Chapter 3
Experimental accuracy analysis

Because exact solution formulas do not exist for many problems, our numerical methods are typically indirect. One common approach is to express a desired quantity as a limit of a sequence: $x = \lim_{n\to\infty} x_n$. As the sequence is traversed, the terms get closer and closer to the desired answer. Rather than taking the limit $n \to \infty$, many of our numerical methods stop at some large n. The resulting error is called *truncation error*.

Commonly available computers store real numbers to about 15 or 16 digits of precision. For this reason, a sequence of approximations may settle on a fixed value rather than becoming forever more accurate. When this happens, we say that *numerical convergence* has occurred. This often indicates that high accuracy has been achieved.

This chapter presents a number of examples in which numerical convergence is sought and rate of convergence is measured empirically. These experimental techniques can justify high confidence in the accuracy of computations, and in later chapters, they are complemented by rigorous proofs.

3.1 ▪ Numerical convergence

When generating a sequence of approximations in the limited precision of a computer, numerical convergence occurs if the terms cease to change significantly. If this happens, then the computation can be concluded.

Example 3.1. While studying compound interest in 1683, Jakob Bernoulli investigated $(1 + 1/n)^n$, showing that the $n \to \infty$ limit is between 2 and 3. Nowadays, we know that the limit equals e, the base of the natural logarithm:

$$e = \lim_{n\to\infty} \left(1 + \tfrac{1}{n}\right)^n . \tag{3.1}$$

Approximate e by computing $(1 + 1/n)^n$ for increasing values of n. Does numerical convergence occur? When? To what value?

Solution. By trial and error, we have found that extremely large values of n are required to understand the behavior of the sequence. Below, the nth term is computed for $n = 1$, 100, 10^4, 10^6, . . . , 10^{18}.

```
>> n = 10.^(0:2:18);
>> eapprox = nan(size(n));
```

```
>> for k = 1:length(n)
     eapprox(k) = (1+1/n(k))^n(k);
   end
```

The results are displayed in a table using a command `disptable` provided with this book. The columns of the table are specified one at a time, and each column specification consists of a column heading, a data vector, and an `sprintf` format string (as seen in Example 2.3).

```
>> disptable('n',n,'%5.0e','(1+1/n)^n',eapprox,'%17.15f');
    n            (1+1/n)^n
1e+00   2.000000000000000
1e+02   2.704813829421528
1e+04   2.718145926824926
1e+06   2.718280469095753
1e+08   2.718281798347358
1e+10   2.718282053234788
1e+12   2.718523496037238
1e+14   2.716110034087023
1e+16   1.000000000000000
1e+18   1.000000000000000
```

Technically, numerical convergence occurs around $n = 10^{16}$, but the final value according to the computer is 1. The computation is a failure. At first, the terms appear to be approaching a number around 2.718282, presumably a decent approximation for e, but then they veer away. An effective method would converge numerically to an accurate approximation. ∎

The sequence $(1 + 1/n)^n$ in the previous example converges very slowly. Before a highly accurate approximation for e is found, *roundoff error* overwhelms the computation. In the extreme case, $1/n$ becomes so small that $1 + 1/n$ is rounded to 1 by the computer, so that the formula $(1 + 1/n)^n$ is effectively replaced by $1^n = 1$. A progression of less and less accurate computations precedes this total collapse. Roundoff error is explored in more detail in Chapter 5.

The next example investigates a different method for computing e.

Example 3.2. About a half-century after Bernoulli, Leonhard Euler stated the following equation:

$$e = \sum_{i=0}^{\infty} \frac{1}{i!}. \tag{3.2}$$

Approximate e by truncating the series. Does numerical convergence occur? When? To what value?

Solution. Euler's series is also a limit formula because it can be rewritten as

$$e = \lim_{n\to\infty} \sum_{i=0}^{n} \frac{1}{i!}.$$

Experimentally, the sequence of partial sums

$$\sum_{i=0}^{n} \frac{1}{i!}, \quad n = 0, 1, 2, 3, \ldots,$$

is found to converge rapidly.

```
>> n = 0:2:20;
>> eapprox = nan(size(n));
>> for k = 1:length(n)
     eapprox(k) = sum(arrayfun(@(i) 1/factorial(i),0:n(k)));
   end
>> disptable('n',n,'%2d','partial sum',eapprox,'%17.15f');
 n          partial sum
 0   1.000000000000000
 2   2.500000000000000
 4   2.708333333333333
 6   2.718055555555555
 8   2.718278769841270
10   2.718281801146385
12   2.718281828286169
14   2.718281828458230
16   2.718281828459043
18   2.718281828459046
20   2.718281828459046
```

