Jacobi and Gauss-Seidel Relaxation

- Key idea for relaxation techniques intuitive.
- Associate a single equation, corresponding single unknown, $u_{i,j}$, with each mesh point in Ω^h.
- Then repeatedly "sweep" through mesh, visiting each mesh point in some prescribed order.
- Each time point is visited, adjust value of unknown at grid point so corresponding equation is ("instantaneously") satisfied.
- Adopt a "residual based" approach to locally satisfying the discrete equations.
- Proceed in a manner that generalizes immediately to the solution of *non-linear* elliptic PDEs
- Ignore the treatment of boundary conditions (if conditions are *differential*, will also need to be relaxed)

Jacobi and Gauss-Seidel Relaxation

- Again, adopt "residual-based" approach to the problem of locally satisfying equations via relaxation

- Consider general form of discretized BVP

$$L^h u^h = f^h \tag{1}$$

and recast in canonical form

$$F^h \left[u^h \right] = 0 \,. \tag{2}$$

- Quantity u^h which appears above is the *exact* solution of the difference equations.

- Can generally only compute u^h in the limit of infinite iteration.

- Thus introduce $\tilde{u}^h$: "current" or "working" approximation to u^h, labelling the iteration number by n, and assuming iterative technique *does* converges, have

$$\lim_{n \to \infty} \tilde{u}^h = u^h \tag{3}$$

Jacobi and Gauss-Seidel Relaxation

- Associated with $\tilde{u}^h$ is residual, $\tilde{r}^h$

$$\tilde{r}^h \equiv L^h \tilde{u}^h - f^h \tag{4}$$

or in terms of canonical form (2),

$$\tilde{r}^h \equiv F^h \left[\tilde{u}^h\right] . \tag{5}$$

- For specific *component* (grid value) of residual, $\tilde{r}^h_{i,j}$, drop the h superscript

$$\tilde{r}_{i,j} = \left[L^h \tilde{u}^h - f^h\right]_{i,j} \equiv \left[F^h \left[u^h\right]\right]_{i,j} \tag{6}$$

- For model problem have

$$\tilde{r}_{i,j} = h^{-2} \left(\tilde{u}_{i+1,j} + \tilde{u}_{i-1,j} + \tilde{u}_{i,j+1} + \tilde{u}_{i,j-1} - 4\tilde{u}_{i,j}\right) - f_{i,j} \tag{7}$$

- Relaxation: adjust $\tilde{u}_{i,j}$ so corresponding residual is "instantaneously" zeroed

Jacobi and Gauss-Seidel Relaxation

- Useful to appeal to Newton's method for single non-linear equation in a single unknown.
- In current case, difference equation is *linear* in $\tilde{u}_{i,j}$: can *solve* equation with single Newton step.
- However, can also apply relaxation to *non-linear* difference equations, then can only zero residual in limit of infinite number of Newton steps.
- Thus write relaxation in terms of update, $\delta\tilde{u}_{i,j}^{(n)}$, of unknown

$$\tilde{u}_{i,j}^{(n)} \longrightarrow \tilde{u}_{i,j}^{(n+1)} = \tilde{u}_{i,j}^{(n)} + \delta\tilde{u}_{i,j}^{(n)} \tag{8}$$

where, again, (n) labels the iteration number.

Jacobi and Gauss-Seidel Relaxation

- Using Newton's method, have

$$\tilde{u}_{i,j}^{(n+1)} = \tilde{u}_{i,j}^{(n)} - \tilde{r}_{i,j} \left[\frac{\partial F_{i,j}^{h}}{\partial u_{i,j}} \Big|_{u_{i,j}=\tilde{u}_{i,j}^{(n)}} \right]^{-1} \tag{9}$$

$$= \tilde{u}_{i,j}^{(n)} - \frac{\tilde{r}_{i,j}}{-4h^{-2}} \tag{10}$$

$$= \tilde{u}_{i,j}^{(n)} + \frac{1}{4} h^2 \tilde{r}_{i,j} \tag{11}$$

- Precise computation of the residual needs clarification.

