Solving PDEs with Multigrid Methods

Scott MacLachlan

maclachl@colorado.edu

Department of Applied Mathematics, University of Colorado at Boulder

Acknowledgments

- This work has been supported by
  - Chancellor's Fellowship, University of Colorado
  - DOE SciDAC TOPS program
  - Center for Applied Scientific Computing at Lawrence Livermore National Laboratory
- This work has been performed in collaboration with
  - Steve McCormick, Tom Manteuffel, John Ruge, and Marian Brezina at CU-Boulder
  - Rob Falgout at CASC
  - David Moulton at LANL.

Outline

- Background & Motivation
- Multigrid Methods
- Self-Correcting Multigrid Methods
- Current and Future Work

The Need for Optimal Linear Solvers

- Significant interest in simulating complex physical systems with features, and hence solutions, that vary on multiple scales
- Accuracy constraints lead to discretizations with tens of millions, or even billions, of degrees of freedom (DOFs)
  - 3D Tsunami Model: 200 million cells
  - Transport: 500 million to 1 billion degrees of freedom
- Without optimal methods, solving three-dimensional problems can be prohibitively expensive

Properties of Matrices

- We consider (primarily) discretizations of the underlying continuum models (differential equations) via finite elements or finite differences
- The matrices from these discretizations tend to be sparse and ill-conditioned
- The matrices inherit properties of the continuum model (e.g. symmetry, definiteness)

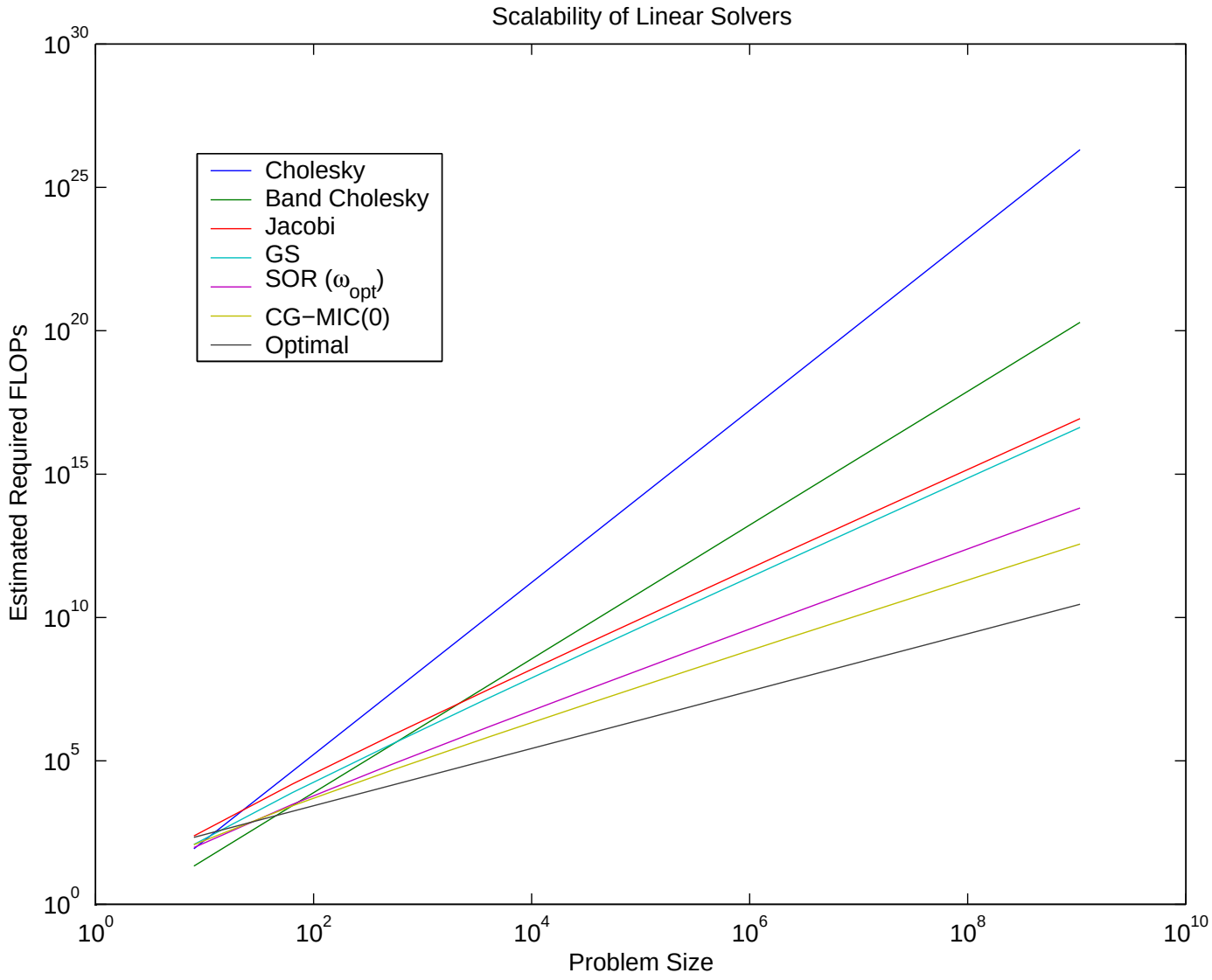
Stationary Iterative Methods

- Stationary iterative methods choose approximations $B \approx A^{-1}$ and iterate using the error equation
- If $Ae = A(x - x^k) = b - Ax^k$, then $x^{k+1} = x^k + B(b - Ax^k)$
- The Jacobi iteration chooses B to be the diagonal of A
- The Gauss-Seidel iteration chooses B to be the lower-triangular part of A
- SOR chooses B from within a one-parameter family to try and minimize $\rho(I - BA)$

Krylov Methods

- Krylov methods find the optimal approximation to the solution in a given subspace
- Iteratively increase the size of the subspace to improve accuracy
- These methods are non-stationary: the results of an iteration affect later iterations

