IMA Journal of Numerical Analysis (2025) **00**, 1–30
https://doi.org/10.1093/imanum/draf102

Quasi-optimal complexity *hp*-FEM for the Poisson equation on a rectangle

Kars Knook
Mathematical Institute, University of Oxford, Oxford, OX2 6GG, UK

Sheehan Olver*
Department of Mathematics, Imperial College, London, SW7 2AZ, UK
*Corresponding author: s.olver@imperial.ac.uk

and

Ioannis P. A. Papadopoulos
Weierstrass Institute for Applied Analysis and Stochastics, Berlin, 10117, Germany

[Received on 14 May 2024; revised on 28 April 2025]

We show, in one dimension, that an *hp*-Finite Element Method (*hp*-FEM) discretization can be solved in optimal complexity because the discretization has a special sparsity structure that ensures that the *reverse Cholesky factorization*—Cholesky starting from the bottom right instead of the top left—remains sparse. Moreover, computing and inverting the factorization may parallelize across different elements. By incorporating this approach into an Alternating Direction Implicit method (Fortunato D. and Townsend A. (2020) Fast Poisson solvers for spectral methods. *IMA J. Numer. Anal.*, **40**, 1994–2018) we can solve, within a prescribed tolerance, an *hp*-FEM discretization of the (screened) Poisson equation on a rectangle with quasi-optimal complexity: $O(N^2 \log N)$ operations where N is the maximal total degrees of freedom in each dimension. When combined with fast Legendre transforms we can also solve nonlinear time-evolution partial differential equations (PDEs) in a quasi-optimal complexity of $O(N^2 \log^2 N)$ operations, which we demonstrate on the (viscid) Burgers' equation. We also demonstrate how the solver can be used as an effective preconditioner for PDEs with variable coefficients, including coefficients that support a singularity.

Keywords: *hp*-FEM; Alternating Direction Implicit method; fast direct solvers; quasi-optimal complexity.

1. Introduction

Consider the classic problem of solving the (screened) Poisson equation in a rectangle:

$$-\Delta u(x,y) + \omega^2 u(x,y) = f(x,y) \quad \text{for} \quad a \le x \le b,\ c \le y \le d, \tag{1.1}$$

where $\Delta u := u_{xx} + u_{yy}$ is the Laplacian and we assume vanishing Dirichlet or Neumann boundary conditions. An effective and fast approach to solving this equation is the Fast Poisson Solver: using finite-differences to discretize the partial differential equation (PDE) we can diagonalize the discretization using the Discrete Cosine Transform in a way that leads to quasi-optimal complexity, that is, $O(N^2 \log N)$ operations where N is the maximal degrees of freedom along each dimension.

In this paper we introduce an alternative approach that also achieves quasi-optimal complexity, but for a high-order (*hp*) framework. We utilize the work of Babuška and Suri (Babuška *et al.*, 1991), which introduced a basis for the Finite Element Method (FEM) built from tensor products of piecewise

integrated Legendre polynomials that achieved sparse discretizations for constant coefficient PDEs on rectangles. A fact that, perhaps, has been inadequately emphasized is that this approach enables quasi-optimal[1] application of the discretization when combined with fast Legendre transforms (Alpert & Rokhlin, 1991; Keiner, 2011; Townsend *et al.*, 2018): the complexity is $O(p^2n^2 \log^2 p)$ for a discretization of a tensor product of p-degree polynomials where the rectangle is subdivided into n^2 rectangles (that is, $h = 1/n$ on the unit rectangle). Applying the discretization quasi-optimally in an iterative framework is, therefore, a solved problem.

Inverting the discretization is another story. While Fortunato and Townsend (Fortunato & Townsend, 2020) introduced the first spectral[2] Fast Poisson Solver, which achieves quasi-optimal complexity for solving the two-dimensional Poisson equation with the aforementioned basis using the Alternating Direction Implicit (ADI) method, it was only applicable when there was a single element. The aim of this work is to extend their approach to an arbitrary number of elements in a manner that is robust to h- and p-refinement.

The ingredients that made Fortunato & Townsend (2020) successful were as follows:

1. A fast solve for one-dimensional discretizations.
2. Control on the separation of the spectrum from the origin.

For (1) we introduce an optimal complexity hp-FEM solver in one dimension in Section 4 for Symmetric Positive Definite (SPD) problems: the complexity is $O(pn)$, where p is the polynomial degree and n is the number of elements. For (2) we observe that the smallest eigenvalue can be computed in optimal complexity and we prove bounds built on known hp-FEM results that guarantee that it has the needed behaviour to achieve quasi-optimal complexity.

Sparse p- and hp-FEM have a rich history. They can be traced to the work of Szabó (1979), see also (Szabó & Babuška, 2011, chapter 2.5.2) and (Schwab, 1998, chapter 3.1). Extensions to two dimensions were further developed by Babuška and Suri (Babuška *et al.*, 1991) and Beuchler and Schöberl (Beuchler & Schoeberl, 2006), where they construct a p-FEM on quadrilaterals and simplices, respectively. Other works of a similar theme include Dubiner (1991); Pavarino (1993); Babuška & Suri (1994); Schwab (1998); Karniadakis *et al.* (2005); Beuchler & Pillwein (2007); Beuchler *et al.* (2012a,b); Jia *et al.* (2022) and (Snowball & Olver, 2020, Appendix A). The focus of hp-FEM literature is often deriving the necessary frameworks, proving optimal mesh adaptivity strategies, and obtaining exponential convergence rates (Schwab, 1998; Houston *et al.*, 2002).

The literature on fast solvers for the Poisson equation is extensive. To name but a few techniques: fast Fourier transforms, cyclic reductions (Buzbee *et al.*, 1970), fast direct solvers for boundary element and multipole methods (McKenney *et al.*, 1995; Martinsson, 2019), pseudospectral Fourier with polynomial subtraction (Averbuch *et al.*, 1998; Braverman *et al.*, 1998), the fast diagonalization method (Lynch *et al.*, 1964), multigrid methods (Lottes & Fischer, 2005; Schöberl *et al.*, 2008; Gholami *et al.*, 2016; Huismann *et al.*, 2019; Brubeck & Farrell, 2022) and domain decomposition (Martinsson, 2009; Gillman *et al.*, 2012; Gillman & Martinsson, 2014). Almost always there is a trade-off between asymptotic

[1] There are false misconceptions in the literature originating in Orszag (1979) that optimal complexity is $O(p^{d+1})$ operations as $p \to \infty$. We contend that optimal is $O(p^d)$ operations, and indeed our quasi-optimal complexity outperforms the misreported 'optimal' complexity.

[2] 'First' and 'spectral' are up to debate: it is only spectral if the solution itself is smooth. But certainly 'Fast' is an undisputed adjective.

complexity, speed and flexibility of the methods, e.g., the structure required in the mesh. To our knowledge, except for the solver described in this work, there exists no fast Poisson solver in two dimensions that simultaneously (1) converges spectrally when the solution is smooth, (2) can mesh the domain into rectangular elements and, therefore, efficiently capture discontinuities in the data and (3) asymptotically requires only $O(N^2 \log N) = O((pn)^2 \log(pn))$ operations for the solve and $O(N^2 \log^2 N) = O((pn)^2 \log^2(pn))$ operations for the set-up.

REMARK 1.1. There are many other high-performance techniques to solve PDEs on similar meshes as we consider. For moderate choices of p these may outperform the method we propose, and so for many applications our contribution may be mostly theoretical. However, to our knowledge none of the existing work achieves optimal complexity for both h and p refinement, hence for very large p our approach will outperform existing techniques. In particular, multigrid techniques that report $O(p^{d+1})$ flops for the solve (Brubeck & Farrell, 2022) (we achieve $O(p^2 \log p)$ in two dimensions) or h- and p-independent Krylov iteration counts (Schöberl *et al.*, 2008) do not discuss the complexity of the set-up (such as the quadrature for assembling the load vectors). Eventually, the set-up will become a bottleneck in such solvers unlike in our method where the set-up remains quasi-optimal. A thorough investigation of the choice of p where our approach becomes competitive would require a distributed memory implementation in order to effectively parallelize the algorithm, which is beyond the scope of this current paper.

The structure of the paper is as follows:

Section 2: We review the integrated Legendre functions of Babuška *et al.* (1991) (see also (Schwab, 1998, chapter 3.1.4)) and see how they lead to discretizations of differential operators with a special sparsity structure, which we call *Banded-Block-Banded Arrowhead (B^3-Arrowhead) Matrices*.

Section 3: We explain how the Poisson equation can be recast as a simple linear system involving a B^3-Arrowhead matrix in one dimension and a Sylvester equation involving B^3-Arrowhead matrices in two dimensions.

Section 4: We show that a reverse Cholesky factorization—a factorization of a matrix as $A = L^\top L$ where L is lower triangular—for B^3-Arrowhead matrices can be computed and the inverse applied in optimal complexity. Moreover, the factorization potentially parallelizes between different elements.

Section 5: We discuss the ADI method and how it can be used to solve the (screened) Poisson equation in quasi-optimal complexity. This requires spectral analysis of the underlying discretizations to control the number of iterations needed in ADI.

Section 6: We discuss how to transform between coefficients and values efficiently, using visicid Burgers' equation in two dimensions with a discontinuous initial condition as an example.

Section 7: Our solver can be used effectively as a preconditioner for PDEs with variable coefficients, where each iteration achieves quasi-optimal complexity. We demonstrate this on a time-independent Schrödinger equation with a variable coefficient with a singularity, where the number of iterations is roughly independent of h and p.

2. Integrated Legendre functions

In this section we introduce the one-dimensional basis of integrated Legendre functions that underlies our discretization of the Poisson equation.

2.1 *A basis for a single interval*

Define the weighted ultraspherical/Jacobi polynomials[3] as

$$W_k(x) := \frac{(1-x^2)C_k^{(3/2)}(x)}{(k+1)(k+2)} = \frac{(1-x^2)P_k^{(1,1)}(x)}{2(k+1)},$$

where $C_k^{(\lambda)}$ are orthogonal with respect to $(1-x^2)^{\lambda-1/2}$ on $[-1,1]$ for $\lambda > -1/2$, $\lambda \neq 0$ with normalization constant

$$C_k^{(\lambda)}(x) = \frac{2^k(\lambda)_k}{k!}x^k + O(x^{k-1}),$$

where $(\lambda)_k = \lambda(\lambda+1)\cdots(\lambda+k-1)$ is the Pochammer symbol. $P_k^{(a,b)}(x)$ are Jacobi polynomials orthogonal with respect to $(1-x)^a(1+x)^b$ on $[-1,1]$ with normalization constant given in (Olver *et al.*, 18.3).

The choice of normalization is chosen because it leads to the simple formula

$$W_k'(x) = -P_{k+1}(x) \tag{2.1}$$

for the Legendre polynomials $P_k(x) \equiv C_k^{(1/2)}(x)$ (Olver *et al.*, 18.9.16). In other words, they are the integral of Legendre polynomials: up to a constant they are precisely the *integrated Legendre functions* used by Babuška. They are also equivalent to the basis defined by Schwab (Schwab, 1998, chapter 3.1) and utilized by Fortunato and Townsend (Fortunato & Townsend, 2020).

