SIAM J. SCI. COMPUT.
Vol. 43, No. 1, pp. A446–A471

THE SMOOTH EXTENSION EMBEDDING METHOD*

DANIEL J. AGRESS† AND PATRICK Q. GUIDOTTI†

Abstract. A new approach to the solution of boundary value problems within the so-called *fictitious domain method* philosophy is proposed, which avoids well-known shortcomings of other methods of this type, including the need to generate extensions of the data. The salient feature of the novel method, which we refer to as *SEEM (smooth extension embedding method)*, is that it reduces the whole boundary value problem to a linear constraint for an appropriate optimization problem formulated in a larger, simpler set containing the domain on which the boundary value problem is posed and which allows for the use of straightforward discretizations. It can also be viewed as a fully discrete *meshfree method,* which uses a novel class of basis functions and thus builds a bridge between fictitious domain and meshfree methods. The proposed method, in essence, computes a (discrete) extension of the solution to the boundary value problem by selecting it as a smooth element of the complete affine family of solutions of the original equations now yielding an underdetermined problem for an unknown defined in the whole fictitious domain. The actual regularity of this extension is determined by that of the analytic solution and by the choice of objective functional. Numerical experiments demonstrate that it is stable enough to efficiently solve boundary value problems on general geometries and that it produces solutions of tunable (and high) accuracy.

Key words. fictitious domain methods, meshfree methods, numerical solutions of boundary value problems, optimization, high order discretizations

AMS subject classifications. 68Q25, 68R10, 68U05

DOI. 10.1137/19M1300844

1. Introduction.

In this paper an optimization approach is proposed for the numerical solution of general boundary value problems (BVPs) on complex geometries. The approach is a hybrid of the *meshfree collocation* and *fictitious domain methods* for solving BVPs. As will be demonstrated, this approach not only establishes a direct connection between these two general approaches but also combines their strengths.

The most well established and commonly used numerical methods for solving BVPs are the finite element method and the finite difference method, along with spectral methods. While finite element methods come with the heavy burden of generating a mesh (which becomes a serious limiting factor when dealing with some problems, such as, for instance, moving boundary problems or those in three space dimensions), straightforward finite difference methods and spectral methods are limited by the small number of allowable shapes for the domain Ω.

Two widely used methods which seek to avoid these difficulties are known as *meshfree collocation* and *fictitious domain methods*. In *meshfree collocation methods*, the solution is sought as a linear combination of radial basis functions centered at collocation points scattered throughout the domain Ω and on its boundary Γ. While these methods can achieve very high orders of convergence, in their simplest implementations, the resulting matrices are dense and poorly conditioned; see, for example, [7, Chapter 16]. This leads to difficulties when scaling to denser grids. As described in [7, 20] and as we will explain below, these methods can also be viewed as a refor-

*Submitted to the journal's Methods and Algorithms for Scientific Computing section November 19, 2019; accepted for publication (in revised form) October 27, 2020; published electronically February 1, 2021.
https://doi.org/10.1137/19M1300844
†Department of Mathematics, University of California, Irvine, Irvine, CA 92697 USA (dagress@uci.edu, gpatrick@math.uci.edu).

mulation of BVPs as constrained optimization problems on $\mathbb{R}^d$. In *fictitious domain methods*, the problem is transplanted from the original domain Ω to an encompassing simple region $\mathbb{B}$, where well-studied and understood discretizations and solvers can be utilized. The computed solution is an approximation of u within the domain Ω and an extension of this approximation on $\mathbb{B}\backslash\Omega$. A drawback common to many of these methods is a lack of regularity across the boundary which often leads to a lack of regularity of the global extension and hence to a low order of convergence; see [18] for a discussion of this issue with regard to the immersed boundary method. Additionally, these methods often require a very different treatment of the various types of commonly occurring elliptic operators and boundary conditions; see, for example, the introduction to [15].

The approach proposed here is a hybrid of these two methods. In a way similar to *meshfree methods*, it reduces the entire BVP to playing the role of a linear constraint to an optimization problem for an appropriately chosen objective functional defined on a larger domain. However, borrowing from the *fictitious domain* framework, we carry out this constrained optimization procedure on a regular grid with straightforward and efficient discretizations. As we will explain below, this hybrid procedure will combine the simplicity, speed, and scalability of *fictitious domain methods* with the high order accuracy and wide ranging versatility of *meshfree collocation methods*.

Next a detailed description is given of the proposed method as it is developed from the ground up. Later, we will describe how the method fits into the framework of existing meshfree methods, and we will argue that implementing meshfree methods in a fictitious domain setting provides significant advantages. We will also compare our method with several existing fictitious domain methods and illustrate where the proposed method delivers the most benefits.

1.1. Description of the method. While its ideas and formulation readily apply to a wide variety of PDE problems, the method will be illustrated by means of second order BVPs of type

$$\begin{cases} \mathcal{A}u = f & \text{in } \Omega, \\ \mathcal{B}u = g & \text{on } \Gamma = \partial\Omega \end{cases} \tag{1.1}$$

for an elliptic operator $\mathcal{A}$ such as, e.g., the Laplacian $-\Delta$, and an admissible boundary operator $\mathcal{B}$ such as, e.g., the trace γ_Γ (Dirichlet problem), the unit outer normal derivative ∂_ν (Neumann problem), or a combination thereof (Robin type problem).

In the spirit of fictitious domain methods, the domain $\overline{\Omega}$ is embedded into a simple (square or rectangular) "container" domain $\mathbb{B}$, for which $\overline{\Omega} \subset \mathbb{B}$. In this paper $\mathbb{B}$ is chosen to be the periodic box $[-\pi, \pi)^d \subset \mathbb{R}^d$. Denote by $\mathbb{B}^m$ a regular uniform discretization of $\mathbb{B}$ consisting of N_m points, where m is the number of discretization points along each dimension. See Figure 1 below. Replace the continuous differential operator by a discrete counterpart $A = A^m$, defined as a discrete evaluation of $\mathcal{A}$ at grid-points which lie inside Ω,

$$x \in \Omega^m = \Omega \cap \mathbb{B}^m = \{x_k \,|\, k = 1, \ldots, N_m^\Omega\}, \quad N_m^\Omega = |\Omega^m| \in \mathbb{N},$$

where A^m acts on "discrete functions" defined on $\mathbb{B}^m$. Given a set of points

$$\Gamma^m = \{y_j \,|\, j = 1, \ldots, N_m^\Gamma\} \subset \Gamma,$$

it is possible to discretize the boundary condition using any kind of interpolation and any kind of discrete differentiation (where needed) based on the grid $\mathbb{B}^m$ and

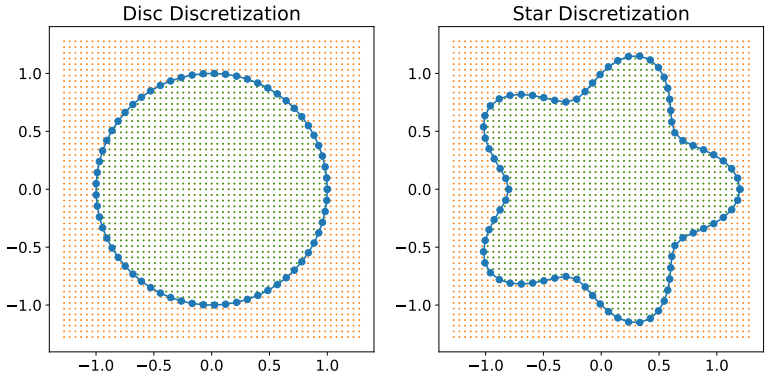

FIG. 1. *Discretizing the domain and its boundary.*

to obtain a corresponding discrete equation $Bu = B^m u^m = g^m$ for the unknown vector $u^m : \mathbb{B}^m \to \mathbb{R}$ and a discretization $g^m : \Gamma^m \to \mathbb{R}$ of the boundary function g, defined on Γ^m. In this way, the continuous BVP (1.1) can be replaced by the discrete underdetermined system given by

$$Cu = C_m u^m = \begin{bmatrix} A \\ B \end{bmatrix} u = \begin{bmatrix} A^m \\ B^m \end{bmatrix} u^m = \begin{bmatrix} f^m \\ g^m \end{bmatrix} = b^m = b, \tag{1.2}$$

where f^m is a discretization of f at grid-points in $\mathbb{B}^m \cap \Omega$. Observe that we often suppress the superscripts and subscripts in order to simplify the notation. Notice that

$$u^m \in \mathbb{R}^{N_m}, \quad f^m \in \mathbb{R}^{N_m^\Omega}, \quad \text{and } g^m \in \mathbb{R}^{N_m^\Gamma}.$$

In numerical experiments, the dimensions are always chosen in such a way that $N_m^\Omega + N_m^\Gamma < N_m$ is satisfied. While not strictly necessary, care is also taken to make sure that all equations in the system are independent of one another. The reason for this is numerical conditioning of the relevant matrices (more will be said on this later). We emphasize that the operators A^m and B^m can be constructed using any form of interpolation and discrete differentiation on the regular grid. For example, both finite differences and spectral differentiation could be used.

To deal with the fact that the system is underdetermined, a common fictitious domain approach (see, e.g., [13]) is to extend the original PDE to the entire larger domain $\mathbb{B}$. Unfortunately, beyond the obvious difficulty of finding a smooth extension for the given data, such an extension of the problem will usually introduce a singularity along the boundary Γ and prevent the resulting solution from attaining a high order of convergence. In contrast to these existing methods, we do not try to modify the problem by extending it to the encompassing domain/grid $\mathbb{B}/\mathbb{B}^m$ but rather simply try to find "the best" among the solutions of the underdetermined problem (1.2). After all, if you use high order $\mathbb{B}^m$-based discretizations of derivatives and evaluations, the equations should be sufficient for determining a solution that achieves their order of accuracy (up to what is allowed by the regularity of the data/solution themselves, of course).

A straightforward approach (which works fine when no regularity at all is expected) consists of finding a minimal norm solution of the problem, i.e., in solving the

linearly constrained optimization problem

$$\operatorname{argmin}_{\{Cu=b\}} \frac{1}{2}\|u\|_2^2, \tag{1.3}$$

where $\|\cdot\|_2$ denotes the Euclidean norm on $\mathbb{R}^{N_m}$. This would lead to the so-called normal equations and to the solution

$$u = C^\top \big(CC^\top\big)^{-1} b. \tag{1.4}$$

Given that the matrix $C = C_m$ consists of differential operators including the evaluation (restriction) in the domain Ω^m and on the boundary Γ^m, its transpose then corresponds to differential operators containing trivial extensions (read extensions by 0), and this leads to oscillations generated by the lack of regularity. As a matter of fact, the solution of the continuous optimization problem

$$\operatorname{argmin}_{\{-\Delta_\Omega u=f,\gamma_\Gamma u=g\}} \|u\|^2_{\mathrm{L}^2(\mathbb{B})}$$

is simply given by $\mathrm{ext}_0(u_{f,g})$, where ext_0 denotes the trivial extension (i.e., by zero) of a function defined on $(-1,1)$ to $(-\pi,\pi)$, and $u_{f,g}$ is the unique solution of the BVP $-\Delta u = f$ with the given Dirichlet boundary condition. Even in the case $g \equiv 0$, however, the solution generated by the optimization problem takes the form of a difference of singular solutions, which do not even belong to L^2. This is the origin of the oscillations that are observed in numerical implementations (see Figure 2). The next remark offers a detailed explanation of this phenomenon in a one-dimensional context.