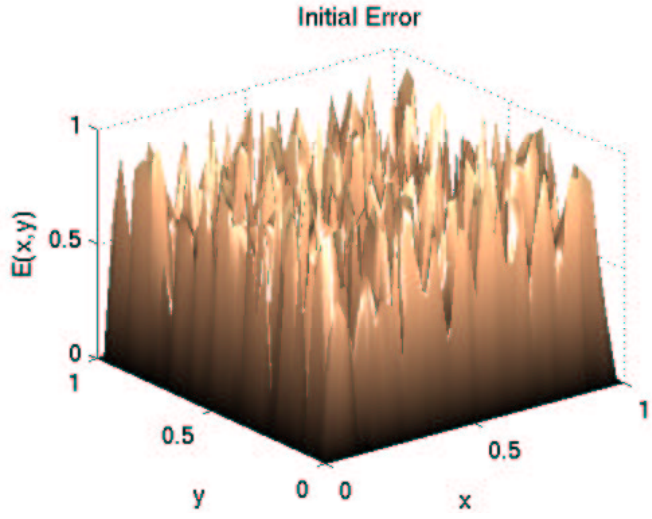
Classical Methods do not Suffice

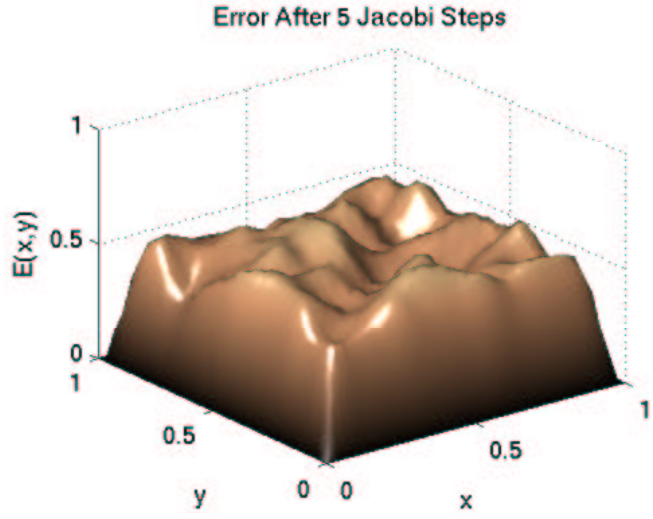


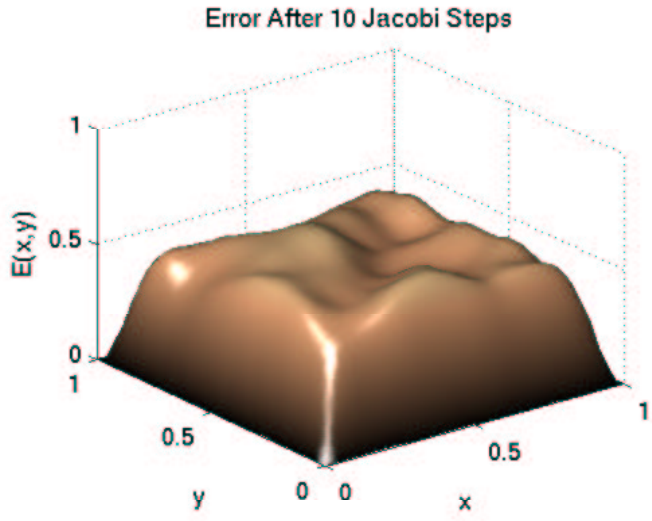
Stationary Iterative Methods . . .

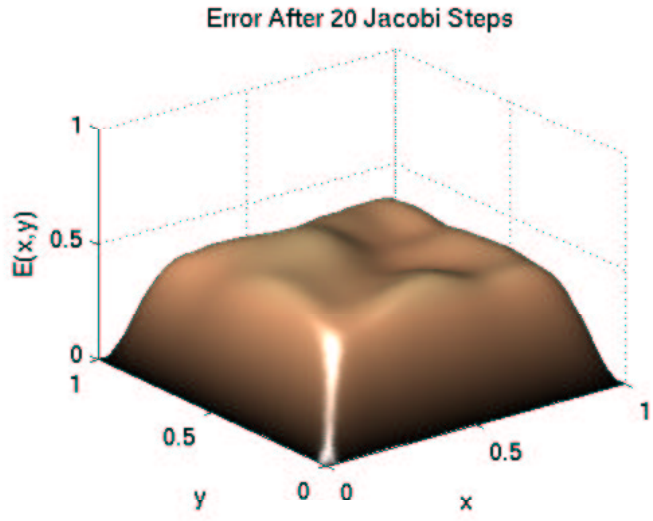
- The Jacobi and Gauss-Seidel iterations do converge for FE discretizations of elliptic operators, but are not require $O(N^{\frac{5}{3}})$ operations for 3-D problems
- These methods do, however, resolve some components much faster than others
- For example, for the Laplacian, it is the geometrically smoothest components of the solution that are the slowest to be resolved
- For this reason, Jacobi and Gauss-Seidel are often called smoothers - they smooth the error in the approximation

Smoother Performance









Complementing Relaxation

- If the error left after relaxing is smooth, it can be accurately represented using fewer degrees of freedom
- Problems with fewer degrees of freedom can be solved with less effort
- Error which appears smooth across many degrees of freedom is more oscillatory when represented on fewer degrees of freedom
- We choose to represent such error using a subset of the fine-grid degrees of freedom

Multigrid Basics

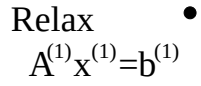
Multigrid Methods achieve optimality through complementarity

Multigrid Basics

Multigrid Methods achieve optimality through complementarity

Multigrid Components

- Relaxation



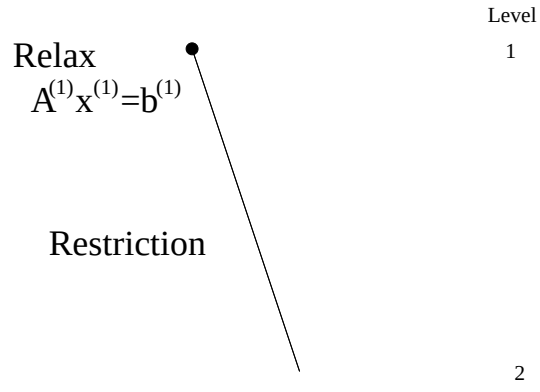
- Use a smoothing process (such as Gauss-Seidel) to eliminate oscillatory errors
- Remaining error satisfies $Ae = r$

Multigrid Basics

Multigrid Methods achieve optimality through complementarity

Multigrid Components

- Relaxation
- Restriction



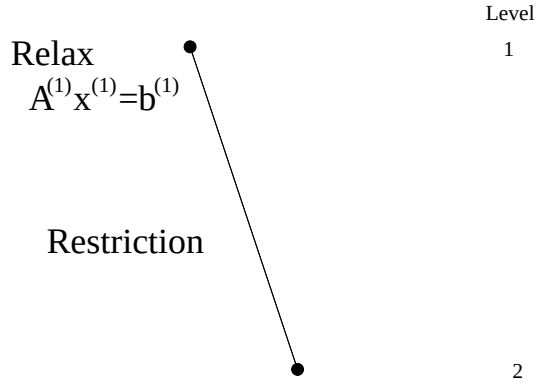
- Need to transfer residual to coarse-grid
  - use Restriction operator

