Relaxation-corrected Bootstrap Algebraic Multigrid (rBAMG)

by

Minho Park

B.S., Kyungpook National State University, 2004

M.S., University of Colorado at Boulder, 2008

A thesis submitted to the

Faculty of the Graduate School of the

University of Colorado in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

Department of Applied Mathematics

2010

This thesis entitled:
Relaxation-corrected Bootstrap Algebraic Multigrid (rBAMG)
written by Minho Park
has been approved for the Department of Applied Mathematics

Prof. Stephen F. McCormick

Prof. Thomas A. Manteuffel

Date

The final copy of this thesis has been examined by the signatories, and we find that both the content and the form meet acceptable presentation standards of scholarly work in the above mentioned discipline.

Park, Minho (Ph.D., Applied Mathematics)

Relaxation-corrected Bootstrap Algebraic Multigrid (rBAMG)

Thesis directed by Prof. Stephen F. McCormick

Bootstrap Algebraic Multigrid (BAMG) is a multigrid-based solver for matrix equations of the form $Ax = b$. Its aim is to automatically determine the interpolation weights used in algebraic multigrid (AMG) by locally fitting a set of test vectors that have been relaxed as solutions to the corresponding homogeneous equation, $Ax = 0$. This thesis introduces an indirect operator-based interpolation scheme for BAMG that determines the interpolation weights indirectly by "collapsing" the unwanted connections in "operator interpolation". Compared to BAMG, this *indirect* BAMG approach (iBAMG) is more in the spirit of classical AMG, which collapses unwanted connections in operator interpolation based on the (restrictive) assumption that smooth error is locally constant.

This thesis also develops another form of BAMG, called rBAMG, that involves modifying the least-squares process by temporarily relaxing on the test vectors at the fine-grid interpolation points. The theory here shows that, under fairly general conditions, iBAMG and rBAMG are equivalent. Simplicity and potentially greater generality favor rBAMG, so this algorithm is at the focus of the numerical performance study here.

The rBAMG setup process involves several components that are developed in this thesis. Besides the new least-squares principle involving the residuals of the test vectors, a simple extrapolation scheme is developed to accurately estimate the convergence factors of the evolving AMG solver. Such a capability is essential to effective development of a fast solver, and the approach introduced here proves to be much more effective than the conventional approach of just observing successive error reduction factors. Another component of the setup process is the use of the current V-cycle to ensure its effectiveness or, when poor convergence is observed, to expose error components that are not being properly attenuated. How we coordinate use of these evolving error components together with the original test vectors to direct the setup process is a critical issue to

rBAMG's effectiveness. Another related component is the scaling and recombination Ritz process that targets the so-called weak approximation property in an attempt to reveal the important elements of these evolving error and test vector spaces. The details of the components used here are spelled out in what follows.

The study of rBAMG here is an attempt to systematically analyze the behavior of the algorithm in terms relative to several parameters. The focus here is on the number of test vectors, the number of relaxations applied to them, and the dimension of the matrix to which the scheme is applied. A large number of other parameters and options could also be considered, including different cycling strategies, other coarsening strategies (e. g., computing several eigenvector approximations on coarse levels), different numbers of relaxation sweeps on coarse levels, different possible strategies for combining test vectors and error components produced by the current cycles, and so on. Studying all of these options and parameters would not be feasible here. Instead, reasonable choices are made based on some sample studies (that, in the interest of space, we choose not to document here), with the hope that the rBAMG algorithm studied here is generally fairly effective and robust. Our analysis is thus able to focus on how this scheme behaves numerically in the face of increasing the numbers of test vectors and relaxation sweeps performed on them, as well as the problem sizes.

Dedication

To

my parents

Jongil Park and Sunhee Kim

my wife

Jimin Shon

my daughter

Lucy Park

Acknowledgements

First and foremost, I would like to express my deep and sincere appreciation to my advisors, Steve McCormick and Tom Manteuffel, who provided me with encouragement, insights, good advice, and continuous feedback, which was invaluable and helped my achievement that you will find documented in this thesis. Working under their supervision has been privilege, and I greatly appreciate their constant support.

My thanks also go to other members of my thesis committee who guided my research: John Ruge, Marian Brezina, and Scott MacLachlan They were generous with their help whenever I met difficulties in my study. Their knowledge and experience helped me overcome many obstacles.

I would also like to thank my Korean advisor, Sangdong Kim, who led me to the world of Mathematics. Without his help, I would not even have started my degree at Boulder.

I would also like to thank my colleagues, Christian Ketelsen, Jacob Schroder, Jose Garcia, Kuo Liu, and Lei Tang, for their co-operation and contribution, and former graduate students, James Adler and Geoff Sanders, for their kind a fruitful collaboration and advice.

Last, but not least I wish to extend my heartfelt appreciation to my parents for their encouragement and love, and my wife, Jimin, who deserves my deepest gratitude and love for her dedication and support during my graduate studies.

Contents

Chapter

Tables

Table

Figures

Figure

Chapter 1

Introduction

Due to its potential to solve $N \times N$ sparse linear system of equations,

$$Ax = b, \tag{1.1}$$

with only $O(N)$ work, multigrid methods have gained a widespread use for solving the large sparse linear system that arises from the discretization of partial differential equations (PDEs). This popularity is due to the efficiency that results from two complementary processes: *smoothing* and *coarse-grid correction*. The basic idea of the classical geometric multigrid method is that relaxation is inexpensive and efficient at eliminating high-frequency (oscillatory) error, while low-frequency (smooth) error that remains after relaxation can be eliminated on a coarser grid, again by relaxation and further elimination on yet coarser grids. Unfortunately, many modern problems involve discontinuous coefficients, complex geometries, and unstructured grids, which inhibit geometric multigrid from being applied.

In contrast to standard multigrid methods, algebraic multigrid (AMG [7, 8, 16, 25]) methods need not take into account any geometric information about the underlying problem. Instead, AMG relies on an algebraic sense of smoothness to construct a hierarchy of matrices and intergrid transfer operators automatically based on the information of the original matrix. AMG can be classified according to the way in which the interpolation operator is defined based on the expected characteristics of *algebraically smooth error* components, which are defined simply as the error that cannot be attenuated efficiently by relaxation. Classical AMG is based on the assumption that relaxation cannot efficiently resolve errors that are locally constant. In classical smoothed aggregation

(SA [27, 28]), representative smooth vectors supplied by the user are employed to define columns of the interpolation operator locally. These user-supplied vectors then define implicitly what is expected of algebraically smooth errors. While appropriate use of the characteristics of algebraic smoothness seems essential for obtaining effective solvers, these additional assumptions limit the scope of applicability of such methods. What is needed are self-learning algebraic multigrid solvers that automatically determine the full character of algebraically smooth errors. Robust multigrid solvers with this capability could dramatically increase the applicability of optimal multigrid solvers over a wide range of discrete problems.

The most recent research in this direction is concerned primarily with the development of self-learning multigrid algorithms, including the original *adaptive* AMG algorithm introduced in [8], the bootstrap AMG approach introduced in [4] and developed further for quantum chromodynamics in [6], an adaptive scheme based on smoothed aggregation (SA) developed in [13] and [14] and developed further for lattice QCD in [9], adaptive AMG schemes developed further in [15], and an adaptive reduction-based AMG algorithm introduced in [23] and [11]. One principal difference among these self-learning multigrid schemes is that the so-called adaptive approaches have typically started with just one test vector, while the so-called bootstrap approaches generally start with several. Both schemes produce the test vectors initially by starting with random initial guesses to the corresponding homogeneous problem, $Ax = 0$. The adaptive approach usually constructs interpolation initially to fit a single initial test vector, and then tests the resulting solver on the homogeneous problem, starting from another random initial vector. If observed convergence is not yet acceptable, then the resulting vector is either used to enhance the original test vector or else added to the test-vector set. The process then continues until acceptable convergence is observed. BAMG instead constructs interpolation to fit (in a least-squares sense) typically several initial test vectors. The adaptive schemes are advantageous in that they naturally sort out a rich and locally independent set of test vectors: a properly implemented adaptive scheme should be able to generate a local representation of algebraic smoothness that is rich and independent in the sense that each local vector represents a new character of algebraic smoothness. Unfortunately,

adaptive approaches that begin with a single test vector and introduce additional vectors one at a time can be costly in their initial stages. This can happen because an effective coarse-grid solver must be developed before returning to the fine grid if the quality of interpolation is to be safely assessed. Otherwise, it would be difficult to determine whether slow convergence is due to a poor interpolation operator or a poor coarse-level solver.

To address this concern, we develop a version here of adaptive AMG that begins with several test vectors that are chosen randomly and subjected to relaxation (for solving the homogeneous equation), and are then used in a modified least-squares process of fitting interpolation. While this version fits into the adaptive methodology, we refer instead to it as a BAMG scheme because using multiple initial vectors is more in the spirit of the bootstrap methodology.

The aim of the least-squares process is to produce a V-cycle for solving (1.1) that converges quickly in a sense that we describe below. Our approach is to use the initially constructed interpolation operator to create a V-cycle and, in the subsequent adaptive phase, test this current solver applied to a random initial guess for the homogeneous equation, $Ax = 0$. For focus and simplicity of analysis, we define this adaptive phase in a non-recursive way in that we only test the effectiveness of the current solver on the finest level. As we said, the usual adaptive AMG approach is to test solver effectiveness recursively on all levels, and not return to finer levels until adequate performance is observed on the coarse levels. (The original adaptive AMG takes a different approach by returning directly to the finer grid.) A related recursive approach is also what is now being studied in the BAMG methods [6]. However, we choose a non-recursive form of the adaptive phase because it facilitates a systematic focus on the other components of the AMG processes.

The focus of this thesis is on developing strategies that allow for fewer test vectors than are currently used in bootstrap methods and more efficient self-learning algorithms that are able to fit several smooth vectors by least-squares approximation. Because of this focus, we obviate the problem of choosing a good coarse grid by considering cases where this is known in advance. For a compatible relaxation approach to coarse-grid selection, see [3], [5], [6], and [10].

This thesis is organized into six chapters. Chapter 2 discusses the multigrid methods we use

in detail. In Section 2.1, we discuss relevant fundamental multigrid principles. Section 2.2 reviews the basic components of classical AMG and how we define the AMG interpolation operator. Section 2.3 briefly shows how to compute adaptive AMG interpolation weights.

Chapter 3 introduces a new version of BAMG that determines the interpolation weights indirectly by "collapsing" the unwanted connections in "operator interpolation". Compared to BAMG, this *indirect* BAMG approach (iBAMG) is more in the spirit of classical AMG, which collapses unwanted connections in operator interpolation based on the assumption that smooth error is locally constant. This chapter also introduces a relaxation-corrected version, rBAMG, that is equivalent to, but possibly easier to implement, than iBAMG.

Chapter 4 introduces the non-recursive adaptive algorithm based on bootstrap AMG interpolation. Section 4.1 explains the algorithm more in detail. Section 4.2 introduces and tests the convergence estimation model that we use in the adaptive process as an accurate indicator of the speed of convergence of the current solver. In Section 4.3, we consider the theoretical setup cost of this adaptive process.

In Chapter 5, we report on numerical experiments that confirm effectiveness of the indirect approach and the non-recursive adaptive algorithm.

Chapter 6 summarizes the results presented here and provides some recommendations for future work.

Chapter 2

Background

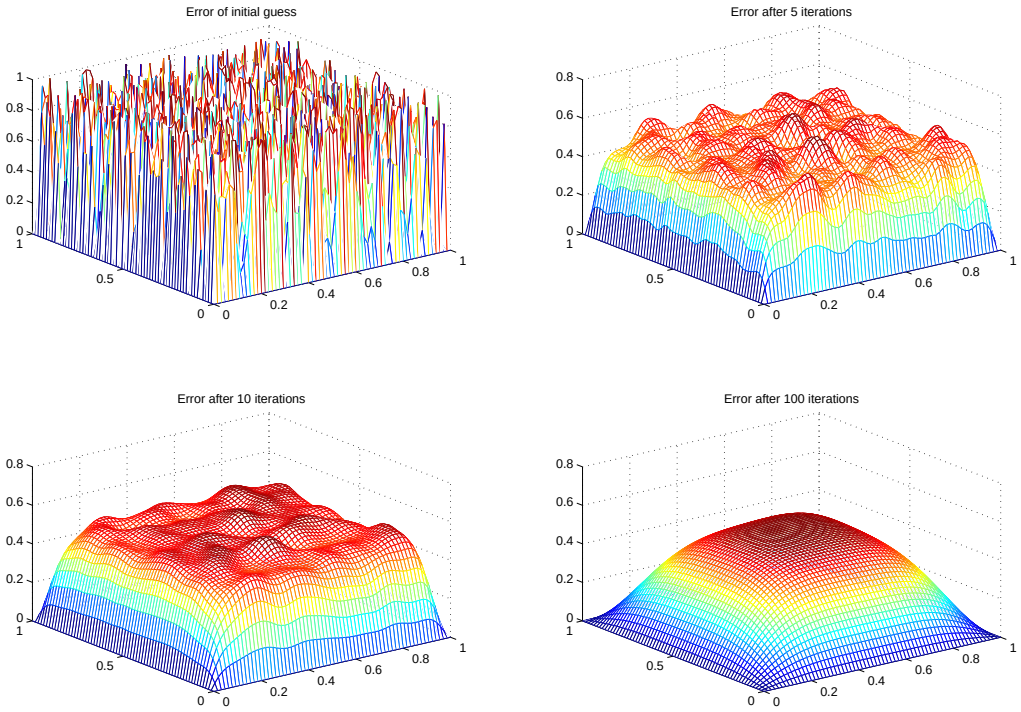

Figure 2.1: Influence of Gauss-Seidel iteration on 2D Poisson.

2.1 Multigrid Principles

In this section, we briefly introduce and discuss two ingredients of multigrid methods: *smoothing* and *coarse-grid correction*. Pointwise relaxation methods like Jacobi and Gauss-Seidel are ef-

fective at reducing high-frequency (oscillatory) error, but are often poor at reducing low-frequency (smooth) error. After a few iteration sweeps, the error thus tends to become very smooth. Figure 2 illustrates this error smoothing property of the Gauss-Seidel method.

The principal aim of the multigrid processes, then, is to eliminate the smooth error that remains after relaxation. Since a coarse-grid problem is much cheaper to solve than a fine-grid one, the remedy is to approximate this error on a coarse grid where it can again be treated by relaxation and correction from yet coarser grids. This process is motivated in part by the property (shown in Figure 2.2) that smooth fine-grid error becomes relatively more oscillatory on coarse levels. Given an approximation, v, to the exact solution, the algebraic error, e, is the difference between the exact solution and the current approximation, i.e., $e = A^{-1}b - v$. The error is unknown, but determined by the residual:

$$r = b - Av = A(A^{-1}b - v) = Ae.$$

Each two grid correction scheme consists of *presmoothing*, *coarse-grid correction*, and *postsmoothing*. One step of such a method proceeds as follows:

Algorithm 2.1 Two-Grid Correction Scheme

(Presmoothing) Relax ν_1 times on $A^h x^h = b^h$ on the fine level with initial guess v^h.
(Coarse-grid correction)
- Compute the fine-grid residual $r^h = b^h - A^h v^h$.
- Restrict the residual by $r^{2h} = Rr^h$.
- Solve $A^{2h}e^{2h} = r^{2h}$ on the coarse grid.
- Interpolate the coarse-grid error approximation to the fine grid by $e^h = Pe^2h$
- Compute a new approximation by $v^h \leftarrow v^h + e^h$

(Postsmoothing) Relax ν_2 times on $A^h x^h = b^h$ on the fine level with initial guess v^h.

The method can be extended to a V-cycle scheme when it is done recursively in that the coarse-grid problem is solved in the same way as the fine-grid equations, by introducing yet coarser grids.

Unfortunately, many problems cannot be easily treated by coarsening in a geometrically based way (e. g., those arising from discretizations based on highly irregular grids) and many more still do not exhibit the property that relaxation produces geometrically smooth error (e. g., highly

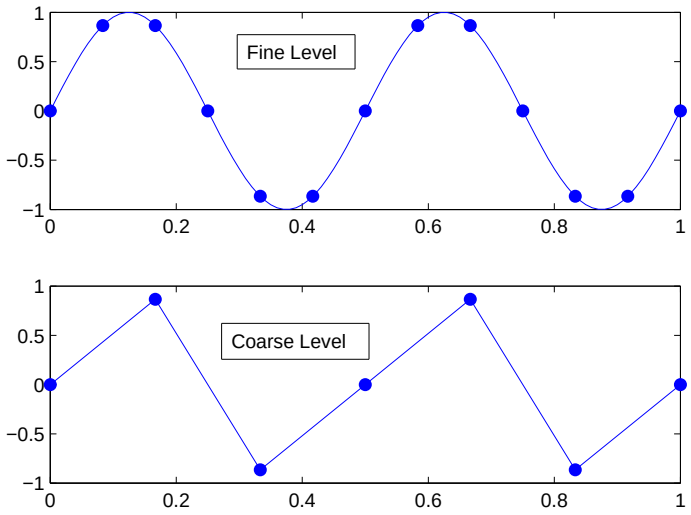

Figure 2.2: Smooth error becomes more oscillatory on the coarse grid.

anisotropic problems, stochastic problems like those that arise in quantum chromodynamics, and problems whose unknowns are scaled in a way that is not available to the solver).

2.2 Classical Algebraic Multigrid

Assume in what follows that $A = (a_{ij}) \in \Re^{n \times n}$. This section is devoted to describing how classical AMG applied to $Ax = b$ determines the weights in the interpolation operator, which we denote by P. Because our focus is on the weights of interpolation as opposed to how the coarse grids are selected, we assume that the fine-level points have already been partitioned into points that are identified with the coarse grid, the set of which we denote by C, and its complement, which we denote by F. This partition into C and F points gives rise to the AMG form of interpolation described as follows: the ith entry of Pe is given by

$$(Pe)_i = \begin{cases} e_i & \text{if } i \in C, \\ \sum_{j \in C_i} w_{ij} e_j & \text{if } i \in F. \end{cases} \tag{2.1}$$

Here, C_i is the subset of C consisting of points that are used to interpolate to point $i \notin C$. Our task now is to describe in abstract terms how the interpolation weights, w_{ij}, are determined. Therefore,

for the remainder of this section, we consider a given fixed $i \in F$.

Interpolation only needs to approximate error that is not easily attenuated by relaxation. This observation gives rise to the first AMG premise: relaxation produces error, e, that is algebraically smooth in the sense that it exhibits a relatively small residual. Therefore, we can assume that $(Ae)_i \approx 0$, that is, that

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

Consider now the splitting of this sum into its component sums over C_i, the coarse interpolatory set, and its complement C_i^c, the remaining grid points in the *neighborhood* of i, by which we mean the set of points that are connected to i in A (i. e., all points ℓ such that $a_{i\ell} \neq 0$). We assume for simplicity in this thesis, unless otherwise stated, that C_i^c is taken only from the neighborhood of i with strong influence, so it consists of connected F points and other connected C points that are not in C_i. See [16] for more detailed information about strong and weak connections. With this splitting, we obtain

$$a_{ii}e_i \approx -\sum_{j \in C_i} a_{ij}e_j - \sum_{k \in C_i^c} a_{ik}e_k. \tag{2.2}$$

Observe that, if the second component sum happens to vanish in this approximation (e. g., $a_{ik} = 0$ for $k \in C_i^c$), then we would immediately have a formula that expresses the value of any algebraically smooth error at point i by its value at points of C_i. This "operator interpolation" formula would then yield appropriate weights for P given by $w_{ij} = -a_{ij}/a_{ii}$. This observation suggests, for the general case $\sum_{k \in C_i^c} a_{ik}e_k \neq 0$, that we may want to "collapse" the unwanted connections (a_{ik} for $k \in C_i^c$) to C_i. Thus, we are led to replacing the e_k in the second sum on the right side of (2.2) with sums that involve only e_j for $j \in C_i$.