Remark 1.1. We illustrate this point more thoroughly with a one-dimensional example. We reformulate the BVP

$$\begin{cases} -\partial_{xx} u = f & \text{in } (-1,1), \\ u(\pm 1) = 0 \end{cases}$$

as the optimization problem (1.3), which we approach by introducing a Lagrange multiplier $\lambda = \big(\lambda_{(-1,1)}, \lambda_{-1}, \lambda_1\big)$ and reducing it to

$$\operatorname{argmin}_{v,\lambda} \Big\{ \frac{1}{2}\int_{-\pi}^{\pi} v^2(x)\,dx + \int_{-1}^{1} \lambda_{(-1,1)}(x)\big[\partial_{xx} v + f\big](x)\,dx + \lambda_{-1} v(-1) + \lambda_1 v(1) \Big\},$$

where $v : (-\pi,\pi) \to \mathbb{R}$ is a periodic function. Taking a variation with respect to v yields the validity of

$$\int_{-\pi}^{\pi} v\varphi\,dx = \int \lambda_{(-1,1)}(x)\partial_{xx}\varphi(x)\,dx - \lambda_{-1}\varphi(-1) + \lambda_1\varphi(1)$$

for any and all periodic testfunctions $\varphi \in \mathrm{C}^\infty_\pi$. Taking testfunctions satisfying $\mathrm{supp}(\varphi) \subset (-1,1)^{\mathsf{c}}$, one shows that $v = 0$ in $(-1,1)^{\mathsf{c}}$. Choosing testfunctions supported in $(-1,1)$ shows that $\partial_{xx}\lambda_{(-1,1)} = v$ in $(-1,1)$, and, finally, choosing testfunctions with $\varphi(\pm 1) = 0$ and others with $\partial_x\varphi(\pm 1) = 0$, one obtains the validity of

$$\lambda_{(-1,1)}(\pm 1) = 0 \quad \text{and} \quad \lambda_{\pm 1} = \pm\lambda'_{(-1,1)}(\pm 1).$$

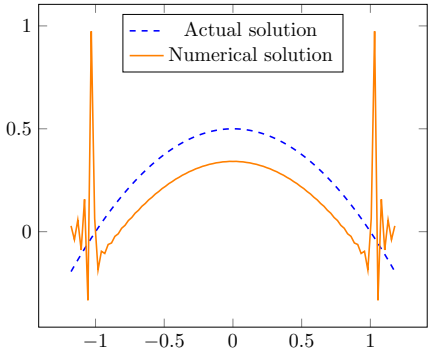

FIG. 2. *A 1D visualization of the oscillations caused by trivial extension with no regularization. The plot shows a region that is only slightly larger than Ω since the oscillations occur in a neighborhood of $\partial\Omega$.*

One can therefore solve for $\lambda_{(-1,1)}$ to see that $\lambda_{(-1,1)} = S_D u$, where S_D denotes the solution operator to $-\partial_{xx}$ with homogeneous Dirichlet conditions at the end points ± 1, and then determine $\lambda_{\pm 1}$. As it holds that

$$\lambda = (CC^\top)^{-1} \begin{bmatrix} f \\ 0 \end{bmatrix} \text{ and } v = C^\top \lambda,$$

it follows that

$$\begin{aligned} v &= \begin{bmatrix} -\partial_{xx} \circ \mathrm{ext}_0 & \delta_{-1} & \delta_1 \end{bmatrix} \begin{bmatrix} \lambda_{(-1,1)} \\ \lambda_{-1} \\ \lambda_1 \end{bmatrix} \\ &= -\partial_{xx}\big(\mathrm{ext}_0(\lambda_{(-1,1)})\big) + \lambda'_{(-1,1)}(-1)\delta_{-1} - \lambda'_{(-1,1)}(1)\delta_1 = \mathrm{ext}_0(u). \end{aligned}$$

The easily verified fact that the dual of the trace at a point is the Dirac distribution at the point, i.e., that $\gamma'_{\pm 1} = \delta_{\pm 1}$, was used to derive the above representation. Thus the solution v of the optimization problem is the trivial extension of the solution u of the original BVP, but it is obtained as the sum of singular terms with cancellation. In a discretization, the singular terms are generally not supported on grid-points and thus appear in the numerical solution as oscillations. This is made apparent in Figure 2. Moreover, the exact analytical cancellation cannot be expected to also happen at the discrete level in general. While we used a differentiated notation for u and v in this argument for clarity of exposition, the same will not be done in what follows; instead, the same notation will be used for the solution of the original problem and that of the optimization problem.

Reverting to the general discussion, we observe that while the solution obtained by the normal equation (1.4) exhibits oscillations in a discrete computation, "the good" (regular and nonoscillatory) solution is, however, among those of the underdetermined problem (1.2). It can be obtained by requiring additional regularity. As already pointed out, the discretizations A^m and B^m are, after all, chosen to be of a desired accuracy, and the truncations/trivial extensions destroy it. Thus, enforcing an appropriate degree of regularity should allow for the recovery of the intrinsic accuracy of the chosen discretizations, again, compatibly with the expected regularity of the solution itself. We refer to the proposed method as the smooth extension embedding

method (SEEM) since it implicitly selects a smooth extension of the solution. While this selection is done in a way that is natural from the point of view of optimization [3, Chapter 10], it has a nice analytic interpretation which will greatly help with the practical implementation of the method. Let $\|\cdot\|_S$ be the discretization of a high order norm such as, for instance, $\|\cdot\|_{\mathcal{S}} = \|(1-\Delta_\pi)^{p/2}\cdot\|_2$, where $-\Delta_\pi$ denotes the periodic Laplacian on $[-\pi,\pi]^d$ and $p \geq 1$. Now the problem becomes

$$\operatorname{argmin}_{\{Cu=b\}} \frac{1}{2}\|u\|_S^2, \tag{1.5}$$

where the indices have again been dropped for ease of reading. The constrained optimization problem (1.5) can be reformulated as the unconstrained minimization

$$\operatorname{argmin}_{u\in\mathbb{R}^{N_m},\,\Lambda\in\mathbb{R}^{N_\Lambda}} \frac{1}{2}\|u\|_S^2 + \Lambda^\top\big(Cu-b\big)$$

upon introduction of Lagrange multipliers $\Lambda \in \mathbb{R}^{N_\Lambda}$, where $N_\Lambda = N_m^\Omega + N_m^\Gamma$. A direct computation yields the saddle point system

$$\begin{bmatrix} S & C^\top \\ C & 0 \end{bmatrix}\begin{bmatrix} u \\ \Lambda \end{bmatrix} = \begin{bmatrix} 0 \\ b \end{bmatrix}, \tag{1.6}$$

where S is the (invertible) operator corresponding to the norm $\|\cdot\|_S$. Many techniques exist to solve such systems; see [1]. The simplest of these consists of forming the Schur complement, $CS^{-1}C^\top$, and of then obtaining the regularized normal equation

$$u = S^{-1}C^\top\big(CS^{-1}C^\top\big)^{-1}b. \tag{1.7}$$

Now, recalling that C and $C^\top$ are truncated differential operators (more precisely, containing differentiations, evaluations on subdomains, and extensions), we see that the effect of the norm is to replace the operator $C^\top$, which, upon being hit by C, is the cause of the oscillations in the straightforward method, by the regularized operator $S^{-1}C^\top$, which can be captured numerically to a higher degree of accuracy (no oscillations) when hit by C.

Remark 1.2. To illustrate the effect of regularization in the one-dimensional setting of Remark 1.1, consider the minimizer v corresponding to the objective functional given by the higher order expression $\frac{1}{2}\big[\|v\|_2^2 + \|\partial_x v\|_2^2\big]$. Proceeding in a similar way as in Remark 1.1, v is seen to be given by

$$(1-\partial_{xx})_\pi^{-1}\Big\{-\partial_{xx}\circ \operatorname{ext}_0\big[S_D(u+f)\big] + \sum_{j=-1,1} j\big[\big(\partial_x S_D(u+f)\big)(j) - u_x(j)\big]\delta_j\Big\}.$$

The singular terms are now regularized. Observe that the index in the regularizer indicates inversion of the operator in the periodic sense and that S_D is defined as in Remark 1.1.

Remark 1.3. Formula (1.7) can be used as a starting point without any knowledge of a norm generating the operator S. One can choose any convenient smoothing kernel acting on (generalized) functions defined on the box $\mathbb{B}$ instead of S^{-1}.

Remark 1.4. Although in this paper the method was implemented on the periodic torus using Fourier spectral methods, we emphasize that the method could easily be implemented in any container domain which admits the following:

- Efficient and well-developed discretizations of the differential operators and of the interpolation operators.
- An efficient smoothing operator S which enforces the regularity of functions on the grid.

In section 4.3, we use the finite element method to enforce the interior PDE in a weak sense. This allows us to look for an H^3 solution, where the strong pointwise formulation of the equations cannot be used for $d \geq 2$. Finally note that, in general, spectral methods will be most efficient for problems where a uniform grid is used, while finite difference and finite element methods are best deployed when adaptivity is useful or, in the case of the finite element method, when only a weak formulation is possible.

2. The smooth extension embedding method. We now give a detailed description of the implementation of our method. As described in subsection 1.1 and equation (1.3), the original BVP on Ω is rewritten as an optimization problem on an encompassing domain $\mathbb{B}$, given by

$$\operatorname{argmin}_{u\in\mathbb{R}^{N_m},\,\Lambda\in\mathbb{R}^{N_\Lambda}} \frac{1}{2}\|u\|_S^2 + \Lambda^\top\big(Cu-b\big),$$

where $\|\cdot\|_S$ is a higher order regularizing norm. In this paper we have used the H^p norms given by

$$\|\cdot\|_{S_p} := \|(1-\Delta_\pi)^{p/2}\cdot\|_2.$$

As mentioned in subsection 1.1, this leads to the saddle point system

$$\begin{bmatrix} S_p & C^\top \\ C & 0 \end{bmatrix}\begin{bmatrix} u \\ \Lambda \end{bmatrix} = \begin{bmatrix} 0 \\ b \end{bmatrix}.$$

We begin by describing the specific discretizations which we used to obtain the operators C, $C^\top$, and S. We will then describe some aspects regarding the choice of a smoother; in particular, we will discuss the proper order p to use for a given problem.

