2018-2019学年第一学期

计 算 方 法

第一章 引论

主讲人：张治国
zgzhang@szu.edu.cn

深圳大学 医学部 生物医学工程学院
SHENZHEN UNIVERSITY HEALTH SCIENCE CENTER SCHOOL OF BIOMEDICAL ENGINEERING

本章内容

第一章 引论

§1 计算机数值方法的研究对象与特点

§2 数值方法的基本内容

2-1 数值代数的基本工具与方法

2-2 数值微积分的工具与方法

2-3 计算机数值方法

本章内容

第一章　引论

§3 数值算法及其设计

3-1 算法设计

3-2 算法表达式

§4 误差分析简介

✓ 4-1 误差的基本概念

4-2 浮点基本运算的误差

✓ 4-3 数值方法的稳定性与算法设计原则

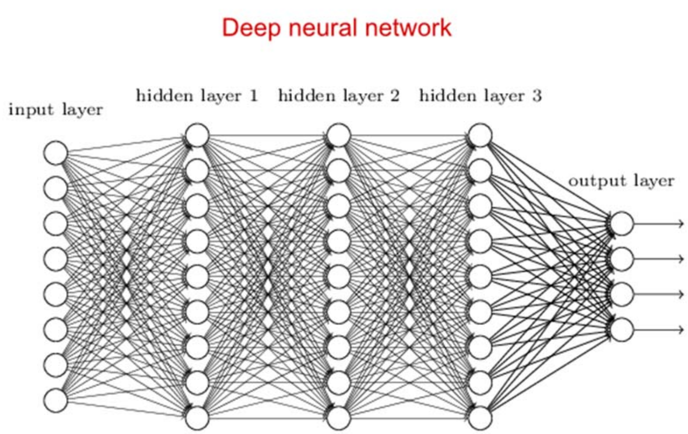
§1 计算机数值方法的研究对象与特点

- 计算方法：又称为数值方法、数值分析、计算数学等
- 计算方法是一种研究并解决数学问题的数值近似解方法，是一种在计算机上使用的解决数学问题的方法。
- 计算方法是一门古老的数学，例如计算圆周率、《九章算术》、牛顿和莱布尼兹等提出的微分、积分计算、等等。
- 计算方法是一门年轻的数学，近代计算机的诞生产生了数学的计算机计算。

§1 计算机数值方法的研究对象与特点

- 数学与计算机科学关系密切：数学理论不能完整解决实际问题，只有与计算机科学结合才能研制出实用的好算法。

例：深度学习中的深度神经网络（DNN）数学理论和架构早已完备，但是直到约10年前合适的算法提出后才得到广泛应用。



"A fast learning algorithm for deep belief nets"
-- Hinton et al., 2006

"Reducing the dimensionality of data with neural networks"
-- Hinton & Salakhutdinov, 2006

Geoffrey Hinton
University of Toronto

§1 计算机数值方法的研究对象与特点

- 计算方法是数学与计算机科学的交叉学科
- 数学的发展极大地促进了计算机科学的发展
  - Leibniz发现二进制编码
  - Von Neumann提出现代计算机建构理论
  - Bohm和Jacopini为结构化程序设计奠定了基础
- 计算机科学为数学提供先进手段，并对数学发展产生了重大影响
  - 为利用数学解决实际问题提供了工具
  - 解决了一些数学难题，并提出了新的研究课题
  - 促进了“并行算法”的研究和新一代计算机的发展

§1 计算机数值方法的研究对象与特点

- 计算方法的发展，产生了大量适合计算机求解的现代数值方法，大大推动现代科学的发展。
- 计算、理论、和实验并称为现代科学研究的三大支柱。

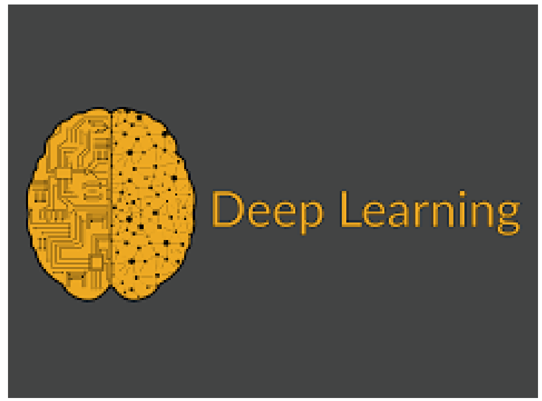
§1 计算机数值方法的研究对象与特点

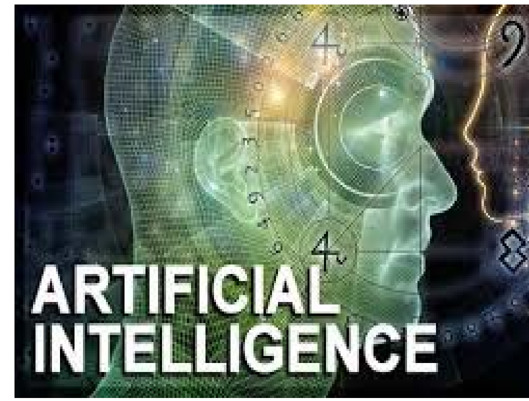
- 今日（21世纪信息/大数据/人工智能时代）的“计算方法”

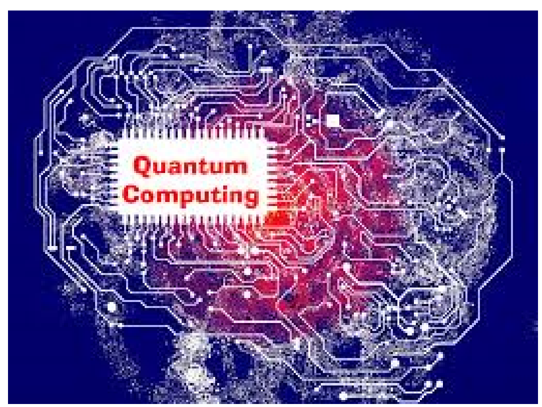
21世纪信息/大数据/人工智能时代的三个主要特征：	21世纪信息/大数据/人工智能时代对科技人才的要求：
▪ 数据无处不在	▪ 会用传感器获取数据
▪ 计算无处不在	▪ 会用数学建模实际问题
▪ 智能无处不在	▪ 会用计算机进行科学计算

