THE SMOOTH EXTENSION EMBEDDING METHOD WITH CHEBYSHEV POLYNOMIALS

DANIEL AGRESS, PATRICK GUIDOTTI, AND DONG YAN

Abstract. We propose an implementation of the Smooth Extension Embedding Method (SEEM), first described in [1], in the setting of Chebyshev polynomials. SEEM is a hybrid fictitious domain / collocation method which solves general boundary value problems in complex domains by recasting them as constrained optimization problems in a simple encompassing set. Previously, SEEM was introduced and implemented using a periodic box (read a torus) using Fourier series; here, it is implemented on a (non-periodic) rectangle using Chebyshev polynomial expansions. This implementation has faster convergence on smaller grids. Numerical experiments will demonstrate that the method provides a simple, robust, efficient, and high order fictitious domain method which can solve elliptic and parabolic problems in complex geometries, with non-constant coefficients, and for general boundary conditions. We consider applications to two and three dimensional boundary value problems as well as an initial boundary value problem via a genuinely space-time discretization.

1. Introduction

This paper focuses on the implementation of SEEM, previously introduced in [1], in the setting of Chebyshev polynomials. We begin with a brief overview which serves as a motivation and a description of SEEM's philosophy.

1.1. **SEEM.** We illustrate SEEM by studying a second order boundary value problem.

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

Here, $\mathcal{A}$ is a second order differential operator such as, e.g., the Laplace operator $-\Delta$, while $\mathcal{B}$ is a boundary operator such as, e.g., the trace for the Dirichlet boundary condition. Efficient and accurate spectral methods exist to solve (1.1) when Ω is a simple domain, such as a periodic box or a rectangle. However, for a more complicated Ω, these methods are not directly applicable. Fictitious domain methods seek to apply numerical procedures to general domains Ω by embedding into a simpler larger (fictitious) domain, $\mathbb{B}$, for which simple numerical methods (such as spectal ones, for instance) are available and simple to implement. One of the fundamental obstacles to such an approach is the fact that the BVP is only defined on Ω, which is a proper subset of $\mathbb{B}$. Consequently, the original problem only provides an under-determined set of equations for the unknowns, which are defined on the larger set $\mathbb{B}$. In fact, any extension of the solution u of the original BVP is a member of the affine family of solutions to the under-determined problem. Among these are solutions with low regularity and, consequently, discretizations of the problem will be unable to accurately approximate them. Previous methods have dealt with this issue by smoothly extending the BVP to the entire fictitious domain in such a way that the problem is no longer under-determined. However, this introduces the difficulty of properly extending all data of the problem while guaranteeing that the original equations are still satisfied, see [2].

Key words and phrases. Fictitious domain methods, embedding methods, numerical solution of boundary value problems, optimization problems, high order discretizations of PDEs, pseudo-spectral methods.

SEEM, by contrast, treats the original under-determined problem as a constraint and seeks to find a smooth representative of the affine family of solutions as the minimizer of an optimization problem defined on the whole fictitious domain $\mathbb{B}$. Specifically, given a norm $\|\cdot\|_{\mathcal{S}}$ on $\mathbb{B}$, the goal is to solve the constrained optimization problem

$$\operatorname{argmin}_{\{\mathcal{C}u=b\}} \frac{1}{2}\|u\|_{\mathcal{S}}^2, \tag{1.2}$$

Here, in order to simplify the notation, the entire BVP is rewritten as one equation

$$\mathcal{C}u = b, \text{ where } \mathcal{C} = \begin{bmatrix} \mathcal{A} \\ \mathcal{B} \end{bmatrix} \text{ and } b = \begin{bmatrix} f \\ g \end{bmatrix}.$$

We emphasize that the operators $\mathcal{A}$ and $\mathcal{B}$ are left in their original form, and only constrain the function on their domains Ω and Γ, respectively. The norm $\|\cdot\|_{\mathcal{S}}$ on $\mathbb{B}$ is chosen to enforce the desired degree of regularity. In this way, a smooth solution satisfying the constraint $\mathcal{C}u = b$ is selected. Because the selected solution is now smooth, a spectral discretization can approximate it to a high degree of accuracy. As the numerical experiments will demonstrate, the higher the regularity enforced by the norm $\|\cdot\|_{\mathcal{S}}$ on $\mathbb{B}$, the more accurate the approximation will be (compatibly with the expected regularity of the solution, of course.).

Next we briefly describe the discretization and the method for solving the optimization problem. To begin, the encompassing domain $\mathbb{B}$ is discretized by a regular grid $\mathbb{B}^m$, selected for the use of spectral methods. Let

$$\Omega^m = \Omega \cap \mathbb{B}^m$$

be the discretization of the interior and

$$\Gamma^m = \{y_1, \dots, y_{N_m^\Gamma}\}$$

be a discretization of the boundary obtained by placing roughly equally spaced points y_j along it. See Figure 1 for a depiction of two such grids and the corresponding boundary discretizations: one consisiting of Chebyshev roots and one consisting of trigonometric functions' roots, the latter yielding the standard Fourier grid of equally spaced points. The discrete unknown u^m can be thought of as a function on $\mathbb{B}^m$, whereas the discrete data f^m and g^m as functions on Ω^m and Γ^m, respectively. Then we take $b^m = \left[f^m, g^m\right]^\top$. The operators $\mathcal{C}$ and $\mathcal{S}$ are discretized as

$$C^m = \begin{bmatrix} A^m \\ B^m \end{bmatrix}$$

and S^m, where A^m and B^m represent an interior discrete differential operator at the points of Ω^m discretizing $\mathcal{A}$ and a discrete realization of the boundary conditions on Γ^m encoded by $\mathcal{B}$, respectively. The matrix S^m is a discretization approximating the norm $\|\cdot\|_{\mathcal{S}}$. Spectral discretizations are chosen for these operators in order to preserve the accuracy of the method. The original optimization problem can now be described using the discretized operators. Dropping indices for clarity, we obtain the problem

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

This optimization problem reduces to the saddle point problem given by

$$\begin{bmatrix} S & C^\top \\ C & 0 \end{bmatrix} \begin{bmatrix} u \\ \lambda \end{bmatrix} = \begin{bmatrix} 0 \\ b \end{bmatrix}, \tag{1.4}$$

where λ is a Lagrange multiplier for the constraint $Cu = b$. Multiple methods exist to solve such problems when the matrix S^{-1} cannot be formed directly. However, since we work with spectral

methods for which the matrix S^{-1} can be efficiently calculated using the FFT, we focus on two direct methods.

1. Equation (1.4) is easily seen to be equivalent to its **Schur complement** formulation
$$u = S^{-1}C^\top \left(CS^{-1}C^\top\right)^{-1} b. \tag{1.5}$$
Using sparse representations of the matrices C, S^{-1}, and $C^\top$, the Schur complement matrix $CS^{-1}C^\top$ can be efficiently inverted using the preconditioned conjugate gradient method (PCG). Of course, proper preconditioning of the Schur complement matrix will be necessary. As this method does not require explicit computation of the matrices, it can be used on very dense grids.
2. In the **pseudoinverse method** (1.4) is equivalent to yet another formulation and is given by
$$u = S^{-1/2}(CS^{-1/2})^+ b, \tag{1.6}$$
where $(CS^{-1/2})^+$ is the pseudoinverse of $CS^{-1/2}$. The pseudoinverse can be efficiently and stably calculated using a QR decomposition of the explicit matrix $CS^{-1/2}$. The advantage of using the pseudoinverse is that the condition number of the matrix $CS^{-1/2}$ is the square root of that of the full Schur complement matrix $CS^{-1}C^\top$. Ill-conditioning is therefore less of a problem and higher order norms can be used for regularization. In this way, greater accuracy can be obtained, but, because dense matrices are involved, the method is limited to coarser grids than are allowed in the Schur complement approach.

As the numerical experiments performed later will show, the SEEM method can be used to efficiently achieve spectral accuracy for general boundary conditions. Its implementation, as we shall see, is quite simple as it merely requires discretizing the BVP matrix $\mathcal{C}$ and the regularizing matrix $\mathcal{S}$ on a regular, rectangular grid.

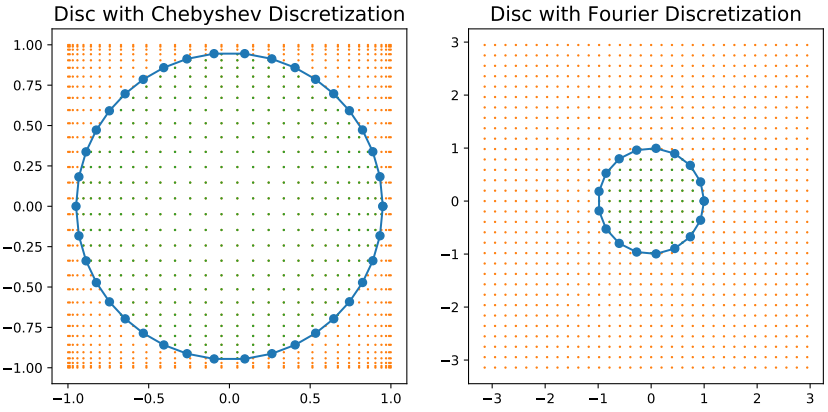

FIGURE 1. Contrasting the discretization of a disc on the Fourier and Chebyshev grids.