- Iterative method takes an entire vector of unknowns $\mathbf{u}^{(n)}$ to new estimate $\mathbf{u}^{(n+1)}$, but works on a component by component basis.

Jacobi and Gauss-Seidel Relaxation

- In computing individual residuals, could either choose only "old" values; i.e. values from iteration n, or, wherever available, could use "new" values from iteration $n+1$, with the rest from iteration n.

- First approach is known as *Jacobi relaxation*, residual computed as

$$\tilde{r}_{i,j} = h^{-2}\left(\tilde{u}^{(n)}_{i+1,j} + \tilde{u}^{(n)}_{i-1,j} + \tilde{u}^{(n)}_{i,j+1} + \tilde{u}^{(n)}_{i,j-1} - 4\tilde{u}^{(n)}_{i,j}\right) - f_{i,j} \qquad (12)$$

- Second approach is known as *Gauss-Seidel relaxation*: assuming lexicographic ordering of unknowns, $i = 1, 2, \cdots n$, $j = 1, 2, \cdots n$, i index varies most rapidly, residual is

$$\tilde{r}_{i,j} = h^{-2}\left(\tilde{u}^{(n)}_{i+1,j} + \tilde{u}^{(n+1)}_{i-1,j} + \tilde{u}^{(n)}_{i,j+1} + \tilde{u}^{(n+1)}_{i,j-1} - 4\tilde{u}^{(n)}_{i,j}\right) - f_{i,j} \qquad (13)$$

- Make few observations/comments about these two relaxation methods.

Jacobi and Gauss-Seidel Relaxation

- At each iteration "visit" each/every unknown *exactly once*, modifying its value so that local equation is instantaneously satisfied.
- Complete pass through the mesh of unknowns (i.e. a complete iteration) is known as a *relaxation sweep*.
- For Jacobi, visit order clearly irrelevant to what values are obtained at end of each iteration
- Fact is advantageous for parallelization but storage is required for *both* the new and old values of $\tilde{u}_{i,j}$.
- For Gauss-Seidel (GS), need storage for current estimate of $\tilde{u}_{i,j}$: sweeping order *does* impact the details of the solution process.
- Thus, lexicographic ordering does not parallelize, but for difference equations such as those for model problem, with nearest-neighbour couplings, so called *red-black* ordering *can* be parallelized, has other advantages.
- Red-black ordering: appeal to the red/black squares on a chess board—two subiterations

  1. Visit and modify all "red" points, i.e. all (i, j) such that $\mathrm{mod}(i + 1, 2) = 0$
  2. Visit and modify all "black" points, i.e. all (i, j) such that $\mathrm{mod}(i + 1, 2) = 1$

Convergence of Relaxation Methods

- Key issue with any iterative technique is whether or not the iteration will converge
- For Gauss-Seidel and related, was examined comprehensively in the 1960's and 1970's: won't go into much detail here
- Very rough rule-of-thumb: GS will converge if linear system *diagonally dominant*
- Write (linear) difference equations

$$L^h u^h = f^h \tag{14}$$

in matrix form

$$\mathbf{A}\mathbf{u} = \mathbf{b} \tag{15}$$

where $\mathbf{A}$ is an $N \times N$ matrix ($N \equiv$ total number of unknowns); $\mathbf{u}$ and $\mathbf{b}$ are N-component vectors

Convergence of Relaxation Methods

- $\mathbf{A}$ diagonally dominant if

$$|a_{ij}| \leq \sum_{j=1, j\neq i}^{N} |a_{ij}|\,, i = 1, 2, \cdots N \tag{16}$$

  with strict inequality holding for at least one value of i.

- Another practical issue: *how* to monitor convergence?

