A FULLY ADAPTIVE, HIGH-ORDER, FAST POISSON SOLVER FOR COMPLEX TWO-DIMENSIONAL GEOMETRIES*

DANIEL FORTUNATO†, DAVID B. STEIN‡, AND ALEX H. BARNETT§

Abstract. We present a new framework for the fast solution of inhomogeneous elliptic boundary value problems in domains with smooth boundaries. High-order solvers based on adaptive box codes or the fast Fourier transform can efficiently treat the volumetric inhomogeneity, but require care to be taken near the boundary to ensure that the volume data is globally smooth. We avoid function extension or cut-cell quadratures near the boundary by dividing the domain into two regions: a bulk region away from the boundary that is efficiently treated with a truncated free-space box code, and a variable-width boundary-conforming strip region that is treated with a spectral collocation method and accompanying fast direct solver. Particular solutions in each region are then combined with Laplace layer potentials to yield the global solution. The resulting solver has an optimal computational complexity of $\mathcal{O}(N)$ for an adaptive discretization with N degrees of freedom. With an efficient two-dimensional (2D) implementation we demonstrate adaptive resolution of volumetric data, boundary data, and geometric features across a wide range of length scales, to typically 10-digit accuracy. The cost of all boundary corrections remains small relative to that of the bulk box code. The extension to 3D is expected to be straightforward in many cases because the strip "thickens" an existing boundary quadrature.

Key words. fast Poisson solver, adaptivity, inhomogeneous PDE, complex geometry, boundary integral equation

AMS subject classifications. 65N35, 65N50, 65N38

1. Introduction.

Inhomogeneous elliptic partial differential equations (PDEs) play a central role in many areas of science and engineering, and often arise in conjunction with boundary conditions on complicated domains. The many fields in which this occurs include electrostatics in the presence of a space charge, elastostatics with a body load, steady-state heat or chemical reaction-diffusion equations, and (in the oscillatory case) acoustics and electromagnetics with a distributed source. Poisson type boundary value problems (BVPs) also arise as components of more elaborate solvers, where they may be called a large number of times. One example is that to solve a nonlinear elliptic BVP a (linear) Poisson solve is needed at each quasi-Newton iteration [6]. A second broad example area is time-dependent solvers, in which the inhomogeneity is derived from the solution at previous time steps. Common applications include: (1) in computational fluid dynamics, the pressure solve that follows each time step in the velocity formulation for incompressible Navier–Stokes [11] (reviewed in [30]); (2) in non-Newtonian fluids, internal stresses act as volumetric source terms which often may combine with advection to generate large gradients at surfaces [49]; and (3) implicit time stepping a parabolic PDE, such as the heat or Navier–Stokes equations, using Rothe's method [41], which demands solvers for the modified Helmholtz or modified Stokes (i.e., Brinkman) inhomogeneous BVPs, respectively. In time-dependent applications, the boundary geometry may itself be evolving.

*Submitted to the editors January 29, 2025.

Funding: This work was funded by the Simons Foundation.

†Center for Computational Mathematics & Center for Computational Biology, Flatiron Institute, New York, NY 10010 (dfortunato@flatironinstitute.org).

‡Center for Computational Biology, Flatiron Institute, New York, NY 10010 (dstein@flatironinsitute.org).

§Center for Computational Mathematics, Flatiron Institute, New York, NY 10010 (abarnett@flatironinstitute.org).

Let $\Omega \subset \mathbb{R}^2$ be a simply connected domain with smooth boundary $\Gamma = \partial\Omega$, and let $f(\boldsymbol{x})$ for $\boldsymbol{x} \in \Omega$ and $g(\boldsymbol{x})$ for $\boldsymbol{x} \in \Gamma$ be smooth functions. The model inhomogeneous elliptic PDE on Ω is the Poisson equation,

$$\Delta u(\boldsymbol{x}) = f(\boldsymbol{x}), \qquad \boldsymbol{x} \in \Omega, \tag{1.1a}$$

$$u(\boldsymbol{x}) = g(\boldsymbol{x}), \qquad \boldsymbol{x} \in \Gamma. \tag{1.1b}$$

Note that the technique that we present generalizes straightforwardly to above-mentioned other PDEs, and to other boundary conditions.

Many techniques exist to discretize and solve inhomogeneous elliptic BVPs on complex geometries. Perhaps most ubiquitous are methods that operate on an unstructured volumetric mesh of the domain interior, such as finite element methods (FEMs). While they are geometrically flexible, allow for variable coefficients, and are well supported with software, the size of the resulting linear systems scales with the number of interior unknowns needed to represent u, which may be huge when f has fine-scale variations. Furthermore, the linear systems may be ill-conditioned, and can become more so upon mesh refinement. Despite this, fast iterative methods such as multigrid [7] and sparse direct methods [12] have made this a popular approach. However, the cost of meshing the volume, especially to high order, may be prohibitive in the setting of time-stepping with evolving geometry. This has led to FEMs based on adaptive refinement of Cartesian meshes with cut cells [42, 43], giving decreased meshing cost. This extends ideas from immersed interface and level set methods for regular finite difference grids.

However, in the case of constant coefficients—as in (1.1a) and many of the applications mentioned above—a potential-theory-based alternative to direct discretization allows for a much reduced number of unknowns [38, 39]. One splits the solution u as the sum of a particular solution, v, and a homogeneous solution, w, satisfying

$$\Delta v(\boldsymbol{x}) = f(\boldsymbol{x}), \qquad \boldsymbol{x} \in \Omega, \tag{1.2}$$

and

$$\begin{aligned} \Delta w(\boldsymbol{x}) &= 0, && \boldsymbol{x} \in \Omega, \\ w(\boldsymbol{x}) &= g(\boldsymbol{x}) - v(\boldsymbol{x}), && \boldsymbol{x} \in \Gamma. \end{aligned} \tag{1.3}$$

The function $u = v + w$ then satisfies (1.1a) and (1.1b). Since the particular solution v is far from unique (it need not obey any specific boundary conditions), a numerically convenient choice can be made so that evaluation of v is fast and stable, as reviewed shortly.

Once v is evaluated, the homogeneous BVP (1.3) for w must be solved; this is conveniently done using boundary element or boundary integral methods, needing only a number of unknowns sufficient to discretize the boundary and its data [35]. This is typically orders of magnitude smaller than the system size needed with FEM. Iterative solvers such as GMRES converge rapidly when a representation is chosen that results in a Fredholm second-kind boundary integral equation (BIE), and high-order discretizations of the boundary integral operators are available [31]. Although the resulting linear system is dense, matrix-vector products may be performed via, e.g., a fast multipole method (FMM) [29] for the PDE fundamental solution, resulting in a solution time linear in the number of boundary unknowns. Subsequent evaluation of w in the interior may then also exploit an FMM. (Note that in the non-oscillatory case the homogeneous BVP could also be solved via boundary-concentrated FEM, with a similar cost scaling [34].)

The evaluation of a particular solution obeying (1.2) is our main topic. There have been two main prior approaches to this:

1. One idea [1, 8, 38, 39, 46–48] exploits the availability of fast solvers for the Poisson problem on various simple domains. Commonly this is a uniform grid on the rectangle, with some simple boundary condition, for which there exist fast low-order finite-difference solvers via cyclic reduction [9] or the 2D fast Fourier transform (FFT); note that if the solution is smooth, the latter may also be used for a spectrally-accurate solution on this uniform grid. If the complex geometry Ω lives within a simple domain R, then a particular solution on R—found using such a fast solver—is also a particular solution on Ω. (We note that fast Poisson solvers on the rectangle have recently been extended to nonuniform spectral discretizations [22], and to spheres and balls [53, 56], using low-rank alternating direction implicit methods.)

2. Another approach [2, 4, 18, 24, 44] finds a particular solution by convolution of f with the fundamental solution (free-space Green's function), the latter being $\frac{1}{2\pi}\log\frac{1}{\|\boldsymbol{x}\|}$ for the 2D Poisson equation. This has the advantage that *there is no linear solve*, merely an evaluation of a volume potential. Furthermore, the discretization of f may be spatially adaptive and thus more efficient than the above FFT solvers when f has fine-scale features. To evaluate this convolution in linear time, so-called "box codes" (or "VFMMs" [23]) have been developed—FMMs specialized to an adaptive quadrature grid living on a Cartesian quadtree and reaching near-FFT speeds [4, 18, 28].

However, in both of the above approaches, the particular solution v needs to be smooth enough on Ω to achieve the desired order of accuracy in the overall solution $u = v + w$. There have also been two major ways to tackle this smoothness requirement:

(a) The classical approach is to discretize the convolution over the domain, $v(\boldsymbol{x}) = -\int_\Omega \Phi(\boldsymbol{x},\boldsymbol{y})f(\boldsymbol{y})d\boldsymbol{y}$, where Φ is either the simple-domain or free-space Green's function. If done accurately, this generally gives v as smooth in Ω as the solution u (see Remark 4.4). However, efficient high-order accurate approximation of the volume potential on the complex geometry is challenging. Early work in the uniform finite-difference setting extended f by zero in $R\setminus\Omega$ and then added careful near-boundary node corrections to recover the bulk convergence order [38, 39]. Plain use of a box code does not solve the problem, since high accuracy would demand an excessive level of adaptive refinement towards Γ. Recent works evaluate potentials from triangular mesh elements or irregular cut cells, to medium or high order, by conversion to line integrals [2], by density interpolation [3], by a two-level Ewald-type heat potential split and the nonuniform FFT [24], or by so-called *anti-Laplacians* (Green's 3rd identity applied elementwise) [44]. Each of these works uses a fast algorithm to achieve $\mathcal{O}(N)$ scaling in N, the number of discretization nodes. However, of these four, in their current forms, only the work of Shen and Serkh [44] could preserve linear scaling in a truly adaptive mesh. Furthermore, the cost of generating the needed unstructured mesh is notoriously high, especially in 3D.

(b) An alternative is to use *function extension* (extrapolation outside Ω), meaning

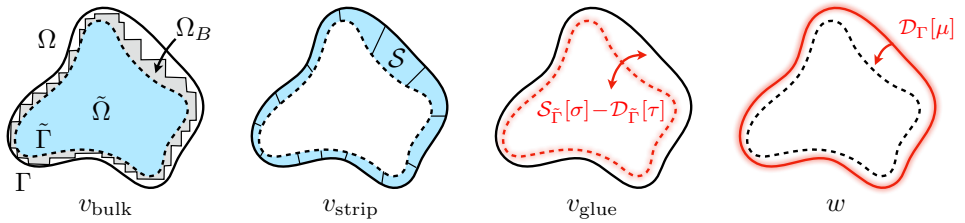

FIG. 1.1. *Overview of our adaptive Poisson solver for a domain* Ω. *The left three terms comprise the particular solution* $v = v_{\text{bulk}} + v_{\text{strip}} + v_{\text{glue}}$, *while* w *is the homogeneous solution. In the left two panels, blue indicates where the particular solution is evaluated. The grey region* (Ω_B, *which includes* $\tilde{\Omega}$) *created by well-separated box truncation generates the volume source for* v_{bulk}. *In the right two panels, layer densities are shown in red, and evaluated throughout* Ω. *See Algorithm* 1.1.

the construction of a function f_e with $f_e = f$ in Ω and with some specified degree of smoothness throughout an enclosing simple domain R. Then $v(\boldsymbol{x}) = -\int_R \Phi(\boldsymbol{x},\boldsymbol{y}) f_e(\boldsymbol{y}) d\boldsymbol{y}$ is a particular solution that is smooth in Ω, and whose evaluation (via either a fast Poisson solver or free-space box code) does not require refinement near Γ. This has been used with non-adaptive spectral FFT Poisson solvers via fixed-order immersed-boundary smooth extension (IBSE) [47, 48], high-order partition-of-unity extension (PUX) using radial basis functions [1,25,26], or 1D extension along normals [8,16]. In the adaptive free-space box code setting, C^0 extension has been done via an exterior Laplace BVP solution [4], or at high order by an adaptive variant of PUX [23]. Aside from the extra computational cost (sometimes involving linear solves), the idea has two key difficulties: extrapolation is inherently ill-conditioned [14] (becoming more so the higher the order of smoothness [16, Tbl. 1]), generating large values outside Ω; and, close-to-touching boundaries Γ may simply not allow room for a single-valued smooth f_e to exist in $R \setminus \Omega$.

The obstacles inherent in both above approaches to the creation of a smooth v motivated one of the authors recently to propose "function intension" [46], namely the smooth roll-off to zero of the source term f in a constant-width boundary strip region $\mathcal{S}$ *inside* Ω, followed by an FFT-based Poisson solve to generate a v valid only in $\Omega \setminus \mathcal{S}$; a distinct particular solution is used within $\mathcal{S}$.

