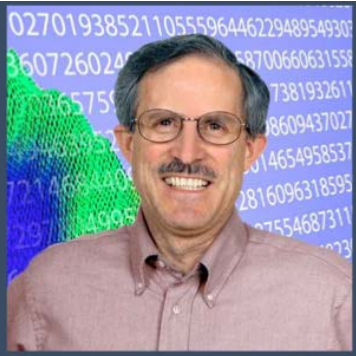


Fooling the masses: Reproducibility in high-performance computing

Speakers: Prof David H. Bailey (Lawrence Berkeley Lab (retired) and U.C. Davis, USA) and Prof Jonathan Borwein (University of Newcastle)

Reproducibility in scientific computing

A December 2012 workshop on reproducibility in computing, held at Brown University in Rhode Island, USA, noted that

> *Science is built upon the foundations of theory and experiment validated and improved through open, transparent communication. With the increasingly central role of computation in scientific discovery, this means communicating all details of the computations needed for others to replicate the experiment. ... The "reproducible research" movement recognizes that traditional scientific research and publication practices now fall short of this ideal, and encourages all those involved in the production of computational science ... to facilitate and practice really reproducible research.*

- V. Stodden, D. H. Bailey, J. Borwein, R. J. LeVeque, W. Rider and W. Stein, "Setting the default to reproducible: Reproducibility in computational and experimental mathematics," `http://www.davidhbailey.com/dhbpapers/icerm-report.pdf`.

Reproducibility in scientific computing, continued

Issues identified in the ICERM report and other studies include:

- The need to carefully document the full context of computational experiments—system environment, input data, code used, computed results, etc.
- The need to save the code and output data in a permanent repository.
- The need for reviewers, research institutions and funding agencies to recognize the importance of computing and computing professionals, and to allocate funding for after-the-grant support and repositories.
- The increasing importance of numerical reproducibility, and the need for tools to ensure and enhance numerical reliability.
- The need to encourage publication of negative results—other researchers can often learn from them.
- The re-emergence of the need to ensure responsible reporting of performance.

Reproducibility in biomedicine

The biomedical field has been stung by numerous cases where pharma products or lab discoveries look great, but then fall flat or cannot be reproduced. Examples:

- In 2004, GlaxoSmithKline acknowledged that while some trials of Paxil found it effective for depression in children, other unpublished studies showed no benefit.
- In 2011, Bayer researchers reported that they were able to reproduce the results of only 17 of 67 published studies they examined.
- In 2012, Amgen researchers reported that they were able to reproduce the results of only 6 of 53 published cancer studies.
- In 2014, a review of Tamiflu found that while it made flu symptoms disappear a bit sooner, it did not stop serious complications or keep people out of the hospital.
- In June 2014, Japanese retracted an earlier, highly publicized claim that they had produced pluripotent stem cells from ordinary skin or blood cells.

These experiences have exposed a fundamental flaw in methodology:
Only publicizing the results of successful trials introduces a bias into the results.

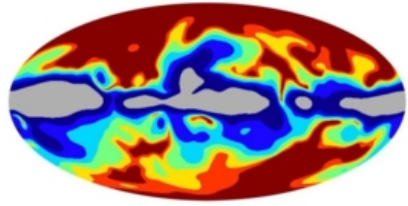
The AllTrials movement would require all results to be public: `http://www.alltrials.net`

Reproducibility in physics

In March 2014, a team of researchers from Harvard University made the dramatic announcement that they had discovered an interesting "twisting" pattern in cosmic microwave background data, measured using their BICEP2 experimental system.

This pattern fit very well with the hypothesized pattern of the most commonly assumed model of "inflation" in the first tiny fraction of a second after the big bang, and thus has been trumpeted as the first experimental evidence of the inflationary cosmology.

But other researchers had difficulty reconstructing the claimed results. Finally, two teams challenged the BICEP2 findings, saying that the results could more readily be explained by dust in the Milky Way.



- Ron Cowen, "Doubt grows about gravitational waves detection," *Scientific American*, 2 Jun 2014.

