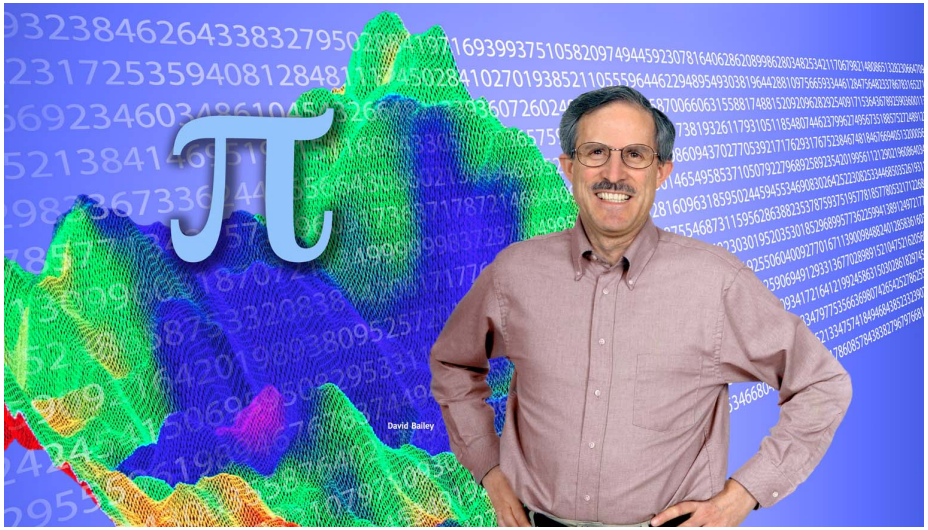
High-Precision Computation and Reproducibility

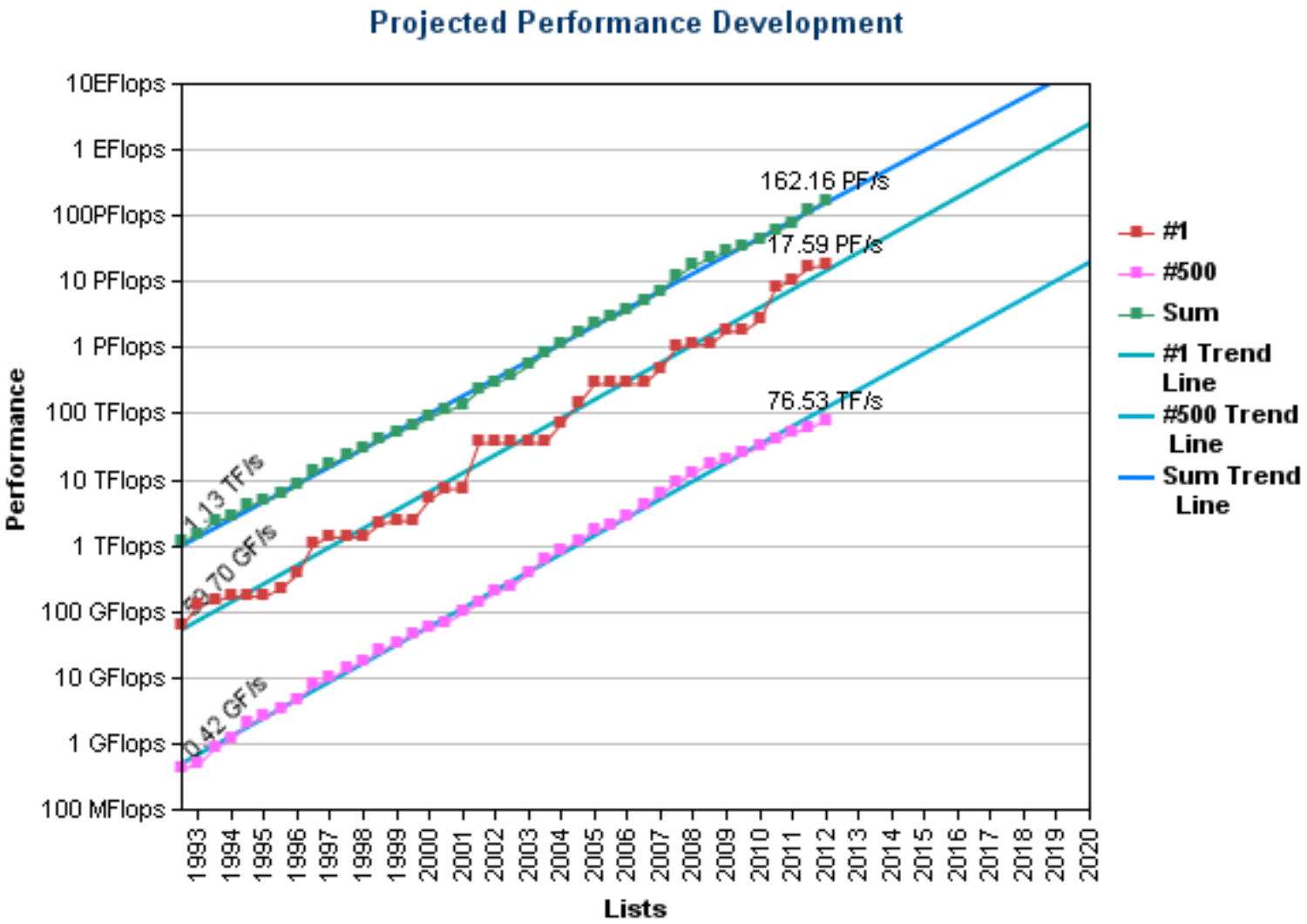
David H Bailey, Lawrence Berkeley National Lab, USA

This talk is available at:

http://www.davidhbailey.com/dhbtalks/dhb-icerm.pdf



Progress of scientific supercomputers: Data from the Nov 2012 Top500 list



Large-scale parallel computing and numerical reproducibility

- As numerical computations are ported from single-processor systems to large-scale, highly parallel supercomputers, problems are typically scaled up by factors of millions.
- As a result, computations that previously had satisfactory numerical behavior now may be highly ill-conditioned, and results are reproducible to fewer digits (or none at all).
- Computational scientists who develop codes are, in most cases, not experts in numerical analysis and may not realize the potential difficulties.

Example: Colleagues at LBNL who work with a large code for analyzing Large Hadron Collider data recently reported that when they merely changed the floating-point library (for transcendental functions, etc.), they observed cases where particle events no longer occurred.

- Do their results have any numerical significance at all in such cases?

Large-scale parallel computing and numerical reproducibility

- Porting a code to a parallel computer inevitably destroys any specified order of operations, particularly for global summations. As a result, digit-for-digit reproducibility cannot be guaranteed in most cases.
- Even after the port to the parallel system, the order of operation changes when the number of processors used is changed, as is very often done by programmers dealing with batch queues.

One potential solution:

Perform global summations using double-double arithmetic, then reduce back to double precision for final results.

Aren't 64 bits enough?

Almost all scientific computers (from PCs to supercomputers) now feature IEEE-754 64-bit floating-point arithmetic. However, for a growing body of numerical algorithms and applications, 64 bits aren't enough:

- Ill-conditioned linear systems.
- Large summations, with cancellations.
- Long-time, iterative simulations.
- Large-scale simulations.
- Resolving small-scale phenomena.
- Studies in experimental mathematics – hundreds or thousands of digits.

Using high-precision arithmetic is often the easiest way to solve numerical problems, even more sophisticated algorithmic solutions are possible.

The fact that high-precision arithmetic is not only useful, but is in fact essential for some important applications, has encountered surprisingly strong resistance from several quarters.

Innocuous-looking example where standard precision is inadequate

Find a polynomial to fit the data (1, 1048579, 16777489, 84941299, 268501249, 655751251, 1360635409, 2523398179, 4311748609) for arguments 0, 1, …, 8.

The usual approach is to solve the linear system:

$$\begin{bmatrix} n & \sum_{k=1}^{n} x_k & \cdots & \sum_{k=1}^{n} x_k^n \\ \sum_{k=1}^{n} x_k & \sum_{k=1}^{n} x_k^2 & \cdots & \sum_{k=1}^{n} x_k^{n+1} \\ \vdots & \vdots & \ddots & \vdots \\ \sum_{k=1}^{n} x_k^n & \sum_{k=1} x_k^{n+1} & \cdots & \sum_{k=1}^{n} x_k^{2n} \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{bmatrix} = \begin{bmatrix} \sum_{k=1}^{n} y_k \\ \sum_{k=1}^{n} x_k y_k \\ \vdots \\ \sum_{k=1}^{n} x_k^n y_k \end{bmatrix}$$

using Matlab, Linpack or LAPACK. However, computation with 64-bit floating-point arithmetic fails to find the correct result in this instance.

