New computations of Poisson polynomials

David H. Bailey
http://www.davidhbailey.com
Lawrence Berkeley National Lab (retired)
University of California, Davis

January 11, 2023

Polynomials arising from the Poisson potential function

The Poisson potential function appears in numerous applied math contexts, ranging from mathematical physics to sharpening iPhone images:

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

In a 2013 study, colleagues and I numerically discovered and then proved the intriguing fact that for rational (x, y), the 2-D Poisson potential function $\phi_2(x, y)$ satisfies

$$\phi_2(x,y) = \frac{1}{\pi} \log \alpha$$

where α is algebraic, i.e., the root of an integer polynomial of some degree m.

By computing high-precision numerical values of $\phi_2(x, y)$ for various specific rational x and y, and applying variants of the PSLQ program, we were able to produce the explicit minimal polynomials for α in several simple specific cases.

- 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), 115201.

The PSLQ integer relation algorithm

Let $X = (x_k)$ be an $(m+1)$-long real or complex vector. An integer relation algorithm finds a nontrivial integer vector $A = (a_k)$ such that

$$a_0 x_0 + a_1 x_1 + \cdots + a_m x_m \quad = \quad 0.$$

- The PSLQ algorithm and the multipair PSLQ algorithm are commonly used for integer relation detection. Variants of the LLL algorithm can also be used.
- Integer relation detection (by any algorithm) requires very high precision: at least $(m+1) \cdot \max_k \log_{10} |a_k|$ digits, both in the input data and the algorithm.
- Fast variations of PSLQ and multipair PSLQ utilize two, three or even four levels of precision, doing as much computation as possible with only double precision, and switching to higher levels only when needed.

- D. H. Bailey and D. J. Broadhurst, "Parallel integer relation detection: Techniques and applications," *Mathematics of Computation*, vol. 70, no. 236 (Oct 2000), 1719–1736.

Numerically finding minimal polynomials using PSLQ

PSLQ can be used to recognize a computed numerical value as the root of an integer polynomial of degree m. Example: The following constant is suspected to be an algebraic number:

$$\alpha = 1.23268891306144344533147286961125564706898882454793057605763468477 8\ldots$$

What is its minimal polynomial?

Method: Compute the vector $(1, \alpha, \alpha^2, \cdots, \alpha^m)$ for $m = 30$, then input this vector to PSLQ or some other integer relation algorithm.

Answer (using 250-digit arithmetic):

$$\begin{aligned} 0 = {} & 697 - 1440\alpha - 20520\alpha^2 - 98280\alpha^3 - 102060\alpha^4 - 1458\alpha^5 + 80\alpha^6 - 43920\alpha^7 \\ & + 538380\alpha^8 - 336420\alpha^9 + 1215\alpha^{10} - 80\alpha^{12} - 56160\alpha^{13} - 135540\alpha^{14} - 540\alpha^{15} \\ & + 40\alpha^{18} - 7380\alpha^{19} + 135\alpha^{20} - 10\alpha^{24} - 18\alpha^{25} + \alpha^{30} \end{aligned}$$

Some initial Poisson polynomials found using PSLQ

s	Minimal polynomial corresponding to $x = y = 1/s$:
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	$1 - 216\alpha + 860\alpha^2 - 744\alpha^3 + 454\alpha^4 - 744\alpha^5 + 860\alpha^6 - 216\alpha^7 + \alpha^8$

These computations were very expensive. The case $x = y = 1/32$, for instance, required 10,000-digit arithmetic and ran for 45 hours. Some larger runs, using even higher precision, failed.

Help! More powerful computational tools are required.

