计算方法

Ch01 绪论

程勇

buctcourse@163.com

http://www.buct.edu.cn

Dept. of Computer
Beijing University of Chemical Technology

Mar. 4, 2020

提纲

本章提纲

课程介绍

数值计算问题

算法

课程基本情况

1 课程基本情况

课程名称：计算方法(MAT36200C)
总 学 时：32 学时(理论课时 24 学时，上机课时 8 课时)
学　　分：2 学分
课程性质：专业选修
面向专业：自控 A18 级
考核方式：考查
上课时间：**3-14 周(周三 18:50-20:25)，第 14 周闭卷考试**
上课地点：**A 阶 203**
上机时间地点：待定
课件邮箱：buctcourse@163.com
考核方式：**期末考试 60%，作业(论文作业 18%，练习作业 10%)28%，实验(4 次实验)12%**
教　　材：计算方法讲义
上机语言：首选 C，其次 C++
期末考试：**2020.5.20(周三 18:50-20:25)**

课程电子资源

1. 课程主页。
   https://course-proxy2.buct.edu.cn/meol/jpk/course/layout/newpage/index.jsp?courseId=14468
2. 课程 PPT。主页课程 → 课程资源 → 课程幻灯片
3. 课程视频。主页课程 → 课程资源 → 学习视频
4. 课程讲义。主页课程 → 课程资源 → 上课讲义
5. 实验指导书。主页课程 → 课程资源 → 上机实验
6. 课程答疑更新。主页课程 → 课程资源 →New
7. 提交问题 (电子邮件方式: buctcourse@163.com)
8. 提交问题 (电子问卷方式)

课程教学目标

本课程的研究对象

计算方法是一门应用数值计算方法 (近似计算) 来求解数学问题的算法体系。

课程教学目标

- 帮助学生理解利用计算工具 (计算机) 求复杂数学模型数值解的重要意义、所面临的问题、以及常用的基本方法;
- 帮助学生掌握高等工程数学中一些典型数学问题求数值解的算法原理以及相应的 C 语言程序设计方法;
- 结合典型数学问题求解的计算机方法的教学，培养学生自己动手编程灵活利用计算机求解复杂数学模型的能力，并为后继课程的学习奠定相应的基础;

课程参考书

主要教材

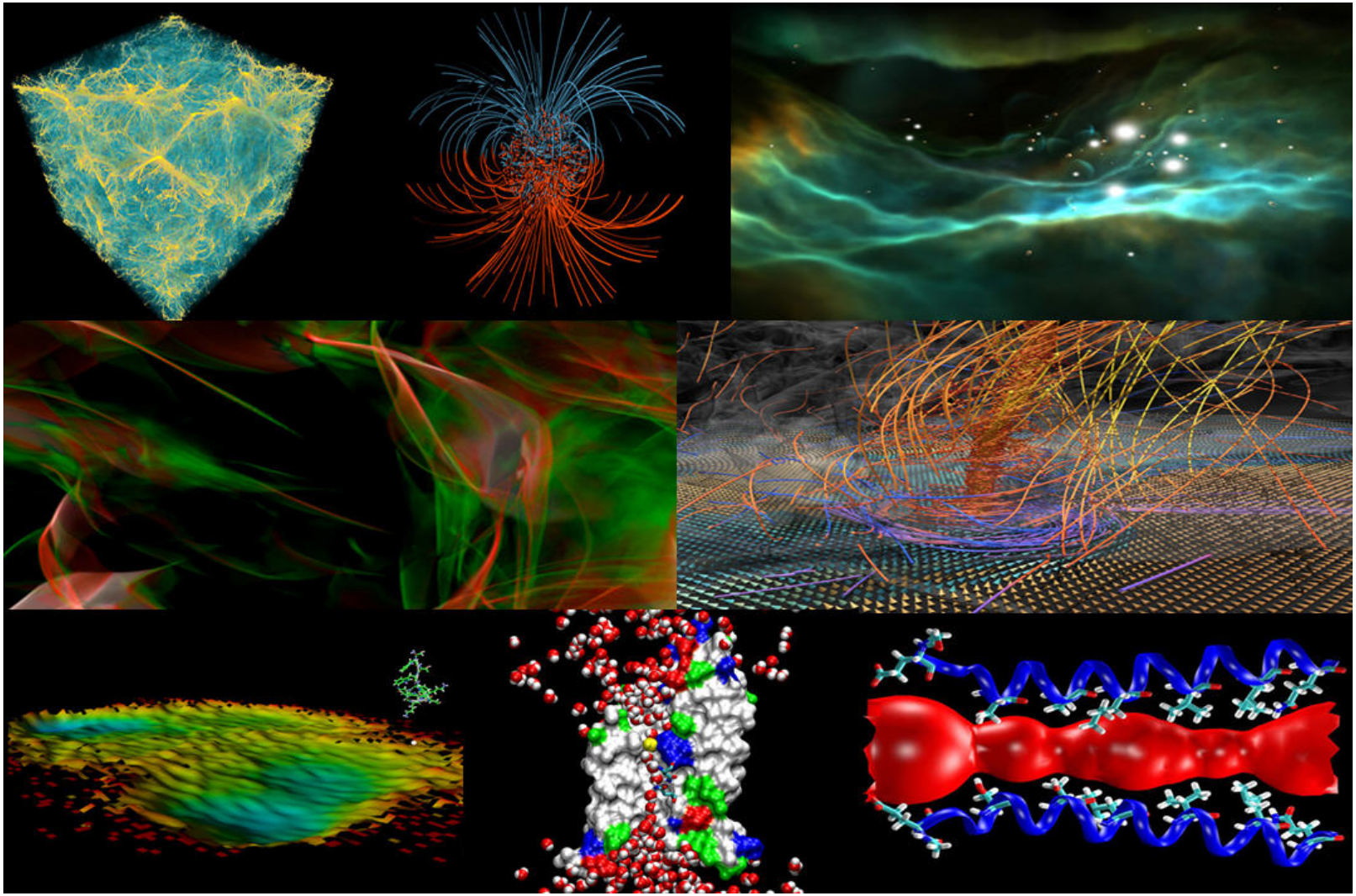
计算方法讲义 (电子版)，程勇，2020 年.

