数值计算方法

Numerical Methods

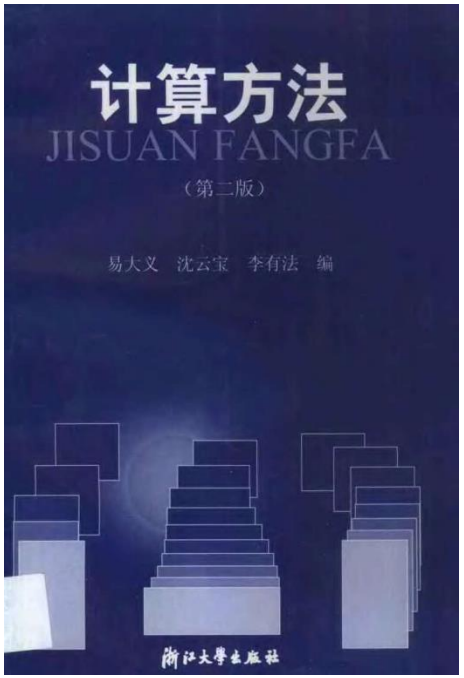
浙江大学控制学院

课程相关信息

- 教师：刘山
  - 办公室：玉泉校区控制学院老楼404
  - 电话：13958055563
  - Email：sliu@zju.edu.cn
- 助教
  - 周鹏威：15010989128，335131843@qq.com
  - 陈泱白：18072783897，22032138@zju.edu.cn
  - 崔卓凡：13521810362，22132049@zju.edu.cn
- 课程网站（“学在浙大”课程平台）
  - http://course.zju.edu.cn

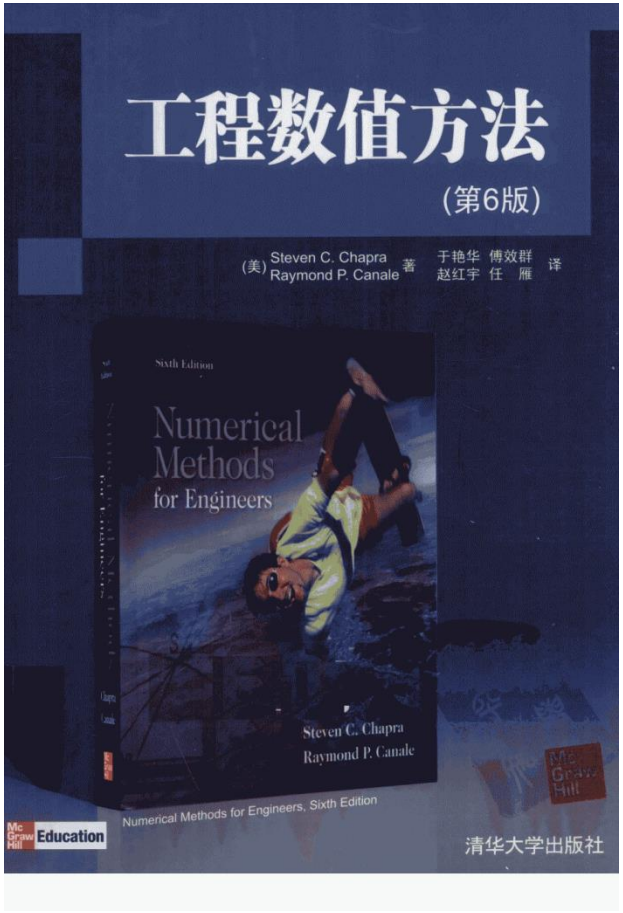
参考资料

- 易大义. 计算方法. 第二版. 浙江大学出版社， 2002



参考资料

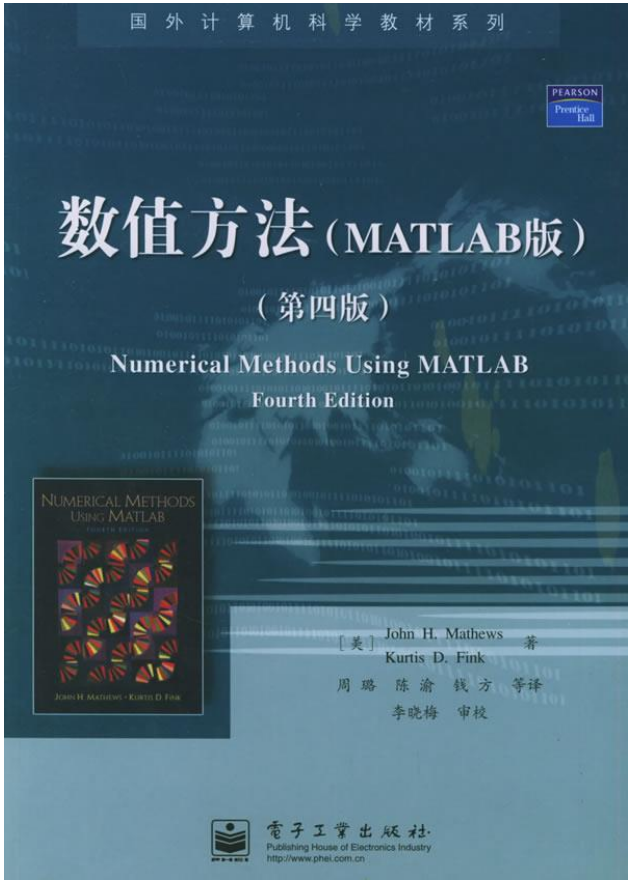
- Steven C Chapra等著，唐玲艳等译. 工程数值方法. 第五版. 清华大学出版社，2007
- Steven C Chapra等著，于艳华等译. 工程数值方法. 第六版. 清华大学出版社，2010



参考资料

- John H Mathews等著，周璐等译. 数值方法(Matlab版). 第四版. 电子工业出版社，2005

  (http://mathfaculty.fullerton.edu/mathews/numerical.html)



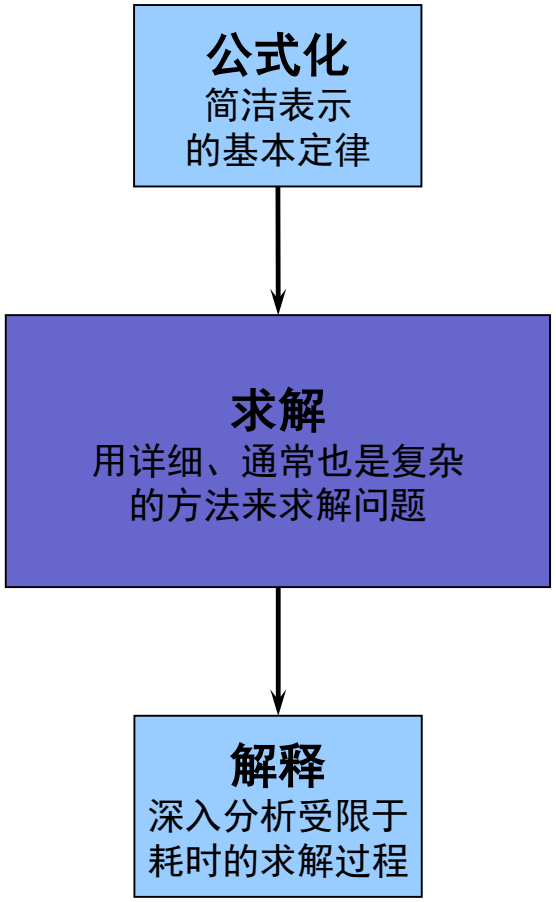
第一章 绪论

- 课程简介
  - 什么是数值计算方法？
  - 为什么学习数值计算方法？
  - 数值计算方法的主要内容
- 数值计算中的误差
  - 误差的种类及其来源
  - 绝对误差与相对误差
  - 有效数字与误差
  - 舍入误差与截断误差
  - 误差的传播与估计
  - 算法的数值稳定性

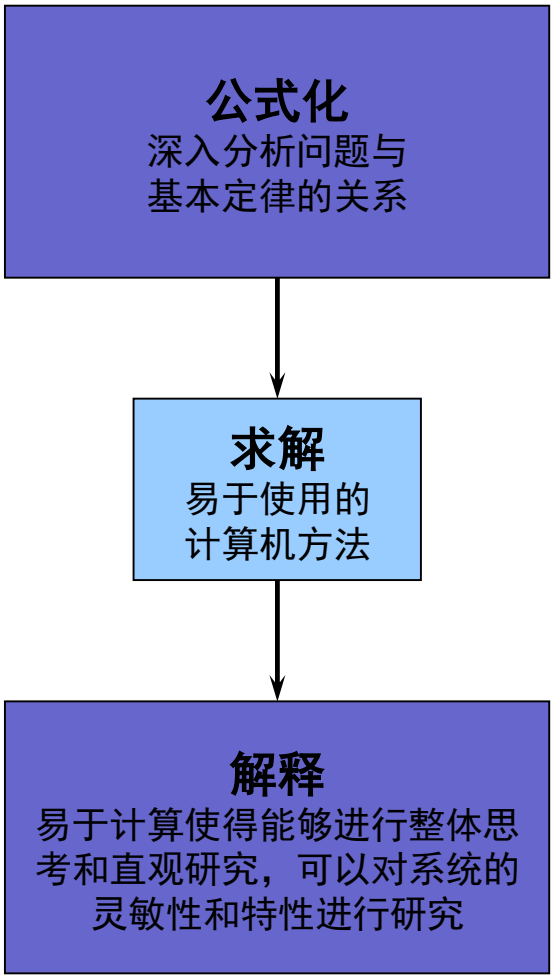
非计算机方法

- 解析方法
  - 简单问题
  - 实际价值有限
- 图解法
  - 结果准确?
  - 三维及以下
- 手工方法
  - 计算器
  - 速度慢，很容易出现低级错误

