Improving Robustness in Multigrid Methods

Scott MacLachlan

September 30, 2002

Abstract

An important constraint on our ability to numerically simulate physical processes is our ability to solve the resulting linear systems. Multiscale methods, such as multigrid, provide optimal or near optimal order solution techniques for a wide range of problems. The biggest drawback to current multigrid methods (both geometric and algebraic) is their fragility. Classical geometric multigrid methods are effective only in simple geometries and only for a relatively small class of problems. Algebraic multigrid methods are free from most constraints on geometry, but they are still only effective for a relatively small class of problems. We seek to identify current roadblocks to obtaining optimal multigrid efficiency and use this knowledge to gain insight into the operation of current methods. We also propose to investigate the construction of more robust multigrid methods.

1 Introduction

We are interested in the numerical solution of the partial differential equations resulting from the mathematical modeling of physical systems. We assume that these PDEs have already been discretized in a sensible manner through the use of finite differences or finite elements [2], and our primary focus is on the efficient solution of the linear systems that arise from such discretizations. These systems are typically large (current problems of interest involve millions or even billions of degrees of freedom (dofs)), sparse (a fixed number of non-zero entries per row or column, regardless of problem size), and ill-conditioned (with condition number approaching zero as problem size increases).

Classical linear solvers are quite impractical for these problems ([17], §1.4). Consider, for example, a three-dimensional discretization of the Poisson Equation on a grid of size $N = n \times n \times n$. The matrix resulting from a trilinear finite element discretization of this problem has up to 27 non-zero entries per row or column, but a bandwidth of n^2. Thus, full Gaussian Elimination would take $O(N^3) = O(n^9)$ operations, and Gaussian Elimination only within the band still requires $O(n^7) = O(N^{\frac{7}{3}})$ operations. For even moderately sized n, this cost is obviously prohibitive. Stationary iterative methods, such as Jacobi, Gauss-Seidel, and SOR are typically more efficient than Gaussian Elimination, but are still prohibitively expensive. It can be shown that Jacobi and Gauss-Seidel for the Poisson problem take $O(n^2)$ iterations to converge to a level of accuracy proportional to discretization error, giving a total cost then of $O(n^5) = O(N^{\frac{5}{3}})$. Choosing the optimal over-relaxation parameter ω for SOR results in a method that requires $O(n^4) = O(N^{\frac{4}{3}})$ operations once ω is known. Krylov subspace methods, such as the Conjugate Gradient Algorithm, are also typically more efficient than Gaussian Elimination, with CG requiring $O(n^4)$ operations in its simplest form. Preconditioning can be used to improve the performance of CG, and some incomplete factorization based preconditioners can result in a cost as low as $O(n^{3.5})$.

Because of the size of the problems we are interested in, we seek a linear solver that is optimal in both computational cost (measured in total number of operations) and storage (measured in the amount of memory needed during the computation). The difference between a method that is $O(N)$ or $O(N \log N)$ and one that is $O(N^{\frac{4}{3}})$ may seem slight, but it is very real. Consider, for example, $N = 10^6$, then $N^{\frac{4}{3}} = 10^8$, and an $O(N^{\frac{4}{3}})$ method requires 100 times the number of operations that an $O(N)$ method does and about 20 times more than an $O(N \log N)$ method. When $N = 10^9$, an $O(N^{\frac{4}{3}})$ method requires 1000 times the number of operations that an $O(N)$ method would, and over 100 times the number of operations as an $O(N \log N)$ method would. As we consider larger and larger problems, this difference becomes more and more critical.

There are a number of different methods that offer nearly optimal efficiency, all based on multiscale analysis. For problems with smooth solutions, Fast Fourier Transform based methods offer $O(N \log N)$ algorithms for accurate solution. For problems with more complex solutions, there are Multigrid [7], Wavelet [9], Domain Decomposition, and Fast Multipole methods. Here, we consider multigrid methods because of their proven effectiveness for finite element discretizations [3]. These methods are based on the complementary use of stationary linear iterations and coarse-grid correction to obtain what is known as *optimal multigrid efficiency.* These methods, however, are not yet applicable to every problem of interest. Indeed, the performance of multigrid methods can be quite problem dependent and quite discretization dependent. Thus, a multigrid algorithm that works well for one problem may not work at all for a similar problem, although careful parameter tuning can result in acceptable performance.

Overcoming this sensitivity is one of our primary research goals. We envision an algorithm that is readily applicable to a large class of problems, without any significant need for parameter tuning. We further hope to develop an algorithm that is optimal in an even more general sense - an algorithm that does less work if presented with a relatively simple problem. The development of such algorithms is a major focus of the proposed thesis.

We are also interested in generating a better understanding of both the individual components of a multigrid algorithm and of their interaction, to help in the development of our new algorithms and to increase our understanding of currently existing algorithms. One potentially revealing tool is the homogenization point of view. Homogenization theory [18] originates in the study of porous media and composite materials, where one knows fine-scale information about material properties and is asked to determine effective coarse-scale properties. The determination of these effective properties is closely related to the determination of the coarse-grid operators in a multigrid setting. Through the study of multigrid methods in the homogenization framework, we expect to gain valuable insight into the interaction between relaxation and coarse-grid correction that is fundamental to a multigrid algorithm.

2 Current Multigrid Algorithms

There are many different algorithms that fall into the class of multigrid methods. These methods range from algorithms designed for a specific problem to those applicable in a much more general sense. In this section, we discuss some of the main features of these methods. With a clear understanding of multigrid principles, the development of our new multigrid algorithm is more easily understood.

The essential multigrid principle is one of complementarity. For sensible discretizations (e.g. Finite Difference or Finite Element) of elliptic PDEs, it can be shown that the Jacobi and Gauss-Seidel iterations exhibit stalling behavior([7], chapter 2). That is, these iterative schemes quickly and effectively reduce error in particular subspaces, but are quite slow at reducing the error in the complementary spaces. Thus the first few iterations of these schemes are quite effective at reducing the overall error (as measured by a global ℓ^2 norm), but further iterations are much less effective. Because of this stalling behavior, it is reasonable to seek a way of reducing the problem we are considering to one which only involves the subspace that relaxation did not effectively treat. For elliptic problems, it can be shown that this subspace is the space of *smooth* vectors ([7], chapter 2).

Geometric multigrid schemes are designed based on discretizations that are geometrically regular. For such discretizations, the error left after relaxation is smooth in a geometric sense - that is, it varies slowly between nearby gridpoints. In this case, it is natural to think about resolving the error using fewer overall degrees of freedom - a coarser grid. In this setting, we see that many of the components of a multigrid algorithm can come naturally from the geometry.

Consider the discretized problem in the form $Ax = b$. After relaxation, we have an approximate solution $\tilde{x}$. Notice that the error, e, in this approximation satisfies the *residual equation*:

$$Ae = A(x - \tilde{x}) = b - A\tilde{x} = r.$$

The error itself lives in the subspace of geometrically smooth functions (vectors), and so we can consider it (and the residual equation) on a coarser grid by simply eliminating points (or dofs) in a judicious fashion. If we choose a reasonable set of points to eliminate, very little information is actually lost by this procedure. If, further, we eliminate these points in a geometrically regular fashion (as depicted in Figure 1), then we could determine an operator on the

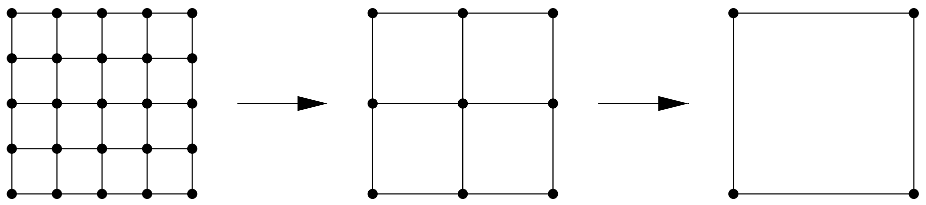

Figure 1: Typical Geometric Coarsening

reduced space simply by rediscretization of the original PDE. Finally, once we have solved the residual equation, we need to interpolate the correction back to the original (fine) grid. Using the fact that we have eliminated points in a regular fashion, we see that interpolation could be done easily by simple linear interpolation between points.

To obtain true efficiency, however, we must solve the coarse-grid system in an efficient manner. Our procedure leads to coarse-grid equations whose character is the same as the fine-grid equations, and so it is reasonable to employ the same solution technique. That is, we solve the coarse-grid equations by recursion. Thus, if we use coarsening as indicated in Figure 1, our algorithm can be depicted pictorially as in Figure 2.

