



《数值分析 B》笔记、习题与实验

作者：Chad Holton

组织：哈尔滨工业大学能源科学与工程学院

修订日期：October 19, 2025

邮箱：chadholton@qq.com

目录

第一章 绪论	2
1.1 误差的定义	2
1.2 一个算法稳定性的例子	3
1.3 误差与有效数字	4
1.4 函数的误差估计	5
1.5 几点注意事项	6
1.6 思考题与习题	6
第二章 函数逼近与快速傅里叶变换	7
2.1 函数逼近的基本概念	7
第三章 非线性方程与方程组的数值解法	8
3.1 方程求根与二分法	8
3.2 习题	8
3.3 计算实习题	8
第四章 非线性方程（组）的数值解法	9
第五章 线性方程组的数值解法	10
第六章 插值方法与数值逼近	11
第七章 数值积分	12
第八章 矩阵特征值与特征向量的计算	13
第九章 常微分方程初值问题的数值解法	14
9.1 习题	14

前言

《数值分析》是许多专业的研究生阶段的必修科目, 本文档是我学习整理的笔记、习题和实验部分, 适用于哈尔滨工业大学的数值分析 B 课程. 本科数学例如高数的大一和考研的题型与难度也不一样. 高等数学因为存在考研所以内容规范且稳定, 而数值分析每个学校内容重点都不同. 2025 级 32 课时 +4 次上机实验, 很多内容比较简略, 话说是不是压缩课时了, 讲真教材、PPT 与习题需要相应更新以适应压缩课时的情形. 你问下 AI 让它设计 32 学时的授课方案, 它认为课时有限会删很多东西. \LaTeX 源代码在本人的 GitHub 主页 [phychi](#), 虽然也没有什么内容. 本文档部分内容可能有些刻意, 不像笔记.

感谢 [ElegantLaTeX](#) 提供的精美模板, 停止更新太可惜了.

第一章 绪论

数值计算的根本任务是研究算法. 计算机基础运算是加减乘除, 其他计算转化为这, 根据步数等衡量算法的优劣.

- 计算 $\sin x$, 可以用泰勒公式展开 $\sin x = x - \frac{x^3}{3!} + \frac{x^5}{5!} \cdots (-1)^n \frac{x^{2n+1}}{(2n+1)!} + R_{2n+1}(x)$ 取前几项近似.
- 解线性方程组有 Cramer 法则, 但是按照这个方法, 计算 n 阶行列式的值如果按照定义, 随着 n 的增加, 计算量会极速增长, 需要更好的算法.

1.1 误差的定义

误差, 即一个物理量的真实值与计算值之间的差异, 来源可以分为四类.

- 从实际问题中抽象出数学模型, 即**模型误差**. 例如在建模时的非线性模型的线性化.
- 通过**测量**得到模型中参数的值, 即**观测误差**.
- 求近似解, 展开取前几项, 即**截断误差**.
- 机器字长有限, 保留几位有效数字, 即**舍入误差**.

如下图, 泰勒展开取前 4 项, 剩余项即截断误差, 而前四项求和有两项要小数四舍五入的舍入误差.

例: 近似计算 $\int_0^1 e^{-x^2} dx = 0.747... ..$

解法之一: 将 e^{-x^2} 作 Taylor 展开后再积分

$$\begin{aligned}\int_0^1 e^{-x^2} dx &= \int_0^1 \left(1 - x^2 + \frac{x^4}{2!} - \frac{x^6}{3!} + \frac{x^8}{4!} - \cdots \right) dx \\ &= 1 - \frac{1}{3} + \frac{1}{2!} \times \frac{1}{5} - \frac{1}{3!} \times \frac{1}{7} + \frac{1}{4!} \times \frac{1}{9} - \cdots\end{aligned}$$

取 $\int_0^1 e^{-x^2} dx \approx S_4$, S_4 R_4 /* Remainder */

则 $R_4 = \frac{1}{4!} \times \frac{1}{9} - \frac{1}{5!} \times \frac{1}{11} + \cdots$ 称为**截断误差** /* Truncation Error */

这里 $|R_4| < \frac{1}{4!} \times \frac{1}{9} < 0.005$

$$S_4 = 1 - \frac{1}{3} + \frac{1}{10} - \frac{1}{42} \approx 1 - 0.333 + 0.1 - 0.024 = 0.743$$

| **舍入误差** /* Roundoff Error */ | $< 0.0005 \times 2 = 0.001$

| 计算 $\int_0^1 e^{-x^2} dx$ 的总体误差 | $< 0.005 + 0.001 = 0.006$

图 1.1: 误差举例

1.2 一个算法稳定性的例子

定义 1.1

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,则称此算法是稳定的;否则称此算法是不稳定的。



例题 1.1 计算 $I_n = e^{-1} \int_0^1 x^n e^x$ 并估计误差.

