

Recognizing Numerical Constants  
David H. Bailey and Simon Plouffe  
December 15, 1995

Ref: *Canadian Mathematical Society*, vol. 20 (1997), pg. 73-88

**Abstract**

The advent of inexpensive, high-performance computers and new efficient algorithms have made possible the automatic recognition of numerically computed constants. In other words, techniques now exist for determining, within certain limits, whether a computed real or complex number can be written as a simple expression involving the classical constants of mathematics.

These techniques will be illustrated by discussing the recognition of Euler sum constants, and also the discovery of new formulas for  $\pi$  and other constants, formulas that permit individual digits to be extracted from their expansions.

Bailey: NASA Ames Research Center, Mail Stop T27A-1, Moffett Field, CA  
94035-1000, USA; [dbailey@nas.nasa.gov](mailto:dbailey@nas.nasa.gov).

Plouffe: Department of Mathematics and Statistics, Simon Fraser University,  
Burnaby, BC V5A 1S6, Canada; [plouffe@cecm.sfu.ca](mailto:plouffe@cecm.sfu.ca).

## 1. Introduction

The advent of inexpensive, high-performance computers and new efficient algorithms have made possible the automatic recognition of numerically computed constants. In other words, techniques now exist for determining, within certain limits, whether a computed real or complex number can be written as a simple expression involving the classical constants of mathematics.

The fundamental technique involved is that of finding integer relations. Let  $x = (x_1, x_2, \dots, x_n)$  be a vector of real numbers.  $x$  is said to possess an integer relation if there exist integers  $a_i$  not all zero such that  $a_1x_1 + a_2x_2 + \dots + a_nx_n = 0$ . By an integer relation algorithm, we mean an algorithm that is guaranteed (provided the computer implementation has sufficient numeric precision) to recover the vector of integers  $a_i$ , if it exists, or to produce bounds within which no integer relation can exist.

The problem of finding integer relations among a set of real numbers was first studied by Euclid, who gave an iterative algorithm (the Euclidean algorithm), which when applied to two real numbers, either terminates, yielding an exact relation, or produces an infinite sequence of approximate relations. The generalization of this problem for  $n > 2$  has been attempted by Euler, Jacobi, Poincare, Minkowski, Perron, Brun, Bernstein, among others. However, none of their algorithms has been proven to work for  $n > 3$ , and numerous counterexamples have been found.

The first integer relation algorithm with the desired properties mentioned above was discovered by Ferguson and Forcade in 1977 [14]. In the intervening years a number of other integer relation algorithms have been discovered, including the “LLL” algorithm [16], the “HJLS” algorithm [15], and the “PSOS” [6] algorithm.

## 2. The PSLQ Integer Relation Algorithm

In 1991 a new algorithm, known as “PSLQ” algorithm, was developed by Ferguson [12]. It appears to combine some of the best features separately possessed by previous algorithms, including fast run times, numerical stability, numerical efficiency (i.e. successfully recovering a relation when the input is known to only limited precision), and a

guaranteed completion in a polynomially bounded number of iterations. More recently a much simpler formulation of this algorithm was developed, and it has been extended to the complex number field [13]. This newer, simpler version of PSLQ can be stated as follows:

Let  $x$  be the  $n$ -long input real vector, and let  $\text{nint}$  denote the nearest integer function (for exact half-integer values, define  $\text{nint}$  to be the integer with greater absolute value). Let  $\gamma := \sqrt{4/3}$ . Then perform the following:

Initialize:

1. Set the  $n \times n$  matrices  $A$  and  $B$  to the identity.
2. For  $k := 1$  to  $n$ : compute  $s_k := \sqrt{\sum_{j=k}^n x_j^2}$ ; endfor. Set  $t = 1/s_1$ . For  $k := 1$  to  $n$ :  $y_k := tx_k$ ;  $s_k := ts_k$ ; endfor.
3. Compute the  $n \times (n - 1)$  matrix  $H$  as follows:

For  $i := 1$  to  $n$ : for  $j := i + 1$  to  $n - 1$ : set  $H_{ij} := 0$ ; endfor; if  $i \leq n - 1$  then set  $H_{ii} := s_{i+1}/s_i$ ; for  $j := 1$  to  $i - 1$ : set  $H_{ij} := -y_i y_j / (s_j s_{j+1})$ ; endfor; endfor.

