Partial Differential Equations

- Types of PDEs: hyperbolic, parabolic, elliptic; initial value (Cauchy) or boundary value problems.

Partial Differential Equations

- Types of PDEs: hyperbolic, parabolic, elliptic; initial value (Cauchy) or boundary value problems.
- Hyperbolic: wave equation $\frac{1}{c^2}\frac{\partial^2\psi}{\partial t^2} = \frac{\partial^2\psi}{\partial x^2}$; advection equation $\frac{\partial\phi}{\partial t} + v\frac{\partial\phi}{\partial x} = 0$; EM, GR field equations.

Partial Differential Equations

- Types of PDEs: hyperbolic, parabolic, elliptic; initial value (Cauchy) or boundary value problems.
- Hyperbolic: wave equation $\frac{1}{c^2}\frac{\partial^2\psi}{\partial t^2} = \frac{\partial^2\psi}{\partial x^2}$; advection equation $\frac{\partial\phi}{\partial t} + v\frac{\partial\phi}{\partial x} = 0$; EM, GR field equations.
- Parabolic: diffusion equation $\frac{\partial\psi}{\partial t} = \frac{\partial}{\partial x}\left(D\frac{\partial\psi}{\partial x}\right)$.

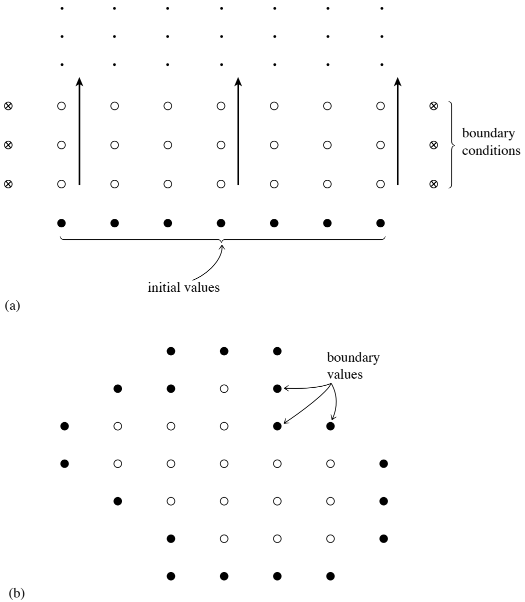
Partial Differential Equations

- Types of PDEs: hyperbolic, parabolic, elliptic; initial value (Cauchy) or boundary value problems.
- Hyperbolic: wave equation $\frac{1}{c^2}\frac{\partial^2\psi}{\partial t^2} = \frac{\partial^2\psi}{\partial x^2}$; advection equation $\frac{\partial\phi}{\partial t} + v\frac{\partial\phi}{\partial x} = 0$; EM, GR field equations.
- Parabolic: diffusion equation $\frac{\partial\psi}{\partial t} = \frac{\partial}{\partial x}\left(D\frac{\partial\psi}{\partial x}\right)$.
- Elliptic: Poisson equation $\nabla^2\psi = \rho$.

Partial Differential Equations

- Types of PDEs: hyperbolic, parabolic, elliptic; initial value (Cauchy) or boundary value problems.
- Hyperbolic: wave equation $\frac{1}{c^2}\frac{\partial^2\psi}{\partial t^2} = \frac{\partial^2\psi}{\partial x^2}$; advection equation $\frac{\partial\phi}{\partial t} + v\frac{\partial\phi}{\partial x} = 0$; EM, GR field equations.
- Parabolic: diffusion equation $\frac{\partial\psi}{\partial t} = \frac{\partial}{\partial x}\left(D\frac{\partial\psi}{\partial x}\right)$.
- Elliptic: Poisson equation $\nabla^2\psi = \rho$.
- Of course these are supplemented by appropriate initial/boundary conditions.

Initial & Boundary value problems



Boundary value problems

- Model problem: 2-D Poisson equation $\nabla^2 u = \rho$

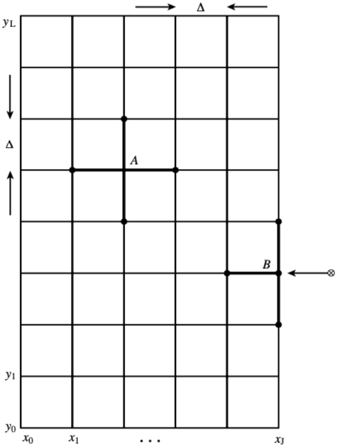
Boundary value problems

- Model problem: 2-D Poisson equation $\nabla^2 u = \rho$
- finite difference approx.

