An Introduction to Iterative Solvers and Preconditioning

Alison Ramage
Dept of Mathematics and Statistics
University of Strathclyde
A.Ramage@strath.ac.uk

Revision of Linear Algebra Ideas (1)

- Consider a real square $N \times N$ matrix A.

Revision of Linear Algebra Ideas (1)

- Consider a real square $N \times N$ matrix A.
- A is symmetric if $A{=}A^T$.

Revision of Linear Algebra Ideas (1)

- Consider a real square $N \times N$ matrix A.
- A is symmetric if $A{=}A^T$.
- The trace of A, $tr(A)$, is the sum of its diagonal entries.

Revision of Linear Algebra Ideas (1)

- Consider a real square $N \times N$ matrix A.
- A is symmetric if $A{=}A^T$.
- The trace of A, $tr(A)$, is the sum of its diagonal entries.
- If $A\mathbf{v} = \lambda\mathbf{v}$ for $\lambda \in \mathbb{R}$ and $\mathbf{v} \in \mathbb{R}^N$, then λ is called the eigenvalue of A with corresponding eigenvector $\mathbf{v}$.

Revision of Linear Algebra Ideas (1)

- Consider a real square $N \times N$ matrix A.
- A is symmetric if $A{=}A^T$.
- The trace of A, $tr(A)$, is the sum of its diagonal entries.
- If $A\mathbf{v} = \lambda\mathbf{v}$ for $\lambda \in \mathbb{R}$ and $\mathbf{v} \in \mathbb{R}^N$, then λ is called the eigenvalue of A with corresponding eigenvector $\mathbf{v}$.
- A is called
  - positive definite if all of its eigenvalues are positive;
  - negative definite if all of its eigenvalues are negative;
  - indefinite if it has positive and negative eigenvalues.

Revision of Linear Algebra Ideas (2)

- Common vector norms:

$$\|\mathbf{v}\|_\infty = \max_i |v_i| \qquad \text{infinity norm}$$

$$\|\mathbf{v}\|_1 = \sum_{i=1}^{N} |v_i| \qquad \text{1-norm}$$

$$\|\mathbf{v}\|_2 = \sqrt{v_1^2 + v_2^2 + \ldots + v_n^2} = \sqrt{\mathbf{v}^T\mathbf{v}} \qquad \text{2-norm}$$

Revision of Linear Algebra Ideas (2)

- Common vector norms:

$$\|\mathbf{v}\|_\infty = \max_i |v_i| \qquad \text{infinity norm}$$

$$\|\mathbf{v}\|_1 = \sum_{i=1}^{N} |v_i| \qquad \text{1-norm}$$

$$\|\mathbf{v}\|_2 = \sqrt{v_1^2 + v_2^2 + \ldots + v_n^2} = \sqrt{\mathbf{v}^T\mathbf{v}} \qquad \text{2-norm}$$

- Common matrix norms:

$$\|A\|_\infty = \max_i \sum_{j=1}^{N} |a_{ij}| \qquad \text{max abs row sum}$$

$$\|A\|_1 = \max_j \sum_{i=1}^{N} |a_{ij}| \qquad \text{max abs column sum}$$

$$\|A\|_2 = \sqrt{\lambda_{\max}(A^T A)} \qquad \text{spectral norm}$$

$$\|A\|_F = \sqrt{tr(AA^T)} \qquad \text{Frobenius norm}$$

Revision of Linear Algebra Ideas (3)

- Vectors $\mathbf{v}, \mathbf{w}$ in a vector space V over $\mathbb{R}$ satisfy

$$\mathbf{v}, \mathbf{w} \in V \Rightarrow \mathbf{v} + \mathbf{w} \in V, \qquad \mathbf{v} \in V, \alpha \in \mathbb{R} \Rightarrow \alpha \mathbf{v} \in V.$$

Revision of Linear Algebra Ideas (3)

- Vectors $\mathbf{v}, \mathbf{w}$ in a vector space V over $\mathbb{R}$ satisfy

$$\mathbf{v}, \mathbf{w} \in V \Rightarrow \mathbf{v} + \mathbf{w} \in V, \qquad \mathbf{v} \in V, \alpha \in \mathbb{R} \Rightarrow \alpha \mathbf{v} \in V.$$

- Vectors $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_m \in V$ are linearly independent if

$$\sum_{j=1}^{m} a_j \mathbf{v}_j = a_1 \mathbf{v}_1 + \ldots + a_m \mathbf{v}_m = 0 \Leftrightarrow a_1, a_2, \ldots, a_m = 0.$$

Revision of Linear Algebra Ideas (3)

- Vectors $\mathbf{v}, \mathbf{w}$ in a vector space V over $\mathbb{R}$ satisfy

$$\mathbf{v}, \mathbf{w} \in V \Rightarrow \mathbf{v} + \mathbf{w} \in V, \qquad \mathbf{v} \in V, \alpha \in \mathbb{R} \Rightarrow \alpha \mathbf{v} \in V.$$

- Vectors $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_m \in V$ are linearly independent if

$$\sum_{j=1}^{m} a_j \mathbf{v}_j = a_1 \mathbf{v}_1 + \ldots + a_m \mathbf{v}_m = 0 \Leftrightarrow a_1, a_2, \ldots, a_m = 0.$$

- A vector space V has dimension d if there exist d linearly independent basis vectors $\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_d$ such that any vector in V can be written as a linear combination of the basis vectors, i.e.,

$$\mathbf{v} = \sum_{j=1}^{d} b_j \mathbf{y}_j.$$

Motivation: PDE-based Problems

- Large-scale simulations occur in e.g. fluid/solid mechanics, structural engineering, elasticity, medical applications, meteorology ...

Motivation: PDE-based Problems

- Large-scale simulations occur in e.g. fluid/solid mechanics, structural engineering, elasticity, medical applications, meteorology ...

- They are often modelled using, e.g., finite element, finite difference or finite volume discretisations.

Motivation: PDE-based Problems

- Large-scale simulations occur in e.g. fluid/solid mechanics, structural engineering, elasticity, medical applications, meteorology ...
- They are often modelled using, e.g., finite element, finite difference or finite volume discretisations.
- These often have only local connections between unknowns on the computational grid.

Motivation: PDE-based Problems

- Large-scale simulations occur in e.g. fluid/solid mechanics, structural engineering, elasticity, medical applications, meteorology ...
- They are often modelled using, e.g., finite element, finite difference or finite volume discretisations.
- These often have only local connections between unknowns on the computational grid.
- This can lead to very large, very sparse linear systems.

Motivation: PDE-based Problems

- Large-scale simulations occur in e.g. fluid/solid mechanics, structural engineering, elasticity, medical applications, meteorology ...
- They are often modelled using, e.g., finite element, finite difference or finite volume discretisations.
- These often have only local connections between unknowns on the computational grid.
- This can lead to very large, very sparse linear systems.
- Linear algebra costs often dominate.

Motivation: PDE-based Problems

- Large-scale simulations occur in e.g. fluid/solid mechanics, structural engineering, elasticity, medical applications, meteorology ...
- They are often modelled using, e.g., finite element, finite difference or finite volume discretisations.
- These often have only local connections between unknowns on the computational grid.
- This can lead to very large, very sparse linear systems.
- Linear algebra costs often dominate.
- Issues magnified for modern higher-dimensional PDE problems.

Model Finite Element BVP

- Laplace's equation
- d-dimensional uniform grid, discretisation parameter h
- $n = \frac{1}{h}$ nodes in each dimension
- coefficient matrix A is $N \times N$ where $N = O(n^d) = O(h^{-d})$
- e.g. bilinear finite elements
  - $d = 2$: 9 nonzeros per row
  - $d = 3$: 27 nonzeros per row

Solving Linear Systems

PROBLEM: solve $A\mathbf{x} = \mathbf{b}$ where A is very large and sparse.

Solving Linear Systems

PROBLEM: solve $A\mathbf{x} = \mathbf{b}$ where A is very large and sparse.

- Direct methods: e.g. Gaussian elimination
  - Factorise A as L (lower triangular matrix) and U (upper triangular matrix).
  - Solve for $\mathbf{x}$ via back-substitution.
  - Direct computation of exact solution $\hat{\mathbf{x}}$.

Solving Linear Systems

PROBLEM: solve $A\mathbf{x} = \mathbf{b}$ where A is very large and sparse.

- Direct methods: e.g. Gaussian elimination
  - Factorise A as L (lower triangular matrix) and U (upper triangular matrix).
  - Solve for $\mathbf{x}$ via back-substitution.
  - Direct computation of exact solution $\hat{\mathbf{x}}$.
- Iterative methods:
  - Choose an initial guess $\mathbf{x}_0$.
  - Generate a sequence of iterates $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \ldots$
  - Stop when converged in some sense to $\hat{\mathbf{x}}$.
  - Usual to iterate until $\|\mathbf{x}_k - \mathbf{x}_{k-1}\| \leq \epsilon$ for some given tolerance ϵ.

Direct or Iterative Solvers?

Direct or Iterative Solvers?

- Direct methods:
  - efficient for full matrices;
  - good for lots of RHS vectors.

Direct or Iterative Solvers?

- Direct methods:
  - efficient for full matrices;
  - good for lots of RHS vectors.

  - sparse matrices may lead to fill-in;
  - node ordering often important;
  - storage and CPU restrictions.

Direct or Iterative Solvers?

- Direct methods:
  - efficient for full matrices;
  - good for lots of RHS vectors.

  - sparse matrices may lead to fill-in;
  - node ordering often important;
  - storage and CPU restrictions.
- Iterative methods:

Direct or Iterative Solvers?

- Direct methods:
  - efficient for full matrices;
  - good for lots of RHS vectors.

  - sparse matrices may lead to fill-in;
  - node ordering often important;
  - storage and CPU restrictions.
- Iterative methods:
  - data structures predetermined;
  - no need for special node ordering;
  - efficient for extremely large sparse problems;
  - last iterate can give a good starting vector.

Direct or Iterative Solvers?

- Direct methods:
  - efficient for full matrices;
  - good for lots of RHS vectors.

  - sparse matrices may lead to fill-in;
  - node ordering often important;
  - storage and CPU restrictions.

- Iterative methods:
  - data structures predetermined;
  - no need for special node ordering;
  - efficient for extremely large sparse problems;
  - last iterate can give a good starting vector.

  - some expertise needed;
  - difficult to judge when to stop;
  - lack of robustness.

Asymptotic Work Estimates

Iterative method: Conjugate Gradient Method

Direct method: Gaussian Elimination with band-minimising node ordering

Asymptotic Work Estimates

Iterative method: Conjugate Gradient Method

Direct method: Gaussian Elimination with band-minimising node ordering

Computational Work

	$d = 2$	$d = 3$
CG	$O(N^{\frac{3}{2}})$	$O(N^{\frac{4}{3}})$
GE factorise	$O(N^2)$	$O(N^{\frac{7}{3}})$
GE solve	$O(N^{\frac{3}{2}})$	$O(N^{\frac{5}{3}})$

Asymptotic Work Estimates

Iterative method: Conjugate Gradient Method

Direct method: Gaussian Elimination with band-minimising node ordering

Computational Work

	$d = 2$	$d = 3$
CG	$O(N^{\frac{3}{2}})$	$O(N^{\frac{4}{3}})$
GE factorise	$O(N^2)$	$O(N^{\frac{7}{3}})$
GE solve	$O(N^{\frac{3}{2}})$	$O(N^{\frac{5}{3}})$

Storage

	$d = 2$	$d = 3$
CG	$O(N)$	$O(N)$
GE factorise	$O(N^{\frac{3}{2}})$	$O(N^{\frac{5}{3}})$
GE solve	$O(N^{\frac{3}{2}})$	$O(N^{\frac{5}{3}})$

Stationary Iterative Methods

- Matrix splitting $A = M - N$ (M invertible):

$$A\mathbf{x} = \mathbf{b} \quad \Rightarrow \quad (M - N)\mathbf{x} = \mathbf{b} \Rightarrow \quad M\mathbf{x} = N\mathbf{x} + \mathbf{b}.$$

Stationary Iterative Methods

- Matrix splitting $A = M - N$ (M invertible):

$$A\mathbf{x} = \mathbf{b} \quad \Rightarrow \quad (M - N)\mathbf{x} = \mathbf{b} \Rightarrow \quad M\mathbf{x} = N\mathbf{x} + \mathbf{b}.$$

- Generate iterates via

$$M\mathbf{x}_k = N\mathbf{x}_{k-1} + \mathbf{b} \Rightarrow \mathbf{x}_k = M^{-1}N\mathbf{x}_{k-1} + M^{-1}\mathbf{b}.$$

$$\text{iteration matrix} \quad R = M^{-1}N$$

Stationary Iterative Methods

- Matrix splitting $A = M - N$ (M invertible):

$$A\mathbf{x} = \mathbf{b} \quad \Rightarrow \quad (M - N)\mathbf{x} = \mathbf{b} \Rightarrow \quad M\mathbf{x} = N\mathbf{x} + \mathbf{b}.$$

- Generate iterates via

$$M\mathbf{x}_k = N\mathbf{x}_{k-1} + \mathbf{b} \Rightarrow \mathbf{x}_k = M^{-1}N\mathbf{x}_{k-1} + M^{-1}\mathbf{b}.$$

$$\text{iteration matrix} \quad R = M^{-1}N$$

- Typical splittings combine D, L, and U where

$$A = D + L + U.$$

Stationary Iterative Methods

- Matrix splitting $A = M - N$ (M invertible):

$$A\mathbf{x} = \mathbf{b} \quad \Rightarrow \quad (M - N)\mathbf{x} = \mathbf{b} \Rightarrow \quad M\mathbf{x} = N\mathbf{x} + \mathbf{b}.$$

- Generate iterates via

$$M\mathbf{x}_k = N\mathbf{x}_{k-1} + \mathbf{b} \Rightarrow \mathbf{x}_k = M^{-1}N\mathbf{x}_{k-1} + M^{-1}\mathbf{b}.$$

$$\text{iteration matrix} \quad R = M^{-1}N$$

- Typical splittings combine D, L, and U where

$$A = D + L + U.$$

- For stationary methods, M and N do not depend on k.

Common Matrix Splittings

$$A = M - N, \qquad A = D + L + U$$

$$\begin{aligned}
\text{Richardson: } A &= I - (I - A) \\
\text{Jacobi: } A &= D - [-(L + U)] \\
\text{Gauss-Seidel: } A &= (D + L) - (-U) \\
\text{SOR: } A &= M_\omega - N_\omega \\
&= \frac{1}{\omega}(D + \omega L) - \frac{1}{\omega}[(1 - \omega)D - \omega U] \\
\text{SSOR: } A &= \frac{\omega}{2 - \omega}(M_\omega D^{-1} M_\omega^T - N_\omega D^{-1} N_\omega^T)
\end{aligned}$$

Nonstationary Methods

- Solve $A\mathbf{x} = \mathbf{b}$ where
  - A is symmetric and positive definite;
  - A has s distinct (positive) eigenvalues.

Nonstationary Methods

- Solve $A\mathbf{x} = \mathbf{b}$ where
  - A is symmetric and positive definite;
  - A has s distinct (positive) eigenvalues.

