

Matrix calculus



Systems of Linear equations

• The problem:

Solve: $\underline{A} \underline{x} = \underline{b}$ with $\underline{A} = \begin{bmatrix} a_{11} & \dots & a_{1n} \\ \vdots & & \vdots \\ a_{n1} & \dots & a_{nn} \end{bmatrix}$; $\underline{b} = \begin{bmatrix} b_1 \\ \vdots \\ b_n \end{bmatrix}$

Matrix inversion: \underline{A}^{-1} known, then $\underline{x} = \underline{A}^{-1} \underline{b}$

• Gaussian Elimination:

Idea: - rearrangements + linear combinations of equations, such that:

$$\underline{A} \underline{x} = \underline{b}$$

goes over into:

$$\underline{R} \underline{x} = \underline{c}; \quad \underline{R} = \begin{bmatrix} r_{11} & \dots & r_{1n} \\ & \ddots & \vdots \\ 0 & & r_{nn} \end{bmatrix}$$

upper triangular matrix

- back(ward) substitution: ($r_{ii} \neq 0$)

$$x_i = \frac{c_i - \sum_{k=i+1}^n r_{ik} x_k}{r_{ii}} \quad \text{for } i = n, n-1, \dots, 1$$

Continue process => $\underline{\underline{L}} \underline{\underline{R}} = \underline{\underline{P}} \underline{\underline{A}}$ (Triangular decompositions)

$$\underline{\underline{L}} = \begin{bmatrix} 1 & & & \\ t_{21} & \ddots & & \\ & t_{32} & \ddots & \\ & & & 1 \end{bmatrix}$$

Lower triangular matrix

$$\underline{\underline{R}} = \begin{bmatrix} + & a_{11} & \dots & t_{1n} \\ & \ddots & & \\ 0 & & & \\ & & & t_{nn} \end{bmatrix}$$

upper triangular matrix

$$\underline{\underline{P}} = \underline{\underline{P}}_{n-1} \underline{\underline{P}}_{n-2} \dots \underline{\underline{P}}_1$$

product of all permutations

Example:

$$\begin{bmatrix} 3 & 1 & 6 \\ 2 & 1 & 3 \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 2 \\ 7 \\ 4 \end{bmatrix}$$

Pivot

$$\left[\begin{array}{ccc|c} 3 & 1 & 6 & 2 \\ 2 & 1 & 3 & 7 \\ 1 & 1 & 1 & 4 \end{array} \right]$$

$$\rightarrow \left[\begin{array}{ccc|c} 3 & 1 & 6 & 2 \\ \frac{2}{3} & \frac{1}{3} & -1 & \frac{17}{3} \\ \frac{1}{3} & \frac{2}{3} & -1 & \frac{10}{3} \end{array} \right]$$

$$\rightarrow \left[\begin{array}{ccc|c} 3 & 1 & 6 & 2 \\ \frac{1}{3} & \frac{2}{3} & -1 & \frac{10}{3} \\ \frac{2}{3} & \frac{1}{2} & -\frac{1}{2} & 4 \end{array} \right]$$

thus:

$$\begin{bmatrix} 3 & 1 & 6 \\ 0 & \frac{2}{3} & -1 \\ 0 & 0 & -\frac{1}{2} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 2 \\ \frac{10}{3} \\ 4 \end{bmatrix}$$

d.h.

$$\begin{cases} x_3 = -8 \\ x_2 = -7 \\ x_1 = 19 \end{cases}$$

$$\underline{\underline{P}} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}; \quad \underline{\underline{P}} \underline{\underline{A}} = \begin{bmatrix} 3 & 1 & 6 \\ 1 & 1 & 1 \\ 2 & 1 & 3 \end{bmatrix}$$

$$\underline{\underline{L}} = \begin{bmatrix} 1 & 0 & 0 \\ \frac{1}{3} & 1 & 0 \\ \frac{2}{3} & \frac{1}{2} & 1 \end{bmatrix}; \quad \underline{\underline{R}} = \begin{bmatrix} 3 & 1 & 6 \\ 0 & \frac{2}{3} & -1 \\ 0 & 0 & -\frac{1}{2} \end{bmatrix}$$

Conversely:

$\underline{\underline{L}}, \underline{\underline{R}}, \underline{\underline{P}}$ known

$$\underline{\underline{A}} \underline{\underline{x}} = \underline{\underline{b}} \rightarrow \underline{\underline{P}} \underline{\underline{A}} \underline{\underline{x}} = \underline{\underline{L}} \underline{\underline{R}} \underline{\underline{x}} = \underline{\underline{P}} \underline{\underline{b}}$$

Solve two triangular systems:

$$\begin{bmatrix} \underline{\underline{L}} \underline{\underline{u}} = \underline{\underline{P}} \underline{\underline{b}} \\ \underline{\underline{R}} \underline{\underline{x}} = \underline{\underline{u}} \end{bmatrix}$$

Cholesky Decomposition

Positive definite matrix:

$$Q(\underline{x}) = \underline{x}^T \underline{A} \underline{x} = \sum_{i=1}^n \sum_{k=1}^n a_{ik} x_i x_k \geq 0 ; \quad \forall \underline{x} \neq 0$$

$$= 0 ; \quad \underline{x} = \underline{0}$$

\underline{A} symmetric matrix

Idea: Reduce $Q(\underline{x})$ to a sum of quadratic terms

$$Q(\underline{x}) = \sum_{i=1}^n \sum_{k=1}^n a_{ik} x_i x_k = \sum_{k=1}^n \left[\sum_{i=k}^n l_{ik} x_i \right]^2$$

First step: $Q(\underline{x}) = a_{11} x_1^2 + 2 \sum_{i=2}^n a_{i1} x_1 x_i + \sum_{i=2}^n \sum_{k=2}^n a_{ik} x_i x_k$

$$= \left[\underbrace{\sqrt{a_{11}}}_{l_{11}} x_1 + \sum_{i=2}^n \underbrace{\frac{a_{i1}}{\sqrt{a_{11}}}}_{l_{i1}} x_i \right]^2 + \sum_{i=2}^n \sum_{k=2}^n \underbrace{\left(a_{ik} - \frac{a_{i1} a_{1k}}{a_{11}} \right)}_{\substack{(1) \\ a_{ik}}} x_i x_k$$

$Q^{(1)}(\underline{x}^{(1)})$
 $\underline{x}^{(1)} = (x_2, \dots, x_n)$

K-th step:

$$l_{kk} = \sqrt{a_{kk}^{(k-1)}} ; \quad l_{ik} = \frac{a_{ik}^{(k-1)}}{l_{kk}} ; \quad i = k+1, \dots, n$$

$$a_{ij}^{(k)} = a_{ij}^{(k-1)} - l_{ik} l_{jk} ; \quad i, j = k+1, \dots, n$$

$$\Rightarrow \underline{L} = \begin{pmatrix} l_{11} & \dots & 0 \\ l_{21} & l_{22} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ l_{n1} & \dots & \dots & l_{nn} \end{pmatrix}$$

Method:

(a) $\underline{A} = \underline{L} \underline{L}^T$ (Cholesky-decomp.)

(b) $\underline{L} \underline{c} - \underline{b} = \underline{0}$ (forward insertion)

(c) $\underline{L}^T \underline{x} + \underline{c} = \underline{0}$ (back insertion)

• Least squares problems:

We want to determine certain constants $x_1 \dots x_n$ by Sampling $y_k = f(\underbrace{z_k}_{\text{"conditions"}}; x_1 \dots x_n)$; $k = 1, \dots, m$
 $m \geq n$

Prescription (Gauss):

Minimize

$$\mathbb{I} = \sum_{k=1}^m (y_k - f_k(x_1 \dots x_n))^2 = \underline{y}^T \underline{r}$$

residuals

- Linear least squares:

$$\begin{bmatrix} f_1(x_1 \dots x_n) \\ \vdots \\ f_m(x_1 \dots x_n) \end{bmatrix} = \underline{C} \underline{x}$$

$$\min_{\underline{x}} \|\underline{y} - \underline{C}\underline{x}\| \rightarrow \underbrace{\underline{C}^T \underline{C}}_{\underline{A}} \underline{x} - \underbrace{\underline{C}^T \underline{y}}_{\underline{b}} = \underline{0}$$

Solution:

$$\underline{A} = \underline{L} \underline{L}^T \text{ (Cholesky - decomp.)}$$

$$\underline{L} \underline{y} = \underline{b} \text{ (forward insertion)}$$

$$\underline{L}^T \underline{x} + \underline{y} = \underline{0} \text{ (back insertion)}$$

$$\underline{r} = \underline{C} \underline{x} + \underline{d} \text{ (residuals)}$$

