Numerical Linear Algebra

Numerical Linear Algebra

Notation: $A \in \mathbb{R}^{m \times n}$: real matrix, m rows and n columns

$\mathbb{R}^{m \times 1} = \mathbb{R}^m$ are (column) vectors

Common Problems

- Eigenvalue Problem: given $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ find $\boldsymbol{\lambda} \in \mathbb{R}$ or $\mathbb{C}$

  and $\boldsymbol{x} \in \mathbb{R}^n$ or $\mathbb{C}^n$ such that $\boldsymbol{Ax} = \boldsymbol{\lambda x}$

- Linear System: given $\boldsymbol{A} \in \mathbb{R}^{n \times n}, \boldsymbol{b} \in \mathbb{R}^n$ find $\boldsymbol{x} \in \mathbb{R}^n$ s.t. $\boldsymbol{Ax} = \boldsymbol{b}$

- Nonlinear system: almost always reduce to sequence of linear systems eg. Newtons method for $\mathbf{F}(\mathbf{U}) = \mathbf{0}$

  Guess $\mathbf{U_0}$ and for $\boldsymbol{k} = 1, 2, \ldots$
  solve $\boldsymbol{J\delta} = -\mathbf{F}(\mathbf{U}_k)$ and set $\mathbf{U}_{k+1} = \mathbf{U}_k + \boldsymbol{\delta}$

  $$\boldsymbol{J} = \{\boldsymbol{J}_{i,j}\}, \boldsymbol{J}_{i,j} = \frac{\boldsymbol{\partial F}_i}{\boldsymbol{\partial x}_j}(\mathbf{U}_k), \quad \text{Jacobian}$$

- Least Squares (Regression): given $\boldsymbol{A} \in \mathbb{R}^{m \times n}$, $\boldsymbol{m} > \boldsymbol{n}$

In different situations A can be:

- Full (dense) - most entries non-zero
- Banded: $\exists b < n$ with $a_{i,j} = 0$ if $|i - j| > b$
  - diagonal: $b = 0$
  - tridiagonal: $b = 1$
- Triangular: upper triangular: $a_{i,j} = 0$ if $i > j$, correspondingly lower
- Hessenberg: upper Hessenberg: $a_{i,j} = 0$ if $i > j + 1$, correspondingly lower
- in Block form: naturally expressed in terms of sub-matrices (matrix blocks)
- Sparse: many zero entries, often very few non-zeros per row

Block Matrices: If $(m_1, \cdots, m_p)$ and $(n_1, \cdots, n_q)$ are sets of positive integers with $\sum_i m_i = m$, $\sum_j n_j = n$, and $A_{ij} \in \mathbb{R}^{m_i \times n_j}$, $i = 1, \cdots, p$, $j = 1, \cdots, q$ then

$$A = [A_{ij}] = \begin{bmatrix} A_{11} & A_{12} & \cdots & A_{1q} \\ A_{21} & A_{22} & \cdots & A_{2q} \\ \vdots & & & \vdots \\ A_{p1} & A_{p2} & \cdots & A_{pq} \end{bmatrix} \in \mathbb{R}^{m \times n}$$

is a $p \times q$ block matrix.

Names for ordinary matrices apply to block matrices, e.g.

$$T = \begin{bmatrix} A & B \\ O & C \end{bmatrix} \begin{matrix} r \\ s \end{matrix}$$
$$\begin{matrix} r & s \end{matrix}$$

is 2×2 block upper triangular;

Matrix and vector multiplication work for block matrices: eg if as above

$$T = \begin{bmatrix} A & B \\ O & C \end{bmatrix} \begin{matrix} r \\ s \end{matrix}, \quad U = \begin{bmatrix} D & E \\ O & F \end{bmatrix} \begin{matrix} r \\ s \end{matrix}, \quad x = \begin{bmatrix} y \\ z \end{bmatrix}$$
$$\begin{matrix} r & s \end{matrix} \qquad\qquad \begin{matrix} r & s \end{matrix}$$

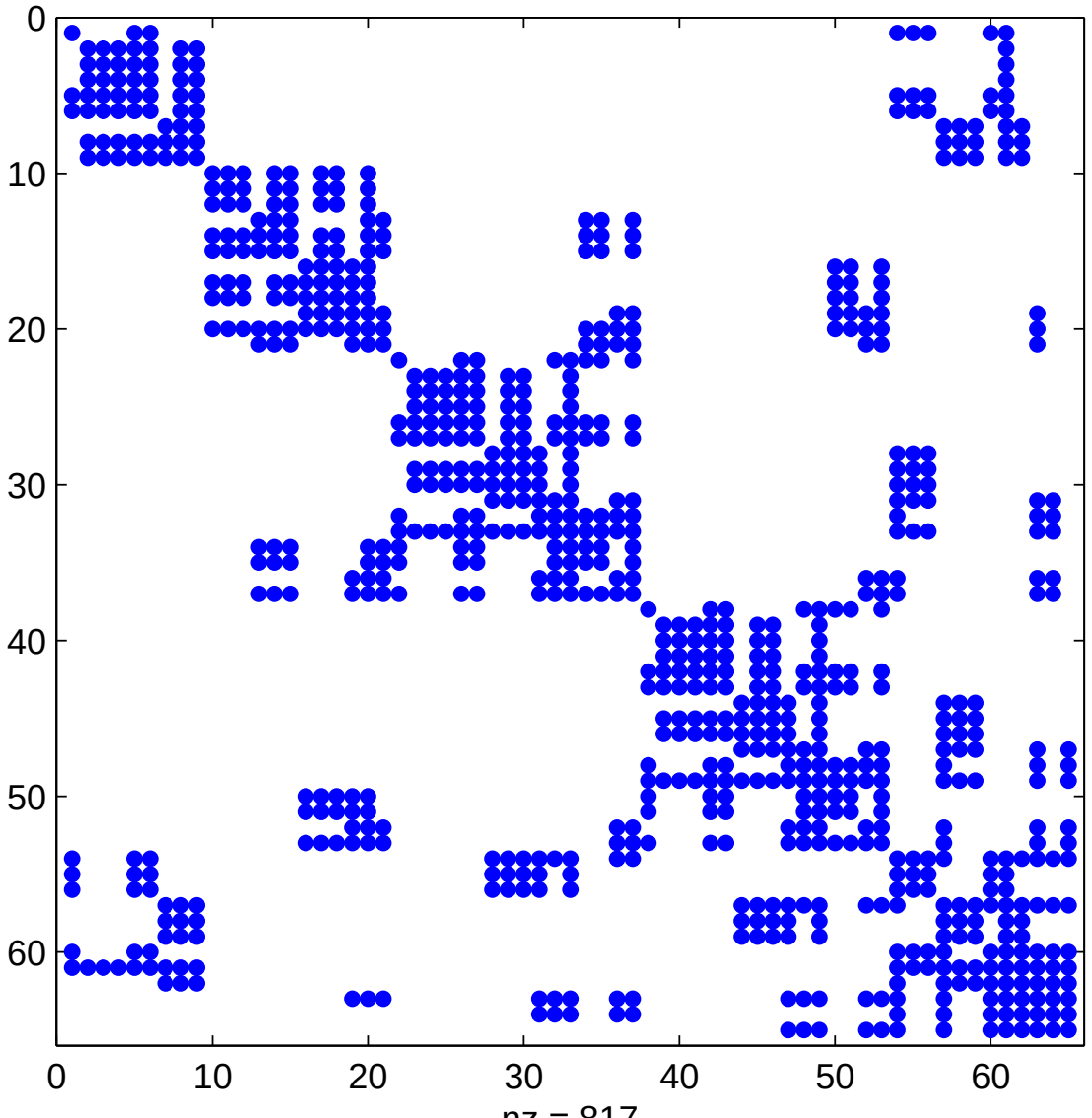
then

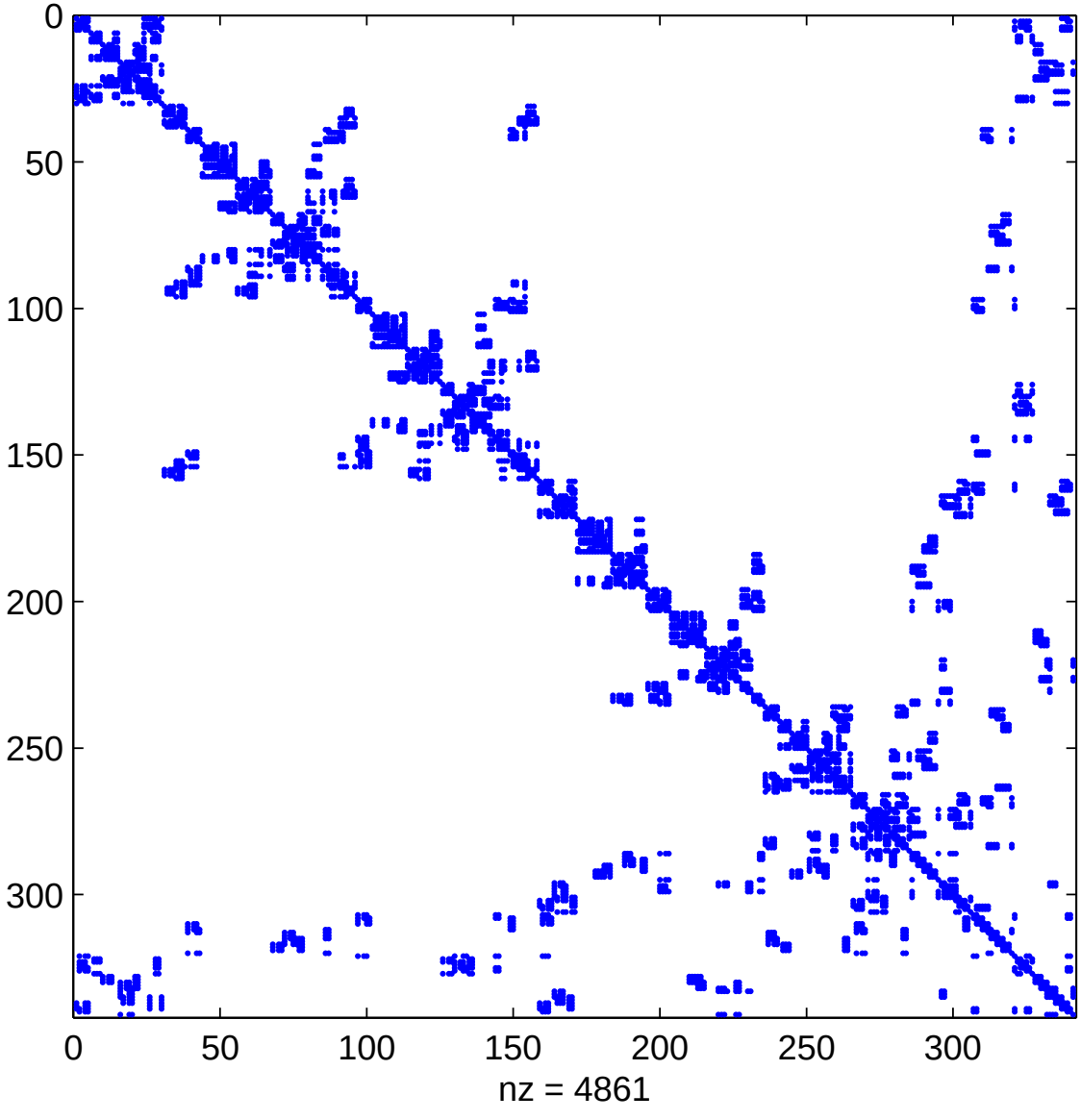
$$TU = \begin{bmatrix} AD & AE + BF \\ O & CF \end{bmatrix}, \quad Tx = \begin{bmatrix} Ay + Bz \\ Cz \end{bmatrix}.$$

Exercise: Deduce that T^{-1} exists if and only if A^{-1} and C^{-1} do, and equals

$$\begin{bmatrix} A^{-1} & -A^{-1}BC^{-1} \\ O & C^{-1} \end{bmatrix}$$

Sparse Matrices




0
50
100
150
200
250
300
0
50
100
150
200
250
300
nz = 4861

and in any of these structures a square matrix $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ may be

- symmetric: $\boldsymbol{A}^T = \boldsymbol{A} \qquad (\boldsymbol{a}_{i,j} = \boldsymbol{a}_{j,i})$
- skew-symmetric: $\boldsymbol{A}^T = -\boldsymbol{A}$
- positive definite: $\boldsymbol{x}^T \boldsymbol{A} \boldsymbol{x} > 0$ for all non-zero $\boldsymbol{x} \in \mathbb{R}^n$
- positive semi-definite: $\boldsymbol{x}^T \boldsymbol{A} \boldsymbol{x} \geq 0$ for all non-zero $\boldsymbol{x} \in \mathbb{R}^n$
- indefinite: $(\boldsymbol{x}^T \boldsymbol{A} \boldsymbol{x})(\boldsymbol{y}^T \boldsymbol{A} \boldsymbol{y}) < 0$ for some $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$
- $\ldots$

Different structures call for different numerical techniques

Goal of a numerical method is often to convert a given problem to a simpler form (where the solution may be

Orthogonal Matrices

$Q \in \mathbb{R}^{n \times n}$ is orthogonal if

(i) $Q^T Q = I$ ie. $Q^T = Q^{-1}$

equivalently

(ii) $QQ^T = I$

(iii) columns of Q are an orthonormal basis for $\mathbb{R}^n$

(iv) rows of Q when transposed are an orthonormal basis for $\mathbb{R}^n$

Exercise: **(i)⇔(ii)⇔(iii)⇔(iv)**

2 important orthogonal matrices for computation:

