




数值分析(1)

Numerical Analysis

计算机系 软件所 喻文健



本课件更适合于用电脑观看
(手机用于雨课堂答题)

现场的同学务必佩戴口罩，
不要看雨课堂视频

建议上课时将**纸**、**笔**放在旁边!

课程信息

■ 授课教师：喻文健

□ **Web:** <http://numbda.cs.tsinghua.edu.cn>

□ **E-mail:** yu-wj@tsinghua.edu.cn

□ **Tel:** 62773440, 东主楼8区407室

□ **助教:** 杨明, yang-m17@mails.tsinghua.edu.cn
冯栩, fx17@mails.tsinghua.edu.cn
李凌劼, li-lj18@mails.tsinghua.edu.cn
杨定澄, ydc19@mails.tsinghua.edu.cn
刘志强, liu-zq20@mails.tsinghua.edu.cn

■ 上课时间：周三上午9:50–12:15

■ 答疑：网络答疑(email, 微信), 周三下午2:00-3:00

教学团队



□ 喻文健



□ 冯栩

计81/86
计7级



□ 杨定澄

计82/85
计9级



□ 刘志强

计83/84



□ 李凌劼

Wenjian Yu

计科
91/92



□ 杨明

其他
班号

课程简介

- 计算方法
- 数值分析与算法
- 科学计算导论 (*scientific computing*)
- 数值计算基础 (*numerical computing*)
- 课程目标
 - 介绍广泛应用于科学与工程领域的各种数值计算方法
 - 巩固连续数学基础知识、增强实际应用能力

教材

■ 数值分析与算法(第3版)

- 喻文健 编著
- 清华大学出版社, 2020年
- 课件上的补充内容

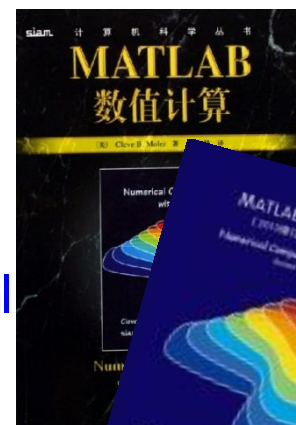


■ 参考书

- 《Matlab数值计算》，机械工业出版社, 或北航出版社

https://www.mathworks.com/moler/index_ncm.html

- 李庆扬 等, 数值分析(第5版), 清华大学出版社, 2009年



教学方式与考核

■ 方式

- 课堂讲授、作业、编程实验
- 网络学堂：课后提供课件、作业，其他资料
(作业、实验报告交到网络学堂上)

■ 考评方法

- 平时：雨课堂/作业/编程实验(5%+10%+15%)
只取10次
- 期中：考试(30%)
- 期末：考试/调研报告(40%) 再转换为等级制
- 总成绩= $\max\{\text{平时}+\text{期中}+\text{期末}, \text{平时}+\text{期末}\}$

若无法举行期中考试, 平时/期末成绩占比重新调整

主要教学内容

■ 一.数值计算导论

- 概述、误差分析基础(近似、有效数字、误差分类、问题的敏感性、算法的稳定性)、浮点算术系统与舍入误差(“抵消”现象、减小舍入误差的建议等)

■ 二.非线性方程解法

■ 三.线性方程组的直接解法

■ 四.线性方程组的迭代解法

■ 五.矩阵特征值计算

主要教学内容

- 六-1.函数逼近与线性最小二乘
- 六-2.函数插值
- 七.数值积分与微分
- 八.常微分方程初值问题
- 附加.**Matlab**数值计算与应用
 - 补充内容、非考试要求
 - 穿插在各章内容中

日程安排Syllabus

类似数学基础课：公式多、推导多、还有理论证明

又像工程专业课：要上机编程、实验，有时还需要点经验

学习建议

■ 往届学生的情况 (2020年春)

- 283→247人中, 约1/3获A-及以上
- 12人不及格(含1人缺考)
- 考试失败的典型情况

■ 措施和建议

- 认真对待, 把握听课、作业、实验环节
- 重点理解问题背景、算法思路和具体步骤(会算)
- 适当进行公式推导、算法复杂度分析与比较
- 多用Matlab软件做实验, 提升学习兴趣!
- ~ 欢迎对教学和考核方式提出合理建议



数值计算的背景与概况

数值分析、科学计算、数值计算

数值计算(数值仿真), 也称为科学计算, 已成为当今科学研究的**三种基本手段**之一。它是计算数学、计算机科学和其他工程学科相结合的产物, 并随着计算机的普及和各类科学技术的迅速发展日益受到人们的重视。

——参考书译者序

科学计算的发展涉及硬件和软件两个方面, 这里我们只考虑软件方面, 即**数值计算的有关算法 (数值仿真软件)**

“数值分析”、“数值计算”是研究求解连续数学问题的**算法**的学科 (而不仅仅局限于计算误差的研究)

核心

对象

数值算法与非数值算法

..... *The art of computer programming* 系列

We might call the subject of these books “nonnumerical analysis.” Computers have traditionally been associated with the solution of numerical problems such as Numerical computer programming is an extremely interesting and rapidly expanding field, and many books have been written about it.

