REVERSE SYMBOLIC COMPUTATIONS, THE *IDENTIFY* FUNCTION

PETER BORWEIN[1], KEVIN G. HARE[2], AND ALAN MEICHSNER[3]

ABSTRACT. Although mathematicians commonly recognize the decimal expansions of numbers such as π and $\sqrt{2}$, few recognize the decimal expansions of numbers such as $2\pi + \ln(2)$ or $1 + \sqrt{2 + \sqrt{3}}$. The objective of this paper is to demonstrate a set of algorithms that aid in the identification of numerically computed constants. Most of these algorithms rely upon the ability to find a small integer relation for a vector $\mathbf{x} \in \mathbb{R}^n$ when one exists, that is a nonzero vector $\mathbf{m} \in \mathbb{Z}^n$ such that $\mathbf{x} \cdot \mathbf{m} = 0$.

[1]Department of Mathematics, Simon Fraser University, Burnaby, British Columbia, Canada V5A 1S6, pborwein@cecm.math.sfu.ca

[2]Department of Mathematics, Simon Fraser University, Burnaby, British Columbia, Canada V5A 1S6, kghare@cecm.math.sfu.ca

[3]Department of Mathematics, Simon Fraser University, Burnaby, British Columbia, Canada V5A 1S6, ameichsn@sfu.ca

This research is part of the MITACS Symbolic Analysis Project and is based on a section of the Master's thesis by Alan Meichsner, *Integer Relation Algorithms and the Recognition of Numerical Constants* [5]. Additional thanks go to Matt Klassen and Steve Funk for their contributions in the early stages.

1. Introduction

The calculation of transcendental and algebraic constants has been of great interest historically. The ancient Babylonians developed algorithms for computing $\sqrt{2}$. Archimedes method for computing π was one of the triumphs of ancient Greek mathematics. Named constants are, by and large, particular values of special functions and throughout the ages, faster and more efficient algorithms have been devised for such explicit evaluations.

The most ambitious projects so far that attempt to find closed form expressions for numerical values are *A Dictionary of Real Numbers* [1] and *The Inverse Symbolic Calculator* [6]. Although both work reasonably well, the fact that they are primarily table based imposes limitations stemming from the restricted number of entries in a table and the fixed precision to which each entry is given. Even though a certain value may have a simple closed form expression, approaches of this type will find it only if it has been included in a table. A failure to find a corresponding expression cannot exclude the possibility that the number is from a given class. Furthermore, in the event that the decimal expansion of a number is known to greater accuracy than the values given in a table, these additional digits will not be used in an attempt to identify the number. Whereas the expressions one gets from a set of tables may agree with the expansion of the number in the first few places, they may be very poor approximations of the original value.

Rather than taking a table based approach, we attempt to construct closed form expressions for a numerical value using various algorithms. Although this does not allow us to search in such a wide class of numbers as the previous two projects, it allows for more thorough searches within a given class and allows us to know exactly what types of numbers have and have not been considered.

This paper presents an algorithmic method, called *identify*, to determine whether a numerical value most probably has a closed form. The identify function is based on algorithm and code given in the Master's thesis by Alan Meichsner, *Integer Relation Algorithms and the Recognition of Numerical Constants* [5].

Consider a problem such as:

$$\int_{1.414213562}^{4.555806216} \cos(1.045493011x)dx.$$

Running *identify* on this will return:

$$\int_{\sqrt{2}}^{\sqrt{2}+\pi} \cos(\frac{5e}{13}x)dx.$$

The function *identify* takes a floating point number (or an expression containing floating point numbers) to some precision, and tries to determine which known constant or combination of constants best fits this floating point number. This algorithm makes significant use of integer relation algorithms to perform a large number of checks to determine, with some probability, to what explicit constant a numerical value corresponds.

Definition 1.1 (Integer Relation)**.** *There exists an integer relation amongst the numbers* $x_1, x_2, \ldots, x_n$ *if there are integers* $a_1, a_2, \ldots, a_n$*, not all zero, such that* $\sum_{i=1}^{n} a_i x_i = 0$*. For the vector* $\mathbf{x} = [x_1, x_2, \ldots, x_n]^T$*, the nonzero vector* $\mathbf{a} \in \mathbb{Z}^n$ *is an integer relation for* $\mathbf{x}$ *if* $\mathbf{a} \cdot \mathbf{x} = 0$*.*

Although the Euclidean and Continued Fraction Algorithms solve the problem of finding integer relations for the vector $[x_1, x_2, \cdots, x_n]$ when $n = 2$, until recently there were no known polynomial time algorithms that solved the problem for $n \geq 3$. A breakthrough was made in 1977 with Ferguson and Forcade's *Generalized Euclidean Algorithm* [2], a recursive algorithm that was guaranteed to find an integer relation when one existed. Following this, a number of non-recursive algorithms were developed, among which are the PSLQ algorithm, the HJLS algorithm, and a method based on the LLL algorithm [3, 4].