(a) Householder matrices (elementary reflections)

for $w \in \mathbb{R}^n, w \neq 0$ defined by

$$H(w) = I - 2\frac{ww^T}{w^T w} \qquad \in \mathbb{R}^{n \times n}$$

Note

- $w^T w \in \mathbb{R}$: inner product
  $ww^T \in \mathbb{R}^{n \times n}$: outer product
- $H(w) = H(w)^T = H(w)^{-1}$ Exercise: prove this
- Geometrically $H(w)x$ is the reflection of x in the hyperplane $\{y : w^T y = 0\}$

Norms (ways of measuring size)

for vectors:

$$x \in \mathbb{R}^n, \quad \|x\|_2 = \left(\sum_{i=1}^{n} x_i^2 \right)^{\frac{1}{2}} = (x^T x)^{\frac{1}{2}}$$

or more generally $\|x\|_p = \left(\sum_{i=1}^{n} |x_i|^p \right)^{1/p}$

Special/useful cases: $\|x\|_1 = \sum |x_i|, \quad \|x\|_\infty = \max_i |x_i|$

All satisfy the axioms for a norm:

- $\|\alpha x\| = |\alpha| \|x\|, \quad \alpha \in \mathbb{R}, \; x \in \mathbb{R}^n$
- $\|x\| \geq 0$ and $\|x\| = 0 \Leftrightarrow x = 0$
- $\|x + y\| \leq \|x\| + \|y\|$, triangle inequality

Exercise: check

Norms for matrices: operator norm:

$$\|\boldsymbol{A}\|_{\boldsymbol{p}} = \sup_{\boldsymbol{x}\in\mathbb{R}^{\boldsymbol{n}},\boldsymbol{x}\neq\boldsymbol{0}} \frac{\|\boldsymbol{Ax}\|_{\boldsymbol{p}}}{\|\boldsymbol{x}\|_{\boldsymbol{p}}} = \sup_{\boldsymbol{x}\in\mathbb{R}^{\boldsymbol{n}},\|\boldsymbol{x}\|=\boldsymbol{1}} \|\boldsymbol{Ax}\|_{\boldsymbol{p}}$$

satisfies matrix norm axioms:

- $\|\alpha \boldsymbol{A}\| = |\alpha| \|\boldsymbol{A}\|, \quad \alpha \in \mathbb{R},\ \boldsymbol{A} \in \mathbb{R}^{m\times n}$
- $\|\boldsymbol{A}\| \geq 0$ and $\|\boldsymbol{A}\| = 0 \Leftrightarrow \boldsymbol{A} = 0$ ($a_{i,j} = 0$ for all i, j)
- $\|\boldsymbol{A} + \boldsymbol{B}\| \leq \|\boldsymbol{A}\| + \|\boldsymbol{B}\|$, triangle inequality for matrices

Special cases:

$\|\boldsymbol{A}\|_1 = \max_j \sum_i |a_{i,j}|$ (max absolute column sum)

$\|\boldsymbol{A}\|_\infty = \max_i \sum_j |a_{i,j}|$ (max absolute row sum)

Also: Frobenius norm:

$$\|\boldsymbol{A}\|_{\boldsymbol{F}} = \left(\sum_{\boldsymbol{i}} \sum_{\boldsymbol{j}} \boldsymbol{a}_{\boldsymbol{i},\boldsymbol{j}}^{\boldsymbol{2}} \right)^{\frac{\boldsymbol{1}}{\boldsymbol{2}}}$$

satisfies norm axioms.

Fact: because of compactness (finite dimensionality)

$$\exists \boldsymbol{x} \in \mathbb{R}^{\boldsymbol{n}} \text{ with } \|\boldsymbol{x}\| = 1 \text{ and } \|\boldsymbol{A}\| = \|\boldsymbol{A}\boldsymbol{x}\|$$

Orthogonal Matrices and Norms:

If Q, Z orthogonal of appropriate dimensions

$$\|Qx\|_2 = \|x\|_2, \quad \forall x$$

because $\|Qx\|_2^2 = x^T Q^T Q x = x^T x = \|x\|_2^2$

and so also

$$\|QAZ\|_2 = \|A\|_2, \quad \forall A$$

because

$$\frac{\|QAZx\|_2}{\|x\|_2} = \frac{\|AZx\|_2}{\|x\|_2} = \frac{\|A(Zx)\|_2}{\|Zx\|_2} = \frac{\|Ay\|_2}{\|y\|_2}, y = Zx$$

and take supremum over x.

Also $\|QAZ\|_F = \|A\|_F$ (see exercises)

Basic/Important point:

orthogonal matrices do not change ‘size’ (in $\|\cdot\|_2$ and $\|\cdot\|_F$) so have good numerical properties in finite precision arithmetic

The Singular Value Decomposition (SVD)

Theorem: Given $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ there exist orthogonal matrices

$$\boldsymbol{U} = [\boldsymbol{u_1}, \boldsymbol{u_2}, \ldots, \boldsymbol{u_m}] \in \mathbb{R}^{m \times m} \quad (\boldsymbol{u_i} \in \mathbb{R}^m \text{ each } \boldsymbol{i})$$

$$\boldsymbol{V} = [\boldsymbol{v_1}, \boldsymbol{v_2}, \ldots, \boldsymbol{v_n}] \in \mathbb{R}^{n \times n} \quad (\boldsymbol{v_i} \in \mathbb{R}^n \text{ each } \boldsymbol{i})$$

such that

$$\boldsymbol{U^T A V} = \boldsymbol{\Sigma} = \text{diag}(\boldsymbol{\sigma_1}, \boldsymbol{\sigma_2}, \ldots, \boldsymbol{\sigma_p}), \boldsymbol{p} = \mathbf{min}\{\boldsymbol{m}, \boldsymbol{n}\}$$

with $\boldsymbol{\sigma_1} \geq \boldsymbol{\sigma_2} \geq \ldots \geq \boldsymbol{\sigma_p} \geq \mathbf{0}$ (the singular values).

$\{\boldsymbol{u_i}\}$ are the left singular vectors,
$\{\boldsymbol{v_i}\}$ are the right singular vectors

Properties of SVD:
if $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_r > 0, \sigma_{r+1} = \ldots = \sigma_p = 0$, then

- r= rank(A) = dim$\{Ay : y \in \mathbb{R}^n\}$ = dim(range(A))
- $v_{r+1}, \ldots, v_n \in \mathbb{R}^n$ are an orthonormal basis for ker(A) (=Null(A))
- $u_1, \ldots, u_r \in \mathbb{R}^m$ are an orthonormal basis for range(A)
- $\|A\|_2 = \sigma_1$
- $\|A\|_F^2 = \sigma_1^2 + \sigma_2^2 + \ldots + \sigma_r^2$

Proof: Exercise except all basically follow from

any $x \in \mathbb{R}^n$ can be written as $\sum_{i=1}^{n} \alpha_i v_i$ so

$$
\begin{aligned}
Ax &= U\Sigma V^T x = U\Sigma \left(\sum_{i=1}^{n} \alpha_i V^T v_i \right) \\
&= U\Sigma \sum_{i=1}^{n} \begin{pmatrix} 0 \\ \vdots \\ 0 \\ \alpha_i \\ 0 \\ \vdots \\ 0 \end{pmatrix} = U \sum_{i=1}^{n} \begin{pmatrix} 0 \\ \vdots \\ 0 \\ \sigma_i \alpha_i \\ 0 \\ \vdots \\ 0 \end{pmatrix} \\
&= \sum_{i=1}^{n} \alpha_i \sigma_i u_i
\end{aligned}
$$

Also there are ‘nearness’ results like

$$\|A - \sum_{i=1}^{s} \sigma_i u_i v_i^T\|_2 \leq \|A - B\|_2$$

for all B of rank s
lead to data compression.

The Golub-Reinsch bidiagonalisation algorithm is used to compute the SVD: available in matlab (svd), but not covered here

A final comment:
For a symmetric matrix which is positive (semi-)definite, the
SVD is simply diagonalisation:

$\boldsymbol{U = V}$ and $\boldsymbol{\Sigma}$ is the diagonal matrix of the (non-negative) eigenvalues:

$$\boldsymbol{A = U\Sigma U^T} \quad \text{or} \quad \boldsymbol{AU = U\Sigma}$$

That is, the columns, $\boldsymbol{u_i}$ of $\boldsymbol{U}$ are the eigenvectors of $\boldsymbol{A}$ and
$\boldsymbol{Au_i = \sigma_i u_i = \lambda_i u_i}$

For a symmetrix and indefinite matrix, the singular values are the absolute values of the eigenvalues.

QR Factorisation

Lemma: Given any two vectors $u, v \in \mathbb{R}^n$ with $\|u\|_2 = \|v\|_2$ $\exists w \in \mathbb{R}^n$ s.t. $H(w)u = v$.

Proof: take $w = r(u - v)$, any $r \in \mathbb{R} \setminus \{0\}$, so

$$\begin{aligned} w^T w &= r^2(u^T u - 2v^T u + v^T v) \\ &= 2r^2(u^T u - v^T u) \qquad \text{as } \|u\| = \|v\| \\ &= 2r^2(u - v)^T u = 2rw^T u \end{aligned}$$

so $w^T u = (1/2r)w^T w$. Thus

$$\left(I - \frac{2}{w^T w} w w^T\right) u = u - \frac{2}{w^T w}\frac{w^T w}{2r}\ r(u - v) = v \quad \square$$

In particular

$$v = (\|u\|_2, 0, \ldots, 0) = H\left(r\left[u_1 - \|u\|_2, u_2, \ldots, u_n\right]\right) u.$$

Can be applied to matrices:

if u is 1^{st} column of A: write $H(w) = H_1, \alpha = \alpha_1$

$$H_1 A = \left[\begin{array}{c|c} \alpha_1 & \times \dots \times \\ \hline 0 & \\ \vdots & B \\ 0 & \end{array}\right] \text{ and if } H(\hat{w})B = \left[\begin{array}{c|c} \alpha_2 & \times \dots \times \\ \hline 0 & \\ \vdots & C \\ 0 & \end{array}\right]$$

then

$$H_2 = \begin{bmatrix} 1 & 0 \\ 0 & H(\hat{w}) \end{bmatrix} = H\left([0, \hat{w}]\right)$$

satisfies

$$H_2 H_1 A = \left[\begin{array}{c|c|c} \alpha_1 & \times & \times \dots \times \\ \hline 0 & \alpha_2 & \times \dots \times \\ \hline 0 & 0 & \\ \vdots & \vdots & C \\ 0 & 0 & \end{array}\right]$$

continuing inductively for n steps if $m > n$ gives

$$\boldsymbol{H_n \dots H_2 H_1 A} = \begin{bmatrix} \boldsymbol{\alpha_1} & & \boldsymbol{*} \\ & \boldsymbol{\ddots} & \\ \boldsymbol{0} & & \boldsymbol{\alpha_n} \end{bmatrix} = \boldsymbol{R} \in \mathbb{R}^{m \times n}$$

or for $m - 1$ steps if $m \leq n$ gives

$$\boldsymbol{H_{m-1} \dots H_2 H_1 A} = \begin{bmatrix} \boldsymbol{\alpha_1} & & \boldsymbol{*} \\ & \boldsymbol{\ddots} & \\ \boldsymbol{0} & & \boldsymbol{\alpha_m} \end{bmatrix} = \boldsymbol{R} \in \mathbb{R}^{m \times n}$$

Writing

$\boldsymbol{Q = (H_n \dots H_2 H_1)^{-1} = H_1^T H_2^T \dots H_n^T = H_1 H_2 \dots H_n}$

as Householder matrices are symmetric gives

Theorem: Given any $\boldsymbol{A} \in \mathbb{R}^{m \times n}$, $\exists$ an orthogonal matrix $\boldsymbol{Q} \in \mathbb{R}^{m \times m}$ and an upper triangular matrix $\boldsymbol{R} \in \mathbb{R}^{m \times n}$ s.t. $\boldsymbol{A = QR}$

Proof: Just take $\boldsymbol{H_i = I}$ if i^{th} column is already zero below diagonal and the above procedure can not break down

Remark: If $A = \begin{bmatrix} | & | & & | \\ a_1 & a_2 & \dots & a_n \\ | & | & & | \end{bmatrix}$

and $A = QR$

then $Q = \begin{bmatrix} | & | & & | \\ q_1 & q_2 & \dots & q_n \\ | & | & & | \end{bmatrix}$

and $\{q_1, q_2, \dots, q_n\}$ is an orthonormal basis for span$\{a_1, a_2, \dots, a_n\}$ if this set is linearly independent.

$\Rightarrow QR$ factorization essentially same as Gramm-Schmidt.

Exercise: What happens to QR if $\{a_1, \dots, a_{n-1}\}$ is linearly independent with $a_n \in$ span$\{a_1, \dots, a_{n-1}\}$?

Example: given data $y_1, \ldots, y_m$ at points $x_1, \ldots, x_m$
find parameters in a model e.g. linear model $y = ax + b$
(parameters a, b)
such that

$\sum [y_i - (ax_i + b)]^2$ is min (regression)

same as

$$\min_{a,b} \left\| \begin{bmatrix} x_1 & 1 \\ x_2 & 1 \\ \vdots & \\ x_m & 1 \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} - \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix} \right\|_2$$

QR Algorithm for eigenvalues (eig in matlab)

Set $A = A_1$

for $k = 1, 2, \ldots$ $\begin{cases} \text{factor} & A_k = Q_k R_k \quad \text{(QR factorisat} \\ \text{set} & A_{k+1} = R_k Q_k \quad \text{(matrix multip} \end{cases}$

Lemma: $\{A_k\}$ are all similar matrices and so have same eigenvalues

Proof: $A_{k+1} = R_k Q_k = Q_k^T Q_k R_k Q_k = Q_k^{-1} (A_k) Q_k$ □

Fact: $A_k \to$ upper triangular matrix as $k \to \infty$.

So for large k, $diag(A_k)$ are good approximations to the eigenvalues.

When complex conjugate eigenvalues arise, 2×2 real diagonal blocks remain in A_k which each have a pair of the complex conjugate eigenvalues.