- Minimal polynomial is

$$A^s + m_1 A^{s-1} + \cdots + m_{s-1} A + m_s I = 0$$

so

$$A^{-1} = -\frac{1}{m_s} A^{s-1} - \frac{m_1}{m_s} A^{s-2} - \cdots - \frac{m_{s-1}}{m_s} I.$$

Nonstationary Methods

- Solve $A\mathbf{x} = \mathbf{b}$ where
  - A is symmetric and positive definite;
  - A has s distinct (positive) eigenvalues.

- Minimal polynomial is

$$A^s + m_1 A^{s-1} + \cdots + m_{s-1} A + m_s I = 0$$

so

$$A^{-1} = -\frac{1}{m_s} A^{s-1} - \frac{m_1}{m_s} A^{s-2} - \cdots - \frac{m_{s-1}}{m_s} I.$$

- This means that $\hat{\mathbf{x}} = A^{-1}\mathbf{b} \in \mathcal{K}(A, \mathbf{b}, s)$ where

$$\mathcal{K}(A, \mathbf{b}, s) \equiv \text{span}\{\mathbf{b}, A\mathbf{b}, A^2\mathbf{b}, \ldots, A^{s-1}\mathbf{b}\}$$

is called a Krylov subspace.

Three equivalent problems

1. Solve the linear equations $A\mathbf{x} = \mathbf{b}$.

Three equivalent problems

1. Solve the linear equations $A\mathbf{x} = \mathbf{b}$.

2. Minimise the quadratic functional $\Phi(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T A\mathbf{x} - \mathbf{x}^T\mathbf{b}$.

$$\nabla\Phi(\mathbf{x}) = A\mathbf{x} - \mathbf{b} = \mathbf{0} \quad \Leftrightarrow \quad A\mathbf{x} = \mathbf{b}.$$

Three equivalent problems

1. Solve the linear equations $A\mathbf{x} = \mathbf{b}$.

2. Minimise the quadratic functional $\Phi(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T A\mathbf{x} - \mathbf{x}^T\mathbf{b}$.

$$\nabla\Phi(\mathbf{x}) = A\mathbf{x} - \mathbf{b} = \mathbf{0} \quad \Leftrightarrow \quad A\mathbf{x} = \mathbf{b}.$$

3. Minimise the norm $\|\mathbf{x} - \hat{\mathbf{x}}\|_A$ where $\|\mathbf{v}\|_A = \left\{\mathbf{v}^T A\mathbf{v}\right\}^{\frac{1}{2}}$.

$$\|\mathbf{x} - \hat{\mathbf{x}}\|_A^2 = (\mathbf{x} - \hat{\mathbf{x}})^T A(\mathbf{x} - \hat{\mathbf{x}}) = \mathbf{b}^T A^{-1}\mathbf{b} + 2\Phi(\mathbf{x}).$$

Steepest Descent Method

$$\Phi(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T A\mathbf{x} - \mathbf{x}^T\mathbf{b}$$

Steepest Descent Method

$$\Phi(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T A\mathbf{x} - \mathbf{x}^T\mathbf{b}$$

- At a point $\mathbf{x}_k$, Φ decreases most rapidly in the direction

$$-\nabla\Phi(\mathbf{x}_k) = \mathbf{b} - A\mathbf{x}_k = \mathbf{r}_k.$$

Steepest Descent Method

$$\Phi(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T A\mathbf{x} - \mathbf{x}^T\mathbf{b}$$

- At a point $\mathbf{x}_k$, Φ decreases most rapidly in the direction

$$-\nabla\Phi(\mathbf{x}_k) = \mathbf{b} - A\mathbf{x}_k = \mathbf{r}_k.$$

- The value of Φ at point $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k\mathbf{r}_k$ minimised when

$$\alpha_k = \frac{\mathbf{r}_k^T\mathbf{r}_k}{\mathbf{r}_k^T A\mathbf{r}_k}.$$

Steepest Descent Method

$$\Phi(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T A\mathbf{x} - \mathbf{x}^T\mathbf{b}$$

- At a point $\mathbf{x}_k$, Φ decreases most rapidly in the direction

$$-\nabla\Phi(\mathbf{x}_k) = \mathbf{b} - A\mathbf{x}_k = \mathbf{r}_k.$$

- The value of Φ at point $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k\mathbf{r}_k$ minimised when

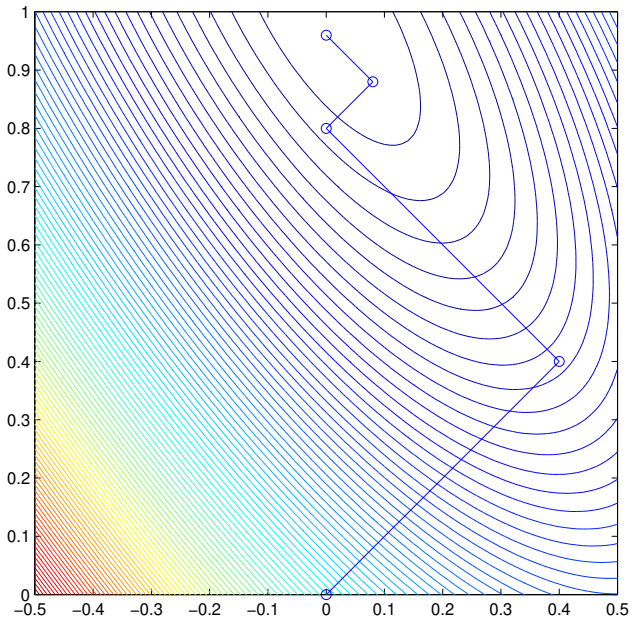
$$\alpha_k = \frac{\mathbf{r}_k^T\mathbf{r}_k}{\mathbf{r}_k^T A\mathbf{r}_k}.$$

- This choice enforces

$$\Phi(\mathbf{x}_{k+1}) < \Phi(\mathbf{x}_k).$$

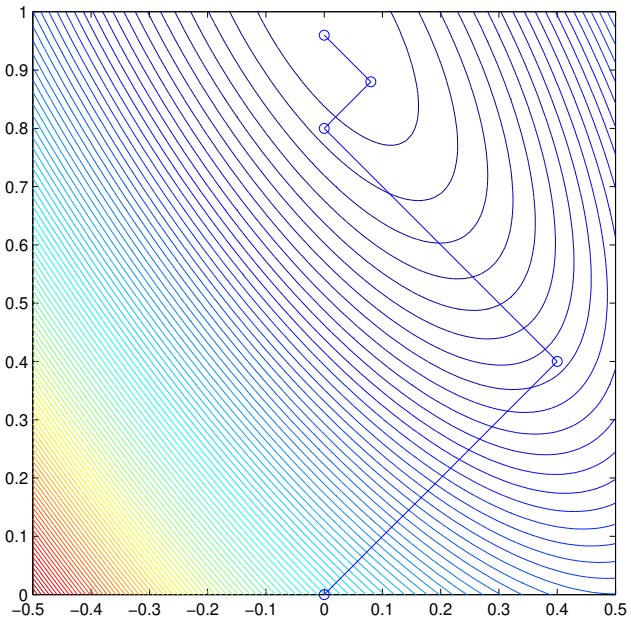
A Practical SD Example

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

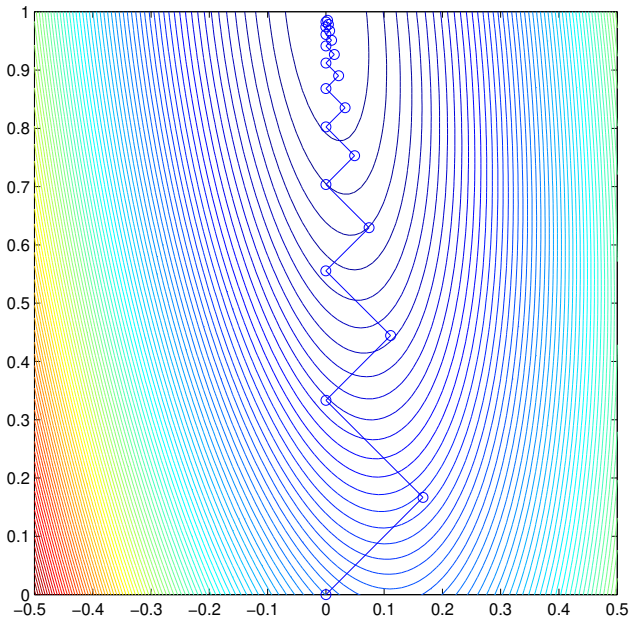


A Practical SD Example

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



$$A = \begin{bmatrix} 9 & 1 \\ 1 & 1 \end{bmatrix}$$



Conjugate Gradient Method

- Choose new search directions $\{\mathbf{p}_0, \mathbf{p}_1, \ldots\}$ and write

$$P_{k+1} \equiv \mathrm{span}\{\mathbf{p}_0, \ldots, \mathbf{p}_k\}.$$

Conjugate Gradient Method

- Choose new search directions $\{\mathbf{p}_0, \mathbf{p}_1, \ldots\}$ and write
$$P_{k+1} \equiv \mathrm{span}\{\mathbf{p}_0, \ldots, \mathbf{p}_k\}.$$
- Construct iterates
$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha \mathbf{p}_k.$$

Conjugate Gradient Method

- Choose new search directions $\{\mathbf{p}_0, \mathbf{p}_1, \ldots\}$ and write
$$P_{k+1} \equiv \text{span}\{\mathbf{p}_0, \ldots, \mathbf{p}_k\}.$$
- Construct iterates
$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha \mathbf{p}_k.$$
- We would like

$$\text{(i)} \qquad \mathbf{x}_{k+1} = \min_{\mathbf{x} \in P_{k+1}} \Phi(\mathbf{x});$$

$$\text{(ii)} \qquad \min_{\alpha} \Phi(\mathbf{x}_k + \alpha \mathbf{p}_k).$$

Conjugate Gradient Method

- Choose new search directions $\{\mathbf{p}_0, \mathbf{p}_1, \ldots\}$ and write
$$P_{k+1} \equiv \text{span}\{\mathbf{p}_0, \ldots, \mathbf{p}_k\}.$$
- Construct iterates
$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha \mathbf{p}_k.$$
- We would like
$$\text{(i)} \qquad \mathbf{x}_{k+1} = \min_{\mathbf{x} \in P_{k+1}} \Phi(\mathbf{x});$$
$$\text{(ii)} \qquad \min_{\alpha} \Phi(\mathbf{x}_k + \alpha \mathbf{p}_k).$$
- QUESTION: can we choose $\mathbf{p}_k$ so that $\mathbf{x}_{k+1}$ satisfies (i) and (ii) simultaneously?

Conjugate Gradient Method

- Choose new search directions $\{\mathbf{p}_0, \mathbf{p}_1, \ldots\}$ and write
$$P_{k+1} \equiv \text{span}\{\mathbf{p}_0, \ldots, \mathbf{p}_k\}.$$
- Construct iterates
$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha \mathbf{p}_k.$$
- We would like
$$\text{(i)} \quad \mathbf{x}_{k+1} = \min_{\mathbf{x} \in P_{k+1}} \Phi(\mathbf{x});$$
$$\text{(ii)} \quad \min_{\alpha} \Phi(\mathbf{x}_k + \alpha \mathbf{p}_k).$$
- QUESTION: can we choose $\mathbf{p}_k$ so that $\mathbf{x}_{k+1}$ satisfies (i) and (ii) simultaneously?
- ANSWER: Yes! Choose the vectors $\mathbf{p}_k$ to be A-conjugate, i.e.
$$\mathbf{p}_j^T A \mathbf{p}_k = 0, \quad j < k.$$

Conjugate Gradient Method

Hestenes & Stiefel (1952)

choose $\mathbf{x}_0$
compute $\mathbf{r}_0 = \mathbf{b} - A\mathbf{x}_0$
set $\mathbf{p}_0 = \mathbf{r}_0$
for $k = 0$ until convergence do
$\alpha_k = \mathbf{r}_k^T \mathbf{r}_k / \mathbf{p}_k^T A\mathbf{p}_k$
$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k$
$\mathbf{r}_{k+1} = \mathbf{r}_k - \alpha_k A\mathbf{p}_k$
$\beta_k = \mathbf{r}_{k+1}^T \mathbf{r}_{k+1} / \mathbf{r}_k^T \mathbf{r}_k$
$\mathbf{p}_{k+1} = \mathbf{r}_{k+1} + \beta_k \mathbf{p}_k$
end do

can be implemented with one MVM per iteration

Finite Termination

- The CG method constructs iterates

$$\mathbf{x}_k \in \mathbf{x}_0 + \text{span}\{\mathbf{r}_0, A\mathbf{r}_0, \ldots, A^{k-1}\mathbf{r}_0\}$$

  with the properties

  - $\mathbf{x}_k$ minimises $\|\mathbf{x}_k - \hat{\mathbf{x}}\|_A$;
  - iterates can be generated by a three-term recurrence relation.

Finite Termination

- The CG method constructs iterates

$$\mathbf{x}_k \in \mathbf{x}_0 + \text{span}\{\mathbf{r}_0, A\mathbf{r}_0, \ldots, A^{k-1}\mathbf{r}_0\}$$

  with the properties

  - $\mathbf{x}_k$ minimises $\|\mathbf{x}_k - \hat{\mathbf{x}}\|_A$;
  - iterates can be generated by a three-term recurrence relation.

- Theorem: The CG method finds $\hat{\mathbf{x}}$ in s steps.

Finite Termination

- The CG method constructs iterates

$$\mathbf{x}_k \in \mathbf{x}_0 + \text{span}\{\mathbf{r}_0, A\mathbf{r}_0, \ldots, A^{k-1}\mathbf{r}_0\}$$

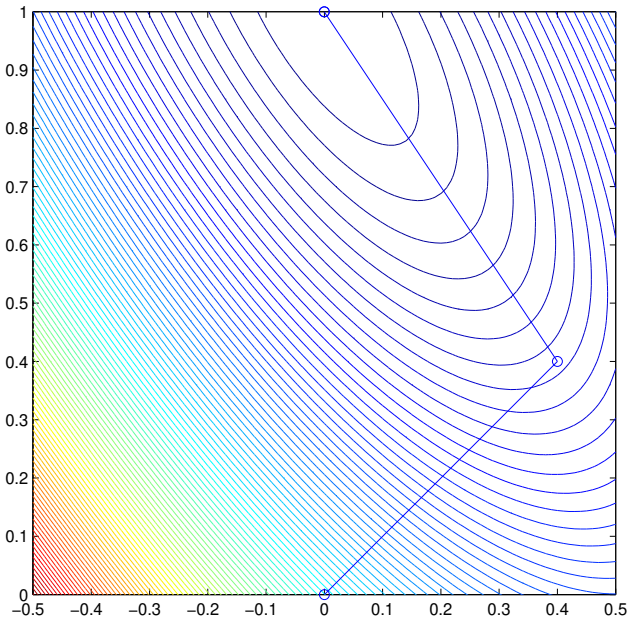
  with the properties

  - $\mathbf{x}_k$ minimises $\|\mathbf{x}_k - \hat{\mathbf{x}}\|_A$;
  - iterates can be generated by a three-term recurrence relation.

- Theorem: The CG method finds $\hat{\mathbf{x}}$ in s steps.
- In exact arithmetic, CG is a direct method!