4. Perform full reduction on  $H$ , simultaneously updating  $y$ ,  $A$  and  $B$ :

For  $i := 2$  to  $n$ : for  $j := i - 1$  to  $1$  step  $-1$ :  $t := \text{nint}(H_{ij}/H_{jj})$ ;  $y_j := y_j + ty_i$ ; for  $k := 1$  to  $j$ :  $H_{ik} := H_{ik} - tH_{jk}$ ; endfor; for  $k := 1$  to  $n$ :  $A_{ik} := A_{ik} - tA_{jk}$ ,  $B_{kj} := B_{kj} + tB_{ki}$ ; endfor; endfor; endfor.

Repeat until precision is exhausted or a relation has been detected:

1. Select  $m$  such that  $\gamma^i |H_{ii}|$  is maximal when  $i = m$ .
2. Exchange entries  $m$  and  $m+1$  of  $y$ , corresponding rows of  $A$  and  $H$ , and corresponding columns of  $B$ .
3. If  $m \leq n - 2$  then update  $H$  as follows:

Set  $t_0 := \sqrt{H_{mm}^2 + H_{m,m+1}^2}$ ,  $t_1 := H_{mm}/t_0$  and  $t_2 := H_{m,m+1}/t_0$ . Then for  $i := m$  to  $n$ :  $t_3 := H_{im}$ ;  $t_4 := H_{i,m+1}$ ;  $H_{im} := t_1 t_3 + t_2 t_4$ ;  $H_{i,m+1} := -t_2 t_3 + t_1 t_4$ ; endfor.

4. Perform block reduction on  $H$ , simultaneously updating  $y$ ,  $A$  and  $B$ :

For  $i := m + 1$  to  $n$ : for  $j := \min(i - 1, m + 1)$  to 1 step  $-1$ :  $t := \text{nint}(H_{ij}/H_{jj})$ ;  $y_j := y_j + ty_i$ ; for  $k := 1$  to  $j$ :  $H_{ik} := H_{ik} - tH_{jk}$ ; ; endfor; for  $k := 1$  to  $n$ :  $A_{ik} := A_{ik} - tA_{jk}$ ,  $B_{kj} := B_{kj} + tB_{ki}$ ; endfor; endfor; endfor.

5. Norm bound: Compute  $M := 1/\max_j |H_j|$ , where  $H_j$  denotes the  $j$ -th row of  $H$ . Then there can exist no relation vector whose Euclidean norm is less than  $M$ .

6. Termination test: If the largest entry of  $A$  exceeds the level of numeric precision used, then precision is exhausted. If the smallest entry of the  $y$  vector is less than the detection threshold, a relation has been detected and is given in the corresponding column of  $B$ .

With regards to the termination criteria in step 6, it sometimes happens that a relation is missed at the point of potential detection because the  $y$  entry is not quite as small as the detection threshold being used (the threshold is typically set to the “epsilon” of the precision level). When this happens, however, one will note that the ratio of the smallest and largest  $y$  vector entries is suddenly very small, provided sufficient numeric precision is being used. In a normal computer run using the PSLQ algorithm, prior to the detection of a relation, this ratio is seldom smaller than  $10^{-2}$ . Thus if this ratio suddenly decreases to a very small value, such as  $10^{-20}$ , then almost certainly a relation has been detected — one need only adjust the detection threshold for the algorithm to terminate properly and output the relation. When detection does occur, this ratio may be thought of as a “confidence level” of the detection.

As a general rule, one can expect to detect a relation of degree  $n$ , with coefficients of size  $10^m$ , provided that the input vector is known to somewhat greater than  $mn$  digit precision, and provided that computations are performed using at least this level of numeric precision.

### 3. Applications of the PSLQ Algorithm

There are a number of applications of integer relation detection algorithms in computational mathematics. One application is to analyze whether or not a given constant  $\alpha$ , 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  $x = (1, \alpha, \alpha^2, \dots, \alpha^n)$  to high precision and then applying an integer relation algorithm to the vector  $x$ . If a relation is found, this integer vector is precisely the set of coefficients of a polynomial satisfied by  $\alpha$ . Even if a relation is not found, the resulting bound means that  $\alpha$  cannot possibly be the root of a polynomial of degree  $n$ , with coefficients of size less than the established bound. Even negative results of this sort are often of interest.

