Numerical Techniques for the Solution of Partial Differential and Integral Equations on Irregular Domains with Applications to Problems in Electrowetting

by

Patrick McKendree Young

B.S., Linfield College, 2005

A thesis submitted to the

Faculty of the Graduate School of the

University of Colorado in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

Department of Applied Mathematics

2011

This thesis entitled:

Numerical Techniques for the Solution of Partial Differential and Integral Equations on Irregular
Domains with Applications to Problems in Electrowetting
written by Patrick McKendree Young
has been approved for the Department of Applied Mathematics

_____

Kamran Mohseni

_____

Per-Gunnar Martinsson

Date _____

The final copy of this thesis has been examined by the signatories, and we find that both the
content and the form meet acceptable presentation standards of scholarly work in the above
mentioned discipline.

Young, Patrick McKendree (Ph.D., Applied Mathematics)

Numerical Techniques for the Solution of Partial Differential and Integral Equations on Irregular
Domains with Applications to Problems in Electrowetting

Thesis directed by Prof. Kamran Mohseni and Prof. Per-Gunnar Martinsson

Digital microfluidics is a rapidly growing field wherein droplets are manipulated for use in small-scale applications such as variable focus lenses, display technology, fiber optics, and lab-on-a-chip devices. There has been considerable interest in digital microfluidics and the various methods for liquid actuation by thermal, chemical, and electrical means, where each of the actuation methods make use of the favorable scaling relationship of surface tension forces at the micro scale.

Another increasingly important field is addressing the ever growing need for improved heat transfer techniques in the next generation of electronic devices. As device size decreases and device efficiency increases, high heat flux removal capabilities (100 - 1000 W/cm^2) are critical to achieve the lower device operating temperatures necessary to ensure reliably and performance.

In this thesis, we investigate the nature of the forcing that occurs in the transport of liquid drops by electrical means. The effects of system parameters on the force density and its net integral are considered in the case of dielectrophoresis (insulating fluids) and electrowetting-on-dielectric (conductive fluids). Moreover, we explore the effectiveness of a new heat transfer technique called digitized heat transfer (DHT), where droplets are utilized to enhance the removal of heat from electronic devices. Numerical computations of the Nusselt number for these types of flows provide strong evidence of the effectiveness of DHT in comparison to continuous flows.

These two physical phenomena are but two examples that illustrate the growing need for numerical techniques that simply and efficiently handle problems on irregular domains. We present two algorithms appropriate in this environment. The first extends the recently introduced Immersed Boundary Projection Method (IBPM), originally developed for the incompressible Navier-Stokes equations, to elliptic and parabolic problems on irregular domains in a second-order accurate man-

ner. The second algorithm employs a boundary integral approach to the solution of elliptic problems in three-dimensional axisymmetric domains with non-axisymmetric boundary conditions. By using Fourier transforms to reduce the three-dimensional problem to a series of problems defined on the generating curve of the surface, a Nyström discretization employing generalized Gaussian quadratures can be applied to rapidly compute the solution with high accuracy. We demonstrate the high order nature of the discretization. An accelerated technique for computing the kernels of the reduced integral equations is developed for those kernels arising from Laplace's equation, overcoming what was previously the major obstacle in the solution to such problems. We extend this technique to a wide class of kernels, with a particular emphasis on those arising from the Helmholtz equation, and provide strong numerical evidence of the efficiency of this approach. By combining the above approach with the Fast Multipole Method, we develop an efficient and accurate technique for solving boundary integral equations on multiply connected domains.

Acknowledgements

First and foremost, I would like to thank my advisors Kamran Mohseni and Gunnar Martinsson for everything they've taught me throughout my time at the University of Colorado. This dissertation would never have been possible without their knowledge, help, and support and for that, I am grateful.

The faculty in the Department of Applied Mathematics has been wonderful. In particular, Tom Manteuffel has been a good friend and bicycle riding comrade, and been exceedingly helpful in my job search. Jim Curry has given me some great advice over the years and has always made time to help me out.

Van Henson was nice enough to bring me out to Lawrence Livermore National Laboratory for a summer for an internship that was both insightful and a nice distraction from my dissertation.

Stephen Bricher and Joelle Murray really started my interest in applied mathematics and computation and were excellent mentors to me during my time at Linfield.

I have been fortunate enough to make some great friends while in Colorado, without whom I would have never been able to enjoy the many bike rides, trail runs, ski slopes, and all the other incredible things to do in this state that kept me going while in school.

Finally, my parents and family have been so very supportive of me and I am so very thankful to them for that.

Contents

Chapter

Bibliography   113

Appendix

Tables

Figures

Figure

Chapter 1

Introduction

This thesis represents a collected body of work by the author (under the guidance of his advisors) that closely investigates two problems that arise in the field of microfluidics, and develops several numerical algorithms that are relevant to simulating these phenomena.

In Chapters 2 and 3 of this thesis, we provide an in depth investigation into two physical phenomena. In Chapter 2, we investigate a novel technique for heat removal called digitized heat transfer (DHT). This technique relies on a *discretized* flow in place of a continuous flow. We provide strong numerical evidence of the increased Nusselt numbers in DHT in comparison to the classical Graetz flow. In Chapter 3, we focus on the nature of forcing that arises in the electrical actuation of liquid drops. We consider the limiting cases of electrically conducting and electrically insulating fluids. The former is commonly known as electrowetting-on-dielectric (EWOD) and the latter is referred to as dielectrophoresis. We formulate the equations governing the forcing for these actuation methods and clear up a common discrepancy regarding the nature of the forcing in dielectrophoresis. We give an in depth numerical investigation for the the net force experienced by a droplet, particularly in regards to edge effects when the liquid is near the fringing field of an electrode. From this, we give several recommendations for the design of devices that utilize electrowetting and dielectrophoresis to transport droplets.

The work presented in Chapters 2 and 3 has also been published, and can additionally be found in the following papers:

- P.M. Young and K. Mohseni. DEP forcing for application in digital microfluidics. Annals

of the New York Academy of Sciences, 1161, 463-471, 2009.

- P.M. Young and K. Mohseni. Calculation of DEP and EWOD forces for application in digital microuidics. ASME J. Fluid Eng., 130(8), 2008.

- P.M. Young and K. Mohseni. The effect of droplet length on Nusselt numbers in digitized heat transfer. Proceedings of the IEEE ITherm 2008: Eleventh Intersociety Conference on Thermal and Thermomechanical Phenomena in Electronic Systems, Lake Buena Vista, FL, May 28-31, 2008.

- E. Baird, P.M. Young, and K. Mohseni. Electrostatic force calculation for an EWOD-actuated droplet. Microfluid Nanofluid, 3(6) 635-644, 2007.

Chapters 2 and 3 give but a few examples of complex problems that are commonly encountered by today's engineers and scientists. Of course, as the complexity of such problems grows, we are in an ever growing need of more accurate and refined numerical techniques to model them.

In Chapter 4, we present a numerical technique that is applicable to elliptic and parabolic problems containing an immersed boundary upon which a Dirichlet boundary condition is present. By incorporating ideas from both the Immersed Boundary Method (IBM) and the Immersed Interface Method (IIM), we can overcome major limitations of both procedures - the lack of spatial accuracy in the IBM and the requirement on *a priori* knowledge of jump conditions in the IIM. A force along this boundary is found implicitly to enforce the boundary condition using a structure similar to the Immersed Boundary Projection Method (IBPM). By modifying the IBPM with the methodology employed by the IIM and avoiding the use of discrete delta functions when interpolating to the immersed boundary, second-order spatial accuracy is achieved at all points in the Cartesian mesh.

In Chapter 5, we present a technique for solving boundary integral equations on axisymmetric domains. We make no such assumption on the boundary load; it is a fully three dimensional problem. Under the axisymmetric assumption, it is possible to decouple the integral equation into a

series of integral equations defined on the generating curve of the body. We combine this approach with a high order Nyström discretization that employs special purpose Gaussian quadratures to integrate the weakly singular kernel of the reduced equations. The reduction to a series of integral equations defined on the generating curve presents an additional difficulty in that the kernels of the new integral equations are no longer given explicitly; they are defined in terms of a Fourier integral on the torus. In past approaches, this has been a major bottleneck in utilizing the axisymmetric approach. We have overcome this difficulty for the kernels related to Laplace's equation by exploiting recursion relations and the FFT. Further, we have extended this technique to a wide variety of kernels, with particular emphasis on the single and double layer kernels associated with the Helmholtz equation. We have developed efficient algorithms for computing solutions to the Laplace and Helmholtz equations on simply connected domains. These algorithms are very efficient; problems with hundreds of thousands of unknowns can be setup, inverted, and applied to a vector in a handful of minutes. If additional boundary loads are required to be solved, this can be done in a fraction of a second. A further advantage of our approach is that direct solvers can be utilized, mostly circumventing issues regarding any ill conditioning of the linear systems, which is particularly prevalent for the Helmholtz equation. We have also combined the axisymmetric methodology with the Fast Multipole Method (FMM) to efficiently solve problems on multiply connected domains. This class of problems demonstrates clearly the benefit of the axisymmetric approach; we observe that the bottleneck in this algorithm arises the FMM, an algorithm that runs in linear time for the problems we consider.

We also include two appendices, one with tables of the special purpose quadratures used to discretize the integral equations, and a second on numerical homogenization via the approximation of the solution operator. We include the latter as a further demonstration of the usefulness of the integral equation approach when dealing with complex domains.

The algorithms presented in Chapters 4 and 5 and the appendices are currently (or soon to be) under review for publication. The papers currently under review are listed below, and we are in the process of finishing preprints on articles related to the extension of the BIE algorithm to the

Helmholtz equation and multiply connected domains.

- P.M. Young and P.G. Martinsson. A direct solver for the rapid solution of boundary integral equations on axisymmetric surfaces in three dimensions. 2010.
  In review. http://arxiv.org/abs/1002.2001v1

- A. Gillman, P.M. Young, P.G. Martinsson. Numerical homogenization via approximation of the solution operator. To appear in the proceedings of the Numerical Analysis of Multiscale Computations workshop in Banff, December 2010.

- P.M. Young and K. Mohseni. A second-order immersed boundary projection method for elliptic and parabolic problems. 2010. In review.

We have made concluding remarks at the end of each chapter, indicating how results can be extended and future directions of work. The complexity of the physical problems we have explored in Chapters 2 and 3 are extremely challenging when coupled together into a full, direct simulation. The analysis and algorithms developed in this thesis should represent a major step in achieving such a goal.

Chapter 2

Digitized Heat Transfer

2.1   Introduction

The next generation of electronic devices is facing an ever growing need for improved heat transfer techniques. As device size decreases and device efficiency increases, high heat flux removal capabilities (100 - 1000 W/cm^2) are critical to achieve the lower device operating temperatures necessary to ensure reliably and performance [58].

Candidates for achieving high flux heat removal include jet-impingement cooling, liquid-immersion cooling, microchannel heat sinks, pool boiling, thermoelectric cooling, and spray cooling [33, 52, 81, 82, 103, 113, 127, 131, 132, 154]. The cooling mechanism investigated here is most closely related to liquid cooling through microchannels, first proposed in 1981 by Tuckerman and Pease [144]. A comprehensive review of microchannel transport can be found in [57] and a detailed summary of experimental results for single-phase heat transfer in microchannels can be found in [112].

In addition to steady state cooling requirements, "hot spots" are a major consideration for thermal management of integrated circuit technology [75, 154]. Rather than attempt to increase the heat removal capabilities of the entire device to handle these heat spikes, it would be more efficient if a technique could target hot spots individually and on-demand while working in conjunction with conventional heat transfer technology.

Digital microfluidics is an fast growing field [38, 141] wherein droplets are manipulated for small-scale applications such as variable focus lenses, display technology, fiber-optics, and lab-on-a-

(a) Continuous Flow

(b) DHT Flow
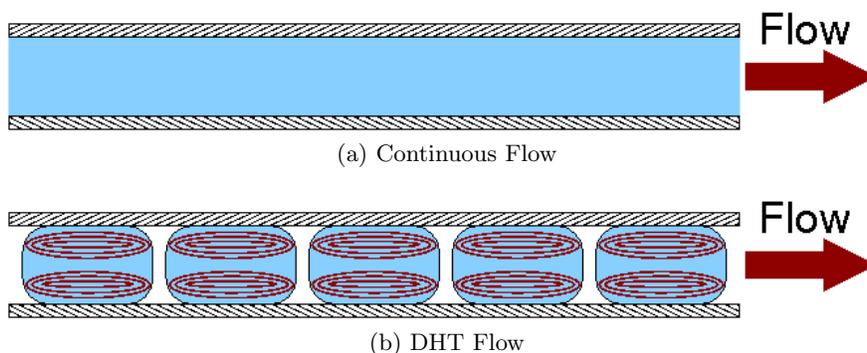
Figure 2.1: Different flows in a parallel plate geometry.

chip devices. There has been considerable interest in digital microfluidics and the various methods for liquid actuation including thermal [39, 115], chemical [56], and electrical means [6, 79, 88, 114, 123, 133, 148], where the actuation methods make use of the favorable scaling relationship of surface tension forces at the micro scale. A detailed investigation into the nature of the forcing that induces motion in liquid droplets by electrical means is provided in Chapter 3 of this thesis. The use of droplets for thermal management, referred to here as "digital heat transfer" (DHT), has been proposed in previous works [105, 106, 108, 109, 116], but an investigation of the heat transfer capabilities of DHT is lacking. This chapter investigates the heat transfer capabilities of DHT in comparison with continuous flow in the parallel plate geometry, commonly known as the Graetz problem [10], see Figure 2.1.

2.2   Digitized Heat Transfer

For the following investigation, it is assumed that droplet digitization and transport has been achieved using either chemical, thermal, or electrical methods. Electrical methods of actuation appear to be particularly well suited to DHT. Electrodes are patterned in such a way that a droplet can be transported anywhere along the surface where the heat is to be removed. If the droplet is conductive, a dielectric layer is required to insulate the droplet away from the electrodes. This transport mechanism is commonly referred to as electrowetting-on-dielectric (EWOD). No moving parts are required to activate a drop by this method, it requires little power, and has
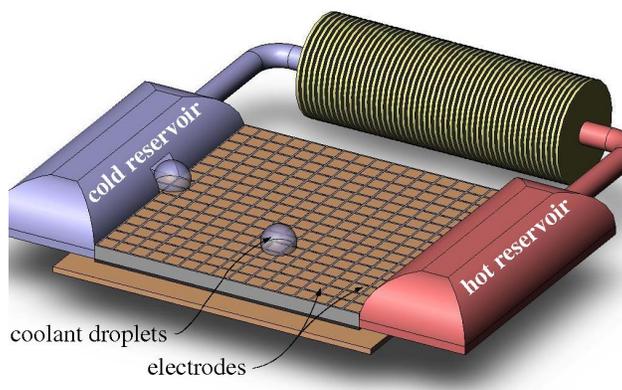
Figure 2.2: Concept of a DHT system driven by EWOD actuation.

experimentally observed droplet velocities of 25 cm/s [32]. Furthermore, liquids that are excellent electrical conductors often perform well as thermal conductors, such as liquid metals. Their use has been proposed for high heat flux chip cooling [103]. See Figure 2.2 for a concept schematic of a EWOD driven DHT system and Chapter 3 for more details on EWOD and dielectrophoresis.

Using patterned electrodes to transport a droplet allows for droplet trajectories to be fully programmable. In view of the increasing demand to handle hot spots effectively, DHT provides a possible method for handling hot spots. If a hot spot was detected, a droplet could immediately be drawn from a reservoir and brought to the hot spot along a programmed path anywhere on the chip. Once the temperature spike had been absorbed, the droplet would simply be transported to a reservoir off of the chip for cooling.

Regardless of the actuation method, DHT only requires the use of a digital flow in place of a continuous flow. It is the discrete nature of the flow that allows for the improvement in heat transfer over continuous flows.

2.3   Governing Equations

For the DHT flow considered in this chapter, a droplet has replaced a continuous flow between two parallel plates, and the flow is assumed to be two dimensional. The no-slip condition present at the wall induces a circulation inside the droplet as it travels between the plates [106], and this

characteristic behavior of the flow has been observed experimentally [93]. Twin vortices form inside the droplet, and it is these vortices that enhance the convection inside the droplet in comparison to the continuous flow, where the velocity field is simply that of Poiseuille flow. A recirculation time scale for such a flow has been identified previously [5].

Throughout this discussion surface forces are neglected. It is assumed that the mean droplet velocity U is constant, and that the droplet is symmetric about the horizontal plane that divides it in half.

The governing equations for an incompressible DHT flow are those that arise from the conservation of mass, momentum, and energy. These are given by

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = 0, \tag{2.1}$$

$$\rho \left(\frac{\partial u}{\partial t} + \frac{\partial (u^2)}{\partial x} + \frac{\partial (uv)}{\partial y} \right) = -\frac{\partial p}{\partial x} + \mu \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right), \tag{2.2}$$

$$\rho \left(\frac{\partial v}{\partial t} + \frac{\partial (uv)}{\partial x} + \frac{\partial (v^2)}{\partial y} \right) = -\frac{\partial p}{\partial y} + \mu \left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} \right), \tag{2.3}$$

$$\rho c_p \left(\frac{\partial T}{\partial t} + u\frac{\partial T}{\partial x} + v\frac{\partial T}{\partial y} \right) = k \left(\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} \right), \tag{2.4}$$

respectively, where t is time, x and y are the horizontal and vertical coordinates, u and v are the horizontal and vertical velocities, p is the fluid static pressure, T is the fluid temperature, and ρ, μ, c_p, and k are the density, dynamic viscosity, specific heat, and thermal conductivity of the fluid.

2.3.1   Nondimensionalization

Define

$$\bar{t} = \frac{U}{H}t, \quad \bar{x} = \frac{1}{L}x, \quad \bar{y} = \frac{1}{H}y, \quad \bar{u} = \frac{1}{U}u, \quad \bar{v} = \frac{L}{HU}v, \quad \bar{p} = \frac{p - p_\infty}{\rho U^2}, \quad \text{and} \quad \bar{T} = \frac{T - T_{ref}}{T_c},$$

where the overbar denotes a nondimensional variable, H and L are the height and length of the droplet, U is the mean velocity of the droplet, p_∞ is the reference pressure, and T_{ref} and T_c are reference temperatures that are defined later in this section. Define

$$A = \frac{H}{L}, \quad \nu = \frac{\mu}{\rho}, \quad \text{Re} = \frac{HU}{\nu}, \quad \alpha = \frac{k}{\rho c_p}, \quad \text{and} \quad \text{Pe} = \frac{HU}{\alpha},$$
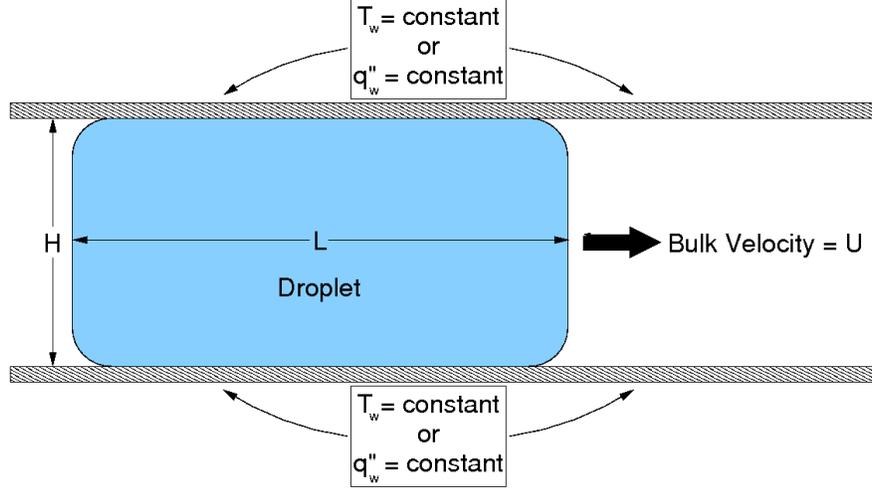
Figure 2.3: Geometry of the DHT setup.

where A is the aspect ratio of the droplet, ν is the kinematic viscosity, Re is the Reynolds number, α is the thermal diffusivity, and Pe is the Peclet number. Then equations (2.1) - (2.4) become

$$\frac{\partial \bar{u}}{\partial \bar{x}} + \frac{\partial \bar{v}}{\partial \bar{y}} = 0, \tag{2.5}$$

$$\frac{\partial \bar{u}}{\partial \bar{t}} + A\left(\frac{\partial(\bar{u}^2)}{\partial \bar{x}} + \frac{\partial(\bar{u}\bar{v})}{\partial \bar{y}}\right) = -A\frac{\partial \bar{p}}{\partial \bar{x}} + \frac{1}{\text{Re}}\left(A^2\frac{\partial^2 \bar{u}}{\partial \bar{x}^2} + \frac{\partial^2 \bar{u}}{\partial \bar{y}^2}\right), \tag{2.6}$$

$$\frac{\partial \bar{v}}{\partial \bar{t}} + A\left(\frac{\partial(\bar{u}\bar{v})}{\partial \bar{x}} + \frac{\partial(\bar{v}^2)}{\partial \bar{y}}\right) = -\frac{1}{A}\frac{\partial \bar{p}}{\partial \bar{y}} + \frac{1}{\text{Re}}\left(A^2\frac{\partial^2 \bar{v}}{\partial \bar{x}^2} + \frac{\partial^2 \bar{v}}{\partial \bar{y}^2}\right), \tag{2.7}$$

$$\frac{\partial \bar{T}}{\partial \bar{t}} + A\left(\bar{u}\frac{\partial \bar{T}}{\partial \bar{x}} + \bar{v}\frac{\partial \bar{T}}{\partial \bar{y}}\right) = \frac{1}{\text{Pe}}\left(A^2\frac{\partial^2 \bar{T}}{\partial \bar{x}^2} + \frac{\partial^2 \bar{T}}{\partial \bar{y}^2}\right). \tag{2.8}$$

There are two common boundary conditions to consider, constant wall temperature and constant wall flux.

2.3.1.1   Constant Wall Temperature

If the wall temperature is held constant, we set $T_{ref} = T_w$ and $T_c = T_w - T_m$, where T_w and T_m are the fluid temperature at the wall and the mean temperature of the fluid.

2.3.1.2   Constant Wall Flux

If the heat flux is held constant at the wall, we set $T_{ref} = T_I$ and $T_c = Hq''/k$, where T_I is the initial temperature of the fluid and q'' is the heat flux at the wall.

2.3.2 Nusselt Number

In laminar duct flow problems, the heat transfer coefficient h is conventionally defined as [10]

$$h = \frac{q''}{T_w - T_m},$$ (2.9)

the heat flux at the wall is given by

$$q'' = k\frac{\partial T}{\partial \boldsymbol{n}}\bigg|_w,$$ (2.10)

where $\boldsymbol{n}$ is the normal at the wall, and the mean temperature of the fluid at a given point in the channel is defined as

$$T_m = \frac{1}{HU} \int_0^H uT\, dy.$$ (2.11)

If T is nondimensionalized using the relation

$$\tilde{T} = \frac{T - T_w}{T_w - T_m},$$

the natural way of defining the local Nusselt number Nu_ℓ is given by

$$\mathrm{Nu}_\ell = \frac{Hh}{k} = \frac{\partial \tilde{T}}{\partial \bar{y}}\bigg|_{\bar{y}=0}.$$ (2.12)

This definition of Nu_ℓ cannot be used directly because temperature is not necessarily nondimensionalized in terms of T_m and T_w. However, Nu_ℓ can be related to nondimensional temperature used for the constant wall temperature and flux boundary conditions [10]. For the constant wall temperature boundary condition,

$$\mathrm{Nu}_\ell = \frac{1}{\bar{T}_m} \frac{\partial \bar{T}}{\partial \bar{y}}\bigg|_{\bar{y}=0}$$ (2.13)

and for the constant heat flux at the wall boundary condition

$$\mathrm{Nu}_\ell = \frac{1}{\bar{T}_w - \bar{T}_m}.$$ (2.14)

Note that when there is a constant wall temperature, Nu_ℓ is a measure of the change in heat flux at the wall, while if the heat flux is constant, Nu_ℓ is a measure of the wall temperature itself.

The mean Nusselt number Nu_m is given by

$$\mathrm{Nu}_m = \frac{1}{L} \int_0^L \mathrm{Nu}_\ell\, dx.$$ (2.15)

(a) Horizontal Velocity

(b) Vertical Velocity
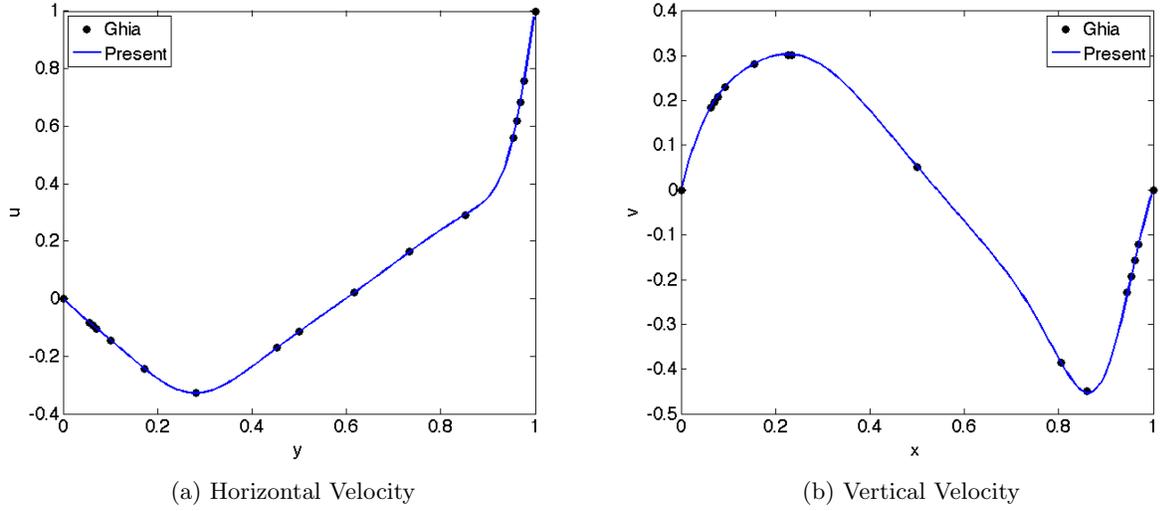
Figure 2.4: Velocities along lines through the geometric center of the cavity for Re = 400. The velocity field was developed in time until the condition given in (2.16) was met. Results were compared with [60].

2.4   Computational Technique

In the frame of reference of the droplet, we see that the flow arising in DHT is very similar to that of the lid driven cavity, with the caveat that the DHT flow is driven from both the top and the bottom [5]. With this in mind, the technique used to compute the velocity field is the well known projection method for incompressible flows, with use of a MAC mesh [122] (a staggered grid with horizontal velocity nodes on the left and right, vertical velocity nodes on the top and bottom, and pressure nodes at the center of each cell in the grid) and standard central finite differences in space. A first-order explicit time discretization is used, and the velocity field is iterated until steady state is reached, given by the condition

$$\|u^{n+1} - u^n\| \leq 10^{-8} \tag{2.16}$$

where the norm above is the discrete l_2 norm.

Boundary conditions for the droplet are assumed to be no slip at the walls. To ensure that the solver is performing as desired, the algorithm was benchmarked against the results of the standard driven cavity flow given by Ghia [60]. As seen in Figure 2.4, the convergence criterion given in

(a) $A = 1$

(b) $A = \frac{1}{2}$

(c) $A = \frac{1}{4}$

(d) $A = \frac{1}{8}$
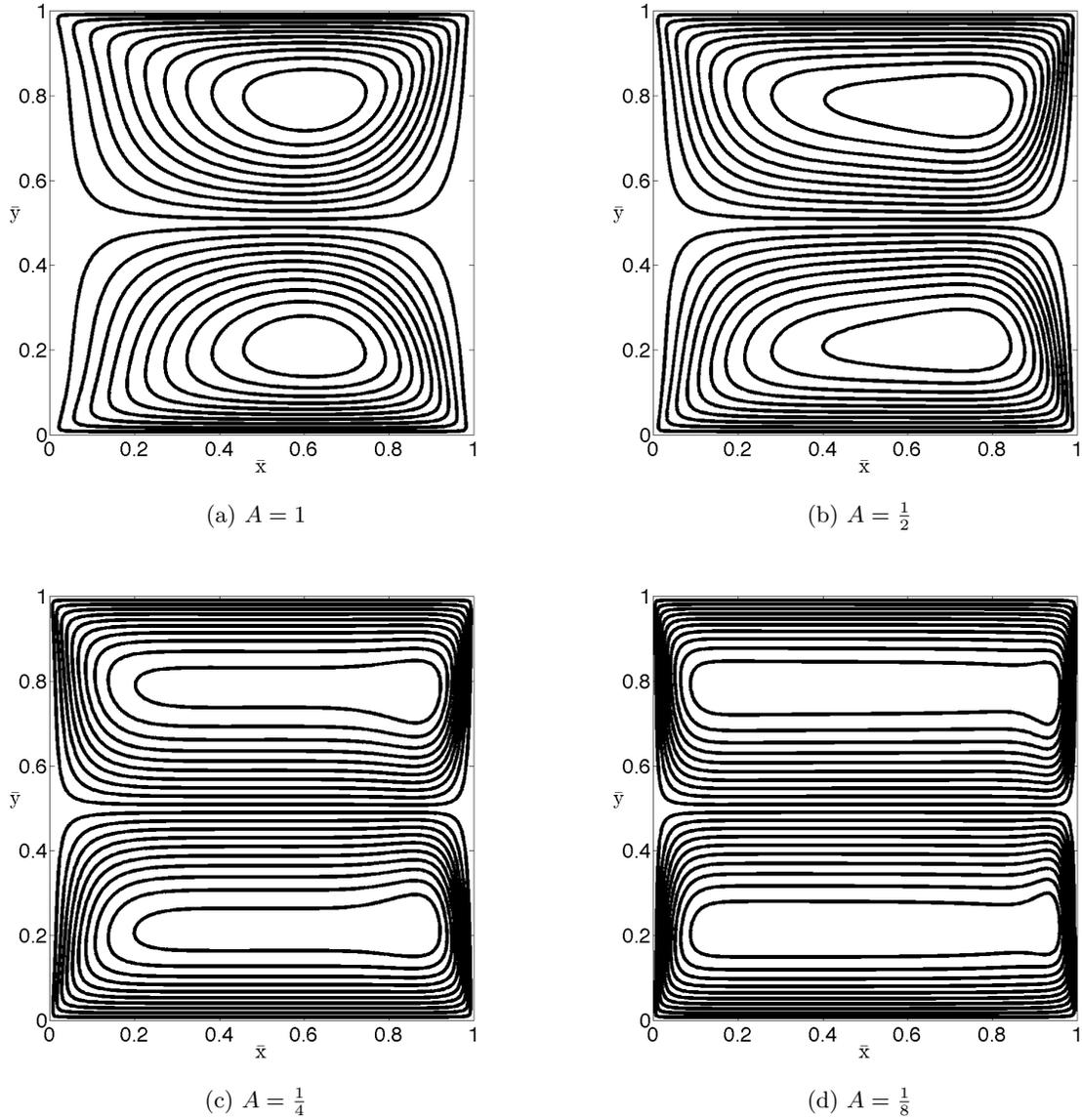
Figure 2.5: Streamlines of flow inside droplets with various aspect ratios A at Re $= 100$.

(2.16) is sufficient in driving the velocity field to steady state.

Figure 2.5 shows the steady-state streamlines for droplets of increasing length in a DHT system with Re $= 100$. As the aspect ratio A decreases, corresponding to the droplet growing in length, the streamlines begin to straighten out horizontally.

Once the steady-state velocity is obtained for a particular value of A, the energy equation (2.8) can be solved. $\bar{T}$ is solved at the cell centers of the MAC mesh, with centered second-order

(a) T_w constant   (b) q_w'' constant
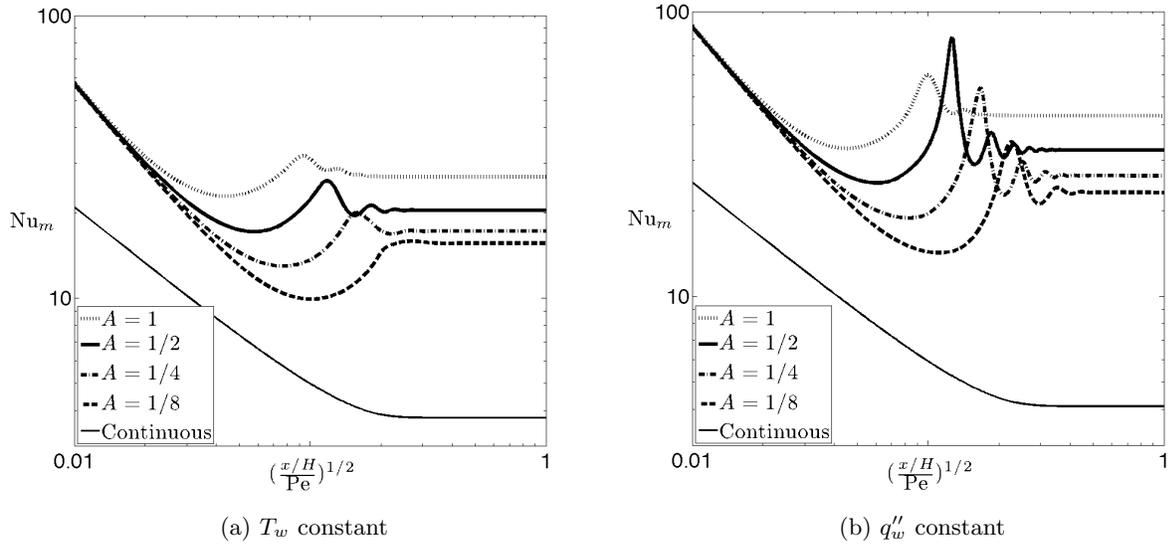
Figure 2.6: The Nusselt number Nu_m plotted against the nondimensional spatial coordinate $\left(\frac{x/H}{\mathrm{Pe}}\right)$. This is coordinate that is usually used when giving results for the continuous Graetz problem [10].

finite differences applied in space and a explicit first-order finite difference discretization used in time. At each time step, Nu_m is calculated using either (2.14) or (2.15) depending upon which temperature boundary condition is present at the wall. The droplet interface itself is assumed to be thermally insulating. The setup is given in Figure 2.3.

2.5   Results

Figure 2.6 displays Nu_m for the constant wall temperature and constant wall flux boundary conditions plotted against the nondimensionalized spatial coordinate $(x/H)/\mathrm{Pe}$. This coordinate can also be viewed as the nondimensionalized distance that the drop has travelled down the heated channel. Included in this figure is the Nusselt number for a continuous flow, the classical Graetz problem. Independent of the boundary conditions, a strong oscillatory behavior is seen initially in the case of a discrete flow. The values of Nu_m then proceed to asymptote to a constant value. In all investigated cases of droplets with different aspect ratios, it is seen that the values of Nu_m achieved are greater than that achieved in continuous Graetz flow.

(a) T_w constant

(b) q_w'' constant
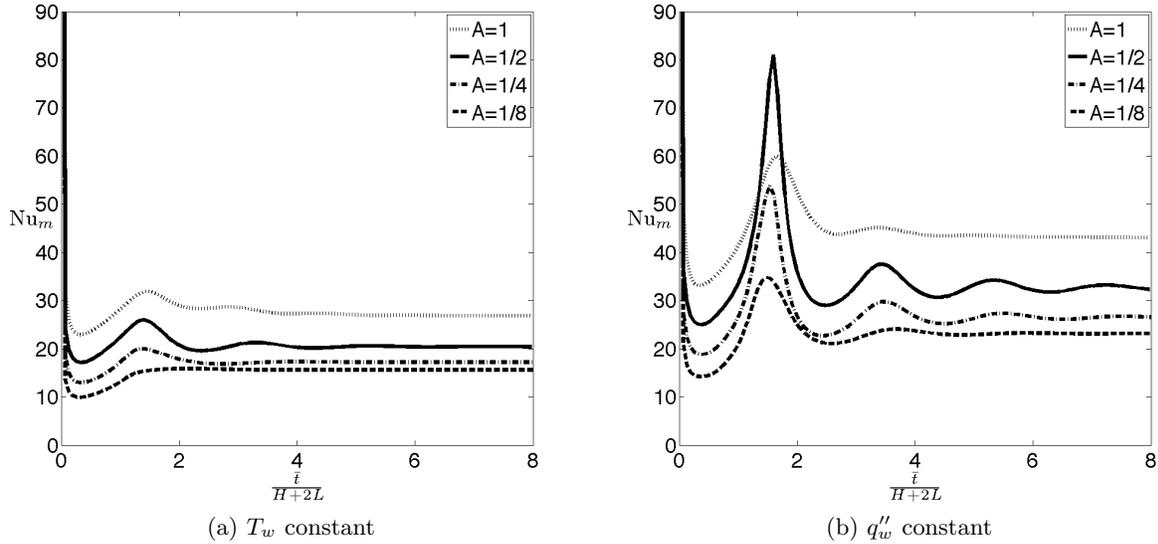
Figure 2.7: The Nusselt number Nu plotted against the period of fluid rotation $\approx \frac{1}{H+2L}$.

This increase in Nusselt number in DHT flows is a direct consequence of the addition of interfaces inducing a wall-normal velocity component to the flow. This additional convection increases the mixing inside the droplet in comparison to continuous flows, providing the larger Nusselt number. A larger aspect ratio (corresponding to a shorter droplet) results in a larger asymptotic Nusselt number because there is relatively more cool fluid inside the droplet that can be brought into contact with the heated wall in comparison to a longer droplet.

Figure 2.7 displays the same values of Nu_m as in Figure 2.6, except the scaling used in plotting Nu_m is the approximate recirculation time in the cavity, $\bar{t}/(H + 2L)$ [5]. Using this scaling, the peaks in Nu_m all align. This gives a strong indication that the peaks in Nu_m are a direct result of droplet circulation. Heuristically, the initial climb in Nu_m can be attributed to the cold fluid that is initially present in the center of the droplet being transported to the heated walls by the droplet's internal circulation. This rise in Nu_m reaches a maximum, and then begins to decline as fluid that has previously been heated completes its circulation and is brought back to the heated walls. This cycle repeats, resulting in a chain of oscillations, until a balance between droplet circulation and heat diffusion is reached. At this point, the oscillations have been completely damped, and the
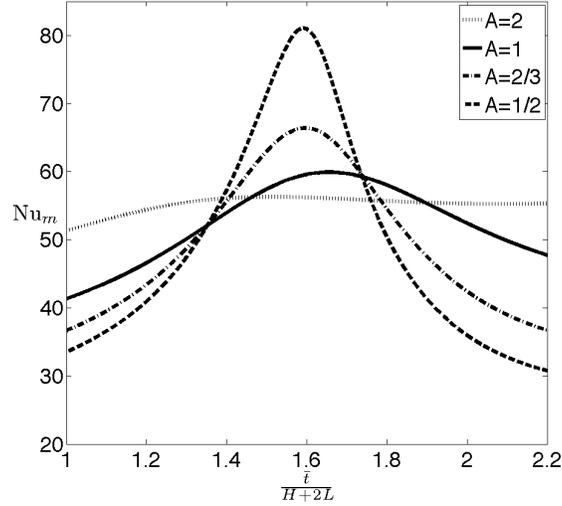
Figure 2.8: Peak of Nusslet number for various aspect ratios.

value of Nu_m is steady.

A curious phenomenon can be seen in Figure 2.7(b); the peak value of Nu_m is greater for $A = 1/2$ than for $A = 1$. This behavior is not present in Figure 2.7a, the constant wall temperature case, where a greater aspect ratio results in a greater value of Nu_m for any given time. Figure 2.8 gives a detailed view of the peak in Nu_m for aspect ratios of $A = 2$, $A = 1$, $A = 2/3$, and $A = 1/2$ with constant wall flux.

This phenomenon can be explained by looking to the characteristic time scales for heat diffusion. For a longer droplet, the majority of the heat diffusion is occurring in the wall-normal direction. As the droplet becomes shorter, diffusion in the wall-tangent direction plays a growing role. When these two diffusion times are near equal in magnitude, the highest peaks in Nu_m are achieved. As the wall-tangent diffusion begins to dominate, the peak in Nu_m begins to decline. Figures 2.9, 2.10, and 2.11 show this transition occurring from longer to shorter droplets.

This phenomenon is not seen in the constant wall temperature case, because diffusion time in the wall normal direction is greater for this boundary condition. This is a result of the constant flux boundary condition introducing more heat to the fluid as it travels along the heated wall. In the constant temperature case, the fluid becomes more and more saturated as it travels along the

(a) Tempurature
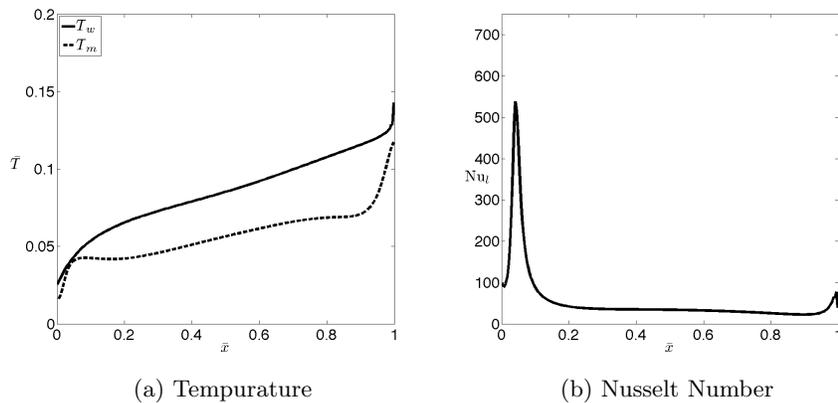
(b) Nusselt Number

Figure 2.9: Local temperature and Nusselt number plots at peak value of Nu_m for $A = 1/4$.

(a) Tempurature
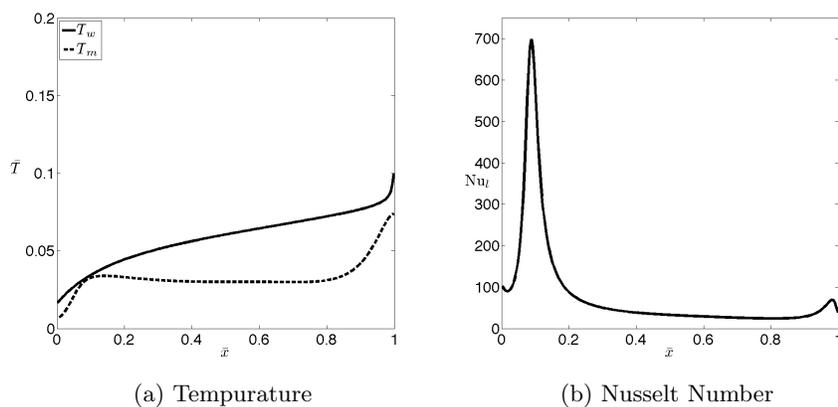
(b) Nusselt Number

Figure 2.10: Local temperature and Nusselt number plots at peak value of Nu_m for $A = 1/2$.

(a) Tempurature
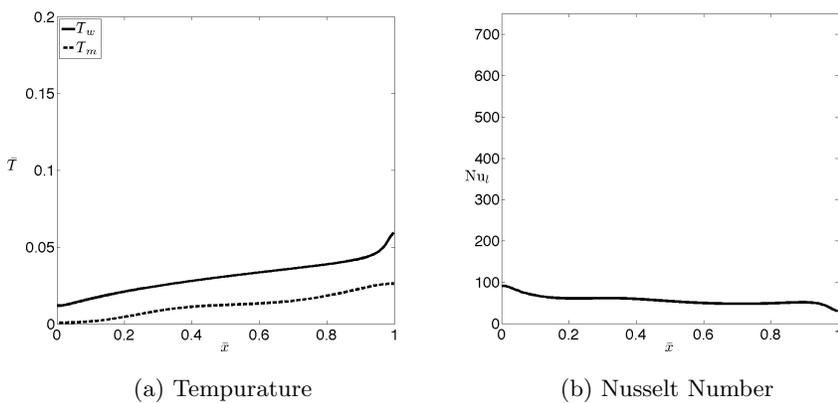
(b) Nusselt Number

Figure 2.11: Local temperature and Nusselt number plots at peak value of Nu_m for $A = 2$.

heated wall, limiting the amount of heat it can absorb. This is not the case for constant heat flux boundary conditions at the wall. It is possible that the same effect in peak height may occur for the constant wall temperature case, but for larger values of A.

The oscillations of Nu_m discussed previously are a phenomenon that occurs when the droplet first begins to remove heat. It is damped out once the droplet has traveled its own length in distance a few times. It may well be that a device utilizing digitized heat transfer may require the droplet to travel a distance considerably greater than the droplet length, in which case the oscillations would have little effect on the overall heat transfer capability of the device.