From D. E. Knuth, *The art of computer programming*, Vol. 1 (《计算机程序设计艺术》)

- 算法分为“数值算法”和“非数值算法”
- 数值算法用途广泛，发展迅速，具有跨学科的特点
- “非数值算法”的研究则通常归于“计算机科学”

Top ten algorithms of the century

*“We tried to assemble the 10 algorithms with the **greatest influence** on the **development** and **practice** of science and engineering in the 20th century”*

—— Editors of *IEEE Computing in Science & Engineering*,
Jan. 2000 (后被SIAM转载)

- **1.1946** 美国Los Alamos国家实验室的**J. von Neumann, S. Ulam**和**N. Metropolis**发展的**Metropolis**算法（属于Monte Carlo方法；拒绝采样/MCMC, 生成给定概率分布随机点）
- **2.1947** 美国RAND公司的**G. Dantzig**提出的解线性规划的单纯形算法（**simplex method**）
- **3.1950** 美国UCLA大学与美国国家标准局数值分析所的**M. Hestenes, E. Stiefel**和**C. Lanczos**开创的**Krylov**子空间迭代法(**CG**算法、**Lanczos**过程)
- **4.1950's** 矩阵分解方法，由美国Oak Ridge国家实验室的**A. Householder**引入数值线性代数中(矩阵计算研究掀起革命)

Top ten algorithms of the century

- 5. **1957** 美国IBM的**J. Backus**领导开发的**Fortran**最优编译器
→ **Newton/quasi-Newton法**
- 6. **1959-61** 英国Ferranti Ltd.的**John G.F. Francis**发明**QR**算法，能稳定地计算矩阵的特征值
(QZ, SVD)
- 7. **1962** 英国Elliot Brothers, Ltd.的**Tony Hoare**提出快速排序算法（**Quicksort**）
→ **Page rank**
- 8. **1965** 美国IBM Watson研究中心的**J. Cooley**与普林斯顿大学及AT&T Bell实验室的**J. Turkey**共同提出了**FFT**算法
- 9. **1977** 美国Brigham Young大学的**H. Ferguson**和**R. Forcède**提出的整数关系侦察算法(实验数学/简化量子场理论计算)
→ **JPEG**
- 10. **1987** 美国Yale大学的**L. Greengard**和**V. Rokhlin**发明的快速多极算法(**fast multipole algorithm**)
→ **Kalman filter**

大多属于/涉及数值计算!

N. Higham, *The Princeton Companion to Applied Mathematics*, Princeton University Press, 2015

数值计算与数值算法

■ 数值计算的特点

- 处理连续数学的量(实数量), 问题中还可能涉及微分、积分和非线性。被求解的问题**一般**没有解析解、或理论上无法通过有限步计算求解
 - 无解析解: $33x^5 + 3x^4 - 17x^3 + 2x^2 + 4x - 39 = 0$
 - 有解析解, 但需无限步计算: $\sin(x)$
 - 更多的实际应用问题通过数值仿真(simulation)来解决
- **目标**: 寻找迅速完成的(迭代)算法, 评估结果的准确度

