APPM 6640
Multigrid Methods

Steve McCormick & John Ruge
303-492-0662
stevem@colorado.edu & john.ruge@colorado.edu
http://amath.colorado.edu/faculty/stevem

Office Hours
MW 2-4:45 pm (except for class!) in ECOT 322

Sources

MGNet Newsletter & software repository
http://www.mgnet.org/

MathSciNet Many papers electronically available
http://www.ams.org/mathscinet

Copper Mountain Conference March 22-27, 2009
http://amath.colorado.edu/faculty/copper
We hope to support your attendance there.

Homework exercises

Due 1 week after the relevant chapter is covered.

1: 3, 4
6: 1, 2, 6
2: 1-3, 8, 10, 13, 16
7: 1, 2, 7, 10, 12, 15, 21
3: 2-6
8: 2, 6, 8
4: 6, 7, 11
9: 1, 3
5: 1-3, 12, 13
10: 1, 3, 5, 8, 9
Computing assignments

Due 2 weeks after the relevant chapter is covered.
Start your assignments early:
multigrid algorithms can be tricky to code!
Use whatever computing language/platform you like.

2: 20  4: 13, 15  6: 6 by computer!!!

7: Convert your 2-D code developed for Chapter 4
(standard coarsening & point relaxation) to solve
\((7.14)\).
Verify the code using \(\varepsilon = 1\). Test its
performance using \(\varepsilon = 1/2, 1/5, 1/10, 2, 5, 10\).
Interpret your results.

Project & presentation

Due at the end of the semester.

- Project definition is intentionally vague.
- Think about what has captured your attention.
- See us, but only after you've thought about it.
- Computing &/or theory &/or application &/or exploration.
- I'd like to see a presentation (10 minutes) plus a short
\((<5 \text{ pages})\) writeup about what you did.
- Presentation: Prepare! Speak to your peers, not me.
- Check course web site for suggestions.

Successful Scientific Inquiry

- Attitude
  - You can do it! Be positive.
  - But is it really right? Be critical.
  - Don't hope or guess. THINK!
  - Control your emotions! Expect ups & downs.
- Method
  - Start simply. Reduce issue to the simplest possible case.
  - Take tiny steps, but keep the big picture in mind.
  - Study concrete examples.
  - Look for analogies. Can A be done in any way like how B was done?
- Creativity
  - What do you really want? What end are you really aiming for?
  - What do you really need? What you're trying may be sufficient to do
what you want, but would an easier-to-prove weaker result do instead?
- Intelligence
  - It doesn't hurt to try to be "smart" too.

A Multigrid Tutorial

2nd Edition, 2nd Printing

By
William L. Briggs  
CU-Denver
Van Emden Henson  
LLNL
Steve McCormick  
CU-Boulder

THANKS!

THANKS!
### Outline

1. **Model Problems**  
2. **Basic Iterative Methods**  
   Convergence tests  
   Analysis  
3. **Elements of Multigrid**  
   Relaxation  
   Coarsening  
4. **Implementation**  
   Complexity  
   Diagnostics  
5. **Some Theory**  
   Spectral vs. algebraic  
6. **Nonlinear Problems**  
   Full approximation scheme  
7. **Selected Applications**  
   Neumann boundaries  
   Anisotropic problems  
   Variable meshes  
   Variable coefficients  
8. **Algebraic Multigrid (AMG)**  
   Matrix coarsening  
9. **Multilevel Adaptive Methods**  
   FAC  
10. **Finite Elements**  
    Variational methodology

### Suggested reading

**CHECK THE MG LIBRARY & MGNET REPOSITORY**


### Multilevel methods have been developed for...

- PDEs, CFD, porous media, elasticity, electromagnetics.
- Purely algebraic problems, with no physical grid; for example, network & geodetic survey problems.
- Image reconstruction & tomography.
- Optimization (e.g., the traveling salesman & long transportation problems).
- Statistical mechanics, Ising spin models.
- Quantum chromo dynamics.
- Quadrature & generalized FFTs.
- Integral equations.

### Everyone uses multilevel methods

- Multigrid, multilevel, multiscale, multiphysics, ...
  Use local "governing rules" on the finest level to resolve the state of the system at these detailed scales, but--recognizing that these "rules" have broader implications that are hard to determine there--use coarser levels to resolve larger scales. Continual feedback is essential because improving one scale impacts other scales.

- Common uses
  Sight, art, politics, thinking (scientific research), cooking, team sports, ...
1. Model problems

- 1-D boundary value problem:
  \[-u''(x) + \sigma u(x) = f(x) \quad 0 < x < 1, \quad \sigma \geq 0\]
  \[u(0) = u(1) = 0\]

- Grid:
  \[h = \frac{1}{N+1}, \quad x_i = ih, \quad i = 0,1,\ldots,N+1\]

- Let \(v_i = u(x_i)\) & \(f_i \approx f(x_i)\) for \(i = 0,1,\ldots,N+1\).

This discretizes the variables, but what about the equations?

Approximate equation via finite differences

- Approximate the BVP
  \[-u''(x) + \sigma u(x) = f(x) \quad 0 < x < 1, \quad \sigma \geq 0\]
  \[u(0) = u(1) = 0\]

  by a finite difference scheme:
  \[-v_{i-1} + 2v_i - v_{i+1} \quad \frac{h^2}{h^2} + \sigma v_i = f_i \quad i = 1,2,\ldots,N\]

  \[v_0 = v_{N+1} = 0\]

Approximate \(u''(x)\) via Taylor series

- Approximate 2\(^{nd}\) derivative using Taylor series:
  \[u(x_{i+1}) = u(x_i) + h u'(x_i) + \frac{h^2}{2!} u''(x_i) + O(h^4)\]
  \[u(x_{i-1}) = u(x_i) - h u'(x_i) + \frac{h^2}{2!} u''(x_i) - \frac{h^4}{3!} u'''(x_i) + O(h^4)\]

- Summing & solving:
  \[u''(x_i) = \frac{u(x_{i+1}) - 2u(x_i) + u(x_{i-1})}{h^2} + O(h^2)\]

Discrete model problem

Letting \(v = (v_1, v_2, \ldots, v_N)^T\) & \(f = (f_1, f_2, \ldots, f_N)^T\)

we obtain the matrix equation \(Av = f\), where \(A\) is \(N \times N\), symmetric, positive definite, &

\[
A = \frac{1}{h^2} \begin{pmatrix}
2+\sigma h^2 & -1 & & \\
-1 & 2+\sigma h^2 & -1 & & \\
& \ddots & \ddots & \ddots & -1 \\
& & -1 & 2+\sigma h^2 & -1 \\
& & & -1 & 2+\sigma h^2
\end{pmatrix}
\]

\[v = \begin{pmatrix}
v_1 \\
v_2 \\
\vdots \\
v_{N-1} \\
v_N
\end{pmatrix}, \quad f = \begin{pmatrix}
f_1 \\
f_2 \\
\vdots \\
 f_{N-1} \\
f_N
\end{pmatrix}\]
Stencil notation

\[ A = [-1 \ 2 \ -1] \]

dropping \( h^{-2} \) & \( \sigma \) for convenience

Basic solution methods

- Direct
  - Gaussian elimination
  - Factorization
  - Fast Poisson solvers (FFT-based, reduction-based, …)

- Iterative
  - Richardson, Jacobi, Gauss-Seidel, …
  - Steepest Descent, Conjugate Gradients, …
  - Incomplete Factorization, …

- Notes:
  - This simple 1-D problem can be solved efficiently in many ways. Pretend it can’t & that it’s very hard, because it shares many characteristics with some very hard problems. If we keep things as simple as possible by studying this model, we’ve got a chance to really understand what’s going on.
  - But, to keep our feet on the ground, let’s go to 2-D anyway…

2-D model problem

- Consider the problem
  
  \[ -u_{xx} - u_{yy} + \sigma u = f(x,y), \quad 0 < x < 1, \quad 0 < y < 1 \]

  \( u = 0 \) when \( x = 0, x = 1, y = 0, \) or \( y = 1 \quad \sigma \geq 0 \)

- Consider the grid
  
  \( h_x = \frac{1}{L+1}, \quad h_y = \frac{1}{M+1}, \]

  \( (x_i, y_j) = (i h_x, j h_y) \)

  \( 0 \leq i \leq L+1 \]

  \( 0 \leq j \leq M+1 \)

Discretizing the 2-D problem

- Let \( v_{ij} = u(x_i, y_j) \) & \( f_{ij} = f(x_i, y_j) \). Again, using 2nd-order finite differences to approximate \( u_{xx} \) & \( u_{yy} \) we arrive at the approximate equation for the unknown \( u(x_i, y_j) \), for \( i = 1, 2, \ldots, L \) & \( j = 1, 2, \ldots, M \):

  \[
  \begin{align*}
  -v_{i-1,j} + 2v_{ij} - v_{i+1,j} + \frac{h_x^2}{\sigma} (-v_{i-1,j} + 2v_{ij} - v_{i+1,j}) + \frac{h_y^2}{\sigma} (-v_{i,j-1} + 2v_{ij} - v_{i,j+1}) + \sigma v_{ij} &= f_{ij} \\
  v_{i,j} &= 0: \quad i = 0, \quad i = L + 1, \quad j = 0, \quad j = M + 1
  \end{align*}
  \]

- Order the unknowns ( & also the vector \( f \) ) lexicographically by \( y \)-lines:

  \[ v = (v_{1,1}, v_{1,2}, \ldots, v_{1,M}, v_{2,1}, v_{2,2}, \ldots, v_{2,M}, \ldots, v_{L,1}, v_{L,2}, \ldots, v_{L,M})^T \]
Resulting linear system

- We obtain a block-tridiagonal system $Av = f$:

$$
\begin{pmatrix}
A_1 & -I_x & & & \\
-I_x & A_2 & -I_x & & \\
& \ddots & \ddots & \ddots & \\
& & -I_x & A_{k-1} & -I_x \\
& & & -I_x & A_k
\end{pmatrix}
\begin{pmatrix}
v_1 \\
v_2 \\
\vdots \\
v_{k-1} \\
v_k
\end{pmatrix} =
\begin{pmatrix}
f_1 \\
f_2 \\
\vdots \\
f_{k-1} \\
f_k
\end{pmatrix}
$$

where $I_x$ is the $h^{-2}$ times the identity matrix &

$$
A = \begin{pmatrix}
\frac{2}{h_x^2} + \frac{\sigma}{h_x} & \frac{1}{h_x} & & & \\
-\frac{1}{h_x} & \frac{2}{h_x^2} + \frac{\sigma}{h_x} & \frac{2}{h_x} & & \\
& \ddots & \ddots & \ddots & \\
& & -\frac{1}{h_x} & \frac{2}{h_x^2} + \frac{\sigma}{h_x} & \frac{1}{h_x} \\
& & & -\frac{1}{h_x} & \frac{2}{h_x^2} + \frac{\sigma}{h_x}
\end{pmatrix}
$$

Outline

Chapters 1-5:
- Model Problems
- Basic Iterative Methods
- Convergence tests
- Analysis
- Elements of Multigrid
- Relaxation
- Coarsening
- Implementation
- Complexity
- Diagnostics
- Some Theory
- Spectral vs. algebraic

Chapters 6-10:
- Nonlinear Problems
  - Full approximation scheme
- Selected Applications
  - Neumann boundaries
  - Anisotropic problems
- Algebraic Multigrid (AMG)
  - Matrix coarsening
- Multilevel Adaptive Methods
  - FAC
- Finite Elements
  - Variational methodology

Stencils preferred for grid issues

Stencils are much better for showing the grid picture:

$$
\begin{bmatrix}
0 & -1/h_y & 0 \\
-1/h_x & \frac{2}{h_x^2} + \frac{\sigma}{h_x} & -1 \\
0 & -1/h_y & 0
\end{bmatrix}
$$

Stencils show local relationships--grid point interactions.

2. Basic iterative methods

- Consider the $N \times N$ matrix equation $Au = f$ & let $v$ be an approximation to $u$.

- Two important measures:
  - The Error: $e = u - v$ with norms
    $$
    \|e\|_\infty = \max_i e_i \quad \& \quad \|e\|_2 = \sqrt{\sum_i e_i^2}
    $$
    What does $\|e\|_\infty$ measure???
  - The Residual: $r = f - Av$ with
    $$
    \|r\|_\infty \quad \& \quad \|r\|_2
    $$
    Why have both $r$ & $e$???
**Residual correction**

- Since $e = u - v$, we can write $Au = f$ as $A(v + e) = f$
  
  which means that $Ae = f - Av \equiv r$.

- Residual Equation: $Ae = r$
  
  What does this do for us???

- Residual Correction: $u = v + e$

**Relaxation**

- Consider the 1-D model problem
  $$-u_{i-1} + 2u_i - u_{i+1} = h^2 f_i \quad 1 \leq i \leq N$$
  \hspace{1cm} u_0 = u_{N+1} = 0

- Jacobi (simultaneous displacement): Solve the $i^{th}$ equation for $v_i$ holding all other variables fixed:
  $$v_i^{(new)} = \frac{1}{2} (v_i^{(old)} + v_{i+1}^{(old)} + h^2 f_i) \quad 1 \leq i \leq N$$

**Jacobi in matrix form**

- Let $A = D - L - U$, where $D$ is diagonal & $-L$ & $-U$ are the strictly lower & upper triangular parts of $A$.

- Then $Au = f$ becomes
  $$(D-L-U)u = f$$
  $$Du = (L+U)u + f$$
  $$u = D^{-1} (L+U)u + D^{-1} f$$

- Let $R_J = D^{-1} (L+U)$, \hspace{1cm} $R_J = D^{-1} (D-A) = I - D^{-1} A$
  
  - $R_J$ is called the error propagation or iteration matrix.

- Then the iteration is
  $$v^{(new)} = R_J v^{(old)} + D^{-1} f$$

**Error propagation matrix & the error**

- From the derivation,
  $$u = D^{-1} (L+U)u + D^{-1} f$$
  \hspace{1cm} $u = R_J u + D^{-1} f$

- the iteration is
  $$v^{(new)} = R_J v^{(old)} + D^{-1} f$$

- subtracting,
  $$u - v^{(new)} = R_J u + D^{-1} f - (R_J v^{(old)} + D^{-1} f)$$
  \hspace{1cm} or
  $$u - v^{(new)} = R_J u - R_J v^{(old)}$$

- hence, Error propagation!
  \hspace{1cm} $e^{(new)} = R_J e^{(old)}$ \hspace{1cm} $R_J = I - D^{-1} A$
A picture

\[ R_I = D^{-1} (L + U) = \begin{bmatrix} \frac{1}{2} & 0 \\ 0 & \frac{1}{2} \end{bmatrix} \]

so Jacobi is an error averaging process:

\[ e_i^{(new)} \leftarrow (e_i^{(old)} + e_{i+1}^{(old)})/2 \]

Another matrix look at Jacobi
\[
v^{(new)} \leftarrow D^{-1} (L + U) v^{(old)} + D^{-1} f \quad (L + U = D - A)
\]
\[ = (I - D^{-1}A) v^{(old)} + D^{-1} f
\]
\[ v^{(new)} = v^{(old)} - D^{-1} (Av^{(old)} - f) = v^{(old)} + D^{-1} r
\]

- Exact: \[ u = u - D^{-1} (Au - f) \]
- Subtracting: \[ e^{(new)} = e^{(old)} - D^{-1} A e^{(old)} \]
- Exact: \[ u = u - A^{-1} (Au - f) = A^{-1} f \]
- General form: \[ u = u - B (Au - f) \] with \( B \sim A^{-1} \)
- Damped Jacobi: \[ u = u - \omega D^{-1} (Au - f) \] with \( 0 < \omega \approx 1 \)
- Gauss-Seidel: \[ u = u - (D - L)^{-1} (Au - f) \]

Note that \( R_0 = I - \omega D^{-1} A \) is a polynomial in \( A \) when \( D = I \). We exploit this simplicity (symmetry, etc.) in what follows.!!!This special property doesn't usually hold in practice!!!

But...

Weighted Jacobi

safer \((0 < \omega < 1)\) changes

- Consider the iteration

\[ v_i^{(new)} \leftarrow (1 - \omega) v_i^{(old)} + \frac{\omega}{2} (v_{i-1}^{(old)} + v_{i+1}^{(old)}) + h^2 f_i \]

- Letting \( A = D - L - U \), the matrix form is

\[ v^{(new)} = \left[ (1 - \omega) I + \omega D^{-1} (L + U) \right] v^{(old)} + \omega h^2 D^{-1} f
\]

- Note that

\[ R_\omega = \left[ (1 - \omega) I + \omega R_f \right] \]

- It is easy to see that if \( e^{(approx)} = u - v^{(approx)} \), then

\[ e^{(new)} = R_\omega e^{(old)} \]
Gauss-Seidel (1-D)

- Solve equation \( i \) for \( u \) & update immediately.
- Equivalently: set each component of \( r \) to zero in turn.
- Component form: for \( i = 1, 2, \ldots, N \), set
  \[
  v_i \leftarrow \frac{1}{2} (v_{i-1} + v_{i+1} + h^2 f_i)
  \]
- Matrix form:
  \[
  A = (D - L - U)
  \]
  \[
  (D - L) u = f
  \]
  \[
  u = (D - L)^{-1} f
  \]
- Let \( R_G = (D - L)^{-1} U \)
- Then iterate:
  \[
  v^{(new)} \leftarrow R_G v^{(old)} + (D - L)^{-1} f
  \]
- Error propagation:
  \[
  e^{(new)} \leftarrow R_G e^{(old)}
  \]

Red-black Gauss-Seidel

- Update the EVEN points:
  \[
  v_{2i} \leftarrow \frac{1}{2} (v_{2i-1} + v_{2i+1} + h^2 f_{2i})
  \]
- Update the ODD points:
  \[
  v_{2i+1} \leftarrow \frac{1}{2} (v_{2i} + v_{2i+2} + h^2 f_{2i+1})
  \]

Test?

\[ Au = f \]

Need to know how we're doing!!

- What \( f \)?
  \[ Au = 0 \]

- What \( v \)?
  \[ v = \text{rand} \]

Numerical experiments

- Solve \( Au = 0 \), \[-u_{j-1} + 2u_j - u_{j+1} = 0\]
- Use Fourier modes as initial iterates, with \( N = 63 \):
  \[
  \bar{v}_k = (v_i)_k = \sin \left( \frac{ik\pi}{N+1} \right) \quad 1 \leq i \leq N, \quad 1 \leq k \leq N
  \]
  \[
  \sin(k\pi x), \quad x = i/N+1
  \]

\[ k = 1 \]

\[ k = 3 \]

\[ k = 6 \]
Convergence factors differ for different error components

Error, $\|e\|_\infty$, in weighted ($\omega=2/3$) Jacobi on $Au=0$ for 100 iterations using initial guesses $v_1, v_3, \& v_6$

Analysis of stationary linear iteration

- Let $v^{(\text{new})} = Rv^{(\text{old})} + g$. The exact solution is unchanged by the iteration: $u = Ru + g$.
- Subtracting: $e^{(\text{new})} = Re^{(\text{old})}$.
- Let $e^{(0)}$ be the initial error & $e^{(i)}$ be the error after the $i^{th}$ iteration. After $n$ iterations, we have $e^{(n)} = R^n e^{(0)}$.

Quick review of eigenvectors & eigenvalues

- The number $\lambda$ is an eigenvalue of a matrix $B$ & $w \neq 0$ its associated eigenvector if $Bw = \lambda w$.
- The eigenvalues & eigenvectors are characteristics of a given matrix.
- Eigenvectors are linearly independent, & if there is a complete set of $N$ distinct eigenvectors for an $N \times N$ matrix, then they form a basis: for any $v$, there exist unique scalars $v_k$ such that $v = \sum_{k=1}^{N} v_k w_k$.
- Propagation: $B^n v = \sum_{k=1}^{N} \lambda^n v_k w_k$.
“Fundamental Theorem of Iteration”

$R$ is convergent ($R^n \to 0$ as $n \to \infty$) iff

$$ \rho(R) = \max |\lambda| < 1. $$

Thus, $e^{(n)} = R^n e^{(0)} \to 0$ for any initial vector $v^{(0)}$ iff $\rho(R) < 1$.

$\rho(R) \times 1$ assures convergence of $R$ iteration.

$\rho(R)$ is the spectral convergence factor.

But $\rho$ is doesn’t tell you much by itself—it’s generally valid only asymptotically. It’s useful for the symmetric case in particular because it’s equal to $\|R\|_2$, so we’ll use it here.

Rayleigh quotient vs. spectral radius

assume $A$ is symmetric ($w_k$ orthonormal) & nonnegative definite ($\lambda \geq 0$)

- $RQ(v) \leq \rho(A)$:

$$ RQ(v) = \frac{< Av, v >}{< v, v >} = \frac{< A \sum v_k w_k, \sum v_k w_k >}{< \sum v_k w_k, \sum v_k w_k >} = \frac{< \sum \lambda_k v_k w_k, \sum v_k w_k >}{< \sum v_k w_k, \sum v_k w_k >} = \frac{\sum \lambda_k v_k}{\sum v_k} \leq \lambda_N = \rho(A) $$

- $\sup_{v \neq 0} RQ(v) = \rho(A)$:

$$ RQ(w_N) = \frac{< Aw_N, w_N >}{< w_N, w_N >} = \frac{\lambda_N w_N, w_N}{< w_N, w_N >} = \lambda_N = \rho(A) $$

Euclidean norm vs. spectral radius

use $RQ$

- $\|R\|_2 = \rho^{1/2}(R^TR)$:

$$ \|R\|_2^2 = \sup_{e \neq 0} \|Re\|_2^2 / \|e\|_2^2 $$

$$ = \sup_{e \neq 0} < Re, Re > / < e, e > $$

$$ = \sup_{e \neq 0} < R^TRe, e > / < e, e > = \rho(R^TR) $$

note: $\|Re\|_2 \leq \|R\|_2 \cdot \|e\|_2$

- $\|A\|_2 = \rho^{1/2}(A^2) = \rho(A)$ for symmetric $A$!!!
Example: \( R = \begin{pmatrix} 0 & K \\ 0 & 0 \end{pmatrix} \), \( K \) large

\[ \rho(R) = 0 \text{ but } \|R\|_2 = K! \]

\[ e^{(0)} = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \Rightarrow \|e^{(0)}\|_2 = \|Re^{(0)}\|_2 = K \]

Thus, 1 iteration with \( e^{(0)} \) shows dramatic \( L^2 \) divergence!

But \( R^2 = 0 \), so \( e^{(2)} = Re^{(1)} = R^2e^{(0)} = 0! \)
Thus, 2 iterations with \( e^{(0)} \) show complete convergence!

On one hand, this is special (\( \lambda = 0 \), large \( K \), 2x2), so this behavior would be more subtle & persistent in general.

On the other, this behavior would vanish for symmetric \( R \).

