1.2.3 A Little Information about Computers

Although many important numerical algorithms predate the development of computers, the use of digital computers now plays such a major role in numerical mathematics that numerical analysis is often known as "scientific computing." Computers allow us to solve problems that are much, much larger than anything that could have been attempted previously. However, although computers do what they are programmed to do with extraordinary accuracy, certain consequences of how they do computation can have unexpected effects on the results.

By the end of the 1950s, the use of floating-point numbers had become standard for digital computers, although the specifics still varied from machine to machine until the IEEE standard was adopted in 1985. The basic idea of floating-point arithmetic is essentially the same as for scientific notation, except that for a computer the base is (almost always) 2 rather than 10. A number N is represented as

$$N = 0.d_1d_2d_3...d_p \times 2^E = [(d_1)\frac{1}{2} + (d_2)\frac{1}{2^2} + (d_3)\frac{1}{2^3} + \cdots + (d_p)\frac{1}{2^p}] \times 2^E$$

where each digit $d_1, d_2, d_3, \ldots, d_p$ is either 0 or 1. The decimal part, $0.d_1d_2d_3...d_p$ is called the mantissa. The precision of the number depends on the number of digits in the mantissa, i.e., the value of p. The range of numbers that can be represented depends on the range of values for the exponent, E; we indicate that range as $L \leq E \leq U$. If we assume that the binary floating-point number system is normalized, so that the leading digit $d_1 = 1$ (unless $N = 0$), then the representation of each number is unique, no digits are wasted on leading zeros, and there is no need to store the value of d_1 (since it is known to be one).

The number of positive values that can be represented exactly in a normalized binary floating-point system is $V = 2^{p-1}(U - L + 1)$; allowing for representation of positive and negative values, along with zero, gives the total number of "machine numbers" (that is, numbers that can be represented exactly) as $2V + 1$.

It is useful to note that the numbers that can be represented exactly are not evenly distributed between the largest and smallest such numbers, but rather are evenly distributed between successive powers of the base.

EXAMPLE 1.5 A Small Binary Floating-Point Number System

To illustrate these ideas, consider a very small binary floating-point system, with $p = 2$ digits and exponents of $-1, 0$, and 1.

The number of positive machine numbers is $V = 2^1(3) = 6$. They are illustrated in Figure 1.7.

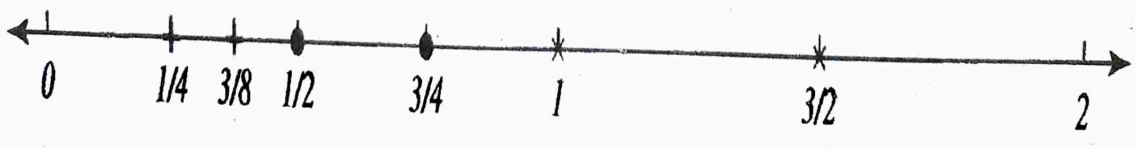


FIGURE 1.7: Exact numbers in base 2.

In the IEEE standard for binary floating-point arithmetic, a (double-precision) real number occupies a 64-bit word, with 11 bits for the signed exponent and 53 bits for the signed digits. The exponents can be any value from $L = -2^{10} = -1024$ to $U = 2^{10} = 1024$, so the numbers that can be represented range in size from $2^{-1024} \approx 10^{-308}$ to $2^{1024} \approx 10^{308}$. Numbers less than 2^{-1024} cause *underflow* and are set to zero. Numbers larger than $0.1111...1 \times 2^{1024}$ cause *overflow*, which usually halts the computation.

The precision is $2^{-53} \approx 10^{-16}$, so numbers are represented to a relative accuracy of about 16 digits.

The unequal distribution of machine numbers leaves a gap between the smallest positive number, 2^{-1024} for double-precision IEEE standard, and zero. To remedy this, the restriction on normalization (that the leading digit must be 1) is relaxed to allow for numbers (sometimes called *subnormals*) with leading digit(s) that are zero, when the exponent is the smallest possible value. This fills in the interval from -1 to 1, but with numbers which have less precision (because they have fewer significant digits in the mantissa). The IEEE standard provides for subnormals. An extended floating-point system with subnormals is said to exhibit *gradual underflow*.

The IEEE standard allows for two exceptional circumstances: `Inf` indicates the result of dividing a finite number by zero, and `NaN` denotes the result of an indeterminant operation, such as $0/0$. How these are treated varies depending on the computing environment. In MATLAB these quantities are carried through a computation in a reasonable manner, although they should be avoided if possible.

Computer Arithmetic

In addition to questions about the representation of real numbers on the computer, there is also the question of the accuracy of computations with real numbers. If there is no underflow or overflow, the result of adding, subtracting, multiplying or dividing two real numbers is "almost exact"; that is, the results of the computer computation differ from the exact results by a factor of $(1+\epsilon)$, where ϵ is no larger than the quantity *eps* which measures the accuracy of the computer (also called "machine epsilon"). For the IEEE standard, $eps = 2^{-53} \approx 10^{-16}$. In general, the value of *eps* depends on the type of rounding used by the system; the IEEE standard specifies rounding to the nearest machine number, which provides for about half of the error that would occur if a number were "chopped." Rounding is considered further in the next section.

There are several slightly different definitions of machine epsilon, but they all have the intent of measuring the "relative granularity" of the floating-point system. If we denote the floating-point representation of a number x as $\mathrm{fl}(x)$, these can be summarized as

- Relative error in representing a nonzero real number: $\left|\dfrac{\mathrm{fl}(x) - x}{x}\right| \leq eps$.
- Smallest number such that $\mathrm{fl}(1 + eps) > 1$.
- Distance from 1 to the next larger floating-point number.

In adding or subtracting floating-point numbers, the exponents must agree. If they do not, the mantissa of the smaller (magnitude) number must be shifted, and some digits may be lost. In floating-point multiplication, the mantissas are multiplied and the exponents added. The additional digits in the mantissa are rounded.

EXAMPLE 1.6 Floating-Point Addition and Multiplication

For example, consider a small base-10 system, with four digits of precision. Let $a = 0.4321 \times 10^2 = 43.21$ and $b = 0.1234 \times 10^{-1} = 0.01234$.

To add, b must be shifted to the form $b = 0.0001234 \times 10^2$; adding gives $a + b = 0.4322234 \times 10^2$, which rounds to $a + b = 0.4322 \times 10^2$. Thus only the first digit of b has any effect on the result.

To multiply, we have $a \times b = 0.05332114 \times 10^{2+(-1)}$, which rounds to $a \times b = 0.0533 \times 10^1$.

Floating-point division illustrates the effect of inexact representation as an additional source of round-off error, in that although 1 and 10 can be represented exactly in binary, the fraction 1/10 cannot.

EXAMPLE 1.7 Conversion from Decimal to Binary

The decimal fractions that can be represented exactly with $p = 3$ digits and $E = -1$ or $E = 0$ are shown in the following table. The leading decimal is 1 for each number, since we are assuming a normalized floating-point system.

Exponent	Binary decimal	Expansion	Decimal
-1	0.0100_2	$(1)(1/4) + (0)(1/8) + (0)(1/16)$	0.25
-1	0.0101_2	$(1)(1/4) + (0)(1/8) + (1)(1/16)$	0.3125
-1	0.0110_2	$(1)(1/4) + (1)(1/8) + (0)(1/16)$	0.375
-1	0.0111_2	$(1)(1/4) + (1)(1/8) + (1)(1/16)$	0.4375
0	0.1000_2	$(1)(1/2) + (0)(1/4) + (0)(1/8)$	0.5
0	0.1010_2	$(1)(1/2) + (0)(1/4) + (1)(1/8)$	0.625
0	0.1100_2	$(1)(1/2) + (1)(1/4) + (0)(1/8)$	0.75
0	0.1110_2	$(1)(1/2) + (1)(1/4) + (1)(1/8)$	0.875

In general, the conversions from decimal to binary and from binary to decimal are sources of round-off error that are not addressed by the IEEE standard, which deals only with internal arithmetic operations.

1.3 SOME BASIC ISSUES

We now introduce some of the recurring themes in the analysis of numerical methods; these ideas are revisited in various settings in the remainder of the text. First we consider some issues related to the question of how good the result of a numerical method is (including measuring error, sources of error, and so on.) Then we investigate the two primary issues for iterative methods: "Does the process converge?" and "When do we stop?" We then introduce the question of how much computational effort is required. Finally we consider some methods for reducing error.

1.3.1 Error

There are several reasons that the results of a numerical solution to a problem from the "real world" may not be the exact answer. Simplifying assumptions made in modeling the original problem are one source of inaccuracies. Errors arising from data collection are another.

In this section we illustrate several basic types of errors that are more directly linked to the numerical solution of the stated problem. We first define some standard terminology for describing errors that occur in numerical methods. We then illustrate two types of error that arise because computers do not represent most numbers in an exact form; that is, they do not do exact arithmetic. The third example in this section illustrates one common form of error introduced in replacing a continuous process by a discrete approximation.

Measuring Error

If x is our approximate result, and the exact (but usually unknown) result is denoted x^*, then the error in using the approximate result is

$$\text{Error}(x) = x^* - x$$

However, especially for problems in which the magnitude of the true value may be very large, or very small, the relative error may be more important than the actual error.

$$\text{Rel Error}(x) = \frac{x^* - x}{x^*}$$

In computing the relative error, the approximate value is often used in the denominator in place of the unknown true value x^*.

Significant Digits

The number x is said to approximate x^* to t significant digits if t is the largest nonnegative integer for which

$$\left|\frac{x - x^*}{x}\right| < 5 \times 10^{-t}$$

Errors from Inexact Representation

The error introduced by shortening a number that has more digits than can be represented by the available floating-point system is known as *round-off error*. There are two basic approaches. The simplest is to chop the number, discarding any digits beyond what the system can accommodate. The other is to round the number; the result depends on the value of the first digit to be discarded. If the system allows for n digits, rounding produces the same result as chopping if the $(n+1)^{st}$ digit is 0, 1, 2, 3, or 4. If the $(n+1)^{st}$ digit is 6, 7, 8, or 9, the n^{th} digit is increased by 1. If the $(n+1)^{st}$ digit is 5, it is common to round so that the n^{th} digit is even, rounding up about half of the time.

The errors that occur from rounding are less likely to accumulate during repeated calculations, since the true value is larger than the rounded value about half of the time and smaller about half of the time. Furthermore, the largest absolute error that can occur is twice as large for chopping as for rounding. On the other hand, chopping requires no decisions as to whether to change the last retained digit. The inaccuracies that result from either rounding or chopping are known as round-off errors.

We now consider some examples of the difficulties that can occur due to inexact computations.

EXAMPLE 1.8 Effect of Order of Operations

As an example of the effect of round-off, consider the following addition problem:

$$0.99 + 0.0044 + 0.0042.$$

With exact arithmetic, the result is 0.9986, regardless of the order in which the additions are performed. However, if we have three-digit arithmetic, and the operations are nested from left to right, we find that

$$(0.99 + 0.0044) + 0.0042 \approx 0.994 + 0.0042 \approx 0.998$$

On the other hand, if we change the nesting so that the small numbers are added together first, we have

$$0.99 + (0.0044 + 0.0042) = 0.99 + 0.0086 \approx 0.999$$

Using the definition of x approximating x^* to t significant digits, we see that 0.998 approximates the true solution, $x^* = 0.9986$, to three significant digits, since

$$\left|\frac{0.998 - 0.9986}{0.998}\right| = 6.012 \times 10^{-4} < 5 \times 10^{-3}$$

On the other hand, 0.999 approximates the true solution, $x^* = 0.9986$, to four significant digits, since

$$\left|\frac{0.999 - 0.9986}{0.999}\right| = 4.004 \times 10^{-4} < 5 \times 10^{-4}$$

For cases with greater difference in the sizes of the numbers, and with more terms, the loss of significance can be extreme.

Cancellation Error

A second example of the effect of inexact calculations occurs when a computation involves the subtraction of two nearly equal numbers. It is advisable to rewrite the formula to avoid the difficulty if possible. Consider the problem of using the quadratic formula to solve the quadratic equation

$$x^2 - bx + 1 = 0$$

The effect of rounding the discriminant $r = \sqrt{b^2 - 4}$ is illustrated in this example; note that for large b, say $(b \gg 4)$, r is quite close to b.

The standard quadratic formula (SQF) gives

$$x_1 = \frac{b + r}{2} \quad \text{and} \quad x_2 = \frac{b - r}{2}$$

If b is positive, x_2 will involve the difference of two numbers that are very close to each other, a dangerous situation.

This difficulty can be avoided by rationalizing the numerator in the quadratic formula, yielding the rationalized quadratic formula (RQF)

$$x_2 = \frac{(b - r)}{2}\frac{(b + r)}{(b + r)} = \frac{(b^2 - r^2)}{2(b + r)} = \frac{4}{2(b + r)} = \frac{2}{(b + r)}$$

If the same process is applied to the formula for x_1 (with which the standard quadratic formula does not have a problem), the rationalized numerator formula results in division by a quantity that is close to zero, which is a much worse situation.

EXAMPLE 1.9 Cancellation Errors

To illustrate the effect of rounding, consider the problem of solving the quadratic equation $x^2 - 97x + 1 = 0$. The exact roots (shown to nine digits) and the approximations computed with rounding to five digits are summarized in the following table.

	x_1	x_2
Exact	96.9896896	0.0103103743
SQF, rounded	96.990	0.01050
RQF, rounded	95.238	0.01031

Effect of Rounding for Roots of Quadratic Equation

For the standard quadratic formula (SQF), rounding has a much larger effect on x_2 than on x_1, as expected.

Using the rationalized quadratic formula (RQF) for x_2 gives the correct result to the number of digits used in the rounded computation. On the other hand, if the rationalized formula is used for x_1, for which it is not appropriate, the results for the rounded computations approximate the true solution only to two digits.

Errors from Mathematical Approximations

Many numerical methods are based on using a few terms of a Taylor series expansion to approximate a function. The error introduced by neglecting higher-order terms (or the remainder term) is known as *truncation error*. As an example, consider the Taylor polynomial approximation with remainder:

$$f(a+h) = f(a) + hf'(a) + \frac{h^2}{2!}f''(c), \quad \text{for some} \quad c \in [a, a+h]$$

In analyzing the error associated with using the trapezoid rule, a Taylor polynomial approximation is used both for the function being integrated and for the function defined as the indefinite integral.

