

# Smoothed aggregation for Helmholtz problems

Luke N. Olson and Jacob B. Schroder<sup>\*,†</sup>

*Siebel Center for Computer Science, University of Illinois at Urbana-Champaign, 201 N. Goodwin Ave.,  
Urbana, IL 61801, U.S.A.*

## SUMMARY

We outline a smoothed aggregation algebraic multigrid method for 1D and 2D scalar Helmholtz problems with exterior radiation boundary conditions. We consider standard 1D finite difference discretizations and 2D discontinuous Galerkin discretizations. The scalar Helmholtz problem is particularly difficult for algebraic multigrid solvers. Not only can the discrete operator be complex-valued, indefinite, and non-self-adjoint, but it also allows for oscillatory error components that yield relatively small residuals. These oscillatory error components are not effectively handled by either standard relaxation or standard coarsening procedures. We address these difficulties through modifications of SA and by providing the SA setup phase with appropriate wave-like near null-space candidates. Much is known *a priori* about the character of the near null-space, and our method uses this knowledge in an adaptive fashion to find appropriate candidate vectors. Our results for GMRES preconditioned with the proposed SA method exhibit consistent performance for fixed points-per-wavelength and decreasing mesh size. Copyright © 2010 John Wiley & Sons, Ltd.

Received 15 May 2009; Revised 23 October 2009; Accepted 23 October 2009

KEY WORDS: algebraic multigrid (AMG); smoothed aggregation (SA); Helmholtz; indefinite; non-symmetric; discontinuous Galerkin

## 1. INTRODUCTION

We consider the scattering of waves through the scalar exterior Helmholtz problem

$$-\Delta u - \omega^2 u = f \quad \text{in } \Omega, \quad (1a)$$

$$u = 0 \quad \text{on } \Gamma_{\text{int}}, \quad (1b)$$

$$\frac{\partial u}{\partial r} - i\omega u = o(\|\mathbf{x}\|^{-1/2}) \quad \text{as } \|\mathbf{x}\| \rightarrow \infty, \quad (1c)$$

\*Correspondence to: Jacob B. Schroder, Siebel Center for Computer Science, University of Illinois at Urbana-Champaign, 201 N. Goodwin Ave., Urbana, IL 61801, U.S.A.

†E-mail: jschrod3@illinois.edu

Contract/grant sponsor: NSF; contract/grant number: DMS-0612448

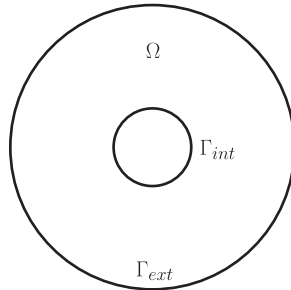


Figure 1. Example annulus-shaped domain.

where  $\Gamma_{int}$  is an interior boundary, e.g. a scatterer, and  $\partial/\partial r$  is the derivative in the radial direction. A typical representation of  $\Omega$  is given in Figure 1. The Sommerfeld boundary condition (1c) ensures that waves do not non-physically reflect back into the interior of the domain. In our numerical experiments, we truncate the infinite domain to a new boundary  $\Gamma_{ext}$  (Figure 1) and apply a common first-order approximation to (1c) given by

$$\mathbf{n} \cdot \nabla u = i\omega u \quad \text{on } \Gamma_{ext}. \quad (2)$$

The focus of this paper is on using smoothed aggregation (SA) techniques to solve systems of equations,

$$A\mathbf{x} = \mathbf{b},$$

arising from 1D and 2D discretizations of (1). Specifically, we consider standard 1D finite difference discretizations and 2D discontinuous Galerkin (DG) discretizations. There are a number of challenges in solving these resulting matrix problems. The Helmholtz problem with Sommerfeld boundary conditions typically yields a complex-symmetric, non-Hermitian, and indefinite matrix with a wave-like near null-space. The special Sommerfeld boundary conditions give the matrix its complex and non-Hermitian character. Such matrices are generally intractable for traditional algebraic multigrid (AMG) or SA techniques. It is also important to note that while the boundary conditions ensure a non-singular matrix  $A$ , the matrix may be poorly conditioned, although not numerically singular.

There are a number of features of the PDE (1) that are important to our analysis of the underlying matrix problem. The value  $\omega$  determines the wave speed of the solution, and impacts the oscillatory nature of the near null-space. Moreover, the  $\omega$  term often shifts the operator to be indefinite. Last, the boundary condition (2) introduces a non-Hermitian but complex-symmetric part into the discrete matrix problem.

## 2. SMOOTHED AGGREGATION

SA [1–3] style AMG [4, 5] is a popular and effective iterative solver and preconditioner, yet there are a number of robustness issues associated with this algorithm. For wave-like problems, e.g. Helmholtz or frequency-domain Maxwell problems, the matrix is often indefinite, complex-valued, and non-Hermitian. In addition, the near null-space may no longer be slowly varying, which is

problematic for standard interpolation based on the constant vector. Our goal is to generalize the SA framework to the exterior Helmholtz problem.

To address complex-valued matrices, an interesting direction [6] is to consider the equivalent real system for (1) that preserves its complex-symmetric nature:

$$\begin{bmatrix} -A_{\text{real}} & A_{\text{imag}} \\ A_{\text{imag}} & A_{\text{real}} \end{bmatrix} \begin{bmatrix} \mathbf{x}_{\text{real}} \\ \mathbf{x}_{\text{imag}} \end{bmatrix} = \begin{bmatrix} -\mathbf{b}_{\text{real}} \\ \mathbf{b}_{\text{imag}} \end{bmatrix}. \quad (3)$$

However, the matrix yields an unattractive eigenspectrum [7] for iterative solvers, since if  $\lambda$  is an eigenvalue of the equivalent real system, then so are  $-\lambda$ ,  $\bar{\lambda}$ , and  $-\bar{\lambda}$ .

The matrix from (1) is complex-symmetric, which necessitates a different approach than standard SA to the construction of prolongation and coarse operators. A useful observation [8] is that  $R = P^T$  is a suitable choice for complex-symmetric matrices. Moreover, it is also observed [9, 10] that Petrov–Galerkin coarsening for non-Hermitian matrices is in general useful. Restriction should approximate low left singular modes and prolongation should approximate low right singular modes. In addition, as the problem is non-Hermitian, it is advantageous to use  $\sqrt{A^*A}$  as the energy principle [10]. Consequently, we use these as guidelines in our development of an effective SA solver.

One successful approach to the Helmholtz problem is to apply complex shifts of the PDE (1) in a geometric MG setting [11, 12]. The preconditioner is based on the PDE,  $-\Delta u - (1 - \alpha i)\omega^2 u = f$ , where  $\alpha$  is a user-defined scalar. While this preconditioner is effective at low wavenumbers, it is not as effective at higher wavenumbers. Progress has been made in extending the shifted-Laplace approach to an algebraic setting [13] and for a wider range of frequencies [14]. This recent work establishes an effective multilevel method for the Helmholtz problem, but further underscores the need for more robust and algebraic solvers. In addition, many of the algebraic approaches, such as [15], focus on well-resolved discretizations—i.e. moderately high points-per-wavelength (ppw).

Multilevel techniques, in general, have been employed previously for the Helmholtz equation [16–21] and to general indefinite problems [22, 23]. Each component of the process needs to be reconsidered. For example, classic relaxation methods, such as Gauss–Seidel and weighted-Jacobi, break down for indefinite problems. One solution is to use a Krylov iteration as the smoother [17], whereas another approach is to use Kaczmarz-like relaxation (Gauss–Seidel on the normal equations) [18, 19].

Coarse grids are also a challenge, particularly when the coarse space is based on a constant representation of algebraically smooth error (cf. [16, 17, 21–23]). With a constant view of the near null-space, restriction samples the residual at a fixed number of points. As the near null-space for the Helmholtz problem is wave-like, this eventually creates a Nyquist-rate violation and the resulting coarse space is not representative of algebraically smooth error. Hence, the typical solution is to either excessively relax on coarse grids, e.g. 20–40, or to truncate the hierarchy to a small number of levels before the Nyquist-rate has been violated. The Nyquist-rate is the minimum sampling rate required to avoid aliasing.

As a consequence, several approaches replicate the wave behavior in coarse spaces, by incorporating the wave structure into interpolation [18–20]. The approach taken in [18] is insightful since the near null-space is obtained adaptively, and not from *a priori* knowledge of the problem. The coarse spaces for these approaches are generated directly from the indefinite matrix [18, 19] or from the definite matrix [20], for the case of a first-order system least-squares formulation

of the Helmholtz problem. The limitation of these approaches is that they are all geometric (i.e. not algebraic) and are not designed for either DG discretizations or high-order discretizations. Moreover, implementing these methods requires extensive modifications to a standard AMG code and, yet, the functionality is limited to a Helmholtz problem. We are motivated to capture the advantageous wave-like behavior of these methods in an algebraic and broader setting.

The DG method provides an excellent framework for adaptive- $hp$  methods and naturally provides great flexibility for non-conforming meshes and neighboring elements of differing  $p$ . For wave-like problems, e.g. Helmholtz, DG discretizations are effective because of their superior handling of dispersion error when compared with typical finite element discretizations [24].

Yet, DG discretizations also pose a problem for classic SA. For example, classic SA does not effectively aggregate in the DG setting, because of the complicated non- $M$ -matrix stencil and due to the difference between points on the edge of an element and points in the interior. There are several multilevel approaches to solving systems arising from DG discretizations [25–30]; however, most are for the positive-definite Poisson problem and are two-level or geometric in construction. The  $p$ -multigrid methods [25–27, 30] coarsen to low-order polynomial representations of the discretization and follow with standard geometric or AMG approaches. Alternatively, the approach [28, 29] rediscrretizes the problem at the original  $p$  for a hierarchy of nested meshes. These constructions are often costly and further motivate a more algebraic approach.

An alternative to nodal DG discretizations for electromagnetics problems is edge elements. For such discretizations of curl–curl operators, effective SA solvers [31–33] have been developed. We specifically target discrete problems based on a nodal basis.

The principles of SA remain attractive since there are relatively few components to the SA framework that need to be addressed. It is well known that classic strength-of-connection measures fail for non- $M$ -matrices [34]. This is particularly true for DG discretizations [35]. In addition, for DG discretizations and indefinite problems, standard prolongation smoothing breaks down. Moreover, for wave-like problems, the near null-space is no longer slowly varying, but is rich with wave-like modes. Hence, the goal is threefold: to develop appropriate near null-space vectors from which to base interpolation, an appropriate prolongation smoothing strategy and a coarsening scheme which deals with the richness of the near null-space.

We seek a solver with several characteristics. First, we target an algebraic solver using only the matrix, the finest level mesh coordinates of the problem, and the frequency of the Helmholtz operator (or wavelength of the near null-space). Moreover, we expect the algorithm to fit easily into an existing SA implementation. In terms of efficiency, we construct a solver with a fixed, moderate amount of relaxation on each level. The maximum coarse grid size should be fixed to ensure proper multilevel scaling. And finally, we consider numerical simulations of fixed ppw with decreasing mesh size in order to properly gauge the scalability with respect to the wavenumber.

