PSLQ: An Algorithm to Discover Integer Relations

David H. Bailey* and J.M. Borwein†

May 14, 2020

1. Introduction. Let $x = (x_1, x_2, \cdots, x_n)$ be a vector of real or complex numbers. x is said to possess an integer relation if there exist integers a_i, not all zero, such that

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

By an *integer relation algorithm*, we mean a practical computational scheme that can recover the vector of integers a_i, if it exists, or can produce bounds within which no integer relation exists. As we will see in the examples below, an integer relation algorithm can be used to recognize a computed constant in terms of a formula involving known constants, or to discover an underlying relation between quantities that can be computed to high precision.

At the present time, the most widely used algorithm for integer relation detection is the "PSLQ" algorithm of mathematician-sculptor Helaman Ferguson [11, 4], although the "LLL" algorithm is also used for this purpose. One detailed comparison of these two methods found that PSLQ appears to be more numerically stable than LLL, in the sense that PSLQ reliably finds a relation, beginning nearly at the minimal precision level for the relation, whereas LLL sometimes finds a relation at one level but fails at a somewhat higher level [10]. This study also found that tuned implementations of PSLQ (which select multiple pairs of indices, and which employ two or three levels of precision [4]) are significantly more efficient than typical implementations of LLL. Additional research may further cast light on the relative merits of these two schemes. In the following, though, we will focus on PSLQ.

PSLQ operates by constructing a sequence of integer-valued matrices B_n that reduces the vector $y = xB_n$, 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 vector y abruptly 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

*Lawrence Berkeley National Laboratory, Berkeley, CA 94720, dhbailey@lbl.gov. Supported in part by the Director, Office of Computational and Technology Research, Division of Mathematical, Information, and Computational Sciences of the U.S. Department of Energy, under contract number DE-AC02-05CH11231.

†School of Mathematical and Physical Sciences, University of Newcastle, Callaghan, NSW 2308, Australia jonathan.borwein@newcastle.edu.au

the relation is real and not merely a numerical artifact. A drop of 20 or more orders of magnitude almost always indicates a real relation (see Figure 1).

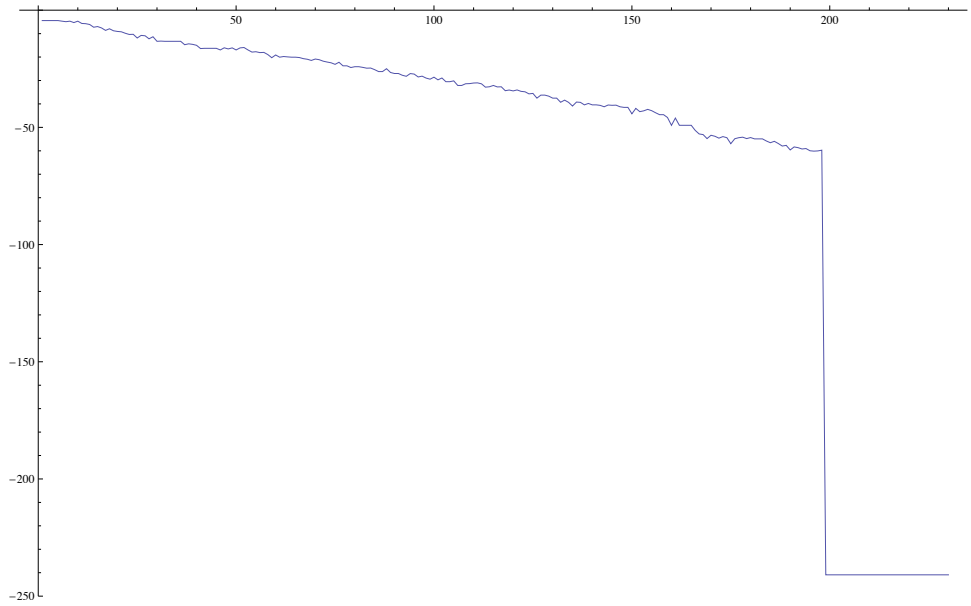

Figure 1: $\log_{10} \min_k |y_k|$ versus iteration number in a typical PSLQ run

Very high precision arithmetic must be used with PSLQ or any other integer relation scheme. If one wishes to recover a relation of length n, with coefficients of maximum size d digits, then the input vector x must be specified to at least nd digits, and one must employ nd-digit floating-point arithmetic. *Maple* and *Mathematica* include multiple precision arithmetic facilities and *Maple* ships with a full implementation of PSLQ. One may also use any of several freeware multiprecision software packages, for example the ARPREC package by the first author and colleagues at LBNL [7]. In the remaining sections we describe various representative applications of PSLQ. More detail about these examples is given in [8] and the references therein.

2. Finding Algebraic Relations Using PSLQ. One immediate and impressive application of PSLQ in the field of mathematical number theory is to determine whether or not a given constant α, whose value can be computed to high precision, is algebraic of some degree n or less. This can be done by first computing the vector $x = (1, \alpha, \alpha^2, \cdots, \alpha^n)$ to high precision and then applying an integer relation algorithm to the resulting $(n+1)$-long vector. If a relation is found for x, then this relation vector is precisely the set of integer coefficients of a polynomial satisfied by α (to the precision specified).