Local Truncation Error

To introduce the analysis of the error produced by approximating a function by a simpler function, consider again the trapezoid rule for numerical integration, which we write now as

$$\int_a^{a+h} f(x)\,dx \approx \frac{h}{2}[f(a) + f(a+h)]$$

Define the function

$$F(t) = \int_a^t f(x)\,dx$$

We can represent F by its Taylor polynomial with remainder, as follows:

$$F(a+h) = F(a) + hF'(a) + \frac{h^2}{2!}F''(a) + \frac{h^3}{3!}F'''(c_1), \quad \text{for some} \quad c_1 \in [a, a+h]$$

Since $F' = f, F'' = f', F''' = f'', \ldots$, and $F(a) = 0$, we have

$$\int_a^{a+h} f(x)\,dx = F(a+h) = hf(a) + \frac{h^2}{2!}f'(a) + \frac{h^3}{3!}f''(c_1) \tag{1.1}$$

On the other hand, the Taylor polynomial with remainder for f is

$$f(a+h) = f(a) + hf'(a) + \frac{h^2}{2!}f''(c_2), \quad \text{for some} \quad c_2 \in [a, a+h]$$

which gives (after a little algebra)

$$\frac{h}{2}[f(a+h) + f(a)] = hf(a) + \frac{h^2}{2}f'(a) + \frac{h^3}{4}f''(c_2) \tag{1.2}$$

The error in the trapezoid rule is the difference of Eqs. (1.1) and (1.2), or

$$\int_a^{a+h} f(x)\,dx - \frac{h}{2}[f(a+h) + f(a)] = \frac{h^3}{6}f''(c_1) - \frac{h^3}{4}f''(c_2)$$

We now show that if f is sufficiently smooth — i.e., f'' is continuous and bounded on $[a, a+h]$ — we can combine these two remainder terms. If

$$m \le f''(x) \le M, \quad \text{for all} \quad x \in [a, a+h]$$

then

$$\frac{h^3}{6}m \le \frac{h^3}{6}f''(c_1) \le \frac{h^3}{6}M$$

and

$$\frac{h^3}{4}m \le \frac{h^3}{4}f''(c_2) \le \frac{h^3}{4}M$$

so

$$-\frac{h^3}{12}m \le \frac{h^3}{6}f''(c_1) - \frac{h^3}{4}f''(c_2) \le -\frac{h^3}{12}M$$

By the intermediate value theorem (applied to f'') there is some point η in the interval $[a, a+h]$ such that

$$f''(\eta) = -\frac{12}{h^3}\left[\frac{h^3}{6}f''(c_1) - \frac{h^3}{4}f''(c_2)\right]$$

Thus we have

$$\int_a^{a+h} f(x)\,dx - \frac{h}{2}[f(a+h) + f(a)] = -\frac{h^3}{12}f''(\eta) \quad \text{for some} \quad \eta \in [a, a+h]$$

This is the local truncation error, which comes from truncating the Taylor series expansions, for one step of the trapezoid rule.

Global Truncation Error

To improve the results, it is useful to subdivide the interval into n equal subintervals $[a, x_1], [x_1, x_2], \ldots, [x_{n-1}, b]$ and apply the method in each region. The length of each subinterval is $h = (b-a)/n$. This gives the more general (composite) trapezoid rule:

$$\int_a^b f(x)\,dx \approx \frac{h}{2}[f(a) + 2f(x_1) + \cdots + 2f(x_{n-1}) + f(b)]$$

The global error is the result of adding the local error in each of the regions. If f is sufficiently smooth, the global error can represented as $\dfrac{-(b-a)h^2}{12}f''(\eta)$ for some $a \le \eta \le b$.

Thus, the global error for the trapezoid rule is proportional to h^2, and the method is $O(h^2)$. This means that if the step size is cut in half, the bound on the global truncation error is reduced by a factor of one-fourth.

Using the Truncation Error

Representation of the approximation error by a term that depends on a derivative of the function, evaluated at some (unknown) point in the interval, and a power of the step size, is useful in several ways. It is sometimes not too difficult to find bounds on the derivative, so that the error term gives bounds on how large or how small the error can be on the interval. Even without finding bounds on the derivative, the power of the step size that occurs in the error term (that is, the order of the method) gives some indication as to how different methods may be expected to compare. A higher-order method does not always give better results than a lower-order method, but in general one can at least expect to obtain more improvement in the results by reducing the step size in a higher-order method.

For some analysis it is useful to represent the truncation error as a series, by retaining all of the terms in the Taylor series expansion used in deriving the method. For some methods, including the trapezoid rule, the series can be expressed as a power series in h with coefficients that do not depend on the derivatives of the function. Especially if that power series depends only on even powers of h, as is the case for the trapezoid rule and several other important numerical methods, there is a technique, known as acceleration (or extrapolation), by which two applications of the method may be combined to obtain an even more accurate result. The series form for the error in the trapezoid rule can be written as

$$\int_a^b f(x)\,dx - T(h) = a_2h^2 + a_4h^4 + a_6h^6 + \cdots$$

Order of a Method

For errors that come from using a finite step size, h, in approximating a continuous process by a discrete one, it is often useful to describe how the error depends on h, as h approaches zero.

We say that a function $f(h)$ is "Big Oh" of h if $|f(h)| \leq c|h|$ for some constant c, when h is near 0. This is written $f(h) = O(h)$. Similarly, $f(h) = O(h^2)$ means that $|f(h)| \leq c|h^2|$ for some constant c, when h is near 0.

If a method has an error term that is $O(h^k)$, the method is often called a k^{th} order method. For example, if we use a Taylor polynomial to approximate the function f at $x = a + h$, we have

$$f(x) = f(a+h) = f(a) + hf'(a) + \frac{h^2}{2!}f''(a) + \frac{h^3}{3!}f'''(\eta), \quad \text{for some} \quad \eta \in [a, a+h]$$

Assuming that f is sufficiently smooth, we let M be the maximum of $f'''(x)$ for $a \leq x \leq a+h$. Then this approximation is $O(h^3)$, since the error, $\frac{h^3}{3!}f'''(\eta)$, satisfies

$$|\frac{h^3}{3!}f'''(\eta)| \leq \frac{1}{3!}M|h^3|$$

1.3.2 Convergence

For iterative techniques, it is imperative to know whether the method converges, i.e., whether the sequence of approximate results approaches the true solution. If the method does converge, we must decide when to terminate the process.

Fixed-Point Convergence

It is very useful to be able to analyze algebraically whether a fixed-point formula will converge, and if so, to estimate how rapidly. The following theorem states conditions which guarantee that the equation $x = g(x)$ has a fixed point in the interval $I = [a, b]$, and the iterative procedure $x_k = g(x_{k-1})$ will converge to that fixed point.

Fixed-Point Convergence Theorem
If

1. $g(x)$ maps $[a, b]$ into $[a, b]$,
2. $g'(x)$ is continuous on $[a, b]$, and
3. there is a number $N < 1$ such that $|g'(x)| \leq N$ for all $x \in [a, b]$,

then

1. $x = g(x)$ has exactly one solution (call it x^*) in the interval $[a, b]$, and
2. the fixed-point iteration $x_k = g(x_{k-1})$ converges to x^*, for any starting estimate in $[a, b]$.

Furthermore, the value of N gives an estimate of the error at any stage of the iteration, in that the error $e_k = x_k - x^*$ satisfies the inequalities

$$|e_{k+1}| \leq N|e_k| \quad \text{and} \quad |e_{k+1}| \leq N^{k+1}|e_0|$$

This theorem includes quite a bit of information; to clarify what it says we may wish to break it down into several parts. We may ask whether an iteration scheme of the form $x_{k+1} = g(x_k)$ has a fixed point. In fact it is quite easy to see that if $g(x)$ is continuous on an interval $[a, b]$ and if $a \leq g(x) \leq b$ for all $a \leq x \leq b$ (that is, $g(x)$ maps the interval $[a, b]$ into $[a, b]$), then $x = g(x)$ has at least one solution in $[a, b]$. This follows from applying the intermediate value theorem to the continuous function $h(x) = g(x) - x$.

The next question might be whether the fixed point is unique. In fact both the uniqueness of the fixed point and the convergence of the iterations follows if there is a constant λ between 0 and 1 so that $|g(x) - g(y)| \leq \lambda|x - y|$ for all $x, y \in [a, b]$ (where, as before, $g(x)$ is continuous and maps $[a, b]$ into $[a, b]$). To show uniqueness, one need only assume that there are two distinct fixed points and show a contradiction; we denote this unique fixed point as x^*. Now, to show convergence, consider the error at the k^{th} stage,

$$|x^* - x_{k+1}| = |g(x^*) - g(x_k)| \leq \lambda|x^* - x_k|$$

Since $\lambda < 1$, the error is reduced at each stage.

Finally, we consider how we can find λ. If $g(x)$ is differentiable, then using the mean value theorem, we find that we can take $\lambda = \text{Max}(|g'(x)|)$ for $x \in [a, b]$. Thus we obtain the condition that $\text{Max}(|g'(x)|) < 1$ in order to guarantee convergence of the iterations.

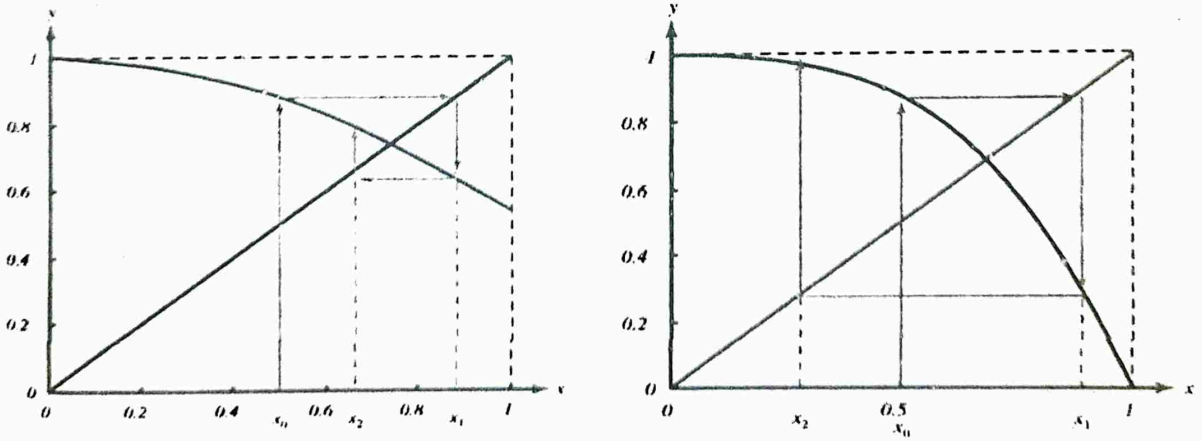
It is easy to construct examples of functions $g(x)$ that do not map a given interval $I = [a, b]$ into I and for which the curves $y = x$ and $y = g(x)$ do not cross (at least for x in I). If g is continuous and does map I into I, then the curves will cross (at least once) in the interval. The guarantee of convergence of the iterative process hinges on the magnitude of $g'(x)$ being less than 1, at least near the fixed point.

The next example illustrates geometrically the convergence of fixed-point iteration for one function, and the divergence of the method for another. The use of the fixed-point convergence theorem is also illustrated for both functions shown.

EXAMPLE 1.10 Fixed-Point Iteration

To illustrate what the theorem says, consider the fixed-point iteration $x = \cos(x)$, starting with $x_0 = 0.5$, illustrated in Figure 1.8a. For $0 \le x \le 1, \cos(x)$ is also between 0 and 1, so $g(x)$ does map $[0, 1]$ into $[0, 1]$. Furthermore, $g'(x) = -\sin(x)$ ranges between 0 and $-\sin(1)$, so $|g'(x)| \le 0.85$ for all x in $[0, 1]$, and the theorem guarantees convergence of the iterations.

On the other hand, for $x = g(x) = 1 - x^3$, shown in Figure 1.8b, we have $g'(x) = -3x^2$, which is not bounded by a number less than 1 on $[0, 1]$. Although there is a fixed point in the interval (and $g(x)$ does map $[0, 1]$ into $[0, 1]$), the conditions of the theorem are not satisfied, and in fact, the iterations do not converge. The difficulty is that in any neighborhood of the fixed point, $|g'(x)| > 1$.



a. Convergent fixed-point iteration

b. Divergent fixed-point iteration

FIGURE 1.8: Fixed-point iterations.

As a practical guide, this kind of iteration is seldom used unless $|g'(x)| \le 1/2$ in the vicinity of the fixed point. If $-1 < g'(x) \le 1/2$ and x^* is the true fixed point, then $|x_k - x^*| \le |x_k - x_{k-1}|$, so the change in successive iterates gives a bound on the error. (See Jensen and Rowland, 1975, for further discussion.)

Measuring Convergence

For a convergent sequence, two quantities describe how rapidly the sequence converges, or, in other words, how rapidly the error is reduced. The first describes the "order of convergence." The second describes the "rate of convergence."

The standard method of comparing how fast various iterative zero-finding methods converge is to investigate the behavior of

$$\frac{|x_k - x^*|}{|x_{k-1} - x^*|^p}$$

for large values of k; x^* denotes the true solution.

Order of Convergence

There are several equivalent statements of the condition for a sequence to converge with a given order and rate. Each involves the existence of a constant $\lambda > 0$:

If, for some $\lambda > 0$,

$$\lim_{k \to \infty} \frac{|x_k - x^*|}{|x_{k-1} - x^*|^p} = \lambda$$

or

$$|x_k - x^*| \approx \lambda |x_{k-1} - x^*|^p$$

or

$$|x_k - x^*| \leq \lambda |x_{k-1} - x^*|^p$$

we say that the sequence x_k converges to x^* with order $p > 0$. In general, higher values of p give faster convergence.

The number λ is called the asymptotic error constant (it gives the rate of convergence).

If a sequence converges with order $p = 1$, we say it is *linearly convergent*. The rate of convergence for linear convergence must be less than 1. For example, the fixed-point iteration scheme discussed above is linearly convergent (as long as the magnitude of the derivative is less than 1, and the other conditions of the theorem are satisfied); the rate of convergence depends on the maximum magnitude of the derivative. Using induction, an alternative (and sometimes more convenient) form of linear convergence is given by

$$|x_k - x^*| \leq \lambda^k |x_0 - x^*|$$

If a sequence converges with $p = 2$, we say it *converges quadratically*.

