FAST POISSON SOLVERS FOR SPECTRAL METHODS

DANIEL FORTUNATO* AND ALEX TOWNSEND†

Abstract. Poisson's equation is the canonical elliptic partial differential equation. While there exist fast Poisson solvers for finite difference and finite element methods, fast Poisson solvers for spectral methods have remained elusive. Here, we derive spectral methods for solving Poisson's equation on a square, cylinder, solid sphere, and cube that have an optimal complexity (up to polylogarithmic terms) in terms of the degrees of freedom required to represent the solution. Whereas FFT-based fast Poisson solvers exploit structured eigenvectors of finite difference matrices, our solver exploits a separated spectra property that holds for our spectral discretizations. Without parallelization, we can solve Poisson's equation on a square with 100 million degrees of freedom in under two minutes on a standard laptop.

Key words. fast Poisson solvers, spectral methods, alternating direction implicit method, ultraspherical polynomials

AMS subject classifications. 65N35, 35J05, 33C45

1. Introduction. Consider Poisson's equation on a square with zero homogeneous Dirichlet conditions:

$$u_{xx} + u_{yy} = f, \quad (x, y) \in [-1, 1]^2, \qquad u(\pm 1, \cdot) = u(\cdot, \pm 1) = 0, \tag{1.1}$$

where f is a known continuous function and u is the desired solution. When (1.1) is discretized by the finite difference (FD) method with a five-point stencil on an $(n+1) \times (n+1)$ equispaced grid, there is a FFT-based algorithm that computes the values of the solution in an optimal[1] $\mathcal{O}(n^2 \log n)$ operations [16]. Many fast Poisson solvers have been developed for low-order approximation schemes using uniform and nonuniform discretizations based on cyclic reduction [8], the fast multipole method [21], and multigrid [13]. This work began with a question:

Is there an optimal complexity spectral method for (1.1)?

We find that the answer is yes. In section 3, we describe a practical $\mathcal{O}(n^2(\log n)^2)$ algorithm based on the alternating direction implicit (ADI) method. We go on to derive optimal complexity spectral methods for Poisson's equation with homogeneous Dirichlet conditions for the cylinder and solid sphere in section 4 and for the cube in section 5. In section 6, we extend our approach to Poisson's equation with Neumann and Robin boundary conditions. Optimal complexity spectral methods already exist for Poisson's equation on the disk [37, 38] and surface of the sphere [32]. This paper can be seen as an extension of that work.

Our first idea for deriving an optimal complexity spectral method for (1.1) was to extend a fast Poisson solver from the FD literature. The FD discretization of (1.1) with a five-point stencil on an $(n+1) \times (n+1)$ equispaced grid can be written as the following Sylvester matrix equation:

*School of Engineering and Applied Sciences, Harvard University, Pierce Hall, Cambridge, MA 02138. (dfortunato@g.harvard.edu) This work is supported by the National Defense Science and Engineering Graduate Fellowship.

†Department of Mathematics, Cornell University, Ithaca, NY 14853. (townsend@cornell.edu) This work is supported by National Science Foundation grant No. 1645445.

[1]Throughout this paper, "optimal complexity" means a computational complexity that is optimal up to polylogarithmic factors.

$$KX + XK^T = F, \qquad K = -\frac{1}{h^2}\begin{bmatrix} 2 & -1 & & \\ -1 & 2 & \ddots & \\ & \ddots & \ddots & -1 \\ & & -1 & 2 \end{bmatrix} \in \mathbb{R}^{(n-1)\times(n-1)}, \tag{1.2}$$

where $h = 2/n$, $X_{jk} = u(-1+kh, -1+jh)$, and $F_{jk} = f(-1+kh, -1+jh)$ for $1 \le j, k \le n-1$. Here, the matrix X represents the values of the solution on the interior nodes of the $(n+1)\times(n+1)$ equispaced grid. The eigendecomposition of K is $K = S\Lambda S^{-1}$, where S is the normalized discrete sine transformation (of type I) matrix [18, (2.24)] and $\Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_{n-1})$ with $\lambda_k = -4/h^2 \sin^2(\pi k/(2n))$ for $1 \le k \le n-1$ [18, (2.23)]. Substituting $K = S\Lambda S^{-1}$ into $KX + XK^T = F$ and rearranging, we find a simple formula for X:

$$X = S\left(C \circ (S^{-1}FS^{-T})\right)S^T, \qquad C_{jk} = \frac{1}{\lambda_j + \lambda_k}, \tag{1.3}$$

where '$\circ$' is the Hadamard matrix product, i.e., $(A \circ B)_{jk} = A_{jk}B_{jk}$. Since $S = S^T = S^{-1}$ and matrix-vector products with S can be computed in $\mathcal{O}(n \log n)$ operations using the FFT [7], X can be computed via (1.3) in a total of $\mathcal{O}(n^2 \log n)$ operations.

Now suppose that K in (1.2) is replaced by a diagonalizable matrix A so that (1.1) has a spectrally accurate discretization of the form $AX + XA^T = F$. Then, an analogous formula to (1.2) still holds by using the eigendecomposition of A. However, the corresponding formula to (1.3) does not lead to a fast Poisson solver because the eigenvectors of A are not known in closed form [36], and deriving an optimal matrix-vector product for the eigenvector matrix of A is an ambitious project in itself. While FFT-based Poisson solvers exploit structured eigenvectors—which spectral discretization matrices do not possess—our method exploits the fact that the spectra of A and $-A$ are separated (see section 3).

We have also considered extending other fast Poisson solvers based on (i) cyclic reduction, (ii) multigrid, (iii) the fast multipole method, and (iv) the Fourier method with polynomial subtraction. These efforts were unsuccessful for various reasons: (i) cyclic reduction is not applicable because spectral discretizations of (1.1) do not involve matrices with Toeplitz structure; (ii) multigrid methods seem fruitless because the number of multigrid cycles is prohibitive with spectrally accurate methods [13]; (iii) the fast multipole method has a complexity that depends on the order of accuracy and is suboptimal in the spectral regime [14, 21]; and, (iv) pseudospectral Fourier with polynomial subtraction can be employed to derived an arbitrary-order Poisson solver [1, 6], but any approach based on uniform grids cannot be both numerically stable and spectrally accurate [26]. We conclude that many of the approaches in the literature for fast Poisson solvers do not readily extend to practical optimal complexity spectral methods for solving (1.1).

We did eventually find a fast Poisson solver based on the ADI method [25] that, with some tricks, extends from FD discretizations to spectral methods. The ADI method is an iterative method for solving Sylvester matrix equations of the form $AX - XB = F$. It is computationally efficient, compared to the $\mathcal{O}(n^3)$ Bartels–Stewart algorithm [2], when A and B have certain properties (see, for example, P1, P2, and P3 in section 2). By carefully designing spectral discretizations for Poisson's equation on the square (see section 3), cylinder (see section 4.1), solid sphere (see section 4.2),

and cube (see section 5) as Sylvester matrix equations with desired properties, we are able to derive optimal complexity, spectrally accurate Poisson solvers.

In 1979, Haidvogel and Zhang derived a Chebyshev-tau spectral method that discretizes (1.1) as a Sylvester matrix equation of the form $AX + XA^T = F$ with the matrix A being pentadiagonal except for two rows. They then applied the ADI method after precomputing the LU decomposition of A [15]. However, they advocated against their ADI-based Poisson solver in favor of an $\mathcal{O}(n^3)$ algorithm, because their Sylvester matrix equation does not possess favorable properties for the ADI method and the precomputation costs $\mathcal{O}(n^3)$ operations. In section 3, we employ a spectral discretization of (1.1) that is specifically designed for the ADI method and requires no precomputation, so that we have a provable algorithmic complexity of $\mathcal{O}(n^2(\log n)^2)$.

A typical objection to the practical relevance of spectral methods for Poisson's equation on domains such as the square and cylinder is that the solution generically has weak corner singularities, which necessarily restricts the convergence rate of classical spectral methods to subexponential convergence [5, (2.39)]. Since our spectrally accurate Poisson solvers have optimal complexity, our computational cost is comparable to low-order methods with the same number of degrees of freedom. Therefore, this objection is no longer valid.

The paper is structured as follows: In section 2, we review the ADI method for solving Sylvester matrix equations. In section 3 we derive an optimal complexity, spectrally accurate Poisson solver for (1.1). In section 4, we use partial regularity to derive fast spectral methods for Poisson's equation on the cylinder and solid sphere before discussing how to do the cube in section 5. In section 6, we describe how our methods can be used to solve Poisson's equation with general boundary conditions.

For notational convenience, throughout the paper we discretize using the same number of degrees of freedom in each variable, though our code and algorithms do not have this restriction. All code used in the paper is publicly available [11]. The Poisson solver on the square (see section 3) is implemented in Chebfun [10, 30] and can be accessed via the command `chebfun2.poisson`. It is automatically executed in Chebop2 [29] when the user inputs Poisson's equation, and can handle rectangular domains and general Dirichlet boundary conditions (see section 6).

2. The alternating direction implicit method. The alternating direction implicit method is an iterative algorithm, originally devised by Peaceman and Rachford [25], which solves Sylvester matrix equations of the following form [19]:

$$AX - XB = F, \qquad A, B, F \in \mathbb{C}^{n\times n} \tag{2.1}$$

where A, B, and F are known and $X \in \mathbb{C}^{n\times n}$ is the desired solution. In general, the ADI method is executed in an iterative fashion where iterates $X_0, X_1, \ldots,$ are computed in the hope that $\|X - X_j\|_2 \to 0$ as $j \to \infty$. Algorithm 1 summarizes the ADI method in this iterative form. At the start of the jth iteration, two shifts p_j and q_j are selected, and at the end of each iteration a test is performed to decide if the iterative method should be terminated. There are numerous strategies for selecting the shift parameters and determining when to terminate the iteration [27]. In practice, selecting good shifts for each iteration is of crucial importance for the ADI method to rapidly converge.

For an integer J, we would like to know upper bounds on $\|X - X_J\|_2$ so that we can determine a priori how many ADI iterations are required to achieve a relative accuracy of $0 < \epsilon < 1$. To develop error bounds on $\|X - X_J\|_2$, we desire (2.1) to satisfy three properties. Later, in section 3, we will design a spectral discretization

Algorithm 1 The standard ADI method to solve $AX - XB = F$

Input: $A, B, F \in \mathbb{C}^{n\times n}$
Output: $X_j \in \mathbb{C}^{n\times n}$, an approximate solution to $AX - XB = F$

$X_0 := 0$
$j := 0$
do
  Select ADI shifts p_j and q_j
  Solve $X_{j+1/2}(B - p_jI) = F - (A - p_jI)X_j$ for $X_{j+1/2}$
  Solve $(A - q_jI)X_{j+1} = F - X_{j+1/2}(B - q_jI)$ for X_{j+1}
  $j := j + 1$
while not converged
return X_j

FIG. 1. *Pseudocode for the ADI method described as an iterative algorithm for solving $AX-XB = F$. The convergence of X_j to X in the ADI method is particularly sensitive to the shifts $p_0, p_1, \ldots$ and $q_0, q_1, \ldots$. The convergence test at the end of each iteration can also be subtle [27, Sec. 2.2]. We do not use this general form of the ADI method as it does not lead to an algorithm with a provable computational complexity. Instead, we employ the ADI method on Sylvester matrix equations that satisfy P1–P3, where a different variant of the ADI method can be employed (see Algorithm 2).*

of (1.1) as a Sylvester matrix equation with these three properties.

