1 Iterative algorithms

Modern iterative algorithms are among the most efficient ways to compute billion of digits of π. The first of those algorithms were discovered during the years 1970. We give without proof a selection and a brief description of some important iterations.

1.1 Early iterations

Before the discovery of extremely fast converging modern iterations, a few algorithms were already known but with slow rates of convergence.

1.1.1 Viete

Based on the well known formula discovered by François Viète (1540-1603)

$$\frac{2}{\pi} = \left(\sqrt{\frac{1}{2}}\right)\left(\sqrt{\frac{1}{2}+\frac{1}{2}\left(\sqrt{\frac{1}{2}}\right)}\right)\left(\sqrt{\frac{1}{2}+\frac{1}{2}\left(\sqrt{\frac{1}{2}+\frac{1}{2}\sqrt{[1/2]}}\right)}\right)\ldots$$

an algorithm may be written as

$$\begin{aligned} x_{k+1} &= x_k y_k \\ y_{k+1} &= \left(\sqrt{\frac{2y_k}{y_k+1}}\right) \end{aligned}$$

and

$$\begin{aligned} x_0 &= 1 \\ y_0 &= \sqrt{2} \end{aligned}$$

then

$$\lim_{k \to \infty} x_k = \frac{\pi}{2}.$$

It converges with a *geometrical* rate and for example it produces

$$
\begin{aligned}
2x_1 &= 2.8284... \\
2x_2 &= 3.(061...) \\
2x_3 &= 3.1(214...) \\
2x_4 &= 3.1(365...) \\
2x_5 &= 3.14(033...)
\end{aligned}
$$

1.1.2 Archimedes-Pfaff

Archimedes' method is equivalent to the *Pfaff-algorithm* published in 1800 (Pfaff was Gauss's teacher).

$$
\begin{aligned}
x_{k+1} &= \frac{2x_k y_k}{x_k + y_k} \\
y_{k+1} &= \sqrt{x_{k+1} y_k}
\end{aligned}
$$

starting with

$$x_0 = 2\sqrt{3}, y_0 = 3,$$

then

$$\lim_{k \to \infty} x_k = \lim_{k \to \infty} y_k = \pi$$

again, the rate of convergence is *geometric* with an error $O(4^{-k})$ (that is you need roughly about 5 iterations to obtain 3 new digits). It can be accelerated by

$$\lim_{k \to \infty} z_k = \lim_{k \to \infty} \left(\frac{1}{3} x_k + \frac{2}{3} y_k \right) = \pi$$

which also converges geometrically, but with a faster rate of convergence ($O(16^{-k})$). The first iterates of the two sequences are

$$
\begin{aligned}
y_1 &= 3.1(05828541230249...), z_1 = 3.14(2349130544656...) \\
y_2 &= 3.1(32628613281238...), z_2 = 3.141(639056219992...)
\end{aligned}
$$

$$y_3 = 3.1(39350203046867...), z_3 = 3.14159(5540408389...)$$
$$y_4 = 3.141(031950890509...), z_4 = 3.141592(833808795...)$$
$$y_5 = 3.141(452472285462...), z_5 = 3.1415926(64850249...)$$

1.2 Quadratic

There are a few *quadratic* algorithms converging to the constant π. Quadratic convergence means that the number of correct digits is doubled at each iteration. The existence of such extraordinarily rapid algorithms using only basic operations (multiplication, addition, division) and square roots extraction is a remarkable fact.

1.2.1 First algorithm

The first known quadratic algorithm was published in 1976 by Eugene Salamin in [6] and it was also discovered independently by Richard Brent the same year [5]. It uses the real AGM iteration.

Algorithm

$$x_{k+1} = \frac{1}{2}(x_k + y_k)$$
$$y_{k+1} = \sqrt{x_k y_k}$$
$$a_{k+1} = a_k - 2^{k+1}(x_{k+1}^2 - y_{k+1}^2)$$

with the initial values

$$x_0 = 1, y_0 = \frac{1}{\sqrt{2}}, a_0 = \frac{1}{2}$$

then

$$\lim_{k \to \infty} p_k = \lim_{k \to \infty} \frac{2x_k^2}{a_k} = \pi.$$

The first four iterations are

$$p_1 = 3.1(876...)$$

$$p_2 = 3.141(680...)$$

$$p_3 = 3.141592653(895...)$$

$$p_4 = 3.14159265358979323846(636...)$$

and computing p_{30} will provide more than a billion digits of π ! So the computation of only 30 square roots and less than a hundred multiplications is required to find a huge number of digits.