In this work, we introduce a high-order linear-scaling Poisson solver for complex geometries that is fully adaptive in handling both the inhomogeneity and the boundary, while avoiding both volume potentials on the complex geometry and function extension. Building upon [46], our solver computes a particular solution v using a simple decomposition of Ω into two regions (see Figure 1.1):

- a bulk region $\tilde{\Omega} \subset \Omega$ with boundary $\tilde{\Gamma} = \partial\tilde{\Omega}$, in which we can use a free-space box-code to evaluate a particular solution v_{bulk} satisfying

$$\Delta v_{\text{bulk}}(\boldsymbol{x}) = f(\boldsymbol{x}), \quad \boldsymbol{x} \in \tilde{\Omega};$$

- a thin *variable-width* boundary-fitted strip region $\mathcal{S} = \Omega \setminus \tilde{\Omega}$, with boundary $\Gamma \cup \tilde{\Gamma}$, in which we propose high-order curvilinear spectral collocation to compute a particular solution v_{strip} satisfying

$$\Delta v_{\text{strip}}(\boldsymbol{x}) = f(\boldsymbol{x}), \quad \boldsymbol{x} \in \mathcal{S}.$$

As these particular solutions are piecewise defined, they generally possess jumps in both value and normal derivative across the interface $\tilde{\Gamma}$. We correct for these jumps by adding single- and double-layer Laplace potentials v_{glue} along $\tilde{\Gamma}$ to v_{bulk} and v_{strip}, which effectively patch the solutions together[1] to recover a globally smooth particular solution v defined everywhere inside Ω. Unlike in the non-adaptive work [46] where this partitioning is defined by a constant shift in the local normal direction on Γ, we seek to handle multiscale geometries and inhomogeneities which may require adaptivity. Thus, care must be taken to define the curve $\tilde{\Gamma}$ in a smooth geometry-aware fashion (see subsection 4.1). This process is outlined in Algorithm 1.1. We also improve upon [46] by replacing the idea of "function intension" with "well-separated box truncation," which is simpler, faster, and amenable to rigorous analysis (see Theorem 1).

The paper is structured as follows. In section 2, we present our standard discretization of the boundary Γ into a set of high-order panels. Section 3 briefly describes how a homogeneous solution w may be computed using standard potential-theoretic techniques. Subsection 4.1 describes an algorithm to construct a smoothly-defined strip region in an adaptive fashion. In subsection 4.2, we describe how v_{bulk} can be efficiently computed as a truncated volume potential and analyze the effect that truncation has on its smoothness. Subsection 4.3 describes a curvilinear, composite spectral collocation method to compute v_{strip}. In subsection 4.4, we show how v_{bulk} and v_{strip} may be patched together using layer potentials to yield a globally smooth particular solution v. We conclude with numerical results and examples in section 5.

Algorithm 1.1 Adaptive solution of interior Dirichlet Poisson problem

Input: Boundary panelization of Γ, inhomogeneity f, Dirichlet data g
Output: Solution u to (1.1a)–(1.1b)

1: Construct fictitious boundary $\tilde{\Gamma}$ (see subsection 4.1).
2: **if** f is unresolved on any elements of $\mathcal{S}$ **then**
3: ⌊ Split the corresponding panels of Γ and **goto** 1
4: Construct quadtree approximation to f with truncation (see subsection 4.2.1).
5: Compute bulk solution v_{bulk} using a truncated box code (see subsection 4.2.2).
6: Compute strip solution v_{strip} using spectral collocation (see subsection 4.3).
7: Compute the jumps in value and normal derivative between v_{bulk} and v_{strip}:

$$\tau = v_{\text{bulk}}|_{\tilde{\Gamma}} - v_{\text{strip}}|_{\tilde{\Gamma}}, \qquad \sigma = \partial_{\boldsymbol{n}} v_{\text{bulk}}|_{\tilde{\Gamma}} - \partial_{\boldsymbol{n}} v_{\text{strip}}|_{\tilde{\Gamma}}.$$

8: Define a piecewise harmonic function to correct the jumps (see subsection 4.4):

$$v_{\text{glue}}(\boldsymbol{x}) \coloneqq \mathcal{S}_{\tilde{\Gamma}}[\sigma](\boldsymbol{x}) - \mathcal{D}_{\tilde{\Gamma}}[\tau](\boldsymbol{x}).$$

9: Define the particular solution v:

$$v(\boldsymbol{x}) = \begin{cases} v_{\text{bulk}}(\boldsymbol{x}) + v_{\text{glue}}(\boldsymbol{x}), & \boldsymbol{x} \in \tilde{\Omega}, \\ v_{\text{strip}}(\boldsymbol{x}) + v_{\text{glue}}(\boldsymbol{x}), & \boldsymbol{x} \in \mathcal{S}. \end{cases}$$

10: Compute the homogeneous solution w by solving (1.3) (see section 3).
11: **return** $u = v + w$

[1] Recall that, since the PDE is second-order elliptic, values and normal derivatives are precisely the Cauchy data needed for smooth matching of solutions [19].

2. Boundary discretization and geometry format. Our Poisson solver needs as input a description of the smooth domain boundary Γ, a forcing function $f : \Omega \to \mathbb{R}$, and a function $g : \Gamma \to \mathbb{R}$ evaluating the boundary data. We assume that the boundary is supplied as a set of disjoint panels $\{\gamma_k\}_{k=1}^{n_{\text{panel}}}$ that resolve the boundary and such that $\Gamma = \bigcup_k \gamma_k$. Specifically, the kth panel is described by a set of user-supplied nodes $\{\boldsymbol{x}_{j,k}\}_{j=1}^{p+1}$, each node being $\boldsymbol{x}_{j,k} = (x_{j,k}, y_{j,k}) \in \mathbb{R}^2$, that are assumed to be the image of the standard Gauss–Legendre nodes $\{t_j\}_{j=1}^{p+1}$ on $[-1, 1]$ under some smooth map $\boldsymbol{\Lambda}_k$. Then $\boldsymbol{\Lambda}_k([-1, 1]) = \gamma_k$. Such an input format is typical for high-order boundary integral solvers [57]. We typically choose the order p in the range 10–20.

Remark 2.1. For the purposes of numerical tests we will need to generate resolved panel discretizations of various boundaries Γ described by an analytic or image-extracted function. This generation is common in the boundary integral equation setting and may be automatically performed in an adaptive fashion [57]. In practice, identifying $\mathbb{C}$ with $\mathbb{R}^2$, we construct an adaptive panelization from a given parametrized curve $z(t) = x(t) + \mathrm{i}y(t)$ by numerically resolving to a specified tolerance ϵ a set of monitor functions: the curve $z(t)$, its parametrization "speed" $|z'(t)|$, and the bending energy density $|\operatorname{Im}(z''(t)/z'(t))|^2/|z'(t)|$ [57]. For each monitor function ζ in this set and on each panel γ_k, we compute the $2p + 1$ Legendre coefficients $\{\hat{\zeta}_{j,k}\}_{j=1}^{2p+1}$ of ζ on γ_k. Then we say that ζ is resolved on γ_k if

$$\sqrt{\frac{1}{p}\frac{\sum_{j=p+1}^{2p+1}\hat{\zeta}_{j,k}^2}{\sum_{j=1}^{p}\hat{\zeta}_{j,k}^2}} < \epsilon,$$

i.e., if the tail of the Legendre coefficients has decayed to a relative tolerance of ϵ. If ζ is not resolved on γ_k, the panel is further subdivided. If all panels $\{\gamma_k\}_{k=1}^{n_{\text{panel}}}$ are resolved to the given ϵ, we say that the panel set resolves Γ. Note that in tests of the solver, the panel nodes $\{\boldsymbol{x}_{j,k}\}_{j=1}^{p+1}$ for $1 \le k \le n_{\text{panel}}$ alone are passed in to describe the geometry; the underlying parametrization is discarded.

For boundary integrals with respect to arc length measure ds we will need a quadrature weight $w_{j,k}$ for each of the user-supplied set of panel nodes. These weights are created as follows. Let D_{leg} be the size-$(p+1)$ square spectral differentiation matrix that maps values to derivatives on the standard Gauss–Legendre nodes $\{t_j\}_{j=1}^{p+1}$; the jth column is given by the derivative of the jth Lagrange basis function evaluated at the nodes. For the kth panel, stacking its node coordinates as vectors and using MATLAB-style notation $x_{:,k} \coloneqq \{x_{j,k}\}_{j=1}^{p+1}$, the speed $\|\boldsymbol{x}'(t)\|$ at the jth node is approximated by

$$s_{j,k} = \sqrt{[(D_{\text{leg}}x_{:,k})_j]^2 + [(D_{\text{leg}}y_{:,k})_j]^2}. \tag{2.1}$$

The quadrature weight is then $w_{j,k} = W_j s_{j,k}$, where W_j are the Gauss–Legendre weights for $[-1, 1]$. Then, to high-order accuracy, for any smooth function $h : \Gamma \to \mathbb{R}$ the change of variables from arc length to t on each panel shows that

$$\int_\Gamma h(\boldsymbol{x}) ds_{\boldsymbol{x}} \;\approx\; \sum_{k=1}^{n_{\text{panel}}} \sum_{j=1}^{p+1} w_{j,k} h(\boldsymbol{x}_{j,k}). \tag{2.2}$$

The solver also needs access to the underlying arc-length parametrization induced by the set of user-supplied Gauss–Legendre panel nodes. For each panel, say the

kth, this is done as follows. There exists a spectral antidifferentiation matrix $A_{\text{leg}} \in R^{(p+1)\times(p+1)}$ that maps derivatives to values (up to an overall constant) on the standard Gauss–Legendre nodes $\{t_j\}_{j=1}^{p+1}$; this matrix could be constructed via applying quadrature to Lagrange basis functions, but is more easily found via the pseudoinverse of D_{leg}. Then the set $a_{j,k} = [A_{\text{leg}} s_{:,k}]_j$ are approximate within-panel arc-length coordinates of the user-supplied nodes. From this, arc-length coordinates $a(t)$ of an arbitrary t in the panel may be found by polynomial interpolation from the nodes. In particular, the panel arc length is then $a_k = a(1) - a(-1)$. By cumulatively summing arc lengths from a fixed fiduciary panel endpoint, a global approximate arc-length parametrization of Γ is then easily built. We will denote this by $\mathbf{\Lambda}(a)$, so that $\mathbf{\Lambda}([0, L]) \approx \Gamma$ to accuracy ϵ, where $L = \sum_{k=1}^{n_{\text{panel}}} a_k$ is the perimeter. Note that we do not modify the user-supplied nodes (i.e., we do not reparametrize the given nodes to be Gauss–Legendre in arc length); in subsection 4.1 we will only need the $a_{j,k}$ and a_k computed above.

We further require that panels are sufficiently far away from their non-neighboring panels. Specifically, the distance between a panel and any non-neighboring panel should be larger than three times the arc length of the panel [57]. The given panelization is refined until this criterion is met. We use a k-d tree to efficiently calculate approximate panel distances [51]. Finally, we require that the user-supplied panelization be level restricted, so that no two neighboring panels differ in arc length by more than a factor of two. If the given panelization does not satisfy this criterion, we refine the panelization until level restriction is satisfied.

3. Potential theory for the homogeneous problem. Our main focus in the present work is the development of a fast, adaptive, and high-order accurate scheme to compute a particular solution v to the inhomogeneous PDE in a complex geometry. However, we also need potential-theoretic techniques for computing the homogeneous solution w, which are standard and briefly described here. The homogeneous solution w to the interior Dirichlet problem (1.3) is represented as the double-layer potential on Γ induced by an unknown density function μ,

$$w(\boldsymbol{x}) = \mathcal{D}_\Gamma[\mu](\boldsymbol{x}) := \int_\Gamma \frac{\partial \Phi(\boldsymbol{x}, \boldsymbol{y})}{\partial \boldsymbol{n}_{\boldsymbol{y}}} \mu(\boldsymbol{y})\, ds_{\boldsymbol{y}}, \tag{3.1}$$

where $\Phi(\boldsymbol{x}, \boldsymbol{y}) = \frac{1}{2\pi} \log \frac{1}{\|\boldsymbol{x}-\boldsymbol{y}\|}$ is the fundamental solution of Laplace's equation in two dimensions. Using the jump relations of the double layer potential [35, Thm. 6.18] leads to a second-kind integral equation for the unknown density μ:

$$-\frac{1}{2}\mu(\boldsymbol{x}) + \mathcal{D}_\Gamma[\mu](\boldsymbol{x}) = g(\boldsymbol{x}) - v(\boldsymbol{x}), \qquad \boldsymbol{x} \in \Gamma. \tag{3.2}$$

For a smooth boundary the operator (3.1) has a smooth kernel, so that a plain Nyström discretization [35, Sec. 12.2; 31] using the quadrature scheme (2.2) is high-order accurate. The resulting linear system for the values of μ at the set of panel nodes is well-conditioned. We solve it using GMRES, and accelerate matrix-vector products at each iteration with the 2D Laplace FMM [29] implemented by FMMLIB2D [27]. For evaluation of (3.1) close to the boundary, and for evaluation of the blocks of the system matrix between non-adjacent panels that fall sufficiently close to each other (e.g., in the case of re-entrant or close-to-touching geometries) we use the specialized panel quadrature scheme of Helsing and Ojala [32]. In practice, we use the Helsing–Ojala scheme for any target point or target panel whose distance to the source panel is less than 1.2 times the length of the source panel.

4. Constructing a particular solution. We now describe our piecewise construction of a particular solution satisfying (1.2). Recall from Algorithm 1.1 that we form particular solutions in two regions separated by a fictitious curve $\tilde{\Gamma}$; see Figure 1.1. We begin with the construction of $\tilde{\Gamma}$.