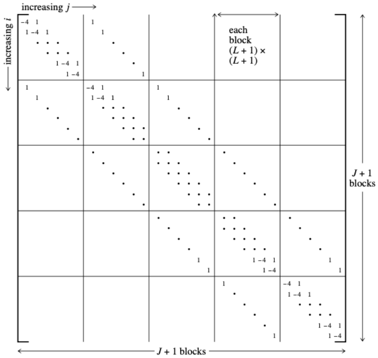
  $u_{j+1,l} + u_{j-1,l} + u_{j,l+1} + u_{j,l-1} - 4u_{j,l} = \Delta^2 \rho_{j,l}$

Boundary value problems

- Model problem: 2-D Poisson equation $\nabla^2 u = \rho$
- finite difference approx.
  $u_{j+1,l} + u_{j-1,l} + u_{j,l+1} + u_{j,l-1} - 4u_{j,l} = \Delta^2 \rho_{j,l}$
- can translate it into a matrix equation $\mathbf{A} \cdot \mathbf{u} = \mathbf{b}$; boundary terms taken to RHS

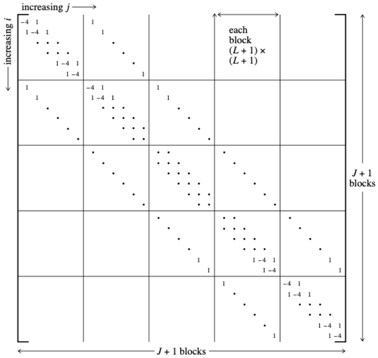


- Matrix structure



Boundary value problems

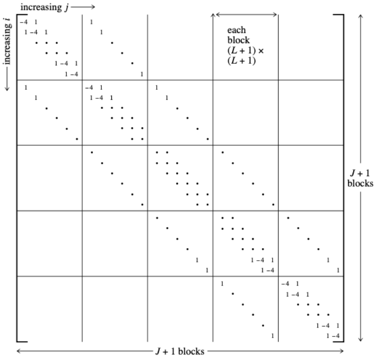
- Matrix structure



- tridiagonal with fringes; can solve directly ($\mathcal{O}[N^{3d}]$ for d dimensions) or iteratively (as fast as $\mathcal{O}[N^d \ln N]$)

Boundary value problems

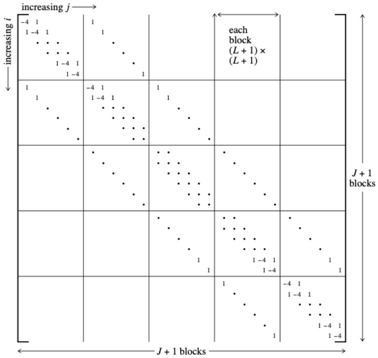
- Matrix structure



- tridiagonal with fringes; can solve directly ($\mathcal{O}[N^{3d}]$ for d dimensions) or iteratively (as fast as $\mathcal{O}[N^d \ln N]$)
- latter preferred as matrix is sparse

Boundary value problems

- Matrix structure

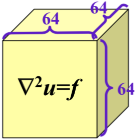


- tridiagonal with fringes; can solve directly ($\mathcal{O}[N^{3d}]$ for d dimensions) or iteratively (as fast as $\mathcal{O}[N^d \ln N]$)
- latter preferred as matrix is sparse
- Fourier methods for periodic boundary conditions $\mathcal{O}[N^d \ln N]$)

The power of optimal algorithms

- Advances in algorithmic efficiency can rival advances in hardware architecture
- Consider Poisson's equation on a cube of size $N=n^3$

Year	*Method*	*Reference*	*Storage*	*Flops*
1947	GE (banded)	Von Neumann & Goldstine	n^5	n^7
1950	Optimal SOR	Young	n^3	$n^4 \log n$
1971	CG	Reid Conjugate Gradients w/ Gustaffson's modified ILU preconditioner	n^3	$n^{3.5} \log n$
1984	Full MG	Brandt Multigrid	n^3	n^3