Reproducibility in social science

The "blank slate" paradigm (1920–1990):

- The human mind at birth is a *tabula rasa* ("blank slate").
- Heredity and biology play no significant role in human psychology; all personality and behavioral traits are socially constructed.
- Pre-modern societies were serene, devoid of most psychological and social problems that afflict modern societies.

As Ashley Montagu wrote in 1973,

> *Man is man because he has no instincts, because everything he is and has become he has learned, acquired, from his culture, from the man-made part of the environment, from other human beings.*

The fall of the blank slate

The current consensus in the field, based on latest research:

- Humans at birth possess sophisticated facilities for social interaction, language acquisition, pattern recognition, navigation and counting.
- Heredity, evolution and biology are major factors in human personality.
- Some personality traits are more than 50% heritable.
- Pre-modern societies had more crime, war and social problems than modern western societies.

How did the early 20th century social scientists err so badly?

- Sloppy experimental methodology and analysis.
- Pervasive wishful thinking and politically correct biases.
- Ignoring or dismissing data that runs counter to predisposition.

- S. Pinker, *The Blank Slate: The Modern Denial of Human Nature*, Penguin Books, 2003.

Science adds statistical checks to peer-review

The journal Science is adding an extra round of statistical checks to its peer-review process, editor-in-chief Marcia McNutt announced on July 3. The policy follows similar efforts from other journals, after widespread concern that basic mistakes in data analysis are contributing to the irreproducibility of many published research findings.

"Readers must have confidence in the conclusions published in our journal," writes McNutt in an editorial today. Working with the American Statistical Association, the journal has appointed seven experts to a statistics board of reviewing editors (SBoRE). Manuscript will be flagged up for additional scrutiny by the journal's internal editors, or by its existing Board of Reviewing Editors (more than 100 scientists whom the journal regularly consults on papers) or by outside peer reviewers. The SBoRE panel will then find external statisticians to review these manuscripts.

- Richard Van Noorden, "Major scientific journal joins push to screen statistics in papers it publishes," *Nature*, 6 Jul 2014.

Reproducibility in finance

Finance, like the pharmaceutical world, has been stung with numerous instances of investment strategies that look great on paper, but fall flat in practice. A primary cause is *statistical overfitting of backtest (historical market) data*.

When a computer can analyze thousands or millions of variations of a given strategy, it is almost certain that the best such strategy, measured by backtests, will be overfit and thus of dubious value.

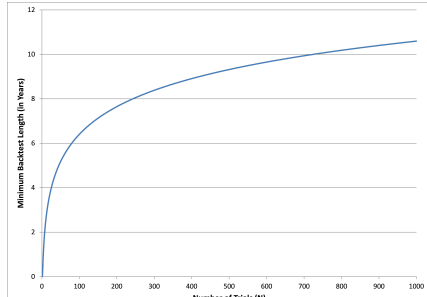
In two 2014 papers by myself and three co-authors, we derive (a) a formula relating the number of trials to the minimum backtest length, and (b) a formula for the probability of backtest overfitting. We also show that under the assumption of memory in markets, overfit strategies are actually somewhat prone to *lose* money.

- D. H. Bailey, J. M. Borwein, M. Lopez de Prado and Q. J. Zhu, "Pseudo-mathematics and financial charlatanism: The effects of backtest over fitting on out-of-sample performance," *Notices of the American Mathematical Society*, May 2014, pg. 458–471.
- D. H. Bailey, J. M. Borwein, M. Lopez de Prado and Q. J. Zhu, "The probability of backtest overfitting," manuscript, 12 Feb 2014, `http://ssrn.com/abstract=2326253`.

How easy is it to overfit a backtest? Answer: Very easy!