We have performed several computations of this type [6]. These computations have established, for example, that if Euler's constant  $\gamma$  satisfies an integer polynomial of degree 50 or less, then the Euclidean norm of the coefficients must exceed  $7 \times 10^{17}$ . Computations of this sort have also been applied to study a certain conjecture regarding the Riemann zeta function. It is well known [10] that

$$\begin{aligned}\zeta(2) &= 3 \sum_{k=1}^{\infty} \frac{1}{k^2 \binom{2k}{k}} \\ \zeta(3) &= \frac{5}{2} \sum_{k=1}^{\infty} \frac{(-1)^{k-1}}{k^3 \binom{2k}{k}} \\ \zeta(4) &= \frac{36}{17} \sum_{k=1}^{\infty} \frac{1}{k^4 \binom{2k}{k}}\end{aligned}$$

These results have led some to suggest that

$$Z_5 = \zeta(5) / \sum_{k=1}^{\infty} \frac{(-1)^{k-1}}{k^5 \binom{2k}{k}}$$

might also be a simple rational or algebraic number. Unfortunately, integer relation calculations [3] have established that if  $Z_5$  satisfies a polynomial of degree 25 or less, then the Euclidean norm of the coefficients must exceed  $2 \times 10^{37}$ .

## 4. Euler Sums

In response to a letter from Goldbach, Euler considered sums of the form

$$\sum_{k=1}^{\infty} \left(1 + \frac{1}{2^m} + \cdots + \frac{1}{k^m}\right) (k+1)^{-n}.$$

Euler was able to give explicit values for certain of these sums in terms of the Riemann zeta function. For example, Euler found an explicit formula for the case  $m = 1, n \geq 2$ . Little progress has been made on this problem in the intervening years, although special cases of Euler's results have been rediscovered numerous times (see [7] for some references).

In April 1993, Enrico Au-Yeung, an undergraduate at the University of Waterloo, brought to the attention of Prof. Jonathan Borwein the curious fact that

$$\sum_{k=1}^{\infty} \left(1 + \frac{1}{2} + \cdots + \frac{1}{k}\right)^2 k^{-2} = 4.59987 \cdots \approx \frac{17}{4} \zeta(4) = \frac{17\pi^4}{360}$$

based on a computation to 500,000 terms. Borwein's reaction was to compute the value of this constant to a higher level of precision in order to dispel this conjecture. Surprisingly, a computation to 30 and later to 100 decimal digits still affirmed it.

Intrigued by this empirical result, numerical values were computed for several of these and similar sums, which were termed Euler sums. The resulting values were analyzed using integer relation searches. These efforts produced even more empirical evaluations, suggesting broad patterns and general conjectures. Ultimately proofs were found for many of these experimental results.

We will consider here only the following two classes of Euler sums. Other classes are studied in [4].

$$\begin{aligned} s_h(m, n) &= \sum_{k=1}^{\infty} \left(1 + \frac{1}{2} + \cdots + \frac{1}{k}\right)^m (k+1)^{-n} & m \geq 1, n \geq 2, \\ s_a(m, n) &= \sum_{k=1}^{\infty} \left(1 - \frac{1}{2} + \cdots + \frac{(-1)^{k+1}}{k}\right)^m (k+1)^{-n} & m \geq 1, n \geq 2, \end{aligned}$$

Explicit evaluations of some of the constants in these classes are presented with proofs in [8] and [9].

## 5. Numerical Techniques

It is not easy to compute numerical values of any of these Euler sums to high precision. Straightforward evaluation using the defining formulas, to some upper limit feasible on present-day computers, yields only about eight digits accuracy. Because integer relation algorithms require much higher precision to obtain reliable results, more advanced techniques must be employed.

Our approach to computing numerical values of these sums involves the compound application of the Euler-Maclaurin summation formula (see [2, p. 289]), which can be stated as follows. Suppose  $f(t)$  has at least  $2p + 2$  continuous derivatives on  $(a, b)$ . Let  $D$  be the differentiation operator, let  $B_k$  denote the  $k$ -th Bernoulli number, and let  $B_k(\cdot)$  denote the  $k$ -th Bernoulli polynomial. Then

$$\begin{aligned} \sum_{j=a}^b f(j) &= \int_a^b f(t) dt + \frac{1}{2}[f(a) + f(b)] \\ &+ \sum_{j=1}^p \frac{B_{2j}}{(2j)!} [D^{2j-1} f(b) - D^{2j-1} f(a)] + R_p(a, b). \end{aligned} \quad (1)$$

where the remainder  $R_p(a, b)$  is given [2, p. 289] by

$$R_p(a, b) = \frac{-1}{(2p+2)!} \int_a^b B_{2p+2}(t - [t]) D^{2p+2} f(t) dt.$$

We will briefly present a method for computing  $s_h(m, n)$ . See [4] for more details. Let  $h(k) = \sum_{j=1}^k 1/j$  and  $f(t) = 1/t$ . By applying the Euler-Maclaurin summation formula, it can be seen that

$$\begin{aligned} h(k) &= \gamma + \ln k + \frac{1}{2k} - \frac{1}{12k^2} + \frac{1}{120k^4} - \frac{1}{252k^6} + \frac{1}{240k^8} \\ &- \frac{1}{132k^{10}} + \frac{691}{32760k^{12}} - \frac{1}{12k^{14}} + \frac{3617}{8160k^{16}} + O(k^{-18}). \end{aligned} \quad (2)$$