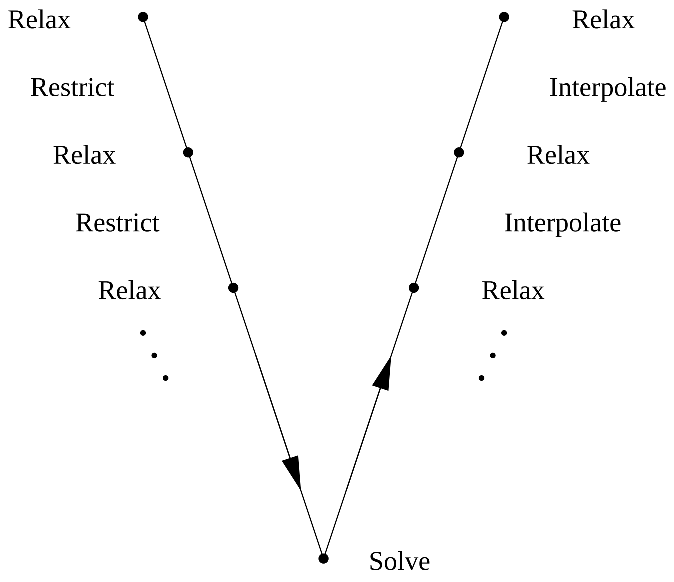

Figure 2: The V-Cycle algorithm

When our original discretization is not geometrically regular (for example, because of accuracy concerns), we must use greater generality in designing our algorithm. Algebraic Multigrid methods (such as Ruge-Stueben AMG [16]) make no assumptions regarding the geometry of the problem, rather they treat it in a purely algebraic fashion. Point-wise relaxation is again used to smooth the error, but now we do not have a geometric intuition into the resulting smoothness. Instead, we use the idea of algebraic smoothness - that after a few sweeps of relaxation $(Ae)_i \approx 0$ at each point i. We then seek to restrict this algebraically smooth error to a coarser space (i.e. a space with fewer degrees of freedom). More importantly, we wish to determine a coarse-grid system whose solution, when interpolated back to the fine-grid, provides a good correction to the algebraically smooth components of the fine-grid solution.

Consider the goal of interpolating algebraically smooth functions. For such a smooth vector, e, we have that

$$(Ae)_i \approx 0$$

and so

$$a_{ii} e_i \approx - \sum_{j \neq i} a_{ij} e_j. \tag{1}$$

Further, suppose that we have chosen a coarse-grid (the process for which we discuss shortly), and so the fine-grid degrees of freedom can be partitioned into two (disjoint) sets, $\{1, 2, \ldots, N\} = C \cup F$ (where C is the set of coarse-grid points and F is all the remaining points (i.e. the points that exist only on the fine-grid)). Since the matrix A is sparse, we also introduce additional notation: $N_i = \{j : a_{ij} \neq 0\}$, $C_i = C \cap N_i$, and $F_i = F \cap N_i$. Then, we can rewrite Equation 1 as

$$a_{ii} e_i \approx - \sum_{j \in C_i} a_{ij} e_j - \sum_{k \in F_i} a_{ik} e_k. \tag{2}$$

Now, to derive an interpolation formula that takes e on the coarse grid to e on the whole (fine) grid, we seek to collapse the connections from point i to points $k \in F_i$ onto the points $j \in C_i$. The Ruge-Stueben algorithm for doing this involves a classification of the points in F_i into strong and weak connections. We do not need this distinction, and thus omit the details, which are described in [16]. The main assumption needed to collapse the stencil is one of representability of the fine-grid portion of a function. That is, functions in the range of interpolation must be representable in terms of their coarse-grid values. Writing

$$e_k \approx \sum_{j \in C_i} w_{kj} e_j + w_{ki} e_i$$

for any $k \in F_i$, this assumption says that for any $k \in F_i$ we can find a set of weights $\{w_{kj}\}$ such that for the error we seek to eliminate through coarse-grid correction a good approximation can be made. In classical AMG algorithms, $\{w_{kj}\}$ are given as functions of $\{a_{ij}\}$ and $\{a_{kj}\}$ and are based on the assumption that the global smooth vectors are locally non-oscillatory [16](i.e. globally smooth vectors look like constant vectors on local neighborhoods).

The selection of the coarse-grid points is of primary importance to the success of an algebraic multigrid method. Fundamentally, the coarse-grid must be one such that algebraically smooth error can be accurately approximated on it and interpolated from it. As well, it must have significantly fewer points than the fine-grid, for it is the reduction in degrees of freedom from one grid to the next that gives multigrid its efficiency. The algorithm used in Ruge-Stueben AMG [16] concerns itself with strong and weak dependence and influence, as these concepts are central both in the definition of interpolation and in a measure of adequacy for the coarse grid.

Once we have chosen a coarse-grid and an interpolation operator, we must still choose a restriction operator to move the residual on a fine-level to the coarse-level and a coarse-grid operator (CGO) on that level. A common technique (and one that we use) relies on what are known as the variational properties. These properties are that we take restriction (commonly denoted R) as the transpose of interpolation (P), and the CGO as the product $A_c = P^TAP$ (called the Galerkin condition). These choices arise naturally from the consideration of Finite Element discretizations posed as minimization problems. The variational properties are a result of choosing R and A_c such that the coarse-grid correction is the correction in the range of interpolation that minimizes a fine-grid functional ([7], chapter 10).

AMG is, and was conceived of as, a generalization of classical, geometric multigrid methods. It is an efficient solver for many problems, including those involving discretizations on stretched or irregular grids, or discretizations of many problems with anisotropic or variable coefficients. There are, however, still many problems for which AMG is not an effective solver [8]. These include problems with highly anisotropic or highly variable coefficients, and problems coming from the discretization of systems of PDEs (such as elasticity). Simply put, the further the algebraically smooth components of a problems are from the single constant vector, the poorer the performance of AMG is.

3 The Self-Correcting (Algebraic) Multigrid Algorithm

Our goal in developing a new type of multigrid method was to move away from the weaknesses of classical AMG schemes. Thus, we first concerned ourselves with generalizing the definition of interpolation in AMG. Our guiding principles for this generalization come from basic properties of all multigrid algorithms:

- Simple relaxation is inefficient for solving $Ax = b$ on error components e such that Ae is small relative to e [4], and
- Efficient multigrid performance is dependent on the appropriate complementarity of relaxation and coarse-grid correction.

In developing the new interpolation procedure, we consider the case of purely algebraic coarsening; however, for practical reasons we chose to first implement the algorithm in the case of regular, geometric coarsening. The numerical results presented in Section 5 are from this implementation in the case of a scalar PDE. We discuss our overall intent for coarsening in Section 7.

Since the success of our methods depends on the complementarity of relaxation and coarse-grid correction, we see that a good starting point for defining interpolation would be to consider a vector, e, that is not well treated by relaxation. Using a simple (point-wise) relaxation scheme, such as Gauss-Seidel, this also means that $Ae \approx 0$, or

$$a_{ii}e_i \approx -\sum_{j\in C_i} a_{ij}e_j - \sum_{k\in F_i} a_{ik}e_k,$$

as in Equation 2, assuming we already have a splitting of N_i into C_i and F_i. Again writing

$$e_k \approx \sum_{j\in C_i} w_{kj}e_j + w_{ki}e_i,$$

we come up with a general interpolation formula for a point $i \in F$,

$$e_i = -\sum_{j\in C_i} \left(\frac{a_{ij} + \sum_{k\in F_i} a_{ik}w_{kj}}{a_{ii} + \sum_{k\in F_i} a_{ik}w_{ki}} \right) e_j. \tag{3}$$

The difference between our interpolation and the interpolation used in classical AMG [16] is that we choose $\{w_{kj}\}$ to depend on both the entries in the matrix A and some algebraically smooth vector, $x^{(1)}$, which we treat as a representative of a number of algebraically smooth components.