2.6   Conclusions

The increasing demand for efficient, compact heat removal techniques has made thermal management an increasingly greater factor in the design of electronic devices. In this chapter, we have demonstrated the promise of DHT as a candidate for high heat flux removal and hot spot cooling. In both the constant wall and temperature cases, the higher Nusselt numbers obtained by DHT in comparison to continuous Graetz flow is a strong indicator of DHT's potential as a heat transfer mechanism. A periodic fluctuation in Nusselt number has been observed, and its origins investigated. There are many challenges in building a fully integrated DHT device, but in light of the encouraging results presented within this chapter, experimental research is desired to fully assess DHT as a method for heat removal.

Chapter 3

The Characterization of Forcing in Electrowetting and Dielectrophoresis

3.1   Introduction

The field of digital microfluidics, in which discrete droplets are manipulated in place of continuous flows, has seen rapid development over the last few years for a variety of applications, from engineering to the life sciences [32, 37, 89, 123, 124, 133, 148], including variable focus lenses, display technology, fiber-optics, and lab-on-a-chip devices. In particular, efficient and cost-effective lab-on-a-chip devices are in great demand, as they allow for highly repetitive laboratory tasks to become automated with the introduction of miniaturized and integrated systems [29]. This technique typically makes use of forces possessing favorable scaling relationships; prevalent examples include thermal or chemical surface tension modulation, electrowetting, and dielectrophoresis. In our group, digital microfluidics has been employed for active thermal management of compact electronic devices [5, 105, 106], design of a zero leakage microvalve [110], investigation of droplet morphology under electrowetting actuation [42], and design of an electrowetting microlens [28]. In Chapter 3, we investigated the heat removal technique *digitized heat transfer*. This technique relies on the rapid and efficient movement of liquid droplets across a substrate, and as we will see, electrical methods for inducing droplet motion are excellent candidates for use in digitized heat transfer.

Accurate descriptions of actuation forces and resultant droplet velocities must be available when designing an integrated device making use of discretized flows. Currently, the most promising methods of droplet actuation in microfluidic devices are electrowetting on dielectric (EWOD) for

conductive droplets [49, 111] and dielectrophoresis (DEP) for electrically insulating droplets [40, 59, 77, 78, 79], where in both cases droplets are transported by sweeping an applied voltage along a microchannel ahead of the droplet. Numerical modeling of the droplet dynamics for EWOD and DEP configurations has been done using approximations of the electrostatic effect [42, 147], but incorporation of the electrostatic force density into a direct simulation of the fluid mechanics is desired. For an example of such a simulation, see [136]. This chapter presents numerical results describing the forces in DEP and EWOD for a droplet of fixed geometry. The lumped parameter result for the net droplet force is stated for DEP, and then compared with results from direct numerical simulation of electrostatic forces. An analysis of the lumped force acting upon an EWOD droplet has previously been presented by our group and others [4, 6, 37]. Here we extend these results to DEP, investigate the net forces and force distributions in DEP actuation, and provide comparison with EWOD.

The primary difference between a EWOD and DEP actuated droplet is the nature of the fluid and its effect on the electric field's penetration into the media, see Figure 3.1 for the basic setup of EWOD and DEP. For EWOD, an electrically conducting droplet is placed in a dielectric-coated channel lined with electrodes. A given electrode is then activated, creating an electric field that induces a charge accumulation on the surface of the fluid. This charge accumulation allows for the creation of a net force upon the droplet, drawing the droplet towards the actuated electrode. Investigation of the interface profile and electrostatic distribution in EWOD has been explored in several papers [23, 80, 145]. The charge distribution near the contact line was found analytically, giving rise to a force distribution that is clustered in a region near the contact line on the order of h, the thickness of the dielectric layer. The interface profile used in this chapter is assumed circular and fixed for all time, so any dynamic response to the electric field is not included.

DEP differs from EWOD in that the liquid is insulating, charge does not accumulate on the surface, and the electric field penetrates into the liquid. It is well known that a dielectric material is drawn into the gap between the parallel plate of a charged capacitor [76]. This is a result of the non-uniform fringing field located at the edge of the capacitor, providing a force pointing towards

its center [76, 87]. As opposed to EWOD, the dielectrophoretic force can act over the droplet's front face or within the bulk of the fluid itself. Jones [77, 78] has explored the close relationship between DEP and EWOD on a theoretical basis, but for direct simulation of EWOD and DEP flows including electrostatic effects, a clear description of the force distribution is required. The cases considered here are for a perfect conductor (EWOD) and a perfect insulator (DEP), but there exists many fluids that exhibit properties of both a conductor and dielectric, namely leaky dielectrics [130].

3.2 Governing Equations

In the problems considered in this chapter, the focus is on droplet flow resulting from electrical forces. In many microfluidic applications the dynamic currents are so small that the magnetic field can be ignored. In this situation the governing equations for the electrical field are the electrostatic laws [1] .

The governing equations of motion for incompressible fluids under electric effects are the mass, momentum, and electrostatic equations. Aside from the mechanical forces (pressure and shear stress), there exists Columbic forces due to any existing free charges as well as forces due to polarization. The corresponding boundary conditions at a fluid interface are obtained by integrating the mass and momentum equations through the interface.

The net effect of an applied electrical field on a given fluid is represented by an extra body force on the right hand side of the Navier-Stokes equations. The body force density $\mathbf{f_b}$ in a fluid resulting from the influence of an electric field can be written as

$$\mathbf{f_b} = \rho_f \mathbf{E} - \frac{1}{2} E^2 \nabla \epsilon + \nabla (\frac{1}{2} \mathbf{E} \cdot \mathbf{E} \frac{\partial \epsilon}{\partial \rho}) \tag{3.1}$$

where ϵ is the fluid permittivity, ρ is the density of the fluid, ρ_f is the free electric charge density, and $\mathbf{E}$ is the electric field. This is the Korteweg-Helmholtz electric force density formulation

---

[1] For the electrostatic approximation to apply, the characteristic time scale for electric phenomena, $\tau = \epsilon/\sigma$ must be small. Note that τ is the ratio of dielectric permeability to conductivity of the medium. For the microfluidic applications considered here, this condition will usually be valid. For example, pure water has a electrical relaxation time of $\approx 10^{-4}$ and a hydrodynamic time scale of ≈ 0.045. See [99] for more details.

[117, 139, 150]. The last term in this equation, the electrostriction force density term, can be ignored for incompressible flows. Hence, the body force density considered here is given by

$$\mathbf{f_b} = \rho_f \mathbf{E} - \frac{1}{2}E^2 \nabla \epsilon. \tag{3.2}$$

This force density provides a coupling between the droplet hydrodynamics and electric field. The first term of (3.2) is attributed to free charge in the system, while the second term is the contribution from the polarization of the medium.

The body force density (3.2) can be identically written as the divergence of a stress tensor, namely the Maxwell stress tensor

$$\mathbf{f_b} = \nabla \cdot \mathbf{T}^{\mathrm{M}}, \tag{3.3}$$

or in tensorial notation

$$T_{ij}^{\mathrm{M}} = \epsilon E_i E_j - \frac{\epsilon}{2}\delta_{ij}E_k E_k. \tag{3.4}$$

This stress term accounts for both the forces due to free electric charges and the forces due to polarization of the material.

For a system consisting of perfectly insulating fluids (such as in DEP) with no free charge present, the first term in (3.2) can be disregarded. The second term in (3.2) is only nonzero at the interface between the two fluids, where there exists a gradient in ϵ. This indicates that the force density is located at the fluid interface, not distributed throughout the bulk of the fluid as the physical interpretation of the polarization effect indicates. The debate as to what point of view is correct is long standing, but these seemingly contradictory interpretations of the body force density can be shown to be equivalent if care is taken [13]. The body force density due to polarization forces is traditionally given by the Kelvin polarization body force density [98],

$$\mathbf{f_b} = (\mathbf{P} \cdot \nabla)\mathbf{E} = ((\epsilon - \epsilon_0)\mathbf{E} \cdot \nabla)\mathbf{E} = \epsilon \nabla \left(\frac{1}{2}E^2\right) - \nabla \left(\frac{\epsilon_0}{2}E^2\right) \tag{3.5}$$

where $\mathbf{P} = (\epsilon - \epsilon_0)\mathbf{E}$ is the polarization field. For an incompressible fluid, the scalar pressure only appears in the Navier-Stokes equations in terms of a gradient. The role of pressure is to ensure continuity of the vector field is satisfied, and it takes on whatever value is needed to guarantee this

condition is always fulfilled. Hence, any other term that appears in the Navier-Stokes equations as the gradient of a scalar can be absorbed into the pressure [98]. This is true of the last term in (3.5), and so the effective body force density in view of the Kelvin representation is

$$\mathbf{f_b} = \epsilon \nabla \left(\frac{1}{2} E^2 \right). \tag{3.6}$$

Now consider the Korteweg-Helmholtz formulation. Using the vector identity $\nabla(\phi\psi) = \phi\nabla\psi + \psi\nabla\phi$ and recalling that there is no free charge present in this system, (3.2) becomes

$$\mathbf{f_b} = \epsilon \nabla \left(\frac{1}{2} E^2 \right) - \nabla \left(\frac{\epsilon}{2} E^2 \right). \tag{3.7}$$

Since the last term in (3.7) is the gradient of a scalar, it can be absorbed into the pressure. Therefore, the difference in the Kelvin and Korteweg-Helmholtz body force densities is the gradient of a scalar, a term that has no dynamic significance for incompressible flows.

For a system where one fluid is significantly more conductive than the other (such as EWOD), the electric relaxation time for the conductive liquid can be assumed to be significantly shorter than the relevant hydrodynamic time scales. As a result, the interface can be regarded as a perfect conductor. For such a conductor under the influence of an electric field, all free charge in the system accumulates on the surface of the droplet. For such a setup, the force density is given by the first term in (3.2), except that for an ideal conductor we have $\rho_f \to \sigma_f$, where σ_f is the surface charge density. Note that this is no longer a body force density, it is confined to the surface of the droplet. The surface charge density can be rewritten using Gauss' Law and the field itself is expressed as the average electric field intensity at the surface. Applying these two conditions and recalling that the electric field is always normal to the surface of a conductor gives the surface force density $\mathbf{f_s}$ at the interface,

$$\mathbf{f_s} = \sigma_f \mathbf{E} = \frac{\epsilon_{ext}}{2} E^2, \tag{3.8}$$

where ϵ_{ext} refers to the dielectric constant of the fluid external to the conductive droplet [151].

Thus, the difference in the force distribution for dielectric and conductive fluids is that of a body force density as opposed to a surface force density. This is an essential difference in regards to a numerical implementation of these forces coupled with the fluid equations.
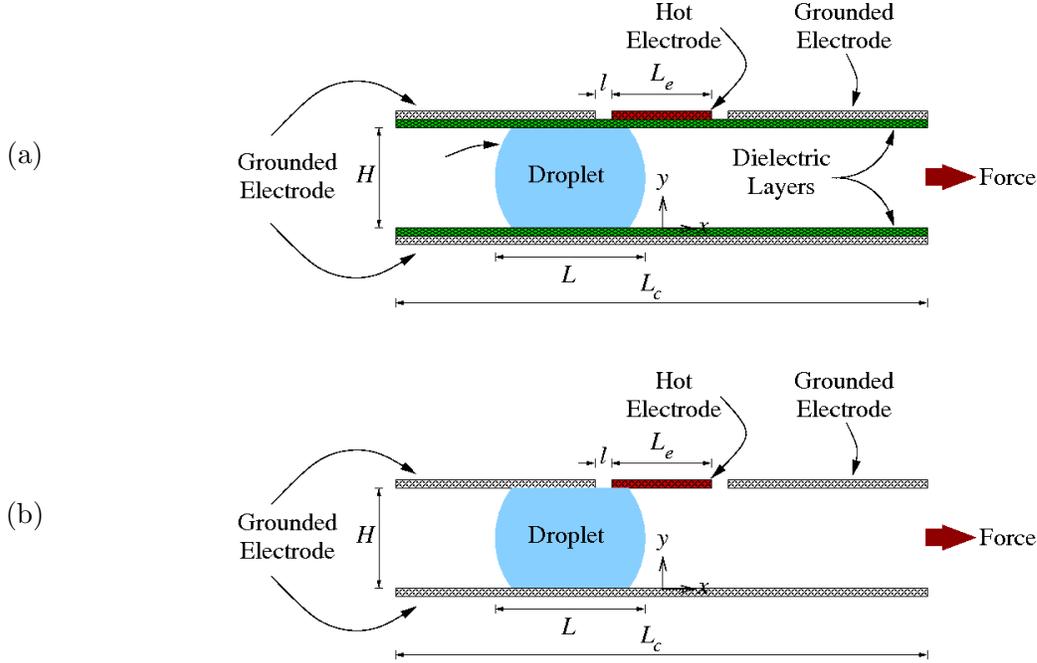
Figure 3.1: The (a) EWOD and (b) DEP configurations. H is the droplet and the channel height and L, L_e, and L_c, are the droplet, electrode, and channel widths. l is the spacing between the hot and grounded electrodes.

The investigation of the electric force acting upon droplets that are perfectly conductive and perfectly insulating is now considered in detail. The following introduces the nondimensionalization of the system considered in this chapter.

3.2.1 Nondimensionalization

Figure 3.1 shows the setup for both the EWOD and DEP systems considered in this chapter. Let the primes denote the nondimensionalized variables and set

$$x' = \frac{x}{H}, \quad y' = \frac{y}{H}, \quad V' = \frac{V}{V_0}, \quad \text{and} \quad \epsilon' = \frac{\epsilon}{\epsilon_0} \tag{3.9}$$

where H is the channel height, V_0 is the charged electrode voltage, and ϵ_0 is the electric permittivity of free space. This nomenclature is used to nondimensionalize the governing equations.

3.2.1.1   DEP

In electrostatics, $\mathbf{E} = -\nabla V$ defines the relationship between the electric field $\mathbf{E}$ and the electric potential V. Using the potential formulation, Gauss' Law reads

$$\nabla \cdot (\epsilon \nabla V) = \epsilon \left(\frac{\partial^2 V}{\partial x^2} + \frac{\partial^2 V}{\partial y^2} \right) + \frac{\partial \epsilon}{\partial x} \frac{\partial V}{\partial x} + \frac{\partial \epsilon}{\partial y} \frac{\partial V}{\partial y} = 0. \tag{3.10}$$

For the system considered in this chapter, ϵ is a continuous function that takes on the value ϵ_{ext} outside the droplet, ϵ_{int} inside the droplet, and continuously transitions between ϵ_{ext} and ϵ_{int} linearly on the scale of the grid. Nondimensionalization of (3.10) gives

$$\epsilon' \left(\frac{\partial^2 V'}{\partial x'^2} + \frac{\partial^2 V'}{\partial y'^2} \right) + \frac{\partial \epsilon'}{\partial x'} \frac{\partial V'}{\partial x'} + \frac{\partial \epsilon'}{\partial y'} \frac{\partial V'}{\partial y'} = 0 \tag{3.11}$$

In the presence of the electric field, the atoms in a dielectric material polarize. The resulting dipoles experience a force from the electric field, and this electrostatic force density $\mathbf{f_b}$ of the electrically insulating droplet is given by (3.5). Equation (3.5) can then be integrated over the entire domain to give the net force acting upon the insulating droplet. The relationship between the nondimensional body force density $\mathbf{f_b}'$ and the dimensional body force density $\mathbf{f_b}$ is

$$\mathbf{f_b} = \frac{\epsilon_0 V_0^2}{H^3} \mathbf{f_b}'.$$

3.2.1.2   EWOD

In EWOD, the droplet is a conductor and hence there is no electric field present internally. Therefore, equation (3.10) reduces to Laplace's equation,

$$\frac{\partial^2 V'}{\partial x'^2} + \frac{\partial^2 V'}{\partial y'^2} = 0$$

with the appropriate boundary conditions.

The charge distribution resulting from the electric field present on the conducting droplet feels a force from the external electric field, giving rise to an electrostatic force always felt normal to the surface, given by (3.8). The integration of this force density gives the net force on the

conductive droplet. The relationship between the nondimensional surface force density $\mathbf{f_s}'$ and the dimensional surface force density $\mathbf{f_s}$ for the EWOD configuration considered here is

$$\mathbf{f_s} = \frac{\epsilon_0 V_0^2}{H^2}\mathbf{f_s}'.$$

The use of the primes has been dropped in the remainder of this chapter for notational clarity.

3.3 Lumped Forced Calculations

The next two sections pertain to the forces at work in DEP and EWOD. The *total* force, per unit area, experienced by a droplet can be directly derived from capacitive energy considerations [76, 87]. Differentiation of the system energy U gives the net force F in the horizontal direction, per unit area,

$$F = \frac{dU}{dx}.$$

This method is demonstrated for both EWOD and DEP droplets in the section below. To calculate force *distributions*, numerical methods must be utilized, and this is presented in the second section.

3.3.1 Lumped EWOD Force

Under EWOD actuation, the droplet is a conductor, and so all the charge is located at the fluid interface. Hence, the droplet experiences a surface force near its front and rear contact lines. We consider here only the case of a continuously grounded electrode, with the opposing side of the droplet in contact with a hot electrode with potential V on its advancing face, and a grounded electrode on its receding face as seen in Figure 3.1(a). Ignoring the contributions of edge effects at the contact lines and the hot/cold electrode interface, the total capacitive energy in the system is

$$W = \frac{1}{2}c_l L V_{drop}^2 + \frac{1}{2}c_u(x + \frac{L}{2})(V - V_{drop})^2 \qquad (3.12)$$
$$+ \frac{1}{2}c_u(\frac{L}{2} - x)(-V_{drop})^2,$$

where c_u and c_l are the capacitances per unit area of the top and bottom dielectric coatings, V_{drop} is the voltage of the droplet, d is the thickness of the dielectric layers insulating the droplet from the

electrode, and $x = 0$ when the center of the droplet is directly under the left edge of the electrode. Assume $c_u = c_l = c = \epsilon_{lay}/d$ where ϵ_{lay} is the dielectric constant of the insulating layers. Then the droplet voltage is found by minimizing the total energy with respect to V_{drop}, giving a result of [150, 151, 152]

$$V_{drop} = \frac{V}{2}\left(\frac{x}{L} + \frac{1}{2}\right).$$ (3.13)

Adding (3.13) to (3.12) gives the system energy, which can be differentiated by x to give the net force in the horizontal direction,

$$F = \frac{dU}{dx} = \frac{\epsilon_{lay}}{4d}V^2\left(1 - \frac{2x}{L}\right).$$ (3.14)

Note that the total force on the droplet is a function of the droplet position with respect to the electrode. As a result, a droplet under EWOD actuation experiences a cyclic total force as it moves over a periodic array of electrodes. However, one should note that this lumped model breaks down when the droplet interfaces approach the electrode edges, as edge effects play an important role [6]. In such cases one requires direct numerical simulation of the governing equations to obtain the net force, as described in the following sections.

3.3.2 Lumped DEP Force

Let ϵ_{int} and ϵ_{ext} be the dielectric constant of the droplet and the external fluid respectively. Consider the system in Figure 3.1(b) and recall that $x = 0$ when the center of the droplet is directly under the left edge of the electrode in this section. The energy stored in the region where the droplet is under the charged electrode is given by $\frac{1}{2}c_{int}V^2$, where $c_{int} = \frac{\epsilon_{int}}{H}(x - L)V^2$ is the capacitance, per unit area, of this region. The energy stored in the region under the charged electrode where the droplet is **not** present is given by $\frac{1}{2}c_{ext}V^2$, where $c_{ext} = \frac{\epsilon_{ext}}{H}(L_e - x)V^2$ is the capacitance, per unit area, of this second region. Then the net force, per unit area, is given by

$$F = \frac{dU}{dx} = \frac{d}{dx}\left(\frac{1}{2}c_{int}V^2 + \frac{1}{2}c_{ext}V^2\right) = \frac{1}{2}\frac{\epsilon_{int} - \epsilon_{ext}}{H}V^2,$$ (3.15)

where ϵ_{int} is the dielectric constant of the fluid itself and ϵ_{ext} is the dielectric constant of the external fluid. This force is seen to be the difference in capacitive energy between a dielectric-filled
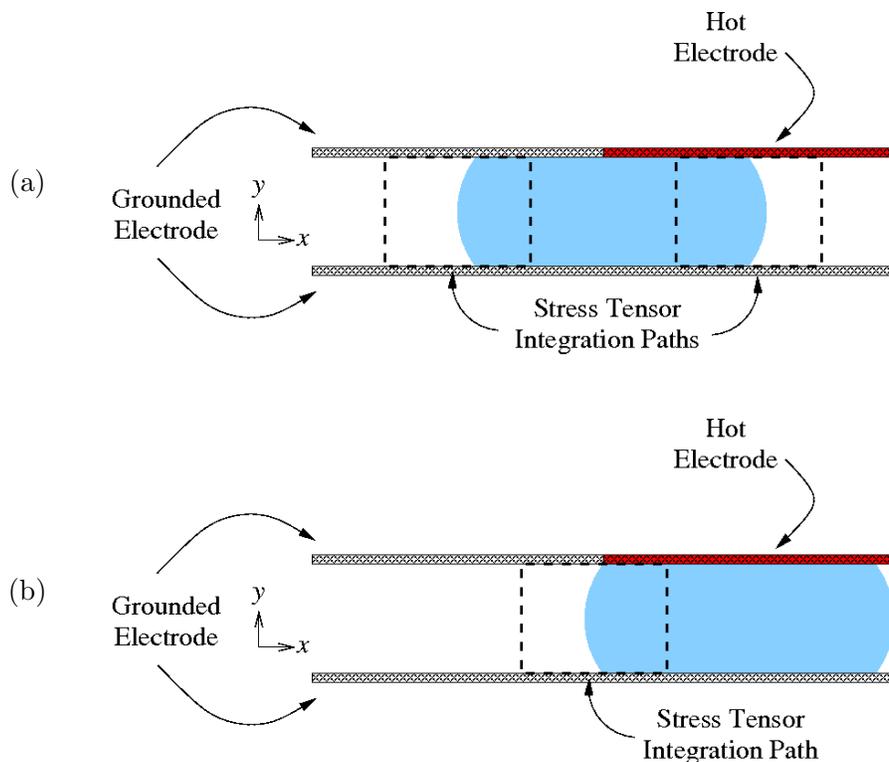
Figure 3.2: Integration paths used when calculating the force with the Maxwell stress tensor when (a) the droplet interface is far away from the electrode interface and (b) when the droplet interface is near the electrode interface.

channel and an empty channel. Again, this model is only valid when the droplet interfaces are well away from any fringing fields. Note that the droplet height, H, appears directly in this expression for F; this factor gives dielectrophoresis a different scaling than EWOD [6, 107]. The velocity of an EWOD droplet depends on H/L, where L is the length of the droplet. In contrast, the velocity of a dielectric droplet depends only on $1/L$, making dielectrophoresis increasingly effective for very small channel sizes.

3.3.2.1   Maxwell Stress Tensor

The above lumped parameter analysis can equivalently be derived using the Maxwell stress tensor with a judicious choice of integration path [77]. However, the stress tensor calculation fails when the droplet interface nears the edge of the electrode.

First consider the integration paths in Figure 3.2(a). Here the droplet interface is well away from the electrode interface. The vertical components of the integration path are assumed to be far enough away from regions of nonuniformity that $\mathbf{E}$ is known analytically. The contributions along the horizontal components of the integration path take the form (when only concerned with force in the horizontal direction)

$$\int T_{12} da = \int \epsilon E_1 E_2 da$$

where the subscripts 1 and 2 refer to the x and y directions respectively. Since the horizontal components lie along the electrodes, $E_1 = 0$ and so these parts of the integration path contribute nothing to the stress tensor integration.

Now consider the integration path in Figure 3.2(b). The droplet interface is now near the electrode interface, and the integration path must lie across the hot/cold electrode interface to ensure that the vertical components of the path are in regions where $\mathbf{E}$ is known analytically. Unfortunately, it can no longer be assumed that $E_1 = 0$ all along the upper horizontal path because of the discontinuity, and so the stress tensor method fails when droplet interfaces are near electrode interfaces. In order to calculate the net force in these regions, numerical methods need to be utilized.

Note that the same problem persists even when a gap is added between the hot and cold electrodes, as one would need to integrate around the corners of the electrode, where the normal direction is ambiguous.

3.4   Force Distribution

In this section we present a detailed analysis of the electric force distribution of a droplet for both EWOD and DEP. The lumped analysis is only valid when the interface is far from the fringing fields. While the total force experienced by a droplet under EWOD and DEP was given in the previous section, a complete solution of the Navier-Stokes equations including electrohydrodynamic effects requires a detailed knowledge of the force distribution, as given below.
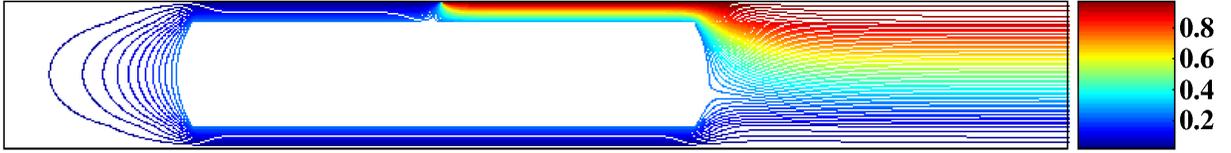
Figure 3.3: Example calculation of the electric equipotentials surrounding a droplet in EWOD configuration, the contact angle is $10°$. In nondimensional units, $h = 1$, $L = 2$, and the length of the computational domain $= 10$. See [6] for more details.

3.4.1   EWOD Force Distribution

Consider a droplet with no net charge while it is electrically isolated from the electrodes. Since no volumetric free charge exists in the solution region, the potential is found by solving Laplace's equation outside of the droplet with the appropriate boundary conditions[2] . The boundary condition on the surface of the conductor is

$$\sigma_f = -\epsilon_{ext} \frac{\partial V}{\partial n}, \tag{3.16}$$

with the entire droplet itself held at a constant potential V_{drop}. The charge distribution and droplet potential are not in general known *a priori*, and must be found as part of the numerical solution.[3]

This is accomplished using a technique in which two initial guesses for the droplet voltage are assumed, the net charge on the surface is calculated by integrating $\sigma_f = -\epsilon_{ext} \frac{\partial V}{\partial n}$ over the boundary, and a subsequent guess for V_{drop} is calculated via the secant method. Convergence is reached when the integration of the surface charge is near enough to zero, specified by the user (a value of 10^{-6} was used in these calculations). See [6] for details.

Laplace's equation is discretized using second-order centered differences and the resulting system is solved iteratively. An example calculation of the electric potential around a conducting droplet is shown in Figure 3.3. The droplet is centered over the voltage step, and the channel is lined with dielectric layers equal thicknesses given by $d = 0.1H$. The contour lines are densely

---

[2] If the droplet is initially charged or if it is short-cut to the electrodes on one side of the channel while isolated from the electrodes on the other side there will be another source of force on the droplet represented locally by $\rho_f \mathbf{E}$. Our approach here can be easily extended to this case as well.

[3] Unless the droplet is isolated only from the electrodes on one side of the channel. In this case the droplet stays at the voltage of the electrodes on the other side of the channel.
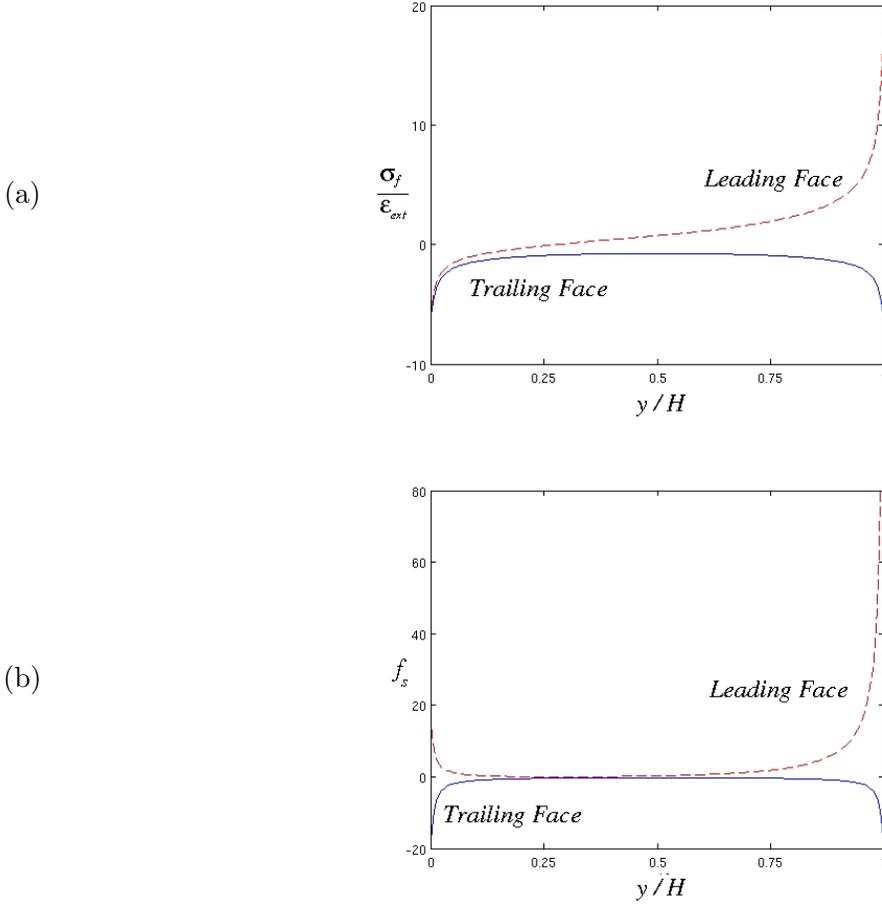
Figure 3.4: Charge distributions (a) and force densities (b) on the leading and trailing interfaces of an EWOD-activated droplet. Contact angle is $0°$. See [6] for more details.

concentrated around the four corners of the droplet, indicating the outward pressure along both faces.

The charge distribution and force densities on the surfaces of a straight-sided centered droplet are shown in Figure 3.4. On the rear interface, the distributions are symmetric about the center line and always of the same sign. On the front of the droplet, the charge changes sign at a location given roughly by

$$y = \frac{V_{drop}}{V} H,$$ (3.17)

and the force density is most strongly peaked near the hot electrode. Note that the force distribu-

tions are localized within a distance of order d near the droplet edges. In many EWOD applications, d is less than 1% of H, and the force is often treated as if it were a point force acting exactly at the contact line [23]. For more details on the numerical force calculation in EWOD including verification, see [6].

3.4.2 DEP Force Distribution

The simple bulk parameter analysis presented earlier for a dielectric medium is only valid for certain locations of the dielectric fluid slug relative to the actuating electrode. When the fluid interface is near a voltage jump, the form of the system energy will be strongly dependent on the exact location of the fluid. Equation (3.15) is therefore expected to be accurate only when the droplet is straddling the hot/cold electrode interface with its own faces reasonably far away. In addition, Equation (3.15) does not address the force on a droplet approaching, but not yet in contact with, a voltage jump. To address these situations, numerical methods must be utilized.

To calculate the desired force density, we must first solve Equation (3.10) numerically to obtain the electric field of the system. The net force upon the droplet is an artifact of the increase in the gradient of the square of the electric field. If the interface is located in a region where the divergence of E^2 is large, the contribution to the net body force of this region is greatly increased. Because of this, it is critically important that the electric field is accurately resolved near the fringing fields located at the edges of the electrodes. To achieve this resolution, a non-uniform grid is applied in the x-direction. Let x and x' denote the horizontal coordinate in the physical and computational planes respectively. We define the relationship between the two planes as

$$x' = x_{\max}\left(A + \frac{1}{\beta_x}\sinh^{-1}\left(\left(\frac{x}{x_0} - 1\right)B\right)\right)$$

where

$$A = \frac{1}{2\beta_x}\log\left(\frac{1 + (e^{\beta_x} - 1)x_0/x_{\max}}{1 + (e^{-\beta_x} - 1)x_0/x_{\max}}\right),$$
$$B = \sinh(\beta_x A),$$
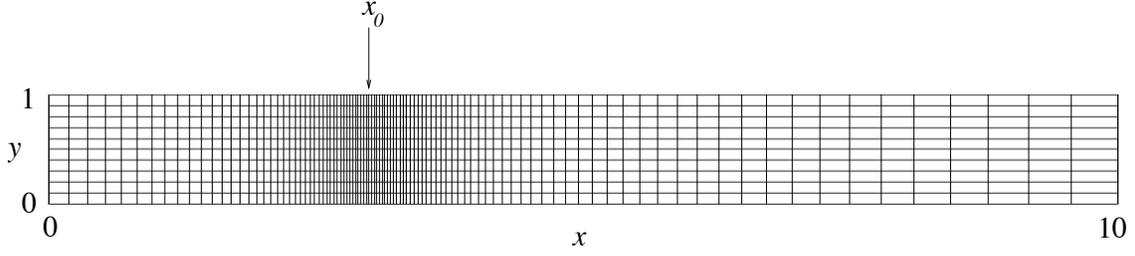
Figure 3.5: Representative grid used in computation, not all computational nodes are shown. x_0 is the point of maximum clustering.

x_{max} is the maximum value in the physical domain, x_0 is where we desire to cluster nodes, and β_x controls the amount of clustering about x_0, where greater values of β_x lead to a higher concentration of points about x_0 [74]. See Figure 3.5 for an schematics of the grid distribution in the computational domain.

Transforming (3.10) to the computational plane, we arrive at

$$\epsilon \left(\frac{\partial^2 V}{\partial x'^2} \left(\frac{\partial x'}{\partial x} \right)^2 + \frac{\partial V}{\partial x'} \frac{\partial^2 x'}{\partial x^2} + \frac{\partial^2 V}{\partial y^2} \right) + \frac{\partial \epsilon}{\partial x'} \frac{\partial V}{\partial x'} \left(\frac{\partial x'}{\partial x} \right)^2 + \frac{\partial \epsilon}{\partial y} \frac{\partial V}{\partial y} = 0, \tag{3.18}$$

where

$$\frac{\partial x'}{\partial x} = \frac{x_{\text{max}} B}{\beta_x x_0 ((x/x_0 - 1)^2 B^2 + 1)^{1/2}},$$
$$\frac{\partial^2 x'}{\partial x^2} = -\frac{x_{\text{max}} B^3 (x/x_0 - 1)}{\beta_x x_0^2 ((x/x_0 - 1)^2 B^2 + 1)^{3/2}}.$$

Each term in Equation (3.18) is discretized using the standard second-order centered finite difference approximation. For a given node (i, j), the discretization of Equation (3.18) yields the following
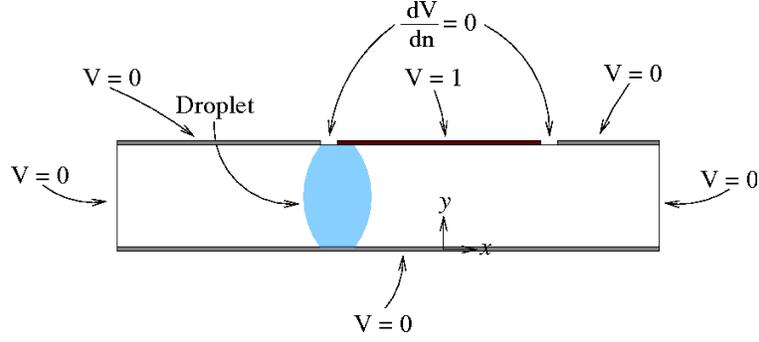
Figure 3.6: Boundary conditions used in numerically solving for the electric potential surrounding a dielectric fluid in DEP configuration.

relationship:

$$\frac{1}{(\Delta y)^2}\left(\epsilon_{i,j} - \frac{\epsilon_{i,j+1} - \epsilon_{i,j-1}}{4}\right)V_{i,j-1}+$$

$$\left(\frac{\epsilon_{i,j}}{(\Delta x)^2}\left(\frac{\partial x'}{\partial x}\right)^2 - \frac{\epsilon_{i,j}}{2\Delta x}\frac{\partial^2 x'}{\partial x^2} - \frac{\epsilon_{i+1,j} - \epsilon_{i-1,j}}{4(\Delta x)^2}\left(\frac{\partial x'}{\partial x}\right)^2\right)V_{i-1,j}-$$

$$2\epsilon_{i,j}\left(\frac{1}{(\Delta x)^2}\left(\frac{\partial x'}{\partial x}\right)^2 + \frac{1}{(\Delta y)^2}\right)V_{i,j}+$$

$$\left(\frac{\epsilon_{i,j}}{(\Delta x)^2}\left(\frac{\partial x'}{\partial x}\right)^2 + \frac{\epsilon_{i,j}}{2\Delta x}\frac{\partial^2 x'}{\partial x^2} + \frac{\epsilon_{i+1,j} - \epsilon_{i-1,j}}{4(\Delta x)^2}\left(\frac{\partial x'}{\partial x}\right)^2\right)V_{i+1,j}+$$

$$\frac{1}{(\Delta y)^2}\left(\epsilon_{i,j} + \frac{\epsilon_{i,j+1} - \epsilon_{i,j-1}}{4}\right)V_{i,j+1} = 0.$$

The domain in Figure 3.6 is discretized using this formula, and the resulting matrix equation is then solved using iterative methods. For the gap between the hot and cold electrodes, the boundary condition $\frac{\partial V}{\partial n} = 0$, where n designates the normal direction, is applied.

Recall that the electrostatic force density is given by (3.6). To find the net force on a fluid slug, it is necessary to integrate (3.6) all along the channel, not just over the volume of droplet itself. This is because the fringing fields at the edge of the charged capacitor exert an inward force on whatever material is present, including the external fluid surrounding the droplet; this force is felt as a pressure on the droplet interfaces, and the flow proceeds such that the material with greatest electric permittivity is drawn into the 'hot' capacitive region. The force on a droplet thus changes sign as it moves over a charged electrode. In addition, the DEP force becomes repulsive for droplets
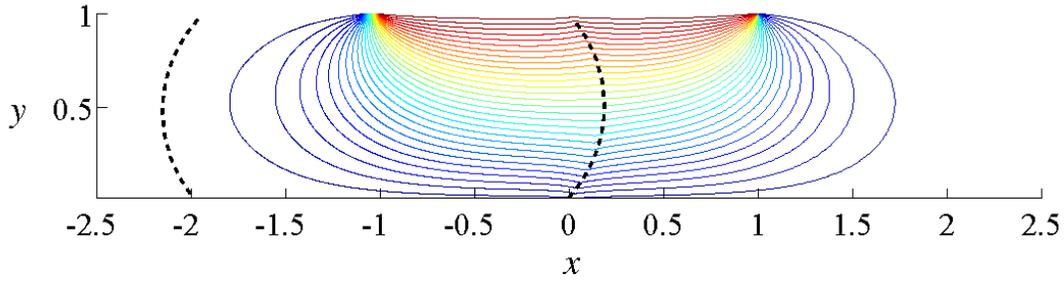
Figure 3.7: Example calculation of the electric potential V for a droplet centered under the left edge of an electrode.

immersed in a fluid of higher dielectric permittivity. Note that, for completeness, in this chapter we consider the full, symmetric case of a droplet moving into and out of a capacitor, resulting in antisymmetric force peaks on the fluid. Therefore, it is recommended that the electrodes in a DEP device be narrower than the droplet's length, and that they be actuated so that a fluid slug is pulled forward with a force always of the same sign.

The nondimensionalized values used in the following computations are $H = 1$, $L = 2$, $L_c = 10$, $L_e = 2$, $\epsilon_{ext} = 1$, and $\epsilon_{int} = 3$, unless otherwise stated. Generally 2001 nodes were used in the horizontal direction with $B_x = 2$ and x_0 located at the leading edge of the droplet, and 51 nodes were used in the vertical direction unless otherwise specified. Figure 3.7 shows an example calculation of the electric potential for a droplet centered underneath the left edge of the electrode. The electric field is found by taking the gradient of this potential and the electrostatic force density is then found by squaring the electric field and again taking the gradient. As a consequence, the force density is very strongly localized in regions where the potential gradient is largest; this occurs near the hot/cold electrode interface.

To ensure that the resolution is high enough to capture the behavior of the net force, the force was calculated as the droplet moved over the edge of the electrode. The convergence of the method for a varied number of points in the horizontal direction is seen in Figure 3.8. Note that capturing the height of the peak requires more resolution than nearby values of the total force.
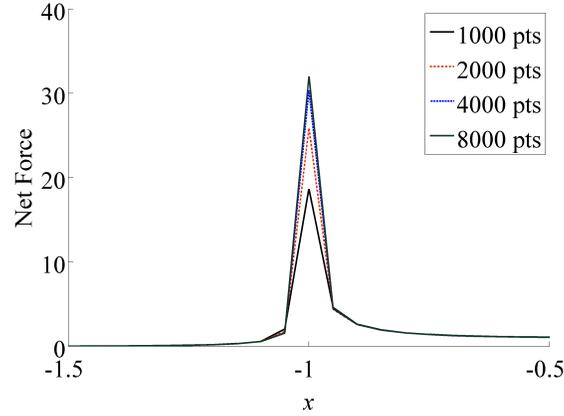
Figure 3.8: Convergence of force calculation as resolution increases. The number of points is in terms of the discretization in the x-direction.

Figure 3.9(a) shows the net force experienced by the droplet as a function of position. The gap width between the hot and cold electrodes is set to 2.5% of H. A sharp spike in force as the droplet approaches the electrode can be seen. Once the leading droplet edge has passed underneath the electrode, the force experienced by the droplet approaches a net nondimensionalized force of 1, which agrees with the theoretical result (3.15) from the lumped force calculation. As the droplet continues to move, it experiences a strong *increase* in force to the right as the leading edge of the droplet approaches the far edge of the electrode. It is at this point that one would want to activate the next electrode in the series to keep the droplet moving to the right. This force is counteracted as the leading edge exits the region covered by the hot electrode and the behavior described above is repeated, but in the opposite direction.

Figures 3.9(b) and 3.9(c) show a similar force plot, but but with wider electrodes so that the active electrode is longer than the entire length of the droplet. In this case, we observe a *decrease* in net force as the leading droplet edge approaches the right edge of the active electrode. Because the electrode is longer than the droplet, the trailing edge of the droplet experiences a negative force as it nears the left edge of the electrode, causing the force to be directed to the left. This occurs before the leading edge of the droplet nears the right edge of the electrode, as in Figure 3.9(a). This situation is not ideal if one hopes to use a sequence of electrodes to transport the droplet to

(a) $L_e = 1.8$ and $L = 2.0$

(b) $L_e = 2.0$ and $L = 2.0$

(c) $L_e = 2.2$ and $L = 2.0$

(d) Setup for force calculations in (a)-(c)
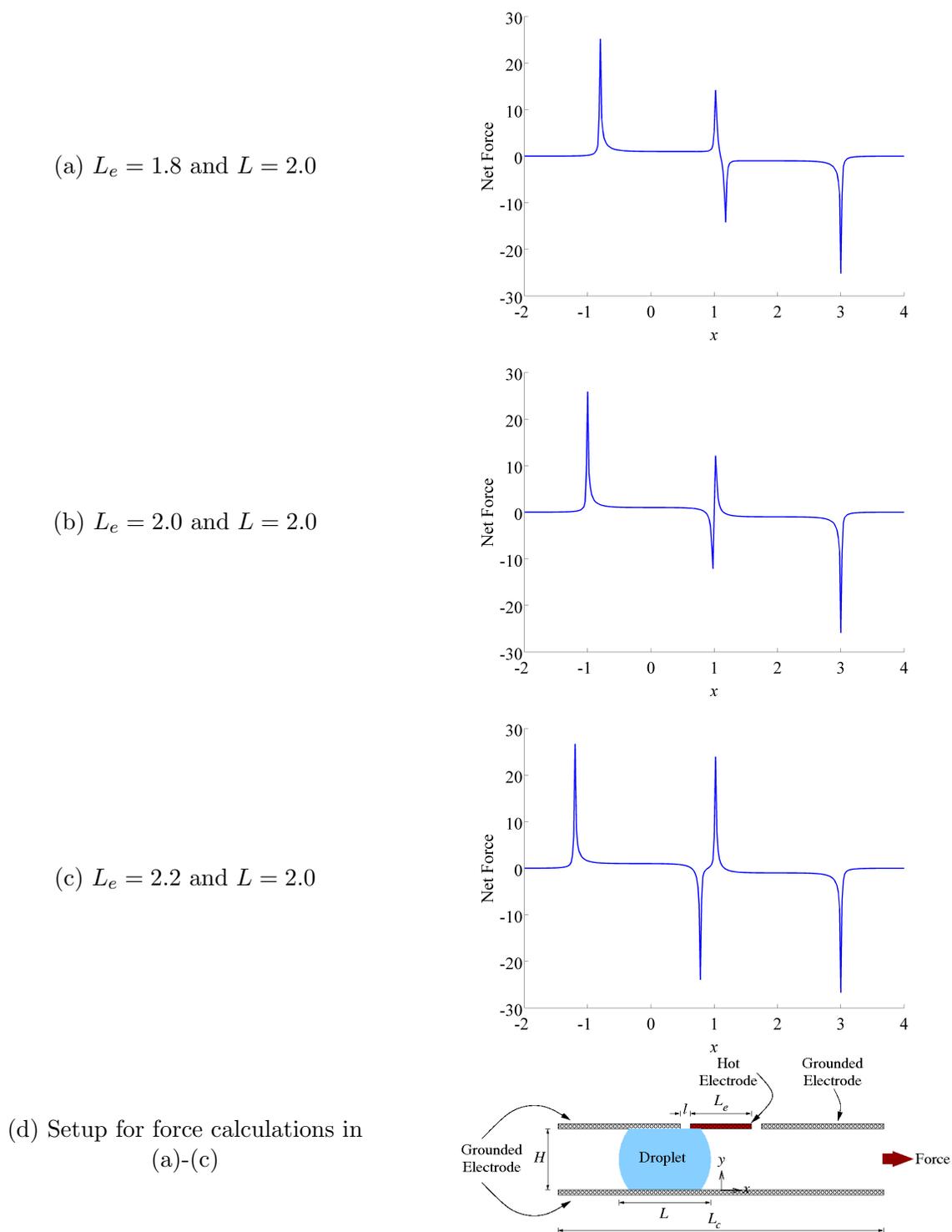
Figure 3.9: The net horizontal force experienced by the droplet as a function of position. Position is given by the location of the center of the droplet with respect to the center of the electrode.

the right. Clearly, a smaller electrode is desirable in applications. This phenomenon demonstrates how the use of narrow electrodes and timely electrode actuation is critical in dielectrophoresis to avoid 'stalling' the droplet in the channel.

