叶其孝

综合报告

探索性实验和计算

David H. Bailey Jonathan M. Borwein

作者的理念 —— 曾经备受争议，现已成为常识 —— 是：计算机能够成为数学研究的有用的、甚至是关键的助手. ——Jeff Shallit

Jeff Shallit 在他最近对 [10] 的评论 (MR 2427663) 中写下了这句话. 为了搞清楚这一点, Shallit 在如下的意义上是完全正确的：如果不是大多数，至少也有许多数学家在他们的研究中以各种方式使用计算机，包括画图、分析数值数据、处理符号表达式，还有模拟计算. 但是在我们看来，无论是在研究论文中，教科书中和课堂讲授中，或是在如何组织数学的发现过程方面，数学知识的表达还没有取得实质性的和足够理性意义上的进步.

1. 数学家是人

我们同意 George Pólya (1887—1985) 的如下观点 [25, 卷 2, p.128]: 在学习时，

"我们的直观感觉比形式论证来得要快得多，并且受外界的影响也少得多."

不过 Pólya 接着重申，在学校里应该教授证明技巧.

我们再说说观察的作用. 关于这一点，在一些合著的书中，如《用实验做数学 (Mathematics by Experiment)》 [10], 以及《实验数学在行动 (Experimental Mathematics in Action)》 [3], 我们都指出了数学知识的本性正在发生变化. 因此，就要问诸如 "我们应该如何向学生教授，教 '什么' 和 '为什么' 教？", 以及 "为什么我们要证明一些东西？" 之类的问题. 对后一个问题的回答是 "根据情况而定", 因为有些时候我们需要直觉和领悟，而有些时候，特别是对于一些辅助性的结果，我们更愿意知道它们是如何证明的. 对这两个任务，计算机都有能力去帮助实现.

Smail [27, p.113] 写道："人类的大脑经历了 170 万年的进化，使人能够驾驭在社会生活中出现的各种日益增长的复杂性." 其结果是：人类找到了更加得心应手的论证模式，但也是更易犯某些错误的论证模式. 类似地，著名的进化心理学家 Steve Pinker 注意到：语言 [24, p.83] 是建立在 "空间、时间、因果、占有和目标等超凡概念之上的. 正是这些概念构成了思维的语言."

这些论断还没有离开数学的范围. 计算机则不一样, 它提供了一种工作平台, 不仅能推进数学推理 —— 例如最近与李群 E8 有关的计算 (见 http://www.aimath.org/E8/computer-details.html), 而且也能遏制数学错误.

译自： Notices of the AMS, Vol.58 (2011), No.10, p.1410–1419, Exploratory Experimentation and Computation, David H. Bailey and Jonathan M. Borwein, figure number 2.

2. 实验方法学

法官 Potter Stewart 在 1964 年有一句著名的评语："当我看到它时，我就知道它了."《计算机作为熔炉 (The Computer as Crucible)》[13] 一书即以此引语开头. 用稍微非正式一点的话说，我们所说的实验数学指的是 [10]:

(a) 获取洞察和 直观 (*intuition*);

(b) 使数学原理 可视化;

(c) 发现 新的关系;

(d) 测试, 特别是 证伪 猜想;

(e) 探索 一个可能的结果以判断是否 值得 为之构造一个形式证明;

(f) 建议 构造形式证明的途径;

(g) 用 计算 取代冗长的推导;

(h) 用分析方法 确认 推导所得的结果.

其中, (a) 到 (e) 是主要的, 而 (f) 对我们也起重要作用, 但是它意指计算机辅助的、或者计算机导向的证明, 因此很不同于 2008 年 12 月《美国数学会通讯 (Notices of the AMS)》的一期专刊所说的那种形式证明; 例如, 见 [20].

数字完整性 I. 对于我们来说, (g) 已经是无处不在了, 并且我们发现 (h) 在保证所发表的数学的完整性方面特别有效. 例如, 我们经常会通过计算一个等式左边和右边的值到很高的精度, 并且比较其结果来检查和纠正数学手稿中的等式 —— 并且在必要时用软件来修复缺陷.

作为第一个例子, 在当前研究 "字符和" 时我们想使用 [14] 中推导的如下结果:

$$\sum_{m=1}^{\infty}\sum_{n=1}^{\infty}\frac{(-1)^{m+n-1}}{(2m-1)(m+n-1)^3} \tag{1}$$
$$\stackrel{?}{=}4\mathrm{Li}_4\left(\frac{1}{2}\right)-\frac{51}{2880}\pi^4-\frac{1}{6}\pi^2\log^2(2)+\frac{1}{6}\log^4(2)+\frac{7}{2}\log(2)\zeta(3).$$

在这里, $\mathrm{Li}_4(1/2)$ 是一个多重对数值. 但是, 为了检查结果而做的后续计算却揭示出: 计算该等式左边得到 $-0.872929289\ldots$, 而计算右边则得到 $2.509330815\ldots$. 这太奇怪了. 于是我们重新计算了和式, 也计算了右边的每个项 (不包括它们的系数) 一直精确到小数点后 500 位, 然后应用于 PSLQ 算法. 该算法搜索一组常数之间的整数关系 [16]. PSLQ 很快就找到了如下的关系:

$$\sum_{m=1}^{\infty}\sum_{n=1}^{\infty}\frac{(-1)^{m+n-1}}{(2m-1)(m+n-1)^3} \tag{2}$$
$$\stackrel{?}{=}4\mathrm{Li}_4\left(\frac{1}{2}\right)-\frac{151}{2880}\pi^4-\frac{1}{6}\pi^2\log^2(2)+\frac{1}{6}\log^4(2)+\frac{7}{2}\log(2)\zeta(3).$$

换句话说, 在把 (1) 式抄进最初的手稿时 "151" 变成了 "51". 如果我们没有用计算的方法来查验和纠错的话, 很有可能这个错误会不被发现和纠正. 当然并非每次都必须这样做, 但是它有可能是关键的.