Kimberley's formula for the degree of the polynomial

Based on these preliminary results, Jason Kimberley of the University of Newcastle, Australia observed that the degree $m(s)$ of the minimal polynomial associated with the case $x = y = 1/s$ appears to be given by the following:

Set $m(2) = 1/2$. Otherwise for primes p congruent to 1 mod 4, set $m(p) = \mathrm{int}^2(p/2)$, where int denotes greatest integer, and for primes p congruent to 3 mod 4, set $m(p) = \mathrm{int}\,(p/2)(\mathrm{int}\,(p/2) + 1)$. Then for any other positive integer s whose prime factorization is $s = p_1^{e_1} p_2^{e_2} \cdots p_r^{e_r}$,

$$m(s) = 4^{r-1} \prod_{i=1}^{r} p_i^{2(e_i-1)}\, m(p_i).$$

Does Kimberley's formula hold for larger s?

What is the true mathematical connection between the pair of rationals (x, y) and the algebraic number α?

2015: Improvements to the Poisson polynomial computation program

1. A new multiprecision package: 3X faster than before, suitable for use in a multi-threaded parallel environment.
2. A new 3-level multipair PSLQ program, up to 5X faster:
   a. Double precision (appr. 15 digit accuracy).
   b. Medium multiprecision: varies from 100 digits to 1200 digits on Poisson problems.
   c. Full multiprecision: varies from 2,500 digits to 50,000 digits on Poisson problems.
3. Faster hardware: a new 8-core MacPro system with Intel processor.

Key breakthrough: A fast algorithm to compute $\phi_2(x, y)$

The original formula for $\phi_2(x, y)$ converges *much* too slowly for numerical evaluation. But this formula, found by Jonathan Borwein, converges very rapidly:

$$\phi_2(x, y) = \frac{1}{2\pi} \log \left| \frac{\theta_2(z, q)\theta_4(z, q)}{\theta_1(z, q)\theta_3(z, q)} \right|,$$

where $q = e^{-\pi}$ and $z = \frac{\pi}{2}(y + ix)$, and where

$$\theta_1(z, q) = 2 \sum_{k=1}^{\infty} (-1)^{k-1} q^{(2k-1)^2/4} \sin((2k-1)z),$$

$$\theta_2(z, q) = 2 \sum_{k=1}^{\infty} q^{(2k-1)^2/4} \cos((2k-1)z),$$

$$\theta_3(z, q) = 1 + 2 \sum_{k=1}^{\infty} q^{k^2} \cos(2kz),$$

$$\theta_4(z, q) = 1 + 2 \sum_{k=1}^{\infty} (-1)^k q^{k^2} \cos(2kz).$$

High-level computational algorithm

1. Given rationals $x = p/s$ and $y = q/s$, select a conjectured minimal polynomial degree $m(s)$ (using Kimberley's formula) and other parameters for the run.
2. Calculate $\phi_2(x, y)$ to P_2-digit precision using the formulas from the previous viewgraph. When done, calculate $\alpha = \exp(8\pi\phi_2(x, y))$ and generate the $(m + 1)$-long vector $X = (1, \alpha, \alpha^2, \cdots, \alpha^m)$, to P_2-digit precision.
3. Apply the three-level multipair PSLQ algorithm to X.
4. If no numerically significant relation is found, try again with a larger degree m or higher precision P_2. If a relation is found, employ the polynomial factorization facilities in *Mathematica* and *Maple* to ensure that the polynomial is irreducible.

192-degree minimal polynomial found by multipair PSLQ for $x = y = 1/35$

This distinctive semi-elliptical pattern is strong evidence that the result is correct.