Property 1: Normal matrices. This simplifies the error analysis of the ADI method:

P1. *The matrices A and B are normal matrices.*

In particular, when P1 holds there is a bound on the error $\|X - X_J\|_2$ that only depends on the eigenvalues of A and B and the shifts $p_0, \ldots, p_{J-1}$ and $q_0, \ldots, q_{J-1}$ [4]. Specifically,

$$\|X - X_J\|_2 \le \frac{\sup_{z\in\sigma(A)} |r(z)|}{\inf_{z\in\sigma(B)} |r(z)|}\|X\|_2, \qquad r(z) = \frac{\prod_{j=0}^{J-1}(z - p_j)}{\prod_{j=0}^{J-1}(z - q_j)},$$

where $\sigma(A)$ and $\sigma(B)$ denote the spectra of A and B, respectively. To make the upper bound on $\|X - X_J\|_2$ as small as possible, one hopes to select shifts so that

$$\frac{\sup_{z\in\sigma(A)} |r(z)|}{\inf_{z\in\sigma(B)} |r(z)|} = \inf_{s\in\mathcal{R}_J} \frac{\sup_{z\in\sigma(A)} |s(z)|}{\inf_{z\in\sigma(B)} |s(z)|}, \tag{2.2}$$

where $\mathcal{R}_J$ denotes the space of degree (J, J) rational functions. In general, it is challenging to calculate explicit shifts so that $r(z)$ attains the infimum in (2.2). However, this problem is (approximately) solved if the next property holds.

Property 2: Real and disjoint spectra. The following property of (2.1) allows us to derive explicit expressions for the ADI shifts:

P2. *There are real disjoint non-empty intervals $[a, b]$ and $[c, d]$ such that $\sigma(A) \subset [a, b]$ and $\sigma(B) \subset [c, d]$.*

If P1 and P2 both hold, then we can relax (2.2) and select ADI shifts so that

$$\|X - X_J\|_2 \le Z_J([a,b],[c,d])\|X\|_2, \quad Z_J([a,b],[c,d]) = \inf_{s\in\mathcal{R}_J} \frac{\sup_{z\in[a,b]}|s(z)|}{\inf_{z\in[c,d]}|s(z)|}, \tag{2.3}$$

where $Z_J = Z_J([a,b],[c,d])$ is referred to as a Zolotarev number. Since Zolotarev numbers have been extensively studied in the literature [3, 17, 19, 39], we are able to derive explicit expressions for the ADI shifts so that (2.3) holds. Moreover, we have an explicit upper bound on Z_J.

THEOREM 2.1. *Let J be a fixed integer and let X satisfy $AX - XB = F$, where P1 and P2 hold. Run the ADI method with the shifts*

$$p_j = T\left(-\alpha\,\mathrm{dn}\left[\frac{2j+1}{2J}K(\kappa),\kappa\right]\right), \quad q_j = T\left(\alpha\,\mathrm{dn}\left[\frac{2j+1}{2J}K(\kappa),\kappa\right]\right), \tag{2.4}$$

for $0 \le j \le J-1$, where $\kappa = \sqrt{1-1/\alpha^2}$, $K(\kappa)$ is the complete elliptic integral of the first kind [23, (19.2.8)], and $\mathrm{dn}(z,\kappa)$ is the Jacobi elliptic function of the third kind [23, (22.2.6)]. Here, α is the real number given by $\alpha = -1 + 2\gamma + 2\sqrt{\gamma^2-\gamma}$ with $\gamma = |c-a||d-b|/(|c-b||d-a|)$ and T is the Möbius transformation that maps $\{-\alpha,-1,1,\alpha\}$ to $\{a,b,c,d\}$. Then, the ADI iterate X_J satisfies

$$\|X - X_J\|_2 \le Z_J\|X\|_2, \qquad Z_J([a,b],[c,d]) \le 4\left[\exp\left(\frac{\pi^2}{4\mu(1/\sqrt{\gamma})}\right)\right]^{-2J}, \tag{2.5}$$

where $\mu(\lambda) = \frac{\pi}{2}K(\sqrt{1-\lambda^2})/K(\lambda)$ is the Grötzsch ring function.

Proof. If $c = -b$ and $d = -a$, then the ADI shifts to ensure that $\|X - X_J\|_2 \le Z_J([-b,-a],[a,b])\|X\|_2$ are given in [19, (2.18)] as

$$p_j = -b\,\mathrm{dn}\left[\frac{2j+1}{2J}K(\sqrt{1-a^2/b^2}),\sqrt{1-a^2/b^2}\right], \quad q_j = -p_j, \quad 0 \le j \le J-1. \tag{2.6}$$

For the α given in the statement of the theorem, there exists a Möbius transformation T that maps $\{-\alpha,-1,1,\alpha\}$ to $\{a,b,c,d\}$ because the two sets of collinear points have the same absolute cross-ratio. Since any Möbius transformation maps rational functions to rational functions, $Z_J([-\alpha,-1],[1,\alpha]) = Z_J([a,b],[c,d])$ with the zeros and poles of the associated rational functions (see (2.3)) related by the Möbius transformation T. The formula (2.4) is immediately derived as $T(p_j)$ and $T(q_j)$, where p_j and q_j in (2.6) are taken with $a = 1$ and $b = \alpha$. ∎

We often prefer to simplify the bound in (2.5) by removing the Grötzsch ring function from the bound on Z_J. For example, the bound in (2.5) remains valid, but is slightly weakened, if $4\mu(1/\sqrt{\gamma})$ is replaced by the upper bound $2\log(16\gamma)$ [3], i.e.,

$$\|X - X_J\|_2 \le 4\left[\exp\left(\frac{\pi^2}{2\log(16\gamma)}\right)\right]^{-2J}\|X\|_2, \qquad \gamma = \frac{|c-a||d-b|}{|c-b||d-a|}. \tag{2.7}$$

Moreover, if $c = -b$ and $d = -a$ (which commonly occurs when $B = -A^T$), then the bound simplifies even more as $4\mu(1/\sqrt{\gamma}) = 2\mu(a/b)$ and the bound remains valid if $2\mu(a/b)$ is replaced by $\log(4b/a)$. That is,

$$\|X - X_J\|_2 \le 4\left[\exp\left(\frac{\pi^2}{\log(4b/a)}\right)\right]^{-2J}\|X\|_2. \tag{2.8}$$

Algorithm 2 The ADI method to solve $AX - XB = F$ when P1 and P2 hold

Input: $A, B, F \in \mathbb{C}^{n\times n}$, $a, b, c, d \in \mathbb{R}$ satisfying P2, and a tolerance $0 < \epsilon < 1$
Output: $X_J \in \mathbb{C}^{n\times n}$ such that $\|X - X_J\|_2 \le \epsilon\|X\|_2$

```
γ := |c − a||d − b|/(|c − b||d − a|)
J := ⌈log(16γ) log(4/ϵ)/π²⌉
Set p_j and q_j for 0 ≤ j ≤ J − 1 as given in (2.4)
X_0 := 0
for j = 0, ..., J − 1 do
    Solve X_{j+1/2}(B − p_j I) = F − (A − p_j I)X_j for X_{j+1/2}
    Solve (A − q_j I)X_{j+1} = F − X_{j+1/2}(B − q_j I) for X_{j+1}
end for
return X_J
```

FIG. 2. *Pseudocode for the ADI method for solving $AX - XB = F$ when P1 and P2 hold. Here, for any relative accuracy $0 < \epsilon < 1$ the number of ADI iterations, J, and shifts $p_0, \ldots, p_{J-1}$ and $q_0, \ldots, q_{J-1}$ are known such that $\|X - X_J\|_2 \le \epsilon\|X\|_2$.*

Theorem 2.1 is very fruitful as it allows us to use the ADI method more like a direct method to solve $AX - XB = F$ when P1 and P2 hold. For a relative accuracy of $0 < \epsilon < 1$, the simplified bound in (2.7) shows that $\|X - X_J\|_2 \le \epsilon\|X\|_2$ if we take

$$J = \left\lceil \frac{\log(16\gamma)\log(4/\epsilon)}{\pi^2} \right\rceil \tag{2.9}$$

and we run the ADI method with the shifts given in (2.4). Algorithm 2 summarizes the ADI method on $AX - XB = F$ when P1 and P2 hold. This is the variant of the ADI method that we employ throughout this paper.

We appreciate that it is awkward to calculate the shifts in (2.4) because they involve complete elliptic integrals and Jacobi elliptic functions. For the reader's convenience, we provide MATLAB code to compute the shifts in Appendix A. Note that computing the shifts can be done in $\mathcal{O}(1)$ operations, independent of n.

Property 3: Fast shifted linear solves. There is still one more important property of $AX - XB = F$. The shifted linear solves in Algorithm 2 need to be computationally cheap:

P3. *For any $p, q \in \mathbb{C}$, the linear systems $(A - pI)x = b$ and $(B - qI)x = b$ can be solved in $\mathcal{O}(n)$ operations.*

If P3 holds, then each ADI iteration costs only $\mathcal{O}(n^2)$ operations and the overall cost of the ADI method with J iterations is $\mathcal{O}(Jn^2)$ operations.

In summary, properties P1, P2, and P3 are sufficient conditions on $AX - XB = F$ so that (i) we can determine the number of ADI iterations to attain a relative accuracy of $0 < \epsilon < 1$, (ii) we can derive explicit expressions for the ADI shifts, and (iii) we can compute each ADI iteration in $\mathcal{O}(n^2)$ operations.

2.1. An ADI-based fast Poisson solver for finite difference methods. We now describe the ADI-based fast Poisson solver with the second-order five-point FD stencil, though the approach easily extends to fourth- and sixth-order FD methods. Recall that the FD discretization of (1.1) with a five-point stencil on an $(n+1)\times(n+1)$

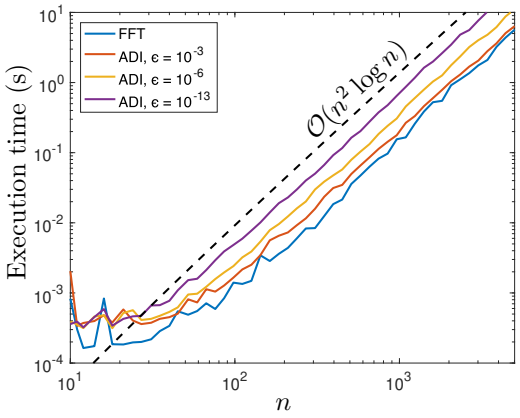


FIG. 3. *Execution times for the ADI- and FFT-based fast Poisson solvers for a 5-point FD discretization with* $10 \leq n \leq 5000$. *The ADI-based solver is comparable to the FFT-based solver when* $\epsilon = 10^{-3}$. *While the ADI-based fast Poisson solver is computationally more expensive, it is applicable to a carefully designed spectral discretization. Since FFT-based fast Poisson solver necessarily require uniform grids, they cannot provide a practical optimal complexity spectral method [26].*

equispaced grid is given by the Sylvester matrix equation $KX + XK^T = F$ (see (1.2)). We now verify that P1, P2, and P3 hold for $KX + XK^T = F$:

P1: $A = K$ and $B = -K^T$ are real and symmetric, so they are normal matrices.

