. Magnetically-actuated artificial cilia for microfluidic propulsion. *Lab on a Chip*, 11(12):2002–2010, 2011.

[101] Sangtae Kim and Seppo J Karrila. *Microhydrodynamics: principles and selected applications*. Courier Corporation, 2013.

[102] Eisuke Kita and Norio Kamiya. Error estimation and adaptive mesh refinement in boundary element method, an overview. *Engineering Analysis with Boundary Elements*, 25(7):479–495, 2001.

[103] Andreas Klöckner, Alexander Barnett, Leslie Greengard, and Michael O'Neil. Quadrature by expansion: A new method for the evaluation of layer potentials. *Journal of Computational Physics*, 252:332–349, 2013.

[104] E W Knight-Jones. Relations between metachronism and the direction of ciliary beat in Metazoa. *Quarterly Journal of Microscopical Science*, 95:503–521, 1954.

[105] Ryan Kohl, Eduardo Corona, Vani Cheruvu, and Shravan Veerapaneni. Fast and accurate solvers for simulating Janus particle suspensions in Stokes flow. *arXiv preprint arXiv:2104.14068*, 2021.

[106] Rainer Kress. *Linear Integral Equations*, volume 82 of *Applied Mathematical Sciences*. Springer, 2nd edition, 1999.

[107] Rainer Kress, V Maz'ya, and V Kozlov. *Linear integral equations*, volume 82. Springer, 1989.

[108] M. C. A. Kropinski. An efficient numerical method for studying interfacial motion in two-dimensional creeping flows. *Journal of Computational Physics*, 171(2):479–508, 2001.

[109] Amit Kumar and Michael D Graham. Accelerated boundary integral method for multiphase flow in non-periodic geometries. *Journal of Computational Physics*, 231(20):6682–6713, 2012.

[110] K Kyoya, D Matsunaga, Y Imai, T Omori, and T Ishikawa. Shape matters: Near-field fluid mechanics dominate the collective motions of ellipsoidal squirmers. *Physical Review E*, 92(6):063027, 2015.

[111] Olga Aleksandrovna Ladyzhenskaya. *The mathematical theory of viscous incompressible flow*, volume 2. Gordon and Breach New York, 1969.

[112] Harper Langston, Leslie Greengard, and Denis Zorin. A free-space adaptive FMM-based PDE solver in three dimensions. *Communications in Applied Mathematics and Computational Science*, 6(1):79–122, 2011.

[113] Eric Lauga and Thomas R Powers. The hydrodynamics of swimming microorganisms. *Reports on Progress in Physics*, 72(9):096601, 2009.

[114] Kate Lawrenson, Maria Notaridou, Nathan Lee, Elizabeth Benjamin, Ian J Jacobs, Christopher Jones, and Simon A Gayther. In vitro three-dimensional modeling of fallopian tube secretory epithelial cells. *BMC cell biology*, 14(1):43, 2013.

[115] W L Lee, P G Jayathilake, Zhijun Tan, D V Le, H P Lee, and B C Khoo. Muco-ciliary transport: effect of mucus viscosity, cilia beat frequency and cilia density. *Computers & Fluids*, 49(1):214–221, 2011.

[116] Alexander M Leshansky, Oded Kenneth, Omri Gat, and Joseph E Avron. A frictionless microswimmer. *New Journal of Physics*, 9(5):145, 2007.

[117] Zhe Li, Julien Favier, Umberto D'Ortona, and Sébastien Poncet. An immersed boundary-lattice boltzmann method for single-and multi-component fluid flows. *Journal of Computational Physics*, 304:424–440, 2016.

[118] James Lighthill. On the squirming motion of nearly spherical deformable bodies through liquids at very small reynolds numbers. *Communications on Pure and Applied Mathematics*, 5(2):109–118, 1952.

[119] Dag Lindbo and Anna-Karin Tornberg. Spectrally accurate fast summation for periodic Stokes potentials. *Journal of Computational Physics*, 229(23):8994–9010, 2010.

[120] Y Liu. *Fast Multipole Boundary Element Method: Theory and Applications in Engineering.* Cambridge University Press, 2009.