(The format string `'%2d'` above specifies an integer with a width of two digits.) Numerical convergence occurs by $n = 18$. We suspect that the final approximation

$$e \approx 2.718281828459046 \tag{3.3}$$

is highly accurate. ∎

Examples 3.1 and 3.2 teach an important lesson: A slow-converging method may be costly or unstable, while a fast-converging method may deliver an accurate solution in short time.

3.2 ▪ Rate of convergence

Associated to a sequence of approximations $x_1, x_2, x_3, \ldots$ converging to a limit x is the sequence of absolute errors $|x - x_n|$, $n = 1, 2, 3, \ldots$. Often, such a sequence resembles a power function n^{-p} or an exponential function r^{-n}. Figures 3.1 and 3.2 display examples of these and other familiar functions. We will compare our experimental data with these plots.

The plots on the left of Figures 3.1 and 3.2 use familiar linear scales, but the plots on the right may be less familiar. The plot on the right of Figure 3.1 is called a *log-log* plot because it uses logarithmic scales for the vertical and horizontal axes. Note that the value on each axis grows by a constant *factor* with each tick mark. (The notations $\cdot^{-15}$, $\cdot^{-10}$, and $\cdot^{-5}$ are short for 10^{-15}, 10^{-10}, and 10^{-5}, respectively.) The plot on the right of Figure 3.2 is called a *log-linear* plot because only the vertical axis is measured on a logarithmic scale.

A power function n^{-p} appears as a straight line on a log-log plot because $y = \log_{10} n^{-p}$ is related to $x = \log_{10} n$ by $y = -px$. An exponential function r^{-n} appears as a straight line on a log-linear plot because $y = \log_{10} r^{-n}$ is related to $x = n$ by $y = -(\log_{10} r)x$. Exponential functions r^{-n} converge to zero much more quickly than power functions n^{-p}.

In MATLAB code, `xlog` or `ylog` specifies a logarithmic scale for the horizontal or vertical axis, respectively.

Example 3.3. Our best approximation for e is (3.3). Using this value, measure the absolute errors $|e - (1 + 1/n)^n|$ associated with Bernoulli's limit formula. Plot the

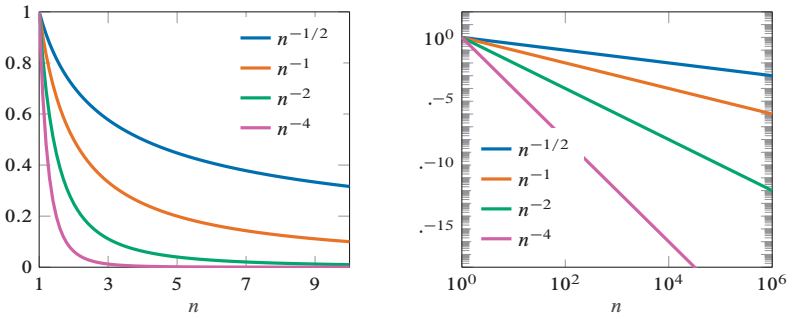

Figure 3.1. *Power functions on standard (left) and log-log (right) axes. In the plot on the right, the marks* $\cdot^{-5}$, $\cdot^{-10}$, *and* $\cdot^{-15}$ *are abbreviations for* 10^{-5}, 10^{-10}, *and* 10^{-15}.

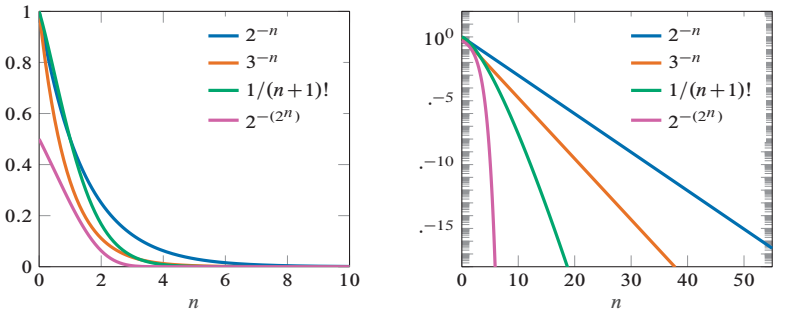

Figure 3.2. *Exponential functions (and more) on standard (left) and log-linear (right) axes.*

errors, and, if possible, find a power function or exponential function that converges to zero at the same rate.