计算方法 (C 语言版)，靳天飞、杜忠友等，清华大学出版社，2010 年，ISBN 9787302221753

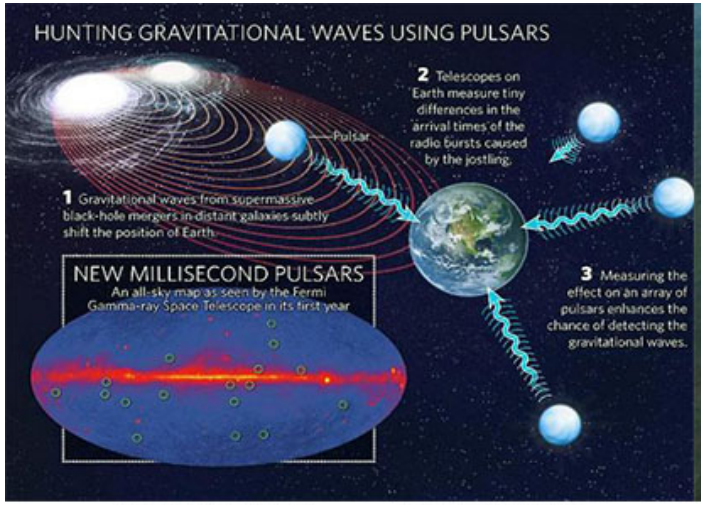
参考书

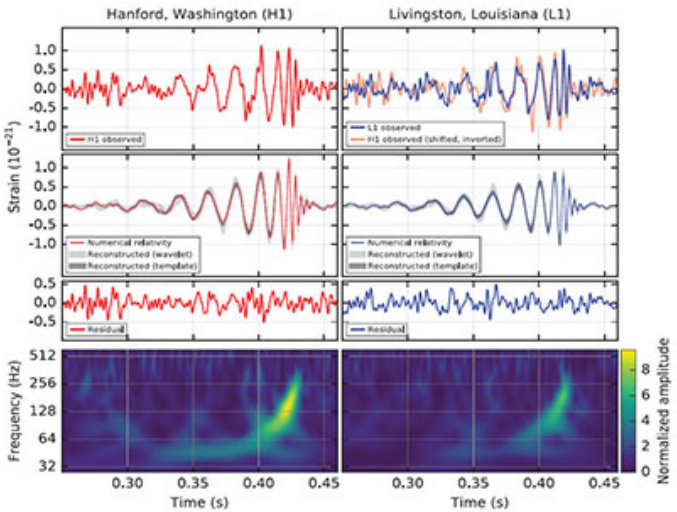
1. 计算方法，吕同富、康兆敏、方秀男编著，清华大学出版社，2013 年，ISBN 9787302326991
2. 数值分析 (第 5 版)，李庆扬等，清华大学出版社，2008 年

科学计算需求



引力波





PRL 116, 061102 (2016)
Selected for a Viewpoint in *Physics*
PHYSICAL REVIEW LETTERS
week ending 12 FEBRUARY 2016

Observation of Gravitational Waves from a Binary Black Hole Merger

B. P. Abbott *et al.*
(LIGO Scientific Collaboration and Virgo Collaboration)
(Received 21 January 2016; published 11 February 2016)

On September 14, 2015 at 09:50:45 UTC the two detectors of the Laser Interferometer Gravitational-Wave Observatory simultaneously observed a transient gravitational-wave signal. The signal sweeps upwards in frequency from 35 to 250 Hz with a peak gravitational-wave strain of 1.0×10^{-21}. It matches the waveform predicted by general relativity for the inspiral and merger of a pair of black holes and the ringdown of the resulting single black hole. The signal was observed with a matched-filter signal-to-noise ratio of 24 and a false alarm rate estimated to be less than 1 event per 203 000 years, equivalent to a significance greater than 5.1σ. The source lies at a luminosity distance of 410^{+160}_{-180} Mpc corresponding to a redshift $z = 0.09^{+0.03}_{-0.04}$. In the source frame, the initial black hole masses are $36^{+5}_{-4} M_\odot$ and $29^{+4}_{-4} M_\odot$, and the final black hole mass is $62^{+4}_{-4} M_\odot$, with $3.0^{+0.5}_{-0.5} M_\odot c^2$ radiated in gravitational waves. All uncertainties define 90% credible intervals. These observations demonstrate the existence of binary stellar-mass black hole systems. This is the first direct detection of gravitational waves and the first observation of a binary black hole merger.