[121] Sarah Lukens, Xingzhou Yang, and Lisa Fauci. Using lagrangian coherent structures to analyze fluid mixing by cilia. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 20(1):017511, 2010.

[122] Denis Lynn. *The ciliated protozoa: characterization, classification, and guide to the literature.* Springer Science & Business Media, 2008.

[123] Hans Machemer. Mechanoresponses in protozoa. In *Sensory Perception and Transduction in Aneural Organisms*, pages 179–209. Springer, 1985.

[124] Vanesa Magar, Tomonobu Goto, and Timothy J Pedley. Nutrient uptake by a self-propelled steady squirmer. *The Quarterly Journal of Mechanics and Applied Mathematics*, 56(1):65–91, 2003.

[125] Vanesa Magar and Timothy J Pedley. Average nutrient uptake by a self-propelled unsteady squirmer. *Journal of fluid mechanics*, 539:93–112, 2005.

[126] Dhairya Malhotra and George Biros. PVFMM: A parallel kernel independent FMM for particle and volume potentials. *Communications in Computational Physics*, 18(3):808–830, 2015.

[127] Gary Marple, Alexander H. Barnett, Adrianna Gillman, and Shravan K. Veerapaneni. A fast algorithm for simulating multiphase flows through periodic geometries of arbitrary shape. *SIAM Journal on Scientific Computing*, 38(5):B740–B772, 2016.

[128] Sébastien Michelin and Eric Lauga. Efficiency optimization and symmetry-breaking in a model of ciliary locomotion. *Physics of Fluids*, 22(11):111901, 2010.

[129] Sébastien Michelin and Eric Lauga. Optimal feeding is optimal swimming for all péclet numbers. *Physics of Fluids*, 23(10):101901, 2011.

[130] Sébastien Michelin and Eric Lauga. Unsteady feeding and optimal strokes of model ciliates. *Journal of Fluid Mechanics*, 715:1–31, 2013.

[131] Sorin M Mitran. Metachronal wave formation in a model of pulmonary cilia. *Computers & structures*, 85(11):763–774, 2007.

[132] Adam R Morgan, Alan B Dawson, Holly S Mckenzie, Thomas S Skelhon, Richard Beanland, Henry PW Franks, and Stefan AF Bon. Chemotaxis of catalytic silica–manganese oxide 'matchstick' particles. *Materials Horizons*, 1(1):65–68, 2014.

[133] Matthew J Morse, Abtin Rahimian, and Denis Zorin. A robust solver for elliptic PDEs in 3D complex geometries. *Journal of Computational Physics*, page 110511, 2021.

[134] Babak Nasouri, Andrej Vilfan, and Ramin Golestanian. Minimum dissipation theorem for microswimmers. *Physical Review Letters*, 126(3):034503, 2021.

[135] Janna C Nawroth, Hanliang Guo, Eric Koch, Elizabeth AC Heath-Heckman, John C Hermanson, Edward G Ruby, John O Dabiri, Eva Kanso, and Margaret McFall-Ngai. Motile cilia create fluid-mechanical microhabitats for the active recruitment of the host microbiome. *Proceedings of the National Academy of Sciences*, 114(36):9510–9516, 2017.

[136] Ehssan Nazockdast, Abtin Rahimian, Denis Zorin, and Michael Shelley. A fast platform for simulating semi-flexible fiber suspensions applied to cell mechanics. *Journal of Computational Physics*, 329:173–209, 2017.

[137] Jorge Nocedal and Stephen Wright. *Numerical optimization*. Springer Science & Business Media, 2006.

[138] R. Ojala. A robust and accurate solver of Laplace's equation with general boundary conditions on general domains in the plane. *J. Comput. Math.*, 30(4):433–448, 2012.

[139] Rikard Ojala and Anna-Karin Tornberg. An accurate integral equation method for simulating multi-phase Stokes flow. *Journal of Computational Physics*, 298:145–160, 2015.

[140] Emilie W Olstad, Christa Ringers, Jan N Hansen, Adinda Wens, Cecilia Brandt, Dagmar Wachten, Emre Yaksi, and Nathalie Jurisch-Yaksi. Ciliary beating compartmentalizes cerebrospinal fluid flow in the brain and regulates ventricular development. *Current Biology*, 29(2):229–241, 2019.