The spikes in force seen in Figures 3.9(a), (b), and (c) can be explained by the drastic change in the gradient of the electric field when the droplet edge (and hence the dielectric medium) passes through the fringing field induced by the electrodes. The effect of droplet contact angle can enhance this effect, see Figure 3.10 and Figure 3.11, increasing the net force upon the droplet while it is in this region. This effect will only be seen when the droplet interface is located near the fringing field, otherwise the net force can be obtained by the previously mentioned analytical results. This is a result of the interface physically extending farther into the domain when the contact angle is increased; the nonuniformity of the fringing field gives rise to the increase in force as the interface extends further into the region.

The scaling in dielectrophoresis varies with L, when the droplet interfaces are well removed from the electrode edges. However, when the interfaces are near the edges of the electrode, the force scaling is dependent upon l, the distance between the hot and cold electrodes, as seen in Figure 3.12. In EWOD, a surface force density is distributed over the droplet's front and rear interfaces, and is localized within a length scale equal to the thickness of the dielectric layers lining the channel. Dielectrophoresis differs from EWOD in that a volumetric force density is localized within a radius on the order of the gap between the hot and cold electrodes. While varying the electrode gap width does not affect the net force when the droplet interfaces are sufficiently far from the voltage jump, it does spread the force distribution over a larger area.

Figure 3.13 (a) shows the decrease in total force on the droplet as l increases. As the leading edge of the droplet approaches the electrode, the net force felt by the droplet begins to grow. The rise in force occurs over the same distance as l, resulting in a broadening of the peak as l increases and a reduction in maximum force obtained. Reducing the magnitude of the force by increasing the gap width of the electrodes may be beneficial in applications, as strong electric fields can cause local dielectric breakdown of the fluid.
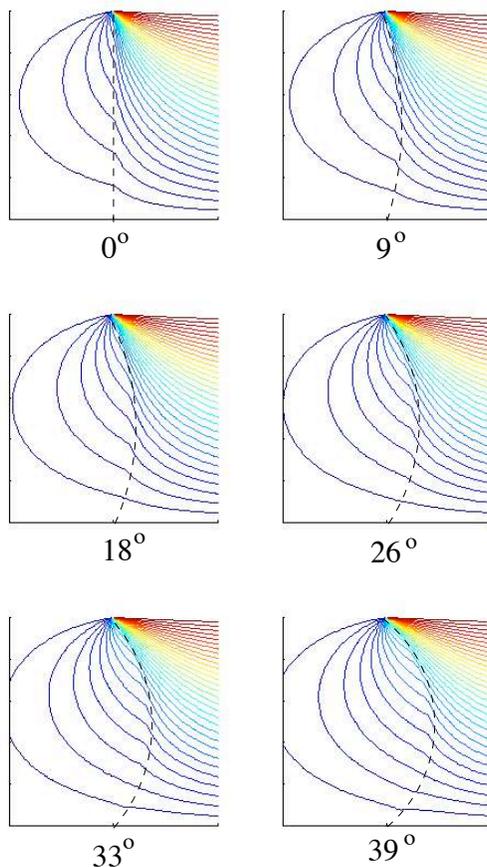
Figure 3.10: Electric potential for various contact angles. The increased gradient from the curvature results in greater net horizontal force.

Scaling these peaks with respect to the gap width demonstrates the linearity of the relationship. We see the same characteristic features of the peaks align when scaled in this manner, as seen in Figure 3.13(b). This is a beneficial relationship for the fabrication of microfluidic devices, as the behavior of droplets with different gap scalings can be extrapolated from known data.

Note that the net forces calculated in this chapter pertain to a solid as well as a liquid; the dynamics of the medium do not play a role in these calculations. To fully integrate the electrostatics into a fluid solver, the force density (3.2) resulting from the electric field can be added as a body
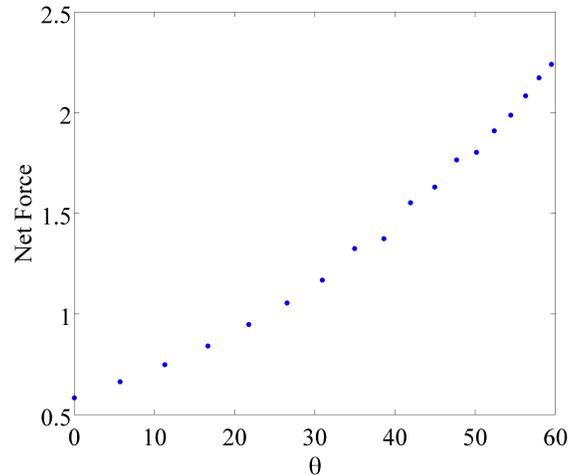
Figure 3.11: Increase in net horizontal force as the contact angle θ increases for a droplet with the leading edge placed near the left edge of the electrode near the fringing field.
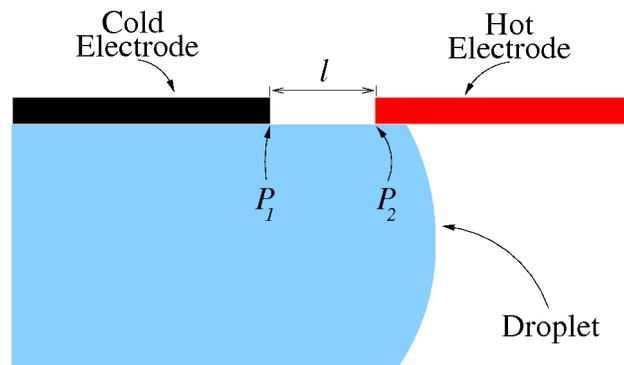
Figure 3.12: DEP configuration. The gap between the adjacent electrodes is l.

force acting upon the fluid. Investigation into fluid solvers coupled with the electrostatic effect are beginning to occur, for example see [136]. Besides the bulk transport of the droplet, it is expected that some interesting dynamics would arise from adding the force distribution into a fluid solver. For instance, consider a droplet centered over the edge of a hot electrode as seen in Figure 3.7. The droplet experiences a net force pulling it into the region of the hot electrode resulting from nonuniformity of the field. The field is particularly nonuniform in the region of the droplet located near the gap separating the hot and cold electrodes (i.e. the points P_1 and P_2 seen in Figure 3.12). This region contains two peaks of extremely large magnitude but opposite sign. The net effect from
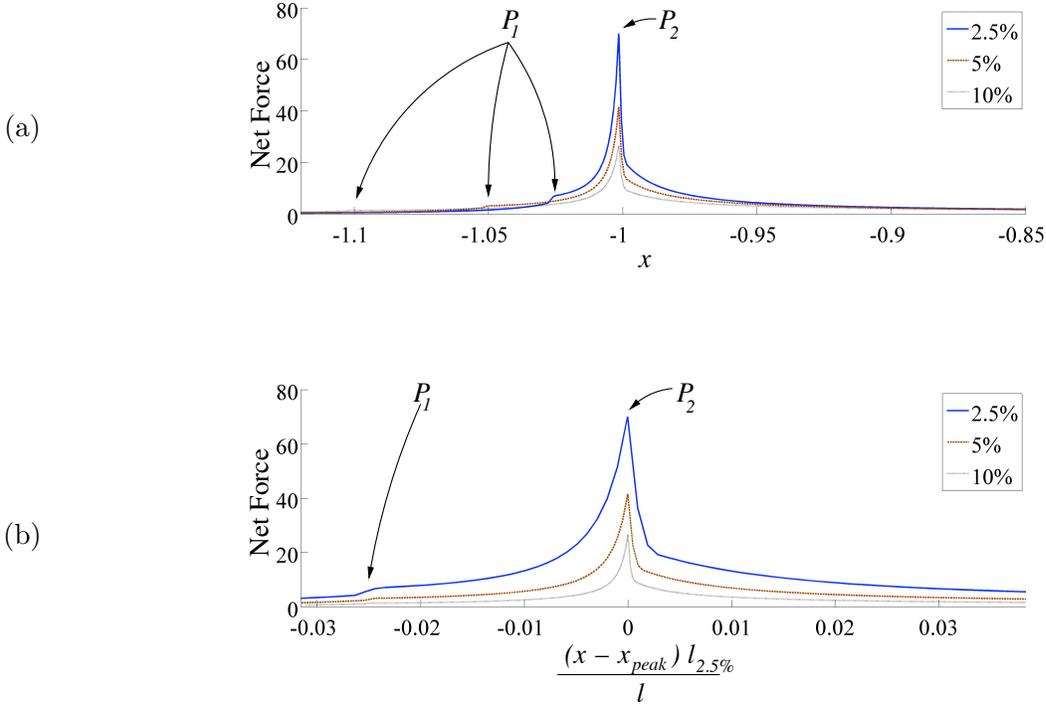
Figure 3.13: Net horizontal force vs x, for various values of l. The percentage shown indicates the width of the gap with respect to H. P_1 and P_2 are the same as shown in Figure 3.12. (a) Force vs position. $x = -1$ when the leading edge of the droplet is flush with the edge of the electrode. (b) Net horizontal force vs nondimensional position (relative to the length of the gap between the electrodes). $l_{2.5\%}$ is the length of the gap for $l = 0.025H$ and x_{peak} is the location of the maximum net force.

this region is canceled out in the integration of the force density, but one would expect the fluid to respond to these peaks locally, creating some local fluid circulation. Furthermore, the net force is only in the horizontal direction, but forces are present in the vertical direction as well which certainly will change the shape of the droplet.

3.5 Conclusions

The two primary methods for droplet transport in digital microfluidics, namely DEP and EWOD have been investigated. EWOD works with conductive fluids while DEP pertains to insulating fluids. In both cases, droplet transport is achieved by sweeping a voltage along a microchannel ahead of the droplet. A review of the Korteweg-Helmholtz and Kelvin force density formulations has been given as well as how these force densities apply to DEP and EWOD.

An energy minimization approach was used to calculate the total force acting upon a droplet under EWOD and DEP actuation. It is seen that the total force for EWOD scales as $1/d$, where d is the dielectric layer thickness, while for DEP the total force scales as $1/H$, where H is the channel height, which leads to a different velocity scaling of H/L and $1/L$ for EWOD and DEP respectively. This indicates that DEP will be increasingly effective for small channel heights.

Investigation of the force distributions for EWOD shows how the density is confined to the surface of the droplet while in DEP it is spread throughout the bulk. This is a critical difference to note when implementing any computational simulation of EWOD and DEP. Two methods were demonstrated for numerically calculating the force distribution for EWOD and DEP. To fully resolve the force when interfaces are located near regions of non-uniformity in the electric field, greater resolution is required. In DEP, it is noted that interface curvature can enhance the net force experienced by the droplet as a result of the fluid interface extending further into the electric field. When a droplet is under DEP actuation, small electrode sizes in comparison to droplet length are preferable, as they keep the net force pointing in the same direction as the droplet moves under the electrode and avoiding any possible stalling of the droplet in the channel. Investigation of the gap width l between electrodes demonstrates how DEP scales with respect to l. As the interface travels through the region represented by l, the net force acting on the droplet increases. A decrease in the maximum force obtained also occurs as l grows. In engineering applications, this parameter could be used to lessen the magnitude of the force if dielectric breakdown of the fluid was a concern.

Chapter 4

A Second-Order Immersed Boundary Method for Applying Dirichlet Boundary Conditions in Elliptic and Parabolic Problems

4.1   Introduction

The numerical solution to partial differential equations on irregular domains is a ubiquitous problem in mathematics, physics, and engineering, and a wide variety of solution techniques are available for such problems. In regards to the Navier-Stokes equations, immersed boundary methods (IBMs) [104, 121] are a popular class of techniques that naturally handle complex moving and deformable boundaries. Usually, immersed boundary methods employ a Cartesian grid to discretize the governing equations on a regular domain in which the immersed boundary is imbedded. The immersed boundary is itself discretized by another set of points which in general, do not conform to the Cartesian mesh. At these points, appropriate forces (e.g. surface tension) are determined to enforce the conditions present along the immersed boundary. Discrete delta functions are used to both interpolate information from the grid to the immersed boundary, as well as to communicate the boundary forces to nearby Cartesian grid points.

The original IBM was developed to simulate blood flow around flexible heart valves. The walls of the heart were assumed to be an elastic boundary, and as such, their influence on the flow was found by applying a boundary force determined by Hooke's law [119, 120]. The IBM has been extended to handle rigid bodies [62, 86, 129] by treating the immersed boundary as an extremely stiff elastic interface. In these approaches, the choice of parameters governing the stiffness of the boundary is done in an *ad hoc* manner, and this treatment can lead to stiffness in the discretized

equations.

To circumvent the issue of selecting parameters that govern the stiffness of the immersed boundary when dealing with flow around irregular rigid bodies, Taira and Colonius developed the Immersed Boundary Projection Method (IBPM) [36, 140]. In the IBPM, the boundary force is determined implicitly by treating the no-slip boundary conditions along the immersed boundary as a Lagrange multiplier, similar to how pressure is used to satisfy the divergence-free condition in projection methods for the incompressible Navier-Stokes equations, e.g. [118]. The IBPM uses operators constructed from discrete delta functions to both interpolate values of the velocity field to the immersed boundary, and to spread the resulting boundary forces back to the Cartesian grid. The IBPM achieves second-order accuracy temporally, but not spatially, when measured in the L_2-norm. This lack of spatial accuracy has been reported in previous methods that utilize discrete delta functions to spread forces located along an immersed boundary to a Cartesian grid, e.g. [142].

In part motivated by the lack of accuracy observed when using discrete delta functions to spread forces located along an immersed boundary to Cartesian grid points, Li and LeVeque developed the Immersed Interface Method (IIM) [90], which provides a second-order accurate technique for discretizing partial differential equations (PDEs) containing irregular interfaces where parameters of the differential equation or even the solution itself are discontinuous or contain singular sources. By carefully modifying both the stencil and the right hand side of the discretized equations, accuracy can be restored at the interface. The IIM has been applied to many problems, see Li and Ito [91] for a comprehensive review and the references within. In general, the IIM requires *a priori* knowledge of the discontinuities present along the immersed interface, and as such, is not immediately applicable to problems where Dirichlet boundary conditions are to be applied along the irregular boundary.

The method presented in this chapter provides a modification of the IBPM applicable to elliptic and parabolic problems containing an immersed boundary upon which a Dirichlet boundary condition is present. By incorporating ideas from both the IBM and the IIM, we can overcome major limitations of both procedures - the lack of spatial accuracy in the IBM and the requirement

on *a priori* knowledge of jump conditions in the IIM. A force along this boundary is found implicitly to enforce the boundary condition using a structure similar to the IBPM. By modifying the IBPM with the methodology employed by the IIM and avoiding the use of discrete delta functions when interpolating to the immersed boundary, second-order spatial accuracy is achieved at all points in the Cartesian mesh.

This chapter proceeds as follows: In Section 4.2, we introduce the governing equations and recast them in a form applicable to the method of this chapter. In Sections 4.3 and 4.4, we review the IIM and IBPM in the context of elliptic and parabolic problems. Section 4.5 describes the modifications used to restore the accuracy of the IBPM at the interface. In Section 4.6, we summarize the new algorithm, and Section 4.7 presents several numerical examples demonstrating the method for elliptic and parabolic problems. In Section 4.9, we give some concluding remarks.

4.2   Preliminaries
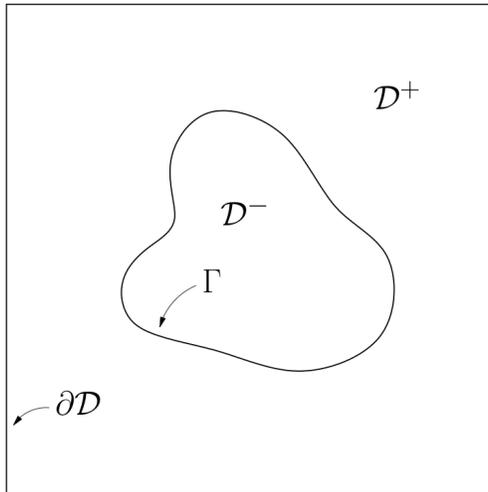
Figure 4.1: A typical domain of interest.

Let $\mathcal{D} = \mathcal{D}^+ \cup \mathcal{D}^- \in \mathbb{R}^2$ be an easily discretized domain (a square, etc.), and let Γ be a closed, smooth contour of codimension 1 contained inside of $\mathcal{D}$, for example see Figure 4.1. Consider the

Poisson equation defined on $\mathcal{D}$,

$$\Delta u = f,$$

$$u = h(\boldsymbol{x}), \quad \boldsymbol{x} \in \Gamma, \tag{4.1}$$

$$u = g(\boldsymbol{x}), \quad \boldsymbol{x} \in \partial\mathcal{D},$$

as well as the heat equation

$$\frac{\partial u}{\partial t} = \Delta u + f,$$

$$u = h(\boldsymbol{x}), \quad \boldsymbol{x} \in \Gamma, \tag{4.2}$$

$$u = g(\boldsymbol{x}), \quad \boldsymbol{x} \in \partial\mathcal{D},$$

where f is smooth and g and h are the Dirichlet boundary conditions on $\partial\mathcal{D}$ and Γ. These functions may be time dependent in the parabolic case, but for notational clarity we will not explicitly include this dependence in the arguments. Letting Γ be parameterized by a vector-valued smooth function $\boldsymbol{X} : [0, S] \to \mathbb{R}^2$, we define the following notation for $u(\boldsymbol{X})$,

$$u^{\pm}(\boldsymbol{X}) = \lim_{\boldsymbol{x}\to\boldsymbol{X}, \, \boldsymbol{x}\in\mathcal{D}^{\pm}} u(\boldsymbol{x}),$$

$$u_{\boldsymbol{n}}^{\pm}(\boldsymbol{X}) = \left(\lim_{\boldsymbol{x}\to\boldsymbol{X}, \, \boldsymbol{x}\in\mathcal{D}^{\pm}} \nabla u(\boldsymbol{x}) \right) \cdot \boldsymbol{n}(\boldsymbol{X}),$$

where $\boldsymbol{n}(\boldsymbol{X})$ is the normal direction of Γ pointing into $\mathcal{D}^+$ at $\boldsymbol{X}$.

With a Dirichlet boundary condition imposed along Γ, the solution u will be continuous across Γ but may experience a jump in its normal derivative (the tangential derivates must be continuous in this case). Letting $\mathcal{L}$ represent the differential operator in (4.1) and (4.2), these equations can equivalently be written as [91]

$$\mathcal{L}u = f + \int_{\Gamma} v(s)\delta(\boldsymbol{x} - \boldsymbol{X}(s)) \, d\Gamma(s), \quad \boldsymbol{x} \in \mathcal{D}, \; \boldsymbol{X} \in \Gamma,$$

$$u = g(\boldsymbol{x}), \quad \boldsymbol{x} \in \partial\mathcal{D}, \tag{4.3}$$

where δ is the Dirac delta function and $v(s) = [u_{\boldsymbol{n}}]_{\boldsymbol{X}(s)\in\Gamma} = u_{\boldsymbol{n}}^+(\boldsymbol{X}(s)) - u_{\boldsymbol{n}}^-(\boldsymbol{X}(s))$ is the jump in the normal derivative of u along Γ at $\boldsymbol{X}(s)$. We can explicitly include the Dirichlet boundary

condition along Γ by writing (4.3) as

$$\mathcal{L}u = f + \int_{\Gamma} v(s)\delta(\boldsymbol{x} - \boldsymbol{X}(s))\, d\Gamma(s), \quad \boldsymbol{x} \in \mathcal{D}, \ \boldsymbol{X} \in \Gamma,$$

$$u(\boldsymbol{X}(s)) = \int_{\mathcal{D}} u(\boldsymbol{x})\delta(\boldsymbol{x} - \boldsymbol{X}(s))dA = h(\boldsymbol{X}(s)), \quad \boldsymbol{x} \in \mathcal{D}, \ \boldsymbol{X} \in \Gamma, \tag{4.4}$$

$$u = g(\boldsymbol{x}), \quad \boldsymbol{x} \in \partial\mathcal{D}.$$

We will now review how to find the numerical solution to (4.1) using the IIM and IBPM by solving (4.3) and (4.4), respectively. The extension to the heat equation is described in Section 4.5. For both methods, we will assume that the domain $\mathcal{D}$ is a square and discretize it using a uniform Cartesian mesh, the grid locations interior to $\partial\mathcal{D}$ are given by $\{\boldsymbol{x}_i\}_{i=1}^{N} = \{(x_i^{(1)}, x_i^{(2)})\}_{i=1}^{N}$. The standard five-point stencil is used to discretize the spatial derivatives. In the absence of Γ, the discrete linear system associated with the discretization of (4.1) can be written as

$$L\tilde{u} = \tilde{f}, \tag{4.5}$$

where L is the discrete Laplacian matrix of size $N \times N$ associated with the five-point stencil, $\tilde{u}$ is a vector of length N containing the approximation to u at the grid points, and $\tilde{f}$ is a vector of length N containing the values of f at the grid points as well as contributions from the boundary conditions defined on $\partial\mathcal{D}$. Such a discretization is of second-order. In what follows, Γ is discretized at K points $\{\boldsymbol{X}(s_k)\}_{k=1}^{K} = \{(X^{(1)}(s_k), X^{(2)}(s_k))\}_{k=1}^{K}$ where $\{s_k\}_{k=1}^{K} \in [0, S]$, and these points are used to enforce the condition along Γ. The selection of these points differs in the IIM and IBPM, and will be described in the following two sections.

4.3   The Immersed Interface Method

In this section, we review how to apply the IIM to find the solution of (4.1). It is assumed that the jump in the normal derivative along Γ is known *a priori*, and so (4.3) can be utilized. The details of this method can be found in the original chapter on the IIM [90], and a summary is given below.

In the presence of Γ, modifications to the linear system (4.5) must be made to incorporate the boundary conditions defined along Γ. The IIM systematically modifies the stencil and right
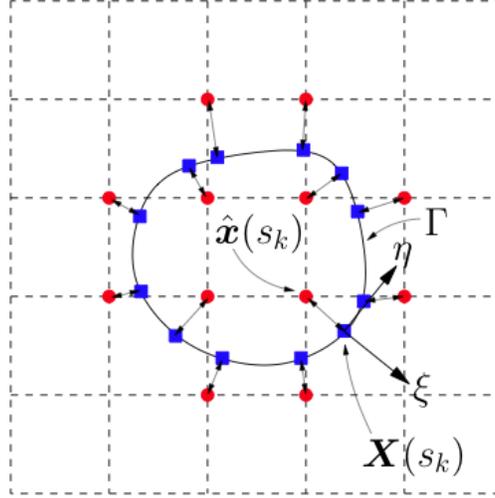
Figure 4.2: Schematic for the IIM. The red dots represent irregular grid points and the blue squares are the points on Γ associated with each irregular grid point. We have drawn the local coordinates (ξ, η) about one of the immersed boundary points $\boldsymbol{X}(s_k)$.

hand side at points near Γ such that second-order accuracy is still achieved. In the particular case where the only condition along Γ is $v(s) = [u_{\boldsymbol{n}}]_{\boldsymbol{X} \in \Gamma}$, corrections are only required to the right hand side of (4.5). That is, L remains the usual $N \times N$ discrete Laplacian matrix generated from the five-point stencil and the discrete linear system takes the form

$$L\tilde{u} = \tilde{f} + H\tilde{v}, \tag{4.6}$$

where $\tilde{v}$ is a vector of length K containing the known values $v(\boldsymbol{X}(s_k))$ and H is the $N \times K$ matrix that when applied to $\tilde{v}$, communicates these values to the uniform Cartesian grid.

The construction of H is as follows. Recall that the locations of the grid points in the uniform Cartesian mesh are given by $\{\boldsymbol{x}_i\}_{i=1}^N = \{(x_i^{(1)}, x_i^{(2)})\}_{i=1}^N$, and let the set $\{\hat{\boldsymbol{x}}_k\}_{k=1}^K \subset \{\boldsymbol{x}_i\}_{i=1}^N$ be the grid points where the finite difference stencil centered at $\boldsymbol{x}_i$ is cut by Γ, for example see Figure 4.2. We will refer to these points as *irregular* grid points. Points on the grid that are not irregular will be referred to as *regular* grid points.

We select the set $\{\boldsymbol{X}(s_k)\}_{k=1}^K$ discretizing Γ such that $\boldsymbol{X}(s_k)$ is near $\hat{\boldsymbol{x}}_i$, usually chosen so that $\boldsymbol{X}(s_k)$ is the closest point on Γ to $\hat{\boldsymbol{x}}_k$. Note that there is a one-to-one correspondence between the irregular grid points $\{\hat{\boldsymbol{x}}_k\}_{k=1}^K$ and the points $\{\boldsymbol{X}(s_k)\}_{k=1}^K$ used to discretize Γ. A coordinate

transformation composed of a shift and a rotation of the form

$$\xi = (x^{(1)} - X^{(1)}(s_k)) \cos \theta + (x^{(2)} - X^{(2)}(s_k)) \sin \theta,$$

$$\eta = -(x^{(1)} - X^{(1)}(s_k)) \sin \theta + (x^{(2)} - X^{(2)}(s_k)) \cos \theta,$$

is then defined at the points $\{\boldsymbol{X}(s_k)\}_{k=1}^K$, where θ is the angle between the $x^{(1)}$-axis and the outward normal at $\boldsymbol{X}(s_k)$. Using this transformation, we can assume that Γ has the local parameterization $\xi = \chi(\eta)$, $\eta = \eta$ near $\boldsymbol{X}(s_k)$. Note that χ'' is the negative of the curvature of Γ at $\boldsymbol{X}(s_k)$.

We can now define the nonzero entries of the $N \times K$ matrix H as

$$H_{i_k,k} = (a_k + \frac{1}{2}(b_k - c_k)\chi''), \quad k = 1, 2, \ldots, K, \tag{4.7}$$

where i_k is the row of H associated with $\hat{\boldsymbol{x}}_k$ and

$$a_k = \sum_{n \in N_k^+} \xi_n \gamma_n, \quad b_k = \sum_{n \in N_k^+} \xi_n^2 \gamma_n, \quad \text{and} \quad c_k = \sum_{n \in N_k^+} \eta_n^2 \gamma_n,$$

where N_k^+ is the index vector marking grid points contained in the stencil centered at $\hat{\boldsymbol{x}}_k$ that are located in $\mathcal{D}^+$, and γ_n is the coefficient of the stencil at that point. Note that (4.7) can be viewed as a discretization of the integral term in (4.3), and that H contains K nonzero entries. This form of H is chosen to restore the truncation error in the discretization of the PDE, see [90] for details.

Given $v(s)$, this provides a simple second-order accurate method for solving (4.3). However, in general the IIM cannot be directly applied to (4.1), unless $v(s)$ known beforehand. As an alternative, we now describe how the IBPM can be applied to the solution of (4.1) when v is unknown.

4.4   The Immersed Boundary Projection Method

In this section, we describe the application of the IBPM to the solution of (4.1). The IBPM was originally developed to simulate incompressible flow around irregular objects that may have some pre-prescribed motion [140]. Given an easily discretized domain containing solid bodies that do not necessarily conform to the grid, a set of points is introduced along the boundaries of the solid

bodies. The IBPM implicitly finds appropriate forces located at these points to apply the desired no-slip boundary conditions along the edges of these bodies. In essence, this implicit force acts as a Lagrange multiplier of the governing equation, enforcing the no-slip constraint at the boundary.

The methodology of the IBPM can easily be extended to other environments. To apply the IBPM to (4.1), we instead consider (4.4). We define $\mathcal{D}$ and construct the matrix L as in the previous sections, and discretize Γ by selecting a set of points $\{\boldsymbol{X}(s_k)\}_{k=1}^{K}$ on Γ. In contrast to the IIM (where points on Γ are determined by the location of the irregular Cartesian grid points), these points are selected to ensure that Γ is sufficiently resolved; usually these points are separated by a distance approximately equal to the Cartesian grid spacing.
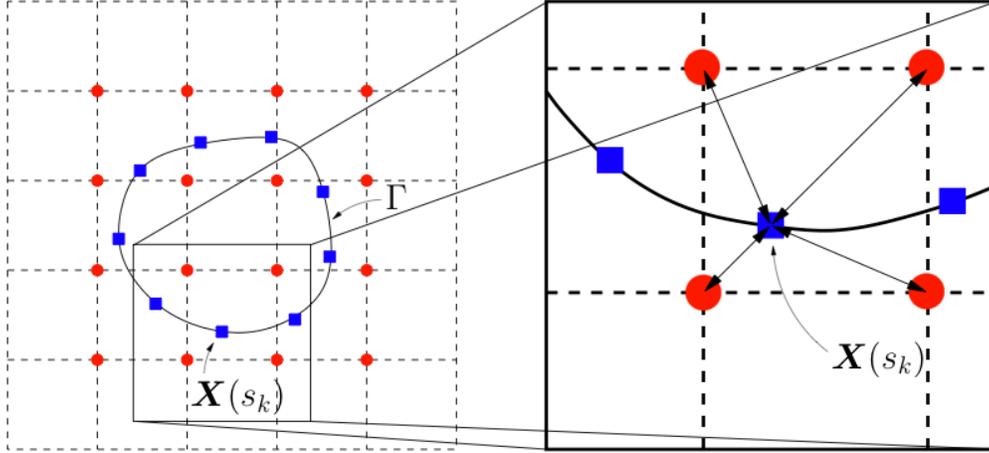
Figure 4.3: Schematic for the IBPM. The red dots represent irregular grid points and the blue squares are the points on Γ. On the left, the box represents the area of a discrete delta function with a radius equal to the grid spacing centered at the point $\boldsymbol{X}(s_k)$. On the right, we see the Cartesian grid points (in red) that are used to interpolate to $\boldsymbol{X}(s_k)$ via the operator E. The action of the operator H is to spread the force at $\boldsymbol{X}(s_k)$ to these same Cartesian grid points.

Discrete representations of the integrals in the first and third equations of (4.4) are utilized, which are given by the $N \times K$ matrix H (defined differently than as in the IIM) and the $K \times N$ matrix E. These are constructed using discrete delta functions, which are described in detail in Sections 4.4.1, 4.4.2, and 4.4.3. The algebraic system of equations associated with the discretization

of (4.4) can now be written as

$$
\begin{bmatrix} L & -H \\ E & 0 \end{bmatrix} \begin{bmatrix} \tilde{u} \\ \tilde{v} \end{bmatrix} = \begin{bmatrix} \tilde{f} \\ \tilde{h} \end{bmatrix},
\tag{4.8}
$$

where $\tilde{u}$ and $\tilde{f}$ are vectors of length N representing the values to u and f at the Cartesian grid points, and $\tilde{v}$ and $\tilde{h}$ are vectors of length K representing the values of v and h at the points discretizing Γ. Note that H is of size $N \times K$ and acts to "smear" the K values of $\tilde{v}$ to the uniform Cartesian grid points while E is of size $K \times N$ and acts to interpolate $\tilde{u}$ from the grid to the points $\{\boldsymbol{X}(s_k)\}_{k=1}^{K}$. See Figure 4.3 for a representative example of the setup in the IBPM.

We now explain the construction of H and E in detail.

4.4.1   Discrete Delta Functions in One Dimension

The properties of discrete delta functions for both interpolation and "smearing" (often referred to as regularizing) has been investigated by many authors [11, 142, 146]. The properties of a discrete delta function in one spatial dimension when interpolating a sufficiently continuous function f to a point X are given by certain *discrete moment conditions* [11, 142].

Definition 4.4.1 Let $X \in \mathbb{R}$ and define $x_j = jh$, where $h > 0$ and $j \in \mathbb{Z}$. Then we say a discrete delta function $\delta_\epsilon(x) \in \mathcal{Q}^q$, if it has compact support in $[-\epsilon, \epsilon]$, where $\epsilon = mh$ for $m > 0$, and satisfies the q discrete moment conditions

$$
M_i(\delta_\epsilon, X, h) = h \sum_{j \in \mathbb{Z}} \delta_\epsilon(x_j - X)(x_j - X)^i = \begin{cases} 1, & i = 0 \\ 0, & 1 \leq i < q \end{cases}.
\tag{4.9}
$$

The accuracy achieved by a discrete delta function interpolating a sufficiently smooth function is given in the following lemma [11, 142].

Lemma 4.4.2 Suppose $\delta_\epsilon(x)$ satisfies $0, 1, \ldots, q$ discrete moment conditions such that $\delta_\epsilon(x) \in \mathcal{Q}^q$, and suppose $f(x) \in C^q(\mathbb{R})$. Then for a point $X \in \mathbb{R}$,

$$
f(X) - h \sum_{j \in \mathbb{Z}} f(x_j)\delta_\epsilon(x_j - X) = O(h^q).
\tag{4.10}
$$

A wide variety of discrete delta functions can be generated using polynomial interpolation. Let $\{x_j\}_{j=0}^n$ be the set of points such that $x_j \in [X - \epsilon, X + \epsilon]$ for $j = 0, \ldots, n$, and define

$$\ell_j(X) = \prod_{i \neq j} \left(\frac{X - x_i}{x_j - x_i} \right), \quad i = 0, 1, \ldots, n. \tag{4.11}$$

Then the Lagrange interpolating polynomial that interpolates a function f at the points $\{x_j\}_{j=0}^n$ to the point X can be written as

$$f(X) \approx \sum_{j=0}^n f(x_j) \ell_j(X). \tag{4.12}$$

From Lemma 4.4.2, we have that

$$f(X) \approx h \sum_{j \in \mathbb{Z}} f(x_j) \delta_\epsilon(x_j - X). \tag{4.13}$$

Comparing (4.12) with (4.13), we see that the value of δ_ϵ at each point x_j is given by

$$\delta_\epsilon(x_j - X) = \begin{cases} \ell_j(X)/h, & x_j \in [X - \epsilon, X + \epsilon] \\ 0, & \text{otherwise.} \end{cases} \tag{4.14}$$

Using this representation, the discrete moment conditions can be written as

$$M_i(\delta_\epsilon, X, h) = \sum_{j=0}^n \ell_j(X)(x_j - X)^i. \tag{4.15}$$

Setting $f(x_j)$ in (4.12) equal to $(x_j - X)^i$ we see that the $M_0, M_1, \ldots, M_n$ moment conditions are satisfied by the discrete delta function given in (4.14). Figure 4.4 shows second, fourth, sixth, and eight order accurate discrete delta functions generated by Lagrange interpolating polynomials. Note that Lemma 4.4.2 assumes that the function and its $1, 2, \ldots, n$ derivatives are continuous at the point X we are interpolating to.

Now suppose the function f we are interpolating is continuous everywhere, but does not have continuous derivatives at X. Interpolating f using the discrete delta functions described previously will not achieve a high level of accuracy, as the Taylor expansion used in the proof of Lemma 4.4.2 will not be valid for higher order derivatives. Accuracy can be restored by building interpolation
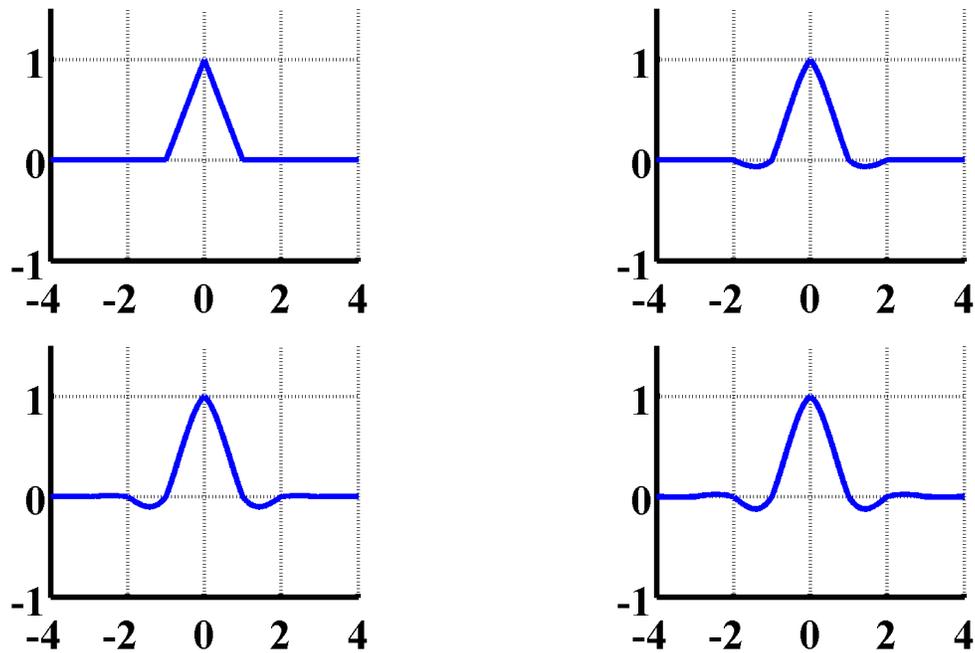
Figure 4.4: From top left to bottom right, second, forth, sixth, and eighth order accurate discrete delta functions for interpolating a smooth function. The grid spacing is $h = 1$ and the x-axis is distance from X.
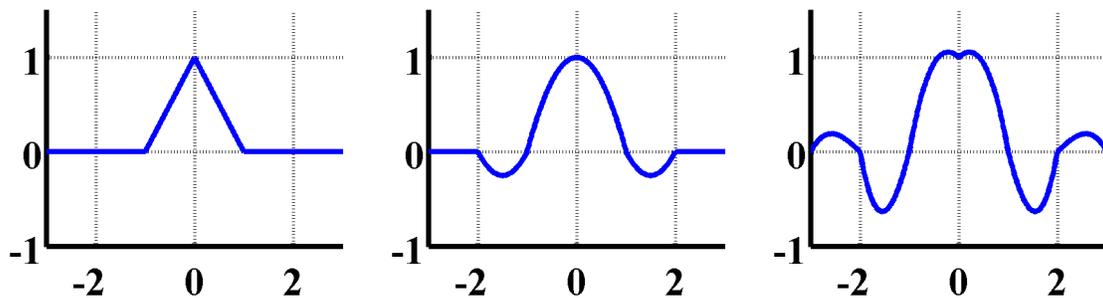
Figure 4.5: From left to right, first, second and third order accurate discrete delta functions for interpolating a continuous function with discontinuous derivatives where the interpolant is evaluated. The grid spacing is $h = 1$ and the x-axis is distance from X.

rules using nodes located on the *same* side of X, and then combining these rules using a weighted average. Then, Lemma 4.4.2 will still be valid so long as we use derivatives on the correct side of the discontinuity. Figure 4.5 shows first, second, and third order accurate regularizations for interpolating functions without smooth derivatives at X.

4.4.2 Discrete Delta Functions in Higher Dimensions

Building discrete delta functions in higher dimensions using their one-dimensional counterparts is a relatively simple task. Suppose we wish to construct a discrete delta function in the space $\mathcal{D} \in \mathbb{R}^d$, where $d > 1$. Let $\boldsymbol{X} = (X^{(1)}, X^{(2)}, \ldots, X^{(d)})$ represent a point on a boundary contained inside of $\mathcal{D}$ and $\boldsymbol{x} = (x^{(1)}, x^{(2)}, \ldots, x^{(d)})$ be a point in $\mathcal{D}$. Following Peskin [121], we define a multidimensional delta function as the product of one-dimensional delta functions,

$$\delta(\boldsymbol{x} - \boldsymbol{X}) = \prod_{i=1}^{d} \delta(x^{(i)} - X^{(i)}). \tag{4.16}$$

The multidimensional discrete delta functions based on Lagrange interpolation follow directly from substitution of the discrete one-dimensional versions described in Section 4.4.1 into (4.16). A multidimensional analog to Lemma 4.4.2 is given by Tornburg [142] when the interpolated function f is sufficiently smooth at $\boldsymbol{X}$. However, as in the one-dimensional case the function that we are interpolating to $\boldsymbol{X}$ will not necessarily have smooth derivatives along the surface that $\boldsymbol{X}$ is located on. In this case, a regularization based on (4.16) combined with any of the one-dimensional versions we described previously will fail to achieve the desired level of accuracy, since they inherently use points on both sides of the surface.

4.4.3 Construction of H and E Using Discrete Delta Functions

We can now describe the construction of the discrete operators H and E used in the IBPM. Assuming that the domain $\mathcal{D} \in \mathbb{R}^2$ is discretized with a uniform Cartesian grid with grid spacing h, recall that $\{\boldsymbol{X}(s_k)\}_{k=1}^{K}$ are the points discretizing Γ and $\{\boldsymbol{x}_i\}_{i=1}^{N}$ are the Cartesian grid points used in the discretization of $\mathcal{D}$. The entries of the kth row of the $K \times N$ interpolation operator E acting on u is then given by

$$E_{k,i} = h^2 \delta_\epsilon(\boldsymbol{x}_i - \boldsymbol{X}(s_k)), \quad i = 1, 2, \ldots, N.$$

Similarly, The entries of the kth column of the $N \times K$ operator H acting on v is given by

$$H_{i,k} = \beta \delta_\epsilon(\boldsymbol{x}_i - \boldsymbol{X}(s_k)), \quad i = 1, 2, \ldots, N.$$

where β is the parameterization factor associated with $d\Gamma(s)$. These operators are simply discrete delta functions centered at the points $\boldsymbol{X}(s_k)$ along the boundary, and note that most of the entries in E and H are zeros, since the discrete delta functions have very small support.

With H and E readily available, solving the linear system (4.8) provides a simple method for computing the solution to (4.1). Unfortunately, both H and E suffer from defects that reduce the overall accuracy of the solution. Suppose we know the value of v that is required along Γ to enforce the boundary condition there. As shown by previous authors [90, 142], smearing this force with a discrete delta function leads to first-order accuracy at Cartesian grid points near Γ (a defect not seen when using (4.7) in passing v to the grid). Similarly, when E is used to interpolate u at grid points to Γ, where u is continuous but has discontinuous derivates, the accuracy is greatly reduced.

In the following section, we correct the deficiencies in E and H to regain second-order accuracy in the solution to (4.8).

4.5 Modification of the IBPM

We now present modifications to the operators E and H that will yield second-order accurate solutions at all grid points in $\mathcal{D}$.

4.5.1 Modifying H

Rather than constructing H from discrete delta functions, we instead use (4.7) as defined by the IIM to transfer v defined on Γ to the grid. As stated before, when v is known *a priori*, this form of H will yield a second-order accurate solution. With this form of H, the points $\{\boldsymbol{X}(s_k)\}_{k=1}^K$ placed on Γ are selected as in the IIM. This differs from the IBPM, where points along Γ are uniformly distributed.

4.5.2 Modifying E

From the construction of H, the locations of the points $\{\boldsymbol{X}(s_k)\}_{k=1}^K$ along Γ have already been selected. What remains is to build an accurate interpolation operator E to interpolate grid

values to these points. Discrete delta functions generated from the product of one-dimensional discrete delta functions will usually yield low accuracy, as they use nodes located on both sides of Γ.

Recall that a point $\boldsymbol{X}(s_k)$ is associated with a single irregular grid point $\hat{\boldsymbol{x}}_k$ that lies either just inside or just outside of Γ. We will construct an interpolation rule that interpolates to $\boldsymbol{X}(s_k)$ using its associated grid point $\hat{\boldsymbol{x}}_k$, and nearby grid points on the *same* side of the boundary. To facilitate the construction of such a rule, we will use radial basis function (RBF) interpolation, which allows for simple and accurate construction of an interpolant using scattered grid points. Other interpolants can be used, but RBF interpolants are often times simpler to construct when dealing with irregular node placements. Further, the complexity in finding the RBF interpolant does not increase as the dimension grows, making it particularly appealing for multidimensional applications.