Termination Conditions

A variety of conditions can be used for deciding when to stop an iterative procedure; however, there is no perfect test for a "stopping condition." The conditions may be characterized as being of the general types: the problem is "solved," the iteration has "converged," the iteration process has continued "long enough," or "the solution is looking bad." For the example of finding the zero of $f(x) = x^2 - 3$, with the true zero denoted x^*, possible convergence tests include the following:

The problem is "solved": function value is sufficiently small
$|f(x_k)| \leq f_{\text{tol}}$.

The iteration has "converged": absolute change is sufficiently small
$|x_{k+1} - x_k| \leq \text{tol}$;
if $\text{tol} = 10^{-n}$, then x_{k+1} should approximate x^* to n decimal places.

The iteration has "converged": relative change is sufficiently small
$\left|\frac{x_{k+1} - x_k}{x_{k+1}}\right| \leq \text{tol}$;
if $\text{tol} = 10^{-n}$, then x_{k+1} should approximate x^* to n significant digits.

The iterations have gone on "long enough": number of iterations is too large
$k \geq \text{kmax}$.

The solution is "looking bad": function value is too large
$|f(x_k)| \geq f_{\text{big}}$.

The solution is "looking bad": value of variable is too large
$|x_k| \geq x_{\text{big}}$.

Caveats

It is important to realize that none of these tests guarantees the desired result, namely that $|x_k - x^*| < \text{tol}$.

In addition, an iterative process could pass the successive-iterates test at the same time that the iterates were diverging to ∞. As an example, consider a process in which

$$x_k = 1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{k}$$

The difference between successive iterates, $|x_{k+1} - x_k| = 1/(k+1) \to 0$ as $k \to \infty$, but $x_{k+1} \to \infty$ as $k \to \infty$.

Choosing an Appropriate Test

The relative-change test is appropriate for problems in which the desired roots may be of greatly differing magnitudes. It is not suitable, however, if $x = 0$ is a desired root and the method is converging rapidly, since that would produce a relatively large change in the iterates.

- Etter, D. M., and D. C. Kincaid, [illegible], Upper Saddle River, NJ, 1999.
- Hanselman, D., and B. Littlefield, *Mastering MATLAB 6: A Comprehensive Tutorial and Reference*, Prentice Hall, Upper Saddle River, NJ, 2001.

Practice the Techniques

P1.1. Use the Babylonian method (as in Example 1.1) to find

(a) $\sqrt{5}$

(b) $\sqrt{7}$

(c) $\sqrt{8}$

For Problems P1.2–P1.6, use Gaussian elimination (as in Example 1.2) to solve the linear system.

(a) *Solve the system as given.*

(b) *Graph the given equations, and graph the transformed second equation after the elimination step.*

(c) *Repeat parts (a) and (b) for the system consisting of the same two equations, but given in reverse order.*

P1.2.
$$\begin{aligned} 4x + y &= 6 \\ -x + 2y &= 3 \end{aligned}$$

P1.3.
$$\begin{aligned} 5x + y &= 17 \\ -10x + 17y &= 4 \end{aligned}$$

P1.4.
$$\begin{aligned} 3x + y &= 4 \\ 6x + 7y &= 13 \end{aligned}$$

P1.5.
$$\begin{aligned} 4x + y &= 9 \\ x + 2y &= 4 \end{aligned}$$

P1.6.
$$\begin{aligned} 3x + y &= 6 \\ x + 5y &= 16 \end{aligned}$$

For Problems P1.9–P1.7 approximate the integral

(a) *using the basic trapezoid rule.*
(b) *using the composite trapezoid rule ($n = 2$).*
(c) *using acceleration with the results from parts (a) and (b).*

P1.7.
$$\int_0^2 2^x \, dx$$

P1.8.
$$\int_0^{\pi/2} \sin(x) \, dx$$

P1.9.
$$\int_0^2 \frac{1}{1+x^2} \, dx$$

Problems P1.10–P1.14 illustrate the use of the Gerschgorin theorem to find bounds on the eigenvalues of a matrix. Find the Gerschgorin circles for each row of the given matrix. Graph the regions, and give bounds on the eigenvalues.

P1.10.
$$\mathbf{A} = \begin{bmatrix} 1 & 1/8 & 1/4 \\ 1/2 & 2 & 0 \\ 0 & 0 & 3 \end{bmatrix}$$

P1.11.
$$\mathbf{A} = \begin{bmatrix} 1 & 1/3 & 1/3 \\ 1/4 & 2 & 1/3 \\ 1/2 & 1/4 & 3 \end{bmatrix}$$

P1.12.
$$\mathbf{A} = \begin{bmatrix} 1 & 1/4 & 0 \\ 1/4 & 2 & 1/4 \\ 0 & 1/4 & 3 \end{bmatrix}$$

P1.13.
$$\mathbf{A} = \begin{bmatrix} 1 & 3/4 & 0 \\ 1/2 & 2 & -1/8 \\ 0 & 1/8 & 3 \end{bmatrix}$$

P1.14.
$$\mathbf{A} = \begin{bmatrix} 1 & -1/2 & 0 \\ 1/2 & 2 & 1/8 \\ 0 & 1/8 & 3 \end{bmatrix}$$

Problems P1.15–P1.16 illustrate the effect of round-off error in adding numbers of differing magnitudes.

(a) *Add from left to right, rounding to three digits at each step.*
(b) *Add from right to left, rounding to three digits at each step.*
(c) *Compare the relative error for parts (a) and (b).*

P1.15. $100 + 0.49 + 0.49$

P1.16. $10.0 + 0.333 + 0.333 + 0.333$

Problems P1.17–P1.18 illustrate the effect of round-off in the quadratic formula.

(a) *Use the standard quadratic formula, rounding to four digits.*
(b) *Use the rationalized-numerator quadratic formula, rounding to four digits.*
(c) *Compare the results from parts (a) and (b) with the results found without rounding.*

P1.17.
$$x^2 - 57x + 1 = 0.$$

P1.18.

$$x^2 - 23x + 1 = 0.$$

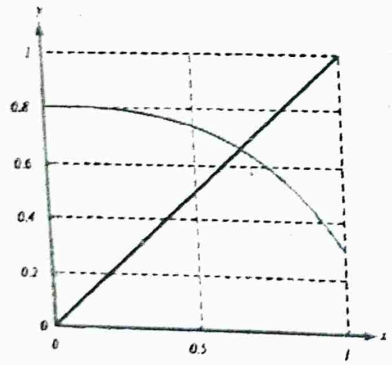
For Problems P1.19–P1.22, investigate the convergence of the fixed-point iteration formula.

(a) *Show the first three iterations graphically.*

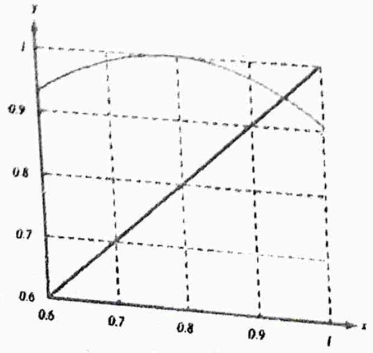
(b) *Find the first three iterates algebraically.*

(c) *Determine which of the conditions of the fixed-point convergence theorem are satisfied, and which (if any) are not.*

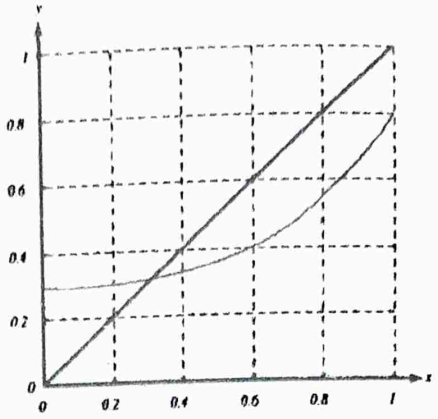
P1.19. $x = g(x) = -0.5x^3 + 0.8$



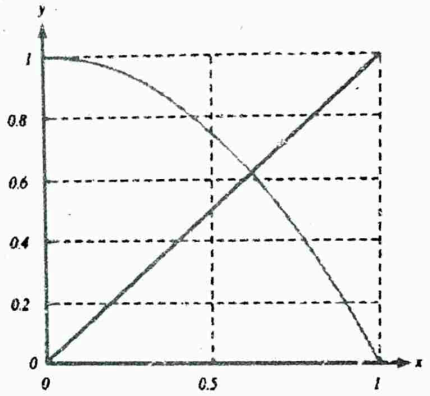
P1.20. $x = g(x) = \sin(2x)$



P1.21. $x = g(x) = 0.5x^3 + 0.3$



P1.22. $x = g(x) = 1 - x^2$



Problems P1.23–P1.26 illustrate the effect of the order of the equations in Gaussian elimination (with error).

(a) *Solve the equations in the order given; determine bounds on x and y. Graph the given equations and the transformed second equation.*

(b) *Solve the equations in reverse order; determine bounds on x and y. Graph the given equations and the transformed second equation.*

(c) *Compare the relative error in the values of x and y found in parts (a) and (b).*

P1.23.
$$\begin{aligned} 4x + y &= 6 \pm 0.1 \\ -x + 2y &= 3 \pm 0.1 \end{aligned}$$

P1.24.
$$\begin{aligned} 5x + y &= 17 \pm 0.2 \\ -10x + 17y &= 4 \pm 0.2 \end{aligned}$$

P1.25.
$$\begin{aligned} 3x + y &= 6 \pm 0.2 \\ x + 5y &= 16 \pm 0.2 \end{aligned}$$

Answers

Answers to selected problems.
Answers rounded to 4 decimal places, or truncated to 2 decimal places.

Chapter 1

P1.1 (a) x = 2.2361
(b) x = 2.6458
(c) x = 2.8284

P1.3 x = 3, y = 2

P1.5 x = 2, y = 1

P1.7 (a) S1 = 5
(b) S2 = 4.5
c) S = (4 S2 - S1)/3 = 4.33333

P1.9 (a) S1 = 1.2
(b) S2 = 1.1
c) S = (4 S2 - S1)/3 = 1.0667

P1.11 1/3 < m1, m2, m3 < 15/4

P1.13 1/4 < m1, m2 < 21/8,
23/8 < m3 < 25/8

P1.15 exact 100.98
(a) 100, RelEr = 0.0097
(b) 101, RelEr = -0.00019806

P1.17 (a) x1 =56.98, x2 =0.02
(b) x1 =50.00, x2 =0.0176
(c) x1 =56.9825, x2 =0.0175

P1.19 x0 =0; x1 =0.3;
x2 =0.3135; x3 =0.3154
cond. 3 fails, |g'| not < 0

P1.21 x0 =0; x1 =0.8;
x2 =0.5440; x3 =0.7195
cond. 3 fails, |g'| not < 0

P1.23 actual intersections
(1.0111,2.0556), (0.9889,1.9444)
(0.9667,2.0333), (1.0333,1.0667)
approx. solutions, given order
(1.0111,2.0556), (0.9889,1.9444)
(0.9611,2.0556), (1.0389,1.9444)
approx. solutions, reverse order
(1.0111,2.0556), (0.9889,1.9444)
(1.2112,2.0556), (0.7888,1.9444)

P1.25 actual intersections
(1.0400,0.9800), (0.9600,1.0200)
(1.0533,0.9400), (0.9467,1.0600)
approx. solutions, given order
(1.0400,0.9800), (0.9600,1.0200)
(0.9733,0.9800), (1.0267,1.0200)
approx. solutions, reverse order
(1.0400,0.9800), (0.9600,1.0200)
(1.0067,0.9800), (0.9933,1.0200)

Chapter 2

P2.1 x = 1.4142

P2.3 x = 2.6458

P2.5 x = 1.5874

P2.7 x = 0.8190

P2.9 x = 0.4949

P2.11 x = -3.1055, 0.2235, 2.8820

P2.13 x = -2.8794, -0.6527, 0.5321

P2.15 x = 2.8063

P2.17 x = 0, 1.3333, 2.5

P2.19 x = -3.3240, -1.6197, 5.9437

P2.21 (a) x alternates
(b) x = 0.2541;
(c) x0=1, x alternates
x0=0.5, x = 0.4472;
x0=0, f'(x0)=0; method fails.
(d) x = 0.8706

P2.23 x = 2.0288, 4.9132, 7.9787

P2.25 x = -0.70347

P2.27 (a) x = 2.4781, 3
(b) x = 2.7368, 2.7

P2.29 x = 0.12313

Chapter 3

P3.1 x =[1; 2; 3]

P3.3 x =[0.1; 0.7; -0.9]

P3.5 x =[-1; 3; 2]

P3.7 x =[-1; 2; 0; 1]

P3.9 x =[1; 2; 6]

(ii) Several million operations are performed per [illegible] check the intermediate results for possible build up of large errors during calculations.

In what follows, we examine these two aspects in detail.

1.2 COMPUTER ARITHMETIC

The basic arithmetic operations performed by the computer are addition, subtraction, multiplication and division. The decimal numbers are first converted to the machine numbers consisting of only the digits 0 and 1 with a **base** or **radix** depending on the computer. If the base is two, eight or sixteen the number system is called **binary, octal** or **hexadecimal** respectively. The decimal number system has the base 10. The decimal integer number 4987 actually means

$$(4987)_{10} = 4 \times 10^3 + 9 \times 10^2 + 8 \times 10^1 + 7 \times 10^0 \qquad (1.1)$$

which represents a polynomial in the base 10. Similarly, a fractional decimal number 0.6251 means

$$(0.6251)_{10} = 6 \times 10^{-1} + 2 \times 10^{-2} + 5 \times 10^{-3} + 1 \times 10^{-4} \qquad (1.2)$$

which is a polynomial in 10^{-1}.

Combining (1.1) and (1.2), we may write the number 4987.6251 in decimal system as

$$\begin{aligned}(4987.6251)_{10} = {} & 4 \times 10^3 + 9 \times 10^2 + 8 \times 10^1 + 7 \times 10^0 \\ & + 6 \times 10^{-1} + 2 \times 10^{-2} + 5 \times 10^{-3} + 1 \times 10^{-4}\end{aligned} \qquad (1.3)$$

where the subscripts denote the base of the number system.