[141] Toshihiro Omori, Hiroaki Ito, and Takuji Ishikawa. Swimming microorganisms acquire optimal efficiency with multiple cilia. *Proceedings of the National Academy of Sciences*, 117(48):30201–30207, 2020.

[142] Natan Osterman and Andrej Vilfan. Finding the ciliary beating pattern with optimal efficiency. *Proceedings of the National Academy of Sciences*, 108(38):15727–15732, 2011.

[143] Julio M Ottino and JM Ottino. *The kinematics of mixing: stretching, chaos, and transport*, volume 3. Cambridge university press, 1989.

[144] Jérémie Palacci, Stefano Sacanna, Anaïs Abramian, Jérémie Barral, Kasey Hanson, Alexander Y Grosberg, David J Pine, and Paul M Chaikin. Artificial rheotaxis. *Science advances*, 1(4):e1400214, 2015.

[145] V Y Pan. How bad are Vandermonde matrices? *SIAM J. Matrix Anal. Appl.*, 37(2):676–694, 2016.

[146] Walter F Paxton, Kevin C Kistler, Christine C Olmeda, Ayusman Sen, Sarah K St. Angelo, Yanyan Cao, Thomas E Mallouk, Paul E Lammert, and Vincent H Crespi. Catalytic nanomotors: autonomous movement of striped nanorods. *Journal of the American Chemical Society*, 126(41):13424–13431, 2004.

[147] Timothy J Pedley. Spherical squirmers: models for swimming micro-organisms. *IMA Journal of Applied Mathematics*, 81(3):488–521, 2016.

[148] Rachel E Pepper, Marcus Roper, Sangjin Ryu, Nobuyoshi Matsumoto, Moeto Nagai, and Howard A Stone. A new angle on microscopic suspension feeders near boundaries. *Biophysical journal*, 105(8):1796–1804, 2013.

[149] Carlos Pérez-Arancibia, Luiz M Faria, and Catalin Turc. Harmonic density interpolation methods for high-order evaluation of laplace layer potentials in 2D and 3D. *Journal of Computational Physics*, 376:411–434, 2019.

[150] Olivier Pironneau. On optimum profiles in Stokes flow. *Journal of Fluid Mechanics*, 59(1):117–128, 1973.

[151] Henry Power and Guillermo Miranda. Second kind integral equation formulation of Stokes' flows past a particle of arbitrary shape. *SIAM Journal on Applied Mathematics*, 47(4):689–698, 1987.

[152] Constantine Pozrikidis et al. *Boundary integral and singularity methods for linearized viscous flow*. Cambridge university press, 1992.

[153] Edward M Purcell. Life at low reynolds number. *Am. J. Phys*, 45(1):3–11, 1977.

[154] M Rachh. `bhfmm2d`: parallel Fortran code for the biharmonic FMM in 2D, 2012.

[155] Manas Rachh, Andreas Klöckner, and Michael O'Neil. Fast algorithms for Quadrature by Expansion I: Globally valid expansions. *Journal of Computational Physics*, 345:706–731, 2017.

[156] Manas Rachh and Kirill Serkh. On the solution of the Stokes equation on regions with corners. *Communications on Pure and Applied Mathematics*, 73(11):2295–2369, 2020.

[157] Abtin Rahimian, Alex Barnett, and Denis Zorin. Ubiquitous evaluation of layer potentials using quadrature by kernel-independent expansion. *BIT Numerical Mathematics*, 58(2):423–456, 2018.

[158] Abtin Rahimian, Ilya Lashuk, Shravan Veerapaneni, Aparna Chandramowlishwaran, Dhairya Malhotra, Logan Moon, Rahul Sampath, Aashay Shringarpure, Jeffrey Vetter, Richard Vuduc, Denis Zorin, and George Biros. Petascale direct numerical simulation of blood flow on 200k cores and heterogeneous architectures. In *Proceedings of the 2010 ACM/IEEE International Conference for High Performance Computing, Networking, Storage and Analysis*, SC '10, pages 1–11, 2010.

