第一章 课程引论

殷东生
yindongsheng@tsinghua.edu.cn

清华大学数学科学系

2023年秋季学期

课程基本介绍

- 课程名称：数值分析
- 参考书： 数值分析基础（第三版） 关治、陆金甫著，高等教育出版社。
- 任课教师：殷东生，理科楼A408，Tel：62772871，yindongsheng@tsinghua.edu.cn
- 助教：唐逸航，tyh23@mails.tsinghua.edu.cn；
  邵惟至，2362595573@qq.com。
- 考核方式：作业：15%，数值实验：15%，期中考试：30%，期末考试： 30%，课堂测试%10。
- 课程所需准备知识：微积分，高等代数和计算机编程。

计算科学

20世纪40年代，电子计算机的发明为计算成为第三种科学研究手段提供了可能。半个多世纪以来，随着计算机 技术的飞速发展，计算作为科学研究方法的地位不断上升。现在试验、理论分析和计算“三足鼎立”，已成为 科学活动的主要方式。

计算科学是通过计算的手段来解决实际问题的一门科学，其处理问题的过程主要有如下三个环节：

- 建立数学模型；
- 设计计算方法—编制程序—用计算机计算—得到数值结果；
- 将计算结果与理论分析和实际得到的数据结合给出实际问题的答案，或提出对模型的修正。

上述第二个环节的核心是算法的分析和设计，也是这门课的主要内容。

各系开设的计算类课程

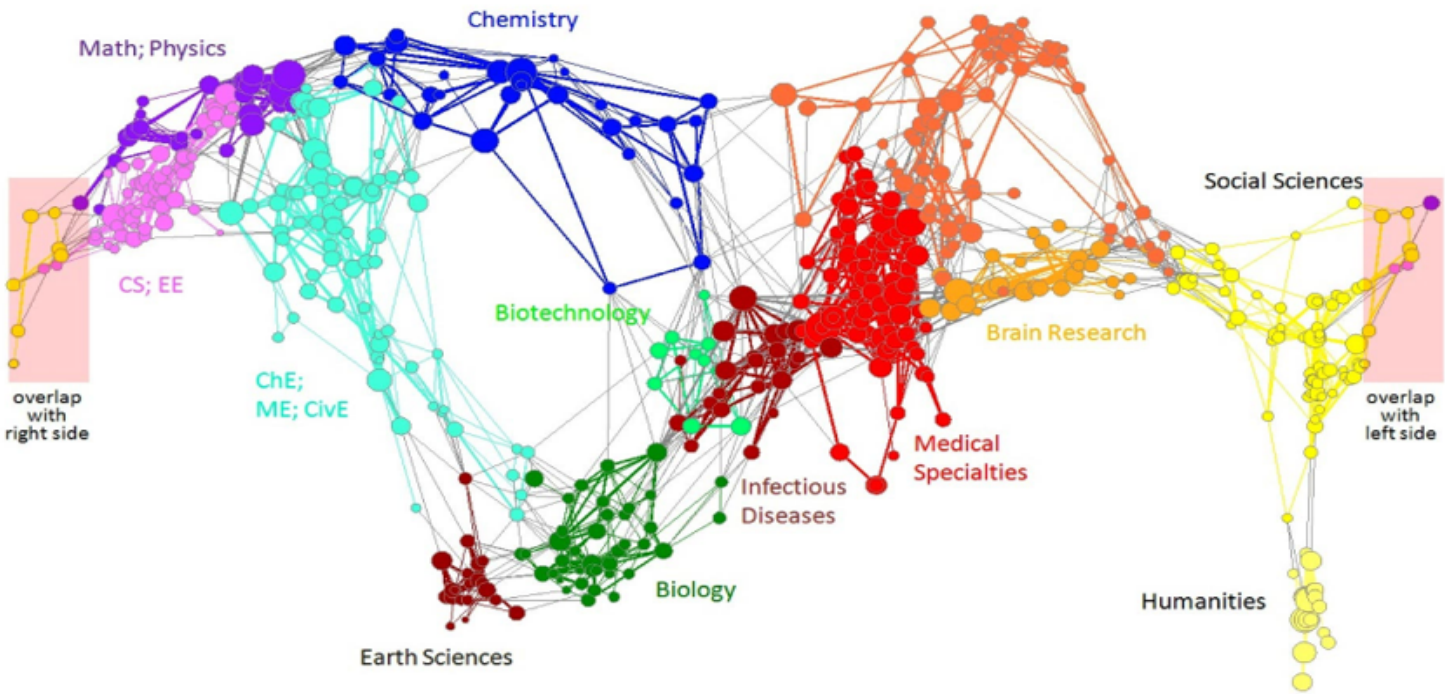
- 工物系： 电磁场数值计算；反应堆物理与数值计算；计算机模拟物理；反应堆热工流体数值计算；流动与传热传质过程的数值模拟。
- 水利系：计算流体力学；弹性力学与有限元；计算河流及河口海岸动力学；渗流力学与计算分析；非线性计算力学；浅水流动的特性与数值模拟。
- 自动化系：机器学习；人工智能基础；大数据学科前沿热点；计算分子生物学引论。
- 航天航空学院：计算流体力学；计算固体力学；有限元法基础；热物理数值计算；计算流体动力学； 航空宇航推进的数值模拟方法。
- 能动系：数值传热学（英）；弹性力学与有限元，计算与分析流体力学；高等流体力学数值方法；燃烧过程的数值方法。
- 电机系：高电压工程与数值计算；电磁场数值计算；电力系统计算方法。

各系开设的计算类课程

- 土木系：弹性力学及有限元基础；计算结构力学概论；土力学理论及数值方法；计算流体动力学。
- 物理系：计算物理；量子物理计算方法选讲；计算凝聚态物理选讲；第一原理计算方法。
- 交叉信息院：计算生物学；计算理论；机器学习；大数据基础；人工智能；高等计算经济学；计算机应用数学；计算能源经济学。
- 生医系：系统与计算神经科学；脑科学与人工智能。
- 材料学院：计算材料学；材料加工计算机模拟与仿真。
- 车辆学院：计算流体力学。电子系：机器学习；高等机器学习。
- 机械系：机械工程数值计算；计算机分子模拟；有限元分析；制造过程数值模拟技术。
- 化学系：计算化学导论；计算化学实验；理论与计算化学。

计算社会科学(Computational Social Sciences)

引用


Math; Physics
Chemistry
Social Sciences
CS; EE
Biotechnology
Brain Research
overlap
with
right side
ChE;
ME; CivE
Medical
Specialties
overlap
with
left side
Infectious
Diseases
Biology
Humanities
Earth Sciences

数值分析的基本内容

本课程主要讲述以下内容

1. 线性方程组的直接法;
2. 线性方程组的经典迭代法;
3. 非线性方程组的迭代法;
4. 矩阵特征值问题;
5. 函数的插值;
6. 函数的最佳平方逼近、有理逼近与最小二乘法;
7. 数值积分与微分;
8. 常微分方程数值解。

研究数值算法的必要性

线性代数方程组的求解

$$A\boldsymbol{x} = \boldsymbol{b},$$

其中$A = \left(a_{ij}\right)_{n\times n}$ 是 n 阶方阵，

$$\boldsymbol{b} = (b_1, b_2, \cdots, b_n)^T, \boldsymbol{x} = (x_1, x_2, \cdots, x_n)^T \in \mathbb{R}^n.$$

已知矩阵 A，向量 $\boldsymbol{b}$，求解未知量 $\boldsymbol{x}$。若行列式

$$d = \det A = |A| \neq 0$$

则方程有唯一解：

$$\boldsymbol{x} = A^{-1}\boldsymbol{b}, \quad A^{-1} \quad \text{为}A\text{的逆矩阵}$$