在 Berkley 现在的一位研究助手 Alex Kaiser 的帮助下, 我们开始设计软件来使这个过程精细化和自动化, 并且在每篇有很多等式的稿件投出之前运行此软件来检验它. 当

由符号处理生成的可验证输出达到了一部 Salinger 小说那样的长度时，半自动化的完整性检查就成为非常迫切的需求. 例如，不久前在研究超立方体内点的期望半径 [12] 时，有必要证明下列式子的 "封闭形式" 的存在性：

$$J(t) := \int_{[0,1]^2} \frac{\log\left(t + x^2 + y^2\right)}{(1+x^2)(1+y^2)} dxdy. \tag{3}$$

[12, 定理 5.1] 的计算机验证很快就给出了一个由 10 万个字符组成的 "解答"，此解答可以用数值方法非常快速地证实，直至数百位数字. 一个高度交互式的过程可以极为漂亮地把这个表达式的一个基本实例归约为如下的精简形式：

$$J(2) = \frac{\pi^2}{8}\log 2 - \frac{7}{48}\zeta(3) + \frac{11}{24}\pi \mathrm{Cl}_2\left(\frac{\pi}{6}\right) - \frac{29}{24}\pi \mathrm{Cl}_2\left(\frac{5\pi}{6}\right). \tag{4}$$

其中 Cl_2 是 *Clausen 函数* $\mathrm{Cl}_2(\theta) := \sum_{n\geq 1} \sin(n\theta)/n^2$，其中 Cl_2 是最简单的非初等傅里叶级数. 为了让这种归约自动化，需要一个精巧的简化模式，并且还要有一个非常大的可扩充的知识库.

3. 发现真理

Giaquinto 说过一句引人注意的、概括性的话 [18, p.50]: "总而言之，发现一个真理就是以一种独立的、可靠的而且理性的方式相信这个真理." 由此完全可以推出结论：一个学生可以合法地重新发现他的老师已经 "知道" 的事物. 而且也没有必要要求每个命题都是绝对原创的 —— 只要该命题是独立发现的即可. 例如，一个微分方程命题的价值不会因为事后发现该命题的主要结果已在一个月之前在该学生不了解的情况下被一个控制理论刊物所接受而受影响，只要该命题确实是独立发现的即可. 在科学上，常见同一命题被不同的人几乎同时独立发现，而且这种现象以后会更频繁地出现，因为地球的 "新神经系统" (希拉里 · 克林顿在最近一次政治演讲中的用语) 在继续渗透进研究中.

尽管人们常常把数学等同于演绎推理，Kurt Gödel (1906—1978) 在他 1951 年的 Gibbs 演讲中说：

"如果数学像物理那样描述一个客观世界，那么就没有理由反对把归纳方法应用于数学，就像把它应用于物理一样."

他坚持这个观点直至他生命的终结，尽管他做出了世纪性的演绎推理的成就 —— 也许，正是由于他做出了这样的成就 —— 即他的不完全性的成果.

我们还要强调，许多伟大的数学家，从阿基米德和伽利略 (据称他们说过类似的著名的话："所有的真理一旦被发现，都是容易理解的. 困难在于去发现它们.")，到高斯、庞加莱和卡勒松 (Carleson)，他们都曾强调：事先 "知道" 答案是何等的重要. 两千年前，阿基米德在他的长期丢失，近期才找到并复原的《方法 (Method)》手稿的引言中写道：

"比起没有任何预先知识来，如果我们能使用这种方法，事先知道所论问题的答案，那么提供一个证明是相对容易的."

阿基米德的《方法》可以看作是现代交互式几何证明软件的先驱，但是也要指出，例如，Cinderella 软件包可以证明相当一部分欧几里得几何定理.

当 2006 年 Abel 奖得主 Lennart Carleson 在他的 1966 年国际数学家大会演讲中提

及他肯定地解决了 Luzin 在 1913 年提出的猜想 (平方可和函数的傅里叶级数点点收敛到该函数) 时，说到他曾用许多年时间试图找到该猜想的反例，最后他判定这种反例并不存在. 他是这样描述这种信心的重要性的：

"在解决一个数学问题时最重要的方面是深信什么样的结果是对的. 然后用两年或 3 年时间去运用过去 20 年左右发展起来的技术来证明它."

4. 数字辅助

数字辅助 的意思就是运用以下手段：

(a) *集成数学软件*, 如 Maple 和 Mathematica, 最好是 MATLAB 及其各种开源版本;

(b) *专用的软件包*, 如 CPLEX, PARI, SnapPea, Cinderella 和 MAGMA;

(c) *通用程序设计语言*, 如 C, C++ 和 Fortran-2000;

(d) *基于因特网的应用程序*, 如 Sloane 的 Integer Sequences 百科全书, Inverse Symbolic Calculator,[1] Fractal Explorer, Jeff Weeks 的 Topological Games, 或 Euclid in Java;[2]

(e) *因特网数据库和服务设施*, 包括 Google, MathSciNet, arXiv, Wikipedia, MathWorld, MacTutor, Amazon, Amazon Kindle, 还有许多其它的并不那么出名的设施.

所有这些都涉及不同形式的数据挖掘. 在 Kindle 内部可以随时访问 Oxford 字典和 Wikipedia, 这样的功能对传统的阅读方式是一种颠覆式的改变. Franklin [17] 坚持认为，通过 "技术的推广" 和 "广义的仪器使用" 实现的 Steinle 式的 "探索性实验", 亦即人们通常在药剂学、天文学、医学和生物技术中所做的那样，正在导致重新评估究竟在什么情况下应该做实验，因为现在不再需要有一个 "局部模型" 作为前提条件. 例如一个制药公司现在可以快速地考察并舍弃上万种候选药物，并把资源集中使用于那些幸存下来的候选者，而不需要事先确定哪些候选者最有希望取得成功. 类似地，航空工程师们可以通过计算机模拟舍弃成千种可能的设计，而只把那些最好的候选者投入正式的开发和测试.

Hendrik Sϕrenson [28] 曾概括地说, 上面所定义的实验数学就是把 Mathematica, Maple 和 MATLAB 这样的软件当作广义的仪器来做数学实验：

"探索性实验和广义仪器的思路来自于 (自然) 科学哲学，在实验数学的意义上它还没有获得充分发展. 但是我敢断言，对于在包括概念形成在内的探索性实验研究中使用广义仪器，其重要性同样适用于数学."