However, if the Linpack routines are converted to use double-double arithmetic (31-digit accuracy), the above computation quickly produces the correct polynomial:

$$f(x) = 1 + 1048577x^4 + x^8 \;=\; 1 + (2^{20} + 1)x^4 + x^8$$

Algorithm changes versus double-double: Double-double is the pragmatic choice

- The result on the previous page can be obtained with double precision using Lagrange interpolation or the Demmel-Koev algorithm.
- But few computational scientists, outside of expert numerical analysts, are aware of these schemes – most people use Linpack or home-grown code.
- Besides, even these schemes fail for higher-degree problems, for example:
  - (1, 134217731, 8589938753, 97845255883, 549772595201, 2097396156251, 6264239146561, 15804422886323, 35253091827713, 71611233653971, 135217729000001, 240913322581691, 409688091758593).
  - This is generated by:

$$f(x) = 1 + 134217729x^6 + x^{12} \;=\; 1 + (2^{27} + 1)x^6 + x^{12}$$

In contrast, a straightforward Linpack scheme, implemented with double-double arithmetic, works fine for this and a wide range of similar problems.

Which is a more practical solution to a numerical anomaly?
(a) modify an existing code to use double-double, or
(b) implement a new scheme from scratch?
With new easy-to-use double-double software, choice (a) is much preferable.

Free software for high-precision computation

- ARPREC. Arbitrary precision, with many algebraic and transcendental functions. High-level interfaces for C++ and Fortran-90 make code conversion easy. Available at http://crd.lbl.gov/~dhbailey/mpdist.
- GMP. Produced by a volunteer effort and is distributed under the GNU license by the Free Software Foundation. Available at http://gmplib.org.
- MPFR. C library for multiple-precision floating-point computations with exact rounding, based on GMP. Available at http://www.mpfr.org.
- MPFR++. High-level C++ interface to MPFR. Available at http://perso.ens-lyon.fr/nathalie.revol/software.html.
- GMPFRXX. Similar to MPFR++. Available at http://math.berkeley.edu/~wilken/code/gmpfrxx.
- MPFUN90. Similar to ARPREC, but is written entirely in Fortran-90 and provides only a Fortran-90 interface. Available at http://crd.lbl.gov/~dhbailey/mpdist.
- QD. This package perform "double-double" (approx. 31 digits) and "quad-double" (approx. 62 digits) arithmetic. C++ and Fortran-90 high-level interfaces makes code conversion easy. Available at http://crd.lbl.gov/~dhbailey/mpdist.

All of these packages greatly increase run time – from ~5X for double-double to ~1000X for arbitrary precision with 1000 digits.

Berkeley's CORVETTE project

- CORVETTE: Correctness Verification and Testing of Parallel Programs
- Tools to find bugs in hybrid (conventional/GPU) and large-scale systems.
- One key component: numerical reliability
  - Tools to easily test the level of numerical accuracy of an application.
  - Tools to delimit the portions of code that are the principal sources of inaccuracy.
  - Tools to ameliorate numerical difficulties when they are uncovered, including usage of double-double or higher precision arithmetic.
  - Tools to navigate through a hierarchy of precision levels (half, double, double-double, higher).

Some applications where high-precision arithmetic is essential

- Planetary orbit calculations (32 digits).
- Supernova simulations (32-64 digits).
- Climate modeling (32 digits).
- Coulomb *n*-body atomic system simulations (32-120 digits).
- Schrodinger solutions for lithium and helium atoms (32 digits).
- Electromagnetic scattering theory (32-100 digits).
- Scattering amplitudes of quarks, gluons and bosons (32 digits).
- Discrete dynamical systems (32 digits).
- Theory of nonlinear oscillators (64 digits).
- Detecting “strange” nonchaotic attractors (32 digits).
- The Taylor algorithm for ordinary differential equations (100-550 digits).
- Ising integrals from mathematical physics (100-1000 digits).
- Other examples from experimental mathematics (100-20,000 digits).

For details and references, see:

David H. Bailey, Roberto Barrio, and Jonathan M. Borwein, "High precision computation: Mathematical physics and dynamics," *Applied Mathematics and Computation*, vol. 218 (2012), pg. 10106-10121.

http://www.davidhbailey.com/dhbpapers/hpmpd.pdf

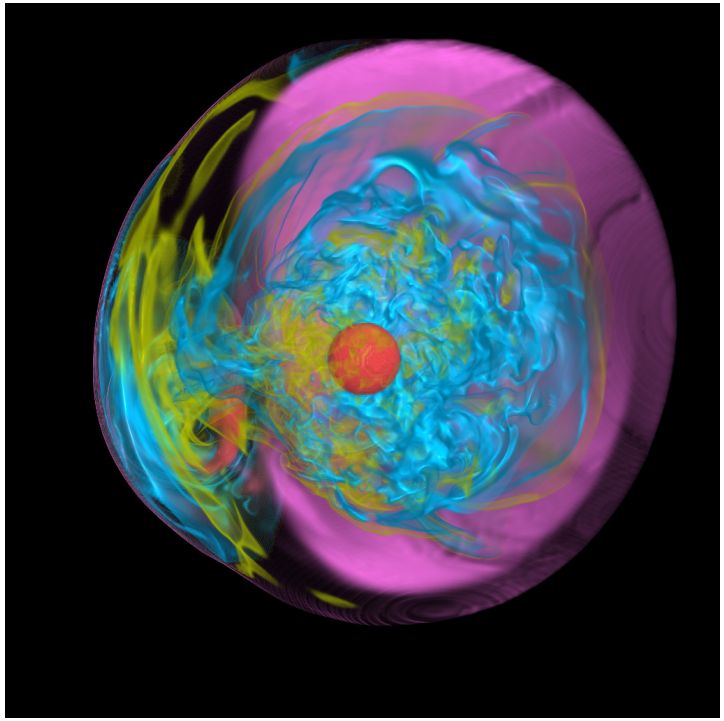
Long-term planetary orbits

- Researchers have recognized for centuries that planetary orbits exhibit chaotic behavior:
  - "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, *Principia*, 1687]
- Long-term simulations of planetary orbits using double precision do fairly well for long periods, but then fail at certain key junctures.
- Researchers have found that double-double or quad-double arithmetic is required to avoid severe inaccuracies, even if other techniques are employed to reduce numerical error.

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," *Proc. of the 8th ACM-SIAM Symp. on Discrete Algorithms*, 1997, 1-10.

Supernova simulations

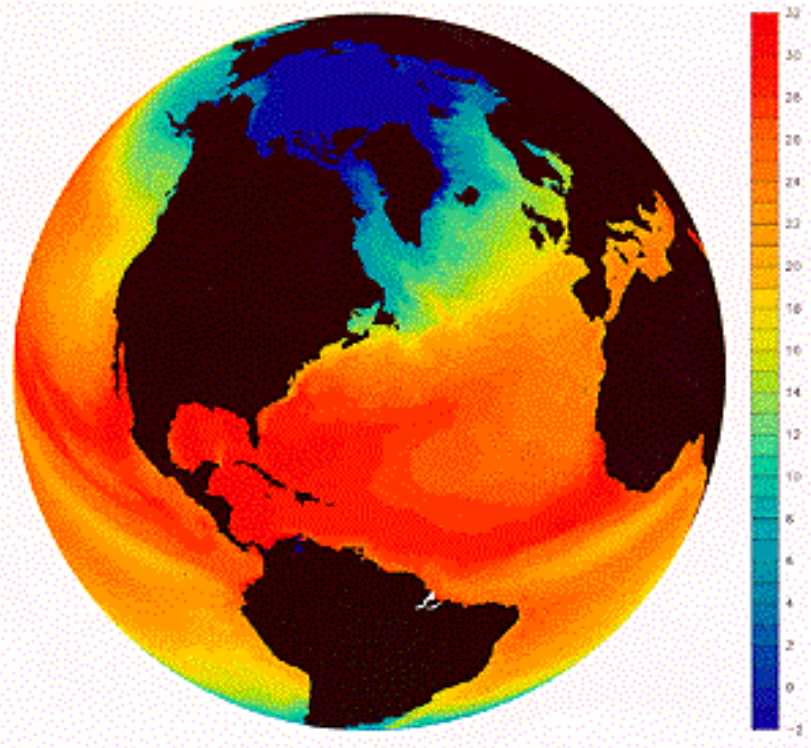
- Researchers at LBNL have used quad-double arithmetic to solve for non-local thermodynamic equilibrium populations of iron and other atoms in the atmospheres of supernovas.
- Iron may exist in several species, so it is necessary to solve for all species simultaneously.
- Since the relative population of any state from the dominant state is proportional to the exponential of the ionization energy, the dynamic range of these values can be very large.
- The quad-double 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.