To replace each e_k, $k \in C_i^c$, with a linear combination of the e_j, $j \in C_i$, we need to make a further assumption about the nature of smooth error. Since the historical target for AMG are partial differential equations of elliptic type, the classical AMG premise is that smooth error is locally almost constant. This second AMG premise means that we can assume that each e_k is any convex linear combination of the e_j, $j \in C_i$, that preserves constants. AMG is based on the

particular linear combination where the coefficients are proportional to the connections from point k to each point j, that is, it is based on the approximation

$$e_k \approx \sum_{j \in C_i} \frac{a_{kj}}{\sum\limits_{\ell \in C_i} a_{k\ell}} e_j. \tag{2.3}$$

Substituting this expression into (2.2) and dividing the result by a_{ii} yields interpolation weights given by

$$w_{ij} = \frac{1}{a_{ii}} \left(-a_{ij} - \sum_{k \in C_i^c} \left(a_{ik} \frac{a_{kj}}{\sum\limits_{\ell \in C_i} a_{k\ell}} \right) \right). \tag{2.4}$$

This process of collapsing the unwanted connections in the operator interpolation formula expressed by (2.2) can be viewed as using a crude but properly scaled *truncated* interpolation formula, expressed by (2.3), to interpolate from C_i to e_k. (We refer to (2.3) as truncated because it amounts to operator interpolation at point k where we have simply deleted the unwanted terms that do not belong to C_i. It is properly scaled in the sense that it is exact for constants.) This indirect process has the effect of collapsing the unwanted connections, and it leads to the direct formula for interpolation weights defined in (2.4).

2.3 Adaptive AMG Interpolation

To complement relaxation, all algebraically smooth errors must be approximated well by vectors in the range of interpolation. Hence, we need to focus on the development of an interpolation operator that is highly accurate for algebraically smooth vectors, which is one of subjects of this thesis. The original adaptive AMG (αAMG [15, 22]) interpolation process is different from that used in classical AMG in that not only the entries in A, but also a representative algebraically smooth vector x, are used to compute the interpolation weights. We seek interpolation weights that fit x locally. With this in mind, the formula we use for collapsing unwanted connections is given by

$$e_k \approx \sum_{j \in C_i} \frac{a_{kj} x_k}{\sum\limits_{\ell \in C_i} a_{k\ell} x_l} e_j. \tag{2.5}$$

Note that (2.5) holds when e is replaced by x, so both classical AMG and adaptive AMG are identical when x is constant. One of the objectives of this thesis is to develop a similar process for adaptive AMG that accounts for several representative algebraically smooth vectors. This is what we do in effect in the next chapter when we develop an indirect-interpolation least-squares process for BAMG.

Chapter 3

Least-Squares-Based Interpolation

Assume for the remainder of this thesis that $A \in \Re^{n \times n}$ is symmetric and positive definite. Suppose now that we are given a set of test vectors, $e^{(l)}$, $l = 1, 2, \ldots, q$, that result from ν fine-level relaxation sweeps on the homogeneous equation, $Ae = 0$, starting from q distinct random approximations. (We assume that the initial random vectors are of unit length in the Euclidean norm to avoid significant scale disparity in the least-squares processes that follow.) Since the focus is on the process of determining interpolation weights, we continue to assume that the fine-level points have already been partitioned into the C-points that are identified with the coarse grid and its F-point complement set. We also assume that the coarse-grid interpolatory set, C_i, has already been determined for each F-point i. Also, unless otherwise noted (in particular, see Section 3.5), we assume that the vectors are locally rich in that they are locally independent and numerous enough to ensure that all of the least-squares problems we introduce are uniquely solvable.

3.1 Bootstrap Algebraic Multigrid

The general form of interpolation operator, P, for AMG is given in (2.1). As described in the previous section, the choice of weights w_{ij} is dictated by two basic premises: relaxation produces small residuals and relaxed errors are locally almost constant. The first premise is very general: many matrix equations can be treated by relaxation schemes that produce small residuals in a sense that leads to usable local operator interpolation formulas. However, many circumstances arise where errors are not approximately constant in any local sense, so the second premise seriously

restricts the applicability of AMG methods. The basic idea behind BAMG is to glean the local character of algebraically smooth errors from the set of test vectors. This leads to a determination of the interpolation weights by a direct least-squares fit of the target vectors. Thus, for each $i \in F$, we compute

$$\text{(BAMG)} \qquad \{w_{ij} : j \in C_i\} = \arg\min_{w_{ij}} \sum_{l=1}^{q} (e_i^{(l)} - \sum_{j \in C_i} w_{ij} e_j^{(l)})^2. \tag{3.1}$$

Note that we assume that all target vectors have the same quality of smoothness, so sorting is not necessary. Scaling, on the other hand, becomes crucial in the case of varying smoothness. We discuss this in more detail in Section 4.3.4.

BAMG has generally taken this direct approach to determining the weights of interpolation. More in the spirit of classical AMG as described in Chapter 2, we now introduce an indirect approach based on collapsing the unwanted connections in operator interpolation.

3.2 Indirect Bootstrap Algebraic Multigrid (*i*BAMG)

As with classical AMG, the starting point for determining the interpolation weights is the residual relation expressed in (2.2), again with C_i^c denoting the complement of C_i in the neighborhood of i. In particular, we assume nonzero *unwanted* connections: $a_{ik} \neq 0$ for all $k \in C_i^c$. The objective now is to collapse these connections by approximating the last sum in the residual equation by a linear combination of the errors at the C_i points. The departure point here is that we can no longer assume that the target error is approximately constant. Instead, we use the test vectors to provide the sense of smoothness that we need. As before, once the unwanted connections have been collapsed, we can use the residual relation to write the F-point error, e_i, directly as a linear combination of the e_j for $j \in C_i$, which then yields the desired interpolation weights.

In classical AMG, an approximation is made separately for each e_k with $k \in C_i^c$, so this is a natural approach to take here: for each $k \in C_i^c$, we seek weights β_{kj}, dependent on i, such that

$$e_k \approx \sum_{j \in C_i} \beta_{kj} e_j.$$

Analogous to the BAMG approach, for this indirect interpolation problem, we use least squares to determine each β_{kj}:

$$(i\text{BAMGa}) \qquad \{\beta_{kj} : j \in C_i\} = \arg\min_{\beta_{kj}} \sum_{l=1}^{q} (e_k^{(l)} - \sum_{j \in C_i} \beta_{kj} e_j^{(l)})^2. \tag{3.2}$$

This process results in the approximation

$$\sum_{k \in C_i^c} a_{ik} e_k \approx \sum_{j \in C_i} \sum_{k \in C_i^c} a_{ik} \beta_{kj} e_j$$

and resulting interpolation weights

$$w_{ij} = \frac{1}{a_{ii}} \left(-a_{ij} - \sum_{k \in C_i^c} a_{ik} \beta_{kj} \right). \tag{3.3}$$

Compare this expression with the weights for classical AMG given in (2.4). Note that these weight formulas agree for the case where the β_{kj} reduce to the truncated interpolation formula given in (2.3).

An alternative to separately collapsing unwanted connections is to approximate all of the connections at once: for each $k \in C_i^c$, we seek weights α_j, again dependent on i, such that

$$(i\text{BAMGb}) \qquad \{\alpha_j : j \in C_i\} = \arg\min_{\alpha_j} \sum_{l=1}^{q} (\sum_{k \in C_i^c} a_{ik} e_k^{(l)} - \sum_{j \in C_i} \alpha_j e_j^{(l)})^2. \tag{3.4}$$

This yields the simpler approximation

$$\sum_{k \in C_i^c} a_{ik} e_k \approx \sum_{j \in C_i} \alpha_j e_j$$

and resulting interpolation weights

$$w_{ij} = \frac{1}{a_{ii}} (-a_{ij} - \alpha_j). \tag{3.5}$$

3.2.0.1 *i*BAMGa and *i*BAMGb Equivalence

Fitting the interpolation weights of all of the unwanted connections at once for each $i \in F$ as iBAMGb does is simpler and less expensive than fitting these weights individually as iBAMGa does. So it is important to know that these two approaches actually yield the same weights, provided the least-squares problems are well posed in the sense that the normal equation operator is nonsingular.

Lemma 3.2.1. *Denote the vector of values of $e^{(l)}$ at points of C_i by $\varepsilon^{(l)}$ and let L be the $|C_i| \times |C_i|$ matrix defined by*

$$L = \sum_{l=1}^{q} \varepsilon^{(l)} \varepsilon^{(l)T}. \tag{3.6}$$

If L is nonsingular, then definitions (3.3) and (3.5) are equivalent.

Proof. For each $k \in C_i^c$, let β_k denote the vector of values β_{kj}, $j \in C_i$. Also let α denote the vector of values α_j, $j \in C_i$. Then least-squares problem (3.2) results in the normal equation

$$L\beta_k = \sum_{l=1}^{q} e_k^{(l)} \varepsilon^{(l)}, \tag{3.7}$$

for each $k \in C_i^c$, while least-squares problem (3.3) results in just the one normal equation

$$L\alpha = \sum_{l=1}^{q} \sum_{k \in C_i^c} a_{ik} e_k^{(l)} \varepsilon^{(l)}. \tag{3.8}$$

The equivalence between (3.3) and (3.5) now follows from the unique solvability of (3.7) and (3.8) and the relation

$$\alpha = \sum_{k \in C_i^c} a_{ik} \beta_k. \tag{3.9}$$

□

An important implication of this lemma is that, if each connection to C_i^c is collapsed to all C_i points using a rich-enough set of test vectors (note, in particular, that we must have at least $|C_i|$ test vectors), then the combined approach of *i*BAMGb is to be preferred because it is equivalent to, but less expensive than, *i*BAMGa. However, because of its greater flexibility, *i*BAMGa may be useful in cases where different subsets of the interpolatory points are used for each unwanted connection or the set of test vectors is somehow deficient. We demonstrate this flexibility in Section 3.5 below.

3.2.1 BAMG and *i*BAMG Conditional Equivalence

The motive behind *i*BAMG is to attempt to insulate coarsening from a crude interpolation formula by relegating this formula to the unwanted and hopefully less important connections to i

from C_i^c. The hope is that any crudeness in determining the weights would have less impact if it were used for collapsing the connections indirectly than it would with the approach of determining interpolation weights directly. It is interesting to observe that the indirect and direct approaches are also equivalent in the special case that the residuals for all test vectors at point i are 0.

Lemma 3.2.2. *Suppose again that the $|C_i| \times |C_i|$ matrix L defined by (3.6) is nonsingular. Then BAMG and iBAMG (either version) are conditionally equivalent in the sense that they give the same interpolation weights at any point i for which all test-vector residuals are null: $r_i^{(l)} \equiv \left(Ae^{(l)}\right)_i = 0$, $l = 1, 2, \cdots, q$.*

Proof. Denote the vector of values of w_{ij} at points of C_i by w_i and note that the normal equation for the BAMG least-squares problem in (3.1) can be written as

$$Lw_i = \sum_{l=1}^{q} e_i^{(l)} \varepsilon^{(l)}. \tag{3.10}$$

The right side of this equation can be rewritten as

$$\sum_{l=1}^{q} e_i^{(l)} \varepsilon^{(l)} = \frac{1}{a_{ii}} \left(\sum_{l=1}^{q} (r_i - \sum_{j \in C_i} a_{ij} e_j^{(l)} - \sum_{k \in C_i^c} a_{ik} e_k^{(l)}) \varepsilon^{(l)} \right).$$

Letting a_i be the vector of coefficients a_{ij}, $j \in C_i$, and using the premise that $r_i = 0$, we then have

$$\sum_{l=1}^{q} e_i^{(l)} \varepsilon^{(l)} = -\frac{1}{a_{ii}} \left(La_i + \sum_{l=1}^{q} \sum_{k \in C_i^c} a_{ik} e_k^{(l)} \varepsilon^{(l)} \right).$$

From (3.8), we can then rewrite the right side of (3.10) as

$$\sum_{l=1}^{q} e_i^{(l)} \varepsilon^{(l)} = -\frac{1}{a_{ii}} L \left(a_i + \alpha \right),$$

which in turn yields

$$w_i = \frac{1}{a_{ii}} (-a_i - \alpha).$$

□

3.2.2 Residual-corrected BAMG (rBAMG)

This equivalence for the case that $r_i = 0$ can be exploited to improve BAMG simply by incorporating the residual in the least-squares process. Specifically, the least-squares problem for BAMG given by (3.1) can be modified by the addition of the local scaled residual as follows:

$$(r\text{BAMG}) \qquad \{w_{ij} : j \in C_i\} = \arg\min_{w_{ij}} \sum_{l=1}^{q} (e_i^{(l)} - \sum_{j \in C_i} w_{ij} e_j^{(l)} - \frac{r_i^{(l)}}{a_{ii}})^2. \tag{3.11}$$

This change to the fitting process yields a new scheme, which we call rBAMG, that is equivalent to iBAMG for the case that unwanted connections are collapsed to all of C_i and the target vectors are rich enough locally to guarantee a unique fit. This change should therefore improve the direct approach insofar as our numerical tests show the superiority of iBAMG. Thus, we can expect this improved approach to offer better performance for a given number of target vectors and relaxation steps applied to them.

Note that this modification to the direct scheme is equivalent to temporarily relaxing the equation at point i and then applying the standard BAMG minimization approach. As such, rBAMG is related in spirit to the adaptive relaxation scheme described by Brandt in [1] (and suggested in [3]) that applies relaxation selectively to points exhibiting especially large residuals.

An important implication of this equivalence is that all of the machinery that has so far been developed for BAMG now applies in effect to iBAMG. For example, this includes processes for assessing the quality of the current coarse level (i. e., the $C - F$ partition and C_i) as well as the processes that are designed to improve them (see [4] and [5] for further detail).

A result in [15] shows that adaptive AMG is invariant to diagonal scaling in the sense that symmetrically scaling A by any positive diagonal matrix does not change the results, provided the random test vectors are commensurately scaled. This invariance property is important in part because it confirms some sense of stability of the algorithm. As our next lemma shows, rBAMG is also scale invariant.

To be specific, let D be any positive diagonal matrix. With the given C/F-splitting, matrices

A and D can be written in block form as follows:

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

Lemma 3.2.3. *Let $\hat{A} = DAD$. Then rBAMG applied to $Ax = b$ with target vectors $e^{(l)}$, $l = 1, ..., q$, and rBAMG applied to $\hat{A}\hat{x} = \hat{b}$ with target vectors $\hat{e}^{(l)} = D^{-1}e^{(l)}$, $l = 1, ..., q$, are equivalent in the sense that the resulting interpolation operators are related by $\hat{P} = D^{-1}PD_c$.*

Proof. Noting that $\hat{r}_i^{(l)} = \left(\hat{A}\hat{e}^{(l)}\right)_i = \left(DADD^{-1}e\right)_i = (DAe)_i = d_{ii}r_i$, then the weights for $\hat{A}$ are given by

$$\begin{aligned} \{\hat{w}_{ij}\} &= \arg\min \sum_{l=1}^{q} (\hat{e}_i^{(l)} - \sum_{j \in C_i} \hat{w}_{ij}\hat{e}_j^{(l)} - \frac{\hat{r}_i^{(l)}}{\hat{a}_{ii}})^2 \\ &= \arg\min \sum_{l=1}^{q} (\frac{e_i^{(l)}}{d_{ii}} - \sum_{j \in C_i} \hat{w}_{ij}\frac{e_j^{(l)}}{d_{jj}} - \frac{d_{ii}r_i^{(l)}}{d_{ii}^2 a_{ii}})^2 \\ &= \arg\min \frac{1}{d_{ii}^2} \sum_{l=1}^{q} (e_i^{(l)} - \sum_{j \in C_i} \hat{w}_{ij}\frac{d_{ii}}{d_{jj}}e_j^{(l)} - \frac{r_i^{(l)}}{a_{ii}})^2 . \end{aligned}$$

Thus, if w_{ij} minimizes (3.11), then so does $\hat{w}_{ij}\frac{d_{ii}}{d_{jj}}$. Hence, we can write the interpolation operator, $\hat{P}$, for $\hat{A}$ in the form

$$\hat{P} = \begin{pmatrix} \hat{W} \\ I \end{pmatrix} = \begin{pmatrix} D_f^{-1}WD_c \\ I \end{pmatrix} = D^{-1}PD_c ,$$

where P is the interpolation operator for A. This proves the assertion. □

This lemma confirms that rBAMG is invariant under symmetric positive diagonal scaling in the sense that the convergence of the process is unchanged and the resulting interpolation operators are related via the diagonal transformation. This also confirms that the resulting multigrid solvers are related in the same way, provided the relaxation processes possess this invariance property.

3.2.3 Underdetermined Case

The equivalence results obtained above, together with the improved performance of iBAMG observed in the next section, suggests that our *residual-correction* modification to BAMG should

generally lead to the need for fewer targets smoothed fewer times. In fact, we may want to consider how well rBAMG performs when the number of targets is smaller than the number of points of C_i, that is, when $q < |C_i|$ so that least-squares problem (3.11) has infinitely many solutions. To decide how to select a sensible solution, we take our cue from iBAMG. When the least-squares problem for the indirect approach has many solutions, it is important for accuracy alone to control the size of the resulting weights. It thus seems natural to select the solution of (3.4) with minimal Euclidean norm. This translates to computing the weights for rBAMG that deviate least in the Euclidean norm from those obtained by operator truncation: find the least-squares solution, w_{ij}, of (3.11) with *minimal deviation* from $-a_{ij}/a_{ii}$ in the sense of minimizing

$$\sum_{j \in C_i} (w_{ij} + \frac{a_{ij}}{a_{ii}})^2. \tag{3.12}$$

The scaled operator coefficients given by $-a_{ij}/a_{ii}$ in (3.12) are "default" weights in the sense that the objective is to stay as close to them as possible when the least-squares fit to the targets is underdetermined. These defaults are not necessarily good weights to use in the adaptive process because they correspond to truncating the unwanted connections, which, in the model problem, leads to improperly scaled weights. (Properly scaling in this case can instead be obtained by collapsing the unwanted connections to the diagonal, for example.) However, it should be kept in mind that, generally, these defaults would be selected only in the unrealistic case that no targets are available. Just one target is usually enough to adjust the weights from these targets to obtain interpolation that is properly scaled.

Note that we are not prevented from using i in the definition of the weights in (3.12) and the form of interpolation in (3.1). Note also that nothing is forcing us to restrict C_i to the immediate neighborhood of i: it may include points outside of i's nearest neighborhood, perhaps at times *only* points outside this neighborhood. However, studying these possibilities is beyond the scope of this thesis and is therefore left for further research.

3.3 Direct-Connection-Based BAMG (*di*BAMG)

In both classical and adaptive AMG [15, 22], each e_k, where k is a fine-grid point, is replaced by a linear combination of values of e_j from set C_i, and weights in this linear combination are defined in proportion to the matrix entries a_{kj}, so only direct connections have nonzero weights. Unlike these two methods, iBAMG collapses each F-F connection (or all F-F connections) to all C_i points with no explicit concern for the proportions of the matrix entries. To understand the effect of this property, this section introduces a modified iBAMG algorithm that collapses each F-F connection only to its strong connections. We achieve this strategy by replacing interpolatory set C_i in (3.2) by $C_i \cap N_k$, where N_k is the set of neighboring points of k. This yields

$$(di\text{BAMG}) \qquad \{\beta_{kj} : j \in C_i\} = \arg\min_{\beta_{kj}} \sum_{l=1}^{q} (e_k^{(l)} - \sum_{j \in C_i \cap N_k} \beta_{kj} e_j^{(l)})^2. \tag{3.13}$$

We compare the observed two-level convergence factors of iBAMG and diBAMG in Table 3.1. Although the convergence of diBAMG seems to require fewer vectors and relaxation sweeps to get the same convergence factor as iBAMG, it is not as efficient overall, especially for the matrices involving many unwanted connections because the setup cost for diBAMG depends more substantially on the number of these connections. The equivalence between (3.4) and (3.2) is due to preserving the same interpolatory set for each unwanted connection, so we can save on computational cost by collapsing all unwanted connections at once. It is not possible to collapse all unwanted connections at once in diBAMG.

q/ν	1	2	3	4	5	6
4	.57 (.25)	.38 (.13)	.34 (.10)	.34 (.06)	.26 (.06)	.27 (.06)
5	.35 (.19)	.27 (.07)	.24 (.06)	.18 (.06)	.17 (.06)	.11 (.06)

Table 3.1: Two-level convergence factors of iBAMG (diBAMG) with q vectors and ν relaxations for 2D Poisson with 9-point stencils and standard coarsening on a 64x64 grid.

