Progress in Computing Pi, 250 BCE to the Present

From Polygons to Elliptic Integrals

Donald Byrd
Woodrow Wilson Indiana Teaching Fellow
and
Research Technologies and School of Informatics
Indiana University Bloomington

rev. late April 2014

The history of computing π offers a high-level view of several aspects of mathematics and its history. It also says something about aspects of computing and the history of computers. And, since the practical value of more accurate approximations to π was exhausted by 1706, if not earlier, and the mathematical value of high-precision approximations is almost zero, it says something about human culture. In this article, I give background for and outline the four fundamentally different techniques (measurement and three methods of calculation) of finding approximations that have been used, and I discuss some surprising facts about what it really takes, other than infinite series that converge with astounding rapidity, to compute π to the trillions of decimal places that are now known.

This article assumes mathematical background about the level of what used to be called "college algebra" but is now known as precalculus. Two non-obvious conventions it uses are: (1) Numbers of decimal places or approximate values for π in ***boldface italics*** indicate values that were world records when they were found. (2) References giving only a page number are to Beckmann [2].

The record for accuracy in computing approximations to π now stands at ***12.1 trillion decimal places*** [15]. There's a great deal of history behind this remarkable feat. Indeed, Beckmann comments in his fascinating book *A History of Pi* [2]*,* "The history of π is a quaint little mirror of the history of man." (p. 3)[1] But I'll concentrate in this short paper exclusively on its *mathematical* history, a history that divides into periods—perhaps "phase" is a better term, since the last two overlap—surprisingly well.

Phase 0 (before ca. 250 BCE): Measurement. In earliest times, of course, no method for actually *computing* π was known. The first values were found by the Egyptians, Babylonians, and others by measuring the circumferences and diameters of circles, and dividing the former by the latter.

[1] My references are to the third (1974) edition of this book. However, based on what the preface to the third edition says, that "edition" is essentially just the second edition with some minor errors corrected. In fact, the title page of an old copy of what is actually the third edition—as indicated by the presence of the third-edition preface—claims it's the second edition! A more recent copy of the alleged third edition simply removes that claim; the title page still does not say it's the third edition, but it includes the same "third edition" preface as the old copy.

Phase 1 (ca. 250 BCE to 1654): The “method of exhaustion”, using polygons circumscribed and inscribed in a circle (Archimedes and others). Borwein et al. [4] comment that “despite the long pedigree of the problem, all nonempirical calculations have employed, up to minor variations, only three techniques”; this was the first.

Phase 2 (ca. 1665 to 1990): The discovery of increasingly efficient infinite series for computing π, starting with algorithms based on calculus (Newton, John Machin, Euler, and others) and ending with those based on the transformation theory of elliptic integrals (following up work by Gauss, Legendre, and Ramanujan). These are Borwein et al.’s second and third techniques, respectively.

Phase 3 (ca. 1930 to the present, overlapping phase 2): The application of increasingly powerful digital computers (including electromechanical calculators, the immediate predecessors of digital computers) to the problem, using Borwein et al.’s second and third techniques.

Needless to say, not much precision could be obtained by measurement; the most accurate value of the first period may have been ***25/8 ≅ 3.125,*** apparently known to the Babylonians by around 1800 BCE [13] (though they and other ancient peoples generally used less accurate values), and about 0.01659 too low.

Even in the purely mathematical history of computing π, there is far more than can be covered in one article. I’ll say no more about the “measurement” phase, but will summarize the other phases, focusing on one major event or feature of each.

Infinite Series and Rapidity of Convergence

As I have said, two of the three known families of techniques for computing π involve infinite series. For an infinite series to be usable for computing anything, it must, of course, converge. For practical use, it must converge fairly rapidly. And to compute something to billions of significant figures, much less the trillions to which π is now known, the series must converge *extremely* rapidly. The history of computing π furnishes extraordinary demonstrations of this fact. To fully appreciate how much more quickly modern π-computing series converge than the earliest ones requires an idea from numerical analysis: *order of convergence. Linear* convergence means that for each iteration you get the same number of new digits. For example, it might be 10 new correct digits per iteration: after one iteration, 10 correct digits; after two iterations, 20 correct; after three, 30 correct, etc. *Quadratic* convergence means the number of new correct digits doubles each iteration. Thus, the first iteration might provide 4 correct digits; the second, 8 new correct digits; the third, 16; and so forth. *Cubic* convergence means the number of new correct digits triples each time. These are examples of *superlinear* convergence, where the number of new correct digits per iteration grows without bound. By contrast, there is also *sublinear* convergence, where the number of iterations needed to get one new correct digit grows without bound.

