大学数学实验

Mathematical Experiments

小结：科学计算中的基本概念

清 华 大 学 数 学 科 学 系　谢金星
办公室：理科楼A202　电话：62787812
E-mail: xiejx@tsinghua.edu.cn
http: //www.mcm.edu.cn/jxie

A Joke

$$\lim_{n\to\infty} \frac{\sin x}{n} = ?$$

$$\lim_{n\to\infty} \frac{\sin x}{n} = six = 6$$

Another Joke

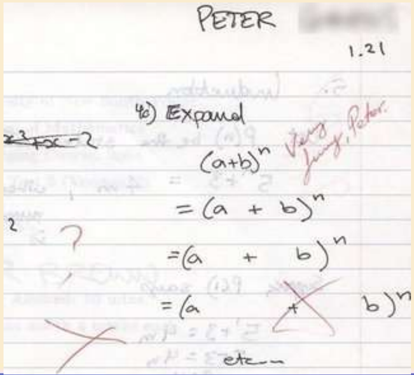
After explaining to a student through various lessons and examples that:

$$\lim_{x\to 8} \frac{1}{x-8} = \infty$$

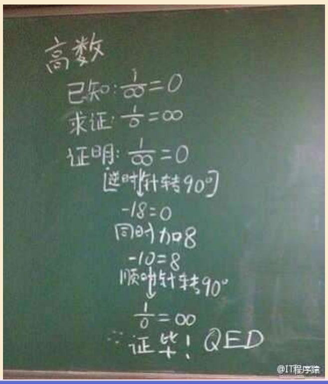
I tried to check if she really understood that, so I gave her a different example. This was the result:

$$\lim_{x\to 5} \frac{1}{x-5} = \text{ω}$$

Another Joke



Another Joke



科学计算研究的核心问题

算法的构造
算法的分析

了解、理解、选择、使用
分析、改进、创新、创造

构造算法的基本手段: 近似

研究算法的核心问题: 近似对计算结果的影响

科学计算中的基本概念

- 收敛性 (or 复杂度)
  -----误差估计和分析
  -----收敛速度
- 病态性
- 稳定性

研究的出发点: 误差（error） !!

误差的基本类型

- 计算地球的表面积: $A=4\pi r^2$
  - 模型误差: 地球被看成是一个球
    - 地球的简单理想模型
  - 观测误差（测量/数据误差）: 测量仪器误差
    - 地球的半径要经过测量和计算得到
  - 截断误差（方法误差）: π是无理数, 取为3.1415
    - 更典型的如：Taylor级数展开，取前有限项
  - 舍入误差（计算误差）: 浮点数的计算
    - π在计算中按“四舍五入”取为3.1416
      (计算机中的二进制数也无法精确表示3.1416)

近似数与误差

真值（精确数）x	1/3
近似值（近似数） x^*	0.33
绝对误差 $e = x - x^*$ (或 $x^* - x$)	0.0033…
相对误差 $e_r = e / x$	0.0099…
$\approx e / x^*$ (实际常用)	0.0101…
绝对误差限(上界)ε: $\lvert e\rvert \le \varepsilon$	0.0034, 0.004, …
相对误差限(上界)ε_r: $\lvert e_r\rvert \le \varepsilon_r$	0.01, 0.011, …

单选题 2分 设置

如果分别用 3.1415 和 3.1416 作为圆周率 π 的近似值，各（至多）有几位有效数字?

A 5位，5位
B 4位，4位
C 4位，5位
D 3位，5位
E 3位，4位
F 以上都不对

提交

误差与有效数字

观察：四舍五入得到的最后一位数字，是有效数字

● 若近似值x^*满足 $\boxed{|x-x^*|\le \frac{1}{2}\times 10^{-n}}$，则称$x^*$准确到小数点后第$n$位. 并把从第一个非零数字到这一位的所有数字均称为有效数字.

例: $\pi = 3.1415926535897932......;\quad \pi^* = 3.1415$

问: π^*有几位有效数字? (一般指“至多”几位)

$\because |\pi^*-\pi| < 0.5\times 10^{-3};\quad |\pi^*-\pi| > 0.5\times 10^{-4}$

$\therefore \pi^*$ 有4位有效数字，精确到小数点后第3位

误差与有效数字

● 有效数字的另一等价定义

数x^*总可以写成如下形式

$$x^* = \pm 0.a_1a_2\cdots a_n\cdots \times 10^m.$$

其中m是整数, a_i是0到9中的一个数字, $a_1 \ne 0$.

x^* 作为x的近似值, 具有n位有效数字当且仅当

$$\boxed{|x^*-x| \le \frac{1}{2}\times 10^{m-n}}$$

由此可见, 近似值的有效数字越多, 其绝对误差越小.

误差与有效数字

例 为了使 $x=\sqrt{2}$ 的近似值的绝对误差不大于10^{-5}，问应取几位有效数字？ (一般指“至少”取几位)

解 由于$\sqrt{2}=1.414\cdots$，则近似值x^*可写为

$$x^*=0.a_1a_2\cdots a_n\times 10^1,\quad a_1=1\neq 0.$$

令 $\left|\sqrt{2}-x^*\right|\le \frac{1}{2}\times 10^{1-n}\le 10^{-5}$

故取 n=6, 即取 6 位有效数字. 此时 x^*=1.41421.

