1. Floating point number and arithmetic

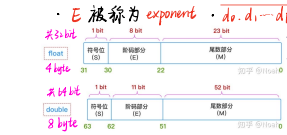
§1 Floating-point numbers

1. 基本表示形式

① 一个 floating-point number system 由4个量刻画：β：base / index　p：precision（可被精确表示的位数）　$[L, U]$：exponent range

② 实数 x 被表示为：$x = \pm\left[d_0 + \frac{d_1}{\beta} + \frac{d_2}{\beta^2} + \cdots + \frac{d_{p-1}}{\beta^{p-1}}\right]\beta^E$

· E 被称为 exponent　· $d_0.d_1\cdots d_{p-1}$ 被称为 mantissa　· $d_1\cdots d_{p-1}$：fraction



System	β	p	L	U
IEEE HP	2	11	-14	15
IEEE SP (single precision)	2	24	-126	127
IEEE DP (double precision)	2	53	-1022	1023
IEEE QP (quadruple precision)	2	113	-16382	16383

2. Normalization：leading digit d_0 不为零（除非表示数字0）$\Rightarrow 1 \le m < \beta$

3. Floating-point system 的性质

① normalized 浮点数总数为 $2(\beta-1)\beta^{p-1}(U-L+1)+1$

② UFL $= \beta^L$（十进制）　OFL $= \beta^{U+1}(1-\beta^{-p})$（十进制）

③ 仅在给定 power 时 β 相同的情况下，floating-point numbers 为 equally spaced

④ 不是所有实数均 exactly representable，能被表示的为 machine numbers

e.g. $\beta=2, p=3, L=-1, U=1$，则 $OFL = (1.11)_2 \cdot 2^1 = (11.)_2 = (3.5)_{10}$

$UFL = (1.00)_2 \cdot 2^{-1} = (0.10)_2 = (0.5)_{10}$

§2 Rounding rules

1. 两个 rounding rules：将 x 近似为一个“邻近”的浮点数 $fl(x)$

① chop / rounding towards zero：truncate 第 $(p-1)$ 位后的 base-β expansion of x

② round to nearest / round to even：$fl(x)$ 为最接近 x 的浮点数，若出现 tie，则选择最后一位为偶数的浮点数.（IEEE 标准默认的 rounding rule）

2. Machine precision 系统的 accuracy 由 machine precision ε_{mach} 刻画

① 第一种定义：chopping：$\varepsilon_{mach} = \beta^{1-p}$，rounding to nearest：$\varepsilon_{mach} = \frac{1}{2}\beta^{1-p}$

② 第二种定义：使得 $fl(1+\varepsilon) > 1$ 的最小的 ε

3. Proposition：$\forall x \in \mathbb{R}, |fl(x) - x| \le \varepsilon_{mach}|x|$

$\iff fl(x) = (1+\varepsilon)x$ for some ε with $|\varepsilon| \le \varepsilon_{mach}$

4. Subnormal 与 gradual underflow 当 exponent 取最小时，leading digit 允许取 0

§3 Floating-point arithmetic

1. Overflow 与 underflow：Overflow：产生的问题，underflow：set to zero

关于 overflow：　两数相加：① 同号，可能 overflow　② 异号，不可能 overflow

2. Floating-point arithmetic 的运算规则（IEEE standard 754）

对任意运算 $*\in\{+,-,\times,/\}$ 与任意一对 machine numbers x, y，有

$x \circledast y = fl(x * y) = (1+\varepsilon)\cdot(x*y)$ for some $|\varepsilon| < \varepsilon_{mach}$

浮点数加法与乘法满足 commutative（交换律）但不满足 associative（结合律）

3. 减法运算时的 Cancellation

两个有相同 sign 与相似 magnitudes 的 p-digit 数相减，尽管结果可以 exactly representable，但会导致严重的 loss of information（leading terms 变为 0）

注：可通过因式分解、分子有理化、倍角展开等方法处理两数相减

① 二倍角公式：$\cos(2x) = 1 - 2\sin^2 x$，$\sin(2x) = 2\sin x\cos x$

② Taylor expansion：① $e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots + \frac{x^n}{n!} + \cdots \quad (-\infty < x < +\infty)$

② $\sin x = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \cdots + (-1)^n \frac{x^{2n+1}}{(2n+1)!} + \cdots \quad (-\infty < x < +\infty)$

③ $\frac{1}{1-x} = 1 + x + x^2 + x^3 + \cdots + x^n + \cdots \quad (-1 < x < 1)$

④ $\ln(1+x) = x - \frac{x^2}{2} + \frac{x^3}{3} - \frac{x^4}{4} + \cdots + (-1)^{n-1}\frac{x^n}{n} + \cdots \quad (-1 < x \le 1)$

⑤ $\cos x = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \cdots + (-1)^n \frac{x^{2n}}{(2n)!} + \cdots \quad (-\infty < x < +\infty)$

⑥ $a^x = 1 + x\ln a + \frac{x^2(\ln a)^2}{2!} + \cdots + \frac{x^n(\ln a)^n}{n!} + \cdots \quad (-\infty < x < +\infty)$

⑦ $\frac{1}{1+x^2} = 1 - x^2 + x^4 - x^6 + \cdots + (-1)^n x^{2n} + \cdots \quad (-1 < x < 1)$

⑧ $\arctan x = x - \frac{x^3}{3} + \frac{x^5}{5} - \frac{x^7}{7} + \cdots + (-1)^n \frac{x^{2n+1}}{2n+1} \quad (-1 \le x \le 1)$

2. Error Analysis

§1 Approximation 与 errors

1. Absolute error（绝对误差）与 relative error（相对误差）

① Absolute error = approximate value − true value（近似值−真实值）

② Relative error = $\frac{\text{absolute error}}{\text{true value}}$（绝对误差/真实值）

2. Data error（使播数据误差）与 computational error（计算误差）

Total error $= \hat{f}(\hat{x}) - f(x) = \hat{f}(\hat{x}) - f(\hat{x}) + f(\hat{x}) - f(x)$

$=$ computational error $+$ propagated data error

① computational error 表示仅由计算（近似于）产生的 error，与 algorithm 有关

② propagated data error 表示仅由 input value 产生的 error，与 algorithm 无关

3. Truncation error（截断误差）与 rounding error（舍入误差）

computational error $= \hat{f}_{\text{round}}(\hat{x}) - \hat{f}_{\text{without round}}(\hat{x}) + \hat{f}_{\text{without round}}(\hat{x}) - f(\hat{x})$

$=$ rounding error $+$ truncation error

① Rounding error 表示仅由 rounding 产生的 error　② Truncation error 表示在 exact arithmetic（不进行 rounding）的情况下，仅由计算（近似于）产生的 error

③ 通常其中一个会 dominates（当 $\hat{f}$ 接近于时，会 cancelling，增大 rounding error）

e.g. Finite difference approximation 的 error $f'(x) \approx \frac{f(x+h)-f(x)}{h}$

$f(x+h) = f(x) + f'(x)h + \frac{1}{2}f''(x+th)h^2, t\in[0,1]$　$f''(x)$ 在 x 附近 is bounded by M

Truncation error $= \frac{f(x+h)-f(x)}{h} - f'(x) = \frac{1}{2}f''(x+th)h$ bounded by $\frac{1}{2}Mh$

Assumption：仅在计算 $\hat{f}$ 时 make an error bdd by $\varepsilon \Rightarrow$ Rounding error bdd by $\frac{2\varepsilon}{h}$

因此，total error is bounded by $\frac{1}{2}Mh + \frac{2\varepsilon}{h}$，且当 $h \approx 2\sqrt{\varepsilon/M}$ 时取最小值

§2 Forward and backward error

1. forward and backward error（前向误差与后向误差）计算 $y = f(x)$

① Forward error：$\Delta y = \hat{y} - y = \hat{f}(x) - f(x)$

计算结果 $\hat{y}$ 与真实结果 y 的差值（$\hat{f}$ 包含了近似于 f 的误差与 input data 的误差）

② Relative forward error：$\frac{|\Delta y|}{|y|} = \frac{|\hat{y}-y|}{|y|} = \frac{|\hat{f}(x)-f(x)|}{|f(x)|}$

③ Backward error：$\Delta x = \hat{x} - x = f^{-1}(\hat{f}(x)) - x$

能通过原函数得出 $\hat{y}$ 的 input $\hat{x}$ 与 actual input x 的差值

④ Relative backward error：$\frac{|\Delta x|}{|x|} = \frac{|\hat{x}-x|}{|x|} = \frac{|f^{-1}(\hat{f}(x)) - x|}{|x|}$

§3 Conditioning

1. well-posed problem：solution ① exists ② unique ③ depends continuously on problem data

2. sensitivity 与 conditioning　一个 problem 被称为

① insensitive/well-conditioned：input 的 relative change $\approx$ solution 的 relative change

② sensitive/ill-conditioned：input 的 relative change $\ll$ solution 的 relative change

3. Condition number（条件数）

$\text{cond} = \frac{|\text{relative change in solution}|}{|\text{relative change in input data}|} = \frac{|(\hat{f}(\hat{x})-f(x))/f(x)|}{|(\hat{x}-x)/x|} = \frac{|\Delta y/y|}{|\Delta x/x|}$

注：$\text{cond} \le 10^3/10^4 \Rightarrow$ well-conditioned；$\text{cond} \ge 10^8/10^8 \Rightarrow$ ill-conditioned

表示了 Relative forward error 相对于 Relative backward error 的 amplification factor

实际中通常会求 $|\text{Relative forward error}| \lesssim \text{cond} \cdot |\text{Relative backward error}|$

4. 条件数的计算：以 evaluating a function 为例

Condition number $\text{cond} \approx \left|\frac{f(x)\Delta x/f(x)}{\Delta x/x}\right| = \left|\frac{x f'(x)}{f(x)}\right|$

注：在上例中，有 $\text{cond}(f^{-1}) = \frac{1}{\text{cond}(f)}$

证 $\frac{1}{\text{cond}(f)(x)} = \left|\frac{f(x)}{x f'(x)}\right|$　$\text{cond}(f^{-1})(y) = \left|\frac{y}{f^{-1}(y)\cdot\frac{f'(y)}{f(x)}}\right|$　令 $y = f(x)$

§4 Stability

1. Stability：一个 problem $f: \mathbb{R}^n \to \mathbb{R}^m$ 的 algorithm $\hat{f}: \mathbb{R}^n \to \mathbb{R}^m$ 被称为 stable，

若对 $\forall x \in \mathbb{R}^n$，有 $\frac{\|\hat{f}(\hat{x}) - f(x)\|}{\|f(x)\|} = O(\varepsilon_{mach})$ for some $\hat{x}$：$\frac{\|\hat{x}-x\|}{\|x\|} = O(\varepsilon_{mach})$

对所有 input x，algorithm $\hat{f}(x)$ 的结果都近似等于（非）近似问题 $(f(\hat{x}))$ 的解

2. big-O：$f(x) = O(g(x)) \iff \exists \varepsilon > 0, C > 0$ s.t. $|f(x)| \le C|g(x)|, \forall\, |x-a| < \varepsilon$

3. small-o：$f(x) = o(g(x))$ for $x \to a \iff \lim_{x\to a} |f(x)|/|g(x)| = 0$

§5 Accuracy

1. Accuracy　一个 problem $f: \mathbb{R}^n \to \mathbb{R}^m$ 的 algorithm $\hat{f}: \mathbb{R}^n \to \mathbb{R}^m$ 被称为 accurate，

若对 $\forall x \in \mathbb{R}^n$，有 $\frac{\|\hat{f}(x) - f(x)\|}{\|f(x)\|} = O(\varepsilon_{mach})$（for $\varepsilon_{mach} \to 0$）

所有 input data 经 algorithm 处理后得到的结果都接近原问题的结果

2. Theorem：若 ① $f: \mathbb{R}^n \to \mathbb{R}^m$ 的 relative condition number 为 cond

② $\hat{f}$ 为 stable algorithm，则 $\frac{\|\hat{f}(x) - f(x)\|}{\|f(x)\|} \approx O(\text{cond}(x)\cdot\varepsilon_{mach})$

证 若 $\hat{x}$ 满足 $\frac{\|\hat{x}-x\|}{\|x\|} = O(\varepsilon_{mach})$ 则根据 Stability，有

$$\frac{\|\hat{f}(x) - f(x)\|}{\|f(x)\|} = \frac{\|\hat{f}(x) - f(\hat{x}) + f(\hat{x}) - f(x)\|}{\|f(x)\|} \le \frac{\|\hat{f}(x) - f(\hat{x})\|}{\|f(x)\|} + \frac{\|f(\hat{x}) - f(x)\|}{\|f(x)\|}$$

$$= \frac{\|\hat{f}(x) - f(\hat{x})\|}{\|f(x)\|} + \left[\frac{\|f(\hat{x}) - f(x)\|}{\|f(\hat{x})\|} \Big/ \frac{\|\hat{x}-x\|}{\|\hat{x}\|}\right]\cdot\frac{\|\hat{x}-x\|}{\|\hat{x}\|} = O\left[(1+\text{cond})\varepsilon_{mach}\right]$$

1. Consider the evaluation of $f(x) = e^x$. Let the algorithm be $\hat{f}(x) = 1 \oplus fl(x)$. Show the algorithm is **inaccurate**.

2. Consider the evaluation of $f(x) = 2x$. Let the algorithm be $\hat{f}(x) = 2 \otimes fl(x)$. Show the algorithm is **accurate** and **stable**.