4.5.2.1 Interpolation using radial basis functions
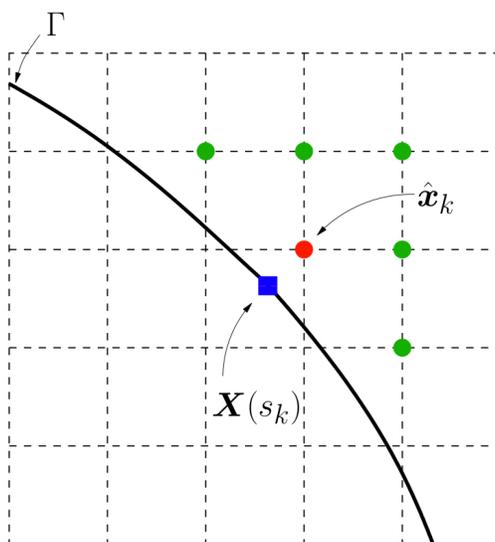
Figure 4.6: An illustration of the Cartesian grid points (red and green circles) used for RBF interpolation to $\boldsymbol{X}(s_k)$ (the blue square). Note that $\hat{\boldsymbol{x}}_k$, the irregular Cartesian grid point that the force centered at $\boldsymbol{X}(s_k)$ (the red circle) is transferred to, is included as a node in defining the interpolant.

Let $\phi(\|\boldsymbol{x}\|)$ be a univariate basis function, where $\|\cdot\|$ is the Euclidean norm. Then the RBF interpolant that interpolates values given at a set of grid nodes $\{\bar{\boldsymbol{x}}_j\}_{j=1}^n \subset \{\boldsymbol{x}_i\}_{i=1}^N$ to a point $\boldsymbol{X}(s_k)$ can be written as

$$s(\boldsymbol{X}) = \sum_{j=1}^n \lambda_j \phi(\|\boldsymbol{X} - \bar{\boldsymbol{x}}_j\|), \tag{4.17}$$

where $\lambda_1, \ldots, \lambda_n$ are expansion coefficients dependent upon the values of the function at $\{\bar{\boldsymbol{x}}_j\}_{j=1}^n$. Three common choices for ϕ in (4.17) that are smooth include the multiquadric (MQ), inverse multiquadric (IQ), and Gaussian (GA),

$$\phi(r) = \sqrt{1 + (\epsilon r)^2} \quad \text{(MQ)},$$

$$\phi(r) = \frac{1}{1 + (\epsilon r)^2} \quad \text{(IQ)},$$

$$\phi(r) = e^{-(\epsilon r)^2} \quad \text{(GA)},$$

where ϵ is a user-defined shape parameter. Smaller and larger values of ϵ correspond to interpolation with flatter and sharply peaked radial functions, respectively. We will focus on GA RBFs constructed in the $\epsilon \to 0$ limit for reasons that will become clear later on. The expansion coefficients can be determined by solving the following linear system,

$$\underbrace{\begin{bmatrix} \phi(\|\bar{\boldsymbol{x}}_1 - \bar{\boldsymbol{x}}_1\|) & \phi(\|\bar{\boldsymbol{x}}_1 - \bar{\boldsymbol{x}}_2\|) & \cdots & \phi(\|\bar{\boldsymbol{x}}_1 - \bar{\boldsymbol{x}}_n\|) \\ \phi(\|\bar{\boldsymbol{x}}_2 - \bar{\boldsymbol{x}}_1\|) & \phi(\|\bar{\boldsymbol{x}}_2 - \bar{\boldsymbol{x}}_2\|) & \cdots & \phi(\|\bar{\boldsymbol{x}}_2 - \bar{\boldsymbol{x}}_n\|) \\ \vdots & \vdots & \ddots & \vdots \\ \phi(\|\bar{\boldsymbol{x}}_n - \bar{\boldsymbol{x}}_1\|) & \phi(\|\bar{\boldsymbol{x}}_n - \bar{\boldsymbol{x}}_2\|) & \cdots & \phi(\|\bar{\boldsymbol{x}}_n - \bar{\boldsymbol{x}}_n\|) \end{bmatrix}}_{R} \underbrace{\begin{pmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_n \end{pmatrix}}_{\lambda} = \underbrace{\begin{pmatrix} u(\bar{\boldsymbol{x}}_1) \\ u(\bar{\boldsymbol{x}}_2) \\ \vdots \\ u(\bar{\boldsymbol{x}}_n) \end{pmatrix}}_{\bar{u}}. \tag{4.18}$$

The existence and uniqueness of the interpolant s has been proven in one-dimension for IQ, GA, and MQ [30, 100, 125]. Generally, small values of ϵ are desirable as they can lead to especially accurate interpolants. Letting h be the typical distance between the data nodes, then as h and ϵ approach zero, the entries of R approach unity and R becomes increasingly ill-conditioned, limiting the values of ϵ that can be used in solving (4.18). This ill-conditioning can be overcome through the use of the Contour-Padé algorithm [54] or the RBF-QR algorithm [53]. Due to the availability of these stable algorithms and the increased accuracy of the interpolate in the flat limit, the $\epsilon \to 0$

limit is used in the numerical examples presented in this chapter. Note that in the limit as $\epsilon \to 0$, RBF interpolation (under mild restrictions) is equivalent to Lagrange interpolation [44].

The choice of using Gaussians in constructing the RBF interpolant is based on several properties. Several features of GA RBFs have been proven or conjectured [55]. Notably, GA interpolants will not diverge as $\epsilon \to 0$ for data placed on a finite rectangular lattice in d dimensions. Further, GA interpolants are unique in that they can be written as a product, $\phi(r) = \sum_{i=1}^{d} \phi(\tilde{x}^{(i)})$, where $r^2 = \sum_{i=1}^{d} (\tilde{x}^{(i)})^2$, similar to the product rule interpolation given by (4.16).

To construct the interpolation operator E using RBF interpolation, select n Cartesian grid points, which include the grid point $\hat{\boldsymbol{x}}_k$ and the rest given by nearby grid points that are located on the *same* side of Γ. We designate this set of points $\{\tilde{\boldsymbol{x}}_i\}_{i=1}^{n}$, and Figure 4.6 shows an example of the grid points used for interpolation. We need E to operate on the vector $\tilde{u}$ to find the interpolated values $\{u(\boldsymbol{X}(s_k))\}_{k=1}^{K}$.

Letting $i_1, i_2, \ldots, i_n$ be the indices of the locations of the values of $\bar{u}$ in the vector $\tilde{u}$, we see that by combining (4.17) with $\lambda = R^{-1} \bar{u}$, the nonzero entries in the kth row of E are given by

$$[E_{k,i_1} \quad E_{k,i_2} \quad \cdots \quad E_{k,i_n}] = [\phi(\|\boldsymbol{X} - \bar{\boldsymbol{x}}_1\|) \quad \phi(\|\boldsymbol{X} - \bar{\boldsymbol{x}}_2\|) \quad \cdots \quad \phi(\|\boldsymbol{X} - \bar{\boldsymbol{x}}_n\|)] R^{-1}. \quad (4.19)$$

These values can be found by directly inverting R, or if ϵ is small, using the Contour-Padè or RBF-QR algorithm.

4.5.3 Application to the Heat Equation

So far, we have presented an algorithm for solving Poisson's equation with an irregular boundary imbedded in a uniform domain. Without modification, it is equally applicable to the heat equation; that is, setting the differential operator in (4.1) to $\partial/\partial t - \Delta$ and discretizing using the Crank-Nicolson method, the same technique described here can be used. This problem is investigated in Section 4.7.

4.6   Summary of Algorithm

We now know the modifications of the original IBPM necessary to extend it to the numerical solution of (4.1) by means of (4.4) in a second-order accurate manner. To summarize:

(1) Form the $N \times N$ matrix L and its associated right hand side $\tilde{f}$ using the standard five-point stencil.

(2) Following the methodology of the IIM, identify the irregular grid points $\{\hat{\boldsymbol{x}}_k\}_{k=1}^K$ where the stencil centered at $\hat{\boldsymbol{x}}_k$ is cut by Γ. From this, the points $\{\boldsymbol{X}(s_k)\}_{k=1}^K$ on Γ can then be found and the $N \times K$ matrix H can be constructed using (4.7).

(3) Form the $K \times N$ matrix E that interpolates values from the Cartesian grid to the points $\{\boldsymbol{X}(s_k)\}_{k=1}^K$ on Γ, using RBF interpolation as given by (4.19). Recall that the grid points used to interpolate to each point $\boldsymbol{X}(s_k)$ are selected from the same side of Γ.

(4) With L, H, and E available, form the linear system (4.8) and solve to obtain $\tilde{u}$.

Although not explored in this chapter, efficient means should be utilized to solve (4.8). By design, the IBPM constructed the off-diagonal blocks in (4.8) as transposes of each other, preserving the symmetry of the system and allowing for the use of fast solvers. The modifications we have presented no longer preserve this property. Nevertheless, the system can be decomposed using Schur compliments yielding a projection-like procedure for solving (4.8) as in the IBPM, allowing for one to use fast solvers (e.g. multigrid) to first solve a Poisson-like problem and then solve a smaller system of size $K \times K$ directly to arrive at the solution.

4.7   Numerical Results

We have implemented the algorithm summarized in Section 4.6 and present several numerical examples pertaining to elliptic and parabolic partial differential equations. All spacial derivatives are computed using the second-order five-point stencil on a uniform Cartesian grid. For the

parabolic problems, the Crank-Nicolson discretization is used in time. The RBF interpolants were formed with the user-defined shape parameter ϵ set equal to 0.

4.7.1   Elliptic Problems

4.7.1.1   Example 1

Let $\mathcal{D} = [-1, 1] \times [-1, 1]$, let Γ be a circle of radius $1/2$ centered at the origin, and consider the solution to Laplace's equation,

$$\Delta u = 0. \tag{4.20}$$

The exact solution is given by

$$u(\boldsymbol{x}) = \begin{cases} 1, & |\boldsymbol{x}| \leq 1/2 \\ 1 - \log(2|\boldsymbol{x}|)/2, & |\boldsymbol{x}| \geq 1/2, \end{cases}$$

and from this, the boundary conditions on $\partial\mathcal{D}$ and Γ are determined. For this problem, note that $[u_{\boldsymbol{n}}]_\Gamma = 2$, and so we can compare the algorithm presented in this chapter directly with the IIM.
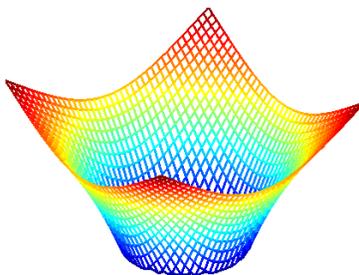
Figure 4.7: The exact solution in Example 1.

To assess the accuracy of the new method, the solution to (4.20) was computed on increasingly finer grids and the ratio between the errors was calculated. A ratio of 4 corresponds to second-order convergence.

Table 4.1 displays numerical results for the new algorithm and the original IIM applied to the solution of (4.20). We have freedom in choosing how many points to use in interpolating to the boundary. In building the interpolation matrix E, we have chosen to use the grid points contained

(a)   (b)   (c)
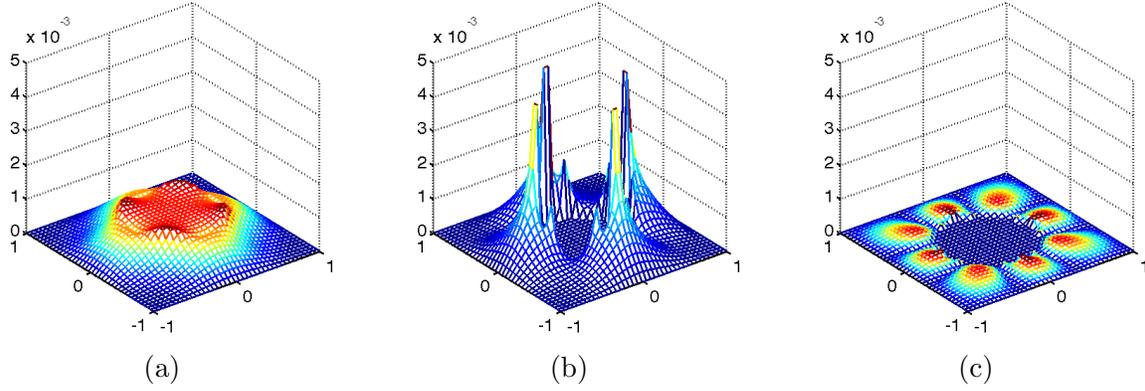
Figure 4.8: Error in the numerical solution to Example 1 on a 40×40 grid. (a). The original IIM. (b). Current method with $E = E_{2h}$. (c). Current method with $E = E_{4h}$.

	IIM		$E = E_{2h}$		$E = E_{4h}$	
h	$\|u - \tilde{u}\|_\infty$	Ratio	$\|u - \tilde{u}\|_\infty$	Ratio	$\|u - \tilde{u}\|_\infty$	Ratio
1.0000e-01	4.3350e-03	-	1.6255e-02	-	2.6321e-03	-
5.0000e-02	1.1096e-03	3.907	4.5153e-03	3.600	2.2293e-04	11.807
2.5000e-02	2.6564e-04	4.178	1.3215e-03	3.417	1.0061e-04	2.216
1.2500e-02	6.4824e-05	4.098	3.6191e-04	3.652	1.5614e-05	6.444
6.2500e-03	1.6968e-05	3.820	1.0083e-04	3.589	3.1856e-06	4.901
3.1250e-03	4.2166e-06	4.024	2.7896e-05	3.615	7.9941e-07	3.985
1.5625e-03	1.0403e-06	4.053	7.4677e-06	3.736	2.0035e-07	3.990

Table 4.1: Errors and convergence for the solution to (4.20) using the original IIM and the method described in this chapter, with the grid points used to interpolate to the boundary of various size ($E = E_{2h}$, $E = E_{4h}$).

in a square centered about each grid point $\hat{\boldsymbol{x}}_k$ that is near the boundary. The points selected from this square are on the same side of Γ. We have tested for two lengths of this square, either $2h$ (see an example in Figure 4.6) or $4h$. We refer to these cases as $E = E_{2h}$ and $E = E_{4h}$.

All three cases presented in Table 4.1 display second-order convergence in the infinity norm. Note that when we set $E = E_{2h}$ the algorithm performs worse than the IIM, but with $E = E_{4h}$, the error is actually less than in the IIM. This phenomenon can be explained by looking at a plot of the error in the solution as given in Figure 4.8. We see that in the $E = E_{2h}$ case, spikes in the error are present along Γ, while in the case $E = E_{4h}$, these spikes are greatly reduced. Evidently,

the accuracy of the interpolant used plays a key role in determining the overall accuracy of the solution; increasing the number of grid points used in interpolation leads to an increase in accuracy.

Another interesting aspect of Figure 4.8 is that the numerical solution obtained by the new method interior to Γ is exact (to machine precision). Since the analytic solution is constant in this region of the domain, the interpolation from grid points interior to Γ is exact. Further, the finite-difference approximation centered at grid points interior to Γ whose stencil is not cut by Γ is also exact, so it must be the case that the values of $\tilde{v}$ determined by the algorithm reduce the truncation error to zero for the grid points that are interior to Γ whose stencil is cut by Γ. Note that this is not the case for the IIM, as seen in Figure 4.8(a).

4.7.1.2   Example 2

We now consider an example for solving Laplace's equation interior to an irregular boundary. Let Γ be the starfish shaped boundary given by the parameterization

$$\boldsymbol{x} = \left(\frac{\cos s(9\pi + 7\sin 5s)}{63}, \frac{\sin s(9\pi + 7\sin 5s)}{63} \right), \quad s \in [0, 2\pi].$$

and set the exact solution given interior to Γ as

$$u(\boldsymbol{x}) = \log(\sqrt{(x^{(1)} + 0.75)^2 + (x^{(2)} - 1)^2}) - \log(\sqrt{(x^{(1)} - 0.8)^2 + (x^{(2)} + 0.9)^2}))$$

We imbed Γ in the square $\mathcal{D} = [-1, 1] \times [-1, 1]$ and set the boundary condition on $\mathcal{D}$ to zero. The boundary conditions along Γ are given by the exact solution. We then proceed to calculate the numerical solution to this problem on $\mathcal{D}$, using the new method.

Table 4.2 gives the errors in the infinity norm calculated at grid points interior to Γ (the closed form of the solution outside of Γ is not readily available), calculated for both $E = E_{2h}$ and $E = E_{4h}$. The convergence of the method in the $E = E_{2h}$ case appears not to be of second-order, and in Figure 4.9(b), we can see that that spikes in error are generated near Γ. Convergence is restored in the case $E = E_{4h}$, and again we see that increasing the accuracy of interpolation increases the accuracy of the numerical solution.
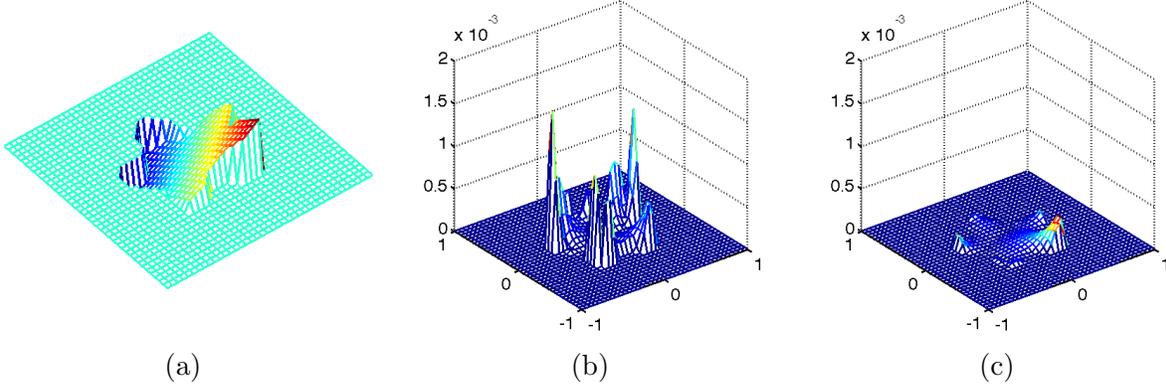
Figure 4.9: Results for Example 2 on a 40×40 grid. The values outside of Γ have artificially been set to zero. (a). The exact solution. (b). Error of current method with $E = E_{2h}$. (c). Error of current method with $E = E_{4h}$.

	$E = E_{2h}$		$E = E_{4h}$	
h	$\|u - \tilde{u}\|_\infty$	Ratio	$\|u - \tilde{u}\|_\infty$	Ratio
1.0000e-01	1.0225e-02	-	4.4562e-02	-
5.0000e-02	1.6268e-03	6.285	3.2765e-04	136.005
2.5000e-02	3.7317e-04	4.360	1.4711e-05	22.272
1.2500e-02	9.7871e-05	3.813	4.3026e-06	3.419
6.2500e-03	2.1774e-05	4.495	8.1993e-07	5.248
3.1250e-03	6.5866e-06	3.306	1.7518e-07	4.681
1.5625e-03	2.1282e-06	3.095	4.3244e-08	4.051

Table 4.2: Numerical results for Example 2 using the current method with $E = E_{2h}$ and $E = E_{4h}$.

4.7.1.3   Example 3

We repeat Example 2 but with the exact solution interior to Γ given by

$$u(\boldsymbol{x}) = (x^{(1)})^2 - (x^{(2)})^2.$$

Note that this solution is quadratic function, and that the truncation error of the five-point stencil is zero at all points whose stencil is not cut by Γ. Using the new algorithm with $E = E_{4h}$, the calculated numerical solution is accurate to within machine precision at all interior grid points, but this is not the case for $E = E_{2h}$. As noted before, RBF interpolants limit to polynomial interpolants as the shape parameter $\epsilon \to 0$. Evidently, the interpolant for $E = E_{4h}$ is of high enough order to

be exact for a quadratic function, while $E = E_{2h}$ is not.

4.7.2   Parabolic Problems

4.7.2.1   Example 4
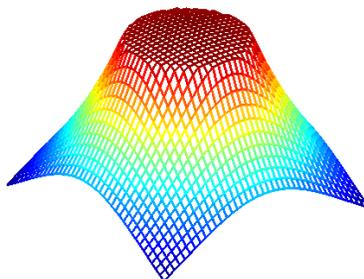
Figure 4.10: The exact solution in Example 4.

We now apply the method presented in this chapter to the heat equation,

$$\frac{\partial u}{\partial t} = \Delta u. \tag{4.21}$$

Again, we set $\mathcal{D} = [-1, 1] \times [-1, 1]$ and determine the initial and boundary conditions from the exact solution

$$u(x) = \begin{cases} J_0(|\boldsymbol{x}|)e^{-t}, & |\boldsymbol{x}| \leq 1/2 \\ J_0(1/2)(Y_0(|\boldsymbol{x}|)/Y_0(1/2))e^{-t}, & |\boldsymbol{x}| \geq 1/2, \end{cases}$$

where J_0 and Y_0 are Bessel functions of the first and second kind of order 0. In this case, $[u_{\boldsymbol{n}}]_\Gamma = e^{-t}(Y_0'(1/2)J_0(1/2)/Y_0(1/2) - J_0'(1/2))$ and so a comparison with the IIM can be made. The Crank-Nicolson scheme is used to discretize in time. Starting with $t = 0$, the numerical solution is calculated at $t = 0.1$ and the time step is made sufficiently small to ensure that the errors in the calculated solution are dominated by the spatial discretization.

The results here are much the same as in Example 1. An increase in the accuracy of the interpolation operator leads to an increase in accuracy. We also achieve higher accuracy when $E = E_{4h}$ than when using the IIM.
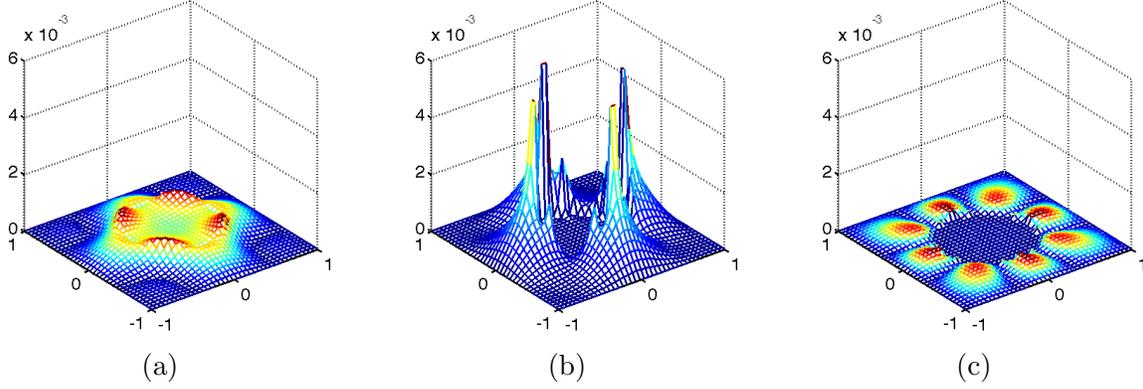
Figure 4.11: Error in the numerical solution to Example 4 on a 40×40 grid. (a). The original IIM. (b). Current method with $E = E_{2h}$. (c). Current method with $E = E_{4h}$.

	IIM		$E = E_{2h}$		$E = E_{4h}$	
h	$\|u - \tilde{u}\|_\infty$	Ratio	$\|u - \tilde{u}\|_\infty$	Ratio	$\|u - \tilde{u}\|_\infty$	Ratio
1.0000e-01	3.4281e-03	-	1.9970e-02	-	3.2870e-03	-
5.0000e-02	8.9818e-04	3.817	5.4614e-03	3.657	2.8553e-04	11.512
2.5000e-02	2.2190e-04	4.048	1.5869e-03	3.441	1.1395e-04	2.506
1.2500e-02	5.4819e-05	4.048	4.3306e-04	3.664	1.7717e-05	6.432
6.2500e-03	1.4245e-05	3.848	1.2046e-04	3.595	4.0035e-06	4.425
3.1250e-03	3.5270e-06	4.039	3.3303e-05	3.617	1.0031e-06	3.991
1.5625e-03	8.7157e-07	4.047	8.9114e-06	3.737	2.5121e-07	3.993

Table 4.3: Errors and convergence for the solution to (4.21) for Example 4 using the original IIM and the current method.

4.7.2.2 Example 5

We demonstrate that we can imbed irregular boundaries inside the regular domain and solve for the solution inside the irregular boundary for (4.21). We set Γ as in Example 2 and set the boundary conditions along the irregular boundary and initial condition interior to the domain from the exact solution

$$u(x) = \frac{e^{-((x-0.05)^2 + (y-0.15)^2)/(4(t+0.01))}}{4\pi(t+0.01)}.$$

Outside the irregular boundary, the initial condition is again set to 0 as well as along $\partial\mathcal{D}$. We only measure the error interior to the irregular boundary at time $t = 0.1$.

In Table 4.4, we again see how increasing the accuracy of the interpolation operator E
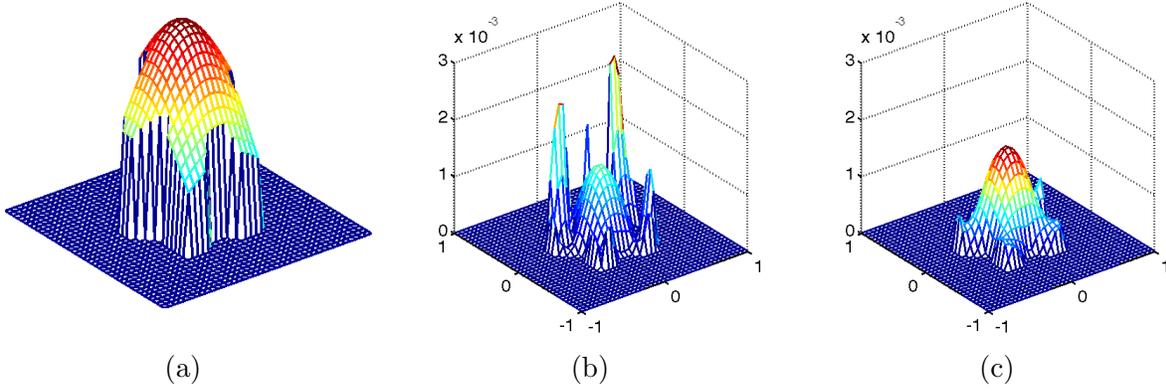
Figure 4.12: Results for Example 5 on a 40×40 grid. The values outside of Γ have artificially been set to zero. (a). The exact solution. (b). Error for the current method with $E = E_{2h}$. (c). Error for the current method with $E = E_{4h}$.

	$E = E_{2h}$		$E = E_{4h}$	
h	$\|u - \tilde{u}\|_\infty$	Ratio	$\|u - \tilde{u}\|_\infty$	Ratio
1.0000e-01	1.3643e-02	-	1.3302e-02	-
5.0000e-02	2.9340e-03	4.650	1.6373e-03	8.124
2.5000e-02	1.2812e-03	2.290	3.9582e-04	4.137
1.2500e-02	1.5227e-04	8.414	9.3478e-05	4.234
6.2500e-03	5.2229e-05	2.915	2.2877e-05	4.086
3.1250e-03	2.2515e-05	2.320	5.4312e-06	4.212
1.5625e-03	5.1017e-06	4.413	1.1049e-06	4.916

Table 4.4: Numerical results for Example 5 using the method presented in this chapter with $E = E_{2h}$ and $E = E_{4h}$.

increases the accuracy of the solution, just as in the previous examples.

4.8   On Extension to the Navier-Stokes Equations

Consider the incompressible Navier-Stokes equations in two dimensions,

$$\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} = -\nabla p + \frac{1}{\text{Re}} \nabla^2 \mathbf{u} + \int_\Gamma \mathbf{f}(s) \delta(\boldsymbol{x} - \boldsymbol{X}(s)) \, d\Gamma(s),$$

$$\nabla \cdot \mathbf{u} = 0, \tag{4.22}$$

$$\mathbf{u}(\boldsymbol{X}(s)) = \int_\mathcal{D} \mathbf{u}(\boldsymbol{x}) \delta(\boldsymbol{x} - \boldsymbol{X}(s)) dA = \mathbf{u}_0(\boldsymbol{X}(s)), \quad \boldsymbol{x} \in \mathcal{D}, \ \boldsymbol{X} \in \Gamma,$$

where the last equation in (4.22) represents the velocity boundary condition on the irregular boundary, and $\mathbf{f}$ is an unknown force that enforces the velocity boundary condition. In the original immersed boundary projection method [140], this system was discretized with a staggered grid formulation using the implicit Crank-Nicolson integration for the viscous terms and the explicit second- order Adams-Bashforth scheme for the convective terms. The delta functions were discretized using the discrete delta function methodology, leading to a linear system of the form

$$
\begin{bmatrix} A & G & -H \\ D & 0 & 0 \\ E & 0 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ p \\ \mathbf{f} \end{bmatrix} = \begin{bmatrix} r \\ 0 \\ \mathbf{u}_0 \end{bmatrix},
$$

where A is the implicit components of the discretization, G is the discrete gradient operator, D is the discrete divergence operator, and r is the explicit components of the discretization. This approach suffers from the same drawbacks as discussed previously.

To provide an accurate method for this problem, we can replace the interpolation operator E as described previously. To modify H, we can use the immersed interface approach for (4.22) as described in [92]. However, the correction terms described in [92] not only depend on $\mathbf{f}$, but also on its derivative with respect to its position along the boundary. This is the primary obstacle in forming such a method, as the new linear system corresponding to this approach takes a more complicated form in comparison to (4.22) to reflect this change.

4.9 Conclusions

We have presented a modification to the original immersed boundary projection method that increases its accuracy to second-order at all points in the domain. We have discussed several aspects of discrete delta functions used in the original immersed boundary projection method (IBPM) and the difficulties in using them in dimensions greater than 1 while retaining second-order accuracy. The method described here modifies the original IBPM by using the methodology of the immersed interface method (IIM) in constructing the operator that communicates the boundary force to the Cartesian grid, and utilizes an interpolation operator that retains its accuracy by using nodes only

contained on the same side of the immersed boundary. With these modifications, the algorithm successfully applies Dirichlet boundary conditions present along irregular boundaries while allowing for the use of a standard finite difference discretization and a uniform Cartesian grid. The method preserves second-order accuracy at all grid points and is simple to implement.

The algorithm has been applied to several elliptic and parabolic partial differential equations. We have observed second-order convergence in all the tested cases, and have shown how the method can be utilized to find solutions to problems on nonuniform domains by first imbedding the irregular domain inside a regular domain, discretizing using a standard Cartesian grid, and employing the techniques described by this chapter.

The expectation is that the methodology presented here can be extended to other differential equations where immersed boundaries are present, providing a technique for sharpening the resolution at the irregular boundaries. As Chapters 2 and 3 clearly demonstrate, a unified technique for computing electrostatic forces, heat transfer, and fluid flow is highly desirable.

<div align="center">

Chapter 5

</div>

<div align="center">

On the Solution to Boundary Integral Equations Defined on Rotationally Symmetric Surfaces: High Order Discretization, Fast Kernel Computation, and Multibody Problems

</div>

5.1   Introduction

In this chapter, we present a numerical technique for discretizing and solving boundary integral equations (BIEs) defined on axisymmetric surfaces in $\mathbb{R}^3$. Specifically, we consider second kind Fredholm equations of the form

$$\sigma(\boldsymbol{x}) + \int_\Gamma k(\boldsymbol{x}, \boldsymbol{x}')\, \sigma(\boldsymbol{x}')\, dA(\boldsymbol{x}') = f(\boldsymbol{x}), \quad \boldsymbol{x} \in \Gamma, \tag{5.1}$$

under two assumptions: First, that Γ is a surface in $\mathbb{R}^3$ obtained by rotating a curve γ about an axis. Second, that the kernel k is invariant under rotation about the symmetry axis in the sense that

$$k(\boldsymbol{x}, \boldsymbol{x}') = k(\theta - \theta', r, z, r', z'), \tag{5.2}$$

where (r, z, θ) and (r', z', θ') are cylindrical coordinates for $\boldsymbol{x}$ and $\boldsymbol{x}'$, respectively,

$$\boldsymbol{x} = (r\,\cos\theta,\, r\,\sin\theta,\, z),$$

$$\boldsymbol{x}' = (r'\,\cos\theta',\, r'\,\sin\theta',\, z').$$

For an example of such a geometry, see Figure 5.1. Under these assumptions, the BIE given by (5.1), which is defined on the two-dimensional surface Γ, can via a Fourier transform in the azimuthal variable be recast as a sequence of equations defined on the one-dimensional curve γ. To be precise,

letting σ_n, f_n, and k_n denote the Fourier coefficients of σ, f, and k, respectively (so that (5.11), (5.12), and (5.13) hold), the equation (5.1) is equivalent to the sequence of equations

$$\sigma_n(r,z) + \sqrt{2\pi} \int_\gamma k_n(r,z,r',z')\,\sigma_n(r',z')\,r'\,dl(r',z') = f_n(r,z), \tag{5.3}$$

where $(r,z) \in \gamma$, $\ n \in \mathbb{Z}$. Whenever f can be represented with a moderate number of Fourier modes, the formula (5.3) provides an efficient technique for computing the corresponding modes of σ. The conversion of (5.1) to (5.3) appears in, *e.g.*, [128], and is described in detail in Section 5.2.

Equations of the type (5.1) arise in many areas of mathematical physics and engineering, commonly as reformulations of elliptic partial differential equations. Advantages of a BIE approach include a reduction in dimensionality, often a radical improvement in the conditioning of the mathematical equation to be solved, a natural way of handling problems defined on exterior domains, and a relative ease in implementing high-order discretization schemes, see, *e.g.*, [3].

The numerical solution of BIEs such as (5.1) poses certain difficulties, the foremost being that the discretizations generally involve dense matrices. Until the 1980s, this issue often times made it prohibitively expensive to use BIE formulations as numerical tools. However, with the advent of "fast" algorithms (the Fast Multipole Method [65, 66], panel clustering [69], etc.) for matrix-vector multiplication and the inversion of dense matrices arising from the discretization of BIE operators, these problems have largely been overcome for problems in two dimensions. This is not necessarily the case in three dimensions; issues such as surface representation and the construction of quadrature rules in a three dimensional environment still pose unresolved questions. The point of recasting the single BIE (5.1) defined on a surface as the sequence of BIEs (5.3) defined on a curve is in part to avoid these difficulties in discretizing surfaces, and in part to exploit the exceptionally high speed of the Fast Fourier Transform (FFT).

The reduction of (5.1) to (5.3) is only applicable when the geometry of the boundary is axisymmetric, but presents no such restriction in regard to the boundary load. Formulations of this kind have been known for a long time, and have been applied to problems in stress analysis [8], scattering [51, 85, 137, 143, 149], and potential theory [68, 126, 128, 135]. Most of these

approaches have relied on collocation or Galerkin discretizations and have generally employed low-order accurate discretizations. A complication of the axisymmetric formulation is the need to determine the kernels k_n for a large number of Fourier modes n, since direct integration of (5.2) through the azimuthal variable tends to be prohibitively expensive. When k is smooth, this calculation can rapidly be accomplished using the FFT, but when k is near-singular, other techniques are required (quadrature, local refinement, etc.) that can lead to significant slowdown in the construction of the linear systems.

The technique described in this chapter improves upon previous work in terms of both accuracy and speed. The gain in accuracy is attained by constructing a high-order quadrature scheme for kernels with integrable singularities. This quadrature is obtained by locally modifying a Gaussian quadrature scheme, in a manner similar to that of [17, 19]. Numerical experiments indicate that for simple surfaces, a relative accuracy of 10^{-10} is obtained using as few as a hundred points along the generating curve. The rapid convergence of the discretization leads to linear systems of small size that can be solved *directly* via, *e.g.*, Gaussian elimination, making the algorithm particularly effective in environments involving multiple right hand sides and when the linear system is ill-conditioned. To describe the asymptotic complexity of the method, we need to introduce some notation. We let N_γ denote the number of points used to discretize the generating curve γ, and we let N_F denote the number of Fourier modes included in the calculation. Then, assuming $N_\gamma \approx N_\mathrm{F}$, the total number of degrees of freedom in the discretization is $N = N_\gamma N_\mathrm{F}$. Splitting the computational cost into a "set-up" cost that needs to be incurred only once for a given geometry and given discretization parameters, and a "solve" cost representing the time required to process each right hand side, we have

$$T_{\text{setup}} \sim \underbrace{N^{3/2} \log(N)}_{\text{construction of linear systems}} + \underbrace{N^2}_{\text{inversion of systems}} , \tag{5.4}$$

and

$$T_{\text{solve}} \sim \underbrace{N \log(N)}_{\text{FFT of boundary data}} + \underbrace{N^{3/2}}_{\text{application of inverses}} . \tag{5.5}$$

The technique described gets particularly efficient for problems of the form (5.1) in which the kernel k is either the single or the double layer potential associated with Laplace's equation and the Helmholtz equation. We demonstrate that for such problems, it is possible to exploit recursion relations for Legendre functions to very rapidly construct the Fourier coefficients k_n in (5.3). This greatly reduces the constant of proportionality of the setup (which requires the construction of a sequence of dense matrices), as one would alternatively have to use composite quadrature in calculating the kernels. Numerical experiments demonstrate that for a problem with $N_\mathrm{P} = 80$, $N_\mathrm{G} = 10$, and $N_\mathrm{F} = 400$ (for a total of $80 \times 10 \times 400 = 320\,000$ degrees of freedom), this accelerated scheme requires only 57 seconds for the setup, and 0.39 seconds for each solve when implemented on a standard desktop PC.

We further extend the algorithm to problems involving multiply connected domains. By combining the high order discretization described in this paper with the Fast Multipole Method, we can achieve ten digits of accuracy for a problem containing 27 bodies and over $500\,000$ unknowns in a handful of minutes.

The final extension of the technique described in this paper is the development of an accelerated method for calculating the kernels of the Helmholtz equation in the axisymmetric environment. The high accuracy of the discretization combined with a direct solver allows us to solve problems of up to 25 wavelengths in size in 92 seconds with 10 digits of accuracy, with additional solves computed in 0.56 seconds on a standard PC.

The techniques described in this chapter can be accelerated further by combining it with a fast solver applied to each of the equations in (5.3), such as those based on the Fast Multipole Method, or the fast direct solver of [95]. This would result in a highly accurate scheme with near optimal complexity.

The chapter is organized as follows: Section 5.2 describes the reduction of (5.1) to (5.3) and quantifies the error incurred by truncating the Fourier series. Section 5.3 presents the Nyström discretization of the reduced equations using high-order quadrature applicable to kernels with integrable singularities, and the construction of the resulting linear systems. Section 5.4 summarizes
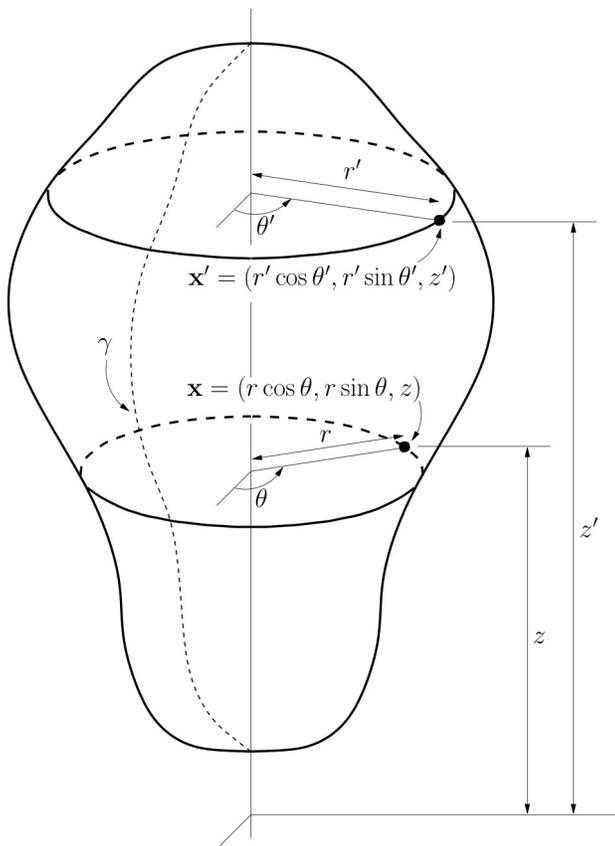
Figure 5.1: The axisymmetric domain Γ generated by the curve γ.

the algorithm for the numerical solution of (5.3) and describes its computational costs. Section 5.5 presents the application of the algorithm for BIE formulations of Laplace's equation and describes the rapid calculation of k_n in this setting. Section 5.6 presents numerical examples applied to Laplace's equation. Section 5.7 extends our approach to multiply connected domains and Section 5.8 presents numerical results for this geometry. Section 5.9 presents an accelerated technique for calculating the kernels related to the Helmholtz equation and presents numerical results. Section 5.10 gives conclusions and possible extensions and generalizations.

5.2   Fourier Representation of BIE

5.2.1   Problem Formulation

Suppose that Γ is a surface in $\mathbb{R}^3$ obtained by rotating a smooth contour γ about a fixed axis and consider the boundary integral equation

$$\sigma(\boldsymbol{x}) + \int_\Gamma k(\boldsymbol{x}, \boldsymbol{x}')\, \sigma(\boldsymbol{x}')\, dA(\boldsymbol{x}') = f(\boldsymbol{x}), \quad \boldsymbol{x} \in \Gamma. \tag{5.6}$$

In this section, we will demonstrate that if the kernel k is rotationally symmetric in a sense to be made precise, then by taking the Fourier transform in the azimuthal variable, (5.6) can be recast as a sequence of BIEs defined on the curve γ. To this end, we introduce a Cartesian coordinate system in $\mathbb{R}^3$ with the third coordinate axis being the axis of symmetry. Then cylindrical coordinates (r, z, θ) are defined such that

$$x_1 = r \cos \theta,$$

$$x_2 = r \sin \theta,$$

$$x_3 = z.$$

Figure 5.1 illustrates the coordinate system.

The kernel k in (5.6) is now rotationally symmetric if for any two points $\boldsymbol{x}, \boldsymbol{x}' \in \Gamma$,

$$k(\boldsymbol{x}, \boldsymbol{x}') = k(\theta - \theta', r, z, r', z'), \tag{5.7}$$

where (θ', r', z') are the cylindrical coordinates of $\boldsymbol{x}'$.