- For relaxation methods, two natural quantities to look at: the *residual norm* $\|\tilde{r}^h\|$ and the *solution update norm* $\|\tilde{u}^{(n+1)} - \tilde{u}^{(n)}\|$

- Both should tend to 0 in limit of infinite iteration, for convergent method be convergent.

- In practice, monitoring residual norm straightforward and often sufficient.

Convergence of Relaxation Methods

- Consider the issue of convergence in more detail.
- For a general iterative process leading to solution vector $\mathbf{u}$, and starting from some initial estimate $\mathbf{u}^{(0)}$, have

$$\mathbf{u}^{(0)} \rightarrow \mathbf{u}^{(1)} \rightarrow \cdots \rightarrow \mathbf{u}^{(n)} \rightarrow \mathbf{u}^{(n+1)} \rightarrow \cdots \rightarrow \mathbf{u} \tag{17}$$

- For the residual vector, have

$$\mathbf{r}^{(0)} \rightarrow \mathbf{r}^{(1)} \rightarrow \cdots \rightarrow \mathbf{r}^{(n)} \rightarrow \mathbf{r}^{(n+1)} \rightarrow \cdots \rightarrow \mathbf{0} \tag{18}$$

- For the solution error, $\mathbf{e}^{(n)} \equiv \mathbf{u} - \mathbf{u}^{(n)}$, have

$$\mathbf{e}^{(0)} \rightarrow \mathbf{e}^{(1)} \rightarrow \cdots \rightarrow \mathbf{e}^{(n)} \rightarrow \mathbf{e}^{(n+1)} \rightarrow \cdots \rightarrow \mathbf{0} \tag{19}$$

Convergence of Relaxation Methods

- For linear relaxation (and basic idea generalizes to non-linear case), can view transformation of error, $\mathbf{e}^{(n)}$, at each iteration in terms of linear matrix (operator), $\mathbf{G}$, known as the *error amplification matrix*.

- Then have

$$\mathbf{e}^{(n+1)} = \mathbf{G}\mathbf{e}^{(n)} = \mathbf{G}^2\mathbf{e}^{(n-1)} = \cdots = \mathbf{G}^n\mathbf{e}^{(0)} \tag{20}$$

- Asymptotically convergence determined by the *spectral radius*, $\rho(\mathbf{G})$:

$$\rho(\mathbf{G}) \equiv \max_i |\lambda_i(\mathbf{G})| \tag{21}$$

  and the λ_i are the eigenvalues of $\mathbf{G}$.

- That is, in general have

$$\lim_{n\to\infty} \frac{\|\mathbf{e}^{(n+1)}\|}{\|\mathbf{e}^n\|} = \rho(\mathbf{G}) \tag{22}$$

- Can then define *asymptotic convergence rate*, R

$$R \equiv \log_{10}\left(\rho^{-1}\right) \tag{23}$$

Convergence of Relaxation Methods

- Useful interpretation of R: R^{-1} is number of iterations needed asympotically decrease $\|\mathbf{e}^{(n+1)}\|$ by an order of magnitude.

- Now consider the computational cost of solving discrete BVPs using relaxation.

- "Cost" is to be roughly identified with CPU time, and want to know how cost scales with key numerical parameters

- Here, assume that there is *one* key parameter: total number of grid points, N

- For concreteness/simplicity assume following

  1. The domain Ω is d-dimensional ($d = 1, 2, 3$ being most common)
  2. There are n grid points per edge of the (uniform) discrete domain, Ω^h

- Then total number of grid points, N, is

$$N = n^d \tag{24}$$

- Further define the computational cost/work, $W \equiv W(N)$ to be work required to reduce the error, $\|\mathbf{e}\|$ by an order of magnitude; definition suffices to compare methods

Convergence of Relaxation Methods

- Clearly, best case situation is $W(N) = O(N)$.
- For Gauss-Seidel relaxation, state without proof that for model problem (and for many other elliptic systems in $d = 1, 2, 3$), have

$$\rho\left(\mathbf{G}_{\mathrm{GS}}\right) = 1 - O(h^2) \tag{25}$$

- Implies that relaxation work, W_{GS}, required to reduce the solution error by order of magnitude is

$$W_{\mathrm{GS}} = O\left(h^{-2}\,\mathrm{sweeps}\right) = O\left(n^2\,\mathrm{sweeps}\right) \tag{26}$$