In order to gain insight into solving higher-dimensional discretizations, we develop a SA solver for the 1D Helmholtz problem in Section 3. The 1D case still presents a challenging problem since the operator is non-Hermitian, indefinite, and the near null-space is wave-like. This requires suitable modifications to standard relaxation, the standard near null-space mode, and prolongation smoothing. We extend this in Section 4 to a SA solver for a DG discretization of the 2D Helmholtz problem. In particular interest, an algebraic multiple coarsening strategy is developed in order to incorporate the rich wave-like nature of the near null-space in 2D. Both Sections 3 and 4 take the approach of combining our *a priori* knowledge of the wave-like near null-space and then adapting that knowledge to the current matrix. Sections 3 and 4 each end with encouraging numerical results for the problems presented. In Section 5, concluding remarks are made.

Algorithm 1: sa\_setup( $A, B$ )

---

```

1  $A_0 = A$ 
2  $B_0 = B$ 
3  $k = 0$ 
4 while size( $A_k$ ) > max_coarse
5    $S_k = \text{Strength}(A_k)$ 
6    $C_k = \text{Aggregate}(S_k)$ 
7    $T_k, B_{k+1} = \text{InjectCandidates}(C_k, B_k)$ 
8    $P_k = \text{SmoothProlongator}(A_k, T_k)$ 
9   if  $A$  is Hermitian
10     $A_{k+1} = P_k^* A_k P_k$ 
11  else if  $A$  is Complex-symmetric
12     $A_{k+1} = P_k^T A_k P_k$ 
13     $k = k + 1$ 
14 return  $A_0, \dots, A_k, P_0, \dots, P_{k-1}$ 

```

---

### 2.1. SA algorithm

In this section, we present an overview of the SA setup algorithm [1–3]. The function of the SA setup phase is to construct coarse operators  $A_k$ , with  $A_0 = A$  being the index associated with the finest level, through the construction of interpolation operators  $P_k: \mathbb{R}^{n_{k+1}} \rightarrow \mathbb{R}^{n_k}$ , where  $n_k$  and  $n_{k+1}$  are sizes of two successively coarser grids. Algorithm 1 describes this process. The input is a matrix,  $A$ , and a set of  $m$  near null-space candidates,  $B$ .  $B$  is a set of algebraically smooth modes that are slow to relax on the fine mesh, and is intended to form a basis for interpolation. For diffusion and linearized elasticity, these vectors are the constant and rigid-body modes, respectively.

For a given level, Algorithm 1 proceeds first by determining strong connections in the matrix graph of  $A_k$  [1, 34, 36], resulting in the sparse matrix  $S_k$  with non-zero entry  $(i, j)$  if the two degrees-of-freedom  $i$  and  $j$  are deemed strongly connected. From strength matrix  $S_k$ , the  $n_k$  vertices in the matrix graph are aggregated into  $a_{k+1}$  groups. This determines the initial sparsity structure of the interpolation operator  $P_k$ . If  $a_{k+1}$  aggregates are chosen, then the sparsity structure of  $P_k$  is represented by  $C_k \in \mathbb{R}^{n_k \times a_{k+1}}$ .  $C_k$  is a sparse matrix such that the entry  $(i, j)$  is non-zero only if degree-of-freedom  $i$  is in aggregate  $j$ . From the aggregation matrix  $C_k$ , a tentative prolongator,  $T_k$ , is formed.  $T_k$  is formed by injecting  $B$  into  $C_k$  in a block column-wise fashion. Thus,  $T_k$  is of size  $n_k \times n_{k+1}$ , where  $n_{k+1} = ma_{k+1}$  and  $m$  is the number of near null-space vectors.

The tentative prolongator represents the near null-space components, which are algebraically smooth and slow to relax on the fine mesh. A local QR factorization is then applied to each block column with the  $Q$  replacing the non-zero part of the block column, and the  $R$ 's forming  $B_{k+1}$  such that  $T_k B_{k+1} = B_k$ . Following the construction of  $T_k$ , an effective prolongator is constructed by smoothing  $T_k$  in order to widen the interpolation stencil and lower the energy of each column. A common prolongation smoother for definite problems is weighted-Jacobi. Finally, a coarse level operator,  $A_{k+1}$ , is formed based on whether  $A$  is Hermitian or complex-symmetric. If  $A$  is Hermitian, then the standard Galerkin condition is used. If  $A$  is complex-symmetric, a Petrov–Galerkin condition is used. The algorithm halts when the size of  $A_k$  drops below a threshold.

Algorithm 2 carries out standard multigrid cycling given an appropriate hierarchy of operators,  $A_k$ , and prolongators,  $P_k$ . Each recursive call to sa\_solve() performs the following for a given level,  $k$ . First, presmoothing, e.g. Gauss–Seidel or weighted-Jacobi, is carried out. Second, the  $k$ th level residual,  $r_k$ , is formed and then restricted to level  $k+1$  to give  $b_{k+1}$ . The restriction operator

Algorithm 2: sa\_solve( $k, \text{cycle}, A_0, \dots, A_N, P_0, \dots, P_{N-1}, x_k, b_k$ )

---

```

1  if  $k=N$ 
2    return Solve( $A_k, b_k$ )
3  else
4     $x_k = \text{Presmooth}(A_k, x_k, b_k)$ 
5     $r_k = b_k - A_k x_k$ 
6    if  $A$  is Hermitian
7       $b_{k+1} = P_n^* r_k$ 
8    else if  $A$  is Complex-symmetric
9       $b_{k+1} = P_n^T r_k$ 
10    $x_{k+1} = 0$ 
11   if cycle is V
12      $x_{k+1} = \text{sa\_solve}(k+1, \text{cycle}, A_0, \dots, A_N, P_0, \dots, P_{N-1}, x_{k+1}, b_{k+1})$ 
13   else if cycle is W
14      $x_{k+1} = \text{sa\_solve}(k+1, \text{cycle}, A_0, \dots, A_N, P_0, \dots, P_{N-1}, x_{k+1}, b_{k+1})$ 
15      $x_{k+1} = \text{sa\_solve}(k+1, \text{cycle}, A_0, \dots, A_N, P_0, \dots, P_{N-1}, x_{k+1}, b_{k+1})$ 
16    $x_k = x_k + P_k x_{k+1}$ 
17    $x_k = \text{Postsmooth}(A_k, x_k, b_k)$ 
18   return  $x_k$ 

```

---

is chosen according to  $A$ 's symmetry properties, i.e. either  $P_k^*$  if  $A$  is Hermitian, or  $P_k^T$  if  $A$  is complex-symmetric. Third, the initial guess for level  $k+1$ ,  $x_{k+1}$ , is set to 0. Fourth, sa\_solve() is called recursively to solve the residual equation on level  $k+1$ . Fifth upon completion of the recursive call, the error estimate from level  $k+1$  is interpolated to level  $k$  and added to  $x_k$ . Finally, the updated system is post-smoothed, before returning the new solution estimate,  $x_k$ .

The number of presmoothing and postsmoothing steps defines the type of V or W cycle. For instance, a V(1,2) cycle refers to the V-cycle option in Algorithm 2 with 1 presmoothing step, e.g. one iteration of Gauss–Seidel, and 2 postsmoothing steps, e.g. two iterations of Gauss–Seidel. A W(1,2) cycle is similarly defined.

### 3. 1D HELMHOLTZ SOLVER

In this section, we develop a solver for the 1D Helmholtz problem discretized with finite differences. The 1D problem is a simple setting in comparison with higher dimensions, yet the development exposes the key features of the solver which are necessary as we extend to 2D. In 1D, we address fundamental challenges, such as the impact of indefiniteness on both relaxation and interpolation, as well as complex-valued matrices and the lack of Hermitian symmetry. Moreover, the central difficulty of a wave-like near null-space is readily apparent.

Before designing a SA solver, we first define the energy inner-product under which we enforce the complementary relationship between relaxation and interpolation. The indefinite and non-Hermitian  $A$  does not induce an inner-product; hence, we propose an  $A^*A$  inner-product for practical implementation purposes, although the normal equations are never explicitly formed.  $A^*A$  is used (instead of  $AA^*$ , for example) so that user-provided near null-space modes are preserved—i.e.  $AB \approx 0$  implies  $A^*AB \approx 0$ . We also choose this energy principle because it easily extends to complex-valued systems.

*Remark 1*

Notably, in [10],  $\sqrt{A^*A}$  is used as the energy principle when designing algorithms for non-Hermitian problems, despite the impractical calculation of  $\sqrt{A^*A}$ .

*3.1. Solver components*

In this section, we describe the multilevel components of the proposed solver, such as prolongation, restriction, and relaxation. A discussion of coarsening is omitted since aggregation generally results in groupings of three fine degrees-of-freedom per aggregate as expected in 1D.

We first propose an effective prolongation smoother for non-Hermitian and indefinite problems based on a variant of the energy-minimizing prolongation smoother [2], but posed in the  $A^*A$  norm. Specifically, we minimize the energy for each of the  $n$  columns of  $P$  with

$$\min f_j(P) = \|P_{:,j}\|_{A^*A} \quad \text{for } j = 1, \dots, n. \quad (4)$$

The modified algorithm executes conjugate gradient on the normal equations (CGNR) to minimize (4) subject to two constraints. The first constraint forces the sparsity pattern of  $P$  to be a subset of the sparsity pattern of  $|S|^k |T|$ , where  $T$  is the tentative prolongator,  $S$  is a strength-of-connection matrix,  $k$  is a positive integer, and  $|\cdot|$  is an element-wise magnitude function. Applying  $|\cdot|$  ensures that the interpolation stencil will grow as expected. This constraint is enforced by zeroing out entries in each new search direction, which are not a part of the matrix graph of  $|S|^k |T|$ . The second constraint forces  $P B_c = B$ , where  $B_c$  is the set of near null-space modes on the coarser level. The second constraint enforces exact representation of  $B$  throughout the hierarchy, i.e. the span of the product of all prolongation operators,  $\text{span}(P_0 P_1 \dots P_k)$ , exactly preserves  $B$ . This constraint is enforced by starting with a standard tentative prolongator,  $T$ , such that  $T B_c = B$ . Then each update  $Y$  to  $T$  has a row-wise projection strategy applied so that  $Y B_c = 0$ . The use of a Krylov method as the minimization technique and the extension to the  $A^*A$  norm are new to this approach.

In Algorithm 3, we describe the energy-minimizing prolongation smoother. Conceptually, the algorithm runs CGNR to solve the equation  $A^*AT = 0$  for each column of  $T$  inside a constraint space.  $\langle \cdot, \cdot \rangle_F$  denotes a Frobenius inner-product, i.e. an element-wise multiplication of two matrices followed by an element-wise summation of the result. Overbar notation on a matrix represents an element-wise conjugation. We explicitly form  $A^*AP$  as opposed to forming only  $AP$  as in standard CGNR, so that we can enforce the constraints on each new search direction. For 1D,  $S$  has the same matrix graph as  $A$ .