- Nonlinear least squares:

$$\underline{f}(\underline{x}) = \underline{f}(\underline{x}^0) + \underline{Df}(\underline{x}^0)(\underline{x} - \underline{x}^0) + \dots$$

(Taylor expansion)

Gauss-Newton Algorithm:

$$\underline{x}^{(0)}, \underline{x}^{(1)}, \dots, \underline{x}^{(i)} \text{ (iterate...)}$$

(a) Determine $\underline{s}^{(i)}$ from: $\min_{\underline{s}} \|\underline{y} - \underline{f}(\underline{x}^{(i)}) - \underline{Df}(\underline{x}^{(i)})\underline{s}\|^2$

(b) Let $\varphi(\alpha) = \|\underline{y} - \underline{f}(\underline{x}^{(i)} + \alpha \underline{s}^{(i)})\|^2$ and k smallest intge with $\varphi(2^{-k}) < \varphi(0)$

(c) Define: $\underline{x}^{(i+1)} = \underline{x}^{(i)} + 2^{-k} \underline{s}^{(i)}$

Iterative methods:

Generate a sequence of vectors $\underline{x}^{(0)} \rightarrow \underline{x}^{(1)} \rightarrow \underline{x}^{(2)} \rightarrow \dots$ which converge toward the desired solution \underline{x} of $\underline{A}\underline{x} = \underline{b}$

$$\left. \begin{aligned} \underline{x}^{(i+1)} &= \underline{\phi}(\underline{x}^{(i)}) \\ \underline{B}\underline{x} + (\underline{A} - \underline{B})\underline{x} &= \underline{b} \\ \uparrow &\text{arbitrary matrix} \end{aligned} \right\} \underline{B}\underline{x}^{(i+1)} + (\underline{A} - \underline{B})\underline{x}^{(i)} = \underline{b}$$

solve for $\underline{x}^{(i+1)}$ (easy!)

standard decomposition:

$$\underline{A} = \underline{D} - \underline{E} - \underline{F}$$

$$\underline{D} = \begin{bmatrix} a_{11} & & 0 \\ & \ddots & \\ 0 & & a_{nn} \end{bmatrix}; \quad \underline{E} = - \begin{bmatrix} 0 & & 0 \\ a_{21} & & \\ & \ddots & \\ a_{n1} & & a_{nn} \end{bmatrix}$$

$$\underline{F} = \begin{bmatrix} 0 & a_{12} & \dots & a_{1n} \\ & \ddots & & \\ 0 & & & a_{n,n-1} \\ & & & 0 \end{bmatrix}$$

Jacobi method: $\underline{B} = \underline{D}$;

$$a_{jj} x_j^{(i+1)} + \sum_{k \neq j} a_{jk} x_k^{(i)} = b_j; \quad j=1, \dots, n; \quad i=0, 1, \dots$$

Gauss-Seidel method: $\underline{B} = \underline{D} - \underline{E}$

$$\sum_{k < j} a_{jk} x_k^{(i+1)} + a_{jj} x_j^{(i+1)} + \sum_{k > j} a_{jk} x_k^{(i)} = b_j$$

$j=1, 2, \dots, n; \quad i=0, 1, \dots$

Use: $\underline{x}^{(i+1)} = \underline{x}^{(i)} - \underline{B}^{-1}(\underline{A}\underline{x}^{(i)} - \underline{b}) = (\underline{I} - \underline{B}^{-1}\underline{A})\underline{x}^{(i)} + \underline{B}^{-1}\underline{b}$

Relaxation methods:

$$\underline{B}(\omega) = \frac{1}{\omega} \underline{D} (\underline{I} - \omega \underline{D}^{-1} \underline{E})$$

guess!