Climate modeling: high-precision arithmetic for reproducibility

- Climate and weather simulations are fundamentally chaotic – if microscopic changes are made to the current state, soon the future state is quite different.
- In practice, computational results are altered even if minor changes are made to the code or the system.
- This numerical variation is a major nuisance for code maintenance.
- He and Ding of LBNL found that by using double-double arithmetic to implement a key inner product loop, most of this numerical variation disappeared.



Y. He and C. Ding, "Using Accurate Arithmetics to Improve Numerical Reproducibility and Stability in Parallel Applications," *Journal of Supercomputing*, vol. 18, no. 3 (Mar 2001), pg. 259-277.

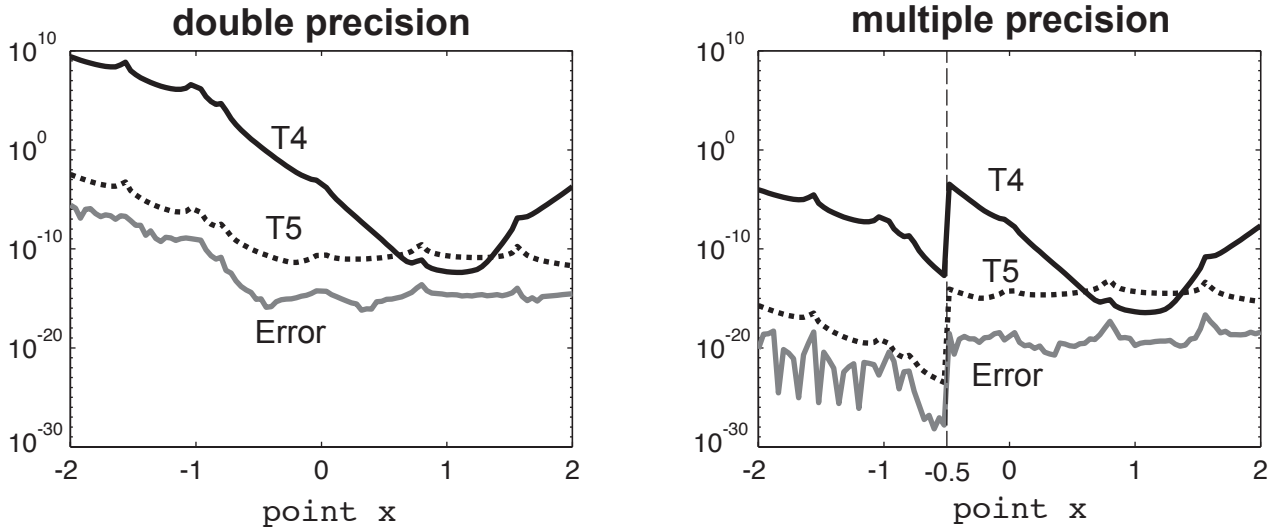
Coulomb n-body atomic system simulations

- Alexei Frolov of Queen's University in Canada has used high-precision arithmetic to solve a generalized eigenvalue problem that arises in Coulomb n-body interactions.
- Matrices are typically 5,000 x 5,000 and are very nearly singular.
- Computations typically involve massive cancellation, and high-precision arithmetic must be employed to obtain numerically reproducible results.
- Frolov has also computed elements of the Hamiltonian matrix and the overlap matrix in four- and five-body systems.
- These computations typically require 120-digit arithmetic.

"We can consider and solve the bound state few-body problems which have been beyond our imagination even four years ago." – Frolov

A. M. Frolov and DHB, "Highly Accurate Evaluation of the Few-Body Auxiliary Functions and Four-Body Integrals," *Journal of Physics B*, vol. 36, no. 9 (14 May 2003), pg. 1857-1867.

Error bounds in Chebyshev-Sobolev: double vs high precision



Behavior of the theoretical error bounds (T4 a backward error bound and T5 for the running error bound) and the relative error in the double- and higher-precision evaluation of the Chebyshev-Sobolev approximation of degree 50 of the function $f(x) = (x+1)^2 \sin(4x)$, where the discrete Sobolev measure have one mass point (c = 1) up to first derivative in the discrete part of the inner product.

R. Barrio and S. Serrano, “Generation and evaluation of orthogonal polynomials in discrete Sobolev spaces II. Numerical stability, *J. Comput. Appl. Math.*, vol. 181 (2005), pg. 299-320.

Taylor's method for the solution of ODEs

Taylor's method is one of the oldest numerical schemes for solving ODEs, but in recent years has re-emerged as the method of choice in the computational dynamics community because of speed to convergence.

Consider the initial value problem y' = f(t, y). The solution at time t = t_i is:

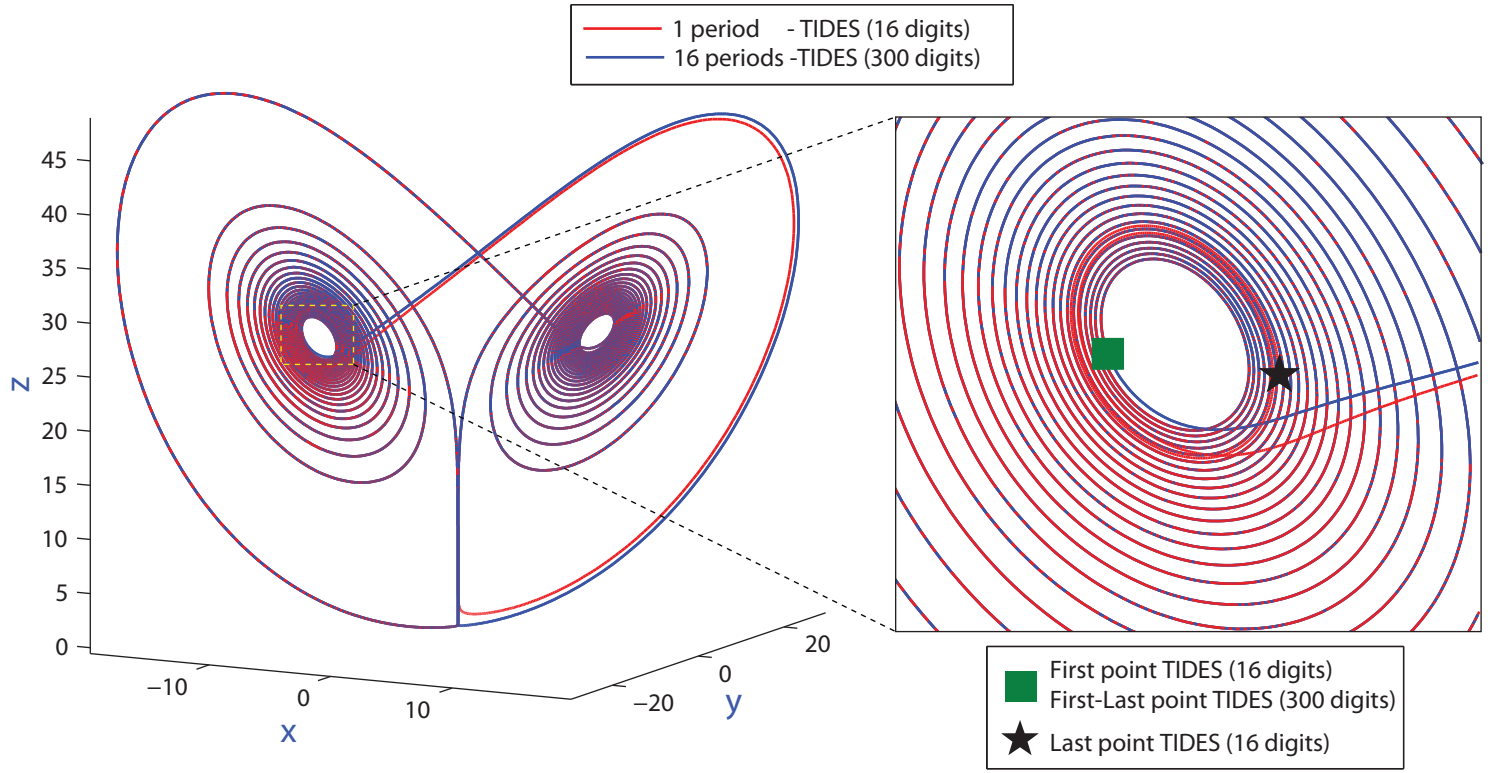
$$
\begin{aligned}
y(t_0) &=: y_0, \\
y(t_i) &\simeq y_{i-1} + f(t_{i-1}, y_{i-1})\, h_i + \cdots + \frac{1}{n!} \frac{d^{n-1} f(t_{i-1}, y_{i-1})}{dt^{n-1}} h_i^n \\
&=: y_i
\end{aligned}
$$

