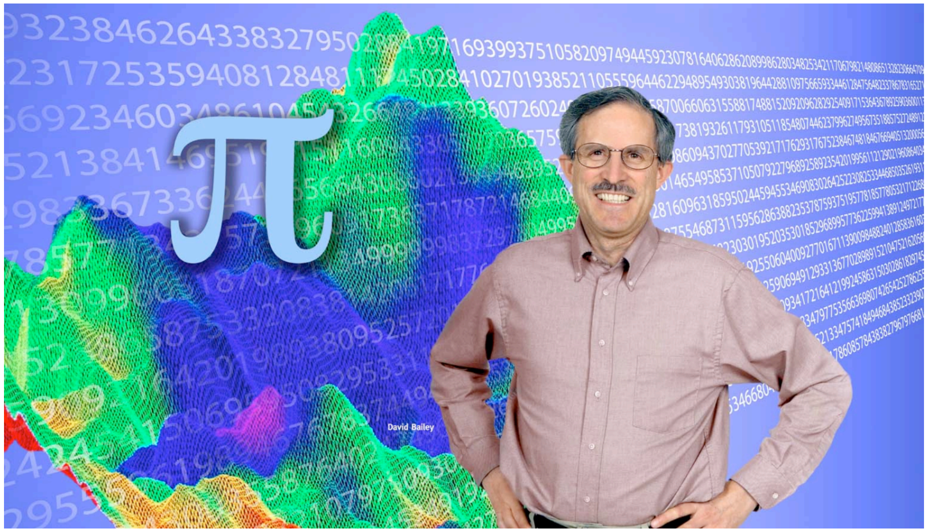
Experimental Mathematics and High-Performance Computing

David H Bailey
Lawrence Berkeley National Laboratory, USA

This talk is available at:

http://crd.lbl.gov/~dhbailey/dhbtalks/dhb-expmath-hpc.pdf



High-Performance Computing: Progress and Challenges

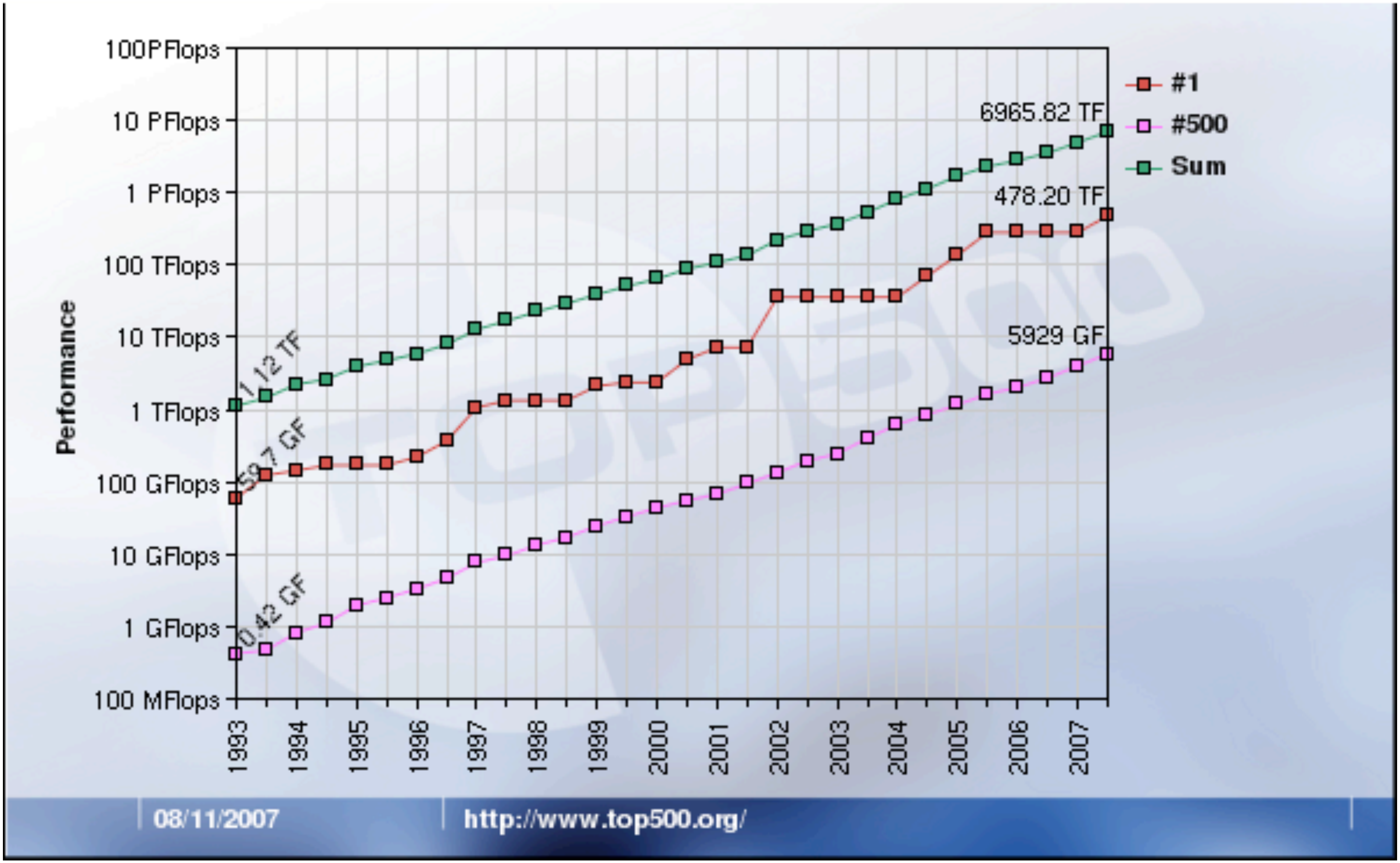
Progress to date:

- Thousands of demanding scientific applications have been successfully implemented on both single-user and highly parallel systems.
- Performance has advanced at an exponential rate (following Moore's Law) for over 30 years.
- Major advances have also resulted from improved numerical algorithms and progamming techniques.
- Numerous new disciplines have recently begun to use this technology.

Challenges that lie ahead:

- The era of relentless increase in clock speed is over.
- High-end systems of the future will feature millions of multi-core processors.
- Scientific applications on future high-end systems must exhibit and exploit enormous concurrency (e.g., roughly 10^{10}-way concurrency on exascale systems).
- System software (operating systems, compilers, performance tools etc.) must also be retargeted to these extremely highly parallel systems.

Top500 Performance Trends



Future Fusion Simulation Requirements

Tokamak turbulence (GTC) -- For ITER experiment, etc:

- Grid size: 10,000 x 4000 x 256, or about 10^{10} gridpoints.
- Each grid cell contains 8 particles, for total of 8 x 10^{10}.
- 200,000 time steps required.
- Improved plasma model (increases by 10-100X).
- Total cost: 6 x 10^{21} flop = 1 hours on 1 Eflop/s system; 10 Pbyte main memory.

All-Orders Spectral Algorithm (AORSA) – to address absorption of RF electromagnetic waves in plasmas.

Present Day:

- 120,000 x 120,000 complex linear system requires 230 Gbyte memory, 1.3 hours on 1 Tflop/s.
- 300,000 x 300,000 linear system requires 8 hours.

Future (ITER scale):

- 6,000,000 x 6,000,000 system = 1 hour on 1 Eflop/s system; 1 Pbyte memory.

Future Astrophysics Computation Requirements

Supernova simulation:

- 3-D model calculations will require 1M processor-hours per run, on 1 Pflop/s system, or 1000 hours per run on 1 Eflop/s system.

Analysis of cosmic microwave background data:

- WMAP (now) — 3×10^{21} flops, 16 Tbyte mem
- PLANCK (2007) — 2×10^{24} flops, 1.6 Pbyte mem
- CMBpol (2015) — 1×10^{27} flops, 1 Ebyte mem

Note: Microwave background data analysis, and also supernova data analysis, involves mountains of experimental data, not simulation data.

The NERSC Computer Center at the Berkeley Laboratory

- Seaborg: 6656-CPU IBM P3 system, 10 Tflop/s peak.
- Bassi: 976-CPU IBM P5 system, 6.7 Tflop/s peak.
- Jacquard: 356 dual-processor Opteron nodes, 3.13 Tflop/s peak.

- Now available: Franklin: 9672 dual-core Opteron CPUs, 100 Tflop/s peak.

The SciDAC Program of the U.S. Department of Energy

Scientific Discovery through Advanced Computing (SciDAC) – a $75 M/yr program to advance high-performance computing for DOE missions.

