JOURNAL OF NUMERICAL ANALYSIS AND APPROXIMATION THEORY
J. Numer. Anal. Approx. Theory, vol. 50 (2021) no. 2, pp. 108–125
ictp.acad.ro/jnaat

A CATALOGUE OF MATHEMATICAL FORMULAS INVOLVING π, WITH ANALYSIS

DAVID H. BAILEY[*]

Abstract. This paper presents a catalogue of mathematical formulas and iterative algorithms for evaluating the mathematical constant π, ranging from Archimedes' 2200-year-old iteration to some formulas that were discovered only in the past few decades. Computer implementations and timing results for these formulas and algorithms are also included. In particular, timings are presented for evaluations of various infinite series formulas to approximately 10,000-digit precision, for evaluations of various integral formulas to approximately 4,000-digit precision, and for evaluations of several iterative algorithms to approximately 100,000-digit precision, all based on carefully designed comparative computer runs.

MSC 2010. 00A05, 68U01, 01-A05.

Keywords. Pi, computation of pi, history of pi.

1. BACKGROUND

The mathematical constant known as $\pi = 3.141592653589793\ldots$ is undeniably the most famous and arguably the most important mathematical constant. Mathematicians since the days of Archimedes, up to and including the present day, have analyzed its properties and computed its numerical value.

The question of whether π is given by a simple ratio or algebraic construction has transfixed mathematicians since ancient times. Squaring the circle, *i.e.*, constructing a square with the same area as a given circle using classical ruler-and-compass procedures, was one of the three premier unsolved problems of ancient Greek mathematics. In the 1760s, the Swiss-French mathematician Johann Heinrich Lambert first proved that π is irrational [27]. Then in 1882, the German mathematician Ferdinand von Lindemann proved that π is transcendental [28], meaning that π is not the root of any polynomial with integer or rational coefficients. Among other things, Lindemann's result brought a merciful end to the countless attempts over the centuries to square the circle. This is because any point or line segment that can be constructed using classical ruler-and-compass procedures is provably given by a finite algebraic

[*]Lawrence Berkeley National Laboratory, Berkeley, CA 94720 (retired) and University of California, Davis, Department of Computer Science, dhbailey@lbl.gov.

expression involving only the basic arithmetic operations and square roots, and thus is the root of an integer coefficient polynomial of degree 2^d for some integer d.

Attempts to compute numerical values of π are as old as π itself. Archimedes was the first to devise a rigorous scheme, based on inscribed and circumscribed polygons. He obtained the bounds $3\frac{10}{71} < \pi < 3\frac{1}{7}$, or, in other words, $3.1408\ldots < \pi < 3.1428\ldots$ [4]. Subsequently other mathematicians, in Europe, India, China and the Middle East, used Archimedes' approach to compute more accurate values. For example, about 500 CE, the Indian mathematician Aryabhata found π to four digits, and, at roughly the same time if not before, the Chinese mathematician Tsu Chung-Chih found π to seven digits. These and later computations were spurred by the invention of positional base-10 arithmetic with zero, by unknown Indian mathematicians in the first two or three centuries CE, a discovery which certainly deserves to be ranked as among the most important mathematical discoveries of all time [7].

With the development of calculus by Newton and Leibniz in the 1600s, and the discovery of infinite series and other formulas using calculus, π was computed to tens, then hundreds of digits, culminating with Shanks' 1874 hand computation to 707 digits (alas, only the first 527 were correct). With the advent of the computer in the 1940s, π was computed first to thousands of digits, then to millions, then to billions of digits by the end of the twentieth century. Since then, the pace of progress has continued uninterrupted. In the latest computation, announced on 18 August 2021, a record 62.8 trillion decimal digits were computed by a team of Swiss researchers [22]. For other details on the historical computation of π, see [9].

One intriguing development in this area was the discovery, in 1997, of a formula for π that permits one to calculate a string of binary or base-16 digits of π, beginning at an arbitrary starting position, without needing to calculate any of the digits that came before [14]. Since 1997, numerous other formulas with this property have been found for a variety of other mathematical constants. One of these formulas was used in 2010 to calculate a string of binary digits of π beginning at the two quadrillionth position [30]. Others were used to calculate base-64 digits of π^2, base-729 digits of π^2, and base-4096 digits of Catalan's constant, in each case beginning at position 10 trillion. For additional details, see [13] and [3].

This paper presents a collection of 72 formulas and algorithms that have been found by mathematicians over the years involving π. While a comprehensive collection is of course not possible, preference is given in this collection for formulas that satisfy the following criteria:

- Formulas that give π or a very simple expression involving π explicitly, as opposed to implicit relations such as $e^{i\pi} + 1 = 0$.
- Formulas that give π or a very simple expression involving π as an infinite series, definite integral or simple iterative algorithm.

- Formulas that involve simple notation, such as summations, integrals, binomial coefficients, exponentials, logarithms, etc., that would be familiar to anyone who has completed a beginning calculus course.
- Formulas that are relatively new, discovered within the last 100 years or so.

Included in this listing are several formulas for π that have actually have been used in large calculations of π, both before and since the rise of computer technology. These include formulas (2) through (5) prior to the 20th century, and formulas (6), (7), (11), (12), (13), (14), (16), (18), (69) and (71) in the late 20th and early 21st century. Several of these formulas, as we will see, are quite efficient. Formula (11) (known as the Ramanujan-Sato formula), for example, adds roughly eight correct digits per term, while formula (12) (due to the Chudnovskys) adds roughly 14 digits per term.

Formulas (13) through (18) have the intriguing property, mentioned above, that they permit digits (in certain specific bases) of the constant specified on the left-hand side to be calculated beginning at an arbitrary starting position, without needing to calculate any of the digits that came before, by means of relatively simple algorithms. Formulas (13) and (14) have been used in computations of high-order binary digits of π [17, Sec 3.4–3.6], while formula (16) has been used in computations of high-order binary digits of π^2, and formula (18) has been used in computations of high-order base-3 digits of π^2 [13]. Numerous other recently-discovered formulas that possess the arbitrary digit-computation property for various mathematical constants are catalogued in [3].

Many of these formulas are relatively new, in the sense that they were discovered only in the past few decades. The formulas mentioned in the previous paragraph are certainly in this category, having been discovered only since 1996. Many of the formulas from (19) through (50) were not well known until recently. Formulas (64) through (67) are also relatively new, in the sense that they are part of a class of integral formulas that are the subject of current research [10, 11, 12]. Formula (69) was discovered in 1976. Formulas (70), (71) and (72) were first published in 1984.

