数值分析

40 学时，学科基础课，可作学位课

课堂讲授，考勤
课后完成作业
随堂闭卷考试

可拍照上传全部或部分作业到课程网站
存为 pdf 文件，文件名：姓名学号第几次

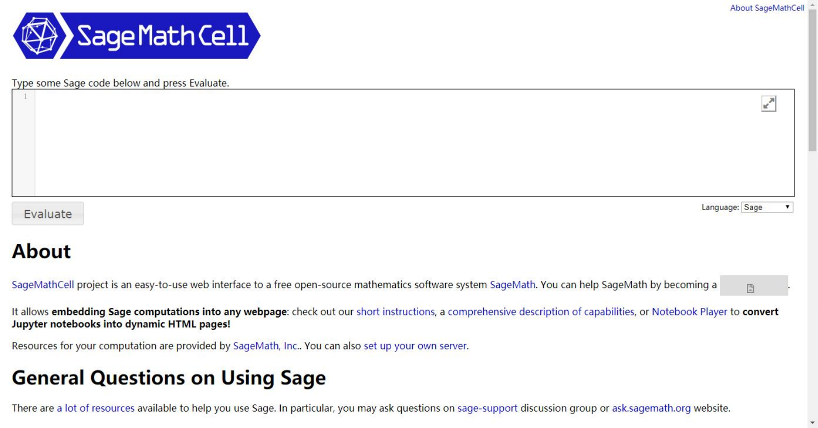
上机:

高级语言：C/C++，Python 等

数值计算软件：Matlab，Octave，Sagemath 等

https://sagecell.sagemath.org/

https://octave-online.net/


About SageMathCell
SageMathCell
Type some Sage code below and press Evaluate.
Evaluate
Language: Sage
About
SageMathCell project is an easy-to-use web interface to a free open-source mathematics software system SageMath. You can help SageMath by becoming a .
It allows embedding Sage computations into any webpage: check out our short instructions, a comprehensive description of capabilities, or Notebook Player to convert Jupyter notebooks into dynamic HTML pages!
Resources for your computation are provided by SageMath, Inc.. You can also set up your own server.
General Questions on Using Sage
There are a lot of resources available to help you use Sage. In particular, you may ask questions on sage-support discussion group or ask.sagemath.org website.

主要参考书：

1. 喻文健著，数值分析与算法，清华大学出版社，2015

2. 李庆扬等著，数值分析，清华大学出版社，2008

3. Cleve B. Moler，Numerical Computing with Matlab，Society for Industrial and Applied Mathematics，2013

4. https://blogs.mathworks.com/

其它参考书：

1. 关治、陆金甫，数值分析基础，高等教育出版社，1998。

2. 奚梅成，数值分析方法，中国科学技术大学出版社，2007。

其它参考书：

1. 关治、陆金甫，数值分析基础，高等教育出版社，1998。

2. 奚梅成，数值分析方法，中国科学技术大学出版社，2007。

预备知识：微积分、线性代数的基本知识。

主要内容

- 数值计算导论
- 非线性方程求根
- 解线性方程组
- 函数逼近和插值
- 数值微分和积分
- 常微分方程的初值问题

背景

数：一元方程的解，如：$\sqrt{2}$，π，$\cdots$；

向量：方程组的解；

函数：微分方程的解。

中国：

刘徽，《九章算术注》，割圆术；

秦九韶，《数书九章》，一元高次方程；

朱世杰，《四元玉鉴》，高次招差法

……

上世纪初，理查森：天气变化是由偏微分方程确定的；离散化它们可以得到“近似解”。

1916 年他组织了大量人力进行了第一次数值预报尝试。用他设计的手摇计算机一人不停计算需要 6 万 4 千天; 或者一个 6 万 4 千人同时工作的计算工厂可预报 24 小时天气。