We will use  $\bar{h}(k)$  to denote this particular approximation (i.e., (2) without the error term). Now consider the sum

$$s_h(m, n) = \sum_{k=1}^{\infty} \left(1 + \frac{1}{2} + \cdots + \frac{1}{k}\right)^m (k+1)^{-n}.$$

Let  $c$  be a large integer, and let  $g(t) = \bar{h}^m(t)(t+1)^{-n}$ . Applying the Euler-Maclaurin summation formula (1) again, we can write

$$\begin{aligned}
s_h(m, n) &= \sum_{k=1}^c \left(1 + \frac{1}{2} + \cdots + \frac{1}{k}\right)^m (k+1)^{-n} \\
&\quad + \sum_{k=c+1}^{\infty} \left(1 + \frac{1}{2} + \cdots + \frac{1}{k}\right)^m (k+1)^{-n} \\
&= \sum_{k=1}^c h^m(k)(k+1)^{-n} + \int_{c+1}^{\infty} g(t) dt + \frac{1}{2}g(c+1) \\
&\quad - \sum_{k=1}^9 \frac{B_{2k}}{(2k)!} D^{2k-1}g(c+1) + O(c^{-18}). \tag{3}
\end{aligned}$$

This formula suggests the following computational scheme. First, explicitly evaluate the sum  $\sum_{k=1}^c h^m(k)(k+1)^{-n}$  for  $c = 10^8$ , using a numeric working precision of 150 digits. Secondly, perform the symbolic integration and differentiation steps indicated in formula (3). Finally, evaluate the resulting expression, again using a working precision of 150 digits. The final result should be equal to  $s_h(m, n)$  to approximately 135 significant digits.

We have performed many computations of this type. The integration and differentiation operations required in (3) can be handled using a symbolic mathematics package, such as Maple [11] or Mathematica [17]. The explicit summation of the first  $c$  terms, as indicated in (3), could be performed by utilizing the multiple precision facility in the Maple or Mathematica packages. However, it was found that the MPFUN multiple precision package and translator developed by one of us [3] was significantly faster for this purpose.

Whatever software is used, this explicit summation is an expensive operation. For example, the evaluation of  $s_h(3, 4)$  to  $10^8$  terms, using the MPFUN package with 150-digit precision arithmetic, requires 20 hours on a ‘‘Crimson’’ workstation manufactured by Silicon Graphics, Inc. Thus while such runs can be made, clearly this is pressing the limits of current workstation technology. Fortunately, it is possible to perform such computations on a highly parallel computer system. The details of this parallel algorithm are given in [4].



## 6. Application of PSLQ to Euler Sums

The problem of recognizing Euler sum constants is well suited to analysis with integer relation algorithms. We will present but one example of these computations. Consider

$$\begin{aligned} s_a(2, 3) &= \sum_{k=1}^{\infty} \left( 1 - \frac{1}{2} + \cdots + \frac{(-1)^{k+1}}{k} \right)^2 (k+1)^{-3} \\ &= 0.156166933381176915881035909687988193685776709840 \cdots \end{aligned}$$

Based on experience with other constants, we conjectured that this constant satisfies a relation involving homogeneous combinations of  $\zeta(2), \zeta(3), \zeta(4), \zeta(5), \ln(2), \text{Li}_4(1/2)$  and  $\text{Li}_5(1/2)$ , where  $\text{Li}_n(x) = \sum_{k=1}^{\infty} x^k k^{-n}$  denotes the polylogarithm function.