- Each relaxation sweep costs $O(n^d) = O(N)$, so

$$W_{\mathrm{GS}} = O\left(n^2 N\right) = O\left(N^{2/d} N\right) = O\left(N^{(d+2)/d}\right) \tag{27}$$

Convergence of Relaxation Methods

- Tabulating the values for $d = 1, 2$ and 3

d	W_{GS}
1	$O(N^3)$
2	$O(N^2)$
3	$O(N^{5/3})$

- Scaling improves as d, increases, but $O(N^2)$ and $O(N^{5/3})$ for the cases $d = 2$ and $d = 3$ are pretty bad

- Reason that Gauss-Seidel is rarely used in practice, particularly for large problems (large values of n, small values of h).

Successive Over Relaxation (SOR)

- Historically, researchers studying Gauss-Seidel found that convergence of the method could often be substantially improved by systematically "over correcting" the solution, relative to what the usual GS computation would give
- Algorithmically, change from GS to SOR is very simple, and is given by

$$\tilde{u}_{i,j}^{(n+1)} = \omega\, \hat{u}_{i,j}^{(n+1)} + (1-\omega)\, \tilde{u}_{i,j}^{(n)} \tag{28}$$

- Here, ω is the *overrelaxation parameter*, typically chosen in the range $1 \leq \omega < 2$, and $\hat{u}_{i,j}^{(n+1)}$ is the value computed from the normal Gauss-Seidel iteration.
- When $\omega = 1$, recover GS iteration itself: for large enough ω, e.g. $\omega \geq 2$, iteration will become unstable.
- *At best*, the use of SOR reduces number of relaxation sweeps required to drop the error norm by an order of magnitude to $O(n)$:

$$W_{\mathrm{SOR}} = O\left(nN\right) = O\left(N^{1/d}N\right) = O\left(N^{(d+1)/d}\right) \tag{29}$$

Successive Over Relaxation (SOR)

- Again tabulating the values for $d = 1, 2$ and 3 find

d	W_{SOR}
1	$O(N^2)$
2	$O(N^{3/2})$
3	$O(N^{4/3})$

- Thus note that *optimal* SOR is not unreasonable for "moderately-sized" $d = 3$ problems.

- Key issue: How to choose $\omega = \omega_{\mathrm{OPT}}$ in order to get optimal performance

- Except for cases where $\rho_{\mathrm{GS}} \equiv \rho(G_{\mathrm{GS}})$ is known explicitly, ω_{OPT} needs to be determined *empirically* on a case-by-case *and* resolution-by-resolution basis.

- If ρ_{GS} *is* known, then one has

$$\omega_{\mathrm{OPT}} = \frac{2}{1 + \sqrt{1 - \rho_{\mathrm{GS}}}} \tag{30}$$

Relaxation as Smoother (Prelude to Multi-Grid)

- Slow convergence rate of relaxation methods such as Gauss-Seidel $\longrightarrow$ not much use for solving discretized elliptic problems
- Now consider an even simper model problem to show what relaxation *does* tend to do well
- Will elucidate why relaxation is so essential to the multi-grid method.
- Consider a one dimensional ($d = 1$) "ellliptic" model problem, i.e. an *ordinary* differential equation to be solved as a two-point boundary value problem.
- Equation to be solved is

$$Lu(x) \equiv \frac{d^2u}{d^2x} = f(x) \tag{31}$$

where $f(x)$ is specified source function.