Solution. The approximations are computed as in Example 3.1.

```
>> n = 10.^(0:2:18);
>> eapprox = nan(size(n));
>> for k = 1:length(n)
     eapprox(k) = (1+1/n(k))^n(k);
   end
```

We plan to make two graphs: one of the approximations themselves, and one of their absolute errors. An m-by-n grid of subfigures is created with `ax = newfig(m,n)`.

```
>> ax = newfig(1,2);
```

The approximations $(1 + 1/n)^n$ are plotted in the left graph of Figure 3.3.

```
>> xlog;
>> plot(n,eapprox,'*','displayname','(1+1/n)^n');
>> ylim([0 3.5]);
>> xlabel('n');
```

At first, the computation makes progress toward a value around 2.7, but things go haywire sometime before $n = 10^{16}$.

The absolute errors are estimated by comparing with (3.3), the best available approximation.

```
>> err = abs(2.718281828459046-eapprox);
>> disptable('n'          ,n       ,'%5.0e'    ...
             ,'(1+1/n)^n' ,eapprox,'%17.15f' ...
```

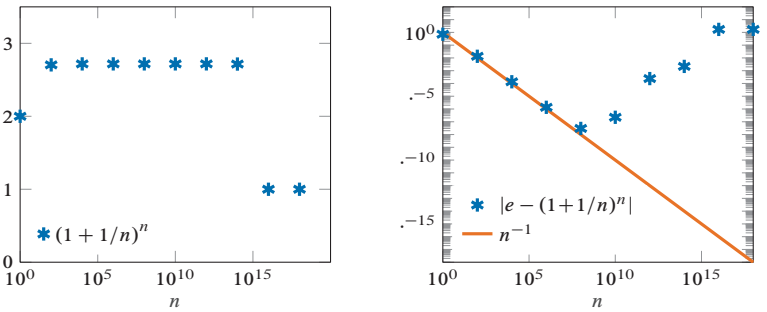

Figure 3.3. *Approximations for e from Bernoulli's limit (left) and their absolute errors (right).*

```
           ,'abs. error',err     ,'%17.15f');
   n           (1+1/n)^n          abs. error
1e+00   2.000000000000000   0.718281828459046
1e+02   2.704813829421528   0.013467999037517
1e+04   2.718145926824926   0.000135901634120
1e+06   2.718280469095753   0.000001359363293
1e+08   2.718281798347358   0.000000030111688
1e+10   2.718282053234788   0.000000224775742
1e+12   2.718523496037238   0.000241667578192
1e+14   2.716110034087023   0.002171794372023
1e+16   1.000000000000000   1.718281828459046
1e+18   1.000000000000000   1.718281828459046
```

The most accurate computation occurs around $n = 10^8$, with an absolute error of about 3×10^{-8}. The absolute errors are also plotted in the right graph of Figure 3.3. The invocation of `subplot` below selects the second subfigure before plotting.

```
>> subplot(ax(2));
>> xlog; ylog; ylim([1e-18 1e2]);
>> plot(n,err,'*','displayname','|e-(1+1/n)^n|');
```

The absolute errors are intended to converge to zero. At first, they appear to decrease at a predictable rate, but eventually they veer upward. By trial and error, the graph of n^{-1} is found to be a good fit for the initial trend. This function appears as a straight line on the log-log plot.

```
>> plotfun(@(n) n^(-1),[n(1) n(end)],'displayname','n^{-1}');
>> xlabel('n');
```

Mathematically, $|e - (1 + 1/n)^n|$ appears to converge to 0 at the same rate as n^{-1}, but evidently the computer is unable to calculate $(1 + 1/n)^n$ accurately for $n > 10^8$. ∎

In the previous example, the truncation error is comparable to a power function n^{-p}. Better numerical methods converge at least as fast as an exponential function r^{-n}. To emphasize such distinctions, some language is introduced.