随着计算机的出现，理查森的梦想实现了：数值求解微分方程可以预测天气！

微分方程准确描述许多物理现象，但是函数空间是“无穷维”的，很难找到精确解。

计算机，只能做有限多个数的运算，想在计算机上得到微分方程的解，是不可能的。

退而求次之：寻找误差很小的“近似解”，画出图像，从而“眼见为实”。

即找到一个算法，编程，在计算机上运算。误差可以达到人们各种“得寸进尺”的要求，只不过要增加计算机时间。

第 1 章 数值计算引论

误差、浮点系统、条件数、减少舍人误差的若干建议。

§1　概述

一、数值问题与数值算法

数值分析研究适合计算机进行科学计算的方法，是计算数学与计算机学科的交叉学科。

数值问题要确定输入数据与输出数据之间函数关系，且输出数据是有限个数。数值问题通常是连续数学问题，求解过程往往需要通过有限步得到近似解。

二、数值问题求解的策略

数值计算的问题来自各个科学和工程分支，可归纳为：

(1) 不计误差，需有限步运算求得解的数学问题。

如：大型线性方程组求解问题。

(2) 不计误差，需无限步运算求得解的数学问题。
如：求导、求积分和解微分方程。

(3) 数值模拟或计算机仿真。

问题:

1. 误差为何? 如何衡量?

2. 数值问题对误差的敏感性如何? 条件数

3. 算法对误差的敏感性如何? 稳定性

4. 算法的收敛性如何? 收敛快慢?

...

§2 误差分析基础

§2 误差分析基础

一、数值计算的近似

数值计算前的误差

模型误差：数学模型与实际问题的误差。

观测误差：由观测产生的误差。

数值计算过程的误差

截断误差(方法误差)：由算法造成的近似解与准确解之间的误差。

舍入误差：计算机进行数值计算时，每一步运算均需近似 (舍入)，由此产出的误差称为舍入误差。

二、绝对误差和相对误差

1. 误差与有效数字

准确值 x，其近似值 $\hat{x}$。

二、绝对误差和相对误差

1. 误差与有效数字

准确值 x，其近似值 $\hat{x}$。

(绝对) 误差：$e(\hat{x}) = \hat{x} - x$。

有量纲，可正可负。

二、绝对误差和相对误差

1. 误差与有效数字

准确值 x，其近似值 $\hat{x}$。

相对误差：$e_r(\hat{x}) = \frac{\hat{x}-x}{x} = \frac{e(\hat{x})}{x}$，

也可 $e_r(\hat{x}) = \frac{\hat{x}-x}{\hat{x}}$。

无量纲，可正可负。

如果相对误差的大小超过 100%，其计算结果完全错误。

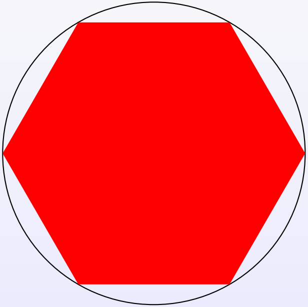
(绝对) 误差限 $\varepsilon(\hat{x})$：(绝对) 误差的绝对值上限。
相对误差限 $\varepsilon_r(\hat{x})$：相对误差的绝对值上限。

(绝对) 误差限 $\varepsilon(\hat{x})$：(绝对) 误差的绝对值上限。
相对误差限 $\varepsilon_r(\hat{x})$：相对误差的绝对值上限。
绝对值很大的数不宜讨论绝对误差 (限)，绝对值很小的数不宜讨论相对误差 (限)。

例：刘徽割圆术

正 192 边形，$\pi \approx 3.14$;

正 3072 边形，$\pi \approx 3.1416$。



例：铯原子钟

用于国防、天文、大地测量等

早期：王义遒

中国计量科学院：

NIM4，600 万年不差一秒；

NIM5，2000 万年不差一秒；

NIM6，5400 万年不差一秒。

例：高能同步辐射光源

10^{-9} 米，10^{-12} 秒。

误差“变形”

有效数字：从左至右第一位非零数字开始的所有数字。

有 p 位正确的有效数字的近似值。

保留 p 位有效数字的近似值。

误差“变形”