Multigrid Basics

Multigrid Methods achieve optimality through complementarity

Multigrid Components

- Relaxation
- Restriction
- Coarse Grid Correction



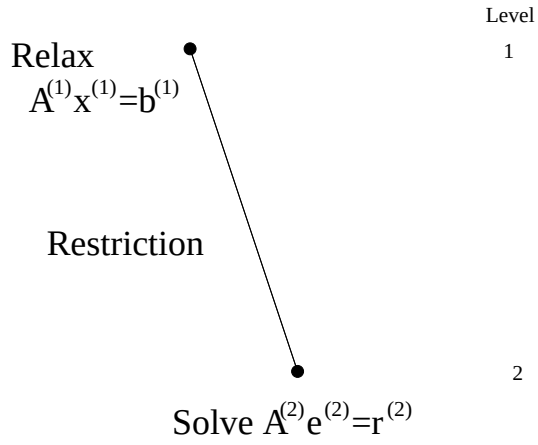
- Use coarse grid correction to eliminate smooth errors

Multigrid Basics

Multigrid Methods achieve optimality through complementarity

Multigrid Components

- Relaxation
- Restriction
- Coarse Grid Correction



- To solve for error on coarse-grid, use residual equation

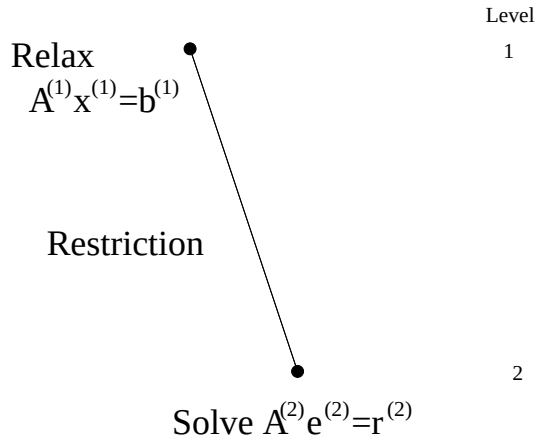
$$A^{(2)}e^{(2)} = r^{(2)}$$

Multigrid Basics

Multigrid Methods achieve optimality through complementarity

Multigrid Components

- Relaxation
- Restriction
- Coarse Grid Correction



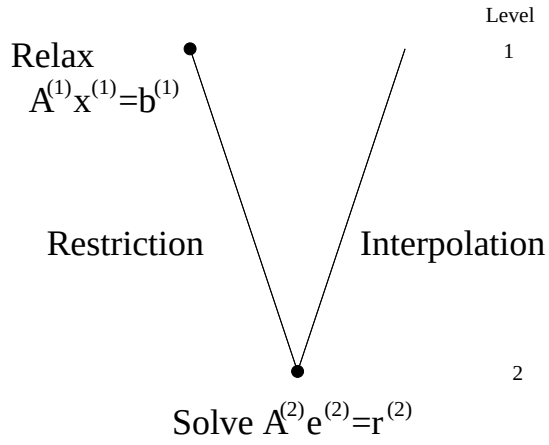
- Solving on coarse-grid requires an operator on this grid which well approximates the fine-grid operator
- The coarse-grid operator can be formed by rediscretization or using a variational principal

Multigrid Basics

Multigrid Methods achieve optimality through complementarity

Multigrid Components

- Relaxation
- Restriction
- Coarse Grid Correction
- Interpolation



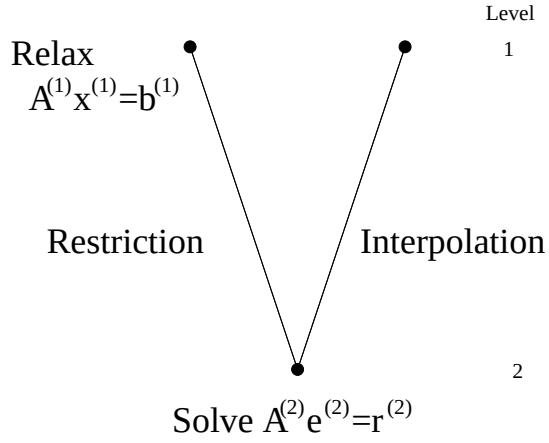
- Need to transfer correction to fine-grid
  - use Interpolation (Prolongation) operator
- Often pick a form of interpolation (P) and take restriction $R = P^T$ (theoretical benefits)

Multigrid Basics

Multigrid Methods achieve optimality through complementarity

Multigrid Components

- Relaxation
- Restriction
- Coarse Grid Correction
- Interpolation
- Relaxation

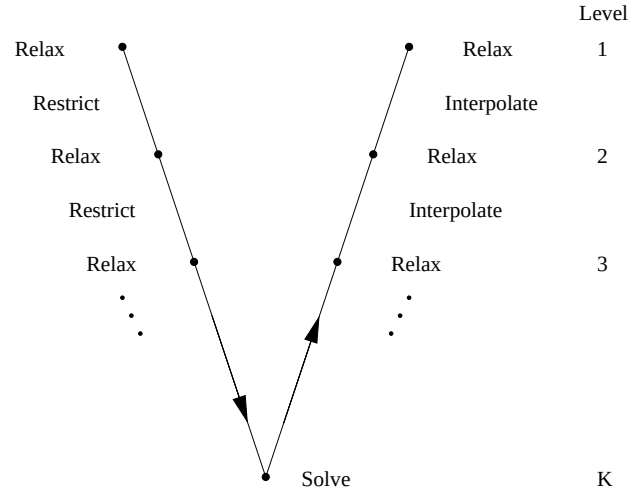


Multigrid Basics

Multigrid Methods achieve optimality through complementarity

Multigrid Components

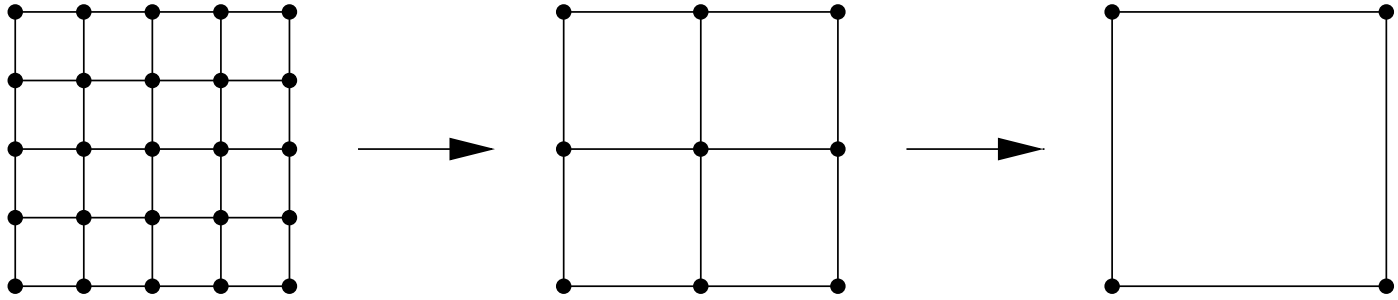
- Relaxation
- Restriction
- Coarse Grid Correction
- Interpolation
- Relaxation



Obtain optimal efficiency through recursion

Geometric Multigrid

- Multigrid algorithms can be broadly classified by how they pick their coarse grids
- If we start with a geometrically regular grid, coarse grids can easily be chosen



Geometric Multigrid

- Interpolation that is accurate for geometrically smooth functions is easy to choose
- Can use linear/bilinear/trilinear averaging to get values at fine-grid points that are not also coarse-grid points
- Restriction can be chosen either by simply taking the fine-grid values at coarse-grid points (injection), or as the transpose of interpolation
- Coarse grid equations can be chosen by rediscretizing the PDE on the coarser grid or . . .