Floating-point arithmetic, there exists $|\varepsilon_1|, |\varepsilon_2| \le \epsilon_{mach}$ such that

$\hat{f}(x) = x(1+\varepsilon_1)$,　$\hat{f}(x) = [1+x(1+\varepsilon_1)](1+\varepsilon_2)$.

have

$$\frac{|\hat{f}(x) - f(x)|}{|f(x)|} = \frac{\|[1+x(1+\varepsilon_1)](1+\varepsilon_2) - e^x\|}{\|e^x\|} = \frac{\|1 + x - e^x + \varepsilon_2 + x(\varepsilon_1+\varepsilon_2+\varepsilon_1\varepsilon_2)\|}{\|e^x\|}$$

$$\to \frac{|1+x-e^x|}{\|e^x\|} (\text{as } \epsilon_{mach} \to 0) \neq \mathcal{O}(\epsilon_{mach}).$$

2. By floating-point arithmetic, there exists $|\varepsilon_1|, |\varepsilon_2| \le \epsilon_{mach}$ such that

$\hat{f}(x) = x(1+\varepsilon_1)$,　$\hat{f}(x) = 2x(1+\varepsilon_1)$,　$\hat{f}(x) = 2x(1+\varepsilon_1)(1+\varepsilon_2)$.

$\varepsilon_2 = \mathcal{O}(\epsilon_{mach})$

Then we have

$$\frac{|\hat{f}(x) - f(x)|}{|f(x)|} = \frac{\|2x(1+\varepsilon_1)(1+\varepsilon_2) - 2x\|}{\|2x\|} = |\varepsilon_1 + \varepsilon_2 + \varepsilon_1\varepsilon_2| = \mathcal{O}(\epsilon_{mach}).$$

This shows the algorithm is **accurate**. Moreover, we have

$$\frac{|\hat{f}(x) - f(\hat{x})|}{|f(\hat{x})|} = \frac{|2x(1+\varepsilon_1)(1+\varepsilon_2) - 2x(1+\varepsilon_1)|}{\|2x\|} = |\varepsilon_2 + \varepsilon_1\varepsilon_2| = \mathcal{O}(\epsilon_{mach}).$$

This shows the algorithm is **stable**, where we pick $\hat{x} = fl(x)$ which satisfies $\|\hat{x} - x\|/\|x\| = \mathcal{O}(\epsilon_{mach})$.

e.g. $f(x,y) = x-y$ 与 $\hat{f}(x,y) = x \ominus y = fl(fl(x) - fl(y))$ 证 inaccuracy

构造 $x = 1 + \alpha\varepsilon_{mach}$，$y = 1 + \frac{1}{2}\varepsilon_{mach}$，$\alpha > 1$

assume $fl(x) = x$（为 machine number），$fl(y) = 1$ 则有

$x - y = (x - \frac{1}{2})\varepsilon_{mach}$　$x \ominus y = fl(1 + \alpha\varepsilon_{mach} - 1) = \alpha\varepsilon_{mach}$

$$\frac{|\hat{f}(x,y) - f(x,y)|}{|f(x,y)|} \ge \frac{\frac{1}{2}\varepsilon_{mach}}{(x-\frac{1}{2})\varepsilon_{mach}} = \frac{1}{x^2 - 1} \neq O(\varepsilon_{mach}) \text{ as } \varepsilon_{mach} \to 0$$

3. Systems of Linear Equations

§1 Vector norm

① $\|x\|_1 \ge \|x\|_2 \ge \|x\|_\infty$　② $\|x\|_2 \le \sqrt{n}\cdot\|x\|_\infty$

③ $\|x\|_2 \le \sqrt{n}\cdot\|x\|_\infty$　④ $\|x\|_1 = \sum_{i=1}^n |x_i| \le \sum_{i=1}^n \max_j |x_j| = n\cdot\|x\|_\infty$

§2 Matrix norms

1. Matrix norms 的基本要求

① Positivity：$\|A\| > 0$ 若 $A \neq 0$；$\|A\| = 0$ 当且仅当 $A = 0$

② Positive homogeneity：$\|\alpha A\| = |\alpha|\cdot\|A\|$ for $\forall$ scalar $\alpha \in \mathbb{R}$

③ Triangle inequality：$\|A+B\| \le \|A\| + \|B\|$, for $\forall\, A, B$

④ Sub-multiplicative：$\|AB\| \le \|A\|\cdot\|B\|$ for $\forall\, A, B$（CS inequality 的延伸）

⑤ 与 vector norm compatible：$\|Ax\| \le \|A\|\cdot\|x\|$ for $\forall$ A, vector x

（vector norm 的性质）

2. Induced matrix norm（表示在给定 vector norm 下，A 能造成多大的相对拉伸）

$$\|A\| := \max_{x\neq 0} \frac{\|Ax\|}{\|x\|} = \max_{\|x\|=1} \|Ax\|$$

证 induced matrix norm 满足 5 条 norm properties

由于 $\|Ax\|$ 为 vector norm，由 vector norm 的定义可得到 Property ① ② ③

对于 property ③，$\|A\| := \max_{x\neq 0} \frac{\|Ax\|}{\|x\|} \ge \frac{\|Ax\|}{\|x\|}$ 即 $\|Ax\| \le \|A\|\|x\|$

对于 property ④，$\|AB\| = \max_{\|x\|=1} \|ABx\| \le \|A\| \max_{\|x\|=1}\|Bx\| = \|A\|\|B\|$

3. 常见的 induced matrix norms

① l_1 norm（列和范数）$\|A\|_1 := \max_{j\in\{1,\cdots,m\}} \sum_{i=1}^n |a_{ij}|$

② l_2 norm/spectral norm（谱范数）$\|A\|_2 := \sqrt{\lambda_{max}(A^TA)}$

③ ∞-norm（行和范数）$\|A\|_\infty := \max_{i\in\{1,\cdots,n\}} \sum_{j=1}^m |a_{ij}|$

证 Derivation for l_1 matrix norm

定义 $A_{m\times n} = \begin{bmatrix} a_1^T \\ a_2^T \\ \vdots \\ a_m^T \end{bmatrix} = [A_1, A_2, \cdots, A_n]$ 有 $\|A_j\|_2 = \sum_{i=1}^m |a_{ij}|$

① 证明 $\frac{\|Ax\|_1}{\|x\|_1} \le \max_{j\in\{1,\cdots,n\}} \sum_{i=1}^m |a_{ij}|$

$\|Ax\|_1 = \|[a_1^Tx \; a_2^Tx \; \cdots]\|_1 = \sum_{i=1}^m |a_i^Tx| = \sum_{i=1}^m |\sum_{j=1}^n a_{ij}x_j| \le \sum_{i=1}^m \sum_{j=1}^n |a_{ij}|\cdot|x_j|$

$= \sum_{j=1}^n \sum_{i=1}^m |a_{ij}|\cdot|x_j| = \sum_{j=1}^n \|A_j\|_1\cdot|x_j| \le \max_{j\in\{1,\cdots,n\}}\|A_j\|_1 \sum_{j=1}^n |x_j| = \max_{j\in\{1,\cdots,n\}}\|A_j\|_1\cdot\|x\|_1$

$\Rightarrow \frac{\|Ax\|_1}{\|x\|_1} \le \max_{j\in\{1,\cdots,n\}}\|A_j\|_1 = \max_{j\in\{1,\cdots,n\}} \sum_{i=1}^m |a_{ij}|$

② 证明 $\exists x$, s.t. $\frac{\|Ax\|_1}{\|x\|_1} = \max_{j\in\{1,\cdots,n\}} \sum_{i=1}^m |a_{ij}|$

若 $\max_{j\in\{1,\cdots,n\}} \sum_{i=1}^m |a_{ij}| = \sum_{i=1}^m |a_{ij_0}|$，则选 x 为 unit vector，且 $x_{j_0} = 1$

证 Derivation for l_2 matrix norm

令 $(\lambda_i, u_i)_{i=1}^n$ 表示 A^TA 的 eigenvalue-eigenvector pairs，有

① $A^TAu_i = \lambda_i u_i$　② $u_i^Tu_i = 1$　③ $u_i^Tu_j = 0$ for $i \neq j$

由此，$\exists$ parameters $(\beta_j)_{j=1}^n$, s.t. $x = \sum_{i=1}^n \beta_i u_i$，可以推出

① $A^TAx = A^TA\sum_{i=1}^n \beta_i u_i = \sum_{i=1}^n \lambda_i\beta_i u_i$

② $x^TA^TAx = (\sum_{i=1}^n \beta_i u_i)^T(\sum_{i=1}^n \lambda_i\beta_i u_i) = \sum_{i=1}^n \lambda_i\beta_i^2$

③ $\|x\|_2^2 = x^Tx = (\sum_{i=1}^n \beta_i u_i)^T(\sum_{i=1}^n \beta_i u_i) = \sum_{i=1}^n \beta_i^2$ 有

$\|A\|_2^2 = \max_{\|x\|_2=1}\|Ax\|_2^2 = \max_{\|x\|_2=1} x^TA^TAx = \max_{x\neq 0}\frac{x^TA^TAx}{x^Tx}$

$\le \lambda_{max}(A^TA)\sum_{i=1}^n \beta_i^2 = \lambda_{max}(A^TA)\|x\|_2^2 = \lambda_{max}(A^TA)$

4. Frobenius norm $\|A\|_F = \left(\sum_{i=1}^m\sum_{j=1}^n |a_{ij}|^2\right)^{\frac{1}{2}} = \sqrt{tr(A^TA)}$

① F-norm 满足 sub-multiplicative：$\|AB\|_F \le \|A\|_F\|B\|_F$

② $\|\cdot\|_F$ is compatible with $\|\cdot\|_2$：$\|Ax\|_F \le \|A\|_F\|x\|_2$

证 Denote $A = (a_{ij})_{i,j}$，$B = (b_{ij})_{i,j}$，则 $AB = (\sum_k a_{ik}b_{kj})_{i,j}$

$\|AB\|_F^2 = \sum_{ij}(\sum_k a_{ik}b_{kj})^2 \le \sum_{ij}(\sum_k a_{ik}^2)(\sum_k b_{kj}^2)$

$= (\sum_{ik} a_{ik}^2)(\sum_{kj} b_{kj}^2) = \|A\|_F^2\cdot\|B\|_F^2$

证 Trace 有以下性质：$trace(AB) = trace(BA)$

Eigenvalue decomposition 有 $A^TA = V\Sigma V^T$.

$\Sigma = diag(\lambda_1, \cdots, \lambda_n)$, $V = [v_1, v_2, \cdots, v_n]$, $V^TV = VV^T = I$

$\|AB\|_F^2 = trace(B^TA^TAB) = trace(BB^TA^TA) = trace(B^TV\Sigma V^TB)$

$= trace(V^TB^TV\Sigma) \le \lambda_{max}(A^TA)\, trace(V^TB^TV)$

$= \lambda_{max}(A^TA)\, trace(VV^TBB^T) = \|A\|_2^2\, trace(BB^T) = \|A\|_2^2\|B\|_F^2$

§3 Condition number

1. Condition number　一个 nonsingular square matrix A 的 condition number 为：

$cond(A) := \|A\|\cdot\|A^{-1}\|$

① 若 A 为 singular，则 $cond(A) = \infty$　② 使用 induced matrix norms

② $cond(A) = (\max_{x\neq 0}\frac{\|Ax\|}{\|x\|})\cdot(\min_{x\neq 0}\frac{\|Ax\|}{\|x\|})^{-1}$

大 condition number $\Rightarrow$ matrix nearly singular

证 $\|A^{-1}\| = (\max_{x\neq 0}\frac{\|A^{-1}x\|}{\|x\|}) = (\max_{y\neq 0}\frac{\|A^{-1}Ay\|}{\|Ay\|})$

$= (\max_{y\neq 0}\frac{\|y\|}{\|Ay\|}) = [(\min_{y\neq 0}\frac{\|Ay\|}{\|y\|})^{-1}]^{-1} = (\min_{y\neq 0}\frac{\|Ay\|}{\|y\|})^{-1}$

2. Condition number 的性质

① $cond(A) = \|A\|\cdot\|A^{-1}\| \ge \|AA^{-1}\| = \|I\| = 1$, $\forall\, A$

② $cond(I) = 1$　③ $cond(\alpha A) = cond(A)$ $\forall$ matrix A 与 $\alpha \in \mathbb{R}$

④ $\forall$ diagonal matrix $D = diag(d)$, $cond(D) = \frac{\max_{i\in\{1,\cdots,m\}}|d_i|}{\min_{i\in\{1,\cdots,m\}}|d_i|}$

3. 计算对称矩阵的 cond$_{l_2}$：若 A symmetric 有特征值

$\{\lambda_1, \cdots, \lambda_n\}$，则 $cond_{l_2}(A) = \frac{\max\{|\lambda_1|, \cdots, |\lambda_n|\}}{\min\{|\lambda_1|, \cdots, |\lambda_n|\}}$

§4 Assessing accuracy

1. Error bound（关于 $'b$）（$Ax = b$, $A\hat{x} = b + \Delta b$）

$$\frac{\|\Delta x\|}{\|x\|} \le cond(A)\cdot\frac{\|\Delta b\|}{\|b\|}$$

证 $\frac{\|\Delta x\|}{\|x\|} = \frac{\|\hat{x}-x\|}{\|x\|} = \frac{\|A^{-1}(b+\Delta b) - A^{-1}b\|}{\|A^{-1}b\|} = \frac{\|A^{-1}\Delta b\|}{\|A^{-1}b\|}$

$\le \|A^{-1}\|\cdot\frac{\|\Delta b\|}{\|A^{-1}b\|} = \|A^{-1}\|\cdot\frac{\|b\|}{\|A^{-1}b\|}\cdot\frac{\|\Delta b\|}{\|b\|}$