DOI: 10.1103/PhysRevLett.116.061102

引力波发现中的科学计算

www.einsteinathome.org/apps.php 百度 <Ctrl+K>

计算程序

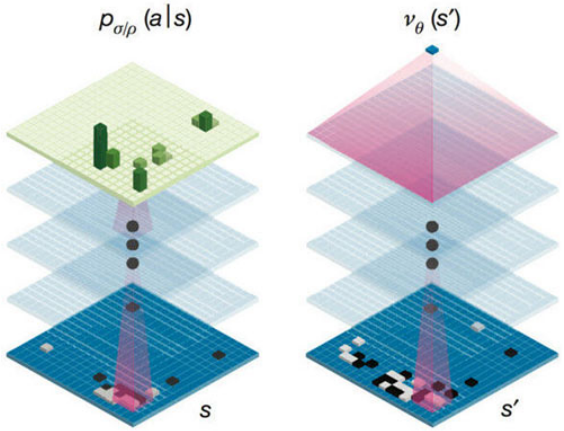
登录

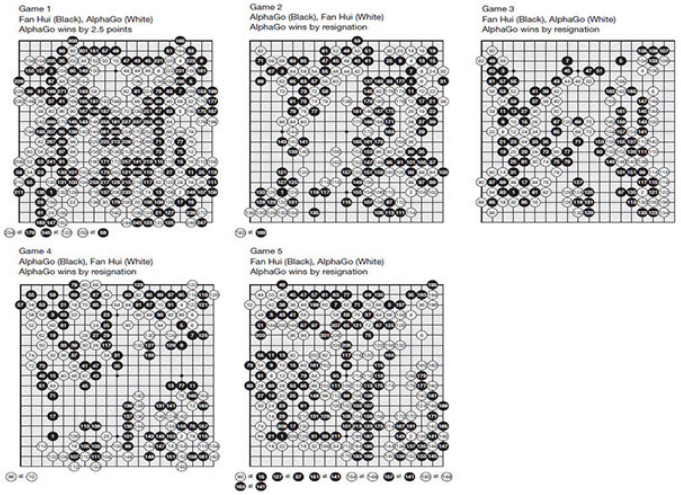
Einstein@Home currently has the following applications. When you participate in Einstein@Home, tasks for one or more of these applications will be assigned to your computer. The current version of the application will be downloaded to your computer. This happens automatically; you don't have to do anything.

Gravitational Wave search O1 all-sky tuning (beta test)		
平台	版本	创建时间
Linux/x86	1.02 (SSE2)	13 Feb 2016, 19:08:18 UTC
Windows/x86	1.02 (SSE2)	13 Feb 2016, 19:08:18 UTC
Linux running on an AMD x86_64 or Intel EM64T CPU	1.02	13 Feb 2016, 19:08:18 UTC
Linux running on an AMD x86_64 or Intel EM64T CPU	1.02 (AVX)	13 Feb 2016, 19:08:18 UTC
Microsoft Windows running on an AMD x86_64 or Intel EM64T CPU	1.02 (AVX)	14 Feb 2016, 12:07:35 UTC
Mac OS 10.5 or later running on Intel 64 Bit	1.02	13 Feb 2016, 19:08:18 UTC
Gamma-ray pulsar binary search #1		
平台	版本	创建时间
Linux/x86	1.00	19 Jan 2016, 11:17:53 UTC
Windows/x86	1.00	19 Jan 2016, 11:17:53 UTC
Mac OS X on Intel	1.00	19 Jan 2016, 11:17:53 UTC
Linux running on an AMD x86_64 or Intel EM64T CPU	1.00	19 Jan 2016, 11:17:53 UTC
Mac OS 10.5 or later running on Intel 64 Bit	1.00	19 Jan 2016, 11:17:53 UTC
Binary Radio Pulsar Search (Parkes PMPS XT)		
平台	版本	创建时间
Linux/x86	1.52 (BRP6-cuda32-nv270)	25 Mar 2015, 10:18:00 UTC
Linux/x86	1.52 (BRP6-opencl-ati)	25 Mar 2015, 10:18:00 UTC
Windows/x86	1.52 (BRP6-Beta-opencl-intel_gpu)	9 Mar 2015, 14:03:59 UTC
Windows/x86	1.52 (BRP6-cuda32)	25 Mar 2015, 10:18:00 UTC
Windows/x86	1.52 (BRP6-cuda32-nv301)	25 Mar 2015, 10:18:00 UTC
Windows/x86	1.52 (BRP6-opencl-ati)	25 Mar 2015, 10:18:00 UTC

