2013
APPM 6640
Multigrid Methods

Lectures: Mondays & Wednesdays 10-10^{50}am ECCR 1B51
Computing Lab: Mondays 12-12^{50} ECCR 143
Office: Mondays 11-11^{50} & Wednesdays 9-9^{50} ECCR 257

Steve McCormick
stevem@colorado.edu

Toby Jones
Tobias.Jones@colorado.edu

Chris Leibs
chris.leibs@gmail.com

303-492-0662 (email is better!)
http://amath.colorado.edu/faculty/stevem

APPM 6640
Syllabus

- Coursework
  - No exams
  - Homework exercises & computing assignments (Lab)
  - Team project: identify & pursue a target application
- Philosophy (the course is not just about multigrid!!!)
  - Guiding rule: understanding trumps knowledge
  - Look for fundamentals & basic principles to guide you
  - Solutions are often straightforward consequences
  - Think about the scientific method in general
  - Understand by concrete examples & experience
  - Know the whole picture
  - Ask questions & interject comments
  - Make sure I explain what you need
- Text
  - A Multigrid Tutorial, 2nd edition, 2nd printing
  - Supplemental material as needed

Sources

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

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

Copper Mountain Conference March 17-22, 2013
http://grandmaster.colorado.edu/~copper
We plan to support your attendance there.

Course web site:
http://grandmaster.colorado.edu/appm6640/
These slides are there.

Homework exercises

Due one week after the relevant chapter is covered in class.

1:	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

You will learn about this in the lab &/or from the course website.

Team project

main objective: understand applied math research

- Group into teams of possible common interest.
- Meet in lab as time permits later in the semester.
- Identify an application area of interest.
- Identify a problem in that area & learn about it.
- Learn current methods & their limitations.
- Get experience with these methods on typical cases.
- Try cases where these methods (begin to) fail.
- Brainstorm a better (multilevel?) method.
- Implement & test your idea.
- Inform the rest of the class along the way.

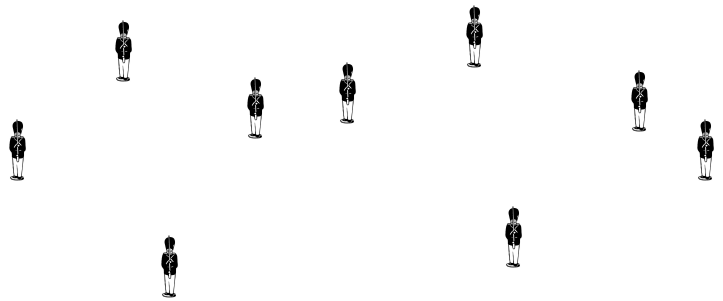
Prolog: Multigrid in Action

The following soldier slides were created by

Irad Yavneh
Department of Computer Science
Technion – Israel Institute of Technology
irad@cs.technion.ac.il

Basic Concepts: Local vs. Global processing.

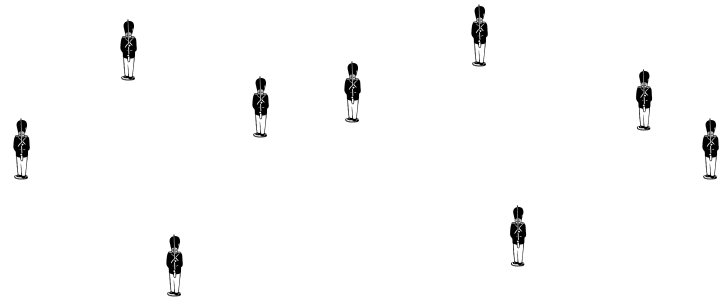
- Imagine a large number of soldiers who need to be arranged in a straight line and at equal distances from each other.
- The two soldiers at the ends of the line are fixed. Suppose we number the soldiers 0 to N , and that the length of the entire line is L.



Initial Position

Final Position

Global processing. Let soldier number j stand on the line connecting soldier 0 to soldier N at a distance jL/N from soldier number 0.



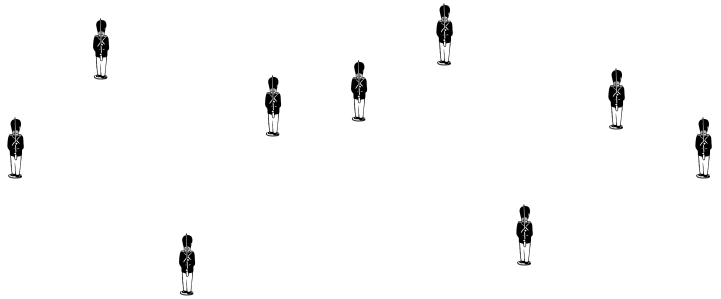
This method solves the problem directly, but requires substantial sophistication: recognition of the extreme soldiers and some pretty fancy arithmetic.

Local processing (iterative method). Suppose that the initial position of inner soldier j is x_j. Then if every j moves all at once to the point midway between the initial locations of neighboring soldiers, j-1 & j+1, we get

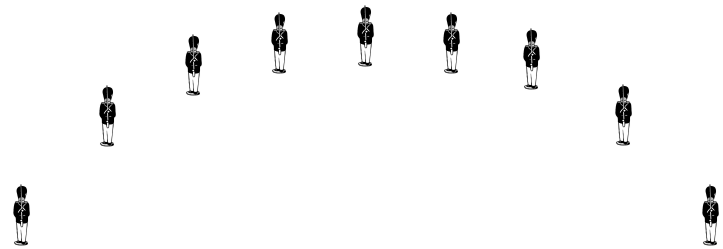
$$x_j \leftarrow (x_{j-1} + x_{j+1})/2.$$

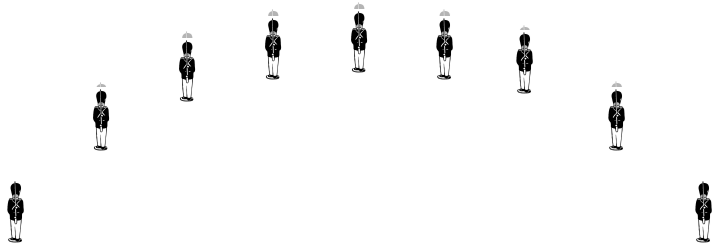
(Assume for simplicity that the soldiers have guides to make sure they're evenly spaced, so they only have to get in a straight line. Thus, x_j is their signed distance from that line.)

- This is an iterative process.
- Each step brings us closer to the solution (convergence).
- The arithmetic is trivial.
- The process is local.

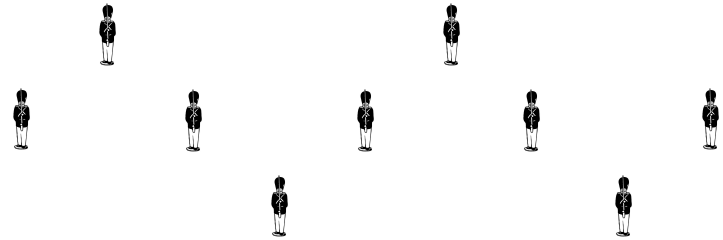


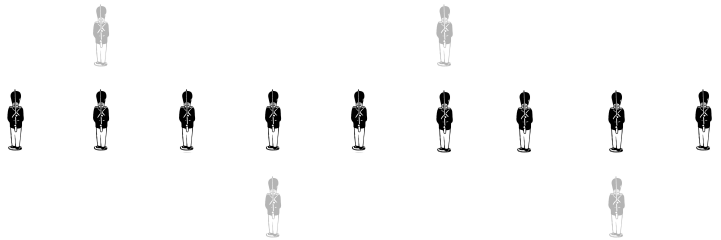
A step in the right direction



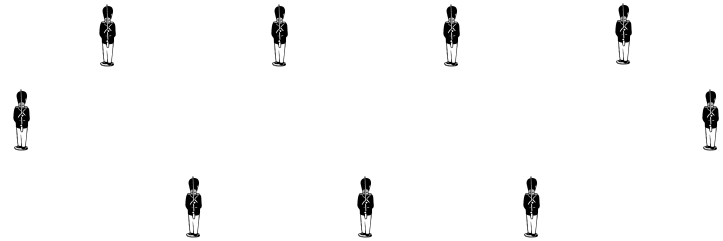


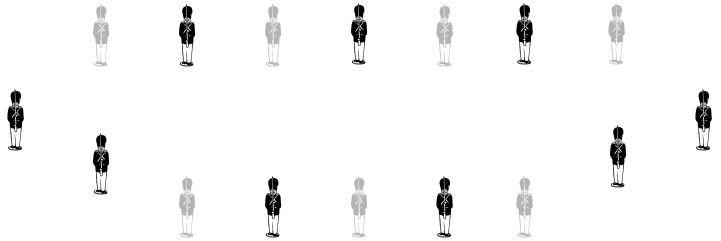
Slow convergence



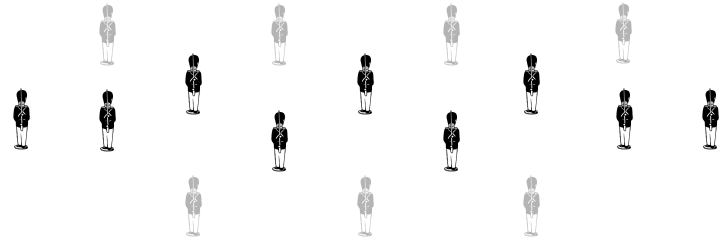


Fast convergence





Slow convergence



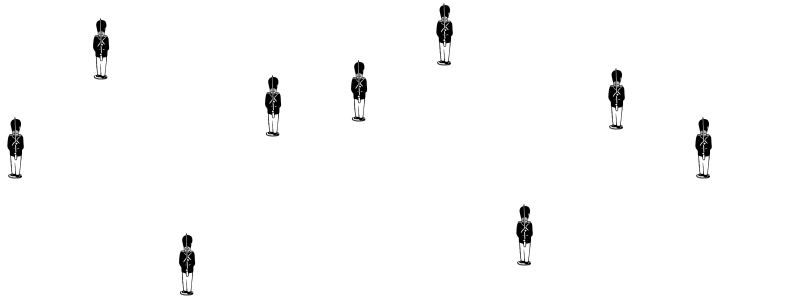
Local solution: damping

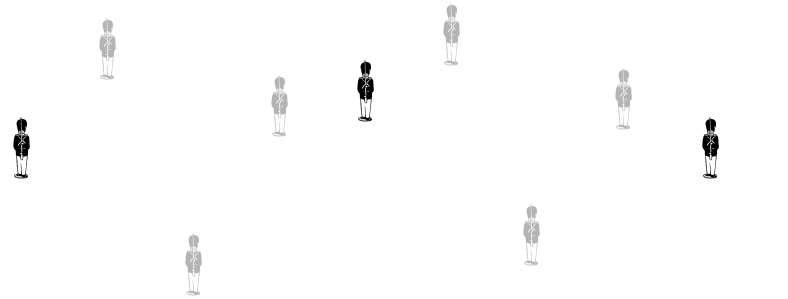
Local solution: damping

Local solution: damping

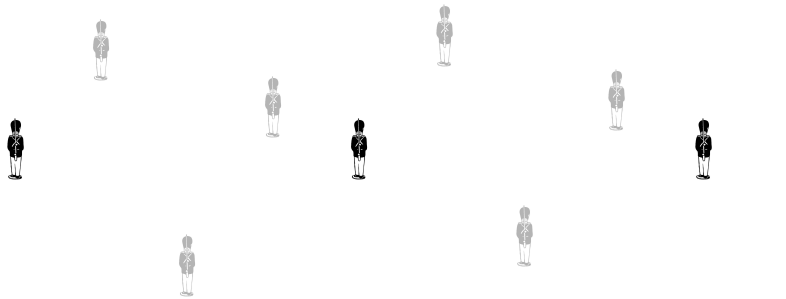
Local solution: damping

The multiscale idea: Employ the local processing with simple arithmetic. But do this on all the different scales.

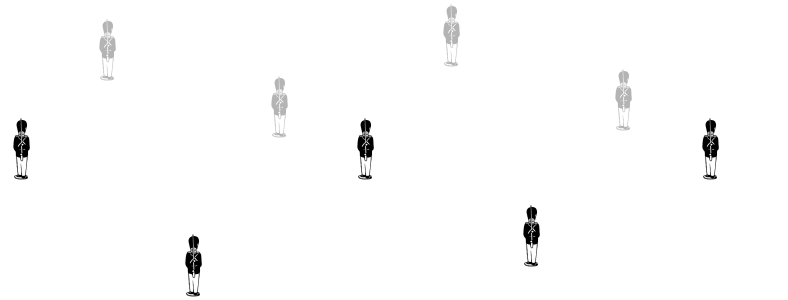




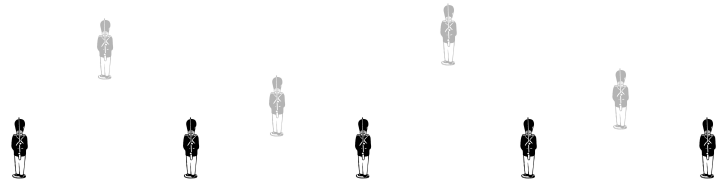
Large scale



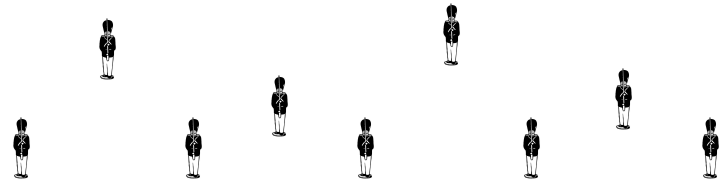
Large scale



Intermediate scale



Intermediate scale



Small scale

How much work do we save?

Jacobi needs about N^2 iterations & $N^2 \times N = N^3$ ops to improve accuracy by an order of magnitude.

Brandt solves the problem in only about N operations.

Example: for N = 1000, MG needs about

1,000 operations

instead of about

1,000,000,000 operations ! ! !

How important is computational efficiency?

Suppose we have 3 different algorithms for a given problem, with different computational complexities for input size N :

Algorithm 1: 10^6 N ops

Algorithm 2: 10^3 N^2 ops

Algorithm 3: N^3 ops

Suppose N is such that algorithm 1 requires one second.

How long do the others require?

Computer Speed (ops/sec)	N	Algorithm 1 O(N)	Algorithm 2 $O(N^2)$	Algorithm 3 $O(N^3)$
1M (~10^6) (1980's)	1	1 sec	0.001 sec	0.000001 sec
1G (~10^9) (1990's)	1K	1 sec	1 sec	1 sec
1T (~10^{12}) (2000's)	1M	1 sec	17 min	12 days
1P (~10^{15}) (2010's)	1G	1 sec	12 days	31,710 years

Stronger computers ⇒ more gain!

The catch

In less trivial problems, we can't construct appropriate equations on the large scales without first propagating information from the small scales.

Skill in developing efficient multigrid is needed for:

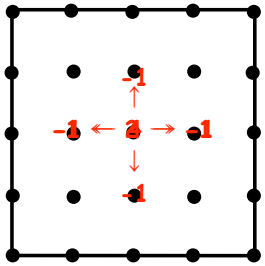
1. Choosing a good local iteration.
2. Choosing appropriate coarse-scale variables.
3. Choosing inter-scale transfer operators.
4. Constructing coarse-scale approximations to fine scales.

What about two dimensions?

Put points midway between horizontal (or vertical) neighbors.

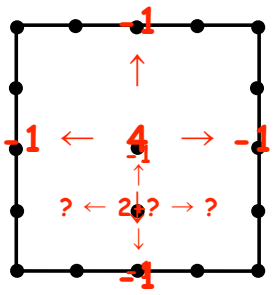
This is just imposing $x_i = (x_{i-1} + x_{i+1})/2$ on each row j or

$2x_i - x_{i-1} - x_{i+1} = 0$.



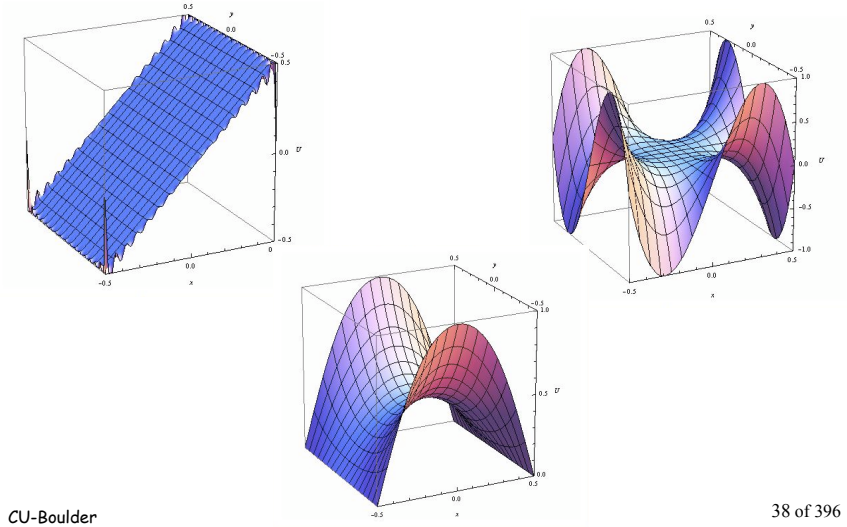
The hitch is that this is not a common physical problem. More common is to ask that some physical quantity at each point be an average of its FOUR neighbors (Poisson).

Poisson

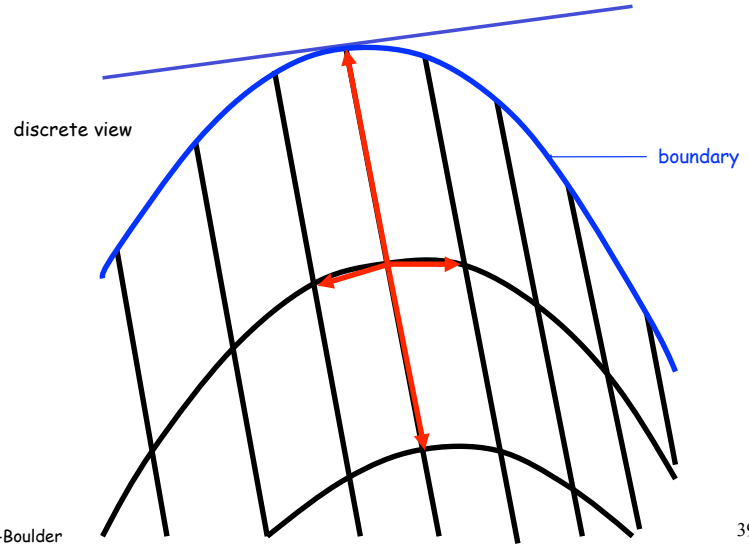


Poisson: minimal surface

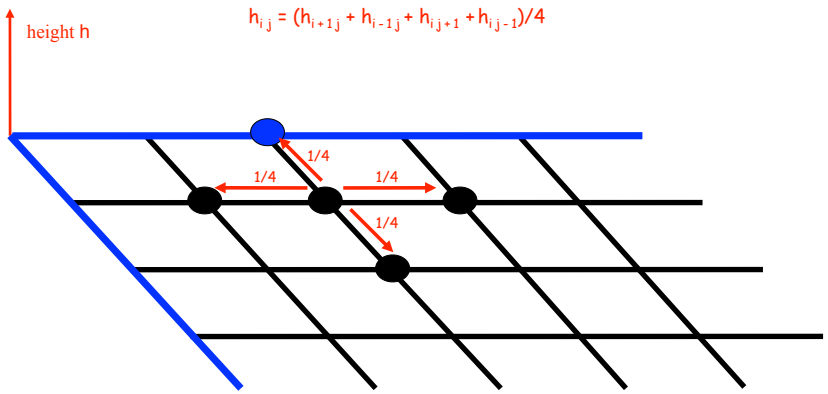
given boundary values, minimize surface area

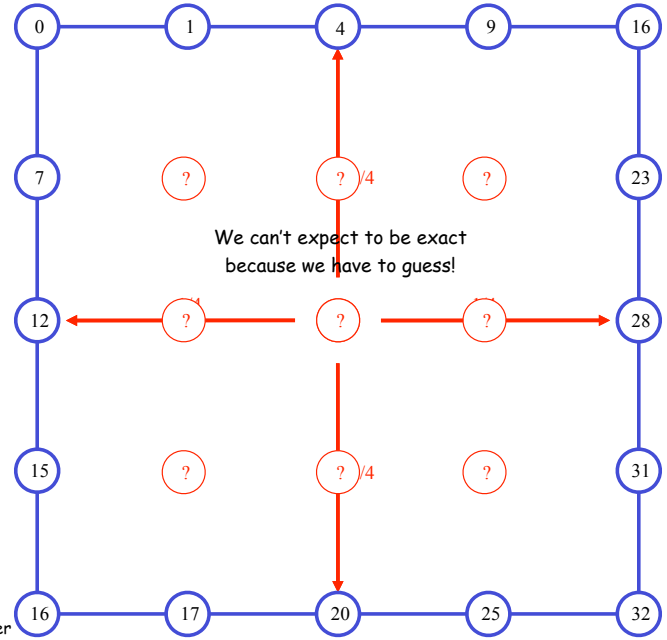


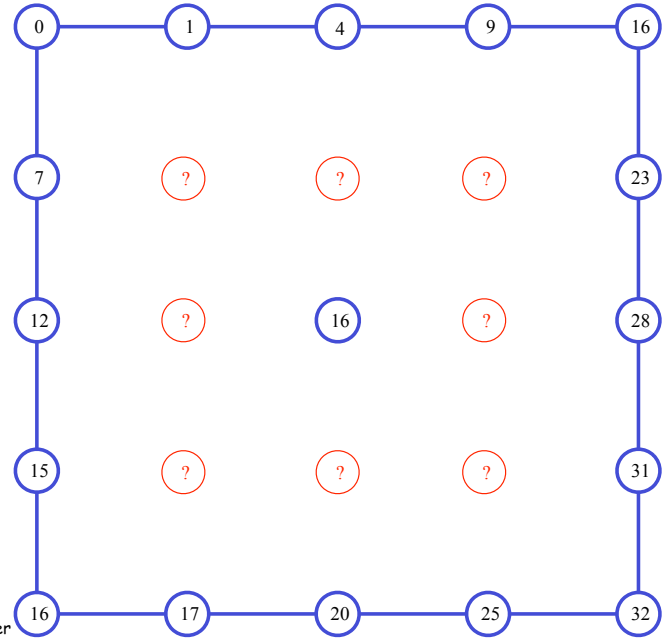
Physical principle

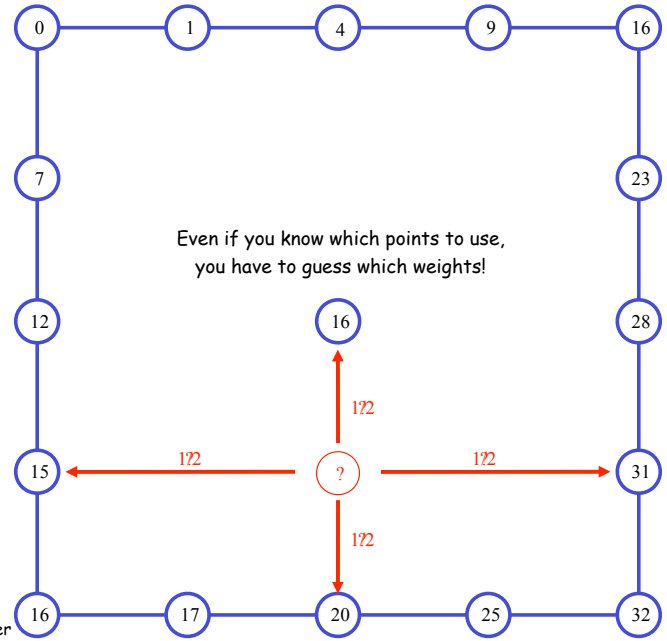


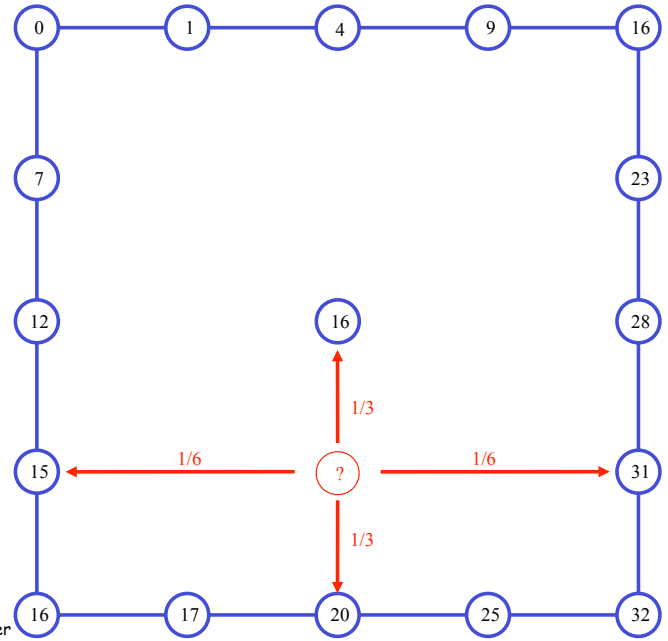
Local relationship

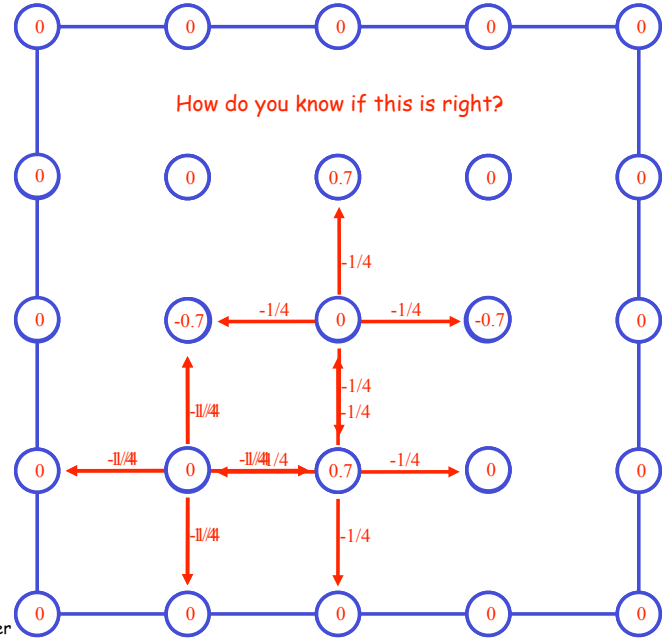


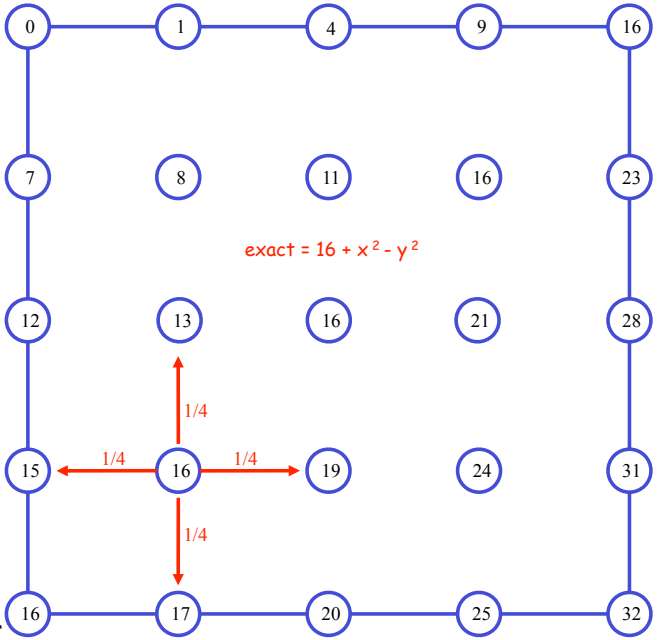

0
1
4
9
16
7
?
?
/4
?
23
We can't expect to be exact
because we have to guess!
12
?
?
?
28
15
?
?
/4
?
31
16
17
20
25
32

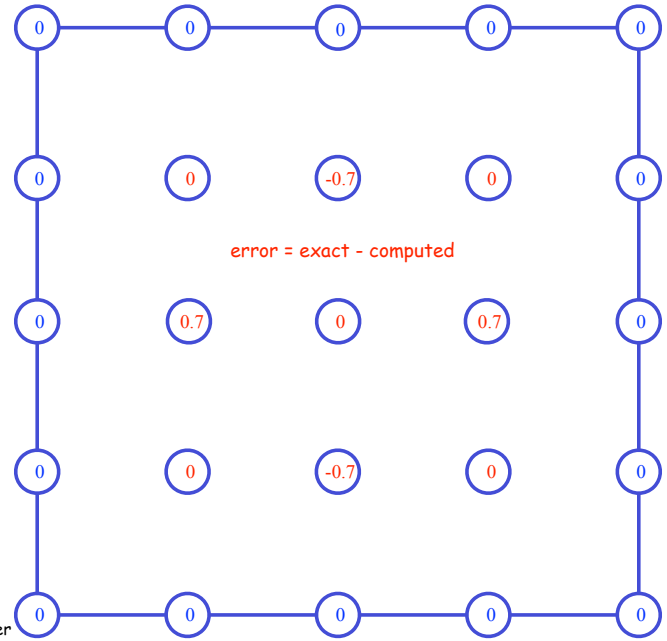

0
1
4
9
16
7
?
?
?
23
12
?
16
?
28
15
?
?
?
31
16
17
20
25
32

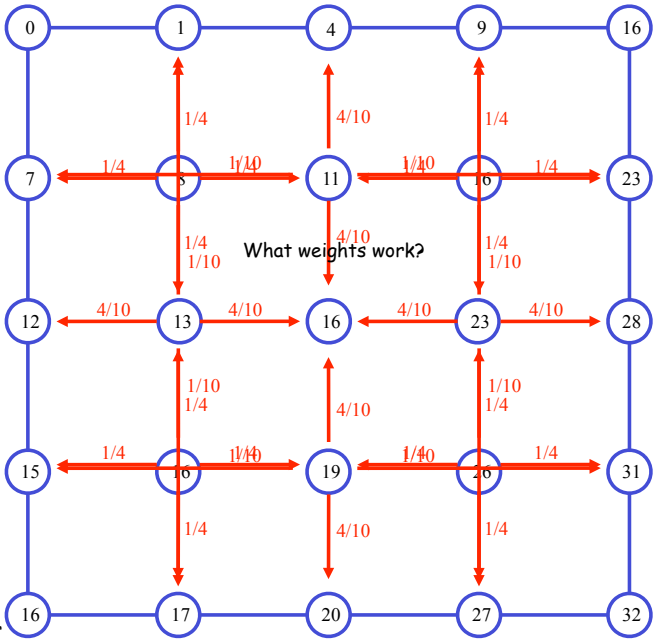

0
1
4
9
16
7
23
Even if you know which points to use,
you have to guess which weights!
12
16
28
1?2
1?2
?
1?2
15
31
1?2
16
17
20
25
32


0
1
4
9
16
7
23
12
16
28
1/3
1/6
?
1/6
15
31
1/3
16
17
20
25
32


How do you know if this is right?
0 0 0 0 0
0 0 0.7 0 0
0 -0.7 -1/4 0 -1/4 -0.7 0
0 -1/4 -1/4 0
0 -1/4 -1/4 0 -1/4 0 0
0 -1/4 -1/4 0
0 0 0 0 0


0 1 4 9 16
7 8 11 16 23
exact = 16 + x² - y²
12 13 16 21 28
1/4
15 1/4 16 1/4 19 24 31
1/4
16 17 20 25 32


0 0 0 0 0
0 0 -0.7 0 0
error = exact - computed
0 0.7 0 0.7 0
0 0 -0.7 0 0
0 0 0 0 0


0 1 4 9 16
1/4 4/10 1/4
7 1/4 8 1/10 11 1/10 16 1/4 23
1/4 4/10 1/4
1/10 What weights work? 1/10
12 4/10 13 4/10 16 4/10 23 4/10 28
1/10 1/10
1/4 4/10 1/4
15 1/4 16 1/10 19 1/10 26 1/4 31
1/4 4/10 1/4
16 17 20 27 32

A Multigrid Tutorial

2nd Edition, 2nd Printing

By

William L. Briggs

CU-Denver

Van Emden Henson

LLNL

Steve McCormick

CU-Boulder

Outline

by chapter

Suggested reading

CHECK THE MG LIBRARY & MGNET REPOSITORY

- A. Brandt, "Multi-level Adaptive Solutions to Boundary Value Problems," Math Comp., 31, 1977, pp 333-390.
- A. Brandt, "Multigrid techniques: 1984 guide with applications to computational fluid dynamics," GMD, 1984.
- W. Hackbusch & U. Trottenberg, "Multigrid Methods", Springer-Verlag, 1982.
- S. McCormick, ed., "Multigrid Methods," SIAM Frontiers in Applied Math. III, 1987.
- U. Trottenberg, C. Oosterlee, & A. Schüller, "Multigrid," Academic Press, 2000.
- P. Wesseling, "An Introduction to Multigrid Methods," Wylie, 1992.