One way of looking at the choice of $\{w_{kj}\}$ is to consider the idea of twice removed interpolation. Suppose we have a point i, whose neighbors have been partitioned into the two sets C_i and F_i. The problem of collapsing the fine-fine connections is equivalent to that of determining a way to interpolate to a point $k \in F_i$ from points $j \in C_i$ (or, more generally, $j \in C_i \cup \{i\}$). That is, we seek to write (as before)

$$e_k = \sum_{j\in C_i} w_{kj}e_j. \tag{4}$$

If we have a particular vector, $x^{(1)}$, which we want in the range of interpolation, we ask that Equation 4 hold for $x^{(1)}$. For this interpolation problem, we have many choices for the $\{w_{kj}\}$. We choose to take this interpolation to F_i of the form $-D^{-1}A_{fc}$ (where A_{fc} is the matrix of connections between F and C, and D is an arbitrary diagonal matrix - this choice is motivated by the discussion in [6]). That is, we write

$$d_{kk}x_k^{(1)} = -\sum_{j\in C_i} a_{kj}x_j^{(1)}. \tag{5}$$

And so,

$$d_{kk} = \frac{-\sum\limits_{j \in C_i} a_{kj} x_j^{(1)}}{x_k^{(1)}}.$$

Choosing $w_{kj} = d_{kk}^{-1} a_{kj}$, we get the interpolation formula

$$\begin{aligned} e_k &= -\sum_{j \in C_i} \frac{a_{kj}}{d_{kk}} e_j & (6) \\ &= \sum_{j \in C_i} \frac{a_{kj} x_k^{(1)}}{\sum\limits_{j' \in C_i} a_{kj'} x_{j'}^{(1)}} e_j & (7) \\ &= \sum_{j \in C_i} w_{kj} e_j & (8) \end{aligned}$$

Interpolation to $i \in F$ is then given by Equation 3 and has the particular form

$$e_i = -\sum_{j \in C_i} \left(\frac{a_{ij} + \sum\limits_{k \in F_i} a_{ik} \frac{a_{kj} x_k^{(1)}}{\sum\limits_{j' \in C_i} a_{kj'} x_{j'}^{(1)}}}{a_{ii}} \right) e_j. \qquad (9)$$

Treating the interpolation operator, P as an operator from C to $F \cup C$, we see that it has the form

$$P = \begin{bmatrix} W \\ I \end{bmatrix}$$

where W is the matrix of coefficients as in Equations 3 and 9.

Another way of looking at the twice-removed interpolation is to consider how it relates to the interpolation used in its place in Ruge-Stueben AMG. For strongly connected points, AMG uses interpolation of the form

$$w_{kj} = \frac{a_{kj}}{\sum\limits_{m \in C_i} a_{km}}$$

One of the primary assumptions of AMG is that the global smoothest error component is the constant vector [16], and thus it must be in the range of interpolation. In our terms, AMG assumes that $x^{(1)} \equiv 1$. Looking at our interpolation formula, we see that it reduces to the classical AMG strong interpolation formula in this case. So, we can look at the scAMG interpolation formula as a simple generalization of the classical AMG formula for the case where the smoothest error component (and thus one that needs to be in the range of interpolation) is any known vector.

The above ideas for interpolation can also be generalized to systems of PDEs. Consider discretizing a system so that the degrees of freedom in the system are located on the same grid, i.e. there are d degrees of freedom co-located at each node. Since we seek to generalize the ideas from the scalar case, we will start by generalizing the notation: A_{kj} becomes the matrix of connections between the dofs located at nodes k and those located at node j, D_{kk} becomes a matrix instead of a scalar, $x^{(1)}$ remains a single vector ("bad guy"), but we also make use of $x^{(2)}, \ldots, x^{(d)}$, and use the matrix $X^{(1)} = [x^{(1)}, \ldots, x^{(d)}]$ and its restriction to the d dofs at node k, $X_k^{(1)}$.

The analogue of Equation 5 is then

$$D_{kk} X_k^{(1)} = -\sum_{j \in C_i} A_{kj} X_j^{(1)},$$

which gives us

$$D_{kk} = -\left(\sum_{j \in C_i} A_{kj} X_j^{(1)} \right) \left(X_k^{(1)} \right)^{-1}.$$

The corresponding equations to Equations 6-8 are then

$$\begin{aligned} e_k &= -\sum_{j \in C_i} D_{kk}^{-1} A_{kj} e_j \\ &= \sum_{j \in C_i} \left(X_k^{(1)} \right) \left(\sum_{j' \in C_i} A_{kj'} X_{j'}^{(1)} \right)^{-1} A_{kj} e_j \\ &= \sum_{j \in C_i} W_{kj} e_j. \end{aligned}$$

We can then derive the new nodal interpolation operator for $i \in F$ as

$$A_{ii} e_i = -\sum_{j \in C_i} \left(A_{ij} + \sum_{k \in F_i} A_{ik} W_{kj} \right) e_j,$$

or, as

$$e_i = -A_{ii}^{-1} \left(\sum_{j \in C_i} \left(A_{ij} + \sum_{k \in F_i} A_{ik} W_{kj} \right) \right) e_j.$$

Successful implementation of these schemes for interpolation thus rely upon having an appropriate vector, $x^{(1)}$, (or vectors $X^{(1)}$) to put in the range of interpolation. Since we need the complementarity of relaxation and coarse-grid correction, it follows that the best choice for $x^{(1)}$ would be a representative of the vectors for which relaxation is inefficient. Thus, a straight-forward method for generating this vector would be to start with a vector (or vectors) equally rich in all components (i.e. eigenvectors of symmetric A), relax on $Ax = b$

for some b, and then determine the error in the approximate solution after a sufficient number of relaxations.

To generate a vector rich in all components is, of course, difficult to do explicitly as it requires knowing what these components are. Instead, we choose to start with a vector generated at random (elements are taken from a uniform distribution on $[0, 1)$). While a vector generated like this is, in general, not equally rich in all components, in practice it is sufficiently rich such that after relaxation we can get a good representative of the slow to converge components.

Getting the representative vector is then easy if we choose the right side for our relaxation properly. We are interested in the error after a number of relaxations on a problem $Ax = b$. Clearly this error is easiest to calculate if we choose b in a manner such that we know the true solution. This is easily done by choosing a solution x and then calculating the appropriate right side, b, but we can make this problem even easier by choosing $x \equiv 0$, so that $b \equiv 0$. Then, starting with any initial guess and relaxing on $Ax = 0$, we can identify the error in the solution after relaxation as simply being the current approximation, since the true solution is 0. So, starting with a random initial guess and performing relaxation on $Ax = 0$ generates a vector $x^{(1)}$ representative of the slow to converge components that we can then use in the interpolation formula.

In practice, however, it requires (in our opinion) too many relaxation sweeps to generate a suitable representative. Thus, to improve performance, we have implemented a multilevel scheme to quickly generate a vector $x^{(1)}$ such that $Ax^{(1)} \approx 0$, and so point-wise relaxation is slow to resolve this vector. To do this, we start with a random guess on the fine grid and perform a few relaxation sweeps to generate a tentative $x^{(1)}$. From this, we generate an interpolation operator (as above) and form the CGO using the Galerkin condition. We use injection (direct restriction of the values on the C-points) to form a coarse-grid initial guess and recurse to the coarsest level. From this coarsest level, we interpolate the vector all the way to the finest level and repeat this process using that vector as an overall initial guess. As we discuss in Section 5, this procedure has led to grid independent convergence factors all the way to 1024×1024 grids for many scalar problems.

One important benefit of generating the initial representative vector in a multilevel fashion is the ability to implement a graceful degradation to simplicity in the algorithm. That is, since we begin by relaxing on a random vector (assumed to be rich in all components), we can easily tell if relaxation is sufficient to solve either the fine grid problem or one of the generated coarse grid problems. If this is indeed the case, then we need not invest any additional labor in designing an algorithm as we already have an efficient solver.

For systems, an added wrinkle is the need to generate multiple representative vectors. We expect that a technique similar to the one described above can generate the components to a sufficient degree of accuracy to result in an acceptable scheme, but need to further investigate the initial guesses. One possible strategy is to simply choose the initial guesses at random, over all degrees of freedom, and expect that the resulting vectors do span a subspace of algebraically smooth components. Another strategy would be to take initial guesses at random on each degree of freedom, but to be zero on all others (i.e. each dof on a node is chosen to be nonzero for only one of the starting vectors). This strategy seems more likely to lead to a rich subspace of algebraically smooth components, but we must ensure that we are not harming ourselves by making this choice. The extensions to systems is discussed again in Section 7.

4 Theoretical Properties

We have already seen one advantageous property of our algorithm, namely that it reduces to a form of the classical Ruge-Stueben AMG algorithm in the case when $x^{(1)} = 1$. We claim this property to be advantageous in that we can reasonably expect our algorithm to perform well on the class of problems that AMG performs well on (dependent, of course, on sufficient resolution of $x^{(1)}$). However, we believe that our choice of interpolation for $x^{(1)} \neq 1$ leads to an algorithm that is much more successful.

One situation that causes difficulty for AMG is when the matrix for a problem it is effective on is simply rescaled. In particular, consider taking the matrix A and multiplying it by a diagonal scaling matrix on both the left and the right sides (to preserve symmetry). That is, replace A with $\tilde{A} = DAD$ for some diagonal matrix D. If $Ax^{(1)} = 0$, then $\tilde{A}(D^{-1}x^{(1)}) = 0$, so the new near null-space component is actually $D^{-1}x^{(1)}$. If the diagonal entries of D have significant variation in them, then $D^{-1}x^{(1)}$ has a significantly different character than $x^{(1)}$. In the case of classical AMG, this can cause a significant deterioration in convergence rates.