AlphaGo by Google

Policy network Value network





ARTICLE

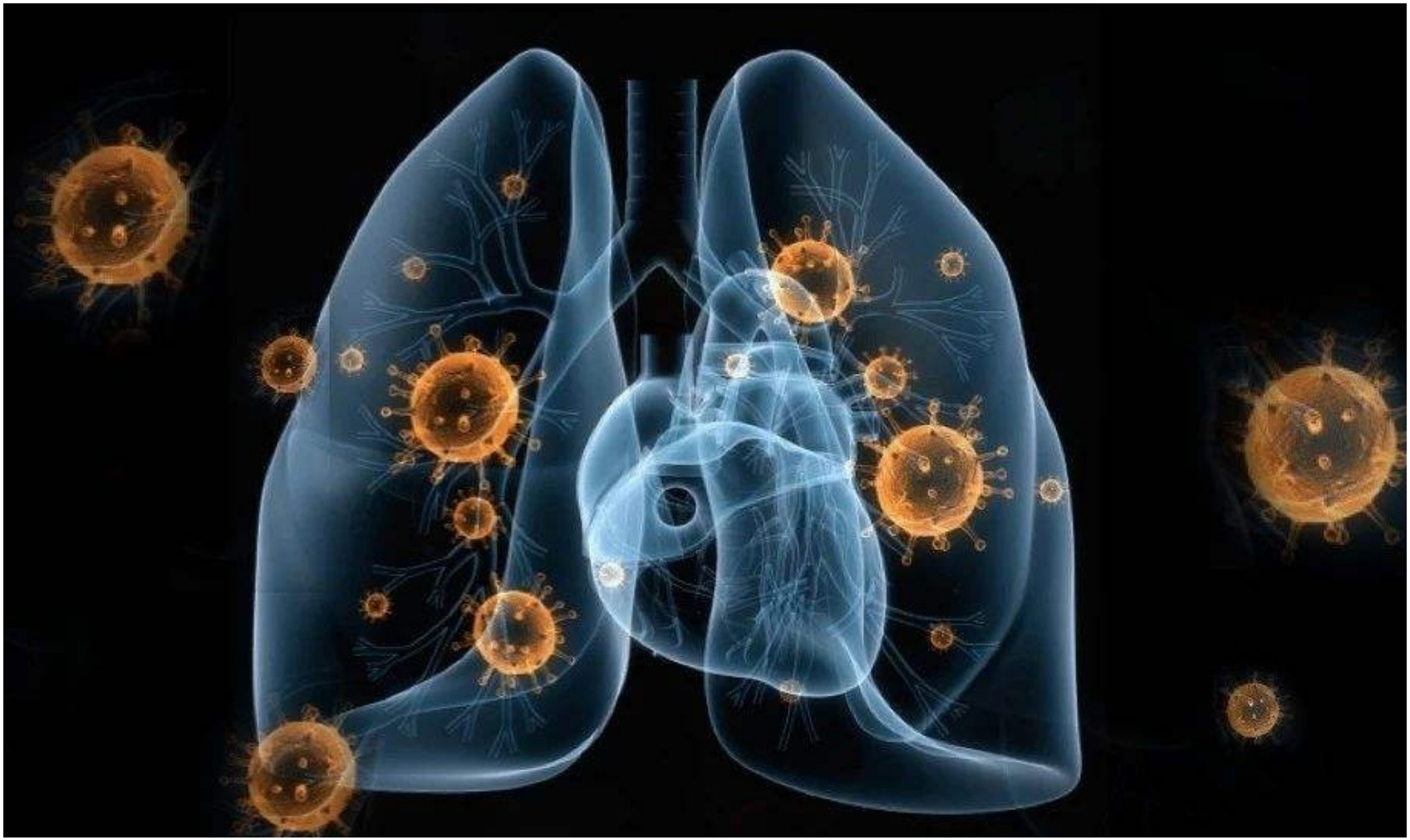
doi:10.1038/nature16961

Mastering the game of Go with deep neural networks and tree search

David Silver[1]*, Aja Huang[1]*, Chris J. Maddison[1], Arthur Guez[1], Laurent Sifre[1], George van den Driessche[1], Julian Schrittwieser[1], Ioannis Antonoglou[1], Veda Panneershelvam[1], Marc Lanctot[1], Sander Dieleman[1], Dominik Grewe[1], John Nham[2], Nal Kalchbrenner[1], Ilya Sutskever[2], Timothy Lillicrap[1], Madeleine Leach[1], Koray Kavukcuoglu[1], Thore Graepel[1] & Demis Hassabis[1]

The game of Go has long been viewed as the most challenging of classic games for artificial intelligence owing to its enormous search space and the difficulty of evaluating board positions and moves. Here we introduce a new approach to computer Go that uses 'value networks' to evaluate board positions and 'policy networks' to select moves. These deep neural networks are trained by a novel combination of supervised learning from human expert games, and reinforcement learning from games of self-play. Without any lookahead search, the neural networks play Go at the level of state-of-the-art Monte Carlo tree search programs that simulate thousands of random games of self-play. We also introduce a new search algorithm that combines Monte Carlo simulation with value and policy networks. Using this search algorithm, our program AlphaGo achieved a 99.8% winning rate against other Go programs, and defeated the human European Go champion by 5 games to 0. This is the first time that a computer program has defeated a human professional player in the full-sized game of Go, a feat previously thought to be at least a decade away.

新型冠状病毒肺炎



新型冠状病毒肺炎预测

钟南山：新型冠状病毒肺炎疫情有望4月前结束

2020年02月12日 00:11 一财网 作者：童兰

原标题：钟南山：新型冠状病毒肺炎疫情有望4月前结束

钟南山院士2月11日在接受路透社采访时表示，新型冠状病毒肺炎新增感染病例已经在一些地区出现下滑，疫情有望出现缓解。他预计峰值将会在2月中下旬出现，4月前可能结束。

钟南山表示，做出上述预测是基于现有的数学模型，近期的疫情情况，以及政府所采取的措施。

不过他承认，目前人们对新型冠状病毒还有很多未知。他说道：“我们还不知道病毒为何有如此大的传染性，这是最大的问题。”

其他的不确定性还包括新型冠状病毒能否通过粪便传播，是否有“超级传播者”等等。

钟南山同时指出，武汉地方政府和卫生部门没有在疫情发生早期采取足够的措施，应当为此承担责任。“他们没有把工作做好。”钟南山表示。

钟南山表示在此次疫情中，中国政府采取的措施比SARS期间更加有力，比如在透明性和与世卫组织的合作方面做得更好。他还呼吁政府应该采取更多的措施，比如终结野生动物市场交易，在卫生技术方面进行更多的国际合作，提升疾控中心的运营能力，以及建立潜在流行病的全球预警系统。

数学求解一般步骤

笛卡尔曾提出过被后世尊称为“万能法则”的一般模式：

1. 将实际问题化归为数学问题；
2. 将数学问题化归为代数问题；
3. 将代数问题化归为解方程；

现在科学计算求解步骤：

1. 提出实际问题；
2. 建立数学模型；
3. 提出数值问题；
4. 设计可靠、高效的算法；
5. 程序设计、上机实践计算结果；
6. 计算结果的可视化；

主要数值求解问题

本课程主要讲解五类数值求解问题，分别是：

1. 插值问题；
2. 数值求积问题；
3. 非线性方程求根问题；
4. 线性方程组求根问题；
5. 常微分方程求解问题；

多项式计算的例子

例题：计算多项式的值。

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

解：在计算过程中，需要执行乘法次数为 $n + (n-1) + (n-2) + \cdots + 1 = \frac{(1+n)\cdot n}{2}$ 次，执行加法为 n 次。