3.4 *i*BAMG Theory

In this section, we develop a basic theory for BAMG and iBAMG convergence in an AMG reduction-based framework. The idea behind AMGr is to assume that we solved the F-point equations exactly in relaxation. (See the comment at the end of this section on the more practical case that the F-point equations are only approximately solved.) We show here that BAMG and iBAMG provide good exact coarse-grid correction in the AMGr context, with conditions that confirm when iBAMG is superior. Here, we only reach a certain point in this direction, with estimates that still need to be confirmed in practice.

We assume that the target matrix is symmetric and positive definite, has only 1s on the diagonal (otherwise, we simply diagonally scale it), and is represented in F-C block form as follows:

$$A = \begin{bmatrix} I - X & -Y \\ -Y^T & I - Z \end{bmatrix}.$$

Letting BAMG interpolation be expressed as

$$P_B = \begin{bmatrix} B \\ I \end{bmatrix}, \tag{3.14}$$

then a variant of iBAMG interpolation (that collapses F-F connections to C_j as opposed to C_i) is given by

$$P_I = \begin{bmatrix} XB + Y \\ I \end{bmatrix}. \tag{3.15}$$

'Ideal' interpolation is given by

$$P = \begin{bmatrix} (I - X)^{-1}Y \\ I \end{bmatrix}. \tag{3.16}$$

(We refer to P here as ideal because the fact that exact F-point relaxation produces error that is in the range of P means that an exact coarse-grid correction based on P produces an exact solver.) First notice that an exact F-point relaxation step is monotonically nonincreasing in energy because

it is just an energy-orthogonal projection of the error onto the range of P. Two-grid convergence based on P_B or P_I can thus be established simply by showing that coarse-grid correction reduces the error in the range of P. That is, with subscript A denoting the energy norm ($||x||_A = (x^T Ax)^{1/2}$), we can assume that the initial error satisfies

$$\begin{aligned}
\|Pe\|_A^2 =& < \begin{bmatrix} (I-X)^{-1}Y \\ I \end{bmatrix} e, A \begin{bmatrix} (I-X)^{-1}Y \\ I \end{bmatrix} e > \\
=& < \begin{bmatrix} (I-X)^{-1}Y \\ I \end{bmatrix} e, \begin{bmatrix} 0 \\ I - Z - Y^T(I-X)^{-1}Y \end{bmatrix} e > \\
=& < e, Se >,
\end{aligned}$$

where we have denoted the Schur complement by $S = (I - Z - Y^T(I-X)^{-1}Y)$.

Now, using P_B to do an exact coarse-grid correction on this error (that is, applying $\mathrm{CGC}_B = I - P_B(P_B^T A P_B)^{-1} P_B^T A$), we compute the squared-energy convergence factor by

$$\begin{aligned}
&\frac{\|\mathrm{CGC}_B Pe\|_A^2}{\|Pe\|_A^2} \\
=&\frac{< A\mathrm{CGC}_B Pe, \mathrm{CGC}_B Pe >}{< APe, Pe >} \\
=&\frac{< APe, Pe > - < AP_B(P_B^T AP_B)^{-1} P_B^T APe, Pe >}{< APe, Pe >} \\
=&1 - \frac{< (P_B^T AP_B)^{-1} Se, Se >}{< e, Se >}.
\end{aligned}$$

Then the worst case squared-energy convergence factor is the smallest $\epsilon_B \geq 0$ for which

$$(P_B^T AP_B)^{-1} \geq (1 - \epsilon_B) S^{-1}.$$

Note that $\epsilon_B \leq 1$ because $(P_B^T AP_B)^{-1}$ is SPD and $\epsilon_B \geq 0$ because $S^{-1} \geq (P_B^T AP_B)^{-1}$ ('ideal' interpolation minimizes $e^T \left(\begin{bmatrix} W \\ I \end{bmatrix}^T A \begin{bmatrix} W \\ I \end{bmatrix} \right)^{-1} e$ over W for any fixed e). We can rewrite this bound as

$$P_B^T AP_B \leq \frac{1}{1 - \epsilon_B} S. \tag{3.17}$$

The key now is to realize that the best P_B is the ideal one, so we write its F block as

$$B = (I - X)^{-1}Y + E_B. \tag{3.18}$$

Substituting (3.18) into (3.17), we obtain

$$E_B^T(I - X)E_B \leq \frac{\epsilon_B}{1 - \epsilon_B}S. \tag{3.19}$$

We can just assume that this holds for some $\epsilon_B < 1$ if all we want to do is compare iBAMG to BAMG. Following this line of thinking, we get a similar expression for P_I of course, except with B_I replacing B:

$$E_I^T(I - X)E_I \leq \frac{\epsilon_I}{1 - \epsilon_I}S,$$

where ϵ_I is the squared-energy convergence factor for iBAMG. It is interesting that E_B and E_I have a very simple relationship:

$$E_I = XE_B. \tag{3.20}$$

So, for iBAMG, a little algebra shows that we want the smallest ϵ_I for which

$$E_B^T X(I - X)XE_B \leq \frac{\epsilon_I}{1 - \epsilon_I}S. \tag{3.21}$$

Now, the left sides of (3.19) and (3.21) are related as follows:

$$E_B^T X(I - X)XE_B \leq \delta^2 E_B^T(I - X)E_B,$$

where δ bounds X in the sense that $X \leq \delta I$. Note that δ can be interpreted as the fraction of strength represented in C_i^c relative to C_i, so we can assume that δ is bounded by a constant that is less than 1 (e.g., $\delta \leq \frac{1}{2}$ for the 9-point Laplacian and standard coarsening). You can then conclude (after a little more algebra) that

$$\epsilon_I \leq (\frac{\delta^2}{1 - \epsilon_B + \delta^2\epsilon_B})\epsilon_B.$$

Note that the factor in front of ϵ_B here is always less than 1 by our assumption on δ. Thus, iBAMG always produces a better convergence factor than BAMG. More importantly, when BAMG

converges fairly well, that is, when ϵ_B is somewhat small, then this factor is approximately bounded by $\delta^2\epsilon_B$. Thus, the (*unsquared*) energy convergence factor for iBAMG in this case is roughly δ times that of BAMG.

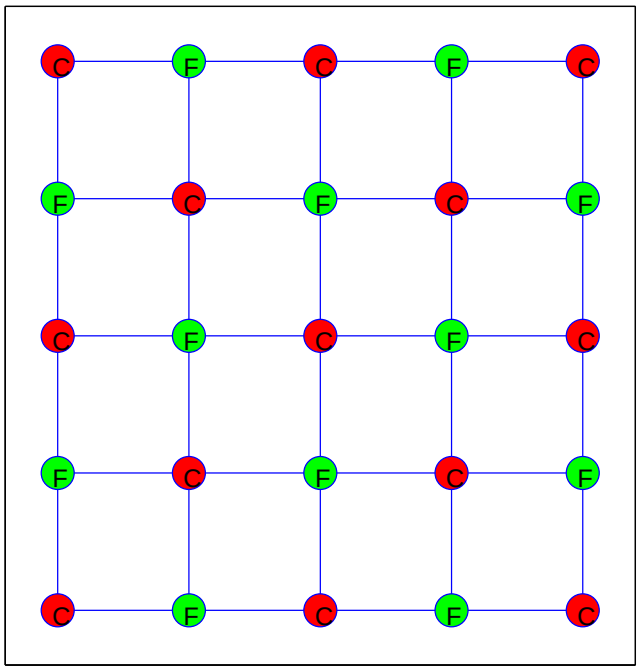

Figure 3.1: Red-black coarsening in case of an isotropic five-point stencil.

We want to emphasize that iBAMG, in contrast to classical BAMG, can provide reasonable interpolation even when the test vectors are poor. For example, consider the case, illustrated by Figure 3.1, of a five-point stencil and red-black coarsening. Since no F points have immediate F-point neighbors, then no test vectors are needed because there is no work for iBAMG to do. Said differently, this case is characterized by $X = 0$, and iBAMG therefore yields the ideal interpolation operator:

$$P_I = P = \begin{bmatrix} Y \\ I \end{bmatrix}.$$

This case is, of course, very special, but it emphasizes the point that iBAMG does not necessarily

need especially smooth test vectors, particularly when the F points are very strongly connected to the C points.

The theory here assumed exact F-point relaxation, which presumably requires exact inversion of $I - X$. It is more practical to assume that the F-point equations are only approximately solved. To study this case, we can write the current error in the energy-orthogonal decomposition form $Pe + Rz$, where $R = \begin{pmatrix} I \\ 0 \end{pmatrix}$. Now, relaxation on the F points is presumably local, so its effect on the error is to change only z. If Rz is large compared to Pe in energy, then relaxation alone can effectively reduce the error because $I - X$ is well conditioned (by assumption on δ). We can therefore assume that we apply a fixed but sufficient number of F-point relaxations to ensure that Rz is smaller in energy relative to Pe than any desired factor, η. We can thus conclude that the effect of a coarse-grid correction based on P_B is dominated by Pe:

$$\|\mathrm{CGC}_B(Pe + Rz)\|_A \leq \|\mathrm{CGC}_B Pe\|_A + \|\mathrm{CGC}_B Rz\|_A \leq (1 + \eta)\|\mathrm{CGC}_B Pe\|_A$$

and, similarly,

$$\|\mathrm{CGC}_B(Pe + Rz)\|_A \geq (1 - \eta)\|\mathrm{CGC}_B Pe\|_A.$$

Using this expression, together with an analogous result for the coarse-grid correction based on P_B, allows us to modify the theoretical bounds presented above by an arbitrarily small quantity, η. This, in turn, allows us to apply theory to the practical case that the F-point equations are only treated by simple relaxation, provided we assume a sufficient number of relaxation sweeps.

3.5 Weighted Iterative Interpolation

The variant of iBAMG determined by (3.15) is generally impractical: instead of collapsing the e_j to C_i as standard iBAMG does, they are collapsed to C_j, which generally leads to an enlarged coarse-grid stencil. Collapsing to C_i greatly complicates analysis because it depends on i: an e_j may be collapsed to certain C points for one i and the same e_j may be collapsed to different C points for another i. Nevertheless, the theory and form of this variant provide insight into the

nature of the standard iBAMG approach and possible methods for improvement. In this direction, note that we can still write the standard iBAMG interpolation process as a perturbation from the ideal:

$$P_I = \begin{bmatrix} W + Y \\ I \end{bmatrix}. \tag{3.22}$$

Note also that, using the equivalence between iBAMG and rBAMG, we can write

$$P_I = P_R = \begin{bmatrix} B - R \\ I \end{bmatrix}, \tag{3.23}$$

where R corresponds to the residual term in (3.11).

Note finally that the variant of iBAMG defined by (3.15) can be viewed as iterative interpolation using one step of Jacobi applied to B as the initial guess for the linear system

$$(I - X)Q = Y,$$

whose solution, $Q = (I-X)^{-1}Y$, gives ideal interpolation. The point here is that this suggests that weighting the Jacobi step may result in improvement to standard iBAMG, and that this translates to a simple weighting of the relaxation-corrected term, R, in (3.23) as the following suggests:

$$\begin{aligned} Q &\leftarrow \omega(XB + Y) + (1 - \omega)B \\ &\approx \omega(W + Y) + (1 - \omega)B \\ &= \omega(B - R) + (1 - \omega)B \\ &= B - \omega R. \end{aligned} \tag{3.24}$$

Note that W in the second line of (3.24) is used in real iBAMG interpolation (that collapses F-F connections to C_i) and the third line comes from the equivalence between iBAMG and rBAMG. The last line in (3.24) can be achieved to define a weighted rBAMG (wrBAMG) as follows:

$$(wr\text{BAMG}) \qquad \{w_{ij} : j \in C_i\} = \arg\min_{w_{ij}} \sum_{l=1}^{q} (e_i^{(l)} - \sum_{j \in C_i} w_{ij} e_j^{(l)} - \omega \frac{r_i^{(l)}}{a_{ii}})^2. \tag{3.25}$$

The question arises as to what a good choice would be for the relaxation parameter, ω. The ultimate goal is to produce a multigrid solver with a fairly small and possibly optimal convergence

factor. In the AMGr setting that we assume here, optimality translates to determining a suitable Z for $P_Z = \begin{pmatrix} Z \\ I \end{pmatrix}$ to minimize

$$\max_{x \neq 0} \frac{\|[I - P_Z(P_Z^T A P_Z)^{-1} P_Z^T A]Px\|_A^2}{\|Px\|_A^2}.$$

Now, since $I - P_Z(P_Z^T A P_Z)^{-1} P_Z^T A$ is an orthogonal projection in the energy inner product, we can rewrite this objective as one of minimizing the largest eigenvalue of the generalized eigenproblem

$$[P^T AP - P^T AP_Z(P_Z^T AP_Z)^{-1} P_Z B^T AP]x = \mu P^T APx.$$

Since

$$AP = \begin{bmatrix} 0 \\ S \end{bmatrix},$$

then this eigenproblem becomes

$$[S - S(P_Z^T AP_Z)^{-1} S]x = \mu Sx.$$

Multiplying both sides of this eigenproblem by S^{-1} and rearranging yields

$$(P_Z^T AP_Z)^{-1} x = (1 - \mu) S^{-1} x.$$

Choosing Z to minimize the largest μ for this problem is equivalent to choosing $\gamma = \frac{1}{1-\mu}$ to minimize the largest eigenvalue of

$$P_Z^T AP_Z x = \gamma Sx.$$

Noting that

$$P_Z = P + \begin{bmatrix} E_Z \\ 0 \end{bmatrix},$$

which implies that

$$P_Z^T AP_Z = S + E_Z^T (I - X) E_Z,$$

we can then conclude that our ultimate goal is to minimize the largest eigenvalue, $\lambda = \gamma - 1$, of

$$E_Z^T (I - X) E_Z x = \lambda Sx.$$

Now, it is easy to see that the iterative interpolation step that characterizes iBAMG can be written in terms of the error, E_Z, as

$$E_Z \leftarrow (I - \omega(I - X))E_Z.$$

Putting these last two expressions together shows that our aim is to choose ω to minimize the largest eigenvalue of $S^{-1/2}E_Z^T(I - \omega(I - X))^2(I - X)E_Z S^{-1/2}$. Now, we have no control over E_Z, so we have no control over $E_Z S^{-1/2}$ either. This reasoning suggests that our ultimate goal is to choose ω to minimize the largest eigenvalue of $(I - \omega(I - X))^2(I - X)$, that is,

$$\omega = \omega_{\text{opt}} = \frac{1}{2 - \lambda_{\min}(X) - \lambda_{\max}(X) - \sqrt{(1 - \lambda_{\min}(X))(1 - \lambda_{\max}(X))}}.$$

We could estimate this optimal choice in general by using a shifted power method to determine the extreme eigenvalues of X. However, it is important to note that this analysis is for the generally impractical variant of iBAMG, so it is unclear if this kind of reasoning holds for the practical version. It is, nevertheless, interesting to note that some model problems yield an X with equal magnitude but oppositely signed extreme eigenvalues, say, $\pm\gamma$, which would lead us to choose $\omega = 1/(2 - \sqrt{1 - \gamma^2}) \leq 1$. It is also interesting that our numerical experience for the model problem suggested that $\omega \leq 1$ is not as effective as other choices.

Specifically, Tables 3.2 through 3.5 represent observed two-level and five-level convergence factors of standard iBAMG and wrBAMG using a weight of $\omega = 1.3$, which was observed in several separate tests to yield the best overall performance. (Each result reported in these tables actually represents an average of ten runs made for the particular l, q, ν, and N, starting from a different set of random vectors. We proceeded in this way here and in all of the tests reported in this thesis to avoid any anomalies with particular runs resulting from unlikely special characteristics of the initial random vectors.) For a small number of vectors (e.g., q = 4 and 5), wrBAMG results seem more promising than iBAMG except for the first column, which is generally not of much interest because of the typically poor performance achieved by a single relaxation of the test vectors. Nevertheless, it is not clear how best to choose the weight, ω, in (3.25), and more study of this approach is left for future research.

The convergence factor tables shown here and in subsequent chapters are meant to show the trends in performance of the adaptive/bootstrap methods in terms of the number of initial test vectors and the amount that they are relaxed. The general trend is naturally dependent on N, with more work required to achieve the same performance as N increases. These tables also clearly show the dangers of too little work and the diminishing returns of too much work. Also evident in these tables are the limiting convergence factors themselves, that is, the best factors that one can expect if work is not taken into account. These test results may thus provide a guide for those applications that are expected to require many solutions of linear equations with the same matrix. On the other extreme, we would assume that the resulting solver would be used to solve just one linear equation. In this case, taking work into account becomes crucial. We begin reporting on results of a systematic study of this issue in Section 5.2.

q/ν	1	2	3	4	5	6	7	8	9	10
4	.55	.42	.35	.33	.30	.28	.30	.27	.28	.29
	(.58)	(.40)	(.28)	(.24)	(.23)	(.18)	(.18)	(.17)	(.18)	(.18)
5	.38	.27	.21	.18	.17	.17	.15	.14	.12	.12
	(.43)	(.23)	(.18)	(.10)	(.09)	(.08)	(.10)	(.09)	(.10)	(.11)
6	.28	.18	.11	.10	.08	.09	.07	.07	.08	.08
	(.35)	(.16)	(.11)	(.09)	(.07)	(.08)	(.08)	(.09)	(.08)	(.09)
7	.23	.13	.09	.07	.07	.07	.07	.07	.07	.07
	(.30)	(.12)	(.09)	(.07)	(.07)	(.08)	(.08)	(.07)	(.08)	(.08)
8	.19	.11	.07	.07	.07	.07	.07	.07	.07	.07
	(.27)	(.11)	(.07)	(.07)	(.07)	(.07)	(.07)	(.07)	(.07)	(.08)
9	.17	.08	.06	.06	.06	.06	.06	.07	.07	.07
	(.24)	(.10)	(.07)	(.07)	(.07)	(.07)	(.07)	(.07)	(.07)	(.07)
10	.15	.08	.06	.06	.06	.06	.06	.06	.07	.06
	(.22)	(.08)	(.07)	(.07)	(.07)	(.07)	(.07)	(.07)	(.07)	(.07)

Table 3.2: Observed two-level convergence factors of standard iBAMG (wrBAMG with $\omega = 1.3$) applied to 2D Poisson with 9-point stencils on a 64x64 grid.

q/ν	1	2	3	4	5	6	7	8	9	10
4	.95	.91	.91	.91	.90	.89	.89	.88	.89	.88
	(.94)	(.89)	(.81)	(.57)	(.57)	(.51)	(.53)	(.56)	(.45)	(.50)
5	.67	.45	.31	.32	.33	.37	.30	.31	.30	.32
	(.68)	(.35)	(.21)	(.16)	(.14)	(.15)	(.12)	(.13)	(.13)	(.13)
6	.42	.23	.19	.14	.14	.12	.12	.11	.12	.17
	(.50)	(.24)	(.15)	(.13)	(.11)	(.11)	(.10)	(.11)	(.12)	(.11)
7	.32	.16	.12	.10	.11	.10	.12	.10	.10	.10
	(.41)	(.18)	(.13)	(.11)	(.11)	(.11)	(.11)	(.10)	(.11)	(.11)
8	.27	.15	.11	.10	.10	.10	.10	.10	.10	.10
	(.36)	(.17)	(.12)	(.11)	(.10)	(.10)	(.10)	(.10)	(.10)	(.11)
9	.24	.12	.10	.10	.10	.10	.10	.10	.10	.10
	(.33)	(.14)	(.11)	(.10)	(.10)	(.10)	(.10)	(.10)	(.10)	(.10)
10	.22	.11	.10	.10	.10	.10	.10	.10	.10	.10
	(.30)	(.13)	(.11)	(.10)	(.10)	(.10)	(.10)	(.10)	(.10)	(.10)