Algorithm 3: energy\_min\_prolongation\_smoother( $A, T, B_c, S, k, \text{iter}$ )

---

```

1 # A      := Discrete operator
2 # T      := Tentative prolongator, such that TBc = B
3 # Bc    := Near null-space modes for coarse grid
4 # S      := Strength-of-connection matrix
5 # k      := Interpolation stencil width
6 # iter   := Number of minimization steps
7
8 Tsparsity pattern = |T|
9 for i = 1 to k
10   Tsparsity pattern = |S| Tsparsity pattern
11
```

```

12  $i=0$ 
13  $D = \text{diag}(A^*A)$  # Diagonal preconditioner
14  $R = -A^*(AT)$  # Initial residual
15  $R = \text{enforce}(R, T_{\text{sparsity pattern}})$  # Enforce sparsity pattern
16 # of  $T_{\text{sparsity pattern}}$  on  $R$ 
17  $R = \text{project\_out}(R, B_c)$  # Enforce  $RB_c = 0$ 
18
19 while  $i < \text{iter}$ 
20  $Z = D^{-1}R$ 
21  $\gamma = \langle \overline{R}, Z \rangle_F$ 
22
23 if  $i$  is 0
24  $\beta = 0$ 
25 else
26  $\beta = \gamma / \gamma_{\text{old}}$ 
27  $Y = Z + \beta Y$  #  $Y$  is new search direction
28  $\gamma_{\text{old}} = \gamma$ 
29
30  $Y_{A^*A} = A^*(AY)$ 
31  $Y_{A^*A} = \text{enforce}(Y_{A^*A}, T_{\text{sparsity pattern}})$  # Enforce sparsity pattern
32 # of  $T_{\text{sparsity pattern}}$  on  $Y_{A^*A}$ 
33  $Y_{A^*A} = \text{project\_out}(Y_{A^*A}, B_c)$  # Enforce  $Y_{A^*A}B_c = 0$ 
34  $\alpha = \gamma / \langle \overline{Y}, Y_{A^*A} \rangle_F$ 
35  $T = T + \alpha Y$  # Update tentative prolongator
36  $R = R - \alpha Y_{A^*A}$  # Update residual
37  $i = i + 1$ 
38
39  $P = T$ 
40 return  $P$ 

```

Once  $P$  is obtained,  $R$  must be determined. Petrov–Galerkin coarsening [8–10], i.e.  $R \neq P^*$ , is desirable for AMG when applied to non-Hermitian matrices. As we expect  $\text{span}(R^*)$  to approximate low-energy left singular vectors and  $\text{span}(P)$  to approximate low-energy right singular vectors, it follows that  $R$  should be the adjoint of interpolation for the adjoint of  $A$ . If complex conjugation distributes over the prolongation smoother algorithm such that the complex symmetry of  $A$  is preserved, then  $R = P^T$  [8], the non-Hermitian transpose of  $P$ . More specifically, letting  $P_A$  and  $P_{A^*}$  be the prolongators for  $A$  and  $A^*$ , respectively, if  $P_A = P_{A,\text{real}} + iP_{A,\text{imag}}$  then  $P_{A^*} = P_{A^*,\text{real}} + iP_{A^*,\text{imag}} = P_{A,\text{real}} - iP_{A,\text{imag}}$ . As a result,  $P_A^T = P_{A^*} = R$  and  $\text{span}(R^*)$  approximates low-energy left singular vectors. This is a property enjoyed by energy-minimizing prolongation smoother.

*Remark 2*

Let  $A$  be a complex-symmetric, i.e.  $A = A_{\text{real}} + iA_{\text{imag}}$  and  $A^* = A_{\text{real}} - iA_{\text{imag}}$ . Then,  $AB \approx 0$  implies  $A^*\overline{B} \approx 0$ .

*Theorem 1*

Let  $A$  be a complex-symmetric matrix. Let  $P_A$  and  $P_{A^*}$  be the prolongation operators defined by Algorithm 3 for  $A$  and  $A^*$ , respectively, with near null-space modes  $B_A$  and  $B_{A^*} = \overline{B}_A$ , respectively. Subscript  $A$  and subscript  $A^*$  generally refer to a quantity in Algorithm 3 when applied to  $A$  and  $A^*$ , respectively. Let strength-of-connection matrices,  $S_A$  and  $S_{A^*}$  be identical. Then  $P_A^T = P_{A^*}$ , i.e.  $P_A = \overline{P_{A^*}}$ .

*Proof*

The proof proceeds by induction, by first showing equivalence for  $i=0$  and then showing the inductive step.

We denote  $D = \text{diag}(A^*A) = \text{diag}(AA^*)$ , which is purely real.

The two constraint enforcement functions preserve the desired conjugate equivalence, and hence their affects are ignored during the inductive part of the proof. Specifically, if  $X_A = \overline{X_{A^*}}$ , then  $\hat{X}_A = \text{enforce}(X_A, T_{\text{sparsity pattern}})$  and  $\hat{X}_{A^*} = \text{enforce}(X_{A^*}, T_{\text{sparsity pattern}})$  preserve this property such that  $\hat{X}_A = \overline{\hat{X}_{A^*}}$ . The enforce function only zeros out entries not in the matrix graph of  $T_{\text{sparsity pattern}}$ , which is the same for  $A$  and  $A^*$  because  $S_A = S_{A^*}$ . Hence, conjugate equivalence is preserved.

Similarly, if  $X_A = \overline{X_{A^*}}$  and  $B_A = \overline{B_{A^*}}$ , then  $\hat{X}_A = \text{project\_out}(X_A, B_A)$  and  $\hat{X}_{A^*} = \text{project\_out}(X_{A^*}, B_{A^*})$  preserve this property, such that  $\hat{X}_A = \overline{\hat{X}_{A^*}}$ . The project\_out function only applies a projection operator in the Euclidean inner-product on each row—i.e. it can be expressed in terms of matrix–matrix multiplication such that  $X_A$  is right multiplied by  $(I - B_A(B_A^*B_A)^{-1}B_A^*)$  and  $X_{A^*}$  by  $(I - B_{A^*}(B_{A^*}^*B_{A^*})^{-1}B_{A^*}^*)$ . Conjugation distributes over matrix–matrix multiplication and matrix inversion; hence, conjugate equivalence is preserved.

Consider iteration  $i=0$ . Algorithm 3 directly yields

$$\begin{aligned} P_A &= T_A + \alpha_A Y_A \\ &= T_A + \alpha_A D^{-1} R_A \\ &= T_A - \alpha_A D^{-1} A^* A T_A \\ &= \overline{T_{A^*}} - \alpha_A D^{-1} A^* A \overline{T_{A^*}}, \end{aligned} \quad (5)$$

using the identity,  $T_A = \overline{T_{A^*}}$ , for the initial tentative prolongators. This identity is readily apparent by considering the standard tentative prolongator algorithm as outlined in Section 2. Identical strength-of-connection matrices give  $T_A$  and  $T_{A^*}$  identical sparsity patterns. The tentative prolongators are then constructed by injecting according to the sparsity pattern  $B_A$  into  $T_A$  and  $B_{A^*}$  into  $T_{A^*}$ . Finally, local QR factorizations are performed on each block in  $T_A$  and  $T_{A^*}$ . Using  $B_A = \overline{B_{A^*}}$  and the fact that  $\text{QR}(\overline{X}) = \overline{\text{QR}(X)}$  yields conjugate equivalence of entries such that  $T_A = \overline{T_{A^*}}$ .

Next,

$$P_A = \overline{T_{A^*}} - \overline{\alpha_{A^*}} D^{-1} A^* A \overline{T_{A^*}}, \quad (6)$$

using the identity,

$$\begin{aligned} \alpha_A &= \frac{\langle \overline{R}_A, Z_A \rangle_F}{\langle \overline{Y}_A, Y_{A, A^* A} \rangle_F} && \text{by Algorithm 3} \\ &= \frac{\langle -A^* A T_A, -D^{-1} A^* A T_A \rangle_F}{\langle -D^{-1} A^* A T_A, -A^* A D^{-1} A^* A T_A \rangle_F} && \text{by complex symmetry, } A = \overline{A^*} \\ &= \frac{\langle -A A^* T_{A^*}, -D^{-1} A A^* T_{A^*} \rangle_F}{\langle -D^{-1} A A^* T_{A^*}, -A A^* D^{-1} A A^* T_{A^*} \rangle_F} && \text{conjugation distributes over } \langle \cdot, \cdot \rangle_F \\ &= \overline{\alpha_{A^*}}. \end{aligned} \quad (7)$$

Now using the complex symmetry of  $A$ ,

$$P_A = \overline{T}_{A^*} - \overline{\alpha}_{A^*} D^{-1} \overline{A A^*} \overline{T}_{A^*} = \overline{P}_{A^*}, \tag{8}$$

which yields the expected result for the first iteration.

Last, comes the inductive step. Suppose that  $P_A = \overline{P}_{A^*}$  at iteration  $i = n - 1$ . Using a superscript index, this implies

$$T_A^{(n-1)} = \overline{T}_{A^*}^{(n-1)}, \tag{9}$$

which also yields

$$Y_A^{(n-1)} = \overline{Y}_{A^*}^{(n-1)}, \quad R_A^{(n-2)} = \overline{R}_{A^*}^{(n-2)} \quad \text{and} \quad \alpha_A^{(n-1)} = \overline{\alpha}_{A^*}^{(n-1)}. \tag{10}$$

To define  $R_A^{(-1)}$  and  $R_{A^*}^{(-1)}$ , use the calculation in line 14.