Convergence analysis: Weighted Jacobi

\[ R_\omega = (1 - \omega)I + \omega D^{-1}(L + U) = I - \omega D^{-1}A \]

\[ R_\omega = I - \frac{\omega}{2} \begin{pmatrix} 2 & -1 & \cdot & \cdot & \cdot \\ -1 & 2 & -1 & \cdot & \cdot \\ \vdots & \vdots & \ddots & \ddots & \cdot \\ -1 & 2 & \cdot & \cdot & -1 \\ -1 & 2 \end{pmatrix} \]

For our 1-D model, the eigenvectors of weighted Jacobi \( R_\omega \) & the eigenvectors of \( A \) are the same! Why???

\[ \lambda(R_\omega) = 1 - \frac{\omega}{2} \lambda(A) \]

This is very special!!! Remember that \( A \) is without \( h^2 \) here!

Convergence factor & rate

\[ \cdot \text{ How many iterations are enough to guarantee reduction of the initial error by } 10^{-d}? \]
\[ \|e^{(n)}\| \leq \|e^{(0)}\| \leq \|R\|^n \sim 10^{-d} \]
\[ \cdot \text{ So, we have } \]
\[ n \sim -\log_{10} \rho(R) \]
\[ \cdot \text{ Convergence factor } = \|R\| \text{ or } \rho(R). \]
\[ \cdot \text{ Convergence rate } = -\log_{10} \|R\| \frac{\text{digits}}{\text{iteration}} \text{ or } -\log_{10}(\rho(R)). \]

Eigenpairs of (scaled) \( A \)

The eigenvectors of \( A \) are Fourier modes!

\[ \lambda_j(A) = 4\sin^2 \left( \frac{k\pi}{2(N+1)} \right), \quad w_{k,j} = \sin \left( \frac{ik\pi}{N+1} \right) \]

\[ \lambda_N \equiv \frac{-2}{N+1} \quad \lambda_1 \equiv \sqrt{\pi^2 h^2} \]

\( N = 63 \)

\[ \begin{array}{ccc}
\cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot \\
\end{array} \]

\( k = 1 \)  \( k = 3 \)  \( k = 8 \)  \( k = 16 \)  \( k = 32 \)
Eigenvectors of $R_\omega = \text{eigenvectors of } A$

$$\lambda_k(R_\omega) = 1 - 2\omega \sin^2\left(\frac{k\pi}{2(N+1)}\right)$$

- Expand the initial error in terms of the eigenvectors:
  $$e^{(0)}(\omega) = \sum_{k=1}^{N} c_k w_k$$
- After $n$ iterations:
  $$R^n e^{(0)}(\omega) = \sum_{k=1}^{N} c_k \lambda_k^n w_k$$
- The $k^\text{th}$ error mode is reduced by $\lambda_k(R_\omega)$ each iteration.

Relaxation suppresses eigenmodes unevenly

- Look carefully at $\lambda_k(R_\omega) = 1 - 2\omega \sin^2\left(\frac{k\pi}{2(N+1)}\right)$.

Note that if $0 < \omega \leq 1$, then $|\lambda_k(R_\omega)| < 1$ for $k = 1, 2, \ldots, N$.

For $0 < \omega \leq 1$,

$$\lambda = 1 - 2\omega \sin^2\left(\frac{\pi}{2}\right) = 1 - 2\omega \sin^2\left(\frac{\pi h}{2}\right) = 1 - O(h^2) = 1$$

Low frequencies are “undamped”

Notice that no value of $\omega$ will efficiently damp out long waves or low frequencies.

What value of $\omega$ gives the best damping of short waves or high frequencies $N/2 \leq k \leq N$?

Choose $\omega$ such that

$$\lambda_{N/2}(R_\omega) = -\lambda_N(R_\omega)$$

$$\Rightarrow \omega \approx \frac{2}{3}$$

Smoothing factor

- The smoothing factor is the largest magnitude of the iteration matrix eigenvalues corresponding to the oscillatory Fourier modes:
  $$\text{smoothing factor} = \max |\lambda_k(R)|$$
  for $N/2 \leq k \leq N$.
- Why only the upper spectrum? “MG” spectral radius?
- For $R_\omega$ with $\omega=2/3$, the smoothing factor is 1/3:
  $$|\lambda_{N/2}| = |\lambda_N| = 1/3 \text{ & } |\lambda_k| < 1/3 \text{ for } N/2 < k < N.$$
- But $|\lambda_k| \approx 1 - \omega k^2 \pi^2 h^2$ for long waves ($k \ll N/2$).
**Convergence of Jacobi on** $Au = 0$

- Jacobi on $Au = 0$ with $N = 63$. Number of iterations needed to reduce initial error $||e||_\infty$ by 0.01.
- Initial guess:
  $$v_{k,i} = \sin\left(\frac{ik\pi}{N+1}\right)$$

**Weighted Jacobi = smoother (error)**

- Initial error:
  $$v_i = \sin\left(\frac{2i\pi}{N+1}\right) + \frac{1}{2} \sin\left(\frac{16i\pi}{N+1}\right) + \frac{1}{2} \sin\left(\frac{32i\pi}{N+1}\right)$$
- Error after 35 iteration sweeps:

**Similar analysis for other smoothers**

- Gauss-Seidel relaxation applied to the 3-point difference matrix $A$ (1-D model problem):
  $$R_G = (D-L)^{-1}U$$
- A little algebra & trigonometry shows that
  $$\lambda_k(R_G) = \cos\left(\frac{k\pi}{N+1}\right)$$
  $$\lambda_{3/2}(R_G) = \sin(3\pi x_i)$$
  $$\lambda_k^{1/2} \sin(3\pi x_i)$$
- What's $w_k$ look like for large $k$?

**Gauss-Seidel eigenvectors**

These a VERY different from Jacobi’s eigenvectors.
It’s not clear how smoothness depends on $k$.
You cannot expect G-S to quickly reduce Fourier modes.
You can only expect G-S to produce smooth results!
Gauss-Seidel convergence

\[ Au = 0 \]

Eigenvectors of \( R_G \) are not the same as those of \( A \)!!

Gauss-Seidel mixes the modes of \( A \).

Gauss-Seidel on \( Au = 0 \), with \( N = 63 \). Number of iterations needed to reduce initial error \( \| e \|_\infty \) by 0.01.

Initial guess (modes of \( A \)):

\[ v_{ki} = \sin \left( \frac{ik\pi}{N + 1} \right) \]

OK, so G-S does reduce oscillatory Fourier modes. But you will see that this depends on ordering. In fact, red-black G-S does not reduce high Fourier modes.

3. Elements of multigrid

1st observation toward multigrid

- Many relaxation schemes have the smoothing property: oscillatory error modes are quickly eliminated, while smooth modes are often very slow to disappear.
- We’ll turn this adversity around: the idea is to use coarse grids to take advantage of smoothing.

Reason #1 for coarse grids:

Nested iteration

- Coarse grids can be used to compute an improved initial guess for the fine-grid relaxation. This is advantageous because:
  - Relaxation on the coarse-grid is much cheaper: half as many points in 1-D, one-fourth in 2-D, one-eighth in 3-D, ...
  - Relaxation on the coarse grid has a marginally faster convergence factor (\( |\lambda_1(R)| \approx 1 - \omega^2h^2 \)):

\[ 1 - O(4h^2) \quad \text{instead of} \quad 1 - O(h^2) \]

Outline

Chapters 1-5:
- Model Problems
- Basic Iterative Methods
  - Convergence tests
  - Analysis
- Elements of Multigrid
  - Relaxation
  - Coarsening
- Implementation
  - Complexity
  - Diagnostics
- Some Theory
  - Spectral vs. algebraic

Chapters 6-10:
- Nonlinear Problems
  - Full approximation scheme
- Selected Applications
  - Neumann boundaries
  - Anisotropic problems
  - Variable meshes
  - Variable coefficients
- Algebraic Multigrid (AMG)
  - Matrix coarsening
- Multilevel Adaptive Methods
  - FAC
- Finite Elements
  - Variational methodology

Homework Due Computing Tool

Chapters 1-5:

Chapters 6-10:
Nested iteration

- Relax on $Au = f$ on $\Omega^4h$ to obtain initial guess $v^2h$.
- Relax on $Au = f$ on $\Omega^2h$ to obtain initial guess $v^h$.
- Relax on $Au = f$ on $\Omega^h$ to obtain ... final solution???

- What is $A^{2h}v^{2h} = f^h$?
  Analogous to $A^h v^h = f^h$ for now.
- How do we migrate between grids?
  Hang on...
- What if the error still has large smooth components when we get to the fine grid $\Omega^h$?
  Hang on...

What if the error still has large smooth components when we get to the fine grid $\Omega^h$?

$A^{2h}u^{2h} = f^h$ can be represented by linear interpolation from a coarser grid:

For $k=1,2,...(N-1)/2$, the $k^{th}$ mode is preserved on the coarse grid.

$$w_{2i}^h = \sin\left(\frac{2i\pi}{N+1}\right) = \sin\left(\frac{ik\pi}{(N+1)/2}\right) = w_{2i}^{2h}$$

Also, note that $w_{(N+1)/2}^h \to 0$ on the coarse grid.

What happens to the modes between $(N+1)/2$ & $N$?

For $k > (N+1)/2$, $w_k^h$ is disguised on the coarse grid: aliasing!

For $k > (N+1)/2$, the $k^{th}$ mode on the fine grid is aliased & appears as the $(N+1 - k)^{th}$ mode on the coarse grid:

$$w_k^h = \sin\left(\frac{2i\pi}{N+1}\right) = -\sin\left(\frac{2i\pi(N+1-k)}{N+1}\right) = -\sin\left(\frac{i\pi(N+1-k)}{(N+1)/2}\right) = -\left(w_{N+1-k}^{2h}\right)$$

Reason #2 for coarse grids:
Smooth error becomes more oscillatory

- A smooth function:
- On the coarse grid, the smooth error appears to be relatively higher in frequency: in this example it is the 4-mode out of a possible 15 on the fine grid, $\sim 1/4$ the way up the spectrum. On the coarse grid, it is the 4-mode out of a possible 7, $\sim 1/2$ the way up the spectrum.

Relaxation on $2h$ is cheaper & faster on this mode!!!
1-D interpolation (prolongation)

- Values at points on the coarse grid map unchanged to the fine grid.
- Values at fine-grid points NOT on the coarse grid are the averages of their coarse-grid neighbors.

\[ I_{2h}^h : \Omega^{2h} \rightarrow \Omega^h \]

- Mapping from the coarse grid to the fine grid:

\[ v_{2h}^h = v^h \]

where

\[ v_{2i+1}^h = \frac{1}{2} (v_{2i}^h + v_{i+1}^h) \quad \text{for } 0 \leq i \leq \frac{N-1}{2}. \]

1-D prolongation operator \( P \)

- \( P = I_{2h}^h \) is a linear operator: \( \mathcal{R}^{(N-1)/2} \rightarrow \mathcal{R}^N. \)
- \( N = 7: \)

\[
\begin{bmatrix}
\frac{1}{2} & 1 & \frac{1}{2} \\
1 & 0 & \frac{1}{2} \\
\frac{1}{2} & 1 & \frac{1}{2} \\
0 & 1 & 0
\end{bmatrix}
\begin{bmatrix}
v_1^h \\
v_2^h \\
v_3^h \\
v_4^h \\
v_5^h \\
v_6^h \\
v_7^h
\end{bmatrix}
= 
\begin{bmatrix}
v_1^h \\
v_2^h \\
v_3^h \\
v_4^h \\
v_5^h \\
v_6^h \\
v_7^h
\end{bmatrix}
\]

- \( I_{2h}^h \) has full rank, so \( \eta(P) = \{0\}. \)

\[ \text{rank} = \text{max # linearly independent cols or rows} \]

“Give To” stencil for \( P \)

\[
\begin{bmatrix}
\frac{1}{2} & 1 & \frac{1}{2}
\end{bmatrix}
\]

\[ 0 \times 1 0 2 \quad 1 0 2 \times 0 \]
How well could $v^{2h}$ approximate $u$?

- Imagine that a coarse-grid approximation $v^{2h}$ has been found. How well could it approximate the exact solution $u$?

- If $u$ is smooth, a coarse-grid interpolant $v^{2h}$ might do very well.

How well could $v^{2h}$ approximate $u$?

- Imagine that a coarse-grid approximation $v^{2h}$ has been found. How well could it approximate the exact solution $u$?

- If $u$ is oscillatory, a coarse-grid interpolant $v^{2h}$ cannot work well.

Moral

- If what we want to compute is smooth, a coarse-grid interpolant could do very well.
- If what we want to compute is oscillatory, a coarse-grid interpolant cannot do very well.
- What if $u$ is not smooth? Can we make it so?
- Can we make something smooth?

- How about $e$? Can we smooth it? How would we get $e$ & use it to get $u$? $Ae = r$ & $u ← v + e$!
- Thus, use nested iteration on residual equation to approximate the error after smoothing!!
- Just because the coarse grid can approximate $e$ well doesn’t mean we know how to do it! But we will soon!

2nd observation toward multigrid

- The residual equation: Let $v$ be an approximation to the solution of $Au = f$, where the residual $r = f - Av$. Then the error $e = u - v$ satisfies $Ae = r$.

- After relaxing on $Au = f$ on the fine grid, $e$ will be smooth, so the coarse grid can approximate $e$ well. This will be cheaper & $e$ should be more oscillatory there, so relaxation will be more effective.

- Therefore, we go to a coarse grid & relax on the residual equation $Ae = r$.

What's a good initial guess on grid $2h$? $e = 0$!

How do we get to grid $2h$? Stay tuned…
Coarse-grid correction

2-grid

- Relax on $Au = f$ on $\Omega^h$ to get an approximation $v^h$.
- Compute $r = f - Av^h$.
- Transfer $Ae = r$ to $\Omega^{2h}$ somehow & relax on it to obtain an approximation to the error, $e^{2h}$.
- Correct the approximation $v^h \leftarrow v^h + I_{2h}^h e^{2h}$.

This is the essence of multigrid.

We need a way to transfer $Ae = r$ to $\Omega^{2h}$.

A way to coarsen $Ae = r$

- Assume we’ve relaxed so much that $e$ is smooth.
- Ansatz: $e = P v^{2h}$ for some coarse-grid $v^{2h}$.
- How do we characterize $e$ so we can hope to compute it?
  $$Ae = r \quad \Rightarrow \quad A P v^{2h} = r$$
  $7x7 \ 7x3 \ 3x1 = 7x1$

- Too many equations now & too few unknowns!
- How about just eliminating every other equation?
- How about multiplying both sides by some $3x7$ matrix?
  $$P^T A^2h P v^{2h} = P^T r$$
  $3x7 \ 7x7 \ 7x3 \ 3x1 = 3x1$

1-D restriction by injection

- Mapping from the fine grid to the coarse grid:
  $$I_{2h}^h : \Omega^h \rightarrow \Omega^{2h}$$

- Let $v^h, v^{2h}$ be defined on $\Omega^h, \Omega^{2h}$. Then
  $$I_{2h}^h v^h = v^{2h}$$
  $\Omega^h \rightarrow \Omega^{2h}$

where $v_i^{2h} = v_{2i}^h$.

1-D restriction by full weighting

- Let $v^h, v^{2h}$ be defined on $\Omega^h, \Omega^{2h}$. Then
  $$I_{2h}^h v^h = v^{2h}$$
  where
  $$v_i^{2h} = \frac{1}{4} \left( v_{i-1}^h + 2v_i^h + v_{i+1}^h \right)$$
1-D restriction (full-weighting)

- $R = I^h_2$ is a linear operator: $\mathcal{R}^N \rightarrow \mathcal{R}^{(N-1)/2}$.

- $N = 7$:
  \[
  \begin{pmatrix}
  1/4 & 1/2 & 1/4 & 1/4 & 1/4 & 1/2 & 1/4 \\
  1/4 & 1/2 & 1/4 & 1/4 & 1/2 & 1/4 & 1/4 \\
  \end{pmatrix}
  \begin{pmatrix}
  v_i^1 \\
  v_i^2 \\
  v_i^3 \\
  v_i^4 \\
  v_i^5 \\
  v_i^6 \\
  v_i^7 \\
  \end{pmatrix}
  =
  \begin{pmatrix}
  v_i^{2h}^1 \\
  v_i^{2h}^2 \\
  \end{pmatrix}.
  \]

- $I^h_2$ has rank $\frac{N-1}{2}$, so $\text{dim}(\eta(R)) = \frac{N+1}{2}$.

Look at the columns of $R$ associated with grid $2h$.

Prolongation & restriction are often nicely related

- For the 1-D examples, linear interpolation & full weighting are
  
  \[
  I^h_2 = \frac{1}{2} \begin{pmatrix}
  1 & 1 & 1 & 1 \\
  2 & 1 & 1 & 1 \\
  \end{pmatrix}
  \]
  \[
  I^{2h}_h = \frac{1}{4} \begin{pmatrix}
  1 & 1 & 1 & 1 \\
  1 & 1 & 1 & 1 \\
  \end{pmatrix}
  \]

- So they're related by the variational condition
  \[
  I^h_2 = c \ (I^{2h}_h)^T 
  \]
  for $c$ in $\mathcal{R}$.

2-D prolongation

- $v^h_{2i,2j} = v^{2h}_{ij}$
- $v^h_{2i+1,2j} = \frac{1}{2} (v^{2h}_{ij} + v^{h}_{i+1,j})$
- $v^h_{2i,2j+1} = \frac{1}{2} (v^{2h}_{ij} + v^{h}_{i,j+1})$
- $v^h_{2i+1,2j+1} = \frac{1}{4} (v^{2h}_{ij} + v^{h}_{i+1,j} + v^{h}_{i,j+1} + v^{h}_{i+1,j+1})$

"Get From" interpolation

\[
\begin{pmatrix}
1/2 & 1/2 \\
1/4 & 1/4 \\
\end{pmatrix}
\]

Centered over a fine-grid point $\bullet$.

We denote the operator by using a "give to" stencil $[\cdot]$.
Centered over a C-point $\bullet$, it shows what fraction of the C-point's value contributes to a neighboring F-point $\bullet$. 

CU-Boulder
2-D restriction (full weighting)

\[
\begin{bmatrix}
1 & 1 & 1 \\
16 & 8 & 16 \\
1 & 1 & 1 \\
8 & 4 & 8 \\
1 & 1 & 1 \\
16 & 8 & 16
\end{bmatrix}
\]

We denote the operator by using a "get from" stencil [ ]. Centered over a C-point ●, it shows what fraction of the value of the neighboring F-point ● contributes to the value at the C-point.

Now we put all these ideas together

- Nested Iteration
  - effective on smooth solution (components).
- Relaxation
  - effective on oscillatory error (components).
- Residual Equation
  - characterizes the error.
  - enables nested iteration for smooth error (components)!!
- Prolongation (variables) & Restriction (equations)
  - provides pathways between coarse & fine grids.

2-grid coarse-grid correction

\[ v^h \leftarrow CG( v^h, f^h), \alpha_1, \alpha_2 \]

1) Relax \( \alpha_1 \) times on \( A^h v^h = f^h \) on \( \Omega^h \) with arbitrary initial guess \( v^h \). If \( h = h_{\text{coarsest}} \), then go to 6.
2) Compute \( r^h = f^h - A^h v^h \).
3) Compute \( r^{2h} = I^h r^h \).
4) "Solve" \( A^{2h} e^{2h} = r^{2h} \) on \( \Omega^{2h} \).
5) Correct fine-grid solution \( v^h \leftarrow v^h + I^h e^{2h} \).
6) Relax \( \alpha_2 \) times on \( A^h v^h = f^h \) on \( \Omega^h \).

What is \( e^{2h} \) here?
What is $A^{2h}$?

- For this scheme to work, we must have $A^{2h}$, a coarse-grid operator. For the moment, we will simply assume that $A^{2h}$ is “the coarse-grid version” of the fine-grid operator $A^h$.

- Later we’ll return to the question of constructing $A^{2h}$.

**V-cycle (recursive form)**

$$v^h \leftarrow MV^h(v^h, f^h), \alpha_1, \alpha_2$$

1) Relax $\alpha_1$ times on $A^h u^h = f^h$ with initial $v^h$ given.

2) If $\Omega^h$ is the coarsest grid, go to 4:
   
   else:
   
   - $f^{2h} \leftarrow f^h - A^h v^h$
   - $v^{2h} \leftarrow 0$
   - $v^{2h} \leftarrow MV^{2h}(v^{2h}, f^{2h})$

3) Correct: $v^h \leftarrow v^h + I_{2h}^h v^{2h}$.

4) Relax $\alpha_2$ times on $A^h u^h = f^h$ with initial guess $v^h$.

**How do we “solve” the coarse-grid residual equation? Recursion!**

**Outline**

**Chapters 1-5:**
- Model Problems
- Basic Iterative Methods
  - Convergence tests
  - Analysis
- Elements of Multigrid
  - Relaxation
  - Coarsening
- Implementation
  - Complexity
  - Diagnostics
- Some Theory
  - Spectral vs. algebraic

**Chapters 6-10:**
- Nonlinear Problems
  - Full approximation scheme
- Selected Applications
  - Neumann boundaries
- Algebraic multigrid problems
  - Variable meshes
  - Variable coefficients
- Algebraic multigrid (AMG)
  - Matrix coarsening
- Multilevel Adaptive Methods
  - FAC
- Finite Elements
  - Variational methodology

**Homework Due**
4. Implementation

Storage cost: \( v^h \) & \( f^h \) on each level.

- In 1-D, a coarse grid has about half as many points as the fine grid.
- In 2-D, a coarse grid has about one-fourth as many points as the fine grid.
- In \( d \)-dimensions, a coarse grid has about \( 2^{-d} \) as many points as the fine grid.
- Total storage cost: \( N^d (1 + 2^{-d} + 2^{-2d} + 2^{-3d} + \ldots + 2^{-Md}) < \frac{N^d}{1 - 2^{-d}} \) less than \( 2, \frac{4}{3}, \) & \( \frac{8}{7} \) the cost of storage on the fine grid for 1-D, 2-D, & 3-D problems, respectively.

Computational cost

- Let one Work Unit (WU) be the cost of one relaxation sweep on the fine grid.
- Ignore the cost of restriction & interpolation (typically about 20% of the total cost).
- Consider a V-cycle with 2 pre-coarse-grid correction sweeps (\( \alpha_1 = 2 \)) & 1 post-coarse-grid correction sweep (\( \alpha_2 = 1 \)).
- Cost of a V-cycle (in WUs):
  \[
  3 \left(1 + 2^{-d} + 2^{-2d} + 2^{-3d} + \ldots + 2^{-Md}\right) < \frac{3}{1 - 2^{-d}}
  \]
- Cost is about \( 2, \frac{4}{3}, \) & \( \frac{8}{7} \times 3 \) WUs per V-cycle in 1-, 2-, & 3-dimensional problems, respectively.