- If only 2 years of daily backtest data are available, then no more than 7 strategy variations should be tried.
- If only 5 years of daily backtest data are available, then no more than 45 strategy variations should be tried.
- *A backtest that does not report the number of trials N makes it impossible to assess the risk of overfitting.*
- Given any desired performance level, a financial researcher just needs to keep trying alternative parameters for that strategy!

$$MinBTL \approx \left(\frac{(1-\gamma)Z^{-1}\left[1-\frac{1}{N}\right] + \gamma Z^{-1}\left[1-\frac{1}{N}e^{-1}\right]}{E[\max_N]} \right)^2$$



An absurd investment strategy

- A financial advisor sends letters to $10,240 = 10 \times 2^{10}$ potential clients, with 5120 letters predicting a certain security will go up, and the other half predicting it will go down.
- One month later, the advisor sends letters only to the 5120 investors who were previously sent the correct prediction, with 2560 letters predicting a certain security will go up, and the other half predicting it will go down.
- The advisor continues this process for 10 months.
- The remaining ten investors, so impressed by the advisor's ten consecutive spot-on predictions, will entrust to him/her all of their assets!

This strategy is absurd; even fraudulent.

But why is marketing a statistically overfit strategy, where potential investors are not informed of the number of trials behind the strategy, any different?

Why the silence in the mathematical finance community?

- Historically scientists have led the way in exposing those who utilize pseudoscience to extract a commercial benefit: e.g., in the 18th century, physicists exposed the nonsense of astrologers.
- Yet financial mathematicians in the 21st century have remained disappointingly silent with the regards to those in the community who, knowingly or not:
  1. Fail to disclose the number of models or variations that were used to develop an investment strategy.
  2. Make vague predictions that do not permit rigorous testing and falsification.
  3. Misuse probability theory, statistics and stochastic calculus.
  4. Use dubious technical jargon: "stochastic oscillators," "Fibonacci ratios," "cycles," "Elliot wave," "Golden ratio," "parabolic SAR," "pivot point," "momentum," etc.

As we recently wrote in our paper "Pseudo-Mathematics and Financial Charlatanism": **"Our silence is consent, making us accomplices in these abuses."**

- D. H. Bailey, J. M. Borwein, M. Lopez de Prado and Q. J. Zhu, "Pseudo-mathematics and financial charlatanism: The effects of backtest over fitting on out-of-sample performance," *Notices of the American Mathematical Society*, May 2014, pg. 458–471.

Reproducibility in highly parallel computing: History from 1990-1994

Background:

- Many new parallel systems had been introduced; each claimed theirs was best.
- Many researchers were excited about the potential of highly parallel systems.
- Few standard benchmarks and testing methodologies had been established.
- It was hard to reproduce published performance results; much confusion reigned.
- Overall, the level of rigor and peer review in the field was rather low.

In response, in 1991 I published a humorous essay "Twelve Ways to Fool the Masses," poking fun at some of the abuses.

Since abuses continued, I presented a talk at Supercomputing 1992 and published a paper with specific examples.

- D. H. Bailey, "Misleading performance reporting in the supercomputing field," *Scientific Programming*, vol. 1., no. 2 (Winter 1992), pg. 141–151.

1991 paper: "Twelve ways to fool the masses in highly parallel computing"

1. Quote 32-bit performance results, not 64-bit results, but don't mention this in paper.
2. Present performance figures for an inner kernel, then represent these figures as the performance of the entire application.
3. Quietly employ assembly code and other low-level language constructs.
4. Scale up the problem size with the number of processors, but omit any mention of this.
5. Quote performance results projected to a full system.
6. Compare your results against scalar, unoptimized code on conventional systems.
7. When run times are compared, compare with an old code on an obsolete system.
8. Base Mflop/s rates on the operation count of the parallel implementation, instead of the best practical serial algorithm.
9. Quote performance as processor utilization, parallel speedups or Mflop/s per dollar.
10. Mutilate the algorithm used in the parallel implementation to match the architecture.
11. Measure parallel run times on a dedicated system, but measure conventional run times in a busy environment.
12. If all else fails, show pretty pictures and animated videos, and don't discuss performance.

1992 paper: Scaling performance results to full-sized system

In some published papers and conference presentations, performance results on small-sized parallel systems were linearly scaled to full-sized systems, without even clearly disclosing this fact.

Example: 8,192-CPU performance results were linearly scaled to 65,536-CPU results, simply by multiplying by eight.