Given sequences $a_1, a_2, a_3, \ldots$ and $b_1, b_2, b_3, \ldots$ of nonnegative real numbers that converge to zero, we say that a_n *converges as fast as* b_n if there exist a constant M and a positive integer N such that $|a_n| \le M b_n$ for all $n \ge N$. We say that a_n and b_n *converge at the same rate* if a_n converges as fast as b_n and vice versa. Consider a sequence $x_1, x_2, x_3, \ldots$ that converges to x.

- The convergence is *algebraic* if $|x - x_n|$ converges to zero as fast as n^{-p} for some $p > 0$. On a log-log plot, the absolute errors eventually lie on or below a line with slope $-p$.

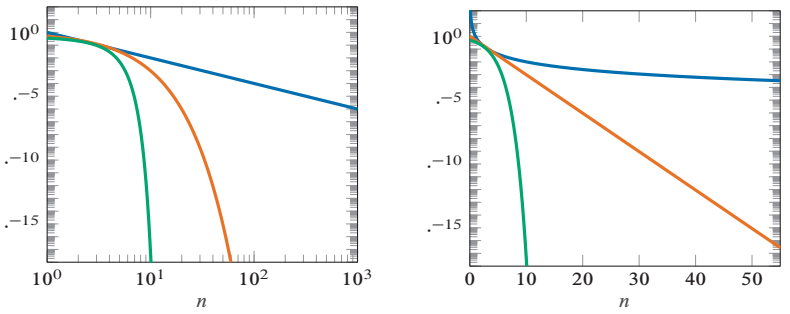

Figure 3.4. *Rates of convergence: algebraic (blue), geometric (orange), and supergeometric (green), on log-log axes (left) and log-linear axes (right).*

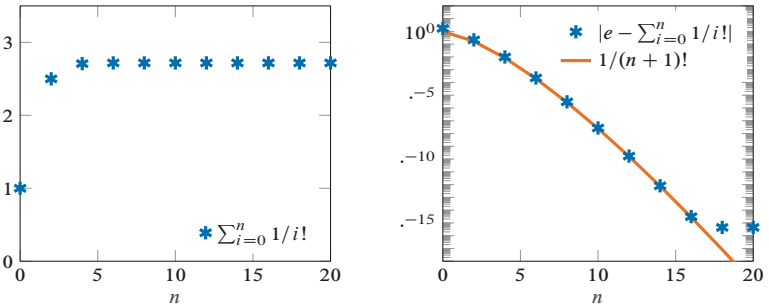

Figure 3.5. *Approximations for e from Euler's series (left) and their absolute errors (right).*

- The convergence is *geometric* if $|x - x_n|$ converges to zero as fast as r^{-n} for some $r > 1$. On a log-linear plot, the absolute errors eventually lie on or below a line with slope $-\log_{10} r$.

- The convergence is *supergeometric* if $|x - x_n|$ converges to zero as fast as r^{-n} for *any* $r > 1$. On a log-linear plot, the absolute errors fall toward zero faster than any straight line.

These classes are illustrated in Figure 3.4.

Bernoulli's formula for e converges algebraically because the associated truncation errors converge to zero, discounting roundoff error, as fast as the power function n^{-1}. Graphically, the truncation errors converge to zero as fast as a straight line in a log-log plot.

Example 3.4. How fast does Euler's series $\sum_{i=0}^{n} 1/i!$ converge? Classify the rate as algebraic, geometric, or supergeometric.

Solution. Several partial sums are computed as in Example 3.2.

```
>> n = 0:2:20;
>> eapprox = nan(size(n));
>> for k = 1:length(n)
     eapprox(k) = sum(arrayfun(@(i) 1/factorial(i),0:n(k)));
   end
```

The approximations are plotted in Figure 3.5.

```
>> ax = newfig(1,2);
>> plot(n,eapprox,'*','displayname','\Sigma_{i=0}^n 1/i!');
>> ylim([0 3.5]);
>> xlabel('n');
```

In place of the true absolute error $|e - \sum_{i=0}^{n} 1/i!|$, we consider the difference

$$\left| 2.718281828459046 - \sum_{i=0}^{n} \frac{1}{i!} \right|.$$

```
>> err = abs(2.718281828459046-eapprox);
>> disptable('n'          ,n      ,'%2d'      ...
            ,'partial sum',eapprox,'%17.15f' ...
            ,'abs. error' ,err    ,'%17.15f');
 n         partial sum          abs. error
 0   1.000000000000000   1.718281828459046
 2   2.500000000000000   0.218281828459046
 4   2.708333333333333   0.009948495125713
 6   2.718055555555555   0.000226272903491
 8   2.718278769841270   0.000003058617776
10   2.718281801146385   0.000000027312661
12   2.718281828286169   0.000000000172877
14   2.718281828458230   0.000000000000816
16   2.718281828459043   0.000000000000003
18   2.718281828459046   0.000000000000000
20   2.718281828459046   0.000000000000000
```