4.1. Defining the fictitious curve. Given the curve Γ defined by a panelization $\{\gamma_k\}_{k=1}^{n_{\text{panel}}}$, we aim to compute another panelized curve $\tilde{\Gamma}$ lying inside Γ. The region between Γ and $\tilde{\Gamma}$ then defines the strip region $\mathcal{S}$. To obtain a high-order accurate and scalable method, there are a number of criteria that $\tilde{\Gamma}$ should satisfy. The fictitious curve should be:

- as smooth as the given curve Γ (or high-order accuracy may be lost);
- resolved using $\mathcal{O}(n_{\text{panel}})$ panels (or optimal complexity may be lost);
- not too close to Γ (or the box code would have to adapt to overly small scales); and,
- not too far from Γ (or the strip region would require too many nodes in the radial direction).

For simplicity, and since it induces curvilinear (as opposed to highly skew) coordinates in the strip, we use extension in the local normal direction on Γ to define $\tilde{\Gamma}$. We do not anticipate that a significant reduction in degrees of freedom is possible with a more complicated scheme. Thus we define the fictitious curve $\tilde{\Gamma}$ according to a positive local width function, $h(t) : [0, L] \mapsto \mathbb{R}^+$. Let $\boldsymbol{n}_{j,k}$ be the outward pointing unit normal vector at the jth node of panel k. Then, given a width function $h(t)$, $\tilde{\Gamma}$ may be defined via perturbation in the normal direction, with each panel's nodes given by

$$\tilde{\boldsymbol{x}}_{j,k} = \boldsymbol{x}_{j,k} - h(t_{j,k})\,\boldsymbol{n}_{j,k} \tag{4.1}$$

for $j = 1, \ldots, p+1$ and $k = 1, \ldots, n_{\text{panel}}$, where $t_{j,k} \coloneqq A_k + a_{j,k}$ are the arc-length parameters of the user-supplied nodes. If h is a smooth function (or at least as smooth as Γ), then the fictitious curve $\tilde{\Gamma}$ will be as smooth as Γ.

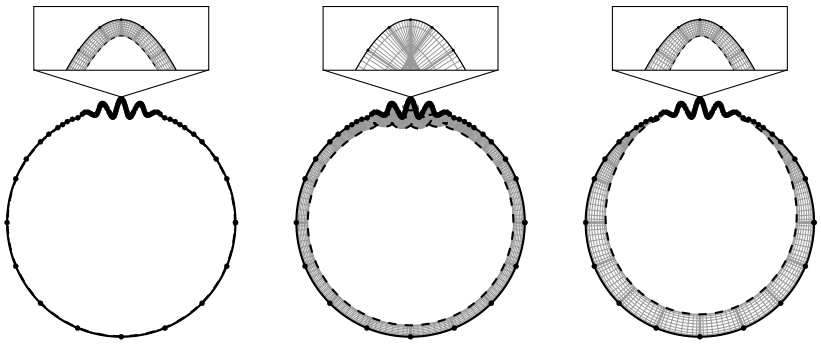

FIG. 4.1. *Definition of the fictitious curve $\tilde{\Gamma}$ by inward normal extension. (Left) On a multiscale geometry, using a uniform strip width based on the smallest length scale results in an unnecessarily thin strip where the panel size is large. (Center) Using a width based on the largest length scale leads to self-intersection at the smallest length scales. (Right) Using a width function $h(t)$ that smoothly adapts to local panel size gives a strip that correctly resolves all geometric features.*

It remains to set up a width function h that satisfies the above criteria. Figure 4.1 depicts the effects that different choices of h can have on an adaptive geometry. Assuming that the given panelization correctly resolves all multiscale features of Γ, and

has been post-processed so that all non-neighboring panels are sufficiently separated, these criteria suggest that h should be proportional to the local panel size.

To define a suitable width function h, we begin with a crude, piecewise linear interpolation of local panel size. Let a_k be the arc length of panel k, constructed as in section 2, and define the panel endpoints $A_k \coloneqq \sum_{k'=1}^{k-1} a_{k'}$ so that $\mathbf{\Lambda}([A_k, A_{k+1}]) = \gamma_k$. Define the local linear functions, $h_k^{\text{lin}}(t) = a_k + \frac{a_{k+1}-a_k}{a_k}(t - A_k)$. Then our starting point for a width function that adapts to local panel size is given by piecewise linear interpolation, i.e., $h(t) = h_k^{\text{lin}}(t)$ for $t \in [A_k, A_{k+1}]$ and $k = 1, \ldots, n_{\text{panel}}$. See Figure 4.2(b). (Note that, as written, the value at the endpoint $h(A_k) = a_k$ matches the panel size to its right rather than left; in practice, once A_k are computed, we then replace a_k with a local average of the panel sizes from the $2K$ neighboring panels centered on A_k, for some parameter $1 \le K \le 5$.) However, this $h(t)$ is continuous but not generally C^1, due to kinks at endpoints. A rounded approximation to the kink occurring between panels γ_{k-1} and γ_k at the point A_k can be constructed as

$$h_k^{\text{round}}(t) = a_k + \tfrac{a_k - a_{k-1}}{a_{k-1}}(t - A_k) + \left(\tfrac{a_{k+1}-a_k}{a_k} - \tfrac{a_k - a_{k-1}}{a_{k-1}}\right) r_k(t - A_k),$$

where the middle term sets the slope for $t < A_k$ and the last term smoothly adjusts this slope to that of h_k^{lin} when $t > A_k$. Here r_k is a "softplus" (or "smooth ReLU") function with *panel-dependent* length scale $1/\beta_k$,

$$r_k(t) = \tfrac{1}{\beta_k} \log\left(1 + e^{\beta_k t}\right),$$

which blends from $r_k(t) \approx 0$, when $t \ll -1/\beta_k$, to $r_k(t) \approx t$, when $t \gg 1/\beta_k$. For a length scale commensurate with local panel size, we choose $\beta_k = 2/(a_{k-1} + a_k)$.

We then combine the "inner" rounded approximations to each kink into smooth functions defined on each panel, subtracting off the "outer" expansion h^{lin} in the manner of matched asymptotics [36, §3.3.3]. Specifically, on each panel γ_k we add together the rounded approximations from a small set of $2K$ neighboring panels, $\{\gamma_{k-K}, \ldots, \gamma_{k-1}, \gamma_{k+1}, \ldots, \gamma_{k+K}\}$, so that

$$h_k^{\text{neigh}}(t) = h_{k+K+1}^{\text{round}}(t) + \sum_{j=k-K}^{k+K} \left(h_j^{\text{round}}(t) - h_j^{\text{lin}}(t)\right).$$

In practice, we choose $1 \le K \le 5$. The function $h_k^{\text{neigh}}(t)$ is locally a smooth function on panel γ_k and its adjacent $2K$ neighbors.

Finally, in order to arrive at a width function $h(t)$ that is globally smooth across all panels, we blend together the functions $\{h_j^{\text{neigh}}(t)\}_{j=k-K}^{k+K}$ on panel γ_k using a partition of unity, yielding

$$h(t) = \sum_{j=k-K}^{k+K} \hat{w}_j(t) h_j^{\text{neigh}}(t), \qquad t \in [A_k, A_{k+1}], \tag{4.2}$$

with normalized blending functions $\hat{w}_k(t) = w_k(t) / \sum_{j=k-K}^{k+K} w_j(t)$, $w_k(t) = \rho_c(\frac{t - A_k}{\delta})$, and bump function $\rho_c(t)$ taken to be the prolate spheroidal wavefunction of order zero and bandwidth c [45], implemented by `pswf.m` in Chebfun [15]. We typically choose $\delta = \frac{a_{k-1}+a_k}{8}$ and $c = 30$. It is (4.2) that is used as $h(t)$ in (4.1).

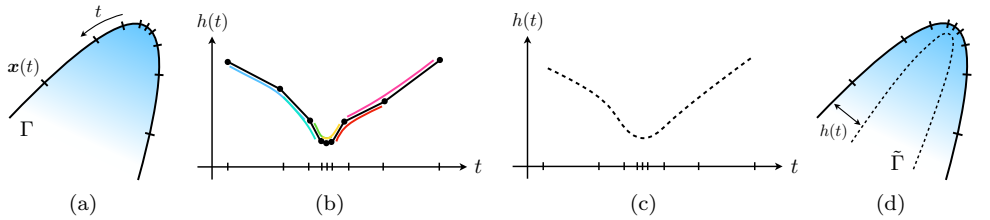

FIG. 4.2. *Schematic for creating a smooth fictitious curve that adapts to local panel size. (a) The input panelization, given as a set of high-order Gauss–Legendre nodes sampled from the original curve* Γ. *(b) A crude piecewise linear width function* $h(t)$ *is constructed from average local panel size (black circles). A rounded approximation to each kink is created, with the amount of smoothing commensurate with the local panel size (colored curves). (c) The rounded approximations are blended together into a globally smooth width function* $h(t)$ *using a partition of unity. (d) The original panelization is perturbed in the normal direction by* $h(t)$*, yielding a smooth fictitious curve* $\tilde{\Gamma}$ *that adapts to local panel size.*

4.2. The bulk problem. With a suitably defined fictitious curve $\tilde{\Gamma}$ separating the bulk region from the strip region, we now turn to computing a particular solution v_{bulk} in the bulk region $\tilde{\Omega}$, meaning that it satisfies

$$\Delta v_{\text{bulk}}(\boldsymbol{x}) = f(\boldsymbol{x}), \quad \boldsymbol{x} \in \tilde{\Omega}. \tag{4.3}$$

For this, our procedure is similar to that of a box code (fast evaluation of a volume potential on a grid adapted to resolve f [4, 17, 18, 28]), but with a modified criterion for refinement, as well as for exclusion of well-separated boxes. Its result will be a v_{bulk} that is a high-order approximation to

$$v_{\text{bulk}}(\boldsymbol{x}) = -\int_{\Omega_B} \Phi(\boldsymbol{x}, \boldsymbol{y})\, f(\boldsymbol{y})\, d\boldsymbol{y}, \tag{4.4}$$

where Ω_B is some union of boxes such that $\tilde{\Omega} \subset \Omega_B \subset \Omega$, as sketched by the grey region with irregular boundary in the left panel of Figure 1.1. The requirement $\Omega_B \subset \Omega$ simply comes from the fact that f is not defined outside Ω, and, for reasons discussed in the introduction, extending f smoothly outside of Ω has difficulties that we wish to avoid.

It is clear that (4.4) would satisfy (4.3) for *any* choice of source domain $\Omega_B \subset \Omega$ that entirely covers $\tilde{\Omega}$. However, what is needed is a choice of Ω_B that results in v_{bulk} being smooth on $\tilde{\Gamma}$ and in $\tilde{\Omega}$. This would indeed hold if $\Omega_B = \tilde{\Omega}$, but would require excessive box refinement around $\tilde{\Gamma}$ down to the scale of the desired tolerance ϵ (thus adding $\mathcal{O}(\epsilon^{-1} \log \epsilon^{-1})$ boxes to resolve the boundary), hence would be very inefficient. The same would be true at the other extreme case $\Omega_B = \Omega$, with the excessive refinement now occurring at Γ. Our choice of Ω_B lies between these two extremes, and is based on selectively truncating boxes in the strip region. Despite its seemingly awkward shape, we will show that it retains high-order accuracy while inducing minimal extra box refinement.

Remark 4.1. The known values of f outside $\tilde{\Omega}$ may also be used to obtain a globally smooth function $\tilde{f}$ through multiplication by a high-order smooth blending function which rolls off to zero in the strip region $\mathcal{S}$, in the style of "function intension" [46]. Since $\tilde{f}$ is now less smooth than f, this would typically require smaller adaptive boxes in $\mathcal{S}$, and hence would be less efficient than the box truncation that we propose.

4.2.1. Constructing a quadtree approximation to f. The problem defines the domain Ω and the source function f available on Ω, and we have at this point fixed the fictitious split of Ω into bulk $\tilde{\Omega}$ and strip $\mathcal{S}$ along the curve $\tilde{\Gamma}$. The user specifies the order p and tolerance $\epsilon > 0$. We will take $p = 16$ in all examples. We now aim to construct a quadtree consisting of a set of boxes $\{B_k\}_{k=0}^{n_{\text{box}}-1}$, where each box is either a parent or a leaf, and f is approximated by an order-p polynomial up to a tolerance ϵ on leaf boxes.