Excuse: "We can't afford a full-sized system."

This and the other examples mentioned in the next few viewgraphs are presented in:

- D. H. Bailey, "Misleading performance reporting in the supercomputing field," *Scientific Programming*, vol. 1., no. 2 (Winter 1992), pg. 141–151.

1992 paper: Using inefficient algorithms on highly parallel systems

In many cases, inefficient algorithms were employed for the highly parallel implementation, requiring many more operations, thus producing artificially high Mflop/s rates:

- Numerous researchers cited parallel PDE performance based explicit schemes, where implicit schemes were known to be much better.
  Excuse: Explicit schemes "run better" on the researchers' parallel system.
- One paper cited performance for computing a 3D discrete Fourier transform by direct evaluation of the defining formula ($8n^2$ operations), rather than by using a fast Fourier transform ($5n \log_2 n$).
  Excuse: Direct computation was "more appropriate" for the architecture being analyzed.

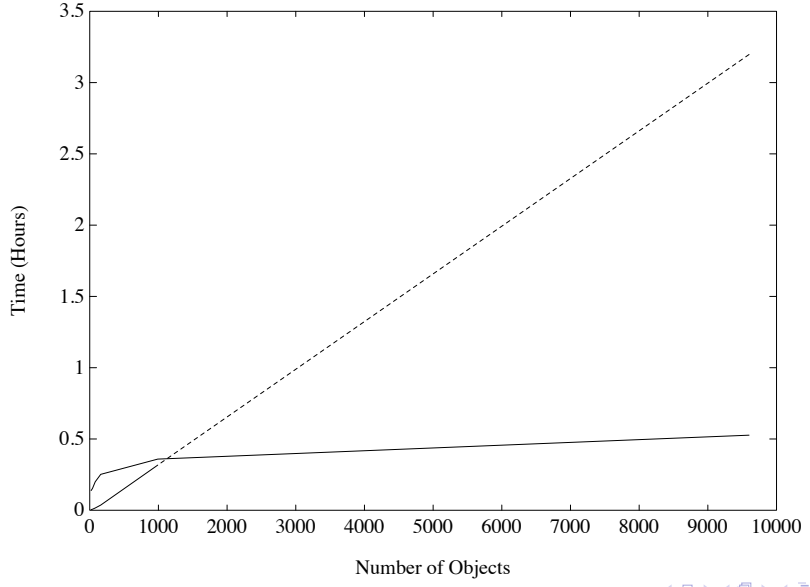
Both examples violate a rule of professional performance reporting, namely to base the operation count (when computing Mflop/s or Gflop/s rates) on the *best practical serial algorithm*, no matter what scheme was actually used on the parallel system.

1992 paper: Not actually performing a claimed computation

Abstract of published paper: "The current Connection Machine implementation runs at 300-800 Mflop/s on a full [64K] CM-2, or at the speed of a single processor of a Cray-2 on 1/4 of a CM-2."

- Excerpt from text: "This computation requires 568 iterations (taking 272 seconds) on a 16K Connection Machine."
  In other words, the computation was run on a 16K system, not on a 64K system; the figures cited in the Abstract were merely multiplied by four.
- Excerpt from text: "In contrast, a Convex C210 requires 909 seconds to compute this example. Experience indicates that for a wide range of problems, a C210 is about 1/4 the speed of a single processor Cray-2."
  In other words, the computation mentioned in the Abstract was not actually run on a Cray-2; instead, it was run on a Convex system, and a questionable rule-of-thumb scaling factor was used to produce the Cray-2 rate.

1992 paper: Performance plot [parallel (lower) vs vector (upper)]



1992 paper: Data for performance plot

Problem size (x axis)	Parallel system run time	Vector system run time
20	8:18	0:16
40	9:11	0:26
80	11:59	0:57
160	15:07	2:11
990	21:32	19:00
9600	31:36	3:11:50*

Details in text of paper:

- In last entry, the 3:11:50 figure is an "estimate."
- The vector system code is "not optimized."

