# Approximating square roots using Pascal's triangle

Andrés Casillas García de Presno

July 2025

## 1 Framework

Let $a_0 = 1$, $b_0 = 1$, $k \in \mathbb{Z}^+, k > 1$ and consider the following system of recursions

$$a_{n+1} = a_n + kb_n \tag{1}$$
$$b_{n+1} = a_n + b_n \tag{2}$$

$\forall n \in \mathbb{N}$.

In this article we prove that

$$lim_{n\to\infty} \frac{a_n}{b_n} = \sqrt{k}$$

and prove a connection of this algorithm with Pascal's triangle through the following closed forms

$$a_n = \sum_{i\geq 0} \binom{n+1}{2i} k^i$$
$$b_n = \sum_{i\geq 0} \binom{n+1}{2i+1} k^i$$

$\forall n \in \mathbb{N}$.

These can be nicely visualized by coloring $a_n$ and $b_n$ in Pascal's triangle

$$\begin{array}{ccccccccccc}
 & & & & & 1 & & & & & \\
 & & & & 1 & & 1 & & & & \\
 & & & 1 & & 2 & & 1 & & & \\
 & & 1 & & 3 & & 3 & & 1 & & \\
 & 1 & & 4 & & 6 & & 4 & & 1 & \\
 & & & & & \vdots & & & & & \\
\binom{n+1}{0}k^0 & & \binom{n+1}{1}k^1 & & \binom{n+1}{2}k^2 & & \binom{n+1}{3}k^3 & & \dots & & \binom{n+1}{n+1}k^{n+1}
\end{array}$$

as we will later show.

We also prove that the error decays linearly with each iteration by a factor of $|\frac{1-\sqrt{k}}{1+\sqrt{k}}|$ and that the sequence approaches $\sqrt{k}$ in an alternating fashion: overestimating and underestimating the real value at every successive iteration.

We further investigate generalizations of the form

$$x_{n+1} = ax_n + by_n \tag{3}$$
$$y_{n+1} = ca_n + dy_n \tag{4}$$

$\forall n \in \mathbb{N}, a, b, c, d \in \mathbb{N}$, where the matrix of coeafficients $T := \begin{pmatrix} a & b \\ c & d \end{pmatrix}$ is assumed to be diagonalizable with real eigenvalues. We show that one can approximate, by choosing the right coefficients, any algebraic number which is the root of a monic polynomial of degree 2 (given by the eigenvalue equation). We prove that in the general case, the error term shrinks exponentially with $n$ with contraction rate $|\frac{\lambda_2}{\lambda_1}|^n$, where $\lambda_{1,2}$ are the eigenvalues of $T$ and $|\lambda_1| > |\lambda_2|$.

Lastly, numerical experiments are presented to show the results proved in theory.

It is worth noting that even though faster methods for estimating $\sqrt{k}$ exist (such as those based on continued fractions or Netwon-Raphson methods), the present methods allow us to approximate $\sqrt{k}$ using only sums and products, making it suitable for approximations when no sophisticated software is at hand.

## 2 Derivation

It is straightforward to see that

$$x^2 = k \iff x = 1 + \frac{k-1}{1+x} \tag{5}$$

If we take a rational approximation of $x := \frac{a}{b}$, the equation below states that

$$\frac{a}{b} = \frac{a+kb}{a+b}$$

from where the system of recursions 1 arise. From this derivation it is clear that

$$lim_{n\to\infty} \frac{a_n}{b_n} = \sqrt{k}. \tag{6}$$

Consider the closely related function

$$f(x) = \frac{x+k}{x+1}$$

which is simply equation 5 (once can verify that $\sqrt{k}$ is a fixed point of $f(x)$). By looking at

$$f'(x) = \frac{1-k}{(1+x)^2}$$

we know that $f(x)$ is a monotonically decreasing function, since $k > 1$ and $f'(x) < 0$ for all $x \in \mathbb{R}$. Denote by $x_n := \frac{a_n}{b_n}$, and notice that $x_{n+1} = f(x_n)$ and that $\sqrt{k} = f(\sqrt{k})$; we then have the following behavior of the sequence $\{x_n\}_{n\in\mathbb{N}}$:

$$\begin{aligned} x_0 &< \sqrt{k} \\ x_1 &> \sqrt{k} \\ x_2 &< \sqrt{k} \\ &\vdots \\ x_{2n-1} &> \sqrt{k} \\ x_{2n} &< \sqrt{k} \\ &\vdots \end{aligned}$$