2.1. Discretization of the domain. As described in the introduction, we begin by embedding the domain Ω into a torus $\mathbb{B}$ in order to make use of spectral methods and of the Fourier transform. The periodicity box $\mathbb{B}$ is discretized with a uniform grid $\mathbb{B}^m$. The boundary Γ is approximated with a discretization Γ^m, which is just a set of N_m^Γ points lying on $\partial\Omega$. In practice, it is best for these points to be uniformly distributed across the boundary. In two dimensions, this can be accomplished easily by equally spacing points along an arc length parametrization of the curve. In three dimensions, equally distributing the points around a surface is more challenging, although straightforward algorithms exist for obtaining roughly equally spaced points on a given surface; see, for example, [17].

A choice also needs to be made concerning the density of boundary points, that is, the value of N_m^Γ. When using an insufficient number of points on the boundary, the accuracy suffers, while too many points can drive up the condition number. In this paper, where our encompassing domain is $[-\pi,\pi)^2$, we have found a density of $\frac{m}{4\pi}$ boundary points per unit length, where m is the number of grid-points along one dimension, to be most effective. This guarantees that 1–2 regular grid-points lie between any two boundary points and thereby allows the regular grid $\mathbb{B}^m$ to easily "distinguish" the different boundary points and to keep the condition number relatively low. In Figure 1, we show the discretization of the unit disc and of a star-shaped domain. For better visualization, we have only plotted $[-1.3,1.3]^2$, as opposed to the

TABLE 1

Number of collocation points used in the discretization of the disc of radius 2 used in subsections 4.2 and 4.3. N_m is the size of the full computational grid, N_m^Ω is the number of interior collocation points, and N_m^Γ is the number of boundary collocation points.

N_m	N_m^Ω	N_m^Γ	N_m	N_m^Ω	N_m^Γ
16^2	81	17	256^2	20,865	257
32^2	325	33	512^2	83,421	513
64^2	1,305	65	1024^2	333,669	1,025
128^2	5,209	129	2048^2	1,335,029	2,049

entire computational domain $[-\pi,\pi)^2$. In Table 1, we show the number of boundary collocation points used to discretize the disc of radius 2 in the numerical experiments in subsections 4.2 and 4.3. In three-dimensional problems, we have found that with a grid of size m^3 points on $[\pi,\pi)^3$, a boundary spacing of $\frac{m^2}{2\pi^2}$ per unit area is most effective.

Concerning the optimal size of the buffer zone $\mathbb{B}\setminus\Omega$, we note that it needs to allow for a smooth function on Ω to be extended into a smooth periodic function of the bounding periodicity box. If the buffer is too small, the extension will necessarily have large H^p norms and thereby decrease the accuracy of the solution. In the numerical experiments of subsections 4.2 and 4.3, we have studied a disc of radius 2. We have found that with a radius larger than 2.75, the accuracy of the solution begins to decrease. The need for a large buffer can be removed if one used a discretization which does not require periodic boundary conditions, such as a Chebyshev grid or a finite difference grid.

2.2. Discretization of the PDE operator. We recall that

$$C = \begin{bmatrix} A \\ B \end{bmatrix},$$

where A is a discretization of the interior differential operator $\mathcal{A}$, and B is a discretization of the boundary operator $\mathcal{B}$. As we explained in the introduction, the advantage of using a fictitious domain method is that these discretizations can be carried out in a straightforward and efficient way on a regular grid. We will discuss how to form each of these discretizations.

2.2.1. Discretization of interior PDE $\mathcal{A}$. Recall that $\mathcal{A}$ is a second order differential operator of the form

$$\mathcal{A}u(x) = \sum_{i,j=1}^{d} a_{ij}(x)\partial_i\partial_j u + \sum_{i=1}^{d} b_i(x)\partial_i u + c(x)u.$$

To discretize $\mathcal{A}$, we will need to introduce the restriction operator

$$R_\Omega : \mathbb{R}^{N_m} \to \mathbb{R}^{N_m^\Omega},$$

which maps a function defined on $\mathbb{B}^m$ to its values at the points of Ω^m. We now let D_i be the discretization of the derivative ∂_i on $\mathbb{B}^m$. We let $\bar{a}_{ij}$, $\bar{b}_i$, and $\bar{c}$ be the vectors containing the values of the functions u, a_{ij}, b_i, and c, respectively, at the points of Ω^m. Similarly, let $\bar{u}$ be a discrete function defined on $\mathbb{B}^m$, and let $\bar{v}$ be a discrete function defined on Ω^m. The discretization of $\mathcal{A}$ is then given by

$$A\bar{u} = \sum_{i,j=1}^{d} \bar{a}_{ij} R_\Omega D_i D_j \bar{u} + \sum_{i}^{d} \bar{b}_i R_\Omega D_i \bar{u} + \bar{c} R_\Omega \bar{u}.$$

Similarly,

$$A^\top \bar{w} = \sum_{i,j=1}^{d} D_j^\top D_i^\top R_\Omega^\top \bar{a}_{ij} \bar{w} + \sum_{i}^{d} D_i^\top R_\Omega^\top \bar{b}_i \bar{w} + R_\Omega^\top \bar{c} \bar{w}.$$

Here, $R_\Omega^\top$ is the transpose of R_Ω and amounts to an extension by 0 from Ω^m to $\mathbb{B}^m$.

The derivative matrices D_i can be obtained using any straightforward method to implement discretization on the background grid $\mathbb{B}^m$. For a regularly spaced grid on the periodic box, which we adopt in this paper, the matrices D_i are a spectral discretization implemented via the use of the FFT. Such a discretization allows one to achieve up to spectral accuracy. Limitations in accuracy are, of course, imposed by the regularity of the solution and the regularity of the selected smoothing operator. Finite difference discretizations, such as the five point stencil, offer an alternative to a spectral discretization and would enhance efficiency. In the numerical experiments in subsections 4.1 and 4.2 below, we evaluate the derivatives appearing in the operator A spectrally. More specifically, whenever taking the Laplacian, for instance, we compute

$$(-\Delta)^m = \left(\mathcal{F}^m\right)^{-1} \operatorname{diag}\Big(\big(|k|^2\big)_{k\in\mathbb{Z}_m^d}\Big)\mathcal{F}^m,$$

where $\mathcal{F}^m$ is the discrete FFT, and $k \in \mathbb{Z}_m^d$ is the frequency vector at discretization level m. In subsection 4.3, where a weak formulation of the BVP is studied, derivatives will be taken by means of a five point stencil. This is equivalent to a finite element discretization and will therefore provide a proper discrete weak formulation of the problem.

2.2.2. Discretization of the boundary operator $\mathcal{B}$. Recall that $\mathcal{B}$ is a boundary operator on Γ given by

$$\mathcal{B}u = a(y)\gamma_\Gamma u + b(y)\partial_{\nu_\Gamma} u.$$

B is a discretization of $\mathcal{B}$ on the set $\Gamma^m \subset \Gamma$. Since the boundary points Γ^m do not lie on the regular grid, interpolation operators are needed when imposing (discrete) boundary conditions. We let

$$I_{y_i} : \mathbb{B}^m \to \mathbb{R}$$

be an interpolant which estimates a grid function at y_i. As before, we denote by $\bar{a}$ and $\bar{b}$, respectively, the vectors of values of the coefficients a and b at the points of Γ^m. Finally, we let $\bar{\nu}$ be the vector of values of ν at the points Γ^m. The ith row of B is then given by

$$B_{i\bullet} = \bar{a}_i I_{y_i} + \bar{b}_i \sum_{j=1}^{d} (\bar{\nu}_i)_j I_{y_i} D_j.$$

As before, because we are interpolating on a regular grid, the interpolation operators are simple. Linear, cubic, or spectral interpolation can be used. The latter is used in subsection 4.1, where we compute the QR factorization of the explicit matrices. We do this in order to demonstrate the high order of convergence that the method can achieve. In subsections 4.2 and 4.3, where iterative methods are implemented, cubic polynomial interpolation is used in order to sparsify the interpolation matrix and thereby allow for computation on denser grids. When using polynomial interpolation, the computational complexity of the operator B is $O(N_m^\Gamma)$.

2.3. Discretization of the smoother $\mathcal{S}_p$. We now discuss the discretization of the continuous smoother

$$\mathcal{S}_p^{-1} u = (1 - \Delta_\pi)^{-p} u.$$

This choice is made because $\mathcal{S}_p$ is diagonalized by the Fourier transform, leading to simple and efficient computations. In particular, we let $k \in \mathbb{Z}_m^d$ be the vector of frequencies on the discrete grid $\mathbb{B}^m$. Then, the discrete smoother is given by

$$S_p^{-1} b = \left(\mathcal{F}^m\right)^{-1} \operatorname{diag}\left[(1 + |k|^2)^{-p}\right] \mathcal{F}^m b.$$

Using the FFT, we see that application of this operator has computational complexity $O(N_m \log N_m)$.

2.4. Choice of p for the smoother S_p. We recall that the purpose of the smoother is to enforce the H_π^p regularity of the selected solution to the underdetermined problem. Thus, when p is chosen to be larger, a more regular approximate solution is produced, and the true solution can be approximated with a higher order of convergence. In particular, in accordance with Remark 1.3, if the true solution $u \in H^{p+2}$, then using the smoother S_p will yield an order p convergence of the L_2 error. This will also be demonstrated experimentally in subsection 4.1. Thus, it is advantageous to use p, which gives the optimal order of convergence for the regularity class of the true solution u. For example, if the true solution $u \in H^6$, one should use the S_4 smoother to obtain fourth order convergence of the L_2 error.

Although the order of convergence increases with p, using a higher order smoother greatly increases the condition number of the resulting matrices. In particular, the matrix S_p will have a condition number which grows like m^p (before preconditioning), where m is the number of grid-points along each direction. Thus, using a larger p will generally require more iterations for convergence than using a smaller p on an equivalent grid. Furthermore, for a given grid size, the order p of the smoother cannot be pushed too high without hitting the limits of numerical precision. We recall that the smoother is given by $(1 - \Delta_\pi)^{-p}$. In Fourier space, this corresponds to multiplication by the function $(1 + |k|^2)^{-p}$. If k_* is the largest mode, as soon as $|k_*|^{-2p}$ drops below machine precision, which is roughly $1e - 16$, some matrix entries can no longer be captured numerically, and the benefits of accuracy are lost. For example, on a grid of size 64^2, the highest order smoother which can be used is $p = 5$ (a way around this issue will be discussed in subsection 3.2). Thus, the smoother must be chosen to balance the greater numerical accuracy obtained with the numerical issues which arise as p is increased. These issues will be more fully discussed in subsections 3.1 and 3.3.

Remark 2.1. As discussed above, the order of convergence of the solution is constrained by the regularity of the true solution, the order of the smoother, the interpolation operators, and the differential operators. To avoid wasting computational resources, the order of accuracy of all the various discretizations should be made to match.