Thus, a number $N = (d_{n-1}\, d_{n-2} \cdots d_0 \cdot d_{-1}\, d_{-2} \cdots d_{-m})$ in decimal system can always be expressed in the form

$$(N)_{10} = d_{n-1}\, 10^{n-1} + d_{n-2}\, 10^{n-2} + \cdots + d_1\, 10^1 + d_0 10^0 + d_{-1}\, 10^{-1} + d_{-2}\, 10^{-2} + \cdots + d_{-m}\, 10^{-m} \tag{1.4}$$

where $d_{n-1}, d_{n-2}, \cdots, d_{-m}$ are any digits between 0 and 9.

Binary Number System

Binary number system has base 2 with digits 0 and 1 called **bits** and any number N can be written as

$$(N)_2 = b_{n-1}\, b_{n-2} \cdots b_1\, b_0 \cdot b_{-1}\, b_{-2} \cdots b_{-m} \tag{1.5}$$

where $b_{n-1}, b_{n-2}, \cdots, b_{-m}$ are binary bits 0 or 1 and the point is called the binary point. The corresponding decimal number is easily calculated by using the formula

$$(N)_{10} = b_{n-1}\, 2^{n-1} + b_{n-2}\, 2^{n-2} + \cdots + b_1 2^1 + b_0 2^0 + b_{-1}\, 2^{-1} + b_{-2}\, 2^{-2} + \cdots + b_{-m}\, 2^{-m}. \tag{1.6}$$

Example 1.1 Find the decimal number corresponding to the binary number $(111.011)_2$.

We have

$$\begin{aligned}(111.011)_2 &= 1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 + 0 \times 2^{-1} \\ &\quad + 1 \times 2^{-2} + 1 \times 2^{-3} \\ &= (7.375)_{10}\end{aligned}$$

We now consider the conversion of the integer N in decimal system to the binary number $b_{n-1}\, b_{n-2} \cdots b_1\, b_0$. We write

$$b_{n-1}\, 2^{n-1} + b_{n-2}\, 2^{n-2} + \cdots + b_1\, 2^1 + b_0 = N. \tag{1.7}$$

The last binary digit b_0 in (1.7) is zero if and only if N is even. The second digit b_1 is zero if and only if $(N - b_0)/2$ is even, and so on. Thus, we have

$$\begin{aligned} N_0 &= N \\ N_{k+1} &= \frac{N_k - b_k}{2},\ k = 0, 1, 2 \cdots \end{aligned} \tag{1.8}$$

until $N_k = 0$, where

$$b_k = \begin{cases} 1, & \text{if } N_k \text{ is odd} \\ 0, & \text{if } N_k \text{ is even} \end{cases}$$

Example 1.2 Convert $(58)_{10}$ to the corresponding binary number.

We have

$$N_0 = 58, \qquad b_0 = 0$$
$$N_1 = (58 - 0)/2 = 29, \qquad b_1 = 1$$

$$N_2 = (29 - 1)/2 = 14, \qquad b_2 = 0$$
$$N_3 = (14 - 0)/2 = 7, \qquad b_3 = 1$$
$$N_4 = (7 - 1)/2 = 3, \qquad b_4 = 1$$
$$N_5 = (3 - 1)/2 = 1, \qquad b_5 = 1$$
$$N_6 = (1 - 1)/2 = 0$$

Thus, the binary number is 111010.

Next, we convert the fraction N in decimal system to binary fraction.

$b_{-1}\ b_{-2} \cdots b_{-m}$.

We write

$$b_{-1}\ 2^{-1} + b_{-2}\ 2^{-2} + \cdots + b_{-m}\ 2^{-m} = N,\ 0 < N < 1 \tag{1.9}$$

where $b_{-1}, b_{-2}, \cdots, b_{-m}$ are 0 or 1.

It is easily seen that b_{-1} is 1 if and only if $2N \geq 1$ and 0 if and only if $2N < 1$. Similarly, using (1.9), we may determine b_{-k} recursively as follows:

$$N_1 = N$$
$$b_{-k} = \begin{cases} 1, & \text{if } 2N_k \geq 1 \\ 0, & \text{if } 2N_k < 1 \end{cases}$$
$$N_{k+1} = 2N_k - b_{-k},\ k = 1, 2, \cdots \tag{1.10}$$

Example 1.3 Convert $(0.859375)_{10}$ to the corresponding binary fraction.

We have

k	b_{-k}	N_{k+1}
0		0.859375
		× 2
1	1	0.718750
		× 2
2	1	0.437500
		× 2
3	0	0.875000
		× 2
4	1	0.750000
		× 2
5	1	0.500000
		× 2
6	1	0.000000

The required binary fraction becomes

$$(0.859375)_{10} = (0.110111)_2$$

Example 1.4 Convert $(0.7)_{10}$ to the corresponding binary fraction.

We have

k	b_{-k}	N_{k+1}
0		0.7
		× 2
1	1	0.4
		× 2
2	0	0.8
		× 2
3	1	0.6
		× 2
4	1	0.2
		× 2
5	0	0.4
		× 2
6	0	0.8
		× 2
7	1	0.6
		× 2
8	1	0.2
		× 2
9	0	0.4

Thus we obtain

$$(0.7)_{10} = (.101100110\cdots)_2$$

which is a never ending sequence. If only 7 bits are retained in the binary fraction then the corresponding decimal number becomes

$$\begin{aligned}(0.1011001)_2 &= 1 \times 2^{-1} + 0 \times 2^{-2} + 1 \times 2^{-3} + 1 \times 2^{-4} + 0 \times 2^{-5} \\ &\quad + 0 \times 2^{-6} + 1 \times 2^{-7} \\ &= 0.6953125\end{aligned}$$

which is not exactly the same as the given number. The difference

$$0.7 - 0.6953125 = 0.0046875$$

is the round-off error.

Octal and Hexadecimal System

The octal system has base 8 and uses the digits 0, 1, 2, 3, 4, 5, 6, 7. Similarly, the hexadecimal system has base 16 and uses the digits 0 to 9 and A, B, C, D, E, F to represent 10, 11, 12, 13, 14, 15 respectively.

Decimal numbers can be converted to octal or hexadecimal numbers in a similar manner. The conversion between binary and octal or between binary and hexadecimal is simple due to the

relationship between the bases, $8 = 2^3$ for octal and $16 = 2^4$ for hexadecimal. We convert a binary number to an octal number by grouping the binary bits in groups of three to the right and left of the binary point by adding sufficient zeros to complete the groups and replacing each group of three bits by its octal equivalent. Similarly, to convert a binary number to a hexadecimal number, we form groups of four of the binary bits and replace it by the corresponding digit in the hexadecimal system. The hexadecimal system is sometimes referred to as **hex**.

Example 1.5 Convert the binary number

$$1101001.1110011$$

to the octal and the hexadecimal systems.

We have

$$(1101001.1110011)_2 = (001101001.111001100)_2$$
$$= (151.714)_8$$

and

$$(1101001.1110011)_2 = (01101001.11100110)_2$$
$$= (69.E6)_{16}$$

Floating Point Arithmetic

The first step in the computation with digital computers is to convert the decimal numbers to another number system with the base (say) β understandable to that particular computer and then to store these converted numbers in computer memory. The memory of the digital computer is divided into separate cells called **words**. Each word can hold the same number of digits called bits, with respect to its base plus a sign. Negative numbers are stored as absolute values plus a sign or in complement form. The number of digits which can be stored in a computer word is called its word length. The word length varies from one computer to another. The numbers in the computer word can be stored in two forms:

(i) Fixed-point form.

(ii) Floating-point form.

In **fixed-point** form a t digit number is assumed to have its decimal point at the left-hand end of the word. This implies that all numbers are assumed to be less than 1 in magnitude. The fixed-point number with base β and t digits word length may be written as

$$\pm \sum_{k=1}^{t} a_k \beta^{-k}$$

where $0 \le a_k < \beta$.

If y is any real number and y^* is its machine representation, then the error in y^* is at most $a_{t+1}\, \beta^{-(t+1)}$ or

$$|y - y^*| \le \beta^{-t}. \tag{1.11}$$

To avoid the difficulty of keeping every number less than 1 in magnitude during computation, most computers use **floating-point** representation for a real number. A floating-point number is characterized by four parameters, the base β, the number of digits t, and the exponent range (m, M).

Definition 1.1 A floating-point number is a number represented in the form

$$\cdot d_1 d_2 \cdots d_t \times \beta^e \tag{1.12}$$

where $d_1, d_2, \cdots, d_t$ are integers and satisfy $0 \le d_i < \beta$ and the exponent e is such that $m \le e \le M$.

The fractional part $\cdot d_1 d_2 \cdots d_t$ is called the **mantissa** and it lies between + 1 and – 1. The number 0 is written as

$$+ .00 \cdots 0 \times \beta^e \tag{1.13}$$

Definition 1.2 A non-zero floating-point number as defined in (1.12) is in **normal form** if the value of the mantissa lies in the interval $(-1, -1/\beta]$ or in the interval $[1/\beta, 1)$.

Example 1.6 Subtract the floating-point numbers 0.36143447×10^7 and 0.36132346×10^7.

We have

$$\begin{array}{r} 0.36143447 \times 10^7 \\ -\,0.36132346 \times 10^7 \\ \hline 0.00011101 \times 10^7 \end{array}$$

The result is a floating-point number, but not a normalized floating-point number due to the presence of three leading zeros. Shifting the fractional part three places to the left, we get the result 0.11101×10^4 which is a normalized floating-point number.

Definition 1.3 A non-zero floating-point number as defined in (1.12) is in **t-digit-mantissa standard form** if it is normalized and its mantissa consists of exactly t-digits.

If a number x has the representation in the form

$$x = \cdot d_1\, d_2 \cdots d_t\, d_{t+1} \cdots \times \beta^e \tag{1.14}$$

then the floating-point number $fl(x)$ in t-digit-mantissa standard form can be obtained in the following two ways:

(i) *Chopping*: Here we neglect $d_{t+1}, d_{t+2}, \cdots$ in (1.14) and obtain

$$fl(x) = \cdot d_1\, d_2 \cdots d_t \times \beta^e \tag{1.15}$$

(ii) *Rounding*: Here the fractional part in (1.14) is written as

$$\cdot d_1\, d_2 \cdots d_t\, d_{t+1} + \frac{1}{2}\beta \tag{1.16}$$

and the first t digits are taken to write the floating-point number.

Example 1.7 Find the sum of $.123 \times 10^3$ and $.456 \times 10^2$ and write the result in three-digit mantissa form.

The number of the smaller magnitude is adjusted so that its exponent is same as that of the number of larger magnitude. We have

$$\begin{array}{r} .1230 \times 10^3 \\ .0456 \times 10^3 \\ \hline .1686 \times 10^3 \end{array} = \begin{cases} .168 \times 10^3, & \text{for chopping} \\ .169 \times 10^3, & \text{for rounding.} \end{cases}$$

1.3 ERRORS

A computer has a finite word length and so only a fixed number of digits are stored and used during computation. This would mean that even in storing an exact decimal number in its converted form in the computer memory, an error is introduced. This error is machine dependent and is called **machine epsilon**. After the computation is over, the result in the machine form (with base β) is again converted to decimal form understandable to the users and some more error may be introduced at this stage.

We now discuss the effect of the errors on the results. The quantity,

$$|\text{True value} - \text{Approximate value}|$$

is called the **error**. In order to determine the accuracy in an approximate solution to a problem, either we find the bound of the

$$\textbf{Relative error} = \frac{|\text{Error}|}{|\text{True value}|} \tag{1.17}$$

or of the

$$\textbf{Absolute error} = |\text{Error}| \tag{1.18}$$

Neglecting a blunder or mistake, the errors may be classified into the following types:

Definition 1.4 The **inherent error** is that quantity which is already present in the statement of the problem before its solution.

The inherent error arises either due to the simplified assumptions in the mathematical formulation of the problem or due to the errors in the physical measurements of the parameters of the problem.

Definition 1.5 The **round-off error** is the quantity R which must be added to the finite representation of a computed number in order to make it the true representation of that number.

Thus, if x is the computed number given by (1.14)

$$x = \cdot d_1\ d_2 \cdots d_t\ d_{t+1} \cdots \times \beta^e$$

then the relative error (1.17) for t-digit mantissa standard form representation of x becomes

$$\frac{|x - fl(x)|}{|x|} \leq \begin{cases} \beta^{1-t} & \text{for chopping} \\ \frac{1}{2}\beta^{1-t} & \text{for rounding.} \end{cases} \tag{1.19}$$

Thus, the bound on the relative error of a floating-point number is reduced by half when rounding is used than chopping. It is for this reason that on most computers rounding is used. We write

$$fl(x) = x(1 + \delta) \tag{1.20}$$

where $\delta = \delta(x)$, some number depending on x, is called the relative round-off error in $fl(x)$. The number δ is called the **machine epsilon** and is denoted by EPS. From (1.19) we have

$$|\delta(x)| = EPS = \begin{cases} \beta^{1-t} & \text{for chopping} \\ \frac{1}{2}\beta^{1-t} & \text{for rounding.} \end{cases} \tag{1.21}$$

Definition 1.6 The **truncation error** is the quantity T which must be added to the true representation of the quantity in order that the result be exactly equal to the quantity we are seeking to generate.

This error is a result of the approximate formulas used which are generally based on truncated series. The Taylor series with a remainder is an invaluable tool in the study of the truncation error.

Example 1.8 Obtain a second degree polynomial approximation to

$$f(x) = (1 + x)^{1/2},\ x \in [0, 0.1]$$

using the Taylor series expansion about $x = 0$. Use the expansion to approximate $f(0.05)$ and find a bound of the truncation error.