Scientific applications:

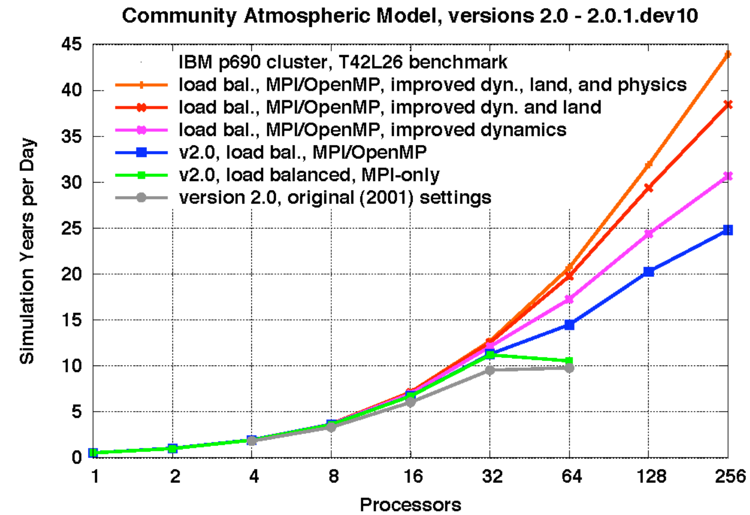
- Physics: Accelerator design, astrophysics, particle physics, cosmology.
- Climate modeling: Full-scale, long-term modeling of climate.
- Groundwater simulation: Long-term environmental hazards.
- Fusion energy: Plasma physics, reactor design.
- Biology: Ethanol production, protein function.
- Material science and chemistry: Nanoscience, quantum chemistry, electronic structure calculations.

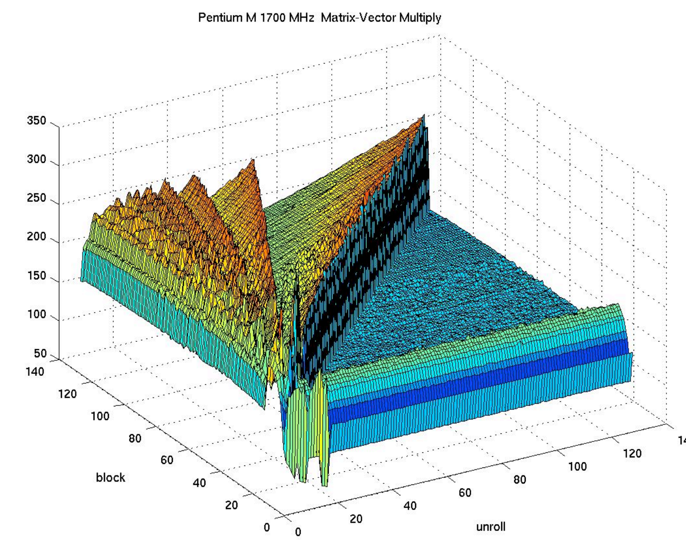
Mathematics and computer science:

- Applied mathematics: Improved algorithms and parallelization schemes.
- Computer science: Performance tools, data storage, networking.
- Visualization: "Seeing" the data on petascale systems.

The SciDAC Performance Engineering Research Institute (PERI)

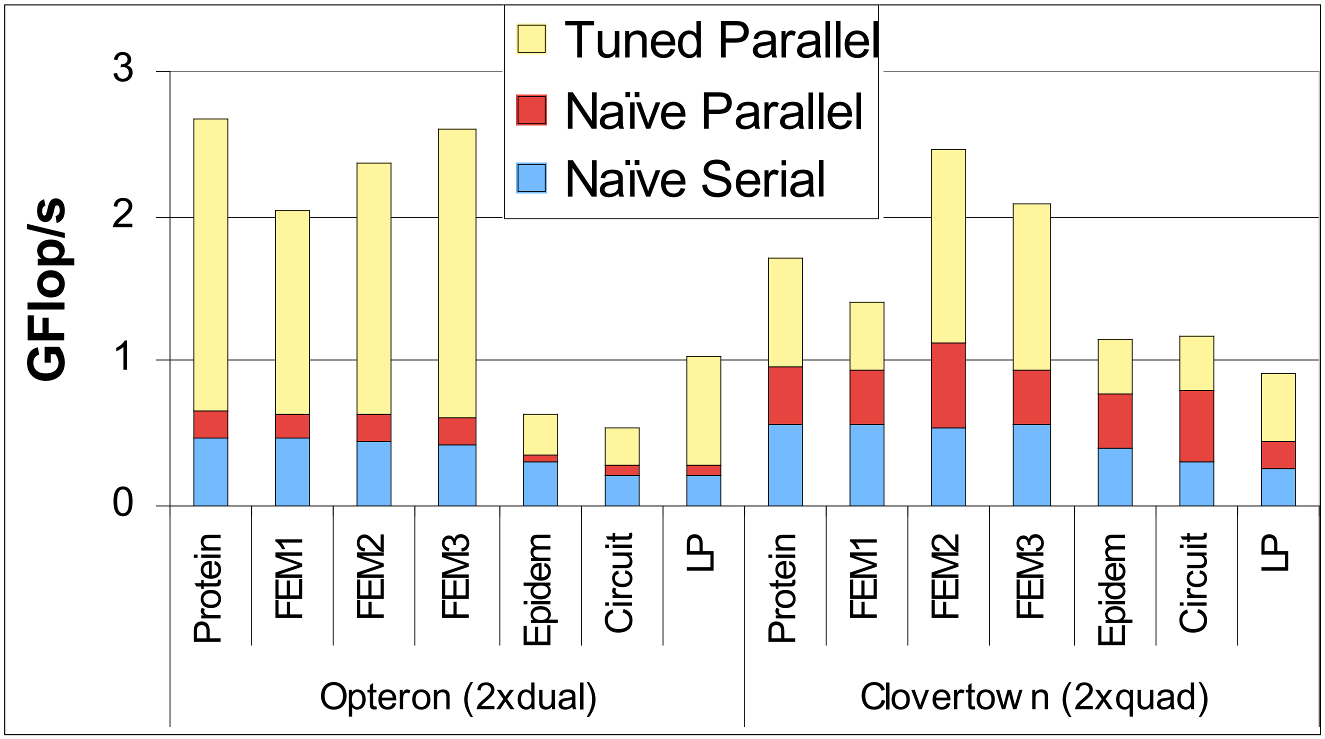
- Participating institutions: Argonne, LBNL, LLNL, Oak Ridge, Rice, UCSD, U Maryland, UNC, USC, U Tennessee.
- Lead investigators: Robert Lucas, USC/ISI and David H Bailey, LBNL.
- Funding: $4 million per year.
- Mission: To improve the performance of DOE-funded science applications on high-end computing platforms.
- Component activities:
  - Performance modeling.
  - Automatic performance tuning.
  - Application engagement.





Autotuning a Sparse Matrix Multiply Kernel

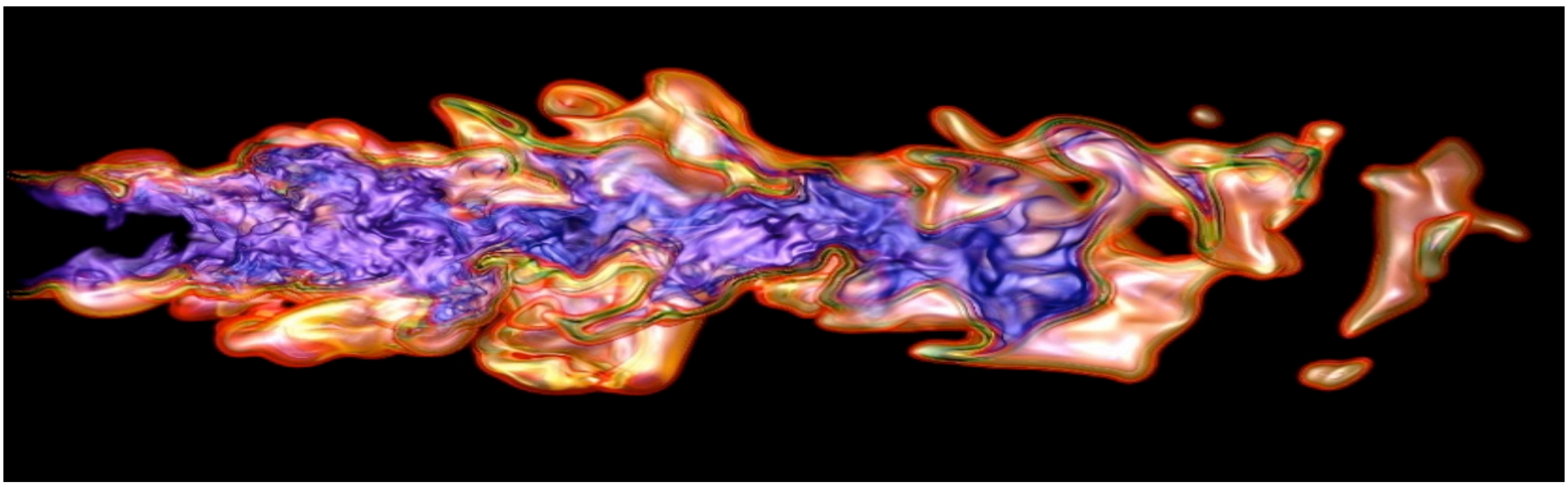
- Register Block tuning to compress matrix data structure (choose *r1xr2*).
- Cache Block tuning so corresponding vectors fit in local memory (*c1xc2*).
- Dividing matrix evenly among multiple cores and sockets (*p1xp2*).
- Prefetching for some distance *d.*
- Using machine-specific code (SSE, etc).