Algorithm 3 directly yields

$$\begin{aligned} T_A^{(n)} &= T_A^{(n-1)} + \alpha_A^{(n)} Y_A^{(n)} \\ &= T_A^{(n-1)} + \alpha_A^{(n)} (D^{-1} R_A^{(n-1)} + \beta_A^{(n)} Y_A^{(n-1)}) \quad \text{from (9) and (10)} \\ &= \overline{T}_{A^*}^{(n-1)} + \alpha_A^{(n)} (D^{-1} R_A^{(n-1)} + \beta_A^{(n)} \overline{Y}_{A^*}^{(n-1)}) \\ &= \overline{T}_{A^*}^{(n-1)} + \alpha_A^{(n)} (D^{-1} \overline{R}_{A^*}^{(n-1)} + \beta_A^{(n)} \overline{Y}_{A^*}^{(n-1)}), \end{aligned} \tag{11}$$

using the identity,

$$\begin{aligned} R_A^{(n-1)} &= R_A^{(n-2)} + \alpha_A^{(n-1)} Y_{A, A^* A}^{(n-1)} \\ &= R_A^{(n-2)} + \alpha_A^{(n-1)} A^* A Y_A^{(n-1)} \quad \text{from (10)} \\ &= \overline{R}_{A^*}^{(n-2)} + \overline{\alpha}_{A^*}^{(n-1)} A^* A \overline{Y}_{A^*}^{(n-1)} \quad \text{by complex symmetry, } A = \overline{A^*} \\ &= \overline{R}_{A^*}^{(n-2)} + \overline{\alpha}_{A^*}^{(n-1)} \overline{A A^*} \overline{Y}_{A^*}^{(n-1)} \quad \text{conjugation distributes giving} \\ &= \overline{R}_{A^*}^{(n-1)}. \end{aligned} \tag{12}$$

Next, we state the identities

$$\alpha_A^{(n)} = \overline{\alpha}_{A^*}^{(n)} \quad \text{and} \quad \beta_A^{(n)} = \overline{\beta}_{A^*}^{(n)}, \tag{13}$$

which can easily be shown as for  $\alpha_A^{(0)} = \overline{\alpha}_{A^*}^{(0)}$ . The single if/else statement in Algorithm 3 does not affect the inductive argument, because in either case, the only necessary property of  $\beta$  still holds, i.e.  $\beta_A^{(n)} = \overline{\beta}_{A^*}^{(n)}$ . The application of these two identities finally yields

$$T_A^{(n)} = \overline{T}_{A^*}^{(n-1)} + \overline{\alpha}_{A^*}^{(n)} (D^{-1} \overline{R}_{A^*}^{(n-1)} + \overline{\beta}_{A^*}^{(n)} \overline{Y}_{A^*}^{(n-1)}) = \overline{T}_{A^*}^{(n)}, \tag{14}$$

which is equivalent to  $P_A^{(n)} = \overline{P}_{A^*}^{(n)}$ . Thus by induction, as the energy-minimizing prolongation smoother is composed of basic operations over which complex conjugation distributes, computation of  $P_A$  and  $P_{A^*}$  preserves the complex symmetry of  $A$ —i.e.  $P_A^T = P_{A^*}^*$ .  $\square$

Relaxation for indefinite, complex problems is a challenge as standard methods, such as weighted-Jacobi often excite low-energy modes while attenuating the high-energy components. Relaxation methods on the normal equations are a common approach [10]. One effective method is Gauss–Seidel NR [37] for indefinite problems [18, 19, 23], and is simply defined as Gauss–Seidel on  $A^*A$ .

An alternative relaxation strategy is to use a non-stationary Krylov or Krylov-like method, such as GMRES [17] or USYMQR [10, 38]. In particular, USYMQR forms a favorable Krylov-like subspace in which the residual is reduced. There are, however, two important drawbacks to non-stationary relaxation methods, such as GMRES or USYMQR. If SA is used as a preconditioner, then a more expensive flexible Krylov outer-iteration such as fGMRES [37] must be used. As residual components are not reduced by a fixed amount each iteration, stopping criteria for relaxation are necessary. Owing to these limitations and computational experiments, we advocate the use of Gauss–Seidel NR.

### 3.2. Near null-space selection

In 1D, the null-space of the unrestricted PDE (1), i.e. with no boundary conditions, consists of two functions,

$$\exp(i\omega x) = \cos(\omega x) + i \sin(\omega x)$$

and

$$\exp(-i\omega x) = \cos(\omega x) - i \sin(\omega x). \quad (15)$$

This implies that a suitable construction of  $B$  is from either the exponentials (15) or separately from  $\cos(\omega x)$  and  $\sin(\omega x)$ , which are themselves null-space modes of the unrestricted PDE and form a basis for the functions (15).

Another possibility is  $B = \mathbf{1}$ , as in classical multigrid. However, the choice of a constant near null-space is dubious because of the wave-like nature of the problem. Basing restriction on the constant, samples the residual at a fixed number of points at each level resulting in a coarse level with unresolved waves. As a consequence, the wave-like near null-space modes become aliased since the Nyquist-rate is violated, and the solver's performance degrades sharply.

Therefore, constructing  $B$  based on the exponential or cosine and sine modes is critical. An important component in representing  $B$  is the wavenumber  $\omega$ , but the  $\omega$  described by the PDE (1) is inaccurate for the low-energy modes of the system. For example, we present in Figure 2(a), the near null-space mode  $v$  generated by the adaptive-SA [39] next to the mode based on the natural wavenumber of the PDE. The adaptive mode  $v$  is identified with a different wavenumber and this difference becomes more pronounced for less resolved problems, which highlights the effect of discretization error. The difference is also likely related to satisfying the boundary conditions.

Adaptive-SA [39] is an approach that assumes no *a priori* knowledge of the near null-space of the operator. Columns in  $B$  are generated by a process that begins with a random initial guess, which is then refined through a strategy that resembles multigrid cycling until the guess becomes a suitable approximation of a low-energy mode not yet captured by the coarse space. We have extended the standard adaptive-SA approach to the  $A^*A$  norm by using the solver components of relaxation, restriction, and prolongation smoothing from Section 3.1, in order to provide insight into the nature of the near null-space. Adaptive-SA produces one view of the near null-space and using a  $B$  that mimics  $v$  yields an effective stand-alone SA solver. Moreover,  $v$  is similar to the lowest

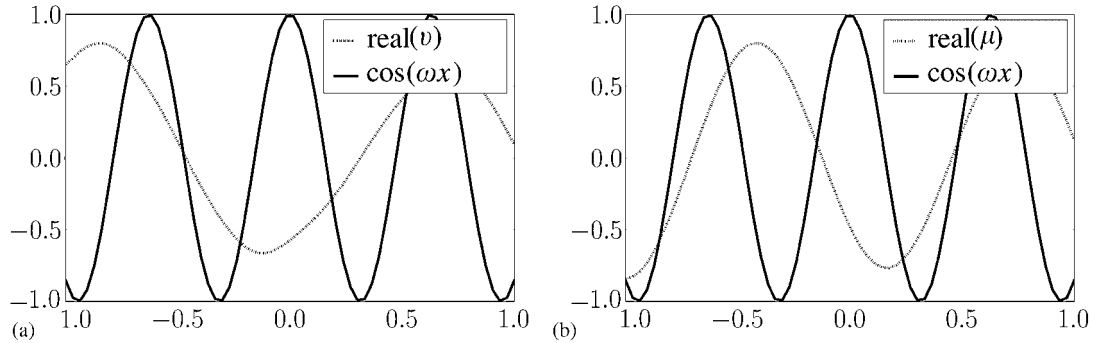


Figure 2. 1D near null-space modes: (a) adaptive-SA mode and (b) lowest right singular vector.

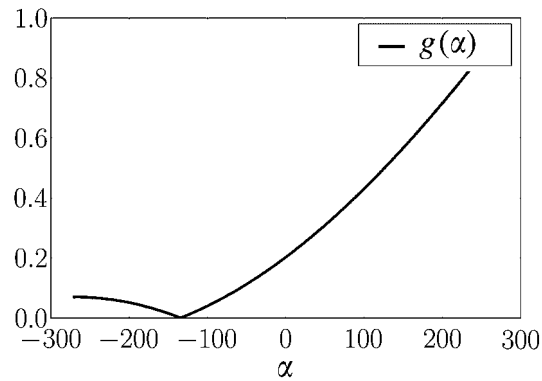


Figure 3. Plot of sample  $g(\alpha)$ .

right singular vector  $\mu$  of  $A$ , as shown in Figure 2(b). As the lowest right singular vector must be captured by  $\text{span}(P)$ , this confirms the accuracy of the adaptively generated mode. Unfortunately, adaptive-SA is expensive, but we already know much about the nature of the near null-space of the system for this problem. We therefore seek a representative wavenumber to describe the near null-space through  $B$  in an inexpensive, but adaptive way. Specifically, we find the wavenumber that defines the lowest energy cosine function by minimizing the function

$$g(\alpha) = \frac{\|\cos((\omega + \alpha)x)\|_{A^*A}}{\|\cos((\omega + \alpha)x)\|}. \quad (16)$$

Grid information, other than spacing, is not represented in matrix  $A$  for this problem. Matrix  $A$  is influenced by a particular grid only from boundary information. As a result, we reduce this dependence when computing  $A^*A$  norms by ignoring boundary information. That is, when evaluating  $g(\alpha)$ , we suppress boundary degrees-of-freedom from the matrix–vector product  $Ac$  when calculating  $\|c\|_{A^*A} = \sqrt{\langle Ac, Ac \rangle}$ .

Brent's method [40] is used to minimize (16) and typically only a few iterations are required as  $g(\alpha)$  behaves as in Figure 3.

Table I. GMRES iterations for optimized  $B$ .

$B$	ppw	$h = \frac{1}{127}$	$\frac{h}{2}$	$\frac{h}{4}$	$\frac{h}{8}$	$\frac{h}{16}$
$\exp(i(\omega + \alpha)x)$	5.0	14	20	26	33	36
	10.0	15	14	16	16	19
	30.0	11	11	12	12	13
	90.0	10	10	12	12	12
[ $\cos((\omega + \alpha)x), \sin((\omega + \alpha)x)$ ]	5.0	6	6	6	7	7
	10.0	7	8	7	8	7
	30.0	10	11	9	10	10
	90.0	9	9	10	9	9

### 3.3. Results

Numerical experiments are conducted by implementing the above SA method in the PyAMG package [41]. We employ W(4,4)-cycles to precondition GMRES to a relative residual tolerance of  $1e-8$ . W-cycles were found to be superior to V-cycles because they exhibited grid-independent performance. The grid was evenly spaced points on  $[-1, 1]$ , although the use of other intervals such as  $[0, 1]$  produced essentially the same results. The sparsity pattern of  $ST$ , where  $T$  is the tentative prolongator and  $S$  is a strength-of-connection matrix, i.e.  $k=1$  in Algorithm 3. The test problem is a random initial guess with a zero right-hand side and the maximum size for the coarsest level in a hierarchy was 10 degrees-of-freedom, in order to ensure a true multilevel method.

Our goal is a Helmholtz solver that scales with respect to the wavenumber  $\omega$ . Hence, our testing strategy is to examine the performance on refined grids with constant ppw. For constant ppw, as  $h$  decreases,  $\omega$  increases proportionately.

In Table I, we show the performance using the optimal shift of  $\omega$  from (16) to generate  $B$ . For the more expensive solver, where  $B$  comprises two vectors, a cosine and a sine,  $h$ -independence and ppw-independence is observed. For the less expensive solver, where  $B$  comprises only one vector, the first exponential of (15), deterioration at low ppw is observed. Yet, the solver maintains scalable performance at higher ppw. It is expected that using both the cosine and sine will be superior to using just a single exponential, because the former spans the entire null-space of the unrestricted PDE, whereas the latter does not.

