Numerical Methods
Numerical Linear Algebra

Martijn Boussé, Pieter Collins & Başak Sakçak

Department of Advanced Computing Sciences
Maastricht University
m.bousse,pieter.collins,basak.sakcak@maastrichtuniversity.nl

KEN1540 & KEN2540, Block 5, April-May 2025

Matrix form

System of linear algebraic equations:

$$\begin{aligned} 3x_1 - 7x_2 - 2x_3 + 2x_4 &= -9 \\ -3x_1 + 5x_2 + \ x_3 \qquad &= \ \ 5; \\ 6x_1 - 4x_2 + 2x_3 - 5x_4 &= \ \ 7; \\ -9x_1 + 5x_2 - 5x_3 + 6x_4 &= -19. \end{aligned}$$

Write using matrix form $Ax = b$ where

$$A = \begin{pmatrix} 3 & -7 & -2 & 2 \\ -3 & 5 & 1 & 0 \\ 6 & -4 & 2 & -5 \\ -9 & 5 & -5 & 6 \end{pmatrix}, \quad x = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix}, \quad b = \begin{pmatrix} -9 \\ 5 \\ 7 \\ -19 \end{pmatrix}.$$

Gaussian Elimination

Solve by Gaussian elimination:

$$\left(\begin{array}{cccc|c} 3 & -7 & -2 & 2 & -9 \\ -3 & 5 & 1 & 0 & 5 \\ 6 & -4 & 2 & -5 & 7 \\ -9 & 5 & -5 & 6 & -19 \end{array}\right) \sim \begin{array}{l} r_1 \\ r_2 - (-1)r_1 \\ r_3 - (+2)r_1 \\ r_4 - (-3)r_1 \end{array} \left(\begin{array}{cccc|c} 3 & -7 & -2 & 2 & -9 \\ 0 & -2 & -1 & 2 & -4 \\ 0 & 10 & 6 & -9 & 25 \\ 0 & -16 & -11 & 12 & -46 \end{array}\right) 4$$

$$\sim \begin{array}{l} r_1 \\ r_2 \\ r_3 - (-5)r_2 \\ r_4 - (+8)r_2 \end{array} \left(\begin{array}{cccc|c} 3 & -7 & -2 & 2 & -9 \\ 0 & -2 & -1 & 2 & -4 \\ 0 & 0 & 1 & 1 & 5 \\ 0 & 0 & -3 & -4 & -14 \end{array}\right)$$

$$\sim \begin{array}{l} r_1 \\ r_2 \\ r_3 \\ r_4 - (-3)r_3 \end{array} \left(\begin{array}{cccc|c} 3 & -7 & -2 & 2 & -9 \\ 0 & -2 & -1 & 2 & -4 \\ 0 & 0 & 1 & 1 & 5 \\ 0 & 0 & 0 & -1 & 1 \end{array}\right)$$

Solve by *backsubstitution* $x_4 = -1,\ x_3 = 6,\ x_2 = -2,\ x_1 = -3.$

LU Factorisation

Define L by letting L_{ij} be the multiple of row j *subtracted* from row i, and U the tableau matrix before backsubstitution.

$$L = \begin{pmatrix} 1 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 \\ 2 & -5 & 1 & 0 \\ -3 & 8 & -3 & 1 \end{pmatrix}, \quad U = \begin{pmatrix} 3 & -7 & -2 & 2 \\ 0 & -2 & -1 & 2 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & -1 \end{pmatrix}$$

Notice that $LU = A$!

$$LU = \begin{pmatrix} 1 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 \\ 2 & -5 & 1 & 0 \\ -3 & 8 & -3 & 1 \end{pmatrix} \begin{pmatrix} 3 & -7 & -2 & 2 \\ 0 & -2 & -1 & 2 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & -1 \end{pmatrix} = \begin{pmatrix} 3 & -7 & -2 & 2 \\ -3 & 5 & 1 & 0 \\ 6 & -4 & 2 & -5 \\ -9 & 5 & -5 & 6 \end{pmatrix} = A.$$

Solving $Ly = b$ gives $y = (-9, -4, 5, 1)^T$. Notice that this y is the right-hand column of the tableau before backsubstitution.

LU Factorisation

Factorization $A = LU$ where L is *unit lower triangular* $l_{ii} = 1, l_{ij} = 0$ for $j > i$ and U is *upper triangular* $u_{ij} = 0$ for $i > j$.

Backsubstitution The linear system $LUx = b$ can be solved by computing $Ly = b$ and $Ux = y$. Since L and U are triangular, the linear systems can be easily solved by *backsubstitution*

$$y_i = b_i - \sum_{j<i} l_{ij} y_j; \qquad x_i = (y_i - \sum_{j>i} u_{ij} x_j)/u_{ii}.$$

LU Factorisation—Complexity

Complexity Computing the LU-factorisation:

$$n(n-1) + (n-1)(n-2) + \cdots + 3 \cdot 2 + 2 \cdot 1 = (n^3 - n)/3 \sim (n^3/3) = O(n^3).$$

Solving the system $Ly = b$ requires

$$(n-1) + (n-2) + \cdots + 2 + 1 = (n^2 - n)/2$$

operations, and solving $Ux = y$ requires

$$n + (n-1) + \cdots + 2 + 1 = (n^2 + n)/2$$

operations, so computing x from L, U, b requires n^2 operations.

Hence for large systems, most of the work in solving $Ax = b$ lies in computing the LU-factorisation.

Once the LU-factorisation has been computed for a given A, the systems $Ax_i = b_i$ can easily be solved for different vectors b_i.

LU Factorisation—Pivoting

Let $A^{(k-1)}$ be the matrix obtained after working on the first $k-1$ columns.
Pivoting If $A_{kk}^{(k-1)}$ is equal to zero, then Gaussian elimination needs to swap rows.

Obtain a factorisation $PA = LU$, were P is a *permutation matrix*.

Partial pivoting Make the *pivot* element $A_{kk}^{(k-1)}$ the largest of $A_{kj}^{(k-1)}$ for $j \geq k$.

Scaled partial pivoting Define scale factor $s_k = \max_{1\leq j\leq k} |a_{kj}|$.
Choose first row to miaximise $|a_{k1}|/s_k = |a_{k1}|/\max_{1\leq j\leq k} |a_{kj}|$.

LU Factorisation—Pivoting Example (Omit)

Example Compute the LU-factorisation of A using partial pivoting and solve $Ax = b$ where

$$A = \begin{pmatrix} 2.11 & -4.21 & 0.921 \\ 1.09 & 0.987 & 0.832 \\ 4.01 & 10.2 & -1.12 \end{pmatrix}, \quad b = \begin{pmatrix} 2.01 \\ 4.21 \\ -3.09 \end{pmatrix}.$$

Swap row 1 and row 3 and eliminate x_1:

$$T_1 A = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} 2.11 & -4.21 & 0.921 \\ 1.09 & 0.987 & 0.832 \\ 4.01 & 10.2 & -1.12 \end{pmatrix} = \begin{pmatrix} 4.01 & 10.2 & -1.12 \\ 1.09 & 0.987 & 0.832 \\ 2.11 & -4.21 & 0.921 \end{pmatrix}$$

$$= \begin{pmatrix} 1 & 0 & 0 \\ 0.272 & 1 & 0 \\ 0.526 & 0 & 1 \end{pmatrix} \begin{pmatrix} 4.01 & 10.2 & -1.12 \\ 0 & -1.79 & 1.14 \\ 0 & -9.58 & 1.51 \end{pmatrix} = L_1 U_1$$

LU Factorisation—Pivoting Example (Omit)

Example $T_1 A = L_1 U_1$ given by

$$\begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} 1.09 & 0.987 & 0.832 \\ 2.11 & -4.21 & 0.921 \\ 4.01 & 10.2 & -1.12 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0.272 & 1 & 0 \\ 0.526 & 0 & 1 \end{pmatrix} \begin{pmatrix} 4.01 & 10.2 & -1.12 \\ 0 & -1.79 & 1.14 \\ 0 & -9.58 & 1.51 \end{pmatrix}$$

Swap row 2 and row 3, so $T_2 T_1 A = (T_2 L_1 T_2^{-1})(T_2 U_1)$, and eliminate x_2:

$$T_2 T_1 A = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} 2.11 & -4.21 & 0.921 \\ 1.09 & 0.987 & 0.832 \\ 4.01 & 10.2 & -1.12 \end{pmatrix}$$

$$= \begin{pmatrix} 1 & 0 & 0 \\ 0.526 & 1 & 0 \\ 0.272 & 0 & 1 \end{pmatrix} \begin{pmatrix} 4.01 & 10.2 & -1.12 \\ 0 & -9.58 & 1.51 \\ 0 & -1.79 & 1.14 \end{pmatrix}$$

$$= \begin{pmatrix} 1 & 0 & 0 \\ 0.526 & 1 & 0 \\ 0.272 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0.186 & 1 \end{pmatrix} \begin{pmatrix} 4.01 & 10.2 & -1.12 \\ 0 & -9.58 & 1.51 \\ 0 & 0 & 0.855 \end{pmatrix}$$

$$= (T_2 L_1 T_2^{-1})\, L_2\, U_2$$

LU Factorisation—Pivoting Example (Omit)

Example Obtain $PA = LU$ with

$$PA = \begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} 2.11 & -4.21 & 0.921 \\ 1.09 & 0.987 & 0.832 \\ 4.01 & 10.2 & -1.12 \end{pmatrix} = \begin{pmatrix} 4.01 & 10.2 & -1.12 \\ 2.11 & -4.21 & 0.921 \\ 1.09 & 0.987 & 0.832 \end{pmatrix}$$

$$= \begin{pmatrix} 1 & 0 & 0 \\ 0.526 & 1 & 0 \\ 0.272 & 0.186 & 1 \end{pmatrix} \begin{pmatrix} 4.01 & 10.2 & -1.12 \\ 0 & -9.58 & 1.51 \\ 0 & 0 & 0.855 \end{pmatrix} = LU$$

Solve $Ax = b$ using $PAx = LUx = Pb$, so $Ux = y$ where $Ly = Pb$.

$$Pb = \begin{pmatrix} -3.09 \\ 2.01 \\ 4.21 \end{pmatrix}; \quad y = L \backslash Pb = \begin{pmatrix} -3.09 \\ 3.64 \\ 4.37 \end{pmatrix}; \quad x = U \backslash y = \begin{pmatrix} -0.428 \\ 0.427 \\ 5.11 \end{pmatrix}.$$

11 / 77

LU Factorisation—Pivoting Example (Omit)