The set of terms involving these constants with degree five (see section 7) are as follows:  $\text{Li}_5(1/2), \text{Li}_4(1/2) \ln(2), \ln^5(2), \zeta(5), \zeta(4) \ln(2), \zeta(3) \ln^2(2), \zeta(2) \ln^3(2), \zeta(2)\zeta(3)$ . When  $s_a(2, 3)$  is augmented with this set of terms, all computed to 135 decimal digits accuracy, and the resulting 9-long vector is input to the PSLQ algorithm, it detects the relation  $(480, -1920, 0, 16, 255, 660, -840, -160, 360)$  at iteration 390. Solving this relation for  $s_a(2, 3)$ , we obtain the formula

$$\begin{aligned} s_a(2, 3) &= 4 \text{Li}_5(1/2) - \frac{1}{30} \ln^5(2) - \frac{17}{32} \zeta(5) - \frac{11}{8} \zeta(4) \ln(2) + \frac{7}{4} \zeta(3) \ln^2(2) \\ &\quad + \frac{1}{3} \zeta(2) \ln^3(2) - \frac{3}{4} \zeta(2) \zeta(3) \\ &= 4 \text{Li}_5(1/2) - \frac{1}{30} \ln^5(2) - \frac{17}{32} \zeta(5) - \frac{11}{720} \pi^4 \ln(2) + \frac{7}{4} \zeta(3) \ln^2(2) \\ &\quad + \frac{1}{18} \pi^2 \ln^3(2) - \frac{3}{24} \pi^2 \zeta(3) \end{aligned}$$

(recall that  $\zeta(2n) = (2\pi)^{2n} |B_{2n}| / [2(2n)!]$ ).

When the relation is detected, the minimum and maximum  $y$  vector entries are  $1.60 \times 10^{-134}$  and  $5.98 \times 10^{-29}$ , respectively. Thus the confidence level of this detection is on the order of  $10^{-105}$ , indicating a very reliable detection.

Although 135-digit input values and 150-digit working precision were used when this relation was originally detected, the fact that the maximum  $y$ -vector entry is only  $10^{-29}$  at detection implies that such high levels of numeric precision are not required in this case.

Indeed, the above relation can be successfully detected using only the 50-digit input values listed above and 50-digit working precision when performing the PSLQ algorithm.

Table 1 lists a number of the formulas that have been found by this procedure. Table 2 lists some relations between the  $s_h$  and  $s_a$  constants. Others of both classes be found in [4]. It should be emphasized that the results in Tables 1 and 2 are not established in any rigorous mathematical sense by these calculations. However, in each case the “confidence level” (see section 3) of these detections is less than  $10^{-50}$ , and in most cases is in the neighborhood of  $10^{-100}$ .

## 7. New Formulas for $\pi$ and Related Constants

Many readers may be already familiar with the recent paper by the authors and Peter Borwein that gives a new method for computing individual digits of  $\pi$  and certain other related constants. A brief review of these results is as follows. First, we inquire whether  $\pi$  satisfies a relation of the form

$$\pi = \sum_{k=0}^{\infty} \frac{1}{16^k} \left[ \frac{a_0}{8k} + \frac{a_1}{8k+1} + \frac{a_2}{8k+2} + \frac{a_3}{8k+3} + \frac{a_4}{8k+4} + \frac{a_5}{8k+5} + \frac{a_6}{8k+6} + \frac{a_7}{8k+7} \right]$$

where  $a_i$  are rational numbers. Indeed it does. Such a formula can be found by separating the right hand side of the above expression into eight summations, numerically evaluating each to high precision, appending the numerical value of  $\pi$ , and applying PSLQ to the resulting 9-long vector. When this is done, PSLQ discovers the following formula:

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

A similar formula was discovered by Ferguson. These two formulas form the basis of a two-dimensional lattice of formulas of this form.

The significance of these formulas for the computation of  $\pi$  can be seen as follows. Let  $S_1$  be the first of the sums in the above formula (4). Then we can write

$$\text{frac}(16^d S_1) = \sum_{k=0}^{\infty} \frac{16^{d-k}}{8k+1} \pmod{1} \quad (5)$$