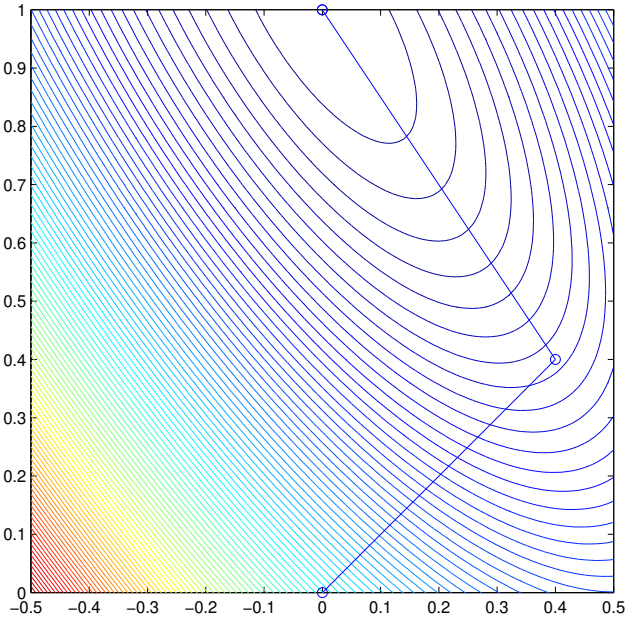
A Practical CG Example

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

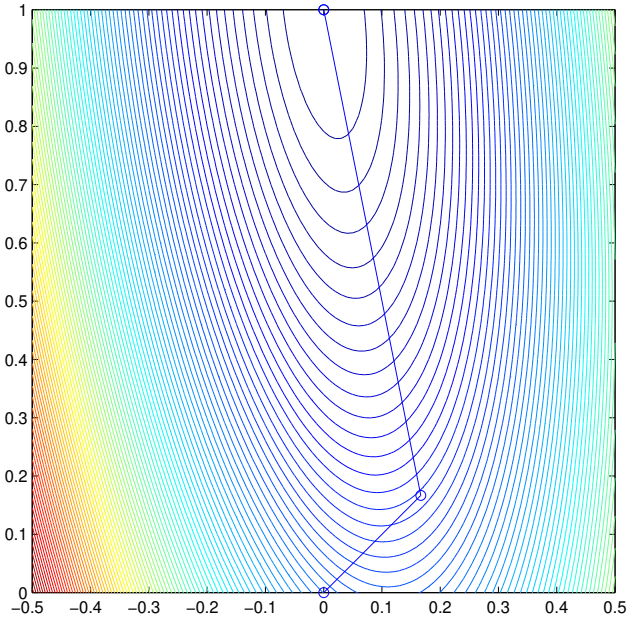


A Practical CG Example

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



$$A = \begin{bmatrix} 9 & 1 \\ 1 & 1 \end{bmatrix}$$



CG Convergence

$$\mathbf{x}_k \in \mathbf{x}_0 + \text{span}\{\mathbf{r}_0, A\mathbf{r}_0, \ldots, A^{k-1}\mathbf{r}_0\}$$

CG Convergence

$$\mathbf{x}_k \in \mathbf{x}_0 + \text{span}\{\mathbf{r}_0, A\mathbf{r}_0, \ldots, A^{k-1}\mathbf{r}_0\}$$

- Each residual can be written as a polynomial in A times $\mathbf{r}_0$:

$$\mathbf{r}_k = \mathbf{b} - A\mathbf{x}_k = \mathbf{b} - A\left(\mathbf{x}_0 + \sum_{i=1}^{k} \gamma_i A^{i-1}\mathbf{r}_0\right) = \mathbf{r}_0 - \sum_{i=1}^{k} \gamma_i A^i \mathbf{r}_0$$

$$\mathbf{r}_k = \hat{P}_k(A)\mathbf{r}_0$$

$\hat{P}_k \in \Pi_k^1 \equiv$ polynomials of degree k with constant term 1

CG Convergence

$$\mathbf{x}_k \in \mathbf{x}_0 + \text{span}\{\mathbf{r}_0, A\mathbf{r}_0, \ldots, A^{k-1}\mathbf{r}_0\}$$

- Each residual can be written as a polynomial in A times $\mathbf{r}_0$:

$$\mathbf{r}_k = \mathbf{b} - A\mathbf{x}_k = \mathbf{b} - A\left(\mathbf{x}_0 + \sum_{i=1}^{k} \gamma_i A^{i-1}\mathbf{r}_0\right) = \mathbf{r}_0 - \sum_{i=1}^{k} \gamma_i A^i \mathbf{r}_0$$

$$\mathbf{r}_k = \hat{P}_k(A)\mathbf{r}_0$$

$\hat{P}_k \in \Pi_k^1 \equiv$ polynomials of degree k with constant term 1

- This gives

$$\|\mathbf{x}_k - \hat{\mathbf{x}}\|_A = \|\mathbf{r}_k\|_{A^{-1}} = \min_{P_k \in \Pi_k^1} \|P_k(A)\mathbf{r}_0\|_{A^{-1}}.$$

- Now expand $\mathbf{r}_0$ in terms of orthonormal eigenvectors:

$$\mathbf{r}_0 = \sum_{i=1}^{n} \rho_i \mathbf{v}_i, \qquad \rho_i = \mathbf{v}_i^T \mathbf{r}_0, \qquad A\mathbf{v}_i = \lambda_i \mathbf{v}_i.$$

- Now expand $\mathbf{r}_0$ in terms of orthonormal eigenvectors:

$$\mathbf{r}_0 = \sum_{i=1}^{n} \rho_i \mathbf{v}_i, \qquad \rho_i = \mathbf{v}_i^T \mathbf{r}_0, \qquad A\mathbf{v}_i = \lambda_i \mathbf{v}_i.$$

- This gives

$$\begin{aligned}
\|\mathbf{x}_k - \hat{\mathbf{x}}\|_A &= \min_{P_k \in \Pi_k^1} \|P_k(A) \sum_{i=1}^{n} \rho_i \mathbf{v}_i\|_{A^{-1}} \\
&= \left\{ \min_{P_k \in \Pi_k^1} \sum_{i=1}^{n} P_k(\lambda_i)^2 (\rho_i \mathbf{v}_i)^T A^{-1} (\rho_i \mathbf{v}_i) \right\}^{\frac{1}{2}} \\
&\leq \min_{P_k \in \Pi_k^1} \max_i |P_k(\lambda_i)| \left\{ \mathbf{r}_0^T A^{-1} \mathbf{r}_0 \right\}^{\frac{1}{2}} \\
&= \min_{P_k \in \Pi_k^1} \max_i |P_k(\lambda_i)| \|\mathbf{r}_0\|_{A^{-1}}
\end{aligned}$$

so $$\|\mathbf{x}_k - \hat{\mathbf{x}}\|_A \leq \min_{P_k \in \Pi_k^1} \max_i |P_k(\lambda_i)| \|\mathbf{x}_0 - \hat{\mathbf{x}}\|_A.$$

Ideal Bound

- Supppose the polynomial $P_{\min}$ is such that

$$M = \max_i |P_{\min}(\lambda_i)| = \min_{P_k \in \Pi_k^1} \max_i |P_k(\lambda_i)|.$$

MINIMAX APPROXIMATION

Ideal Bound

- Supppose the polynomial $P_{\min}$ is such that

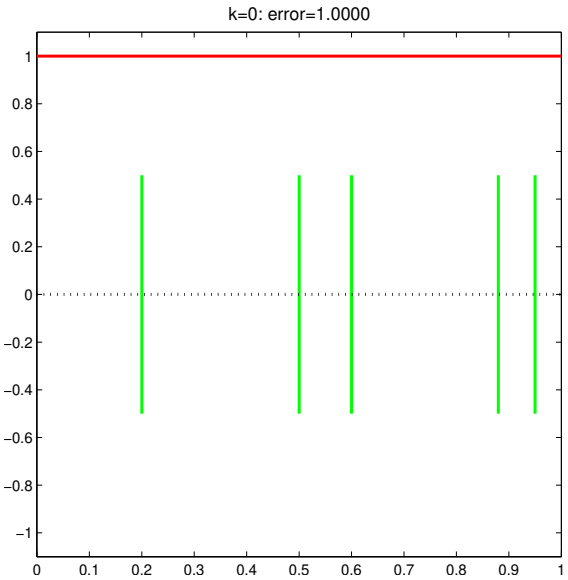
$$M = \max_i |P_{\min}(\lambda_i)| = \min_{P_k \in \Pi_k^1} \max_i |P_k(\lambda_i)|.$$

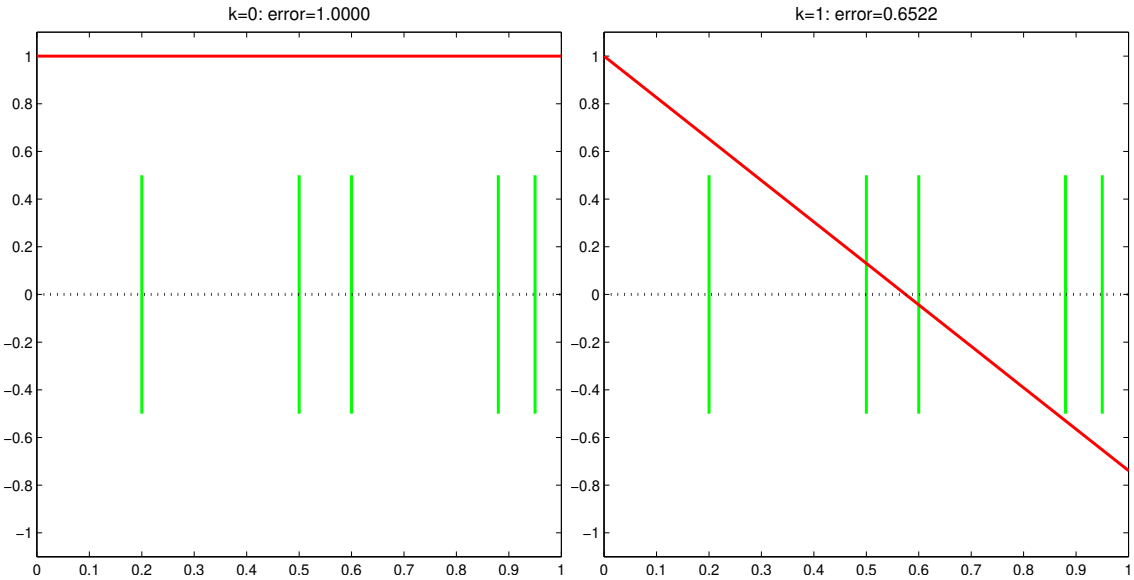
MINIMAX APPROXIMATION

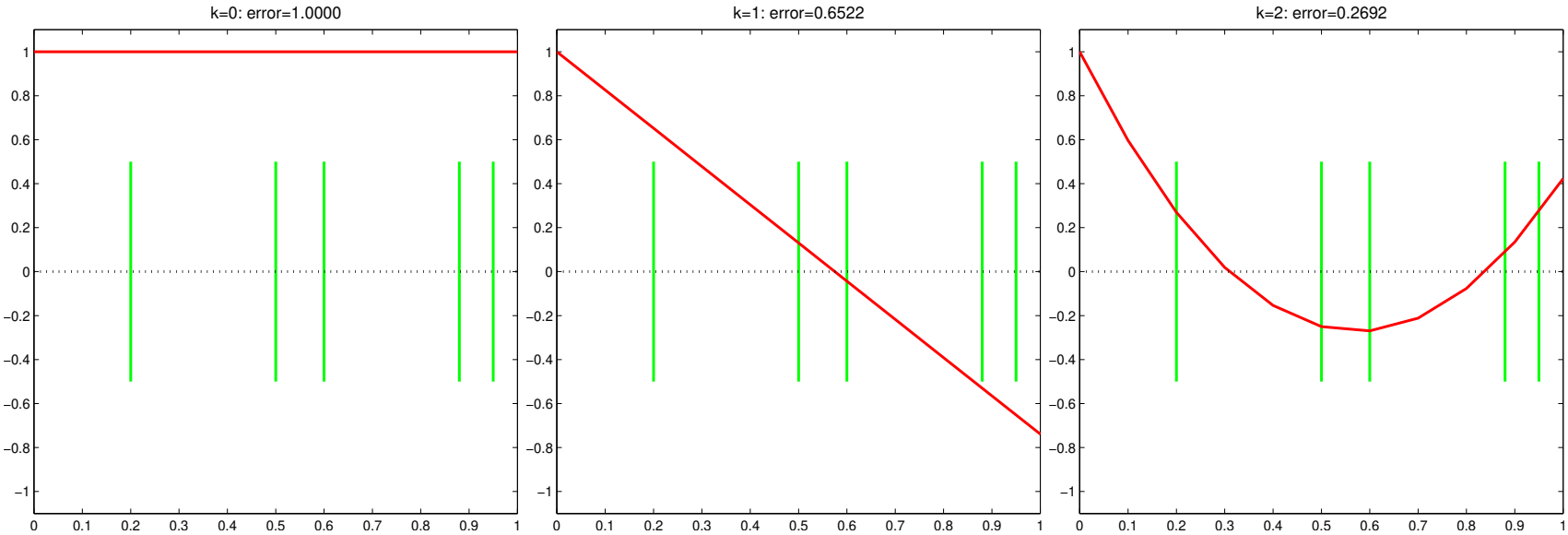
- Theorem: Greenbaum (1979)
  This error bound is sharp, i.e. there is always some $\mathbf{x}_0$ such that the discrete minimax bound

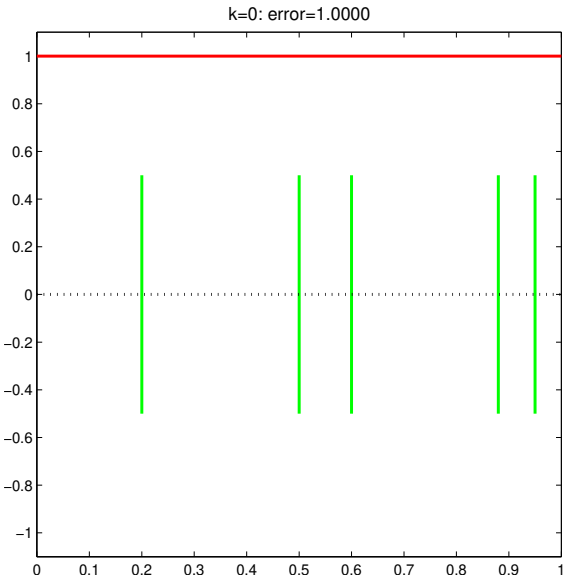
$$\|\mathbf{x}_k - \hat{\mathbf{x}}\|_A \leq M \|\mathbf{x}_0 - \hat{\mathbf{x}}\|_A$$

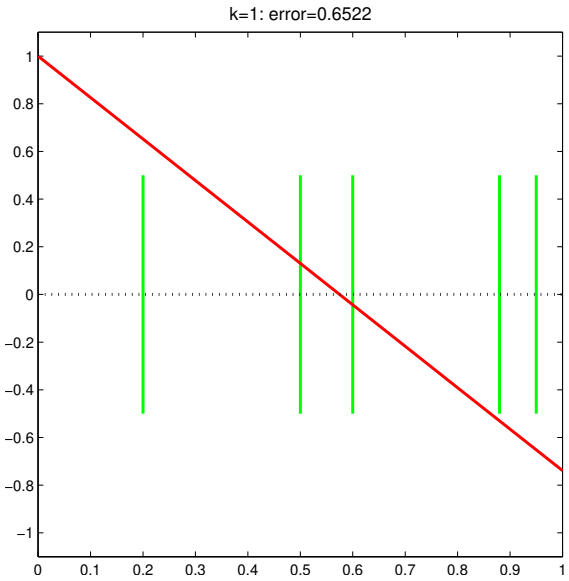
  is attained.