One of the first results of this sort was the identification of the constant $\hat{B}_3 = 3.54409035955\ldots$. $\hat{B}_3$ is the third bifurcation point of the logistic map $x_{k+1} = rx_k(1-x_k)$, which exhibits period doubling shortly before the onset of chaos. To be precise, $\hat{B}_3$ is the smallest value of the parameter r such that successive iterates x_k exhibit eight-way periodicity instead of four-way periodicity. $\hat{B}_3$ can be computed to arbitrarily high precision by means of an iterative algorithm [6]. When PSLQ is applied to the 13-long vector

$(1, \hat{B}_3, \hat{B}_3^2, \hat{B}_3^3, \cdots, \hat{B}_3^{12})$, one obtains the result that $\hat{B}_3$ is a root of the polynomial

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

Recently, $\hat{B}_4 = 3.564407268705\cdots$, the fourth bifurcation point of the logistic map, was identified using PSLQ by British physicist David Broadhurst [4]. Some conjectural reasoning had suggested that $\hat{B}_4$ might satisfy a 240-degree polynomial, and some further analysis had suggested that the constant $\alpha = -\hat{B}_4(\hat{B}_4 - 2)$ might satisfy a 120-degree polynomial. In order to test this hypothesis, Broadhurst applied a PSLQ program to the 121-long vector $(1, \alpha, \alpha^2, \cdots, \alpha^{120})$. Indeed, a relation was found, though 10,000-digit arithmetic was required. The recovered integer coefficients descend monotonically from $257^{30} \approx 1.986 \times 10^{72}$ to 1. This was subsequently proven using Groebner bases [6].

3. A New Formula for Pi. Through the centuries mathematicians have assumed that there is no shortcut to computing digits of π beginning at some position n. Thus, it came as no small surprise when such an algorithm was discovered in 1996 [3]. In particular, this simple scheme allows one to compute binary or hexadecimal (base-16) digits of π starting at an arbitrary position, without computing any of the preceding digits. For instance, the one millionth hex digit of π can be computed in this manner on a current-generation personal computer in only about 10 seconds run time. This scheme is based on the following new formula, which was discovered in 1996 using PSLQ:

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

Since 1996, numerous formulas of this same type have been found for various constants [1, 8]. For example, a similar formula was found that permits arbitrary-position binary digits of π^2 to be calculated; here is a formula for π^2 that permits arbitrary *ternary* (base-3) digits of π^2 to be calculated:

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

Sadly, it has recently been proven that there is no formula of this type for π itself in other than a binary base [8, Ch. 3].

4. Identification of Multiple Zeta Constants A large number of results has been found over the last 15 years using PSLQ in the course of research on *multiple zeta sums*, such as those shown in Table 1. After computing the numerical values of these constants, a PSLQ program was used to determine if a given constant satisfied an identity of a conjectured form. These efforts produced numerous empirical evaluations and suggested

general results [2, 9]. Eventually, elegant proofs were found for many of these specific and general results. Three examples of PSLQ results that were subsequently proven are given in Table 1. In the table, $\zeta(t) = \sum_{j=1}^{\infty} j^{-t}$ is the Riemann zeta function, and $\mathrm{Li}_n(x) = \sum_{j=1}^{\infty} x^j j^{-n}$ denotes the polylogarithm function.

$$\begin{array}{|l|}\hline \sum_{k=1}^{\infty} \left(1+\frac{1}{2}+\cdots+\frac{1}{k}\right)^2 (k+1)^{-4} \;=\; \frac{37}{22680}\pi^6 - \zeta^2(3) \\ \sum_{k=1}^{\infty} \left(1+\frac{1}{2}+\cdots+\frac{1}{k}\right)^3 (k+1)^{-6} \;=\; \zeta^3(3) + \frac{197}{24}\zeta(9) + \frac{1}{2}\pi^2\zeta(7) \\ \qquad\qquad - \frac{11}{120}\pi^4\zeta(5) - \frac{37}{7560}\pi^6\zeta(3) \\ \sum_{k=1}^{\infty} \left(1-\frac{1}{2}+\cdots+(-1)^{k+1}\frac{1}{k}\right)^2 (k+1)^{-3} \;=\; 4\,\mathrm{Li}_5(\tfrac{1}{2}) - \frac{1}{30}\ln^5(2) - \frac{17}{32}\zeta(5) \\ \qquad - \frac{11}{720}\pi^4\ln(2) + \frac{7}{4}\zeta(3)\ln^2(2) + \frac{1}{18}\pi^2\ln^3(2) - \frac{1}{8}\pi^2\zeta(3) \\ \hline \end{array}$$