Example Compute y by backsubstitution:

$$Ly = \begin{pmatrix} 1 & 0 & 0 \\ 0.526 & 1 & 0 \\ 0.272 & 0.186 & 1 \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} = \begin{pmatrix} -3.09 \\ 2.01 \\ 4.21 \end{pmatrix} = Pb$$

$$y_1 = -3.09;$$

$$0.526 y_1 + y_2 = 2.01 \implies$$

$$y_2 = 2.01 - 0.526 y_1 = 2.01 - 0.526 \times (-0.309)$$

$$= 3.64$$

$$0.272 y_1 + 0.186 y_2 + y_3 = 4.21 \implies$$

$$y_3 = 4.21 - 0.272 y_1 - 0.186 y_2 = 4.21 - 0.272 \times (-0.309) - 0.186 \times 3.64$$

$$= 4.37$$

12 / 77

LU Factorisation—Pivoting Example (Omit)

Example Compute x by backsubstitution:

$$Ux = \begin{pmatrix} 4.01 & 10.2 & -1.12 \\ 0 & -9.58 & 1.51 \\ 0 & 0 & 0.855 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} -3.09 \\ 3.64 \\ 4.37 \end{pmatrix} = y$$

$$\begin{aligned} 0.855x_3 &= 4.37 \implies \\ x_3 &= 4.37 \div 0.855 \\ &= 5.11 \end{aligned}$$

$$\begin{aligned} -9.58x_2 + 1.51x_3 &= 3.64 \implies \\ x_2 &= (3.64 - 1.51x_3) \div (-9.58) = (3.64 - 1.51\times 5.11) \div (-9.58) \\ &= 0.427 \end{aligned}$$

$$\begin{aligned} 4.01x_1 + 10.2x_2 - 1.12x_3 &= -3.09 \implies \\ x_1 &= \big(-3.09 - 10.2x_2 - (-1.12)x_3\big) \div 4.01 \\ &= \big(-3.09 - 10.2\times 0.427 + 1.12\times 5.11\big) \div 4.01 \\ &= -0.428 \end{aligned}$$

Sparsity

Sparsity A matrix is *sparse* if it has many zero elements.

Fill-in The inverse of a sparse matrix is usually dense.

Tridiagonal matrices A is *tridiagonal* if $a_{ij} = 0$ for $|i - j| > 1$.

LU-factorisation *preserves* the zeros for a *banded* matrix!

Complexity Computing $A^{-1}b$ if A is tridiagonal requires n^2 operations; solving $LUx = b$ requires $\sim 3n$ operations!

Sparsity

Example For the given tridiagonal matrix, the inverse is

$$\begin{pmatrix} 4 & 1 & 0 & 0 & 0 \\ 1 & 4 & 1 & 0 & 0 \\ 0 & 1 & 4 & 1 & 0 \\ 0 & 0 & 1 & 4 & 1 \\ 0 & 0 & 0 & 1 & 4 \end{pmatrix}^{-1} = \begin{pmatrix} 0.268 & -0.072 & 0.019 & -0.005 & 0.001 \\ -0.072 & 0.287 & -0.077 & 0.021 & -0.005 \\ 0.019 & -0.077 & 0.288 & -0.077 & 0.019 \\ -0.005 & 0.021 & -0.077 & 0.287 & -0.072 \\ 0.001 & -0.005 & 0.019 & -0.072 & 0.268 \end{pmatrix}$$

and the LU-factorisation is

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0.250 & 1 & 0 & 0 & 0 \\ 0 & 0.267 & 1 & 0 & 0 \\ 0 & 0 & 0.268 & 1 & 0 \\ 0 & 0 & 0 & 0.268 & 1 \end{pmatrix} \begin{pmatrix} 4.000 & 1 & 0 & 0 & 0 \\ 0 & 3.750 & 1 & 0 & 0 \\ 0 & 0 & 3.733 & 1 & 0 \\ 0 & 0 & 0 & 3.732 & 1 \\ 0 & 0 & 0 & 0 & 3.732 \end{pmatrix}$$

Clearly, solving $LUx = b$ by backsubstitution is faster then computing $A^{-1}x$.

Sparsity

Example For the LU-factorisation

$$LU = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0.250 & 1 & 0 & 0 & 0 \\ 0 & 0.267 & 1 & 0 & 0 \\ 0 & 0 & 0.268 & 1 & 0 \\ 0 & 0 & 0 & 0.268 & 1 \end{pmatrix} \begin{pmatrix} 4.000 & 1 & 0 & 0 & 0 \\ 0 & 3.750 & 1 & 0 & 0 \\ 0 & 0 & 3.733 & 1 & 0 \\ 0 & 0 & 0 & 3.732 & 1 \\ 0 & 0 & 0 & 0 & 3.732 \end{pmatrix}$$

we have $(LU)^{-1} = U^{-1}L^{-1}$ given by

$$\begin{pmatrix} 0.250 & -0.067 & 0.018 & -0.005 & 0.001 \\ 0 & 0.267 & -0.071 & 0.019 & -0.005 \\ 0 & 0 & 0.268 & -0.072 & 0.019 \\ 0 & 0 & 0 & 0.268 & -0.072 \\ 0 & 0 & 0 & 0 & 0.268 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ -0.250 & 1 & 0 & 0 & 0 \\ 0.067 & -0.267 & 1 & 0 & 0 \\ -0.018 & 0.071 & -0.268 & 1 & 0 \\ 0.005 & -0.019 & 0.072 & -0.268 & 1 \end{pmatrix}$$

Clearly, solving $LUx = b$ by backsubstitution is faster then computing L^{-1}, U^{-1} and $x = U^{-1}(L^{-1}b)$

Symmetric matrices

Symmetric matrices A symmetric matrix ($A = A^T$) can be factorised $A = LDL^T$ where L is unit-lower-triangular and D is diagonal.

Positive-definite matrices A matrix is *positive definite* if $x^T Ax > 0$ whenever $x \neq 0$; equivalently, if all eigenvalues are positive, or if $A = LDL^T$ with D having strictly-positive diagonal elements.

Cholesky factorisation If A is positive definite, then there is an upper-triangular matrix U such that $A = U^T U$ (or a lower-triangular matrix L such that $A = LL^T$.)

Symmetric matrices—Example (Omit)

Example Compute the LDL$^\mathrm{T}$ factorisation:

$$A = \begin{pmatrix} 2 & 1 & 0 \\ 1 & 2 & 1 \\ 0 & 1 & 2 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 1/2 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 2 & 1 & 0 \\ 0 & 3/2 & 1 \\ 0 & 1 & 2 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 1/2 & 1 & 0 \\ 0 & 2/3 & 1 \end{pmatrix} \begin{pmatrix} 2 & 1 & 0 \\ 0 & 3/2 & 1 \\ 0 & 0 & 4/3 \end{pmatrix}$$

$$= \begin{pmatrix} 1 & 0 & 0 \\ 1/2 & 1 & 0 \\ 0 & 2/3 & 1 \end{pmatrix} \begin{pmatrix} 2 & 0 & 0 \\ 0 & 3/2 & 0 \\ 0 & 0 & 4/3 \end{pmatrix} \begin{pmatrix} 1 & 1/2 & 0 \\ 0 & 1 & 2/3 \\ 0 & 0 & 1 \end{pmatrix} = LDL^T$$

The Cholseky factorisation is given by $A = U^T U$ where

$$U = D^{1/2} L^T = \begin{pmatrix} \sqrt{2} & 0 & 0 \\ 0 & \sqrt{3/2} & 0 \\ 0 & 0 & \sqrt{4/3} \end{pmatrix} \begin{pmatrix} 1 & 1/2 & 0 \\ 0 & 1 & 2/3 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} \sqrt{2/1} & \sqrt{1/2} & 0 \\ 0 & \sqrt{3/2} & \sqrt{2/3} \\ 0 & 0 & \sqrt{4/3} \end{pmatrix}$$

Vector and matrix norms

Properties A *norm* is a measure of the *magnitude* of a vector (or matrix).

A vector norm $\|\cdot\|$ is a function $\mathbb{R}^n \to \mathbb{R}$ which satisfies

$$\|v\| \geq 0,\ \|v\| = 0 \iff v = 0;$$
$$\|\alpha v\| = |\alpha| \cdot \|v\|; \quad \|u + v\| \leq \|u\| + \|v\|.$$

A matrix norm additionally satisfies

$$\|AB\| \leq \|A\| \cdot \|B\|.$$

Given a vector norm $\|\cdot\|_*$, the corresponding matrix norm is

$$\|A\|_* = \max\{\|Ax\|_* \mid \|x\|_* = 1\}$$

and satisfies

$$\|Ax\|_* \leq \|A\|_* \times \|x\|_*$$

Vector and matrix norms

***p*-norms** Important vector norms are

$$\|v\|_p := \left(\sum_{i=1}^n |v_i|^p\right)^{1/p}.$$
$$\|v\|_\infty := \lim_{p\to\infty} \|v\|_p.$$

Note

$$\|v\|_1 = \sum_{i=1}^n |v_i|;$$
$$\|v\|_2 = \sqrt{\sum_{i=1}^n v_i^2} = \sqrt{v \cdot v};$$
$$\|v\|_\infty = \max_{i=1,\ldots,n} |v_i|\,.$$

The two-norm $\|v\|_2$ gives the Euclidean length of v. The *uniform* norm $\|v\|_\infty$ gives the maximum absolute value of the components, and is usually easiest to compute.

The corresponding matrix norms are

$$\|A\|_1 = \max_{j=1,\ldots,n} \sum_{i=1}^m |a_{ij}|;$$
$$\|A\|_2 = \max(\mathrm{eig}(A^T A))^{1/2};$$
$$\|A\|_\infty = \max_{i=1,\ldots,m} \sum_{j=1}^n |a_{ij}|.$$

Condition number

Conditioning For a given vector norm $\|\cdot\|$ and corresponding matrix norm, define the matrix *condition number* $K(A) := ||A|| \times ||A^{-1}||$.

Suppose $\tilde{x}$ is an approximate solution to $Ax = b$. Then

$$||\tilde{x} - x|| = \|A^{-1}(A\tilde{x} - Ax)\| \leq ||A^{-1}|| \; ||A\tilde{x} - b||$$

so the error satisfies:

$$||\tilde{x} - x|| \leq K(A) \frac{||A\tilde{x} - b||}{||A||} .$$

Further, since $\|A\| \cdot \|x\| \geq \|Ax\| = \|b\|$, we have $1/\|x\| \leq \|A\|/\|b\|$, so the relative error satisfies:

$$\frac{||\tilde{x} - x||}{||x||} \leq K(A) \frac{||A\tilde{x} - b||}{||b||}$$

Setting $\tilde{b} = A\tilde{x}$, then

$$\frac{||\tilde{x} - x||}{||x||} \leq K(A) \frac{||\tilde{b} - b||}{||b||}$$

Iterative refinement

Approximate solution Suppose $\tilde{x}$ is an approximate solution to $Ax = b$.