It is convenient to express this relationship in terms of *quasi-matrices*, which can be viewed as matrices that are continuous in the first dimension, or equivalently as a row-vector whose columns are functions:

$$\frac{\mathrm{d}}{\mathrm{d}x}\underbrace{[W_0, W_1, W_2, \ldots]}_{\mathbf{W}} = \underbrace{[P_0, P_1, P_2, \ldots]}_{\mathbf{P}} \underbrace{\begin{bmatrix} 0 & & & & \\ -1 & & & & \\ & -1 & & & \\ & & -1 & & \\ & & & \ddots & \end{bmatrix}}_{D_W^P}.$$

If we have a single element in one dimension we can use this basis as the test and trial basis in the weak formulation of a differential equation. Let $\langle \cdot, \cdot \rangle$ denote the $L^2(-1,1)$-inner product. Then, the Gram/mass matrix associated with Legendre polynomials is

$$\left\langle \mathbf{P}^\top, \mathbf{P} \right\rangle := \begin{bmatrix} \langle P_0, P_0\rangle & \langle P_0, P_1\rangle & \cdots \\ \langle P_1, P_0\rangle & \langle P_1, P_1\rangle & \cdots \\ \vdots & \vdots & \ddots \end{bmatrix} = \underbrace{\begin{bmatrix} 2 & & & \\ & 2/3 & & \\ & & 2/5 & \\ & & & \ddots \end{bmatrix}}_{M_P},$$

[3] This definition is also equal to the ultraspherical polynomial $C_{k+2}^{(-1/2)}(x)$, but to avoid discussion of orthogonal polynomials with nonintegrable weights we do not use this relationship.

while the discretization of the weak one-dimensional Laplacian is diagonal:

$$-\Delta_W := \left\langle (\mathbf{W}')^\top, \mathbf{W}' \right\rangle = \left\langle (\mathbf{P}D_W^P)^\top, \mathbf{P}D_W^P \right\rangle$$
$$= (D_W^P)^\top M_P D_W^P = \begin{bmatrix} 2/3 & & & \\ & 2/5 & & \\ & & 2/7 & \\ & & & \ddots \end{bmatrix},$$

which is another way to write the formula:

$$\langle W_k', W_j' \rangle = \langle P_{k+1}, P_{j+1} \rangle = \frac{2}{2k+3}\delta_{kj}.$$

The mass matrix can be deduced by using the lowering relationship:

$$\underbrace{[W_0, W_1, W_2, \ldots]}_{\mathbf{W}} = \underbrace{[P_0, P_1, P_2, \ldots]}_{\mathbf{P}} \underbrace{\begin{bmatrix} \times & & & & \\ 0 & \times & & & \\ \times & 0 & \times & & \\ & \times & 0 & \times & \\ & & \ddots & \ddots & \ddots \end{bmatrix}}_{L_W}, \tag{2.2}$$

where the exact formulæ for the entries are in Appendix A. For now we focus on sparsity structure. Subsequently, the mass matrix can be expressed as a truncation of an infinite pentadiagonal matrix:

$$M_W := \langle \mathbf{W}^\top, \mathbf{W} \rangle = L_W^\top \langle \mathbf{P}^\top, \mathbf{P} \rangle L_W = L_W^\top M_P L_W$$
$$= \begin{bmatrix} \times & 0 & \times & & & \\ 0 & \times & 0 & \times & & \\ \times & 0 & \times & 0 & \times & \\ & \times & 0 & \times & 0 & \ddots \\ & & \times & 0 & \times & \ddots \\ & & & \ddots & \ddots & \ddots \end{bmatrix}.$$

The entries have simple explicit rational expressions, or alternatively one can view this as a product of banded matrices. The latter approach is slightly less efficient, but we will use it for clarity in exposition. We can similarly find the matrix of the inner products that arise in testing with this basis:

$$\left\langle \mathbf{W}^\top, \mathbf{P} \right\rangle = L_W^\top \left\langle \mathbf{P}^\top, \mathbf{P} \right\rangle = L_W^\top M_P.$$

2.2 *Multiple intervals*

Partitioning an interval $[a, b]$ into n subintervals $a = x_0 < x_1 < \cdots < x_n = b$ we can use an affine map

$$a_j(x) = \frac{2x - x_{j-1} - x_j}{x_j - x_{j-1}}$$

to the reference interval $[-1, 1]$ to construct mapped *bubble functions* as

$$W_{kj}^{\mathbf{x}}(x) := \begin{cases} W_k(a_j(x)) & x \in [x_{j-1}, x_j] \\ 0 & \text{otherwise} \end{cases}$$

on each interval. We combine these with the standard piecewise linear hat basis

$$h_0^{\mathbf{x}}(x) := \begin{cases} \frac{x_1 - x}{x_1 - x_0} & x \in [x_0, x_1], \\ 0 & \text{otherwise}, \end{cases} \qquad h_n^{\mathbf{x}}(x) := \begin{cases} \frac{x - x_{n-1}}{x_n - x_{n-1}} & x \in [x_{n-1}, x_n], \\ 0 & \text{otherwise}, \end{cases}$$

$$h_j^{\mathbf{x}}(x) := \begin{cases} \frac{x - x_{j-1}}{x_j - x_{j-1}} & x \in [x_{j-1}, x_j], \\ \frac{x_{j+1} - x}{x_{j+1} - x_j} & x \in [x_j, x_{j+1}], \\ 0 & \text{otherwise}, \end{cases} \qquad \text{for} \qquad j = 1, \ldots, n-1.$$

The hat and bubble functions are sometimes known as the internal and external shape functions, respectively (Schwab, 1998, definition 3.4). We form a block quasi-matrix by grouping together the hat and bubble functions of the same degree:

$$\mathbf{C}^{\mathbf{x}} := [\underbrace{h_0^{\mathbf{x}}, \ldots, h_n^{\mathbf{x}}}_{\mathbf{H}^{\mathbf{x}}} \mid \underbrace{W_{01}^{\mathbf{x}}, \ldots, W_{0n}^{\mathbf{x}}}_{\mathbf{W}_0^{\mathbf{x}}} \mid \underbrace{W_{11}^{\mathbf{x}}, \ldots, W_{1n}^{\mathbf{x}}}_{\mathbf{W}_1^{\mathbf{x}}} \mid \cdots].$$

We relate this to the piecewise Legendre basis

$$\mathbf{P}^{\mathbf{x}} := [\underbrace{P_{01}^{\mathbf{x}}, \ldots, P_{0n}^{\mathbf{x}}}_{\mathbf{P}_0^{\mathbf{x}}} \mid \underbrace{P_{11}^{\mathbf{x}}, \ldots, P_{1n}^{\mathbf{x}}}_{\mathbf{P}_1^{\mathbf{x}}} \mid \cdots],$$

where

$$P_{kj}^{\mathbf{x}}(x) := \begin{cases} P_k(a_j(x)) & x \in [x_{j-1}, x_j] \\ 0 & \text{otherwise} \end{cases}.$$

In what follows we often omit the dependence on $\mathbf{x}$.

2.2.1 *The mass matrix.* Restricting to each panel our basis is equivalent to a mapped version of the one panel basis defined above, hence we can re-expand $\mathbf{C}^{\mathbf{x}}$ in terms of $\mathbf{P}^{\mathbf{x}}$. First, note that the mass matrix is diagonal, which we write in block form as

$$\left\langle (\mathbf{P}^{\mathbf{x}})^{\top}, \mathbf{P}^{\mathbf{x}} \right\rangle = \underbrace{\begin{bmatrix} M_{11} & & & & \\ & M_{22} & & & \\ & & M_{33} & & \\ & & & \ddots & \end{bmatrix}}_{M_P^{\mathbf{x}}},$$

where $\langle \cdot, \cdot \rangle$ denotes the $L^2(a,b)$-inner product. Since piecewise Legendre polynomials completely decouple we can view this matrix as a direct sum:

$$M_P^{\mathbf{x}} = \left(\frac{x_1 - x_0}{2} M_P \right) \oplus \cdots \oplus \left(\frac{x_n - x_{n-1}}{2} M_P \right),$$

where the direct sum corresponds to interlacing the entries of the matrix, i.e.,

$$\mathbf{e}_\ell^{\top} M_{kj} \mathbf{e}_\ell = \mathbf{e}_k^{\top} \left(\frac{x_\ell - x_{\ell-1}}{2} M_P \right) \mathbf{e}_j.$$

Note that, given a piecewise polynomial f, its coefficients in the basis $\mathbf{P}^{\mathbf{x}}$ can be expressed as

$$(M_P^{\mathbf{x}})^{-1} \left\langle (\mathbf{P}^{\mathbf{x}})^{\top}, f \right\rangle.$$

We use this to determine the (block) connection matrix

$$\mathbf{C} = \mathbf{P} \underbrace{\begin{bmatrix} R_{00} & R_{01} & & & \\ R_{10} & \mathbf{0} & R_{12} & & \\ & R_{21} & \mathbf{0} & R_{23} & \\ & & \ddots & \ddots & \ddots \end{bmatrix}}_{R_C^{\mathbf{x}}},$$

where the blocks are

$$R_{k0} := M_{kk}^{-1} \left\langle \mathbf{P}_k^{\top}, \mathbf{H} \right\rangle = \begin{bmatrix} \times & \times & & \\ & \ddots & \ddots & \\ & & \times & \times \end{bmatrix} \in \mathbb{R}^{n \times (n+1)}, \tag{2.3}$$

$$R_{kj} := M_{kk}^{-1} \left\langle \mathbf{P}_k^{\top}, \mathbf{W}_{j-1} \right\rangle = \begin{bmatrix} \times & & \\ & \ddots & \\ & & \times \end{bmatrix} \in \mathbb{R}^{n \times n}, \quad \text{for} \quad j > 0, \tag{2.4}$$

with the explicit formulæ for the entries given in Appendix A. We thus have the mass matrix

$$M_C^{\mathbf{x}} := \left\langle \mathbf{C}^\top, \mathbf{C} \right\rangle = R_C^\top M_P R_C$$

$$= \left[\begin{array}{ccc|c|c|c|c} \times & \times & & \times & & \times & & \\ \times & \times & \ddots & \times & \ddots & \times & \ddots & \\ & \ddots & \ddots & \times & \ddots & \times & \ddots & \times \\ & & \times & \times & & \times & & \times \\ \hline \times & \times & & \times & & & \times & \\ & \ddots & \ddots & & \ddots & & & \ddots \\ & & \times & \times & & \times & & \times \\ \hline \times & \times & & & & \times & & \\ & \ddots & \ddots & & & & \ddots & \\ & & \times & \times & & & & \times \\ \hline & & & \times & & & \times & \\ & & & & \ddots & & & \ddots \\ & & & & & \times & & \times \\ \hline & & & & & & & \\ & & & & & & & \\ & & & & & & & \end{array}\right],$$

where again the entries have simple rational expressions that can be deduced from the components. The structure is important here: every block is banded, and every block not in the first row or column is diagonal.

2.2.2 *The weak Laplacian.* Similarly, we can express differentiation as a block-diagonal matrix:

$$\frac{\mathrm{d}}{\mathrm{d}x}\mathbf{C} = \mathbf{P} \underbrace{\left[\begin{array}{c|c|c} D_{00} & & \\ \hline & D_{11} & \\ \hline & & \ddots \end{array}\right]}_{D^{\mathbf{x}}},$$

where the blocks are

$$D_{00} := M_{\mathbf{P}_0}^{-1} \left\langle \mathbf{P}_0^\top, \mathbf{H}' \right\rangle = \begin{bmatrix} \times & \times & & \\ & \ddots & \ddots & \\ & & \times & \times \end{bmatrix} \in \mathbb{R}^{n \times (n+1)}, \tag{2.5}$$

$$D_{kj} := M_{\mathbf{P}_k}^{-1} \left\langle \mathbf{P}_k^\top, \mathbf{W}_{j-1}' \right\rangle = \begin{bmatrix} \times & & \\ & \ddots & \\ & & \times \end{bmatrix} \in \mathbb{R}^{n \times n}, \quad \text{for} \quad j > 0, \tag{2.6}$$

with the explicit entries given in Appendix A. We thus deduce that the weak Laplacian is also block diagonal with structure

$$-\Delta_C^{\mathbf{x}} := \left\langle (\mathbf{C}')^\top, \mathbf{C}' \right\rangle = (D)^\top M_P D$$

$$= \left[\begin{array}{ccc|c|c} \times & \times & & & \\ \times & \times & \ddots & & \\ & \ddots & \ddots & \times & \\ & & \times & \times & \\ \hline & & & \times & \\ & & & \ddots & \\ & & & & \times \\ \hline & & & & \ddots \end{array}\right].$$

Again the structure is important: we have a block diagonal matrix whose blocks are all diagonal, apart from the first, which is tridiagonal.