The absolute errors are added to the figure.

```
>> subplot(ax(2));
>> ylog; ylim([1e-18 1e2]);
>> plot(n,err,'*','displayname','|e-\Sigma_{i=0}^n 1/i!|');
```

Note that the error plot is on log-linear axes, in contrast to the log-log axes of Example 3.3. Because the errors fall toward zero even faster than a straight line on a log-linear plot (until numerical convergence occurs at $n = 18$), the convergence is supergeometric. In fact, the absolute error is close to $1/(n+1)!$, as seen in the plot.

```
>> plot(n,1./factorial(n+1),'displayname','1/(n+1)!');
>> xlabel('n');
```

■

Bernoulli's sequence $(1 + 1/n)^n$ converges algebraically, whereas Euler's series $\sum_{i=0}^{n} 1/i!$ converges supergeometrically. Euler's series begets a superior numerical method.

So far, we have seen examples of algebraic and supergeometric convergence. The next example illustrates a geometric rate of convergence.

Example 3.5. The quantity $\sqrt{2}$ is the limit of the sequence $x_0, x_1, x_2, \ldots$ defined by $x_0 = 1$ and

$$x_{n+1} = \begin{cases} x_n + 2^{-n-1} & \text{if } (x_n + 2^{-n-1})^2 \le 2, \\ x_n & \text{otherwise.} \end{cases}$$

The recursion starts with the low estimate of 1 and adds powers of two as long as the

sum does not exceed the desired quantity. Here are the first few terms:

$$\begin{aligned}
x_0 &= 1 &&= 1,\\
x_1 = x_0 &= 1 &&= 1,\\
x_2 = x_1 + 2^{-2} = 1 + 1/4 &= 5/4 &&= 1.25,\\
x_3 = x_2 + 2^{-3} = 5/4 + 1/8 &= 11/8 &&= 1.375,\\
x_4 = x_3 &= 11/8 &&= 1.375,\\
x_5 = x_4 + 2^{-5} = 11/8 + 1/32 &= 45/32 &&= 1.40625.
\end{aligned}$$

Classify the rate of convergence as algebraic, geometric, or supergeometric.

Solution. The sequence of approximations $x_0, x_1, x_2, \ldots, x_{60}$ is computed.

```
>> n = 0:60;
>> x = nan(size(n));
>> x(1) = 1;
>> for k = 1:length(n)-1
     if (x(k)+2^(-n(k)-1))^2<=2
       x(k+1) = x(k)+2^(-n(k)-1);
     else
       x(k+1) = x(k);
     end
   end
```

The sequence of approximations is plotted in the left graph of Figure 3.6.

```
>> ax = newfig(1,2);
>> plot(n,x,'*','displayname','x_n');
>> ylim([0 2]);
>> xlabel('n');
```

The final approximation for $\sqrt{2}$ is the following:

```
>> x(end)
ans =
   1.414213562373095
```

The absolute errors $|\sqrt{2} - x_n|$ are measured using the built-in function `sqrt`. The errors are shown in the right graph of Figure 3.6.

```
>> subplot(ax(2));
>> ylog; ylim([1e-18 1e2]);
>> plot(n,abs(sqrt(2)-x),'*','displayname','|sqrt(2)-x_n|');
```

The errors appear to lie below a straight line on a log-linear plot. This is evidence of geometric convergence. More specifically, the errors appear to converge at the same rate as 2^{-n}.

```
>> plotfun(@(n) 2^(-n),[n(1) n(end)],'displayname','2^{-n}');
>> xlabel('n');
```

■

3.3 ▪ Residuals

A major difficulty in measuring error is a chicken-and-egg problem: to measure the error $|x - \hat{x}|$ directly, the exact value x is required, but the whole reason for computing the approximation $\hat{x}$ is that x is unknown.