2. A CATALOGUE OF FORMULAS FOR π

$$\frac{\pi}{4} = \sum_{n=0}^{\infty} \frac{(-1)^n}{(2n+1)} \tag{1}$$

$$\frac{\pi}{4} = \sum_{n=0}^{\infty} \frac{(-1)^n}{(2n+1)2^{2n+1}} + \sum_{n=0}^{\infty} \frac{(-1)^n}{(2n+1)3^{2n+1}} \tag{2}$$

$$\frac{\pi}{4} = 4\sum_{n=0}^{\infty} \frac{(-1)^n}{(2n+1)5^{2n+1}} - \sum_{n=0}^{\infty} \frac{(-1)^n}{(2n+1)239^{2n+1}} \tag{3}$$

$$\tfrac{\pi}{4} = \sum_{n=0}^{\infty} \tfrac{(-1)^n}{(2n+1)2^{2n+1}} + \sum_{n=0}^{\infty} \tfrac{(-1)^n}{(2n+1)5^{2n+1}} + \sum_{n=0}^{\infty} \tfrac{(-1)^n}{(2n+1)8^{2n+1}} \tag{4}$$

$$\tfrac{\pi}{4} = 3\sum_{n=0}^{\infty} \tfrac{(-1)^n}{(2n+1)4^{2n+1}} + \sum_{n=0}^{\infty} \tfrac{(-1)^n}{(2n+1)20^{2n+1}} + \sum_{n=0}^{\infty} \tfrac{(-1)^n}{(2n+1)1985^{2n+1}} \tag{5}$$

$$\begin{aligned}\tfrac{\pi}{4} = {} & 12\sum_{n=0}^{\infty} \tfrac{(-1)^n}{(2n+1)49^{2n+1}} + 32\sum_{n=0}^{\infty} \tfrac{(-1)^n}{(2n+1)57^{2n+1}} - 5\sum_{n=0}^{\infty} \tfrac{(-1)^n}{(2n+1)239^{2n+1}} \\ & + 12\sum_{n=0}^{\infty} \tfrac{(-1)^n}{(2n+1)110443^{2n+1}}\end{aligned} \tag{6}$$

$$\begin{aligned}\tfrac{\pi}{4} = {} & 44\sum_{n=0}^{\infty} \tfrac{(-1)^n}{(2n+1)57^{2n+1}} + 7\sum_{n=0}^{\infty} \tfrac{(-1)^n}{(2n+1)239^{2n+1}} - 12\sum_{n=0}^{\infty} \tfrac{(-1)^n}{(2n+1)682^{2n+1}} \\ & + 24\sum_{n=0}^{\infty} \tfrac{(-1)^n}{(2n+1)12943^{2n+1}}\end{aligned} \tag{7}$$

$$\pi = \sqrt{12}\sum_{n=0}^{\infty} \tfrac{(-1)^n}{(2n+1)3^n} \tag{8}$$

$$\pi = \tfrac{3\sqrt{3}}{4} - 24\sum_{n=0}^{\infty} \tfrac{\binom{2n}{n}}{(2n+3)(2n-1)4^{2n+1}} \tag{9}$$

$$\tfrac{\pi}{2} = \prod_{n=1}^{\infty} \tfrac{4n^2}{4n^2-1} \tag{10}$$

$$\tfrac{1}{\pi} = \tfrac{2\sqrt{2}}{9801}\sum_{n=0}^{\infty} \tfrac{(4n)!(1103+26390n)}{(n!)^4 396^{4n}} \tag{11}$$

$$\tfrac{1}{\pi} = 12\sum_{n=0}^{\infty} \tfrac{(-1)^n(6n)!(13591409+545140134n)}{(3n)!(n!)^3 640320^{3n+3/2}} \tag{12}$$

$$\pi = \sum_{n=0}^{\infty} \tfrac{1}{16^n}\left(\tfrac{4}{8n+1} - \tfrac{2}{8n+4} - \tfrac{1}{8n+5} - \tfrac{1}{8n+6}\right) \tag{13}$$

$$\pi = 4\sum_{n=0}^{\infty} \tfrac{(-1)^n}{4^n(2n+1)} - \tfrac{1}{64}\sum_{n=0}^{\infty} \tfrac{(-1)^n}{1024^n}\left(\tfrac{32}{4n+1} + \tfrac{8}{4n+2} + \tfrac{1}{4n+3}\right) \tag{14}$$

$$\pi = \sum_{n=0}^{\infty} \tfrac{(-1)^n}{4^n}\left(\tfrac{2}{4n+1} + \tfrac{2}{4n+2} + \tfrac{1}{4n+3}\right) \tag{15}$$

$$\pi^2 = \tfrac{9}{8}\sum_{n=0}^{\infty} \tfrac{1}{64^n}\left(\tfrac{16}{(6n+1)^2} - \tfrac{24}{(6n+2)^2} - \tfrac{8}{(6n+3)^2} - \tfrac{6}{(6n+4)^2} + \tfrac{1}{(6n+5)^2}\right) \tag{16}$$

$$\begin{aligned}\pi\sqrt{3} = {} & \\ = {} & \tfrac{1}{9}\sum_{n=0}^{\infty} \tfrac{1}{729^n}\left(\tfrac{81}{12n+1} - \tfrac{54}{12n+2} - \tfrac{9}{12n+4} - \tfrac{12}{12n+6} - \tfrac{3}{12n+7} - \tfrac{2}{12n+8} - \tfrac{1}{12n+10}\right)\end{aligned} \tag{17}$$

$$\pi^2 = \frac{2}{27}\sum_{n=0}^{\infty}\frac{1}{729^n}\left(\frac{243}{(12n+1)^2} - \frac{405}{(12n+2)^2} - \frac{81}{(12n+4)^2} - \frac{27}{(12n+5)^2} - \frac{72}{(12n+6)^2}\right. \tag{18}$$
$$\left. - \frac{9}{(12n+7)^2} - \frac{9}{(12n+8)^2} - \frac{5}{(12n+10)^2} + \frac{1}{(12n+11)^2}\right)$$

$$3\pi + 8 = \sum_{n=0}^{\infty}\frac{12n2^{2n}}{\binom{4n}{2n}} \tag{19}$$

$$\frac{\pi^2}{6} - 2\log^2 2 = \sum_{n=1}^{\infty}\frac{\binom{2n}{n}}{n^2 4^n} \tag{20}$$

$$15\pi + 52 = \sum_{n=0}^{\infty}\frac{(126n^2-24n+8)2^{3n}}{\binom{6n}{3n}} \tag{21}$$

$$105\pi + 304 = \sum_{n=0}^{\infty}\frac{(1920n^3-928n^2+424n-16)2^{4n}}{\binom{8n}{4n}} \tag{22}$$