S. Williams, L. Oliker, R. Vuduc, J. Shalf, K. Yelick and J. Demmel, "Optimization of Sparse Matrix-Vector Multiplication on Emerging Multicore Platforms," *Proceedings of SC07*, 2007.

S3D Performance Improvement

- S3D is a large SciDAC scientific application code for simulation of combustion and turbulence.
- Recently a “tiger team” of researchers in the PERI project, in conjunction with the S3D team, analyzed the performance profile of S3D.
- After code changes to improve cache blocking, and insertion of a custom exp routine, preliminary figures indicate roughly 20% speedup on a Cray system.
- These changes are potentially worth hundreds of thousands of dollars per year in high-end computer time.



Contributors: B. de Supinski, J. Mellor-Crummey, M. Fagan, N. Wright, A. Snavely, D. Bailey, R. Vuduc, S. Shende, A. Morris, A. Malony, K. Huck, J. Larkin, J. Chen, D. Lignell, K. Roche, P. Worley.

Experimental Mathematics

"Experimental mathematics" is the name for the emerging discipline of applying high-performance computing in novel ways to problems in mathematics and mathematical physics:

- **High-precision computation -- the electron microscope of mathematics**. [numerical analysis, computer science]
- **Numerical evaluations of series, integrals, limits**. [numerical analysis]
- **Integer relation detection**. [math, computer science]
- **Symbolic computation**. [math, computer science]
- **Computer-based proofs**. [math, computer science]
- **Visualization and data management**. [computer science]
- **Applications**. [math, physics]
- **Highly parallel implementations of all the above**. [computer science]

Mathematicians, computer scientists, numerical analysts and physicists (undergraduates as well as senior researchers) have all made contributions.

LBNL’s High-Precision Software: ARPREC and QD

- QD: Double-double (32 digits) and quad-double (64 digits) .
- ARPREC: Arbitrary precision (hundreds or thousands of digits).
- Low-level routines written in C++.
- High-level C++ and F-90 translation modules permit use with existing programs with only minor code changes.
- Integer, real and complex datatypes.
- Many common functions: sqrt, cos, exp, gamma, etc.
- PSLQ, root finding, numerical integration.
- An interactive “Experimental Mathematician’s Toolkit.”
- Can easily be incorporated into a highly parallel program.

Available at: http://www.experimentalmath.info

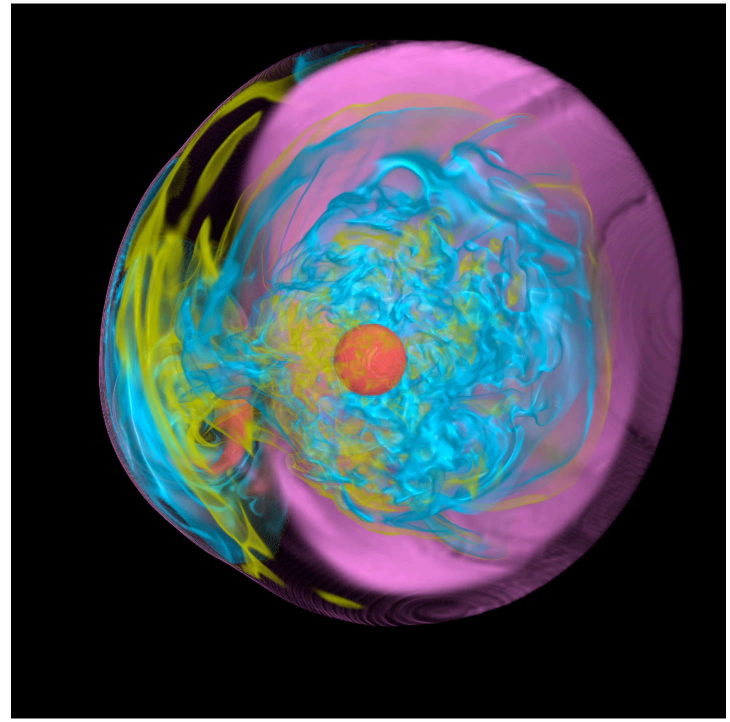
Other widely used high-precision software:

- GMP: http://gmplib.org
- MPFR: http://www.mpfr.org

D. H. Bailey, Y. Hida, X. S. Li and B. Thompson, "ARPREC: An Arbitrary Precision Computation Package," manuscript, Sept 2002, http://crd.lbl.gov/~dhbailey/dhbpapers/arprec.pdf.

High-Precision Arithmetic: Supernova Simulations

- Researchers are using QD to solve for equilibrium populations of iron and other atoms in the atmospheres of supernovas.
- Iron may exist in several species, so it is necessary to solve for all species simultaneously.
- Since the relative population of a species is proportional to the exponential of the ionization energy, the dynamic range of these values can be very large.
- The quad-double (64-digit) portion now dominates the entire computation.



P. H. Hauschildt and E. Baron, “The Numerical Solution of the Expanding Stellar Atmosphere Problem,” *Journal Computational and Applied Mathematics*, vol. 109 (1999), pg. 41-63.

High-Precision Arithmetic: Planetary Orbit Calculations

- A key question of planetary theory is whether the solar system is stable over cosmological time frames (billions of years).
- Scientists have studied this question by performing very long-term simulations of chaotic planetary motions.
- Simulations typically do well for long periods of time, but then fail at certain key junctures, unless special measures are taken.
- Researchers have found that double-double or quad-double arithmetic is required to avoid severe numerical inaccuracies, even if other techniques are employed.

"The orbit of any one planet depends on the combined motions of all the planets, not to mention the actions of all these on each other. To consider simultaneously all these causes of motion and to define these motions by exact laws allowing of convenient calculation exceeds, unless I am mistaken, the forces of the entire human intellect." [Isaac Newton, 1687]

G. Lake, T. Quinn and D. C. Richardson, "From Sir Isaac to the Sloan Survey: Calculating the Structure and Chaos Due to Gravity in the Universe," *Proceedings of the Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, SIAM, Philadelphia, 1997, pg. 1-10.

The PSLQ Integer Relation Algorithm: A Tool to Recognize Numerical Constants

Let (x_n) be a given vector of real numbers. An integer relation algorithm finds integers (a_n) such that

$$a_1 x_1 + a_2 x_2 + \cdots + a_n x_n = 0$$

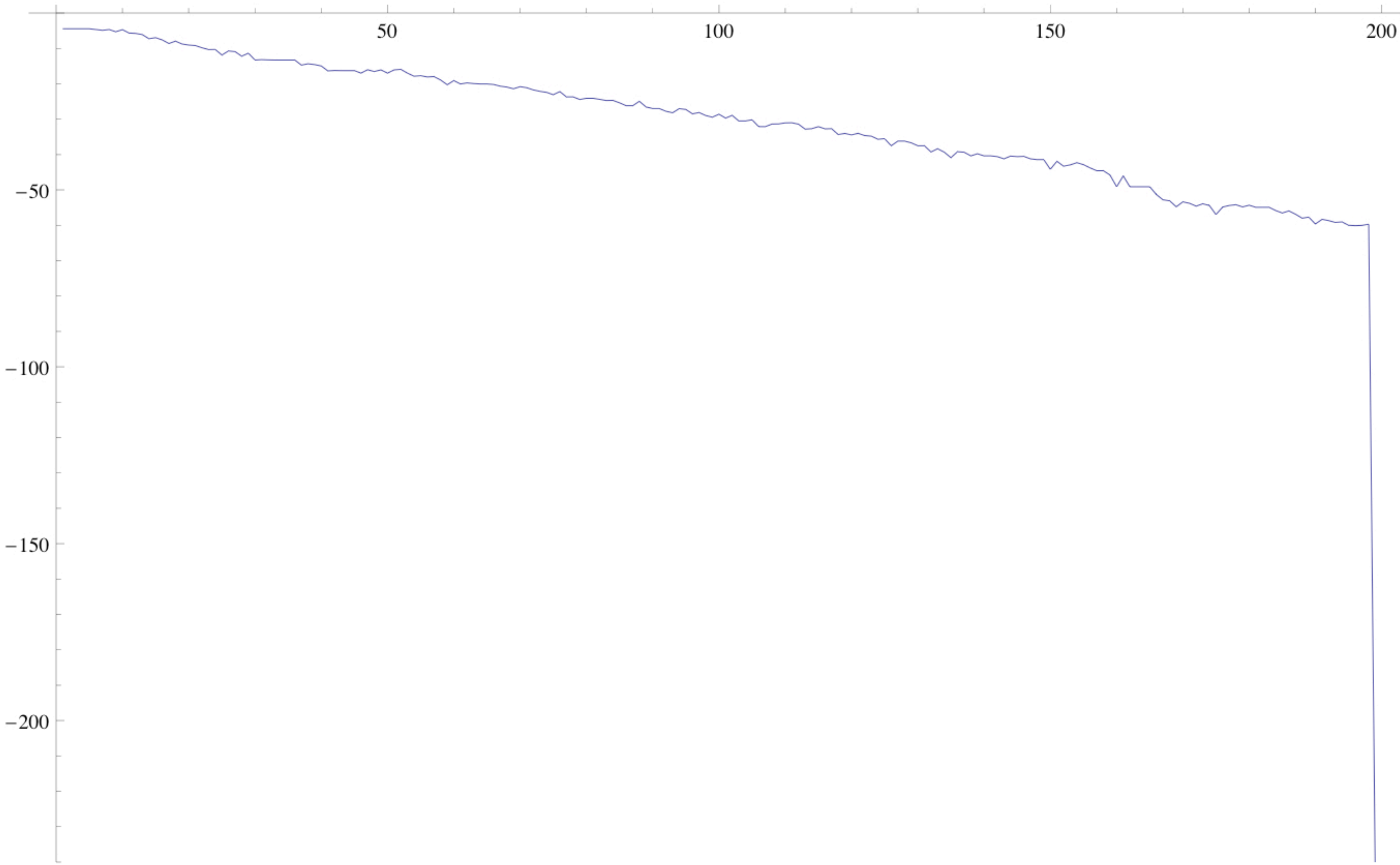
(or within "epsilon" of zero, where epsilon = 10^{-p} and p is the precision).