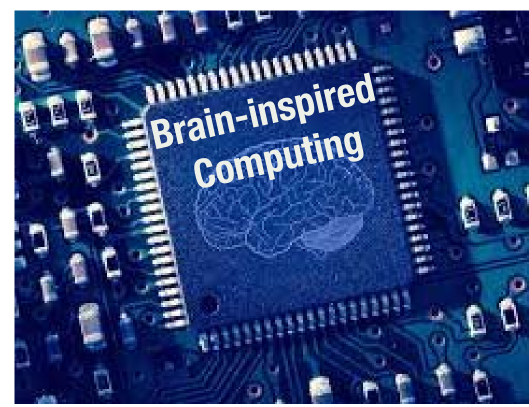
§1 计算机数值方法的研究对象与特点

数据 + 计算 = 智能







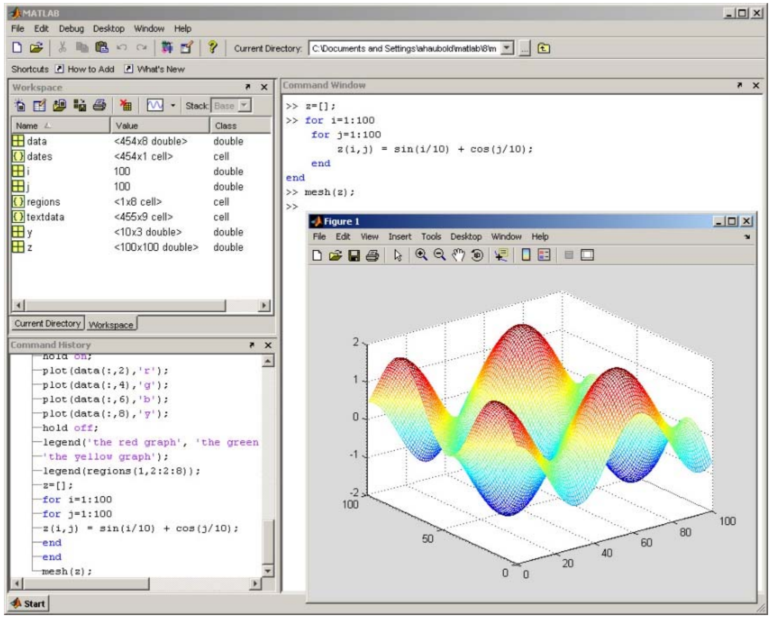


§1 计算机数值方法的研究对象与特点

- 使用计算方法解决实际问题，需要使用者数学基础和计算机编程能力都很强，这阻碍了计算方法的广泛应用。
- 现代计算方法的一个显著特点是已产生大量实用的综合数学软件库，并形成了数值软件产业。
  - MATLAB：集数值计算、图形演示等一体的综合数值软件库
  - Mathematica：综合数学软件包，集符号演算、数值计算和图形演示
  - R：自由软件程序语言与操作环境，主要用于统计、绘图、数据挖掘
  - Python：面向对象直译式的程序语言，包含了多种第三方算法库

§1 计算机数值方法的研究对象与特点

- 本课程部分作业要求使用MATLAB完成



下载地址：

- ftp://116.13.96.74/matlab/matlab_2018a_szu.zip

教程：

- https://ww2.mathworks.cn/help/matlab/getting-started-with-matlab.html
- http://read.pudn.com/downloads156/ebook/691210/matlab%E5%85%A5%E9%97%A8%E6%95%99%E7%A8%8B.pdf