Convergence analysis

- First, a heuristic argument:
  - The convergence factor for the oscillatory error modes (smoothing factor) is small & bounded uniformly in \( h \).
  - Multigrid focuses the relaxation process on attenuating the oscillatory components on each level.

  \[
  \text{smoothing factor} = \max |\lambda_n(R)| \quad \text{for } N/2 \leq k \leq N.
  \]

  \[
  \Rightarrow \text{The overall multigrid convergence factor is small & bounded uniformly in } h!
  \]

  Bounded uniformly in \( h \neq \) independent of \( h \).

Revisiting the model problem

- The BVP:
  \[
  -u''(x) = f(x) \quad 0 < x < 1, \quad \sigma \geq 0
  \]
  \[
  u(0) = u(1) = 0
  \]

  \[
  \text{The finite difference scheme:}
  \]
  \[
  \frac{-v_{i-1} + 2v_i - v_{i+1}}{h^2} = f_i \quad i = 1, \ldots, N
  \]
  \[
  v_0 = v_{N+1} = 0
  \]

  \[
  \text{Truncation error:}
  \]
  \[
  u^{(h)} = \text{exact PDE solution vector}
  \]
  \[
  \frac{-u^{(h)}_{i-1} + 2u^{(h)}_i - u^{(h)}_{i+1}}{h^2} + O(h^2) = f_i \quad i = 1, \ldots, N
  \]
Actual error

\[ Au^{(h)} + O(h^2) = f \]
\[ \Rightarrow \quad Au^{(h)} = f + O(h^2) \]
\[ \Rightarrow \quad Au = f \quad (u = \text{discrete sol'n}) \]
\[ \Rightarrow \quad A(u^{(h)} - u) = O(h^2) \]
\[ \Rightarrow \quad AE = O(h^2). \]

So \[ ||E|| = ||A^{-1}O(h^2)|| \]
\[ \leq ||A^{-1}|| ||O(h^2)|| \]
\[ = \lambda_{\text{max}}(A^{-1}) O(h^2) = O(h^2)/\lambda_{\text{min}}(A) \]

or \[ ||E|| = O(h^2). \]

We can satisfy the convergence objective by imposing two conditions

1) \[ ||E|| \leq \varepsilon/2. \] Achieve this condition by choosing an appropriately small grid spacing \( h \):

\[ Kh^p = \varepsilon/2. \]

2) \[ ||e^h|| \leq \varepsilon/2. \] Achieve this condition by iterating until

\[ ||e^h|| \leq \varepsilon/2 = Kh^p \] on grid \( h \); then we've

converged to the level of discretization error.

Convergence to the level of discretization error

- Use an MV scheme with convergence factor \( \gamma < 1 \) bounded uniformly in \( h \) (fixed \( \alpha_1 \) & \( \alpha_2 \)).
- Assume a \( d \)-dimensional problem on an \( N \times N \ldots \times N \) grid with \( h = N^{-1} \).
- Initial error is \[ ||e^h|| = ||u^h - 0|| = ||u^h|| = O(1). \]
- Must reduce this to \[ ||e^h|| = O(h^p) = O(N^{-p}). \]
- We can determine the number of V-cycles needed for this, if we can bound the convergence factor, \( \gamma \).
Work to converge to the level of discretization error

- Using $\theta$ V-cycles with convergence factor $\gamma$ gives an overall convergence factor of $\gamma^\theta$.
- We therefore have $\gamma^\theta = O(N^{-p})$, or $\theta = O(\log N)$.
- Since 1 V-cycle costs $O(1)$ WUs & 1 WUs is $O(N^2)$, then converging to the level of discretization error using the MV method cost $O(N^2 \log N)$.
- This compares to fast direct methods (fast Poisson solvers). But multigrid can do even better.

<table>
<thead>
<tr>
<th>$\gamma$</th>
<th>$\gamma^\theta$</th>
<th>$\theta$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>0.9</td>
<td>0.1</td>
</tr>
<tr>
<td>0.2</td>
<td>0.4</td>
<td>0.3</td>
</tr>
<tr>
<td>0.3</td>
<td>0.1</td>
<td>0.9</td>
</tr>
<tr>
<td>0.4</td>
<td>0.0</td>
<td>0.0</td>
</tr>
</tbody>
</table>

Numerical results

MV cycling

Shown are the results of 15 V(2,1)-cycles. We display, after each cycle, residual norms, total error norms, & ratios of these norms to their values after the previous cycle.

$N = 16, 32, 64, 128$.

- Bounds like $||e_{n+1}|| \leq \gamma ||e_n||$ & $||u^{(h)} - u^h|| = O(h)$ are only just that—bounds!
- If you see behavior that suggests that these bounds are sharp (e.g., halving $h$ halves the discretization error), then great. If you don’t see this behavior, don’t assume things are wrong.
- Think about this: $O(h^2) = O(h)$ but generally $O(h) \neq O(h^2)$ !!!

A warning about bounds

<table>
<thead>
<tr>
<th>$\gamma$</th>
<th>$\gamma^\theta$</th>
<th>$\theta$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>0.9</td>
<td>0.1</td>
</tr>
<tr>
<td>0.2</td>
<td>0.4</td>
<td>0.3</td>
</tr>
<tr>
<td>0.3</td>
<td>0.1</td>
<td>0.9</td>
</tr>
<tr>
<td>0.4</td>
<td>0.0</td>
<td>0.0</td>
</tr>
</tbody>
</table>

Numerical example

- Consider the 2-D model problem (with $\sigma = 0$):
  $$ u_{xx} - u_{yy} = 2[(1 - 6x^2)y^2(1 - y^2) + (1 - 6y^2)x^2(1 - x^2)] $$
  in the unit square, with $u = 0$ Dirichlet boundary.
- The solution to this problem is
  $$ u(x,y) = -(x^4 - x^2)(y^4 - y^2). $$
- We examine effectiveness of MV cycling to solve this problem on $(N+1)\times(N+1)$ grids $[(N-1) \times (N-1)$ interior points] for $N = 16, 32, 64, 128$. 
You want to approximate $u^h$.

A good iteration is the V-cycle.

What's a good way to start it?

Can you do better than $v^h \leftarrow 0$?

Start on the coarse grid!

Use nested iteration for the V-cycle!

**Look** again at nested iteration

- **Idea:** It’s cheaper to solve a problem (fewer iterations) if the initial guess is good.

- How to get a good initial guess:
  - “Solve” the problem on the coarse grid first.
  - Interpolate the coarse solution to the fine grid.

- Now, let’s use the V-cycle as the solver on each grid level! This defines the **Full Multigrid (FMG)** cycle.

**Full multigrid (FMG)**

$v^h \leftarrow \text{FMG}(f^h)$

- Initialize $f^h, f^{2h}, f^{4h}, \ldots, f^H$

- Solve on coarsest grid

  $$v^H = (A^H)^{-1}f^H$$

- Interpolate initial guess

  $$v^{2h} \leftarrow I_{2h}^{2h}v^{4h}$$

- Perform V-cycle

  $$v^{2h} \leftarrow MV^{2h}(v^{2h}, f^{2h})$$

- Interpolate initial guess

  $$v^h \leftarrow I_{2h}^{h}v^{2h}$$

- Perform V-cycle

  $$v^h \leftarrow MV^h(v^h, f^h)$$

**FMG-cycle**

- Restriction

- Interpolation

- High-Order Interpolation?
FMG-cycle (recursive form)

\[ v^h \leftarrow \text{FMG}(f^h), \eta \]

1) Initialize \( f^h, f^{2h}, ..., f^H \).

2) If \( h = H \), then go to 4 (where MV is a direct solve);
   else: \( v^{2h} \leftarrow \text{FMG}(f^{2h}) \).

3) Set initial guess: \( v^h \leftarrow \Pi_{2h} v^{2h} \).

4) Perform \( v^h \leftarrow \text{MV}(v^h, f^h) \) \( \eta \) times.

Has discretization error been reached by FMG?

If discretization error is achieved, then \( ||e^h|| = O(h^2) \)
& the error norms at the “solution” points in the cycle
should form a Cauchy sequence:

\[ ||e^h|| \approx 0.25 ||e^{2h}|| \]

We need to be more careful...

FMG cycle cost

One \( V(2,1) \)-cycle is performed per level, at a cost of
\[ \frac{3}{(1 - 2^{-d})} \] WUs per grid (where the WU is for the
size of the finest grid involved).

The size for the WU for coarse-grid \( j \) is \( 2^{-jd} \) times
the size for the WU for the fine grid (grid 0).

Hence, the cost of the FMG\((2,1)\) cycle in WUs is less than

\[ \left[ \frac{3}{(1 - 2^{-d})} \right] \left( 1 + 2^{-d} + 2^{-2d} + ... \right) = \frac{3}{(1 - 2^{-d})^2}. \]

\( d = 1: \) 12 WUs; \( d = 2: \) 16/3 WUs; \( d = 3: \) 192/49 WUs.

Comparing the right things

- Problem: We are thinking that \( u^{2h} \) approximates \( u^h \)
to order \( O(h^2) \), when all we really know is that \( u^h \)
approximates \( u^{(h)} \) to order \( O(h^2) \) (any \( h \)).

- We know that \( u^{(2h)} \) and \( u^{(h)} \) are the “same”, right?
So, if \( u^{2h} \) approximates \( u^{(2h)} \) to order \( O(4h^2) \) and
\( u^h \) approximates \( u^{(h)} \) to order \( O(h^2) \), shouldn't \( u^{2h} \)
approximate \( u^h \) to order \( O(4h^2) \)? How, exactly?

- When we interpolate \( u^{2h} \) to grid \( h \), what errors does
interpolation introduce?

- Sorting out these comparison is a bit technical.
- In other words, here comes the algebra...
**Interpolation stability**  
how interpolation affects error

- Property:  \[ \| P e^{2h} \| \leq \beta \| e^{2h} \| \]
- Reasoning:  
  \[
  \| P e^{2h} \| = \| P^{T} P \|^{1/2} \| e^{2h} \| \\
  P^{T} P = \begin{pmatrix}
  \frac{1}{4} & \frac{1}{2} & \frac{1}{4} \\
  \frac{1}{2} & 1 & \frac{1}{2} \\
  \frac{1}{4} & \frac{1}{2} & \frac{1}{4}
\end{pmatrix} \begin{pmatrix}
  \frac{1}{4} & \frac{1}{2} & \frac{1}{4} \\
  \frac{1}{2} & 1 & \frac{1}{2} \\
  \frac{1}{4} & \frac{1}{2} & \frac{1}{4}
\end{pmatrix} = \begin{pmatrix}
  \frac{1}{4} & \frac{1}{2} & \frac{1}{4} \\
  \frac{1}{2} & 1 & \frac{1}{2} \\
  \frac{1}{4} & \frac{1}{2} & \frac{1}{4}
\end{pmatrix}
\]

\[ \Rightarrow \| P^{T} P \|^{1/2} \leq \sqrt{2} \quad \text{In practice, } \beta = 1. \]

**FMG accuracy**  
\[ \| e^{h} \| \leq K h^{2} \]

Assume:
\[
\| e^{2h} \| \leq K (2h)^{2} \quad \text{induction hypothesis}
\]
\[
\| u^{h} - P u^{2h} \| \leq \alpha K h^{2} \quad \text{approximation property (}\alpha = 5\text{)}
\]
\[
\| P v^{2h} \| \leq \beta \| v^{2h} \| \quad \text{interpolation stability (}\beta = 1\text{)}
\]

Triangle inequality:
\[
\| e^{h} \| = \| u^{h} - P v^{2h} \| \\
\leq \| u^{h} - P u^{2h} \| + \| P (u^{2h} - v^{2h}) \| \\
\leq \alpha K h^{2} + \beta(2h)^{2} \\
= \alpha(1 + 4\beta)K h^{2}
\]

\[ \Rightarrow \| e^{h} \| \leq 9\alpha K h^{2} \]
So we need only reduce \( \| e^{h} \| \) by "0.1"!!!
FMG results
3 FMG cycles & comparison with MV cycle results

\[ \| \mathbf{e} \|_h = h \| \mathbf{u}^{(h)} - \mathbf{v}^{h} \|_2 \]

scaled discrete total error

<table>
<thead>
<tr>
<th>( N )</th>
<th>FMG(1,0) ( | \mathbf{e} |_h ) ratio</th>
<th>FMG(1,1) ( | \mathbf{e} |_h ) ratio</th>
<th>FMG(2,1) ( | \mathbf{e} |_h ) ratio</th>
<th>FMG(1,1) WU</th>
<th>V(2,1) cycles</th>
<th>V(2,1) WU</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>5.86e−03</td>
<td>5.86e−03</td>
<td>5.86e−03</td>
<td>7/2</td>
<td>3</td>
<td>12</td>
</tr>
<tr>
<td>4</td>
<td>5.37e−03</td>
<td>0.017</td>
<td>2.93e−03</td>
<td>4.24</td>
<td>2/3</td>
<td>0.347</td>
</tr>
<tr>
<td>8</td>
<td>2.78e−03</td>
<td>0.018</td>
<td>9.12e−04</td>
<td>0.267</td>
<td>6.85e−04</td>
<td>0.329</td>
</tr>
<tr>
<td>16</td>
<td>1.19e−03</td>
<td>0.027</td>
<td>2.52e−04</td>
<td>1.03e−04</td>
<td>1.72e−04</td>
<td>0.257</td>
</tr>
<tr>
<td>32</td>
<td>4.70e−04</td>
<td>0.305</td>
<td>6.00e−05</td>
<td>2.58e−05</td>
<td>4.00e−05</td>
<td>0.233</td>
</tr>
<tr>
<td>64</td>
<td>1.77e−04</td>
<td>0.377</td>
<td>1.36e−05</td>
<td>6.44e−06</td>
<td>9.36e−06</td>
<td>0.234</td>
</tr>
<tr>
<td>128</td>
<td>6.42e−05</td>
<td>0.366</td>
<td>3.12e−06</td>
<td>1.61e−06</td>
<td>2.96e−06</td>
<td>0.241</td>
</tr>
<tr>
<td>256</td>
<td>2.35e−05</td>
<td>0.399</td>
<td>7.35e−07</td>
<td>3.35</td>
<td>5.56e−07</td>
<td>0.246</td>
</tr>
<tr>
<td>512</td>
<td>8.25e−06</td>
<td>0.454</td>
<td>1.77e−07</td>
<td>0.241</td>
<td>1.38e−07</td>
<td>0.248</td>
</tr>
<tr>
<td>1024</td>
<td>2.90e−06</td>
<td>0.452</td>
<td>4.35e−08</td>
<td>0.245</td>
<td>3.41e−08</td>
<td>0.249</td>
</tr>
<tr>
<td>2048</td>
<td>1.02e−06</td>
<td>0.451</td>
<td>1.08e−08</td>
<td>0.347</td>
<td>8.38e−09</td>
<td>0.350</td>
</tr>
</tbody>
</table>

Diagnostic tools
for debugging the code, the method, the problem

- Finding mistakes in codes, algorithms, concepts, & the problem itself challenges our scientific abilities.
- This challenge is especially tough for multigrid:
  - Interactions between multilevel processes can be very subtle.
  - It’s often not easy to know how well multigrid should perform.
- Achi Brandt:
  - “The amount of computational work should be proportional to the amount of real physical changes in the computed solution.”
  - “Stalling numerical processes must be wrong.”
- The “computational culture” is best learned by lots of experience & interaction, but some discussion helps.

Tool # 1: Be methodical

- Modularize your code.
- Test the algebraic solver first.
- Test the discretization next.
- Test the FMG solver last.
- Beware of boundaries, scales, & concepts.
- Ask whether the problem itself is well posed.

Tool # 2: Start simply

- Start from something that already works if you can.
- Introduce complexities slowly & methodically, testing thoroughly along the way.
- Start with a very coarse fine grid (no oxymoron intended).
- Start with two levels if you can, using a direct solver or lots of cycles on coarse grids if nothing else.

If you find trouble, your first job is to find the simplest case where that trouble is still evident!!!
Tool # 3: Expose trouble

Start simply, but don’t let niceties mask trouble:

- Set reaction/Helmholtz terms to zero.
- Take infinite or very big time steps.
- Don’t take 1-D too seriously, not even 2-D.

Tool # 4: Test fixed point property

Relaxation shouldn’t alter the exact solution of the linear system (up to machine precision).

- Create a right side: \( f = Au^* \) with \( u^* \) given.
- Make sure \( u^* \) satisfies the right boundary conditions.
- Test relaxation starting with \( u^* \): Is \( r = 0 \), is it zero after relaxation, does \( u^* \) change?
- Test coarse-grid correction starting with \( u^* \): Is the correction zero?

Tool # 5: Test on \( Au=0 \)

- The exact solution \( u^* = 0 \) is known!
- Residual norm \( ||Au|| \) & error norm \( ||u|| \) are computable.
- Norms \( ||Au|| \) & \( ||u|| \) should eventually decrease steadily with a rate that might be predicted by mode analysis.
- Multigrid can converge so fast that early stalling suggests trouble when it’s just that all machine-representable numbers in a nonzero \( u^* \) have already been computed! Computing \( r = f - Au \) & updating \( u \) shouldn’t have trouble with machine precision if you have \( u^* = 0 \) & thus \( f = 0 \).

Tool # 6: Zero out residual

- Using a normal test, try multiplying the residual by 0 before you go to the coarse grid.
- Check to see that the coarse-grid corrections are 0.
- Compare this test with a relaxation-only test--the results should be identical.
Tool # 7: Print out residual norms

- Use the discrete $L^2$ norm:
  \[ ||r||_h = (h^d \sum r_i^2)^{1/2} = h^{d/2} ||r||_2 \]
- Output $||r||_h$ after each pre- & post-relaxation sweep.
- These norms should decline to zero steadily for each $h$.
- The norm after post-relaxation should be consistently smaller than after pre-relaxation--by the predictable convergence factor at least.

A warning about residuals

- Residuals could mask large smooth errors:
  - If $||e||=O(h^2)$, then $||r||=O(h^2)$ for smooth $e$, but $||r||=O(1)$ for oscillatory $e$. ($\lambda_N/\lambda_1 = O(h^2)$!!)
  - This means that if we are controlling the error by monitoring $||r||$, if we don’t know the nature of $e$, \& if we don’t want to risk large $e$ by requiring only $||r||=O(1)$, then we may have to work very hard to make $||r||=O(h^2)$. (We could in effect be trying to make $||e||=O(h^4)$!!!)
- Multigrid tends to balance the errors, so $||r||=O(h^2)$ tends to mean $||e||=O(h^2)$.

Tool # 8: Graph the error

- Run a test on a problem with known solution ($Au = 0$).
- Plot algebraic error before \& after fine-grid relaxation.
- Is the error oscillatory after coarse-grid correction?
- Is the error much smoother after fine-grid relaxation?
- Are there any strange characteristics near boundaries, interfaces, \& other special phenomena?

Tool # 9: Test two-level cycling

- Replace the coarse-grid V-cycle recursive call with a direct solver if possible, \& iterate many times with some method known to “work” (test $||r||$ to be sure it’s very small), \& use many recursive V-cycle calls.
- This can be used to test performance between two coarser levels, especially if residual norm behavior identifies trouble on a particular level.
Tool # 10: Beware of boundaries

- Boundaries usually require special treatment of the stencils, intergrid transfers, & sometimes relaxation.
- Special treatment often means special trouble, typically exposed in later cycles as it begins to infect the interior.
- Replace the boundary by periodic or Dirichlet conditions.
- Relax more at the boundary, perhaps using direct solvers.
- Make sure your coarse-grid approximation at the boundary is guided by good discretization at the fine-grid boundary.

Tool # 11: Test for symmetry

- If your problem is symmetric or includes a symmetric case, test for it.
- Check symmetry of the fine-grid & coarse-grid matrices: are $a_{ij}$ & $a_{ji}$ relatively equal (to machine precision).
- Be especially watchful for asymmetries near boundaries.

Tool # 12: Check for compatibility

- This is a bit ahead of schedule, but consider the problem $-u'' = f$ with $u'(0) = u'(1) = 0$.
- It’s singular: If $u = 1$, then $-u'' = 0$ & $u'(0) = u'(1) = 0$.
- It’s is solvable iff $f \in \text{Range}(\partial_{xx}) = \eta^1(\partial_{xx}) = \{1\}^1$ or $f \perp 1$.
- First fix the grid $h$ right side: $f^h \leftarrow f^h - \langle f^h, 1 \rangle/\langle 1,1 \rangle$.
- Do this on coarse grids too: $f^{2h} \leftarrow f^{2h} - \langle f^{2h}, 1 \rangle/\langle 1,1 \rangle$.
- Uniqueness is also a worry: $u^h \leftarrow u^h - \langle u^h, 1 \rangle/\langle 1,1 \rangle$.

Tool # 13: Test for linearity

- Again, this is a bit ahead of schedule, but if you’re writing a nonlinear FAS code, it should agree with the linear code when you test it on a linear problem. Try it.
- Move gradually to the target nonlinear test problem by putting a parameter in front of the nonlinear term, then running tests as the parameter changes slowly from 0 to 1.
Tool # 14: Use a known PDE solution

- Set up the source term ($f = Au^*$ in $\Omega$) & data ($g = u^*$ on $\Gamma$).
- Do multigrid results compare qualitatively with sampled $u^*$?
- Monitor $||u^* - u^h||_h$.
- Test a case with no discretization error (let $u^*$ = 2nd degree polynomial). The algebraic error should tend steadily to 0.
- Test a case with discretization error (let $u^*$ have a nonzero 3rd derivative). The algebraic error should decrease rapidly at first, then stall at discretization error level. Check error behavior as you decrease $h$. Does it behave like $O(h^2)$ ($h$ halved $\Rightarrow$ error halved) or whatever it should behave like?

Tool # 15: Test FMG accuracy

- Make sure first that the algebraic solver converges as predicted, with uniformly bounded convergence factors.
- Test the discretization using Tool # 14.
- Compare FMG total error to discretization error for various $h$. You might need to tune the FMG process here (play with the number of cycles & relaxation sweeps).

Computing assignments

- Document: norms/weights, $V(v_1, v_2)$, errors, labels (table, graph)
- Use various scenarios:
  - $Ax = 0, Ax = f$, varying $N$ & $v_i$ & $\omega$, Jacobi/Gauss-Seidel
- Thoroughly test:
  - don't stop until you get what you expect.
  - compare with known solution, text, others.
  - study discretization & algebraic errors.
  - report on “asymptotic” factors.
- Be kind to the reader:
  - code = zzz...
  - tables = +
  - tables & graphs = ++
  - tables & graphs & discussion (clear, concise) = +++
- Discuss, discuss, discuss:
  - what do you see & think? what did you learn?