Our algorithm, however, is not as sensitive to this distinction. In fact, under minimal assumptions, our algorithm can be shown to be unaffected by such scaling. Let A and D have the forms

$$A = \begin{bmatrix} A_{ff} & A_{fc} \\ A_{cf} & A_{cc} \end{bmatrix} \text{ and } D = \begin{bmatrix} D_f & 0 \\ 0 & D_c \end{bmatrix},$$

so that

$$\tilde{A} = \begin{bmatrix} D_f A_{ff} D_f & D_f A_{fc} D_c \\ D_c A_{cf} D_f & D_c A_{cc} D_c \end{bmatrix}.$$

Suppose that if the process for generating the representative vector generates $x^{(1)}$ when given the matrix A, then it generates $\tilde{x}^{(1)} = D^{-1}x^{(1)}$ when given the matrix $\tilde{A}$. Our choice for interpolation for the matrix $\tilde{A}$ is then given by taking

$$\begin{aligned} \tilde{w}_{kj} &= \frac{\tilde{a}_{kj}\tilde{x}_k^{(1)}}{\sum\limits_{j' \in C_i} \tilde{a}_{kj'}\tilde{x}_{j'}^{(1)}} \\ &= \frac{d_k a_{kj} d_j d_k^{-1} x_k^{(1)}}{\sum\limits_{j' \in C_i} d_k a_{kj'} d_{j'} d_{j'}^{-1} x_{j'}^{(1)}} \\ &= d_k^{-1} w_{kj} d_j, \end{aligned}$$

which gives us

$$\begin{aligned} e_i &= -\sum_{j\in C_i} \left(\frac{\tilde{a}_{ij} + \sum\limits_{k\in F_i} \tilde{a}_{ik}\tilde{w}_{kj}}{\tilde{a}_{ii}} \right) e_j \\ &= -\sum_{j\in C_i} \left(\frac{d_i a_{ij} d_j + \sum\limits_{k\in F_i} d_i a_{ik} d_k d_k^{-1} w_{kj} d_j}{d_i a_{ii} d_i} \right) e_j \\ &= -\sum_{j\in C_i} d_i^{-1} \left(\frac{a_{ij} + \sum\limits_{k\in F_i} a_{ik} w_{kj}}{a_{ii}} \right) d_j e_j \end{aligned}$$

for $i \in F$. For $i \in C$, we simply take the value from the coarse-grid and assign it as the value on the fine-grid. Thus, we get an interpolation operator $\tilde{P}$ of the form

$$\tilde{P} = \begin{bmatrix} \tilde{W} \\ I \end{bmatrix} = \begin{bmatrix} D_f^{-1} W D_c \\ I \end{bmatrix} = D^{-1} P D_c,$$

where P is the interpolation operator from the non-scaled case. Further, if we consider the coarse-grid operators A_c and $\tilde{A}_c$, we see that

$$\tilde{A}_c = \tilde{P}^T \tilde{A} \tilde{P} = (D_c P^T D^{-1})(DAD)(D^{-1} P D_c) = D_c P^T A P D_c = D_c A_c D_c.$$

That is, the coarse-grid operator for the scaled problem is simply the scaled version of the coarse-grid operator for the unscaled problem. Noticing that standard relaxation techniques such as Gauss-Seidel or Jacobi (both point-wise and block relaxations) are scaling invariant (that is, if we scale the matrix A to DAD as above, scale the initial guess $x^{(0)}$ to $D^{(-1)}x^{(0)}$ and the initial right side b to Db, then the approximation generated changes from $x^{(1)}$ to $D^{-1}x^{(1)}$), we see that the entire process is independent of any diagonal scaling. That is, we expect to get similar convergence factors (measured in the energy norm) for the diagonally scaled problem as we get for the unscaled problem.

The one assumption this result uses is that for the scaled problem we can generate the scaled vector $D^{-1}x^{(1)}$ just as easily as the unscaled vector $x^{(1)}$. In practice this is more difficult. If we do happen to know the scaling matrix D a priori, then it is true that by scaling the initial guess we can ensure equivalence between the two vectors. This is, of course, an unrealistic situation - if we knew the scaling beforehand, we could simply unscale the problem. Starting with the same initial guess for both the scaled and unscaled problems tends to give slightly worse convergence rates for the unscaled problem (as is discussed in Section 5), but not a significant performance hit.

The same analysis can also be performed for the extension to systems mentioned in Section 3. Replacing diagonal scaling by nodal scaling and point-wise relaxation by node-wise relaxation, we get the same invariance to the scaling. This is particularly of interest for industrial problems, where matrices are often rescaled node-wise, destroying the physical interpretation of the degrees of freedom in a problem.

5 Numerical Results

To examine the feasability of our approach, we have implemented our solver for the special case of a rectangular grid in 2-dimensions with full coarsening (as pictured in Figure 1). This restriction in generality has a notable effect on the range of problems that we are able to reasonably consider (e.g. anisotropy becomes much more difficult to account for in this setting). However, if we examine problems outside of this range, we feel that we can get a good indication of the performance of our ideas. The results presented here are preliminary and are not intended to answer the overall question of determining if the added cost results in proportionally added accuracy for these problems.

To test the algorithms, we have chosen a small test set, but one that we feel is representative of a reasonable class of problems. All of the PDEs examined were discretized using bilinear finite elements on a tensor-product grid over the canonical unit square. Boundary conditions were implemented in such a way that the resulting matrices are symmetric and positive-definite or positive-semi-definite (in which case we project out the exact null space component on the finest grid when measuring convergence). We provide convergence factor data, that is the factor by which the energy norm (given a symmetric positive-definite matrix A, the energy norm (or A-norm) of x is $\sqrt{\langle Ax, x\rangle}$) of the error is reduced, tested on problems with zero right sides.

For this stage of testing, we have chosen four problems for which we expect differing degrees of success. Our first two problems are Laplace's equation, with Dirichlet boundary conditions (Problem 1) and Neumann boundary conditions (Problem 2). Our second two problems come from porous media flow problems (which we discuss in Section 6), and come from the PDE

$$-\nabla \cdot D(x,y)\nabla p(x,y) = 0$$

for discontinuous $D(x,y)$. We choose the boundary conditions for both problems to be Dirichlet along the East and West boundaries and Neumann along the North and South boundaries. For Problem 3, we take

$$D(x,y) = \begin{cases} 10^2 & (x,y) \in [\frac{1}{3}, \frac{2}{3}]^2 \\ 1 & \text{otherwise} \end{cases},$$

and for Problem 4, we take

$$D(x,y) = \begin{cases} 10^5 & y \leq \frac{1}{2} \\ 1 & \text{otherwise} \end{cases}.$$

Our first series of tests are for the matrices straight from these discretizations (i.e. unscaled) on a series of grids, ranging from 32×32 to 1024×1024. These results are summarized in Table 1. For Problems 1,2, and 4, we see excellent results - the convergence factors are bounded above by 0.07 regardless of mesh-size. The results for Problem 3, however, appear somewhat disappointing. The main feature in Problem 3 is the interior square, which is not grid-aligned. Our studies thus far indicate that results are much-improved if the feature is grid-aligned, and the cause of the discrepancy is a question of interest to us.

Problem Size	Problem 1	Problem 2	Problem 3	Problem 4
32×32	0.058	0.064	0.078	0.060
64×64	0.065	0.067	0.094	0.067
128×128	0.068	0.070	0.081	0.069
256×256	0.070	0.070	0.106	0.070
512×512	0.070	0.070	0.169	0.070
1024×1024	0.070	0.070	0.384	0.070

Table 1: Convergence Factors for the Unscaled Matrices

As discussed in Section 4, one of the advantages that our definition of interpolation has over classical AMG is the invariance to diagonal scaling. We have, of course, many options for the scaling of the matrix. One which we chose to examine was choosing it based on the (x, y) position of the gridpoint in question, and we took as a scaling function

$$1 + \sin(547\pi x)\sin(496\pi y) + 10^{-7}.$$

This function was chosen as for the values of h we are considering (ranging from $\frac{1}{32}$ to $\frac{1}{1024}$, it looks like a random scaling on each node in the range $[10^{-7}, 2 + 10^{-7}]$. If we scale the matrix in this way, and scale our initial guesses in the same manner, we get convergence factors that are unchanged from the unscaled matrices, and are shown in Table 2.