Note that the parallel system is actually slower than the vector system for all cases, except for the last (estimated) entry. Also, except for the last entry, all real data in the graph is in the lower left corner.

Fast forward to 2014: New ways to fool the masses

- Cite performance rates for a run with only one processor core active in a shared-memory multi-core node, producing artificially inflated performance (since there is no shared memory interference) and wasting resources (since most cores are idle).
  - Example: Cite performance on "1024 cores," even though the code was run on 1024 multicore nodes, one core per node, with 15 out of 16 cores idle on each node.
- Claim that since one is using a graphics processing unit (GPU) system, that efficient parallel algorithms must be discarded in favor of more basic algorithms.
- Cite performance rates only for a core algorithm (such as FFT or linear system solution), even though full-scale applications have been run on the system.
- List only the best performance figure in the paper, even though the run was made numerous times (recall the experience of pharmaceutical tests).
- Employ special hardware, operating system or compiler settings that are not appropriate for real-world production usage.
- Define "scalability" as successful execution on a large number of CPUs, regardless of performance.

Numerical reproducibility in high-performance computing

The report mentioned above on reproducibility in high-performance computing noted:

> *Numerical round-off error and numerical differences are greatly magnified as computational simulations are scaled up to run on highly parallel systems. As a result, it is increasingly difficult to determine whether a code has been correctly ported to a new system, because computational results quickly diverge from standard benchmark cases. And it is doubly difficult for other researchers, using independently written codes and distinct computer systems, to reproduce published results.*

- V. Stodden, D. H. Bailey, J. Borwein, R. J. LeVeque, W. Rider and W. Stein, "Setting the default to reproducible: Reproducibility in computational and experimental mathematics," Jan 2013, available at `http://www.davidhbailey.com/dhbpapers/icerm-report.pdf`.

Numerical reliability

Many applications routinely use either 32-bit or 64-bit IEEE arithmetic, and employ fairly simple algorithms, assuming that all is well. But problems can arise:

1. Large-scale, highly parallel simulations, running on systems with hundreds of thousands or millions of processors — numerical sensitivities are greatly magnified.
2. Certain applications with highly ill-conditioned linear systems.
3. Large summations, especially those involving $+/-$ terms and cancellations.
4. Long-time, iterative simulations (such as molecular dynamics or climate models).
5. Computations to resolve small-scale phenomena.
6. Studies in computational physics or experimental mathematics often require huge precision levels.

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

Analysis of collisions at the Large Hadron Collider

- The 2012 discovery of the Higgs boson at the ATLAS experiment in the LHC relied crucially on the ability to track charged particles with exquisite precision (10 microns over a 10m length) and high reliability (over 99% of roughly 1000 charged particles per collision correctly identified).
- Software: 5 millions line of C++ and python code, developed by roughly 2000 physicists and engineers over 15 years.
- Recently, in an attempt to speed up the calculation, researchers found that merely changing the underlying math library resulted in some collisions being missed or misidentified.

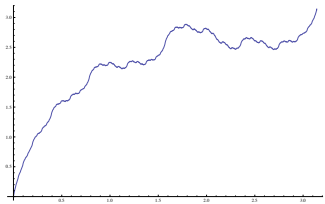
Questions:

- How serious are these numerical difficulties?
- How can they be tracked down?
- How can the library be maintained, producing numerically reliable results?

Enhancing reproducibility with selective usage of high-precision arithmetic

Problem: Find the arc length of the irregular function $g(x) = x + \sum_{0 \le k \le 10} 2^{-k} \sin(2^k x)$, over the interval $(0, \pi)$ (using 10^7 abscissa points).

- If this computation is done with ordinary double precision arithmetic, the calculation takes 2.59 seconds and yields the result 7.073157029008510.
- If it is done using all double-double arithmetic (31-digit accuracy), it takes 47.39 seconds seconds and yields the result 7.073157029007832.
- But if only the summation is changed to double-double, the result is identical to the double-double result (to 15 digits), yet the computation only takes 3.47 seconds.