P2: The eigenvalues of K are given by $-4/h^2 \sin^2(\pi k/(2n))$ for $1 \leq k \leq n-1$ with $h = 2/n$ [18, (2.23)]. Since $(2/\pi)x \leq \sin x \leq 1$ for $x \in [0, \pi/2]$ and $h = 2/n$, the eigenvalues of $A = K$ are contained in the interval $[-n^2, -1]$. The eigenvalues of $B = -K^T$ are contained in $[1, n^2]$.

P3: For any $p, q \in \mathbb{C}$, the linear systems $(A - pI)x = b$ and $(B - qI)x = b$ are tridiagonal and hence can be solved via the Thomas algorithm in $\mathcal{O}(n)$ operations [9, p. 162].

From the simplified bound in (2.8), we conclude that $J = \lceil \log(2n) \log(4/\epsilon)/\pi^2 \rceil$ ADI iterations are sufficient to ensure that $\|X - X_J\|_2 \leq \epsilon \|X\|_2$ for $0 < \epsilon < 1$, where the shifts are given in Theorem 2.1. Moreover, since P3 holds each ADI iteration only costs $\mathcal{O}(n^2)$ iterations. We conclude that the ADI method in Algorithm 2 solves $KX + XK^T = F$ in a total of $\mathcal{O}(n^2 \log n \log(1/\epsilon))$ operations. Figure 3 demonstrates the execution time[2] of this approach in comparison to the FFT-based fast Poisson solver for $10 \leq n \leq 5000$. While we are not advocating the use of the ADI-based fast Poisson solver for the five-point FD stencil, it does provide flexibility through the choice of an error tolerance ϵ and may be useful for higher-order FD methods and non-uniform grids. As we will show in the next section, ADI-based solvers extend to carefully designed spectrally accurate discretizations (see section 3).

We expect that one can also derive ADI-based fast Poisson solvers for any $(4w+1)$-point FD stencil, $1 \leq w \leq \lfloor (n-1)/2 \rfloor$, that run in an optimal number of $\mathcal{O}(n^2 \log n \log(1/\epsilon))$ operations. Because FD discretization matrices have Toeplitz structure, one shifted linear solve only costs $\mathcal{O}(n \log n)$ operations using FFTs [20].

[2] All timings in the paper were performed in MATLAB R2017a on a 2017 Macbook Pro with no explicit parallelization.

Unfortunately, for $w = \lfloor (n-1)/2 \rfloor$ the resulting spectrally accurate method must be numerically unstable because it is based on equispaced nodes [26].

3. A fast spectral Poisson solver on the square. Consider Poisson's equation on the square with zero homogeneous Dirichlet conditions:

$$u_{xx} + u_{yy} = f, \quad (x,y) \in [-1,1]^2, \qquad u(\pm 1, \cdot) = u(\cdot, \pm 1) = 0. \tag{3.1}$$

Since (3.1) has homogeneous Dirichlet conditions, we know that the solution can be written as $u(x,y) = (1-x^2)(1-y^2)v(x,y)$ for some function $v(x,y)$. To ensure that we are deriving a stable spectral method, we expand $v(x,y)$ in a standard orthogonal polynomial basis[3] [33]. That is,

$$u(x,y) \approx \sum_{i=0}^{n-1} \sum_{j=0}^{n-1} X_{ij} (1-y^2)(1-x^2)\phi_i(y)\phi_j(x), \qquad (x,y) \in [-1,1]^2, \tag{3.2}$$

where $\phi_0, \phi_1, \ldots,$ are a sequence of orthogonal polynomials on $[-1,1]$ and the degree of ϕ_j is exactly j for $j \geq 0$. Here, $X \in \mathbb{C}^{n \times n}$ is the matrix of expansion coefficients of the solution and we wish to find X so that the first $n \times n$ coefficients of $u_{xx} + u_{yy}$ match those of f. The choice of the orthogonal polynomial basis is critically important to derive our optimal complexity ADI-based fast Poisson solver. In particular, we want to construct a Sylvester matrix equation for which P1, P2, and P3 hold. If, for example, the Chebyshev basis is selected, then the resulting Sylvester matrix equation does not satisfy P1 from section 2.

3.1. An ultraspherical polynomial basis. To simplify the discretization of u_{xx} in (3.1), we select ϕ_j so that $\frac{d^2}{dx^2}\big[(1-x^2)\phi_j(x)\big]$ has a simple form in terms of $\phi_j(x)$. By the chain rule, we have

$$\frac{d^2}{dx^2}\big[(1-x^2)\phi_j(x)\big] = (1-x^2)\phi_j^{''}(x) - 4x\phi_j^{'}(x) - 2\phi_j(x), \tag{3.3}$$

where a prime indicates one derivative in x. In [23, Chap. 18], one finds that the normalized ultraspherical polynomial,[4] denoted by $\tilde{C}_j^{(3/2)}(x)$, of degree j and parameter $3/2$ satisfies the second-order differential equation [23, Table 18.8.1]

$$(1-x^2)\tilde{C}_j^{(3/2)''}(x) - 4x\tilde{C}_j^{(3/2)'}(x) + j(j+3)\tilde{C}_j^{(3/2)}(x) = 0, \qquad x \in [-1,1]. \tag{3.4}$$

In particular, this means that $\tilde{C}_j^{(3/2)}(x)$ is a eigenfunction of the differential operator $u \mapsto \frac{d^2}{dx^2}\big[(1-x^2)u\big]$, i.e.,

$$\frac{d^2}{dx^2}\Big[(1-x^2)\tilde{C}_j^{(3/2)}(x)\Big] = -(j(j+3)+2)\tilde{C}_j^{(3/2)}(x), \qquad j \geq 0.$$

Encouraged by this simplification, we select $\phi_j = \tilde{C}_j^{(3/2)}$ in (3.2).

[3] Additional benefits of choosing standard orthogonal polynomials include fast evaluation using Clenshaw's algorithm and fast transforms.

[4] The ultraspherical polynomial of degree j and parameter $\lambda > 0$ is denoted by $C_j^{(\lambda)}$, where $C_0^{(\lambda)}, C_1^{(\lambda)}, \ldots$ are orthogonal on $[-1,1]$ with respect to the weight function $(1-x^2)^{\lambda - 1/2}$. The normalized ultraspherical polynomials of parameter $3/2$, denoted by $\tilde{C}_j^{(3/2)}$, satisfy

$$\tilde{C}_j^{(3/2)}(x) = \sqrt{\frac{j+3/2}{(j+1)(j+2)}} C_j^{(3/2)}(x), \qquad j \geq 0,$$

so that $\int_{-1}^{1} (\tilde{C}_j^{(3/2)}(x))^2 (1-x^2) dx = 1$.

3.2. A spectral discretization of Poisson's equation. To construct a discretization of (3.1), we apply the Laplacian to the expansion in (3.2) to derive a set of equations that the matrix X must satisfy. The action of the Laplacian on each element of our basis is given by

$$\begin{aligned}&\nabla^2\left[(1-y^2)(1-x^2)\tilde{C}_i^{(3/2)}(y)\tilde{C}_j^{(3/2)}(x)\right]\\&=-\Big[(j(j+3)+2)(1-y^2)+(i(i+3)+2)(1-x^2)\Big]\tilde{C}_i^{(3/2)}(y)\tilde{C}_j^{(3/2)}(x).\end{aligned}\tag{3.5}$$

Therefore, we can discretize (3.1) as a generalized Sylvester matrix equation

$$MXD^T+DXM^T=F,\tag{3.6}$$

where X is the matrix of $(1-y^2)(1-x^2)\tilde{C}^{(3/2)}(y)\tilde{C}^{(3/2)}(x)$ expansion coefficients for the solution $u(x,y)$ in (3.2), F is the matrix of bivariate $\tilde{C}^{(3/2)}$ expansion coefficients for f (see section 3.4), D is a diagonal matrix with $D_{jj}=-(j(j+3)+2)$, and M is the $n\times n$ matrix that represents multiplication by $1-x^2$ in the $\tilde{C}^{(3/2)}$ basis. Since the recurrence relation for the unnormalized ultraspherical polynomials, $C^{(3/2)}$, is given by [23, (18.9.7) & (18.9.8)]

$$\begin{aligned}(1-x^2)C_j^{(3/2)}(x)=-\frac{(j+1)(j+2)}{(2j+1)(2j+3)(2j+5)}\Big[&(2j+1)C_{j+2}^{(3/2)}(x)-2(2j+3)C_j^{(3/2)}(x)\\&+(2j+5)C_{j-2}^{(3/2)}(x)\Big],\end{aligned}$$

we find—after algebraic manipulations—that M is a symmetric pentadiagonal matrix with

$$M_{j,j}=\frac{2(j+1)(j+2)}{(2j+1)(2j+5)},\quad M_{j,j+1}=0,\quad M_{j,j+2}=\frac{-1}{(2j+3)(2j+5)}\sqrt{\frac{(j+4)!(2j+3)}{j!(2j+7)}}.\tag{3.7}$$

We can rearrange (3.6) by applying D^{-1} to obtain the standard Sylvester matrix equation

$$AX-XB=D^{-1}FD^{-1},\qquad A=D^{-1}M,\quad B=-M^TD^{-1}.\tag{3.8}$$

3.3. Verifying that P1, P2, and P3 hold. To guarantee that the ADI method for solving (3.8) has optimal complexity, we want the Sylvester matrix equation to satisfy P1, P2, and P3 (see section 2). Unfortunately, the matrices A and B in (3.8) are not normal matrices, so we do not solve (3.8) using the ADI method directly. Instead, we note that A and $B=-A^T$ are pentadiagonal matrices with zeros on the sub- and super-diagonals so that there exists a diagonal matrix D_s for which $\tilde{A}=D_s^{-1}AD_s$ and $\tilde{B}=-\tilde{A}^T=-\tilde{A}$ are real symmetric pentadiagonal matrices. Therefore, to solve (3.8) we solve the following Sylvester matrix equation:

$$\tilde{A}Y-Y\tilde{B}=D_s^{-1}(D^{-1}FD^{-1})D_s^{-1},\qquad Y=D_s^{-1}XD_s,\tag{3.9}$$

and recover X via $X=D_sYD_s^{-1}$. We now verify that P1, P2, and P3 hold for (3.9):

- P1: $\tilde{A}$ and $\tilde{B}$ are real and symmetric so are normal matrices,
- P2: The eigenvalues of $\tilde{A}$ are contained in the interval $[-1,-1/(30n^4)]$ (see Appendix B). The eigenvalues of $\tilde{B}=-\tilde{A}^T$ are contained in $[1/(30n^4),1]$.