To sum up, with *linear* convergence, the amount of computation per digit is constant; with *superlinear* convergence, the computation per digit goes down and down; with *sublinear* convergence, computation per digit goes up and up. (For technical details, see the Appendix: Rapidity of Convergence.) Here is a striking example of the contrast. The first π-computing series ever found, which I call the “Madhava–Gregory–Leibniz” series, converges so slowly that, if an efficient implementation on the fastest supercomputer now in existence had started running at the Big Bang, it would have computed no more than about 65 decimal places so far[2]. On the other hand, a reasonable implementation of a state-of-the-art series running on an ordinary desktop computer can produce millions of digits of π in under 30 seconds.[3] Each term of the Madhava–Gregory–Leibniz series, though it can be computed very quickly, gives you very little information—in fact, less and less as you go on; by contrast, each successive term of a modern series, though it takes considerably more computation, gives you more and more information. In the terminology discussed above, the Madhava–Gregory–Leibniz series exhibits sublinear convergence; the modern series, superlinear convergence.

Phase 1. The Method of Exhaustion

Around 250 BCE, Archimedes devised the first way to obtain π to any desired accuracy, given enough patience, by applying the method of exhaustion invented by Eudoxus a century earlier. Archimedes realized that the perimeters of regular polygons circumscribed about and inscribed within a circle give upper and lower bounds for π. The more sides the polygon has, the more closely it approximates the circle—"exhausting" the space between the circle and the polygon; hence the name—and the tighter the bounds. (Strogatz [10] gives an unusually clear and readable description, addressed to the layperson, of both this process and the way Archimedes found the area of the circle.) The results can easily be computed without trigonometry; in the words of [4]:

> [Recursively] calculate the length of circumscribed and inscribed regular 6 x 2^n-gons about a circle of diameter 1. Call these quantities a_n and b_n, respectively. Then $a_0 := 2\sqrt{3}$ and $b_0 := 3$ and, as Gauss' teacher Pfaff discovered in 1800,
>
> $$a_{n+1} = \frac{2a_n b_n}{a_n + b_n} \quad \text{and} \quad b_{n+1} = \sqrt{a_{n+1} b_n}$$

This process converges to π, with the bounds a_n and b_n tightening by a factor of four on each iteration [3]. (Thus, the sequence of means converges linearly, with a rate of convergence of 1/4.) With

[2] Beckmann [2] (p. 140) cites an estimate that "to obtain 100 decimal places, the number of necessary terms would be no less than 10^{50}!" At first glance, that number seems far too low: the partial sums after 10^{50} and $10^{50} + 1$ terms differ by about 10^{-50}, i.e., after only about 50 decimal places. However, the arithmetic mean of two consecutive partial sums of the Madhava–Gregory–Leibniz series is accurate to roughly the product of the last two terms, giving about twice as many correct decimal places. So, for 60 decimal places, 10^{30} terms is a reasonable estimate; for 70 places, 10^{35} terms. Now, the age of the universe is around 4.3 x 10^{17} sec. As of this writing, the fastest supercomputer in existence does about 34 petaFLOPS, i.e., 3.4 x 10^{16} floating-point operations per sec. It's not easy to convert this figure to an estimate of speed evaluating the Madhava–Gregory–Leibniz series, but assuming the equivalent of 10 floating-point operations per 100-digit term —surely far too low—yields 3.4 x 10^{15} terms per sec. But even at that rate, fewer than 1.5 x 10^{33} terms could have been produced by now.