如果对上述多项式进行如下变换：

$$\begin{aligned} p(x) &= a_n x^n + a_{n-1} x^{n-1} + a_{n-2} x^{n-2} + \cdots + a_1 x + a_0 \\ &= (a_n x + a_{n-1}) x^{n-1} + a_{n-2} x^{n-2} + \cdots + a_1 x + a_0 \\ &= ((a_n x + a_{n-1}) x + a_{n-2}) x^{n-2} + \cdots + a_1 x + a_0 \\ &= (\cdots (a_n x + a_{n-1}) x + a_{n-2}) x + \cdots + a_1) x + a_0 \end{aligned}$$

此时进行多项式计算，需要执行乘法次数为 n 次，执行加法为 n 次，由此可见大大地减少了乘法计算次数，减少了计算时间。

行列式计算的例子

例题：计算如下行列式的值。

$$\begin{vmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{vmatrix}$$

解：$n=1$ 时，需要 0 次乘法，2 阶需要 2 次。

假设 n 阶需要 $f(n)$ 次乘法运算。则 $f(n)=n*f(n-1)$。

很容易就可以看出，n 阶需要 $n!$ 次乘法运算 ($n\geq 2$ 时)。

当 $n=24$ 时，$2^{24}=16777216$，而 $24!\approx 6.20449*10^{23}$，现在的世界最强的天河计算机每秒执行 5.49 亿亿次乘法，执行 $24!$ 次乘法大约需要 1 年。

当 $n=64$ 时，需要计算约 $5.0895*10^{54}$ 个宇宙年龄。

人脑和计算机计算模式比较

比较	大脑 	计算机 
创造性	有	无
问题规模	小规模	大中规模
求解精度	解析解	近似解
解决方法	公式定理定律	迭代法，校正法
遗忘性	容易	不容易
适用领域	科学理论分析	工程应用实践

算法

科学计算离不开算法设计。所谓算法 (Algorithm) 是指解题方案的准确而完整的描述，是一系列解决问题的清晰指令，算法代表着用系统的方法描述解决问题的策略机制。也就是说，能够对一定规范的输入，在有限时间内获得所要求的输出。

算法的主要性质：

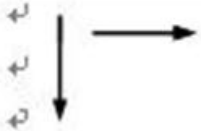
- 有穷性。算法的有穷性是指算法必须能在执行有限个步骤之后终止；
- 确切性。算法的每一步骤必须有确切的定义；
- 输入项。一个算法有 0 个或多个输入，以刻画运算对象的初始情况，所谓 0 个输入是指算法本身定出了初始条件；
- 输出项。一个算法有一个或多个输出，以反映对输入数据加工后的结果。没有输出的算法是毫无意义的；
- 可行性。算法中执行的任何计算步骤都是可以被分解为基本的可执行的操作步；

算法的描述方法

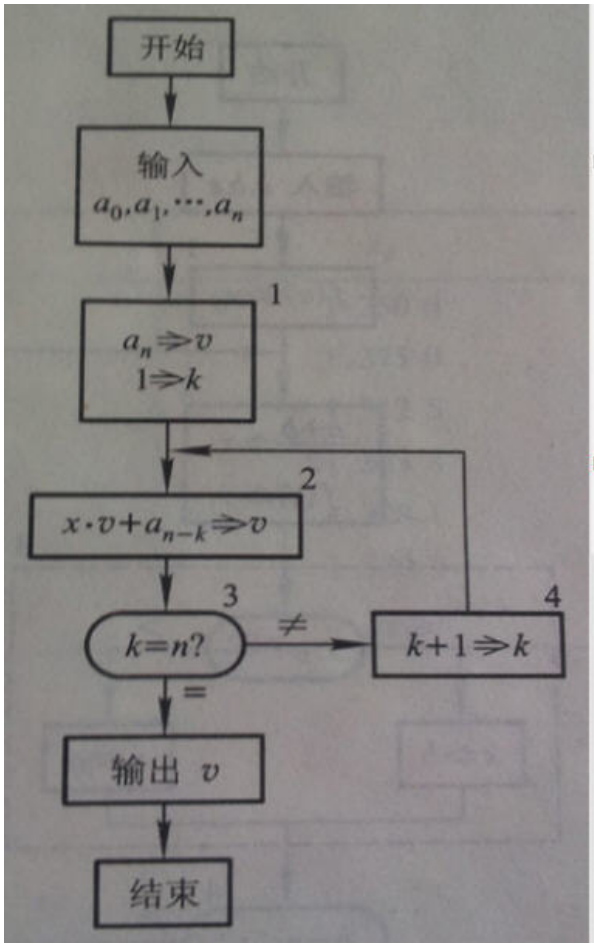
算法的描述方法包括：

- 自然语言。自然语言就是用人们日常使用的语言描述解决问题的方法和步骤，这种描述方法通俗易懂，在语法和语义上往往具有多义性；
- 伪代码。伪代码是介于自然语言和计算机语言之间的文字和符号；
- 传统流程图。传统流程图，使用不同的几何图形来表示不同性质的操作，使用流程线来表示算法的执行方向，比起前两种描述方式，其具有直观形象、逻辑清楚、易于理解等特点，但它占用篇幅较大，流程随意转向，较大的流程图不易读懂；
- N-S 结构化流程图。N-S 结构化流程图是 1973 年美国学者 Nassi 和 Shneiderman 提出的一种符合结构化程序设计原则的描述算法的图形方法。

传统流程图

流程图符号	名称	说明
	起止框	表示算法的开始和结束
	处理框	表示完成某种操作，如初始化或运算赋值等
	判断框	表示根据一个条件成立与否，决定执行两种不同操作的其中一个
	输入输出框	表示数据的输入输出操作
	流程线	用箭头表示程序执行的流向
	连接点	用于流程分支的连接