定义 (伴随矩阵)

设n阶方阵$A = (a_{ij})_{n\times n}$，A_{ij}是A中元素a_{ij}的代数余子式。则矩阵

$$A^* = \begin{pmatrix} A_{11} & A_{21} & \cdots & A_{n1} \\ A_{12} & A_{22} & \cdots & A_{n2} \\ \vdots & \vdots & \ddots & \vdots \\ A_{1n} & A_{2n} & \cdots & A_{nn} \end{pmatrix}$$

称为矩阵A的伴随矩阵。

定理

矩阵A可逆的充要条件为$\det A \neq 0$，且

$$A^{-1} = \frac{1}{\det A} A^*.$$

而行列式可以如下计算

$$d = \sum_{j_1 j_2 \cdots j_n} \tau(j_1 j_2 \cdots j_n) a_{1j_1} a_{2j_2} \cdots a_{nj_n},$$

这里 $j_1 \cdots j_n$ 是$1, 2, \cdots n$的一个排列，而

$$\tau(j_1 j_2 \cdots j_n) = \begin{cases} 1, & j_1 \cdots j_n \text{ 偶排列} \\ -1, & j_1 \cdots j_n \text{ 奇排列} \end{cases}$$

d 中共有 $n!$项求和，每次要做$n-1$次乘法运算，因而计算 d需要 $n!(n-1)$次乘法和 $n!-1$ 次加法，总计算量

$$\begin{array}{ll} \times\div & \mathcal{O}\big((n+1)!\big) \\ +- & \mathcal{O}\big((n+1)!\big) \end{array}$$

Cramer法则

定理

令$A = (\boldsymbol{a}_1|\cdots|\boldsymbol{a}_n)$是列向量为$\boldsymbol{a}_i \in \mathbb{R}^n$的非奇异矩阵，则$A\boldsymbol{x} = \boldsymbol{b}$的解为

$$x_i = \frac{\det(\boldsymbol{a}_1|\cdots|\boldsymbol{a}_{i-1}|\boldsymbol{b}|\boldsymbol{a}_{i+1}|\cdots|\boldsymbol{a}_n)}{\det A}, \quad 1 \le i \le n.$$

计算量

当$n = 25$时大约需要10^{26}次运算。
世界最快的计算机美国橡树岭国家实验室的Frontier（前沿），每秒钟运算速度为1×10^{18}次，求解一个25阶的代数方程组需要10^8秒。

$$10^8\text{秒} \approx 4\text{年}$$

这说明不能通过求逆矩阵在计算机上求解线性代数方程组。
而在实际问题中方程组的规模大都为

$$n = 10^6 \sim 10^8$$

必须有数值上有效的算法来求解这些方程组。
用Gauss 消去法求解线性方程组，其计算量为$O(n^3)$。

九章算术中的Gauss消去法

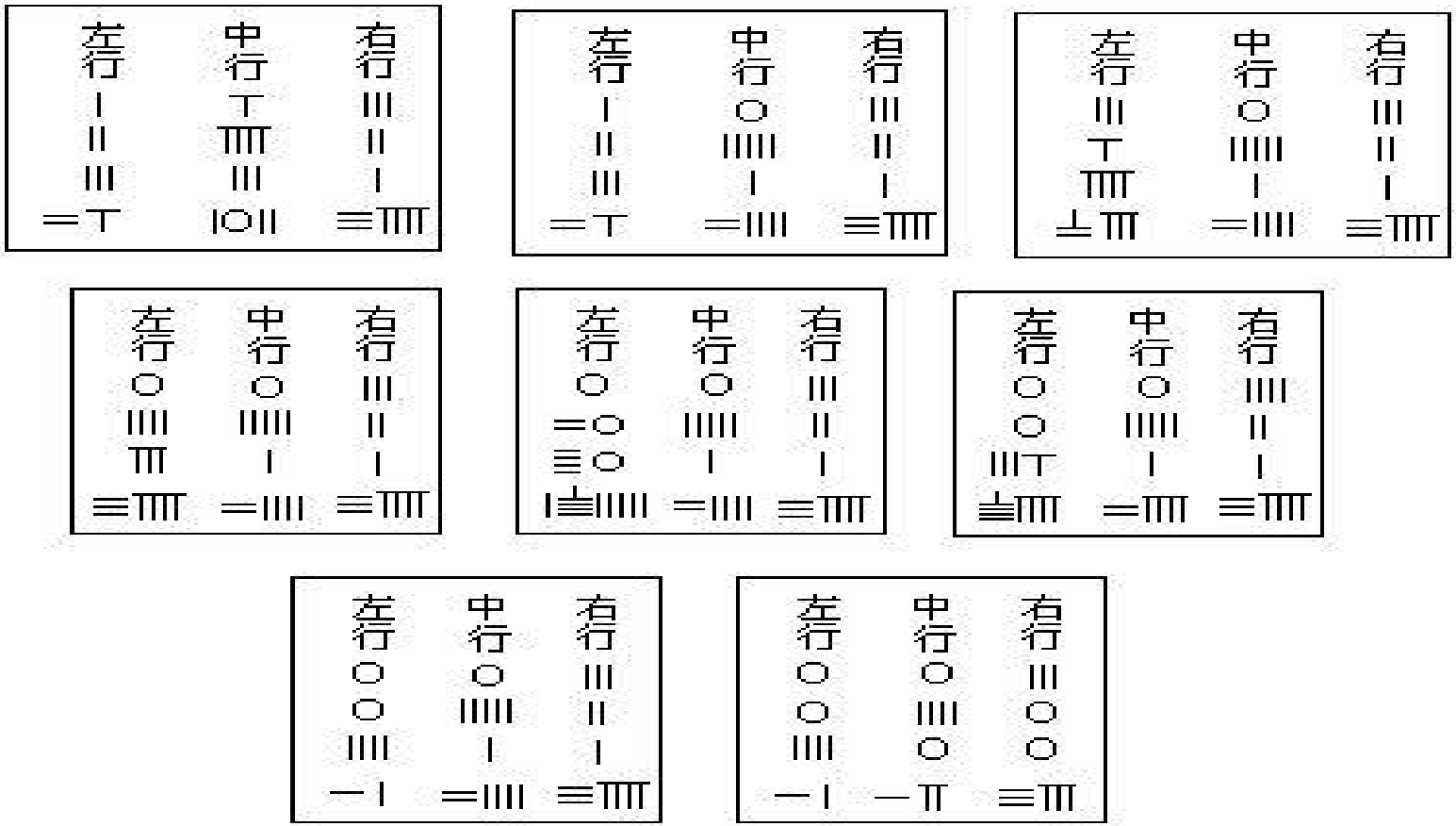


图 1-29

Hilbert矩阵

求解线性方程组

$$\begin{cases} x_1 + \frac{1}{2}x_2 + \frac{1}{3}x_3 = \frac{11}{6} \\ \frac{1}{2}x_1 + \frac{1}{3}x_2 + \frac{1}{4}x_3 = \frac{13}{12} \\ \frac{1}{3}x_1 + \frac{1}{4}x_2 + \frac{1}{5}x_3 = \frac{47}{60} \end{cases}$$