$$\begin{aligned}
s_h(3, 2) &= \frac{15}{2}\zeta(5) + \zeta(2)\zeta(3) \\
s_h(3, 3) &= -\frac{33}{16}\zeta(6) + 2\zeta^2(3) \\
s_h(3, 4) &= \frac{119}{16}\zeta(7) - \frac{33}{4}\zeta(3)\zeta(4) + 2\zeta(2)\zeta(5) \\
s_h(3, 6) &= \frac{197}{24}\zeta(9) - \frac{33}{4}\zeta(4)\zeta(5) - \frac{37}{8}\zeta(3)\zeta(6) + \zeta^3(3) + 3\zeta(2)\zeta(7) \\
s_h(4, 3) &= -\frac{109}{8}\zeta(7) + \frac{37}{2}\zeta(3)\zeta(4) - 5\zeta(2)\zeta(5) \\
s_h(5, 4) &= \frac{890}{9}\zeta(9) + 66\zeta(4)\zeta(5) - \frac{4295}{24}\zeta(3)\zeta(6) - 5\zeta^3(3) + \frac{265}{8}\zeta(2)\zeta(7) \\
s_h(6, 3) &= -\frac{3073}{12}\zeta(9) - 243\zeta(4)\zeta(5) + \frac{2097}{4}\zeta(3)\zeta(6) + \frac{67}{3}\zeta^3(3) - \frac{651}{8}\zeta(2)\zeta(7) \\
s_h(7, 2) &= \frac{134701}{36}\zeta(9) + \frac{15697}{8}\zeta(4)\zeta(5) + \frac{29555}{24}\zeta(3)\zeta(6) + 56\zeta^3(3) \\
&\quad + \frac{3287}{4}\zeta(2)\zeta(7) \\
s_a(2, 2) &= 6\text{Li}_4(1/2) + \frac{1}{4}\ln^4(2) - \frac{29}{8}\zeta(4) + \frac{3}{2}\zeta(2)\ln^2(2) \\
s_a(2, 3) &= 4\text{Li}_5(1/2) - \frac{1}{30}\ln^5(2) - \frac{17}{32}\zeta(5) - \frac{11}{8}\zeta(4)\ln(2) + \frac{7}{4}\zeta(3)\ln^2(2) \\
&\quad + \frac{1}{3}\zeta(2)\ln^3(2) - \frac{3}{4}\zeta(2)\zeta(3) \\
s_a(3, 2) &= -24\text{Li}_5(1/2) + 6\ln(2)\text{Li}_4(1/2) + \frac{9}{20}\ln^5(2) + \frac{659}{32}\zeta(5) - \frac{285}{16}\zeta(4)\ln(2) \\
&\quad + \frac{5}{2}\zeta(2)\ln^3(2) + \frac{1}{2}\zeta(2)\zeta(3)
\end{aligned}$$

Table 1: Experimentally Detected Formulas

$$\begin{aligned}
0 &= 84549s_h(1, 7) + 211468s_h(2, 6) + 148902s_h(3, 5) - 13360s_h(4, 4) - 1978s_h(5, 3) \\
0 &= -2718587s_h(1, 8) - 164525664s_h(2, 7) - 178042944s_h(3, 6) - 88947862s_h(4, 5) \\
&\quad + 3863940s_h(5, 4) + 672100s_h(6, 3) \\
0 &= -14269408s_h(1, 9) + 2578470s_h(2, 8) + 2815376s_h(3, 7) + 5814550s_h(4, 6) \\
&\quad + 6238884s_h(5, 5) + 3938912s_h(6, 4) + 1122784s_h(7, 3) - 1860s_h(8, 2) \\
&\quad + 63164285\zeta(10) \\
0 &= 321\zeta(10) - 440\zeta^2(5) - 720\zeta(3)\zeta(7) - 80\zeta^2(3)\zeta(4) + 560\zeta(2)\zeta(3)\zeta(5) \\
&\quad - 40s_h(2, 8) + 160s_h(3, 7) \\
0 &= -1691755503s_h(1, 10) - 3172589688s_h(2, 9) + 837511504s_h(3, 8) \\
&\quad - 7302717576s_h(4, 7) - 13958660016s_h(5, 6) - 12910466064s_h(6, 5) \\
&\quad - 7099332912s_h(7, 4) - 1773212688s_h(8, 3) + 658360s_h(9, 2) \\
&\quad + 53491434679\zeta(11) - 21868248971\zeta(2)\zeta(9) \\
0 &= -589\zeta(11) + 322\zeta(5)\zeta(6) + 756\zeta(4)\zeta(7) + 254\zeta(3)\zeta(8) - 336\zeta^2(3)\zeta(5) \\
&\quad - 368\zeta(2)\zeta(9) + 80\zeta(2)\zeta^3(3) - 16s_h(3, 8) - 48s_h(4, 7) \\
0 &= 1152s_a(2, 4) + 640s_a(3, 3) - 7680\ln(2)\text{Li}_5(1/2) + 64\ln^6(2) - 1881\zeta(6) \\
&\quad + 7440\zeta(5)\ln(2) - 1680\zeta(4)\ln^2(2) - 1120\zeta(3)\ln^3(2) + 864\zeta(3)\zeta(3) \\
&\quad - 640\zeta(2)\ln^4(2) - 432\zeta(2)\zeta(3)\ln(2)
\end{aligned}$$

Table 2: Experimentally Detected Relations

$$= \sum_{k=0}^d \frac{16^{d-k} \pmod{8k+1}}{8k+1} + \sum_{k=d+1}^{\infty} \frac{16^{d-k}}{8k+1} \pmod{1} \quad (6)$$

where  $\text{frac}$  denotes fractional part, i.e. the value mod 1.