Variational Multigrid

- Multigrid with $R = P^T$ and $A_c = RAP$ is called a *variational formulation*
- Terminology comes from minimization form of $Ax = b$:

$$F(v) = \frac{1}{2}\langle Av, v\rangle - \langle b, v\rangle$$

$$x = \arg\min_{v \in \mathcal{H}} F(v)$$

- Given an approximation v to the solution on the fine level, it can be shown that the optimal coarse grid correction Pw solves

$$(P^T AP)w = P^T(b - Av)$$

Nested Multigrid

- Properties of Multigrid are most easily understood in the case of nested grids
- This means that not only each coarse grid a subset of the finer grid, but that the coarse-grid operators are also appropriately nested
- To have this, consider a finite element framework

Nested Framework

- Consider the bilinear form $A(\cdot,\cdot)$ on a finite-dimensional space M
- Let $M \equiv M_J \supset M_{J-1} \supset \cdots \supset M_1$
- Define

$$A_k : M_k \to M_k \quad \text{s.t. } A(\psi,\phi) = \langle A_k\psi, \phi\rangle \qquad \forall \psi,\phi \in M_k$$

$$P_k : M_J \to M_k \quad \text{s.t. } A(P_k\psi,\phi) = A(\psi,\phi) \qquad \forall \psi \in M_J, \phi \in M_k$$

$$Q_k : M_J \to M_k \quad \text{s.t. } \langle Q_k\psi,\phi\rangle = \langle \psi,\phi\rangle \qquad \forall \psi \in M_J, \phi \in M_k$$

- Consider on each grid a smoother, $R_k : M_k \to M_k$, with $R_1 = A_1^{-1}$

Nested Approximate Inverses

- Want to construct the approximate inverse B to $A \equiv A_J$ on $M \equiv M_J$
- Do this recursively, by defining approximate inverses B_k for A_k
- On the coarsest grid we perform an exact solve, so $B_1 = A_1^{-1}$
- For $g \in M_k$,
  - Take $x_1 = R_k^T g$
  - Take $x_2 = x_1 + B_{k-1} Q_{k-1} (g - A_k x_1)$
  - Define $B_k g = x_2 + r_k (g - A_k x_2)$

Stationarity

- Now, consider the grid k error-propagation matrix

$$I - B_k A_k$$

- Can show that

$$I - B_k A_k P_k = (I - R_k A_k P_k)(I - B_{k-1} A_{k-1} P_{k-1})(I - R_k A_k P_k)^T$$

- So, taking $T_k = R_k A_k P_k$ have

$$I - BA = (I - T_J)(I - T_{J-1}) \cdots (I - T_1)(I - T_1^T) \cdots (I - T_J^T)$$

- So, nested multigrid is a stationary linear iteration

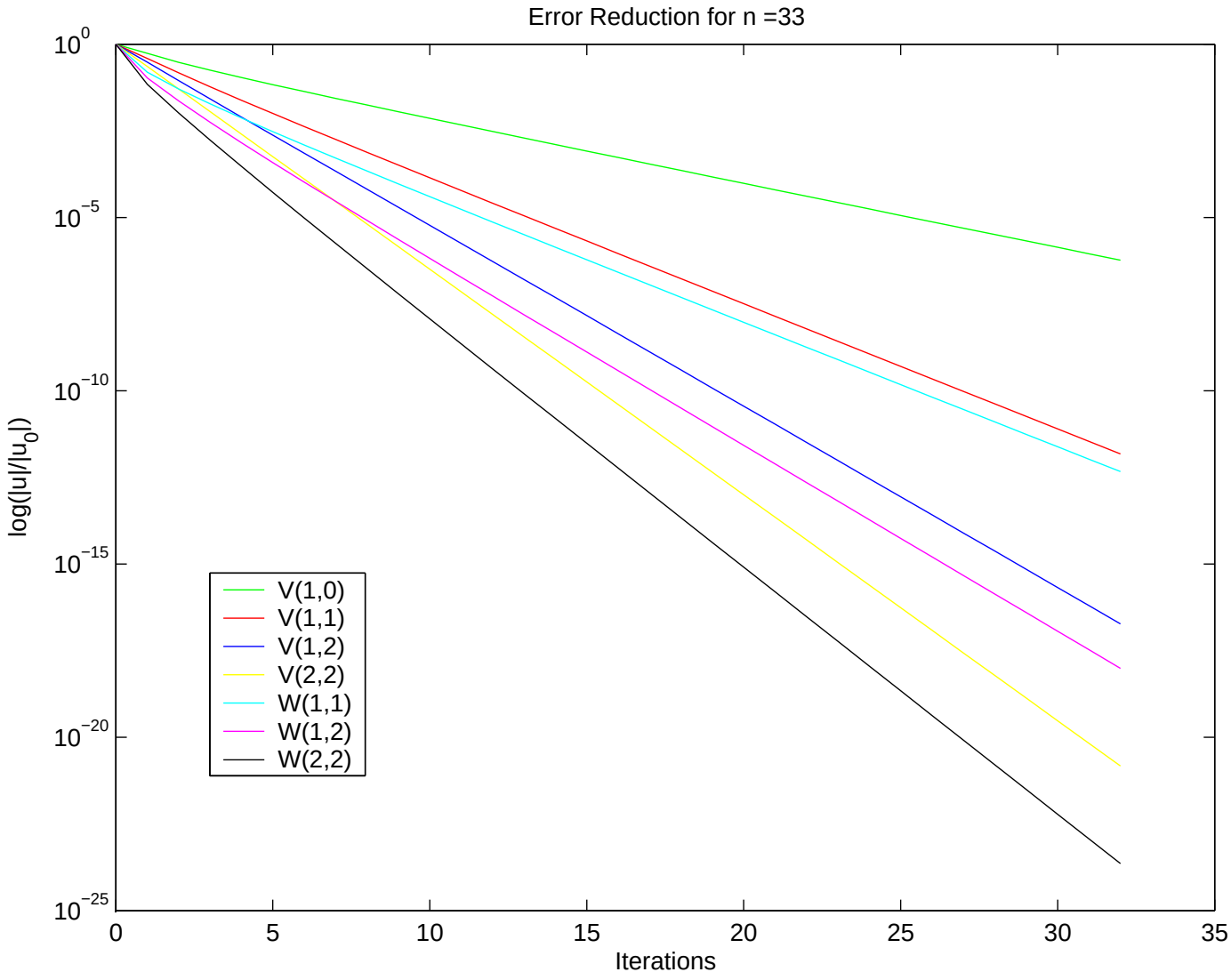
Convergence Theory

- For finite elements, the above theory leads to an optimal convergence result
- For a bounded, open, connected domain $\Omega \in \mathbb{R}^n$ with a smooth enough boundary $\partial\Omega$ and an H^1-elliptic bilinear form $A(\cdot,\cdot)$, the nested multigrid method using Gauss-Seidel relaxation converges (in the norm induced by $A(\cdot,\cdot)$) in a fixed, finite number of iterations