Refinement Then

$$A(x - \tilde{x}) = Ax - A\tilde{x} = b - A\tilde{x} \approx 0,$$

so

$$x = \tilde{x} + A^{-1}(b - A\tilde{x}).$$

Accuracy If $A^{-1}b$ can be computed less accurately than Ax, refinement typically improves the accuracy of a solution.

Iterative Methods

General iterative method

Fixed-point For the linear system $Ax = b$, write $A = D + E$, where D is “easy” to invert.

Then $Ax = Dx + Ex = b$, so $Dx = b - Ex$ and

$$x = D^{-1}(b - Ex).$$

Alternatively, we can write

$$x = x - D^{-1}(Ax - b).$$

Update Attempt to improve x using the update

$$x' = D^{-1}(b - Ex) = x - D^{-1}(Ax - b).$$

Iteration Use this as a basis for an iterative method

$$x^{(n+1)} = D^{-1}(b - Ex^{(n)}) = x^{(n)} - D^{-1}(Ax^{(n)} - b).$$

Jacobi method

Fixed-point formula We have $\sum_{j=1}^{n} a_{ij}x_j = b_i$ for $i = 1, \ldots, n$.

Write $a_{ii}x_i + \sum_{j=1,j\neq i}^{n} a_{ij}x_j = b_i$. Rearranging gives

$$x_i = \frac{b_i - \sum_{j\neq i} a_{ij}x_j}{a_{ii}}.$$

We can use this as the basis for an *iterative* method.

Jacobi Method Iterate using

$$x_i' = \frac{b_i - \sum_{j\neq i} a_{ij}x_j}{a_{ii}}.$$

An alternative formula is

$$x_i' = x_i - \frac{\sum_j a_{ij}x_j - b_i}{a_{ii}}.$$

In matrix form

$$x' = D^{-1}(b - Ex) = x - D^{-1}(Ax - b)$$

where D is the diagonal of A and $E = A - D$.

Jacobi method

Example Solve $Ax = b$ using the Jacobi method starting at $x^{(0)} = 0$ for

$$A = \begin{pmatrix} 6 & 2 & 0 \\ 3 & 5 & -1 \\ -2 & 1 & 4 \end{pmatrix}, \quad b = \begin{pmatrix} 3 \\ 4 \\ 1 \end{pmatrix}.$$

$$x_1^{(1)} = (b_1 - a_{12}x_2^{(0)} - a_{13}x_3^{(0)})/a_{11} = (3 - 2\times 0.0 - 0\times 0.0)/6 = 0.500;$$
$$x_2^{(1)} = (b_2 - a_{21}x_1^{(0)} - a_{23}x_3^{(0)})/a_{22} = (4 - 3\times 0.0 - (-1)\times 0.0)/5 = 0.800;$$
$$x_3^{(1)} = (b_3 - a_{31}x_1^{(0)} - a_{32}x_2^{(0)})/a_{33} = (1 - (-2)\times 0.0 - 1\times 0.0)/4 = 0.250.$$

$$x_1^{(2)} = (b_1 - a_{12}x_2^{(1)} - a_{13}x_3^{(1)})/a_{11} = (3 - 2\times 0.800 - 0\times 0.250)/6 = 0.233;$$
$$x_2^{(2)} = (b_2 - a_{21}x_1^{(1)} - a_{23}x_3^{(1)})/a_{22} = (4 - 3\times 0.500 + 1\times 0.250)/5 = 0.550;$$
$$x_3^{(2)} = (b_3 - a_{31}x_1^{(1)} - a_{32}x_2^{(1)})/a_{33} = (1 + 2\times 0.500 - 1\times 0.800)/4 = 0.300.$$

Continuing yields
$$x^{(3)} = \begin{pmatrix} 0.3167 \\ 0.7200 \\ 0.2292 \end{pmatrix}, \quad x^{(4)} = \begin{pmatrix} 0.2600 \\ 0.6558 \\ 0.2283 \end{pmatrix}, \quad x^{(5)} = \begin{pmatrix} 0.2814 \\ 0.6897 \\ 0.2160 \end{pmatrix}.$$

Estimate error $\|x^{(5)} - x\| \lesssim \|x^{(5)} - x^{(4)}\} = 0.024$. Residual $\|Ax^{(5)} - b\| = 0.076$.

Gauss-Seidel method

Gauss-Seidel Method Rather than update all the x_i simultaneously, we can update in turn. Then

$$x'_i = \frac{b_i - \sum_{j=1}^{i-1} a_{ij}x'_j - \sum_{j=i+1}^{n} a_{ij}x_j}{a_{ii}}.$$

In other words,

$$\text{for } i = 1, \dots, n, \text{ set } x_i = \big(b_i - \textstyle\sum_{j\neq i} a_{ij}x_j\big)/a_{ii} = x_i - \big(\textstyle\sum_{j=1}^{n} a_{ij}x_j - b_i\big)/a_{ii}.$$

This is more easily implemented in code:

```
for i=1:n, x(i)=x(i)-(A(i,:)*x-b(i))/A(i,i); end;
```

or using an explicit loop:

```
for i=1:n,
  ri=-b(i);
  for j=1:n, ri=ri+A(i,j)*x(j); end;
  x(i)=x(i)-ri/A(i,i);
end;
```


Gauss-Seidel method

Example Solve $Ax = b$ using the Gauss-Seidel method for:

$$A = \begin{pmatrix} 6 & 2 & 0 \\ 3 & 5 & -1 \\ -2 & 1 & 4 \end{pmatrix}, \quad b = \begin{pmatrix} 3 \\ 4 \\ 1 \end{pmatrix}, \quad x^{(0)} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

$$\begin{aligned}
x_1^{(1)} &= (b_1 - a_{12}x_2^{(0)} - a_{13}x_3^{(0)})/a_{11} = (3 - 2\times 0.0000 - 0\times 0.0000)/6 = 0.5000;\\
x_2^{(1)} &= (b_2 - a_{21}x_1^{(1)} - a_{23}x_3^{(0)})/a_{22} = (4 - 3\times 0.5000 - (-1)\times 0.0000)/5 = 0.5000;\\
x_3^{(1)} &= (b_3 - a_{31}x_1^{(1)} - a_{32}x_2^{(1)})/a_{33} = (1 - (-2)\times 0.5000 - 1\times 0.5000)/4 = 0.3750.
\end{aligned}$$

$$\begin{aligned}
x_1^{(2)} &= (b_1 - a_{12}x_2^{(1)} - a_{13}x_3^{(1)})/a_{11} = (3 - 2\times 0.5000 - 0\times 0.3750)/6 = 0.3333;\\
x_2^{(2)} &= (b_2 - a_{21}x_1^{(2)} - a_{23}x_3^{(1)})/a_{22} = (4 - 3\times 0.3333 + 1\times 0.3750)/5 = 0.6750;\\
x_3^{(2)} &= (b_3 - a_{31}x_1^{(2)} - a_{32}x_2^{(2)})/a_{33} = (1 + 2\times 0.3333 - 1\times 0.6750)/4 = 0.2479.
\end{aligned}$$

$$x^{(1)} = \begin{pmatrix} 0.50000 \\ 0.50000 \\ 0.37500 \end{pmatrix}, \; x^{(2)} = \begin{pmatrix} 0.33333 \\ 0.67500 \\ 0.2479 \end{pmatrix}, \; x^{(3)} = \begin{pmatrix} 0.27500 \\ 0.68458 \\ 0.21635 \end{pmatrix}, \; x^{(4)} = \begin{pmatrix} 0.27181 \\ 0.68019 \\ 0.21568 \end{pmatrix},$$

Convergence is rapid.

Gauss-Seidel method

Example For

$$A = \begin{pmatrix} 3 & -7 & -2 & 2 \\ -3 & 5 & 1 & 0 \\ 6 & -4 & 2 & -5 \\ -9 & 5 & -5 & 6 \end{pmatrix}, \quad b = \begin{pmatrix} -9 \\ 5 \\ 7 \\ -19 \end{pmatrix}, \quad x^{(0)} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

the Gauss-Seidel method gives

$$x^{(1)} = \begin{pmatrix} -3.000 \\ 1.000 \\ 3.500 \\ -3.167 \end{pmatrix}, \; x^{(2)} = \begin{pmatrix} 6.778 \\ -2.500 \\ 3.083 \\ -2.417 \end{pmatrix}, \; x^{(3)} = \begin{pmatrix} -11.944 \\ 6.950 \\ -30.958 \\ 14.069 \end{pmatrix}, \; x^{(4)} = \begin{pmatrix} -4.857 \\ -6.925 \\ 119.365 \\ -66.743 \end{pmatrix}$$

so does not converge!

Gauss-Seidel method

Matrix form Write $A = L + D + U$, where L is strictly lower-triangular, D is diagonal and U is strictly upper triangular.

i.e. $L_{i,j} = 0$ if $i \leq j$, $D_{i,j} = 0$ if $i \neq j$, $U_{i,j} = 0$ if $i \geq j$.

Note that L, U here are not the L and U of the LU factorisation!!

Then the Gauss-Seidel method is given by $x' = D^{-1}(b - Lx' + Ux)$, and rearranging gives

$$\begin{aligned} x' &= (L + D)^{-1}(b - Ux) \\ &= x - (L + D)^{-1}(Ax - b). \end{aligned}$$

Convergence

Theorem An iterative method $x' = Tx + r$ converges if $||T|| < 1$ for some matrix norm $|| \cdot ||$.

Definition A matrix A is *diagonally-dominant* if $|a_{ii}| > \sum_{j \neq i} |a_{ij}|$ for all i.

Theorem (Convergence of Jacobi / Gauss-Seidel) If A is diagonally-dominant, then the Jacobi and Gauss-Seidel iterations converge to the solution of $Ax = b$.

Preconditioning Can *precondition* A by multiplying by a matrix P to obtain $(PA)x = Pb$.

Approximate inverse Since if $J \approx I$, then J is diagonally-dominant, precondition by $P \approx A^{-1}$ to obtain $J = PA \approx I$.