[3] A Java version of T. Ooura's program pi_css5, which is nowhere near the fastest program available [6], running on an 2.4 GHz MacBook found over 4 million places in 27 sec. See the table under Phase 3, below.

polygons of 96 sides (i.e., $n = 4$ in the above formulation), Archimedes obtained ***223/71 < π < 22/7;*** in decimal form, roughly ***3.140845 < π < 3.142857*** (average ≅ 3.141873).[4]

For the next 1800 years, others improved on his results simply by using much larger numbers of sides. Among them, in the 5th century CE, the Chinese Tsu Ch'ung Chi found ***7 decimal places;*** in 1615, after years of effort, the Dutchman Ludolph van Ceulen found ***35 places,*** using polygons of no less than 2^{62} (roughly $4.6 \cdot 10^{18}$) sides! A few years later (1621) another Dutchman, Willebrord Snellius, observed that using regular polygons to approximate circles in a different way makes it possible to bring the bounds for a given number of sides much closer together (p. 111). Where Archimedes had gotten only 2 decimal places from polygons of 96 sides, Snellius obtained $3.1415926272 < \pi < 3.1415928320$, for 6 places.

Phase 2. Infinite Series from Calculus and Transformations of Elliptic Integrals

The invention of differential calculus by Newton, Leibniz, and others in the mid-17th century led to the discovery of a long list of infinite series for computing π, all (or almost all) exploiting the Taylor series for trigonometric functions [1, 2]. Much later, the transformation theory of elliptic integrals provided a basis for a very different set of series to compute π.

The earliest series for π is a special case of the series for arctangent, known to Indian mathematicians by the early 1500's, but apparently discovered as early as 1400 by Madhava of Sangamagrama [7]. This early Indian work was unknown in the west until quite recently; the arctangent series was rediscovered in 1671 by James Gregory and a few years later by Leibniz (p. 132–133). The series is:

$$\arctan x = 1 - \frac{x^3}{3} + \frac{x^5}{5} - \frac{x^7}{7} + \ldots + \frac{(-1)^n x^{2n+1}}{2n+1} + ...$$

Since arctan 1 = π/4, substituting 1 for x yields:

$$\frac{\pi}{4} = 1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \ldots + \frac{(-1)^n}{2n+1} + ...$$

Unfortunately, this elegant series—most often called the "Gregory", "Leibniz", "Madhava", or some combination of those, formula—converges sublinearly. In fact, as I have already said, it converges so slowly, it's virtually useless for calculation. The number of terms needed to compute n decimal places increases exponentially, specifically as $10^{n/2}$, so that adding 12 decimal places (for example) requires increasing the number of terms by a factor of 10^6. But there are many ways to improve its performance.

The "digit hunters" (as [2] calls them) of the 17th and 18th centuries used the Madhava–Gregory–Leibniz series, which they modified in different ways to accelerate its convergence, or a series for arctangent discovered by Euler (p. 101). The astronomer Abraham Sharp computed ***72 decimal places*** (p. 102) merely by using the Madhava–Gregory–Leibniz series for π/6, i.e., $x = \frac{1}{\sqrt{3}}$. After factoring

[4] Some evidence exists that Archimedes went further and obtained considerably more accurate results. See [1], p. 175, or [2], p. 66.

out $\frac{1}{\sqrt{3}}$, this makes the general term $\frac{(-1)^n}{3^n \times (2n+1)}$. But another astronomer, one named Machin, did something much more far reaching.

Machin's Formula and Its Derivation

In 1706, John Machin, a professor of astronomy in London, came up with a strategy that both made the Madhava–Gregory–Leibniz series converge rapidly, and simplified the calculations it required. His formula states:

$$\frac{\pi}{4} = 4\tan^{-1}\frac{1}{5} - \tan^{-1}\frac{1}{239}$$

Substituting the Madhava–Gregory–Leibniz series for the arctangents gives:

$$\frac{\pi}{4} = 4\left(\frac{1}{5} - \frac{1}{3\times 5^3} + \frac{1}{5\times 5^5} - \ldots\right) - \left(\frac{1}{239} - \frac{1}{3\times 239^3} + \frac{1}{5\times 239^5} - \ldots\right)$$

As Beckmann ([2], p. 145) says, "This was a neat little trick, for the second series converges very rapidly, and the first is well suited for decimal calculations." But where does Machin's formula come from? Deriving it requires only the tangent double-angle and subtraction identities plus a bit of algebra.