■ 好数值算法的特点

- 计算效率高、计算复杂度低
- 可靠性好: 在考虑**实际计算**的各种误差情况下, 结果尽可能地准确

数值计算的步骤

- 建立数学模型（需要相关学科背景）
 - 研究数值计算、求解方程的算法
 - 通过计算机语言编程实现算法
- } 本课程学习重点

- 在计算机上运行程序进行数值实验、仿真
- 将计算结果用较直观的方式输出，如图形可视化方法
- 解释和验证计算结果，如果需要重复上面的某些步骤

上述各步骤相互间紧密地关联，影响着最终的计算结果和效率（问题的实际背景和要求也左右着方法的选择）

- 设计数值方法(算法)的关键：将问题简化或加以近似(估计带来的误差)，然后求解简化后的问题

数值软件/程序包

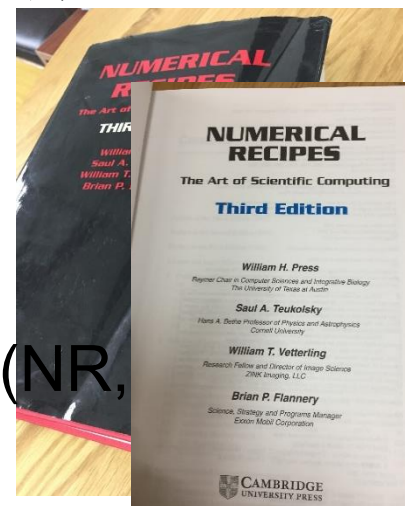
■ 数值计算的软件与程序包

- 解决常见问题，促进各个科学和工程领域的科研
- 了解基本原理，学习算法设计和实现技巧
- 成为聪明的软件/程序包使用者

■ 存在形式和资源

- 互联网(github, ...), 免费(TOMS)/商业代码(NR,
- Fortran, C, C++, Matlab, Python, Julia, ...
- 源代码, 或API调用, 或集成开发环境 www.mathworks.com

<https://blogs.mathworks.com/cleve/>



Cleve's Corner: Cleve Moler on Mathematics and Computing

Scientific computing, math & more

数值软件/程序包

2016 Dense Linear Algebra
Software Packages Survey (J. Dongarra)

<http://www.netlib.org/lapack/lawnspdf/lawn290.pdf>

■ 矩阵计算有关程序包

□ LAPACK, www.netlib.org/lapack/index.html 87%

“as much as possible computation is performed by calls to BLAS” “exploit Level 3 BLAS”

□ LAPACK: C language APIs for LAPACK

□ BLAS: www.netlib.org/blas/ (ATLAS, GotoBLAS, OpenBLAS*)

□ ScaLAPACK: Distributed-memory variant 40%

□ MAGMA: a computer algebra system 22%

■ 更高层的程序库与环境

□ Intel公司Math Kernel Library (in C) No. 1

□ Eigen (a C++ template library): eigen.tuxfamily.org No. 2

□ Matlab, Octave, Python (Numpy), ... (有20%被调查者要自己写线性代数子程序)
Linux: 95%, Mac: 32%, Win: 22%

■ 数值计算相关研究的好帮手：Matlab软件

- 集成环境：交互式计算系统，高级编程语言

	Matlab(作为编程语言)	C, C++, Fortran
	第四代编程语言	第三代编程语言
编译方式	解释器，或JIT加速器(v 6.5后)	编译器
申明变量？	不需要	需要
开发时间	较快	较慢
运行时间	较慢	较快
开发环境	集成环境(编辑器、调试器、命令历史、变量空间、profiler、编译器)	--

- 数值计算、矩阵计算功能强大(包含很多先进算法)
- 大量专题工具箱(**Toolbox**)，为专业应用提供便利
- 开发环境、可视化功能等方面比**Python**强

- 数值计算知识应用广泛 (以计算机相关方向为例)
 - 人工智能、机器人控制：矩阵特征值、奇异值分解、常微分方程数值解、数据拟合(回归)
 - 计算机图形学**CAD**：函数插值、逼近、微分方程数值解
 - 集成电路**CAD (EDA)**：大规模线性方程组求解、常微分方程、偏微分方程
 - 系统软件、编译、网络等方向：线性方程组求解、非线性方程组求解
 - 高性能计算：性能评测、算法实现与优化、更多应用
 - 电力系统仿真、大气仿真，。。。。。
- 广义的数值计算还包括了“数学规划/最优化”，也是大数据分析、机器学习中复杂算法的基础



误差分析基础

误差分析基础

- § 1.2.1 误差的来源
- § 1.2.2 误差及其分类
 - 误差与有效数字
 - 数据传递误差与计算误差
 - 截断误差与舍入误差
- § 1.2.3 问题的敏感性与数据传递误差
- § 1.2.4 算法的稳定性

误差的来源

计算前 {
■ 模型误差
■ 数据误差

(忽略摩擦、空气阻力)

常数或测量值、前一步计算的结果

计算中 {
■ 截断误差
■ 舍入误差

方法误差 例: $\sin(x) = \dots$

计算时表示数的位数有限

例1.1 用球表面积公式计算地球表面积 “四舍五入” (默认)

$$A = 4\pi r^2$$

➤ 将地球近似成球体

➤ 取半径 $r \approx 6370km$

➤ 将 π 的值取到有限位 (如3.14)