2. Recognizing Rational Numbers

Given a truncated decimal expansion $\overline{x}$ of the value x, we would like to decide whether or not x is rational. Although we cannot make any claims with certainty that are based only on the value $\overline{x}$, we will claim x is rational if there exist small integers p and q such that p/q is a good approximation of $\overline{x}$.

As a number is rational if and only if it has either a terminating or repeating decimal expansion, one method to search for appropriate values of p and q is to examine $\overline{x}$ and see if it appears periodic. However, in order for such a method to succeed, $\overline{x}$ must contain at least one complete period. This is not a reasonable expectation as it is possible for rational numbers with small denominators to have decimal expansions with long periods. For example, the decimal expansion of 13/877 has a period of length 438. It is unreasonable to expect that we will have enough digits in the expansion of $\overline{x}$ to contain a complete period. A further problem with this method is that a failure cannot be used to rule out the possibility that there exist small integers p and q with p/q being a good rational approximation of $\overline{x}$. Instead, we will consider the continued fraction convergents of $\overline{x}$ which have the following qualities [7]:

1. For any rational number r/s and continued fraction convergent p_k/q_k of $\overline{x}$, if $|\overline{x} - r/s| < |\overline{x} - p_k/q_k|$ then $|s| > q_k$.
2. For any continued fraction convergent p_k/q_k of $\overline{x}$, $|\overline{x} - p_k/q_k| < 1/q_k^2$.

It should be noted that although these results are usually given for the convergents of an irrational number, they also hold for the convergents of rational values.

We will define a convergent of $\overline{x}$ to be a good, small rational approximation of $\overline{x}$ if $|\overline{x} - p_k/q_k| \leq \epsilon$ where ϵ is the maximum error we are willing to tolerate and the number of digits in q_k is less than or equal to 1/3 the number of digits to which we know $\overline{x}$. If ϵ is small enough, then this is roughly equivalent to requiring both $|\overline{x} - p_k/q_k| \leq \epsilon$ and $|\overline{x} - p_k/q_k| < 1/q_k^3$ where we have increased the exponent of q_k to 3 in order to single out exceptional convergents.

To find a good, small rational approximation of $\overline{x}$, we need only compute the continued fraction convergents of $\overline{x}$ until either $|\overline{x} - p_k/q_k| \leq \epsilon$ or the number of digits in q_k is greater than 1/3 the number of digits to which we know $\overline{x}$. If this process terminates with the length of q_k less than or equal to 1/3 the length of $\overline{x}$, then $|\overline{x} - p_k/q_k| \leq \epsilon$ and so p_k/q_k satisfies our definition of a good, small rational approximation. In this case we will claim x is the rational number p_k/q_k.

On the other hand, if the process terminates with the length of q_k greater than 1/3 the length of $\overline{x}$, then we shall consider the convergent p_{k-1}/q_{k-1} of $\overline{x}$. As $|\overline{x} - p_{k-1}/q_{k-1}| > \epsilon$, there exist no rational values r/s such that $|s| < q_{k-1}$ and $|\overline{x} - r/s| \leq \epsilon$. We can claim that $\overline{x}$ is not a rational number with denominator less than or equal to q_{k-1} and so we will claim x is not a small rational number. Even though $|\overline{x} - p_{k-1}/q_{k-1}| > \epsilon$, the convergent p_{k-1}/q_{k-1} may still be of interest to us when looking for functions of small rationals that are good approximations of the input.

Example 1. *An example of identifying a rational number*

```
>   v := evalf(1/13);
```

$$v := .07692307692$$

```
>   identify(v);
```

$$\frac{1}{13}$$

3. Recognizing Algebraic Numbers

Again, given a truncated decimal expansion $\overline{x}$ of the value x, we would like to decide if x is algebraic. As an algebraic number is a root of its minimal polynomial there is an obvious way to proceed. If we can find an irreducible polynomial p with integer coefficients for which $\overline{x}$ is a good approximation of a root, and p is in some sense simple compared to $\overline{x}$, then we will say x is algebraic and identify it as a root of p. We will claim $\overline{x}$ is a good approximation of a root α if $|\alpha - \overline{x}| \leq \epsilon$ where ϵ is the maximum error we are willing to tolerate and we will claim p is simple compared to $\overline{x}$ if the total number of digits in the coefficients of p is no more than $1/3$ the number of digits in $\overline{x}$. Although there is some reason behind the choice of $1/3$ in the rational case, the choice here is fairly arbitrary.