Table 3.3: Observed five-level convergence factors of standard iBAMG (wrBAMG with $\omega = 1.3$) applied to 2D Poisson with 9-point stencils on a 64x64 grid.

q/ν	1	2	3	4	5	6	7	8	9	10
4	.71	.47	.42	.37	.35	.36	.37	.35	.36	.34
	(.77)	(.43)	(.33)	(.28)	(.26)	(.23)	(.22)	(.22)	(.22)	(.23)
5	.58	.29	.26	.24	.22	.21	.20	.19	.16	.17
	(.68)	(.26)	(.18)	(.13)	(.13)	(.13)	(.12)	(.11)	(.11)	(.13)
6	.49	.21	.17	.14	.13	.10	.12	.13	.08	.10
	(.61)	(.18)	(.13)	(.10)	(.10)	(.09)	(.08)	(.09)	(.09)	(.09)
7	.44	.16	.11	.09	.08	.07	.07	.08	.08	.07
	(.57)	(.16)	(.12)	(.08)	(.09)	(.07)	(.08)	(.08)	(.09)	(.07)
8	.39	.13	.09	.07	.07	.07	.07	.07	.07	.07
	(.52)	(.13)	(.09)	(.07)	(.07)	(.07)	(.08)	(.07)	(.07)	(.07)
9	.36	.10	.07	.07	.07	.07	.07	.07	.07	.07
	(.50)	(.12)	(.08)	(.07)	(.07)	(.07)	(.07)	(.07)	(.07)	(.07)
10	.33	.09	.07	.07	.07	.07	.07	.07	.07	.07
	(.46)	(.10)	(.07)	(.07)	(.07)	(.07)	(.07)	(.07)	(.07)	(.07)

Table 3.4: Observed two-level convergence factors of standard iBAMG (wrBAMG with $\omega = 1.3$) applied to 2D Poisson with 9-point stencils on a 128x128 grid.

q/ν	1	2	3	4	5	6	7	8	9	10
4	.98	.97	.97	.97	.97	.97	.97	.97	.97	.97
	(.98)	(.96)	(.95)	(.92)	(.90)	(.89)	(.88)	(.88)	(.88)	(.85)
5	.87	.71	.65	.64	.60	.68	.62	.55	.60	.63
	(.88)	(.55)	(.33)	(.21)	(.21)	(.14)	(.17)	(.14)	(.16)	(.13)
6	.60	.32	.28	.19	.19	.20	.17	.15	.18	.15
	(.70)	(.28)	(.22)	(.15)	(.12)	(.16)	(.11)	(.12)	(.11)	(.11)
7	.52	.20	.15	.12	.11	.11	.11	.13	.10	.10
	(.63)	(.23)	(.15)	(.11)	(.11)	(.10)	(.10)	(.10)	(.11)	(.10)
8	.46	.17	.11	.11	.10	.10	.10	.10	.10	.10
	(.59)	(.20)	(.12)	(.10)	(.10)	(.10)	(.10)	(.10)	(.10)	(.10)
9	.42	.15	.11	.10	.10	.10	.10	.10	.10	.10
	(.56)	(.18)	(.11)	(.10)	(.10)	(.10)	(.10)	(.10)	(.10)	(.10)
10	.39	.12	.10	.10	.10	.10	.10	.10	.10	.10
	(.53)	(.17)	(.11)	(.10)	(.10)	(.10)	(.10)	(.10)	(.10)	(.10)

Table 3.5: Observed six-level convergence factors of standard iBAMG (wrBAMG with $\omega = 1.3$) applied to 2D Poisson with 9-point stencils on a 128x128 grid.

Chapter 4

Fully Adaptive AMG Process

As we said, the adaptive or bootstrap AMG methodology allows for many options, including where in the coarsening process the current method is tested and how it is improved, how the original test vectors and emerging error components are combined and used to improve the current solver, what the schedule should be for visiting the coarse grids in the current solver, and many other possible aspects of the approach. The focus here is on choosing the number of initial test vectors and the number of initial relaxation sweeps on them, and how optimal choices here depend on the problem and its size. We have therefore fixed the design of the other aspects of the methodology based on our experience during several tests of the various options. The most significant choice here is to assume that the coarse grid has already been determined. In fact, our numerical study assumes a uniform fine grid in two dimensions that is coarsened in the standard way by eliminating every other line of points in each coordinate direction. We also adopt a nonrecursive adaptive process that is described in detail in the next section. Another important choice we make is to apply a Ritz process to the set that includes the initial relaxed test vectors and the evolving error components produced by the current solver. As explained in the next subsection, we also fix the relaxation process in both the setup and solver phases to be pointwise lexicographic Gauss-Seidel, with the current solver defined as four *indivisible* V(1,1) cycles. Other choices that we make in our approach are revealed later in the process of describing the basic methodology.

The purpose of the Ritz process is to sort out the various levels of smoothness in the subspace of target vectors. This is needed because fitting interpolation must pay more attention to smooth

vectors than it does to oscillatory ones. This is articulated in the so-called weak approximation property that provides the basic principle we use to determine the proper weights used in the least-squares process. See [14] for a discussion of this principle and how it is used in the context of adaptive smoothed aggregation.

For the initial set of randomly generated test vectors, sorting is generally unnecessary because random unit vectors tend to have the same quality of smoothness. However, as we add error components to the set of target vectors produced by the current solver, scaling becomes crucial. We thus use Ritz here to properly sort out vectors based on their smoothness property and scale the results so that they are of unit length in the energy inner product. The net effect of this scaling is that the subsequent least-square fit of interpolation will pay most attention to those target vectors with small Rayleigh quotients, as the weak approximation property dictates.

We should note here that some of the vectors produced by the Ritz process might not exhibit enough algebraic smoothness to be of use in the least-squares process. That is, scaling by their energy might produce a very small vector in the Euclidean norm sense. It would thus be possible, without ill effects, to reduce complexity by discarding these vectors from the target set. We have not done any filtering in the tests reported here, but plan to incorporate such a feature in future work.

4.1 The Adaptive MG Algorithm

The importance of two complementary multigrid principles cannot be stressed enough. The development of an efficient multigrid method depends on these two ingredients. The main idea behind the adaptive process is to improve interpolation based on slow-to-converge components. Unlike standard adaptive AMG that typically begins with one test vector, we choose here to begin by relaxing ν times on a number of random test vectors as initial guesses for solving the homogeneous problem, $Ax = 0$. We then compute an interpolation operator, P, based on a least-squares fit of the target vectors. The coarse-grid operator is then formed by the usual Galerkin approach, $P^T AP$. On the coarse grid, the pre-images of the fine-grid test vectors under interpolation P (which, for

AMG, is just their restrictions to the C points) are used as an initial set of test vectors. These q vectors are in turn relaxed ν times (in terms of the homogeneous Galerkin coarse-grid problem) and used in a similar least-squares process to define interpolation to a yet coarser grid. This process continues to the coarsest level (determined in advance to contain just a few points), where no further processes or test vectors are needed in the setup because coarsening is not needed there. This completes the setup phase. Note here that we fix the number of test-vector iterations over all levels.

Once an initial MG hierarchy has been computed, we test the current method by running several $V(1,1)$ cycles. In the next section, we introduce a convergence model that is used to estimate the asymptotic convergence factors of these cycles, and we describe there how these estimates are used to determine the progress of the adaptive process. This model also includes a formula for estimating the cost of the overall solution process. These cost estimates, described in Section 4.3, are used in the setup process to monitor current and projected future costs, which allows us to determine whether expected improvements in the convergence factors we estimate are worth continuing the adaptive cycles. The overall flow of our approach is shown in Algorithm 4.1. Note that it assumes two input parameters, ρ_{good} and ρ_{bad}, that guide our judgement as to whether the estimated convergence factor, ρ_{est}, of the current solver is acceptable ($\rho_{est} \leq \rho_{good}$), unacceptable ($\rho_{bad} < \rho_{est}$), or indeterminate ($\rho_{good} < \rho_{est} < \rho_{bad}$). In the first case, we terminate the process; in the second, we continue; and, in the third, we use the work estimate to decide whether to terminate or continue. This algorithm also assumes three input cycling parameters, α, ν_1, and ν_2, that determine the form of the current solver. Specifically, we define the current solver to consist of indivisible α $V(\nu_1,\nu_2)$ cycles, meaning that we are testing the overall convergence factor for α applications of a V-cycle that use ν_1 relaxation on the descent through coarser levels and ν_2 relaxations on the ascent back to the fine grid. Our experience with several numerical tests of other options on the problems we study below suggest that $\alpha = 4$ and $\nu_1 = \nu_2 = 1$ are reasonable choices to make in terms of overall efficiency of the setup and solver process. These choices are what we use in all of our experiments documented in this thesis.

Algorithm 4.1 Non-recursive adaptive AMG algorithm

for $j = 0$ to *maxAdapt* **do**
  for $k = 1$ to *coarsest* - 1 **do**
    if $k == 1$ **then**
      if $j == 0$ **then**
        Pick the set of q random vectors, $\{x_{(1)}^{(1)}, ..., x_{(q)}^{(1)}\}$
      end if
    else
      Let $x_{(i)}^{(k)} = (x_{(i)}^{(k-1)})_c$, $i = 1, ..., q + j$.
    end if
    if $j > 0$ **then**
      $\{x_{(1)}^{(k)}, ..., x_{(q+j)}^{(k)}\}$ =RITZ($\{x_{(1)}^{(k)}, ..., x_{(q+j)}^{(k)}\}$)
    end if
    if $j == 0$ OR ($j > 0$ AND $k > 1$) **then**
      Relax on $A^{(k)}x_{(i)}^{(k)} = 0$ ν times, $i = 1, ..., q + j$.
    end if
    Compute interpolation, I_{k+1}^k.
    Compute the coarse-grid operator, $A^{(k+1)} = (I_{k+1}^k)^T A^{(k)} I_{k+1}^k$.
  end for
  Pick random initial $x^{(1)}$.
  Apply α V(ν_1,ν_2) cycles to the homogeneous fine-grid problem, $A^{(1)}x^{(1)} = 0$.
  Estimate the convergence factor, ρ_{est}.
  Compute *ncycle* such that $\rho_{est}^{ncycle} < tol$.
  Compute the total cost, $W_j^{total} = W_j^{setup} + W_j^{cycle}(ncycle)$.
  if $\rho_{est} \leq \rho_{good}$ **then**
    stop.
  else if $\rho_{est} > \rho_{bad}$ **then**
    $x_{q+j+1}^{(1)} \longleftarrow x^{(1)}$.
    continue.
  else
    if $W_j^{total} > W_{j-1}^{total}$ **then**
      stop
    else
      $x_{q+j+1}^{(1)} \longleftarrow x^{(1)}$.
      continue.
    end if
  end if
end for

4.2 Convergence Estimation

Access to a simple and efficient stopping criterion is an essential ingredient of our adaptive methodology, the core component of which is a reliable estimate the quality of the current solver. We could obtain such an estimate simply by applying the current method enough times to the homogeneous equation to ensure that a reliable measure of the worst-case convergence factor can be observed. Unfortunately, in practice, this naive approach usually requires so many cycles that it has a deleterious effect on overall complexity of the setup process. Our approach instead is to use a simple extrapolation scheme that is based on just a few observed cycles of the current solver. Our aim, then, is to develop a very simple model of convergence of the solver that involves just a few parameters that can be determined by very few observed convergence factors, typically three or four in our case.

Specifically, the convergence model we develop here is based on the assumptions that each error is dominated by two orthonormal components, e_1 and e_2, and that these are eigenvectors of the cycle's error propagation matrix with eigenvalues β_1 and β_2, respectively ($\beta_1 > \beta_2$). Suppose now that the error after k cycles, $e^{(k)}$, is a linear combination of these two orthogonal components with coefficients α_1 and α_2, respectively. Then, since there are four unknowns, α_1, α_2, β_1, and β_2, we need at least four consecutive errors to determine them:

$$\begin{aligned} e^{(k)} &= \alpha_1 e_1 + \alpha_2 e_2, \\ e^{(k+1)} &= \alpha_1 \beta_1 e_1 + \alpha_2 \beta_2 e_2, \\ e^{(k+2)} &= \alpha_1 \beta_1^2 e_1 + \alpha_2 \beta_2^2 e_2, \\ e^{(k+3)} &= \alpha_1 \beta_1^3 e_1 + \alpha_2 \beta_2^3 e_2. \end{aligned} \tag{4.1}$$

We first compute the squared Euclidean norm of these four consecutive errors. Using orthogonality

of the two components, one gets

$$\begin{aligned} ||e^{(k)}||^2 = &\quad \alpha_1^2 + \alpha_2^2, \\ ||e^{(k+1)}||^2 = &\quad \alpha_1^2\beta_1^2 + \alpha_2\beta_2^2, \\ ||e^{(k+2)}||^2 = &\quad \alpha_1^2\beta_1^4 + \alpha_2\beta_2^4, \\ ||e^{(k+3)}||^2 = &\quad \alpha_1^2\beta_1^6 + \alpha_2\beta_2^6. \end{aligned} \tag{4.2}$$

For the sake of convenience, define $C_i = ||e^{(k+i)}||^2$, $i = 0, ..., 3$, and $a_j = \alpha_j^2$ and $b_j = \beta_j^2$, j = 1,2. Then

$$\begin{aligned} C_0 &= a_1 + a_2, \\ C_1 &= a_1b_1 + a_2b_2, \\ C_2 &= a_1b_1^2 + a_2b_2^2, \\ C_3 &= a_1b_1^3 + a_2b_2^3. \end{aligned} \tag{4.3}$$

Since β_1 is of interest in the estimation and prediction model, we eliminate unknowns a_1 and a_2 in turn. Multiplying the first equation by -b_1 and adding the result to the second equation, and similarly for the third and fourth equation, we obtain

$$\begin{aligned} C_1 - C_0b_1 &= a_2(b_2 - b_1), \\ C_2 - C_0b_1^2 &= a_2(b_2^2 - b_1^2), \\ C_3 - C_0b_1^3 &= a_2(b_2^3 - b_1^3). \end{aligned} \tag{4.4}$$

Similarly, we eliminate a_2, arriving at

$$\begin{aligned} C_0b_1b_2 - C_1(b_1 + b_2) &= -C_2, \\ C_1b_1b_2 - C_2(b_1 + b_2) &= -C_3, \end{aligned} \tag{4.5}$$

Thus, b_1 is the larger root of the following equation:

$$x^2 - \gamma x + \delta = 0, \tag{4.6}$$

where δ and γ solve

$$\begin{bmatrix} C_0 & -C_1 \\ C_1 & -C_2 \end{bmatrix} \begin{bmatrix} \delta \\ \gamma \end{bmatrix} = - \begin{bmatrix} C_2 \\ C_3 \end{bmatrix}.$$

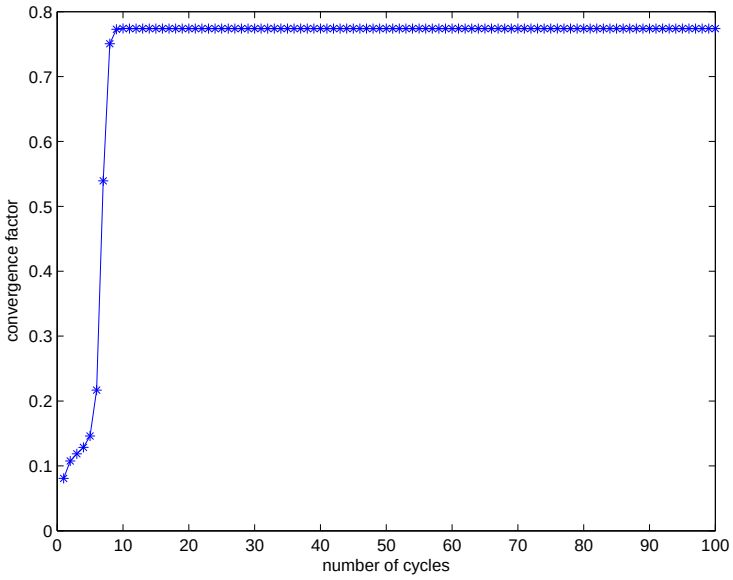

Figure 4.1: Convergence factors for V(1,1) cycles using rBAMG with $q = \nu = 10$.

4.2.1 Numerical Illustrations for Convergence Estimation

In this subsection, we test the performance of our convergence estimation model. We do this by first applying just the rBAMG setup process using two different extreme values for the number of test vectors, q, and number of relaxation sweeps, ν, applied to them, namely, $q = \nu = 1$ and $q = \nu = 10$. Our target problem here is the shifted 5-point finite difference discretization of the gauge Laplacian (c.f., [24]) in two dimensions on a uniform doubly periodic grid with red-black coarse grids. After discretizing the equation (scaled so that $\frac{4}{h^2}$ is on the diagonal), the smallest eigenvalue of the operator is shifted to be $\frac{1}{N^2} = \frac{1}{64^2}$. We first estimate the asymptotic convergence factors of the resulting $V(1,1)$ cycles by running 100 cycles applied to the homogeneous problem and observing the convergence factor for each cycle. See Figures 4.1 and 4.2. Note that these figures also clearly illustrate that it takes many iterations to obtain a reliable estimate of the convergence

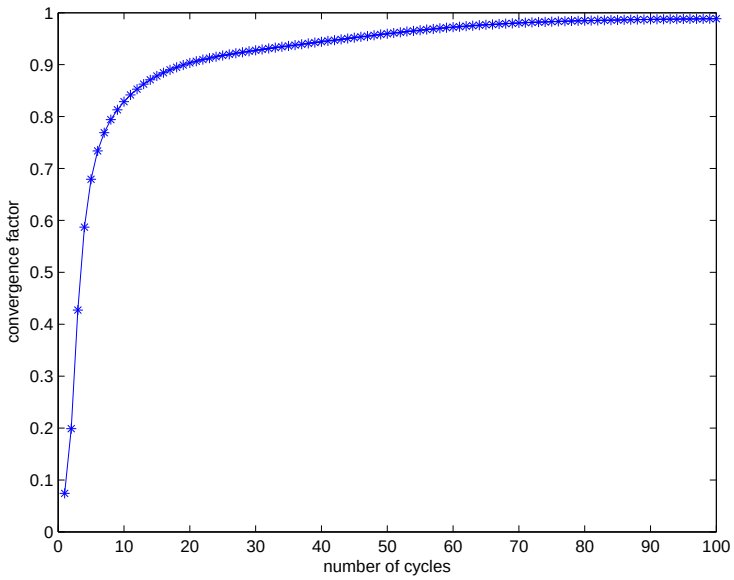

Figure 4.2: Convergence factors for V(1,1) cycles using rBAMG with $q = \nu = 1$.

factor. Tables 4.1 and 4.2 compare the asymptotic convergence factors observed in Figure 4.1 and 4.2, respectively, to the estimations obtained from our extrapolation process. Note that, as Table 4.1 shows, our predictions based on just a few cycles are quite accurate even for the somewhat slowly converging process shown in Figure 4.1. This quality of the estimation degrades in the face of poor solver convergence, as Table 4.2 shows. However, the ability of our estimation to signal good convergence is the key here and the approach is very effective in that regard.

Asymptotic Convergence Factor	k = 0	k = 1	k = 2
0.77	0.76	0.78	0.78

Table 4.1: Asymptotic convergence factor for V(1,1) cycles and its approximations using rBAMG with q= ν=10).

Asymptotic Convergence Factor	k = 0	k = 1	k = 2
0.99	0.77	0.84	0.88

Table 4.2: Asymptotic convergence factor for V(1,1) cycles and its approximations using rBAMG with q= ν=1).