3. Methods for solving the linear system. As mentioned previously, in the proposed method the BVP is reformulated as the saddle point system

$$\begin{bmatrix} S_p & C^\top \\ C & 0 \end{bmatrix} \begin{bmatrix} u \\ \Lambda \end{bmatrix} = \begin{bmatrix} 0 \\ b \end{bmatrix}.$$

Such problems arise in many areas of computational mathematics and particularly in optimization. Many techniques exist for solving this system implicitly without forming S_p^{-1}. However, because of our special choice of smoothing operator, S_p^{-1} can be evaluated explicitly using the FFT. We describe here two methods for solving such systems.

3.1. The Schur complement method. The system can be most simply solved by forming the Schur complement, $CS_p^{-1}C^\top$. It is straightforward to see that the solution then satisfies

$$u = S_p^{-1}C^\top \left(CS_p^{-1}C^\top\right)^{-1} b.$$

The Schur complement $CS_p^{-1}C^\top$ is a symmetric positive definite matrix and can therefore be inverted using the conjugate gradient method. As mentioned previously, all operators can be implemented using sparse matrices and the FFT. The Schur complement matrix can therefore be applied implicitly very efficiently. Although the matrix S_p is of order $2p$ and is therefore ill-conditioned for large grids, straightforward preconditioning exists which allows the preconditioned conjugate gradient (PCG) method to converge quickly; see subsection 3.3. Thus, the PCG method can be used to efficiently compute the solution on very dense grids. In the numerical experiments of section 4, we use the PCG method with the preconditioner described in subsection 3.3 in order to compute the solution.

3.2. The pseudoinverse method. Alternatively, the solution satisfies

$$u = S_p^{-1/2}(CS_p^{-1/2})^+ b,$$

where $(CS_p^{-1/2})^+$ is the pseudoinverse of $CS_p^{-1/2}$. To see this, we consider the QR decomposition of the matrix

$$S_p^{-1/2}C^\top = QR.$$

Here, using the notation of subsection 1.1, $Q \in \mathbb{R}^{N_m \times N_m^\Lambda}$ is an orthogonal matrix satisfying $Q^TQ = I$, while $R \in \mathbb{R}^{N_m^\Lambda \times N_m^\Lambda}$ is an upper triangular matrix. We recall that the pseudoinverse of $CS_p^{-1/2}$ is given by $Q(R^\top)^{-1}$. We then see that

$$\begin{aligned} S_p^{-1}C^\top(CS_p^{-1}C^\top)^{-1} &= S_p^{-1/2}QR(R^\top Q^\top QR)^{-1} \\ &= S_p^{-1/2}QRR^{-1}(R^\top)^{-1} \\ &= S_p^{-1/2}Q(R^\top)^{-1} \\ &= S_p^{-1/2}(CS_p^{-1/2})^+. \end{aligned}$$

The pseudoinverse can be calculated by using either the QR or the SVD decomposition. Furthermore, these computations can be performed by either using the explicit matrices or implicitly using sparse SVD algorithms. Using the pseudoinverse reduces the condition number of the matrix $CS_p^{-1/2}$ to the square root of that of the full Schur complement. This makes it feasible to use denser grids before the problems of ill-conditioning described in subsection 2.4 set in. In the numerical experiments of subsection 4.1, the pseudoinverse method is used in addition to the Schur complement method. In order to compute the pseudoinverse, the QR decomposition of the explicit matrix $CS_p^{-1/2}$ is obtained first.

Remark 3.1. The pseudoinverse method can also be viewed as a nonsymmetric collocation method (also known as Kansa's method) for solving PDEs. Given a kernel $\mathcal{K} : \Omega \times \Omega \to \mathbb{R}$, solutions are sought in the form

$$u = \sum_{i=1}^{|\mathbb{B}^m|} \alpha_i \mathcal{K}(\cdot, x_i).$$

This results in a nonsymmetric matrix, whose analysis is more difficult than that of symmetric collocation techniques. The pseudoinverse method can be viewed as a nonsymmetric collocation method, where the kernel function is given by

$$\mathcal{K}_p(\cdot, x) = (1 - \Delta_\pi)^{-p/2} \delta_x.$$

Thus, in this case the nonsymmetric collocation method is seen to be equivalent to the symmetric one.

3.3. Preconditioning for the Schur complement method. We now discuss the appropriate preconditioner for the PCG algorithm in the Schur complement method. As observed in subsection 3.1, the normal matrix $CS_p^{-1}C^\top$ is ill-conditioned and requires a good preconditioner to be inverted using the conjugate gradient method. The ill-conditioning stems from the use of the high order operator S_p^{-1} and from the difference in order between the boundary and the interior operators. Think of the operator $CS_p^{-1}C^\top$ as a block matrix

$$CS_p^{-1}C^\top = \begin{bmatrix} A^m S_p^{-1}(A^m)^\top & A^m S_p^{-1}(B^m)^\top \\ B^m S_p^{-1}(A^m)^\top & B^m S_p^{-1}(B^m)^\top \end{bmatrix} = \begin{bmatrix} C_1 & C_2 \\ C_2^T & C_3 \end{bmatrix}.$$

As S_p^{-1} represents an operator of order $-2p$ and A^m one of order 2, the matrix C_1 corresponds to an operator of order $4-2p$, C_2 to one of order $2-2p$, and C_3 to one of order $-2p$ (when choosing a discrete boundary operator B^m discretizing a continuous one of order 0). In general, if an operator is of order $-2p$, the condition number of its matrix approximation will grow like a polynomial of degree $2p$ as the grid size increases (for example, on a grid of size m, the largest eigenvalue of the Laplace operator will be of size m^2). Thus, the large order, together with the mismatch in scaling, causes a very large condition number. We will describe a simple preconditioner which works effectively for S_2, S_3, and S_4. The preconditioning consists of finding approximate inverses to the C_1 and C_3 blocks independently. Specifically, we use the preconditioner

$$\widetilde{C}^{-1} = \begin{bmatrix} \widetilde{C}_1^{-1} & 0 \\ 0 & \widetilde{C}_3^{-1} \end{bmatrix},$$

where $\widetilde{C}_1^{-1}$ and $\widetilde{C}_3^{-1}$ are approximate inverses to C_1 and C_3, respectively. The general philosophy consists of preconditioning these two operators so that they become order 0.

We begin by describing $\widetilde{C}_1^{-1}$. The matrix C_1 depends on the order of the smoother we have chosen. For S_2, we note that the matrix C_1 is of order 0. Thus, no preconditioning is necessary, and $\widetilde{C}_1^{-1}$ can be taken as the identity. For S_3 and S_4, we note that the operator C_1 is the discretization of a differential operator of order $4 - 2p$. We wish to precondition in such a way as to reduce the order of the operator to 0. Thus, we define the preconditioner

$$\widetilde{C}_1^{-1} u = \left(1 - \Delta_\Omega\right)^{\frac{2p-4}{2}} u.$$

Here, Δ_Ω is the Laplace operator on Ω. In order to implement it, we use the domain discretization $\Omega^m = \mathbb{B}^m \cap \Omega$ and a finite difference scheme to discretize the Laplacian on Ω^m. In the examples the five point stencil (seven point in three dimensions) was chosen.

As for an approximate inverse for the C_3 block, we will consider a Dirichlet problem, where the boundary operator $\mathcal{B}$ consists of evaluations on the boundary. The discrete boundary points belonging to Γ^m will be denoted by y_i for $1 \leq i \leq N_m^\Gamma$. We recall that

$$B^m : \mathbb{R}^{\mathbb{B}^m} \to \mathbb{R}^{\Gamma^m} \text{ and } S_p : \mathbb{R}^{\mathbb{B}^m} \to \mathbb{R}^{\mathbb{B}^m}.$$

Here the rows of B^m are discretizations of the delta distribution supported at the various boundary points. Thus, the matrix $C_3 = B^m S_p^{-1} (B^m)^\top$ is a discretization of the collocation matrix

$$\mathcal{M}_{ij} = \mathcal{K}_p(y_i, y_j) = \langle \delta_i, \mathcal{S}_p^{-1} \delta_j \rangle.$$

Thus, an excellent preconditioner for the C_3 block is obtained by inverting the explicit collocation matrix. Although this involves solving a collocation problem, seemingly defeating the whole purpose of discretizing the collocation method as we do in this paper, two crucial points must be recognized. First, the new collocation problem only resides on the boundary. Thus, the number of points is only N_m^Γ, as opposed to the $N_m^\Omega + N_m^\Gamma$ points of the full BVP. This dimensional reduction allows working with far denser grids. Second, because the matrix $\widetilde{C}_3^{-1}$ is only used as a preconditioner, we do not need to actually calculate the inverse of the collocation matrix; any crude approximation or even any pseudodifferential operator of the proper order will do. Finally, as we will discuss shortly, the collocation matrix that is inverted does not depend on the actual differential and boundary operators arising from the problem, so a standard one can be used for all such operators.

In all the numerical experiments performed in this paper, we use few enough boundary points that it is possible to directly invert the collocation matrix. To build the latter, observe that $\mathcal{S}_p^{-1}$ is given by convolution with a kernel. Thus, it suffices to calculate it at one point and simply shift its "center" to the different collocation points. A discretization of the fundamental solutions of the continuous smoother $\mathcal{S}_p$, given by

$$h(y) = (\mathcal{S}_p^{-1} \delta)(y),$$

is computed on a dense grid. In our experiments we use a grid of size 4096^2 in two dimensions and 512^3 in three dimensions. This solution can be stored in a lookup table. The collocation matrix is then given by

$$\mathcal{M}_{ij} = h(y_i - y_j),$$

where the values are interpolated from those in the lookup table.

For the Neumann and Robin problems, we note that the order of the matrix C_3 is decreased by 2, because both C and $C^\top$ evaluate one derivative on the boundary. Thus, rather than using the function $\mathcal{S}_p^{-1}\delta$ in the collocation matrix, we instead resort to the function $\mathcal{S}_{p-1}^{-1}\delta$. See the supplementary material (M130084SupMat.pdf [local/web 2.01MB]) for the effect of this preconditioning on the Robin BVP.

Using the described preconditioning technique, the PCG method converges fast enough for an efficient evaluation on very dense grids. Numerically, we will find in subsection 4.1 that preconditioning is most effective for the S_2 smoother. As mentioned above, for a second order BVP, the matrix corresponding to the interior equations "is" order 0 with no preconditioning. As p increases, the preconditioning is

somewhat less effective. In general, the ill-conditioning can be somewhat improved by increasing the distance between the boundary points. Another approach to improving the condition number consists of modifying the smoother to $(1 - \epsilon\Delta_\pi)^{-p/2}$ for a parameter $0 < \epsilon < 1$. This did not prove necessary for the problems studied here. We refer the reader to Table 4 for actual condition numbers of the preconditioned matrices and to the numerical experiments for the CPU times of the corresponding computations.