- If n=64, this implies an overall reduction in flops of ~16 million *

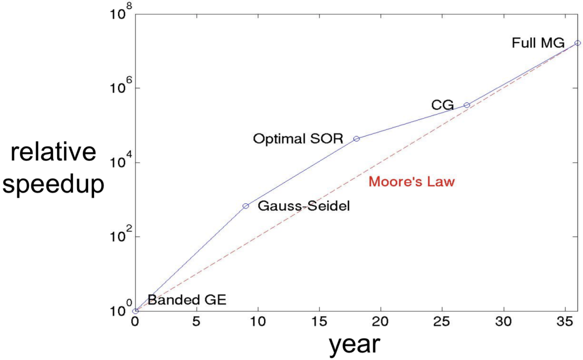
more recently: FMM Fast Multipole Method

*Six-months is reduced to 1 s

PPPL Colloquium, 25 Jan 2006
David Keyes, Columbia Univ.

Algorithms and Moore's Law

- This advance took place over a span of about 36 years, or 24 doubling times for Moore's Law
- $2^{24} \approx 16$ million $\Rightarrow$ the same as the factor from algorithms alone!



- $\mathbf{A} \cdot \mathbf{u} = \mathbf{b}$; split $\mathbf{A} = \mathbf{E} - \mathbf{F}$ where $\mathbf{E}$ is easily invertible

- $\mathbf{A} \cdot \mathbf{u} = \mathbf{b}$; split $\mathbf{A} = \mathbf{E} - \mathbf{F}$ where $\mathbf{E}$ is easily invertible
- $\mathbf{E} \cdot \mathbf{u} = \mathbf{F} \cdot \mathbf{u} + \mathbf{b}$ suggests an iterative scheme
  $\mathbf{E} \cdot \mathbf{u}^n = \mathbf{F} \cdot \mathbf{u}^{n-1} + \mathbf{b}$

Relaxation Methods: Jacobi, Gauss-Seidel

- $\mathbf{A} \cdot \mathbf{u} = \mathbf{b}$; split $\mathbf{A} = \mathbf{E} - \mathbf{F}$ where $\mathbf{E}$ is easily invertible
- $\mathbf{E} \cdot \mathbf{u} = \mathbf{F} \cdot \mathbf{u} + \mathbf{b}$ suggests an iterative scheme $\mathbf{E} \cdot \mathbf{u}^n = \mathbf{F} \cdot \mathbf{u}^{n-1} + \mathbf{b}$
- relaxation $x = f(x)$; $x^{n+1} = f(x^n)$; needn't converge

Relaxation Methods: Jacobi, Gauss-Seidel

- $\mathbf{A} \cdot \mathbf{u} = \mathbf{b}$; split $\mathbf{A} = \mathbf{E} - \mathbf{F}$ where $\mathbf{E}$ is easily invertible
- $\mathbf{E} \cdot \mathbf{u} = \mathbf{F} \cdot \mathbf{u} + \mathbf{b}$ suggests an iterative scheme
  $\mathbf{E} \cdot \mathbf{u}^n = \mathbf{F} \cdot \mathbf{u}^{n-1} + \mathbf{b}$
- relaxation $x = f(x)$; $x^{n+1} = f(x^n)$; needn't converge
- $\mathbf{A} = \mathbf{L} + \mathbf{D} + \mathbf{U}$ sum of lower, diagonal, upper matrices

- $\mathbf{A} \cdot \mathbf{u} = \mathbf{b}$; split $\mathbf{A} = \mathbf{E} - \mathbf{F}$ where $\mathbf{E}$ is easily invertible
- $\mathbf{E} \cdot \mathbf{u} = \mathbf{F} \cdot \mathbf{u} + \mathbf{b}$ suggests an iterative scheme $\mathbf{E} \cdot \mathbf{u}^n = \mathbf{F} \cdot \mathbf{u}^{n-1} + \mathbf{b}$
- relaxation $x = f(x)$; $x^{n+1} = f(x^n)$; needn't converge
- $\mathbf{A} = \mathbf{L} + \mathbf{D} + \mathbf{U}$ sum of lower, diagonal, upper matrices
- **Jacobi:** $\mathbf{D} \cdot \mathbf{x}^n = -(\mathbf{L} + \mathbf{U}) \cdot \mathbf{x}^{n-1} + \mathbf{b}$
- **Gauss-Seidel:** $(\mathbf{L} + \mathbf{D}) \cdot \mathbf{x}^n = -\mathbf{U} \cdot \mathbf{x}^{n-1} + \mathbf{b}$