$\le \|A^{-1}\|\cdot\|A\|\cdot\frac{\|\Delta b\|}{\|b\|} = cond(A)\cdot\frac{\|\Delta b\|}{\|b\|}$

2. Error bound（关于 A）（$Ax = b$, $(A+\Delta A)\hat{x} = b$）

$$\frac{\|\Delta x\|}{\|\hat{x}\|} \le cond(A)\cdot\frac{\|\Delta A\|}{\|A\|}$$

证 $\frac{\|\Delta x\|}{\|\hat{x}\|} = \frac{\|\hat{x}-x\|}{\|\hat{x}\|} = \frac{\|A^{-1}(b-\Delta A\hat{x}) - A^{-1}b\|}{\|\hat{x}\|} = \frac{\|A^{-1}\Delta A\hat{x}\|}{\|\hat{x}\|}$

$\le \|A^{-1}\|\cdot\|\Delta A\|\cdot\frac{\|\hat{x}\|}{\|\hat{x}\|} = \|A^{-1}\|\cdot\|A\|\cdot\frac{\|\Delta A\|}{\|A\|} = cond(A)\cdot\frac{\|\Delta A\|}{\|A\|}$

3. Conditioning 与 condition number

对于 problem $f(b) = A^{-1}b$ $(= x)$，定义 condition number 为

$cond = \frac{|\text{rel. forward error}|}{|\text{rel. backward error}|} = \frac{\|\hat{f}(b)-f(b)\|/\|f(b)\|}{\|\hat{b}-b\|/\|b\|} = \frac{\|\Delta x\|/\|x\|}{\|\Delta b\|/\|b\|}$

由此可知，cond(A) 为 cond 的“uniform” worst-case bound

$\sup_{b,\Delta b}\text{cond} = \sup_{b,\Delta b}\frac{\|\Delta x\|/\|x\|}{\|\Delta b\|/\|b\|} = cond(A)$

因此，cond(A) 为 independent of b, $\hat{b}$ 的一个 version of cond

4. Some useful tools

- The representation of $AB = (\sum_{k=1}^n a_{ik}b_{kj})_{i,j}$
- Cauchy-Schwartz inequality $(\sum_{k=1}^n a_{ik}b_{kj})^2 \le (\sum_{k=1}^n a_{ik}^2)(\sum_{k=1}^n b_{kj}^2)$
- Eigenvalue-eigenvector decomposition of real symmetric matrices $A = U\Sigma U^\top$
- Property of trace $trace(AB) = trace(BA)$

4. Solving Linear Systems

§1 Triangular linear system

1. triangular matrix ① Lower: $a_{ij}=0$ for $i<j$ ② Upper: $a_{ij}=0$ for $i>j$

2. Forward substitution

① 对于 lower triangular system $Lx=b$, forward substitution 为:

$x_1 = b_1/l_{11}$; $x_i = (b_i - \sum_{j=1}^{i-1} l_{ij}x_j)/l_{ii}$, $i=2,\dots,n$

②
```
for j = 1 to n            /* loop over columns */
  if l_jj = 0 then stop   /* stop if matrix is singular */
  x_j = b_j/l_jj          /* compute solution component */   (1 flop)
  for i = j+1 to n
    b_i = b_i - l_ij x_j  /* update right-hand side */       (2 flops)
  end                     (overwrite the vector b)
end
```
↳ 在计算第j行的未知数x_j后, 对于第j行以下的第i行, 用b_i减去x_j·当前第i行的系数l_{ij}

③ Total flops: $\sum_{j=1}^{n} 2(n-j)+1 = n^2$

3. Backward substitution

① 对于 upper triangular system $Ux=b$, backward substitution 为:

$x_n = b_n/u_{nn}$; $x_i = (b_i - \sum_{j=i+1}^{n} u_{ij}x_j)/u_{ii}$, $i=n-1,\dots,1$

②
```
for j = n to 1            /* loop backwards over columns */
  if u_jj = 0 then stop   /* stop if matrix is singular */
  x_j = b_j/u_jj          /* compute solution component */
  for i = 1 to j-1
    b_i = b_i - u_ij x_j  /* update right-hand side */
  end                     (overwrite the vector b)
end
```
↳ 在计算第j行的未知数x_j后, 对于第j行以上的第i行, 用b_i减去x_j·当前第i行的系数u_{ij}

③ Total flops: $\sum_{j=1}^{n} 2(n-j)+1 = n^2$

§2 Transform general systems

1. Permutation: $P^{-1}=P^T$

① $PAx=Pb$ 相当于对 A 和 b 的 rows 进行 reorder, solution x 不变

② $APx'=b$ 相当于对 A 的 columns 进行 reorder, solution $x'=P^T(A^{-1}b)$ 改变

2. diagonal scaling (nonsingular diagonal matrix D)

① $DAx=Db$ 相当于对 A 的 rows 和 b 的乘以 D 的 diagonal entry, x 不变

② $ADx'=b$ 相当于对 A 的 columns 乘以 D 的 diagonal entry, $x'=D^{-1}(A^{-1}b)$

§3 Elementary elimination matrix

1. $M_k a = \begin{bmatrix}1&\cdots&0&0&\cdots&0\\ \vdots&\ddots&\vdots&\vdots&&\vdots\\ 0&\cdots&1&0&\cdots&0\\ 0&\cdots&-m_{k+1}&1&\cdots&0\\ \vdots&&\vdots&\vdots&\ddots&\vdots\\ 0&\cdots&-m_n&0&\cdots&1\end{bmatrix}\begin{bmatrix}a_1\\ \vdots\\ a_k\\ a_{k+1}\\ \vdots\\ a_n\end{bmatrix} = \begin{bmatrix}a_1\\ \vdots\\ a_k\\ 0\\ \vdots\\ 0\end{bmatrix}$

$m_i = \frac{a_i}{a_k}$ for $\forall i = k+1,\dots,n$; a_k 称为 pivot, 必须为 nonzero; M_k 为 elementary elimination matrix

2. Elementary elimination matrix 的性质

① $M_k = I - m_k e_k^T$, $m_k=[0,\dots,0,m_{k+1},\dots,m_n]^T$, e_k 为 k-th unit vector

② $L_k := M_k^{-1} = I + m_k e_k^T$

Sherman-Morrison formula: $(A+bc^T)^{-1} = A^{-1} - \frac{A^{-1}bc^TA^{-1}}{1+c^TA^{-1}b}$

③ $M_kM_j = I - m_ke_k^T - m_je_j^T$ $(k<j)$ (M_k与M_j的"union")

④ $L_kL_j = I + m_ke_k^T + m_je_j^T$ $(k<j)$ (L_k与L_j的"union")

可推广至 $L_1L_2\cdots L_n = I + \sum_{i=1}^{n} m_ie_i^T$

§4 LU Factorization by Gaussian elimination

1. Gaussian elimination

$M_{n-1}\cdots M_1 A x = M_{n-1}\cdots M_1 b \Rightarrow MAx = Mb$ 其中 $MA=U$

用 back-substitution 得到最终结果

2. LU Factorization

① 矩阵 $L_i = M_i^{-1}$ 为 unit lower triangular

② $M = M_{n-1}\cdots M_1 \Rightarrow L = M^{-1} = M_1^{-1}\cdots M_{n-1}^{-1} = L_1\cdots L_{n-1}$

注: 根据 L_i 的性质, $L = L_1\cdots L_{n-1}$ 可视作 $L_1,\dots,L_{n-1}$ 的 union

③ 由 Gaussian elimination $MA=U \Rightarrow A = LU$, L 为 unit Δ

3. 利用 LU factorization 解 linear system

$LUx=b \Rightarrow$ solve $Ly=b$ (forward sub) $\Rightarrow$ solve $Ux=y$ (backward sub)

4. LU factorization algorithm

1° 逐步分析

① Step 1: $A^1 = \underbrace{\begin{bmatrix}1 & 0\\ m_1 & I\end{bmatrix}}_{M_1}\underbrace{\begin{bmatrix}a_{11} & A(1,2:n)\\ * & A(2:n,2:n)\end{bmatrix}}_{A} = \begin{bmatrix}a_{11} & A(1,2:n)\\ 0 & m_1 A(1,2:n)+A(2:n,2:n)\end{bmatrix}$

其中 $m_1 = -A(2:n,1)/a_{11}$

② Step 2: $A^2 = \begin{bmatrix}1&0&0\\0&1&0\\0&m_2&I\end{bmatrix}\begin{bmatrix}a_{11}^1 & a_{12}^1 & A^1(1,3:n)\\ 0 & a_{22}^1 & A^1(2,3:n)\\ 0 & * & A^1(3:n,3:n)\end{bmatrix} = \begin{bmatrix} a_{11}^1 & a_{12}^1 & A^1(1,3:n)\\ 0 & a_{22}^1 & A^1(2,3:n)\\ 0&0& m_2A^1(2,3:n)+A^1(3:n,3:n)\end{bmatrix}$

其中 $m_2 = -A^1(3:n,2)/a_{22}^1$

- overwrite A, 使 output 矩阵的 upper triangular part 对应 U
- 将 $l_k = -m_k$ 储存在 output 矩阵的 lower triangular part

2° Algorithmic procedure

```
for k = 1 to n-1               /* loop over columns */
  if a_kk = 0 then stop        /* stop if pivot is zero */
  for i = k+1 to n             /* compute multipliers
    l_ik = a_ik/a_kk  (= -m_k)    for current column */
  end
  for j = k+1 to n
    for i = k+1 to n           /* apply transformation
      a_ij = a_ij - l_ik a_kj     to remaining submatrix */
    end
  end
end
```

Total flops: $\frac{2}{3}n^3+O(n^2) = O(n^3)$

证 每轮 iter: $n-(k+1)+1$ (计算 $l_k=-m_k$) $+ 2(n-k)^2$ (更新 block) $= (n-k)+2(n-k)^2$

共需要 $\sum_{k=1}^{n-1}(n-k)+2(n-k)^2 = \sum_{i=1}^{n-1} 2i^2+i = 2\cdot\frac{n(n-1)(2n-1)}{6} + \frac{n(n-1)}{2} = \frac{2}{3}n^3+O(n^2)$

5. 比较: inversion 与 factorization

① 计算 A^{-1} 的方法: 求解 $A^{-1} \Leftrightarrow$ 求解 $Ax_i = e_i$ for all i

② 计算 A^{-1} 的 complexity: (对 A LU分解 + n次 forward & backward substitution) $O(n^3)$

③ 比较: inversion 与 factorization

(1) Initial cost: LU factorization of A < 计算 A^{-1}

(2) Later cost: forward & backward substitution ≈ 计算 $A^{-1}b_i = O(n^2)$

§5 Partial pivoting

1. Pivoting strategy: partial pivoting

每个 stage, 选取该 column 对角线以下绝对值最大的 entry 作为 pivot

(↓ error propagation) 每个 M_k 前都要乘以 permutation matrix P_k 以交换两行

2. Partial pivoting: 第一种方法 仍然得到 $MA=U$, 此时

① $M = M_{n-1}P_{n-1}\cdots M_1P_1$ ② $L = M^{-1}$ 不一定为 lower triangular

3. Partial pivoting: 第二种方法 希望得到 $PA = LU$, 此时

① $P = P_{n-1}\cdots P_1$ 对 A 的 rows 重排列 ② $L = PM^{-1}$ 为 lower triangular

1° 逐步分析 (step k) ($A' = P_{k-1}\cdots P_1 A$)

$P_k A_{k-1}' = P_k L_k' P_k^T P_k U_k'$ (L_k' 合并了之前所有的 L_i, P_i, P_i^T)

由于 $L_k = I + m_ke_k^T$, 对于 $\forall j>k$, 有

$P_jL_kP_j^T = P_jP_j^T + P_jm_ke_k^TP_j^T = I + (P_jm_k)e_k^T = \tilde{L}_k$

$P_kL_{k-1}'P_k^T = P_k(\tilde{L}_{k-1}\tilde{L}_{k-2}\cdots\tilde{L}_1)P_k^T = (P_k\tilde L_{k-1}P_k^T)(P_k\tilde L_{k-2}P_k^T)\cdots(P_k\tilde L_1P_k^T)$

相当于交换了 L_i 对应的两行 multiplier

eg. 对下列矩阵做 LU factorization (with pivoting) $A = \begin{bmatrix}1&-4&2\\-2&3&0\\4&-1&2\end{bmatrix}$

① swap row 1 & 3 并求解 L_1 & U_1

$P_1A = \begin{bmatrix}4&-1&2\\-2&3&0\\1&-4&2\end{bmatrix} = \begin{bmatrix}1&0&0\\-1/2&1&0\\1/4&0&1\end{bmatrix}\begin{bmatrix}4&-1&2\\0&5/2&1\\0&-15/4&3/2\end{bmatrix}$

② swap row 2 & 3 并求解 L_2 & U_2

$P_2P_1A = \begin{bmatrix}4&-1&2\\1&-4&2\\-2&3&0\end{bmatrix} = \begin{bmatrix}1&0&0\\1/4&1&0\\-1/2&0&1\end{bmatrix}\begin{bmatrix}4&-1&2\\0&-15/4&3/2\\0&5/2&1\end{bmatrix} = \begin{bmatrix}1&0&0\\1/4&1&0\\-1/2&-2/3&1\end{bmatrix}\begin{bmatrix}4&-1&2\\0&-15/4&3/2\\0&0&2\end{bmatrix}$

③ 因此, $PA = \begin{bmatrix}4&-1&2\\1&-4&2\\-2&3&0\end{bmatrix} = \begin{bmatrix}1&0&0\\1/4&1&0\\-1/2&-2/3&1\end{bmatrix}\begin{bmatrix}4&-1&2\\0&-15/4&3/2\\0&0&2\end{bmatrix}$

2° Total flops: $\frac{2}{3}n^3+O(n^2)$

§6 LU Factorization: Summary & Stability

1. LU Factorization with pivoting 的存在性定理

若 $A\in R^{n\times n}$ nonsingular matrix, 则∃ $PA=LU$ 且 $|l_{ij}|\le 1, \forall i,j$ & L 为 unit Δ

注: 该定理为存在性定理, LU分解可能不唯一.