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 chromodynamics.
- Quadrature & generalized FFTs.
- Integral equations.
-
-
-

Everyone uses multilevel methods

- Multigrid, multilevel, multiscale, multiphysics, …
  Use local "governing rules" at the finest resolution to resolve details of the state of the system, but use coarser resolution to resolve larger scales. Continual feedback is essential because improving one scale impacts other scales.
- Common uses
  Sight, art, team sports, politics, society, thinking, scientific research, …

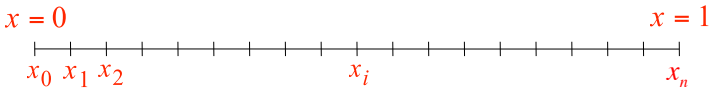
1. Model problems

- 1-D boundary value problem:

$$-u''(x) + \sigma u(x) = f(x) \qquad 0 < x < 1, \qquad \sigma \geq 0$$
$$u(0) = u(1) = 0$$

- Grid:

$$h = \frac{1}{n}, \quad x_i = ih, \qquad i = 0,1,\ldots,n$$



- Let $v_i \approx u(x_i)$ & $f_i \approx f(x_i)$ for i = 0,1,…,n .

This discretizes the variables, but what about the equations?

Approximate u''(x) via Taylor series

$O(h^2)$ means a quantity bounded in norm by Ch^2 for some constant C.

- Approximate 2nd derivative using Taylor series:

$$u(x_{i+1}) = u(x_i) + h\underbrace{u'(x_i)}_{0} + \frac{h^2}{2!}u''(x_i) + \frac{h^3}{3!}\underbrace{u'''(x_i)}_{0} + O(h^4)$$
$$+$$
$$u(x_{i-1}) = u(x_i) - h\underbrace{u'(x_i)}_{0} + \frac{h^2}{2!}u''(x_i) - \frac{h^3}{3!}\underbrace{u'''(x_i)}_{0} + O(h^4)$$

- Summing & solving:

$$u''(x_i) = \frac{u(x_{i+1}) - 2u(x_i) + u(x_{i-1})}{h^2} + O(h^2)$$

Approximate equation via finite differences

Approximate the BVP

$$-u''(x) + \sigma u(x) = f(x) \qquad 0 < x < 1, \qquad \sigma \geq 0$$
$$u(0) = u(1) = 0$$

by a finite difference scheme:

$$\frac{-v_{i-1} + 2v_i - v_{i+1}}{h^2} = -\frac{\frac{v_{i+1} - v_i}{h} - \frac{v_i - v_{i-1}}{h}}{h}$$
$$v_0 = v_n = 0$$

Discrete model problem

Letting $v = (v_1, v_2, \ldots, v_{n-1})^T$ & $f = (f_1, f_2, \ldots, f_{n-1})^T$,

we obtain the matrix equation

$$Av = f,$$

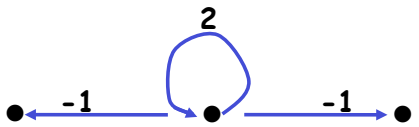
where A is (n-1)x(n-1), symmetric, positive definite,

$$A = \frac{1}{h^2}\begin{pmatrix} 2+\sigma h^2 & -1 & & & & \\ -1 & 2+\sigma h^2 & -1 & & & \\ & -1 & 2+\sigma h^2 & \ddots & & \\ & & \ddots & \ddots & -1 & \\ & & & -1 & 2+\sigma h^2 & -1 \\ & & & & -1 & 2+\sigma h^2 \end{pmatrix}, \quad v = \begin{pmatrix} v_1 \\ v_2 \\ v_3 \\ \vdots \\ v_{n-2} \\ v_{n-1} \end{pmatrix}, \quad f = \begin{pmatrix} f_1 \\ f_2 \\ f_3 \\ \vdots \\ f_{n-2} \\ f_{n-1} \end{pmatrix}.$$

Stencil notation

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

dropping h^{-2} & σ 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 $\qquad \sigma \geq 0$

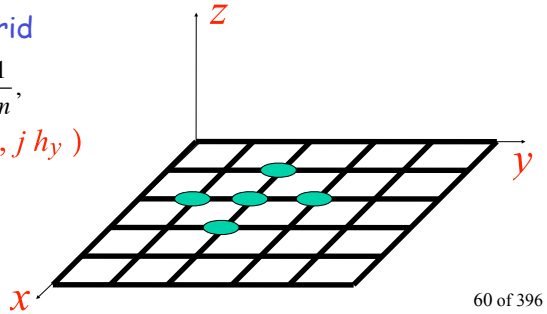
- Consider the grid

$$h_x = \frac{1}{l}, \quad h_y = \frac{1}{m},$$

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

$$0 \leq i \leq l$$

$$0 \leq j \leq m$$



Discretizing the 2-D problem

- Let $v_{ij} \approx u(x_i, y_j)$ & $f_{ij} \approx f(x_i, y_j)$. 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,…,l-1 & j =1,2, …,m-1 :

$$\frac{-v_{i-1,j} + 2v_{ij} - v_{i+1,j}}{h_x^2} + \frac{-v_{i,j-1} + 2v_{ij} - v_{i,j+1}}{h_y^2} + \sigma v_{ij} = f_{ij}$$

$$v_{i,j} = 0: \quad i = 0, \quad i = l, \quad j = 0, \text{ or } j = m$$

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

$$v = (v_{1,1}, v_{1,2}, \ldots, v_{1,m-1}, v_{2,1}, v_{2,2}, \ldots, v_{2,m-1}, \ldots, v_{l-1,1}, v_{l-1,2}, \ldots, v_{l-1,m-1})^T$$

Resulting linear system

We obtain a block-tridiagonal system Av = f :

$$\begin{pmatrix} A_1 & -I_x & & & \\ -\frac{2}{h_y^2} & A_2 & -I_x & & \\ & -I_x & A_3 & \ddots & \\ & & \ddots & \ddots & -I_x \\ & & & -I_x & A_{l-1} \end{pmatrix} \begin{pmatrix} v_1 \\ v_2 \\ v_3 \\ \vdots \\ v_{l-2} \\ v_{l-1} \end{pmatrix} = \begin{pmatrix} f_1 \\ f_2 \\ f_3 \\ \vdots \\ f_{l-2} \\ f_{l-1} \end{pmatrix}.$$

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

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

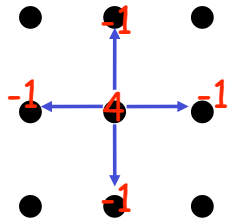
Stencils

preferred for grid issues

Stencils are much better for showing the grid picture:

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

dropping the mesh sizes & σ



Stencils show local relationships--grid point interactions.

Inhomogeneous boundary conditions

superposition

- Consider a boundary-value problem on domain Ω with nonzero data, g, on the boundary, ∂Ω :

$$Lu = f \quad \text{on } \Omega, \qquad Mu = g \quad \text{on } \partial\Omega.$$

Dirichlet: M = I

- Find a suitable w satisfying the boundary condition:

$$Mw = g \quad \text{on } \partial\Omega.$$

- Now just find z to correct w so that w + z = u, that is, z = u - w:

$$Lz = f - Lw \equiv \hat{f} \quad \text{on } \Omega, \qquad Mz = g - Mw = 0 \quad \text{on } \partial\Omega.$$

- Message: Don't look for u. Instead, look for w so that Mw = g on ∂Ω & then look for z = u - w such that Lz = f-Lw on Ω & Mz = 0 on ∂Ω.
- In the discrete Dirichlet case, set w^h to g^h on ∂Ω and 0 inside Ω.
- So we consider only the homogeneous case from now on.

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!

2. Basic iterative methods

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

- Two important measures:

  The Error: $e = u - v$ with norms
  $||e||_\infty = \max |e_i|$ & $||e||_2 = (\Sigma e_i^2)^{1/2}$.

  The Residual: $r = f - Av$ with
  $||r||_\infty$ & $||r||_2$.

What does $||r||$ measure???
Why have both r & e???

Residual correction

- Note: $e = u - v \Rightarrow Ae = A(u - v) = f - Av = r$.

- Residual Equation:
  $Ae = r$.

  What does this do for us?

- Residual Correction:
  $u = v + e$.

Solve $Au = f$ with guess v
or $Ae = r$ with guess 0.

Relaxation

- Consider the 1-D model problem

$$-u_{i-1} + 2u_i - u_{i+1} = h^2 f_i \qquad 1 \le i \le n\text{-}1 \qquad u_0 = u_n = 0$$

- Jacobi (simultaneous displacement): Solve the i^{th} equation for v_i holding all other variables fixed:

$$v_i^{(new)} = \frac{1}{2}(v_{i-1}^{(old)} + v_{i+1}^{(old)} + h^2 f_i) \qquad 1 \le i \le n\text{-}1$$

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

$$(D - L - U)\,u = f$$
$$D\,u = (L + U)\,u + f$$
$$u = D^{-1}(L + U)\,u + D^{-1}f$$

- Let $R_J = D^{-1}(L + U)$. $R_J = D^{-1}(D-A) = I - D^{-1}A$

"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,

$$u = D^{-1}(L + U)u + D^{-1}f$$
$$u = R_J u + D^{-1}f$$

the iteration is

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

subtracting,

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

or

$$u - v^{(new)} = R_J u - R_J v^{(old)}$$

hence,

Error propagation! $\boxed{e^{(new)} = R_J e^{(old)}}$ $R_J = I - D^{-1}A$

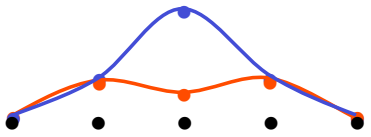
A picture

1D Poisson

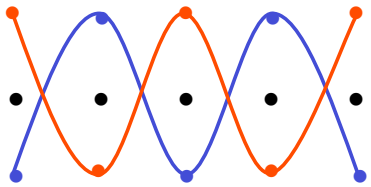
$R_J = D^{-1}(L + U) = [\ \frac{1}{2}\ 0\ \frac{1}{2}\ ]$

so Jacobi is an error averaging process:

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



But…



Another matrix look at Jacobi

$v^{(new)} \leftarrow D^{-1}(L + U)\, v^{(old)} + D^{-1} f$ (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: $e^{(new)} = e^{(old)} - D^{-1} Ae^{(old)} = (I-D^{-1}A)e^{(old)}$
- 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 $0 < \omega < 2/\rho(D^{-1}A)$
- Gauss-Seidel: $u = u - (D - L)^{-1}(Au - f)$

Note that $R_\omega = I - \omega D^{-1}A$ 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!!!

Weighted Jacobi

safer changes: $0 < \omega < 2/\rho(D^{-1}A) \approx 1$

- 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

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

$$= R_\omega v^{(old)} + \omega h^2 D^{-1} f$$

- Note that

$$R_\omega = [(1-\omega)\, I + \omega\, R_J]$$

- 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_i & update immediately.
- Equivalently: set each component of r to zero in turn.
- Component form: for i = 1, 2, …, n-1, set

$$v_i \leftarrow \frac{1}{2}(v_{i-1} + v_{i+1} + h^2 f_i)$$

- Matrix form:

$$A = (D - L - U)$$

$$(D-L)\, u = Uu + f$$

$$u = (D-L)^{-1} Uu + (D-L)^{-1} f$$

- Let $R_G = (D-L)^{-1}U$ $R_G = (D-L)^{-1}(D-L-A) = I - (D-L)^{-1}A$
- 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

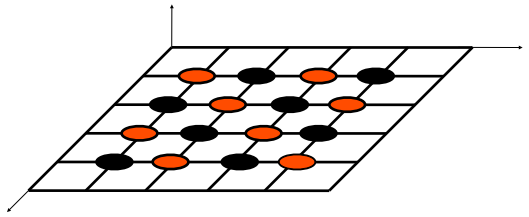
- Update the EVEN points:

$$v_{2i} \leftarrow \frac{1}{2}(v_{2i-1} + v_{2i+1} + h^2 f_{2i})$$

- Update the ODD points:

$$v_{2i+1} \leftarrow \frac{1}{2}(v_{2i} + v_{2i+2} + h^2 f_{2i+1})$$

- 2-D:



Test?

$Au = f$

Need to know how we're doing!!!

- What f ?

$Au = 0$

- What v ?

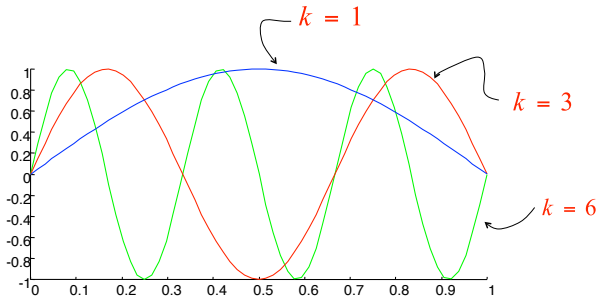
v = rand

Numerical experiments

- Solve $Au = 0$, $-u_{i-1} + 2u_i - u_{i+1} = 0$
- Use Fourier modes as initial iterates, with n = 64:

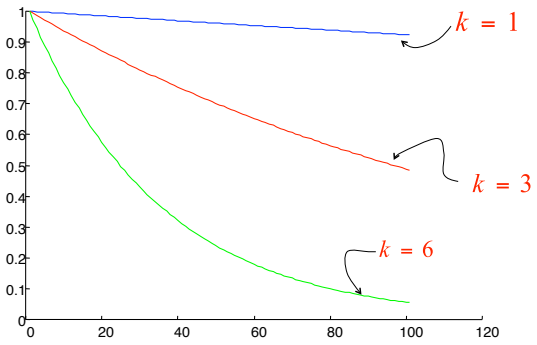
$v_k = \sin(k\pi x_i)$, $x_i = i/n$, $1 \le i \le n-1$, $1 \le k \le n-1$

component mode



Convergence factors differ for different error components

Error, $||e||_\infty$, in weighted ($\omega = 2/3$) Jacobi on Au = 0 using initial guesses v_1, v_3, & v_6 & n = 64:



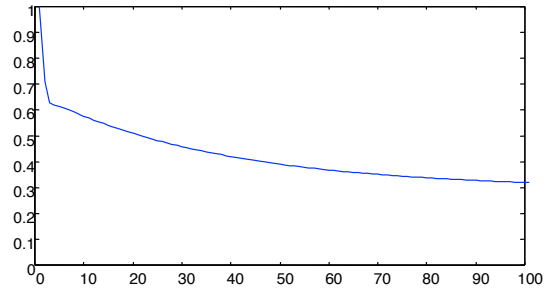
Stalling convergence

relaxation shoots itself in the foot

- Weighted ($\omega = 2/3$) Jacobi on 1-D problem & n = 64.
- Initial guess:

$$v = \frac{1}{3}\left[\left(\sin\left(\frac{1i\pi}{n}\right)\right) + \left(\sin\left(\frac{6i\pi}{n}\right)\right) + \left(\sin\left(\frac{32i\pi}{n}\right)\right)\right]$$

- Error, $||e||_\infty$, plotted against iteration number:



Analysis of stationary linear iteration

- The iteration is $v^{(new)} = Rv^{(old)} + g$.
- Exact solution doesn't change: $u = Ru + g$.
- Subtracting: $e^{(new)} = Re^{(old)}$.
- Let $e^{(0)}$ be the initial error & $e^{(i)}$ be the error after the i^{th} iteration. After n iterations, we have

$$e^{(m)} = Re^{(m-1)} = R^2e^{(m-2)} = \ldots = R^me^{(0)}.$$

We can deal with 2^4, but $\begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix}^4$???

What if $\begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix}^4 e = 2^4 e$???

Review of eigenvectors & eigenvalues

Bold for vectors here temporarily

- The real number λ is an eigenvalue of matrix B & $\mathbf{w} \neq 0$ is its associated eigenvector if $B\mathbf{w} = \lambda\mathbf{w}$.
- 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 NxN matrix, then they form a basis: for any $\mathbf{v}$, there exist unique scalars v_k such that

$$\mathbf{v} = \sum_{k=1}^{N} v_k \mathbf{w}_k.$$

- Propagation:

$$B^m \mathbf{v} = \sum_{k=1}^{N} \lambda^m v_k \mathbf{w}_k.$$

Why is an eigenvector useful???

"Fundamental Theorem of Iteration"

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

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

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

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

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

But ρ 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 ($\mathbf{w}_k$ orthonormal) & nonnegative definite ($\lambda \geq 0$)

- $RQ(\mathbf{v}) \leq \rho(A)$: $\mathbf{v} = \Sigma v_k \mathbf{w}_k$

$$RQ(\mathbf{v}) = \frac{<A\mathbf{v},\mathbf{v}>}{<\mathbf{v},\mathbf{v}>} = \frac{<A\sum v_k \mathbf{w}_k, \sum v_k \mathbf{w}_k>}{<\sum v_k \mathbf{w}_k, \sum v_k \mathbf{w}_k>}$$

$$= \frac{<\sum \lambda_k v_k \mathbf{w}_k, \sum v_k \mathbf{w}_k>}{<\sum v_k \mathbf{w}_k, \sum v_k \mathbf{w}_k>} = \frac{\sum \lambda_k v_k^2}{\sum v_k^2} \leq \lambda_N = \rho(A)$$

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

$$RQ(\mathbf{w}_N) = \frac{<A\mathbf{w}_N, \mathbf{w}_N>}{<\mathbf{w}_N, \mathbf{w}_N>} = \frac{<\lambda_N \mathbf{w}_N, \mathbf{w}_N>}{<\mathbf{w}_N, \mathbf{w}_N>} = \lambda_N = \rho(A)$$

Euclidean norm vs. spectral radius

use RQ

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

$$||R||_2^2 = \sup_{e\neq 0} ||Re||_2^2 / ||e||_2^2$$
$$= \sup_{e\neq 0} \langle Re, Re\rangle / \langle e, e\rangle$$
$$= \sup_{e\neq 0} \langle R^TRe, e\rangle / \langle e, e\rangle = \rho(R^TR)$$

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

- $||A||_2 = \rho^{1/2}(A^2) = \rho(A)$ for symmetric A!!!

$\rho(R)$ vs. $||R||_2$

$\rho(R) = \sup |\lambda(R)|$. — Definition? — $||R||_2 = \sup_{e\neq 0} ||Re||_2 / ||e||_2 = \rho^{1/2}(R^TR)$.

Norm independent. — Norm? — Depends on $||\cdot||_2$.

Asymptotic: $\rho(R) < 1 \ll ||R||_2$ means that it will converge someday. — Error bound? — $||e^{(m+1)}||_2 \le ||R||_2 ||e^{(m)}||_2$: worst case! Probably pessimistic initially, but sharp sooner or later.

$\rho(R) = \inf ||R||$ over all norms.

$\rho(R) = ||R||_2$ for symmetric R.

Example: $R = \begin{pmatrix} 0 & K \\ 0 & 0 \end{pmatrix}$, $K \gg 0$ large

$\rho(R) = 0$ but $||R||_2 = \rho^{1/2}(R^TR) = \rho^{1/2}\begin{pmatrix} K^2 & 0 \\ 0 & 0 \end{pmatrix} = K$!

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

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 (λ=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 factor & rate

- How many iterations are enough to guarantee reduction of the initial error by 10^{-d}?

$$\frac{\|e^{(m)}\|}{\|e^{(0)}\|} \le \|R^m\| \le \|R\|^m \sim 10^{-d}.$$

- So, including the asymptotic estimate, we have

$$m \sim \frac{d}{-\log_{10}\|R\|} \text{ or } \frac{d}{-\log_{10}\rho(R)}.$$

- Convergence factor = $||R||$ or $\rho(R)$ error reduction/iterate.

- Convergence rate = $-\log_{10}(||R||)$ or $-\log_{10}(\rho(R))$ digits/iterate.

Convergence analysis: Weighted Jacobi

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

$$R_\omega = I - \frac{\omega}{2}\begin{pmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & & \\ & -1 & 2 & -1 & \\ & & \ddots & \ddots & \ddots \\ & & & -1 & 2 \end{pmatrix}$$ 1-D

For our 1-D model, the eigenvectors of weighted Jacobi R_ω & the eigenvectors of A are the same!

Why??? Special!!!

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

Remember that A is without h^2 here!

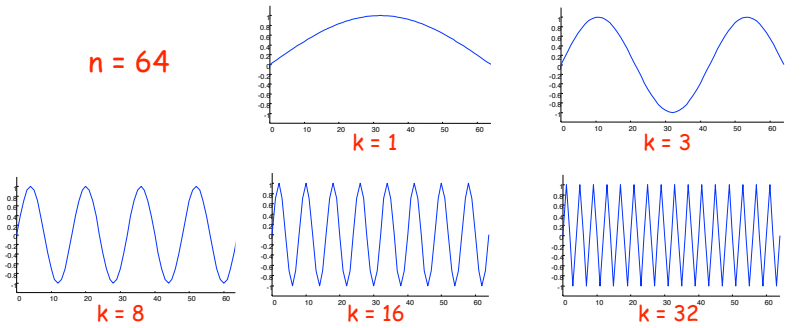
Eigenpairs of (scaled) A

The eigenvectors of A are (discrete) Fourier modes!

$$\lambda_k(A) = 4\sin^2\left(\frac{k\pi}{2n}\right), \qquad w_{k,i} = \sin\left(\frac{ik\pi}{n}\right)$$

[-1 2 -1]
$\lambda_{n-1} \cong 4$
$\lambda_1 \cong \pi^2 h^2$

n = 64



Eigenvectors of R_ω = eigenvectors of A

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

- Expand the initial error in terms of the eigenvectors:

$$e^{(0)} = \sum_{k=1}^{n} c_k w_k$$ ← drop bold for vectors

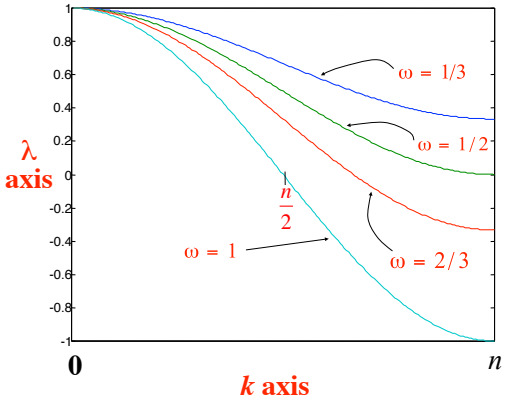
- After m iterations:

$$R^m e^{(0)} = \sum_{k=1}^{n} c_k \lambda_k^m w_k$$

- The k^{th} error mode is reduced by $\lambda_k(R_\omega)$ each iteration.

Relaxation suppresses eigenmodes unevenly

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



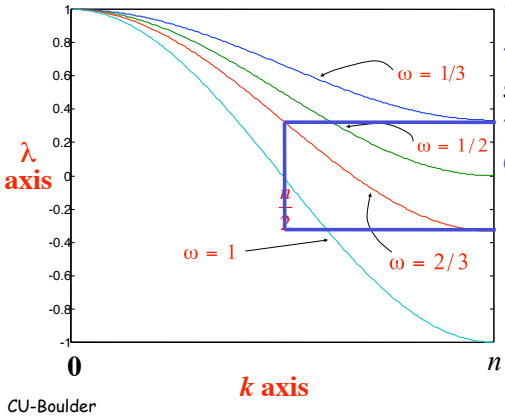
Note that if $0 < \omega \le 1$, then $|\lambda_k(R_\omega)| < 1$ for $k = 1,2,\ldots,n-1$.

For $0 < \omega \le 1$,

$$\lambda_1 = 1 - 2\omega\sin^2\left(\frac{\pi}{2n}\right)$$
$$= 1 - 2\omega\sin^2\left(\frac{\pi h}{2}\right)$$
$$= 1 - O(h^2) \approx 1$$

Low frequencies are "undamped"

Notice that no value of ω will efficiently damp out long waves or low frequencies.



What value of ω gives the best damping of short waves or high frequencies n/2≤k≤n-1? Choose ω such that

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

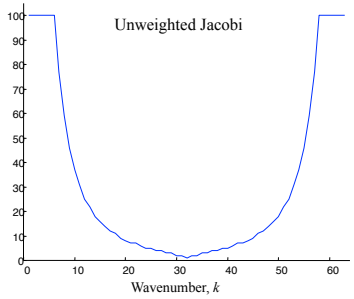
$$\Rightarrow \omega \approx \frac{2}{3}$$

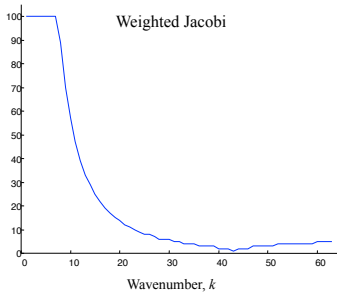
For 2D: $\omega \approx \frac{4}{5}$

Smoothing factor

- The smoothing factor is the largest magnitude of the iteration matrix eigenvalues corresponding to the oscillatory Fourier modes:
  smoothing factor = max $|\lambda_k(R)|$ for $n/2 \le k \le n-1$.
  "MG" spectral radius?
- Why only the upper spectrum?
- For R_ω with ω = 2/3, the smoothing factor is 1/3:
  $|\lambda_{n/2}| = |\lambda_{n-1}| = 1/3$ & $|\lambda_k| < 1/3$ for $n/2 < k < n-1$.
- But $|\lambda_k| \approx 1 - \omega k^2\pi^2h^2$ for long waves ($k \ll n/2$).

Convergence of Jacobi on Au = 0





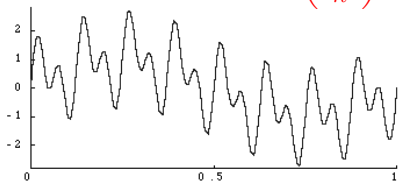
- Jacobi on Au = 0 with n = 64. Number of iterations needed to reduce initial error $||e||_\infty$ by 0.01.
- Initial guess :

$$v_{k,i} = \sin\left(\frac{ik\pi}{n}\right)$$

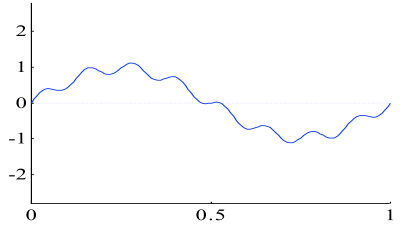
Weighted Jacobi = smoother (error)

- Initial error: $v_i = \sin\left(\frac{2i\pi}{n}\right) + \frac{1}{2}\sin\left(\frac{16i\pi}{n}\right) + \frac{1}{2}\sin\left(\frac{32i\pi}{n}\right)$

not solution or approximation!!!



- Error after 35 iteration sweeps:



Many relaxation schemes are smoothers: oscillatory error modes are quickly eliminated, but smooth modes are slowly damped.

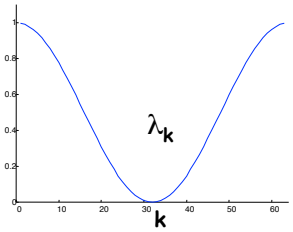
Similar analysis for other smoothers

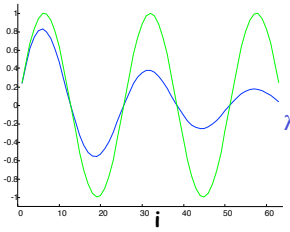
- Gauss-Seidel relaxation applied to the 3-point difference matrix A (1-D model problem):

$$R_G = (D - L)^{-1} U.$$

- A little algebra & trigonometry shows that

$$\lambda_k(R_G) = \cos^2\left(\frac{k\pi}{n}\right) \qquad (w_k)_i = \cos^i\left(\frac{k\pi}{n}\right)\sin\left(\frac{ik\pi}{n}\right) = \lambda_k^{i/2}\sin(k\pi x_i)$$



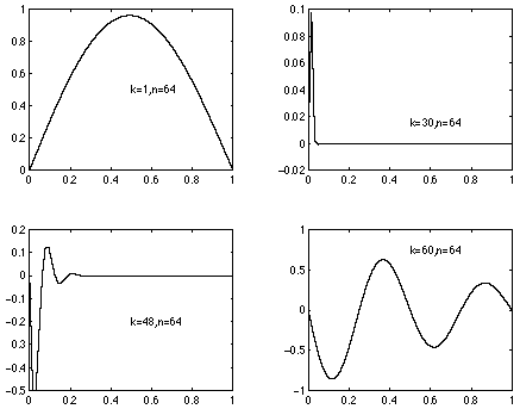


$\sin(3\pi x_i)$

$\lambda_3^{i/2}\sin(3\pi x_i)$

What's w_k look like for large k ?

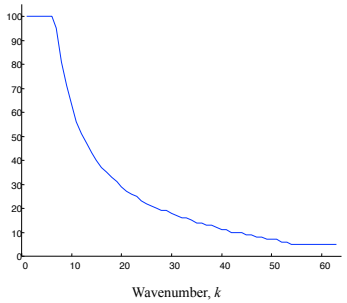
Gauss-Seidel eigenvectors



These are 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 hope that G-S produces smooth results!

Gauss-Seidel convergence

Eigenvectors of R_G are not the same as those of A!!!
Gauss-Seidel mixes the modes of A.

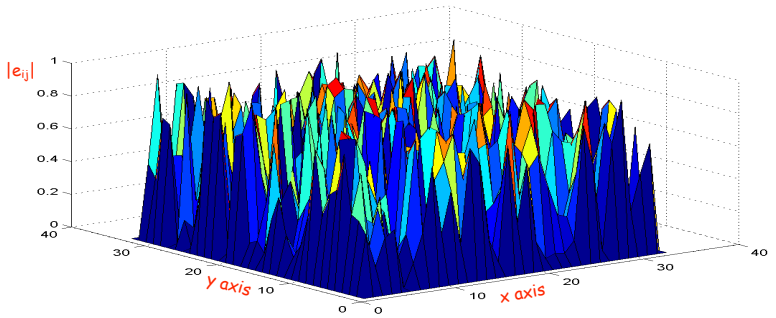


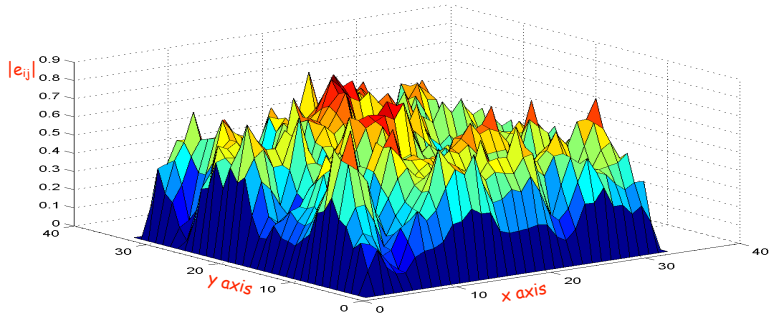
Gauss-Seidel on Au = 0, with n = 64. Number of iterations needed to reduce initial error $||e||_\infty$ by 0.01.

