Numerical Methods for Elliptic Equations

Instructor: Hong G. Im
University of Michigan
Fall 2001

Outline

Solution Methods for Elliptic Equations

Direct Methods

- Cramer’s rule
- Gaussian elimination
- Tridiagonal matrix algorithm (Thomas algorithm)
- Block tridiagonal matrix algorithm

Iterative Methods

- Jacobi, Gauss-Seidel iteration
- Successive overrelaxation (SOR)
- Multigrid method

A model equation for elliptic problems

$$\frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} = S; \quad \nabla^2 f = S$$

Poisson equation (Laplace equation if $S = 0$).

On the boundaries (BC)

$$f = f_0(x, y) \qquad \text{Dirichlet}$$

$$\frac{\partial f}{\partial n} = g_0(x, y) \qquad \text{Neumann}$$

Examples of Elliptic Equations

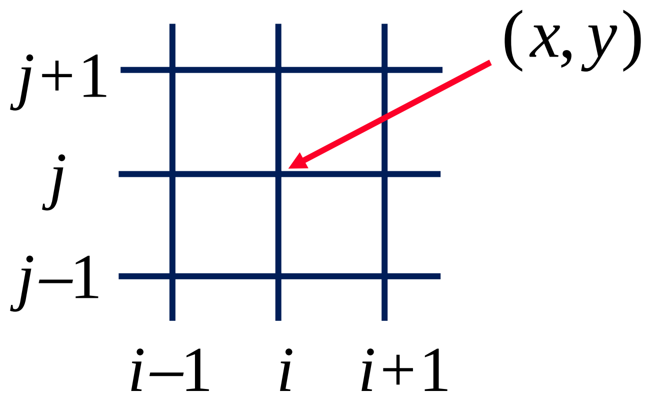
$$\nabla^2 T = -\frac{\dot{q}}{k}$$ Steady conduction equation

$$\nabla^2 \psi = -\omega$$ 2-D stream function equation

$$\nabla_h^2 P_{i,j} = \frac{1}{\Delta t} \nabla_h \cdot \mathbf{u}_{i,j}^t$$ Projection method (Step 2)

Solving the Poisson equation

$$\frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} = S$$



Applying central differencing

$$\frac{f_{i+1,j} - 2f_{i,j} + f_{i-1,j}}{\Delta x^2} + \frac{f_{i,j+1} - 2f_{i,j} + f_{i,j-1}}{\Delta y^2} = S_{i,j}$$

for which the modified equation becomes

$$f_{xx} + f_{yy} = S - \frac{1}{12}\left[f_{xxxx}\Delta x^2 + f_{yyyy}\Delta y^2\right] + \cdots$$

If $\Delta x = \Delta y = h \Rightarrow \; f_{i+1,j} + f_{i-1,j} + f_{i,j+1} + f_{i,j-1} - 4 f_{i,j} = h^2 S_{i,j}$

$$
\begin{bmatrix}
-4 & 1 & 0 & 0 & \cdots & 1 & 0 & \cdots & \cdots & \cdots \\
1 & -4 & 1 & 0 & \cdots & 0 & 1 & \cdots & \cdots & \cdots \\
0 & 1 & -4 & 1 & \cdots & \cdots & 0 & 1 & \cdots & \cdots \\
\vdots & & & & & & & & & \\
\vdots & & & & & & & & & \\
1 & 0 & & & & & & & & \\
0 & 1 & & & & & & & & \\
0 & 0 & 1 & & & & & & & \\
 & & & & & & & & & \\
 & & & & & & & 0 & 1 & -4
\end{bmatrix}
\begin{bmatrix}
f_{1,1} \\ f_{1,2} \\ \vdots \\ f_{1,J} \\ f_{2,1} \\ f_{2,2} \\ \vdots \\ \vdots \\ f_{I,J-1} \\ f_{I,J}
\end{bmatrix}
= h^2
\begin{bmatrix}
S_{1,1} \\ S_{1,2} \\ \vdots \\ S_{1,J} \\ S_{2,1} \\ S_{2,2} \\ \vdots \\ \vdots \\ S_{I,J-1} \\ S_{I,J}
\end{bmatrix}
$$

Sparse matrix: only 5 non-zero entries in each row

Ultimately, the difference form of the Poisson equation boils down to solving for

$$[A]\mathbf{f} = \mathbf{b}$$

Hence,

$$\mathbf{f} = [A]^{-1}\mathbf{b}$$

Direct method:

- Solving inverse matrix directly (Cramer's rule)
- Inverting matrix more cleverly (Gaussian elimination)
- Other (L-U decomposition, Thomas algorithm)

Direct Method - 1

Cramer's Rule

- Check elementary linear algebra book
- Extremely time consuming for large matrices

Operation count: $(n+1)!$

For example, 11×11 matrix:

$$(9\times 9+1)! = 4.75\times 10^{22}$$

Using 1 Gigaflops (10^9 flops) computer

$$3.20\times 10^{100}\ \text{years}!!!!!!$$

WPI

Computational Fluid Dynamics I
Direct Method - 2

Gaussian Elimination

- Pivoting: rearranging equations to put the largest coefficient on the main diagonal.
- Eliminate the column below main diagonal.
- Repeat until the last equation is reached.
- Back-substitution

$$\begin{array}{l} a_{11} f_1 + a_{12} f_2 + \cdots\cdots = c_1 \\ a_{21} f_1 + a_{22} f_2 + \cdots\cdots = c_2 \\ \vdots \qquad\qquad\qquad\qquad \vdots \\ a_{n1} f_1 + a_{n2} f_2 + \cdots\cdots = c_n \end{array} \quad \Rightarrow \quad \begin{array}{r} a_{11} f_1 + a_{12} f_2 + \cdots\cdots = c_1 \\ a'_{22} f_2 + \cdots\cdots = c_2 \\ \vdots \\ a'_{nn} f_n = c_n \end{array}$$

- Special case: tri-diagonal matrix - Thomas algorithm

Computational Fluid Dynamics I

Iterative Method

Discretized Poisson Equation

$$f_{i+1,j} + f_{i-1,j} + f_{i,j+1} + f_{i,j-1} - 4f_{i,j} = h^2 S_{i,j}$$

Rearranging for $f_{i,j}$

$$f_{i,j} = \frac{1}{4}\left[f_{i+1,j} + f_{i-1,j} + f_{i,j+1} + f_{i,j-1} - h^2 S_{i,j}\right]$$

$$f_{i,j}^{n+1} = \frac{1}{4}\left[f_{i+1,j}^{n} + f_{i-1,j}^{n} + f_{i,j+1}^{n} + f_{i,j-1}^{n} - h^2 S_{i,j}\right]$$ Jacobi

$$f_{i,j}^{n+1} = \frac{1}{4}\left[f_{i+1,j}^{n} + f_{i-1,j}^{n+1} + f_{i,j+1}^{n} + f_{i,j-1}^{n+1} - h^2 S_{i,j}\right]$$ Gauss-Seidel

$$f_{i,j}^{n+1} = \frac{\beta}{4}\left[f_{i+1,j}^{n} + f_{i-1,j}^{n+1} + f_{i,j+1}^{n} + f_{i,j-1}^{n+1} - h^2 S_{i,j}\right] + (1-\beta) f_{i,j}^{n}$$ SOR