Graph of $g(x) = x + \sum_{0 \le k \le 10} 2^{-k} \sin(2^k x)$, over $(0, \pi)$.

U.C. Berkeley's CORVETTE project and the "Precimonious" tool

Objective: Develop software facilities to find and ameliorate numerical anomalies in large-scale computations:

- Facilities to test the level of numerical accuracy required for an application.
- Facilities to delimit the portions of code that are inaccurate.
- Facilities to search the space of possible code modifications.
- Facilities to repair numerical difficulties, including usage of high-precision arithmetic.
- Facilities to navigate through a hierarchy of precision levels (32-bit, 64-bit, 80-bit or higher as needed).

The current version of this tool is known as "Precimonious."

- C. Rubio-Gonzalez, C. Nguyen, H. D. Nguyen, J. Demmel, W. Kahan, K. Sen, D. H. Bailey and C. Iancu, "Precimonious: Tuning assistant for floating-point precision," manuscript, May 2013.

Innocuous example where high precision is required for reproducible results

Problem: Find a polynomial to fit the data $(1, 1048579, 16777489, 84941299, 268501249, 655751251, 1360635409, 2523398179, 4311748609)$ for arguments $0, 1, \cdots, 8$. The usual approach is to solve the linear system:

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

A 64-bit computation (e.g., using Matlab, Linpack or LAPACK) fails to find the correct polynomial in this instance, even if one rounds results to nearest integer.

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

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

Innocuous example, continued

The result on the previous page can be obtained with double precision using Lagrange interpolation or the Demmel-Koev algorithm. But few scientists, outside of expert numerical analysts, are aware of these schemes.

Besides, even these schemes fail for higher-degree problems. For example:
$(1, 134217731, 8589938753, 97845255883, 549772595201,$
$2097396156251, 6264239146561, 15804422886323, 35253091827713,$
$71611233653971, 135217729000001, 240913322581691, 409688091758593)$
is generated by:

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

Neither the Lagrange or Demmel-Koev algorithms get the right answer with double precision. In contrast, a straightforward Linpack scheme, implemented with double-double arithmetic, works fine for this and a wide range of similar problems.

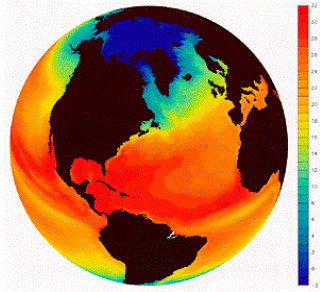
Other applications where high-precision arithmetic is useful or essential

1. Planetary orbit calculations (32 digits).
2. Supernova simulations (32–64 digits).
3. Certain components of climate modeling (32 digits).
4. Coulomb n-body atomic system simulations (32–120 digits).
5. Schrodinger solutions for lithium and helium atoms (32 digits).
6. Electromagnetic scattering theory (32–100 digits).
7. Scattering amplitudes of fundamental particles (32 digits).
8. Discrete dynamical systems (32 digits).
9. Theory of nonlinear oscillators (64 digits).
10. The Taylor algorithm for ODEs (100–600 digits).
11. Ising integrals from mathematical physics (100–1000 digits).
12. Problems in experimental mathematics (100–50,000 digits).

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

Climate modeling

- 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 found that by using double-double arithmetic in two key inner loops of an atmospheric modeling code, 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.

Using high-precision arithmetic to discover new mathematical relations

Very high-precision arithmetic is essential to obtain reproducible results in experimental mathematics.

Methodology:

1. Compute various mathematical entities (limits, infinite series sums, definite integrals, etc.) to high precision, typically 100–10,000 digits.
2. Use an algorithm such as "PSLQ" to recognize these numerical values in terms of well-known mathematical constants.
3. When results are found experimentally, seek 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*

Ising integrals from mathematical physics

We recently applied this methodology to study three classes of integrals, two of which arise in mathematical physics:

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

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

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

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

- 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.