When a computed quantity $\hat{x}$ is an approximate solution to an equation $f(x) = b$, the corresponding *residual* is $b - f(\hat{x})$, and the *absolute residual* is $|b - f(\hat{x})|$. A small residual indicates that the computed value nearly solves the equation. Unlike absolute or

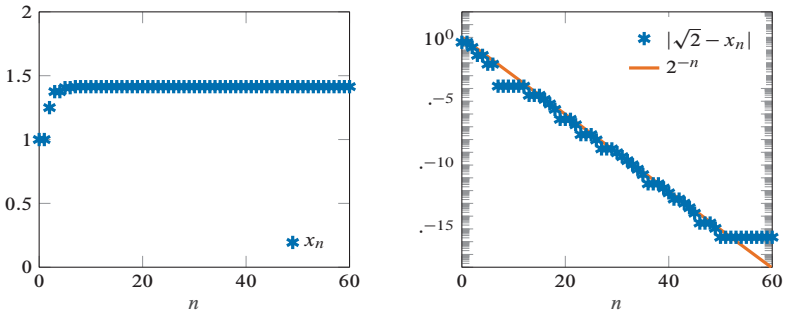

Figure 3.6. *Approximations for $\sqrt{2}$ (left) and their absolute errors (right).*

relative error, a residual can be measured without knowing the true value of the quantity of interest.

Example 3.6. Approximate $\sqrt{2}$ by x_n as in Example 3.5. Because $\sqrt{2}$ is a solution of $x^2 = 2$, an appropriate residual for evaluating x_n is $2 - x_n^2$. Plot the absolute residuals $|2 - x_n^2|$ and measure the rate of convergence.

Solution. The approximations are computed again.

```
>> n = 0:60;
>> x = nan(size(n));
>> x(1) = 1;
>> for k = 1:length(n)-1
     if (x(k)+2^(-n(k)-1))^2<=2
       x(k+1) = x(k)+2^(-n(k)-1);
     else
       x(k+1) = x(k);
     end
   end
```

The absolute residuals $|2 - x_n^2|$ are measured and displayed in Figure 3.7.

```
>> ax = newfig;
>> ylog; ylim([1e-18 1e2]);
>> plot(n,abs(2-x.^2),'*','displayname','|2-x_n^2|');
```

The residuals appear to converge at the rate 2^{-n}, the same rate seen for the absolute errors in Example 3.5.

```
>> plotfun(@(n) 2^(-n),[n(1) n(end)],'displayname','2^{-n}');
>> xlabel('n');
```

■

In Example 3.6, the residuals are measured without knowledge of the true value of $\sqrt{2}$, and they suggest the same geometric rate of convergence as the absolute errors in Example 3.5. Clearly, residuals are useful. However, the connection between error and residual is subtle, and for some problems, a small residual may not indicate a small error. The full story must wait for now.

Notes

The notion that a_n converges to zero as fast as b_n can be expressed using the popular "big-O" notation as $a_n = O(b_n)$, $n \to \infty$.

Havil's book *The Irrationals* provides additional historical background on the number e [41].

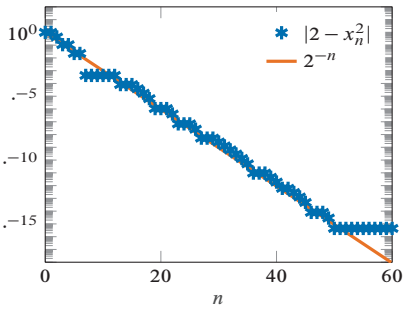

Figure 3.7. *Absolute residuals for the approximations of* $\sqrt{2}$ *in Figure* 3.6.

Exercises

Exercises 1–9: A mathematical constant is expressed as a limit of a sequence. Approximate the constant by truncating the sequence and seek numerical convergence. Either report the value of n at which the sequence converges numerically, or, if numerical convergence is not achieved with reasonable resources, find approximately the value of n at which the absolute error falls below 10^{-4}. Either way, provide evidence. You may compare with the known values $\pi = 3.141592653589793\ldots$, $\gamma = 0.577215664901533\ldots$, and $e = 2.718281828459046\ldots$.