The quadtree construction process starts from a single box $B_0 \supset \Omega$ containing the entire domain, and proceeds to recursively split a box B_k into four child boxes according to two criteria:

1. **Function resolution**. If $B_k \subset \Omega$, then f may be fully evaluated on B_k and its Chebyshev coefficients $\hat{f}_{ij}^k$ computed according to the order-p approximation,

$$f(x,y) \approx \sum_{i=1}^{p+1}\sum_{j=1}^{p+1} \hat{f}_{ij}^k\, T_j^k(x)\, T_i^k(y), \qquad (x,y) \in B_k,$$

   where T_j^k is the jth Chebyshev polynomial of the first kind scaled to the domain of box B_k. If the coefficients of f on B_k do not decay below the given tolerance ϵ, then B_k is marked for refinement. If B_k is entirely outside of Ω, it is discarded and does not contribute. However, if only part of B_k falls outside Ω (i.e., the boundary Γ cuts the box), then extending f by zero outside Ω will cause its polynomial approximation to fail. This necessitates the use of the second criterion.
2. **Separation from truncation**. If B_k intersects Γ, then the corresponding volume potential computed on B_k will not be an accurate particular solution even in the part of the box lying in Ω. One simple option is to set the coefficients $\hat{f}_{ij}^k$ to zero in B_k. But how does this truncation affect the neighbors of B_k? To examine this, we perform an experiment on a simple domain in Figure 4.3. Starting from an inhomogeneity f known everywhere inside a circle, we construct a quadtree which approximates f to high order and zero any boxes which overlap the outside of the domain (Figure 4.3, top row). We then compute the volume potential v_{bulk} induced by this truncated data and evaluate its residual (Figure 4.3, middle row). One might expect that the truncation generally induces *fake corner singularities* in the resulting v_{bulk} which cause large residuals on any adjacent box—and indeed this is what is seen. Boxes with large residual, as well as those that were cut by Γ, are marked for refinement. Moving left-to-right across the figure, each column shows a recomputation after boxes marked for refinement are subdivided. To examine the global effect, we solve a test problem inside $\tilde{\Omega}$ using v_{bulk} as the particular solution, and measure the error against a known reference solution (Figure 4.3, bottom row). This shows that the global error in $\tilde{\Omega}$ is controlled by this truncated-neighbor effect. The final quadtree suggests that any box that is well separated from truncation (i.e., having no truncated neighbors) will be accurate. Hence, for all boxes in $\tilde{\Omega}$ to be accurate, all boxes touching the strip region $\mathcal{S}$ should be refined until their diameter is less than half of the strip width. This is our second criterion.

A quadtree construction process based on these criteria is given in Algorithm 4.1. As a consequence of the size-based splitting criterion, all truncation is pushed to the *outer half* of the strip $\mathcal{S}$. As a final post-processing step, we perform a 2:1 balance of the resulting quadtree [13, 37], so that every box is no more than one refinement

level apart from its neighbors. The truncated volume potential is computed using the box code available in the `boxcode2d` library [5], which is called by the `treefun` package [20].

Remark 4.2. In practice, we choose to utilize the Chebyshev coefficients of f even on cut boxes (where f is taken to be extended by zero outside Ω), rather than setting the function to zero uniformly over the entire cut box. Though the resulting polynomial approximation of f on these cut boxes is unresolved, we have found that including such cut values of f can increase the overall accuracy by half a digit or more, by effectively moving the truncation location slightly further from $\tilde{\Gamma}$.

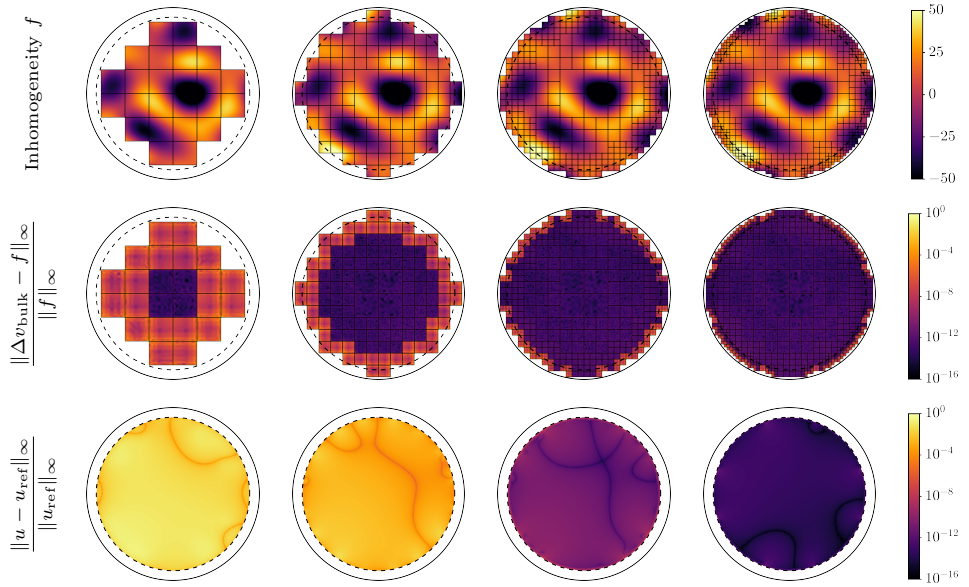

FIG. 4.3. *Visualization of the effect of truncation when computing a volume potential. (Top row) An inhomogeneity f known everywhere inside the domain (solid circle) is converted to a quadtree, with each leaf approximating f to high order. Any boxes overlapping the outside of the domain are discarded, as we assume f is known only inside the domain. (Middle row) The volume potential v_{bulk} induced by this truncated data is computed with a box code and the relative error of the residual is used as an indicator for refinement. Boxes with large residuals are subdivided and solutions are successively computed as we move column-wise to the right. (Bottom row) To visualize the effect of truncation on the final solution, a test problem is solved inside the fictitious region (dotted circle) using v_{bulk} as the particular solution and the computed solution u is compared to a known reference solution u_{ref}. The final quadtree suggests that boxes touching the strip region should be refined until their diameter is less than half of the strip width.*

4.2.2. Smoothness of truncated volume potentials. We now analyze the effect of truncating the volume potential in terms of the smoothness of the computed particular solution on each panel of $\tilde{\Gamma}$. Recall from the introduction that a classical particular solution using the full solution domain Ω is

$$v(\boldsymbol{x}) = -\int_{\Omega} \Phi(\boldsymbol{x}, \boldsymbol{y})\, f(\boldsymbol{y})\, d\boldsymbol{y}. \tag{4.5}$$

This is smooth in the interior if f is smooth, as follows from standard elliptic regularity [19, §6.3.2] since $\Delta v = f$ in Ω.

Algorithm 4.1 Adaptive construction of truncated quadtree approximation to f

Input: Inhomogeneity f, boundaries Γ and $\tilde{\Gamma}$, bounding box $B_0 \supset \Omega$, tolerance $\epsilon > 0$
Output: Quadtree approximation to f as a set of boxes $\{B_k\}_{k=0}^{n_{\text{box}}-1}$

Initialize stack of boxes $B \leftarrow \{B_0\}$ and $n_{\text{box}} \leftarrow 1$
while $B \neq \emptyset$ **do**
  Pop B_k from B
  Determine where B_k lies in relation to Γ and $\tilde{\Gamma}$
  `resolved` $\leftarrow$ `true`
  ▷ *Function resolution criterion*
  Compute Chebyshev coefficients $\hat{f}_{ij}^k$ of f on B_k (with f taken to be extended by zero outside Ω)
  if $B_k \subset \Omega$ **and** $\hat{f}_{ij}^k$ is **not** resolved to ϵ **then**
    `resolved` $\leftarrow$ `false`
  ▷ *Truncation separation criterion*
  if $B_k \cap \tilde{\Omega} \neq \emptyset$ **and** $B_k \cap \mathbb{R}^2 \backslash \Omega \neq \emptyset$ **then**
    `resolved` $\leftarrow$ `false`
  else if $B_k \cap \mathcal{S} \neq \emptyset$ **then**
    Find nodes $\boldsymbol{x} \in \Gamma$ and $\tilde{\boldsymbol{x}} \in \tilde{\Gamma}$ nearest to the center of B_k
    Compute local strip width $s \leftarrow \|\boldsymbol{x} - \tilde{\boldsymbol{x}}\|_2$
    if diagonal length of $B_k > s/2$ **then** `resolved` $\leftarrow$ `false`
  if `resolved` **then**
    Store B_k as a leaf box with Chebyshev coefficients $\hat{f}_{ij}^k$
  else
    Store B_k as a parent box
    Refine B_k and push children onto B
    $n_{\text{box}} \leftarrow n_{\text{box}} + 4$
return $\{B_k\}_{k=0}^{n_{\text{box}}-1}$

The previous subsection described our method to truncate the volume potential to give (4.4), produced by limiting the support of the source to Ω_B, a union of boxes. Then the difference from the classical potential (4.5),

$$\tilde{v} := v_{\text{bulk}} - v,$$

is simply the potential due to the difference $\tilde{f}$ in the source terms, namely the free space convolution

$$\tilde{v} = -\Phi * \tilde{f}, \qquad \text{where} \quad \tilde{f} := \chi_{\Omega \backslash \Omega_B} f, \tag{4.6}$$

where χ_S denotes the characteristic function of a set S, and we note that $\operatorname{supp} \tilde{f} \subset \mathbb{R}^2 \backslash \Omega_B$ and $\|\tilde{f}\|_{L^1(\mathbb{R}^2)} < \infty$. Additional cut cells included as per Remark 4.2 change $\tilde{f}$ but do not change the fact that the *support lies outside of* Ω_B. The proof that $\tilde{v}$ is smooth on $\tilde{\Gamma}$ will rely entirely on this fact that $\tilde{\Gamma}$ is well-separated from $\operatorname{supp} \tilde{f}$; the roughness of $\tilde{f}$ is not relevant.

Our analysis, being based on spatial well-separation, is of a different flavor from that of prior work such as [4, Sec. 4], which showed that for f discontinuous in a box

one expects nearly two orders (with respect to the box size) better convergence in v than in the Chebyshev representation of f.

To state the result we need some definitions. Given $\rho > 1$, recall (e.g., [55]) the open Bernstein ellipse for the standard interval $[-1, 1]$,

$$E_\rho := \left\{ (z + z^{-1})/2 \,:\, z \in \mathbb{C},\ \rho^{-1} < |z| < \rho \right\}. \tag{4.7}$$

We now identify $\mathbb{R}^2$ with $\mathbb{C}$. We use $k = 1, \dots, n_{\text{panel}}$ to index panels. The kth panel is described by a map or chart $X_p : \mathbb{C} \to \mathbb{C}$ such that $X_k([-1, 1]) = \tilde{\gamma}_k$, and X_k is one-to-one and analytic in some open neighborhood of $[-1, 1]$. For any $\rho > 1$ such that E_ρ lies in this neighborhood, we define the *Bernstein mapped ellipse* for the panel $\tilde{\gamma}_k$ by

$$E_{\rho,\tilde{\gamma}_k} := X_k(E_\rho). \tag{4.8}$$

See Figure 4.4(a). For any function $q : \tilde{\gamma}_k \to \mathbb{R}$ we define its pullback Q such that $Q(t) = q(X(t))$ for $-1 \le t \le 1$. Then define the degree-n Chebyshev approximation Q_n of any function Q on $[-1, 1]$ as the usual truncation of its Chebyshev expansion to degree n [55, Ch. 4]. Finally, it is a useful shorthand to refer to the Chebyshev approximation of a function on $\tilde{\gamma}_k$ as the pushforward of the Chebyshev approximation on $[-1, 1]$ of its pullback, where the pushforward of a function Q simply means $Q \circ X^{-1}$.

The main result shows that, if the source truncation is entirely outside the Bernstein mapped ellipse for a panel, then $\tilde{v}$ and its first derivatives are analytic on that panel, as indicated by a specific geometric convergence rate of their Chebyshev approximations. The latter immediately implies geometric convergence of interpolants or quadrature on the panel.

Theorem 1. Let $\tilde{v} = v_{\text{bulk}} - v$ be the change in particular solution (4.4) from the classical volume potential (4.5). Fix a fictitious panel $k \in \{1, \dots, n_{\text{panel}}\}$. Let $\tilde{v}_n$ be the Chebyshev projection of $\tilde{v}$ on this panel $\tilde{\gamma}_k$, and similarly for $\nabla\tilde{v}_n$. Let $\rho > 1$ be such that the panel map X_k is analytic and one-to-one in $\overline{E_\rho}$, and $\overline{E_{\rho,\tilde{\gamma}_k}} \subset \Omega_B$. Then

$$\|\tilde{v}_n - \tilde{v}\|_{\infty,\tilde{\gamma}_k} = \mathcal{O}(\rho^{-n}) \quad \text{and} \quad \|\nabla\tilde{v}_n - \nabla\tilde{v}\|_{\infty,\tilde{\gamma}_k} = \mathcal{O}(\rho^{-n}), \qquad n \to \infty. \tag{4.9}$$

In particular, $\tilde{v}$ and $\nabla\tilde{v}$ are real analytic on the fictitious panel.

Combining this theorem with the result that v is itself smooth on $\tilde{\Gamma}$ (by elliptic regularity in the interior of Ω), then v_{bulk} is also smooth on each fictitious panel. This is our main result for the section. It provides theoretical support for the high-order convergence observed using the induced panelization of $\tilde{\Gamma}$. Note that high-order quadtree approximation is also guaranteed, since v_{bulk}, being harmonic in $\tilde{\Omega}$, must be at least as smooth in $\tilde{\Omega}$ as on $\tilde{\Gamma}$.