有效数字与相对误差限

☞ 有效数字 ⇒ 相对误差限

已知 $x^*=\pm 0.a_1a_2\ldots a_n\times 10^m$有 n 位有效数字, 则其相对误差限为

$$\varepsilon_r=\left|\frac{\varepsilon}{x^*}\right|=\frac{0.5\times 10^{m-n}}{0.a_1a_2\ldots a_n\times 10^m}=\frac{10^{-n}}{2\times 0.a_1\ldots}\le\frac{1}{2a_1}\times 10^{-n+1}$$

与绝对误差（限）不同，
相对误差（限）与 m 无关！

有效数字与相对误差限

☞ 相对误差限 ⇒ 有效数字

已知 x^* 的相对误差限可写为 $\varepsilon_r\le\frac{1}{2(a_1+1)}\times 10^{-n+1}$

则

$$|x-x^*|\le\varepsilon_r\cdot|x^*|\le\frac{10^{-n+1}}{2(a_1+1)}\times 0.a_1a_2\ldots\times 10^m$$

$$<\frac{10^{-n+1}}{2(a_1+1)}\cdot(a_1+1)\times 10^{m-1}=0.5\times 10^{m-n}$$

可见 x^* 至少有 n 位有效数字.

单选题 2分 设置

(a) 用1.414*2.236 作为 sqrt (10) 的近似值，结果有几位有效数字?
(b) 近似数的计算 1.2 – 1.1 = 0.1 的结果，有几位有效数字?

A (a) 4位，(b) 1位
B (a) 4位，(b) 0位
C (a) 3位，(b) 1位
D (a) 3位，(b) 0位
E 以上都不对

提交

误差传播

如 $|\sqrt{2}-1.414|\le 0.5\times 10^{-3}$,

$|\sqrt{5}-2.236|\le 0.5\times 10^{-3}$,

问 $|\sqrt{5}\times\sqrt{2}-2.236\times 1.414|\le ?$

$\left|\frac{\sqrt{2}}{\sqrt{5}}-\frac{1.414}{2.236}\right|\le ?$

a^* = 1.414 * 2.236
= 3.161704

a = 10 ^ 0.5
= 3.16227766…

$a-a^*$ = 5.7366…e-04
> 0.5 e-03
(<0.5 e-02)

a^*只有三位有效数字

有没有一般规律？

误差传播

计算 $A=f(x_1,x_2)$. 如果x_1,x_2的近似值为 x_1^*,x_2^*, 则A的近似值为 $A^*=f(x_1^*,x_2^*)$, 用多元函数微分近似公式可以得到

$$e(A^*)=A-A^*=f(x_1,x_2)-f(x_1^*,x_2^*)$$

$$\approx\frac{\partial f(x_1^*,x_2^*)}{\partial x_1}(x_1-x_1^*)+\frac{\partial f(x_1^*,x_2^*)}{\partial x_2}(x_2-x_2^*)$$

$$=\frac{\partial f(x_1^*,x_2^*)}{\partial x_1}e(x_1^*)+\frac{\partial f(x_1^*,x_2^*)}{\partial x_2}e(x_2^*)$$

绝对误差 e 可近似“类比”微分运算！

误差传播

$$e(A^*)=\frac{\partial f(x_1^*,x_2^*)}{\partial x_1}e(x_1^*)+\frac{\partial f(x_1^*,x_2^*)}{\partial x_2}e(x_2^*)$$

$$e_r(A^*)\approx\frac{\partial f(x_1^*,x_2^*)}{\partial x_1}\frac{x_1^*}{f(x_1^*,x_2^*)}e_r(x_1^*)+\frac{\partial f(x_1^*,x_2^*)}{\partial x_2}\frac{x_2^*}{f(x_1^*,x_2^*)}e_r(x_2^*)$$

条件数： $\boxed{\frac{\partial f(x_1,x_2,\dots,x_n)}{\partial x_i}}$ $\boxed{\frac{x_i}{f(x_1,x_2,\dots,x_n)}\frac{\partial f(x_1,\dots,x_n)}{\partial x_i}}$

条件数大的问题，是病态问题

误差传播

四则运算

$$|e(x_1\pm x_2)|\le|e(x_1)|+|e(x_2)|$$

$$|e_r(x_1x_2)|\le|e_r(x_1)|+|e_r(x_2)|$$

$$\left|e_r\left(\frac{x_1}{x_2}\right)\right|\le|e_r(x_1)|+|e_r(x_2)|$$

误差传播

例 设 $y=x^n$，求 y 的相对误差与 x 的相对误差之间的关系.

解 $e(y)=e(x^n)=nx^{n-1}e(x)$

$$e_r(y)=\frac{e(y)}{y}=\frac{nx^{n-1}e(x)}{x^n}=n\frac{e(x)}{x}=ne_r(x)$$

所以x^n 的相对误差是 x 的相对误差的n倍.

x^2的相对误差是 x 的相对误差的 2 倍，

$\sqrt{x}$ 的相对误差是 x 的相对误差的 1/2 倍.

算法（程序）设计的注意事项

例：近似数的运算 1.2 - 1.1 = 0.1 结果存在有效数字吗？

NO, 因为[1.15, 1.25] – [1.05, 1.15] ϵ [0, 0.2] 区间数学

➢ **1. 避免两个相近的数相减**

如果 x，y 的近似值分别为x^*, y^*，则 $z^*=x^*-y^*$ 是 $z=x-y$ 的近似值. 此时，相对误差满足估计式

$$|e_r(z^*)|=\left|\frac{e(x^*-y^*)}{x^*-y^*}\right|,$$

可见，当x^*与 y^*很接近时，z^*的相对误差有可能很大.