[159] Abtin Rahimian, Shravan K Veerapaneni, Denis Zorin, and George Biros. Boundary integral method for the flow of vesicles with viscosity contrast in three dimensions. *Journal of Computational Physics*, 298:766–786, 2015.

[160] Abtin Rahimian, Shravan Kumar Veerapaneni, and George Biros. Dynamic simulation of locally inextensible vesicles suspended in an arbitrary two-dimensional domain, a boundary integral method. *Journal of Computational Physics*, 229(18):6466–6484, 2010.

[161] Hans Ulrik Riisgård and Poul S Larsen. Particle capture mechanisms in suspension-feeding invertebrates. *Marine Ecology Progress Series*, 418:255–293, 2010.

[162] Minghao W Rostami and Sarah D Olson. Fast algorithms for large dense matrices with applications to biofluids. *Journal of Computational Physics*, 2019.

[163] Benedikt Sabass and Udo Seifert. Dynamics and efficiency of a self-propelled, diffusiophoretic swimmer. *The Journal of chemical physics*, 136(6):064508, 2012.

[164] Aref Saberi, Shuaizhong Zhang, Carola van den Bersselaar, Harkamaljot Kandail, Jaap MJ den Toonder, and Nicholas A Kurniawan. A stirring system using suspended magnetically-actuated pillars for controlled cell clustering. *Soft matter*, 15(6):1435–1443, 2019.

[165] David Saintillan, Eric Darve, and Eric SG Shaqfeh. A smooth particle-mesh Ewald algorithm for Stokes suspension simulations: The sedimentation of fibers. *Physics of Fluids*, 17(3):033301, 2005.

[166] David Saintillan and Michael J Shelley. Theory of active suspensions. In *Complex Fluids in biological systems*, pages 319–355. Springer, 2015.

[167] Ashok S Sangani and Guobiao Mo. Inclusion of lubrication forces in dynamic simulations. *Physics of fluids*, 6(5):1653–1662, 1994.

[168] Ashok S Sangani and Guobiao Mo. An $\mathcal{O}(\mathcal{N})$ algorithm for Stokes and Laplace interactions of particles. *Physics of Fluids*, 8(8):1990–2010, 1996.

[169] Peter Satir and Søren Tvorup Christensen. Overview of structure and function of mammalian cilia. *Annual Review of Physiology*, 69(1):377–400, 2007.

[170] A R Shields, B L Fiser, B A Evans, M R Falvo, S Washburn, and R Superfine. Biomimetic cilia arrays generate simultaneous pumping and mixing regimes. *Proceedings of the National Academy of Sciences*, 107(36):15670–15675, September 2010.

[171] Michael Siegel and Anna-Karin Tornberg. A local target specific quadrature by expansion method for evaluation of layer potentials in 3D. *Journal of Computational Physics*, 364:365–392, 2018.

[172] Josef Sifuentes, Zydrunas Gimbutas, and Leslie Greengard. Randomized methods for rank-deficient linear systems. *Electronic Transactions on Numerical Analysis*, 44(Electronic Transactions on Numerical Analysis), 2015.

[173] Juliane Simmchen, Alejandro Baeza, Albert Miguel-Lopez, Morgan M Stanton, Maria Vallet-Regi, Daniel Ruiz-Molina, and Samuel Sánchez. Dynamics of novel photoactive agcl microstars and their environmental applications. *ChemNanoMat*, 3(1):65–71, 2017.

[174] Michael A Sleigh. *The Biology of Cilia and Flagella.* Macmillan Co., New York, 1962.

[175] David J Smith. A boundary element regularized Stokeslet method applied to cilia-and flagella-driven flow. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 465(2112):3605–3626, 2009.

[176] David J Smith, John R Blake, and Eamonn A Gaffney. Fluid mechanics of nodal flow due to embryonic primary cilia. *Journal of The Royal Society Interface*, 5(22):567–573, 2008.

[177] David J Smith, Eamonn A Gaffney, and John R Blake. Modelling mucociliary clearance. *Respiratory physiology & neurobiology*, 163(1):178–188, 2008.