To carry out these tests further, we proceeded in the same way but incorporated a single adaptive cycle in the setup after the least-squares fit of interpolation. We thus applied 4 $V(1,1)$ cycles to a random initial guess for $Ax = 0$ and combined the result with the initial test vectors in a Ritz process. We observe similar behavior of the ability of the extrapolation process to estimate the convergence factor fairly quickly, especially in the important case of reasonably good convergence of the current solver.

Our experience here and in other tests suggest that our extrapolation process reduces the need for cycles by typically a factor of two or more.

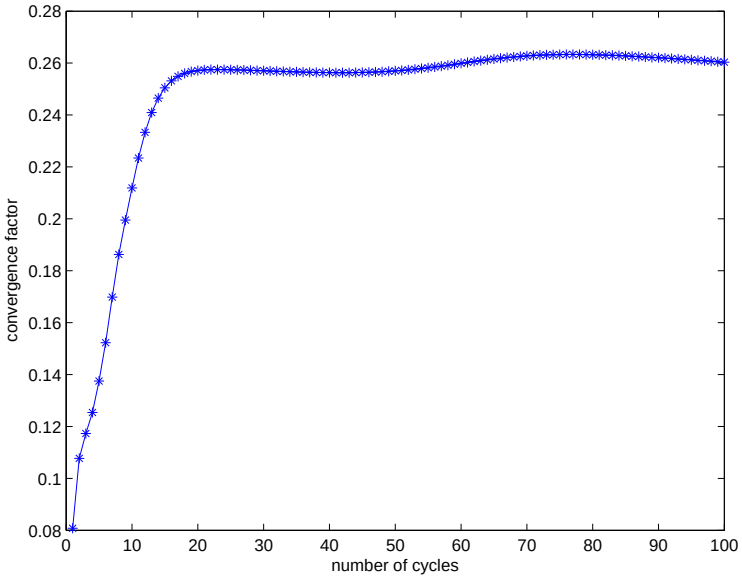

Figure 4.3: Convergence factors for V(1,1) cycles using rBAMG with $q = \nu = 10$ incorporated a single adaptive cycle.

Asymptotic Convergence Factor	k = 0	k = 1	k = 2
0.26	0.22	0.22	0.26

Table 4.3: Asymptotic convergence factor for V(1,1) cycles and its approximations using rBAMG with q= ν=10) incorporated a single adaptive cycle.

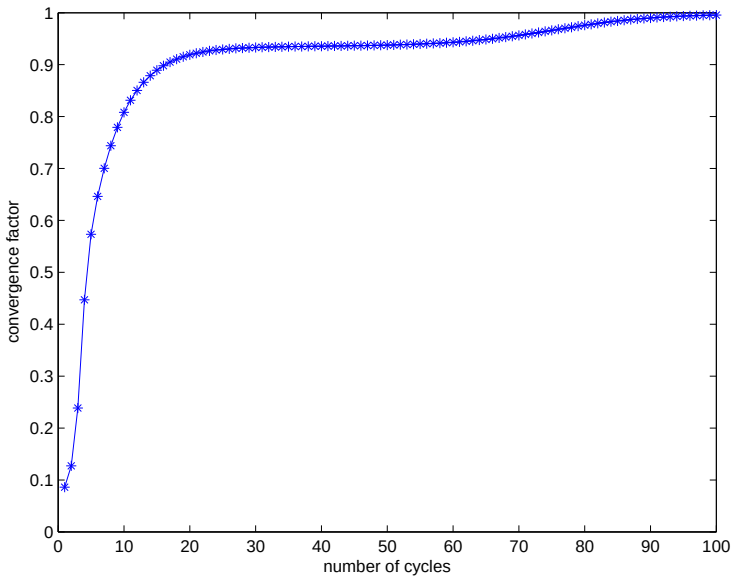

Figure 4.4: Convergence factors for V(1,1) cycles using rBAMG with $q = \nu = 1$ incorporated a single adaptive cycle.

Asymptotic Convergence Factor	k = 0	k = 1	k = 2
0.99	0.76	0.84	0.90

Table 4.4: Asymptotic convergence factor for V(1,1) cycles and its approximations using rBAMG with q= ν=1) incorporated a single adaptive cycle.

4.3 Setup Cost Considerations

In this subsection, we consider the setup cost of the adaptive phase of rBAMG. Assuming as we do a uniform mesh in two dimensions with a 9-point stencil and standard coarsening, then we can bound grid and matrix complexity based on the estimate that the number of grid points is reduced by a factor of approximately $\frac{1}{4}$ in each coarsening. Thus, the cost of an operation performed on all grid levels can be approximately bounded by $\frac{4}{3}$ of the cost of the corresponding fine-grid operation. The dominant operation is a matrix-vector product, which done once on all levels therefore costs about $\frac{4}{3}$ of a fine-grid *work unit* (the cost of a residual evaluation or matrix-vector product on the finest grid). We define the following variables and functions used to describe the setup process:

ν the number of relaxations applied to each test vector on every grid but the coarsest;

q the number of initial test vectors;

M the number of adaptive setup cycles;

ν_1 the number in relaxations on the descent portion of each setup cycle;

ν_2 the number in relaxations on the ascent portion of each setup cycle;

α the number of V cycles in each indivisible adaptive cycle;

INTERP(q,N) the cost of forming the interpolation operator on the finest-grid.

P(N) the cost of interpolating a vector to the finest grid;

CGO(N) the cost of forming the Galerkin coarse-grid operator, $P^T AP$; and

RITZ(q,N) the cost of performing the Ritz projection.

We estimate cost of individual setup processes in terms of fine-grid work units as follows:

(1) initial least-squares fit $\frac{4}{3}(q \times \nu + \text{INTERP}(q, N) + \text{CGO}(N))$;

(2) V cycle $\frac{4}{3}(\nu_1 + \nu_2 + \text{P}(N))$;

(3) j^{th} Ritz process $\frac{4}{3}$RITZ($q + j$,N); and

(4) j^{th} additional least-squares fit $\frac{4}{3}(\frac{1}{4}(q + j) \times \nu + \text{INTERP}(q + j, N) + \text{CGO}(N))$.

This gives the following form for the total cost of the setup phase:

$$\begin{aligned}
\text{W}_M^{setup} &= \frac{4}{3}(q \cdot \nu + \text{INTERP}(q, N) + \text{CGO}(N)) \\
&+ \sum_{j=1}^{M}(\alpha\frac{4}{3}(\nu_1 + \nu_2 + \text{P}(n)) + \frac{4}{3}\text{RITZ}(q + j, N) \\
&+ \frac{4}{3}(\frac{1}{4}(q + j)\nu + \text{INTERP}(q + j, N) + \text{CGO}(N)) \\
&= \frac{1}{3}((M + 3)q + \frac{M(M - 1)}{2})\nu + \frac{4}{3}M \cdot \text{CGO}(N) + \alpha(M - 1)(\nu_1 + \nu_2 + \text{P}(N)) \\
&+ \frac{4}{3}\sum_{j=0}^{M}\text{INTERP}(q + j, N) + \frac{4}{3}\sum_{j=1}^{M-1}\text{RITZ}(q + j, N).
\end{aligned}$$

To be more specific here, we analyze each general term in this expression in as follows.

4.3.1 P(N)

Because of the assumed grid coarsening factor, this cost is about $\frac{1}{4}$ of a work unit.

4.3.2 INTERP(q,N)

For each fine-grid point i, we solve the least-squares problem specified in (3.11). We rewrite this in matrix form as follows:

$$\begin{pmatrix} e_{j_1}^{(1)} & \dots & e_{j_{|C_i|}}^{(1)} \\ \vdots & \ddots & \vdots \\ e_{j_1}^{(q)} & \dots & e_{j_{|C_i|}}^{(q)} \end{pmatrix}_{q\times|C_i|} \begin{pmatrix} w_{kj_1} \\ \vdots \\ w_{kj_{|C_i|}} \end{pmatrix}_{|C_i|\times 1} = \begin{pmatrix} e_i^{(1)} - \frac{r_i^{(1)}}{a_{ii}} \\ \vdots \\ e_i^{(q)} - \frac{r_i^{(q)}}{a_{ii}} \end{pmatrix}_{q\times 1}, \tag{4.7}$$

where $r_i^{(l)} = (Ae^{(l)})_i$, $l = 1, ..., q$. Computing the right side of (4.7) costs $2q$ flops (requiring a negation and a division for each right side). Then we solve this equation via the normal equations, which requires multiplying the matrix in (4.7) and the right side vector by the matrix transpose, computing a Cholesky decomposition, and performing backward and forward substitutions. Forming the normal equation requires $(2q-1)|C_i|^2$ flops for the matrix-matrix multiplication and $(2q-1)|C_i|$ flops for the matrix-vector multiplication. Then Cholesky decomposition requires approximately $\frac{1}{3}|C_i|^3$ flops and each substitution costs $|C_i|^2$ flops. Thus, the total cost in flops of defining P is given by

$$\sum_{i\in F}(\frac{1}{3}|C_i|^3 + (2q+1)|C_i|^2 + (2q-1)|C_i|^2 + 2q). \tag{4.8}$$

There are now three types of fine-grid points in our fully coarsened grid. One-fourth of the grid is composed of points that lie at the center of a coarse-grid cell, having 4 coarse-grid neighbors and 4 fine-grid neighbors. One-fourth lie at midpoints of horizontal coarse-grid lines, and one-fourth lie at midpoints of vertical coarse-grid lines, both having 2 coarse-grid neighbors and 6 fine-grid neighbors. The final fourth of the fine-grid nodes are also coarse-grid nodes. Taking all of this into

account into the expression in (4.8) allows us to estimate it as follow:

$$\begin{aligned}
&\frac{N}{4}(\frac{1}{3}4^3 + (2q+1)4^2 + (2q-1)4 + 2q) \\
&+ \frac{2N}{4}(\frac{1}{3}2^3 + (2q+1)2^2 + (2q-1)2 + 2q) \\
&= \frac{N}{6}(105q + 64) \text{ flops.}
\end{aligned}$$

Now, one work unit equals the cost of a residual evaluation. For our 9-point operator, this takes 18N flops (one addition and one multiplication per non-zero in the matrix). Note that rBAMG requires q work units to compute q residual vectors. Thus, we find that forming rBAMG interpolation costs $\frac{105q+64}{6\times 18} + q \approx 2q + 0.6$ work units.

4.3.3 CGO(N)

The product of an $N \times N$ matrix A with an $N \times N_c$ matrix P, where N_c is the size of the first coarse grid, is an $N \times N_c$ matrix whose entries are

$$(AP)_{i,j} = \sum_{k=1}^{N} A_{ik} P_{kj},$$

where $1 \leq i \leq N$ is the row index and $1 \leq j \leq N_c$ is the column index.

The triple product of $N_c \times N$ matrix R with AP is $N_c \times N_c$ matrix RAP, whose entries are

$$(R(AP))_{i,j} = \sum_{k=1}^{N_c} R_{ik} (\sum_{l=1}^{N} A_{kl} P_{lj}),$$

where $1 \leq i \leq N_c$ is the row index and $1 \leq j \leq N_c$ is the column index. Under our model problem assumption, the cost of forming a coarse-grid operator is thus approximately 6 work units. See [22] for more detail.

4.3.4 RITZ(q,N)

The idea behind the standard Ritz projection is to extract eigenvalue and eigenvector approximations of a large matrix $A \in \Re^{N \times N}$ from a given subspace. More precisely, the process first extracts the element of the subspace that best represents the smallest eigenvector of A, then

chooses the next from the orthogonal complement of this vector that best approximates the second eigenvector of A.

We start by finding a L2-orthonormal basis that spans the same space as $V \in \Re^{N\times q}$. The QR factorization of the matrix V is given by

$$V = QR,$$

where $Q \in \Re^{N\times q}$ is orthogonal and $R \in \Re^{q\times q}$ is upper triangular. Note that Q and V have the same range, but the columns of Q are orthonormal. We then look at the spectrum of the matrix $A_Q = Q^T AQ$, which is typically a matrix of much smaller dimension than A (assuming the column dimension of V is much smaller than the number of rows in A). One has to solve the eigenvalue problems

$$A_Q w_i = D_{ii} w_i,$$

where $1 \leq i \leq q$, which consumes relatively little computational effort as long as q is not too large. Each Qw_i (Ritz vector) is an approximation to an eigenvector of A with corresponding eigenvalue approximation (Ritz value) $D_{ii} = (Qw_i)^T A(Qw_i)/(Qw_i)^T(Qw_i)$. An algorithm for the Ritz projection is described in what follows.

Algorithm 4.2 Standard Ritz projection

Given matrices A$\in \Re^{N\times N}$ and V$\in \Re^{N\times q}$,
Apply the QR decomposition to V : $V = Q \times R$ where $Q' \times Q = I$.
Compute $A_Q = Q' \times A \times Q$.
Compute the spectrum of A_Q by $[WD] = eig(A_Q)$, where each column of W is an eigenvector of A_Q.
Return V = Q*W.

The matrix V we use in Algorithm 4.2 can be factored into upper triangular and orthogonal matrices, which can be computed in $2q^2(N - \frac{q}{3})$ flops [18]. Then the cost of performing the matrix triple product, $Q^T AQ$, can be treated as a sparse-dense matrix product and dense-dense matrix product. Since we consider work units as matrix-vector product on the finest grid, the cost of multiplying A and Q is q work units, where q is number of columns in Q. Then A_Q can be computed by multiplying Q^T to AQ. The cost for multiplying two dense matrices, B and C, with

$B \in R^{m\times n}$ and $C \in R^{n\times q}$, is $m^2(2n-1)$ flops (or $2m^2n$ if n is large). We can then compute A_Q in $q^2(2N-1)$ flops for a dense matrix product and q work units for a sparse-dense matrix multiplication. We can ignore the cost of computing the spectrum of A_Q because, compared to the size of A, namely, N, q is relatively small so its cost can be ignored. The last line in (4.2) is a dense matrix multiplication, which costs $Nq(2q-1)$. Dividing the operation counts by $18N$, we get that the cost of the Ritz projection is approximately $q + \frac{q^2}{3}$ work units.

4.3.5 Total Setup Cost

Combining these individual setup costs, we see that the total cost of the setup processes is

$$\frac{4}{3}(q(\nu+2)+6.6), \text{ if } M=0, \text{ or}$$

$$\begin{aligned}
&\frac{2}{27}M(M+1)(2M+1)+\frac{1}{6}(\nu+8q+8)M(M+1)\\
&+\frac{1}{3}(4\alpha(\nu_1+\nu_2+\frac{1}{4})+q\cdot\nu+4(2q+6.6)+24+4(\frac{q}{3}+q^2))M\\
&+\frac{4}{3}(q(\nu+2)+6.6), \text{ if } M>0.
\end{aligned}$$

Chapter 5

Numerical Experiments

5.1 BAMG vs *i*BAMG

The focus of this section is a comparative study of the performance of BAMG and iBAMG. As is usually done in AMG, the coarse-grid matrices are formed by way of the Galerkin approach based on the computed interpolation operator, namely, $A^c = P^T AP$. We produce further coarsening by applying the least-squares processes in BAMG and iBAMG to A^c. Since our focus is on how these approaches compare in their determination of the weights of interpolation, we force standard coarsening on every level. In both the BAMG and iBAMG setup phases, we relax q different random vectors of unit Euclidean length ν times for the homogeneous problem. In all cases, we use pointwise Gauss-Seidel iteration with lexicographic ordering of the grid points as the relaxation method. All of the results in the following tables reflect an average residual reduction factor of the resulting AMG solver over a total residual reduction by a factor of 10^{10} (or over 50 cycles, whichever comes first). (All of the tests reported in this thesis are for the basic AMG solver, without the use any acceleration processes like conjugate gradients.)

5.1.1 2D Poisson

Consider the two-dimensional Poisson problem with homogeneous Dirichlet boundary conditions on the unit square given by

$$\begin{aligned} -\Delta u &= f \quad \text{in } \Omega = (0,1)^2, \\ u &= 0 \quad \text{on } \delta\Omega. \end{aligned} \tag{5.1}$$

We discretize (5.1) using standard bilinear elements on a uniform grid, which yields the nine-point stencil given by

$$A = \frac{1}{3h^2} \begin{pmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{pmatrix}. \tag{5.2}$$

(We scaled A here by h^{-2} for later convenience.) Although this model problem is not the ultimate target for these methods, it is important to compare the two bootstrap approaches in this simple setting because we can take advantage of knowing optimal coarsening and interpolation weights for the classical AMG approach.

Because of our use of standard coarsening, some of the F points have four neighbors in their interpolatory set. Thus, the use of fewer than four targets ensures an underdetermined least-squares system for these points. Accordingly, the first three rows of each table show the results of using the minimal-deviation iBAMG approach described in Section 3.2.3. (Standard BAMG does not apply as is to the underdetermined case, since it is generally assumed there that $q \geq |C_i|$.) Note that the use of just one target in Tables 5.1 to 5.4 yields remarkably good performance in all cases, while performance degrades substantially for $q = 2$ and 3. This result is somewhat special to this problem. For $q = 1$, minimizing (3.12) amounts to choosing the equal interpolation weights that match the target vector (with the sum of the weights becoming more accurate as the number of relaxation sweeps is increased). This choice is exactly what is needed for the Poisson problem, so one vector is sufficient here. In fact, using two vectors results in degradation of convergence, as the tables show. The results of the next section show a somewhat more monotone pattern of increasing improvement with increasing q.

Tables 5.1 to 5.4 show results for BAMG (iBAMG) using two levels with $h = 1/64$, followed by V-cycles with $h = 1/64$, $1/128$, and $1/256$. It is worth pointing out some general trends that are common to all the results presented. First (ignoring $q \leq 3$ for the moment, which will be discussed below), not surprisingly, convergence generally improves with increasing q and ν. Generally, increasing q is better than increasing ν. A rough measure of setup cost is $q \times \nu$, although,

due to the cost of computing interpolation weights, work actually increases somewhat more than linearly with q. Some trends to note are that V-cycle results, again not surprisingly, are somewhat worse than corresponding two-level results. This effect is more pronounced for smaller q and ν. In addition, while the best V-cycle convergence factors for each mesh size are the same, convergence degrades somewhat with mesh size in more borderline cases. However, the minimal q and ν required to reach near-optimal convergence increases only mildly, at least for the range of problem sizes considered.