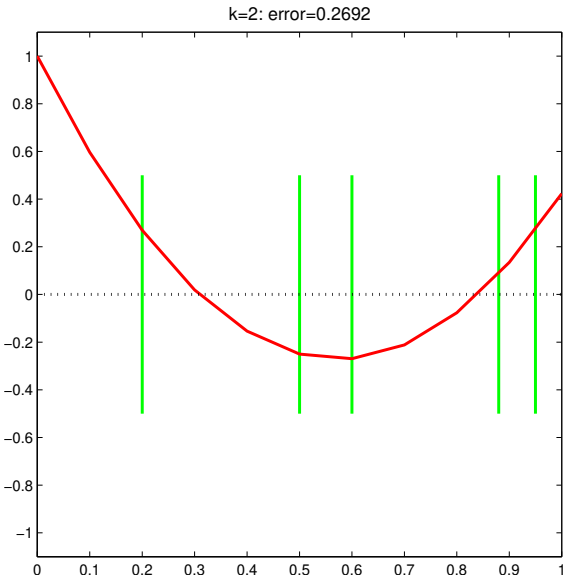

k=0: error=1.0000
1
0.8
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
−1
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

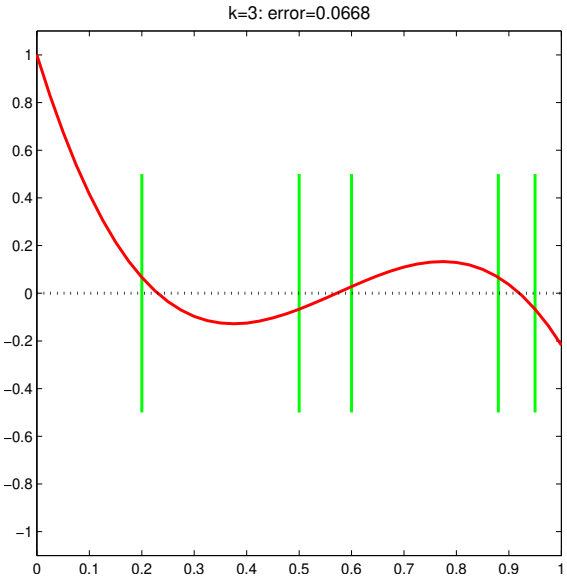

k=0: error=1.0000
k=1: error=0.6522
1
0.8
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
−1
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

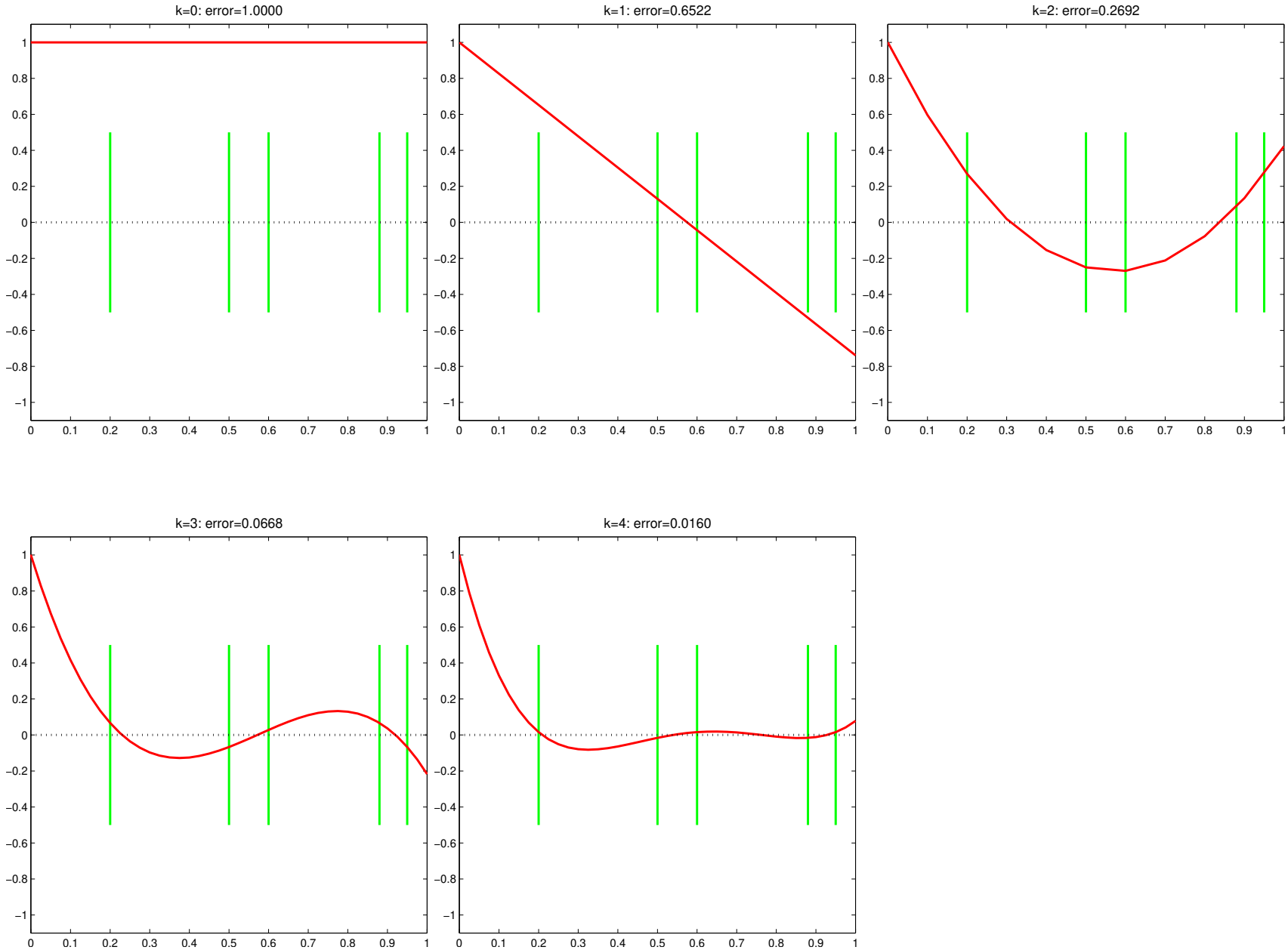

k=0: error=1.0000
k=1: error=0.6522
k=2: error=0.2692

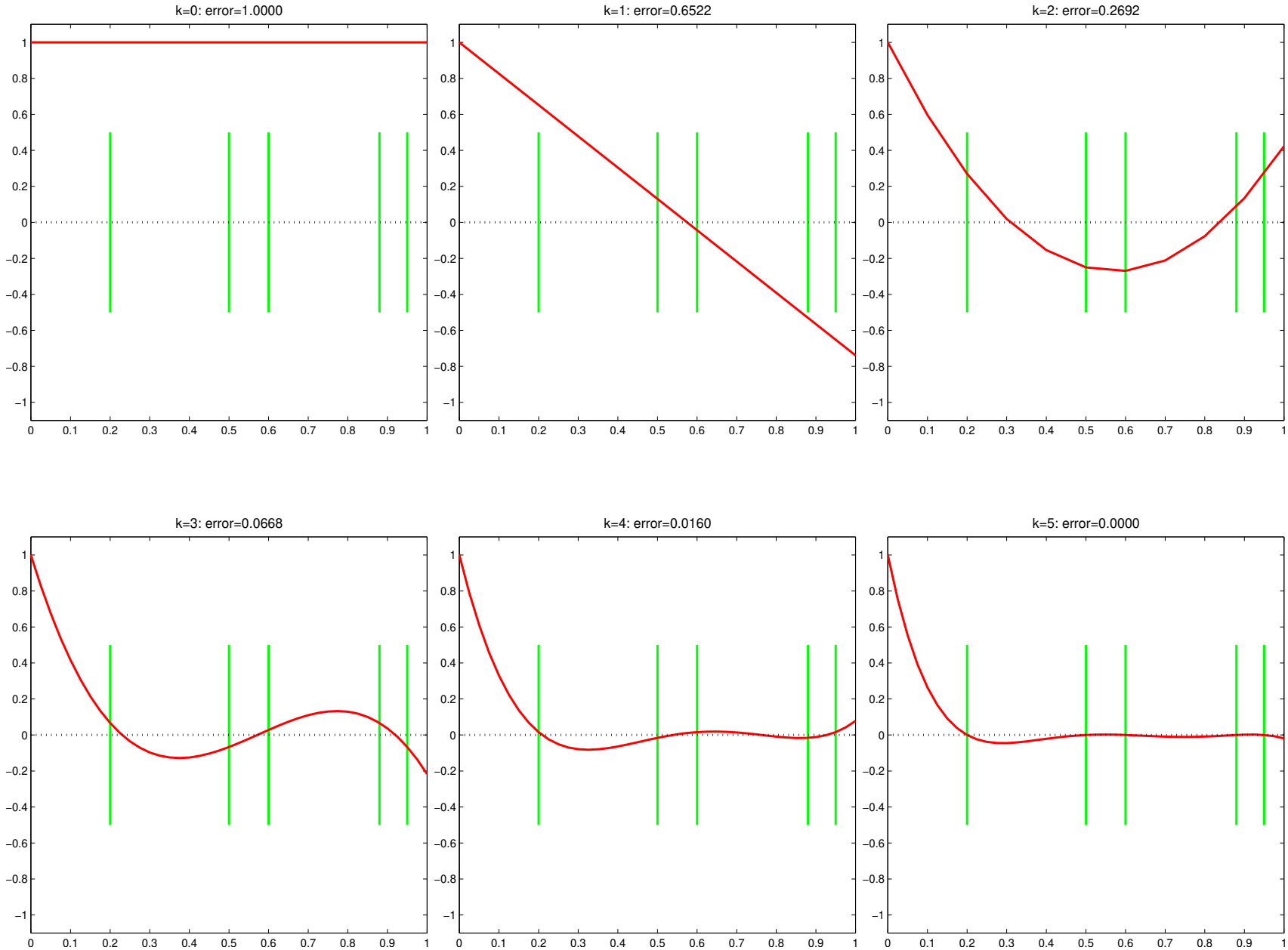

k=0: error=1.0000


k=1: error=0.6522


k=2: error=0.2692


k=3: error=0.0668


k=0: error=1.0000
k=1: error=0.6522
k=2: error=0.2692
k=3: error=0.0668
k=4: error=0.0160


k=0: error=1.0000
k=1: error=0.6522
k=2: error=0.2692
k=3: error=0.0668
k=4: error=0.0160
k=5: error=0.0000

Practical Bound

- Based on knowledge of $\lambda_{\max}$ and $\lambda_{\min}$ alone, bound involves

$$\hat{T}_k(\lambda) = \frac{T_k\left[\frac{\lambda_{\max}+\lambda_{\min}-2\lambda}{\lambda_{\max}-\lambda_{\min}}\right]}{T_k\left[\frac{\lambda_{\max}+\lambda_{\min}}{\lambda_{\max}-\lambda_{\min}}\right]}, \qquad \text{condition number} \quad \kappa = \frac{\lambda_{\max}}{\lambda_{\min}}.$$

Practical Bound

- Based on knowledge of $\lambda_{\max}$ and $\lambda_{\min}$ alone, bound involves

$$\hat{T}_k(\lambda) = \frac{T_k\left[\frac{\lambda_{\max}+\lambda_{\min}-2\lambda}{\lambda_{\max}-\lambda_{\min}}\right]}{T_k\left[\frac{\lambda_{\max}+\lambda_{\min}}{\lambda_{\max}-\lambda_{\min}}\right]}, \qquad \text{condition number} \quad \kappa = \frac{\lambda_{\max}}{\lambda_{\min}}.$$

$$M = \max_i |\hat{T}_k(\lambda_i)| = \frac{1}{T_k\left[\frac{\lambda_{\max}+\lambda_{\min}}{\lambda_{\max}-\lambda_{\min}}\right]} = \frac{1}{T_k\left[\frac{\kappa+1}{\kappa-1}\right]}$$

CHEBYSHEV APPROXIMATION

Practical Bound

- Based on knowledge of $\lambda_{\max}$ and $\lambda_{\min}$ alone, bound involves

$$\hat{T}_k(\lambda) = \frac{T_k\left[\frac{\lambda_{\max}+\lambda_{\min}-2\lambda}{\lambda_{\max}-\lambda_{\min}}\right]}{T_k\left[\frac{\lambda_{\max}+\lambda_{\min}}{\lambda_{\max}-\lambda_{\min}}\right]}, \qquad \text{condition number} \quad \kappa = \frac{\lambda_{\max}}{\lambda_{\min}}.$$

$$M = \max_i |\hat{T}_k(\lambda_i)| = \frac{1}{T_k\left[\frac{\lambda_{\max}+\lambda_{\min}}{\lambda_{\max}-\lambda_{\min}}\right]} = \frac{1}{T_k\left[\frac{\kappa+1}{\kappa-1}\right]}$$

CHEBYSHEV APPROXIMATION

- Number of iterations required for CG convergence is

$$k \simeq \frac{1}{2}\ln\left(\frac{2}{\epsilon}\sqrt{\kappa}\right).$$

PDE examples

Laplace's eqn, 3D uniform grid with n nodes per dimension

$$r, s, t = 1, \ldots, n$$

7 point Finite Difference Stencil

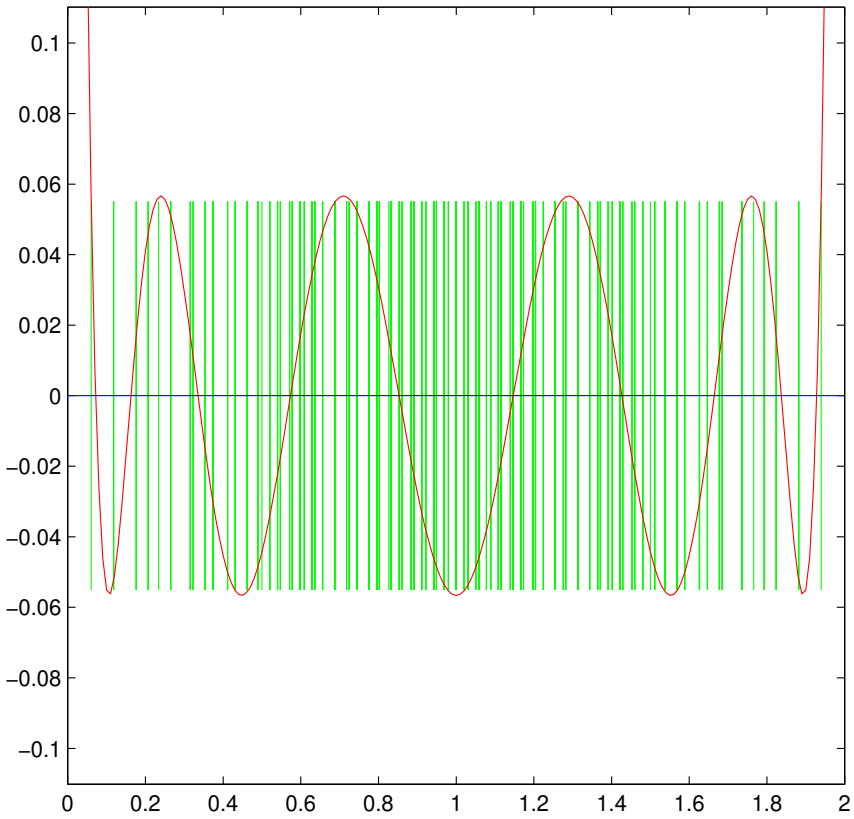
$$\lambda^{rst} = 1 - \frac{1}{3}\cos\frac{r\pi}{n+1} - \frac{1}{3}\cos\frac{s\pi}{n+1} - \frac{1}{3}\cos\frac{t\pi}{n+1}$$