We now proceed to prove linearity of the error term for the propsed method. Let $e_n = x_n - \sqrt{k}$ for all $n \in \mathbb{N}$. By using $x_{n+1} = f(x_n)$ and expressing $x_n = e_n + \sqrt{k}$ we get

$$\frac{e_{n+1}}{e_n} = \frac{1-\sqrt{k}}{\sqrt{k}+1+e_n}$$

and by means of the convergence of the limit 6 we get the asymptotic behavior as $n \to \infty (e_n \to 0)$ to be

$$lim_{n\to\infty} \frac{e_{n+1}}{e_n} = \frac{|1-\sqrt{k}|}{\sqrt{k}+1}$$

showing the linear decaying of the error by a factor of $\frac{|1-\sqrt{k}|}{\sqrt{k}+1}$. In particular

$$lim_{n\to\infty} e_n = \left(\frac{|1-\sqrt{k}|}{\sqrt{k}+1}\right)^n e_0 \tag{7}$$

which shows that a good initial approximation can help reduce the error at each iteration.

## 3 Connection to Pascal's triangle

It is a nice exercise to prove the following closed forms of the system of recursions 1 by induction

$$a_n = \sum_{i\geq 0} \binom{n+1}{2i} k^i$$
$$b_n = \sum_{i\geq 0} \binom{n+1}{2i+1} k^i$$

$\forall n \in \mathbb{N}$ and $a_0 = 1$, $b_0 = 1$, $k \in \mathbb{Z}^+, k > 1$.

However, one can have a good intuitive feeling about it simply by calculating a few terms of the sequence $\{x_n\}_{n\in\mathbb{N}}$, and working on the inductive step (with some colors), as we now show

$$x_1 = \frac{a_1}{b_1} = \frac{1+k}{2}, \tag{8}$$
$$x_2 = \frac{a_2}{b_2} = \frac{1+3k}{3+k}, \tag{9}$$
$$x_3 = \frac{a_3}{b_3} = \frac{1+6k+k^2}{4+4k} \tag{10}$$
$$\vdots \tag{11}$$

In order to understad where this formula comes from, let's take a look at $x_3 = \frac{a_3}{b_3} = \frac{1+6k+k^2}{4+4k}$, together with the recurrence relations

$$a_{n+1} = a_n + kb_n$$
$$b_{n+1} = a_n + b_n.$$

Notice that $a_3 = 1 + 6k + k^2$ is simply the even numbered columns of the 4-th row of Pascal's triangle (from left to right, starting by 0), multiplied by powers of $k$, while $b_3 = 4 + 4k$ is the odd numbered columns of the same row, multiplied by powers of $k$.

$$\begin{array}{ccccccccccccc}
 & & & & & & 1 & & & & & & \\
 & & & & & 1 & & 1 & & & & & \\
 & & & & 1 & & 2 & & 1 & & & & \\
 & & & 1 & & 3 & & 3 & & 1 & & & \\
 & & 1k^0 & & 4k^0 & & 6k^1 & & 4k^1 & & 1k^2 & & \\
1 & & & 5 & & 10 & & 10 & & 5 & & & 1
\end{array}$$

Now, the recurrence relations 1 state that $a_{n+1} = a_n + kb_n$, meaning that we need to sum all the blue entries with $k *$ red entries (each blue entry has to be summed with the red entry to its left), giving rise to the number in the next row of Pascal's triangle.

$$
\begin{array}{ccccccccccccc}
 & & & & & & 1 & & & & & & \\
 & & & & & 1 & & 1 & & & & & \\
 & & & & 1 & & 2 & & 1 & & & & \\
 & & & 1 & & 3 & & 3 & & 1 & & & \\
 & & \xleftarrow{+} & 1k^0 & & 4k^0 & \xleftarrow{+} & 6k^1 & & 4k^1 & \xleftarrow{+} & 1k^2 & \\
 & & 1k^0 & & 5 & & 10k^1 & & 10 & & 5k^2 & & 1
\end{array}
$$

Analogously, from $b_{n+1} = a_n + b_n$ we know that we have to sum all the blue entries with red entries (each blue entry has to be summed with the red entry to its right), completing the next level of Pascal's triangle.