因而，数学和自然科学之间的边界，以及归纳和演绎推理之间的边界是模糊的，并且正在变得越来越模糊 [2]. 通常，许多数学家在申请基金时力图把他们建议的研究方法描述得让那些严厉的传统的科学家能够满意；而上面所说的这种殊途同归的现象能多少让他们从这种顾虑中解脱出来. 至于说是否存在真正意义上的数学实验 (如同在 [10] 中描述的那样), 这是一个在哲学上很能迷惑人，但在数学上是微不足道的小问题，即使对于那

1) 在 http://isc.carma.newcastle.edu.au/ 上可以访问 ISC. 它的大多数功能现已包含进 Maple 9.5 以上版本的 "identify" 函数之中. 例如, Maple 指令 identify (4.45033263602792) 返回值 $\sqrt{3}+e$, 表示此十进制值被 $\sqrt{3}+e$ 所逼近.—— 原注

2) 有代表性的基于因特网的数学资源可以在如下网址上找到：http://carma.newcastle.edu.au/portal/ 和 http://www.experimentalmath.info.—— 原注

些对数学存在持完全理想主义观点的人也是如此. 我们不想深入探讨这个问题. 我们认为这样的数学实验确实存在.

5. π, 分拆和素数

现今的数学家很难想象能够在身旁没有一台计算机的情况下研究数学. 例如, 特征多项式和极小多项式, 这些当我们做学生的时候对我们还是十分抽象的概念, 如今却成了快速增长的具体的符号处理工具箱中的一员. 当人们试图确定包括如下实例:

$$M_4=\begin{bmatrix} 2 & -21 & 63 & -105 \\ 1 & -12 & 36 & -55 \\ 1 & -8 & 20 & -25 \\ 1 & -5 & 9 & -8 \end{bmatrix}, \quad M_6=\begin{bmatrix} 2 & -33 & 165 & -495 & 990 & -1386 \\ 1 & -20 & 100 & -285 & 540 & -714 \\ 1 & -16 & 72 & -177 & 288 & -336 \\ 1 & -13 & 53 & -112 & 148 & -140 \\ 1 & -10 & 36 & -66 & 70 & -49 \\ 1 & -7 & 20 & -30 & 25 & -12 \end{bmatrix}$$

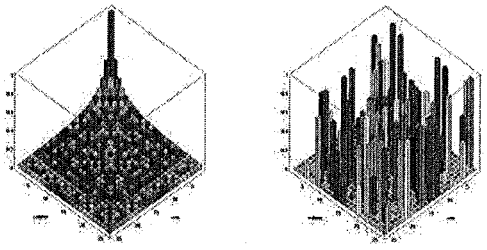

图 1 一个 25×25 希尔伯特矩阵 (左) 和一个带有 50% 稀疏和随机 [0, 1] 元素的矩阵 (右) 形成的点集

在内的一组无穷多个矩阵的结构时, 他们也许会感到无从下手. 但是只需发出一行指令, 一个计算机代数系统就会揭示这两个矩阵满足公式 $M_4^3-3M_4-2I=0$ 和 $M_6^3-3M_6-2I=0$. 同样, 越来越多的矩阵运算用图形学方法处理显得有益甚至是必需的. 现在已经众所周知, 为了在数值线性代数中发现定性信息, 例如特大矩阵的块结构, 使用图形学工具是必需的. 例如见图 1.

同样可供使用的是许多矩阵分解算法, 如 Groebner 基, Risch 判定算法 (判定一个初等函数何时具有一个初等不定积分), 图和群的一览表, 等等. 计算机代数系统在 20 年以前还像是一个玩具, 但如今它们的一些算法部件的计算效率已有极大的提高. 在极高精度计算方面情况也类似 —— 这是我们自身的许多工作的先决条件 [8, 11, 9]. 下面我们将说明, 在以往 30 年我们为把计算实验结合进研究工作而做的努力中, 我们至少经历了 12 次计算机处理能力和存储容量的摩尔定律翻番 [10, 13], 这种进步再加上高度并行机群的使用 (有数以千计的处理器核) 和光纤网络, 产生的效果是使许多运算的速度提高了 6—7 个数量级.

6. 分拆函数

考察一个自然数 n 的加法分拆的个数 $p(n)$, 其中我们忽略排序和零. 例如, $5=4+1=3+2=3+1+1=2+2+1=2+1+1+1=1+1+1+1+1$, 因此 $p(5)=7$. 由欧拉发现的常规生成函数 (5) 是:

$$\sum_{n=0}^{\infty} p(n)q^n=\prod_{k=1}^{\infty}(1-q^k)^{-1}. \tag{5}$$

(此公式可以通过使用 $1/(1-q^k)$ 的几何公式去展开每一个项, 并且观察诸 q^n 的幂的排

列来证明.)

MacMahon 在 20 世纪初做的著名计算 $p(200) = 3972999029388$, 到了 1991 年, 仅仅依靠公式 (5) 和使用符号方法, 在一个平常的笔记本计算机上, 只用 20 分钟就完成了, 如今甚至仅仅用 0.17 秒即可完成. 而更加复杂得多的计算:

$$p(2000) = 4720819175619413888601432406799959512200344166$$

于 2009 年仅用两分钟就完成了. 不仅如此, Crandall 在 2008 年 12 月甚至仅用 3 秒钟就在他的笔记本计算机上完成了 $p(10^9)$ 的计算, 其中使用了 Hardy-Ramanujan-Rademacher 的 $p(n)$ 的 "有穷" 级数, 再加上 FFT 方法. 使用这些技术, Crandall 计算了疑似素数 $p(1000046356)$ 和 $p(1000007396)$, 其中每一个都有约 35000 个十进制位.

这样的结果令人不禁要问: 是否会由于计算太容易了而挫伤人们创新的积极性: 如果 Hardy 和 Ramanujan 有强大的计算机可用, 他们还会发现他们美妙的 $p(n)$ 公式吗?