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 $K_0(t)$:

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

We computed 1000-digit values of these integrals using this formula and tanh-sinh quadrature, and observed that the values approach a limit. For example:

$$C_{1024} = 0.630473503374386796122040192710878904354587078 7\ldots$$

What is this number? We copied the digits into the online Inverse Symbolic Calculator (`http://carma-lx1.newcastle.edu.au:8087`). The result was:

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

where $\gamma = 0.57714\ldots$ is Euler's constant. This result and numerous related results were subsequently proven.

Other Ising integral evaluations found using PSLQ and 500-digit arithmetic

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

where $\zeta(x)$ is the Riemann zeta function and $\mathrm{Li}_n(x)$ is the polylogarithm function.

Algebraic numbers in Poisson potential functions and lattice sums

Lattice sums arising from the Poisson equation have been studied widely in mathematical physics and also in image processing. 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 $(1, \alpha, \alpha^2, \cdots, \alpha^n)$ as input, where $\alpha = \exp(8\pi\phi_2(x,y))$. PSLQ returned the vector of integer coefficients $(a_0, a_1, a_2, \cdot, a_n)$ as output.

- D. H. Bailey, J. M. Borwein, R. E. Crandall and J. Zucker, "Lattice sums arising from the Poisson equation," *Journal of Physics A: Mathematical and Theoretical*, vol. 46 (2013), pg. 115201.

Samples of minimal polynomials found by PSLQ

k	Minimal polynomial for $\exp(8\pi\phi_2(1/k, 1/k))$
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 corresponding to $x = y = 1/32$ has degree 128, with individual coefficients ranging from 1 to over 10^{56}. This PSLQ computation required 10,000-digit precision.

Other polynomials required up to 50,000-digit precision.

Reproducibility in symbolic computing

Mathematicians, physicists and many others utilize symbolic computing packages such as *Maple* and *Mathematica* to perform symbolic computing. But, like all software, they have bugs.

For example, consider

$$W_2 = \int_0^1 \int_0^1 \left| e^{2\pi i x} + e^{2\pi i y} \right| \mathrm{d}x \, \mathrm{d}y, \tag{1}$$

The currently available versions of *Maple* and *Mathematica* both report that this integral is zero, despite the obvious fact that it must be strictly positive. The correct result is $4/\pi = 1.2732395447351626862\ldots$

Numerous other examples could be cited. Sometimes the bugs are not even consistent — correct results in some runs, incorrect results in others.

Potential sources of computational errors

- User programming errors.
- Bugs in symbolic or numerical software packages.
- Bugs in compilers.
- Bugs in operating systems or other system software.
- Hardware errors — memory errors, processor errors, data communication errors. In current exascale and petascale systems, such errors must be expected, not ignored.
- Errors in data storage media, or in communication to/from such media.

So why should anyone believe the results of any computation?

Methods to enhance reliability of computed results

1. Perform a computation using two different algorithms, then compare final results.
   - This is widely employed, say in computing digits of π: if the results of two such calculation are the same, except for a few trailing digits, then this is strong evidence that both calculations are almost certainly correct.
2. Perform a computation using two different software packages (e.g., *Maple and Mathematica*).
3. Perform a computation with two teams of programmers on different systems — climate models are sometimes validated in this way.
4. Perform a computation using two different levels of numeric precision, to see how many digits agree.
5. Perform internal validity checks:
   - For example, if a matrix and its inverse are both advanced in the computation, periodically multiply them together to ensure that the result is still the identity, to within acceptable numerical error.

Summary

- The issue of reproducibility has recently emerged in scientific computing, just as it has in the pharmaceutical industry, finance and others.
- The field of scientific computing is woefully behind other fields in reproducible practices — making careful records of runs, saving source code, analyzing results, etc. New tools may help.
- The issue of questionable performance reporting practices is again emerging, particularly with the deployment of new advanced architectures.
- Numerical reproducibility is of particular concern, in light of huge petascale and exadscale calculations being attempted.
- High-precision arithmetic may alleviate some numerical problems. Other applications require tens, hundreds or even thousands of digits.
- All studies involving computations need to present evidence of internal checks or replications to ensure that the results are reliable.

The credibility of high-performance scientific computing is at stake. Have we learned the lessons of history?