2.3 *Homogeneous Dirichlet boundary condition*

To enforce homogeneous Dirichlet boundary conditions, we need to drop the basis functions that do not vanish at the boundary, that is, the first and last hat function. Thus, we use the following basis:

$$\mathbf{Q}^{\mathbf{x}} := \mathbf{C}^{\mathbf{x}} \underbrace{\left[\begin{array}{ccc|c|c} 0 & & & & \\ 1 & & & & \\ & \ddots & & & \\ & & 1 & & \\ & & 0 & & \\ \hline & & & 1 & \\ & & & \ddots & \\ & & & & 1 \\ \hline & & & & \\ & & & & \ddots \end{array}\right]}_{P_{D,n}}$$

$$= [\underbrace{h_1, \ldots, h_{n-1}}_{\tilde{\mathbf{H}}} \mid \underbrace{W_{01}, \ldots, W_{n1}}_{\mathbf{W}_1} \mid \underbrace{W_{02}, \ldots, W_{n2}}_{\mathbf{W}_2} \mid \cdots].$$

The discretized operators are the same as above, but with the first and last row of the first row/column blocks removed, which modifies the band structure, i.e., we have

$$M_Q^{\mathbf{x}} := \left\langle (\mathbf{Q})^\top, \mathbf{Q} \right\rangle = P_D^\top M_C P_D$$

$$= \left[\begin{array}{cccc|ccc|cc|cc}
\times & \times & & & \times & & & \times & & & \\
\times & \times & \ddots & & \times & \ddots & & \times & \ddots & & \\
 & \ddots & \ddots & \times & & \ddots & \times & & \ddots & \times & \\
 & & \times & \times & & & \times & & & \times & \\
\hline
\times & \times & & & \times & & & & & \times & \\
 & \ddots & \ddots & & & \ddots & & & & & \ddots \\
 & & \times & \times & & & \times & & & \times & \\
\hline
\times & \times & & & & & & \times & & & \\
 & \ddots & \ddots & & & & & & \ddots & & \ddots \\
 & & \times & \times & & & & & & \times & \\
\hline
 & & & & \times & & & & & \times & \\
 & & & & & \ddots & & & & & \ddots \\
 & & & & & & \times & & & \times & \\
\hline
 & & & & & & & & \ddots & & \ddots \\
\end{array}\right].$$

As before, every block is banded, and every block not in the first row or column is diagonal. The primary difference with the $\mathbf{C}^{\mathbf{x}}$ case above is the bandwidths of some of the blocks.

3. Discretizations of the screened Poisson equation

In order to discuss the FEM discretization of (1.1) we first recast it in variational form. Let $\Omega = [a, b] \times [c, d]$ and $H^s(\Omega) := W^{s,2}(\Omega)$, where $W^{s,q}(\Omega), s > 0, q \geq 1$, denote the standard Sobolev spaces (Adams & Fournier, 2003). We use $L^q(\Omega)$, $q \geq 1$, to denote the Lebesgue spaces and the notation $H_0^1(\Omega)$ for the space $\{v \in H^1(\Omega) : v|_{\partial\Omega} = 0\}$, where $|_{\partial\Omega} : H^1(\Omega) \to H^{1/2}(\partial\Omega)$ denotes the usual trace operator (Gagliardo, 1957). Let $H^{-1}(\Omega) = H_0^1(\Omega)^*$ denote the dual space of $H_0^1(\Omega)$. Moreover, given a Banach space X and Hilbert space H $\langle \cdot, \cdot \rangle_{X^*,X}$ denotes the duality pairing between a function in X and a functional in the dual space X^*, and $\langle \cdot, \cdot \rangle_H$ denotes the inner product in H. As in the previous section we drop the subscript when utilizing the $L^2(\Omega)$-inner product. Given an $f \in H^{-1}(\Omega)$ we may rewrite the Poisson equation in variational (weak) form as follows: find $u \in H_0^1(\Omega)$ that satisfies

$$\langle \nabla v, \nabla u \rangle + \omega^2 \langle v, u \rangle = \langle v, f \rangle_{H_0^1(\Omega), H^{-1}(\Omega)} \quad \text{for all} \quad v \in H_0^1(\Omega), \tag{3.1}$$

where $\nabla u = (u_x, u_y)^\top$ is the gradient operator. To construct a discretization in the FEM framework one chooses subspaces of $H^1(\Omega)$ as *trial space* (discretization of u) and *test space* (discretization of v), specified by their bases, which are termed *trial* and *test bases*, respectively.

3.1 *Screened Poisson in one dimension*

For zero Dirichlet problems we use as both the test and trial basis $\mathbf{Q}$ up to degree p:

$$\mathbf{Q}_{0:p} := \mathbf{Q} \underbrace{\begin{bmatrix} I_{n-1} & & & & \\ & I_n & & & \\ & & \ddots & & \\ & & & I_n & \\ & & & 0_{n\times n} & \\ & & & \vdots & \end{bmatrix}}_{I_{0:p}\in\mathbb{R}^{\infty\times N}} = [\tilde{\mathbf{H}}|\mathbf{W}_0|\cdots|\mathbf{W}_{p-2}],$$

where $N = pn - 1$ is the total degrees of freedom in our basis up to degree p. That is, we approximate the solution, for $\mathbf{u}_p \in \mathbb{R}^N$: $u(x) \approx u_p(x) := \mathbf{Q}_{0:p}(x)\mathbf{u}_p$ and represent our test functions as $v_p = \mathbf{Q}_{0:p}\mathbf{v}_p$. The discretization of our weak formulation then becomes

$$\begin{aligned}\left\langle v_p', u_p' \right\rangle + \omega \left\langle v_p, u_p \right\rangle &= \mathbf{v}_p^\top \left\langle ((\mathbf{Q}_{0:p})')^\top, (\mathbf{Q}_{0:p})' \right\rangle \mathbf{u}_p + \omega \mathbf{v}_p^\top \left\langle (\mathbf{Q}_{0:p})^\top, (\mathbf{Q}_{0:p})' \right\rangle \mathbf{u}_p \\ &= \mathbf{v}_p^\top (-\underbrace{I_{0:p}^\top \Delta_Q I_{0:p}}_{\Delta_{Q,p}} + \omega^2 \underbrace{I_{0:p}^\top M_Q I_{0:p}}_{M_{Q,p}}) \mathbf{u}_p.\end{aligned}$$

If we further assume that we have made a piecewise polynomial approximation of our right-hand side as $f \approx f_p := \mathbf{P}_{0:p}\mathbf{u}_p$, computed via a fast Legendre transform (Townsend *et al.* (2018) or otherwise), the right-hand side becomes

$$\begin{aligned}\left\langle v_p, f \right\rangle &= \mathbf{v}_p^\top I_{0:p}^\top \left\langle \mathbf{Q}^\top, \mathbf{P} \right\rangle I_{0:p} \mathbf{f}_p = \mathbf{v}_p^\top I_{0:p}^\top R_Q^\top \left\langle \mathbf{P}^\top, \mathbf{P} \right\rangle I_{0:p} \mathbf{f}_p \\ &= \mathbf{v}_p^\top \underbrace{I_{0:p}^\top R_Q^\top I_{0:p}}_{R_{Q,p}^\top} \underbrace{I_{0:p}^\top M_P I_{0:p}}_{M_{P,p}} \mathbf{f}_p.\end{aligned}$$

Enforcing this equation for all $\mathbf{v}_p \in \mathbb{R}^N$ leads us to an $N \times N$ system of equations:

$$(-\Delta_{Q,p} + \omega^2 M_{Q,p})\mathbf{u}_p = R_{Q,p}^\top M_{P,p} \mathbf{f}_p.$$

3.2 *Screened Poisson in two dimensions*

For two-dimensional problems we consider partitions $a = x_1 < \cdots < x_m = b$ and $c = y_1 < \cdots < y_n = d$ with truncation up to degrees p and q, respectively. Then, our basis for zero Dirichlet problems is given by the tensor product of $\mathbf{Q}^{\mathbf{x}}$ and $\mathbf{Q}^{\mathbf{y}}$; more specifically, we have

$$u(x, y) \approx u_{pq}(x, y) := \mathbf{Q}_{0:p}^{\mathbf{x}}(x) U_{pq} \mathbf{Q}_{0:q}^{\mathbf{y}}(y)^\top,$$

where we represent the unknown coefficients as a matrix $U_{pq} \in \mathbb{R}^{M\times N}$ for $M = (p+1)m - 1$ and $N = (q+1)n - 1$. Similarly, we may express the right-hand side as

$$f(x,y) \approx f_{pq}(x,y) = \mathbf{P}^{\mathbf{x}}_{0:p}(x) F_{pq} \mathbf{P}^{\mathbf{y}}_{0:q}(y)^\top.$$

Consider an arbitrary test function $v_{pq}(x,y) = \mathbf{Q}^{\mathbf{x}}_{0:p}(x) V_{pq} \mathbf{Q}^{\mathbf{y}}_{0:q}(y)^\top$. By substituting in the expressions for u, v and f into (3.1), then akin to the one-dimensional case, we arrive at

$$\begin{aligned}(-\Delta^{\mathbf{x}}_{Q,p}) U_{pq} M^{\mathbf{y}}_{Q,q} + M^{\mathbf{x}}_{Q,p} U_{pq} (-\Delta^{\mathbf{y}}_{Q,q}) + \omega^2 M^{\mathbf{x}}_{Q,p} U_{pq} M^{\mathbf{y}}_{Q,q} \\ = (R^{\mathbf{x}}_{Q,p})^\top M^{\mathbf{x}}_{P,p} F_{pq} M^{\mathbf{y}}_{P,q} R^{\mathbf{y}}_{Q,q}.\end{aligned}$$

We can modify this into a Sylvester's equation:

$$\begin{aligned}\left(-\Delta^{\mathbf{x}}_{Q,p} + \frac{\omega^2}{2} M^{\mathbf{x}}_{Q,p}\right) U_{pq} M^{\mathbf{y}}_{Q,q} + M^{\mathbf{x}}_{Q,p} U_{pq} \left(-\Delta^{\mathbf{y}}_{Q,q} + \frac{\omega^2}{2} M^{\mathbf{y}}_{Q,q}\right) \\ = (R^{\mathbf{x}}_{Q,p})^\top M^{\mathbf{x}}_{P,p} F_{pq} M^{\mathbf{y}}_{P,q} R^{\mathbf{y}}_{Q,q}.\end{aligned} \tag{3.2}$$

In Section 5 we will discuss how to solve (3.2) for U_{pq} in quasi-optimal complexity.

3.3 *General boundary conditions*

Akin to more traditional FEMs our method supports inhomogeneous Robin boundary conditions, $\alpha u + (\nabla u)\cdot n = g$ on $\partial\Omega$, for which zero Neumann conditions $(\nabla u)\cdot n = 0$ is a special case. This is achieved by utilizing the full basis $\mathbf{C}^{\mathbf{x}}$ (or its tensor product in higher dimensions), rather than $\mathbf{Q}^{\mathbf{x}}$, in the discretization of the variational form

$$\langle \nabla v, \nabla u\rangle + \alpha \langle v, u\rangle_{L^2(\partial\Omega)} = \langle v, f\rangle + \langle v, g\rangle_{L^2(\partial\Omega)}. \tag{3.3}$$

Mixed boundary conditions may be handled similarly. For instance setting $\alpha = g = 0$ and including $h^{\mathbf{x}}_0$ in the discretization basis, but not $h^{\mathbf{x}}_n$, imposes a zero Neumann boundary condition on the left and a zero Dirichlet boundary condition on the right.

4. Optimal complexity Cholesky factorization

As noted, the mass matrices M_C/M_Q and weak Laplacians Δ_C/Δ_Q have a special sparsity structure:

Definition 4.1. A *Banded-Block-Banded-Arrowhead (B^3-Arrowhead) Matrix* $A \in \mathbb{R}^{m+pn\times m+pn}$ with block-bandwidths (ℓ, u) and sub-block-bandwidth $\lambda + \mu$ has the following properties:

1. It is a block-banded matrix with block-bandwidths (ℓ, u).
2. The top-left block $A_0 \in \mathbb{R}^{m\times m}$ is banded with bandwidths $(\lambda+\mu, \lambda+\mu)$.
3. The remaining blocks in the first row $B_k \in \mathbb{R}^{m\times n}$ have bandwidths (λ, μ).

