Computer only + − × ÷

Babylonian Method:

$x_{k+1} = \frac{1}{2}\left(x_k + \frac{c}{x_k}\right)$ obtain $\sqrt{c}$.

Fixed representation: xxxx.xxxx Antidated.

Floating point: $\pm \underbrace{M}_{\text{Mantissa}} \times 2^{\overbrace{E}^{\text{Exponent}}}$, $M \in [1,2)$ Defines floating point number.

Every multiplication by 2 moves decimal to the right by 1 place.

single precision $1+8+23=32$

double precision $1+11+52=64$

$\frac{1-2^{11}}{1-2} = 2^{11}-1 = 2047 - 1023 = 1024$

To represent negative exponents:

$X = \underbrace{\pm}_{1\text{bit}} \underbrace{M}_{\substack{1.\underbrace{0\cdots0}_{52\text{ bits}}}} \times 2^{\underbrace{E-1023}_{11\text{ bits}}}$

$[-1023, 1024]$

$E = (01111111111) \Rightarrow 2^0$

$E = (10000000000) \Rightarrow 2^1$

Special: Inf $\Rightarrow M=0,\ E=1\cdots1$

$\pm 0 \Rightarrow M=0,\ E=0\cdots0$

NaN $=\sqrt{-1} \Rightarrow M\neq0,\ E=1\cdots1$.

Why not a normal num?

Rounding:

$ARE = |round(x) - x|$

$RRE = \frac{|round(x)-x|}{|x|}$

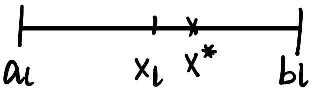
$round(a+b) = (a+b)(1+\delta)$, $|\delta| < \epsilon$ machine precision

$\epsilon = (0|01\cdots1|0\cdots00) - (0|01\cdots1|0\cdots01)$

$= 2^{-52} \approx 2.2\times10^{-16}$

$\frac{|round(a+b)-(a+b)|}{|a+b|} \leq \epsilon$

$|round(a+b) - (a+b)| \leq 2\max(|a|,|b|)\epsilon$.

Bisection: 

$$|x_l - x^*| \le \frac{b_l - a_l}{2} = \frac{1}{2}\frac{b_0 - a_0}{2^l} = \frac{L}{2^{l+1}} < \epsilon .$$

$$l > 1 + \log_2 \frac{L}{\epsilon}$$

$e_{l+1} \le \frac{1}{2} e_l$, $e_{k+1} = \frac{1}{2} e_k$

$$\lim_{k\to\infty} \frac{e_{k+1}}{e_k} = \frac{1}{2}$$

Secant Method:

$$x_{k+1} = x_k - f_k \frac{x_k - x_{k-1}}{f_k - f_{k-1}}$$

$$\lim_{k\to\infty} \frac{e_{k+1}}{e_k^{1.62}} = C$$

Newton's Method:

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}$$

$$e_{k+1} \approx A e_k^2$$

$$\lim_{k\to\infty} \frac{|\xi - x_{k+1}|}{|\xi - x_k|^2} = \frac{1}{2}\left|\frac{f''(\xi)}{f'(\xi)}\right|$$ defines quadratic convergence.

failures: ① $f' = 0$ at root, $\frac{f''(x)}{f'(y)}$ not bounded.

② intial guess horizontal / oscillating

$$\lim_{k\to\infty} \frac{e_{k+1}}{e_k^{\alpha}} = \mu$$, α is order of convergence

When $\alpha = 1$, μ is rate of convergence.

For $\lim_{k\to\infty} \frac{e_{k+1}}{e_k} = \mu < 1$. set $\rho = -\log_{10} \mu :=$ asmptopic rate of convergence

$\rho =$ # correct decimel digits gained on successive iterations.

Vectors and Matrix norms:

Def: (1). $\|\vec{x}\| \geq 0$, $\|\vec{x}\| = 0$ iff $\vec{x} = \vec{0}$

(2) $\|\alpha \vec{x}\| = |\alpha| \|\vec{x}\|$, $\alpha \in \mathbb{C}$

(3). $\|\vec{u} + \vec{v}\| \leq \|\vec{u}\| + \|\vec{v}\|$ triangular inequality.

$\|\vec{u}\|_\infty = \max_i |u_i|$

$\|\vec{u}\|_1 = \sum_i |u_i|$

$\|\vec{u}\|_p = \left(\sum_i |u_i|^p \right)^{\frac{1}{p}}$

Def: (1). $\|A\| \geq 0$, $\|A\| = 0$ iff $A = 0$

(2). $\|\alpha A\| = |\alpha| \, \|A\|$, $\alpha \in \mathbb{C}$

(3) $\|A+B\| \leq \|A\| + \|B\|$

[(4) $\|AB\| \leq \|A\| \cdot \|B\|$]

$\|\cdot\|$ is <u>any</u> vector norm, induced matrix norm:

$$\|A\| = \max_{\|\vec{u}\|=1} \|A\vec{u}\| = \max_{\vec{u} \neq 0} \frac{\|A\vec{u}\|}{\|\vec{u}\|} \Rightarrow \|A\vec{u}\| \leq \|A\| \cdot \|\vec{u}\|$$

$\|A\|_1 = \max_{j=1,\dots,n} \sum_{i=1}^{m} |a_{ij}|$ largest column absolute value sum

[Proof]

$\|A\|_\infty = \max_{i=1,\dots,m} \sum_{j=1}^{n} |a_{ij}|$ largest row

$\|A\|_2 = \sqrt{\max_j \lambda_j}$, λ_j is eigenvalue of $A^T A$.