To proceed, we shall try to construct a minimal polynomial for x by looking for an integer relation amongst the powers of $\overline{x}$. Working with the default size of n=6 and using the PSLQ algorithm, we will attempt to construct an integer relation for the vector $[1, \overline{x}, \ldots, \overline{x}^n]^T$. If a suspected relation is returned, then we need to examine the corresponding polynomial p. If $\overline{x}$ is a good approximation for a root of p, and p is simple in comparison to $\overline{x}$, then we shall say x is an algebraic number and identify it as a root of p. On the other hand, if the PSLQ algorithm returns a lower bound T on the norm of any possible integer relation, then we can claim there does not exist a polynomial of degree n and height less than T for which $\overline{x}$ is a root.

Although the implementation given here defaults to looking only for polynomials of degree less than or equal to 6, this method can be used to find a minimal polynomial of arbitrary degree provided we are given the value $\overline{x}$ to high enough precision. An option exists to allow a higher degree polynomial to be examined.

Example 2. *An example of identifying an algebraic number*

```
>   v := evalf((2+3^(1/3))^(1/3), 20);
```

$$v := 1.5098974493323553401$$

```
>   identify(v, basisSizeAlg=9 );
```

$$(2 + 3^{(1/3)})^{(1/3)}$$

Note that *identify* returns a closed form as opposed to a minimal polynomial.

4. Recognizing Sums and Products of Known Constants

Given an approximation $\overline{x}$ of the value x, if x is neither rational nor algebraic, then we will attempt to identify it by trying to construct an integer relation between $\overline{x}$ and a set of preselected constants. If one has some insight into what constants may be related to x, then the chance of success can be increased. Now suppose $c_1, c_2, \ldots, c_n$ are the constants we have selected and $[-a_0, a_1, \ldots, a_n]^T$ is a suspected integer relation for the vector $[\overline{x}, c_1, c_2, \ldots, c_n]^T$ in which a_0 does not equal zero. Let ϵ be the maximum error we are willing to tolerate in an approximation of $\overline{x}$ and for a rational number p/q in reduced form, define the size of p/q to be the maximum of the number of digits in p and the number of digits in q. In the event that $\left|\overline{x} - (\frac{a_1}{a_0}c_1 + \frac{a_2}{a_0}c_2 + \ldots + \frac{a_n}{a_0}c_n)\right| \le \epsilon$ and the sum of the sizes of the nonzero rationals $\frac{a_i}{a_0}$ is no more than $1/3$ the number of digits in $\overline{x}$, we will claim x is the value $(\frac{a_1}{a_0}c_1 + \frac{a_2}{a_0}c_2 + \ldots + \frac{a_n}{a_0}c_n)$. Again, the choice of $1/3$ as a measure of simplicity is somewhat arbitrary.

In our implementation, this procedure is applied in the following three ways. In each case, the constant n is dependent upon the accuracy to which we know x.

1. The first application is used to identify a sum of powers of a given transcendental number. For each member ξ in a list of selected transcendental numbers, we attempt to construct an integer relation for the vector $[\overline{x}, \xi, \xi^{p_1}, \xi^{p_2}, \ldots, \xi^{p_n}]^T$. The default choices for the ξ are π, e and $\ln(2)$, although an option exists to search other possibilities.
2. The second application is used to identify a sum of various constants. For all subsets of size n from a selected set of constants $\{c_1, c_2, \ldots, c_m\}$, we attempt to construct an integer relation for the vector $[\overline{x}, c_{i_1}, c_{i_2}, \ldots, c_{i_n}]^T$. To avoid combinatorial explosion, the constant m should not be too large. The default choice for the set of constants is $\{1, \sqrt{2}, \sqrt{3}, \zeta(3), \pi, \zeta(5), \mathrm{e}, \ln(2), \ln(3)\}$, although an option exists to search other possibilities.
3. The third method of application is used to identify a product of various constants. For all subsets of size n from a preselected set of positive constants $\{k_1, k_2, \ldots, k_m\}$, we attempt to construct an integer relation for the vector $[\log|\overline{x}|, \log(k_{i_1}), \log(k_{i_2}), \ldots, \log(k_{i_n})]^T$. If $[-a_0, a_1, \ldots, a_n]^T$ is found to be a suitable integer relation, then we claim $|x|$ is the value $\prod_{j=1}^n (k_{i_j})^{a_j/a_0}$. It is useful in this case for some of the constants k_i to be small primes. The default choice for the set of constants is $\{2, 3, 5, 7, \pi, \mathrm{e}, \ln(2), \ln(3), \zeta(3), \zeta(5)\}$, although an option exists to search other possibilities.