Notes:

1. Speed of convergence depends on the size of gaps between the eigenvalues: more well separated $\Rightarrow$ faster convergence.
2. Convergence is accelerated by the use of shifts (see problem sheet)
3. An orthogonal similarity reduction to Hessenberg form is always employed as a 1^{st} step before apply the QR algorithm with shifts.

Reduction to Hessenberg form

Want a similarity transform $H_p \dots H_2 H_1 A H_1 H_2 \dots H_p$ where H_i are Householder matrices but if

$$H_1 A = \left[\begin{array}{c|c} \times & \times \cdots \times \\ \hline 0 & \times \cdots \times \\ \vdots & \vdots \quad \vdots \\ 0 & \times \cdots \times \end{array}\right]$$

then $H_1 A H_1$ is full i.e. postmultiplication destroys zeros created (otherwise a direct method for eigenvalues!)

So instead let

$$H_1 = \begin{array}{c} \left[\begin{array}{c|c} 1 & 0\cdots 0 \\ \hline 0 & \\ \vdots & K_1 \\ 0 & \end{array}\right] \\ \begin{array}{cc} 1 & n-1 \end{array} \end{array} \begin{array}{c} 1 \\ \\ n-1 \end{array} \quad , \quad A = \left[\begin{array}{c|c} a_1 & v_1^T \\ \hline u_1 & B_1 \end{array}\right] \quad \text{then}$$

$$H_1 A_1 H_1 =$$

$$\left[\begin{array}{cc} a_1 & v_1^T \\ K_1 u_1 & K_1 B_1 \end{array}\right] \left[\begin{array}{cc} 1 & 0 \\ 0 & K_1 \end{array}\right] = \left[\begin{array}{cc} a_1 & v_1^T K_1 \\ K_1 u_1 & K_1 B_1 K_1 \end{array}\right]$$

and choosing K_1 to be a Householder matrix satisfying $K_1 u_1 = (\alpha_1, 0, \ldots, 0)^T$ we have

$$A^{(2)} = H_1 A H_1 = \begin{array}{c} \left[\begin{array}{ccc} a_1 & b_1 & \hat{v}_2^T \\ \alpha_1 & a_2 & v_2^T \\ O & u_2 & B_2 \end{array}\right] \\ \begin{array}{ccc} 1 & 1 & n-2 \end{array} \end{array} \begin{array}{c} 1 \\ 1 \\ n-2 \end{array}$$

inductively (similar to before)

$$A^{(n-1)} = H_{n-2} \dots H_2 H_1 A H_1 H_2 \dots H_{n-2}$$

is in Hessenberg form

QR factorization can now be achieved by $n-1$ Givens rotation matrices (see problem sheet) and the Hessenberg form is preserved by the QR algorithm ie. all of the A_k's are upper Hessenberg.

Direct Methods for linear systems $Ax = b$

basic point: easy to solve triangular systems

$$\begin{bmatrix} \ddots & & & \\ & \times & \times & \times \\ & 0 & \times & \times \\ & 0 & 0 & \times \end{bmatrix}$$

etc.

$a_{n-1,n-1}x_{n-1} = b_{n-1} - a_{n-1,n}x_n$

$\leftarrow$ solve $a_{n,n}x_n = b_n$ then $\nwarrow$

back substitution: takes $\sim n^2$ operations. Need $a_{ii} \neq 0$.

Similar lower triangular (1^{st} equation, then 2^{nd} etc):
forward substitution.

So could solve $Ax = b$ by

$A = QR$ and $\begin{cases} Qy = b & \Rightarrow y = Q^T b \\ Rx = y & \text{back subs. as} R \text{ upper triangu} \end{cases}$

But $\frac{1}{2}$ the number of operations (and other advantages e.g. for sparse) to perform LU factorisation: based on Gauss elimination (successively create zeros below diagonal by following algorithm)

Gauss Elimination:

for columns $j = 1, \ldots, n-1$
  for rows $i = j+1, \ldots, n$
    calculate multiplier $l_{ij} = (a_{ij}/a_{jj})$, (a_{jj} is the pivo
    row $i \longleftarrow$ row $i - l_{ij} *$ row j $(\star)$
  end i
end j

$(\star)$ for $k = j+1, \ldots, n$
  $a_{ik} \longleftarrow a_{ik} - l_{ij} a_{jk}$
  end k
  $b_i \longleftarrow b_i - l_{ij} b_j$

reduces to upper triangular matrix U without changing solution in $\sim \frac{2}{3} n^3$ operations.

Back substitution $\rightarrow$ solution

If store multiplier l_{ij} used to zero a_{ij} as i, j entry of a unit lower triangular matrix L then

$A = LU$ with $\begin{cases} Ly = b & \text{forward subs.} \\ Ux = y & \text{back subs.} \end{cases}$

solves $Ax = b$.

Note: For many b's need only 1 LU factorization.

Recall $a_{ii} \neq 0$ necessary for Gauss Elimination so fails on e.g. $\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$ which is non-singular.

Pivoting:

Row interchanges: often expressed as $PA = LU$, P permutation.

Partial pivoting: when zeroing subdiagonal of p^{th} column

find max $|a_{ip}| = |m|$, $i = p, p+1, \ldots, n$;

m becomes pivot

swap row p with row which gives this max.

Fails if and only if A singular as
$a_{pp} = 0, m = 0 \Rightarrow$ det $A = 0$

Special forms

- A Symmetric positive definite: $A = LL^T$, L lower triangular, Cholesky factorisation.
- A Symmetric Indefinite: $A = LDL^T$, L lower triangular, D block diagonal, 1×1 and 2×2 blocks: Bunch - Parlett, Bunch - Kaufmann factorizations.
- A Banded: eliminate only in band, $\sim \frac{1}{3}nb^2$ operations for LU
  (NB pivoting generally destroys bandedness)
- A Sparse: good software e.g. HSL or $\backslash$ for sparse in matlab.

Ill-conditioning

Proposition: If $Ax = b \quad (1)$ and $A(x + \delta x) = b + \delta b \quad (2)$ then

$$\frac{\|\delta x\|}{\|x\|} \leq \|A\|\|A^{-1}\|\frac{\|\delta b\|}{\|b\|}$$

Proof: $A^{-1}((2) - (1)) \Rightarrow \delta x = A^{-1}\delta b$

$$\begin{aligned}
\text{so } \|\delta x\| = \|A^{-1}\delta b\| &\leq \|A^{-1}\|\|\delta b\| \\
\text{also } \|b\| = \|Ax\| &\leq \|A\|\|x\| \\
\text{or } \frac{1}{\|x\|} &\leq \frac{\|A\|}{\|b\|} \\
\text{so } \frac{\|\delta x\|}{\|x\|} &\leq \|A\|\|A^{-1}\|\frac{\|\delta b\|}{\|b\|} \\
&\uparrow \qquad\qquad \uparrow \qquad\quad \uparrow
\end{aligned}$$

Also if A is perturbated to $A + \delta A$ then

$$\frac{\|\delta x\|}{\|x + \delta x\|} \leq \|A\|\|A^{-1}\|\frac{\|\delta A\|}{\|A\|}$$

(Exercise: Show this)
These results identify $\kappa = \|A\|\|A^{-1}\|$ (the 'condition number' for solution of linear systems) as a measure of ill-conditioning.

Usually necessary if large κ to reformulate problem because:

Gauss elimination finds $\tilde{x}$ such that $r = b - A\tilde{x}$ is small (not exactly x s.t. $Ax = b$) on a computer.

For many A, r small $\Rightarrow e = x - \tilde{x}$ is small but not when κ is large as indicated by the above results.

Example: Interpolation: Given N and data $f(x_i)$ at distinct points $x_i, i = 0, 1, \ldots, N$, find polynomial $p(x) = \sum_{k=0}^{n} a_k x^k \quad \in \Pi_n$ such that

$$p(x_i) = f(x_i).$$

This can be written as: solve

$$\begin{bmatrix} 1 & x_0 & x_0^2 & \cdots & x_0^n \\ 1 & x_1 & x_1^2 & \cdots & x_1^n \\ 1 & x_2 & x_2^2 & \cdots & x_2^n \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_n & x_n^2 & \cdots & x_n^n \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \\ a_n \end{bmatrix} = \begin{bmatrix} f(x_0) \\ f(x_1) \\ f(x_2) \\ \vdots \\ f(x_n) \end{bmatrix}$$

For $x_k = k + 1$,

		expected accuracy
$n = 4$	$\kappa = 2 \cdot 6 \times 10^{4}$	12 decimal places
$n = 8$	$\kappa = 4 \cdot 2 \times 10^{10}$	6 decimal places
$n = 12$	$\kappa = 4 \cdot 2 \times 10^{17}$	0 decimal places
$n = 16$	$\kappa = 1 \cdot 9 \times 10^{25}$	no hope of accurate solution

but can reformulate the interpolation problem in many ways
e.g. use a better basis for Π_N than $\{1, x, x^2, \ldots, x^N\}$.
In fact for this problem there are reliable and faster ($O(N^2)$)
methods (GVL p183 Vandermonde)

Iterative solution methods for $Ax = b$

idea: split $A = M - N$, so easy to solve systems with M, then iterate:

Guess $x^{(0)}$

solve $Mx^{(k)} = Nx^{(k-1)} + b$ for $k = 1, 2, ..$

basic point: if $\{x^{(k)}\}$ converges (to x, say) then

$$Mx = Nx + b, \quad \text{ie. } Ax = b$$

ie. it converges to the solution.

Iterative solution methods for $Ax = b$, $A \in \mathbb{R}^{n \times n}$

idea: split $A = M - N$, so easy to solve systems with M, then iterate:

Guess $x^{(0)}$

solve $Mx^{(k)} = Nx^{(k-1)} + b$ for $k = 1, 2, ..$

basic point: if $\{x^{(k)}\}$ converges (to x, say) then

$$Mx = Nx + b, \quad \text{ie. } Ax = b$$

ie. it converges to the solution.

Jacobi's method: $M = \text{diag}(A), \quad (N = A - M)$

In practice: componentwise

for iterates $k = 1, 2, \ldots$

    for rows (equations) $i = 1, \ldots, n$

$$x_i^{(k)} = \frac{1}{a_{i,i}} \left(- \sum_{j=1, j \neq i}^{n} a_{i,j} x_i^{(k-1)} + b_i \right)$$

    endo

endo

better ? use most recently updated value of x_i

for iterates $k = 1, 2, \ldots$
  for rows $i = 1, \ldots, n$

$$x_i^{(k)} = \frac{1}{a_{i,i}} \left(-\sum_{j=1}^{i-1} a_{i,j} x_j^{(k)} - \sum_{j=i+1}^{n} a_{i,j} x_j^{(k-1)} + b_i \right)$$

  endo
endo

This is Gauss-Seidel iteration: rearranging

$$\sum_{j=1}^{i} a_{ij} x_j^{(k)} = -\sum_{j=i+1}^{n} a_{ij} x_j^{(k-1)} + b_i$$

which is $(L + D)x^{(k)} = -Ux^{(k-1)} + b$

when $D = \text{diag}(A)$, $L =$ strict lower triangular of A, $U =$ strict upper triangular of A

i.e. Solve $Mx^{(k)} = Nx^{(k-1)} + b$, $M = L + D$ is

better still ? take Gauss-Seidel $x^{(k)}$ iterate and average with $x^{(k-1)}$

$$x_i^{(k)} = \omega \underbrace{\left(b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(k)} - \sum_{j=i+1}^{n} a_{ij} x_j^{(k-1)} \right) \frac{1}{a_{ii}}}_{\text{exactly as in Gauss-Seidel}} + (1-\omega) x_i^{(k}$$

$\omega \in \mathbb{R}$ relaxation parameter

$\omega < 1$	gives underelaxation	cautious & slow
$\omega = 1$	Gauss-Seidel	
$\omega > 1$	overelaxation	$\Rightarrow$ Successive Overelaxa Method (SOR)

By rearranging as for Gauss-Seidel: matrix form

$$(\boldsymbol{D} + \omega \boldsymbol{L}) \boldsymbol{x}^{(k)} = \omega \boldsymbol{b} + [(1-\omega)\boldsymbol{D} - \omega \boldsymbol{U}]\, \boldsymbol{x}^{(k-1)}$$

Symmetry sometimes useful to preserve, so if A symmetric ($\Leftrightarrow U = L^T$):
Symmetric SOR (SSOR)

$$\begin{aligned}(D + \omega L)x^{(k-\frac{1}{2})} &= \omega b + [(1-\omega)D - \omega U]\, x^{(k-1)} \\ (D + \omega U)x^{(k)} &= \omega b + [(1-\omega)D - \omega L]\, x^{(k-\frac{1}{2})}\end{aligned}$$

corresponds to $M = (D + \omega L)D^{-1}(D + \omega U)$ which is symmetric.

Important point: if A sparse then these methods only need use the non-zero entries of A e.g.

$$\sum_{j=1}^{i-1} a_{ij}x_j^{(k)} \text{ becomes } \sum_{\{j<i : a_{ij}\neq 0\}} a_{ij}x_j^{(k)}$$

Convergence of simple iterations:

$$
\begin{aligned}
M\boldsymbol{x}^{(k)} &= N\boldsymbol{x}^{(k-1)} + \boldsymbol{b} \\
\text{and } A\boldsymbol{x} = \boldsymbol{b} \Rightarrow \quad M\boldsymbol{x} &= N\boldsymbol{x} + \boldsymbol{b} \qquad\qquad (A = M - N) \\
\text{so } M(\boldsymbol{x} - \boldsymbol{x}^{(k)}) &= N(\boldsymbol{x} - \boldsymbol{x}^{(k-1)}) \\
\boldsymbol{x} - \boldsymbol{x}^{(k)} &= M^{-1}N(\boldsymbol{x} - \boldsymbol{x}^{(k-1)}) \\
&= (M^{-1}N)^{k}(\boldsymbol{x} - \boldsymbol{x}^{(0)})
\end{aligned}
$$

$M^{-1}N$ is called the iteration matrix

So $\|\boldsymbol{x} - \boldsymbol{x}^{(k)}\| \to 0$ at least if

$$
\|(M^{-1}N)^{k}\|\|\boldsymbol{x} - \boldsymbol{x}^{(0)}\| \leq \underbrace{\|M^{-1}N\|^{k}}_{\uparrow} \quad \underbrace{\|\boldsymbol{x} - \boldsymbol{x}^{(0)}\|}_{\text{unknown error in initial gue}}
$$

$$
\to 0 \text{ if } \|M^{-1}N\| < 1
$$

this is a sufficient condition for convergence.