Condition number:

$C(x) \approx \dfrac{|y-y'|}{|x-x'|} = |f'(x)|$ Absolute

$K(x) = \left| \dfrac{y-y'}{y} \right| \left| \dfrac{x}{x-x'} \right| = \left| \dfrac{x f'(x)}{f(x)} \right|$ Relative

Interpretation:

$$\|\vec{x} - \vec{x}'\| = \|A^{-1}\vec{b} - A^{-1}\vec{b}'\| \leq \underbrace{\|A^{-1}\|}_{\text{absolute CN of } A} \|\vec{b} - \vec{b}'\|$$

$$\frac{\|\vec{x} - \vec{x}'\|}{\|\vec{x}\|} \leq \underbrace{\|A\| \|A^{-1}\|}_{\text{relative CN of } A} \frac{\|\vec{b} - \vec{b}'\|}{\|\vec{b}\|}$$

functions:

$\|f\|_\infty = \max_{x \in [a,b]} |f(x)|$

$\|f\|_2 = \sqrt{\int_a^b |f(x)|^2 dx}$

$\|f\|_1 = \int_a^b |f(x)| dx.$

$\|f\|_{2,w} = \sqrt{\int_a^b |f(x)|^2 w(x) dx}.$

Solving nonlinear systems:

$$\left.\begin{array}{l} f_1(x_1,\dots,x_n)=0 \\ f_2(x_1,\dots,x_n)=0 \\ \text{— — — —} \\ f_n(x_1,\dots,x_n)=0 \end{array}\right\} \quad \vec{f}(\vec{x})=\vec{0} \quad w/ \quad \vec{f}=\begin{pmatrix} f_1(\vec{x}) \\ \vdots \\ f_n(\vec{x}) \end{pmatrix} \quad \vec{x}=\begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}$$

Multivariable Taylor Series:

$$f(\vec{x})=f(\vec{y})+\sum_{l=1}^{n}\frac{\partial f(\vec{y})}{\partial x_l}(x_l-y_l)+\frac{1}{2!}\sum_{k=1}^{n}\sum_{l=1}^{n}\frac{\partial^2 f(\vec{y})}{\partial x_k \partial x_l}(x_k-y_k)(x_l-y_l)+\frac{1}{3!}\sum_{k=1}^{n}\sum_{m=1}^{n}\sum_{l=1}^{n}\frac{\partial^3 f}{\partial x_k \partial x_m \partial x_l}(\vec{y})(x_k-y_k)(x_m-y_m)(x_l-y_l)+\cdots$$

Extend to $\vec{f}(\vec{x})$

$$\vec{f}(\vec{x})=\vec{f}(\vec{y})+\underbrace{J(\vec{y})}_{\text{Jacobian}}(\vec{x}-\vec{y})+(\vec{x}-\vec{y})^T\underbrace{\frac{Q(\vec{y})}{2!}}_{\text{Tensor}}(\vec{x}-\vec{y})+\cdots$$

$$J(\vec{y})=\begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \ddots & & \\ \vdots & & & \\ \frac{\partial f_n}{\partial x_n} & & & \frac{\partial f_n}{\partial x_n} \end{pmatrix}$$

i-th entry $(\vec{x}-\vec{y})^T\frac{Q(\vec{y})}{2!}(\vec{x}-\vec{y})$ is $\frac{1}{2}(\vec{x}-\vec{y})^T\underbrace{H_i(\vec{y})}_{\text{Hessian}}(\vec{x}-\vec{y})$ [?]

Multivariable Newton's Method:

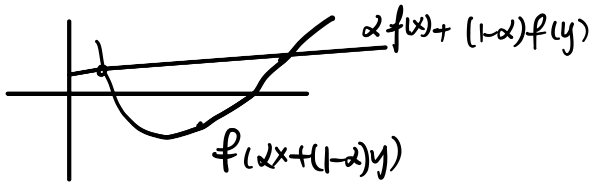
$$\boxed{\vec{x}_{k+1}=\vec{x}_k-J^{-1}(\vec{x}_k)\vec{f}(\vec{x}_k)} \qquad \|\vec{x}_{k+1}-\vec{\xi}\|_2 \approx \|\vec{x}_k-\vec{\xi}\|_2^2$$

If $f(\xi)=0$, may have singular Jacobian Matrix.

Truncating $\vec{f}(\vec{x})=\vec{f}(\vec{y})+J(\vec{y})(\vec{x}-\vec{y})$ $\qquad f(x)=f(y)+f'(x_0)(x-x_0)$

Optimization; find max/min.

convex: f is convex on $[a,b]$ if $f(\alpha x+(1-\alpha)y) \leq \alpha f(x)+(1-\alpha)f(y)$ for all $\alpha \in [0,1]$



Hessian of a strictly convex function is positive definite.