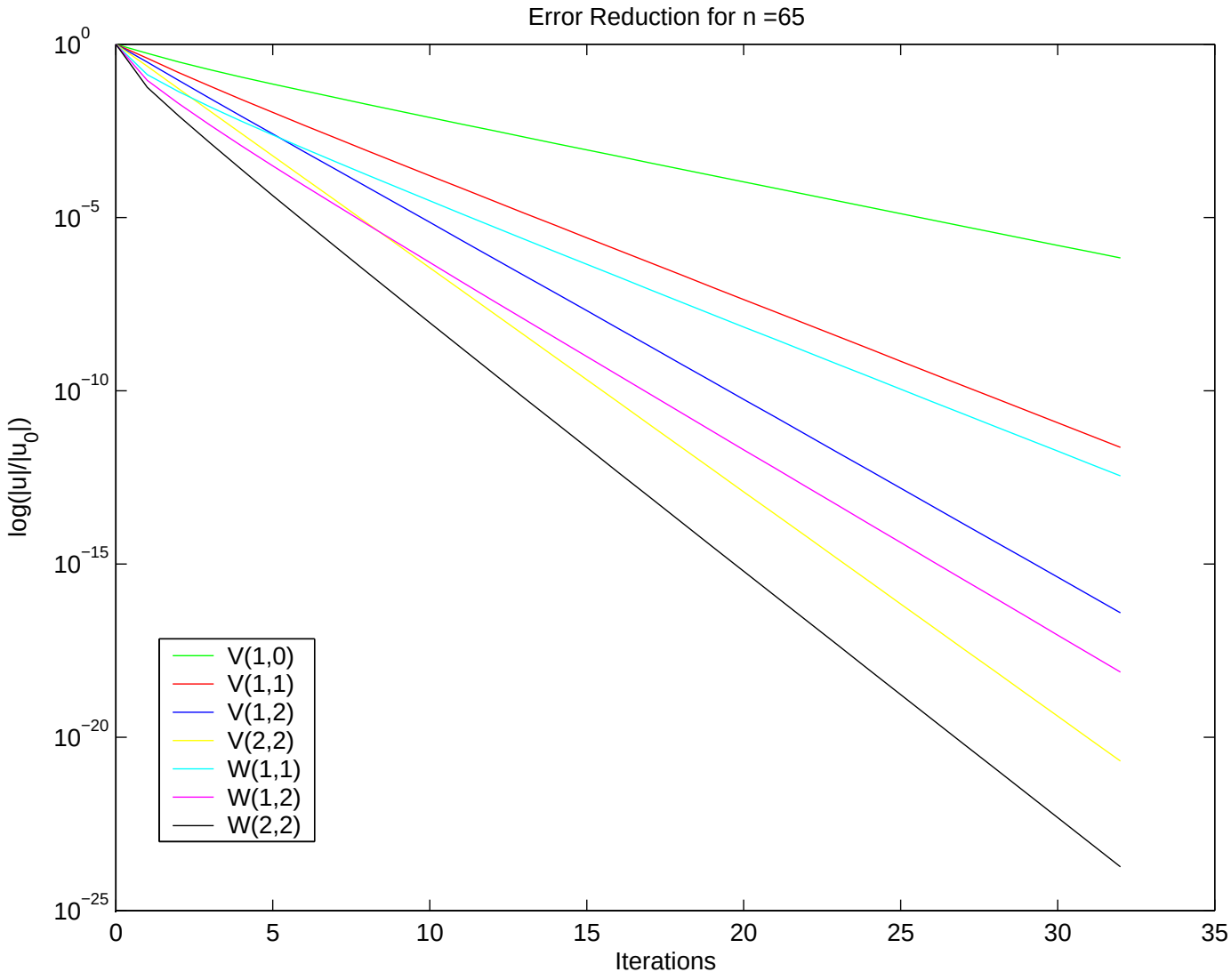
$$0 \leq A((I - B_J A_J)v, v) \leq (1 - \tfrac{1}{C})A(v, v)$$

- Moreover, each iteration has cost bounded by $O(N)$.
- Fourier analysis gives similar results for finite difference discretizations