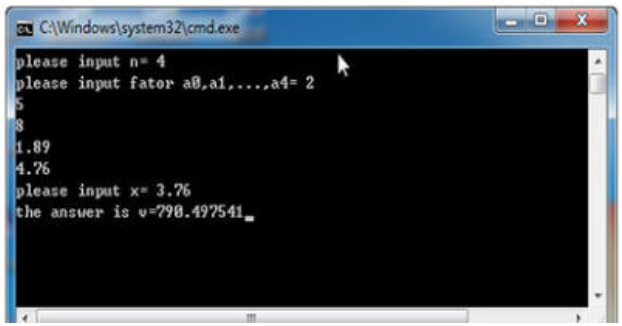
秦九韶算法的 C 语言实现




```c
/*秦九韶算法*/

#include "stdio.h"
#define N 100
int main() {
   int n,k,i;
   float a[N],x;
   double v;

   /*输入数据*/
   printf("please input n= ");
   scanf("%d",&n);
   printf("please input fator a0,a1,...,a%d= ",n);
   for(i=0; i<=n; i++) {
      scanf("%f",&a[i]);
   }
   printf("please input x= ");
   scanf("%f",&x);
   v=a[0];
   /*判断 k是否等于 n*/
   for(k=1; k<=n; k++)
      v=x*v+a[k];  // 这一步非常关键
   /*输出数据*/
   printf("the answer is v=%lf",v);
   getch();
   return 0;
}
```



数值算法设计思想

数值算法主要设计思想：

- 化大为小的缩减技术；
- 化难为易的校正技术；
- 化粗为精的松弛技术；

化大为小的缩减技术

缩减技术是数值算法设计的一种基本思想。

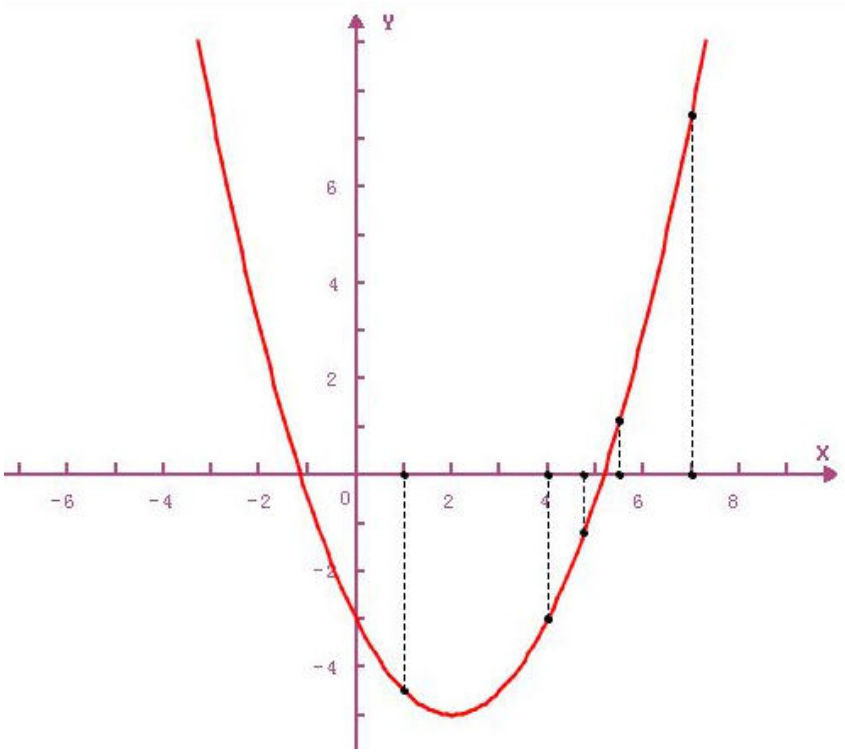
许多数值计算问题可以引进某个实数—即所谓问题的规模来刻画其“大小”，而问题的解恰好是其规模为足够小的退化情形。求解这类问题的一种行之有效的方法是通过某种简单的运算手续逐步缩减问题的规模，直到加工出问题的解。

所谓大事化小即指逐步压缩问题的规模，这个处理过程应具有结构递归和规模递减两项基本特征。小事化了即是指当问题的规模变得足够小时即可直接或方便得出问题的解。通常对于规模大但有限的问题如上述数列求和问题设计出来的一类算法统称直接法。

算法设计的这种技术称为规模缩减技术，简称缩减技术。下面将举例说明这种设计思想的具体运用。

二分法例子

对于区间 $[a,b]$ 上连续不断且 $f(a)\cdot f(b)<0$ 的函数 $y=f(x)$，通过不断地把函数 $f(x)$ 的零点所在的区间一分为二，使区间的两个端点逐步逼近零点，进而得到零点近似值的方法叫二分法。



化难为易的校正技术

校正技术的设计思想

校正技术得到的迭代法突出地体现了删繁就简，逐步求精的设计思想。“删繁就简”的含义是，删去所给复杂方程中某些高阶小量而简化生成所谓的校正方程以确定所给预报值的校正量，其中校正方程应满足逼近性和简单性两项要求。利用校正方程获得所给复杂方程解的一种行之有效的途径就是递推化，反复预报校正直到达到精度要求为止。

缩减技术主要是采用大事化小，小事化了的方法解决问题的，有些问题的“大事化小”过程似乎无法了结。这是一类无限逼近的过程，适于用所谓预报校正技术来处理。

开方法例子

给定 $a > 0$，求开方值 $\sqrt{a}$ 的问题就是要解方程：

$$x^2 - a = 0$$

设给定某个预报值 x_0，希望借助于某种简单方法确定校正量 Δx，使校正值 $x_1 = x_0 + \Delta x$ 能够比较准确地满足所给方程，即使

$$(x_0 + \Delta x)^2 \approx a$$

成立，设校正值 Δx 是个小量，舍去上式中的高阶小量 $(\Delta x)^2$，令

$$x_0^2 + 2x_0\Delta x = a$$

从中解出 Δx，可得校正值

$$\Delta x = \frac{1}{2}(\frac{a}{x_0} - x_0)$$

开方法例子 (续)