5.2.2   Separation of Variables

We define for $n \in \mathbb{Z}$ the functions f_n, σ_n, and k_n via

$$f_n(r, z) = \int_\mathbb{T} \frac{e^{-in\theta}}{\sqrt{2\pi}}\, f(\theta, r, z)\, d\theta, \tag{5.8}$$

$$\sigma_n(r, z) = \int_\mathbb{T} \frac{e^{-in\theta}}{\sqrt{2\pi}}\, \sigma(\theta, r, z)\, d\theta, \tag{5.9}$$

$$k_n(r, z, r', z') = \int_\mathbb{T} \frac{e^{-in\theta}}{\sqrt{2\pi}}\, k(\theta, r, z, r', z')\, d\theta. \tag{5.10}$$

The definitions (5.8), (5.9), and (5.10) define f_n, σ_n, and k_n as the coefficients in the Fourier series

of the functions f, σ, and k about the azimuthal variable,

$$f(\boldsymbol{x}) = \sum_{n \in \mathbb{Z}} \frac{e^{in\theta}}{\sqrt{2\pi}} f_n(r, z), \tag{5.11}$$

$$\sigma(\boldsymbol{x}) = \sum_{n \in \mathbb{Z}} \frac{e^{in\theta}}{\sqrt{2\pi}} \sigma_n(r, z), \tag{5.12}$$

$$k(\boldsymbol{x}, \boldsymbol{x}') = k(\theta - \theta', r, z, r', z') = \sum_{n \in \mathbb{Z}} \frac{e^{in(\theta-\theta')}}{\sqrt{2\pi}} k_n(r, z, r', z'). \tag{5.13}$$

To determine the Fourier representation of (5.6), we multiply the equation by $e^{-in\theta}/\sqrt{2\pi}$ and

integrate θ over $\mathbb{T}$ (for our purposes, we can think of $\mathbb{T}$ as simply the interval $[-\pi, \pi]$). Equation

(5.6) can then be said to be equivalent to the sequence of equations

$$\sigma_n(r, z) + \int_{\gamma \times \mathbb{T}} \left[\int_{\mathbb{T}} \frac{e^{-in\theta}}{\sqrt{2\pi}} k(\boldsymbol{x}, \boldsymbol{x}') \, d\theta \right] \sigma(\boldsymbol{x}') \, dA(\boldsymbol{x}') = f_n(r, z), \qquad n \in \mathbb{Z}. \tag{5.14}$$

Invoking (5.13), we evaluate the bracketed factor in (5.14) as

$$\int_{\mathbb{T}} \frac{e^{-in\theta}}{\sqrt{2\pi}} k(\boldsymbol{x}, \boldsymbol{x}') \, d\theta = \int_{\mathbb{T}} \frac{e^{-in\theta}}{\sqrt{2\pi}} k(\theta - \theta', r, z, r', z') \, d\theta$$

$$= e^{-in\theta'} \int_{\mathbb{T}} \frac{e^{-in(\theta-\theta')}}{\sqrt{2\pi}} k(\theta - \theta', r, z, r', z') \, d\theta = e^{-in\theta'} k_n(r, z, r', z'). \tag{5.15}$$

Inserting (5.15) into (5.14) and executing the integration of θ' over $\mathbb{T}$, we find that (5.6) is equivalent

to the sequence of equations

$$\sigma_n(r, z) + \sqrt{2\pi} \int_{\gamma} k_n(r, z, r', z') \, \sigma_n(r', z') \, r' \, dl(r', z') = f_n(r, z), \quad n \in \mathbb{Z}. \tag{5.16}$$

For future reference, we define for $n \in \mathbb{Z}$ the boundary integral operators $\mathcal{K}_n$ via

$$[\mathcal{K}_n \sigma_n](r, z) = \sqrt{2\pi} \int_{\gamma} k_n(r, z, r', z') \, \sigma_n(r', z') \, r' \, dl(r', z'). \tag{5.17}$$

Then equation (5.16) can be written

$$\left(I + \mathcal{K}_n \right) \sigma_n = f_n, \qquad n \in \mathbb{Z}. \tag{5.18}$$

When each operator $I + \mathcal{K}_n$ is continuously invertible, we can write the solution of (5.6) as

$$\sigma(r, z, \theta) = \sum_{n \in \mathbb{Z}} \frac{e^{in\theta}}{\sqrt{2\pi}} [(I + \mathcal{K}_n)^{-1} f_n](r, z). \tag{5.19}$$

5.2.3   Truncation of the Fourier series

When evaluating the solution operator (5.19) in practice, we will choose a truncation parameter N_{F}, and evaluate only the lowest $2N_{\mathrm{F}} + 1$ Fourier modes. If N_{F} is chosen so that the given function f is well-represented by its lowest $2N_{\mathrm{F}}+1$ Fourier modes, then in typical environments the solution obtained by truncating the sum (5.19) will also be accurate. To substantiate this claim, suppose that ε is a given tolerance, and that N_{F} has been chosen so that

$$||f - \sum_{n=-N_{\mathrm{F}}}^{N_{\mathrm{F}}} \frac{e^{in\theta}}{\sqrt{2\pi}} f_n|| \leq \varepsilon, \tag{5.20}$$

We define an approximate solution via

$$\sigma_{\mathrm{approx}} = \sum_{n=-N_{\mathrm{F}}}^{N_{\mathrm{F}}} \frac{e^{in\theta}}{\sqrt{2\pi}} (I + \mathcal{K}_n)^{-1} f_n. \tag{5.21}$$

From Parseval's identity, we then find that the error in the solution satisfies

$$||\sigma - \sigma_{\mathrm{approx}}||^2 = \sum_{|n|>N_{\mathrm{F}}} ||(I + \mathcal{K}_n)^{-1} f_n||^2 \leq \sum_{|n|>N_{\mathrm{F}}} ||(I + \mathcal{K}_n)^{-1}||^2 \, ||f_n||^2$$

$$\leq \left(\max_{|n|>N_{\mathrm{F}}} ||(I + \mathcal{K}_n)^{-1}||^2 \right) \sum_{|n|>N_{\mathrm{F}}} ||f_n||^2 \leq \left(\max_{|n|>N_{\mathrm{F}}} ||(I + \mathcal{K}_n)^{-1}||^2 \right) \varepsilon^2.$$

It is typically the case that the kernel $k(\boldsymbol{x}, \boldsymbol{x}')$ has sufficient smoothness such that the Fourier modes $k_n(r, z, r', z')$ decay as $n \to \infty$. Then $||\mathcal{K}_n|| \to 0$ as $n \to \infty$ and $||(I + \mathcal{K}_n)^{-1}|| \to 1$. Thus, an accurate approximation of f leads to an approximation in σ that is of the same order of accuracy. Figure 5.4 illustrates that when k is the double layer kernel associated with the Laplace equation, and γ is a simple curve, then $||(I + \mathcal{K}_n)^{-1}|| \to 1$ with rapid convergence.

5.3   Discretization of BIEs in Two Dimensions

The technique described in Section 5.2 reduces the BIE (5.1) defined on an axisymmetric surface $\Gamma = \gamma \times \mathbb{T}$ contained in $\mathbb{R}^3$, to a sequence of BIEs defined on the curve γ contained in $\mathbb{R}^2$. These equations take the form

$$\sigma(\boldsymbol{x}) + \sqrt{2\pi} \int_\gamma k_n(\boldsymbol{x}, \boldsymbol{x}') \, \sigma(\boldsymbol{x}') \, r' \, dl(\boldsymbol{x}') = f(\boldsymbol{x}), \qquad \boldsymbol{x} \in \gamma, \tag{5.22}$$

where the kernel k_n is defined as in (5.10). In this section, we describe a standard technique for discretizing an equation such as (5.22). For simplicity, we limit attention to the case where γ is a smooth closed curve, but extensions to non-smooth curves can be handled by slight variations of the techniques described here, [17, 19, 72].

5.3.1   Parameterization of the Curve

Let γ be parameterized by a vector-valued smooth function $\boldsymbol{\tau} : [0, T] \to \mathbb{R}^2$. The parameterization converts (5.22) to an integral equation defined on the interval $[0, T]$:

$$\sigma(\boldsymbol{\tau}(t)) + \sqrt{2\pi} \int_0^T k_n(\boldsymbol{\tau}(t), \boldsymbol{\tau}(s))\, \sigma(\boldsymbol{\tau}(s))\, r'(\boldsymbol{\tau}(s))\, |d\boldsymbol{\tau}/ds|\, ds = f(\boldsymbol{\tau}(t)), \qquad (5.23)$$

where $t \in [0, T]$. To keep our formulas uncluttered, we suppress the parameterization of the curve and the dependence on n and introduce a new kernel

$$K(t, s) = \sqrt{2\pi}\, k_n(\boldsymbol{\tau}(t), \boldsymbol{\tau}(s))\, r'(\boldsymbol{\tau}(s))\, |d\boldsymbol{\tau}/ds|, \qquad (5.24)$$

as well as the functions

$$\varphi(t) = \sigma(\boldsymbol{\tau}(t)) \quad \text{and} \quad \psi(t) = f(\boldsymbol{\tau}(t)).$$

Then techniques for solving

$$\varphi(t) + \int_0^T K(t, s)\, \varphi(s)\, ds = \psi(t), \qquad t \in [0, T], \qquad (5.25)$$

where ψ is given and φ is to be determined, will be equally applicable to (5.23).

5.3.2   Nyström Method

We will discretize (5.25) via Nyström discretization on standard Gaussian quadrature nodes, see [3]. To this end, we divide the interval $\Omega = [0, T]$ into a disjoint partition of N_{P} intervals,

$$\Omega = \bigcup_{p=1}^{N_{\mathrm{P}}} \Omega_p,$$

where each Ω_p is a subinterval called a *panel*. On each panel Ω_p, we place the nodes of a standard N_{G}-point Gaussian quadrature rule $\{t_i^{(p)}\}_{i=1}^{N_{\mathrm{G}}}$. The idea is now to enforce (5.25) at each of the $N_{\mathrm{P}}N_{\mathrm{G}}$ nodes:

$$\sigma(t_i^{(p)}) + \int_0^T K(t_i^{(p)}, s)\,\varphi(s)\,ds = \psi(t_i^{(p)}), \qquad (i,p) \in \{1, 2, \ldots, N_{\mathrm{G}}\} \times \{1, 2, \ldots, N_{\mathrm{P}}\}.$$

To obtain a numerical method, suppose that we can construct for $p, q \in \{1, 2, \ldots, N_{\mathrm{P}}\}$ and $i, j \in \{1, 2, \ldots, N_{\mathrm{G}}\}$ numbers $A_{i,j}^{(p,q)}$ such that

$$\int_0^T K(t_i^{(p)}, s)\,\varphi(s)\,ds \approx \sum_{q=1}^{N_{\mathrm{P}}} \sum_{j=1}^{N_{\mathrm{G}}} A_{i,j}^{(p,q)}\,\varphi(t_j^{(q)}). \tag{5.26}$$

Then the Nyström method is given by solving the linear system

$$\varphi_i^{(p)} + \sum_{q=1}^{N_{\mathrm{P}}} \sum_{j=1}^{N_{\mathrm{G}}} A_{i,j}^{(p,q)}\,\varphi_j^{(q)} = \psi_i^{(p)}, \qquad (i,p) \in \{1, 2, \ldots, N_{\mathrm{G}}\} \times \{1, 2, \ldots, N_{\mathrm{P}}\}, \tag{5.27}$$

where $\psi_i^{(p)} = \psi(t_i^{(p)})$ and $\varphi_i^{(p)}$ approximates $\varphi(t_i^{(p)})$. We write (5.27) compactly as

$$(I + A)\,\varphi = \psi$$

where A is a matrix formed by $N_{\mathrm{P}} \times N_{\mathrm{P}}$ blocks, each of size $N_{\mathrm{G}} \times N_{\mathrm{G}}$. We let $A^{(p,q)}$ denote the block of A representing the interactions between the panels Ω_p and Ω_q.

5.3.3 Quadrature and Interpolation

We need to determine the numbers $A_{i,j}^{(p,q)}$ such that (5.26) holds. The detailed construction is given in Section 5.3.4, and utilizes some well-known techniques of quadrature and interpolation, which we review in this section.

5.3.3.1 Standard Gaussian Quadratures

Given an interval $[0, h]$ and a positive integer N_{G}, the N_{G}-point standard Gaussian quadrature rule consists of a set of N_{G} *nodes* $\{t_j\}_{j=1}^{N_{\mathrm{G}}} \subset [0, h]$, and N_{G} *weights* $\{w_j\}_{j=1}^{N_{\mathrm{G}}}$ such that

$$\int_0^h g(s)\,ds = \sum_{j=1}^{N_{\mathrm{G}}} w_j\,g(t_j),$$

whenever g is a polynomial of degree at most $2N_G - 1$, and such that

$$\int_0^h g(s)\,ds = \sum_{j=1}^{N_G} w_j\, g(t_j) + O(h^{2N_G}),$$

whenever g is a function with $2N_G$ continuous derivatives, see [1].

5.3.3.2   Quadrature Rules for Singular Functions

Now suppose that given an interval $[0,\ h]$ and a point $t \in [-h,\ 2h]$, we seek to integrate over $[0,\ h]$ functions g that take the form

$$g(s) = \phi_1(s) \log|s - t| + \phi_2(s), \tag{5.28}$$

where ϕ_1 and ϕ_2 are polynomials of degree at most $2N_G - 1$. Standard Gaussian quadrature would be highly inaccurate if applied to integrate (5.28). Rather, we seek a N_G'-node quadrature that will evaluate

$$\int_0^h g(s)\,ds \tag{5.29}$$

exactly. Techniques for constructing such generalized quadratures are readily available in the literature, see for example [83]. These quadratures will be of degree $2N_G - 1$, just as with standard Gaussian quadratures and exhibit comparable accuracy, although in general $N_G' > N_G$. The generalized quadratures used in this chapter were determined using the techniques of [83], and can be found in the appendix.

We observe that the quadrature nodes constructed by such methods are typically different from the nodes of the standard Gaussian quadrature. This complicates the the construction of the matrix A, as described in Section 5.3.4.

5.3.3.3   Lagrange Interpolation

Let $\{t_j\}_{j=1}^{N_G}$ denote the nodes of a N_G-point Gaussian quadrature rule on the interval $[0, h]$. If the values of a polynomial g of degree at most $N_G - 1$ are specified at these nodes, the entire

polynomial g can be recovered via the formula

$$g(s) = \sum_{j=1}^{N_G} L_j(s)\, g(t_j),$$

where the functions L_j are the Lagrange interpolating polynomials

$$L_j(s) = \prod_{i \neq j} \left(\frac{s - t_i}{t_j - t_i} \right).$$

If g is a smooth function with N_G continuous derivatives that is not a polynomial, then the Lagrange interpolant provides an approximation to g satisfying

$$\left| g(s) - \sum_{j=1}^{N_G} L_j(s)\, g(t_j) \right| \leq C\, h^{N_G},$$

where

$$C = \left(\sup_{s \in [0,h]} |g^{(N_G)}(s)| \right) / N_G!.$$

5.3.4   Constructing the Matrix A

Using the tools reviewed in Section 5.3.3, we are now in position to construct numbers $A_{i,j}^{(p,q)}$ such that (5.26) holds. We first note that in forming block $A^{(p,q)}$ of A, we need to find numbers $A_{i,j}^{(p,q)}$ such that

$$\int_{\Omega_q} K(t_i^{(p)}, s)\, \varphi(s)\, ds \approx \sum_{j=1}^{N_G} A_{i,j}^{(p,q)}\, \varphi(t_j^{(q)}), \qquad i = 1,\, 2,\, \ldots,\, N_G. \tag{5.30}$$

When Ω_p and Ω_q are well separated, the integrand in (5.30) is smooth, and our task is easily solved using standard Gaussian quadrature (as described in Section 5.3.3.1):

$$\int_{\Omega_q} K(t_i^{(p)}, s)\, \varphi(s)\, ds \approx \sum_{j=1}^{N_G} w_j\, K(t_i^{(p)}, t_j^{(q)})\, \varphi(t_j^{(q)}).$$

It directly follows that the ij entry of the block $A^{(p,q)}$ takes the form

$$A_{i,j}^{(p,q)} = w_j\, K(t_i^{(p)},\, t_j^{(q)}). \tag{5.31}$$

Complications arise when we seek to form a diagonal block $A^{(p,p)}$, or even a block that is adjacent to a diagonal block. The difficulty is that the kernel $K(t, s)$ has a singularity as $s \to t$.

To be precise, for the kernels considered in this chapter, we have that for any fixed t, there exist smooth functions u_t and v_t such that

$$K(t, s) = \log |t - s| \, u_t(s) + v_t(s).$$

We see that when $t_i^{(p)}$ is a point in Ω_q the integrand in (5.30) becomes singular. When $t_i^{(p)}$ is a point in a panel neighboring Ω_q, the problem is less severe, but Gaussian quadrature would still be inaccurate. To maintain full accuracy, we use the modified quadrature rules described in Section 5.3.3.2. For every node $t_i^{(p)} \in \Omega_p$, we construct a quadrature $\{\hat{w}_{i,\ell}^{(p,q)}, \hat{t}_{i,\ell}^{(p,q)}\}_{\ell=1}^{N_G'}$ such that

$$\int_{\Omega_q} K(t_i^{(p)}, s) \, \varphi(s) \, ds \approx \sum_{\ell=1}^{N_G'} \hat{w}_{i,\ell}^{(p,q)} \, K(t_i^{(p)}, \hat{t}_{i,\ell}^{(p,q)}) \, \varphi(\hat{t}_{i,\ell}^{(p,q)}). \tag{5.32}$$

In order to have a quadrature evaluated at the Gaussian nodes $t_j^{(q)} \in \Omega_q$, we next use Lagrange interpolation as described in Section 5.3.3.3. With $\{L_j^{(q)}\}_{j=1}^{N_G}$ denoting the Lagrange interpolants of order $N_G - 1$ defined on Ω_q, we have

$$\varphi(t) \approx \sum_{j=1}^{N_G} L_j^{(q)}(t) \, \varphi(t_j^{(q)}), \qquad t \in \Omega_q. \tag{5.33}$$

Inserting (5.33) into (5.32), we find that

$$\int_{\Omega_q} K(t_i^{(p)}, s) \, \varphi(s) \, ds \approx \sum_{\ell=1}^{N_G'} \hat{w}_{i,\ell}^{(p,q)} \, K(t_i^{(p)}, \hat{t}_{i,\ell}^{(p,q)}) \sum_{j=1}^{N_G} L_j^{(q)}(\hat{t}_{i,\ell}^{(p,q)}) \, \varphi(t_j^{(q)}).$$

We now find that the block $A^{(p,q)}$ of A has entries

$$A_{i,j}^{(p,q)} = \sum_{\ell=1}^{N_G'} \hat{w}_{i,\ell}^{(p,q)} \, K(t_i^{(p)}, \hat{t}_{i,\ell}^{(p,q)}) \, L_j^{(q)}(\hat{t}_{i,\ell}^{(p,q)}), \qquad i, j \in \{1, 2, \ldots, N_G\}. \tag{5.34}$$

We observe that the formula (5.34) is quite expensive to evaluate; in addition to the summation, it requires the construction of a quadrature rule for each point $t_i^{(p)}$ and evaluation of Lagrange interpolants. Fortunately, this process must be executed for at most three blocks in each row of blocks of A.

5.4   A General Algorithm

5.4.1   Summary

At this point, we have shown how to convert a BIE defined on an axisymmetric surface in $\mathbb{R}^3$ to a sequence of equations defined on a curve in $\mathbb{R}^2$ (Section 5.2), and then how to discretize each of these reduced equations (Section 5.3). Putting everything together, we obtain the following algorithm for solving (5.1):

(1) Given the right hand side f, and a computational tolerance ε, determine a truncation parameter N_{F} such that (5.20) holds.

(2) Form for $n = -N_{\mathrm{F}},\ -N_{\mathrm{F}} + 1,\ -N_{\mathrm{F}} + 2,\ \ldots,\ N_{\mathrm{F}}$ the matrix A_n discretizing the equation (5.18) encapsulating the n'th Fourier mode. The matrix is formed via Nyström discretization as described in Section 5.3 with the discretization parameters N_{P} and N_{G} chosen to meet the computational tolerance ε.

(3) Evaluate via the FFT the terms $\{f_n\}_{n=-N_{\mathrm{F}}}^{N_{\mathrm{F}}}$ in the Fourier representation of f (as defined by (5.8)), and solve for $n = -N_{\mathrm{F}},\ -N_{\mathrm{F}}+1,\ -N_{\mathrm{F}}+2,\ \ldots,\ N_{\mathrm{F}}$ the equation $(I + A_n)\,\sigma_n = f_n$ for σ_n. Construct σ_{approx} using formula (5.21) evaluated via the FFT.

The construction of the matrices A_n in Step 2 can be accelerated using the FFT (as described in Section 5.4.2), but even with such acceleration, it is typically by a wide margin the most expensive part of the algorithm. However, this step needs to be performed only once for any given geometry, and given discretization parameters N_{F}, N_{P}, and N_{G}. The method therefore becomes particularly efficient when (5.1) needs to be solved for a sequence of right-hand sides. In this case, it may be worth the cost to pre-compute the inverse of each matrix $I + A_n$.

5.4.2   Techniques for Forming the Matrices

We need to construct for each Fourier mode n, a matrix A_n consisting of $N_{\mathrm{P}} \times N_{\mathrm{P}}$ blocks $A_n^{(p,q)}$, each of size $N_{\mathrm{G}} \times N_{\mathrm{G}}$. Constructing an off-diagonal block $A_n^{(p,q)}$ when Ω_p and Ω_q are not

directly adjacent is straightforward. For any pair of nodes $t_i^{(p)} \in \Omega_p$ and $t_j^{(q)} \in \Omega_q$, we need to construct the numbers, *cf.* (5.24) and (5.31),

$$A_{n;i,j}^{(p,q)} = \sqrt{2\pi}\, w_j\, k_n(\boldsymbol{\tau}(t_i^{(p)}),\, \boldsymbol{\tau}(t_j^{(q)}))\, r'(\boldsymbol{\tau}(t_j^{(q)}))\, |d\boldsymbol{\tau}(t_j^{(q)})/ds|, \tag{5.35}$$

for $n = -N_{\mathrm{F}}, -N_{\mathrm{F}} + 1, \ldots, N_{\mathrm{F}}$, where $\boldsymbol{\tau}$ is a parameterization of γ (see Section 5.3.1) and the kernel k_n is defined by (5.10). Fortunately, we do not need to explicitly evaluate the integrals in (5.10) since all the $2N_{\mathrm{F}} + 1$ numbers can be evaluated by a single application of the FFT to the function

$$\theta \mapsto k(\theta,\, \boldsymbol{\tau}(t_i^{(p)}),\, \boldsymbol{\tau}(t_j^{(q)})). \tag{5.36}$$

When $\boldsymbol{\tau}(t_i^{(p)})$ is not close to $\boldsymbol{\tau}(t_j^{(q)})$, the function in (5.36) is smooth, and the trapezoidal rule implicit in applying the FFT is highly accurate.

Evaluating the blocks on the diagonal, or directly adjacent to the diagonal is somewhat more involved. The matrix entries are now given by the formula, *cf.* (5.24) and (5.34),

$$A_{k;i,j}^{(p,q)} = \sum_{\ell=1}^{N_{\mathrm{G}}'} \hat{w}_{i,\ell}^{(p,q)}\, k_n(\boldsymbol{\tau}(t_i^{(p)}), \boldsymbol{\tau}(\hat{t}_{i,\ell}^{(p,q)}))\, r'(\hat{t}_{i,\ell}^{(p,q)})\, |d\boldsymbol{\tau}(\hat{t}_{i,\ell}^{(p,q)})/ds|\, L_j^{(q)}(\hat{t}_{i,\ell}^{(p,q)}), \tag{5.37}$$

where $\boldsymbol{\tau}$ and k_n are as in (5.35). To further complicate things, the points $\boldsymbol{\tau}(t_i^{(p)})$ and $\boldsymbol{\tau}(\hat{t}_{i,\ell}^{(p,q)})$ are now in close proximity to each other, and so the functions

$$\theta \mapsto k(\theta,\, \boldsymbol{\tau}(t_i^{(p)}),\, \boldsymbol{\tau}(\hat{t}_{i,\ell}^{(p,q)})) \tag{5.38}$$

have a sharp peak around the point $\theta = 0$. They are typically still easy to integrate away from the origin, so the integrals in (5.10) can for a general kernel be evaluated relatively efficiently using quadratures that are adaptively refined near the origin.

Even with the accelerations described in this section, the cost of forming the matrices A_n tends to dominate the computation whenever the kernels k_n must be evaluated via formula (5.10). In particular environments, it is possible to side-step this problem by evaluating the integral in (5.10) analytically. That this can be done for the single and double layer kernels associated with Laplace's equation is demonstrated in Section 5.5.

5.4.3 Computational Costs

The asymptotic cost of the algorithm described in Section 5.4.1 has three components: (a) the cost of forming the matrices $\{A_n\}_{n=-N_F}^{N_F}$, (b) the cost of transforming functions from physical space to Fourier space and back, and (c) the cost of solving the linear systems $(I + A_n)\, \sigma_n = f_n$. In this section, we investigate the asymptotic cost of these steps. We consider a situation where N_F Fourier modes need to be resolved, and where $N_P N_G$ nodes are used to discretize the curve γ. For simplicity of the presentation, we will assume that the parameter N_G is fixed, and set $N_\gamma = N_G N_P$. Further, we will make the assumption that $N_F \approx N_\gamma$ so that the total number of degrees of freedom used in the discretization by N, where $N^{1/2} = N_F = N_\gamma$.

(a) Cost of forming the linear systems: Suppose first that we have an analytic formula for each kernel k_n (As we do, *e.g.*, when the original BIE (5.1) involves either the single or the double layer kernel associated with Laplace's equation, see Section 5.5). Then the cost T_{mat} of forming the matrices satisfies

$$T_{\text{mat}} \quad \sim \quad \underbrace{N^{3/2}}_{\text{cost from kernel evaluations}} \quad + \quad \underbrace{N}_{\text{cost from composite quadrature}} \quad = N^{3/2}.$$

When the kernels have to be evaluated numerically via formula (5.10), the cost of forming the matrices is still moderate. In the rare situations where the kernel is smooth, standard Gaussian quadrature can be used everywhere and the FFT acceleration described in Section 5.4.2 can be used for all entries. In this situation,

$$T_{\text{mat}} \sim N^{3/2} \log N$$

Note that the constant of proportionality is very small in this case. In the more typical situation where each kernel k_n involves an integrable singularity at the diagonal, the FFT acceleration can still be used to rapidly evaluate all entries well-removed from the diagonal. However, entries close to the diagonal must be formed via the composite quadrature rule combined with numerical evaluation of k_n via an adaptive quadrature. In this situation,

$$T_{\text{mat}} \sim N^{3/2} \log N + N^{3/2} = N^{3/2} \log N,$$

but the constant of proportionality can be very large in this situation.

(b) Cost of Fourier transforms: The boundary data defined on the surface must be converted into the Fourier domain. This is executed via the FFT at a cost T_{fft} satisfying

$$T_{\text{fft}} \sim N \log(N). \tag{5.39}$$

We observe that the constant of proportionality in (5.39) is very small, and the cost of this step is typically negligible compared to the costs of the other steps.

(c) Cost of linear solves: Using standard Gaussian elimination, the cost T_{solve} of solving N_{F} linear systems $(I + A_n)\, \sigma_n = f_n$, each of size $N_\gamma \times N_\gamma$ under the assumptions stated previously satisfies

$$T_{\text{solve}} \sim N^2.$$

In situations where the equations need to be solved for multiple right hand sides, it pays off to first compute the inverses $(I + A_n)^{-1}$, and then simply apply these to each right hand side (or, alternatively, to form the LU factorizations, and then perform triangular solves). The cost T_{inv} of computing the inverses, and the cost T_{apply} of applying them then satisfy

$$T_{\text{inv}} \sim N^2,$$

$$T_{\text{apply}} \sim N^{3/2}.$$

We make some practical observations:

- The cost of forming the matrices and inverting them dominates the other costs of the algorithm unless the kernel is either smooth, or analytic formulas for k_n are available.

- The scheme is highly efficient in situations where the same equation needs to be solved for a sequence of different right hand sides. Given an additional right hand side, the added cost T_{solve} is given by

$$T_{\text{solve}} \sim N \log N + N^{3/2},$$

  with a very small constant of proportionality. We note that this cost remains small even if an analytic formula for k_n is not available.

- The system matrices $I + A_n$ often have internal structure that allow them to be inverted using "fast methods" such as, *e.g.*, those in [95]. The cost of inversion and application can then be accelerated to near optimal complexity.

- The high order discretization employed requires a small number of points to to obtain a highly accurate solution. In practical terms, this means that despite non-optimal asymptotic work estimates, the time to solution for a given problem is extremely rapid.

- The estimates for this section critically rely on the assumption that $N_\gamma \approx N_F$. When this is not true, similar estimates hold but the bookkeeping is more involved.

5.5   Accelerations for the Single and Double Layer Kernels Associated with Laplace's Equation

In this section, we present a rapid technique based on recursion relations for calculating the single and double layer kernels associated with Laplace's equation. These results are then extended in Section 5.9 to the single and double layer kernels associated with the Helmholtz equation.

5.5.1   The Double Layer Kernels of Laplace's Equation

Let $D \subset \mathbb{R}^3$ be a bounded domain whose boundary is given by a smooth surface Γ, let $E = \bar{D}^c$ denote the domain exterior to D, and let $\boldsymbol{n}$ and be the outward unit normal to D. Consider the interior and exterior Dirichlet problems of potential theory [67],

$$\Delta u = 0 \ \text{ in } \ D, \quad u = f \ \text{ on } \ \Gamma, \qquad \text{(interior Dirichlet problem)} \tag{5.40}$$

$$\Delta u = 0 \ \text{ in } \ E, \quad u = f \ \text{ on } \ \Gamma. \qquad \text{(exterior Dirichlet problem)} \tag{5.41}$$

The solutions to (5.40) and (5.41) can be written in the respective forms

$$u(\boldsymbol{x}) = \int_\Gamma \frac{\boldsymbol{n}(\boldsymbol{x}') \cdot (\boldsymbol{x} - \boldsymbol{x}')}{4\pi |\boldsymbol{x} - \boldsymbol{x}'|^3} \sigma(\boldsymbol{x}') \, dA(\boldsymbol{x}'), \quad \boldsymbol{x} \in D,$$

$$u(\boldsymbol{x}) = \int_\Gamma \left(-\frac{\boldsymbol{n}(\boldsymbol{x}') \cdot (\boldsymbol{x} - \boldsymbol{x}')}{4\pi |\boldsymbol{x} - \boldsymbol{x}'|^3} + \frac{1}{4\pi |\boldsymbol{x} - \boldsymbol{x}_0|} \right) \sigma(\boldsymbol{x}') \, dA(\boldsymbol{x}'), \quad \boldsymbol{x} \in E, \ \boldsymbol{x}_0 \in D,$$

where σ is a boundary charge distribution that can be determined using the boundary conditions. The resulting equations are

$$-\frac{1}{2}\sigma(\boldsymbol{x}) + \int_\Gamma \frac{\boldsymbol{n}(\boldsymbol{x}') \cdot (\boldsymbol{x} - \boldsymbol{x}')}{4\pi|\boldsymbol{x} - \boldsymbol{x}'|^3} \sigma(\boldsymbol{x}')\, dA(\boldsymbol{x}') = f(\boldsymbol{x}), \tag{5.42}$$

$$-\frac{1}{2}\sigma(\boldsymbol{x}) + \int_\Gamma \left(-\frac{\boldsymbol{n}(\boldsymbol{x}') \cdot (\boldsymbol{x} - \boldsymbol{x}')}{4\pi|\boldsymbol{x} - \boldsymbol{x}'|^3} + \frac{1}{4\pi|\boldsymbol{x} - \boldsymbol{x}_0|} \right) \sigma(\boldsymbol{x}')\, dA(\boldsymbol{x}') = f(\boldsymbol{x}), \tag{5.43}$$

where $\boldsymbol{x} \in \Gamma$ in (5.42) and (5.43).

Remark 1 There are other integral formulations for the solution to Laplace's equation. The double layer formulation presented here is a good choice in that it provides an integral operator that leads to well conditioned linear systems. However, the methodology of this chapter is equally applicable to single-layer formulations that lead to first kind Fredholm BIEs.

5.5.2 Separation of Variables

Using the procedure given in Section 5.2, if $\Gamma = \gamma \times \mathbb{T}$, then (5.40) and (5.41) can be recast as a series of BIEs defined along γ. We express $\boldsymbol{n}$ in cylindrical coordinates as

$$\boldsymbol{n}(\boldsymbol{x}') = (n_{r'} \cos\theta', n_{r'} \sin\theta', n_{z'}).$$

Further,

$$|\boldsymbol{x} - \boldsymbol{x}'|^2 = (r\cos\theta - r'\cos\theta')^2 + (r\sin\theta - r'\sin\theta')^2 + (z - z')^2$$

$$= r^2 + (r')^2 - 2rr'(\sin\theta\sin\theta' + \cos\theta\cos\theta') + (z - z')^2$$

$$= r^2 + (r')^2 - 2rr'\cos(\theta - \theta') + (z - z')^2$$

and

$$\boldsymbol{n}(\boldsymbol{x}') \cdot (\boldsymbol{x} - \boldsymbol{x}') = (n_{r'}\cos\theta', n_{r'}\sin\theta', n_{z'}) \cdot (r\cos\theta - r'\cos\theta', r\sin\theta - r'\sin\theta', z - z')$$

$$= n_{r'}r(\sin\theta\sin\theta' + \cos\theta\cos\theta') - n_{r'}r' + n_{z'}(z - z')$$

$$= n_{r'}(r\cos(\theta - \theta') - r') + n_{z'}(z - z').$$

Then for a point $\boldsymbol{x}' \in \Gamma$, the kernel of the internal Dirichlet problem can be expanded as

$$\frac{\boldsymbol{n}(\boldsymbol{x}') \cdot (\boldsymbol{x} - \boldsymbol{x}')}{4\pi |\boldsymbol{x} - \boldsymbol{x}'|^3} = \frac{1}{\sqrt{2\pi}} \sum_{n \in \mathbb{Z}} e^{in(\theta - \theta')} d_n^{(i)}(r, z, r', z'),$$

where

$$d_n^{(i)}(r, z, r', z') = \frac{1}{\sqrt{32\pi^3}} \int_{\mathbb{T}} e^{-in\theta} \left[\frac{n_{r'}(r\cos\theta - r') + n_{z'}(z - z')}{(r^2 + (r')^2 - 2rr'\cos\theta + (z - z')^2)^{3/2}} \right] d\theta.$$

Similarly, the kernel of the external Dirichlet problem can be written as

$$-\frac{\boldsymbol{n}(\boldsymbol{x}') \cdot (\boldsymbol{x} - \boldsymbol{x}')}{4\pi |\boldsymbol{x} - \boldsymbol{x}'|^3} + \frac{1}{4\pi |\boldsymbol{x} - \boldsymbol{x}_0|} = \frac{1}{\sqrt{2\pi}} \sum_{n \in \mathbb{Z}} e^{in(\theta - \theta')} d_n^{(e)}(r, z, r', z'),$$

with

$$d_n^{(e)}(r, z, r', z') = \frac{1}{\sqrt{32\pi^3}} \int_{\mathbb{T}} e^{-in\theta} \left(-\frac{n_{r'}(r\cos\theta - r') + n_{z'}(z - z')}{(r^2 + (r')^2 - 2rr'\cos\theta + (z - z')^2)^{3/2}} + \right.$$
$$\left. + \frac{1}{(r^2 + r_0^2 - 2rr_0\cos\theta + (z - z_0)^2)^{1/2}} \right) d\theta,$$

where $\boldsymbol{x}_0$ has been written in cylindrical coordinates as $(r_0\cos(\theta_0), r_0\sin(\theta_0), z_0)$. With the expansions of the kernels available, the procedure described in Section 5.4 can be used to solve (5.42) and (5.43) by solving

$$\sigma_n(r, z) + \sqrt{2\pi} \int_\gamma d_n^{(i)}(r, r', z, z') \sigma_n(r', z') \, r' \, dl(r', z') = f_n(r, z) \tag{5.44}$$

and

$$\sigma_n(r, z) + \sqrt{2\pi} \int_\gamma d_n^{(e)}(r, r', z, z') \sigma_n(r', z') \, r' \, dl(r', z') = f_n(r, z), \tag{5.45}$$

respectively for $n = -N_{\mathrm{F}}, -N_{\mathrm{F}} + 1, \ldots, N_{\mathrm{F}}$. Note that the kernels $d_n^{(i)}$ and $d_n^{(e)}$ contain a log-singularity when both $r' = r$ and $z' = z$.

Equivalently, (5.44) and (5.45) can be arrived at by considering Laplace's equation written in cylindrical coordinates,

$$\frac{\partial^2 u}{\partial^2 r} + \frac{1}{r}\frac{\partial u}{\partial r} + \frac{1}{r^2}\frac{\partial^2 u}{\partial^2 \theta} + \frac{\partial^2 u}{\partial^2 z} = 0,$$

Taking the Fourier transform of u with respect to theta θ gives

$$\frac{\partial^2 u_n}{\partial^2 r} + \frac{1}{r}\frac{\partial u_n}{\partial r} - \frac{n^2}{r^2}\frac{\partial^2 u_n}{\partial^2 \theta} + \frac{\partial^2 u_n}{\partial^2 z} = 0, \quad n \in \mathbb{Z},$$

where $e_n = e_n(\theta) = e^{in\theta}/\sqrt{2\pi}$ and $u = \sum_{n \in \mathbb{Z}} e_n u_n$. Then (5.44) and (5.45) are now associated with this sequence of PDEs.

5.5.3 Evaluation of Kernels

The values of $d_n^{(i)}$ and $d_n^{(e)}$ for $n = -N_F, -N_F + 1, \ldots, N_F$ need to be computed efficiently and with high accuracy to construct the Nyström discretization of (5.44) and (5.45). Note that the integrands of $d_n^{(i)}$ and $d_n^{(e)}$ are real valued and even functions on the interval $[-\pi, \pi]$. Therefore, $d_n^{(i)}$ can be written as

$$d_n^{(i)}(r, z, r', z') = \frac{1}{\sqrt{32\pi^3}} \int_{\mathbb{T}} \left[\frac{n_{r'}(r\cos t - r') + n_{z'}(z - z')}{(r^2 + (r')^2 - 2rr'\cos t + (z - z')^2)^{3/2}} \right] \cos(nt)\, dt. \tag{5.46}$$

Note that $d_n^{(e)}$ can be written in a similar form.

This integrand is oscillatory and increasingly peaked at the origin as both $r' \to r$ and $z' \to z$. As long as r' and r as well as z' and z are well separated, the integrand does not experience peaks near the origin, and as mentioned before, the FFT provides a fast and accurate way for calculating $d_n^{(i)}$ and $d_n^{(e)}$.

In regimes where the integrand is peaked, the FFT no longer provides a means of evaluating $d_n^{(i)}$ and $d_n^{(e)}$ with the desired accuracy. One possible solution to this issue is applying adaptive quadrature to fully resolve the peak. However, this must be done for each value of n required and becomes prohibitively expensive if N_F is large.

Fortunately, an analytical solution to (5.46) exists. As noted in [35], the single-layer kernel can be expanded with respect to the azimuthal variable as

$$s(\boldsymbol{x}, \boldsymbol{x}') = \frac{1}{4\pi|\boldsymbol{x} - \boldsymbol{x}'|} = \frac{1}{4\pi(r^2 + (r')^2 - 2rr'\cos(\theta - \theta') + (z - z')^2)^{1/2}}$$
$$= \frac{1}{\sqrt{2\pi}} \sum_{n \in \mathbb{Z}} e^{in(\theta - \theta')} s_n(r, z, r', z'),$$

where

$$s_n(r, z, r', z') = \frac{1}{\sqrt{32\pi^3}} \int_{\mathbb{T}} \frac{\cos(nt)}{(r^2 + (r')^2 - 2rr'\cos(t) + (z - z')^2)^{1/2}}\, dt$$
$$= \frac{1}{\sqrt{8\pi^3 rr'}} \int_{\mathbb{T}} \frac{\cos(nt)}{\sqrt{8(\chi - \cos(t))}}\, dt$$
$$= \frac{1}{\sqrt{8\pi^3 rr'}} \mathcal{Q}_{n-1/2}(\chi),$$

$\mathcal{Q}_{n-1/2}$ is the half-integer degree Legendre function of the second kind, and

$$\chi = \frac{r^2 + (r')^2 + (z - z')^2}{2rr'}.$$

To find an analytical form for (5.46), first note that in cylindrical coordinates the double-layer kernel can be written in terms of the single-layer kernel,

$$\frac{\boldsymbol{n}(\boldsymbol{x}') \cdot (\boldsymbol{x} - \boldsymbol{x}')}{4\pi|\boldsymbol{x} - \boldsymbol{x}'|^3} = \frac{n_{r'}(r\cos(\theta - \theta') - r') + n_{z'}(z - z')}{4\pi(r^2 + (r')^2 - 2rr'\cos(\theta - \theta') + (z - z')^2)^{3/2}}$$

$$= \frac{1}{4\pi}\left[n_{r'}\frac{\partial}{\partial r'}\left(\frac{1}{(r^2 + (r')^2 - 2rr'\cos(\theta - \theta') + (z - z')^2)^{1/2}} \right) + \right.$$

$$\left. + n_{z'}\frac{\partial}{\partial z'}\left(\frac{1}{(r^2 + (r')^2 - 2rr'\cos(\theta - \theta') + (z - z')^2)^{1/2}} \right) \right].$$

The coefficients of the Fourier series expansion of the double-layer kernel are then given by $d_n^{(i)}$, which can be written using the previous equation as

$$d_n^{(i)}(r, z, r', z') = n_{r'}\int_{\mathbb{T}} \frac{\partial}{\partial r'}\left(\frac{\cos(nt)}{(32\pi^3(r^2 + (r')^2 - 2rr'\cos(t) + (z - z')^2))^{1/2}} \right) dt +$$

$$+ n_{z'}\int_{\mathbb{T}} \frac{\partial}{\partial z'}\left(\frac{\cos(nt)}{(32\pi^3(r^2 + (r')^2 - 2rr'\cos(t) + (z - z')^2))^{1/2}} \right) dt$$

$$= n_{r'}\frac{\partial}{\partial r'}\left(\frac{1}{\sqrt{8\pi^3 rr'}}\mathcal{Q}_{n-1/2}(\chi) \right) + n_{z'}\frac{\partial}{\partial z'}\left(\frac{1}{\sqrt{8\pi^3 rr'}}\mathcal{Q}_{n-1/2}(\chi) \right)$$

$$= \frac{1}{\sqrt{8\pi^3 rr'}}\left[n_{r'}\left(\frac{\partial\mathcal{Q}_{n-1/2}(\chi)}{\partial\chi}\frac{\partial\chi}{\partial r'} - \frac{\mathcal{Q}_{n-1/2}(\chi)}{2r'} \right) + n_{z'}\frac{\partial\mathcal{Q}_{n-1/2}(\chi)}{\partial\chi}\frac{\partial\chi}{\partial z'} \right].$$

To utilize this form of $d_n^{(i)}$, set $\mu = \sqrt{\frac{2}{\chi+1}}$ and note that

$$\frac{\partial\chi}{\partial r'} = \frac{(r')^2 - r^2 - (z - z')^2}{2r(r')^2},$$

$$\frac{\partial\chi}{\partial z'} = \frac{z' - z}{rr'},$$

$$\mathcal{Q}_{-1/2}(\chi) = \mu K(\mu),$$

$$\mathcal{Q}_{1/2}(\chi) = \chi\mu K(\mu) - \sqrt{2(\chi + 1)}E(\mu),$$

$$\mathcal{Q}_{-n-1/2}(\chi) = \mathcal{Q}_{n-1/2}(\chi),$$

$$\mathcal{Q}_{n-1/2}(\chi) = 4\frac{n - 1}{2n - 1}\chi\mathcal{Q}_{n-3/2}(\chi) - \frac{2n - 3}{2n - 1}\mathcal{Q}_{n-5/2}(\chi),$$

$$\frac{\partial\mathcal{Q}_{n-1/2}(\chi)}{\partial\chi} = \frac{2n - 1}{2(\chi^2 - 1)}\left(\chi\mathcal{Q}_{n-1/2} - \mathcal{Q}_{n-3/2} \right),$$

where K and E are the complete elliptic integrals of the first and second kinds, respectively. The first two relations follow immediately from the definition of χ and the relations for the Legendre functions of the second kind can be found in [1]. With these relations in hand, the calculation of $d_n^{(i)}$ for $n = -N_\mathrm{F}, -N_\mathrm{F} + 1, \ldots, N_\mathrm{F}$ can be done accurately and efficiently when r' and r as well as z' and z are in close proximity. The calculation of $d_n^{(e)}$ can be done analogously.

Remark 2 Note that the forward recursion relation for the Legendre functions $\mathcal{Q}_{n-1/2}(\chi)$ is unstable when $\chi > 1$. In practice, the instability is mild when χ is near 1 and the recursion relation can still be employed to accurately compute values in this regime. Additionally, if stability becomes an issue, Miller's algorithm [61] can be used to calculate the values of the Legendre functions using the backwards recursion relation, which is stable for $\chi > 1$.

5.6 Numerical Results for the Single Body Laplace Equation

This section describes several numerical experiments performed to assess the efficiency and accuracy of the the numerical scheme outlined in Section 5.4.1. All experiments were executed for the double layer kernels associated with Laplace's equation, calculated using the recursion relation described in Section 5.5.3. Note that the kernels in this case give us the property that $A_{-n} = A_n$, and so we need only to invert $N_\mathrm{F} + 1$ matrices. Further, the FFT used here is complex-valued, and a real-valued FFT would yield a significant decrease in computation time. The geometries investigated are described in Figure 5.2. The generating curves were parameterized by arc length, and split into N_P panels of equal length. A 10-point Gaussian quadrature has been used along each panel, with the modified quadratures used to handle the integrable singularities in the kernel. These quadratures are listed in the appendix. The algorithm was implemented in FORTRAN, using BLAS, LAPACK, and the FFT library provided by Intel's MKL library. All numerical experiments in this section have been carried out on a Macbook Pro with a 2.5 GHz Intel Core 2 Duo and 2GB of RAM.
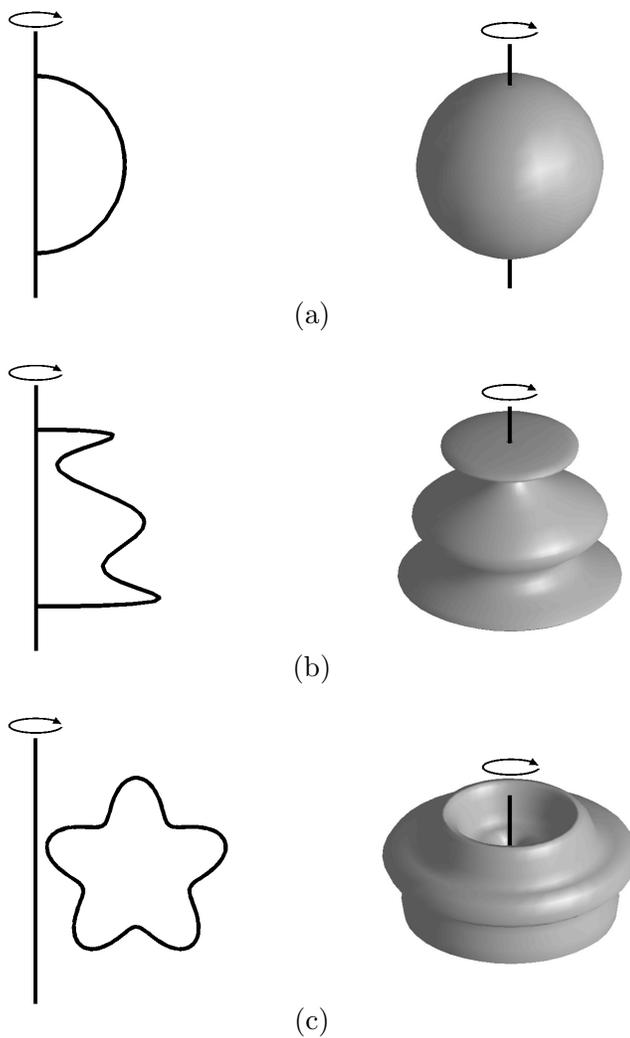
Figure 5.2: Domains used in numerical examples. All items are rotated about the vertical axis. (a) A sphere. (b) A wavy block. (c) A starfish torus.