For $q \geq 4 \geq |C_i|$, the consistent trend is that better performance of the resulting solver is obtained by more vectors and/or more relaxation sweeps. Moreover, iBAMG tends to provide substantially better performance for the same number of vectors and sweeps.

q/ν	1	2	3	4	5	6	7	8	9	10
1	(.45)	(.08)	(.06)	(.06)	(.06)	(.06)	(.06)	(.06)	(.06)	(.06)
2	(.62)	(.47)	(.40)	(.30)	(.33)	(.36)	(.35)	(.33)	(.41)	(.29)
3	(.42)	(.21)	(.20)	(.17)	(.17)	(.14)	(.14)	(.18)	(.18)	(.16)
4	.70	.54	.44	.41	.37	.38	.39	.39	.38	.36
	(.46)	(.32)	(.27)	(.25)	(.22)	(.23)	(.21)	(.21)	(.20)	(.21)
5	.63	.39	.32	.27	.25	.25	.24	.24	.25	.25
	(.31)	(.18)	(.15)	(.14)	(.13)	(.10)	(.09)	(.11)	(.11)	(.10)
6	.57	.31	.23	.21	.20	.20	.18	.19	.18	.19
	(.21)	(.12)	(.09)	(.07)	(.07)	(.06)	(.06)	(.06)	(.06)	(.06)
7	.53	.26	.20	.16	.16	.15	.14	.14	.14	.13
	(.17)	(.08)	(.06)	(.06)	(.06)	(.06)	(.06)	(.06)	(.06)	(.06)
8	.48	.21	.16	.14	.14	.12	.12	.13	.11	.11
	(.14)	(.07)	(.06)	(.06)	(.06)	(.06)	(.06)	(.06)	(.06)	(.06)
9	.44	.19	.14	.13	.12	.11	.10	.10	.11	.10
	(.12)	(.06)	(.06)	(.06)	(.06)	(.06)	(.06)	(.06)	(.06)	(.06)
10	.41	.16	.13	.11	.11	.10	.09	.10	.09	.09
	(.11)	(.06)	(.06)	(.06)	(.06)	(.06)	(.06)	(.06)	(.06)	(.06)

Table 5.1: Average V(1,1) two-level convergence factors for residual reduction by a factor of 10^{10} (or at most 50 cycles) on a 64×64 grid 9-point discretization of 2-dimensional model problem (5.1) for various combinations of the number of relaxation sweeps, ν, and the number of random test vectors, q. Shown here are the average convergence factors using BAMG (iBAMG). In all cases, a random initial guess was used to test the resulting cycle.

q/ν	1	2	3	4	5	6	7	8	9	10
1	(.54)	(.12)	(.08)	(.08)	(.08)	(.08)	(.08)	(.08)	(.08)	(.08)
2	(.87)	(.86)	(.85)	(.84)	(.84)	(.84)	(.81)	(.81)	(.80)	(.81)
3	(.81)	(.70)	(.59)	(.50)	(.41)	(.35)	(.39)	(.30)	(.28)	(.26)
4	.87	.87	.87	.87	.87	.87	.87	.87	.87	.87
	(.87)	(.86)	(.86)	(.86)	(.86)	(.86)	(.85)	(.85)	(.85)	(.85)
5	.86	.83	.79	.78	.78	.78	.76	.72	.77	.79
	(.72)	(.52)	(.39)	(.33)	(.49)	(.48)	(.33)	(.56)	(.29)	(.31)
6	.81	.69	.57	.61	.51	.59	.61	.55	.57	.59
	(.44)	(.24)	(.11)	(.11)	(.09)	(.09)	(.13)	(.17)	(.09)	(.11)
7	.77	.54	.39	.34	.35	.40	.30	.27	.37	.34
	(.31)	(.12)	(.08)	(.08)	(.08)	(.08)	(.08)	(.08)	(.08)	(.08)
8	.73	.45	.31	.23	.23	.19	.25	.18	20	.21
	(.26)	(.10)	(.08)	(.08)	(.08)	(.08)	(.08)	(.08)	(.08)	(.08)
9	.69	.36	.22	.19	.17	.17	.16	.16	.15	.15
	(.22)	(.09)	(.08)	(.08)	(.08)	(.08)	(.08)	(.08)	(.08)	(.08)
10	.66	.33	.20	.16	.14	.16	.12	.12	.12	.12
	(.20)	(.08)	(.08)	(.08)	(.08)	(.08)	(.08)	(.08)	(.08)	(.08)

Table 5.2: Average five-level V(1,1) convergence factors for residual reduction by a factor of 10^{10} (or at most 50 cycles) on a 64×64 grid 9-point discretization of 2-dimensional model problem (5.1) for various combinations of the number of relaxation sweeps, ν, and the number of random test vectors, q. Shown here are the average convergence factors using BAMG (iBAMG). In all cases, a random initial guess was used to test the resulting cycle.

q/ν	1	2	3	4	5	6	7	8	9	10
1	(.75)	(.31)	(.11)	(.08)	(.08)	(.08)	(.08)	(.08)	(.08)	(.08)
2	(.86)	(.86)	(.86)	(.86)	(.86)	(.85)	(.85)	(.85)	(.85)	(.85)
3	(.84)	(.80)	(.76)	(.73)	(.67)	(.65)	(.61)	(.56)	(.51)	(.48)
4	.87	.87	.87	.87	.87	.87	.87	.87	.87	.87
	(.86)	(.86)	(.86)	(.86)	(.86)	(.86)	(.86)	(.86)	(.86)	(.86)
5	.86	.85	.85	.85	.84	.84	.84	.84	.84	.84
	(.81)	(.75)	(.69)	(.68)	(.67)	(.72)	(.72)	(.70)	(.69)	(.67)
6	.84	.79	.74	.71	.74	.73	.75	.73	.75	.75
	(.66)	(.35)	(.21)	(.22)	(.21)	(.26)	(.18)	(.18)	(.15)	(.21)
7	.82	.71	.61	.56	.58	.54	.51	.53	.60	.60
	(.53)	(.16)	(.10)	(.09)	(.08)	(.08)	(.11)	(.08)	(.09)	(.08)
8	.80	.61	.43	.36	.40	.43	.34	.35	.39	.35
	(.47)	(.13)	(.08)	(.08)	(.08)	(.08)	(.08)	(.08)	(.08)	(.08)
9	.79	.53	.33	.26	.24	.24	.19	.22	.20	.23
	(.42)	(.11)	(.08)	(.08)	(.08)	(.08)	(.08)	(.08)	(.08)	(.08)
10	.77	.48	.27	.21	.17	.17	.14	.15	.16	.14
	(.38)	(.10)	(.08)	(.08)	(.08)	(.08)	(.08)	(.08)	(.08)	(.08)

Table 5.3: Average six-level V(1,1) convergence factors for residual reduction by a factor of 10^{10} (or at most 50 cycles) on a 128×128 grid 9-point discretization of 2-dimensional model problem (5.1) for various combinations of the number of relaxation sweeps, ν, and the number of random test vectors, q. Shown here are the average convergence factors using BAMG (iBAMG). In all cases, a random initial guess was used to test the resulting cycle.

q/ν	1	2	3	4	5	6	7	8	9	10
1	(.82)	(.61)	(.29)	(.14)	(.08)	(.08)	(.08)	(.08)	(.08)	(.08)
2	(.86)	(.85)	(.85)	(.85)	(.85)	(.85)	(.85)	(.85)	(.85)	(.85)
3	(.84)	(.83)	(.82)	(.82)	(.80)	(.78)	(.79)	(.76)	(.77)	(.73)
4	.86	.86	.86	.86	.86	.86	.86	.86	.86	.86
	(.86)	(.85)	(.85)	(.85)	(.85)	(.85)	(.85)	(.85)	(.85)	(.85)
5	.85	.85	.84	.84	.84	.84	.84	.84	.84	.84
	(.83)	(.82)	(.82)	(.81)	(.81)	(.82)	(.82)	(.82)	(.82)	(.82)
6	.84	.82	.81	.81	.81	.82	.82	.82	.82	.82
	(.77)	(.53)	(.43)	(.31)	(.46)	(.47)	(.43)	(.38)	(.51)	(.36)
7	.83	.78	.75	.73	.74	.76	.74	.76	.77	.77
	(.72)	(.31)	(.17)	(.13)	(.14)	(.15)	(.09)	(.08)	(.09)	(.08)
8	.83	.73	.62	.61	.61	.65	.64	.66	.65	.65
	(.69)	(.26)	(.10)	(.08)	(.08)	(.08)	(.08)	(.08)	(.08)	(.08)
9	.82	.68	.48	.43	.39	.41	.41	.39	.36	.35
	(.66)	(.22)	(.09)	(.08)	(.08)	(.08)	(.08)	(.08)	(.08)	(.08)
10	.82	.65	.38	.28	.23	.25	.23	.27	.28	.22
	(.64)	(.20)	(.08)	(.08)	(.08)	(.08)	(.08)	(.08)	(.08)	(.08)

Table 5.4: Average seven-level V(1,1) convergence factors for residual reduction by a factor of 10^{10} (or at most 50 cycles) on a 256×256 grid 9-point discretization of 2-dimensional model problem (5.1) for various combinations of the number of relaxation sweeps, ν, and the number of random test vectors, q. Shown here are the average convergence factors using BAMG (iBAMG). In all cases, a random initial guess was used to test the resulting cycle.

5.1.2 Scaled 2D Laplacian

To maintain the geometric simplicity of our model problem but test performance in the presence of widely varying coefficients, our next example is produced by symmetrically scaling the matrix resulting from (5.1) by a positive diagonal matrix $D = (D_{ii}) = \left(e^{(r_i)}\right)$, where r_i is a random number between -5 and 5. The scaling is done as follows:

$$A \leftarrow DAD. \tag{5.3}$$

Results for this test are shown in Table 5.5. For both methods, we see mostly improved convergence as the number of target vectors and/or smoothing steps increases. The principal exception is again the case $q = 1$, although the results are not as remarkable as they were for the standard model problem. Note that the performance of iBAMG is again substantially better than that of BAMG in most cases.

Comparing Table 5.5 with 5.1, results are similar for $\nu \geq 4$. For $\nu < 4$, convergence becomes increasingly worse (relatively) with smaller ν.

q/ν	1	2	3	4	5	6	7	8	9	10
1	(.77)	(.58)	(.43)	(.35)	(.30)	(.26)	(.22)	(.23)	(.22)	(.21)
2	(.77)	(.65)	(.52)	(.42)	(.40)	(.38)	(.36)	(.35)	(.36)	(.35)
3	(.76)	(.54)	(.32)	(.25)	(.22)	(.21)	(.19)	(.19)	(.19)	(.19)
4	.79	.69	.58	.49	.42	.43	.39	.38	.37	.37
	(.76)	(.54)	(.34)	(.27)	(.26)	(.23)	(.22)	(.20)	(.20)	(.21)
5	.78	.66	.47	.35	.30	.28	.26	.26	.24	.24
	(.75)	(.43)	(.20)	(.13)	(.13)	(.11)	(.10)	(.09)	(.11)	(.11)
6	.78	.64	.43	.27	.23	.20	.20	.20	.19	.19
	(.74)	(.38)	(.15)	(.09)	(.07)	(.07)	(.06)	(.06)	(.06)	(.06)
7	.77	.63	.38	.24	.19	.18	.16	.15	.15	.14
	(.74)	(.35)	(.13)	(.07)	(.05)	(.06)	(.06)	(.06)	(.06)	(.06)
8	.77	.62	.36	.21	.15	.14	.13	.13	.12	.12
	(.73)	(.34)	(.12)	(.06)	(.06)	(.06)	(.06)	(.06)	(.06)	(.06)
9	.77	.62	.33	.19	.14	.12	.12	.10	.10	.10
	(.73)	(.33)	(.11)	(.06)	(.06)	(.06)	(.06)	(.06)	(.06)	(.05)
10	.76	.62	.33	.19	.13	.11	.10	.10	.09	.09
	(.73)	(.32)	(.11)	(.06)	(.05)	(.05)	(.05)	(.05)	(.05)	(.05)

Table 5.5: Average two-level V(1,1) convergence factors for residual reduction by a factor of 10^{10} (or at most 50 cycles) on a 64×64 grid 9-point discretization of 2-dimensional model problem (5.1) subject to diagonal scaling according to (5.3) for various combinations of the number of relaxation sweeps, ν, and the number of random test vectors, q. Shown here are the average convergence factors using BAMG (iBAMG). In all cases, a random initial guess was used to test the resulting cycle.

5.1.3 Long-range Interpolation

BAMG naturally allows for long-range interpolation, using any of the given coarse grids. *i*BAMG also naturally provides this capability by allowing the collapse of unwanted connections to any specified subset of the C points. To illustrate this capability, Table 5.6 shows two-level result for the 5-point Laplace operator with standard coarsening. (See Figure 5.1, where some F points are not directly connected to any points in C_i.) Figures 5.2 and 5.3 illustrate the collapsing patterns for BAMG and *i*BAMG, respectively, with 5-point stencils and standard coarsening.

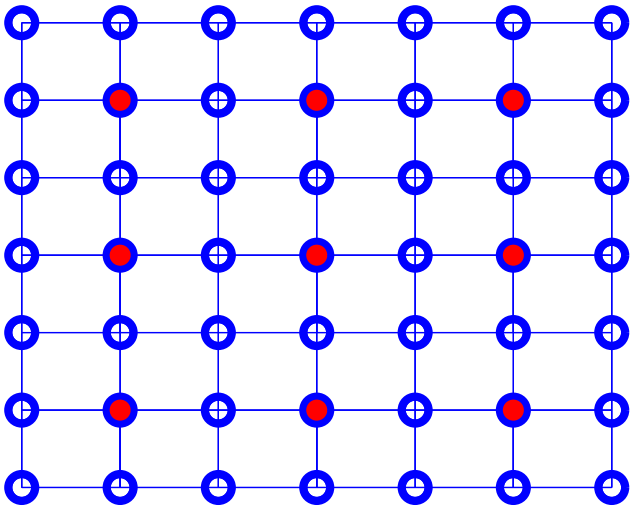

Figure 5.1: Standard coarsening. The white-centered blue circles represent F points and the red-centered ones represent C points.

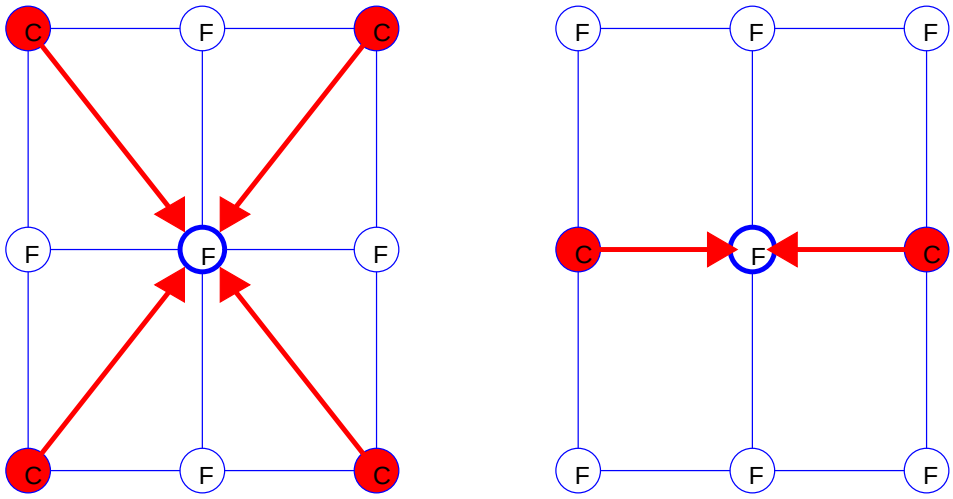

Figure 5.2: Least-squares approximation for BAMG. Shown here are two types of F points, one involving interpolation from 4 C points and the other involving just 2.

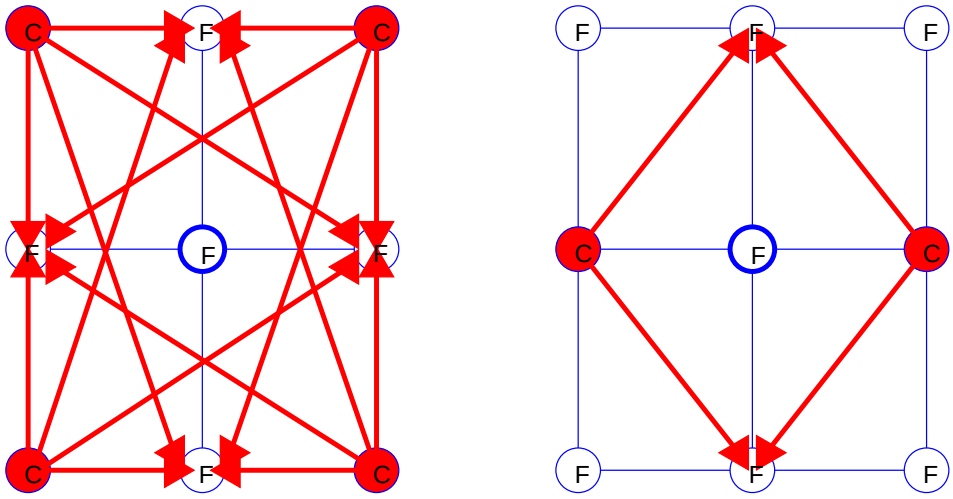

Figure 5.3: Least squares approximation for iBAMG.

q/ν	1	2	3	4	5	6	7	8	9	10
4	.74 (.67)	.65 (.60)	.58 (.52)	.55 (.45)	.52 (.42)	.50 (.40)	.50 (.41)	.51 (.39)	.49 (.41)	.49 (.38)
5	.72 (.58)	.55 (.44)	.44 (.35)	.39 (.31)	.38 (.28)	.37 (.26)	.36 (.26)	.36 (.25)	.35 (.25)	.35 (.24)
6	.69 (.53)	.45 (.34)	.36 (.28)	.34 (.23)	.30 (.21)	.30 (.19)	.29 (.18)	.27 (.18)	.28 (.19)	.30 (.17)
7	.66 (.46)	.39 (.31)	.31 (.22)	.28 (.16)	.27 (.15)	.25 (.14)	.24 (.14)	.24 (.14)	.22 (.16)	.23 (.13)
8	.65 (.43)	.35 (.25)	.27 (.17)	.24 (.14)	.23 (.13)	.22 (.12)	.21 (.12)	.21 (.12)	.20 (.13)	.20 (.12)
9	.63 (.38)	.31 (.22)	.24 (.15)	.22 (.12)	.21 (.11)	.19 (.11)	.19 (.11)	.18 (.12)	.18 (.12)	.17 (.12)
10	.61 (.36)	.28 (.19)	.23 (.13)	.20 (.11)	.19 (.11)	.18 (.11)	.18 (.11)	.16 (.11)	.16 (.12)	.17 (.12)

Table 5.6: Average two-level V(1,1) convergence factors for residual reduction by a factor of 10^{10} (or at most 50 cycles) on a 64×64 grid 5-point discretization of 2-dimensional model problem (5.1) for various combinations of the number of relaxation sweeps, ν, and the number of random test vectors, q. Shown here are the average convergence factors using BAMG (iBAMG) with standard coarsening. In all cases, a random initial guess was used to test the resulting cycle.

5.1.4 Variable-coefficient 2D diffusion

Our next example represents a more taxing problem for optimal solvers. It was chosen here because it can cause difficulty with some implementations of standard MG. The results for this specific case are also fairly representative of our experience so far with variable-coefficient problems, including those with much larger jumps in the coefficients.

Consider the two-dimensional variable-coefficient problem with homogeneous Dirichlet boundary conditions on the unit square given by

$$\begin{aligned} -\nabla \cdot (d(x,y)\nabla u) &= f \quad \text{in } \Omega = (0,1)^2, \\ u &= 0 \quad \text{on } \delta\Omega\,. \end{aligned} \tag{5.4}$$

where, as shown in Figure 5.4, we have

$$d(x,y) = \begin{cases} 1 & .25 < \max(|x-.5|, |y-.5|) < .375\,, \\ 1000 & \text{otherwise.} \end{cases} \tag{5.5}$$

We discretize (5.4) by the Galerkin finite element method using piecewise bilinear elements, which yields the nine-point stencil given by

$$A = \frac{1}{3h^2} \begin{pmatrix} -d_{nw} & -\frac{(d_{nw}+d_{ne})}{2} & -d_{ne} \\ -\frac{(d_{nw}+d_{sw})}{2} & 2(d_{nw}+d_{ne}+d_{sw}+d_{se}) & -\frac{(d_{se}+d_{ne})}{2} \\ -d_{sw} & -\frac{(d_{sw}+d_{se})}{2} & -d_{se} \end{pmatrix}.$$

Our element boundaries align with the discontinuities in d so that the entries in this stencil refer in an obvious way to the values of d in neighboring elements of each grid point. Table 5.7 shows convergence factors, comparisons, and trends for a 64×64 grid that are quite similar to what we saw for the Poisson case.