Preconditioning

Example Let

$$A = \begin{pmatrix} 3 & -7 & -2 & 2 \\ -3 & 5 & 1 & 0 \\ 6 & -4 & 2 & -5 \\ -9 & 5 & -5 & 6 \end{pmatrix}, \quad b = \begin{pmatrix} -9 \\ 5 \\ 7 \\ -19 \end{pmatrix}, \quad P = \begin{pmatrix} 0 & 0 & -1 & -1 \\ 1 & 2 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}.$$

Then

$$PA = \begin{pmatrix} 3 & -1 & 3 & -1 \\ -3 & 3 & 0 & 2 \\ 6 & -4 & 2 & -5 \\ 3 & -7 & -2 & 2 \end{pmatrix}, \quad Pb = \begin{pmatrix} 12 \\ 1 \\ 7 \\ -9 \end{pmatrix}.$$

Applying the Gauss-Seidel method to $(PA)x = (Pb)$ gives iterates:

$$x^{(1)} = \begin{pmatrix} 4.0 \\ 4.3 \\ 0.2 \\ 4.8 \end{pmatrix}, \; x^{(2)} = \begin{pmatrix} 6.9 \\ 4.0 \\ 2.9 \\ 2.1 \end{pmatrix}, \; x^{(4)} = \begin{pmatrix} 1.7 \\ 1.0 \\ 4.2 \\ 0.8 \end{pmatrix}, \; x^{(8)} = \begin{pmatrix} -1.69 \\ -1.15 \\ 5.50 \\ -0.40 \end{pmatrix}, \; x^{(12)} = \begin{pmatrix} -2.63 \\ -1.76 \\ 5.86 \\ -0.85 \end{pmatrix}.$$

Method converges slowly to $x^{(\infty)} = (-3, -2, 6, -1)^T$.

Successive Over-Relaxation Method

Successive Over-Relaxation Use slightly larger update in the Gauss-Seidel method:

$$\begin{aligned} x_i' &= x_i - \omega \, \frac{\sum_{j<i} a_{ij} x_j' + \sum_{j \geq i} a_{ij} x_j - b_i}{a_{ii}} \\ &= (1-\omega) x_i + \omega \, \frac{b_i - \sum_{j<i} a_{ij} x_j' - \sum_{j>i} a_{ij} x_j}{a_{ii}}. \end{aligned}$$

with $\omega \gtrsim 1$.

Typically ω is taken to be in the range $1.1 \leq \omega \leq 1.3$.

Implement in Matlab as:

```
for i=1:n, x(i)=x(i)-omega*(A(i,:)*x-b(i))/A(i,i); end;
```

Explicitly in matrix form,

$$x' = x - \omega \, (\omega L + D)^{-1} (Ax - b).$$

Successive Over-Relaxation Method

Example Solve $Ax = b$ using the successive over-relaxation method with

$$A = \begin{pmatrix} 6 & 2 & 0 \\ 3 & 5 & -1 \\ -2 & 1 & 4 \end{pmatrix}, \quad b = \begin{pmatrix} 3 \\ 4 \\ 1 \end{pmatrix}, \quad x^{(0)} = 0, \quad \omega = 1.1.$$

First step gives $x^{(1)} = (0.5500, 0.5170, 0.4353)^T$. Second step:

$$\begin{aligned} x_1^{(2)} &= (1-\omega)x_1^{(1)} + \omega(b_1 - a_{12}x_2^{(1)} - a_{13}x_3^{(1)})/a_{11} \\ &= -0.1 \times 0.5500 + 1.1 \times (3 - 2{\times}0.5170 - 0{\times}0.4353)/6 = 0.3054; \\ x_2^{(2)} &= (1-\omega)x_2^{(1)} + \omega(b_2 - a_{21}x_1^{(2)} - a_{23}x_3^{(1)})/a_{22} \\ &= -0.1 \times 0.5170 + 1.1 \times (4 - 3{\times}0.3054 - (-1){\times}0.4353 - 4)/5 = 0.7225; \\ x_3^{(2)} &= (1-\omega)x_3^{(1)} + \omega(b_3 - a_{31}x_1^{(2)} - a_{32}x_2^{(2)})/a_{33} \\ &= -0.1 \times 0.4353 + 1.1 \times (1 - (-2){\times}0.3054 - 1{\times}0.7225)/4 = 0.2008. \end{aligned}$$

Further iterates give:

$$x^{(3)} = \begin{pmatrix} 0.25455 \\ 0.68392 \\ 0.20684 \end{pmatrix}, \ x^{(4)} = \begin{pmatrix} 0.27377 \\ 0.67642 \\ 0.21888 \end{pmatrix}, \ x^{(5)} = \begin{pmatrix} 0.27460 \\ 0.67927 \\ 0.21734 \end{pmatrix}, \quad x^{(\infty)} = \begin{pmatrix} 0.27358 \\ 0.67925 \\ 0.21698 \end{pmatrix}.$$

Conjugate-Gradient

Conjugate-Gradient Method

Method for solving $Ax = b$ where A is positive-definite.

Idea Construct sequence x_k minimising residual $||Ax_k - b||_2$ in $\mathrm{span}\{b, Ab, \ldots, A^{k-1}b\}$.

Use *inner products*

$$\langle u, S, v\rangle = \textstyle\sum_{i,j=1}^n u_i S_{ij} v_j \text{ and } \langle u, v\rangle = \langle u, I, v\rangle = \sum_i^n u_i v_i.$$

Algorithm

Initialise

$x_0 = 0, \ \ r_0 = b - Ax_0, \ \ v_1 = r_0;$

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

$s_k = \langle r_{k-1}, r_{k-1}\rangle / \langle r_{k-2}, r_{k-2}\rangle$, s_1 unused.

$v_k = r_{k-1} + s_k v_{k-1}$, $v_1 = r_0$;

$t_k = \langle r_{k-1}, r_{k-1}\rangle / \langle v_k, A, v_k\rangle$,

$x_k = x_{k-1} + t_k v_k$,

$r_k = r_{k-1} - A\, t_k v_k$,

Note $r_k = b - Ax_k$ and $\langle v_i, A, v_j\rangle = 0$ for $i \neq j$.

Conjugate-Gradient Method

Example Solve $Ax = b$ using the conjugate-gradient method, where

$$A = \begin{pmatrix} 6 & 3 & -1 \\ 3 & 5 & 2 \\ -1 & 2 & 4 \end{pmatrix}, \quad b = \begin{pmatrix} 3 \\ 4 \\ 1 \end{pmatrix}.$$

$$x_0 = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}, \quad r_0 = b = \begin{pmatrix} 3 \\ 4 \\ 1 \end{pmatrix};$$

$$v_1 = r_0 = \begin{pmatrix} 3 \\ 4 \\ 1 \end{pmatrix}$$

$$t_1 = \frac{\langle r_0, r_0 \rangle}{\langle v_1, A, v_1 \rangle} = \frac{26}{220} = 0.11818$$

$$x_1 = x_0 + t_1 v_1 = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} + 0.11818 \times \begin{pmatrix} 3 \\ 4 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} + \begin{pmatrix} 0.35455 \\ 0.47273 \\ 0.11818 \end{pmatrix} = \begin{pmatrix} 0.35455 \\ 0.47273 \\ 0.11818 \end{pmatrix}$$

$$r_1 = r_0 - A\,t_1 v_1 = \begin{pmatrix} 3 \\ 4 \\ 1 \end{pmatrix} - \begin{pmatrix} 6 & 3 & -1 \\ 3 & 5 & 2 \\ -1 & 2 & 4 \end{pmatrix} \times \begin{pmatrix} 0.35455 \\ 0.47273 \\ 0.11818 \end{pmatrix} = \begin{pmatrix} -0.427273 \\ 0.336364 \\ -0.063636 \end{pmatrix}$$

38 / 77

Conjugate-Gradient Method

Example

$$v_1 = \begin{pmatrix} 3.00000 \\ 4.00000 \\ 1.00000 \end{pmatrix}, \quad x_1 = \begin{pmatrix} 0.35455 \\ 0.47273 \\ 0.11818 \end{pmatrix}, \quad r_1 = \begin{pmatrix} -0.427273 \\ 0.336364 \\ -0.063636 \end{pmatrix}.$$

$$s_2 = \frac{\langle r_1, r_1 \rangle}{\langle r_0, r_0 \rangle} = \frac{0.29975}{26.00000} = 0.011529$$

$$v_2 = r_1 + s_2 v_1 = \begin{pmatrix} -0.427273 \\ 0.336364 \\ -0.063636 \end{pmatrix} + 0.011529 \times \begin{pmatrix} 3.00000 \\ 4.00000 \\ 1.00000 \end{pmatrix} = \begin{pmatrix} -0.39269 \\ 0.38248 \\ -0.05211 \end{pmatrix}$$

$$t_2 = \frac{\langle r_1, r_1 \rangle}{\langle v_2, A, v_2 \rangle} = \frac{0.29975}{0.64572} = 0.46422$$

$$x_2 = x_1 + t_2 v_2 = \begin{pmatrix} 0.355 \\ 0.473 \\ 0.118 \end{pmatrix} + 0.464 \times \begin{pmatrix} -0.393 \\ 0.382 \\ -0.052 \end{pmatrix} = \begin{pmatrix} 0.355 \\ 0.473 \\ 0.118 \end{pmatrix} + \begin{pmatrix} -0.1823 \\ 0.1776 \\ -0.0242 \end{pmatrix} = \begin{pmatrix} 0.17225 \\ 0.65028 \\ 0.09399 \end{pmatrix}$$

$$r_2 = r_1 - A\,t_2 v_2 = \begin{pmatrix} -0.4273 \\ 0.3364 \\ -0.0636 \end{pmatrix} - \begin{pmatrix} 6 & 3 & -1 \\ 3 & 5 & 2 \\ -1 & 2 & 4 \end{pmatrix} \times \begin{pmatrix} -0.1823 \\ 0.1776 \\ -0.0242 \end{pmatrix} = \begin{pmatrix} 0.109625 \\ 0.043850 \\ -0.504277 \end{pmatrix}$$

39 / 77

Conjugate-Gradient Method

Example

$$v_2 = \begin{pmatrix} -0.39269 \\ 0.38248 \\ -0.05211 \end{pmatrix}, \quad x_2 = \begin{pmatrix} 0.17225 \\ 0.65028 \\ 0.09399 \end{pmatrix}, \quad r_2 = \begin{pmatrix} 0.109625 \\ 0.043850 \\ -0.504277 \end{pmatrix}.$$

$$s_3 = \frac{\langle r_2, r_2 \rangle}{\langle r_1, r_1 \rangle} = \frac{0.26824}{0.29975} = 0.89486$$

$$v_3 = r_2 + s_3 v_2 = \begin{pmatrix} -0.24177 \\ 0.38612 \\ -0.55091 \end{pmatrix}$$

$$t_3 = \frac{\langle r_2, r_2 \rangle}{\langle v_3, A, v_3 \rangle} = \frac{0.26824}{0.63278} = 0.42390$$

$$\text{Solution } x = x_3 = x_2 + t_3 v_3 = \begin{pmatrix} 0.06977 \\ 0.81395 \\ -0.13953 \end{pmatrix}$$

$$r_3 = r_2 - A\, t_3 v_3 = \begin{pmatrix} 0.0 \\ 0.0 \\ 0.0 \end{pmatrix} = b - Ax \text{ (residual)}$$

Conjugate-Gradient Method

Convergence The conjugate-gradient method yields the exact result (up to roundoff errors) for an $n \times n$ system after n steps.