$$\vec{x}^* = \underset{\vec{x}\in X}{\operatorname{argmin}} f(x_1, x_2, \cdots, x_n) \qquad \nabla f = \vec{0}$$

$$J_{jk} = \frac{\partial f_j}{\partial x_k}, \quad f_j = \frac{\partial f}{\partial x_j} \Rightarrow J_{jk} = \frac{\partial^2 f}{\partial x_j \partial x_k} = H_{ij} \leftarrow \text{the Hessian.}$$

$$\boxed{\vec{x}_{k+1} = \vec{x}_k - H^{-1}(\vec{x}_k)\nabla f(\vec{x}_k)}$$

where evaluating H: $O(n) = \frac{n^2}{2}$

inverting H: $O(n) = n^3$

Consider quasi-newton's method:

Broyden's update

linearization: $(*)\ \nabla f(\vec{x}_{k+1}) \approx \nabla f(\vec{x}_k) + H(\vec{x}_k)(\vec{x}_{k+1} - \vec{x}_k) \approx \vec{0}$

$$\Rightarrow H_k \vec{s}_k = -\nabla f^{(k)}$$

By secant: $\Rightarrow H_{k+1}\vec{s}_k = \nabla f_{k+1} - \nabla f_k$

Extra: $H_{k+1} - H_k$ is of rank one.

$$H_{k+1} = H_k + \underbrace{\frac{1}{\vec{s}_k^T \vec{s}_k}}_{\text{number}} \underbrace{(\nabla f_{k+1} - \nabla f_k - H^k \vec{s}_k)\vec{s}_k^T}_{\text{outer product of rank 1}}$$

Algorithm:

- How to initialize H_0
- Set $\vec{x}_1 = \vec{x}_0 - H_0^{-1}\nabla f(\vec{x}_0)$
- $H_1 = H_0 + \cdots\cdots$
- $\vec{x}_2 = \vec{x}_1 - H_1^{-1}\nabla f(\vec{x}_1)$

To better compute inverse: Sherman-Morriso formula

$$(A+\vec{u}\vec{v}^T)^{-1} = A^{-1} - \frac{A^{-1}\vec{u}\vec{v}^T A^{-1}}{1+\vec{v}^T A^{-1}\vec{u}} \qquad H_{k+1}^{-1} = H_k^{-1} - \frac{H_k^{-1}\vec{u}\vec{v}^T H_k^{-1}}{1+\vec{v}^T H_k^{-1}\vec{u}}$$

Numerical Linear Algebra:

FLOP: floating point operation $(+ - \times \div)$

Put A in echelon form.

Loop over column $j=1,\dots,n-1$

Loop over row $i=j+1,\dots,n$.

① Compute $\frac{a_{ij}}{a_{jj}}$ (1 flop)

② Compute row $i - \frac{a_{ij}}{a_{jj}}$ row j ($2(n-j)$ flops)

③ Compute $b_i - \frac{a_{ij}}{a_{jj}} b_j$ (2 flops)

$$\sum_{j=1}^{n-1} \sum_{i=j+1}^{n} \left(2(n-j)+3\right) = \sum_{j=1}^{n-1} (n-j)\left[(2n-2j)+3\right] \approx O(n^3)$$

LU factorization:

$A = LU$, $LU\vec{x} = \vec{b}$ Solve $L\vec{y} = \vec{b}$ forward substitution $O(n^2)$

Solve $\vec{y} = U\vec{x}$ backward substitution $O(n^2)$

Pivoting: Prioritize large pivots

$PA = LU$

$$\begin{pmatrix} \epsilon & 1 \\ 1 & 1 \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \end{pmatrix} \qquad \left(\begin{array}{cc|c} \epsilon & 1 & 1 \\ 1 & 1 & 1 \end{array}\right) \sim \left(\begin{array}{cc|c} \epsilon & 1 & 1 \\ 0 & -\frac{1}{\epsilon} & -\frac{1}{\epsilon} \end{array}\right)$$

since ϵ is really small

- <u>Not</u> Necessary for SPD (i.e. $\vec{x}^T A \vec{x} > 0$).

Cholesky factorization: If A is SPD, assume $A = U^T U$

$$U^T U = \begin{pmatrix} u_{11}^2 & u_{11}u_{12} & u_{11}u_{13} \cdots \\ u_{12}u_{11} & u_{12}^2 + u_{22}^2 & \\ u_{13}u_{11} & & \ddots \\ \vdots & & \end{pmatrix} \Rightarrow u_{11}^2 = a_{11} \Rightarrow u_{11} = \sqrt{a_{11}} \text{ then } u_{12} = \frac{a_{12}}{u_{11}}$$

... cost is $O(\frac{n^3}{3})$

- Row pivoting

$L_m P_{m-1} \cdots L_3 P_2 L_2 P_1 L_1 A = U$ Note $P_j = P_j^{-1} = P_j^T$

E.X. $L_2 P_1 L_1 A = U$

$$\begin{pmatrix} 1 & & \\ 2 & 3 & \\ 6 & 5 & 4 \end{pmatrix} \to \begin{matrix} 6 & 5 & 4 \\ 2 & 3 & \\ 1 & & \end{matrix}$$

Conditioning. Backward stability.

THM $A(\vec{x}+\delta\vec{x})=(\vec{b}+\delta\vec{b}) \Rightarrow \frac{\|\delta\vec{x}\|}{\|\vec{x}\|} \leq \kappa(A)\frac{\|\delta\vec{b}\|}{\|\vec{b}\|}$

Perturbation $(A+\delta A)(\vec{x}+\delta\vec{x})=\vec{b} \Rightarrow \frac{\|\delta\vec{x}\|}{\|\vec{x}\|} \leq \kappa(A)\frac{\|\delta A\|}{\|A\|}$ as $\|\delta A\| \to 0$.