Relaxation as Smoother (Prelude to Multi-Grid)

- Solve (31) on a domain, Ω, given by

$$\Omega \; : \; 0 \leq x \leq 1 \tag{32}$$

subject to the (homogeneous) boundary conditions

$$u(0) = u(1) = 0 \tag{33}$$

- To discretize this problem, introduce a uniform mesh, Ω^h

$$\Omega^h = \{(i-1)h, \; i = 1, 2, \cdots n\} \tag{34}$$

where $n = h^{-1} + 1$ and h, as usual, is the mesh spacing.

- Now discretize (31) to $O(h^2)$ using usual FDA for second derivative, getting

$$\frac{u_{i+1} - 2u_i + u_{i-1}}{h^2} = f_i \, , \; i = 2, 3, \cdots n-1 \tag{35}$$

Relaxation as Smoother (Prelude to Multi-Grid)

- Last equations, combined with boundary conditions

$$u_1 = u_n = 0 \tag{36}$$

yield set of n linear equations in n unknowns, $u_i, i = 1, 2, \cdots n$.

- For analysis, useful to eliminate u_1 and u_n from the discretization , so that in terms of generic form

$$L^h u^h = f^h \tag{37}$$

u^h and f^h are $n-2$-component vectors; L^h is an $(n-2) \times (n-2)$ matrix.

- Consider a specific relaxation method, known as *damped Jacobi*, chosen for ease of analysis, but representative of other relaxation techniques, including Gauss-Seidel.

- Using vector notation, damped Jacobi iteration is given by

$$\tilde{\mathbf{u}}^{(n+1)} = \tilde{\mathbf{u}}^{(n)} - \omega \mathbf{D}^{-1} \tilde{\mathbf{r}}^{(n)} \tag{38}$$

where $\tilde{\mathbf{u}}$ is the approximate solution of the difference equations, $\tilde{\mathbf{r}}$ is the residual vector, ω is an (underrelaxation) parameter, and $\mathbf{D}$ is the main diagonal of L^h.

Relaxation as Smoother (Prelude to Multi-Grid)

- For model problem, have

$$\mathrm{D} = -2h^{-2}\mathbf{1} \tag{39}$$

- Note that when $\omega = 1$, this is just the usual Jacobi relaxation method discussed previously.

- Examine the effect of (38) on residual vector, $\tilde{\mathbf{r}}$.

- Have

$$\begin{aligned}
\tilde{\mathbf{r}}^{(n+1)} &\equiv L^h\tilde{\mathbf{u}}^{(n+1)} - f^h && (40)\\
&= L^h\left(\tilde{\mathbf{u}}^{(n)} - \omega\mathbf{D}^{-1}\tilde{\mathbf{r}}^{(n)}\right) - f^h && (41)\\
&= \left(\mathbf{1} - \omega L^h\mathbf{D}^{-1}\right)\tilde{\mathbf{r}}^{(n)} && (42)\\
&\equiv \mathbf{G}_{\mathrm{R}}\tilde{\mathbf{r}}^{(n)} && (43)
\end{aligned}$$

where the *residual amplification matrix*, $\mathbf{G}_{\mathrm{R}}$, is

$$\mathbf{G}_{\mathrm{R}} \equiv \mathbf{1} - \omega L^h\mathbf{D}^{-1} \tag{44}$$

Relaxation as Smoother (Prelude to Multi-Grid)

- Have following relation for residual at the n-th iteration, $\tilde{\mathbf{r}}^{(n)}$, in terms of initial residual, $\tilde{\mathbf{r}}^{(0)}$:

$$\tilde{\mathbf{r}}^{(n)} = \left(\mathbf{G}_{\mathrm{R}}\right)^{n} \, \tilde{\mathbf{r}}^{(0)} \tag{45}$$

- Now, $\mathbf{G}_{\mathrm{R}}$, has complete set of orthogonal eigenvectors, ϕ_m , $m = 1, 2, \cdots n-2$ with corresponding eigenvalues μ_m.

- Thus, can expand initial residual, $\tilde{\mathbf{r}}^{(0)}$, in terms of the ϕ_m:

$$\tilde{\mathbf{r}}^{(0)} = \sum_{m=1}^{n-2} c_m \phi_m \tag{46}$$

  where the c_m are coefficients.