Termination Often, good accuracy is achieved after $m \ll n$ steps! This can be seen if the residual r is small.

Sparse systems If A is sparse, with $p \ll n^2$ nonzero elements, then each step takes $O(p)$ operations.

Efficiency For many practical problems, the conjugate-gradient method only needs $O(mp)$ steps for good accuracy, much less than $O(n^3)$ for Gaussian elimination!

Preconditioned Conjugate-Gradient (Non-examinable)

Preconditioning Choose a *preconditioning* matrix P.

Apply the conjugate-gradient method to the equations

$$(PAP^T)y = Pb$$

and set

$$x = P^T y.$$

A typical choice is $P = (\mathrm{diag}(A))^{1/2}$.

Eigenvalues and eigenvectors

Eigenvalues and Eigenvectors If $Av = \lambda v$ and $v \neq 0$, then λ is an *eigenvalue* of A with corresponding eigenvector v.

Similarity Matrices A and B are *similar* if $B = PAP^{-1}$ for some invertible matrix P.

If $Av = \lambda v$, then $B(Pv) = (PAP^{-1})Pv = PAv = P(\lambda v) = \lambda(Pv)$, so Pv is an eigenvector of B with eigenvalue λ.

Triangular If A is a lower- or upper-triangular matrix, then the eigenvalues of A are the diagonal entries; $\lambda_i = a_{ii}$

Diagonal If D is a diagonal matrix, then the eigenvectors are the standard unit basis vectors e_i, with $De_i = d_{ii}e_i$.

Notation Often write eigenvalues in order, $|\lambda_1| \geq |\lambda_2| \geq \cdots \geq |\lambda_n|$, with corresponding eigenvectors v_i.

Approximation theorems

Gersgorin Circle Theorem For $i = 1, \ldots, n$, there exists an eigenvalue λ_i of A within the circle

$$\{z \in \mathbb{C} \mid |z - a_{ii}| \leq \textstyle\sum_{j=1,j\neq i}^{n} |a_{ij}|\}.$$

Similarly, for $j = 1, \ldots, n$, there esists an eigenvalue λ_j of A in the circle

$$\{z \in \mathbb{C} \mid |z - a_{jj}| \leq \textstyle\sum_{i=1,i\neq j}^{n} |a_{ij}|\}.$$

Example

$$A = \begin{pmatrix} 10 & -2 & 1 \\ 1 & 3 & -1 \\ 0 & 1 & -2 \end{pmatrix}.$$

The eigenvalues $\lambda_{1,2,3}$ satisfy:

$$|\lambda_1 - 10| \leq |-2|+|1| = 3,\ |\lambda_2 - 3| \leq |1|+|-1| = 2,\ |\lambda_3 - (-2)| \leq 1.$$

If $\lambda_{1,2,3}$ are real, then $\lambda_1 \in [7, 13]$, $\lambda_2 \in [1, 5]$, $\lambda_3 \in [-3, -1]$.

By considering the first column, see $|\lambda_1 - 10| \leq 1$.

The power method

Power method Iterate

$$y^{(n)} = Ax^{(n)}, \quad x^{(n+1)} = y^{(n)}/\pm\|y^{(n)}\|.$$

Note $x^{(n)} = \pm A^n x^{(0)}/\|A^n x^{(0)}\|$.

Although any norm can be used, typically use supremum norm $\|y\|_\infty$, chosing sign so that maximum absolute value of x is one:

$$x^{(n+1)} = y^{(n)}/y^{(n)}_{i_{\mathrm{abs\,max}}} \text{ where } |y_{i_{\mathrm{abs\,max}}}| \geq |y_i| \text{ for all } i.$$

Take eigenvalue approximation

$$\mu^{(n)} = (Ax^{(n)})_{i_{\mathrm{abs\,max}}}/x^{(n)}_{i_{\mathrm{abs\,max}}} = y^{(n)}_{i_{\mathrm{abs\,max}}}/x^{(n)}_{i_{\mathrm{abs\,max}}}.$$

It is usually more accurate, notably for symmetric matrices, to take the alternative eigenvalue approximation

$$\mu^{(n)} = \frac{x^{(n)T}Ax^{(n)}}{x^{(n)T}x^{(n)}} = \frac{x^{(n)} \cdot y^{(n)}}{x^{(n)} \cdot x^{(n)}}.$$

If the Euclidean norm is used, then $\|x^{(n)}\|_2 = 1$, and the formula reduces to

$$\mu^{(n)} = x^{(n)T}Ax^{(n)} = x^{(n)} \cdot y^{(n)}.$$

The power method

Theorem If A has an eigenvalue λ_1 such that $|\lambda_1| > |\lambda_i|$ for all other eigenvalues, then the power method converges, with $\lim_{n\to\infty} x^{(n)} = v_1$ and $\lim_{n\to\infty} \mu^{(n)} = \lambda_1$.

Proof.

For simplicity, suppose $\mathbb{R}^n$ has basis $\{v_1, v_2, \dots, v_n\}$ of eigenvectors of A.

Write $x = \alpha_1 v_1 + \alpha_2 v_2 + \cdots + \alpha_n v_n$. Then

$$\begin{aligned}
A^k x &= A^k(\alpha_1 v_1 + \alpha_2 v_2 + \cdots + \alpha_n v_n) \\
&= \alpha_1 A^k v_1 + \alpha_2 A^k v_2 + \cdots + \alpha_n A^k v_n \\
&= \alpha_1 \lambda_1^k v_1 + \alpha_2 \lambda_2^k v_2 + \cdots + \alpha_n \lambda_n^k v_n \\
&= \lambda_1^k(\alpha_1 v_1 + \alpha_2 (\lambda_2/\lambda_1)^k v_2 + \cdots + \alpha_n (\lambda_n/\lambda_1)^k v_n.
\end{aligned}$$

Hence $\lim_{k\to\infty} A^k x/\lambda_1^k = \alpha_1 v_1$, so

$$\lim_{k\to\infty} x^{(k)} = \lim_{k\to\infty} \frac{A^k x}{\pm||A^k x||} = \lim_{k\to\infty} \pm\frac{A^k x/\lambda_1^k}{||A^k x/\lambda_1^k||} = \pm\frac{\alpha_1 v_1}{||\alpha_1 v_1||} = \pm\,\hat{v}_1.$$

The power method

Example

$$A = \begin{pmatrix} 1 & 2 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}, \qquad x = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}.$$

$$y^{(0)} = Ax^{(0)} = \begin{pmatrix} 1 & 2 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 3 \\ 1 \\ 1 \end{pmatrix}; \; x^{(1)} = \frac{y^{(0)}}{\|y^{(0)}\|_\infty} = \begin{pmatrix} 1 \\ 1/3 \\ 1/3 \end{pmatrix} = \begin{pmatrix} 1.0000 \\ 0.3333 \\ 0.3333 \end{pmatrix}$$

$$y^{(1)} = Ax^{(1)} = \begin{pmatrix} 1 & 2 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} 1.0000 \\ 0.3333 \\ 0.3333 \end{pmatrix} = \begin{pmatrix} 1.6667 \\ 0.3333 \\ 1.0000 \end{pmatrix}; \quad x^{(2)} = \frac{y^{(1)}}{\|y^{(1)}\|_\infty} = \begin{pmatrix} 1.0000 \\ 0.2000 \\ 0.6000 \end{pmatrix}$$

$$y^{(2)} = Ax^{(2)} = \begin{pmatrix} 1 & 2 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} 1.0000 \\ 0.2000 \\ 0.6000 \end{pmatrix} = \begin{pmatrix} 1.4000 \\ 0.6000 \\ 1.0000 \end{pmatrix}; \quad x^{(3)} = \frac{y^{(2)}}{\|y^{(2)}\|_\infty} = \begin{pmatrix} 1.0000 \\ 0.4286 \\ 0.7143 \end{pmatrix}$$

$$y^{(3)} = Ax^{(3)} = \begin{pmatrix} 1 & 2 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} 1.0000 \\ 0.4286 \\ 0.7143 \end{pmatrix} = \begin{pmatrix} 1.8571 \\ 0.7143 \\ 1.0000 \end{pmatrix}; \quad x^{(4)} = \frac{y^{(3)}}{\|y^{(3)}\|_\infty} = \begin{pmatrix} 1.0000 \\ 0.3846 \\ 0.5385 \end{pmatrix}$$

48 / 77

The power method

Example

$$y^{(4)} = Ax^{(4)} = \begin{pmatrix} 1 & 2 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} 1.0000 \\ 0.3846 \\ 0.5385 \end{pmatrix} = \begin{pmatrix} 1.7692 \\ 0.5385 \\ 1.0000 \end{pmatrix}; \quad x^{(5)} = \frac{y^{(4)}}{\|y^{(4)}\|_\infty} = \begin{pmatrix} 1.0000 \\ 0.3043 \\ 0.5652 \end{pmatrix}$$

$$y^{(5)} = Ax^{(5)} = \begin{pmatrix} 1 & 2 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} 1.0000 \\ 0.3846 \\ 0.5385 \end{pmatrix} = \begin{pmatrix} 1.6087 \\ 0.5652 \\ 1.0000 \end{pmatrix}.$$

Estimate

$$v \approx x^{(5)} = \begin{pmatrix} 1.0000 \\ 0.3043 \\ 0.5652 \end{pmatrix}; \; \lambda \approx \mu^{(5)} = \frac{(Ax^{(5)})_{i_{\text{abs max}}}}{(x^{(5)})_{i_{\text{abs max}}}} = (Ax^{(5)})_1 = 1.6087.$$

Alternative estimate

$$v \approx \hat{x}^{(5)} = \frac{x^{(5)}}{||x^{(5)}||_2} = \begin{pmatrix} 0.8415 \\ 0.2561 \\ 0.4756 \end{pmatrix}; \; \lambda \approx \hat{\mu}^{(5)} = \frac{x^{(5)T} A x^{(5)}}{x^{(5)T} x^{(5)}} = \hat{x}^{(5)T} A \hat{x}^{(5)} = 1.6613.$$

Actual eigenvalue/vector $\lambda = 1.6841$, $v = \big(1.0000, 0.3514, 0.6216\big)^T$ or normalised $\hat{v} = \big(0.8138, 0.2859, 0.5059\big)^T$.