Remark 4.3. One can lower-bound ρ: the construction of Ω_B in the previous subsection showed that Ω_B includes the inner half of the strip. If strip panels are chosen with a typical 2:1 aspect ratio, then the Bernstein ellipse preimage for $[-1, 1]$ includes $i/2$, so that $\rho > (1 + \sqrt{5})/2 \approx 1.618$. For the typically used number of upsampled fictitious panel interpolation nodes $n = 24$, the factor $\rho^{-n} \approx 10^{-5}$, implying that at least 5 correct digits are expected (ignoring unknown prefactors). This is rather pessimistic: our results in fact show around 10 correct digits, we believe due to the use of Remark 4.2. This in effect pushes the source $\tilde{f}$ out to Γ, doubling the exponential convergence rate.

Proof. We identify $\mathbb{R}^2$ with $\mathbb{C}$, and thus write $\tilde{v} = \operatorname{Re} V$ for the complex logarithmic potential

$$V(\boldsymbol{x}) = \frac{-1}{2\pi} \int_{\mathbb{C}\setminus\Omega_B} L_{\boldsymbol{y}}(\boldsymbol{x}) \tilde{f}(\boldsymbol{y})\, d\boldsymbol{y}.$$

Here $L_{\boldsymbol{y}}(\boldsymbol{x}) \coloneqq \log(\boldsymbol{x}-\boldsymbol{y})$, but with its branch cut in the $\boldsymbol{x}$ variable chosen (for each fixed $\boldsymbol{y}$) to connect $\boldsymbol{y}$ to ∞ while avoiding the panel Bernstein mapped ellipse. Since there is some nonzero distance between the compact sets $\operatorname{supp}\tilde{f}$ and $\overline{E_{\rho,\tilde{\gamma}_k}}$, then $L_{\boldsymbol{y}}(\boldsymbol{x})$ is uniformly bounded over $y \in \operatorname{supp}\tilde{f}$ and $x \in \overline{E_{\rho,\tilde{\gamma}_k}}$. Combining this with $\|\tilde{f}\|_{L_1(\mathbb{R}^2)} < \infty$ gives $\|V\|_{\infty,\overline{E_{\rho,\tilde{\gamma}_k}}} \le M$ for some M. Since $L_{\boldsymbol{y}}(\cdot)$ is analytic in $\overline{E_{\rho,\tilde{\gamma}_k}}$, V is also analytic in that set (see, e.g., [50, Thm. 5.4]). Then $Q(t) = V(X(t))$, the pullback of V to the complex t-plane, being the composition of two analytic functions, is analytic in $t \in \overline{E_\rho}$ and bounded by M. Let Q_n be the Chebyshev truncation of Q on $[-1,1]$. Then,

$$\|\tilde{v}_n - \tilde{v}\|_{\infty,\tilde{\gamma}_k} \le \|V_n - V\|_{\infty,\tilde{\gamma}_k} = \|Q_n - Q\|_{\infty,[-1,1]} \le \frac{2M}{\rho-1}\rho^{-n},$$

where the first inequality follows by taking the real part, and the second inequality is a standard approximation theory result [55, Thm. 8.2] for functions bounded and analytic in the Bernstein ellipse E_ρ. This concludes the proof for $\tilde{v}$.

Finally, since V is analytic on the closed set $\overline{E_{\rho,\tilde{\gamma}_k}}$ then V' must also be analytic and bounded on this set. Using $\nabla\tilde{v} = \operatorname{Re}(V', iV')$, we can then apply the above argument replacing V by V', with some other choice for M, to show that the two components of $\nabla\tilde{v}$ obey the same result. ∎

Remark 4.4 (Boundary regularity of classical Newton potential). In the classical potential-theory approach [38,39], if the homogeneous BVP (1.3) is to have smooth data (allowing its high-order accurate solution), then the boundary data of the Newton potential (4.5) generated by $f \in C^\infty(\Omega)$ also needs to be smooth (along Γ, since we know that it can only in general be C^1 in the normal direction). The latter is somewhat of a folk theorem. It certainly requires Γ to be smooth (consider a corner where the jump in f induces a weak singularity in $v|_\Gamma$). We do not know of literature stating the result, but note the following[2]. The Newton potential v in $\mathbb{R}^2$ is equal to the interior solution (extended by zero outside of Ω) to $-\Delta u = f$ with $u = 0$ on Γ, minus the single-layer potential $\mathcal{S}_\Gamma[\partial_{\boldsymbol{n}} u]$. By regularity up to the boundary [19, §6.3.2, Thm. 6], $\partial_{\boldsymbol{n}} u \in C^\infty(\Gamma)$, and since the single-layer boundary integral operator on a smooth surface Γ is one order smoothing [40, Thm. 7.2], then $v \in C^\infty(\Gamma)$. Alternatively, proof sketches directly tackling (4.5) exist [52] (also see [33] which needs Ω convex).

4.3. The strip problem. The volume potential formulation above yields a particular solution v_{bulk} valid everywhere inside $\tilde{\Omega}$. We now turn to the problem of constructing a particular solution v_{strip} in the remaining boundary-fitted region $\mathcal{S}$, such that

$$\begin{aligned} \Delta v_{\text{strip}}(\boldsymbol{x}) &= f(\boldsymbol{x}), \qquad && \boldsymbol{x} \in \mathcal{S}, \\ v_{\text{strip}}(\boldsymbol{x}) &= 0, && \boldsymbol{x} \in \Gamma \cup \tilde{\Gamma}. \end{aligned} \tag{4.10}$$

Note that the region $\mathcal{S}$ is thin, as its width was chosen so that the distance between corresponding nodes on Γ and $\tilde{\Gamma}$ is on the order of the local panel size. Therefore, we

[2]The argument is due to Leslie Greengard, personal communication.

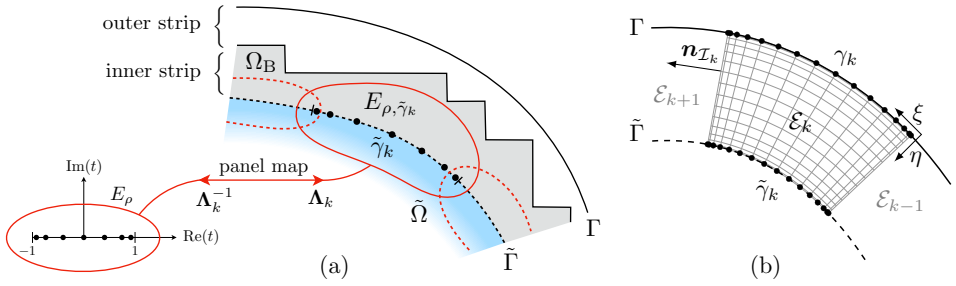

FIG. 4.4. *(a) Zoom near part of the boundary, showing geometry of kth panel $\tilde\gamma_k$ discretizing the fictitious curve $\tilde\Gamma$, and its bijection to the standard panel $[-1,1]$. The box code accurately computes the volume potential due to f in Ω_B, whose boundary is well separated from $\tilde\Gamma$. $E_{\rho,\tilde\gamma_k}$ is the Bernstein mapped ellipse for the kth panel. (b) The spectral collocation grid in the strip region. Chebyshev nodes on panels γ_k and $\tilde\gamma_k$ are connected to form a curvilinear tensor-product grid on which differential operators are numerically constructed. Continuity and continuity of the normal derivative is enforced between elements $\mathcal{E}_k$ and its neighbors $\mathcal{E}_{k-1}$ and $\mathcal{E}_{k+1}$.*

choose to solve in this region using a multidomain spectral collocation method, with each panel creating an element that spans the entirety of the thickness of $\mathcal{S}$ between Γ and $\tilde\Gamma$. Specifically, let $\mathcal{E}_k$ be the region bounded by Γ, $\tilde\Gamma$, and the straight lines connecting the left and right endpoints of panel γ_k with the left and right endpoints of panel $\tilde\gamma_k$, respectively. See Figure 4.4(b) for reference. Denote by $\mathcal{I}_k$ the interface between elements $\mathcal{E}_k$ and $\mathcal{E}_{k+1}$, and $\boldsymbol{n}_{\mathcal{I}_k}$ the unit normal vector to $\mathcal{I}_k$ pointing from $\mathcal{E}_k$ to $\mathcal{E}_{k+1}$. (As the strip is periodic in the annular direction, we let $\mathcal{I}_{n_{\rm panel}}$ be the interface between elements $\mathcal{E}_{n_{\rm panel}}$ and $\mathcal{E}_1$, and $\boldsymbol{n}_{\mathcal{I}_{n_{\rm panel}}}$ its corresponding normal vector). The multidomain boundary value problem is then formulated as

$$
\begin{aligned}
\Delta v^k_{\rm strip}(\boldsymbol{x}) &= f(\boldsymbol{x}), && \boldsymbol{x}\in\mathcal{E}_k,\\
v^k_{\rm strip}(\boldsymbol{x}) &= 0, && \boldsymbol{x}\in\partial\mathcal{E}_k\cap\left(\Gamma\cup\tilde\Gamma\right),\\
v^k_{\rm strip}(\boldsymbol{x}) &= v^{k+1}_{\rm strip}(\boldsymbol{x}), && \boldsymbol{x}\in\mathcal{I}_k,\\
\tfrac{v^k_{\rm strip}}{\partial \boldsymbol{n}_{\mathcal{I}_k}}(\boldsymbol{x}) &= \tfrac{v^{k+1}_{\rm strip}}{\partial \boldsymbol{n}_{\mathcal{I}_k}}(\boldsymbol{x}), && \boldsymbol{x}\in\mathcal{I}_k,
\end{aligned}
\tag{4.11}
$$

for $k=1,\dots,n_{\rm panel}$, where $v^k_{\rm strip}$ is the solution on element $\mathcal{E}_k$ (with $v^{n_{\rm panel}+1}_{\rm strip} := v^1_{\rm strip}$).

We use a spectral collocation method to discretize (4.11). Let $I^{\rm cheb}_{\rm leg}$ be the $(p+1)\times(p+1)$ interpolation matrix which maps function values at $p+1$ Gauss–Legendre nodes to function values at $p+1$ second-kind Chebyshev nodes [55]. On panel k, the nodes $\boldsymbol{x}^{\rm cheb}_{:,k} = I^{\rm cheb}_{\rm leg}\boldsymbol{x}_{:,k}$ are then Chebyshev nodes. Chebyshev nodes on $\tilde\Gamma$ may be similarly defined. Letting $r^{\rm cheb}$ be the order-p second-kind Chebyshev nodes on $[-1,1]$, a curvilinear tensor product grid of nodes $\boldsymbol{X}_{ij,k}$ for element $\mathcal{E}_k$ may be constructed as

$$
\boldsymbol{X}_{ij,k} = \left(\frac{1+r^{\rm cheb}_i}{2}\right)\tilde{\boldsymbol{x}}^{\rm cheb}_{j,k} + \left(\frac{1-r^{\rm cheb}_i}{2}\right)\boldsymbol{x}^{\rm cheb}_{j,k}.
$$

Let $\{\boldsymbol{\xi}_{ij}=(\xi_{ij},\eta_{ij})\}_{i,j=1}^{p+1}$ be the set of tensor-product second-kind Chebyshev nodes of order p over the reference square $[-1,1]^2$. Then for each element, the nodes $\boldsymbol{X}_{ij,k}$ numerically define a mapping from the reference square $[-1,1]^2$ to $\mathcal{E}_k$. That is, the

coordinate mapping for each element is a function $\boldsymbol{X}_k(\xi,\eta) = (X_k(\xi,\eta), Y_k(\xi,\eta)) : [-1,1]^2 \to \mathbb{R}^2$ such that $\boldsymbol{X}_k(\xi_{ij},\eta_{ij}) = \boldsymbol{X}_{ij,k}$ for $i,j = 1,\ldots,p+1$. We may numerically approximate the coordinate mapping through interpolation at the nodes via

$$\boldsymbol{X}_k(\xi,\eta) \approx \sum_{i=1}^{p+1}\sum_{j=1}^{p+1} \boldsymbol{X}_{ij,k}\,\ell_j(\xi)\,\ell_i(\eta), \qquad (\xi,\eta)\in[-1,1]^2,$$

where ℓ_j is the jth Lagrange polynomial associated with the second-kind Chebyshev nodes. Partial derivatives of the elemental coordinate maps, $\partial\boldsymbol{X}_k/\partial\xi$ and $\partial\boldsymbol{X}_k/\partial\eta$, may then be computed through numerical spectral differentiation [54], and derivatives of the inverse coordinate mappings may be derived via the chain rule,

$$\begin{aligned} \tfrac{\partial\xi}{\partial x} &= \;\;\tfrac{1}{J_k}\tfrac{\partial Y_k}{\partial\eta}, & \tfrac{\partial\xi}{\partial y} &= -\tfrac{1}{J_k}\tfrac{\partial X_k}{\partial\eta},\\ \tfrac{\partial\eta}{\partial x} &= -\tfrac{1}{J_k}\tfrac{\partial Y_k}{\partial\xi}, & \tfrac{\partial\eta}{\partial y} &= \;\;\tfrac{1}{J_k}\tfrac{\partial X_k}{\partial\xi}, \end{aligned} \tag{4.12}$$

where $J_k = \frac{\partial X_k}{\partial\xi}\frac{\partial Y_k}{\partial\eta} - \frac{\partial X_k}{\partial\eta}\frac{\partial Y_k}{\partial\xi}$ is the Jacobian of the coordinate mapping $\boldsymbol{X}_k$.