Problem Size	Problem 1	Problem 2	Problem 3	Problem 4
32×32	0.058	0.064	0.078	0.060
64×64	0.065	0.067	0.094	0.067
128×128	0.068	0.070	0.081	0.069
256×256	0.070	0.070	0.106	0.070
512×512	0.070	0.070	0.169	0.070
1024×1024	0.070	0.070	0.384	0.070

Table 2: Convergence Factors for the Scaled Matrices with Scaled Initial Guesses

The scaling of the initial guess is, of course, only of interest to validate the earlier theoretical results. In practice, we would not have the scaling factor readily available, and so a more realistic comparison is obtained by considering the performance of the algorithm on the scaled matrices without scaling the initial guess. These results are presented in Table 3. Of particular interest is that the convergence rates are nearly unchanged for the three problems that previously performed well. For Problem 3, however, our difficulties continue.

Problem Size	Problem 1	Problem 2	Problem 3	Problem 4
32×32	0.058	0.064	0.078	0.060
64×64	0.066	0.067	0.091	0.067
128×128	0.068	0.070	0.087	0.069
256×256	0.070	0.070	0.348	0.070
512×512	0.070	0.070	0.190	0.070
1024×1024	0.070	0.070	0.918	0.070

Table 3: Convergence Factors for the Scaled Matrices with Unscaled Initial Guesses

To ensure a complete comparison, we also investigated the use of scaling of the form $\sqrt{r_1 10^{5r_2}}$ where r_1, r_2 were chosen from a uniform distribution on $[0, 1]$. Results for this scaling are presented in Table 4. Most notably, we see some degradation in the results for large meshes. This suggests that we are not doing a good enough job generating interpolation operators for these meshes. We believe the primary responsibility for this is in our procedure for generating a representatively smooth vector. If we increase the number of relaxations done on each level or the number of cycles we perform to form the initial representative vector (and so generate a smoother vector), we can reduce these convergence factors to the previous levels. In Table 5, we use a third cycle to generate the representative vector and do, indeed, recover better convergence factors. A similar phenomenon may be at work in Problem 3, where we interpolate across the feature boundary on the finest grid. If we do not properly resolve this boundary in the representative vector, we cannot hope to accurately account for it in our interpolation operator. These results indicate that further study of the determination of the representative vector are necessary for us to obtain the full efficiency we seek.

Problem Size	Problem 1	Problem 2	Problem 3	Problem 4
32×32	0.057	0.065	0.076	0.060
64×64	0.066	0.067	0.095	0.066
128×128	0.068	0.069	0.134	0.068
256×256	0.070	0.070	0.303	0.070
512×512	0.070	0.132	0.908	0.108
1024×1024	0.119	0.182	0.783	0.159

Table 4: Convergence Factors for the Randomly Scaled Matrices with Unscaled Initial Guesses

Problem Size	Problem 1	Problem 2	Problem 3	Problem 4
32×32	0.058	0.065	0.079	0.060
64×64	0.065	0.067	0.095	0.066
128×128	0.068	0.069	0.083	0.068
256×256	0.070	0.070	0.106	0.070
512×512	0.070	0.070	0.676	0.070
1024×1024	0.070	0.070	0.125	0.070

Table 5: Convergence Factors for the Randomly Scaled Matrices with Unscaled Initial Guesses Based on the Improved Cycling

6 The Homogenization Viewpoint

Numerical homogenization (or upscaling) is a process for reducing the number of degrees of freedom in a computational simulation. One common application of these techniques is in porous media flow [18], where the governing equations are Darcy's Law and conservation of mass:

$$-D(x)\nabla p(x) = u(x) \tag{10}$$
$$\nabla \cdot u(x) = f(x), \tag{11}$$

where $p(x)$ represents the pressure, $D(x)$ is a spatially varying permeability coefficient, possibly tensor-valued (although it must be symmetric and positive-definite at every point), $u(x)$ is the Darcy velocity, and $f(x)$ represents mass sources and sinks in the domain of interest. Thus, Equation 10 says the Darcy velocity is proportional to the pressure gradient while Equation 11 states that changes in the overall quantity of fluid are due to external sources or sinks.

The need for a reduction in the number of degrees of freedom comes from the significant disparity between the scale of variation of the permeability and the size of the domain. Typical problems have material properties (such as permeability) that vary on the scale of millimeters, with an overall domain several kilometers across. Thus, a fully resolved three-dimensional simulation would call for a number of degrees of freedom on the order of 10^{18} - much more than even the most powerful modern supercomputer can support. Numerical homogenization techniques are used to seek a coarser discretization that preserves the effects of the fine-scale variation in the fully resolved problem.

For arbitrary permeability tensors, it is insufficient to generate the coarse-scale material properties by taking simple averages. While it can be shown that the homogenized permeability in a given direction is bounded below by the harmonic average and above by the arithmetic average [13], these averages alone are insufficient to accurately capture the full effects of the fine-scale variation. In particular, these bounds may approach zero or infinity in some limiting cases [1] and thus are not practical for defining an equivalent medium. For two-scale periodic media, a two-scale asymptotic analysis is possible to generate the homogenized problem [12], but we typically expect the permeability field to vary on more than 2 scales, indeed, it may vary on infinitely many (fractal) scales. Rather than use these analytical techniques artificially (that is, outside their range of validity), we seek to determine the homogenized permeabilities numerically.

Many numerical methods have been devised to compute the effective permeabilities. *Additive* methods are based on direct local averaging of the permeability [18]. One significant disadvantage of these methods is that the computed effective permeabilities are naturally diagonal tensors, and thus it is difficult to capture any non-grid aligned global permeability without knowing of it beforehand. *Laplacian* methods involve solving localized PDEs over each region for which the effective permeability is to be calculated and then using the solutions to infer this permeability. A typical Laplacian method [11] considers the solution of the PDE restricted to the region for which the effective permeability is to be calculated under a series of boundary conditions chosen to induce given (constant) pressure gradients. From the solutions of these PDEs, the average flow can be calculated and by comparison with the pressure gradient, the components of the effective permeability tensor can be recovered. These methods are based on the two-scale asymptotic analysis mentioned above, and thus are strictly valid only for two-scale periodic media.

In seeking a more efficient algorithm, one can consider how multigrid methods create their own versions of the coarsened problem in the multigrid hierarchy. A truly effective (robust) multilevel solver must somehow capture the effects of the multiple-scale features present in the fine-scale discretization of a PDE. Multigrid Homogenization is based on the idea that if the multigrid coarsening strategy is based on physically relevant criteria, then the coarse-grid operators should preserve the effects of the fine-scale physics. Recovering the effective material properties from these coarse-scale problems should then give good approximations to the true effective material properties.

An example of a multigrid code based on physically relevant criteria is Dendy's Black Box Multigrid (BoxMG) [10]. As shown in [15], the operator-induced interpolation used in BoxMG can be derived by approximately enforcing the continuity of normal flux across fine-grid boundaries of a bilinear finite element discretization. Based on this physical motivation, we hope to be able to accurately recover a good approximation to the coarse-scale material properties from the multigrid coarse-grid operators. Indeed, in [15] this was shown to be the case both analytically and numerically for periodic problems under certain assumptions on the variation of the coefficients in the coarse-grid operator. It has also been shown [14] that if the coarse-grid operator is constant over the grid then the coarse-grid operator is equivalent to the bilinear finite element discretization of

$$-\nabla \cdot \hat{D}\nabla p + \partial_{xy}\hat{E}\partial_{xy}p = \hat{f} \tag{12}$$

for some $\hat{E}$, constant on a given grid.

We have considered the same problem subject to Neumann boundary conditions and obtained similar results. With Neumann BCs, similar terms again appear and provide a necessary regularization to the coarse-grid problem. In 2D, the regularization term takes the same form as for the constant-operator case. In 3D, there are six fourth-order terms and one sixth-order term that appear to perform the same task (although we have not numerically investigated the 3D problem as yet). By accounting for the fourth-order regularization term, we have been able to numerically recover the effective material properties in some example problems.