[178] Chiara Sorgentone and Anna-Karin Tornberg. A highly accurate boundary integral equation method for surfactant-laden drops in 3D. *Journal of Computational Physics*, 360:167–191, 2018.

[179] Nathalie Spassky and Alice Meunier. The development and functions of multiciliated epithelia. *Nature reviews Molecular cell biology*, 18(7):423, 2017.

[180] Michael Spivak. *Calculus on manifolds: a modern approach to classical theorems of advanced calculus.* CRC Press, 2018.

[181] David B. Stein and Michael J. Shelley. Coarse graining the dynamics of immersed and driven fiber assemblies. *Phys. Rev. Fluids*, 4:073302, Jul 2019.

[182] ZB Stone and HA Stone. Imaging and quantifying mixing in a model droplet micromixer. *Physics of Fluids*, 17(6):063103, 2005.

[183] W Supatto, S E Fraser, and J Vermot. An all-optical approach for probing microscopic flows in living embryos. *Biophysical journal*, 95:L29–L31, 2008.

[184] Andrea Tagliasacchi. `kdtree` code. *https://www.mathworks.com/matlabcentral/fileexchange/21512-ataiya-kdtree*, 2017.

[185] Kenshi Takayama, Daniele Panozzo, Alexander Sorkine-Hornung, and Olga Sorkine-Hornung. Sketch-based generation and editing of quad meshes. *ACM Transactions on Graphics (TOG)*, 32(4):1–8, 2013.

[186] Geoffrey Taylor. Analysis of the swimming of microscopic organisms. *Proceedings of the Royal Society of London. Series A*, 209(1099):447–461, 1951.

[187] Mario Theers, Elmar Westphal, Gerhard Gompper, and Roland G Winkler. Modeling a spheroidal microswimmer and cooperative swimming in a narrow slit. *Soft Matter*, 12(35):7372–7385, 2016.

[188] Ann E. Tilley, Matthew S. Walters, Renat Shaykhiev, and Ronald G. Crystal. Cilia dysfunction in lung disease. *Annual Review of Physiology*, 77(1):379–406, 2015.

[189] Svetlana Tlupova and J Thomas Beale. Regularized single and double layer integrals in 3D Stokes flow. *Journal of Computational Physics*, 386:568–584, 2019.

[190] Anna-Karin Tornberg and Leslie Greengard. A fast multipole method for the three-dimensional Stokes equations. *Journal of Computational Physics*, 227(3):1613–1619, 2008.

[191] Lloyd N Trefethen. *Approximation theory and approximation practice*, volume 128. SIAM, 2013.

[192] Greg Turk and Marc Levoy. Zippered polygon meshes from range images. In *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, pages 311–318, 1994.

[193] William E Uspal, Mikhail N Popescu, Mykola Tasinkevych, and Siegfried Dietrich. Shape-dependent guidance of active janus particles by chemically patterned surfaces. *New Journal of Physics*, 20(1):015013, 2018.

[194] Leonardo F Valadares, Yu-Guo Tao, Nicole S Zacharia, Vladimir Kitaev, Fernando Galembeck, Raymond Kapral, and Geoffrey A Ozin. Catalytic nanomotors: Self-propelled sphere dimers. *Small*, 6(4):565–572, 2010.

[195] H van den Ende, A Musgrave, and Klis F. M. The role of flagella in the sexual reproduction of chylamydomonas gametes. In *Ciliary and flagellar membranes*, pages 129–148. Springer, 1990.

[196] Shravan Veerapaneni. Integral equation methods for vesicle electrohydrodynamics in three dimensions. *Journal of Computational Physics*, 326:278–289, 2016.

[197] Shravan K Veerapaneni, Denis Gueyffier, George Biros, and Denis Zorin. A numerical method for simulating the dynamics of 3d axisymmetric vesicles suspended in viscous flows. *Journal of Computational Physics*, 228(19):7233–7249, 2009.

[198] Shravan K Veerapaneni, Denis Gueyffier, Denis Zorin, and George Biros. A boundary integral method for simulating the dynamics of inextensible vesicles suspended in a viscous fluid in 2d. *Journal of Computational Physics*, 228(7):2334–2353, 2009.