27 point Finite Element Stencil

$$\lambda^{rst} = 1 - \frac{1}{4}\cos\frac{r\pi}{n+1}\cos\frac{s\pi}{n+1} - \frac{1}{4}\cos\frac{r\pi}{n+1}\cos\frac{t\pi}{n+1}$$
$$-\frac{1}{4}\cos\frac{s\pi}{n+1}\cos\frac{t\pi}{n+1} - \frac{1}{4}\cos\frac{r\pi}{n+1}\cos\frac{s\pi}{n+1}\cos\frac{t\pi}{n+1}$$

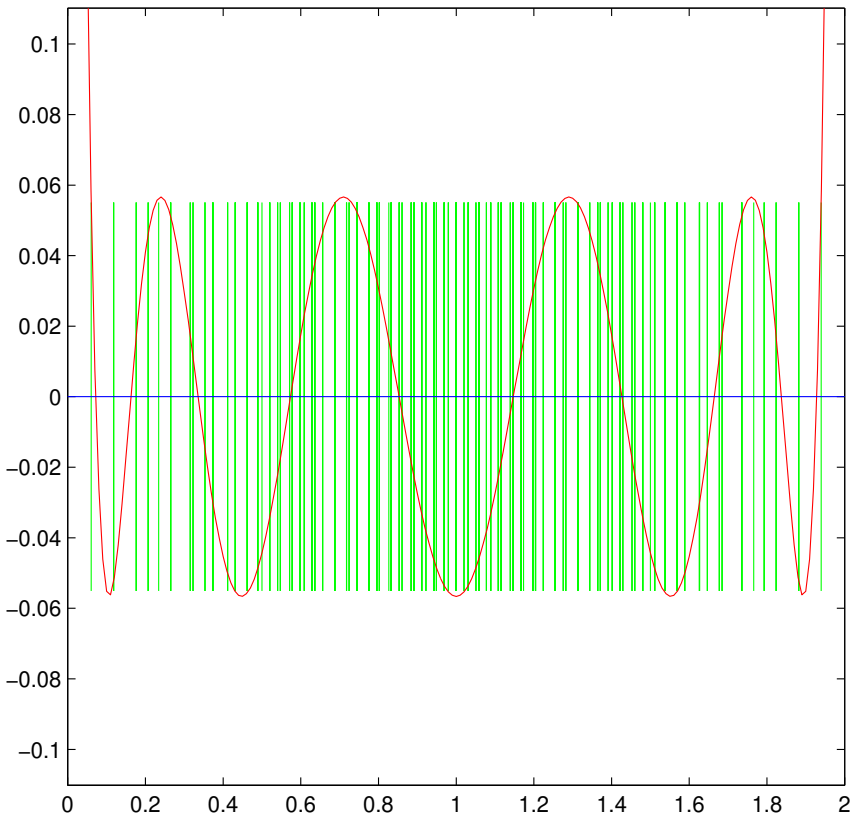
7 point Finite Difference Stencil

Tchebyshev error: 0.5662e-1

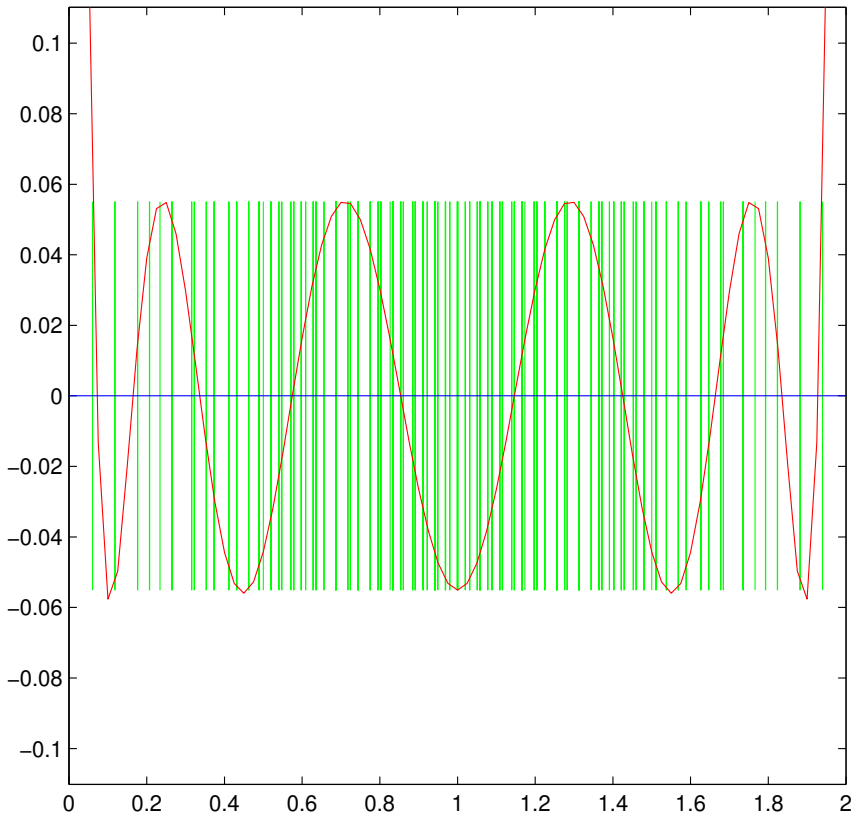


7 point Finite Difference Stencil

Tchebyshev error: 0.5662e-1

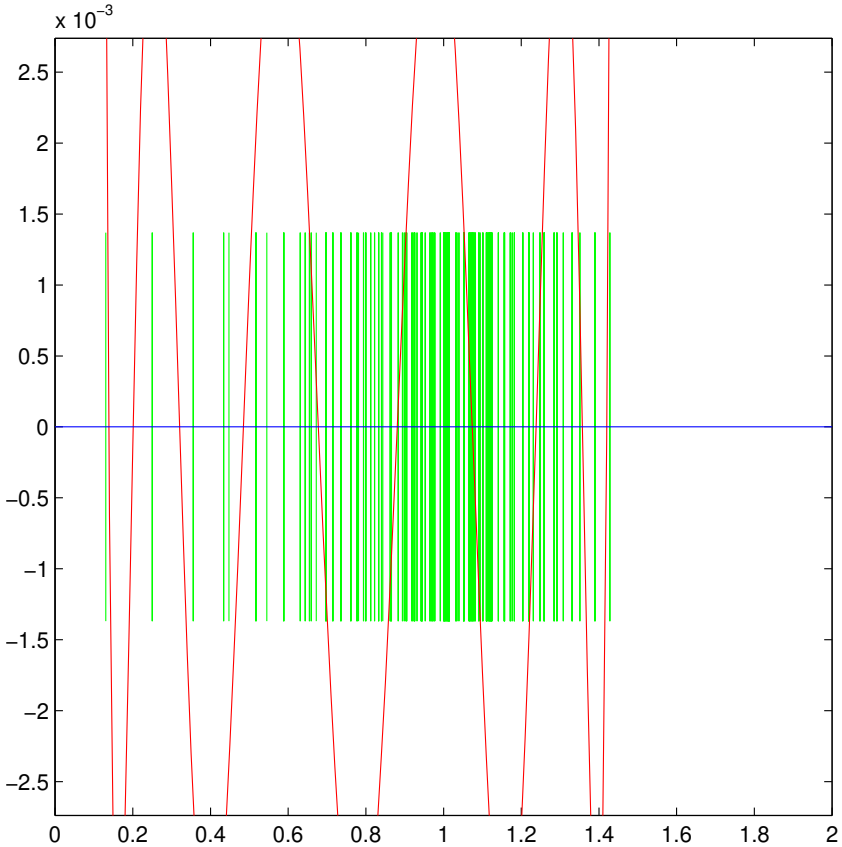


Minimax error: 0.5507e-1



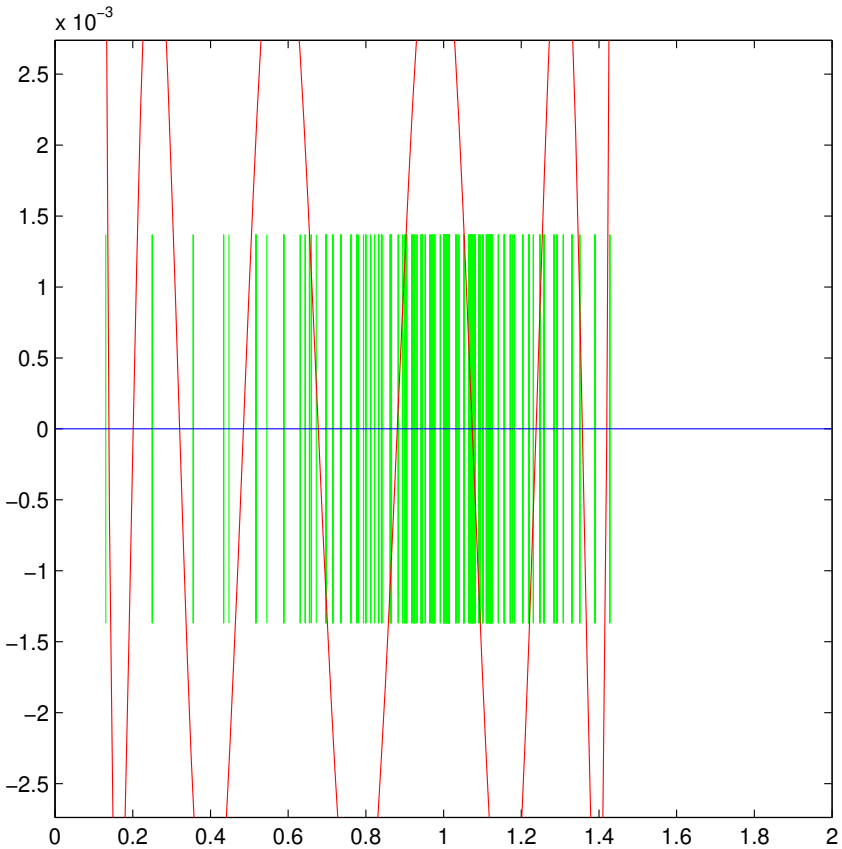
27 point Finite Element Stencil

Tchebyshev error: 0.3918e-2

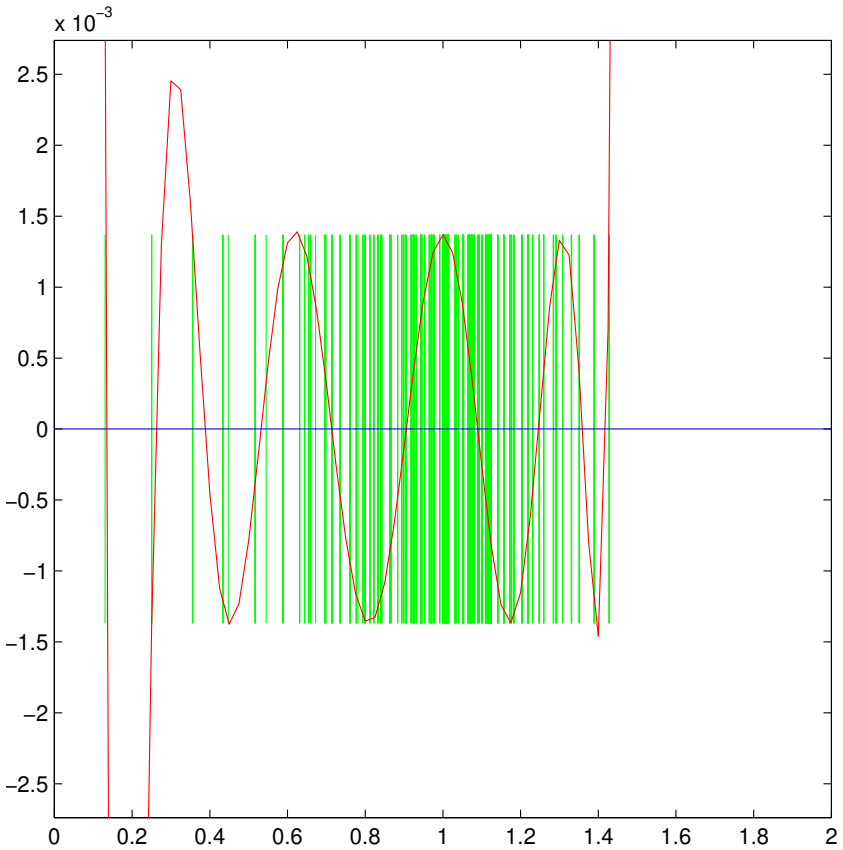


27 point Finite Element Stencil

Tchebyshev error: 0.3918e-2



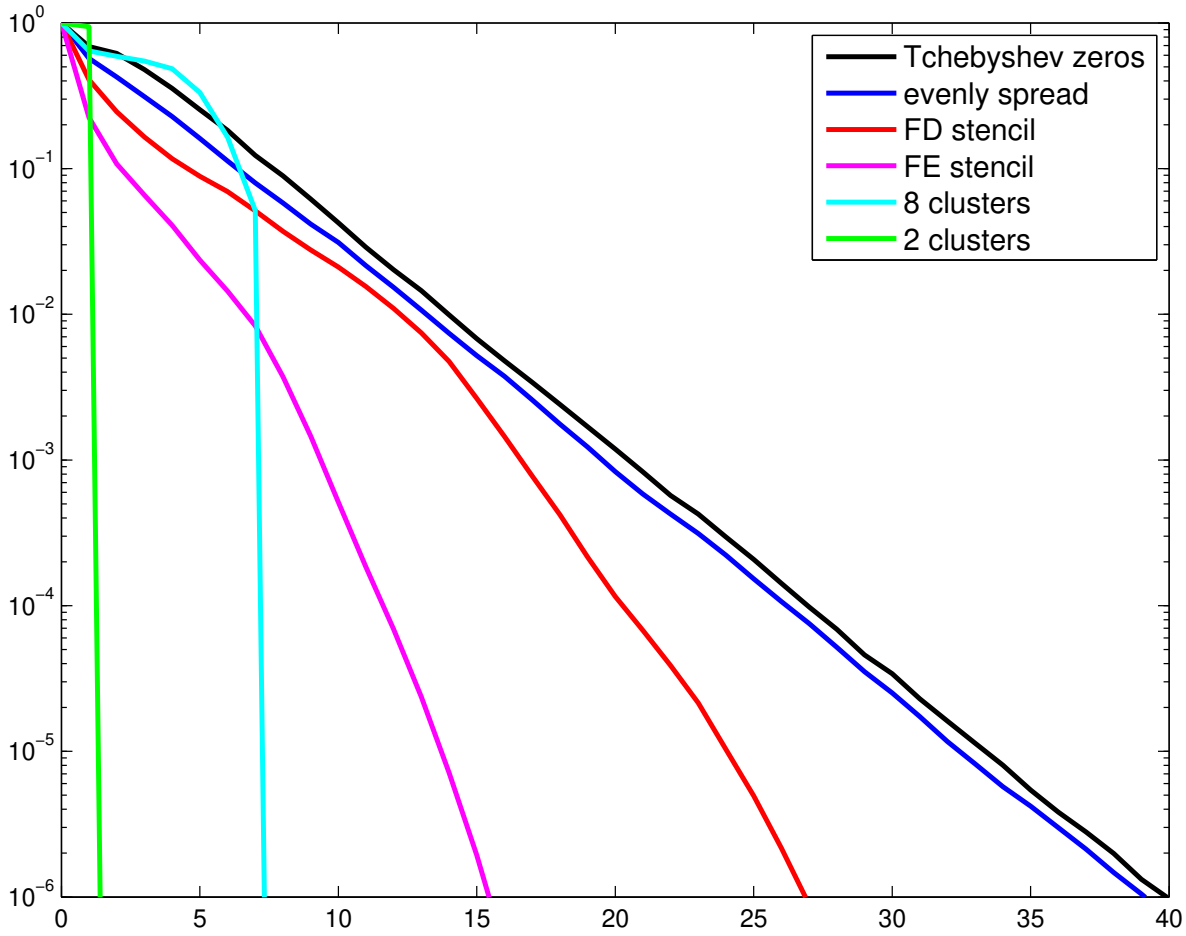
Minimax error: 0.1369e-2



CG residual reduction

512×512 matrices, zero initial guess, random RHS

different eigenvalue spectra, same condition number



CG Method In Practice

- Advantages:
  - involves only matrix-vector and dot products;
  - exact solution obtained in at most s iterations.