$\vec{r}=\vec{b}-A\tilde{x}$ ← computed solution

$A\tilde{x}=\vec{b}-\vec{r}$, $\|\vec{r}\|$ = backward error forward error $= \tilde{x}-\vec{x}$

= the perturbation from the original problem to the one that is solved exactly

small $\|\vec{r}\|$ doesn't imply small $\|\tilde{x}-\vec{x}\|$

$\frac{\|\vec{x}-\tilde{x}\|}{\|\vec{x}\|} \leq \kappa(A)\frac{\|\vec{r}\|}{\|\vec{b}\|}$ how do you get to this? ✓

may be much larger than the relative error in $\tilde{x}$. Kappa can be large.

$f(n) \sim O(g(n))$ if $f(n) \leq C g(n)$ for all $n > n_0 > 0$.

QR Factorization:

$P = A(A^TA)^{-1}A^T$, $A^TA\vec{\hat{x}} = A^T\vec{b}$, $Q^T = Q^{-1}$, $Q^TQ = QQ^T = I$

least squares solution:

$\vec{x} = R^{-1}Q^T\vec{b}$

$A = QR$

$$\left.\begin{aligned} \vec{a}_1 &= r_{11}\vec{q}_1 \\ \vec{a}_2 &= r_{12}\vec{q}_1 + r_{22}\vec{q}_2 \\ &\vdots \\ \vec{a}_n &= r_{1n}\vec{q}_1 + \cdots + r_{nn}\vec{q}_n \end{aligned}\right\}$$

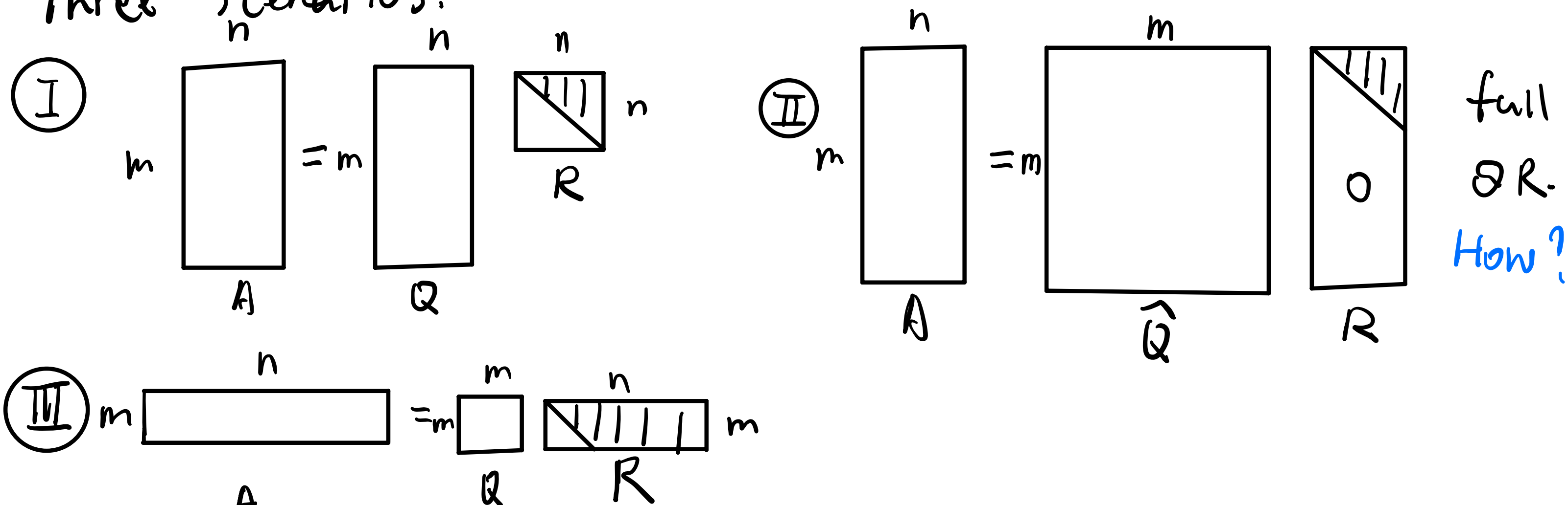
G-S Algorithm: On step j,

set $\vec{v}_j = \vec{a}_j - (\vec{q}_1^{\,T}\vec{a}_j)\vec{q}_1 - \cdots - (\vec{q}_{j-1}^{\,T}\vec{a}_j)\vec{q}_{j-1}$

then $\vec{q}_j = \dfrac{\vec{v}_j}{\|\vec{v}_j\|}$ $\boxed{\text{NUM Unstable}}$

Combined gives $r_{ij} = (\vec{q}_i, \vec{a}_j) = \vec{q}_i^{\,T}\vec{a}_j$. $|r_{jj}| = \|\vec{a}_j - \sum_{i=1}^{j-1} r_{ij}\vec{q}_i\|_2 = \|\vec{v}_j\|$

Three Scenarios:



$P_j = I - \hat{Q}_{j-1} Q_{j-1}^*$, $\hat{Q}_{j-1} = (\vec{q}_1 \cdots \vec{q}_{j-1})$ projection into col$(\hat{Q}_{j-1})$