Let tan β = 1/5. Then we have

$$\tan 2\beta = \frac{2\tan\beta}{1-(\tan\beta)^2} = \frac{\frac{2}{5}}{1-\frac{1}{25}} = \frac{\frac{2}{5}}{\frac{24}{25}} = \frac{5}{12}$$

and

$$\tan 4\beta = \frac{2\tan 2\beta}{1-(\tan 2\beta)^2} = \frac{2(\frac{5}{12})}{1-(\frac{5}{12})^2} = \frac{\frac{10}{12}}{\frac{119}{144}} = \frac{120}{119}$$

Now, tan (π/4) = 1, so $\tan^{-1}(4\beta)$ is already very nearly π/4; and $\tan(u-v) = \frac{\tan u - \tan v}{1 - \tan u \tan v}$.

Therefore

$$\tan(\frac{\pi}{4} - 4\beta) = \frac{\tan\frac{\pi}{4} - \tan 4\beta}{1 + \tan\frac{\pi}{4}\tan 4\beta}$$

$$= \frac{1-\tan 4\beta}{1+\tan 4\beta} = \frac{1-\frac{120}{119}}{1+\frac{120}{119}} = \frac{-\frac{1}{119}}{\frac{239}{119}} = -\frac{1}{239}$$

Continuing with some elementary algebra:

$$\tan\left(4\beta - \frac{\pi}{4}\right) = \frac{1}{239} \quad \text{, or, equivalently,} \quad 4\beta - \frac{\pi}{4} = \tan^{-1}\frac{1}{239} \text{ , hence}$$

$$4\beta - \tan^{-1}\frac{1}{239} = \frac{\pi}{4}$$

Finally, since tan β = 1/5, we have the desired formula:

$$4\tan^{-1}\frac{1}{5} - \tan^{-1}\frac{1}{239} = \frac{\pi}{4}$$

Machin used his new formula to compute π to 100 decimal places (p. 145), and in 1873, William Shanks used it to find a remarkable ***527 places*** (p. 103), a record that stood till 1946. (Shanks actually calculated 707 places, but only the first 527 were correct.)

Transformation Theory of Elliptic Integrals

According to [4], “the third technique, based on the transformation theory of elliptic integrals, provides the algorithms for the most recent set of computations.” Elliptic integrals are so called because they were originally studied in connection with finding the arc lengths of ellipses. Since a circle is an ellipse, their connection to π is plausible. Some of these algorithms are based on the early 19th-century work of Gauss and Legendre with the arithmetic/geometric mean, others on the more recent work of Srinivasa Ramanujan. In 1914, Ramanujan published no fewer than 19 new infinite series for 1/ π; perhaps the best known is this “amazing sum” [4]:

$$\frac{1}{\pi} = \frac{\sqrt{8}}{9801}\sum_{n=0}^{\infty}\frac{(4n)!}{(n!)^4}\frac{[1103 + 26390n]}{396^{4n}}$$

Following up his work many years later led to series for π that converge with stunning rapidity, series that converge with order 4 (quadrupling the number of correct digits with every term) or order 5 (quintupling the number of correct digits with every term).[5] However, the mathematics involved is far from simple, and discussing it is beyond the scope of this article.

Phase 3. Digital Computers

The advent of digital computers made the ease of base-10 computation of Machin’s series irrelevant, but its rapid convergence was advantage enough, and several “Machin-like” formulae dominated the computation of π from the time of its discovery till the 1970’s. In the late 1940’s, Shanks’ record was broken repeatedly by Ferguson and Wrench, who used Machin-like formulae with desk calculators—essentially, electromechanical computers with extremely limited programmability—to find several values, from ***710 to 1,120 places***. Then fully electronic digital computers became available and things changed dramatically. From a ***2,037-place value*** computed in 1949 by one of the first such computers, the ENIAC, the record shot up to ***1,001,250 places*** in 1970, thanks to one of the first supercomputers, the CDC 7600 [12].

[5] These series appear in [4], which also comments that convergence with order 5 is close to the theoretical optimum.