算法（程序）设计的注意事项

● 例如 当$x_1\approx x_2$时， $\log x_1-\log x_2=\log\frac{x_1}{x_2}$

当$x\approx 0$时， $1-\cos x=2\sin^2\frac{x}{2}$

当$x>>1$时， $\sqrt{x+1}-\sqrt{x}=\frac{1}{\sqrt{x+1}+\sqrt{x}}$

● **例** 求方程 $x^2-64x+1=0$的两个根($\sqrt{1023}\approx31.984$)

解 由求根公式有 $x_1=32+\sqrt{1023}\approx63.984$

若由 $x_2=32-\sqrt{1023}\approx0.016$，仅有两位有效数字，

但若采用 $x_2=1/x_1\approx0.01563$，则有四位有效数字.

算法（程序）设计的注意事项

例：近似数的运算 1.200 + 1e20 = ?

➢ **2. 防止大数“吃掉”小数**

● 因为计算机上只能采用有限位数计算，若参加运算的数量级差很大，在它们的加、减运算中，绝对值很小的数往往被绝对值较大的数“吃掉”，造成计算结果失真. （因为对于加减法，“数位要对齐”）

● 在求和或差的过程中应采用由小到大的运算过程.

算法（程序）设计的注意事项

例：近似数的运算 1.200 / 1.100e-200 = ?

➤ **3. 绝对值太小的数不宜作除数**

由于除数很小，将导致商很大，有可能出现"溢出"现象. 另外，设 x，y 的近似值分别为 x^*，y^*，则 $z^*=x^*/y^*$ 是 $z=x/y$ 的近似值. 此时，

$$|e(z)| = \left|\frac{ye(x)-xe(y)}{y^2}\right| \le \frac{|y||e(x)|+|x||e(y)|}{y^2}$$

算法（程序）设计的注意事项

例 用Cramer法则求n阶线性方程组$Ax=b$的解，用n阶行列式定义来计算，乘法运算次数 > $(n+1)n!$

当n=25时，在每秒百亿次乘除运算计算机上求解时间为

$$\frac{26!}{10^{10}\times 3600\times 24\times 365} \approx 13\ (\text{亿年})$$

➤ **4. 注意简化计算程序，减少计算次数**

- 首先，若算法计算量太大，实际计算无法完成
- 其次，即使是可行算法，则计算量越大积累的误差也越大. 因此，算法的计算量越小越好.

浮点数（float）的表示与进制

---- IEEE-754(-1985)标准

定点 (fixed-point) 数：数的定点格式

- 定点数：机器中存储数据的小数点位置固定不变
- 缺点：表示的数的精度和范围都很受限
- 优点：节省硬件成本 (主要用于早期的计算机中)

- 计算机中通常采用两种简单约定（除符号位外）：
- 定点小数：小数点的位置固定在数据最高位之前
- 定点整数：小数点的位置固定在数据最低位之后

- 定点小数：纯小数(pure decimal)，整数部分为0
- 非零纯小数二进制：$x = x_0.x_1x_2\ldots x_t$ $(x_0$: 0正，1负$)$
- 表示(非零)数的范围：$2\^(-t) \le |x| \le 1 - 2\^(-t)$

多选题 2分 设置

关于计算机中的浮点数（按照国际标准），下列表述正确的是

A 对于[0,1]区间内的某些实数，在计算机内部无法用浮点数精确表示

B 对于任意一个真分数，在计算机内部都可以用浮点数精确表示

C 对于[0,1]区间内不同实数的浮点数表示，绝对误差和相对误差都可能不同

D 同一个非零实数，可能有多种浮点数表示方法

E 以上都不对

提交

数的表示与进制 以0.33为例

10进制	2进制 $(0.010101\ldots)_2$		
$3*10^{-1}$	$0*2^{-1}$	0.33 * 2 = 0.66	取整数位0
$+3*10^{-2}$	$+1*2^{-2}$	0.66 * 2 = 1.32	取整数位1
	$+0*2^{-3}$	0.32 * 2 = 0.64	取整数位0
	$+1*2^{-4}$	0.64 * 2 = 1.28	取整数位1
	$+0*2^{-5}$	0.28 * 2 = 0.56	取整数位0
	$+1*2^{-6}$	0.56 * 2 = 1.12	取整数位1
	+…	……	无法有限位精确表示！

浮点数：是属于有理数中某特定子集的数的数字表示

在计算机中用以近似表示任意某个实数

浮点数

(有效数字)

符号　尾数　阶码

一般非零 β 进制数(d_1非0) $\pm .d_1d_2\cdots d_t \times \beta^e$ $(L \le e \le U)$

尾数(mantissa)亦称定点部分(fixed point part)、浮点系数(floating point coefficient), 是纯小数部分

(非零)浮点数的 x范围　$m \le |x| \le M$

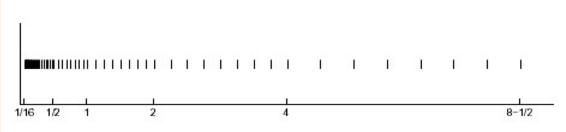
$$m = \beta^{L-1}, M = \beta^U \sum_{i=1}^{t} (\beta - 1)\beta^{-i} = \beta^U (1 - \beta^{-t})$$