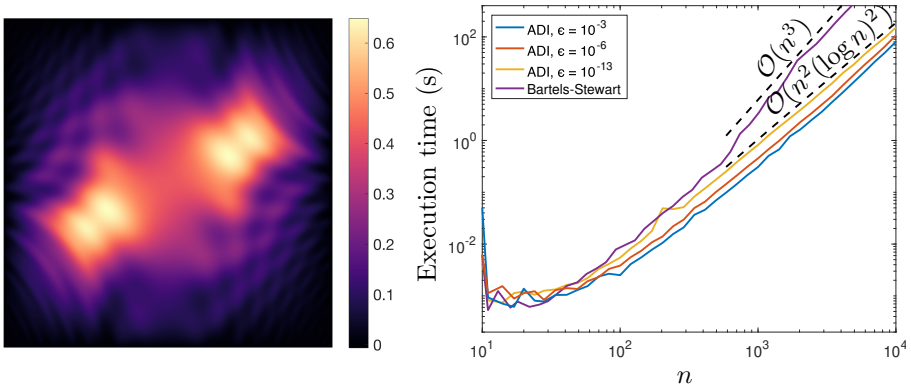


FIG. 4. *Left: A computed solution to Poisson's equation on the square with right-hand side* $f(x,y) = -100x\sin(20\pi x^2 y)\cos(4\pi(x+y))$ *and* $n = 200$*, using an error tolerance of* $\epsilon = 10^{-13}$*. Right: Execution times for solving* $u_{xx} + u_{yy} = f$ *on* $[-1,1]^2$ *with zero homogeneous Dirichlet boundary conditions, using both our ADI-based solver with various error tolerances and the Bartels–Stewart algorithm [2].*

P3: For any $p, q \in \mathbb{C}$, the linear systems $(\tilde{A} - pI)x = b$ and $(\tilde{B} - qI)x = b$ are pentadiagonal matrices with zero sub- and super-diagonals. Hence, they can be solved in $\mathcal{O}(n)$ operations using the Thomas algorithm [9, p. 162].

By Theorem 2.1, we need at most

$$J = \lceil \log(120n^4)\log(1/\epsilon))/(2\pi^2) \rceil.$$

ADI iterations to ensure that we solve (3.9) to within a relative accuracy of $0 < \epsilon < 1$. Since P3 holds, the ADI method solves (3.9) in $\mathcal{O}(n^2 \log n \log(1/\epsilon))$ operations, and an additional $\mathcal{O}(n^2)$ operations recovers X from Y.

3.4. Computing the ultraspherical coefficients of a function. So far our Poisson solver assumes that (a) one is given the $\tilde{C}^{(3/2)}$ expansion coefficients for f in (3.1) and (b) one is satisfied with the solution returned in the form (3.2).

It is known how to compute the Legendre expansion coefficients F_{leg} from f in $\mathcal{O}(n^2(\log n)^2 \log(1/\epsilon))$ operations [31].[5] Using the fact that [23, (18.7.9) & (18.9.7)]

$$(j+\tfrac{1}{2})P_j(x) = \sqrt{\frac{(j+1)(j+2)}{(j+3/2)}}\tilde{C}_j^{(3/2)}(x) - \sqrt{\frac{j(j-1)}{(j-1/2)}}\tilde{C}_{j-2}^{(3/2)}(x), \qquad j \geq 2,$$

there is a sparse upper-triangular matrix S that converts Legendre coefficients to $\tilde{C}^{(3/2)}$ coefficients. Moreover, we can compute $F = S^{-1}F_{\text{leg}}S^{-T}$ in $\mathcal{O}(n^2)$ operations by backwards substitution.

Once the expansion coefficients X in (3.2) are known, one can convert the expansion coefficients to a Legendre or Chebyshev basis. The normalized ultraspherical coefficients are given by $X_{\text{ultra}} = MXM^T$ because of the $(1-y^2)(1-x^2)$ factor in (3.2). To obtain the Legendre coefficients for u, we note that $X_{\text{leg}} = SX_{\text{ultra}}S^T$. One can now construct a bivariate Chebyshev expansion of u.[6]

[5]The Chebfun code to compute the $n \times n$ Legendre coefficients of f is `g = chebfun2(@(x,y) f(x,y)); Fleg = cheb2leg(cheb2leg(chebcoeffs2(g,n,n)).').';` [10].

[6]The Chebfun code to construct a bivariate Chebyshev expansion from a matrix of Legendre coefficients is `u = chebfun2( leg2cheb(leg2cheb(Xleg).').', 'coeffs' )` [10].

TABLE 1

Summary of our optimal complexity, spectrally accurate Poisson solver on the square with an $n \times n$ discretization. The algorithm costs $\mathcal{O}(n^2(\log n)^2 \log(1/\epsilon))$ operations for a working tolerance of $0 < \epsilon < 1$. For $n \leq 5000$, the dominating computational cost in practice is the ADI method.

Algorithmic step	Cost
1. Compute the $\tilde{C}^{(3/2)}$ coefficients of f in (3.1) using [31]	$\mathcal{O}(n^2(\log n)^2 \log(1/\epsilon))$
2. Solve (3.9) via the ADI method	$\mathcal{O}(n^2 \log n \log(1/\epsilon))$
3. Compute the solution to (3.8) as $X = D_s Y D_s^{-1}$	$\mathcal{O}(n^2)$
4. Compute the Chebyshev coefficients of u using [31]	$\mathcal{O}(n^2(\log n)^2 \log(1/\epsilon))$

Table 1 summarizes our spectrally accurate and optimal complexity Poisson solver. The overall complexity is $\mathcal{O}(n^2(\log n)^2 \log(1/\epsilon))$, after the coefficient transforms are taken into account.

Figure 4 shows our method compared to the Bartels–Stewart algorithm [2] (invoked via the `lyap` command in MATLAB) used to solve the Sylvester equation (3.8). The Bartels–Stewart algorithm costs $\mathcal{O}(n^3)$ operations; as the timings demonstrate, our method is significantly faster once n is larger than a few hundred. In addition, there are important advantages of ADI in our setting: we are able to relax the tolerance ϵ according to the application, allowing the algorithm to exploit that parameter for a reduced computational cost. The solver can also easily be extended to any rectangular domain $[a, b] \times [c, d]$. Our Poisson solver on the rectangle can be accessed in [11] via the command `poisson_rectangle(F, lbc, rbc, dbc, ubc, [a b c d], tol)`, where `F` is the matrix of bivariate Chebyshev coefficients for the right-hand side, `lbc`, `rbc`, `dbc`, and `ubc` denote the left, right, bottom and top Dirichlet data, respectively, and `tol` is the error tolerance.

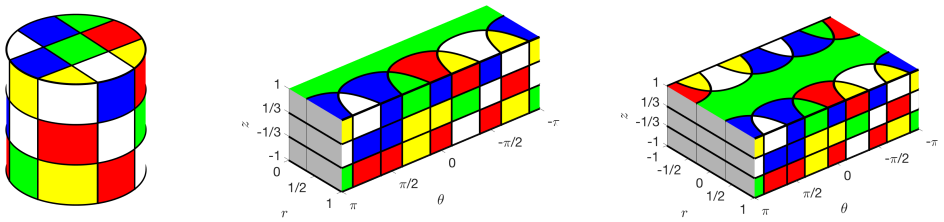
4. Fast spectral Poisson solvers on cylindrical and spherical geometries. We now describe how to extend our fast Poisson solver to cylindrical and spherical geometries. We exploit the fact that both the cylindrical and spherical Laplacians decouple in the azimuthal variable, allowing us to reduce the full three-dimensional problem into n independent two-dimensional problems that can be solved by ADI. On both geometries, we employ a variant of the double Fourier sphere method [22] (see section 4.1.1) and impose partial regularity on the solution to ensure smoothness.

4.1. A fast spectral Poisson solver on the cylinder. Here, we consider solving Poisson's equation on the cylinder, i.e., $u_{xx} + u_{yy} + u_{zz} = f$ on $x^2 + y^2 \in [0, 1]$ and $z \in [-1, 1]$ with homogeneous Dirichlet conditions. Our first step is to change to the cylindrical coordinate system, i.e., $(x, y, z) = (r\cos\theta, r\sin\theta, z)$ where $r \in [0, 1]$ is the radial variable and $\theta \in [-\pi, \pi]$ is the angular variable. This change-of-variables transforms Poisson's equation to

$$\frac{\partial^2 u}{\partial r^2} + \frac{1}{r}\frac{\partial u}{\partial r} + \frac{1}{r^2}\frac{\partial^2 u}{\partial \theta^2} + \frac{\partial^2 u}{\partial z^2} = f, \qquad (r, \theta, z) \in [0, 1] \times [-\pi, \pi] \times [-1, 1], \tag{4.1}$$

where $u(1, \theta, z) = 0$ for $(\theta, z) \in [-\pi, \pi] \times [-1, 1]$ and $u(r, \theta, \pm 1) = 0$ for $(r, \theta) \in [0, 1] \times [-\pi, \pi]$.

The coordinate transform has simplified the domain of the differential equation to a rectangle, but has several issues: (1) Any point of the form $(0, \theta, z)$ with $\theta \in [-\pi, \pi]$



(a) (b) (c)

FIG. 5. *Illustration of the DFS method for a Rubik's cube-colored cylinder. (a) The Rubik's cube-colored cylinder. (b) The Rubik's cube-colored cylinder projected into cylindrical coordinates. (c) The Rubik's cube-colored cylinder after applying the DFS method. The DFS method represents a smooth function $f(x, y, z)$ on the cylinder with a function $f(r, \theta, z)$ on $[-1, 1] \times [-\pi, \pi] \times [-1, 1]$ that is 2π-periodic in θ and $f(0, \theta, z)$ is a constant for each $\theta \in [-\pi, \pi]$ and $z \in [-1, 1]$.*

and $z \in [-1, 1]$ maps to $(0, 0, z)$ in Cartesian coordinates, introducing an artificial singularity along the center line $r = 0$, (2) The differential equation in (4.1) is second-order in the r-variable, but we do not have a natural boundary condition to impose at $r = 0$, and (3) Not every function in the variables (r, θ, z) is a well-defined function on the cylinder, so additional constraints must be satisfied by $u = u(r, \theta, z)$ in (4.1).

4.1.1. The double Fourier sphere method for the cylinder. The double Fourier sphere (DFS) method, originally proposed for computations on the surface of the sphere [22, 32], is a simple technique that alleviates many of the concerns with cylindrical coordinate transforms. Instead of solving (4.1), we "double-up" u and f to $\tilde{u}$ and $\tilde{f}$ and solve

$$\frac{\partial^2 \tilde{u}}{\partial r^2} + \frac{1}{r}\frac{\partial \tilde{u}}{\partial r} + \frac{1}{r^2}\frac{\partial^2 \tilde{u}}{\partial \theta^2} + \frac{\partial^2 \tilde{u}}{\partial z^2} = \tilde{f}, \qquad (r, \theta, z) \in [-1, 1] \times [-\pi, \pi] \times [-1, 1], \tag{4.2}$$

where the r-variable is now over $[-1, 1]$, instead of $[0, 1]$. Here, the solution u (resp. f) is doubled-up as follows:

$$\tilde{u}(r, \theta, z) = \begin{cases} u(r, \theta, z), & (r, \theta, z) \in [0, 1] \times [-\pi, \pi] \times [-1, 1], \\ u(-r, \theta + \pi, z), & (r, \theta, z) \in [-1, 0] \times [-\pi, \pi] \times [-1, 1] \end{cases} \tag{4.3}$$

and the homogeneous Dirichlet conditions become $\tilde{u}(\pm 1, \theta, z) = 0$ for $(\theta, z) \in [-\pi, \pi] \times [-1, 1]$ and $\tilde{u}(r, \theta, \pm 1) = 0$ for $(r, \theta) \in [-1, 1] \times [-\pi, \pi]$. Figure 5 illustrates the DFS method when applied to a Rubik's cube-colored cylinder.