$\vec{q}_1 = \dfrac{P_1 \vec{a}_1}{\|P_1 \vec{a}_1\|}$, - - - , $\vec{q}_j = \dfrac{P_j \vec{a}_j}{\|P_j \vec{a}_j\|}$, $\|Q\|_2 = 1$

$$P_j = I - \hat{Q}_{j-1} Q_{j-1}^* = \underbrace{(I - \hat{q}_{j-1} \hat{q}_{j-1}^*)}_{P_{\perp \hat{q}_{j-1}}} (I - \hat{q}_{j-2} \hat{q}_{j-2}^*) \cdots (I - \vec{q}_1 \vec{q}_1^*)$$

CODE (ref lec-10) $\rightarrow$ $O(2mn^2)$ cubic (like Gaussian)

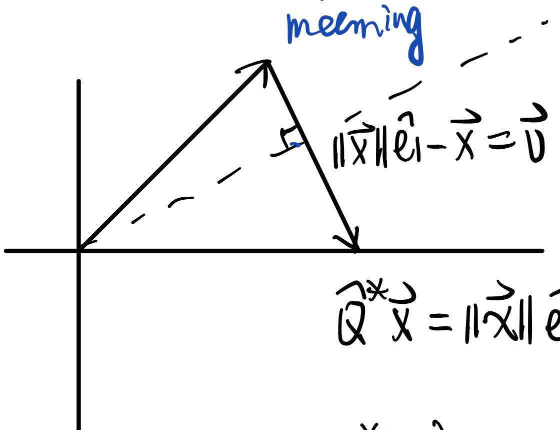
Code:
```
j = 1 : n
 v_j = a_j
 for i = 1 ; j-1
   r_ij = (q_i, q_j) , v_j = v_j - r_ij q_i   <- round off error accumulate
 end
 r_jj = ||v_j|| , q_j = v_j / r_jj
end
```

Type I

Householder reflection othogonal triangularization:

Construct $\hat{Q}^*$ that transforms A into upper triangular.

meaning



$$\hat{Q}_1^* \begin{pmatrix} x_1 \\ \vdots \\ x_m \end{pmatrix} = \begin{pmatrix} \|\vec{x}\| \\ 0 \\ \vdots \\ 0 \end{pmatrix} = \|\vec{x}\| \hat{e}_1$$

$$\hat{Q}_1^* = I - 2 \frac{\vec{v}\vec{v}^*}{\vec{v}^*\vec{v}}$$ how?

$$Q_k^* \vec{X} = \begin{pmatrix} x_1 \\ \vdots \\ x_{k-1} \\ \|\vec{x}_k - \vec{x}_m\| \end{pmatrix} = \begin{pmatrix} I_{k-1} & 0 \\ 0 & F_{m-k+1} \end{pmatrix}$$ $\leftarrow$ reflector.

Type II

$$\hat{Q}_j^* = \begin{pmatrix} I_j & 0 \\ 0 & R_{m-j} \end{pmatrix}, \quad R_{m-j+1}\begin{pmatrix} \tilde{a}_{jj} \\ \vdots \\ \hat{a}_{mj} \end{pmatrix} = \begin{pmatrix} \|\vec{a}_j\| \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

Algorithm and why is it more stable.

Truncated QR.

$\tilde{Q} \in \mathbb{C}^{m\times k}$, $\tilde{R} \in \mathbb{C}^{k\times n}$

$\downarrow \qquad\qquad \downarrow$

$Q(:,1:k) \qquad R(1:k,:)$

$\|A - \tilde{Q}\tilde{R}\| = \|QR - \tilde{Q}\tilde{R}\| = \|R - Q^*\tilde{Q}\tilde{R}\|$ 乘 $\boxed{\text{othogonal matrix}}$ 不改变模

$= \left\|R - \begin{pmatrix} I_k \\ 0 \end{pmatrix}\tilde{R}\right\|$

$= \left\|R - \begin{pmatrix} \tilde{R} \\ 0 \end{pmatrix}\right\|$

$= \left\|\begin{array}{c} \begin{pmatrix} 0 \\ \hline \\ 0 \end{pmatrix} \end{array}\begin{array}{l} k \\ k+1 \\ n \\ m \end{array}\right\|$ $\Leftarrow$ Type II

Linear Regression:

↓ why no square here?

$$\min \sum_{i=1}^{n} (y_i - a - bx_i - cx_i^2)^2 = \min \| X\beta - y \| \leftarrow \text{design matrix.}$$

solve $\beta = (X^T X)^{-1} X^T y$

SVD: $A = USV^*$

$A \in \mathbb{C}^{m \times n}$, $A^* A$ is positive semi definite

$\bar{A}^T A = V S^2 V^T$ $\quad U = AVS^{-1}$ $\quad A = USV^*$

$\kappa(A^* A) = \kappa(A)^2$ numerically unstable

Alternative: only iff $m = n$.

$H = \begin{pmatrix} 0 & A^* \\ A & 0 \end{pmatrix} \in \mathbb{C}^{2m \times 2n}$ is Hermtian ($a_{ij} = \overline{a_{ji}}$)

$$\underbrace{H \begin{pmatrix} V & V \\ U & -U \end{pmatrix} = \begin{pmatrix} V & V \\ U & -U \end{pmatrix} \begin{pmatrix} S & 0 \\ 0 & -S \end{pmatrix} = \begin{pmatrix} VS & -VS \\ US & US \end{pmatrix}}_{\text{eigen decomposition of } H}$$

Pseudo-inverse: (A not square)

$A^+ = VS^{-1}U^*$

$A^+ A = (VS^{-1}U^*)(USV^*)$

$= I$

Application to least square problems.