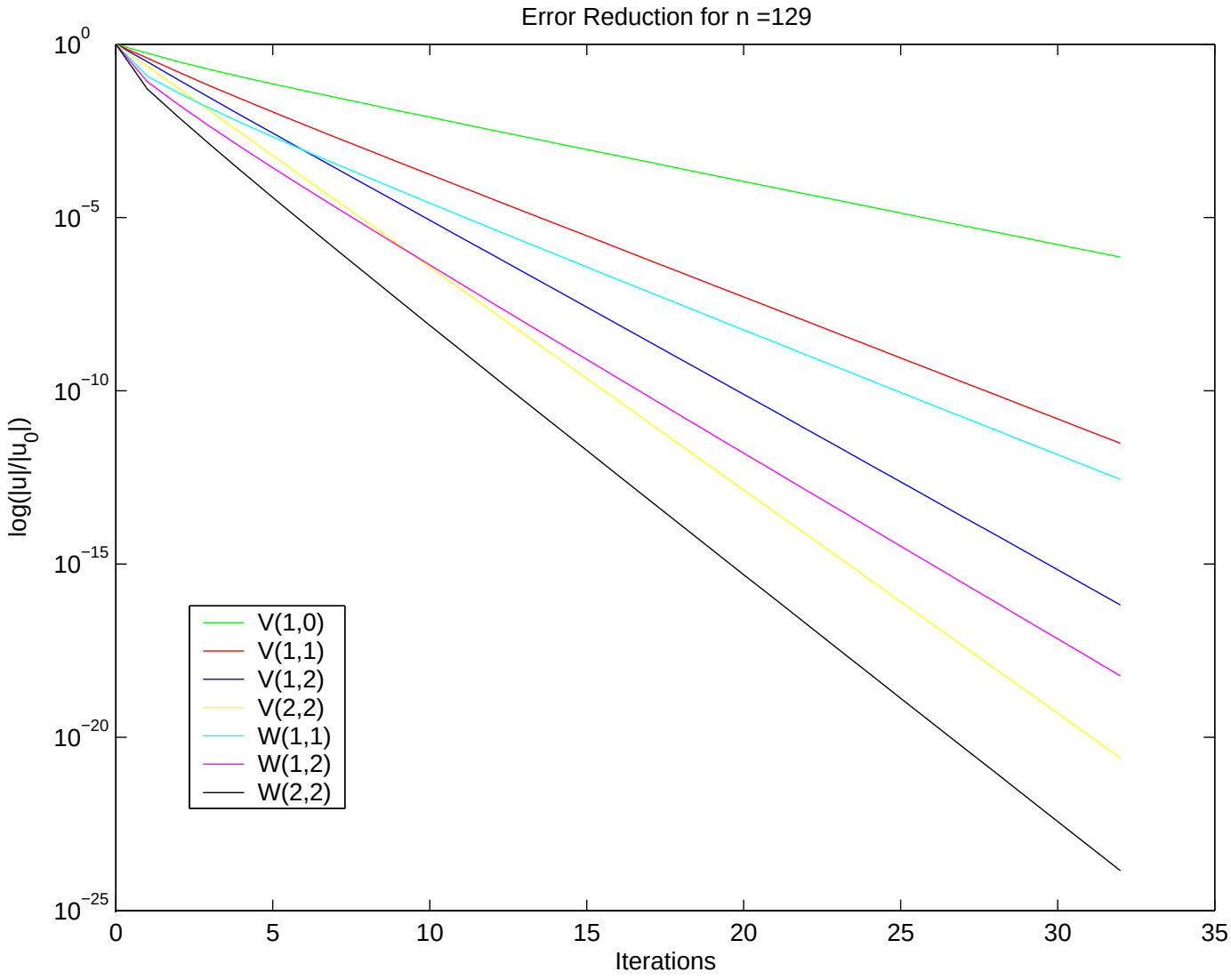
Numerical Results



Numerical Results



Numerical Results



Complications

- Difficult to work out appropriate interpolation for arbitrary geometries
- Some problems don’t have associated geometry (e.g. graph problems)
- Linear interpolation is not optimal across material boundaries (discontinuities in PDE coefficients)
- Linear interpolation is inefficient in cases of strong anisotropy or convection

Philosophy

- All of the above problems can be solved by tweaking the standard, geometric multigrid algorithm
- Different smoothers and different interpolations can be used (although theory may no longer hold)
- Each problem requires its own tuning
- Instead, we concentrate on developing an algorithm which is nearly-optimal on a larger number of problems

Algebraic Multigrid

- In the absence of geometric information, choices can be made based on algebraic information
- Interpolation and coarse grids must be chosen based on the ability to interpolate a suitable correction
- Coarse grid operators must be chosen based on the fine-grid operator - Galerkin coarsening may be the most natural choice

Smoothness

- Without geometric information, we can't talk about a vector being "smooth" in the same sense
- We define a vector, e, to be algebraically smooth if it is slow to be reduced by relaxation on $Ae = 0$
- For a stationary iteration $I - BA$, this means that $(I - BA)e \approx e$, or $BAe \ll e$ in some measure
- In practice, ask that $\|(I - BA)e\|_A \approx \|e\|_A$, so ask that

$$\langle BAe, Ae \rangle \ll \langle e, Ae \rangle$$

- In general, we think of e as being algebraically smooth if $Ae \ll e$ in some measure

Influence and Dependence

- Classical (Ruge-Stueben) AMG focuses on how one gridpoint affects another
- Two gridpoints, i and j are said to be strongly connected if a_{ij} is large
- In particular, for fixed $\theta < 1$, we say i strongly influences j if

$$|a_{ij}| > \theta \max_{k \neq j} |a_{kj}|$$

- We say i strongly depends on j if

$$|a_{ij}| > \theta \max_{k \neq i} |a_{ik}|$$

Coarsening Heuristics

- An good choice of a coarse grid is one which can be effectively used to complement relaxation
- That is, we want to choose a coarse grid to allow us to correct the algebraically smooth components on the fine grid
- Ideally, to interpolate to a point i, we would want to have values at all points that it strongly depends on
- In practice, this would yield far too many coarse-grid points
- Instead, we say that for each point j that strongly influences i, either j is a coarse grid point or it is itself strongly dependent on one coarse-grid neighbour of i

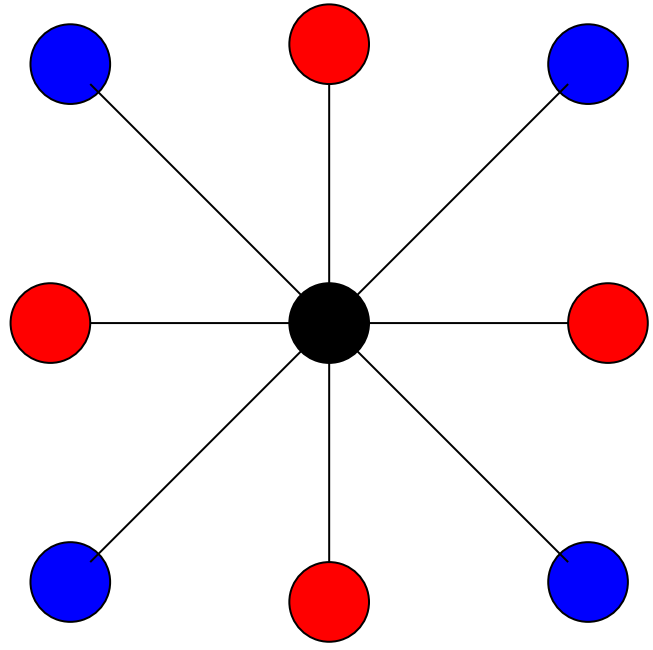
Coarsening Heuristics

- We must also, however, balance the desire for a good interpolation with the need to have a small coarse-grid
- To do this, we insist that the set of coarse points is a maximal subset of the fine-grid such that no coarse-grid point strongly depends on another coarse-grid point
- Implementing these heuristics is accomplished using a colouring algorithm

Defining Interpolation

- For each fine-grid point, i, we want to interpolate its values from neighbouring coarse-grid points

N_i, the Neighbourhood of i



Fine Grid Points

Coarse Grid Points

Defining Interpolation

- For each fine-grid point, i, we want to interpolate its values from neighbouring coarse-grid points
- We consider an interpolation operator that must be accurate for algebraically smooth components, so we start by considering $(Ae)_i = 0$, or

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

- Must get rid of connections to points $k \in F_i$

Defining Interpolation

- Points $k \in F_i$ can be either strongly or weakly connected to i
- If k is weakly connected, it isn't important in interpolation, so collapse to the diagonal (i.e. consider $e_k \approx e_i$)
- If k is strongly connected, then we've ensured it is strongly dependent on something in C_i
- So, approximate e_k by a weighted average of $e_j, j \in C_i$

$$e_k \approx \frac{\sum_{j \in C_i} a_{kj} e_j}{\sum_{j \in C_i} a_{kj}}$$

Theoretical Aspects

- The AMG iteration is still stationary
- Relies implicitly on properties of M-matrices or positive-type matrices
- Convergence theory much harder
- Still a $O(N)$ or $O(N \log N)$ solver

Improvements

- Resulting algorithm can easily handle jumps in coefficients
- No need to know underlying geometry
- Can be adapted to handle anisotropy
- Can be modified to handle more complicated problems, e.g. Elasticity, Stokes Flow, Maxwell's Equations, Hyperbolic PDEs, . . .