As a comparison, in Table II, we show the performance for both a constant  $B$  and for a  $B$  generated by an unshifted  $\omega$ . As expected, the wave-like  $B$  results in a poor preconditioner because of the wavenumber mismatch between the  $\omega$  from the PDE (1) and the  $\omega$  representative of low-energy modes. The best solver in Table II is for  $B = \mathbf{1}$  at 90 ppw. At this level of resolution, the SA solver should encounter few Nyquist-rate-related problems. Not surprisingly, the performance for  $B = \mathbf{1}$  degrades seriously for less well-resolved problems.

## 4. 2D HELMHOLTZ SOLVER

We use the approach developed in 1D to motivate a more general method for the use on matrices generated from more practical discretizations in 2D. The 1D finite difference discretization and linear continuous Galerkin finite element formulations in general suffer from large dispersion

Table II. GMRES iterations.

$B$	ppw	$h = \frac{1}{127}$	$\frac{h}{2}$	$\frac{h}{4}$	$\frac{h}{8}$	$\frac{h}{16}$
<b>1</b>	5.0	39	61	*	*	*
	10.0	23	38	59	98	*
	30.0	13	16	24	35	58
	90.0	11	11	15	17	24
$\exp(i\omega x)$	5.0	38	58	*	*	*
	10.0	56	94	*	*	*
	30.0	21	32	52	97	*
	90.0	12	13	18	26	43
$[\cos(\omega x), \sin(\omega x)]$	5.0	24	42	81	*	*
	10.0	13	23	40	69	*
	30.0	19	28	43	74	*
	90.0	12	15	22	38	75

\* denotes  $>100$  iterations.

error when approximating wave problems [42–45]. Moving to high-order discretizations in the continuous Galerkin framework alleviates dispersion error greatly [44–46]. There are also new problem-specific discretizations that incorporate wave-like basis functions alongside nodal polynomial basis functions [47–49]. These methods are attractive from an efficiency point of view in that they reduce the dispersion error for a given number of degrees-of-freedom more quickly than an equivalent-order nodal polynomial continuous Galerkin discretization. However, in addition to being highly specialized, these methods can also suffer from stability and conditioning problems.

In this section, we target a DG discretization [50–52]. The dispersion error for DG discretizations is of a higher order in the limit of mesh refinement than for an equivalent order continuous formulation [24]. DG discretizations are flexible with respect to  $h$  and  $p$  refinement, which would be useful for the case of a variable wavenumber  $\omega(x, y)$ . It is also a general method applicable to a large variety of problems. We develop components for the Helmholtz solver in an algebraic fashion with the expectation that they extend to both similar problems and other discretizations of the Helmholtz problem.

#### 4.1. Near null-space selection and algebraic multiple coarsening

The near null-space of the 2D problem requires special attention. The null-space of the unrestricted PDE (1) is extremely rich and consists of an infinite number of planewaves given by the two functions

$$\exp(i\omega(\cos(\theta)x + \sin(\theta)y))$$

and

$$\exp(-i\omega(\cos(\theta)x + \sin(\theta)y)), \quad (17)$$

where  $\theta$  is a given angle of travel. Any  $\theta$  value yields a null-space mode and we define  $B$  based on a collection of modes (17) using regularly spaced angles. A similar approach has been developed in a geometric setting [19, 20].

We therefore seek a balance in the coarse space between complexity and the number of planewaves for different angles. A multiple coarsening strategy, extended from a geometric setting

[20] to SA, is developed. Specifically, we introduce additional grids at coarse levels through new interpolation basis functions. We adapt this approach to SA by appending new columns to  $B_k$  during the construction of the multilevel hierarchy. However, in an algebraic setting, there is no underlying physical grid or angle  $\theta$  at coarser levels. Hence in order to enrich  $B_k$ , we generate the new planewaves on level 0 and restrict them to level  $k$ . To enrich the coarse space, generate new planewaves,  $\hat{B}_k$ , on level 0, relax on  $A_0 \hat{B}_k = 0$ , restrict  $\hat{B}_k$  to the desired level,  $k$ , and then relax on  $A_k \hat{B}_k = 0$ . At this point,  $\hat{B}_k$  is either appended to or replaces  $B_k$ . The final relaxation step is critical because it accounts for any imperfections in restriction that introduced non-algebraically-smooth modes. For an explanation of the initial relaxation step, see Section 4.2.

Owing to the discontinuous elements, we advocate a conservative approach with  $B_0 = \mathbf{1}$ , after which  $B_1$  is replaced by planewaves. Moreover, we aggregate degrees-of-freedom at the same spatial location initially (Section 4.2), resulting in coarse level 1 having the same grid spacing. This avoids introducing aliasing error between the first two levels. If the fine level problem is well-resolved, then it may be beneficial to further delay the enrichment by planewaves; however, we instead advocate a uniform strategy for all resolutions.

Planewaves are only introduced at levels 1 and 2.  $B_1$  is replaced with planewaves  $\hat{B}_1$  and we append to  $B_2$  yielding  $B_2 \leftarrow [B_2, \hat{B}_2]$ . Planewaves added at levels coarser than 2 do not further enrich the coarse space for the problems considered here. If projected to the finest level, such coarse level planewaves exhibit little wave-like behavior. As a summary, Table III presents the hierarchy of  $B$ s used in this work.

Let  $\Theta_k \subset [0, \pi)$  denote the set of regularly spaced angles used to generate  $\hat{B}_k$ . We determine  $\Theta_{k+1}$  from  $\Theta_k$ , based on the geometric approach [20]. Each angle in  $\Theta_k$  is perturbed by both adding and subtracting a fixed amount, so that the number of angles is doubled when going from  $\Theta_k$  to  $\Theta_{k+1}$  and so that the spacing in  $\Theta_{k+1}$  is itself even. In this way, angles are not repeated on subsequent levels. For example, the spacing at level 1 is quartered when going to level 2, and then halved when going to each subsequent level. Although we only introduce planewaves at two levels here, the process of determining  $\Theta_k$  is general to any level. The two  $\Theta$  hierarchies used in this work are given in Table IV.

Table III.  $B$  hierarchy.

Level	0	1	2	3	...
$B$ update	$B_0 \leftarrow \mathbf{1}$	$B_1 \leftarrow [\hat{B}_1]$	$B_2 \leftarrow [B_2, \hat{B}_2]$	$B_3 \leftarrow B_3$	...
	constant-based interpolation	constant-based $B_1$ overwritten	append new planewaves	no change	...

Table IV. Suitable  $\Theta$  spacings.

Level	$\Theta_1$	$\Theta_2$	...
	$[0, \frac{\pi}{2}]$	$[-\frac{\pi}{8}, \frac{\pi}{8}, \frac{3\pi}{8}, \frac{5\pi}{8}]$	...
	$[0, \frac{\pi}{3}, \frac{2\pi}{3}]$	$[-\frac{\pi}{12}, \frac{\pi}{12}, \dots, \frac{9\pi}{12}]$	...

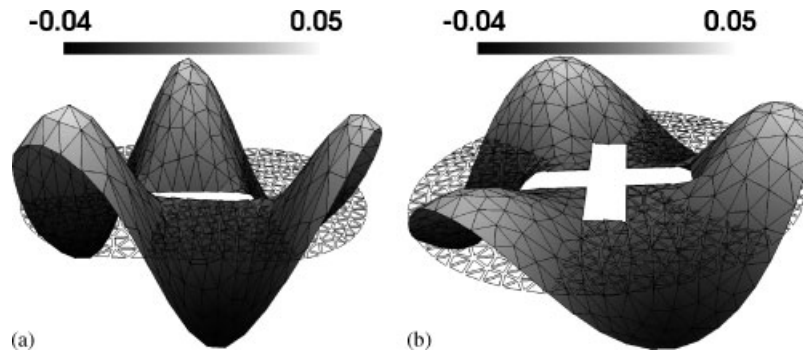


Figure 4. Example lowest right singular vectors: (a) airfoil domain and (b) cross domain.

Generating planewaves from a predetermined set of angles,  $\Theta_k$  is simple. As with the 1D solver, we find it beneficial to use the simplest basis of the null-space (17) as possible. We therefore construct  $B$  by using the real and imaginary parts of the first exponential in (17) separately. This provides us with a basis for (17) for a given set of angles,  $\Theta_k$ . In addition, both the real and the imaginary parts of a planewave are themselves null-space modes of the unrestricted PDE (1). This is analogous to the 1D strategy of using the cosine and sine as separate null-space functions when constructing  $B$ . As an example, if  $\Theta_1 = [0, \pi/2]$ , then  $\hat{B}_1$  contains four vectors corresponding to  $[\text{real}(\exp(i\omega\mathbf{x})), \text{imag}(\exp(i\omega\mathbf{x})), \text{real}(\exp(i\omega\mathbf{y})), \text{imag}(\exp(i\omega\mathbf{y}))]$ , where  $\mathbf{x} = x$  and  $\mathbf{y} = \cos(\pi/2)x + \sin(\pi/2)y$ .

To facilitate the understanding of the near null-space, we present in Figure 4 the real part of example lowest right singular vectors for two domains, an airfoil and a cross-shaped scatter. For these relatively coarse examples, the wavenumber is correspondingly small, but the wave-like nature of the near null-space is nonetheless apparent. The modes are, as expected, compositions of planewaves that satisfy the boundary conditions.

#### 4.2. Solver components

In this section, we detail the multilevel components, such as aggregation, prolongation, restriction, and relaxation. We motivate our construction in 2D from the 1D case, and specifically target discontinuous elements with nodal linear basis functions.

Aggregation in 2D in a discontinuous setting is not immediately obvious. We mimic the coarsening generated by the Evolution Measure [34] for discontinuous discretizations of isotropic Poisson problems. At level 0, degrees-of-freedom at the same spatial location in adjacent elements are aggregated together. Aggregation is explicitly set at level 0, since it is straightforward to compute the desired aggregates and since using the Evolution Measure in an indefinite setting requires more expensive iteration with the normal equations. In Figure 5, an example of level 0 aggregation is given. The triangular mesh is represented by solid black lines, mesh points are represented as squares and the shaded polygons are aggregates. The mesh is represented visually as being discontinuous by shrinking each element toward its barycenter, so that overlapping edges are visually side-by-side. Standard aggregation based on the matrix graph is then used on subsequent levels. Another difference in 2D is the fitness of the near null-space for the discrete problem. In 1D,

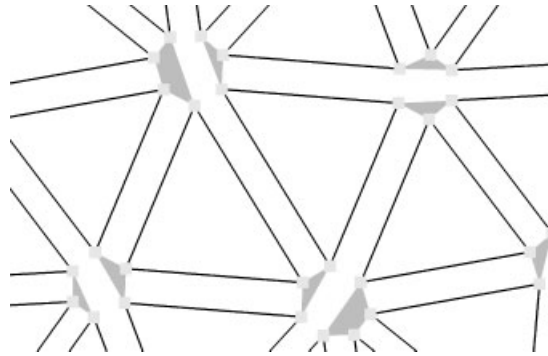


Figure 5. Sample aggregates.

we utilize (16) to find an appropriate wavenumber  $\omega + \alpha$  for generating  $B$ . However, DG discretizations are superior to standard finite-differencing for wave-like problems. Hence, the wavenumber implied by the PDE (1) and the wavenumber of low-energy modes are more compatible. On the other hand, any  $B$  incorporated into the coarse space still requires adaptation to the matrix. Specifically, DG discretizations leave all Dirichlet degrees-of-freedom in the matrix and enforce the Dirichlet conditions weakly. It is necessary to amend  $B$  to properly represent the boundary conditions in the algebraically smooth error, especially because Algorithm 3 will exactly incorporate  $B$  into the coarse space. For this problem, relaxing on  $A_0 \hat{B}_k = 0$  sufficiently enforces the boundary conditions on  $B$ .