只用有限位表示，能够精确表达的数总是有限的！

尾数决定表示精度（有效数字），阶码决定表示范围！

浮点数

- $\beta=2, t=4$，$L=-3$，$U=3$，则(正)浮点数的集合X为



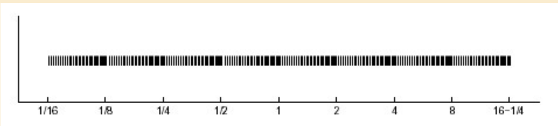
$$m = 2^{L-1} = 1/16, M = 2^U(1-2^{-t}) = 8(1-1/16) = 7.5$$

特点：分布不均匀，各处的绝对误差（限）不相同

但各处的相对误差（限）基本相同

浮点数

- $\beta=2$, $t=6$，$L=-3$，$U=4$，这时采用对数坐标，则正浮点数的集合X为



特点：分布仍不均匀，对均匀程度有很大改善

规格化浮点数

符号　尾数　阶码

一般非零 β 进制数(d_1非0) $\pm .d_1d_2\cdots d_t \times \beta^e$ $(L \le e \le U)$

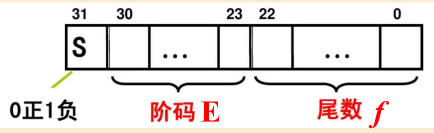
(非零)浮点数 x　$m \le |x| \le M$

$$m = \beta^{L-1}, M = \beta^U \sum_{i=1}^{t} (\beta - 1)\beta^{-i} = \beta^U (1 - \beta^{-t})$$

IEEE-754(-1985)标准: 2进制(Float)　$\pm(1+f)\cdot 2^e$, $0 \le f < 1$

	符号	尾数f	阶码E	位移(Bias)	(阶码全0/全1保留)
single	1	23	8	127	$-126 \le e \le 127$
double	1	52	11	1023	$-1022 \le e \le 1023$

以32位规格化浮点数为例



阶码：8位以2为底，阶码 = 阶码真值 + 127

尾数：23位，采用隐含尾数最高位1的表示方法，实际尾数24位，尾数真值 = 1 + 尾数

真值为：$(-1)^S \times 2^{阶码-127} \times (1 + 尾数)$

填空题　2分　设置

将十进制数 “– 0.75” 用浮点数表示（32位二进制表示），从左至右前12位为:

[填空1]

正常使用填空题需3.0以上版本雨课堂

作答

IEEE754：几个特殊数据的存储规则

正/负0: 最高位(符号位)为0/1，其它数据位是0；
正/负无穷: 符号位为0/1，阶码位全为1，有效数字（尾数f）全为0；

NaN: Not-a-Number
非法的浮点数，阶码位全为1，有效数字不全为0；（如：无穷除以无穷时的结果）

非规格化数：阶码位全为0 (即移码-127)，尾数没有隐含位“1”，扩大数的表示范围（但精度降低）

这样IEEE-754有5种类型浮点数据, 如下表 (single)

Sign	E (e=E-127)	f	意义
0/1	0	0	±0
0/1	0	非0	非规格化数
0/1	1~254	任意	规格化数
0/1	255	0	±Inf
0/1	255	非0	NaN

例: 把100.25转换成协处理器中的浮点数

进制转换：$(100.25)_{10}=(1100100.01)_2$
规格化： $(1100100.01)_2=1.10010001\times 2^6$
$=1.10010001\times 2^{(110)_2}$
计算阶码： 110+01111111＝10000101
数值符号位：0，
阶码为：10000101，
尾数为：1001 0001 0000 0000 0000 000
结果(single)：
0 10000101 10010001000000000000000

也经常用16进制表示为（结尾H表示hexadecimal） 42C88000H

填空题 2分 设置

浮点数C1C90000H表示的十进制数为

[填空1]

正常使用填空题需3.0以上版本雨课堂

作答

Matlab中的浮点数: 遵循IEEE754标准

缺省为双精度（double, 64 bits, i.e., 8 Bytes ）

realmin: 2.2251e-308， 即 2^(-1022)
realmin('single'): 1.1755e-38， 即 2^(-126)

realmax: 1.7977e+308, 即(2-eps)*2^1023
realmax('single'): 3.4028e+38， 即(2-eps('single'))*2^127

eps 或 eps(1) 或 eps('double'): 2.2204e-16， 即 2^(-52)
eps('single') 或 eps(single(1)): 1.1921e-07， 即 2^(-23)

绝对误差各不相同, 如: eps(0) = 2^(-1074)=4.9407e-324

舍入误差相对误差限eps/2 (相邻两数相对差不超过eps)

Matlab中的整数

（缺省为 32 bits）

类型： 'int8', 'uint8', 'int16', 'uint16',
'int32', 'uint32', 'int64', 'uint64'

intmax 或 intmax('int32'): 2147483647, 即2^31 -1
intmin 或 intmin('int32'): -2147483648, 即-2^31

intmax('int64'): 9223372036854775807, 即2^63 -1
intmax('uint64'): 18446744073709551615, 即2^64 -1

intmin('uint8'), …, intmin('int64'): 0

其他相关函数：round, floor, ceil, fix; int, int8, uint8, …

浮点数运算--- example (MATLAB)

```
format long
x = 4/3 - 1
y = 3*x
z = 1 - y
```

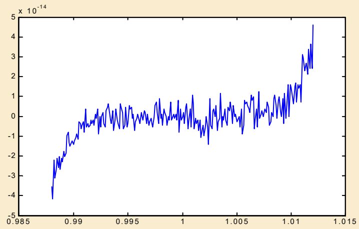
Results

```
x =
   0.33333333333333
y =
   1.00000000000000
z =
   2.220446049250313e-016
```

浮点数运算--- example (MATLAB)

```
x=0.988:.0001:1.012;
y=x.^7-7*x.^6+21*x.^5-35*x.^4+35*x.^3-21*x.^2+7*x-1;
plot(x,y)
```

Results



浮点数运算--- example (MATLAB)

浮点数计算满足

（加法，乘法）交换率？结合律？分配律？

Results

NO !!