The Taylor coefficients here may be found using automatic differentiation methods.

One significant advantage with Taylor's method is that it can be easily implemented using high-precision arithmetic. When this is done, Taylor's method typically gives superior results, compared with other available schemes.

A. Abad, R. Barrio, F. Blesa and M. Rodriguez, "TIDES: a Taylor series Integrator for Differential EquationS," preprint, 2010.

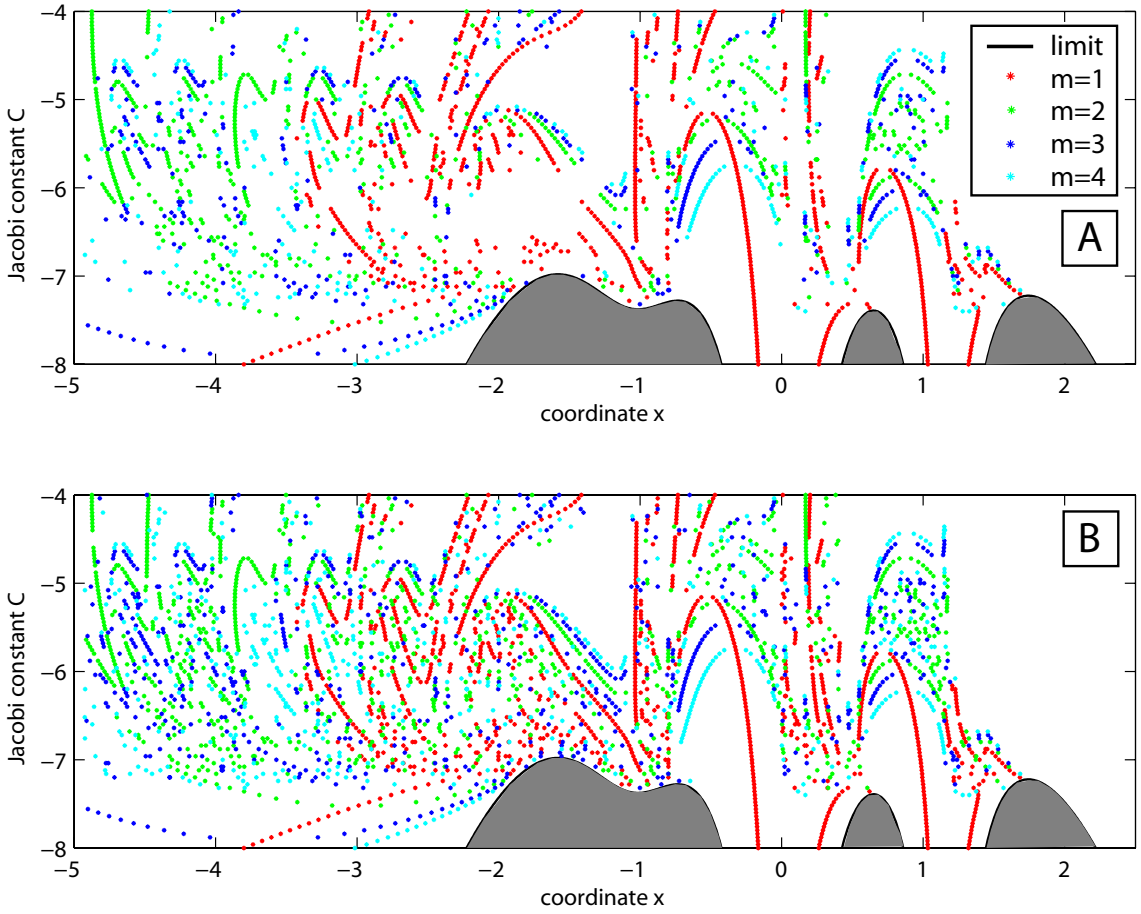
Taylor's method with high-precision arithmetic



Numerical integration of the L25-R25 unstable periodic orbit for the Lorenz model during 16 time periods using the TIDES code with 300 digits, and 1 time periods using double precision.

DHB, R. Barrio and J. M. Borwein, "High precision computation: Mathematical physics and dynamics," *Applied Mathematics and Computation*, vol. 218 (2012), pg. 10106-10121.

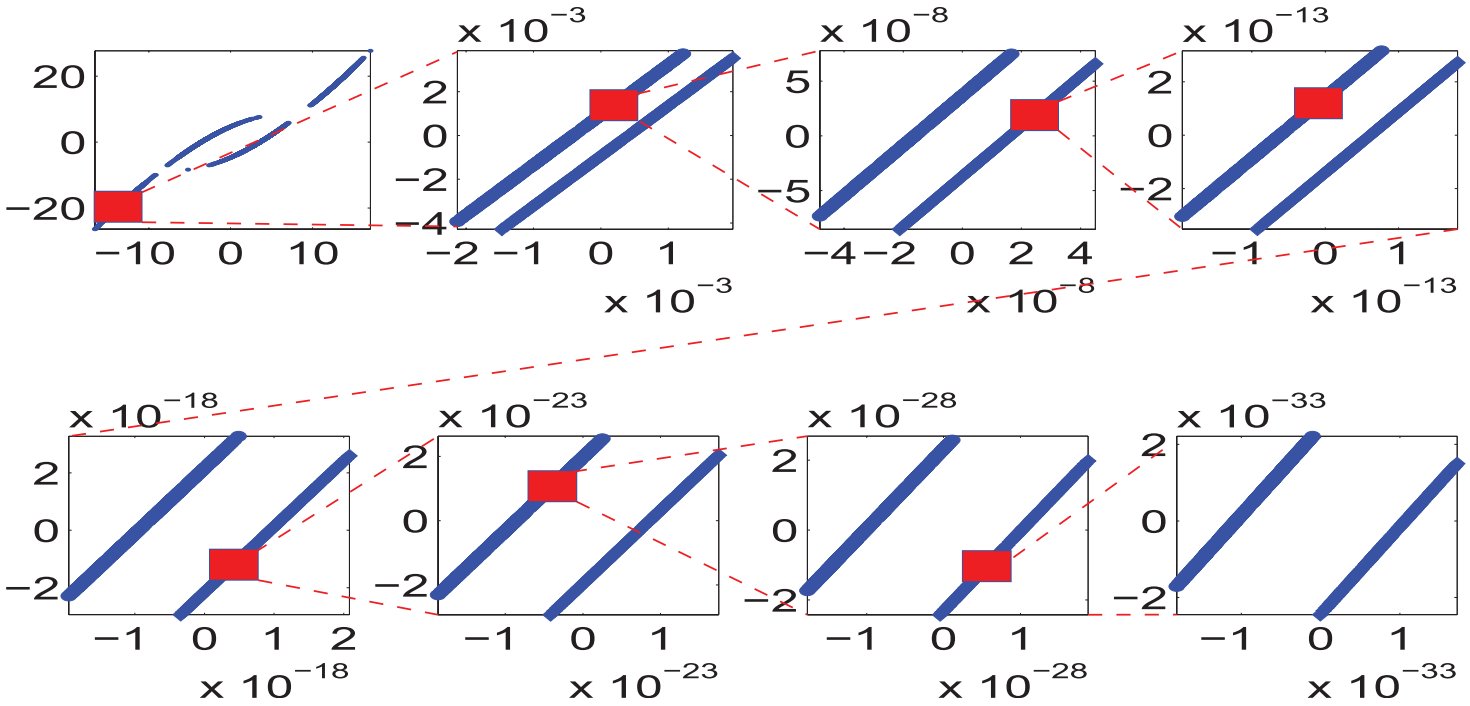
Computing the "skeleton" of periodic orbits



Symmetric periodic orbits (m denotes the multiplicity of the periodic orbit) in the most chaotic zone of the (7+2) ring problem using double (A) and quadruple (B) precision.