The doubled-up functions $\tilde{u}$ and $\tilde{f}$ are non-periodic in the r- and z-variables, and 2π-periodic in the θ-variable. Therefore, we seek the coefficients for $\tilde{u}$ in a Chebyshev–Fourier–Chebyshev expansion:

$$\tilde{u}(r, \theta, z) \approx \sum_{k=-n/2}^{n/2-1} \tilde{u}_k(r, z) e^{\mathrm{i}k\theta}, \qquad \tilde{u}_k(r, z) = \sum_{i=0}^{n-1}\sum_{j=0}^{n-1} X_{ij}^{(k)} T_i(r) T_j(z), \tag{4.4}$$

where we assume that n is an even integer and $\tilde{u}_k(r, z)$ denotes the kth Fourier mode of $\tilde{u}(r, \cdot, z)$. We have written the Chebyshev–Fourier–Chebyshev expansion in this form because it turns out that each Fourier mode can be solved for separately. Since

$\tilde{f}(r,\theta,z) \approx \sum_{k=-n/2}^{n/2-1} \tilde{f}_k(r,z)e^{\mathrm{i}k\theta}$, we can plug (4.4) into (4.2) to find that

$$\frac{\partial^2 \tilde{u}_k}{\partial r^2} + \frac{1}{r}\frac{\partial \tilde{u}_k}{\partial r} - \frac{k^2}{r^2}\tilde{u}_k + \frac{\partial^2 \tilde{u}_k}{\partial z^2} = \tilde{f}_k, \qquad (r,z) \in [-1,1]\times[-1,1], \tag{4.5}$$

for each $-n/2 \le k \le n/2-1$. This allows us to solve the trivariate PDE in (4.2) with a system of n independent bivariate PDEs for each $u_k(r,z)$.

4.1.2. Imposing partial regularity on the solution. The issue with (4.4) is that a Chebyshev–Fourier–Chebyshev expansion in (r,θ,z) does not necessarily represent a smooth function in (x,y,z) on the cylinder. For instance, $\tilde{u}(0,\theta,z)$ must be a function of the z-variable only for the corresponding function on the cylinder to be continuous. Since we have $x = r\cos\theta$ and $y = r\sin\theta$, we know that the kth Fourier mode $\tilde{u}_k(r,z)$ must decay like $\mathcal{O}(r^{|k|})$ as $r \to 0$. By the uniqueness of Fourier expansions, we also know that $\tilde{u}_k(\pm 1,z) = 0$ and $\tilde{u}_k(r,\pm 1) = 0$ for $-n/2 \le k \le n/2-1$. Therefore, we know that there must be a function[7] $\tilde{v}_k(r,z)$ such that

$$\tilde{u}_k(r,z) = (1-r^2)(1-z^2)r^{|k|}\tilde{v}_k(r,z), \qquad -\frac{n}{2} \le k \le \frac{n}{2}-1. \tag{4.6}$$

Ideally, we would like to numerically compute for a bivariate Chebyshev expansion for $\tilde{v}_k(r,z)$ and then recover $\tilde{u}_k(r,z)$ from (4.6). This would ensure that the solution $\tilde{u}(r,\theta,z)$ corresponds to a smooth function on the cylinder.

Unfortunately, imposing full regularity on $\tilde{u}_k(r,z)$ is numerically problematic because the regularity condition involves high-order monomial powers. The idea of imposing *partial regularity* on $\tilde{u}_k(r,z)$ avoids the high degree monomial terms [28], and instead $\tilde{u}_k(r,z)$ is written as:

$$\tilde{u}_k(r,z) = (1-r^2)(1-z^2)r^{\min(|k|,2)}\tilde{\omega}_k(r,z), \qquad -\frac{n}{2} \le k \le \frac{n}{2}-1, \tag{4.7}$$

where the regularity requirements from (4.6) is relaxed. If the functions $\tilde{\omega}_k(r,z)$ are additionally imposed to be even (odd) in r if k is even (odd), then the the function $\tilde{u}(r,\theta,z)$ corresponds to at least a continuously differentiable function on the cylinder.

4.1.3. A solution method for each Fourier mode. The partial regularity conditions in (4.7) naturally split into three cases that we treat separately: $|k| \ge 2$ (Case 1), $|k| = 1$ (Case 2), and $k = 0$ (Case 3) . In terms of developing a fast Poisson solver for (4.1), it is only important that the PDEs in (4.5) for $|k| \ge 2$ are solved in optimal complexity.

Case 1: $|k| \ge 2$**.** The idea is to solve for the function $\tilde{\omega}_k(r,z)$, where $\tilde{u}_k(r,z) = r^2(1-r^2)(1-z^2)\tilde{\omega}_k(r,z)$ and afterwards to recover $\tilde{u}_k(r,z)$. To achieve this, we find the differential equation that $\tilde{\omega}_k(r,z)$ satisfies by substituting (4.7) into (4.5). After simplifying, we obtain the following equation:

$$\Bigg[\underbrace{r^2(1-r^2)\frac{\partial^2\tilde{\omega}_k}{\partial r^2} + (5-9r^2)r\frac{\partial\tilde{\omega}_k}{\partial r} + 4(1-4r^2)\tilde{\omega}_k}_{=\mathcal{L}_1} - k^2(1-r^2)\tilde{\omega}_k\Bigg](1-z^2) + r^2(1-r^2)\underbrace{\left[(1-z^2)\frac{\partial^2\tilde{\omega}_k}{\partial z^2} - 4z\frac{\partial\tilde{\omega}_k}{\partial z} - 2\tilde{\omega}_k\right]}_{=\mathcal{L}_2} = \tilde{f}_k, \tag{4.8}$$

[7] One can also show that $\tilde{v}_k(r,z)$ must be an even (odd) function of r if k is even (odd).

where no boundary conditions are required. Focusing on the z-variable, we observe that $\mathcal{L}_2$ is identical to the differential equation in section 3.1. Therefore, we represent the z-variable of $\tilde{\omega}_k(r,z)$ in an ultraspherical expansion because $\tilde{C}_j^{(3/2)}$ is an eigenfunction of $\mathcal{L}_2$. For the r-variable, we also use the $\tilde{C}^{(3/2)}$ basis because the multiplication matrix for $(1-r^2)$ is a normal matrix (see (3.7)).

Since the $k^2(1-r^2)\tilde{\omega}_k$ term dominates $\mathcal{L}_1$ when k is large, the discretization of $\mathcal{L}_1 - k^2(1-r^2)\tilde{\omega}_k$ in the $\tilde{C}^{(3/2)}$ basis is a near-normal[8] matrix; the matrix tends to a normal matrix as $k \to \infty$. Therefore, we represent $\tilde{\omega}_k(r,z)$ as

$$\tilde{\omega}_k(r,z) \approx \sum_{i=0}^{n-1}\sum_{j=0}^{n-1} Y_{ij}^{(k)} \tilde{C}_i^{(3/2)}(r)\tilde{C}_j^{(3/2)}(z). \tag{4.9}$$

One can show that an $n \times n$ discretization of $\mathcal{L}_1$ is given by

$$L_1 = M_{r^2}D + 5M_rM_{1-r^2}D_1 + 14M_{1-r^2} - 6I,$$

where D is given in (3.6), $M_{1-r^2} = M$ (see (3.7)), I is the $n \times n$ identity matrix, $M_{r^2} = I - M_{1-r^2}$, M_r is multiplication by r in the $\tilde{C}^{(3/2)}$ basis and D_1 is the first-order differentiation matrix. While D_1 is a upper-triangular dense matrix, we note that $M_{1-r^2}D_1$ is a tridiagonal matrix from [23, (18.9.8) & (18.9.19)]. Moreover, M_r is a tridiagonal matrix [23, Tab. 18.9.1] and hence, L_1 is a pentadiagonal matrix.

Looking at (4.8), we find that the coefficient matrix $Y^{(k)}$ in (4.9) satisfies

$$(L_1 - k^2M_{1-r^2})Y^{(k)}M_{1-r^2}^T + M_{r^2}M_{1-r^2}Y^{(k)}D = F_k,$$

which after rearranging becomes the following Sylvester matrix equation:

$$AY^{(k)} - Y^{(k)}B = (L_1 - k^2M_{1-r^2})^{-1}F_kD^{-1}, \tag{4.10}$$

where $A = (L_1 - k^2M_{1-r^2})^{-1}M_{1-r^2}$ and $B = -M_{1-r^2}^TD^{-1}$. Here, B is a normal pentadiagonal matrix after a diagonal similarity transform and A is a near-normal matrix which tends to a normal matrix as k gets large. Moreover, we observe that A has real eigenvalues that are well-separated from the eigenvalues of B and we can solve linear systems of the form $(A - pI)x = b$ in $\mathcal{O}(n)$ operations as $(M_{1-r^2} - p(L_1 - k^2M_{1-r^2}))x = (L_1 - k^2M_{1-r^2})b$. Therefore, we can apply ADI to (4.10) to solve for each $Y^{(k)}$ in $\mathcal{O}(n^2(\log n)^2\log(1/\epsilon))$ operations. Since there are $\mathcal{O}(n)$ such $Y^{(k)}$, the total complexity is $\mathcal{O}(n^3(\log n)^2\log(1/\epsilon))$. We recover $\tilde{u}_k(r,z)$ via the relation $\tilde{u}_k(r,z) = r^2(1-r^2)(1-z^2)\tilde{\omega}_k(r,z)$.

Case 2: $|k| = 1$. We continue to represent $\tilde{\omega}_k(r,z)$ in the expansion (4.9). When $|k| = 1$, we find that $\tilde{\omega}_k(r,z)$ satisfies the following partial differential equation:

$$\underbrace{\left[r(1-r^2)\frac{\partial^2\tilde{\omega}_k}{\partial r^2} + (3-7r^2)\frac{\partial\tilde{\omega}_k}{\partial r} - 8r\tilde{\omega}_k\right]}_{=\mathcal{L}_3}(1-z^2)$$
$$+ r(1-r^2)\left[(1-z^2)\frac{\partial^2\tilde{\omega}_k}{\partial z^2} - 4z\frac{\partial\tilde{\omega}_k}{\partial z} - 2\tilde{\omega}_k\right] = \tilde{f}_k.$$

We can discretize this as

$$L_3Y^{(k)}M_{1-r^2}^T + M_rM_{1-r^2}Y^{(k)}D = F_k$$

[8] A matrix is *near-normal* if the condition number of its eigenvector matrix is small.