$\omega > 1$ overrelaxation; $\omega = 1$ underrelaxation

Eigen values

• The problem:

(A: real $n \times n$ matrix)

$$\underline{A} \underline{x}_k = \lambda_k \underline{x}_k$$

$$\Updownarrow$$

$$(\underline{A} - \lambda_k \underline{I}) \underline{x}_k = \underline{0}$$

as $\underline{x}_k \neq \underline{0} \Rightarrow \det(\underline{A} - \lambda_k \underline{I}) = |\underline{A} - \lambda_k \underline{I}| \stackrel{!}{=} 0$

or: $P(\lambda_k) = (-)^n |\underline{A} - \lambda_k \underline{I}|$

$$= \lambda_k^n + p_{n-1} \lambda_k^{n-1} + p_{n-2} \lambda_k^{n-2} + \dots + p_1 \lambda_k + p_0 \stackrel{!}{=} 0$$

(characteristic polynomial)

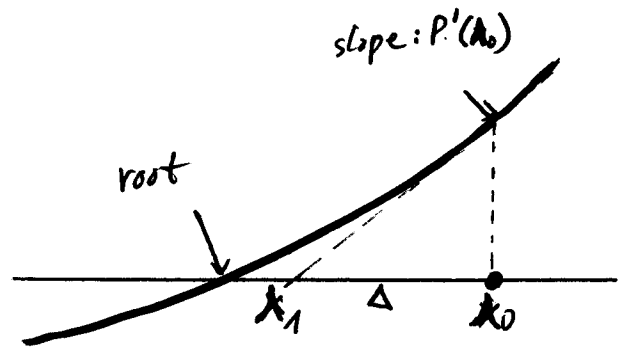
• Naive approach:

- Evaluate: $|\underline{A} - \lambda \underline{I}| = (-)^n P(\lambda)$

- Determine roots of $P(\lambda)$, i.e. $P(\lambda) = 0$.

Standard method:

Newton's method



$$\lambda_{i+1} = \lambda_i - \frac{P(\lambda_i)}{\underbrace{P'(\lambda_i)}_{\Delta}}$$

\Rightarrow inefficient, subject to massive rounding errors.

• Iteration method ("power method")

- Multiplication: $\underline{v} = \underline{A} \underline{y}$ (\underline{y} : arbitrary vector)
 $\Rightarrow \underline{v}$ will differ from \underline{y} in length
 and direction

- Repeated (pre-) multiplication:

$$\underline{y}_{i+1} = \underline{A} \underline{y}_i = \underline{A}^{i+1} \underline{y}_0$$

with: $\underline{y}_0 = b_1 \underline{x}_1 + b_2 \underline{x}_2 + b_3 \underline{x}_3 + \dots$ (expansion using eigenvectors \underline{x}_i)

$$\Rightarrow \underline{y}_n = \lambda_1^n \left[b_1 \underline{x}_1 + \left(\frac{\lambda_2}{\lambda_1}\right)^n b_2 \underline{x}_2 + \left(\frac{\lambda_3}{\lambda_1}\right)^n b_3 \underline{x}_3 + \dots \right]$$

↑
largest eigenvalue

$$\rightarrow C_n \underline{x}_1 \text{ (for large } n)$$

\Rightarrow method to determine largest eigenvector
 and eigenvalue(s) via: $\lambda_1 = \frac{\underline{y}_n^T \underline{A} \underline{y}_n}{\underline{y}_n^T \underline{y}_n}$

- Shifting:

$$\underline{A} \underline{x} = \lambda \underline{x} = (\lambda - s) \underline{x} + s \underline{x}$$

$$\text{or: } \underbrace{(\underline{A} - s \underline{I})}_{\underline{B}} \underline{x} = \underbrace{(\lambda - s)}_{\mu} \underline{x} \quad \left. \vphantom{\underline{B}} \right\} \text{ same eigenvalue problem}$$

\Rightarrow accelerate convergence: if $|\lambda_i| \approx |\lambda_1|$, iterative method converges very slowly
nevertheless: $|\lambda_i + c| \neq |\lambda_1 + c|$, leading to much better convergence.

- Other eigenvalues: (Hotelling)

$$\underline{B} = \underline{A} - \lambda_1 \underline{x}_1 \underline{x}_1^T \text{ with } \underline{x}_1^T \underline{x}_1 = 1 \Rightarrow \underline{B} \underline{x}_i = \lambda_i \underline{x}_i$$

$i = 2, 3, \dots$
 (deflation)

\Rightarrow plagued by rounding errors.