The spectral collocation method proceeds by discretizing the differential operator on each element and enforcing the PDE and boundary conditions on its interior and boundary nodes, respectively. For each k, denote by $v_{ij,k} \approx v^k_{\text{strip}}(\boldsymbol{X}_{ij,k})$; that is, $v_{ij,k}$ is simply v^k_{strip} sampled on the grid of element $\mathcal{E}_k$. Then we may approximate each function by

$$v^k_{\text{strip}}(\xi,\eta) \approx \sum_{i=1}^{p+1}\sum_{j=1}^{p+1} v_{ij,k}\,\ell_j(\xi)\,\ell_i(\eta), \qquad (\xi,\eta)\in[-1,1]^2,$$

for $k = 1,\ldots,n_{\text{panel}}$, where we have introduced the slight abuse of notation $v^k_{\text{strip}}(\xi,\eta) \coloneqq v^k_{\text{strip}}(\boldsymbol{X}_k(\xi,\eta))$. Now, let $D \in \mathbb{C}^{(p+1)\times(p+1)}$ be the one-dimensional spectral differentiation matrix associated with the order-p second-kind Chebyshev nodes on the interval $[-1,1]$ [54], and let $I \in \mathbb{C}^{(p+1)\times(p+1)}$ be the identity matrix. Then $D_\xi = D\otimes I$ and $D_\eta = I\otimes D$ are the two-dimensional differentiation matrices in the ξ- and η-directions on the reference square, of size $(p+1)^2\times(p+1)^2$. Let $M[v] \in \mathbb{C}^{(p+1)\times(p+1)}$ denote the diagonal multiplication matrix formed by placing the entries of v_{ij} along the diagonal. Using (4.12), one may show that differentiation matrices in the x- and y-directions are given by

$$\begin{aligned} D_{X_k} &= M\Big[\tfrac{\partial\xi}{\partial x}\Big]\,D_\xi + M\Big[\tfrac{\partial\eta}{\partial x}\Big]\,D_\eta,\\ D_{Y_k} &= M\Big[\tfrac{\partial\xi}{\partial y}\Big]\,D_\xi + M\Big[\tfrac{\partial\eta}{\partial y}\Big]\,D_\eta. \end{aligned}$$

The discrete Laplacian on element $\mathcal{E}_k$ is then given by $\Delta \approx (D_{X_k})^2 + (D_{Y_k})^2$.

As the multidomain formulation only couples elements to their left and right neighbors, the resulting linear system is block tridiagonal, aside from a corner block due to the periodicity of the strip region. Therefore, direct matrix inversion via block banded LU factorization—along with the Woodbury formula to correct for the corner block—takes only $\mathcal{O}(p^3 n_{\text{panel}})$ operations to compute the strip solutions v^k_{strip} for $k = 1,\ldots,n_{\text{panel}}$.

Remark 4.5. It may happen that $f(\boldsymbol{x})$ is unresolved on the strip grid induced by the given panelization of Γ. To handle this case, our solver first checks if $f(\boldsymbol{x})$ is

resolved on each element of the strip to the given tolerance; if f is unresolved on some elements, the solver splits the corresponding panels in Γ and the algorithm restarts. This process is recursive and happens automatically at the start of Algorithm 1.1, before quadtree construction.

4.4. Patching together v_{bulk} and v_{strip}. We now have particular solutions in both regions of Ω: v_{bulk} satisfying $\Delta v_{\text{bulk}}(\boldsymbol{x}) = f(\boldsymbol{x})$ for $\boldsymbol{x} \in \tilde{\Omega}$, and v_{strip} satisfying $\Delta v_{\text{strip}}(\boldsymbol{x}) = f(\boldsymbol{x})$ for $\boldsymbol{x} \in \mathcal{S}$. However, the piecewise function

$$v(\boldsymbol{x}) = \begin{cases} v_{\text{bulk}}(\boldsymbol{x}), & \boldsymbol{x} \in \tilde{\Omega}, \\ v_{\text{strip}}(\boldsymbol{x}), & \boldsymbol{x} \in \mathcal{S}, \end{cases} \tag{4.13}$$

is not a globally smooth particular solution in Ω, since for each $\boldsymbol{y} \in \tilde{\Gamma}$,

$$\lim_{\boldsymbol{x}\to\boldsymbol{y}^+} v(\boldsymbol{x}) \neq \lim_{\boldsymbol{x}\to\boldsymbol{y}^-} v(\boldsymbol{x}) \quad \text{and} \quad \lim_{\boldsymbol{x}\to\boldsymbol{y}^+} \partial_{\boldsymbol{n}_{\boldsymbol{x}}} v(\boldsymbol{x}) \neq \lim_{\boldsymbol{x}\to\boldsymbol{y}^-} \partial_{\boldsymbol{n}_{\boldsymbol{x}}} v(\boldsymbol{x}),$$

where superscripts of $-$ and $+$ denote limits taken from the interior and exterior of the domain, respectively. That is, the values and normal derivatives of v_{bulk} and v_{strip} do not match across the interface $\tilde{\Gamma}$.

Denote by $\mathcal{S}_{\tilde{\Gamma}}[\sigma]$ and $\mathcal{D}_{\tilde{\Gamma}}[\tau]$ the Laplace single and double layer potentials induced by the densities σ and τ, respectively, on the boundary $\tilde{\Gamma}$, given by

$$\mathcal{S}_{\tilde{\Gamma}}[\sigma](\boldsymbol{x}) \coloneqq \int_{\tilde{\Gamma}} \Phi(\boldsymbol{x},\boldsymbol{y})\sigma(\boldsymbol{y})\, d\boldsymbol{y}, \qquad \mathcal{D}_{\tilde{\Gamma}}[\tau](\boldsymbol{x}) \coloneqq \int_{\tilde{\Gamma}} \frac{\partial \Phi(\boldsymbol{x},\boldsymbol{y})}{\partial \boldsymbol{n}_{\boldsymbol{y}}} \tau(\boldsymbol{y})\, d\boldsymbol{y},$$

Such layer potentials are harmonic functions, satisfying $\Delta \mathcal{S}_{\tilde{\Gamma}}[\sigma] = 0$ and $\Delta \mathcal{D}_{\tilde{\Gamma}}[\sigma] = 0$ in all of $\mathbb{R}^2 \setminus \tilde{\Gamma}$. It can be shown that $\mathcal{S}_{\tilde{\Gamma}}$ and $\mathcal{D}_{\tilde{\Gamma}}$ satisfy the jump relations [35, Ch. 6]

$$\lim_{\boldsymbol{x}\to\boldsymbol{y}^+} \mathcal{S}_{\tilde{\Gamma}}[\sigma](\boldsymbol{x}) - \lim_{\boldsymbol{x}\to\boldsymbol{y}^-} \mathcal{S}_{\tilde{\Gamma}}[\sigma](\boldsymbol{x}) = 0,$$
$$\lim_{\boldsymbol{x}\to\boldsymbol{y}^+} \partial_{\boldsymbol{n}_{\boldsymbol{x}}} \mathcal{S}_{\tilde{\Gamma}}[\sigma](\boldsymbol{x}) - \lim_{\boldsymbol{x}\to\boldsymbol{y}^-} \partial_{\boldsymbol{n}_{\boldsymbol{x}}} \mathcal{S}_{\tilde{\Gamma}}[\sigma](\boldsymbol{x}) = -\sigma,$$

and

$$\lim_{\boldsymbol{x}\to\boldsymbol{y}^+} \mathcal{D}_{\tilde{\Gamma}}[\tau](\boldsymbol{x}) - \lim_{\boldsymbol{x}\to\boldsymbol{y}^-} \mathcal{D}_{\tilde{\Gamma}}[\tau](\boldsymbol{x}) = \tau,$$
$$\lim_{\boldsymbol{x}\to\boldsymbol{y}^+} \partial_{\boldsymbol{n}_{\boldsymbol{x}}} \mathcal{D}_{\tilde{\Gamma}}[\tau](\boldsymbol{x}) - \lim_{\boldsymbol{x}\to\boldsymbol{y}^-} \partial_{\boldsymbol{n}_{\boldsymbol{x}}} \mathcal{D}_{\tilde{\Gamma}}[\tau](\boldsymbol{x}) = 0,$$

for each $\boldsymbol{y} \in \tilde{\Gamma}$. The single layer potential is continuous across its boundary, with a jump in normal derivative equal to the negative of the given density. Similarly, the double layer potential has continuous normal derivative across its boundary, with a jump in value equal to the given density. Thus, we set

$$\tau = v_{\text{bulk}}|_{\tilde{\Gamma}} - v_{\text{strip}}|_{\tilde{\Gamma}},$$
$$\sigma = \partial_{\boldsymbol{n}} v_{\text{bulk}}|_{\tilde{\Gamma}} - \partial_{\boldsymbol{n}} v_{\text{strip}}|_{\tilde{\Gamma}},$$

and define the function

$$v_{\text{glue}}(\boldsymbol{x}) = \mathcal{S}_{\tilde{\Gamma}}[\sigma](\boldsymbol{x}) - \mathcal{D}_{\tilde{\Gamma}}[\tau](\boldsymbol{x}),$$

for all $\boldsymbol{x} \in \Omega$. Adding this function to the piecewise-defined particular solution above results in a globally smooth particular solution to (1.1a), given by

$$v(\boldsymbol{x}) = \begin{cases} v_{\text{bulk}}(\boldsymbol{x}) + v_{\text{glue}}(\boldsymbol{x}), & \boldsymbol{x} \in \tilde{\Omega}, \\ v_{\text{strip}}(\boldsymbol{x}) + v_{\text{glue}}(\boldsymbol{x}), & \boldsymbol{x} \in \mathcal{S}, \end{cases} \tag{4.14}$$

with values on $\tilde{\Gamma}$ defined by their limit from either side (which are equal, to discretization accuracy). As the layer potentials in v_{glue} may be rapidly evaluated using the FMM, the overall particular solution v may be rapidly evaluated at any point $\boldsymbol{x} \in \Omega$.

5. Numerical results. We now demonstrate our adaptive Poisson solver on some challenging geometries, requiring adaptivity both along the boundary to resolve geometric features and in the bulk to resolve spatial inhomogeneities. All numerical examples were run in MATLAB R2024b on single core of an M4 Max MacBook Pro with 128 GB of memory. Our code is open source and freely available [21].

5.1. A rounded raindrop. We first demonstrate our adaptive Poisson solver on a raindrop-shaped geometry shown in Figure 5.1, consisting of 56 panels of order 16. The corner of the raindrop is rounded to a length scale of 10^{-3}. While the pinched end of the raindrop-like shape would force a uniform-grid method to over-refine the largest length scales, our variable-width strip region smoothly captures the transition in panel size across three orders of magnitude.

We run the solver with a requested tolerance of 10^{-10}. Our test solution consists of a sum of Gaussians exponentially clustering into the cusp with decreasing variance, along with the smooth background function $2x\cos 3\pi y$ added for variation along the boundary. We analytically compute the inhomogeneity f corresponding to this solution. The precomputation phase for this domain, which includes constructing the strip region and building a 16th-order quadtree to satisfy the refinement criterion, takes 0.3 s. The resulting quadtree possesses 2,965 leaf nodes with 594,944 degrees of freedom, and computation of v_{bulk} using a 16th-order box code [5] takes 0.06 s. The strip region contains 56 elements, with each element upsampled slightly to a resolution of 16×24 to yield a strip mesh with 21,504 degrees of freedom. Computation of the strip particular solution, v_{strip}, takes 0.3 s. The homogeneous solution w is computed in 0.2 s by solving a boundary integral equation using GMRES, with matrix-vector products accelerated by the FMM. Evaluating the solution back on all quadtree and strip nodes using the FMM takes 2.7 s.

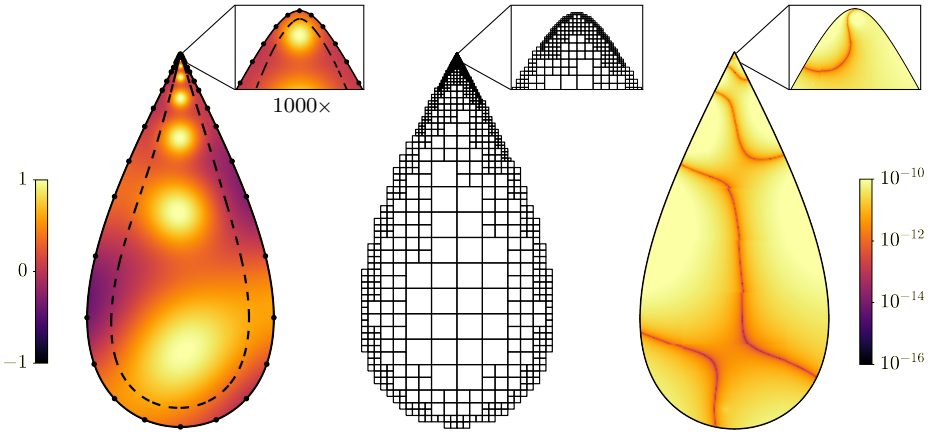

FIG. 5.1. *(Left) The raindrop shape is adaptively panelized, with small panels clustering in the rounded cusp. Black circles correspond to panel endpoints. The strip region (plotted as a dashed line) conforms to this adaptive panelization. The test solution is also shown and consists of a series of Gaussians with decreasing variances clustering inside the cusp. (Center) The inhomogeneity induced by the given solution is adaptively resolved on a 16th-order background quadtree with* 594,944 *unknowns and truncated according to the strip refinement criterion outlined in subsection* 4.2.1. *(Right) The maximum absolute error is around* 10^{-10} *over the whole domain.*

5.2. Comparison to FFT-based schemes. We now compare our adaptive scheme against an FFT-based uniform bulk solver. To do this, we define a series of problems with decreasing length scale η, driven by both geometry and righthand side. We generalize the raindrop shape from subsection 5.1 by rounding the corner to a length scale of η. Such a curve is parametrized in $\mathbb{C}$ by

$$z(t) = -\tfrac{1}{4}\sin t - \mathrm{i}\sqrt{\sin^2 \tfrac{t}{2} + \eta^2}, \qquad t \in [0, 2\pi].$$

Its minimum radius of curvature is $R_{\min} = \eta/4 + \mathcal{O}(\eta^2)$, at $t = 0$. As in subsection 5.1, we set the righthand side to a series of Gaussians clustering into the corner with decreasing variance, with the numerical support of the narrowest Gaussian proportional to the length scale η.