Relaxation Methods: Jacobi, Gauss-Seidel

- $\mathbf{A} \cdot \mathbf{u} = \mathbf{b}$; split $\mathbf{A} = \mathbf{E} - \mathbf{F}$ where $\mathbf{E}$ is easily invertible
- $\mathbf{E} \cdot \mathbf{u} = \mathbf{F} \cdot \mathbf{u} + \mathbf{b}$ suggests an iterative scheme $\mathbf{E} \cdot \mathbf{u}^n = \mathbf{F} \cdot \mathbf{u}^{n-1} + \mathbf{b}$
- relaxation $x = f(x)$; $x^{n+1} = f(x^n)$; needn't converge
- $\mathbf{A} = \mathbf{L} + \mathbf{D} + \mathbf{U}$ sum of lower, diagonal, upper matrices
- **Jacobi:** $\mathbf{D} \cdot \mathbf{x}^n = -(\mathbf{L} + \mathbf{U}) \cdot \mathbf{x}^{n-1} + \mathbf{b}$
- $u^n_{j,l} = \frac{1}{4}(u^{n-1}_{j+1,l} + u^{n-1}_{j-1,l} + u^{n-1}_{j,l+1} + u^{n-1}_{j,l-1}) + \frac{\Delta^2}{4}\rho_{j,l}$
- **Gauss-Seidel:** $(\mathbf{L} + \mathbf{D}) \cdot \mathbf{x}^n = -\mathbf{U} \cdot \mathbf{x}^{n-1} + \mathbf{b}$
- $u^n_{j,l} = \frac{1}{4}(u^{n-1}_{j+1,l} + u^{n}_{j-1,l} + u^{n-1}_{j,l+1} + u^{n}_{j,l-1}) + \frac{\Delta^2}{4}\rho_{j,l}$

Relaxation Methods: Jacobi, Gauss-Seidel

- $\mathbf{A} \cdot \mathbf{u} = \mathbf{b}$; split $\mathbf{A} = \mathbf{E} - \mathbf{F}$ where $\mathbf{E}$ is easily invertible
- $\mathbf{E} \cdot \mathbf{u} = \mathbf{F} \cdot \mathbf{u} + \mathbf{b}$ suggests an iterative scheme $\mathbf{E} \cdot \mathbf{u}^n = \mathbf{F} \cdot \mathbf{u}^{n-1} + \mathbf{b}$
- relaxation $x = f(x)$; $x^{n+1} = f(x^n)$; needn't converge
- $\mathbf{A} = \mathbf{L} + \mathbf{D} + \mathbf{U}$ sum of lower, diagonal, upper matrices
- **Jacobi:** $\mathbf{D} \cdot \mathbf{x}^n = -(\mathbf{L} + \mathbf{U}) \cdot \mathbf{x}^{n-1} + \mathbf{b}$
- $u^n_{j,l} = \frac{1}{4}(u^{n-1}_{j+1,l} + u^{n-1}_{j-1,l} + u^{n-1}_{j,l+1} + u^{n-1}_{j,l-1}) + \frac{\Delta^2}{4}\rho_{j,l}$
- **Gauss-Seidel:** $(\mathbf{L} + \mathbf{D}) \cdot \mathbf{x}^n = -\mathbf{U} \cdot \mathbf{x}^{n-1} + \mathbf{b}$
- $u^n_{j,l} = \frac{1}{4}(u^{n-1}_{j+1,l} + u^{n}_{j-1,l} + u^{n-1}_{j,l+1} + u^{n}_{j,l-1}) + \frac{\Delta^2}{4}\rho_{j,l}$
- no variables from previous iteration; just use current value

Convergence of Jacobi, Gauss-Seidel

- **error** $\mathbf{e}^n = \mathbf{x} - \mathbf{x}^n$; **residual** $\mathbf{r}^n = \mathbf{b} - \mathbf{A} \cdot \mathbf{x}^n$;
  $\mathbf{A} \cdot \mathbf{e}^n = \mathbf{r}^n$