The 2D solver uses the same energy-minimizing prolongation smoother algorithm for prolongation, but with a wider interpolation stencil to account for the more rapid coarsening that occurs for the wide matrix stencils resulting from the DG discretizations. The sparsity pattern constraint on  $P$  is extended to the sparsity pattern of  $SST$ , where  $T$  is the tentative prolongator and  $S$  is a strength-of-connection matrix, i.e.  $k=2$  in Algorithm 3 for all 2D experiments. For level 0,  $S$  contains connections only between degrees-of-freedom at the same spatial location and in adjacent elements, and for levels 1 and greater,  $S=A$ . Algebraic multiple coarsening corrupts the near null-space preservation property preserved by the 1D solver in that the span of the product of all prolongation operators,  $\text{span}(P_0 P_1 \dots P_k)$ , exactly preserves  $B_0$ . However, the basic mechanics of constructing  $P$  remain the same and the constraint,  $P_k B_{k+1} = B_k$ , is still enforced. Restriction continues to be the non-Hermitian transpose of  $P$ , because extending the interpolation stencil has not affected Lemma 1.

Finally for relaxation, a mixed scheme is used with standard Gauss–Seidel on level 0 and Gauss–Seidel NR on levels 1 and greater. Standard Gauss–Seidel is an effective smoother for resolved Helmholtz problems [18, 19], but diverges quickly for unresolved Helmholtz problems.

#### 4.3. Algorithm

We now detail the algorithm used to generate the multilevel hierarchy for our 2D Helmholtz solver. The algorithm constructs a standard multigrid hierarchy of coarse level matrices and prolongation operators. For comparison, it is useful to note the changes made from the standard hierarchy construction Algorithm 1. After the hierarchy construction, Algorithm 2 is used for the solve phase.

Algorithm 4: sa\_helmholtz\_hierarchy ( $A, E2V, V, \Theta$ )

---

```

1 #  $A$  := Discrete Helmholtz operator
2 #  $E2V$  := Element-to-vertex map for level 0 aggregation
3 #  $V$  := Vertex list
4 #  $\Theta$  :=  $[\Theta_k]$ , a set angles for planewaves per Table IV
5
6  $A_0 = A$ 
7  $B_0 = \mathbf{1}$ 
8  $k = 0$ 
9 while size( $A_k$ ) > 100
10   if  $k$  is 1 or 2
11      $\hat{B}_k = [\text{planewaves}(V, \Theta_k)]$  # Generate planewaves with (17)
12      $\hat{B}_k = [\text{real}(\hat{B}_k), \text{imag}(\hat{B}_k)]$  # Split into real and imag parts
13     Gauss-Seidel NR on  $A_0 \hat{B}_k = 0$ 
14     for  $j = 0, 1, \dots, k-1$ 
15        $\hat{B}_k = R_j \hat{B}_k$ 
16     Gauss-Seidel NR on  $A_k \hat{B}_k = 0$ 
17
18     if  $k$  is 1
19        $B_k = \hat{B}_k$  # Replace constant
20     else
21        $B_k = [B_k, \hat{B}_k]$  # Append to other planewaves
22
23     if  $k$  is 0
24        $S_k = \text{Strength}(E2V, V)$  # Connect spatial dofs
25     else
26        $S_k = \text{Strength}(A_k)$  #  $S_k$  has same graph as  $A_k$ 
27
28      $C_k = \text{Aggregate}(S_k)$ 
29      $T_k, B_{k+1} = \text{InjectCandidates}(C_k, B_k)$ 
30      $P_k = \text{energy\_minimization}(A_k, T_k, S_k)$ 
31      $A_{k+1} = P_k^T A_k P_k$  #  $A$  is complex-symmetric
32      $k = k + 1$ 
33 return  $A_0, \dots, A_k, P_0, \dots, P_{k-1}$ 

```

---

Our approach differs in a number of ways from the approaches in [19, 20]. As opposed to the approach in [20], we consider indefinite formulations of the problem. In addition, the methods [19, 20] are geometric in nature and attractive for regular grids. Moreover, they are also not designed for DG discretizations. Implementing either of these methods requires extensive modifications to any existing AMG code by introducing new components to the AMG hierarchy and yet, the functionality is limited to Helmholtz problems. In [19], for example, a new ray-cycle is proposed. Moreover in [20], the multiple coarsening proposed would require extensive changes to interpolation, as opposed to a simple appending of new vectors to the existing  $B$  as is done here. In addition, the proposed solver when interpolating for any given degree-of-freedom, does so with respect to all planewaves, as opposed to treating planewaves separately. In summary, the proposed solver is broader given its algebraic nature and more practical since it requires only straightforward modification to an existing SA framework.

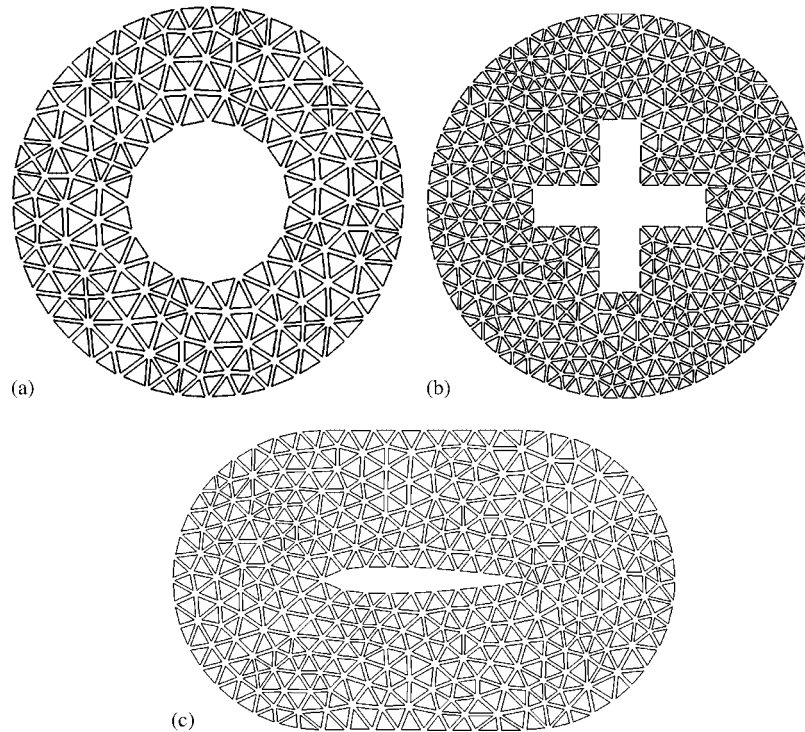


Figure 6. Example meshes: (a) coarse annulus mesh; (b) coarse cross mesh; and (c) coarse airfoil mesh.

#### 4.4. Results

In this section, we examine the solver's performance for three types of scattering problems on unstructured simplicial meshes with linear discontinuous FEs. The strong form of the local DG method (LDG) [50, 51] is used to discretize the Laplacian part of the PDE (1). The Sommerfeld boundary condition is incorporated into the formulation as one would a Robin boundary condition. In general, DG methods are known [52] to be effective for the Laplace eigenvalue problem.

One test problem is for the annulus-shaped domain in Figure 1 with a sample coarse mesh shown in Figure 6(a). The other two test problems are for an airfoil and a cross-shaped scatterer, with sample coarse meshes shown in Figures 6(c) and (b), respectively. For all test problems, the coarsest discretization denoted by  $h$  had around 400 degrees-of-freedom and the finest discretization denoted by  $h/12$  had around 80 000 degrees-of-freedom. Tests are conducted using PyAMG [41] for GMRES preconditioned by V(4,4), W(4,4), and W(2,2) cycles. The maximum size for the coarsest level in a hierarchy was 100 degrees-of-freedom, in order to ensure a true multilevel method.

In Table V, we report the diameter of the three domains for the highest frequency (5.0 ppw) and lowest frequency (90.0 ppw) on the finest grid ( $h/12$ ). The diameter of the circular-shaped annulus and cross domains is clear, but for the airfoil domain, the diameter is taken at the widest point of the domain.

We again examine the solver's performance as the mesh is refined and ppw is held constant. In this way, we test for a solver's scalability with respect to increasing wavenumber. In calculating

Table V. Diameter of domain in wavelengths for finest grids.

	5.0 ppw	90.0 ppw
Airfoil	19.6	1.09
Annulus	18.7	1.03
Cross	17.2	0.95

ppw, we define mesh size to be the longest edge length. This is an appropriate choice, because longest edge length will be the limiting factor by which an arbitrary planewave can be represented and is a pessimistic view of resolution. All of the meshes are isotropic, so that the longest edge length does not vary greatly from other measures such as inscribed circles.

Table VI presents the GMRES iterations using V(4,4), W(4,4), and W(2,2) cycles for both sets of angle hierarchies, which we denote by their level 1 spacings,  $\Theta_1 = [0, \pi/2]$  and  $\Theta_1 = [\pi, \pi/3, 2\pi/3]$ . The proposed solver appears to be scalable with respect to ppw for moderately to highly resolved problems, when using W(4,4) and W(2,2) cycles. There is, however, some degradation at low ppw, especially for the V-cycle. As expected, using  $\Theta_1 = [\pi, \pi/3, 2\pi/3]$  as the initial planewave angles is more robust at low ppw, where the near null-space is richest. Preconditioning with the W(4,4) cycle exhibits the best convergence, but is also the most expensive. Yet at higher ppw, the W(2,2) cycle appears to yield a suitable method for a much lower complexity than the W(4,4) cycle. The results for the annulus- and cross-shaped scatters are similar to those for the airfoil scatterer, which highlights the robustness of our method. We present abbreviated results for these problems in Table VI.

The stand-alone SA solver scales with ppw for the highly resolved problems. However, for low ppw, the stand-alone solver becomes erratic. In general, we find that preconditioning GMRES with the SA solver provides much more robustness.