Convergence – 1-D Problem (1)

A One Dimensional Example

An equation in the form

$$x = F(x)$$

can be solved by iterative procedure:

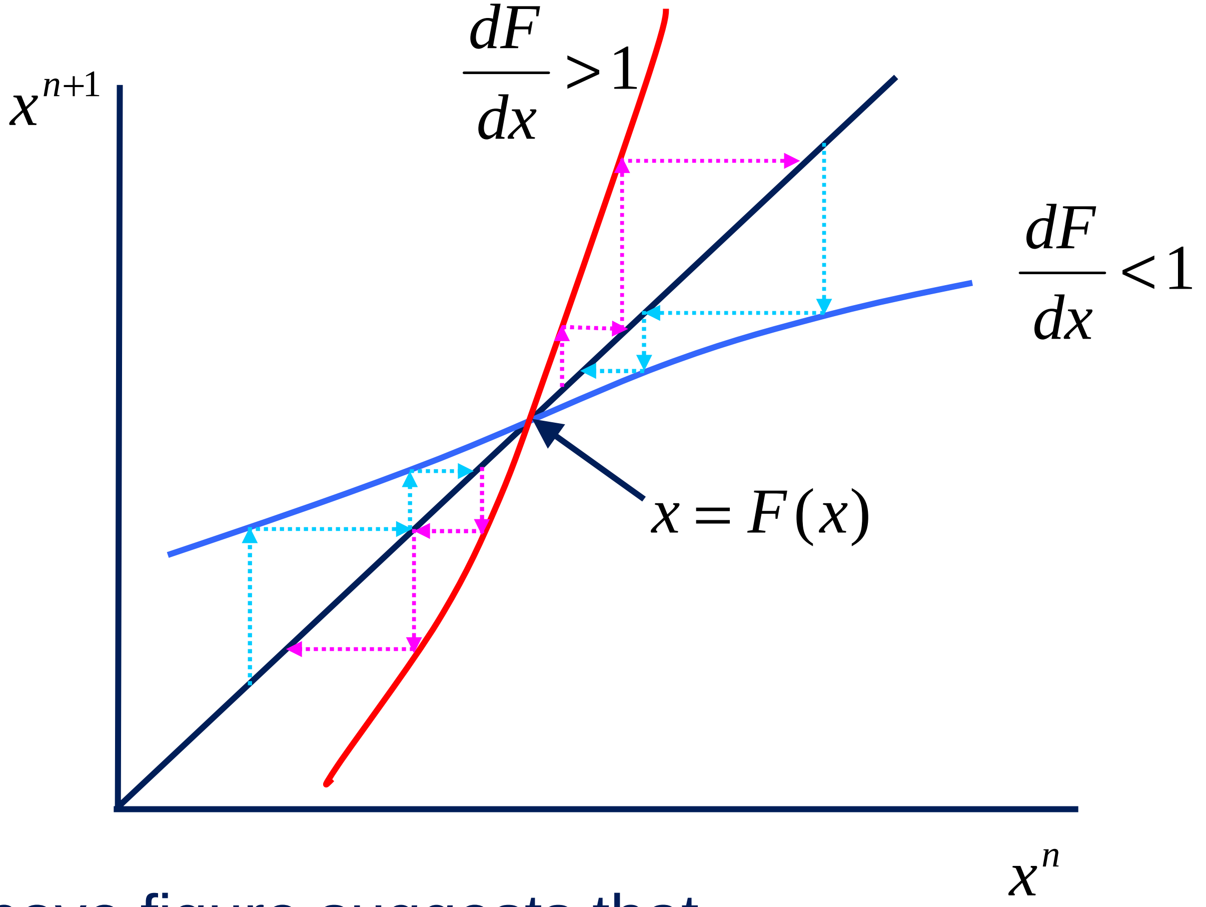
$$x^{n+1} = F(x^n)$$

for which convergence is achieved if

$$x^{n+1} \approx x^n \qquad \text{or} \qquad \left| \frac{x^{n+1}}{x^n} - 1 \right| < \varepsilon$$

Computational Fluid Dynamics I

Convergence – 1-D Problem (2)



The above figure suggests that

$$\left|\frac{dF}{dx}\right| \leq 1 \quad \text{for convergence}$$

Convergence – Multi Dimensional (1)

For multi-dimensional problems

$$\mathbf{f}^{n+1} = [M]\mathbf{f}^{n}$$

It can be shown that the space of $\mathbf{f}$ can be “spanned” by eigenvectors

$$[M]\mathbf{q}_i = \lambda_i \mathbf{q}_i, \quad i = 1, \cdots M \times N$$

such that

$$\mathbf{f} = y_1\mathbf{q}_1 + y_2\mathbf{q}_2 + \cdots + y_{M\times N}\mathbf{q}_{M\times N}$$

Hence,

$$[M]\mathbf{f} = [M]\left(y_1\mathbf{q}_1 + y_2\mathbf{q}_2 + \cdots + y_{M\times N}\mathbf{q}_{M\times N}\right)$$

$$= y_1[M]\mathbf{q}_1 + y_2[M]\mathbf{q}_2 + \cdots$$

$$= y_1\lambda_1\mathbf{q}_1 + y_2\lambda_2\mathbf{q}_2 + \cdots$$

Convergence – Multi Dimensional (2)

Therefore, the problem

$$\mathbf{f}^{n+1} = [M]\mathbf{f}^{n}$$

is equivalent to

$$y_1^{n+1}\mathbf{q}_1 + y_2^{n+1}\mathbf{q}_2 + \cdots = y_1^{n}\lambda_1\mathbf{q}_1 + y_2^{n}\lambda_2\mathbf{q}_2 + \cdots$$

or

$$\begin{aligned} y_1^{n+1} &= \lambda_1 y_1^{n} \\ y_2^{n+1} &= \lambda_2 y_2^{n} \\ &\vdots \end{aligned}$$

yielding a convergence condition

$$\lambda_{\max} \leq 1$$

WPI

Computational Fluid Dynamics I

Convergence – Multi Dimensional (3)

General iterative procedure

$$[A]\mathbf{f} = \mathbf{b}$$

Let $[A] = [A_1] - [A_2]$

$$[A_1]\mathbf{f} = [A_2]\mathbf{f} + \mathbf{b}$$

An iterative scheme is constructed as

$$[A_1]\mathbf{f}^{n+1} = [A_2]\mathbf{f}^{n} + \mathbf{b}$$

For example,

$[A_1] = [D] = -\frac{1}{4}[I], \quad [A_2] = [B] = [A_1] - [D]$ Jacobi

$[A_1] = [D] - [L], \quad [A_2] = [U]$ Gauss-Seidel

WPI

Computational Fluid Dynamics I

Convergence – Multi Dimensional (4)

$$[A_1]\mathbf{f}^{n+1} = [A_2]\mathbf{f}^n + \mathbf{b}$$

Requirements:

1. $[A_1]$ should be invertible.
2. Iteration should converge, i.e.

$$\lim_{n\to\infty} \mathbf{f}^n = \mathbf{f}$$

Define error at n-th iteration

$$\boldsymbol{\varepsilon}^n = \mathbf{f} - \mathbf{f}^n$$

$$[A_1]\boldsymbol{\varepsilon}^{n+1} = [A_2]\boldsymbol{\varepsilon}^n \qquad \boldsymbol{\varepsilon}^{n+1} = [A_1]^{-1}[A_2]\boldsymbol{\varepsilon}^n$$

$$\boldsymbol{\varepsilon}^n = \left([A_1]^{-1}[A_2]\right)^n \boldsymbol{\varepsilon}^0$$

WPI

Computational Fluid Dynamics I

Convergence – Multi Dimensional (5)

Condition for convergence

$$\lim_{n\to\infty} \boldsymbol{\varepsilon}^n = 0$$

which requires $\lim_{n\to\infty}\left([A_1]^{-1}[A_2]\right)^n = 0$

This is achieved if the modulus of the eigenvalues are less than unity.