有效数字：从左至右第一位非零数字开始的所有数字。

有 p 位正确的有效数字的近似值。

保留 p 位有效数字的近似值。

例：$x = \pi = 3.14159265\cdots$ 无限位有效数字，有 4 位正确的有效数字 $\hat{x} = 3.141$，保留 4 位有效数字的近似值 3.142。

例：若 x 未知，若 $\hat{x}=21.8$，有 3 位正确有效数字的近似值，$\varepsilon(\hat{x})$ 和 $\varepsilon_r(\hat{x})$ 各为多少?
若 $\hat{x}=21.8$，是保留 3 位有效数字的近似值，$\varepsilon(\hat{x})$ 和 $\varepsilon_r(\hat{x})$ 各为多少?
若 $\hat{x}=21.8\times 10^k$，又如何?

有效数字位数越多，相对误差越小。
如：9.99×10^k 的相对误差限 $\frac{0.005}{9.99} \approx \frac{1}{2000}$。
即相对误差限小于 $\frac{1}{2000}$ 时，至少保留 3 位正确有效数字。
1.00×10^k 的相对误差限 $\frac{0.005}{1.00} = \frac{1}{200}$。即相对误差限大于 $\frac{1}{200}$ 时，保留正确有效数字位数达不到 3 位。

有效数字位数越多，相对误差越小。
如：9.99×10^k 的相对误差限 $\frac{0.005}{9.99} \approx \frac{1}{2000}$。
即相对误差限小于 $\frac{1}{2000}$ 时，至少保留 3 位正确有效数字。
1.00×10^k 的相对误差限 $\frac{0.005}{1.00} = \frac{1}{200}$。即相对误差限大于 $\frac{1}{200}$ 时，保留正确有效数字位数达不到 3 位。
小数点后精确几位，保留几位?
小数点后精确位数越多，绝对误差越小。

2. 截断误差与舍入误差

例：按 $\frac{f(x+h)-f(x)}{h}$ 计算 $f'(x)$。

解：$f(x+h) = f(x) + hf'(x) + \frac{h^2}{2} f''(\xi)$，$\xi \in (x, x+h)$。

2. 截断误差与舍入误差

例：按 $\frac{f(x+h)-f(x)}{h}$ 计算 $f'(x)$。

解：$f(x+h)=f(x)+hf'(x)+\frac{h^2}{2}f''(\xi)$，

$\xi\in(x,x+h)$。

截断误差限 $Mh/2$，$M=\max|f''(\xi)|$。

2. 截断误差与舍入误差

例：按 $\frac{f(x+h)-f(x)}{h}$ 计算 $f'(x)$。

解：$f(x+h) = f(x) + hf'(x) + \frac{h^2}{2}f''(\xi)$，$\xi \in (x, x+h)$。

假设 ε 为计算一次函数值的误差上界。舍人误差限为 $2\varepsilon/h$。

$\varepsilon_{out} = \frac{Mh}{2} + \frac{2\varepsilon}{h}$。当 $h = 2\sqrt{\varepsilon/M}$ 时总误差达到最小。

三、敏感性与误差估算

问题的条件数为输出与输入的误差，或相对误差的比值的绝对值。

绝对条件数

$cond_{abs}(x) = |f'(x)|(\approx \frac{|\Delta y|}{|\Delta x|})$,。

相对条件数 $cond(d) = |\frac{f'(x)x}{f(x)}|(\approx \frac{|\Delta y|/|y|}{|\Delta x|/|x|})$。

条件数反映了 y 关于 x 的敏感性，也称为病态性。

良态问题，病态问题。

误差估计

例：设 $f(x) = 3x - \frac{3}{2}x^2 + 2x^3 + \cdots$，对微小的 x 值，对以下近似值估计误差。

取近似值 $3x$，误差

取近似值 $3x - \frac{3}{2}x^2$，误差

取近似值 $-e^{-2x} + e^x$，误差

$$-e^{-2x} + e^x = 3x - \frac{3}{2}x^2 + \frac{3}{2}x^3 + \cdots$$

多元函数的误差估计：

设 $y = f(x_1, \ldots, x_n)$，实际数据为 $\hat{x}_1, \hat{x}_2, \ldots, \hat{x}_n$，而 $\hat{y} = f(\hat{x}_1, \ldots, \hat{x}_n)$，则

$$y - \hat{y} \approx \sum_{k=1}^{n} \frac{\partial f}{\partial x_k}(\hat{x}_1, \hat{x}_2, \ldots, \hat{x}_n)(x_k - \hat{x}_k)。$$