The numerator of the first summation in (6) can be rapidly evaluated by means of the binary algorithm for exponentiation, where each operation is performed modulo the integer  $8k+1$ . These calculations can be done with either integer or floating-point arithmetic, provided the format being used has enough accuracy to exactly represent the integer  $d^2$ . Once an individual exponentiation operation is complete, the resulting integer value is divided by  $8k+1$ , using floating-point arithmetic, and added to the sum modulo 1. Only a few terms are required of the second summation in (6), since they rapidly become smaller than the “epsilon” of the floating-point arithmetic system being used. The resulting fractional value, when expressed in base 16 notation, gives the hexadecimal digits of  $\pi$  beginning at position  $d+1$ .

Here are a number of other formulas of this type. As before, these formulas were originally found using PSLQ searches.

$$\begin{aligned} \pi^2 &= \frac{1}{8} \sum_{k=0}^{\infty} \frac{1}{64^k} \left[ \frac{144}{(6k+1)^2} - \frac{216}{(6k+2)^2} - \frac{72}{(6k+3)^2} - \frac{54}{(6k+4)^2} + \frac{9}{(k+5)^2} \right] \\ \pi^2 &= \sum_{k=0}^{\infty} \frac{1}{16^k} \left[ \frac{16}{(8k+1)^2} - \frac{16}{(8k+2)^2} - \frac{8}{(8k+3)^2} - \frac{16}{(8k+4)^2} \right. \\ &\quad \left. - \frac{4}{(8k+5)^2} - \frac{4}{(8k+6)^2} + \frac{2}{(8k+7)^2} \right] \\ \log^2(2) &= \frac{1}{6} \sum_{k=0}^{\infty} \frac{1}{16^k} \left[ \frac{3}{(8k)^2} + \frac{16}{(8k+1)^2} + \frac{40}{(8k+2)^2} + \frac{8}{(8k+3)^2} + \frac{28}{(8k+4)^2} \right. \\ &\quad \left. + \frac{4}{(8k+5)^2} - \frac{10}{(8k+6)^2} + \frac{2}{(8k+7)^2} \right] \end{aligned}$$

Full details of these calculations, as well as formal proofs of the above formulas, can be found in [5].

## 8. A General Constant Recognition Procedure

In all of the cases mentioned above, the authors of the respective studies had “hunches” beforehand as to what form the resulting formulas might take. Frankly, some insight of

sort is invaluable in avoiding what otherwise is a combinatorial explosion in the number of possible terms. It simply is not possible to perform integer relation searches with every conceivable term. In fact, if the constant is known to only limited precision, the number of terms that can be considered in an integer relation search may be limited to a handful.

Nonetheless, it does appear feasible to define general search procedures that are often successful in recovering the analytic form of many constants that naturally appear in mathematical calculations. The authors present the following procedure as an example:

1. Using PSLQ and full precision, check if  $\alpha^j$  is algebraic of degree  $n$ , for  $j$  up to  $m$ .
2. Using PSLQ and full precision, check if  $\alpha$  is given by a multiplicative formula of the form

$$0 = a_1 \log(\alpha) + a_2 \log(2) + a_3 \log(3) + a_4 \log(5) + \cdots + a_r \log(p_r) \\ + a_{r+1} \log(c_1) + a_{r+2} \log(c_2) + \cdots + a_{r+s} \log(c_s)$$

where  $p_k$  is the  $k$ -th prime, and where  $c_k$  are a selected set of transcendentals.

3. Using PSLQ and quad precision, check if  $\alpha$  is given by a linear formula of the form

$$0 = a_0 + a_1 \alpha + a_2 x_1 + a_3 x_2 + \cdots + a_{t+1} x_t$$

where  $x_1, x_2, \dots, x_t$  are each a product of up to three constants from a selected set of algebraic and transcendental constants.

4. If a tentative relation is found in the previous step using quad precision, then check it using full precision.

In the above,  $m, n, r, s, t$  are parameters that can be set to adjust the amount of computer time one is willing to expend in the search. The authors have tried, for example,  $m = 8$ ,  $n = 32$ ,  $r = 20$ ,  $s = 20$ , and  $t = 3$ .