工程问题求解的三个阶段



计算机时代到来之前



计算机时代

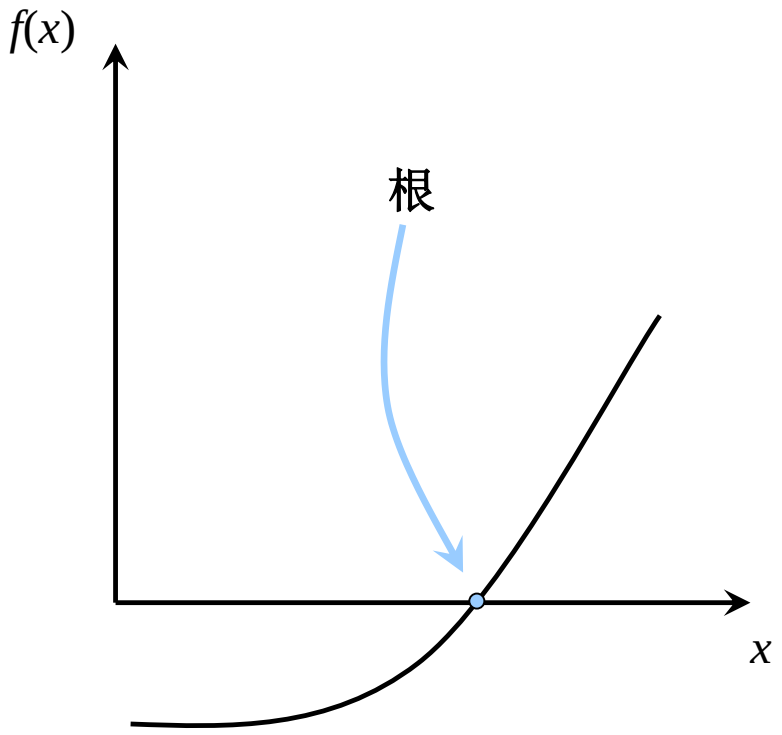
学习数值计算方法的意义

- 增强求解问题的技能
- 在理解的基础上使用一些商品化软件
- 解决一些软件所不能解决的问题
- 学习使用计算机的一个有效载体
- 加深对数学的理解

数值计算方法的内容（一）

- 方程求根（Roots of Equation）

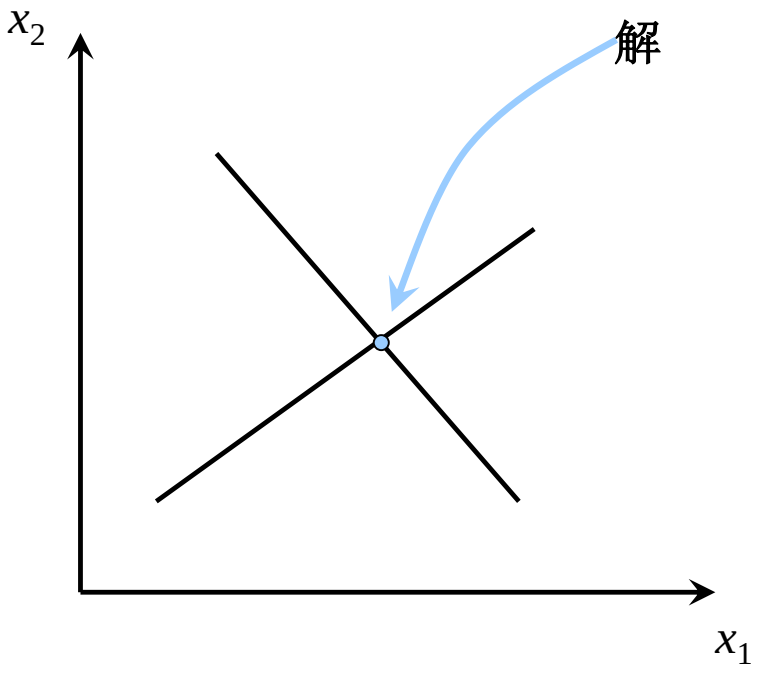
$$f(x)=0$$



数值计算方法的内容（二）

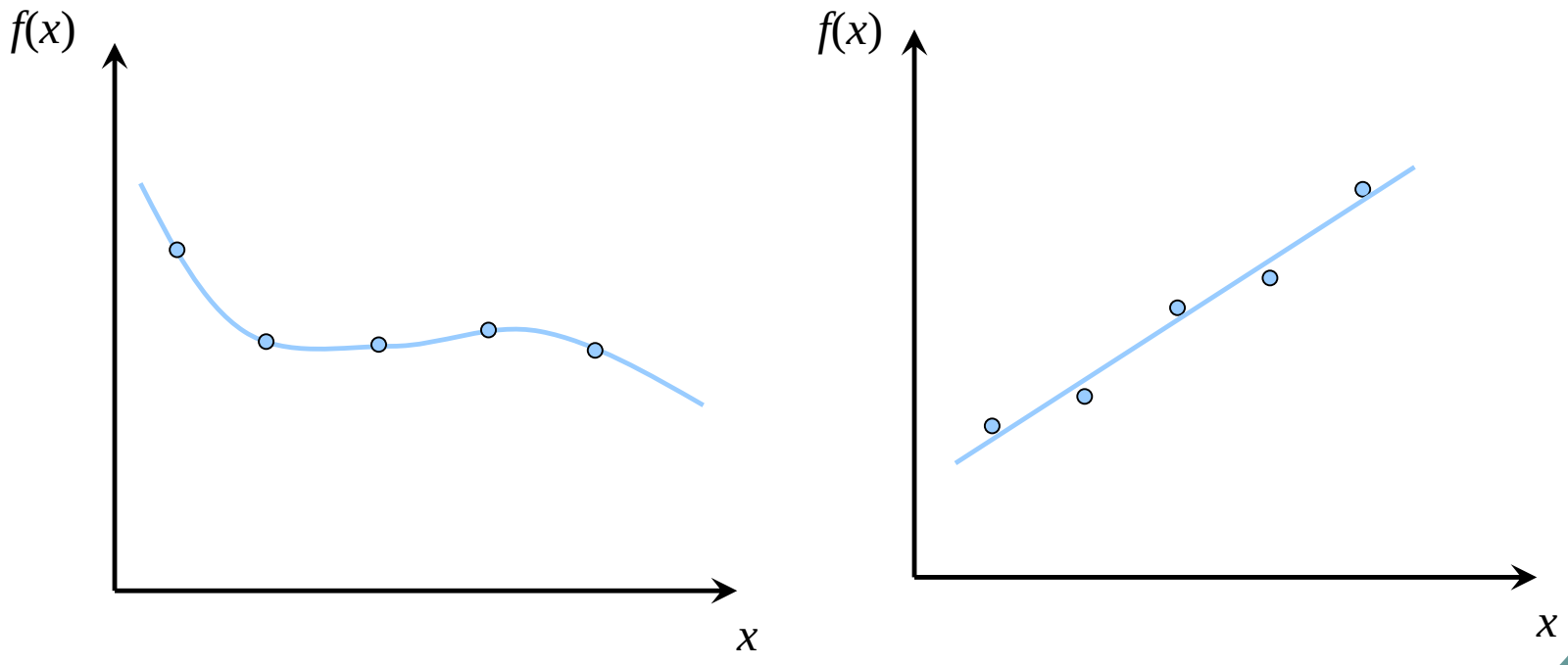
- 线性代数方程组（System of Linear Algebraic Equations）

$$a_{11}x_1 + a_{12}x_2 = c_1$$
$$a_{21}x_1 + a_{22}x_2 = c_2$$



数值计算方法的内容（三）

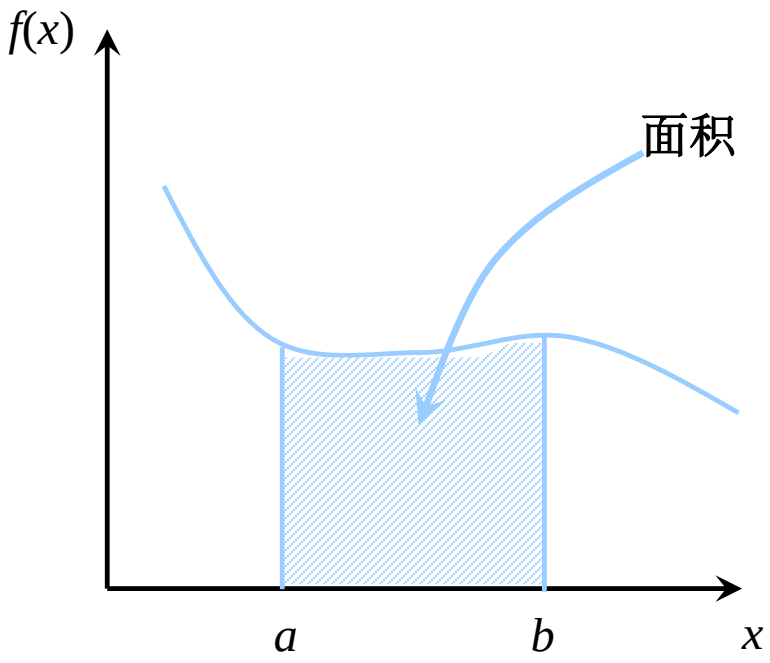
- 插值（Interpolation）和曲线拟合（Curve Fitting）



数值计算方法的内容（四）

- 数值微分（numerical differentiation）和数值积分（numerical integration）

$$I = \int_a^b f(x)dx$$



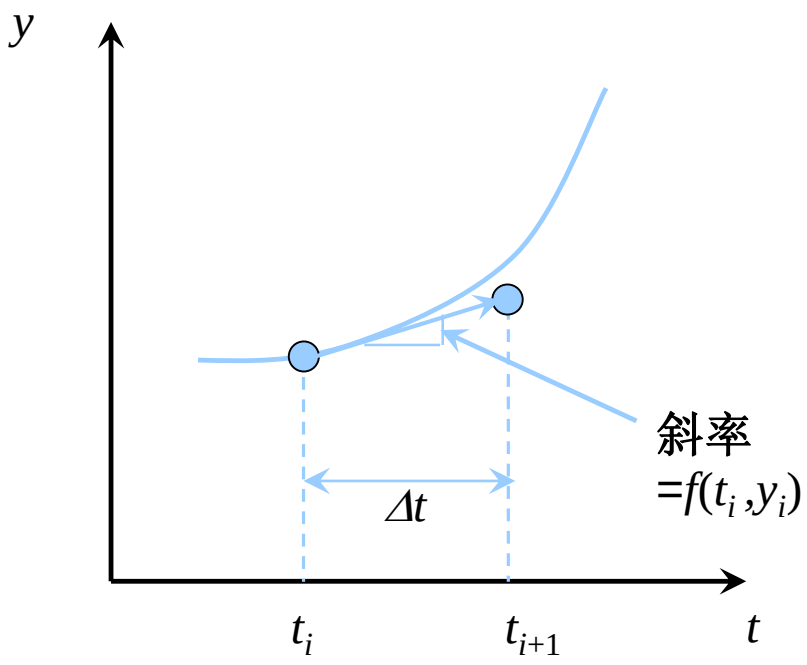
数值计算方法的内容（五）

- 常微分方程（Ordinary Differential Equation ）

$$\frac{dy}{dt} \approx \frac{\Delta y}{\Delta t} = f(t, y)$$

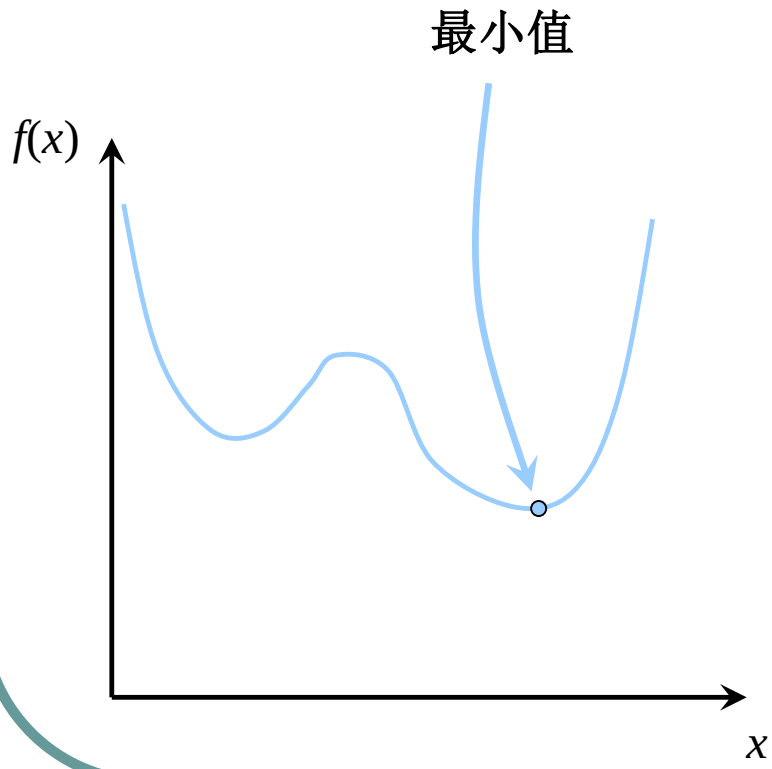
求关于自变量t的函数y

$$y_{i+1} = y_i + f(t_i, y_i)\Delta t$$

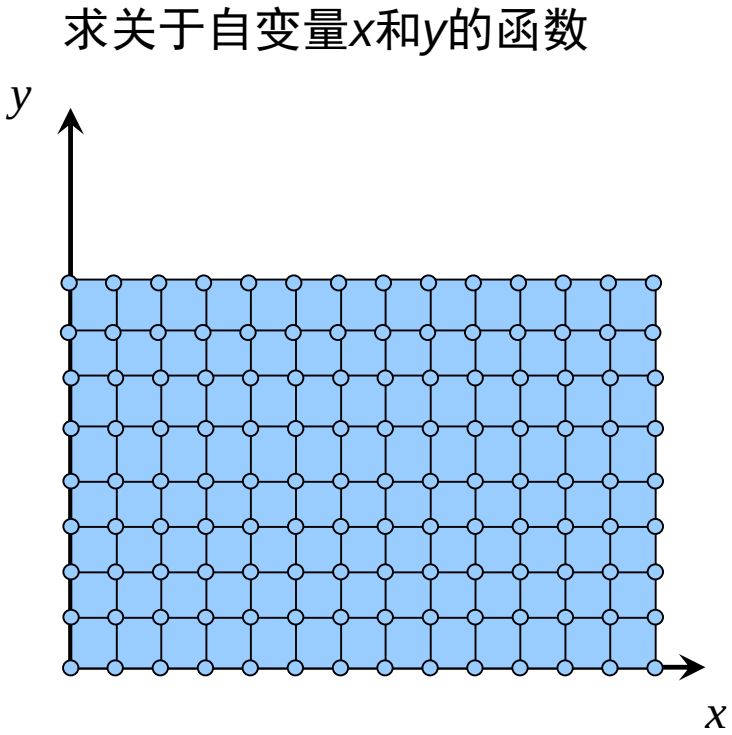


数值计算方法的内容（六）

- 最优化（Optimization）



- 偏微分方程（Partial Differential Equation）



课堂教学内容

- 绪论
- 非线性方程求根
- 求解线性方程组的数值方法
- 插值和曲线拟合
- 数值微分和数值积分
- 常微分方程数值解
- 最优化
- 矩阵特征值与特征向量

教学安排

- 理论
  - 周二13:15～15:40
- 上机（助教负责）
  - 周二15:55～17:30
  - Matlab介绍一次（第1周，钉钉群直播）
  - 上机作业共四次（第2-5周），需到机房刷卡签到。

春学期							
月		三月				四	
一	二	三	四	五	六	七	八
21	28	7	14	21	28	4	11
22	1	8	15	22	29	5	12
23	2	9	16	23	30	6	13
24	3	10	17	24	31	7	14
25	4	11	18	25	1	8	15
26	5	12	19	26	2	9	16
27	6	13	20	27	3	10	17

考核方式

- 平时成绩（70%）
  - 课堂（5%）+上机（20%）
  - 作业（45%）
  - 提交时间：注意截止时间
  - 提交到课程网站，程序+报告文档，一个压缩文件
- 期末作业（30%）
  - 自选题目
    - 利用数值方法解决问题
      - 专业领域、以往课程、社会生活
      - 解析解？
      - 数值解？
      - 分析

上机作业要求

- 程序
- 运行结果
- 心得体会

作业要求

- **完整性：**源程序+文档
- **报告要求：**
  - 问题分析
  - 算法求解讨论
  - 程序需有必要的注释
  - 结果及分析
- **排版格式：**清晰、美观
- 网站上有
  - 课程课件
  - 作业示例
  - 大作业范例
- 必须遵守学术道德规范，鼓励相互之间进行讨论，但提交的作业必须独立完成。
- 杜绝抄袭！

作业题例

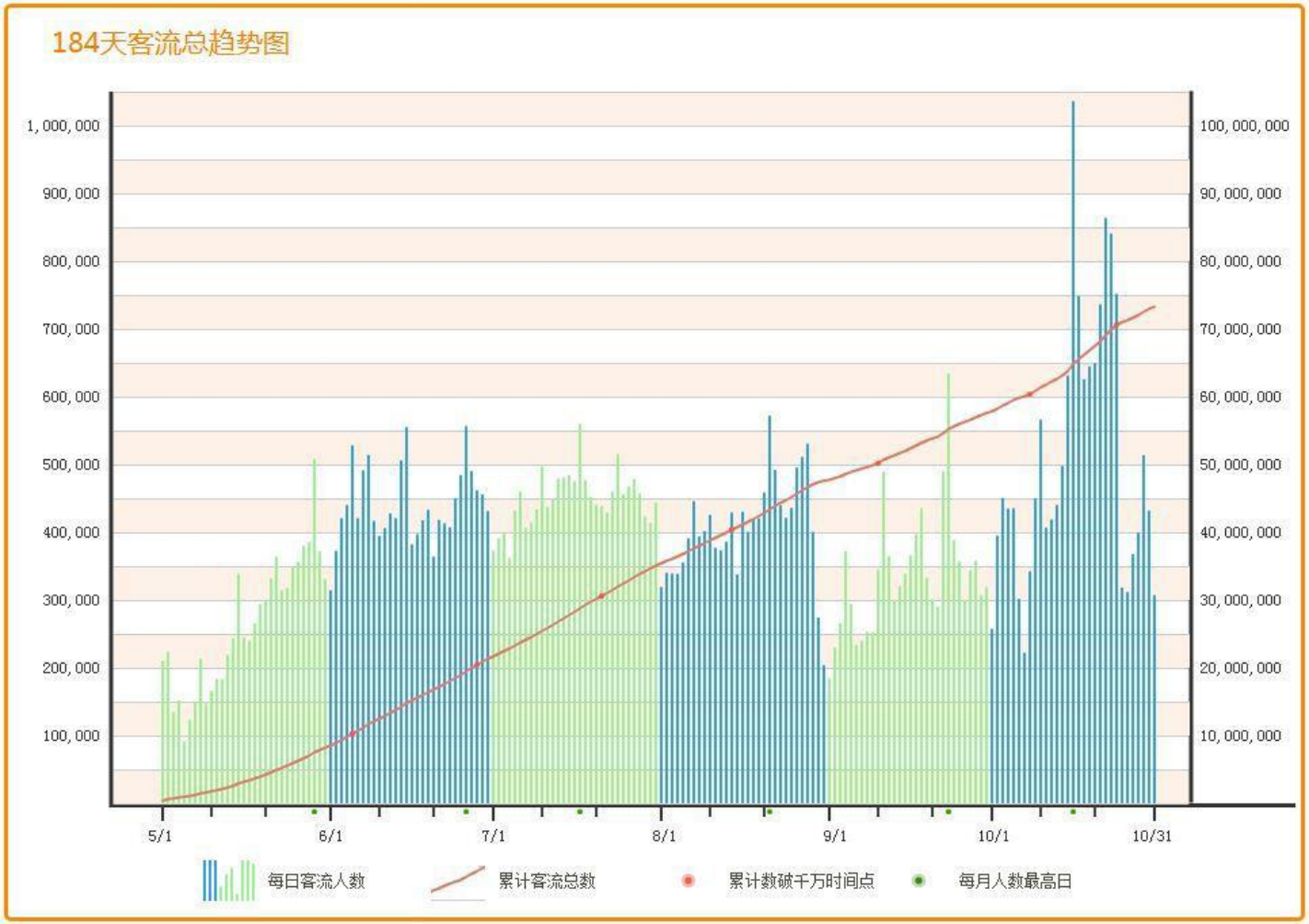
- 在对汽车、机器人以及其他运动器械的控制时，经常需要获得其移动距离的数据
- 通过对于一连串采样点（即时速度）的插值，得到速度关于时间的函数，再对于速度进行积分，即可得到对象所运动的距离