Discretizing Equations 10 and 11 on a given rectangular grid subject to Neumann boundary conditions results in a symmetric, zero row-sum operator. Galerkin coarsening preserves these properties, and so all coarse-grid problems will also be symmetric and have a zero row-sum. If we coarsen in a regular fashion, such as by full- or semi-coarsening, the coarse-grid operator is also an operator on a rectangular grid. If we seek to interpret the coarse-grid operators as bilinear FE discretizations on those grids, then the operator must be a linear combination of the symmetric, zero row-sum bilinear FE operators. In 2D, this means that the coarse grid operators are linear combinations of the $\partial_{xx}, \partial_{xy}, \partial_{yy}$, and ∂_{xxyy} bilinear FE operators, and that the coarse-grid operator can be treated as a discretization of Equation 12 with Neumann boundary conditions. Following Moulton et. al. [15] and Knapek [14], we assume that the effective coarse-grid material properties are constant on each element, and denote them on the element centered at $(x_{i+\frac{1}{2}}, y_{j+\frac{1}{2}})$ by $\hat{D}^{i+\frac{1}{2},j+\frac{1}{2}}$ and $\hat{E}^{i+\frac{1}{2},j+\frac{1}{2}}$, where

$$\hat{D}^{i+\frac{1}{2},j+\frac{1}{2}} = \begin{bmatrix} \hat{D}_{11}^{i+\frac{1}{2},j+\frac{1}{2}} & \hat{D}_{12}^{i+\frac{1}{2},j+\frac{1}{2}} \\ \hat{D}_{12}^{i+\frac{1}{2},j+\frac{1}{2}} & \hat{D}_{22}^{i+\frac{1}{2},j+\frac{1}{2}} \end{bmatrix}.$$

At a node (i,j), we will denote the operator in stencil form

$$\begin{bmatrix} -S_{i,j}^{NW} & -S_{i,j}^{N} & -S_{i,j}^{NE} \\ -S_{i,j}^{W} & S_{i,j}^{O} & -S_{i,j}^{E} \\ -S_{i,j}^{SW} & -S_{i,j}^{S} & -S_{i,j}^{SE} \end{bmatrix}.$$

With this setup, we can easily recover the off-diagonal component of $\hat{D}$, by noticing that

$$S_{i+1,j}^{NW} = \frac{1}{6}\frac{h_y}{h_x}\hat{D}_{11}^{i+\frac{1}{2},j+\frac{1}{2}} + \frac{1}{6}\frac{h_x}{h_y}\hat{D}_{22}^{i+\frac{1}{2},j+\frac{1}{2}} - \frac{1}{2}\hat{D}_{12}^{i+\frac{1}{2},j+\frac{1}{2}} - \frac{1}{h_x h_y}\hat{E}^{i+\frac{1}{2},j+\frac{1}{2}}$$

$$S_{i,j}^{NE} = \frac{1}{6}\frac{h_y}{h_x}\hat{D}_{11}^{i+\frac{1}{2},j+\frac{1}{2}} + \frac{1}{6}\frac{h_x}{h_y}\hat{D}_{22}^{i+\frac{1}{2},j+\frac{1}{2}} + \frac{1}{2}\hat{D}_{12}^{i+\frac{1}{2},j+\frac{1}{2}} - \frac{1}{h_x h_y}\hat{E}^{i+\frac{1}{2},j+\frac{1}{2}}$$

and so $\hat{D}_{12}^{i+\frac{1}{2},j+\frac{1}{2}} = S_{i,j}^{NE} - S_{i+1,j}^{NW}$.

To recover the diagonal components of $\hat{D}$, we begin by determining a relationship between them valid on any given element. Adding $S_{i+1,j}^{NW}$ and $S_{i,j}^{NE}$ gives us

$$S_{i,j}^{NE} + S_{i+1,j}^{NW} = \frac{1}{3}\left(\frac{h_y}{h_x}\hat{D}_{11}^{i+\frac{1}{2},j+\frac{1}{2}} + \frac{h_x}{h_y}\hat{D}_{22}^{i+\frac{1}{2},j+\frac{1}{2}}\right) - \frac{2}{h_x h_y}\hat{E}^{i+\frac{1}{2},j+\frac{1}{2}},$$

and so we have

$$\frac{1}{6}\frac{h_x}{h_y}\hat{D}_{22}^{i+\frac{1}{2},j+\frac{1}{2}} - \frac{1}{h_x h_y}\hat{E}^{i+\frac{1}{2},j+\frac{1}{2}} = \frac{1}{2}\left(S_{i,j}^{NE} + S_{i+1,j}^{NW}\right) - \frac{1}{6}\frac{h_y}{h_x}\hat{D}_{11}^{i+\frac{1}{2},j+\frac{1}{2}}$$

and

$$\frac{1}{6}\frac{h_y}{h_x}\hat{D}_{11}^{i+\frac{1}{2},j+\frac{1}{2}} - \frac{1}{h_x h_y}\hat{E}^{i+\frac{1}{2},j+\frac{1}{2}} = \frac{1}{2}\left(S_{i,j}^{NE} + S_{i+1,j}^{NW}\right) - \frac{1}{6}\frac{h_x}{h_y}\hat{D}_{22}^{i+\frac{1}{2},j+\frac{1}{2}}.$$

Now, we have that

$$\begin{aligned} S_{i,j}^{E} = \frac{1}{3}\frac{h_y}{h_x}\left(\hat{D}_{11}^{i+\frac{1}{2},j+\frac{1}{2}} + \hat{D}_{11}^{i+\frac{1}{2},j-\frac{1}{2}}\right) &- \frac{1}{6}\frac{h_x}{h_y}\left(\hat{D}_{22}^{i+\frac{1}{2},j+\frac{1}{2}} + \hat{D}_{22}^{i+\frac{1}{2},j-\frac{1}{2}}\right) \\ &+ \frac{1}{h_x h_y}\left(\hat{E}^{i+\frac{1}{2},j+\frac{1}{2}} + \hat{E}^{i+\frac{1}{2},j-\frac{1}{2}}\right) \end{aligned}$$

and so

$$\begin{aligned} S_{i,j}^{E} = \frac{1}{3}\frac{h_y}{h_x}\left(\hat{D}_{11}^{i+\frac{1}{2},j+\frac{1}{2}} + \hat{D}_{11}^{i+\frac{1}{2},j-\frac{1}{2}}\right) &- \frac{1}{2}\left(S_{i,j}^{NE} + S_{i+1,j}^{NW}\right) + \frac{1}{6}\frac{h_y}{h_x}\hat{D}_{11}^{i+\frac{1}{2},j+\frac{1}{2}} \\ &- \frac{1}{2}\left(S_{i,j-1}^{NE} + S_{i+1,j-1}^{NW}\right) + \frac{1}{6}\frac{h_y}{h_x}\hat{D}_{11}^{i+\frac{1}{2},j+\frac{1}{2}}. \end{aligned}$$

Gathering terms, we get

$$\hat{D}_{11}^{i+\frac{1}{2},j+\frac{1}{2}} + \hat{D}_{11}^{i+\frac{1}{2},j-\frac{1}{2}} = 2\frac{h_x}{h_y}\left(S_{i,j}^{E} + \frac{1}{2}\left(S_{i,j}^{NE} + S_{i+1,j}^{NW}\right) + \frac{1}{2}\left(S_{i,j-1}^{NE} + S_{i+1,j-1}^{NW}\right)\right).$$

Now, for given (i,j), this gives us relations between the value of $\hat{D}_{11}$ on the element centered at $(x_{i+\frac{1}{2}}, y_{j+\frac{1}{2}})$ and its northern and southern neighbors. For this to be most useful for us, we now need expressions for $\hat{D}_{11}$ on the southern-most or nothern-most elements. Since the matrix is zero row-sum, we conclude that the effective boundary conditions are the natural boundary conditions, and we get

$$\begin{aligned} \hat{D}_{11}^{i+\frac{1}{2},\frac{1}{2}} &= 2\frac{h_x}{h_y}\left(S_{i,0}^{E} + \frac{1}{2}\left(S_{i,0}^{NE} + S_{i+1,0}^{NW}\right)\right) \\ \text{and } \hat{D}_{11}^{i+\frac{1}{2},n_y-\frac{1}{2}} &= 2\frac{h_x}{h_y}\left(S_{i,n_y}^{E} + \frac{1}{2}\left(S_{i,n_y-1}^{NE} + S_{i+1,n_y-1}^{NW}\right)\right). \end{aligned}$$

A similar procedure can be undertaken for the $(2,2)$ component of $\hat{D}$. Omitting the computational details, which are quite similar to the $(1,1)$ component case, we get

$$\hat{D}_{22}^{i-\frac{1}{2},j+\frac{1}{2}} + \hat{D}_{22}^{i+\frac{1}{2},j+\frac{1}{2}} = 2\frac{h_y}{h_x}\left(S_{i,j}^{N} + \frac{1}{2}\left(S_{i,j}^{NE} + S_{i+1,j}^{NW}\right) + \frac{1}{2}\left(S_{i-1,j}^{NE} + S_{i,j}^{NW}\right)\right),$$

with conditions

$$\begin{aligned} \hat{D}_{22}^{\frac{1}{2},j+\frac{1}{2}} &= 2\frac{h_y}{h_x}\left(S_{0,j}^{N} + \frac{1}{2}\left(S_{0,j}^{NE} + S_{1,j}^{NW}\right)\right) \\ \text{and } \hat{D}_{22}^{n_x-\frac{1}{2},j+\frac{1}{2}} &= 2\frac{h_y}{h_x}\left(S_{n_x,j}^{N} + \frac{1}{2}\left(S_{n_x-1,j}^{NE} + S_{n_x,j}^{NW}\right)\right). \end{aligned}$$

With all the values in $\hat{D}^{i+\frac{1}{2},j+\frac{1}{2}}$ recovered, we can easily recover $\hat{E}^{i+\frac{1}{2},j+\frac{1}{2}}$ from, for example, the equation for $S_{i,j}^{NE}$.