Outline

Chapters 1-5:
- Model Problems
- Basic Iterative Methods
  - Convergence tests
  - Analysis
- Elements of Multigrid
  - Relaxation
  - Coarsening
- Implementation
  - Complexity
  - Diagnostics
- Some Theory
  - Spectral vs. algebraic

Chapters 6-10:
- Nonlinear Problems
  - Full approximation scheme
- Selected Applications
  - Neumann boundaries
  - Isotropic problems
  - Variable meshes
  - Variable coefficients
- Algebraic Multigrid (AMG)
  - Matrix coarsening
- Multilevel Adaptive Methods
  - FAC
- Finite Elements
  - Variational methodology
5. Some theory
   What is $A^{2h}$?
   - Recall the 2-grid coarse-grid correction scheme:
     1) Relax on $A^h u^h = f^h$ on $\Omega^h$ to get $v^h$.
     2) Compute $f^{2h} = I^{2h}_h (f^h - A^h v^h)$.
     4) Solve $A^{2h} u^{2h} = f^{2h}$ on $\Omega^{2h}$.
     5) Correct fine-grid solution $v^h \leftarrow v^h + I^{2h}_h u^{2h}$.
   - Assume that $e^h \in \text{Range}(I^{2h}_h)$. Then the residual equation can be written
     \[ r^h = A^h e^h = A^h I^{2h}_h u^{2h} \text{ for some } u^{2h} \in \Omega^{2h}. \]
     This characterizes $u^{2h}$, but with too many equations.
   - How does $A^h$ act on $\text{Range}(I^{2h}_h)$?

Building $A^{2h}$: The Galerkin condition

- The residual equation on the coarse grid is
  \[ I^{2h}_h A^h I^{2h}_h u^{2h} = I^{2h}_h r^h \]
- We thus identify the coarse-grid operator as
  \[ A^{2h} = I^{2h}_h A^h I^{2h}_h \]
- How do we know $\text{RAP}$ is symmetric?
  If $P^T = \alpha R$ (so that $R^T = (1/\alpha) P$), then
  \[ (\text{RAP})^T = P^T A^T R^T = \alpha (1/\alpha) \text{RAP} = \text{RAP}. \]

How does $A^h$ act on $\text{Range}(I^{2h}_h)$?

Thus, the odd rows of $A^h I^{2h}_h$ are zero (1-D only) & $r_{2i+1} = 0$.
So we keep the even rows of $A^h I^{2h}_h$ for the residual equations
on $\Omega^{2h}$. We do this by applying restriction, either injection or
full weighting: $I^{2h}_h A^h I^{2h}_h u^{2h} = I^{2h}_h r^h$.
We use full weighting from now on unless otherwise stated.

Computing the $i^{\text{th}}$ row of $A^{2h}$

- Compute $A^{2h} \hat{e}^{2h}_i$, where $\hat{e}^{2h}_i = (0, 0, ..., 0, 1, 0, ..., 0)^T$.
  Why is this the $i^{\text{th}}$ row of $A^{2h}$?

\[
\begin{array}{ccccc}
 0 & 1 & 0 \\
-\frac{1}{2h^2} & 0 & \frac{1}{h^2} & 0 \\
-\frac{1}{(2h)^2} & \frac{2}{(2h)^2} & -\frac{1}{(2h)^2} & 0 \\
\end{array}
\]

CU-Boulder 133 of 322

CU-Boulder 134 of 322
The $i^{th}$ row of $A^{2h}$ looks a lot like the $i^{th}$ row of $A^h$!

- The $i^{th}$ row of $A^{2h}$ is $\frac{1}{(2h)^2} [-1 2 -1]$, which is the $\Omega^{2h}$ version of $A^h$.
- Note that IF relaxation on $\Omega^h$ leaves only error in the range of interpolation, then solving $A^{2h} u^{2h} = f^{2h}$ determines the error exactly!
- This is generally not feasible, but this logic leads to a very plausible representation for $A^{2h}$.

Variational properties of coarsening

- The definition for $A^{2h}$ that resulted from the foregoing line of reasoning is useful for both theoretical & practical reasons. Together with the commonly used relationship between restriction & prolongation, we have the variational properties:

$$A^{2h} = I_h^{2h} A^h I_{2h}^h \quad \text{Galerkin Condition}$$
$$I_{2h}^h = c (I_h^{2h})^T \quad \text{for } c \in \mathcal{R}.$$

Properties of restriction

in a little more detail...

- Full Weighting: $I_h^{2h} : \Omega^h \rightarrow \Omega^{2h}$ or $I_h^{2h} : \mathcal{R}^N \rightarrow \mathcal{R}^{(N-1)/2}$
- $N = 7$:
  $$I_h^{2h} = \frac{1}{4} \begin{bmatrix} 1 & 2 & 1 \\ 1 & 2 & 1 \\ 1 & 2 & 1 \end{bmatrix}_{3 \times 7}$$
- $I_h^{2h}$ has rank $\frac{N-1}{2}$ & null space $\eta(I_h^{2h})$ with dim $\frac{N+1}{2}$.

Spectral properties of restriction

- How does $I_h^{2h}$ act on the eigenvectors of $A^h$?
- Consider $w_k^{h} = \sin \left( \frac{jk\pi}{N+1} \right)$, $1 \leq k \leq N$, $0 \leq j \leq N$.
- A little algebra & trigonometry shows that
  $$\left( I_h^{2h} w_k^{h} \right)_j = \cos^2 \left( \frac{k\pi}{2(N+1)} \right) w_{k,j}^{2h}$$
  for $1 \leq k \leq (N-1)/2$. 
**Spectral properties (cont'd)**

i.e., \( I^2_{(N+1-k)} [k^{th} \text{ mode on } \Omega^h] = -s_k [k^{th} \text{ mode on } \Omega^{2h}] \)

\[ \Omega^h: \ N = 7, \ k = 6 \]

\[ \Omega^{2h}: \ N = 3, \ k = 2 \]

**Spectral properties (cont'd)**

- Let \( k' = N + 1 - k \) for \( 1 \leq k \leq (N - 1)/2 \), so that \( (N - 3)/2 \leq k' \leq N \).

- A little algebra & trigonometry shows that

\[ (I^2_h w^h_{(k')})_j = -\sin^2\left(\frac{k\pi}{2(N+1)}\right) w^{2h}_{k,j} \]

\[ \equiv s_k w^{2h}_{k,j} \]

**Spectral properties (cont'd)**

- Summarizing:

\[ \begin{cases} 
I^2_h w^h_{k} = c_k w^{2h}_{k} & \text{for } 1 \leq k \leq \frac{N-1}{2} \\
I^2_h w^h_{k'} = -s_k w^{2h}_{k'} & k' = N + 1 - k \\
I^2_h w^h_{(N+1)/2} = 0 
\end{cases} \]

**Spectral properties (cont'd)**

- Complementary modes:

\[ W_k = \text{span} \{ w^h_{k}, w^{2h}_{k} \} \]

\[ I^2_h W_k \rightarrow \{ w^{2h}_{k} \}_\infty \]
Null space of restriction

• Observe that \( \eta(I_h^{2h}) = \text{span}(A^h \hat{e}^h) \), where \( i \) is odd & \( \hat{e}^h \) is the \( i \)th unit vector.

• Let \( \eta_i = A^h \hat{e}^h \).

• While the \( \eta_i \) looks oscillatory, it generally contains all Fourier modes of \( A^h \):

\[
\eta_i = \sum_{k=1}^{N} a_k w_k, \quad a_k \neq 0
\]

• All the Fourier modes of \( A^h \) are needed to represent the null space of restriction!

Properties of interpolation

• Interpolation: \( I_{2h}^h : \Omega^{2h} \rightarrow \Omega^h \) or \( I_{2h}^h : \mathcal{R}(N-1)/2 \rightarrow \mathcal{R}^N \)

• \( N = 7 \):

\[
I_{2h}^h = \begin{bmatrix}
1 \\
2 \\
1 \\
2 \\
1 \\
2 \\
1
\end{bmatrix}
\]

• \( I_{2h}^h \) has full rank & null space \{0\}.

Spectral properties of interpolation

• How does \( I_{2h}^h \) act on the eigenvectors of \( A^{2h} \) ?

• Consider \( (w_k^{2h})_j = \sin \left( \frac{jk \pi}{(N+1)/2} \right) \), \( 1 \leq k \leq (N-1)/2, \)

\( 0 \leq j \leq (N+1)/2 \).

• A bit of work shows that the modes of \( A^{2h} \) are NOT "preserved" by \( I_{2h}^h \), but that the space \( W_k \) is "preserved":

\[
I_{2h}^h w_k^{2h} = \cos \left( \frac{k \pi}{2(N+1)} \right) w_k^h - \sin \left( \frac{k \pi}{2(N+1)} \right) w_k^h = c_k w_k^h - s_k w_k^h.
\]

Spectral properties of interpolation

• Interpolation of smooth \( \Omega^{2h} \) modes excites oscillatory modes on \( \Omega^h \).

• Note that if \( k \ll \frac{N}{2} \), then

\[
I_{2h}^h w_k^{2h} = \left( 1 - O \left( \frac{k^2}{N^2} \right) \right) w_k^h + O \left( \frac{k^2}{N^2} \right) w_k^h
\]

\( \approx w_k^h \)

• \( I_{2h}^h \) is 2nd-order interpolation.
Range of interpolation

- The range of \( I_{2h}^h \) is the span of the columns of \( I_{2h}^h \).
- Let \( \xi_i \) be the \( i^{th} \) column of \( I_{2h}^h \).

\[
\xi_i^h = \sum_{k=1}^{N} b_k w_k^h, \quad b_k \neq 0
\]

- All the Fourier modes of \( A^h \) are needed to represent \( \text{Range}( I_{2h}^h ) \).

**CG error propagation**

- Subtracting the previous two expressions, we get

\[
e^h \left\{ \begin{array}{l}
I - I_{2h}^h (A^{2h})^{-1} I_{2h}^h A^h \end{array} \right\} Re^h
\]

\[
e^h \left\{ \begin{array}{l}
CG e^h
\end{array} \right\}
\]

- How does CG act on the modes of \( A^h \)? Assume \( e^h \) consists of the modes \( w_k^h \) & \( w_{k'}^h \) for \( 1 \leq k \leq \frac{N-1}{2} \) & \( k' = N + 1 - k \).

- We know how \( R^\alpha, A^h, I_{2h}^h, (A^{2h})^{-1}, I_{2h}^h \) act on \( w_k^h \) & \( w_{k'}^h \).

**Use all the facts to analyze the coarse-grid correction scheme**

1) Relax once on \( \Omega^h \): \( v^h \leftarrow R v^h + B f^h \).

2) Compute & restrict residual \( f^{2h} \leftarrow I_{2h}^h (f^h - A^h v^h) \).

3) Solve residual equation \( \forall^h = (A^h)^{-1} f^{2h} \).

4) Correct fine-grid solution \( v^h \leftarrow v^h + I_{2h}^h v^{2h} \).

- The entire process appears as

\[
\begin{array}{c}
v^h \leftarrow R v^h + B f^h + I_{2h}^h (A^{2h})^{-1} I_{2h}^h (f^h - A^h (R v^h + B f^h))
\end{array}
\]

- The exact solution satisfies

\[
u^h = R u^h + B f^h + I_{2h}^h (A^{2h})^{-1} I_{2h}^h (f^h - A^h (R u^h + B f^h))
\]

**CG error propagation**

- For now, assume no relaxation. Then \( W_k = \text{span} \{ w_k^h, w_{k'}^h \} \) is invariant under CG:

\[
CG w_k^h = s_k w_k^h + s_k w_{k'}^h,
\]

\[
CG w_{k'}^h = c_k w_k^h + c_k w_{k'}^h
\]

where

\[
c_k = \cos^2 \left( \frac{k \pi}{2(N+1)} \right) \quad s_k = \sin^2 \left( \frac{k \pi}{2(N+1)} \right)
\]
**CG error propagation for \( k \ll N \)**

- Consider the case \( k \ll N \) (extremely smooth & oscillatory modes):

\[
w_k \to O\left(\frac{k^2}{N^2}\right) w_k + O\left(\frac{k^2}{N^2}\right) w_k'
\]

\[
w_k' \to \left(1 - O\left(\frac{k^2}{N^2}\right)\right) w_k + \left(1 - O\left(\frac{k^2}{N^2}\right)\right) w_k'
\]

- Hence, CG eliminates the smooth modes but does not damp the oscillatory modes of the error!

**CG with relaxation**

- Next, include one relaxation sweep. Assume that the relaxation \( R \) preserves the modes of \( A^h \) (although this is often unnecessary). Let \( \lambda_k \) denote the eigenvalue of \( R \) associated with \( w_k \). For \( k \ll N/2 \):

\[
w_k \to \lambda_k s_k w_k + \lambda_k s_k w_k' \quad \text{Small!}
\]

\[
w_k' \to \lambda_k' c_k w_k + \lambda_k' c_k w_k' \quad \text{Small!}
\]

**Crucial observation**

- Between relaxation & coarse-grid correction, both smooth & oscillatory components of the error are effectively damped.

- This is the “spectral” picture of how multigrid works. We examine now another viewpoint, the “algebraic” picture of multigrid.

**Recall the variational properties**

- All the analysis that follows assumes that the variational properties hold:

\[
A^{2h} = I_h^{2h} A^h I_{2h}^{2h}
\]

\[
I_{2h}^{2h} = c \left( I_h^{2h} \right)^T
\]

Galerkin Condition

For \( c \) in \( \mathbb{R} \).
Fundamental Theorem of Linear Algebra
\[ N(B) = R(B^T)^\perp \]

- If \( x \in N(B) \), then, for any \( y \in R(B^T) \), we have \( y = B^Tz \) and, hence,
  \[ \langle x, y \rangle = \langle x, B^Tz \rangle = \langle Bx, z \rangle = 0. \]
  So, \( x \in R(B^T)^\perp \) and, thus, \( N(B) \subset R(B^T)^\perp \).

- But if \( x \in R(B^T)^\perp \), then \( x \perp B^Tz \) for any \( z \), so
  \[ 0 = \langle x, B^Tz \rangle = \langle Bx, z \rangle. \]
Picking \( z = Bx \) shows \( x \in N(B) \), so \( N(B) \supset R(B^T)^\perp \).

Subspace decomposition of \( \Omega^h \)
- If \( u^h \in N(I_{2h}^h A^h) \), then, for any \( u^{2h} \), we have
  \[ 0 = \langle I_{2h}^h A^h u^h, u^{2h} \rangle = \langle A^h u^h, I_{2h} A^{2h} \rangle, \]
  so
  \[ R(I_{2h}^h) \perp A^{2h} N(I_{2h}^h A^h), \]
  where \( x \perp A^h y \) means \( \langle A^h x, y \rangle = 0 \). “energy”
- Moreover, any \( e^h \) can be written as \( e^h = s^h + t^h \),
  where \( s^h \in R(I_{2h}^h) \) & \( t^h \in N(I_{2h}^h A^h) \).
- Hence, we get the “energy-orthogonal” decomposition
  \[ \Omega^h = R(I_{2h}^h) \oplus N(I_{2h}^h A^h). \]

Algebraic interpretation of CG
Consider the subspaces that make up \( \Omega^h \) & \( \Omega^{2h} \).

From now on, ‘R( )’ refers to the Range of a linear operator & ‘N( )’ to its Null Space.

\[ \Omega^h \]
\[ R(I_{2h}^h) \]
\[ N(I_{2h}^h) \]
\[ I_{2h}^h \]
\[ I_{2h} \]
\[ N(I_{2h}^h) = R(\text{ } I_{2h}^h)^T \]
\[ R(I_{2h}^h) \]
\[ N(I_{2h}^h) \]

But we really care about \( N(I_{2h}^h A^h) \) !?

Characteristics of the subspaces
- Since \( s^h = I_{2h}^h q^{2h} \) for some \( q^{2h} \in \Omega^{2h} \), we associate \( s^h \) with the smooth components of \( e^h \).
  But, \( s^h \) generally has all Fourier modes in it.
  Recall the basis vectors for \( I_{2h}^h \):
  \[ \mathbf{\ast} \]

- Similarly, we associate \( t^h \) with oscillatory components of \( e^h \), although \( t^h \) generally has all Fourier modes in it as well. Recall that \( N(I_{2h}^h) \) is spanned by \( \eta_i = A^{2h} \hat{e}_i \), so \( N(I_{2h}^h A^h) \) is spanned by the unit vectors \( \hat{e}_i = (0, 0, ..., 0, 1, 0, ..., 0)^T \) for odd \( i \),
  which “look” oscillatory.
**Algebraic analysis of CG**

- Recall that (without relaxation)
  
  \[
  CG = I - I_{2h}^h (A^{2h})^{-1} I_{2h}^h A^h.
  \]

- First note that if \( s^h \in R(I_{2h}^h) \), then \( CG s^h = 0 \).
  This follows since \( s^h = I_{2h}^h q^{2h} \) for some \( q^{2h} \in \Omega^{2h} \)
  & therefore
  \[
  CG s^h = \left[ I - I_{2h}^h (A^{2h})^{-1} I_{2h}^h A^h \right] I_{2h}^h q^{2h} = 0.
  \]
  
  by Galerkin property

- It follows that \( N(CG)=R(I_{2h}^h) \), that is, the null space of \( CG \) is the range of interpolation.

  What does this imply?

**More algebraic analysis of CG**

- Next, note that if \( t^h \in N(I_{2h}^h A^h) \), then
  \[
  CG t^h = \left[ I - I_{2h}^h (A^{2h})^{-1} I_{2h}^h A^h \right] t^h = 0
  \]
  \[
  \Rightarrow CG t^h = t^h
  \]

- Thus, \( CG \) is the identity on \( N(I_{2h}^h A^h) \).

  What does this imply?

**How does the algebraic picture fit with the spectral view?**

- We may view \( \Omega^h \) in two ways:
  \[
  \Omega^h = \left\{ \begin{array}{c}
  \text{Low frequency modes} \\
  1 \leq k \leq N/2
  \end{array} \right\} \oplus \left\{ \begin{array}{c}
  \text{High frequency modes} \\
  N/2 < k < N
  \end{array} \right\}
  \]

  that is,
  \[ \Omega^h = L \oplus H \]

  or
  \[ \Omega^h = R(I_{2h}^h) \oplus N(I_{2h}^h A^h) \].

  Are these "orthogonal" decompositions?

**Actually, each view is just part of the picture**

- The operations we have examined work on different spaces!

- While \( N(I_{2h}^h A^h) \) is mostly oscillatory, it isn't \( H \),
  & while \( R(I_{2h}^h) \) is mostly smooth, it isn't \( L \).

- Relaxation eliminates error from \( H \).

- Coarse-grid correction eliminates error from \( R(I_{2h}^h) \).
How it actually works (cartoon)

Relaxation eliminates $H$, but increases the error in $R(\mathbf{I}_{2h}^h)$

$CG$ eliminates error in $R(\mathbf{I}_{2h}^h)$, but increases error in $H$

Why is this working well?
6. Nonlinear problems

HANG ON !!!

- How should we approach the nonlinear system
  \[ A(u) = f \]
  & can we use MG to solve it?

- A fundamental relation we’ve relied on is the linear residual equation:
  \[ Au - Av = f - Av \Rightarrow Ae = r. \]
- We can’t rely on this now since a nonlinear \( A(u) \) generally means
  \[ A(u) - A(v) \neq A(e). \]

Some reflection

- What do you want out of this course?
- Do you seem to be getting there?
- Can we get there more directly?
- Start thinking about your project!
The nonlinear residual equation

We still base our development around the residual equation, now the nonlinear residual equation:

\[ A(u) = f \]

\[ \Rightarrow A(u) - A(v) = f - A(v) \]

\[ \Rightarrow A(u) - A(v) = r \]

How can we use this equation as the basis for a solution method?

Newton's method for scalar \( F: \mathbb{R} \rightarrow \mathbb{R} \)

- Best known & most important nonlinear problem solver!
- We wish to solve \( F(x) = 0 \).
  
  Ex:
  
  \[ F(x) = x e^x - 1, \quad F'(x) = (1 + x) e^x. \]

- Expand \( F \) in a Taylor series about \( x \):
  
  \[ F(x + s) = F(x) + s F'(x) + s^2 F''(\xi). \]

  Ex:
  
  \[ (x + s)e^{(x+s)} - 1 = x e^x - 1 + s (1 + x) e^x + \text{h.o.t.} \]

- Dropping higher-order terms (h.o.t.), if \( x + s \) is a solution,
  
  \[ 0 = F(x) + s F'(x) \Rightarrow s = -F(x)/F'(x). \]

- We thus arrive at Newton's method:
  
  \[ x \leftarrow x - F(x)/F'(x) \]

Newton's method for systems

- Ex: \( -u''(x) + u(x) e^{u(x)} = f \) may be discretized as

  \[
  A(u) = \begin{pmatrix}
  \frac{2 u_1 - u_2}{h^2} + u_1 e^{u_1} \\
  \vdots \\
  \frac{-u_{N-1} + 2 u_N - u_1}{h^2} + u_N e^{u_N}
  \end{pmatrix}
  \begin{pmatrix}
  f(x_1) \\
  \vdots \\
  f(x_N)
  \end{pmatrix}
  \]

- Taylor series about \( v \):
  
  \[ A(v + e) = A(v) + J(v)e + \text{h.o.t.} \]

- Expanding \( A(v + e) \) in a Taylor series about \( v \):

  \[ A(v + e) = A(v) + J(v)e + \text{h.o.t.} \]

  \[
  J(v) = \begin{pmatrix}
  \frac{\partial a_1}{\partial u_1}(v) & \cdots & \frac{\partial a_1}{\partial u_N}(v) \\
  \vdots & \ddots & \vdots \\
  \frac{\partial a_N}{\partial u_1}(v) & \cdots & \frac{\partial a_N}{\partial u_N}(v)
  \end{pmatrix}
  \]

  \[ J(v) = \begin{pmatrix}
  \frac{\partial a_1}{\partial u_1}(v) \\
  \vdots \\
  \frac{\partial a_N}{\partial u_1}(v)
  \end{pmatrix}
  \]

Newton's method for general systems

- The system \( A(u) = f \) in vector form is

  \[
  \begin{pmatrix}
  a_1(u_1, u_2, \ldots, u_N) \\
  a_2(u_1, u_2, \ldots, u_N) \\
  \vdots \\
  a_N(u_1, u_2, \ldots, u_N)
  \end{pmatrix} = \begin{pmatrix}
  f_1 \\
  f_2 \\
  \vdots \\
  f_N
  \end{pmatrix}
  \]

- Expanding \( A(v + e) \) in a Taylor series about \( v \):

  \[ A(v + e) = A(v) + J(v)e + \text{h.o.t.} \]

  \[
  J(v) = \begin{pmatrix}
  \frac{\partial a_1}{\partial u_1}(v) & \cdots & \frac{\partial a_1}{\partial u_N}(v) \\
  \vdots & \ddots & \vdots \\
  \frac{\partial a_N}{\partial u_1}(v) & \cdots & \frac{\partial a_N}{\partial u_N}(v)
  \end{pmatrix}
  \]

  \[ J(v) = \begin{pmatrix}
  \frac{\partial a_1}{\partial u_1}(v) \\
  \vdots \\
  \frac{\partial a_N}{\partial u_1}(v)
  \end{pmatrix}
  \]
Newton for systems (cont’d)

- $\mathcal{J}(v)$ is the Jacobian

\[
\mathcal{J}(v) = \begin{pmatrix}
\frac{\partial a_1}{\partial u_1} & \frac{\partial a_1}{\partial u_2} & \cdots & \frac{\partial a_1}{\partial u_N} \\
\frac{\partial a_2}{\partial u_1} & \frac{\partial a_2}{\partial u_2} & \cdots & \frac{\partial a_2}{\partial u_N} \\
\vdots & \vdots & \ddots & \vdots \\
\frac{\partial a_N}{\partial u_1} & \frac{\partial a_N}{\partial u_2} & \cdots & \frac{\partial a_N}{\partial u_N}
\end{pmatrix}_{|v}
\]

- If $u = v + e$ is a solution, $f = A(v) + \mathcal{J}(v)e + \text{h.o.t.}$, so $e \approx [\mathcal{J}(v)]^{-1}(f - A(v))$.