1.2.2 Second algorithm

This second algorithm is due to Jonathan and Peter Borwein and was published in 1984 [1].

Algorithm

$$x_{k+1} = \frac{1}{2}\left(\sqrt{x_k} + \frac{1}{\sqrt{x_k}}\right)$$

$$y_{k+1} = \sqrt{x_k}\left(\frac{y_k+1}{y_k+x_k}\right)$$

$$a_{k+1} = a_k y_{k+1}\left(\frac{x_{k+1}+1}{y_{k+1}+1}\right)$$

starting with

$$x_0 = \sqrt{2}, y_0 = 0, a_0 = 2+\sqrt{2}$$

then

$$|a_k - \pi| < 10^{-2^k}.$$

The rate of convergence is also *quadratic* and the first iterations are

$$a_1 = 3.14(260...)$$

$$a_2 = 3.1415926(609...)$$

$$a_3 = 3.14159265358979323238(645...)$$

1.2.3 Third algorithm

In [2], the following algorithm is given :

Algorithm

$$y_{k+1} = \frac{1-\sqrt{1-y_k^2}}{1+\sqrt{1-y_k^2}}$$

$$a_{k+1} = (1+y_{k+1})^2 a_k - 2^{k+1} y_{k+1},$$

starting with

$$y_0 = 1/\sqrt{2}, a_0 = 1/2,$$

then

$$0 < a_k - \frac{1}{\pi} < 16.2^k e^{-\pi.2^k}.$$

The rate of convergence is again *quadratic* (the number of digits is doubled at each iteration). The first iterates are given by

$$1/a_1 = 2.914213562373095048801688 7...$$

$$1/a_2 = 3.14(057...)$$

$$1/a_3 = 3.1415926(462...)$$

$$1/a_4 = 3.141592653589793238(279...)$$

and $1/a_5$ has 40 correct digits, $1/a_6$ has 83 correct digits, ...

1.3 Cubic

In [3], a cubic algorithm is described :

Algorithm

$$x_{k+1} = \frac{3}{1+2(1-y_k^3)^{1/3}}$$

$$y_{k+1} = \frac{x_{k+1}-1}{2}$$

$$a_{k+1} = x_{k+1}^2 a_k - 3^k(x_{k+1}^2-1),$$

with the initial values

$$y_0 = (\sqrt{3}-1)/2, a_0 = 1/3,$$

then

$$\lim_{k\to\infty} \frac{1}{a_k} = \pi$$

and the rate of convergence is now *cubic* (the number of digits is tripled at each iteration). Again let's take a look to the first iterations

$$1/a_1 = 3.14159(058...)$$

$$1/a_2 = 3.14159265358979323846 2(359...)$$

and $1/a_3$ has 70 correct digits...

1.4 Quartic

Two iterations of the third quadratic algorithm gives a new iteration [2] :

Algorithm

$$y_{k+1} = \frac{1-(1-y_k^4)^{1/4}}{1+(1-y_k^4)^{1/4}}$$

$$a_{k+1} = (1+y_{k+1})^4 a_k - 2^{2k+3} y_{k+1}(1+y_{k+1}+y_{k+1}^2),$$

starting with

$$y_0 = \sqrt{2}-1, a_0 = 6-4\sqrt{2},$$

then

$$0 < a_k - \frac{1}{\pi} < 16.4^k e^{-2\pi.4^k}$$

and it has the impressive *quartic* rate of convergence (each successive iteration quadruples the number of correct digits). The first two iterates are

$$1/a_1 = 3.1415926(462...)$$

$$1/a_2 = 3.14159265358979323846264338327950288419 71(146...)$$

This algorithm is among the most efficient and is often used with a quadratic one to compute billions of digits of the constant π (one is used for the computation and the other is used to check the result of the first one).