Notation

$\rho(A) =$ max $\{|\lambda| : \lambda$ an eigenvalue of $A\}$

the spectral radius

Theorem

If $M^{-1}N$ is diagonalisable, then $||x - x^{(k)}|| \to 0$ as $k \to \infty$ for any initial guess $x^{(0)}$ if and only if $\rho(M^{-1}N) < 1$

not diagonalisable, it is triangularisable

i.e. $\exists$ triangular matrix $\boldsymbol{T}$ with $\boldsymbol{M^{-1}N = QTQ^T}$ for some orthogonal matrix $\boldsymbol{Q}$ (Schur decomposition).

More useful for our purpose here (only!) is the existence of a *Jordan canonical form*:

$$\boldsymbol{M^{-1}N = XJX^{-1}}, \boldsymbol{J} = \begin{bmatrix} \boldsymbol{J_1} & & \boldsymbol{O} \\ & \ddots & \\ \boldsymbol{O} & & \boldsymbol{J_p} \end{bmatrix}$$

where $\boldsymbol{J_i} \in \mathbb{R}^{\boldsymbol{n_i} \times \boldsymbol{n_i}}$ and $\sum_{\boldsymbol{i=1,\ldots,p}} \boldsymbol{n_i} = \boldsymbol{n}$ with

$$\boldsymbol{J_i} = \begin{bmatrix} \boldsymbol{\lambda_i} & \boldsymbol{1} & & \\ & \ddots & \ddots & \\ & & \ddots & \boldsymbol{1} \\ & & & \boldsymbol{\lambda_i} \end{bmatrix}$$

Thus $(M^{-1}N)^k = (XJX^{-1})^k = XJ^kX^{-1}$ and, as $k \to \infty$

$$(M^{-1}N)^k \to 0 \Leftrightarrow J^k \to 0 \Leftrightarrow J_i^k \to 0 \text{ all } i.$$

That is, we obtain convergence if and only if for every Jordan block, its powers tend to zero as $k \to \infty$.

First consider if $\lambda_i = 0$, then write $J_i = \widehat{J} \in \mathbb{R}^{\widehat{n} \times \widehat{n}}$ and

$$\widehat{J} = \begin{bmatrix} 0 & 1 & & & & \\ & 0 & 1 & & & \\ & & \ddots & \ddots & & \\ & & & \ddots & \ddots & \\ & & & & 0 & 1 \\ & & & & & 0 \end{bmatrix}, \widehat{J^2} = \begin{bmatrix} 0 & 0 & 1 & & & \\ & 0 & 0 & 1 & & \\ & & & \ddots & \ddots & \\ & & & & 0 & 0 \\ & & & & & 0 \\ & & & & & \end{bmatrix}$$

and generally the diagonal of 1's moves up toward the top right for each succesive power. Thus $\widehat{J^n} = 0$

Now consider when $\lambda_i \neq 0$. we have

$$
\begin{aligned}
J_i^k &= \left(\lambda_i I + \widehat{J}\right)^k \\
&= \sum_{r=0}^{k} \binom{k}{r} \widehat{J}^r \, \lambda_i^{k-r} \quad \text{since } I, \widehat{J} \text{ commute} \\
&= \sum_{r=0}^{n_i} \binom{k}{r} \widehat{J}^r \, \lambda_i^{k-r} \\
&\to 0 \quad \text{as } k \to \infty \text{ since } \lambda_i^{k-r} \to 0 \\
&\qquad \text{if and only if } |\lambda_i| < 1 \text{ each } i.
\end{aligned}
$$

Thus $(M^{-1}N)^k \to 0$ as $k \to \infty \Leftrightarrow |\lambda_i| < 1$ each i, hence convergence since $x - x^{(k)} = (M^{-1}N)^k (x - x^{(0)})$.

Note the powers of J can grow considerably before eventual convergence.

Practical Example

$$4u_{j,k} - u_{j+1,k} - u_{j-1,k} - u_{j,k+1} - u_{j,k-1} = h^2 f_{j,k}$$

for $j, k = 1, \ldots, n$ with $u_{0,k}, u_{n+1,k}, u_{j,0}, u_{j,n+1}$ given.

$$4u_{j,k} - u_{j+1,k} - u_{j-1,k} - u_{j,k+1} - u_{j,k-1} = h^2 f_{j,k}$$

for $j, k = 1, \ldots, n$ with $u_{0,k}, u_{n+1,k}, u_{j,0}, u_{j,n+1}$ given.

eg.Jacobi iteration for this problem is: guess $u^{(0)}$

for iterates $i = 1, 2, \ldots$
  for $j = 1, \ldots, n$
    for $k = 1, \ldots, n$

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

    endo
  endo
endo

Proposition for $r, s = 1, \ldots, n$

$$\lambda^{r,s} = \frac{1}{2}(\cos r\pi h + \cos s\pi h)$$

is an eigenvalue of the Jacobi iteration matrix for A with eigenvector $\underline{v}^{r,s}$ having entries

$$v_{jk}^{r,s} = \sin rj\pi h \ \sin sk\pi h$$

Proof direct calculation: see exercises

Remark shows Jacobi iteration converges as $-1 < \lambda^{r,s} < 1$ for each r, s but

$$\rho(\text{Jacobi}) = \frac{1}{2}(\cos \pi h + \cos \pi h) = \cos \pi h = 1 - \frac{\pi^2 h^2}{2} + O($$

so very close to 1 for small $h \Rightarrow$ slow convergence.

But recalling (in this notation)

$$\underline{u} - \underline{u}^{(i)} = (M^{-1}N)^i(\underline{u} - \underline{u}^{(0)}) = \sum_{r,s=1}^{n} \alpha_{r,s}(\lambda^{r,s})^i \underline{v}^{r,s}$$

we see that $\lambda^{r,s}$ small $\Rightarrow$ error component in $\underline{v}^{r,s}$ reduces very quickly to zero, so $\underline{u} - \underline{u}^{(i)}$ is quickly dominated by components $\underline{v}^{r,s}$ for which $|\lambda^{r,s}| \simeq 1$.

More interesting for our purpose: relaxed Jacobi: for $\theta Ax = \theta b$, $\theta \in \mathbb{R}^+$, $M = D$, $N = (1-\theta)D - \theta(L+U)$

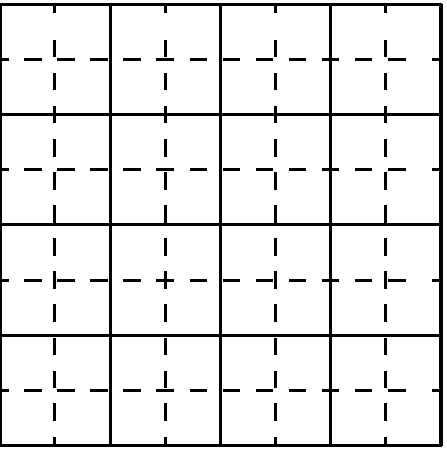
$$\Rightarrow M^{-1}N = (1-\theta)I - \theta D^{-1}(L+U)$$

has eigenvalues $1 - \theta + \frac{\theta}{2}(\cos r\pi h + \cos s\pi h)$

so e.g. for $\theta = \frac{1}{2}$, eigenvalues
$\frac{1}{2} + \frac{1}{4}(\cos r\pi h + \cos s\pi h) \in (0,1)$

AND high frequency eigenvectors (r, s large) correspond to

Idea of a ‘smoother’ leads to Multigrid



guess u^0

on fine grid smooth (i.e. for 3 relaxed Jacobi iteration)

$u^0 \longrightarrow u^s$

$u - u^s = e^s$ smoother than $u - u^0 = e^0$

$$Ae^s = Au - Au^s = b - Au^s = r^s \quad \text{(residual)}$$

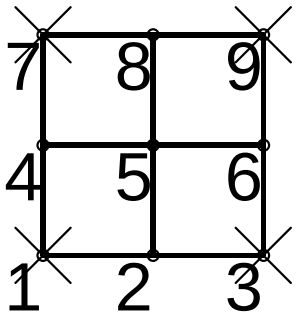
same as original problem, but e^s smoother $\Rightarrow$ solve on coarser grid i.e. use a coarse grid representation $\overline{A}$ of A and solve $\overline{A}\overline{e}^s = \overline{r}^s$ where $\overline{e}^s, \overline{r}^s$ are coarse grid

So need grid transfer operators:

Restriction: fine $\longrightarrow$ coarse

Prolongation: coarse $\longrightarrow$ fine

Prolongation:



$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ 0 & 1 & 0 & 0 \\ \frac{1}{2} & 0 & \frac{1}{2} & 0 \\ \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} \\ 0 & \frac{1}{2} & 0 & \frac{1}{2} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & \frac{1}{2} & \frac{1}{2} \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_3 \\ x_7 \\ x_9 \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \\ x_9 \end{bmatrix}$$

$\uparrow P$

Restriction:

normally $\boldsymbol{R = \alpha P^T}$ where $\boldsymbol{\alpha \in \mathbb{R}}$ is such that

$$\boldsymbol{\alpha P^T P \mathbf{e} = \mathbf{R}\mathbf{P}\mathbf{e} = \mathbf{e}}$$

$\mathbf{e}$ being the vector of all ones

eg. $\boldsymbol{\alpha = \frac{4}{9}}$ for the $\boldsymbol{P}$ above.

Basic point:

$$\begin{array}{rrcll} \text{if} & x & = & Px & \overline{x} \text{ 'short'} \\ \text{or} & \overline{x} & = & Rx & x \text{ 'long'} \end{array}$$

then little loss of accuracy if and only if x is a 'smooth vector' i.e. a vector of coefficients representing a non-oscillatory function.

Coarse grid operator: $\overline{A}$: 2 possibilities:

(i) 5 point formula on $2h$ mesh

(ii) $\overline{A} = RAP = \alpha P^T AP$
(Galerkin coarse grid operator)

2-grid algorithm:

Choose u_0

for two − grid iterations $i = 0$ until convergence do

$(\text{pre}-)\text{smooth}: \; u_i \longrightarrow u^s$

calculate residual : $r^s = b - Au^s$

restrict residual : $r^s \longrightarrow \overline{r}^s \qquad (\overline{r}^s = Rr^s)$

solve $\overline{A}\,\overline{e}^s = \overline{r}^s$ to get coarse grid correction

prolong : $\overline{e}^s \longrightarrow e^s \qquad (e^s = P\overline{e}^s)$

update : $u_{i+1} \leftarrow u^s + e^s$

$(\text{sometimes})\ \text{post} - \text{smooth}: \; u_{i+1} \longrightarrow u_{i+1}$

enddo

Note:

$$u_{i+1} \leftarrow u^s + P\overline{A}^{-1}R(b - Au^s)$$

If smoother is based on a splitting $A = M - N$ then the iteration matrix is $M^{-1}N$ and we have eg. for 2 smoothing steps (so $u^s = u^{(2)}$)

$$\begin{aligned} u^{(1)} &= (M^{-1}N)u^{(0)} + M^{-1}b \\ u^{(2)} &= (M^{-1}N)u^{(1)} + M^{-1}b \\ &= (M^{-1}N)^2 u^{(0)} + (I + M^{-1}N)M^{-1}b \qquad (\star) \end{aligned}$$

but also the exact solution satisfies

$$u = (M^{-1}N)u + M^{-1}b$$

$$\Rightarrow u = (M^{-1}N)^2 u + (I + M^{-1}N)M^{-1}b \qquad (+)$$

so $(+) - (\star)$ gives

$$u - u^{(2)} = (M^{-1}N)^2 (u - u^{(0)}).$$

Note also for the residual using $(\star)$ and $(+)$ we have

$$b - Au^{(2)} = A(u - u^{(2)}) = A(M^{-1}N)^2(u - u^{(0)}).$$

So 2-grid iterate is

$$\begin{aligned} u^{(2)} &+ P\overline{A}^{-1}R(b - Au^{(2)}) \\ &= u^{(2)} + P\overline{A}^{-1}R \;\; A\,(M^{-1}N)^2(u - u^{(0)}) \end{aligned}$$

so that the error after a single 2-grid iteration is

$$u - u^{(2)} - P\overline{A}^{-1}R \;\; A\,(M^{-1}N)^2(u - u^{(0)})$$

$$= (A^{-1} - P\overline{A}^{-1}R) \;\; A\,(M^{-1}N)^2(u - u^{(0)})$$

In general the j^{th} 2-grid iteration (for iterate u_j with $u^{(0)} = u_0$) is

$$u_j = \left[(M^{-1}N)^2 u_{j-1} + (I + M^{-1}N)M^{-1}b\right] +$$
$$P\overline{A}^{-1}R\left(b - A\left[(M^{-1}N)^2 u_{j-1} + (I + M^{-1}N)M^{-1}\right.\right.$$

and the error $e_j = u - u_j$ therefore satisfies

$$e_j = (A^{-1} - P\overline{A}^{-1}R)\ \ A\,(M^{-1}N)^2 e_{j-1}$$

or in general if ν pre-smoothing steps and μ post-smooting steps are used

$$e_j = (M^{-1}N)^{\mu}(A^{-1} - P\overline{A}^{-1}R)\ \ A\,(M^{-1}N)^{\nu} e_{j-1}$$

$$e_j = (M^{-1}N)^{\mu}(A^{-1} - P\overline{A}^{-1}R) \;\; A \; (M^{-1}N)^{\nu} e_{j-1}$$

Convergence depends on

- Smoothing (as above)
- Approximation: R and P must sufficiently accurately reproduce smooth vectors and $\overline{A}$ sufficiently accurately represent A

In mathematical terms: we use the regular Euclidean norm $\|\cdot\|$ and $\|\cdot\|_A$ defined by $\|x\|_A^2 = x^T Ax$ which is a norm when A is symmetric and positive definite, and establish

- the Smoothing Property: for all y

$$\|A\ (M^{-1}N)^\nu y\| \leq \eta(\nu)\ \|y\|_A$$

  with $\eta(\nu) \to 0$ as $\nu \to \infty$ being independent of n (n =dimension of A)

- the Approximation Property: for all y, there exists C independent of n with

$$\|(A^{-1} - P\overline{A}^{-1}R)y\|_A \leq C\|y\|.$$

Immediately we have

Theorem: if the Smoothing and Approximation properties hold then 2-grid iteration with no post-smoothing converges at a rate independent of n.