Eigenvalue:

direct: $\det(A-\lambda I)=0$. costs $n!$ flops to form $p(x)$

expensive. non-linear root finding

$$A=PDP^{-1} \quad \vec{b}=PDP^{-1}\vec{x} \Rightarrow \underbrace{P^{-1}\vec{b}}_{\vec{u}}=D\underbrace{P^{-1}\vec{x}}_{\vec{y}} \qquad \vec{u}=D\vec{y}, \quad \vec{x}=P\vec{y}$$

Gerschgorin's Thm:

$$D_i=\{z\in\mathbb{C} \text{ s.t. } |z-a_{ii}|\le\sum_{j\ne i}|a_{ij}|\}$$

Power Method: (A is diagonalizable) (largest $\lambda_1, \vec{v}_1$)

$$A^k\vec{w}=\sum_j c_j\lambda_j^k\vec{v}_j \approx c_1\lambda_1^k\vec{v}_1$$

normalize every step:

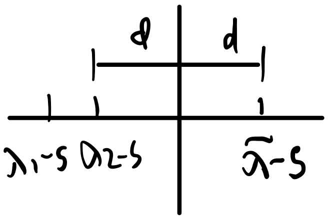
$$w_k^T A\vec{w}_k \approx \lambda_1 \underbrace{\vec{w}_k}_{\vec{w}_k'} \Rightarrow \lambda_1=(A\vec{w}_k, \vec{w}_k)$$

convergence:

$$\vec{v}_1 \approx \frac{1}{c_1\lambda_1^k}A^k\vec{w}=\vec{v}_1+\frac{c_2}{c_1}\left(\frac{\lambda_2}{\lambda_1}\right)^k\vec{v}_2+\cdots$$

$\|\vec{w}_k-\vec{v}_1\| \sim O\left(\left|\frac{\lambda_2}{\lambda_1}\right|^k\right)$ depends on gap between λ_1, λ_2.

To accelerate, use power method with shift



To get middle λ's. Inverse Power Method with shift

choose s s.t. $\frac{1}{\lambda-s}$ is large $(A-sI)^{-1}$ converges fast.

expensive due to inversion

Algorithm:

1. random $\vec{w}_0$
2. solve $(A-sI)\vec{y}_1 = \vec{w}_0 \iff \vec{y}_1 = (A-sI)^{-1}\vec{w}_0$
3. set $\vec{w}_1 = \frac{\vec{y}_1}{\|\vec{y}_1\|}$

Jacobi's Method ($\overset{A}{\checkmark}$ diagonalizable, every eigen).

1. $A^{(0)} = A$.
2. Find pq element max.
3. $\varphi_k = \frac{1}{2}\arctan\left(\frac{2a_{pq}^{(k)}}{a_{qq}^{(k)} - a_{pp}^{(k)}}\right) \leftarrow \frac{2b}{d-a}$
4. Set $A^{(k+1)} = R^{pq}(\varphi_k)^T A^{(k)} R^{pq}(\varphi_k)$

Continue until $|a_{pq}^{(k)}| < \varepsilon$, $p \neq q$.

$R^{(k)} = R(\varphi_1) \cdots R(\varphi_k) \to (\vec{v}_1 \cdots \vec{v}_n)$ eigenvectors

Lemma: R is orthogal. $A^T = A$, then

$\|A\|_F = \|R^T A R\|_F$

proof. $\|A\|_F^2 = \text{trace}(A^T A) = \text{trace}(A^2) = \text{trace}(B^T B) = \|B\|_F^2$

Convergence:

$L(A) \le n(n-1)a_{pq}^2$. $\quad L(B) = L(A) - 2a_{pq}^2 \le L(A)\left(1 - \frac{2}{n(n-1)}\right)$

$L(A^{(k)}) \le \underbrace{\left(1 - \frac{2}{n(n-1)}\right)}_{<1} L(A^{(0)})$ faster than it seems.

QR Method: for general matrices.

Algorithm: $A^{(k-1)} = Q^{(k)} R^{(k)}$, $A^{(k)} = R^{(k)} Q^{(k)}$

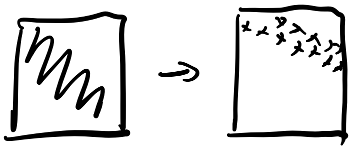
① reduce A to ~~bi~~tridigonal

② apply to shifted matrices. $A^{(k)} - \mu^{(k)} I$, μ estimates λ.

③ use deflection

$$A^{(k-1)} = R^{(k)} Q^{(k)} + \mu^{(k)} I$$

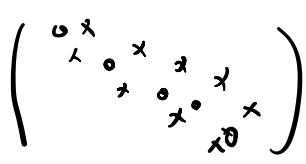
Compute SVD:

A =  B via householder's reflection.

