数值分析
Numerical Analysis

主讲教师： 曲延云
yyqu@xmu.edu.cn

- 教材：

  数值分析（第四版） 李庆扬，王能超，易大义（清华大学出版社）

- 参考资料：

  数值计算方法　林成森　（科学出版社）

  数值分析　陈昌明　（厦门大学出版社）

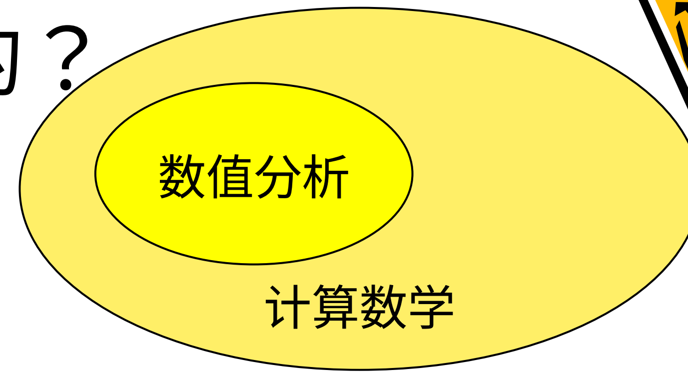
课程评分方法

- 课程成绩（100%）= 考试成绩(80%)+
  作业与实验
  （20%）

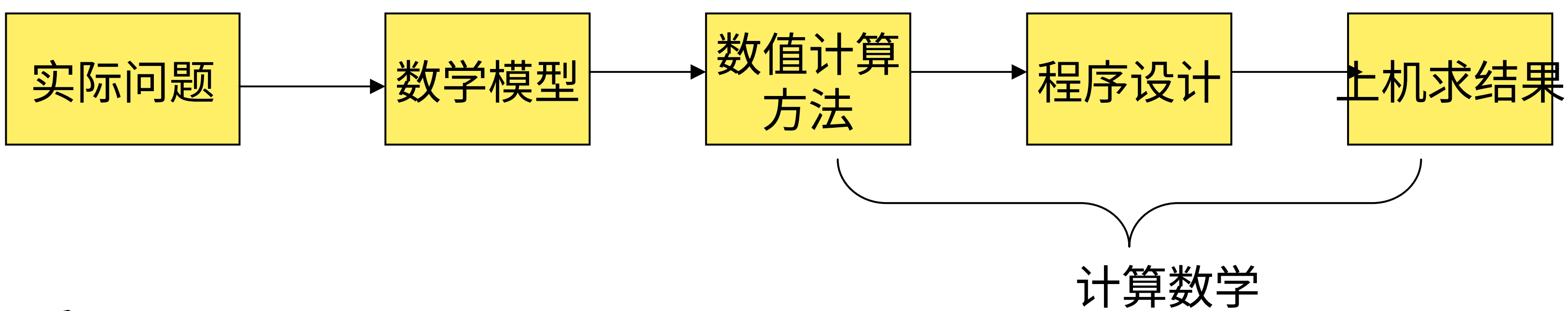
Chapter 1 绪论

- 数值分析研究对象与特点
- 数值计算的误差
- 误差定性分析与避免误差危害

- 数值分析是做什么用的？



计算数学是数学科学的一个分支，研究用计算机求解各种数学问题的数值计算方法及其理论与软件实现.