R. Barrio and F. Blesa, "Systematic search of symmetric periodic orbits in 2DOF Hamiltonian systems," *Chaos, Solitons and Fractals*, vol. 41 (2009), 560-582.

Fractal properties of Lorenz attractors: using high-precision to "zoom in"



On the first plot, the intersection of an arbitrary trajectory on the Lorenz attractor with the section z = 27. The plot shows a rectangle in the x-y plane. All later plots zoom in on a tiny region (too small to be seen by the unaided eye) at the center of the red rectangle of the preceding plot to show that what appears to be a line is in fact not a line. Very high precision (hundreds of digits) arithmetic is required for these results.

1. D. Viswanath, "The fractal property of the Lorenz attractor," *Journal of Physics D*, vol. 190 (2004), 115-128.
2. D. Viswanath and S. Sahutoglu, "Complex singularities and the Lorenz attractor," *SIAM Review*, to appear.

Lions-Mercer iterations

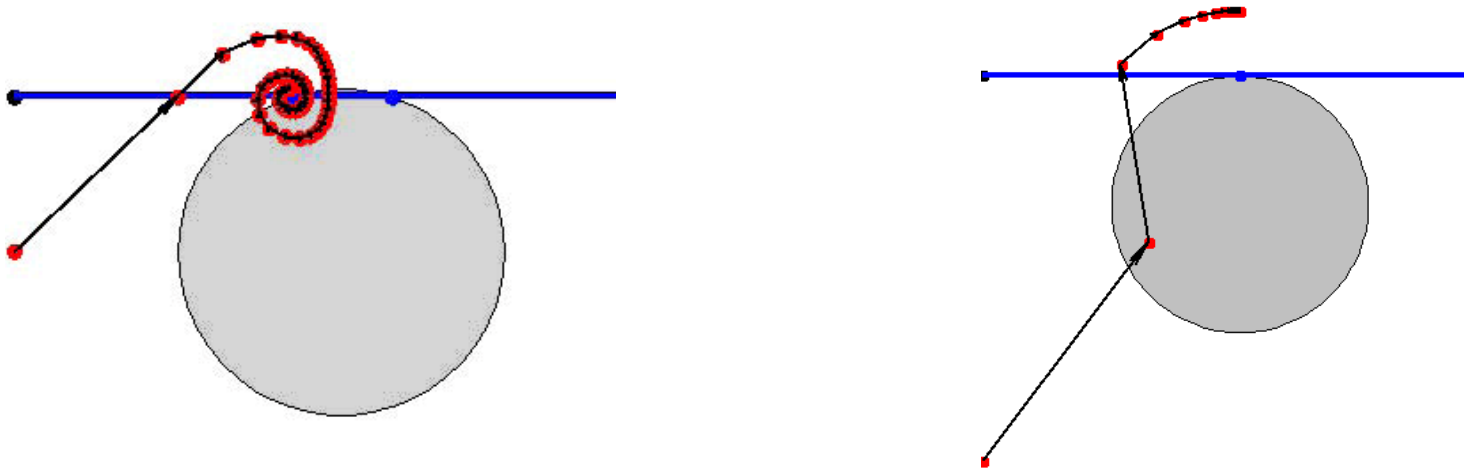
The Lions-Mercer iteration, also known as the Douglas-Rachford or Feinup iteration, is defined by the procedure: reflect, reflect and average:

$$x \mapsto T(x) := \frac{x + R_A\left(R_B(x)\right)}{2}$$

In the simple 2-D case of a horizontal line of height α, we obtain the explicit iteration:

$$x_{n+1} := \cos\theta_n, \quad y_{n+1} := y_n + \alpha - \sin\theta_n, \quad (\theta_n := \arg z_n)$$

For $0 < \alpha < 1$, spiraling is ubiquitous: (α = 0.95 on left, and 1.0 on right):



Exploring iterations using Cinderella

Iterations such as this, as well as many other graphical phenomena, may be explored using the Cinderella online tool: http://www.cinderella.de.

Two applets have been defined, working with Cinderella, for exploring Lions-Mercer iterations:

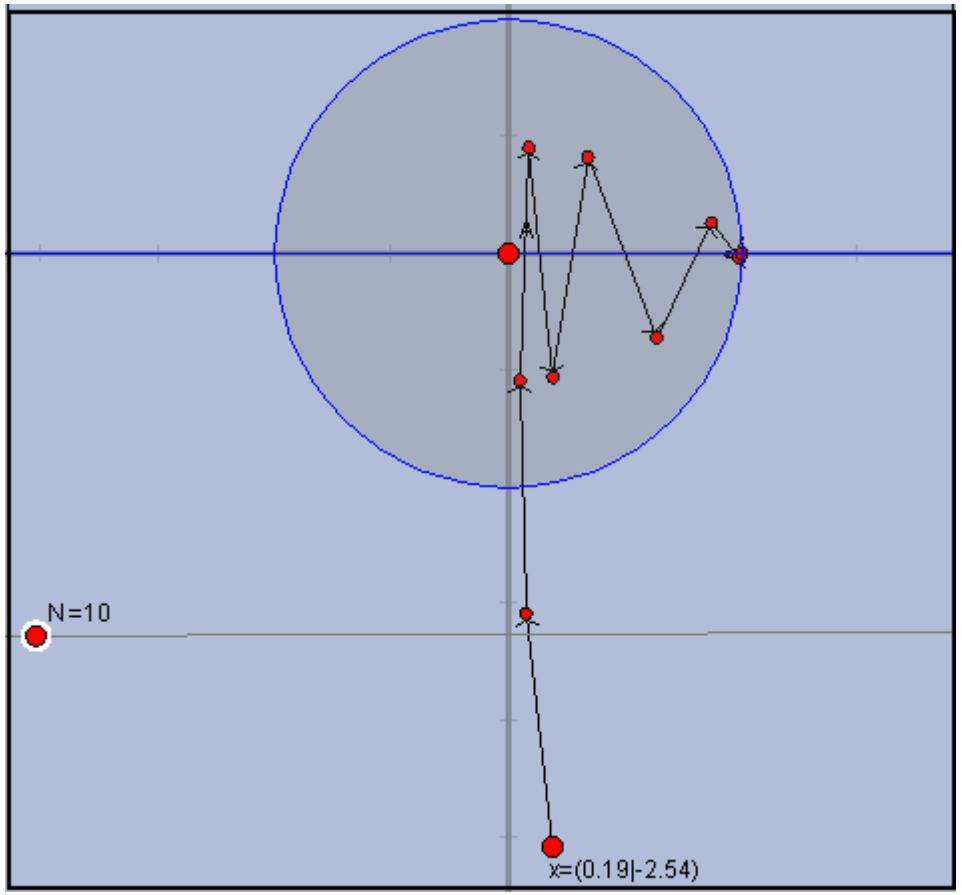
A1. http://users.cs.dal.ca/~jborwein/reflection.html