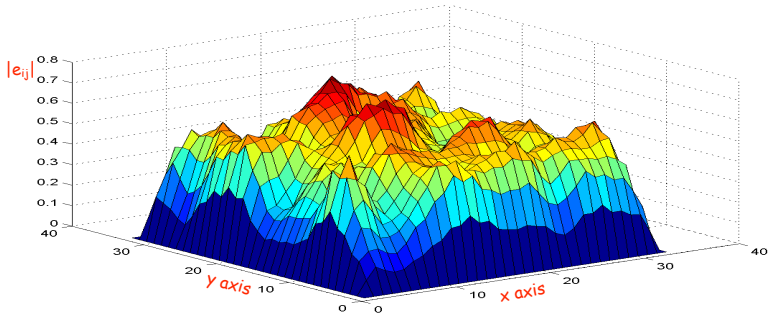
Initial guess (modes of A):

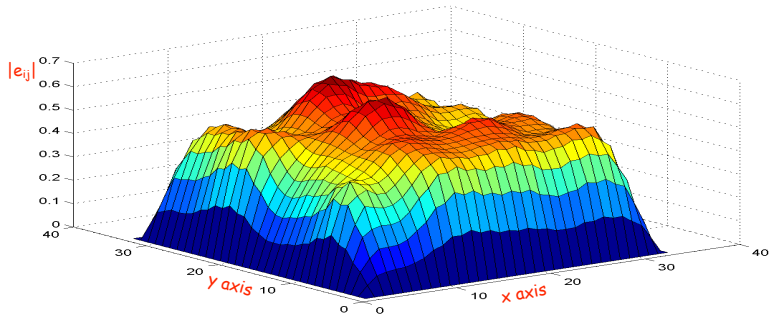
$$v_{ki} = \sin\left(\frac{ik\pi}{n}\right)$$

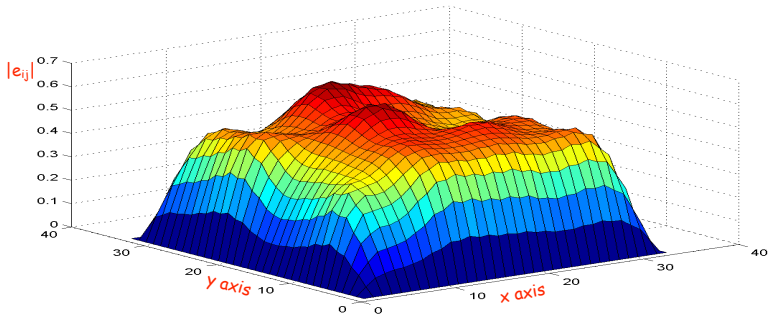
So G-S does reduce oscillatory Fourier modes.

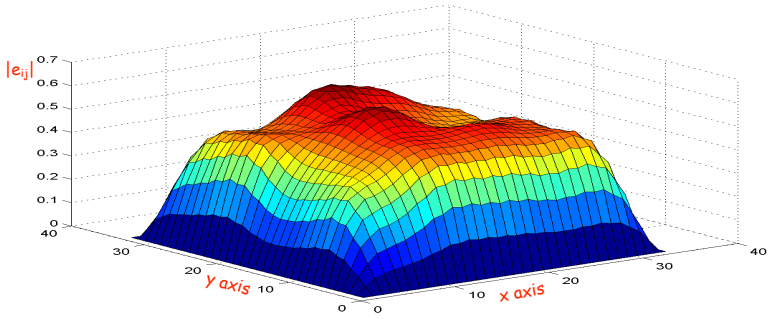


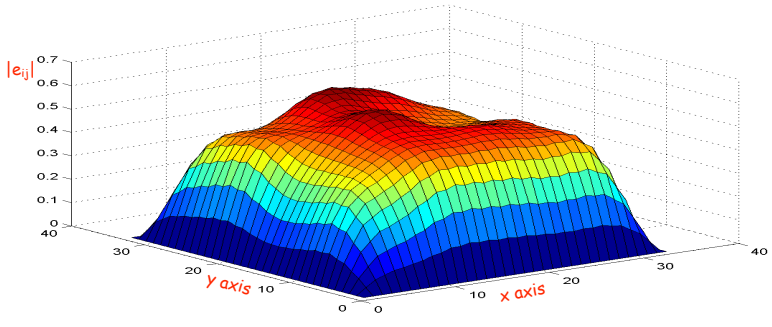

|eij|
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
40
30
20
y axis
10
0
0
10
20
30
40
x axis

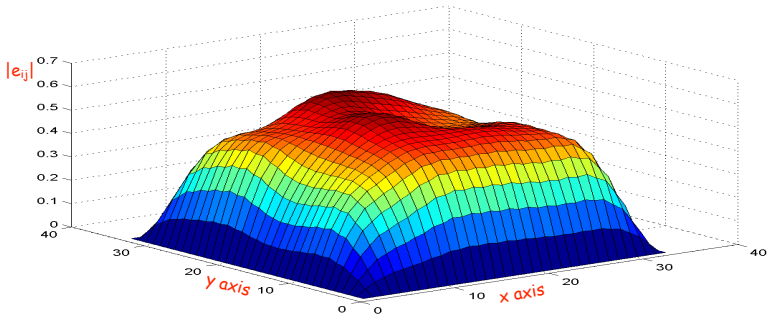

|eij|
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
40
30
20
y axis
10
0
0
10
20
30
40
x axis

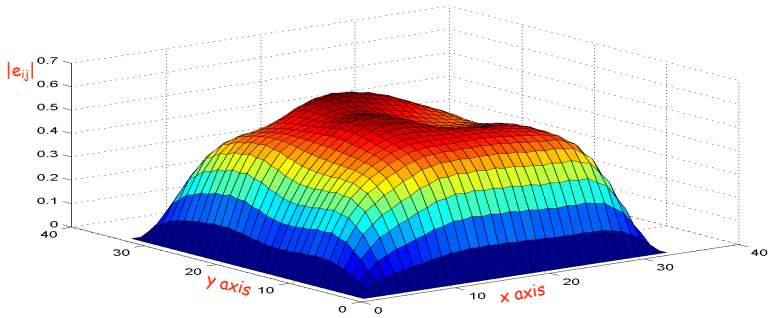

|eij|
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
40
30
20
y axis
10
0
0
10
20
30
40
x axis

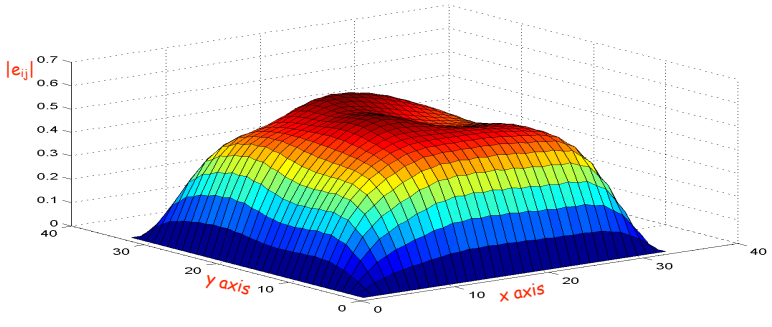

|eij|
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
40
30
20
y axis
10
0
0
10
20
30
40
x axis

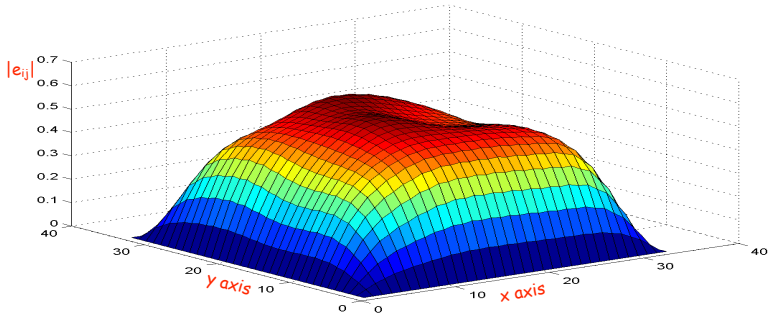

|eij|
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
40
30
20
y axis
10
0
0
10
20
30
40
x axis

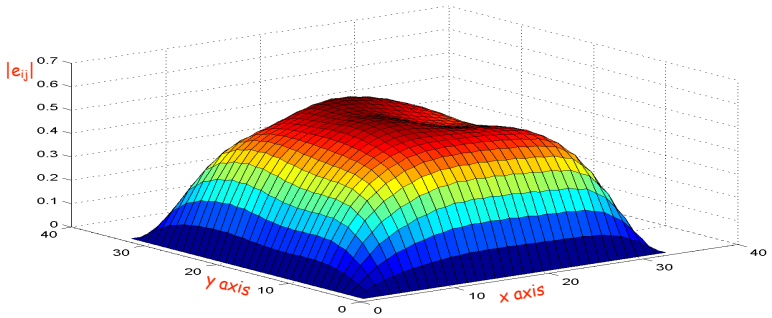

|eij|
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
40
30
20
y axis
10
0
0
10
20
30
40
x axis

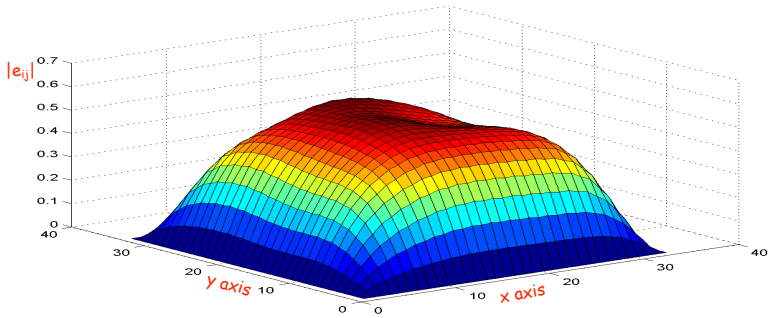

|eij|
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
40
30
20
y axis
10
0
0
10
20
30
40
x axis

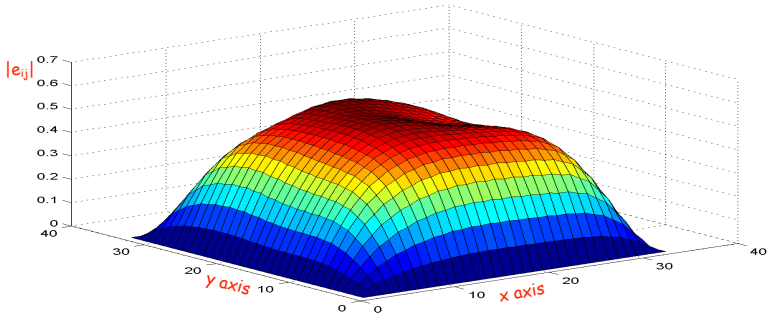

|eij|
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
40
30
20
y axis
10
0
0
10
20
30
40
x axis

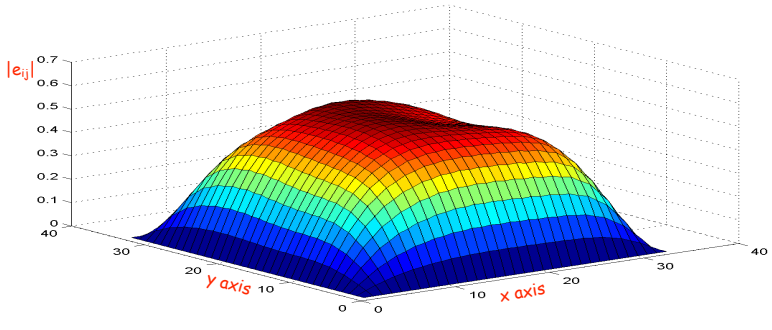

|eij|
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
40
30
20
y axis
10
0
0
10
20
30
40
x axis

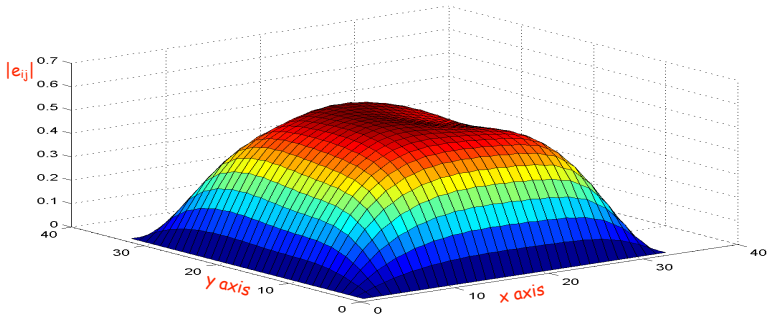

|eij|
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
40
30
20
y axis
10
0
0
10
20
30
40
x axis

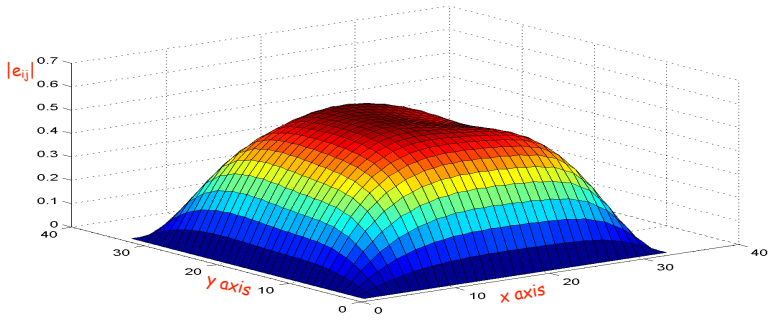

|eij|
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
40
30
20
y axis
10
0
0
10
20
30
40
x axis

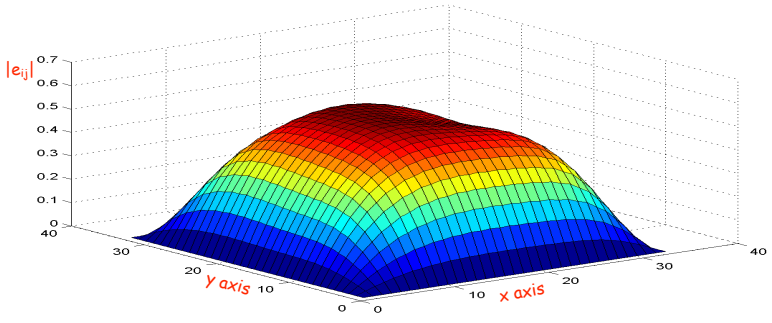

|eij|
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
40
30
20
y axis
10
0
0
10
20
30
40
x axis

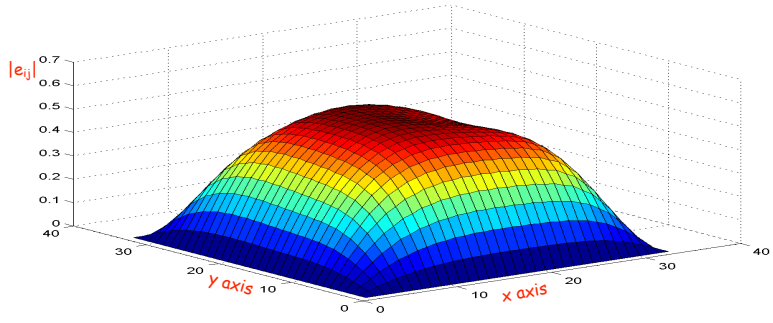

|eij|
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
40
30
20
y axis
10
0
0
10
20
30
40
x axis

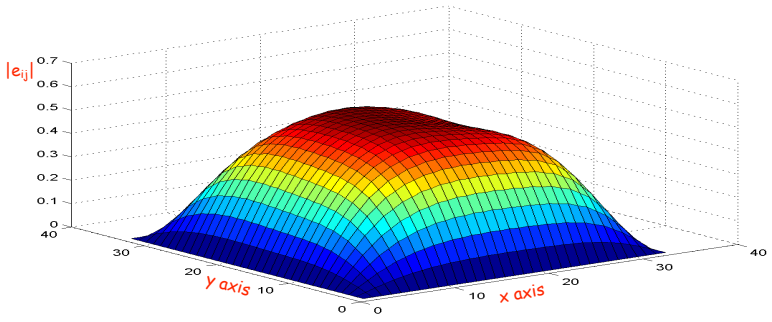

|eij|
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
40
30
20
y axis
10
0
0
10
20
30
40
x axis


|eij|
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
40
30
20
y axis
10
0
0
10
20
30
40
x axis


|eij|
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
40
30
20
y axis
10
0
0
10
20
30
40
x axis







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:

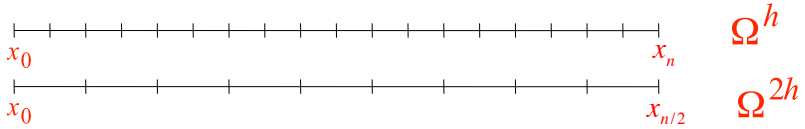
- Nonlinear Problems
  - Full approximation scheme
- Selected Applications
  - Neumann boundaries
  - Anisotropic problems
  - Variable meshes
  - Variable coefficients
- Algebraic Multigrid (AMG)
  - Matrix coarsening
- Multilevel Adaptive Methods
  - FAC
- Finite Elements
  - Variational methodology

Homework Due !

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.



How?

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 ($|\lambda_1(R)| \approx 1 - \omega\pi^2h^2$):

    $1 - O(4h^2)$ instead of $1 - O(h^2)$.

Idea! Nested iteration

⋮

- Relax on $Au = f$ on Ω^{4h} to obtain initial guess v^{2h}.
- Relax on $Au = f$ on Ω^{2h} to obtain initial guess v^h.
- Relax on $Au = f$ on Ω^h to obtain … final solution???
- What is $A^{2h}u^{2h} = f^{2h}$?

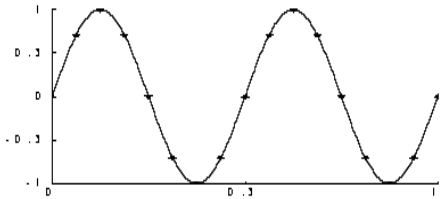
  Analogous to $A^hu^h = f^h$ for now.
- How do we migrate between grids?

  Hang on…
- What if the error still has large smooth components when we get to the fine grid Ω^h ?

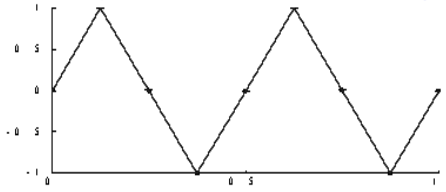
  Hang on…

Reason #2 for coarse grids:

- A smooth function:



can be represented by linear interpolation from a coarser grid:



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

Relaxation on 2h is cheaper & faster on this mode!!!

We need to see what happens to the modes on the coarse grid...

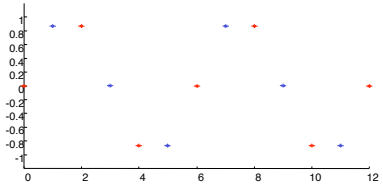
$$w^h_{k,2i} = \sin\left(\frac{2ik\pi}{n}\right) = \sin\left(\frac{ik\pi}{n/2}\right) = w^{2h}_{k,i}$$

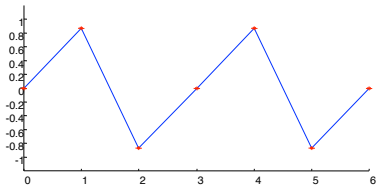
Also, note that

$w^h_{n/2} \to 0$

on the coarse grid.

What happens to the modes $k > n/2$?

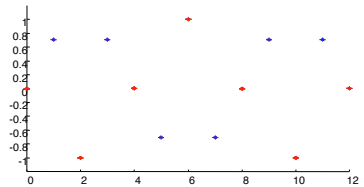


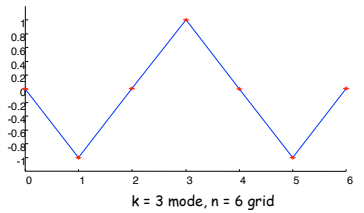


For $k > n/2$, w_k^h is disguised on the coarse grid: aliasing!!!

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

$$\begin{aligned}\left(w^h_k\right)_{2i} &= \sin\left(\frac{(2i)\pi k}{n}\right) \\ &= -\sin\left(\frac{2i\pi(n-k)}{n}\right) \\ &= -\sin\left(\frac{i\pi(n-k)}{n/2}\right) \\ &= -\left(w^{2h}_{n-k}\right)_i\end{aligned}$$

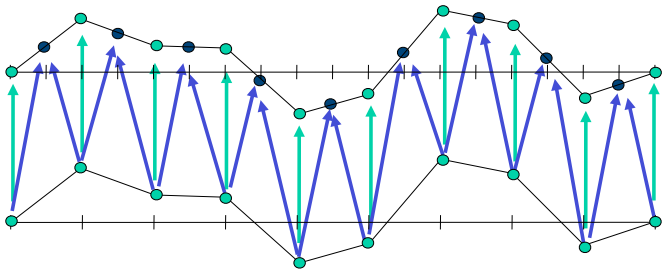




1-D interpolation (prolongation)

More precisely, let's relate grid 2h functions to grid h functions...

- 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.



We will often identify Ω^{2h} with a subset of Ω^h.

1-D interpolation (prolongation)

to migrate from coarse to fine grids

- Mapping from the coarse grid to the fine grid:

$$I^h_{2h} : \Omega^{2h} \longrightarrow \Omega^h \qquad (\Omega^h = \Re^{n-1})$$

- Let v^h, v^{2h} be defined on Ω^h, Ω^{2h}. Then

$$I^h_{2h}\, v^{2h} = v^h$$

where

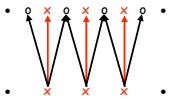
$$v^h_{2i} = v^{2h}_i \quad \text{for } 0 \le i \le \frac{n}{2} \text{ (including boundaries),}$$

$$v^h_{2i+1} = \frac{1}{2}\left(v^{2h}_i + v^{2h}_{i+1}\right) \quad \text{for } 0 \le i \le \frac{n}{2} - 1.$$

1-D prolongation operator P

- P = I_{2h}^{h} is a linear operator: $\Re^{n/2-1} \longrightarrow \Re^{n-1}$.
- n = 8:

$$\begin{pmatrix} \frac{1}{2} & & \\ 1 & & \\ \frac{1}{2} & \frac{1}{2} & \\ & 1 & \\ & \frac{1}{2} & \frac{1}{2} \\ & & 1 \\ & & \frac{1}{2} \end{pmatrix}_{7x3} \begin{pmatrix} v_1^{2h} \\ v_2^{2h} \\ v_3^{2h} \end{pmatrix}_{3x1} = \begin{pmatrix} v_1^h \\ v_2^h \\ v_3^h \\ v_4^h \\ v_5^h \\ v_6^h \\ v_7^h \end{pmatrix}_{7x1}$$



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

rank = max # linearly independent cols or rows

When is I_{2h}^{h} $v^{2h} = 0$?

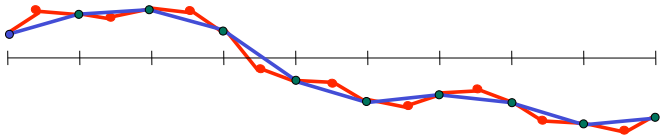
"Scatter" stencil for P

] 1/2 1 1/2 [

o x 1/2 1 1/2 x o

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

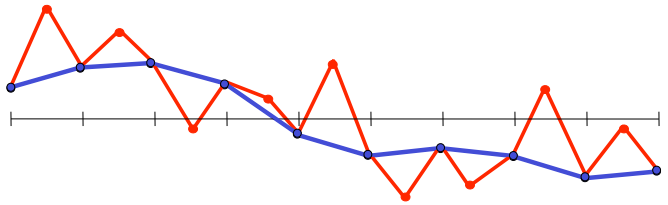
- Imagine that a coarse-grid approximation v^{2h} has been found. How well could it approximate the exact solution u ?



- 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.

Where do we stand?

	smooth components	oscillatory components
relaxation		
nested iteration		

The Key Step to Multigrid

- 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? →error←
- Can we smooth e? Can we get e & use it to get u ?

  $Ae = r$ & $u \leftarrow v + e$!
- So, use nested iteration on the 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…

Idea! Coarse-grid correction

2-grid

- Relax on $Au = f$ on Ω^h to get an approximation v^h.
- Compute $r = f - Av^h$.
- Transfer $Ae = r$ to Ω^{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 Ω^{2h}.

A way to coarsen Ae = r

Galerkin

- Assume we've relaxed so much that e is smooth.
- Ansatz: $e = Pv^{2h}$ for some coarse-grid v^{2h}.
- How do we characterize e so we can hope to compute it?

P when grids understood, else I_{2h}^{h}.

$$Ae = r \quad \Rightarrow \quad \underset{7x7}{A}\ \underset{7x3}{P}\ \underset{3x1}{v^{2h}} = \underset{7x1}{r}$$

- Too many equations now & too few unknowns!
- How about just eliminating every other equation?
- How about multiplying both sides by some 3x7 matrix?

$$\underbrace{\underbrace{P^T}_{3x7}\ \underbrace{A}_{7x7}\ \underbrace{P}_{7x3}}_{A^{2h}\ 3x3}\ \underbrace{v^{2h}}_{3x1} = \underbrace{P^T r}_{3x1}$$

We might write R instead of P^T or maybe I_h^{2h}.

1-D restriction by injection

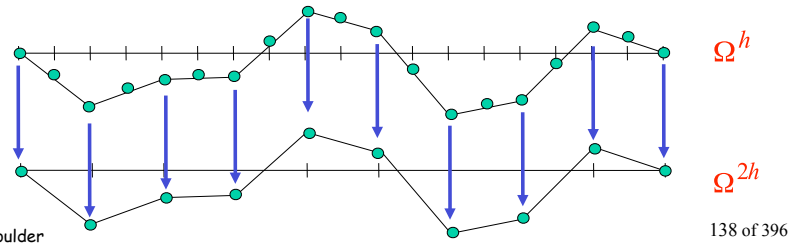
- Mapping from the fine grid to the coarse grid:

$$R = I_h^{2h} : \Omega^h \rightarrow \Omega^{2h}.$$ R is not P^T here!!!

- Let v^h, v^{2h} be defined on Ω^h, Ω^{2h}. Then

$$Rv^h = I_h^{2h} v^h = v^{2h},$$

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



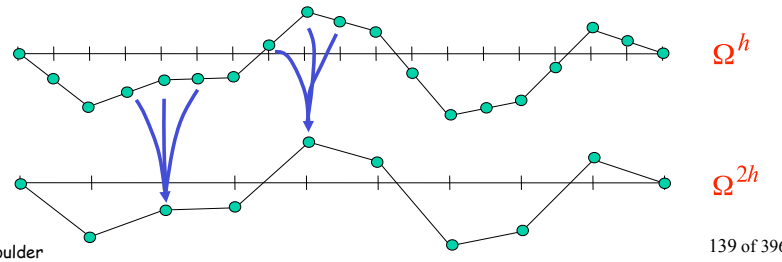
1-D restriction by full weighting

- Let v^h, v^{2h} be defined on Ω^h, Ω^{2h}. Then

$$Rv^h = I_h^{2h} v^h = v^{2h},$$ R is cP^T here!!!

where

$$v_i^{2h} = \frac{1}{4}\left(v_{2i-1}^h + 2v_{2i}^h + v_{2i+1}^h\right).$$



1-D restriction (full-weighting)

- R = I_h^{2h} is a linear operator: $\Re^{n-1} \longrightarrow \Re^{n/2-1}$.

Don't confuse R here with error propagator notation.

- n = 8:

$$\underset{3x7}{\begin{pmatrix} \frac{1}{4} & \frac{1}{2} & \frac{1}{4} & & & & \\ & & \frac{1}{4} & \frac{1}{2} & \frac{1}{4} & & \\ & & & & \frac{1}{4} & \frac{1}{2} & \frac{1}{4} \end{pmatrix}} \underset{7x1}{\begin{pmatrix} v_1^h \\ v_2^h \\ v_3^h \\ v_4^h \\ v_5^h \\ v_6^h \\ v_7^h \end{pmatrix}} = \underset{3x1}{\begin{pmatrix} v_1^{2h} \\ v_2^{2h} \\ v_3^{2h} \end{pmatrix}}.$$

- I_h^{2h} has rank $\frac{n}{2}-1$ because dim(Range(R)) $= \frac{n}{2}-1$.

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

$$I_{2h}^{h} = \frac{1}{2}\begin{pmatrix} 1 & & \\ 2 & & \\ 1 & 1 & \\ & 2 & \\ & 1 & 1 \\ & & 2 \\ & & 1 \end{pmatrix}, \qquad I_{h}^{2h} = \frac{1}{4}\begin{pmatrix} 1 & 2 & 1 & & & \\ & & 1 & 2 & 1 & \\ & & & & 1 & 2 & 1 \end{pmatrix}$$

- So they're related by the variational condition

$$I_{2h}^{h} = c\,(I_{h}^{2h})^{T}, \quad c \text{ in } \Re. \qquad P = cR^{T}$$

2-D prolongation

$$v_{2i,2j}^{h} = v_{ij}^{2h}$$

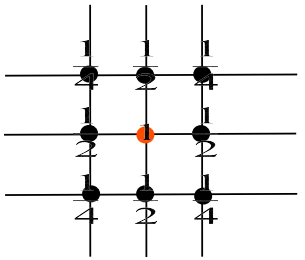
$$v_{2i+1,2j}^{h} = \frac{1}{2}(v_{ij}^{2h} + v_{i+1,j}^{h})$$

$$v_{2i,2j+1}^{h} = \frac{1}{2}(v_{ij}^{2h} + v_{i,j+1}^{h})$$

$$v_{2i+1,2j+1}^{h} = \frac{1}{4}(v_{ij}^{2h} + v_{i+1,j}^{h} + v_{i,j+1}^{h} + v_{i+1,j+1}^{h})$$

$$\begin{bmatrix} \frac{1}{4} & \frac{1}{2} & \frac{1}{4} \\ \frac{1}{2} & 1 & \frac{1}{2} \\ \frac{1}{4} & \frac{1}{2} & \frac{1}{4} \end{bmatrix}$$

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

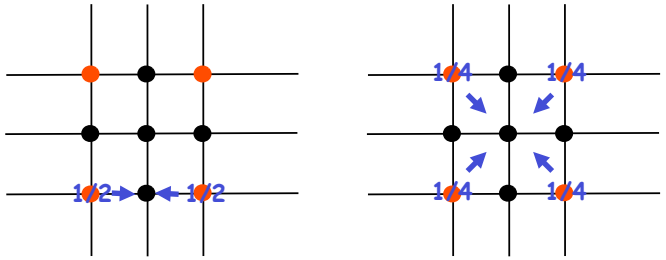


"Gather" interpolation stencil

[1/2 1/2]

$$\begin{bmatrix} 1/4 & 1/4 \\ & \\ 1/4 & 1/4 \end{bmatrix}$$

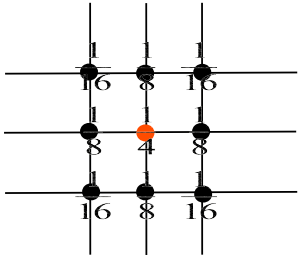
Centered over a fine-grid point ●.