49 / 77

Inverse power method

Theorem If λ is an eigenvalue of A, then $1/(\lambda-\mu)$ is an eigenvalue of $(A-\mu I)^{-1}$ with the same eigenvector v. Hence if $(A-\mu I)^{-1}v=\kappa v$, then $1/(\lambda-\mu)=\kappa$, so $\lambda=\mu+1/\kappa$.

Proof. $(A-\mu I)v = Av-\mu I v=\lambda v-\mu v=(\lambda-\mu)v$,
so $(A-\mu I)^{-1}v=(\lambda-\mu)^{-1}v$. □

Inverse power method To estimate an eigenvalue $\lambda\approx\mu$, apply the power method to $(A-\mu I)^{-1}$.

Iterate
$$y^{(n)}=(A-\mu I)^{-1}x^{(n)},\ x^{(n+1)}=y^{(n)}/\pm\|y^{(n)}\|.$$

Estimate
$$\lambda\approx\lambda^{(n)}=\mu+1/\kappa^{(n)} \text{ where } \kappa^{(n)}=y_i^{(n)}/x_i^{(n)}.$$

In practise, first compute LU-factorisation $A-\mu I=L_\mu U_\mu$, and solve
$$(A-\mu I)y^{(n)}=L_\mu U_\mu y^{(n)}=x^{(n)}.$$

Can update μ at each step, setting $\mu^{(n+1)}=\mu^{(n)}+1/\kappa^{(n)}$. This speeds up convergence, but requires an expensive matrix factorisation at each step.

Inverse power method

Example
$$A=\begin{pmatrix}2&-1&1\\-1&3&-2\\1&2&3\end{pmatrix};\quad \mu=1.5.\qquad \text{Use } x^{(0)}=\begin{pmatrix}1\\0\\0\end{pmatrix}.$$

$$A-\mu I=\begin{pmatrix}0.5&-1&1\\-1&1.5&-2\\1&2&1.5\end{pmatrix}=\begin{pmatrix}1&0&0\\-2.0&1&0\\2.0&-8.0&1\end{pmatrix}\begin{pmatrix}0.5&-1.0&1.0\\0&-0.5&0.0\\0&0&-0.5\end{pmatrix}=L_\mu U_\mu$$

$$(A-\mu I)^{-1}=\begin{pmatrix}0.5&-1&1\\-1&1.5&-2\\1&2&1.5\end{pmatrix}^{-1}=\begin{pmatrix}50&28&4\\-4&-2&0\\-28&-16&-2\end{pmatrix}$$

$$y^{(0)}=(A-\mu I)^{-1}x^{(0)}=\begin{pmatrix}50.0000\\-4.0000\\-28.0000\end{pmatrix};\quad x^{(1)}=\frac{y^{(0)}}{y^{(0)}_{i_{\mathrm{abs\,max}}}}=\begin{pmatrix}1.0000\\-0.0800\\-0.5600\end{pmatrix}$$

$$y^{(1)}=(A-\mu I)^{-1}x^{(1)}=\begin{pmatrix}45.5200\\-3.8400\\-25.6000\end{pmatrix};\quad x^{(2)}=\frac{y^{(1)}}{y^{(1)}_{i_{\mathrm{abs\,max}}}}=\begin{pmatrix}1.0000\\-0.0844\\-0.5624\end{pmatrix}$$

$$y^{(2)}=(A-\mu I)^{-1}x^{(2)}=\begin{pmatrix}45.3884\\-3.8313\\-25.5255\end{pmatrix};\quad x^{(3)}=\frac{y^{(2)}}{y^{(2)}_{i_{\mathrm{abs\,max}}}}=\begin{pmatrix}1.0000\\-0.0844\\-0.5624\end{pmatrix}$$

Eigenvalue/vector $\lambda\approx(Ax^{(3)})_1=1.5220$, $v\approx(1.0000,-0.0844,-0.5624)$.

Deflation (Off-syllabus)

Deflation Suppose A has eigenvalue/vector pairs (λ_i,v_i).
If x is a vector such that $x^Tv_1=1$, and $B=A-\lambda_1 v_1 x^T$,
then B has eigenvalues 0 and $\lambda_i-\lambda_1$ for $i=2,\ldots,n$
with eigenvectors w_i satisfying $v_i=(\lambda_i-\lambda_1)w_i+\lambda_1(x'w_i)v_1$.

Wielandt deflation Define x by $x_j=a_{k,j}/\big(\lambda_1(v_1)_k\big)$ for some k. Then the k^{th} row of B is identically 0, so $(w_i)_k=0$ for all $i=2,\ldots,n$.

Orthogonalisation

Orthogonality

Normal vectors A vector v is *normal* if $\|v\|_2 := \sqrt{\sum_{i=1}^n {v_i}^2} = 1$, equivalently if $v \cdot v = 1$.

Orthogonal vectors Vectors $\{v_1, \ldots, v_n\}$ are *orthogonal* if $v_i \cdot v_j = 0$ for all $i \neq j$.

Orthonormal vectors Vectors $\{v_1, \ldots, v_n\}$ are *orthonormal* if they are orthogonal and each is normal.

Orthogonal matrices A matrix Q is *orthogonal* if $Q^{-1} = Q^T$; equivalently, if the columns of Q are ortho*normal* vectors.

Gram-Schmidt orthogonalisation

Gram-Schmidt orthogonalisation Let $x_1, \ldots, x_m$ be vectors in $\mathbb{R}^m$.

Recursively define

$$v_i = x_i - \sum_{j=1}^{i-1} \frac{x_i \cdot v_j}{v_j \cdot v_j}\, v_j = x_i - \sum_{j=1}^{i-1} (x_i \cdot u_j)\, u_j; \quad u_i = \frac{v_i}{\|v_i\|_2}.$$

Theorem The v_i are an orthogonal set, and the u_i are an orthonormal set, such that for all $k = 1, \ldots, m$,

$$\operatorname{span}\{x_1, \ldots, x_k\} = \operatorname{span}\{u_1, \ldots, u_k\} = \operatorname{span}\{v_1, \ldots, v_k\}.$$

Proof. Fix i and assume $\{u_1, \ldots, u_{i-1}\}$ are orthonormal. Then for $i > j$,

$$v_i \cdot u_j = x_i \cdot u_j - \textstyle\sum_{k=1}^{i-1} (x_i \cdot u_k) u_k \cdot u_j = x_i \cdot u_j - (x_i \cdot u_j)(u_j \cdot u_j) = 0,$$

so $u_i \cdot u_j = (v_i / \|v_i\|_2) \cdot u_j = (v_i \cdot u_j)/\|v_i\|_2 = 0$. □

Gram-Schmidt orthogonalisation

Example Apply the Gram-Schmidt orthogonalisation procedure to:

$$x_1 = \begin{pmatrix} 2 \\ -1 \\ 1 \end{pmatrix}, \quad x_2 = \begin{pmatrix} -1 \\ 3 \\ 2 \end{pmatrix}, \quad x_3 = \begin{pmatrix} 1 \\ -2 \\ 3 \end{pmatrix}.$$

$$v_1 = x_1 = \begin{pmatrix} 2 \\ -1 \\ 1 \end{pmatrix}; \quad v_2 = x_2 - \frac{x_2 \cdot v_1}{v_1 \cdot v_1} v_1 = \begin{pmatrix} -1 \\ 3 \\ 2 \end{pmatrix} - \frac{-3}{6} \begin{pmatrix} 2 \\ -1 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 5/2 \\ 5/2 \end{pmatrix};$$

$$v_3 = x_3 - \frac{x_3 \cdot v_1}{v_1 \cdot v_1} v_1 - \frac{x_3 \cdot v_2}{v_2 \cdot v_2} v_2 = \begin{pmatrix} 1 \\ -2 \\ 3 \end{pmatrix} - \frac{7}{6} \begin{pmatrix} 2 \\ -1 \\ 1 \end{pmatrix} - \frac{5/2}{25/2} \begin{pmatrix} 0 \\ 5/2 \\ 5/2 \end{pmatrix} = \begin{pmatrix} -4/3 \\ -4/3 \\ 4/3 \end{pmatrix}.$$

$$\|v_1\|_2 = \sqrt{6}; \quad \|v_2\|_2 = 5/\sqrt{2}; \quad \|v_3\|_2 = 4/\sqrt{3}.$$

$$u_1 = \frac{v_1}{\|v_1\|_2} = \frac{1}{\sqrt{6}} \begin{pmatrix} 2 \\ -1 \\ 1 \end{pmatrix}; \quad u_2 = \frac{1}{\sqrt{2}} \begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix}; \quad u_3 = \frac{1}{\sqrt{3}} \begin{pmatrix} -1 \\ -1 \\ 1 \end{pmatrix}.$$

Check orthogonality:

$$u_1 \cdot u_2 = \frac{2 \times 0 + (-1) \times 1 + 1 \times 1}{\sqrt{6} \times \sqrt{2}} = 0; \quad u_1 \cdot u_3 = \frac{-2 + 1 + 1}{\sqrt{18}} = 0; \quad u_2 \cdot u_3 = 0.$$

QR factorisation

QR factorisation In the Gram-Schmidt orthogonalisation, define $r_{i,i} = \|v_i\|_2$ and $r_{i,j} = x_j \cdot u_i = x_j \cdot v_i / \|v_i\|_2$. Then

$$v_j = x_j - \textstyle\sum_{i=1}^{j-1} r_{i,j} u_i, \text{ and } u_i = v_i / r_{i,i}$$

so

$$x_j = r_{j,j} u_j + \textstyle\sum_{i=1}^{j-1} r_{i,j}\, u_i.$$

Let

$$A = (x_1, \ldots, x_n), \quad Q = (u_1, \ldots, u_n) \text{ and } (R)_{i,j} = r_{i,j} \text{ for } i \le j.$$

Then Q is orthogonal, R is upper-triangular, and

$$A = QR.$$

57 / 77

QR factorisation

Example Compute the QR-factorisation of:

$$A = \begin{pmatrix} 2 & -1 & 1 \\ -1 & 3 & -2 \\ 1 & 2 & 3 \end{pmatrix}$$

From the Gram-Schmidt orthogonalisation,

$$Q = \begin{pmatrix} 2/\sqrt{6} & 0 & -1/\sqrt{3} \\ -1/\sqrt{6} & 1/\sqrt{2} & -1\sqrt{3} \\ 1/\sqrt{6} & 1/\sqrt{2} & 1/\sqrt{3} \end{pmatrix} = \begin{pmatrix} 0.816 & 0.000 & -0.577 \\ -0.408 & 0.707 & -0.577 \\ 0.408 & 0.707 & 0.577 \end{pmatrix}$$

$$R = \begin{pmatrix} 6/\sqrt{6} & -3/\sqrt{6} & 7/\sqrt{6} \\ 0 & 5/\sqrt{2} & 1/\sqrt{2} \\ 0 & 0 & 4/\sqrt{3} \end{pmatrix} = \begin{pmatrix} 2.449 & -1.225 & 2.858 \\ 0 & 3.536 & 0.707 \\ 0 & 0 & 2.309 \end{pmatrix}.$$

58 / 77

The QR Method

59 / 77

The QR method

The QR Method The QR method is an iterative algorithm for finding all the eigenvalues of A.