➤ 计算 $4\pi r^2$ (计算乘法)

模型误差

数据误差

数据误差

舍入误差

误差及其分类

■ 1.误差与有效数字

■ **定义1.1** x ~准确值, \hat{x} ~近似值, 绝对误差 $e(\hat{x}) = \hat{x} - x$

□ (绝对)误差往往不能反映误差严重程度

□ **例:** 方法一测量长约1公里的物体, 误差1cm; 方法二测量长约1米的物体, 误差也是1cm

■ **定义1.2** 相对误差 $e_r(\hat{x}) = \frac{\hat{x} - x}{x}$

□ 无论误差、相对误差, 都可正可负

□ 准确值为0, 相对误差无意义

□ 准确值未知, 估计误差上限, 误差限 $\varepsilon(\hat{x})$, $\varepsilon_r(\hat{x})$

□ 误差较小时, $e_r(\hat{x}) \approx \frac{\hat{x} - x}{\hat{x}}$

绝对误差限→相对误差限

误差及其分类

- **定义1.3** 一个数的有效数字指: 从左至右第一个非零数字开始的所有数字

□ **前几位有效数字正确**与相对误差有何关系?

- **定理1.1** 设 \hat{x} 是 x 的近似值, 若 \hat{x} 的前 p 位有效数字正确, $p \geq 1$, 则相对误差 $|e_r(\hat{x})| < \frac{1}{d_0} \times 10^{-p+1}$ d_0 为 x 的第一位有效数字

□ **证明:** 设 $\hat{x} = \pm 10^m \times (d_0 + \frac{d_1}{10} + \cdots + \frac{d_{p-1}}{10^{p-1}} + \cdots)$

由于前 p 位正确, $|\hat{x} - x| < 10^m \times \frac{1}{10^{p-1}}$

而 $|x| \geq 10^m \times d_0 \Rightarrow |e_r(\hat{x})| = \frac{|\hat{x} - x|}{|x|} < \frac{1}{d_0} \times 10^{-p+1}$

误差及其分类

(是否经过“四舍五入”?)

- 易与“ \hat{x} 前 p 位有效数字正确”混淆的一种表达:
对某数 x “保留 p 位有效数字”后得到近似值 \hat{x}
- 后者涉及最后一个数位的四舍五入, 对应近似值的误差限是定理1.1结论的一半 (定理1.2) $\sim \frac{1}{2d_0} \times 10^{-p+1}$

- 例1.2 $x = \pi = 3.14159265 \dots$, 保留3位有效数字
 $\hat{x} = 3.14$, $|e(\hat{x})| \leq \frac{1}{2} \times 10^{-3+1}$, $|e_r(\hat{x})| \leq \frac{1}{2} \times \frac{1}{3} \times 10^{-3+1}$

做作业题时应注意区分上述两种表达方式

以上是根据正确的有效数字位数判断相对误差限

反过来呢?

误差及其分类

- **定理1.3** 设 x 的第一位有效数字为 d_0 , 若近似值 \hat{x} 的相对误差满足: $|e_r(\hat{x})| \leq \frac{1}{2(d_0+1)} \times 10^{-p+1}$, 则 \hat{x} 的前 p 位有效数字正确, 或保留 p 位有效数字后与 x 相同

□ **证明:** 设 $x = \pm 10^m \times (d_0 + \frac{d_1}{10} + \dots + \frac{d_{p-1}}{10^{p-1}} + \frac{d_p}{10^p} + \dots)$

$$\Rightarrow |x| < 10^m (d_0 + 1) \Rightarrow |e(\hat{x})| = |x| |e_r(\hat{x})| < 10^m \times \frac{1}{2} \times \frac{1}{10^{p-1}}$$

\hat{x} 与 x 差的首位有效数字在 d_p 所在的数位上, 值 $\in (-5, 5)$

若两个数的第 p 位有效数字均为 d_{p-1} , 则...