复杂度—行列式，an example

- 回忆：2阶问题，3阶问题
- 考虑一般矩阵的行列式定义

$$\det(A)=\sum_{(i_1,i_2,\cdots,i_n)\in\pi}(-1)^{\tau}\, a_{i_1}a_{i_2}\ldots a_{i_n}$$

- 按定义计算，需要的乘法次数

$$(n-1)\,n!$$

复杂度

- **指数型算法**
  - 算法计算量是问题规模的指数函数
  - 只能够处理规模很小的问题
- **多项式型算法**
  - 算法计算量是问题规模的多项式函数
  - 可以处理规模较大的问题

复杂度— examples

Descriptor	Data Set Size in Bytes	Storage Mode
Tiny	10^2	Piece of Paper
Small	10^4	A Few Pieces of Paper
Medium	10^6	A Floppy Disk
Large	10^8	Hard Disk
Huge	10^{10}	Multiple Hard Disks
Massive	10^{12}	Robotic Magnetic Tape Storage Silos
Super-massive	10^{15}	Distributed Data Archives

The Huber-Wegman Taxonomy of Data Set Sizes

复杂度— examples

Algorithmic Complexity

$O(n^{1/2})$	Plot a Scatter-plot
$O(n)$	Calculate Means, Variances, Kernel Density Estimates
$O(n \log(n))$	Calculate Fast Fourier Transforms
$O(n\,c)$	Calculate Singular Value Decomposition of an r * c Matrix; Solve a Multiple Linear Regression
$O(n^2)$	Solve most Clustering Algorithms
$O(a^n)$	Detect Multivariate Outliers

Complexity

Number of Operations for Algorithms of Various Computational Complexities and Various Data Set Sizes

n	$n^{1/2}$	n	$n \log(n)$	$n^{3/2}$	n^2
tiny	10	10^2	$2x10^2$	10^3	10^4
small	10^2	10^4	$4x10^4$	10^6	10^8
medium	10^3	10^6	$6x10^6$	10^9	10^{12}
large	10^4	10^8	$8x10^8$	10^{12}	10^{16}
huge	10^5	10^{10}	10^{11}	10^{15}	10^{20}

Complexity

Computational Feasibility on a Pentium PC
10 megaflop performance assumed

n	$n^{1/2}$	n	$n \log(n)$	$n^{3/2}$	n^2
tiny	10^{-6} seconds	10^{-5} seconds	$2x10^{-5}$ seconds	.0001 seconds	.001 seconds
small	10^{-5} seconds	.001 seconds	.004 seconds	.1 seconds	10 seconds
medium	.0001 seconds	.1 seconds	.6 seconds	1.67 minutes	1.16 days
large	.001 seconds	10 seconds	1.3 minutes	1.16 days	31.7 years
huge	.01 seconds	16.7 minutes	2.78 hours	3.17 years	317,000 years

Complexity

Computational Feasibility on a Silicon Graphics Onyx Workstation
300 megaflop performance assumed

n	$n^{1/2}$	n	$n \log(n)$	$n^{3/2}$	n^2
tiny	$3.3x10^{-8}$ seconds	$3.3x10^{-7}$ seconds	$6.7x10^{-7}$ seconds	$3.3x10^{-6}$ seconds	$3.3x10^{-5}$ seconds
small	$3.3x10^{-7}$ seconds	$3.3x10^{-5}$ seconds	$1.3x10^{-4}$ seconds	$3.3x10^{-3}$ seconds	.33 seconds
medium	$3.3x10^{-6}$ seconds	$3.3x10^{-3}$ seconds	.02 seconds	3.3 seconds	55 minutes
large	$3.3x10^{-5}$ seconds	.33 seconds	2.7 seconds	55 minutes	1.04 years
huge	$3.3x10^{-4}$ seconds	33 seconds	5.5 minutes	38.2 days	10,464 years

复杂度

----对于直接方法的度量标准

- **Ax=b 的Gauss 消去法**
- **线性规划问题的Simplex方法**
- **组合优化的问题和方法**

例：多项式计算

f(x)= ax^5 + bx^4 +cx^3 +dx^2 +ex +f

f(x)= ((((ax+ b) x + c) x +d) x +e) x + f

收敛性

----刻划算法的另外一个重要概念

- 误差

$$\varepsilon_n = \left\| x_n - x^* \right\|$$

- 收敛性

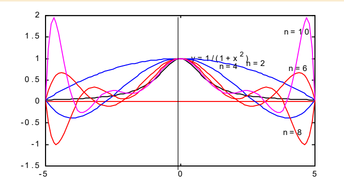
$$\lim_{n \to \infty} \varepsilon_n = 0 .$$