5.6.1   Computational Costs

Using the domain in Figure 5.2(a) and the interior Dirichlet problem, timing results are given in Table 5.1. The reported results include:

N_P the number of panels used to discretize the contour

N_F the Fourier truncation parameter (we keep $2N_\mathrm{F} + 1$ modes)

T_mat time to construct the linear systems (utilizing the recursion relation)

T_inv time to invert the linear systems

T_fft time to Fourier transform the right hand side and the solution

T_apply time to apply the inverse to the right hand side

The most expensive component of the calculation is the construction of the linear systems and their inversion. This is primarily a result of the cost of evaluating the kernel and applying the modified quadrature rules, and the use of dense matrix algebra to invert the matrices. Table 5.2 compares the use of the recursion relation in evaluating the kernel when it is near-singular to using an adaptive Gaussian quadrature. The efficiency of the recursion relation is clearly evident in this case.

Figure 5.3 plots the time to construct the linear systems as the number of degrees of freedom $N = N_\gamma(2N_F + 1)$ increases, for the case when $N_\gamma \approx 2N_F + 1$. The estimated asymptotic costs given in this Figure match well with the estimates derived in Section 5.4.3. It is also clear that as N grows, the cost of inversion will eventually dominate. We remark that this cost can be greatly lowered by using fast techniques for the inversion of boundary integral operators, but the algorithm is quite fast for the problem sizes considered here.

We observe that the largest problem reported in Table 5.1 involves $320\,000$ degrees of freedom. The method requires 57 seconds of pre-computation for this example, and is then capable of computing a solution u from a given data function f in 0.39 seconds.

5.6.2   Accuracy and Conditioning of the Discretization

The accuracy of the discretization has been tested using the interior and exterior Dirichlet problems on the domains given in Figure 5.2. Exact solutions were generated by placing a few random point charges outside of the domain where the solution was calculated. The solution was evaluated at points defined on a sphere encompassing (or interior to) the boundary. The errors reported in Tables 5.3-5.5 are relative errors measured in the l^∞-norm, $||u_\epsilon - u||_\infty / ||u||_\infty$, where u

N_P	$2N_\mathrm{F}+1$	T_mat	T_inv	T_fft	T_apply
5	25	1.62946e-02	1.36140e-03	6.97000e-05	3.80000e-05
10	25	3.38219e-02	5.64690e-03	1.37100e-04	1.40100e-04
20	25	9.29729e-02	3.24038e-02	2.84600e-04	8.61000e-04
40	25	3.08036e-01	2.18462e-01	6.31300e-04	6.04500e-03
80	25	1.04748e+00	1.62193e+00	1.17040e-03	2.05759e-02
5	51	2.43118e-02	2.61800e-03	2.00400e-04	6.54000e-05
10	51	5.94078e-02	1.11537e-02	3.93100e-04	2.57300e-04
20	51	1.83887e-01	6.55384e-02	8.11200e-04	3.09950e-03
40	51	6.56846e-01	4.36035e-01	1.68690e-03	1.08002e-02
80	51	2.37916e+00	3.25536e+00	3.23780e-03	4.66582e-02
5	101	5.17718e-02	5.01950e-03	8.18000e-04	1.24700e-04
10	101	1.48422e-01	2.17932e-02	1.64440e-03	6.77100e-04
20	101	4.98615e-01	1.28403e-01	3.34810e-03	5.83030e-03
40	101	1.83856e+00	8.57550e-01	6.71140e-03	1.99509e-02
80	101	7.78611e+00	6.36054e+00	1.33346e-02	8.29963e-02
5	201	9.96437e-02	9.79930e-03	1.64870e-03	2.48900e-04
10	201	3.05792e-01	4.42244e-02	3.38140e-03	3.16150e-03
20	201	1.05618e+00	2.54758e-01	6.76460e-03	1.04266e-02
40	201	3.98176e+00	1.69139e+00	1.37381e-02	3.93247e-02
80	201	1.67122e+01	1.25839e+01	2.83274e-02	1.61082e-01
5	401	1.74281e-01	1.95047e-02	3.15610e-03	6.96600e-04
10	401	5.48732e-01	8.80375e-02	6.47000e-03	5.94290e-03
20	401	1.91440e+00	5.07339e-01	1.29201e-02	1.94150e-02
40	401	7.95480e+00	3.36656e+00	2.68350e-02	7.65999e-02
80	401	3.17200e+01	2.55822e+01	5.58125e-02	3.35029e-01

Table 5.1: Timing results in seconds performed for the domain given in Figure 5.2(a) for the interior Dirichlet problem.

$2N_\mathrm{F}+1$	Composite Quadrature	Recursion Relation
25	1.9	0.016
50	3.1	0.024
100	6.6	0.052
200	18.9	0.10

Table 5.2: Timing comparison in seconds for constructing the matrices $(I + A_n)$ using composite Gaussian quadrature and the recursion relation described in Section 5.5.3 to evaluate k_n for diagonal and near diagonal blocks. The FFT is used to evaluate k_n at all other entries. $2N_F + 1$ is the total number of Fourier modes used. 5 panels were used to discretize the boundary.

is the exact potential and u_ϵ is the potential obtained from the numerical solution.
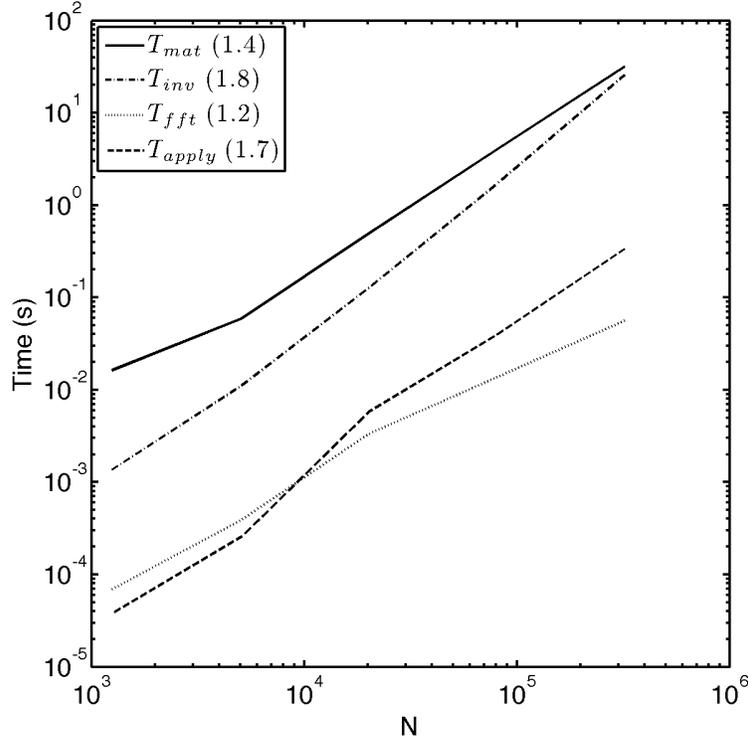
Figure 5.3: Timings of the algorithm as the number of degrees of freedom $N = N_\gamma(2N_F + 1)$ increases. The timings reported here are for the case $N_\gamma \approx 2N_F + 1$. The numbers in parentheses correspond to the scaling of the timings.

In all cases, 10 digits of accuracy has been obtained from a discretization involving a relatively small number of panels, due to the rapid convergence of the Gaussian quadrature. This is especially advantageous, as the most expensive component of the algorithm is the construction of the linear systems, the majority of the cost being directly related to the number of panels used. Further, the number of Fourier modes required to obtain 10 digits of accuracy is on the order of 100 modes. Although not investigated here, the discretization technique naturally lends itself to nonuniform refinement of the surface, allowing one to resolve features of the surface that require finer resolution.

The number of correct digits obtained as the number of panels and number of Fourier modes increases eventually stalls. This is a result of a loss of precision in determining the kernels, as well as cancelation errors incurred when evaluating interactions between nearby points. This is especially prominent with the use of Gaussian quadratures, as points cluster near the ends of the

panels. If more digits are required, high precision arithmetic can be employed in the setup phase of the algorithm.

N_P	$2N_F + 1$				
-	25	50	100	200	400
5	1.93869e-04	4.10935e-07	5.37883e-08	5.37880e-08	5.37880e-08
10	1.93869e-04	4.10513e-07	3.27169e-12	6.72270e-13	6.72270e-13
20	1.93869e-04	4.10513e-07	3.30601e-12	1.66132e-13	1.66132e-13
40	1.93869e-04	4.10513e-07	3.23162e-12	8.28568e-14	8.28568e-14
80	1.93869e-04	4.10512e-07	2.92918e-12	2.92091e-13	2.92091e-13

Table 5.3: Error in internal Dirichlet problem solved on domain (a) in Figure 5.2.

N_P	$2N_F + 1$				
-	25	50	100	200	400
5	9.11452e-04	9.11464e-04	9.11464e-04	9.11464e-04	9.11464e-04
10	4.15377e-05	4.15416e-05	4.15416e-05	4.15416e-05	4.15416e-05
20	6.31923e-07	1.29234e-07	1.29235e-07	1.29235e-07	1.29235e-07
40	7.04741e-07	3.10049e-11	3.08152e-11	3.08305e-11	3.08359e-11
80	7.04779e-07	5.62558e-11	5.05306e-11	5.05257e-11	5.05232e-11

Table 5.4: Error in external Dirichlet problem solved on domain (b) in Figure 5.2.

N_P	$2N_F + 1$				
-	25	50	100	200	400
5	3.80837e-04	3.83707e-04	3.83707e-04	3.83707e-04	3.83707e-04
10	2.41602e-05	6.81564e-06	6.81556e-06	6.81556e-06	6.81556e-06
20	3.03272e-05	5.98506e-09	2.53980e-11	2.54112e-11	2.54118e-11
40	3.03272e-05	6.01273e-09	6.95662e-12	6.94592e-12	6.94546e-12
80	3.03272e-05	6.01059e-09	5.25217e-12	5.26674e-12	5.26515e-12

Table 5.5: Error in external Dirichlet problem solved on domain (c) in Figure 5.2.

Figure 5.4 shows the singular values as well as the condition numbers of an 80 panel discretization for the $N_F = -200, \ldots, 200$ Fourier modes used in the discretization of the interior Dirichlet problem, on the domain shown in Figure 5.2(a). The integral equations of this chapter are second kind Fredholm equations, and generally lead to well-conditioned systems. As seen in
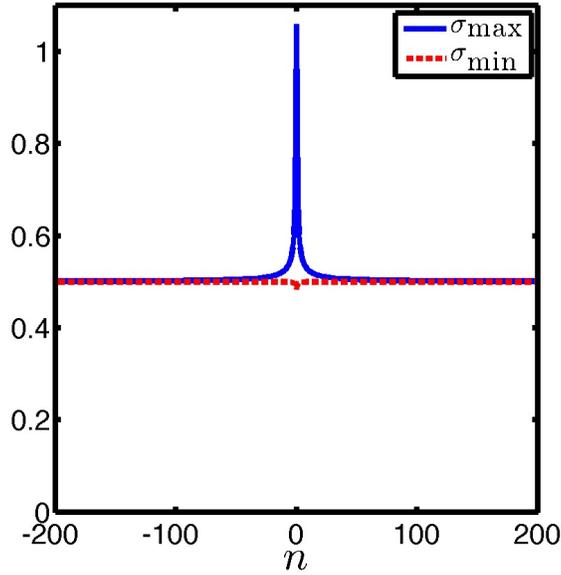
Figure 5.4: Maximum and minimum singular values for the matrices resulting from an 80 panel discretization of a sphere using 400 Fourier modes, where n is the the matrix associated with the nth Fourier mode.

Figure 5.4, this holds true for the discretization presented in this chapter.

5.7   Extension to Multiply Connected Domains

We have so far restricted ourselves to the solution of boundary integral equations defined on a single surface in space. In this section, we extend the results of the previous sections to multiply connected domains. Specifically, we utilize the high order Gaussian quadrature to discretize each surface, and then solve the system via an iterative solver that is accelerated with the Fast Multipole Method. We will restrict our attention to the exterior Dirichlet problem, but the treatment of other integral equations, boundary conditions, and domains is analogous.

5.7.1   Formulation

Multiply connected domains introduce an additional complication in obtaining the solution of Laplace's equation via a boundary integral approach. Denote the number of bodies defining the boundary Γ as N_B, so that $\Gamma = \Gamma_1 \cup \Gamma_2 \cup \ldots \cup \Gamma_{N_B}$, and assume that the bodies are well separated.

We will also assume that each of these bodies individually satisfies the axisymmetric constraint, but they need not be oriented in the same direction or along the same axis. Then the boundary integral operator defined by (5.43) has a nontrivial null space of dimension $N_B - 1$ [102]. A standard approach to remove this null space is provided by variants of Mikhlin's method, *c.f.* [64, 73, 155].

Alternatively, one can represent the solution to Laplace's equation on a multiply connected domain at a point $\boldsymbol{x}$ outside of the bodies by a combination of a single and a double layer [73],

$$u(\boldsymbol{x}) = \int_\Gamma \frac{1}{4\pi} \left(\frac{1}{|\boldsymbol{x} - \boldsymbol{x}'|} + \frac{\boldsymbol{n}(\boldsymbol{x}') \cdot (\boldsymbol{x} - \boldsymbol{x}')}{|\boldsymbol{x} - \boldsymbol{x}'|^3} \right) \sigma(\boldsymbol{x}')\, dA(\boldsymbol{x}').$$

The resulting boundary integral equation is given by

$$\frac{1}{2}\sigma(\boldsymbol{x}) + \int_\Gamma \frac{1}{4\pi} \left(\frac{1}{|\boldsymbol{x} - \boldsymbol{x}'|} + \frac{\boldsymbol{n}(\boldsymbol{x}') \cdot (\boldsymbol{x} - \boldsymbol{x}')}{|\boldsymbol{x} - \boldsymbol{x}'|^3} \right) \sigma(\boldsymbol{x}')\, dA(\boldsymbol{x}') = f(\boldsymbol{x}). \tag{5.47}$$

This is the approach we take, as it leads to particularly simple linear systems and is effective for a moderate number of bodies.

5.7.2 Construction of the Linear System

There is no straightforward way to decouple the multiply connected problem into a series of integral equations defined on a generating curve, as the axes of symmetry for each body are not necessarily aligned. Instead, we consider a Nyström discretization of the surfaces defining the boundary Γ. Let $K_{i,j}$ denote the interactions between the surfaces Γ_i and Γ_j. Then (5.47) can be written as

$$\frac{1}{2}\sigma I + \begin{pmatrix} K_{1,1} & K_{1,2} & \cdots & K_{1,N_B} \\ K_{2,1} & K_{2,2} & \cdots & K_{2,N_B} \\ \vdots & \vdots & \ddots & \vdots \\ K_{N_B,1} & K_{N_B,2} & \cdots & K_{N_B,N_B} \end{pmatrix} \begin{pmatrix} \sigma_1 \\ \sigma_2 \\ \vdots \\ \sigma_{N_B} \end{pmatrix} = \begin{pmatrix} f_1 \\ f_2 \\ \vdots \\ f_{N_B} \end{pmatrix}, \tag{5.48}$$

where σ_i and f_i are the charge potential and boundary conditions on the ith body. The blocks $K_{i,j}$ when $i \neq j$ represent integral operators with smooth kernels, and thus any standard quadrature can be used to discretize these blocks in (5.48). The natural choice in light of the discretization used

previously in this chapter is to discretize each body with a standard Gaussian quadrature along the body's generating curve, and to use the trapezoidal rule in the other direction. This gives a highly accurate representation of $K_{i,j}$ when $i \neq j$, and leads to $N = N_B N_\gamma (2N_F + 1)$ unknowns in the linear system, assuming each body is discretized with the same number of points.

Unfortunately, using this discretization on the diagonal blocks $K_{i,i}$, for $i = 1, \ldots, N_B$, does not represent the self interactions of the ith body in an accurate manner; the kernel is singular in this regime. To construct a high-order discretization of the diagonal blocks in (5.48), we will use the technique described in Section 5.3 coupled with the Fourier representation of the BIE given in Section 5.2. Let $\hat{\mathcal{K}}_{i,i}$ be the block-diagonal operator whose diagonal blocks are given by the operators $\mathcal{K}_n$ defined by (5.17), for the surface Γ_i. Then the action of the diagonal block $K_{i,i}$ in (5.48) can be represented by

$$K_{i,i} = \mathcal{F}_i^{-1} \hat{\mathcal{K}}_{i,i} \mathcal{F}_i, \tag{5.49}$$

where the periodic Fourier transform $\mathcal{F}_i : L^2(\mathbb{T}) \mapsto \ell^2(\mathbb{Z})$ maps a function's azimuthal dependence to its azimuthal Fourier coefficients. The expression given by (5.49) is discretized by the appropriate number of Fourier modes and panels to ensure that the nodes coincide with those resulting from the discretization of $K_{i,j}$ when $i \neq j$.

Note that the kernels in (5.49) can be calculated efficiently using the procedure described in Section 5.5.3.

5.7.3   Solution to the Linear System

In general, the linear system defined by (5.48) is of size $N = N_B N_\gamma (2N_F + 1)$, assuming each body is discretized with the same number of panels and Fourier modes, and is too large to be explicitly formed and inverted. Further, since the system of equations cannot be formulated in its entirety with the Fourier representation used when the domain is simply connected, we do not have access to the inverse of the matrix in (5.48) as given by (5.19).

However, because the off diagonal blocks in (5.48) are discretized by a conventional Nyström method in $\mathbb{R}^3$, this portion of the matrix can be applied to a vector with a conventional fast

multipole method in $O(N)$ operations, *c.f.* [31, 66].

The diagonal blocks of the matrix can be applied to a vector using (5.49). This costs $O(N_B N_\gamma^2 N_F \log N_F)$ operations. Asymptotically, this cost would dominate the cost of applying the matrix, but in practice, the high accuracy of the discretization results in a small number of unknowns on each body and as shown in Section 5.8, the FMM is the primary bottleneck in the application of the matrix.

With access to a fast matrix-vector multiply for applying the operator in (5.48), iterative methods become accessible for obtaining the solution. Further, since we are solving a second kind Fredholm equation containing a compact operator, the resulting discretized system will be well conditioned and an iterative solver like GMRES will converge in a small number of iterations (at least when the number of bodies is of moderate size).

5.7.4   An Algorithm for Multiply Connected Domains

A summary of the algorithm described in Sections 5.7.1-5.7.3 is given below:

(1) Given the right hand side f, and a computational tolerance ε, for each body $i = 1, \ldots, N_B$ determine a truncation parameter N_F such that (5.20) holds on that body.

(2) For each body $i = 1, \ldots, N_B$, form the block diagonal matrices as given by (5.49), using the techniques described in Section 5.4 and the accelerations given in Section 5.5.3 for calculating the kernels.

(3) Using an iterative solver appropriate for nonsymmetric systems, solve the system given by (5.48). Apply the off diagonal component of the matrix with the FMM, and apply the diagonal blocks directly as described in Section 5.7.3.

We make no claim that this algorithm is optimal for multiply connected domains. Certainly, the low-rank of the interactions between bodies could be further exploited by techniques such as skeletonization [95] to further increase the speed. However, we have found the algorithm described in this section is very simple to implement and effective enough for problems of reasonable size.

5.8   Numerical Results for Multiply Connected Domains

The algorithm described in Section 5.7.4 has been implemented in FORTRAN, using BLAS, LAPACK, and the FFT library provided by Intel's MKL library. The adaptive fast multiple algorithm described in [31] has been utilized in applying the off diagonal parts of a matrix, and GMRES is employed to solve the linear system of equations. All numerical experiments in this section have been carried out on a Macbook Pro with a 2.5 GHz Intel Core 2 Duo and 2GB of RAM.

Table 5.6 gives the results for the exterior Dirichlet problem for the domain given in Figure 5.5. The domain consists of 27 ellipses contained in the box $[0, 6.1] \times [0, 6.1] \times [0, 6.1]$, where each ellipse has a major axis of length 2, and a minor axis of length 1. The minimal distance between any two ellipses is 0.05. We solve the linear system via GMRES until the residual is less than 10^{-9}, applying the matrix as described in Section 5.7.3. We have employed the adaptive FMM described in [31], with 9 digits of accuracy requested. The boundary conditions are generated by placing point charges inside the ellipse, and the solution is evaluated at 1000 random points on a sphere encompassing the domain.

Note that this geometry consists of one surface that repeatedly translated and rotated throughout the domain. This allows us to construct the discretization representing the self interactions of an ellipse just once, and repeatedly use it in applying the block-diagonal of the matrix in (5.48), greatly reducing the memory requirements. The parameters in Table 5.6 are:
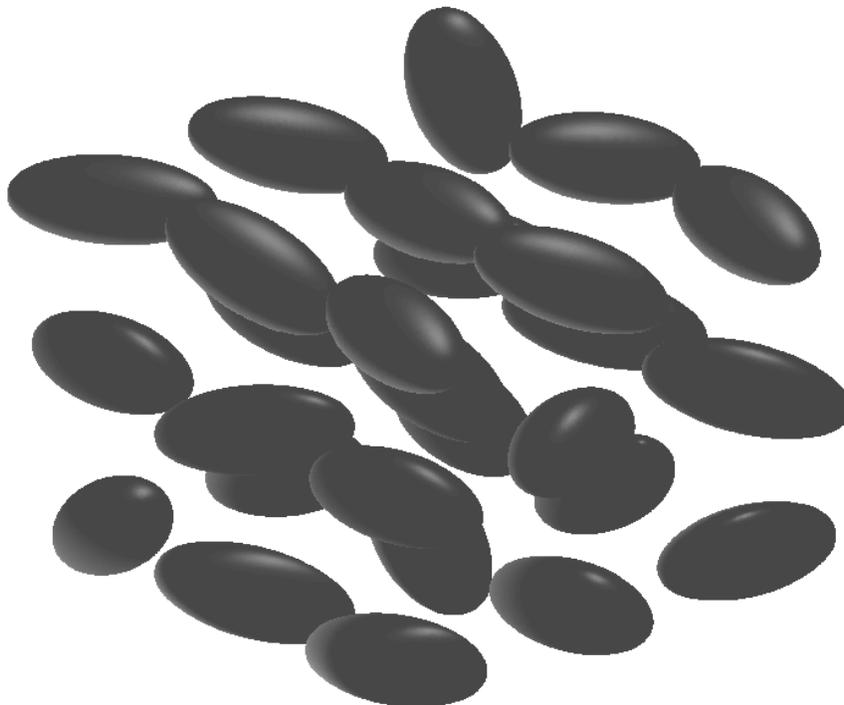
Figure 5.5: Randomly oriented ellipses contained in the box $[0, 6.1] \times [0, 6.1] \times [0, 6.1]$. Each ellipse has a major axis of length 2, and a minor axis of length 1. The minimal distance between any two ellipses is 0.05.

N the total number of unknowns in the system

N_{BODY} the number of points discretizing each body

T_{diag} time (in seconds) to apply the diagonal blocks to a vector

$T_{\text{off-diag}}$ time (in seconds) to apply the off diagonal blocks to a vector via the FMM

I_{GMRES} The number of GMRES iterations required to reduce the residual to 10^{-9}

T_{GMRES} time (in seconds) to solution via GMRES

Error   the maximal error in evaluating the solution at 1000 random points on a

            sphere encompassing the boundary

N	N_{BODY}	T_{setup}	T_{diag}	$T_{\text{off-diag}}$	I_{GMRES}	T_{GMRES}	Error
542700	20100	6.72e-01	1.43e-01	2.27e02	22	5.00e03	8.41e-11

Table 5.6: Results for the domain given in Figure 5.5 for the exterior Dirichlet problem.

Several comments are in order. We observe that the time to apply the block diagonal of the matrix is 3 orders of magnitude faster than the time to apply the off diagonal blocks via the FMM. This confirms the comments made in Section 5.7.3; for practical situations, the cost of the FMM is by far the dominate cost in the algorithm. The primary reason for this is the high accuracy of the discretization. We require only a small number of panels and Fourier modes to discretize the surface, and this results in diagonal blocks of small size. The small size of these matrices results in a rapid execution time. We remark that the cost of the FMM as implemented can be roughly halved. This is because the algorithm is called twice per iteration, once to apply the full matrix, and again to remove contributions from the diagonal blocks that it incorrectly calculated.

We have tested the algorithm on a variety of geometrical domains. The results are similar in nature; in general the number of GMRES iterations required mildly increases with the number of bodies and also increases if the bodies are in close proximity. Note that a finer discretization is required if the bodies are extremely close together.

5.9   Fast Kernel Evaluation for the Helmholtz Equation

The algorithm described in Section 5.4 applies to a wide variety of integral equations. The primary obstacle of this approach is the need to calculate the kernel of the reduced integral equation, which is a Fourier expansion of the original kernel in the azimuthal variable. For a general kernel, this can be done via a brute force approach, but this can be prohibitively expensive for large problem sizes. In Section 5.5, we described a procedure for rapidly computing the single and double layer kernels for Laplace's equation, and as demonstrated in Sections 5.6 and 5.7, the accelerated kernel evaluation brings large problem sizes into practical reach. In this section, we demonstrate how to extend this approach for a wide variety of kernels that are of practical importance, with a particular emphasis on the Helmholtz equation.

5.9.1 Rapid Kernel Calculation via Convolution

Consider a kernel of the form

$$f(\boldsymbol{x}, \boldsymbol{x}') = s(\boldsymbol{x}, \boldsymbol{x}')\, g(\boldsymbol{x}, \boldsymbol{x}'), \tag{5.50}$$

where

$$s(\boldsymbol{x}, \boldsymbol{x}') = \frac{1}{4\pi |\boldsymbol{x} - \boldsymbol{x}'|}$$

is the single layer kernel of Laplace's equation and $g(\boldsymbol{x}, \boldsymbol{x}')$ is a smooth function for all $\boldsymbol{x}, \boldsymbol{x}' \in \mathbb{R}^3$. Common examples of kernels that take this form include the fundamental solution of the Helmholtz equation and screened Coulomb (Yukawa) potentials.

Letting

$$\boldsymbol{x} = (r \cos\theta,\, r \sin\theta,\, z),$$

$$\boldsymbol{x}' = (r' \cos\theta',\, r' \sin\theta',\, z'),$$

we are interested in calculating the Fourier expansion of (5.50) in terms of the azimuthal variable. When $g(\boldsymbol{x}, \boldsymbol{x}') = 1$ (the Laplace kernel), we know how to rapidly compute these Fourier coefficients rapidly and efficiently. However, when g takes a nontrivial form, this is not generally true; there is no known analytical formula for calculating the Fourier coefficients of (5.50).

We will now describe an efficient technique for calculating the the Fourier coefficients of (5.50), when the function g is sufficiently smooth. For a fixed value of (r, z) and (r', z'), the functions s and g are periodic in the azimuthal variable over the interval $\mathbb{T}$. Dropping the dependence of s and g on (r, z) and (r', z') for notational clarity, we define $t = \theta - \theta' \in \mathbb{T}$ and the Fourier series expansions of s and g as

$$s(t) = \sum_{n \in \mathbb{Z}} \frac{e^{int}}{\sqrt{2\pi}}\, s_n, \tag{5.51}$$

$$g(t) = \sum_{n \in \mathbb{Z}} \frac{e^{int}}{\sqrt{2\pi}}\, g_n, \tag{5.52}$$

where

$$s_n = \int_{\mathbb{T}} \frac{e^{-int}}{\sqrt{2\pi}}\, s(t)\, dt, \qquad (5.53)$$

$$g_n = \int_{\mathbb{T}} \frac{e^{-int}}{\sqrt{2\pi}}\, g(t)\, dt. \qquad (5.54)$$

The values given by (5.53) can be calculated as described in Section 5.5, while the values given by (5.54) can be rapidly and accurately computed using the FFT.

Assuming the Fourier series defined by (5.51) and (5.52) are uniformly convergent, we have that

$$\begin{aligned}
f_n &= \frac{1}{\sqrt{2\pi}} \int_{\mathbb{T}} s(t)\, g(t)\, e^{-int}\, dt \\
&= \frac{1}{2\pi} \sum_{k\in\mathbb{Z}} \sum_{\ell\in\mathbb{Z}} \int_{\mathbb{T}} s_k\, g_\ell\, e^{i(k+\ell-n)}\, dt \\
&= \sum_{k\in\mathbb{Z}} s_k\, g_{n-k} \\
&= s_n * g_n,
\end{aligned}$$

where $s_n * g_n$ is the discrete convolution of the sequences defined by (5.53) and (5.54).

In a practical setting, the Fourier series are finite in length. Assuming that we have kept $-N_F, -N_F+1, \ldots, N_F+1$ terms, directly calculating the convolution would require $O(N_F^2)$ operations. Fortunately, this computation can be accelerated to $O(N_F \log N_F)$ operations by employing the discrete convolution theorem and the FFT [21].

Letting $\mathcal{D}$ denote the discrete Fourier transform (DFT), the discrete convolution theorem states that the convolution of two periodic sequences $\{a_n\}$ and $\{b_n\}$ is related by

$$\mathcal{D}\{a_n * b_n\}_k = \alpha A_k B_k,$$

where $\{A_n\} = \mathcal{D}\{a_n\}$, $\{B_n\} = \mathcal{D}\{b_n\}$, and α is a known constant depending upon the length of the periodic sequences and the definition of the DFT. Thus, we can rapidly calculate the convolution of two periodic sequences by taking the FFT of each sequence, computing the pointwise product of the result, and then applying the inverse FFT.

Of course, the sequences that we need to convolute are not periodic. Applying the discrete convolution to the sequences defined by (5.53) and (5.54) will not be exact, but the error incurred will be small assuming that the Fourier coefficients decay rapidly and that N_F is large enough. To see this, assume that

$$s(t) = \sum_{n=-N_F}^{N_F} \frac{e^{int}}{\sqrt{2\pi}} s_n,$$

$$g(t) = \sum_{n=-N_F}^{N_F} \frac{e^{int}}{\sqrt{2\pi}} g_n.$$

Then the exact Fourier representation of f can be found by taking the product of these two series, which will be of length $4N_F + 1$. As is well known, the coefficients of this product is given by the discrete convolution of the sequences containing the coefficients of the two series, and these sequences must first be padded with $2N_F$ zeros. Thus, we can effectively calculate the Fourier coefficients of the function given by (5.50) by calculating $2N_F + 1$ Fourier coefficients of s and g, padding these sequences with zeros, calculating the discrete convolution of these sequences, and truncating the resulting sequence. In practice, padding may not even be required if the Fourier coefficients of s and g decay sufficiently fast.

Note that the procedure described in this section is quite general. The azimuthal Fourier coefficients of any kernel that can be represented as the product of a singular function and a smooth function can be found, assuming that there is an accurate technique for determining the coefficients of the singular function.

5.9.2   Application to the Helmholtz Equation

In this section, we will apply the fast kernel calculation technique described in Section 5.9 to the exterior Dirichlet problem for the Helmholtz equation. Let $D \subset \mathbb{R}^3$ be a bounded domain whose boundary is given by a smooth surface Γ, let $E = \bar{D}^c$ denote the domain exterior to D, and let $\boldsymbol{n}$ and be the outward unit normal to D. The partial differential equation representing this problem is given by

$$\Delta u + k^2 u = 0 \ \text{ in } E, \quad u = f \ \text{ on } \Gamma, \tag{5.55}$$

where $k > 0$ is the wavenumber, and u satisfies the Sommerfeld radiation condition

$$\lim_{r\to\infty} r \left(\frac{\partial u}{\partial r} - i\,k\,u \right) = 0,$$ (5.56)

where $r = |\boldsymbol{x}|$ and the limit holds uniformly in all directions $\boldsymbol{x}/|\boldsymbol{x}|$. Let the single and double layer potentials for the Helmholtz equation be given by

$$\phi(\boldsymbol{x}, \boldsymbol{x}') = \frac{e^{ik|\boldsymbol{x}-\boldsymbol{x}'|}}{4\pi|\boldsymbol{x}-\boldsymbol{x}'|}, \qquad \text{(single layer)}$$ (5.57)

$$\frac{\partial\phi(\boldsymbol{x}, \boldsymbol{x}')}{\partial\boldsymbol{n}(\boldsymbol{x}')} = \frac{\boldsymbol{n}(\boldsymbol{x}')\cdot(\boldsymbol{x}-\boldsymbol{x}')}{4\pi|\boldsymbol{x}-\boldsymbol{x}'|^3} \left[\left(1 - ik|\boldsymbol{x}-\boldsymbol{x}'|\right) e^{ik|\boldsymbol{x}-\boldsymbol{x}'|} \right]. \qquad \text{(double layer)}$$ (5.58)

The solution to (5.55) can be written in in terms of the double layer potential,

$$u(\boldsymbol{x}) = \int_\Gamma \frac{\partial\phi(\boldsymbol{x}, \boldsymbol{x}')}{\partial\boldsymbol{n}(\boldsymbol{x}')} \sigma(\boldsymbol{x}')\, dA(\boldsymbol{x}'), \quad \boldsymbol{x} \in E,$$

where σ is a boundary charge density that can be determined using the boundary conditions. The resulting boundary integral equation is given by

$$\frac{1}{2}\sigma(\boldsymbol{x}) + \int_\Gamma \frac{\partial\phi(\boldsymbol{x}, \boldsymbol{x}')}{\partial\boldsymbol{n}(\boldsymbol{x}')} \sigma(\boldsymbol{x}')\, dA(\boldsymbol{x}') = f(\boldsymbol{x}).$$ (5.59)

As is well known, (5.59) is not always uniquely solvable, even though (5.55) is uniquely solvable for all $k > 0$. A common solution to this is to represent the the solution to (5.55) as a combined single and double layer potential,

$$u(\boldsymbol{x}) = \int_\Gamma \left(\frac{\partial\phi(\boldsymbol{x}, \boldsymbol{x}')}{\partial\boldsymbol{n}(\boldsymbol{x}')} - i\,\nu\,\phi(\boldsymbol{x}, \boldsymbol{x}') \right) \sigma(\boldsymbol{x}')\, dA(\boldsymbol{x}'), \quad \boldsymbol{x} \in E,$$

where $\nu > 0$. We have freedom in choosing ν, see *e.g.*, [22, 84], for some analysis on this choice. The boundary integral equation we need to solve is given by

$$\frac{1}{2}\sigma(\boldsymbol{x}) + \int_\Gamma \left(\frac{\partial\phi(\boldsymbol{x}, \boldsymbol{x}')}{\partial\boldsymbol{n}(\boldsymbol{x}')} - i\,\nu\,\phi(\boldsymbol{x}, \boldsymbol{x}') \right) \sigma(\boldsymbol{x}')\, dA(\boldsymbol{x}') = f(\boldsymbol{x}).$$ (5.60)

5.9.3   Numerical Results for the Single Body Helmholtz Equation

In this section, we present numerical solutions for the Helmholtz equation on exterior domains and with Dirichlet boundary conditions. We use the formulation described in Section 5.9.2 to

reformulate the problem as a boundary integral equation. The algorithm employed to solve the integral equation is the same as described in Section 5.4, with the caveat that the kernels are calculated using the fast procedure described in Section 5.9.1.

The algorithm has been implemented in FORTRAN, using BLAS, LAPACK, and the FFT library provided by Intel's MKL library. All numerical experiments in this section have been carried out on a Macbook Pro with a 2.5 GHz Intel Core 2 Duo and 2GB of RAM.

Table 5.7 presents timing results for the algorithm, and include:

N_{P} the number of panels used to discretize the contour

N_{F} the Fourier truncation parameter (we keep $2N_{\mathrm{F}} + 1$ modes)

T_{mat} time to construct the linear systems (utilizing the recursion relation)

T_{inv} time to invert the linear systems

T_{fft} time to Fourier transform the right hand side and the solution

T_{apply} time to apply the inverse to the right hand side

The largest problem size considered here has 40 panels and 201 Fourier modes, leading to $160\,800$ unknowns discretizing the surface. Note that this problem size is slightly smaller than the largest considered in Section 5.6, due to memory constraints. This is because the matrices now contain complex entries, and thus use twice the memory compared with the Laplace case. The total running time of the algorithm is 92 seconds for this problem size. If given additional right hand sides, we can solve them in 0.56 seconds. We also remark that the asymptotic scaling of the cost of this algorithm is identical to that of the Laplace case, it simply takes more operations (roughly twice as many) to calculate the kernels and perform the required matrix operations.

We have assessed the accuracy of the algorithm for various domains, discretization parameters, and Fourier modes. The boundary conditions used are determined by placing point charges inside the domains, and we evaluate the solution at random points placed on a sphere that encompasses the boundary. We have set the parameter $\nu = k$. We note that this parameter is usually chosen to improve the conditioning of the system, but this is less of a concern for us, as we are

N_{P}	$2N_{\mathrm{F}}+1$	T_{mat}	T_{inv}	T_{fft}	T_{apply}
5	25	3.81e-02	5.73e-03	1.99e-04	1.00e-04
10	25	1.41e-01	1.78e-02	3.65e-04	3.39e-04
20	25	3.34e-01	1.18e-01	7.65e-04	3.72e-03
40	25	1.19e+00	8.47e-01	1.54e-03	1.23e-02
80	25	4.32e+00	6.45e+00	3.03e-03	6.51e-02
5	51	7.07e-02	5.98e-03	4.10e-04	1.78e-04
10	51	1.76e-01	3.53e-02	8.37e-04	8.36e-04
20	51	6.59e-01	2.36e-01	1.74e-03	6.88e-03
40	51	2.39e+00	1.70e+00	3.44e-03	2.36e-02
80	51	9.14e+00	1.30e+01	6.86e-03	1.21e-01
5	101	1.52e-01	1.16e-02	1.33e-03	3.53e-04
10	101	4.72e-01	6.95e-02	2.71e-03	3.63e-03
20	101	1.60e+00	4.61e-01	5.44e-03	1.22e-02
40	101	5.86e+00	3.32e+00	1.10e-02	4.51e-02
80	101	2.21e+01	2.53e+01	2.19e-02	2.61e-01
5	201	2.70e-01	2.30e-02	2.28e-03	8.04e-04
10	201	8.54e-01	1.38e-01	4.63e-03	6.81e-03
20	201	2.92e+00	9.13e-01	9.41e-03	2.92e-02
40	201	1.12e+01	6.58e+00	1.94e-02	8.81e-02
80	201	4.11e+01	5.02e+01	3.83e-02	5.18e-01
5	401	5.54e-01	4.62e-02	5.06e-03	3.77e-03
10	401	1.76e+00	2.74e-01	1.01e-02	1.23e-02
20	401	6.16e+00	1.81e+00	2.10e-02	4.43e-02
40	401	2.33e+01	1.31e+01	4.23e-02	1.74e-01

Table 5.7: Timing results in seconds performed for a spherical domain.

employing a direct solver in solving the linear systems.

N_{P}	$2N_{\mathrm{F}}+1$				
-	25	51	101	201	401
5	1.40471e-03	2.51347e-05	8.85603e-08	1.00143e-07	1.00144e-07
10	1.74067e-03	3.84922e-05	3.04542e-08	1.60822e-10	1.60840e-10
20	1.34819e-03	2.33130e-05	1.38725e-08	1.84954e-10	1.84947e-10
40	1.34590e-03	2.45615e-05	1.69741e-08	1.92141e-10	1.92131e-10
80	1.33110e-03	2.45351e-05	1.95272e-08	2.07781e-10	-

Table 5.8: Relative error in external Helmholtz problem for the domain in Figure 5.6(a). The domain is 1 wavelength in length (the major axis).

First, we consider the ellipsoidal domain given in Figure 5.6(a). The major axis of this

(a)

(b)
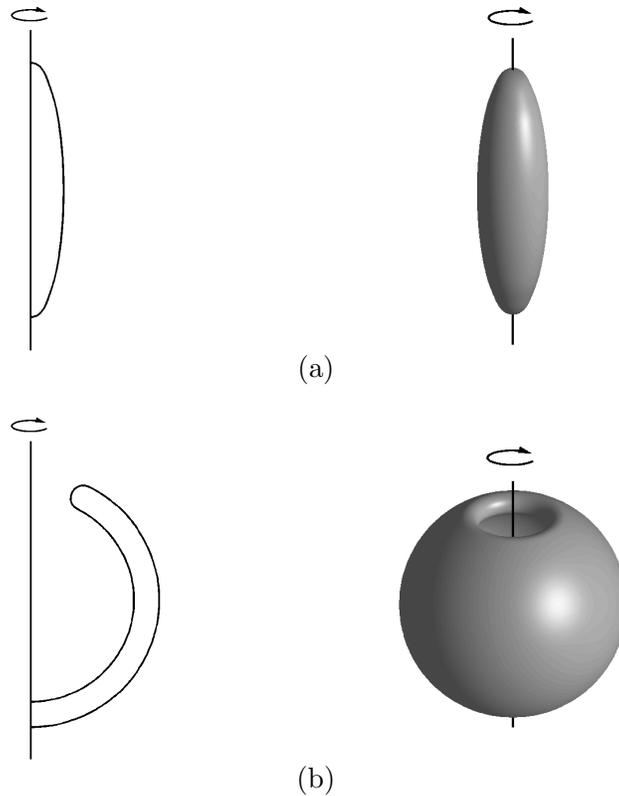
Figure 5.6: Domains used in numerical examples for the Helmholtz problem. All items are rotated about the vertical axis. (a) An ellipse. (b) A bowl shaped cavity.

N_P	$2N_\mathrm{F} + 1$				
-	25	51	101	201	401
5	5.29342e+00	5.75699e+00	5.78126e+00	5.77903e+00	5.78141e+00
10	1.49940e+00	1.58969e-02	1.20480e-02	1.20492e-02	1.20492e-02
20	1.51959e+00	1.62647e-02	1.21925e-07	5.73071e-08	5.73070e-08
40	1.55201e+00	1.58736e-02	9.27649e-08	8.83812e-09	8.83812e-09
80	1.54093e+00	1.63505e-02	9.97691e-08	9.13593e-09	-

Table 5.9: Relative error in external Helmholtz problem for the domain in Figure 5.6(a). The domain is 25 wavelengths in length (the major axis).

ellipse has a diameter of 2, and its minor axes have a diameter of 1/2. Table 5.8 lists the accuracy

achieved. We achieve 10 digits of accuracy in this problem with 10 panels and 201 Fourier modes. As

implemented, the algorithm generally requires more discretization points in the azimuthal direction

than along the generating curve. This is because the accuracy of the kernel calculations is dependent

upon the number of Fourier modes used. We have not padded the two sequences, but as more modes are used the tails of the sequence rapidly approach zero, increasing the accuracy of the convolution algorithm utilized to calculate the kernels. Table 5.9 displays the same data, but with the wavenumber increased so that there is 25 wavelengths along the length of the ellipsoid. We see a minor decrease in accuracy as we would expect, but it only takes 20 panels to achieve 8 digits of accuracy.

N_P	$2N_F + 1$				
-	25	51	101	201	401
5	8.04443e+00	2.86947e+01	1.51245e+01	1.51155e+01	1.51197e+01
10	5.69627e+00	1.23791e+00	2.34526e-02	6.84333e-03	6.84330e-03
20	5.71855e+00	1.25654e+00	2.39729e-02	1.27589e-05	1.27591e-05
40	6.05163e+00	1.26475e+00	2.26774e-02	1.25417e-08	1.27665e-09
80	5.94335e+00	1.25812e+00	2.30597e-02	1.22779e-08	-

Table 5.10: Relative error in external Helmholtz problem for the domain in Figure 5.6(b). The domain is 10 wavelengths in length (the major axis).

We now consider the more complex domain given in Figure 5.6(b). This is a spherical shaped cavity with the outer diameter of size 2. The accuracy for various values of N_P and N_F are given in Table 5.10. We achieve 8 digits of accuracy with 40 panels; this is expected as the length of the contour is roughly twice that of the contour in Figure 5.6(a). Note that we would expect a geometry such as this to experience near resonance at certain frequencies. Any iterative solver would suffer in this regime, but the natural inversion of the operator given by the discretization circumvents such issues.

5.10   Generalizations and Conclusions

This chapter describes a numerical technique for computing solutions to boundary integral equations defined on axisymmetric surfaces in $\mathbb{R}^3$ with no assumption on the loads being axisymmetric. The technique is introduced as a generic method with only very mild conditions imposed on the kernel; specifically, we assume that the kernel has an integrable singularity at the diagonal, and

that it is rotationally symmetric (in the sense that (5.7) holds). The technique described improves upon previous work in several regards:

(1) A highly accurate quadrature scheme for kernels with integrable singularities is introduced. Numerical experiments indicate that solutions with a relative accuracy of 10^{-10} or better can easily be constructed.

(2) A rapid technique for numerically constructing the kernel functions k_n in (5.3) is introduced. It works when k is either the single or the double layer potential associated with Laplace's equation. The technique is a hybrid scheme that relies on the FFT when possible, and uses recursion relations for Legendre functions when not. The resulting scheme is fast enough that a problem involving $320\,000$ degrees of freedom can be solved in 2.2 minutes on a standard desktop PC. Once one problem has been solved, additional right hand sides can be processed in 0.46 seconds.