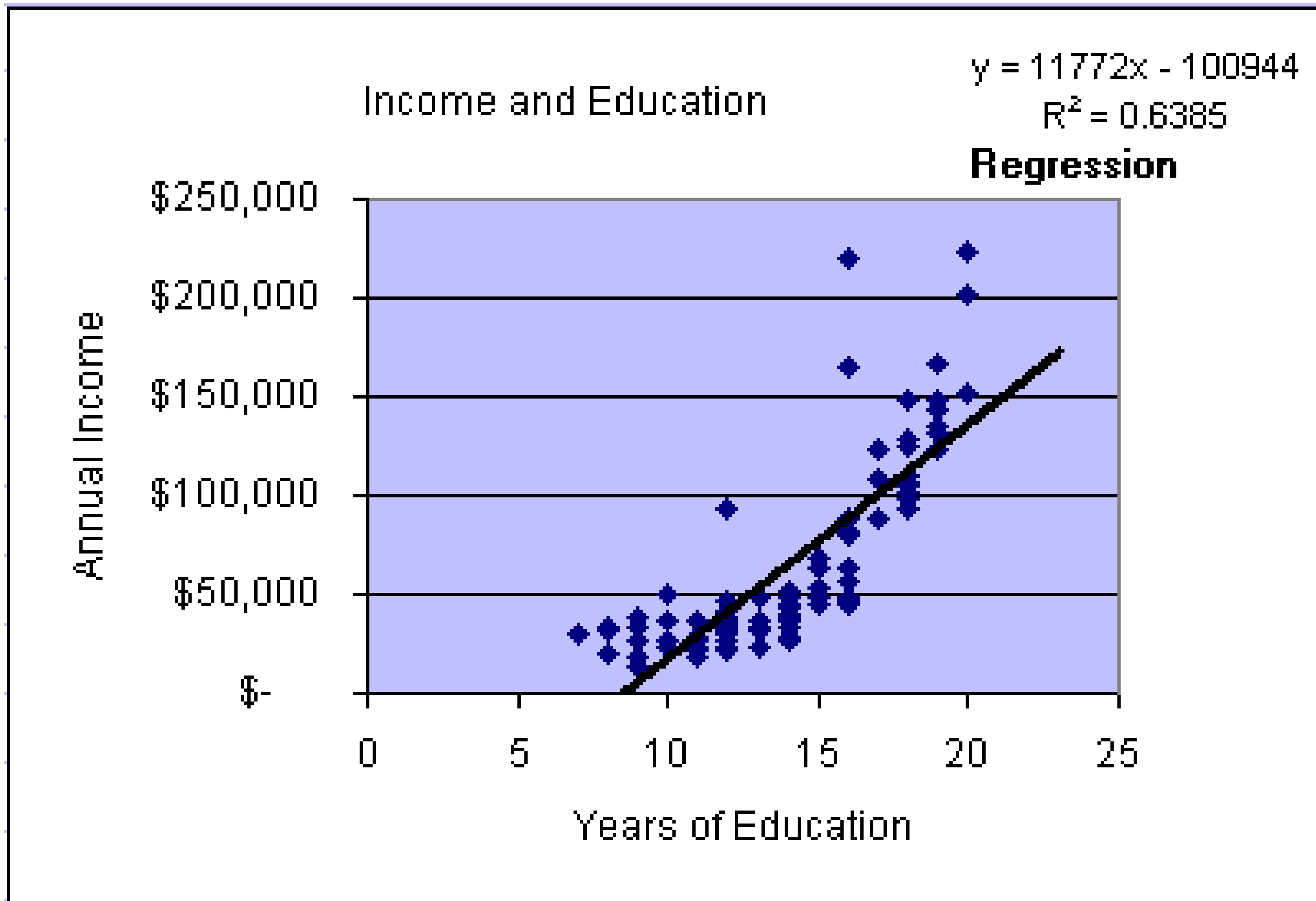
§1 计算机数值方法的研究对象与特点

- 为什么要学习计算方法这门课?
  - 利用计算机求解实际问题的核心过程，非常重要；
  - 虽然已有大量数值算法的软件包，但需要我们了解算法设计的原理，以便更好地应用；
  - 随着计算机的应用越来越广泛，数据量越来越大，计算问题越来越复杂，规模越来越大，现成的数值方法软件包不能满足特定需要，如信号处理、数字图像处理、模型仿真、天气预报、地震预测、股市预测等。

本课程为即将从事工程技术工作和科学研究的本科生打下使用计算机解决数值问题的基础。

§1 计算机数值方法的研究对象与特点

- 例：计算两个变量的线性关系



§1 计算机数值方法的研究对象与特点

- 本课程特点：
  - 不追求完美的数学演绎、论证和详尽的公式推导
  - 不以数学课程的类别教授计算方法
  - 以计算方法内在联系为主线，着重介绍计算方法及它们之间的关系和结构
  - 尽量结合生物医学工程的应用举例

- 本课程目标：
  - 了解与掌握一般常用的计算方法
  - 为进一步研究新算法奠定基础
  - 掌握如何在生物医学工程应用中使用计算方法

§1 计算机数值方法的研究对象与特点

- 本课程内容：

主要内容

1) 数值方法的基本内容 (§2简介)
   将微积分与代数中的数学问题化为数值问题，进行简化后形成数值方法，使其成为计算目标的最简单形式并便于在计算机上计算
2) 算法及其设计 (§3简介)
   为缩小数值方法与程序设计的距离，介绍有代表性的算法设计，研究在计算机上可执行且有效的一系列计算公式
3) 误差分析简介 (§4简介)
   简单介绍 (i) 数值问题对误差的敏感性及其度量与判别，(ii) 算法误差在运算过程中能否得到控制

§2 数值方法的基本内容

- 利用数学和计算机知识解决实际问题可分两步：

  1. 建立数学模型：需要利用有关专业知识和数学理论，这属于应用数学范围；

  2. 提出数值问题与数值方法：将数学模型变成数值问题，研究求解数值问题的数值方法，进而设计有效的数值算法的过程，这属于计算方法的范围。

§2 数值方法的基本内容

- 数值问题： 输入数据与输出数据之间函数关系的一个确定而无歧义的描述，包括三种类型：
  1) 分析问题（连续问题）
  2) 代数问题（离散问题）
  3) 概率与统计问题（随机问题）

- 数值方法：指求解“数值问题”在“计算机上可执行”的一系列运算和数据。
  - “计算机上可执行”的运算只能是四则运算和逻辑运算（与、或、非等）。

§2 数值方法的基本内容

- 研究数值方法时，必须考虑计算机运算的特点。
- 计算机运算基础：
  - 采用二进制表示数据，用具有固定位数的单元存储数据。因此，计算机处理和存储的数据离散化/数字化，精度有限。
  - 计算由逻辑电路实现，其执行运算存储的最基本单元是门电路，功能有限。因此，复杂运算必须转化为简单运算（例如，四则运算都用加法实现）。

§2 数值方法的基本内容

- 什么“数学模型”是“数值问题”？

例1. 求方程 $ax^2+bx+c=0$ 的根

输入 a，b，c，则输出数据是根 x_1 和 x_2，故是“数值问题”。

例2. 求常微分方程的解

$$\begin{cases} y' = 2x+3, \quad x \in [0,a] \\ y(0) = 0 \end{cases}$$

因输入数据为2和3，$x=0$ 和 $y=0$ 等，输出是函数 $y=x^2+3x$ ，故不是“数值问题”。

§2 数值方法的基本内容

- 如何将“数学模型”变成“数值问题”？
- 将非“数值问题”转化为“数值问题”的方法：“离散化”。
- 例如， 将求解 $y = x^2 + 3x$ 的问题变成求

$y(x_1), y(x_2), \cdots, y(x_n)$ ，$0 < x_1 < x_2 < \cdots < x_n = a$,

如可将 x_1 , x_2 , …… , x_{100} 取为 0.1, 0.2, 0.3, …

…,10=a 。

§2 数值方法的基本内容

- 计算公式不一定都是数值方法，如求 $\sqrt{3}$ 。
- 类似地，求根公式 $x_{1,2}=\dfrac{-b\pm\sqrt{b^2-4ac}}{2a}$