CG Method In Practice

- Advantages:
  - involves only matrix-vector and dot products;
  - exact solution obtained in at most s iterations.
- Problems:
  - s may be very large;
  - rounding error means theoretical properties lost.

CG Method In Practice

- Advantages:
  - involves only matrix-vector and dot products;
  - exact solution obtained in at most s iterations.
- Problems:
  - s may be very large;
  - rounding error means theoretical properties lost.
- Solution:
  - treat CG as an iterative method;
  - reduce the number of CG steps required by applying PRECONDITIONING (more later . . .).

Symmetric Indefinite Systems

- If A is symmetric and indefinite:
  - A has both positive and negative (nonzero) eigenvalues;
  - $\mathbf{v}^T A\mathbf{v}$ may equal zero for some N-vector $\mathbf{v} \neq 0$.

Symmetric Indefinite Systems

- If A is symmetric and indefinite:
  - A has both positive and negative (nonzero) eigenvalues;
  - $\mathbf{v}^T A\mathbf{v}$ may equal zero for some N-vector $\mathbf{v} \neq 0$.
- Potential problems with CG:
  - A can no longer be used to define a norm;
  - breakdown may occur: denominator of α_k could be zero (or close to zero).

Conjugate Residual Method (Stiefel (1955))

- Solve $A^2\mathbf{x} = A\mathbf{b}$ by CG method.

CR method constructs iterates

$$\mathbf{x}_k \in \mathbf{x}_0 + \text{span}\{\mathbf{r}_0, A\mathbf{r}_0, \ldots, A^{k-1}\mathbf{r}_0\}$$

with properties

- $\mathbf{x}_k$ minimises $\|\mathbf{x}_k - \hat{\mathbf{x}}\|_{A^2} = \|\mathbf{r}_k\|_2$;
- uses a three-term recurrence relation.

Conjugate Residual Method (Stiefel (1955))

- Solve $A^2\mathbf{x} = A\mathbf{b}$ by CG method.

CR method constructs iterates

$$\mathbf{x}_k \in \mathbf{x}_0 + \text{span}\{\mathbf{r}_0, A\mathbf{r}_0, \ldots, A^{k-1}\mathbf{r}_0\}$$

with properties

- $\mathbf{x}_k$ minimises $\|\mathbf{x}_k - \hat{\mathbf{x}}\|_{A^2} = \|\mathbf{r}_k\|_2$;
- uses a three-term recurrence relation.

- Symmetric eigenvalue intervals: convergence bound gives

$$k \propto \sqrt{\kappa(A^2)} = \kappa(A).$$

CR Algorithm

choose $\mathbf{x}_0$
compute $\mathbf{r}_0 = \mathbf{b} - A\mathbf{x}_0$
set $\mathbf{p}_0 = \mathbf{r}_0$
compute $A\mathbf{p}_0$
for $k = 0$ until convergence do
$$\alpha_k = \mathbf{r}_k^T \mathbf{r}_k / (A\mathbf{p}_k)^T A\mathbf{p}_k$$
$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k$$
$$\mathbf{r}_{k+1} = \mathbf{r}_k - \alpha_k A\mathbf{p}_k$$
$$\beta_k = \mathbf{r}_{k+1}^T \mathbf{r}_{k+1} / \mathbf{r}_k^T A\mathbf{r}_k$$
$$\mathbf{p}_{k+1} = \mathbf{r}_{k+1} + \beta_k \mathbf{p}_k$$
$$A\mathbf{p}_{k+1} = A\mathbf{r}_{k+1} + \beta_k A\mathbf{p}_k$$
end do

can be implemented with one MVM per iteration

Alternative approach

Potential problems with CR:

- breakdown may occur: denominator of α_k could be zero (or close to zero);
- CR algorithm is unstable in this form.

Alternative approach

Potential problems with CR:

- breakdown may occur: denominator of α_k could be zero (or close to zero);
- CR algorithm is unstable in this form.

Possible solution:

- generate an orthonormal basis for $\kappa(A, \mathbf{r}_0, k)$ in a more stable way, retaining the cheap three-term recurrence.

$\Rightarrow$ mathematically equivalent but stable method ...

MINRES

Paige and Saunders (1975)

Construct iterates $\mathbf{x}_k = \mathbf{x}_0 + V_k \mathbf{y}_k$ with properties

- $\mathbf{x}_k$ minimises $\|\mathbf{r}_k\|_2$
- uses three-term recurrence relation

$$V_k = [\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_k]$$

$\mathbf{v}_k$ form an orthonormal basis for $\kappa(A, \mathbf{r}_0, k)$

- use Lanczos method to find $\mathbf{v}_k$
- solve resulting least squares problem for $\mathbf{y}_k$ using Givens rotations and QR factorisation

MINRES Algorithm

Fischer (1996)

choose $\mathbf{x}_0$
compute $\hat{\mathbf{v}}_0 = \mathbf{b} - A\mathbf{x}_0$ initialise
set $\beta_0 = \|\hat{\mathbf{v}}_0\|_2$, $\eta_0 = \beta_0$
set $c_0 = 1$, $c_{-1} = 1$, $s_0 = 0$, $s_{-1} = 0$

for $k = 0$ until convergence do

$$\mathbf{v}_{k+1} = \hat{\mathbf{v}}_k / \beta_k$$
$$\alpha_{k+1} = \mathbf{v}_{k+1}^T A \mathbf{v}_{k+1} \quad \text{Lanczos}$$
$$\hat{\mathbf{v}}_{k+1} = (A - \alpha_{k+1} I)\mathbf{v}_{k+1} - \beta_k \mathbf{v}_k$$
$$\beta_{k+1} = \|\hat{\mathbf{v}}_0\|_2$$

$$\hat{r_1} = c_k \alpha_{k+1} - c_{k-1} s_k \beta_k$$
$$r_1 = \sqrt{\hat{r_1}^2 + \beta_{k+1}^2} \qquad \text{QR}$$
$$r_2 = s_k \alpha_{k+1} + c_{k-1} c_k \beta_k$$
$$r_3 = s_{k-1} \beta_k$$

$$c_{k+1} = \hat{r_1}/r_1 \qquad \text{Givens}$$
$$s_{k+1} = \beta_{k+1}/r_1$$

$$\mathbf{w}_{k+1} = (\mathbf{v}_{k+1} - r_2 \mathbf{w}_k - r_3 \mathbf{w}_{k-1})/r_1$$
$$\mathbf{x}_{k+1} = \mathbf{x}_k + c\eta_k \mathbf{w} \qquad \text{update}$$
$$\eta_{k+1} = -s\eta_k$$

end do

Some alternative methods

SYMMLQ

Paige and Saunders (1975)

Also has a strong Lanczos connection, but minimises the 2-norm of the error rather than the residual.

Some alternative methods

SYMMLQ

Paige and Saunders (1975)

Also has a strong Lanczos connection, but minimises the 2-norm of the error rather than the residual.

ORTHODIR

Fletcher (1976)

ORTHOMIN/ORTHORES

Chandra et al. (1977)

Equivalent to MINRES, closer in implementation to CG/CR.

Nonsymmetric Systems

Faber and Manteuffel (1984 & 1987): there is no Krylov type method which retains both

(i) minimisation property

(ii) short-term recurrence

Nonsymmetric Systems

Faber and Manteuffel (1984 & 1987): there is no Krylov type method which retains both

(i) minimisation property

(ii) short-term recurrence

- Normal Equations
  solve $A^T A\mathbf{x} = A^T\mathbf{b}$ using CG
- Minimum Residual Methods
  retain (i), sacrifice (ii)
- Biorthogonalisation Methods
  retain (ii), sacrifice (i)

Hestenes and Stiefel (1952)

- Apply CG to normal equations $A^TA\mathbf{x} = A^T\mathbf{b}$.

Construct iterates

$$\mathbf{x}_k \in \mathbf{x}_0 + \mathtt{span}\{A^T\mathbf{r}_0, (A^TA)A^T\mathbf{r}_0, \ldots, (A^TA)^{k-1}A^T\mathbf{r}_0\}$$

satisfying

- $\mathbf{x}_k$ minimises $\|\mathbf{x}_k - \hat{\mathbf{x}}\|_{A^TA} = \|\mathbf{r}_k\|_2$
- uses three-term recurrence relation

Hestenes and Stiefel (1952)

- Apply CG to normal equations $A^TA\mathbf{x} = A^T\mathbf{b}$.

Construct iterates

$$\mathbf{x}_k \in \mathbf{x}_0 + \mathtt{span}\{A^T\mathbf{r}_0, (A^TA)A^T\mathbf{r}_0, \ldots, (A^TA)^{k-1}A^T\mathbf{r}_0\}$$

satisfying

- $\mathbf{x}_k$ minimises $\|\mathbf{x}_k - \hat{\mathbf{x}}\|_{A^TA} = \|\mathbf{r}_k\|_2$
- uses three-term recurrence relation

- Convergence analysis gives

$$k \propto \sqrt{\kappa(A^TA)} = \kappa(A)$$

Generalised Minimal Residual Method (GMRES)

Saad and Schultz (1986)

Construct iterates $\mathbf{x}_k = \mathbf{x}_0 + V_k \mathbf{y}_k$ with properties

- $\mathbf{x}_k$ minimises $\|\mathbf{r}_k\|_2$
- no short-term recurrence

$$V_k = [\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_k]$$

$\mathbf{v}_k$ form an orthonormal basis for $\kappa(A, \mathbf{r}_0, k)$

- Use the Arnoldi method to find $\mathbf{v}_k$
- solve resulting least squares problem for $\mathbf{y}_k$ using Givens rotations and QR factorisation

Convergence of GMRES

- Suppose A is diagonalisable as $A = V\Lambda V^{-1}$.

Convergence of GMRES

- Suppose A is diagonalisable as $A = V\Lambda V^{-1}$.

- Residual minimisation property gives

$$\|\mathbf{r}_k\|_2 \leq \min_{p \in \Pi_k^1} \max_{z \in R} |p(z)| \|V\|_2 \|V^{-1}\|_2 \|\mathbf{r}_0\|_2$$

  where R is any region containing the eigenvalues.

Convergence of GMRES

- Suppose A is diagonalisable as $A = V\Lambda V^{-1}$.

- Residual minimisation property gives

$$\|\mathbf{r}_k\|_2 \leq \min_{p \in \Pi_k^1} \max_{z \in R} |p(z)| \|V\|_2 \|V^{-1}\|_2 \|\mathbf{r}_0\|_2$$

  where R is any region containing the eigenvalues.

- Usually very difficult to obtain any reasonable estimate of the eigenvector condition number $\|V\|_2 \|V^{-1}\|_2$.

Some observations

- Other convergence analysis based on singular values, pseudo-eigenvalues or field of values.

Some observations

- Other convergence analysis based on singular values, pseudo-eigenvalues or field of values.
- Many alternative implementations of GMRES available e.g. based on Householder orthogonalisation: extra work but better numerical properties.

Some observations

- Other convergence analysis based on singular values, pseudo-eigenvalues or field of values.
- Many alternative implementations of GMRES available e.g. based on Householder orthogonalisation: extra work but better numerical properties.
- Restarted GMRES
  - restart GMRES every m steps;
  - no simple rule for choosing m: convergence speed may vary drastically with different values;
  - some convergence analysis available.

Some observations

- Other convergence analysis based on singular values, pseudo-eigenvalues or field of values.
- Many alternative implementations of GMRES available e.g. based on Householder orthogonalisation: extra work but better numerical properties.
- Restarted GMRES
  - restart GMRES every m steps;
  - no simple rule for choosing m: convergence speed may vary drastically with different values;
  - some convergence analysis available.
- Lots of work on adapting and improving GMRES variants.

BiCG method

Construct iterates

$$\mathbf{x}_k = \mathbf{x}_0 + \mathrm{span}\{\mathbf{r}_0, A\mathbf{r}_0, \ldots, A^{k-1}\mathbf{r}_0\}$$

with properties

- $\mathbf{r}_k \perp \mathrm{span}\{\hat{\mathbf{r}}_0, A\hat{\mathbf{r}}_0, \ldots, A^{k-1}\hat{\mathbf{r}}_0\}$
- uses three-term recurrence relation

- Uses nonsymmetric Lanczos: generate two sets of biorthogonal vectors.

BiCG method

Construct iterates

$$\mathbf{x}_k = \mathbf{x}_0 + \mathrm{span}\{\mathbf{r}_0, A\mathbf{r}_0, \ldots, A^{k-1}\mathbf{r}_0\}$$

with properties

- $\mathbf{r}_k \perp \mathrm{span}\{\hat{\mathbf{r}}_0, A\hat{\mathbf{r}}_0, \ldots, A^{k-1}\hat{\mathbf{r}}_0\}$
- uses three-term recurrence relation

- Uses nonsymmetric Lanczos: generate two sets of biorthogonal vectors.
- Potential problems:
  - wild oscillations in $\|\mathbf{r}_k\|_2$
  - possible breakdowns: $\hat{\mathbf{p}}_{k-1}^T A\mathbf{p}_{k-1} = 0$, $\hat{\mathbf{r}}_{k-1}^T \mathbf{r}_{k-1} = 0$
  - possible solution: look-ahead Lanczos

Biconjugate Gradient Method (BiCG)

Lanczos (1952), Fletcher (1976)

choose $\mathbf{x}_0$, compute $\mathbf{p}_0 = \mathbf{r}_0 = \mathbf{b} - A\mathbf{x}_0$

choose $\hat{\mathbf{r}}_0$

set $\hat{\mathbf{p}}_0 = \hat{\mathbf{r}}_0, \rho_0 = \hat{\mathbf{r}}_0^T \mathbf{r}_0$

for $k = 1, 2, \ldots$ until convergence do

$$\sigma_{k-1} = \hat{\mathbf{p}}_{k-1}^T A\mathbf{p}_{k-1}$$
$$\alpha_{k-1} = \rho_{k-1}/\sigma_{k-1}$$
$$\mathbf{x}_k = \mathbf{x}_{k-1} + \alpha_{k-1}\mathbf{p}_{k-1}$$
$$\mathbf{r}_k = \mathbf{r}_{k-1} - \alpha_{k-1} A\mathbf{p}_{k-1}$$
$$\hat{\mathbf{r}}_k = \hat{\mathbf{r}}_{k-1} - \alpha_{k-1} A^T \hat{\mathbf{p}}_{k-1}$$
$$\rho_k = \hat{\mathbf{r}}_k^T \mathbf{r}_k$$
$$\beta_{k-1} = \rho_k/\rho_{k-1}$$
$$\mathbf{p}_k = \mathbf{r}_k + \beta_k \mathbf{p}_{k-1}$$
$$\hat{\mathbf{p}}_k = \hat{\mathbf{r}}_k + \beta_k \hat{\mathbf{p}}_{k-1}$$

end

Quasi-Minimal Residual Method (QMR)

Freund and Nachtigal (1991)

- Based on GMRES using nonsymmetric Lanczos with a biorthogonal basis.
- Too expensive to minimise $\|\mathbf{r}_k\|_2$: minimise "nearby" quantity: quasi-minimal.
- Avoid Lanczos breakdown: do l steps of look-ahead Lanczos.
- Incurable breakdown (unlikely due to round-off).
- Some convergence results are available.
- If A is symmetric, QMR $\equiv$ MINRES.