4. The remaining blocks in the first column $C_k \in \mathbb{R}^{n\times m}$ have bandwidths (μ, λ).
5. All other blocks $D_{k,j} \in \mathbb{R}^{n\times n}$ are diagonal.

We represent the matrix in block form

$$\begin{bmatrix} A_0 & B \\ C & D \end{bmatrix} = \left[\begin{array}{c|c|c|c|c|c|c} A_0 & B_1 & \cdots & B_u & & & \\ \hline C_1 & D_{1,1} & \cdots & D_{1,u} & D_{1,1+u} & & \\ \hline \vdots & \vdots & \ddots & \ddots & \ddots & \ddots & \\ \hline C_l & D_{\ell,1} & \ddots & \ddots & \ddots & \ddots & D_{p-u,p} \\ \hline & D_{\ell+1,1} & \ddots & \ddots & \ddots & \ddots & D_{p-u+1,p} \\ \hline & & \ddots & \ddots & \ddots & \ddots & \vdots \\ \hline & & & D_{p,p-\ell} & D_{p,p-\ell+1} & \cdots & D_{p,p} \end{array}\right], \tag{4.1}$$

where $A_0, B_1, \ldots, B_u, C_1, \ldots, C_\ell$ are all banded matrices with bandwidths (λ, μ), while $D_{k,j}$ are diagonal matrices. To store the diagonal blocks in D we write

$$D = D_1 \oplus \cdots \oplus D_n,$$

where D_k are banded matrices with bandwidths (ℓ, u), where as above the direct sum corresponds to interlacing the entries of the matrix.

If we apply a Cholesky factorization $A = LL^\top$ directly we will have fill-in coming from the banded initial rows/columns. The key observation is that if we use a *reverse Cholesky factorization*, that is, a factorization of the form $A = L^\top L$, which begins in the bottom right, we avoid fill-in.

THEOREM 4.2. If A is an SPD B^3-Arrowhead Matrix that has block-bandwidth (ℓ, ℓ) and sub-block-bandwidth $\lambda + \mu$ then it has a reverse Cholesky factorization

$$A = L^\top L,$$

where L is a B^3-Arrowhead Matrix with block-bandwidth $(\ell, 0)$ and sub-block-bandwidth $\lambda + \mu$.

Proof. We begin by writing A as a block matrix:

$$A = \begin{bmatrix} A_0 & B \\ B^\top & D \end{bmatrix},$$

where

$$D = D_1 \oplus \cdots \oplus D_n.$$

The reverse Cholesky factorization $D = \tilde{L}^\top \tilde{L}$ can be deduced from the reverse Cholesky factorizations of $D_j = L_j^\top L_j$. In particular, we have that

$$\tilde{L} := L_1 \oplus \cdots \oplus L_n \qquad \text{and} \qquad \tilde{L}^{-1} = L_1^{-1} \oplus \cdots \oplus L_n^{-1}.$$

Now, write

$$A = \begin{bmatrix} A_0 & B \\ B^\top & \tilde{L}^\top \tilde{L} \end{bmatrix} = \begin{bmatrix} I & B\tilde{L}^{-1} \\ & \tilde{L}^\top \end{bmatrix} \begin{bmatrix} A_0 - B\tilde{L}^{-1}\tilde{L}^{-\top}B^\top & \\ & I \end{bmatrix} \begin{bmatrix} I & \\ \tilde{L}^{-\top}B^\top & \tilde{L} \end{bmatrix}.$$

We write $\tilde{L}^{-1}$ in block form

$$\tilde{L}^{-1} = \left[\begin{array}{c|c|c} \tilde{L}_{1,1} & & \\ \hline \vdots & \ddots & \\ \hline \tilde{L}_{p,1} & \cdots & \tilde{L}_{p,p} \end{array}\right]$$

noting each block is diagonal. We see that

$$B\tilde{L}^{-1} = [B_1\tilde{L}_{1,1} + \cdots + B_\ell \tilde{L}_{\ell,1} | B_2\tilde{L}_{2,2} + \cdots + B_\ell \tilde{L}_{\ell,2} | \cdots | B_\ell \tilde{L}_{\ell,\ell} | 0 | \cdots | 0],$$

where each block has bandwidth (λ, μ). Thus,

$$B\tilde{L}^{-1}\tilde{L}^{-\top}B^\top = (B_1\tilde{L}_{1,1} + \cdots + B_\ell\tilde{L}_{\ell,1})(\tilde{L}_{1,1}B_1^\top + \cdots + \tilde{L}_{\ell,1}B_\ell^\top) + \cdots + B_\ell \tilde{L}_{\ell,\ell}^2 B_\ell^\top$$

has bandwidths $(\lambda + \mu, \lambda + \mu)$, as multiplying banded matrices adds the bandwidths. Thus, $A_0 - BL^{-1}L^{-\top}B^\top = L_0^\top L_0$ also has bandwidths $(\lambda+\mu, \lambda+\mu)$ and its reverse Cholesky factor has bandwidth $(\lambda + \mu, 0)$. Thus,

$$L = \begin{bmatrix} L_0 & \\ \tilde{L}^{-\top}B^\top & \tilde{L} \end{bmatrix}$$

is a lower triangular B^3-Arrowhead matrix with the prescribed sparsity. □

Encoded in this proof is a simple algorithm for computing the reverse Cholesky factorization, see Algorithm 1.

COROLLARY 4.3. If $A \in \mathbb{R}^{N\times N}$ is an SPD B^3-Arrowhead Matrix then the reverse Cholesky factorization can be computed and its inverse applied in optimal complexity ($O(N)$ operations).

Proof. The reverse Cholesky factorizations of banded matrices can be computed in optimal complexity so lines (1–3) take $O(np)$ operations. Multiplying banded matrices by diagonal matrices and adding them is also optimal complexity, hence lines (4–7) take $O(\max(n, m))$ operations. Finally, line (8) is another banded reverse Cholesky, which is $O(m)$ operations. Hence, the total complexity of the reverse Cholesky factorization is $O(np + \max(n, m)) = O(N)$ operations.

Algorithm 1 Reverse Cholesky for B^3-Arrowhead Matrices

Input: Symmetric positive definite B^3-Arrowhead Matrix A with block-bandwidths (ℓ, ℓ) and sub-block-bandwidths $\lambda + \lambda$.

Output: Lower triangular B^3-Arrowhead Matrix with block-bandwidths $(\ell, 0)$ and sub-block-bandwidths $\lambda + \lambda$ satisfying $A = L^\top L$.

1: **for** $k = 1, \dots, n$ **do**
2: Compute the banded reverse Cholesky factorizations $D_k = L_k^\top L_k$.
3: **end for**
4: **for** $k = 1, \dots, \ell$ **do**
5: Construct banded matrices

$$M_k = B_k \tilde{L}_{k,k} + \cdots + B_\ell \tilde{L}_{\ell,k},$$

6: **end for**
7: Form the banded matrix

$$\tilde{A}_0 = M_1 M_1^\top + \cdots + M_\ell M_\ell^\top.$$

8: Compute the banded reverse Cholesky factorization $\tilde{A}_0 = L_0^\top L_0$.
9: Return the lower triangular B^3-Arrowhead matrix

$$\begin{bmatrix} L_0 & \\ C & L_1 \oplus \cdots \oplus L_n \end{bmatrix},$$

where $C^\top = [M_1 | \cdots | M_\ell | 0 | \cdots | 0]$.

Once the factorization is computed it is straightforward to solve linear systems in optimal complexity: we write

$$L = \begin{bmatrix} L_0 & \\ L_1 & \tilde{L} \end{bmatrix} \quad \text{so that} \quad L^{-1} = \begin{bmatrix} L_0^{-1} & \\ -\tilde{L}^{-1} L_1 L_0^{-1} & \tilde{L}^{-1} \end{bmatrix}.$$

Since L_0 is banded and $\tilde{L} = L_1 \oplus \cdots \oplus L_n$, where L_k are banded, their inverses can be applied in optimal complexity. □

In Fig. 1 we demonstrate the timing[4] for this algorithm for solving the one-dimensional screened Poisson equation

$$(-\Delta + \omega^2) u = f$$

with a zero Dirichlet boundary condition, which is discretized via

$$(-\Delta_Q + \omega^2 M_Q) u = R^\top M_P \mathbf{f},$$

[4] All computations performed on an M2 MacBook Air with four threads unless otherwise stated.

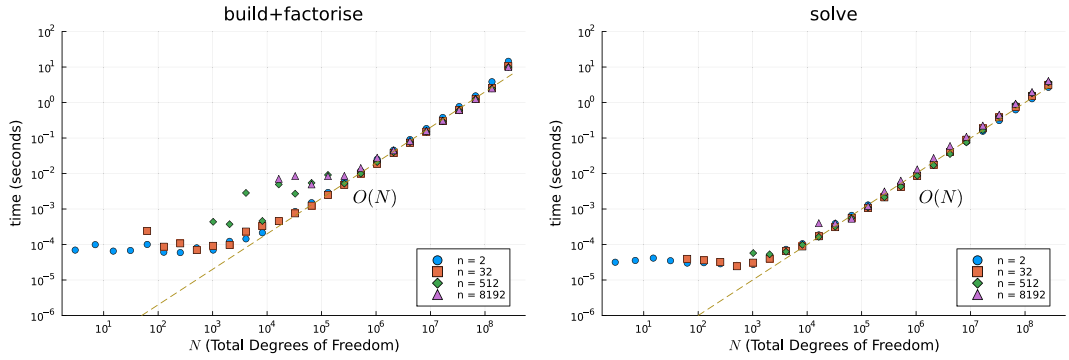

FIG. 1. Time taken to build/factorize and solve a discretization of a one-dimensional (screened) Poisson equation up to degree p, where n is the number of elements. The x-axis is $N = np - 1$, the total number of degrees of freedom, and demonstrates that the complexity is optimal as either n $(= 2/h)$ or p become large, and largely only depends on the total number of degrees of freedom.

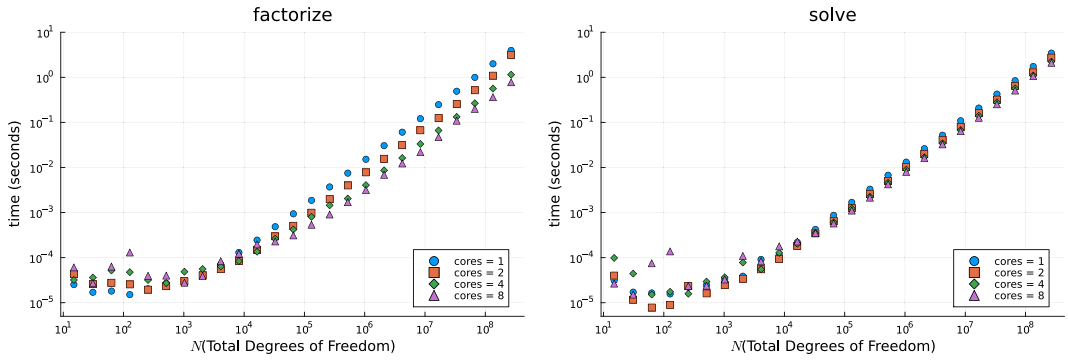

FIG. 2. Time taken to factorize and solve a one-dimensional (screened) Poisson equation up to degree p, where $n = 8$ is the number of elements, where we parallelize over multiple cores, on an M2 MacBook Air with four high-performance and four high-efficiency cores, using shared memory. We see substantial speed-up for factorization when utilizing the four high-performance cores, while a marginal speed-up when also using the high-efficiency cores.

where $\mathbf{f}$ are given Legendre coefficients. We choose $\omega = 1$ and f is random samples as these do not impact the speed of the simulation. The first plot shows the precomputation cost: building the discretization and computing its Cholesky factorization, achieving optimal complexity. The second plot shows the solve time, which is also optimal complexity. The timings of both are roughly independent of n, the number of elements, demonstrating uniform computational cost.

REMARK 4.4. An $O(N)$ solve for the matrix induced by the FEM discretization of the one-dimensional screened Poisson equation is also admissible via *static condensation* (Schwab, 1998, chapter 3.2).