We have implemented this algorithm using Dendy's BoxMG to do the multigrid coarsening, and have investigated its performance on a number of standard homogenization test problems. Given a permeability distribution on $[0,1]^2$, we create a $k \times k$ tiling of it by scaling $[0,1]^2$ to $[0,\frac{1}{3}]^2$ and extending periodically). For these problems, we begin with a fine-grid problem on a grid of $k2^p \times k2^p$ elements and coarsen to a grid of $k \times k$ elements so that each coarse-grid element corresponds to a full coarsening of the given permeability. We have chosen $k = 3$, to ensure that we have at least one coarse-grid cell that is isolated from any potential boundary effects. We have chosen as examples, a subset of those examined in [15] and use those results (for periodic boundary conditions) to validate our algorithm.

Our first example is based on the square inhomogeneity depicted in Figure 3. In this case, the permeability coefficient on the fine-grid is scalar-valued, and given by

$$D(x,y) = \begin{cases} \lambda & (x,y) \in [\frac{1}{3},\frac{2}{3}]^2 \\ 1 & \text{otherwise} \end{cases} .$$

We report our results for the values of $\hat{D}$ and $\hat{E}$ in the center cell of the coarse-grid for $\lambda = 10$, as well as the results of Moulton et. al. [15] (denoted by $\hat{D}^M$ in Table 6). For this example, $\hat{D}$ was a scalar to machine precision, and so we only report the single component. We see that our results for this problem are the same as those of [15]. For the periodic problem, Bourgat [1] computes the homogenized permeability to be comparable with our results on the fine grids.

The L-shaped inhomogeneity pictured in Figure 4 with fine-grid permeability

$$D(x,y) = \begin{cases} \lambda & (x,y) \in \Omega_1 \\ 1 & \text{otherwise} \end{cases}$$

is interesting because despite its scalar fine-grid permeability, the homogenized permeability is tensor-valued. Bourgat computes the homogenized permeability

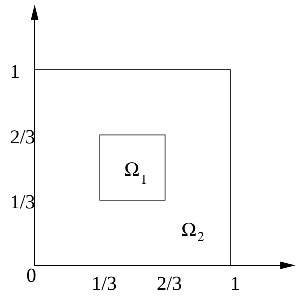

Figure 3: Square Inhomogeneity

Problem Size	$\hat{D}$	$\hat{E}$	$\hat{D}^M$
12×12	1.5979	-0.0708	1.5979
24×24	1.1243	-0.0190	1.1243
48×48	1.2897	-0.0437	1.2897
96×96	1.1934	-0.0294	1.1934
192×192	1.2372	-0.0360	1.2372
384×384	1.2143	-0.0325	1.2143
768×768	1.2254	-0.0342	1.2254

Table 6: Recovered coefficients for Square Inhomogeneity

to be

$$\begin{bmatrix} 1.915 & -0.101 \\ -0.101 & 1.915 \end{bmatrix} = Q \begin{bmatrix} 2.016 & 0 \\ 0 & 1.814 \end{bmatrix} Q^T,$$

where Q is the orthonormal matrix

$$\frac{1}{\sqrt{2}} \begin{bmatrix} -1 & 1 \\ 1 & 1 \end{bmatrix}.$$

Q actually defines the principal axes of the diffusion being considered, and in this case corresponds to a rotation by 45°. Results for this problem for $\lambda = 10$ are shown in Table 7. It is interesting to note that while we do not recover the same diagonal coefficients as in [15], we do recover the same off-diagonal value. Further, if we take our result for the 768×768 grid, we have

$$\hat{D} = \begin{bmatrix} 1.9339 & -0.1532 \\ -0.1532 & 1.9339 \end{bmatrix} = Q \begin{bmatrix} 2.0871 & 0 \\ 0 & 1.7807 \end{bmatrix} Q^T.$$

Thus, we see that we have recovered the principal axes exactly (as is done in [15]), and have errors in the scaling in the directions of diffusion of 3.5% and 1.8%, where we are more accurate than the results of [15] in the x-direction, but less accurate in the y-direction.

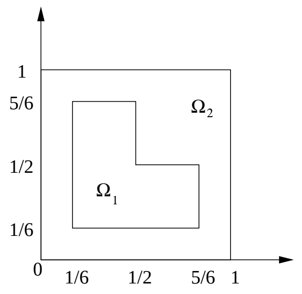

Figure 4: L-shaped Inhomogeneity

Problem Size	$\hat{D}_{11} = \hat{D}_{22}$	$\hat{D}_{12}$	$\hat{E}$	$\hat{D}_{11}^M$	$\hat{D}_{12}^M$
12×12	1.5649	-0.0853	-0.0338	1.4972	-0.0853
24×24	2.0864	-0.1760	-0.2631	2.3766	-0.1760
48×48	1.8619	-0.1401	-0.0903	1.8280	-0.1401
96×96	1.9719	-0.1588	-0.1662	2.0515	-0.1588
192×192	1.9207	-0.1509	-0.1234	1.9316	-0.1509
384×384	1.9467	-0.1552	-0.1437	1.9887	-0.1552
768×768	1.9339	-0.1532	-0.1332	1.9594	-0.1532

Table 7: Recovered coefficients for L-shaped Inhomogeneity

Another classical homogenization problem is one of layered media, such as that pictured in Figure 5. For this problem, the fine-grid permeability is given by

$$D(x,y) = \begin{cases} \lambda_1 & y \leq \frac{1}{2} \\ \lambda_2 & \text{otherwise} \end{cases},$$

and the fully homogenized permeability is known to be

$$\hat{D} = \begin{bmatrix} \frac{\lambda_1+\lambda_2}{2} & 0 \\ 0 & \frac{2\lambda_1\lambda_2}{\lambda_1+\lambda_2} \end{bmatrix}.$$

Moulton et. al. [15] show that they can successfully recover the homogenized permeability in a similar problem with error proportional to h. We have considered this problem with $\lambda_1 = 10^{-2}$ and $\lambda_2 = 1$. For all mesh sizes considered, we recover the $(2,2)$ component of $\hat{D}$ to machine accuracy. However, we do not recover a good approximation to the $(1,1)$ component of $\hat{D}$. Rather, along the southern boundary, we recover $\hat{D}_{11} = 10^{-2}$, the permeability from the fine-grid cells along that boundary. Using the relations derived above, this gives us $\hat{D}_{11}^{i,\frac{3}{2}} = 1$, and then an alternating behavior from this point onwards. We recover $\hat{E}$ alternating between -1.634×10^{-3} and -1.634×10^{-1}, respectively. It is also interesting to note that the actual homogenized permeability for this problem satisfies our interior equations, but not our boundary equations for $\hat{D}_{11}$. This inconsistency along the boundary is a question for future investigation.

Accounting for the regularization term has also given us insight into the interaction between the Galerkin coarsening common in multigrid techniques and coarse-grid relaxation. As discussed in the numerical results above, fine-grid variation in the coefficients of the PDE may, in fact, result in coarse-scale anisotropy of the discretization. If the coarse-grid operators produced by a multigrid method reflected this anisotropy, the effectiveness of relaxation on the coarse-grids would suffer, as would the overall convergence rates of the multigrid algorithm. In practice, however, we see no such degradation in performance. This can be attributed to the presence of the regularization term that allows accurate representation of the coarse-grid anisotropy in the diffusion term by providing a compensation so the overall coarse-grid operator is closer to an isotropic one.

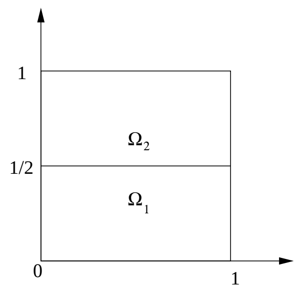

Figure 5: Layered Inhomogeneity

The homogenization viewpoint provides a number of key insights into the performance of multigrid methods. Proper understanding of the relation between scales can lead to improved choices of coarse-grids and of interpolation and restriction operators. In combination, these can give improved coarse-grid operators resulting in faster convergence rates. Of particular interest may be the identification of coarse-grid anisotropies, which could be used to further reduce the number of degrees of freedom present on coarse-grids. Aggressive coarsening such as this would result in faster multigrid cycling and an overall reduced operation count.