We have

$$f(x) = (1 + x)^{1/2}, \qquad f(0) = 1$$

$$f'(x) = \frac{1}{2}(1 + x)^{-1/2}, \qquad f'(0) = \frac{1}{2}$$

$$f''(x) = -\frac{1}{4}(1 + x)^{-3/2}, \qquad f''(0) = -\frac{1}{4}$$

$$f'''(x) = \frac{3}{8}(1 + x)^{-5/2}$$

Thus, the Taylor series expansion with remainder term may be written as

$$(1 + x)^{1/2} = 1 + \frac{x}{2} - \frac{x^2}{8} + \frac{1}{16}\frac{x^3}{\left[(1+\xi)^{1/2}\right]^5},\ 0 < \xi < 0.1.$$

The truncation term is given by

$$T = (1 + x)^{1/2} - \left(1 + \frac{x}{2} - \frac{x^2}{8}\right)$$

$$= \frac{1}{16}\frac{x^3}{\left[(1+\xi)^{1/2}\right]^5}.$$

We have $f(0.05) \approx 1 + \dfrac{0.05}{2} - \dfrac{(0.05)^2}{8} = 0.10246875 \times 10^1.$

The bound of the truncation error, for $x \in [0, 0.1]$ is

$$|T| \le \max_{0 \le x \le 0.1} \frac{(0.1)^3}{16\left[(1 + x)^{1/2}\right]^5}$$

$$\le \frac{(0.1)^3}{16} = 0.625 \times 10^{-4}.$$

Significant Digits and Numerical Instability

When a number x is written in normalized floating point form with t digits in base β as given in (1.15), we say that the number has t **significant digits**. The leading digit d_1 is called the most significant digit. In other words, a number x^* is an approximation to x to t significant digits if

$$\frac{|x - x^*|}{|x|} \leq \frac{1}{2}\beta^{1-t} \tag{1.22}$$

in the base β. As an example, $x^* = 0.3$ agrees with 1/3 to one significant digit and $x^* = 0.3333$ agrees with 1/3 to four significant digits in the base $\beta = 10$. Suppose now, that x^* and y^* are approximations to x and y to t significant digits and we wish to calculate the number $z = x - y$. Then $z^* = x^* - y^*$ is an approximation to z which is also correct to t significant digits unless x^* and y^* have one or more leading digits same. In this case, there will be cancellation of digits during subtraction and z^* will not be accurate to t significant digits. For example, if we have

$$x^* = 0.178693 \times 10^1$$

$$y^* = 0.178439 \times 10^1$$

each correct to six digits in decimal system, then

$$z^* = 0.000254 \times 10^1$$

is correct to only three significant digits.

It may be noted that the relative error in x is $(1/2) \times 10^{-5}/(0.178693 \times 10^1) \approx 2.8 \times 10^{-6}$, while that in $x - y$ is $((1/2) \times 10^{-5} + (1/2) \times 10^{-5})/(0.000254 \times 10^1) \approx 3.9 \times 10^{-3}$. When this number z^* is used in further arithmetic calculations, error may considerably increase. Thus, we find that subtraction of two nearly equal numbers causes a considerable loss of significant digits and may greatly magnify the error in later calculations.

A similar loss in significant digits occurs when a number is divided by a small number. For example, consider

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

and we want to calculate $f(x)$ for $x = 0.9$. The exact value of $f(x)$ for $x = 0.9$ correct to six digits is

$$f(x) = 0.526316 \times 10^1.$$

If x is approximated to $x^* = 0.900005$, that is, an error is introduced in the sixth decimal place, we obtain the value

$$f(x^*) = 0.526341 \times 10^1.$$

Therefore, an error in the sixth place in x has caused an error in the fifth place in $f(x)$. Thus, when a number is divided by a very small number (or multiplied by a large number) there is loss of significant digits and magnification of errors in the result.

It is noticed, therefore, that every arithmetic operation performed during computation, gives rise to some error, which once generated may decay or grow in subsequent calculations. In some cases, errors may grow so large as to make the computed result totally redundant and we call such a procedure

numerically unstable. In some cases it can be avoided by changing the calculation procedure, which avoids subtractions of nearly equal numbers or division by a small number or by retaining more digits in the mantissa.

While finding numerical solution of problems of scientific nature, we often encounter problems with inherent instability or induced instability. Inherent instability may arise due to the ill-conditionedness of the problem. We cannot avoid inherent instability by changing the method of solution. It is the property of the problem itself. Sometimes, we can avoid this instability by suitable reformulation of the problem. For example, consider the **Wilkinson's** example of finding the zeros of a polynomial. The polynomial

$$\begin{aligned} P_{20}(x) &= (x-1)\,(x-2)\,\cdots\,(x-20) \\ &= x^{20} - 210x^{19} + \cdots + 20! \end{aligned} \tag{1.23}$$

has the zeros 1, 2, ⋯, 20. Let the coefficient of x^{19} be changed from -210 to $-(210 + 2^{-23})$. This is a very small absolute change, of magnitude 10^{-7} approximately. Most computers neglect this small change which occurs after 23 binary bits. If the solution of the new equation is now obtained, we find that the smaller roots are obtained with good accuracy, while the roots of larger magnitude are changed by a large amount. The largest change occurs in the roots 16 and 17. They are now obtained as the complex pair $16.73\cdots \pm i\,2.81\cdots$ whose magnitude is 17 approximately. The change is very substantial and it is due to the inherent instability or ill-conditionedness of the polynomial. If the problem cannot be reformulated, then the method that we are using must at least provide some information about the degree of ill-conditioning.

The second type of instability that we encounter is the induced instability which arises usually because of the wrong choice of the method of solution. The problem is often well conditioned in this case. Induced instability can be avoided by a suitable modification or change of the method of solution. For example, to evaluate the integral

$$I_n = \int_0^1 \frac{x^n}{x+6}\,dx, \quad n = 1, 2, \cdots, 10$$

we may use the recurrence relation

$$I_n = \frac{1}{n} - 6\,I_{n-1}, \quad n = 1, 2, \cdots, 10 \tag{1.24}$$

where

$$I_0 = \int_0^1 \frac{dx}{x+6} = \ln(7/6) = 0.15415.$$

Using the recurrence relation (1.24) we obtain

$$\begin{aligned} I_1 &\approx 0.07510, & I_2 &\approx 0.04940 \\ I_3 &\approx 0.03693, & I_4 &\approx 0.02842 \\ I_5 &\approx 0.02948, & I_6 &\approx -0.01021 \\ I_7 &\approx 0.20412, & I_8 &\approx -1.09972 \\ I_9 &\approx 6.70943, & I_{10} &\approx -40.15658\,. \end{aligned}$$

The exact value is $I_{10} = 0.01449$. This explosion has occurred because of the induced instability. This problem is well conditioned and accurate solutions can be obtained by choosing a suitable method.

We may write the recurrence relation (1.24) as

$$I_{n-1} = \frac{1}{6}\left(\frac{1}{n} - I_n\right), \quad n = 10, 9, \cdots, 1. \tag{1.25}$$

Since I_n decreases as n increases, choose $I_{10} = 0$. We obtain

$$I_9 \approx 0.01666, \qquad I_8 \approx 0.01574$$
$$I_7 \approx 0.01821, \qquad I_6 \approx 0.02077$$
$$I_5 \approx 0.02432, \qquad I_4 \approx 0.02928$$
$$I_3 \approx 0.03679, \qquad I_2 \approx 0.04942$$
$$I_1 \approx 0.07510, \qquad I_0 \approx 0.15415.$$

The exact value is $I_0 = 0.15415$.

The criterion when to stop an infinite sequence of computation should be carefully given to the computer. To illustrate this point, consider finding the sum of the series

$$1 - \frac{1}{2} + \frac{1}{3} - \frac{1}{4} + \cdots$$

If the computer is asked to stop when the absolute value of the next term is less than the tolerable error, then nothing would happen and we get accurate solution. However, if the same criterion is applied to the series

$$1 + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \cdots$$

the computer would give a finite solution, while the sum is infinite.

Example 1.9 Find the smaller root of the equation

$$x^2 - 400x + 1 = 0$$

using four digit arithmetic.

For the equation $ax^2 - bx + c = 0$, $b > 0$, the smaller root is given by

$$x = \frac{b - \sqrt{b^2 - 4ac}}{2a}$$

Here

$$a = 1 = 0.1000 \times 10^1$$
$$b = 400 = 0.4000 \times 10^3$$
$$c = 1 = 0.1000 \times 10^1$$
$$b^2 - 4ac = 0.1600 \times 10^6 - 0.4000 \times 10^1$$
$$= 0.1600 \times 10^6 \text{ (to four digit accuracy)}$$

$$\sqrt{b^2-4ac} = 0.4000 \times 10^3.$$

Substituting in the above formula we get $x = 0.0000$.

However, if we write the above formula in the form

$$x = \frac{2c}{b+\sqrt{b^2-4ac}}$$

we get

$$x = \frac{0.2000\times10^1}{0.4000\times10^3+0.4000\times10^3}$$

$$= \frac{0.2000\times10^1}{0.8000\times10^3} = 0.0025$$

which is the exact root.

Example 1.10 Compute the middle value of the numbers

$$a = 4.568 \text{ and } b = 6.762$$

using the four digit arithmetic.

If we take the middle value as the mean of the numbers, we have

$$c = \frac{a+b}{2} = \frac{0.4568\times10^1+0.6762\times10^1}{2}$$

$$= \frac{0.1133\times10^2}{2} = 0.5660 \times 10^1$$

However, if we use the formula

$$c = a + \frac{b-a}{2}$$

we get

$$c = 0.4568 \times 10^1 + \frac{0.6762\times10^1-0.4568\times10^1}{2}$$

$$= 0.4568 \times 10^1 + 0.1097 \times 10^1$$

$$= 0.5665 \times 10^1$$

which is the correct result.

1.4 MACHINE COMPUTATION

To obtain meaningful results for a given problem using computers, there are five distinct phases:

(i) Choice of a method
(ii) Designing the algorithm
(iii) Flow charting
(iv) Programming
(v) Computer execution

2.1 BISECTION METHOD

Bisection is a systematic search technique for finding a zero of a continuous function. The method is based on finding an interval in which a zero is known to occur, dividing the interval into two equal subintervals, and determining which subinterval contains a zero.

Suppose that an interval $[a, b]$ has been located which is known to contain a zero, since the function changes signs between a and b. (See Figure 2.3.) The approximate solution is the midpoint of the interval,

$$m = \frac{a+b}{2}$$

and a zero must lie in either $[a, m]$ or $[m, b]$. The appropriate subinterval is determined by testing the function to see whether it changes sign on $[a, m]$. If so, the search continues on that interval; otherwise it continues on $[m, b]$.

Figure 2.3 illustrates the first two approximations to the zero of the function

$$y = f(x) = x^2 - 3$$

starting with the interval $[1, 2]$. We illustrate this process in the next example.

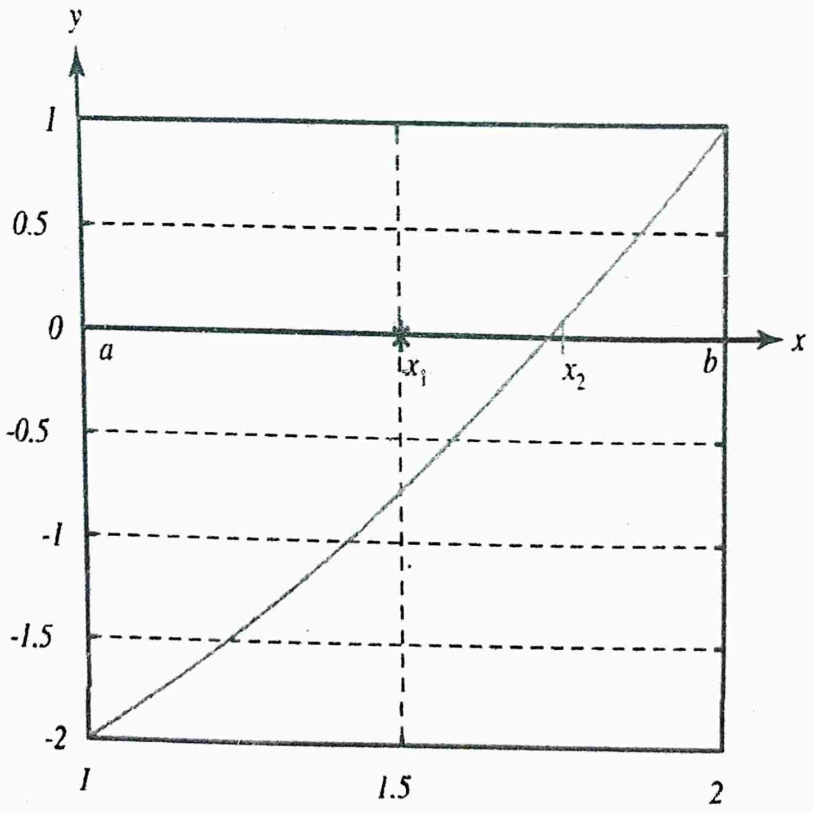


FIGURE 2.3: The graph of $y = x^2 - 3$, and the first two approximations to its zero on [1, 2] using the bisection method.

EXAMPLE 2.1 Finding the Square Root of 3 Using Bisection

To find a numerical approximation to $\sqrt{3}$, we approximate the zero of

$$y = f(x) = x^2 - 3.$$

Since $f(1) = -2$ and $f(2) = 1$, we take as our starting bounds on the zero $a = 1$ and $b = 2$; the corresponding function values are $y_a = f(a) = -2$ and $y_b = f(b) = 1$. Our first approximation to the zero is the midpoint of this interval, namely, $m = (1+2)/2 = 1.5$. We then find the value of the function, $y_m = f(m) = (1.5)^2 - 3 = -0.75$. Since y_a and y_m have the same sign, but y_m and y_b have opposite signs, we know that there is a zero in the interval $[m, b]$.

By setting $a = 1.5$, the improved left-hand endpoint of the interval, we can repeat the process; we now have $y_a = -0.75$. The right-hand end of the interval remains unchanged, so $b = 2$ and $y_b = 1$. It is convenient to keep track of the calculations in a tabular form. The following table shows the results of five iterations through this process. The values of y_a and y_b are shown to emphasize the fact that they have opposite signs at each stage of the process.

step	a	b	m	ya	yb	ym
1	1.0000	2.0000	1.5000	-2.0000	1.0000	-0.7500
2	1.5000	2.0000	1.7500	-0.7500	1.0000	0.0625
3	1.5000	1.7500	1.6250	-0.7500	0.0625	-0.3594
4	1.6250	1.7500	1.6875	-0.3594	0.0625	-0.1523
5	1.6875	1.7500	1.7188	-0.1523	0.0625	-0.0459

As indicated by the values of the function at the approximate zeros, it is possible to be quite close to the true zero at some stage of the iteration and then move away from the zero before returning to a good approximation later (with a smaller error bracket for the zero). In the foregoing calculations, we had a better approximation (i.e., $|y_m|$ was smaller) at step 2 than at step 3 or step 4.