$$
\begin{array}{ccccccccccccc}
 & & & & & & 1 & & & & & & \\
 & & & & & 1 & & 1 & & & & & \\
 & & & & 1 & & 2 & & 1 & & & & \\
 & & & 1 & & 3 & & 3 & & 1 & & & \\
 & & 1k^0 & \xrightarrow{+} & 4k^0 & & 6k^1 & \xrightarrow{+} & 4k^1 & & 1k^2 & \xrightarrow{+} & \\
 & 1 & & 5k^0 & & 10 & & 10k^1 & & 5 & & 1k^2 &
\end{array}
$$

It is now clear that $x_4 = \frac{a_4}{b_4} = \frac{1+10k+5k^2}{5+10k+k^2}$, as expected.

$$
\begin{array}{ccccccccccccc}
 & & & & & & 1 & & & & & & \\
 & & & & & 1 & & 1 & & & & & \\
 & & & & 1 & & 2 & & 1 & & & & \\
 & & & 1 & & 3 & & 3 & & 1 & & & \\
 & & 1 & & 4 & & 6 & & 4 & & 1 & & \\
 & 1k^0 & & 5k^0 & & 10k^1 & & 10k^1 & & 5k^2 & & 1k^2 &
\end{array}
$$

Of course, the formal proof proceeds by induction (try it!), but we omit it here, since the ideas are clear and the proof does not add much clarity to the result.

As shown by the error analysis 7, a good initial condition reduced the overall error. One might think of choosing better initial values for $a_0$ and $b_0$, which will result in a modification of Pascal's triangle. It is clear that the best initial condition that one can choose is $a_0 = \lfloor \sqrt{k} \rceil, b_0 = 1$, where $\lfloor \sqrt{k} \rceil$ denotes rounding $\sqrt{k}$ to the nearest integer. One might think that this is a circular argument: we are trying to calculate $\sqrt{k}$ with an initial condition that takes $\sqrt{k}$ and rounds it to the nearest integer. However, this is not the case, since $a_0$ simply means the best guess on $\sqrt{k}$ (for example, for $k = 51$, one can set $a_0 = 7$ without knowing the exact value of $\sqrt{51}$).

With that being said, the modified Pascal's triangle on which the algorithm would work would be the following

$$
\begin{array}{ccccccccc}
 & & & & 1 & & & & \\
 & & & \lfloor\sqrt{k}\rfloor & & 1 & & & \\
 & & \lfloor\sqrt{k}\rfloor & & \lfloor\sqrt{k}\rfloor+1 & & 1 & & \\
 & \lfloor\sqrt{k}\rfloor & & 2\lfloor\sqrt{k}\rfloor+1 & & \lfloor\sqrt{k}\rfloor+2 & & 1 & \\
\lfloor\sqrt{k}\rfloor & & 3\lfloor\sqrt{k}\rfloor+1 & & 3\lfloor\sqrt{k}\rfloor+3 & & \lfloor\sqrt{k}\rfloor+3 & & 1 \\
 & & & & \vdots & & & &
\end{array}
$$

# 4 Generlizations

Consider the following system of recursions

$$x_{n+1} = ax_n + by_n \tag{12}$$
$$y_{n+1} = ca_n + dy_n \tag{13}$$

$\forall n \in \mathbb{N}, a, b, c, d \in \mathbb{N}$, where the matrix of coeafficients $T := \begin{pmatrix} a & b \\ c & d \end{pmatrix}$ is assumed to be diagonalizable with real eigenvalues. Clearly,

$$\begin{pmatrix} x_n \\ y_n \end{pmatrix} = T^n \begin{pmatrix} x_0 \\ y_0 \end{pmatrix} = PD^nP^{-1} \begin{pmatrix} x_0 \\ y_0 \end{pmatrix}$$

where $P$ is the matrix whose columns are the eigenvectors of $T$. With elementary algebra one can derive the general form of the eigenvectors, namely

$$\lambda_{1,2} = \frac{\mathrm{tr}(T) \pm \sqrt{\mathrm{tr}(T)^2 - 4\ \det(T)}}{2},$$

where

$$\mathrm{tr}(T) = a + d, \quad \det(T) = ad - bc$$

with eigenvectors

$$v_1 = \begin{pmatrix} 1 \\ \frac{\lambda_1 - a}{b} \end{pmatrix}, \qquad v_2 = \begin{pmatrix} 1 \\ \frac{\lambda_2 - a}{b} \end{pmatrix}.$$