存在两种计算方法,对比效果.

公式一: $I_n = 1 - nI_{n-1}$, 可以看到随着 n 增长计算结果越来越离谱.

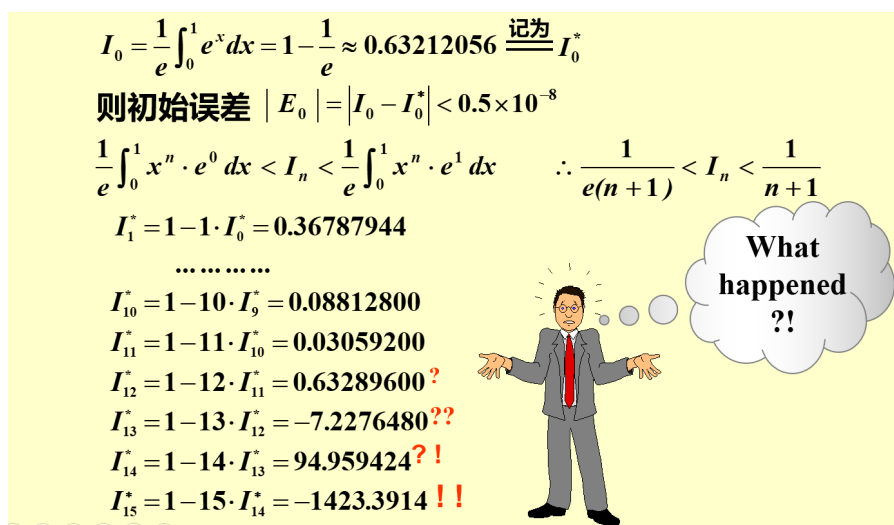


图 1.2: 不稳定算法

根据递推式 $|E_n| = |I_n - I_n^*| = n|E_{n-1}| = \cdots = n!|E_0|$, 可见初始的小扰动 $|E_0| < 0.5 \times 10^{-8}$ 迅速积累, 误差快速增长, 这是不稳定的算法

公式二: $I_{n-1} = \frac{1}{n}(1 - I_n)$, 先估计一个 I_N , 再反推要求的 $I_n (n < N)$. 可取

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

当 $N \rightarrow +\infty$ 时, $|E_N| = |I_N - I_N^*| \rightarrow 0$, 其绝对误差很小, 但相对误差可能比较大, 使用该方法理由后面会知道.

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:

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

图 1.3: 稳定算法

表 1.1: 两种方法与标准值的比较

N	标准值 y_{ref}	方法 1 结果 y_1	方法 2 结果 y_2	方法 1 相对误差	方法 2 相对误差
0	0.632120559	0.63212056	0.63212056	$1.58 \times 10^{-6}\%$	$1.58 \times 10^{-6}\%$
1	0.367879441	0.36787944	0.36787944	$2.72 \times 10^{-6}\%$	$2.72 \times 10^{-6}\%$
2	0.264241118	0.26424112	0.26424112	$7.57 \times 10^{-6}\%$	$7.57 \times 10^{-6}\%$
3	0.207276647	0.20727664	0.20727665	$3.38 \times 10^{-6}\%$	$1.45 \times 10^{-5}\%$
4	0.170893412	0.17089344	0.17089341	$1.64 \times 10^{-5}\%$	$1.17 \times 10^{-5}\%$
5	0.145532941	0.1455328	0.14553294	0.097%	$6.87 \times 10^{-6}\%$
6	0.126802357	0.1268032	0.12680236	0.66%	$2.36 \times 10^{-5}\%$
7	0.112383504	0.1123776	0.11238350	0.0053%	$3.55 \times 10^{-5}\%$
8	0.100931967	0.1009792	0.10093197	0.047%	$2.82 \times 10^{-6}\%$
9	0.091612293	0.0911872	0.09161229	0.46%	$3.38 \times 10^{-6}\%$
10	0.0838770701	0.088128	0.08387707	5.07%	$1.19 \times 10^{-5}\%$
11	0.0773522289	0.030592	0.07735225	60.5%	$2.66 \times 10^{-4}\%$
12	0.0717732536	0.632896	0.07177296	782%	$4.01 \times 10^{-4}\%$
13	0.0669477026	-7.227648	0.06695156	10800%	0.0056%
14	0.0627321639	102.187072	0.06267811	162900%	0.084%
15	0.0590175409	-1531.80608	0.05982836	$2.60 \times 10^6\%$	1.35%