Convergence of Jacobi, Gauss-Seidel

- **error** $\mathbf{e}^n = \mathbf{x} - \mathbf{x}^n$; **residual** $\mathbf{r}^n = \mathbf{b} - \mathbf{A} \cdot \mathbf{x}^n$; $\mathbf{A} \cdot \mathbf{e}^n = \mathbf{r}^n$
- error satisfies the residual equation: $\mathbf{A} \cdot \mathbf{e}^n = \mathbf{r}^n$
- residual vanishes if error vanishes

Convergence of Jacobi, Gauss-Seidel

- **error** $\mathbf{e}^n = \mathbf{x} - \mathbf{x}^n$; **residual** $\mathbf{r}^n = \mathbf{b} - \mathbf{A} \cdot \mathbf{x}^n$; $\mathbf{A} \cdot \mathbf{e}^n = \mathbf{r}^n$
- error satisfies the residual equation: $\mathbf{A} \cdot \mathbf{e}^n = \mathbf{r}^n$
- residual vanishes if error vanishes
- $\mathbf{x}^n = \mathbf{P} \cdot \mathbf{x}^{n-1} + \mathbf{g}$ implies $\mathbf{e}^n = \mathbf{P} \cdot \mathbf{e}^{n-1}$
- error amplification factor determined by eigenvalues of $\mathbf{P}$
- **spectral radius** $\rho_P = \max|\lambda_i(P)| < 1$ for convergence
- smallest factor by which error reduces in an iteration

Convergence of Jacobi, Gauss-Seidel

- **error** $\mathbf{e}^n = \mathbf{x} - \mathbf{x}^n$; **residual** $\mathbf{r}^n = \mathbf{b} - \mathbf{A} \cdot \mathbf{x}^n$; $\mathbf{A} \cdot \mathbf{e}^n = \mathbf{r}^n$
- error satisfies the residual equation: $\mathbf{A} \cdot \mathbf{e}^n = \mathbf{r}^n$
- residual vanishes if error vanishes
- $\mathbf{x}^n = \mathbf{P} \cdot \mathbf{x}^{n-1} + \mathbf{g}$ implies $\mathbf{e}^n = \mathbf{P} \cdot \mathbf{e}^{n-1}$
- error amplification factor determined by eigenvalues of $\mathbf{P}$
- **spectral radius** $\rho_P = \max|\lambda_i(P)| < 1$ for convergence
- smallest factor by which error reduces in an iteration
- Jacobi: $\mathbf{P} = -\mathbf{D}^{-1}(\mathbf{L} + \mathbf{U})$

- **error** $\mathbf{e}^n = \mathbf{x} - \mathbf{x}^n$; **residual** $\mathbf{r}^n = \mathbf{b} - \mathbf{A} \cdot \mathbf{x}^n$; $\mathbf{A} \cdot \mathbf{e}^n = \mathbf{r}^n$
- error satisfies the residual equation: $\mathbf{A} \cdot \mathbf{e}^n = \mathbf{r}^n$
- residual vanishes if error vanishes
- $\mathbf{x}^n = \mathbf{P} \cdot \mathbf{x}^{n-1} + \mathbf{g}$ implies $\mathbf{e}^n = \mathbf{P} \cdot \mathbf{e}^{n-1}$
- error amplification factor determined by eigenvalues of $\mathbf{P}$
- **spectral radius** $\rho_P = \max|\lambda_i(P)| < 1$ for convergence
- smallest factor by which error reduces in an iteration
- Jacobi: $\mathbf{P} = -\mathbf{D}^{-1}(\mathbf{L} + \mathbf{U})$
- Gauss-Seidel: $\mathbf{P} = -(\mathbf{L} + \mathbf{D})^{-1}\mathbf{U}$

Analysis of 1-D Poisson equation

- Jacobi: $\phi_i^{n+1} = (\phi_{i-1}^n + \phi_{i+1}^n)/2 - \Delta^2 \rho_i/2$
- matrix form: $P = I - \frac{1}{2}A$ with

$$A = \begin{bmatrix} 2 & -1 & & & & \\ -1 & 2 & -1 & & & \\ & . & . & . & & \\ & & & -1 & 2 & -1 \\ & & & & -1 & 2 \end{bmatrix}$$