4. Numerical experiments. Next, several numerical experiments are discussed which demonstrate the effectiveness of the proposed method. Several more are presented in the supplementary material (M130084SupMat.pdf [local/web 2.01MB]). Recall that the order of convergence of the method depends on the regularity of the solution as well as on the choice of smoother S_p. We will begin by considering a Dirichlet BVP with analytic solution; with this choice, the orders of convergence of the different smoothers can be easily compared. Next, we will present two problems with lower global regularity. First, a BVP with global H^4 solution will be presented and solved using the S_2 smoother. It exhibits a convergence order of 2. Next, a function with H^3 regularity will be used. In this case, the equation cannot be verified pointwise; instead, a weak formulation will need to be introduced.

In the supplementary material (M130084SupMat.pdf [local/web 2.01MB]), we include four additional experiments. First, a Robin BVP with nonconstant coefficients with analytic solution and a three-dimensional Dirichlet BVP with analytic solution will be presented. Next, we will solve Dirichlet BVPs with H^6 and H^5 solutions. These will be solved with the S_4 and S_3 smoothers, respectively, and will exhibit convergence orders of 4 and 3, respectively; see Remark 1.3.

In the latter experiments, because it is difficult to explicitly write down a function which is globally H^p, we instead generate such functions by taking the inverse Fourier transform of a random sequence of coefficients which decay like $(1 + |k|^2)^{-p/2}$ and by restricting the resulting function to the domain Ω. This is done on a dense grid of sizes 8192^2. The right-hand side is obtained by evaluating the interior and boundary differential operators using spectral methods at the collocation points.

In subsections 4.2 and 4.3, the proposed method is compared with a baseline multiquadric (global) radial basis function (RBF) method in order to demonstrate its effectiveness. For the purpose of comparison, we have restricted these latter experiments to constant coefficient Dirichlet problems. We note that the RBF method has not been implemented for denser grids due to the memory and computational constraints of working with dense matrices; see subsection 5.2. We also compare the proposed method with an RBF-FD (finite difference) method using a stencil of size 25 and using third order polyharmonic splines as RBFs. The solution is calculated using the BiCG algorithm. We note for the RBF-FD, we were unable to obtain convergence for the densest grid with 4096^2 points. Furthermore, on the grid of size 2048^2, time to convergence was 3000 seconds, which is about 50 times slower than our implementation of SEEM.

All numerical experiments are performed on an Intel i7-7700HQ CPU. Experiments were implemented in Python, using the Numpy and Scipy libraries. For the comparisons with the RBF-FD method, we used a publicly available Python RBF library. For the PCG and BiCG methods, we used a tolerance of $1e-8$ as a stopping criterion.

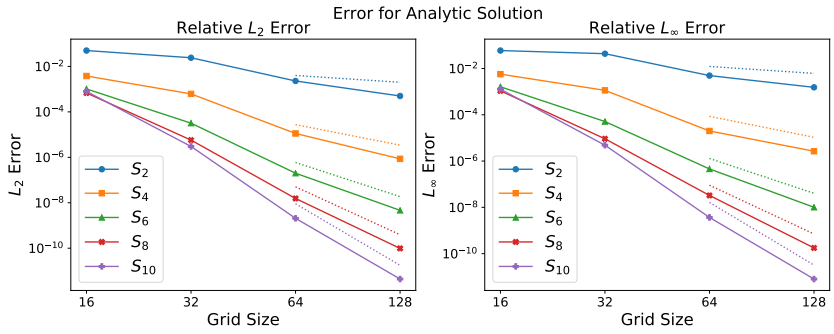

FIG. 3. *Convergence of the L_2 and L_∞ relative errors for different order smoothers solving (4.1). The light dotted lines are reference lines of slope $\frac{1}{m^p}$, where m is the number of grid points along one dimension.*

4.1. A Dirichlet BVP with analytic solution. The first choice of u is

$$u(x,y) = x^2 - y^2, \quad (x,y) \in D.$$

Since this function is analytic, any S_p could be used to give p order convergence. Although a problem as smooth as this could be trivially solved with a spectral order of convergence by simply collocating polynomials or other smooth RBFs, we still perform the experiment to demonstrate numerically how the order of convergence depends on the smoother S_p. We study the Dirichlet problem on the unit disc, with the given choice of u. The right-hand side is given by

$$\begin{cases} -\Delta u = 0 & \text{in } D, \\ u = x^2 - y^2 & \text{on } \partial D. \end{cases} \tag{4.1}$$

We begin by solving the problem using explicit matrices and the pseudoinverse method, as described in subsection 3.2. We show the convergence of the relative L_2 and L_∞ errors for the different smoothers S_p with $p = 2, 4, 6, 8, 10$. As we are using explicit matrices for this experiment, we have limited our grid size to 128^2 points. We show the errors in Figure 3 and in Tables 2 and 3. The order is calculated by comparing the errors on the densest grid to those on the sparsest grids. We next turn to demonstrating the effectiveness of our preconditioning techniques. We solve the same problem described in (4.1) using the Schur complement method with a PCG iteration, with the preconditioner described in subsection 3.3 and the S_p smoother for $p = 2, 3, 4$. For grids up to 128^2, we record the condition number of the preconditioned Schur complement matrix. We then solve the problem on grids up to size 512^2 and record the CPU times together with the number of PCG iterations for each of the three smoothers. As the grid becomes denser, computational times increase due to the computation of the Schur complement matrix and of its preconditioner as well as to the larger number of iterations needed for convergence. We note that, as described in subsection 2.4, the higher order smoothers are poorly conditioned and require more iterations to converge, although they achieve greater accuracy. The results are listed in Table 4.

4.2. A Dirichlet BVP with H^4 solution. We now consider a problem with lower global regularity, where our method is advantageous. To generate a globally H^4

TABLE 2
Relative L_2 error solving (4.1).

Grid size	Relative L_2 error				
	S_2	S_4	S_6	S_8	S_{10}
16^2	$5.00E-02$	$3.79E-03$	$1.02E-03$	$6.78E-04$	$7.96E-04$
32^2	$2.41E-02$	$6.19E-04$	$3.19E-05$	$5.74E-06$	$3.01E-06$
64^2	$2.29E-03$	$1.12E-05$	$2.00E-07$	$1.55E-08$	$2.07E-09$
128^2	$5.00E-04$	$8.58E-07$	$4.64E-09$	$9.82E-11$	$4.40E-12$
Order	2.21	4.04	5.92	7.57	9.14

TABLE 3
Relative L_∞ error for (4.1).

Grid size	Relative L_∞ error				
	S_2	S_4	S_6	S_8	S_{10}
16^2	$5.96E-02$	$5.70E-03$	$1.62E-03$	$1.08E-03$	$1.27E-03$
32^2	$4.36E-02$	$1.14E-03$	$5.11E-05$	$9.18E-06$	$4.85E-06$
64^2	$4.94E-03$	$2.00E-05$	$4.57E-07$	$3.27E-08$	$3.67E-09$
128^2	$1.53E-03$	$2.67E-06$	$1.01E-08$	$1.77E-10$	$8.00E-12$
Order	1.76	3.69	5.76	7.52	9.08

function, we take a random Fourier series $\{s_{ij}\}$ with $-2047 \le i, j \le 2048$, where we impose the decay rate

$$|s_{ij}| \le \frac{1}{(1+|i|^2+|j|^2)^3}.$$

We then use the inverse discrete Fourier transform to generate a globally H^4 function defined on a grid of size 4096^2. We use this function as our solution. A contour plot of the (unrestricted) function can be seen in Figure 4. We then solve the Dirichlet BVP on the disc of radius 2. The Laplacian of u as well as its values on the boundary are then generated from the original random Fourier series. As $u \in H^4$, in accordance with Remark 1.3 and the considerations of subsection 2.4, we use the S_2 smoother and expect second order convergence for the L_2 error. The calculated L_2 and L_∞ errors as well as the CPU times are shown in Figure 5 and Table 5. In the graph, we compare the convergence with that of a multiquadric RBF with scaling parameters $\epsilon = 3$ and 5. These values of ϵ were chosen for the best accuracy. Grids of size larger than 128^2 were not solved using the RBFs due to the RAM constraints of the dense matrices.

4.3. A Dirichlet BVP with H^3 solution. Next, consider $u \in H^3$ generated using the same technique used in the previous section. A contour plot of the (unrestricted) function can be seen in Figure 6. In this case, $-\Delta u \in H^1$ and thus cannot be defined pointwise. We therefore need to turn to a weak formulation of the problem. We proceed as follows. Rather than imposing the pointwise conditions

$$-\Delta u(x_i) = f(x_i), \quad x_i \in \Omega^m$$

(because f cannot be evaluated at a point), we instead resort to a weak formulation on the interior. Letting $\{\phi_i\}$ be the standard finite element basis of piecewise linear functions on the regular grid $\mathbb{B}^m$, we seek a solution of the form

$$u = \sum_{i=1}^{N_m} c_i \phi_i.$$

TABLE 4

CPU times, number of iterations, and condition number for the Schur complement PCG method solving (4.1) *using the method described in subsection* 3.1. *The condition number is that of the preconditioned Schur complement matrix* $\widetilde{C}^{-1/2}(CS_p^{-1}C^\top)\widetilde{C}^{-1/2}$.

Grid size	PCG iterations			CPU time			Condition number		
	S_2	S_3	S_4	S_2	S_3	S_4	S_2	S_3	S_4
16^2	18	22	27	0.01	0.01	0.01	6	17	61
32^2	25	36	51	0.01	0.03	0.04	8	22	84
64^2	27	41	73	0.02	0.05	0.07	10	24	152
128^2	31	47	99	0.06	0.11	0.25	12	27	305
256^2	34	48	158	0.53	0.64	2.08	-	-	-
512^2	38	69	347	2.28	5.41	34.82	-	-	-

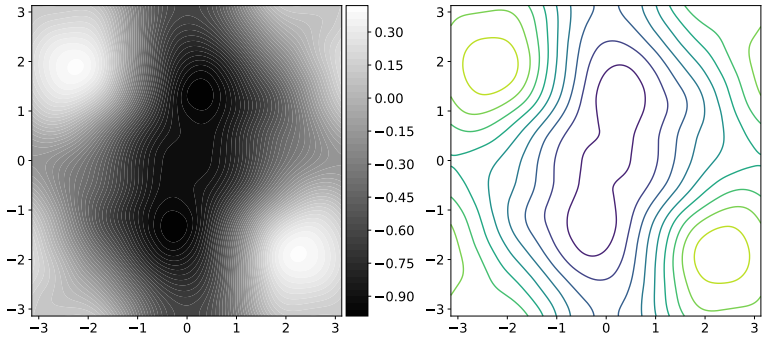

FIG. 4. *Contour plots of* H^4 *solution.*

We then take the subset of those ϕ_i's, the support of which is entirely within the domain Ω. We denote this set Ω^m and set $|\Omega^m| = N_m^\Omega$. The interior conditions

$$\int_\Omega \nabla u \nabla \phi_i = \int_\Omega f \phi_i$$

are imposed for $\phi_i \in \Omega^m$. Here, the integral $\int_\Omega f\phi$ is calculated using Fourier series on a dense grid. The boundary constraint is unchanged from the strong formulation. In line with our proposed method, we then seek a solution which minimizes the discrete H^2 norm and is computed using the discrete Fourier transform of c. Notice that for a uniform grid, the differentiation matrix for the basis ϕ_i coincides with the standard five point stencil finite difference discretization. We solve the Dirichlet BVP on the disc with radius 2. The resulting L_2 and L_∞ errors, as well as the CPU times, are shown in Figure 7 and Table 6. We note that in [9], the claim is made that the usage of meshfree methods for second order elliptic problems is restricted to $u \in H^p$ with $p > 2 + \frac{d}{2}$, which leads to a $p-2$ order of convergence of the L^2 error. Here, using a discrete collocation procedure, we are able to treat weaker solutions as well, which leads to second order convergence for an H^3 solution as opposed to first order convergence for the meshfree implementation of [9] for $H^{3+\epsilon}$ solutions.