Transpose-free Methods

Transpose-Free QMR (TFQMR)

Freund (1991), Chan et al. (1991), Freund and Szeto (1991)

- A^T can be eliminated by choosing a suitable starting vector

Transpose-free Methods

Transpose-Free QMR (TFQMR)

Freund (1991), Chan et al. (1991), Freund and Szeto (1991)

- A^T can be eliminated by choosing a suitable starting vector

Conjugate Gradients Squared (CGS)

Sonneveld (1989)

- construct iterates $\mathbf{x}_{2k} = \mathbf{x}_0 + \kappa(A, \mathbf{r}_0, 2k)$
- residual polynomials of CG are squared
- magnifies erratic convergence of BiCG
- may diverge when BiCG converges

BiCGSTAB

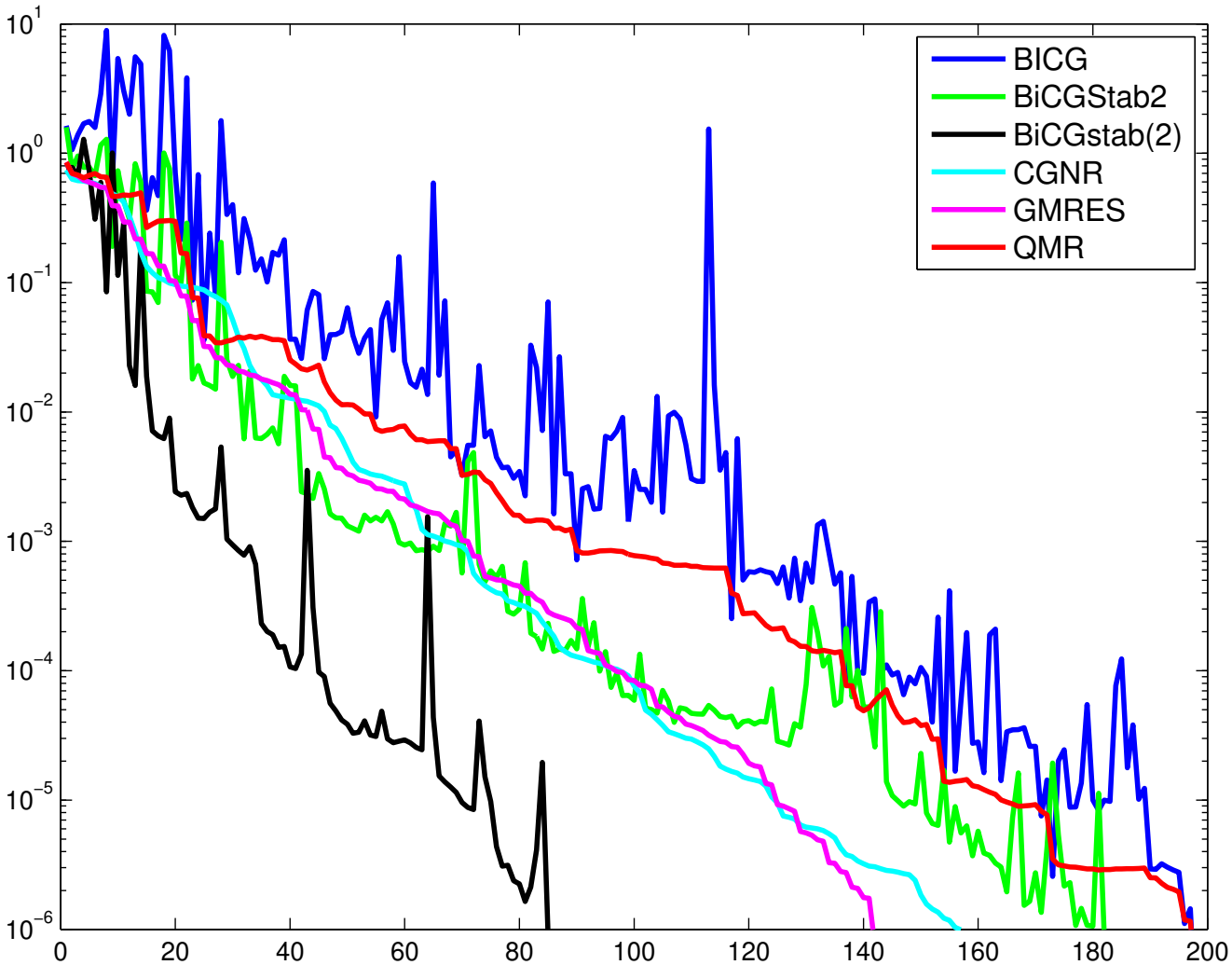
van der Vorst (1990)

- Construct iterates $\mathbf{x}_{2k} = \mathbf{x}_0 + \kappa(A, \mathbf{r}_0, 2k)$.
- Residual polynomial updated with a linear factor at each step.
- Free parameter μ_k determined via a local steepest descents problem.
- Convergence typically much smoother than CGS.

BiCGStab2, Gutnecht (1993)
BiCGstab(l), Sleijpen and Fokkema (1993)

Example: Calculation of Invariant Tori

$$a(X,Y)\frac{\partial s}{\partial X} + b(X,Y)\frac{\partial s}{\partial Y} + c(X,Y)s = \psi$$



Which method to choose?

How Fast Are Nonsymmetric Matrix Iterations?

N.M. Nachtigal, S.C. Reddy & L.N. Trefethen

SIAM J. MATRIX ANAL. APPL. 13(3), 1992

Which method to choose?

How Fast Are Nonsymmetric Matrix Iterations?

N.M. Nachtigal, S.C. Reddy & L.N. Trefethen

SIAM J. MATRIX ANAL. APPL. 13(3), 1992

- Compare three different methods:
  - CGNR (CG for normal equations)
  - GMRES (minimises but no short-term recurrence)
  - CGS (short-term recurrence but no minimisation)

Which method to choose?

How Fast Are Nonsymmetric Matrix Iterations?

N.M. Nachtigal, S.C. Reddy & L.N. Trefethen

SIAM J. MATRIX ANAL. APPL. 13(3), 1992

- Compare three different methods:
  - CGNR (CG for normal equations)
  - GMRES (minimises but no short-term recurrence)
  - CGS (short-term recurrence but no minimisation)

- Eight test examples to compare the performance.
  - 1 example: all methods good.
  - 1 example: all methods bad.
  - 3 examples: each method is best.
  - 3 examples: each method is worst.

Which method to choose?

How Fast Are Nonsymmetric Matrix Iterations?
N.M. Nachtigal, S.C. Reddy & L.N. Trefethen
SIAM J. MATRIX ANAL. APPL. 13(3), 1992

- Compare three different methods:
  - CGNR (CG for normal equations)
  - GMRES (minimises but no short-term recurrence)
  - CGS (short-term recurrence but no minimisation)

- Eight test examples to compare the performance.
  - 1 example: all methods good.
  - 1 example: all methods bad.
  - 3 examples: each method is best.
  - 3 examples: each method is worst.

- Final choice often depends on application...

Summary

- symmetric positive definite CG
- symmetric indefinite CR, MINRES, SYMMLQ
- nonsymmetric
  - normal equations
    CGNR
  - minimisation
    GMRES, GMRES(m)
  - biorthogonalisation
    BiCG, CGS, BiCGSTAB, BiCGstab(l), QMR

Examples of Stopping Criteria

- standard tests: $\|\mathbf{r}_k\|_2 \leq \epsilon, \quad \frac{\|\mathbf{r}_k\|_2}{\|\mathbf{r}_0\|_2} \leq \epsilon$

Examples of Stopping Criteria

- standard tests: $\|\mathbf{r}_k\|_2 \leq \epsilon, \quad \frac{\|\mathbf{r}_k\|_2}{\|\mathbf{r}_0\|_2} \leq \epsilon$

- condition number dependent (e.g. Ashby et al. (1990)):

$$\frac{\|\mathbf{x}_k - \hat{\mathbf{x}}\|_2}{\|\mathbf{x}_0 - \hat{\mathbf{x}}\|_2} \leq \kappa(A) \frac{\|\mathbf{r}_k\|_2}{\|\mathbf{r}_0\|_2} \leq \epsilon$$

$$\frac{\|\mathbf{x}_k - \hat{\mathbf{x}}\|_A}{\|\mathbf{x}_0 - \hat{\mathbf{x}}\|_A} \leq \left(\kappa_A(A) \left| \frac{\gamma_k}{\gamma_0} \right| \right)^{\frac{1}{2}} \leq \epsilon$$

Examples of Stopping Criteria

- standard tests: $\|\mathbf{r}_k\|_2 \leq \epsilon, \quad \frac{\|\mathbf{r}_k\|_2}{\|\mathbf{r}_0\|_2} \leq \epsilon$

- condition number dependent (e.g. Ashby et al. (1990)):

$$\frac{\|\mathbf{x}_k - \hat{\mathbf{x}}\|_2}{\|\mathbf{x}_0 - \hat{\mathbf{x}}\|_2} \leq \kappa(A) \frac{\|\mathbf{r}_k\|_2}{\|\mathbf{r}_0\|_2} \leq \epsilon$$

$$\frac{\|\mathbf{x}_k - \hat{\mathbf{x}}\|_A}{\|\mathbf{x}_0 - \hat{\mathbf{x}}\|_A} \leq \left(\kappa_A(A) \left| \frac{\gamma_k}{\gamma_0} \right| \right)^{\frac{1}{2}} \leq \epsilon$$

- backward error analysis (e.g. Arioli et al. (1991)):

$$\frac{\|\mathbf{r}_k\|_\infty}{\|A\|_\infty \|\mathbf{x}_k\|_1 + \|\mathbf{r}_0\|_\infty} \leq \epsilon$$

Some Relevant Books and Review Papers

- Iterative Solution of Linear Systems, Freund, Golub and Nachtigal, Acta Numerica (1991)
- Templates for the Solution of Linear Systems. . . , Barrett et al., SIAM (1994)
- Iterative Solution Methods, Axelsson, CUP (1996)
- Iterative Methods for Sparse Linear Systems, Saad, PWS (1996)
- Iterative Methods for Solving Linear Systems, Greenbaum, SIAM (1997)
- Iterative Solution of Linear Systems in the 20th Century, Saad and van der Vorst, J. Comp. and Appl. Math. 123 (2000)
- Iterative Krylov Methods for Large Linear Systems, van der Vorst, CUP (2003)

Preconditioning

- Idea: instead of solving $A\mathbf{x} = \mathbf{b}$, solve

$$M^{-1}A\mathbf{x} = M^{-1}\mathbf{b}$$

for some preconditioner M.

Preconditioning

- Idea: instead of solving $A\mathbf{x} = \mathbf{b}$, solve

$$M^{-1}A\mathbf{x} = M^{-1}\mathbf{b}$$

  for some preconditioner M.

- Choose M so that

  (i) eigenvalues of $M^{-1}A$ are well clustered;
  (ii) $M\mathbf{u} = \mathbf{r}$ is easily solved.

Preconditioning

- Idea: instead of solving $A\mathbf{x} = \mathbf{b}$, solve

$$M^{-1}A\mathbf{x} = M^{-1}\mathbf{b}$$

  for some preconditioner M.

- Choose M so that

  (i) eigenvalues of $M^{-1}A$ are well clustered;
  (ii) $M\mathbf{u} = \mathbf{r}$ is easily solved.

- Extreme cases:
  - $M = A$: good for (i), bad for (ii);
  - $M = I$: good for (ii), bad for (i).

Preconditioned Conjugate Gradient Method

Concus, Golub & O'Leary (1976)

choose $\mathbf{x}_0$
compute $\mathbf{r}_0 = \mathbf{b} - A\mathbf{x}_0$
solve $M\hat{\mathbf{r}}_0 = \mathbf{r}_0$
set $\mathbf{p}_0 = \mathbf{r}_0$
for $k = 0$ until convergence do
$\alpha_k = \mathbf{r}_k^T \hat{\mathbf{r}}_k / \mathbf{p}_k^T A\mathbf{p}_k$
$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k$
$\mathbf{r}_{k+1} = \mathbf{r}_k - \alpha_k A\mathbf{p}_k$
solve $M\hat{\mathbf{r}}_{k+1} = \mathbf{r}_{k+1}$
$\beta_k = \mathbf{r}_{k+1}^T \hat{\mathbf{r}}_{k+1} / \mathbf{r}_k^T \hat{\mathbf{r}}_k$
$\mathbf{p}_{k+1} = \hat{\mathbf{r}}_{k+1} + \beta_k \mathbf{p}_k$
end do

Practical Implementation

- Preconditioner M:
  - Left preconditioning

$$M^{-1}A\mathbf{x} = M^{-1}\mathbf{b}$$

  - Right preconditioning

$$AM^{-1}\mathbf{y} = \mathbf{b}, \qquad \mathbf{x} = M^{-1}\mathbf{y}$$

Practical Implementation

- Preconditioner M:
  - Left preconditioning

$$M^{-1}A\mathbf{x} = M^{-1}\mathbf{b}$$

  - Right preconditioning

$$AM^{-1}\mathbf{y} = \mathbf{b}, \qquad \mathbf{x} = M^{-1}\mathbf{y}$$

- Preconditioner $M = M_1 M_2$:
  - Split preconditioning

$$[M_1^{-1}AM_2^{-1}]\mathbf{y} = M_1^{-1}\mathbf{b}, \qquad \mathbf{x} = M_2^{-1}\mathbf{y}$$

- symmetric: split preconditioner retains symmetry.
  - positive definite: if $M_2 = M_1^T$, resulting system is also symmetric positive definite.
  - indefinite: M must be symmetric positive definite for MINRES; M can be indefinite with QMR.

- symmetric: split preconditioner retains symmetry.
  - positive definite: if $M_2 = M_1^T$, resulting system is also symmetric positive definite.
  - indefinite: M must be symmetric positive definite for MINRES; M can be indefinite with QMR.