两种方法对比, 可以看到方法 1 即使初始误差较小, 也会快速积累; 方法 2 在 N 较大估计一个初值, 倒推误差快速缩小. 方法 2, 求 I_{15} , 取 $I_{20}^* = \frac{1}{2}[\frac{1}{e \cdot 21} + \frac{1}{21}] \approx 0.03256855812$, 递推得到 $I_{15} \approx 0.05901754785$, 相对误差 $1.18 \times 10^{-5}\%$.

注 在数值计算中, 算法的数值稳定性对结果的准确性至关重要. 本文通过计算积分递推公式

$$I_n = 1 - nI_{n-1}, I_0 = 1 - e^{-1}$$

发现即使采用双精度浮点运算 (MATLAB 或 Python 默认计算方式), 由于舍入误差的累积和放大, 显著偏差的计算结果只是推迟但仍会出现 (如 $I_{18} = -0.0294536708$, 理论值应为严格正数). 即使采用高精度算术, 不稳定的算法仍会导致结果失效.

1.3 误差与有效数字

- 绝对误差: $e^* = x^* - x$, 其中 x 为精确值, x^* 为 x 的近似值.
- $|e^*|$ 的上限记为 ε^* , 称为绝对误差限.
- 相对误差: $e_r^* = \frac{e^*}{x}$
- 相对误差上限定义为: $\varepsilon_r^* = \frac{\varepsilon^*}{|x^*|}$, 注意分母使用的是近似值 x^* , 在合理的情况下反映了数量级.

定义 1.2

若近似值 x^* 的误差限是某一位的半个单位, 该位到 x^* 的第一位非零数字共有 n 位, 则 x^* 有 n 位有效数字.



用科学计数法, 记 $x^* = \pm 0.a_1a_2 \cdots a_n \times 10^m (a_1 \neq 0)$, 若 $|x - x^*| < 0.5 \times 10^{m-n}$, 则称 x^* 有 n 位有效数字.

有效数字 \Rightarrow 相对误差限

已知 x^* 有 n 位有效数字, 则其相对误差限为

$$\begin{aligned} \varepsilon_r^* &= \left| \frac{\varepsilon^*}{x^*} \right| = \frac{0.5 \times 10^{m-n}}{0.a_1a_2 \cdots a_n \times 10^m} = \frac{10^{-n}}{2 \times 0.a_1 \cdots} \\ &\leq \frac{1}{2a_1} \times 10^{-n+1} \end{aligned}$$

相对误差限 \Rightarrow 有效数字

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


$$\begin{aligned} \text{则 } |x - x^*| &\leq \varepsilon_r^* \cdot |x^*| = \frac{10^{-n+1}}{2(a_1 + 1)} \times 0.a_1a_2 \cdots \times 10^m \\ &< \frac{10^{-n+1}}{2(a_1 + 1)} \cdot (a_1 + 1) \times 10^{m-1} = 0.5 \times 10^{m-n} \end{aligned}$$

可见 x^* 至少有 n 位有效数字。

图 1.4: 有效数字与相对误差的关系

1.4 函数的误差估计

问题: 对于 $A = f(x)$, 若用 x^* 取代 x , 将对 A 产生什么影响?