- This leads to the iteration

$\begin{align*}
v &\leftarrow v + [\mathcal{J}(v)]^{-1}(f - A(v))
\end{align*}$

Newton’s via the residual equation

- The nonlinear residual equation is

$A(v + e) - A(v) = r$.

- Expanding $A(v + e)$ in a two-term Taylor series about $v$ & ignoring h.o.t.:

$A(v) + \mathcal{J}(v)e - A(v) = r, \quad \hat{e} \approx e$

or

$\mathcal{J}(v)\hat{e} = r$.

- Newton’s method is thus:

$\begin{align*}
v &\leftarrow v + [\mathcal{J}(v)]^{-1}r, \quad r = f - A(v)
\end{align*}$

How does multigrid fit in?

- One obvious method is to use multigrid to solve $\mathcal{J}(v)\hat{e} = r$ at each iteration step. This method is called Newton-MG & can be very effective.

- However, we might want to use multigrid ideas to treat the nonlinearity directly.

- To do that, we need to specialize multigrid components (relaxation & coarsening) for the nonlinear case.

What is nonlinear relaxation?

- Several of the common relaxation schemes have nonlinear counterparts. For $A(u) = f$, we describe nonlinear Gauss-Seidel:

For each $i = 1, 2, \ldots, N$:

- Change the value of $v$, so that the $i$th equation is satisfied, that is, so that $(A(v))_i = f_i$.

- Equivalently:

For each $i = 1, 2, \ldots, N$:

- Find $s \in \mathbb{R}$ such that $(A(v + s \varepsilon_i))_i = f_i,$ where $\varepsilon_i$ is the $i$th canonical unit basis vector.
How is nonlinear Gauss-Seidel done?

• Each \((A(v)_i) = f_i\) is a nonlinear scalar equation for \(v_i\).
  We can use the scalar Newton's method to solve.

• Example: \(-u''(x) + u(x) e^{u(x)} = f\) may be discretized
  so that \((A(v)_i) = f_i\) is given by
  
  \[
  -v_{i-1} - 2v_i + v_{i+1} = f_i + v_i e^{v_i} + f_i
  \]

  \(1 \leq i \leq N\)

• Newton iteration for \(v_i\) is given by
  
  \[
  v_i \leftarrow v_i - \frac{v_{i-1} + 2v_i - v_{i+1}}{h^2} + \frac{1}{h^2} + (1 + v_i) e^{v_i}
  \]

• Consider the coarse-grid equation:
  
  \[
  A^{2h}(v^{2h} + e^{2h}) = A^{2h}(v^{2h}) = r^{2h}.
  \]

• Given \(v^h\), a fine-grid approximation, we restrict
  the residual to the coarse grid:
  
  \[
  r^{2h} = I^{2h}_h(f^h - A^h(v^h)).
  \]

• For \(v^{2h}\), we restrict \(v^h\) by \(v^{2h} = I^{2h}_h v^h\).

Thus,

\[
A^{2h}(I^{2h}_h v^h + e^{2h}) = A^{2h}(I^{2h}_h v^h) + I^{2h}_h (f^h - A^h(v^h))
\]

How do we do coarsening for nonlinear multigrid?

• Recall the nonlinear residual equation
  
  \[
  A(v + e) - A(v) = r.
  \]

• In multigrid, we obtain an approximate
  solution \(v^h\) on the fine grid, then solve the
  residual equation on the coarse grid.

• The residual equation on \(\Omega^{2h}\) appears as
  
  \[
  A^{2h}(v^{2h} + e^{2h}) - A^{2h}(v^{2h}) = r^{2h}.
  \]

We've obtained a coarse-grid

equation of the form \(A^{2h}(u^{2h}) = f^{2h}\)

• Consider the coarse-grid equation:
  
  \[
  A^{2h}(I^{2h}_h v^h + e^{2h}) = A^{2h}(I^{2h}_h v^h) + I^{2h}_h (f^h - A^h(v^h))
  \]

  \(u^{2h}\)

  \(f^{2h}\)

  coarse-grid unknown

  all quantities are known

• We solve \(A^{2h}(u^{2h}) = f^{2h}\) for \(u^{2h} = I^{2h}_h v^h + e^{2h}\) &
  obtain

  \[
  e^{2h} = u^{2h} - I^{2h}_h v^h
  \]

• We then apply the correction:

  \[
  v^h \leftarrow v^h + I^{2h}_h e^{2h}
  \]
Full approximation scheme (FAS)

2-grid form

• Perform nonlinear relaxation on \( A^h(u^h) = f^h \) to obtain an approximation \( v^h \).

• Restrict the approximation & its residual:
  \[ v^{2h} = I_h^{2h} v^h \quad r^{2h} = I_h^{2h} (f^h - A(v^h)) \]

• Solve the coarse-grid equation:
  \[ A^{2h}(u^{2h}) = A^{2h}(v^{2h}) + r^{2h} \]

• Extract \( 2h \) approximation to \( h \) error:
  \[ e^{2h} = u^{2h} - v^{2h} \]

• Interpolate & correct:
  \[ v^h \leftarrow v^h + I_{2h}^h e^{2h} \]

A few observations about FAS

- If \( A \) is a linear operator, then FAS reduces directly to the linear two-grid correction scheme:
  \[ A^{2h}(I_h^{2h} v^h + e^{2h}) = A^{2h}(I_h^{2h} v^h) + I_h^{2h} (f^h - A(v^h)) \]

- An exact solution to the fine-grid problem is a fixed point of the FAS iteration:
  \[ A^{2h}(I_h^{2h} v^h + e^{2h}) = A^{2h}(I_h^{2h} v^h) + I_h^{2h} (f^h - A(v^h)) \]

A few more observations about FAS

- The FAS coarse-grid equation can be written as
  \[ A^{2h}(u^{2h}) = f^{2h} + \tau^{2h}_h \]
  where \( \tau^{2h}_h = A^{2h}(I_h^{2h} v^h) - I_h^{2h} A^h(v^h) \)

- In general, since \( \tau^{2h}_h \neq 0 \), the solution \( u^{2h} \) to the FAS coarse-grid equation is not the same as the solution to the original coarse-grid problem:
  \[ A^{2h}(u^{2h}) = f^{2h} \]

- The tau correction is as a way to alter the coarse-grid equation to enhance its approximation properties.

Still more observations about FAS

- A true multilevel FAS process is recursive, using FAS to solve the nonlinear \( \Omega^{2h} \) problem using \( \Omega^{4h} \).

- Hence, FAS is generally employed in a V- or W-cycling scheme.
Even more observations about FAS

- For linear problems, we use FMG to obtain a good initial guess on the fine grid. Convergence of nonlinear iterations depends critically on having a good initial guess.
- When FMG is used for nonlinear problems, the interpolant $I_{2h}^h u^{2h}$ is generally accurate enough to be in the basin of attraction of the fine-grid solver.
- Thus, whether FAS, Newton, or Newton-multigrid is used on each level, one FMG cycle should provide a solution accurate to the level of discretization, unless the nonlinearity is extremely strong.

Intergrid transfers for FAS

- Generally speaking, the standard operators (linear interpolation, full weighting) work effectively in FAS schemes.
- For strongly nonlinear problems or for the coarse-grid approximation that is to become a fine-grid initial guess, higher-order interpolation (e.g., cubic interpolation) may be beneficial.

What is $A^{2h}(u^{2h})$ in FAS?

As in the linear case, there are two basic possibilities:

1. $A^{2h}(u^{2h})$ is determined by discretizing the nonlinear operator, $A(u)$, in the same fashion as was employed to obtain $A^h(u^h)$, except that the coarser mesh spacing is used.
2. $A^{2h}(u^{2h})$ is determined from the Galerkin condition

$$A^{2h}(u^{2h}) = I_{2h}^h A^h (I_{2h}^h u^{2h})$$

where the action of the Galerkin product can be captured in an implementable formula.

The first method is usually easier & more common.

Nonlinear problems: An example

- Consider

$$-\Delta u(x,y) + \gamma u(x,y) e^{u(x,y)} = f(x,y)$$

on the unit square, $[0,1] \times [0,1]$, with homogeneous Dirichlet boundary conditions & a regular $h = 1/128$ Cartesian grid.

- Suppose the exact solution is

$$u(x,y) = (x^2 - x^3) \sin(3\pi y).$$
Discretization of the nonlinear example

- The operator can be written (sloppily) as
  \[
  \frac{1}{h^2} \begin{pmatrix}
  -1 & -1 \\
  4 & -1
  \end{pmatrix}
  u^h_{i,j} + \gamma u^h_{i,j} e^{u^h_{i,j}} = f_{i,j}
  \]

- Relaxation (nonlinear Gauss-Seidel) is given by
  \[
  v^h_{i,j} \leftarrow v^h_{i,j} - \frac{(A^h(v^h))_{i,j} - f_{i,j}}{\frac{4}{h^2} + \gamma (1 + v^h_{i,j}) e^{v^h_{i,j}}}
  \]

FAS & Newton's method on

\[-\Delta u(x,y) + \gamma u(x,y) e^{u(x,y)} = f(x,y)\]

\[N = 127\]

- FAS (V(2,1))-cycles until \(||r|| < 10^{-10}\).

<table>
<thead>
<tr>
<th>γ</th>
<th>1</th>
<th>10</th>
<th>100</th>
<th>1000</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.135</td>
<td>0.124</td>
<td>0.098</td>
<td>0.072</td>
</tr>
<tr>
<td>100</td>
<td>11</td>
<td>11</td>
<td>10</td>
<td></td>
</tr>
</tbody>
</table>

- Newton's Method with exact inner solves until \(||r|| < 10^{-10}\).

<table>
<thead>
<tr>
<th>γ</th>
<th>1</th>
<th>10</th>
<th>100</th>
<th>1000</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4.00E-05</td>
<td>7.00E-05</td>
<td>3.00E-04</td>
<td>2.00E-04</td>
</tr>
<tr>
<td>100</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td></td>
</tr>
</tbody>
</table>

Newton, Newton-MG, & FAS on

\[-\Delta u(x,y) + \gamma u(x,y) e^{u(x,y)} = f(x,y)\]

\[N = 127, \ \gamma = 10\]

- Newton uses exact solves, Newton-MG is inexact Newton with a fixed number of inner V(2,1)-cycles for the Jacobian problem, overall stopping criterion \(||r|| < 10^{-10}\).

<table>
<thead>
<tr>
<th>Method</th>
<th>Outer iterations</th>
<th>Inner iterations</th>
<th>Megaflops</th>
</tr>
</thead>
<tbody>
<tr>
<td>Newton</td>
<td>3</td>
<td></td>
<td>1660.6</td>
</tr>
<tr>
<td>Newton-MG</td>
<td>3</td>
<td>20</td>
<td>56.4</td>
</tr>
<tr>
<td>Newton-MG</td>
<td>4</td>
<td>10</td>
<td>38.5</td>
</tr>
<tr>
<td>Newton-MG</td>
<td>5</td>
<td>5</td>
<td>25.1</td>
</tr>
<tr>
<td>Newton-MG</td>
<td>10</td>
<td>2</td>
<td>22.3</td>
</tr>
<tr>
<td>Newton-MG</td>
<td>19</td>
<td>1</td>
<td>24.6</td>
</tr>
<tr>
<td>FAS</td>
<td>11</td>
<td></td>
<td>27.1</td>
</tr>
</tbody>
</table>

Compare FMG-FAS & FMG-Newton-MG

\[-\Delta u(x,y) + \gamma u(x,y) e^{u(x,y)} = f(x,y)\]

- We will do one FMG cycle using a single FAS V(2,1)-cycle as the "solver" at each new level. We then follow that with as many FAS V(2,1)-cycles as is needed to obtain \(||r|| < 10^{-10}\).

- Next, we will do one FMG cycle using a Newton-MG step at each new level (with one linear V(2,1)-cycle as the Jacobian "solver." We then follow that with as many Newton-multigrid steps as is needed to obtain \(||r|| < 10^{-10}\).
Compare FMG-FAS & FMG-Newton-MG

\[- \Delta u(x,y) + \gamma u(x,y) \ e^u(x,y) = f(x,y)\]

\[N = 127, \ \gamma = 10\]

<table>
<thead>
<tr>
<th>Cycle</th>
<th>(t^+1)</th>
<th>(t^+)</th>
<th>Mflops</th>
<th>(t^+1)</th>
<th>(t^)</th>
<th>Mflops</th>
<th>Cycle</th>
</tr>
</thead>
<tbody>
<tr>
<td>FMG-FAS</td>
<td>1.10E-02</td>
<td>2.00E-05</td>
<td>3.1</td>
<td>1.06E-02</td>
<td>2.50E-05</td>
<td>2.4</td>
<td>FMG-Newton</td>
</tr>
<tr>
<td>FAS V</td>
<td>6.80E-04</td>
<td>2.40E-05</td>
<td>5.4</td>
<td>6.70E-04</td>
<td>2.40E-05</td>
<td>4.1</td>
<td>Newton-MG</td>
</tr>
<tr>
<td>FAS V</td>
<td>5.00E-05</td>
<td>2.49E-05</td>
<td>7.6</td>
<td>5.10E-05</td>
<td>2.40E-05</td>
<td>5.8</td>
<td>Newton-MG</td>
</tr>
<tr>
<td>FAS V</td>
<td>3.90E-06</td>
<td>2.49E-05</td>
<td>9.9</td>
<td>6.30E-06</td>
<td>2.40E-05</td>
<td>7.5</td>
<td>Newton-MG</td>
</tr>
<tr>
<td>FAS V</td>
<td>3.20E-07</td>
<td>2.49E-05</td>
<td>12.2</td>
<td>1.70E-06</td>
<td>2.40E-05</td>
<td>9.2</td>
<td>Newton-MG</td>
</tr>
<tr>
<td>FAS V</td>
<td>3.00E-08</td>
<td>2.49E-05</td>
<td>14.4</td>
<td>5.30E-07</td>
<td>2.40E-05</td>
<td>10.9</td>
<td>Newton-MG</td>
</tr>
<tr>
<td>FAS V</td>
<td>2.90E-09</td>
<td>2.49E-05</td>
<td>18.7</td>
<td>1.70E-07</td>
<td>2.40E-05</td>
<td>12.6</td>
<td>Newton-MG</td>
</tr>
<tr>
<td>FAS V</td>
<td>3.00E-10</td>
<td>2.49E-05</td>
<td>18.9</td>
<td>5.40E-08</td>
<td>2.40E-05</td>
<td>14.3</td>
<td>Newton-MG</td>
</tr>
<tr>
<td>FAS V</td>
<td>3.20E-11</td>
<td>2.49E-05</td>
<td>21.2</td>
<td>1.70E-08</td>
<td>2.40E-05</td>
<td>16.0</td>
<td>Newton-MG</td>
</tr>
</tbody>
</table>

Motivating FAS for nonlinear \(A\)

\[A(v + e) = f \rightarrow A^{2h}(v^{2h} + e^{2h}) = "f^{2h}"\]

\[A^{2h}(I^2_h v^h + e^{2h}) = A^{2h}(I^2_h v^h) + I^2_h (f^{2h} - A^{2h}(v^h))\]

- What overriding principle can we find to get from \(h\) to \(2h\)?
- With known \(v\), how do we discretize a PDE of the form \(A(v + e) = f\)?
- Example:
  \[\gamma uu' - u'' = f \rightarrow \gamma (v + e)(v + e)' - (v + e)'' = f\]
- Expand to get an equation in \(e\) of form \(g + ae + be' + \ldots\):
  \[\gamma vv' - v'' + \gamma (ve + ve' + ee') - e'' = f\]

Remembering coarse-grid correction

\[Au = f\]

- Relax (damped Jacobi) to smooth \(e = u - v\):
  \[v \leftarrow v - \omega D^{-1}(Av - f)\]
- Form the residual equation \(Ae = r\):
  \[Ae = A(u - v) = f - Av = r\]
- Use premise that smooth error \(\Rightarrow e = I_{2h}^he^{2h}\):
  \[A I_{2h}^he^{2h} = r\]

Differential residual equation

\[\gamma (v'e + ve' + ee') - e'' = f - A(v) = r(v)\]

- Right side (analogous to injection):
  \[r(v) \rightarrow r_j^h = r(x_j)\]
- Coefficients:
  \[v \rightarrow v_j^h = v(x_j), \quad v' \rightarrow (v')_j^h = (v(x_{j+1}) - v(x_{j-1}))(2h)\]
- Unknowns:
  \[e \rightarrow e_j^h, \quad e' \rightarrow (e')_j^h = (e_{j+1}^h - e_{j-1}^h)/(2h), \quad e'' \rightarrow (e'')_j^h = (e_{j+1}^h - 2e_j^h + e_{j-1}^h)/h^2\]
Leading to FAS...

\[ \gamma u' - u'' = f \rightarrow A(v+e) = f \rightarrow \gamma (v' + ve + ee') - e'' = f - A(v) \]

- Fine-grid residual equation (at \( h \) point \( 2j \)):

\[
\frac{\gamma v_{j-1}^{h} - v_{j+1}^{h}}{2h} e_{j}^{h} + \frac{\gamma v_{j+1}^{h} e_{j}^{h}}{2h} + \frac{\gamma v_{j+1}^{h} - v_{j}^{h}}{2h} e_{j+1}^{h} + \frac{\gamma v_{j}^{h} e_{j+1}^{h}}{2h} + \frac{-h}{h^2} + 2e_{j}^{h} - e_{j+1}^{h} = r_{j}^{h}
\]

- Coarse-grid residual equation (at \( 2h \) point \( j \)):

\[
\frac{\gamma v_{2j-1}^{h} - v_{2j+1}^{h}}{4h} e_{j}^{h} + \gamma e_{j}^{h} e_{j}^{h} - \frac{\gamma v_{2j+1}^{h} - v_{2j}^{h}}{4h} e_{j+1}^{h} + \frac{\gamma v_{2j}^{h} e_{j+1}^{h}}{4h} + \frac{-h^2}{h^2} + 2e_{j}^{h} - e_{j+1}^{h} = (l_{2j}^{h})
\]

- FAS

\[
A^{2h} (f_{h}^{2h} v_{h} + e^{2h}) - A^{2h} (f_{h}^{2h} v_{h}) = f_{h}^{2h} (f - A^{2h} (v^{h}))
\]

One Newton-MG step

- **Step 0.** Given \( v \), form the grid \( h \) linear correction equation:

\[
-\frac{e_{j}^{h} + 2e_{j+1}^{h} - e_{j+2}^{h}}{h^2} + (1 + v_{j}^{h}) e_{j}^{h} = r_{j} = f_{j} - \left( -\frac{v_{j+1}^{h} + 2v_{j}^{h} - v_{j-1}^{h}}{h^2} + v_{j}^{h} e_{j}^{h} \right)
\]

Initialize the Newton correction approximation: \( e = 0 \).

- **Step 1:** Relax on the grid \( h \) linear equation.

- **Step 2:** Solve the grid \( 2h \) error correction equation:

\[
\frac{-e_{j}^{2h} + 2e_{j+1}^{2h} - e_{j+2}^{2h}}{(2h)^2} + (1 + v_{j}^{2h}) e_{j}^{2h} = r_{2j} = \left( -\frac{e_{j}^{2h} + 2e_{j+1}^{2h} - e_{j+2}^{2h}}{(2h)^2} + (1 + v_{j}^{2h}) e_{j}^{2h} \right)
\]

- **Step 3:** Correct the grid \( h \) Newton correction:

\[
e \leftarrow e + l_{2h}^{h} e^{2h}
\]

- **Step 4:** Stop if you’ve "solved" the linear equation well enough for Newton correction \( e \) & set \( v \leftarrow v + e \). Else, leave \( v \) alone & return to Step 1.

One FAS step

- **Step 0.** Given \( v \), form the grid \( h \) nonlinear equation:

\[
-\frac{v_{j-1}^{h} + 2v_{j}^{h} - v_{j+1}^{h}}{h^2} + v_{j}^{h} e_{j}^{h} = f_{j}
\]

- **Step 1:** Relax on the grid \( h \) nonlinear equation to improve \( v \).

- **Step 2:** Solve the grid \( 2h \) FAS correction equation:

\[
\frac{-e_{j}^{2h} + 2e_{j+1}^{2h} - e_{j+2}^{2h}}{(2h)^2} + (v_{j}^{2h} + e_{j}^{2h}) e_{j}^{2h} = \left( l_{2h}^{h} \right)
\]

- **Step 3:** Correct the grid \( h \) approximation \( v \):

\[
v \leftarrow v + l_{2h}^{h} e^{2h}
\]

Example: Newton-MG vs. FAS

- **PDE** (er, ODE):

\[ -u''(x) + u(x) e^{u(x)} = f(x). \]

- **Discretization**:

\[
\frac{-v_{j}^{h} + 2v_{j}^{h} - v_{j+1}^{h}}{h^2} + v_{j}^{h} e_{j}^{h} = f_{j}.
\]
7. Selected applications

7a. Neumann boundary conditions

Consider the 1-D problem

\[ u''(x) = f(x), \ 0 < x < 1, \]
\[ u'(0) = u'(1) = 0. \]

We discretize on the interval [0,1] with \( h = 1/(N+1) \)
grid spacing & nodes \( x_j = jh, \ j = 0,1,2, ..., N+1. \)

We extend the interval with two ghost points:

-1 0 1 j-1 j j+1 N N+1 N+2

Eliminating the ghost points

\[ \frac{u_{1} - u_{-1}}{2h} = 0 \quad \Rightarrow \quad u_{-1} = u_{1} \quad \frac{u_{N+2} - u_{N}}{2h} = 0 \quad \Rightarrow \quad u_{N+2} = u_{N} \]

We use the boundary conditions to eliminate \( u_{-1} \) and \( u_{N+2} \)

Eliminating the ghost points in the \( j = 0 \) & \( j = N+1 \)
equations gives the \((N+2)\times(N+2)\) system of equations:

\[ -u_{j-1} + 2u_{j} - u_{j+1} \]
\[ \frac{h^2}{u_{j}} = f_{j} \quad 1 \leq j \leq N \]

\[ \frac{u_{1} - u_{-1}}{2h} = 0 \quad u_{N+2} - u_{N} \frac{2h}{2h} = 0 \]

\[ \frac{2u_{0} - 2u_{1}}{h^2} = f_{0} \quad \frac{-2u_{N} + 2u_{N+1}}{h^2} = f_{N+1} \]
Write the system in matrix form

- We can write $A^h u^h = f^h$, where
  
  $$A^h = \frac{1}{h^2} \begin{pmatrix} 2 & -2 & 0 & \cdots & 0 & -2 \\ -1 & 2 & -1 & \cdots & -1 \\ 0 & -1 & 2 & -1 & \cdots \\ 0 & 0 & \ddots & \ddots & \ddots \\ 0 & 0 & \cdots & 0 & -1 \\ 0 & 0 & \cdots & 0 & -2 \end{pmatrix}$$

- Note that $A^h$ is $(N+2) \times (N+2)$ & nonsymmetric, & the system involves unknowns $u_0^h$ & $u_{N+1}^h$ at the boundaries.