• The grand strategy:

- Similarity transformations:

$$\underline{C} = \underline{T}^{-1} \underline{A} \underline{T}$$

assume: $\underline{A} \underline{x} = \lambda \underline{x} \implies \underline{T}^{-1} \underline{A} \underline{T} \underline{y} = \lambda \underline{T}^{-1} \underline{T} \underline{y} = \lambda \underline{y}$

\implies eigenvalues are preserved

- Theorem (Schur): For every matrix \underline{A} , there exists a unitary matrix \underline{Q} , such that:

$$\underline{Q}^T \underline{A} \underline{Q} = \underline{T} = \text{diag}(\lambda_1, \lambda_2, \dots, \lambda_n) + \underline{N}$$

upper triang. matrix

- Eigenvalues of triangular matrices:

$$\underline{L} = \begin{pmatrix} \lambda_{11} & & 0 \\ & \ddots & \\ 0 & & \lambda_{nn} \end{pmatrix}; \quad \underline{R} = \begin{pmatrix} r_{11} & & r_{1n} \\ & \ddots & \\ 0 & & r_{nn} \end{pmatrix}$$

$$\lambda(\underline{L}) = \{\lambda_{11}, \lambda_{22}, \dots, \lambda_{nn}\}; \quad \lambda(\underline{R}) = \{r_{11}, r_{22}, \dots, r_{nn}\}$$

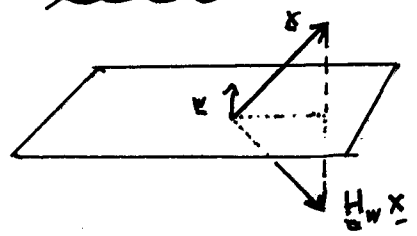
- Standard tools:

Plane rotations (Givens):

$$\underline{R}(\varphi) = \begin{bmatrix} \cos \varphi & -\sin \varphi & & 0 \\ \sin \varphi & \cos \varphi & & \\ & & \ddots & \\ 0 & & & 1 \end{bmatrix}$$

$\underline{B} = \underline{R}^T \underline{A} \underline{R}$
 \implies the same, up to columns i, j and rows i, j .

Reflections (Householder):



$$\underline{H}_w = \underline{I} - 2 \underline{w} \underline{w}^T \quad \text{with } \|\underline{w}\| = 1$$
$$\underline{y} = \underline{H}_w \underline{x} = (\underline{I} - 2 \underline{w} \underline{w}^T) \underline{x} = \underline{x} - 2 \langle \underline{w}, \underline{x} \rangle \underline{w}$$

(Reflection by plane whose normal is \underline{w})

• The QR - algorithm:

- Theorem: Every $n \times n$ matrix \underline{A} may be factored as follows:

$$\underline{A} = \underline{Q} \underline{R}$$

where:

- \underline{Q} : orthogonal matrix
- \underline{R} : triangular matrix

- Implementation: Employ special sequence of Givens rotations or Householder transformations.

- Eigenvalues (Francis):

Set $\underline{A}_0 = \underline{A}$; iterate:

$$\underline{A}_k = \underline{Q}_k \underline{R}_k; \underline{A}_{k+1} = \underline{R}_k \underline{Q}_k \quad (k=1, 2, \dots)$$

\Rightarrow for $k \rightarrow \infty$, \underline{A}_k converges to triangular matrix \underline{R} :

$$\underline{A}_{k+1} = \underline{R}_k \underline{Q}_k = \underline{Q}_k^T (\underline{Q}_k \underline{R}_k) \underline{Q}_k = \underline{Q}_k^T \underline{A}_k \underline{Q}_k$$

$$\underline{Q} = \underline{Q}_0 \underline{Q}_1 \dots \underline{Q}_M \Rightarrow \underline{R} = \underline{Q}^T \underline{A} \underline{Q}$$

\Rightarrow Eigenvalues: $\underline{R} \underline{y} = \lambda \underline{y}$

- Eigen vectors: set $\underline{y} = \underline{Q}^T \underline{x}$, then from $\underline{R} \underline{y} = \lambda \underline{y}$

we obtain: $\underline{A} \underline{x} = \lambda \underline{x} \Rightarrow \underline{x} = \underline{Q} \underline{y}$

- Accelerate convergence (shift):

$$\underline{A}_k - \sigma_k \underline{I} = \underline{Q}_k \underline{R}_k; \underline{A}_{k+1} = \underline{R}_k \underline{Q}_k + \sigma_k \underline{I}$$

$(k=1, 2, \dots); \sigma_k = a_{nn}^{(k)}$

- In practice: Reduce workload by transforming \underline{A} into tridiagonal matrix or Hessenberg matrix before applying QR-algorithm!