$$16\pi\sqrt{3} + 81 = \sum_{n=0}^{\infty}\frac{(49n+1)8^n}{3^n\binom{3n}{n}} \tag{23}$$

$$162 - 6\pi\sqrt{3} - 18\log 3 = \sum_{n=0}^{\infty}\frac{(-245n+338)8^n}{3^n\binom{3n}{n}} \tag{24}$$

$$\pi = \sum_{n=0}^{\infty}\frac{(50n-6)}{2^n\binom{3n}{n}} \tag{25}$$

$$15\pi + 42 = \sum_{n=1}^{\infty}\frac{(-4)^n(2n)!^2(3n)!(201-952n)}{(6n)!n!} \tag{26}$$

$$15\pi\sqrt{2} + 27 = \sum_{n=0}^{\infty}\frac{8^n(2n)!^2(3n)!(350n-17)}{(6n)!n!} \tag{27}$$

$$40\pi\sqrt{3} + 243 = \sum_{n=1}^{\infty}\frac{(-27)^n(2n)!^2(3n)!(81-1080n)}{(6n)!n!} \tag{28}$$

$$20\pi\sqrt{3} + 89 = \sum_{n=1}^{\infty}\frac{(-\frac{1}{3})^n(2n)!^2(3n)!(4123-22100n)}{(6n)!n!} \tag{29}$$

$$15\pi + 240\log 2 - 528 = \sum_{n=1}^{\infty}\frac{(-\frac{1}{2})^n(89012n^3-77362n^2+482n+3028)}{\binom{5n}{2n}} \tag{30}$$

$$24516 - 360\pi\sqrt{3} = \sum_{n=1}^{\infty}\frac{9^n(2743n^2-130971n-12724)}{\binom{4n}{n}} \tag{31}$$

$$45\pi + 1164 = \sum_{n=1}^{\infty}\frac{8^n(430n^2-6240n-520)}{\binom{4n}{n}} \tag{32}$$

$$40\pi\sqrt{3} + 1872 = \sum_{n=1}^{\infty}\frac{3^n(7175n^2-15215n+480)}{\binom{4n}{n}} \tag{33}$$

$$288\pi\sqrt{3} - 576\log 2 + 324 = \sum_{n=0}^{\infty} \frac{(\frac{9}{8})^n(5692+6335n-5415n^2)}{\binom{4n}{n}} \tag{34}$$

$$1008\pi\sqrt{3} - 576\log 2 + 7587 = \sum_{n=0}^{\infty} \frac{(\frac{9}{8})^n(7517+1145n+18050n^2)}{\binom{4n}{n}} \tag{35}$$

$$\frac{16}{\pi} = \sum_{n=0}^{\infty} \frac{42n+5}{4096^n}\binom{2n}{n}^3 \tag{36}$$

$$\frac{4}{\pi} = \sum_{n=0}^{\infty} \frac{(-1)^n(4n)!(20n+3)}{4^{4n}(n!)^4 2^{2n+1}} \tag{37}$$

$$\frac{4}{\pi} = \sum_{n=0}^{\infty} \frac{(-1)^n(4n)!(260n+23)}{4^{4n}(n!)^4 18^{2n+1}} \tag{38}$$

$$\frac{4}{\pi} = \sum_{n=0}^{\infty} \frac{(-1)^n(4n)!(21460n+1123)}{4^{4n}(n!)^4 882^{2n+1}} \tag{39}$$

$$\frac{2}{\pi\sqrt{3}} = \sum_{n=0}^{\infty} \frac{(4n)!(8n+1)}{4^{4n}(n!)^4 3^{2n+1}} \tag{40}$$

$$\frac{1}{2\pi\sqrt{2}} = \sum_{n=0}^{\infty} \frac{(4n)!(10n+1)}{4^{4n}(n!)^4 9^{2n+1}} \tag{41}$$

$$\frac{4}{\pi\sqrt{3}} = \sum_{n=0}^{\infty} \frac{(-1)^n(4n)!(28n+3)}{4^{4n}(n!)^4 3^n 4^{2n+1}} \tag{42}$$

$$\frac{4}{\pi\sqrt{5}} = \sum_{n=0}^{\infty} \frac{(-1)^n(4n)!(644n+41)}{4^{4n}(n!)^4 5^n 72^{2n+1}} \tag{43}$$

$$\frac{1}{3\pi\sqrt{3}} = \sum_{n=0}^{\infty} \frac{(4n)!(40n+3)}{4^{4n}(n!)^4 49^{2n+1}} \tag{44}$$

$$\frac{32}{\pi^2} = \sum_{n=0}^{\infty} \frac{\binom{4n}{2n}\binom{2n}{n}^4(120n^2+34n+3)}{2^{16n}} \tag{45}$$

$$\frac{128}{\pi^2} = \sum_{n=0}^{\infty} \frac{(-1)^n\binom{2n}{n}^5(820n^2+180n+13)}{2^{20n}} \tag{46}$$

$$\frac{2}{\pi} = \sum_{n=0}^{\infty} (-1)^n\binom{2n}{n}^3\frac{4n+1}{64^n} \tag{47}$$

$$\frac{4}{\pi} = \sum_{n=0}^{\infty} \binom{2n}{n}^3\frac{6n+1}{256^n} \tag{48}$$

$$\pi + 4 = \sum_{n=0}^{\infty} \frac{2^{n+1}}{\binom{2n}{n}} \tag{49}$$

$$\frac{6}{\pi^2} = 64 \sum_{n=0}^{\infty} \frac{(6n)!(532n^2+126n+9)}{(n!)^6 10^{6n+3}} \tag{50}$$

$$\frac{\pi}{4} = \int_0^1 \frac{\mathrm{d}x}{1+x^2} \tag{51}$$

$$\frac{\pi}{4} = \int_0^1 \sqrt{1-x^2}\,\mathrm{d}x \tag{52}$$

$$\frac{\pi-2}{4} = \int_0^1 x \tan^{-1} x\,\mathrm{d}x \tag{53}$$

$$\frac{\pi(\pi-12)}{48} + \frac{\log 2}{2} = \int_0^1 \log x \tan^{-1} x\,\mathrm{d}x \tag{54}$$

$$\frac{22}{7} - \pi = \int_0^1 \frac{x^4(1-x)^4\,\mathrm{d}x}{1+x^2} \tag{55}$$

$$\frac{\pi}{8} = \int_0^1 \frac{x^2\,\mathrm{d}x}{(1+x^4)\sqrt{1-x^4}} \tag{56}$$

$$\frac{\pi(1+2\log 2)}{8} = \int_0^\infty x e^{-x}\sqrt{1-e^{-2x}}\,\mathrm{d}x \tag{57}$$