精确解:

$$x_1 = x_2 = x_3 = 1$$

取三位十进制有效数字在计算机用消去法计算

$$x_1 = 1.089,\ x_2 = 0.488,\ x_3 = 0.491$$

条件数

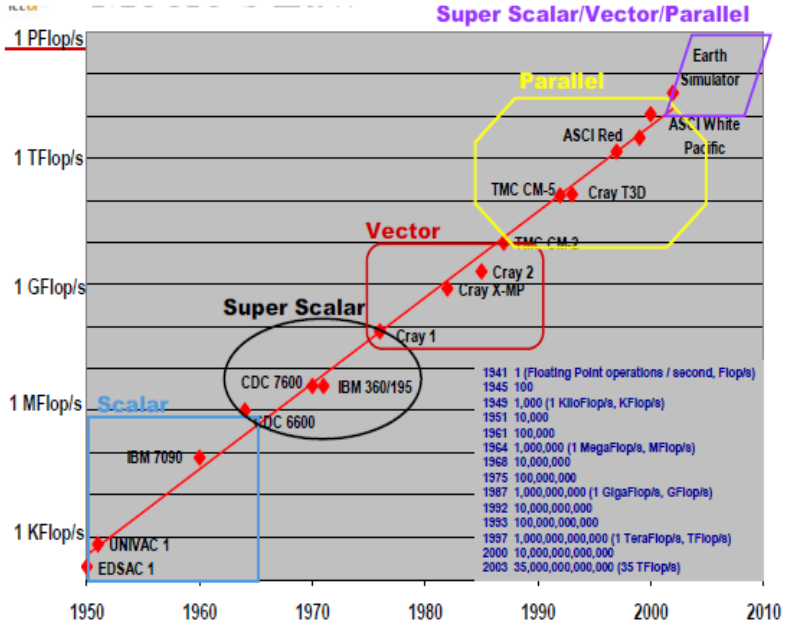
$$\mathrm{cond}(A) = \|A\|\|A^{-1}\| = 748.$$

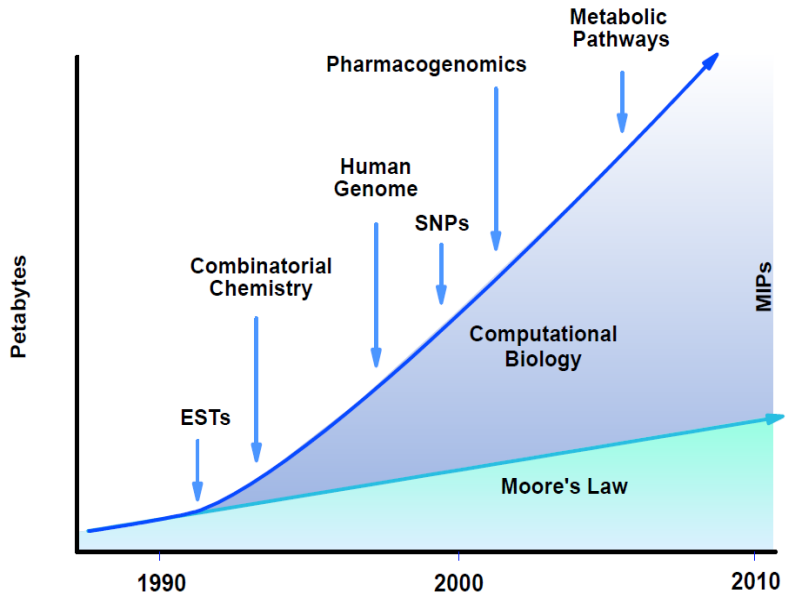
微分方程与人类文明

是微分方程引领着人类文明进程：

- Newton运动方程
- LapLace 方程
- Maxwell电磁学方程
- Fourier热传导方程
- Einstein广义相对论方程
- Yang-Mills规范场方程
- 量子力学的Schrödinger方程
- Navier-Stokes方程
- 弹塑性力学方程
- Black-Scholes方程（期权定价理论）

计算机硬件发展和实际计算需求的差距





1. 大规模复杂问题的建模：非稳态，多物理，多尺度问题；
2. 处理不确定和随机效应；
3. 人工智能，机器学习；
4. 大数据和从数据到知识；
5. 大型的优化、评估和设计问题。