Proof:

$$\begin{aligned}
\|e_j\|_A &= \|(A^{-1} - P\overline{A}^{-1}R)\ A\ (M^{-1}N)^{\nu} e_{j-1}\|_A \\
&\leq C\|A\ (M^{-1}N)^{\nu} e_{j-1}\| \qquad \text{(Approx. Property)} \\
&\leq \underbrace{C\eta(\nu)}_{\to 0 \text{ as } \nu\to\infty} \|e_{j-1}\|_A \qquad \text{(Smoothing Property)}
\end{aligned}$$

so $\exists$ a number of smoothing steps ν independent of n with

$$\|e_j\|_A \leq \gamma \|e_{j-1}\|_A$$

with $\gamma < 1$. $\square$

Approximation Property: rough sketch (depends on finite difference error)

$$A^{-1}b \;\leftrightarrow\; \text{mesh solution on mesh } h \leftrightarrow u_h$$
$$P\overline{A}^{-1}Rb \;\leftrightarrow\; \text{mesh solution on mesh } 2h \leftrightarrow u_{2h}$$

so $\|(A^{-1} - P\overline{A}^{-1}R)b\|_A \sim |u_h - u_{2h}| \sim \|b\|$ any b.

Smoothing Property: we prove only for relaxed Jacobi :

$$M^{-1}N = (1-\theta)I - \theta D^{-1}(L+U) = I - \theta D^{-1}A = I - \frac{\theta}{4}A$$

as $D = 4I$ for 5 point formula.

Theorem: if the eigenvalues of $M^{-1}N$ lie in $[-\sigma, 1]$ with $0 \le \sigma < 1$ being independent of n, then the smoothing property holds.

(Recall: $\theta = \frac{1}{2} \Rightarrow$ eigenvalues of $M^{-1}N \in (0, 1)$.)

Proof: let $\{z_1, \ldots, z_n\}$ be the orthonormal eigenvector basis of $I - (\theta/4)A$ and $y = \sum c_i z_i$,
$(I - (\theta/4)A)z_i = \lambda_i z_i$. Then $Az_i = (4/\theta)(1 - \lambda_i)z_i$ so

$$A(M^{-1}N)^\nu y = (4/\theta) \sum c_i \lambda_i^\nu (1 - \lambda_i) z_i,$$

and $$\|A(M^{-1}N)^\nu y\|^2 = (16/\theta^2) \sum c_i^2 \lambda_i^{2\nu} (1 - \lambda_i)^2$$

as $z_i^T z_j = \delta_{i,j}$. Now $\lambda_i \in [-\sigma, 1] \Rightarrow \lambda_i^{2\nu}(1 - \lambda_i)$ is maximal either at the stationary point $\lambda_i = 2\nu/(2\nu + 1)$ or when $\lambda_i = -\sigma$ so that

$$\max_{\lambda_i \in [-\sigma, 1]} \lambda_i^{2\nu}(1 - \lambda_i) \leq \max \left\{ \frac{1}{2\nu} \frac{1}{e}, \sigma^{2\nu}(1 + \sigma) \right\}$$

since $$\left(\frac{2\nu}{2\nu + 1} \right)^{2\nu} \frac{1}{2\nu + 1} = \frac{1}{2\nu} \frac{1}{(1 + 1/2\nu)^{2\nu+1}} \leq \frac{1}{2\nu} \frac{1}{e}$$

and $(1 + 1/2\nu)^{2\nu+1} \searrow e (= 2.718\ldots)$ as $\nu \to \infty$.

Thus

$$\|A(M^{-1}N)^{\nu}y\|^2$$

$$\begin{aligned} &\leq \max\left\{\frac{4}{e\theta}\frac{1}{2\nu}, \frac{4}{\theta}\sigma^{2\nu}(1+\sigma)\right\}\sum c_i^2\frac{4}{\theta}(1-\lambda_i) \\ &= \underbrace{\max\left\{\frac{4}{e\theta}\frac{1}{2\nu}, \frac{4}{\theta}\sigma^{2\nu}(1+\sigma)\right\}}_{\eta(\nu)\to 0 \text{ as } \nu\to\infty}\|y\|_A^2 \end{aligned}$$

Notes:

- Can be extended to Multigrid by replacing the coarse grid solve $\overline{A}\,\overline{e}^s = \overline{r}^s$ recursively by a 2-grid iteration: just apply Gauss Elimination when very small dimensional coarse space. n-independent convergence is preserved.
- Other smoothers,prolongation and restriction operators and more general problems can be analysed in a similar way.
- Work per iteration depends linearly on problem size (n^2). Number of iterations for convergence independent of $n \Rightarrow$ optimal solver (ie. $O(N)$ work to solve an $N \times N$ linear system.
  (cf. $O(N^3)$ for Gauss Elimination).

Polynomial Iterative Methods

Simple iteration

$$
\begin{aligned}
x^{(k)} &= (M^{-1}N)x^{(k-1)} + \hat{b} \ , \ \ \hat{b} = M^{-1}b \\
\text{and} \quad x &= (M^{-1}N)x + \hat{b} \\
\Rightarrow x - x^{(k)} &= (M^{-1}N)(x - x^{(k-1)}) \\
\Rightarrow x - x^{(k)} &= (M^{-1}N)^k(x - x^{(0)}) = S^k(x - x^{(0)})
\end{aligned}
$$

if $S = M^{-1}N$. ie.

$$x - x^{(k)} = p_k(S)(x - x^{(0)}) = S^k(x - x^{(0)}) \ \ , \ \ p_k(z) = z^$$

Now if

$$x - x^{(0)} = \sum_{i=1}^{n} \alpha_i v_i, \quad Sv_i = \lambda_i v_i$$

$$x - y^{(k)} = \sum_{i=1}^{n} \alpha_i p_k(S) v_i = \sum_{i=1}^{n} \alpha_i p_k(\lambda_i) v_i$$

Idea: $x - y^{(k)}$ should be small if $p_k(\lambda_i)$ is small

If S is symmetric we can say more
(since S orthogonaly diagonalisable)

$$\Rightarrow S = U\Lambda U^T \,,\; UU^T = I$$

$\uparrow$ diag Matrix of eigenvalues

$$\Rightarrow S^2 = U\Lambda U^T U\Lambda U^T = U\Lambda^2 U^T, \ldots, S^k = U\Lambda^k U^T$$

$$\Rightarrow p(S) = Up(\Lambda)U^T \qquad \text{any polynomial } p$$

$$\Rightarrow \|p(S)\|_2 = \|Up(\Lambda)U^T\|_2 = \|p(\Lambda)\|_2$$

$$= \left\| \begin{bmatrix} p(\lambda_1) & & & O \\ & p(\lambda_2) & & \\ & & \ddots & \\ O & & & p(\lambda_n) \end{bmatrix} \right\|_2$$

$$= \max |p(\lambda_i)|$$

So for polynomial iteration $x - y^{(k)} = p_k(S)(x - x^0)$

$$\|x-y^{(k)}\|_2 \leq \|p_k(S)\|_2\|x-x^{(0)}\|_2 = \max_i |p_k(\lambda_i)| \ \|x-x^{(0}$$

So desire $p_k \in \Pi_k$ is small at eigenvalues with $p_k(1) = 1$.

Notes:

- if $\lambda_i = 1$ for some i
  $\Rightarrow Sv_i = v_i \Leftrightarrow Mv_i = Nv_i \Leftrightarrow Av_i = 0$ ie. A singular
- If only k distinct eigenvalues of S: choose p_k to have these as roots.

  For such a p_k, $\|x - y^{(k)}\|_2 = 0$ i.e. termination after k steps!

Candidates for $\{p_k\}$? : Chebyshev polynomials

Suppose $\lambda_i(S) \in [a,b]$, $1 \notin [a,b]$ then

$$\begin{aligned} \|x - y^{(k)}\|_2 &\leq \max_i |p_k(\lambda_i)| \|x - x^{(0)}\|_2 \\ &\leq \max_{t \in [a,b]} |p_k(t)| \|x - x^{(0)}\|_2 \end{aligned}$$

and the polynomials which

$$\underset{p \in \Pi_k,\ p(1)=1}{\text{minimise}} \quad \max_{t \in [a,b]} |p(t)|$$

are shifted and scaled Chebyshev polynomials

Chebyshev polynomials are defined on $[-1, 1]$ by $T_0(t) =$
and for $m = 1, 2, \ldots$ by

$$T_m(t) = \begin{cases} \frac{1}{2^{m-1}} \cos m\theta & (0 \le \theta \le \pi) \\ \quad \text{where } t = \cos\theta & -1 \le t \le 1 \\ \\ \frac{1}{2^{m-1}} \cosh m\theta & \\ \quad \text{where } t = \cosh\theta & t \ge 1 \\ \\ (-1)^m T_m(-t) & t \le -1 \end{cases}$$

$$T_0 = 1, \quad T_1 = t,$$

$$T_2 = \frac{1}{2} \cos\, 2\theta = \frac{1}{2}(2\cos^2\theta - 1) = t^2 - \frac{1}{2}, \quad \ldots$$

In general since

$$\begin{aligned}\cos(m+1)\,\theta &= \cos m\theta\cos\theta - \sin m\theta\sin\theta \\ \cos(m-1)\,\theta &= \cos m\theta\cos\theta + \sin m\theta\sin\theta\end{aligned}$$

we have

$$\cos(m+1)\,\theta + \cos(m-1)\,\theta = 2\cos m\theta\cos\theta$$

i.e. $2^m T_{m+1}(t) + 2^{m-2}T_{m-1}(t) = 2\cdot 2^{m-1}T_m(t)t$

or

$$T_{m+1}(t) = t\,T_m(t) - \frac{1}{4}T_{m-1}(t)\ ,\ m = 2,3,\ldots \quad (\star)$$

so

$$T_3(t) = t(t^2 - \frac{1}{2}) - \frac{1}{4}t = t^3 - \frac{3}{4}t\ ,\ \text{ etc.}$$

If $\lambda_i(S) \in [a,b], 1 \notin [a,b]$, need to shift using linear map

$$\begin{array}{ccccc} [a,b] & \mapsto & [-1,1] & : & t = \frac{2r-a-b}{b-a} \\ \in & & \in & & \\ r & & t & & \end{array}$$

$$\Rightarrow \hat{T}_k(r) = T_k\left(\frac{2r-a-b}{b-a}\right) \Big/ T_k\left(\frac{2-a-b}{b-a}\right)$$

satisfies $\hat{T}_k(1) = 1$ and minimises

$$\max_{r \in [a,b]} |p(r)|$$

over all polynomials of degree $\leq k$.

Using $p_k = \hat{T}_k$ is called the
Chebyshev semi-iterative method.

Remarks:

1. need estimates $a \lesssim \lambda_{\min}(S)$ and $b \gtrsim \lambda_{\max}(S)$ in order to construct $\hat{T}_k$'s on a reasonable interval
2. can use the 3-term recurrance $(\star)$ to make the algorithm more efficient than using the coefficients $\beta_e^{(k)}$ of the polynomials $T_k(z)$ explicitly (see exercise)
3. Convergence: we have

$$\|x - y^{(k)}\|_2 \leq \max_{r \in [a,b]} |\hat{T}_k(r)| \|x - x^{(0)}\|$$

if $\lambda_i \in [a,b]$, $1 \notin [a,b]$

$$\max_{r \in [a,b]} |\hat{T}_k(r)| = |\hat{T}_k(a)| = |\hat{T}_k(b)|$$

$$= \frac{|T_k(1)|}{|T_k(\frac{2-a-b}{b-a})|}$$

as max error always attained at end points.

Krylov Subspace Methods

$$A \in \mathbb{R}^{n \times n}, r \in \mathbb{R}^n$$

If A sparse or specially structured, so easy to compute

$$Ar, A(Ar), \ldots$$

i.e.

$$r, Ar, A^2 r, \ldots$$

are easy to compute, then Krylov subspaces

$$\mathcal{K}_k(A, r) = \text{ span } \{r, Ar, \ldots, A^{k-1} r\}$$

are convenient nested vector subspaces (exercise: check).

Note $$y \in \mathcal{K}_k(A, r) \Leftrightarrow y = q_{k-1}(A) r_0$$

where $q_{k-1} \in \Pi_{k-1}$ (real polynomials of degree $\leq k - 1$.)