One drawback of the W-cycling is the complexity as shown in Table VII. However, growth in these complexities is low as the mesh is refined. Cycle complexity is calculated as the total cost of relaxation at all levels for one cycle divided by the cost of one Gauss–Seidel or weighted-Jacobi sweep on the finest level. Cycle complexity is the appropriate way to measure cost when using multiple relaxation sweeps on a level. While the operator complexity only considers the number of non-zeros in the operators for each level, the cycle complexity essentially weights the number of non-zeros on each level by the number of relaxation steps taken on that level. Cycle complexity is also the appropriate measure of cost here because it accurately conveys the cost of using Gauss–Seidel NR, which costs roughly twice as much as Gauss–Seidel or weighted-Jacobi.

Despite the high cycle complexities, the SA setup time and the preconditioned solve times are commensurate for our implementation at the resolutions of 10.0, 30.0, and 90.0 ppw. In part, this reflects the fact that the prolongation smoothing strategy of Algorithm 3 is computationally more expensive than standard prolongation smoothing. However, it is typical for multigrid algorithms to have similar setup and solve times.

To better understand the AMG setup cost, Table VIII presents the relative cost of the prolongation smoothing strategy of Algorithm 3, presmoothing the near null-space vectors  $B$ , and computing  $RAP$ . The relative cost is presented for each level during the construction of the hierarchy for the finest airfoil scattering problem. These results are for our hybrid Python/C++ implementation in PyAMG on a 32-bit single core Pentium IV 3.06 GHz machine with 3 GB of main memory.

Table VI. GMRES iterations.

Problem	Solver	ppw	$h$	$\frac{h}{2}$	$\frac{h}{4}$	$\frac{h}{8}$	$\frac{h}{12}$		
Airfoil	V(4,4) $\Theta_1 = [0, \frac{\pi}{2}]$	5.0	7	9	16	33	61		
		10.0	7	9	12	12	15		
		30.0	7	8	10	12	15		
		90.0	7	8	9	11	13		
	$\Theta_1 = [0, \frac{\pi}{3}, \frac{2\pi}{3}]$	V(4,4)	5.0	7	10	14	17	28	
		10.0	7	8	11	11	15		
		30.0	7	8	10	14	17		
		90.0	7	8	9	11	14		
	$\Theta_1 = [0, \frac{\pi}{2}]$	W(4,4)	5.0	7	8	12	18	40	
		10.0	7	8	9	8	10		
		30.0	7	7	8	8	8		
		90.0	7	7	8	8	8		
	$\Theta_1 = [0, \frac{\pi}{3}, \frac{2\pi}{3}]$	W(4,4)	5.0	7	10	13	11	20	
		10.0	7	7	9	8	9		
		30.0	7	7	8	8	8		
		90.0	7	7	8	8	9		
	$\Theta_1 = [0, \frac{\pi}{2}]$	W(2,2)	5.0	9	11	15	25	54	
		10.0	9	11	11	11	14		
		30.0	9	10	10	11	11		
		90.0	9	10	11	11	11		
	Annulus	$\Theta_1 = [0, \frac{\pi}{3}, \frac{2\pi}{3}]$	W(4,4)	5.0	7	11	7	10	20
			10.0	7	8	7	7	10	
			30.0	7	7	7	8	8	
			90.0	7	7	7	7	8	
$\Theta_1 = [0, \frac{\pi}{2}]$		W(2,2)	5.0	9	13	11	25	55	
		10.0	9	12	10	11	15		
		30.0	9	10	10	11	11		
		90.0	9	10	10	11	11		
Cross	$\Theta_1 = [0, \frac{\pi}{3}, \frac{2\pi}{3}]$	W(4,4)	5.0	7	10	8	10	15	
		10.0	7	8	8	8	8		
		30.0	7	7	7	8	9		
		90.0	7	7	8	8	8		
	$\Theta_1 = [0, \frac{\pi}{2}]$	W(2,2)	5.0	9	11	11	20	45	
		10.0	9	11	10	11	12		
		30.0	9	10	10	11	12		
		90.0	9	10	10	11	11		

Computing *RAP* is usually one of the most expensive operations when constructing a multigrid hierarchy. Now, Algorithm 3 and presmoothing the near null-space vectors *B* dominate the AMG setup phase. However, the setup cost still scales linearly with the problem size.

Table VII. Cycle complexity.

Problem	Solver	$h$	$\frac{h}{2}$	$\frac{h}{4}$	$\frac{h}{8}$	$\frac{h}{12}$
Airfoil	$V(4,4)$ $\Theta_1 = [0, \frac{\pi}{2}]$	8.2	15.3	16.1	17.0	17.3
	$V(4,4)$ $\Theta_1 = [0, \frac{\pi}{3}, \frac{2\pi}{3}]$	8.2	20.4	22.2	24.2	24.8
	$W(4,4)$ $\Theta_1 = [0, \frac{\pi}{2}]$	8.2	30.7	34.0	41.7	43.4
	$W(4,4)$ $\Theta_1 = [0, \frac{\pi}{3}, \frac{2\pi}{3}]$	8.2	51.2	58.4	75.5	79.4
	$W(2,2)$ $\Theta_1 = [0, \frac{\pi}{2}]$	4.2	15.4	17.0	20.8	21.7
Annulus	$W(4,4)$ $\Theta_1 = [0, \frac{\pi}{3}, \frac{2\pi}{3}]$	15.0	58.4	79.9	79.0	73.9
	$W(2,2)$ $\Theta_1 = [0, \frac{\pi}{2}]$	7.5	17.1	21.8	21.8	20.6
Cross	$W(4,4)$ $\Theta_1 = [0, \frac{\pi}{3}, \frac{2\pi}{3}]$	8.2	51.3	81.6	77.2	80.3
	$W(2,2)$ $\Theta_1 = [0, \frac{\pi}{2}]$	4.2	15.3	22.2	21.2	21.9

Table VIII. AMG setup timings.

Operation	% of setup (s)		
	Level 0	Level 1	Level 2
<i>RAP</i>	1.5 (0.24)	2.2 (0.37)	2.2 (0.37)
smooth( <i>P</i> )	17.0 (2.8)	23.0 (3.7)	23 (3.8)
smooth( <i>B</i> )	6.4 (1.0)	16.0 (2.6)	1e-5 (2e-5)

## 5. CONCLUSIONS

In conclusion, we have outlined algebraic solvers for the 1D and 2D Helmholtz problem. The solvers each generate a standard multigrid hierarchy that uses a fixed moderate amount of smoothing on each level. The 1D solver is applicable to both poorly- and well-resolved problems, whereas the 2D solver is applicable to moderately- and well-resolved problems. These are the first such algebraic solvers known to the authors, for a nodal discretization.

The first coarse grid of Algorithm 4 is very similar to a linear continuous Galerkin discretization of the same problem. It is therefore easy to extend Algorithm 4 to the continuous case where we

have observed similar convergence results. Algorithm 4 is modified such that the hierarchy setup begins at level 1 and not level 0.

SA has been extended to address the following difficulties that are raised by the Helmholtz problem and are beyond the classic SAs scope: indefiniteness, the rich wave-like near null-space, a lack of Hermitian symmetry, and the complex-valued matrix. This task is divided into four general areas: strength-of-connection (i.e. aggregation), prolongation smoothing, relaxation, and accounting for a rich non-standard wave-like near null-space. The strategy behind our techniques begins with the complementary relationship between interpolation and relaxation that is at the heart of any effective multigrid solver. Hence, the techniques developed here are not limited to only the Helmholtz problem. The aggregation scheme should be applicable to isotropic-like problems with a DG discretization. The energy-minimizing prolongation smoother advocated here requires only a generic tentative prolongator, corresponding set of near null-space modes, and the strength-of-connection matrix. The relaxation methods are not problem specific and the algebraic multiple coarsening procedure could be used to capture any rich, but not necessarily wave-like, near null-space.

Prolongation smoothing for non-Hermitian and indefinite problems is an open problem, and we believe that this method is a useful contribution. The proposed prolongation smoothing algorithm is an extension of [2] to the indefinite, non-Hermitian case through the use of the  $A^*A$  norm. Moreover, a constrained Krylov method is used as the minimization algorithm.

Our approach to generating the near null-space modes,  $B$ , is effective. SA generally has two strategies for generating  $B$ . There is the purely adaptive approach [39], which can unfortunately be expensive. The other approach uses the null-space of the unrestricted PDE, which can be inaccurate, especially near the boundaries. The method for generating  $B$  here combines both approaches and adapts *a priori* knowledge about the near null-space to the matrix.

We propose a novel method of algebraic multiple coarsening to account for the richness of the 2D problem's near null-space. The multiple coarsening approach allows the solver to effectively translate the residual equation into a wave-space thereby allowing for further reduction of the residual. One of the motivations for this algebraic approach is that  $\hat{B}_k$  could be any function, not just planewaves. From this point of view, algebraic multiple coarsening could prove useful for other problems with a rich near null-space. If applied to other problems, there are modifications which may be needed.  $B_k$  could be enlarged at levels greater than 2. In addition, other problems or restriction strategies may require relaxation for  $\hat{B}_k$  after each restriction operation as opposed to only relaxing after restriction to the desired level,  $k$ .

#### ACKNOWLEDGEMENTS

We thank Dr Raymond Tuminaro of Sandia National Laboratories for guidance in a CG version of Algorithm 3 that motivated the current state.

#### REFERENCES

1. Vaněk P, Mandel J, Brezina M. Algebraic multigrid based on smoothed aggregation for second and fourth order problems. *Computing* 1996; **56**:179–196.
2. Mandel J, Brezina M, Vaněk P. Energy optimization of algebraic multigrid bases. *Computing* 1999; **62**: 205–228.
3. Vaněk P, Brezina M, Mandel J. Convergence of algebraic multigrid based on smoothed aggregation. *Numerische Mathematik* 2001; **88**(3):559–579.