反复实施这种预报校正方法，即可求出开方公式：

$$x_{k+1} = \frac{1}{2}(x_k + \frac{a}{x_k}) \quad k = 1, 2, 3, \cdots$$

直到 $x_{k+1} - x_k < \varepsilon(\varepsilon$ 为给定阈值)，此时 x_{k+1} 即为所求的根。
当 $a = 2, x_0 = 1, \varepsilon = 10^{-6}$ 时，计算结果如下：

k	x_k
0	1
1	1.500000000
2	1.416666667
3	1.414215686
4	1.414213562
5	1.414213562

求得 $\sqrt{2} \approx= 1.414214$。

化粗为精的松弛技术

在实际计算中常常可以获得目标值 F^* 的两个相伴的近似值 F_0 与 F_1，将它们加工成更高精度的结果的方法之一就是取两者的某种加权平均作为改进值：

$$\begin{aligned} F' &= (1-\omega)F_0 + \omega F_1 \\ &= F_0 + \omega(F_1 - F_0) \end{aligned}$$

即通过适当选取权系数 ω 来调整校正量 $\omega(F_1 - F_0)$，以加工得到更高精度的 F'，这种基于校正量的调整与松动的方法通常称为松弛技术。

超松弛

有一种情况：所提供的一对近似值与 F_1 和 F_0 有优劣之分，譬如 F_1 优而 F_0 劣，这时所采用如下松弛方式为：

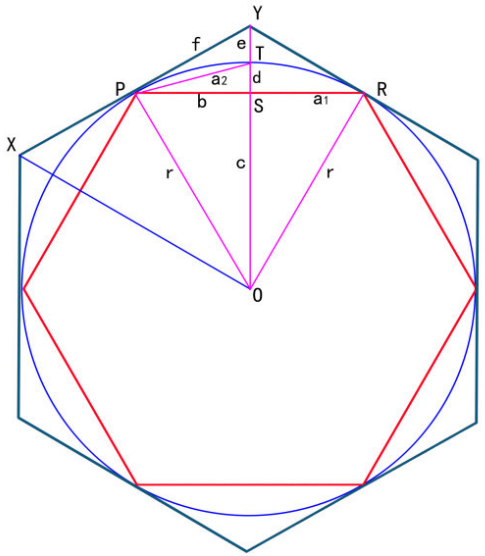
$$F' = (1+\omega)F_1 - \omega F_0 \qquad \omega > 0$$

即在松弛过程中张扬 F_1 的优势而抑制 F_0 的劣势，这种设计策略称作外推松弛技术，简称超松弛。

总之，超松弛的设计机理是优劣互补，化粗为精。松弛技术的关键在于松弛因子的选取，而这往往是相当困难的。

割圆术例子

刘徽 (约公元 225 年-295 年，魏晋期间)，是中国数学史上一个非常伟大的数学家，他提出了“割圆术”，这是将圆周用内接或外切正多边形穷竭的一种求圆面积和圆周长的方法。他用割圆术，从直径为 2 尺的圆内接正六边形开始割圆，依次得正 12 边形、正 24 边形，$\cdots$，割得越细，正多边形面积和圆面积之差越小。他计算了 3072 边形面积，得到 $\pi \approx 3927/1250 = 3.1416$。



割圆术例子 (续)

他发现利用 $S_{96} = 313\frac{584}{625}$ 与 $S_{192} = 314\frac{64}{625}$ 两个粗糙的数据进行松弛，可以得到高精度的值 S_{3072}，加工过程如下：

$$
\begin{aligned}
S^* &\approx S_{192} + \frac{36}{105}(S_{192} - S_{96}) \\
&= 314\frac{64}{625} + \frac{36}{105} \times \left(314\frac{64}{625} - 313\frac{584}{625}\right) \\
&= 314\frac{64}{625} + \frac{36}{105} \times \frac{105}{625} \\
&= 314\frac{4}{25} \\
&= S_{3072}
\end{aligned}
$$

误差的来源

- 模型误差 (描述误差)。反映实际问题有关量之间的计算公式 (数学模型) 通常是近似的;
- 观测误差。数学模型中包含的某些参数是通过观测得到的;
- 截断误差 (方法误差)。数值方法精确解与待求解模型的理论分析解之间的差异;
- 舍入误差。由于计算机表示浮点数采用的是有限字长;这样在计算机中表示的原始输入数据、中间计算数据、以及最终输出结果必然产生误差,称此类误差为舍入误差;
- 初值误差。由于初始值选取不合理所造成的最终结果的差异;

在计算方法中不研究前两类误差,总是假定数学模型是正确合理的反映了客观实际问题。

误差的度量

绝对误差定义

设 x 是某个量的准确值，x^* 是其近似值，则两者的差值称为近似值 x^* 的绝对误差，简称误差。

$$e(x^*) \triangleq x - x^*$$

在不引起混淆的情况下，可以简记 $e(x^*)$ 为 e。

绝对误差限

如果存在正数 ε，使得有绝对误差

$$|e(x^*)| = |x - x^*| \leq \varepsilon$$

则称 ε 为 x^* 近似 x 的一个绝对误差限。

$$x \in [x^* - \varepsilon, x^* + \varepsilon], \quad x = x^* \pm \varepsilon$$

误差的度量 (续)

绝对误差限虽然能够刻画对同一真值不同近似的好坏，但它不能刻画对不同真值近似程度的好坏。

相对误差定义

设 x^* 是对准确值 $x(\neq 0)$ 的一个近似，则称：

$$e_r(x^*) \triangleq \frac{e(x^*)}{x} = \frac{x - x^*}{x} \approx \frac{e(x^*)}{x^*}$$