2-D restriction (full weighting)

$$\begin{bmatrix} \frac{1}{16} & \frac{1}{8} & \frac{1}{16} \\ \frac{1}{8} & \frac{1}{4} & \frac{1}{8} \\ \frac{1}{16} & \frac{1}{8} & \frac{1}{16} \end{bmatrix}$$

We denote the operator by using a "gather" 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 (Relaxation on Coarse Grids)
  - effective on smooth solution (components).
- Relaxation on Fine Grid
  - effective on oscillatory error (components).
- Residual Equation on Fine Grid
  - 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

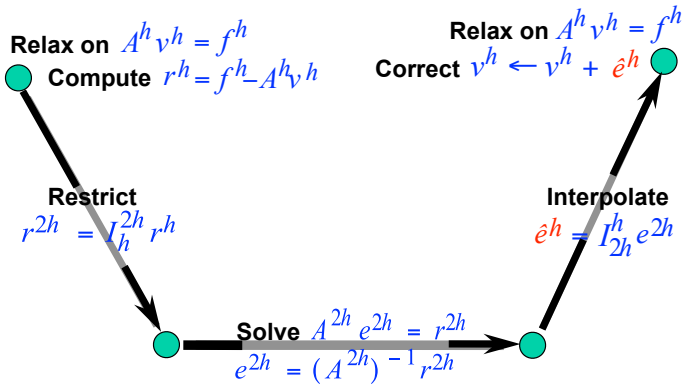
$v^h \leftarrow CG(v^h, f^h)$

1) Relax α_1 times on $A^h v^h = f^h$ on Ω^h with arbitrary initial guess v^h. If $h = h_{coarsest}$, then go to 6.
2) Compute $r^h = f^h - A^h v^h$.
3) Compute $r^{2h} = I_h^{2h} r^h$.
4) "Solve" $A^{2h} e^{2h} = r^{2h}$ on Ω^{2h}.
5) Correct fine-grid solution $v^h \leftarrow v^h + I_{2h}^h e^{2h}$.
6) Relax α_2 times on $A^h v^h = f^h$ on Ω^h.

$A^{2h} = I_h^{2h} A^h I_{2h}^h = RAP$ (Galerkin) or direct discretization

What does e^{2h} represent here?

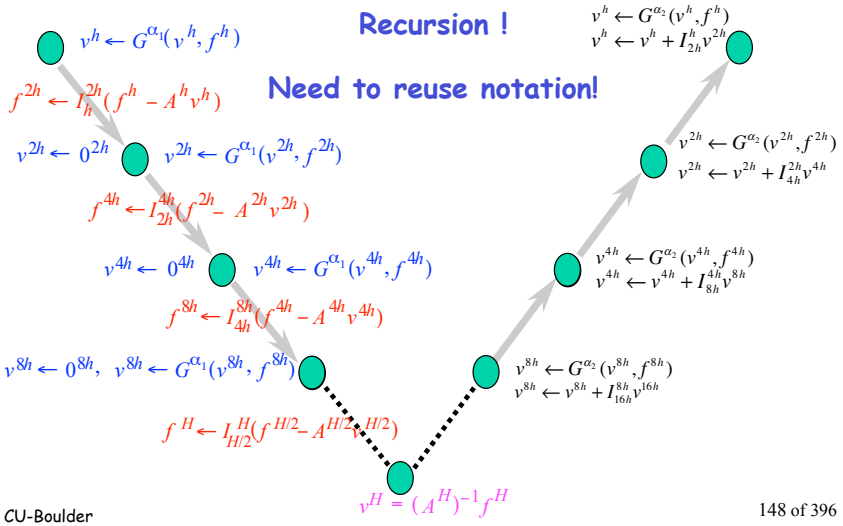
2-grid coarse-grid correction



How do we "solve" $A^{2h} e^{2h} = r^{2h}$?

Recursion !

Need to reuse notation!



V-cycle (recursive form)

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

1) Relax α_1 times on $A^h u^h = f^h$ with initial v^h given.

2) If Ω^h is the coarsest grid, go to 4;

else: $f^{2h} \leftarrow I_h^{2h}(f^h - A^h v^h)$

$v^{2h} \leftarrow 0$

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

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

4) Relax α_2 times on $A^h u^h = f^h$ with initial guess v^h.

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

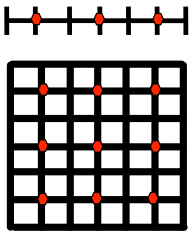
Homework Due !

4. Implementation

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

Estimates are approximate (n, d, …).

- 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, 4/3, & 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 sweep ($\alpha_1 = 2$) & 1 post-coarse-grid correction sweep ($\alpha_2 = 1$) .
- Cost of a V-cycle (in WUs):

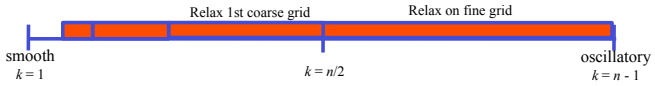
$$3(1 + 2^{-d} + 2^{-2d} + 2^{-3d} + \ldots + 2^{-md}) < \frac{3}{1-2^{-d}}$$

- Cost is about 2, 4/3, & 8/7 X 3 WUs per V-cycle in 1, 2, & 3 dimensions, respectively.

Convergence analysis

- First, a heuristic argument:
  - The convergence factor for the oscillatory error modes (smoothing factor) is small & bounded uniformly in h.

    smoothing factor = max $|\lambda_k(R)|$ for $n/2 \le k \le n-1$.
  - Multigrid focuses the relaxation process on attenuating the oscillatory components on each level.



⇒ The overall multigrid convergence factor is small & bounded uniformly in h !

Bounded uniformly in h ≠ independent of h.

Revisiting the model problem

Reminder: Approximate u''(x) via Taylor series

$O(h^2)$ means a quantity bounded in norm by Ch^2 for some constant C.

- The BVP:
- $u''(x) = f(x)$ 2nd $0 < x < 1$
- Approximate 2nd derivative using Taylor series:
- $u(0) = u(1) = 0$ 0
- The finite difference scheme:

$$u(x_{i+1}) = u(x_i) + h u'(x_i) + \frac{h^2}{2} u''(x_i) + \frac{h^3}{3!} u'''(x_i) + O(h^4) \quad 0$$

$$+$$

$$u(x_{i-1}) = u(x_i) - h u'(x_i) + \frac{h^2}{2!} u''(x_i) - \frac{h^3}{3!} u'''(x_i) + O(h^4)$$

$$\frac{-u_{i-1}^h + 2u_i^h - u_{i+1}^h}{h^2} = f_i \qquad i = 1,2,\ldots,n-1$$

- $u_0^h = u_n^h = 0$
- Summing & solving:
- Truncation error:

$$\frac{u(x_{i-1}) - 2u(x_i) + u(x_{i+1})}{h^2} + O(h^2) = f_i$$

$u(x)$ = exact PDE solution vector

$$u''(x_i) = \frac{u_{i-1}^{(h)} - 2u_i^{(h)} + u_{i+1}^{(h)}}{h^2} + O(h^2) \qquad i = 1,2,\ldots,n-1$$

earlier page 55

Actual error

A has h^{-2} in it here

$Au^{(h)} + O(h^2) = f$

$\Rightarrow \quad Au^{(h)} = f + O(h^2)$

$\Rightarrow \quad Au^h = f \quad (u^h$ = discrete sol'n)

$\Rightarrow \quad A(u^{(h)} - u^h) = O(h^2)$

$\Rightarrow \quad AE = O(h^2)$. ← consistency

(E = discretization error)

So $||E|| = ||A^{-1} O(h^2)||$

$\le ||A^{-1}|| \cdot ||O(h^2)||$

$= \lambda_{max}(A^{-1}) \cdot O(h^2) = O(h^2)/\lambda_{min}(A) \sim O(h^2)/\pi^2$

↑ stability

or

$||E|| = O(h^2)$. ← convergence

Overall goal of computation

- Continuous problem: $Au = f$, $u_i = u(x_i)$
- Discrete problem: $A^h u^h = f^h$, $v^h \approx u^h$
- Global/discretization error: $E_i = u(x_i) - u_i^h$

  $||E|| \le Kh^p$

  (p = 2 for model problem & proper norm)
- Algebraic error: $e_i^h = u_i^h - v_i^h$
- For tolerance ε, assume the aim is to find v^h so that the total error, $||e|| = ||u^{(h)} - v^h|| \le \varepsilon$, where $u^{(h)} = (u(x_i))$.
- Then this objective is assured as follows:

  $||e|| \le ||u^{(h)} - u^h|| + ||u^h - v^h|| = ||E|| + ||e^h|| \le \varepsilon$.

We can satisfy the convergence objective by imposing two conditions

1) $||E|| \le \varepsilon/2$. Achieve this condition by choosing an appropriately small grid spacing h:

$$Kh^p = \varepsilon/2.$$

2) $||e^h|| \le \varepsilon/2$. Achieve this condition by iterating until $||e^h|| \le \varepsilon/2 = Kh^p$ on grid h; then we've

converged to the level of discretization error.

→ Once discretization error & algebraic error are in balance, then it would be better to go to grid $h/2$ than to iterate more!

Convergence to the level of discretization error

- Use an MV scheme with convergence factor $\gamma < 1$ bounded uniformly in h (fixed α_1 & α_2).
- Assume a d-dimensional problem on an n^d grid with $h = 1/n$.
- Initial relative error: $||e^h||/||u^h|| = ||u^h - 0||/||u^h|| = 1$.
- Must reduce this to $||e^h||/||u^h|| = O(h^p) = O(n^{-p})$.
- We can determine the number of V-cycles needed for this if we can bound the convergence factor, γ.

Work to converge to the level of discretization error

- Using θ V-cycles with convergence factor γ gives an overall convergence factor of γ^θ.
- We therefore have $\gamma^\theta = O(n^{-p})$, or $\theta = O(\log n)$.
- Since 1 V-cycle costs $O(1)$ WUs & 1 WUs is $O(n^d)$, then converging to the level of discretization error using the MV method cost

$$O(n^d \log n).$$

- Compares to fast direct methods (fast Poisson solvers).

But multigrid can do even better…

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 $(n+1)x(n+1)$ grids [$(n-1)x(n-1)$ interior points] for $n = 16, 32, 64, 128$.

Numerical results

norms 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.

$||r^h||_2$.

$||r^h||_h = h\ ||r^h||_2$

scaled residual error

$||e||_h = h\ ||u^{(h)} - v^h||_2$

scaled discrete total error

	$n = 16$				$n = 32$			
V-cycle	$\|\|r^h\|\|_h$	ratio	$\|\|e\|\|_h$	ratio	$\|\|r^h\|\|_h$	ratio	$\|\|e\|\|_h$	ratio
0	6.75e+02		5.45e−01		2.60e+03		5.61e−01	
1	4.01e+00	0.01	1.05e−02	0.02	1.97e+01	0.01	1.38e−02	0.02
2	1.11e−01	0.03	4.10e−04	0.04	5.32e−01	0.03	6.32e−04	0.05
3	3.96e−03	0.04	1.05e−04	0.26	2.06e−02	0.04	4.41e−05	0.07
4	1.63e−04	0.04	1.03e−04	0.98*	9.79e−04	0.05	2.59e−05	0.59
5	7.45e−06	0.05	1.03e−04	1.00*	5.20e−05	0.05	2.58e−05	1.00*
6	3.75e−07	0.05	1.03e−04	1.00*	2.96e−06	0.06	2.58e−05	1.00*
7	2.08e−08	0.06	1.03e−04	1.00*	1.77e−07	0.06	2.58e−05	1.00*
8	1.24e−09	0.06	1.03e−04	1.00*	1.10e−08	0.06	2.58e−05	1.00*
9	7.74e−11	0.06	1.03e−04	1.00*	7.16e−10	0.06	2.58e−05	1.00*
10	4.99e−12	0.06	1.03e−04	1.00*	4.79e−11	0.07	2.58e−05	1.00*
11	3.27e−13	0.07	1.03e−04	1.00*	3.29e−12	0.07	2.58e−05	1.00*
12	2.18e−14	0.07	1.03e−04	1.00*	2.31e−13	0.07	2.58e−05	1.00*
13	2.33e−15	0.11	1.03e−04	1.00*	1.80e−14	0.08	2.58e−05	1.00*
14	1.04e−15	0.45	1.03e−04	1.00*	6.47e−15	0.36	2.58e−05	1.00*
15	6.61e−16	0.63	1.03e−04	1.00*	5.11e−15	0.79	2.58e−05	1.00*

	$n = 64$				$n = 128$			
V-cycle	$\|\|r^h\|\|_h$	ratio	$\|\|e\|\|_h$	ratio	$\|\|r^h\|\|_h$	ratio	$\|\|e\|\|_h$	ratio
0	1.06e+04		5.72e−01		4.16e+04		5.74e−01	
1	7.56e+01	0.01	1.39e−02	0.02	2.97e+02	0.01	1.39e−02	0.02
2	2.07e+00	0.03	6.87e−04	0.05	8.25e+00	0.03	6.92e−04	0.05
3	8.30e−02	0.04	4.21e−05	0.06	3.37e−01	0.04	4.22e−05	0.06
4	4.10e−03	0.05	7.05e−06	0.17	1.65e−02	0.05	3.28e−06	0.08
5	2.29e−04	0.06	6.45e−06	0.91*	8.99e−04	0.05	1.63e−06	0.50
6	1.39e−05	0.06	6.44e−06	1.00*	5.29e−05	0.06	1.61e−06	0.99*
7	8.92e−07	0.06	6.44e−06	1.00*	3.29e−06	0.06	1.61e−06	1.00*
8	5.97e−08	0.07	6.44e−06	1.00*	2.14e−07	0.06	1.61e−06	1.00*
9	4.10e−09	0.07	6.44e−06	1.00*	1.43e−08	0.07	1.61e−06	1.00*
10	2.87e−10	0.07	6.44e−06	1.00*	9.82e−10	0.07	1.61e−06	1.00*
11	2.04e−11	0.07	6.44e−06	1.00*	6.84e−11	0.07	1.61e−06	1.00*
12	1.46e−12	0.07	6.44e−06	1.00*	4.83e−12	0.07	1.61e−06	1.00*
13	1.08e−13	0.07	6.44e−06	1.00*	3.64e−13	0.08	1.61e−06	1.00*
14	2.60e−14	0.24	6.44e−06	1.00*	1.03e−13	0.28	1.61e−06	1.00*
15	2.30e−14	0.88	6.44e−06	1.00*	9.19e−14	0.89	1.61e−06	1.00*

A warning about bounds

- Bounds like $||e_{n+1}|| \le \gamma\ ||e_n||$ & $||u^{(h)} - u^h|| = O(h)$ 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 $O(h) \neq O(h^2)$!!!

(Any process that is $O(h^2)$ is also $O(h)$, but the converse isn't necessarily true.)

Reconsideration

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)

$v^h \leftarrow FMG(f^h)$

- Initialize $f^h, f^{2h}, f^{4h}, \ldots, f^H$
- Solve on coarsest grid $v^H = (A^H)^{-1} f^H$

  ⋮
- Interpolate initial guess $v^{2h} \leftarrow I_{4h}^{2h} v^{4h}$
- 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 (recursive form)

$v^h \leftarrow FMG(f^h), \eta$

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

2) If h = H, then go to 4 (where MV is a direct solve);

   else: $v^{2h} \leftarrow FMG(f^{2h})$.

3) Set initial guess: $v^h \leftarrow I_{2h}^h v^{2h}$.

We use $\eta = 1$.

4) Perform $v^h \leftarrow MV(v^h, f^h)$ η times.

FMG cycle cost

One V(2,1)-cycle is performed per level, at a cost of $3/(1 - 2^{-d})$ 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

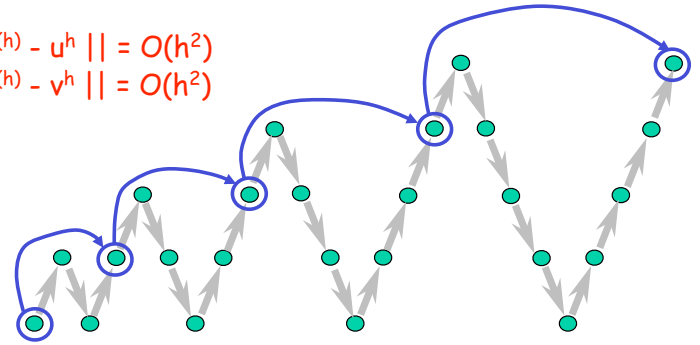
$$[3/(1 - 2^{-d})](1 + 2^{-d} + 2^{-2d} + \ldots) = 3/(1 - 2^{-d})^2.$$

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

Has discretization error been reached by FMG?

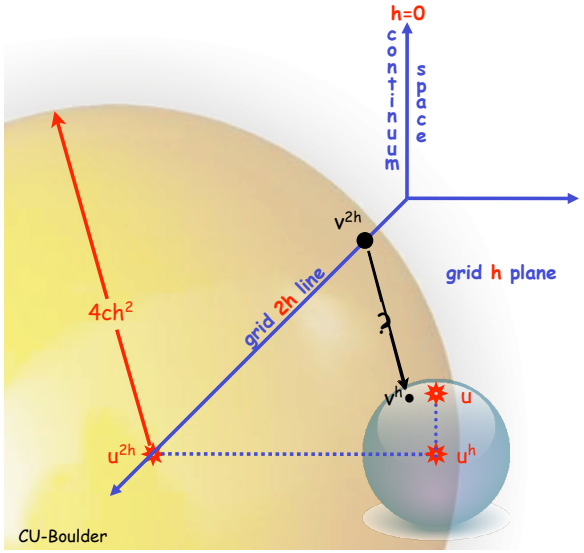
If discretization error is achieved, then $||e^h|| = O(h^2)$ & the V-cycle approximation converges to the solution of the PDE about as well as the discrete solution does:

$||u^{(h)} - u^h|| = O(h^2)$
$||u^{(h)} - v^h|| = O(h^2)$



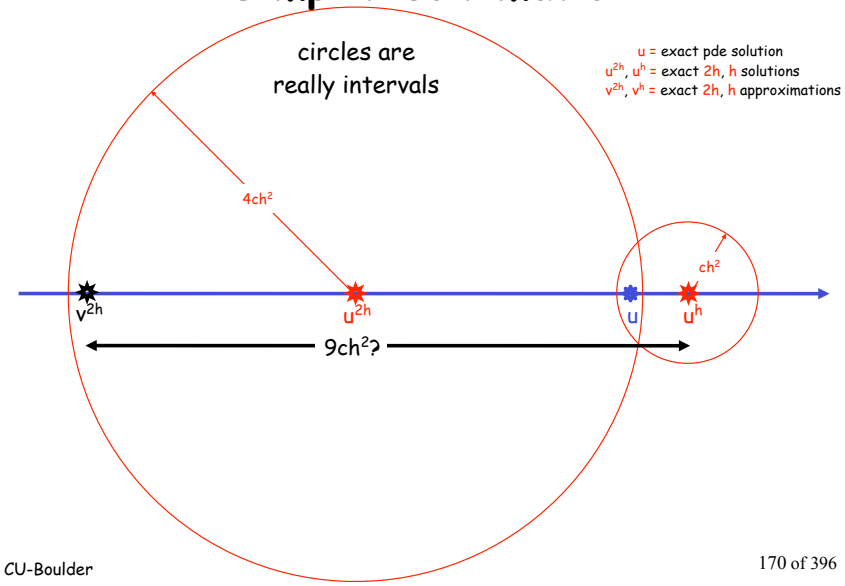
We need to be more careful…

The basic FMG principle



u = exact pde solution
u^{2h}, u^h = exact 2h, h solutions
v^{2h}, v^h = exact 2h, h approximations

Simpler schematic



u = exact pde solution
u^{2h}, u^h = exact 2h, h solutions
v^{2h}, v^h = exact 2h, h approximations

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 $u^{(2h)}$ & $u^{(h)}$ are the "same", right? So, if u^{2h} approximates $u^{(2h)}$ to order $O(4h^2)$ & 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 comparisons is a bit technical.
- In other words, here comes the algebra...

Interpolation stability

how interpolation affects error

- Property: $||Pe^{2h}|| \le \beta ||e^{2h}||$
- Reasoning:

$$||Pe^{2h}|| \le ||P||\ ||e^{2h}||$$
$$= ||P^TP||^{1/2}\ ||e^{2h}||$$

$||P||^2 = \max ||Pe||^2/||e||^2$
$= \max \langle Pe,Pe\rangle/\langle e,e\rangle$
$= \max \langle P^TPe,e\rangle/\langle e,e\rangle$
$= \rho(P^TP)$
$= ||P^TP||$

$$P^TP = \begin{pmatrix} 1/2 & 1 & 1/2 & & \\ & 1/2 & 1 & 1/2 & \\ & & 1/2 & 1 & 1/2 \end{pmatrix} \begin{pmatrix} 1/2 & & & & \\ & 1 & & & \\ 1/2 & 1/2 & & & \\ & 1 & 1/2 & & \\ & & 1/2 & 1 & \\ & & & 1/2 & \end{pmatrix} = \begin{pmatrix} 3/2 & 1/4 & & \\ 1/4 & 3/2 & 1/4 & \\ & 1/4 & 3/2 & \end{pmatrix}$$

$$\Rightarrow \quad ||P^TP||^{1/2} \le \sqrt{2}$$

In practice, $\beta \approx 1$.

Approximation property

how the discrete solutions approximate each other

$$\|u^h - Pu^{2h}\| \le \alpha K h^2$$

$$|(u^{(h)} - Pu^{(2h)})_i| = |u(x_i) - (u(x_{i-1}) + u(x_{i+1}))/2| \le |u''(\xi_i)|h^2/2$$

$$\Rightarrow \|u^{(h)} - Pu^{(2h)}\| \le ch^2 \qquad (\|u^h - u^{(h)}\| \approx Kh^2)$$

(We need $\|u''\| << \infty$, so the norm is scaled by h here.)

$$\Rightarrow \|u^h - Pu^{2h}\|$$
$$\le \|u^h - u^{(h)}\| + \|u^{(h)} - Pu^{(2h)}\| + \|Pu^{(2h)} - Pu^{2h}\|$$
$$\le Kh^2 + ch^2 + \|P\|\cdot\|u^{(2h)} - u^{2h}\|$$
$$\le Kh^2 + ch^2 + \beta K(2h)^2$$
$$\le (1 + c + 4\beta)Kh^2$$

In practice, $\alpha = 1 + c + 4\beta \approx 5$.

FMG accuracy

$\|e^h\| \le Kh^2$

Assume:

$\|e^{2h}\| \le K(2h)^2$ induction hypothesis

$\|u^h - Pu^{2h}\| \le \alpha Kh^2$ approximation property ($\alpha \approx 5$)

$\|Pw^{2h}\| \le \beta\|w^{2h}\|$ interpolation stability ($\beta \approx 1$)

Triangle inequality:

$$\|e^h\| = \|u^h - Pv^{2h}\|$$
$$\le \|u^h - Pu^{2h}\| + \|P(u^{2h} - v^{2h})\|$$
$$\le \alpha Kh^2 + \beta K(2h)^2$$
$$= (\alpha + 4\beta)Kh^2$$

$$\Rightarrow \|e^h\| \le \text{“9”}Kh^2$$

So we need only reduce $\|e^h\|$ by “0.1”!!!

Numerical example

- Consider again the 2-D model problem (with $\sigma = 0$):

  $-u_{xx} - u_{yy} = 2[(1 - 6x^2)y^2(1 - y^2) + (1 - 6y^2)x^2(1 - x^2)]$

  inside the unit square, with $u = 0$ on the boundary.

- We examine the effectiveness of FMG cycling to solve the problem on $(n+1)\times(n+1)$ grids [$(n-1)\times(n-1)$ interior points] for $n = 2, 4, \ldots, 2048$.

FMG results

FMG cycle results & comparison with MV cycle costs

N	FMG(1,0) $\|e\|_h$	ratio	FMG(1,1) $\|e\|_h$	ratio	FMG(2,1) $\|e\|_h$	ratio	FMG(1,1) WU	V(2,1) cycles	V(2,1) WU
2	5.86e−03		5.86e−03		5.86e−03				
4	5.37e−03	0.917	2.49e−03	0.424	2.03e−03	0.347	7/2	3	12
8	2.78e−03	0.518	9.12e−04	0.367	6.68e−04	0.328	7/2	4	16
16	1.19e−03	0.427	2.52e−04	0.277	1.72e−04	0.257	1.03e-04	4	16
32	4.70e−04	0.395	6.00e−05	0.238	4.00e−05	0.233	2.58e-05	5	20
64	1.77e−04	0.377	1.36e−05	0.227	9.36e−06	0.234	6.44e-06	5	20
128	6.49e−05	0.366	3.12e−06	0.229	2.26e−06	0.241	1.61e-06	6	24
256	2.33e−05	0.359	7.35e−07	0.235	5.56e−07	0.246	7/2	7	28
512	8.26e−06	0.354	1.77e−07	0.241	1.38e−07	0.248	7/2	7	28
1024	2.90e−06	0.352	4.35e−08	0.245	3.44e−08	0.249	7/2	8	32
2048	1.02e−06	0.351	1.08e−08	0.247	8.59e−09	0.250	7/2	9	36

$\|e\|_h = h\,\|u^{(h)} - v^h\|_2$

scaled discrete total error

Successful Scientific Inquiry

- Attitude
  - Knowledge is good, but understanding rules!
  - Look for the underlying principle!
  - 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 weak result do instead?
- Intelligence
  - It doesn't hurt to try to be "smart" too.

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: $f^h = A^h u^h$ with u^h given.
- Make sure u^h satisfies the right boundary conditions.
- Test relaxation starting with u^h: Is $r^h = 0$, is it zero after relaxation, does u^h change?
- Test coarse-grid correction starting with u^h: Is the correction zero?

Tool # 5: Test on $Au^h = 0$

- The exact solution $u^h = 0$ is known!
- Residual norm $||Av^h||$ & error norm $||v^h||$ are computable.
- Norms $||Av^h||$ & $||v^h||$ 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 v^h have already been computed! Computing $r^h = f^h - Av^h$ & updating v^h shouldn't have trouble with machine precision if you have $u^h = 0$ & thus $f^h = 0$.

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

dropping superscript h when it's clear by context

- Use the discrete L^2 norm:

$$||r||_h = (h^d \Sigma r_i^2)^{1/2} = h^{d/2} ||r||_2.$$

- 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.

Beware of residuals

example h^{-2} (-1 2 -1)

$e_k = \sin(k\pi x)$

$\lambda_1 \approx \pi^2$, $\lambda_n \approx 4h^{-2}$

$$Ae = r$$

- Relative errors: $\frac{||e||_h}{||u||_h}$ vs. $\frac{||Ae||_h}{||Au||_h} = \frac{||r||_h}{||f||_h}$
- Absolute range: $||e_1||_h \approx 1$ & $||Ae_1||_h \approx \pi^2$

  $||e_n||_h \approx 1$ & $||Ae_n||_h \approx 4h^{-2}$!!!
- Relative errors: consider the case $u = e_n$

  $\frac{||e_n||_h}{||u||_h} = 1$ & $\frac{||Ae_n||_h}{||Au||_h} = 1$

  $\frac{||e_1||_h}{||u||_h} = 1$ & $\frac{||Ae_1||_h}{||Au||_h} \approx (\pi^2/4)h^2$!!!

Moral: residuals can falsely signal convergence when the error is smooth.

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, or other special phenomena?

Tool # 9: Test two-level cycling

- Replace the coarse-grid V-cycle recursive call with a direct solver if possible, or iterate many times with some method known to "work" (test ||r|| to be sure it's very small), or 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

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 $f \in \text{Range}(\partial_{xx}) = \eta^{\perp}(\partial_{xx}) = \{1\}^{\perp}$ or $f \perp 1$.
- First fix the grid h right side: $f^h \leftarrow f^h - (\langle f^h, 1\rangle/\langle 1,1\rangle)1$.
- Do this on coarse grids too: $f^{2h} \leftarrow f^{2h} - (\langle f^{2h}, 1\rangle/\langle 1,1\rangle)1$.
- Uniqueness is also a worry: $u^h \leftarrow u^h - (\langle u^h, 1\rangle/\langle 1,1\rangle)1$.

Tool # 13: Test for linearity

also a bit ahead of schedule…

- 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 ($f = -u''$ in Ω) & data ($g = u$ on Γ).
- Do multigrid results compare qualitatively with sampled u?
- Monitor $||u - u^h||_h$.
- Test a case with no discretization error ($u = ax^2 + bx + c$). The algebraic error should tend steadily to 0.
- Test discretization error ($u^{iv} \neq 0$). 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 $O(h^2)$ ($h \to h/2 \Rightarrow e \to e/2$) or however it should?

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, $V(\nu_1, \nu_2)$, errors, labels (table, graph)
- Use various scenarios:
  $Ax = 0$, $Ax = f$, varying n & ν_i & ω, 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:
  by hand ok ↓ ↓
  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
  - Anisotropic problems
  - Variable meshes
  - Variable coefficients
- Algebraic Multigrid (AMG)
  - Matrix coarsening
- Multilevel Adaptive Methods
  - FAC
- Finite Elements
  - Variational methodology

Homework Due !

5. Some theory

What is A^{2h}?

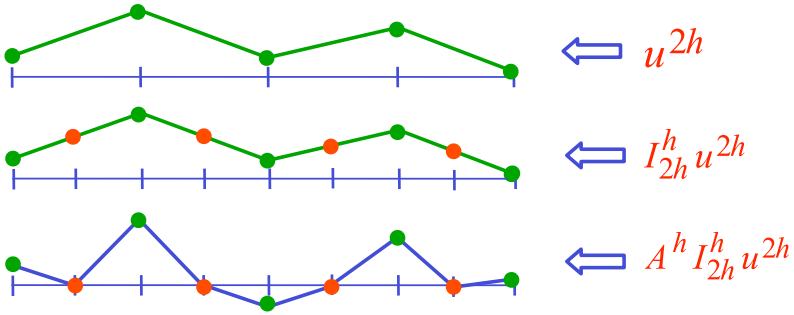
- Recall the 2-grid coarse-grid correction scheme:
  - 1) Relax on $A^h u^h = f^h$ on Ω^h to get v^h.
  - 2) Compute $f^{2h} = I_h^{2h}(f^h - A^h v^h)$.
  - 4) Solve $A^{2h}u^{2h} = f^{2h}$ on Ω^{2h}.
  - 5) Correct fine-grid solution $v^h \leftarrow v^h + I_{2h}^h u^{2h}$.
- Assume that $e^h \in \text{Range}(I_{2h}^h)$, i. e., $e^h = I_{2h}^h u^{2h}$ for some $u^{2h} \in \Omega^{2h}$. Then the residual equation can be written

$$A^h e^h = A^h I_{2h}^h u^{2h} = r^h.$$

This characterizes u^{2h}, but with too many equations.

- How does A^h act on I_{2h}^h?

How does A^h act on 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 Ω^{2h}. We do this by applying restriction, either injection or full weighting:

$$I_h^{2h} A^h I_{2h}^h u^{2h} = I_h^{2h} r^h.$$

We use full weighting from now on unless otherwise stated.

Building A^{2h}: The Galerkin condition

- The residual equation on the coarse grid is

$$I_h^{2h} A^h I_{2h}^h u^{2h} = I_h^{2h} r^h$$

- We thus identify the coarse-grid operator as

$$A^{2h} = I_h^{2h} A^h I_{2h}^h \qquad A^{2h} = RA^hP$$

- RAP is symmetric:

  If $P^T = \alpha R$ (so that $R^T = (1/\alpha)P$), then
  $(RAP)^T = P^T A^T R^T = \alpha(1/\alpha)RAP = RAP$.

- RAP is positive definite: P full rank $\Rightarrow Px \neq 0$.

  If $x \neq 0$, then $\langle RAPx, x\rangle = (1/\alpha)\langle APx, Px\rangle > 0$!

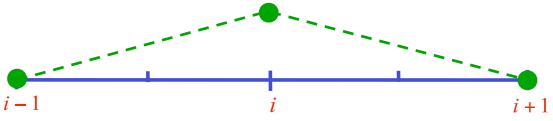
Computing the i^{th} row of A^{2h}

- Compute $A^{2h}\hat{e}_i^{2h}$, where $\hat{e}_i^{2h} = (0, 0, \ldots, 0, 1, 0, \ldots, 0)^T$.

↑ Why is this the i^{th} row of A^{2h}?