算法的重要性

许多科学与工程计算问题都有如下特点：

1. 高维数、多尺度、非线性、不适定、大时间、奇异性、 复杂区域、高度病态；
2. 计算规模大；
3. 精度要求高。

人类的计算能力主要依赖两个方面：

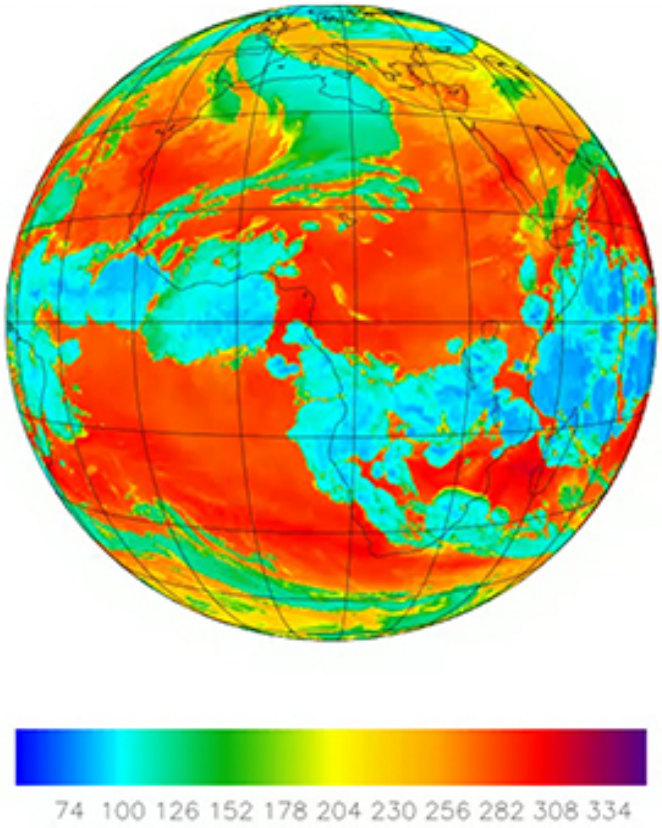
1. 计算机的性能，
2. 计算方法的效率。

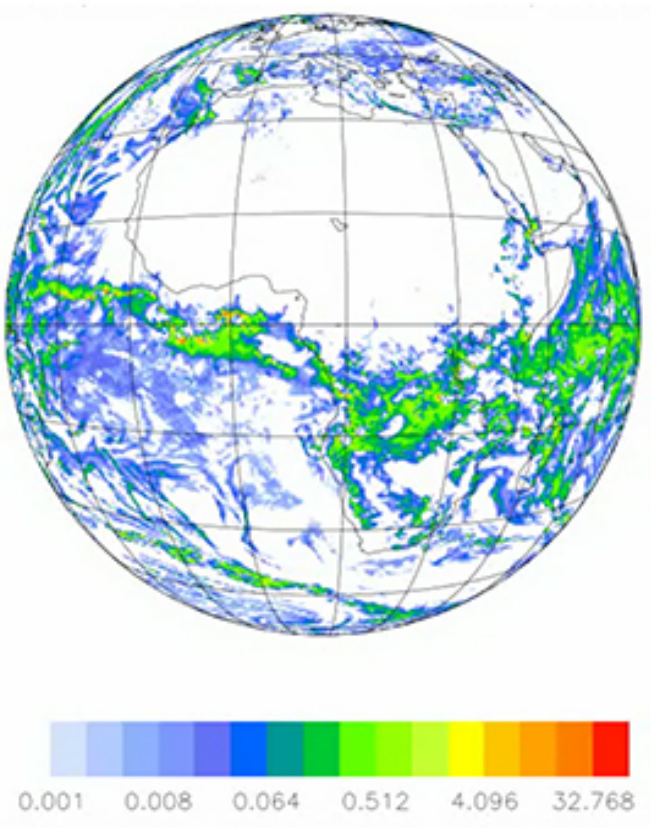
计算方法的发展对于计算能力的提高和计算机的进步是同等重要的。

北盘江大桥

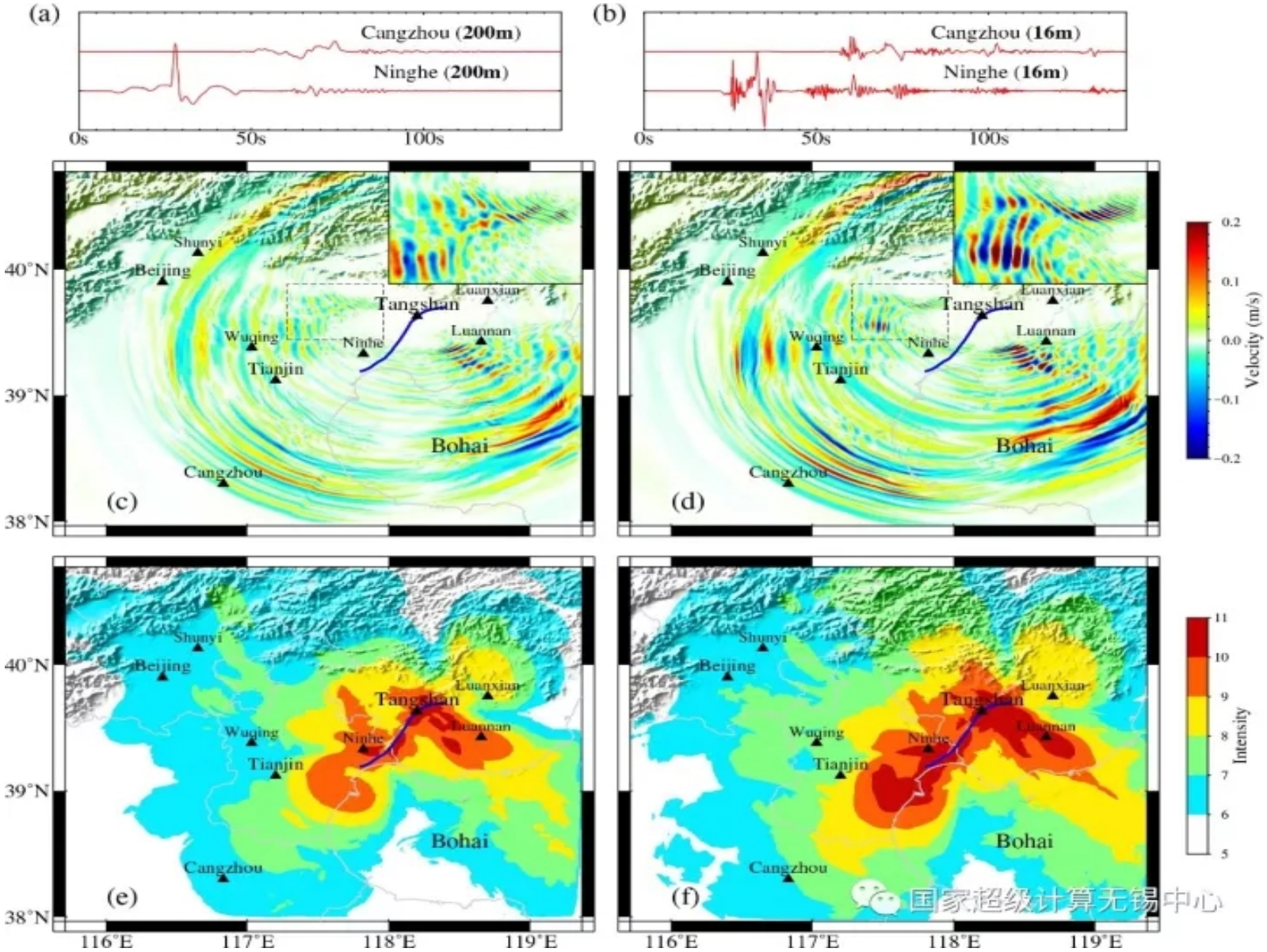
北盘江大桥

神威·太湖之光连获戈登·贝尔奖–2016





神威·太湖之光连获戈登·贝尔奖–2017

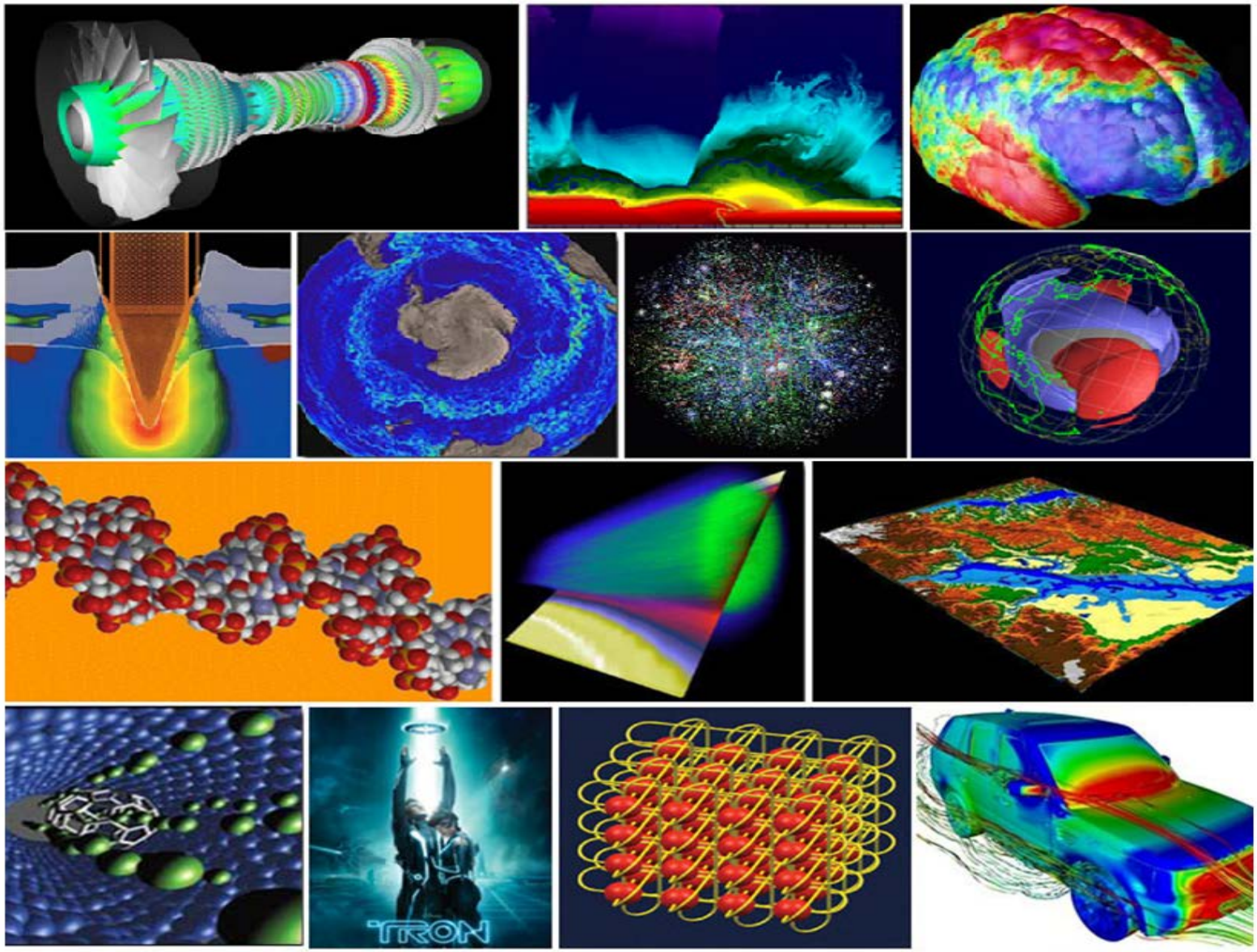


20世纪十大算法

Computing in Science and Engineering 杂志评出了10大算法：

1. Metropolis算法（Metropolis Algorithm）；
2. 单纯型法（Simplex method）；
3. Krylov 子空间迭代法(Krylov subspace iteration methods)
4. 矩阵分解技术（Matrix decomposition approach）；
5. Fortran 编译器（Fortran Complier）；
6. QR 算法（QR algorithm）；
7. 快速排序算法（Quicksort algorithm）；
8. 快速Fourier 变换（FFT）；
9. 整数关系检测（Integer relation detector）；
10. 快速多极子算法（Fast multipole method）.