The bisection method provides a bound on the zero at each stage of the iteration; at each stage, the distance between the true zero and the computed zero is no more than half the length of the current interval, i.e., $(b-a)/2$, where a and b are the current values of the ends of the interval.

Since the exact solution to this simple problem is known, we can illustrate the relationship between the actual error in the computed value of the zero, $|m - \sqrt{3}|$, and the bound on the error.

step	a	b	m	y	error	bound
1	1.0000	2.0000	1.5000	-0.7500	0.2321	0.5000
2	1.5000	2.0000	1.7500	0.0625	0.0179	0.1250
3	1.5000	1.7500	1.6250	-0.3594	0.1071	0.0313
4	1.6250	1.7500	1.6875	-0.1523	0.0446	0.0078
5	1.6875	1.7500	1.7188	-0.0459	0.0133	0.0020

The following MATLAB script finds a zero of a function (defined as an inline function in the script) in the interval $[a, b]$ using bisection. Note that $f(a)$ and $f(b)$ must have opposite signs. The script computes the bound on the zero as one-half the current length of the interval. The process stops if either:

1. The magnitude of the function value at the current stage is less than `tol` or
2. The maximum number of iterations `kmax` has been reached.

Bisection

```
f = inline('x.^3-3*x.^2+1')          % bisection method for example 2.2
a = 0;          b = 1;        kmax = 6;        tol = 0.00001;
ya = f(a);      yb = f(b);
if sign(ya) == sign(yb), error('function has same sign at end points'), end
disp('   step       a        b        m        ym       bound')
for k = 1:kmax
    m = (a+b)/2;    ym = f(m);    iter = k;  bound = (b-a)/2;
    out = [ iter, a, b, m, ym,  bound ];  disp( out  )
    if abs(ym)< tol,  disp('bisection has converged'); break;   end
    if sign(ym) ~= sign(ya)
        b = m;          yb = ym;
    else
        a = m;          ya = ym;
    end
    if (iter >= kmax), disp('zero not found to desired tolerance'), end
end
```

EXAMPLE 2.2 Finding the Floating Depth for a Cork Ball by Bisection

To find the floating depth for a cork ball of radius 1 whose density is one-fourth that of water, we must find the zero (between 0 and 1) of $y = f(x) = x^3 - 3x^2 + 1$.

The script above specifies using a maximum of six iterations, or stopping if the magnitude of the function value is less than 0.00001. The zero is not found to the specified tolerance.

step	a	b	m	ym	bound
1	0	1.0000	0.5000	0.3750	0.5000
2	0.5000	1.0000	0.7500	-0.2656	0.2500
3	0.5000	0.7500	0.6250	0.0723	0.1250
4	0.6250	0.7500	0.6875	-0.0930	0.0625
5	0.6250	0.6875	0.6563	-0.0094	0.0313
6	0.6250	0.6563	0.6406	0.0317	0.0156

Analysis

The bisection method is based on a well-known property of continuous functions, the intermediate value theorem. Applied specifically to the cases we are interested in, the theorem states that if $f(a) > 0$ and $f(b) < 0$, or if $f(a) < 0$ and $f(b) > 0$, then there is a number c between a and b such that $f(c) = 0$.

Error Bounds

It is helpful to know how good our answers are for any numerical technique. At each stage of the bisection method, we can determine the maximum possible error in our estimate of the solution.

At the first stage, the length of the interval is $b - a$. The furthest that our estimated zero (the midpoint of the interval) can be from the true solution is $(b - a)/2$. At each stage the length of the interval is halved, so the maximum possible error is reduced by a factor of one-half. If a and b denote the ends of the original interval that brackets the zero, then the error at stage k is at most

$$\frac{b-a}{2^k}$$

Convergence

For an iterative process which produces a sequence of approximations to the true solution of a problem, it is important to know whether (or under what conditions) the sequence converges to the true solution, and if so, how rapidly.

For a convergent sequence, two quantities describe how rapidly the sequence converges, or, in other words, how rapidly the error is reduced. The first describes the "order of convergence." The second describes the "rate of convergence." Convergence is called linear if the order of convergence is one and quadratic if the order is two.

One form of the definition of linear convergence, which is particularly convenient for the bisection method, is that the ratio of the error at the k^{th} stage divided by the error at the $(k-1)^{\text{st}}$ stage approaches a constant value (the convergence rate) as $k \to \infty$. Although we do not know that the actual error is reduced by $1/2$ at each stage of the bisection method, the bound on the error is reduced by $1/2$ at each step, and the limit of the ratio of the error also approaches $1/2$. Thus, bisection is linearly convergent with rate $1/2$.

Computational Considerations

Bisection works well with problems that cause difficulties for other methods. It is also useful as a preprocessing algorithm for the methods we discuss in the remainder of the chapter. In general, bisection is slow, but sure.

2.2 SECANT-TYPE METHODS

Bisection makes no use of information about the shape of the function $y = f(x)$ whose zero is desired. The first way to incorporate such information is to consider a straight-line approximation to the function. Since we know how to find the zero of a linear function, it is not much more work to find our approximation to the zero of $f(x)$ not as the midpoint of the interval, but as the point where the straight-line approximation to $f(x)$ crosses the x-axis. In this section, we consider two closely related methods based on straight-line approximations using two initial values of the independent variable that bracket the desired zero.

The regula falsi and secant methods start with two points, $(a, f(a))$ and $(b, f(b))$ [which we will also denote as (a, y_a) and (b, y_b)], satisfying the condition that $f(a)$ and $f(b)$ have opposite signs. The next approximation to the zero is the value of x where the straight line through the initial points crosses the x-axis; this approximate zero is

$$s = b - \frac{b - a}{y_b - y_a} y_b$$

The regula falsi and secant methods may differ in the choice of the points to be used to define the next iteration.

The geometric basis for this new approximate zero is illustrated in Figure 2.4.

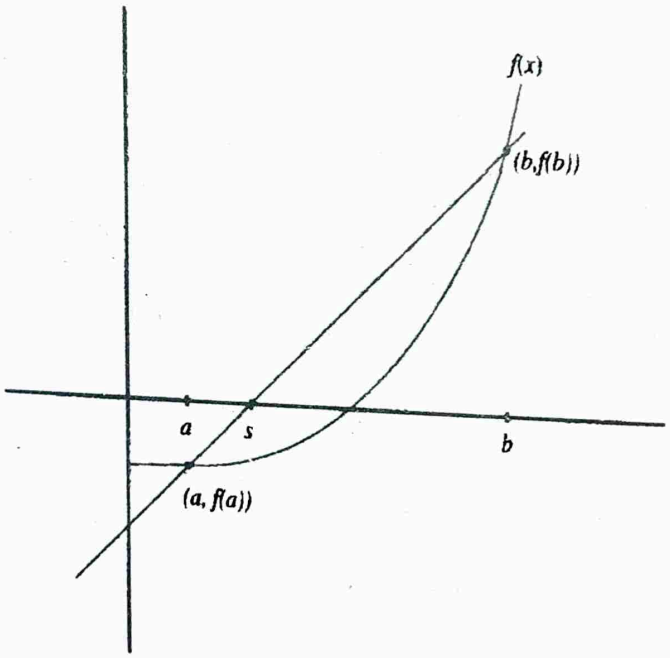


FIGURE 2.4: New approximate zero.

Algebraically, the method is based on solving the equation for the line through the points (a, y_a) and (b, y_b) to find the point $(s, 0)$. The equation of the line may be written as

$$y - y_b = \frac{y_a - y_b}{a - b}(x - b)$$

so we substitute $(s, 0)$:

$$-y_b = \frac{y_a - y_b}{a - b}(s - b)$$

and solve for s to obtain the equation given above.

2.2.1 Regula Falsi

The regula falsi method, or the rule of false position, proceeds as in bisection to find the subinterval $[a, s]$ or $[s, b]$ that contains the zero by testing whether y_a and y have different signs. If there is a zero in the interval $[a, s]$, we leave the value of a unchanged and set $b = s$. On the other hand, if there is no zero in $[a, s]$, the zero must be in the interval $[s, b]$; so we set $a = s$ and leave b unchanged.

The following MATLAB function finds a zero of a function in the interval $[a, b]$ using the regula falsi method; $f(a)$ and $f(b)$ must have opposite signs. The function whose zero is desired is defined as an inline function, as illustrated in the next example. As with the bisection script in the previous section, we choose to display the results at each stage, but not to save all intermediate values.

Regula Falsi

```
function  [s, y] =  Falsi(f, a, b, tol, kmax)
% f is an inline function
ya = f(a);      yb = f(b);
if sign(ya) == sign(yb)
    error('function has same sign at end points')
end
disp('    step        a        b        s        y  ')
for k = 1:kmax
    s = b- yb*(b-a)/(yb-ya);
    y = f(s);
    iter = k;
    out = [ iter, a, b, s, y ];
    disp( out  )
    if abs(y)< tol
        disp('regula falsi has converged'); break;
    end
    if sign(y) ~= sign(ya)
        b = s;
        yb = y;
    else
        a = s;
        ya = y;
    end
    if (iter >= kmax)
        disp('zero not found to desired tolerance')
    end
end
```

EXAMPLE 2.3 Finding the Cube Root of 2 Using Regula Falsi

To find a numerical approximation to $\sqrt[3]{2}$, we seek the zero of $y = f(x) = x^3 - 2$, illustrated, with the first approximation, in Figure 2.5.

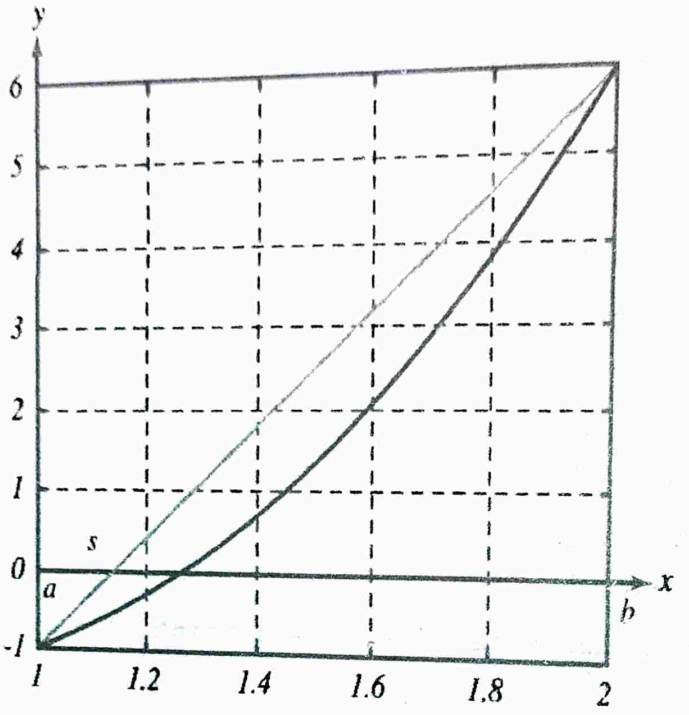


FIGURE 2.5: Graph of $y = x^3 - 2$ and approximation line in the interval [1, 2].

Since $f(1) = -1$ and $f(2) = 6$, we take as our starting bounds on the zero $a = 1$ and $b = 2$; the corresponding function values are $y_a = -1$ and $y_b = 6$. Our first approximation to the zero is

$$x = b - \frac{b-a}{y_b - y_a}\, y(b) = 2 - \frac{2-1}{6+1}(6) = 2 - \frac{6}{7} = \frac{8}{7} \approx 1.1429$$

We then find the value of the function, $y = f(x) = (8/7)^3 - 2 \approx -0.5073$. Since y_a and y are both negative, but y and y_b have opposite signs, we know that there is a zero in the interval [8/7, 2]. By setting $a = 8/7$, we can repeat the process. Now $y_a \approx -0.5073$; the right-hand end of the interval remains unchanged, so $b = 2$ and $y_b = 6$. The results using the MATLAB function `Falsi` are summarized here.

```
[s, y] = Falsi(inline('x^3-2'), 1, 2, 0.0001, 8)
 step      a         b         s          y
    1    1.0000    2.0000    1.1429   -0.5073
    2    1.1429    2.0000    1.2097   -0.2299
    3    1.2097    2.0000    1.2388   -0.0987
    4    1.2388    2.0000    1.2512   -0.0414
    5    1.2512    2.0000    1.2563   -0.0172
    6    1.2563    2.0000    1.2584   -0.0071
    7    1.2584    2.0000    1.2593   -0.0029
    8    1.2593    2.0000    1.2597   -0.0012
zero not found to desired tolerance
```

EXAMPLE 2.4 Floating Depth for a Cork Ball Using False Position

We consider again the problem (introduced in Application 2-A) of finding the floating depth for a cork ball of radius 1 whose density is one-fourth that of water. We must find the zero (between 0 and 1) of $y = f(x) = x^3 - 3x^2 + 1$.

In Example 2.2, we used bisection for this problem. The function and the first two approximate solutions using bisection are shown in Figure 2.6.

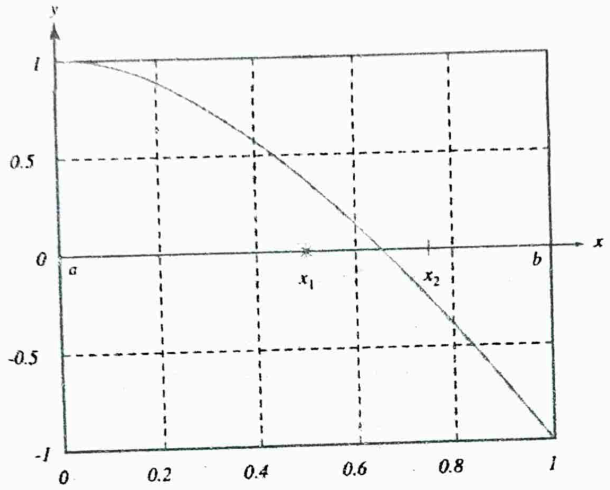


FIGURE 2.6: The graph of $y = x^3 - 3x^2 + 1$
(with approximations by bisection shown for comparison).