$$4\pi \log^2 2 + \frac{\pi^3}{3} = \int_0^\infty \frac{x^2\,\mathrm{d}x}{\sqrt{e^x-1}} \tag{58}$$

$$\pi \log 2 = \int_0^{\frac{\pi}{2}} \frac{x^2\,\mathrm{d}x}{\sin^2 x} \tag{59}$$

$$\frac{\pi^3}{24} + \frac{\pi \log^2 2}{2} = \int_0^{\frac{\pi}{2}} \log^2(\cos x)\,\mathrm{d}x \tag{60}$$

$$\frac{8\pi^3}{81\sqrt{3}} = \int_0^1 \frac{\log^2 x\,\mathrm{d}x}{x^2+x+1} \tag{61}$$

$$\frac{\pi}{2} - \log 2 = \int_0^1 \frac{\log(1+x^2)\,\mathrm{d}x}{x^2} \tag{62}$$

$$\sqrt{\pi} = \Gamma(\tfrac{1}{2}) = \int_0^\infty x^{-\frac{1}{2}} e^{-x}\,\mathrm{d}x \tag{63}$$

$$-\frac{\pi}{4} + \frac{3\log(2+\sqrt{3})}{2} = \int_0^1\int_0^1\int_0^1 \frac{\mathrm{d}x\,\mathrm{d}y\,\mathrm{d}z}{\sqrt{x^2+y^2+z^2}} \tag{64}$$

$$-\frac{\pi}{24} + \frac{\sqrt{3}}{4} + \frac{\log(2+\sqrt{3})}{2} = \int_0^1\int_0^1\int_0^1 \sqrt{x^2+y^2+z^2}\,\mathrm{d}x\,\mathrm{d}y\,\mathrm{d}z \tag{65}$$

$$-\frac{\pi}{60} + \frac{2\sqrt{3}}{5} + \frac{7\log(2+\sqrt{3})}{20} = \int_0^1\int_0^1\int_0^1 (x^2+y^2+z^2)^{3/2}\mathrm{d}x\,\mathrm{d}y\,\mathrm{d}z \tag{66}$$

$$5 - \pi^2 - 4\log 2 + 16\log^2 2 = \int_0^1\int_0^1 \left(\frac{x-1}{x+1}\right)^2 \left(\frac{y-1}{y+1}\right)^2 \left(\frac{xy-1}{xy+1}\right)^2 \mathrm{d}x\,\mathrm{d}y \tag{67}$$

3. ITERATIVE ALGORITHMS FOR π

- (The Archimedes iteration). Set $a_0 = 2\sqrt{3}$ and $b_0 = 3$. Iterate, beginning with $k = 0$,
$$a_{k+1} = \tfrac{2a_k b_k}{a_k + b_k}, \quad b_{k+1} = \sqrt{a_{k+1} b_k}. \tag{68}$$
Then both a_k and b_k converge to π: each iteration decreases the distance between a_k and b_k (which interval contains π) by a factor of approximately four.
- (The Brent-Salamin iteration). Set $a_0 = 1$, $b_0 = 1/\sqrt{2}$ and $s_0 = 1/2$. Iterate, beginning with $k = 0$,
$$\begin{aligned} a_{k+1} &= \tfrac{a_k + b_k}{2}, \quad b_{k+1} = \sqrt{a_k b_k}, \\ c_{k+1} &= a_{k+1}^2 - b_{k+1}^2, \quad s_{k+1} = s_k - 2^{k+1} c_{k+1}, \\ p_{k+1} &= \tfrac{2a_{k+1}^2}{s_{k+1}}. \end{aligned} \tag{69}$$
Then p_k converges quadratically to π: each iteration approximately *doubles* the number of correct digits.
- (The Borwein cubic iteration). Set $a_0 = 1/3$ and $s_0 = (\sqrt{3} - 1)/2$. Iterate, beginning with $k = 0$,
$$\begin{aligned} r_{k+1} &= \tfrac{3}{1 + 2(1 - s_k^3)^{1/3}}, \quad s_{k+1} = \tfrac{r_{k+1} - 1}{2}, \\ a_{k+1} &= r_{k+1}^2 a_k - 3^k (r_{k+1}^2 - 1). \end{aligned} \tag{70}$$
Then $1/a_k$ converges cubically to π: each iteration approximately *triples* the number of correct digits. The Borweins also published a quadratically convergent algorithm, but that is not listed here.
- (The Borwein quartic iteration). Set $a_0 = 6 - 4\sqrt{2}$ and $y_0 = \sqrt{2} - 1$. Iterate, beginning with $k = 0$,
$$\begin{aligned} y_{k+1} &= \tfrac{1 - (1 - y_k^4)^{1/4}}{1 + (1 - y_k^4)^{1/4}} \\ a_{k+1} &= a_k (1 + y_{k+1})^4 - 2^{2k+3} y_{k+1} (1 + y_{k+1} + y_{k+1}^2). \end{aligned} \tag{71}$$
Then $1/a_k$ converges quartically to π: each iteration approximately *quadruples* the number of correct digits. Brent has shown that the Borwein quartic iteration is in fact mathematically equivalent to two iterations of the Brent-Salamin algorithm [20].
- (The Borwein nonic iteration). Set $a_0 = 1/3$, $r_0 = (\sqrt{3} - 1)/2$, $s_0 = (1 - r_0^3)^{1/3}$. Then iterate, beginning with $k = 0$,
$$\begin{aligned} t_{n+1} &= 1 + 2r_n, \quad u_{n+1} = (9r_n(1 + r_n + r_n^2))^{1/3} \\ v_{n+1} &= t_{n+1}^2 + t_{n+1} u_{n+1} + u_{n+1}^2, \quad w_{n+1} = \tfrac{27(1 + s_n + s_n^2)}{v_{n+1}} \\ a_{n+1} &= w_{n+1} a_n + 3^{2n-1} (1 - w_{n+1}) \\ s_{n+1} &= \tfrac{(1 - r_n)^3}{(t_{n+1} + 2u_{n+1}) v_{n+1}}, \quad r_{n+1} = (1 - s_{n+1}^3)^{1/3}. \end{aligned} \tag{72}$$

Then $1/a_k$ converge nonically to π: each iteration approximately *nine-times* the number of correct digits.