- eigenvalues of A: $\lambda_k(A) = 4\sin^2(\frac{k\pi}{2N})$, $1 \leq k \leq N-1$
- eigenvalues of P: $\lambda_k(P) = 1 - 2\sin^2(\frac{k\pi}{2N})$, $1 \leq k \leq N-1$

Analysis of 1-D Poisson equation

- Jacobi: $\phi_i^{n+1} = (\phi_{i-1}^n + \phi_{i+1}^n)/2 - \Delta^2 \rho_i/2$
- matrix form: $P = I - \frac{1}{2}A$ with

$$A = \begin{bmatrix} 2 & -1 & & & & \\ -1 & 2 & -1 & & & \\ & . & . & . & & \\ & & & -1 & 2 & -1 \\ & & & & -1 & 2 \end{bmatrix}$$

- eigenvalues of A: $\lambda_k(A) = 4\sin^2(\frac{k\pi}{2N})$, $1 \le k \le N-1$
- eigenvalues of P: $\lambda_k(P) = 1 - 2\sin^2(\frac{k\pi}{2N})$, $1 \le k \le N-1$
- Gauss-Seidel: $\phi_i^{n+1} = (\phi_{i-1}^{n+1} + \phi_{i+1}^n)/2 - \Delta^2 \rho_i/2$
- eigenvalues of P: $\lambda_k(P) = \cos^2(\frac{k\pi}{N})$, $1 \le k \le N-1$

Analysis of 1-D Poisson equation

- Jacobi: $\phi_i^{n+1} = (\phi_{i-1}^n + \phi_{i+1}^n)/2 - \Delta^2 \rho_i/2$
- matrix form: $P = I - \frac{1}{2}A$ with

$$A = \begin{bmatrix} 2 & -1 & & & & \\ -1 & 2 & -1 & & & \\ & . & . & . & & \\ & & & -1 & 2 & -1 \\ & & & & -1 & 2 \end{bmatrix}$$

- eigenvalues of A: $\lambda_k(A) = 4\sin^2(\frac{k\pi}{2N})$, $1 \le k \le N-1$
- eigenvalues of P: $\lambda_k(P) = 1 - 2\sin^2(\frac{k\pi}{2N})$, $1 \le k \le N-1$
- Gauss-Seidel: $\phi_i^{n+1} = (\phi_{i-1}^{n+1} + \phi_{i+1}^n)/2 - \Delta^2 \rho_i/2$
- eigenvalues of P: $\lambda_k(P) = \cos^2(\frac{k\pi}{N})$, $1 \le k \le N-1$
- spectral radius $\simeq 1 - \pi^2/2N^2,\ 1 - \pi^2/N^2$ in two cases

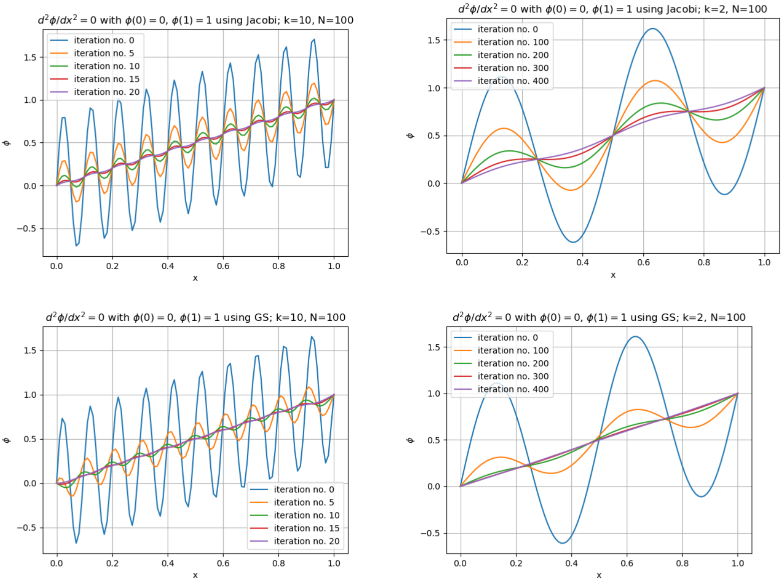
Analysis of 1-D Poisson equation

- Jacobi: $\phi_i^{n+1} = (\phi_{i-1}^n + \phi_{i+1}^n)/2 - \Delta^2 \rho_i/2$
- matrix form: $P = I - \frac{1}{2}A$ with

$$A = \begin{bmatrix} 2 & -1 & & & & \\ -1 & 2 & -1 & & & \\ & . & . & . & & \\ & & & -1 & 2 & -1 \\ & & & & -1 & 2 \end{bmatrix}$$