The well-posed system

- The compatibility condition is necessary for a solution to exist. In general, it is also sufficient:
  
  $$-\frac{\partial^2}{\partial x^2}$$
  
  is a well-behaved operator on the space of functions $u(x)$ that have zero mean.

- Thus, we may conclude that if $f(x)$ satisfies the compatibility condition, then the problem is well-posed:
  
  $$-u''(x) = f(x), \quad 0 < x < 1,$n
  $$u'(0) = u'(1) = 0,$n
  $$\int_0^1 u(x) dx = 0.$n

- The last says: of all possible solutions $u(x) + \text{constant}$, we choose the one with zero mean.

We must consider compatibility

- The problem - $u''(x) = f(x)$, for $0 < x < 1$, with
  
  $$u'(0) = u'(1) = 0,$$n

  is not well-posed!

- If $u(x)$ is a solution, then so is $u(x) + \text{constant}$.

- We cannot be certain a solution exists. If one does, it must satisfy
  
  $$\int_0^1 u''(x) dx = \int_0^1 f(x) dx \quad \Leftrightarrow \quad \int_0^1 [u''(x) - u'(0)] dx = \int_0^1 f(x) dx$$

- This integral compatibility condition is necessary!
  
  If $f(x)$ doesn't satisfy it, there is no solution!

The discrete problem is not well posed

- Since all row sums of $A^h$ are zero, then
  
  $$1^h \in N(A^h).$$

- It's easy to see that $\dim(N(A^h)) = 1$, so $N(A^h) = \text{span}\{1^h\}$.

- By the Fundamental Theorem of Linear Algebra,
  
  $A^h u^h = f^h$ has a solution if & only if
  
  $$f^h \in \text{N}(A^h)^\perp.$n

- It is easy to show that $\text{N}(A^h)^\perp = c(1/2, 1, 1, ..., 1, 1/2)^T$.

- Thus, $A^h u^h = f^h$ has a solution if & only if
  
  $$f^h \perp c(1/2, 1, 1, ..., 1, 1/2)^T.$$n

- That is,
  
  $$\frac{1}{2} f^h_0 + \sum_{j=1}^N f^h_j + \frac{1}{2} f^h_{N+1} = 0.$$
We have two issues to consider

\[ A^h u^h = f^h \]

• Solvability: A solution exists iff \( f^h \in \mathcal{N}(A^h)^\perp \).

• Uniqueness: If \( u^h \) is a solution, then so is \( u^h + v^h \) for any \( v^h \in \mathcal{N}(A^h) \).

• Note that if \( A^h = (A^h)^\top \), then \( \mathcal{N}(A^h)^\perp = \mathcal{N}(A^h) \) & solvability & uniqueness can be handled together.

This is easily done. Multiply the first & last equations by \( 1/2 \), giving

\[
\begin{pmatrix}
1 & -1 \\
-1 & 2 & -1 \\
& & \ddots & \ddots \ \\
& & & 1 & -1 \\
& & & & & 2 & -1 \\
& & & & & & & 1 & 1
\end{pmatrix}
\frac{1}{h^2}
\begin{pmatrix}
u_0^h \\
u_1^h \\
\vdots \\
u_N^h \\
u_{N+1}^h
\end{pmatrix}
= \begin{pmatrix}
\frac{1}{h^2} u_0 \\
\frac{1}{h^2} (u_0 - 2u_j + u_{j+1}) \\
\vdots \\
\frac{1}{h^2} (u_N - 2u_j + u_{j+1}) \\
\frac{1}{h^2} u_{N+1}
\end{pmatrix}
\]

The new system is symmetric

• We have the symmetric system

\[
\hat{A}^h u^h = \hat{f}^h
\]

\[
\begin{pmatrix}
1 & -1 \\
-1 & 2 & -1 \\
& & \ddots & \ddots \ \\
& & & 1 & -1 \\
& & & & & 2 & -1 \\
& & & & & & & 1 & 1
\end{pmatrix}
\frac{1}{h^2}
\begin{pmatrix}
u_0^h \\
u_1^h \\
\vdots \\
u_N^h \\
u_{N+1}^h
\end{pmatrix}
= \begin{pmatrix}
\frac{1}{h^2} f_0 \\
\frac{1}{h^2} (f_0 - 2f_j + f_{j+1}) \\
\vdots \\
\frac{1}{h^2} (f_N - 2f_j + f_{j+1}) \\
\frac{1}{h^2} f_{N+1}
\end{pmatrix}
\]

• Solvability is guaranteed by ensuring that \( \hat{f}^h \) is orthogonal to the constant vector \( 1^h \):

\[
\langle \hat{f}^h, 1^h \rangle = \sum_{j=0}^{N+1} \hat{f}^h_j = 0
\]

One-sided differences at boundary

• No ghost points:

\[
\begin{array}{cccccccccccccccc}
0 & 1 & j-1 & j & j+1 & N & N+1 \\
\hline
u'(0) &=& \frac{u_1 - u_0}{h} \\
u'(j) &=& \frac{-u_{j-1} + 2u_j - u_{j+1}}{h^2} \\
u'(1) &=& \frac{u_N - u_{N+1} - u_0}{h}
\end{array}
\]

• This yields the system

\[
\begin{pmatrix}
-\frac{u_{j-1}}{h^2} + \frac{2u_j}{h^2} - \frac{u_{j+1}}{h^2} \\
\frac{u_1 - u_0}{h^2} - \frac{u_{N+1} - u_N}{h^2}
\end{pmatrix}
= \begin{pmatrix}
f_j \\
0
\end{pmatrix}
\quad 0 \leq j \leq N
\]

or, more simply

\[
\langle \hat{A}^h u^h, 1^h \rangle = \hat{f}^h_0 = \hat{f}_{N+1} = 0
\]

The well-posed discrete system

• The \((N+3) \times (N+2)\) system is:

\[
\begin{pmatrix}
-u_{j-1} + 2u_j - u_{j+1} \\
\frac{u_0 - u_1}{h^2}
\end{pmatrix}
= \begin{pmatrix}
f_j \\
0
\end{pmatrix}
\quad 1 \leq j \leq N
\]

for one-sided scheme

\[
\begin{pmatrix}
-\frac{u_N + u_{N+1}}{h^2} + \frac{u_{N+1}}{h^2} \\
\frac{N+1}{h^2} \sum_{i=0}^{N+1} u_i^h - \frac{u_0}{h^2}
\end{pmatrix}
= \begin{pmatrix}
f_N \\
0
\end{pmatrix}
\]

(choose the zero mean solution)
Multigrid for the Neumann problem

- We must have the interval endpoints on all grids

\[
x_0 \quad x_1^h \quad x_{(N+1)/2}^h \quad x_{N+1}^h\\
x_0^h \quad x_{(N+1)/4}^h \quad x_{(N+1)/2}^h
\]

- Relaxation is performed at all points, including endpoints:

\[
v_h^{(i+1)} = v_h^{(i)} + h^2 f_h^{(i)}
\]

\[
v_{j+1/2}^{(i+1)} = \frac{v_{j+1/2}^{(i)} + v_{j+1}^{(i)} + h^2 f_{j+1/2}^{(i)}}{2}
\]

- We add a global Gram-Schmidt step after relaxation on each level to enforce the zero-mean condition:

\[
v_h \leftarrow v_h - \left( \frac{v_h, v^h}{v^h, v^h} \right) v^h
\]

Restriction also treats the endpoints

For restriction, we use \( I_{2h}^2 = \frac{1}{2} (I_{2h}^h)^T \), yielding the values

\[
\begin{align*}
\hat{f}_0^{2h} &= \frac{1}{2} f_0^h + \frac{1}{4} f_1^h \\
\hat{f}_j^{2h} &= \frac{1}{4} f_{2j-1}^h + \frac{1}{2} f_{2j}^h + \frac{1}{4} f_{2j+1}^h \\
\hat{f}_{N+1}^{2h} &= \frac{1}{4} f_N^h + \frac{1}{2} f_{N+1}^h
\end{align*}
\]

Interpolation must include the endpoints

- We use linear interpolation:

\[
I_{2h}^h = \begin{pmatrix}
1 & 1/2 \\
1/2 & 1/2
\end{pmatrix}
\]

The coarse-grid operator

- We compute the coarse-grid operator using the Galerkin condition

\[
\hat{A}_{2h}^h = I_{2h}^h \hat{A} I_{2h}^h
\]

\[
\begin{pmatrix}
1 & 0 \\
1 & 2 & 0
\end{pmatrix}
\]

\[
\begin{pmatrix}
1/2h^2 & 1/4h^2 & -1/4h^2 \\
1/4h^2 & -1/2h^2 & 1/2h^2 \\
1/2h^2 & -1/2h^2 & 1/4h^2
\end{pmatrix}
\]
Coarse-grid solvability

• Assuming \( f^h \) satisfies \( \langle f^h, v^h \rangle = 0 \), the solvability condition, we can show that theoretically the coarse-grid problem \( A^{2h} u^{2h} = I^{2h} (f^h - A^{h} v^h) \) is also solvable.

• To be certain numerical round-off does not perturb solvability, we incorporate a Gram-Schmidt-like step each time a new right-hand side \( f^h \) is generated for the coarse grid:

\[
 f^{2h} \leftarrow f^{2h} - \frac{\langle f^{2h}, v^{2h} \rangle}{\langle v^{2h}, v^{2h} \rangle} v^{2h}.
\]

Neumann problem: An example

Consider the problem

\[-u''(x) = 2x - 1, \quad 0 < x < 1, \quad u'(0) = u'(1) = 0,\]

which has \( u(x) = x^2 - \frac{x^3}{3} + c \) as a solution for any \( c \) (\( c = -1/12 \) gives the zero mean solution).

<table>
<thead>
<tr>
<th>grid size</th>
<th>( |e^h| )</th>
<th>average conv. factor</th>
<th>( |e^h| )</th>
<th>number of cycles</th>
</tr>
</thead>
<tbody>
<tr>
<td>31</td>
<td>6.30E-11</td>
<td>0.079</td>
<td>9.70E-05</td>
<td>9</td>
</tr>
<tr>
<td>63</td>
<td>1.90E-11</td>
<td>0.089</td>
<td>2.40E-05</td>
<td>10</td>
</tr>
<tr>
<td>127</td>
<td>2.60E-11</td>
<td>0.093</td>
<td>5.90E-06</td>
<td>10</td>
</tr>
<tr>
<td>255</td>
<td>3.70E-11</td>
<td>0.096</td>
<td>1.50E-06</td>
<td>10</td>
</tr>
<tr>
<td>511</td>
<td>5.70E-11</td>
<td>0.100</td>
<td>3.70E-07</td>
<td>10</td>
</tr>
<tr>
<td>1027</td>
<td>8.60E-11</td>
<td>0.104</td>
<td>9.20E-08</td>
<td>10</td>
</tr>
<tr>
<td>2047</td>
<td>2.10E-11</td>
<td>0.112</td>
<td>2.30E-08</td>
<td>10</td>
</tr>
<tr>
<td>4095</td>
<td>5.20E-11</td>
<td>0.122</td>
<td>5.70E-09</td>
<td>10</td>
</tr>
</tbody>
</table>

7. Selected applications

7b. Anisotropic problems

• All problems considered thus far have had \( -h^2 \) as the off-diagonal entries.

• We consider two situations when the matrix has two different constants on the off-diagonals.

These situations arise when

- the (2-d) differential equation has constant but different coefficients for the derivatives in the coordinate directions
- the discretization has constant but different mesh spacing in the different coordinate directions

We consider two types of anisotropy

• Different coefficients on the derivatives

\[-u_{xx} - \alpha u_{yy} = f\]
discretized on a uniform grid with spacing \( h \).

• Different mesh spacings:

\[
h_x = h = \frac{1}{N+1} \quad \text{and} \quad h_y = \frac{h_x}{\sqrt{\alpha}}
\]
Both problems lead to the same stencil

\[ \begin{align*}
- & u_{j-1,k} + 2u_{j,k} - u_{j+1,k} \\
+ & u_{j-1,k} + 2u_{j,k} - u_{j+1,k} \\
\frac{1}{h^2} + & \alpha \frac{1}{h^2} \\
\end{align*} \]

\[ A_h^\alpha = \frac{1}{h^2} \begin{pmatrix} 1 & 2 + 2\alpha & 1 \\
-1 & 2 & -1 \\
-2 + 2\alpha & -1 & -\alpha \end{pmatrix} \]

Why standard multigrid fails

- Note that \( A_h^\alpha \) has weak connections in the \( y \)-direction. MG convergence factors degrade as \( \alpha \) gets small, with poor performance already at \( \alpha = 0.1 \).
- Consider the limiting case \( \alpha \to 0 \):
  \[ A_h = \frac{1}{h^2} \begin{pmatrix} 1 & 0 \\
-1 & 2 \\
0 & -1 \end{pmatrix} \]
- Collection of disconnected 1-D problems!
- Point relaxation will smooth oscillatory errors in the \( x \)-direction (strong connections), but with no connections in the \( y \)-direction, the errors in that direction will generally be random; point relaxation provides no smoothing in the \( y \)-direction.

We analyze weighted Jacobi

- The eigenvalues of the weighted Jacobi iteration matrix for this problem are

\[ \lambda_{i,l} = 1 - \frac{2\omega}{1+\alpha} \left( \sin^2 \left( \frac{i\pi}{2(N+1)} \right) + \alpha \sin^2 \left( \frac{l\pi}{2(N+1)} \right) \right) \]

Two strategies for anisotropy

- Semicoarsening: Because we expect MG-like convergence for the 1-D problems along lines of constant \( y \), we could coarsen the grid in the \( x \)-direction, but not in the \( y \)-direction.
- Line relaxation: Because the equations are strongly coupled in the \( x \)-direction, we could solve simultaneously for entire lines of unknowns in the \( x \)-direction (along lines of constant \( y \)).
Semicoarsening with point relaxation

- Point relaxation on $A^h = \frac{1}{h^2} \begin{pmatrix} -1 & -\alpha \\ 2+2\alpha & -1 \end{pmatrix}$ smoothes in the $x$-direction. Coarsen by removing every other $y$-line.
- We do not coarsen along the remaining $y$-lines.
- Semicoarsening is not as “fast” as full coarsening. The number of points on $\Omega^{2h}$ is about half the number of points on $\Omega^h$, instead of the usual one-fourth.

Interpolation with semicoarsening

- We interpolate in the 1-D way along each line of constant $y$.
- The formulas for interpolating the correction from the coarse to the fine grid for the 2D model problem are

$$v_{2j,k}^h = v_{2j,k}^h + v_{j,k}^{2h}$$

$$v_{2j+1,k}^h = v_{2j+1,k}^h + \frac{v_{j,k}^{2h} + v_{j+1,k}^{2h}}{2}$$

Line relaxation with full coarsening

- The other approach to this problem is to do full coarsening, but to relax entire $x$-lines (constant $y$) of variables simultaneously.
- Instead of lexicographic ordering, it’s easier here to order by $x$-lines. Write $A^h$ in block form as

$$A^h = \begin{pmatrix} D & -cI \\ -cI & D \\ & & \ddots & -cI \\ & & & -cI \\ & & & & D \end{pmatrix}$$

where $c = \frac{\alpha}{h^2}$

$$D = \frac{1}{h^2} \begin{pmatrix} 2 + 2\alpha & -1 \\ -1 & 2 + 2\alpha & -1 \\ & -1 & 2 + 2\alpha \end{pmatrix}$$

Line relaxation

- One sweep of line relaxation consists of solving a tridiagonal system for each line of constant $y$.
- The $k^{th}$ such system has the form $D v_k^h = g_k^h$, where $v_k^h$ is the $k^{th}$ subvector of $v^h$ with entries $(v_k^h)_j = v_{j,k}^h$ & the $k^{th}$ right-hand side subvector is

$$(g_k^h)_j = f_{j,k}^h + \frac{\alpha}{h^2} (v_{j,k-1}^h + v_{j,k+1}^h).$$

- Because $D$ is tridiagonal, the $k^{th}$ system can be solved very efficiently.
Why line relaxation works

- The eigenvalues of the weighted block Jacobi iteration matrix are

\[
\lambda_{i,l} = 1 - \frac{2\cos(i\pi/2)}{2\cos((l+1)/2)} + \alpha \left( \frac{1}{2} \sin^2 \left( \frac{i\pi}{2(N+1)} \right) + \alpha \sin^2 \left( \frac{l\pi}{2(N+1)} \right) \right)
\]

Semicoarsening & line relaxation

- We might not know the direction of weak coupling or it might vary.

- Suppose we want a method that can handle either

\[
A_1^h = \frac{1}{h^2} \begin{pmatrix}
-1 & -\alpha \\
\alpha & -1
\end{pmatrix}
\]

or

\[
A_2^h = \frac{1}{h^2} \begin{pmatrix}
-1 & 2 + 2\alpha \\
-\alpha & -1
\end{pmatrix}
\]

- We could use semicoarsening in the \(x\)-direction to handle \(A_1^h\) & line relaxation in the \(y\)-direction to take care of \(A_2^h\).

An anisotropic example

- Consider \( - u_{xx} - \alpha u_{yy} = f \) with \( u = 0 \) on the boundaries of the unit square, & stencil given by

\[
A^h = \frac{1}{h^2} \begin{pmatrix}
-1 & -\alpha \\
\alpha & -1
\end{pmatrix}
\]

- Suppose that \( f(x,y) = 2(y - y^2) + 2\alpha(x - x^2) \) so that the exact solution is \( u(x,y) = (y - y^2)(x - x^2) \).

- Note: If \( \alpha \) is small, then the \(x\)-direction dominates, while if \( \alpha \) is large, then the \(y\)-direction dominates.
What is smooth error?

- Consider $\alpha = 0.001$ & suppose point Gauss-Seidel is applied to a random initial guess. The error after 50 sweeps appears as:

We experiment with 3 methods

- Standard $V(2,1)$-cycling, with point Gauss-Seidel relaxation, full coarsening, & linear interpolation

- Semicoarsening in the $x$-direction. Coarse & fine grids have the same number of points in the $y$-direction. 1-D full weighting & linear interpolation are used in the $x$-direction, with no $y$-coupling in the intergrid transfers

- Semicoarsening in the $x$-direction combined with line relaxation in the $y$-direction. 1-D full weighting & interpolation.

With semicoarsening, the operator must change

- To account for unequal mesh spacing, the residual & relaxation operators must use a modified stencil

$$A = \begin{pmatrix} \frac{\alpha}{h_y^2} & - \frac{1}{h_x^2} \left( \frac{2}{h_x^2} \cdot \frac{2 \alpha}{h_y^2} \right) + \frac{1}{h_y^2} \\ - \frac{1}{h_x^2} \left( \frac{2}{h_x^2} \cdot \frac{2 \alpha}{h_y^2} \right) & \frac{\alpha}{h_y^2} \end{pmatrix}.$$  

- Note that as grids become coarser, $h_x$ grows while $h_y$ remains constant.

How do the 3 methods work for various values of $\alpha$?

$N = 15$

- Asymptotic convergence factors:

<table>
<thead>
<tr>
<th>scheme</th>
<th>$\alpha = 1000$</th>
<th>$\alpha = 100$</th>
<th>$\alpha = 10$</th>
<th>$\alpha = 1$</th>
<th>$\alpha = 0.1$</th>
<th>$\alpha = 0.01$</th>
<th>$\alpha = 0.001$</th>
<th>$\alpha = 1E-04$</th>
</tr>
</thead>
<tbody>
<tr>
<td>V(2,1)-cycles</td>
<td>0.95</td>
<td>0.94</td>
<td>0.58</td>
<td>0.13</td>
<td>0.58</td>
<td>0.90</td>
<td>0.95</td>
<td>0.95</td>
</tr>
<tr>
<td>semic in $x$</td>
<td>0.94</td>
<td>0.99</td>
<td>0.98</td>
<td>0.93</td>
<td>0.71</td>
<td>0.28</td>
<td>0.07</td>
<td>0.07</td>
</tr>
<tr>
<td>semic / line relax</td>
<td>0.04</td>
<td>0.08</td>
<td>0.08</td>
<td>0.08</td>
<td>0.07</td>
<td>0.07</td>
<td>0.08</td>
<td>0.08</td>
</tr>
</tbody>
</table>

- Note: semicoarsening in $x$ works well for $\alpha < 0.001$ but degrades noticeably even at $\alpha = 0.1$.  

y-direction strong  

x-direction strong
A semicoarsening subtlety

• Suppose $\alpha$ is small, so that semicoarsening in $x$ is used. As we progress to coarser grids, $h_x^{-2}$ gets small but $h_y^{-2}$ remains constant.

• If, on some coarse grid, $h_x^{-2}$ becomes comparable to $\alpha h_y^{-2}$, then the problem effectively becomes recoupled in the $y$-direction. Continued semicoarsening can produce artificial anisotropy, strong in the $y$-direction.

• When this occurs, it is best to stop semicoarsening & use full coarsening on any further coarse grids.

7. Selected applications

7c. Variable meshes

• Non-uniform grids are commonly used for domain or data irregularities or emerging solution features.

• Consider how we might approach the 1-D problem

$$- u''(x) = f(x), \quad 0 < x < 1,$$

$$u(0) = u(1) = 0$$

posed on the following nonuniform grid:

We need some notation for the mesh spacing

Let $N$ be a positive integer. We define the spacing interval between $x_j$ & $x_{j+1}$:

$$h_{j+1/2} = x_{j+1} - x_j, \quad j = 0, 1, \ldots, N.$$

Building second divided differences

$$u_{j+1/2}'' = (u_{j+1/2} - u_{j-1/2})/h_j$$

$$u_{j+1/2}' = (u_{j+1} - u_j)/h_{j+1/2}$$

$$u_{j-1/2}' = (u_{j-1} - u_j)/h_{j-1/2}$$

$$h_j = (h_{j-1/2} + h_{j+1/2})/2$$
We define the discrete differential operator

- Using second-order finite differences (and wading through a mess of algebra), we obtain the discrete representation
  \[-\alpha_j^h u_{j-1}^h + (\alpha_j^h + \beta_j^h) u_j^h - \beta_j^h u_{j+1}^h = f_j^h \quad 1 \leq j \leq N\]
  \[u_0^h = u_{N+1}^h = 0\]

where
  \[\alpha_j^h = \frac{2}{h_{j-1/2}(h_{j-1/2} + h_{j+1/2})} \quad \text{and} \quad \beta_j^h = \frac{2}{h_{j+1/2}(h_{j-1/2} + h_{j+1/2})}.\]

- Multiply by \((h_{j-1/2} + h_{j+1/2})/2\) yields an SPD matrix.