The following results from the function `Falsi` show that regula falsi finds the desired zero to the specified tolerance in six iterations.

```
[s, y] =  Falsi_s(inline('x.^3-3*x.^2 + 1'), 0, 1, 0.00001, 8)
    step     a          b          s          y
      1     0.0000     1.0000     0.5000     0.3750
      2     0.5000     1.0000     0.6364     0.0428
      3     0.6364     1.0000     0.6513     0.0037
      4     0.6513     1.0000     0.6526     0.0003
      5     0.6526     1.0000     0.6527     0.0000
      6     0.6527     1.0000     0.6527     0.0000
regula falsi has converged
s =  0.6527              y =  2.1880e-006
```

If we compute the change in the computed zero (which is an estimate of the error at each stage) for the first few iterations, we see that the change is reduced by approximately a factor of 0.1 at each stage. This indicates that the method of false position has linear convergence.

```
Dx(1) = 0.6364 - 0.5000  =  0.1364
Dx(2) = 0.6513 - 0.6364  =  0.0149
Dx(3) = 0.6526 - 0.6513  =  0.0013
Dx(4) = 0.6527 - 0.6526  =  0.0001
```

2.2.2 Secant Method

Although it is appealing to know that the zero is bracketed at each step of the process, as it is in the regula falsi method, there are advantages in simply forming the next approximation from the previous two points. This choice gives the secant method:

$$x_2 = x_1 - \frac{x_1 - x_0}{y_1 - y_0} y_1$$

At the k^{th} stage, the new approximation to the zero is

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

This process does not require any testing to determine the action to take at the next step. It also converges more rapidly than the regula falsi method, in general, even though the zero is not required to lie within the interval at each stage.

EXAMPLE 2.5 Finding the Square Root of 3 by the Secant Method

To find a numerical approximation to $\sqrt{3}$, we seek the zero of $y = f(x) = x^2 - 3$. Since $f(1) = -2$ and $f(2) = 1$, we take as our starting bounds on the zero $x_0 = 1$ and $x_1 = 2$; the corresponding function values are $y_0 = -2$ and $y_1 = 1$. Our first approximation to the zero is

$$\begin{aligned} x_2 &= x_1 - \frac{x_1 - x_0}{y_1 - y_0} y_1 \\ &= 2 - \frac{2 - 1}{1 - (-2)}(1) \\ &= \frac{5}{3} \approx 1.667 \end{aligned}$$

The secant method does not require that the points used to compute the next approximation bracket the zero (or even that $x_0 < x_1$).

It is convenient to keep track of the calculations in a tabular form; the results of four iterations through the process are shown in the following table. The secant method has converged with a tolerance of 10^{-4}.

step	x(k-1)	x(k)	x(k+1)	y(k+1)
1	1.0000	2.0000	1.6667	-0.2222
2	2.0000	1.6667	1.7273	-0.0165
3	1.6667	1.7273	1.7321	0.0003
4	1.7273	1.7321	1.7321	-0.0000

Using the MATLAB function for the secant method (on the next page), we find that it takes only one more iteration to find the $x(k+1) \approx \sqrt{3}$ satisfying the condition that abs$(y(k+1)) < 0.00000001$. The regula falsi method requires eight iterations to achieve the same tolerance.

```
[xx, yy] = Secant(inline('x.^2 - 3'), 1, 2, 0.00000001, 10)
secant method has converged
xx =    1.73205080756550,        yy =    -1.170263885796885e-011
```

The following MATLAB function utilizes the secant method to find a zero of the function f (given as an inline function), using the starting estimates $x(1) = a$ and $x(2) = b$.

In contrast to the bisection and regula falsi methods, $f(a)$ and $f(b)$ need not have opposite signs, and there is no guarantee that there is a zero in the interval between two successive approximations, either initially or at any stage of the iteration.

The difference between successive approximations to the zero, given as `Dx` in the table of output, provides an estimate of the actual error at each stage of the computations.

This function uses the same stopping condition as the function `Falsi` in order to facilitate comparison of the methods. However, a stopping condition based on `Dx` could also be used.

Secant Method

```
function  [xx, yy] =  Secant(f, a, b, tol, kmax)
% f is an inline function
y(1)  = f(a);
y(2)  = f(b);
x(1)  = a;
x(2)  = b;
Dx(1) = 0;
Dx(2) = 0;
disp('    step        x(k-1)       x(k)       x(k+1)       y(k+1)     Dx(k+1)')
for k = 2:kmax
    x(k+1) = x(k) - y(k)*(x(k)-x(k-1))/(y(k)-y(k-1));
    y(k+1) = f(x(k+1));
    Dx(k+1)= x(k+1)-x(k);
    iter = k-1;
    out = [ iter, x(k-1), x(k), x(k+1), y(k+1), Dx(k+1) ];
    disp( out  )
    xx = x(k+1);
    yy = y(k+1);
    if abs(y(k+1))< tol
        disp('secant method has converged'); break;
    end
    if (iter >= kmax)
        disp('zero not found to desired tolerance')
     end
end
```

EXAMPLE 2.6 Central Angle of an Elliptical Orbit

As introduced in Application 2-B, the central angle (at any time t) of an elliptical orbit, with period of revolution $P = 100$ days and eccentricity $e = 0.5$, can be found by finding a zero of the function

$$\begin{aligned} f(\theta, t) &= 2\pi t - P\theta + Pe\sin(\theta) \\ &= 2\pi t - 100\theta + 50\sin(\theta) \end{aligned}$$

for a specified value of t. The graphs of $f(\theta, t)$ for several values of t are shown in Figure 2.7.

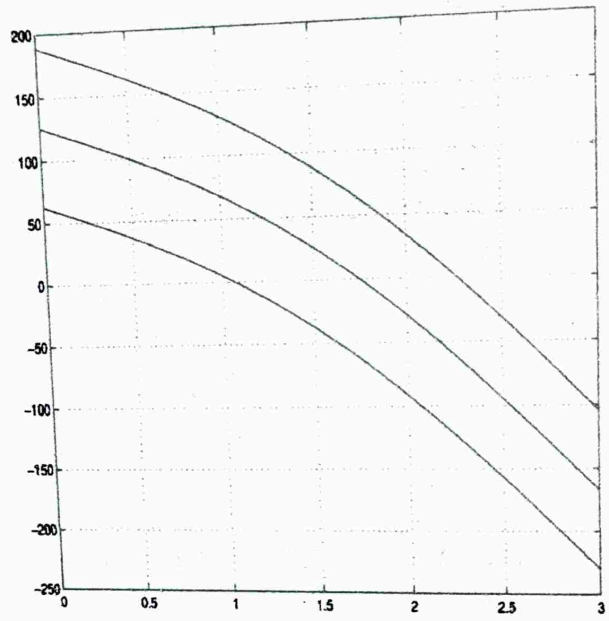


FIGURE 2.7: Graph of $f(\theta, t) = 2\pi t - 50\theta + 25\sin(\theta)$ for $t = 10, 20, 30$.

The calculations to find the central angle θ at $t = 10$:

```
f = inline('20*pi -100*x + 50*sin(x)')
[xx, yy] =  Secant( f, 1, 2, 0.0001, 10)
   step      x(k-1)     x(k)       x(k+1)      y(k+1)      Dx(k+1)
   1         1.0000     2.0000     1.0508      1.1447     -0.9492
   2         2.0000     1.0508     1.0625      0.2622      0.0117
   3         1.0508     1.0625     1.0660     -0.0012      0.0035
   4         1.0625     1.0660     1.0659      0.0000     -0.0000
```

The calculations to find the central angle θ at $t = 20$:.

```
f = inline('40*pi -100*x + 50*sin(x)')
[xx, yy] =  Secant(f, 2, 3, 0.0001, 10)
   step      x(k-1)     x(k)       x(k+1)      y(k+1)      Dx(k+1)
   1         2.0000     3.0000     1.7914     -4.6886     -1.2086
   2         3.0000     1.7914     1.7566     -0.8518     -0.0349
   3         1.7914     1.7566     1.7488     -0.0081     -0.0077
   4         1.7566     1.7488     1.7487     -0.0000     -0.0001
```

2.2.3 Analysis

As introduced in Chapter 1, a method converges with order p if

$$|x_k - x^*| \approx \lambda |x_{k-1} - x^*|^p$$

for large values of k, where the true zero is x^* and $\lambda > 0$ is the asymptotic error constant. If $p = 1$ and $\lambda < 1$, the method is called linearly convergent.

Convergence of Regula Falsi

It is not too hard to see that if a function does not change its concavity on the interval $[a_k, b_k]$, then either the point $(b_k, y(b_k))$ will not change during the regula falsi iterations $(k+1, \ldots)$, or the point $(a_k, y(a_k))$ does not change. From that stage onward, the convergence is linear.

Although it is appealing to know that the zero is bracketed at each step of the process, some effort is required to find the appropriate subinterval at each stage. Also, an iterative process in which the form of the function being iterated can change at each stage (as is the case for a process where a choice must be made as to which subinterval to pursue) is more difficult to analyze.

Convergence of Secant Method

Because the secant method does not bracket the zero at each iteration, it is not guaranteed to converge. However, when it does, the convergence is usually more rapid than for either bisection or regula falsi. Typically, as the iterations progress, the secants become increasingly more accurate approximations to $f(x)$.

The rate of convergence is $p = (1 + \sqrt{5})/2 \approx 1.62$, so convergence is faster than linear, but less than quadratic. The asymptotic error constant is

$$\lambda = \left|\frac{f''(x^*)}{2f'(x^*)}\right|^b, \quad \text{with} \quad b = \frac{\sqrt{5} - 1}{2}$$

Convergence Theorem for the Secant Method

If

1. $f(x), f'(x)$, and $f''(x)$ are continuous on $I = [x^* - e, x^* + e]$
2. $f'(x^*) \neq 0$, and
3. the initial estimates x_0 and x_1 (in I) are sufficiently close to x^*,

then the secant method will converge.

The requirements for "sufficiently close" can be made more precise by defining $M = \dfrac{\max|f''|}{2\min|f'|}$, where max and min are for all x in the interval I. Then if $\max(M|x^* - x_0|, M|x^* - x_1|) < 1$, the secant method will converge. It may converge for starting values that do not satisfy this inequality, but in general, the larger the value of M (if it can be computed), the closer the starting values should be to the zero. (See Atkinson, 1989, pp. 65–73, for a development of these results.)

Comparison

We conclude our discussion of the regula falsi and secant methods by considering the challenging problem of finding the zero of a function that is quite flat near the desired zero.

EXAMPLE 2.7 A Challenging Problem

To illustrate the difficulties that may occur when a function is relatively flat near a zero, consider the simple problem of finding the positive real zero of

$$y = x^5 - 0.5$$

The function and the first three straight-line approximations are illustrated in Figure 2.8.

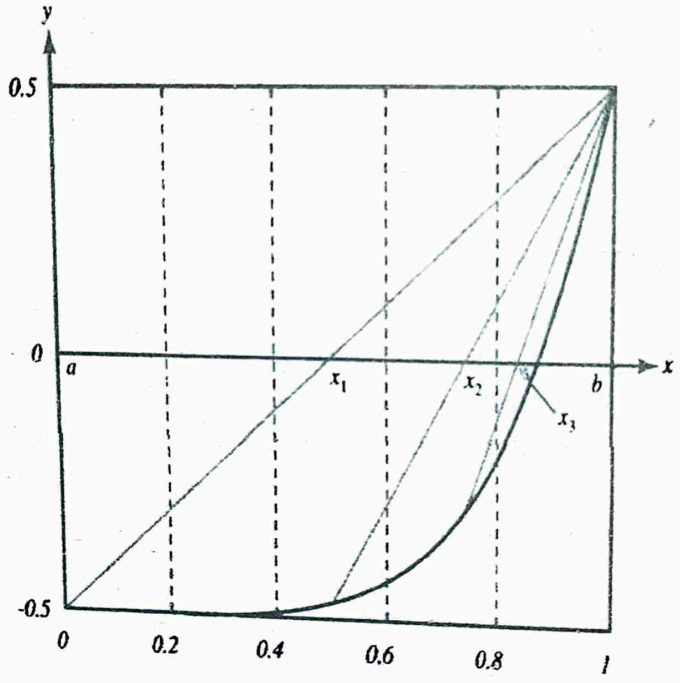


FIGURE 2.8: $y = x^5 - 0.5$ and first three approximations using regula falsi.

The calculations are summarized in the following table. The regula falsi method converges in nine iterations with tol $= 0.0001$.

step	a	b	s	y
1	0	1	0.5	-0.46875
2	0.5	1	0.74194	-0.27518
3	0.74194	1	0.83355	-0.09761
4	0.83355	1	0.86073	-0.027564
5	0.86073	1	0.86801	-0.0072543
6	0.86801	1	0.8699	-0.0018732
7	0.8699	1	0.87038	-0.00048132
8	0.87038	1	0.87051	-0.00012352
9	0.87051	1	0.87054	-3.1687e-005

Now consider the problem of finding the positive real zero of $y = x^5 - 0.5$ using the secant method. (See Figure 2.9.)

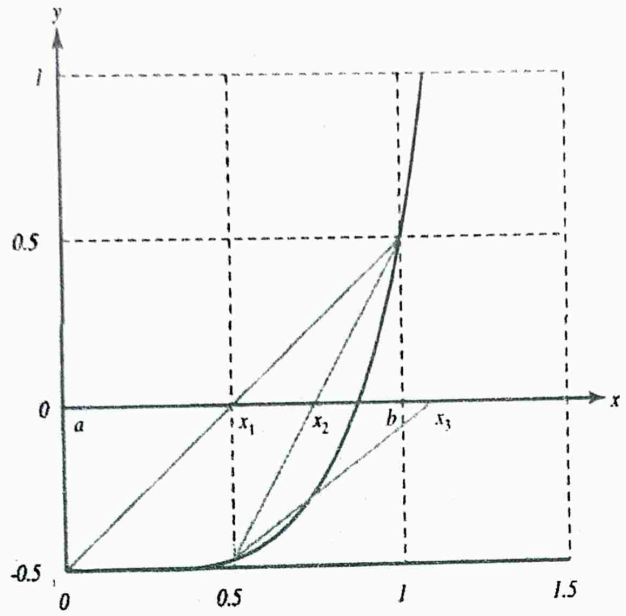


FIGURE 2.9: $y = x^5 - 0.5$ and first three secant approximations.