1.2. **Motivation for using Chebyshev polynomials.** In [1], SEEM was implemented for Fourier series on a periodic torus. Such an implementation was chosen because it allowed for the use of the FFT, which made the discretizations straightforward and the computations efficient. While the method was shown to be quite effective at solving BVPs, it suffered from two significant drawbacks. First, because the generated smooth extension of the BVP is periodic, only a small fraction of the periodic domain could be included in Ω. The rest was essentially needed as a buffer in order to allow the extension to smoothly morph into a periodic function. This wasted significant

computational resources, because the extension was solved on a far larger grid than was necessary for the solution of the BVP. Put differently, the ratio

$$\frac{|\Omega^m|}{|\mathbb{B}^m|}$$

was much smaller than the geometry of Ω actually required. A second problem was that a 2π-periodic extension of u had far larger optimization norm than u itself. The derivatives outside Ω were required to be large to force the extension to be periodic. This negatively affected the accuracy of the discretized solution. Table 1 below shows how, in the periodic setting, higher order Sobolev norms of the extension grow even if the smoother used is chosen to control these higher order norms, corresponding to the growth along the columns of the periodic section of the table. Clearly a norm of the extension is expected to grow if the optimization norm used is of lower order, explaining the growth along the rows of the table.

	Chebyshev Extension			Fourier Extension		
Smoother	$\|\nabla^2 u\|_{L_2}$	$\|\nabla^3 u\|_{L_2}$	$\|\nabla^4 u\|_{L_2}$	$\|\nabla^2 u\|_{L_2}$	$\|\nabla^3 u\|_{L_2}$	$\|\nabla^4 u\|_{L_2}$
S_2	16.0956	14.5451	319.2180	95.4606	172.2919	488.2472
S_3	16.0111	0.5137	2.3840	137.6518	177.6702	281.2872
S_4	16.0002	.0064	.0410	153.0669	184.4029	278.7028

TABLE 1. Various norms of the extensions obtained by the optimization procedure based on different smoothers.

Both of these problems can be remedied by using Chebyshev expansions instead of Fourier series. Instead of working on the periodic torus $[-\pi, \pi)^d$, one can work on the nonperiodic box $\mathbb{B} = [-1, 1]^d$. Using Chebyshev polynomials discretized on the Chebyshev roots' grid, functions can still be approximated with spectral accuracy on $\mathbb{B}$. At the same time, because the encompassing domain does not require the periodicity of the extension, far more of the domain $\mathbb{B}$ can be included in Ω. Additionally, for smooth, extendable u, the derivative norms of the extension will not be significantly larger than those of u itself. Thus, as demonstrated in the numerical experiments, equivalent accuracy can be achieved on far smaller grids, leading to faster computational times. As calculations on the Chebyshev grid can also be carried out using the FFT, the method retains the efficiency of SEEM based on Fourier series.

1.3. **The Chebyshev smoother.** The most significant difference between implementing SEEM on a Chebyshev grid versus on a Fourier grid lies in the choice of regularizing norm used in Equation (1.2). In the Fourier case, the norm

$$\|\cdot\|_{S_p} = \|(1-\Delta_\pi)^{p/2}\cdot\|_{L_2}$$

was used as a smoothing penalty. We note that this norm imposes H^p regularity on the solution, leading, as described in [1], to a convergence rate of order p for the error. In addition, the operator $(1-\Delta_\pi)^{-p/2}$ can be diagonalized by the Fourier transform and the operator $\mathcal{S}^{-1/2}$ takes the simple form

$$(1-\Delta_\pi)^{p/2}u = \mathcal{F}^{-1}\circ M\Big[(1+|\bullet|^2)^{-p/2}\Big]\circ \mathcal{F}u.$$

Here, $M[f]$ denotes multiplication by the function f, where, depending on the occurrence, the function f is either defined everywhere or on the discretization set. The symbol $\bullet$ is used in stead of the independent variable of the function f and therefore stands for either $x \in B$ or $k \in \mathbb{Z}^d$

depending on the context (continuous or discrete). The discretized operator $S^{-1/2}$ in (1.6) is very simply and efficiently computed using the FFT.

Motivated by this choice of smoother for the Fourier case and by the eigenvalue equation described below in Section A.5, we choose a smoothing norm for the Chebyshev grid in an analogous manner. As it is done in Section A.5, define the operator

$$\mathcal{D} := M\Big[\sqrt{1-\bullet^2}\Big] \circ \frac{\partial}{\partial x}.$$

By means of Equation (A.5.1) we find that

$$(1-\mathcal{D}^2)^{p/2} T_m = (1+m^2)^{p/2} T_m$$

holds true for the m-th Chebyshev polynomial T_m. Exploiting this, we define the norm

$$\|\cdot\|_{\mathcal{S}_p}^2 = \|(1-\sum_{i=1}^{d} \mathcal{D}_i^2)^{p/2} \cdot \|_{L_2}^2.$$

Clearly, away from the degeneracies at -1 and 1, this norm imposes H^p regularity on the function u. In addition, due to the eigenvalue equation, the operator $\mathcal{S}_p$ is diagonalized by the Chebyshev transform. In particular, if we denote the latter by $\mathfrak{C}$ and let $(k)_{k\in\mathbb{N}^d}$ be the (Chebyshev) frequency vector, we have that

$$\mathcal{S}_p^{-1/2} u = \mathfrak{C}^{-1} \circ M\Big[(1+|\bullet|^2)^{-p/2}\Big] \circ \mathfrak{C} u.$$

As described later in Section 2.2.3, this allows for simple and efficient numerical discretizations based on the discrete Chebyshev transform as described in A.3. The numerical experiments of Section 3 demonstrate that, as in the Fourier case, using the $\mathcal{S}_p$ norm leads to a p rate of convergence for the error.

Remark 1.1. *In the numerical experiments, the observed rate of convergence for the $\mathcal{S}_p$ smoother is somewhat faster than the expected rate p, which was observed in the Fourier case. We suspect that this may have to do with the higher density of points near the boundary of the domain Ω, due to the non-regular spacing of the Chebyshev grid.*

Remark 1.2. *An alternative choice for the norm, which gives a spectral rate of convergence, is*

$$\|\cdot\|_{S_{\exp}} = \|\mathfrak{C}^{-1} \circ M\Big[\exp^{\frac{|\bullet|}{2}}\Big] \circ \mathfrak{C} u\|_{L_2}.$$

As motivation, notice that the operator $\mathcal{S}_{\exp}^{-1/2} = \mathfrak{C}^{-1} \circ M\Big[\exp^{-\frac{|\bullet|}{2}}\Big] \circ \mathfrak{C} u$ is a pseudodifferential operator of heat type. As with $\mathcal{S}_p$, because the operator is diagonalized by the Chebyshev transform, $\mathcal{S}_{\exp}$ can be efficiently discretized and computed.

Remark 1.3. *While spectral discretizations based on Chebyshev polynomials were chosen for this paper, the proposed method can also be implemented with respect to any other spectral basis. It is enough to embed Ω into a larger domain $\mathbb{B}$ for which a full spectral resolution is known for some canonical self-adjoint and positive definite differential operator D with compact resolvent. If the operator admits natural discretizations $\mathbb{B}^m$ for the domain $\mathbb{B}$, $\{\psi_i\}_{i=1}^m$ for its (orthonormal) eigenfunctions, which are also orthonormal for the appropriate discrete quadrature rule, and satisfy*

$$\mathcal{D}_m \psi_i^m = \lambda_i^2 \psi_i^m,$$

for the eigenvalues $\lambda_\bullet^2$ of D, then a good smoothing norm is given by

$$\|\cdot\|_{S_p} = \|\mathfrak{C}_m^{-1} \circ M\Big[(1+\lambda_\bullet^2)^{p/2}\Big] \circ \mathfrak{C}_m \cdot \|_{L_2},$$

where $\mathfrak{C}_m$ is the discrete transformation which computes the coefficients of the (discrete and finite) eigenfunction expansion and $\lambda_\bullet^2$ is the corresponding vector of eigenvalues. The Fourier approach of [1] *clearly fits in this category.*

1.4. **Relationship with the radial basis collocation method.** Even though SEEM is an example of a so-called embedding method, it can nicely be understood by means of the the so-called radial basis collocation method (RBCM) framework. Surveys of the latter are found in [3] and [4]. In its simplest implementation, RBCM seeks the solution u to the problem at hand as a linear combination of (smooth) radial functions ϕ,

$$u = \sum_i c_i \phi(x - x_i),$$

where u is constrained to satisfy the equation with corresponding differential operator $\mathcal{C}$ at the points x_i. However, in RBCM's theoretical framework, the function ϕ is recognized to more properly represent a smoothing kernel applied to a δ distribution. The smoothing kernel corresponds to the $\mathcal{S}$ operator of our method, see [5]. In the most general form of the symmetric RBCM method, described in [3, Chapter 9], the solution is given by

$$u = K \star C^\top \big(C(K \star C^\top)\big)^{-1} b,$$

where $K = K(x, y)$ is any symmetric and positive definite smoothing kernel. It is also recognized that such kernels can often be represented as a multiplication operator in Fourier space, leading to both practical and analytical insights for the method. Clearly, the operator $\mathcal{S}$ used in our method fits in the general category of smoothing kernels. In fact, the smoothers $\mathcal{S}_p$ can be viewed as an analog of the Matérn kernels, used in RBCM. The Matérn functions (see [4, Chapter 4]) have Fourier transform $(1 + |\xi|^2)^{-p/2}$, while the kernel $\mathcal{S}_p$ has Chebyshev transform $(1 + |k|^2)^{-p/2}$. However, to the best of our knowledge, SEEM distinguishes itself from these other implementations of the RBCM in two main aspects, one practical and one philosophical.