Numerical Results

- We start with 2 test problems on $[0,1]^2$, both from bilinear FE discretizations
- Problem 1 is Poisson with pure Dirichlet Boundary Conditions
- Problem 2 is $-\nabla \cdot D(x,y)\nabla p(x,y) = 0$ with Dirichlet BCs on the left and right and Neumann BCs on top and bottom, and

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

Numerical Results

Convergence Factors for AMG

h	Problem 1	Problem 2
$1/32$	0.09	0.14
$1/64$	0.10	0.13
$1/128$	0.14	0.16
$1/256$	0.13	0.15
$1/512$	0.15	0.21

Complications

- Each new type of problem requires a new adaptation
- Coupled Systems become complicated - should tune AMG to each piece of the system
- Very hard to predict what tuning will be necessary
- Many knobs to turn

AMG Assumptions

- Algebraic Multigrid methods attempt to mimic geometric methods in their choices of interpolation operators and coarse grids
- Typically use a fixed, pointwise relaxation scheme
- Classical (Ruge-Stueben) AMG assumes that algebraically smooth error varies slowly along strong connections
- This is equivalent to assuming that algebraically smooth error is essentially (locally) constant

Importance of Interpolation

- Complementarity is key in multigrid - error components that are not quickly reduced by relaxation must be reduced by coarse-grid correction
- A component can only be corrected from the coarse-grid if it is properly interpolated from that grid
- Interpolation must be most accurate for components that relaxation is slowest to resolve

Choosing Interpolation

- Seek to define interpolation to fit an algebraically smooth vector
- Algebraic smoothness means

$$(Ae)_i \approx 0$$

$$\text{or } a_{ii}e_i \approx -\sum_{j \in N_i} a_{ij}e_j$$

$$= -\sum_{j \in C_i} a_{ij}e_j - \sum_{k \in F_i} a_{ik}e_k$$

- To define interpolation, need to collapse connections from F_i to C_i

Choosing Interpolation . . .

- Seek to define interpolation to fit an algebraically smooth vector
- If $k \in F_i$ is connected to a set of $j \in C_i$, we want to write

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

- Then, using the definition of algebraic smoothness, we have

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

$$= - \sum_{j \in C_i} \left(a_{ij} + \sum_{k \in F_i} a_{ik} w_{kj} \right) e_j$$

Choosing w_{kj}

- If we have a vector, $x^{(1)}$, such that $(Ax^{(1)})_k \approx 0$ and so

$$a_{kk}x_k^{(1)} = -\sum_{j \in C_i} a_{kj}x_j^{(1)} - \sum_{j \notin C_i} a_{kj}x_j^{(1)}$$

- Eliminate extra terms by replacing matrix entry a_{kk} with arbitrary d_{kk}

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

Choosing w_{kj} ...

- Taking the value of d_{kk} given here, we can write

$$x_k^{(1)} = -\sum_{j \in C_i} \frac{a_{kj}}{d_{kk}} x_j^{(1)} = \sum_{j \in C_i} \frac{a_{kj} x_k^{(1)}}{\sum_{j' \in C_i} a_{kj'} x_{j'}^{(1)}} x_j^{(1)}$$

- Use this formula to collapse all algebraically smooth error

$$e_k = \sum_{j \in C_i} \left(\frac{a_{kj} x_k^{(1)}}{\sum_{j' \in C_i} a_{kj'} x_{j'}^{(1)}} \right) e_j = \sum_{j \in C_i} w_{kj} e_j$$

Adaptive Interpolation

So, we define interpolation to a fine grid point i as

$$e_i = -\sum_{j \in C_i} \frac{a_{ij} + \sum_{k \in F_i} a_{ik} w_{kj}}{a_{ii}} e_j$$

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

Relation to Ruge-Stueben

- Ruge-Stueben AMG takes $x^{(1)} = 1$
- Substituting this into our interpolation formula gives

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

- This is the same as the AMG strong-connection-only interpolation formula

Scaling Invariance

- Combining our interpolation with pointwise relaxation leads to an algorithm that is nearly insensitive to any diagonal scaling
- In particular, if A is scaled to DAD, and $x^{(1)}$ is scaled to $D^{-1}x^{(1)}$, then we achieve the same convergence rates for the scaled problem as for the unscaled problem
- Difficulty lies in generating the scaled vector $D^{-1}x^{(1)}$

Determining $x^{(1)}$

- Choosing a good interpolation operator requires a good approximation, $x^{(1)}$, to the algebraically-smoothest vector of a given matrix A
- Such an approximation could be determined by sufficient relaxation on a random initial guess with a zero right-hand side
- In practice, this requires too much computation to be feasible
- Instead, we use preliminary V-cycles to accelerate the exposure of components for which $Ax \approx 0$

Test Problems

- We start with 2 test problems on $[0,1]^2$, both from bilinear FE discretizations
- Problem 1 is Poisson with pure Dirichlet Boundary Conditions
- Problem 2 is $-\nabla \cdot D(x,y)\nabla p(x,y) = 0$ with Dirichlet BCs on the left and right and Neumann BCs on top and bottom, and

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

Test Problems

- The second pair of problems come from diagonally scaling Problems 1 and 2
- To scale, we use the node-wise scaling function

$$1 + \sin(547\pi x_i)\sin(496\pi y_j) + 10^{-7}$$

- This function gives variable scaling on each node, but does not change its character with h

Numerical Results

- Coarse grids are chosen geometrically, based on full-coarsening
- Coarse grid operators are determined by the Galerkin condition.
- Setup involves creation of a preliminary V-cycle, and improvement on that cycle
- Compute asymptotic convergence factor, then use this to estimate number of V(1,1)-cycles needed to reduce error by 10^{-6}
- From number and cost of cycles ($\frac{8}{3}$ work units), can estimate total cost of solution stage

AMG-Equivalent Results

- By fixing $x^{(1)} = 1$, we can generate results indicative of AMG's performance