Selected runs (degrees, precision, timings, etc.) for $x = y = 1/s$

s	m	$\log_{10}(D)$	P_1	P_2	N	M	C	T (sec.)	$C \cdot T$ (sec.)
20	32	-463.84	160	700	1967	0.81	1	$2.19 \cdot 10^0$	$2.19 \cdot 10^0$
24	64	-1883.78	320	2200	9297	11.33	1	$7.73 \cdot 10^1$	$7.73 \cdot 10^1$
30	64	-1868.01	350	2300	9064	11.33	1	$1.02 \cdot 10^2$	$1.02 \cdot 10^2$
32	128	-7577.07	650	8200	45893	168.20	1	$5.13 \cdot 10^3$	$5.13 \cdot 10^3$
34	128	-7574.93	650	8200	45914	168.20	1	$5.16 \cdot 10^3$	$5.16 \cdot 10^3$
36	144	-9570.86	750	10300	62282	267.10	1	$9.54 \cdot 10^3$	$9.54 \cdot 10^3$
37	324	-48431.32	1650	51000	931254	6579.66	16	$4.84 \cdot 10^5$	$7.74 \cdot 10^6$
38	180	-14951.64	900	16000	120984	642.98	1	$3.88 \cdot 10^4$	$3.88 \cdot 10^4$
39	288	-38330.14	1450	40000	667153	4124.24	16	$2.68 \cdot 10^5$	$4.29 \cdot 10^6$
40	128	-7580.00	650	8200	45655	168.20	1	$5.02 \cdot 10^3$	$5.02 \cdot 10^3$
42	192	-16993.99	1000	18000	150364	829.41	8	$1.57 \cdot 10^4$	$1.26 \cdot 10^5$
44	240	-26604.14	1200	28000	323762	2003.33	8	$7.43 \cdot 10^4$	$5.94 \cdot 10^5$
45	288	-38315.08	1450	40000	660001	4124.24	16	$2.09 \cdot 10^5$	$3.35 \cdot 10^6$
46	264	-32036.34	1350	34000	476902	2921.57	16	$1.06 \cdot 10^5$	$1.70 \cdot 10^6$
48	256	-30248.55	1350	32000	415316	2586.39	16	$8.98 \cdot 10^4$	$1.44 \cdot 10^6$
50	200	-18421.18	1000	20000	168947	974.44	8	$2.12 \cdot 10^4$	$1.69 \cdot 10^5$

s = denominator; m = degree; D = detection level; P_1 = medium precision; P_2 = full precision; N = number of iterations; M = Mbytes; C = cores; T = wall clock time; $C \cdot T$ = total core-seconds.

Palindromic polynomials

From our results, in the case $(1/s, 1/s)$ where s is even, the resulting polynomial is always palindromic ($a_k = a_{m-k}$). For instance, when $s = 16$,

$$
\begin{aligned}
p_{16}(\alpha) = 1 &- 1376\alpha^1 - 12560\alpha^2 - 3550496\alpha^3 + 81241720\alpha^4 - 169589984\alpha^5 \\
&+ 1334964944\alpha^6 - 24307725984\alpha^7 + 238934926108\alpha^8 - 1043027124704\alpha^9 \\
&+ 2328675366384\alpha^{10} - 3219896325280\alpha^{11} + 4238551472456\alpha^{12} \\
&- 10247414430048\alpha^{13} + 28552105805904\alpha^{14} - 55832851687968\alpha^{15} \\
&+ 70020268309062\alpha^{16} \\
&- 55832851687968\alpha^{17} + 28552105805904\alpha^{18} - 10247414430048\alpha^{19} \\
&+ 4238551472456\alpha^{20} - 3219896325280\alpha^{21} + 2328675366384\alpha^{22} \\
&- 1043027124704\alpha^{23} + 238934926108\alpha^{24} - 24307725984\alpha^{25} + 1334964944\alpha^{26} \\
&- 169589984\alpha^{27} + 81241720\alpha^{28} - 3550496\alpha^{29} - 12560\alpha^{30} - 1376\alpha^{31} + \alpha^{32}
\end{aligned}
$$

Nitya Mani, a student at Stanford University, pointed out that if α is a root of a palindromic polynomial such as this, then $\alpha + 1/\alpha$ is a root of a transformed polynomial of half the degree. This fact can be used to greatly accelerate the computation of Poisson polynomials when s is an even integer.

2016: A proof of Kimberley formula for the case $(1/s, 1/s)$

Some observations from the polynomials produced by the program:

- The algebraic number α_s is the *largest* real root of the associated polynomial.
- The polynomial has $\varphi(s)$ real roots, where φ is the Euler totient function.
- $\sqrt{-\alpha_s}$ is the largest purely imaginary root of $\psi_s(x, 1)$, where ψ is a sequence of polynomials defined in a 2010 paper by Savin and Quarfoot of the Univ. of Utah.