证 ① Base case: $n=1$ 对于 $A\in R$, 有 $A = 1\cdot A$, 因此选取 $P=L=1$, $U=A$ 即可

② Inductive step: $n-1\to n$

令 $A\in R^{n\times n}$ 为 nonsingular, 则其第一列 $A(:,1)$ 不全为 0, 因此,

$\alpha := \max_{i=1,\dots,n}|A(i,1)| \neq 0$, ∃ $P_1A = \begin{bmatrix}\alpha & u^T\\ v & A_*\end{bmatrix}$

进行 LU-factorization 的第一步, 有

$P_1A = \begin{bmatrix}\alpha & u^T\\ v & A_*\end{bmatrix} = \underbrace{\begin{bmatrix}1 & \\ v/\alpha & I\end{bmatrix}}_{=L_1}\begin{bmatrix}\alpha & u^T\\ 0 & A_*-vu^T/\alpha\end{bmatrix}$

$B = A_* - vu^T/\alpha$ 也为 nonsingular, $P_*B = P_*(A_*-\frac{1}{\alpha}vu^T) = L_*U_*$

$\underbrace{\begin{bmatrix}1 & \\ & P_*\end{bmatrix}P_1}_{P}A = \begin{bmatrix}1&\\&P_*\end{bmatrix}\begin{bmatrix}1&\\ v/\alpha & I\end{bmatrix}\begin{bmatrix}1 & \\ & P_*^T\end{bmatrix}\begin{bmatrix}1&\\&P_*\end{bmatrix}\begin{bmatrix}\alpha&u^T\\0&B\end{bmatrix} = \underbrace{\begin{bmatrix}1&\\P_*v/\alpha & L_*\end{bmatrix}}_{L}\underbrace{\begin{bmatrix}\alpha & u^T\\0&U_*\end{bmatrix}}_{U}$

2. Pivoting 与 stability

① Gaussian elimination without pivoting 为 unstable; with pivoting 为 stable

② accuracy 仍取决于 A 的 conditioning

§7 Cholesky factorization

1. Cholesky factorization 若 $A=A^T$ and $x^TAx>0$, $\forall x\neq 0$ 则 $A=LL^T$

2. Cholesky factorization 的计算

① 2×2 case: $l_{11} = \sqrt{a_{11}}$, $l_{21} = a_{21}/l_{11}$, $l_{22} = \sqrt{a_{22}-l_{21}^2}$

② n×n case: $\hat A = \begin{bmatrix}LL^T & Ll\\ l^TL^T & l_{nn}^2+\|l\|^2\end{bmatrix}$, $l = L^{-1}a$ 与 $l_{nn} = \sqrt{a_{nn}-\|l\|^2}$

3. Cholesky algorithm 的 features ① 不需要 pivoting ② 仅需储存 upper (或 lower) triangular part ③ Total flops: $\frac{1}{3}n^3+O(n^2)$

$\begin{bmatrix}a&b\\c&d\end{bmatrix}^{-1} = \frac{1}{ad-bc}\begin{bmatrix}d&-b\\-c&a\end{bmatrix}$

5. Linear Least Square

§1 Linear least square

1. Linear least square 对于 overdetermined linear system: $Ax=b$, $A\in R^{m\times n}$, $m>n$

Least squares solution x 会 minimize residual vector $r = Ax-b$ 的 $\|\cdot\|_2$:

$\min_x \|r\|_2^2 = \min_x \|Ax-b\|_2^2$

2. Euclidean projection of b onto range(A)

$P_{range(A)}(b) = y^* := \arg\min_{y\in range(A)} \frac{1}{2}\|y-b\|_2^2$

由于 $y^*\in range(A)$, 则∃ x^* s.t. $y^*=Ax^*$

证 uniqueness: 利用 p 为 quadratic, second-order Taylor expansion 为 exact

$p(\hat y) = p(y^*) + \underbrace{(y^*-b)^T(\hat y-y^*)}_{=0\ \text{(optimality condition)}} + \frac12(\hat y-y^*)^T(\hat y-y^*) \Rightarrow \|\hat y-y^*\|=0$

证 existence: 先将 y^* 的范围缩到 compact set 上, 再利用最值定理

$L := \{y: p(y)\le p(0)\} \cap range(A)$

3. least square solution 对于 least square problem $\min_x\|Ax-b\|_2^2$

① 一定有 solution ② x^* 为 optimal 当且仅当满足 normal equations: $A^TAx^* = A^Tb$

③ solution 为 unique 当且仅当 A 满秩 (rank(A)=n) $x^* = (A^TA)^{-1}A^Tb$

若 rank < n, 即 A 为 rank-deficient, 则 solution 不唯一

证 "⇒" 令 $f(x) = \|r\|_2^2 = (Ax-b)^T(Ax-b) = x^TA^TAx - 2x^TA^Tb + b^Tb$

若 x^* 为 (local) solution, 其必须满足 $\nabla f(x^*) = 2A^TAx^* - 2A^Tb = 0$

"⇐" 令 x^* 为 normal equation 的一个 solution, 即 $f'(x)=0$

$f(x)-f(x^*) = \nabla f(x^*)^T(x-x^*) + \frac12(x-x^*)^T\nabla^2f(x)(x-x^*)$

$= 0 + \frac12(x-x^*)^T\cdot 2A^TA(x-x^*) = \|A(x-x^*)\|_2^2 \ge 0$, $\forall x\in R^n$

§2 Orthogonal projectors 与 conditioning

1. Orthogonal projectors ① $P^2=P$ ② $P^T=P$

2. vector 的分解 $P_\perp = I-P$ 为 orthogonal projector onto range$(P)^\perp$

3. 最小二乘问题的正交投影算子 $P_{range(A)} = P = A(A^TA)^{-1}A^T$

① $b = Pb + P_\perp b = Ax^* + (b-Ax^*) = y^* + (-r)$

② $y^* = P_{range(A)}(b) = Ax^* = A(A^TA)^{-1}A^Tb = P_{range(A)}\cdot b$

4. Pseudoinverse 对于 non-square m×n matrix A

若 rank(A)=n 则 pseudoinverse 为: $A^+ = (A^TA)^{-1}A^T$

注: $x^* = (A^TA)^{-1}A^Tb = A^+b$

5. condition number 对于 non-square m×n matrix A

若 rank(A)=n 则 condition number 为: $cond(A) := \|A\|_2\cdot\|A^+\|_2$

① 若 rank(A)<n, 则 $cond(A) := \infty$

② condition number 衡量了 closeness to rank deficiency

6. Conditioning 与 sensitivity (条件数的应用) θ 为 b 与 $y=Ax$ 的夹角

$\frac{\|\Delta x\|_2}{\|x\|_2} \le \frac{cond(A)}{\cos(\theta)}\cdot\frac{\|\Delta b\|_2}{\|b\|_2}$, $\cos(\theta) = \frac{\|y\|_2}{\|b\|_2}$

6. Solving Least Square

§1 Normal equation method

1. Normal equation method: 用 $A^TA = LL^T$ 解 $A^TAx = A^Tb$

2. 缺点: ① 求 A^TA 与 A^Tb 时 lose information ② $cond(A^TA) = (cond(A))^2$

§2 Orthogonalization methods

1. Orthogonal matrix ① Q 的列向量为 R^n 的规范正交基 ② $Q^{-1}=Q^T$

③ $\|Qx\| = \|x\|$ ④ $\langle Qx, Qy\rangle = \langle x,y\rangle$ ⑤ AB 也为正交矩阵

2. Orthogonal transformation $\min_x\|Ax-b\|_2^2 = \min_x\|TAx-Tb\|_2^2$

$T_{m\times m}A_{m\times n} = \begin{bmatrix}R_{n\times n}\\ 0_{(m-n)\times n}\end{bmatrix}$, $T_{m\times m}b_{m\times 1} = \begin{bmatrix}c_{1\,n\times 1}\\ c_{2\,(m-n)\times 1}\end{bmatrix}$ $= \min_x\|Rx-c_1\|_2^2 + \|c_2\|_2^2$ ⟺ 解 $Rx=c_1$; $\|c_2\|_2^2 = \|Ax-b\|_2^2 = \|r\|_2^2$

3. QR Factorization $A_{m\times n} = Q_{m\times m}\begin{bmatrix}R_{n\times n}\\0_{(m-n)\times n}\end{bmatrix}$; $\min_x\|Ax-b\|_2^2 \Leftrightarrow \min_x\|Rx-c_1\|_2^2$; 其中 $Q^Tb = \begin{bmatrix}c_1\\c_2\end{bmatrix}$

4. Reduced QR Factorization $Q_{m\times m} = [Q_{r\,m\times n}\ \ Q_{n\,m\times(m-n)}]$

$A = Q\begin{bmatrix}R\\0\end{bmatrix} = [Q_r\ Q_n]\begin{bmatrix}R\\0\end{bmatrix} = Q_rR$; $\min_x\|Ax-b\|_2^2 \Leftrightarrow Rx = c_1 = Q_r^Tb$

5. Reduced QR Factorization 的性质 ① A 有 full rank ⟺ $r_{ii}\neq0$, $\forall i$

② 若 A 有 full rank, Q_r 的 columns 为 range(A) 的 orthonormal basis; Q_n 的 columns 为 range$(A)^\perp$ 的 orthonormal basis

③ $Q_rQ_r^T$ 为 range(A) 的 orthogonal projector

§3 Householder QR factorization

1. reflection matrix: $H = I - 2P$ ① $H = H^T = H^{-1}$ ② Hx 将 x 关于 range$(P_\perp)$ reflect

2. Householder matrix: $H_v = I - 2\frac{vv^T}{\|v\|_2^2}$ for $v\in R^m\setminus\{0\}$

① $\frac{vv^T}{\|v\|_2^2}$ 为 orthogonal projector ② $H_v = H_v^T = H_v^{-1}$ ③ $\begin{bmatrix}I_r & 0\\0 & H_v\end{bmatrix}$

3. Householder transformation

① $\alpha = \pm\|a\|_2$ ② $v = a - \alpha e_1$ ⟺ $H_va = \alpha e_1$

4. Householder QR factorization (stable)

```
for k = 1 to n
  Set A = Q_k A, where          /* update A */
      Q_k = [ I_{k-1}         ]
            [          H_{v_k}]
  I_{k-1} is (k-1)-dim. identity matrix; H_{v_k} is the Householder trans-
  formation using the first column of A_{k-1} to build v_k.
end
```

实际的 update ① 求 $w = \frac{\sqrt2 v_k}{\|v_k\|}$ ② $H_vA_{k-1} = A_{k-1} - w(w^TA_{k-1}) = \begin{bmatrix}\alpha_k & *\\0 & A_k\end{bmatrix}$

① $Q = (Q_n\cdots Q_1)^T = Q_1^T\cdots Q_n^T = Q_1\cdots Q_n$

② Flops $= \sum_{k=1}^n 3(m-k+1)+1+[4(m-k+1)-1](n-k) \approx 2mn^2 - \frac23n^3$

③ normal equation method: $mn^2 + \frac13n^3$ flops

§4 Givens QR Factorization

$G_{i,k}a = \begin{bmatrix}I&&&&\\&c&&s&\\&&I&&\\&-s&&c&\\&&&&I\end{bmatrix}a = \begin{bmatrix}\vdots\\ \alpha\\ \vdots\\ 0\\ \vdots\end{bmatrix}$ ← i, ← k; $s = \frac{a_k}{\sqrt{a_i^2+a_k^2}}$, $c = \frac{a_i}{\sqrt{a_i^2+a_k^2}}$, $\alpha = \sqrt{a_i^2+a_k^2}$

$\underbrace{(G_{n,m}\cdots G_{n,n+2}G_{n,n+1})\cdots(G_{2,m}\cdots G_{2,4}G_{2,3})(G_{1,m}\cdots G_{1,3}G_{1,2})}_{=:Q^T}A = R$

Use Givens transformation to do the QR factorization of matrix

$A = \begin{pmatrix}4&2\\3&4\\0&\sqrt5\end{pmatrix}$

Step 1: Find c_{12}, s_{12} s.t. $G(1,2,c_{1,2},s_{1,2})A_{:,1} = (*\ 0\ *)^T$

a) $c_{1,2} = \frac{A_{1,1}}{\sqrt{A_{1,1}^2+A_{2,1}^2}} = \frac45$, $s_{1,2} = \frac{A_{2,1}}{\sqrt{A_{1,1}^2+A_{2,1}^2}} = \frac35$

b) $\tilde A = G(1,2,c_{1,2},s_{1,2})A = \begin{pmatrix}c_{1,2}A_{1,:}+s_{1,2}A_{2,:}\\ -s_{1,2}A_{1,:}+c_{1,2}A_{2,:}\\ A_{3,:}\end{pmatrix} = \begin{pmatrix}5&4\\0&2\\0&\sqrt5\end{pmatrix}$

Step 2: Find $c_{1,3}, s_{1,3}$ s.t. $G(1,3,c_{1,3},s_{1,3})\tilde A_{:,1} = (*\ 0\ 0)^T$

$c_{1,3}=1$, $s_{1,3}=0$, $\hat A = G(1,3,c_{1,3},s_{1,3})\tilde A = I\tilde A = \tilde A$