4. CREDITS

- Formula (1) was discovered by Leibniz and Gregory in the 1600s. Formula (2) was attributed to Euler in 1738. Formula (3) was discovered about the same time by Machin [17, 105]. The related arctangent-based formulas (4), (5), (6) and (7) were used by Dase, Ferguson, Kanada and Kanada, respectively [17, 106, 107, 111].
- Formula (8) is due to the Indian mathematician Madhava of Sangamagramma, who lived in the late 1300s and early 1400s [17, 107]. Formula (9) was discovered by Newton in the mid-1600s [17, 106]. Formula (10) was discovered by Wallis at about the same time.
- Formula (11) is due to Ramanujan, and was used by Gosper in 1986 to compute π to over 17 million digits. The similar but more complicated formula (12) is due to David and Gregory Chudnovsky, and was used by them to compute π to over one billion decimal digits [17, 108].
- Formula (13) is known as the "BBP" formula for π, named for the initials of the co-authors of the 1997 paper where it was first presented [14], [17, 119–124]. It was discovered by a computer program running the "PSLQ" algorithm of mathematician-sculptor Helaman Ferguson [24, 15]. Formula (14) is a variant of the BBP formula due to Bellard [17, 124]. Formula (15) was found by Helaman Ferguson and independently by Adamchik and Wagon, who first published it [1].
- Formula (16) appeared in [14]. Formulas (17) and (18) are due to David Broadhurst [21].
- Some of the summation formulas involving factorials and combinatorial coefficients (*i.e.*, formulas (19) through (50)) were found by Ramanujan; others are due to David and Gregory Chudnovsky. The Chudnovskys had these and many other formulas of this general type inscribed on the floor of their research center at Brooklynn Polytechnic University in New York City [23]. Four exceptions are formula (36), which is due to Ramanujan but appeared in [19, 188], formulas (45) and (46), which are due to Guillera [26], and formula (50), which is due to Almkvist and Guillera [2].
- Formulas (51) through (63) have been known for many years; many are from [18, 5, 48, 320–321].
- Formulas (64) through (66) are examples of recent discoveries, by computational methods involving the PSLQ algorithm [24, 15], in the theory of box integrals [11, 12]. Formula (65), for instance, can be thought of as specifying the average distance from the origin to a point in the unit 3-cube.

- Formula (67) is an example of numerous formulas, also obtained by computational methods involving the PSLQ algorithm [24, 15], in studies of the Ising theory of mathematical physics [10].
- Formula (68), is mathematically equivalent to Archimedes' approach involving computing the areas of inscribed and circumscribed regular polygons [4]. Archimedes' scheme was used for all computations of π in ancient times, including by the fifth century Chinese mathematician Tsu Chung-Chih and, evidently, by the fifth century Indian mathematician Aryabhata [7].
- Formula (69) is the Brent-Salamin iteration, the first quadratically convergent scheme for π, which was discovered independently by Richard Brent and Eugene Salamin in 1976 [17, 109–110]. Formula (70) (a cubically convergent iteration), formula (71) (a quartically convergent iteration) and formula (72) (a nonically convergent iteration) are due to Jonathan and Peter Borwein [17, 110], [9].

5. PERFORMANCE RESULTS

One question that frequently arises in discussions of formulas and algorithms for π is how they compare when implemented on the computer. To that end, we present here timings for a carefully designed set of comparative computer runs. Timings are presented for the infinite series summation formulas (using 10,000-digit precision), for the integral formulas (using 4,000-digit precision), and for the iterative algorithm formulas (using 100,000-digit precision).

5.1. Software. The present author tried several different approaches to these timings, including Mathematica software and various high-precision arithmetic libraries. One difficulty with Mathematica implementations is that it is difficult to control how much symbolic manipulation and simplification is being done "under the covers" of the user code. Also, evaluation of the integral formulas is problematic using Mathematica, because it is difficult to control details of the implementation, even when specifying the method to be used.

In the end, the author decided to base the timings below on Fortran implementations utilizing the author's MPFUN-2015 package [5], in particular the MPFUN-MPFR version of MPFUN-2015. This is a high-level multiprecision package, in the sense that it permits one to perform arbitrarily high-precision computations in a Fortran program by making only a few changes to standard double-precision code. For the most part, one only needs to declare high-precision variables to be of a certain datatype, and then the software automatically calls the requisite lower-level routines from the package whenever one of these variables appears in an expression. The package supports both high-precision real and high-precision complex datatypes. The MPFUN-2015 package is entirely thread-safe, so that applications using the package can be performed safely in shared-memory parallel implementations. For full details, see [5].

The MPFUN-2015 package is available in two versions, MPFUN-Fort and MPFUN-2015. The MPFUN-Fort version is written in Fortran, and thus it is a simple matter to compile, install and use. The MPFUN-MPFR version has the same functionality as MPFUN-Fort, but calls the MPFR library [25] for all lower-level computations. At the present time the MPFR library features the fastest runtimes of any arbitrary precision floating-point library [29]. It also produces results that are correctly rounded to the last bit. Thus while the the installation process of the MPFUN-MPFR is more involved (because both the MPFR library and the GMP library must be installed first, using administrator privilege), it features very fast run times.

A newer version of this package, MPFUN-2020, which features significantly faster all-Fortran run times, is now available [6], although it was not used for these computations.

5.2. Evaluation of summation formulas. The summation formulas (formula (1) through formula (50)) were all evaluated using a consistent approach and coding style. It is important to note that these are purely numerical evaluations – symbolic manipulations and simplifications, such as by noting that $\sum_{n\geq 0}(-1)^n/((2n+1)8^{2n+1}) = \arctan(1/8)$, were not employed. Some straightforward computational simplifications were employed, typical of those that would be utilized in any efficient implementation. For example, numerators and denominators were separately evaluated, because they are integers, and powers such as 2^{2n+1} and binomial coefficients such as $\binom{2n}{n}$ were evaluated incrementally from iteration to iteration in a loop. Each individual summation was performed only until its terms were less than 10^{-10000}.

It should be added, though, that advanced techniques such as "multisectioning" and "divide and conquer" strategies were not performed. These terms refer to evaluating sections of consecutive terms in the summation, with integer coefficients in the numerators and denominators factored out as much as possible. Such techniques do not make much difference for runs up to 10,000 digits or so, as in these tests, but have been employed in computations of π to many millions of digits. The most successful of these implementations is the "y-cruncher" program of Alexander J. Yee [31], which has been used in the most recent computations of π, based on the Chudnovsky formula (12). In the latest computation, announced on 18 August 2021, a record 62.8 trillion decimal digits were computed by a team of Swiss researchers [22].

The timings for the summation formula computer runs are presented in Table 1. As mentioned above, these timings are based on runs to 10,000-digit precision. In some cases, timings are not listed, because these formulas would require astronomical numbers of terms to produce 10,000-digit results.