假设数据误差很小，则绝对误差限：

$$\varepsilon(\hat{y}) = \sum_{k=1}^{n} \left| \frac{\partial f}{\partial x_k}(\hat{x}_1, \hat{x}_2, \ldots, \hat{x}_n) \right| \varepsilon(\hat{x}_k)。$$

相对误差限：

$$\varepsilon_r(\hat{y}) = \sum_{k=1}^{n} \left| \frac{\partial f}{\partial x_k}(\hat{x}_1, \hat{x}_2, \ldots, \hat{x}_n) \right| \frac{\varepsilon(\hat{x}_k)}{|\hat{y}|}。$$

例：作为二元函数，四则运算的误差限

$\varepsilon(\hat{x}_1 \pm \hat{x}_2) \leq \varepsilon(\hat{x}_1) + \varepsilon(\hat{x}_2)$，

$\varepsilon(\hat{x}_1\hat{x}_2) \leq |\hat{x}_1|\varepsilon(\hat{x}_2) + |\hat{x}_2|\varepsilon(\hat{x}_1)$，

$\varepsilon\left(\frac{\hat{x}_1}{\hat{x}_2}\right) \leq \frac{|\hat{x}_1|\varepsilon(\hat{x}_2)+|\hat{x}_2|\varepsilon(\hat{x}_1)}{|\hat{x}_2|^2}$，$(\hat{x}_2 \neq 0)$。

四、算法的稳定性

算法的稳定性：计算过程中的扰动对计算结果的影响程度的大小。

(1) 若计算结果对计算过程中的舍入误差不敏感，则相应的算法为稳定的算法。

(2) 对于包含一系列步骤的计算过程，若计算中的小扰动不被放大 (传播) 或放大不严重，则相应的算法为稳定的算法。

§3 计算机浮点数系统

一、计算机浮点数系统

威廉・卡亨（1933. 6. 5–）

1989 年图灵奖获得者

William Morton Kahan，加拿大计算机学者，浮点运算的先驱。
贡献：浮点运算部件设计、浮点运算的标准 · · ·

浮点数系统：由浮点数及其运算构成。
$x \in \mathbb{F}$，$x = \pm(d_0.d_1d_2\cdots d_{p-1})_\beta \times \beta^E$，
整数 d_i 满足 $0 \le d_i \le \beta - 1$，
指数 E 满足 $L \le E \le U$，尾数
$m = (d_0.d_1\cdots d_{p-1})_\beta$。
浮点数系统由 β、p、L 和 U 确定。

浮点数系统：由浮点数及其运算构成。
$x \in \mathbb{F}$，$x = \pm(d_0.d_1d_2\cdots d_{p-1})_\beta \times \beta^E$，
整数 d_i 满足 $0 \le d_i \le \beta - 1$，
指数 E 满足 $L \le E \le U$，尾数
$m = (d_0.d_1\cdots d_{p-1})_\beta$。
浮点数系统由 β、p、L 和 U 确定。
为使有效数字位数尽可能高，E 尽量小。
当 $|x| \ge \beta^L$ 时，首位数字 $d_0 \ne 0$;
当 $|x| < \beta^L$ 时，$d_0 = 0$，指数 $E = L$。
历史上曾出现过多种浮点数系统。