In Fig. 2 we demonstrate the speed-up observed in computing factorizations and solve when parallelized over multiple cores, showing evidence of strong scaling for the factorization problem. Note that in two dimensions the communication costs mitigate any improvement from parallelization in the current implementation, and so to effectively take advantage of the potential to parallelize would likely require a more robust distributed memory implementation.

5. A generalized ADI method

Recall from Subsection 3.2 the generalized Sylvester equation for the two-dimensional screened Poisson equation (dropping the superscripts $^{\mathbf{x}}$ and $^{\mathbf{y}}$) is

$$\left(-\Delta_{Y,p} + \frac{\omega^2}{2} M_{Y,p}\right) U_{pp} M_{Y,p} + M_{Y,p} U_{pp} \left(-\Delta_{Y,p} + \frac{\omega^2}{2} M_{Y,p}\right) = R_{Y,p}^{\top} M_{P,p} F_{pp} M_{P,p} R_{Y,p} =: G_{pp}. \tag{5.1}$$

Here, $\mathbf{Y}_{0:p}(x) = \mathbf{C}_{0:p}(x)$ if we consider the screened Poisson equation with a zero Neumann boundary condition and $\omega^2 > 0$. Otherwise, $\mathbf{Y}_{0:p}(x) = \mathbf{Q}_{0:p}(x)$ if we impose a zero Dirichlet boundary condition and $\omega^2 \geq 0$.

We will solve (5.1) using a variant of the ADI method cf. Peaceman & Rachford (1955); Douglas (1962); Douglas & Gunn (1963, 1964); Bruno & Lyon (2010) and Fortunato & Townsend (2020). The first use of ADI to solve PDEs is attributed to Peaceman & Rachford (1955) and has seen many extensions, e.g., to hyperbolic and nonlinear problems as well as non-Cartesian domains (Bruno & Lyon, 2010). ADI is an iterative approach to approximate X that solves the Sylvester equation $AX - XB = F$, but in a manner that permits precise error control: given two assumptions on real-valued matrices A and B one is able to explicitly find the number of iterations required for the algorithm to compute X up to a maximum tolerance ϵ. The two assumptions are (Fortunato & Townsend, 2020):

P1. A and B are symmetric matrices.

P2. There exist real disjoint nonempty intervals $[a, b]$ and $[c, d]$ such that $\sigma(A) \subset [a, b]$ and $\sigma(B) \subset [c, d]$, where σ denotes the spectrum of a matrix.

The algorithm proceeds iteratively. First one fixes the initial matrix $X_0 = 0$. Then, iteratively for $j = 1, \ldots, J$, we compute

$$\text{for } X_{j-1/2} \text{ solve} \quad X_{j-1/2}(B - p_j I) = F - (A - p_j I)X_{j-1}, \tag{5.2}$$

$$\text{for } X_j \text{ solve} \quad (A - q_j I)X_j = F - X_{j-1/2}(B - q_j I). \tag{5.3}$$

5.1 *Generalized ADI*

In the case of the two-dimensional (Screened) Poisson equation (3.1) we have a generalized Sylvester equation, which we write in the general form as

$$AUC - DUB = F.$$

We first extend the ADI method to generalized problems in Algorithm 2.

LEMMA 5.1. Write $C = L^{\top}L$ and $D = V^{\top}V$ where L and V are lower triangular. Algorithm 2 computes U_J satisfying

$$\left\| V(U - U_J)L^{\top} \right\| \leq \epsilon \left\| VUL^{\top} \right\|.$$

Algorithm 2 Generalized Alternating Direction Implicit (ADI)

Input: Symmetric matrices $A \in \mathbb{R}^{M\times M}$, $B \in \mathbb{R}^{N\times N}$, $C \in \mathbb{R}^{N\times N}$ and $D \in \mathbb{R}^{M\times M}$, matrix $F \in \mathbb{R}^{M\times N}$, tolerance ϵ.
Output: Matrix $U \in \mathbb{R}^{M\times N}$ satisfying $AUC - DUB \approx F$.
Precomputation:

1: Use the banded symmetric generalized eigenvalue algorithms (18) to compute generalized eigenvalues $\sigma(A, D)$ and $\sigma(B, C)$, where $\sigma(A, B) := \{\lambda : \|(A - \lambda B)^{-1}\| = \infty\}$. The largest and smallest eigenvalues give us a, b, c, d such that $\sigma(\tilde{A}) \subset [a, b]$ and $\sigma(\tilde{B}) \subset [c, d]$.
2: Let the number of iterations equal $J = \lceil \log(16\gamma)\log(4/\epsilon)/\pi^2 \rceil$ where $\gamma = |c - a||d - b|/(|c - b||d - a|)$.
3: Compute the ADI shifts p_j and q_j which have the explicit formulæ depending on γ (Fortunato & Townsend 2020, equation (2.4)). Notably, we have that $p_j > 0$ and $q_j < 0$ for all $j \in \{1, \ldots, J\}$.
4: **for** $j = 1, \ldots, J$ **do**
5: Compute the reverse Cholesky factorizations of $A - q_j D$ and $B - p_j C$.
6: **end for**

Solve:

1: Let $W_0 = \mathbf{0}$
2: **for** $j = 1, \ldots, J$ **do**
3: Use the precomputed Cholesky factorizations to compute

$$W_{j-1/2} = (F - (A - p_j D)W_{j-1})(B - p_j C)^{-1},$$
$$W_j = (A - q_j D)^{-1}(F - W_{j-1/2}(B - q_j C)),$$

4: **end for**
5: **return** $W_J C^{-1}$.

Proof. We reduce a generalized Sylvester equation to a standard Sylvester equation as follows: define $X := VUL^\top$ so that our equation becomes

$$\underbrace{V^{-\top}AV^{-1}}_{\tilde{A}} X - X \underbrace{L^{-\top}BL^{-1}}_{\tilde{B}} = \underbrace{V^{-\top}FL^{-1}}_{G}.$$

$\tilde{A}$ and $\tilde{B}$ are symmetric matrices whose eigenvalues satisfy $\sigma(\tilde{A}) = \sigma(A, D)$ and $\sigma(\tilde{B}) = \sigma(B, C)$. The ADI iterations satisfy, for $X_0 = 0$,

$$X_{j-1/2}(\tilde{B} - p_j I) = G - (\tilde{A} - p_j I)X_{j-1},$$
$$(\tilde{A} - q_j I)X_j = G - X_{j-1/2}(\tilde{B} - q_j I),$$

where by convergence of the ADI algorithm (Fortunato & Townsend, 2020, theorem 2.1):

$$\|X - X_J\| \le \epsilon \|X\|. \tag{5.4}$$

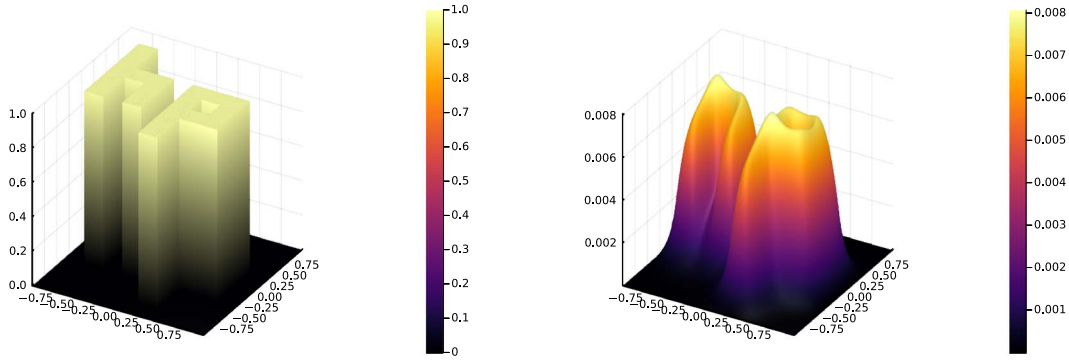

FIG. 3. Example PDE with a discontinuous right-hand side f (left) and solution (right) of the screened Poisson equation $-\Delta u + 10^2 u = f$ with a zero Dirichlet boundary condition. By using a 9×9 elements we can resolve the right-hand side exactly, and then use high p to achieve convergence.

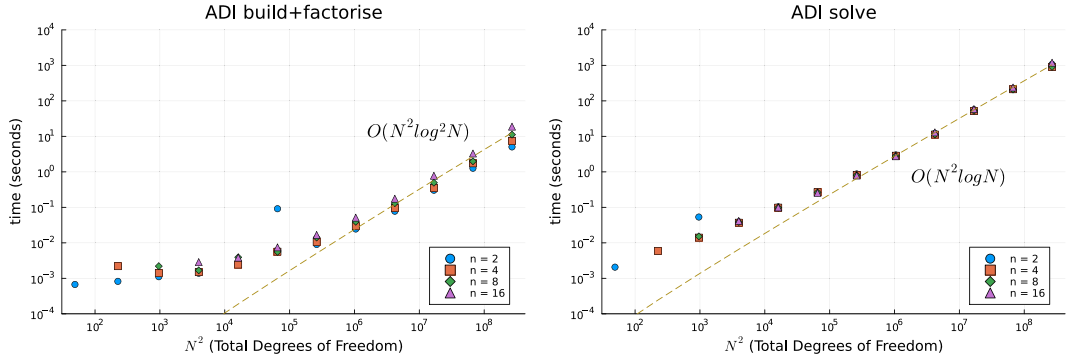

FIG. 4. Time taken to build/factorize and solve a discretization of the Poisson equation in two dimensions using ADI up to degree p, where n is the number of elements. The x-axis in the second row of timings is the total number of degrees of freedom (DOF) and demonstrates that the complexity is optimal as either $n(= 2/h)$ or p become large, and largely only depends on the total number of DOF.

Writing $W_j := V^{-1} X_j L$ and $W_{j+1/2} := V^\top X_{j+1/2} L^{-\top}$ this iteration becomes equivalent to that of Algorithm 2. We thus have, for $U_J = W_J C^{-1} = V^{-1} X_J L^{-\top}$, that

$$\left\| V(U - U_J) L^\top \right\| = \left\| X - X_J \right\| \leq \epsilon \left\| X \right\| = \epsilon \left\| VUL^\top \right\|.$$

□

In Fig. 3 we show the solution for a discontinuous right-hand side using ADI with a fixed h and high p to compute the solution. In Fig. 4 we show the computational cost of Algorithm 2 for different h and p. This shows that in practice we achieve quasi-optimal complexity, both for the precomputation and the solve. Finally, in Fig. 5 we show the solve time remains quasi-optimal for a zero Neumann boundary condition, and that the computational cost improves as ω increases.

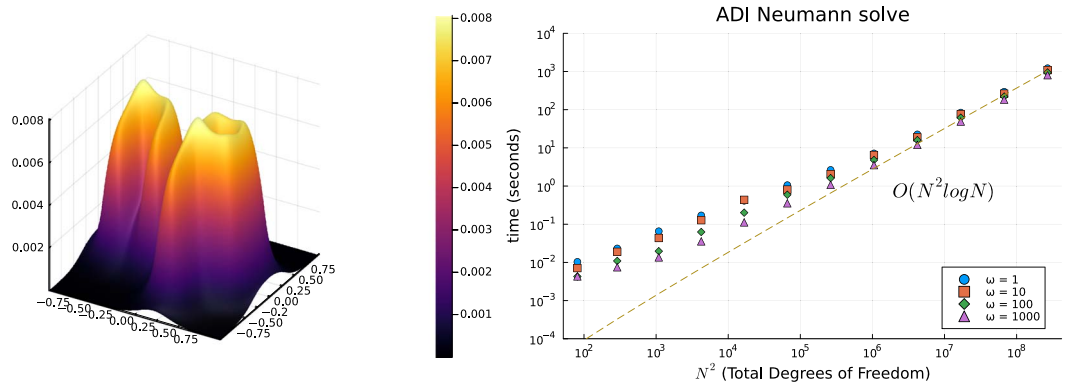

FIG. 5. Left: solution of $-\Delta u + 10^2 u = f$ with a zero Neumann boundary condition, using the same right-hand side as in Fig. 3. Right: solve timings for $n = 2$ (two elements) and increasing p with varying choices of ω.