Set $A^{(0)} = A$. Iteratively find $Q^{(n)}, R^{(n)}$ such that $A^{(n)} = Q^{(n)} R^{(n)}$ and set $A^{(n+1)} = R^{(n)} Q^{(n)}$.

Theorem Assuming eigenvalues of A have distinct absolute value, $A^{(n)}$ converges to an upper-triangular matrix with the eigenvalues of A on the diagonal.

60 / 77

The QR method

Example Use the QR method to approximate eigenvalues of:

$$A = \begin{pmatrix} 4 & -1 & 1 \\ -1 & 3 & -2 \\ 1 & -2 & 3 \end{pmatrix}.$$

$$Q^{(0)} = \begin{pmatrix} -0.9428 & -0.3244 & -0.0765 \\ 0.2357 & -0.8111 & 0.5353 \\ -0.2357 & 0.4867 & 0.8412 \end{pmatrix}, \quad R^{(0)} = \begin{pmatrix} -4.2426 & 2.1213 & -2.1213 \\ 0 & -3.0822 & 2.7578 \\ 0 & 0 & 1.3765 \end{pmatrix}$$

$$A^{(1)} = R^{(0)}Q^{(0)} = \begin{pmatrix} 5.0000 & -1.3765 & -0.3244 \\ -1.3765 & 3.8421 & 0.6699 \\ -0.3244 & 0.6699 & 1.1579 \end{pmatrix}$$

$$A^{(2)} = R^{(1)}Q^{(1)} = \begin{pmatrix} 5.6667 & -0.9406 & 0.0640 \\ -0.9406 & 3.3226 & -0.1580 \\ 0.0640 & -0.1580 & 1.0108 \end{pmatrix}$$

$$A^{(3)} = \begin{pmatrix} 5.909 & -0.514 & -0.011 \\ -0.514 & 3.090 & 0.045 \\ -0.011 & 0.045 & 1.001 \end{pmatrix} \quad A^{(4)} = \begin{pmatrix} 5.977 & -0.263 & 0.002 \\ -0.263 & 3.023 & -0.014 \\ 0.002 & -0.014 & 1.000 \end{pmatrix}$$

Hence $\lambda_1 = 5.977 \pm 0.265$, $\lambda_2 = 3.023 \pm 0.277$, $\lambda_3 = 1.000 \pm 0.016$.

The QR method—Properties (Non-examinable)

Conjugacies Note

$$A^{(k)} = Q^{(k)}R^{(k)} = R^{(k-1)}Q^{(k-1)};$$

$$Q^{(k)}A^{(k+1)} = Q^{(k)}R^{(k)}Q^{(k)} = A^{(k)}Q^{(k)}.$$

Similarity Set $P^{(k)} = Q^{(0)}Q^{(1)} \cdots Q^{(k-1)}$. Then

$$P^{(k)}A^{(k)} = A\,P^{(k)}\,.$$

Equivalently,

$$A = P^{(k)}A^{(k)}\big(P^{(k)}\big)^{-1}$$

Power Set $S^{(k)} = R^{(k)} \cdots R^{(1)}R^{(0)}$. Then

$$A^k = P^{(k)}S^{(k)}.$$

The QR method—Convergence (Non-examinable)

Convergence Since $P^{(k)}$ is orthogonal and $S^{(k)}$ upper-triangular, we have

$$A^k e_1 = P^{(k)} S^{(k)} e_1 = s_{1,1}^{(k)} P^{(k)} e_1,$$
$$A^k e_2 = P^{(k)} S^{(k)} e_2 = P^{(k)} (s_{1,2}^{(k)} e_1 + s_{2,2}^{(k)} e_2) = s_{1,2}^{(k)} P^{(k)} e_1 + s_{2,2}^{(k)} P^{(k)} e_2.$$

We deduce

$$P^{(k)} e_i \in \operatorname{span}\{A^k e_1, \ldots, A^k e_i\}$$

and is orthogonal to $\{A^k e_1, \ldots, A^k e_{i-1}\}$.

Writing $e_i = \sum \alpha_{i,j} v_j$ gives

$$A^k e_i = \sum \alpha_{i,j} \lambda_j^k v_j.$$

Then

$$P^{(k)} e_1 \propto A^k e_1 = \alpha_{1,1} \lambda_1^k (v_1 + \textstyle\sum (\alpha_{1,j}/\alpha_{1,1})(\lambda_j/\lambda_1)^k v_j \sim v_1 \text{ as } k \to \infty.$$

Similarly, we see that

$$\lim_{k\to\infty} P^{(k)} e_i \in \operatorname{span}\{v_1, v_2, \ldots, v_i\}$$

and is orthogonal to $\{v_1, \ldots, v_{i-1}\}$.

The QR method (Non-examinable)

Shifted QR The QR method can be *shifted* i.e. applied to $A^{(n)} - \mu^{(n)} I$ for scalars $\mu^{(n)}$, as

$$A^{(n)} - \mu^{(n)} I = Q^{(n)} R^{(n)}; \quad A^{(n+1)} = R^{(n)} Q^{(n)} + \mu^{(n)} I\,.$$

This can accelerate convergence, especially when eigenvalues have nearly the same absolute value.

Partial QR The QR method can also be modified to find only *some* of the eigenvalues of A by iterating $Q^{(k)} R^{(k)} = A Q^{(k-1)}$, where the $Q^{(k)}$ are n-by-m matrices with orthonormal columns, and the $R^{(k)}$ are m-by-m upper-triangular matrices.

Householder matrices (Advanced)

Householder matrices A *Householder matrix* is a matrix of the form

$$H = I - 2\frac{v\,v^T}{v^T v}$$

or

$$H = I - 2ww^T \text{ where } \|w\|_2 = 1.$$

Symmetry A Householder matrix is symmetric, since

$$H^T = (I - 2ww^T)^T = I^T - 2(ww^T)^T = I^T - 2(w^T)^T w^T = I - 2ww^T.$$

Orthogonality A Householder matrix is orthogonal, since

$$H^T H = HH = (I - 2ww^T)(I - 2ww^T)$$
$$= I - 2ww^T - 2ww^T + 4ww^T ww^T = I - 4ww^T + 4w(w^T w)w^T = I$$

Theorem If $H = 1 - 2ww^T$ is a Householder matrix, then $H = H^T = H^{-1}$.

Householder matrices (Advanced)

Example Take $v = (0, 2, -1, 1)^T$, $w = v/\sqrt{6}$,

$$H = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -\frac{1}{3} & \frac{2}{3} & -\frac{2}{3} \\ 0 & \frac{2}{3} & \frac{2}{3} & \frac{1}{3} \\ 0 & -\frac{2}{3} & \frac{1}{3} & \frac{2}{3} \end{pmatrix} = \frac{1}{3}\begin{pmatrix} 3 & 0 & 0 & 0 \\ 0 & -1 & 2 & -2 \\ 0 & 2 & 2 & 1 \\ 0 & -2 & 1 & 2 \end{pmatrix}$$

Householder matrices (Advanced)

Upper-Hessenberg form For a matrix A, aim to find an orthogonal matrix Q such that $Q^T AQ$ has *upper-Hessenberg* form $A_{i,j} = 0$ for $i > j + 1$.

$$\begin{pmatrix} * & * & * & * & * \\ * & * & * & * & * \\ 0 & * & * & * & * \\ 0 & 0 & * & * & * \\ 0 & 0 & 0 & * & * \end{pmatrix}$$

Note that if A is symmetric and upper-Hessenberg, then it is *tridiagonal* $A_{i,j} = 0$ for $|i - j| > 1$.

$$\begin{pmatrix} * & * & 0 & 0 & 0 \\ * & * & * & 0 & 0 \\ 0 & * & * & * & 0 \\ 0 & 0 & * & * & * \\ 0 & 0 & 0 & * & * \end{pmatrix}$$

Upper-Hessenberg form greatly increases the efficiency of the QR method; time $O(n^2)$ per step instead of $O(n^3)$, and $O(n)$ for symmetric matrices.

Householder matrices (Advanced)

Conversion to upper-Hessenberg form First find a Householder matrix $H = H_1$ such that $H_1^T AH_1$ has first column $(H_1^T AH_1)_{i,1} = 0$ for $i \geq 3$:

1. Set $\alpha = \left(\sum_{i=2}^n a_{i,1}^2\right)^{1/2}$, $v_1 = 0, v_2 = a_{2,1} \pm \alpha, v_i = a_{i,1}$ for $i \geq 3$.
   Typically choose sign so that $v_2 = a_{2,1} + \operatorname{sgn}(a_{2,1})\alpha$.
2. Take $w = v/r$ where $r = \|v\|_2 = \left(2\alpha(\alpha \pm a_{2,1})\right)^{1/2}$.
3. Set $H = 1 - 2ww^T$, so $H_{i,1} = 0$ for $i > 2$.

Continue by applying the method to the sub-matrix $(H_1^T AH_1)_{2:n,2:n}$ to find a Householder matrix H_2 such that $H_2^T H_1^T AH_1 H_2$ has $H_{i,j} = 0$ for $j = 1, 2$ and $i > j + 1$.