- Immediately yields following expansion for the residual at the n-th iteration:

$$\tilde{\mathbf{r}}^{(n)} = \sum_{m=1}^{n-2} c_m \left(\mu_m\right)^n \phi_m \, . \tag{47}$$

Relaxation as Smoother (Prelude to Multi-Grid)

- Thus, after n sweeps, the m-th "Fourier" component of the initial residual vector, $\tilde{r}^{(0)}$ is reduced by a factor of $(\mu_m)^n$.
- Now consider the specific value of the underrelaxation parameter, $\omega = 1/2$.
- Left as exercise to verify that eigenvectors, ϕ_m are

$$\phi_m = \left[\sin(\pi m h), \sin(2\pi m h), \cdots, \sin((n-2)\pi m h)\right]^T, \ m = 1, 2, \cdots, n-2 \tag{48}$$

while eigenvalues, μ_m, are

$$\mu_m = \cos^2\left(\frac{1}{2}\pi m h\right), \ m = 1, 2, \cdots, n-2 \tag{49}$$

- Note that each eigenvector, ϕ_m, has associated "wavelength", λ_m

$$\sin(\pi m x) = \sin(2\pi x/\lambda_m). \tag{50}$$

- Thus

$$\lambda_m = \frac{2}{m} \tag{51}$$

Relaxation as Smoother (Prelude to Multi-Grid)

- As m increases (and λ_m decreases), "frequency" of ϕ_m *increases*, and, from (49), the eigenvalues, μ_m, *decrease* in magnitude.

- Asymptotic convergence rate of the relaxation scheme is determined by *largest* of the μ_m, μ_1

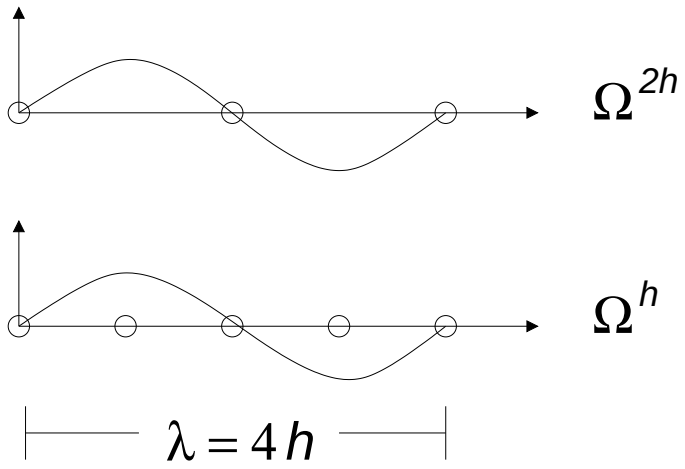
$$\mu_1 = \cos^2\left(\frac{1}{2}\pi h\right) = 1 - \frac{1}{4}\pi^2 h^2 + \cdots = 1 - O(h^2)\,. \tag{52}$$

- Implies that $O(n^2)$ sweeps are needed to reduce norm of residual by order of magnitude.

- Thus see that asymptotic convergence rate of relaxation scheme is dominated by the (slow) convergence of *smooth* (low frequency, long wavelength) components of the residual, $\tilde{\mathbf{r}}^{(n)}$, or, equivalently, the solution error, $\tilde{\mathbf{e}}^{(n)}$.

- Comment applies to many other relaxation schemes applied to typical elliptic problems, including Gauss-Seidel.

Definition of "High Frequency"



- Illustration of the definition of "high frequency" components of the residual or solution error, on a fine grid, Ω^h, relative to a coarse grid, Ω^{2h}.

- As illustrated in the figure, any wave with $\lambda < 4h$ *cannot* be represented on the coarse grid (i.e. the Nyquist limit of the coarse grid is $4h$.)