Therefore, the convergence condition becomes:

$$\boxed{\rho = |\lambda_i|_{\max} \le 1}$$

ρ Spectral radius of convergence

λ_i Eigenvalues of matrix $[A_1]^{-1}[A_2]$

WPI

Computational Fluid Dynamics I

Convergence – Jacobi Method (1)

Jacobi Iteration Method for Poisson Equation
(2nd order central difference with uniform mesh)

$$f_{i,j}^{n+1} = \frac{1}{4}\left[f_{i+1,j}^{n} + f_{i-1,j}^{n} + f_{i,j+1}^{n} + f_{i,j-1}^{n} - h^2 S_{i,j} \right]$$

$$[A_1]^{-1} = [I]$$

and using a discrete analog of separation of variables, it can be shown that the eigenvalues of $[A_2]$ are

$$\lambda_{mn} = \frac{1}{2}\left[\cos\frac{m\pi}{M} + \cos\frac{n\pi}{N} \right], \; m = 1, \cdots, M-1, \; n = 1, \cdots, N-1$$

Therefore, $|\lambda_{mn}| \le 1$ and the Jacobi method converges.

Convergence – Jacobi Method (2)

For a large matrix

$$\lambda_{mn,\max} = \frac{1}{2}\left[\cos\frac{\pi}{M} + \cos\frac{\pi}{N}\right]$$

$$\cong 1 - \frac{1}{4}\left[\frac{\pi^2}{M^2} + \frac{\pi^2}{N^2}\right] + \cdots$$

largest eigenvalues for $m = 1,\ n = 1$

Thus, for a large matrix, $\left|\lambda_{mn,\max}\right|$ is only slightly less than unity.

$\Rightarrow$ Very slow convergence

Convergence – Gauss-Seidel (1)

Gauss-Seidel Iteration Method for Poisson Equation
(2nd order central difference with uniform mesh)

$$f_{i,j}^{n+1} = \frac{1}{4}\left[f_{i+1,j}^{n} + f_{i-1,j}^{n+1} + f_{i,j+1}^{n} + f_{i,j-1}^{n+1} - h^2 S_{i,j}\right]$$

$$[A_1] = [D] - [L], \qquad [A_2] = [U]$$

$$\begin{bmatrix} 4 & 0 & 0 & 0 & \cdots & 0 \\ -1 & 4 & 0 & 0 & \cdots & 0 \\ 0 & -1 & 4 & 0 & \cdots & 0 \\ 0 & 0 & -1 & 4 & \cdots & 0 \\ \vdots & \vdots & \vdots & & \cdots & \vdots \\ 0 & \cdots & \cdots & 0 & -1 & 4 \end{bmatrix} \begin{bmatrix} f_{1,1}^{n+1} \\ f_{1,2}^{n+1} \\ \vdots \\ \vdots \\ \vdots \\ f_{I,J}^{n+1} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & & \vdots \\ \vdots & \vdots & & & 0 & \vdots \\ 0 & 0 & 0 & 0 & \cdots & 0 \end{bmatrix} \begin{bmatrix} f_{1,1}^{n} \\ f_{1,2}^{n} \\ \vdots \\ \vdots \\ \vdots \\ f_{M,N}^{n} \end{bmatrix} - h^2 \begin{bmatrix} S_{1,1} \\ S_{1,2} \\ \vdots \\ \vdots \\ \vdots \\ S_{M,N} \end{bmatrix}$$

A_1 $\qquad$ A_2

WPI

Computational Fluid Dynamics I

Convergence – Gauss-Seidel (2)

It can be shown that the eigenvalues of matrix $[A_1]^{-1}[A_2]$ Are simply square of the eigenvalues of Jacobi method

$$\lambda_{mn,\max} = \frac{1}{4}\left[\cos\frac{\pi}{M} + \cos\frac{\pi}{N}\right]^2$$

Thus, Gauss-Seidel method is twice as fast as the Jacobi method.

WPI

Computational Fluid Dynamics I

Successive Overrelaxation - 1

Successive Overrelaxation

Consider the Gauss-Seidel method

$$\left([D]-[L]\right)\mathbf{f}^{n+1} = [U]\mathbf{f}^{n} + \mathbf{b}$$

If Gauss-Seidel is an attempt to change the solution as

$$\mathbf{f}^{n+1} = \mathbf{f}^{n} + \mathbf{d}$$

Can we accelerate the change by a parameter?

$$\mathbf{f}^{n+1} = \mathbf{f}^{n} + \beta\mathbf{d}, \ \ \beta > 1$$

WPI

Computational Fluid Dynamics I

Successive Overrelaxation - 2

Hence, SOR first uses Gauss-Seidel to compute intermediate solution, $\tilde{\mathbf{f}}$

$$\left([D]-[L]\right)\tilde{\mathbf{f}} = [U]\mathbf{f}^n + \mathbf{b} \quad \text{or} \quad [D]\tilde{\mathbf{f}} = [L]\tilde{\mathbf{f}} + [U]\mathbf{f}^n + \mathbf{b}$$

Then accelerate the next iteration solution

$$\mathbf{f}^{n+1} = \mathbf{f}^n + \beta\left(\tilde{\mathbf{f}} - \mathbf{f}^n\right) = \beta\tilde{\mathbf{f}} + (1-\beta)\mathbf{f}^n$$

Combining the two steps

$$\mathbf{f}^{n+1} = \frac{\beta}{c}\left([L]\tilde{\mathbf{f}} + [U]\mathbf{f}^n + \mathbf{b}\right) + (1-\beta)\mathbf{f}^n$$

Note that $[D] = c[I]$ $\qquad [L]\tilde{\mathbf{f}} = [L]\mathbf{f}^{n+1}$

Computational Fluid Dynamics I

Successive Overrelaxation - 3

Example: Poisson Equation (2nd order CS, uniform mesh)

$$f_{i+1,j} + f_{i-1,j} + f_{i,j+1} + f_{i,j-1} - 4f_{i,j} = h^2 S_{i,j}$$

G-S

$$[D]\tilde{\mathbf{f}} = [L]\tilde{\mathbf{f}} + [U]\mathbf{f}^n + \mathbf{b}$$

$$f_{i,j}^{n+1} = \frac{1}{4}\left[f_{i-1,j}^{n+1} + f_{i,j-1}^{n+1} + f_{i+1,j}^{n} + f_{i,j+1}^{n} - h^2 S_{i,j}\right]$$