分析: $e^*(A) = f(x^*) - f(x)$  $e^*(x) = x^* - x$
 $= f'(\xi)(x^* - x)$

x^* 与 x 非常接近时, 可认为 $f'(\xi) \approx f'(x^*)$, 则有:

$$|e^*(A)| \approx |f'(x^*)| \cdot |e^*(x)|$$

图 1.5: 函数的误差估计 1

即: x^* 产生的误差经过 f 作用后被放大/缩小了 $|f'(x^*)|$ 倍. 故称 $|f'(x^*)|$ 为放大因子或绝对条件数. 而

$$\begin{aligned}
 |e_r^*(A)| &= \left| \frac{e^*(A)}{f(x^*)} \right| & |e_r^*(x)| &= \left| \frac{e^*(x)}{x^*} \right| \\
 &= \left| \frac{f(x^*) - f(x)}{x^* - x} \cdot \frac{x^*}{f(x^*)} \cdot \frac{x^* - x}{x^*} \right| \\
 &\approx \left| \frac{x^* \cdot f'(x^*)}{f(x^*)} \right| \cdot |e_r^*(x)|
 \end{aligned}$$

f 的条件数在某一点是**小\大**，则称 f 在该点是**好条件的**
 /* well-conditioned */ \ **坏条件的** /* ill-conditioned */。

图 1.6: 函数的误差估计 2

1.5 几点注意事项

- 避免相近二数直接相减, 可能减少有效数字位数

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

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

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

图 1.7: 几种经验性避免方法

- 避免小分母: 分母小会造成浮点精度不足;
- 避免大数吃小数导致浮点数精度问题;
- 先化简再计算, 减少步骤, 避免误差积累;
- 选用稳定的算法.

1.6 思考题与习题

第二章 函数逼近与快速傅里叶变换

这一章存在一个典型问题, 就是大多数非数学专业的人本科并没有学线性空间, 也不理解线性代数怎么与多项式有关系了, 因此就学的云里雾里.

已知 $x_1, \dots, x_N; y_1, \dots, y_N$, 第二章有插值法求得近似函数 $P(x) \approx f(x)$, 但是 N 很大超过拟合需要的点数目, 并且 y_i 本身是不准确的测量值, 插值法将误差完全包含进拟合函数. 此时没必要使用插值, 而应该拟合逼近, 例如最小二乘拟合.

2.1 函数逼近的基本概念

$$\sin x - \frac{x^2}{2} = 0 \text{ 在 } (0,1) \text{ 内的根的近似值 } (\zeta = 0.4 * 10^{-5})$$

第三章 非线性方程与方程组的数值解法

为什么方程需要数值解法? 首先 n 次代数方程 n 个根, $n \geq 5$ 时没有求根公式, $n = 3, 4$ 时求根公式也比较复杂; 其次包含对数函数、指数函数、三角函数等超越函数的超越方程本身大多没有解析解.

3.1 方程求根与二分法


二分法的原理高中就学过, 但是不能用计算器所以高中不会考, 在此熟悉步骤.

零点定理: 在 $[a, b]$ 连续且 $f(a)f(b) < 0$, 则说明在 (a, b) 至少有一个实根, 称 $[a, b]$ 为方程的**有根区间**, 但没有确定数目, 配合单调性可以确定根的唯一性.

二分法的思想是将有根区间折半进行搜索, 即对有根区间 $[a, b]$, 取中点 $x = \frac{a+b}{2}$ 将它分为两半, 检查 $f(x_1)$ 与 $f(a)$ 是否同号, 如果确系同号, 说明所求的根 x^* 在 x_1 的右侧, 这时令 $a_1 = x_1, b_1 = b$; 否则 x^* 必在 x_1 的左侧, 这时令 $a_1 = a, b_1 = x_1$, 不管出现哪一种情况, 新的有根区间仅为原来的一半.


设置终止条件 $|x_{k+1} - x_k| < \varepsilon_1$ 或 $|f(x)| < \varepsilon_2$, 但是不能保证 x 的精度.

3.2 习题

 **练习 3.1** 用二分法求方程 $x^2 - x - 1 = 0$ 的正根, 要求误差小于 0.05.

解: 设 $f(x) = x^2 - x - 1$, 得到 $f'(x) = 2x - 1$, 在 $(0, \frac{1}{2}) < 0, (\frac{1}{2}, +) > 0$