拉格朗日插值多项式的不收敛性

$$n\uparrow \;\Rightarrow\; L_n(x)\,? \Rightarrow |R_n(x)|\downarrow\ ?$$

Example $\quad g(x)=\dfrac{1}{1+x^2},\ -5\le x\le 5$

Runge现象



$$\lim_{n\to\infty} L_n(x)=g(x),\ -3.63\le x\le 3.63$$

梯形公式和辛普森公式的收敛性

若对I某个数值积分 I_n 有

$$\lim_{n\to\infty}\frac{I-I_n}{h^p}=c \text{ (非零常数)}$$

则称 I_n是 p 阶收敛的。

梯形公式 2 阶收敛，辛普森公式 4 阶收敛。

微分方程初值问题 算法的收敛性 与收敛速度

一个算法的局部截断误差为$O(h^{p+1})$

⇩

该算法具有p阶精度

	局部截断误差	精度
向前欧拉公式	$O(h^2)$	1阶
向后欧拉公式	$O(h^2)$	1阶
梯形公式	$O(h^3)$	2阶
改进欧拉公式	$O(h^3)$	2阶
经典龙格-库塔公式	$O(h^5)$	4阶

线性方程组迭代法的收敛性

一般迭代形式 $\quad x^{(k+1)}=Bx^{(k)}+f$

迭代k次得到 $\quad x^{(k)}-x^*=B^k(x^{(0)}-x^*)$

序列收敛 $x^{(k)}\to x^*(k\to\infty)$的充要条件

$B^k\to 0(k\to\infty)$ $\boxed{\Leftrightarrow B\text{的所有特征根（取模）小于 }1}$

B的谱半径 $\rho(B)=\max\limits_{1\le i\le n}|\lambda_i|$ ⇕ $\boxed{\rho(B)<1}$

$\lambda_i(i=1,\cdots n)$是B的特征根

非线性方程（组）解法

（牛顿）切线法 $\quad x_{k+1}=x_k-\dfrac{f(x_k)}{f'(x_k)}$

单根➔2阶收敛

重根➔1阶收敛

重数越高，收敛越慢

（牛顿）割线法 $\quad x_{k+1}=x_k-\dfrac{f(x_k)(x_k-x_{k-1})}{f(x_k)-f(x_{k-1})}$

单根➔收敛阶1.618

推广到方程组：拟牛顿法（DFP; BFGS等）

病态性

-----刻划模型的概念

- 考虑如下的问题

f(x)=(x-1)(x-2)…….(x-20)

显然方程 f(x)=0 的解是

1 2 3 4 ……… 19 20

请问: 如下方程的解是什么？

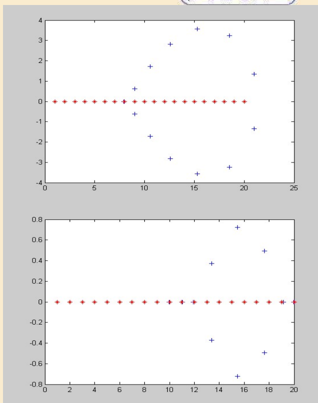
$$f_\varepsilon(x)=f(x)+\varepsilon x^{18}=0$$

Matlab program

```
p=poly(1:20);
ep=zeros(1,21);
ep(3)=1.0e-5;
re=roots(p+ep)
plot(re,'b+');
hold on
plot(1:20,0,'r*');
hold off
```


ε=10e-5

ε=10e-8



较小的根不敏感

较大的根较敏感

病态！！

病态的线性方程组

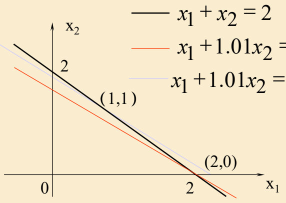
$$A \quad x=b$$

$$\begin{bmatrix}1 & 1\\ 1 & 1.01\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix}=\begin{bmatrix}2\\ 2\end{bmatrix}$$

$x_1+x_2=2$

$x_1+1.01x_2=2$

$x_1+1.01x_2=2.01$



$$x=\begin{bmatrix}2\\ 0\end{bmatrix} \qquad b=\begin{bmatrix}2\\ 2.01\end{bmatrix}$$

$$x=\begin{bmatrix}1\\ 1\end{bmatrix}$$

x对b的扰动敏感

$Ax=b$，如果解 x 对 b 或 A 的扰动敏感，就称方程组是病态的，也称系数矩阵 A 是病态的。

稳 定 性

-----刻划算法的关键概念

- **考虑如下的序列**

$$E_n=\int_0^1 x^n e^x\,dx, \quad n=1,2,\cdots$$

- **可以证明**

$$E_n=e-nE_{n-1}$$

$$0<E_n<e/(n+1)$$

两个算法

----有什么差别，哪个可以用??

Algorithm 1

$$E_1=1$$

$$E_n=e-nE_{n-1}$$

$$n=2,3,\cdots$$

Algorithm 2

$$E_N=0,$$

$$E_{n-1}=(e-E_n)/n$$

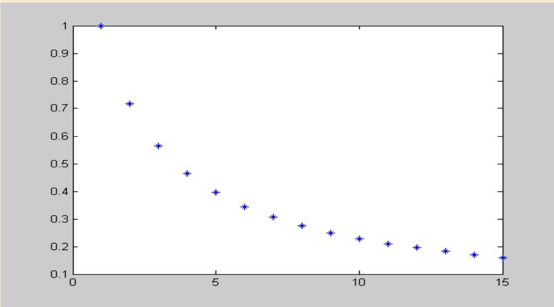
$$n=N,N-1,\cdots,1$$

Program of Algorithm 1

```
clear
ep(1)=1
for n=2:100
    ep(n)=exp(1.0)-n*ep(n-1)
end
plot(ep,'b*');
```