Remark 4.1. Analytically, only nodal basis functions which are compactly supported inside Ω should be used. However, numerically it is better to include any basis functions with support intersecting Ω. For such basis functions ϕ, we impose the

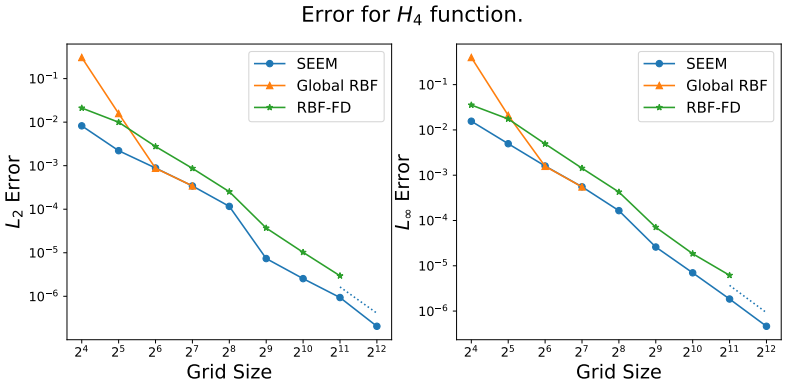

FIG. 5. *Plots of relative L_2 and L_∞ errors for H^4 solution. The dotted reference lines have a slope $\frac{1}{m^2}$, where m is the number of grid points along one dimension.*

TABLE 5
Error and computation times for H^4 solution.

Grid size	L_2 error	L_2 order	L_∞ error	PCG iterations	CPU time
16^2	$8.23E-03$	-	$1.56E-02$	26	0.36
32^2	$2.21E-03$	1.65	$4.96E-03$	27	0.24
64^2	$8.84E-04$	1.64	$1.60E-03$	28	0.29
128^2	$3.42E-04$	1.52	$5.55E-04$	29	0.43
256^2	$1.16E-04$	1.75	$1.65E-04$	31	0.89
512^2	$7.36E-06$	2.67	$2.59E-05$	31	2.53
1024^2	$2.54E-06$	1.89	$6.99E-06$	34	10.05
2048^2	$9.36E-07$	1.92	$1.84E-06$	44	47.87
4096^2	$2.05E-07$	2.00	$4.61E-07$	58	277.31

condition

$$c\int_{\mathbb{B}} \nabla u \nabla \phi = \int_{\Omega} f\phi, \qquad \text{where } c = \frac{\int_{\Omega}\phi}{\int_{\mathbb{B}}\phi}.$$

5. Relation to other methods.

5.1. Symmetric kernel-based collocation. The method described in this paper falls within the theoretical framework of symmetric kernel-based collocation. The latter was mainly developed by Franke and Schaback in [10, 9] in the context of meshfree methods. For the benefit of the reader, we give a brief outline of the theory of kernel-based collocation and describe how our method fits into the general theory. We will then describe how the proposed method differs from existing meshfree methods.

A symmetric, positive definite kernel $\mathcal{K}$ on a domain Ω is a function

$$\mathcal{K} : \Omega \times \Omega \to \mathbb{R},$$

which satisfies

(i) $\mathcal{K}(x,y) = \mathcal{K}(y,x)$, $x, y \in \Omega$; and
(ii) $\sum_{i=1}^{n}\sum_{j=1}^{n} c_i c_j \mathcal{K}(x_i, x_j) \geq 0$ for any set of distinct points $\{x_1, \ldots, x_n\} \subset \Omega$ and for any $c \in \mathbb{R}^n$. Equality holds only if $c = 0$.

To each positive definite kernel $\mathcal{K}$ there corresponds a unique reproducing kernel Hilbert space. The latter is a Hilbert space $\mathcal{N}_{\mathcal{K}}$ of functions defined in Ω which satisfies

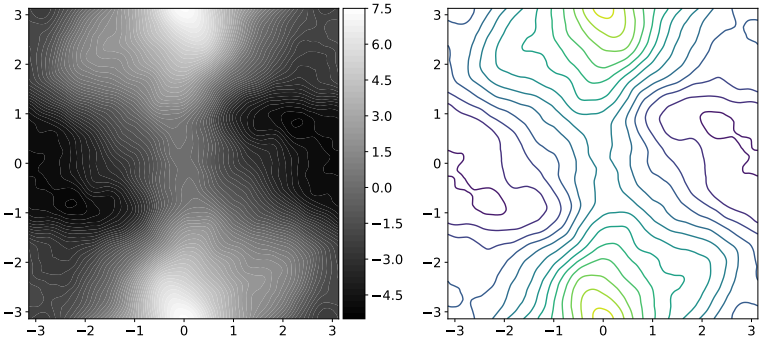

FIG. 6. *Contour plots of H^3 solution.*

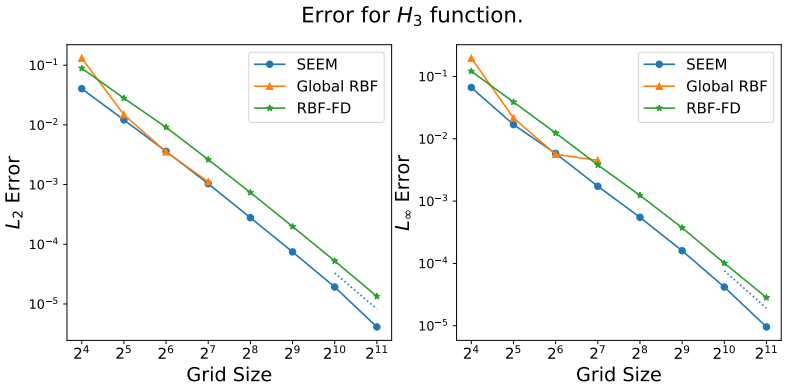

FIG. 7. *Plots of relative L_2 and L_∞ errors for H^3 solution. The dotted reference line have a slope $\frac{1}{m^2}$, where m is the number of grid points along one dimension.*

(i) $\mathcal{K}(\cdot, x) \in \mathcal{N}_\mathcal{K}$ for $x \in \Omega$; and
(ii) $f(x) = \langle f, \mathcal{K}(\cdot, x)\rangle$ for $f \in \mathcal{N}_\mathcal{K}$.

Given a symmetric positive definite kernel on $\overline{\Omega}$, in symmetric kernel-based collocation, a BVP is discretized by selecting two sets of collocation points,

$$\Omega^m = \{x_1, \dots, x_{N_m^\Omega}\} \subset \Omega \quad \text{and} \quad \Gamma^m = \{y_1, \dots, y_{N_n^\Gamma}\} \subset \Gamma.$$

The operator $\mathcal{C}$ is defined to act as $\mathcal{A}$ on Ω^m and as $\mathcal{B}$ on Γ^m. A solution is then sought in the form

$$\tilde{u}(\cdot) = \sum_{j=1}^{|\mathcal{Z}|} \alpha_j \mathcal{C}^{z_j} \mathcal{K}(\cdot, z_j),$$

where $\mathcal{Z} = \Omega^m \cup \Gamma^m$, and the coefficients α_j are chosen so that $\mathcal{C}(\tilde{u})(z_j) = b_j$ for $z_j \in \mathcal{Z}$. Here b is the right-hand side of the BVP evaluated at the points of $\mathcal{Z}$. In order to obtain the unknown coefficients α_j, one solves the linear system $\mathcal{M}\alpha = b$, where the *collocation matrix* $\mathcal{M}$ is given by

$$\mathcal{M}_{ij} = \mathcal{C}^{z_i} \mathcal{C}^{z_j} \mathcal{K}(z_i, z_j). \tag{5.1}$$

TABLE 6
Errors and CPU times for H^3 solution.

Grid size	L_2 error	L_2 order	L_∞ error	PCG iterations	CPU time
16^2	$4.04E-02$	-	$6.66E-02$	25	0.24
32^2	$1.21E-02$	1.74	$1.69E-02$	27	0.18
64^2	$3.61E-03$	1.75	$5.82E-03$	28	1.36
128^2	$1.03E-03$	1.81	$1.73E-03$	29	0.54
256^2	$2.79E-04$	1.89	$5.49E-04$	31	0.85
512^2	$7.45E-05$	1.91	$1.60E-04$	31	2.2
1024^2	$1.92E-05$	1.95	$4.18E-05$	34	11.4
2048^2	$4.12E-06$	2.22	$9.56E-06$	45	59.84

As shown in [20, Chapter 13], the solution $\tilde{u}$ obtained through this process is the $\|\cdot\|_{\mathcal{N}_\mathcal{K}}$-minimizing function satisfying $\mathcal{C}(\tilde{u})(z_i) = b_i$ for $z_i \in \mathcal{Z}$. In the special case where $\mathcal{N}_\mathcal{K}$ is the Sobolev space $H^{p+\epsilon}$ for $0 < \epsilon < 1$, the following convergence theorem holds.

THEOREM 5.1. *If $p > 2 + \frac{d}{2}$, $\mathcal{N}_\mathcal{K} = H^{p+\epsilon}$, $\mathcal{C}$ is a second order elliptic differential operator, $u \in H^{p+\epsilon}$, and $\tilde{u}$ is the $\mathcal{N}_\mathcal{K}$ norm minimizing function satisfying*

$$\mathcal{C}(\tilde{u})(z_i) = b_i,$$

then

$$\|u - \tilde{u}\|_{L^2} \leq Ch^{p+\epsilon-2}(\|f\|_{H^{p+\epsilon-2}} + \|g\|_{H^{p+\epsilon-\frac{1}{2}}}),$$

where h is the fill distance defined as

$$h = \max\left\{ \sup_{x\in\Omega} \min_{z\in\Omega^m} |x - z|, \sup_{x\in\Gamma} \min_{z\in\Gamma^m} |x - z| \right\}.$$

Proof. The theorem is a straightforward corollary of [20, Proposition 11.30], which states the following.