5.3. Tanh-sinh quadrature. From a computational standpoint, integral formulas are no match to the most efficient summation formulas and iterative

Formula	Run time	Terms of series	Formula	Run time	Terms of series
1	—	—	26	0.69	6693
2	2.49	16602, 10475	27	0.88	8855
3	0.83	7150, 2101	28	1.64	16627
4	2.62	16602, 7150, 5534	29	0.39	3988
5	1.23	8301, 3842, 1516	30	0.60	5685
6	0.79	2957, 2847, 2101, 991	31	45.53	442610
7	0.71	2847, 2101, 1764, 1216	32	14.05	135741
8	1.87	20950	33	2.07	20040
9	1.77	16590	34	1.12	10818
10	—	—	35	1.12	10819
11	0.12	1253	36	2.00	5536
12	0.72	706	37	1.78	16607
13	2.93	8298	38	0.38	3983
14	2.65	16603, 3322	39	0.16	1698
15	5.83	16603	40	0.99	10477
16	1.98	5533	41	0.50	5239
17	0.40	3491	42	0.56	5947
18	0.43	3491	43	0.21	2266
19	1.61	16621	44	0.28	2958
20	—	—	45	2.95	8304
21	1.12	11088	46	1.21	3322
22	0.88	8320	47	—	—
23	2.43	24815	48	6.60	16607
24	2.43	24817	49	3.09	33229
25	0.86	8854	50	0.72	7510

Table 1. Timings for evaluations of summation formulas to approximately 10,000-digit precision.

algorithms, but are still nonetheless fairly reasonable if performed using an efficient quadrature (numerical integration) scheme. To that end, the author used variations of the tanh-sinh algorithm for all of the integral formulas (formulas (51) through (67)). The tanh-sinh scheme, while often not quite as efficient as Gaussian quadrature for entirely regular integrand functions, nonetheless has significant advantages for this type of very high-precision computation [16]. Given a function $f(t)$ defined on $[-1, 1]$, the tanh-sinh quadrature rule is

$$\int_{-1}^{1} f(x)\,\mathrm{d}x \quad = \quad \int_{-\infty}^{\infty} f(g(t))g'(t)\mathrm{d}t \;\approx\; h\sum_{j=-N}^{N} w_j f(x_j), \tag{73}$$

where $g(t) = \tanh(\frac{\pi}{2}\cdot\sinh t)$ and the abscissas x_j and weights w_j are given by

$$\begin{aligned} x_j &= g(h_j) = \tanh(\tfrac{\pi}{2}\cdot\sinh(hj)) \\ w_j &= g'(hj) = \tfrac{\pi}{2}\cdot\cosh(hj)/\cosh^2(\tfrac{\pi}{2}\cdot\sinh(hj)) \end{aligned} \tag{74}$$

The tanh-sinh scheme can be used for functions on any finite interval. A variation of the tanh-sinh scheme known as the exp-sinh scheme, based on the function $g(t) = \exp(\frac{\pi}{2}\cdot\sinh t)$, can be employed for integrals on a semi-infinite interval such as $(0, \infty)$. The sinh-sinh scheme, based on $g(t) = \sinh(\frac{\pi}{2} \cdot \sinh t)$, can be employed for integrals on the entire real line.

One advantage of the tanh-sinh scheme is that it can be readily used for problems, such as formulas (56) and (58), whose integrand functions (or their higher-order derivatives) have a vertical derivatives or singularities at one or both endpoints. It is often not easy to determine whether or not an integral has such a singularity. For example, consider the integral $\int_0^1 \sin^p(\pi x)\zeta(p,x)\,\mathrm{d}t$, where $\zeta(p,x)$ denotes the Hurwitz zeta function. When $p = 3$, this integrand function and its higher derivatives are all regular, and can be integrated using Gaussian quadrature, but when $p = 7/2 = 3.5$, while the plot of this function looks unremarkable, its fourth and higher derivatives have singularities at the endpoints, and Gaussian quadrature fails badly. By contrast, the tanh-sinh quadrature rule easily integrates this function to high precision [8].

Another major advantage of the tanh-sinh scheme for very-high-precision computation is that the cost of computation of abscissas and weights, using (74), increases only linearly with N, whereas the abscissa-weight computation in Gaussian quadrature increases quadratically with N. Given that the number of evaluation points required to achieve a given precision increases roughly linearly with the number of digits, this means that the cost of computing abscissas and weights for the Gaussian scheme increases roughly cubically with the precision desired, compared with quadratically with tanh-sinh, even before the increases in the cost of the arithmetic with higher precision are considered. For example, computation of tanh-sinh abscissas and weights sufficient to evaluate the integral problems in this paper to approximately 4,000-digit accuracy required only 167 seconds, whereas the corresponding calculations for Gaussian quadrature would require over 100 hours run time (and the Gaussian scheme, as mentioned above, could not be used for formulas (56) and (58), because of singularities at endpoints).

Timings for evaluations of the integral formulas are shown in Table 2. The tanh-sinh scheme was employed for all formulas except formulas (57), (58) and (63), which employed the exp-sinh scheme.

5.4. Evaluations of multiple integrals. As noted in the introduction, formulas (64), (65) and (66) derive from a recent study of box integrals [11, 12]. In this study, high-precision numerical values of these integrals (and others) were used, in conjunction with the PSLQ integer relation algorithm [24, 15], to numerically discover the relations indicated. A similar approach was taken to numerically discover evaluations of integrals such as formula (67) that derive from the Ising theory of mathematical physics [10]. Indeed, such studies overwhelmingly demonstrate the value of very high-precision quadrature in experimental mathematics.

Numerical evaluations of formulas (64), (65) and (66) were facilitated by the discovery [11, 12] that these multidimensional box integrals can be rewritten, in most cases, in terms of 1-D integrals, and similarly for formula (67) from

Formula	Run time	Subdivisions
51	1.85	4096
52	1.26	4096
53	34.15	4096
54	59.62	4096
55	2.46	4096
56	8.89	8192
57	100.45	16384
58	79.25	16384
59	82.58	4096
60	79.03	4096
61	28.11	4096
62	21.03	4096
63	82.43	16384
64	2.18	4096
65	6.39	4096
66	7.20	4096
67	35.20	4096

Table 2. Timings for evaluations of integral formulas to approximately 4,000-digit precision.