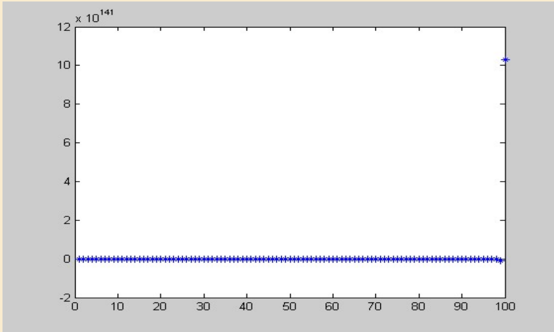
清華大学 大学数学实验

Algorithm 1 with n=15



清華大学 大学数学实验

Algorithm 1 with n=100



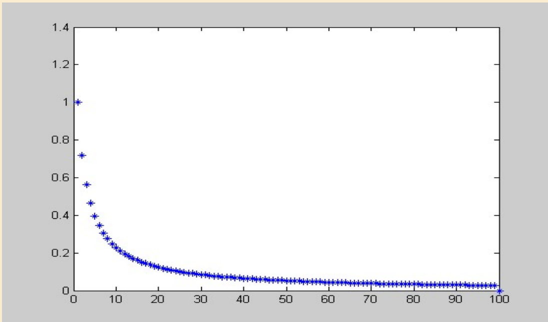
清華大学 大学数学实验

Program of Algorithm 2

```
clear
ep(100)=0
for n=100:-1:2
    ep(n-1)=(exp(1.0)-ep(n))/n;
end
plot(ep,'b*');
```

清華大学 大学数学实验

Algorithm 2 with n=100



简单的计算发现，可以使用的算法是 -- **Algorithm 2 !**

清華大学 大学数学实验

科学结论的取得，不能仅依靠感觉

- 计算中误差并不可怕，重要的是误差在算法中的传播。
- 稳定----算法中产生的任何误差，对后续计算的影响是衰减或可以控制的。
- 不稳定的算法 = 不能用的垃圾！

清華大学 大学数学实验

微分方程初值问题数值算法的稳定性

稳定性 计算中舍入误差不会随步数的增加无限增大

y_n的误差ε_n $\boxed{|\varepsilon_{n+k}| \le |\varepsilon_n|,\ \ k=1,2,\cdots}$ $\Rightarrow$ **算法稳定**

$y' = f(x,y)$ $\Rightarrow$ $\boxed{y' = f(x^*,y^*) + f_x(x^*,y^*)(x-x^*) + f_y(x^*,y^*)(y-y^*)}$

$\Rightarrow$ $y' = -\lambda y,\ \lambda > 0$ (特征根 -λ) $\Rightarrow$ $y = ce^{-\lambda x}$ λ>0→微分方程稳定

向前欧拉公式 $y_{n+1} = y_n + hf(x_n, y_n) = (1-h\lambda)y_n$ $\Rightarrow$ $\varepsilon_{n+1} = (1-\lambda h)\varepsilon_n$

$|\varepsilon_{n+k}| \le |\varepsilon_n|$ $\Rightarrow$ $|1 - h\lambda| \le 1$ $\Rightarrow$ $h \le 2/\lambda$

向后欧拉公式 $y_{n+1} = y_n - h\lambda y_{n+1}$ $\Rightarrow$ $\varepsilon_{n+1} = \frac{1}{1+h\lambda}\varepsilon_n$ $\Rightarrow$ ***h*任意**

经典龙格-库塔公式 $\boxed{h \le 2.785/\lambda}$

大学数学实验

实验3 插值与数值积分

补充：数值积分的收敛性与稳定性

清 华 大 学 数 学 科 学 系

插值求积公式

设已知f(x)在节点 $x_k(k=0,1,\cdots,n)$ 有函数值 $f(x_k)$,作n次拉格朗日插值多项式

$$P(x)=\sum_{k=0}^{n} f(x_k)l_k(x)$$

其中，对k=0,...,n

$$l_k(x)=\prod_{\substack{j=0\\ j\neq k}}^{n}\frac{x-x_j}{x_k-x_j}=\frac{\omega(x)}{(x-x_k)\omega'(x_k)}$$

$$\omega(x)=(x-x_0)(x-x_1)\cdots(x-x_n)$$

取 $\int_a^b P(x)dx$ 作为 $\int_a^b f(x)dx$ 的近似值，即

$$\int_a^b f(x)dx \approx \int_a^b P(x)dx=\int_a^b \sum_{k=0}^{n} f(x_k)l_k(x)dx$$