输入复杂问题或运算

$\sqrt{x}$, a^x, $\ln x$, $A\vec{x}=\vec{b}$,

$\int_a^b f(x)dx$, $\frac{d}{dx}f(x)$, ……

数值分析

$+\quad -$

$\times\quad \div$

计算机

近似解

内容：函数的数值逼近，数值微分与积分，非线性方程数值解，数值线性代数，微分方程数值解

- 与纯数学课程的区别（以线性代数为例）

➢ 线性代数：只介绍解的存在性、唯一性、及有关理论和精确解法。不能在计算机上求解几百甚至上千个未知数的方程。

➢ 数值分析：根据方程特点，研究适合计算机使用的，满足精度要求，计算时间省的有效算法及相关理论。

数值分析的特点：

- ✓ 面向计算机，根据计算机特点提供切实可行的有效算法。（+，-，×，÷，逻辑运算）
- ✓ 有可靠的理论分析，能任意逼近并达到精度要求，对近似算法要保证收敛性和数值稳定性，并对误差分析。
- ✓ 要有好的计算复杂性，时间复杂性好是指节省时间，空间复杂性好是指节省存储量，这些关系到算法能否在计算机上实现。
- ✓ 要有数值实验，即算法除了理论上满足上述三点外，还要通过数值实验证明其有效性。

数值计算的误差

误差的来源与分类

- ✓ 模型误差——数学模型与实际问题之间出现的误差
- ✓ 观测误差——由于观察造成的误差
- ✓ 截断误差——当数学模型得不到精确解时，用数值方法求近似解，其近似解与精确解之间的误差，称为截断误差或方法误差。
- ✓ 舍入误差——由于计算机字长有限，原始数据在计算机上表示会产生误差，计算过程有可能产生新的误差。

- 截断误差

$$P_n(x) = f(0) + \frac{f'(0)}{1!}x + \frac{f''(0)}{2!}x^2 + \cdots + \frac{f^{(n)}(0)}{n!}x^n$$

$$R_n(x) = f(x) - P_n(x) = \frac{f^{(n+1)}(\xi)}{(n+1)!}x^{n+1}$$

- 舍入误差

$$R = \pi - 3.14159 = 0.0000026\cdots$$

- 误差与有效数字

绝对误差：e*=x*-x

误差限：|e*|≤ ε，通常取末位数字的半个单位

相对误差： $e_r^* = \dfrac{e^*}{x} = \dfrac{x^* - x}{x}$

常用替代公式： $e_r^* = \dfrac{e^*}{x^*} = \dfrac{x^* - x}{x^*}$

注 $\dfrac{e^*}{x} - \dfrac{e^*}{x^*} = \dfrac{(e^*/x^*)^2}{1-(e^*/x^*)}$

相对误差限： $\left|e_r^*\right| \le \varepsilon_r^* = \dfrac{\varepsilon^*}{x^*}$

- 有效数字：若近似值x*的误差限是某一位的半个单位，该位到第一位非零数字共有n位，就说x*有n位有效数字，可表示为

$$x^* = \pm 10^m \times (a_1 + a_2 \times 10^{-1} + \cdots + a_n \times 10^{-(n-1)})$$

- 其中 $0 \le a_i \le 9, a_1 \ne 0$
- 且 $\left| x - x^* \right| \le \frac{1}{2} \times 10^{m-n+1}$
- 例1 按四舍五入原则写出下列各数具有5位有效数字的近似数 187.9325,0.0378551,8.000033

- 例2 重力加速度 $g \approx 9.80 m/s^2$

$$g \approx 0.00980 km/s^2$$

都具有三位有效数字。

$$|g - 9.80| \le \frac{1}{2} \times 10^{-2} \qquad \varepsilon_1^* = \frac{1}{2} \times 10^{-3} km/s^2$$

$$|g - 0.00980| \le \frac{1}{2} \times 10^{-5} \qquad \varepsilon_2^* = \frac{1}{2} \times 10^{-5} km/s^2$$

相对误差都是 $\varepsilon_r^* = \dfrac{0.005}{9.80} = \dfrac{0.000005}{0.00980}$

- 相对误差与相对误差限是无量纲的，绝对误差与相对误差限是有量纲的。
- 有效数字与小数点位数无关
- 有效位数越多，绝对误差限越小。

- Th1 设近似数x*表示为

$$x^* = \pm 10^m \times (a_1 + a_2 \times 10^{-1} + \cdots + a_n \times 10^{-(n-1)})$$