否则, 两者第 p 位为相邻数字 $\Rightarrow \hat{x}$ 与 x 保留 p 位后相同

- **例1.3** $x = 9.9993$, $\hat{x}_1 = 9.9997$, $\hat{x}_2 = 9.9989$. 相对误差均满足 $|e_r(\hat{x})| \leq \frac{1}{2(9+1)} \times 10^{-4+1}$. 但 \hat{x}_1 和 \hat{x}_2 情况不同

误差及其分类

$$\frac{1}{2} \times 2^{-p}$$

思考：对2进制数呢？

- **定理1.3** 设 x 的第一位有效数字为 d_0 , 若近似值 \hat{x} 的相对误差满足: $|e_r(\hat{x})| \leq \frac{1}{2(d_0+1)} \times 10^{-p+1}$, 则 \hat{x} 前 p 位有效数字正确, 或保留 p 位有效数字后与 x 相同
- **说明**: 1. 可认为“保留 p 位有效数字正确”的两种含义
2. 推论: 若相对误差 $\leq \frac{1}{2} \times 10^{-p}$, 则保留 p 位有效数字正确(定理1.4). 相对误差 $<10^{-p}$ 意味保留 p 位有效数字正确
- 区分两种说法
 - 精度: 与表示数的有效数字位数有关 (单/双精度)
 - 准确度: 与误差大小有关
- 3.111111有7位十进制精度, 但它近似 π 准确度并不高

误差及其分类

■ 2.数据传递误差与计算误差

■ 以简单的函数求值问题为例

□ $x \rightarrow f(x), \hat{x} \rightarrow \hat{f}(\hat{x})$

□ 误差 $\hat{f}(\hat{x}) - f(x) = \underbrace{[\hat{f}(\hat{x}) - f(\hat{x})]}_{\text{单纯的计算误差}} + \underbrace{[f(\hat{x}) - f(x)]}_{\text{数据误差传递到结果}}$

单纯的计算误差 数据误差传递到结果
称之为: 数据传递误差

- 说明: 这里定义的数据传递误差不考虑具体的计算方法, 分析它时考虑精确计算过程, 它受问题本身影响

误差及其分类

■ 3. 截断误差与舍入误差

- 数值方法近似、有限精度运算 (计算误差的两部分)

■ 例1.4 用差商近似一阶导数 $f'(x) \approx \frac{f(x+h) - f(x)}{h}$

- h 为步长, 分析两种误差与 h 关系

- 截断误差 $e_T = hf''(\xi)/2$

- $\varepsilon_T = Mh/2$, M 是 $|f''(\xi)|$ 上界

- 设计算 $f(x)$ 误差限为 ϵ , 则

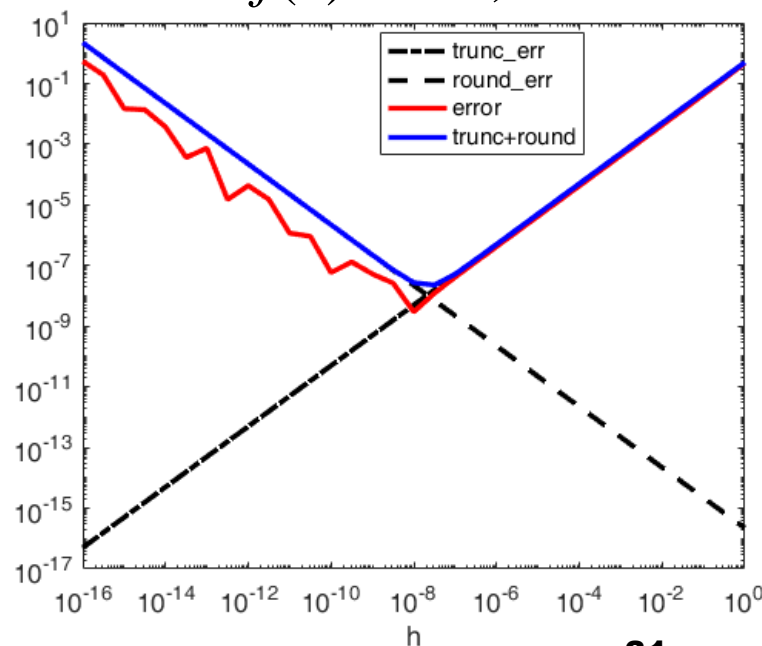
$$\varepsilon_R = 2\epsilon/h \longrightarrow \varepsilon_{tot} = \frac{Mh}{2} + \frac{2\epsilon}{h}$$

- 实验: 看 h 取何值使 ε_{tot} 最小

- $\epsilon \approx 10^{-16}$, 最佳 $h = 2 \times 10^{-8}$

$$\hat{f}(\hat{x}) - f(\hat{x})$$

$$f(x) = \sin x, x=1$$



问题的敏感性 (数据传递误差)