At the present time the "PSLQ" algorithm of mathematician-sculptor Helaman Ferguson is the most widely used integer relation algorithm. It was named one of ten "algorithms of the century" by *Computing in Science and Engineering*.

PSLQ (or any other integer relation scheme) requires very high precision (at least n*d digits, where d is the size in digits of the largest a_k), both in the input data and in the operation of the algorithm.

1. H. R. P. Ferguson, D. H. Bailey and S. Arno, "Analysis of PSLQ, An Integer Relation Finding Algorithm," *Mathematics of Computation*, vol. 68, no. 225 (Jan 1999), pg. 351-369.
2. D. H. Bailey and D. J. Broadhurst, "Parallel Integer Relation Detection: Techniques and Applications," *Mathematics of Computation*, vol. 70, no. 236 (Oct 2000), pg. 1719-1736.

Decrease of $\log_{10}(\min |x_i|)$ in PSLQ

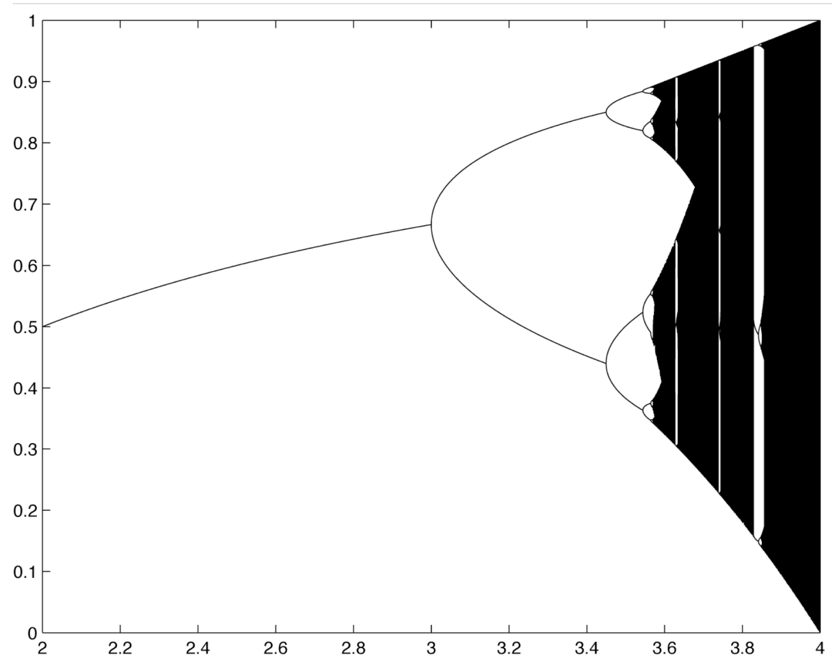


Application of PSLQ: Bifurcation Points in Chaos Theory

Let t be the smallest r such that the "logistic iteration"

$$x_{n+1} = r x_n (1 - x_n)$$

exhibits 8-way periodicity instead of 4-way periodicity.



By means of an iterative scheme, one can obtain the numerical value of t to any desired precision:

3.544090359551922853615965986604804540583099845444573675457 81…

Applying PSLQ to the vector (1, t, t^2, t^3, …, t^{12}), one finds that t is a root of the polynomial:

$$\begin{aligned} 0 &= 4913 + 2108t^2 - 604t^3 - 977t^4 + 8t^5 + 44t^6 + 392t^7 \\ &\quad -193t^8 - 40t^9 + 48t^{10} - 12t^{11} + t^{12} \end{aligned}$$

Some Supercomputer-Class PSLQ Solutions

- Identification of B_4, the fourth bifurcation point of the logistic iteration:
  Integer relation of size 121. 10,000-digit arithmetic.
- Identification of Apery sums.
  15 integer relation problems, with size up to 118. 5,000-digit arithmetic.
- Identification of Euler-zeta sums.
  Hundreds of integer relation problems, each of size 145. 5,000-digit arithmetic.
- Finding recursions in Ising integrals.
  Over 2600 high-precision numerical integrations, and integer relation detections. 1500-digit arithmetic. Run on Apple system at Virginia Tech – 12 hours on 64 CPUs.
- Finding a relation involving a root of Lehmer's polynomial.
  Integer relation of size 125. 50,000-digit arithmetic. Utilizes 3-level, multi-pair parallel PSLQ program. Run on IBM parallel system – 16 hours on 64 CPUs.

At the present time, the dominant cost in most applications of PSLQ is computing the constants involved to high precision. However, this may change in the future.

1. D. H. Bailey and D. J. Broadhurst, "Parallel Integer Relation Detection: Techniques and Applications," *Mathematics of Computation*, vol. 70, no. 236 (Oct 2000), pg. 1719-1736.
2. D. H. Bailey, D. Borwein, J. M. Borwein and R. Crandall, "Hypergeometric Forms for Ising-Class Integrals," *Experimental Mathematics*, to appear, 2007, http://crd.lbl.gov/~dhbailey/dhbpapers/meijer.pdf.

Fascination With Pi

Newton (1670):

"I am ashamed to tell you to how many figures I carried these computations, having no other business at the time."

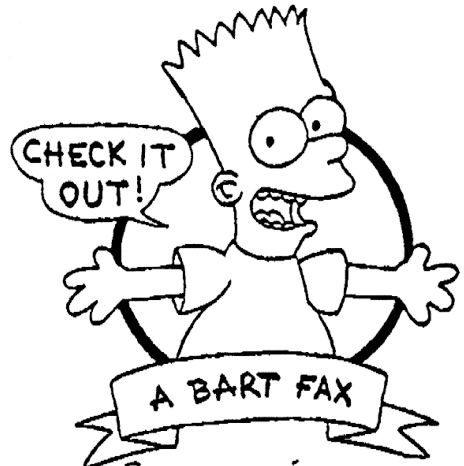
Carl Sagan (1986):

In his book "Contact," the lead scientist (played by Jodie Foster in the movie) looked for patterns in the digits of pi.

New York Times (2007):

On March 14 (03/14) the daily crossword puzzle featured a pi theme: key answers included "pi" in the place of a single character.

Fax from “The Simpsons” Show



TO: DAVID BAILEY

FROM: JACQUELINE ATKINS

DATE: 10/9/92

NUMBER OF PAGES: 1

FAX (310) 203-3852

PHONE (310) 203-3959

A Professor at UCLA told me that you might be able to give me the answer to: What is the 40,000th digit of Pi?

We would like to use the answer in our show. Can you help?

The Borwein-Plouffe Observation

In 1996, Peter Borwein and Simon Plouffe observed that the following well-known formula for $\log_e 2$

$$\log 2 = \sum_{n=1}^{\infty} \frac{1}{n 2^n} = 0.69314718055994530942\ldots$$

leads to a simple scheme for computing binary digits at an arbitrary starting position (here {} denotes fractional part):

$$\begin{aligned} \left\{2^d \log 2\right\} &= \left\{\sum_{n=1}^{d} \frac{2^{d-n}}{n}\right\} + \sum_{n=d+1}^{\infty} \frac{2^{d-n}}{n} \\ &= \left\{\sum_{n=1}^{d} \frac{2^{d-n} \bmod n}{n}\right\} + \sum_{n=d+1}^{\infty} \frac{2^{d-n}}{n} \end{aligned}$$

Fast Exponentiation Mod n

The exponentiation (2^{d-n} mod n) in this formula can be evaluated very rapidly by means of the binary algorithm for exponentiation, performed modulo n:

Simple example problem: Calculate the 3^{17} mod 10.