  不能在计算机上直接运行（因含平方根运算）。
- 方法：将开方运算化成计算机上可执行的等价或近似运算；然后写成标准函数如SQRT(D)，供其它程序调用。

§2 数值方法的基本内容

- 计算机上不能直接执行的运算包括：开方、超越函数、极限、微分、积分等。
- 以上运算需要利用等价或近似等价的方法转化，使其可在计算机上运算。
- 转化例子
  - 超越函数 e^x 应化为 $e^x \approx 1 + x + \dfrac{x^2}{2!} + \cdots + \dfrac{x^n}{n!}$
  - 函数 $y(x)$ 的导数 $y'(x)$ 的计算应化为

$$y'(x) \approx \frac{y(x+h) - y(x)}{h}$$

§2 数值方法的基本内容

- 本课程重点是介绍新型（以计算机为工具的）有效的数值方法。
  - 一些传统方法未考虑以计算机为工具进行计算的特点（近似计算，计算复杂度等问题），必须要研究计算机为工具的数值方法
- 数值方法的有效性和数值问题本身的好坏（数值问题的形态）有关，本课程将对此做概念和结论性介绍。
- 数值方法是数值算法的主要内容，但不是全部。

§3 数值算法及其设计

- 数值算法（简称算法）：有步骤地完成求解数值问题的过程。

- 数值算法的解题过程必须具备以下五个特性：
  1) 目的性：算法目的明确，条件和结论均应明确
  2) 确定性：算法必须精确地给出每一步的操作
  3) 可执行性：算法中的每个操作都是可执行的
  4) 有穷性：算法必须在有限步内结束解题过程
  5) 通用性：算法应是针对某一类通用问题的计算

§3 数值算法及其设计

- 计算机上的算法分类
  - 按求解问题的不同可分为：
    - 数值算法
    - 非数值算法（符号推理、公式推导）
  - 按面向计算机的不同可分为：
    - 面向串行计算机的串行算法，只有一个进程
    - 面向并行计算机的并行算法，含两个以上的进程
  - 根据算法内部的特点可分为：
    - 确定性算法，每完成一步确切知道下一步该做什么
    - 非确定性算法（智能算法）

§ 3 数值算法及其设计

- 计算机数值方法只研究计算机上串行确定型数值算法。
- 串行确定型数值算法通过按规定顺序执行一个完整且有限的运算序列后，将输入数据向量变成输出数据向量。

  - 例：给出等差数列1,2,3,……,10000的求和算法

    (1). 取 N=0，S=0 ；

    (2). N+1→N，S+N→S；

    (3). 若 N<10000 转(2)，否则

    (4). 输出N和S

§3 数值算法及其设计

- 算法设计中选择数值方法非常重要。
  - 例如对于大型数值问题，不同算法及其计算复杂性有很大差异：
    - 利用Gramer法则求解20阶线性方程组，需要乘除法运算次数共需 9.7×10^{20} 次；
    - 利用Gauss消去法求解，只需2670次乘除法。

- 算法设计的主要目的：

1) 研制可靠性好的方法（精度和稳定性要求）
2) 选择计算复杂性小的方法（速度快存贮少）
3) 为程序设计作好准备（方便编码和维护）

§3 数值算法及其设计

- 算法是编码的主要依据，算法设计是数值软件中详细设计的主要内容。
- 目前流行的数值软件开发方法有两种：
  1) 面向过程的“自顶向下、逐步细化”的结构化方法；
  2) 面向对象的“自下而上”的组装开发方法，其主要工具是“类”（特殊模块），利用它可组装算法。
- 本课程只介绍“自顶向下，逐步细化”法，其关键有三个方面：
  1) 划分模块，原则是模块功能要单一，即独立性好；
  2) 设计或选择模块算法，先总体，后具体；
  3) 充实细节，考虑计算公式的效率和其他要求。

§3 数值算法及其设计

- 在数值软件中，算法常用的表达方法有两类：
  - 自然语言法：用文字有步骤的表示算法
  - 图示法：又分为“流程图”和“结构化框图”

- 下面以求解二次方程为例说明算法设计

$$ax^2 + bx + c = 0$$

  - 解上述方程有两种方法：直接法、迭代法
  - 利用直接法需要考虑三个细节：

$$x_{1,2} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

1) 判别式：$d = b^2 - 4ac$ 大于或小于0；
2) 当 $d > 0$ 且 $\sqrt{d} \approx b$ 时，会出现两个近似数相减而影响有效数字的位数；
3) 若 $|a|$ 比 $|b|$ 和 $|c|$ 小很多时，可能出现舍入误差增大的问题。