总长度为5451. 24米

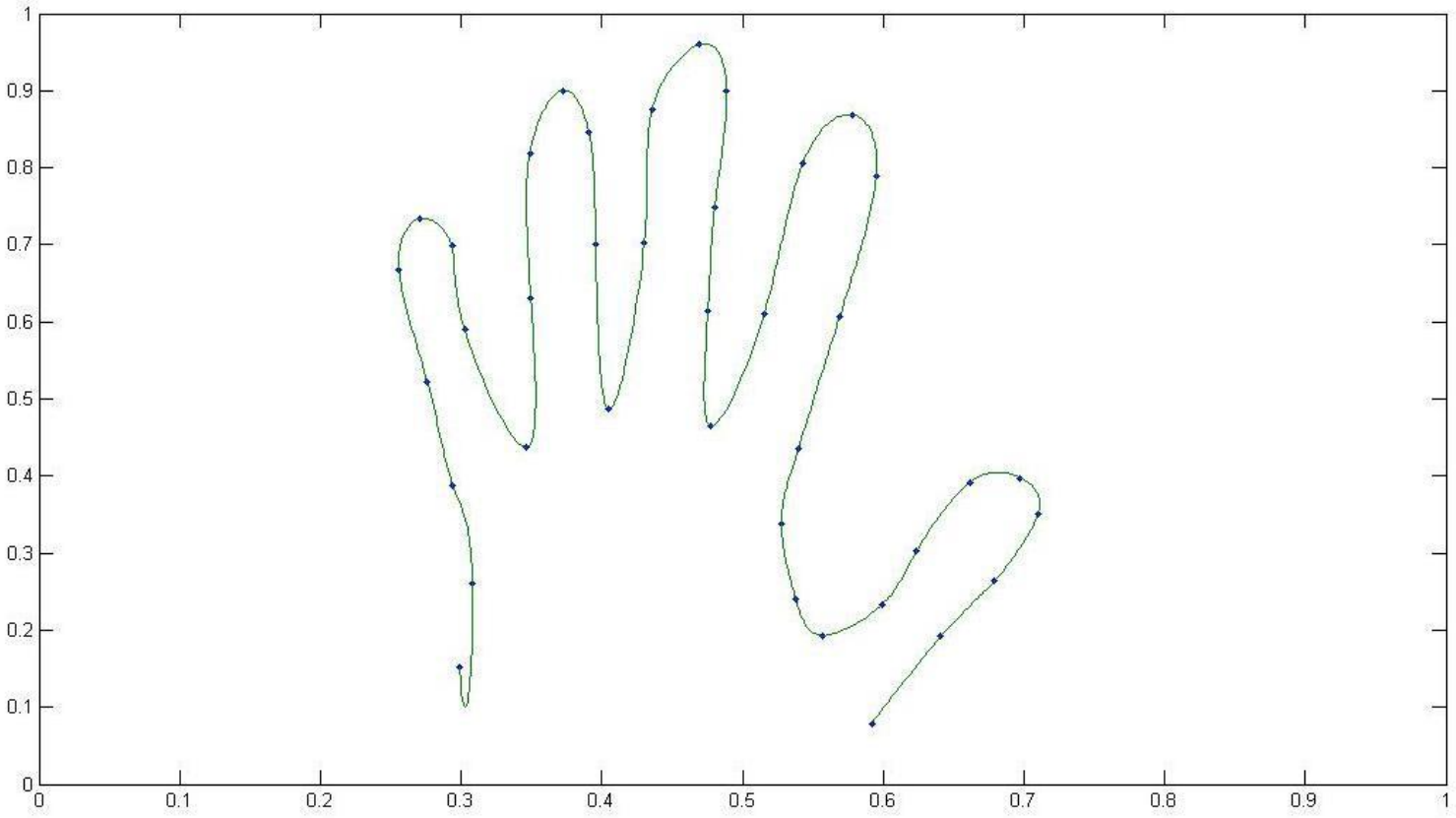
作业题例——西湖面积


总长：1.3公里
总长：1.4公里
总长：288米
总长：146米
总长：700米
总长：350米
起点
北湖
里西湖
西湖
曙光路
北山路
杨公堤
苏堤
延安路
中河中路
中河高架
西湖大道
庆春路
西湖国宾馆
浙江宾馆
吴山景区
上城区政府
正阳酒店
五洋假日酒店
吴山酒楼大酒店
长桥公园
湖滨公园
中环宾馆
中良大厦
清波商厦
定安东苑
龙禧湖滨饭店
西湖铭楼
竹竿浙报公寓
五公园码头
通诚商务楼
里约酒店杭州西湖店
杭州基督教女青年会
金鼓洞
抱朴道院
宝石流霞
保俶塔
挂牌山
如意洞
黄源旧居
普提精舍
栖霞山庄
木兰山茶园
曲院风荷
孤山公园
平湖秋月
西泠书画院
金溪山庄
玉带桥
湖心亭
柳浪新苑
三潭印月
夕佳亭
盖叫天墓
白云庵
花港观鱼
雷峰夕照
西湖
俞曲园墓
花港海航度假酒店
杭州市职

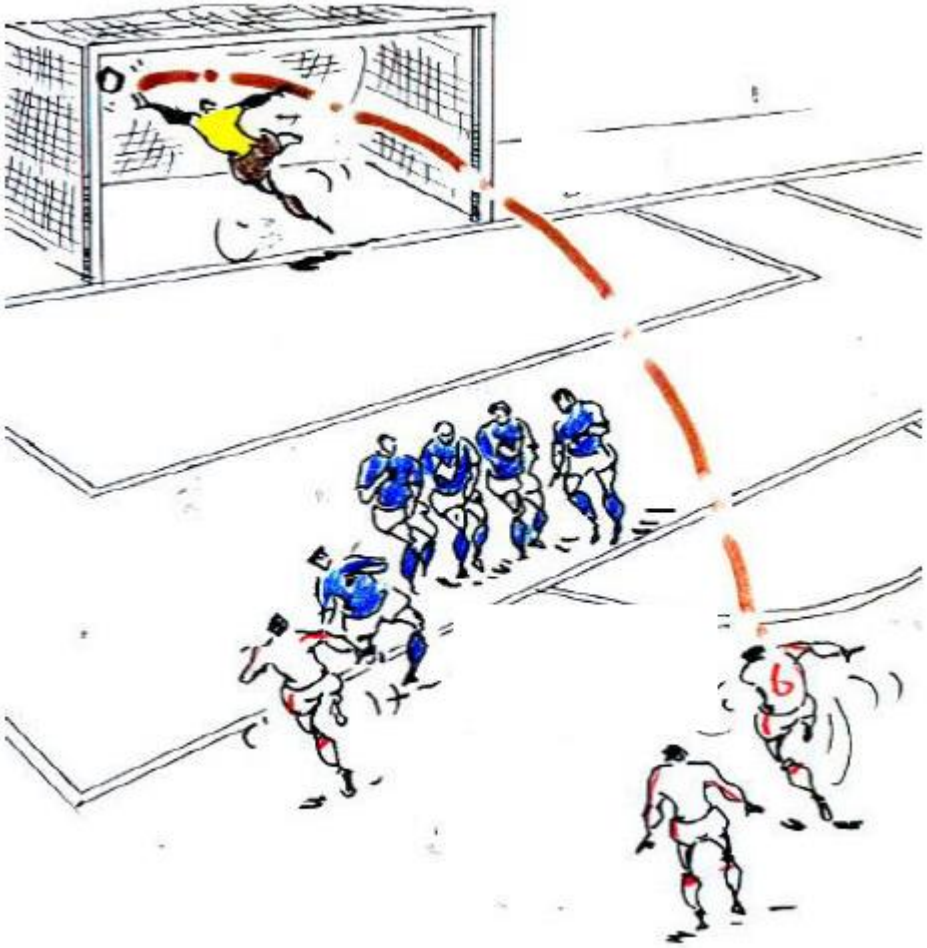
作业题例——上海世博会参观人数过千万时间点



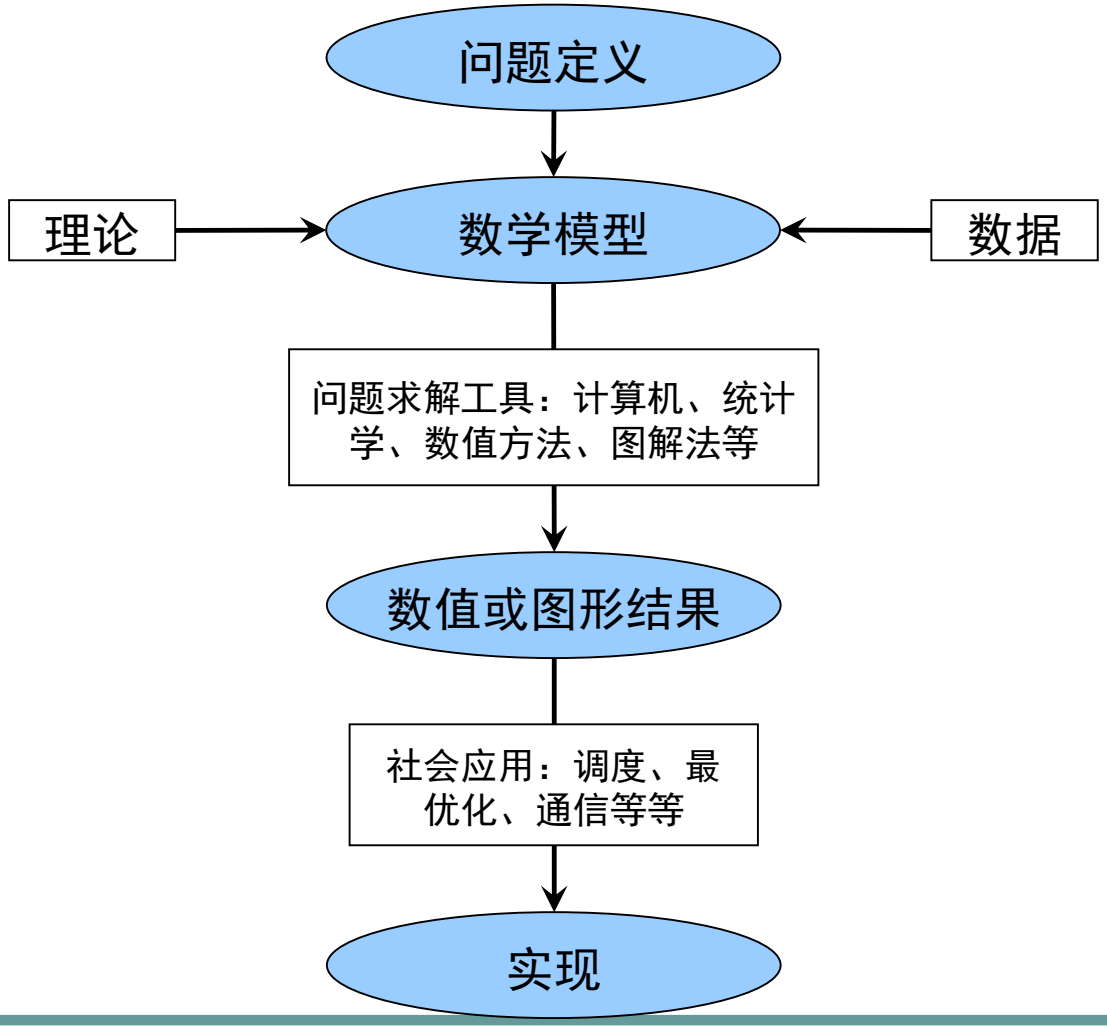
作业题例——手



作业题例——任意球



数学建模与工程问题求解



数学模型（Mathematical Model）

- 用数学语言来表达物理系统或过程本质特征的公式或方程。
  - 因变量=*f*(自变量，参数，强制函数)
    - 因变量（dependent variable）：用来刻画系统行为或状态的特征量；
    - 自变量（independent variable）：通常为维度，如时间和空间，系统的行为是用自变量来确定的；
    - 参数（parameter）：反映系统的性质或组成；
    - 强制函数（forcing function）：外部对系统施加的影响。

一个简单的数学模型——牛顿第二定律

- $F=ma \Rightarrow a=F/m$
  - a为因变量，表示了系统的行为；
  - F为强制函数；
  - m为表示系统性质的参数；
  - 在这种简单的情况下，没有自变量，没有涉及到加速度a在随时间或空间的变化。
- 物理世界中数学模型的几个典型特点：
  - 用数学语言描述自然过程或系统；
  - 代表了一种理想情况，或是对现实的简化。即忽略了自然过程的不重要的细节，而集中于自然过程的本质特征；
  - 得到的是一个可以重现的结果，可以用来做预测。