5.2 *Complexity analysis*

In the previous experiments we observed that Algorithm 2 appears to achieve quasi-optimal complexity. In this section we prove this is guaranteed to be the case. In order to control the complexity it is necessary to control the number of iterations J, which depends on the spectral information of the operators. For simplicity, throughout this section, we set $p = q$, i.e., we consider an equal discretization degree in x and y. However, we note that all the results generalize.

A key result to derive the complexity of applying the ADI algorithm solely in terms of N and p are bounds on the spectrum for $L^{-\top} M_{Y,p} L^{-1}$, where $L^\top L = -\Delta_{Y,p} + (\omega^2/2) M_{Y,p}$. This allows us to derive the asymptotic behaviour for J.

LEMMA 5.2 (Spectrum). Consider the interval domain (a, b) and a family of quasi-uniform subdivisions $\{\mathcal{T}_h\}_h$ of the interval, where h denotes the mesh size (the minimum diameter of all the cells in the mesh) (Brenner & Scott, 2008, definition 4.4.13). For the (screened) Poisson equation with Neumann boundary conditions consider the quasi-matrix $\mathbf{Y}_{0:p}(x) = \mathbf{C}_{0:p}(x)$ and with a zero Dirichlet boundary condition consider $\mathbf{Y}_{0:p}(x) = \mathbf{Q}_{0:p}(x)$, where p is the truncation degree on each element. Suppose that $L^\top L = A_{Y,p} := (-\Delta)_{Y,p} + (\omega^2/2) M_{Y,p}$, where $\omega \neq 0$ in the case of Neumann boundary conditions. Then,

$$\sigma(L^{-\top} M_{Y,p} L^{-1}) \subseteq \left[\frac{2h^2}{24p^4 + \omega^2 h^2}, C \right], \tag{5.5}$$

where $C = \min((b-a)^2/\pi^2, \max(1, 2/\omega^2))$ in the case of a zero Dirichlet boundary condition and $C = \max(1, 2/\omega^2)$ in the case of a zero Neumann boundary condition.

Proof. The eigenvalue problem to consider is

$$L^{-\top} M_{Y,p} L^{-1} \mathbf{v}_p = \lambda \mathbf{v}_p, \tag{5.6}$$

where λ and $\mathbf{v}_p$ denote an eigenvalue and corresponding eigenvector, respectively. First, note that $L^{-\top} M_{Y,p} L^{-1}$ is congruent to an SPD matrix and, therefore, λ must be real and positive. Left multiplying

(5.6) by $L^\top$, and considering $\mathbf{w}_p = L^{-1}\mathbf{v}_p$, we deduce that

$$M_{Y,p}\mathbf{w}_p = -\lambda \Delta_{Y,p}\mathbf{w}_p + \frac{\lambda\omega^2}{2} M_{Y,p}\mathbf{w}_p. \tag{5.7}$$

For $w \in H^1(a,b)$ let $|w|_{H^1(a,b)} := \|w'\|_{L^2(a,b)}$. Left multiplying (5.7) by $\mathbf{w}_p^\top$ implies that

$$\|\mathbf{Y}_{0:p}\mathbf{w}_p\|^2_{L^2(a,b)} = \lambda |\mathbf{Y}_{0:p}\mathbf{w}_p|^2_{H^1(a,b)} + \frac{\lambda\omega^2}{2}\|\mathbf{Y}_{0:p}\mathbf{w}_p\|^2_{L^2(a,b)}. \tag{5.8}$$

(Upper bound). The upper bound can be split into three cases: (I) a zero Dirichlet boundary condition, (II) $0 < \omega^2 < 2$ and (III) $\omega^2 \geq 2$. In case (I) then $\mathbf{Y}_{0:p}(a)\mathbf{w}_p = \mathbf{Y}_{0:p}(b)\mathbf{w}_p = \mathbf{Q}_{0:p}(a)\mathbf{w}_p = \mathbf{Q}_{0:p}(b)\mathbf{w}_p = 0$. Hence, the Poincaré inequality (with the optimal Poincaré constant) implies that (Payne & Weinberger, 1960)

$$\frac{\pi^2}{(b-a)^2}\|\mathbf{Y}_{0:p}\mathbf{w}_p\|^2_{L^2(a,b)} \leq |\mathbf{Y}_{0:p}\mathbf{w}_p|^2_{H^1(a,b)} \leq \lambda^{-1}\|\mathbf{Y}_{0:p}\mathbf{w}_p\|^2_{L^2(a,b)}. \tag{5.9}$$

Thus, $\lambda \leq (b-a)^2/\pi^2$. In cases (II) and (III) we see that

$$\begin{aligned} C^{-1}\|\mathbf{Y}_{0:p}\mathbf{w}_p\|^2_{L^2(a,b)} &\leq |\mathbf{Y}_{0:p}\mathbf{w}_p|^2_{H^1(a,b)} + \frac{\omega^2}{2}\|\mathbf{Y}_{0:p}\mathbf{w}_p\|^2_{L^2(a,b)} \\ &= \lambda^{-1}\|\mathbf{Y}_{0:p}\mathbf{w}_p\|^2_{L^2(a,b)}, \end{aligned} \tag{5.10}$$

where $C^{-1} = \omega^2/2$ in case (II) and $C^{-1} = 1$ in case (III). Thus, $\lambda \leq \max(1, 2/\omega^2)$. Combining the results from cases (I)–(III) we conclude the upper bound on the spectrum.

(Lower bound). Consider a degree p polynomial π_p defined on the interval $(0,h)$. Then, the following inverse inequality holds (Schwab, 1998, theorem 3.91):

$$|\pi_p|_{H^1(0,h)} \leq 2\sqrt{3}h^{-1}p^2\|\pi_p\|_{L^2(0,h)}. \tag{5.11}$$

Consequently, (5.11) implies that

$$\begin{aligned} \lambda^{-1}\|\mathbf{Y}_{0:p}\mathbf{w}_p\|^2_{L^2(a,b)} &= |\mathbf{Y}_{0:p}\mathbf{w}_p|^2_{H^1(a,b)} + \frac{\omega^2}{2}\|\mathbf{Y}_{0:p}\mathbf{w}_p\|^2_{L^2(a,b)} \\ &\leq 12h^{-2}p^4\|\mathbf{Y}_{0:p}\mathbf{w}_p\|^2_{L^2(a,b)} + \frac{\omega^2}{2}\|\mathbf{Y}_{0:p}\mathbf{w}_p\|^2_{L^2(a,b)}. \end{aligned} \tag{5.12}$$

Hence, $\lambda \geq \frac{2h^2}{24p^4+\omega^2h^2}$. □

From the formula of J we arrive at the following:

LEMMA 5.3. Under the conditions of the previous proposition $J = O(\log N \log \epsilon^{-1})$.

This allows us to establish complexity results:

THEOREM 5.4. **Precomputation** in Algorithm 2 can be accomplished in $O(nN^2 + JN)$ operations, where we assume we can compute special functions (hyperbolic trigonometric functions, log and the elliptic integral dn) in $O(1)$ operations, where N is the maximal degrees of freedom of either coordinate direction. **Solve** in Algorithm 2 can be accomplished in $O(JN^2)$ operations. Using the bound on J in the previous result shows quasi-optimal complexity for the precomputation as $p \to \infty$.

Proof. (**Precomputation**): The B^3-Arrowhead matrices involved can be viewed as square banded matrices with bandwidth $O(n)$ and dimensions that scale like $O(N)$, hence line (1) can be computed in $O(nN^2)$ operations following Crawford (1973). By the complexity of computing reverse Cholesky factorizations of B^3-Arrowhead matrices we know lines (4–6) take $O(JN)$ operations.

(**Solve**): Multiplying and inverting B^3-Arrowhead matrices can be done on each column of W_j in $O(N)$ operations, which immediately gives the result. □

REMARK 5.5. Using inverse iteration it is likely that the precomputation cost can be reduced to $O(N)$ operations, but this would require more information on the gap between the eigenvalues. Note also that eigenvalue algorithms have errors that can alter the number of iterations J, but we have neglected taking this into consideration as it is unlikely to have a material impact.

6. Transforms and time-evolution

To utilize ADI solvers in an iterative framework for nonlinear elliptic PDEs or in time-evolution problems it is essential to be able to efficiently transform between values on a grid and coefficients. To accomplish this we need the following transforms in one dimension and two dimensions:

1. Given a grid find the expansion coefficients of the right-hand side into piecewise Legendre polynomials.
2. Given coefficients of the solution in the basis $\mathbf{Q}$ find the values on a grid.

The first stage can be tackled by transforming from values at piecewise Chebyshev grids to Chebyshev coefficients using the DCT, and thence to Legendre coefficients via a fast Chebyshev–Legendre transform (Alpert & Rokhlin, 1991; Keiner, 2011; Townsend *et al.*, 2018). We denote the p Chebyshev points of the first kind as

$$\mathbf{x}_p^{\mathrm{T}} := \left[\sin\left(\pi \frac{p-2k+1}{2p}\right)\right]_{k=1}^{p}.$$

We denote the transform from Chebyshev points to Legendre coefficients (which combines the DCT with the Chebyshev–Legendre transform) as $\mathcal{F}_p$ and its inverse as $\mathcal{F}_p^{-1}$. That is, if $f(x) = \mathbf{P}_{0:p}\mathbf{c}$ then $\mathbf{c} = \mathcal{F}_p f(\mathbf{x}_p^{\mathrm{T}})$. Now for multiple elements we affine transform the grid to get a matrix of values. That is, for a matrix of grid points $X_p^n = [\mathbf{x}_p^1 | \cdots | \mathbf{x}_p^n]$ we transform each column: $\mathcal{F}_p f(X_p^n)$. Reinterpreting this matrix as a block-vector, whose rows correspond to blocks, gives the coefficients in the basis $\mathbf{P}^{\mathbf{x}}$. That is, we use

$$\mathrm{vec}((\mathcal{F}_p f(X_p^n))^\top),$$

where vec : $\mathbb{R}^{p\times n} \to \mathbb{R}^{pn}$ is the operator from matrices to (block) vectors that concatenates the columns.

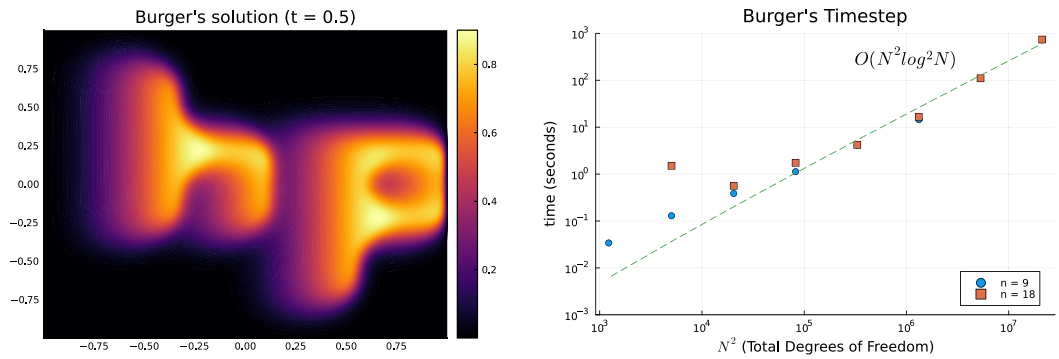

FIG. 6. Left: solution of Burgers' equation $u_t + uu_x = \epsilon \Delta u$ with a zero Dirichlet boundary condition, where the initial condition is the discontinuous right-hand side as in Fig. 3, with $\epsilon = 0.1$. Right: time taken for a single time-step, using an implicit–explicit splitting method where the linear part is solved using implicit Euler and the nonlinear part with explicit Euler, after transforming back to a grid.

To extend this to two dimensions we use the grids $\mathbf{x} = \text{vec}(X_p^n)$ and $\mathbf{y} = \text{vec}(X_q^m)$, and hence we want to transform from a matrix of values on the tensor product grid i.e.,

$$F := \left[f(x_k, y_j)\right]_{k=1,j=1}^{k=pn,j=qm}.$$

The two-dimensional transform is then $\mathcal{F}_p^n F (\mathcal{F}_q^m)^\top$.