$U^T A V = B$

$A = (U U_B) S (V V_B)^T$, share singular values.

$$H = \begin{pmatrix} 0 & B^T \\ B & 0 \end{pmatrix}, \quad H \begin{pmatrix} V_B & V_B \\ U_B & -U_B \end{pmatrix} = \begin{pmatrix} V_B & V_B \\ U_B & -U_B \end{pmatrix} \begin{pmatrix} S & 0 \\ 0 & -S \end{pmatrix}$$ eigendecomp

$P^T H P =$  ← Apply QR

$$\tilde{Q}^T P^T H P \tilde{Q} = \begin{pmatrix} S & 0 \\ 0 & -S \end{pmatrix}$$

Interpolation:

$$\underbrace{\begin{pmatrix} 1 & x_0 & x_0^2 & \cdots & x_0^n \\ 1 & x_1 & x_1^2 & \cdots & x_1^n \\ & & & & \\ 1 & x_n & x_n^2 & \cdots & x_n^n \end{pmatrix}}_{} \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{pmatrix} = \begin{pmatrix} y_0 \\ y_1 \\ \vdots \\ y_n \end{pmatrix}$$

Vandermonde matrix A [Don't invert it]

Lagrange: inefficient, unstable

$$L_k(x_j) = \begin{cases} 1, & \text{if } j=k \\ 0, & \text{if } j \neq k \end{cases} \qquad p_n(x) = \sum_{k=0}^{n} y_k L_k(x) \qquad L_k(x) = \prod_{\substack{j=0 \\ j\neq k}}^{n} \frac{x-x_j}{x_k-x_j} \qquad O(n^2)$$

interpolation error depends on the points

Horner's Method:

$$p_n(x) = a_0 + x(a_1 + a_2 x + \cdots + a_n x^{n-1})$$

$$= a_0 + x(a_1 + x(a_2 + \cdots + a_n x^{n-2})) \qquad \Rightarrow O(2n)$$

$$= a_0 + x(a_1 + x(a_2 + x(\cdots)))$$

$b_{n-1} = a_{n-1} + a_n x$

$b_{n-2} = a_{n-2} + b_{n-1} x$ recurrent

Baricentric form: (modified lagrange)

$$p_n(x) = \sum_{k=0}^{n} y_k \prod_{\substack{k\neq j \\ j=0}}^{n} \frac{x-x_j}{x_k-x_j} = \sum_{k=0}^{n} \left(\prod_{j=0}^{n} (x-x_j)\right) \frac{1}{x-x_k} \left(\prod_{j\neq k} \frac{1}{(x_k-x_j)}\right) y_k$$

$$= \left(\prod_{j=0}^{n} (x-x_j)\right) \sum_{k=0} \frac{1}{x-x_k} \left(\prod_{j\neq k} \frac{1}{x_k-x_j}\right) y_k$$

$$= \varphi(x) \sum_{k=0}^{n} \frac{w_k}{x-x_k} y_k$$ First Baricentric Formula.

$$1 = \varphi(x) \sum_{k=0}^{n} \frac{w_k}{x-x_k}$$

$$p_n(x) = \frac{\sum_{k=0}^{n} w_k/x-x_k \cdot y_k}{\sum_{k=0}^{n} w_k/x-x_k}$$ Second Baricentric form

Convergence: $\lim\limits_{n\to\infty} \max\limits_{x} |f(x)-p_n(x)| = \max\limits_{\xi} \frac{|f^{(n+1)}(\xi)|}{(n+1)!} \cdot \max\limits_{x} \prod_{j=0}^{n} |x-x_j|$

Function Approximation: $\min\limits_{p_n\in P_n} \|p_n-f\|_\infty$. generally can't find p_n.

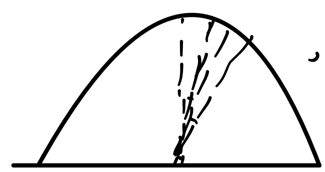
THM: $f(x)=x^{n+1}$, $\|p_n-f\|_\infty$ is minimized when

$$p_n(x)=x^{n+1}-\frac{1}{2^n}\cos((n+1)\arccos(x))$$

$T_n(x)=\cos(n\arccos x)$: Chebyshev polynomial of degree n. $x\in[-1,1]$.

$T_{n+1}(x)=2xT_n(x)-T_{n-1}(x)$.

$x_j=-\cos\left(\frac{(2j-1)\pi}{2n}\right)$. $j=1,\dots,n$ (n roots)? on $[-1,1]$



interpolations at these roots yield near minmax polynomial p_n

since $\prod_{j=0}^{n}(x-x_j)=\frac{1}{2^n}T_n(x)$ ← the minimum norm monic polynomial

$$\|f-p_n\|_\infty \le \frac{M_{n+1}}{(n+1)!}\left\|\prod_{j=0}^{n}(x-x_j)\right\|_\infty$$

Approximation in 2-norm: to minimize $\|f-p_n\|_2$

$$(f,g)=\int_a^b f(x)g(x)\,dx$$

$$A=\int_a^b\Big(f(x)-\sum_{j=0}^{n}c_jp_j(x)\Big)^2dx=(f,f)-2\sum_{j=0}^{n}c_j(f,p_j)+\sum_{j=0}^{n}\sum_{k=0}^{n}c_jc_k(p_j,p_k)$$

$$\frac{\partial A}{\partial c_i}=-2(f,p_i)+2\sum_{k=0}^{n}c_k(p_i,p_k)=0 \;\Rightarrow\; \sum_{k=0}^{n}c_k(p_i,p_k)=(f,p_i)$$

$$\begin{pmatrix}(p_0,p_0) & \cdots & (p_0,p_n)\\ (p_1,p_0) & & (p_1,p_n)\\ \vdots & & \vdots\\ (p_n,p_0) & \cdots & (p_n,p_n)\end{pmatrix}\begin{pmatrix}c_0\\ c_1\\ \vdots\\ c_n\end{pmatrix}=\begin{pmatrix}(f,p_0)\\ (f,p_1)\\ \vdots\\ (f,p_n)\end{pmatrix}$$