Ising studies [10]. In particular, the computations reported here for formulas (64), (65), (66) and (67) are based, respectively, on the following reductions to 1-D integrals:

$$\int_0^1\int_0^1\int_0^1 \frac{\mathrm{d}x\,\mathrm{d}y\,\mathrm{d}z}{\sqrt{x^2+y^2+z^2}} = 3\int_0^1 \frac{\sqrt{t^2+2}-1}{t^2+1}\,\mathrm{d}t \tag{75}$$

$$\int_0^1\int_0^1\int_0^1 \sqrt{x^2+y^2+z^2}\,\mathrm{d}x\,\mathrm{d}y\,\mathrm{d}z = \frac{1}{2}\int_0^1 \frac{(t^2+2)^{3/2}-1}{t^2+1}\,\mathrm{d}t \tag{76}$$

$$\int_0^1\int_0^1\int_0^1 (x^2+y^2+z^2)^{3/2}\mathrm{d}x\,\mathrm{d}y\,\mathrm{d}z = \frac{1}{5}\int_0^1 \frac{(t^2+2)^{5/2}-1}{t^2+1}\,\mathrm{d}t \tag{77}$$

$$\int_0^1\int_0^1 \left(\frac{x-1}{x+1}\right)^2\left(\frac{y-1}{y+1}\right)^2\left(\frac{xy-1}{xy+1}\right)^2\mathrm{d}x\,\mathrm{d}y = \int_0^1 \Big[\frac{t(-7+4\log(2))+t^2(1+20\log(2))+t^3(3+12\log(2))}{t(t-1)(t+1)^2} + \frac{t^4(3-4\log(2))-4(1+t)(-1+4t+t^2)\log(1+t)}{t(t-1)(t+1)^2}\Big]\mathrm{d}t \tag{78}$$

5.5. **Evaluations of iterative algorithms.** In addition to the summation integral formulas, results are presented here for five iterative algorithms, namely the Archimedes iteration (formula (68)), the Brent-Salamin iteration (formula (69)), the Borwein cubic iteration (formula (70)), the Borwein quartic iteration (formula (71)) and the Borwein nonic iteration (formula (72)).

The Archimedes iteration is mathematically equivalent to the scheme sketched by the ancient Greek mathematician Archimedes in approximately 250 BCE. For details on how this iterative formula is derived, and a rigorous proof that it converges to π, see [4].

The remaining iterative algorithms have the remarkable property that they converge quadratically (formula (69)), cubically (formula (70)), quartically (formula (71)) and nonically (formula (72)), respectively, meaning that the number of correct digits approximately doubles, triples, quadruples and nine-times, respectively, with each iteration, provided of course that all iterations are performed with a level of numeric precision that is at least as high as the precision desired for the final result.

These iterations were implemented in an entirely straightforward fashion. The only change from the formulas as stated were to save results of some intermediate expressions, such as the value of $(1 - y_k^4)^{1/4}$ in (71)), rather than recomputing them each time they appear.

6. TIMING RESULTS

Timing results for the summation formulas (using 10,000-digit precision), the integral formulas (using 4,000-digit precision) and the iterative algorithms (using 100,000-precision) are given in Table 1, Table 2 and Table 3, respectively. These runs were performed on a single processor of a 2019 MacPro with a 3 GHz 8-core Intel Xeon E5 processor and 32 GB RAM. It utilized the MPFUN-MPFR package, version 9, with version 4.0.2 of the MPFR library and version 6.1.2 of the GMP library.

The final results of each calculation were checked against the reference values to verify that the relative errors met the prescribed tolerance. Each of the summation formula results met the relative error tolerance 10^{-10000}, except formulas (9), (19), (31) and (50) (which met a relative error of 10^{-9998}). Each of the integral formula results met the relative error tolerance 10^{-4000}, except formula (51) (10^{-3948}), formula (58) (10^{-3688}) and formula (63) (10^{-3954}). Each of the iterative algorithm results met the relative error tolerance $10^{-100000}$.

These results clearly indicate a very wide range in timings, even among formulas of the same class. Among the summation formulas, timings ranged from 0.12 seconds for formula (11) (a formula due to Ramanujan that has been used in some recent large computations of π) to 45.53 seconds for formula (31), which in spite of its similar outward appearance to other formulas involving binomial coefficients, converges very slowly. And, of course, several other formulas ((1), (10), (20) and (47)) converge so slowly that no timings are presented, since evaluations of π to 10,000-digit precision using these formulas would require astronomically long run times. For example, evaluating π to 10,000-digit precision strictly using Gregory's series for π (formula (1)) would require evaluating roughly 10^{10000} terms and vastly more run time than the age of the universe.

From a computational perspective, the integral formulas are more challenging, as they require advanced techniques for evaluation to high precision, as noted above. Even here, though, we observe vast differences in run time, ranging from 1.26 seconds using the simple integral in formula (52) to 100.45 seconds for the integral in formula (57). The large run time here mostly reflects the cost of computing the exponential function in (57). Note, however, that a complicated integrand function by itself does not guarantee a very long run time. The most complicated integral in the list, namely formula (67), which was evaluated using the equivalent but still very complicated 1-D integral in formula (78), required only 35.2 seconds to produce a 4,000-digit value.

The iterative algorithm formulas show a particularly dramatic contrast in run times. Here, as mentioned above, the timings are for computations using 100,000-digit arithmetic. The Archimedean iteration, namely formula (68), required 166,133 iterations and 1015.39 seconds run time to produce a result accurate to 100,000 digits. But each of the more modern iterations, specifically the Brent-Salamin algorithm ((69)) and the three Borwein algorithms ((70), (71), (72)), all required between 0.11 and 0.14 seconds, which are truly remarkable speeds for 100,000-digit results.

It should be noted, however, that although the Brent-Salamin algorithm (69) and the Borwein quartic algorithm (71) have been used in several recent large computations of π, the Chudnovsky formula (12) is now the most widely used formula for very large computations of π. Even though it is significantly slower than the Brent-Salamin and Borwein formulas, by the tests reported in this paper, advanced techniques such as multisectioning and divide-and-conquer strategies can be employed with this formula, which techniques prevail in computations to millions, billions or trillions of digits.

Formula	Run time	Iterations
68	1015.39	83093
69	0.13	16
70	0.11	9
71	0.11	7
72	0.14	5

Table 3. Timings for evaluations of iterative algorithm formulas to approximately 100,000-digit precision.

REFERENCES

[1] V. Adamchik, S. Wagon, *A simple formula for pi,* Amer. Math. Monthly, **104** (1997), 852–855, https://www.maa.org/sites/default/files/pdf/pubs/amm_supplements/Monthly_Reference_9.pdf.

[2] G. Almkvist, J. Guillera, *Ramanujan-like series for $1/\pi^2$ and string theory*, Experimental Mathematics, **21** (2012) no. 3, pp. 223–234, https://doi.org/10.1080/10586458.2012.656059.