Combining

$$\mathbf{f}^{n+1} = \frac{\beta}{c}\left([L]\tilde{\mathbf{f}} + [U]\mathbf{f}^n + \mathbf{b}\right) + (1-\beta)\mathbf{f}^n$$

$$f_{i,j}^{n+1} = \frac{\beta}{4}\left[f_{i-1,j}^{n+1} + f_{i,j-1}^{n+1} + f_{i+1,j}^{n} + f_{i,j+1}^{n} - h^2 S_{i,j}\right] + (1-\beta) f_{i,j}^{n}$$

WPI

Computational Fluid Dynamics I

Successive Overrelaxation - 4

Convergence of SOR

From $$[D]\tilde{\mathbf{f}} = [L]\tilde{\mathbf{f}} + [U]\mathbf{f}^n + \mathbf{b}$$

$$\mathbf{f}^{n+1} = \mathbf{f}^n + \beta\left(\tilde{\mathbf{f}} - \mathbf{f}^n\right)$$

Eliminating $\tilde{\mathbf{f}}$ and solving for $\mathbf{f}^{n+1}$

$$\mathbf{f}^{n+1} = \overbrace{\left([I] - \beta[D]^{-1}[L]\right)^{-1}\left\{(1-\beta)[I] + \beta[D]^{-1}[U]\right\}}^{[M]_{SOR}}\mathbf{f}^n$$

$$+\left([I] - \beta[D]^{-1}[L]\right)^{-1}\beta[D]^{-1}\mathbf{b}$$

Convergence depends on the eigenvalues of $[M]_{SOR}$

WPI

Computational Fluid Dynamics I

Successive Overrelaxation - 5

For the discretized Poisson operator, it can be shown that

$$\mu^{1/2} = \frac{1}{2}\left[\lambda\beta + \sqrt{\lambda^2\beta^2 - 4(\beta - 1)}\right]$$

where λ is an eigenvalue of the Jacobi matrix

$$[M]_J = [D]^{-1}([L] + [U])$$

Note that $\mu = \lambda^2$ if $\beta = 1$ (Gauss-Seidel)

Minimum μ occurs at

$$\beta_{\text{opt}} = \frac{2}{1 + \sqrt{1 - \lambda_{\max}^2}}$$

Typically

$$\beta_{\text{opt}} \approx 1.7 \sim 1.9$$

$$|\lambda_{\max}| \approx 1$$

Computational Fluid Dynamics I

Successive Overrelaxation - 6

For problems with irregular geometry and non-uniform mesh, β_{opt} must be found by trial and error.

Typical Comparison Chart

	$\lambda_{\max}\ (\mu_{\max})$	Iterations
Jacobi	0.9945	1250
Gauss-Seidel	0.9890	625
SOR	0.7906	29

Ferziger, J. H., *Numerical Method for Engineering Application* (1981)

WPI

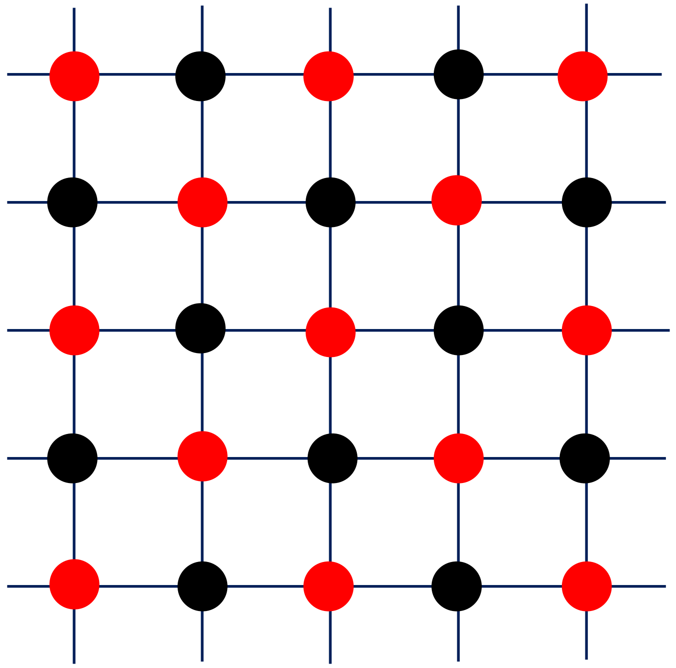
Computational Fluid Dynamics I

Coloring Scheme (Red & Black)

In large computer application (vector or parallel platform), SOR faces difficulties in using constantly updated values.

Remedy: Two separate grid system (red & black)

```
do i =1, nx, 2
    $f_{i,j} = \frac{\beta}{4}\left[f_{i-1,j} + f_{i,j-1} + f_{i+1,j} + f_{i,j+1} - h^2 S_{i,j}\right] + (1-\beta) f_{i,j}$
enddo
do i=2, nx, 2
    $f_{i,j} = \frac{\beta}{4}\left[f_{i-1,j} + f_{i,j-1} + f_{i+1,j} + f_{i,j+1} - h^2 S_{i,j}\right] + (1-\beta) f_{i,j}$
enddo
```

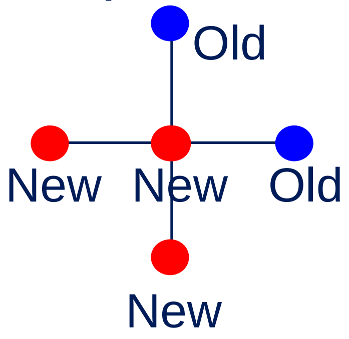


Computational Fluid Dynamics I

Successive Line Overrelaxation (SLOR) - 1

Line Relaxation Method (Line Gauss-Seidel Method)

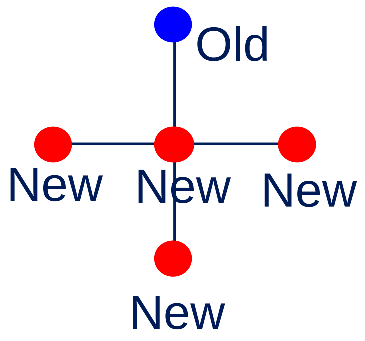
$$f_{i,j}^{n+1} = \frac{1}{4}\left[f_{i-1,j}^{n+1} + f_{i,j-1}^{n+1} + f_{i+1,j}^{n} + f_{i,j+1}^{n} - h^2 S_{i,j}\right]$$



Adding one more coupling (E)

$$f_{i,j}^{n+1} = \frac{1}{4}\left[f_{i-1,j}^{n+1} + f_{i,j-1}^{n+1} + f_{i+1,j}^{n+1} + f_{i,j+1}^{n} - h^2 S_{i,j}\right]$$

$$-\frac{1}{4} f_{i-1,j}^{n+1} + f_{i,j}^{n+1} - \frac{1}{4} f_{i+1,j}^{n+1} = \frac{1}{4}\left[f_{i,j-1}^{n+1} + f_{i,j+1}^{n} - h^2 S_{i,j}\right]$$



$\Rightarrow$ Thomas algorithm

WPI

Computational Fluid Dynamics I

Successive Line Overrelaxation (SLOR) - 2

SLOR = Line Relaxation + Overrelaxation

Apply line relaxation for intermediate solution

$$-\frac{1}{4}\tilde{f}_{i-1,j} + \tilde{f}_{i,j} - \frac{1}{4}\tilde{f}_{i+1,j} = \frac{1}{4}\left[f_{i,j-1}^{n+1} + f_{i,j+1}^{n} - h^2 S_{i,j}\right]$$

and then overrelax

$$f_{i,j}^{n+1} = \beta\,\tilde{f}_{i,j} + (1-\beta) f_{i,j}^{n}$$

which is no more complicated than line relaxation.