7. π 的四次方算法

同样地, π 的计算记录从 1986 年的 2937 万十进制位上升到 2010 年的五万亿十进制位. 由于所有的这类计算都要用到下面的算法, 来比较一下它们的效率是很有意思的: 令 $a_0 := 6 - 4\sqrt{2}$, $y_0 := \sqrt{2} - 1$, 然后迭代下式:

$$\begin{aligned} y_{k+1} &= \frac{1 - (1 - y_k^4)^{1/4}}{1 + (1 - y_k^4)^{1/4}} \\ a_{k+1} &= a_k(1 + y_{k+1})^4 - 2^{2k+3} y_{k+1}(1 + y_{k+1} + y_{k+1}^2). \end{aligned} \tag{6}$$

则 a_k 以四次方的速度收敛于 $1/\pi$. 每次迭代产生的十进制位数都近似地把正确的十进制位数翻番成原来的四倍. 1983 年在一个 16 K 的 Radio Shack 便携机上对 (6) 做 21 次完全精确的迭代, 得到一个代数数, 它一直到 6 万亿位以上都能很好地符合 π. 这组公式, 以及 1976 年的 Salamin-Brent 公式 [10, 第 3 章] 在过去的 1/4 世纪中被频繁采用. 下面是一个关于 π 计算精度的十分简略的年表 (根据 http://en.wikipedia.org/wiki/Chronology_of_computation_of_pi):

- 1986 : 利用公式 (6), 在 NASA Ames 研究中心的新 Cray-2 计算机上以一个 CPU 计算 π 到 2940 万位用了 28 个小时. 用另外一个算法来确认此结果用了 40 小时. 这次计算发现了 Cray-2 的硬件和软件错误. 为取得此次成功, 还开发了更快的 FFT 算法 [10, 第 3 章].
- 2009 年 1 月: 利用公式 (6), 在 Appro Xtreme-X3 系统上以 1024 个核 (和 6.348 T 字节的主存) 计算 π 到 1.649 万亿位用了 73.5 个小时. 用 Salamin-Brent 公式来检查此结果用了 64.2 小时和 6.732 T 字节的主存. 这两次计算的结果仅仅在最后 139 位上有差别.
- 2009 年 4 月: Takahashi 把他的记录提高到令人惊讶的 2.576 万亿位.
- 2009 年 12 月: Bellard 利用下面的 Chudnovsky 级数计算 π 到 2.7 万亿十进制位 (首先用二进制计算, 然后转换成十进制). 该计算一共花了 131 天, 但是他后来只使用了一个四核的工作站, 大量的磁盘, 甚至还更多地用上了人的智能!
- 2010 年 8 月: Kondo 和 Yee 用同一个 Chudnovsky 公式 (14) 计算 π 到 5 万亿十进

制位. 他首先用二进制计算, 然后转换成十进制. 二进制结果的确认是通过计算 π 的 32 个 16 进制数字, 到第 4,152,410,118,610 位结尾, 完成的, 其中采用了 Bellard 和 Plouffe 的 BBP 型 π 计算公式. 进一步的细节见: http://www.numberworld.org/misc_runs/pi-5t/announce_en.html. 也见 [6]. 这些数字看上去 "非常正常".

Daniel Shanks 曾在 1961 年计算 π 超过 10 万位. 他有一次告诉 Phil Davis 说, 计算 π 到 10 亿位是永远不可能的. 但是 Kanada 和 Chudnovsky 在 1989 年都做到了这一点. 类似地, 直觉主义者 Brouwer 和 Heyting 曾断言, 想要知道数字序列 0123456789 是否会出现在 π 的十进制展开中是完全不可能的. 但 Kanada 却在 1997 年发现该序列出现在 π 十进制展开的第 17387594880 位处. 直到 1989 年, Roger Penrose 还在他的《皇帝的新意 (The Emperor's New Mind)》一书第一版中大胆断言说我们将永远不会知道在 π 的十进制展开中是否会出现 10 个连续的 7. 而这个序列已被 Kanada 于 1997 年发现. 它位于第 22869046249 位处.

图 2 表明 1970 年以来 π 计算的进步, 其中那一条直线描绘了摩尔定律的长期发展趋势. 值得一提的是, 尽管 π 计算的进步在 1990 年代超过了摩尔定律, 在最近 10 年却落后在了摩尔定律的后面. 其原因可能部分地由于 π 计算程序不再能在系统规模上把快速傅里叶变换应用于乘法 (因为多数现代的超级计算机没有足够的网络带宽), 从而只能用效率较低的混合算法来计算.

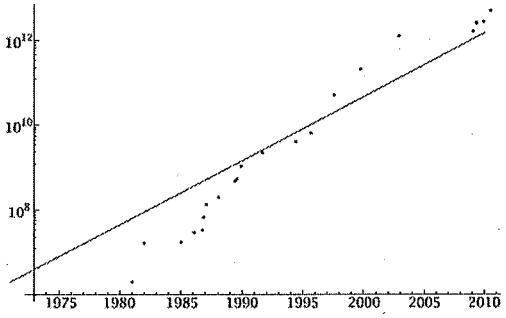

图 2 用数字 (点集) 给出的 π 计算示意图, 与摩尔定律的 (直线) 长期斜率相比

数字完整性 II. 在这些或者其他的大规模计算中有许多原因可以产生错误:

- 所使用的基本公式有错, 这一点可以想象;
- 实现这些算法的计算机程序一般都使用精巧的算法, 例如用快速傅里叶变换加速乘法, 因此很容易发生人为的编程错误;
- 这些计算通常都是在高度并行的计算机系统上运行的, 往往使用很容易出错的程序设计结构来控制并行;
- 硬件错误可能会出现. 这是 1986 年 π 计算中的一个问题, 参见上面的说明.

那么为什么还会有人相信这种计算的结果呢? 答案就是这种计算总是用另一个算法的独立计算来做双重检验的, 有时甚至是多重检验. 例如, Kanada 在 2002 年做的 1.3 万亿位 π 计算需要首先进行 1 万亿多一点的 16 进制位计算. 他发现 π 在 $10^{12}+1$ 位处开始的 20 个 16 进制位是 B4466E8D21 5388C4E014.

Kanada 然后用 "BBP" 算法 [7] 计算了这些 16 进制数字. 计算 π 的 BBP 算法是基于下列公式:

$$\pi = \sum_{i=0}^{\infty} \frac{1}{16^i}\left(\frac{4}{8i+1} - \frac{2}{8i+4} - \frac{1}{8i+5} - \frac{1}{8i+6}\right), \tag{7}$$

此公式是用 "PSLQ" 整数关系算法发现的 [16]. 整数关系方法找到或排除实数向量之间

的可能的有理关系. 在本千年之初, 它们被《科学和工程中的计算 (Computing in Science and Engineering)》封为 20 世纪 10 大算法之一. 其中最有效的是 Helaman Ferguson 的 PSLQ 算法 [10, 3].