Some examples of constants recognized by above procedure are the following:

1. The root near 1.3851367 of the polynomial

$$\begin{aligned} & -14250992566272 + 1934517374976t^6 - 37548447232t^{12} + 3072863296t^{18} \\ & + 3789095144t^{24} - 408473063t^{30} - 43879700t^{36} - 5815353t^{42} \\ & + 319671t^{48} - 76384t^{54} - 852t^{60} - 444t^{66} + 132t^{72} + 23t^{78} + 3t^{84} + t^{90} \end{aligned}$$

2. The definite integral

$$\int_0^\infty t^{7/4} e^{-t} dt = \frac{21\pi\sqrt{2}}{16\Gamma(1/4)}$$

3. The definite integral

$$\int_0^{\pi/4} \frac{t^2 dt}{\sin^2(t)} = -\pi^2/16 + \pi \ln(2)/4 + G$$

where  $G$  denotes Catalan's constant

4. The definite integral

$$\int_0^1 \frac{t^2 \ln(t) dt}{(t^2 - 1)(t^4 + 1)} = \frac{\pi^2}{16(2 + \sqrt{2})}$$

## References

- [1] M. Abramowitz and I. A. Stegun, *Handbook of Mathematical Functions*, Dover Publications, New York, 1972.
- [2] K. E. Atkinson, *An Introduction to Numerical Analysis*, John Wiley, New York, 1989.
- [3] D. H. Bailey, “Multiprecision Translation and Execution of Fortran Programs,” *ACM Transactions on Mathematical Software*, to appear. This software and documentation may be obtained by sending electronic mail to `mp-request@nas.nasa.gov`.
- [4] D. H. Bailey, J. M. Borwein and R. Girgensohn, “Experimental Evaluation of Euler Sums,” *Experimental Mathematics*, vol. 3, no. 1 (1994), p. 17-30.
- [5] D. H. Bailey, P. B. Borwein and S. Plouffe, “On the Rapid Computation of Various Polylogarithmic Constants,” manuscript, 1995. Available from <http://www.cecm.sfu/~pborwein>.
- [6] D. H. Bailey and H. R. P. Ferguson, “Numerical Results on Relations Between Numerical Constants Using a New Algorithm,” *Mathematics of Computation*, vol. 53 (October 1989), p. 649 - 656.
- [7] B. C. Berndt, *Ramanujan’s Notebook*, Part I, Springer Verlag, New York, 1985.
- [8] D. Borwein and J. M. Borwein, “On An Intriguing Integral and Some Series Related to  $\zeta(4)$ ,” to appear in *Proceedings of the American Mathematical Society*.
- [9] D. Borwein, J. M. Borwein and R. Girgensohn, “Explicit Evaluation of Euler Sums,” to appear in *Proceedings of the Edinburgh Mathematical Society*.
- [10] J. M. Borwein and P. B. Borwein, *Pi and the AGM*, John Wiley, New York, 1987.
- [11] B. W. Char, K. O. Geddes, G. H. Gonnet, B. L. Leong, M. B. Monagan, S. M. Watt, *Maple V Language Reference Manual*, Springer-Verlag, New York, 1991.



- [12] H. R. P. Ferguson and D. H. Bailey, “A Polynomial Time, Numerically Stable Integer Relation Algorithm,” RNR Technical Report RNR-91-032, NASA Ames Research Center, MS T045-1, Moffett Field, CA 94035-1000.
- [13] H. R. P. Ferguson, D. H. Bailey and S. Arno, “Analysis of PSLQ, An Integer Relation Finding Algorithm,” manuscript, 1995.
- [14] H. R. P. Ferguson and R. W. Forcade, “Generalization of the Euclidean Algorithm for Real Numbers to All Dimensions Higher Than Two,” *Bulletin of the American Mathematical Society*, 1 (1979), p. 912 - 914.
- [15] J. Hastad, B. Just, J. C. Lagarias and C. P. Schnorr, “Polynomial Time Algorithms for Finding Integer Relations Among Real Numbers,” *SIAM Journal on Computing*, vol. 18 (1988), p. 859 - 881.
- [16] A. K. Lenstra, H. W. Lenstra and L. Lovasz, “Factoring Polynomials with Rational Coefficients”, *Math. Annalen*, vol. 261 (1982), p. 515 - 534.
- [17] S. Wolfram, *Mathematica: A System for Doing Mathematics by Computer*, Addison Wesley, Menlo Park, 1988.