In this setting,

$$\begin{aligned} T^n = PD^nP^{-1} &= \frac{1}{\lambda_2 - \lambda_1} \begin{pmatrix} 1 & 1 \\ \frac{\lambda_1 - a}{b} & \frac{\lambda_2 - a}{b} \end{pmatrix} \begin{pmatrix} \lambda_1^n & 0 \\ 0 & \lambda_2^n \end{pmatrix} \begin{pmatrix} \lambda_2 - a & -b \\ -(\lambda_1 - a) & b \end{pmatrix} \\ &= \frac{1}{y_2 - y_1} \begin{pmatrix} \lambda_1^n y_2 - \lambda_2^n y_1 & \lambda_2^n - \lambda_1^n \\ \lambda_1^n y_1 y_2 - \lambda_2^n y_1 y_2 & \lambda_2^n y_2 - \lambda_1^n y_1 \end{pmatrix} \end{aligned}$$

where $y_i = \frac{\lambda_i - a}{b}$ for $i = 1, 2$.

If $|\lambda_2| < |\lambda_1|$ then

$$lim_{n\to\infty} T^n = \frac{1}{y_2 - y_1} \begin{pmatrix} \lambda_1^n y_2 & -\lambda_1^n \\ \lambda_1^n y_1 y_2 & -\lambda_1^n y_1 \end{pmatrix}$$

in which case

$$lim_{n\to\infty} \frac{x_n}{y_n} = lim_{n\to\infty} T^n \begin{pmatrix} x_0 \\ y_0 \end{pmatrix} = \frac{1}{y_1} = \frac{b}{\lambda_1 - a}.$$

It is then clear that by choosing appropriate coefficients for the matrix $T$, one can make the limit converge to $\frac{b}{\lambda_1 - a}$ where $\lambda_1$ is the root of the characteristic polynomial of $T$ (a desgree 2 monic polynomial).

One can verify that system 1 corresponds to $T = \begin{pmatrix} 1 & k \\ 1 & 1 \end{pmatrix}$ in which case $\lambda_1 = 1 + \sqrt{k}, \quad \lambda_2 = 1 - \sqrt{k}$ and thus

$$lim_{n\to\infty} \frac{x_n}{y_n} = \frac{k}{1 + \sqrt{k} - 1} = \sqrt{k}.$$

Another interesting example comes from considering $T = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}$ in which case $\lambda_1 = \frac{1+\sqrt{5}}{2}, \quad \lambda_2 = \frac{1-\sqrt{5}}{2}$ and thus

$$lim_{n\to\infty} \frac{x_n}{y_n} = \frac{1}{\frac{1+\sqrt{5}}{2} - 1} = \frac{1 + \sqrt{5}}{2} = \varphi$$

the famous golden ratio.

One can approximate rational square roots by considering $T = \begin{pmatrix} 1 & b \\ c & 0 \end{pmatrix}$

in which case $\lambda_1 = 1 + \sqrt{bc}, \quad \lambda_2 = 1 - \sqrt{bc}$ and thus

$$lim_{n\to\infty} \frac{x_n}{y_n} = \sqrt{\frac{b}{c}}.$$

## 4.1 The error term

How good is our algorithm? Our linear algebra framework provides us with an elegant analysis of the error term in the general case.

By 12 we can write, for $v_n := (x_n, y_n)$

$$v_n = T \begin{pmatrix} x_0 \\ y_0 \end{pmatrix} = PD^n P^{-1} \begin{pmatrix} x_0 \\ y_0 \end{pmatrix} = PD^n \begin{pmatrix} \gamma_x \\ \gamma_y \end{pmatrix} = P \begin{pmatrix} \lambda_1^n \gamma_x \\ \lambda_2^n \gamma_y \end{pmatrix} = \lambda_1^n \gamma_x \boldsymbol{\beta_1} + \lambda_2^n \gamma_y \boldsymbol{\beta_2}$$

where $\boldsymbol{\beta_1}, \boldsymbol{\beta_2}$ is an eigenbasis of $T$.

Since $|\lambda_1| > |\lambda_2|$ we know that

$$L := lim_{n\to\infty} v_n = \lambda_1^n \gamma_x \boldsymbol{\beta_1},$$

so the normalized error term is

$$e_n := \frac{|L - v_n|}{\lambda_1^n} = \big(\big|\frac{\lambda_2}{\lambda_1}\big|\big)^n \gamma_y \boldsymbol{\beta_2}$$

in which case the error term shrinks exponentially with $n$ (linearly when compared to $e_{n-1}$) with contraction rate $\big|\frac{\lambda_2}{\lambda_1}\big|^n$.