The second stage can be accomplished by first computing the coefficients in a piecewise Legendre basis via applying the matrix $R^{\mathbf{x}}$, transforming to Chebyshev coefficients via a fast Legendre–Chebyshev transform, then applying the inverse DCT to recover the values on a Chebyshev grid. That is, if we have

$$u(x, y) = \mathbf{Q}_{0:p}(x) U \mathbf{Q}_{0:q}(y)^\top$$

then we can transform back to a grid via

$$u(\mathbf{x}, \mathbf{y}^\top) = (\mathcal{F}_p^n)^{-1} R^{\mathbf{x}} U (R^{\mathbf{y}})^\top \mathcal{F}_q^{-\top}.$$

As an example of the utility of fast transforms, Fig. 6 considers the classic Burgers' equation:

$$(u_t + uu_x)(x, y, t) = \epsilon \Delta u(x, y, t)$$

with a zero Dirichlet boundary condition and a discontinuous initial condition. We discretize in time using a simple implicit–explicit splitting method, taking a linear step via implicit Euler followed by a nonlinear step via explicit Euler:

$$u_{k+1/2}(x, y) = (I - (\delta t)\epsilon \Delta)^{-1} u_k(x, y),$$
$$u_{k+1}(x, y) = (u_{k+1/2} + (\delta t) u_{k+1/2,x} u_{k+1/2})(x, y).$$

We represent $u_k(x, y)$ as a matrix U_k containing coefficients in an expansion of tensor products of piecewise Legendre polynomials, i.e., using the basis $\mathbf{P}^{\mathbf{x}}$. The half time-steps $u_{k+1/2}(x, y)$ are then represented as a matrix $U_{k+1/2}$ giving coefficients in tensor products of $\mathbf{Q}^{\mathbf{x}}$, where the coefficients are computed using ADI as described above. To determine $u_{k+1/2}(x, y)$ on a grid we simply convert down to Legendre and then apply the inverse fast Legendre transform, that is, values are approximated by

$$V_{k+1/2} := \mathcal{F}^{-1} R U_{k+1/2} R^{\top} \mathcal{F}^{-\top}.$$

For $u_{(k+1/2),x}(x, y)$ we compute its Legendre coefficients using the derivative matrix alongside the conversion matrix:

$$V_{k+1/2,x} := \mathcal{F}^{-1} D U_{k+1/2} R^{\top} \mathcal{F}^{-\top}.$$

We can then determine the Legendre coefficients U_{k+1} as

$$U_{k+1} := \mathcal{F}\left[V_{k+1/2} + (\delta t) V_{k+1/2,x} \otimes V_{k+1/2}\right].$$

The right-hand side plot in Fig. 6 roughly demonstrates the predicted $O(N^2 \log^2 N)$ complexity.

7. ADI as a preconditioner

In this section we explore how our ADI-based solver may be used as a preconditioner for an iterative Krylov method to tackle problems with variable coefficients, including an example with a singularity. In particular, one may use a graded mesh to isolate the singularity so that the variable coefficient is well approximated by piecewise polynomials. Experimentally, we see that the ADI preconditioner is *hp*-robust and we, therefore, retain the quasi-optimal complexity of the solver as $h \to 0$ and $p \to \infty$.

REMARK 7.1 (Non-Cartesian cells and curved boundaries). The preconditioning strategy outlined in this section may also be used for discretizations of elliptic problems with non-Cartesian cells or posed on domains with curved boundaries. The underlying idea utilizes equivalent operator preconditioning (Axelsson & Karátson, 2009). We omit a description or implementation in this work, but refer the interested reader to an excellent introduction by Brubeck and Farrell in (Brubeck & Farrell, 2022, section 2.7) who prove that the preconditioner is spectrally equivalent to the original problem. See also Couzy & Deville (1995); Fischer *et al.* (2000) and Witte *et al.* (2021).

Consider the domain $\Omega = (-1, 1)^2$ and the following singular variable coefficient PDE problem:

$$\left(-\Delta - 10 \log \sqrt{x^2 + y^2}\right) u(x, y) = 1 \quad \text{with } u|_{\partial\Omega} = 0. \tag{7.1}$$

Note that the variable coefficient $-10 \log \sqrt{x^2 + y^2} \to \infty$ as $|(x, y)|_{\ell^2} \to 0$, i.e., at the origin in the centre of the domain. After an FEM discretization, the residual may be evaluated in a matrix-free manner via a quasi-optimal complexity transform as outlined in Section 6. In particular, let $G := [-10 \log \sqrt{x_k^2 + y_j^2}]_{k=1,j=1}^{k=pn,j=pn}$. Then, given the FEM coefficient matrix U of a two-dimensional

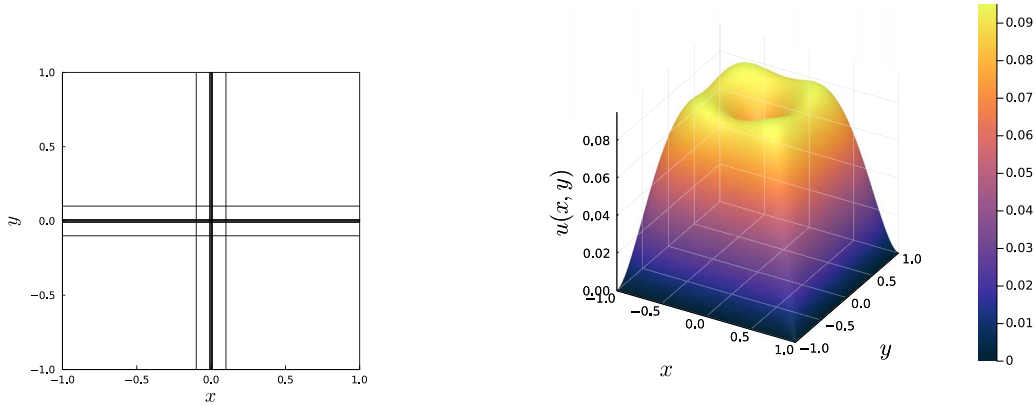

FIG. 7. Left: graded tensor-product mesh (7.3) with $m = 3$. Right: solution of singular variable coefficient problem (7.1) with partial (pre-tensor) degree $p = 128$ and graded mesh (7.3) with $m = 3$.

FEM function u_p, we may approximately evaluate $F := \langle v_p, (-\log\sqrt{x^2+y^2})u_p\rangle$, for all basis functions v_p, via

$$F = \mathrm{vec}(R^{-1}M_P\mathcal{F}[G \odot (\mathcal{F}^{-1}RUR^\top\mathcal{F}^{-\top})]\mathcal{F}^\top M_P R^{-\top}), \tag{7.2}$$

where $\odot$ denotes the Hadamard product (element-wise multiplication between two matrices). We emphasize (7.2) is computed in quasi-optimal complexity. This motivates the use of an iterative Krylov method to solve the *hp*-FEM discretization of (7.1). We opt for the conjugate gradient method (CG) preconditioned with the inverse discretized weak Laplacian matrix applied via the quasi-optimal ADI strategy of Section 5 with the tolerance 10^{-4}.

We use a tensor-product mesh graded towards the origin with the cells endpoints

$$\mathcal{T}_m = (-1, -10^{-1}, \ldots, 10^{-m}, 0, 10^{-m}, \ldots, 10^{-1}, 1)^2 \tag{7.3}$$

for a given $m \geq 1$. We plot the solution in Fig. 7 and provide the number of preconditioned CG iterations for various p and number of cells in Table 1. We observe *hp*-robustness—the number of preconditioned CG iterations is independent of the degree and number of cells of the mesh—leading to a quasi-optimal complexity solver for (7.1).

8. Future work

We have constructed the first provably quasi-optimal complexity *hp*-FEM method for the (screened) Poisson equation on a rectangle, built on taking advantage of the sparsity structure. There are some clear extensions to this work:

1. For nonpositive definite, but symmetric operators, it is possible to do $L^\top DL$ factorizations of the B^3-Arrowhead matrices in optimal complexity. However, this may lead to ill-conditioning. Unfortunately, stable factorizations such as QL only achieve $O(pn + n^3)$ complexity as there is fill-in in the top blocks.

TABLE 1 *The number of preconditioned CG iterations to solve the hp-FEM discretization of (7.1) to a relative tolerance of* 10^{-8} *with increasing partial degree p and number of cells (in brackets) in the graded mesh (7.3) as controlled by the parameter m. The preconditioner is the inverse weak Laplacian matrix applied via the quasi-optimal ADI strategy of Section 5 with tolerance* 10^{-4}

	p				
m (# cells)	2^3	2^4	2^5	2^6	2^7
1 (16)	8	7	7	7	7
2 (36)	7	7	7	7	7
3 (64)	7	7	7	7	7

2. Extensions to quasi-optimal solves on cylinders and box domains is possible. Cylinders have already been considered in Papadopoulos & Olver (2025) and a box domain could be tackled via a nested ADI approach as discussed in (Fortunato & Townsend, 2020, section 5). However, Fortunato and Townsend note that the convergence of the nested approach is highly sensitive to the termination tolerances of the inner solves. Subsequently, a more robust approach via tensor-trains was introduced in Shi & Townsend (2021). Our techniques directly translate to the tensor-train context; however, the implementation is nontrivial and left for future work.
3. A fully parallelized implementation with distributed memory in two dimensions and three dimensions. To minimize communication between workers this would involve distributing on elements, but in principle the ADI steps can be largely parallelized with only minimal communication corresponding to data on the interface needed to be shared.
4. Extensions to higher-order time-steppers. For linear problems Backward Differentiation Formulæ are a natural choice for taking advantage of the efficient solves of our spacial discretization while achieving higher-order accuracy. For nonlinear time-evolution problems a straightforward extension would be to use Strang splitting, which would achieve the same computational complexity as the simple implicit–explicit time-stepper considered with higher-order accuracy.

Acknowledgements

We would like to thank Dan Fortunato, Marcus Webb and Matt Colbrook. I.P. would like to thank Pablo Brubeck for their discussion on optimal complexity *p*-multigrid methods.

Funding

Engineering and Physical Sciences Research Council (EP/T022132/1 to S.O. and I.P.); Deutsche Forschungsgemeinschaft (EXC-2046/1 to I.P.).

Data availability

The numerical experiments found in this manuscript were conducted in Julia and can be found at ADIPoisson.jl (Knook *et al.*, 2025a,b).

References

Adams, R. A. & Fournier, J. J. (2003) *Sobolev Spaces*. Oxford, UK: Elsevier.

Alpert, B. K. & Rokhlin, V. (1991) A fast algorithm for the evaluation of Legendre expansions. *SIAM J. Sci. Stat. Comput.*, **12**, 158–179.

Averbuch, A., Israeli, M. & Vozovoi, L. (1998) A fast Poisson solver of arbitrary order accuracy in rectangular regions. *SIAM J. Sci. Comput.*, **19**, 933–952.

Axelsson, O. & Karátson, J. (2009) Equivalent operator preconditioning for elliptic problems. *Numer. Algorithms*, **50**, 297–380.

Babuška, I. & Suri, M. (1994) The p and h-p versions of the finite element method, basic principles and properties. *SIAM Rev.*, **36**, 578–632.

Babuška, I., Craig, A., Mandel, J. & Pitkäranta, J. (1991) Efficient preconditioning for the p-version finite element method in two dimensions. *SIAM J. Numer. Anal.*, **28**, 624–661.

Beuchler, S. & Pillwein, V. (2007) Sparse shape functions for tetrahedral p-FEM using integrated Jacobi polynomials. *Computing*, **80**, 345–375.

Beuchler, S. & Schoeberl, J. (2006) New shape functions for triangular p-FEM using integrated Jacobi polynomials. *Numer. Math.*, **103**, 339–366.

Beuchler, S., Pechstein, C. & Wachsmuth, D. (2012a) Boundary concentrated finite elements for optimal boundary control problems of elliptic PDEs. *Comput. Optim. Appl.*, **51**, 883–908.