降落伞问题（Falling Parachutist Problem）

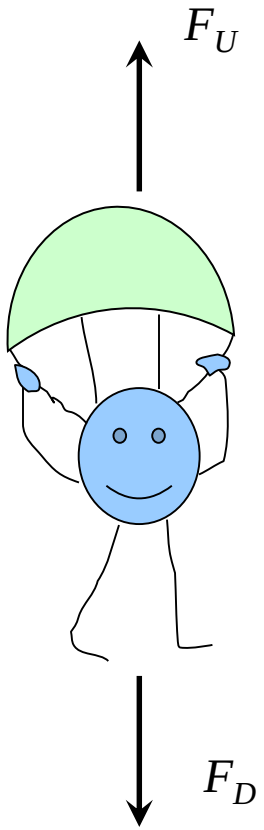
- 确定降落伞的最后速度
  - 加速度表示为速度的变化率

$$\frac{dv}{dt} = \frac{F}{m}$$

  - 如果净受力为正，物体加速运动；如果为负，物体减速运动；如果为0，物体速度不变。
  - 假定向下的力为正，

$$F_D = mg \qquad F_U = -cv$$

    - c为比例系数，称为阻力系数（drag coefficient(kg/s)）。参数c说明了下降物体的特征，如形状或表面的粗糙程度。



降落伞问题（Falling Parachutist Problem）

- 模型——微分方程（differential equation）

$$\frac{dv}{dt}=\frac{mg-cv}{m}=g-\frac{cv}{m}$$

  - 若初始时处于静止状态（t=0时，v=0），可得

$$v(t)=\frac{gm}{c}\left(1-e^{-(c/m)t}\right)$$

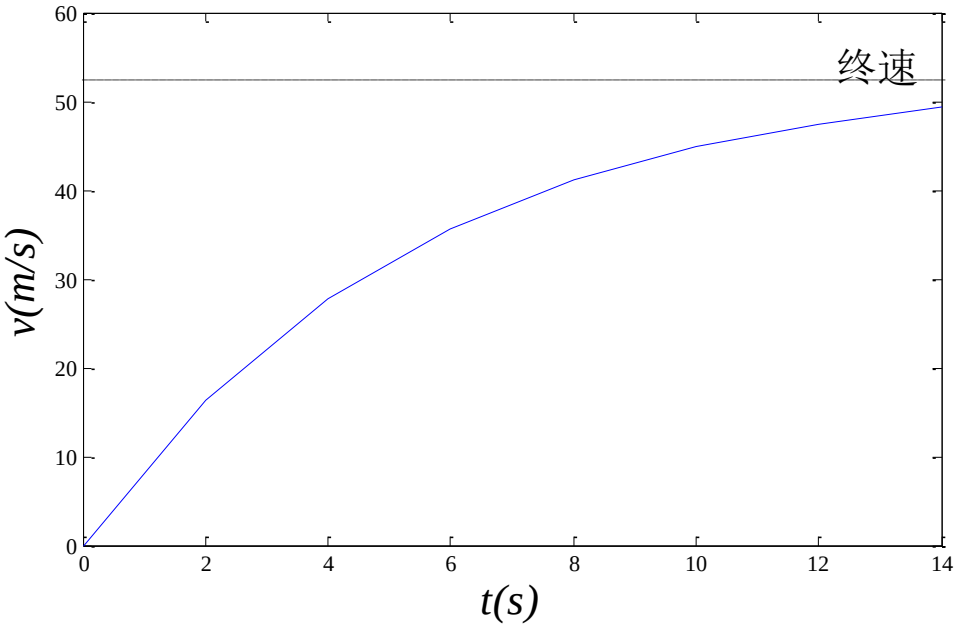
$v(t)$为因变量；t为自变量；c和m为参数；g为强制函数。

  - 问题：从静止热气球中弹出一个质量为68.1kg的降落伞，阻力系数等于12.5kg/s

$$v(t)=\frac{9.8\times 68.1}{12.5}\left(1-e^{-(12.5/68.1)t}\right)=53.39\left(1-e^{-0.18355t}\right)$$

降落伞问题（Falling Parachutist Problem）

$t(s)$	$v(m/s)$
0	0.00
2	16.40
4	27.77
6	35.64
8	41.09
10	44.87
12	47.48
∞	53.39



- 解析解或精确解
- 但许多模型不能精确地求解
- 数值方法——对数学问题进行变换，使得它可以用算术运算来求解

降落伞问题（Falling Parachutist Problem）

- 数值方法：

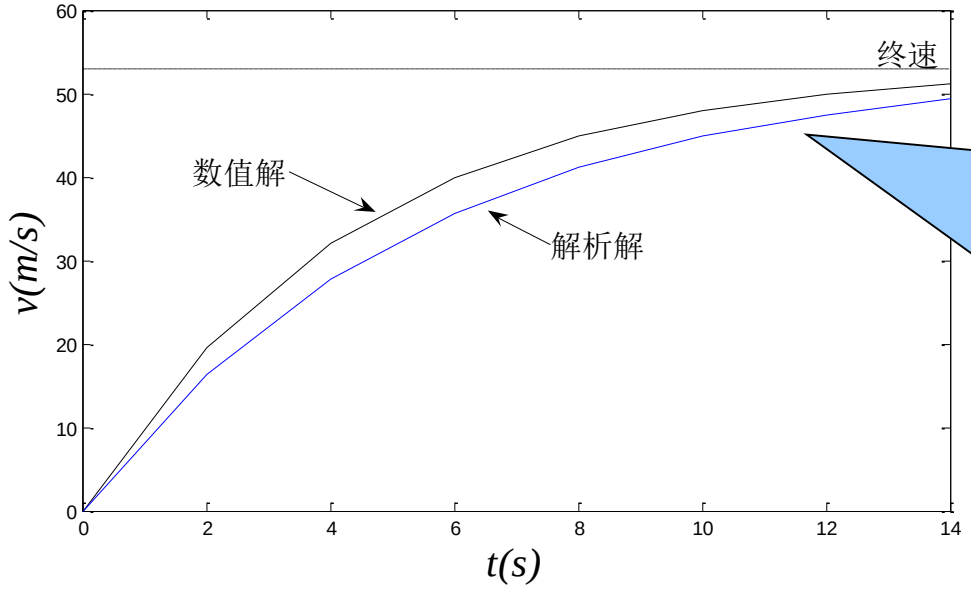
$$\frac{dv}{dt}=\lim_{\Delta t\to 0}\frac{\Delta v}{\Delta t}\quad\Rightarrow\quad\frac{dv}{dt}\approx\frac{\Delta v}{\Delta t}=\frac{v(t_{i+1})-v(t_i)}{t_{i+1}-t_i}$$

  - 在时刻t_i处导数的有限差商近似

$$v(t_{i+1})=v(t_i)+\left[g-\frac{c}{m}v(t_i)\right](t_{i+1}-t_i)$$

欧拉法

$t(s)$	$v(m/s)$
0	0.00
2	19.6
4	32.00
6	39.86
8	44.82
10	47.97
12	49.96
∞	53.39



用直线段来近似表示连续的曲线函数，因此两者之间存在一定差异。可以用更小的步长使误差减小——计算量与精度之间的**权衡**

数值计算方法

- **数值计算方法是一门根据计算机特点，研究通过计算机求工程问题满足精度要求的近似解的学科。**
- 将所欲求解的数学模型（数学问题）简化成一系列算术运算和逻辑运算，以便在计算机上求出问题的数值解，并对算法的收敛性、稳定性和误差进行分析、计算。

选择算法的三个重要依据

算法

- 基本运算和运算顺序的规定所组成的整个解题方案和步骤。
- 一般可以通过框图（流程图）来较直观地描述算法的全貌。
- 选择合适的算法是整个数值计算中非常重要的一个环节。

算法的选择

- 例：计算多项式的值

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

  - 直接计算 $a_i x^i (i=0,1,2,\cdots,n)$ ，再逐次相加，共需

$$1+2+\cdots+(n-1)+n=\frac{n(n+1)}{2}$$

次乘法和n次加法。

  - 秦九韶（我国宋朝数学家）算法（1247年）

$$P(x)=((\cdots((a_n x+a_{n-1})x+a_{n-2})x+\cdots+a_2)x+a_1)x+a_0$$

只要n次乘法和n次加法即可。（霍纳Horner1819年）

算法

- 算法的优劣直接影响计算的速度和效率
  - 对于小型问题，计算速度和占用计算机内存的多寡似乎意义不大，但对复杂的大型问题却起着决定性作用。
- 不合适的算法还会由于计算机计算的近似性和误差的传播、积累直接影响到计算结果的精度，甚至影响到计算的成败。

——算法的数值稳定性

第一章 绪论

- 课程简介
  - 什么是数值计算方法？
  - 为什么学习数值计算方法？
  - 数值计算方法的主要内容
- **数值计算中的误差**
  - **误差的种类及其来源**
  - **绝对误差与相对误差**
  - **有效数字与误差**
  - 舍入误差与截断误差
  - 误差的传播与估计
  - 算法的数值稳定性

误差的种类

- 舍入误差（round-off error）
  - 由于计算机只能表示有限位数的量而引起的
- 截断误差（truncation error）
  - 由于数值方法可能运用近似方法表示准确数值运算或数量而引起的
- 不与数值方法本身直接相关的误差
  - 粗差
  - 形式化或模型误差
  - 数据不确定性误差

有效数字（significant digit）

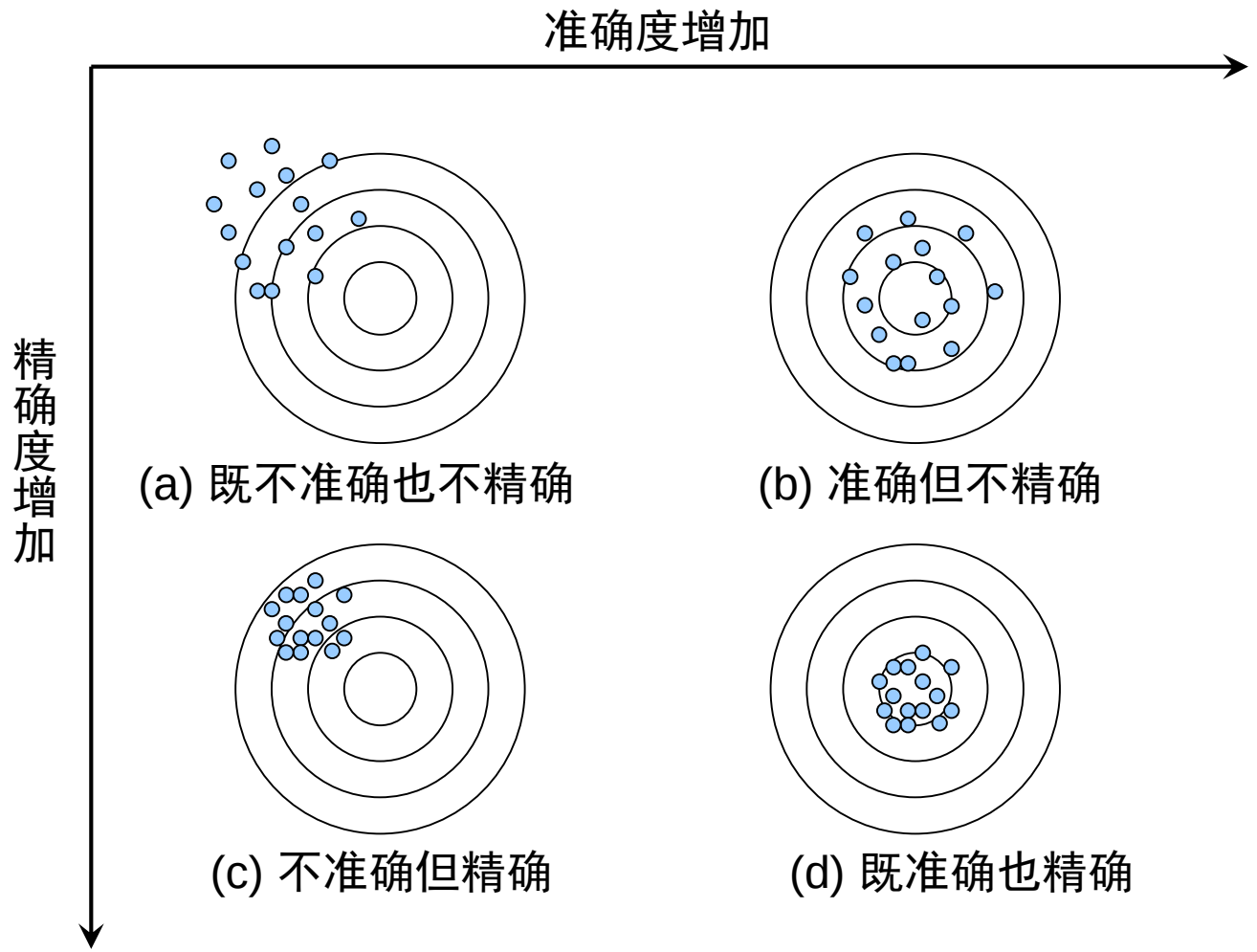
- 为了正式规定数值的可靠程度
- 一个数的有效数字是指可以放心使用的那些数字
- 有效数字对应于确定数字和一个估计数字的总位数
  - 数值方法得到的是近似解，所以必须建立准则来规定近似结果的可信度。建立准则的一种方法是基于有效数字实现的，例如：如果有4位有效数字是正确的，可能就认定得到的近似结果是可以接受的。
  - 计算机只能保留有限个有效数字。保留有效数字后被省略的部分称为舍入误差。

如： $\pi = 3.14159265358979323846264 3\cdots$

准确度（accuracy）与精确度（precision）

- 准确度
  - 计算值或测量值与真值接近的程度
- 精度或精确度
  - 各计算值或测量值相互之间的集中程度

准确度（accuracy）与精确度（precision）——射击的例子



误差的定义

- 数值误差源于用近似方法表示准确的数学运算和准确的数量
- 数值误差包括截断误差和舍入误差
  - 当用近似方法表示准确数学过程时会出现截断误差
  - 当用有限个有效数字表示准确的数时会引起舍入误差
- 绝对误差
  - 真值=近似值+误差 ⟹ 误差=真值-近似值

$$E = x - x^*$$

绝对误差可正可负，一般E的准确值很难求出。可用真值的最优估计值代替真值，或者给出误差的一个上界$\delta(x^*)$（误差限）。

- 相对误差

$$\varepsilon_r = \frac{E}{x} \times 100\%$$

（1）当x=0时，相对误差无意义。
（2）准确值x往往未知，故常用x^*代替x，相对误差限为$\delta(x^*)/|x^*|$。

误差计算——例1

- 问题：假设对一座桥梁和一个铆钉的长度进行了测量，测量的结果分别为9999cm和9cm。如果真值分别为10000cm和10cm，计算两种情况下的真误差和真相对误差
- 解：(a)桥梁的真误差为 $E_t = 10000 - 9999 = 1cm$