1. In collocation methods, the evaluations of the smoothing operator are carried out analytically, and the resulting analytic functions are evaluated at the collocation points. The smoothing kernel itself, thought of as an operator on the encompassing domain $\mathbb{R}^d$, is not explicitly discretized. SEEM, in contrast, embeds the BVP into a finite domain with a straightforward, regular discretization, e.g. $[-1, 1]^d$ in the Chebyshev case. The chosen smoothing operators then have simple, discrete approximations on the encompassing domain. Skipping the discretization of the smoother, as it is done in collocation methods, definitely has some advantages. In particular, the collocation points can be placed arbitrarily close, and do not need to take the size of a grid discretization into account. Furthermore, there is no need to perform unnecessary computations on grid points outside of Ω. On the other hand, we believe that having an explicit discretization of the smoother has significant benefits of its own. First, it removes the need for detailed analytic computations of the radial basis functions and their derivatives, and replaces them with a natural multiplication operator in the frequency space of the discretized domain. In fact, for many smoothing operators, a simple explicit representation of the basis functions may not even exist. Second, it allows all computations to be done with a straightforward application of the FFT; thus, the evaluation of the matrix requires $O(n \log n)$ as opposed to $O(n^2)$. Finally, having an explicit representation of the smoother allows for the computation of the matrix $CS^{-1/2}$ as opposed to the full matrix $CS^{-1}C^\top$. As described in Section 1.1, the condition number of this matrix is the square root of that of the full matrix, which significantly slows the onset of numerical inaccuracy due to ill-conditioning.
2. Philosophically, as an embedding method, SEEM looks for the optimal way to embed the solution into an encompassing domain. Collocation methods, in contrast, look for the optimal set of basis functions to place at the collocation points. While in certain cases, the two formulations are equivalent, we believe that the optimization perspective offers many benefits. Firstly, it may often provide natural smoothing operators, suited to the encompassing domain, which might not be apparent at first glance. The norms

used in this paper certainly fit in this category. Secondly, the optimization viewpoint allows for the choice of more complex norms which may not be accessible in a collocation framework. For example, non-quadratic objective functionals could be considered. For non-regular problems, weighted norms would be a natural choice. Finally, we note that the optimization perspective is cited in [3, Chapter 6] as a mathematical justification for the usefulness of the RBCM.

2. Method

We now detail the implementation of SEEM on the Chebyshev grid. As described in (1.4), the method boils down to solving the saddle point problem

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

either using the Schur complement or the pseudoinverse method. The matrix $C = \begin{bmatrix} A & B \end{bmatrix}^\top$ represents a discretization of the BVP and S a discretization of the smoothing operator described in Section 1.3. A description is provided next of how to generate the proper discretizations of C and $S^{-1/2}$ and of the solution method used to deal with the resulting linear system.

2.1. **Discretization of the domain.** We recall that the BVP is posed in $\Omega \subseteq [-1,1]^d$. The domain $[-1,1]^d$ is discretized by a product set of the one-dimensional Chebyshev grid $\mathbb{C}^m$, described in Section A.1 and given by

$$\mathbb{B}^m = \left\{ (x^1, \ldots, x^d) \,\middle|\, x^i \in \mathbb{C}^m \text{ for } 1 \leq i \leq d \right\},$$

where

$$\mathbb{C}^m = \left\{ \cos\left(\pi \frac{2k+1}{2m} \right) \,\middle|\, 0 \leq k \leq m-1 \right\}.$$

The discretized interior of the domain is then given as $\Omega^m = \Omega \cap \mathbb{B}^m$, containing $N_\Omega^m := |\Omega^m|$ points. The boundary is discretized by choosing equally spaced points along the boundary Γ, yielding a set Γ^m containing $N_\Gamma^m := |\Gamma^m|$ points. We emphasize that these boundary points do not need to lie on the regular grid, but rather lie on the actual boundary Γ. In two dimensions, the discretization can be achieved by equally spacing points along an arclength parametrization of the boundary curve. In three dimensions, it is not possible to get a perfectly even distribution of points along a two-dimensional boundary surface. However, methods exist to obtain good approximations; see [6] for an example of such an algorithm.

A slightly better boundary discretization, particularly well adapted to the density of the Chebyshev grid, can be obtained as follows. If the boundary Γ is a hypersurface contained in $\mathbb{B}$ and parametrized by

$$\Big(\Gamma_1(\mathbf{z}), \ldots, \Gamma_d(\mathbf{z}) \Big), \ \mathbf{z} \in S^{d-1},$$

where S^{d-1} is the $d-1$-dimensional unit sphere, we can create an even distribution of points

$$\{\widetilde{y}^{\,i}\}_{i=1}^{N_m^\Gamma} = \left\{ (\widetilde{y}^{\,i}_1, \ldots, \widetilde{y}^{\,i}_d) \,\middle|\, i = 1, \ldots, N_m^\Gamma \right\}$$

along

$$\widetilde{\Gamma} := \Big(\arccos(\Gamma_1(\mathbf{z})), \ldots, \arccos(\Gamma_d(\mathbf{z})) \Big),$$

now a hypersurface of $[0,\pi)^d$. We note that applying the arccos function componentwise to the points of $\mathbb{B}^m$ leaves one with a regular grid on $[0,\pi)^d$; thus, setting

$$\Gamma^m = \left\{ (y_1^i, \ldots, y_d^i) \right\}_{i=1}^{N_m^\Gamma} = \left\{ (\cos(\widetilde{y}^{\,i}_1), \ldots, \cos(\widetilde{y}^{\,i}_d)) \right\}_{i=1}^{N_m^\Gamma}$$

produces a boundary discretization of Γ the density of which is proportional to the density of the Chebyshev points in $\mathbb{B}$. In the two dimensional numerical experiments, this method was used to discretize the boundary.

A choice is left as to the specific density of the points on the boundary. Increasing the number of discretization points increases the accuracy. However, if the boundary points are more closely spaced than the regular grid points, the regular grid will be unable to distinguish/resolve the boundary points and the matrix will become severely ill-conditioned (or even singular). In our numerical experiments, we have placed $\frac{m}{2}$ points per unit length in $\widetilde{\Gamma}$ for two dimensional problems. In three dimensions, we have used $4\pi m^2$ points per unit area on Γ. These densities seem to provide a good balance of accuracy versus condition number.

Remark 2.1. *The grid described above is frequently referred to as Chebyshev roots grid or Chebyshev points of the first kind. An alternative choice of grid could have been made with the Chebyshev extrema grid, also known as Chebyshev points of the second kind. Similar rates of convergence are observed with these points. However, in our numerical experiments, the roots grid appears to be more numerically stable. Furthermore, the regularizer S_p^{-1} described in Section 1.3 is only symmetric for the Chebyshev roots grid, which makes it more convenient for the use of iterative solvers. Henceforth, the Chebyshev grid will refer to the Chebyshev roots grid.*

2.2. **Discretization of the differential operators.** We recall that the boundary value problem we are solving is of the form

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

As described in the Introduction, in SEEM, the entire BVP, both the interior operator $\mathcal{A}$ and the boundary operator $\mathcal{B}$, are used in a combined system of constraints complementary to an optimization problem formulated on the encompassing domain $\mathbb{B}$. This is done by requiring that the discrete solution satisfy an equation corresponding to a discretization A of the operator $\mathcal{A}$ at the points Ω^m, the discretized interior differential equation, and to a discretization B of $\mathcal{B}$ at the points of Γ^m, the discretized boundary condition. We now describe exactly how to construct these discretizations in the setting of the Chebyshev grid.

2.2.1. *Construction of A.* The matrix A will be an $(N_\Omega^m \times m^d)$ matrix which uses values on the entire grid $\mathbb{B}^m$ to approximate the operator $\mathcal{A}$ at the points of Ω^m. For a general second order elliptic BVP, the interior operator is of the form

$$\mathcal{A}u = -a_{ij}u_{x_ix_j} + b_iu_{x_i} + cu.$$

To evaluate the derivatives, we use the discrete differentiation matrices D_i for the Chebyshev grid described in Section A.4. Here D_i corresponds to differentiation along the x_i direction. Similarly, D_{ij}^2 corresponds to taking two derivatives, one in the x_i and one in the x_j direction, respectively. Note that, in Section A.4, the differentiation matrices are given implicitly, as linear operators using the discrete cosine and sine transforms DCT and DST. This is done to speed the calculations up and to limit RAM usage, although explicit matrices can certainly be used as well. A restriction operator $R : \mathbb{R}^{\mathbb{B}^m} \to \mathbb{R}^{\Omega^m}$ is also needed which acts by restricting a grid function u to its values on Ω^m. Its transpose, $R^\top : \mathbb{R}^{\Omega^m} \to \mathbb{R}^{\mathbb{B}^m}$ acts as extension by 0 from Ω^m to $\mathbb{B}^m$. For a function u defined on $\mathbb{B}^m$, the operator A is then defined as

$$A(u) = -a_{ij}R\left(D_{ij}^2u\right) + b_iR\left(D_iu\right) + cu.$$

For a function v defined on Ω^m, $A^\top$, used in Section 2.3, is similarly given by

$$A^\top(v) = -D_{ij}^2R^\top\left(a_{ij}v\right) + D_iR^\top\left(b_iv\right) + R^\top\left(cv\right).$$