Since then, programs based on the transformation theory of elliptic integrals have wiped out all competition, producing first billions and then trillions of digits of π. The first programs of this type used the so-called Gauss-Legendre formula; the more recent programs are based on Ramanujan's work. The current record of ***12.1 trillion decimal places*** was set in 2013 by Shigeru Kondo and Alexander Yee, with the Ramanujan-based Chudnovsky formula for "main computation" [15]. But the phrase "main computation" implies that some other kind of computation is involved; what is it?

Other Factors in Computation

Computing π to an enormous number of decimal places obviously means doing arithmetic with enormously long numbers. Doing that accurately and efficiently brings up questions of numerical analysis and especially analysis of algorithms, two areas that are close to the boundary between mathematics and theoretical computer science. Aside from the "main computation" using series that converge incredibly rapidly, a major factor in the speed of the latest generation of π-computing programs is simply that they perform multiplication of numbers billions or trillions of digits long via unusual and extraordinarily efficient methods. Conventional multiplication has a *time complexity* of O(n^2); that is, the number of operations necessary to multiply two numbers of n digits each increases asymptotically as n^2, so that doubling the number of digits quadruples the number of operations necessary. This is probably what most people would intuitively expect, but it's possible to do a great deal better. The well-known Fast Fourier Transform (FFT) is one example of an algorithm that *can* do much better (Borwein 1989), though it's rarely used for this purpose. Kondo and Yee's 12.1 trillion decimal-place calculation relied on Alexander Yee's program "y-cruncher", which is essentially a framework for computing irrational numbers to extremely high precision with great efficiency. Indeed, y-cruncher uses the FFT as well as several other algorithms to do arithmetic (Yee 2011). In addition to π, it has been used to compute *e* to a trillion digits, and several other constants—the square root and natural log of 2, Apery's constant, etc.—to many billions of digits.

It seems unlikely that most of the programs based on Machin-like formulae do arithmetic in unusual ways. But writing a program to compute anything very accurately involves other practical considerations, for example, avoiding fractions for as long as possible in order to minimize roundoff error. With Machin's formula, one implication is the need to substitute $\cot^{-1}(5)$ and $\cot^{-1}(239)$ for the original $\tan^{-1}(1/5)$ and $\tan^{-1}(1/239)$.

Finally, of course, the programming language and (to a lesser extent) compiler used can make a big difference. Yee's y-cruncher is written in C++, which generally runs very quickly as compared to more recent languages like Java, C-sharp, and Python. A major reason that these "more recent languages" tend to run slowly is that they are generally compiled to instructions, not for the actual computer hardware, but for a virtual machine; then those instructions are interpreted. In fact, y-cruncher was originally in Java. Though Yee doesn't say, it seems likely the rewrite was motivated primarily by the desire for more speed.

Comparison of Three Methods and Programs

Here's a simple comparison of three ways to compute π, and programs for each. The programs are:

• CalcPiGregory: my own straightforward implementation of the Madhava-Gregory series, written in Java, with Java's standard arithmetic.

• CalcPiMachin: my own adaptation of a straightforward implementation of Machin's formula, written in Java, and doing arithmetic via BigDecimal, the standard Java arbitrary-precision arithmetic package.

• pi_fftcs (also known as pi_css5): an implementation of the Gauss-Legendre formula by T.Ooura, using FFTs for arithmetic, converted to Java by Hazeghi [6]. Despite its overwhelming increase in efficiency over even Machin's formula, recall that algorithms based on Ramanujan's work can be even faster.

The first two programs are available at http://www.informatics.indiana.edu/donbyrd/Teach/math/ . pi_fftcs is available at http://www.kurims.kyoto-u.ac.jp/~ooura/pi_fft.html .

Timings were done on MacBook with a 2.4 GHz Intel Core 2 Duo CPU and 2 GB of RAM, averaging three runs for each.