  铆钉的真误差为 $E_t = 10 - 9 = 1cm$

  (b)桥梁的相对误差为 $\varepsilon_r = \frac{1}{10000} \times 100\% = 0.01\%$

  铆钉的相对误差为 $\varepsilon_r = \frac{1}{10} \times 100\% = 10\%$

尽管两个绝对误差都是**1cm**，但测量铆钉的相对误差要大得多，因此，我们对桥梁的测量已经够准确了，但对铆钉的估计在某种程度上偏离了希望值。

误差——例2

- 已知π=3.1415926…，若取近似数为 $x^* = 3.14$，则 $\varepsilon = \pi - x^* = 0.0015926\ldots$，$|\varepsilon| \le 0.002 = \delta(x^*)$ 为 x^* 的误差限，相对误差限为

$$\delta_r(x^*) = \frac{\delta(x^*)}{x^*} < 0.007$$

- 通常在x有多位数字时，若取前有限位数的数字作为近似值，都采用四舍五入原则，他们的误差限都不超过近似数x^*末位数的半个单位，即

$$|\pi - 3.14| \le \frac{1}{2} * 10^{-2} \qquad |\pi - 3.1416| \le \frac{1}{2} * 10^{-4}$$

有效数字与误差的关系

- 定义：设x^*是x的一个近似数，表示为$x^* = \pm 10^k \times 0.a_1 a_2 \cdots a_n$，其中每个$a_i$（$i$=1，2，…，$n$）均为0，1，2，…，9中的一个数字，且$a_1 \neq 0$，如果

$$\left| x - x^* \right| \leq \frac{1}{2} * 10^{k-n}$$

则称x^*近似x有n位有效数字。

近似数的有效数字越多，相对误差限就越小，反之亦然。

- 如果x^*具有n位有效数字，则其相对误差限为

$$\delta_r(x^*) \leq \frac{1}{2a_1} \times 10^{-(n-1)}$$

可用于估计相对误差

迭代方法的误差估计

- 当前迭代结果的误差：

$$\varepsilon_a = \frac{\text{当前近似值} - \text{前一近似值}}{\text{当前近似值}} \times 100\%$$

- 采用绝对值　　（不考虑正负号）
- 是否小于预先设定好的容限ε_s

$$|\varepsilon_a| < \varepsilon_s$$

- 如果下面的准则成立，则可以保证至少n位有效数字是正确的：

$$\varepsilon_s = \left(0.5 \times 10^{2-n}\right)\% = 0.5 \times 10^{-n}$$

非常有用的公式，可用于设定误差容限

迭代方法的误差估计——例

- 问题：指数函数可以用下式表示（麦克劳林级数展开）

$$e^x = 1 + x + \frac{x^2}{2} + \frac{x^3}{3!} + \cdots + \frac{x^n}{n!}$$

- 计算 $e^{0.5}$ 的值，每加入一个新项后计算真误差和近似百分比相对误差。真值为 $e^{0.5} = 1.648721\cdots$，直到近似误差小于 ε_s，ε_s 必须符合3位有效数字的要求。
- 解：误差准则：$\varepsilon_s = (0.5 \times 10^{2-3})\% = 0.05\%$

项数	结果	ε_t(%)	ε_a(%)
1	1	39.3	
2	1.5	9.02	33.3
3	1.625	1.44	7.69
4	1.645833333	0.175	1.27
5	1.648437500	0.0172	0.158
6	**1.6486**97917	0.00142	0.0158

第一章 绪论

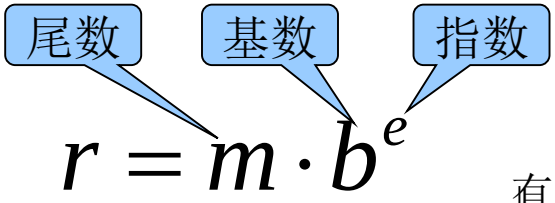
- 课程简介
  - 什么是数值计算方法?
  - 为什么学习数值计算方法?
  - 数值计算方法的主要内容
- **数值计算中的误差**
  - 误差的种类及其来源
  - 绝对误差与相对误差
  - 有效数字与误差
  - **舍入误差与截断误差**
  - 误差的传播与估计
  - 算法的数值稳定性

舍入误差

- 数的计算机表示
- 计算机中的算术运算

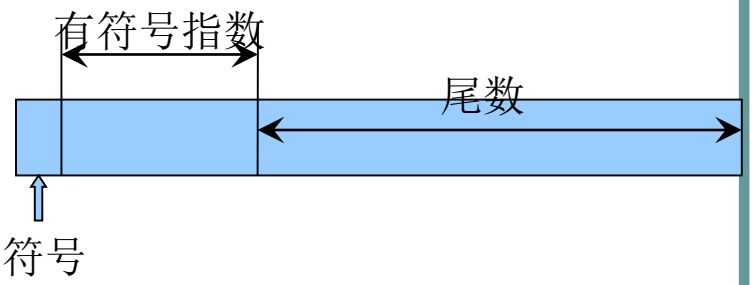
数的计算机表示

- 浮点表示



$$\frac{1}{b} \le |m| < 1$$

归一化



- 例：1/34=0.029411765…以十进制浮点形式存储，并且只准许4个十进制为可用。⟹ 0.0294×10^0

⇩

0.2941×10^{-1}可以多保存一个有效数字

➢尾数仅保存了有限的有效数字，因此，会引入舍入误差。

数的计算机表示

- 例：对于一个用7位的字存储信息的计算机，建立一个假定的浮点数集合。用数的第一位表示符号位，接着的3位表示指数的符号和大小，最后的3位存放尾数。
- 最小可能的正数为

$$0.5\times 2^{-3}=(0.0625)_{10}$$

$$0111101=(1\times 2^{-1}+0\times 2^{-2}+1\times 2^{-3})\times 2^{-3}=(0.078125)_{10}$$

$$0111110=(1\times 2^{-1}+1\times 2^{-2}+0\times 2^{-3})\times 2^{-3}=(0.093750)_{10}$$

$$0111111=(1\times 2^{-1}+1\times 2^{-2}+1\times 2^{-3})\times 2^{-3}=(0.109375)_{10}$$

相邻两数间隔为0.015625

$$0110100=(1\times 2^{-1}+0\times 2^{-2}+0\times 2^{-3})\times 2^{-2}=(0.125000)_{10}$$

$$0110101=(1\times 2^{-1}+0\times 2^{-2}+1\times 2^{-3})\times 2^{-2}=(0.156250)_{10}$$

$$0110110=(1\times 2^{-1}+1\times 2^{-2}+0\times 2^{-3})\times 2^{-2}=(0.187500)_{10}$$

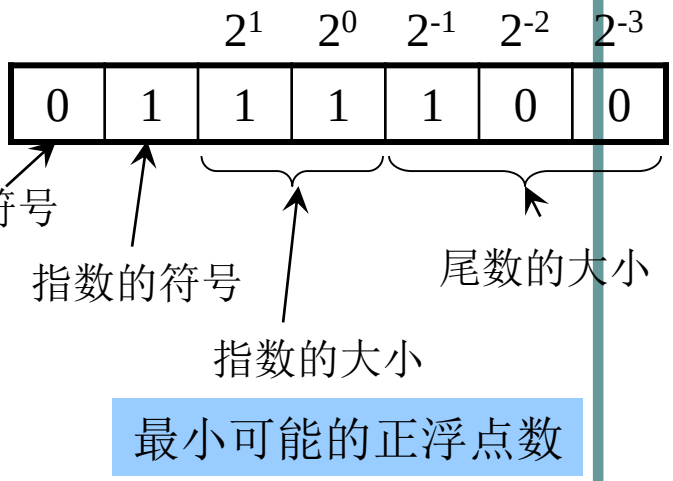
$$0110111=(1\times 2^{-1}+1\times 2^{-2}+1\times 2^{-3})\times 2^{-2}=(0.218750)_{10}$$

相邻两数间隔为0.03125

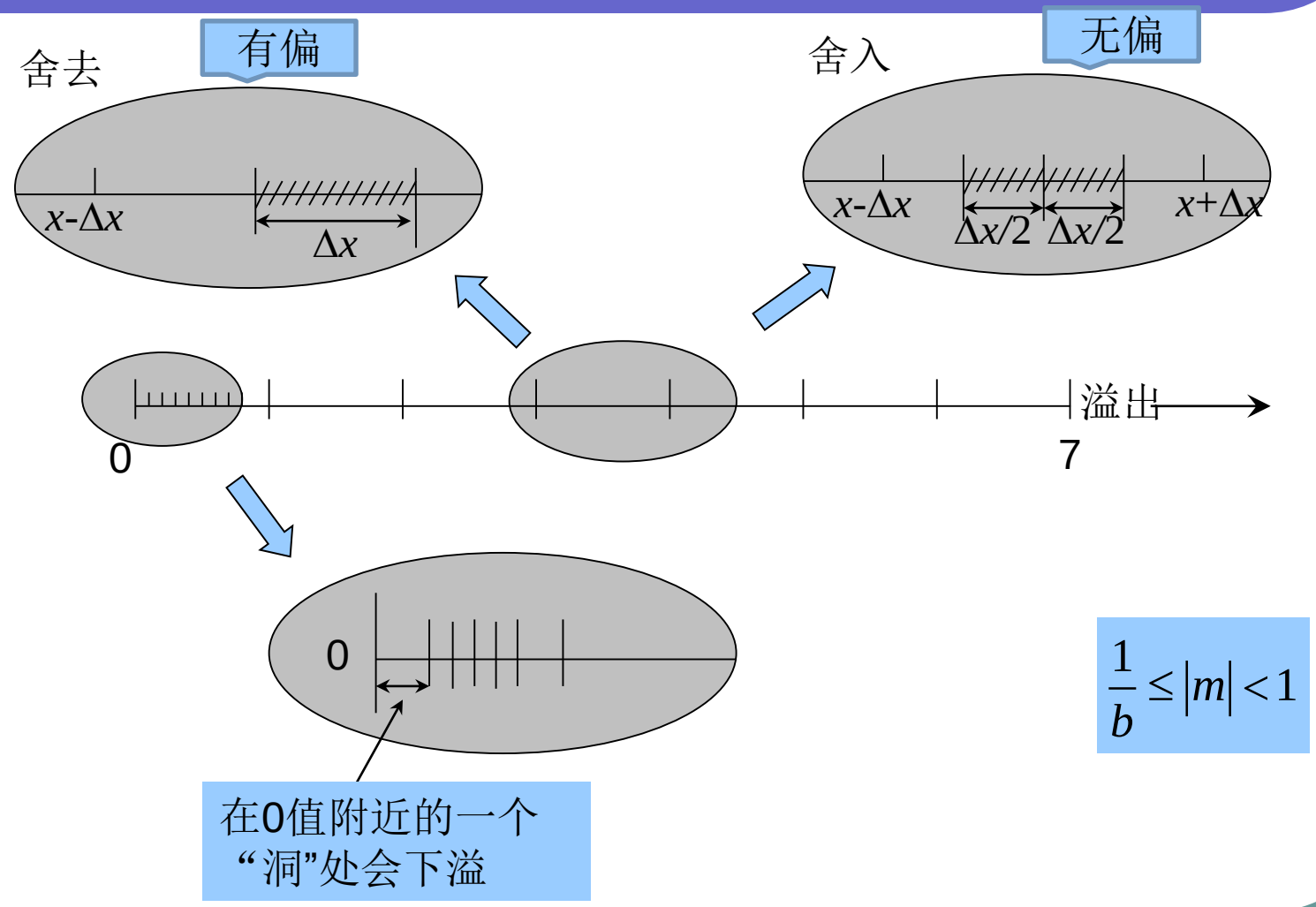
增大

$$0011111=(1\times 2^{-1}+1\times 2^{-2}+1\times 2^{-3})\times 2^{3}=(7)_{10}$$

最大的数



数的计算机表示



$$\frac{1}{b} \le |m| < 1$$

数的计算机表示

- 仅能表示有限范围内的数
  - 使用可接受范围以外的数会导致溢出。
  - 不能表示非常小的数，在0与第一个正数之间存在一个“洞”。
- 在可接受数的范围内也只能表示有限个数
  - 无理数以及不与该集合中的数对应的有理数都不可能准确地表示——量化误差
  - 舍入（rounded floating-point representation）和舍去（chopped floating-point representation）
  - 有效数字足够多的情况下，总的舍入误差通常可以忽略不计。
- 数之间的间隔随着数大小的增加而增大
  - 相对量化误差：舍去的情况 $\frac{|\Delta x|}{|x|} \le \varepsilon$ 舍入的情况 $\frac{|\Delta x|}{|x|} \le \frac{\varepsilon}{2}$
  - ε为机器精度 $\varepsilon = b^{1-t}$ （b为基数，t为尾数中有效数字的个数）

机器精度定义，与浮点表示时相对误差的上限有关，规定了最坏情况下误差的大小

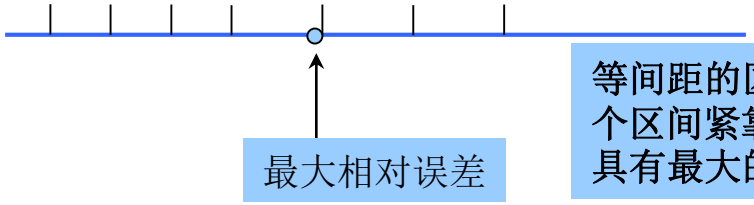
数的计算机表示

IEEE标准

单精度浮点数：总32位，尾数24位

双精度浮点数：总64位，尾数53位

- 机器精度：
  - 采用上例中的浮点系统，$b=2$，$t=3$，则$\varepsilon=2^{1-3}=0.25$。
  - 对于舍去方法，相对量化误差小于0.25。



等间距的区间，第一个区间紧靠上界的数，具有最大的相对误差

- 误差幅度与量化误差的相关性有很多实际的应用：
  - 测试两个数是否相等
    - 在迭代过程中，判断一个量的收敛性或迭代过程是否中止，可测试它们的差是否小于某个可接受的容限。
    - 用相对误差。
    - 可用机器精度作为停止或收敛准则。

二进制计算机决定机器精度的伪代码

```
epsilon = 1
DO
        IF (epsilon+1≤1) EXIT
        epsilon = epsilon/2
END DO
epsilon = 2 × epsilon
```