[199] Shravan K. Veerapaneni, Abtin Rahimian, George Biros, and Denis Zorin. A fast algorithm for simulating vesicle flows in three dimensions. *J. Comput. Phys.*, 230(14):5610–5634, 2011.

[200] Marcos F Velho Rodrigues, Maciej Lisicki, and Eric Lauga. The bank of swimming organisms at the micron scale (boso-micro). *Plos one*, 16(6):e0252291, 2021.

[201] P Verdugo, WI Lee, SA Halbert, RJ Blandau, and PY Tam. A stochastic model for oviductal egg transport. *Biophysical journal*, 29(2):257, 1980.

[202] Andrej Vilfan. Optimal shapes of surface slip driven self-propelled microswimmers. *Physical review letters*, 109(12):128105, 2012.

[203] Matt Wala and Andreas Klöckner. A fast algorithm for quadrature by expansion in three dimensions. *Journal of Computational Physics*, 388:655–689, 2019.

[204] Matt Wala and Andreas Klöckner. Optimization of fast algorithms for global Quadrature by Expansion using target-specific expansions. *Journal of Computational Physics*, 403:108976, 2020.

[205] Haitao Wang, Ting Lei, Jin Li, Jingfang Huang, and Zhenhan Yao. A parallel fast multipole accelerated integral equation scheme for 3D Stokes equations. *International journal for numerical methods in engineering*, 70(7):812–839, 2007.

[206] Jun Wang, Leslie Greengard, Shidong Jiang, and Shravan Veerapaneni. Fast integral equation methods for linear and semilinear heat equations in moving domains. *arXiv preprint arXiv:1910.00755*, 2019.

[207] Lei Wang, Svetlana Tlupova, and Robert Krasny. A treecode algorithm for 3D Stokeslets and stresslets. *Advances in Applied Mathematics and Mechanics*, 11:737–756, 2019.

[208] Xin Wang, Joe Kanapka, Wenjing Ye, Narayan R Aluru, and Jacob White. Algorithms in FastStokes and its application to micromachined device simulation. *IEEE Transactions on computer-aided design of integrated circuits and systems*, 25(2):248–257, 2006.

[209] George B Witman. Introduction to cilia and flagella. In *Ciliary and flagellar membranes*, pages 1–30. Springer, 1990.

[210] Bowei Wu and Shravan Veerapaneni. Electrohydrodynamics of deflated vesicles: budding, rheology and pairwise interactions. *Journal of Fluid Mechanics*, 867:334–347, 2019.

[211] Bowei Wu, Hai Zhu, Alex Barnett, and Shravan Veerapaneni. Solution of Stokes flow in complex nonsmooth 2d geometries via a linear-scaling high-order adaptive integral equation scheme. *Journal of Computational Physics*, 410:109361, 2020.

[212] Wen Yan, Eduardo Corona, Dhairya Malhotra, Shravan Veerapaneni, and Michael Shelley. A scalable computational platform for particulate Stokes suspensions. *Journal of Computational Physics*, 416:109524, 2020.

[213] L. Ying, G. Biros, and D. Zorin. A kernel-independent adaptive fast multipole method in two and three dimensions. *Journal of Computational Physics*, 196(2):591–626, 2004.

[214] Lexing Ying, George Biros, and Denis Zorin. A high-order 3D boundary integral equation solver for elliptic PDEs in smooth domains. *Journal of Computational Physics*, 219(1):247–275, 2006.

[215] Yabin Zhang and Adrianna Gillman. A fast direct solver for boundary value problems on locally perturbed geometries. *Journal of Computational Physics*, 356:356–371, 2018.

[216] Hai Zhu and Shravan Veerapaneni. High-order close evaluation of Laplace layer potentials: A differential geometric approach. *arXiv preprint arXiv:2105.12683*, 2021.

[217] A.Z. Zinchenko and R.H. Davis. An efficient algorithm for hydrodynamical interaction of many deformable drops. *Journal of Computational Physics*, 157(2):539–587, 2000.