If solving $Ax = b$, (A invertible), guess x_0, $r_0 = b - Ax_0$, then look for

$$x_k \in x_0 + \mathcal{K}_k(A, r_0) \quad , r_k = b - Ax_k,\ k = 1, 2, \ldots$$

$$\begin{aligned} & & x_k &\in x_0 + \mathcal{K}_k(A, r_0) \\ \Leftrightarrow & & x_k &= x_0 + q_{k-1}(A) r_0 \\ \Leftrightarrow & x - x_k &= x - x_0 - q_{k-1}(A) r_0 \\ \Leftrightarrow & \underbrace{A(x - x_k)}_{\substack{b - Ax_k \\ \| \\ r_k}} &= \underbrace{A(x - x_0)}_{r_0} - A\, q_{k-1}(A) r_0 \end{aligned}$$

i.e. $\quad r_k = p_k(A) r_0 \quad , p_k \in \Pi_k \ , p_k(0) = 1$

$\Leftrightarrow A^{-1} r_k = p_k(A) A^{-1} r_0$, i.e. $e_k = p_k(A) e_0$, $e_k = x - x_k$

Krylov Subspace Methods

Most common Krylov subspace methods are characterised by

$$r_k = p_k(A) r_0 \quad , p_k \in \Pi_k \ , p_k(0) = 1$$

AND some optimality condition

e.g. $\|r_k\|_2$ should be minimal over $x_k \in x_0 + \mathcal{K}_k(A, r_0)$.

$$\mathcal{K}_k(A, r_0) = \text{ span } \{r_0, A r_0, \ldots, A^{k-1} r_0\}$$

First step is however to compute a good basis for $\mathcal{K}_k(A, r_0$ since $A^k r_0$ tends to point in a single direction:

If $r_0 = \sum \alpha_i z_i, Az_i = \lambda_i z_i$ and if $|\lambda_1| > |\lambda_j|, j \neq 1$ then

$$\begin{aligned} A^k r_0 &= \sum_{i=1}^{n} \alpha_i A^k z_i = \sum_{i=1}^{n} \alpha_i \lambda_i^k z_i \\ &= \lambda_1^k \left[\alpha_1 z_1 + \sum_{i=2}^{n} \alpha_i \underbrace{\left(\frac{\lambda_i}{\lambda_1}\right)^k}_{\to 0 \text{ as } k \to \infty} z_i \right] \end{aligned}$$

Note also $\|A^k r_0\| \to 0$ or ∞ depending on whether $|\lambda_1| < 1$ or $|\lambda_1| > 1$

Arnoldi's method:

guess x_0 , $r_0 = b - Ax_0$, $v_1 = r_0/\|r_0\|_2$

for $l = 1, 2, \ldots$

$w = Av_l$

for $j = 1, \ldots, l$

$h_{jl} = v_j^T w$

$w = w - h_{jl} v_j$

end

$h_{l+1,l} = \|w\|_2$

$v_{l+1} = w/h_{l+1,l}$

end

is a way of generating an orthonormal basis $\{v_1, v_2, \ldots, v_k\}$ for $\mathcal{K}_k(A, r_0)$

In matrix form we can write Arnoldi as

$$\boldsymbol{AV_k = V_k H_k + h_{k+1,k}} \begin{bmatrix} | & | & & | \\ \mathbf{0} & \mathbf{0} & \cdots & \boldsymbol{v_{k+1}} \\ | & | & & | \end{bmatrix} = \boldsymbol{V_{k+1}\hat{H}_k}$$

where

$$\boldsymbol{V_k} = \begin{bmatrix} | & | & & | \\ \boldsymbol{v_1} & \boldsymbol{v_2} & \cdots & \boldsymbol{v_k} \\ | & | & & | \end{bmatrix}, \quad \text{has orthogonal columns,}$$

$$\boldsymbol{H_k} = \begin{bmatrix} \boldsymbol{h_{11}} & \boldsymbol{h_{12}} & \cdots & \cdots & \boldsymbol{h_{1k}} \\ \boldsymbol{h_{21}} & \boldsymbol{h_{22}} & \cdots & \cdots & \boldsymbol{h_{2k}} \\ \mathbf{0} & \boldsymbol{h_{32}} & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \mathbf{0} & \cdots & \mathbf{0} & \boldsymbol{h_{k,k-1}} & \boldsymbol{h_{kk}} \end{bmatrix} \in \mathbb{R}^{\boldsymbol{k \times k}}$$

is upper Hessenberg

and $\hat{H}_k = \begin{bmatrix} h_{11} & h_{12} & \cdots & \cdots & h_{1k} \\ h_{21} & h_{22} & \cdots & \cdots & h_{2k} \\ 0 & h_{32} & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & h_{k,k-1} & h_{kk} \\ 0 & \cdots & 0 & 0 & h_{k+1,k} \end{bmatrix} \in \mathbb{R}^{(k+1)}$

$= \begin{bmatrix} & H_k & & & \\ 0 & \cdots & 0 & 0 & h_{k+1,k} \end{bmatrix} \in \mathbb{R}^{(k+1)\times k}$

Note: $V_k \in \mathbb{R}^{n\times k}$ has orthogonal columns
$\Rightarrow V_k^T V_k = I \in \mathbb{R}^{k\times k}$ so

$AV_k = V_k H_k + h_{k+1,k} \begin{bmatrix} | & | & & | \\ 0 & 0 & \cdots & v_{k+1} \\ | & | & & | \end{bmatrix} \Rightarrow V_k^T A V_k =$

Now

$$x_k \in x_0 + \mathcal{K}_k(A, r_0) \Leftrightarrow x_k = x_0 + V_k y$$

for some $y \in \mathbb{R}^k$ since $\{v_1 \dots v_k\}$ is an orthonormal basis for $\mathcal{K}_k(A, r_0)$. Also

$$\begin{aligned} x_k = x_0 + V_k y \quad \Leftrightarrow \qquad x - x_k &= x - x_0 - V_k y \\ A(x - x_k) &= A(x - x_0) - AV_k y \\ \text{i.e.} \qquad r_k &= r_0 - AV_k y \end{aligned}$$

so $\|r_k\|_2$ is minimal $\Leftrightarrow$ $y \in \mathbb{R}^k$ is such that $\|r_0 - AV_k y\|_2$ is minimal.

$\|r_k\|_2$ is minimal $\Leftrightarrow$ $y \in \mathbb{R}^k$ is such that $\|r_0 - AV_k y\|_2$ is minimal.

But $r_0 = \|r_0\| v_1 = \|r_0\| V_k e_1 = \|r_0\| V_{k+1} e_1$,
$e_1^T = [1, 0, \ldots, 0]$

and by above $AV_k = V_{k+1}\hat{H}_k$ so

$$\|r_k\|_2 \text{ is min } \Leftrightarrow \left\| V_{k+1}(\|r_0\| e_1 - \hat{H}_k y) \right\|_2 \text{ is min}$$

but $V_{k+1}^T V_{k+1} = I \in \mathbb{R}^{(k+1)\times(k+1)}$ as $\{v_1, \ldots, v_{k+1}\}$

are orthonormal

so required vector y is that which minimises the linear least squares problem

$$\left\| \|r_0\| e_1 - \hat{H}_k y \right\|_2$$

Required vector y is that which minimises the linear least squares problem

$$\left\| \|r_0\| e_1 - \hat{H}_k y \right\|_2$$

$\Rightarrow$ need QR factorisation of the rectangular Hessenberg matrix $\hat{H}_k \in \mathbb{R}^{k+1 \times k}$
which can be achieved by one additional Givens rotation fo each k since $\hat{H}_k$ is built up by appending the last column for each k.

(see Exercises)

This is the basis of the GMRES algorithm
(Generalised Minimal Residual Method)

$\boldsymbol{x_0}$, $\boldsymbol{r_0 = b - Ax_0}$, $\boldsymbol{v_1 = r_0 / \|r_0\|}$

For $\boldsymbol{k = 1, 2, \ldots}$

do step $\boldsymbol{k}$ of the Arnoldi algorithm

($\Rightarrow$ have $\boldsymbol{v_1, v_2 \ldots \underbrace{v_{k+1}}_{\text{new}}}$ and $\underbrace{\hat{\boldsymbol{H}}_{\boldsymbol{k}}}_{\text{last column new}}$)

solve the Hessenberg linear least squares problem

$\boldsymbol{y} =$ arg min $\left\| \|\boldsymbol{r_0}\| \boldsymbol{e_1} - \hat{\boldsymbol{H}}_{\boldsymbol{k}} \boldsymbol{y} \right\|_{\boldsymbol{2}}$

$\boldsymbol{x_k = x_0 + V_k y}$

end

As before the Hessenberg least squares problem is solved by QR factorisation of $\hat{H}_k$ using $k+1$ Givens rotations, but since $\hat{H}_k$ is the same as $\hat{H}_{k-1}$ except for one additional row and column this can be implemented as only 1 Givens rotation for each k. (see exercises)

Notes:

1. $\|r_k\|_2$ is the linear least squares error in the Hessenberg least squares problem.
2. x_k only needs to be calculated if $\|r_k\|$ satisfies the stopping criterion $\|r_k\| \leq$ TOL.
3. work at k^{th} GMRES iteration is $O(k^2)$ for the least squares solution + 1 matrix vector product + vector operations. So for a sparse matrix with $O(1)$ entries per row work $\simeq O(k^2 n) \Rightarrow$ cheap method if only relatively few iterations (k) are needed.

Notes (continued);

4. GMRES gets expensive in storage of $v_1 \ldots v_k$ and the orthogonalisation computation if k gets too large, so sometimes is restarted: do a fixed number l of GMRES iterations then reset $r_0 \leftarrow r_l$ and repeat. This is GMRES (l) (which is not guaranteed to work!). Unrestarted GMRES is often called FULL GMRES.

5. If $A \in \mathbb{R}^{n \times n}$ then if GMRES does not stop before n steps, $\{v_1, \ldots, v_n\}$ is an orthonormal basis for $\mathbb{R}^n$ $\Rightarrow x_k = x$ because $\|r_k\|_2$ is minimimal for $x_k \in x_0 + \mathcal{K}_n(A, r_0)$ ie. for $x_k \in \mathbb{R}^n$.

6. Also if continues for n steps $AV_n = V_n H_n$, ie. $A = V_n H_n V_n^T$, $V_n, H_n \in \mathbb{R}^{n \times n}$. So A has been reduced by orthogonal similarity transform to Hessenberg form.

Convergence of GMRES

$$r_k = p_k(A) r_0 \text{ with } \|r_k\| \text{ minimal}$$

$\Rightarrow$ GMRES implicitly finds $p_k \in \Pi_k \ , \ p_k(0) = 1$ such that $\|p_k(A) r_0\|_2$ is minimal.

If A is diagonalisable

$$A = X \Lambda X^{-1} \Rightarrow p_k(A) = X p_k(\Lambda) X^{-1}$$

$$A = X\Lambda X^{-1} \Rightarrow p_k(A) = X p_k(\Lambda) X^{-1}$$

implies

$$\|r_k\|_2 = \|p_k(A) r_0\|_2 = \|X p_k(\Lambda) X^{-1} r_0\|$$

$$\leq \min_{p \in \Pi_k, p(0)=1} \|X\|_2 \|X^{-1}\|_2 \, \|p(\Lambda)\|_2 \, \|r_0\|$$

or similar to before

$$\frac{\|r_k\|_2}{\|r_0\|_2} \leq \|X\|_2 \|X^{-1}\|_2 \min_{p \in \Pi_k, p(0)=1} \max_{\lambda_j} |p(\lambda_j$$

λ_j ↑ eigenvalues of A

Comments:

1. $\|X\|_2\|X^{-1}\|_2 = \kappa(X)$ is a number independent of k: i
   it is large $\Rightarrow$ $(\star)$ is not very useful convergence
   estimate. If it is of moderate size (not known in
   practice!) then fast convergence if $\exists p \in \Pi_k\ ,\ p(0) = 1$
   such that $p(\lambda_j)$ small for all eigenvalues λ_j of A.
2. Other GMRES convergence bounds exist, but so far
   non is descriptive over a range of problems.
3. the expense of GMRES for k large has led to
   development of algorithms for non-symmetric matrices
   with fixed work per iteration, but these necessarily mus
   loose any optimality property in general.

However: If $A = A^T$ then GMRES has fixed work per step
To see this we have

$$V_k^T A V_k = H_k$$

but $A = A^T$ so LHS is symmetric $\Rightarrow$ H_k is symmetric and Hessenberg $\Rightarrow$ tridiagonal

i.e. Arnoldi's algorithm would calculate lots of zeros in this case!

In fact the symmetric Lanczos algorithm is used:

If $A = A^T$ is positive definite, there is an even more efficient Krylov subspace method than MINRES, namely the method of *Conjugate Gradients* (for $Ax = b$) which computes $x_k \in x_0 + \mathcal{K}_k(A, r_0)$ such that $\|x - x_k\|_A$ is minimal where $y^T A y = \|y\|_A^2$.

Note $\|\cdot\|_A$ defines a norm when A is symmetric and positive definite, indeed $< z, y >_A = < Az, y > = \ y^T A z$ defines a scalar product (inner product) for which $< y, y >_A = \ \|y\|_A^2$ in this case.

Lemma Conjugate Gradient Algorithm

choose x_0 , $r_0 = b - Ax_0 = p_0$ and for $k = 0, 1, 2, \ldots$

$$\begin{aligned} \alpha_k &= p_k^T r_k / p_k^T A p_k \\ x_{k+1} &= x_k + \alpha_k p_k \\ r_{k+1} &= b - A x_{k+1} \\ \beta_k &= -p_k^T A r_{k+1} / p_k^T A p_k \\ p_{k+1} &= r_{k+1} + \beta_k p_k \end{aligned}$$

computes iterates $\{x_k\}$, corresponding residuals $\{r_k\}$ and search directions $\{p_k\}$ so that as long as $x_k \neq x$ we have

$$\begin{aligned} r_k^T p_j &= r_k^T r_j = 0 \ , \ j < k \qquad (1) \\ p_k^T A p_j &= 0 \ , \ j < k \ , \ (p_k^T A p_k \neq 0) \ (2 \\ \text{span}\{r_0, r_1, \ldots, r_{k-1}\} &= \text{span}\{p_0, p_1, \ldots, p_{k-1}\} \end{aligned}$$

It is now an easy induction using $x_k = x_{k-1} + \alpha_{k-1} p_{k-1}$ t
show that $x_k \in x_0 + \mathcal{K}_k(A, r_0)$ (exercise)
and also

Theorem

$$\|x - x_k\|_A \leq \|x - y\|_A \quad , \quad y \in x_0 + \mathcal{K}_k(A, r_0)$$

Proof

let $c = x - x_0$ and $c_k = x_k - x_0 \in \mathcal{K}_k(A, r_0)$
then $Ac = r_0$ and $x - x_k = c - c_k$
hence $r_k = A(x - x_k) = A(c - c_k)$.