In Table 5.8, we show observed convergence factors for the same problem except that the red island in Figure 5.4 is shifted up and right $h = \frac{1}{64}$ so that discontinuities do not align with any

coarse-grid line:

$$d(x,y) = \begin{cases} 1 & .25 < \max(|x - .5 - \frac{1}{64}|, |y - .5 - \frac{1}{64}|) < .375\,, \\ 1000 & \text{otherwise.} \end{cases} \tag{5.6}$$

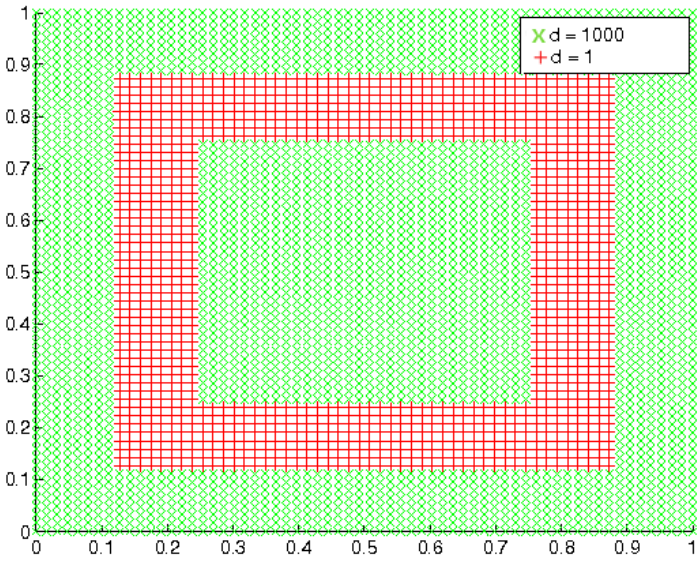

Figure 5.4: The distribution of the coefficient for (5.4) on $[0, 1]^2$.

q/ν	1	2	3	4	5	6	7	8	9	10
1	(.80)	(.80)	(.72)	(.60)	(.45)	(.35)	(.25)	(.21)	(.16)	(.15)
2	(.81)	(.81)	(.78)	(.74)	(.66)	(.62)	(.54)	(.51)	(.47)	(.46)
3	(.81)	(.79)	(.70)	(.58)	(.47)	(.36)	(.28)	(.27)	(.27)	(.28)
4	.81	.81	.78	.73	.67	.59	.54	.50	.45	.44
	(.81)	(.79)	(.70)	(.58)	(.44)	(.35)	(.29)	(.26)	(.24)	(.26)
5	.81	.81	.73	.65	.55	.43	.35	.32	.28	.30
	(.81)	(.75)	(.57)	(.36)	(.23)	(.16)	(.14)	(.11)	(.12)	(.11)
6	.81	.80	.73	.58	.45	.32	.27	.23	.23	.22
	(.81)	(.71)	(.49)	(.30)	(.17)	(.11)	(.08)	(.09)	(.09)	(.07)
7	.81	.79	.68	.51	.36	.25	.20	.17	.15	.16
	(.80)	(.69)	(.44)	(.23)	(.13)	(.09)	(.07)	(.06)	(.06)	(.06)
8	.81	.78	.65	.47	.30	.21	.16	.14	.13	.13
	(.80)	(.67)	(.41)	(.20)	(.11)	(.08)	(.06)	(.06)	(.06)	(.06)
9	.81	.78	.63	.43	.25	.17	.13	.12	.11	.12
	(.80)	(.65)	(.39)	(.19)	(.09)	(.07)	(.06)	(.06)	(.06)	(.06)
10	.81	.78	.62	.39	.22	.15	.11	.10	.11	.11
	(.80)	(.62)	(.33)	(.17)	(.08)	(.06)	(.06)	(.06)	(.06)	(.06)

Table 5.7: Average two-level V(1,1) convergence factors for residual reduction by a factor of 10^{10} (or at most 50 cycles) on a 64×64 grid 9-point discretization of 2-dimensional model problem (5.4) with the coefficient in (5.5) for various combinations of the number of relaxation sweeps, ν, and the number of random test vectors, q. Shown here are the average convergence factors using BAMG (iBAMG). In all cases, a random initial guess was used to test the resulting cycle.

q/ν	1	2	3	4	5	6	7	8	9	10
1	(.76)	(.78)	(.73)	(.61)	(.49)	(.35)	(.28)	(.24)	(.24)	(.23)
2	(.77)	(.79)	(.77)	(.74)	(.70)	(.63)	(.54)	(.53)	(.49)	(.47)
3	(.78)	(.78)	(.70)	(.57)	(.46)	(.36)	(.33)	(.26)	(.27)	(.25)
4	.76	.78	.77	.74	.65	.60	.55	.50	.48	.43
	(.78)	(.77)	(.70)	(.57)	(.49)	(.36)	(.29)	(.31)	(.29)	(.27)
5	.77	.78	.74	.65	.54	.42	.37	.33	.32	.31
	(.78)	(.73)	(.58)	(.37)	(.27)	(.22)	(.20)	(.18)	(.18)	(.17)
6	.78	.78	.72	.56	.45	.32	.27	.24	.24	.24
	(.78)	(.72)	(.52)	(.30)	(.20)	(.19)	(.18)	(.17)	(.17)	(.15)
7	.78	.78	.68	.51	.37	.27	.22	.19	.20	.19
	(.77)	(.69)	(.44)	(.25)	(.19)	(.20)	(.17)	(.16)	(.14)	(.13)
8	.77	.76	.66	.47	.30	.23	.20	.19	.17	.19
	(.77)	(.66)	(.44)	(.23)	(.18)	(.17)	(.14)	(.15)	(.15)	(.12)
9	.78	.76	.63	.43	.26	.20	.20	.19	.18	.18
	(.77)	(.65)	(.38)	(.20)	(.18)	(.16)	(.16)	(.14)	(.14)	(.13)
10	.78	.75	.60	.38	.26	.20	.19	.16	.16	.17
	(.77)	(.62)	(.35)	(.19)	(.18)	(.15)	(.14)	(.15)	(.13)	(.11)

Table 5.8: Average two-level V(1,1) convergence factors for residual reduction by a factor of 10^{10} (or at most 50 cycles) on a 64×64 grid 9-point discretization of 2-dimensional model problem (5.4) with the coefficient in (5.6) for various combinations of the number of relaxation sweeps, ν, and the number of random test vectors, q. Shown here are the average convergence factors using BAMG (iBAMG) with standard coarsening. In all cases, a random initial guess was used to test the resulting cycle.

5.1.5 Diagonally scaled variable-coefficient 2D diffusion

Our next example comes from diagonally scaling the matrix resulting from (5.4) with the coefficient in (5.5) by the positive diagonal matrix $D = (D_{ii}) = \left(a_{ii}^{-1/2}\right)$ as follows:

$$A \leftarrow DAD. \tag{5.7}$$

The results as shown in Table 5.9 again are similar to what we have seen in the other examples.

q/ν	1	2	3	4	5	6	7	8	9	10
1	(.79)	(.79)	(.81)	(.81)	(.81)	(.82)	(.82)	(.82)	(.83)	(.83)
2	(.77)	(.77)	(.79)	(.80)	(.80)	(.81)	(.80)	(.80)	(.80)	(.80)
3	(.77)	(.78)	(.79)	(.79)	(.79)	(.75)	(.74)	(.74)	(.69)	(.59)
4	.77	.76	.78	.79	.79	.79	.77	.76	.72	.70
	(.77)	(.77)	(.78)	(.76)	(.75)	(.71)	(.72)	(.65)	(.55)	(.44)
5	.78	.77	.78	.78	.76	.74	.69	.65	.57	.47
	(.77)	(.78)	(.75)	(.69)	(.57)	(.45)	(.44)	(.38)	(.27)	(.25)
6	.78	.76	.77	.77	.75	.69	.63	.56	.48	.34
	(.77)	(.77)	(.71)	(.64)	(.48)	(.38)	(.27)	(.25)	(.24)	(.18)
7	.78	.78	.77	.76	.70	.66	.61	.49	.40	.30
	(.76)	(.77)	(.71)	(.59)	(.47)	(.33)	(.26)	(.21)	(.19)	(.16)
8	.78	.77	.77	.75	.68	.66	.54	.43	.34	.31
	(.75)	(.76)	(.70)	(.57)	(.42)	(.32)	(.25)	(.21)	(.18)	(.15)
9	.78	.78	.77	.75	.70	.63	.55	.40	.33	.26
	(.77)	(.75)	(.67)	(.53)	(.38)	(.28)	(.24)	(.18)	(.16)	(.14)
10	.78	.77	.76	.72	.68	.59	.49	.38	.31	.26
	(.76)	(.76)	(.67)	(.50)	(.37)	(.27)	(.22)	(.18)	(.14)	(.14)

Table 5.9: Average two-level V(1,1) convergence factors for residual reduction by a factor of 10^{10} (or at most 50 cycles) on a 64×64 grid 9-point discretization of 2-dimensional model problem (5.4) subject to diagonal scaling according to (5.7) with coefficient in (5.5) for various combinations of the number of relaxation sweeps, ν, and the number of random test vectors, q. Shown here are the average convergence factors using BAMG (iBAMG). In all cases, a random initial guess was used to test the resulting cycle.

5.2 Adaptive BAMG

Our final tests focus on performance of the full setup process of rBAMG that uses both a least-squares phase that determines interpolation based on q initial random vectors relaxed v times and a subsequent adaptive phase that aims to test and possibly improve the resulting solver. Guided by our convergence estimation process, each iteration of the adaptive phase assesses the current solver's fine-grid convergence factor on random initial guesses for $Ax = 0$ and, when poor convergence is observed, combines the resulting error with the test vectors and errors from earlier adaptive iterations.

In all of our numerical experiments, we used 4 V(1,1) cycles for each adaptive iteration. The values of two threshold parameters used here for convergence estimation are $\rho_{good} = .3$ and $\rho_{bad} = .8$. Note that the adaptive iterations continue until the estimated convergence factor, ρ_{est}, is below ρ_{good} or the estimated total cost for the full solution process is expected to increase. Because of the five-point form of the gauge Laplacian, it is customary for the first coarsening to be red-black coarsening to be red-black, making direct (and 'ideal') operator interpolation possible. We used this coarsening here. Because this produces a 9-point coarse-grid stencil on a rotated uniform coarse grid, we were able to use rotated standard coarsening for the coarser levels. All setup processes and solvers used pointwise lexicographic Gauss-Seidel as the relaxation scheme.

Each set of test results for each of our two sample problems and each value N is reported in a pair of tables. The first of each pair assesses the total setup and solver costs in terms of work units. The setup costs are calculated using the total cost estimates described in the previous chapter. The solver costs are computed in a separate test phase that applies V(1,1) cycles of the final solver to a random initial guess for $Ax = 0$. The solver costs are computed based on using the observed asymptotic convergence factor to determine how many of these cycles would be needed to reduce the error for a general linear solve by ten orders of magnitude. The second of the pair of tables for each test set depicts the number of 'target' vectors (test vectors and added adaptive error components) that were used to fit interpolation for the final adaptive iteration. For both sample

problems, we also include a table that summarizes the best results observed for each N.

5.2.1 Shifted 2D Poisson

Here we consider the 9-point stencil in (5.2) shifted by subtracting a constant from the diagonal of the matrix that is computed so that its smallest eigenvalue is $\frac{1}{N^2}$, with $N = 64, 128,$ and 256.

q/ν	1	2	3	4	5	6	7	8	9	10
4	247	228	201	224	203	231	246	231	233	260
5	210	162	158	155	177	185	180	185	194	202
6	174	152	156	172	177	192	203	205	214	226
7	166	158	166	177	189	201	213	225	237	194
8	165	170	189	192	206	220	233	251	193	185
9	177	181	193	208	223	239	254	242	215	195
10	183	214	208	225	242	259	276	209	198	209

Table 5.10: Total cost (= setup + solver cost) for rBAMG applied to shifted 2D Poisson and N = 64.

q/ν	1	2	3	4	5	6	7	8	9	10
4	6.80 (0.24)	6.20 (0.26)	6.00 (0.16)	6.00 (0.23)	5.70 (0.19)	5.90 (0.21)	6.10 (0.15)	5.70 (0.19)	5.60 (0.21)	5.80 (0.20)
5	7.00 (0.26)	6.20 (0.24)	6.10 (0.18)	6.00 (0.14)	6.20 (0.13)	6.20 (0.11)	6.00 (0.15)	6.00 (0.11)	6.00 (0.11)	6.00 (0.10)
6	7.30 (0.27)	7.00 (0.19)	7.00 (0.13)	7.10 (0.11)	7.00 (0.13)	7.10 (0.10)	7.10 (0.10)	7.00 (0.10)	7.00 (0.10)	7.00 (0.11)
7	8.10 (0.24)	8.00 (0.15)	8.00 (0.10)	8.00 (0.10)	8.00 (0.10)	8.00 (0.10)	8.00 (0.10)	8.00 (0.10)	8.00 (0.10)	7.30 (0.23)
8	9.00 (0.22)	9.00 (0.13)	9.10 (0.10)	9.00 (0.10)	9.00 (0.10)	9.00 (0.10)	9.00 (0.09)	9.00 (0.12)	8.20 (0.24)	8.00 (0.27)
9	10.00 (0.22)	10.00 (0.11)	10.00 (0.10)	10.00 (0.10)	10.00 (0.10)	10.00 (0.10)	10.00 (0.10)	9.70 (0.16)	9.20 (0.29)	9.00 (0.24)
10	11.00 (0.18)	11.20 (0.11)	11.00 (0.10)	11.00 (0.10)	11.00 (0.10)	11.00 (0.10)	11.00 (0.10)	10.20 (0.25)	10.00 (0.26)	10.00 (0.24)

Table 5.11: Final number of target vectors and solver convergence factor for rBAMG applied to shifted 2D Poisson and N = 64.

Table 5.18 summarizes the optimal results observed in the other tables for the shifted Poisson problem. For each N, the smallest value is identified in each total cost table, and the corresponding

q/ν	1	2	3	4	5	6	7	8	9	10
4	300	247	266	270	267	290	285	286	294	321
5	240	206	171	188	195	198	206	231	241	225
6	192	168	161	165	174	183	203	214	214	226
7	166	160	185	177	189	201	213	225	237	249
8	168	170	179	192	206	220	233	247	261	274
9	176	182	211	208	223	239	254	269	285	290
10	184	192	208	225	242	259	276	293	310	247

Table 5.12: Total cost (= setup + solver cost) for rBAMG applied to shifted 2D Poisson and N = 128.

q/ν	1	2	3	4	5	6	7	8	9	10
4	7.50 (.27)	6.70 (.21)	6.80 (.21)	6.70 (.19)	6.60 (.17)	6.60 (.21)	6.50 (.20)	6.40 (.20)	6.40 (.19)	6.60 (.19)
5	7.40 (.27)	6.90 (.20)	6.20 (.25)	6.40 (.18)	6.40 (.15)	6.30 (.17)	6.30 (.15)	6.50 (.15)	6.50 (.15)	6.20 (.16)
6	7.50 (.28)	7.20 (.20)	7.00 (.17)	7.00 (.11)	7.00 (.11)	7.00 (.10)	7.10 (.10)	7.10 (.10)	7.00 (.10)	7.00 (.11)
7	8.00 (.31)	8.00 (.17)	8.20 (.11)	8.00 (.10)	8.00 (.10)	8.00 (.10)	8.00 (.10)	8.00 (.10)	8.00 (.10)	8.00 (.10)
8	9.00 (.25)	9.00 (.14)	9.00 (.10)	9.00 (.10)	9.00 (.10)	9.00 (.10)	9.00 (.10)	9.00 (.10)	9.00 (.10)	9.00 (.10)
9	10.00 (.23)	10.00 (.12)	10.00 (.16)	10.00 (.10)	10.00 (.10)	10.00 (.10)	10.00 (.10)	10.00 (.10)	10.00 (.10)	9.90 (.12)
10	11.00 (.21)	11.00 (.11)	11.00 (.10)	11.00 (.10)	11.00 (.10)	11.00 (.10)	11.00 (.10)	11.00 (.10)	11.00 (.10)	10.30 (.23)

Table 5.13: Final number of target vectors and solver convergence factor for rBAMG applied to shifted 2D Poisson and N = 128.

q/ν	1	2	3	4	5	6	7	8	9	10
4	352	326	314	302	320	310	332	362	385	368
5	271	253	220	227	238	241	255	257	286	293
6	238	167	167	176	183	209	219	227	221	254
7	229	176	175	191	194	206	218	230	242	254
8	207	179	184	207	211	225	238	252	277	279
9	239	187	198	213	228	248	263	274	295	305
10	225	200	212	229	246	269	280	297	315	340

Table 5.14: Total cost (= setup + solver cost) for rBAMG applied to shifted 2D Poisson and N = 256.

q, ν, number of target vectors, and final convergence factors are depicted. The results here suggest

q/ν	1	2	3	4	5	6	7	8	9	10
4	8.20	7.80	7.50	7.20	7.10	6.90	7.10	7.30	7.40	7.00
	(.30)	(.25)	(.22)	(.23)	(.31)	(.34)	(.26)	(.25)	(.26)	(.38)
5	7.80	7.50	6.90	6.80	6.80	6.70	6.70	6.70	6.90	6.80
	(.30)	(.21)	(.21)	(.25)	(.22)	(.24)	(.27)	(.21)	(.19)	(.29)
6	8.10	7.00	7.00	7.00	7.00	7.20	7.20	7.10	7.00	7.20
	(.29)	(.25)	(.20)	(.18)	(.14)	(.13)	(.13)	(.18)	(.11)	(.15)
7	8.80	8.10	8.00	8.10	8.00	8.00	8.00	8.00	8.00	8.00
	(.26)	(.18)	(.14)	(.11)	(.11)	(.11)	(.10)	(.10)	(.10)	(.10)
8	9.30	9.00	9.00	9.10	9.00	9.00	9.00	9.00	9.10	9.00
	(.27)	(.18)	(.11)	(.10)	(.11)	(.11)	(.10)	(.10)	(.10)	(.10)
9	10.60	10.00	10.00	10.00	10.00	10.00	10.00	10.00	10.00	10.00
	(.21)	(.14)	(.11)	(.10)	(.10)	(.13)	(.15)	(.10)	(.16)	(.10)
10	11.30	11.00	11.00	11.00	11.00	11.00	11.00	11.00	11.00	11.00
	(.22)	(.13)	(.10)	(.10)	(.10)	(.15)	(.10)	(.10)	(.11)	(.19)

Table 5.15: Final number of target vectors and solver convergence factor for rBAMG applied to shifted 2D Poisson and N = 256.

	64	128	256
Total Cost	152	160	167
(q, ν)	(6,2)	(7,2)	(6,2)
Final number of target vectors	7	8	7
Final convergence factor	.19	.17	.20

Table 5.16: Summary of the optimal total costs of rBAMG for the shifted Poisson problem, N = 64, 128, and 256.

that the total cost as measured in work unit is fairly insensitive the size of the grid.

5.2.2 Shifted Gauge Laplacian

This section extends the results to a shifted two-dimensional gauge Laplacian (c.f., [24]). The unshifted matrix represents a stencil of the same form as the 5-point Laplacian, but it instead involves unit random complex numbers in the off-diagonal (more precisely, numbers of form $\frac{e^{i\theta}}{h^2}$ on the off-diagonal and $\frac{4}{h^2}$ on the diagonal), and it corresponds to a uniform doubly periodic grid. We then shifted the matrix so that the smallest eigenvalue is h^2, making the condition number $O(h^{-4})$. This is a very challenging problem on which all classical matrix solvers and conventional multigrid methods are unacceptably slow. The principle difficulty with the gauge Laplacian is that algebraically smooth error tends to have very oscillatory geometric character. This provides a perfect opportunity for adaptive/bootstrap AMG methods to demonstrate their capabilities in the automatic determination of representative smooth error components.

For comparison to the adaptive scheme based on a single test vector, we add a first row to each table that shows the results of applying the adaptive AMG scheme (αAMG) developed in [15] to the shifted gauge Laplacian. This adaptive scheme is similar to rBAMG with $q = \nu = 1$, except that the number of target vectors is not allowed to increase beyond one in subsequent adaptive iterations. (Any error produced in an adaptive iteration is instead used just to correct the current target vector.) Note that the final αAMG convergence factor is always greater than ρ_{good}, so one near null space component is not enough for this problem. Note also that the optimal results depicted in Table 5.23 show much more dependence on N, in contrast to the simpler shifted Laplacian results in Table 5.16. Figure 5.2.2 gives a clearer picture of these comparative trends.

q/ν	1	2	3	4	5	6	7	8	9	10
αAMG	361	293	286	288	305	293	354	299	323	351
1	308	266	286	288	255	256	244	244	244	253
2	358	262	241	244	239	230	220	238	236	219
3	289	236	232	208	222	231	227	222	240	240
4	294	233	214	205	197	192	219	221	230	215
5	331	211	217	227	226	228	215	255	276	253
6	303	242	198	233	221	231	241	261	272	294
7	322	229	215	241	238	266	271	284	277	300
8	327	255	266	242	289	277	273	319	312	318
9	368	250	263	269	283	299	330	320	326	328
10	410	312	294	289	315	310	328	319	393	354

Table 5.17: Total cost (= setup + solver cost) for rBAMG applied to shifted gauge Laplacian and N = 64.