WPI

Successive Line Overrelaxation (SLOR) - 3

Notes on SLOR

- Exact eigenvalues are unknown.
- To ensure convergence, $\beta \leq 2$
- Converges approximately twice as fast as Gauss-Seidel.
- May be faster than pointwise SOR, but each iteration takes longer with Thomas algorithm.
- Improved convergence is due to the direct effect of the boundary condition in each row.

WPI

Computational Fluid Dynamics I

Alternating-Direction Implicit - 1

ADI for elliptic equation is analogous to ADI in parabolic equation

$$\frac{\partial f}{\partial t} = \alpha\left[\frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}\right] - S$$

In discrete form

$$f_{i,j}^{n+1} - f_{i,j}^{n} = \alpha\Delta t\left[\delta_{xx} f + \delta_{yy} f - S\right]$$

$$\delta_{xx}(\ ) = \frac{1}{\Delta x^2}\left[(\ )_{i-1,j} - 2(\ )_{i,j} + (\ )_{i+1,j}\right]$$

$$\delta_{yy}(\ ) = \frac{1}{\Delta y^2}\left[(\ )_{i,j-1} - 2(\ )_{i,j} + (\ )_{i,j+1}\right]$$

and take it to the limit to obtain the steady solution. $\left(\frac{\partial f}{\partial t} = 0\right)$

WPI

Computational Fluid Dynamics I

Alternating-Direction Implicit - 2

ADI for $$f_{i,j}^{n+1} - f_{i,j}^{n} = \alpha\Delta t\left[\delta_{xx} f + \delta_{yy} f\right] - S$$

is written as

$$f^{n+1/2} - f^{n} = \frac{\alpha\Delta t}{2h^2}\left[\left(f_{i+1,j}^{n+1/2} - 2f_{i,j}^{n+1/2} + f_{i-1,j}^{n+1/2}\right) + \left(f_{i,j+1}^{n} - 2f_{i,j}^{n} + f_{i,j-1}^{n}\right) - S_{i,j}\right]$$

$$f^{n+1} - f^{n+1/2} = \frac{\alpha\Delta t}{2h^2}\left[\left(f_{i+1,j}^{n+1/2} - 2f_{i,j}^{n+1/2} + f_{i-1,j}^{n+1/2}\right) + \left(f_{i,j+1}^{n+1} - 2f_{i,j}^{n+1} + f_{i,j-1}^{n+1}\right) - S_{i,j}\right]$$

or

$$\left(1 - \rho_n\delta_{xx}\right)f^{n+1/2} = \left(1 + \rho_n\delta_{yy}\right)f^{n} - \frac{\alpha\Delta t}{2h^2}S_{i,j}$$

$$\left(1 - \rho_n\delta_{yy}\right)f^{n+1} = \left(1 + \rho_n\delta_{xx}\right)f^{n+1/2} - \frac{\alpha\Delta t}{2h^2}S_{i,j}$$

$$\left(\rho_n = \frac{\alpha\Delta t_n}{2}\right)$$

Computational Fluid Dynamics I

Alternating-Direction Implicit - 3

Convergence of ADI

- Iteration parameter $\rho_n = \dfrac{\alpha \Delta t_n}{2}$ usually varies with iteration
- For example, Wachspress

$$\frac{\rho_k}{h^2} = \frac{\alpha \Delta t_k}{2h^2} = b\left(\frac{a}{b}\right)^{(k-1)/(n-1)}, \quad k = 1, \cdots, n$$

a lower bound eigenvalue

b upper bound eigenvalue

- Comparison with SOR is difficult
- ADI can be efficient if appropriate parameters are found.

Computational Fluid Dynamics I

Boundary Conditions for Iterative Method

Dirichlet conditions are easily implemented.

For Neumann condition, the simplest approach is

$$\frac{\partial f}{\partial n} = 0 \Rightarrow f_{i,0} - f_{i,1} = 0 \qquad \text{(1st order)}$$

Update interior points $f_{i,1}, f_{i,2}, f_{i,3}, \cdots$ and then set $f_{i,0} = f_{i,1}$

$\Rightarrow$ This generally does not converge.

Instead, incorporate BC directly into the equations

$$f_{i,1} = \frac{1}{4}\left[f_{i-1,1} + f_{i+1,1} + f_{i,2} + \overset{= f_{i,1}}{f_{i,0}} - h^2 S_{i,j}\right]$$

$$f_{i,1} = \frac{1}{3}\left[f_{i-1,1} + f_{i+1,1} + f_{i,2} - h^2 S_{i,j}\right]$$

WPI

Computational Fluid Dynamics I
Multigrid Method - 1

Basic Idea

Suppose we want to solve a 1-D elliptic equation

$$\frac{\partial^2 f}{\partial x^2} - g = 0$$

An iterative process is analogous to solving an unsteady problem (recall ADI)

$$\frac{\partial f}{\partial t} = \alpha\left(\frac{\partial^2 f}{\partial x^2} - g\right)$$

And take it to the limit $t \to \infty$

Computational Fluid Dynamics I

Multigrid Method - 2

Analytic Solution Behavior $\quad \dfrac{\partial f}{\partial t} = \alpha\left(\dfrac{\partial^2 f}{\partial x^2} - g\right)$

The equation can be solved by Fourier series

$$f = \sum_k a_k(t) e^{ikx}$$

and assume that g can be expanded as

$$g = \sum_k b_k k^2 e^{ikx}$$

Substituting into the equation,

$$\sum_k \frac{da_k}{dt} e^{ikx} = -\alpha \sum_k \left(a_k(t) - b_k\right) k^2 e^{ikx}$$

Computational Fluid Dynamics I
Multigrid Method - 3

Solving for each k

$$\frac{da_k}{dt} = -\alpha k^2 \left(a_k(t) - b_k\right)$$

$$a_k(t) - b_k = \left(a_k(0) - b_k\right) e^{-\alpha k^2 t} \qquad \left(\varepsilon_k = \varepsilon_0 e^{-\alpha k^2 t}\right)$$

Therefore, $a_k(t \to \infty) = b_k$

Rate of convergence $\sim \alpha k^2$ $\qquad \left(\tau_c = \dfrac{1}{\alpha k^2}\right)$

High wave number modes damps out faster.

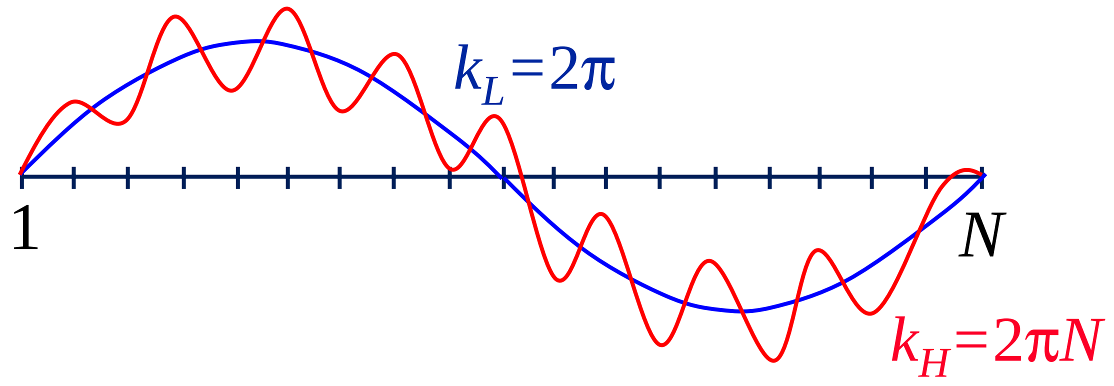
WPI

Computational Fluid Dynamics I

Multigrid Method - 4

For iterative method to solve elliptic equations, there are high wave number and low wave number components of errors that need to be damped out.