In Figure 5.2 we vary η from 10^0 to 10^{-8} and plot the total runtime and memory consumption of our adaptive solver applied to each problem with a requested tolerance of 10^{-10}. For comparison, we also run the 2D FFT (as implemented in MATLAB's `fft2`) on a series of successively refined $n \times n$ uniform grids and plot the same, with n a power of two to allow the fastest FFTs. The length scale parameter η that the $n \times n$ grid for any variety of FFT-based solver would be able to resolve cannot shrink faster than $\mathcal{O}(1/n)$. We choose a specific relation $\eta = 10h$, where $h = 1/n$ is the grid spacing; in other words $R_{\min} = 2.5\,h$. The figure shows the stark contrast between the runtime or memory use of our adaptive solver versus that of any nonadaptive FFT-based solver: runtime and memory grow very weakly with $1/\eta$ (one expects logarithmically) for the adaptive case, while they are both $\mathcal{O}(1/\eta^2)$ for the FFT. The upshot is that on a single shared memory node one cannot reach $\eta < 10^{-4}$, whereas the adaptive solver easily reaches $\eta = 10^{-8}$ in a few seconds and a few tens of MB of memory. Note that this $R_{\min}$ is still about eight times smaller than needed in the FFT spectral solver of the second author [46], whose Figure 7 shows that $R_{\min}/h \approx 20$ is needed for 10-digit accuracy. This prefactor would move the red FFT lines to the left, only strengthening our point.

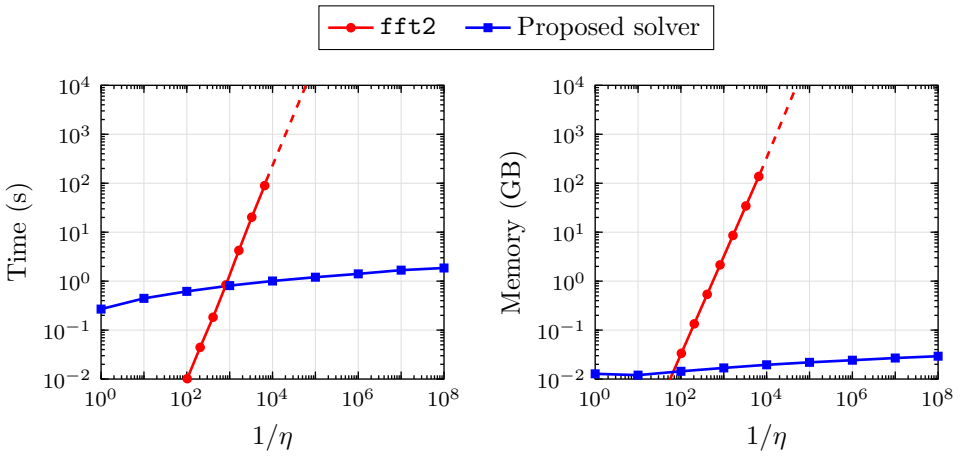

FIG. 5.2. *Performance comparison between the 2D FFT (red circles) and our adaptive solver (blue squares). We benchmark our adaptive solver on a series of teardrop problems with decreasing length scale η and a requested error tolerance of 10^{-10}, and record runtime and memory consumption of the entire solver. For comparison, we benchmark the 2D FFT on successively refined uniform grids using MATLAB's* `fft2`*, using ten gridpoints to resolve η; see subsection 5.2. Dashed lines indicate predicted values for the FFT. Note that our adaptive solver is solving the full Poisson problem, while the 2D FFT would only solve the bulk problem.*

5.3. A close-to-touching multiscale geometry and inhomogeneity. We now turn to a more extreme multiscale geometry, with length scales continuously spanning four orders of magnitude and close-to-touching regions throughout all scales. This is constructed in $\mathbb{C}$ by spectrally-accurate blending of simple sine wave functions, followed by overall exponentiation [21]. Moreover, we prescribe the inhomogeneity f to be the polar angle $\theta = \tan^{-1}(y/x)$ with the branch cut lying between the "teeth" of the geometry. This choice of f would pose a problem for methods based on function extension, as the values of f coming from the top and bottom sides conflict. The geometry and inhomogeneity are depicted in Figure 5.3, alongside an example 16th-order quadtree and the corresponding pointwise error in the computed solution.

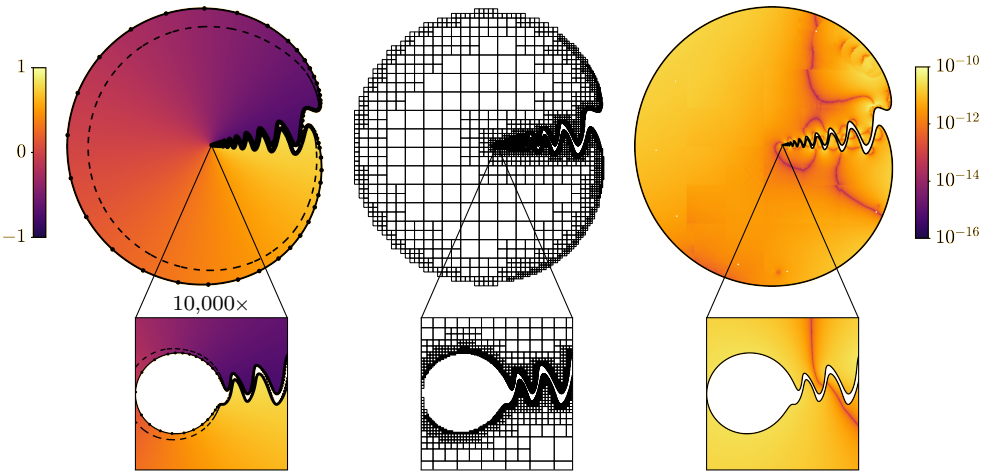

FIG. 5.3. *(Left) A multiscale geometry with length scales spanning four orders of magnitude and close-to-touching "teeth" is adaptively panelized into* 1,970 *panels. The test solution is chosen so that the inhomogeneity is given by the polar angle, with the branch cut taken between the "teeth" of the domain. (Center) The truncated 16th-order quadtree that resolves the inhomogeneity contains over 8 million degrees of freedom. (Right) The computed solution is accurate to about* 10^{-10}.

To further illustrate the adaptive performance of our solver, we solve a series of Poisson problems on this multiscale geometry with requested input tolerances $\epsilon_{\text{ask}} \in \{10^{-3}, 10^{-6}, 10^{-9}\}$, and measure the maximum pointwise error ϵ_{get} in the resulting solution. Table 5.1 shows the results, along with other metrics: the polynomial degree $p \sim \log(1/\epsilon_{\text{ask}})$ used to discretize the boundary, quadtree, and strip region; the total number of degrees of freedom N used to represent the solution; the time taken to set up (T_{setup}), compute a particular solution (T_{part}), compute a homogeneous correction (T_{homo}), and evaluate the solution back on the set of quadtree and strip nodes (T_{eval}); the total time (T_{total}); and the overall speed of the solver in points per second.

5.4. A geometry inspired by cell blebbing. We now apply our solver to a biologically-inspired geometry. Using image processing, we extract the boundary of a cell membrane undergoing blebbing from an image taken from [10, Fig. 6]. We then fit a Fourier series to the sampled boundary and smooth it by convolving with a fixed-width Gaussian. The resulting geometry is shown in Figure 5.4 and possesses many small folds and thin filaments that would requirement very fine panels everywhere using a method based on uniform grids. We prescribe the solution inside this geometry to be 200 randomly placed Gaussians with variances ranging between 1 and 10^{-5}, with

TABLE 5.1

Performance results for the adaptive Poisson solver for different requested input tolerances ϵ_{ask}, applied to the multiscale geometry depicted in Figure 5.3. We report the maximum pointwise error achieved (ϵ_{get}); the polynomial degree used for the boundary, quadtree, and strip discretizations (p); the total number of degrees of freedom used to represent the solution (N); the time taken to set up the solver (T_{part}); the time taken to compute a particular solution (T_{solve}); the time taken to compute the homogeneous correction (T_{homo}); the time taken to evaluate the solution back on the set of quadtree and strip nodes (T_{eval}); the total time (T_{total}); and the speed in points per second (pps). All times are measured in seconds. Note that we choose p so that $p \sim \log(1/\epsilon_{ask})$.

ϵ_{ask}	ϵ_{get}	p	N	T_{setup}	T_{part}	T_{homo}	T_{eval}	T_{total}	Speed (pps)
10^{-3}	6.6×10^{-4}	3	1,327,108	0.99	0.97	1.91	8.68	12.55	105,756
10^{-6}	8.2×10^{-8}	6	4,109,162	1.46	1.46	2.78	21.53	27.23	150,931
10^{-9}	1.8×10^{-10}	9	8,418,450	2.04	2.19	3.70	41.99	49.92	168,633

the inhomogeneity defined accordingly.

Due to the random placement of the Gaussians, some Gaussians occur very close to the boundary with length scales much smaller than the boundary panelization. Thus, this setup tests the case mentioned in Theorem 4.5. The solver automatically splits the boundary panels where f was unresolved on the strip grid, and restarts; this process happens a three times, at which point the refined panelization induces a strip which resolved f everywhere. The resulting 16th-order quadtree to resolve f contains 9.3 million degrees of freedom. The maximum absolute error in the computed solution is 10^{-10}.

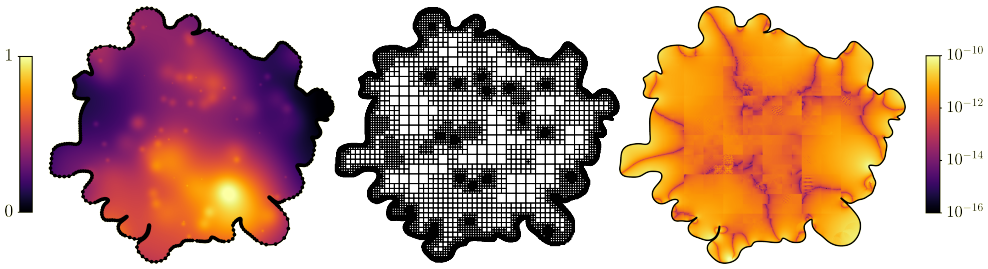

FIG. 5.4. *(Left) Computed solution to Poisson's equation on the bleb geometry, with an inhomogeneity consisting of 200 randomly placed Gaussians with variances ranging between 1 and 10^{-5}. (Center) The inhomogeneity is adaptively resolved on a 16th-order quadtree with 9.3 million unknowns. (Right) The maximum absolute error in the computed solution is 10^{-10}.*

6. Conclusion. We presented a high-order Poisson solver that is fully adaptive with respect to both boundary geometry and forcing function. It combines convolution with the free-space Green's function on an adaptive quadtree that resolves the forcing function ("box code") in the bulk, with a curvilinear spectral solver in a boundary "strip" region. Layer potentials on the fictitious strip interface repair the Cauchy matching conditions to give a particular solution for the whole domain. The quadtree is truncated within the strip region—we prove that this maintains smoothness in the bulk—preventing the need for over-refinement to resolve the boundary. For this, adapting the strip width function $h(t)$ smoothly to the local boundary panel size is crucial. We show how our solver efficiently handles various multiscale geometries, with features spanning up to 8 orders of magnitude, and compare against FFT-based uniform solvers (which are impractical for 4 or more orders of magnitude).

We expect the adaptive solver presented here to naturally extend to three-dimensional problems, as the strip is based on "thickening" an existing boundary discretization. Defining a smooth fictitious "shell" in 3D from an unstructured high-order surface triangulation will require partition-of-unity smoothing based on local coordinate charts, as an arc-length parametrization is unavailable in three dimensions. We hope to develop such a solver in future work.