$A^{2h} = RA^hP$

$A^h = h^{-2}[-1\ 2\ -1]$



0		1		0	$\hat{e}_i^{2h}$
0	$\frac{1}{2}$	1	$\frac{1}{2}$	0	$I_{2h}^h \hat{e}_i^{2h}$
$-\frac{1}{2h^2}$	0	$\frac{1}{h^2}$	0	$-\frac{1}{2h^2}$	$A^h I_{2h}^h \hat{e}_i^{2h}$
$=\frac{1}{(2h)^2}$		$\frac{2}{(2h)^2}$		$=\frac{1}{(2h)^2}$	$I_h^{2h} A^h I_{2h}^h \hat{e}_i^{2h}$

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 $\frac{1}{(2h)^2}[-1 \quad 2 \quad -1]$,

  which is the Ω^{2h} version of A^h.

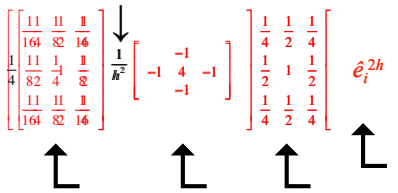
- Note that **IF** relaxation on Ω^h leaves only error in the range of interpolation, then solving

  $$A^{2h} u^{2h} = f^{2h}$$

  determines the error exactly!
- This is generally not feasible, but this logic motivates wanting $e^h \in \text{Range}(I_{2h}^h)$ & it leads to a very plausible representation for A^{2h}.

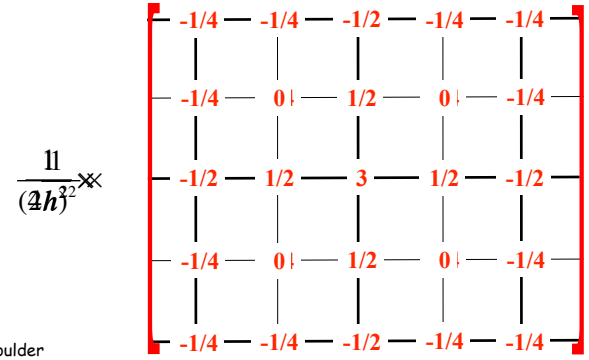
2-D RAP

scale



Try $\frac{1}{h^2}\begin{bmatrix} -1/3 & -1/3 & -1/3 \\ -1/3 & 8/3 & -1/3 \\ -1/3 & -1/3 & -1/3 \end{bmatrix}$

homework !



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:

$$A^{2h} = I_h^{2h} A^h I_{2h}^h \qquad \text{Galerkin Condition}$$

$$I_{2h}^h = c\,(I_h^{2h})^T \qquad c \in \Re$$

Properties of restriction

in a little more detail…

- Full Weighting: $I_h^{2h} : \Omega^h \longrightarrow \Omega^{2h}$ or

  $I_h^{2h} : \Re^{n-1} \longrightarrow \Re^{n/2-1}$

- n = 8:

$$I_h^{2h} = \frac{1}{4}\begin{bmatrix} 1 & 2 & 1 & & & & \\ & & 1 & 2 & 1 & & \\ & & & & 1 & 2 & 1 \end{bmatrix}_{3\times 7}$$

- I_h^{2h} has rank $\frac{n}{2}-1$ & null space $\eta(I_h^{2h})$ with dim $\frac{n}{2}$.

Spectral properties of restriction

- How does I_h^{2h} act on the eigenvectors of A^h?
- Consider $w_{k,j}^h = \sin\left(\frac{jk\pi}{n}\right)$, $1 \le k \le n-1$, $0 \le j \le n-1$.
- A little algebra & trigonometry shows that

$$\left(I_h^{2h} w_k^h\right)_j = \cos^2\left(\frac{k\pi}{2n}\right) w_{k,j}^{2h}$$
$$\equiv c_k w_{k,j}^{2h}$$

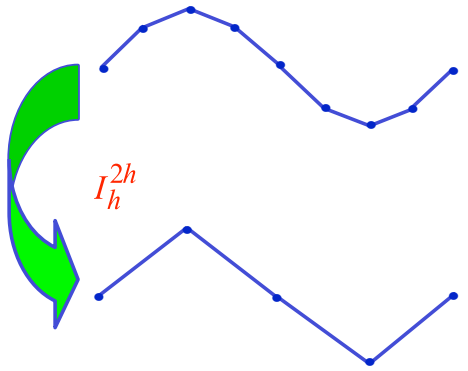
for $1 \le k \le n/2-1$.

$c_k = O(1)$

$c_1 \approx 1 \ldots c_n = O(h^2)$

Spectral properties (cont'd)

- i.e., I_h^{2h} [k th mode on Ω^h] = c_k [k th mode on Ω^{2h}]



Ω^h: n = 8, k = 2

Ω^h: n = 4, k = 2

Spectral properties (cont'd)

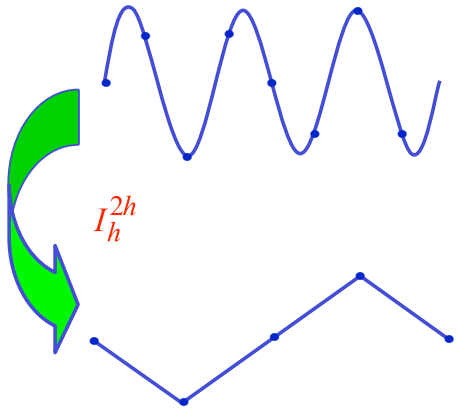
- Let k' = n - k for $1 \le k \le n/2-1$, so that
  $n/2+1 \le k' \le n-1$.
- A little algebra & trigonometry shows that

$$\left(I_h^{2h} w_{k'}^h\right)_j = -\sin^2\left(\frac{k\pi}{2n}\right) w_{k,j}^{2h}$$
$$\equiv -s_k w_{k,j}^{2h}.$$

$s_1 = O(h^2) \ldots s_{n/2-1} = O(1)$

Spectral properties (cont'd)

i.e., I_h^{2h} [(n - k)th mode on Ω^h] = $-s_k$ [k th mode on Ω^{2h}]



Ω^h: n = 8, n - k = 6

Ω^h: n = 4, k = 2

Spectral properties (cont'd)

- Summarizing:

$c_k = O(1)$

$$\left.\begin{aligned} I_h^{2h} w_k^h &= c_k\, w_k^{2h} \\ I_h^{2h} w_{k'}^h &= -s_k\, w_k^{2h} \end{aligned}\right\} \quad \left\{\begin{aligned} 1 \le k \le \frac{n}{2} - 1 \\ k' = n - k \end{aligned}\right.$$

$$I_h^{2h} w_{n/2}^h = 0$$

$s_1 = O(h^2) \ldots s_{n/2-1} \approx O(1)$

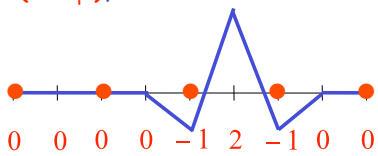
- Complementary modes:

$$W_k = \text{span}\{w_k^h, w_{k'}^h\}$$

$$I_h^{2h} W_k \Longrightarrow \{w_k^{2h}\}^\infty$$

Null space of restriction

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



- While the η_i looks oscillatory, it generally contains all Fourier modes of A^h:

$$\eta_i = \sum_{k=1}^{N} a_k w_k \qquad a_k \neq 0$$

- All the Fourier modes of A^h are needed to represent the null space of restriction!

Properties of interpolation

- Interpolation: $I_{2h}^h : \Omega^{2h} \Longrightarrow \Omega^h$ or

$I_{2h}^h : \Re^{n/2-1} \Longrightarrow \Re^{n-1}$

- n = 8:

$$I_{2h}^h = \frac{1}{2}\begin{bmatrix} 1 & & & & & & & \\ 2 & & & & & & & \\ 1 & 1 & & & & & & \\ & 2 & & & & & & \\ & 1 & 1 & & & & & \\ & & 2 & & & & & \\ & & 1 & 1 & & & & \\ & & & 2 & & & & \\ & & & 1 & & & & \end{bmatrix}$$

- 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 $(w_k^{2h})_j = \sin\left(\frac{jk\pi}{n/2}\right)$, 1 ≤ k ≤ n/2-1, 0 ≤ j ≤ n/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":

$$\begin{aligned} I_{2h}^h w_k^{2h} &= \cos^2\left(\frac{k\pi}{2n}\right) w_k^h - \sin^2\left(\frac{k\pi}{2n}\right) w_{k'}^h \\ &= c_k w_k^h - s_k w_{k'}^h \end{aligned}$$

Spectral properties of interpolation

$$I_{2h}^h w_k^{2h} = c_k w_k^h - s_k w_{k'}^h$$

- Interpolation of smooth Ω^{2h} modes excites oscillatory modes on Ω^h.
- Note that if $k \ll \frac{n}{2}$, then

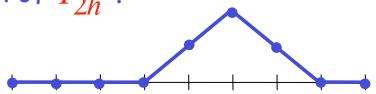
$$I_{2h}^h w_k^{2h} = \left(1 - O\left(\frac{k^2}{(n-1)^2}\right)\right) w_k^h + O\left(\frac{k^2}{(n-1)^2}\right) w_{k'}^h$$
$$\approx w_k^h$$

- 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 ξ_i be the ith column of I_{2h}^h.

ξ_i :



$$\xi_i^h = \sum_{k=1}^{n-1} b_k w_k^h, \quad b_k \neq 0$$

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

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

relax only before correction

1) Relax once on Ω^h: $v^h \leftarrow R_\omega v^h + B f^h$.

error propagation matrix not restriction!

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

3) Solve residual equation: $v^{2h} = (A^{2h})^{-1} f^{2h}$.

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

The entire process appears as

$$v^h \leftarrow R_\omega v^h + B f^h + I_{2h}^h (A^{2h})^{-1} I_h^{2h} (f^h - A^h (R_\omega v^h + B f^h))$$

The exact solution satisfies

$$u^h = R_\omega u^h + B f^h + I_{2h}^h (A^{2h})^{-1} I_h^{2h} (f^h - A^h (R_\omega u^h + B f^h))$$

CG error propagation

- Subtracting the previous two expressions, we get

$$e^h \leftarrow \left[I - I_{2h}^h (A^{2h})^{-1} I_h^{2h} A^h \right] R_\omega e^h$$

$$e^h \leftarrow CG\, e^h$$

- How does CG act on the modes of A^h? Assume e^h consists of the modes w_k^h & $w_{k'}^h$ for $1 \le k \le \frac{n}{2} - 1$ & $k' = n - k$.
- We know how R_ω^α, A^h, I_h^{2h}, $(A^{2h})^{-1}$, I_{2h}^h act on w_k^h & $w_{k'}^h$.

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

$$s_k = \sin^2\left(\frac{k\pi}{2n}\right) \qquad c_k = \cos^2\left(\frac{k\pi}{2n}\right)$$

$s_1 = O(h^2)$ ... $s_{n/2-1} \approx O(1)$ $\quad c_k = O(1)$

CG error propagation for k<<n

- Consider the case k << n (extremely smooth & oscillatory modes):

$$w_k \to O\left(\frac{k^2}{(n-1)^2}\right) w_k + O\left(\frac{k^2}{(n-1)^2}\right) w_{k'}$$

$$w_{k'} \to \left(1 - O\left(\frac{k^2}{(n-1)^2}\right)\right) w_k + \left(1 - O\left(\frac{k^2}{(n-1)^2}\right)\right) w_{k'}$$

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

CG with relaxation

Next, include one relaxation sweep. Note that error propagator R_ω preserves the modes of A^h (although this is often unnecessary). Let λ_k denote the eigenvalue of R_ω associated with w_k.

For k ≤ n/2-1:

$$w_k \to \lambda_k s_k w_k + \lambda_k s_k w_{k'} \quad \text{Small!}$$

$$w_{k'} \to \lambda_{k'} c_k w_k + \lambda_{k'} c_k w_{k'} \quad \text{Small!}$$

$$s_k = \sin^2\left(\frac{k\pi}{2n}\right) \qquad c_k = \cos^2\left(\frac{k\pi}{2n}\right)$$

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:

$$A^{2h} = I_h^{2h} A^h I_{2h}^h \qquad \text{Galerkin Condition}$$

$$I_{2h}^h = c\,(I_h^{2h})^T \qquad c \in \Re$$

Fundamental Theorem of Linear Algebra

$N(B) = R(B^T)^\perp$

- $x \in N(B)$ any $y \in R(B^T)$ some z

  $\Rightarrow \langle x, y \rangle = \langle x, B^T z \rangle = \langle Bx, z \rangle = 0.$

  $\Rightarrow x \in R(B^T)^\perp$

  $\Rightarrow N(B) \subset R(B^T)^\perp.$

- $x \in R(B^T)^\perp$ any z $z = Bx$

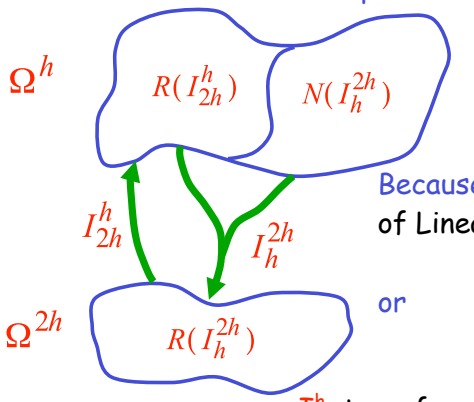
  $\Rightarrow 0 = \langle x, B^T z \rangle = \langle Bx, z \rangle = \langle Bx, Bx \rangle.$

  $\Rightarrow x \in N(B)$

  $\Rightarrow N(B) \supset R(B^T)^\perp.$

Algebraic interpretation of CG

consider the subspaces that make up Ω^h & Ω^{2h}



From now on, 'R( )' means Range & 'N( )' Null Space.

Because the Fundamental Theorem of Linear Algebra shows that

$$N(I_h^{2h}) = R((I_h^{2h})^T)^\perp$$

or

$$N(I_h^{2h}) = R(I_{2h}^h)^\perp$$

I_{2h}^h transfers errors?

Does I_h^{2h} transfer errors?

If not what does I_h^{2h}?

So we really care about $N(I_h^{2h} A^h)$!?!?

"Energy" inner product & norm

- Inner product symmetry:

  $\langle Ax, y \rangle = \langle x, Ay \rangle = \langle Ay, x \rangle.$

- Inner product linearity:

  $\langle A(ax+by), z \rangle = \langle aAx+bAy, z \rangle = a\langle Ax, z \rangle + b\langle Ay, z \rangle.$

- Inner product positive definiteness:

  $\langle Ax, x \rangle \geq 0 \quad \& \quad \langle Ax, x \rangle = 0 \Rightarrow x = 0.$

- Norm:

  $\langle Ax, x \rangle$ is an inner product $\Rightarrow \langle Ax, x \rangle^{1/2}$ is a norm.

Subspace decomposition of Ω^h

- If $u^h \in N(I_h^{2h} A^h)$, then, for any u^{2h}, we have

$$0 = \langle I_h^{2h} A^h u^h, u^{2h} \rangle = \langle A^h u^h, I_{2h}^h u^{2h} \rangle ,$$

so

$$R(I_{2h}^h) \perp_{A^h} N(I_h^{2h} A^h) ,$$

where $x \perp_{A^h} y$ means $\langle A^h x, y \rangle = 0$. "energy" inner product

- Moreover, any e^h can be written as $e^h = s^h + t^h$, where $s^h \in R(I_{2h}^h)$ & $t^h \in N(I_h^{2h} A^h)$.
- Hence, we get the "energy-orthogonal" decomposition

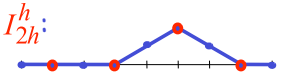
$$\Omega^h = R(I_{2h}^h) \oplus N(I_h^{2h} A^h).$$

errors touched by coarse grid

errors invisible to coarse grid

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:



- 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 $N(I_h^{2h})$ is spanned by $\eta_i = A^h \hat{e}_i$, so $N(I_h^{2h} A^h)$ is spanned by the unit vectors $\hat{e}_i^h = (0,0,\ldots,0,1,0,\ldots,0)^T$ for odd i, which "look" oscillatory.

Algebraic analysis CG

- Recall that (without relaxation)

$$CG = I - I_{2h}^h (A^{2h})^{-1} I_h^{2h} A^h .$$

- 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

$$CG s^h = \left[I - I_{2h}^h (A^{2h})^{-1} \underbrace{I_h^{2h} A^h}_{A^{2h}} \right] I_{2h}^h q^{2h} = 0 .$$

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 $t^h \in N(I_h^{2h} A^h)$, then

$$CG\, t^h = \left[I - I_{2h}^h (A^{2h})^{-1} \underbrace{I_h^{2h} A^h}_{0} \right] t^h$$

$$\Rightarrow \quad CG\, t^h = t^h$$

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

What does this imply?

Together: $CG(s^h + t^h) = t^h$

How does the algebraic picture fit with the spectral view?

We may view Ω^h in two ways:

$$\Omega^h = \{\text{Low frequency modes } 1 \le k \le n/2\} \oplus \{\text{High frequency modes } n/2 < k < n\}$$

that is,

$$\Omega^h = L \oplus H$$

Illuminates relaxation (Jacobi)

or

$$\Omega^h = R(I_{2h}^h) \oplus N(I_h^{2h} A^h).$$

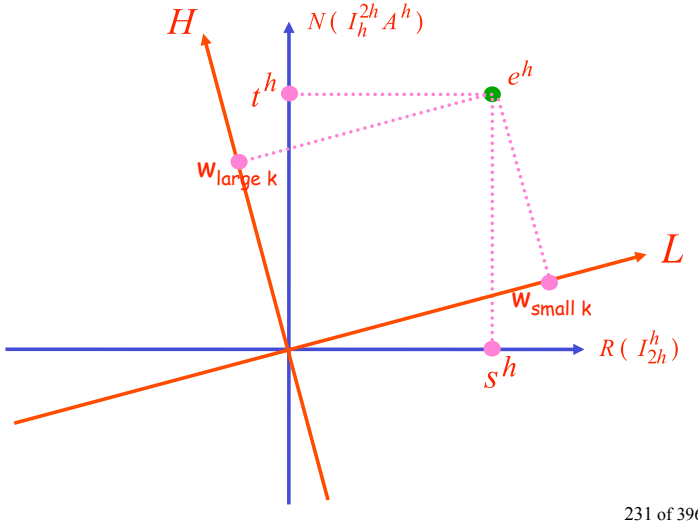
Illuminates CG (exact)

Are these "orthogonal" decompositions?

Actually, each view is just part of the picture

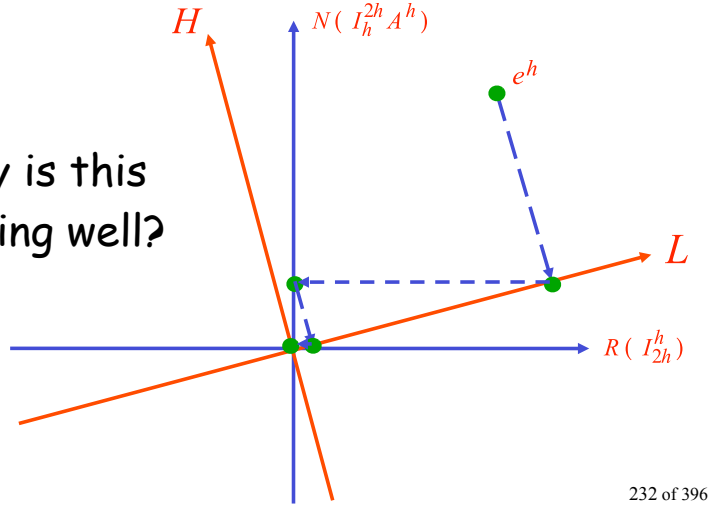
- The operations we've examined work on different spaces!
- While $N(I_h^{2h} A^h)$ is mostly oscillatory, it isn't H, & while $R(I_{2h}^h)$ is mostly smooth, it isn't L.
- Relaxation "eliminates" error from H.
- Coarse-grid correction eliminates error from $R(I_{2h}^h)$.

How it actually works (cartoon)

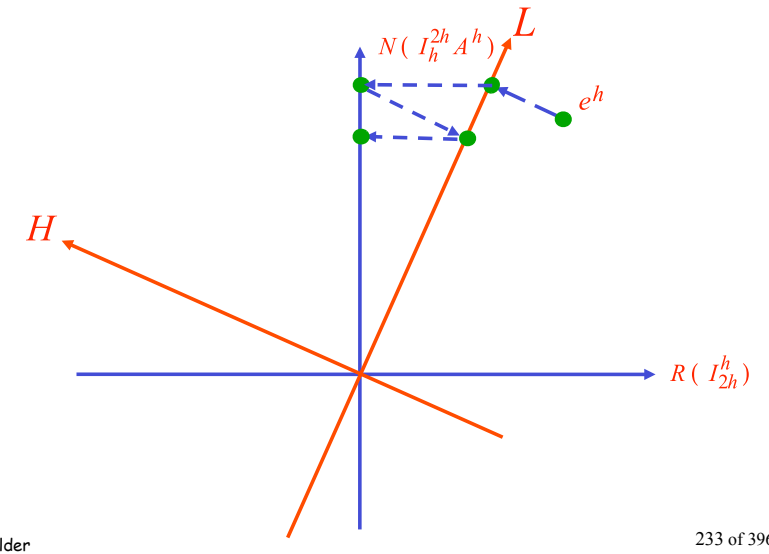


How it actually works (cartoon)

Why is this working well?



What if L points away from $R(I_{2h}^h)$?



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 !

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

  $A(u) - A(v) \neq A(e).$

The nonlinear residual equation

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

$$A(u) = f$$
$$\Rightarrow \quad A(u) - A(v) = f - A(v)$$
$$\Rightarrow \quad \boxed{A(u) - A(v) = r}$$

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

Newton's method for scalar $F: \Re \to \Re$

Best known & most important nonlinear solver!

- We wish to solve $F(x) = 0$.

  Ex: $F(x) = x e^x - 1$, $F'(x) = (1 + x) e^x$.

  exponent, not error

- Expand F in a Taylor series about x:

  $$F(x + s) = F(x) + sF'(x) + s^2 F''(\xi).$$

  Ex: $(x + s)e^{(x+s)} - 1 = x e^x - 1 + s(1 + x) e^x +$ h.o.t.

- Dropping higher-order terms (h.o.t.), if $x + s$ is a solution,

  $$0 = F(x) + sF'(x) \Rightarrow s = -F(x)/F'(x).$$

- 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

$$A(u) \equiv \begin{pmatrix} a_1(u) \\ a_2(u) \\ \vdots \\ a_{n-1}(u) \end{pmatrix} \equiv \begin{pmatrix} \frac{2u_1 - u_2}{h^2} + u_1 e^{u_1} \\ \vdots \\ \frac{-u_{i-1} + 2u_i - u_{i+1}}{h^2} + u_i e^{u_i} \\ \vdots \\ \frac{-u_{n-2} + 2u_{n-1}}{h^2} + u_{n-1} e^{u_{n-1}} \end{pmatrix} = \begin{pmatrix} f(x_1) \\ \vdots \\ f(x_i) \\ \vdots \\ f(x_{n-1}) \end{pmatrix} \equiv f.$$

Jacobian

- Taylor series about v: $A(v + e) = A(v) + J(v)e +$ h.o.t.

error, not exponent

$$J(v) \equiv \left(\frac{\partial a_i(u)}{\partial u_j} \right)_{u=v} = \begin{pmatrix} \frac{2}{h^2} + (1 + v_1) e^{v_1} & \frac{-1}{h^2} & & & \\ \frac{-1}{h^2} & \ddots & \ddots & & \\ & \ddots & \frac{2}{h^2} + (1 + v_i) e^{v_i} & \frac{-1}{h^2} & \\ & & \ddots & \ddots & \frac{-1}{h^2} \\ & & & \frac{-1}{h^2} & \frac{2}{h^2} + (1 + v_{n-1}) e^{v_{n-1}} \end{pmatrix}.$$

Newton for systems

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

$$\begin{pmatrix} a_1(u_1, u_2, \ldots, u_{n-1}) \\ a_2(u_1, u_2, \ldots, u_{n-1}) \\ \vdots \\ a_N(u_1, u_2, \ldots, u_{n-1}) \end{pmatrix} = \begin{pmatrix} f_1 \\ f_2 \\ \vdots \\ f_{n-1} \end{pmatrix}.$$

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

  $$A(v + e) = A(v) + J(v)e +$$ h.o.t.

$$J(v) = \left(\frac{\partial a_i}{\partial u_j}(v) \right).$$

Newton for systems (cont'd)

- $J(v)$ is the Jacobian

$$J(v) = \begin{pmatrix} \frac{\partial a_1}{\partial u_1} & \frac{\partial a_1}{\partial u_2} & \cdots & \frac{\partial a_1}{\partial u_{n-1}} \\ \frac{\partial a_2}{\partial u_1} & \frac{\partial a_2}{\partial u_2} & \cdots & \frac{\partial a_2}{\partial u_{n-1}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial a_{n-1}}{\partial u_1} & \frac{\partial a_{n-1}}{\partial u_2} & \cdots & \frac{\partial a_{n-1}}{\partial u_{n-1}} \end{pmatrix}_{u=v}.$$

- If $u = v + e$ is a solution, $f = A(v) + J(v)e +$ h.o.t., so

  $$e \approx [J(v)]^{-1}(f - A(v)).$$

- This leads to the iteration

  $$v \leftarrow v + [J(v)]^{-1}(f - A(v))$$

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.:

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

or

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

- Newton's method is thus:

$$v \leftarrow v + [J(v)]^{-1} r, \quad r = f - A(v)$$

How does multigrid fit in?

- One obvious method is to use multigrid to solve $J(v)\,\hat{e} = r$ 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?

$$A(u) = f$$

- Nonlinear Gauss-Seidel:

  For each $i = 1, 2, \ldots, n-1$:

  Change the value of v_i so that the i^{th} equation is satisfied: $(A(v))_i = f_i$.

- Equivalently:

  For each $i = 1, 2, \ldots, n-1$:

  Find $s \in \Re$ such that $(A(v + s\,\varepsilon_i))_i = f_i$,

  where ε_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

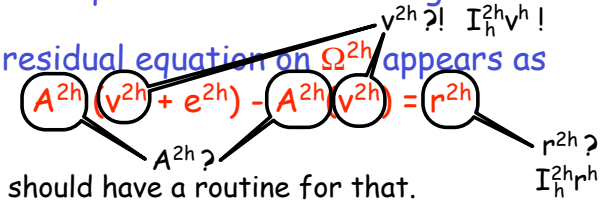
$$\frac{-v_{i-1} + 2v_i - v_{i+1}}{h^2} + v_i e^{v_i} = f_i \qquad 1 \le i \le n-1$$

- Newton iteration for v_i is given by

$$v_i \leftarrow v_i - \frac{\dfrac{-v_{i-1} + 2v_i - v_{i+1}}{h^2} + v_i\,e^{v_i} - f_i}{\dfrac{2}{h^2} + (1 + v_i)\,e^{v_i}}$$

How do we coarsen 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 Ω^{2h} appears as



We should have a routine for that.

Look at the coarse residual equation

- We must evaluate the quantities on Ω^{2h} in
  $A^{2h}(v^{2h} + e^{2h}) - A^{2h}(v^{2h}) = r^{2h}$.
- Given v^h, a fine-grid approximation, we restrict the residual to the coarse grid:
  $r^{2h} = I_h^{2h}(f^h - A^h(v^h))$.
- For v^{2h}, we restrict v^h by $v^{2h} = I_h^{2h} v^h$.

Thus,

$$A^{2h}(I_h^{2h} v^h + e^{2h}) = A^{2h}(I_h^{2h} v^h) + I_h^{2h}(f^h - A^h(v^h))$$

We've obtained a coarse-grid equation of the form $A^{2h}(u^{2h}) = f^{2h}$

- Consider the coarse-grid equation:

$$\underbrace{A^{2h}(I_h^{2h} v^h + e^{2h})}_{u^{2h}} = \underbrace{A^{2h}(I_h^{2h} v^h) + I_h^{2h}(f^h - A^h(v^h))}_{f^{2h}}$$

coarse-grid unknown — all quantities are known

- We solve $A^{2h}(u^{2h}) = f^{2h}$ for $u^{2h} = I_h^{2h} v^h + e^{2h}$ & obtain

$$e^{2h} = u^{2h} - I_h^{2h} v^h$$

- 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:
  $v^{2h} = I_h^{2h} v^h \qquad r^{2h} = I_h^{2h}(f^h - A^h(v^h))$.
- Solve the coarse-grid equation:
  $A^{2h}(u^{2h}) = A^{2h}(v^{2h}) + r^{2h} = I_h^{2h} f^h + A^{2h}(I_h^{2h} v^h) - I_h^{2h} A^h(v^h)$.
- 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

$$A^{2h}(I_h^{2h}v^h + e^{2h}) = A^{2h}(I_h^{2h}v^h) + I_h^{2h}(f^h - A^h(v^h))$$

- If A is a linear operator, then FAS reduces directly to the linear two-grid correction scheme:

$$A^{2h}(I_h^{2h}v^h + e^{2h}) = A^{2h}(I_h^{2h}v^h) + I_h^{2h}(f^h - A^h(v^h))$$

- An exact solution to the fine-grid problem is a fixed point of the FAS iteration:

$$A^{2h}(I_h^{2h}v^h + \overset{0}{e^{2h}}) = A^{2h}(I_h^{2h}v^h) + I_h^{2h}(\overset{0}{f^h - A^h(v^h)})$$

A few more observations about FAS

$$A^{2h}(I_h^{2h}v^h + e^{2h}) = A^{2h}(I_h^{2h}v^h) + I_h^{2h}(f^h - A^h(v^h))$$

τ_h^{2h} ← A(Pv^h)-PA(v^h)

- The FAS coarse-grid equation can be written as

$$A^{2h}(u^{2h}) = f^{2h} + \tau_h^{2h}$$

where $\tau_h^{2h} = A^{2h}(I_h^{2h}v^h) - I_h^{2h}A^{2h}(v^h)$ is the so-called tau correction term & f^{2h} is the original 2h source term, provided you choose it that way: $f^{2h} = I_h^{2h}f^h$.

- In general, since $\tau_h^{2h} \neq 0$, 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 Ω^{2h} problem using Ω^{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
$$A^{2h}(u^{2h}) = I_h^{2h} A^{2h}(I_{2h}^h u^{2h})$$
where the action of the Galerkin product can be captured in an implementable formula.

The first method is usually easier & more common.

Example: Newton-MG vs. FAS

- PDE (er, ODE):

$$-u''(x) + u(x)\, e^{u(x)} = f(x).$$

- Discretization:

$$\frac{-v_{j-1} + 2v_j - v_{j+1}}{h^2} + v_j e^{v_j} = f_j.$$

One Newton-MG step

- **Step 0.** Given v, form the grid h linear correction equation:
$$\frac{-e_{j-1} + 2e_j - e_{j+1}}{h^2} + (1+v_j)e^{v_j} e_j = r_j \equiv f_j - \left(\frac{-v_{j-1} + 2v_j - v_{j+1}}{h^2} + v_j e^{v_j}\right)$$
unknowns
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:
$$\frac{-e_{j-1}^{2h} + 2e_j^{2h} - e_{j+1}^{2h}}{(2h)^2} + (1+v_{2j})e^{v_{2j}} e_j^{2h} = r_{2j} - \left(\frac{-e_{2j-1} + 2e_{2j} - e_{2j+1}}{h^2} + (1+v_{2j})e^{v_{2j}} e_{2j}\right)$$
unknowns
- **Step 3:** Correct the grid h Newton correction:
$$e \leftarrow e + I_{2h}^h e^{2h}$$
- **Step 4:** Stop if you've "solved" the linear equation well enough for Newton correction e & set v ← v + e. Else, leave v alone & return to **Step 1**.