Consider a domain $L = 1$ discretized by N points



WPI

Computational Fluid Dynamics I

Multigrid Method - 5

For explicit time step, stability condition yields

$$\frac{\alpha \Delta t}{h^2} = \frac{1}{2}; \quad \Delta t = \frac{h^2}{2\alpha}$$

If the error at various wave number decays at

$$\varepsilon = \varepsilon_0 e^{-\alpha k^2 t} \qquad \text{where} \qquad t = n\Delta t = \frac{nh^2}{2\alpha}$$

$$\Rightarrow \quad \varepsilon / \varepsilon_0 = \exp\left(-k^2 n h^2 / 2\right)$$

$$\frac{\varepsilon_H}{\varepsilon_0} = \exp\left(-n\pi^2 N^2 h^2\right) \text{ for } k_H = 2\pi N$$

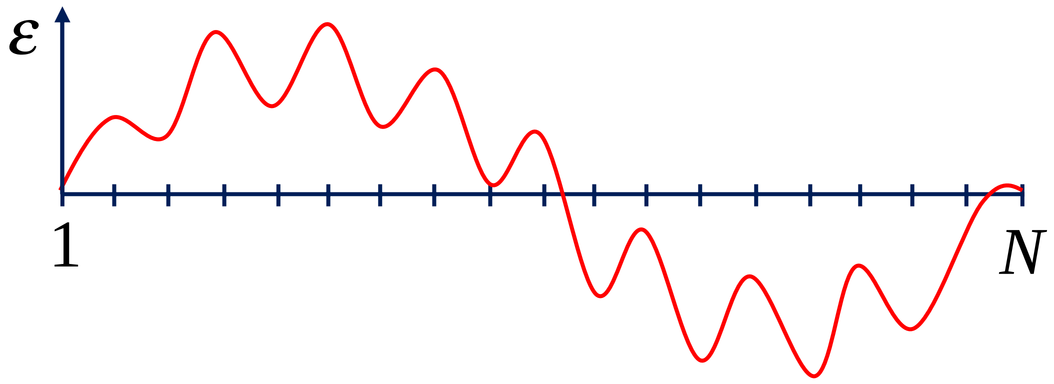
$$\frac{\varepsilon_L}{\varepsilon_0} = \exp\left(-n\pi^2 h^2\right) \text{ for } k_L = 2\pi$$

Short-wave errors decay much faster

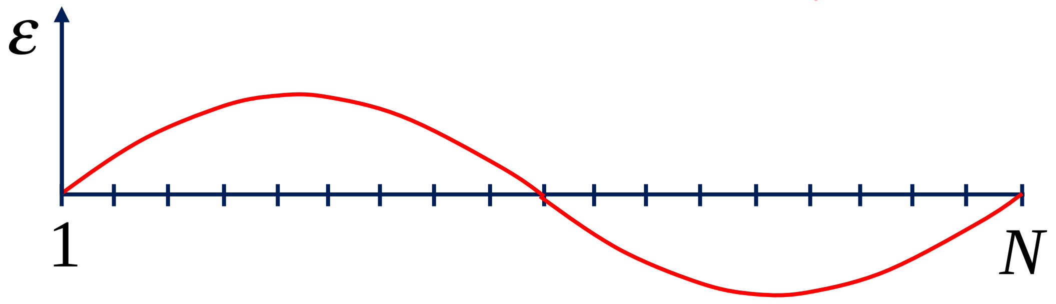
Computational Fluid Dynamics I

Multigrid Method - 6

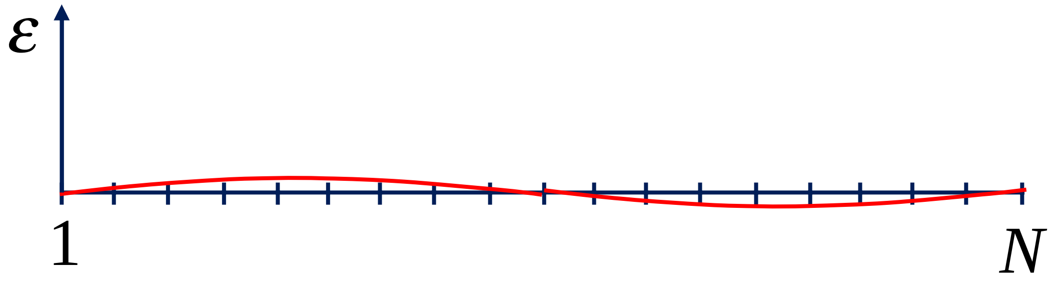
$t = 0$



$t = 5\Delta t$



$t = 500\Delta t$



Multigrid Method – the Idea

- A low wave number component on a fine grid becomes a high wave number component on a coarse grid

$$\varepsilon_L / \varepsilon_0 = \exp\left(-n\pi^2 h^2\right) = \exp\left(-n\pi^2 L^2 / N^2\right)$$

- Use a coarse grid system to converge low wave number component of the solution rapidly
- Map it onto the fine grid system to converge high wave number component.

Computational Fluid Dynamics I

Multigrid Method - 8

Suppose we are solving a Laplace equation:

$$Lf_{i,j} = \frac{f_{i+1,j} - 2f_{i,j} + f_{i-1,j}}{\Delta x^2} + \frac{f_{i,j+1} - 2f_{i,j} + f_{i,j-1}}{\Delta y^2} = R_{i,j}$$

and iterate until residual becomes smaller than tolerance

$$R_{i,j} \to 0 \quad \text{or} \quad R_{i,j} < \varepsilon$$

Define

$$f_{i,j} = f_{i,j}^n + \Delta f_{i,j}$$

$f_{i,j}$	Converged solution
$f_{i,j}^n$	Solution after n-th iteration
$\Delta f_{i,j}$	Correction

Computational Fluid Dynamics I

Multigrid Method - 9

Since

$$Lf_{i,j} = Lf_{i,j}^{n} + L\Delta f_{i,j} = 0$$

and

$$Lf_{i,j}^{n} = R_{i,j}$$

we have

$$L\Delta f_{i,j} = -R_{i,j}$$ Coarse grid correction

Steps:

Fine grid solution
$\Rightarrow$ Interpolation onto coarse grid (restriction)
$\Rightarrow$ Coarse grid correction
$\Rightarrow$ Interpolated onto fine grid (prolongation)
$\Rightarrow$ Converge on fine grid

Computational Fluid Dynamics I

Multigrid Method - 10

Two-Level Multigrid Method - (nx+1, ny+1); (nx/2+1, ny/2+1)