- nonsymmetric:
  - split: analysis may be easier.
  - left: if $M^{-1}A \simeq I$, $\tilde{\mathbf{r}}_k = M^{-1}A(\mathbf{x}_k - \hat{\mathbf{x}}) \simeq \mathbf{x}_k - \hat{\mathbf{x}}$, i.e.,

$$\|\tilde{\mathbf{r}}_k\|_2 \simeq \|\mathbf{x}_k - \hat{\mathbf{x}}\|_2.$$

  - right: minimise in same norm, i.e.,

$$\|\tilde{\mathbf{r}}_k\|_2 = \|\mathbf{r}_k\|_2.$$

Connection with stationary methods

matrix splitting $A = M - N$

- Iterates

$$\mathbf{x}_{k+1} = M^{-1}N\mathbf{x}_k + M^{-1}\mathbf{b} = \mathbf{x}_k + M^{-1}\mathbf{r}_k$$

where the error satisfies

$$\mathbf{x}_k - \hat{\mathbf{x}} = (I - M^{-1}A)^k(\mathbf{x}_0 - \hat{\mathbf{x}}).$$

Connection with stationary methods

matrix splitting $A = M - N$

- Iterates

$$\mathbf{x}_{k+1} = M^{-1}N\mathbf{x}_k + M^{-1}\mathbf{b} = \mathbf{x}_k + M^{-1}\mathbf{r}_k$$

where the error satisfies

$$\mathbf{x}_k - \hat{\mathbf{x}} = (I - M^{-1}A)^k(\mathbf{x}_0 - \hat{\mathbf{x}}).$$

- If $(I - M^{-1}A)$ is small, expect rapid convergence.
- So good preconditioner $\equiv$ good splitting operator.

Stationary Methods as Preconditioners

- Jacobi (diagonal scaling)
  - very simple to implement, minimal storage requirements
  - scales condition number
  - still competitive for extremely large 3D problems: may be better to do more cheaper iterations than fewer expensive ones

Stationary Methods as Preconditioners

- Jacobi (diagonal scaling)
  - very simple to implement, minimal storage requirements
  - scales condition number
  - still competitive for extremely large 3D problems: may be better to do more cheaper iterations than fewer expensive ones
- Gauss-Seidel, SOR
  - for CG, M must be symmetric
  - applying the method twice per iteration (once *forward* then once *backward*)

Stationary Methods as Preconditioners

- Jacobi (diagonal scaling)
  - very simple to implement, minimal storage requirements
  - scales condition number
  - still competitive for extremely large 3D problems: may be better to do more cheaper iterations than fewer expensive ones
- Gauss-Seidel, SOR
  - for CG, M must be symmetric
  - applying the method twice per iteration (once *forward* then once *backward*)
- Preconditioners based on stationary methods are typically easy to use and widely applicable. But there is a whole research field dedicated to other methods...

Incomplete LU Factorisation

Step 1: select set $J = \{(i,j) : 1 \leq i,j \leq N\}$ of index pairs (including all (i,i))

Step 2: perform LU factorisation and restrict all non-zeros to entries in J

$$A = LU - R = M - R$$

$$r_{ij} = 0, \ (i,j) \in J, \qquad r_{ii} = \alpha \sum_{i \neq j} r_{ij}$$

- ILU factorisations do not always exist
- very sequential in nature
- block matrix analogues

Some Variations on ILU

- $J \equiv$ nonzero entries in A

  $\alpha = 0$: ILU, Meijerink and van der Vorst (1977)
  $\alpha = 1$: MILU, Gustafsson (1978)

- ILU(N), MILU(N)
  J includes N extra diagonals

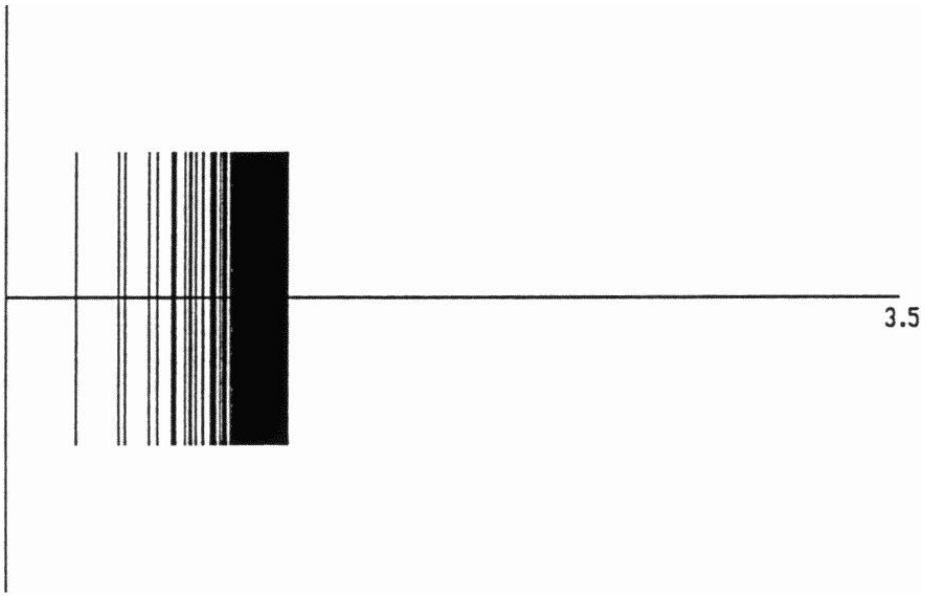
- ILU with Drop Tolerance, Munksgaard (1980)
  Drop all entries of fill-in with absolute value less than $\tau \in [10^{-4}, 10^{-2}]$.

- Shifted ILU, Manteuffel (1978,1980)
  Make A more diagonally dominant by factorising

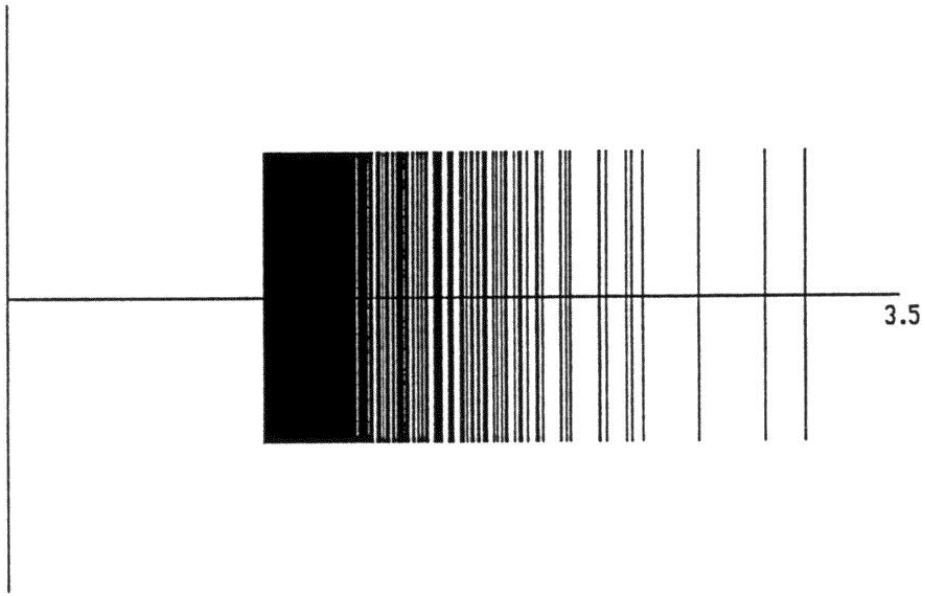
$$\bar{A} = D + \frac{1}{1+\gamma} C.$$

Sample Eigenvalue Plots

- seven point finite difference stencil



Incomplete Cholesky



Modified Incomplete Cholesky

SParse Approximate Inverse (SPAI)

- Minimise $\|AM - I\|$ in the Frobenius norm.
- Approximate inverse computed explicitly so can be applied as a preconditioner.
- Sparsity pattern of approximate inverse calculated dynamically.
- User controls the quality and cost:
  - if M is sparse, it is cheap to compute but may not improve things much;
  - as M becomes dense, it becomes more expensive to compute
  - optimal preconditioner; lies between these two extremes and is problem and computer architecture dependent.

Polynomial Preconditioning

- Apply CG to $p(B^{-1}A)B^{-1}A\mathbf{x} = p(B^{-1}A)B^{-1}\mathbf{b}$ i.e. use

$$M^{-1} = p(B^{-1}A)B^{-1}$$

so that $\mathbf{u} = M^{-1}\mathbf{r}$ is easily solved.

- Choose B to be a matrix splitting from stationary methods: e.g.
  - B from SSOR gives m-step CG method (Adams (1985)),
  - $B = I$ from Richardson (Ashby (1987)),
  - applying preconditioner involves only MVMs so may be good for vector/parallel machines.

Examples of Preconditioners for FE problems

- Element-By-Element Method
  Preconditioner is a product of factors of assembly of individual element matrices.

- Element Factorisation Method
  Preconditioner is a product of assembly of individual element matrix factorisations.

- Hierarchical Basis Preconditioning
  Based on using hierarchical bases for the finite element spaces instead of the usual nodal bases.

- ...

Domain Decomposition

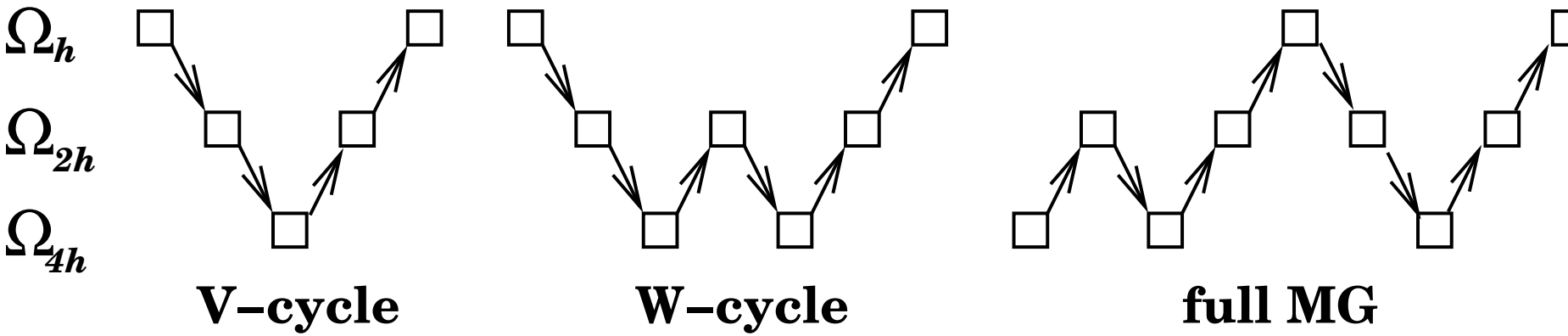
- Some PDE solvers can be used as preconditioners.
- Domain decomposition: break down underlying elliptic PDE problem into distinct parts that can be solved separately.
- Piece results together to get solution to whole problem .
- Use different preconditioners for different parts of the grid.
- Two main types:
  - overlapping methods: additive/multiplicative Schwarz;
  - nonoverlapping methods: substructuring.

Multigrid Methods

- Developed for solving boundary value problems.
- geometric multigrid (GMG)
  - Based on sequence of physical grids associated with the problem.
- algebraic multigrid (AMG)
  - No need for a physical grid, based on sparse matrix properties.
- Both methods very powerful when used either as solvers in their own right or as preconditioners.

A Multigrid Tutorial, Briggs et al. (2000).

Multigrid Cycles



- down arrows represent restriction
- up arrows represent prolongation
- smoothing steps are performed on each level
- patterns extend to as many grids as needed
- use direct method on coarsest grid

GMG Convergence

- For elliptic PDE problems on uniform grids, convergence analysis can be done using Fourier modes but general convergence analysis is tricky
- With the right combination of components, the number of operations involved is of the order of the total number of unknowns in the linear system, i.e. work $\propto N$.
- This is the best possible (it takes N operations to write the solution down!)
- The number of iterations is independent of N and does not grow as the underlying finite element grid is refined.
- MG is scalable and is amenable to parallel implementation.

AMG Convergence

- No grid needed, works on a sparse matrix.
- Coarse-grid unknowns are a subset of the variables, identified by numerical indices.
- Two-grid correction scheme exactly the same as for GMG, with recursive calls used to set up various patterns on a full series of "grids".
- It has a broad range of applicability but the theory of its convergence behaviour is relatively undeveloped.
- AMG replicates the attractive $O(N)$ behaviour shown by GMG, which is great for, e.g., unstructured grid problems.
- Developments towards element-based versions for parallelisation (e.g. BoomerAMG).

Points to note on multigrid

- MG applications include elliptic,parabolic and hyperbolic PDEs, integral equations, evolution problems, ...
- There are also nonlinear and anisotropic versions.
- MG methods often used as preconditioners:
  - typically only one or two V-cycles are required per CG/GMRES iteration;
  - this is often more robust as an MG preconditioner tends to be less sensitive to the tuning of specifics such as grid transfer operators, type and amount of smoothing, etc.

Far too many other methods to mention!

- Block preconditioners.
- Constraint preconditioners for saddle-point problems.
- Low-rank updates to preconditioners for solving a sequence of problems.
- All-at-once preconditioners for large time-dependent problems.
- Preconditioners using randomized linear algebra.
- Stochastic preconditioners for problems with uncertain coefficients.
- Tensor-based preconditioners for reducing memory requirements.
- Preconditioners designed for HPC architechtures.
- ...

Examples of software packages

- Many methods available in Matlab.
- Some examples of packages on `www.netlib.org`:
  - `ITPACK` (FORTRAN),
  - `linalg/laspack` (C),
  - `linalg/qmrpack` (FORTRAN),
  - `linalg/templates` (C, FORTRAN),
  - `linalg/cg` (PVM).
- Some other accessible codes:
  - `HYPRE` (FORTRAN,C),
  - `AZTEC` (MPI),
  - `LAPACK` (FORTRAN),
  - `TRILINOS` (C++).
  - `PETSc` (FORTRAN,C,Python).
  - `IFISS` (Matlab, Octave)
  - `FENicS` (C++,Python)

Examples of software packages

- Many methods available in Matlab.
- Some examples of packages on `www.netlib.org`:
  - `ITPACK` (FORTRAN),
  - `linalg/laspack` (C),
  - `linalg/qmrpack` (FORTRAN),
  - `linalg/templates` (C, FORTRAN),
  - `linalg/cg` (PVM).
- Some other accessible codes:
  - `HYPRE` (FORTRAN,C),
  - `AZTEC` (MPI),
  - `LAPACK` (FORTRAN),
  - `TRILINOS` (C++).
  - `PETSc` (FORTRAN,C,Python).
  - `IFISS` (Matlab, Octave)
  - `FENicS` (C++,Python)
- Don't re-invent the wheel...

Some relevant books and papers

- Preconditioning Techniques for Large Linear Systems: A Survey, Benzi, J. Comput. Phys. (2002)
- Preconditioning, Wathen, Acta Numerica (2015)
- Iterative Methods and Preconditioning for Large and Sparse Linear Systems with Applications, Durastante and Bertaccini, Chapman and Hall (2018)
- Preconditioners for Krylov subspace methods: An overview, Pearson and Pestana, GAMM-Mitteilungen (2020)
- Preconditioning for linear systems, Jarlebring et al., independently published (2020)
- Iterative Methods and Preconditioners for Systems of Linear Equations, Ciaramella and Gander, SIAM (2022)