Now by the above r_k is orthogonal to every vector in $\mathcal{K}_k(A, r_0)$ i.e. $\forall\, v \in \mathcal{K}_k(A, r_0)$

$$0 = \langle r_k, v \rangle = \langle A(c - c_k), v \rangle = \langle c - c_k, v \rangle_A$$

and such (Galerkin) orthogonality $\Rightarrow \|c - y\|_A$ is minimised for $y \in \mathcal{K}_k(A, r_0)$ when $y = c_k$

$\Rightarrow \|x - z\|_A$ is minimised for $z \in x_0 + \mathcal{K}_k(A, r_0)$ by $z = x_k$ since $c = x - x_0$ and $x_k = x_0 + c_k$.

Convergence of Conjugate Gradients

we have $r_k = p_k(A) r_0$ or equivalently
$(x - x_k) = p_k(A)(x - x_0)$, $p_k \in \Pi_k, p_k(0) = 1$ and
$\|x - x_k\|_A = \|r_k\|_{A^{-1}}$ minimal over $x_k \in x_0 + \mathcal{K}_k(A, r_0)$
for each k.

Let $Av_j = \lambda_j v_j, \quad j = 1, \ldots, n, \quad v_j^T v_i = \delta_{ij}$ and
$x - x_0 = \sum_{j=1}^{n} \alpha_j v_j$ then

$$x - x_k = \sum_{j=1}^{n} \alpha_j p_k(A) v_j = \sum_{j=1}^{n} \alpha_j p_k(\lambda_j) v_j$$

so $\langle x - x_k, x - x_k \rangle_A = \sum_{j=1}^{n} \alpha_j^2 p_k(\lambda_j)^2 \langle v_j, v_j \rangle_A$

since $v_j^T v_i = \delta_{ij} \Rightarrow v_j^T A v_i = \lambda_i \delta_{ij} = 0$ if $i \neq j$

hence $\|x - x_k\|_A \leq \min_{p \in \Pi_k, p(0)=1} \max_j |p(\lambda_j)| \; \|x - x_0\|_A$

Convergence bound for Conjugate Gradients:

$$\frac{\|x - x_k\|_A}{\|x - x_0\|_A} \leq \min_{p \in \Pi_k, p(0)=1} \max_j |p(\lambda_j)|$$

If $\lambda_j \in [a, b], a > 0$ for all j then

$$\frac{\|x - x_k\|_A}{\|x - x_0\|_A} \leq \min_{p \in \Pi_k, p(0)=1} \max_{t \in [a,b]} |p(t)|$$

and the minimum here is achieved by the shifted and scale
Chebyshev polynomial

$$p(t) = T_k\left(\frac{2t - b - a}{b - a}\right) \Big/ T_k\left(\frac{-b - a}{b - a}\right)$$

similarly to before (but note different normalisation).

Now for

$$t \in [a,b] \quad , \frac{2t-b-a}{b-a} \in [-1,1]$$

so

$$\max_{t \in [a,b]} |p(t)| = p(b) = \frac{\frac{1}{2^{k-1}}1}{\frac{1}{2^{k-1}}\cosh k\theta} \quad , \cosh\theta = \frac{b+a}{b-a}$$

Note: $e^{\theta} + e^{-\theta} = 2\left(\frac{b+a}{b-a}\right) \Leftrightarrow \left(e^{\theta}\right)^2 - 2\left(\frac{a+b}{b-a}\right)\left(e^{\theta}\right) + 1 =$

$$\Leftrightarrow e^{\theta} = -\left(\frac{\sqrt{b}-\sqrt{a}}{\sqrt{b}+\sqrt{a}}\right) \text{ or } \left(\frac{-\sqrt{b}-\sqrt{a}}{\sqrt{b}-\sqrt{a}}\right),$$

So

$$\min_{p \in \Pi_k, p_0 = 1} \max_{t \in [a,b]} |p(t)| =$$

$$2 \left[\left(\frac{\sqrt{b} - \sqrt{a}}{\sqrt{b} + \sqrt{a}} \right)^k + \left(\frac{\sqrt{b} + \sqrt{a}}{\sqrt{b} - \sqrt{a}} \right)^k \right]^{-1}$$

$$\leq 2 \left(\frac{\sqrt{b} - \sqrt{a}}{\sqrt{b} + \sqrt{a}} \right)^k = 2 \left(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \right)^k$$

when $b = \lambda_{\max}(A), a = \lambda_{\min}(A)$ and $\kappa = \frac{\lambda_{\max}(A)}{\lambda_{\min}(A)}$

κ the $\| \cdot \|_2$ condition number again.

$$CG: \quad \frac{\|x - x_k\|_A}{\|x - x_0\|_A} \leq \min_{p \in \Pi_k, p(0)=1} \max_j |p(\lambda_j)|$$

$$\leq 2\left(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right)^k, \quad \kappa = \frac{\lambda_{\max}(A)}{\lambda_{\min}(A)}$$

For fast convergence:

$$CG: \quad \frac{\|x - x_k\|_A}{\|x - x_0\|_A} \leq \min_{p \in \Pi_k, p(0)=1} \max_j |p(\lambda_j)|$$

$$\leq 2 \left(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \right)^k, \quad \kappa = \frac{\lambda_{\max}(A)}{\lambda_{\min}(A)}$$

For fast convergence:

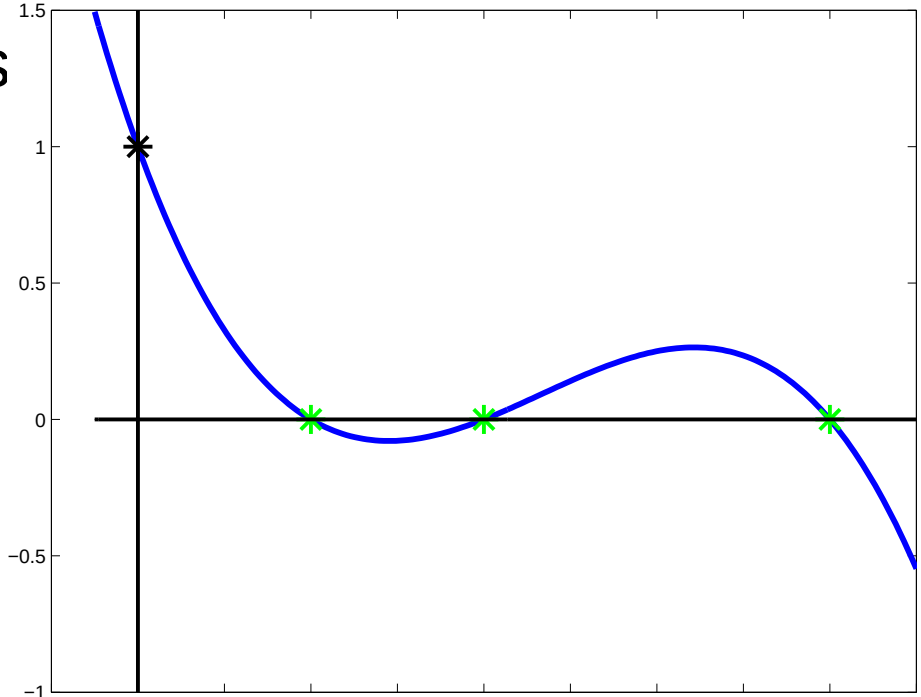
- few distinct eigenvalues
- cluster eigenvalues
- reduce κ

$$CG: \quad \frac{\|x - x_k\|_A}{\|x - x_0\|_A} \leq \min_{p \in \Pi_k, p(0)=1} \max_j |p(\lambda_j)|$$

$$\leq 2\left(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right)^k, \quad \kappa = \frac{\lambda_{\max}(A)}{\lambda_{\min}(A)}$$

For fast convergence:

- few distinct eigenvalues
- cluster eigenvalues
- reduce κ

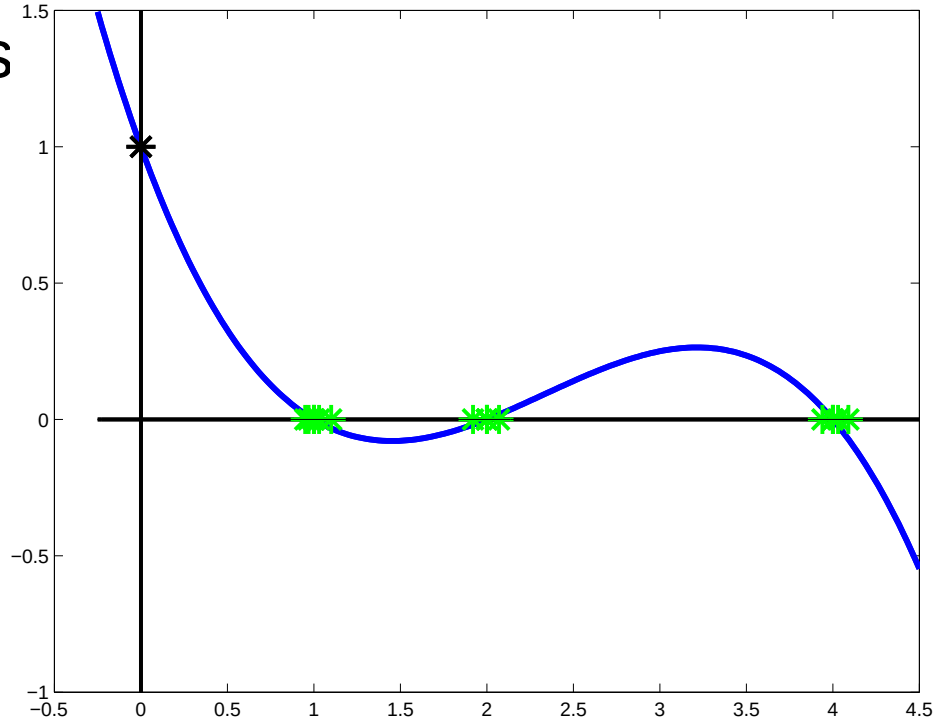


$$CG: \quad \frac{\|x - x_k\|_A}{\|x - x_0\|_A} \leq \min_{p \in \Pi_k, p(0)=1} \max_j |p(\lambda_j)|$$

$$\leq 2 \left(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \right)^k, \quad \kappa = \frac{\lambda_{\max}(A)}{\lambda_{\min}(A)}$$

For fast convergence:

- few distinct eigenvalues
- cluster eigenvalues
- reduce κ

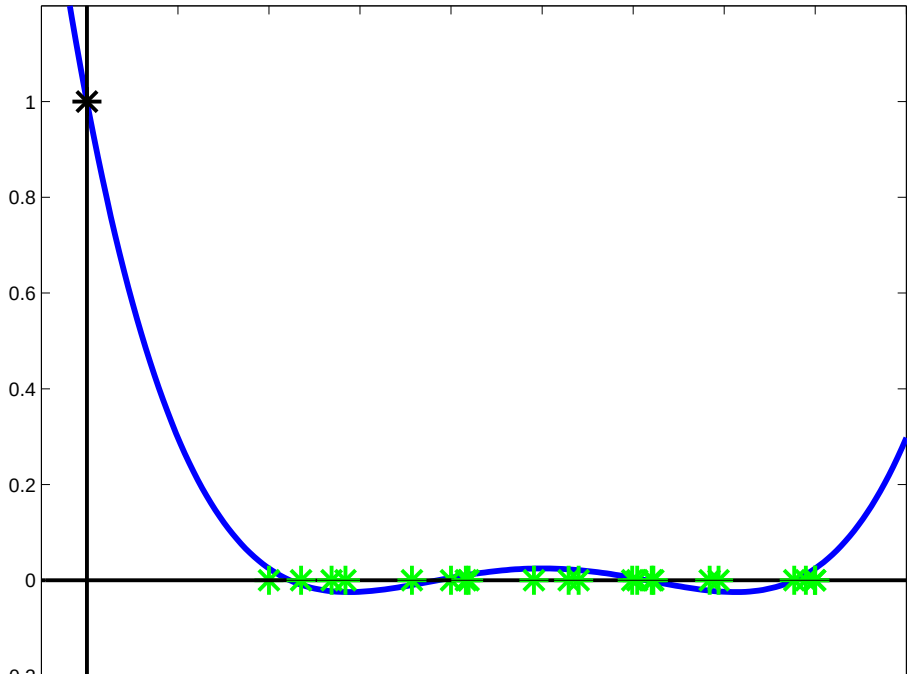


$$CG: \quad \frac{\|x - x_k\|_A}{\|x - x_0\|_A} \leq \min_{p \in \Pi_k, p(0)=1} \max_j |p(\lambda_j)|$$

$$\leq 2\left(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right)^k, \quad \kappa = \frac{\lambda_{\max}(A)}{\lambda_{\min}(A)}$$

For fast convergence:

- few distinct eigenvalues
- cluster eigenvalues
- reduce κ



$$\begin{aligned}
\frac{\|x - x_k\|_A}{\|x - x_0\|_A} &\leq \min_{p \in \Pi_k, p(0)=1} \max_j |p(\lambda_j)| \\
&\leq \min_{p \in \Pi_k, p(0)=1} \max_{t \in [a,b]} |p(t)| \\
&\leq 2 \left(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \right)^k
\end{aligned}$$

Hence Conjugate Gradients guaranteed to converge fast if

(i) A has few distant eigenvalues (or few clusters)

(ii) κ is small, e.g. if $\kappa = 9$

$$\frac{\|x - x_k\|_A}{\|x - x_0\|_A} \leq 2 \left(\frac{3 - 1}{3 + 1} \right)^k = \frac{2}{2^k}$$

error halving at each iteration.