PROPOSITION 5.2. *If $p > \frac{d}{2} + k$, if $u \in H^{p+\epsilon}(\Omega)$ satisfies $u|_\mathcal{Z} = 0$ for some discrete set $\mathcal{Z} \subset \Omega$, and if h is the fill distance defined as*

$$h = \sup_{x\in\Omega} \min_{z\in\mathcal{Z}} |x - z|,$$

then

$$\|u\|_{H^k(\Omega)} \leq Ch^{p+\epsilon-k}\|u\|_{H^{p+\epsilon}}.$$

Assume that $\tilde{u}$ is the $\mathcal{N}_\mathcal{K}$ norm minimizing function satisfying $\mathcal{C}(\tilde{u})(z_i) = b_i$. First, note that by the definitions of $\tilde{u}$ and of the operators $\mathcal{A}$ and $\mathcal{B}$, and by the estimates of elliptic regularity, we know that

$$\begin{aligned} \|\mathcal{A}\tilde{u}\|_{H^{p+\epsilon-2}} + \|\mathcal{B}\tilde{u}\|_{H^{p+\epsilon-\frac{1}{2}}} &\leq \|\tilde{u}\|_{H^{p+\epsilon}} \leq \|u\|_{H^{p+\epsilon}} \\ &\leq C(\|f\|_{H^{p+\epsilon-2}} + \|g\|_{H^{p+\epsilon-\frac{1}{2}}}). \end{aligned} \tag{5.2}$$

Then calculate

$$\begin{aligned} \|u - \tilde{u}\|_{L^2} &\leq \|u - \tilde{u}\|_{H^2} \\ &\leq C(\|\mathcal{A}(u - \tilde{u})\|_{L^2(\Omega)} + \|\mathcal{B}(u - \tilde{u})\|_{H^{\frac{3}{2}}(\Gamma)}) \\ &\leq Ch^{p+\epsilon-2}(\|\mathcal{A}(u - \tilde{u})\|_{H^{p+\epsilon-2}} + \|\mathcal{B}(u - \tilde{u})\|_{H^{p+\epsilon-\frac{1}{2}}}) \\ &\leq Ch^{p+\epsilon-2}(\|f\|_{H^{p+\epsilon-2}} + \|\mathcal{A}\tilde{u}\|_{H^{p+\epsilon-2}} + \|g\|_{H^{p+\epsilon-\frac{1}{2}}} + \|\mathcal{B}\tilde{u}\|_{H^{p+\epsilon-\frac{1}{2}}}) \\ &\leq Ch^{p+\epsilon-2}(\|f\|_{H^{p+\epsilon-2}} + \|g\|_{H^{p+\epsilon-\frac{1}{2}}}). \end{aligned}$$

Here, the second inequality follows from elliptic regularity. The third is an application of Proposition 5.2. The fourth uses the definitions of f and g, while the fifth uses (5.2). The theorem follows. ∎

Returning to (1.7) in subsection 1.1, we see that the matrix $CS^{-1}C^\top$ is a discretized version of the collocation matrix (5.1) corresponding to the kernel and given by

$$\mathcal{K}(x,y) = (\mathcal{S}^{-1}\delta)(x-y).$$

Both operators $\mathcal{C}$ and $\mathcal{S}$ are discretized and evaluated on the regular grid. Furthermore, the discrete norm $\|\cdot\|_S$ is a discretization of the continuous norm $\|\cdot\|_{\mathcal{S}}$, which is the reproducing kernel Hilbert space norm associated with the kernel $\mathcal{S}^{-1}$. Thus, the proposed method can be characterized as a discrete version of kernel-based collocation; the encompassing domain $\mathbb{B}$ and the kernel $\mathcal{K}$ are chosen so that this discretization is straightforward and efficient. Our choice of the Sobolev kernels $\mathcal{S}_p^{-1} = (1-\Delta_\pi)^{-p}$ places us in the setting of Theorem 5.1, which leads to $(p-2)$ order convergence of the L_2 error.

Because the method fits within the framework of kernel-based collocation, the convergence results will follow from those of Theorem 5.1 (with an extra term added to account for the truncation error). However, as discussed in the next section, important computational benefits will derive from the choice of smoothing kernel and its discretization.

Remark 5.3. Although Theorem 5.1 only guarantees $(p-2)$ order convergence when using the pth order Sobolev kernel, in numerical experiments we observe pth order convergence instead, provided that the solution $u \in H^{p+2}$. This has been demonstrated in subsections 4.1–4.3.

5.2. Comparison with meshfree methods. Symmetric meshfree methods using RBFs are widely used for scattered data interpolation and for solving PDEs. The survey books [7, 20] describe both the theory and implementation of these methods. We give a short overview of some of these methods and focus on those that are most relevant to the proposed method.

In symmetric meshfree methods, the kernel $\mathcal{K}$ is generally chosen to be of the form

$$\mathcal{K}(x,y) = \Phi(|x-y|) = \Phi(r),$$

where $\Phi : \mathbb{R}^+ \to \mathbb{R}$ is a positive definite function known as the RBF of the method. Many choices are available including

(i) Gaussians, where $\Phi(r) = e^{-cr^2}$ for some $c > 0$; and
(ii) multiquadrics, where $\Phi(r) = \sqrt{1+cr^2}$ for $c > 0$.

These functions are often chosen as kernels because it is easy to perform computations on them. The corresponding collocation matrix $\mathcal{M}$ is computed explicitly by evaluating $\mathcal{C}^{z_i}\mathcal{C}^{z_j}\Phi(|z_i - z_j|)$. By using smooth, globally supported kernels, we observe high rates of convergence in line with Theorem 5.1. One of the advantages of meshfree methods is that the collocation points can be chosen arbitrarily. This freedom is particularly useful when only scattered data is available. Furthermore, the simplicity of the formulation of the method is very attractive.

When using globally supported RBFs the resulting collocation matrix is dense. For smooth problems this is not an issue, because the high order of convergence can achieve high accuracy with very few collocation points. For less smooth problems, however, where dense grids are necessary to resolve the solution's behavior, it becomes impractical to use such dense matrices. Additionally, when the RBF is smooth, the

collocation matrix is very poorly conditioned and thus severely limits the size of the set of collocation points; see [7, Chapter 16].

These issues lead to the introduction of a number of successful strategies aimed at speeding up the inversion of the collocation matrix and at reducing its condition number. Many of these techniques are described in [7]. Given the large variety of meshfree implementations, we will only briefly address those that are most pertinent to the proposed method.

(a) Globally supported RBFs can be replaced by the compactly supported ones as introduced by Wendland in [19]. In this way, the collocation matrix becomes sparser, and more collocation points can be used. Unfortunately, convergence only occurs if the width of the compactly supported functions is held constant; see [7, Chapter 41]. Consequently, the matrix loses its good sparsity properties as the mesh becomes finer. To remedy this problem, multilevel schemes are used. In these, compactly supported RBFs with varying supports are used. Those with wide support capture the coarse details, while those with narrow support capture the fine details. The use of such multilevel methods can improve the accuracy obtained from compactly supported RBFs; however, convergence issues still remain (see [7, Chapter 41]).

(b) A number of techniques exist which seek to circumvent the issue of the ill-conditioning of the collocation matrix by finding clever ways to compute the interpolant. These include the contour-Padé algorithm [7, Chapter 17] and the RBF-QR method [8]. We note that these methods approximate the RBFs using the truncation of a series expansion. The proposed method, in contrast, is discrete from the onset.

(c) Particularly relevant to us is the NFFT (nonuniform fast Fourier transform) method. In the NFFT method, the collocation matrix $\mathcal{M}$ is evaluated by using the inverse nonuniform FFT to obtain a Fourier series for the function on the torus. The kernel, a convolution operator, is then evaluated as a multiplication operator on the torus. Finally, the NFFT is applied to obtain the function values at the collocation points. By using this method, the dense collocation matrix can be evaluated with $O(N \log N)$ operations rather than with $O(N^2)$. When combined with efficient preconditioning, this method allows for the use of substantially larger grids. We will discuss the relationship of this method with the proposed one below.

(d) In RBF partition of unity (RBF-PUM) methods or RBF-FD methods, the global RBF method is localized to create sparse rather than dense matrices. In RBF-PUM, the global basis functions are multiplied by cutoff functions generated by a partition of unity of the domain. The resulting basis functions are compactly supported in a small region of the domain and therefore generate a sparse matrix. In RBF-FD, a small set of RBF basis functions is chosen at each point. The PDE operator is evaluated at each point using that point's chosen RBF functions to generate a high order finite difference stencil at each point. These methods are quite competitive and are rapidly growing in popularity. However, in localizing the problem, these methods lose the straightforward representation of the function as a linear combination of RBFs. Furthermore, the interpretation of the problem as a constrained optimization problem is lost.

In general, these methods evaluate the smoothing operator $\mathcal{S}^{-1}$ as well as the differential operator $\mathcal{C}$ explicitly. In the proposed method, by contrast, these are evaluated at the discrete level. By choosing the smoothing kernel to be defined on a finite regular encompassing domain (a torus in the implementations of this paper), it is possible to obtain a straightforward and efficient discretization of the operators. The obvious drawbacks are that the method is no longer meshfree and that the minimum distance between collocation points is limited by the distance of the grid-points.

However, the clear benefit is that the operators can be easily described using sparse matrices (either through the use of finite differences, of the FFT, or of other sparse schemes). Furthermore, the use of regular grid-points in the interior allows for simple preconditioning strategies, which substantially speed up computations. We also mention the following other benefits of the proposed method over common global RBF methods:

(i) Since all operators are discretized, other techniques for the solution of saddle point systems can be used. In general, these alternative methods allow one to solve the system while avoiding the direct computation of the dense matrix S^{-1}. The availability of such tools should allow for the choice of denser grids. Another useful tool that becomes viable is QR-factorization, the use of which is described in subsection 3.2. This technique makes it possible to replace the symmetric collocation method by an equivalent nonsymmetric collocation technique. This results in an improvement of the conditioning of the matrices and additionally provides a link between the symmetric and unsymmetric collocation methods.

(ii) Since all operators are described on a regular grid, their evaluation is very simple. For instance, one can use the standard finite difference stencil and cubic interpolation operators at all points. By contrast, in meshfree methods, the values $\mathcal{C}^{x_i}\mathcal{C}^{x_j}\Phi(x_i, x_j)$ can sometimes be complicated to compute, particularly for less straightforward differential operators, where many terms need to be evaluated. This issue is discussed in [7, Chapter 40], for instance.

(iii) Since the kernel matrix is discretized, it can be easily modified and tailored to fit specific problems. For example, in certain singular problems it may be beneficial to use weighted Sobolev norms. While such kernels would be quite difficult to compute explicitly, they can be easily evaluated on a regular grid in the discrete sense. Another possible modification would be to use nonquadratic objective functionals. Such modifications are a subject of current investigation.