例：一个简单的浮点数系统，其中，$\beta=2$，$p=3$，$L=-1$，$U=1$。

例：一个简单的浮点数系统，其中，

$\beta = 2$，$p = 3$，$L = -1$，$U = 1$。

$$(0.01)_2 \times 2^{-1} = 0.125,$$
$$(0.10)_2 \times 2^{-1} = 0.25,$$
$$(0.11)_2 \times 2^{-1} = 0.375;$$
$$(1.00)_2 \times 2^{-1} = 0.5,$$
$$(1.01)_2 \times 2^{-1} = 0.625,$$
$$(1.10)_2 \times 2^{-1} = 0.75,$$
$$(1.11)_2 \times 2^{-1} = 0.875;$$

同样的数，应让有效位数尽可能多

例如，$L=0$时，不再采用 $(0.11)_2 \times 2^0$ 而是用 $(1.10)_2 \times 2^{-1}$

$$
\begin{aligned}
&(1.00)_2 \times 2^0 = 1, \quad (1.01)_2 \times 2^0 = 1.25, \\
&(1.10)_2 \times 2^0 = 1.5, \quad (1.11)_2 \times 2^0 = 1.75; \\
&(1.00)_2 \times 2^1 = 2, \quad (1.01)_2 \times 2^1 = 2.5, \\
&(1.10)_2 \times 2^1 = 3, \quad (1.11)_2 \times 2^1 = 3.5。
\end{aligned}
$$

1985 年以后使用 IEEE 标准，它定义了两个特殊值。为使运算封闭

(1) Inf：表示无穷，如 $1/0$，$\beta * \beta^{U}$ 等操作可以得到。(不是真的无穷大，与操作系统有关) 最大是U，不能 β^{U+1}

(2) NaN：表示不是数，如 $0/0$，$0 * Inf$，Inf/Inf，1^{Inf} 等操作可以得到。

IEEE 单精度浮点数系统，$\beta = 2$，$p = 24$，$L = -126$，$U = 127$。快

IEEE 双精度浮点数系统，$\beta = 2$，$p = 53$，$L = -1022$，$U = 1023$。慢

MATLAB 默认采用双精度系统。

例：0.1 不是双精度浮点数。

解：$0.1 = (1.\dot{1}00\dot{1})_2 \times 2^{-4}$，在相邻浮点数 $(1.\underline{1001\cdots1001})_2 \times 2^{-4}$ 和 $(1.1001\cdots1010)_2 \times 2^{-4}$ 之间。 52位(53位-1)

比上一个相比进了一位

$(1.1001\cdots1001)_2\times2^{-4} < (1.\dot{1}00\dot{1})_2\times2^{-4} < (1.1001\cdots1010)_2\times2^{-4}$

MATLAB 默认采用双精度系统。

太大太小都容易不是双精度.

例： $2^{53}+1$ 不是双精度浮点数。
$2^{53}+1$ 在相邻浮点数
$2^{53}=(1.0\cdots00)_2\times 2^{53}$ 和 52个0
$2^{53}+2=(1.0\cdots01)_2\times 2^{53}$ 之间。
进1位

二、舍入与溢出

用浮点数系统计算，实数 x 舍入到浮点数 $fl(x)$。

Inf 位于 β^{U+1}。

两种常用的舍入原则:

(1) 截断舍入：将第 d_{p-1} 后的尾数截去。也称为“向零舍入法”。

(2) 最近舍入：$fl(x)$ 是与 x 最接近的浮点数。也称为“偶数舍入法”。

如果在中间. 舍入到最后一位为偶数的 (0)

例：$\mathbb{F}$：$\beta = 2$，$p = 3$，$L = -1$，$U = 1$。
$x = 1.375 = (1.011)_2$。$(1.01)_2 = 1.25$ 和 $(1.10)_2 = 1.5$。

最近舍入 $fl(0.0625) = fl((0.001)_2 \times 2^{-1}) = (0.00)_2 \times 2^{-1} = 0$，
$fl(1.375) = (1.10)_2 = 1.5$。(这体系统中，0后第一个浮点数是0.125，中间值是0.0625)