A2. http://users.cs.dal.ca/~jborwein/expansion.html

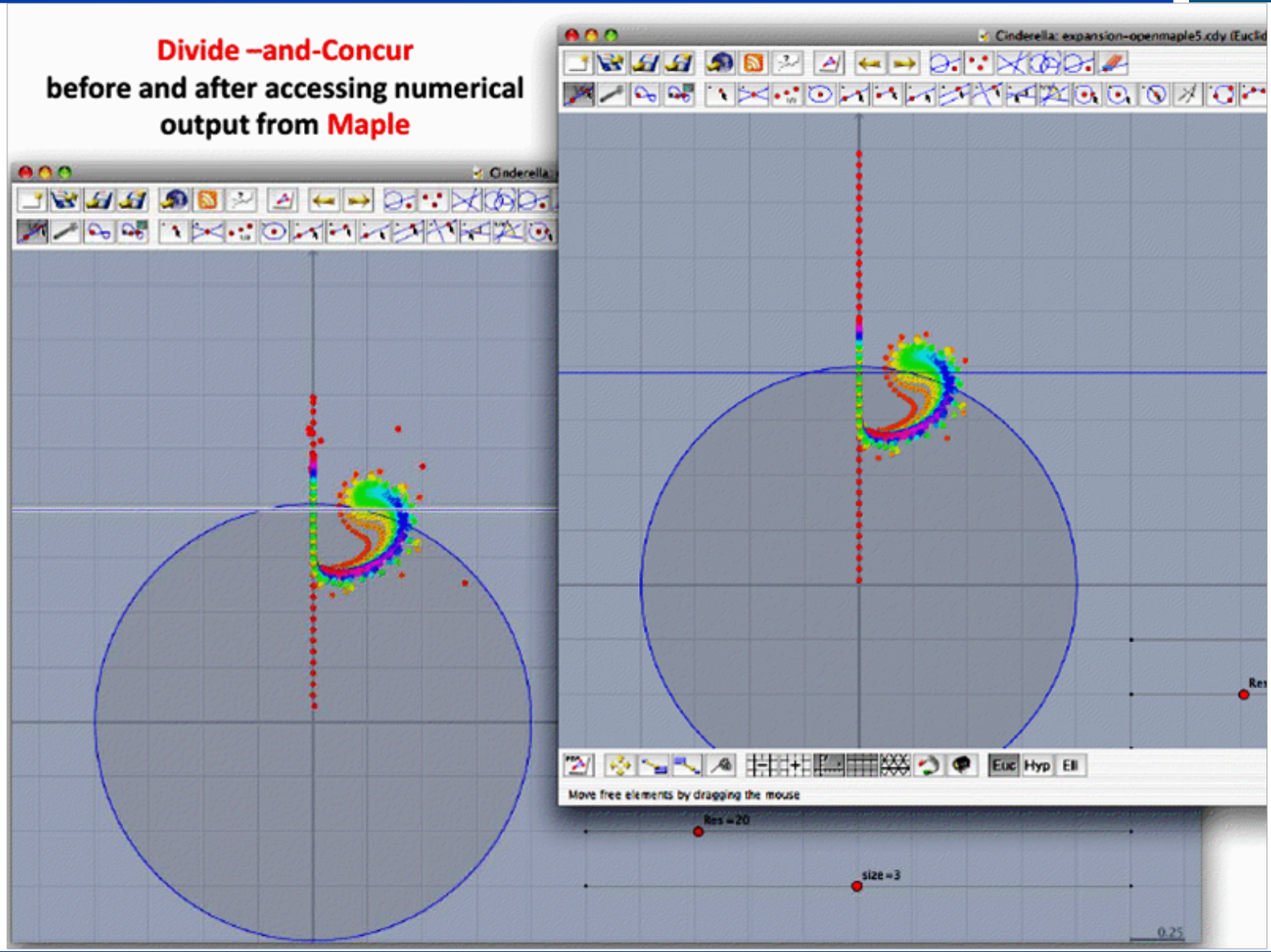
For Applet A1, we observed that (see graphic on next slide):

- As long as the iterate is outside the unit circle the next point is always closer to the origin;
- Once inside the circle the iterate never leaves;
- The angle now oscillates to zero and the trajectory hence converges to(1,0).

Iterations with Applet A1



Iterations with Applet A2: Double vs multiple precision

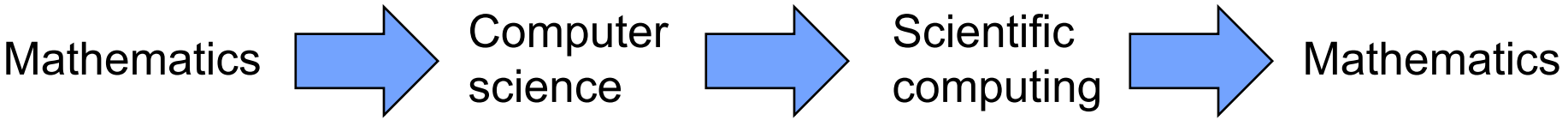


Experimental math: Discovering new mathematical results by computer

- Compute various mathematical entities (limits, infinite series sums, definite integrals) to high precision, typically 100-1000 digits.
- Use algorithms such as PSLQ to recognize these entities in terms of well-known mathematical constants.
- When results are found experimentally, seek to find formal mathematical proofs of the discovered relations.

Many results have recently been found using this methodology, both in pure mathematics and in mathematical physics.

"If mathematics describes an objective world just like physics, there is no reason why inductive methods should not be applied in mathematics just the same as in physics." – Kurt Godel



The PSLQ integer relation algorithm

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

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

(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*.

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

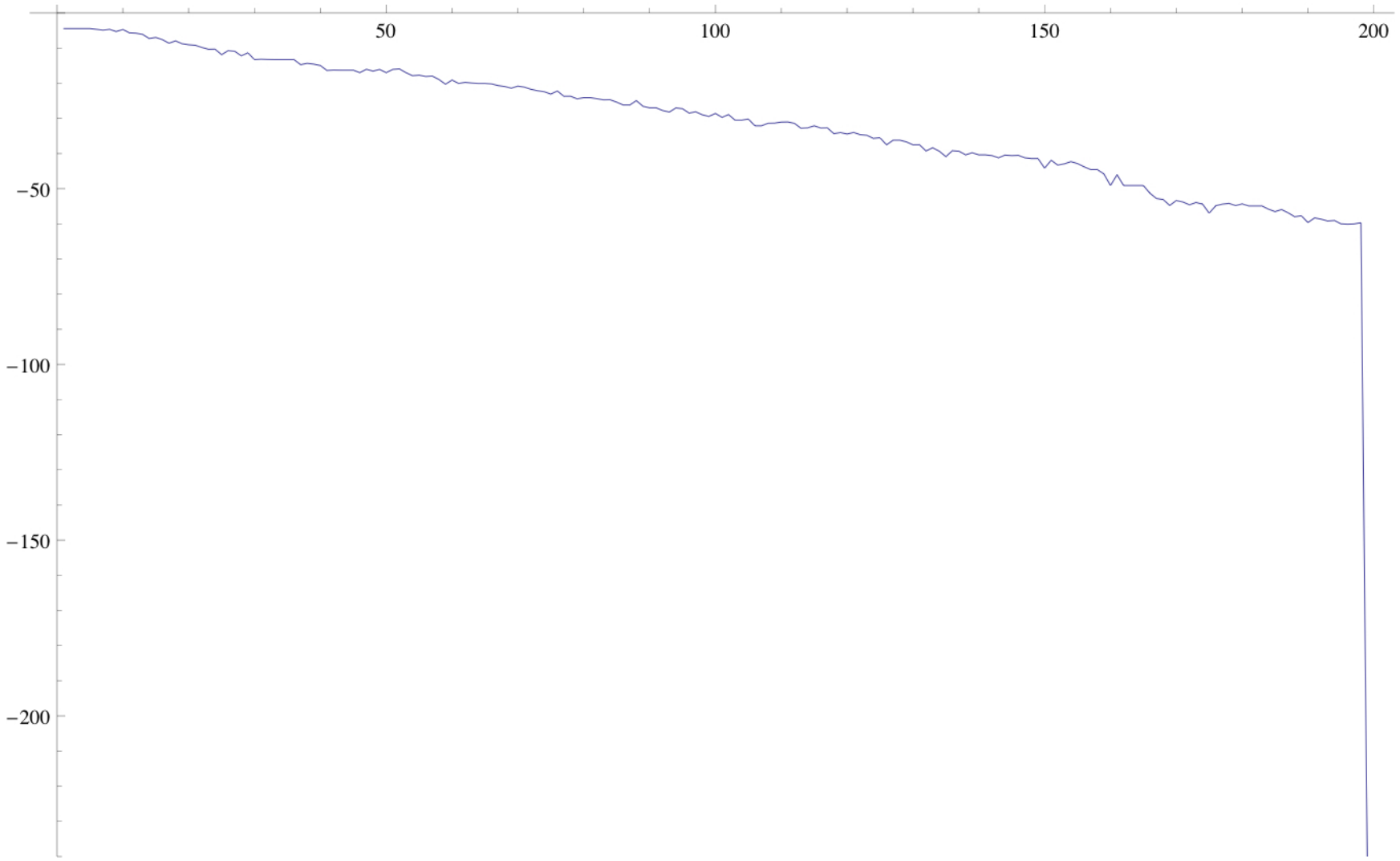
PSLQ, continued

- PSLQ constructs a sequence of integer-valued matrices B_n that reduces the vector $y = x * B_n$, until either the relation is found (as one of the columns of B_n), or else precision is exhausted.
- At the same time, PSLQ generates a steadily growing bound on the size of any possible relation.
- When a relation is found, the size of smallest entry of the *y* vector suddenly drops to roughly "epsilon" (i.e. 10^{-p}, where *p* is the number of digits of precision).
- The size of this drop can be viewed as a "confidence level" that the relation is real and not merely a numerical artifact -- a drop of 20+ orders of magnitude almost always indicates a real relation.