One FAS step

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

$$\frac{-v_{j-1}+2v_j-v_{j+1}}{h^2}+v_j e^{v_j}=f_j$$

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

$$v_j \leftarrow v_j - \frac{\frac{-v_{j-1}+2v_j-v_{j+1}}{h^2}+v_j e^{v_j}-f_j}{\frac{2}{h^2}+(1+v_j)e^{v_j}}$$

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

$$\frac{-e^{2h}_{j-1}+2e^{2h}_j-e^{2h}_{j+1}}{(2h)^2}+(v^h_{2j}+e^{2h}_j)e^{v^h_{2j}+e^{2h}_j}-v^h_{2j}e^{v^h_{2j}}=(I^{2h}_h r^h)_j$$

unknowns

- **Step 3:** Correct the grid h approximation v:

$$v \leftarrow v + I^h_{2h} e^{2h}$$

Nonlinear problems: 2d example

- Consider

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

on the unit square, [0,1] x [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

$$\underbrace{\frac{1}{h^2}\begin{pmatrix} & -1 & \\ -1 & 4 & -1 \\ & -1 & \end{pmatrix}u^h_{i,j}+\gamma u^h_{i,j}e^{u^h_{i,j}}}_{(A^h(v^h))_{i,j}}=f_{i,j}$$

- 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 = 128

convergence factor is for the last cycle

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

	γ			
	1	10	100	1000
convergence factor	0.135	0.124	0.098	0.072
number of FAS cycles	12	11	11	10

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

	γ			
	1	10	100	1000
convergence factor	4.00E-05	7.00E-05	3.00E-04	2.00E-04
number of Newton iterations	3	3	3	4

Newton, Newton-MG, & FAS on

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

n = 128, γ = 10

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

Method	Outer iterations	Inner iterations	Megaflops
Newton	3		1660.6
Newton-MG	3	20	56.4
Newton-MG	4	10	38.5
Newton-MG	5	5	25.1
Newton-MG	10	2	22.3
Newton-MG	19	1	24.6
FAS	11		27.1

Compare FMG-FAS & FMG-Newton-MG

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

- We do one FMG cycle using one 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 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 $||r|| < 10^{-10}$.

Don't try this at home !!!

Compare FMG-FAS & FMG-Newton-MG

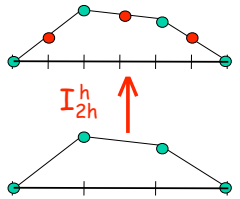
$$-\Delta u(x,y) + \gamma u(x,y)\, e^{u(x,y)} = f(x,y)$$

n = 128, γ = 10

Done !!!

Cycle	$\|r^h\|_\infty$	$\|u^{(k)} - v^h\|_\infty$	Mflops	$\|r^h\|_\infty$	$\|u^{(k)} - v^h\|_\infty$	Mflops	Cycle
FMG-FAS	1.10E-02	2.00E-05	3.1	1.06E-02	2.50E-05	2.4	FMG-Newton
FAS V	6.80E-04	2.40E-05	5.4	6.70E-04	2.49E-05	4.1	Newton-MG
FAS V	5.00E-05	2.49E-05	7.6	5.10E-05	2.49E-05	5.8	Newton-MG
FAS V	3.90E-06	2.49E-05	9.9	6.30E-06	2.49E-05	7.5	Newton-MG
FAS V	3.20E-07	2.49E-05	12.2	1.70E-06	2.49E-05	9.2	Newton-MG
FAS V	3.00E-08	2.49E-05	14.4	5.30E-07	2.49E-05	10.9	Newton-MG
FAS V	2.90E-09	2.49E-05	16.7	1.70E-07	2.49E-05	12.6	Newton-MG
FAS V	3.00E-10	2.49E-05	18.9	5.40E-08	2.49E-05	14.3	Newton-MG
FAS V	3.20E-11	2.49E-05	21.2	1.70E-08	2.49E-05	16.0	Newton-MG
				5.50E-09	2.49E-05	17.7	Newton-MG
				1.80E-09	2.49E-05	19.4	Newton-MG
				5.60E-10	2.49E-05	21.1	Newton-MG
				1.80E-10	2.49E-05	22.8	Newton-MG
				5.70E-11	2.49E-05	24.5	Newton-MG

Remembering coarse-grid correction



- 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 $\Rightarrow e = I_{2h}^h e^{2h}$:

  $AI_{2h}^h e^{2h} = r$. fewer unknowns

- Use transpose $I_h^{2h} = (I_{2h}^h)^T$ to reduce equations:

  $I_h^{2h} A I_{2h}^h e^{2h} = I_h^{2h} r$. fewer equations

  A^{2h}

Motivating FAS for nonlinear A

$A(v + e) = f \rightarrow A^{2h}(v^{2h} + e^{2h}) = \text{“}f^{2h}\text{”}$

$$A^{2h}(I_h^{2h} v^h + e^{2h}) = A^{2h}(I_h^{2h} v^h) + I_h^{2h}(f^h - A^h(v^h))$$

? ?

- 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' + …:

  $\underbrace{\gamma vv' - v''}_{A(v)} + \gamma (v'e + ve' + ee') - e'' = f.$

Differential residual equation

$\gamma (v'e + ve' + ee') - e'' = f - A(v) \equiv r(v)$

- 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:

  $e \rightarrow e_j^h, \quad e' \rightarrow (e')_j^h = (e_{j+1}^h - e_{j-1}^h)/(2h),$

  $e'' \rightarrow (e'')_j^h = (e_{j+1}^h - 2e_j^h + e_{j-1}^h)/h^2.$

Leading to FAS…

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

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

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

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

$$\gamma \frac{v_{2j+2}^h - v_{2j-2}^h}{4h} e_j^{2h} + \gamma v_{2j}^h \frac{e_{j+1}^{2h} - e_{j-1}^{2h}}{4h} + \gamma e_j^{2h} \frac{e_{j+1}^{2h} - e_{j-1}^{2h}}{4h} + \frac{-e_{j+1}^{2h} + 2e_j^{2h} - e_{j-1}^{2h}}{(2h)^2} = (I_h^{2h} r^h)_j$$

- FAS

$$A^{2h}(I_h^{2h} v^h + e^{2h}) - A^{2h}(I_h^{2h} v^h) = I_h^{2h}(f^h - A^h(v^h))$$

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!

7. Selected applications

7a. Neumann boundary conditions

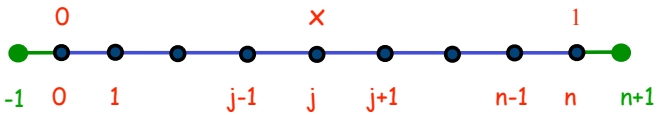
- Consider the 1-D problem

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

  $u'(0) = u'(1) = 0.$ ←???

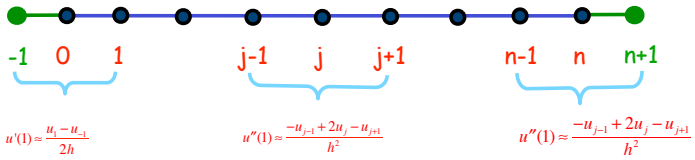
- We discretize on the interval [0,1] with h = 1/n grid spacing & nodes $x_j = jh$, j = 0,1,2, …, n.

- We extend the interval with two ghost points:



Central differences at boundary

- We use differences as before, but now also for the derivative in the Neumann condition:



- This yields the system

$$\frac{-u_{j-1} + 2u_j - u_{j+1}}{h^2} = f_j \qquad 0 \le j \le n$$

$$\frac{u_1 - u_{-1}}{2h} = \frac{u_{n+1} - u_{n-1}}{2h} = 0$$

Eliminating the ghost points

- Use the boundary conditions to eliminate u_{-1} & u_{n+1}:

$$\frac{u_1 - u_{-1}}{2h} = 0 \Rightarrow u_{-1} = u_1 \qquad \frac{u_{n+1} - u_{n-1}}{2h} = 0 \Rightarrow u_{n+1} = u_{n-1}$$

- Eliminating the ghost points in the j = 0 & j = n equations gives the (n+1)x(n+1) system of equations:

$$\frac{-u_{j-1} + 2u_j - u_{j+1}}{h^2} = f_j \qquad 1 \le j \le n-1$$

$$\frac{2u_0 - 2u_1}{h^2} = f_0 \qquad \frac{-2u_{n-1} + 2u_n}{h^2} = f_n$$

Write the system in matrix form

- We can write $A^h u^h = f^h$, where

$$A^h = \frac{1}{h^2}\begin{pmatrix} 2 & -2 & & & & \\ -1 & 2 & -1 & & & \\ & -1 & 2 & -2 & & \\ & & \ddots & \ddots & \ddots & \\ & & & -1 & 2 & -1 \\ & & & & -2 & 2 \end{pmatrix}.$$

- Note that A^h is (n+1)x(n+1) & nonsymmetric, & the system involves unknowns u_0^h & u_n^h at the boundaries.

We must consider compatibility

- The problem - u″(x) = f(x), for 0 < x < 1, with u′(0) =u′ (1) = 0, is not well-posed!
- If u(x) is a solution, then so is u(x) + constant.
- We cannot be certain a solution exists. If one does, it must satisfy

$$-\int_0^1 u''(x)\,dx = \int_0^1 f(x)\,dx \;\Rightarrow\; -[u'(1)-u'(0)] = \int_0^1 f(x)\,dx$$

$$\Rightarrow \boxed{0 = \int_0^1 f(x)\,dx}$$

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

The well-posed system

- The compatibility condition is necessary for a solution to exist. In general, it is also sufficient:

  $-\partial^2/\partial x^2$ is a well-behaved operator on the space of functions u(x) with zero mean.

- Thus, we may conclude that if f(x) satisfies the compatibility condition, then the problem is well-posed:

  - u″(x) = f(x), 0 < x < 1,
  u′(0) = u′(1) = 0,
  $\int_0^1 u(x)dx = 0$.

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

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 $f^h \in N((A^h)^T)^\perp$. ???
- For our simple case: $N((A^h)^T) = c(1/2, 1, 1, \ldots, 1, 1/2)^T$.
- Thus, $A^h u^h = f^h$ has a solution if & only if

  $f^h \perp c(1/2, 1, 1, \ldots, 1, 1/2)^T$.

- So, the discrete compatibility condition is

$$\boxed{\frac{1}{2}f_0^h + \sum_{j=1}^{n-1} f_j^h + \frac{1}{2}f_n^h = 0}$$

We have two issues to consider

- Solvability: A solution exists iff $f^h \in R(A) = N((A^h)^T)^\perp$.
- Uniqueness: If u^h is a solut??? en so is $u^h + v^h$ for any $v^h \in N(A^h)$.
- Note that if $A^h = (A^h)^T$, then $N((A^h)^T) = N(A^h)$ & solvability & uniqueness can be handled together.
- This is easily done. Multiply the first & last equations by 1/2, giving

$$A^h = \frac{1}{h^2}\begin{pmatrix} 1 & -1 & & & & \\ -1 & 2 & -1 & & & \\ & -1 & 2 & -2 & & \\ & & \ddots & \ddots & \ddots & \\ & & & -1 & 2 & -1 \\ & & & & -1 & 1 \end{pmatrix}.$$

The new system is symmetric

- We have the symmetric system $\hat{A}^h u^h = \hat{f}^h$:

$$\frac{1}{h^2}\begin{pmatrix} 1 & -1 & & & & \\ -1 & 2 & -1 & & & \\ & -1 & 2 & -1 & & \\ & & \ddots & \ddots & \ddots & \\ & & & -1 & 2 & -1 \\ & & & & -1 & 1 \end{pmatrix}\begin{pmatrix} u_0^h \\ u_1^h \\ u_2^h \\ \vdots \\ u_{n-1}^h \\ u_n^h \end{pmatrix} = \begin{pmatrix} f_0^h/2 \\ f_1^h \\ f_2^h \\ \vdots \\ f_{n-1}^h \\ f_n^h/2 \end{pmatrix}$$

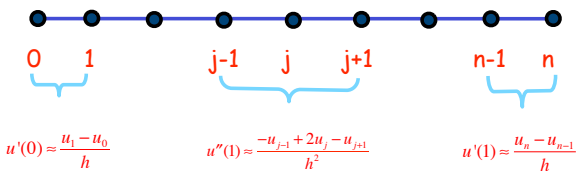
- Solvability is guaranteed by ensuring that $\hat{f}^h$ is orthogonal to the constant vector 1^h:

$$\left\langle \hat{f}^h, 1^h \right\rangle = \sum_{j=0}^{n} \hat{f}_j^h = 0$$

One-sided differences at boundary

a similar result

- No ghost points:



- This yields the system

$$\frac{-u_{j-1} + 2u_j - u_{j+1}}{h^2} = f_j \qquad 1 \le j \le n$$

$$\frac{u_1 - u_0}{h^2} = 0 \qquad \frac{u_n - u_{n-1}}{h^2} = 0$$

The well-posed discrete system

back to central differences @ boundary

- The (n+2)x(n+1) system is:

$$\frac{-u_{j-1} + 2u_j - u_{j+1}}{h^2} = f_j \qquad 1 \le j \le n-1$$

$$\frac{u_0 - u_1}{h^2} = \frac{f_0}{2} \quad \text{or} \quad 0$$

for one-sided scheme

$$\frac{-u_{n-1} + u_n}{h^2} = \frac{f_n}{2} \quad \text{or} \quad 0$$

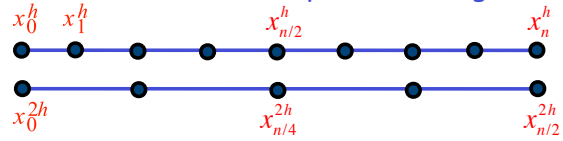
$$\sum_{j=0}^{n} u_i^h = 0 \qquad \text{(choose the zero mean solution)}$$

or, more simply

$$\hat{A}^h u^h = \hat{f}^h$$
$$\left\langle u^h, 1^h \right\rangle = 0$$

Multigrid for the Neumann problem

- We must have the interval endpoints on all grids



- Relaxation is performed at all points, including endpoints:

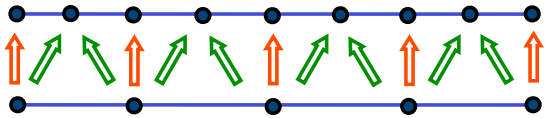
$$v_0^h \leftarrow v_1^h + h^2 \hat{f}_0^h \qquad v_j^h \leftarrow \frac{v_{j-1}^h + v_{j+1}^h + h^2 \hat{f}_j^h}{2} \qquad v_n^h \leftarrow v_{n-1}^h + h^2 \hat{f}_n^h$$

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

$$v^h \leftarrow v^h - \frac{\left\langle v^h, 1^h \right\rangle}{\left\langle 1^h, 1^h \right\rangle} 1^h$$

Interpolation must include the endpoints

We use linear interpolation:



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

Restriction also treats the endpoints

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

$$\hat{f}_0^{2h} = \frac{1}{2}\hat{f}_0^h + \frac{1}{4}\hat{f}_1^h$$

$$\hat{f}_j^{2h} = \frac{1}{4}\hat{f}_{2j-1}^h + \frac{1}{2}\hat{f}_{2j}^h + \frac{1}{4}\hat{f}_{2j+1}^h$$

$$\hat{f}_n^{2h} = \frac{1}{4}\hat{f}_{n-1}^h + \frac{1}{2}\hat{f}_n^h$$

The coarse-grid operator

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

$$\hat{A}^{2h} = I_h^{2h}\hat{A}^h I_{2h}^h.$$

$j^{2h} =$	0		1
ε_0^{2h}	1		0
$I_{2h}^h\varepsilon_0^{2h}$	1	$\frac{1}{2}$	0
$\hat{A}^h I_{2h}^h\varepsilon_0^{2h}$	$\frac{1}{2h^2}$	0	$-\frac{1}{2h^2}$
$I_h^{2h}\hat{A}^h I_{2h}^h\varepsilon_0^{2h}$	$\frac{1}{4h^2}$		$-\frac{1}{4h^2}$

$$\hat{A}^{2h} = \frac{1}{4h^2}\begin{pmatrix} 1 & -1 & & & & & \\ -1 & 2 & -1 & & & & \\ & -1 & 2 & -1 & & & \\ & & & \ddots & \ddots & & \\ & & & -1 & 2 & -1 & \\ & & & & -1 & 1 & \end{pmatrix}$$

Coarse-grid solvability

- Assuming $\hat{f}^h$ satisfies $\langle \hat{f}^h, 1^h \rangle = 0$, the solvability condition, we can show that theoretically the coarse-grid problem $\hat{A}^{2h}u^{2h} = I_h^{2h}(\hat{f}^h - \hat{A}^h v^h)$ 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 $\hat{f}^{2h}$ is generated for the coarse grid:

$$\hat{f}^{2h} \leftarrow \hat{f}^{2h} - \frac{\langle \hat{f}^{2h}, 1^{2h} \rangle}{\langle 1^{2h}, 1^{2h} \rangle} 1^{2h}.$$

Neumann problem: An example

Consider the problem

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

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

grid size n	$\|r^h\|_h$	average conv. factor	$\|u^{(h)} - v^h\|_h$	number of cycles
32	6.30E-11	0.079	9.70E-05	9
64	1.90E-11	0.089	2.40E-05	10
128	2.60E-11	0.093	5.90E-06	10
256	3.70E-11	0.096	1.50E-06	10
512	5.70E-11	0.100	3.70E-07	10
1024	8.60E-11	0.104	9.20E-08	10
2048	2.10E-11	0.112	2.30E-08	10
4096	5.20E-11	0.122	5.70E-09	10

V(2,1) cycles

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

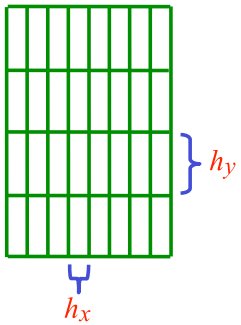
We consider two types of anisotropy

- Different coefficients on the derivatives

  $- u_{xx} - \varepsilon u_{yy} = f$

  discretized on a uniform grid with spacing h .

- Different mesh spacings:



$$h_x = h = \frac{1}{n}$$

$$h_y = \frac{h_x}{\sqrt{\varepsilon}}$$

Both problems lead to the same stencil

$$\frac{-u_{j-1,k} + 2u_{j,k} - u_{j+1,k}}{h^2} + \varepsilon \frac{-u_{j,k-1} + 2u_{j,k} - u_{j,k+1}}{h^2}$$

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

$$\frac{-u_{j-1,k} + 2u_{j,k} - u_{j+1,k}}{h^2} + \frac{-u_{j,k-1} + 2u_{j,k} - u_{j,k+1}}{\left(\frac{h}{\sqrt{\varepsilon}}\right)^2}$$

Why standard multigrid can fail

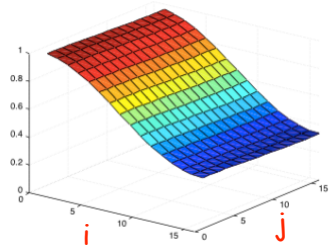
$$A^h = \frac{1}{h^2}\begin{pmatrix} & -\varepsilon & \\ -1 & 2+2\varepsilon & -1 \\ & -\varepsilon & \end{pmatrix}$$

- Note that A^h has weak connections in the y-direction. MG convergence factors degrade as ε gets small, with poor performance already at ε = 0.1.
- Consider the limiting case $\varepsilon \Rightarrow 0$: $A^h = \frac{1}{h^2}\begin{pmatrix} & 0 & \\ -1 & 2 & -1 \\ & 0 & \end{pmatrix}$
- Collection of disconnected 1-D problems!
- Point relaxation smoothes 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

$$\lambda_{i,j} = 1 - \frac{2\omega}{1+\varepsilon}\left(\sin^2\left(\frac{i\pi}{2n}\right) + \varepsilon\left(\sin^2\left(\frac{j\pi}{2n}\right)\right)\right).$$

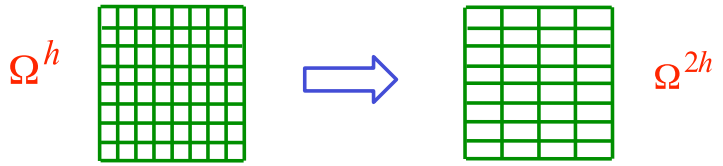


Two strategies for anisotropy

- **Semicoarsening:** The equations are weakly coupled in the y-direction, so we can't expect the error after point relaxation to have any connection to the errors above or below it. We therefore can & should coarsen only in the x-direction.
- **Line relaxation:** The equations are strongly coupled in the x-direction, so we could solve simultaneously for all the unknowns along lines of constant y. This should expose whatever weak smoothness there might be in the x-direction, which should allow standard coarsening.

Semicoarsening with point relaxation

- Point relaxation on $A^h = \frac{1}{h^2}\begin{pmatrix} & -\varepsilon & \\ -1 & 2+2\varepsilon & -1 \\ & -\varepsilon & \end{pmatrix}$ 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 Ω^{2h} is about half the number of points on Ω^{2h}, 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

$$v^h_{2j,k} = v^h_{2j,k} + v^{2h}_{j,k}$$

$$v^h_{2j+1,k} = v^h_{2j+1,k} + \frac{v^{2h}_{j,k} + v^{2h}_{j+1,k}}{2}$$

Line relaxation with full coarsening

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

- The other approach to this problem is to do the usual full coarsening, but to relax entire x-lines (constant y) of variables simultaneously.
- Consider an x-line equation specified by a fixed j:

$$\begin{matrix} & -v_{i,j+1} & \\ -v_{i-1,j} & +(2+2\varepsilon)v_{i,j} & -v_{i+1,j} = h^2 f_{i,j} \\ & -v_{i,j-1} & \end{matrix} \qquad 1 \le i \le n-1$$

$$\Rightarrow \quad -v_{i-1,j} + (2+2\varepsilon)v_{i,j} - v_{i+1,j} = h^2 f_{i,j} + v_{i,j+1} + v_{i,j-1}$$

Line relaxation

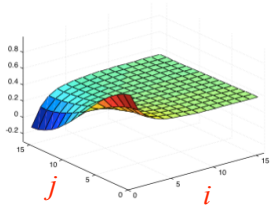
$$-v_{i-1,j} + (2+2\varepsilon)v_{i,j} - v_{i+1,j} = h^2 f_{i,j} + v_{i,j+1} + v_{i,j-1}$$

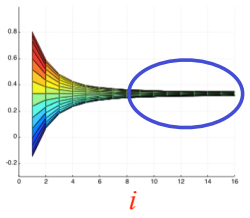
- Nice 1D system, analogous to the discretization of
  $-u'' + \alpha u = g, \qquad \alpha = 2\varepsilon h^{-2} > 0!$
- One sweep of line relaxation consists of solving a tridiagonal system for each constant y. Total cost is an optimal $O(n^2)$.
- Each solve can be done by Gaussian elimination since the system is tridiagonal, or a 1D multigrid solver (useful for generalization to higher dimensions).
- The individual lines can be solved simultaneously in a Jacobi way or sequentially in a Gauss-Seidel way.

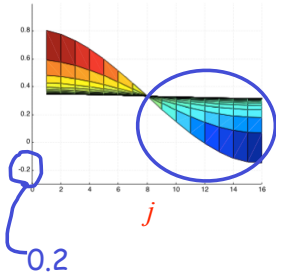
Why line relaxation works

Eigenvalues of the weighted block Jacobi iteration matrix:

$$\lambda_{i,j} = 1 - \frac{2\omega}{2\sin^2\left(\frac{i\pi}{2n}\right) + \varepsilon}\left(\sin^2\left(\frac{i\pi}{2n}\right) + \varepsilon\left(\sin^2\left(\frac{j\pi}{2n}\right)\right)\right).$$







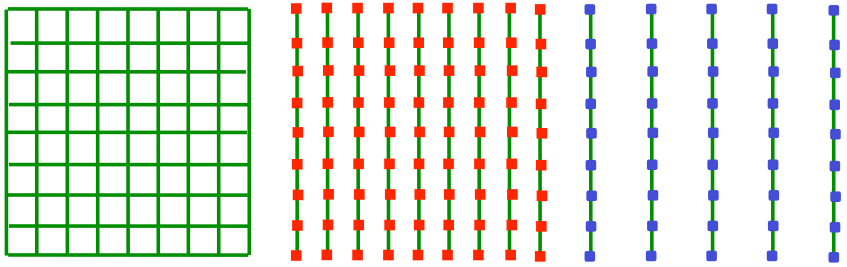
Semicoarsening & line relaxation

- We might not know the direction of weak coupling or it might vary over the domain.
- Suppose we want a method that can handle either

$$A_1^h = \frac{1}{h^2}\begin{pmatrix} & -\varepsilon & \\ -1 & 2+2\varepsilon & -1 \\ & -\varepsilon & \end{pmatrix} \quad \text{or} \quad A_2^h = \frac{1}{h^2}\begin{pmatrix} & -1 & \\ -\varepsilon & 2+2\varepsilon & -\varepsilon \\ & -1 & \end{pmatrix}$$

- 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.

Semicoarsening & line relaxation



- The original grid.
- Original grid viewed as a stack of "pencils." Line relaxation is used to solve problem along each "pencil".
- Coarsening is done by deleting every other pencil.

An anisotropic example

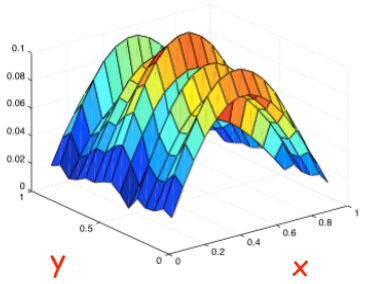
- Consider $-u_{xx} - \varepsilon u_{yy} = f$ with $u = 0$ on the boundaries of the unit square, & stencil given by

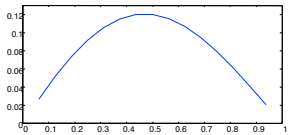
$$A^h = \frac{1}{h^2}\begin{pmatrix} & -\varepsilon & \\ -1 & 2+2\varepsilon & -1 \\ & -\varepsilon & \end{pmatrix}.$$

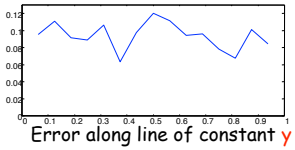
- Suppose that $f(x,y) = 2(y - y^2) + 2\varepsilon(x - x^2)$ so that the exact solution is $u(x,y) = (y - y^2)(x - x^2)$.
- Note: If ε is small, then the x-direction dominates, while if ε is large, then the y-direction dominates.

What is smooth error?

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




Error along line of constant x


Error along line of constant y

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:

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

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

How do the 3 methods work for various values of ε ?

n = 16

Asymptotic convergence factors of V(2,1)-cycles:

scheme	1000	100	10	ε = 1	0.1	0.01	0.001	1E-04
standard method	0.95	0.94	0.58	0.13	0.58	0.90	0.95	0.95
x-semi	0.94	0.99	0.98	0.93	0.71	0.28	0.07	0.07
x-semi & line relax	0.04	0.08	0.08	0.08	0.07	0.07	0.08	0.08

y-direction strong

x-direction strong

why is this bad???

Poisson

Note: semicoarsening in x works well for ε < 0.001 but degrades noticeably even at ε = 0.1.

A semicoarsening subtlety

- Suppose ε 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 $\varepsilon\, h_y^{-2}$, 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

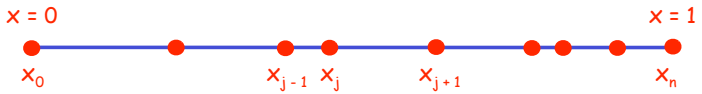
7c. Variable meshes

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

- Consider how we might approach the 1-D problem

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

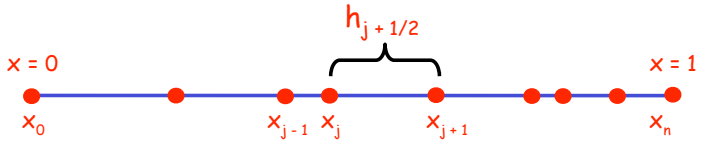
posed on the following nonuniform grid:



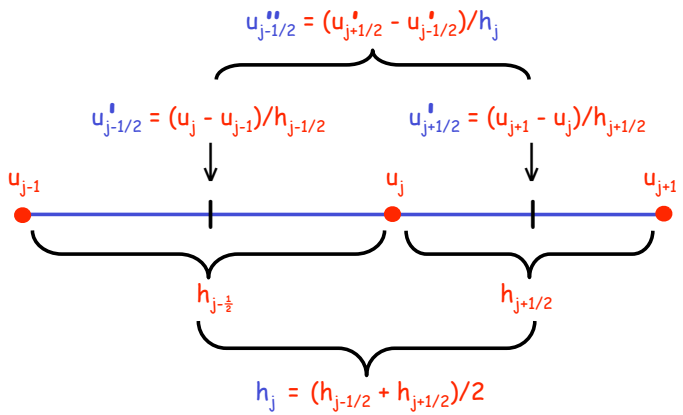
We need some notation for the mesh spacing

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

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



Building second divided differences



The discrete differential operator

- Using 2nd-order finite differences (& messy algebra!), we obtain the discrete representation

$$-\alpha_j^h u_{j-1}^h + (\alpha_j^h + \beta_j^h) u_j^h - \beta_j^h u_{j+1}^h = f_j^h \qquad 1 \le j \le n-1$$
$$u_0^h = u_n^h = 0$$

where

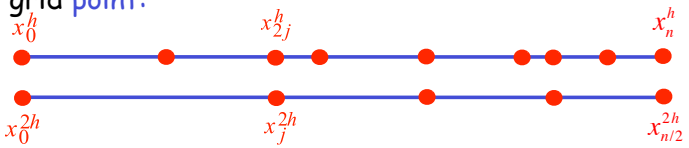
$$\alpha_j^h = \frac{2}{h_{j-\frac{1}{2}}\left(h_{j-\frac{1}{2}} + h_{j+\frac{1}{2}}\right)} \quad \& \quad \beta_j^h = \frac{2}{h_{j+\frac{1}{2}}\left(h_{j-\frac{1}{2}} + h_{j+\frac{1}{2}}\right)} .$$

- Multiplying by $(h_{j-1/2} + h_{j+1/2})/2$ yields an SPD matrix with stencil

$$A^h = \left[\; -\frac{1}{h_{j-\frac{1}{2}}} \quad \frac{1}{h_{j-\frac{1}{2}}} + \frac{1}{h_{j+\frac{1}{2}}} \quad -\frac{1}{h_{j+\frac{1}{2}}} \; \right] .$$

Modify interpolation for variable h

- We choose every other fine-grid point as a coarse-grid point:



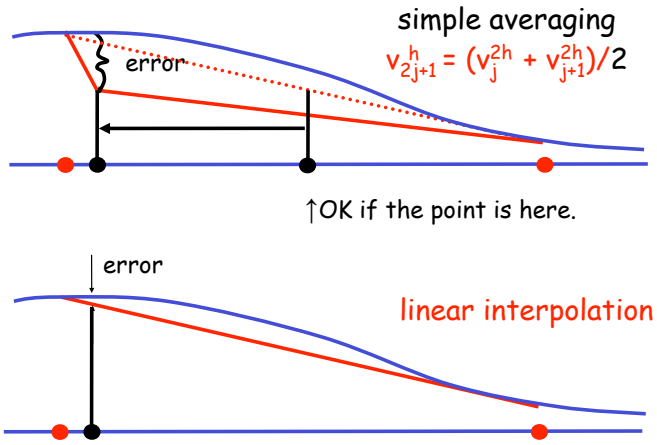
- In $[x_0^h, x_2^h]$, linear means

$$v(x) = v_0^{2h} + (v_1^{2h} - v_0^{2h})\,(x - x_0^h)/(x_2^h - x_0^h).$$

- Plug in $x = x_1^h$: writing $v^h = I_{2h}^h v^h$ yields

$$v_{2j}^h = v_j^{2h}, \quad v_{2j+1}^h = \frac{h_{2j+3/2}\, v_j^{2h} + h_{2j+1/2}\, v_{j+1}^{2h}}{h_{2j+1/2} + h_{2j+3/2}}, \qquad 1 \le j \le n/2-1.$$

Proper linear interpolation is needed

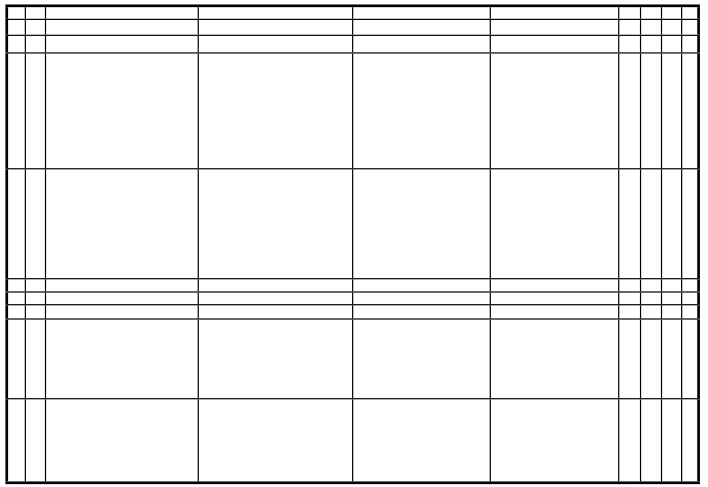


We use the variational properties to derive restriction & A^{2h}

$$A^{2h} = I_h^{2h} A^h I_{2h}^h \qquad I_h^{2h} = \frac{1}{2}(I_{2h}^h)^T$$

- This produces a stencil on Ω^{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.