We modify standard multigrid to accommodate variable spacing

- We choose every other fine-grid point as a coarse-grid point:
  \[x_0^h, x_2^h, \ldots, x_{(N+1)/2}^h\]

In \([x_0^h, x_2^h]\), linear means
  \[v(x) = v_0^{2h} + (v_1^{2h} - v_0^{2h})(x - x_0^h)/(x_2^h - x_0^h).\]

- Now just plug in \(x = x_j^h\) to get...

Interpolation formula

If \(v^h = I_{2h}^h v^{2h}\), then, for \(1 \leq j \leq (N - 1)/2\), we have
  \[v_{2j}^h = v_{2j}^{2h}, \quad v_{2j+1}^h = \frac{h_{2j+3/2} v_{2j}^{2h} + h_{2j+1/2} v_{2j+1}^{2h}}{h_{2j+1/2} + h_{2j+3/2}}.\]

We use the variational properties to derive restriction \(A^{2h}\)

\[A^{2h} = I_{2h}^h A^h I_{2h}^h \quad I_{2h}^h = \frac{1}{2} \left( I_{2h}^h \right)^T.\]

- This produces a stencil on \(\Omega^{2h}\) that is similar, but not identical, to the fine-grid stencil. If the resulting system is scaled by \((h_{j-1/2} + h_{j+1/2})\), then the Galerkin product is the same as the fine-grid stencil.

- For 2-D problems, this approach can be generalized readily to tensor-product grids. However, for general irregular grids, AMG is a better choice.
7. Selected applications

7d. Variable coefficients

- A common difficulty is variable coefficients, exemplified in 1-D by
  
  \[-(a(x)u'(x))' = f(x),\quad 0 < x < 1,\]
  
  \[u(0) = u(1) = 0,\]

  where \(a(x)\) is a positive function on \([0,1]\).

- We seek to develop a conservative, or self-adjoint, method for discretizing this problem.

- Assume we have available to us the values of \(a(x)\) at midpoints of the uniform grid:

\[x_j^h = x_j - \frac{h}{2}, \quad x_{j+1/2}^h = x_j + \frac{h}{2},\]

\[0 = x_0^h < x_1^h < \cdots < x_N^h = 1\]

Discretize using central differences

We can use second-order differences to approximate the derivatives. To use a grid spacing of \(h\), we evaluate \(a(x)u(x)\) at points midway between the gridpoints:

\[\frac{(a(x)u'(x))'_{j+1/2} - (a(x)u'(x))'_{j-1/2}}{h} = O(h^2)\]

The basic stencil

We combine the differences for \(u'\) & for \((au')'\) to obtain the operator

\[-(a(x)u'(x))(x_j) \approx -\frac{a_{j+1/2}^N}{h} \left(\frac{u_{j+1} - u_j}{h}\right) - a_{j-1/2}^N \left(\frac{u_j - u_{j-1}}{h}\right)\]

\[\frac{1}{h^2} \left(-a_{j-1/2}^N u_{j-1} + a_{j+1/2}^N u_j - a_{j+1/2}^N a_{j-1/2}^N u_{j-1}\right) = f_j, \quad 1 \leq j \leq N\]

\[u_0 = u_{N+1} = 0.\]
Coarsening the variable coefficient problem

- A reasonable approach is to use a standard multigrid algorithm with linear interpolation, full weighting, & the stencil
  \[ A^{2h} = \frac{1}{(2h)^2} \left[ -a^{2h}_{j-1/2} a^{2h}_{j+1/2} + a^{2h}_{j+1/2} - a^{2h}_{j+1/2} \right] \]
  where
  \[ a^{2h}_{j+1/2} = \frac{a^{2h}_{j+1/2} + a^{2h}_{j+1/2}}{2} \]
- The same stencil is obtained by the Galerkin relation.

Variable mesh vs. variable coefficients after scaling by \( \eta_j = (h_{j-1/2} + h_{j+1/2})/2 \) & \( h \)

- Variable mesh
  \[ \frac{1}{h_{j-1/2}} u_{j-1}^h + \left( \frac{1}{h_{j-1/2}} + \frac{1}{h_{j+1/2}} \right) u_j^h - \frac{1}{h_{j+1/2}} u_{j+1}^h = \eta_j f_j^h \]
- Variable coefficients
  \[ \frac{1}{h} \left(-a_{j-1/2} u_{j-1} + (a_{j-1/2} + a_{j+1/2}) u_j - a_{j+1/2} u_{j+1}\right) = hf_j \]
- Correspondence
  \[ \frac{1}{h_{j-1/2}} \frac{a_{j-1/2}}{h} \quad \frac{1}{h_{j+1/2}} \frac{a_{j+1/2}}{h} \]

A variable coefficient example

- We use V(2,1) cycle, full weighting, linear interpolation.
- We use \( a(x) = 1 + \rho \sin(k\pi x) \) & \( a(x) = 1 + \rho \text{rand}(x) \).

\[ N = 1024 \]
\[ a(x) = 1 + \rho \sin(k\pi x) \quad a(x) = 1 + \rho \text{rand}(x) \]

<table>
<thead>
<tr>
<th>( \rho )</th>
<th>( k=3 )</th>
<th>( k=25 )</th>
<th>( k=50 )</th>
<th>( k=100 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.085</td>
<td>0.085</td>
<td>0.085</td>
<td>0.085</td>
</tr>
<tr>
<td>0.25</td>
<td>0.084</td>
<td>0.098</td>
<td>0.098</td>
<td>0.094</td>
</tr>
<tr>
<td>0.5</td>
<td>0.093</td>
<td>0.185</td>
<td>0.194</td>
<td>0.196</td>
</tr>
<tr>
<td>0.75</td>
<td>0.119</td>
<td>0.374</td>
<td>0.387</td>
<td>0.391</td>
</tr>
<tr>
<td>0.85</td>
<td>0.142</td>
<td>0.497</td>
<td>0.511</td>
<td>0.514</td>
</tr>
<tr>
<td>0.95</td>
<td>0.191</td>
<td>0.681</td>
<td>0.69</td>
<td>0.694</td>
</tr>
</tbody>
</table>

Standard multigrid degrades if \( a(x) \) is highly variable

- MG for variable coefficients is equivalent to MG (with simple averaging) for Poisson's equation on a variable mesh.
- \((au)' = f\)
- \(-u'' = f\)
- But simple averaging won't accurately represent smooth components if \( x_j^{2h} \) is close to \( x_{j+1}^{2h} \) but far from \( x_{j+2}^{2h} \).
One remedy is to apply operator-induced interpolation

- Assume that relaxation does not change smooth error, so the residual is approximately zero.

Applying at \( x_{j+1}^h \) yields

\[
-a_{j+1/2}^h e_{j+1}^h + (a_{j+1/2}^h + a_{j+3/2}^h) e_{j+1}^h - a_{j+3/2}^h e_{j+2}^h + h^2 = 0
\]

- Solving for \( e_{j+1}^h \):

\[
e_{j+1}^h = \frac{a_{j+1/2}^h e_{j+1}^h + a_{j+3/2}^h e_{j+2}^h + h^2}{a_{j+1/2}^h + a_{j+3/2}^h}
\]

Thus, operator-induced interpolation is

\[
v_{2j+1}^h = v_{2j}^h e_{j+1}^h + a_{j+1/2}^h v_{j+1}^h + a_{j+3/2}^h v_{j+2}^h + h^2
\]

The case \( a(x) \equiv 1 \) reduces to standard interpolation:

\[
v_{j+1}^h = \frac{v_{j}^h + v_{j+1}^h}{2}
\]

As usual, the restriction & coarse-grid operators are defined by the Galerkin relations:

\[
A^h = I^h_2 A^h I_2^h, \quad I_2^h = c(I_2^h)^T.
\]

Outline

Chapters 1-5:
- Model Problems
- Basic Iterative Methods
- Convergence tests
- Analysis
- Elements of Multigrid
- Relaxation
- Coarsening
- Implementation
- Complexity
- Diagnostics
- Some Theory
  - Spectral vs. algebraic

Chapters 6-10:
- Nonlinear Problems
  - Full approximation scheme
  - Selected Applications
  - Neumann boundaries
  - Anisotropic problems
  - Variable meshes
  - Variable coefficients
  - Algebraic Multigrid (AMG)
  - Matrix coarsening
  - Multilevel Adaptive Methods
    - FAC
    - Finite Elements
    - Variational methodology

8. Algebraic multigrid (AMG)
unstructured grids, variable coefficients, ...

- Automatically defines coarse "grid".
- AMG has two distinct phases:
  - setup phase: define MG components.
  - solution phase: perform MG cycles.
- The AMG approach differs from geometric MG from the start:
  - fix relaxation (point Gauss-Seidel).
  - choose coarse "grids" & prolongation, \( P \), so that error not reduced by relaxation is in range(\( P \)).
  - AMG uses the Galerkin principle so that coarse-grid correction eliminates error in range(\( P \)).
- Assume \( A \) is SPD unless otherwise stated.
**AMG has two phases**

- **Setup Phase**
  - Select coarse "grids," \( \Omega^{m+1}, m = 1, 2, \ldots \)
  - Define interpolation, \( I_m^{m+1}, m = 1, 2, \ldots \)
  - Define restriction & coarse-grid operators,
    \( I_m^{m+1} = (I_m^m)^T, A^{m+1} = I_m^{m+1} A^m I_m^{m+1} \).

- **Solve Phase**

  Standard MG processes: V-cycle, W-cycle, FMG, FAS, ...

- Only the selection of coarse grids does not parallelize well using existing techniques!

**AMG fundamental concept:** smooth error = “small” residuals/energy

- Error propagation via weighted Jacobi smoothing:
  \[ e^{k+1} = (I - \omega D^{-1} A) e^k. \]
- Error that is slow to converge satisfies
  \[ (I - \omega D^{-1} A) e \approx e \implies \omega D^{-1} A e \approx 0 \]
  \[ \implies r \approx 0 \]
- A little more precisely, assuming that \( \omega D \approx I \), then slow-to-converge error has relatively small energy:
  \[ \langle Ae, e \rangle \ll \langle e, e \rangle \]

**Operator-induced interpolation**

Pretend that we have a graph of \( A \) & that we've identified coarse points

- Coarse point
- Fine point \( i \)
- Another F-point

**AMG uses strong connection to determine MG components**

- **Smoothing assumption:**
  \[ r \approx 0 \quad \text{or} \quad \langle Ae, e \rangle \approx 0. \]
- We say that \( i \) is strongly connected to \( j \) if
  \[ -a_{ij} \geq \theta \max_{k \neq i} \{-a_{ik}\}, \quad 0 < \theta \leq 1. \]
- For M-matrices, some algebra leads to
  \[ \langle Ae, e \rangle = \sum_{i \neq j} \frac{a_{ij}}{2} (e_i - e_j)^2 \approx 0. \]
- So smooth error is more or less constant along strong connections.
Some useful set definitions

• The set of strong connections of a variable \( u_i \), that is, the variables upon whose values the value of \( u_i \) depends, is defined as
  \[ S_i = \left\{ j : -a_{ij} \geq \theta \max_{k \neq i} \{-a_{ik}\} \right\}. \]

• The set of points strongly connected to a variable \( u_i \) is denoted \( S_i^T = \{ j : i \in S_j \} \).

• The set of coarse-grid variables is denoted \( C \).

• The set of fine-grid variables is denoted \( F \).

• The set of interpolatory coarse-grid variables used to interpolate the value of the fine-grid variable \( u_i \) is denoted \( C_i \).

Choosing the coarse grid

• Two Criteria
  
  - (C1) For each \( i \in F \), each point \( j \in S_i \) should either be in \( C \) or should be strongly connected to at least one point in \( C_i \).
  
  - (C2) \( C \) should be a maximal subset with the property that no two \( C \)-points are strongly connected to each other.

  • Satisfying (C1) & (C2) is sometimes impossible. We use (C2) as a guide while enforcing (C1).

Selecting the coarse-grid points

C-point selected (point with largest “value”)

Neighbors of C-point become F-points

Next C-point selected (after updating “values”)

F-points selected, etc.

Examples: Laplacian operator

5-pt FD, 9-pt FE (quads), & 9-pt FE (stretched quads)
**Prolongation** is based on smooth error, strong connections (from M-matrices)

Smooth error is given by:

\[ r_i = a_{ii} e_i + \sum_{j \in N_i} a_{ij} e_j = 0 \]

For strongly connected points, use:

\[ a_{ii} e_i = - \sum_{j \notin i} a_{ij} e_j \]

\[ a_{ij} e_i = - \sum_{j \in C_i} a_{ij} e_j - \sum_{j \in D_i^C} a_{ij} e_j - \sum_{j \in D_i^W} a_{ij} e_j \]

Finally, the prolongation weights are defined

In the smooth-error relation, use \( e_j = e_i \) for weak connections. For strong \( F \)-points, use

\[ e_j = \left( \sum_{k \in C_i} a_{jk} e_k \right) / \left( \sum_{k \in C_i} a_{jk} \right) \]

yielding the prolongation weights:

\[ a_{ij} + \sum_{j \in D_i^C} a_{ik} e_k = - \sum_{j \in D_i^C} a_{ik} \]

\[ w_{ij} = - \frac{a_{ii} + \sum_{n \in D_i^F} a_{in}}{a_{ii}} \]

**AMG setup costs**

a bad rap

- Many geometric MG methods need to compute prolongation & coarse-grid operators.
- The only additional expense in the AMG setup phase is the coarse-grid selection algorithm.
- AMG setup phase is only 10-25% more expensive than in geometric MG & may be considerably less than that!
**AMG performance:** Sometimes a success story

- AMG performs extremely well on the model problem (Poisson's equation, regular grid)- optimal convergence factor (e.g., 0.14) & scalability w.r.t. problem size.

- AMG appears to be both scalable & efficient on diffusion problems on unstructured grids (e.g., 0.1-0.3).

- AMG handles anisotropic diffusion coefficients on irregular grids reasonably well.

- AMG handles anisotropic operators on structured & unstructured grids relatively well (e.g., 0.35).

---

**How does it perform (vol I)?**

regular grids, plain, old, vanilla problems, unit square, N = 64, Dirichlet boundaries

- The Laplace operator:

<table>
<thead>
<tr>
<th>Stencil</th>
<th>Convergence per cycle</th>
<th>Complexity per Cycle</th>
<th>Time per Cycle</th>
<th>Setup Times</th>
</tr>
</thead>
<tbody>
<tr>
<td>5-pt</td>
<td>0.054</td>
<td>2.21</td>
<td>0.29</td>
<td>1.63</td>
</tr>
<tr>
<td>5-pt skew</td>
<td>0.067</td>
<td>2.12</td>
<td>0.27</td>
<td>1.52</td>
</tr>
<tr>
<td>9-pt (-1, 8)</td>
<td>0.078</td>
<td>1.30</td>
<td>0.26</td>
<td>1.83</td>
</tr>
<tr>
<td>9-pt (-1, -4, 20)</td>
<td>0.109</td>
<td>1.30</td>
<td>0.26</td>
<td>1.83</td>
</tr>
</tbody>
</table>

- Anisotropic 5-Point Laplacian: $-\varepsilon \frac{U_{xx}}{U_{yy}}$

<table>
<thead>
<tr>
<th>Epsilon</th>
<th>Convergence/cycle</th>
<th>0.001</th>
<th>0.01</th>
<th>0.1</th>
<th>0.5</th>
<th>1</th>
<th>2</th>
<th>10</th>
<th>100</th>
<th>1000</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>0.084</td>
<td>0.093</td>
<td>0.058</td>
<td>0.069</td>
<td>0.054</td>
<td>0.079</td>
<td>0.087</td>
<td>0.093</td>
<td>0.083</td>
</tr>
</tbody>
</table>

---

**How does it perform (vol II)?**

structured meshes, rectangular domains

5-point Laplacian on regular rectangular grids

Convergence factor (y-axis) plotted against number of nodes (x-axis)

---

**How does it perform (vol III)?**

unstructured meshes, rectangular domains

Laplacian on random unstructured grids (regular triangulations, 15-20% nodes randomly collapsed into neighboring nodes)

Convergence factor (y-axis) plotted against number of nodes (x-axis)
How does it perform (vol IV)?

How does it perform (vol V)?

Laplacian operator, unstructured grids

Problems used: "a" means parameter \( c = 10 \), "b" means \( c = 1000 \)

6: \( d(x, y) = 1.0 + c |x - y| \) 
7: \( d(x, y) = \begin{cases} 1.0 & x \leq 0.5 \\ c & x > 0.5 \end{cases} \) 
8: \( d(x, y) = \begin{cases} 1.0 & 0.125 \leq \max \{|x-0.5|, |y-0.5|\} \leq 0.25 \\ c & \text{otherwise} \end{cases} \) 
9: \( d(x, y) = \begin{cases} 1.0 & 0.125 \leq \sqrt{(x-0.5)^2 + (y-0.5)^2} \leq 0.25 \\ c & \text{otherwise} \end{cases} \)

AMG for systems

- How can we do AMG on systems?
  \[
  \begin{pmatrix}
  A_{11} & A_{12} \\
  A_{21} & A_{22}
  \end{pmatrix}
  \begin{pmatrix}
  u \\
  v
  \end{pmatrix} =
  \begin{pmatrix}
  f \\
  g
  \end{pmatrix}
  \]

- Naïve approach: "Block" AMG (block Gauss-Seidel, using scalar AMG to "solve" at each cycle)
  \[
  u \leftarrow (A_{11})^{-1}(f - A_{12}v) \\
  v \leftarrow (A_{22})^{-1}(g - A_{21}u)
  \]

Great idea! Except that it often doesn’t work!

Block AMG doesn’t account for strong inter-variable coupling.

AMG for systems: A solution

- To solve the system problem, allow interaction between the unknowns at all levels:
  \[
  A^k = \begin{pmatrix}
  A_{11}^k & A_{12}^k \\
  A_{21}^k & A_{22}^k
  \end{pmatrix} \\
  \] & \[
  I_{k+1}^k = \begin{pmatrix}
  I_{k+1}^k & 0 \\
  0 & I_{k+1}^k
  \end{pmatrix}
  \]

- This is called the "unknown-based" approach.

- 2-D biharmonic, Dirichlet boundaries, unit square, uniform quadrilateral mesh:

| Mesh spacing | 0.125 | 0.0625 | 0.03125 | 0.015625 |
| Convergence factor | 0.22 | 0.35 | 0.42 | 0.44 |
Adaptive AMG (αAMG) to broaden applicability

adaptive interpolation to determine sense of smoothness

+ adaptive C-point choice to determine good coarse point

---

Standard interpolation

Standard AMG collapses stencils by assuming smooth error is locally constant (Poisson "sense of smoothness"):

\[ \begin{align*}
\text{*** smooth } e & \approx c \text{ ***} \\
\text{constant}
\end{align*} \]

\[
e_j = \left( \sum_{k \in C_i} a_{jk} e_k \right) / \left( \sum_{k \in C_i} a_{jk} \right)
\]

Strong C Strong F Weak pts.

---

Isn't standard interpolation OK? Isn't smooth \( e \approx c \) (constant) always?

- Suppose someone tried to make \( A \) "nice" for relaxation by scaling the diagonal so it's the identity:

\[
A \leftarrow D^{-1/2} A D^{-1/2}, \quad D = \text{diag}(a_{ii}).
\]

- Relaxation still gives small residuals:

\[
A e \approx 0.
\]

- But:

\[
A e = D^{-1/2} A D^{-1/2} e \approx 0 \Rightarrow A D^{-1/2} e \approx 0 \Rightarrow e \approx D^{1/2} c.
\]

- So "smooth" here means \( e_i \approx c_i \sqrt{D_{ii}} \). This could vary a lot!

---

Pre-relaxation

- What if we found a smooth error \( x \) that's far from \( c \)?

- If we found that \( x_j = 1.6 x_i \) for \( j \in D^x \), say, then we could just make the replacement \( e_j \rightarrow 1.6 e_i \).

- If, say, \( x_j = 0.2 x_k \) for \( j \in D^x \) & all \( k \in C_i \), then we could set

\[
e_j \rightarrow \left( \sum_{j \in C_i} 0.2 e_k \right) / \left( \sum_{j \in C_i} a_{jk} \right).
\]

- How do we obtain a representative smooth error \( x \)?

Relax on \( A x = 0 \) !!!
Adaptive C-point choice

- AMG uses the ansatz that residuals are 0 at F-points.
- What if we had $r_f = 0$ exactly?

$$\begin{pmatrix} A_{ff} & A_{fc} \\ A_{cf} & A_{cc} \end{pmatrix} \begin{pmatrix} e_f \\ e_c \end{pmatrix} = \begin{pmatrix} 0 \\ r_c \end{pmatrix} \Rightarrow A_{ff}e_f + A_{fc}e_c = 0 \Rightarrow \begin{pmatrix} e_f \\ e_c \end{pmatrix} = \begin{pmatrix} A_{ff}^{-1}A_{fc} \\ I \end{pmatrix} e_c$$

- Choosing $P = \begin{pmatrix} -A_{ff}^{-1}A_{fc} \\ I \end{pmatrix}$ (ideal interpolation) makes MG exact:

$$MGe = [CG]Pe_c = \left[I - P[P^T AP]^{-1}P^T A\right]Pe_c = Pe_c - P[P^T AP]^{-1}[P^T AP]e_c = 0.$$  

Compatible relaxation (CR)

- How do we make the residuals zero at F-points?

$$\text{Relax on } A_{ff}x_f = -A_{fc}x_c!$$

- To ensure that this works so that $r_f \to 0$ fast, we want $A_{ff}$ to be well conditioned. Thus, we can assess whether we have good C-points by "CR":

$$A_{ff}x_f = 0.$$  

- Quick CR convergence means that the F-point residuals will be small after relaxation & that there is a $P$ (the ideal one) that gives good MG convergence.
- Is there a good local $P$? Maybe, since CR is local & fast.
- Hopefully, we can use pre-relaxation to compute a good $P$.