1.5 Quintic

In [4], this remarkable algorithm is given :

Algorithm

$$y_{k+1} = \frac{25}{y_k(c+a/c+1)^2}$$

$$a = \frac{5}{y_k}-1, b = (a-1)^2+7, c = \left(\frac{1}{2}a(b+\sqrt{b^2-4a^3})\right)^{1/5}$$

$$a_{k+1} = y_k^2 a_k - 5^k\left(\frac{1}{2}(y_k^2-5) + \sqrt{y_k(y_k^2-2y_k+5)}\right),$$

starting with

$$y_0 = 5(\sqrt{5}-2), a_0 = 1/2,$$

we have

$$0 < a_k - \frac{1}{p} < 16.5^k e^{-p.5^k}$$

and the rate of convergence is *quintic* (the number of digits is multiplied by five at each iteration).

$$1/a_1 = 3.1415(369...)$$

$$1/a_2 = 3.141592653589793238462643383279(351...)$$

More complex algorithms with higher order are available like *septic* or *nonic* algorithms (in the nonic iteration the number of correct digits is multiplied by 9 at each step !).

1.6 Other algorithms

Starting with any approximation of p, the following algorithm was given by Beeler in 1972 and has a cubic convergence

$$a_{k+1} = a_k + \sin(a_k)$$

and for example using Archimedes' approximation

$$a_0 = \frac{22}{7}$$

it gives

$$a_1 = 3.141592653(926...)$$

$$a_2 = 3.1415926535897932384626433832(858...)$$

and a_3 has 88 correct digits,...

It's possible to find algorithms of this kind *at any order* and the following iterations have respectively *quintic, septic, nonic, ...* order of convergence

$$a_{k+1} = a_k + \sin(a_k) + \frac{1}{6}\sin^3(a_k)$$

$$a_{k+1} = a_k + \sin(a_k) + \frac{1}{6}\sin^3(a_k) + \frac{3}{40}\sin^5(a_k)$$

$$a_{k+1} = a_k + \sin(a_k) + \frac{1}{}\sin^3(a_k) + \frac{3}{}\sin^5(a_k) + \frac{5}{}\sin^7(a_k)$$

6 40 112

the coefficients are those of the series expansion of the arcsine function. If one prefers to evaluate tangent functions we also have

$$a_{k+1} = a_k - \tan(a_k)$$

$$a_{k+1} = a_k - \tan(a_k) + \frac{1}{3}\tan^3(a_k)$$

$$a_{k+1} = a_k - \tan(a_k) + \frac{1}{3}\tan^3(a_k) - \frac{1}{5}\tan^5(a_k)$$

$$a_{k+1} = a_k - \tan(a_k) + \frac{1}{3}\tan^3(a_k) - \frac{1}{5}\tan^5(a_k) + \frac{1}{7}\tan^7(a_k)$$

with respectively *cubic, quintic, septic* and *nonic* convergence and the coefficients are those of the expansion of the arctan function. For example, starting with the famous approximation $a_0 = 355/113$, just one iteration of the nonic iterations gives 60 correct digits of p !. Similar formulae exist with cosines.

The major drawback of those algorithms is to evaluate sine or tangent function at a given accuracy which is not so easy.

References

[1] J.M. Borwein and P.B. Borwein, *The Arithmetic-Geometric Mean and Fast Computation of Elementary Functions,* SIAM review, (1984), vol. 26, p. 351-366

[2] J.M. Borwein and P.B. Borwein, *Pi and the AGM - A study in Analytic Number Theory and Computational Complexity*, A Wiley-Interscience Publication, New York, (1987)

[3] J.M. Borwein and P.B. Borwein, *Ramanujan and Pi*, Scientific American, (1988), p. 112-117

[4] J.M. Borwein, P.B. Borwein and D.H. Bailey, *Ramanujan, Modular Equations, and Approximations to Pi or How to Compute One Billion Digits of Pi*, The American Mathematical Monthly, (1989), vol. 96, p. 201-219

[5] R.P. Brent, *Fast multiple-Precision evaluation of elementary functions*, J. Assoc. Comput. Mach., (1976), vol. 23, p. 242-251

[6] E. Salamin, *Computation of* p *Using Arithmetic-Geometric Mean*, Mathematics of Computation, (1976), vol. 30, p. 565-570

Back to *The constant pi*

File translated from T_EX by T_TH, version 2.32.
On 17 Aug 2000, 21:43.