4. Brandt A, McCormick SF, Ruge JW. Algebraic multigrid (AMG) for sparse matrix equations. In *Sparsity and Its Applications*, Evans DJ (ed.). Cambridge University Press: Cambridge, 1984; 257–284.
5. Ruge JW, Stüben K. Algebraic multigrid (AMG). In *Multigrid Methods*, McCormick SF (ed.). Frontiers in Applied Mathematics. SIAM: Philadelphia, PA, 1987; 73–130.
6. Lahaye D, De Gerssem H, Vandewalle S, Hameyer K. Algebraic multigrid for complex symmetric systems. *IEEE Transactions on Magnetics* 2000; **36**(4):1535–1538. DOI: 10.1109/20.877730.
7. Day D, Heroux MA. Solving complex-valued linear systems via equivalent real formulations. *SIAM Journal on Scientific Computing* 2001; **23**(2):480–498. DOI: <http://dx.doi.org/10.1137/S1064827500372262>.
8. MacLachlan SP, Oosterlee CW. Algebraic multigrid solvers for complex-valued matrices. *SIAM Journal on Scientific Computing* 2008; **30**(3):1548–1571. DOI: <http://dx.doi.org/10.1137/070687232>.
9. Sala M, Tuminaro RS. A new Petrov–Galerkin smoothed aggregation preconditioner for nonsymmetric linear systems. *SIAM Journal on Scientific Computing* 2008; **31**(1):143–166. DOI: <http://dx.doi.org/10.1137/060659545>.
10. Brezina M, Manteuffel T, McCormick S, Ruge J, Sanders G. Towards adaptive smoothed aggregation ( $\alpha$ SA) for nonsymmetric problems. *SIAM Journal on Scientific Computing* 2009; submitted.
11. Erlangga YA, Vuik C, Oosterlee CW. On a class of preconditioners for solving the Helmholtz equation. *Applied Numerical Mathematics* 2004; **50**(3–4):409–425. DOI: <http://dx.doi.org/10.1016/j.apnum.2004.01.009>.
12. Erlangga YA, Oosterlee CW, Vuik C. A novel multigrid based preconditioner for heterogeneous Helmholtz problems. *SIAM Journal on Scientific Computing* 2006; **27**:1471–1492.
13. Airaksinen T, Heikkola E, Pennanen A, Toivanen J. An algebraic multigrid based shifted-Laplacian preconditioner for the Helmholtz equation. *Journal of Computational Physics* 2007; **226**(1):1196–1210.
14. Airaksinen T, Pennanen A, Toivanen J. A damping preconditioner for time-harmonic wave equations in fluid and elastic material. *Journal of Computational Physics* 2009; **228**(5):1466–1479.
15. Bollhöfer M, Grote MJ, Schenk O. Algebraic multilevel preconditioner for the Helmholtz equation in heterogeneous media. *SIAM Journal on Scientific Computing* 2009; **31**(5):3781–3805. DOI: 10.1137/080725702. Available from: <http://link.aip.org/link/?SCE/31/3781/1>.
16. Shapira Y. Multigrid techniques for highly indefinite equations. *The 8th Copper Mountain Conference on Multigrid Methods*, Copper Mountain, Colorado, 1995; 689–705.
17. Elman HC, Ernst OG, O’Leary DP. A multigrid method enhanced by Krylov subspace iteration for discrete Helmholtz equations. *SIAM Journal on Scientific Computing* 2001; **23**(4):1291–1315. DOI: <http://dx.doi.org/10.1137/S1064827501357190>.
18. Brandt A, Ta’asan S. Multigrid method for nearly singular and slightly indefinite problems. In *Multigrid Methods II*, Hackbusch W, Trottenberg U (eds). Lecture Notes in Mathematics. Springer: Berlin, 1986; 99–121. DOI: <http://dx.doi.org/10.1007/BFb0072643>.
19. Brandt A, Livshits I. Wave-ray multigrid method for standing wave equations. *Transactions on Numerical Analysis* 1997; **6**:162–181.
20. Lee B, Manteuffel TA, McCormick SF, Ruge J. First-order system least-squares for the Helmholtz equation. *SIAM Journal on Scientific Computing* 2000; **21**(5):1927–1949. DOI: <http://dx.doi.org/10.1137/S1064827598339773>.
21. Vaněk P, Mandel J, Brezina M. Two-level algebraic multigrid for the Helmholtz problem. *Tenth International Conference on Domain Decomposition, Volume 218 of Contemporary Mathematics*. American Mathematical Society: Providence, RI, 1998; 349–356.
22. Bramble JH, Pasciak JE, Xu J. The analysis of multigrid algorithms for nonsymmetric and indefinite elliptic problems. *Mathematics of Computation* 1988; **51**(184):389–414.
23. Bramble JH, Kwak DY, Pasciak JE. Uniform convergence of multigrid V-cycle iterations for indefinite and nonsymmetric problems. *SIAM Journal on Scientific Computing* 1994; **31**(6):1746–1763. DOI: <http://dx.doi.org/10.1137/0731089>.
24. Ainsworth M. Dispersive and dissipative behaviour of high order discontinuous Galerkin finite element methods. *Journal of Computational Physics* 2004; **198**(1):106–130. DOI: <http://dx.doi.org/10.1016/j.jcp.2004.01.004>.
25. Atkins H, Helenbrook B. Application of  $p$ -multigrid to discontinuous Galerkin formulations of the Poisson operator. *AIAA Computational Fluid Dynamics Conference*, Toronto, Ontario, 2005.
26. Fidkowski KJ, Oliver TA, Lu J, Darmofal DL.  $p$ -multigrid solution of high-order discontinuous Galerkin discretizations of the compressible Navier–Stokes equations. *Journal of Computational Physics* 2005; **207**(1): 92–113. DOI: 10.1016/j.jcp.2005.01.005. Available from: <http://www.sciencedirect.com/science/article/B6WHY-4FJV22R-2/%20ef0030688764d90948ea4f8eb00e5090>.

27. Dobrev VA, Lazarov RD, Vassilevski PS, Zikatanov LT. Two-level preconditioning of discontinuous Galerkin approximations of second-order elliptic equations. *Numerical Linear Algebra with Applications* 2006; **13**(9): 753–770. DOI: <http://dx.doi.org/10.1002/nla.504>.
28. Kanschat G. Preconditioning methods for local discontinuous Galerkin discretizations. *SIAM Journal on Scientific Computing* 2003; **25**(3):815–831. DOI: 10.1137/S1064827502410657. Available from: <http://link.aip.org/link/?SCE/25/815/1>.
29. Gopalakrishnan J, Kanschat G. A multilevel discontinuous Galerkin method. *Numerische Mathematik* 2003; **95**(3):527–550.
30. Hemker PW, Hoffmann W, Raalte MHv. Two-level Fourier analysis of a multigrid approach for discontinuous Galerkin discretization. *SIAM Journal on Scientific Computing* 2003; **25**(3):1018–1041. DOI: <http://dx.doi.org/10.1137/S1064827502405100>.
31. Bochev PB, Garasi CJ, Hu JJ, Robinson AC, Tuminaro RS. An improved algebraic multigrid method for solving Maxwell's equations. *SIAM Journal on Scientific Computing* 2003; **25**(2):623–642. DOI: <http://dx.doi.org/10.1137/S1064827502407706>.
32. Hu JJ, Tuminaro RS, Bochev PB, Garasi CJ, Robinson AC. Toward an h-independent algebraic multigrid method for Maxwell's equations. *SIAM Journal on Scientific Computing* 2006; **27**(5):1669–1688. DOI: <http://dx.doi.org/10.1137/040608118>.
33. Bell W, Olson L. Algebraic multigrid for k-form Laplacians. *Numerical Linear Algebra with Applications* 2008; **15**(2–3):165–185.
34. Olson L, Schroder J, Tuminaro R. A new perspective on strength measures in algebraic multigrid. *Numerical Linear Algebra with Applications* 2008; DOI: 10.1002/nla.669.
35. Olson L. Algebraic multigrid preconditioning of high-order spectral elements for elliptic problems on a simplicial mesh. *SIAM Journal on Scientific Computing* 2007; **29**(5):2189–2209 (electronic).
36. Brannick J, Brezina M, MacLachlan S, Manteuffel T, McCormick S, Ruge J. An energy-based AMG coarsening strategy. *Numerical Linear Algebra with Applications* 2006; **13**(2–3):133–148.
37. Saad Y. *Iterative Methods for Sparse Linear Systems*. SIAM: Philadelphia, PA, 2003.
38. Saunders MA, Simon HD, Yip EL. Two conjugate-gradient-type methods for unsymmetric linear equations. *SIAM Journal on Numerical Analysis* 1988; **25**(4):927–940. DOI: 10.1137/0725052. Available from: <http://link.aip.org/link/?SNA/25/927/1>.
39. Brezina M, Falgout R, MacLachlan S, Manteuffel T, McCormick S, Ruge J. Adaptive smoothed aggregation ( $\alpha$ SA) multigrid. *SIAM Review* 2005; **47**(2):317–346 (electronic).
40. Brent R. *Algorithms for Minimization without Derivatives*. Prentice-Hall: Englewood Cliffs, NJ, 1973.
41. Bell WN, Olson LN, Schroder J. PyAMG: algebraic multigrid solvers in Python v1.0, 2008. Available from: <http://www.pyamg.org>.
42. Babuska IM, Sauter SA. Is the pollution effect of the FEM avoidable for the Helmholtz equation considering high wave numbers? *SIAM Journal on Numerical Analysis* 1997; **34**(6):2392–2423. DOI: <http://dx.doi.org/10.1137/S0036142994269186>.
43. Ihlenburg F, Babuska I. Finite element solution of the Helmholtz equation with high wave number part I: the h-version of the FEM. *Computers and Mathematics with Applications* 1995; **30**(9):9–37. DOI: 10.1016/0898-1221(95)00144-N.
44. Ihlenburg F, Babuska I. Finite element solution of the Helmholtz equation with high wave number part ii: the h-p version of the FEM. *SIAM Journal on Numerical Analysis* 1997; **34**(1):315–358. DOI: <http://dx.doi.org/10.1137/S0036142994272337>.
45. Harari I. Reducing spurious dispersion, anisotropy and reflection in finite element analysis of time-harmonic acoustics. *Computer Methods in Applied Mechanics and Engineering* 1997; **140**(1–2):39–58. DOI: 10.1016/S0045-7825(96)01034-1.
46. Ainsworth M. Discrete dispersion relation for hp-version finite element approximation at high wave number. *SIAM Journal on Numerical Analysis* 2004; **42**(2):553–575. DOI: <http://dx.doi.org/10.1137/S0036142903423460>.
47. Strouboulis T, Babuska I, Hidajat R. The generalized finite element method for Helmholtz equation: theory, computation, and open problems. *Computer Methods in Applied Mechanics and Engineering* 2006; **195**(37–40):4711–4731. DOI: 10.1016/j.cma.2005.09.019.
48. Farhat C, Tezaur R, Weidemann-Goiran P. Higher-order extensions of a discontinuous Galerkin method for mid-frequency Helmholtz problems. *International Journal for Numerical Methods in Engineering* 2004; **61**(11): 1938–1956. DOI: <http://dx.doi.org/10.1002/nme.1139>.

49. Gittelsohn CJ, Hiptmair R, Perugia I. Plane wave discontinuous Galerkin methods: analysis of the  $h$ -version. *M2AN Mathematical Model in Numerical Analysis* 2009; **43**(2):297–331.
50. Castillo P, Cockburn B, Perugia I, Schötzau D. Local discontinuous Galerkin methods for elliptic problems. *Communications in Numerical Methods in Engineering* 2002; **18**(1):69–75. DOI: <http://dx.doi.org/10.1002/cnm.471>.
51. Arnold DN, Brezzi F, Cockburn B, Marini LD. Unified analysis of discontinuous Galerkin methods for elliptic problems. *SIAM Journal on Numerical Analysis* 2002; **39**:1749–1779.
52. Antonietti PF, Buffa A, Perugia I. Discontinuous Galerkin approximation of the Laplace eigenproblem. *Computer Methods in Applied Mechanics and Engineering* 2006; **195**(25–28):3483–3503.