While RBF-PUM and RBF-FD are RBF type methods allowing for the use of sparse matrices, SEEM preserves the global nature of the pure RBF approach. This has advantages from a theoretical standpoint. Furthermore, we believe that the global formulation will have applications to a number of problems where an explicit representation of the basis functions is useful; work is ongoing in these areas. Additionally, forming the RBF-FD and RBF-PUM matrices can be a complex process. In the proposed method, only straightforward spectral or finite difference discretizations need to be considered.

Remark 5.4. As implemented in this paper, the SEEM method uses the H^p norm on the torus to enforce regularity of the numerical solution. The smoothing operator is evaluated using the FFT. This allows for the evaluation of all operators in $O(N \log N)$ operations. This implementation is akin to that of the NFFT methods (see [7, Chapter 28]) in that both use the FFT to evaluate the collocation matrix $\mathcal{M}$. Indeed, using NFFT methods and efficient preconditioning, it is possible to use meshfree methods on dense grids with $O(N \log N)$ operations. However, our method distinguishes itself from these existing NFFT methods in several important ways. First, because all values are first interpolated to the regular grid, the regular inverse FFT (IFFT) can be used rather than the inverse nonuniform FFT (INFFT), which is more computationally complex. Second, rather than evaluating RBFs as convolution operators, we use the simple Sobolev kernels, which are more natural for the torus. As

pointed out above, this allows for the use of various other computational tools from the theory of saddle point problems. Third, while the proposed method has been implemented on the torus in this paper, it is a general procedure which can be deployed with any simple container domain using any discretized/discrete smoothing operator. For example, a Chebyshev grid could be used instead or, as done in subsection 4.3, a finite element discretization can be used to obtain a weak formulation of the BVP.

Remark 5.5. The proposed method is reminiscent of some of the techniques used in the computation of the INFFT; see [14, Chapter 5]. Given scattered data $\{u(x_i)\}_{i=1}^k$ on the torus, the INFFT seeks a Fourier series $\hat{u}$ which agrees with the data at the given points. As the problem is generally underdetermined, a Fourier series of minimal H^p norm is computed instead. The implementation in this paper can be seen as a generalization of the INFFT algorithm from the simple case of point evaluations of u to the case of general linear functionals.

5.3. Comparison with other fictitious domain methods. We recall that fictitious domain methods avoid the problems of mesh generation by embedding the domain into a simple encompassing computational domain and then using well-developed discretizations to evaluate the differential operators. At the heart of all these implementations is the need to resolve the mismatch between the boundary and the simple regular grid. There is a vast literature on these methods as they can be implemented in various discretization contexts, admit a variety of distinct practical implementations within each discretization framework, and be applied to many different BVPs of mathematical physics [16]. We refer the reader to the beginning of [15] for a brief outline of many of these methods. Given the volume of publications, the choice of references made here was merely motivated by the fact that they contain a description of the methods' philosophy and/or many useful additional references in their introduction. We will give a brief summary of several of these methods and state where we believe our method, which integrates the theory of kernel-based collocation, stands out.

5.3.1. Fictitious domain methods. A prominent implementation procedure, developed by Glowinski and coauthors in [5, 13, 12, 11] and known as the distributed Lagrange multiplier method, can be described in more detail as follows: think of the domain Ω as a subset of a larger regular simple domain $\mathbb{B}$, introduce a (uniform) discretization of $\mathbb{B}$, and solve the BVP by modifying the data (the right-hand side and/or the operator $\mathcal{A}$ in the prototypical situation considered here), which is usually done by extending them and by introducing artificially a weighted sum of carefully chosen source terms supported outside the domain Ω, i.e., in $\mathbb{B} \setminus \Omega$ or on its boundary Γ, by determining the weights (Lagrange multipliers) to ensure that the boundary condition is satisfied (or at least well approximated). We remark that a common characteristic of these techniques (and of immersed boundary methods as well) is that Neumann or Robin boundary conditions are "natural" and straightforward to include in the formulation, whereas Dirichlet boundary conditions are more challenging (see, e.g., [6]). These methods clearly have the advantage of not requiring special care or effort in the choice of discretization for $\mathbb{B}$. An often cited criticism of this approach is the need for extending the original elliptic operator $\mathcal{A}$ and/or right-hand-side f to corresponding objects defined on the whole of $\mathbb{B}$. This is not always straightforward, and simple extensions (such as the trivial one by zero outside Ω) introduce singularities into the problem and reduce the overall accuracy of the method. See [2] regarding methods for creating smooth extensions from Ω to $\mathbb{B}$

for the purpose of implementing fictitious domain methods. Another approach, in the context of finite elements, consists of modifying the problem's Dirichlet form to ensure that (nonnatural) boundary conditions are satisfied by possibly adding direct or more subtle penalty or penalty-like terms to it, such as, e.g., the so-called Nitsche method (see [4], for example). The approach proposed here can be viewed as a novel fictitious domain method which does not require any explicit extension of the data or modification of the original BVP. Moreover, it makes apparent that the real problem that any fictitious domain method has to solve is the selection problem among the infinitely many solutions of the original problem, which are generated as the problem is viewed in a larger domain where it becomes underdetermined. The direct way in which this is done here (introduction of a high order smoother) clearly shows how the order of accuracy chosen for the interior and boundary operators can be recovered in the extended problem through an affine shift obtained by a natural (from the point of view of both PDEs and optimization) regularization.

5.3.2. Immersed boundary methods. A very popular method used to deal with complex geometries, which is one of the motivations of this paper as well, is the so-called immersed boundary method, by which a problem is extended to a simple encompassing domain admitting robust and effective discretizations. The extension is obtained by the use of Dirac distributions in the distance from the boundary (more precisely, line and surface integral distributions along the boundary) and hence typically introduces singularities which reduce the overall accuracy of the method to first order. Recently, approaches have been proposed in which the accuracy is improved by the use of extension operators that preserve smoothness. We refer the reader to [18] in particular for an immersed boundary method which includes a smooth extension method, thereby preserving higher order accuracy, albeit at the cost of significant additional computational time (in what is called the preparation phase in [18]). We again point out that the method proposed here does not require any explicit extension since it identifies the solution among the infinitely many solutions of the extended, underdetermined problem by simply requiring smoothness in the full computational domain (and hence across the boundary) along with directly enforcing the PDE in Ω and the boundary conditions on $\partial\Omega$ by resorting only to the regular grid.

6. Conclusion. A new method is proposed for solving general BVP on complex domains, which avoids the need for generating a mesh. The method is a hybrid of the fictitious domain approach and kernel-based collocation. As proved in subsection 5.1 and as shown in the experiments in subsection 4.1, the method is able to achieve high orders of convergence like meshfree methods. Similar to fictitious domain methods, resulting matrices are sparse, which allows the method to scale to dense grids; the method can therefore be used to compute solutions with weaker global regularity; see subsections 4.2 and 4.3. The discretization of the kernel also allows for new insights into the theory of kernel-based collocation, including making a direct connection between symmetric and unsymmetric collocation (Remark 3.1) and allowing for weakly imposing the PDE and hence for computing solutions of weaker global regularity; see subsection 4.3.

Acknowledgments. The authors would like to express their gratitude to the anonymous reviewers for the careful reading of the manuscript and for their suggestions, which helped them craft a better final product.

REFERENCES

[1] M. Benzi, G. H. Golub, and J. Liesen, *Numerical solution of saddle point problems*, Acta Numer., 14 (2005), pp. 1–137.

[2] J. P. Boyd, *Fourier embedded domain methods: Extending a function defined on an irregular region to a rectangle so that the extension is spatially periodic and C^∞*, Appl. Math. Comput., 161 (2005), pp. 591–597.

[3] S. Boyd and L. Vandenberghe, *Convex Optimization*, Cambridge University Press, 2004.

[4] E. Burman and P. Hansbo, *Fictitious domain finite element methods using cut elements* II: *A stabilized Nitsche method*, Appl. Numer. Math., 62 (2012), pp. 328–341.

[5] E. J. Dean, Q. V. Dinh, R. Glowinski, J. He, T. W. Pan, and J. Périaux, *Least squares/domain imbedding methods for Neumann problems: Applications to fluid dynamics*, in Proceedings of the Fifth International Symposium on Domain Decomposition Methods for Partial Differential Equations, D. E. Keyes et al., eds, SIAM, 1992, pp. 451–475.

[6] S. Del Pino and O. Pironneau, *A fictitious domain based general PDE solver*, in Numerical Methods for Scientific Computing Variational Problems and Applications (Barcelona), Y. P. Kuznetsov, P. Neittanmaki, and O. Pironneau, eds., CIMNE, 2003.

[7] G. E. Fasshauer, *Meshfree Approximation Methods with MATLAB*, Interdiscip. Math. Sci. 6, World Scientific, 2007.

[8] B. Fornberg and C. Piret, *A stable algorithm for flat radial basis functions on a sphere*, SIAM J. Sci. Comput., 30 (2007), pp. 60–80, https://doi.org/10.1137/060671991.

[9] C. Franke and R. Schaback, *Convergence order estimates of meshless collocation methods using radial basis functions*, Adv. Comput. Math., 8 (1998), pp. 381–399.

[10] C. Franke and R. Schaback, *Solving partial differential equations by collocation using radial basis functions*, Appl. Math. Comput., 93 (1998), pp. 73–82.

[11] R. Glowinski and Q. He, *A least-squares/fictitious domain method for linear elliptic problems with Robin boundary conditions*, Comm. Comput. Phys., 9 (2011), pp. 587–606.

[12] R. Glowinski and Y. Kuznetsov, *Distributed Lagrange multipliers based on fictitious domain method for second order elliptic problems*, Comput. Methods Appl. Mech. Engrg., 196 (2007), pp. 1498–1506.

[13] R. Glowinski, T. W. Pan, and J. Périaux, *A fictitious domain method for Dirichlet problem and applications*, Comput. Methods Appl. Mech. Engrg., 111 (1994), pp. 283–303.

[14] S. Kunis, *Nonequispaced FFT: Generalisation and Inversion*, Ber. Math., Shaker Verlag, 2007.

[15] X. Li, J. Lowengrub, A. Rätz, and A. Voigt, *Solving PDEs in complex geometries: A diffuse domain approach*, Comm. Math. Sci., 7 (2009), pp. 81–107.

[16] R. Mittal and G. Iaccarino, *Immersed boundary methods*, Annu. Rev. Fluid Mech., 37 (2005), pp. 239–261.

[17] R. Palais, B. Palais, and H. Karcher, *PointClouds: Distributing Points Uniformly on a Surface*, preprint, https://arxiv.org/abs/1611.04690, 2016.

[18] 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.

[19] H. Wendland, *Piecewise polynomial, positive definite and compactly supported radial functions of minimal degree*, Adv. Comput. Math., 4 (1995), pp. 389–396.

[20] H. Wendland, *Scattered Data Approximation*, Cambridge Monogr. Appl. Comput. Math. 17, Cambridge University Press, 2004.