It is worth noticing that this in accordance with out previous result, where we had calculated the error term to be

$$\left|\frac{1-\sqrt{k}}{1+\sqrt{k}}\right|^n$$

which are precisely the eigenvalues of the matrix $T = \begin{pmatrix} 1 & k \\ 1 & 1 \end{pmatrix}$.

We are aware that this type of analysis is able to extend our generalizations even further, to a system of recursions of the form

$$\mathbf{x}_{n+1} = A\mathbf{x}_n, \quad A = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{pmatrix}, \quad \mathbf{x}_n = \begin{pmatrix} x_n^{(1)} \\ x_n^{(2)} \\ \vdots \\ x_n^{(n)} \end{pmatrix}$$

where $|\lambda_1| > |\lambda_2| > \ldots |\lambda_n|$. The normalized error term

$$e_n = \sum_{j=1}^{n} \left|\frac{\lambda_j}{\lambda_1}\right|^n \gamma_{x_j} \boldsymbol{\beta_j}$$

which is dominated by $max_j|\frac{\lambda_j}{\lambda_1}|$. This cases, however, are out of the scope of the current text.

# 5 Numerical examples

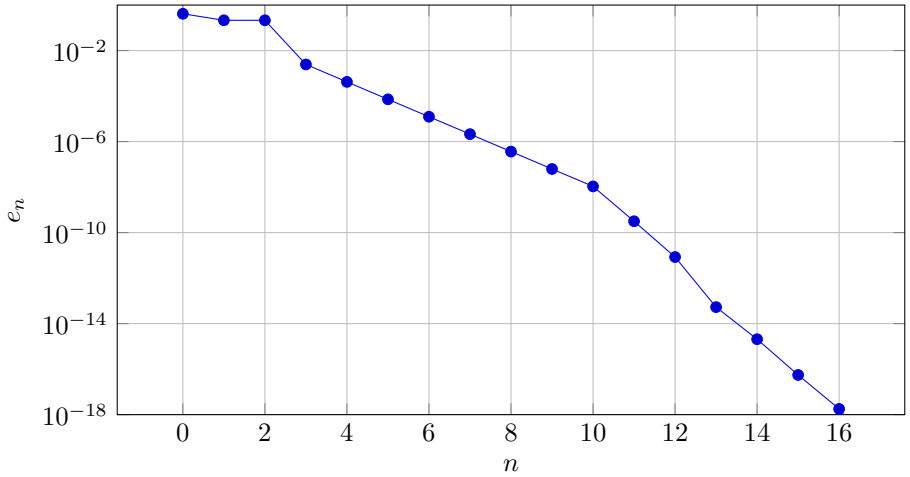
Going back to our original system of recursions given by

$$a_{n+1} = a_n + kb_n$$
$$b_{n+1} = a_n + b_n$$

for all $n \in \mathbb{N}$, we have calculated the error have a contraction factor of $\left|\frac{1-\sqrt{k}}{1+\sqrt{k}}\right|^n$, where $n$ is the number of iterations. We proceed to show this numerically for the case k = 2.

| $n$ | $a_n$ | $b_n$ | $\frac{a_n}{b_n}$ | Error $\lvert\sqrt{2} - \frac{a_n}{b_n}\rvert$ |
|---|---|---|---|---|
| 0 | 1 | 1 | 1 | 0.414213562373095048 |
| 1 | 3 | 2 | 1.5 | 0.214213562373095048 |
| 2 | 7 | 5 | 1.4 | 0.314213562373095048 |
| 3 | 17 | 12 | 1.416666 | 0.002453104293571 |
| 4 | 41 | 29 | 1.41379 | 0.00042045892481 |
| 5 | 99 | 70 | 1.414285714285 | 0.0000721519126 |
| 6 | 239 | 169 | 1.4142011834 | 0.000012378941 |
| 7 | 577 | 408 | 1.41421568627 | 0.0000021239014 |
| 8 | 1393 | 985 | 1.41421319796 | 0.0000003644035 |
| 9 | 3363 | 2378 | 1.41421362489 | 0.000000062520 |
| 10 | 8119 | 5741 | 1.41421355164 | 0.00000001072 |
| 11 | 19601 | 13860 | 1.4142135620573208 | 0.00000000031 |
| 12 | 47321 | 33461 | 1.41421356238153 | 0.000000000008 |