Work Units for standard AMG

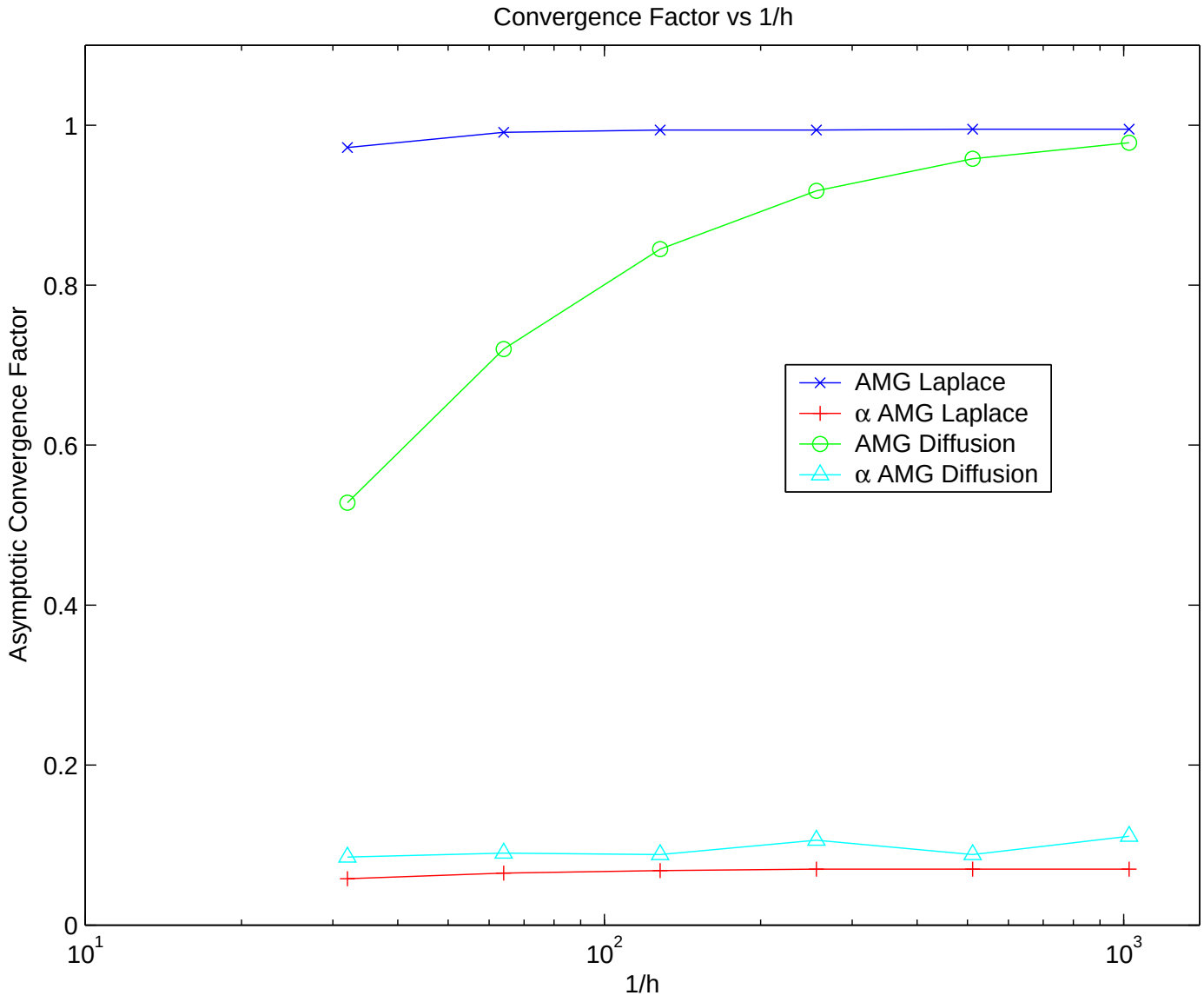
h	Problem 1	Problem 2	Problem 3	Problem 4
$1/32$	12.9	14.5	1297	59.4
$1/64$	13.4	15.6	4075	112.1
$1/128$	13.6	14.9	6122	218.7
$1/256$	13.8	16.4	6122	430.6
$1/512$	13.9	15.2	7350	858.6
$1/1024$	13.9	16.7	7350	1656

Adapted Interpolation Results

Work Units using adapted AMG

h	Problem 1	Problem 2	Problem 3	Problem 4
$1/32$	12.9	14.9	12.9	14.9
$1/64$	13.4	15.6	13.5	15.3
$1/128$	13.6	15.2	13.7	15.3
$1/256$	13.8	16.4	13.8	16.4
$1/512$	13.9	15.2	13.9	15.2
$1/1024$	13.9	16.7	13.9	16.8

Convergence Factors



Full Adaptivity

- Method still relies on having a good representative of an algebraically smooth vector
- If known a priori, then use it
- If not, then develop the representative using the method and adapt interpolation (and possibly coarse grids) as information is available

Relation to PCG

- A simple version of the fully adaptive procedure mimics PCG
- Given x_0, relax on $Ax = 0$, so $x_1 = Sx_0$
- Define $P_1 = [x_1]$, perform (two-level) multigrid correction
- Error after correction is

$$e_1 = (I - P_1(P_1^T A P_1)^{-1} P_1^T A) S e_0$$

- Notice $Q_1 = P_1(P_1^T A P_1)^{-1} P_1^T A$ is an A-orthogonal projection onto the space $\{Sx_0\}$.

Relation to PCG

- At step k:
  - Relax on $Ax = 0$ with the current multigrid and initial guess x_{k-1}, so

$$x_k = (I - Q_{k-1})Sx_{k-1} = (I - Q_{k-1})S^k x_0$$

  - Define $P_k = [P_{k-1}, x_k]$, $Q_k = P_k(P_k A P_k)^{-1} P_k^T A$
  - New multigrid cycle is given by error propagation matrix

$$e_k = (I - Q_k)Se_0$$

- Q_k is an A-orthogonal projection onto

$$K_k = \{Sx_0, S^2 x_0, \ldots, S^k x_0\}$$

Improvement on PCG

- Instead of mimicking PCG, we define P so that the coarse-grid system ($P^T AP$) retains properties of the fine-grid system
- The vector x_k now becomes an indicator of error that is both algebraically smooth and not sufficiently addressed by coarse-grid correction
- Updating interpolation is done so that it remains accurate for x_1 through x_{k-1}, but also addresses x_k
- Results to date indicate that k need be no larger than the null-space of the differential operator

Summary

- For regular grids, with smooth PDE coefficients, geometric MG works well
- For irregular grids, discontinuous coefficients, algebraic MG works well
- For coupled systems, exotic bases, adaptive algebraic MG offers hope
- All are $O(N)$ algorithms, constants are non-trivial, but not prohibitive

Current and Future Work

- Developing a theory for self-correcting AMG
- Developing a fully-algebraic version
- Investigating better coarsening procedures (Compatible Relaxation)
- Natural extension to systems
- Alternate smoothers