- **定义1.8** 问题的敏感性: 输入数据扰动对问题解的影响程度. 不敏感(良态), 敏感(病态)
- **定义1.9** 用(相对)**条件数**反映问题的敏感性

$$\text{cond} = \frac{\| \text{问题的解的相对变化量} \|}{\| \text{输入数据的相对变化量} \|} \rightarrow \text{范数}$$

即问题对数据误差的“放大因子”. cond越大问题越病态

□ 例如函数求值问题: $x \rightarrow f(x)$, $\hat{x} \rightarrow f(\hat{x})$

□ 结果相对误差 $\frac{f(\hat{x})-f(x)}{f(x)}$, 数据相对变化 $\frac{\hat{x}-x}{x}$

$$\Rightarrow \text{cond} = \left| \frac{[f(\hat{x})-f(x)]/f(x)}{(\hat{x}-x)/x} \right| \approx \left| \frac{xf'(x)}{f(x)} \right| \quad (\text{近似公式})$$

问题的敏感性 (数据传递误差)

- 类似于(相对)条件数, 也可定义绝对条件数
- 例如对函数求值问题, 绝对条件数 $\text{cond}_A = \left| \frac{f(\hat{x}) - f(x)}{\hat{x} - x} \right|$
- **说明:** 1. 条件数反映问题的特性, 与计算方法无关. 但它也随数据变化而变化, 因而常考虑其上限
2. 对简单运算, 可利用微积分估计数据传递误差限

例: $\hat{y} = f(\hat{x}_1, \hat{x}_2, \dots, \hat{x}_n)$, $y = f(x_1, x_2, \dots, x_n)$

多元Taylor展开取线性项, $y - \hat{y} \approx \sum_{i=1}^n \frac{\partial f}{\partial x_i}(\hat{x}_1, \dots, \hat{x}_n)(x_i - \hat{x}_i)$

$$\Rightarrow \varepsilon(\hat{y}) = \sum_{i=1}^n \left| \frac{\partial f}{\partial x_i}(\hat{x}_1, \dots, \hat{x}_n) \right| \varepsilon(\hat{x}_i)$$

应用于+, -, ×, /运算, 见课本(1.5)式, 如 $\varepsilon(\hat{x}_1 \hat{x}_2) = |\hat{x}_2| \varepsilon_1 + |\hat{x}_1| \varepsilon_2$

算法的稳定性

- 与问题的敏感性相对应的一个概念；也叫**数值稳定性**
- ¹结果对计算过程中的扰动(精度)不敏感的算法更稳定
- **例1.7** 对长度100的数组求和, 每个数只有**2位**数字精度
- **算法1**: 按存储顺序对这100个数直接累加 sum=1.0
- 若实际数据为1.0, 0.01, ..., 0.01 (**99个**), 则结果?
- **算法2**: 先按元素绝对值递增的顺序排序, 再依次求和
- 对上述数据取值, $\text{sum}=0.99+1.0=2.0$, 更准确!

算法2比算法1更稳定!

- ²对包含一系列计算的过程, 若计算中的小扰动不放大或放大不严重, 则该过程对应的算法更稳定

注意：一般指相对误差!

算法的稳定性

- **例1.8** 依次计算黄金分割比例 $\phi = \frac{\sqrt{5}-1}{2}$ 的前几个幂
- **算法1**: 直接乘法, $f(x) = x^n$, $x=0.618034$ (ϕ 的近似值)
- **算法2**: 利用递推式
$$\begin{cases} \phi^{n+1} = \phi^{n-1} - \phi^n & \text{每步仅做} \\ \phi^0 = 1, \phi^1 = x & \text{一次减法} \end{cases}$$
- **算法2** 的效果

n	ϕ^n 的计算值
2	0.381966
3	0.236068
...	...
18	0.000144
19	0.000154
20	-0.000010

□ 分析误差传播趋势 (算 ϕ^n 误差为 e_n)

$$\begin{cases} e_{n+1} = e_{n-1} - e_n \\ e_0 = 0, e_1 = x - \phi \end{cases}$$

$$e_2 = -e_1, e_3 = e_1 - e_2 = 2e_1, e_4 = -3e_1,$$

$$e_5 = 5e_1, \dots, |e_{20}| = c|e_1|, c \text{ 是?}$$

错误! $|e_r(\hat{\phi}^n)| > 100\%$

算法的稳定性

- **例1.8** 依次计算黄金分割比例 $\phi = \frac{\sqrt{5}-1}{2}$ 的前几个幂
- **算法1**: 直接乘法, $f(x) = x^n$, $x=0.618034$ (ϕ 的近似值)