- eigenvalues of A: $\lambda_k(A) = 4\sin^2(\frac{k\pi}{2N})$, $1 \le k \le N-1$
- eigenvalues of P: $\lambda_k(P) = 1 - 2\sin^2(\frac{k\pi}{2N})$, $1 \le k \le N-1$
- Gauss-Seidel: $\phi_i^{n+1} = (\phi_{i-1}^{n+1} + \phi_{i+1}^n)/2 - \Delta^2 \rho_i/2$
- eigenvalues of P: $\lambda_k(P) = \cos^2(\frac{k\pi}{N})$, $1 \le k \le N-1$
- spectral radius $\simeq 1 - \pi^2/2N^2,\ 1 - \pi^2/N^2$ in two cases
- lowest k converge slowest, worse for large N; key idea of MG methods

1-D Poisson equation: numerical experiments



Successive Overrelaxation (SOR)

- **Gauss-Seidel:**
  $\mathbf{x}^n = -(\mathbf{L}+\mathbf{D})^{-1}\mathbf{U}\cdot\mathbf{x}^{n-1} + (\mathbf{L}+\mathbf{D})^{-1}\cdot\mathbf{b}$
- rearrange, $\mathbf{x}^n = \mathbf{x}^{n-1} - (\mathbf{L}+\mathbf{D})^{-1}(\mathbf{A}\cdot\mathbf{x}^{n-1} - \mathbf{b})$

Successive Overrelaxation (SOR)

- **Gauss-Seidel:**
  $\mathbf{x}^n = -(\mathbf{L}+\mathbf{D})^{-1}\mathbf{U}\cdot\mathbf{x}^{n-1} + (\mathbf{L}+\mathbf{D})^{-1}\cdot\mathbf{b}$
- rearrange, $\mathbf{x}^n = \mathbf{x}^{n-1} - (\mathbf{L}+\mathbf{D})^{-1}(\mathbf{A}\cdot\mathbf{x}^{n-1} - \mathbf{b})$
- overcorrect, $\mathbf{x}^n = \mathbf{x}^{n-1} + \omega(\mathbf{L}+\mathbf{D})^{-1}\cdot\mathbf{r}^{n-1}$

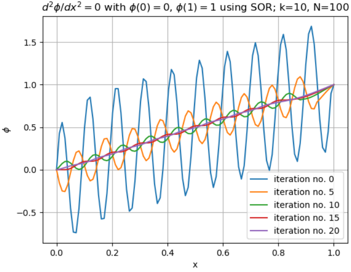
Successive Overrelaxation (SOR)

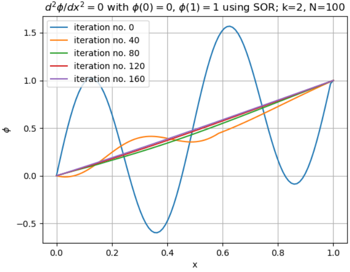
- **Gauss-Seidel:**
  $\mathbf{x}^n = -(\mathbf{L}+\mathbf{D})^{-1}\mathbf{U}\cdot\mathbf{x}^{n-1} + (\mathbf{L}+\mathbf{D})^{-1}\cdot\mathbf{b}$
- rearrange, $\mathbf{x}^n = \mathbf{x}^{n-1} - (\mathbf{L}+\mathbf{D})^{-1}(\mathbf{A}\cdot\mathbf{x}^{n-1} - \mathbf{b})$
- overcorrect, $\mathbf{x}^n = \mathbf{x}^{n-1} + \omega(\mathbf{L}+\mathbf{D})^{-1}\cdot\mathbf{r}^{n-1}$
- relaxation parameter ω; both error and residue should converge to 0
- convergence if $0 < \omega < 2$; $1 < \omega < 2$ faster than GS