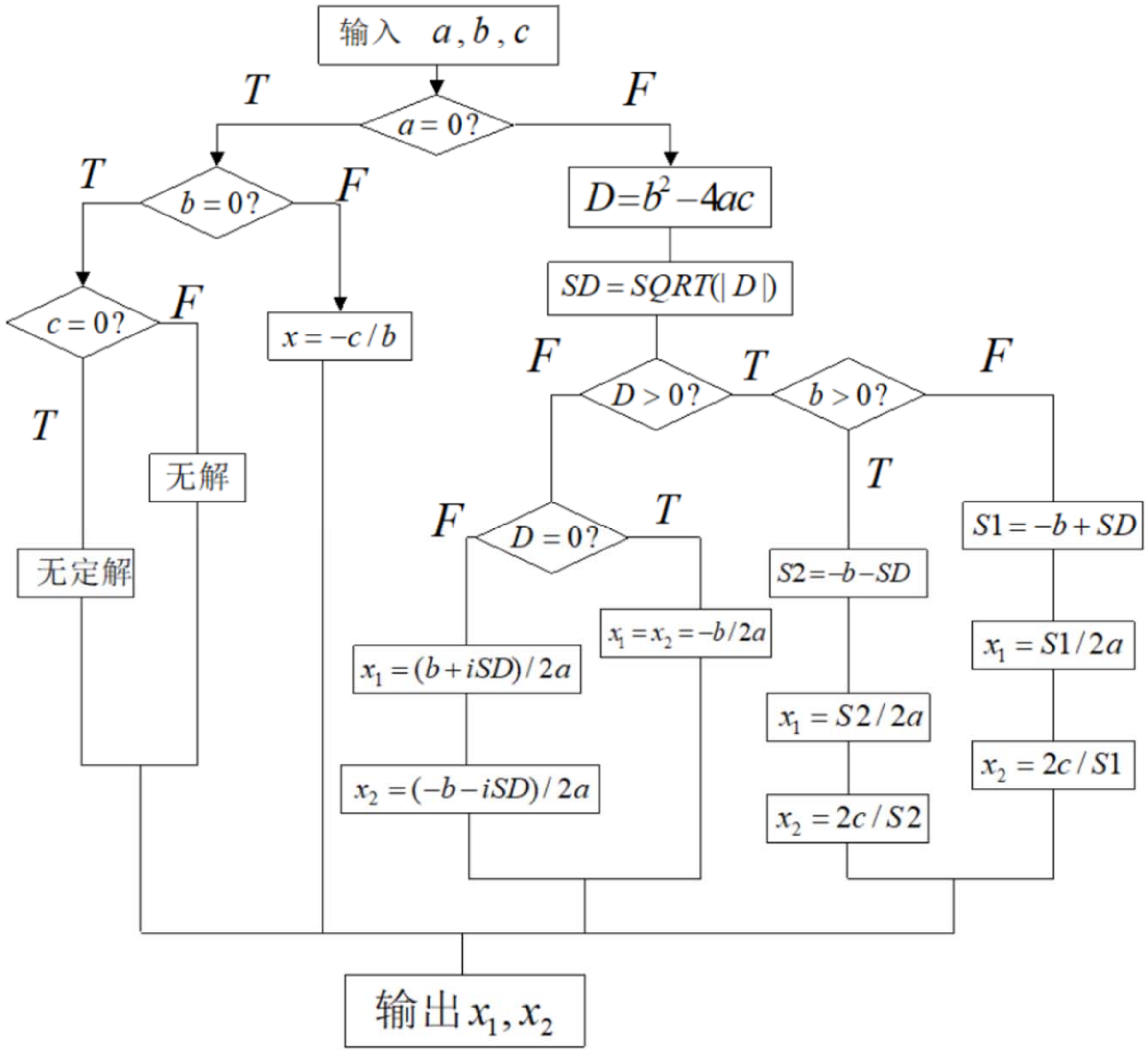
§3 数值算法及其设计

一. 自然语言法

1) 输入数据 a, b, c
2) 如果 $a = 0$，转3)，否则转4)
3) 如果 $b \neq 0$ ，则 $x_1 = -c/b$，转7)；否则，无解停机
4) 设 $D = b^2 - 4ac$ ， $SD = SQRT(|D|)$
   如果 $D = 0$, $x_1 = x_2 = -b/2a$ ，转7)
   如果 $D < 0$, $x_1 = (-b + iSD)/2a$, $x_2 = (-b - iSD)/2a$，转7)
   否则
5) 如果 $b < 0$，$S1 = -b + SD$，$x_1 = S1/2a$, $x_2 = 2c/S1$ ，转7)
   否则
6) $S2 = -b - SD$ ，$x_1 = S2/2a, x_2 = 2c/S2$
7) 输出 x_1 和 x_2

§3 数值算法及其设计

二. 图示法之流程图



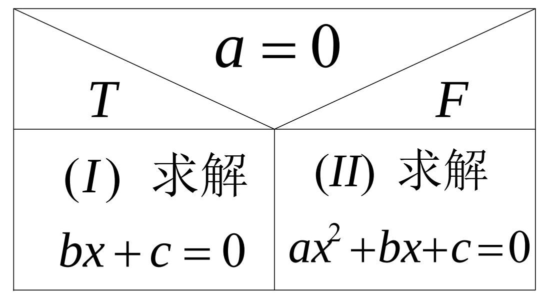
§ 3 数值算法及其设计

二. 图示法之结构化框图法（N-S图示法）

1. 顶层设计：

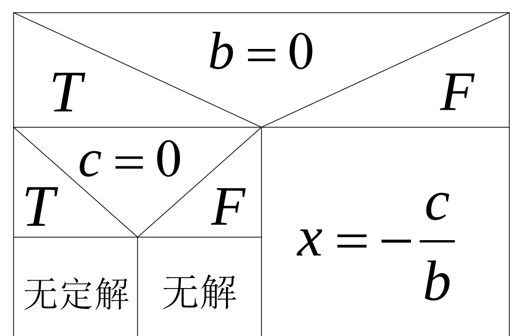
(I) 输入a,b,c
(II) 求解$ax^2+bx+c=0$
(III) 输出根x_1,x_2

2. 第1层设计：细化 1.(II)



3. 第2层设计：

a) 细化 2.(I)