Relaxation as Smoother (Prelude to Multi-Grid)

- Highly instructive to consider what happens to *high* frequency components of the residual (or solution error) as the damped Jacobi iteration is applied.
- Per previous figure, are concerned with eigenvectors, ϕ_m, such that

$$\lambda_m \leq 4h \longrightarrow mh \geq \frac{1}{2} \tag{53}$$

- In this case, have

$$\mu_m = \cos^2\left(\frac{1}{2}\pi mh\right) \leq \cos^2\left(\frac{\pi}{4}\right) \leq \frac{1}{2} \tag{54}$$

- Thus, *all* components of residual (or solution error) that cannot be represented on Ω^{2h} get supressed by a factor of at least $1/2$ *per sweep*.
- Furthermore, rate at which high-frequency components are liquidated is *independent* of the mesh spacing, h.

Relaxation as Smoother (Prelude to Multi-Grid)

- SUMMARY:
  - Relaxation tends to be a *dismal solver* of systems $L^h u^h = f^h$, arising from FDAs of elliptic PDEs.
  - But, tends to be a *very good smoother* of such systems—crucial for the success of multi-grid method.

Multi-grid: Motivation and Introduction

- Again consider model problem

$$u(x,y)_{xx} + u_{yy} = f(x,y) \tag{55}$$

  solved on the unit square subject to homogeneous Dirichlet BCs

- Again discretize system using standard 5-point $O(h^2)$ FDA on a uniform $n \times n$ grid with mesh spacing h

- Yields algebraic system $L^h u^h = f^h$.

- Assume system solved with iterative solution technique: start with some initial estimate $\tilde{\mathbf{u}}^{(0)}$, iterate until $\|\tilde{\mathbf{r}}^{(n)}\| \leq \epsilon$.

- Several questions concerning solution of the discrete problem then arise:

  1. How do we choose n (equivalently, h)?
  2. How do we choose ϵ?
  3. How do we choose $\tilde{\mathbf{u}}^{(0)}$?
  4. How fast can we "solve" $L^h u^h = f^h$?

- Now provide at least partial answers to all of these questions.

Multi-grid: Motivation and Introduction

1) Choosing n (equivalently h)

- Ideally, would choose h so that error $\|\tilde{\mathbf{u}} - \mathbf{u}\|$, satisfies $\|\tilde{\mathbf{u}} - \mathbf{u}\| < \epsilon_u$, where ϵ_u is a user-prescribed error tolerance.

- From discussions of FDAs for time-dependent problems, already know how to estimate solution error for essentially *any* finite difference solution.

- Assume that FDA is centred and $O(h^2)$

- Richardson tells us that for solutions computed on grids with mesh spacings h and $2h$, expect

$$u^h \sim u + h^2 e_2 + h^4 e_4 + \cdots \tag{56}$$

$$u^{2h} \sim u + 4h^2 e_2 + 16h^4 e_4 + \cdots \tag{57}$$

Multi-grid: Motivation and Introduction

- So to leading order in h, error, $u^h - u$, is

$$e \sim \frac{u^{2h} - u^h}{3h^2} \tag{58}$$

- Thus, basic strategy is to perform *convergence tests*—comparing finite difference solutions at different resolutions, increase (or decrease) h until the level of error is satisfactory.

Multi-grid: Motivation and Introduction

2) Choosing ϵ (ϵ_r)

- Consider following 3 expressions:

$$L^h u^h - f^h = 0 \quad (59)$$

$$L^h \tilde{u}^h - f^h = \tilde{r}^h \quad (60)$$

$$L^h u - f^h = \tau^h \quad (61)$$

where u^h is exact solution of finite difference equations, $\tilde{u}^h$ is approximate solution of the FDA, u is the (exact) solution of the continuum problem, $Lu - f = 0$.

- (59) is simply our FDA written in a canonical form, (60) defines the residual, while (61) defines the truncation error.