Algorithm A: 3x3x3x3x3x3x3x3x3x3x3x3x3x3x3x3x3 = 129140163. Ans =3.

Algorithm B: $3^{17} = (((3^2)^2)^2)^2 \times 3$ = 129140163. Ans = 3.

Algorithm C:

3^{17} mod 10 = $((((3^2 \bmod 10)^2 \bmod 10)^2 \bmod 10)^2 \bmod 10) \times 3 \bmod 10 = 3$.

In detail: 3^2 mod 10 = 9; 9^2 mod 10 = 1; 1^2 mod 10 = 1; 1^2 mod 10 = 1; 1 x 3 = 3. Ans = 3.

Note that with Algorithm C, we never have to deal with integers > 81.

The BBP Formula for Pi

In 1996, Simon Plouffe used DHB's PSLQ program and high-precision software to discover this new formula for pi:

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

This formula permits one to compute binary (or hexadecimal) digits of pi beginning at an arbitrary starting position, using a very simple scheme that can run on any system with standard 64-bit or 128-bit arithmetic.

Recently it was proven that no base-n formulas of this type exist for pi, except $n = 2^m$.

1. D. H. Bailey, P. B. Borwein and S. Plouffe, "On the Rapid Computation of Various Polylogarithmic Constants," *Mathematics of Computation*, vol. 66, no. 218 (Apr 1997), pg. 903-913.
2. J. M. Borwein, W. F. Galway and D. Borwein, "Finding and Excluding b-ary Machin-Type BBP Formulae," *Canadian Journal of Mathematics*, vol. 56 (2004), pg 1339-1342.

Some Other BBP-Type Identities

$$\pi^2 = \frac{1}{8}\sum_{k=0}^{\infty}\frac{1}{64^k}\left(\frac{144}{(6k+1)^2}-\frac{216}{(6k+2)^2}-\frac{72}{(6k+3)^2}-\frac{54}{(6k+4)^2}+\frac{9}{(6k+5)^2}\right)$$

$$\pi^2 = \frac{2}{27}\sum_{k=0}^{\infty}\frac{1}{729^k}\left(\frac{243}{(12k+1)^2}-\frac{405}{(12k+2)^2}-\frac{81}{(12k+4)^2}-\frac{27}{(12k+5)^2}\right.$$
$$\left.-\frac{72}{(12k+6)^2}-\frac{9}{(12k+7)^2}-\frac{9}{(12k+8)^2}-\frac{5}{(12k+10)^2}+\frac{1}{(12k+11)^2}\right)$$

$$\zeta(3) = \frac{1}{1792}\sum_{k=0}^{\infty}\frac{1}{2^{12k}}\left(\frac{6144}{(24k+1)^3}-\frac{43008}{(24k+2)^3}+\frac{24576}{(24k+3)^3}+\frac{30720}{(24k+4)^3}\right.$$
$$-\frac{1536}{(24k+5)^3}+\frac{3072}{(24k+6)^3}+\frac{768}{(24k+7)^3}-\frac{3072}{(24k+9)^3}-\frac{2688}{(24k+10)^3}$$
$$-\frac{192}{(24k+11)^3}-\frac{1536}{(24k+12)^3}-\frac{96}{(24k+13)^3}-\frac{672}{(24k+14)^3}-\frac{384}{(24k+15)^3}$$
$$+\frac{24}{(24k+17)^3}+\frac{48}{(24k+18)^3}-\frac{12}{(24k+19)^3}+\frac{120}{(24k+20)^3}+\frac{48}{(24k+21)^3}$$
$$\left.-\frac{42}{(24k+22)^3}+\frac{3}{(24k+23)^3}\right)$$

$$\frac{25}{2}\log\left(\frac{781}{256}\left(\frac{57-5\sqrt{5}}{57+5\sqrt{5}}\right)^{\sqrt{5}}\right) = \sum_{k=0}^{\infty}\frac{1}{5^{5k}}\left(\frac{5}{5k+2}+\frac{1}{5k+3}\right)$$

Papers by D. H. Bailey, P. B. Borwein, S. Plouffe, D. Broadhurst and R. Crandall.

Normality of Mathematical Constants

A real number x is said to be b-normal (or normal base b) if every m-long string of base-b digits appears, in the limit, with frequency b^{-m}.

Whereas it can be shown that almost all real numbers are b-normal (for any b), there are only a handful of proven explicit examples.

It is still not known whether any of the following are b-normal for any b:

$$
\begin{aligned}
\sqrt{2} &= 1.4142135623730950488\ldots \\
\phi = \frac{\sqrt{5}-1}{2} &= 0.6180339887498948482\ldots \\
\pi &= 3.1415926535897932385\ldots \\
e &= 2.7182818284590452354\ldots \\
\log 2 &= 0.6931471805599453094\ldots \\
\log 10 &= 2.3025850929940456840\ldots \\
\zeta(2) &= 1.6449340668482264365\ldots \\
\zeta(3) &= 1.2020569031595942854\ldots
\end{aligned}
$$

A Connection Between BBP Formulas and Normality

Let {} denote fractional part. Consider the sequence defined by $x_0 = 0$,

$$x_n = \left\{ 2x_{n-1} + \frac{1}{n} \right\}$$

Result: log(2) is 2-normal if and only if this sequence is equidistributed in the unit interval.

In a similar vein, consider the sequence $x_0 = 0$, and

$$x_n = \left\{ 16x_{n-1} + \frac{120n^2 - 89n + 16}{512n^4 - 1024n^3 + 712n^2 - 206n + 21} \right\}$$

Result: pi is 16-normal if and only if this sequence is equidistributed in the unit interval.

A similar result holds for any constant that possesses a BBP-type formula.

D. H. Bailey and R. E. Crandall, "On the Random Character of Fundamental Constant Expansions," *Experimental Mathematics*, vol. 10, no. 2 (Jun 2001), pg. 175-190.

A Class of Provably Normal Constants

We have also shown that the following constant is 2-normal:

$$\begin{aligned}\alpha_{2,3} &= \sum_{k=1}^{\infty} \frac{1}{3^k 2^{3^k}} \\ &= 0.0418836808315029850712528986245716824260967\ldots_{10} \\ &= \mathtt{0.0AB8E38F684BDA12F684BF35BA781948B0FCD6E9E0}\ldots_{16}\end{aligned}$$

This was originally proven by Stoneham in 1970, but we have generalized this to an uncountably infinite class.

These results have led to a practical and efficient pseudo-random number generator based on the binary digits of alpha.

1. D. H. Bailey and R. E. Crandall, "Random Generators and Normal Numbers," *Experimental Mathematics*, vol. 11, no. 4 (2002), pg. 527-546.
2. D. H. Bailey, "A Pseudo-Random Number Generator Based on Normal Numbers," manuscript, Dec 2004, http://crd.lbl.gov/~dhbailey/dhbpapers/normal-random.pdf.

The "Hot Spot" Lemma for Proving Normality

We are now able to prove normality for these alpha constants very simply, by means of a new result that we call the "hot spot" lemma, proven using ergodic theory:

Hot Spot Lemma: Let {} denote fractional part. Then x is b-normal if and only if there is no y in [0,1) such that

$$\liminf_{m\to\infty} \limsup_{n\to\infty} \frac{\#_{0\le j<n}(|\{b^j x\} - y| < b^{-m})}{2nb^{-m}} = \infty$$

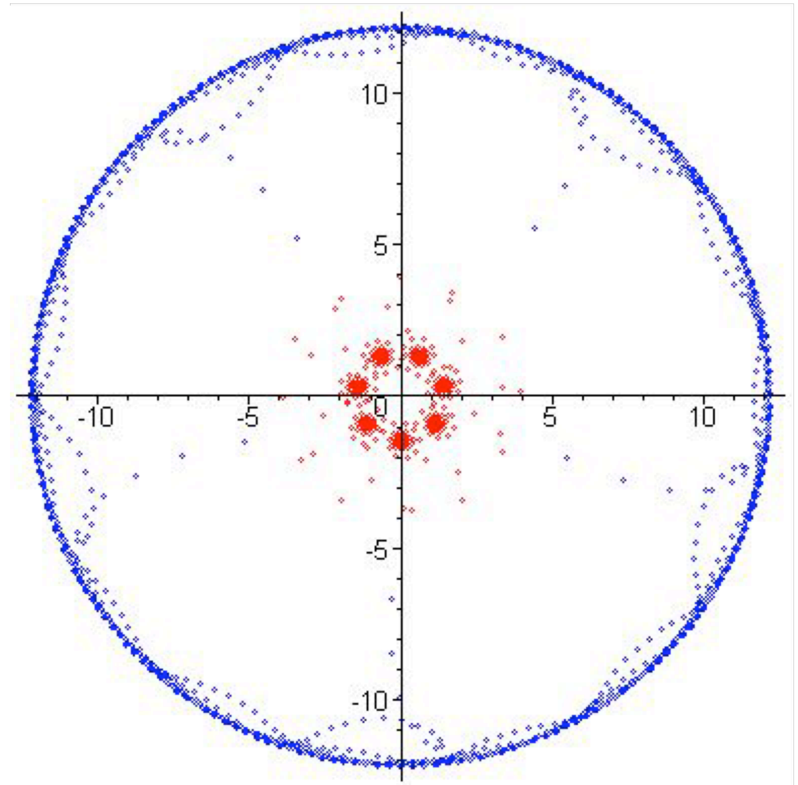
Paraphrase: x is b-normal if and only if it has no base-b hot spots.