The Savin-Quarfoot paper was found by doing an Internet search for "387221579866," which is a coefficient of the polynomial for the case $(1/11, 1/11)$.

These observations ultimately led to a proof, by Watson Ladd of U.C. Berkeley, of Kimberley's formula in the case $x = y = 1/s$.

- D. H. Bailey, J. M. Borwein, J. Kimberley and W. Ladd, "Computer discovery and analysis of large Poisson polynomials," *Experimental Mathematics*, 27 Aug 2016, vol. 26, 349–363, `https://www.davidhbailey.com/dhbpapers/poisson-res.pdf`.

What about the many cases with $x \neq y$?

All of the computations and results mentioned above are for the case $x = y = 1/s$ for some positive integer s.

What about rationals $x = p/s$, $y = q/s$, with $p \neq q$? Initial results indicated that Kimberley's formula does not hold for these more general rationals.

Is there a generalization of Kimberley's formula that holds in these cases? To numerically address this more general problem requires many times more computation than for the $x = y = 1/s$ cases.

A familiar refrain: Help! More powerful computational tools are required.

2022: New computational tools for the Poisson polynomial problem

1. A new multiprecision package, based on integer arithmetic, that is approximately 3X faster than before. As an option, this software can also use the MPFR and GMP packages for low-level operations.
2. A new 4-level multipair PSLQ program, up to 1.5X faster:
   a. Double precision (approx. 15 digit accuracy).
   b. Quad precision (approx. 32 digit accuracy).
   c. Medium multiprecision: varies from 100 digits to 1200 digits on Poisson problems.
   d. Full multiprecision: varies from 2,500 digits to 50,000 digits on Poisson problems.
3. The code does as much computation as possible in double precision; it shifts to higher precision when double precision is exhausted.
4. Faster hardware: It is currently running on a 10-core MacStudio, with Apple's new low-power M1 Pro processor (thus helping the author avoid personal bankruptcy from electric bills).

December 2022: New computer runs

- The new software has been used to compute the minimal polynomials for the entire set of cases $(p/s, q/s)$, where $1 \le p \le q < s/2$, where s ranges from 10 to 42 (except $s = 37, 39, 41$), and where $\gcd(p, q, s) = 1$. A total of 2024 individual cases have been processed.
- These runs required approximately four months run time on all 10 cores of the system, completing a few days ago.

Kimberley's formula is not valid for these more general cases, but the following modification of Kimberley's formula appears to hold:

1. For the cases $x = y = 1/s$, Kimberley's formula holds.
2. For the cases $x = p/s$, $y = q/s$ with s odd, Kimberley's formula holds (except for a few cases when the correct degree is smaller by a power of two factor).
3. For the cases $x = p/s$, $y = q/s$, with s even and both p and q odd, Kimberley's formula holds (with a few exceptions as above).
4. For the cases $x = p/s$, $y = q/s$, with s even and one of p or q is even, the correct degree is twice Kimberley's formula (with a few exceptions as above).

Conclusions

- New computations of Poisson polynomials have been completed, covering the entire set of cases $(p/s, q/s)$, where $1 \le p \le q < s/2$, where s ranges from 10 to 42 (except $s = 37, 39, 41$), and where $\gcd(p, q, s) = 1$. A total of 2024 individual cases have been processed.
- The resulting minimal polynomials have shown that Kimberley's formula does not hold for these more general cases. However, a modification of Kimberley's formula appears to hold.
- We currently have no idea how this modified rule can be proved.
- The three-dimensional and higher-dimensional Poisson formulas have not yet been explored computationally; these cases likely will require even more computation.
- A familiar refrain: Help! More powerful computational tools are required.

This talk is available at
http://www.davidhbailey.com/dhbtalks/dhb-wcnt-2022.pdf.