为 x^* 近似 x 的相对误差，不引起混淆时简记 $e_r(x^*)$ 为 e_r。

相对误差限

数值 $|e_r|$ 的上限，记为 ε_r，相对误差限也可通过 $\varepsilon_r = \frac{\varepsilon}{|x^*|}$ 来计算。

误差的度量 (续)

为了规定一种近似数的表示法，使得用它表示的近似数自身就指示出其误差的大小。这里介绍有效数字和有效数的概念。

有效数字

设 x 的近似值 x^* 有如下标准形式：

$$x^* = \pm 10^m \times \underline{0.x_1x_2x_3\cdots x_nx_{n+1}\cdots x_p}$$

其中 m 为整数，$\{x_i\} \subset \{0,1,2,3,\cdots 9\}$ 且 $x_1 \neq 0, p \geq n$。如果有

$$|e| = |x - x^*| \leq \frac{1}{2} \times 10^{m-n}$$

则称 x^* 为 x 的具有 n 位有效数字的近似数，或称 x^* 准确到 10^{m-n} 位，其中数字 $x_1, x_2, x_3, \cdots, x_n$ 分别被称为 x^* 的第 $1, 2, 3, \cdots, n$ 个有效数字。

有效数字 (续)

有效数

当 x^* 准确到末位，即 $n = p$，则称 x^* 为有效数。

例题：

$x = \pi, x_1^* = 3.141, x_2^* = 3.142, x_3^* = 3.1416, x_4^* = \frac{22}{7}, x_5^* = \frac{355}{113}$，分别计算 $x_1^*, x_2^*, x_3^*, x_4^*, x_5^*$ 分别具有几位有效数字，是否为有效数。

解：

这里取 $\pi = 3.14159265358979323846\cdots$。因此有：

$$|x - x_1^*| = 0.00059\cdots \leq 0.005 = \frac{1}{2} \cdot 10^{1-3}$$

$$|x - x_2^*| = 0.00040\cdots \leq 0.0005 = \frac{1}{2} \cdot 10^{1-4}$$

$$|x - x_3^*| = 0.0000073\cdots \leq 0.00005 = \frac{1}{2} \cdot 10^{1-5}$$

有效数字 (续)

$$|x - x_4^*| = 0.0012\cdots \leq 0.005 = \frac{1}{2}\cdot 10^{1-3}$$

$$|x - x_5^*| = 0.00000026\cdots \leq 0.0000005 = \frac{1}{2}\cdot 10^{1-7}$$

因此，x_1^* 具有 3 位有效数字，为非有效数；x_2^* 具有 4 位有效数字，是有效数；x_3^* 具有 5 位有效数字，是有效数；x_4^* 具有 3 位有效数字，不是有效数；x_5^* 具有 7 位有效数字，也不是有效数。

- 有效数的误差限是末位数单位的一半，可见有效数本身就体现了误差界；
- 对真值进行四舍五入得到有效数；
- 准确数字有无穷多位有效数字；
- 从实验仪器所读的近似数（最后一位是估计位）不是有效数，估计最后一位是为了确保对最后一位进行四舍五入得到有效数；

误差度量之间的联系

1. 绝对误差与相对误差

$$\frac{e(x^*)}{x} = e_r(x^*)$$

2. 绝对误差与有效数字

$$|e(x^*)| \leq \frac{1}{2} \cdot 10^{m-n}$$

3. 相对误差与有效数字

定理

1. 若 x^* 具有 n 位有效数字，则相对误差为 $|e_r| \leq \frac{1}{2x_1} \times 10^{1-n}$；
2. 若相对误差为 $|e_r| \leq \frac{1}{2(x_1+1)} \times 10^{1-n}$，则 x^* 至少具有 n 位有效数字。

该定理实质上给出了一种求相对误差限的方法。仅从 $|e_r| \leq \frac{1}{2x_1} \times 10^{1-n}$ 并不能保证 x^* 一定具有 n 位有效数字。

选用算法应该遵循的原则

- 简化计算步骤以减少运算次数；
- 避免大数“吃掉”小数；
- 尽量避免相近的数相减；
- 避免绝对值很小的数作为分母，防止出现溢出；
- 选用数值稳定性好的算法；

定义

一个算法，如果在运算过程中舍入误差在一定条件下能够得到控制，或者舍入误差的增长不影响产生可靠的结果，则称该算法是数值稳定的，否则称其为数值不稳定。

算法稳定性的例子

例题：计算如下积分近似值。

$$I_n = \int_0^1 \frac{x^n}{x+5} dx$$

方案一：

$$I_0 = \int_0^1 \frac{1}{x+5} dx = \ln \frac{6}{5} \approx 0.1823$$

$$I_n = \frac{1}{n} - 5I_{n-1} \quad (n = 1, 2, 3, \cdots)$$

方案二：

$$\frac{1}{6(n+1)} = \int_0^1 \frac{x^n}{6} dx \leq I_n \leq \int_0^1 \frac{x^n}{5} dx = \frac{1}{5(n+1)}$$

$$I_{n-1} = \frac{1}{5n} - \frac{1}{5} I_n \quad (n = n-1, n-2, n-3, \cdots, 1)$$

本章小结

- 课程介绍；
- 数值计算问题；
- 基本概念；
- 数值算法设计思想；

练习题

1. 复习 C 语言程序设计语言。
2. 编程实现求多项式值的秦九韶算法。

谢谢!

AUTHOR: Cheng Yong
ADDRESS: Dept. of Computer
Beijing University of Chemical Technology
Beijing, 100029, China
EMAIL: buctcourse@163.com