Sample Corollary: If, for each m and n, no m-long string of digits appears in the first n digits of the base-2 expansion of x more often than 1,000 n 2^{-m} times, then x is 2-normal.

D. H. Bailey and M. Misiurewicz, "A Strong Hot Spot Theorem," *Proceedings of the American Mathematical Society*, vol. 134 (2006), no. 9, pg. 2495-2501.

Visualization in Experimental Math: Ramanujan Continued Fractions

- This is a plot of some iterates related to "Ramanujan continued fractions".
- With assistance of such plots, the behavior of these iterates (and the Ramanujan continued fractions themselves) are now quite well understood.
- These studies have ventured into matrix theory, real analysis and even the theory of martingales from probability theory.



$$R_\eta(a,b) = \cfrac{a}{\eta + \cfrac{b^2}{\eta + \cfrac{4a^2}{\eta + \cfrac{9b^2}{\eta + \ddots}}}}$$

Jonathan M. Borwein and David H. Bailey, "Experimental Mathematics: Examples, Methods and Implications", *Notices of the AMS*, May 2005, pg. 502-514 .

The Euler-Maclaurin Formula of Numerical Analysis

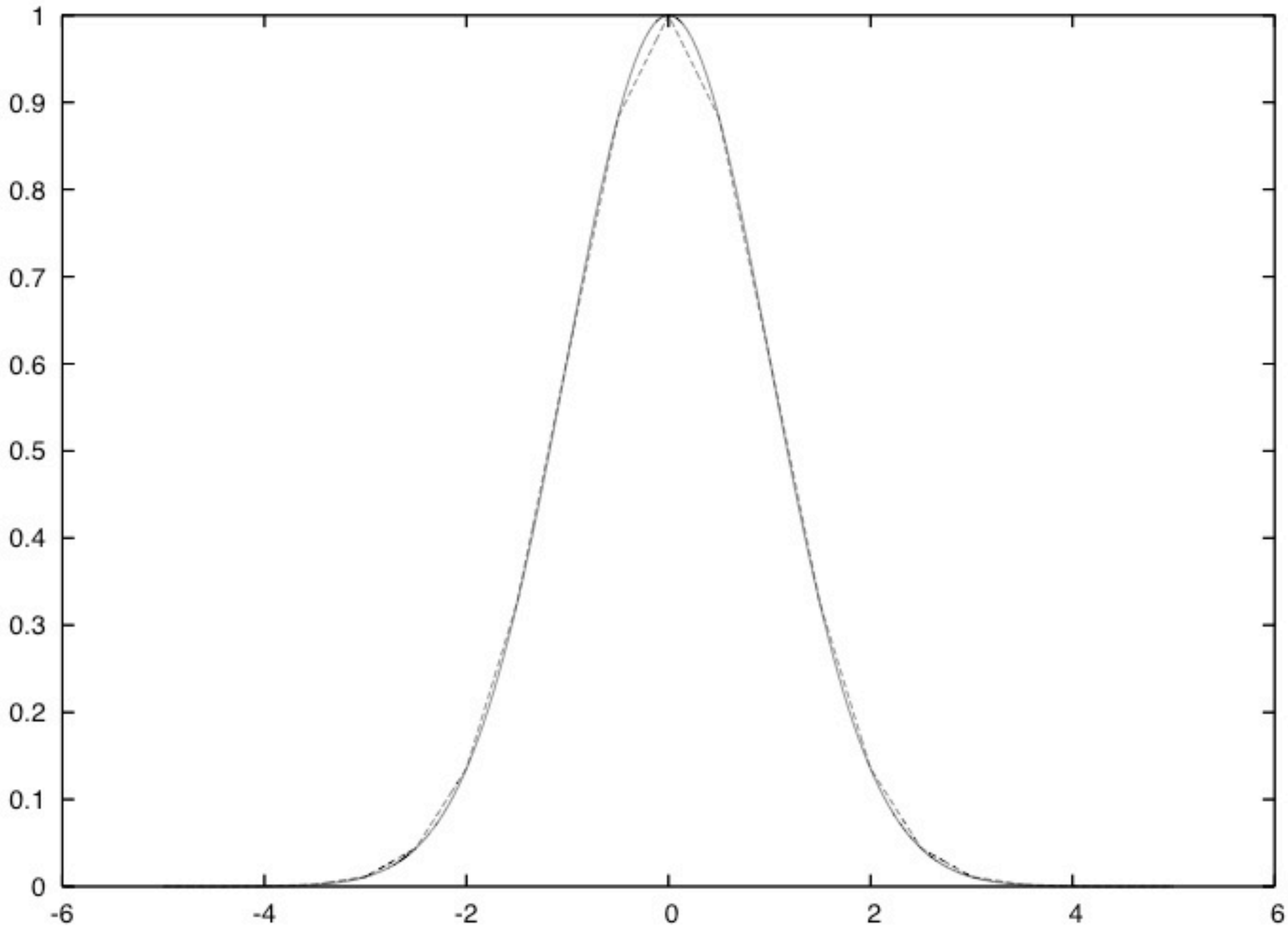
$$
\begin{aligned}
\int_a^b f(x)\,dx &= h \sum_{j=0}^{n} f(x_j) - \frac{h}{2}(f(a)+f(b)) \\
&\quad - \sum_{i=1}^{m} \frac{h^{2i}B_{2i}}{(2i)!}\left(f^{(2i-1)}(b) - f^{(2i-1)}(a)\right) - E(h) \\
|E(h)| &\le 2(b-a)[h/(2\pi)]^{2m+2} \max_{a\le x\le b} |D^{2m+2}f(x)|
\end{aligned}
$$

[Here h = (b - a)/n and x_j = a + j h. D^m f(x) means m-th derivative of f.]

Note when f(t) and all of its derivatives are zero at a and b (as in a bell-shaped curve), the error E(h) of a simple trapezoidal approximation to the integral goes to zero more rapidly than any power of h.

K. Atkinson, *An Introduction to Numerical Analysis*, John Wiley, 1989, pg. 289.

Trapezoidal Approximation to a Bell-Shaped Function



Tanh-Sinh Quadrature

Given f(x) defined on (-1,1), define g(t) = tanh (pi/2 sinh t). Then setting x = g(t) yields

$$\int_{-1}^{1} f(x)\,dx \;=\; \int_{-\infty}^{\infty} f(g(t))g'(t)\,dt \;\approx\; h\sum_{-N}^{N} w_j f(x_j)$$

where x_j = g(hj) and w_j = g'(hj). Since g'(t) goes to zero very rapidly for large t, the product f(g(t)) g'(t) typically is a nice bell-shaped function for which the E-M formula applies. Thus the simple summation above is remarkably accurate. Reducing h by half typically doubles the number of correct digits.

Tanh-sinh quadrature is the best integration scheme for functions with vertical derivatives or blow-up singularities at endpoints, or for any function at very high precision (> 1000 digits).

1. D. H. Bailey, X. S. Li and K. Jeyabalan, "A Comparison of Three High-Precision Quadrature Schemes," *Experimental Mathematics*, vol. 14 (2005), no. 3, pg. 317-329.
2. H. Takahasi and M. Mori, "Double Exponential Formulas for Numerical Integration," Publications of RIMS, Kyoto University, vol. 9 (1974), pg. 721–741.