- **机器精度代表最小的相对误差，1*(1+epsilon)与1对比。**
- **MATLAB中可以使用 single 函数将数据转换为单精度。**

计算机中的算术运算

- 通用算术运算（加、减、乘、除）——有效数字丢失
  - 当两个浮点数相加时，需要对较小的数的尾数进行调整，使两个数的指数相同
    - 例：$0.1557 \cdot 10^{1}+0.4381 \cdot 10^{-1}$
    - 将结果舍去处理后得$0.1600 \cdot 10^{1}$

0.1557	$.10^{1}$
0.004381	$.10^{1}$
0.160081	$.10^{1}$

  - 减法的执行过程类似， $0.0995 \cdot 10^{2}=0.9950 \cdot 10^{1}$

0.7642	$.10^{3}$
-0.7641	$.10^{3}$
0.0001	$.10^{3}$

0.3641	$.10^{2}$
-0.2686	$.10^{2}$
0.0955	$.10^{2}$

    - $0.0001 \cdot 10^{3}=0.1000 \cdot 10^{0}$，之后的计算会把后面的3个0都作为有效数字
    - 当两个数非常接近时，会产生非常大的计算误差。
  - 乘法：指数相加，尾数相乘。两个n位尾数相乘得到$2n$位计算结果。
    $0.1363 \cdot 10^{3} \times 0.6423 \cdot 10^{-1}=0.08754549 \cdot 10^{2} \longrightarrow 0.8754 \cdot 10^{1}$
  - 除法：尾数相除，指数相减。然后对结果进行归一化和舍去处理。

计算机中的算术运算

- 大规模计算
  - 即使单个计算的舍入误差可能很小，但在大量计算过程中的累积效应可能非常严重。
  - 问题：对一个数求和100000次，对数1以单精度方式求和，对数0.00001分别以单精度和双精度方式求和。
    - 结果：对1进行单精度求和得到期望的结果，但对0.00001进行单精度求和的结果却具有较大的偏差，但当对0.00001进行双精度求和时，计算误差显著减小。
    - 根源：量化误差。因为整数1可以在计算机中准确地表示，所以对1可以精确地求和运算。0.00001不能在计算机中准确地表示，只能对其进行量化处理时，得到一个与真值略微不同的值，尽管这是一个非常微小的差异，对于小量计算可以忽略，但在重复进行大量计算后，这个误差会不断累积。在双精度情况下，量化误差要小得多，可以得到改善。
    - *在上述迭代操作中，所有误差具有相同的正负号。很多情况下，误差的符号以随机模式不断交替的，经常相互抵销了，但有些情况下，这些误差没有被抵销掉，就会得到误差较大的结果。*

计算机中的算术运算

- 大数和小数相加
  - 例：一个小数0.0010和一个大数4000相加，使用一个假想的计算机，具有4位尾数和1位指数，对较小的数进行调整，使其指数与较大的数相匹配。

0.4000	$.10^4$
0.0000001	$.10^4$
0.4000001	$.10^4$

    - 将结果进行舍去处理得$0.4000.10^4$
    - 可以不执行这个加法！
  - 无穷级数求和：初始项通常大于后面的项
  - 反向求和：以升序而不是降序对级数求和
  - 当n趋于无穷时，无穷级数

$$f(n)=\sum_{i=1}^{n}\frac{1}{i^4}$$

收敛于$\pi^4/90$。采用单精度表示，编写一个程序计算$n=10000$时$f(n)$的值，分别按从$i=1$到10000和$i=10000$到1的顺序计算，对比误差并分析结果。

计算机中的算术运算

- 减性抵销——两个几乎相等的浮点数相减时所引起的舍入误差
  - 例：二次求根公式 $x_{1,2}=\dfrac{-b\pm\sqrt{b^2-4ac}}{2a}$
  - 当 $b^2 >> 4ac$ 时，分子可能非常小。
  - **可以用双精度**
  - 或变换公式 $x_{1,2}=\dfrac{-2c}{b\pm\sqrt{b^2-4ac}}$
  - 或先计算较大的根，再根据 $x_1x_2=\dfrac{c}{a}$ 计算较小的根。

问题：当a=1，b=3000.001，c=3时，分别以单精度和双精度计算方程的根。（真实根为x_1=-0.001，x_2=-3000）

计算机中的算术运算

- 拖尾效应（smearing）
  - 在求和过程中，如果某一项的值大于和值本身时，就会出现拖尾效应。
  - 在符号交替的级数中会出现这种现象。
  - 问题：指数函数 $y=e^x$ 以下面无穷级数的形式给出：

$$y=1+x+\frac{x^2}{2}+\frac{x^3}{3!}+\cdots$$

分别计算x=10和x=-10时的函数值，研究其舍入误差。

  - 在不断地进行加减运算时，抵销了大量的有效数字。
  - 其他计算策略或扩展精度。
- 内积 $\sum_{i=1}^{n} x_i y_i = x_1 y_1 + x_2 y_2 + \cdots + x_n y_n$ 使用扩展精度

指数函数运行结果

x=10			x=-10		
i	term	sum	i	term	sum
0	1	1	0	1	1
1	10	11	1	-10	-9
2	50	61	2	50	41
3	166.6667	227.6666718	3	-166.6667	-125.6666718
4	416.6667	644.333374	4	416.6667	291
5	833.3334	1477.666748	5	-833.3334	-542.333374
…			…		
27	0.0918369	22026.4179688	41		
28	0.0327989	22026.4511719	42		
29	0.01131	22026.4628906	43	-1.6552107e-010	1.1033645569114e-04
30	0.00377	22026.4667969	44	3.7618422e-011	1.1033649207093e-04
31	0.0012161	22026.46875	45	-8.3596498e-012	1.1033648479497e-04
Exact value	22026.465794806718		Exact value	0.000045399929762	

截断误差与泰勒级数

- 截断误差是由于用近似过程代替准确过程而产生的误差。

$$\frac{dv}{dt} \approx \frac{\Delta v}{\Delta t} = \frac{v(t_{i+1}) - v(t_i)}{t_{i+1} - t_i}$$

近似真导数值

- 利用泰勒级数了解截断误差

泰勒定理

- 在包含a和x的区间上，如果一个函数f及其直至$n+1$阶导数都是连续的，那么该函数在x处的值可以表示为：

$$f(x)=f(a)+f'(a)(x-a)+\frac{f''(a)}{2!}(x-a)^2+\ldots+\frac{f^{(n)}(a)}{n!}(x-a)^n+R_n$$

$$R_n=\int_a^x\frac{(x-t)^n}{n!}f^{n+1}(t)dt$$

积分形式的余项

泰勒定理表明：任何光滑的函数都可以用多项式来逼近

➢应用中值定理，可以得到余项的拉格朗日形式：

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

a和x之间的一个点

泰勒级数

- 0阶近似(zero-order approximation)

$$f(x_{i+1}) \cong f(x_i)$$

- 一阶近似(first-order approximation)

$$f(x_{i+1}) \cong f(x_i) + f'(x_i)(x_{i+1} - x_i)$$

- 完整的泰勒展开

$$f(x_{i+1}) = f(x_i) + f'(x_i)(x_{i+1} - x_i) + \frac{f''}{2!}(x_{i+1} - x_i)^2 + \ldots + \frac{f^{(n)}}{n!}(x_{i+1} - x_i)^n + R_n$$

存在一个值可以给出准确的误差表示

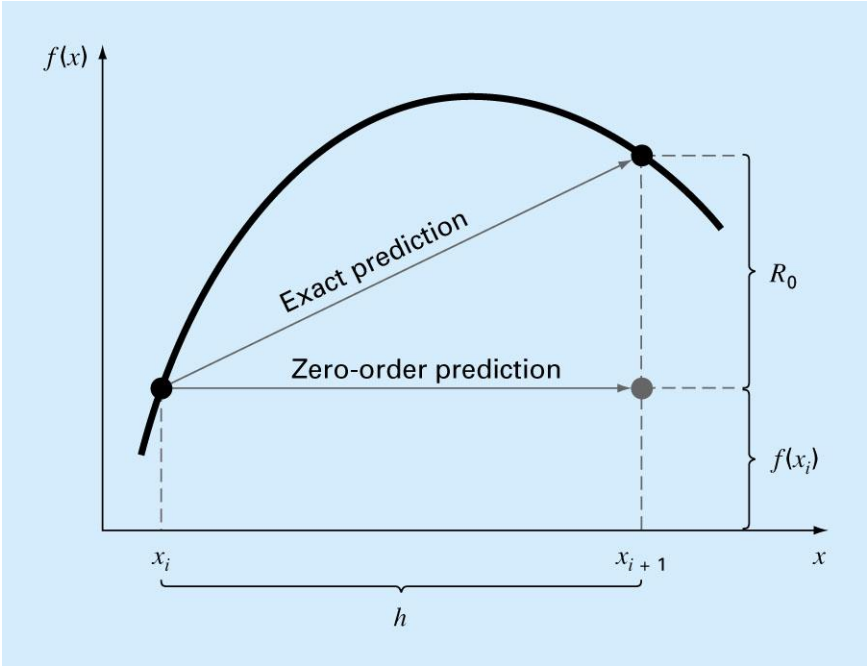
$$R_n = \frac{f^{n+1}(\xi)}{(n+1)!}(x_{i+1} - x_i)^{n+1}$$

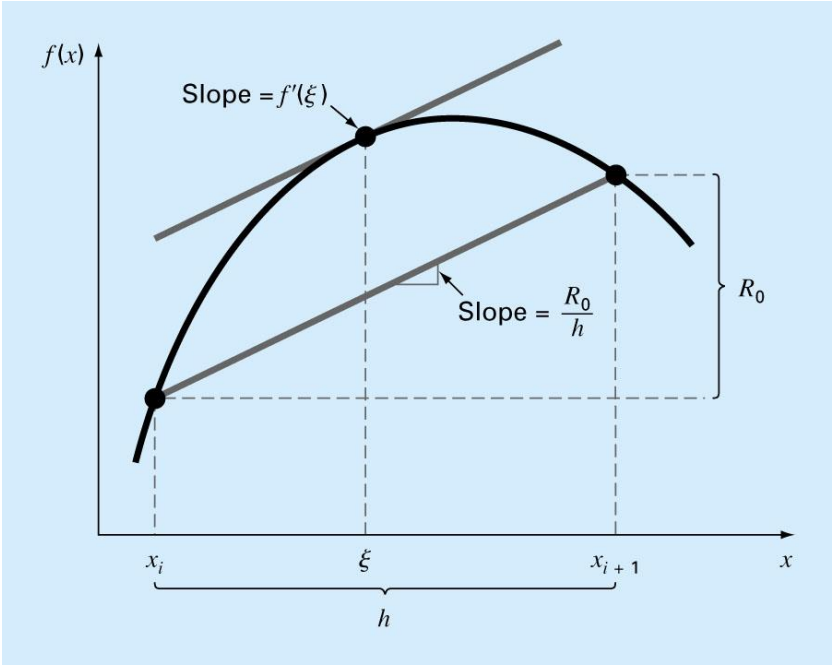
➢定义步长 $h = x_{i+1} - x_i$

$$f(x_{i+1}) = f(x_i) + f'(x_i)h + \frac{f''}{2!}h^2 + \ldots + \frac{f^{(n)}}{n!}h^n + R_n$$

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

泰勒级数展开的余项


f(x)
Exact prediction
Zero-order prediction
R_0
$f(x_i)$
x_i
x_{i+1}
x
h


f(x)
Slope = $f'(\xi)$
Slope = $\frac{R_0}{h}$
R_0
x_i
ξ
x_{i+1}
x
h

泰勒级数的多项式逼近

- 例：当 $x_i=0, h=1$时，用零阶到四阶泰勒级数展开预测函数

$$f(x)=-0.1x^4-0.15x^3-0.5x^2-0.25x+1.2$$

在$x_{i+1}=1$处的函数值：

- 解：$f(0)=1.2$
  - n=0，$f(x_{i+1})\cong 1.2$

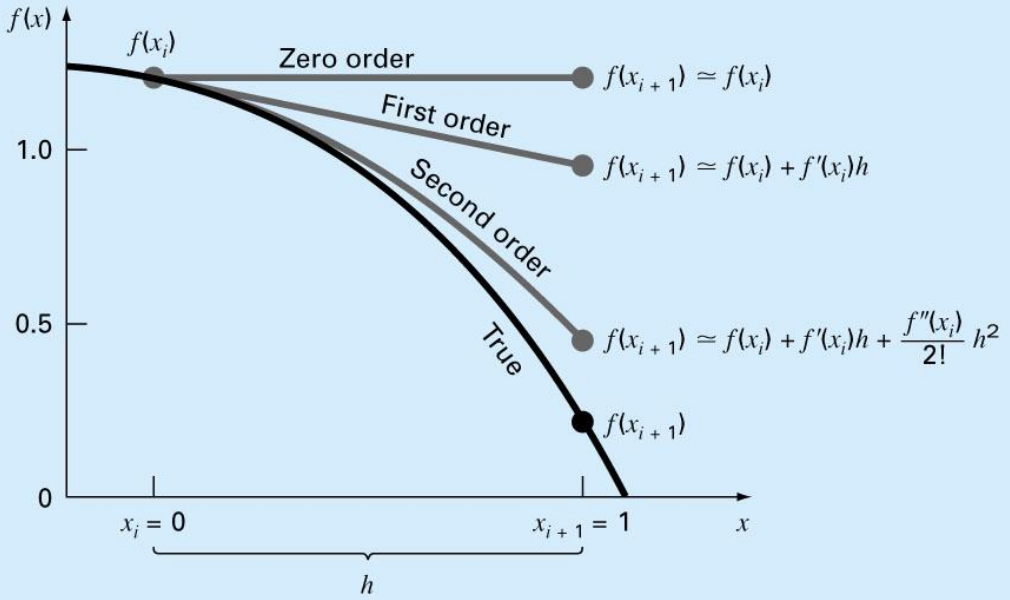
    $E_t=0.2-1.2=-1.0$
  - n=1，$f'(0)=-0.25$