Step 3: Find $c_{2,3}, s_{2,3}$ s.t. $G(2,3,c_{2,3},s_{2,3})\hat A_{:,2} = (*\ *\ 0)^T$

$c_{2,3} = \frac23$, $s_{2,3} = \frac{\sqrt5}{3}$, $R = G(2,3,c_{2,3},s_{2,3})\hat A = \begin{pmatrix}5&4\\0&3\\0&0\end{pmatrix}$

G不满足 $G=G^T$

$Q = G_{1,2}^TG_{1,3}^TG_{2,3}^T$, $R = \begin{pmatrix}5&4\\0&3\\0&0\end{pmatrix}$

§5 Gram-Schmidt QR Factorization

1. Gram-Schmidt process 有线性无关组 $\{u_1,\dots,u_m\}$, 求规范正交集 $\{v_1,\dots,v_m\}$

① $v_1 = \frac{u_1}{\|u_1\|}$ ② $p_1 = \langle u_2, v_1\rangle v_1$, $r_1 = u_2 - p_1 \perp Span(u_1)$, $v_2 = \frac{r_1}{\|r_1\|} = \frac{u_2-p_1}{\|u_2-p_1\|}$

ⓜ $p_{m-1} = \langle u_m, v_1\rangle v_1 + \cdots + \langle u_m, v_{m-1}\rangle v_{m-1}$, $r_{m-1} = u_m - p_{m-1} \perp Span(v_1,\dots,v_{m-1})$

$v_m = \frac{r_{m-1}}{\|r_{m-1}\|} = \frac{u_m-p_{m-1}}{\|u_m-p_{m-1}\|}$ $Span(\{u_1,\dots,u_n\}) = Span(\{v_1,\dots,v_n\})$

2. R 的求解 $r_{ij} := q_i^Ta_j$ (a_j 在 q_i 上的投影长度), $i<j$

$r_{jj} := \|\tilde q_j\| = \|a_j - \sum_{i=1}^{j-1} r_{ij}q_i\|$ (a_j 在 q_j 上的投影长度)

```
input: A collection of vectors a_1, ..., a_n
q~_1 = a_1, q_1 = q~_1/||q~_1||.
for j = 2 to n
    q~_j = a_j - sum_{i=1}^{j-1} (q_i^T a_j) q_i.     (q_i^T a_j = r_ij, i = 1,...,j-1)
    q_j = q~_j/||q~_j||.                               (||q~_j|| = r_jj)
end
output: q_1, ..., q_n.
```

$a_j = \begin{bmatrix}q_1 & \cdots & q_j & \cdots & q_n\end{bmatrix}_{m\times n}\begin{bmatrix}r_{1j}\\ \vdots\\ r_{jj}\\ \vdots\\ 0\end{bmatrix}_{n\times 1}$

$A = Q_{m\times n}\begin{bmatrix}r_{11} & \cdots & r_{1j} & \cdots & r_{1n}\\ & \ddots & \vdots & & \vdots\\ & & r_{jj} & & r_{jn}\\ & & & \ddots & \vdots\\ 0 & & 0 & & r_{nn}\end{bmatrix}_{n\times n}$

3. Classical Gram-Schmidt QR Factorization

```
for j = 1 to n
    q_j = a_j
    for i = 1 to j-1
        r_ij = q_i^T a_j
        q_j = q_j - r_ij q_i
    end
    r_jj = ||q_j||_2
    q_j = q_j/r_jj
end
```

缺点 ① loss of orthogonality ② 需要分开储存 A, Q, 和 R

```
for i = 1 to n
    r_ii = ||a_i||_2
    q_i = a_i/r_ii
    for j = i+1 to n
        r_ij = q_i^T a_j
        a_j = a_j - r_ij q_i
    end
end
```

通常更 stable

4. Modified Gram-Schmidt QR Factorization

$\tilde q_j = (I - q_{j-1}q_{j-1}^T)\cdots(I-q_2q_2^T)(I-q_1q_1^T)a_j = (I - \sum_{i=1}^{j-1} q_iq_i^T)a_j$

在每次算出一个 q_j 后, 我们算出 ① $r_{j(j+1)},\dots,r_{jn}$ ② $(I-q_jq_j^T)a_j$

5. Flops: inner loop 中的 update: $(n-i)(4m-1)$; 总计: $\approx 2mn^2 - \frac12 n$

证 考虑 $y^* = P_{range(A)}(b) := \arg\min_{y\in range(A)} \frac12\|y-b\|_2^2$ ∃ x^* s.t. $y^* = Ax^*$

令 $p(y) = \frac12\|y-b\|_2^2$, 则 $\nabla p(y^*)(y-y^*) = (y^*-b)^T(y-y^*) = 0$,

$\forall y\in range(A)$. 代入 $y^*=Ax^*$, $y=Ax$, 有

$(Ax^*-b)^TA(x-x^*) = (A^TAx^*-A^Tb)^T(x-x^*) = 0$, $\forall x\in R^n$

令 $x = A^TAx^*-A^Tb+x^*$, 有 $\|A^TAx^*-A^Tb\|^2 = 0$

§6 QR factorization 的存在性与唯一性

1. 存在性：所有 matrix $A \in \mathbb{R}^{m\times n}(m\geq n)$ 均有 full QR factorization
注：① Gram-Schmidt 可得到 reduced QR factorization $A=QR$，QR 可被扩展
② 若 A 为 rank deficient ($\text{rank}(A)<n$)，则 QR factorization 仍然存在
2. 唯一性：所有 matrix $A\in\mathbb{R}^{m\times n}(m\geq n)$ with full rank 均有唯一的 reduced QR factorization $A=QR$ with $r_{jj}>0$
注：① 通过 Gram-Schmidt 可以得到 unique 的 Q 与 R（除了 r_{ij} 的 sign）

§7 Rank deficiency 得到的 R 为 singular 最小二乘问题的解有多个 vector 'x

1. Pivoting：选取 remaining unreduced submatrix 中 $\|\cdot\|_2$ 最大的列构建 V_k
① 若 $\text{rank}(A)=r<n$，则 r 步以后，remaining unreduced 的 norms 会（接近）为 0
② Orthogonal factorization 的形式为
$$Q^TAP=\begin{bmatrix}R & S\\ 0 & 0\end{bmatrix}\quad (R\in\mathbb{R}^{r\times r})$$
R 为一个 $r\times r$ 上三角 nonsingular 矩阵
P 为 column interchanges 的 permutation 矩阵
2. 解 basic solution：① 求解 $R_{r\times r}{}'y='c_1$ 其中 $\begin{bmatrix}'c_1\\ 'c_2\end{bmatrix}=Q^Tb$，$'c_1\in\mathbb{R}^r$
② 再求解 $'x$：$'x=P_{n\times n}\begin{bmatrix}'y\\ 0\end{bmatrix}$

7. Eigenvalue problem

§1 Complex number

1. notations ① $\bar{\alpha}=a-b\mathrm{i}$ ② $|\alpha|=\sqrt{a^2+b^2}=\sqrt{\alpha\bar{\alpha}}$
③ $\|'z\|=\sqrt{'z^H{}'z}$，$'z^H='z^T$ ④ $(AC)^H=C^HA^H$
2. conjugate transpose ① $(A^H)^H=A$ ② $(\alpha A+\beta B)^H=\bar{\alpha}A^H+\bar{\beta}B^H$
3. Hermitian matrix 若 $M^H=M$，则称 M 为 Hermitian matrix.
注：① Hermitian matrix 主对角线上的元素为实数.
② Hermitian matrix 的特征值为实数
③ Hermitian matrix 的秩、逆矩阵也为 Hermitian matrix
4. unitary matrix U 的列向量构成 $\mathbb{C}^n$ 中的一个 orthonormal set
U is unitary $\iff U^HU=I\iff U^{-1}=U^H$

§2 Characteristic polynomial & multiplicity

1. Characteristic polynomial：$p(\lambda)=\det(A-\lambda I)=0$
注：① 一定有 n 个 eigenvalues，但 eigenvalues 不一定 distinct 或 real
② 若 A real，且 eigenvalue λ real，则对应 eigenvector v 也 real
③ 一个 real matrix，若 $\alpha+\beta\mathrm{i}$ is eigenvalue，则 $\alpha-\beta\mathrm{i}$ 也是
2. Algebraic multiplicity & Geometric multiplicity
令 $\{\mu_1,\cdots,\mu_k\}$，$k\leq n$ 表示 A 的 eigenvalue 的 (distinct) 取值的集合，则
① Algebraic multiplicity：$r_i=|\{\lambda_j:\lambda_j=\mu_i\}|$，$\forall\ i=1,\cdots,k$
② Geometric multiplicity：$\gamma_i:=\dim(\text{null}(A-\mu_iI))$，$\forall\ i=1,\cdots,k$
注：subspace $\text{null}(A-\mu_iI)$ 被称为 μ_i eigenspace
③ 代数重数大于等于几何重数：$r_i\geq\gamma_i$ for all $i=1,\cdots,k$
3. defective matrix：若 $r_i>\gamma_i$ for some i
注：① $r_i=\dim(\text{null}[(A-\lambda_iI)^n])$ for $\forall$ large enough n
② 在 Jordan normal form 中，$r_i=\lambda_i$ 在 diagonal 中出现的次数
$\gamma_i=\lambda_i$ 对应的 Jordan block 数

§3 Eigendecomposition & diagonalizability

1. Eigendecomposition / diagonalizability：$\exists$ nonsingular $V\in\mathbb{C}^{n\times n}$ s.t.
$$A=V\Lambda V^{-1}\Rightarrow AV=V\Lambda\Rightarrow A'v_i=\lambda_i{}'v_i$$
2. 一个性质：不同 eigenvalue 的 eigenvectors 一定 linearly independent
证 令 $A'v_1=\lambda_1{}'v_1$，$A'v_2=\lambda_2{}'v_2$，且 $\lambda_1\neq\lambda_2$，$'v_1\neq 0$，$'v_2\neq 0$
假设 v_1 与 v_2 linearly dependent，则 $\exists\,\alpha\neq 0$，s.t. $'v_1=\alpha\,'v_2$
$$\Rightarrow\begin{cases}A'v_1=\lambda_1{}'v_1=\lambda_1\alpha\,'v_2\\ A'v_1=\alpha A'v_2=\alpha\lambda_2{}'v_2\end{cases}\Rightarrow\lambda_1=\lambda_2\ (\text{若}\ \|'v_1\|\neq 0)$$
(contradiction)
3. Similarity transformation：$A,B\in\mathbb{C}^{n\times n}$ similar 若
$\exists$ nonsingular $S\in\mathbb{C}^{n\times n}$ s.t. $B=S^{-1}AS$，则
① A,B 的 eigenvalues 相同
② 若 $'u$ 为 B 的 eigenvector，则 $'v=S'u$ 为 A 的 eigenvector
③ A,B 的 characteristic equation，重数 相同
注：A 有 eigendecomposition 若 A is similar to $\Lambda=\text{diag}(\lambda_1,\cdots,\lambda_n)$
证 ①：令 $B=S^{-1}AS$，则 $P_B(\lambda)=\det(S^{-1}AS-\lambda I)$
$=\det[S^{-1}(A-\lambda I)S]=\det(S^{-1})\det(A-\lambda I)\det(S)=\det(A-\lambda I)$
4. diagonalizability：$A=V\Lambda V^{-1}\iff A$ is not defective
证 "$\Rightarrow$" $A=V\Lambda V^{-1}\iff$ similar to Λ $\because\Lambda$ non-def $\therefore A$ non-def
证 "$\Leftarrow$" A non-def $\Rightarrow$ 有 n 个 linearly independent 的 eigenvectors
$\because$ 与不同 eigenvalue 相关联的 eigenvectors 一定 linearly independent
每个 eigenvalue 可对应 $r_i=\gamma_i$ 多个 linearly independent eigenvectors
$\therefore$ 因此可以排列 V 中的 eigenvectors 并写出 $A=V\Lambda V^{-1}$
5. Unitary diagonalization：$\exists$ unitary matrix $Q\in\mathbb{C}^{n\times n}$ s.t. $A=Q\Lambda Q^H$
6. Unitary diagonalizability：Hermitian ($A^H=A$) unitarily diagonalizable
且所有 eigenvalues 均为 real.