$$=\sum_{k=0}^{n} f(x_k)\int_a^b l_k(x)dx$$

记为

$$\boxed{\int_a^b f(x)dx \approx \sum_{k=0}^{n} f(x_k)A_k}$$

其中

$$A_k=\int_a^b l_k(x)dx=\int_a^b \frac{\omega(x)}{(x-x_k)\omega'(x_k)}dx$$

称为求积系数。

定义　　求积公式

$$\boxed{\int_a^b f(x)dx \approx \sum_{k=0}^{n} A_k f(x_k)}$$

当其系数 $A_k=\int_a^b l_k(x)dx$时，则称求积公式为插值求积公式。

某求积公式能对多大次数的多项式f(x)成为准确等式，是衡量该公式的精确程度的重要指标。

代数精度的定义：如果求积公式（4.1）对于一切次数小于等于m的多项式

$$f(x)=a_0+a_1x+a_2x^2+\cdots+a_mx^m$$

是准确的，而对于次数为m+1的多项式是不准确的，则称该求积公式具有m次代数精度。

定理　　n+1个节点的求积公式

插值型求积公式判断条件

$$\int_a^b f(x)dx \approx \sum_{k=0}^{n} A_k f(x_k)$$

为插值型求积公式←→公式至少具有n次代数精度。

证:必要性.设n+1个节点的求积公式

$$\int_a^b f(x)dx \approx \sum_{k=0}^{n} A_k f(x_k)$$

为插值型求积公式，求积系数为：

$$A_k=\int_a^b l_k(x)dx$$

又$f(x)=P(x)+R(x)$, 当f(x)为不高于n次的多项式时, f(x)=P(x), 其余项R(f)=0。因而这时求积公式至少具有n次代数精度。

大学数学实验

充分性：若求积公式至少具有n次代数精度，则对n次多项式

$$l_k(x)=\prod_{\substack{j=0\\ j\neq k}}^{n}\frac{x-x_j}{x_k-x_j}\quad (k=0,1,\cdots,n)$$

精确成立，即

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

其中

$$l_k(x)=\frac{(x-x_0)\ldots(x-x_{k-1})(x-x_{k+1})\ldots(x-x_n)}{(x_k-x_0)\ldots(x_k-x_{k-1})(x_k-x_{k+1})\ldots(x_k-x_n)}$$

注意$l_k(x_k)=1$，而当$j\neq k$的时候，$l_k(x_j)=0$

从而

$$\sum_{j=0}^{n}A_j l_k(x_j)=A_k$$

所以有 $A_k=\int_a^b l_k(x)dx$，即求积公式为插值型求积公式

大学数学实验

求积公式的收敛性和稳定性

一般地，求积公式

$$\int_a^b f(x)dx\approx\sum_{k=0}^{n}A_k f_k$$

通常称为机械求积公式。

插值型求积公式

$$\int_a^b f(x)dx\approx\sum_{k=0}^{n}A_k f(x_k)$$

其中 $a\leq x_0<x_1<\cdots<x_n\leq b$

$$A_k=\int_a^b l_k(x)dx$$

使用了插值基函数的定积分作为系数。

大学数学实验

求积公式的收敛性定义

定义 在求积公式 $\int_a^b f(x)dx\approx\sum_{k=0}^{n}A_k f_k$ 中，若

$$\lim_{\substack{n\to\infty\\ h\to 0}}\sum_{k=0}^{n}A_k f(x_k)=\int_a^b f(x)dx$$

其中 $h=\max_{1\leq i\leq n}(x_i-x_{i-1})$，则称求积公式$\sum_{k=0}^{n}A_k f(x_k)$是收敛的。

大学数学实验

求积公式的稳定性定义

设$f(x_k)$有误差δ_k，即 $f(x_k)-\tilde{f}_k=\delta_k, k=0,1,\ldots,n$，则有

$$|I_n(f)-I_n(\tilde{f})|=|\sum_{k=0}^{n}A_k[f(x_k)-\tilde{f}_k]|$$

定义 若$\forall\varepsilon>0,\exists\delta>0$，只要$|f(x_k)-\tilde{f}_k|\leq\delta$，$k=0,1,\ldots,n$，就有

$$|I_n(f)-I_n(\tilde{f})|=|\sum_{k=0}^{n}A_k[f(x_k)-\tilde{f}_k]|\leq\varepsilon$$

则称公式 是稳定的。

不会由$f(x_k)$的微小误差而导致积分值产生大的变化

单选题 2分 设置

对于插值型求积公式（如右），以下哪个是其稳定的充分条件？

$$\int_a^b f(x)dx\approx\sum_{k=0}^{n}A_k f_k$$

- A $|\sum_{k=0}^{n}A_k|<1$
- B $A_k<1, k=0, 1,\ldots,n$
- C $A_k>1, k=0, 1,\ldots,n$
- D $A_k>0, k=0, 1,\ldots,n$
- E $A_k<0, k=0, 1,\ldots,n$
- F 以上都不对

提交

大学数学实验

定理 若求积公式中的系数$A_k>0$, $k=0, 1,\ldots,n$，则求积公式是稳定的。

证明：

当$|f(x_k)-\tilde{f}_k|\leq\delta$，$k=0,1,\ldots,n$，有

$$|I_n(f)-I_n(\tilde{f})|\leq\sum_{k=0}^{n}A_k\,|[f(x_k)-\tilde{f}_k]|$$

$$\leq\delta\sum_{k=0}^{n}A_k=(b-a)\delta$$

按照稳定性定义，求积公式是稳定的。

清华大学 大学数学实验

思考与练习

- 二次代数方程的求根，下面公式何时更好？

$$\frac{-b\pm\sqrt{b^2-4ac}}{2a} \qquad \frac{2c}{-b\mp\sqrt{b^2-4ac}}$$

- sin(x)用下面公式计算，如何控制精度？

$$\sin(x)=x-x^3/3!+x^5/5!-x^7/7!+\cdots$$

- exp(x): x<0时，用下面公式计算好不好？

$$e^x=1+x+x^2/2!+x^3/3!+x^4/4!+\cdots$$

清华大学 大学数学实验

谢　谢！

孔子曰：

“学而不思则罔，思而不学则殆。”

----- 与同学们共勉！