Multi-grid: Motivation and Introduction

- Comparing (60) and (61), see that it is natural to stop iterative solution process when

$$\|\tilde{r}^h\| \sim \|\tau^h\| \tag{62}$$

- Leaves problem of estimating size of truncation error
- Will see later how estimates of τ^h arise naturally in multi-grid algorithms.

Multi-grid: Motivation and Introduction

3) Choosing $\tilde{\mathbf{u}}^{(0)}$

- Key idea is to use solution of coarse-grid problem as initial estimate for fine-grid problem.

- Assume that have determined satisfactory resolution h; i.e. wish to solve

$$L^h u^h = f^h \tag{63}$$

- Then first pose and solve (perhaps approximately) corresponding problem (i.e. same domain, boundary conditions and source function, f) on mesh with spacing $2h$.

- That is, solve

$$L^{2h} u^{2h} = f^{2h} \tag{64}$$

then set initial estimate, $(u^h)^{(0)}$ via

$$\left(u^h\right)^{(0)} = \bar{I}^h_{2h}\, u^{2h} \tag{65}$$

Multi-grid: Motivation and Introduction

- Here $\bar{I}_{2h}^{h}$ is known as a *prolongation operator* and transfers a coarse-grid function to a fine-grid.
- Typically, $\bar{I}_{2h}^{h}$ will perform d-dimensional polynomial interpolation of the coarse-grid unknown, u^{2h}, to a suitable order in the mesh spacing, h.
- Chief advantage of approach is that solution of coarse-grid problem should be inexpensive to solve relative to fine-grid problem
- Specifically, cost of solving on the coarse-grid should be no more than 2^{-d} of cost of soving the fine-grid equations.
- Furthermore, can apply basic approach *recursively*: initialize u^{2h} with u^{4h} result, initialize u^{4h} with u^{8h} etc
- Thus lead to consider a general *multi-level* technique for treatment of discretized elliptic PDEs
- Solution of a fine-grid problem is preceded by solution of series of coarse-grid problems.

Multi-grid: Motivation and Introduction

- Label each distinct mesh spacing with an integer ℓ

$$\ell = 1, 2, \cdots \ell_{\max} \tag{66}$$

  where $\ell = 1$ labels coarsest spacing h_1, $\ell = \ell_{\max}$ labels finest spacing $h_{\ell_{\max}}$.

- Almost always most convenient (and usually most efficient) to use mesh spacings, h_ℓ, satisfying

$$h_{\ell+1} = \frac{1}{2} h_\ell \tag{67}$$

- This implies

$$n_{\ell+1} \sim 2^d n_\ell \,. \tag{68}$$

- Will assume in following that multi-level algorithms use 2:1 mesh hierarchies

- Use ℓ itself to denote resolution associated with a particular finite difference approximation, or transfer operator.

- That is, define u^ℓ via

$$u^\ell = u^{h_\ell} \tag{69}$$

Multi-grid: Motivation and Introduction

- Could then rewrite (65) as

$$u^{\ell+1} = \bar{I}_{\ell}^{\ell+1} u^{\ell} \tag{70}$$

- Pseudo-code for general multi-level technique:

for $\ell = 1\,,\ \ell_{\max}$
  if $\ell = 1$ **then**
    $u^{\ell} := 0$
  else
    $u^{\ell} := \bar{I}_{\ell-1}^{\ell} u^{\ell-1}$
  end if
  `solve_iteratively` (u^{ℓ})
end for

- Very simple and intuitive algorithm: when novices hear term "multi-grid", sometimes think that this sort of approach is all that is involved.

- However, multi-grid algorithm uses grid hierarchy for entirely different, and more essential purpose!

Multi-grid: Motivation and Introduction

4) How fast can we solve $L^h u^h = f^h$?

- Answer to this question is short and sweet.
- Properly constructed multi-grid algorithm can solve discretized elliptic equation in $O(N)$ operations!
- This is, of course, computationally optimal: main reason that multi-grid methods have become such an important tool for numerical analysts—numerical relativists included.