7 Future Work

Our numerical results indicate that our full-coarsening based version of the scAMG algorithm does offer some advantages when compared to geometric multigrid or classical AMG. The foci of our future work are improving and extending this algorithm as well as understanding the theoretical roots of its performance. There are many possible avenues for this investigation and we hope that by careful study we will result at an algorithm offering significant improvements in robustness over earlier methods without adding undue computational expense.

One major limitation of our current algorithm is the geometric coarsening used. A simple problem for which this coarsening is insufficient is one with fine-grid anisotropy, such as $u_{xx} + \varepsilon u_{yy} = f$, where ε is small. To overcome this barrier, we plan to investigate the use of more general coarsening algorithms. While the coarsening heuristics used in classical AMG have shown to be suitable for a wide variety of problems, we would like to consider more adaptive forms of coarsening. One candidate method we are particularly interested in is Compatible Relaxation.

Compatible Relaxation is a technique in which the coarse-grid points are chosen based on the effectiveness of relaxation on the fine-grid. As discussed in [5], compatible relaxation can be implemented by choosing an initial coarse-grid (using a priori information if available, or simply as the empty set) and then relaxing on the remaining dofs, taking the contributions from the coarse-grid points to zero. If this fine-grid relaxation (F-relaxation) efficiently reduces the error at every point in F, then the coarse-grid set is deemed sufficient. If, however, there exist fine-grid points that do not show sufficient error reduction, then a subset of these points can be added to the coarse-grid set, and the sufficiency of the coarse-grid can be tested again by compatible relaxation.

To make the most efficient use of this method, we intend to experiment on a set of test problems that expose the difficulty of other schemes. By analyzing the effectiveness of compatible relaxation on anisotropic problems, stretched grid problems, and eventually on systems of PDEs, we hope to get a good idea of an appropriate threshold for the F-relaxation, and an effective algorithm for adding points to the coarse-grid as needed. With such an implementation of this coarsening procedure, we would be able to implement scAMG in a completely algebraic context, not limited by assumptions on the geometry of the original discretization.

Another extension of our work that we are very interested in is that to systems of PDES. Problems such as two- and three-dimensional elasticity and Maxwell's equations, as well as the first-order systems arising from least-squares finite element formulations, all require an efficient solver for their discretized forms. While classical AMG is effective in some cases, its primary assumption that the smoothest global vector is nearly constant results in an algorithm that cannot be expected to accurately handle many systems. Elasticity is a prime example where the differential operator (prior to the imposition of boundary conditions) has null-space components corresponding to translations in each of the coordinate directions and rotations in each of the coordinate planes (the rigid-body modes). Indeed, in many systems problems the idea of smoothness that is fundamental to our definition of interpolation can only be stated as a relationship between the system variables.

While the extension to systems discussed in Sections 3 and 4 has many nice properties, it is possible that it, too, will prove insufficient for these problems, mainly because it only allows for d near null-space components to be accounted for. Three-dimensional elasticity, for example, has six distinct near null-space components, and so we expect some difficulty generating an efficient algorithm by accounting for a subspace of them of dimension three. It is for this reason that we expect to investigate the feasibility of adding information from additional near null-space vectors to already existing interpolation operators.

There are many possible ways to implement the addition of this information. Currently, we see one potentially effective way as choosing an update to the current interpolation operator of minimal norm. That is, given a current $C-F$ interpolation operator, W, based on the collection of vectors $x^{(1)}, \ldots, x^{(k-1)}$, and a new vector $x^{(k)}$ (so we can write $X = [x^{(1)}, \ldots, x^{(k)}]$), we choose a new $C-F$ interpolation operator of the form $W + \Delta W$, such that at each point i, ΔW_i is chosen to

$$\min_{\Delta W_i} \|\Delta W_i X_{c_i}^T + (W_i X_{C_i}^T - X_i^T)\|$$

such that $\|\Delta W_i\|$ is also minimized. This choice has the necessary property that it balances appropriate treatment of all vectors previously treated by the interpolation, but also adds in the additional change necessary to treat the new vector, $x^{(k)}$. Important questions that must be addressed include deciding which norms are appropriate for the minimization and also the computability of the minimization.

Finally, we also intend to direct our efforts toward the development of a theory explaining the behavior of both our existing code and its extensions. For example, we are interested in exploring the initial generation of our representative vectors and would like to understand the level of accuracy needed in this process to obtain sufficiently fast convergence rates. In general, we would like to gain a better understanding of the barriers to optimal multigrid convergence, so that we can develop our algorithm to appropriately overcome them.

While we are quite happy with our current results from the trial algorithm, it is clear that there are quite a number of possible avenues for investigation surrounding self-correcting Algebraic Multigrid methods. We hope that by a thorough investigation of these possibilities we will develop an algorithm that exhibits optimal multigrid convergence properties for a large range of problems at a minimal increase in cost. With the improvements already demonstrated and those to be investigated, we feel that the development of a more robust class of multigrid methods can be successfully completed within the proposed thesis research.

References

[1] J. Bourgat, *Numerical experiments of the homogenization method for operators with periodic coefficients*, in Computing Methods in Applied Science and Engineering I, R. Glowinski and J.-L. Lions, eds., Springer, Versailles, 1977, pp. 330–356.

[2] D. Braess, *Finite Elements*, Cambridge University Press, Cambridge, 2001. Second Edition.

[3] J. H. Bramble, *Multigrid Methods*, vol. 294 of Pitman Research Notes in Mathematical Sciences, Longman Scientific & Technical, Essex, England, 1993.

[4] A. Brandt, *Algebraic multigrid theory: The symmetric case*, Appl. Math. Comput., 19 (1986), pp. 23–56.

[5] A. Brandt, *General highly accurate algebraic coarsening*, ETNA, 10 (2000), pp. 1–20. Special issue on the Ninth Copper Mountain Conference on Multilevel Methods.

[6] M. Brezina, A. J. Cleary, R. D. Falgout, V. E. Henson, J. E. Jones, T. A. Manteuffel, S. F. McCormick, and J. W. Ruge, *Algebraic multigrid based on element interpolation (AMGe)*, SIAM J. Sci. Comput., 22 (2000), pp. 1570–1592.

[7] W. L. Briggs, V. E. Henson, and S. F. McCormick, *A Multigrid Tutorial*, SIAM Books, Philadelphia, 2000. Second edition.

[8] A. J. Cleary, R. D. Falgout, V. E. Henson, J. E. Jones, T. A. Manteuffel, S. F. McCormick, G. N. Miranda, and J. W. Ruge, *Robustness and scalability of algebraic multigrid*, SIAM J. Sci. Comput., 21 (2000), pp. 1886–1908.

[9] I. Daubechies, *Ten Lectures on Wavelets*, SIAM Books, Philadelphia, 1992.

[10] J. E. Dendy, *Black box multigrid*, J. Comput. Phys., 48 (1982), pp. 366–386.

[11] L. J. Durlofsky, *Numerical calculation of equivalent grid block permeability tensors for heterogeneous porous media*, Water Resources Research, 27 (1991), pp. 699–708.

[12] M. Holmes, *Introduction to Perturbation Methods*, no. 20 in Texts in Applied Mathematics, Springer-Verlag, New York, 1995.

[13] V. Jikov, S. Kozlov, and O. Oleinik, *Homogenization of Differential Operators and Integral Functionals*, Springer-Verlag, New York, 1994. translated from Russian.

[14] S. Knapek, *Matrix-dependent multigrid homogenization for diffusion problems*, SIAM J. Sci. Comput., 20 (1999), pp. 515–533.

[15] J. D. Moulton, J. E. Dendy, and J. M. Hyman, *The black box multigrid numerical homogenization algorithm*, J. Comput. Phys., 142 (1998), pp. 80–108.

[16] J. W. Ruge and K. Stüben, *Algebraic multigrid (AMG)*, in Multigrid Methods, S. F. McCormick, ed., vol. 3 of Frontiers in Applied Mathematics, SIAM, Philadelphia, PA, 1987, pp. 73–130.

[17] U. Trottenberg, C. Oosterlee, and A. Schüller, *Multigrid*, Academic Press, London, 2001.

[18] X.-H. Wen and J. J. Gómez-Hernández, *Upscaling hydraulic conductivities in heterogeneous media: An overview*, Journal of Hydrology, 183 (1996), pp. ix–xxxii.