PSLQ 最终生成了如下公式:

$$\pi = 4\ {}_2\mathrm{F}_1\left(\begin{matrix}1, \frac{1}{4} \\ \frac{5}{4}\end{matrix}\ \middle|\ -\frac{1}{4}\right) + 2\tan^{-1}\left(\frac{1}{2}\right) - \log 5. \tag{8}$$

其中 ${}_2\mathrm{F}_1\left(\begin{matrix}1, \frac{1}{4} \\ \frac{5}{4}\end{matrix}\ \middle|\ -\frac{1}{4}\right) = 0.955933837\ldots$ 是一个高斯超几何函数.

由 (8) 几乎可以直接推导出级数 (7). BBP 算法是基于 (7) 的. 它可以用来计算 π 的从任意一位开始的二进制或十六进制展开, 而不需要计算在这一位之前的任何数字, 为此只需使用一个简单的公式, 而且不要求很高精度的算术.

前面已经提到, BBP 计算的结果是 B4466E8D21 5388C4E014. 不用说, 尽管在两个计算中都有许多可能的错误的源泉, 它们的最终结果令人惊讶地一致, 从而 (以一种令人信服然而是启发式的方式) 肯定了两个结果都几乎肯定是正确的. 虽然不能严格地对此赋予一个 "概率", 但可以指出, 两个随机的 20 位 16 进制数字完全一致的机会是 $16^{20} \approx 1.2089 \times 10^{24}$ 分之一.

于是我们可以提如下的问题: 到底哪一种断言更靠谱? 是说 π 的 16 进制展开在 $10^{12}+1$ 位到 $10^{12}+20$ 位之间的数字串是 B4466E8D21 5388C4E014? 还是说最终结果是通过非常困难的数学推导得到的, 这个数学推导也许要写上百页或上千页, 要用到其他来源的许多已知结果, 而且 (经常是这样) 除了作者本人以外只有相对一小部分数学家能够或者已经仔细地读过它的细节?

雅虎云计算中心的 Tse-Wo Zse 最近在使用 BBP 公式所作的一次计算中计算了 π 的从 2000 万亿二进制位处开始的 256 个二进制位 [30]. 他用如下的 Bellard 版本的 BBP 公式检查了他的结果:

$$\pi = \frac{1}{64}\sum_{k=0}^{\infty}\frac{(-1)^k}{1024^k}\left(\frac{256}{10k+1} + \frac{1}{10k+1} - \frac{64}{10k+3} - \frac{4}{10k+5} - \frac{4}{10k+7} - \frac{32}{4k+1} - \frac{1}{4k+3}\right). \tag{9}$$

在这个实例中, 两个计算都验证了如下事实: 在 500 万亿个 16 进制位 (即 2000 万亿个二进制位) 后开始的 24 个 16 进制数字是 E6C1294A ED40403F 56D2D764. 在 [6] 中介绍了更多的最新相关计算.

8. 欧拉的 PHI 函数 ϕ

作为衡量什么东西随时间而改变, 什么东西不随时间改变的另一个度量, 考虑与欧拉的 PHI 函数 $\phi(n)$ 有关的两个猜想. 该函数给出所有小于 n, 并且和 n 互素的整数的个数.

Giuga 猜想 (1950) 整数 $n > 1$ 是素数, 当且仅当 $\mathcal{G}_n := \sum_{k=1}^{n-1} k^{n-1} \equiv n-1 \pmod{n}$.

如有反例, 必然是 *Carmichael* 数—— 这样的数比较稀少, 只是在 1994 年证明了

它有无穷多个. 还有更多的判别准则. 在 [11, p.227] 我们研究了如下事实: 如果一个数 $n = p_1 p_2 \cdots p_m$ 是 $m > 1$ 个素因子 p_i 的乘积, 如果它是 Giuga 猜想的反例 (即, 满足式子 $s_n \equiv n - 1 \pmod{n}$), 则对 $i \neq j$ 我们有 $p_i \neq p_j$, 并且

$$\sum_{i=1}^{m} \frac{1}{p_i} > 1,$$

p_i 构成一个 正规序列: 对于 $i \neq j$ 有 $p_i \not\equiv 1 \pmod{p_j}$. 因此, 3 的出现排除了 7, 13, 19, 31, 37, ..., 而 5 的出现排除了 11, 31, 41, ...

这个定理提供了足够的结构: 可以通过某些预测性的实验发现一些启发式规则, 用于构建有效的算法. 1995 年历时数月的计算显示出: 任何反例至少有 3459 个素因子, 因此大于 10^{13886}. 这个数几年以后通过在一个台式机上的 5 天计算增大为 10^{14164}. 所采用的策略是每次程序成功地运行以后都要进行一次自我校验. 但是如果超过 8135 个素因子, 这个方法就会失败; 我们希望有一天能够挖尽这个算法的潜力.

在撰写这篇文章的时候, 我们中的一个获得了相当好的结果: 在一个笔记本计算机上用 110 分钟完成了上界为 3050 个素数的计算, 并且用不到 14 个小时的时间完成了上界为 3486 个素数和 14000 个数字的计算; 然后又用 93 个 CPU 小时把这个成果扩大为 3678 个素数和 17168 个数字, 这次用的是一台 Macintosh Pro, 语言是 Maple 而非 C++, 因此速度要快几个数量级, 但是需要更加艰苦的编程.

还有一个同等困难的有关猜想, 该猜想的研究进展要少得多:

Lehmer猜想 (1932) 当且仅当 n 是素数时 $\phi(n) \mid (n-1)$. Lehmer 称此猜想为 "与存在奇完全数的猜想一样困难".

在此处, 反例的素因子又构成一个正规序列, 但是现在少了一点特殊结构. 在 1997 年 Simon Fraser 的一篇硕士论文中, Erick Wong 对于 14 个素数验证了此猜想. 他使用了正规性和 PARI, C++ 和 Maple 混合编程, 把上界推到 "指数灾难" 的地步. 这个非常聪明的计算一下子就使散布于文献中的前人结果成为昨日黄花, 全部归结成一种计算但也只能把以前的上界 13 推进到 14.