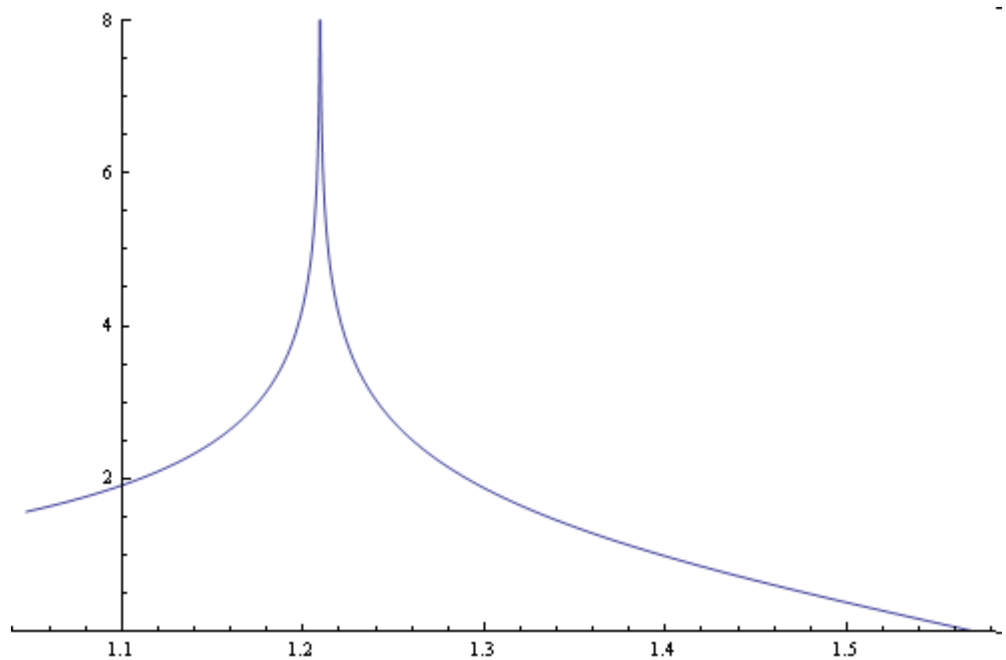
A Log-Tan Integral Identity from Mathematical Physics

$$\frac{24}{7\sqrt{7}} \int_{\pi/3}^{\pi/2} \log \left| \frac{\tan t + \sqrt{7}}{\tan t - \sqrt{7}} \right| dt \stackrel{?}{=} \sum_{n=0}^{\infty} \left[\frac{1}{(7n+1)^2} + \frac{1}{(7n+2)^2} - \frac{1}{(7n+3)^2} + \frac{1}{(7n+4)^2} - \frac{1}{(7n+5)^2} - \frac{1}{(7n+6)^2} \right]$$

This conjectured identity arises in mathematical physics from analysis of volumes of ideal tetrahedra in hyperbolic space.

We have verified this numerically to 20,000 digits using highly parallel tanh-sinh quadrature, but no formal proof is known.

D. H. Bailey, J. M. Borwein, V. Kapoor and E. Weisstein, “Ten Problems in Experimental Mathematics,” *American Mathematical Monthly*, vol. 113, no. 6 (Jun 2006), pg. 481-409 .



Parallel Evaluation of the log-tan Integral

$$\frac{24}{7\sqrt{7}} \int_{\pi/3}^{\pi/2} \log \left| \frac{\tan t + \sqrt{7}}{\tan t - \sqrt{7}} \right| dt = 1.15192547054449104710\ldots$$

CPUs	Init	Integral #1	Integral #2	Total	Speedup
1	190013	1534652	1026692	2751357	1.00
16	12266	101647	64720	178633	15.40
64	3022	24771	16586	44379	62.00
256	770	6333	4194	11297	243.55
1024	199	1536	1034	2769	993.63

1-CPU timings are sums of timings from a 64-CPU run, where barrier waits and communication were not timed.

The performance rate for the 1024-CPU run is 690 Gflop/s.

D. H. Bailey and J. M. Borwein, "Highly Parallel, High-Precision Numerical Integration," *International Journal of Computational Science and Engineering*, to appear, http://crd.lbl.gov/~dhbailey/dhbpapers/quadparallel.pdf.

Integrals from Ising Theory of Mathematical Physics

We recently applied our methods to study three classes of integrals that arise in the Ising theory of mathematical physics:

$$C_n := \frac{4}{n!} \int_0^\infty \cdots \int_0^\infty \frac{1}{\left(\sum_{j=1}^n (u_j + 1/u_j)\right)^2} \frac{du_1}{u_1} \cdots \frac{du_n}{u_n}$$

$$D_n := \frac{4}{n!} \int_0^\infty \cdots \int_0^\infty \frac{\prod_{i<j} \left(\frac{u_i - u_j}{u_i + u_j}\right)^2}{\left(\sum_{j=1}^n (u_j + 1/u_j)\right)^2} \frac{du_1}{u_1} \cdots \frac{du_n}{u_n}$$

$$E_n = 2 \int_0^1 \cdots \int_0^1 \left(\prod_{1 \le j < k \le n} \frac{u_k - u_j}{u_k + u_j}\right)^2 dt_2\, dt_3 \cdots dt_n,$$

where (in the last line) $$u_k = \prod_{i=1}^k t_i$$

D. H. Bailey, J. M. Borwein and R. E. Crandall, "Integrals of the Ising Class," *Journal of Physics A: Mathematical and General*, vol. 39 (2006), pg. 12271-12302.

Computing and Evaluating C_n

We first showed that the multi-dimensional C_n integrals can be transformed to much more manageable 1-D integrals:

$$C_n = \frac{2^n}{n!}\int_0^\infty t K_0^n(t)\ dt$$

where K_0 is the modified Bessel function.

We used this formula to compute 1000-digit numerical values of various C_n, from which the following results and others were found, then proven:

$$\begin{aligned} C_1 &= 2 \\ C_2 &= 1 \\ C_3 &= \mathrm{L}_{-3}(2) = \sum_{n\ge 0}\left(\frac{1}{(3n+1)^2} - \frac{1}{(3n+2)^2}\right) \\ C_4 &= 14\zeta(3) \end{aligned}$$

Limiting Value of C_n

The C_n numerical values approach a limit:

$$\begin{aligned} C_{10} &= 0.63188002414701222229035087366080283... \\ C_{40} &= 0.63047350337836353186994190185909694... \\ C_{100} &= 0.63047350337438679612204019271903171... \\ C_{200} &= 0.63047350337438679612204019271087890... \end{aligned}$$

What is this limit? We copied the first 50 digits of this numerical value into the online Inverse Symbolic Calculator (ISC), now available at

http://ddrive.cs.dal.ca/~isc

The result was:

$$\lim_{n\to\infty} C_n = 2e^{-2\gamma}$$

where gamma denotes Euler's constant. This result is now proven and has been generalized to an asymptotic expansion.

Other Ising Integral Evaluations

$$
\begin{aligned}
D_2 &= 1/3 \\
D_3 &= 8 + 4\pi^2/3 - 27\,\mathrm{L}_{-3}(2) \\
D_4 &= 4\pi^2/9 - 1/6 - 7\zeta(3)/2 \\
E_2 &= 6 - 8\log 2 \\
E_3 &= 10 - 2\pi^2 - 8\log 2 + 32\log^2 2 \\
E_4 &= 22 - 82\zeta(3) - 24\log 2 + 176\log^2 2 - 256(\log^3 2)/3 \\
&\quad + 16\pi^2\log 2 - 22\pi^2/3 \\
E_5 &\stackrel{?}{=} 42 - 1984\,\mathrm{Li}_4(1/2) + 189\pi^4/10 - 74\zeta(3) - 1272\zeta(3)\log 2 \\
&\quad + 40\pi^2\log^2 2 - 62\pi^2/3 + 40(\pi^2\log 2)/3 + 88\log^4 2 \\
&\quad + 464\log^2 2 - 40\log 2
\end{aligned}
$$

The Ising Integral E_5

We were able to reduce E_5, which is a 5-D integral, to an extremely complicated 3-D integral (see below).

We computed this 3-D integral to 250-digit precision, using a parallel high-precision 3-D quadrature program. Then we used PSLQ to discover the evaluation given on the previous page.