下溢：$|x| \leq 0.0625$ 时，$fl(x) = 0$。

上溢：$|x| \geq 3.75$ 时，$fl(x) = \pm\text{Inf}$。

(最后浮点数是3.5，下一个是4(inf)，中间是3.75)

截断舍入 $fl(0.0625) = fl((0.001)_2 \times 2^{-1}) = (0.00)_2 \times 2^{-1} = 0$，$fl(1.375) = 1.25$。

目前的计算机几乎都采用最近舍入。

采用最近舍入规则，单精度浮点数系统，大约有 7 位正确的有效数字。
双精度浮点数系统，大约有 15 位正确的有效数字。

三、浮点运算的舍入误差

例：$\mathbb{F}$：$\beta=2$，$p=3$，$L=-1$，$U=1$。

$x=3=(11.0)_2$，$y=0.25=(0.01)_2$。

$x,y\in\mathbb{F}$。

算术加法：$x+y=3.25$，而 11.01，四位有效数字，不在浮点数系统内，需舍入

$fl(x+y)=3=(11.0)_2$。

浮点加法：$x\ fl+\ y=3$。

一定还是浮点数

记号上对算术运算和浮点运算不加区别。
即：上述浮点系统下，$3 + 0.25 = 3$。
$(3 + 0.25) + 0.25 = 3 + 0.25 = 3$，而
$3 + (0.25 + 0.25) = 3 + 0.5 = 3.5$。
浮点算术不等于算术。 浮点系统下无结合律

病态问题有可能相差很大

例：采用 IEEE 单精度浮点数计算 $\sum_{k=1}^{\infty} \frac{1}{k}$。

实数体系中，发散级数。

不是所有计算都适合用浮点数算

二进制有效位数

```
def sumharmo(p):
    RFP = RealField(p)
    y = RFP(1.); x = RFP(0.); n = 1
    while x != y:
        y = x; x += 1/n; n += 1
    return p, n-1, x
print(sumharmo(2))
```

$p = 2$ 时，结果为(2，4，2.0)，即从 $n = 4$ 开始，上述和不再变化。
单精度浮点数，$p = 24$；
双精度浮点数，$p = 53$。

例 (Jean-Michel Muller):

$$u_n = 111 - \frac{1130}{u_{n-1}} + \frac{3000}{u_{n-1}u_{n-2}},$$

```
x = 2; y = -4;
z = 111 - 1130/y + 3000/(x*y);
for i in range(200):
    x,y = y,z
    z = 111 - 1130/y + 3000/(x*y)
print(float(z))
```

算的过程没有使用浮点数

结果是 6.0。有理数下收敛到 6。

```
x = RDF(2); y = -RDF(4);
z = 111 - 1130/y + 3000/(x*y);
for i in range(100):
    x,y = y,z
    z = 111 - 1130/y + 3000/(x*y)
print(z)
```

浮点数计算

每一步都有误差

结果是 100.0。双精度浮点数下收敛到 100。

递推公式 $u_n = 111 - \frac{1130}{u_{n-1}} + \frac{3000}{u_{n-1}u_{n-2}}$，只能收敛到方程 $x = 111 - \frac{1130}{x} + \frac{3000}{x^2}$ 的根，即 5，6 和 100。

递推公式 $u_n = 111 - \frac{1130}{u_{n-1}} + \frac{3000}{u_{n-1}u_{n-2}}$，只能收敛到方程 $x = 111 - \frac{1130}{x} + \frac{3000}{x^2}$ 的根，即 5，6 和 100。

通项为 $u_n = \frac{\alpha 100^{n+1} + \beta 6^{n+1} + \gamma 5^{n+1}}{\alpha 100^n + \beta 6^n + \gamma 5^n}$。