Note that in practise, we compute

$$HA = (I - 2ww^T)A = A - (2w)(w^T A)$$

which takes $O(n^2)$ operations, whereas computing H first and then HA would take $O(n^3)$ operations.

Householder matrices (Advanced)

Example

$$A = \begin{pmatrix} 4 & 1 & -2 & 2 \\ 1 & 2 & 0 & 1 \\ -2 & 0 & 3 & -2 \\ 2 & 1 & -2 & -1 \end{pmatrix}.$$

$$\alpha_1 = -\operatorname{sgn}(a_{21})\big(\textstyle\sum_{i=2}^{4} a_{21}^2\big)^{1/2} = -\sqrt{9} = -3.$$

$$r_1 = (\tfrac{1}{2}\alpha_1(\alpha_1 - a_{2,1}))^{1/2} = (-\tfrac{1}{2}\alpha_1(a_{2,1} - \alpha_1))^{1/2} = \sqrt{\tfrac{1}{2}3(3+1)} = \sqrt{6}.$$

$$v_1 = \begin{pmatrix} 0 \\ 1-(-3) \\ -2 \\ 2 \end{pmatrix} = \begin{pmatrix} 0 \\ 4 \\ -2 \\ 2 \end{pmatrix}; \quad w_1 = v_1/2r_1 = \frac{1}{\sqrt{6}} \begin{pmatrix} 0 \\ 2 \\ -1 \\ 1 \end{pmatrix}.$$

$$H_1 = I - 2w_1w_1^T = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -\frac{1}{3} & \frac{2}{3} & -\frac{2}{3} \\ 0 & \frac{2}{3} & \frac{2}{3} & \frac{1}{3} \\ 0 & -\frac{2}{3} & \frac{1}{3} & \frac{2}{3} \end{pmatrix} = \frac{1}{3}\begin{pmatrix} 3 & 0 & 0 & 0 \\ 0 & -1 & 2 & -2 \\ 0 & 2 & 2 & 1 \\ 0 & -2 & 1 & 2 \end{pmatrix}$$

Householder matrices (Advanced)

Example (continued)

$$H_1^T A H_1 = \frac{1}{9}\begin{pmatrix} 36 & -27 & 0 & 0 \\ -27 & 30 & 9 & 12 \\ 0 & 9 & 15 & -12 \\ 0 & 12 & -12 & -9 \end{pmatrix} = \begin{pmatrix} 4 & -3 & 0 & 0 \\ -3 & \frac{10}{3} & 1 & \frac{4}{3} \\ 0 & 1 & \frac{5}{3} & -\frac{4}{3} \\ 0 & \frac{4}{3} & -\frac{4}{3} & -1 \end{pmatrix}$$

$$\alpha_2 = \pm\big(\textstyle\sum_{i=3}^{4} a_{i,2}^2\big)^{1/2} = -\sqrt{1^2 + (\frac{4}{3})^2} = -\frac{5}{3}; \quad (v_2)_3 = a_{2,3} - \alpha_2 = \frac{8}{3}$$

$$v_2 = \frac{4}{3}\begin{pmatrix} 0 \\ 0 \\ 2 \\ 1 \end{pmatrix}; \; w_2 = \frac{1}{\sqrt{5}}\begin{pmatrix} 0 \\ 0 \\ 2 \\ 1 \end{pmatrix}; \; H_2 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -\frac{3}{5} & -\frac{4}{5} \\ 0 & 0 & -\frac{4}{5} & \frac{3}{5} \end{pmatrix} = \frac{1}{5}\begin{pmatrix} 5 & 0 & 0 & 0 \\ 0 & 5 & 0 & 0 \\ 0 & 0 & -3 & -4 \\ 0 & 0 & -4 & 3 \end{pmatrix}$$

$$H_2^T(H_1^T A H_1)H_2 = \begin{pmatrix} 4.0000 & -3.0000 & 0 & 0 \\ -3.0000 & 3.3333 & -1.6667 & 0 \\ 0 & -1.6667 & -1.3200 & 0.9067 \\ 0 & 0 & 0.9067 & 1.9867 \end{pmatrix}$$

Givens matrices (Advanced)

Givens rotation A *Givens rotation* is an matrix of the form

$$[G_{k,l}(\alpha,\beta)]_{i,j} = \begin{cases} \alpha \text{ if } i=j=k \text{ or } i=k=l \\ \beta \text{ if } i=l \text{ and } j=k \\ -\beta \text{ if } i=k \text{ and } j=l \\ 0 \text{ otherwise.} \end{cases}$$

where $\alpha^2+\beta^2=1$. We can write $\alpha=\cos\theta$ and $\beta=\sin\theta$ for some θ.

Inverse Givens rotation matrices are orthogonal, with inverse

$$G_{k,l}(\theta)^{-1} = G_{k,l}(\theta)^T = G_{k,l}(-\theta).$$

Examples

$$G_{1,2}(\theta) = \begin{pmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad G_{2,4}(\theta) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & 0 & -\sin\theta \\ 0 & 0 & 1 & 0 \\ 0 & \sin\theta & 0 & \cos\theta \end{pmatrix}$$

Givens matrices (Advanced)

We can implement the QR algorithm for upper-Hessenberg A by a sequence of Givens rotations

$$A = QR = G_{1,2}(\theta_1)\, G_{2,3}(\theta_2)\, G_{3,4}(\theta_3)\; R$$

so

$$R = G_{3,4}^{-1}(\theta_3)\, G_{2,3}^{-1}(\theta_2)\, G_{1,2}^{-1}(\theta_1)\; A$$

and

$$\begin{aligned} RQ &= R\; G_{1,2}(\theta_1)\, G_{2,3}(\theta_2)\, G_{3,4}(\theta_3) \\ &= G_{3,4}^{-1}(\theta_3)\, G_{2,3}^{-1}(\theta_2)\, G_{1,2}^{-1}(\theta_1)\; A\; G_{1,2}(\theta_1)\, G_{2,3}(\theta_2)\, G_{3,4}(\theta_3) \end{aligned}$$

Note that pre-multiplying A by G_{kl} (or G_{kl}^{-1}) changes rows k and l to linear combinations of each other, and leaves all other rows unchanged.
Likewise, post-multiplying by G_{kl} changes columns k and l, and leaves all other columns unchanged.

Givens matrices (Advanced)

Example

$$A = \begin{pmatrix} 4 & 1 & 7 \\ 3 & 7 & 9 \\ 0 & 12 & 2 \end{pmatrix}$$

Take Givens rotation (with $r_1 = (a_{11}^2 + a_{21}^2)^{1/2}$)

$$G_1 = \begin{pmatrix} a_{11}/r_1 & -a_{21}/r_1 & 0 \\ a_{21}/r_1 & a_{11}/r_1 & 0 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} \frac{4}{5} & -\frac{3}{5} & 0 \\ \frac{3}{5} & \frac{4}{5} & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

$$G_1^{-1}A = \begin{pmatrix} \frac{4}{5} & \frac{3}{5} & 0 \\ -\frac{3}{5} & \frac{4}{5} & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 4 & 1 & 7 \\ 3 & 7 & 9 \\ 0 & 12 & 2 \end{pmatrix} = \begin{pmatrix} 5 & 5 & 11 \\ 0 & 5 & 3 \\ 0 & 12 & 2 \end{pmatrix}.$$

$$G_2 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \frac{5}{13} & -\frac{12}{13} \\ 0 & \frac{12}{13} & \frac{5}{13} \end{pmatrix}; \qquad R = G_2^{-1}(G_1^{-1}A) = \begin{pmatrix} 5 & 5 & 11 \\ 0 & 13 & 3 \\ 0 & 0 & -2 \end{pmatrix}.$$

Givens matrices (Advanced)

Example (continued)

$$\begin{aligned} RQ = RG_1G_2 &= (RG_1)G_2 \\ &= \begin{pmatrix} 5 & 5 & 11 \\ 0 & 13 & 3 \\ 0 & 0 & -2 \end{pmatrix} \begin{pmatrix} \frac{4}{5} & -\frac{3}{5} & 0 \\ \frac{3}{5} & \frac{4}{5} & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & \frac{5}{13} & -\frac{12}{13} \\ 0 & \frac{12}{13} & \frac{5}{13} \end{pmatrix} \\ &= \begin{pmatrix} 7 & 1 & 11 \\ 7\frac{4}{5} & 10\frac{2}{5} & 3 \\ 0 & 0 & -2 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & \frac{5}{13} & -\frac{12}{13} \\ 0 & \frac{12}{13} & \frac{5}{13} \end{pmatrix} \\ &= \begin{pmatrix} 7 & 10\frac{7}{13} & 3\frac{4}{13} \\ 7\frac{4}{5} & 6\frac{10}{13} & -9\frac{29}{65} \\ 0 & -1\frac{11}{13} & -\frac{10}{13} \end{pmatrix} = \begin{pmatrix} 7.000 & 10.538 & 3.308 \\ 7.800 & 6.769 & -8.446 \\ 0 & -1.846 & -0.769 \end{pmatrix} \end{aligned}$$

Givens matrices (Advanced)

If A is tridiagonal, then $A = QR$ with $r_{i,j} = 0$ for $i > j$ or $j > i + 2$, so R is banded with three nontrivial bands on the main diagonal and above ($3n$ nonzero entries).

However, Q is upper-Hessenberg, and has $n^2/2$ nonzero entries, even though it is the product of $n - 1$ Givens rotations, which are sparse.

We therefore perform the QR-method using Givens rotations, and do not directly construct Q.

Eigenvalue condition number (Non-examinable)

Eigenvalue condition numbers Suppose λ is a simple eigenvalue of A, $Ax = \lambda x$ and $A^T y = \lambda y$ with $||x||_2 = ||y||_2 = 1$. Suppose $||F||_2 = 1$. Then if $\lambda(\epsilon)$ continuous to $A + \epsilon F$, we have $|\lambda'(\epsilon)| = |Y^T F x| / |Y^T x| \leq 1/|Y \cdot x|$, with the bound attained for $F = yx^T$. Define $s(\lambda) = |y^T x|$ the condition of the eigenvalue.

[From Golub & Van Loon]

Eigenvalue conditioning (Non-examinable)

Eigenvalue conditioning If μ is eigenvalue of perturbation $A + E$ of a nondefective matrix A, then

$$|\mu - \lambda_k| \leq \mathrm{cond}_2(V) ||E||_2$$

where λ_k is closest eigenvalue of A to μ, and V is the matrix of eigenvectors of A.

[From Michael T. Heath: Scientific Computing: An Introductory Survey]