    $f(x_{i+1})\cong 1.2-0.25h$

    $f(1)\cong 0.95$

    $E_t=0.2-0.95=-0.75$
  - n=2，$f''(0)=-1.0$

    $f(x_{i+1})\cong 1.2-0.25h-0.5h^2$　　$f(1)\cong 0.45$　　$E_t=0.2-0.45=-0.25$



泰勒级数的多项式逼近

- 四阶级数的泰勒级数展开将得到一个准确的估计结果：

$$f(x)=1.2-0.25h-0.5h^2-0.15h^3-0.1h^4$$

$$R_4=\frac{f^{(5)}(\xi)}{5!}h^5=0$$

- 通常，n阶多项式的n阶泰勒级数展开得到的结果是准确的。对于其他连续可微函数，如指数函数和正弦函数，有限级数项是不可能得到准确结果的。每增加一项将使近似结果得到一定的改进，但改进程度不显著。
- 多数情况下，泰勒级数展开的实际值只需要包含少数几项就可以得到非常接近真值的近似结果，对于实际应用来说足够了。

泰勒级数展开的逼近误差

- 利用余项估计需要多少项才能得到足够接近真值的近似值？ $R_n = \frac{f^{n+1}(\xi)}{(n+1)!}h^{n+1} \Rightarrow R_n = O(h^{n+1})$
- 误差与步长的$n+1$次方成比例。
  - 如果误差的量级为 $O(h)$，那么步长减半就会使误差也减半。
  - 如果误差的量级为 $O(h^2)$，那么步长减半就会使误差变为原来的四分之一。
- 一般地，假定截断误差随泰勒级数项数的增加而减小。在许多情况下，如果h足够小，一阶、二阶以及其他低阶项通常不成比例地占误差的绝大部分。因此，只需要少数项就可以获得一个足够精确的估计值。

泰勒级数展开的逼近误差

- 例：利用泰勒级数展开，基于$f(x)=\cos(x)$在$x_i=\pi/4$处的函数值和导数值，用n=0~6共7项来逼近$x_{i+1}=\pi/3$的值。
- 解：$h=\pi/3-\pi/4=\pi/12$　　$f(\pi/3)=0.5$

阶数n	$f^{(n)}(x)$	$f(\pi/3)$	百分比相对误差
0	cos$x$	0.707106781	-41.4
1	-sin$x$	0.512986659	-4.4
2	-cos$x$	0.497754491	0.449
3	sin$x$	0.499869147	2.62×10^{-2}
4	cos$x$	0.500007551	-1.51×10^{-3}
5	-sin$x$	0.500000304	-6.08×10^{-5}
6	-cos$x$	0.499999988	2.44×10^{-6}

增加更多的项，误差的改进可以忽略不计。

利用泰勒级数估计截断误差

- 降落伞问题：
  - 将$v(t)$展开成泰勒级数

$$v(t_{i+1}) = v(t_i) + v'(t_i)(t_{i+1} - t_i) + \frac{v''(t_i)}{2!}(t_{i+1} - t_i)^2 + \ldots + R_n$$

  - 截去一阶导数项以后的各项

$$v(t_{i+1}) = v(t_i) + v'(t_i)(t_{i+1} - t_i) + R_1$$

  - 可得：

$$v'(t_i) = \frac{v(t_{i+1}) - v(t_i)}{t_{i+1} - t_i} - \frac{R_1}{t_{i+1} - t_i}$$

一阶逼近　　截断误差

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

$$\frac{R_1}{t_{i+1} - t_i} = \frac{v''(\xi)}{2!}(t_{i+1} - t_i)$$

$$\frac{R_1}{t_{i+1} - t_i} = O(t_{i+1} - t_i)$$

导数的近似误差与步长成比例。如果将步长减半，则导数值的误差也减半。（其它误差可类似处理）

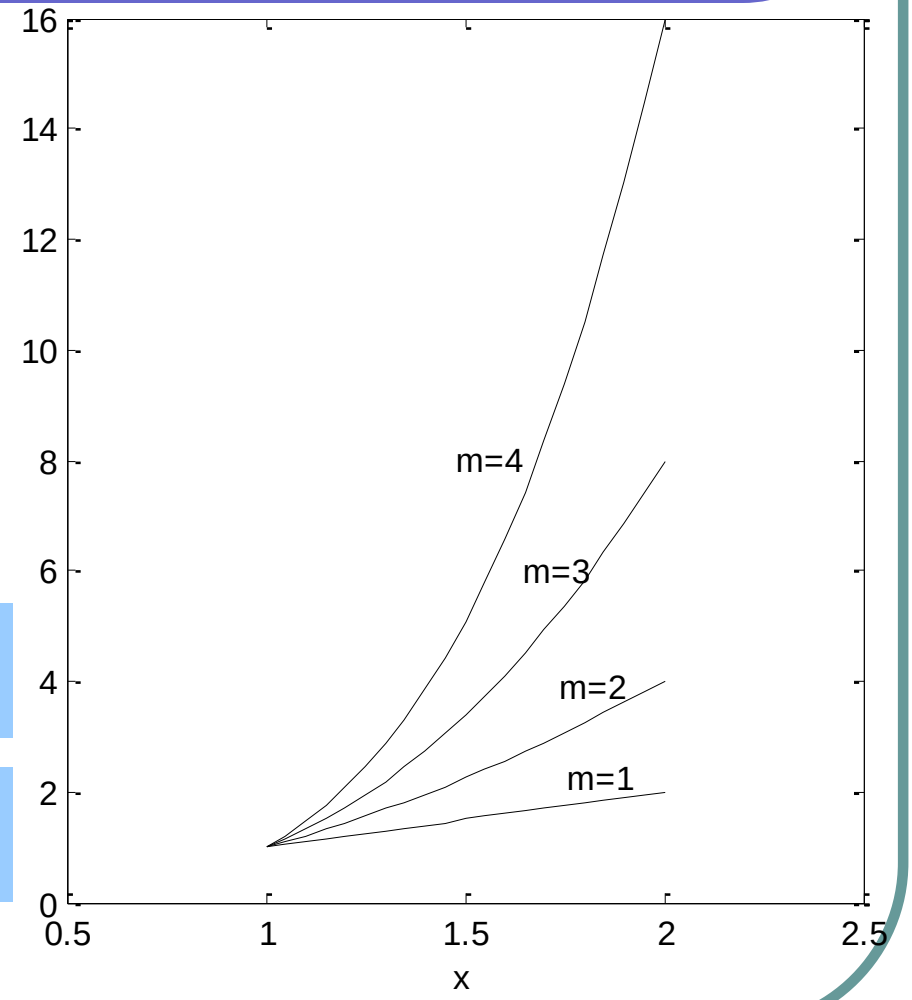
非线性与步长对泰勒级数逼近的影响

- 问题：$f(x)=x^m, m=1,2,3,4$
  $x\in[1,2]$

- 一阶泰勒级数展开：

$$f(x_{i+1})=f(x_i)+mx_i^{m-1}h$$

$$R_1=\frac{f''(x_i)}{2!}h^2+\frac{f^{(3)}(x_i)}{3!}h^3+\frac{f^{(4)}(x_i)}{4!}h^4+\cdots$$



➢随着函数非线性程度的增大，一阶泰勒级数逼近的误差增大。

➢随着步长的减小，一阶泰勒级数逼近的误差减小。

第一章 绪论

- 课程简介
  - 什么是数值计算方法?
  - 为什么学习数值计算方法?
  - 数值计算方法的主要内容
- **数值计算中的误差**
  - 误差的种类及其来源
  - 绝对误差与相对误差
  - 有效数字与误差
  - 舍入误差与截断误差
  - **误差的传播与估计**
  - 算法的数值稳定性

误差传播——单变量函数

- 假设一个函数$f(x)$，设 $\tilde{x}$ 是x的近似值，估计x和 $\tilde{x}$ 的差异对函数值的影响，即

$$\Delta f(\tilde{x}) = |f(x) - f(\tilde{x})|$$

- 用泰勒级数计算逼近 $f(\tilde{x})$ 的 $f(x)$

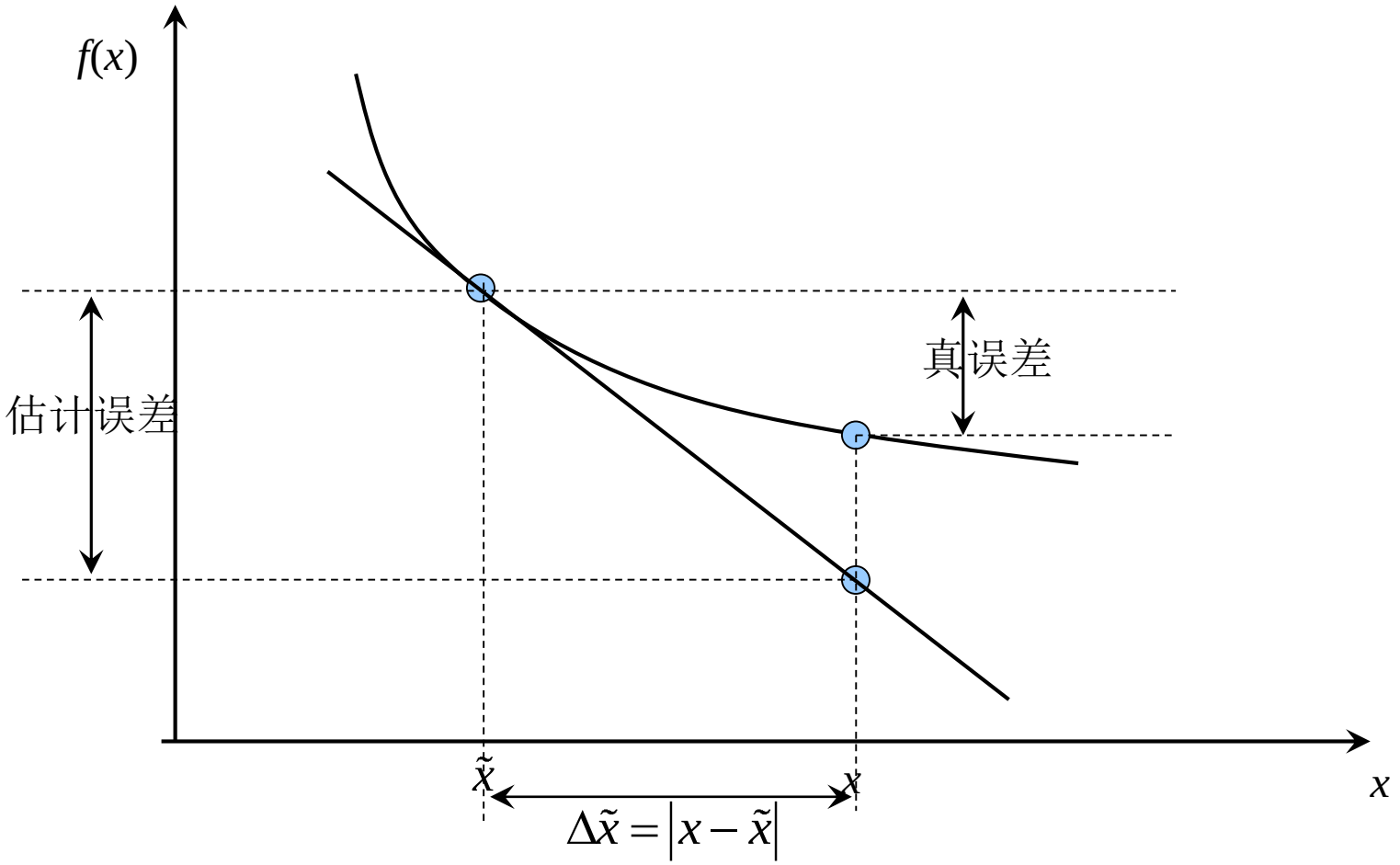
$$f(x) = f(\tilde{x}) + f'(\tilde{x})(x-\tilde{x}) + \frac{f''(\tilde{x})}{2}(x-\tilde{x})^2 + \cdots$$

- 舍去二阶及二阶以上的项，并重新整理：

$$f(x) - f(\tilde{x}) \cong f'(\tilde{x})(x-\tilde{x}) \quad \text{或} \quad \Delta f(\tilde{x}) = |f'(\tilde{x})||x-\tilde{x}|$$

函数值的误差估计　函数的导数　自变量的误差估计值

一阶误差传播的图形描述



单变量函数的误差传播

- 问题：给定一个值 $\tilde{x}=2.5$ ，其误差为 $\Delta\tilde{x}=0.01$ ，估计由此导致的函数值 $f(x)=x^3$ 的误差。
- 解：

$$\Delta f(\tilde{x}) \cong \left|f'(\tilde{x})\right|(x-\tilde{x}) = 3(2.5)^2(0.01) = 0.1875$$

由于

$$f(2.5)=15.625$$

可以预测：

$$f(\tilde{x})=15.625\pm 0.1875$$

误差传播——多变量函数

- 一个具有两个变量u和v的函数，泰勒级数可以写为

$$f(u_{i+1},v_{i+1})=f(u_i,v_i)+\frac{\partial f}{\partial u}(u_{i+1}-u_i)+\frac{\partial f}{\partial v}(v_{i+1}-v_i)$$

$$+\frac{1}{2!}\left[\frac{\partial^2 f}{\partial u^2}(u_{i+1}-u_i)^2+2\frac{\partial^2 f}{\partial u\partial v}(u_{i+1}-u_i)(v_{i+1}-v_i)+\frac{\partial^2 f}{\partial v^2}(v_{i+1}-v_i)^2\right]+\cdots$$

舍去二阶和高阶项

$$\Delta f(\tilde{u},\tilde{v})=\left|\frac{\partial f}{\partial u}\right|\Delta\tilde{u}+\left|\frac{\partial f}{\partial v}\right|\Delta\tilde{v}$$

- 多变量函数

$$\Delta f(\tilde{x}_1,\tilde{x}_2,\cdots,\tilde{x}_n)=\left|\frac{\partial f}{\partial x_1}\right|\Delta\tilde{x}_1+\left|\frac{\partial f}{\partial x_2}\right|\Delta\tilde{x}_2+\cdots+\left|\frac{\partial f}{\partial x_n}\right|\Delta\tilde{x}_n$$