Many more extensions of the adaptive Poisson solver presented here are worthy of exploration. Other inhomogeneous scalar-valued PDEs with known Green's function, such as the Helmholtz or screened Poisson equations, may be solved using essentially the same piecewise representation of the particular solution [46]. Vector-valued PDEs such as the Stokes equations or elastostatics require further development of a vector-valued strip solver. On multiply-connected domains, additional single- and double-layer corrections are needed for each interior boundary. For time-dependent problems with moving geometries or evolving inhomogeneities (e.g., as may arise in a fluid simulation), the truncated quadtree from a previous time step could be updated for the next time step by refining or coarsening only those leaf boxes which overlap the strip, reducing the setup time needed by the solver.

Acknowledgments. We have benefited from many useful conversations with Manas Rachh, Charles Epstein, Dhairya Malhotra, Hai Zhu, Leslie Greengard, and Shidong Jiang. We thank Travis Askham for adding support for arbitrary order discretizations to the `boxcode2d` library [5]. The Flatiron Institute is a division of the Simons Foundation.

REFERENCES

[1] L. af Klinteberg, T. Askham, and M. C. Kropinski, *A fast integral equation method for the two-dimensional Navier-Stokes equations*, J. Comput. Phys., 409 (2020), p. 109353, https://doi.org/10.1016/j.jcp.2020.109353.

[2] T. Anderson, H. Zhu, and S. Veerapaneni, *A fast, high-order scheme for evaluating volume potentials on complex 2D geometries via area-to-line integral conversion and domain mappings*, J. Comput. Phys., 472 (2023), p. 111688, https://doi.org/10.1016/j.jcp.2022.111688.

[3] T. G. Anderson, M. Bonnet, L. M. Faria, and C. Pérez-Arancibia, *Fast, high-order numerical evaluation of volume potentials via polynomial density interpolation*, J. Comput. Phys., 511 (2024), p. 113091, https://doi.org/10.1016/j.jcp.2024.113091.

[4] T. Askham and A. J. Cerfon, *An adaptive fast multipole accelerated Poisson solver for complex geometries*, J. Comput. Phys., 344 (2017), pp. 1–22, https://doi.org/10.1016/j.jcp.2017.04.063.

[5] T. Askham, F. Ethridge, D. Fortunato, Z. Gimbutas, L. Greengard, M. O'Neil, and V. Rokhlin, 2025, https://github.com/flatironinstitute/boxcode2d.

[6] K. Böhmer, *Numerical Methods for Nonlinear Elliptic Differential Equations: A Synopsis*, Oxford University Press, 10 2010, https://doi.org/10.1093/acprof:oso/9780199577040.001.0001.

[7] W. L. Briggs, V. E. Henson, and S. F. McCormick, *A Multigrid Tutorial, Second Edition*, SIAM, Philadelphia, PA, 2000, https://doi.org/10.1137/1.9780898719505.

[8] O. P. Bruno and J. Paul, *Two-dimensional Fourier continuation and applications*, SIAM J. Sci. Comput., 44 (2022), pp. A964–A992, https://doi.org/10.1137/20M1373189.

[9] B. L. Buzbee, G. H. Golub, and C. W. Nielson, *On direct methods for solving Poisson's equations*, SIAM J. Numer. Anal., 7 (1970), pp. 627–656, https://doi.org/10.1137/0707049.

[10] G. T. Charras, *A short history of blebbing*, J. Microscopy, 231 (2008), pp. 466–478, https://doi.org/10.1111/j.1365-2818.2008.02059.x.

[11] A. J. Chorin, *The numerical solution of the Navier-Stokes equations for an incompressible fluid*, Bull. Amer. Math. Soc., 73 (1967), pp. 928–931.

[12] T. A. Davis, S. Rajamanickam, and W. M. Sid-Lakhdar, *A survey of direct methods for sparse linear systems*, Acta Numer., 25 (2016), pp. 383–566, https://doi.org/10.1017/

S0962492916000076.
[13] M. de Berg, O. Cheong, M. van Kreveld, and M. Overmars, *Computational Geometry: Algorithms and Applications*, Springer Berlin Heidelberg, Berlin, Heidelberg, 2008, https://doi.org/10.1007/978-3-540-77974-2.
[14] L. Demanet and A. Townsend, *Stable extrapolation of analytic functions*, Found. Comput. Math., 19 (2019), pp. 297–331.
[15] T. A. Driscoll, N. Hale, and L. N. Trefethen, *Chebfun Guide*, Pafnuty Publications, 2014, http://www.chebfun.org/docs/guide/.
[16] C. Epstein and S. Jiang, *A stable, efficient scheme for $\mathcal{C}^n$ function extensions on smooth domains in $\mathbb{R}^d$*, June 2022, https://arxiv.org/abs/2206.11318.
[17] F. Ethridge, *Fast Algorithms for Volume Integrals in Potential Theory*, PhD thesis, New York University, 2000.
[18] F. Ethridge and L. Greengard, *A new fast-multipole accelerated Poisson solver in two dimensions*, SIAM J. Sci. Comput., 23 (2001), pp. 741–760, https://doi.org/10.1137/S1064827500369967.
[19] L. C. Evans, *Partial Differential Equations*, vol. 19 of Graduate Studies in Mathematics, American Mathematical Society, Providence, RI, 1998.
[20] D. Fortunato, 2022, https://github.com/danfortunato/treefun.
[21] D. Fortunato, 2022, https://github.com/danfortunato/fully-adaptive-poisson.
[22] D. Fortunato and A. Townsend, *Fast Poisson solvers for spectral methods*, IMA J. Numer. Anal., 40 (2020), pp. 1994–2018, https://doi.org/10.1093/imanum/drz034.
[23] F. Fryklund and L. Greengard, *An FMM accelerated Poisson solver for complicated geometries in the plane using function extension*, SIAM J. Sci. Comput., 45 (2023), pp. A3001–A3019, https://doi.org/10.1137/22M153495X.
[24] F. Fryklund, L. Greengard, S. Jiang, and S. Potter, *A lightweight, geometrically flexible fast algorithm for the evaluation of layer and volume potentials*, 2024, https://arxiv.org/abs/2409.11998, https://arxiv.org/abs/2409.11998.
[25] F. Fryklund, M. C. A. Kropinski, and A.-K. Tornberg, *An integral equation–based numerical method for the forced heat equation on complex domains*, Adv. Comput. Math., 46 (2020), p. 69, https://doi.org/10.1007/s10444-020-09804-z.
[26] F. Fryklund, E. Lehto, and A.-K. Tornberg, *Partition of unity extension of functions on complex domains*, J. Comput. Phys., 375 (2018), pp. 57–79, https://doi.org/10.1016/j.jcp.2018.08.012.
[27] Z. Gimbutas and L. Greengard, https://github.com/zgimbutas/fmmlib2d.
[28] L. Greengard and J.-Y. Lee, *A direct adaptive Poisson solver of arbitrary order accuracy*, J. Comput. Phys., 125 (1996), pp. 415–424, https://doi.org/10.1006/jcph.1996.0103.
[29] L. Greengard and V. Rokhlin, *A fast algorithm for particle simulations*, J. Comput. Phys., 73 (1987), pp. 325–348, https://doi.org/10.1016/0021-9991(87)90140-9.
[30] J. Guermond, P. Minev, and J. Shen, *An overview of projection methods for incompressible flows*, Comput. Meth. Appl. Mech. Engs., 195 (2006), pp. 6011–6045, https://doi.org/10.1016/j.cma.2005.10.010.
[31] S. Hao, A. H. Barnett, P. G. Martinsson, and P. Young, *High-order accurate methods for Nyström discretization of integral equations on smooth curves in the plane*, Adv. Comput. Math., 40 (2014), pp. 245–272, https://doi.org/10.1007/s10444-013-9306-3.
[32] J. Helsing and R. Ojala, *On the evaluation of layer potentials close to their sources*, J. Comput. Phys., 227 (2008), pp. 2899–2921, https://doi.org/10.1016/j.jcp.2007.11.024.
[33] J. Hu, Y. Huang, and J. Lu, *Boundary regularity of Riesz potential, smooth solution to the chord log-Minkowski problem*, 2024, https://arxiv.org/abs/2304.14220, https://arxiv.org/abs/2304.14220.
[34] B. N. Khoromskij and J. M. Melenk, *Boundary concentrated finite element methods*, SIAM J. Numer. Anal., 41 (2003), pp. 1–36, https://doi.org/10.1137/S0036142901391852.
[35] R. Kress, *Linear Integral Equations*, vol. 82 of Applied Mathematical Sciences, Springer, New York, NY, 3rd ed. ed., 2014, https://doi.org/10.1007/978-1-4614-9593-2.
[36] D. Logan, *Applied Mathematics*, Wiley, 4th ed., 2013.
[37] D. Malhotra and G. Biros, *Algorithm 967: A distributed-memory fast multipole method for volume potentials*, ACM Trans. Math. Softw., 43 (2016), pp. 17:1–17:27, https://doi.org/10.1145/2898349.
[38] A. Mayo, *The rapid evaluation of volume integrals of potential theory on general regions*, J. Comput. Phys., 100 (1992), pp. 236–245.
[39] A. McKenney, L. Greengard, and A. Mayo, *A fast Poisson solver for complex geometries*, J. Comput. Phys., 118 (1995), pp. 348–355, https://doi.org/10.1006/jcph.1995.1104.
[40] W. C. H. McLean, *Strongly elliptic systems and boundary integral equations*, Cambridge

University Press, 2000.

[41] E. Rothe, *Zweidimensionale parabolische Randwertaufgaben als Grenzfall eindimensionaler Randwertaufgaben*, Math. Ann., 102 (1930), pp. 650–670, https://doi.org/10.1007/BF01782368.

[42] R. Saye, *Implicit mesh discontinuous Galerkin methods and interfacial gauge methods for high-order accurate interface dynamics, with applications to surface tension dynamics, rigid body fluid–structure interaction, and free surface flow: Part I*, J. Comput. Phys., 344 (2017), pp. 647–682, https://doi.org/10.1016/j.jcp.2017.04.076.

[43] R. Saye, *Implicit mesh discontinuous Galerkin methods and interfacial gauge methods for high-order accurate interface dynamics, with applications to surface tension dynamics, rigid body fluid–structure interaction, and free surface flow: Part II*, J. Comput. Phys., 344 (2017), pp. 683–723, https://doi.org/10.1016/j.jcp.2017.05.003.

[44] Z. Shen and K. Serkh, *Rapid evaluation of Newtonian potentials on planar domains*, SIAM J. Sci. Comput., 46 (2024), pp. A609–A628, https://doi.org/10.1137/22M1526666.

[45] D. Slepian and H. O. Pollak, *Prolate spheroidal wave functions, Fourier analysis and uncertainty, I*, Bell Syst. Tech. J., 40 (1961), pp. 43–64.

[46] D. B. Stein, *Spectrally accurate solutions to inhomogeneous elliptic PDE in smooth geometries using function intension*, J. Comput. Phys., 470 (2022), p. 111594, https://doi.org/10.1016/j.jcp.2022.111594.

[47] D. B. Stein, R. D. Guy, and B. Thomases, *Immersed boundary smooth extension: A high-order method for solving PDE on arbitrary smooth domains using Fourier spectral methods*, J. Comput. Phys., 304 (2016), pp. 252–274, https://doi.org/10.1016/j.jcp.2015.10.023.

[48] D. B. Stein, R. D. Guy, and B. Thomases, *Immersed boundary smooth extension (IBSE): A high-order method for solving incompressible flows in arbitrary smooth domains*, J. Comput. Phys., 335 (2017), pp. 155–178, https://doi.org/10.1016/j.jcp.2017.01.010.

[49] D. B. Stein, R. D. Guy, and B. Thomases, *Convergent solutions of Stokes Oldroyd-B boundary value problems using the immersed boundary smooth extension (IBSE) method*, J. Non-Newtonian Fluid Mech., 268 (2019), pp. 56–65.

[50] E. M. Stein and R. Shakarchi, *Complex Analysis*, Princeton Lectures in Analysis, No. 2, Princeton University Press, 2003.

[51] A. Tagliasacchi, 2017, https://github.com/taiya/kdtree.

[52] T. Tao, G. Metafune, et al., *Regularity of Newtonian potential along smooth boundary*, 2023, https://mathoverflow.net/questions/446383/regularity-of-newtonian-potential-along-smooth-boundary.

[53] A. Townsend, H. Wilber, and G. B. Wright, *Computing with functions in spherical and polar geometries I. The sphere*, SIAM J. Sci. Comput., 38 (2016), pp. C403–C425, https://doi.org/10.1137/15M1045855.

[54] L. N. Trefethen, *Spectral Methods in MATLAB*, SIAM, Philadelphia, PA, 2000, https://doi.org/10.1137/1.9780898719598.

[55] L. N. Trefethen, *Approximation Theory and Approximation Practice*, SIAM, Philadelphia, PA, 2013, https://doi.org/10.1137/1.9781611975949.

[56] H. Wilber, A. Townsend, and G. B. Wright, *Computing with functions in spherical and polar geometries II. The disk*, SIAM J. Sci. Comput., 39 (2017), pp. C238–C262, https://doi.org/10.1137/16M1070207.

[57] B. Wu, H. Zhu, A. Barnett, and S. Veerapaneni, *Solution of Stokes flow in complex nonsmooth 2D geometries via a linear-scaling high-order adaptive integral equation scheme*, J. Comput. Phys., 410 (2020), p. 109361, https://doi.org/10.1016/j.jcp.2020.109361.