1. $\pi = \lim_{n\to\infty} 2\prod_{k=1}^{n} \frac{(2k)^2}{(2k-1)(2k+1)}$ (Wallis's product)
2. $\pi = \lim_{n\to\infty} (2 + \frac{1}{2n}) \prod_{k=1}^{n} \frac{(2k)^2}{(2k-1)(2k+1)}$
3. $\pi = \lim_{n\to\infty} 4\sum_{i=0}^{n} \frac{(-1)^i}{2i+1}$ (Gregory series)
4. $\pi = \lim_{n\to\infty} \frac{(-1)^{n+1}}{n} + 4\sum_{i=0}^{n} \frac{(-1)^i}{2i+1}$
5. $\pi = \lim_{n\to\infty} 2\sqrt{3}\sum_{i=0}^{n} \frac{(-1)^i}{(2i+1)3^i}$ (Sharp series)
6. $\pi = \lim_{n\to\infty} \frac{16}{5}\sum_{i=0}^{n} \frac{(-1)^i}{(2i+1)5^{2i}} - \frac{4}{239}\sum_{i=0}^{n} \frac{(-1)^i}{(2i+1)239^{2i}}$ (Machin's formula)
7. $\pi = \lim_{n\to\infty} \left(6\sum_{k=1}^{n} \frac{1}{k^2}\right)^{1/2}$ (the Basel problem)
8. $\gamma = \lim_{n\to\infty} \sum_{k=1}^{n} \frac{1}{k} - \log n$ (Euler–Mascheroni constant)
9. $e = \lim_{n\to\infty} n\prod_{k=1}^{n} k^{-1/n}$ (from Stirling's approximation)

Exercises 10–17: Refer to the specified exercise to find a mathematical constant expressed as a limit of a sequence. Classify the rate of convergence as algebraic, geometric, or supergeometric. If algebraic or geometric, find a function of the form n^{-p} or r^{-n} that converges at approximately the same rate as the absolute error.

10. Exercise 1
11. Exercise 2
12. Exercise 3
13. Exercise 4
14. Exercise 5
15. Exercise 6

16. Exercise 7
17. Exercise 8

18. Modify Example 3.5 to compute $\sqrt{3}$. Run the method until it converges numerically. Report the approximations $x_0, x_1, x_2, \ldots$ and the corresponding sequence of absolute residuals $|3 - x_n^2|$, $n = 0, 1, 2, \ldots$.

19. Modify Example 3.5 to compute $\sqrt[3]{2}$. Run the method until it converges numerically. Report the approximations $x_0, x_1, x_2, \ldots$ and the corresponding absolute residuals $|2 - x_n^3|$, $n = 0, 1, 2, \ldots$.

20. Compute $\sqrt{2}$ using the Babylonian method of Example 1.1, starting from the guess $x_0 = 1.5$. Graph the absolute residuals $|2 - x_n^2|$ and classify the rate of convergence as algebraic, geometric, or supergeometric.

21. Suppose the numerical value of π is unknown and an approximation $\hat{x}$ is computed. Two possible residuals are $\cos \hat{x} - (-1)$ and $\sin \hat{x} - 0$. As $\hat{x}$ approaches π, one of these residuals converges to 0 considerably faster than the other. Which one, and why? Which do you think provides a better measure of the approximation's accuracy? Explain.

Exercises 22–24: The continued fraction $[a_0; a_1, a_2, a_3, \ldots, a_n]$ and an approach to its computation are introduced in Exercise 20 in Chapter 2. The infinite continued fraction $[a_0; a_1, a_2, a_3, \ldots]$ is defined by

$$[a_0; a_1, a_2, a_3, \ldots] = \lim_{n \to \infty} [a_0; a_1, a_2, a_3, \ldots, a_n].$$

For each of the equations below, approximate the mathematical constant on the left-hand side by truncating and evaluating the continued fraction on the right-hand side. Does numerical convergence occur with a reasonable expenditure of computer time? If so, how many terms are necessary, and what is the final approximation?

22. golden ratio $\phi = [1; 1, 1, 1, 1, \ldots]$
23. $\sqrt{2} = [1; 2, 2, 2, 2, \ldots]$
24. $e = [2; 1, 2, 1, 1, 4, 1, 1, 6, 1, 1, 8, 1, \ldots]$

25. The golden ratio ϕ satisfies the equation $\phi^2 - \phi - 1 = 0$. Let x_n be the truncated continued fraction

$$x_n = [1; \underbrace{1, 1, 1, \ldots, 1}_{n}].$$

Compare the rates of convergence for the absolute error $|\phi - x_n|$ and the absolute residual $|x_n^2 - x_n - 1|$. Do the two measures converge at the same rate, in the technical sense introduced in this chapter, or does one converge faster than the other?