Beuchler, S., Pillwein, V., Schöberl, J. & Zaglmayr, S. (2012b) Sparsity optimized high order finite element functions on simplices. In Langer, U. & Paule P. (eds.), *Numerical and Symbolic Scientific Computing*. Springer Vienna. https://doi.org/10.1007/978-3-7091-0794-2_2.

Braverman, E., Israeli, M., Averbuch, A. & Vozovoi, L. (1998) A fast 3D Poisson solver of arbitrary order accuracy. *J. Comput. Phys.*, **144**, 109–136.

Brenner, S. C. & Scott, L. R. (2008) *The mathematical theory of finite element methods*, **15***Of Texts in Applied Mathematics*, New York: Springer, 3 ed., https://doi.org/10.1007/978-0-387-75934-0.

Brubeck, P. D. & Farrell, P. E. (2022) A scalable and robust vertex-star relaxation for high-order FEM. *SIAM J. Sci. Comput.*, **44**, A2991–A3017.

Bruno, O. P. & Lyon, M. (2010) High-order unconditionally stable FC-AD solvers for general smooth domains I. Basic elements. *J. Comput. Phys.*, **229**, 2009–2033.

Buzbee, B. L., Golub, G. H. & Nielson, C. W. (1970) On direct methods for solving Poisson's equations. *SIAM J. Numer. Anal.*, **7**, 627–656.

Couzy, W. & Deville, M. O. (1995) A fast Schur complement method for the spectral element discretization of the incompressible Navier–Stokes equations. *J. Comput. Phys.*, **116**, 135–142.

Crawford, C. (1973) Reduction of a band-symmetric generalized eigenvalue problem. *Commun. ACM*, **16**, 41–44.

Douglas, J. (1962) Alternating direction methods for three space variables. *Numer. Math.*, **4**, 41–63.

Douglas, J. & Gunn, J. E. (1963) Two high-order correct difference analogues for the equation of multidimensional heat flow. *Math. Comp.*, **17**, 71–80.

Douglas, J. & Gunn, J. E. (1964) A general formulation of alternating direction methods: part I. Parabolic and hyperbolic problems. *Numer. Math.*, **6**, 428–453.

Dubiner, M. (1991) Spectral methods on triangles and other domains. *J. Sci. Comput.*, **6**, 345–390.

Fischer, P. F., Tufo, H. M. & Miller, N. (2000) An overlapping Schwarz method for spectral element simulation of three-dimensional incompressible flows. In Børstad, P. & Luskin, M. (eds.), *Parallel Solution of Partial Differential Equations*. New York, NY: Springer, pp. 159–180. https://doi.org/10.1007/978-1-4612-1176-1_7.

Fortunato, D. & Townsend, A. (2020) Fast Poisson solvers for spectral methods. *IMA J. Numer. Anal.*, **40**, 1994–2018.

Gagliardo, E. (1957) Caratterizzazioni delle tracce sulla frontiera relative ad alcune classi di funzioni in n variabili. *Rendiconti Del Seminario Matematico Della Universita di Padova*, **27**, 284–305.

Gholami, A., Malhotra, D., Sundar, H. & Biros, G. (2016) 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**, C280–C306.

Gillman, A. & Martinsson, P.-G. (2014) A direct solver with $O(N)$ complexity for variable coefficient elliptic PDEs discretized via a high-order composite spectral collocation method. *SIAM J. Sci. Comput.*, **36**, A2023–A2046.

Gillman, A., Young, P. M. & Martinsson, P.-G. (2012) A direct solver with O (N) complexity for integral equations on one-dimensional domains. *Front. Math. China*, **7**, 217–247.

Houston, P., Schwab, C. & Süli, E. (2002) Discontinuous hp-finite element methods for advection–diffusion–reaction problems. *SIAM J. Numer. Anal.*, **39**, 2133–2163.

Huismann, I., Stiller, J. & Fröhlich, J. (2019) Scaling to the stars—a linearly scaling elliptic solver for p-multigrid. *J. Comput. Phys.*, **398**, 108868.

Jia, L., Li, H. & Zhang, Z. (2022) Sparse spectral-Galerkin method on an arbitrary tetrahedron using generalized Koornwinder polynomials. *J. Sci. Comput.*, **91**, 22.

Karniadakis, G. E., Karniadakis, G. & Sherwin, S. (2005) Spectral/hp element methods for Momputational fluid dynamics. Oxford, UK: Oxford University Press. https://doi.org/10.1093/acprof:oso/9780198528692.001.0001.

Keiner, J. (2011) *Fast Polynomial Transforms*. Berlin: Logos Verlag Berlin GmbH.

Knook, K., Olver, S. & Papadopoulos, I. P. A. (2025a) *ADIPoisson.Jl*, https://github.com/ioannisPApapadopoulos/ADIPoisson.jl.

Knook, K., Olver, S. & Papadopoulos, I. P. A. (2025b) *ioannisPApapadopoulos/ADIPoisson.Jl: v0.0.4*, https://doi.org/10.5281/zenodo.15295092.

Lottes, J. W. & Fischer, P. F. (2005) Hybrid multigrid/Schwarz algorithms for the spectral element method. *J. Sci. Comput.*, **24**, 45–78.

Lynch, R. E., Rice, J. R. & Thomas, D. H. (1964) Direct solution of partial difference equations by tensor product methods. *Numer. Math.*, **6**, 185–199.

Martinsson, P.-G. (2009) A fast direct solver for a class of elliptic partial differential equations. *J. Sci. Comput.*, **38**, 316–330.

Martinsson, P.-G. (2019) Fast direct solvers for elliptic PDEs. Philadelphia, USA: SIAM. https://doi.org/10.1137/1.9781611976045.

McKenney, A., Greengard, L. & Mayo, A. (1995) A fast Poisson solver for complex geometries. *J. Comput. Phys.*, **118**, 348–355.

Olver, F. W. J., Daalhuis, A. B. O., Lozier, D. W., Schneider, B. I., Boisvert, R. F., Clark, C. W., Miller, B. R., Saunders, B. V., Cohl, H. S. & McClain, M. A. eds. *NIST Digital Library of Mathematical Functions*, https://dlmf.nist.gov/, Release 1.1.11 of 2023-09-15, https://dlmf.nist.gov/.

Orszag, S. A. (1979) Spectral methods for problems in complex geometrics. In Seymour V. Parter (ed.), *Numerical Methods for Partial Differential Equations*. New York, USA: Academic Press, 273–305.

Papadopoulos, I. P. & Olver, S. (2025) A sparse hierarchical hp-finite element method on disks and annuli. *Journal of Scientific Computing*, **104**, 51.

Pavarino, L. F. (1993) Additive Schwarz methods for the p-version finite element method. *Numer. Math.*, **66**, 493–515.

Payne, L. E. & Weinberger, H. F. (1960) An optimal Poincaré inequality for convex domains. *Arch. Rational Mech. Anal.*, **5**, 286–292.

Peaceman, D. W. & RachfordJr., H. H. (1955) The numerical solution of parabolic and elliptic differential equations. *J. Soc. Ind. Appl. Math.*, **3**, 28–41.

Schöberl, J., Melenk, J. M., Pechstein, C. & Zaglmayr, S. (2008) Additive Schwarz preconditioning for p-version triangular and tetrahedral finite elements. *IMA J. Numer. Anal.*, **28**, 1–24.

Schwab, C. (1998) *p-and Hp-Finite Element Methods: Theory and Applications in Solid and Fluid Mechanics*. Oxford, UK: Clarendon Press.

Shi, T. & Townsend, A. (2021) On the compressibility of tensors. *SIAM J. Matrix Anal. Appl.*, **42**, 275–298.

Snowball, B. & Olver, S. (2020) Sparse spectral and finite element methods for partial differential equations on disk slices and trapeziums. *Stud. Appl. Math.*, **145**, 3–35.

Szabó, B. A. (1979) Some recent developments in finite element analysis. *Comput. Math. Appl.*, pp. 99–115.
Szabó, B. & Babuška, I. (2011) *Introduction to Finite Element Analysis: Formulation, Verification and Validation*, vol. **35**. Chichester, UK: John Wiley & Sons.
Townsend, A., Webb, M. & Olver, S. (2018) Fast polynomial transforms based on Toeplitz and Hankel matrices. *Math. Comp.*, **87**, 1913–1934.
Witte, J., Arndt, D. & Kanschat, G. (2021) Fast tensor product Schwarz smoothers for high-order discontinuous Galerkin methods. *Comput. Methods Appli. Math.*, **21**, 709–728.

Appendix A. Recurrences

In this appendix we provide the formulæ for the entries in the matrices (2.2), (2.3)–(2.4) and (2.5)–(2.6). From (Olver *et al.*, 18.9.8) we have that

$$W_k(x) = \frac{1}{2k+3}(P_k(x) - P_{k+2}(x)). \tag{A.1}$$

Thus, we deduce that the entries in (2.2) are

$$\underbrace{[W_0, W_1, W_2, \ldots]}_{\mathbf{W}} = \underbrace{[P_0, P_1, P_2, \ldots]}_{\mathbf{P}} \underbrace{\begin{bmatrix} 1/3 & & & & \\ 0 & 1/5 & & & \\ -1/3 & 0 & 1/7 & & \\ & -1/5 & 0 & 1/9 & \\ & & \ddots & \ddots & \ddots \end{bmatrix}}_{L_W}.$$

Next, we derive the entries in (2.3)–(2.4). Consider the reference cell $(-1, 1)$. Then, there exists two hat functions with nonzero support, $h_0(x) = (1-x)/2$ and $h_1(x) = (x+1)/2$. Since these are degree one polynomials then, for $k \geq 2$, $\langle P_k, h_j \rangle = 0$, and hence $R_{k0} = \mathbf{0}$. Moreover, we have that $M_{00} = 2$, $M_{11} = 2/3$, $\langle P_0, h_j \rangle = 1$ and $\langle P_1, h_j \rangle = (-1)^{j+1}/3$ for $j \in \{0, 1\}$. A scaling argument reveals that these entries are independent of the size of the element. Hence, $R_{k0} \in \mathbb{R}^{n \times (n+1)}$ and the entries in (2.3) are

$$R_{00} = \begin{bmatrix} 1/2 & 1/2 & & \\ & \ddots & \ddots & \\ & & 1/2 & 1/2 \end{bmatrix}, \quad R_{10} = \begin{bmatrix} -1/2 & 1/2 & & \\ & \ddots & \ddots & \\ & & -1/2 & 1/2 \end{bmatrix}. \tag{A.2}$$

Moreover, for $j > 0$, $R_{kj} \in \mathbb{R}^{n \times n}$ and from (A.1) we deduce the entries in (2.4) are

$$-R_{(k+2)j} = R_{kj} = \begin{bmatrix} \frac{1}{1+2j} & & \\ & \ddots & \\ & & \frac{1}{1+2j} \end{bmatrix} \quad \text{if} \quad k = j \pm 1, \tag{A.3}$$

and otherwise $R_{kj} = \mathbf{0}$.

To compute the entries in (2.5)–(2.6) consider the reference cell $(-1, 1)$ and note that $h_0'(x) = -1/2$, $h_1'(x) = 1/2$ and $W_k'(x) = -P_{k+1}(x)$, cf. (2.1). Let $\delta_i = x_i - x_{i-1}$ for $i \in \{1 : n\}$. Then, by a scaling

argument, we deduce that

$$D_{00} = \begin{bmatrix} -1/\delta_1 & 1/\delta_1 & & & \\ & -1/\delta_2 & 1/\delta_2 & & \\ & & \ddots & \ddots & \\ & & & -1/\delta_n & 1/\delta_n \end{bmatrix} \in \mathbb{R}^{n\times(n+1)} \tag{A.4}$$

and, for $k > 0$,

$$D_{kk} = \begin{bmatrix} -2/\delta_1 & & & \\ & -2/\delta_2 & & \\ & & \ddots & \\ & & & -2/\delta_n \end{bmatrix} \in \mathbb{R}^{n\times n} \text{ with } D_{kj} = \mathbf{0} \text{ if } k \neq j. \tag{A.5}$$