Remark 2.2. *While in this implementation, the matrix A enforces the (discrete) differential equation at the points of the Chebyshev grid, this is not strictly necessary. While the skeleton of the method is the Chebyshev grid, the original equations can be imposed at any point of $\Omega \subset \mathbb{B}$, just like it is done for the boundary conditions. Doing so simply requires the use of spectral interpolation beside that of spectral differentiation.*

2.2.2. *Construction of B.* The matrix B will be an $(N_\Gamma^m \times m^d)$ matrix which uses values on the entire grid $\mathbb{B}^m$ to approximate the operator $\mathcal{B}$ at the points of Γ^m. For the boundary conditions considered in this paper (Dirichlet, Neumann, or Robin), the boundary operator is of the form

$$\mathcal{B}u = a\gamma_\Gamma u + b\gamma_\Gamma(\nabla u)\cdot\nu_\Gamma$$

for some smooth functions a and b defined on the boundary Γ. Here, γ_Γ is the trace operator and ν_Γ is the unit outward pointing normal vector to Γ. Note that the choice $a \equiv 1, b \equiv 0$ corresponds to Dirichlet boundary conditions, while $a \equiv 0, b \equiv 1$ corresponds to Neumann boundary conditions. To evaluate the trace and the normal derivative on the boundary, we need to use the spectral interpolation operators, δ_y and $\delta_y \circ \nabla$, described in Section A.6. The vector δ_y is a spectral discretization of the δ distribution located at y, while ∇ is the discretized gradient. The matrix B is constructed by building each row independently. The i-th row of B corresponds to the evaluation of the boundary condition at the i-th point y_i of Γ^m. We therefore set

$$[B]_{i\bullet} = a(y_i)\delta_{y_i} + b(y_i)(\delta_{y_i}\circ\nabla)\cdot\nu_{y_i},$$

where ν_{y_i} is the normal vector to Γ at the point y_i.

2.2.3. *Construction of S^{-1}.* Recall from Section 1.3 that we utilize the smoothers

$$\mathcal{S}_p^{-1} = \mathfrak{C}^{-1}\circ M\Big[(1+|\bullet|^2)^{-p}\Big]\circ\mathfrak{C}.$$

The Chebyshev transform $\mathfrak{C}$ is defined in Section A.3 and $\bullet = (k)_{k\in\mathbb{N}^d}$ is the Chebyshev frequency vector on the d dimensional box $\mathbb{B}$. Each of these can be discretized simply and efficiently using the discrete Chebyshev transform $\mathfrak{C}_m$, also defined in Section A.3. When $\bullet = (k)_{k\in\{0,\dots,m-1\}^d}$ is the discrete Chebyshev frequency vector on the d dimensional grid $\mathbb{B}^m$, we define the discrete smoothers as

$$S_p^{-1} = \mathfrak{C}_m^{-1}\circ M\Big[(1+|\bullet|^2)^{-p}\Big]\circ\mathfrak{C}_m.$$

2.3. **Solution of the resulting system.** After delineating how to implement the matrices A, B, and S_p^{-1}, we turn to solving the linear system. As described in Section 1, the solution u can be obtained using either the Schur complement or the pseudoinverse methods. We now describe how each of these yields an efficient calculation of the solution.

2.3.1. *The Schur Complement Method.* In the Schur complement method, the solution is obtained by solving

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

As the matrix $CS_p^{-1}C^\top$ is symmetric, it can be inverted using the conjugate gradient method. When the single matrices are generated via conjugation with the FFT, their application can be performed in $O(n\log n)$ operations. Preconditioning is required, however, due to the large condition number of the relevant matrix. In order to obtain a preconditioner, we consider the full matrix $CS_p^{-1}C^\top$ as a block matrix, acting on the interior (Ω^m) and boundary (Γ^m) components of b separately.

$$CS_p^{-1}C^\top = \begin{pmatrix} AS_p^{-1}A^\top & AS_p^{-1}B^\top \\ BS_p^{-1}A^\top & BS_p^{-1}B^\top \end{pmatrix}.$$

The preconditioner P is then a block matrix

$$P = \begin{pmatrix} C_1^{-1} & 0 \\ 0 & C_2^{-1} \end{pmatrix},$$

where C_1^{-1} is an approximate inverse for $AS_p^{-1}A^\top$ and C_2^{-1} is one for $BS_p^{-1}B^\top$. Because the number of boundary points grows slowly relative to the number of interior points, the matrix $BS_p^{-1}B^\top$ remains small even on denser grids, and it can be computed explicitly and inverted directly; thus, C_2^{-1} can just be taken to be $(BS_p^{-1}B^\top)^{-1}$. The (continuous) operator (discretized by) $AS_p^{-1}A^\top$ is of order $2p-4$. Thus, a well conditioned operator of order 0 is obtained by using $(1-\mathcal{D}_\Omega^2)^{p-2}$ as a preconditioner. Here, $\mathcal{D}_\Omega^2$ is the operator $\sum_{i=1}^d \mathcal{D}_i^2$ restricted to Ω. Next observe that the operator D_Ω^2 can be thought of as an application of the Laplacian on a uniform grid after composition with the change of independent variables by arccos. In this way, a good discretization of D_Ω^2 is achieved by using a finite difference discretization of the Laplacian on the grid Ω^m, considering the points of Ω^m as if they were equally spaced. The numerical experiments performed in Section 3 confirm that the solution can be efficiently produced using the PCG method for smoothers S_p of order p up to 4, provided the preconditioning just described is applied.

2.3.2. *The Pseudoinverse Method.* In the pseudoinverse method, one needs to calculate $(CS_p^{-1/2})^+$. While iterative methods are an option, in our experiments the pseudoinverse is calculated directly by means of QR decomposition of the explicit matrix. Specifically, if $S_p^{-1/2}C^\top = QR$, then

$$(CS_p^{-1/2})^+ = Q(R^\top)^{-1}.$$

We have found that this delivers the most numerically stable method of calculating the pseudoinverse.

2.3.3. *Comparison of the Two Methods.* We briefly discuss the relative advantages of the two methods described above. As it turns out, the pseudoinverse method is better for small grids, which are a viable choice when studying very smooth problems. The Schur complement method is the better choice for the dense grids needed to study less smooth problems. These conclusions will also be supported by the numerical results of Section 3.

It is important to observe that the numerical computation of the matrix S_p^{-1} is constrained by numerical limitations. Indeed, the implementation of the operator as a multiplication in Fourier space involves multiplication by $(1+\bullet^2)^{-p}$, which can only be accurately rendered if $(1+\bullet^2)^{-p}$ is larger than numerical precision. This results in a limitation on the density of the grid that can be used with a given smoother S_p.

The Schur complement method, as an iterative method, has the advantage of requiring only $O(n\log n)$ computations. This makes it suitable for use on dense grids. However, because it requires the calculation of the matrix S_p^{-1}, it limits the maximal order of the smoother which can be employed. In practice, smoothers of order up to $p=5$ can be utilized. As a consequence, the rate of convergence of the numerical solution, which depends on the choice of p, will be limited. This method is therefore effective for solutions of lower global regularity, where a high rate of convergence is already limited by the lack of regularity of the solution itself and one needs to resort to a dense grid.

The pseudoinverse method, in contrast, only requires the computation of the matrix $S_p^{-1/2}$. Thus, the numerical limitations described above only kick in for smoothers of much higher order. In particular, smoothers of order $p \leq 10$ can be used. On the other hand, because the pseudoinverse is computed explicitly, dense grids are out of reach. This makes the pseudoinverse method a good choice for problems with smooth solutions where very high rates of convergence can be obtained.

The pseudoinverse method is used in Experiments 1 and 2, where the problems have an analytic solution, and the Schur complement method in Experiments 3 and 4 where the solutions are of lower regularity.

3. Numerical Experiments

We offer a series of numerical experiments to demonstrate the efficacy of the method. We will include a Dirichlet and Robin boundary value problem with analytic solution; this will allow us to demonstrate the high order convergence which our method can obtain. These problems will be solved using the pseudoinverse method. We will then solve a Dirichlet problems with solutions of global H^4 and H^6 regularity; here, the rate of convergence will necessarily be of lower order so the Schur complement method will be used. Finally, we will solve a parabolic PDE with analytic solution to demonstrate how the method can be used on a space-time fictitious domain as well. The reported rates of convergence for each experiment are computed as the average of all the rates of convergence observed going from one grid-size to the next.

In the following experiments, we will consider the following domains: a disc and a star-shaped domain.

$$\Omega_1 = \{(r,\theta) \mid r < .95\}.$$
$$\Omega_2 = \{(r,\theta) \mid r < .8(1 + .2\cos(\theta))\}.$$
$$.$$

The domains are shown in Figure 2.

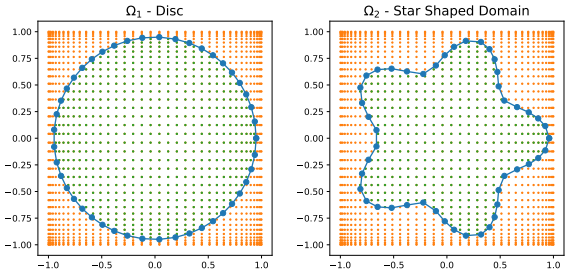

Figure 2. Two different domains discretized on the Chebyshev grid considered in the two-dimensional problems.