对于 Lehmer 在 1932 年提出的问题: 在什么情况下 $\phi(n) \mid (n+1)$? Wong 指出: 素因子不超过 7 个的解共有 8 个 (6 个素因子的解是由 Lehmer 给出的). 令:

$$\mathcal{L}_m := \prod_{k=0}^{m-1} F_k,$$

其中 $F_n := 2^{2^n} + 1$ 表示 费马素数. 该式的解是:

$$2, \mathcal{L}_1, \mathcal{L}_2, \ldots, \mathcal{L}_5,$$

再加上不大相称的一对: 4919055 和 6992962672132095. 但是分析哪怕只有 8 个因子的结果现在还没有看到. 这不啻于说, 在长达 70 年的时间里, 计算机只让排除性上界仅仅增长了一个素数.

Lehmer 在 1932 年未能分解 6992962672132097. 如果该数是素数的话, 第 9 个解就存在了: $\phi(n) \mid (n+1)$ 加上 $n+2$ 是素数这个条件蕴含: $N := n(n+2)$ 满足 $\phi(N) \mid (N+1)$. 我们设想: 如果 Lehmer—— 许多因式分解文献之父 —— 曾经意识到此数会有一个小的

因子的话，他肯定早就发现它能够被 73 整除了．今天，发现

$$6992962672132097 = 73 \cdot 95794009207289$$

只是举手之劳．而要完美回答 Lehmer 原来的问题，其困难程度依然如故．

9. 逆向计算和类 Apéry 级数

关于 Riemann zeta 函数，有 3 个公式特别有意思：

$$(a)\ \zeta(2) = 3\sum_{k=1}^{\infty}\frac{1}{k^2\binom{2k}{k}},\quad (b)\ \zeta(3) = \frac{5}{2}\sum_{k=1}^{\infty}\frac{(-1)^{k+1}}{k^3\binom{2k}{k}},\quad (c)\ \zeta(4) = \frac{36}{17}\sum_{k=1}^{\infty}\frac{1}{k^4\binom{2k}{k}}. \tag{10}$$

二项式恒等式 (10) 之 (a) 在两个世纪以前就已被人所知，而 (b) ——Apéry 在 1978 年给出的关于 $\zeta(3)$ 无理性的一个证明中探讨过 —— 至少在 1890 年就已被 Markov 发现，至于 (c) 则是由 Comtet 给出的 [3].

使用整数关系算法，自举算法，以及 "Pade" 函数 (Mathematica 和 Maple 都能生成很好的有理逼近), David Bradley 和我们中的一个 [3, 11] 在 1996 年意外地发现了如下的 $\zeta(4n+3)$ 的生成函数：

$$\sum_{k=0}^{\infty}\zeta(4k+3)x^{4k} = \frac{5}{2}\sum_{k=0}^{\infty}\frac{(-1)^{k+1}}{k^3\binom{2k}{k}(1-x^4/k^4)}\prod_{m=1}^{k-1}\left(\frac{1+4x^4/m^4}{1-x^4/m^4}\right). \tag{11}$$

注意，此公式使我们能够从中读出 $\zeta(4n+3)$, $n>0$, 的无穷多个公式，这些公式可以从 (10) (b) 开始，通过比较等式左边和右边 x^{4k} 的系数而得到.

10 年以后，通过一个类似的然而专门得多的实验过程 (详见 [3]), 我们成功地发现了 $\zeta(2n+2)$ 的一个类似的一般公式，该公式令人惬意地平行于 (11):

$$\sum_{k=0}^{\infty}\zeta(2k+2)x^{2k} = 3\sum_{k=1}^{\infty}\frac{1}{k^2\binom{2k}{k}(1-x^2/k^2)}\prod_{m=1}^{k-1}\left(\frac{1-4x^2/m^2}{1-x^2/m^2}\right). \tag{12}$$

与 (11) 一样，我们现在也可以读出从 (10) (a) 开始的无穷多个公式．作者在 1996 年成功地把 (11) 归约为一个有穷的形式，但是未能证明它．而一年以后 Almquist 和 Granville 把它证明了．又过了 10 年, Wilf-Zeilberger 算法 [29, 23] 直接通过在 Maple 中的实现证明了 (12) [10, 3], 该算法的发明者们因此得到了 Steele 奖．换句话说，(12) 的发现和证明都是靠了计算机.

当 $x=0$ 时，我们为 $\zeta(2n+4)$ 找到了一个可以比较的生成函数，给出了 (10) (c), 但是未能为 $\zeta(4n+1)$ 找到这样的函数.

10. π 的倒数的级数

大概在 1910 年左右，Ramanujan 在椭圆积分的基础上发现了 $1/\pi$ 的真正新的级数 [3, 10, 31]. 其中之一是：

$$\frac{1}{\pi} = \frac{2\sqrt{2}}{9801}\sum_{k=0}^{\infty}\frac{(4k)!(1103+26390k)}{(k!)^4 396^{4k}}. \tag{13}$$

(13) 的每一项添加 8 个正确的数字. 1985 年, Gosper 利用 (13) 计算了 π 的当时成为创纪录的 1700 万位数字 —— 与此同时完成了 (13) 的第一个证明 [10, 第 3 章]. 不久之后, David Chudnovsky 和 Gregory Chudnovsky 获得了如下的结果，它位于二次数域 $\mathbb{Q}(\sqrt{-163})$ 之

中而不是在 $\mathbb{Q}(\sqrt{58})$ 之中：

$$\frac{1}{\pi} = 12\sum_{k=0}^{\infty} \frac{(-1)^k(6k)!(13591409+545140134k)}{(3k)!(k!)^3 640320^{3k+3/2}}. \tag{14}$$

(14) 的每一项添加 14 个正确的数字. 弟兄俩多次使用这个公式，终于在 1994 年的 π 计算中获得了 40 亿以上的十进制位数字. 《纽约人 (New Yorker)》在一篇获奖的文章 [26] 中登载了这个不平凡的故事. 有意思的是，就像我们前面已经指出的那样, (14) 又在 2009 年下半年被使用而且创造了目前的 π 计算记录.

Wilf-Zeilberger在继续发挥作用. 几年以前, Jesus Guillera 找到了一组类似 Ramanujan 发现过的那样的 π 恒等式. 他使用的是整数关系方法. 3 个最基本 —— 而且完全是有理的 —— 恒等式是：

$$\frac{4}{\pi^2} = \sum_{n=0}^{\infty}(-1)^n r(n)^5(13+180n+820n^2)\left(\frac{1}{32}\right)^{2n+1}, \tag{15}$$

$$\frac{2}{\pi^2} = \sum_{n=0}^{\infty}(-1)^n r(n)^5(1+8n+20n^2)\left(\frac{1}{2}\right)^{2n+1}, \tag{16}$$

$$\frac{4}{\pi^3} \stackrel{?}{=} \sum_{n=0}^{\infty} r(n)^7(1+14n+76n^2+168n^3)\left(\frac{1}{8}\right)^{2n+1}, \tag{17}$$

其中 $r(n) := (1/2\cdot 3/2\cdots(2n-1)/2)/n!$.