数值模拟



好的算法

设计好的算法尤为重要。

公认的好的算法的标准：

- 运算次数少；
- 运算过程具有规律性，便于编制程序；
- 要记录的中间结果少；
- 能够控制误差的传播和积累，以保证精度。

误差来源

误差的来源有四个方面：

1. 模型误差 从实际问题得到数学模型时往往忽略了许多次要因素，因而即使数学模型能够求出 精确解，也与实际问题的真解不同，它们之间的偏差就是模型误差。
2. 观测误差 由观测、试验等方法得到的数值必然带有误差，这种误差称作观测误差。
3. 截断误差 理论上的精确值往往要用无限次的运算才能达到，而实际计算只能用有限次运算的 结果来近似，由此产生的误差叫做截断误差。
4. 舍入误差 计算机只能对有限数字进行运算，每个超出计算机字长的数据都要经过取舍 或截断方法处理，这样引起的误差即为舍入误差。

误差的例子

想知道地球的表面积 A 是多少，近似的把地球看成球体:

$$A = 4\pi r^2, \quad r \text{ 是地球半径}$$

1. 即使这一公式可以精确计算，所得值也和地球表面积的真值有一定的误差， 这就是模型误差。
2. 而在实际计算中如果取$r = 6370km$，这一观测值具有一定的误差，这就是观测误差。
3. 公式中的π 是圆周率，它是一个无理数，必须取它的有限位，例如取$\pi = 3.141$，此处就产生了 截断误差。
4. 最后将$4 \times 3.141 \times 6370^2$ 的计算结果进行四舍五入得$A = 5.0981 \times 10^8$, 这一步就产生了舍入误差。

定义 (绝对误差和相对误差)

设 x是精确值，x_A 是一个近似值，

$$e(x) = |x - x_A| \qquad \text{称为}x_A\text{的 绝对误差，}$$

而

$$e_r(x) = \frac{|x - x_A|}{|x|} \qquad \text{称为 } x_A \text{ 的相对误差。}$$

因为 x 通常是未知的，一般只能给出 $|x - x_A|$ 的上界 ϵ_A，即

$$|x - x_A| \le \epsilon_A, \qquad \text{称}\epsilon_A \text{ 是 } x_A \text{ 的绝对误差界。}$$

类似的

$$\epsilon_r = \frac{\epsilon_A}{|x|} \qquad \text{是}x_A\text{的相对误差界。}$$

误差的例子

绝对误差和相对误差那个更能刻画误差的本质？

1. $x = 0.3000 \times 10, x_A = 0.3100 \times 10,$
   $e(x) = 0.1, e_r(x) = 0.3333 \times 10^{-1}.$
2. $x = 0.3000 \times 10^{-3}, x_A = 0.3100 \times 10^{-3},$
   $e(x) = 0.1 \times 10^{-4}, e_r(x) = 0.3333 \times 10^{-1}.$
3. $x = 0.3000 \times 10^{4}, x_A = 0.3100 \times 10^{4},$
   $e(x) = 0.1 \times 10^{3}, e_r(x) = 0.3333 \times 10^{-1}.$

相对误差更能刻画出误差的本质。

浮点数系统

在计算机上，实数系 $\mathbb{R}$ 是用浮点数系统(floating point)$\mathbb{F}$ 来近似的。实数x在β进制浮点数系统下的表示为：

$$x = (-1)^s \cdot (0.a_1 a_2 \cdots a_t) \cdot \beta^e = (-1)^s \cdot m \cdot \beta^{e-t}$$

其中

- s：符号位，1为负数，0为正数；
- $m = a_1 a_2 \cdots a_t, 0 \le a_i \le \beta - 1$；尾数（有效数字部分），$t$称为字长；
- e, $L \le e \le U$：指数。

浮点数系统一般可表示为

$$\mathbb{F} = \{x \in \mathbb{R} : x = (-1)^s \beta^e \sum_{i=1}^{t} a_i \beta^{-i}\} \cup \{0\},$$

常用的浮点数系统

系统	进制 β	t	L	U
IEEE 单精度	2	23	-126	127
IEEE 双精度	2	52	-1022	1023
Cray	2	48	-16383	16384
HP calculator	10	12	-499	499
IBM mainframe	16	6	-64	63

如2进制的浮点数系统$\mathbb{F}(2,t,L,U)$定义为：

$$\mathbb{F} = \{\pm 0.d_1 d_2 \cdots d_t \times 2^k\} \cup \{0\},$$

其中 $d_1 = 1, d_j (2 \le j \le t)$ 为0 或 1。

二进制表示

$\forall x \in \mathbb{R}$ 总可以写成

$$\begin{aligned} x &= \pm 0.1 d_2 \cdots d_t d_{t+1} \cdots \times 2^k \\ &= \pm (1 \times 2^{-1} + d_2 2^{-2} + \cdots + d_t 2^{-t} + \cdots) 2^k \end{aligned}$$

Q: 请写出81.625的二进制表达式?

记 x 在一个浮点数系统 $\mathbb{F}$ 中的表示为$fl(x)$，一般说来，$fl(x)$ 只是 x 的某种近似。常用的近似方法有两种

- 截断 (chop)，$x = \pm 0.1 d_2 \cdots d_t \times 2^k$；
- 舍入 (rounding)，$x = \pm 0.1 d_2 \cdots \delta_t \times 2^k$，$\delta_t = d_t, \text{if } d_{t+1} = 0$，$\delta_t = d_t + 1, \text{if } d_{t+1} = 1$(0舍1入)。

IEEE 采用的是舍入，则

$$\begin{aligned} \left| \frac{x - fl(x)}{x} \right| &= \left| \frac{0.1 d_2 \cdots d_t d_{t+1} \cdots \times 2^k - 0.1 d_2 \cdots \delta_t \times 2^k}{0.d_1 d_2 \cdots \times 2^k} \right| \\ &\leq \left| \frac{0.d_{t+1} d_{t+2} \cdots}{0.d_1 d_2 \cdots} \right| \times 2^{-t} \leq \frac{2^{k-t-1}}{2^{k-1}} = 2^{-t} \end{aligned}$$

这就是实数的浮点数表示产生的误差，称为机器精度，记为ϵ_{mach}。

1. 单精度：符号1位，阶8位，$t = 23$。

$$\epsilon_{mach} = 2^{-23} \approx 10^{-7}$$

2. 双精度： 符号1位，阶11位，$t = 52$。

$$\epsilon_{mach} = 2^{-52} \approx 10^{-16}$$

因此在浮点数系统中有

$$fl(x) = x(1 \pm \epsilon), \quad \epsilon \leq \epsilon_m.$$

浮点数系统 $\mathbb{F}$ 具有下列值得注意的特点：

- 它是实数系 $\mathbb{R}$ 的一个只包含 $\beta^t(U-L+1)+1$ 个数的有限集，这些数对称地离散分布在区间 $[\mathrm{UFL}, \mathrm{OFL}]$ 和 $[-\mathrm{OFL}, -\mathrm{UFL}]$ 中，这里UFL是下溢限(underflow limit)，OFL 是上溢限(overflow limit)。在2进制下，

$$\mathrm{UFL} = 2^L \times 0.1, \quad \mathrm{OFL} = 0.11\cdots1 \times 2^U,$$

- 具有最小正数 $\mathrm{UFL} = 2^L \times 0.1$；
- 具有最大正数 $\mathrm{OFL} = 0.11\cdots1 \times 2^U$；
- 具有机器精度 $\epsilon_{mach} = 2^{-t}$；
- 其中的四则运算并不满足实数系中的运算规律。