Outline

Chapters 1-5:
- Model Problems
- Basic Iterative Methods
  - Convergence tests
  - Analysis
- Elements of Multigrid
  - Relaxation
  - Coarsening
- Implementation
  - Complexity
  - Diagnostics
- Some Theory
  - Spectral vs. algebraic

Chapters 6-10:
- Nonlinear Problems
  - Full approximation scheme
- Selected Applications
  - Neumann boundaries
  - Anisotropic problems
  - Variable meshes
  - Variable coefficients
- Algebraic Multigrid (AMG)
  - Matrix coarsening
- Multilevel Adaptive Methods
  - FAC
- Finite Elements
  - Variational methodology

9. Multilevel adaptive methods

- fast adaptive composite grid method (FAC)

- two-spike problem

- Local enhancement to resolve special regions of activity or interest.
**Model 1-D problem**

- \( u''(x) = f(x), \quad 0 < x < 1 \)
- \( u(0) = u(1) = 0 \)

\[
\Omega^h = \{ x_0^h, x_1^h, x_2^h, x_3^h, x_4^h, x_5^h, x_6^h, x_7^h, x_8^h \}
\]

\[
h^{-2}(-u_{i-1}^h + 2u_i^h - u_{i+1}^h) = f_i^h, \quad 1 \leq i \leq 7
\]

\( u_0^h = u_8^h = 0, \quad h = \frac{1}{8} \)

**Local refinement**

- Suppose \( f(x) \) has a spike at \( x = 3/4 \)
- but is smooth elsewhere

**Strategy**

- Recognize that there’s little value of having the fine grid in the smooth region, \([0, 1/2] \).

- Start with uniform grid & standard MG, then:
  - first eliminate relaxation in \([0, 1/2] \); and
  - then eliminate intergrid transfers & residual calculations.

- Then interpret this process via the composite grid (= \( 2h \)-points in \([0, 1/2] \) + \( h \)-points in \([1/2, 1] \)).

- We’ll try absurdly hard to eliminate work @ \( x = 1/2 \), but we have in mind multi-dimensions & smaller patches.

**Local-Relaxation/Global-Correction MG**

- WARNING: You won’t get zero residuals on all of grid \( h \).

Remember that the concept is right here, but we need to make it efficient---without changing the results!!!

Consider the usual global grids. Relax only in the local region. \( x = 1/2 \) becomes a boundary point.

Compute & transfer residuals.

Transfer & add correction.

We start by using local relaxation & eliminating unnecessary residual transfers.

To save all of the work of computing & transferring residuals & corrections in regions where they don’t change, we need some messy ALGEBRA!

Trust me on the more messy stuff...
Eliminate relaxation

so $r^h$ changes only on $2h$, $v^h$ changes only here

Initialize $v^h = 0$ & $f_1^{2h} \leftarrow (f_1^h + 2f_2^h + f_3^h)/4$.
Relax on $v^h$ on the local fine grid ($x_5^h$, $x_6^h$, $x_7^h$).
Compute $r^h = f^h - A^h v^h$ & transfer to $2h$:
$f_2^{2h} \leftarrow (r_3^h + 2r_5^h + r_7^h)/4$ & $f_3^{2h} \leftarrow (r_5^h + 2r_6^h + r_7^h)/4$.
Compute an approximation, $v_2^h$, to the solution of the $2h$ residual equation, $A^{2h} u^{2h} = f^{2h}$.
Update the residual at $x_i^{2h}$ for later cycles:
$f_1^{2h} \leftarrow f_1^{2h} - (-v_0^{2h} + 2v_1^{2h} - v_2^{2h})/(2h)^2$.
Correct: $v^h \leftarrow v^h + I_{2h}^h v_2^h$.

Why save $v^h$ outside of local region? 293 of 322

Eliminate the rest

$r_3^h$ doesn't change on $h$.

Compute change in $r_3^h$ on $2h$.

$2h$ residual @ $x_i^{2h} = g_{2i}^{2h} = (-w_{i}^{2h} + 2v_i^{2h} + v_0^{2h})/(2h)^2$,
where $g_{2i}^{2h} = (f_3^h + 2f_4^h + f_5^h)/4$. (Messy algebra)

Initialize $v^h = 0$, $w_{i}^{2h} = 0$, & $f_1^{2h} \leftarrow (f_1^h + 2f_2^h + f_3^h)/4$.
Relax on $v^h$ on the local fine grid ($x_5^h$, $x_6^h$, $x_7^h$).
Compute the right sides for $2h$:
$f_2^{2h} \leftarrow g_{2i}^{2h} - (-w_{i}^{2h} + 2v_i^{2h} - v_0^{2h})/(2h)^2$ & $f_3^{2h} \leftarrow (r_5^h + 2r_6^h + r_7^h)/4$.
Compute an approximation, $v_2^h$, to the solution of the $2h$ residual equation, $A^{2h} u^{2h} = f^{2h}$.
Update the residual at $x_i^{2h}$ for later cycles:
$f_1^{2h} \leftarrow f_1^{2h} - (-v_0^{2h} + 2v_1^{2h} - v_2^{2h})/(2h)^2$.
Accumulate the $2h$ approximation: $w_{i}^{2h} \leftarrow w_{i}^{2h} + v_i^{2h}$.
Correct: $v^h \leftarrow v^h + I_{2h}^h v_2^h$ @ interface. 295 of 322

Eliminate more

$f_2^{2h}$ involves $r_3^h$ & $r_4^h$.
Store $v^h$ only @ $x_5^h$, $x_6^h$, $x_7^h$ & save $v_2^h$ on $2h$ (call it $w_{i}^{2h}$).

Initialize $v^h = 0$, $w_{i}^{2h} = 0$, & $f_1^{2h} \leftarrow (f_1^h + 2f_2^h + f_3^h)/4$.
Relax on $v^h$ on the local fine grid ($x_5^h$, $x_6^h$, $x_7^h$).
Compute $r^h = f^h - A^h v^h$ & transfer to $2h$:
$f_2^{2h} \leftarrow (r_3^h + 2r_4^h + r_5^h)/4$ & $f_3^{2h} \leftarrow (r_5^h + 2r_6^h + r_7^h)/4$.
Compute an approximation, $v_2^h$, to the solution of the $2h$ residual equation, $A^{2h} u^{2h} = f^{2h}$.
Update the residual at $x_i^{2h}$ for later cycles:
$f_1^{2h} \leftarrow f_1^{2h} - (-v_0^{2h} + 2v_1^{2h} - v_2^{2h})/(2h)^2$.
Accumulate the $2h$ approximation: $w_{i}^{2h} \leftarrow w_{i}^{2h} + v_1^{2h}$.
Correct: $v^h \leftarrow v^h + I_{2h}^h v_2^h$. 294 of 322

Interpretation composite grid

What equation in $v^h$ are we solving?

usual

$(-v_0^c + 2v_1^c - v_2^c)/(2h)^2 = f_1^c \equiv (f_1^h + 2f_2^h + f_3^h)/4$

$(-v_{i+1}^c + 2v_i^c - v_{i-1}^c)/h = f_i^c \equiv f_{i+2}^h$, $i = 3, 4, 5$

$( - v_1^c + 3v_2^c - 2v_3^c )/(2h)^2 = f_2^c \equiv (f_3^h + 2f_4^h)/4$ @interface

→ → The key is to treat the interface correctly. →
### Issues

- Adaptivity
- Error estimates
- Norms (proper scaling)
- Multiple dimensions
  - Slave points
  - More complicated stencils
- Data structures
- Parallel algorithms (AFAC)
- Time-space

### Two-spike example (Laplacian)

- Global grid $h$ & one double-patch refined level $h/2$.
- $V(1,0)$-cycles, Gauss-Seidel.
- Asymptotic convergence of the solver.
- Scaled $L^2$ discretization error estimate.

<table>
<thead>
<tr>
<th>Global $h$</th>
<th>Convergence factor</th>
<th>Discrete $L^2$ norm of discretization error</th>
</tr>
</thead>
<tbody>
<tr>
<td>1/32</td>
<td>0.362</td>
<td>2.34e-2</td>
</tr>
<tr>
<td>1/64</td>
<td>0.367</td>
<td>5.742-3</td>
</tr>
<tr>
<td>1/128</td>
<td>0.365</td>
<td>1.43e-3</td>
</tr>
</tbody>
</table>

### Outline

#### Chapters 1-5:
- Model Problems
- Basic Iterative Methods
  - Convergence tests
  - Analysis
- Elements of Multigrid
  - Relaxation
  - Coarsening
- Implementation
  - Complexity
  - Diagnostics
- Some Theory
  - Spectral vs. algebraic

#### Chapters 6-10:
- Nonlinear Problems
  - Full approximation scheme
- Selected Applications
  - Neumann boundaries
  - Anisotropic problems
  - Variable meshes
  - Variable coefficients
- Algebraic Multigrid (AMG)
  - Matrix coarsening
- Multilevel Adaptive Methods
  - FAC
- Finite Elements
  - Variational methodology

### 10. Finite elements

FE, a variational discretization methodology

FD: differences at nodes & truncation error.  
FE: weak form on discrete functions & approximation property

Other methods: finite volume, collocation, spectral, ...
**Discretization**

What do you see?

→ →FIRST KEY POINT ← ←

FE sees grid points (nodes or dofs) only as characterizations of continuum functions.

**Continuous piecewise linear functions**

???[−] u'' = f ???[−]

How do you take 2nd derivatives of uh?

→ →THIRD KEY POINT ← ←

FE integrates away 2nd derivatives.

**Localize**

What choice do you have?

→ →SECOND KEY POINT ← ←

FE functions are localized to ensure a sparse matrix.

**Weak form!**

−u'' = f

⇒ −u''v = fv ∀v

⇒ ∫(−u'')vdx = ∫fvdx ∀v

⇒ ∫u'v'dx − u'|_0^1 = ∫fvdx ∀v

⇒ ∫u'v'dx = ∫fvdx ∀v v(0) = v(1) = 0

⇒ (u',v') = (f,v) ∀v v v(0) = v(1) = 0

u^h

Need basis for space of admissible uh & v.

The ⇒ can actually be reversed to ⇔ if u is smooth enough.

So the weak & strong forms are "equivalent".
**Representation**

A basis of “hat” functions

$\hat{e}_{(i)}^h$ is continuous piecewise linear & 0 at all nodes except node $i$ where it’s 1.

Any continuous piecewise linear function can be represented by

$$u^h = \sum_i u_i^h \hat{e}_{(i)}^h.$$  

Now we’re back to using node values.

**FE basics**

Self-adjoint positive definite $\Rightarrow$ weak functional.

- **Model problem** $\Omega \subset \mathbb{R}^2$:
  
  $Lu = -u'' - u'' = f$ in $\Omega$
  
  $u = 0$ on $\partial \Omega$

- **Sobolev spaces (wider than classical spaces!):**
  
  $L^2(\Omega) = \{u : \int_\Omega u^2 d\Omega < \infty\}$
  
  $H^1_0(\Omega) = \{u : u, u', u, u \in L^2(\Omega), \|u\|_{H^1_0} = 0\}$
  
  $(u, v) = \int_\Omega u v d\Omega$

- **$L$ is self-adjoint positive definite (more later):**
  
  $(Lu, v) = (v, v) = \int_\Omega (u' v' + u v') d\Omega = (Lu, v')$
  
  $(Lu, v) > 0$ if $u \neq 0$

  We use $(Lu, v)$ for simplicity, but really we mean $(\nabla u, \nabla v)$

- **Duality:** Solving $Lu = f$ is equivalent to minimizing the weak functional
  
  $F(u) = \frac{1}{2} (Lu, u)' - (f, u)$
  
  $= \frac{1}{2} (v, v) - (f, u)$

- **Short story:**
  
  1\textsuperscript{st} derivative test
  
  $\nabla F(u) = Lu - f = 0$

  2\textsuperscript{nd} derivative test
  
  $F''(u) = L > 0$

  This is formal: $Lu$ is not defined on all of $H^1_0(\Omega)$.

**Weak form**

\[(u', v') = (f, v) \quad \forall v \ni v(0) = v(1) = 0.\]

\[
\left( \sum_j u_j^h \hat{e}_{(j)}', v' \right) = (f, v) \quad \forall v \ni v(0) = v(1) = 0.
\]

\[
\left( \sum_j u_j^h \hat{e}_{(j)}', \hat{e}_{(i)}' \right) = (f, \hat{e}_{(i)}') \quad \forall i.
\]

\[
\sum_j (\hat{e}_{(i)}', \hat{e}_{(j)}') u_j^h = (f, \hat{e}_{(i)}') \quad \forall i.
\]

So the matrix is

$$A^h = \left( a_{ij}^h \right) = \left( (\hat{e}_{(i)}', \hat{e}_{(j)}') \right).$$

**Long story**

- Using symmetry & linearity of $L$ & bilinearity of the inner product:

  \[
  F(u + v) = (L(u + v), u + v)/2 - (f, u + v)
  \]

  \[
  = (Lu, u)/2 + (Lu, v) + (Lv, v)/2
  \]

  \[
  - (f, u) - (f, v)
  \]

  \[
  = F(u) + (Lu, v) - (f, v) + (Lv, v)/2
  \]

  \[
  = F(u) + (Lu - f, v) + (Lv, v)/2.
  \]

- The last term, $(Lv, v)/2$, is positive for $v \neq 0$, but it can be neglected for very small $v$. Thus,

  \[
  F(u + v) \geq F(u) \quad \forall v \ni H^1_0(\Omega) \Rightarrow \left( Lu, v \right) = (f, v) \quad \forall v \ni H^1_0(\Omega)
  \]

  \[
  \Rightarrow "Lu = f".
  \]
FE constructs

- Assume that $\Omega$ is the unit square.
- Consider an $mn \times n$ grid of square "cells".
- Continuous piecewise bilinear elements:
  \[ H^h \subset H^0(\Omega). \]
- Each $u^h$ in $H^h$ is determined by its node values. This is how we'll represent them!
- Within each square:
  \[ u^h = axy + bx + cy + d \]
  and
  \[ u^h \] is linear in a coordinate direction.
- If $u^h$ on one side of an element matches $u^h$ on the other at the nodes, then it matches on the common edge. So $u^h$ is continuous.

FE discretization

- Minimize $F(u^h) = (Lu^h, u^h)/2 - (f, u^h)$ over $u^h \in H^h$.
- Equivalent to solving
  \[ (Lu^h, v^h) = (f, v^h) \quad \forall \ v^h \in H^h. \]
- Basis: $\psi_{ij}^h$ is the element of $H^h$ that equals 1 @ node $ij$ & 0 elsewhere.
- Expansion: $u^h(x,y) = \sum_{ij} u_{ij}^h \psi_{ij}^h (x,y)$.
- Problem: What is $Lu^h = - u_{xx}^h - u_{yy}^h$??

Continuity

**Weak form**

- The Gauss Divergence Theorem & homogeneous boundary conditions yield
  \[ (Lu, v) = (- u_{xx} - u_{yy}, v) = (-\nabla \cdot \nabla u, v) = (\nabla u, \nabla v). \]
- Note:
  \[ (\nabla u, \nabla v) = \int_{\Omega} (u_{xx}v_x + u_{yy}v_y) \, dx \, dy. \]
- So the problem becomes
  \[ (\nabla u^h, \nabla v^h) = (f, v^h) \quad \forall \ v^h \in H^h \]
  or
  \[ \int_{\Omega} (u_{xx}^hv_x + u_{yy}^hv_y) \, dx \, dy = \int_{\Omega} f^h \, dx \, dy \quad \forall \ v^h \in H^h. \]
Towards the matrix equation

\[(\nabla u^h, \nabla v^h) = (f, v^h) \quad \forall v^h \in H^h\]

- Using \(u^h(x, y) = \sum_i u^h_i \varepsilon_{ij}^h (x, y)\) & choosing \(v^h = \varepsilon_{kj}^h\):
  \[\sum_i u^h_i (\nabla \varepsilon_{ij}^h, \nabla \varepsilon_{kj}^h) = (f, \varepsilon_{kj}^h) \quad \forall k, l.
\]
- Matrix terms \((\nabla \varepsilon_{ij}^h, \nabla \varepsilon_{kl}^h)\) are 0 when \(|i - k|\ or \(|j - l| > 1\).

We compute
\[(\nabla \varepsilon_{ij}^h, \nabla \varepsilon_{ij}^h) = 8/3 \quad \& \quad (\nabla \varepsilon_{ij}^h, \nabla \varepsilon_{ij}^h) = -1/3.
\]
- Assume \(f\) is bilinear:
  \[(f, \varepsilon_{ij}^h) = \int_\Omega f \varepsilon_{ij}^h \, d\Omega \approx h^2 f(x_{kl}).\]
  \[u^h = (u^h_i) \quad \& \quad f^h = (h^2 f(x_{kl}))\]
  actually, \(B^h f^h\) (mass matrix)

Some matrix properties
\[A^h_{ij} = \frac{1}{3} \begin{pmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{pmatrix}
\]
- Symmetric! \(ij\) "reaches" to \(i\&j\pm1\) as \(i\&j\pm1\) to \(ij\).
- Singular? \(A^h 1 = 0\) ?! Depends on boundaries!
  \[\text{Dirichlet boundary:} \quad A^h_{ij} = \frac{1}{3} \begin{pmatrix} -1 & -1 \\ 8 & -1 \\ -1 & -1 \end{pmatrix}.
\]
- Positive definite!
  Diagonally dominant (strictly so @ boundaries).

The matrix equation
\[A^h u^h = f^h\]

where
\[u^h = (u^h_i) \quad \& \quad f^h = (h^2 f(x_{kl}))\]

& the matrix is given by the stencil
\[A^h_{ij} = \frac{1}{3} \begin{pmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{pmatrix}, \text{ stiffness matrix}
\]

Abstract FE relaxation

focus on functions, not nodal vectors

- Relaxation involves "local" changes: \(u^h \leftarrow u^h - s \varepsilon^h\) for some scalar \(s\) & \(\varepsilon^h = \varepsilon_{ij}^h\).
  But how do we pick \(s\) ???

- Use FE principle of minimizing \(F(u^h - s \varepsilon^h)\) over \(s\):
  \[F(u^h - s \varepsilon^h) = (L(u^h - s \varepsilon^h), u^h - s \varepsilon^h)/2 - (f, u^h - s \varepsilon^h)\]
  \[= F(u^h) - s (L(u^h - f, \varepsilon^h) + (s^2/2) (L \varepsilon^h, \varepsilon^h)).\]

- Let \(s\) be the root of the derivative of this quadratic polynomial w.r.t. \(s\): \(s = (L(u^h - f, \varepsilon^h)/(L \varepsilon^h, \varepsilon^h) = r_{ij}/L_{ij,ij}^h\), so
  \[F(u^h - s \varepsilon^h) - F(u^h) = - (L(u^h - f, \varepsilon^h)^2/(2(L \varepsilon^h, \varepsilon^h)) = - r_{ij}^2(2L_{ij,ij}^h).
\]
  This is just Gauss-Seidel!
Abstract FE coarsening
again with focus on functions

- Coarsening involves a “global” change: \( u^h \leftarrow u^h + w^{2h} \) for some coarse-grid function \( w^{2h} \).
  But how do we pick \( w^{2h} \)???

- Use FE principle of minimizing \( F(u^h + w^{2h}) \) over \( w^{2h} \):
  \[
  F(u^h + w^{2h}) = (L(u^h + w^{2h}), u^h + w^{2h})/2 - (f, u^h + w^{2h})
  = F(u^h) + (Lu^h - f, w^{2h}) + (Lw^{2h}, w^{2h})/2.
  \]

Let \( w^{2h} \) be the root of the gradient of this quadratic functional w.r.t. \( w^{2h} \). This is tricky because you need to write the gradient as a function in the subspace \( \mathcal{H}^{2h} \). We go instead from abstract functions to nodal vectors...

Concrete FE coarsening (cont’d)
\( A^h u^h = f^h \)

- Solving this matrix equation is equivalent to minimizing
  \[
  F^h(v^h) \equiv (A^h v^h, v^h)/2 - (f^h, v^h)
  \]
  over \( v^h \in \mathcal{H}^h \). So how do we now correct \( v^h \)???

- We minimize \( F^h(v^h + I_{2h}^h v^{2h}) \) over \( v^{2h} \in \mathcal{H}^{2h} \):
  \[
  F(v^h + I_{2h}^h v^{2h})
  = (A^h (v^h + I_{2h}^h v^{2h}), v^h + I_{2h}^h v^{2h})/2 - (f^h, v^h + I_{2h}^h v^{2h})
  = F(v^h) + (A^{2h} v^h, v^{2h})/2 - (f^{2h}, v^{2h})
  \quad \text{variational conditions}
  \]
  \[
  \equiv F(v^h) + \underbrace{F^{2h}(v^{2h})}_{\text{coarsening}}.
  \]

Concrete FE coarsening: \( I_{2h}^h \)

- Adding nodal representations of \( v^h \) & \( v^{2h} \):
  \[
  v^{2h}(x,y) = \sum_{ij} v_{ij}^{2h} \phi_{ij}(x,y) \quad (\text{sum over } 2h \text{ indices})
  \]
  \[
  = \sum_{ij} u_{ij}^{h} \phi_{ij}(x,y) \quad (\text{sum over } h \text{ indices})
  \]

- We should be able to do this because \( v^{2h} \in \mathcal{H}^{2h} \subset \mathcal{H}^h \).

- Cell “2j+2,i+1”:
  \[
  v_{2j+2,i+1}^h = v_{2j+2,i}^{2h}
  v_{2j+2,i+1}^{2h} = (v_{2j+2,i}^{2h} + v_{2j+2,i+1}^{2h})/2
  \]
  \[
  v_{2j+2,i+1}^{2h} = (v_{2j+2,i+1}^{2h} + v_{2j+2,i}^{2h})/2
  v_{2j+2,i+1}^{2h} = (v_{2j+2,i+1}^{2h} + v_{2j+2,i}^{2h})/4
  \]

Outline

Chapters 1-5:
- \( \checkmark \) Model Problems
- \( \checkmark \) Basic Iterative Methods
  - Convergence tests
  - Analysis
- \( \checkmark \) Elements of Multigrid
  - Relaxation
  - Coarsening
- \( \checkmark \) Implementation
  - Complexity
  - Diagnostics
- \( \checkmark \) Some Theory
  - Spectral vs. algebraic

Chapters 6-10:
- \( \checkmark \) Nonlinear Problems
  - Full approximation scheme
- \( \checkmark \) Selected Applications
  - Variable coefficients
- \( \checkmark \) Algebraic Multigrid (AMG)
  - Matrix coarsening
- \( \checkmark \) Multilevel Adaptive Methods
  - FAC
- \( \checkmark \) Finite Elements
  - Variational methodology

Homework Due Projects Next Week
Multigrid rules!

We conclude with a few observations:

- We have barely scratched the surface of the myriad ways that multigrid has been, & can be, employed.

- With diligence & care, multigrid can be made to handle many types of complications in a robust, efficient manner.

- Further extensions to multigrid methodology are being sought by many people working on many different problems.

Multigrid/multilevel/multiscale
an important methodology

- Multigrid has proved successful on a wide variety of problems, especially elliptic PDEs, but has also found application in parabolic & hyperbolic PDEs, integral equations, evolution problems, geodesic problems, ...

- Multigrid can be optimal, often $O(N)$.

- Multigrid can be robust in a practical sense.

- Multigrid is of great interest because it is one of the very few scalable algorithms, & it can be parallelized readily & efficiently!