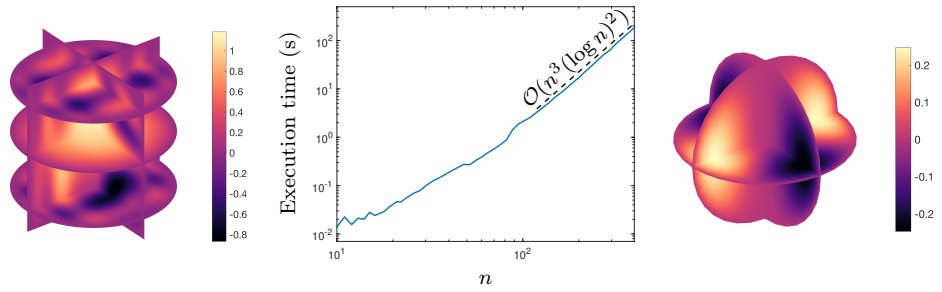


FIG. 6. *Left: A computed solution to Poisson's equation on the cylinder, shown on various slices through the cylinder. The right-hand side f is such that the exact solution is $u(x,y,z) = (1-x^2-y^2)(1-z^2)(z\cos 4\pi x^2 + \cos 4\pi yz)$. Middle: Execution times for the Poisson solver on the cylinder with an error tolerance of $\epsilon = 10^{-13}$. Right: A computed solution to Poisson's equation on the solid sphere, shown on various slices through the sphere. The right-hand side f is such that the exact solution is $u(r,\theta,\phi) = (1-r^2)(r\sin\phi)^2 e^{\mathrm{i}2\theta}$.*

and solve the Bartels–Stewart algorithm, costing $\mathcal{O}(n^3)$ operations. Since there are only two Fourier modes with $|k| = 1$, this does not dominate the overall computational complexity of the Poisson solver. We recover $\tilde{u}_k(r,z)$ via the relation $\tilde{u}_k(r,z) = r(1-r^2)(1-z^2)\tilde{\omega}_k(r,z)$.

Case 3: $k = 0$**.** Finally, the zero Fourier mode satisfies $\tilde{u}_0(r,z) = (1-r^2)(1-z^2)\tilde{\omega}_0(r,z)$ where

$$\underbrace{\left[r^2(1-r^2)\frac{\partial^2\tilde{\omega}_0}{\partial r^2} + (1-5r^2)r\frac{\partial\tilde{\omega}_0}{\partial r} - 4r^2\tilde{\omega}_0\right]}_{=\mathcal{L}_4}(1-z^2)$$
$$+ r^2(1-r^2)\left[(1-z^2)\frac{\partial^2\tilde{\omega}_0}{\partial z^2} - 4z\frac{\partial\tilde{\omega}_0}{\partial z} - 2\tilde{\omega}_0\right] = r^2\tilde{f}_0.$$

We can discretize this as $L_4 Y^{(0)} M_{1-r^2}^T + M_{r^2} M_{1-r^2} Y^{(0)} D = M_{r^2} F_0$ and solve using the Bartels–Stewart algorithm, costing $\mathcal{O}(n^3)$ operations. Again, this cost is negligible since there is only one Fourier mode with $k = 0$.

Figure 6 shows a computed solution to Poisson's equation on the cylinder using this algorithm and confirms the optimal complexity of the resulting solver. Our Poisson solver on the cylinder can be accessed in [11] via the command `poisson_cylinder(F, tol)`, where `F` is the tensor of trivariate Chebyshev–Fourier–Chebyshev coefficients for the doubled-up right-hand side and `tol` is the error tolerance.

4.2. A fast spectral Poisson solver on the solid sphere. Consider Poisson's equation on the unit ball, i.e., $u_{xx} + u_{yy} + u_{zz} = f$ on $x^2 + y^2 + z^2 \in [0,1]$ with homogeneous Dirichlet conditions. Our first step is to change to the spherical coordinate system, i.e., $(x,y,z) = (r\cos\theta\sin\phi, r\sin\theta\sin\phi, r\cos\phi)$ where $r \in [0,1]$ is the radial variable, $\theta \in [-\pi,\pi]$ is the azimuthal variable, and $\phi \in [0,\pi]$ is the polar variable. This change of variables transforms Poisson's equation to

$$\frac{\partial^2 u}{\partial r^2} + \frac{2}{r}\frac{\partial u}{\partial r} + \frac{1}{r^2}\frac{\partial^2 u}{\partial \phi^2} + \frac{\cos\phi}{r^2\sin\phi}\frac{\partial u}{\partial \phi} + \frac{1}{r^2\sin^2\phi}\frac{\partial^2 u}{\partial \theta^2} = f \tag{4.11}$$

for $(r,\theta,\phi) \in [0,1]\times[-\pi,\pi]\times[0,\pi]$, where $u(1,\theta,\phi) = 0$ for $(\theta,\phi) \in [-\pi,\pi]\times[0,\pi]$.

Similar to the cylinder, we use the DFS method to double-up u and f in both the r- and ϕ-variables and solve for $\tilde{u}$ over the domain $(r,\theta,\phi) \in [-1,1]\times[-\pi,\pi]\times[-\pi,\pi]$. The doubled-up functions are non-periodic in the r-variable and 2π-periodic in the θ- and ϕ-variables, leading us to seek the coefficients for $\tilde{u}$ in a Chebyshev–Fourier–Fourier expansion:

$$\tilde{u}(r,\theta,\phi) \approx \sum_{k=-n/2}^{n/2-1} \tilde{u}_k(r,\phi) e^{\mathrm{i}k\theta}, \qquad \tilde{u}_k(r,\phi) = \sum_{j=0}^{n-1} \sum_{\ell=-n/2}^{n/2-1} X_{j\ell}^{(k)} T_j(r) e^{\mathrm{i}\ell\phi},$$

where again we have written the expansion in this form because each Fourier mode in θ can be solved for separately.

As in the cylinder case, to ensure smoothness in (x,y,z) on the solid sphere we will impose partial regularity on $\tilde{u}_k(r,\phi)$. Since we have $x = r\cos\theta\sin\phi$ and $y = r\sin\theta\sin\phi$, we know that the kth θ-Fourier mode $\tilde{u}_k(r,\phi)$ must decay like $\mathcal{O}((r\sin\phi)^{|k|})$ as $r\sin\phi \to 0$. Therefore, we impose the partial regularity condition:

$$\tilde{u}_k(r,\phi) = (1-r^2)(r\sin\phi)^{\min(|k|,2)}\tilde{\omega}_k(r,\phi), \qquad -\frac{n}{2} \le k \le \frac{n}{2}-1,$$

and solve for $\tilde{\omega}_k(r,\phi)$. Again, the partial regularity requirement naturally splits into three cases that we treat separately: $|k| \ge 2$, $|k| = 1$, and $k = 0$. If we represent the r-variable of $\tilde{\omega}_k(r,\phi)$ using the $\tilde{C}_i^{(3/2)}$ basis in r, then for $|k| \ge 2$ we obtain n decoupled sparse Sylvester matrix equations with near-normal matrices which we can solve using ADI in $\mathcal{O}(n^2(\log n)^2\log(1/\epsilon))$ operations. For $k = -1, 0, 1$, we use the Bartels–Stewart algorithm to solve the Sylvester equation directly in $\mathcal{O}(n^3)$ operations.

Figure 6 shows a computed solution to Poisson's equation on the solid sphere using this algorithm. Our Poisson solver on the solid sphere can be accessed in [11] via the command `poisson_solid_sphere(F, tol)`, where `F` is the tensor of trivariate Chebyshev–Fourier–Fourier coefficients for the doubled-up right-hand side and `tol` is the error tolerance.

5. A fast spectral Poisson solver on the cube. Consider Poisson's equation on the cube with homogeneous Dirichlet conditions:

(5.1)
$$u_{xx} + u_{yy} + u_{zz} = f, \quad (x,y,z) \in [-1,1]^3, \quad u(\pm1,\cdot,\cdot) = u(\cdot,\pm1,\cdot) = u(\cdot,\cdot,\pm1) = 0$$

From section 3, we can discretize (5.1) as

$$(D_{xx} + D_{yy} + D_{zz})\,\mathrm{vec}(X) = \mathrm{vec}(F), \tag{5.2}$$

where $X, F \in \mathbb{C}^{n\times n\times n}$, $D_{xx} = A\otimes A\otimes I$, $D_{yy} = A\otimes I\otimes A$, and $D_{zz} = I\otimes A\otimes A$. Here, $A = D^{-1}M$ is the pentadiagonal matrix from section 3, I is the $n\times n$ identity matrix, '$\otimes$' is the Kronecker product, and $\mathrm{vec}(\cdot)$ is the vectorization operator.

Unlike for the cylinder and sphere, there is no decoupling that allows us to reduce the three-term equation into n two-term equations. Therefore, we would like to solve (5.2) using a generalization of the ADI method without constructing the large Kronecker product matrices; however, it is unclear how to generalize ADI to handle more than two terms at a time [35, p. 31]. Instead, we employ the nested ADI method described in [34]. This simply involves grouping the first two terms together and performing the ADI-like iteration given by

$$(D_{zz} - p_{i,1}I)\,\mathrm{vec}(X_{i+1/2}) = \mathrm{vec}(F) - ((D_{xx}+D_{yy}) - p_{i,1}I)\,\mathrm{vec}(X_i) \tag{5.3}$$

$$((D_{xx}+D_{yy}) - q_{i,1}I)\,\mathrm{vec}(X_{i+1}) = \mathrm{vec}(F) - (D_{zz} - q_{i,1}I)\,\mathrm{vec}(X_{i+1/2}) \tag{5.4}$$

for suitable choices of the shift parameters $p_{i,1}$ and $q_{i,1}$. Since the matrices D_{xx}, D_{yy}, and D_{zz} are Kronecker products involving two copies of A and the identity matrix, it can be shown that the eigenvalue bounds on D_{xx}, D_{yy}, and D_{zz} are the same as in section 3, but squared. Thus, we require $\mathcal{O}(\log n)$ iterations of (5.3)–(5.4).

To solve the two-term equation (5.4), we can apply a nested ADI iteration to the matrices $D_{xx} - \frac{q_{i,1}}{2} I$ and $D_{yy} - \frac{q_{i,1}}{2} I$ as follows:

$$\left(\left(D_{xx} - \tfrac{q_{i,1}}{2} I\right) - p_{j,2} I\right) \operatorname{vec}(Y_{j+1/2}) = F_i - \left(\left(D_{yy} - \tfrac{q_{i,1}}{2} I\right) - p_{j,2} I\right) \operatorname{vec}(Y_j) \tag{5.5}$$

$$\left(\left(D_{yy} - \tfrac{q_{i,1}}{2} I\right) - q_{j,2} I\right) \operatorname{vec}(Y_{j+1}) = F_i - \left(\left(D_{xx} - \tfrac{q_{i,1}}{2} I\right) - q_{j,2} I\right) \operatorname{vec}(Y_{j+1/2}) \tag{5.6}$$

where $F_i = \operatorname{vec}(F) - (D_{zz} - q_{i,1} I) \operatorname{vec}(X_{i+1/2})$. After the iteration converges, the solution to (5.4) is obtained as $X_{i+1} := Y_{j+1}$. For the optimal choices of $p_{j,2}$ and $q_{j,2}$ (see section 2) we expect (5.5)–(5.6) to converge in $\mathcal{O}(\log n)$ iterations.