Several efficient variants of PSLQ are available:

- 2-level and 3-level PSLQ: performs almost all PSLQ iterations with only double precision, updating full-precision arrays as needed. Hundreds of times faster than the original full-precision PSLQ algorithm.
- Multi-pair PSLQ: dramatically reduces the number of iterations required. Designed for parallel system, but runs faster even on 1 CPU.

Decrease of $\log_{10}(\min |y_i|)$ in multipair PSLQ run



PSLQ discovery: The BBP formula for Pi

In 1996, this new formula for π was found using a PSLQ program:

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

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

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

1. DHB, 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.

High-precision tanh-sinh quadrature

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

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

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 Euler-Maclaurin formula implies that the simple summation above is remarkably accurate. Reducing h by half typically doubles the number of correct digits.

For our applications, we have found that tanh-sinh is the best general-purpose integration scheme for functions with vertical derivatives or singularities at endpoints, or for any function at very high precision (> 1000 digits). Otherwise we use Gaussian quadrature.

1. DHB, 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.

Ising integrals from mathematical physics

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

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

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

$$E_n = 2\int_0^1 \cdots \int_0^1 \left(\prod_{1\le j<k\le n} \frac{u_k - u_j}{u_k + u_j}\right)^2 \mathrm{d}t_2\, dt_3 \cdots \mathrm{d}t_n$$

where in the last line $u_k = t_1\, t_2 \ldots t_k$.

DHB, 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.

Limiting value of C_n: What is this number?

Key observation: The C_n integrals can be converted to one-dimensional integrals involving the modified Bessel function $K_0(t)$:

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

1000-digit numerical values, computed using this formula, approach a limit:

$$C_{1024} = 0.63047350337438679612204019271087890435458707871273234\ldots$$

What is this limit? We copied the first 50 digits of this numerical value into the online Inverse Symbolic Calculator (ISC):
http://ddrive.cs.dal.ca/~isc or **http://carma-lx1.newcastle.edu.au:8087/**

The result was:

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

where γ denotes Euler's constant.

Other Ising integral evaluations found using high-precision PSLQ

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

where ζ is the Riemann zeta function and $\mathrm{Li}_n(x)$ is the polylog function. D_2, D_3 and D_4 were originally provided to us by mathematical physicist Craig Tracy, who hoped that our tools could help identify D_5.

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.

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

We also computed D_5 to 500 digits, but were unable to identify it. The digits are available if anyone wishes to further explore this question.

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

Recursions in Ising integrals

Consider the 2-parameter class of Ising integrals (which arises in QFT for odd *k*):

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

After computing 1000-digit numerical values for all *n* up to 36 and all *k* up to 75 (performed on a highly parallel computer system), we discovered (using PSLQ) linear relations in the rows of this array. For example, when *n* = 3:

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

Similar, but more complicated, recursions have been found for all *n*.

1. DHB, D. Borwein, J. M. Borwein and R. Crandall, "Hypergeometric Forms for Ising-Class Integrals," *Experimental Mathematics*, vol. 16 (2007), pg. 257-276.

2. J. M. Borwein and B. Salvy, "A Proof of a Recursion for Bessel Moments," *Experimental Mathematics*, vol. 17 (2008), pg. 223-230.

Box integrals

The following integrals appear in numerous arenas of math and physics:

$$B_n(s) := \int_0^1 \cdots \int_0^1 \left(r_1^2 + \cdots + r_n^2\right)^{s/2} \, dr_1 \cdots dr_n$$

$$\Delta_n(s) := \int_0^1 \cdots \int_0^1 \left((r_1 - q_1)^2 + \cdots + (r_n - q_n)^2\right)^{s/2} \, dr_1 \cdots dr_n \, dq_1 \cdots dq_n$$

- $B_n(1)$ is the expected distance of a random point from the origin of *n*-cube.
- $\Delta_n(1)$ is the expected distance between two random points in *n*-cube.
- $B_n(-n+2)$ is the expected electrostatic potential in an *n*-cube whose origin has a unit charge.
- $\Delta_n(-n+2)$ is the expected electrostatic energy between two points in a uniform *n*-cube of charged "jellium."
- Recently integrals of this type have arisen in neuroscience – e.g., the average distance between synapses in a mouse brain.

DHB, J. M. Borwein and R. E. Crandall, "Box integrals," *Journal of Computational and Applied Mathematics*, vol. 206 (2007), pg. 196-208.

Evaluations of box integrals

n	s	$B_n(s)$
any	even $s \geq 0$	rational, e.g., : $B_2(2) = 2/3$
1	$s \neq -1$	$\frac{1}{s+1}$
2	-4	$-\frac{1}{4} - \frac{\pi}{8}$
2	-3	$-\sqrt{2}$
2	-1	$2\log(1+\sqrt{2})$
2	1	$\frac{1}{3}\sqrt{2} + \frac{1}{3}\log(1+\sqrt{2})$
2	3	$\frac{7}{5}\sqrt{2} + \frac{3}{20}\log(1+\sqrt{2})$
2	$s \neq -2$	$\frac{2}{2+s}\ {}_2F_1\left(\frac{1}{2}, -\frac{s}{2}; \frac{3}{2}; -1\right)$
3	-5	$-\frac{1}{6}\sqrt{3} - \frac{1}{12}\pi$
3	-4	$-\frac{3}{2}\sqrt{2}\arctan\frac{1}{\sqrt{2}}$
3	-2	$-3\mathrm{G} + \frac{3}{2}\pi\log(1+\sqrt{2}) + 3\ \mathrm{Ti}_2(3-2\sqrt{2})$
3	-1	$-\frac{1}{4}\pi + \frac{3}{2}\log\left(2+\sqrt{3}\right)$
3	1	$\frac{1}{4}\sqrt{3} - \frac{1}{24}\pi + \frac{1}{2}\log\left(2+\sqrt{3}\right)$
3	3	$\frac{2}{5}\sqrt{3} - \frac{1}{60}\pi - \frac{7}{20}\log\left(2+\sqrt{3}\right)$

Here *F* is hypergeometric function; *G* is Catalan; Ti is Lewin's inverse-tan function.

Elliptic function integrals

The research with ramble integrals led us to study integrals of the form:

$$I(n_0, n_1, n_2, n_3, n_4) \quad := \quad \int_0^1 x^{n_0} K^{n_1}(x) K'^{n_2}(x) E^{n_3}(x) E'^{n_4}(x) \mathrm{d}x,$$

where *K*, *K'*, *E*, *E'* are elliptic integral functions:

$$\begin{aligned} K(x) &:= \int_0^1 \frac{\mathrm{d}t}{\sqrt{(1-t^2)(1-x^2t^2)}} \\ K'(x) &:= K(\sqrt{1-x^2}) \\ E(x) &:= \int_0^1 \frac{\sqrt{1-x^2t^2}}{\sqrt{1-t^2}}\,\mathrm{d}t \\ E'(x) &:= E(\sqrt{1-x^2}) \end{aligned}$$

J. Wan, "Moments of products of elliptic integrals," *Advances in Applied Mathematics*, vol. 48 (2012), available at http://carma.newcastle.edu.au/jamesw/mkint.pdf.
DHB and J. M. Borwein, "Hand-to-hand combat with thousand-digit integrals," *Journal of Computational Science*, vol. 3 (2012), pg. 77-86, http://www.davidhbailey.com/dhbpapers/combat.pdf.