3.1. **A two-dimensional Dirichlet problem.**

We consider the Dirichlet problem

$$\begin{cases} -\Delta u = -6x - 6y & \text{in } \Omega, \\ u = x^3 + y^3 & \text{on } \Gamma. \end{cases} \tag{3.7}$$

on Ω, the disc of radius 0.95 shown in Figure 2. The exact solution is x^3+y^3. We solve the problem using the pseudoinverse method, with explicit matrices. The sizes of the different discretizations considered, together with the L_2 and L_∞ errors, are listed in Table 2. A graph of the L_2 and L_∞ errors is found in Figure 3. We note that because we are using explicit matrices and small grids, the CPU times are less than a second.

$\lvert B^m\rvert$	N_Ω^m	N_Γ^m	L_2 Error					
			S_2	S_4	S_6	S_8	S_{10}	S_{12}
$8^2 = 64$	24	5	2.81E-01	4.80E-01	9.38E-01	1.41E+00	1.68E+00	1.80E+00
$12^2 = 144$	60	8	9.66E-02	5.94E-02	2.74E-02	9.43E-03	2.68E-03	6.23E-04
$16^2 = 256$	104	10	4.30E-02	2.60E-02	1.15E-02	3.86E-03	1.09E-03	2.54E-04
$20^2 = 400$	164	12	1.87E-02	3.85E-03	6.54E-04	7.48E-05	7.61E-06	6.92E-07
$24^2 = 576$	240	15	7.42E-03	5.97E-04	7.44E-05	4.54E-06	2.46E-07	1.36E-08
$28^2 = 784$	324	17	5.38E-03	3.94E-04	4.85E-05	2.83E-06	1.33E-07	5.73E-09
$32^2 = 1024$	408	19	3.78E-03	1.41E-04	1.09E-05	4.32E-07	1.46E-08	4.55E-10
$36^2 = 1296$	520	22	2.25E-03	3.75E-05	1.63E-06	6.22E-08	1.12E-09	2.57E-11
Rate of Convergence:			3.21	6.29	8.82	11.26	14.05	16.60

TABLE 2. L_2 errors for Equation 3.7.

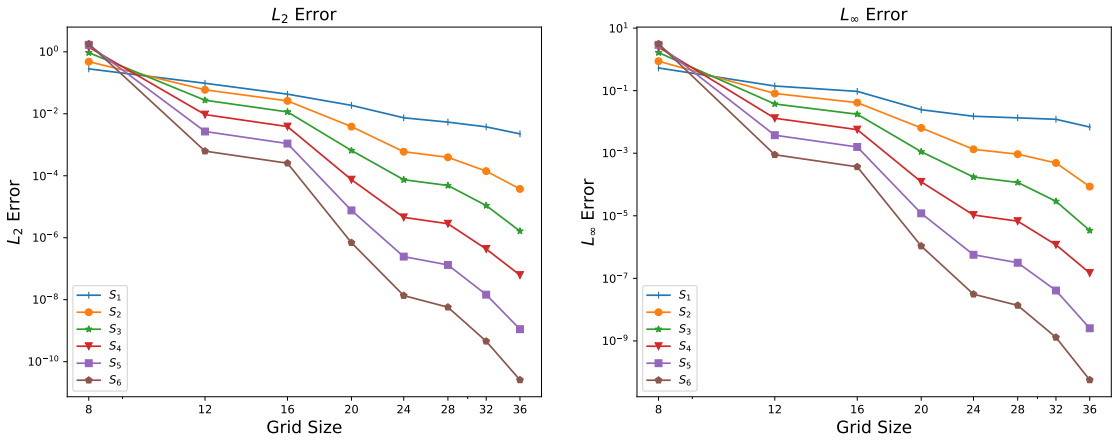

FIGURE 3. Convergence of the L_2 error and L_∞ errors for Equation (3.7).

3.2. **A two-dimensional Robin problem.** Next consider the star shaped domain

$$\Omega = \{(r,\theta) \mid r < .8(1 + .2(\cos(5\theta))\},$$

shown in Figure 2. Consider the Robin boundary value problem

$$\begin{cases} -\Delta u = 0 & \text{in } \Omega, \\ u + \frac{\partial u}{\partial \nu} = (x^2 - y^2) + \langle 2x, -2y\rangle \cdot \nu & \text{on } \Gamma, \end{cases} \tag{3.8}$$

where ν is the outward pointing unit normal vector to Γ. In this case the exact solution is given by $u(x,y) = x^2 - y^2$. The numerical results are summarized in Table 3 and Figure 4. We include this example to demonstrate that the method works across different boundary value problems and domain shapes.

$\lvert B^m\rvert$	N_Ω^m	N_Γ^m	L_2 Error					
			S_2	S_4	S_6	S_8	S_{10}	S_{12}
$6^2 = 36$	10	4	4.74E-01	1.02E-01	3.41E-02	1.14E-02	1.62E-02	4.43E-02
$10^2 = 100$	28	7	4.56E-01	6.46E-02	8.37E-03	1.04E-03	9.84E-05	8.23E-06
$14^2 = 196$	50	9	3.01E-01	4.40E-03	4.97E-04	4.22E-05	2.96E-06	1.78E-07
$18^2 = 324$	86	12	2.56E-01	2.75E-03	8.38E-05	2.57E-06	6.55E-08	1.44E-09
$22^2 = 484$	134	14	1.32E-01	5.03E-04	2.27E-06	8.94E-08	1.58E-09	2.41E-11
$28^2 = 784$	204	18	2.05E-01	4.14E-04	2.22E-06	1.21E-08	9.66E-11	6.13E-12
$34^2 = 1156$	308	21	8.06E-02	5.15E-05	5.29E-07	2.84E-09	1.62E-11	3.53E-12
$40^2 = 1600$	424	25	9.24E-02	2.90E-05	4.70E-08	3.21E-11	1.17E-12	1.86E-11
Rate of Convergence:			0.86	4.30	7.11	10.38	12.31	11.38

TABLE 3. L_2 errors for Equation 3.8.

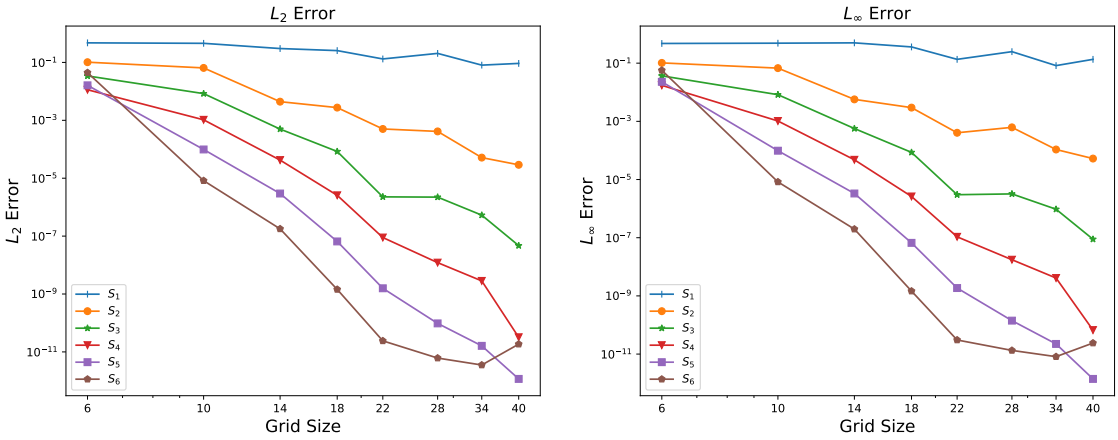

FIGURE 4. Convergence of the L_2 error and L_∞ errors for Equation (3.8).

The above experiments show that very high order methods come at the expense of very high condition numbers of the matrices in the discretized system and push against the intrinsic machine precision limitations in the numerical representation of the regularizers. These facts explain the stagnation observed for higher regularizers in the above examples. In Figure 5, we provide a plot of the condition numbers for these examples.

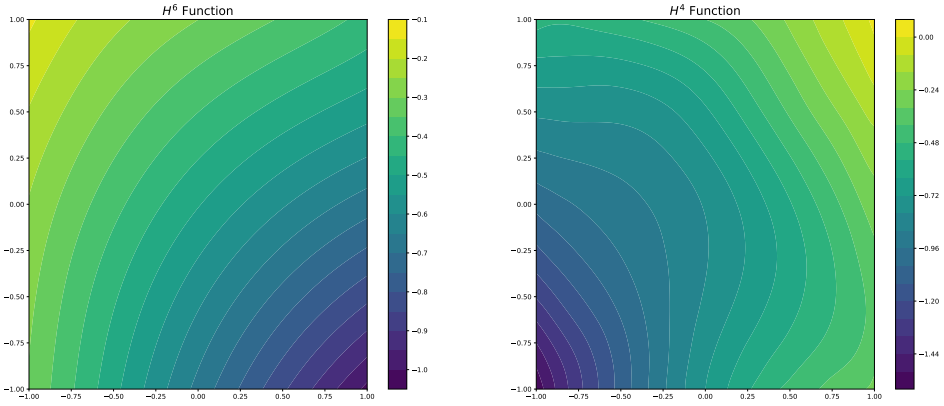

FIGURE 6. Functions of lower global regularity used in Sections 3.3 and 3.4.