在一个浮点数系统中，绝对值超过 OFL的实数值就被认为是无穷大，绝对值小于 UFL 的实数就被认为是零。

Matlab默认为双精度：

- » Xmax=realmax

  Xmax =1.7977e+308
- » Emin=realmin

  Emin =2.2251e-308
- » eps

  ans =2.2204e-16
- » log2(Xmax)

  ans =1024
- » log2(Emin)

  ans =-1022
- » log2(eps)

  ans =-52

定义 (有效数字)

设x_A 是x 的一个近似值，

$$x_A = \pm 0.a_1 a_2 \cdots a_i \cdots \times 10^k,$$

其中 k为整数， $a_i \in \{0, 1, 2, \cdots, 9\}, a_1 \neq 0$。 如果有

$$|x - x_A| \leq \frac{1}{2} \times 10^{k-n},$$

则称 x_A 具有 n 位有效数字： $a_1, a_2, \cdots a_n$。

Q:

$$\pi_A = 3.1416 = 0.31416 \times 10$$

是 π 的近似值，具有几位有效数字？

有效数字与相对误差

定理

设x 的近似值 $x_A = \pm 0.a_1a_2\cdots a_i\cdots \times 10^k$, 则

1. 如果 x_A 有n 位有效数字，则

$$\frac{|x - x_A|}{|x_A|} \leq \frac{1}{2a_1} \times 10^{1-n}.$$

2. 如果

$$\frac{|x - x_A|}{|x_A|} \leq \frac{1}{2(a_1 + 1)} \times 10^{1-n},$$

则 x_A 至少有n位有效数字。

计算机实行对位及有限字长的“大数吃小数”。 在字长为8的10进制浮点数系统下算 $(x+y)+z$ 和 $x+(y+z)$，其中

$$\begin{aligned} x &= 0.23371258 \times 10^{-4}, \\ y &= 0.33678429 \times 10^{2}, \\ z &= -0.33677811 \times 10^{2}, \end{aligned}$$

则由浮点数的运算规则

$$(x+y)+z = 0.64100000 \times 10^{-3},$$
$$x+(y+z) = 0.64137126 \times 10^{-3},$$

精确值 $x+y+z = 0.641371258 \times 10^{-3}$.

向前误差分析

在浮点数系统中设x 的浮点数表示为 x^*，则

$$\begin{aligned} x^* &= x(1+\delta), \\ (x^*)^2 &= x^2(1+2\delta), \end{aligned}$$

每次乘法相对误差加倍!!!

用Gauss 消去计算

$$A\boldsymbol{x} = \boldsymbol{b},\ A \in \mathbb{R}^{n\times n},\ \boldsymbol{x}, \boldsymbol{b} \in \mathbb{R}^n$$

若 $n = 10^6$，则需进行 $O(10^{18})$次浮点数运算。若用双精度的浮点数运算，最后的相对误差

$$10^{-16} \times 10^{18} = 10^2.$$

向后误差分析

设

$$f(x) = x^2, \quad (f(x))^* = x^2(1+\rho),$$

其中 $|\rho| < \epsilon_{mach}$，则

$$\exists \tilde{\rho}, |\tilde{\rho}| < |\rho| \text{ s.t } (1+\tilde{\rho})^2 = 1+\rho$$

因此

$$[f(x)]^* = x^2(1+\tilde{\rho})^2 = f(x(1+\tilde{\rho})).$$

函数 $f(x)$即函数值$\hat{y}$，通常 $\hat{y} \neq f(x)$。但可类似证明：存在 δx 满足

$$\hat{y} = f(x+\delta x), \quad |\delta x| \leq |x|\epsilon_{mach},$$

误差分析的要点

定义 (向后误差分析)

这种把计算结果的误差归结为原始数据经扰动之后的精确的计算结果的误差分析方法叫做 向后误差分析法。

1. 向前误差分析： $(x^*)^2 = x^2(1+2\delta)$，损失精度；
2. 向后误差分析： $[f(x)]^* = f(x(1+\tilde{\rho}))$。若 f 的性质很好，则无需担心浮点数 的舍入误差引起精度损失。

计算结果的精度估计：

$$\frac{|\hat{y} - f(x)|}{|f(x)|} = \frac{|f(x+\delta x) - f(x)|}{|f(x)|} \leq c(x)\frac{|\delta x|}{|x|} \leq c(x)\epsilon.$$

条件数

敏度分析是研究x 有微小变化δx时函数值f会发生多大的变化。

定义 (条件数)

$$\frac{|f(x+\delta x)-f(x)|}{|f(x)|} \leq \kappa(x)\frac{|\delta x|}{|x|}.$$

称$\kappa(x)$ 为 f 在 x 点的条件数。$\kappa(x)$反映了自变量的微小变化对函数值的影响程度。

定义 (病态问题)

当 $\kappa(x)$ 很大时，自变量的微小变化就有可能引起函数值的巨大变化，此时称 f 在 x 点是 病态的；反之，若 $\kappa(x)$ 较小，称f 在 x 点是 良态的。

可微函数的条件数

注记

一个计算问题是否病态是问题本身的固有属性。

设$f(x)$ 是一个可微函数， x_A是x 的近似值，则

$$\begin{aligned}
\kappa(x) &= \lim_{x_A \to x} \frac{\left|\dfrac{f(x)-f(x_A)}{f(x)}\right|}{\left|\dfrac{x-x_A}{x}\right|} \\
&= \left|\frac{x}{f(x)}\right| \lim_{x_A \to x} \left|\frac{f(x)-f(x_A)}{x-x_A}\right| \\
&= \left|\frac{xf'(x)}{f(x)}\right|.
\end{aligned}$$

例

求下列函数的条件数，并说明其是否病态

1. $f(x) = \sqrt{x}$
2. $f(x) = \frac{1}{1-x}$

Wilkinson 多项式

考虑下面的多项式：

$$w(x) = \prod_{i=1}^{20}(x-i) = (x-1)(x-2)\cdots(x-20).$$

展开后 x^{19} 的系数为 -210。

Wilkinson 发现对此系数作一个小的扰动：

$$-210 \to -210 - 2^{-23},$$

其中的根变为

$$x = 16, 17 \to 16.73 \pm 2.81i$$

而系数的扰动：

$$2^{-23} \approx 1.192 \times 10^{-7}$$

数学家简介 (James H. Wilkinson)

詹姆斯·哈迪·威尔金森*(1919—1986)*，英国皇家学会院士、著名的数值分析专家，研制*ACE*计算机的功臣，在建造由图灵设计的*ACE*计算机中扮演了举足轻重甚至最关键的角色，获得了*1970*年的图灵奖。其主要学术贡献有：向后误差分析法；浮点测试等。*1976*年，威尔金森积极参与并推动成立了一个非赢利性的名为*NAG*的公司（*Numerical Algorithms Group Ltd.*）以开发和推广数值分析和统计分析的软件包，吸引了世界上许多大学和政府研究机构的专家共同合作。*NAG*已经为*68*种型号的计算机配备了*Fortran*库，*Ada*、*Pascal*、*C*的通用数学库也已上市。*J. H. Wilkinson*奖从*1991*年开始，*4*年颁发一次，颁给在数值软件领域做出突出贡献的科学家。