BUT A, b given not chosen! So how can fast convergence be achieved for $Ax = b$ when none of (i), (ii) are true?

Preconditioning: choose symmetric positive definite matrix P and for the mathematical analysis only (not for the algorithm - see exercises) let $P = HH^T$ for example be a Cholesky factorisation. Solve

$$\underbrace{(H^{-1}AH^{-T})}_{\text{symmetric}}(H^T x) = H^{-1}b \qquad (\star)$$

by Conjugate Gradients.

It computes iterates $\{x_k\}$ for $(\star)$ and hence for $Ax = b$ such that

$$\frac{\|x - x_k\|_A}{\|x - x_0\|_A} \leq 2 \left(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \right)^k$$

where

$$\begin{aligned} \kappa &= \frac{\lambda_{\max}(H^{-1} A H^{-T})}{\lambda_{\min}(H^{-1} A H^{-T})} \\ &= \frac{\lambda_{\max}(P^{-1} A)}{\lambda_{\min}(P^{-1} A)} \end{aligned}$$

because of the similarity transformation

$$H^{-T}(H^{-1} A H^{-T}) H^T = P^{-1} A.$$

This CG method with preconditioner P is called the Preconditioned Conjugate Gradient method

The preconditioned Conjugate Gradient method requires the solution for z_k given r_k of

$$Pz_k = r_k$$

at each iteration $\Rightarrow$ the (contradictory) requirements:

- $\kappa(P^{-1}A) << \kappa(A) \quad \Rightarrow$ fast convergence
- $Pz_k = r_k$ easy to solve

(1) $P = A \Rightarrow$ convergence in 1 iteration, but solution of linear system with A at that iteration!

(2) $P = I \Rightarrow$ unpreconditioned CG

(2)leads to simple ideas like $P = diag(A)$ or $P = tridiag(A)$ or even $P =$part of A with bandwidth b $\Rightarrow O(nb^2)$ work for $Pz = r$ solve by banded elimination.
(1)leads to consideration of approximate or incomplete

Incomplete Cholesky(0) factorization:

for $i = 1, 2, \ldots n$
  $m = \min\{k : a_{i,k} \neq 0\}$
  for $j = m, \ldots, i-1$
    if $a_{i,j} \neq 0$
      $l_{i,j} \leftarrow \left(a_{i,j} - \sum_{k=m}^{j-1} l_{i,k} l_{j,k}\right) / l_{j,j}$
    endif
  enddo
  $l_{i,i} = \left(a_{i,i} - \sum_{k=m}^{i-1} l_{i,k} l_{i,k}\right)^{\frac{1}{2}}$
enddo

Incomplete Cholesky(0) factorization computes entries of a lower triangular matrix L such that $LL^T = A + R$ where

sparsity pattern of L = sparsity pattern of lower triangle of A

R is the remainder.

Effect of preconditioning in this case: replace $Ax = b$ by

$$L^{-1}AL^{-T}(Lx) = L^{-1}b$$

and note $L^{-1}AL^{-T} = I - L^{-1}RL^{-T}$ so if R small can expect eigenvalues of $L^{-1}AL^{-T}$ to be clustered around 1.

Note: solve $Pz = r$ is solve $LL^Tz = r$ is easily achieved

by forwards and back substitution.

Preconditioning can also be applied with GMRES and MINRES: for GMRES no symmetry to preserve so just solve eg. $\boldsymbol{P^{-1}Ax = P^{-1}b}$.

In the algorithm just replace $\boldsymbol{A*}$vector multiplies by

$$\boldsymbol{z = Av}, \quad \text{solve } \boldsymbol{Ps = z} \qquad \Rightarrow \quad \boldsymbol{s = P^{-1}Az}$$

x_0 , $r_0 = b - Ax_0 = p_0$ and for $k = 0, 1, 2, \ldots$

$$\begin{aligned} \alpha_k &= p_k^T r_k / p_k^T A p_k \\ x_{k+1} &= x_k + \alpha_k p_k \\ r_{k+1} &= b - Ax_{k+1} (= r_k - \alpha_k A p_k) \\ \beta_k &= -p_k^T A r_{k+1} / p_k^T A p_k (= r_{k+1}^T r_{k+1} / r_k^T r_k) \\ p_{k+1} &= r_{k+1} + \beta_k p_k \end{aligned}$$

Preconditioned Conjugate Gradient Algorithm choose x_0, $r_0 = b - Ax_0$, solve $Pz_0 = r_0$, $p_0 = z_0$, $k = 0, 1, \ldots$

$$\begin{aligned} \alpha_k &= z_k^T r_k / p_k^T A p_k \\ x_{k+1} &= x_k + \alpha_k p_k \\ r_{k+1} &= r_k - \alpha_k A p_k \\ \text{Solve } Pz_{k+1} &= r_{k+1} \\ \beta_k &= z_{k+1}^T r_{k+1} / z_k^T r_k \end{aligned}$$

Convergence:

$$
\begin{aligned}
\frac{\|x - x_k\|_A}{\|x - x_0\|_A} &\leq \min_{p \in \Pi_k, p(0)=1} \max_j |p(\lambda_j)| \\
&\leq \min_{p \in \Pi_k, p(0)=1} \max_{t \in [a,b]} |p(t)| \\
&\leq 2 \left(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \right)^k
\end{aligned}
$$

So fast convergence if

(i) A has few distant eigenvalues (or few clusters)

(ii) κ is small, e.g. if $\kappa = 9$

$$\frac{\|x - x_k\|_A}{\|x - x_0\|_A} \leq 2\left(\frac{3-1}{3+1}\right)^k = \frac{2}{2^k}$$

error halving at each iteration.

$$\kappa = \frac{\lambda_{\max}(P^{-1}A)}{\lambda_{\min}(P^{-1}A)}$$

Example: 5-point Finite Difference approx of Laplacian

$$-\nabla^2 u = f \quad \text{in } \Omega, \quad u = g \text{ on } \partial\Omega$$

Finite Differences:

$$A \sim h^{-2} \begin{array}{ccc} & -1 & \\ -1 & 4 & -1 \\ & -1 & \end{array}$$

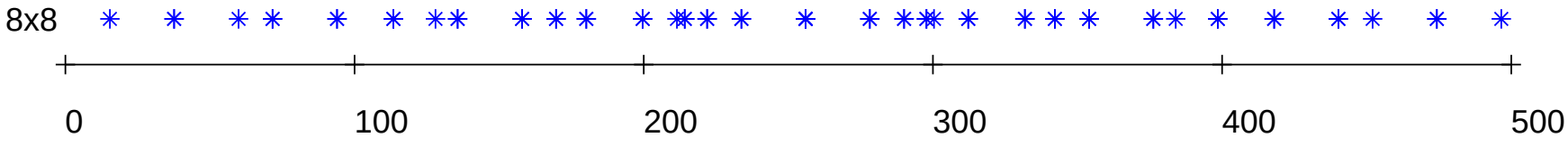
eg. on unit square A is block tridiagonal:

$$\underbrace{h^{-2}\begin{bmatrix} B & -I & & \\ -I & B & -I & \\ & \ddots & \ddots & \ddots \\ & & -I & B & -I \\ & & 0 & -I & B \end{bmatrix}}_{A \in \mathbb{R}^{n^2 \times n^2}}, \quad \underbrace{\begin{bmatrix} 4 & -1 & & \\ -1 & \ddots & \ddots & \\ & \ddots & \ddots & -1 \\ & & -1 & 4 \end{bmatrix}}_{B \in \mathbb{R}^{n \times n}}$$

5-point finite difference approx of Laplacian
Using discrete Fourier analysis eigenvalues known:

$$\lambda = h^{-2}\,\left[4 - 2\cos(r\pi h) - 2\cos(s\pi h)\right], \quad r, s = 1, \ldots, r$$

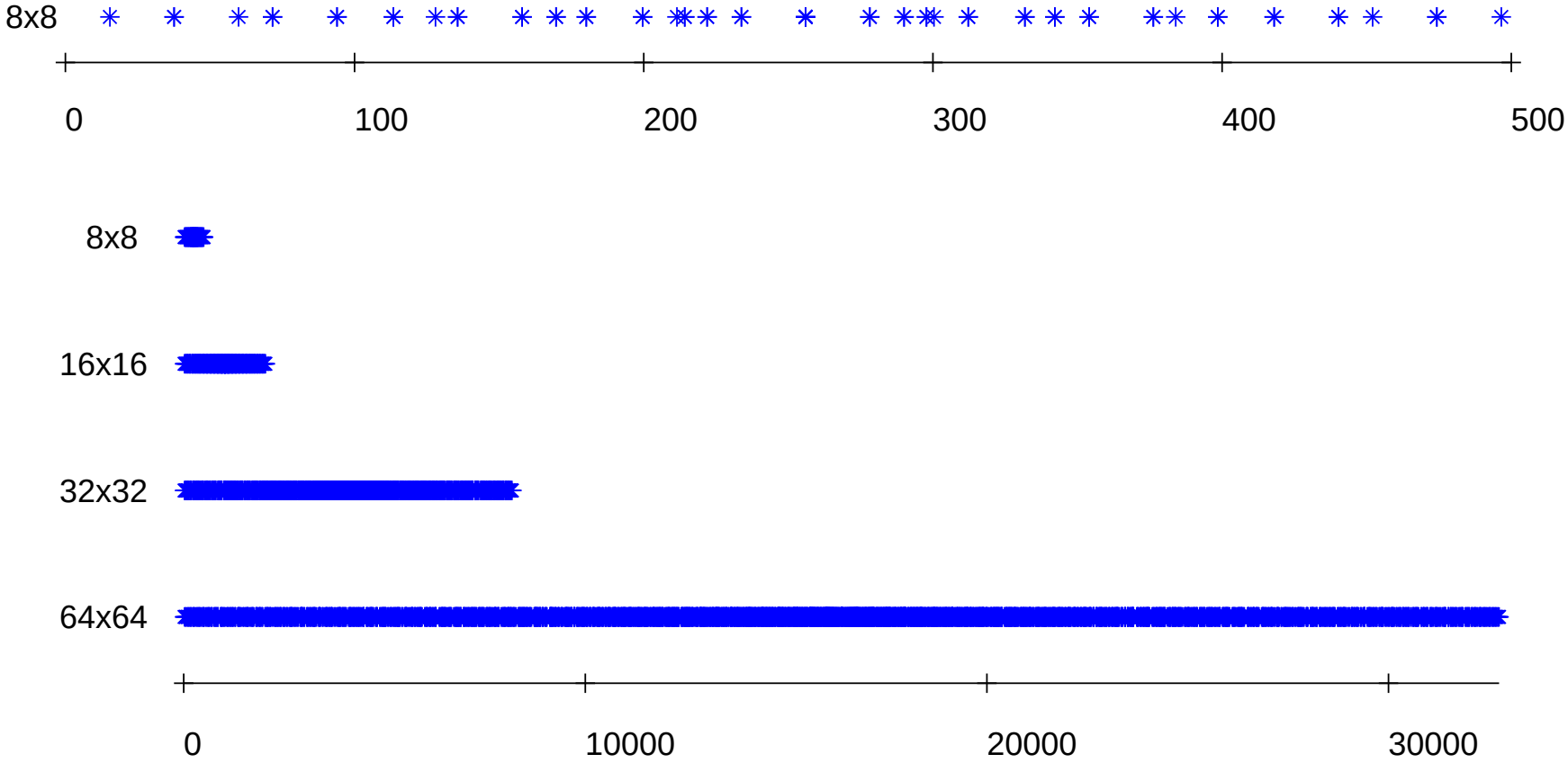
$$\Rightarrow \qquad \kappa = \frac{4}{\pi^2}\,h^{-2}, \qquad \lambda_{\min} \approx 2\pi^2,\ \lambda_{\max} \approx 8h^{-2}$$



5-point finite difference approx of Laplacian
Using discrete Fourier analysis eigenvalues known:

$$\lambda = h^{-2}\,\left[4 - 2\cos(r\pi h) - 2\cos(s\pi h)\right], \quad r, s = 1, \ldots, n$$

$$\Rightarrow \qquad \kappa = \frac{4}{\pi^2}\,h^{-2}, \qquad \lambda_{\min} \approx 2\pi^2,\ \lambda_{\max} \approx 8h^{-2}$$



Multigrid/Multilevel preconditioning

appropriate methods (smoothing and grid transfers) converge in a number of iterations independent of h
$\Rightarrow$ optimal solvers for Laplacian problems

Number of PCG iterations, Preconditioner is 1 V-cycles (contraction factor: η)

$\|\mathbf{r}^{(k)}\|/\|\mathbf{r}^{(0)}\| \leq 10^{-4}$

1 relaxed Jacobi iteration for pre- and post-smoothing.

grid	PCG iterations (MG contraction)	n
8×8	4 (0.10)	49
16×16	4 (0.11)	225
32×32	4 (0.12)	961
64×64	4 (0.14)	3969
128×128	5 (0.16)	16129

Multigrid:

Convergence bound: $\|\mathbf{u} - \mathbf{u}^{(k)}\|_A \leq \eta \|\mathbf{u} - \mathbf{u}^{(k-1)}\|_A$
η typically 0.1

$\Rightarrow$ multigrid is a great preconditioner for Laplacian because

$$\|\mathbf{u} - \mathbf{u}^{(k)}\|_A \leq \eta \|\mathbf{u} - \mathbf{u}^{(k-1)}\|_A$$

$$\Rightarrow 1 - \eta \leq \lambda_{\min}(P^{-1}A), \lambda_{\max}(P^{-1}A) \leq 1 + \eta$$

when P^{-1} is the action of a single multigrid cycle.

Hence $\kappa \leq (1+\eta)/(1-\eta)$ which is typically
$1.1/0.9 \approx 1.22$

Many other uses of these methods in diverse application areas