Table 1: Some multiple zeta identities found by PSLQ

5. Ising integrals. One particularly fruitful application of these methods is the evaluation of definite integrals, such as those that arise in mathematical physics. For example, recently the present authors, together with Richard Crandall, investigated three classes of n-fold integrals, which arise in Ising theory and also (in some cases) in quantum field theory:

$$\begin{aligned} C_n &:= \frac{4}{n!}\int_0^\infty \cdots \int_0^\infty \frac{1}{\left(\sum_{j=1}^n (u_j + 1/u_j)\right)^2} \frac{\mathrm{d}u_1}{u_1} \cdots \frac{\mathrm{d}u_n}{u_n} \\ D_n &:= \frac{4}{n!}\int_0^\infty \cdots \int_0^\infty \frac{\prod_{i<j}\left(\frac{u_i-u_j}{u_i+u_j}\right)^2}{\left(\sum_{j=1}^n (u_j + 1/u_j)\right)^2} \frac{\mathrm{d}u_1}{u_1} \cdots \frac{\mathrm{d}u_n}{u_n} \\ E_n &= 2\int_0^1 \cdots \int_0^1 \left(\prod_{1\le j<k\le n} \frac{u_k-u_j}{u_k+u_j}\right)^2 \mathrm{d}t_2\, dt_3 \cdots \mathrm{d}t_n, \end{aligned}$$

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

Computing high-precision values of n-fold integrals such as this is very difficult for n greater than three or four. But we found a simple substitution that reduces the C integrals to 1-dimensional integrals:

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

where $K(t)$ is the modified Bessel function. In this form, we were able to evaluate C_n to over 1000-digit accuracy, for n up to 1024. With these numerical values in hand, we quickly found that $C_1 = 2,\ C_2 = 1,\ C_3 = L_{-3}(2) = \sum_{n\ge 0}(1/(3n+1)^2 - 1/(3n+2)^2)$, and $C_4 = 7\zeta(3)/12$. We also discovered numerically that

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

where γ is Euler's constant. Further computation established results such as:

$$D_2 = 1/3, \quad D_3 = 8 + 4\pi^2/3 - 27\,\mathrm{L}_{-3}(2), \quad D_4 = 4\pi^2/9 - 1/6 - 7\zeta(3)/2$$

and

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

The E_5 integral was found after transforming its defining 5-fold integral representation into an extremely complicated 3-fold integral. We then computed this 3-fold integral to 250-digit precision, by using a parallel quadrature program implemented on 1024 CPUs of a parallel computer system, and then discovered the above-listed experimental identity by using PSLQ. This identity has a question mark because, unlike the others mentioned in this paper, we do not yet have a formal proof. Nonetheless it is established numerically at least 180 orders of magnitude beyond the level of numerical "chance," and so we are quite confident in the result. Such confidence is typically obtainable if the constants involved can be computed to sufficiently high precision. Sometimes as with C_n this is relatively easy. In other cases, such as E_5, it involves much more labor.

6. Research questions. In spite of these and other successes, there is considerable need for even more efficient schemes for both integer relation detection and numerical integration, especially the evaluation of multi-dimensional integrals. With regards to PSLQ, there is interest in extending PSLQ to more general number fields, such as quadratic number fields. Hopefully future research will yield better schemes that will in turn produce more results of interest in mathematics and mathematical physics.

References

[1] David H. Bailey, "A Compendium of BBP-Type Formulas," available at `http://crd.lbl.gov/~dhbailey/dhbpapers/bbp-formulas.pdf`.

[2] David H. Bailey, Jonathan M. Borwein and Roland Girgensohn, "Experimental Evaluation of Euler Sums," *Experimental Mathematics*, vol. 4, no. 1, 1994, pg. 17–30.

[3] David H. Bailey, Peter B. Borwein and Simon Plouffe, "On The Rapid Computation of Various Polylogarithmic Constants," *Math. of Computation*, vol. 66, no. 218, 1997, pg. 903–913.

[4] David H. Bailey and David J. Broadhurst, "Parallel Integer Relation Detection: Techniques and Applications," *Math. of Computation*, vol. 70, no. 236, pg. 1719–1736.

[5] David 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.

[6] David H. Bailey, Jonathan M. Borwein, Vishal Kapoor and Eric Weisstein, "Ten Problems in Experimental Mathematics," *American Math. Monthly*, vol. 113, no. 6, 2006, pg. 481–409.

[7] David H. Bailey, Yozo Hida, Xiaoye S. Li and Brandon Thompson, "ARPREC: An Arbitrary Precision Computation Package," Sept. 2002, available at `http://crd.lbl.gov/~dhbailey/dhbpapers/arprec.pdf`.

[8] Jonathan M. Borwein and David H. Bailey, *Mathematics by Experiment*, AK Peters, 2003. Second edition, 2008. See also `http://www.experimentalmath.info`.

[9] David Borwein, Jonathan M. Borwein and Roland Girgensohn, "Explicit Evaluation of Euler Sums," *Proc. Edinburgh Math. Society*, vol. 38, 1995, pg. 277–294.

[10] Jonathan M. Borwein and Peter Lisoněk, "Applications of Integer Relation Algorithms," *Discrete Mathematics* (special issue for FPSAC 1997), vol. 217 (2000), pg. 65–82.

[11] Helaman R. P. Ferguson, David H. Bailey and Stephen Arno, "Analysis of PSLQ, An Integer Relation Finding Algorithm," *Math. of Computation*, vol. 68, 1999, pg. 351–369.