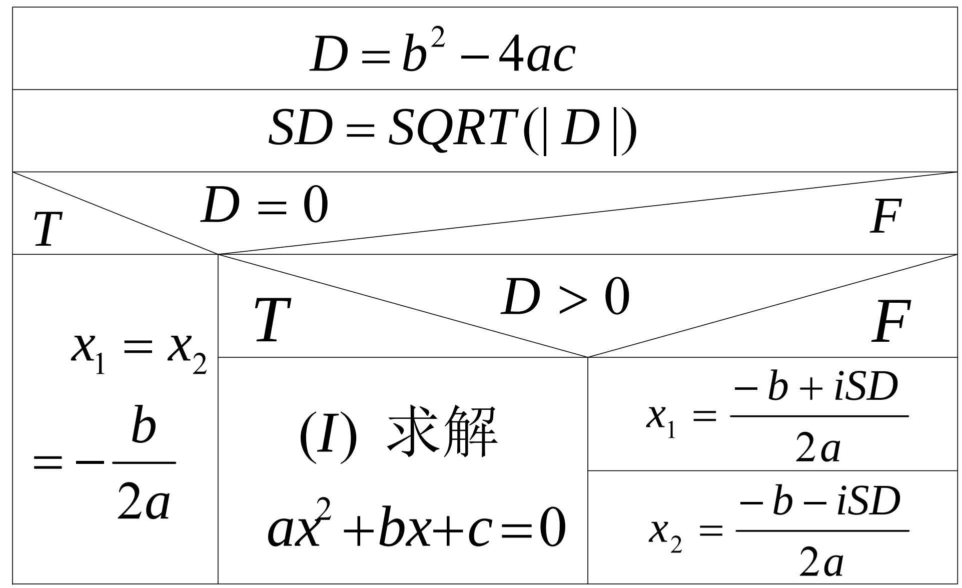


§3 数值算法及其设计

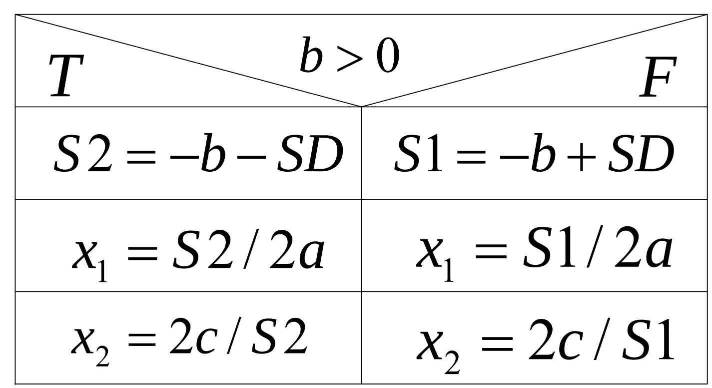
二. 图示法之结构化框图法（N-S图示法）－续

3. 第2层设计：

b）细化 2.(II)



4. 第3层设计：细化



§4 误差分析简介

4−1 误差的基本概念

- 数值计算中的误差来源有两种
  1) 舍入误差：由于计算机的字长有限，原始数据在计算机中的表示、运算产生的误差；
  2) 截断误差：由数学问题化成数值问题产生的误差

- 例（舍入误差）：
  若计算机仅能表示6位十进制数，则将 π 表示为

$$\pi^* = 3.141\,59\text{，误差}\quad R_1 = \pi - \pi^* = 0.000\ \ 0026\cdots$$

若将其与数 9.210 00 进行加法运算，得

$$s = 3.141\ \ 59 + 9.210\ \ 00 \approx 1.235\ \ 16 \times 10$$

$$R_2 = 12.351\ \ 6 - 12.351\ \ 59 = 0.000\ \ 01$$

R_1 和 R_2 都是<u>舍入误差</u>。

4-1 误差的基本概念

- 例（截断误差）：
  计算 e^x 的数值。因

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

由算法的有限性，故利用截断部分和

$$p(x) = 1 + \frac{1}{1!}x + \frac{1}{2!}x^2 + \cdots + \frac{1}{n!}x^n$$

来近似代替，由此产生的误差即为截断误差，为

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

4-1 误差的基本概念

- 例（截断误差）：
  计算 e^x 的数值。因

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

MATLAB代码：取 $x = 2, n = 10$

```
x = 2;
S = 1;
for n = 1:10
     S =  S + 1/factorial(n) * x^n;
end
```

Matlab 运行结果 7.3890，而 $e^x = 7.3891$。

若取 $n = 5$，运行结果 7.2667 。

4-1 误差的基本概念

- 绝对误差：设 x 为准确值，x^* 为 x 的一个近似值，称

$$E(x^*) = x^* - x$$

为近似值 x^* 的（绝对）误差，简记为 E。

- 例：圆周率的近似值 $a = 3.14$，绝对误差 $E = 0.00159\cdots$

- 一般来说，E 的准确值很难求出，只能估计出 $|E|$ 的某个上界 $\varepsilon(x^*)$，即

$$|E| = |x - x^*| \le \varepsilon(x^*)$$

$\varepsilon(x^*)$ 称为近似值 x^* 的（绝对）误差限，简记为 ε 。
于是 $x^* - \varepsilon \le x \le x^* + \varepsilon$ 可表示成：$x = x^* \pm \varepsilon$ 。

4-1 误差的基本概念

- 相对误差：近似值 x^* 的误差与准确值 x 之比

$$E_r(x^*) = \frac{E(x^*)}{x} = \frac{x^* - x}{x}$$

称为近似值 x^* 的相对误差，简记为 E_r 。

- E_r 绝对值的任一上界 $\varepsilon_r(x^*)$，称为相对误差限 ε_r，

$$|E_r(x^*)| = \left|\frac{E(x^*)}{x}\right| = \left|\frac{x^* - x}{x}\right| \le \varepsilon_r(x^*)$$

- 由于 x 的准确值难以确定，通常利用

$$E_r^* = E_r^*(x^*) = \frac{E(x^*)}{x^*} \text{ 代替 } E_r(x^*)$$

- 注：绝对误差限与相对误差限不唯一，它们越小越好。

4-1 误差的基本概念

- 对于准确值 x 取近似值最常用的方法是采用"四舍五入"的原则。由此产生了一个专有名词—有效数字。
- 有效数字： 如果近似值 x^* 的误差的绝对值不超过某一位数字的半个单位，且该位数字到 x^* 的第一位非零数字共有 n 位，则称用 x^*（近似 x 时）具有 n 位有效数字。（最大的 n 值）
- 例如：$\pi^* = 3.1416$ 有 5 位有效数字，

  而 $\pi^* = 3.1415$ 有 4 位有效数字。

4-1 误差的基本概念

- 确定有效数字的等价方法：数的规格化表示

$$x = \pm 10^{k} \times 0.\alpha_1\alpha_2 \cdots \alpha_n\alpha_{n+1} \cdots \qquad (4.2)$$

其中，k 是某个范围内的整数，$\alpha_1 \neq 0$，α_i 为 0, 1, … , 9 中的数字。

- 若将近似值 x^* 表示成 (4.2) 式，且 x^* 满足不等式

$$|x^* - x| \leq 0.5 \times 10^{k-n} \quad \text{（最大的 } n \text{ 值）} \qquad (4.5)$$

则称 x^* 具有 n 位有效数字。

4-1 误差的基本概念

- 例：$\pi^{*}=3.1416=10\times 0.31416$，有 $|\pi^{*}-\pi|\le 0.5\times 10^{1-5}$，因此其有五位有效数字。