Error term $e_n = \left|\sqrt{2} - \frac{a_n}{b_n}\right|$

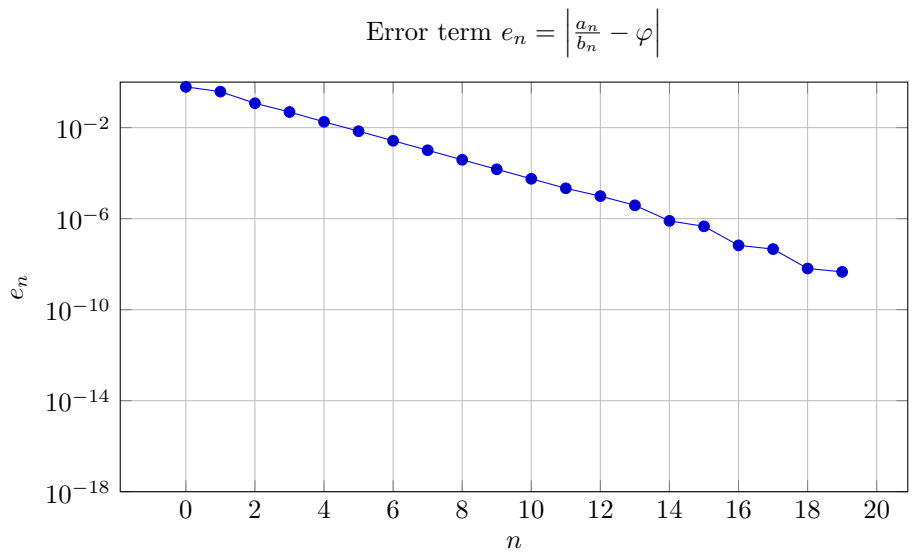



We do the same for the system

$$x_{n+1} = ax_n + by_n$$
$$y_{n+1} = ca_n + dy_n$$

with $T = \begin{pmatrix} a & b \\ c & d \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}$

| $n$ | $a_n$ | $b_n$ | $\frac{a_n}{b_n}$ | Error $\left\lvert \frac{a_n}{b_n} - \varphi \right\rvert$ |
|---|---|---|---|---|
| 0 | 1 | 1 | 1 | 0.61803398874989 |
| 1 | 2 | 1 | 2 | 0.38196601125010 |
| 2 | 3 | 2 | 1.5 | 0.118033988749894 |
| 3 | 5 | 3 | 1.666 | 0.04863267791 |
| 4 | 8 | 5 | 1.6 | 0.0180339887498949 |
| 5 | 13 | 8 | 1.625 | 0.006966011250105 |
| 6 | 21 | 13 | 1.6153846153846153846 | 0.00264937336527951791 |
| 7 | 34 | 21 | 1.619047619047619047619 | 0.001013630297724144906 |
| 8 | 55 | 34 | 1.61764705882352941 | 0.000386929926365490760 |
| 9 | 89 | 55 | 1.618181818181818181 | 0.00014782943192328 |
| 10 | 144 | 89 | 1.61797752808988 | 0.000056460659992253132 |
| 11 | 233 | 144 | 1.61805555555555555 | 0.0000215668056616606 |
| 12 | 377 | 233 | 1.6180257510729613 | 0.00000976312293353 |
| 13 | 610 | 377 | 1.6180371352785145 | 0.00000385391638047968 |
| 14 | 987 | 610 | 1.618032786885245 | 0.000000799864648007000 |
| 15 | 1597 | 987 | 1.618034447821681229 | 0.000000458983786792 |
| 16 | 2584 | 1597 | 1.61803405572755417 | 0.0000000671103600883407 |
| 17 | 4181 | 2584 | 1.61803444843869 | 0.0000004596527808 |
| 18 | 6765 | 4181 | 1.6180340528867513 | 0.0000000645006082 |
| 19 | 10946 | 6765 | 1.61803444769678 | 0.000000458257 |




The golden ratio $\varphi$ is one of the famously difficult irrational numbers to approximate with fractions. Actually, it is the most difficult of all.

# 6 Further comments

The reader might be puzzled as to why one might use this algorithm instead of the faster existing versions. The first answer is that this algorithm has the big advantage that only sums and multiplications are needed to calculate the next term, as opposed to most fast numerical methods, making it convenient when no sophisticated software is at hand, or when dividing is expensive. The second answer is that it is easily analyzable and generalizable, making it predictable and allowing multiple algebraic numbers to be approximated with it.