1. Do *n* iterations on the fine grid (*n* = 3~4) using G-S

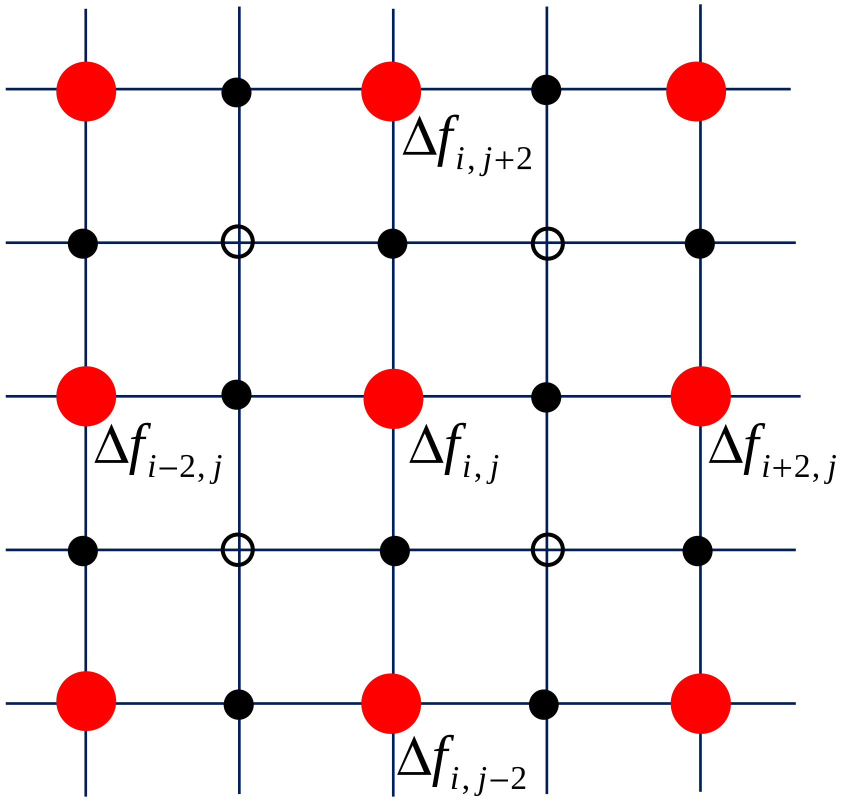
$$Lf_{i,j}^{n} = R_{i,j}$$

2. If $R_{i,j} < \varepsilon$ then stop.
3. Interpolate the residual $R_{i,j}$ onto the coarse grid $I_1^2 R_{i,j}$ (restriction (injection if nx/2))
4. Iterations on the coarse grid

$$L\Delta(f_2)_{i,j} = -I_1^2 R_{i,j}$$

5. Interpolate the correction $\Delta(f_2)_{i,j}$ onto the fine grid (prolongation) $f_{i,j} = f_{i,j}^{n} + I_2^1 \Delta(f_2)_{i,j}$ $\quad I_2^1 \Delta(f_2)_{i,j}$
6. Go to 1.

Details on Prolongation



Coarse Grid

Fine grid

$$\Delta f_{i+1,j} = \frac{1}{2}\left(\Delta f_{i,j} + \Delta f_{i+2,j}\right)$$

$$\Delta f_{i,j+1} = \frac{1}{2}\left(\Delta f_{i,j} + \Delta f_{i,j+2}\right)$$

$$\Delta f_{i+1,j+1}$$

$$= \frac{1}{4}\left(\Delta f_{i+1,j} + \Delta f_{i,j+1} + \Delta f_{i+1,j+2} + \Delta f_{i+2,j+1}\right)$$

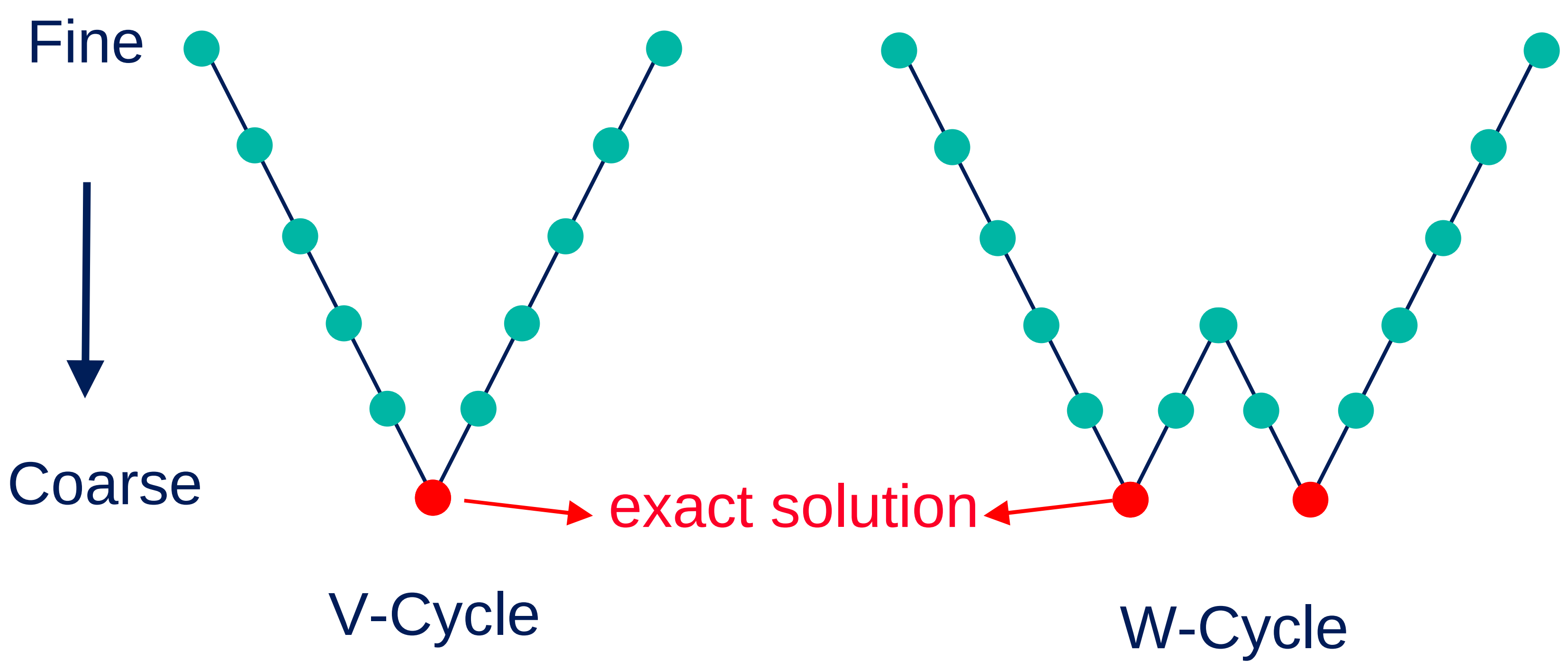
$$= \frac{1}{4}\left(\Delta f_{i,j} + \Delta f_{i+2,j} + \Delta f_{i,j+2} + \Delta f_{i+2,j+2}\right)$$

Computational Fluid Dynamics I

Multigrid Method - 12

The process can be extended to multi-level grids
(nx+1,ny+1), (nx/2+1,ny/2+1), (nx/4+1,ny/4+1), …, (3,3)