当初值 $u_0 = 2$，$u_1 = -4$ 时，$\alpha = 0$，$4\beta + 3\gamma = 0$。理论上收敛到 6。

100 是稳定的，5 和 6 是不稳定的。

带误差时会动态调整 $\alpha . \beta . \gamma$. 因此很容易收敛到 200

四、抵消现象

两个符号相同、值相近的 p 位数相减可能使结果的有效数字远少于 p 位，称这种现象为抵消。

舍入是丢弃末尾数位上的数字，而抵消丢失的是高位数字包含的信息，影响大得多。因此尽可能避免相近数相减。

例 (计算 e^x 时出现的抵消)：当 $x<0$，且 $|x|$ 较大时，利用公式 $e^x=1+x+\frac{x^2}{2!}+\frac{x^3}{3!}+\cdots$ 截断前 n 项计算 e^x，会发生严重的抵消，使数值结果误差很大。由于 $|x|$ 较大；则前若干项的求和式 (部分和) 中每一项的绝对值都远大于准确结果，当按自然顺序逐项累加到部分和上时，每次计算都是将两个较大但符号相反的数作加法，造成抵消。

MATLAB 中的计算结果：假设 $x=-20$，前 n 项求和的计算值为 $S_n(x)$。
$S_{96}(x)=5.62188\times10^{-9}$，
$x^{96}/96!=7.98930\times10^{-26}$。它与 $S_{96}(x)$ 的比值已经小于机器精度，后续的求和不会改变部分和的计算值，因此，e^{-20} 的计算值为 5.62188×10^{-9}(仅显示 6 位有效数字)，而 e^{-20} 的准确值为 2.06115×10^{-9}，计算结果完全错误。

当 $x > 0$ 时，求和式中每项都大于 0，不会有抵消现象，计算是稳定的。例如，当 $x = 20$ 时，计算前 68 项后结果将不再变化，部分和为 $S_{68}(x) \approx 4.85165 \times 10^8$，与准确值完全一样。因此，对于 $x < 0$ 的情况，通过公式 $1/e^{-x}$ 计算 e^x 是有效、可行的算法。

§4 保证数值计算的准确性

一、减少舍入误差的几条建议 变通.

1. 避免中间计算结果出现上溢或下溢

例：计算分式 $y=\frac{x_1}{x_2x_3\cdots x_n}$，其中 y 的准确结果不会超过上溢，而 x_1/x_2 超过上溢。

为避免上溢，先计算分母 $z=x_2x_3\cdots x_n$，然后计算 x/z。

2. 避免“大数吃掉小数”

大小相差悬殊的两数相加，较小数的信息将被“淹没”。一旦“大数吃掉小数”情况发生多了，必然造成很大的计算误差。

3. 避免符号相同的两相近数相减

为防止出现抵消现象，应避免避免符号相同的两相近数相减。

3. 避免符号相同的两相近数相减

为防止出现抵消现象，应避免避免符号相同的两相近数相减。

例：求 $x^2 - 16x + 1 = 0$ 的较小正根。

解：$x_1 = 8 + \sqrt{63}$，$x_2 = 8 - \sqrt{63}$

$\approx 8 - 7.94 = 0.06$ 只有 1 位有效数字。

若改用 $x_2 = 8 - \sqrt{63} = \frac{1}{8+\sqrt{63}}$

$\approx \frac{1}{15.94} = 0.0627$ 具有 3 位有效数字。

4. 注意简化步骤，减少运算次数

例：计算多项式的值，其中

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

直接计算需要 $\frac{n(n+1)}{2}$ 次乘法和 n 次加法运算。

霍纳算法：写 $P_n(x) = (\cdots((a_n x + a_{n-1})x + a_{n-2})x + \cdots + a_1)x + a_0$

需 n 次乘法和 n 次加法运算。

类似:

$a_3(x-x_0)(x-x_1)(x-x_2)+a_2(x-x_0)(x-x_1)\ +a_1(x-x_0)+a_0$

$=(a_3(x-x_1)(x-x_2)+a_2(x-x_1)+a_1)(x-x_0)+a_0$

$=((a_3(x-x_2)+a_2)(x-x_1)+a_1)(x-x_0)+a_0$。

例：计算 $\ln 2$，精确到 10^{-5}。

利用 $\ln(1+x)=\sum\limits_{n=1}^{\infty}(-1)^{n+1}\frac{x^n}{n}$， $x=1$。

对前 10^5 项求和。

利用 $\ln\frac{1+x}{1-x}=2\sum\limits_{n=1}^{\infty}\frac{x^{2n-1}}{2n-1}$， $x=\frac{1}{3}$。