Finally, we are left with solving the three linear systems (5.3), (5.5), and (5.6), which each involve a shifted Kronecker system. Each Kronecker system is actually degenerate in one dimension, due to the presence of the identity matrix. Thus, we can decouple (5.3), (5.5), and (5.6) along that degenerate dimension and solve n decoupled systems independently. For example, to solve (5.3) for $X_{i+1/2}$ we solve

$$A X_{i+1/2}(:,:,k) A^T - p_{i,1} X_{i+1/2}(:,:,k) = F_i(:,:,k), \qquad 1 \le k \le n, \tag{5.7}$$

where $X(:,:,k)$ denotes the kth slice of the tensor X in the z-dimension and $F_i = \operatorname{vec}(F) - ((D_{xx} + D_{yy}) - p_{i,1} I) \operatorname{vec}(X_i)$. To solve each of the decoupled systems (5.7), we can perform yet another nested ADI iteration. If we rewrite (5.7) in the form

$$p_{i,1} A^{-1} X_{i+1/2}(:,:,k) - X_{i+1/2}(:,:,k) A^T = A^{-1} F_i(:,:,k)$$

then the iteration for each k becomes

$$Z_{\ell+1/2}(A^T - p_{\ell,3} I) = A^{-1} F_i(:,:,k) - (p_{i,1} A^{-1} - p_{\ell,3} I) Z_\ell \tag{5.8}$$

$$(p_{i,1} I - q_{\ell,3} A) Z_{\ell+1} = A F_i(:,:,k) - A Z_{\ell+1/2}(A^T - q_{\ell,3} I). \tag{5.9}$$

After the iteration converges, the solution to (5.3) is obtained for each k as $X_{i+1/2}(:,:,k) := Z_{\ell+1}$. Note that we have multiplied (5.9) by A so that (5.8)–(5.9) can be solved fast. For suitable choices of $p_{\ell,3}$ and $q_{\ell,3}$, this will converge in $\mathcal{O}(\log n)$ iterations. Thus, as in section 3, each of the n decoupled equations can be solved in $\mathcal{O}(n^2 \log n)$ operations, allowing (5.3), (5.5), and (5.6) to be solved in $\mathcal{O}(n^3 \log n)$ operations. Since there are two levels of nested ADI iterations above this inner computation, the solution to (5.1) requires $\mathcal{O}(n^3 (\log n)^3 \log(1/\epsilon))$ operations.

Figure 7 shows a computed solution to Poisson's equation on the cube using this algorithm and confirms the optimal complexity of the resulting solver. We stress that though this is observed to be an optimal complexity spectral method to solve (5.1), it is far from a practical algorithm; the inner ADI iterations must be solved to machine precision to assure that the outer iterations will converge, resulting in large algorithmic constants that dominate for realistic choices of n. As in section 3, the solver can also be extended to general box-shaped domains. Our Poisson solver on the cube can be accessed in [11] via the command `poisson_cube(F, tol)`, where `F` is the tensor of trivariate Chebyshev coefficients for the right-hand side and `tol` is the error tolerance.

6. Nontrivial boundary conditions. So far we have assumed zero homogeneous Dirichlet boundary conditions. We now describe how to extend our method to handle more general boundary conditions.

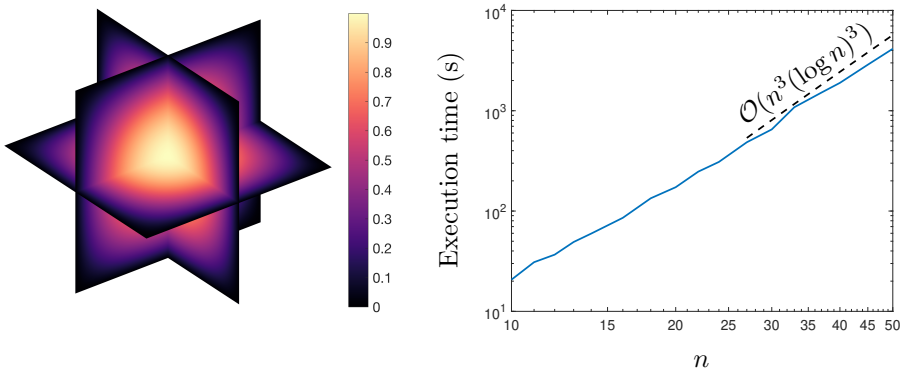


FIG. 7. *Left: A computed solution to Poisson's equation on the cube, shown on various slices through the cube. The right-hand side f is such that the exact solution is $u(x,y,z) = (1-x^2)(1-y^2)(1-z^2)\cos(xyz^2)$. Right: Execution times for the Poisson solver on the cube with an error tolerance of $\epsilon = 10^{-13}$.*

6.1. Nonhomogeneous Dirichlet conditions. To extend our solver to handle nonhomogeneous Dirichlet conditions, we convert the nonhomogeneous problem into a homogeneous one by moving the boundary conditions to the right-hand side. That is,

1. Compute the coefficients X_{bc} of a function u_{bc} satisfying the Dirichlet data but not necessarily satisfying Poisson's equation.
2. Compute the Laplacian of u_{bc}.
3. Solve the modified equation $\nabla^2 u_{\text{rhs}} = f - \nabla^2 u_{\text{bc}}$ with zero homogeneous Dirichlet boundary conditions for the coefficients X_{rhs}.
4. The original solution is then obtained as $X = X_{\text{rhs}} + X_{\text{bc}}$.

Note that the above steps are in coefficient space and can be done fast. This treatment of Dirichlet conditions works for any of the domains discussed in this paper.

6.2. Neumann and Robin. For Neumann or Robin boundary conditions we must abandon bases containing $(1-x^2)$ factors and employ a more general discretization scheme. The ultraspherical spectral method [24, 29] discretizes linear PDEs by generalized Sylvester matrix equations with sparse, well-conditioned matrices and can handle boundary conditions in the form of general linear constraints. For Poisson's equation with Neumann or Robin boundary conditions, the method results in a two-term Sylvester equation with pentadiagonal matrices except for a few dense rows. Experiments indicate that the eigenvalues of the matrices lie within disjoint intervals similar to those in section 3, but this is not theoretically justified. However, in practice we observe that applying the ADI method to these Sylvester matrix equations computes a solution in an optimal number of operations.

Acknowledgments. We are grateful to Heather Wilber, Grady Wright, Marcus Webb, Mikaël Slevinsky, Ricardo Baptista, and Chris Rycroft for their detailed comments on a draft of the paper. Grady Wright wrote the code for Figure 5. We have also benefited from discussions with Sheehan Olver, Gil Strang, and Nick Trefethen.

REFERENCES

[1] A. Averbuch, M. Israeli, and L. Vozovoi, *A fast Poisson solver of arbitrary order accuracy in rectangular regions*, SIAM J. Sci. Comput., 19 (1998), pp. 933–952, https://doi.org/10.1137/S1064827595288589.

[2] R. H. Bartels and G. W. Stewart, *Solution of the matrix equation $AX + XB = C$*, Commun. ACM, 15 (1972), pp. 820–826, https://doi.org/10.1145/361573.361582.

[3] B. Beckermann and A. Townsend, *On the singular values of matrices with displacement structure*, to appear in SIAM J. Mat. Anal. Appl., (2017), https://arxiv.org/abs/1609.09494.

[4] P. Benner, R.-C. Li, and N. Truhar, *On the ADI method for Sylvester equations*, J. Comput. Appl. Math., 233 (2009), pp. 1035–1045, https://doi.org/10.1016/j.cam.2009.08.108.

[5] J. P. Boyd, *Chebyshev and Fourier Spectral Methods*, Courier Corporation, 2001.

[6] E. Braverman, M. Israeli, A. Averbuch, and L. Vozovoi, *A fast 3D Poisson solver of arbitrary order accuracy*, J. Comput. Phys., 144 (1998), pp. 109–136, https://doi.org/10.1006/jcph.1998.6001.

[7] V. Britanak, P. C. Yip, and K. R. Rao, *Discrete Cosine and Sine Transforms: General Properties, Fast Algorithms and Integer Approximations*, Academic Press, 2010.

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

[9] B. N. Datta, *Numerical Linear Algebra and Applications*, SIAM, Philadelpha, PA, 2nd ed., 2010.

[10] T. A. Driscoll, N. Hale, and L. N. Trefethen, eds., *Chebfun Guide*, Pafnuty Publications, Oxford, 2014.

[11] D. Fortunato and A. Townsend. GitHub repository, 2017, https://github.com/danfortunato/fast-poisson-solvers.

[12] S. Gershgorin, *Über die abgrenzung der eigenwerte einer matrix*, Bulletin de l'Académie des Sciences de l'URSS, 6 (1931), pp. 749–754.

[13] A. Gholami, D. Malhotra, H. Sundar, and G. Biros, *FFT, FMM, or multigrid? a comparative study of state-of-the-art Poisson solvers for uniform and nonuniform grids in the unit cube*, SIAM J. Sci. Comput., 38 (2016), pp. C280–C306, https://doi.org/10.1137/15M1010798.

[14] L. Greengard and J. Lee, *A direct adaptive Poisson solver of arbitrary order accuracy*, J. Comput. Phys., 125 (1996), pp. 415–424, https://doi.org/10.1006/jcph.1996.0103.

[15] D. B. Haidvogel and T. Zang, *The accurate solution of Poisson's equation by expansion in Chebyshev polynomials*, J. Comput. Phys., 30 (1979), pp. 167–180, https://doi.org/10.1016/0021-9991(79)90097-4.

[16] P. Henrici, *Fast Fourier methods in computational complex analysis*, SIAM Review, 21 (1979), pp. 481–527, https://doi.org/10.1137/1021093.

[17] V. I. Lebedev, *On a Zolotarev problem in the method of alternating directions*, USSR Comput. Math. Math. Phys., 17 (1977), pp. 58–76, https://doi.org/10.1016/0041-5553(77)90036-2.

[18] R. J. LeVeque, *Finite Difference Methods for Ordinary and Partial Differential Equations*, SIAM, Philadelpha, PA, 2007, https://doi.org/10.1137/1.9780898717839.

[19] A. Lu and E. L. Wachspress, *Solution of Lyapunov equations by alternating direction implicit iteration*, Comput. Math. Appl., 21 (1991), pp. 43–58, https://doi.org/10.1016/0898-1221(91)90124-M.

[20] P. G. Martinsson, V. Rokhlin, and M. Tygert, *A fast algorithm for the inversion of general Toeplitz matrices*, Comput. Math. Appl., 50 (2005), pp. 741–751, https://doi.org/10.1016/j.camwa.2005.03.011.