图灵奖（1970年）

冯·诺伊曼奖（1970年）

Chauvenet奖（1987年）

矩阵特征值的计算

对于方阵$A \in \mathbb{C}^{n\times n}$，若有$\lambda \in \mathbb{C}$和非零向量$\boldsymbol{x} \in \mathbb{C}^n$，使得

$$A\boldsymbol{x} = \lambda \boldsymbol{x}$$

则称$\boldsymbol{x}$为特征值(eigenvalue)λ对应的特征向量(eigenvector)。
矩阵的特征值亦可通过它的特征多项式$p(\lambda) = \det(\lambda I - A)$的根来定义：

$$p(\lambda) = \lambda^n - c_1\lambda^{n-1} + \cdots + (-1)^n c_n$$

其中

$$c_1 = \mathrm{trace}(A), \quad c_n = \det(A).$$

Q：上述特征值的定义是否可以用来数值求解特征值？

运算误差分析

设 $y=f(x_1,x_2,\cdots,x_n)$，如果参量 $x_i, 1\le i\le n$ 有误差，则y也会有误差。若$x_i, 1\le i\le n$ 的近似值为 $\tilde{x}_i, 1\le i\le n$，相应的解 $\tilde{y}$，则$\tilde{y}$的绝对误差和相对误差分别为

$$\begin{aligned} |y-\tilde{y}| &= |f(x_1,x_2,\cdots,x_n)-f(\tilde{x}_1,\tilde{x}_2,\cdots,\tilde{x}_n)| \\ \frac{|y-\tilde{y}|}{|y|} &= \frac{|y-\tilde{y}|}{|f(x_1,x_2,\cdots,x_n)|}. \end{aligned}$$

由此得到误差估计

$$\begin{aligned} |y-\tilde{y}| &\le \sum_{i=1}^{N}\left|\frac{\partial f}{\partial x_i}\right||x_i-\tilde{x}_i|, \\ \frac{|y-\tilde{y}|}{|y|} &\le \sum_{i=1}^{N}\left|\frac{x_i}{y}\frac{\partial f}{\partial x_i}\right|\frac{|x_i-\tilde{x}_i|}{|x_i|}, \end{aligned}$$

把两个数 a, b的$+, -, \times, \div$ 看成二元函数，则有

$$
\begin{aligned}
e(a \pm b) &\leq e(a) + e(b) \\
e(ab) &\leq |b|e(a) + |a|e(b) \\
e_r(a \pm b) &\leq \frac{e(a) + e(b)}{|a \pm b|} = \frac{|a|e_r(a) + |b|e_r(b)}{|a \pm b|} \\
e_r(ab) &\leq \frac{e(a)}{|a|} + \frac{e(b)}{|b|} = e_r(a) + e_r(b) \\
e(\frac{a}{b}) &\leq \frac{e(a)}{|b|} + \left|\frac{a}{b^2}\right| e(b) \\
e_r(\frac{a}{b}) &\leq e_r(a) + e_r(b).
\end{aligned}
$$

尽量避免的运算

运算原则：

- 避免相近的数相减，因为相近数相减会严重丢失有效数字。
- 作乘法运算时，避免和绝对值大的数相乘。
- 作除法运算时，避免和绝对值小的除数相除。

避免有效数字的损失

方程$ax^2 + 2bx + c = 0$ 的两个根的公式为

$$x_1 = \frac{-b + \sqrt{b^2 - ac}}{a}, \quad x_2 = \frac{-b - \sqrt{b^2 - ac}}{a},$$

如果$b^2 \gg |ac|$，则 $\sqrt{b^2 - ac} \approx |b|$，用上面的公式计算$x_1$和$x_2$，其中的一个将 会损失有效数字。如方程 $x^2 - 62.10x + 1 = 0$的根分别为

$$x_1 = -0.01610723, \quad x_2 = -62.08390.$$

而 $\sqrt{b^2 - ac} = 62.06$，则

$$fl(x_1) = \frac{-62.10 + 62.06}{2.000} = \frac{-0.04000}{2.000} = -0.02000$$

而 $e_r(x_1) \approx 2.4 \times 10^{-1}$

数学上等价≠ 计算上等价

很多算法在数学上是等价的，但是在计算机上计算时表现迥异。

例 (方程求根)

上面的二次方程的求根问题，有等价的公式：

$$x_1 = \frac{-b - sgn(b)\sqrt{b^2 - ac}}{a}, \quad x_2 = \frac{c}{ax_1}.$$

例

$f(x) = \sqrt{1+x^2} - 1$, 当 $|x| \ll 1$ 时直接计算会损失精度，数学上可等价的写为

$$f(x) = \frac{x^2}{\sqrt{1+x^2}+1}$$

这个公式可避免有效数字的损失。

- 减少运算次数；

  计算

$$f(x) = x^3 - 6.1x^2 + 3.2x + 1.5$$

  在$x = 4.71$处的值（保留3位有效数字），$f(4.17) = -14.263899$，而$fl(f(4.17)) = -13.4$，相对误差为

$$\left| \frac{-14.263899 + 13.4}{-14.263899} \right| \approx 0.06$$

  而$f(x)$可以写成下面等价的形式

$$f(x) = x^3 - 6.1x^2 + 3.2x + 1.5 = ((x - 6.1)x + 3.2)x + 1.5$$

  用上面的公式计算$fl(f(4.17)) = -14.3$，相对误差为0.0025，精度大大提高。

- 避免计算过程中出现上溢和下溢的问题。

一般的n次多项式

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

有著名的秦九韶算法

$$\begin{aligned} u_n &= a_n, \\ u_k &= u_{k+1} x + a_k, \quad k = n-1, \cdots, n-2, 1, 0 \\ p_n(x) &= u_0. \end{aligned}$$

只需 n 次乘法和 n 次加法。

数学家简介 (秦九韶)

秦九韶（1208年 – 1261年），南宋官员、数学家，与李冶、杨辉、朱世杰并称宋元数学四大家。1247年完成著作《数书九章》，其中有大衍求一术、三斜求积术和秦九韶算法。

选对公式

利用公式

$$\ln(1+x)=\sum_{n=1}^{\infty}(-1)^{n+1}\frac{x^n}{n}$$

可以计算$\ln(1+x)$的近似值。若求$\ln 2$，则在上式中令$x=1$即可求得近似值。如果精确到10^{-5}，则需$n=10^5$！

计算量大，舍入误差累计严重！

若改用级数

$$\ln\frac{1+x}{1-x}=2\left(x+\frac{x^3}{3}+\frac{x^5}{5}+\cdots+\frac{x^{2n+1}}{(2n+1)}+\cdots\right)$$

取$x=\frac{1}{3}$，则只需$n=9$即可使截断误差小于10^{-10}。

复运算

一个给定的复数可写为

$$z = x + \mathrm{i}y, \quad \mathrm{i} = \sqrt{-1}, \quad x = \mathrm{Re}z, \quad y = \mathrm{Im}z$$

则两个复数 $z_1 = x_1 + \mathrm{i}y_1$ 和$z_2 = x_2 + \mathrm{i}y_2$的标准运算定义为

$$\begin{aligned}
z_1 + z_2 &= (x_1 + x_2) + \mathrm{i}(y_1 + y_2), \\
z_1 - z_2 &= (x_1 - x_2) + \mathrm{i}(y_1 - y_2), \\
z_1 \times z_2 &= (x_1 x_2 - y_1 y_2) + \mathrm{i}(x_1 y_2 + x_2 y_1), \\
\frac{z_1}{z_2} &= \left(\frac{x_1 x_2 + y_1 y_2}{x_2^2 + y_2^2}\right) + \mathrm{i}\left(\frac{x_2 y_1 - x_1 y_2}{x_2^2 + y_2^2}\right), \quad z_2 \neq 0.
\end{aligned}$$