对前 10 项求和。

二、影响结果准确性的主要因素

通常需按以下方面依次考虑。

(1) 病态性，是待求解数学问题的性质，与具体算法无关，最先考虑。

(2) 稳定性，是数值算法的性质，应选择稳定性好的算法，减少计算中误差的扩大。

(3) 通过定性分析控制舍入误差，遵循减少舍入误差的几条建议，若可能的话尽量采用位数较多的双精度浮点数。

附：十进制化为二进制方法：

例：整数部分除以 2 取余，逆序排列。

整数部分	商	余数
11	5	1
5	2	1
2	1	0
1	0	1
0	0	0

$$11 = (1011)_2$$

小数部分乘 2 取整，顺序排列。

小数部分	积	整数部分
0.1	0.2	0
0.2	0.4	0
0.4	0.8	0
0.8	1.6	1
0.6	1.2	1
0.2	0.4	0

$0.1 = (0.0\dot{0}01\dot{1})_2 = (1.\dot{1}00\dot{1})_2 \times 2^{-4}$。

$11.1 = (1011.0\dot{0}01\dot{1})_2$。

双精度浮点数存储

以 0.1 为例，相邻浮点数为
$x = (1.\underbrace{1001\cdots1001}_{52})_2 \times 2^{-4}$ 和
$y = (1.\underbrace{1001\cdots1010}_{52})_2 \times 2^{-4}$。
首位为符号位，正数 0，负数 1。首位存储为 0。
2-12 位存储 $e + 1023 = -4 + 1023$
$= 1019$，即存储为 01111111011。

前 12 位存储为 001111111011，写成十六进制，为 3fb。

后 52 位存储尾数小数点后的部分

x 和 y 的后 52 位分别存储为

$1001 \cdots 1001$ 和 $1001 \cdots 1010$。写成十六进制，分别为 3fb9999999999999，3fb999999999999a。

由于 y 离 0.1 较近，数值计算中以 y 代替 0.1 参与计算。实际参加计算的浮点数比 0.1 大一些

作业

习题

$e^x = 1 + x + \frac{1}{2}x^2 + o(x^2)$

$-e^{-2x} = -1 + 2x - 2x^2 + o(x^2)$

$-e^{-2x} + e^x = 3x - \frac{3}{2}x^2 + o(x^2)$

7. 设 $f(x) = -e^{-2x} + e^x$，对微小的 x 值，x，$3x$，和 $3x(1 - x/2)$ 哪个最精确？误差分别是多少？

8. 若在计算 $y = \sqrt{x^2 + 1} - 1$ 中至多丢失两位精度，该对 x 怎样限制？

$\sqrt{x^2+1}$ 大于等于 1.01 的时候损失两位

抵消现象，有限位

（x 很小的时候，$\sqrt{x^2+1}$ 与 1 相近）

7. $f(x) = -e^{-2x} + e^{x}$

泰勒展开： $f(x) = 3x - \frac{3}{2}x^2 + \frac{3}{2}x^3 + o(x^3)$

x 误差：$2x$

$3x$ 误差：$-\frac{3}{2}x^2$

$3x(1-\frac{x}{2})$ 误差：$\frac{3}{2}x^3$

8. $y = \sqrt{x^2+1} - 1$

$\sqrt{x^2+1}$ 与1接近时，易抵消现象

$\sqrt{x^2+1}$ ： $1.01 \to 3$位

1 ： $1.00 \to 3$位

$1.01 - 1.00$
$= 0.01 \to 1$位.

$\therefore \sqrt{x^2+1} \geq 1.01$

即 $x^2 \geq 0.0201$

$x \leq -\sqrt{0.0201}$

$x \geq \sqrt{0.0201}$