Guillera 通过照搬应用 Wilf-Zeitberger 算法 [29, 23] 先后证明了 (15) 和 (16), 并用此结果形式地证明类似超几何风格的恒等式 [10, 3, 19, 31]. 没有其它已知的证明，并且好像当 $N \geq 4$ 时没有 $1/\pi^N$ 一类的公式. 第 3 个公式 (17), 几乎可以肯定是对的. Guillera 把它归功于 Gourevich, 后者使用整数关系方法找到了它.

我们能用 30 位数字算术 "发现" (17), 我们用了 10 秒钟的时间把它一直检查到第 500 位数字，然后又用 6.25 分钟的时间把它检查到 1200 位数字，最后用了 25 分钟把它检查到 1500 位数字，所有这些检查都是用朴素的 Maple 命令行指令完成的. 但是这里并没有证明，也没有人提出过如何去证明它的想法；特别是，正像实验所提示的，它与 (15) 和 (16) 没有任何 "相配" 之处 [3]. 我们的直觉是：如果存在一个证明的话，它应该更像一个验证而不是一个解释，于是我们就放弃了寻找证明. 我们只需知道这个漂亮的恒等式是真的，我们就会很高兴 (当然如果最终证明它不成立的话，会更加引人关注). 也许不需要很好的理由就可以认为它就是真的 —— 也许证明根本就不存在. 这是 Gödel 型命题的一个很好的实例.

2008 年，Guillera [19] 给出了另外一对可爱的第 3 个千禧年不等式 —— 是用整数关系方法发现，并且用创造性的合成方法证明的 —— 这一次是对于 π^2 而不是它的倒数. 它们是：

$$\sum_{n=0}^{\infty}\frac{1}{2^{2n}}\frac{\left(x+\frac{1}{2}\right)_n^3}{(x+1)_n^3}(6(n+x)+1) = 8x\sum_{n=0}^{\infty}\frac{\left(\frac{1}{2}\right)_n^2}{(x+1)_n^2}, \tag{18}$$

和

$$\sum_{n=0}^{\infty}\frac{1}{2^{6n}}\frac{\left(x+\frac{1}{2}\right)_n^3}{(x+1)_n^3}(42(n+x)+5)=32x\sum_{n=0}^{\infty}\frac{\left(x+\frac{1}{2}\right)_n^2}{(2x+1)_n^2},\tag{19}$$

此处 $(a)_n=a(a+1)\cdots(a+n-1)$ 是递增的阶乘. 对 (18) 和 (19) 中用 $x=1/2$ 实施置换，他分别得到了如下公式：

$$\sum_{n=0}^{\infty}\frac{1}{2^{2n}}\frac{(1)_n^3}{\left(\frac{3}{2}\right)_n^3}(3n+2)=\frac{\pi^2}{4},\qquad\sum_{n=0}^{\infty}\frac{1}{2^{6n}}\frac{(1)_n^3}{\left(\frac{3}{2}\right)_n^3}(21n+13)=4\frac{\pi^2}{3}.$$

11. 证明的形式化验证

Kepler 在 1611 年描述了把等积球体堆积成如像我们在食品店里看到的橙子堆积的那种形状. 他断言这种堆积是所有可能的堆积中最紧密的. 这个断言现在被称为 Kepler 猜想，并且持续数世纪无人能严格证明. Hilbert 在他于 1900 年提出的著名未解问题清单的第 18 个未解问题中隐含地提到了 Kepler 猜想的非规范情况 ——是否存在充填空间的非正则多面体?—— 而正则的情况已由 Gauss 在 1831 年解决.

目前正在匹茨堡大学工作的 Thomas Hales 于 1994 年提出了一个 5 步走方案，此方案可能导致一个证明：(a) 处理只含三角面的图；(b) 证明面心立方体和六边封闭的包装是在强意义下的局部极大，亦即，对于同一个图，它们比任何一个 Delaunay 星形获分更多；(c) 处理只含三角面和四角面的图 (除五角棱柱以外)；(d) 处理除三角和四角面以外也包含其他成分的图；(e) 处理五角棱柱.

1998 年,Hales 宣布他的计划已经完成，其中关键的第 5 步是由 Samuel Ferguson (数学家 – 雕塑家 Helaman Ferguson 之子) 完成的. 这项工作涉及大量的计算，使用了一个区间算术包，一个图形生成器和 Mathematica. 包括源代码和计算结果在内的计算机文件占据了 3 个 G 以上的磁盘空间. 其它细节，包括论文，可在 http://www.math.pitt.edu/thales/kepler98 上获取. 由于种种原因 —— 其中不一定每一条都能完全站住脚 —— 《数学年刊 (Annales of Mathematics)》最初决定在发表 Hales 的文章时加一个谨慎的免责附注，但是在出版前还是去掉了.

Hales [20] 现在开始了一个长达数年的努力来用基于计算机的形式化方法验证他的证明. 他把这个项目称为 "污点" 项目. 由于这些技术变得更容易理解了，我们可以预期：一大批数学结果将最终被计算机确认. 与 Hales 的文章发表在同一期《美国数学会通讯》的其它文章也表示了同样的观点.

12. 计算的极限

如下的例子值得注意：

$$\int_0^{\infty}\cos(2x)\prod_{n=1}^{\infty}\cos(x/n)dx\tag{20}$$
$$=0.392699081698724154807830422909937860524645434187231595926\ldots$$

利用在 [5] 中描述的一个算法，可以把这个积分计算到很高的精度. 当我们初次做这个计

算时，我们以为结果将是 $\pi/8$, 但是经过用数值

$$0.392699081698724154807830422909937860524646174921888227621\ldots$$

作仔细检验终于搞清楚了，这两个值从第 43 位起就不一样！

后来 Richard Crandall [15, §7.3] 解释了其中的奥妙. 通过对一个 *燃料用完* 随机游走的基于物理意义的分析，他指出 $\pi/8$ 可以表示为如下的非常迅速收敛的级数展开，而公式 (20) 仅仅是它的首项.

$$\frac{\pi}{8} = \sum_{m=0}^{\infty} \int_0^{\infty} \cos\left[2(2m+1)x\right] \prod_{n=1}^{\infty} \cos\left(x/n\right) dx. \tag{21}$$

仅取此级数的两个项就足以使上述两个值一直到第 500 位都一样.