This procedure is also used in conjunction with the method given in section 5 to identify expressions involving functions of certain constants.

It should be noted that with an appropriate choice of constants, this procedure can be used to identify simple expressions involving both sums and products. If the constants $c_1, c_2, \ldots, c_n$ involve products or the constants $k_1, k_2, \ldots, k_n$ involve sums, then the resulting expressions may involve both sums and products.

Example 3. *An example of identifying sums, and products of known constants.*

```
>  v := evalf(exp(2)+exp(1), 20);
```

$$v := 10.107337927389695463$$

```
> identify(v);
```

$$e + (e)^2$$

```
> v := evalf(Pi+ln(7));
```

$$v := 5.087502803$$

```
> identify(v,extension={ln(7)});
```

$$\pi + \ln(7)$$

```
> v := evalf(Pi*exp(2));
```

$$v := 23.21340436$$

```
> identify(v);
```

$$\pi\, e^2$$

5. Recognizing Functions of Recognizable values

Following the lead given by the ISC, we can apply a set of simple transformations to the input before attempting to identify it. This is the approach we shall use to identify functions of small rational and algebraic numbers, as well as functions of sums and products of known constants. The *identify* function does not check if the value is a function of a $\mathbb{Q}$-linear combination of known constants by default, but there exist options to perform this check. Given a decimal approximation $\overline{x}$ of x and an invertible function F, we will first check to see if $\overline{x}$ is in the domain of F^{-1}. If it is, then we will set $\overline{y} = F^{-1}(\overline{x})$ and apply the previous methods for identifying constants to $\overline{y}$.

When looking for an approximation α of $\overline{y}$, it is not critical to return what we consider to be a good approximation. Depending on the slope of the function F at $\overline{y}$, the difference between α and $\overline{y}$ may not reflect the difference between $F(\alpha)$ and $\overline{x}$. We will instead focus mainly on finding a small recognizable approximations for $\overline{y}$ and then check to see if both α is in the domain of F and $|F(\alpha) - \overline{x}| \leq \epsilon$ where ϵ is the maximum error we are willing to tolerate. In the event that α is in the domain of F and this difference is less than ϵ, we will claim x is the value $F(\alpha)$.

In our implementation, we consider the following functions: sin, arcsin, cos, arccos, tan, arctan, csc, arccsc, sec, arcsec, cot, arccot, sinh, arcsinh, cosh, arccosh, tanh, arctanh, csch, arccsch, sech, arcsech, coth, arccoth, exp, ln, and the LambertW function. The inverse of each of these functions is readily available from Maple's *invfunc* array. Any function for which one can compute the inverse can easily be added. It is important to realize that a failure of this method cannot rule out the possibility that $x = F(\alpha)$ for some small rational or algebraic number α. If α is not in the range of the inverse function, then this method will not succeed.

Example 4. *An example of identifying a function of a recognizable constant.*

```
> v := evalf(cos(sqrt(2)), 20);
```

$$v := .15594369476537447346$$

```
> identify(v);
```

$$\cos(\sqrt{2})$$

References

[1] Jonathan Borwein and Peter Borwein, *A dictionary of real numbers*, Wadsworth & Brooks/Cole Advanced Books & Software, Pacific Grove, CA, 1990.

[2] H. R. P. Ferguson and R. W. Forcade, *Generalization of the Euclidean algorithm for real numbers to all dimensions higher than two*, Bull. Amer. Math. Soc. (N.S.) **1** (1979), no. 6, 912–914.

[3] Helaman R. P. Ferguson, David H. Bailey, and Steve Arno, *Analysis of PSLQ, an integer relation finding algorithm*, Math. Comp. **68** (1999), no. 225, 351–369.

[4] A. K. Lenstra, H. W. Lenstra, Jr., and L. Lovász, *Factoring polynomials with rational coefficients*, Math. Ann. **261** (1982), no. 4, 515–534.

[5] A. Meichsner, *Integer Relation Algorithms and the Recognition of Numerical Constants*, Master's thesis, Simon Fraser University, Burnaby, B.C., June 2001.

[6] S. Plouffe, *The Inverse Symbolic Calculator*, www.cecm.sfu.ca/projects/ISC/.

[7] Kenneth H. Rosen, *Elementary number theory and its applications*, third ed., Addison-Wesley Publishing Company Advanced Book Program, Reading, MA, 1993.