(3) The extension of the axisymmetric methodology to multibody problems. Each body is discretized using the high order Nyström discretization developed for simply connected surfaces that are axisymmetric. The resulting matrix can then be applied efficiently to a vector using a combination of the fast multipole method to apply the interactions between bodies, and the fast Fourier transform can be used in applying self body interactions. This allows for the solution of the linear system via an appropriate iterative solver. We note that the primary bottleneck in this approach is in fact the FMM - the axisymmetric methodology allows for extremely rapid application of the self interactions of bodies to a vector.

(4) A rapid technique for evaluating kernels associated with the Helmholtz equation. This technique also extends to a general class of kernels that are of similar form to the Helmholtz kernels. We have found that for a problem size with $160\,800$ degrees of freedom, the initial setup and solve takes 92 seconds, while additional solves can be processed in 0.56 seconds.

Some possible extensions of this work include:

(1) Application to the Helmholtz equation defined on multiply connected domains and high frequencies. Note that the framework for such problems has already been developed in this chapter. However, these problems can be difficult to solve via iterative methods. We remark that a natural block-diagonal preconditioner arises from the axisymmetric methodology which may prove advantageous in accelerating the convergence of the iterative solver.

(2) Acceleration of the solution of the discretized systems via fast methods. Specifically, the matrices arising from the discretization presented here fall into the class of hierarchically semi-separable matrices, and recently developed fast algorithms for this class of matrices can be exploited, see *e.g.*, [95].

(3) The extension of the algorithms presented in the chapter to other integral equations, including the Stokes equations and the time harmonic Maxwell equations. A particularly appealing direction would be numerically exploring a new integral formulation for the time harmonic Maxwell equations introduced in [48], which appears to have many advantages over classical approaches.

Bibliography

[1] M. Abramowitz and I.A. Stegun, Handbook of mathematical functions with formulas, graphs, and mathematical tables, Dover, New York, 1965.

[2] Ulf Andersson, Björn Engquist, Gunnar Ledfelt, and Olof Runborg, A contribution to wavelet-based subgrid modeling, Appl. Comput. Harmon. Anal., 7 (1999), pp. 151–164.

[3] K. Atkinson, The numerical solution of integral equations of the second kind, Cambridge University Press, Cambridge, 1997.

[4] V. Bahadur and S.V. Garimella, An energy-based model for electrowetting-induced droplet actuation, Journal of Micromechanics and Microengineering, 11 (2006), pp. 1994–1503.

[5] E. Baird and K. Mohseni, Digitized heat transfer: A new paradigm for thermal management of compact micro-systems, IEEE Transactions on Components and packaging Technologies, 31 (2008), pp. 143–151.

[6] E. Baird, P. Young, and K. Mohseni, Electrostatic force calculation for an EWOD-actuated droplet,, Microfluid Nanofluid, 3 (2007), p. 635644.

[7] N. Bakhvalov and G. Panasenko, Homogenisation: averaging processes in periodic media, vol. 36 of Mathematics and its Applications (Soviet Series), Kluwer Academic Publishers Group, Dordrecht, 1989. Mathematical problems in the mechanics of composite materials, Translated from the Russian by D. Leĭtes.

[8] A.A. Bakr, The boundary integral equation method in axisymmetric stress analysis problems, Springer-Verlag, Berlin, 1985.

[9] J. Barnes and P. Hut, A hierarchical $o(n \log n)$ force-calculation algorithm, Nature, 324 (1986).

[10] A. Bejan, Convection Heat Transfer, John Wiley & Sons, Inc, 3rd ed., 2004.

[11] R.P. Beyer and R.J. LeVeque, Analysis of a one-dimensional model for the immersed boundary method, SIAM J. Num. Anal., 29 (1992), pp. 332–364.

[12] G. Beylkin, R. Coifman, and V. Rokhlin, Wavelets in numerical analysis, in Wavelets and their applications, Jones and Bartlett, Boston, MA, 1992, pp. 181–210.

[13] S. Bobbio, Electrodynamics of materials: forces, stresses, and energies in solids and fluids, Academic Press, 2000.

[14] S. Börm, $\mathcal{H}^2$-matrix arithmetics in linear complexity, Computing, 77 (2006), pp. 1–28.

[15] ———, Approximation of solution operators of elliptic partial differential equations by $\mathcal{H}$-and $\mathcal{H}^2$–matrices, Tech. Report 85/2007, Max Planck Institute, 2007.

[16] ———, Construction of data-sparse $\mathcal{H}^2$-matrices by hierarchical compression, Tech. Report 92/2007, Max Planck Institute, 2007.

[17] J. Bremer and V. Rokhlin, Efficient discretization of laplace boundary integral equations on polygonal domains, J. Comput. Phys., 229 (2010), pp. 2507–2525.

[18] J. Bremer and V. Rokhlin, Efficient discretization of Laplace boundary integral equations on polygonal domains, J. Comput. Phys., 229 (2010), pp. 2507–2525.

[19] J. Bremer, V. Rokhlin, and I. Sammis, Universal quadratures for boundary integral equations on two-dimensional domains with corners, Tech. Report TR-1420, Yale University, Department of Computer Science, 2009.

[20] M. E. Brewster and G. Beylkin, A multiresolution strategy for numerical homogenization, Appl. Comput. Harmon. Anal., 2 (1995), pp. 327–349.

[21] B. Briggs and V.E. Henson, The DFT: An Owner's Manual for the Discrete Fourier Transform, SIAM, Philadelphia, 1995.

[22] O.P. Bruno and L.A. Kunyansky, A fast, high-order algorithm for the solution of surface scattering problems: basic implementation, test, and applications, J. Comput. Phys., 169 (2001), pp. 80–110.

[23] J Buehrle, S Herminghaus, and F Mugele, Interface profiles near three-phase contract lines in electric fields, Phys Rev Lett, 91 (2003), pp. (086101)1–4.

[24] S. Chandrasekaran and M. Gu, Fast and stable algorithms for banded plus semiseparable systems of linear equations, SIAM J. Matrix Anal. Appl., 25 (2003), pp. 373–384 (electronic).

[25] S. Chandrasekaran, M. Gu, X.S. Li, and J. Xia, Some fast algorithms for hierarchically semiseparable matrices, Tech. Report 08-24, UCLA/CAM, 2008.

[26] ———, Superfast multifrontal method for structured linear systems of equations, SIAM J. Matrix Anal. Appl., 31 (2009), pp. 1382 – 1411.

[27] S. Chandrasekaran, M. Gu, and W. Lyons, A fast adaptive solver for hierarchically semiseparable representations, Calcolo, 42 (2005), pp. 171–185.

[28] Y-J. Chang, K. Mohseni, and V. Bright, Fabrication of tapered SU-8 structure and effect of sidewall angle for a variable focus microlens using EWOD, Sensors and Actuators A, 136 (2007), pp. 546–553.

[29] K. Charkrabarty and J. Zeng, Design Automation Methods and Tools for Microfluidics-Based Biochips, Springer, 2006.

[30] E.W. Cheney and W.A. Light, A Course in Approximation Theory, Brooks/Cole, 2000.

[31] H. Cheng, L. Greengard, and V. Rokhlin, A fast adaptive multipole algorithm in three dimensions, J. Comput. Phys., 155 (1999), pp. 468–498.

[32] S.K. Cho, H. Moon, and C.J. Kim, Creating, transporting, cutting, and merging liquid droplets by electrowetting-based actuation for digital microfluidic circuits, J. MEMS, 12 (2003), pp. 70–80.

[33] R.C. Chu, R.E. Simons, M.J. Ellsworth, R.R. Schmidt, and V. Cozzolino, Review of cooling technologies for computer products, IEEE Transactions on Device and Materials Reliability, 4 (2005), pp. 568– 585.

[34] Doina Cioranescu and Jeannine Saint Jean Paulin, Homogenization of reticulated structures, vol. 136 of Applied Mathematical Sciences, Springer-Verlag, New York, 1999.

[35] H.S. Cohl and J.E. Tohline, A compact cylindrical green's function expansion for the solution of potential problems, Astrophys. J., 527 (1999), pp. 86–101.

[36] T. Colonius and K. Taira, A fast immersed boundary method using a nullspace approach and multi-domain far-field boundary conditions, Comput. Method. Appl. M., 197 (2008), pp. 2131–2146.

[37] C.G. Cooney, C-Y. Chen, A. Nadim, and J.D. Sterling, Electrowetting droplet microfluidics on a single planar surface, Microfluid Nanofluid, 2 (2006), pp. 435–446.

[38] A. Darhuber and S.M. Troian, Principles of microfluidic actuation by modulation of surface stresses, Annu. Rev. Fluid Mech., 37 (2005), pp. 425–455.

[39] A.A. Darhuber, J.P. Valentino, S.M. Troian, and S. Wagner, Microfluidic actuation by modulation of surface stresses, Appl. Phys. Lett., 82 (2003), pp. 657–.

[40] J. Deval, P. Tabeling, and C.M. Ho, A dielectrophoretic chaotic mixer, in Technical Digest (ISBN-0-7803-7187-9), of the 15th IEEE International Conference on MEMS (MEMS 2002), Las Vegas, Nevada, 2002, pp. 36–39.

[41] Patrick Dewilde and Shivkumar Chandrasekaran, A hierarchical semi-separable Moore-Penrose equation solver, in Wavelets, multiscale systems and hypercomplex analysis, vol. 167 of Oper. Theory Adv. Appl., Birkhäuser, Basel, 2006, pp. 69–85.

[42] A. Dolatabadi, K. Mohseni, and A. Arzpeyma, Behaviour of a moving droplet under electrowetting actuation: Numerical simulation, Canadian J. of Chemical Engineering, 84 (2006), pp. 17–21.

[43] Mihai Dorobantu and Björn Engquist, Wavelet-based numerical homogenization, SIAM J. Numer. Anal., 35 (1998), pp. 540–559 (electronic).

[44] T.A. Driscoll and B. Fornberg, Interpolation in the limit of increasingly flat radial basis functions, Comput. Math. Appl., 43 (2002), pp. 413–422.

[45] Yalchin Efendiev and Thomas Y. Hou, Multiscale finite element methods, vol. 4 of Surveys and Tutorials in the Applied Mathematical Sciences, Springer, New York, 2009. Theory and applications.

[46] BJÖRN ENGQUIST AND OLOF RUNBORG, Wavelet-based numerical homogenization with applications, in Multiscale and multiresolution methods, vol. 20 of Lect. Notes Comput. Sci. Eng., Springer, Berlin, 2002, pp. 97–148.

[47] ——, Wavelet-based numerical homogenization, in Highly oscillatory problems, vol. 366 of London Math. Soc. Lecture Note Ser., Cambridge Univ. Press, Cambridge, 2009, pp. 98–126.

[48] C. EPSTEIN AND L. GREENGARD, Debye sources and the numerical solution of the time harmonic Maxwell equations, Comm. Pure Appl., 63 (2009), pp. 413–463.

[49] R.B. FAIR, V. SRINIVASAN, H. REN, P. PAIK, AND M.G. POLLACK, Electrowetting-based on-chip sample processing for integrated microfluidics, in IEEE Inter. electron devices meeting(IEDM), 2003.

[50] JACOB FISH AND AMIR WAGIMAN, Multiscale finite element method for a locally nonperiodic heterogeneous medium, Computational Mechanics, 12 (1993), pp. 164–180. 10.1007/BF00371991.

[51] J.L. FLEMING, A.W. WOOD, AND W.D. WOOD JR., Locally corrected nyström method for em scattering by bodies of revolution, J. Comput. Phys., 196 (2004), pp. 41–52.

[52] J. FLUERIAL, A. BORSHCEVSKY, M. RYAN, W. PHILLIPS, E. KOLAWA, T. KACISCH, AND R. EWELL, Thermoelectric microcoolers for thermal management applications, in Proceeding of the IEEE Conference on Thermoelectrics, 1997, pp. 641–645.

[53] B. FORNBERG, E. LARSSON, AND N. FLYER, Stable computations with Gaussian radial basis functions in 2-D, Tech. Report 2009-020, Uppsala University, Department of Information Technology, 2009.

[54] B. FORNBERG AND G. WRIGHT, Stable computation of multiquadric interpolants for all values of the shape parameter, Comput. Math. Appl., 48 (2004), pp. 853–867.

[55] B. FORNBERG, G. WRIGHT, AND E. LARSSON, Some observations regarding interpolants in the limit of flat radial basis functions, Comput. Math. Appl., 47 (2004), pp. 37–55.

[56] B.S. GALLARDO, V.K. GUPTA, AND F.D. EAGERTON et al., Electrochemical principles for active control of liquids on submillimeter scales, Science, 283 (1999), pp. 57–60.

[57] S.V. GARIMELLA AND C.B. SOBHAN, Transport in microchannels - a critical review, Annual Review of Heat Transfer, 13 (2003).

[58] S. GARMELLA, Y. JOSHI, A. BAR-COHEN, A. MAHAJONG, K. TOH, V. CAREY, M. BAELLMANS, J. LOHAN, B SAMMAKIA, AND F. ANDROS, Thermal challenges in next generation electronic systems - summary of panel presentations and discussions, IEEE Transactions on Components and Packaging Technologies, 25 (2002), pp. 569–575.

[59] P.R.C. GASCOYNE, J.V. VYKOUKAL, J.A. SCHWARTZ, T.J. ANDERSON, D.M VYKOUKAL, K.W. CURRENT, C. MCCONAGHY, F.F. BECKER, AND C. ANDREWS, Dielectrophoresis-based programmable fluidic processors, Lab Chip, 4 (2004), pp. 299–309.

[60] U. GHAI, K.N. GHAI, AND C.T. SHIN, High-Re solutions for incompressible flow using the navier-stokes equations and a multigrid method, J. Comp. Physics, 48 (1983), pp. 187–411.

[61] A. Gil, J. Segura, and N.M. Temme, Numerical methods for special functions, SIAM, Philadelphia, 2007.

[62] D. Goldstein, R. Handler, and L. Sirovich, Modeling a no-slip flow boundary with an external force field, J. Comput. Phys., 105 (1993), pp. 354–366.

[63] Lars Grasedyck, Ronald Kriemann, and Sabine Le Borne, Domain decomposition based $\mathcal{H}$-LU preconditioning, Numer. Math., 112 (2009), pp. 565–600.

[64] A. Greenbaum and L. Greengard, Laplace's equation and the Dirichlet-Neumann map in multiply connected domains, J. Comp. Physics, 105 (1993), pp. 267–278.

[65] L. Greengard and V. Rokhlin, A fast algorithm for particle simulations, J. Comput. Phys., 73 (1987), pp. 325–348.

[66] Leslie Greengard and Vladimir Rokhlin, A new version of the fast multipole method for the Laplace equation in three dimensions, in Acta numerica, 1997, vol. 6 of Acta Numer., Cambridge Univ. Press, Cambridge, 1997, pp. 229–269.

[67] R.B. Guenther and J.W. Lee, Partial differential equations of mathematical physics and integral equations, Dover, New York, 1988.

[68] A.K. Gupta, The boundary integral equation method for potential problems involving axisymmetric geometry and arbitrary boundary conditions, master's thesis, University of Kentucky, 1979.

[69] W. Hackbusch, The panel clustering technique for the boundary element method (invited contribution), in Boundary elements IX, Vol. 1 (Stuttgart, 1987), Comput. Mech., Southampton, 1987, pp. 463–474.

[70] Wolfgang Hackbusch, A sparse matrix arithmetic based on H-matrices; Part I: Introduction to H-matrices, Computing, 62 (1999), pp. 89–108.

[71] W. Hackbusch, B. Khoromskij, and S. Sauter, On $\mathcal{H}^2$-matrices, in Lectures on Applied Mathematics, Springer Berlin, 2002, pp. 9–29.

[72] J. Helsing and R. Ojala, Corner singularities for elliptic problems: Integral equations, graded meshes, quadrature, and compressed inverse preconditioning, J. Comput. Phys., 227 (2008), pp. 8820–8840.

[73] J. Helsing and E. Wadbro, Laplace's equation and the Dirichlet-Neumann map: a new mode for Mikhlin's method, J. Comp. Physics, 202 (2005), pp. 391–410.

[74] T.L. Holst, Numerical solution of axisymmetric boattail flow fields with plume simulators, AIAA paper AIAA 77-224, NASA Langley Research Center, Los Angeles, CA, 24-26 January 1977. 15th Aerospace Sciences Meeting and Exhibit.

[75] iNEMI, Electronics Manufacturing Initiative Technology Roadmap, 2004. Available: http://www.nemi.org.

[76] J.D. Jackson, Classical Electrodynamics, Wiley & Sons, New York, 1998.

[77] T.B. JONES, On the relationship of dielectrophoresis and electrowetting, Langmuir, 18 (2002), pp. 4437–4443.

[78] T.B. JONES, J.D. FOWLER, Y.S. CHANG, AND C.J. KIM, Frequency-based relationship of electrowetting and dielectrophoretic liquid microactuation, Langmuir, 19 (2003), pp. 7646–7651.

[79] T.B. JONES AND K.L. WANG, Frequency-dependent electromechanics of aqueous liquids: Electrowetting and dielectrophoresis, Langmuir, 20 (2004), pp. 2813–2818.

[80] K.H. KANG, How electrostatic fields change contact angle in electrowetting, Langmuir, 18 (2002), pp. 10318–10322.

[81] D.S. KERCHER, J.-B. LEE, O. BRAND, M.G. ALLEN, AND A. GLEZER, Microjet cooling devices for thermal management of electronics, Proceedings of IEEE Transactions on Components and Packaging Technologies, 26 (2003), pp. 359–366.

[82] J. KIM, Spray cooling heat transfer: the state of the art, Int. J. Heat Fluid Flow, 28 (2007), pp. 753–767.

[83] P. KOLM AND V. ROKHLIN, Numerical quadratures for singular and hypersingular integrals, Comput. Math. Appl., 41 (2001), pp. 327–352.

[84] R. KRESS AND W.T. SPASSOV, On the condition of boundary integral operators for the exterior dirichlet problem for the Helmholtz equation, Numer. Math., 42 (1983), pp. 77–95.

[85] A.H. KUIJPERS, G. VERBEEK, AND J.W. VERHEIJ, An improved acoustic fourier boundary element method formulation using fast fourier transform integration, J. Acoust. Soc. Am., 102 (1997), pp. 1394–1401.

[86] M. LAI AND C.S. PESKIN, An immersed boundary method with formal second-order accuracy and reduced numerical viscosity, J. Comput. Phys., 160 (2000), pp. 705–719.

[87] L.D. LANDAU, E.M. LIFSHITZ, AND L.P. PITAEVSKII, Electrodynamics of continuous media, vol. 8, Pergamon press, New York, second ed., 1984.

[88] J. LEE AND C.J. KIM, Surface tension driven microacuation based on continuous electrowetting, J. MEMS, 9 (2000), pp. 220–225.

[89] J. LEE, H. MOON, J. FOWLER, C.J. KIM, AND T. SCHOELLHAMMER, Addressable micro liquid handling by electric control of surface tension, in Proc. IEEE Int. Conf. MEMS, Interlaken, Switzerland, Jan. 2001, pp. 499–502.

[90] R.J. LEVEQUE AND Z. LI, The immersed interface method for elliptic equations with discontinuous coefficients and singular sources, SIAM J. Num. Anal., 31 (1994), pp. 1019–1044.

[91] Z. LI AND K. ITO, The Immersed Interface Method, SIAM, 2006.

[92] Z. LI AND M.C. LAI, The immersed interface method for the navierstokes equations with singular forces, J. Comput. Phys., 171 (2001), pp. 822–842.

[93] H.W. LU, F. BOTTAUSCI, J.D. FOWLER, A.L. BERTOZZI, AND C. MEINHART C.J. KIM, A study of ewod-driven droplets by piv investigation, Lab Chip, 8 (2008), pp. 457–461.

[94] P.G. MARTINSSON, A fast direct solver for a class of elliptic partial differential equations, J. Sci. Comput., 38 (2009), pp. 316–330.

[95] P.G. MARTINSSON AND V. ROKHLIN, A fast direct solver for boundary integral equations in two dimensions, J. Comput. Phys., 205 (2004), pp. 1–23.

[96] P.G. MARTINSSON AND V. ROKHLIN, A fast direct solver for boundary integral equations in two dimensions, J. Comput. Phys., 205 (2005), pp. 1–23.

[97] ——, An accelerated kernel independent fast multipole method in one dimension, SIAM Journal of Scientific Computing, 29 (2007), pp. 1160–11178.

[98] J.R. MELCHER, Continuum Mechanics, MIT Press, Cambridge, MA, 1981.

[99] J.R. MELCHER AND G.I. TAYLOR, Electrohydrodynamics: a review of the role of interfacial shear stresses, Ann. Rev. Fluid Mech., 1 (1969), pp. 111–146.

[100] C.A. MICCHELLI, Interpolation of scattered data: Distance matrices and conditionally positive definite functions, Constr. Approx., 2 (1986), pp. 11–22.

[101] E. MICHIELSSEN, A. BOAG, AND W. C. CHEW, Scattering from elongated objects: direct solution in $O(N \log^2 N)$ operations, IEE Proc. Microw. Antennas Propag., 143 (1996), pp. 277 – 283.

[102] S.G. MIKHLIN, Integral Equations, Pergammon, London, 1957.

[103] A. MINER AND U. GHOSAL, Cooling of high-power-density microdevices using liquid metal coolants, Appl. Phys. Lett., 85 (2004), pp. 506–508.

[104] R. MITTAL AND G. IACCARINO, Immersed boundary methods, Ann. Rev. Fluid Mech., 37 (2005), pp. 239–261.

[105] K. MOHSENI, Effective cooling of integrated circuits using liquid alloy electrowetting, in Proceedings of the Semiconductor Thermal Measurement, Modeling, and Management Symposium (SEMI-Therm), San Jose, California, USA, March 15 - 17 2005, IEEE.

[106] K. MOHSENI AND E. BAIRD, Digitized heat transfer using electrowetting on dielectric, Nanoscale and Microscale Thermophysical Engineering, 11 (2007), pp. 99 – 108.

[107] ——, A unified velocity model for digital microfluidics, Nanoscale and Microscale Thermophysical Engineering, 11 (2007), pp. 109 – 120.

[108] K. MOHSENI, E. BAIRD, AND H. ZHAO, Digitized heat transfer for thermal management of compact microsystems, in Proceedings of the 2005 ASME International Mechanical Engineering Congress and R & D Expo, Orlando, Florida, November 5-11 2005, ASME, pp. IMECE 2005–79372.

[109] K. MOHSENI AND E. S. BAIRD, Digitized heat transfer: Thermal management of compact micro systems using electrostatic droplet actuation, AIAA paper 2007-4157, 39th AIAA Thermophysics Conference, Miami, FL, June 25 - 28 2007.

[110] K. Mohseni and A. Dolatabadi, An electrowetting microvalve: Numerical simulation, Annals of the New York Academy of Sciences, 1077 (2006), pp. 415–425.

[111] H. Moon, S.K. Cho, R. Garrell, and C.J. Kim, Low voltage electrowetting-on-dielectric, J. Appl. Phys., 92 (2002), pp. 4080–4087.

[112] G.L. Morini, Single-phase convective heat transfer in microchannels: a review of experimental results, Int. J. Thermal Sciences, 43 (2004), pp. 631–651.

[113] I. Mudawar, Assessment of high-heat-flux thermal management schemes, IEEE Transactions on Components and Packaging Technologies, 24 (2001), pp. 122–141.

[114] F. Mugele and J. C. Baret, Electrowetting: From basics to applications, Journal of Physics: Condensed Matter, 17 (2005), pp. 705–774.

[115] S. Oleg and N. Alexander, Thermocapillary flows under an inclined temperature gradient, J. Fluid Mech., 504 (2004), pp. 99–132.

[116] V.K. Pamula and K. Chakrabarty, Cooling of integrated circuits using droplet-based microfluidics, in Proc. ACM Great Lakes Symposium on VLSI, 2003, pp. 84–87.

[117] P.A. Penfield and H.A. Haus, Electrodynamics of Moving Media, MIT Press, Cambridge, Massachuset, 1967.

[118] J.B. Perot, An analysis of the fractional step method, J. Comput. Phys., 108 (1993), pp. 51–58.

[119] C.S. Peskin, Flow patterns around heart valves: A numerical method, J. Comput. Phys., 10 (1972), pp. 252–271.

[120] ——, Numerical analysis of blood flow in the heart, J. Comput. Phys., 25 (1977), pp. 220–252.

[121] ——, The immersed boundary method, Acta Numer., 11 (2002), pp. 479–517.

[122] R. Peyret and T.D. Taylor, Computational Methods for Fluid Flow, Springer-Verlag New York, Inc., 1st ed., 1983.

[123] M.G. Pollack, R.B. Fair, and A.D. Shenderov, Electrowetting-based actuation of liquid droplets for microfluidic applications, Appl. Phys. Lett., 77 (2000), pp. 1725–1726.

[124] M.G. Pollack, A.D. Shenderov, and R.B. Fair, Electrowetting-based actuation of droplets for integrated microfluidics, Lab on a Chip, 2 (2002), pp. 96–101.

[125] M.J.D Powell, The theory of radial basis function approximations in 1990, in Advances in Numerical Analysis, vol. II: Wavelets, Subdivision Algorithms and Radial Functions, Oxford University Press, Oxford, UK, 1990, pp. 105–210.

[126] C. Provatidis, A boundary element method for axisymmetric potential problems with non-axisymmetric boundary conditions using fast fourier transform, Engrg. Comput., 15 (1998), pp. 428–449.

[127] R. Remsburg, ed., Advanced Thermal Design of Electronic Equipment, Chapman & Hall, New York, NY, 1998.

[128] F.J. Rizzo and D.J. Shippy, A boundary integral approach to potential and elasticity problems for axisymmetric bodies with arbitrary boundary conditions, Mech. Res. Commun., 6 (1979), pp. 99–103.

[129] E.M. Saiki and S. Biringen, Numerical simulation of a cylinder in uniform flow: application of a virtual boundary method, J. Comput. Phys., 123 (1996), pp. 450–465.

[130] D.A. Saville, Electrohydrodynamics: The Taylor-Melcher leaky-dielectric model, Annu. Rev. Fluid Mech., 29 (1997), pp. 27–64.

[131] A. Shakouri and J.E. Bowers, Heterostructure integrated thermionic coolers, Appl. Phys. Lett., 71 (1997), pp. 1234–1236.

[132] A. Shakouri, C. Labounty, P. Abraham, J. Piprek, and J.E. Bowers, Enhanced thermionic emission cooling in high barrier superlattice heterostructures, in Material Research Society Symposium Proceedings, vol. 545, December 1998, pp. 449–458.

[133] B. Shapiro, H. Moon, R.L. Garrell, and C.J. Kim, Equilibrium behavior of sessile drops under surface tension, applied external fields, and material variations, J. Appl. Phys., 93 (2003), pp. 5794–5811.

[134] Zhifeng Sheng, Patrick Dewilde, and Shivkumar Chandrasekaran, Algorithms to solve hierarchically semi-separable systems, in System theory, the Schur algorithm and multidimensional analysis, vol. 176 of Oper. Theory Adv. Appl., Birkhäuser, Basel, 2007, pp. 255–294.

[135] D.J. Shippy, F.J. Rizzo, and A.K. Gupta, Boundary-integral solution of potential problems involving axisymmetric bodies and nonsymmetric boundary conditions, in Developments in Theoretical and Applied Mechanics, J.E. Stoneking, ed., 1980, pp. 189–206.

[136] P. Singh and N. Aubry, Transport and deformation of droplets in a microdevice using dielectrophoresis, Electrophoresis, 28 (2007), pp. 644–657.

[137] B. Soenarko, A boundary element formuluation for radiation of acoustic waves from axisymmetric bodies with arbitrary boundary conditions, J. Acoust. Soc. Am., 93 (1993), pp. 631–639.

[138] Page Starr and Vladimir Rokhlin, On the numerical solution of two-point boundary value problems. II, Comm. Pure Appl. Math., 47 (1994), pp. 1117–1159.

[139] J.A. Stratton, Electromagnetic Theory, McGraw-Hill Book Company, 1941.

[140] K. Taira and T. Colonius, The immersed boundary method: A projection approach, J. Comput. Phys., 225 (2007), pp. 2118–2137.

[141] T. Thorsen, S.J. Maerkl, and S.R. Quake, Microfluidic large-scale integration, Science, 298 (2002), pp. 580–584.

[142] A.K. Tornberg and B. Engquist, Numerical approximations of singular source terms in differential equations, J. Comput. Phys., 200 (2004), pp. 462–488.

[143] S.V. TSINOPOULOS, J.P. AGNANTIARIS, AND D. POLYZOS, An advanced boundary element/fast fourier transform axisymmetric formulation for acoustic radiation and wave scattering problems, J. Acoust. Soc. Am., 105 (1999), pp. 1517–1526.

[144] D.B. TUCKERMAN AND R.F. PEASE, High performance heat sinking for VLSI, IEEE Electron Device Letters, 2 (1981), pp. 126–129.

[145] M. VALLET, M. VALLADE, AND B. BERGE, Limiting phenomena for the spreading of water on polymer films by electrowetting, Eur Phys J B, 11 (1999), pp. 583–591.

[146] J. WALDÉN, On the approximation of singular source terms in differential equations, Numer. Meth. Part D E, 15 (1999), pp. 503–520.

[147] S.W. WALKER AND B. SHAPIRO, Modeling the fluid dynamics of electrowetting on dielectric (EWOD), IEEE J. MEMS, 15 (2006), pp. 986–1000.

[148] K.L. WANG AND T.B. JONES, Electrowetting dynamics of microfluidic actuation, Langmuir, 21 (2005), pp. 4211–4217.

[149] W. WANG, N. ATALLA, AND J. NICOLAS, A boundary integral approach for accoustic radiation of axisymmetric bodies with arbitrary boundary conditions valid for all wave numbers, J. Acoust. Soc. Am., 101 (1997), pp. 1468–1478.

[150] H.H. WOODSON AND J.R. MELCHER, Electromechanical Dynamics, Part I: Discrete Systems, John Wiley & Sons, New York, 1968.

[151] ——, Electromechanical Dynamics, Part II: Fields, Forces, and Motion, John Wiley & Sons, New York, 1968.

[152] ——, Electromechanical Dynamics, Part III: Elastic and Fluid Media, John Wiley & Sons, New York, 1968.

[153] JINCHAO XU AND LUDMIL ZIKATANOV, On an energy minimizing basis for algebraic multigrid methods, Computing and Visualization in Science, 7 (2004), pp. 121–127. 10.1007/s00791-004-0147-y.

[154] B. YANG, P. WANGE, AND A. BAR-COHEN, Mini-contact enhanced thermoelectric cooling of hot spots in high power devices, IEEE Transactions on Components and Packaging Technologies, 30 (2007), pp. 432–438.

[155] L. YING, G. BIROS, AND D. ZORIN, A high-order 3D boundary integral equation solver for elliptic PDEs in smooth domains, J. Comp. Physics, 219 (2005), pp. 247–275.

Appendix A

Numerical Homogenization via Approximation of the Solution Operator

A.1   Introduction

A.1.1   Background

The purpose of this report is to draw attention to a number of recent developments in computational harmonic analysis that may prove helpful to the construction of simplified models for heterogeneous media. We consider problems modeled by elliptic PDEs such as electrostatics and linear elasticity in composite materials, and Stokes' flow in porous media.

Many different solution approaches have been proposed for the type of problems under consideration. A classical technique that works relatively well in situations where there is a clear separation of length-scales is to derive so called *homogenized equations* which accurately model the macro-scale behavior of the constitutive equations without fully resolving the micro-structure. The homogenized equations can sometimes be derived analytically, but they are typically obtained from numerically solving a set of equations defined on a *Representative Volume Element* (RVE). An unfortunate aspect of this approach is that its accuracy is held hostage to many factors that are outside of the control of the modeler. Phenomena that tend to lead to less accurate solutions include:

(1) Concentrated loads.

(2) Boundaries, in particular non-smooth boundaries.

(3) Irregular micro-structures.

The accuracy cannot readily be improved using generic techniques, but a number of strategies for developing coarse-grained models for specific situations have been developed. A popular class of such methods consists of variations of finite element methods in which a discretization on the macro-scale is constructed by solving a set of local problems defined on a representative collection of patches of fully resolved micro-structure [45, 50, 153].

We contend that it is in many situations advantageous to approximate the *solution operator*, rather than the *differential operator*. For the elliptic problems under consideration in this paper, the solution operator takes the form of an integral operator with the Green's function of the problem as its kernel. That such operators should in principle allow compressed representations has been known for some time (at least since [12]), but efficient techniques for actually computing them have become available only recently.

To illustrate the viability of the proposed techniques, we demonstrate how they apply to a couple of archetypical model problems. We first consider situations in which the micro-structure needs to be fully resolved and a coarse-grained model be constructed computationally. We show that this computation can be executed efficiently, and that once it has been, the reduced model allows for very fast solves, and is highly accurate even in situations that are challenging to existing coarse-graining methods. We then show that the proposed methods can fully exploit the simplifications possible when an accurate model of the material can be derived from computations on an RVE.

A.1.2   Mathematical problem formulation

While the ideas described are applicable in a broad range of environments, we will for expositional clarity focus on scalar elliptic boundary value problems defined on some regular domain $\Omega \subset \mathbb{R}^2$ with boundary Γ. Specifically, we consider Neumann problems of the form

$$
\begin{cases}
-\nabla \cdot \big(a(x) \cdot \nabla u(x)\big) = 0, & x \in \Omega, \\
\\
u_n(x) = f(x), & x \in \Gamma,
\end{cases}
\tag{A.1}
$$

where $a : \Omega \to \mathbb{R}^{2\times 2}$ is a matrix-valued function that varies "rapidly" (on the length-scale of the micro-structure), and where $u_n(x)$ denotes the normal derivative of u at $x \in \Gamma$. Our objective is to

rapidly construct $u|_\Gamma$, from a given boundary function f. We are interested both in the situation where we are allowed a pre-computation involving some given function a, and in the situation in which a is specified probabilistically.

Some of our numerical work will focus on the special case where (A.1) represents a two-phase material. To be precise, we suppose that Ω can be partitioned into two disjoint "phases," $\bar{\Omega} = \bar{\Omega}_1 \cup \bar{\Omega}_2$, and that there exist constants a_1 and a_2 such that

$$a(x) = \begin{cases} a_1\, I, & x \in \Omega_1, \\ a_2\, I, & x \in \Omega_2, \end{cases}$$

where I is the identity matrix. We further suppose that $\bar{\Omega}_2$ is wholly contained inside Ω, and let Γ_{int} denote the boundary between Ω_1 and Ω_2, see Figure **??**. Then (A.1) can more clearly be written

$$\begin{cases} -a_1 \Delta u(x) = 0, & x \in \Omega_1, \\ -a_2 \Delta u(x) = 0, & x \in \Omega_2, \\ [u](x) = 0, & x \in \Gamma_{\text{int}}, \\ [a\, u_n](x) = 0, & x \in \Gamma_{\text{int}}, \\ u_n(x) = f(x), & x \in \Gamma, \end{cases} \tag{A.2}$$

where for $x \in \Gamma$, $[u](x)$ and $[a\, u_n](x)$ denote the jumps in the potential and in the *flow* $-a(x)\nabla u(x)$ in the normal direction, respectively.

While the current paper concerns only situations modeled by equations of the types (A.1) and (A.2), the methodology extends to more general elliptic differential equations, see Section A.4.

A.1.3 Coarse-graining of the differential operator (homogenization)

A classical technique [7, 34] for handling a problem such as (A.1) with a rapidly varying coefficient function a is to construct a function a_{hom} that varies on the macroscale only (or may even be constant) such that the solution u is in some sense approximated by the solution u_{hom} to

$$\begin{cases} -\nabla \cdot \big(a_{\text{hom}}(x) \cdot \nabla u_{\text{hom}}(x)\big) = 0, & x \in \Omega, \\ \partial_n u_{\text{hom}}(x) = f(x), & x \in \Gamma. \end{cases} \tag{A.3}$$

The derivation of an equation such as (A.3) typically relies on fairly strong assumptions on separation of length-scales, rendering this technique problematic in situations involving boundary effects, concentrated loads, multiple or undifferentiated length-scales, *etc.* A common technique for ameliorating these difficulties is to preserve a piece of the fully resolved micro-structure near the boundary, or the concentrated load, and then to "glue" the two models together.

Another common approach is to forego the construction of a coarse-grained continuum model and construct an equation involving a discretized differential operator whose solution in some sense captures the macro-scale behavior of the solution of (A.3), see *e.g.* [45]. The elements of the discretized matrix are typically constructed via local computations on patches of micro-structure.

A.1.4   Coarse-graining of the solution operator

The premise of our work is that it is possible, and often advantageous, to approximate the *solution operator* of (A.1), rather than the differential operator itself. We will demonstrate that with this approach, many of the difficulties encountered in common coarse-graining strategies can be side-stepped entirely. To be precise, we note that mathematically, the solution to (A.1) takes the form

$$u(x) = [K\, f](x) = \int_{\Gamma} G(x, y)\, f(y)\, ds(y), \qquad x \in \Gamma, \tag{A.4}$$

where G is a kernel function that depends both on the function a, and on the domain Ω. It is known analytically only in the most trivial cases (such as a being constant, and Ω being a square or a circle). However, it turns out that the solution operator can be constructed numerically relatively cheaply, and that it admits very data-sparse representations.

Roughly speaking, our proposal is that instead of seeking an approximation of the form (A.3) of (A.1), it is often advantageous to seek an approximation of the form

$$u_{\mathrm{hom}}(x) = [K_{\mathrm{hom}}\, f](x) = \int_{\Gamma} G_{\mathrm{hom}}(x, y)\, f(y)\, ds(y), \qquad x \in \Gamma.$$

of (A.4). The purpose of the manuscript is to demonstrate the basic viability and desirability of this approach. Specifically, we seek to:

(1) Demonstrate via numerical examples that the solution operators can to high precision be approximated by "data-sparse" representations.

(2) Illustrate a framework in which highly accurate reduced models can be constructed even for situations involving boundary effects, and concentrated loads.

(3) Demonstrate that in situations where the full micro-structure needs to be resolved, there exist highly efficient techniques for doing so, and that the resulting reduced models form natural building blocks in computational models.

Remark 3 In this paper, we focus on problems with no body load, such as (A.1). However, the ideas set out can equally well be applied to problems such as

$$
\begin{cases}
-\nabla \cdot \big(a(x) \cdot \nabla u(x)\big) = h(x), & x \in \Omega, \\
u_n(x) = f(x), & x \in \Gamma.
\end{cases}
\tag{A.5}
$$

The mathematical solution operator then contains two terms, one corresponding to each of the two data functions f and h,

$$
u(x) = \int_\Gamma G(x, y) \, f(y) \, ds(y) + \int_\Omega K(x, y) \, h(y) \, dA(y), \qquad x \in \Omega.
\tag{A.6}
$$

The second term in (A.6) is compressible in a manner very similar to that of the first.

Remark 4 A reason why approximation of the solution operator may prove advantageous compared to approximating the differential operator is hinted at by the spectral properties of the problem. For a bounded domain, an elliptic operator A such as the one defined by equation (A.1) or (A.2) typically has a discrete spectrum $(\lambda_n)_{n=1}^\infty$, where $\lambda_n \to \infty$, and where eigenfunctions get more oscillatory the larger λ_n is. In up-scaling A, we seek to construct an operator A_{hom} whose low eigenvalues and eigenfunctions approximate those of A. Measuring success is tricky, however, since the operator $A - A_{\text{hom}}$ is in many ways dominated by the high eigenvalues. One way of handling this is to consider multi-scale representations of the operators, see, *e.g.*, [2, 20, 43, 46, 47]. Another way is to try to approximate the *inverse* of the operator. We observe that A^{-1} is typically compact,

and its dominant eigenmodes are precisely those that we seek to capture. Roughly speaking, we advocate the numerical construction of a finite dimensional operator T such that $||A^{-1} - T||$ is small.

Remark 5 Our goal with this paper is not to set up a mathematical analysis of the properties of kernels such as the function G in (A.4). However, to give a sense of the type of questions that arise, let us consider a situation where the function a in (A.1) represents a micro-structure with a characteristic length-scale λ. We then let d denote a cut-off parameter that separates the *near-field* from the *far-field*, say $d = 5\,\lambda$, and set

$$G_{\text{near}}(x, y) = \begin{cases} G(x, y), & |x - y| \leq d, \\ 0, & |x - y| > d, \end{cases} \qquad G_{\text{far}}(x, y) = \begin{cases} 0, & |x - y| \leq d, \\ G(x, y), & |x - y| > d, \end{cases}$$

and

$$u_{\text{near}}(x) = \int_{\Gamma} G_{\text{near}}(x, y)\, f(y)\, ds(y), \qquad u_{\text{far}}(x) = \int_{\Gamma} G_{\text{far}}(x, y)\, f(y)\, ds(y).$$

The function $y \mapsto G_{\text{near}}(x, y)$ depends strongly on the local micro-structure near x, and cannot easily be compressed. This part of the operator must be resolved sufficiently finely to fully represent the micro-structure. However, this is a local interaction, and u_{near} can be evaluated cheaply once G_{near} has been determined. In contrast, G_{far} is compressible. If Γ_1 and Γ_2 are two non-touching pieces of the boundary, then the integral operator

$$[T_{\Gamma_1 \leftarrow \Gamma_2} \sigma](x) = \int_{\Gamma_2} G_{\text{far}}(x, y)\, \sigma(y)\, ds(y), \qquad x \in \Gamma_1,$$

is not only compact, but its singular values typically decay exponentially fast, with the rate of decay depending on the sizes of Γ_1 and Γ_2, and on the distance between them. More careful analysis of these issues in an appropriate multi-scale framework can be found in [63].

A.2   Data-sparse matrices

A ubiquitous task in computational science is to rapidly perform linear algebraic operations involving very large matrices. Such operations typically exploit special *structure* in the matrix

since the costs for methods capable of handling general matrices tend to scale prohibitively fast with matrix size: For a general $N \times N$ matrix, it costs $O(N^2)$ operations to perform a matrix-vector multiplication, $O(N^3)$ operations to perform Gaussian elimination or to invert the matrix, *etc.* A well-known form of structure in a matrix is sparsity. When at most a few entries in each row of the matrix are non-zero (as is the case, *e.g.*, for matrices arising upon the discretization of differential equations, or representing the link structure of the World Wide Web) matrix-vector multiplications can be performed in $O(N)$ operations instead of $O(N^2)$. The description *data-sparse* applies to a matrix that may be dense, but that shares the key characteristic of a sparse matrix that some linear algebraic operations, typically the matrix-vector multiplication, can to high precision be executed in fewer than $O(N^2)$ operations (often in close to linear time).

There are many different types of data-sparse representations of a matrix. In this paper, we will utilize techniques for so called *Hierarchically Semi-Separable* (HSS) matrices [25, 27, 134], which arise upon the discretization of many of the integral operators of mathematical physics, in signal processing, in algorithms for inverting certain finite element matrices, and in many other applications, see *e.g.* [26, 94, 134]. An HSS matrix is a dense matrix whose off-diagonal blocks are rank-deficient in a certain sense. Without going into details, we for now simply note that an HSS matrix A can be expressed via a recursive formula in L levels,

$$\mathsf{A}^{(\ell)} = \mathsf{U}^{(\ell)} \, \mathsf{A}^{(\ell-1)} \, \mathsf{V}^{(\ell)} + \mathsf{B}^{(\ell)}, \qquad \ell = 2,\, 3,\, \ldots,\, L, \tag{A.7}$$

where $\mathsf{A} = \mathsf{A}^{(L)}$, and the sequence $\mathsf{A}^{(L)}$, $\mathsf{A}^{(L-1)}$, $\ldots$, $\mathsf{A}^{(1)}$ consists of matrices that are successively smaller (typically, $\mathsf{A}^{(\ell-1)}$ is roughly half the size of $\mathsf{A}^{(\ell)}$). In (A.7), the matrices $\mathsf{U}^{(\ell)}$, $\mathsf{V}^{(\ell)}$ and $\mathsf{B}^{(\ell)}$ are all block-diagonal, so the formula directly leads to a fast technique for evaluating a matrix-vector product. The HSS property is similar to many other data-sparse representations in that it exploits rank-deficiencies in off-diagonal blocks to allow matrix-vector products to be evaluated rapidly; the Fast Multipole Method [65, 66], Barnes-Hut [9], and panel clustering [69] are all similar in this regard. The HSS property is different from these other formats in that it also allows the rapid computation of a matrix inverse, of an LU factorization, *etc*, [24, 25, 41, 96, 138]. The ability

to perform algebraic operations other than the matrix-vector multiplication is also characteristic of the $\mathcal{H}$-matrix format of Hackbusch [70].

Remark 6 There currently is little consistency in terminology when it comes to "data-sparse" matrices. The property that we refer to as the "HSS" property has appeared under different names in, *e.g.*, [96, 97, 101, 138]. It is closely related to the "$\mathcal{H}^2$-matrix" format [14, 15, 16, 71] which is more restrictive than the $\mathcal{H}$-matrix format, and often admits $O(N)$ algorithms.