若x*具有n位有效数字，则其相对误差限为

$$\varepsilon_r^* \le \frac{1}{2a_1} \times 10^{-(n-1)}$$

反之，若x*的相对误差限 $\varepsilon_r^* \le \dfrac{1}{2(a_1+1)} \times 10^{-(n-1)}$

则至少具有n位有效数字。

注：有效位数越多，相对误差越小。

- 例3 要使$\sqrt{20}$ 的近似值相对误差小于0.1%，要取几位有效数字？

- 数值运算的误差估计

两个近似数x^*_1,x^*_2，其误差限分别为 $\varepsilon(x^*_1)$, $\varepsilon(x^*_1)$,则

$$\varepsilon(x_1^* \pm x_2^*) = \varepsilon(x_1^*) + \varepsilon(x_1^*)$$

$$\varepsilon(x_1^* x_2^*) \approx |x_1^*|\varepsilon(x_2^*) + |x_2^*|\varepsilon(x_1^*)$$

$$\varepsilon(x_1^* / x_2^*) \approx \frac{|x_1^*|\varepsilon(x_2^*) + |x_2^*|\varepsilon(x_1^*)}{|x_2^*|^2}$$

- 更一般的情况，当自变量有误差时，计算函数值产生的误差限可利用函数的Taylor展开进行估计。

$$f(x)-f(x^*)=f'(x)(x-x^*)+\frac{f''(\xi)}{2}(x-x^*)^2$$

$$|f(x)-f(x^*)|\le|f'(x)|\varepsilon(x^*)+\frac{|f''(\xi)|}{2}\varepsilon^2(x^*)$$

$$|f(x)-f(x^*)|\approx|f'(x)|\varepsilon(x^*)$$

(一阶导数值与二阶导数值相差不大时)

- 当f为多元函数时，计算函数值误差的方法

$$A = f(x_1, \cdots, x_n) \qquad A^* = f(x_1^*, \cdots, x_n^*)$$

$$e(A) = A^* - A = f(x_1^*, \cdots, x_n^*) - f(x_1, \cdots, x_n)$$

$$\approx \sum_{k=1}^{n} \left(\frac{\partial f(x_1^*, \cdots, x_n^*)}{\partial x_k}\right)(x_k^* - x_k)$$

$$= \sum_{k=1}^{n} \left(\frac{\partial f}{\partial x_k}\right) * \varepsilon(x_k^*)$$

- 误差限： $\varepsilon(A^*) \approx \sum_{k=1}^{n} \left| \left(\frac{\partial f}{\partial x_k}\right)^* \right| \varepsilon(x_k^*)$

- 相对误差限： $\varepsilon_r^* = \sum_{k=1}^{n} \left(\frac{\partial f}{\partial x_k}\right) * \frac{\varepsilon(x_k^*)}{|A^*|}$

例4 已测得某场地长$l^*=110m$,宽$d^*=80m$,已知$|l-l^*|\leq 0.2m$, $|d-d^*|\leq 0.1m$,试求面积的绝对误差限和相对误差限。

误差的定性分析与避免误差危害

病态问题与条件数

- 病态问题：如果输入数据有微小扰动，引起输出数据相对误差很大，就是病态问题。
- 条件数：$C_p = \left|\frac{f(x)-f(x^*)}{f(x)}\right| / \left|\frac{\Delta x}{x}\right| \approx \left|\frac{xf'(x)}{f(x)}\right|$

- 一般情况下，$C_p \geq 10$就认为是病态，C_p越大病态越严重。

算法的数值稳定性

数值不稳定：一个算法如果输入数据有误差，而在计算过程中舍入误差不增长，则称此算法是数值稳定的，否则，称此算法数值不稳定。

例：计算 $I_n = \frac{1}{e}\int_0^1 x^n e^x dx$ ， $n = 0, 1, 2, \ldots\ldots$

公式一： $I_n = 1 - n\, I_{n-1}$

注意此公式精确成立

$$I_0 = \frac{1}{e}\int_0^1 e^x dx = 1 - \frac{1}{e} \approx 0.63212056 \xlongequal{\text{记为}} I_0^*$$