Successive Overrelaxation (SOR)

- **Gauss-Seidel:**
  $\mathbf{x}^n = -(\mathbf{L}+\mathbf{D})^{-1}\mathbf{U}\cdot\mathbf{x}^{n-1} + (\mathbf{L}+\mathbf{D})^{-1}\cdot\mathbf{b}$
- rearrange, $\mathbf{x}^n = \mathbf{x}^{n-1} - (\mathbf{L}+\mathbf{D})^{-1}(\mathbf{A}\cdot\mathbf{x}^{n-1} - \mathbf{b})$
- overcorrect, $\mathbf{x}^n = \mathbf{x}^{n-1} + \omega(\mathbf{L}+\mathbf{D})^{-1}\cdot\mathbf{r}^{n-1}$
- relaxation parameter ω; both error and residue should converge to 0
- convergence if $0<\omega<2$; $1<\omega<2$ faster than GS
- optimum $\omega = \frac{2}{1+\sqrt{1-\rho_J^2}}$
- for Poisson eq. $\rho_{\mathrm{SOR}} \simeq 1-2\pi/N$ for large N
- much faster than GS with $\rho_{\mathrm{GS}} \simeq 1-\pi^2/N^2$

1-D Poisson equation: numerical experiments





- note faster convergence, especially for low k
- here we used optimum ω for a given k
- worse performance for non-optimum ω
- ω determined by trial and error, local approximation

Relaxation as a steady state

- $\frac{\partial u}{\partial t} = \mathcal{L}u - \rho$
- $\mathcal{L}u = \rho$ in steady state, a solution to elliptic equation

Relaxation as a steady state

- $\frac{\partial u}{\partial t} = \mathcal{L}u - \rho$
- $\mathcal{L}u = \rho$ in steady state, a solution to elliptic equation
- FTCS $u_{j,l}^{n+1} =$
  $u_{j,l}^{n} + \frac{\Delta t}{\Delta^2}(u_{j+1,l}^{n} + u_{j-1,l}^{n} + u_{j,l+l}^{n} + u_{j,l-1}^{n} - 4u_{j,l}^{n}) - \rho_{j,l}\Delta t$

Relaxation as a steady state

- $\frac{\partial u}{\partial t} = \mathcal{L}u - \rho$
- $\mathcal{L}u = \rho$ in steady state, a solution to elliptic equation
- FTCS $u_{j,l}^{n+1} =$
  $u_{j,l}^{n} + \frac{\Delta t}{\Delta^2}(u_{j+1,l}^{n} + u_{j-1,l}^{n} + u_{j,l+l}^{n} + u_{j,l-1}^{n} - 4u_{j,l}^{n}) - \rho_{j,l}\Delta t$
- FTCS stability condition in 2-D: $\Delta t \leq \Delta^2/4$; choosing largest stable step in above,
- $u_{j,l}^{n+1} = \frac{1}{4}(u_{j+1,l}^{n} + u_{j-1,l}^{n} + u_{j,l+l}^{n} + u_{j,l-1}^{n}) - \frac{\Delta^2}{4}\rho_{j,l}$ Jacobi!

Relaxation as a steady state

- $\frac{\partial u}{\partial t} = \mathcal{L}u - \rho$
- $\mathcal{L}u = \rho$ in steady state, a solution to elliptic equation
- FTCS $u_{j,l}^{n+1} =$
  $u_{j,l}^{n} + \frac{\Delta t}{\Delta^2}(u_{j+1,l}^{n} + u_{j-1,l}^{n} + u_{j,l+l}^{n} + u_{j,l-1}^{n} - 4u_{j,l}^{n}) - \rho_{j,l}\Delta t$
- FTCS stability condition in 2-D: $\Delta t \leq \Delta^2/4$; choosing largest stable step in above,
- $u_{j,l}^{n+1} = \frac{1}{4}(u_{j+1,l}^{n} + u_{j-1,l}^{n} + u_{j,l+l}^{n} + u_{j,l-1}^{n}) - \frac{\Delta^2}{4}\rho_{j,l}$ Jacobi!
- **Multigrid methods** are the most sophisticated schemes for BVPs, go back and forth between coarse and fine grids