- 特殊情况：有效数字不唯一。如：

  $x_1^{*}=4.0$ 和 $x_2^{*}=3.9$ 都是 $x=3.95$ 的2位有效数字。

- 一般地，近似值的有效位数越多，误差的绝对值越小。但也有个别例外，如对于 $x=1000$：

  $$x_1^{*}=999.9\text{，}\quad x_2^{*}=1000.1$$

  分别有3和4位有效数字。

4-1 误差的基本概念

- 有效数字位数与小数点的位置无关。只有经过四舍五入写成如(4.2)的规格化形式后，小数点后的数字位数才能反映其有效位数的多少。

- 有效数字小数点后面的零不能随便增减。

- 通过对某个数进行四舍五入取近似值可以得到它的有效数字。不是通过四舍五入得到的近似数，它的数字并不都是有效数字。

4-1 误差的基本概念

- 近似值的有效数字与相对误差之间的关系：

  定理4.1 设 x^*是 x 的近似值，它的表达式为

  $$x^* = \pm 10^k \times 0.\alpha_1 \alpha_2 \cdots \alpha_n$$

  则 x^*的有效位数与 x^*的相对误差之间有如下关系：

  1) 若 x^*具有 n 位有效数字，则 x^* 的相对误差 E_r^* 满足

  $$|E_r^*| \le \frac{1}{2} \times 10^{-n+1} \tag{4.6}$$

  2) 若 x^* 的相对误差 E_r^* 满足 $|E_r^*| \le \frac{1}{2} \times 10^{-n}$ (4.7)

  则 x^* 至少具有 n 位有效数字。

  证明：p19

- 近似值的有效位数越多，相对误差越小；反之，相对误差越小，近似值有效数字的位数就可能越多。

4-1 误差的基本概念

- 例： 设 $x=1.986$，$x^*=1.98$，则

$$|E_r^*|=\frac{|x^*-x|}{|x^*|}=\frac{|1.986-1.98|}{1.98}\le\frac{1}{2}\times10^{-2}$$

故，x^*至少具有2位有效数字。

- 因 $|x^*-x|=0.006\le0.5\times10^{-1}$ ，由(4.5)，$k-n=-1$， $k=1$，$n=k+1=2$，所以 x^*具有2位有效数字。

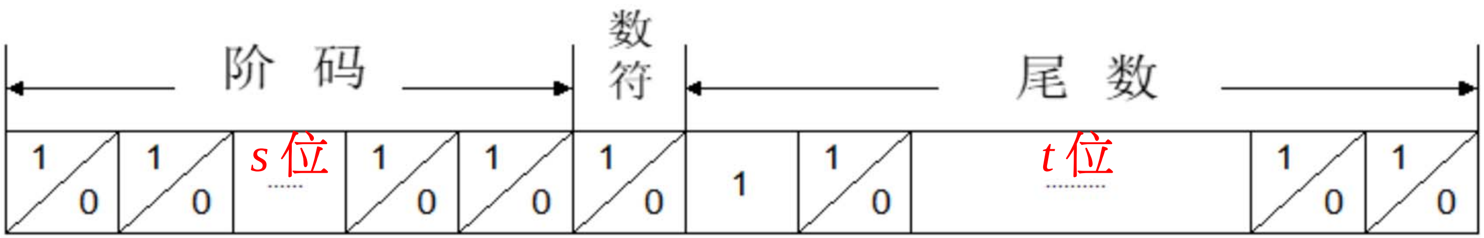
- 另一方面： x^*具有2位有效数字，故由(4.6)，得

$$|E_r^*|\le0.5\times10^{-2+1}$$

注：上界 $0.5\times10^{-2+1}$ 并不是最小的。

4-2 浮点基本运算的误差

- 在计算机中，每个数都用有限位二进制数表示，其规格化浮点数形式由三部分组成：
  1) 阶码：确定小数点的位置；决定了数的取值范围
  2) 数符：表示正、负号； ‘0’表示正， ‘1’表示负
  3) 尾数：表示机器数字长；长度与精度有关



- 形式为：$x^* = \pm 2^{\alpha} \times 0.\beta_1\beta_2\cdots\beta_t$ ，其中，2称为浮点数的基，$\alpha \in [-2^s, 2^s - 1]$，$\beta_1 = 1$，$\beta_2, \beta_3, \cdots, \beta_t$是0或1；称 x^*为 x 的浮点数，记为 $x^* = fl(x)$。

4-2 浮点基本运算的误差

- 设原始数据 $x = 2^{\alpha} \cdot \beta$ ，其尾数

$$\beta = \pm 0.\beta_1 \beta_2 \cdots \beta_t \cdots$$

若浮点数 $fl(x)$ 的尾数 β^* 是 t 位：$\beta^* = \pm 0.\beta_1 \beta_2 \cdots \beta_t$

则 $$fl(x) = 2^{\alpha} \cdot \beta^*$$

$$0.10\cdots 0 \le |\beta^*| \le 0.11\cdots 1$$

$$2^{\alpha}(\beta^* - 2^{-t}) \le x \le 2^{\alpha}(\beta^* + 2^{-t})$$

于是绝对误差 $E = |x - 2^{\alpha}\beta^*| = |x - fl(x)| \le 2^{\alpha - t}$

- $fl(x)$ 的相对误差

因 $|x| = |2^{\alpha} \cdot \beta| \ge 2^{\alpha} \cdot 0.10\cdots = 2^{\alpha-1}$，故 $|\varepsilon| = |E/x| \le 2^{-t+1}$ 。

4-2 浮点基本运算的误差

一. 浮点数的四则运算的误差

- 浮点数 x 和 y 的四则运算结果的浮点表示：

$$fl(x \pm y) = (x \pm y)(1 + \varepsilon_{1,2})$$

$$fl(x \cdot y) = (x \cdot y)(1 + \varepsilon_3)$$

$$fl\left(\frac{x}{y}\right) = \left(\frac{x}{y}\right)(1 + \varepsilon_4)$$

其中，$|\varepsilon_i| \le 2^{-t+1}$， $i = 1,2,3,4$ 。

4-2 浮点基本运算的误差

二. 连加和连乘的误差

- 设 $x_1, x_2, \cdots, x_n$ 为规格化浮点数，求

  $$fl(x_1 + x_2 + \cdots + x_n) \text{ 和 } fl(x_1 \cdot x_2 \cdot \cdots \cdot x_n)$$