- **算法2**: 利用递推式 $\begin{cases} \phi^{n+1} = \phi^{n-1} - \phi^n & \text{每步仅做} \\ \phi^0 = 1, \phi^1 = x & \text{一次减法} \end{cases}$

n	ϕ^n 的计算值
2	0.381966
3	0.236068
...	...
18	0.000144
19	0.000154
20	-0.000010

$|e_{20}| = c|e_1|$, c 是 Fibonacci 序列的第 20 项
 $\sim 6.7 \times 10^3$

□ 反过来看算法1

由于 $x < 1$, $n \nearrow$, 误差 $\searrow |e_{20}| \approx 20\phi^{19}|e_1| \sim 10^{-3}$

□ 上面未分析舍入误差(很小可忽略)

$$e_{n+1} = e_{n-1} - e_n$$

思考: 从条件数角度分析算法2不稳定

算法的稳定性

- 有时还是要考虑舍入误差的。但一般算法含很多步计算, 从输入量开始“向前”舍入误差分析很难
- “向后误差分析”是另一种思路
 - 以函数求值为例
$$y = f(x), \text{ 计算结果为 } \hat{y} = \hat{f}(x)$$
 - 求 \hat{x} 使其满足 $f(\hat{x}) = \hat{y}$, 则 $\Delta x = \hat{x} - x$ 称为**向后误差**, 其大小反映算法过程的稳定性
- 对一些问题, 可基于“向后误差分析”判断算法稳定性, 例如对于求解线性方程组的高斯消去法 (后面第3章会提到)



计算机浮点数系统

计算机浮点数系统与舍入误差

- 浮点数的表示
- 机器精度 (ϵ_{mach})
- 抵消现象 (cancellation)

(课本1.3节的主要内容)

计算机中的浮点数

- 用浮点数表示实数，浮点数 x 为 (**2进制**)

$$x = \pm \left(d_0 + \frac{d_1}{2} + \frac{d_2}{2^2} + \cdots + \frac{d_{p-1}}{2^{p-1}} \right) \times 2^E$$

- **基数**: 2进制 ; **指数** E : 上限值 U , 下限值 L
- p 位**尾数**, 也称 p 为精度(precision) p位有效数字
- IEEE浮点数系统已成为标准, 分**单精度**和**双精度**
- 在标准中采用**规范化**技术, 要求 $d_0 = 1$
- **好处**: 数的表示唯一、尾数都是有效数字、 d_0 不用存储 (该位表示 \pm 的信息)

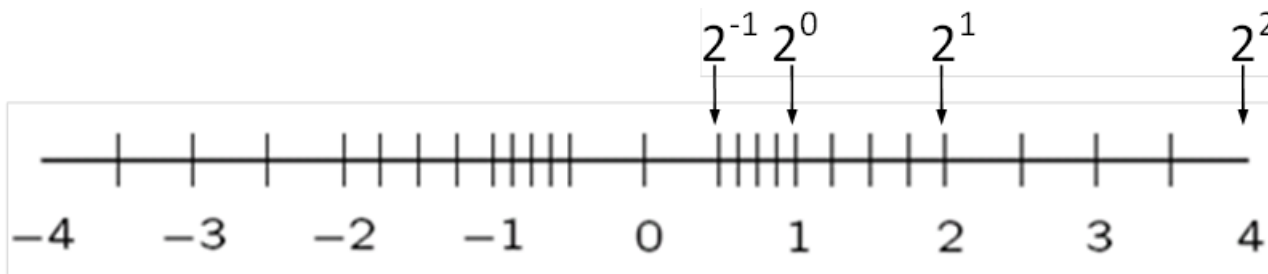
计算机中的浮点数

IEEE双精度数的表示:

$$x = [b_1 b_2 b_3 \cdots b_{12} b_{13} \cdots b_{64}]_2$$

$$E+1023$$

- 浮点数为有限个，且**非均匀**地分布在实数轴上



例: 一个简单浮点数系统, $p=3, L=-1, U=1$ (5位二进制)

- 机器精度 $\epsilon_{\text{mach}} = 2^{-p}$ (1与右边相邻数间隔的一半)

- 下溢值: 2^L $\sim 2.2 \times 10^{-308}$ 上溢值: $(2 - 2^{-p+1}) \times 2^U$

	浮点数系统	β	p	L	U	ϵ_{mach}
32 bits	IEEE单精度	2	24	-126	127	5.960×10^{-8}
64 bits	IEEE双精度	2	53	-1022	1023	1.110×10^{-16}

+*/运算, 以及简单函数的误差
与此同级别(sin, tan, atan, exp)