§4 Schur factorization

1. Schur factorization：$A=QTQ^H$ (A 与 T 的特征值相同)
2. 存在性：所有 square matrix $A\in\mathbb{C}^{n\times n}$ 均有 Schur factorization

§5 不改变 eigenvalue problem 的 transformations

1. Shifts：$(A-\sigma I)'v=(\lambda-\sigma)'v$ (eigenvalues 也 shift σ)
2. Inversion：$A^{-1}v=\lambda^{-1}{}'v$ (eigenvalues 也 inverse)
3. Powers：$A^k{}'v=\lambda^k{}'v$ (eigenvalues 也取 k-th power)
4. Polynomial transformation：$p(A)'v=p(\lambda)'v$
5. Similarity transformation：$A=Q\Lambda Q^H$，$A=QTQ^H$

§6 Power iteration / Power method

1. 基本介绍：重复对一个 nonzero initial vector $'x^0$ 左乘 matrix A
$'x^k=A'x^{k-1}$. 若 A 有唯一的 eigenvalue of maximum modulus λ_1
其对应的 eigenvector 为 v_1，则会收敛至 $\alpha'v_1$
2. 原理：① Assumptions：non-defective + $|\lambda_1|>|\lambda_2|\geq\cdots$
② Step 1：$'x^0=\sum_{i=1}^n\alpha_i{}'v_i$ (A non-defective) ③ Step 2：
$$'x^k=A^k{}'x^0=\sum_{i=1}^n\alpha_iA^k{}'v_i=\lambda_1^k\left[\alpha_1{}'v_1+\sum_{i=2}^n\left(\frac{\lambda_i}{\lambda_1}\right)^k\alpha_i{}'v_i\right]$$
$|\cdot|<1\Rightarrow\to 0$
3. Power iteration 的缺点：power iteration 会 fail：
① starting vector $'x^0$ 没有 $'v_1$ 方向的分量（即 $\alpha_1=0$）
② 不止一个 eigenvalue 有 maximum modulus，会收敛至线性组合
③ 对于 real matrix A 和 real starting vector $'x^0$，不会收敛至 complex vector

§7 Rayleigh quotient 与 normalized power iteration (仅考虑 real case)

1. Rayleigh quotient：$'x\cdot\lambda\approx A'x\Rightarrow\min_\lambda\|'x\lambda-A'x\|_2^2$
$$\Rightarrow\lambda=\frac{'x^TA'x}{'x^T{}'x}=\frac{'x^TA'x}{\|'x\|^2}=:r('x)\quad\left(\frac{'x^HA'x}{'x^H{}'x}=\frac{'x^HA'x}{\|'x\|^2}\right)$$
2. Normalized Power iteration

input: $A\in\mathbb{C}^{n\times n}$ and a starting point x^0 with $\|x^0\|=1$.
for $k=1$ **to** max
  $\tilde{x}^k=Ax^{k-1}$. (左乘 A)
  $x^k=\tilde{x}^k/\|\tilde{x}^k\|$. (normalize $\tilde{x}^k$)
  $\sigma_k=(x^k)^HAx^k$. (计算 Rayleigh quotient) ($\|x^k\|=1$)
end
output: x^k (or $x^{\max}$).

① 每个 iteration 的 complexity 为 $O(n^2)$ (A 为 sparse 则可以减少)
② 计算 σ_k 时的 Ax^k 的计算可被省略 ③ Convergence rate 取决于 $\left|\frac{\lambda_2}{\lambda_1}\right|$
3. Normalized power method 的 convergence
normalized power method 的每步 iterate 满足：
① $\|'x^k-(\pm\,'v_1)\|=O(|\frac{\lambda_2}{\lambda_1}|^k)$ ② $|\sigma_k-\lambda_1|=O(|\frac{\lambda_2}{\lambda_1}|^k)$
注：① 假设 eigenvectors $'v_i$，$i=1,\cdots,n$ 为 normalized
② $\{'x^k\}_k$ 不一定收敛，x^k 的 sign/phase 可能会 "jump"
③ 若 A 为 Hermitian，则有 $|\sigma_k-\lambda_1|=O(|\frac{\lambda_2}{\lambda_1}|^{2k})$
④ x^k 的 sign/phase 取决于 $(\frac{\lambda_1}{|\lambda_1|})^k\cdot\frac{\alpha_1}{|\alpha_1|}=\text{sign}(\alpha_1)\cdot\text{sign}(\lambda_1)^k$
即同时取决于 x_0 和 v_1 的关系 与 λ_1 的符号
⑤ Normalized power method 更 stable 的原因：
· 避免 overflow 和 underflow · Rayleigh quotient 是更好的近似
4. Power iteration with shift
① 选取合适的 σ，将 A 化为 $A-\sigma I$ ($|\frac{\lambda_2-\sigma}{\lambda_1-\sigma}|<|\frac{\lambda_2}{\lambda_1}|$)
② 用 normalized power iteration 求出 $A-\sigma I$ 的 eigen-pair
③ 将 σ 加回求出的 $\lambda_1-\sigma$，得到 A 的一个 eigenvalue λ_1
注：若要提升最大，则应 $\min_\sigma|\frac{\lambda_2}{\lambda_1-\sigma}|$

§8 Inverse iteration

1. inverse iteration 的 scheme
① $A\,'\tilde{x}^k='x^{k-1}$ ($'\tilde{x}^k=A^{-1}{}'x^{k-1}$) ② $'x^k=\frac{'\tilde{x}^k}{\|'\tilde{x}^k\|}$ ③ $\sigma_k=('x^k)^HA'x^k$
注：① $'x^k$ 会收敛至对应 A 的 smallest eigenvalue 的 eigenvector
② Rayleigh quotient σ_k 会收敛至 A 的 smallest eigenvalue
2. Inverse iteration with shifts

input: $A\in\mathbb{C}^{n\times n}$, starting point x^0 with $\|x^0\|=1$, shift $\sigma\in\mathbb{C}$.
for $k=1$ **to** max
  $(A-\sigma I)\tilde{x}^k=x^{k-1}$. ($x^k=(A-\sigma I)^{-1}x^{k-1}$) // inverse iteration
  $x^k=\tilde{x}^k/\|\tilde{x}^k\|$. // normalize
  $\sigma_k=(x^k)^HAx^k$. // Rayleigh quotient
end
output: x^k (or $x^{\max}$).

① complexity：$A-\sigma I$ 的 Initial factorization：$O(n^3)$
每个 iteration 中的 complexity：$O(n^2)$
② Convergence rate：取决于 $\left|\frac{\lambda_j-\sigma}{\lambda_k-\sigma}\right|$
λ_j 为最接近 σ 的 eigenvalue，λ_k 为第二接近 σ 的 eigenvalue

§9 Rayleigh quotient iteration

1. Rayleigh quotient iteration 的 algorithm

input: $A\in\mathbb{C}^{n\times n}$, x^0 with $\|x^0\|=1$, and $\sigma_0=(x^0)^HAx^0$.
for $k=1$ **to** max
  $(A-\sigma_{k-1}I)\tilde{x}^k=x^{k-1}$. /* inverse iteration */
  $x^k=\tilde{x}^k/\|\tilde{x}^k\|$. /* normalization */
  $\sigma_k=(x^k)^HAx^k$. /* Rayleigh quotient */
end
output: x^k (or $x^{\max}$). ① 对 symmetric matrices effective

② 每次迭代均需对 $A-\sigma_{k-1}I$ 重新 factorization.
2. Rayleigh quotient iteration 的 convergence (A 为 Hermitian)
① 对于 almost every (不是所有) x^0，RQI 均会收敛至一个 eigen-pair
② 令 (v_j,λ_j) 为一个 A 的一个 eigen-pair，若 x^0 足够接近 eigenvector v_j，则
· $\|\,'x^{k+1}-(\pm\,'v_j)\|=O(\|\,'x^k-(\pm\,'v_j)\|^3)$
· $|\sigma_{k+1}-\lambda_j|=O(|\sigma_k-\lambda_j|^3)$ as $k\to\infty$

§10 (Pure) QR Iteration

1. QR Iteration 的 algorithm

input: Set $X^0=A$.
for $k=1$ **to** max
  $Q_kR_k=X^{k-1}$. /* QR factorization of X^{k-1} */
  $X^k=R_kQ_k$. /* Recombine factor in reverse order */
end

① 假定假设下，X^k 会收敛至 A 的 Schur factorization：$X^k\to T$，$A=QTQ^H$
② $\{X^k\}_k$ 彼此间 unitarily similar：$X^k=Q_k^HX^{k-1}Q_k=Q_k^H\cdots Q_1^HAQ_1\cdots Q_k$
③ 若 $X^0=A$ 为 symmetric，则 QR iteration 会保持 symmetry，$X^k\to$ diagonal
2. QR Iteration 的原理 Core procedure：重复以下步骤：
(1) Solve $(A-\sigma I)'\tilde{v}='e_n$ (3) 构造 orthogonal matrix：$Q=[\tilde{v}\ \cdots]$
(2) Normalize：$'v='\tilde{v}/\|'\tilde{v}\|$ (4) Update A：$A=Q^TAQ$
one step inverse power iteration
① 上述过程会对 $Q_1\cdots Q_k$ 的最后一列进行 Inverse iteration with shifts
② 上述过程会将 A 的最后一行和最后一列化为 diagonal form.
3. (Pure) QR algorithm 的 convergence
若 A 的某些 eigenvalues 有 same modulus，则该方法不一定收敛.

§11 QR algorithm with shift

1. QR algorithm with shift

input: Set $X^0=A$.
for $k=1$ **to** max
  Pick a shift σ_{k-1}.
  $Q_kR_k=X^{k-1}-\sigma_{k-1}I$. /* QR factorization */
  $X^k=R_kQ_k+\sigma_{k-1}I$. /* Recombine factors */
end

2. σ 的选取：Rayleigh quotient shift
$\sigma_k=\frac{('\hat{q}^k)^TA'\hat{q}^k}{\|'\hat{q}^k\|^2}$ ($'\hat{q}^k$ 为 $\hat{Q}_k=Q_1\cdots Q_k$ 的最后一列) $=('\hat{q}^k)^TA'\hat{q}^k$
$=('e_n)^T\hat{Q}_k^TA\hat{Q}_k{}'e_n=('e_n)^TX^k{}'e_n$ ($\hat{Q}_k^TA\hat{Q}_k=X^k$) $=X^k_{nn}$
收敛速度：cubic local convergence
缺点：对于某些 initial points 可能会 fail (如 $C=[0,1;1,0]$)
3. σ 的选取：Wilkinson shift
令 $X=\begin{bmatrix}x_{n-1}&y_{n-1}\\ y_{n-1}&x_n\end{bmatrix}$ Wilkinson shift σ 为 submatrix X 中接近 x_n 的 eigenvalue
收敛速度：cubic convergence (worst case 为 quadratic convergence)
优点：若 input 矩阵 A 为 real symmetric & tridiagonal，则一定收敛
4. Deflation 若 $A\in\mathbb{R}^{n\times n}$ 为 symmetric，X^k 为 QR algorithm 的 k-th iteration 的结果
$$X^k=\hat{Q}_k^TA\hat{Q}_k=\begin{bmatrix}B&'r_n\\ 'r_n^T&X^k_{nn}\end{bmatrix}\approx\begin{bmatrix}B&'0\\ '0^T&\lambda_n\end{bmatrix}$$
① 基本思路：· 若 $\|'r_n\|<tol$，则停止该 iteration，$X^k_{nn}\approx\lambda_n$
· 接下来对 smaller matrix B 进行 QR algorithm
$$Y^k=\hat{U}_k^TB\hat{U}_k=\begin{bmatrix}C&'r_{n-1}\\ 'r_{n-1}^T&Y^k_{n-1,n-1}\end{bmatrix}\approx\begin{bmatrix}C&'0\\ '0^T&\lambda_{n-1}\end{bmatrix}\ \text{相当于}$$
$$\begin{bmatrix}\hat{U}_k&'0\\ '0^T&1\end{bmatrix}\hat{Q}_k^TA\hat{Q}_k\begin{bmatrix}\hat{U}_k&'0\\ '0^T&1\end{bmatrix}\approx\begin{bmatrix}C&'0&'0\\ '0^T&\lambda_{n-1}&0\\ '0^T&0&\lambda_n\end{bmatrix}$$
· A 的 original eigenvectors 可以通过 将 $\hat{Q}_k$，$\hat{U}_k$，$\cdots$ explicitly 乘起来得到
注：① 该思路同样适用于 nonsymmetric case，但结果矩阵为上三角矩阵
② Flops：# flops：1st deflation = # iter. $\underset{\approx O(1)}{\cdot}$ # flops for one QR step $=O(n^3)$
若 cost：total costs $=O(n^3+(n-1)^3+\cdots+1)=O(n^4)$

§12 The reduction phase (QR iteration 的两个 phases)

1. Hessenberg matrix 在 triangular 上额外多出了一个与主对角线相邻的 diagonal
① 任意矩阵均可以通过 orthogonal transformation 在有限步内化为 Hessenberg
② Symmetric Hessenberg matrix 为 tridiagonal
2. Reduction phase 的步骤
Step 1：将 a_{n1} 以下的 entries 化为 0：
· 若 $a_1=[a_{11}\ \ a^T]^T$ 为 A 的第一列，令
$$R_1^T=\begin{bmatrix}1&'0^T\\ '0&H_{\mathcal{V}}\end{bmatrix}$$，其中 $H_{\mathcal{V}}=I-\frac{2\mathcal{V}\mathcal{V}^T}{\mathcal{V}^T\mathcal{V}}$，$'\mathcal{V}='a-\alpha_1{}'e_1$，$\alpha_1=\pm\|'a\|_2$
$$R_1^T{}'a_1=\begin{bmatrix}a_{11}\\ H_{\mathcal{V}}{}'a\end{bmatrix}=\begin{bmatrix}a_{11}\\ \alpha_1\\ '0\end{bmatrix}$$
· 为了不改变 eigenvalues，再对 A 右乘 R_1 ($R_1^TAR_1$ 为相似变换)
注：full reduction algorithm 的 cost：(最后化为对称阵)
· 一般情况：$\approx\frac{10}{3}n^3$ flops · Symmetric case：$\approx\frac{4}{3}n^3$ flops
3. Reduction phase 能减少 computational cost 的原理
① 若 $H=QR$ 为 upper Hessenberg，则 R 和 RQ 也为 upper Hessenberg
$\Rightarrow$ 可以使用更 cheaper 的方式进行 QR factorization (如 Givens rotations)
② Cost：· General matrices：$O(n^3)\to O(n^2)$ · Symmetric matrices：$\to O(n)$
4. QR iteration：two phase 各 cost
real symmetric：$\frac{4}{3}n^3/\frac{8}{3}n^3$；general matrices：$10n^3/25n^3$