  计算从左向右进行，每次运算都进行截取，再作下一次运算。

- 连和运算而言：
  - 各相对误差限的大小与运算先后有关；
  - 先运算的数，误差也较大；
  - 运算应先安排小数参加运算，可防止“大数吃小数”。

- 连乘运算而言：
  - 各相对误差限的大小与运算先后无关。

4-3 数值方法的稳定性与算法设计原则

- 设计和选择算法首要关心的问题是精度要求，要建立一些定性分析准则用于判断结果的可靠性，这是数值稳定性问题。
- 对于一个数值方法，若对原始数据或某一步有舍入误差，在执行过程中，这些误差能得到控制，则称该数值方法稳定。否则，称为不稳定的。
- 利用两种数值方法A和B解输入和舍入误差规则相同的同一问题，若用A法比B法得到的计算解精度更高，则称A法比B法具有较大的稳定性。

4-3 数值方法的稳定性与算法设计原则

- 例：计算积分 $I_n = \dfrac{1}{e}\int_0^1 x^n e^x dx$， $n = 0,1,2,\cdots,7$

解：由于 $eI_n = x^n e^x \big|_0^1 - n\int_0^1 x^{n-1} e^x dx = e - neI_{n-1}$

故 $I_n = 1 - nI_{n-1}$ (4.16)

方法一：利用(4.16)式，先计算I_0，然后计算I_1, …, I_7 。设计算值 I_i^* 的误差为$\varepsilon(I_i^*)$，若 I_0时误差为δ，则

$$\varepsilon(I_1^*) = \delta, \varepsilon(I_2^*) = 2!\delta, \cdots, \varepsilon(I_7^*) = 7!\delta$$

方法二：利用公式 $I_{n-1} = \dfrac{1 - I_n}{n}$ 先计算I_7，再计算 I_6, I_5, …, I_0 。计算 I_7时产生误差 δ，那么计算 I_0时的误差为 $\varepsilon(I_0^*) = \dfrac{\delta}{7!}$ 。

4-3 数值方法的稳定性与算法设计原则

- 例:求线性方程组

$$
\begin{cases}
x_1 + \frac{1}{2}x_2 + \frac{1}{3}x_3 = \frac{11}{6} \\
\frac{1}{2}x_1 + \frac{1}{3}x_2 + \frac{1}{4}x_3 = \frac{13}{12} \\
\frac{1}{3}x_1 + \frac{1}{4}x_2 + \frac{1}{5}x_3 = \frac{47}{60}
\end{cases}
$$

其准确解为: $x_1 = x_2 = x_3 = 1$

但是，若对数据取3位有效数字，利用Gauss消去法（见第二章）求解，则得到：$x_1 \approx 1.09,\ x_2 \approx 0.488,\ x_3 \approx 0.491$

4-3 数值方法的稳定性与算法设计原则

- 同样对方程
$$\begin{cases} 2x_1 - x_2 + 3x_3 = 1 \\ 4x_1 + 2x_2 + 5x_3 = 4 \\ 2x_1 \qquad\quad + 2x_3 = 6 \end{cases}$$
利用Gauss消去法，取3位有效数字，则得到准确解：
$$x_1 = 9, \quad x_2 = -1, \quad x_3 = -6$$
- 数值问题计算解的精度，不但与数值方法的稳定有关，而且还与数值问题的性态好坏有关。
- 在数值问题中，若输出数据对输入数据的扰动（误差）很敏感（小的变化会引起较大的变化），称这类数值问题为病态问题；否则称为良态问题。

4-3 数值方法的稳定性与算法设计原则

一. 四则运算中的稳定性问题

1) 防止大数吃小数

$$0.3684676+10^7\times0.6327544+0.4932001+0.4800100$$

$$=10^7\times0.6327544$$

- 预防方法：先加小数，由小到大逐次相加。如

$$0.3684676+0.4800100+0.4932001+10^7\times0.6327544$$

$$=10^7\times0.6327545$$

4-3 数值方法的稳定性与算法设计原则

一. 四则运算中的稳定性问题

2) 要避免两个相近数相减

相近数相减会严重损失有效数值的位数

$1-\cos(x)$ 可改为 $2\sin^2(x/2)$

3) 避免小数作除数和大数作乘法

这样可避免误差放大，因

$$|E(x_1x_2)| \le |x_1|\cdot|E_2| + |x_2|\cdot|E_1|$$

$$|E(x_1/x_2)| \le \{|x_1|\cdot|E_2| + |x_2|\cdot|E_1|\}/|x_2|^2$$

4-3 数值方法的稳定性与算法设计原则

二. 提高算法效率问题

1) 尽量减少运算次数：

$$x^{255}=x\cdot x^{2}\cdot x^{4}\cdot x^{8}\cdot x^{16}\cdot x^{32}\cdot x^{64}\cdot x^{128}$$

254次乘法 ⟶ 14次乘法

- 例：计算 $p_n(x)=a_nx^n+a_{n-1}x^{n-1}+\cdots+a_1x+a_0$ 的值。
  - 直接计算需要 $n(n+1)/2$ 次乘法和 n 次加法。
  - 若将公式变成递推公式：

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

  后，再计算 $p_n(x)$，只需做 n 次乘法和 n 次加法。
  - 充分利用递推公式，可提高算法效率。

4-3 数值方法的稳定性与算法设计原则

二. 提高算法效率问题

2) 充分利用耗时少的运算

运算时间：加法/减法 < 乘法 < 除法/平方等

例如：$k+k$ 比 $2k$，$a*a$ 比 a^2，$b*0.25$ 比 $b/4$ 等节省运算时间。

3) 充分利用存贮空间

- 节省原始数据的存贮单元；
- 节省工作单元。

三. 着眼于高质量的软件

- 可移植性，与具体计算机有关的问题
- 易用性，易读性，易维护性

本章重点

- 了解本门课的基本内容和基本概念（数值问题、数值方法、数值算法等）
- 掌握误差分析的基本概念：
  - 舍入误差和截断误差
  - 绝对误差和相对误差
  - 有效数字
- 掌握数值方法的稳定性与算法设计原则