[21] A. McKenney, L. Greengard, and A. Mayo, *A fast Poisson solver for complex geometries*, J. Comput. Phys., 118 (1995), pp. 348–355, https://doi.org/10.1006/jcph.1995.1104.

[22] P. E. Merilees, *The pseudospectral approximation applied to the shallow water equations on a sphere*, Atmosphere, 11 (1973), pp. 13–20, https://doi.org/10.1080/00046973.1973.9648342.

[23] F. W. J. Olver, D. W. Lozier, R. F. Boisvert, and C. W. Clark, *NIST Handbook of Mathematical Functions*, Cambridge University Press, New York, NY, 2010.

[24] S. Olver and A. Townsend, *A practical framework for infinite-dimensional linear algebra*, in Proceedings of the 1st First Workshop for High Performance Technical Computing in Dynamic Languages, 2014, pp. 57–69, https://doi.org/10.1109/HPTCDL.2014.10.

[25] D. W. Peaceman and J. H. H. Rachford, *The numerical solution of parabolic and elliptic differential equations*, J. SIAM, 3 (1955), pp. 28–41, https://doi.org/10.1137/0103003.

[26] R. B. Platte, L. N. Trefethen, and A. B. Kuijlaars, *Impossibility of fast stable approximation of analytic functions from equispaced samples*, SIAM Review, 53 (2011), pp. 308–318, https://doi.org/10.1137/090774707.

[27] J. Sabino, *Solution of large-scale Lyapunov equations via the block modified Smith method*, PhD thesis, Rice University, 2007.

[28] D. J. Torres and E. A. Coutsias, *Pseudospectral solution of the two-dimensional Navier–Stokes equations in a disk*, SIAM J. Sci. Comput., 21 (1999), pp. 378–403, https://doi.org/10.1137/S1064827597330157.
[29] A. Townsend and S. Olver, *The automatic solution of partial differential equations using a global spectral method*, J. Comput. Phys., 299 (2015), pp. 106–123, https://doi.org/10.1016/j.jcp.2015.06.031.
[30] A. Townsend and L. N. Trefethen, *An extension of Chebfun to two dimensions*, SIAM J. Sci. Comput., 35 (2013), pp. C495–C518, https://doi.org/10.1137/130908002.
[31] A. Townsend, M. Webb, and S. Olver, *Fast polynomial transforms based on Toeplitz and Hankel matrices*, to appear in Math. Comput., (2017), https://arxiv.org/abs/1604.07486.
[32] A. Townsend, H. Wilber, and G. B. Wright, *Computing with functions in spherical and polar geometries I. The sphere*, SIAM J. Sci. Comput., 38 (2016), pp. C403–C425, https://doi.org/10.1137/15M1045855.
[33] L. N. Trefethen, *Spectral Methods in MATLAB*, SIAM, Philadelphia, PA, 2000, https://doi.org/10.1137/1.9780898719598.
[34] E. Wachspress, *Three-variable alternating-direction-implicit iteration*, Computers & Mathematics with Applications, 27 (1994), pp. 1–7, https://doi.org/10.1016/0898-1221(94)90040-X.
[35] E. Wachspress, *The ADI Model Problem*, Springer, New York, NY, 2013, https://doi.org/10.1007/978-1-4614-5122-8.
[36] J. A. C. Weideman and L. N. Trefethen, *The eigenvalues of second-order spectral differentiation matrices*, SIAM J. Numer. Anal., 25 (1988), pp. 1279–1298, https://doi.org/10.1137/0725072.
[37] H. Wilber, *Numerical computing with functions on the sphere and disk*, master's thesis, Boise State University, 2016.
[38] H. Wilber, A. Townsend, and G. B. Wright, *Computing with functions in spherical and polar geometries II. The disk*, SIAM Journal on Scientific Computing, 39 (2017), pp. C238–C262, https://doi.org/10.1137/16M1070207.
[39] E. Zolotarev, *Application of elliptic functions to questions of functions deviating least and most from zero*, Zap. Imp. Akad. Nauk. St. Petersburg, 30 (1877), pp. 1–59.

Appendix A. MATLAB code to compute ADI shifts. Below we provide the MATLAB code that we use to compute the ADI shifts in (2.4). Readers may notice that in (2.4) the arguments of the complete elliptic integral and Jacobi elliptic functions involve $\sqrt{1-1/\alpha^2}$, while the arguments in the code involve $1-1/\alpha^2$, i.e., square roots are missing in the code. This is an esoteric MATLAB convention of the `ellipke` and `ellipj` commands, which we believe is for numerical accuracy. If one attempts to rewrite our code in another programming language, then one needs to be careful about the conventions in the analogues of the `ellipke` and `ellipj` commands.

```
function [p, q] = ADIshifts(a, b, c, d, tol)
% ADISHIFTS  ADI shifts for AX-XB=F when the eigenvalues of A (B) are in [a,b] and
% the eigenvalues of B (A) are in [c,d]. WLOG, we require that a<b<c<d and 0<tol<1.

gam = (c-a)*(d-b)/(c-b)/(d-a);                  % Cross-ratio of a,b,c,d
% Calculate Mobius transform T:{-alp,-1,1,alp}->{a,b,c,d} for some alp:
alp = -1 + 2*gam + 2*sqrt(gam^2-gam);           % Mobius exists with this t
A = det([-a*alp a 1; -b b 1 ; c c 1]);          % Determinant formulae for Mobius
B = det([-a*alp -alp a; -b -1 b ; c 1 c]);
C = det([-alp a 1; -1 b 1 ; 1 c 1]);
D = det([-a*alp -alp 1; -b -1 1; c 1 1]);
T = @(z) (A*z+B)./(C*z+D);                      % Mobius transfom
J = ceil( log(16*gam)*log(4/tol)/pi^2 );        % No. of ADI iterations
if ( alp < 1e7 )
    K = ellipke( 1-1/alp^2 );                   % ADI shifts for [-1,-1/t]&[1/t,1]
    [~, ~, dn] = ellipj((1/2:J-1/2)*K/J,1-1/alp^2);
else                                            % Prevent underflow when alp large
    K = (2*log(2)+log(alp)) + (-1+2*log(2)+log(alp))/alp^2/4;
    m1 = 1/alp^2;
    u = (1/2:J-1/2)*K/J;
    dn = sech(u) + .25*m1*(sinh(u).*cosh(u)+u).*tanh(u).*sech(u);
end
p = T( -alp*dn ); q = T( alp*dn );              % ADI shifts for [a,b]&[c,d]
end
```

Appendix B. Bounding eigenvalues using Gershgorin's circle theorem.

In section 3 a spectral discretization of Poisson's equation on the square is derived as $\tilde{A}X - X\tilde{B} = F$, where $\tilde{A}$ is a real symmetric pentadiagonal matrix and $\tilde{B} = -\tilde{A}^T$. Here, we prove that P2 holds for the Sylvester matrix equation by showing that $\sigma(\tilde{A}) \in [-1, -1/(30n^4)]$. Our main tool is Gershgorin's circle theorem [12].

THEOREM B.1 (Gershgorin). *Let $A \in \mathbb{C}^{n\times n}$ and $\lambda(A)$ be an eigenvalue of A. Then, for some $1 \le i \le n$, we have*

$$\lambda(A) \in \left\{ z \in \mathbb{C} : |z - A_{ii}| \le \sum_{\substack{j=1 \\ j\neq i}}^{n} |A_{ij}| \right\}.$$

The bound on the spectrum of $\tilde{A}$ is stated in the following lemma, which we use to determine the number of ADI iterations for our fast Poisson solver on the square.

LEMMA B.2. *Let $\tilde{A} \in \mathbb{C}^{n\times n}$ be the matrix given in (3.9). Then,*

$$\sigma(\tilde{A}) \subset \left[-1, -\frac{1}{30n^4}\right], \tag{B.1}$$

where $\sigma(\tilde{A})$ is the spectrum of $\tilde{A}$.

Proof. If $n = 1$, then $\tilde{A} = -2/5$ and (B.1) trivially holds. For the remainder of the proof we assume that $n > 1$. Moreover, $\tilde{A}$ is a real symmetric matrix so we know that $\sigma(\tilde{A}) \subset \mathbb{R}$.

To apply Theorem B.1 we need the entries of $\tilde{A}$. In section 3, $\tilde{A}$ is defined as the symmetric matrix such that $\tilde{A} = D_s^{-1} A D_s$ for some diagonal matrix D_s. We have analytical formulas for the entries of A, and can therefore derive the diagonal entries of D_s. Hence, we can write down explicit expressions for the entries of $\tilde{A}$.[9]

Since $\tilde{A}$ is a pentadiagonal matrix with zero sub- and super-diagonals, the even and odd entries of the matrix decouple. That is,

$$\tilde{A} = P^{-1} \begin{bmatrix} \tilde{A}_{\mathrm{e,e}} & 0 \\ 0 & \tilde{A}_{\mathrm{o,o}} \end{bmatrix} P, \qquad P = \begin{bmatrix} I_{\mathrm{e},:} \\ I_{\mathrm{o},:} \end{bmatrix},$$

where I is the identity matrix and "e" and "o" denote the even- and odd-indexed entries, respectively. The decoupling means that $\sigma(\tilde{A}) = \sigma(\tilde{A}_{\mathrm{e,e}}) \cup \sigma(\tilde{A}_{\mathrm{o,o}})$ and (B.1) follows from bounding $\sigma(\tilde{A}_{\mathrm{e,e}})$ and $\sigma(\tilde{A}_{\mathrm{o,o}})$ separately.

Since two similar matrices have the same eigenvalues, we know that $\sigma(\tilde{A}_{\mathrm{e,e}}) = \sigma(S^{-1}\tilde{A}_{\mathrm{e,e}}S)$ and $\sigma(\tilde{A}_{\mathrm{o,o}}) = \sigma(S^{-1}\tilde{A}_{\mathrm{o,o}}S)$ for the diagonal matrix S with $S_{ii} = i$. By applying Theorem B.1 to $S^{-1}\tilde{A}_{\mathrm{e,e}}S$ and $S^{-1}\tilde{A}_{\mathrm{o,o}}S$, we can calculate explicit formulas for bounds on the maximum and minimum eigenvalues of $\tilde{A}_{\mathrm{e,e}}$ and $\tilde{A}_{\mathrm{o,o}}$. We obtain simplified bounds on these formulas by doing a Taylor series expansion about $n = \infty$ and using Taylor's theorem to bound the truncation error.[10] We find that

$$\lambda_{\max}(\tilde{A}) < -\frac{3}{64n^4} + \frac{1}{64n^5} < -\frac{1}{30n^4}, \qquad \lambda_{\min}(\tilde{A}) > -1,$$

where $\lambda_{\max}(\tilde{A})$ and $\lambda_{\min}(\tilde{A})$ denote the maximum and minimum eigenvalue of $\tilde{A}$, respectively. ∎

[9] We omit the formulas for the entries because they are cumbersome, and instead use *Mathematica* to perform the algebraic manipulations. The *Mathematica* code is publicly available [11].

[10] Again, we use *Mathematica* to perform the Taylor expansion and to bound the truncation error. In particular, we employ *Mathematica*'s symbolic inequality solver to verify the stated bounds.