Solve this $(n+1)$ linear system to get $\vec{c}$

$\varphi=c_0p_0+c_1p_1+\cdots+c_np_n$

Legendre polynomial: $[-1, 1]$

$$P_2 = m_2 - \frac{(m_2, P_0)}{(P_0, P_0)} P_0 - \frac{(m_2, P_1)}{(P_1, P_2)} P_1 \qquad G\text{-}S.$$

All orthogonal polynomials satisfies

$$\varphi_n(x) = (x + a_n)\varphi_{n-1}(x) + b_n \varphi_{n-2} \quad \text{for } n = 2, 3, \ldots$$

Numerical Integration:

$$\sum_{j=0}^{n} w_j f(x_j) = \int_a^b f(x)dx$$

quadrature weights — quadrature nodes

large n may be inaccurate due to Runge's Phenomenon.

<u>Remedy</u>: Use many smaller lower order interpolations.

Composite trapezoidal rule:

$$\int_a^b f(x)dx \approx T_n f = h\left(\sum_{j=0}^{n} f(x_j) - \frac{1}{2}(f(a)+f(b))\right) \quad \text{error: } O(h^2)$$

THM: $f \in C^{2k}[a,b]$, $[a,b]$ is divided into n intervals, $x_j = a+jh$.

Then $\int_a^b f(x)dx - T_n f =$

Euler Mclaurin formula.

$\Rightarrow$ if $f \in C^\infty[a,b]$ and periodic with $f^{(j)}(a) = f^{(j)}(b)$.

(e.g. fourier series) then error decays $|I - T_n f|$ superalgebraically ~~to~~ as $n \to \infty$

Def. $E_n \to 0$ superalgebraically if $\lim_{n\to\infty} \frac{E_n}{h^{p \cdot n}} = 0$ for any $p > 0$.

Clenshaw - Curtis Quadrature:

special case of <u>Newton-Cotes</u>

- interpolate f at Chebyshev nodes.
- integrate each Chebyshev polynomial.

$$f(\cos\theta) = \sum c_k T_k(x)$$

$$= \sum c_k \cos(k\,\mathrm{acos}\,x)$$

$$= \sum c_k \cos(k\theta)$$

$$\int_{-1}^{1} f(x)\,dx \approx \int_0^{\pi} f(\cos\theta)\sin\theta\,d\theta$$

$$\approx \int_0^{\pi} \underbrace{\left(\sum c_k \cos k\theta\right)\sin\theta}_{\text{periodic}}\,d\theta$$

Richarson Extrapolation:

$\varphi = \varphi_0(h) + c_1 h + c_2 h^2 + \cdots + c_3 h^3$, h is small.

Gaussian Quadrature: (orthogonal polynomials)

A quadrature is called GAUSSIAN if it's exact for $2n$ linearly independent functions

(*). If $x_1, \dots, x_n$ are the zeros of P_n. the degree n Legendre polynomial, then the formula:

$$\int_a^b f(x)\,dx \approx \sum_{j=1}^{n} w_j f(x_j)$$

where $w_j = \int_a^b \varphi_j(x)\,dx$; $\varphi_j(x) = \prod_{l \neq j} \frac{x - x_l}{x_j - x_l}$ is exact for polynomials of degree $2n-1$ or less

Fourier Series:

$f \in L_2[0, 2\pi]$

$$f = \sum_{-\infty}^{\infty} c_k e^{ikx}$$

$$\langle f, g\rangle = \int_0^{2\pi} f(x)\,\overline{g(x)}\,dx.$$

$$c_m = \frac{1}{2\pi}\int_0^{2\pi} f(x)\, e^{-imx}\,dx$$

$$\approx \frac{1}{n}\sum_{j=0}^{n-1} f\left(\frac{2\pi j}{n}\right) e^{-2\pi i j m/n}$$ Composite trape

$w_{kj} = e^{-2\pi i j k/n}$

DFT:

$$F_k = \frac{1}{n}\sum_{j=0}^{n-1} f_j e^{-2\pi i jk/n} \quad \text{for } k=0,\dots n-1 \qquad \vec{F} = \frac{1}{n} W \vec{f}$$

IDFT:

$$f_j = \sum_{k=0}^{n-1} F_k e^{2\pi i jk/n} \quad \text{for } j=0,\dots, n-1 \qquad \vec{f} = W^* \vec{F}$$

Fourier Transform

$$\hat{f}(\omega) = \frac{1}{2\pi}\int_{-\infty}^{\infty} f(x) e^{-i\omega x} dx$$

Inverse

$$f(x) = \int_{-\infty}^{\infty} f(x) e^{i\omega x} dx.$$

Fourier Series.

$$C_m = \int_0^{2\pi} f(x) e^{-imx} dx.$$

Convolution: $(f*g)(x) = \int_{-\infty}^{\infty} f(y)\, g(x-y)\, dy$

$$= \mathcal{F}^{-1}(\mathcal{F}f \cdot \mathcal{F}g)$$