Tensor-product grids?



7. Selected applications

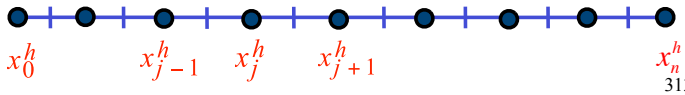
7d. Variable coefficients

- 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].

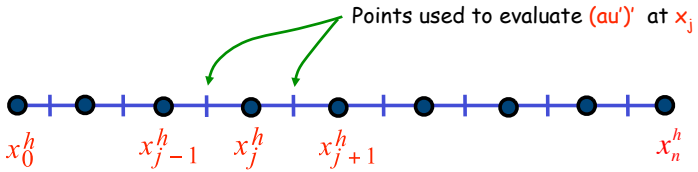
- We seek to develop a conservative, or self-adjoint, method for discretizing this problem.
- Assume we have available to us the values of a(x) ($a_{j+1/2} \equiv a(x_{j+1/2})$) at midpoints of the uniform grid



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:

$$(a(x)\,u'(x))'\Big|_{x_j} = \frac{(au')|_{x_{j+1/2}} - (au')|_{x_{j-1/2}}}{h} + O(h^2)$$



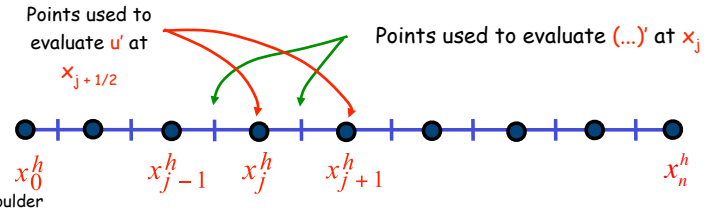
Discretize using central differences (cont'd)

To evaluate $(au')|_{x_{j+1/2}}$, we must sample a(x) at the point $x_{j+1/2}$ & use second-order differences:

$$(au')|_{x_{j+1/2}} \approx a_{j+1/2}\frac{u_{j+1}-u_j}{h} \qquad (au')|_{x_{j-1/2}} \approx a_{j-1/2}\frac{u_j-u_{j-1}}{h}$$

where

$$a_{j+1/2} \equiv a(x_{j+1/2}).$$



The basic stencil

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

$$-(a(x_j)u'(x_j))'(x_j) \approx -\frac{a_{j+\frac{1}{2}}\left(\frac{u_{j+1}-u_j}{h}\right) - a_{j-\frac{1}{2}}\left(\frac{u_j-u_{j-1}}{h}\right)}{h}$$

& the problem becomes

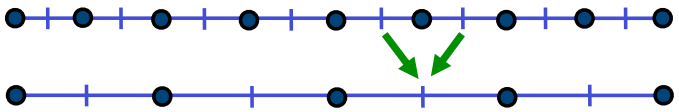
$$\frac{1}{h^2}\left(-a_{j-\frac{1}{2}}u_{j-1} + \left(a_{j-\frac{1}{2}} + a_{j+\frac{1}{2}}\right)u_j - a_{j+\frac{1}{2}}u_{j+1}\right) = f_j \qquad 1 \le j \le n$$
$$u_0 = u_n = 0\ .$$

Coarsening the variable coefficient problem

- A reasonable approach is to use a standard multigrid algorithm with linear interpolation, full weighting, & the stencil

$$A^{2h} = \frac{1}{(2h)^2}\left[\begin{array}{ccc} -a^{2h}_{j-\frac{1}{2}} & a^{2h}_{j-\frac{1}{2}} + a^{2h}_{j+\frac{1}{2}} & -a^{2h}_{j+\frac{1}{2}} \end{array}\right]$$

where $a^{2h}_{j+1/2} = \dfrac{a^h_{2j+1/2} + a^h_{2j+3/2}}{2}$.



- 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

$$-\frac{1}{h_{j-\frac{1}{2}}}u^h_{j-1} + \left(\frac{1}{h_{j-\frac{1}{2}}} + \frac{1}{h_{j+\frac{1}{2}}}\right)u^h_j - \frac{1}{h_{j+\frac{1}{2}}}u^h_{j+1} = \eta_j f^h_j$$

- Variable coefficients

$$\frac{1}{h}\left(-a_{j-\frac{1}{2}}u_{j-1} + \left(a_{j-\frac{1}{2}} + a_{j+\frac{1}{2}}\right)u_j - a_{j+\frac{1}{2}}u_{j+1}\right) = hf_j$$

- Correspondence

$$\frac{1}{h_{j-1/2}} \Leftrightarrow \frac{a_{j-1/2}}{h} \qquad \frac{1}{h_{j+1/2}} \Leftrightarrow \frac{a_{j+1/2}}{h}$$

A variable coefficient example

- We use V(2,1) cycle, full weighting, linear interpolation.
- We use a(x) = 1 + ρ sin(kπx) & a(x) = 1 + ρ rand(x).

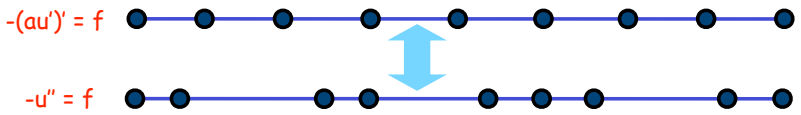
n = 1024

a(x) = 1 + ρ sin(kπx) a(x) = 1 + ρ rand(x)

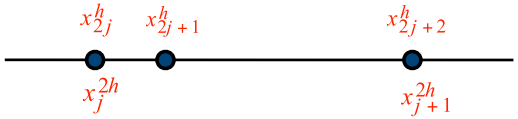
ρ	k=3	k=25	k=50	k=100	
0	0.085	0.085	0.085	0.085	0.085
0.25	0.084	0.098	0.098	0.094	0.083
0.5	0.093	0.185	0.194	0.196	0.173
0.75	0.119	0.374	0.387	0.391	0.394
0.85	0.142	0.497	0.511	0.514	0.472
0.95	0.191	0.681	0.69	0.694	0.672

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.



But simple averaging won't accurately represent smooth components if x^h_{2j+1} is close to x^h_{2j} but far from x^h_{2j+2}.



Pretend variability comes from mesh

$$A^h = \begin{bmatrix} -\frac{a_{j-\frac{1}{2}}}{h} & \frac{a_{j-\frac{1}{2}}}{h}+\frac{a_{j+\frac{1}{2}}}{h} & -\frac{a_{j+\frac{1}{2}}}{h} \end{bmatrix} = \begin{bmatrix} -\frac{1}{h_{j-\frac{1}{2}}} & \frac{1}{h_{j-\frac{1}{2}}}+\frac{1}{h_{j+\frac{1}{2}}} & -\frac{1}{h_{j+\frac{1}{2}}} \end{bmatrix}$$

- We can solve for the mesh sizes: $h_{j-\frac{1}{2}} = \frac{h}{a_{j-\frac{1}{2}}}, \quad h_{j+\frac{1}{2}} = \frac{h}{a_{j+\frac{1}{2}}}.$
- So linear interpolation yields

$$v^h_{2j+1} = \frac{h_{2j+3/2}v^{2h}_j + h_{2j+1/2}v^{2h}_{j+1}}{h_{2j+1/2}+h_{2j+3/2}} = \frac{a_{2j+1/2}v^{2h}_j + a_{2j+3/2}v^{2h}_{j+1}}{a_{2j+1/2}+a_{2j+3/2}}.$$

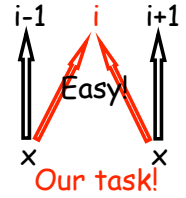
or, assuming that we've already interpolated to C-points 2j & 2i+2, it can be rewritten as

$$\left(\frac{a_{2j+1/2}}{h}+\frac{a_{2j+1/2}}{h}\right)v^h_{2j+1} - \frac{a_{2j+1/2}}{h}v^{2h}_j - \frac{a_{2j+3/2}}{h}v^{2h}_{j+1} = 0.$$

- We just used operator interpolation!
- Works for any stencil: $[-\alpha \quad \alpha+\beta \quad -\beta]$.

Operator interpolation

- Assume that the error is known on the coarse grid:



So we can assume that e is known on the fine grid at F-points i±1. We then just need to relate e_i to $e_{i\pm1}$.

i-1 ⟹ i ⟸ i+1

- Assume smooth error with the ansatz that r = 0.
- Applying this at point i : $-\alpha e_{i-1} + (\alpha+\beta)e_i - \beta e_{i+1} = 0.$
- Solving for i :

$$e_i = \frac{\alpha}{\alpha+\beta}e_{i-1} + \frac{\beta}{\alpha+\beta}e_{i+1}.$$

- Accidental: F-points connect only to C-points. What do we do otherwise???

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:

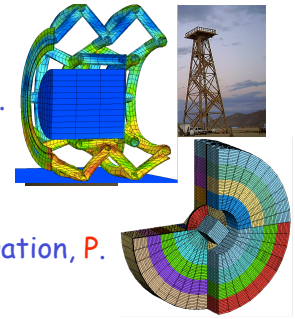
- √ Nonlinear Problems
  - Full approximation scheme
- √ Selected Applications
  - Neumann boundaries
  - Anisotropic problems
  - Variable meshes
  - Variable coefficients
- Algebraic Multigrid (AMG)
  - Matrix coarsening
- Multilevel Adaptive Methods
  - FAC
- Finite Elements
  - Variational methodology

Homework Due!

8. Algebraic multigrid (AMG)

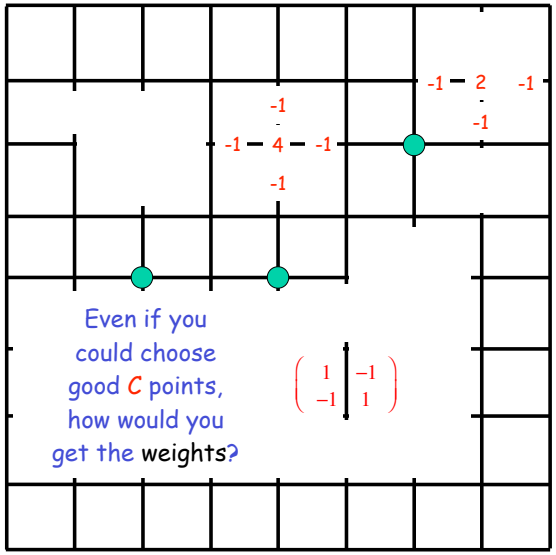
unstructured grids, variable coefficients,... assume SPD A

- Automatically determines coarsening.



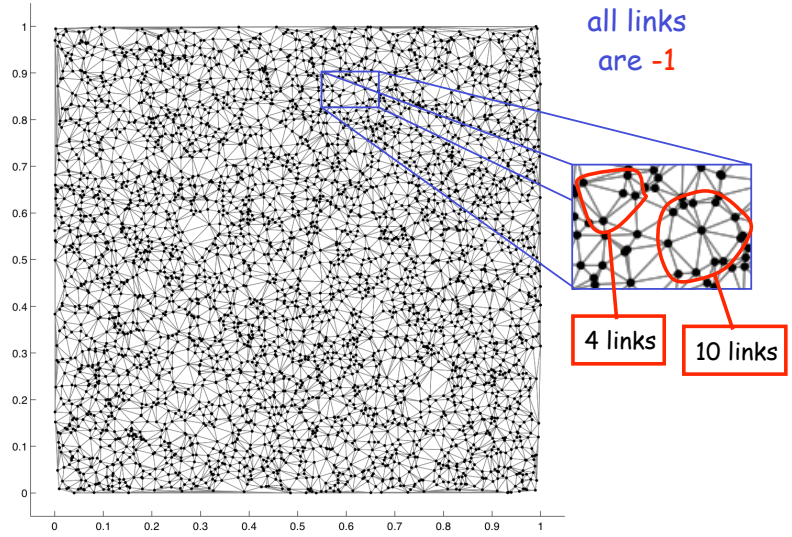
- AMG has two distinct phases:
  - setup phase: define MG components.
  - solution phase: perform MG cycles.
- AMG differs from geometric MG:
  - fix relaxation: point Gauss-Seidel.
  - choose coarsening: "grids" & prolongation, P.
- AMG principles:
  - algebraically smooth errors have small residuals: Ae ≈ 0.
  - "strong" connections mean good neighbors: good C-points.
  - smooth error is locally almost constant: e ≈ c for this A.
  - prolongation must match "smooth" error: e ∈ range(P).
  - variational conditions apply: given P, set $R=P^T$ & $A_c=RAP$.
  - only real task is to compute C & P: write e_F in terms of e_C.

Why AMG?



Even with this nice stencil, the geometry can give us trouble!

Graph Laplacian



AMG has two phases

- Setup Phase
  - Select coarse "grids," $\Omega^{m+1}, m = 1, 2, \ldots$
  - Define interpolation, $I_{m+1}^{m}, m = 1, 2, \ldots$
  - Define restriction & coarse-grid operators,

$$I_m^{m+1} = (I_{m+1}^m)^T, \quad A^{m+1} = I_m^{m+1} A^m I_{m+1}^m.$$

- Solve Phase

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

- All AMG processes parallelize well, although coarse-grid selection must be done with care.

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

  $(I - \omega D^{-1} A)\, e \approx e \;\Rightarrow\; \omega D^{-1} A\, e \approx 0$

$\Rightarrow$ $r \approx 0$

- A little more precisely, assuming that $\omega D \approx I$, then slow-to-converge error has relatively small energy:

$\langle Ae, e \rangle \ll \langle e, e \rangle ||A||$

AMG uses strong connection to determine MG components

- Smoothing assumption:

  $r \approx 0$ or $\langle Ae, e \rangle \approx 0$.

- We say that i is strongly connected to j if

$$-a_{ij} \geq \theta \max_{k \neq i}\{-a_{ik}\}, \qquad 0 < \theta \leq 1.$$

- Zero-row-sum "M-matrices" actually satisfy

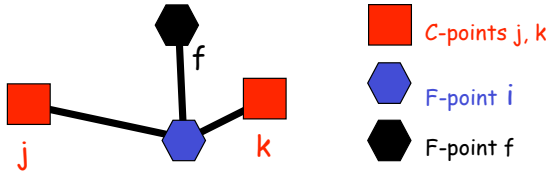
$$\langle Ae, e \rangle \approx \sum_{i \neq j} -\frac{a_{ij}}{2}(e_i - e_j)^2 \approx 0.$$

- So smooth error is more or less constant along strong connections.

We really mean matrices that have stencils like we've seen.

Operator-induced interpolation

assume a graph of A & given coarse points



$r_i = 0 \Rightarrow a_{ii} e_i + a_{ij} e_j + a_{ik} e_k + a_{if} e_f = 0.$ (i)

To define e_i in terms of e_j & e_k, we must eliminate e_f.

Erase it?!

Use the smoothness principle that e is locally almost constant.

$e_f = e_i \Rightarrow$ $(a_{ii} + a_{if})e_i + a_{ij} e_j + a_{ik} e_k = 0$

$\Rightarrow$ F-points determined by C-points!

We'll instead replace e_f by e_j & e_k in proportion to a_{fj} & a_{fk}.

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

$$S_i = \left\{ j : -a_{ij} \geq \theta \max_{k \neq i}\{-a_{ik}\} \right\}.$$

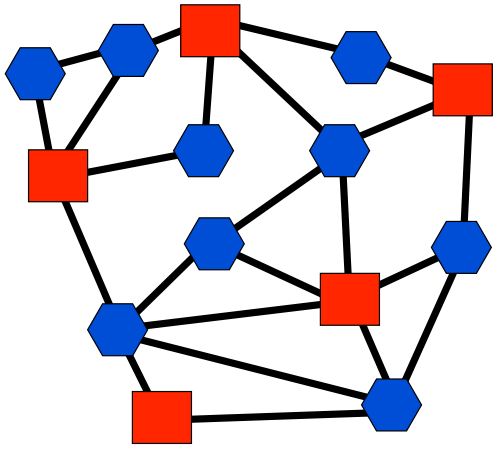
- The set of points strongly connected to variable u_i is denoted $S_i^T = \{ j : i \in S_j \}$.
- The set of coarse-grid variables is denoted C.
- The set of fine-grid variables is denoted F.
- The set of interpolatory coarse-grid variables used to interpolate the value of the fine-grid variable u_i is denoted C_i.

Choosing the coarse grid

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

choose C-point to allow most F-points ("value")



C-point selected (point with largest "value")

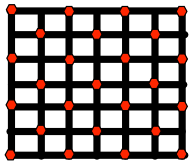
Neighbors of C-point become F-points

Next C-point selected (after updating "values")

F-points selected, etc.

Sample grids for the Laplacian

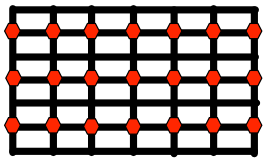
5-pt FD, 9-pt FE (quads), & 9-pt FE (stretched quads)



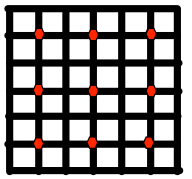
$$\begin{pmatrix} & -1 & \\ -1 & 4 & -1 \\ & -1 & \end{pmatrix}$$

5-pt FD

9-pt FE (stretched quads)



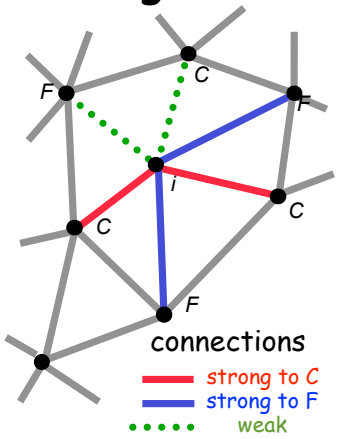
$$\begin{pmatrix} -1 & -4 & -1 \\ 2 & 8 & 2 \\ -1 & -4 & -1 \end{pmatrix}$$



$$\begin{pmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{pmatrix}$$

9-pt FE (quads)

Prolongation is based on smooth error, strong connections (from M-matrices)



Smooth error is given by:

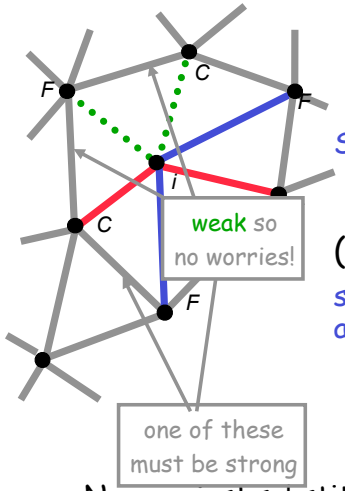
$$r_i = a_{ii}e_i + \sum_{j \in C \cup F} a_{ij}e_j \approx 0.$$

Prolongation :

$$(Pe)_i = \begin{cases} e_i, & i \in C \\ \sum_{k \in C} w_{ik}e_k, & i \in F \end{cases}.$$

Actually, we want to allow for the possibility that we don't interpolate from all of C…

Prolongation is based on smooth error, strong connections (from M-matrices)



C_i — Strongly connected C-pts.

D_i — Strongly connected F-pts.

Weak connections.

Smooth error at i means that

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

(C1) ⇒ j ~~is strongly connected to C_i~~,

so $e_j \approx e_k$ for $k \in C_i$ & we can write e_j as a properly scaled ~~sum of the e_k~~:

$$e_j = \frac{\sum_{k \in C_i} a_{jk}e_k}{\sum_{k \in C_i} a_{jk}}.$$

only if i is strongly connected to j , but a weak connection is no worries!

Now we just substitute to get interpolation weights!

Interpolation weights--the algebra

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

+

$$e_j = \frac{\sum_{k\in C_i} a_{jk}e_k}{\sum_{k\in C_i} a_{jk}} = \sum_{k\in C_i} \frac{a_{jk}}{\sum_{l\in C_i} a_{jl}} e_k$$

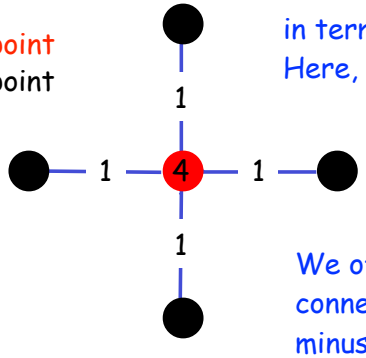
⇓

$$w_{ik} = -\frac{1}{a_{ii}}\left(a_{ik} + \sum_{j\in D_i} a_{ij}\frac{a_{jk}}{\sum_{l\in C_i} a_{jl}}\right)$$

Ideal setting

Suppose F-points are only connected to C-points.
Interpolation only care about smooth e, so assume $r_i = 0$.

● = F-point
● = C-point



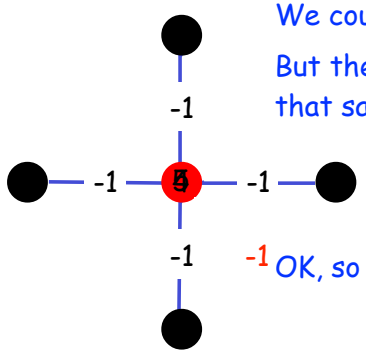
The stencil tells us what e_i is in terms of e_k at the C-points.
Here, $e_i = \Sigma(\frac{1}{4})\, e_k$.

We often just show connections without minus signs or diagonal.

Real setting

Suppose F-points are usually connected to other F-points.
How do we eliminate these F-F connections?

● = F-point
● = C-point



We could just cut it.

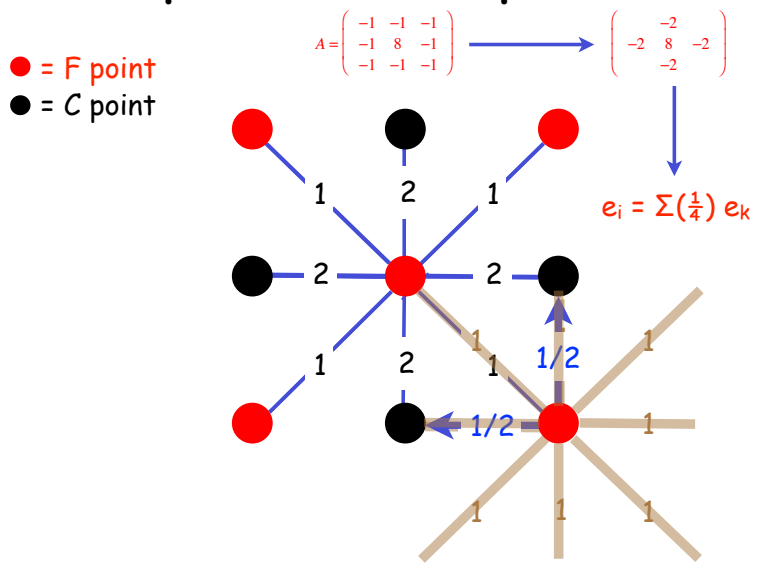
But then we get a stencil that says $e_i = \Sigma(\frac{1}{5})\, e_k$.

OK, so just change a_{ii}.

Can we do better?

- We now have a direct way to determine interpolation.
- We just clip F-F connections & then adjust the diagonal (denominator) to make the weights sum to 1.
- Observe that what we are doing here is trying to write an F-point as a combination of neighboring C-points in a way that reflects the nature of smooth error.
- But if we assume that F-points are in the minority, can we use this our crude direct interpolation idea to eliminate an F-F connection by replacing the offending F-point with a linear combination of C-points in i's neighborhood (C_i)?

Example of AMG compution of P



AMG setup costs

- 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.
- So AMG's setup phase is usually only 10-25% more expensive than in geometric MG.
- But AMG is more robust in terms of geometric difficulties.

AMG performance: Sometimes a success story

- AMG performs extremely well on the model problem (Poisson's equation, regular grid): optimal convergence factors (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 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

- Laplacian: $-u_{xx} - u_{yy} = 0$

$\begin{pmatrix} -1 & & -1 \\ & 4 & \\ -1 & & -1 \end{pmatrix}$
$\begin{pmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{pmatrix}$
$\begin{pmatrix} -1 & -4 & -1 \\ -4 & 20 & -4 \\ -1 & -4 & -1 \end{pmatrix}$

Stencil	Convergence per cycle	Operator Complexity	Time per cycle
5-pt	0.054	2.21	0.29
5-pt skew	0.067	2.12	0.27
9-pt (-1,8)	0.078	1.30	0.26
9-pt (-1,-4,20)	0.109	1.30	0.26

- Anisotropic 5-Point Laplacian: $-\varepsilon u_{xx} - u_{yy} = 0$

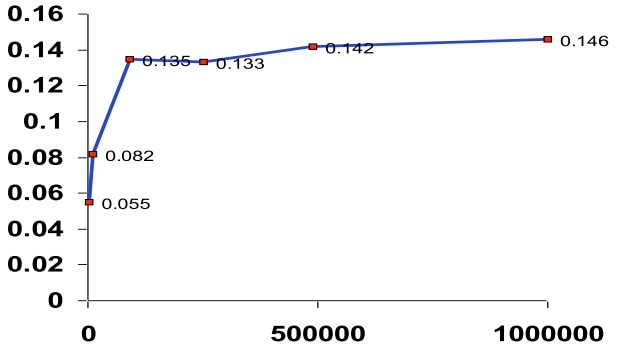
ε	0.001	0.01	0.1	0.5	1	2	10	100	1000
Convergence/cycle	0.084	0.093	0.058	0.069	0.054	0.079	0.087	0.093	0.083

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)

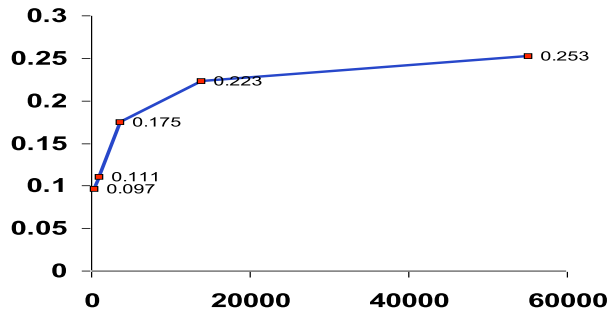


How does it perform (vol III)?

unstructured meshes, rectangular domains

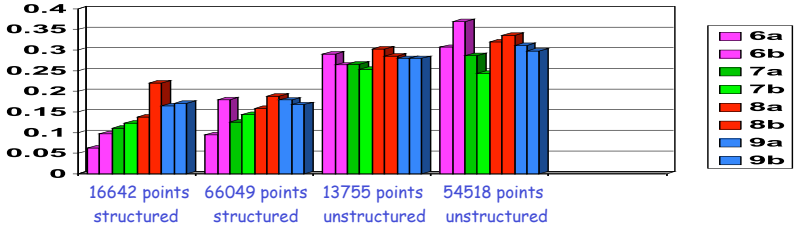
Laplacian on random unstructured grids (regular triangulations, 15-20% nodes randomly collapsed into neighboring nodes)

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



How does it perform (vol IV)?

$-\nabla\cdot(d(x,y)\nabla u)=0$ on structured, unstructured grids



Problems used: "a" means parameter c = 10, "b" means c = 1,000

6: $d(x,y) = 1.0 + c\,|x-y|$

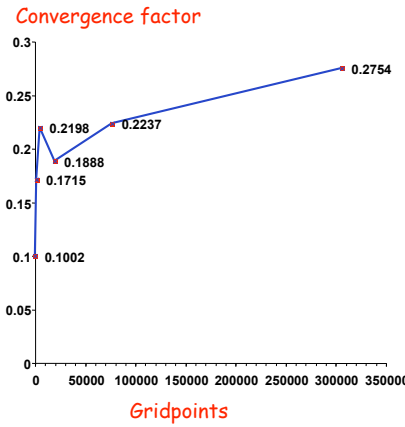
8: $d(x,y) = \begin{cases} 1.0 & 0.125 \le \max\{|x-0.5|, |y-0.5|\} \le 0.25 \\ c & \text{otherwise} \end{cases}$

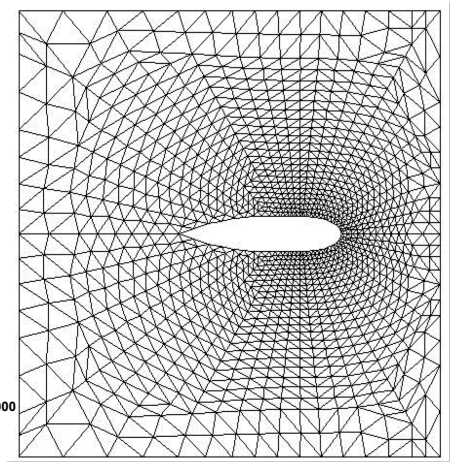
7: $d(x,y) = \begin{cases} 1.0 & x \le 0.5 \\ c & x > 0.5 \end{cases}$

9: $d(x,y) = \begin{cases} 1.0 & 0.125 \le \sqrt{(x-0.5)^2 + (y-0.5)^2} \le 0.25 \\ c & \text{otherwise} \end{cases}$

How does it perform (vol V)?

Laplacian operator, unstructured grids





Now for a glimpse at several other AMG topics...

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:

$$A^k = \begin{pmatrix} A_{11}^k & A_{12}^k \\ A_{21}^k & A_{22}^k \end{pmatrix} \quad \& \quad I_{k+1}^k = \begin{pmatrix} (I_{k+1}^k)_u & 0 \\ 0 & (I_{k+1}^k)_v \end{pmatrix}.$$

- This is called the "unknown-based" approach.
- 2-D biharmonic $(-\Delta)^2 u = f$, Dirichlet & Neumann boundaries, unit square, uniform quadrilateral mesh:

Mesh spacing	0.125	0.0625	0.03135	0.015625
Convergence factor	0.22	0.35	0.42	0.44

Adaptive AMG (αAMG)

to broaden applicability

adaptive interpolation

based on discovering the sense of smoothness

+

adaptive C-point choice

auto-determination of good coarse points

Standard interpolation

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

$$e_j = \frac{\sum_{k \in C_i} a_{jk} e_k}{\sum_{k \in C_i} a_{jk}}$$

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

Strong C F & Weak C

Isn't standard interpolation OK?

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

$$\underline{A} \leftarrow D^{-1/2} A D^{-1/2}, \quad D = \mathrm{diag}(a_{ii}).$$

- Relaxation still gives small residuals:

$$\underline{A}\,\underline{e} \approx 0.$$

- But

$$\underline{A}\,\underline{e} = (D^{-1/2} A D^{-1/2})\underline{e} \approx 0 \Rightarrow A(D^{-1/2}\underline{e}) \approx 0 \Rightarrow \underline{e} \approx D^{1/2} c.$$

- So "smooth" here means $\underline{e}_i \approx c\sqrt{a_{ii}}$. This could vary a lot!

How can we discover what smooth vectors actually look like?