8. Singular value decomposition

§1 Singular value decomposition

1. SVD：一个 real $m\times n$ 矩阵 A：$A=U\Sigma V^T$
① $U\in\mathbb{R}^{m\times m}$ 的 columns $'u_i$ 被称为 left singular vectors
② $V\in\mathbb{R}^{n\times n}$ 的 columns $'v_i$ 被称为 right singular vectors
③ $\Sigma\in\mathbb{R}^{m\times n}$ 为一 diagonal matrix with $\sigma_i=\sigma_i{}'i\geq 0$ 为 A 的 singular values
且通常为 ordered：$\sigma_1\geq\sigma_2\geq\cdots\geq\sigma_p\geq 0$，$p=\min\{m,n\}$
注：① 若 $U=V$，则 SVD 等价于 eigendecomposition (仅仅 $m=n$ 还不能得出该结论)

2. SVD 的 existence 与 construction

所有 real matrix $A \in R^{m\times n}$ 均有 SVD：$A = U\Sigma V^T$，且

① U 为 $AA^T \in R^{m\times m}$ 的 eigenvectors ② V 为 $A^TA \in R^{n\times n}$ 的 eigenvectors

③ singular values $\sigma_i = \sqrt{\lambda_i}$，$i=1,\cdots,p$，其中 $\lambda_1,\cdots,\lambda_p$ 为 AA^T 与 A^TA 中维度较小的的 eigenvalues

3. thin SVD (Economy size)

r 为 matrix A 的非零 singular values 数：$\sigma_1 \geq \cdots \geq \sigma_r > 0 = \sigma_{r+1} = \cdots = \sigma_p$

则 $A = [\underbrace{U_1}_{r} \ \underbrace{U_2}_{m-r}] \begin{bmatrix} \tilde{\Sigma} & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} V_1^T \\ V_2^T \end{bmatrix}$ ，则 thin SVD 为：$A = U_1 \tilde{\Sigma} V_1^T$

4. 对 Left & Right singular vectors 的理解

$$A = U\Sigma V^T \Longleftrightarrow \begin{cases} AV = U\Sigma \\ U^TA = \Sigma V^T \end{cases} \Longleftrightarrow \begin{cases} Av_i = \sigma_i u_i \\ u_i^TA = \sigma_i v_i \end{cases}$$

(v_i 为右奇异向量，类似右乘特征向量)

(u_i 为左奇异向量，类似左乘特征向量)

5. 计算 SVD

$A = \begin{bmatrix} 1 & 0 & -1 \\ 1 & 1 & 0 \end{bmatrix}$

① Step 1：计算 A^TA 与 AA^T 中维度更小的一个 $AA^T = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$

② Step 2：计算 A^TA 或 AA^T 的特征值，求出 Σ

$\det(AA^T - \lambda I) = 0 \Rightarrow \lambda_1 = 3,\ \lambda_2 = 1 \Rightarrow \sigma_1 = \sqrt{3},\ \sigma_2 = 1 \Rightarrow \Sigma = \begin{bmatrix} \sqrt{3} & 0 \\ 0 & 1 \end{bmatrix}$

③ Step 3：计算 A^TA 或 AA^T 的特征向量，求出 V 或 U

$AA^T - \lambda_1 I = 0 \Longleftrightarrow \left[\begin{array}{cc|c} 1 & -1 & 0 \\ 0 & 0 & 0 \end{array}\right]$ 取自由未知量 $x_2 = 1$，有 eigenvector $[1,1]^T$，Normalize 后有：$u_1 = \frac{1}{\sqrt{2}}[1,1]^T$

$AA^T - \lambda_2 I = 0 \Longleftrightarrow \left[\begin{array}{cc|c} 1 & 1 & 0 \\ 0 & 0 & 0 \end{array}\right]$ 取自由未知量 $x_2 = 1$，有 eigenvector $[-1,1]^T$，Normalize 后有：$u_2 = \frac{1}{\sqrt{2}}[-1,1]^T$

$\Rightarrow U = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix}$

④ Step 4：利用 $A = U\Sigma V^T$ 求解另一个 orthorgonal matrix

$A = U\Sigma V^T \Longleftrightarrow A^T = V\Sigma U^T \Longleftrightarrow V = A^TU\Sigma^{-1}$

$\Rightarrow V = \begin{bmatrix} 1 & 1 \\ 0 & 1 \\ -1 & 0 \end{bmatrix} \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} \frac{\sqrt{3}}{3} & 0 \\ 0 & 1 \end{bmatrix} = \frac{1}{\sqrt{6}} \begin{bmatrix} 2 & 0 \\ 1 & \sqrt{3} \\ -1 & \sqrt{3} \end{bmatrix}$

§2 SVD 的应用：Low-rank matrix approximation

1. 利用 SVD 改写 A：$A = \sum_{i=1}^{r} \sigma_i u_i v_i^T$

2. Best Low-rank approximation 若 A 有 rank r，且给定 $0 \leq k \leq r$，则

$\|A - A_{(k)}\|_F^2 = \min_{rank(X) \leq k} \|X - A\|_F^2 = \sigma_{k+1}^2 + \cdots + \sigma_r^2$

即 $A_{(k)} = \sum_{i=1}^{k} \sigma_i S_i = \sum_{i=1}^{k} \sigma_i u_i v_i^T$ 为限定 $rank(X) \leq k$ 时的 best approximation

§3 计算 SVD 的 algorithm

1. 第一种想法：直接计算 $A = U\Sigma V^T$：会导致 loss of precision

2. 第二种想法：将 SVD 化为 eigenvalue problem 给定 $A \in R^{m\times n}$ (假设 $m > n$)，

则 $B = \begin{bmatrix} 0 & A^T \\ A & 0 \end{bmatrix}$ 为 symmetric，由 $A = U_1\Sigma V^T$，$U = [U_1 \ U_2]$ 有

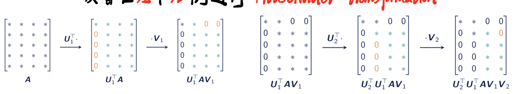
$P^TBP = \begin{bmatrix} \Sigma & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & -\Sigma \end{bmatrix}$ 其中 $P = \frac{1}{\sqrt{2}} \begin{bmatrix} V & 0 & V \\ U_1 & \sqrt{2}U_2 & -U_1 \end{bmatrix}$

因此对 B 进行 Power iteration 或 QR algorithm

3. 第三种想法：Two-phase approach

Phase I：Golub-Kahan bidiagonalization

交替在左和右侧进行 Householder transformation



- Let $b^\top = [b_1 \ b_2 \ b_3 \ b_4] = [b_1 \ \beta^\top]$ be the first row of $U_1^\top A$.
- We can build V_1 as follows:

$$v_1 = \begin{bmatrix} b_2 \\ b_3 \\ b_4 \end{bmatrix} \pm \|\beta\| \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \quad H_{v_1} = I - \frac{2 v_1 v_1^\top}{\|v_1\|^2}, \quad V_1 = \begin{bmatrix} 1 & 0 \\ 0 & H_{v_1} \end{bmatrix}.$$

- The total costs for Phase I are essentially identical to computing two QR factorizations of the matrix A: $\approx 4mn^2 - \frac{4}{3}n^3$ flops.
- There are also faster variants that requires fewer flops.

Phase II：使用 specialized QR algorithm 得到 diagonal form 与 singular values

§4 Minimum norm solutions 与 pseudoinverse

假设 $\sigma_1 \geq \cdots \geq \sigma_r > 0 = \sigma_{r+1} = \cdots = \sigma_p$，有

$A = [U_1 \ U_2] \begin{bmatrix} \tilde{\Sigma} & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} V_1^T \\ V_2^T \end{bmatrix} = U_1 \tilde{\Sigma} V_1^T = \sum_{i=1}^{r} \sigma_i u_i v_i^T$

1. Orthorgonal basis

① U_1 的 columns 为 $range(A)$ 的 orthonormal basis (即 $range(U_1) = range(A)$)

② U_2 的 columns 为 $range(A)^\perp$ 的 orthonormal basis (由 $U_1^TU_2 = 0$ 易得)

③ V_2 的 columns 为 $null(A)$ 的 orthonormal basis，即 $range(V_2) = null(A)$

证 ① (仅证明 $x \in Range(A) \Rightarrow x \in Range(U_1)$)

$x \in Range(A) \Rightarrow \exists y$ s.t. $x = Ay \Rightarrow x = \sum_{i=1}^{r} \sigma_i u_i v_i^T y$

$= \sum_{i=1}^{r} \sigma_i (v_i^T y) u_i \Rightarrow x \in Range(U_1)$

证 ③ (仅证明 $x \in Range(V_2) \Rightarrow x \in null(A)$)

$x \in Range(V_2) \Rightarrow \exists y$ s.t. $x = V_2 y \Rightarrow Ax = U_1 \tilde{\Sigma} V_1^T x =$

$U_1 \tilde{\Sigma} V_1^T V_2 y = 0$ ($V_1^TV_2 = 0$) $\Rightarrow x \in null(A)$

2. 利用 SVD 解 linear system ($A \in R^{m\times n}$，$b \in R^m$，求 $Ax = b$ 的解)

① 由 SVD：$A = U_1 \tilde{\Sigma} V_1^T$，有 $Ax = b \Longleftrightarrow U_1 \tilde{\Sigma} V_1^T x = b$

② 由 $U_1^TU_1 = I$，有 $U_1 \tilde{\Sigma} V_1^T x = b \Longleftrightarrow V_1^T x = \tilde{\Sigma}^{-1} U_1^T b$

由 $U_2^TU_1 = 0$，有 $U_1 \tilde{\Sigma} V_1^T x = b \Rightarrow U_2^T b = 0$

($\Longleftrightarrow b \in range(U_2^T)$ $\Longleftrightarrow b \in range(U_2)^\perp$ $\Longleftrightarrow b \in range(A)$)

③ 由 $V_1V_1^T = I - V_2V_2^T$，所有 $Ax = b$ 的解 x^* 均满足

$V_1 V_1^T x = V_1 \tilde{\Sigma}^{-1} U_1^T b \Longleftrightarrow x = V_1 \tilde{\Sigma}^{-1} U_1^T b + \eta$，where $\eta \in Null(A)$

注：若 A 有 full rank $r = n$，则不存在 V_2，此时 $Ax = b$ 有唯一解 $x^* = V_1 \tilde{\Sigma}^{-1} U_1^T b$ (若 $b \in range(A)$)

3. Minimum norm solution

对于 minimum norm problem：$\min_{x \in R^n} \|x\|_2^2$ s.t. $Ax = b$

令 $x^+ := V_1 \tilde{\Sigma}^{-1} U_1^T b$，则 $Ax = b$ 的解为 $x = x^+ + \eta$，$\eta \in Null(A)$ ($\eta \in range(V_2)$，$x^+ \in range(V_1)$)

$\Rightarrow \|x\|_2^2 = \|x^+\|_2^2 + 2\eta^T x^+ + \|\eta\|_2^2 = \|x^+\|_2^2 + \|\eta\|_2^2$

$\Rightarrow x^+$ 为 the unique min. norm solution

4. Pseudoinverse：$A^+ := V_1 \tilde{\Sigma}^{-1} U_1^T$

① 若 A 为 square 且 nonsingular，则 $A^+ = A^{-1}$

② 若 A 有 full column rank $r = n$，则 $A^+ = (A^TA)^{-1}A^T$

③ $Ax = b$ 的 minimum norm solution 为 $x^+ = A^+b$ $\quad x^+ = A^+b$

④ Least square problem $\min_x \|Ax - b\|^2$ 的 minimum norm solution 为 ($\Longleftrightarrow \min_x \|x\|^2$ s.t. $A^TAx = A^Tb$)

9. Iterative solver

§1 The conjugate direction method 解 $Ax = b$，A symmetric 且 $A \succ 0$

1. Background：问题可被转化为 $\min_{x\in R^n} \phi(x) = \frac{1}{2}x^TAx - b^Tx$

关注 residuals $r^k := Ax^k - b = \nabla\phi(x^k)$ 的 convergence

2. Conjugate vectors：$\{p^0, p^1, \cdots, p^l\}$ 被称为 conjugate w.r.t. $A \succ 0$，若 $(p^i)^TAp^j = 0$，$\forall i \neq j$ ① orthogonal 可视作 conjugate 的特例 ($A = I$)

② $p^0, p^1, \cdots, p^l$ 线性无关 ③ A 的 eigenvectors 为一组 conjugate vectors

3. Conjugate direction method 给定 $x^0 \in R^n$ 与 conjugate directions $\{p^0, p^1, \cdots, p^{n-1}\}$

$x^{k+1} = x^k + \alpha_k p^k$，$\alpha_k = -\frac{(r^k)^Tp^k}{(p^k)^TAp^k}$ (遵循了 exact line search $\min_\alpha \phi(x^k + \alpha_k p^k) := h(\alpha)$)

4. Conjugate directions 与 convergence

$\forall x^0 \in R^n$，至多需要 n steps 找到 $Ax^* = b$ 的解 x^*

§2 The conjugate gradient method

1. The conjugate gradient method：p^k 被选为 $-r^k$ 和 p^{k-1} 的线性组合：

$p^k = -r^k + \beta_k p^{k-1}$，$\beta_k = \frac{(r^k)^TAp^{k-1}}{(p^{k-1})^TAp^{k-1}}$ → s.t. $(p^{k-1})^TAp^k = 0$，$r^k := Ax^k - b$

2. Algorithm：The conjugate gradient method

1. Initialization: Select an initial point $x^0 \in \mathbb{R}^n$ and set $\mathtt{tol} \geq 0$, $r^0 = Ax^0 - b$, and $p^0 = -r^0$.