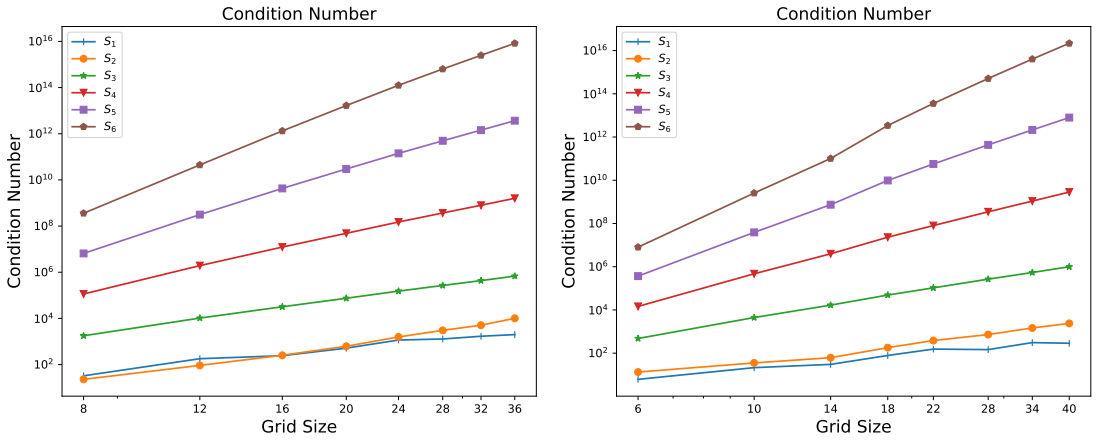

FIGURE 5. Condition numbers for experiments (3.7) and (3.8).

3.3. **A problem with H^6 solution.** The problem is generated by considering a Chebyshev series with random coefficients which satisfy a decay rate $|a_k| \leq (1+|k|^2)^{-3}$; the function is shown in Figure 6. We solve a Dirichlet problem where the chosen function is restricted to the disc of radius .95. Because the solution is globally H^6, the solution cannot converge at a rate larger than 4, i.e. the error cannot decay faster than $O(m^{-4})$. Thus, in this case, the solution must be computed on denser grids using the Schur complement method. We show the rates of convergence below in Figure 7 and Table 4. We also record the number of PCG iterations to convergence and CPU times in Table 5.

$\lvert B^m\rvert$	N_Ω^m	N_Γ^m	L_2 Error		
			S_2	$S3$	S_4
$16^2 = 256$	104	10	1.32E-03	2.44E-04	3.14E-04
$32^2 = 1024$	408	19	2.81E-04	1.39E-05	6.52E-06
$64^2 = 4096$	1660	38	3.37E-05	9.81E-07	1.47E-07
$128^2 = 16384$	6656	75	4.93E-06	5.59E-08	3.25E-09
$256^2 = 65536$	26656	150	8.96E-07	4.49E-09	1.61E-09
Rate of Convergence:			2.63	3.93	4.39

TABLE 4. Convergence of L_2 error for H^6 function.

$\lvert B^m\rvert$	PCG Iterations			CPU Times		
	S_2	S_3	S_4	S_2	S_3	S_4
$16^2 = 256$	36	26	37	0.05	0.03	0.05
$32^2 = 1024$	60	41	60	0.1	0.06	0.11
$64^2 = 4096$	94	67	106	0.37	0.2	0.33
$128^2 = 16384$	122	97	173	1.11	0.94	1.64
$256^2 = 65536$	149	118	328	7.26	5.41	14.9

TABLE 5. PCG Iterations and CPU times for H^6 function.

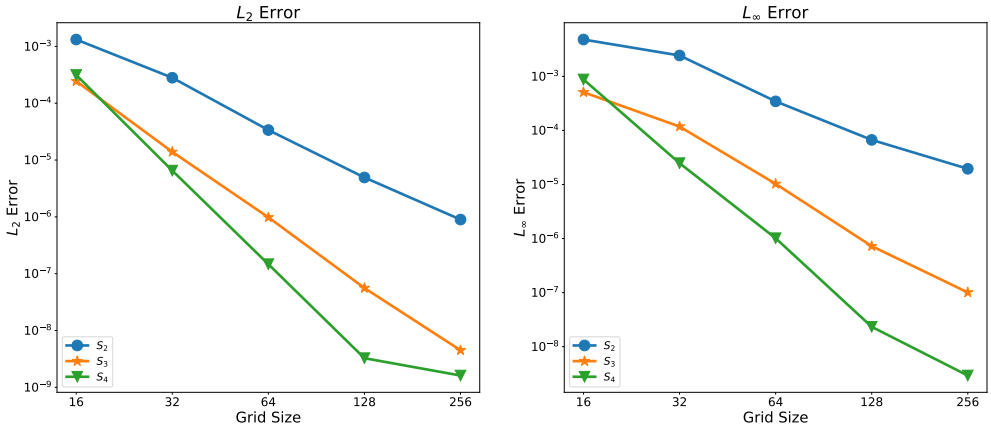

FIGURE 7. Convergence of the L_2 and L_∞ errors for the H^6 solution.

3.4. **A problem with H^4 solution.** We use a Chebyshev series with random coefficients which satisfy a decay rate $|a_k| \leq (1+|k|^2)^{-2}$. Again, we solve a Dirichlet problem on the disc of radius .95. Because the solution is globally H^4, the solution cannot converge at a rate larger than 2. Thus, in this case, the solution must be computed on denser grids using the Schur complement method. We show the rates of convergence below in Figure 8 and Table 6. We note that all three smoothers have the same rate of convergence due to the limited regularity of the solution.

$\lvert B^m\rvert$	N_Ω^m	N_Γ^m	L_2 Error		
			S_2	$S3$	S_4
$16^2 = 256$	104	10	4.40E-03	2.38E-03	2.03E-03
$32^2 = 1024$	408	19	6.05E-04	4.34E-04	4.46E-04
$64^2 = 4096$	1660	38	1.14E-04	1.04E-04	1.02E-04
$128^2 = 16384$	6656	75	1.71E-05	1.43E-05	1.43E-05
$256^2 = 65536$	26656	150	4.59E-06	4.26E-06	3.79E-06
Rate of Convergence:			2.48	2.28	2.27

TABLE 6. Convergence of L_2 error for H^4 function.

$\lvert B^m\rvert$	PCG Iterations			CPU Times		
	S_2	S_3	S_4	S_2	S_3	S_4
$16^2 = 256$	35	28	44	0.05	0.04	0.05
$32^2 = 1024$	60	44	68	0.09	0.07	0.12
$64^2 = 4096$	90	69	125	0.24	0.24	0.37
$128^2 = 16384$	121	100	205	1.04	0.93	1.91
$256^2 = 65536$	144	125	436	6.4	5.01	19.5

TABLE 7. PCG Iterations and CPU times for H^4 function.

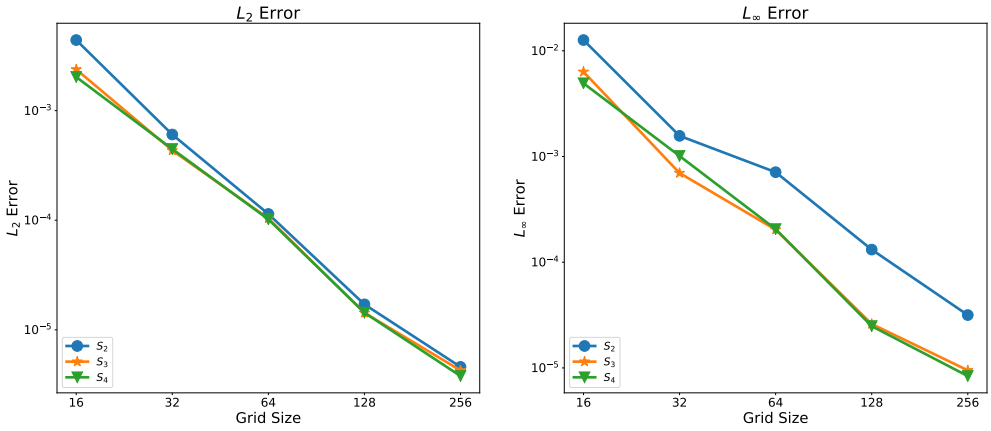

FIGURE 8. Convergence of the L_2 and L_∞ errors for the H^4 solution.

3.5. **A parabolic problem.** Finally we describe a procedure to solve a time dependent problem using SEEM. This example is considered to show both that SEEM is effective for a wide variety of PDEs and to show that it is effective in three dimensions. Since we are calculating the solution across a spacetime cylinder (and not marching in time), the problem is effectively a three dimensional problem. We consider the parabolic cylinder $\Omega \times [0,2]$, where Ω is the star-shaped domain shown in Figure 2. Define j_0 to be the 0-th Bessel function of the first kind. Letting r denote the Euclidean distance from 0, consider the radial function

$$u(t,r) = e^{-t} j_0(r) - e^{-\frac{t}{4}} j_0\Big(\frac{r}{2}\Big).$$

The function u then satisfies the parabolic BVP

$$\begin{cases} u_t - \Delta u = 0 & \text{in } (0,2] \times \Omega, \\ u(0,\cdot) = j_0 - j_0(\frac{\cdot}{2}) & \text{in } \Omega, \\ u(t,\cdot) = e^{-t} j_0 - e^{-\frac{t}{4}} j_0(\frac{\cdot}{2}) & \text{on } \Gamma \text{ for } t \in (0,2]. \end{cases} \tag{3.9}$$

To discretize the domain, we use the Chebyshev grid $\mathbb{B}^m$ described in Section A.1. As for the time interval $[0,2]$, we use a (shifted) Chebyshev extrema grid,

$$\mathbb{B}_E^n = \{t_j\}_{j=0}^n, \text{ where } t_j = -\cos\left(\frac{\pi j}{n}\right) + 1.$$

In this section $n = 10$ is chosen in all of the experiments. The full discretization of the parabolic cylinder $[-1,1]^2 \times [0,2]$ is then given by $\mathbb{B}^m \times \mathbb{B}_E^n$. The choice to use the extrema grid rather than the standard Chebyshev (roots) grid in the time variable was made because imposing the boundary condition at $t = 0$ is slightly more straightforward, since the boundary point $t = 0$ lies on the grid. For a description of how to construct the time differentiation matrix, D_t, we refer to [7].