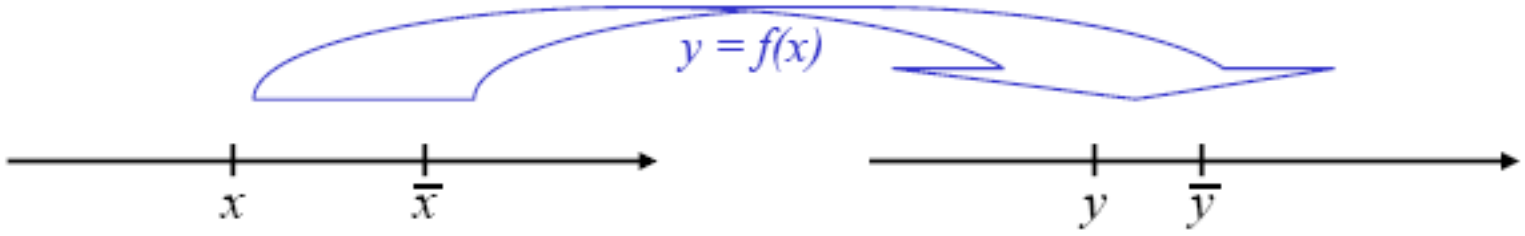
一般数学运算的误差传播关系

运算		估计误差
加法	$\Delta(\tilde{u}+\tilde{v})$	$\Delta\tilde{u}+\Delta\tilde{v}$
减法	$\Delta(\tilde{u}-\tilde{v})$	$\Delta\tilde{u}+\Delta\tilde{v}$
乘法	$\Delta(\tilde{u}\times\tilde{v})$	$\lvert\tilde{u}\rvert\Delta\tilde{v}+\lvert\tilde{v}\rvert\Delta\tilde{u}$
除法	$\Delta\left(\frac{\tilde{u}}{\tilde{v}}\right)$	$\frac{\lvert\tilde{u}\rvert\Delta\tilde{v}+\lvert\tilde{v}\rvert\Delta\tilde{u}}{\lvert\tilde{v}\rvert^2}$

第一章 绪论

- 课程简介
  - 什么是数值计算方法？
  - 为什么学习数值计算方法？
  - 数值计算方法的主要内容
- **数值计算中的误差**
  - 误差的种类及其来源
  - 绝对误差与相对误差
  - 有效数字与误差
  - 舍入误差与截断误差
  - 误差的传播与估计
  - **算法的数值稳定性**

稳定性与条件数



$\bar{x} = x(1+\alpha)$ $\bar{y} = y(1+\beta)$

$$K_P = \frac{|\beta|}{|\alpha|} = \left|\frac{f(\tilde{x})-f(x)}{f(x)}\right| \Big/ \left|\frac{\tilde{x}-x}{x}\right| = \left|\frac{f(\tilde{x})-f(x)}{\tilde{x}-x}\right| \times \left|\frac{x}{f(x)}\right| \cong \left|x\frac{f'(x)}{f(x)}\right|$$

$K_P >> 1$

病态问题

稳定性与条件数

- 一个算法如果原始数据有误差，而计算过程舍入误差不增长，则称此算法是数值稳定的，否则，若误差增长，则称算法是不稳定的（numerically unstable）。（输入值的不确定性通过数值方法在总体上是放大的）
- 条件数（condition number）定义为相对误差之比：
  - $f(x)$与x的相对误差分别为 $\dfrac{f(x)-f(\tilde{x})}{f(\tilde{x})} \cong \dfrac{f'(\tilde{x})(x-\tilde{x})}{f(\tilde{x})}$ $\dfrac{x-\tilde{x}}{\tilde{x}}$
  - 条件数为 $\left|\dfrac{\tilde{x}f'(\tilde{x})}{f(\tilde{x})}\right|$
- 条件数体现了x的不确定性被$f(x)$放大程度。
  - 如果条件数等于1，表示函数的相对误差等于x的相对误差;
  - 如果条件数大于1，表示相对误差被放大了;
  - 如果条件数小于1，表示相对误差减小了。
- 条件数非常大(大于等于10)的函数称为病态（ill-conditioned）函数。

问题条件数与算法条件数

- 例：问题条件数

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

$$\tilde{x}=100 \Rightarrow f(x)=0.5\times10^{-2}$$

由于已知函数形式，由上页条件数计算公式直接计算也得到相同结果

$$K_P=1$$

近似展开

$$f'(x)=\frac{x}{\sqrt{x^2+1}}-1 \Rightarrow f'(\tilde{x})=\frac{\tilde{x}-\sqrt{\tilde{x}^2+1}}{\sqrt{\tilde{x}^2+1}}\approx-\frac{1}{2\tilde{x}^2}=-0.5\times10^{-4}$$

(≈当x>>1时)

- 算法A：4位有效数字

$$\tilde{x}=0.1\times10^3 \Rightarrow f(\tilde{x})=\sqrt{(0.1000+0.00001)\times10^5}-0.1\times10^3=0$$

$$|\beta|=\left|\frac{f(\tilde{x})-f(x)}{f(x)}\right|=1$$

$$|\alpha|=\left|\frac{\tilde{x}-x}{x}\right|\le\frac{1}{2}10^{1-t}\approx\frac{1}{2}10^{-3}$$

$$K_A=\frac{|\beta|}{|\alpha|}\approx2000$$

算法条件数，改善方法：提高精度，重写算法

- 算法B：$f(x)=\frac{1}{\sqrt{x^2+1}+x}$

$$f(\tilde{x})=\frac{1}{0.1\times10^3+0.1\times10^3}=0.5\times10^{-2}$$

$$|\beta|\approx0 \Rightarrow K_A\approx0<<1$$

总的数值误差

- 截断误差和舍入误差之和
  - 通常，最小化舍入误差的方法是增加计算机的有效数字个数
  - 缩短步长使截断误差减小
  - 但缩短步长可能导致减性抵销或增加计算量，使舍入误差增大
  - **大多数计算机可以表示足够多的有效数字，因此舍入误差不会占主导地位**

避免误差危害的原则

- 选择数值稳定的计算方法，避开不稳定的算式。
- 注意简化计算步骤及公式，设法减少运算次数；选用运算次数少的算式，尤其是乘方幂次要低，乘法和加法的次数要少，以减少舍入误差，同时可节约计算机的机时。
- 合理安排计算顺序，防止大数“淹没”小数。多个数相加时，最好从绝对值最小的数到绝对值最大的数依次相加；多个数相乘时，最好从有效位数最多的数到有效位数最少的数依次相乘。
- 避免两相近数相减。
- 避免用绝对值很小的数作为除数。

误差传播——例

- 初始误差可稳定传播或不稳定传播
- 例：用无限精度算法结合下列3个方案可递归生成序列 $\{1/3^n\}_{n=0}^{\infty}$ 中的各项值。

(1) $r_0 = 1, \quad r_n = \frac{1}{3} r_{n-1}, n = 1, 2, \cdots$

(2) $p_0 = 1, p_1 = \frac{1}{3}, p_n = \frac{4}{3} p_{n-1} - \frac{1}{3} p_{n-2}, n = 2, 3, \cdots$

(3) $q_0 = 1, q_1 = \frac{1}{3}, q_n = \frac{10}{3} q_{n-1} - q_{n-2}, n = 2, 3, \cdots$

误差传播——例

- (2)差分方程的通解是

$$p_n = A(1/3^n) + B$$

- 验证如下：

$$\frac{4}{3}p_{n-1} - \frac{1}{3}p_{n-2} = \frac{4}{3}\left(\frac{A}{3^{n-1}} + B\right) - \frac{1}{3}\left(\frac{A}{3^{n-2}} + B\right)$$

$$= \left(\frac{4}{3^n} - \frac{3}{3^n}\right)A + \left(\frac{4}{3} - \frac{1}{3}\right)B = A\frac{1}{3^n} + B$$

$$= p_n$$

- 设A=1且B=0，可得到期望的序列。

误差传播——例

- (3)差分方程的通解是

$$p_n = A(1/3^n) + B3^n$$

- 验证如下：

$$\frac{10}{3}p_{n-1} - p_{n-2} = \frac{10}{3}\left(\frac{A}{3^{n-1}} + B3^{n-1}\right) - \left(\frac{A}{3^{n-2}} + B3^{n-2}\right)$$

$$= \left(\frac{10}{3^n} - \frac{9}{3^n}\right)A + (10-1)3^{n-2}B = A\frac{1}{3^n} + B3^n$$

$$= p_n$$

- 设A=1且B=0，可得到期望的序列。

误差传播——例

- 例：用下列方法求出序列 $\{x_n\}=\left\{1/3^n\right\}$ 的近似值。

(1) $r_0=0.99996,\qquad r_n=\frac{1}{3}r_{n-1},n=1,2,\cdots$

(2) $p_0=1,p_1=0.33332,p_n=\frac{4}{3}p_{n-1}-\frac{1}{3}p_{n-2},n=2,3,\cdots$

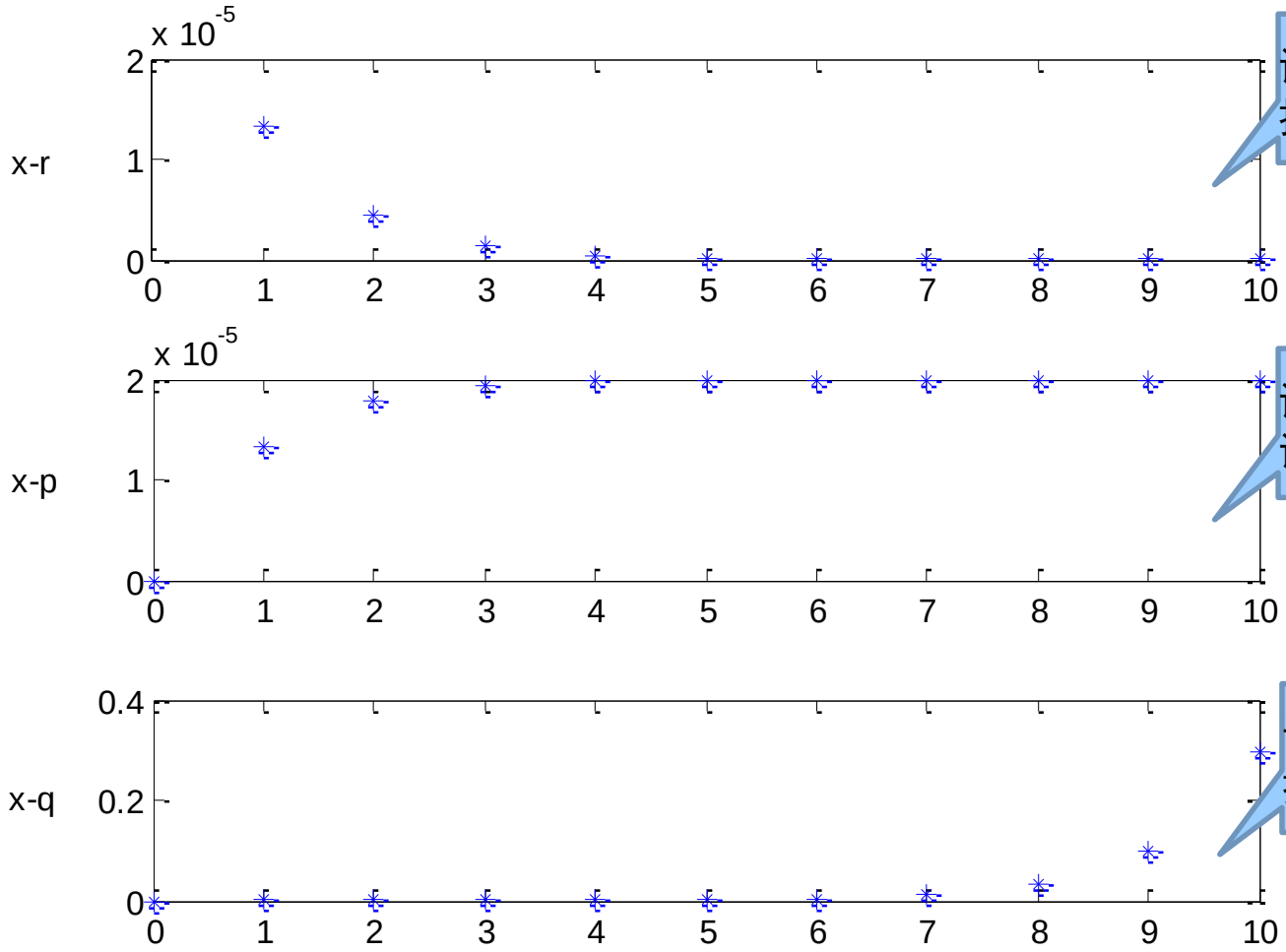
(3) $q_0=1,q_1=0.33332,q_n=\frac{10}{3}q_{n-1}-q_{n-2},n=2,3,\cdots$

- r_0 初始误差为0.00004，p_1,q_1 初始误差为0.00001333，试研究每个算法的误差传播情况。

误差传播——例

n	x_n	r_n	$x_n - r_n (10^{-4})$	p_n	$x_n - p_n (10^{-4})$	q_n	$x_n - q_n (10^{-4})$
0	1.0000000000	0.9999600000	0.400000	1.0000000000	0.000000	1.0000000000	0.0000000000
1	0.3333333333	0.3333200000	0.133333	0.3333200000	0.133333	0.3333200000	0.0000133333
2	0.1111111111	0.1111066667	0.044444	0.1110933333	0.177778	0.1110666667	0.0000444444
3	0.0370370370	0.0370355556	0.014815	0.0370177778	0.192593	0.0369022222	0.0001348148
4	0.0123456790	0.0123451852	0.004938	0.0123259259	0.197531	0.0119407407	0.0004049383
5	0.0041152263	0.0041150617	0.001646	0.0040953086	0.199177	0.0029002469	0.0012149794
6	0.0013717421	0.0013716872	0.000549	0.0013517695	0.199726	-0.0022732510	0.0036449931
7	0.0004572474	0.0004572291	0.000183	0.0004372565	0.199909	-0.0104777503	0.0109349977
8	0.0001524158	0.0001524097	0.000061	0.0001324188	0.199970	-0.0326525834	0.0328049992
9	0.0000508053	0.0000508032	0.000020	0.0000308063	0.199990	-0.0983641945	0.0984149997
10	0.0000169351	0.0000169344	0.000007	-0.0000030646	0.199997	-0.2952280648	0.2952449999

误差传播——例



误差是稳定的，且按指数级递减

误差是稳定的，但误差占支配地位

误差是不稳定的，按指数级增长

第一部分 总结（一）

- 数学问题的类型
- 数值方法的特点
  - 稳定性
  - 准确性与精确性
  - 收敛速度

第一部分 重要关系

- 误差
  - 真误差
  - 真百分比相对误差
  - 近似百分比相对误差
  - 停止准则
- 泰勒级数
  - 泰勒级数展开
  - 余项
- 误差传播

Matlab基础

- Arithmetic Operations （算术符号）
- Built-in Functions(内建函数）
- Assignment Statements(赋值语句)
- Defining Function(函数定义)
- Matrices(矩阵)
- Matrix Operations(矩阵运算)
- Array Operations(数组运算)
- Graphics(图形)
- Loops and Conditionals(循环和条件）
- Programs(程序）