A.3   Case study: Two-phase media

In this section, we briefly investigate the compressibility of the Neumann-to-Dirichlet operator for a two-phase material modeled by equation (A.2). The two geometries we consider are shown in Figure A.1, with the conductivity of the inclusions set to zero. In this case, the operator under consideration is a boundary integral operator T supported on the square outer boundary. Using techniques described in Remark 7, we constructed an 1144×1144 matrix T that approximated T. With this number of nodes, any Neumann data generated by point sources up to a distance of 0.5% of the side length of the square can be resolved to eight digits of accuracy. We compressed the matrix T into the HSS format described in Section A.2 to a relative precision of 10^{-10}. The resulting data required 1.19KB of memory to store for the geometry shown in Figure A.1(a), and 1.22KB of memory for the geometry shown in Figure A.1(b). This corresponds to about 135 words of storage per row in the matrix. The HSS-ranks are reported in Table A.1. We make three observations:

- The two geometries shown in Figure A.1 require about the same amount of memory. This is note-worthy since the one labeled (b) corresponds to an almost singular geometry in which the domain is very close to being split in two halves. The effect is illustrated the solution shown in Figure A.2(b) where steep gradients are seen in middle of the piece. Standard assumptions used when homogenizing an elliptic differential operator are violated in this case.

- In Table A.1, the ranks of HSS-blocks of size 143 are *larger* than those of HSS-blocks of
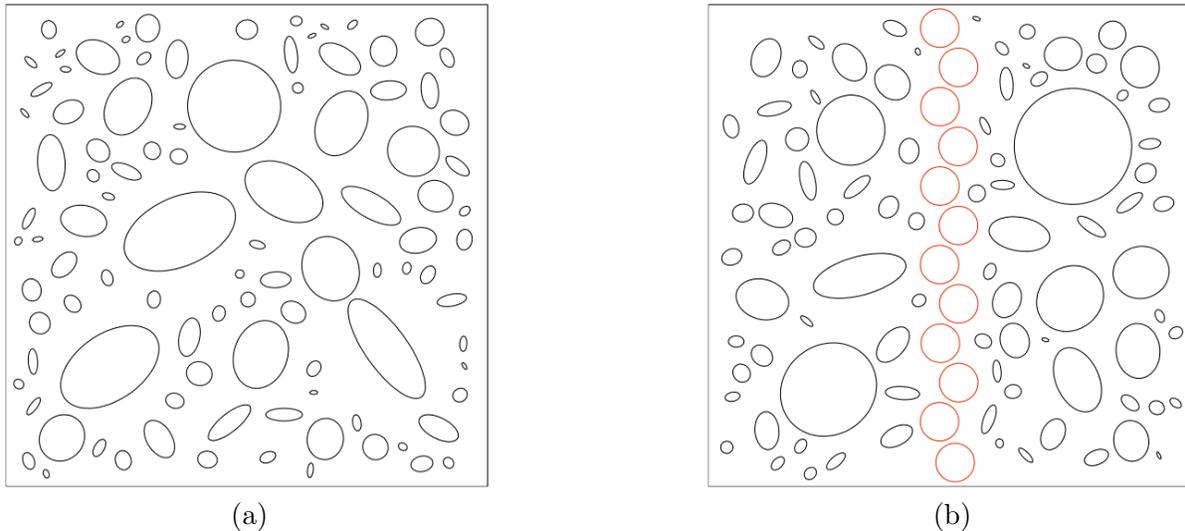
Figure A.1: Geometry for computations in Section A.3. (a) A perforated material. (b) A perforated material with a chain of holes that almost line up.

size 286. We speculate that this unusual situation can be traced to the fact that the larger blocks are larger than the inclusions, and largely do not "see" the heterogeneities.

Remark 7 (Details of computation) To derive our approximation to the Neumann-to-Dirichlet operator, we recast the Neumann Laplace equation (A.2) as a BIE defined on the joint boundary $\Gamma \cup \Gamma_{\text{int}}$. In the present case with non-conducting inclusions, the boundary condition on all interior boundaries simplifies to a homogeneous Neumann condition. We represented the solution as a single layer representation supported on both the outer boundary Γ and the interior boundary Γ_{int}. In other words, we sought a solution of the form

$$u(x) = \int_{\Gamma} \log|x - y|\, \sigma(y)\, ds(y) + \int_{\Gamma_{\text{int}}} \log|x - y|\; \tau(y)\, ds(y). \tag{A.8}$$

The resulting BIE was discretized using a Nyström method combined with trapezoidal quadrature on the interior holes, and a Gaussian quadrature on the exterior boundary supported on 44 panels with 26 nodes each. The quadrature rule was locally modified as described in [18] to maintain eight digit accuracy in the presence of corners. This resulted in a large linear system from which all degrees of freedom associated with internal nodes (those associated with the density τ in (A.8)) were
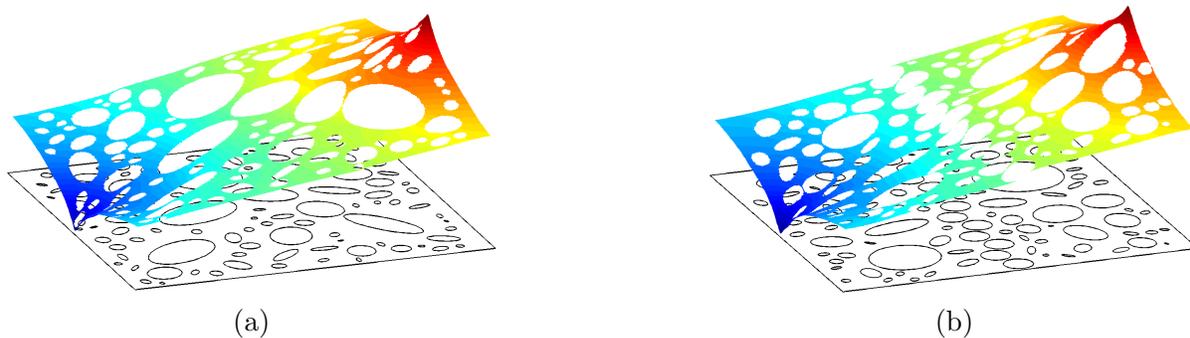
Figure A.2: Solutions to the Laplace's equation with Neumann boundary conditions on the geometries (a) and (b) shown in Figure A.1. The boundary flux is set to be identically zero, except for two point sources of strengths ± 1.

AVERAGE RANKS OF HSS BLOCKS FOR COMPOSITE MATERIAL EXAMPLE IN SECTION A.3

	$N_{\text{block}} = 36$	$N_{\text{block}} = 71$	$N_{\text{block}} = 143$	$N_{\text{block}} = 286$
Geometry shown in Figure A.1(a)	18.2	27.0	39.5	25.8
Geometry shown in Figure A.1(b)	18.3	27.3	41.1	28.0

Table A.1: The average HSS-ranks for the blocks in a data-sparse representation of the Neumann-to-Dirichlet operator for the geometries shown in Figure A.1.

eliminated. The resulting Schur complement was multiplied by a matrix representing evaluation of a single layer potential on the boundary to produce the final discrete approximation T to the "true" analytic Neumann-to-Dirichlet operator T.

A.4   Generalizations

This report focused on problems modeled by simple Laplace-type problems in two dimensions involving no body loads. However, the techniques can be extended to much more general environments:

Other boundary conditions: While we focused on problems with Neumann boundary conditions, the extension to Dirichlet or mixed boundary conditions is trivial.

Other elliptic equations: The methods described extend readily to other elliptic equations whose kernels are non-oscillary such as Stokes, elasticity, Yukawa, *etc.* The extension to wave problems modeled by Helmholtz equation, or the time-harmonic version of Maxwell, is more complicated for two reasons: (1) The presence of resonances (both true ones corresponding to the actual physics, and artificial ones present in the mathematical model only) must be dealt with. This can be done, but requires careful attention. (2) As the wave-number increases, the compressibility of the solution operator deteriorates, and eventually renders the proposed approach wholly unaffordable.

Body loads: The extension to problems involving body loads is in principle straight-forward. However, the compressed solution operator becomes more expensive to store.

Problems in three dimensions: In principle, the methodology proposed extends straight-forwardly to problems in three dimensions. However, the construction of the solution operator does become more expensive, and the method might be best suited for environments where a pre-computation is possible, or where the construction of the solution operator can be accelerated via the use of homogenized models in parts of the domain (as illustrated in Section **??**). Moreover, for problems in three dimensions involving body loads, memory requirements may become prohibitive.

A.5   Conclusions

The purpose of this report is to attempt to draw attention to recent developments in numerical analysis that could be very useful in modeling heterogeneous media. Specifically, it has become possible to inexpensively compute an approximation to the solution operator associated with many elliptic PDEs, and to perform various operations involving such solution operators: addition, multiplication, inversion, merging operators for different sub-domains, *etc.* We argue that such solution operators form excellent "reduced models" for many problems that have proven difficult to handle using traditional homogenization techniques.

Constructing reduced models by approximating the solution operator is particularly advantageous in the following environments:

Domains that are loaded on the boundary only: For problems that involve no body load, the solution operator is defined on the boundary only. This reduction in dimensionality means that once it is computed, it can be stored very efficiently, and applied to vectors sufficiently fast that real time simulations become possible. For some problems in this category, the actual construction of the solution operator requires a large-scale (but very efficient) computation involving the entire micro-structure, but the solution operator can sometimes be dramatically accelerated by using a homogenized model in the interior of the domain.

Situations where a pre-computation is possible: When the entire micro-structure needs to be resolved (as happens when the problem involves a body load, or a micro-structure not suitable for homogenization methods), the initial construction of the solution operator can become somewhat expensive, in particular for problems in three dimensions. However, once it has been constructed, it can usually be applied to a vector very rapidly. This raises the possibility of pre-computing a library of compressed models which can then be used as building blocks in computational simulations.

Problems in two dimensions (whether involving volume loads or not): Given current trends in algorithmic and hardware development, we predict that for a great many problems in two dimensions, it will soon become entirely affordable to resolve the entire micro-structure, and

computationally derive a reduced model of the solution operator. The automatic nature of such a procedure would save much human effort, and would be very robust in the sense that the computed model would be guaranteed to be accurate to whichever tolerance was requested.

Appendix B

Tables of Quadrature Nodes and Weights

The quadrature rules in this appendix are of Gaussian type. They are all designed to integrate polynomials of up to degree 20 exactly. Note that the singular and near singular rules also handle polynomials of up to degree 20 multiplied by the indicated singularities.

B.1 Standard 10 Point Gauss-Legendre Rule

10 Point Gauss-Legendre Rule for integrals of the form $\int_{-1}^{1} f(x)\,dx$	
NODES	WEIGHTS
-9.739065285171716e-01	6.667134430868814e-02
-8.650633666889845e-01	1.494513491505806e-01
-6.794095682990244e-01	2.190863625159820e-01
-4.333953941292472e-01	2.692667193099963e-01
-1.488743389816312e-01	2.955242247147529e-01
1.488743389816312e-01	2.955242247147529e-01
4.333953941292472e-01	2.692667193099963e-01
6.794095682990244e-01	2.190863625159820e-01
8.650633666889845e-01	1.494513491505806e-01
9.739065285171716e-01	6.667134430868814e-02

B.2   20 Point Rules for Log-Singularities at the Standard 10 Point Gauss-Legendre Nodes

20 point quadrature rule for integrals of the form $\int_{-1}^{1} f(x) + g(x) \log|x_1 - x| \, dx$, where x_1 is a Gauss-Legendre node	
NODES	WEIGHTS
-9.981629455677877e-01	4.550772157144354e-03
-9.915520723139890e-01	8.062764683328619e-03
-9.832812993252168e-01	7.845621096866406e-03
-9.767801773920733e-01	4.375212351185101e-03
-9.717169387169078e-01	1.021414662954223e-02
-9.510630103726074e-01	3.157199356768625e-02
-9.075765988474132e-01	5.592493151946541e-02
-8.382582352569804e-01	8.310260847601852e-02
-7.408522006801963e-01	1.118164522164500e-01
-6.147619568252419e-01	1.401105427713687e-01
-4.615244999958006e-01	1.657233639623953e-01
-2.849772954295424e-01	1.863566566231937e-01
-9.117593460489747e-02	1.999093145144455e-01
1.119089520342051e-01	2.046841584582030e-01
3.148842536644393e-01	1.995580161940930e-01
5.075733846631832e-01	1.841025430283230e-01
6.797470718157004e-01	1.586456191174843e-01
8.218833662202629e-01	1.242680229936124e-01
9.258924858821892e-01	8.273794370795576e-02
9.857595961761246e-01	3.643931593123844e-02

20 point quadrature rule for integrals of the form $\int_{-1}^{1} f(x) + g(x) \log|x_2 - x| \, dx$, where x_2 is a Gauss-Legendre node	
NODES	WEIGHTS
-9.954896691005256e-01	1.141744473788874e-02
-9.775532683688947e-01	2.368593568061651e-02
-9.500346715183706e-01	3.027205199814611e-02
-9.192373372373420e-01	3.021809354380292e-02
-8.916563772395616e-01	2.397183723558556e-02
-8.727728136507039e-01	1.253574079839078e-02
-8.607963163061316e-01	2.070840476545303e-02
-8.201318720954396e-01	6.080709508468810e-02
-7.394732321355052e-01	1.002402801599464e-01
-6.204853512352519e-01	1.371499151597280e-01
-4.667290485167077e-01	1.693838059093582e-01
-2.840823320902124e-01	1.945292086962893e-01
-8.079364608026202e-02	2.103223087093422e-01
1.328455136645940e-01	2.149900928447852e-01
3.451233500669768e-01	2.074984762344433e-01
5.437321547508867e-01	1.877085225595498e-01
7.167077216635750e-01	1.564543949958065e-01
8.534299232009863e-01	1.156104890379952e-01
9.458275339169444e-01	6.859369195724087e-02
9.912353127269481e-01	2.390220989094312e-02

20 point quadrature rule for integrals of the form $\int_{-1}^{1} f(x) + g(x) \log|x_3 - x| \, dx$, where x_3 is a Gauss-Legendre node	
NODES	WEIGHTS
-9.930122613589740e-01	1.779185041193254e-02
-9.643941806993207e-01	3.870503119897836e-02
-9.175869559770760e-01	5.371120494602663e-02
-8.596474181980754e-01	6.073467932536858e-02
-7.990442708271941e-01	5.901993373645797e-02
-7.443700671611690e-01	4.905519963921684e-02
-7.031684479828371e-01	3.249237036645046e-02
-6.811221147275545e-01	1.335394660596527e-02
-6.579449960254029e-01	4.151626407911676e-02
-5.949471688137100e-01	8.451456165895121e-02
-4.893032793226841e-01	1.262522607368499e-01
-3.441659232382107e-01	1.628408264966550e-01
-1.665388322404095e-01	1.907085686614375e-01
3.344207582228461e-02	2.071802230953481e-01
2.434356263087524e-01	2.105274833603497e-01
4.498696863725133e-01	2.000282912446872e-01
6.389777518528792e-01	1.760212445284564e-01
7.978632877793501e-01	1.399000904426490e-01
9.155180703268415e-01	9.402669072995991e-02
9.837258757826489e-01	4.161927873514264e-02

20 point quadrature rule for integrals of the form $\int_{-1}^{1} f(x) + g(x) \log|x_4 - x|\, dx$, where x_4 is a Gauss-Legendre node		20 point quadrature rule for integrals of the form $\int_{-1}^{1} f(x) + g(x) \log|x_5 - x|\, dx$, where x_5 is a Gauss-Legendre node	
NODES	WEIGHTS	NODES	WEIGHTS
-9.903478871133073e-01	2.462513260640712e-02	-9.883561797860961e-01	2.974603958509255e-02
-9.504025146897784e-01	5.449201732062665e-02	-9.398305159297058e-01	6.657945456889164e-02
-8.834986023815121e-01	7.799498604905293e-02	-8.572399919019390e-01	9.731775484182564e-02
-7.974523551287549e-01	9.241688894090601e-02	-7.482086250804679e-01	1.190433988432928e-01
-7.022255002503461e-01	9.619882322938848e-02	-6.228514167093102e-01	1.297088242013777e-01
-6.087194789244920e-01	8.902783806614303e-02	-4.928317114329241e-01	1.282900896966494e-01
-5.275278952351541e-01	7.181973054766198e-02	-3.702771193724617e-01	1.148917968875341e-01
-4.677586540799037e-01	4.663017060126023e-02	-2.666412108172461e-01	9.074932908233864e-02
-4.360689210457623e-01	1.794303974050253e-02	-1.916083010783277e-01	5.818196361216740e-02
-4.121945474875853e-01	4.061799823415495e-02	-1.521937160593461e-01	2.224697059733435e-02
-3.494226766911471e-01	8.507517518447759e-02	-1.233125650067164e-01	4.788826761346366e-02
-2.425993523586304e-01	1.277525783357134e-01	-5.257959675044444e-02	9.237500180593534e-02
-9.646839923908594e-02	1.628510773009247e-01	5.877314311857769e-02	1.287410543031414e-01
7.921243716767302e-02	1.863323765408308e-01	2.012559739993003e-01	1.541960911507042e-01
2.715178194484646e-01	1.958227701927855e-01	3.627988191760868e-01	1.665885274544506e-01
4.658440358656903e-01	1.903138548150517e-01	5.297121321076323e-01	1.648585116745725e-01
6.472213975763533e-01	1.700731513381802e-01	6.878399330187783e-01	1.491408089644010e-01
8.015601619414859e-01	1.365784674773513e-01	8.237603202215137e-01	1.207592726093190e-01
9.168056007307982e-01	9.239595239693155e-02	9.259297297557394e-01	8.212177982524418e-02
9.839468743284722e-01	4.103797108164931e-02	9.856881498392895e-01	3.657506268226379e-02

20 point quadrature rule for integrals of the form $\int_{-1}^{1} f(x) + g(x) \log|x_6 - x|\, dx$, where x_6 is a Gauss-Legendre node		20 point quadrature rule for integrals of the form $\int_{-1}^{1} f(x) + g(x) \log|x_7 - x|\, dx$, where x_7 is a Gauss-Legendre node	
NODES	WEIGHTS	NODES	WEIGHTS
-9.856881498392895e-01	3.657506268226379e-02	-9.839468743284722e-01	4.103797108164931e-02
-9.259297297557394e-01	8.212177982524418e-02	-9.168056007307982e-01	9.239595239693155e-02
-8.237603202215137e-01	1.207592726093190e-01	-8.015601619414859e-01	1.365784674773513e-01
-6.878399330187783e-01	1.491408089644010e-01	-6.472213975763533e-01	1.700731513381802e-01
-5.297121321076323e-01	1.648585116745725e-01	-4.658440358656903e-01	1.903138548150517e-01
-3.627988191760868e-01	1.665885274544506e-01	-2.715178194484646e-01	1.958227701927855e-01
-2.012559739993003e-01	1.541960911507042e-01	-7.921243716767302e-02	1.863323765408308e-01
-5.877314311857769e-02	1.287410543031414e-01	9.646839923908594e-02	1.628510773009247e-01
5.257959675044444e-02	9.237500180593534e-02	2.425993523586304e-01	1.277525783357134e-01
1.233125650067164e-01	4.788826761346366e-02	3.494226766911471e-01	8.507517518447759e-02
1.521937160593461e-01	2.224697059733435e-02	4.121945474875853e-01	4.061799823415495e-02
1.916083010783277e-01	5.818196361216740e-02	4.360689210457623e-01	1.794303974050253e-02
2.666412108172461e-01	9.074932908233864e-02	4.677586540799037e-01	4.663017060126023e-02
3.702771193724617e-01	1.148917968875341e-01	5.275278952351541e-01	7.181973054766198e-02
4.928317114329241e-01	1.282900896966494e-01	6.087194789244920e-01	8.902783806614303e-02
6.228514167093102e-01	1.297088242013777e-01	7.022255002503461e-01	9.619882322938848e-02
7.482086250804679e-01	1.190433988432928e-01	7.974523551287549e-01	9.241688894090601e-02
8.572399919019390e-01	9.731775484182564e-02	8.834986023815121e-01	7.799498604905293e-02
9.398305159297058e-01	6.657945456889164e-02	9.504025146897784e-01	5.449201732062665e-02
9.883561797860961e-01	2.974603958509255e-02	9.903478871133073e-01	2.462513260640712e-02

20 point quadrature rule for integrals of the form $\int_{-1}^{1} f(x) + g(x) \log|x_8 - x|\, dx$, where x_8 is a Gauss-Legendre node	
NODES	WEIGHTS
-9.837258757826489e-01	4.161927873514264e-02
-9.155180703268415e-01	9.402669072995991e-02
-7.978632877793501e-01	1.399000904426490e-01
-6.389777518528792e-01	1.760212445284564e-01
-4.498696863725133e-01	2.000282912446872e-01
-2.434356263087524e-01	2.105274833603497e-01
-3.344207582228461e-02	2.071802230953481e-01
1.665388322404095e-01	1.907085686614375e-01
3.441659232382107e-01	1.628408264966550e-01
4.893032793226841e-01	1.262522607368499e-01
5.949471688137100e-01	8.451456165895121e-02
6.579449960254029e-01	4.151626407911676e-02
6.811221147275545e-01	1.335394660596527e-02
7.031684479828371e-01	3.249237036645046e-02
7.443700671611690e-01	4.905519963921684e-02
7.990442708271941e-01	5.901993373645797e-02
8.596474181980754e-01	6.073467932536858e-02
9.175869559770760e-01	5.371120494602663e-02
9.643941806993207e-01	3.870503119897836e-02
9.930122613589740e-01	1.779185041193254e-02

20 point quadrature rule for integrals of the form $\int_{-1}^{1} f(x) + g(x) \log|x_9 - x|\, dx$, where x_9 is a Gauss-Legendre node	
NODES	WEIGHTS
-9.912353127269481e-01	2.390220989094312e-02
-9.458275339169444e-01	6.859369195724087e-02
-8.534299232009863e-01	1.156104890379952e-01
-7.167077216635750e-01	1.564543949958065e-01
-5.437321547508867e-01	1.877085225595498e-01
-3.451233500669768e-01	2.074984762344433e-01
-1.328455136645940e-01	2.149900928447852e-01
8.079364608026202e-02	2.103223087093422e-01
2.840823320902124e-01	1.945292086962893e-01
4.667290485167077e-01	1.693838059093582e-01
6.204853512352519e-01	1.371499151597280e-01
7.394732321355052e-01	1.002402801599464e-01
8.201318720954396e-01	6.080709508468810e-02
8.607963163061316e-01	2.070840476545303e-02
8.727728136507039e-01	1.253574079839078e-02
8.916563772395616e-01	2.397183723558556e-02
9.192373372373420e-01	3.021809354380292e-02
9.500346715183706e-01	3.027205199814611e-02
9.775532683688947e-01	2.368593568061651e-02
9.954896691005256e-01	1.141744473788874e-02

20 point quadrature rule for integrals of the form $\int_{-1}^{1} f(x) + g(x) \log|x_{10} - x|\, dx$, where x_{10} is a Gauss-Legendre node	
NODES	WEIGHTS
-9.857595961761246e-01	3.643931593123844e-02
-9.258924858821892e-01	8.273794370795576e-02
-8.218833662202629e-01	1.242680229936124e-01
-6.797470718157004e-01	1.586456191174843e-01
-5.075733846631832e-01	1.841025430283230e-01
-3.148842536644393e-01	1.995580161940930e-01
-1.119089520342051e-01	2.046841584582030e-01
9.117593460489747e-02	1.999093145144455e-01
2.849772954295424e-01	1.863566566231937e-01
4.615244999958006e-01	1.657233639623953e-01
6.147619568252419e-01	1.401105427713687e-01
7.408522006801963e-01	1.118164522164500e-01
8.382582352569804e-01	8.310260847601852e-02
9.075765988474132e-01	5.592493151946541e-02
9.510630103726074e-01	3.157199356768625e-02
9.717169387169078e-01	1.021414662954223e-02
9.767801773920733e-01	4.375212351185101e-03
9.832812993252168e-01	7.845621096866406e-03
9.915520723139890e-01	8.062764683328619e-03
9.981629455677877e-01	4.550772157144354e-03

B.3 24 Point Rules for Log-Singularities Near the Interval of Integration

24 point quadrature rule for integrals of the form $\int_0^1 f(x) + g(x)\log(x+\bar{x})dx$, where $\bar{x} \geq 10^{-1}$	
NODES	WEIGHTS
3.916216329415252e-02	4.880755296918116e-02
8.135233983530081e-02	3.196002785163611e-02
1.123448211344994e-01	3.883416642507362e-02
1.595931983965030e-01	5.148898992140820e-02
2.085759027831349e-01	4.219328148763533e-02
2.426241962027560e-01	3.420686213633789e-02
2.886190312538522e-01	5.512488680719239e-02
3.469021762354675e-01	6.007112809843418e-02
4.072910101569611e-01	6.022350479415180e-02
4.664019722595442e-01	5.735022004401478e-02
5.182120817844112e-01	4.167923417118068e-02
5.501308436771654e-01	3.346089628879600e-02
5.970302980854608e-01	5.574716218423796e-02
6.548457960388209e-01	5.847838243344473e-02
7.119542126106005e-01	5.464156990092474e-02
7.607920420946340e-01	4.092186343704961e-02
7.953017051155684e-01	3.283728166050225e-02
8.303900341517088e-01	3.438233273473095e-02
8.612724919009394e-01	3.022585192226418e-02
8.954049128027080e-01	3.700769701277380e-02
9.315909369155358e-01	3.410213679365162e-02
9.621742249068356e-01	2.665791885274193e-02
9.843663446380599e-01	1.754420526360429e-02
9.970087425823398e-01	7.662283104388867e-03

24 point quadrature rule for integrals of the form $\int_0^1 f(x) + g(x)\log(x+\bar{x})dx$, where $10^{-2} \le \bar{x} \le 10^{-1}$	
NODES	WEIGHTS
1.940564616937581e-02	2.514022176052795e-02
4.545433992382339e-02	2.703526530535647e-02
7.378866604396420e-02	2.980872487617485e-02
1.054147718077606e-01	3.360626237885489e-02
1.412997888401000e-01	3.829678083416609e-02
1.822325567811081e-01	4.365651045780837e-02
2.287282121202408e-01	4.935846322319046e-02
2.809170925514041e-01	5.495967924055210e-02
3.384320962237970e-01	5.991162198705084e-02
4.003108031244078e-01	6.356960862248889e-02
4.648605571606025e-01	6.506868552467118e-02
5.290714994276687e-01	6.219588235225894e-02
5.829663557386375e-01	3.889986041695310e-02
6.128301889979477e-01	3.573431931940621e-02
6.606072156240962e-01	5.296315368353523e-02
7.139495966128518e-01	5.369033999927759e-02
7.677830914961244e-01	5.340793573367282e-02
8.187382423336450e-01	4.704756013998560e-02
8.587068551739496e-01	3.276576301747068e-02
8.906873285570645e-01	3.449175311880027e-02
9.267772492129903e-01	3.560168848238671e-02
9.592137652582382e-01	2.857367151127661e-02
9.830962712794008e-01	1.894042942442201e-02
9.967621546194148e-01	8.291994770212826e-03

24 point quadrature rule for integrals of the form $\int_0^1 f(x) + g(x)\log(x+\bar{x})dx$, where $10^{-3} \le \bar{x} \le 10^{-2}$	
NODES	WEIGHTS
7.571097817272427e-03	9.878088201321919e-03
1.800655325976786e-02	1.109316819462674e-02
3.003901004577040e-02	1.313311581321880e-02
4.462882147989575e-02	1.624262442061470e-02
6.295732618092606e-02	2.065168462990214e-02
8.644035241970913e-02	2.657795406825320e-02
1.166164809306920e-01	3.399052299072427e-02
1.546690628394902e-01	4.208214612865170e-02
1.999554346680615e-01	4.732516974042797e-02
2.434683359132119e-01	3.618419415803922e-02
2.800846274146029e-01	4.547346840583578e-02
3.368595257878888e-01	6.463153575242817e-02
4.044418359833648e-01	6.859104457897808e-02
4.685002493634456e-01	5.589917935916451e-02
5.185062817085154e-01	5.199232318335285e-02
5.811314144990846e-01	7.089840644422261e-02
6.545700991450585e-01	7.427400331494240e-02
7.276588861478224e-01	7.125308736931726e-02
7.960626077582168e-01	6.513697474660338e-02
8.572037183403355e-01	5.682298546820264e-02
9.091330485015775e-01	4.678000924507099e-02
9.503131649503738e-01	3.538488886617123e-02
9.795718963793163e-01	2.299723483013955e-02
9.961006479199827e-01	9.993597414733579e-03

24 point quadrature rule for integrals of the form $\int_0^1 f(x) + g(x)\log(x + \bar{x})dx$, where $10^{-4} \leq \bar{x} \leq 10^{-3}$		24 point quadrature rule for integrals of the form $\int_0^1 f(x) + g(x)\log(x + \bar{x})dx$, where $10^{-5} \leq \bar{x} \leq 10^{-4}$	
NODES	WEIGHTS	NODES	WEIGHTS
2.625961371586153e-03	3.441901737135120e-03	7.759451679242260e-04	1.049591733965263e-03
6.309383772392260e-03	3.978799794732070e-03	1.952854410117286e-03	1.314968855711329e-03
1.073246133489697e-02	4.958449505644980e-03	3.429053832116395e-03	1.651475072547296e-03
1.645170499644402e-02	6.620822501994994e-03	5.301128540262913e-03	2.135645684467029e-03
2.433800511777796e-02	9.385496468197222e-03	7.878118775220067e-03	3.165043382856636e-03
3.582530925992294e-02	1.396512052439178e-02	1.205537050949829e-02	5.479528688655274e-03
5.315827372101662e-02	2.119383832447796e-02	1.965871512055557e-02	1.028817002915096e-02
7.917327903614484e-02	3.124989308824302e-02	3.403328641997047e-02	1.923291785614007e-02
1.162053707416708e-01	4.291481168916344e-02	5.947430305925957e-02	3.212643438782854e-02
1.648139164451449e-01	5.400832278279924e-02	9.873500543531440e-02	4.638626850049229e-02
2.231934088488800e-01	6.197424674301215e-02	1.518862681939413e-01	5.960676923068444e-02
2.864519293820641e-01	6.297221626131570e-02	2.171724325134259e-01	7.052360405410943e-02
3.466729491189400e-01	5.794981636764223e-02	2.919941878735093e-01	7.863451090237836e-02
4.076175535528108e-01	6.650501614478806e-02	3.734637353255530e-01	8.381771698595157e-02
4.800964107543535e-01	7.716379373230733e-02	4.586710018443288e-01	8.612755554083525e-02
5.594105009204460e-01	8.047814122759604e-02	5.448057416999684e-01	8.569938467103264e-02
6.395390292352857e-01	7.917822434973971e-02	6.292158981939618e-01	8.271051499695768e-02
7.167410782176877e-01	7.477646096014055e-02	7.094415843889587e-01	7.736692567834522e-02
7.882807127957939e-01	6.793424765652059e-02	7.832417328632321e-01	6.990012937760461e-02
8.519356675821297e-01	5.906852968947303e-02	8.486194141302759e-01	6.056687669667680e-02
9.058606177202579e-01	4.853108558910315e-02	9.038469149367938e-01	4.964868706783169e-02
9.485539755760567e-01	3.666228059710319e-02	9.474898150194623e-01	3.745026957972177e-02
9.788566874094059e-01	2.380850649522536e-02	9.784290662963747e-01	2.429741981889855e-02
9.959649506960162e-01	1.034186239262945e-02	9.958843370550371e-01	1.054906616108520e-02

24 point quadrature rule for integrals of the form $\int_0^1 f(x) + g(x)\log(x+\bar{x})dx$, where $10^{-6} \leq \bar{x} \leq 10^{-5}$	
NODES	WEIGHTS
3.126377187332637e-04	4.136479682893960e-04
7.671264269072188e-04	5.068714387414649e-04
1.359575160544077e-03	7.008932527842778e-04
2.238313285727558e-03	1.110264922990352e-03
3.770276623583326e-03	2.120108385941761e-03
7.146583956092048e-03	5.249076343206215e-03
1.635515250548719e-02	1.450809938905405e-02
3.828062855101241e-02	2.987724029376343e-02
7.628984500206759e-02	4.593298717863718e-02
1.294255336121595e-01	5.987634475538021e-02
1.949876755761554e-01	7.065953519392547e-02
2.693852297828856e-01	7.729918562776261e-02
3.469762441631538e-01	7.556635340171830e-02
4.122748928895491e-01	5.234123638339037e-02
4.662499202239145e-01	6.532130125393047e-02
5.421402737123784e-01	8.188272080198840e-02
6.248832413655412e-01	8.237354882288161e-02
7.053258496784840e-01	7.795795664563893e-02
7.798841313231049e-01	7.076514272025076e-02
8.461534275163378e-01	6.145788741452406e-02
9.022312524979976e-01	5.044339641339403e-02
9.465899812310277e-01	3.807817118430632e-02
9.780549563823810e-01	2.471549011101626e-02
9.958125149101927e-01	1.073289672726758e-02

24 point quadrature rule for integrals of the form $\int_0^1 f(x) + g(x)\log(x+\bar{x})dx$, where $10^{-7} \leq \bar{x} \leq 10^{-6}$	
NODES	WEIGHTS
1.019234906342863e-04	1.349775051746596e-04
2.506087227631447e-04	1.663411550150506e-04
4.461429005344285e-04	2.328782111562424e-04
7.422845421202523e-04	3.804721779784063e-04
1.289196091156456e-03	7.930350452911450e-04
2.739287668024851e-03	2.600694722423854e-03
9.075168969969708e-03	1.212249113599252e-02
2.968005234555358e-02	2.946708975720586e-02
6.781742979962609e-02	4.647771960691390e-02
1.217792474402805e-01	6.095376889009233e-02
1.886625378438471e-01	7.224844725827559e-02
2.650602155844836e-01	7.986429603884565e-02
3.465113608339080e-01	8.143206462900546e-02
4.178374197420536e-01	5.040529357007135e-02
4.597624982511183e-01	5.592137651001418e-02
5.348065111487157e-01	8.398073572656715e-02
6.194640153146728e-01	8.402586870225486e-02
7.013481004172354e-01	7.922223490159952e-02
7.770386175609082e-01	7.177919251691964e-02
8.442211768916794e-01	6.227551999401272e-02
9.010272836291835e-01	5.108407212719758e-02
9.459409782755001e-01	3.854783279333592e-02
9.777905486554876e-01	2.501496650831813e-02
9.957622871041650e-01	1.086176801402067e-02

24 point quadrature rule for integrals of the form $\int_0^1 f(x) + g(x)\log(x+\bar{x})dx$, where $10^{-8} \leq \bar{x} \leq 10^{-7}$	
NODES	WEIGHTS
3.421721832247593e-05	4.559730842497453e-05
8.533906255442380e-05	5.840391255974745e-05
1.563524616155011e-04	8.761580900682040e-05
2.746612401575526e-04	1.617264666294872e-04
5.408643931265062e-04	4.433543035169213e-04
1.782382096488333e-03	3.116175111368442e-03
1.101243912052365e-02	1.655494413772595e-02
3.553172024884285e-02	3.242539256461602e-02
7.554170435463801e-02	4.734426463929677e-02
1.295711894941649e-01	6.032614603579952e-02
1.953213037793089e-01	7.069975187373848e-02
2.699680545714222e-01	7.806973621204365e-02
3.503697281371090e-01	8.216350598137868e-02
4.330838596494367e-01	8.261286657092808e-02
5.141801680435878e-01	7.883476216668445e-02
5.895097016206093e-01	7.157205125318401e-02
6.582708672338614e-01	6.703064468754417e-02
7.252543617887320e-01	6.706137273719630e-02
7.914154485613720e-01	6.449984116349734e-02
8.528383935857844e-01	5.775434959088197e-02
9.059696536862878e-01	4.812600239023880e-02
9.484664124578303e-01	3.661415869304224e-02
9.787863313133854e-01	2.386304203446463e-02
9.959482975155097e-01	1.038268695581411e-02

24 point quadrature rule for integrals of the form $\int_0^1 f(x) + g(x)\log(x+\bar{x})dx$, where $10^{-9} \leq \bar{x} \leq 10^{-8}$	
NODES	WEIGHTS
6.538987938840374e-06	1.500332421093607e-05
2.613485075847413e-05	2.367234654253158e-05
5.664183720634991e-05	4.007286246706405e-05
1.179374114362569e-04	9.497743501485505e-05
3.299119431334128e-04	4.619067037944727e-04
3.626828607577001e-03	9.985382463808036e-03
2.265102906572155e-02	2.805741744607257e-02
5.896796231680340e-02	4.404106103008398e-02
1.092496277855923e-01	5.548413172821072e-02
1.666701689499393e-01	5.693235996372726e-02
2.196889385898800e-01	5.087307376046002e-02
2.770352260035617e-01	6.593729718379782e-02
3.483163928268329e-01	7.335680008972614e-02
4.153287664837260e-01	5.675029500743735e-02
4.695624219668608e-01	6.117926027541254e-02
5.421129318998841e-01	8.004805067067550e-02
6.238832212055707e-01	8.196991767042605e-02
7.041842972237081e-01	7.800219127200407e-02
7.788817007552110e-01	7.097175077519494e-02
8.453877637047045e-01	6.171193295041172e-02
9.017178251963006e-01	5.068671319716005e-02
9.462999385952402e-01	3.827738423897266e-02
9.779333485180249e-01	2.485063762733620e-02
9.957890687155009e-01	1.079284973329516e-02

24 point quadrature rule for integrals of the form $\int_0^1 f(x) + g(x)\log(x + \bar{x})dx$, where $10^{-10} \leq \bar{x} \leq 10^{-9}$	
NODES	WEIGHTS
6.725520559705825e-06	8.128391913974039e-05
6.986424152770461e-06	-7.773900735768282e-05
1.217363416714366e-05	1.287386499666193e-05
2.677746219601529e-05	1.895577251914526e-05
5.597036348896741e-05	4.732580352158076e-05
2.729343280943077e-04	9.857909615386162e-04
9.445526806263141e-03	1.756872897270054e-02
3.556725025161542e-02	3.439422017906772e-02
7.765556668177810e-02	4.944188361792970e-02
1.336848150648662e-01	6.219733934997792e-02
2.011576917683550e-01	7.228007436918939e-02
2.772736854314979e-01	7.944986391225688e-02
3.590124362607926e-01	8.347646288178011e-02
4.430074035214462e-01	8.380433020121207e-02
5.247388219574510e-01	7.832768209682506e-02
5.961053238782420e-01	6.300796225242940e-02
6.547331131213409e-01	5.923406014585053e-02
7.192258519628951e-01	6.834293563803810e-02
7.874251789073102e-01	6.660337204499726e-02
8.505852012775045e-01	5.911988751082552e-02
9.047824617894323e-01	4.893575310568894e-02
9.479045131744448e-01	3.708256438629509e-02
9.785770588866582e-01	2.411463784693618e-02
9.959104692340199e-01	1.048087156697020e-02

24 point quadrature rule for integrals of the form $\int_0^1 f(x) + g(x)\log(x + \bar{x})dx$, where $10^{-11} \leq \bar{x} \leq 10^{-10}$	
NODES	WEIGHTS
2.828736694877886e-08	1.665602686704325e-05
2.302233157554212e-06	2.577419924039251e-06
5.853587143444178e-06	4.957941112780975e-06
1.451588770083244e-05	1.537074702915107e-05
9.711965099273031e-05	4.640075239797995e-04
9.004761967373848e-03	1.705687938176189e-02
3.442077924035546e-02	3.349724914160473e-02
7.543926781582543e-02	4.820210872119093e-02
1.300373356318913e-01	6.054547286337976e-02
1.955182772803384e-01	6.984354388121057e-02
2.683608546664295e-01	7.498721497014774e-02
3.430029178740901e-01	7.240620145057083e-02
4.085056107803621e-01	5.774925310174693e-02
4.660198270439085e-01	6.238505554837956e-02
5.336124745634699e-01	6.940394677081842e-02
5.985245800106473e-01	5.910843483407385e-02
6.564089719608276e-01	6.059752321454190e-02
7.216666024232565e-01	6.823362237770209e-02
7.893712241343741e-01	6.593839664071163e-02
8.518883782001418e-01	5.853014420243146e-02
9.055688088881344e-01	4.849217100974983e-02
9.483163097840529e-01	3.677417821170115e-02
9.787413692715607e-01	2.392585642844202e-02
9.959413203611228e-01	1.040149939671874e-02

24 point quadrature rule for integrals of the form $\int_0^1 f(x) + g(x)\log(x+\bar{x})dx$, where $10^{-12} \leq \bar{x} \leq 10^{-11}$	
NODES	WEIGHTS
6.147063879573664e-07	8.763741095000331e-07
2.102921984985835e-06	1.784696796288373e-05
2.188366117432289e-06	-1.795398395983826e-05
3.482602942694880e-06	5.117514567175025e-06
2.768001888608636e-05	1.698863549284390e-04
8.942779215792784e-03	1.701975216672032e-02
3.432218364237253e-02	3.346025972593909e-02
7.530931328026620e-02	4.817949622196712e-02
1.298983048592572e-01	6.055152664710045e-02
1.954020797117703e-01	6.988313730886592e-02
2.682970870436427e-01	7.504602275463067e-02
3.429540704041702e-01	7.230942674874111e-02
4.080399755202422e-01	5.705952259766429e-02
4.652562798154792e-01	6.265021180818162e-02
5.333220999210325e-01	6.993669694523695e-02
5.986982369433125e-01	5.937130986945129e-02
6.564773600603511e-01	6.026572020863567e-02
7.215159032030418e-01	6.815292696374753e-02
7.892098210760941e-01	6.596804590657802e-02
8.517672777806986e-01	5.857483758149194e-02
9.054906995605498e-01	4.853209199396977e-02
9.482736017320823e-01	3.680469214176019e-02
9.787238593479314e-01	2.394561701705853e-02
9.959379852805677e-01	1.041005152890511e-02

24 point quadrature rule for integrals of the form $\int_0^1 f(x) + g(x)\log(x+\bar{x})dx$, where $10^{-13} \leq \bar{x} \leq 10^{-12}$	
NODES	WEIGHTS
4.523740015216508e-08	4.418138082366788e-07
4.281855233588279e-07	4.389108058643120e-07
1.036900153156159e-06	9.539585150737866e-07
7.825849325746907e-06	5.823980947200484e-05
8.617419723953112e-03	1.634464263521301e-02
3.268881163637599e-02	3.129682188728318e-02
6.988441391437043e-02	4.212468617589480e-02
1.142202307676442e-01	4.505120897719191e-02
1.596471081833281e-01	4.769069780026684e-02
2.135336418959620e-01	6.038503382768951e-02
2.781100275296151e-01	6.695343672694180e-02
3.433392803364457e-01	6.163298712826237e-02
4.019960595528027e-01	5.877742624357513e-02
4.656415679416787e-01	6.800053637773440e-02
5.334880548894250e-01	6.516918103589647e-02
5.943298528903542e-01	5.853785375926075e-02
6.562968737815924e-01	6.639396325654251e-02
7.250343344601498e-01	6.948738324081696e-02
7.928820737781136e-01	6.538801703374268e-02
8.546103048745466e-01	5.761503751629250e-02
9.073762310762705e-01	4.761344859555310e-02
9.493253659835347e-01	3.607033097268266e-02
9.791606801267259e-01	2.345690720840071e-02
9.960217573957566e-01	1.019557402722854e-02

24 point quadrature rule for integrals of the form $\int_0^1 f(x) + g(x)\log(x + \bar{x})dx$, where $10^{-14} \leq \bar{x} \leq 10^{-13}$	
NODES	WEIGHTS
6.025980282801020e-08	9.079353616441234e-07
6.411245262925473e-08	-8.390389042773805e-07
1.862815529429129e-07	2.782460677485016e-07
2.029190208906422e-06	1.821115881362725e-05
8.902881307076499e-03	1.695809650660321e-02
3.420089035164912e-02	3.336370146025145e-02
7.508687525931594e-02	4.807898681796971e-02
1.295858123029775e-01	6.047672723211479e-02
1.950409815188335e-01	6.986774906175534e-02
2.679751967812604e-01	7.515608233194288e-02
3.428525062164689e-01	7.264249904037610e-02
4.080941369413548e-01	5.672507168477261e-02
4.646644511900009e-01	6.220316364524964e-02
5.328071517215501e-01	7.032362652293805e-02
5.978508749698001e-01	5.742730804758014e-02
6.521214523350964e-01	5.644075454541152e-02
7.134921670665336e-01	6.318643666150391e-02
7.679317896479284e-01	3.945995610428228e-02
8.029718487208403e-01	4.324200884758527e-02
8.551101435866935e-01	5.478223695609097e-02
9.067319102017767e-01	4.740856250832772e-02
9.487765213293372e-01	3.633314063504751e-02
9.788979796532736e-01	2.372788917088821e-02
9.959684838634199e-01	1.033036588606145e-02