--Moler's blog, 2017.1

计算机中的浮点数

- **定理1.5**: 用浮点数 $\text{fl}(x)$ 近似表示实数 x , 其相对误差 $\left| \frac{\text{fl}(x) - x}{x} \right| \leq \varepsilon_{\text{mach}}$ 应用定理1.2可证明!, 2进制

- 注: 这里都考虑“四舍五入” (最近舍入)

- **定理1.6**: $x_1, x_2 \in \mathbb{R}$, 若 $\left| \frac{x_2}{x_1} \right| \leq \frac{1}{2} \varepsilon_{\text{mach}}$, 则 x_2 的值对浮点运算 $x_1 + x_2$ 的结果毫无影响

类似定理1.4证明!

- 注: 这就是“大数吃掉小数”现象

绝对差 $< 2^E \varepsilon_{\text{mach}}$, 所以 $x_1 + x_2 = x_1$. 若 $\left| \frac{x_2}{x_1} \right| > \varepsilon_{\text{mach}}$, 一定不“吃小数”



十进制系统, 这两个定理如何?

$$\left| \frac{\text{fl}(x) - x}{x} \right| \leq \frac{\varepsilon_{\text{mach}}}{d_0} = \frac{1}{2d_0} 10^{-p+1}$$

若 $\left| \frac{x_2}{x_1} \right| \leq \frac{1}{2} 10^{-p} = \frac{1}{10} \varepsilon_{\text{mach}}$, 一定会“吃小数”, 若 $> \varepsilon_{\text{mach}}$ 一定不“吃小数”

抵消现象

- 两个符号相同、值相近的p位数相减使结果的有效数字远少于p位. 称之为**抵消**(cancellation)
- **例：** $x = 1.92305 \times 10^3$, $y = 1.92137 \times 10^3$, 则 $x - y = 1.68$
- 减法计算未发生舍入, 但其结果**仅有三位**有效数字
- 结果的有效数字位数的减少, 意味着相对误差的放大, 往往会影响后续计算的准确度 **(操作数有误差!)**
- 抵消现象是发生信息丢失、误差变大的信号！

抵消现象

■ 一元二次方程求根公式的例子

$$ax^2 + bx + c = 0$$

解为:

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

若 $b > 0$, 且 $|4ac| \ll b^2$, 计算 x_1 时出现抵消现象

解决办法: (算法的调整)

$$x_1 = \frac{-b + \sqrt{b^2 - 4ac}}{2a} \cdot \frac{-b - \sqrt{b^2 - 4ac}}{-b - \sqrt{b^2 - 4ac}} = \frac{2c}{-b - \sqrt{b^2 - 4ac}}$$

计算 x_2 可能出现的问题也类似地解决



保证计算结果的准确性

减小舍入误差的几条建议/认识

- 采用双精度浮点数
- 对包含大量计算的算法，分析舍入误差很难
- 应遵循如下几条建议

- 避免中间计算结果出现上(下)溢出

例：计算 $\frac{x_1}{x_2 \cdot x_3 \cdots x_n}$ ，若 $x_2 \ll x_1, \dots$

- 避免“大数吃掉小数” (加、减法)

例：计算 $1 + \varepsilon + \varepsilon$ ， $\varepsilon \approx 0.6 \times 10^{-16}$

(调整计算顺序)

- 避免符号相同的两相近数相减

- 注意简化步骤，减少运算次数

总结

	总误差		
	计算误差		数据 传递误差
	截断误差	舍入误差	
如何评估大小?	根据不同问题和方法进行讨论	向后误差分析; 区间分析法; 很难定量分析	问题敏感性(条件数); 直接近似分析
如何减小误差?	计算方法的选择	选稳定的算法; 减小舍入误差的建议; 采用更高精度浮点数	变换问题形式(计算过程), 改善敏感性

(更多例子和讨论, 自学课本1.4节)

演示程序与Matlab

■ 课程演示网站

<http://heath.cs.illinois.edu/iem/>,
有**stand-alone**执行版下载

- <http://numbda.cs.tsinghua.edu.cn/~yuwj/numweb/index.html>
(IE浏览器”安全设置”允许**Java applet**程序!)
(本机**Java**控制面板”安全”-”例外站点”添加!)

■ Matlab的简单演示

- 简单的操作说明
- 浮点数系统有关的参数
- 例1.4: 截断误差与舍入误差的实验

■ 课程的编程实验

- 共**7**次实验, 选做**4**次, 交报告并检查验收