复数运算的成本要远远高于实数运算。

数值稳定性

定义 (数值稳定性)

一个数值方法，如果初始数或计算过程某一步有微小的改变，由此引起的计算结果也只是微小的变化，则称该方法是数值稳定的，否则称为数值不稳定的。

一般说来，如果具有初始误差$\epsilon_0 > 0$，记计算n步后的误差为ϵ_n，若方法是数值稳定的，则存在与n无关的常数C使得

$$|\epsilon_n| \leq C\epsilon_0.$$

常见误差传播模式：

1. $|\epsilon_n| \approx Cn\epsilon_0$，线性型的误差增长；
2. $|\epsilon_n| \approx C^n\epsilon_0$，指数型的误差增长。

数学家简介 (John von Neumann)

20世纪最重要的数学家之一，在现代计算机、博弈论、核武器和生化武器等领域内的科学全才之一。计算数学的缔造者之一，他首先研究线性代数和算术的数值计算，后来着重研究非线性微分方程的离散化以及稳定问题，并给出误差的估计法，发展了蒙特卡罗方法，被后人称为“计算机之父”和“博弈论之父”。

早期以算子理论、共振论、量子理论、集合论等方面的研究闻名，开创了冯·诺依曼代数。第二次世界大战期间为第一颗原子弹的研制作出了贡献。为研制电子数字计算机提供了基础性的方案。1944 年与摩根斯特恩合著《博弈论与经济行为》，是博弈论学科的奠基性著作。晚年，研究自动机理论，著有对人脑和计算机系统进行精确分析的著作《计算机与人脑》。

图: 冯·诺依曼（1903–1957），著名匈牙利裔美籍数学家，计算机科学家，物理学家和化学家。

数值稳定性示例

例

设 $x_n = \frac{1}{3^n}$，可用下面的三种方法生成序列

1. $x_0 = 1,\ x_n = \frac{1}{3}x_{n-1},\ n = 1, 2, \cdots;$
2. $x_0 = 1,\ x_1 = \frac{1}{3}, x_n = \frac{4}{3}x_{n-1} - \frac{1}{3}x_{n-2},\ n = 2, 3, \cdots;$
3. $x_0 = 1,\ x_1 = \frac{1}{3},\ x_n = \frac{10}{3}x_{n-1} - x_{n-2},\ n = 2, 3 \cdots。$

若用有限位的小数运算来实现上面的三方法：

1. $r_0 = 0.99996,\ r_n = \frac{1}{3}x_{n-1},\ n = 1, 2, \cdots;$
2. $p_0 = 1,\ p_1 = 0.33332, p_n = \frac{4}{3}p_{n-1} - \frac{1}{3}p_{n-2},\ n = 2, 3, \cdots;$
3. $q_0 = 1,\ q_1 = 0.33332,\ q_n = \frac{10}{3}q_{n-1} - q_{n-2},\ n = 2, 3 \cdots。$

试分析其计算结果。

商业软件的陷阱

1. 理论上，为了估算计算的精度和数值结果可信度，在计算过程中必须有条件数和精度的估计。程序的用户对计算中可能遇到的困难有所预判并且知道如何规避和克服这些困难。这些问题应该成为科学家和工程师使用和开发现代计算软件 的核心课程。
2. 但是在工业的实践中，这些因素都被忽略了，甚至用某个算法求解问题时可能会出现问题的警示也没有。这种看起来很不负责的行为的原因是什么？原因之一是估计精度和条件数会降低算法的效率，但最主要的原因是用户无法正确理解这些警示。
3. 算法软件的开发者引入这些警示信息并不是为了骚扰用户和显摆数学，真正的目的是为了让科学家和工程师了解要问题和算法会遇到什么困难，并且教会他们如何理解和使用这些警示信息。

常用数学软件介绍

我们要讨论的算法都已经有前人用各种计算机编程语言写成程序并做成软件，这类软件叫数学软件。

互联网 (Internet) 是查找和获得数学软件的主要工具。推荐的三个网站

- http://www.netlib.org
- http://gmas.nist.gov
- http://sourceforge.net

其中第一个网站在中科院计算数学与科学工程计算研究所有镜像：netlib.amss.ac.cn

数学软件的分类

数学软件基本上可以分成三种类型：

- 数学软件包：针对某一类数值计算问题的子程序集。
- 数学软件库：面向各类数值计算问题的子程序集。
- 交互式科学计算环境：面向各类数值计算、符号计算和图形显式问题且具有交互式用户界面的 科学计算工作平台。

数学软件包

- LAPACK 是求线性代数方程组、线性最小二乘问题和矩阵特征值、奇异值问题的 FORTRAN 77 子程序包。
- ITPACK 是迭代求解大型稀疏线性代数方程组的FORTRAN 77 子程序包。
- MINPACK 是求解非线性方程组和非线性最小二乘问题的 FORTRAN 77 子程序包。
- ODEPACK是求解常微分方程的 FORTRAN 77子程序包。
- ScaLAPACK是可扩展的并行数值线性代数软件包。
- DIFFPACK是用Visual C++ 语言开发的求解偏微分方程的商业软件。www.diffpack.com.

数学软件库

- IMSL 是由 Visual Numerics 公司开发的商业软件，是用FORTRAN和 C语言编写的数值 计算与统计程序库，一般与UNIX工作站捆绑在一起销售。 www.vni.com
- NGA是由 Numerical Algorithms Group (NGA)公司开发的数值计算和统计的 FORTRAN和 C 程序库。可在各种硬件平台上运行。www.nag.com
- NR是与数值计算领域的经典工具书 《Numerical Recipes》一起发布的 FORTRAN、 C和 C++语言数值计算软件库，可在www.nr.com上免费下载。
- SLATEC 是由美国能源部发布的大型数值与统计计算 FORTRAN 程序库，可在 www.netlib.org 上免费下载。

交互式科学计算环境

- MATLAB 是 MathWorks公司开发的商业软件。它以矩阵运算为核心而且拥有数百个内部函数和各种工具箱。 其内部函数用于解决基本的数值计算与图形显示问题。而工具箱分为功能性工具箱和学科工具箱。MATLAB被广泛的应用于科学研究和解决各种具体问题。其官方主页为 www.mathworks.com
- MAPLE是由加拿大 Waterloo大学计算机系开发的商业数学分析软件。与MATLAB相比，它的数值计算能力 较弱，但其符号计算功能较强。官方网址 www.maplesoft.com
- MATHEMATICA是由美国 Wolfram Research 公司拥有版权的商业数学分析软件。其功能与 MAPLE类似，长于符号计算。官方主页： www.mathematica.com.

- SCILAB是由法国国家信息与自动化研究院(INRIA)开发的自由数学软件。它的功能和MATLAB相似，而且和 MATLAB 是兼容的。但是它的学科工具箱不如MATLAB丰富。其最大的有点是源代码完全公开，使用者可以根据 需要自行修改利用。官网： www.scilab.org
- GAUSS是由Aptech Systems 公司开发的商业软件。它有很强的统计分析能力，因而在计量经济学中应用 非常广泛。上海和深圳证券交易所每日收盘的综合股价指数以及总交易量的数据都是用GAUSS处理的。 官网：www.aptech.com
- Python是一种面向对象、直译式电脑编程语言，也是一种功能强大的通用型语言。它包含了一组完善而且容易理解的标准库， 能够轻松完成很多常见的任务。它的语法非常简捷和清晰，与其它大多数程序设计语言不一样，它使用缩进来定义语句。最近在数据科学和机器学习领域应用广泛。