Method	*Terms for 100 dec. places*	*Terms for 100K dec. places*	*Implementation*	*Approx. lines of code*	*Execution time for 100K places*
Madhava-Gregory series	$\geq 10^{50}$	$10^{50,000}$(?)	CalcPiGregory	100	--
Machin's formula	77 + 23	71,537 + 21,024	CalcPiMachin: in Java, with Java's standard arbitrary-precision arithmetic	120	40.5 sec.
Gauss-Legendre formula, AGM	3 (176 places)	13 (131,072 places)	pi_fftcs/pi_css5: in Java, with arithmetic via FFT	2,300	1.62 sec.

Conclusions

The obvious question is: Why would anyone wish to compute thousands or millions of digits of π, much less 12.1 trillion? Certainly not for any ordinary scientific or technical purpose. In the words of the American astronomer Simon Newcomb, "Ten decimal places of π are sufficient to give the circumference of the earth to a fraction of an inch, and thirty decimal places would give the circumference of the visible universe to a quantity imperceptible to the most powerful microscope." (This statement is widely attributed to Newcomb and, as far as I know, never to anyone else; but I've been unable to find a reliable source for it.) Since Newcomb's death in 1909, microscope technology has improved tremendously, and the "visible universe" is larger as well, so nowadays we'd need to work harder to elude the scrutiny of "the most powerful microscope": 40 decimal places should suffice.[6]

[6] It seems reasonable to equate Newcomb's "visible universe" with what is now called the "observable universe". The diameter of the observable universe is estimated at about 100 billion light years (this is possible despite the fact that only about 13 billion years have elapsed since the Big Bang because expansion of the early universe was not strictly limited by the speed of light). On the other end of the

The closest thing to a real-world application of vast numbers of digits of π is probably testing newly-developed or overclocked processors. In the words of Wikipedia (2011), "Overclocking is the process of operating a computer component at a higher clock rate (more clock cycles per second) than it was designed for or was specified by the manufacturer. This is practiced more by enthusiasts than professional users seeking an increase in the performance of their computers, as overclocking carries risks of less reliable functioning and damage." The program SuperPi has been popular for this application for years. But it should be pointed out that, while testing such hardware requires an enormous amount of computation, and while calculating π is a handy example of a process that does that, it is by no means the only task that does so.

On the purely mathematical side, people have been wondering for a long time about issues such as whether π is a *normal* number base 10, i.e., whether it has asymptotically equal distribution of digits and of all digit sequences of a given lengths in base 10. Of course no computation to a finite number of digits can decide questions like this, but they *can* provide hints that might lead eventually to proofs one way or the other. (To my knowledge, all evidence so far suggests that π *is* normal. Wagon's paper [11] is from 1985, but more recent publications, e.g., [1] seem to agree.)

A more detailed and discussion of this topic appears under the heading "Why do we care?" in Offner's very interesting article [8].

Challenges for Students

The expansion of π to a large number of decimal places fascinates many people. While teaching high-school algebra, I once offered extra credit to any student who memorized from 20 to 50 decimals place. Quite a few took me up on the offer, and several clearly enjoyed it. But there are several ways in which computing π could be used in either the mathematics or the computer-programming classroom.

- For elementary calculus, of course, it offers many examples of convergent infinite series, some simple and similar to well-known series, some less so.
- For numerical analysis, the rapidity of convergence of series is an important topic.
- For programming, implementing almost any algorithm in "standard" precision in any language is fairly straightforward. Implementing in arbitrary precision is more challenging, and implementing an algorithm in such a way as to be able to compute, say, billions of decimal places in a reasonable amount of time is considerably more so.

Acknowledgements

This article originated several years ago as a paper for a course in the history of mathematics taught by Jeffrey Watt of IUPUI; he was enthusiastic about this topic, for which I'd like to thank him.

scale, transmission electron microscopy has a resolution limit of around 0.05 nanometers. A light year is about 9.46×10^{15} meters, so 10^{11} light years is about 9.46×10^{26} meters; the ratio of that distance to 5×10^{-11} meters is less than 10^{38}. One could argue that the appropriate "yardstick" at the small end of the scale is the Planck distance; that brings in another 20 or so orders of magnitude, which still limits the number of decimal places of π needed to only 60 or so. [8] quotes several other estimates of the number of useful places of π.

Douglas Hofstadter, Jefferson Davis, Frank Lester, Douglas McKenna, John L. Miller, and Rebecca Platt offered additional enthusiasm and helpful comments. Thanks to all.