则初始误差 $|E_0| = |I_0 - I_0^*| < 0.5\times 10^{-8}$

$$\frac{1}{e}\int_0^1 x^n \cdot e^0\, dx < I_n < \frac{1}{e}\int_0^1 x^n \cdot e^1\, dx \qquad \therefore \frac{1}{e(n+1)} < I_n < \frac{1}{n+1}$$

$$I_1^* = 1 - 1\cdot I_0^* = 0.36787944$$

$$\cdots\cdots\cdots\cdots$$

$$I_{10}^* = 1 - 10\cdot I_9^* = 0.08812800$$

$$I_{11}^* = 1 - 11\cdot I_{10}^* = 0.03059200$$

$$I_{12}^* = 1 - 12\cdot I_{11}^* = 0.63289600$$

$$I_{13}^* = 1 - 13\cdot I_{12}^* = -7.2276480$$

$$I_{14}^* = 1 - 14\cdot I_{13}^* = 94.959424$$

$$I_{15}^* = 1 - 15\cdot I_{14}^* = -1423.3914$$

考察第n步的误差$|E_n|$ $|E_n| = |I_n - I_n^*| = |(1-nI_{n-1}) - (1-nI_{n-1}^*)|$

$= n|E_{n-1}|$

可见初始的小扰动 $|E_0| < 0.5\times 10^{-8}$ 迅速积累，误差呈递增走势。

造成这种情况的是不稳定的算法 **/* unstable algorithm */**

✍ 公式二：

$$I_n = 1 - n\,I_{n-1} \quad \Rightarrow \quad I_{n-1} = \frac{1}{n}(1 - I_n)$$

$$\because \quad \frac{1}{e(N+1)} < I_N < \frac{1}{N+1}$$

可取 $$I_N^* = \frac{1}{2}\left[\frac{1}{e(N+1)} + \frac{1}{N+1}\right] \approx I_N$$

当 $N \to +\infty$ 时，$|E_N| = |I_N - I_N^*| \to 0$

取 $I_{15}^* = \frac{1}{2}\left[\frac{1}{e\cdot 16}+\frac{1}{16}\right] \approx 0.042746233$

$$\Rightarrow I_{14}^* = \frac{1}{15}(1-I_{15}^*) \approx 0.063816918$$

$$I_{13}^* = \frac{1}{14}(1-I_{14}^*) \approx 0.066870220$$

$$I_{12}^* = \frac{1}{13}(1-I_{13}^*) \approx 0.071779214$$

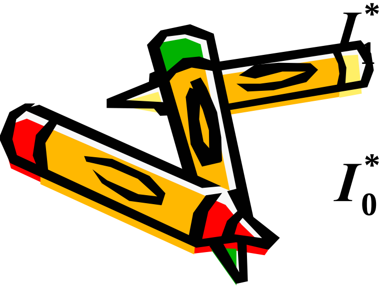
$$I_{11}^* = \frac{1}{12}(1-I_{12}^*) \approx 0.077351732$$

$$I_{10}^* = \frac{1}{11}(1-I_{11}^*) \approx 0.083877115$$

$$\vdots$$

$$I_1^* = \frac{1}{2}(1-I_2^*) \approx 0.36787944$$

$$I_0^* = \frac{1}{1}(1-I_1^*) \approx 0.63212056$$



考察反推一步的误差：

$$|E_{N-1}| = \left|\frac{1}{N}(1-I_N) - \frac{1}{N}(1-I_N^*)\right| = \frac{1}{N}|E_N|$$