$$
\begin{aligned}
E_5 = {} & \int_0^1\int_0^1\int_0^1 [2(1-x)^2(1-y)^2(1-xy)^2(1-z)^2(1-yz)^2(1-xyz)^2 (-[4(x+1)(xy+1)\log(2) (y^5z^3x^7 - y^4z^2(4(y+1)z+3)x^6 - y^3z ((y^2+1) z^2 + 4(y+ \\
& 1)z+5) x^5 + y^2 (4y(y+1)z^3 + 3 (y^2+1) z^2 + 4(y+1)z - 1) x^4 + y (z (z^2+4z+5) y^2 + 4 (z^2+1) y + 5z + 4) x^3 + ((-3z^2-4z+1) y^2 - 4zy + 1) x^2 \\
& -(y(5z+4)+4)x - 1)] / [(x-1)^3(xy-1)^3(xyz-1)^3] + [3(y-1)^2y^4(z-1)^2z^2(yz-1)^2x^6 + 2y^3z (3(z-1)^2z^3y^5 + z^2 (5z^3+3z^2+3z+5) y^4 + (z-1)^2z \\
& (5z^2+16z+5) y^3 + (3z^5+3z^4-22z^3-22z^2+3z+3) y^2 + 3 (-2z^4+z^3+2z^2+z-2) y + 3z^3 + 5z^2 + 5z + 3) x^5 + y^2 (7(z-1)^2z^4y^6 - 2z^3 (z^3+15z^2 \\
& +15z+1) y^5 + 2z^2 (-21z^4+6z^3+14z^2+6z-21) y^4 - 2z (z^5-6z^4-27z^3-27z^2-6z+1) y^3 + (7z^6-30z^5+28z^4+54z^3+28z^2-30z+7) y^2 - 2 (7z^5 \\
& +15z^4-6z^3-6z^2+15z+7) y + 7z^4 - 2z^3 - 42z^2 - 2z + 7) x^4 - 2y (z^3 (z^3-9z^2-9z+1) y^6 + z^2 (7z^4-14z^3-18z^2-14z+7) y^5 + z (7z^5+14z^4+3 \\
& z^3+3z^2+14z+7) y^4 + (z^6-14z^5+3z^4+84z^3+3z^2-14z+1) y^3 - 3 (3z^5+6z^4-z^3-z^2+6z+3) y^2 - (9z^4+14z^3-14z^2+14z+9) y + z^3 + 7z^2 + 7z \\
& +1) x^3 + (z^2 (11z^4+6z^3-66z^2+6z+11) y^6 + 2z (5z^5+13z^4-2z^3-2z^2+13z+5) y^5 + (11z^6+26z^5+44z^4-66z^3+44z^2+26z+11) y^4 + (6z^5-4 \\
& z^4-66z^3-66z^2-4z+6) y^3 - 2 (33z^4+2z^3-22z^2+2z+33) y^2 + (6z^3+26z^2+26z+6) y + 11z^2 + 10z + 11) x^2 - 2 (z^2 (5z^3+3z^2+3z+5) y^5 + z (22z^4 \\
& +5z^3-22z^2+5z+22) y^4 + (5z^5+5z^4-26z^3-26z^2+5z+5) y^3 + (3z^4-22z^3-26z^2-22z+3) y^2 + (3z^3+5z^2+5z+3) y + 5z^2 + 22z + 5) x + 15z^2 + 2z \\
& +2y(z-1)^2(z+1) + 2y^3(z-1)^2z(z+1) + y^4z^2 (15z^2+2z+15) + y^2 (15z^4-2z^3-90z^2-2z+15) + 15] / [(x-1)^2(y-1)^2(xy-1)^2(z-1)^2(yz-1)^2 \\
& (xyz-1)^2] - [4(x+1)(y+1)(yz+1) (-z^2y^4 + 4z(z+1)y^3 + (z^2+1) y^2 - 4(z+1)y + 4x (y^2-1) (y^2z^2-1) + x^2 (z^2y^4 - 4z(z+1)y^3 - (z^2+1) y^2 \\
& +4(z+1)y+1) - 1) \log(x+1)] / [(x-1)^3x(y-1)^3(yz-1)^3] - [4(y+1)(xy+1)(z+1) (x^2 (z^2-4z-1) y^4 + 4x(x+1) (z^2-1) y^3 - (x^2+1) (z^2-4z-1) \\
& y^2 - 4(x+1) (z^2-1) y + z^2 - 4z - 1) \log(xy+1)] / [x(y-1)^3y(xy-1)^3(z-1)^3] - [4(z+1)(yz+1) (x^3y^5z^7 + x^2y^4(4x(y+1)+5)z^6 - xy^3 ((y^2+ \\
& 1) x^2 - 4(y+1)x - 3) z^5 - y^2 (4y(y+1)x^3 + 5 (y^2+1) x^2 + 4(y+1)x + 1) z^4 + y (y^2x^3 - 4y(y+1)x^2 - 3 (y^2+1) x - 4(y+1)) z^3 + (5x^2y^2 + y^2 + 4x(y+1) \\
& y+1) z^2 + ((3x+4)y+4)z - 1) \log(xyz+1)] / [xy(z-1)^3z(yz-1)^3(xyz-1)^3])] / [(x+1)^2(y+1)^2(xy+1)^2(z+1)^2(yz+1)^2(xyz+1)^2] \\
& dx\, dy\, dz
\end{aligned}
$$

Recursions in Ising Integrals

Consider the 2-parameter class of Ising integrals

$$C_{n,k} = \frac{4}{n!} \int_0^\infty \cdots \int_0^\infty \frac{1}{\left(\sum_{j=1}^n (u_j + 1/u_j)\right)^{k+1}} \frac{du_1}{u_1} \cdots \frac{du_n}{u_n}$$

(which have connections to quantum field theory). After computing 1000-digit numerical values for all n <= 36 and all k <= 75 (2660 individual quadrature calculations, performed in parallel), and applying PSLQ, we found linear relations in the rows of this array. For example, when n = 3:

$$\begin{aligned}
0 &= C_{3,0} - 84C_{3,2} + 216C_{3,4} \\
0 &= 2C_{3,1} - 69C_{3,3} + 135C_{3,5} \\
0 &= C_{3,2} - 24C_{3,4} + 40C_{3,6} \\
0 &= 32C_{3,3} - 630C_{3,5} + 945C_{3,7} \\
0 &= 125C_{3,4} - 2172C_{3,6} + 3024C_{3,8}
\end{aligned}$$

These recursions have been proven for n = 1, 2, 3, 4. Similar, but more complicated, recursions have been found for larger n (see next page).

D. H. Bailey, D. Borwein, J. M. Borwein and R. E. Crandall, "Hypergeometric Forms for Ising-Class Integrals," *Experimental Mathematics*, to appear, http://crd.lbl.gov/~dhbailey/dhbpapers/meijer/pdf.

Experimental Recursion for n = 24

$$
\begin{aligned}
0 \stackrel{?}{=}\ & C_{24,1} \\
& -1107296298\, C_{24,3} \\
& +1288574336175660\, C_{24,5} \\
& -88962910652291256000\, C_{24,7} \\
& +1211528914846561331193600\, C_{24,9} \\
& -5367185923241422152980553600\, C_{24,11} \\
& +9857686103738772925980190636800\, C_{24,13} \\
& -8476778037073141951236532459008000\, C_{24,15} \\
& +3590120926882411593645052529049600000\, C_{24,17} \\
& -745759114781380983188217871663104000000\, C_{24,19} \\
& +71215552121869985477578381170258739200000\, C_{24,21} \\
& -2649853457247995406113355087174696960000000\, C_{24,23} \\
& +24912519234220575094208313195233280000000000\, C_{24,25}
\end{aligned}
$$

Jonathan Borwein and Bruno Salvy have now given an explicit form for these recursions, together with code to compute any desired case.

J. M. Borwein and B. Salvy, “A Proof of a Recursion for Bessel Moments,” manuscript, 2007, http://users.cs.dal.ca/~jborwein/recursion.pdf.

Some New Ising Results (Nov 2007)

$$
\begin{aligned}
c_{3,0} &= \frac{3\Gamma^6(1/3)}{32\pi 2^{2/3}} = \frac{\sqrt{3}\pi^3}{8} {}_3F_2\left(\begin{array}{c} 1/2, 1/2, 1/2 \\ 1, 1 \end{array} \middle| 1/4 \right) \\
c_{3,2} &= \frac{\sqrt{3}\pi^3}{8} {}_3F_2\left(\begin{array}{c} 1/2, 1/2, 1/2 \\ 2, 2 \end{array} \middle| 1/4 \right) \\
c_{4,0} &= \frac{\pi^4}{4} \sum_{n=0}^{\infty} \frac{\binom{2n}{n}^4}{4^{4n}} = \frac{\pi^4}{4} {}_4F_3\left(\begin{array}{c} 1/2, 1/2, 1/2, 1/2 \\ 1, 1, 1 \end{array} \middle| 1 \right) \\
c_{4,2} &= \frac{\pi^4}{64} \left[4 {}_4F_3\left(\begin{array}{c} 1/2, 1/2, 1/2, 1/2 \\ 1, 1, 1 \end{array} \middle| 1 \right) \right. \\
&\quad \left. - 3 {}_4F_3\left(\begin{array}{c} 1/2, 1/2, 1/2, 1/2 \\ 2, 1, 1 \end{array} \middle| 1 \right) \right] - \frac{3\pi^2}{16}
\end{aligned}
$$

D. H. Bailey, J. M. Borwein, D. Broadhurst and M. L. Glasser, “Elliptic Integral Evaluations of Bessel Moments,” manuscript, 2007.

Summary

- A wave of new results have been discovered by applying high-performance computing technology in novel ways to problems in mathematics and physics.
- Continued rapid progress is very likely, due both to Moore's Law and also to an influx of young researchers highly skilled in computing.

Questions:

- Can we better understand the theoretical underpinnings of these computational methods?
- Can we develop better methods for tasks such as infinite series summation and multi-dimensional quadrature?
- Can we adapt computational math algorithms and software to extremely highly parallel computer systems?
- Can we better incorporate other recent developments in computer science – advanced visualization, database management, compiler and language tools, etc?
- How can we best train students in these methods?

This talk is available at:

`http://crd.lbl.gov/~dhbailey/dhbtalks/dhb-expmath-hpc.pdf`