Relations found among the I integrals

Thousands of relations have been found among the I integrals. For example, among the class with $n_0 <= D_1 = 4$ and $n_1 + n_2 + n_3 + n_4 = D_2 = 3$ (a set of 100 integrals), we found that all can be expressed in terms of an integer linear combination of 8 simple integrals. For example:

$$81\int_0^1 x^3K^2(x)E(x)\mathrm{d}x \stackrel{?}{=} -6\int_0^1 K^3(x)\mathrm{d}x - 24\int_0^1 x^2K^3(x)\mathrm{d}x$$
$$+51\int_0^1 x^3K^3(x)\mathrm{d}x + 32\int_0^1 x^4K^3(x)\mathrm{d}x$$

$$-243\int_0^1 x^3K(x)E(x)K'(x)\mathrm{d}x \stackrel{?}{=} -59\int_0^1 K^3(x)\mathrm{d}x + 468\int_0^1 x^2K^3(x)\mathrm{d}x$$
$$+156\int_0^1 x^3K^3(x)\mathrm{d}x - 624\int_0^1 x^4K^3(x)\mathrm{d}x - 135\int_0^1 xK(x)E(x)K'(x)\mathrm{d}x$$

$$-20736\int_0^1 x^4E^2(x)K'(x)\mathrm{d}x \stackrel{?}{=} 3901\int_0^1 K^3(x)\mathrm{d}x - 3852\int_0^1 x^2K^3(x)\mathrm{d}x$$
$$-1284\int_0^1 x^3K^3(x)\mathrm{d}x + 5136\int x^4K^3(x)\mathrm{d}x - 2592\int_0^1 x^2K^2(x)K'(x)\mathrm{d}x$$
$$-972\int_0^1 K(x)E(x)K'(x)\mathrm{d}x - 8316\int_0^1 xK(x)E(x)K'(x)\mathrm{d}x.$$

Algebraic numbers in Poisson potential functions associated with lattice sums

Lattice sums arising from the Poisson equation have been studied widely in mathematical physics and also in image processing.

In two 2012 papers (below), we numerically discovered, and then proved, that for rational (*x*, *y*), the two-dimensional Poisson potential function satisfies

$$\phi_2(x,y) = \frac{1}{\pi^2} \sum_{m,n \text{ odd}} \frac{\cos(m\pi x)\cos(n\pi y)}{m^2+n^2} = \frac{1}{\pi}\log\alpha$$

where α is an *algebraic number*, i.e., the root of an integer polynomial:

$$0 = a_0 + a_1\alpha + a_2\alpha^2 + \cdots + a_n\alpha^n$$

The minimal polynomials for these α were found by PSLQ calculations, with the (*n*+1)-long vector (1, α, α^2, …, α^n) as input, where α = exp (π $\phi_2(x,y)$). PSLQ returns the vector of integer coefficients (a_0, a_1, a_2, …, a_n) as output.

1. DHB, J. M. Borwein, R. E. Crandall and J. Zucker, "Lattice sums arising from the Poisson equation," manuscript, http://www.davidhbailey.com/dhbpapers/PoissonLattice.pdf
2. DHB and J. M. Borwein, "Compressed lattice sums arising from the Poisson equation: Dedicated to Professor Hari Sirvastava," manuscript, http://www.davidhbailey.com/dhbpapers/Poissond.pdf.

Samples of minimal polynomials found by PSLQ

k	Minimal polynomial for exp (8 π ϕ_2(1/*k*,1/*k*))
5	$1 + 52\alpha - 26\alpha^2 - 12\alpha^3 + \alpha^4$
6	$1 - 28\alpha + 6\alpha^2 - 28\alpha^3 + \alpha^4$
7	$-1 - 196\alpha + 1302\alpha^2 - 14756\alpha^3 + 15673\alpha^4 + 42168\alpha^5 - 111916\alpha^6 + 82264\alpha^7$ $-35231\alpha^8 + 19852\alpha^9 - 2954\alpha^{10} - 308\alpha^{11} + 7\alpha^{12}$
8	$1 - 88\alpha + 92\alpha^2 - 872\alpha^3 + 1990\alpha^4 - 872\alpha^5 + 92\alpha^6 - 88\alpha^7 + \alpha^8$
9	$-1 - 534\alpha + 10923\alpha^2 - 342864\alpha^3 + 2304684\alpha^4 - 7820712\alpha^5 + 13729068\alpha^6$ $-22321584\alpha^7 + 39775986\alpha^8 - 44431044\alpha^9 + 19899882\alpha^{10} + 3546576\alpha^{11}$ $-8458020\alpha^{12} + 4009176\alpha^{13} - 273348\alpha^{14} + 121392\alpha^{15}$ $-11385\alpha^{16} - 342\alpha^{17} + 3\alpha^{18}$
10	$1 - 216\alpha + 860\alpha^2 - 744\alpha^3 + 454\alpha^4 - 744\alpha^5 + 860\alpha^6 - 216\alpha^7 + \alpha^8$

The minimal polynomial for exp (8 π ϕ_2(1/32,1/32)) has degree 128, with individual coefficients ranging from 1 to over 10^{56}. This PSLQ computation required 10,000-digit precision. See next slide.

Degree-128 minimal polynomial for exp (8 π ϕ_2(1/32,1/32))

Formal proof versus high-precision verification

- Results such as those mentioned above must still be proven rigorously.
- However, strong numerical evidence is often a good impetus to find a proof – “discovery is 9/10 of the proof.”

What is more firmly established?

- A formally proven result, whose proof required hundreds of pages, which crucially relies on tens of earlier results by other authors, and which has only been read in detail by a handful of mathematicians.
- A numerically discovered experimental identity, for which no known formal proof is available, but which has been checked to thousands of digits, independently on separate computers.

Cautionary Example

These constants agree to 42 decimal digit accuracy, but are NOT equal:

$$\int_0^\infty \cos(2x) \prod_{n=1}^{\infty} \cos(x/n)\,\mathrm{d}x \quad =$$

$$0.392699081698724154807830422909937860524645343187231595926$$

$$\frac{\pi}{8} \quad =$$

$$0.392699081698724154807830422909937860524646174921888227621$$

Richard Crandall has now shown that this integral is merely the first term of a very rapidly convergent series that converges to $\pi/8$:

$$\frac{\pi}{8} \quad = \quad \sum_{m=0}^{\infty} \int_0^\infty \cos[2(2m+1)x] \prod_{n=1}^{\infty} \cos(x/n)\,\mathrm{d}x$$

1. 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 .

2. R. E. Crandall, "Theory of ROOF Walks, 2007, available at http://people.reed.edu/~crandall/papers/ROOF.pdf.

Limitations of *Mathematica* and *Maple* for computational mathematics

Mathematica or *Maple* is our first choice whenever symbolic or numeric computations are required. However, both have their limitations.

For example, in a study of Mordell-Tornheim-Witten sums (which arise in mathematical physics), we required high-precision numeric values of derivatives with respect to the order *s* of polylogarithm functions:

$$\frac{\partial \mathrm{Li}_s(z)}{\partial s}, \quad \text{where} \quad \mathrm{Li}_s(z) = \sum_{k=1}^{\infty} \frac{z^k}{k^s}$$

Maple is not able to numerically evaluate these derivatives at all.

Mathematica, when asked for 4000 digits, returned only 400 correct digits (at some arguments).

DHB, J. M. Borwein and R. E. Crandall, "Computation and theory of extended Mordell-Tornheim-Witten sums," *Mathematics of Computation*, to appear, 31 Jul 2012, http://www.davidhbailey.com/dhbpapers/BBC.pdf

Summary

- Large-scale, highly parallel computation places enormous stress on the numerical reliability and reproducibility of scientific computations.
- Double-double or higher precision arithmetic is a practical means of dealing with these numerical difficulties in many cases.
- Many real-world applications have now been identified that *require* high-precision arithmetic.
- Some research studies, particularly in experimental mathematics and mathematical physics, require hundreds or even thousands of digits.
- Software is now available, mostly for free, to facilitate conversion. In most cases, one need only change the type of variables that are to be treated as high-precision variables.