With the discretization $\mathbb{B}^m \times \mathbb{B}_E^n$, the interior of the parabolic cylinder is given by $\Omega^m \times \widetilde{\mathbb{B}}_E^n$, where

$$\widetilde{\mathbb{B}}_E^n = \big\{ t_i \in \mathbb{B}_E^n \,\big|\, t_i > 0 \big\}.$$

The discretized "bottom" boundary of the cylinder is given by $\Omega^m \times \{0\}$, whereas the discretization of the lateral boundary $\Gamma \times (0,2]$ is simply given by $\Gamma^m \times \widetilde{\mathbb{B}}_E^n$. Letting R_{K^m} denote the evaluation

operator on the discrete set K^m, we can define the matrices

$$\begin{aligned} A &= R_{\Omega^m\times\widetilde{\mathbb{B}}^n_E}\circ(D_t - D^2_{x_1} - D^2_{x_2}),\\ B_1 &= R_{\Omega^m\times\{0\}},\\ B_2 &= R_{\Gamma^m\times\widetilde{\mathbb{B}}^n_E}. \end{aligned}$$

Notice that evaluation of a function on the above sets simply amounts to their restriction to the sets since $\Omega^m\times\widetilde{\mathbb{B}}^n_E$ and $\Omega^m\times\{0\}$ are sets of regular grid points. However, because Γ^m does not contain regular grid points in general, the evaluation matrix $R_{\Gamma^m\times\widetilde{\mathbb{B}}^n_E}$ will require the use of the interpolation operators described in Sections 2.2 and A.6. If b_1 and b_2 represent the evaluations of the function $e^{-t}j_0 - e^{-\frac{t}{4}}j_0\Big(\frac{\cdot}{2}\Big)$ at the points of $\Omega^m\times\{0\}$ and $\Gamma^m\times\widetilde{\mathbb{B}}^n_E$, respectively, the BVP is fully discretized by the matrix equation $Cu = b$ where

$$C = \begin{bmatrix} A & B_1 & B_2 \end{bmatrix}^\top \text{ and } b = \begin{bmatrix} 0 & b_1 & b_2 \end{bmatrix}^\top,$$

and, by an abuse of notation, $u = u^m$, is the discrete unknown. As for the elliptic SEEM, the problem is converted to a constrained optimization problem

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

where $\|\cdot\|_S$ is a smoothing norm. In the parabolic case, $\|\cdot\|_S$ needs to be a space-time norm over $\mathbb{B}^m\times\mathbb{B}^n_E$. We recall from Section 1.3 and A.5 the operator

$$(\mathcal{D}^m)^2 = \mathfrak{C}^{-1}_m\circ M\Big[|\bullet|^2\Big]\circ\mathfrak{C}_m.$$

The operators $\mathcal{D}^m_i$ and $\mathcal{D}^m_t$ represent applying the operator in the x_i and t directions, respectively. Motivated by our choice of smoothing norm used in the elliptic case and described in Section 1.3, the norm given by

$$\|u\|_{S_p} = \Big\|\Big(1 - \sum_{i=1}^{2}(\mathcal{D}^m_i)^2 - (\mathcal{D}^m_t)^2\Big)^{p/2}(u)\Big\|,$$

is used in order to enforce space-time regularity of the numerical solution. As with the norms described in the elliptic case, this norm has the benefit of simple implementation using the discrete Chebyshev transform. We remark that while this norm is clearly effective, as demonstrated by our numerical experiments, it is not natural from the point of view of parabolic PDEs and may not be the optimal one to use; we are continuing to investigate the best choice of smoother in the parabolic case.

As in the elliptic case, the problem then reduces to finding

$$u = S_p^{-1/2}(CS_p^{-1/2})^+ f.$$

The solution is obtained using a QR decomposition of $S_p^{-1/2}C^\top$, as described in Section 2.3. The numerical results for the initial boundary value problem are summarized in Table 8 and Figure 9.

$\left\|\mathbb{B}^m \times \mathbb{B}_E^n\right\|$	$\left\|\Omega \times \tilde{\mathbb{B}}_E^n\right\|$	$\left\|\Omega^n \times \{0\}\right\|$	$\left\|\Gamma^n \times \tilde{\mathbb{B}}_E^n\right\|$	L_2 Error				
				S_2	S_4	S_6	S_8	S_{10}
$10^2 \times 11 = 1100$	280	28	130	5.30E-04	1.77E-05	3.69E-06	6.91E-07	2.48E-07
$12^2 \times 11 = 1584$	400	40	150	6.65E-04	1.84E-05	3.86E-06	7.72E-07	4.92E-07
$14^2 \times 11 = 2156$	500	50	180	3.63E-04	5.56E-06	6.95E-07	5.36E-08	2.51E-08
$16^2 \times 11 = 2816$	720	72	200	3.21E-04	3.17E-06	2.85E-07	1.91E-08	4.50E-09
$18^2 \times 11 = 3564$	860	86	230	1.70E-04	1.39E-06	9.84E-08	3.53E-09	2.23E-09
$20^2 \times 11 = 4400$	1060	106	250	2.23E-04	1.81E-06	1.09E-07	3.34E-09	1.12E-09
$22^2 \times 11 = 5324$	1340	134	280	1.46E-04	7.44E-07	3.61E-08	9.19E-10	2.71E-10
$24^2 \times 11 = 6336$	1540	154	300	1.02E-04	3.78E-07	1.20E-08	4.36E-10	1.23E-10
Rate of Convergence:				1.88	4.39	6.54	8.42	8.69

TABLE 8. Grid sizes and L_2 errors for Equation (3.9).

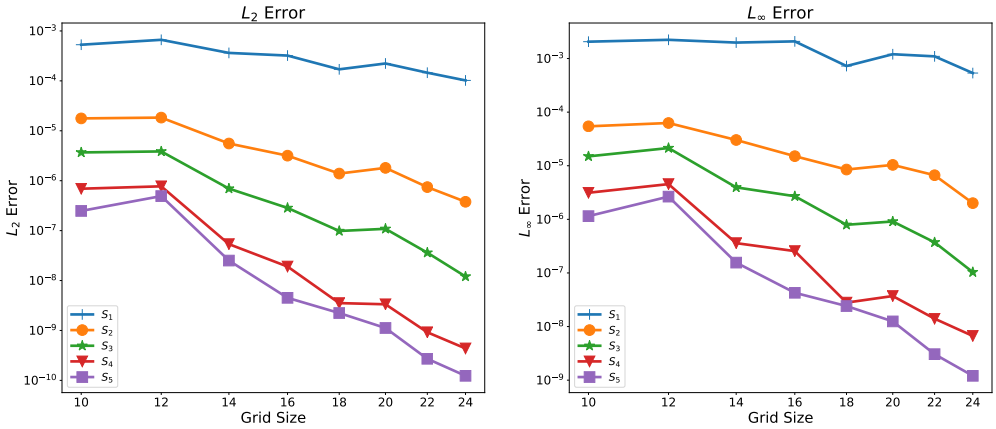

FIGURE 9. Convergence of the L_2 and L_∞ errors for different values of p when solving Equation (3.9).

REFERENCES

[1] D Agress and P Guidotti. The smooth extension embedding method. *SIAM Journal of Scientific Computing*, 43(1):A446–A471, 2021.

[2] R Glowinski, TW Pan, and J Periaux. A fictitious domain method for Dirichlet problem and applications. *Computer Methods in Applied Mechanics and Engineering*, 111(3-4):283–303, 1994.

[3] GE Fasshauer. Meshfree methods. *Handbook of theoretical and computational nanotechnology*, 27:33–97, 2005.

[4] GE Fasshauer. *Meshfree approximation methods with MATLAB*, volume 6. World Scientific, 2007.

[5] R Schaback. Kernel-based meshless methods. *Lecture Notes for Taught Course in Approximation Theory. Georg-August-Universität Göttingen*, 2007.

[6] Richard Palais, Bob Palais, and Hermann Karcher. Pointclouds: Distributing points uniformly on a surface. *arXiv preprint arXiv:1611.04690*, 2016.

[7] LN Trefethen. *Spectral methods in MATLAB*, volume 10. Siam, 2000.
[8] JP Berrut and LN Trefethen. Barycentric Lagrange interpolation. *SIAM review*, 46(3):501–517, 2004.