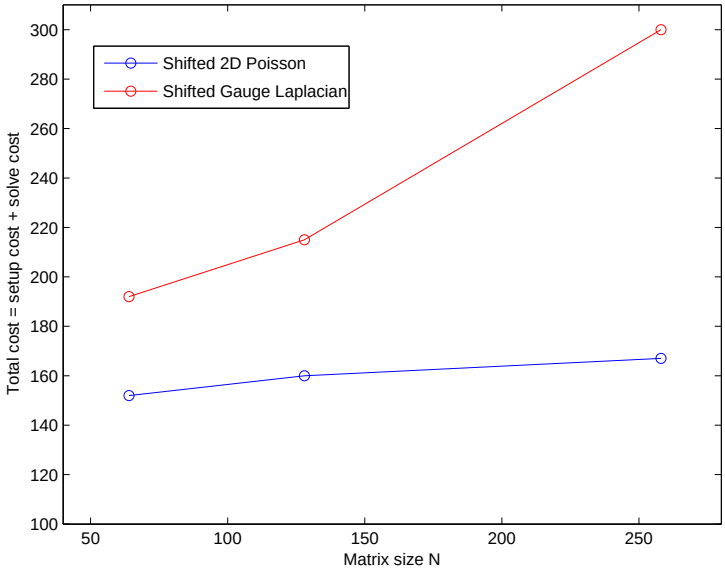

Figure 5.5: Summary of the optimal total costs of rBAMG for the shifted gauge Laplacian problem, N = 64, 128, and 256.

q/ν	1	2	3	4	5	6	7	8	9	10
αAMG	5.60 (.63)	4.90 (.61)	4.20 (.60)	4.50 (.60)	4.30 (.63)	5.20 (.60)	5.50 (.64)	5.60 (.59)	4.70 (.64)	5.10 (.64)
1	5.00 (.39)	4.90 (.31)	4.70 (.35)	4.10 (.44)	4.30 (.33)	4.00 (.37)	3.90 (.34)	3.80 (.35)	3.70 (.35)	3.80 (.35)
2	5.50 (.50)	5.40 (.31)	4.90 (.32)	4.80 (.31)	4.50 (.35)	4.20 (.35)	3.90 (.36)	4.20 (.34)	4.10 (.34)	3.60 (.37)
3	6.50 (.35)	5.60 (.31)	5.40 (.31)	4.90 (.30)	5.00 (.30)	5.00 (.30)	4.80 (.32)	4.60 (.32)	4.80 (.31)	4.70 (.32)
4	7.30 (.31)	6.20 (.33)	5.80 (.30)	5.50 (.32)	5.30 (.30)	5.10 (.30)	5.40 (.29)	5.30 (.30)	5.30 (.30)	5.00 (.30)
5	7.80 (.41)	6.60 (.33)	6.60 (.31)	6.60 (.29)	6.50 (.29)	6.40 (.28)	6.10 (.28)	6.50 (.28)	6.60 (.29)	6.20 (.29)
6	8.80 (.31)	7.90 (.31)	7.10 (.33)	7.50 (.28)	7.20 (.28)	7.20 (.27)	7.20 (.27)	7.30 (.27)	7.30 (.27)	7.40 (.28)
7	9.40 (.38)	8.50 (.30)	8.20 (.29)	8.40 (.27)	8.20 (.28)	8.40 (.26)	8.30 (.26)	8.30 (.26)	8.10 (.27)	8.20 (.26)
8	10.50 (.33)	9.60 (.31)	9.60 (.29)	9.20 (.29)	9.50 (.26)	9.30 (.25)	9.10 (.26)	9.40 (.26)	9.20 (.26)	9.10 (.28)
9	11.70 (.31)	10.40 (.30)	10.40 (.27)	10.30 (.26)	10.30 (.26)	10.30 (.24)	10.40 (.25)	10.20 (.25)	10.10 (.26)	10.00 (.25)
10	12.80 (.30)	11.80 (.30)	11.50 (.28)	11.30 (.27)	11.40 (.24)	11.20 (.25)	11.20 (.25)	11.00 (.24)	11.40 (.25)	11.00 (.25)

Table 5.18: Final number of target vectors and solver convergence factor for rBAMG applied to shifted gauge Laplacian and N = 64.

q/ν	1	2	3	4	5	6	7	8	9	10
αAMG	463	519	444	456	366	461	465	379	393	444
1	395	280	278	280	281	290	304	280	277	285
2	399	324	265	255	259	268	274	272	304	291
3	348	264	253	240	249	244	238	247	250	263
4	346	255	244	255	267	244	231	229	236	241
5	301	269	284	240	250	215	233	251	272	262
6	334	290	256	275	252	252	236	265	276	270
7	351	297	272	273	302	252	293	289	291	348
8	375	330	283	321	307	303	275	324	319	308
9	403	376	342	314	321	340	309	363	368	359
10	443	371	366	369	370	379	360	390	356	375

Table 5.19: Total cost (= setup + solver cost) for rBAMG applied to shifted gauge Laplacian and N = 128.

q/ν	1	2	3	4	5	6	7	8	9	10
αAMG	6.30	5.40	6.30	6.50	5.20	6.00	6.20	6.70	6.10	6.60
	(.78)	(.79)	(.75)	(.76)	(.72)	(.78)	(.77)	(.72)	(.73)	(.76)
1	5.40	5.20	5.00	4.90	4.80	4.80	4.00	4.30	4.10	4.10
	(.51)	(.30)	(.32)	(.33)	(.33)	(.33)	(.47)	(.39)	(.40)	(.42)
2	5.70	5.40	5.10	4.90	5.00	5.00	5.00	4.80	4.30	4.20
	(.57)	(.42)	(.36)	(.34)	(.31)	(.33)	(.32)	(.34)	(.44)	(.45)
3	6.90	5.90	5.60	5.40	5.40	5.20	5.00	4.80	4.80	5.00
	(.38)	(.36)	(.36)	(.32)	(.32)	(.32)	(.33)	(.37)	(.37)	(.33)
4	7.20	6.60	6.20	6.30	6.30	5.80	5.50	5.40	5.30	5.30
	(.42)	(.31)	(.34)	(.30)	(.32)	(.34)	(.33)	(.31)	(.34)	(.33)
5	8.10	7.50	7.10	6.60	6.70	6.10	6.30	6.40	6.50	6.30
	(.32)	(.32)	(.40)	(.34)	(.34)	(.34)	(.31)	(.30)	(.33)	(.30)
6	8.90	8.50	7.80	7.80	7.50	7.40	7.10	7.30	7.30	7.10
	(.39)	(.30)	(.34)	(.34)	(.32)	(.30)	(.29)	(.30)	(.30)	(.30)
7	9.90	9.30	8.90	8.70	8.90	8.20	8.50	8.30	8.20	8.60
	(.38)	(.30)	(.29)	(.30)	(.28)	(.29)	(.29)	(.29)	(.29)	(.30)
8	11.00	10.40	9.70	9.70	9.60	9.50	9.10	9.40	9.20	9.00
	(.34)	(.30)	(.33)	(.40)	(.33)	(.28)	(.28)	(.29)	(.30)	(.29)
9	12.00	11.30	11.00	10.50	10.60	10.60	10.20	10.50	10.40	10.20
	(.33)	(.37)	(.32)	(.36)	(.28)	(.29)	(.28)	(.29)	(.29)	(.29)
10	12.70	12.30	12.10	11.90	11.80	11.70	11.40	11.50	11.10	11.10
	(.41)	(.29)	(.28)	(.32)	(.28)	(.27)	(.30)	(.28)	(.28)	(.29)

Table 5.20: Final number of target vectors and solver convergence factor for rBAMG applied to shifted gauge Laplacian and N = 128.

q/ν	1	2	3	4	5	6	7	8	9	10
αAMG	443	432	455	440	426	438	417	437	443	408
1	440	365	327	375	338	371	417	367	393	370
2	540	368	365	344	310	316	318	341	318	327
3	445	405	413	363	365	344	310	310	333	327
4	467	370	355	315	315	315	351	370	351	338
5	426	360	347	338	312	331	320	307	327	288
6	468	379	365	371	326	308	310	300	329	348
7	527	393	417	333	335	313	383	342	357	349
8	515	435	427	398	344	346	361	399	348	343
9	602	460	456	392	398	396	387	394	401	377
10	618	530	475	497	402	411	369	411	444	406

Table 5.21: Total cost (= setup + solver cost) for rBAMG applied to shifted gauge Laplacian and N = 256.

q/ν	1	2	3	4	5	6	7	8	9	10
αAMG	5.20	4.80	5.20	4.90	4.80	5.10	4.10	4.70	4.60	3.80
	(.81)	(.81)	(.82)	(.76)	(.82)	(.80)	(.79)	(.87)	(.78)	(.84)
1	6.10	5.70	5.20	4.80	4.90	4.40	3.90	4.70	3.90	4.30
	(.50)	(.42)	(.37)	(.47)	(.40)	(.51)	(.60)	(.46)	(.59)	(.49)
2	4.80	6.60	5.80	6.00	5.20	5.30	5.00	4.90	5.00	5.00
	(.86)	(.37)	(.44)	(.36)	(.38)	(.36)	(.39)	(.41)	(.37)	(.37)
3	8.40	6.70	6.00	5.80	5.70	5.70	5.70	5.50	5.30	5.50
	(.35)	(.46)	(.56)	(.46)	(.46)	(.42)	(.34)	(.36)	(.39)	(.36)
4	8.70	7.20	7.10	6.80	6.60	6.00	6.40	6.00	5.90	5.90
	(.43)	(.44)	(.39)	(.34)	(.35)	(.40)	(.39)	(.36)	(.42)	(.39)
5	9.30	7.90	8.10	7.40	7.30	7.10	7.10	6.80	6.60	6.30
	(.35)	(.40)	(.32)	(.36)	(.33)	(.37)	(.33)	(.33)	(.37)	(.35)
6	9.90	9.20	9.00	8.50	8.20	7.80	7.70	7.40	7.60	7.70
	(.44)	(.34)	(.30)	(.36)	(.31)	(.33)	(.32)	(.34)	(.34)	(.32)
7	11.40	10.10	10.20	9.10	9.00	8.60	8.80	8.60	8.60	8.40
	(.39)	(.33)	(.31)	(.33)	(.32)	(.32)	(.37)	(.32)	(.32)	(.33)
8	11.70	11.20	11.00	10.50	9.80	9.70	9.70	9.90	9.30	9.10
	(.42)	(.32)	(.30)	(.31)	(.32)	(.32)	(.30)	(.30)	(.30)	(.33)
9	13.10	12.10	11.90	11.20	11.10	10.90	10.70	10.60	10.50	10.20
	(.41)	(.33)	(.32)	(.30)	(.30)	(.31)	(.29)	(.29)	(.31)	(.30)
10	13.80	13.40	12.80	12.40	11.90	11.80	11.30	11.50	11.60	11.20
	(.45)	(.30)	(.30)	(.38)	(.30)	(.30)	(.32)	(.30)	(.29)	(.29)

Table 5.22: Final number of target vectors and solver convergence factor for rBAMG applied to shifted gauge Laplacian and N = 256.

	64	128	256
Total Cost	192	215	300
(q, ν)	(4,6)	(5,6)	(6,8)
Final number of target vectors	5.1	6.1	7.4
Final convergence factor	.3	.34	.34

Table 5.23: Least squares approximation of BAMG.

Chapter 6

Conclusions and Future Work

The thesis introduced several variants of adaptive/bootstrap AMG in an attempt to improve performance of the AMG solvers that these methods produce. The two variants who showed the most success in the sense are what we called indirect BAMG (iBAMG) and relaxation-corrected BAMG (rBAMG).

The premise behind iBAMG is to insulate the construction of an effective interpolation operator from poor approximation of algebraically smooth components by using interpolation only to collapse F-F connections in an operator interpolation scheme. This approach is more in the spirit of classical AMG and we show here that it is almost always substantially more effective than the direct approach to interpolation that has typically been used in BAMG.

We developed rBAMG by simply adding scaled residuals of the target vectors to the least-squares principles for the direct BAMG approach. We show an equivalence between iBAMG and rBAMG, which is important because it gives insight into the effectiveness of rBAMG and it allows for all of the existing BAMG machinery (especially coarsening by compatible relaxation) to be used with these improved methods. Our numerical experiments then focus on rBAMG because of its simplicity and wider applicability. In particular, while iBAMG relies on the notion of operator interpolation, rBAMG does not. For example, iBAMG interpolation to an F point rests on a stencil there with a nonzero (diagonal) entry associated with the point itself, but this is not at all a requirement for rBAMG. Indeed, the general rBAMG principle is simply to correct each target vector in the direct least-squares principle by an expression that amounts to applying the relevant

relaxation scheme to that vector. Stating the principle in this way shows that rBAMG can be applied in cases where the stencil at an F point does not even involve the point itself. Future plans include the study of the general applicability of rBAMG to problems of this type, starting with conventional discretizations of Stokes and Navier-Stokes equations with zeros on the diagonal of the matrix associated with the incompressibility condition.

The numerical experiments described in this thesis concentrated on two basic goals. The first was to investigate the effectiveness of the initial rBAMG and BAMG least-squares setup process and how it depends on the number of test vectors, the number of relaxations applied to them, and the dimension of the model problem. We showed that the required amount of work to obtain small convergence factors of the resulting solver increased at a modest rate with problem dimension. The second goals was to study the performance of the full setup process that included subsequent adaptive cycles of the current solver that were used to enhance the initial set of test vectors and thereby improve subsequent least-squares fit of interpolation. An important ingredient of this adaptive approach is the convergence estimation model that we derived. It provided an effective means for judging the convergence quality of the current solver without carrying out the large number of cycles that conventional estimates require. The estimates that this model provided, together with an accurate work estimate we developed, allowed the adaptive scheme to make effective decisions along the way. The numerical results we presented documented the overall performance of the full setup and solver, showing almost optimal complexity for the Poisson case. The results also showed more than linear increase with respect to problem size for the gauge Laplacian. We should note that true optimality (i. e., complexity proportional to problem size) is not to be expected for the overall setup and solution process because the required accuracy of interpolation tends to increase as the problem size increases. Thus, the observed near optimality for the Poisson case is probably a result of studying fairly small problems. Our plan is to extend this study to much more complicated and much larger problems, where we expect the complexity trends to be more apparent.

Another aspect we plan to study is how rBAMG compares with adaptive smooth aggregation.

It is especially important to study comparative performance on problems that arise from systems of partial differential equations. Our main target in this direction are the full two- and four-dimensional systems of quantum chromodynamics.

Bibliography

[1] A. Brandt. Multi-level adaptive solutions to boundary value problems. Math. Comp., 31:333–390, 1977.

[2] A. Brandt. Algebraic multigrid theory: the symmetric case. Applied Mathematics and Computation, 9:23–26, 1986.

[3] A. Brandt. General highly accurate algebraic coarsening. Electronic Transactions on Numerical Analysis, 63:521–539, 1992.

[4] A. Brandt. Multiscale scientific computation: review 2001. In Barth, T.J., Chan, T.F. and Haimes, R. (eds.): Multiscale and Multiresolution Methods: Theory and Application, pages 1–96, 2001.

[5] A. Brandt, J. Brannick, K. Kahl, and I. Livshits. A least squares based algebraic multigrid solver for Hermitian and positive definite systems. 2009. Unpublished manuscript.

[6] A. Brandt, J. Brannick, K. Kahl, and I. Livshits. Bootstrap AMG. 2010. Unpublished manuscript.

[7] A. Brandt, S. McCormick, and J. Ruge. Algebraic multigrid (AMG) for automatic multigrid solution with application to geodetic computations. Technical report, Colorado State University, Fort Collins, Colorado, 1983.

[8] A. Brandt, S. McCormick, and J. Ruge. Algebraic multigrid (AMG) for sparse matrix equations. in sparsity and its applications, D.J, Evans (ed.). 1984.

[9] J. Brannick, D. Brezina, M. Keyes, O. Livine, I. Livshits, S. MacLachlan, T. Manteuffel, S. McCormick, J. Ruge, and L. Zikatanov. Adaptive smoothed aggregation in lattice QCD. In Proceedings of DD16, The 16th International Conference on Domain Decomposition Methods. Springer (to appear).

[10] J. Brannick and R. Falgout. Compatible relaxation and coarsening in algebraic multigrid. SIAM J. Sci. Comp, 32:1393–1416, 2010.

[11] J. Brannick, A. Frommer, K. Kahl, S. MacLachlan, and L. Zikatanov. Adaptive reduction-based multigrid for nearly singular and highly disordered physical systems. Electronic Transactions on Numerical Analysis, 37:276–295, 2010.

[12] M. Brezina, A. Cleary, R. Falgout, V. Henson, J. Jones, T. Manteuffel, S. McCormick, and J. Ruge. Algebraic multigrid based on element interplation(AMGe). SIAM J. Sci. Comp, 22:1570–1592, 2000.

[13] M. Brezina, R. Falgout, S. MacLachlan, T. Manteuffel, S. McCormick, and J. Ruge. Adaptive smoothed aggregation (αSA). SIAM J. Sci. Comp., 25(6):1896–1920, 2004.

[14] M. Brezina, R. Falgout, S. MacLachlan, T. Manteuffel, S. McCormick, and J. Ruge. Adaptive smoothed aggregation (αSA) multigrid. SIAM Rev., 47(2):317–346, 2005.

[15] M. Brezina, R. Falgout, S. MacLachlan, T. Manteuffel, S. McCormick, and J. Ruge. Adaptive algebraic multigrid. SIAM J. Sci. Comp., 27(4):1261–1286, 2006.

[16] W. Briggs, V. Henson, and S. McCormick. Multigrid Tutorial. SIAM, 2000.

[17] A. Cleary, R. Falgout, V. Henson, J. Jones, T. Manteuffel, S. McCormick, G. Miranda, and J. Ruge. Robustness and algorithmic scalability of algebraic multigrid (AMG). SIAM J. Sci. Comp, 21:1886–1908, 2000.

[18] G. Golub and C. Van Loan. Matrix Computations. The Johns Hopkins University Press, 1996.

[19] A. Greenbaum. Iterative Methods for Solving Linear Systems. SIAM, Philadelphia, 1997.

[20] W. Hackbusch. Multi-Grid Methods and Applications. Springer-Verlag, 1985.

[21] V. Henson and P. Vassilevski. Element-free AMGe: General algorithm for compution interpolation weights in amg. SIAM J. Sci. Comp, 23(2):629–650, 2001.

[22] S. MacLachlan. Improving Robustness in Multiscale Methods. PhD thesis, University of Colorad at Boulder, 2004.

[23] S. MacLachlan, T. Manteuffel, and S. McCormick. Adaptive reduction-based AMG. Num. Lin. Alg. Appl, 13(8):599–620, 2005.

[24] S. MacLachlan and C. Oosterlee. Algebraic multigrid solvers for complex-valued matrices. SIAM J. Sci. Comput., 30(3):1548–1571, 2008.

[25] J. Ruge and K. Stüben. Algebraic multigrid (AMG). In *Multigrid Methods*, vol. 5, McCormick SF (ed.). SIAM: Philadelphia, PA., 1986.

[26] U. Trottenberg, C. Oosterlee, and A. Schüller. Multigrid. Academic Press, 2001.

[27] P. Vaněk, M. Brezina, and J. Mandel. Convergence of algebraic multigrid based on smoothed aggregation. Numer. Math., 88:559–579, 2001.

[28] P. Vaněk, J. Mandel, and M. Brezina. Algebraic multigrid by smoothed aggregation for second and fourth order elliptic problems. Computing, 56:179–196, 1996.