References

[1] Jörg Arndt & Christoph Haenel, tr. by C. Lischka & D. Lischka, *Pi Unleashed,* Springer, 2001.

[2] Petr Beckmann, *A History of Pi,* 3rd ed., St. Martins Press, 1974.

[3] Jonathan M. Borwein, The Life of Pi: History and Computation, available at http://www.cecm.sfu.ca/%7Ejborwein/pi-slides.pdf, 2003.

[4] J.M. Borwein, P.B. Borwein, & D.H. Bailey, Ramanujan, Modular Equations, and Approximations to Pi, or How to Compute a Billion Digits of Pi, *American Mathematical Monthly* 96, 3 (1989); available at mathdl.maa.org/images/upload_library/22/.../BorweinBorweinBailey.pdf.

[5] Xavier Gourdon & Pascal Sebah, π and its computation through the ages, available at http://numbers.computation.free.fr/Constants/constants.html, 2010.

[6] Dara Hazeghi, Pi Calculation Programs, available at http://myownlittleworld.com/miscellaneous/computers/pilargetable.html, 2002.

[7] MacTutor History of Mathematics archive, Madhava, available at http://www-groups.dcs.st-and.ac.uk/~history/Biographies/Madhava.html, 2011.

[8] Carl D. Offner, Computing the Digits in π , available at www.cs.umb.edu/~offner/files/pi.pdf, 2009.

[9] Srinivasa Ramanujan, Modular Equations and Approximations to π. *Quarterly Journal of Mathematics* **45** (2011). 350–372.

[10] Steven Strogatz, Take It to the Limit, available at http://opinionator.blogs.nytimes.com/2010/04/04/take-it-to-the-limit/, 2010.

[11] Stan Wagon. Is Pi Normal? *Mathematical Intelligencer* **7** (1985), 65–67.

[12] Wikipedia, Chronology of computation of Pi, available at http://en.wikipedia.org/wiki/Chronology_of_computation_of_π, 2013.

[13] Wikipedia, Rate of convergence, available at http://en.wikipedia.org/wiki/Rate_of_convergence, 2011.

[14] Alexander Yee, y-cruncher: a Multi-Threaded Pi Program, available at http://www.numberworld.org/y-cruncher/, 2011.

[15] Alexander Yee & Kondo, Shigeru, 12.1 Trillion Digits of Pi: And we're out of disk space, available at http://www.numberworld.org/misc_runs/pi-12t/, 2013.

Appendix: Rapidity of Convergence

The following discussion refers to sequences, while we are interested mostly in infinite series; but we can apply the same terminology simply by considering the sequence of partial sums of each series. The terms below are standard in numerical analysis; see any textbook, or [13].

The speed at which a convergent sequence approaches its limit is described by its *order of convergence* and, in many cases, *its rate of convergence.* In particular, suppose that the sequence $\{x_k\}$ converges to the number L. We say that this sequence **converges linearly** to L if there exists a number μ with $0 < \mu < 1$ such that

$$\lim_{k\to\infty} \frac{|\, x_{k+1} - L \,|}{|\, x_k - L \,|} = \mu \ .$$

If we think of $|\, x_k - L \,|$ as the error at term k, the ratio in the above expression is the factor by which the error is changing at that point; the smaller the ratio, the faster the error is disappearing. The number μ is called the **rate of convergence**. However, if

- $\mu = 0$, then we say the sequence **converges superlinearly**;
- $\mu = 1$, then we say the sequence **converges sublinearly**.

Furthermore, we say that a superlinearly-convergent sequence converges with order q to L for $q > 1$ if

$$\lim_{k\to\infty} \frac{|\, x_{k+1} - L \,|}{|\, x_k - L \,|^q} = \mu \mid \mu > 0 \ .$$

In particular, convergence with order

- $q = 2$ is called **quadratic convergence**,
- $q = 3$ is called **cubic convergence**,
- etc.

Of course, **linear convergence** is convergence with order 1. This is somewhat counterintuitive, since every geometric sequence that converges has linear convergence, but it’s linear in that the number of new correct digits per term is constant.