For $k = 0, 1, \ldots$:

$r^{k+1} = Ax^{k+1} - b = Ax^k + \alpha_k Ap^k - b = r^k + \alpha_k Ap^k$

2. Compute the updates $\alpha_k = -\frac{(r^k)^\top p^k}{(p^k)^\top Ap^k}$

$x^{k+1} = x^k + \alpha_k p^k$ and $r^{k+1} = r^k + \alpha_k Ap^k$.*

3. If $\|r^{k+1}\| \leq \mathtt{tol}$, then STOP and x^{k+1} is the output.

4. Calculate $\beta_{k+1} = \frac{(r^{k+1})^\top Ap^k}{(p^k)^\top Ap^k}$ and $p^{k+1} = -r^{k+1} + \beta_{k+1}p^k$.

$i < k$

3. conjugate directions：$(p^k)^TAp^i = 0$，$(r^k)^Tr^i = 0$，$(r^k)^Tp^i = 0$

4. convergence rate：$\|x^k - x^*\|_A \leq 2\left[\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right]^k \|x^0 - x^*\|_A$ (linear)

($\|x\|_A^2 = x^TAx$ (A-norm))

5. It is possible to simplify the update rules in the CG-method to:

$-(r^k)^Tp^k = -(r^k)^T(-r^k + \beta_k p^{k-1}) = \|r^k\|^2$

☆ $\alpha_k = \frac{\|r^k\|^2}{(p^k)^\top Ap^k}$ and $\beta_{k+1} = \frac{\|r^{k+1}\|^2}{\|r^k\|^2}$.

10. Nonlinear equations

§1 Nonlinear equations ($F: R^n \to R^n$，求 $F(x) = 0$ 的 roots $x \in R^n$)

1. Existence & uniqueness：$f: R \to R$：利用 intermediate value theo

F: high-dim：任意数量的 solutions

2. Local uniqueness：① $F: R^n \to R^n$ continuously differentiable

② $x^* \in R^n$ 有 $F(x^*) = 0$ ③ Jacobian $DF(x^*)$ nonsingular，则 x^* 为 local unique：$\exists$ nbhd $U \subset R^n$ of x^* s.t. $F(x) \neq 0$ for $\forall x \in U \setminus \{x^*\}$.

$$DF(x) = \begin{bmatrix} \nabla F_1(x)^T \\ \vdots \\ \nabla F_n(x)^T \end{bmatrix} = \begin{bmatrix} \frac{\partial}{\partial x_1}F_1(x) & \cdots & \frac{\partial}{\partial x_n}F_1(x) \\ \vdots & \ddots & \vdots \\ \frac{\partial}{\partial x_1}F_n(x) & \cdots & \frac{\partial}{\partial x_n}F_n(x) \end{bmatrix}$$

§2 Conditioning $f(x^*) = 0 \Longleftrightarrow x^* = f^{-1}(0)$

1. relative condition number：$cond(f) = \left|\frac{xf'(x)}{f(x)}\right| \Rightarrow cond(f^{-1}) = \left|\frac{f(x)}{xf'(x)}\right|$

2. absolute condition number：$cond_{abs}(f) = \frac{|f(x+\Delta x) - f(x)|}{|\Delta x|} \approx |f'(x)|$

$cond_{abs}(f^{-1}) \approx \frac{1}{|f'(x)|}$ 若 $f'(x) \approx 0$，则为 ill-conditioned

3. High-dimensional case：① $cond(F) = \frac{\|x\| \cdot \|DF(x)\|}{\|F(x)\|}$

② $cond_{abs}(F) = \|DF(x^*)^{-1}\|$ for root finding problem

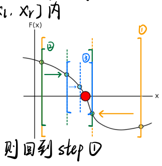
4. Approximate solutions：$\|F(\hat{x})\| \approx 0$ (不是 $\|x - x^*\| \approx 0$)

§3 Bisection method

1. Bisection method

local minimizer 必须满足 FONC：$f'(x) = 0 \Rightarrow g(x) = f'(x)$ 的求根问题

若有两点 x_l 与 x_r 满足 $g(x_l) < 0$，$g(x_r) > 0$，则根据介值定理，若 $g(\cdot)$ 连续，则一定存在一个 $g(\cdot)$ 的根落在 $[x_l, x_r]$ 内



① 定义 $x_m = \frac{x_l + x_r}{2}$

② 若 $g(x_m) = 0$，则 output x_m

③ 否则 · 若 $g(x_m) > 0$，则令 $x_r = x_m$

· 若 $g(x_m) < 0$，则令 $x_l = x_m$

④ 若 $|x_r - x_l| < \varepsilon$，则终止并输出 $\frac{x_l + x_r}{2}$，否则回到 step ①

2. 对 bisection method 的分析

① bisection method 每次将区间长度减小一半 (linear)

② 若要求出 x^* 的一个 ε approximation，则至多进行 $\log_2 \frac{x_r - x_l}{\varepsilon}$ 次迭代

③ 对 $f'(x)$ 使用 bisection method，可求出满足 FONC (approximately) 的点。若 $f(x)$ convex，则可求出 $f(x)$ 的 global minimizer (approximately)

§4 Fixed-point iteration (1D)

1. Fixed-point problem：求 $x = g(x)$ 的 x，用 fixed-point iteration 求：

$x^{k+1} = g(x^k)$ (可以将 $f(x) = 0$ 转化为 $x = g(x)$)

2. Lipschitz continuity

A function $g: \mathbb{R} \to \mathbb{R}$ is Lipschitz continuous with constant $L \geq 0$ if

$|g(x) - g(y)| \leq L \cdot |x - y| \quad \forall x, y \in \mathbb{R}.$ $\Longleftrightarrow |g'(x)| \leq L$ $\forall x \in R$

If $L < 1$, then g is called a contractive mapping or contractive.

3. Fixed-point method 的 convergence

Let $x^* = g(x^*)$ be a fixed-point of g and suppose that there is a neighborhood (interval) U containing x^* such that g is contractive on U with constant $L < 1$. $\Longleftrightarrow |g'(x)| \leq L$，$\forall x \in U$

则 $x^{k+1} = g(x^k)$ 以 rate L linear converges to x^* ($\forall x^0 \in U$)

① $|g'(x^*)| < 1 \Rightarrow |g'(x)| < 1$，$\forall x \in N_\delta(x^*)$

② 若 $g'(x^*) = 0$ 且 g 为 2 阶连续可微 $\Rightarrow$ convergence rate 至少 quadratic

$|x^{k+1} - x^*| = |g(x^k) - g(x^*)| = |g'(x^*)(x^k - x^*) + O(|x^k - x^*|^2)| = O(|x^k - x^*|^2)$

③ 若 $|g'(x^*)| > 1$，则不能收敛到 x^* / g 有 unique fixed-point in U

4. Existence：若 g 在 U 上 contractive，且 $g: U \to U$ ($g(U) \subseteq U$)，则

§5. Newton's method (1D)

1. Newton's method 的特征：① 收敛速度更快 ② 需要二阶导的信息 ③ sensitive to initial point

2. Newton's method 的基本思路：用 x^k 处的 first-order Taylor expansion 来近似任意点 $g(x)$

$g(x) \approx g(x^k) + g'(x^k)(x - x^k)$ 令右侧等于 0，即可解出 x^{k+1}：$x^{k+1} = x^k - \frac{g(x^k)}{g'(x^k)}$

注：① 对于某些 initial point，Newton's method 不一定收敛 ② convex 不能确保收敛

3. Newton's method 的收敛模式

① 若 $g(x)$ 为 twice continuously differentiable，且 x^* 为 $g(x)$ 的根，$g'(x^*) \neq 0$。则当 $|x^0 - x^*|$ 足够小 (确保能收敛) 时，Newton iterations 生成的 sequence 满足 $|x^{k+1} - x^*| \leq C|x^k - x^*|^2$ 其中 $C = \sup_x \frac{1}{2}\left|\frac{g''(x)}{g'(x)}\right|$ (取决于 curvature)，

② 我们称这种 convergence speed 为 quadratic convergence

4. 与 fixed-point iter 的联系：$f(x) = 0 \Longleftrightarrow x = g(x) := x - \frac{f(x)}{f'(x)}$

因此有 $g'(x) = \frac{f(x)f''(x)}{(f'(x))^2}$，若 $f(x^*) = 0$ 且 $f'(x^*) \neq 0$，则 $g'(x^*) = 0$ (二次收敛)

§6 Nonlinear system 的 algorithm：fixed-point iteration

1. Fixed-point problem：求 $x = G(x)$ 的 x，用 fixed-point iteration 求：

$x^{k+1} = G(x^k)$ (可以将 $F(x) = 0$ 转化为 $x = G(x)$)

2. Lipschitz continuity：

- Lipschitz continuity can be generalized to n-dimensional mappings.
- G is Lipschitz continuous (on $\mathbb{R}^n$) with constant $L \geq 0$ if

判断时可利用各 norm equivalent $\|G(x) - G(y)\|_2 \leq L \cdot \|x - y\|_2 \quad \forall x, y \in \mathbb{R}^n.$

- If $L < 1$, then G is said to be contractive.
- The mapping G is Lipschitz continuous on a convex set $U \subset \mathbb{R}^n$ if and only if $\|DG(x)\|_2 \leq L$ for all $x \in U$.

3. Fixed-point method 的 convergence

- Conditions for convergence are similar to the 1D case.
- If G is contractive on U, then the fixed-point scheme $x^{k+1} = G(x^k)$ converges as long as $x^0 \in U$.
- The convergence is typically linear with rate $\eta \approx \|DG(x^*)\|$.
- If $DG(x^*) = 0$, then the convergence is quadratic.

§7 Nonlinear system 的 algorithm：Newton's method

1. 形式：$x^{k+1} = x^k - DF(x^k)^{-1}F(x^k)$

2. 实际实现：$DF(x^k)d^k = -F(x^k) \Rightarrow x^{k+1} = x^k + d^k$

3. convergence：quadratic (但起点要 sufficiently close to x^*)

4. Cost：算 DF：n^2，解 linear system：$O(n^3)$

§8 Globalized Newton's method

1. step size 的选取：① merit function：$\min_x m(x) = \frac{1}{2}\|F(x)\|^2$

Observations:

- x^* solves $F(x^*) = 0 \Longleftrightarrow m(x^*) = 0$, i.e., x^* is a global solution of problem (1).
- It holds that $\nabla m(x) = DF(x)^\top F(x)$.

⇝ If $DF(x)$ is invertible (for all x), then stationary points of m need to satisfy $F(x) = 0$.

- Let d be the Newton direction with $DF(x)d = -F(x)$, then:

$$\nabla m(x)^\top d = F(x)^\top DF(x)d = -\|F(x)\|^2 < 0.$$

⇝ The Newton direction is always a descent direction for m!

The Globalized Newton Method

1. Initialization: Choose an initial point $x^0 \in \mathbb{R}^n$.

For $k = 0, 1, 2, \ldots$:

2. STOP, if $\|\nabla m(x^k)\| \leq \mathtt{tol}$.
3. Compute d^k by solving the Newton equation $DF(x^k)d^k = -F(x^k)$.
4. Perform backtracking on the merit function m to calculate a step size α_k. Set $x^{k+1} = x^k + \alpha_k d^k$.

Comments:

- This is a full analog of the globalized Newton method for optimization problems: $m \rightsquigarrow f$.
- Locally, under certain assumptions, the algorithm turns into a pure Newton method with $\alpha_k = 1$ for all k sufficiently large.

⇝ Fast local (quadratic) convergence of $\{x^k\}_k$ can be observed.

2. α 的选取 (backtracking line search) (回溯线搜索)

假设 d^k 已知，希望求解 α_k，则遵循以下步骤：

① 选取一个较小的常数 $\alpha \in (0, 0.5)$，选取另一个常数 $0 < \beta < 1$ ② 令 $t = 1$

③ 若 $f(x^k + td^k) \leq f(x^k) + \alpha t \nabla f(x^k)^T d^k$，则选取 $\alpha_k = t$ (Armijo condition)

否则令 $t = \beta t$ 并重复这一步骤

3. KKT conditions 若 x 为 local minimizer，则一定存在 λ, ν 与 η 使以下 conditions 满足：

Lagrangian 为：$L(x, \lambda, \nu, \eta) = f(x) + \sum_{i=1}^{m} \lambda_i g_i(x) + \sum_{i=1}^{p} \nu_i h_i(x) + \sum_{i=1}^{r} \eta_i l_i(x)$

① Main condition

$D_jf(x) + \sum_{i=1}^{m} \lambda_i D_jg_i(x) + \sum_{i=1}^{p} \nu_i D_jh_i(x) + \sum_{i=1}^{r} \eta_i D_jl_i(x) \begin{cases} \geq 0 & 若\ x_j \geq 0 \\ \leq 0 & 若\ x_j \leq 0 \\ = 0 & 若\ x_j\ free \end{cases}$

② Dual feasibility

$\lambda_i \leq 0 \quad i = 1, \cdots, m$ (对应 $g_i(x) \geq 0$)

ν_i free $\quad i = 1, \cdots, p$ (对应 $h_i(x) = 0$)

$\eta_i \geq 0 \quad i = 1, \cdots, r$ (对应 $l_i(x) \leq 0$)

③ Complementarity

$\lambda_i \cdot g_i(x) = 0 \quad \forall i$

$\nu_i \cdot h_i(x) = 0 \quad \forall i$

$\eta_i \cdot l_i(x) = 0 \quad \forall i$

} 对应 Lagrangian 中添加的各项为 0

$x_i \cdot D_{x_i}L(x, \lambda, \nu, \eta) = 0 \quad \forall i$ (若 x_i free 则忽略)

④ Primal feasibility

$g_i(x) \geq 0$，$h_i(x) = 0$，$l_i(x) \leq 0$，$\forall i$

$x_i \geq 0$，$\forall i \in M$；$x_i \leq 0$，$\forall i \in N$；x_i free，$\forall i \notin M \cup N$