以此类推，对 $n < N$ 有：

$$|E_n| = \frac{1}{N(N-1)\dots(n+1)}|E_N|$$

误差逐步递减, 这样的算法称为稳定的算法 /* stable algorithm */

- 在我们今后的讨论中，误差将不可回避，算法的稳定性会是一个非常重要的话题。

避免误差危害的若干原则

1. 避免除数绝对值远远小于被除数绝对值的除法。

例**6** 解线性方程组 $\begin{cases} 0.00001x_1 + x_2 = 1 \\ 2x_1 + x_2 = 2 \end{cases}$

精确解 $\mathbf{x_1=200000/399999=0.50000125}$

$\mathbf{x_2=199998/199999=0.999995}$

用四位浮点十进制数进行消去法求解，可表示成

$$\begin{cases} 10^{-4} \bullet 0.1000 x_1 + 10^{1} \bullet 0.1000 x_2 = 10^{1} \bullet 0.1000 \\ 10^{1} \bullet 0.2000 x_1 + 10^{1} \bullet 0.1000 x_2 = \end{cases}$$

在做减法时消失

用1/2(10^{-4}*0.1000)除第一方程减第二方程，

$$\begin{cases} 10^{-4} \bullet 0.1000 x_1 + 10^{1} \bullet 0.1000 x_2 = 10^{1} \bullet 0.1000 \\ 10^{6} \bullet 0.2000 x_2 = 10^{6} \bullet 0.2000 \end{cases}$$

由此解出x_1=0, x_2=10^1*0.1000=1

2. 避免相近二数相减

例：$a_1 = 0.12345$，$a_2 = 0.12346$，各有5位有效数字。

而 $a_2 - a_1 = 0.00001$，只剩下1位有效数字。

$$\sqrt{x+\varepsilon} - \sqrt{x} = \frac{\varepsilon}{\sqrt{x+\varepsilon} + \sqrt{x}}; \qquad \ln(x+\varepsilon) - \ln x = \ln\left(1 + \frac{\varepsilon}{x}\right);$$

当 $|x| << 1$ 时：$$1 - \cos x = 2\sin^2\frac{x}{2};$$

$$e^x - 1 = x\left(1 + \frac{1}{2}x + \frac{1}{6}x^2 + \ldots\right)$$

3. 避免大数吃小数

例：用单精度计算 $x^2-(10^9+1)x+10^9=0$ 的根。

精确解为 $x_1=10^9$，$x_2=1$

✍ 算法1：利用求根公式 $x=\dfrac{-b\pm\sqrt{b^2-4ac}}{2a}$

在计算机内，10^9存为0.1×10^{10}，1存为0.1×10^1。做加法时，两加数的指数先向大指数对齐，再将浮点部分相加。即1的指数部分须变为10^{10}，则：$1 = 0.0000000001\times10^{10}$，取单精度时就成为：

$10^9+1=0.10000000\times10^{10}+0.00000000\times10^{10}=0.10000000\times10^{10}$

$$x_1=\frac{-b+\sqrt{b^2-4ac}}{2a}=10^9,\quad x_2=\frac{-b-\sqrt{b^2-4ac}}{2a}=0$$

✍算法2：先解出 $x_1 = \dfrac{-b - sign(b)\cdot\sqrt{b^2 - 4ac}}{2a} = 10^9$

再利用 $x_1 \cdot x_2 = \dfrac{c}{a} \quad\Rightarrow\quad x_2 = \dfrac{c}{a \cdot x_1} = \dfrac{10^9}{10^9} = 1$

注：求和时从小到大相加，可使和的误差减小。

例：按从小到大、以及从大到小的顺序分别计算

$$1 + 2 + 3 + \ldots + 40 + 10^9$$

4. 先化简再计算，减少步骤，避免误差积累。

一般来说，计算机处理下列运算的速度为 $(+\,,-) > (\times\,,\div) > (\exp)$

5. 选用稳定的算法。

例10 计算多项式

$$P_n(x)=a_nx^n+a_{n-1}x^{n-1}+\cdots+a_xx+a_0$$

直接计算，共需乘法 $\frac{n(n+1)}{2}$ 次 和加法n次。

采用秦九韶算法

$$\begin{cases} s_n=a_n \\ s_k=xs_{k+1}+a_k(k=n-1,\ldots,2,1,0) \\ P_n(x)=s_0 \end{cases}$$

只要乘法和加法各n次。