Discovering smoothness

Relax on $Ax = 0$!!!

- What if we found a smooth error x that's far from c ?
- If, say, $x_j = 1.4 x_k$ for $j \in D_i$ & all $k \in C_i$, then we could set

$$e_j \rightarrow \Big(\sum_{k \in C_i} a_{jk}\, 1.4 e_k\Big) \Big/ \Big(\sum_{k \in C_i} a_{jk}\Big).$$

- If x_j / x_k varies with k, then it's just a bit more complicated.

$\begin{pmatrix} -1 & -4 & -1 \\ 2 & 8 & 2 \\ -1 & -4 & -1 \end{pmatrix}$ Adaptive C-point choice

- What does a "good" C mean?
- We want each i in F to "depend" on C :

  e_k given $\forall\, k \in C \Rightarrow e_i$ well determined $\forall\, i \in F$.

Systems???
Irregular grids???
Variable coeffs???

- Let's look at the matrix: $A = \begin{pmatrix} A_{ff} & A_{fc} \\ A_{cf} & A_{cc} \end{pmatrix}$.
- What do we know about smooth e? $r_f = A_{ff} e_f + A_{fc} e_c = 0$.
- Given e_c, is e_f well determined? Does $r_f = 0$ determine e_f ?
- Given e_c, when does $r_f = 0$ not determine e_f ?
- What if A_{ff} is singular?
- Then e_f cannot fully depend on e_c .
- We want A_{ff} to be well conditioned!!! How can we tell?

Compatible relaxation (CR)

- To ensure that C is a really good set of coarse grid points, we want A_{ff} to be well conditioned. Thus, we can assess whether we have good C-points by "CR":

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

- Fast CR means that F depends on C in the sense that smooth error ($r_f = 0$) is quickly recovered from C.
- Fast CR also has the benefit that the F-point residuals can be made really small after F-point relaxation.
- We can also show that fast CR means that a P exists that gives good MG convergence: $-A_{ff}^{-1} A_{fc}$.
- This P isn't local; hopefully, CR can give us a good local P.

Problem: If A_{ff} isn't well conditioned, what then???

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

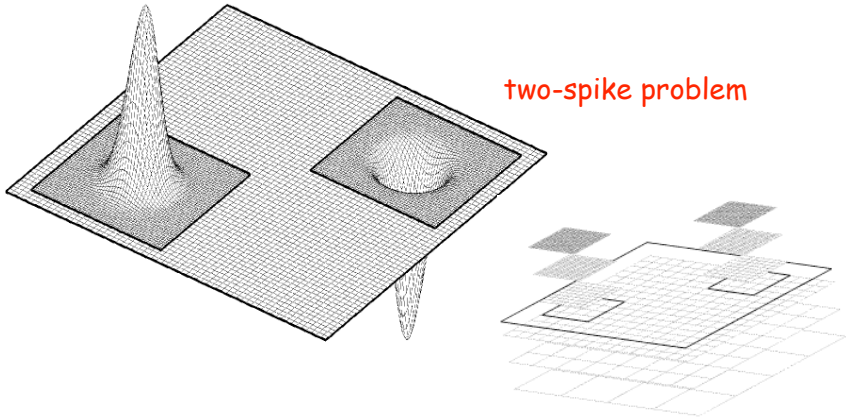
Chapters 6-10:

- √ 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 !

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

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

Ω^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

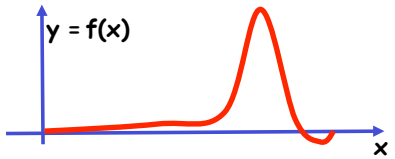
0 ——————————— 1

$$h^{-2}(-u_{i-1}^h + 2u_i^h - u_{i-1}^h) = f_i^h, \qquad 1 \le i \le 7$$
$$u_0^h = u_8^h = 0, \qquad h = \frac{1}{8}$$

Local refinement

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

Unneeded resolution & accuracy.



Ω^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

fine grid (● = interface)

$x =$ 0 $\frac{1}{8}$ $\frac{1}{4}$ $\frac{3}{8}$ $\frac{1}{2}$ $\frac{5}{8}$ $\frac{3}{4}$ $\frac{7}{8}$ 1

Ω^{2h} ⊙ × × × ⊙ coarse grid

Ω^{2h} x_0^{2h} x_1^{2h} x_2^{2h} x_3^{2h} x_4^{2h}

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].
  - 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

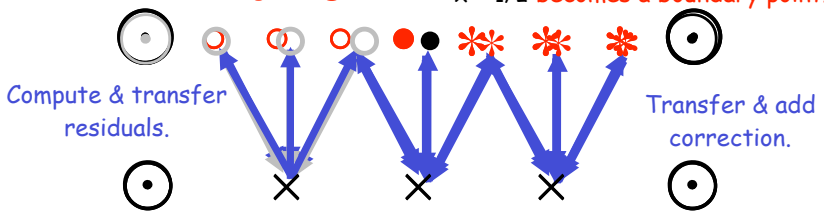
→ → WARNING: You won't get zero residuals on all of grid h. ← ←

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

Relax only in the local region.

Consider the usual global grids as usual.

x = 1/2 becomes a boundary point.



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

notation $v^h = (v_i^h)$

Ω^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

Ω^{2h} x_0^{2h} x_1^{2h} x_2^{2h} x_3^{2h} x_4^{2h}

Once & forall: it changes only on 2h.

- 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 $r^h = f^h - A^h v^h$ (@x_3^h-x_7^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^{2h}, to the solution of the 2h residual equation, $A^{2h} u^{2h} = f^{2h}$.
- Update the residual at x_1^{2h} for later cycles:
  $f_1^{2h} \leftarrow f_1^{2h} - (-v_0^{2h} + 2v_1^{2h} - v_2^{2h})/(2h)^2$.
- Correct: $v^h \leftarrow v^h + I_{2h}^h v^{2h}$.

cycle

Why save v^h outside of local region?

Eliminate more

Ω^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

Ω^{2h} x_0^{2h} x_1^{2h} x_2^{2h} x_3^{2h} x_4^{2h}

f_2^{2h} involves r_3^h & r_4^h, so we "need" v_2^h-v_7^h.

Store v^h only @ x_3^h-x_7^h & save v_2^h on 2h (call it w_1^{2h}).

- Initialize $v^h = 0$, $w_1^{2h} = 0$, & $f_1^{2h} \leftarrow (f_1^h + 2f_2^h + f_3^h)/4$.
- Relax on v^h on the local fine grid (x_5^h, x_6^h, x_7^h).
- Compute $r^h = f^h - A^h v^h$ (@x_2^h-x_7^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^{2h}, to the solution of the 2h residual equation, $A^{2h} u^{2h} = f^{2h}$.
- Update the residual at x_1^{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: $w_1^{2h} \leftarrow w_1^{2h} + v_1^{2h}$.
- Correct: $v^h \leftarrow v^h + I_{2h}^h v^{2h}$ (@x_3^h-x_7^h).

cycle

Eliminate the rest

Ω^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

Ω^{2h} x_0^{2h} x_1^{2h} x_2^{2h} x_3^{2h} x_4^{2h}

r_3^h doesn't change on h.

Compute change in r_3^h on 2h.

2h residual @ $x_2^{2h} = g_2^{2h} - (- w_1^{2h} + 2 v_4^h - v_6^h)/(2h)^2$,
where $g_2^{2h} = (f_3^h + 2f_4^h + f_5^h)/4$. (Messy algebra!)

- Initialize $v^h = 0$, $w_1^{2h} = 0$, & $f_1^{2h} \leftarrow (f_1^h + 2f_2^h + f_3^h)/4$.
- Relax on v^h on the local fine grid (x_5^h, x_6^h, x_7^h).
- Compute the right sides for 2h:
  $f_2^{2h} \leftarrow g_2^{2h} - (- w_1^{2h} + 2 v_4^h - v_6^h)/(2h)^2$ &
  $f_3^{2h} \leftarrow (r_5^h + 2r_6^h + r_7^h)/4$.
- Compute an approximation, v^{2h}, to the solution of the 2h residual equation, $A^{2h} u^{2h} = f^{2h}$.
- Update the residual at x_1^{2h} for later cycles:
  $f_1^{2h} \leftarrow f_1^{2h} - (- v_0^{2h} + 2 v_1^{2h} - v_2^{2h})/(2h)^2$.
- Accumulate the 2h approximation: $w_1^{2h} \leftarrow w_1^{2h} + v_1^{2h}$.
- Correct: $v^h \leftarrow v^h + I_{2h}^h v^{2h}$ (@x_4^h-x_7^h).

cycle

Interpretation

composite grid

Ω^h • * * * ⊙

Ω^{2h} ⊙ × × × ⊙

Ω^c ⊙ × • * * * ⊙

v_0^c v_1^c v_2^c v_3^c v_4^c v_5^c v_6^c

What equation in v^h are we solving?

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

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

} usual

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

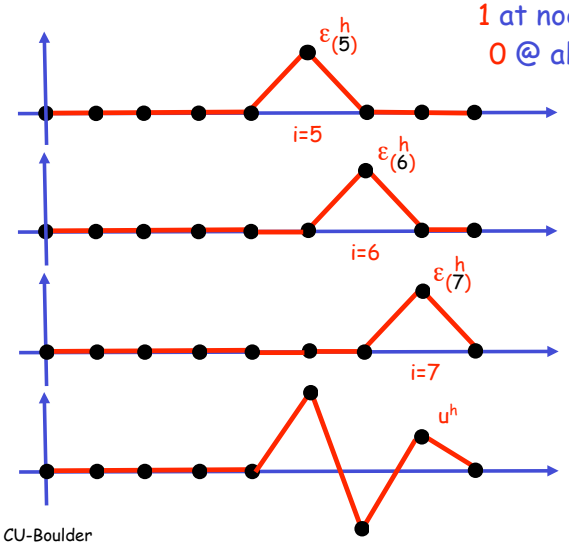
→ → The key is to treat the interface correctly. ← ←

Finite Element Local Refinement

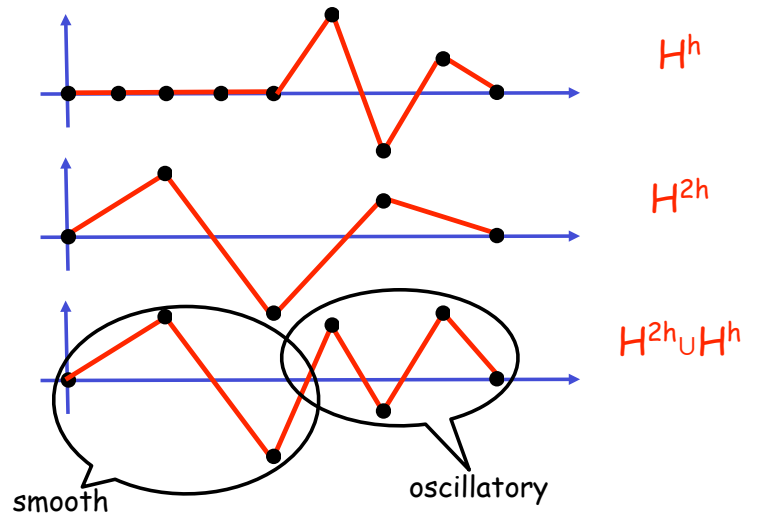
continuous piecewise linear:

1 at node i=5,6,7 only,

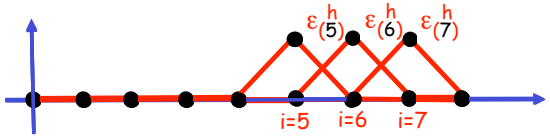
0 @ all other nodes.



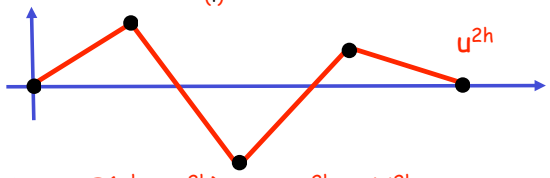
Local Fine-Grid Refinement



Abstract FE relaxation & 2h correction



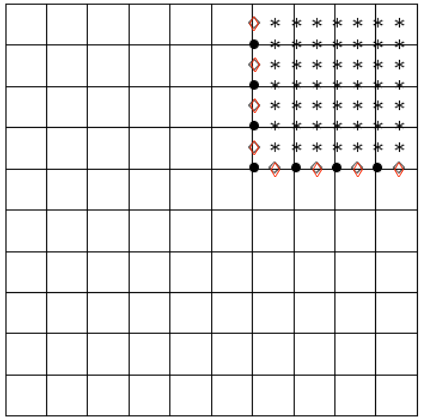
- Minimize $F(u^h - s\,\varepsilon^h_{(i)})$ over s for i = 5,6,7 in turn.



- Minimize $F(u^h + u^{2h})$ over u^{2h} in H^{2h}.

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.

Global h	Convergence factor	Discrete L^2 norm of discretization error
1/32	0.362	2.34e-2
1/64	0.367	5.742-3
1/128	0.365	1.43e-3

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

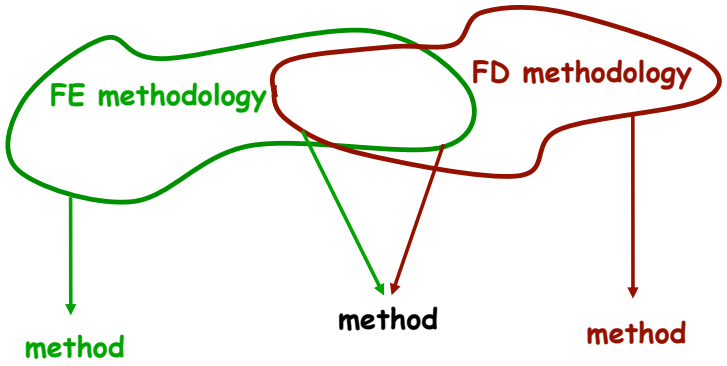
Chapters 6-10:

- √ 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 !

10. Finite elements

FE, a variational discretization methodology



FD: differences at nodes & truncation error.

FE: weak form on discrete functions & approximation property.

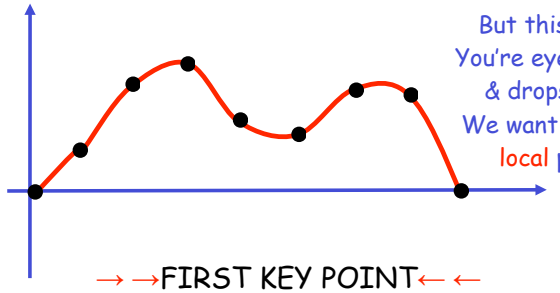
Other methods: finite volume, collocation, spectral, …

Discretization

This is a subspace view.
These discrete functions are in the space of solutions.

What do you see?

Discrete because they are determined by a finite number of nodes

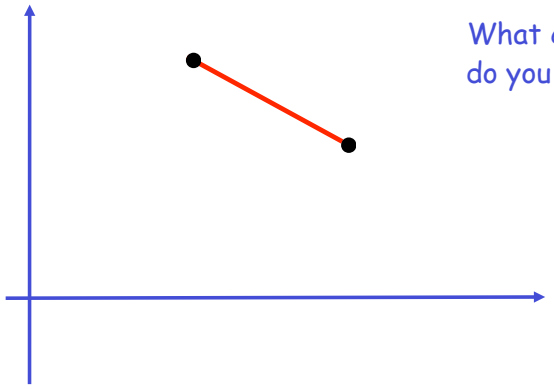


But this is gobal. You're eyes see rises & drops coming. We want to capture local physics.

→ →FIRST KEY POINT← ←

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

Localize



What choice do you have?

→ →SECOND KEY POINT← ←

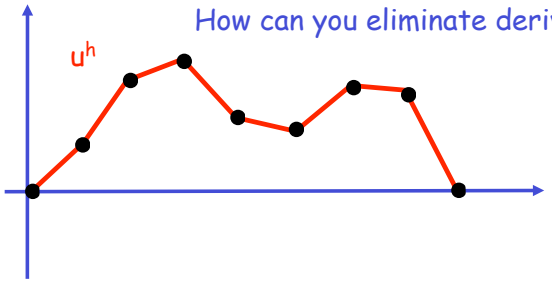
FE functions are localized to ensure a sparse matrix.

Continuous piecewise linear functions

??? - u″ = f ???

How do you take 2nd derivatives of u^h ?

How can you eliminate derivatives?



→ →THIRD KEY POINT← ←

FE integrates away 2nd derivatives.

Weak form!

$-u'' = f$ $\quad u =$ trial function

$\Rightarrow \; -u''v = fv \quad \forall v \quad v =$ test function

$\Rightarrow \; \int(-u'')v\,dx = \int fv\,dx \quad \forall v$

$\Rightarrow \; \int u'v'\,dx - u'v\,|_0^1 = \int fv\,dx \quad \forall v$

$\Rightarrow \; \int u'v'\,dx = \int fv\,dx \quad \forall v \ni v(0) = v(1) = 0$

$\Rightarrow \; \boxed{(u',v') = (f,v) \quad \forall v \ni v(0) = v(1) = 0}$

How is this weaker? Hint: 2 ways.

Why we have $u^h \in H^1$!!!

How would we discretize this?

? Need basis for space of admissible u^h & v.

$\Rightarrow$ can actually be reversed to

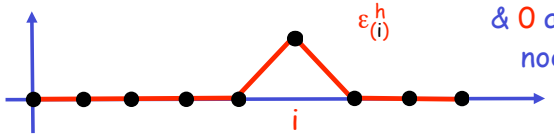
$\Leftarrow$ if u is smooth enough.

So the weak & strong forms are "equivalent".

Representation

a basis of "hat" functions

$\varepsilon^h_{(i)}$ is continuous piecewise linear & 0 at all nodes except node i where it's 1.



Any continuous piecewise linear function can be represented by

$u^h = \Sigma_i u^h_i \varepsilon^h_{(i)}$.

Now we're back to using node values.

Weak form & Galerkin discretization

$(u',v') = (f,v) \quad \forall v \ni v(0) = v(1) = 0.$

$(\sum_j u^h_j \varepsilon^h_{(j)}{}', v') = (f,v) \quad \forall v \ni v(0) = v(1) = 0.$

$(\sum_j u^h_j \varepsilon^h_{(j)}{}', \varepsilon^h_{(i)}{}') = (f, \varepsilon^h_{(i)}) \quad \forall i.$

$\sum_j (\varepsilon^h_{(i)}{}', \varepsilon^h_{(j)}{}') u^h_j = (f, \varepsilon^h_{(i)}) \quad \forall i.$ $\quad \boxed{A^h u^h = f^h}$

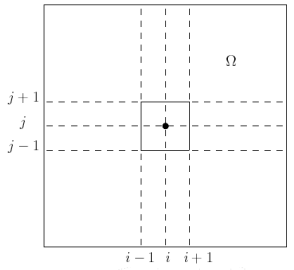
So the unknown is $\boxed{u^h = (u^h_j)}$,

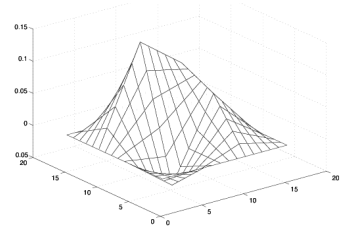
the matrix is $\boxed{A^h = (a^h_{ij}) = ((\varepsilon^h_{(i)}{}', \varepsilon^h_{(j)}{}'))}$, & the right side is $\boxed{f^h = (f^h_i) = ((f, \varepsilon^h_{(i)}))}$.

2D FE constructs

hat function

- Assume that Ω is the unit square.
- Consider an nxn grid of square "cells".
- Continuous piecewise bilinear elements: $H^h \subset H_0^1(\Omega)$.
- Each u^h in H^h is determined by its node values. This is how we'll represent them!
- Within each square:
  $u^h = axy + bx + cy + d$
  (u^h is linear in each coordinate direction).
- If u^h on one side of an element matches u^h on the other at the nodes, then we want to know that it matches on the common edge so that u^h is continuous: we want to know that specifying a piecewise bilinear function at the nodes gives us continuity.



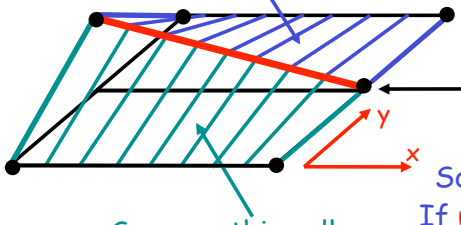


Why do we care???

Continuity

If
$u^h = axy + bx + cy + d$
in this cell,

Linear!!!
then $u^h = (ay_0 + b)x + (cy_0 + d)$
on this $y = y_0$ edge.



Same on this cell.

So u^h is linear on the edge.
If u^h above & below the edge have the same values at the ●, then u^h must be the same linear function on the edge.

Weak form

- The Gauss Divergence Theorem & homogeneous boundary conditions yield

  $(Lu, v) = (- u_{xx} - u_{yy}, v) = (-\nabla\bullet\nabla u, v) = (\nabla u, \nabla v)$.

- Note:

  $\nabla u = \begin{pmatrix} u_x \\ u_y \end{pmatrix}$

  $(\nabla u, \nabla v) = \int_\Omega (u_x v_x + u_y v_y) d\Omega$.

- So the problem becomes

  $(\nabla u^h, \nabla v^h) = (f, v^h) \quad \forall\, v^h \in H^h$

  or

  $\int_\Omega (u^h_x v^h_x + u^h_y v^h_y) d\Omega = \int_\Omega f v^h d\Omega \quad \forall\, v^h \in H^h$.

Towards the matrix equation

for nodal values

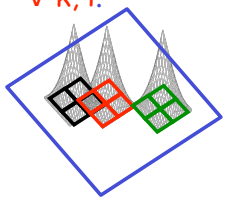
$$(\nabla u^h, \nabla v^h) = (f, v^h) \quad \forall\, v^h \in H^h$$

- Using $u^h(x,y) = \Sigma_{ij}\, u^h_{ij}\, \varepsilon^h_{(ij)}(x,y)$ & choosing $v^h = \varepsilon^h_{(kl)}$:

  $\Sigma_{ij}\, u^h_{ij}\, (\nabla\varepsilon^h_{(ij)}, \nabla\varepsilon^h_{(kl)}) = (f, \varepsilon^h_{(kl)}) \quad \forall\, k, l.$

- Matrix terms $(\nabla\varepsilon^h_{(ij)}, \nabla\varepsilon^h_{(kl)})$ are 0 when

  $|i - k|$ or $|j - l| > 1$.



- We compute

  $(\nabla\varepsilon^h_{(ij)}, \nabla\varepsilon^h_{(ij)}) = 8/3$ & $(\nabla\varepsilon^h_{(ij)}, \nabla\varepsilon^h_{(i\pm1\, j\pm1)}) = -1/3$.

- Assume f is fairly smooth locally:

  $(f, \varepsilon^h_{(kl)}) = \int_\Omega f\, \varepsilon^h_{(kl)} d\Omega \approx h^2 f(x_{kl})$.

$$u^h = (u^h_{ij}) \quad \& \quad f^h = (h^2 f(x_{kl}))$$

actually,
$B^h f^h$
(mass matrix)

The matrix equation

$$A^h u^h = f^h$$

where

$$u^h = (u^h_{ij}) \quad \& \quad f^h = (h^2 f(x_{kl}))$$

& the matrix is given by the stencil

$$A^h_{ij} = \frac{1}{3}\begin{pmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{pmatrix}.$$ stiffness 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±1j±1 as i±1j±1 to ij.
- Singular? $A^h 1 = 0$?! Depends on boundaries!

Dirichlet west boundary:

$$A^h_{ij} = \begin{pmatrix} 0 & & & \\ & -1 & -1 & \\ & -1 & 8 & -1 \\ & -1 & -1 & \end{pmatrix}.$$

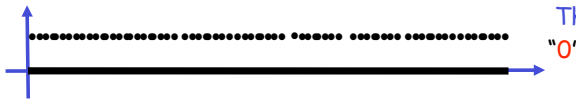
- Positive definite!

Diagonally dominant (strictly so @ boundaries).

But now we need to understand the PDE better, starting with choosing our universe of functions…

A word about Sobolev spaces

- We're mucking about with forms like $\int_\Omega(u_x v_x + u_y v_y)d\Omega$, so we need to know that derivatives of functions in our universe can be multiplied together & integrated. It's enough to have $\int_\Omega(u_x^2 + u_y^2)d\Omega < \infty$, so our universe is $H_0^1(\Omega) = \{u : u, u_x, u_y \in L^2(\Omega), u|_{\partial\Omega} = 0\}$, where $L^2(\Omega) = \{u : \int_\Omega u^2 d\Omega < \infty\}$.
- We want what we compute to get close to what we want, so we're concerned about convergence in our universe: we need to know that limits of things that satisfy $\int_\Omega(u_x^2 + u_y^2)d\Omega < \infty$ also satisfy it, that is, stay in our universe (completeness). Even if you start with nice continuous differential functions, you are led to some strange ones in your space.
- Think of $L^2(\Omega) = \{u : \int_\Omega u^2 d\Omega < \infty\}$. If u = 0 except at finitely many points (say, u (i/n, j/n) = 1 for i, j = 1, 2,…, n,), then $\int_\Omega u^2 d\Omega = 0$, so u = "0" ! This is true for any finite n, so it's true for a countable infinity of nonzeros!



This is what "0" looks like!

FE twin towers

$(\nabla u, \nabla v) = (f, v)$ weak form vs. $(\nabla u, \nabla u)/2 - (f, u)$ functional

- Model problem $\Omega \subset \Re^2$:
  $Lu \equiv -u_{xx} - u_{yy} = f$ in Ω
  $u = 0$ on $\partial\Omega$
- Sobolev spaces:
  $L^2(\Omega) = \{u : \int_\Omega u^2 d\Omega < \infty\}$
  $H_0^1(\Omega) = \{u : u, u_x, u_y \in L^2(\Omega), u|_{\partial\Omega} = 0\}$
  $(u, v) = \int_\Omega u\, v\, d\Omega$
- L is self-adjoint positive definite (more later):
  "(Lu, v)" = $(\nabla u, \nabla v) = \int_\Omega(u_x v_x + u_y v_y)d\Omega$ = "(Lv, u)"
  "(Lu, u)" > 0 if $u \neq 0$: $\int_\Omega(u_x^2 + u_y^2)d\Omega \Rightarrow u = c$, but $u|_{\partial\Omega} = 0$.

We use "(Lu, u)" for simplicity, but we really mean $(\nabla u, \nabla v)$.

Possible because L is self-adjoint.

- Duality: Solving Lu = f is equivalent to minimizing the weak functional
  F(u) = "(Lu, u)"/2 - (f, u)
  = $(\nabla u, \nabla u)/2$ - (f, u)
- Short story:
  1st derivative test
  $\nabla F(u) = Lu - f = 0$
  2nd derivative test
  $F''(u) = L > 0$

This is formal: Lu is not defined on all of $H_0^1(\Omega)$.

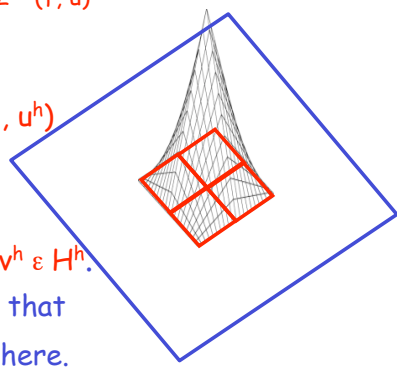
Long story…

dropping " " from $(L\cdot, \cdot)$ for simplicity

- 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.
- Suppose u minimizes F but [(Lu, v) - (f, v)] < 0 for some v. (Just flip the sign of v for the case [(Lu, v) - (f, v)] > 0.) Now replace v by εv:
  $F(u + \varepsilon v) = F(u) + \varepsilon\{[(Lu, v) - (f, v)] + \varepsilon(Lv, v)/2\}]$.
- Small enough ε>0 means [(Lu, v) - (f, v)] + ε(Lv, v)/2 < 0, which leads us to conclude that F(u + εv) < F(u), a contradiction.
- This contradiction shows that (Lu, v) - (f, v) must be 0 for all v. Since this argument can easily be reversed, we thus conclude that
  $F(u + v) \geq F(u) \quad \forall\, v \in H_0^1(\Omega) \quad \Leftrightarrow \quad (Lu, v) = (f, v) \quad \forall\, v \in H_0^1(\Omega)$.

Minimizing F via Rayleigh-Ritz

$F(u) = (Lu, u)/2 - (f, u)$



- Discretize by minimizing
  $F(u^h) = (Lu^h, u^h)/2 - (f, u^h)$
  over $u^h \in H^h$.
- Same as Galerkin that solves
  $(Lu^h, v^h) = (f, v^h) \quad \forall\, v^h \in H^h$.
- Basis: $\varepsilon^h_{(ij)}$ is the element of H^h that equals 1 @ node ij & 0 elsewhere.
- Expansion: $u^h(x,y) = \Sigma_{ij}\, u^h_{ij}\, \varepsilon^h_{(ij)}(x,y)$.
- Old problem: What is $Lu^h \equiv - u^h_{xx} - u^h_{yy}$???
  $(Lu^h, v^h) = (\nabla u^h, \nabla v^h)$

Abstract FE relaxation

- Relaxation involves "local" changes: $u^h \leftarrow u^h - s\, \varepsilon^h_{(ij)}$ for some scalar s & $\varepsilon^h = \varepsilon^h_{(ij)}$.

  But how do we pick s ???

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 H^{2h}. We go instead from abstract functions to nodal vectors…

Concrete FE coarsening: I^h_{2h}

now with focus on nodal vectors

- Adding nodal representations of v^h & v^{2h}:

$$v^{2h}(x,y) = \Sigma_{ij}\, v^{2h}{}_{ij}\, \varepsilon^{2h}{}_{(ij)}(x,y)$$

(sum over 2h indices ↔ even h indices)

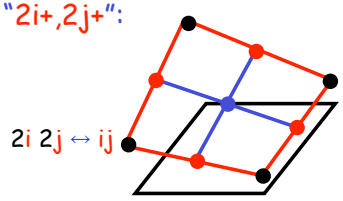
$v^{2h}{}_{ij}$: 2h nodal values

$$= \Sigma_{ij}\, v^h{}_{ij}\, \varepsilon^h{}_{(ij)}(x,y) \text{ ???}$$

(sum over h indices)

$v^h{}_{ij}$: h nodal values

- We should be able to do this because $v^{2h} \in H^{2h} \subset H^h$.
- Cell "2i+,2j+":



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

Bilinear interpolation!

Concrete FE coarsening (cont'd)

$$A^h u^h = f^h$$

$$A^h_{ij} = \frac{1}{3}\begin{pmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{pmatrix}$$

- Solving this matrix equation is equivalent to minimizing
  $F^h(v^h) \equiv (A^h v^h, v^h)/2 - (f^h, v^h)$ (parens here mean Euclidean norm)
  over $v^h \in H^h$. So how do we now correct v^h???
- We minimize $F^h(v^h + I^h_{2h} v^{2h})$ over $v^{2h} \in H^{2h}$:

$$F^h(v^h + I^h_{2h} v^{2h})$$

$$= (A^h (v^h + I^h_{2h} v^{2h}), v^h + I^h_{2h} v^{2h})/2 - (f^h, v^h + I^h_{2h} v^{2h})$$

$$= F^h(v^h) + \underbrace{(A^{2h} v^{2h}, v^{2h})/2 - (f^{2h}, v^{2h})}_{F^{2h}(v^{2h})}$$

$$\equiv F^h(v^h) + F^{2h}(v^{2h}).$$

variational conditions

$$f^{2h} = I^{h\,T}_{2h}(f^h - A^h v^h)$$

$$A^{2h} = I^{h\,T}_{2h} A^h I^h_{2h}$$

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!

Presentations 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, …
- It can be optimal, often O(# points).
- It can be robust in a practical sense.
- It is of great interest because it is one of the very few scalable algorithms, & it can be parallelized readily & efficiently!
- But multigrid can also be a real pain!!!