作为一个最后的严肃的例子，我们给出如下的 "二年级学生之梦" 恒等式:

$$\sigma_{29} := \sum_{n=-\infty}^{\infty} \operatorname{sinc}(n) \operatorname{sinc}(n/3) \operatorname{sinc}(n/5) \cdots \operatorname{sinc}(n/23) \operatorname{sinc}(n/29) \tag{22}$$

$$= \int_{-\infty}^{\infty} \operatorname{sinc}(x) \operatorname{sinc}(x/3) \operatorname{sinc}(x/5) \cdots \operatorname{sinc}(x/23) \operatorname{sinc}(x/29) dx, \tag{23}$$

其中的分母遍历首先通过经验发现的奇素数. 更一般地，考察

$$\begin{aligned} \sigma_p &:= \sum_{n=-\infty}^{\infty} \operatorname{sinc}(n) \operatorname{sinc}(n/3) \operatorname{sinc}(n/5) \operatorname{sinc}(n/7) \cdots \operatorname{sinc}(n/p) \\ &\stackrel{?}{=} \int_{-\infty}^{\infty} \operatorname{sinc}(x) \operatorname{sinc}(x/3) \operatorname{sinc}(x/5) \operatorname{sinc}(x/7) \cdots \operatorname{sinc}(x/p) dx. \end{aligned} \tag{24}$$

可以证明如下的结论是对的：这个关于 σ_p 的 "和等于积分" 的恒等式至少当 p 是前 10176 个素数时是对的，但一旦 p 大于某个大素数，此恒等式就不再成立了. 从那时以后 "和小于积分" 是严格正确的，但是它们的 *差总是远远小于某个大数* (10^{100}) *分之一*. 在假设广义 Riemann 猜想成立的前提下可以得到更强的估计 (见 [15, §7] 和 [8]).

13. 结论

中心问题是如何看待通过实验发现的结果. 这一点在之前已有人讨论过. 1993 年 Arthur Jaffe 和 Frank Quinn 就曾对没有完全严格证明的数学结果之激增提出警告，并且提出了一个框架，指示 "思辨的" 数学应该扮演怎样的 "健康和正面" 角色 [21]. 对此，许多著名的数学家给出了回答 [1]. 例如 Morris Hirsch 回应说，甚至 Gauss 都曾发表过未完成的证明，而且那个关于有限群分类的长达 15000 页的证明也引发了问题：我们到底何时应该证明一个结果. 他建议我们对每一个证明贴上一个标签 —— 例如，"计算机辅助的", "团队合作的", "构造性的", 等等. Saunders Mac Lane 嘲弄说 "我们不是仅仅因为信念而获救的，而是因为信念和工作而获救的." 意思是我们同时需要直观性的工作和精确性的工作.

与此同时，现在的计算工具提供了很值得重视的设施，帮助我们验证用解析方法获得的结果，例如正在开发中的一些工具可用于检查包含许多方程的文稿中的恒等式，又如在 Hales 的项目中用形式化方法验证 Kepler 猜想.

在我们这个充满信息的世界上，要想阻遏那到处激增的信息和工具是不可能的. 我们应该学习和教授如何去判断何时应该使用那些数字化的东西. 这意味着要掌握我们在

上面解释的那种种技术，以及搞清楚为什么一个软件系统能够做它所做的事．这要求我们知道何时一个计算 —— 在原则上或者在实际上 —— 是一个严格的证明，或者能够变成一个严格的证明．我们还应该知道何时一个计算仅仅提供有说服力的证据，何时则完全是误导．例如，甚至最好的商业线性规划程序包，如像 Hales 使用的那种，也不能证实每个解，尽管它的代码是几乎可保证是正确的．这要求重新调整我们对什么是困难的和什么是容易的等问题的看法．

这也要求我们开发一种课程，用它来谨慎地教授实验性的、计算机辅助的数学．在这方面有一些人正在作出努力，包括 Grinnell 学院的 Marc Chamberland (http://www.math.grin.edu/čhamberl/courses/MAT444/syllabus.html), Tulane 大学的 Victor Moll, 墨尔本的 Jan de Gier, 以及 Queensland 大学的 Ole Warnaar.

Judith Grabiner 曾经指出，近代数学中严格性的发展，其动力源自拿破仑时代引进的严格的教程：课程和课本强迫人们接受精确性和程序性，而这是学徒们所拒绝接受的．但是永远不会出现类似归纳的数学取代证明的情况．我们需要找到一种新的平衡．换句话说，我们只是开始探索能够丰富数学的新的道路．正如 Jacques Hadamard 所说 [25]:

"数学严格性的目的是约束直观的领地并使之合法化，除此以外再没有其他目的．"

我们从来没有过像现在这样丰富的手段来制造直观．我们面临的挑战是学习如何去驾驭它们，如何去开发它们，以及如何去传播必需的理论和实践．由我们中的一个领导的"计算机辅助的数学研究及其应用研究中心" (CARMA), http://carma.newcastle.edu.au/, 希望在这个事业中负起领导责任．在我们看来，这个事业包含了探索性实验和严格证明的一个激动人心的组合．

参考文献 (略)

(陆汝钤 译　　袁向东 校)

****

(上接 54 页)

它涉及一些属于非全纯 Maass 形的 $L-$函数，从而最终得到由属于全纯权 2 形式的正交正规基的 $L-$函数表示的恒等式．这个恒等式的完整推导见 [13]．在某些方面，[13] 中的工作涉及兰和我预想的第一步，也就是发现已知的和新的 zeta 函数的新关系，在某种意义上很可能是正则化．

在我与兰完成文章 [19] 后不久，我们详细讨论了从"阶梯"可以获得结果的想法和希望．他曾对我说："我多么希望自己重回到 30 岁，这样我可以专注热核．"鉴于赛尔日献身于数学的方式，让我们把这个声明作为总结他对热核和热核分析的威力的真诚而深切的信念．

参考文献 (略)

(高燕芳 译　　赵振江 校)