Various Strategies



WPI

Computational Fluid Dynamics I

Multigrid Method - 13

Example: 4-Level V-Cycle

Fine $\Rightarrow$ Coarse

1. Iterate on Grid 1 for n times $Lf_{i,j}^{n}=0 \quad \Rightarrow R_{i,j}^{1}$
2. Restriction by injection to Grid 2 $I_1^2 R_{i,j}^{1}$
3. Iterate on Grid 2 $L\Delta(f_2)_{i,j} = -I_1^2 R_{i,j}^{1}$

   $R_{i,j}^{2} = I_1^2 R_{i,j}^{1} + L\Delta(f_2)_{i,j}^{n}$ (Save $\Delta(f_2)_{i,j}, I_1^2 R_{i,j}^{1}$)
4. Restriction by injection to Grid 3 $I_2^3 R_{i,j}^{2}$
5. Iterate on Grid 3 $L\Delta(f_3)_{i,j} = -I_2^3 R_{i,j}^{2}$

   $R_{i,j}^{3} = I_2^3 R_{i,j}^{2} + L\Delta(f_3)_{i,j}^{n}$ (Save $\Delta(f_3)_{i,j}, I_2^3 R_{i,j}^{2}$)
6. Restriction by injection to Grid 4 $I_3^4 R_{i,j}^{3}$
7. Iterate on Grid 4 $L\Delta(f_4)_{i,j} = -I_3^4 R_{i,j}^{3}$

Coarse $\Rightarrow$ Fine

1. Prolongate from Grid 4 to Grid 3 $I_4^3 \Delta(f_4)_{i,j}^n$

$$\Delta(f_3)_{i,j}^n = \Delta(f_3)_{i,j}^n + I_4^3 \Delta(f_4)_{i,j}^n$$

2. Iterate on Grid 3 $L\Delta(f_3)_{i,j} = -I_2^3 R_{i,j}^2$ (saved)
3. Prolongate from Grid 3 to Grid 2 $I_3^2 \Delta(f_3)_{i,j}^n$

$$\Delta(f_2)_{i,j}^n = \Delta(f_2)_{i,j}^n + I_3^2 \Delta(f_3)_{i,j}^n$$

4. Iterate on Grid 2 $L\Delta(f_2)_{i,j} = -I_1^2 R_{i,j}^1$ (saved)
5. Prolongate from Grid 2 to Grid 1 $I_2^1 \Delta(f_2)_{i,j}^n$

$$f_{i,j}^n = f_{i,j}^n + I_2^1 \Delta(f_2)_{i,j}^n$$

6. Iterate on Grid 1 $Lf_{i,j} = 0$
7. If $R_{i,j} < \varepsilon$ then stop. Else, repeat the entire cycle.

Multigrid Method - 15

Test Problem (Tannehill, p. 170)

- Laplace equation in a square domain
- Dirichlet conditions on four boundaries
- 5 Levels of resolution

  9×9, 17×17, 33×33, 65×65, 129×129
- 4 Methods

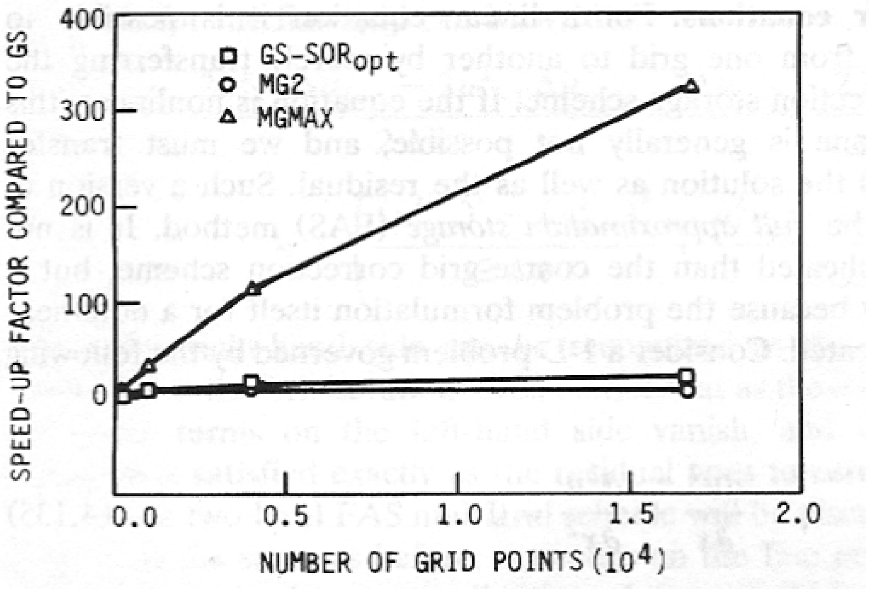
  1. Conventional Gauss-Seidel (GS)
  2. Gauss-Seidel-SOR with optimal ω (GS_{opt})
  3. Multigrid with 2-level grids (MG2)
  4. Multigrid with maximum levels of grid (MGMAX)

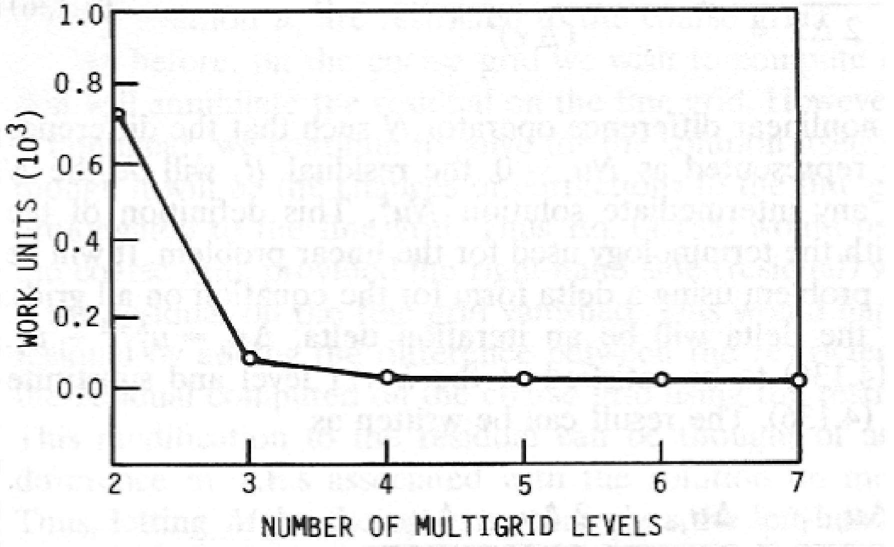
WPI

Computational Fluid Dynamics I

Multigrid Method - 16

Grid size	GS	$GS\omega_{opt}$	MG2	MGMAX
9×9	62	19	19	17
17×17	215	40	40	19
33×33	715	75	95	20
65×65	2282	137	258	20
129×129	6826	282	732	21





Computational Fluid Dynamics I

Multigrid Method - 17

The multigrid method is the fastest technique currently available for general elliptic equations, and are said to compare favorably with the fastest direct solvers for special problems.

MUDPACK

http://www.scd.ucar.edu/css/software/mudpack/

FISHPACK

http://www.scd.ucar.edu/css/software/fishpack/