Appendix A. The Chebyshev Polynomials

Setting $\mathbb{B} = [-1, 1]$, the Chebyshev polynomials of the first kind are given by

$$T_m(x) = \cos\big(m \arccos(x)\big),\ x \in \mathbb{B},\ m \in \mathbb{N}.$$

We briefly describe some relevant properties of the Chebyshev polynomials used in the body of the paper. We refer to [7] for a more detailed discussion.

A.1. **The Chebyshev roots.** For fixed $m \in \mathbb{N}$, the m roots of $T_m(x)$ are given by

$$x_k = \cos\Big(\pi \frac{2k-1}{2m}\Big),\ 0 \leq k \leq m-1.$$

The Chebyshev grid comprising all roots of T_m, given by $\{x_k \,|\, k = 0, \dots, m-1\}$, is well adapted to the spectral calculation of derivatives. In higher dimensions, a tensor product of one-dimensional Chebyshev grids can be used. Throughout the body of the paper, $\mathbb{B}^m$ has been used to denote the Chebyshev grid of the appropriate dimension.

A.2. **Orthogonality relations.** The sequence $(T_m)_{m\in\mathbb{N}}$ forms an orthogonal basis for $L_2(\mathbb{B})$ with respect to the measure $\frac{dx}{\sqrt{1-x^2}}$. More specifically, for $i, j \in \mathbb{N}$,

$$\int_{-1}^{1} T_i(x)T_j(x)\frac{dx}{\sqrt{1-x^2}} = \begin{cases} 0, & \text{if } i \neq j, \\ \pi, & \text{if } i = j = 0, \\ \pi/2, & \text{if } i = j \neq 0. \end{cases}$$

The Chebyshev functions restricted to $\mathbb{B}^m$ also satisfy a discrete orthogonality relation. Indeed, for $0 \leq i, j \leq m-1$, one has that

$$\sum_{k=0}^{m-1} T_i(x_k)T_j(x_k) = \begin{cases} 0, & \text{if } i \neq j, \\ m, & \text{if } i = j = 0, \\ \frac{m}{2}, & \text{if } i = j \neq 0. \end{cases}$$

A.3. **The Chebyshev transform.** Because $(T_m)_{m\in\mathbb{N}}$ forms an orthogonal basis of $\mathrm{L}_2(\mathbb{B})$, any function $u \in \mathrm{L}_2(\mathbb{B})$ can be developed in a "Chebyshev series". We set

$$c_k = \frac{p_k}{\pi}\int_{-1}^{1} u(x)T_k(x)\frac{dx}{\sqrt{1-x^2}},\ \text{where } p_k = \begin{cases} 1, & \text{if } k = 0, \\ 2, & \text{if } k \neq 0. \end{cases}$$

so that

$$u(x) = \sum_{m=0}^{\infty} c_k T_k(x).$$

It is also possible to define the Chebyshev transform, denoted by $\mathfrak{C}$, which maps a function to the sequence of its Chebyshev coefficients.

$$\mathfrak{C}(u) = (c_k)_{k\in\mathbb{N}}.$$

The discrete orthogonality relation also yields a discrete version of a Chebyshev expansion. Given $u : \mathbb{B}^m \to \mathbb{R}$, let

$$c_k = \frac{p_k}{m}\sum_{i=0}^{m-1} u_i T_k(x_i)\ \text{for } p_k = \begin{cases} 1, & \text{if } k = 0, \\ 2, & \text{if } k \neq 0. \end{cases}$$

Then u can be written as a discrete Chebyshev series

$$u_\bullet = \sum_{k=0}^{m-1} c_k T_k(x_\bullet).$$

As in the continuous case, we can define the discrete Chebyshev transform $\mathfrak{C}_m$, which maps u to its discrete Chebyshev series, i.e., we set

$$\mathfrak{C}_m(u) = (c_k)_{k=0,\dots,m-1} =: c.$$

Both $\mathfrak{C}_m$ and its inverse $\mathfrak{C}_m^{-1}$ can be implemented efficiently using an FFT algorithm, in the form of the discrete cosine transform, more specifically,

$$\mathfrak{C}_{\mathfrak{m}}(u)_k = a_k \mathsf{DCT}(u)_k \text{ for } a_k = \begin{cases} \frac{1}{2m}, & \text{if } k = 0, \\ \frac{1}{m}, & \text{if } k > 0, \end{cases}$$

and

$$\mathfrak{C}_{\mathfrak{m}}{}^{-1}(c) = \mathsf{IDCT}(\widetilde{c}) \text{ for } \widetilde{c}_k = \begin{cases} c_k, & \text{if } k = 0, \\ c_k/2, & \text{if } k > 0. \end{cases}$$

In dimension larger than one, $\mathfrak{C}$ and $\mathfrak{C}_m$ will denote the continuous and discrete one-variable Chebyshev transforms applied successively in each direction. Numerically, this can be accomplished with the use of DCTN, where the factors a_k and b_k are raised to the power of the dimension.

A.4. **Derivative formulæ.** Discrete derivatives can be efficiently evaluated on the Chebyshev grid using the DCT and DST. We denote by $\bullet$ the discrete frequency vector $(k)_{k\in\{0,\dots,m-1\}}$ or the continuous variable x depending on the context, and let $M[f]$ represent multiplication by the discrete function f. We also define a shifting operator R with

$$\mathsf{R}_{ij} = \delta_{i+1,j},$$

so that R is the matrix with ones on the superdiagonal. Then, given a function $u = (u_i)_{i\in\{0,\dots,m-1\}}$ defined on the Chebyshev grid $\mathbb{B}^m$, a spectrally accurate discrete derivative Du can be calculated using the matrix given by

$$D = M\Big[\frac{1}{\sqrt{1-\bullet^2}}\Big] \circ \mathsf{IDST} \circ \mathsf{R} \circ M\Big[\frac{\bullet}{2m}\Big] \circ \mathsf{DCT}.$$

Similarly, we can compute

$$D^2 = M\Big[\frac{-1}{1-\bullet^2}\Big] \circ \mathsf{IDCT} \circ M\Big[-\frac{\bullet^2}{2m}\Big] \circ \mathsf{DCT} + M\Big[\frac{\bullet}{(1-\bullet^2)^{3/2}}\Big] \circ \mathsf{IDST} \circ \mathsf{R} \circ M\Big[-\frac{\bullet}{2m}\Big] \circ \mathsf{DCT}.$$

Of course, a corresponding operator can be formed in higher dimensions, where the DCT, DST as well as the frequency vector $\bullet$ are taken along the desired direction of differentiation. In the body of the paper, the derivative operator in the x_i direction is denoted as D_i.

A.5. **Eigenvalue equation.** Setting $\mathcal{D} = M\big[\sqrt{1-x_\bullet^2}\big] \circ \frac{\partial}{\partial x}$, the Chebyshev polynomials satisfy the eigenvalue equation

$$-\mathcal{D}^2 T_m = m^2 T_m. \tag{A.5.1}$$

Similarly, given the Chebyshev grid $\mathbb{B}^m$, the discrete Chebyshev functions $T_j(x_\bullet)$ satisfy a discrete eigenvalue equation. Defining $\mathcal{D}_m = M\big[\sqrt{1-x_\bullet^2}\big] \circ D$, $T_j(x_\bullet)$ satisfies

$$-\mathcal{D}_m^2 T_j(x_\bullet) = j^2 T_j(x_\bullet),\ j \in \{0,\dots,m-1\}.$$

This implies that

$$(1-\mathcal{D}_m^2)^{-p/2} = \mathfrak{C}_m^{-1} \circ M\Big[(1+|\bullet|^2)^{-p/2}\Big] \circ \mathfrak{C}_m.$$

A.6. **Interpolation operators.** Functions defined on the Chebyshev grid can be interpolated at arbitrary points in $\mathbb{B}$. Such interpolation can be stably computed by means of the barycentric interpolation formulæ described in [8]. Define first the vector w by

$$w_k = (-1)^k \sin\Big(\frac{2k-1}{2m}\Big),\ 0 \le k \le m-1.$$

If $y \in \mathbb{B}$ and $(x_i)_{i\in\{0,\dots,m-1\}}$ is the vector of points in $\mathbb{B}^m$, a spectrally accurate interpolation of a discrete function u defined on the Chebyshev grid $\mathbb{B}^m$ can be obtained by

$$u(y) = \delta_y \cdot u \text{ where } (\delta_y)_i = \frac{1}{\sum_{k=0}^{m-1} \frac{w_k}{y-x_k}} \frac{w_i}{y-x_i}.$$

To calculate the interpolation of the first derivative, which will be used in the Neumann problem, we use the derivative of the above formula,

$$Du(y) = \delta_y \circ D \text{ where } (\delta_y \circ D)_i = -\frac{1}{\sum_{k=0}^{m-1} \frac{w_k}{y-x_k}} \frac{w_i}{(y-x_i)^2} + \frac{\sum_{k=0}^{m-1} \frac{w_k}{(y-x_k)^2}}{\Big(\sum_{k=0}^{m-1} \frac{w_k}{y-x_k}\Big)^2} \frac{w_i}{y-x_i}.$$

To interpolate in several dimensions, we use a tensor product of the given interpolants, which are denoted by δ_y and $(\delta_y \cdot \nabla)$. To calculate a directional derivative of the grid function u in the direction ν at the point y, we use $(\delta_y \circ \nabla u) \cdot \nu_y$.

University of California, Irvine, Department of Mathematics, 340 Rowland Hall, Irvine, CA 92697-3875, USA

Email address: dagress@uci.edu and gpatrick@math.uci.edu and dyan6@uci.edu