3.3 计算实习题

 **练习 3.2** 用二分法求方程 $x^2 - x - 1 = 0$ 的正根, 要求误差小于 0.05.

第四章 非线性方程（组）的数值解法

第五章 线性方程组的数值解法


第六章 插值方法与数值逼近

第七章 数值积分

第八章 矩阵特征值与特征向量的计算

第九章 常微分方程初值问题的数值解法

9.1 习题

 **练习 9.1** 利用四阶经典的龙格-库塔法求解初边值问题

$$\begin{cases} y' + y = 0 \\ y(0) = 1 \end{cases}$$

讨论步长 h 应取何值方能保证方法的绝对稳定性? 取步长 $h = 0.2$, 求 $x = 0.2, x = 0.4$ 时的数值解, 要求写出由 h, x_n, y_n 直接计算的迭代公式 (计算过程中保留小数点后 3 位)。

解 1. 写出微分方程与经典四阶龙格-库塔公式

原方程: $y' = -y = f(x, y)$, 初值 $y(0) = 1$ 。

经典四阶龙格-库塔公式:

$$\begin{cases} k_1 = f(x_n, y_n), \\ k_2 = f\left(x_n + \frac{h}{2}, y_n + \frac{h}{2}k_1\right), \\ k_3 = f\left(x_n + \frac{h}{2}, y_n + \frac{h}{2}k_2\right), \\ k_4 = f(x_n + h, y_n + hk_3), \\ y_{n+1} = y_n + \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4). \end{cases}$$

代入 $f(x, y) = -y$:

$$\begin{aligned} k_1 &= -y_n, \\ k_2 &= -\left(y_n + \frac{h}{2}k_1\right) = -y_n\left(1 - \frac{h}{2}\right), \\ k_3 &= -\left(y_n + \frac{h}{2}k_2\right) = -y_n\left[1 - \frac{h}{2} + \frac{h^2}{4}\right], \\ k_4 &= -(y_n + hk_3) = -y_n\left[1 - h + \frac{h^2}{2} - \frac{h^3}{4}\right]. \end{aligned}$$

代入 y_{n+1} 公式:

$$\begin{aligned} y_{n+1} &= y_n + \frac{h}{6}[k_1 + 2k_2 + 2k_3 + k_4] \\ &= y_n - \frac{hy_n}{6}\left[1 + 2\left(1 - \frac{h}{2}\right) + 2\left(1 - \frac{h}{2} + \frac{h^2}{4}\right) + \left(1 - h + \frac{h^2}{2} - \frac{h^3}{4}\right)\right]. \end{aligned}$$

计算括号内:

$$1 + 2 + 2 + 1 = 6, \quad -h - h - h = -3h, \quad \frac{h^2}{2} + \frac{h^2}{2} = h^2, \quad -\frac{h^3}{4}.$$

所以:

$$y_{n+1} = y_n \left[1 - h + \frac{h^2}{2} - \frac{h^3}{6} + \frac{h^4}{24}\right].$$

记

$$R(h) = 1 - h + \frac{h^2}{2} - \frac{h^3}{6} + \frac{h^4}{24}.$$

则迭代公式为：

$$y_{n+1} = R(h) \cdot y_n.$$

2. 绝对稳定性条件

绝对稳定性要求 $|R(h)| < 1$ 。对于 $h > 0$ ， $R(h)$ 是 e^{-h} 的四阶泰勒展开，单调递减。当 h 增大到使 $R(h) < -1$ 时失稳。

经典四阶 R-K 法的绝对稳定区间（对 $y' = \lambda y$ ， $\text{Re}(\lambda) < 0$ ）满足：

$$h|\lambda| \leq 2.785293 \dots$$

这里 $\lambda = -1$ ，所以：

$$h < 2.785$$

可保证绝对稳定性。

3. 取 $h = 0.2$ 计算数值解

$$R(0.2) = 1 - 0.2 + \frac{0.04}{2} - \frac{0.008}{6} + \frac{0.0016}{24}$$

逐步计算（保留小数点后 6 位用于中间过程）：

$$1 - 0.2 = 0.8,$$

$$0.8 + 0.02 = 0.82,$$

$$0.82 - 0.001333 = 0.818667,$$

$$0.818667 + 0.000067 = 0.818734.$$

所以：

$$y_{n+1} = 0.818734 \cdot y_n.$$

$$y_0 = 1:$$

$$y(0.2) \approx y_1 = 0.818734 \approx 0.819 \quad (\text{保留 3 位}),$$

$$y(0.4) \approx y_2 = 0.818734 \times 0.818734 \approx 0.670312 \approx 0.670 \quad (\text{保留 3 位}).$$

最终答案：

0.819, 0.670

绝对稳定性要求步长 $h < 2.785$ 。