[3] D.H. Bailey, *A compendium of BBP-type formulas for mathematical constants*, updated 15 Aug 2017, http://www.davidhbailey.com/dhbpapers/bbp-formulas.pdf.

[4] D.H. Bailey, *Simple proofs: Archimedes' calculation of pi*, Math. Scholar, 9 Feb 2019, https://mathscholar.org/2019/02/simple-proofs-archimedes-calculation-of-pi/.

[5] D.H. Bailey, *MPFUN2015: A thread-safe arbitrary precision package* (full documentation), manuscript, updated 7 Feb 2020, https://www.davidhbailey.com/dhbpapers/mpfun2015.pdf.

[6] D.H. Bailey, *MPFUN2020: A new thread-safe arbitrary precision package* (full documentation), manuscript, updated 20 Jul 2021, https://www.davidhbailey.com/dhbpapers/mpfun2020.pdf.

[7] D.H. Bailey, J.M. Borwein, *Ancient Indian square roots: An exercise in forensic paleo-mathematics*, Amer. Math. Monthly, **119** (2012) no. 8, 646–657, preprint draft at https://doi.org/10.4169/amer.math.monthly.119.08.646.

[8] D.H. Bailey, J.M. Borwein, *Hand-to-hand combat with thousand-digit integrals*, J. Computational Science, **3** (2012), 77–86, preprint draft at https://doi.org/10.1016/j.jocs.2010.12.004.

[9] D.H. Bailey, J.M. Borwein, P.B. Borwein, S. Plouffe, *The quest for Pi,* Mathematical Intelligencer, **19** (1997) no. 1, 50–56, https://doi.org/10.1007/BF03024340.

[10] D.H. Bailey, J.M. Borwein, R.E. Crandall, *Integrals of the Ising class*, J. Phys. A Math. Gen., **39** (2006), 12271–12302, https://doi.org/10.1088/0305-4470/39/40/001.

[11] D.H. Bailey, J.M. Borwein, R.E. Crandall, *Box integrals*, J. Comp. Appl. Math., **206** (2007), 196–208, https://doi.org/10.1016/j.cam.2006.06.010.

[12] D.H. Bailey, J.M. Borwein, R.E. Crandall, *Advances in the theory of box integrals*, Math. Comp., **79** (2010) no. 271, 1839–1866, https://doi.org/10.1090/S0025-5718-10-02338-0.

[13] D.H. Bailey, J.M. Borwein, A. Mattingly, G. Wightwick, *The computation of previously inaccessible digits of π^2 and Catalan's constant*, Notices of the AMS, **60** (2013) no. 7, 844–854, http://dx.doi.org/10.1090/noti1015.

[14] D.H. Bailey, P.B. Borwein, S. Plouffe, *On the rapid computation of various polylogarithmic constants*, Math. Comp., **66** (1997) no. 218, 903–913, https://doi.org/10.1090/S0025-5718-97-00856-9.

[15] D.H. Bailey, D.J. Broadhurst, *Parallel integer relation detection: Techniques and applications,* Math. Comp., **70** (2000) no. 236, 1719–1736, https://doi.org/10.1090/S0025-5718-00-01278-3.

[16] D.H. Bailey, Xiaoye S. Li, K. Jeyabalan, *A comparison of three high-precision quadrature schemes*, Experimental Mathematics, **14** (2005) no. 3, 317–329, https://doi.org/10.1080/10586458.2005.10128931.

[17] J.M. Borwein, D.H. Bailey, *Mathematics by Experiment: Plausible Reasoning in the 21st Century*, AK Peters, Natick, MA, 2008.

[18] J.M. Borwein, D.H. Bailey, R. Girgensohn, *Experimentation in Mathematics: Computational Paths to Discovery,* AK Peters, Natick, MA, 2004.

[19] J.M. Borwein, P.B. Borwein, *Pi and the AGM: A Study in Analytic Number Theory and Computational Complexity*, CMS Series of monographs and Advanced Books in Mathematics, John Wiley, Hoboken, NJ, 1987.

[20] R.P. Brent, *The Borwein brothers, Pi and the AGM,* in: Bailey D. *et al.* (eds) From Analysis to Visualization. JBCC 2017. Springer Proceedings in Mathematics & Statistics, **313**, Springer, Cham., 2020, pp. 323–347, `https://doi.org/10.1007/978-3-030-36568-4_21`.

[21] D.J. Broadhurst, *Massive 3-loop Feynman diagrams reducible to SC* primitives of algebras of the sixth root of unity*, Eur. Phys. J. C., **8** (1999), pp. 311–333, `https://doi.org/10.1007/s100529900935`.

[22] I. Castella-McDonald, *New record for pi calculation: Down to 62.8 trillion digits,* Impakter, 18 Aug 2021, available at `https://impakter.com/new-record-for-pi-62-trillion-digits/`.

[23] D. Chudnovsky, G. Chudnovsky, *Listing of Ramanujan-type formulas*, copy in author's possession, 2000.

[24] H.R.P. Ferguson, D.H. Bailey, S. Arno, *Analysis of PSLQ, an integer relation finding algorithm*, Math. Comp., **68** (1999) no. 225, 351–369, `https://doi.org/10.1090/S0025-5718-99-00995-3`.

[25] L. Fousse, G. Hanrot, V. Lefevre, P. Pelissier, P. Zimmermann, *MPFR: A multiple-precision binary floating-point library with correct rounding*, ACM Trans. Math. Soft., **33** (2007) no. 2, `https://doi.org/10.1145/1236463.1236468`.

[26] J. Guillera, *Some binomial series obtained by the WZ-method,* Adv. Appl. Math., **29** (2002) no. 4, 599–603 `https://doi.org/10.1016/S0196-8858(02)00034-9`.

[27] *"Johann Heinrich Lambert"*, Wikipedia article, viewed 8 Dec 2021, available at `https://en.wikipedia.org/wiki/Johann_Heinrich_Lambert`.

[28] *"Ferdinand von Lindemann"*, Wikipedia article, viewed 8 Dec 2021, available at `https://en.wikipedia.org/wiki/Ferdinand_von_Lindemann`.

[29] MPFR research team, *Comparison of multiple-precision floating-point software*, accessed 12 Mar 2020, `https://www.mpfr.org/mpfr-4.0.1/timings.html`.

[30] J. Palmer, *Pi record smashed as team finds two-quadrillionth digit*, BBC News, 16 Sep 2010, available at `https://www.bbc.com/news/technology-11313194`.

[31] A.J. Yee, *y-cruncher - A multi-threaded pi-program*, updated 12 Mar 2020, `http://www.numberworld.org/y-cruncher/`.

Received by the editors: December 5, 2021; accepted: December 29, 2021; published online: February 17, 2022.