As shown below, the third approximation is outside the original interval that contains the root. Nevertheless, after eight iterations, the method has achieved essentially the same result as regula falsi.

```
[x, y] = Secant(inline('x.^5 - 0.5'), 0, 1, 0.0001, 10)
    step      x(k-1)     x(k)      x(k+1)     y(k+1)     Dx(k+1)
       1      0.0000    1.0000    0.5000    -0.4688    -0.5000
       2      1.0000    0.5000    0.7419    -0.2752     0.2419
       3      0.5000    0.7419    1.0859     1.0098     0.3439
       4      0.7419    1.0859    0.8156    -0.1391    -0.2703
       5      1.0859    0.8156    0.8483    -0.0607     0.0327
       6      0.8156    0.8483    0.8736     0.0089     0.0253
       7      0.8483    0.8736    0.8704    -0.0005    -0.0032
       8      0.8736    0.8704    0.8705    -0.0000     0.0002
secant method has converged
x =    0.8705
y = -3.2437e-006
```

On the other hand, if we give the starting values as $x_0 = 1$ and $x_1 = 0$, we again find $x_2 = 0.5$, but the calculation of x_3 is based on a straight line with almost zero slope (the line determined by the points $(0, -0.5)$ and $(0.5, -0.46875)$). There are extreme oscillations before the method converges to the correct value. Although it may seem contrived to give the starting estimates in this "nonnatural" order, completely analogous behavior occurs for $y = x^5 + 0.5$ when the starting estimates are $x_0 = -1$ and $x_1 = 0$.

2.3 NEWTON'S METHOD

Like the regula falsi and secant methods, Newton's method uses a straight-line approximation to the function whose zero we wish to find, but in this case the line is the tangent to the curve. The next approximation to the zero is the value of x where the tangent crosses the x-axis. This requires additional information about the function (i.e., its derivative). The secant method discussed in the previous section can be viewed as Newton's method with the derivative approximated by a difference quotient.

Given an initial estimate of the zero, x_0; the value of the function at x_0, $y_0 = f(x_0)$; and the value of the derivative at x_0, $y_0' = f'(x_0)$; the x intercept of the tangent line, which is the new approximation to the zero, is

$$x_1 = x_0 - \frac{y_0}{y_0'}.$$

The process continues until the change in the approximations is sufficiently small or some other stopping condition is satisfied. At the k^{th} stage, we have

$$x_{k+1} = x_k - \frac{y_k}{y_k'}$$

The geometric basis for Newton's method is illustrated in Figure 2.10.

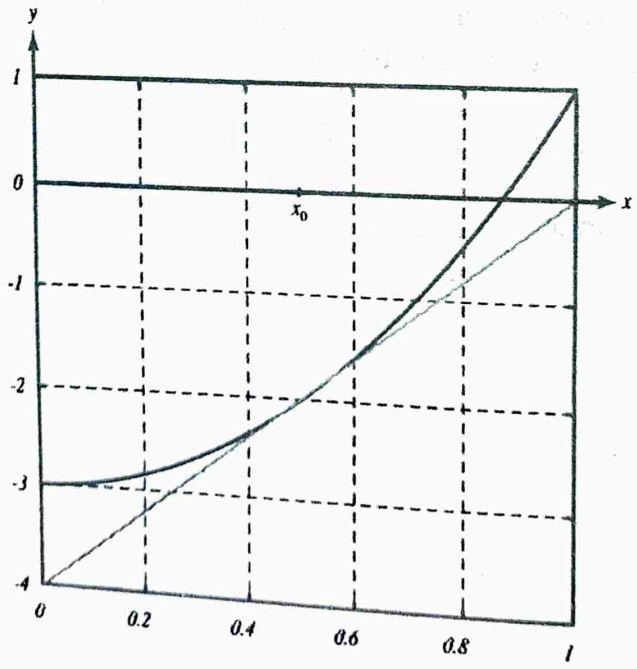


FIGURE 2.10: The graph of $f(x) = 4x^2 - 3$ and the tangent line at $x = 0.5$.

Algebraically, we can write the equation of the tangent line at (x_0, y_0) as

$$y - y_0 = f'(x_0)(x - x_0)$$

The next approximation to the zero of $f(x)$ is the value of x where the tangent crosses the x-axis; substitute the point $(x_1, 0)$ into this equation, and solve for x_1.

EXAMPLE 2.8 Finding the Square Root of 3/4 Using Newton's Method

To find a numerical approximation to $\sqrt{3/4}$, we approximate the zero of

$$y = f(x) = 4x^2 - 3$$

(see Figure 2.11) using the fact that $y' = f'(x) = 8x$.

Since $f(0) = -3$ and $f(1) = 1$, we take as our starting estimate of the zero $x_0 = 0.5$; the corresponding function value is $y_0 = -2$, and the derivative value is $y_0' = 4$. Our first approximation to the zero is

$$x_1 = x_0 - \frac{y_0}{y_0'} = 0.5 - \frac{-2}{4} = 1$$

Continuing for one more step yields

$$x_2 = x_1 - \frac{y_1}{y_1'} = 1.0 - \frac{1}{8} = \frac{7}{8} = 0.875$$

For this simple example, we can write out the iteration formula explicitly, and simplify it if desired, to give

$$x_{k+1} = x_k - \frac{4x_k^2 - 3}{8x_k} = x_k - \frac{4x_k^2}{8x_k} - \frac{-3}{8x_k}$$
$$= \frac{1}{2}x_k + \frac{3}{8x_k} = \frac{1}{2}(x_k + \frac{3/4}{x_k})$$

Thus we see that we have the same iterations as for the "Babylonian method" in Chapter 1. This is also sometimes known as "Heron's method."

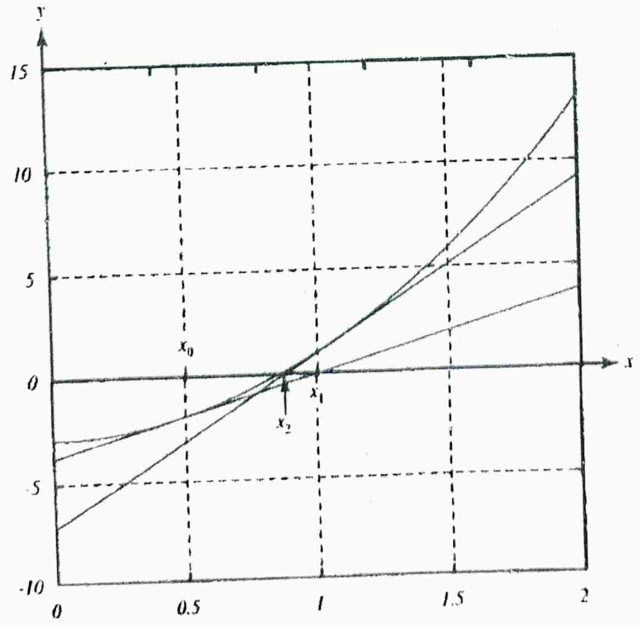


FIGURE 2.11: The graph of $y = 4x^2 - 3$ and the tangent-line approximation at $x_0 = 0.5$ and $x_1 = 1$.

The following MATLAB function finds a zero of a function near the initial estimate x_1 using Newton's method. The process stops if either:

1. The change in successive iterates (which is also the estimate of the error) is less than `tol` or
2. The maximum number of iterations, `kmax`, has been reached.

Newton's Method

```
function  [x, y] = Newton(fun, funpr, x1, tol, kmax)
% Input:
%      fun        function (inline function or m-file function)
%      funpr      derivative function (inline or m-file)
%      x1         starting estimate
%      tol        allowable tolerance in computed zero
%      kmax       maximum number of iterations
% Output:
%      x          (row) vector of approximations to zero
%      y          (row) vector fun(x)
x(1) = x1;
y(1) = feval(fun, x(1));
ypr(1) = feval(funpr, x(1));
for k = 2 : kmax
   x(k) = x(k-1)-y(k-1)/ypr(k-1);
   y(k) = feval(fun, x(k));
   if abs(x(k)-x(k-1)) < tol
        disp('Newton method has converged');  break;
   end
   ypr(k) = feval(funpr, x(k));
   iter = k;
end
if (iter >= kmax)
   disp('zero not found to desired tolerance');
end
n = length(x);
k = 1:n;
out = [k'  x'  y'];
disp('          step                x                  y')
disp(out)
```

EXAMPLE 2.9 Finding the Floating Depth for a Wooden Ball

To find the depth at which a ball of radius 1 whose density is one-third that of water floats, we must find the zero (between 0 and 1) of $y = x^3 - 3x^2 + 4/3$.

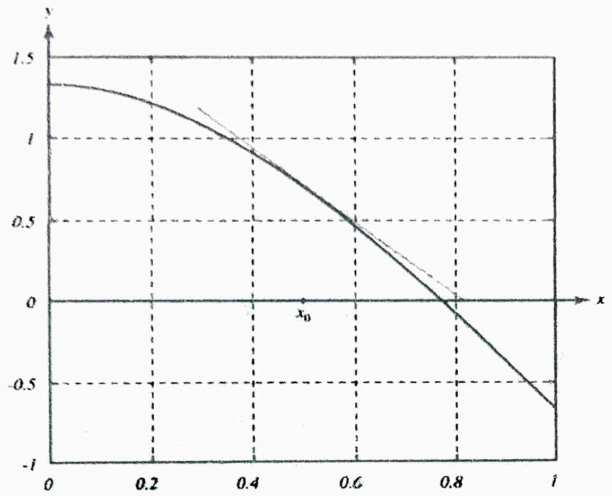


FIGURE 2.12: Floating depth of wooden ball.

The results are summarized in the following table.

step	x	y
1	0.5	0.70833
2	0.81481	−0.11746
3	0.77427	−0.00097989
4	0.77393	-8.0255×10^{-08}
5	0.77393	-4.4409×10^{-16}

The function `Newton` given on the previous page can accept the function whose zero is desired defined as an inline function; for this example we would have

```
f = inline('x.^3 -3*x.^2 +4/3');
df = inline('3*x.^2-6*x ');
[x, y] = Newton(f, df, 0.5, 0.00001, 10);
```

On the other hand, if m-file functions were used for $f(x)$ and $df(x)$, we would define the function (and save it in a file named `f_2_9.m`)

```
function y = f_2_9(x)
y = x.^3 -3*x.^2.+4/3;
```

and, similarly, the function for the derivative (saved in the file `df_2_9.m`)

```
function dy = df_2_9(x)
dy = 3*x.^2-6*x;
```

The function `Newton` would then be called as

```
[x, y] = Newton('f_2_9', 'df_2_9', 0.5, 0.00001, 10);
```

(Note the single quotes around the names of the functions `f_2_9.m` and `df_2_9.m`).

Discussion

We can obtain more information about the approximation to the zero x^* if we consider the Taylor series expansion for $f(x)$ near x_k, i.e.,

$$f(x) = f(x_k) + (x - x_k)f'(x_k) + 0.5(x - x_k)^2 f''(\eta)$$

where η is some unknown point between x and x_k. If $x = x^*$ and $f(x^*) = 0$, then

$$0 = f(x_k) + (x^* - x_k)f'(x_k) + 0.5(x^* - x_k)^2 f''(\eta)$$

and

$$x^* = x_k - \frac{f(x_k)}{f'(x_k)} - 0.5(x^* - x_k)^2 \frac{f''(\eta)}{f'(x_k)}$$

Setting $x_{k+1} = x_k - f(x_k)/f'(x_k)$ and substituting this into the previous equation, we find the error at the $(k+1)^{\text{st}}$ step:

$$x^* - x_{k+1} = -0.5(x^* - x_k)^2 \frac{f''(\eta)}{f'(x_k)} = -(x^* - x_k)^2 \frac{f''(\eta)}{2f'(x_k)}$$

Thus, if x^* is a simple zero (i.e., $f'(x^*) \neq 0$), Newton's method is quadratically convergent with $\lambda = |\dfrac{f''(x*)}{2f'(x*)}|$.

If $f'(x)$ is not changing rapidly near the true zero, so that $f'(x^*) \approx f'(x_k)$, then the true error is given approximately by the change in the most recent iterates:

$$x^* - x_k \approx x_{k+1} - x_k$$

To obtain some information about how close the initial estimate x_0 must be to the actual zero x^*, let I be an interval around x^* such that $f'(x) \neq 0$ on I; then the error at the first stage is

$$|x^* - x_1| = (x^* - x_0)^2 \frac{|f''(\eta)|}{2|f'(x_k)|} \leq M(x^* - x_0)^2$$

where $M = \dfrac{\max|f''(x)|}{2\min|f'(x)|}$ (with the max and min taken over all $x \in I$). If the initial estimate is chosen close enough to the true zero so that $|x^* - x_0| < 1/M$, the error at each stage will be less than or equal to the error at the first stage.

Convergence Theorem for Newton's Method
If

1. $f(x)$, $f'(x)$, and $f''(x)$ are continuous for all x in a neighborhood of x^*,
2. $f'(x*) \neq 0$, and
3. x_0 is chosen sufficiently close to x^*,

then the iterates $x_{k+1} = x_k - f(x_k)/f'(x_k)$ will converge to x^*. Furthermore,

$$\lim_{k \to \infty} \frac{x^* - x_k}{(x^* - x_{k-1})^2} = \frac{f''(x^*)}{2f'(x^*)}$$

Caveats

Newton's method has a high order of convergence and a fairly simple statement; hence, it is often the first method people use. However, the method can encounter difficulties, as illustrated in the next example.

Furthermore, the derivative must not be zero at any approximation to the zero, or Newton's method will fail.

The stopping condition should be a combination of a specified maximum number of iterations and minimum tolerance on the change in the computed zero. Because of the potential for divergence, it is also wise to test for large changes in the value of the computed zero, which might signal difficulties.

EXAMPLE 2.10 Oscillations in Newton's Method

Newton's method can give oscillatory results for some functions and some initial estimates. For example, consider the cubic equation

$$y = x^3 - 3x^2 + x + 3$$

If we don't consider the graph, we might guess $x_0 = 1$ as the initial point, as shown in the following script for using the function `Newton` given previously.

```
f = inline('x.^3 -3*x.^2 + x + 3')
df = inline('3*x.^2 - 6*x + 1')
[x, y] = Newton(f, df, 1, 0.001, 3)
```

The calculations for the first three iterations are summarized in the following table.

step	x_{i-1}	x_i	y_{i-1}	y_i
1	1	2	2	1
2	2	1	1	2
3	1	2	2	1

The tangent approximations for the first two iterations are illustrated in Figure 2.13. It is easy to see that the process will oscillate between these two values.

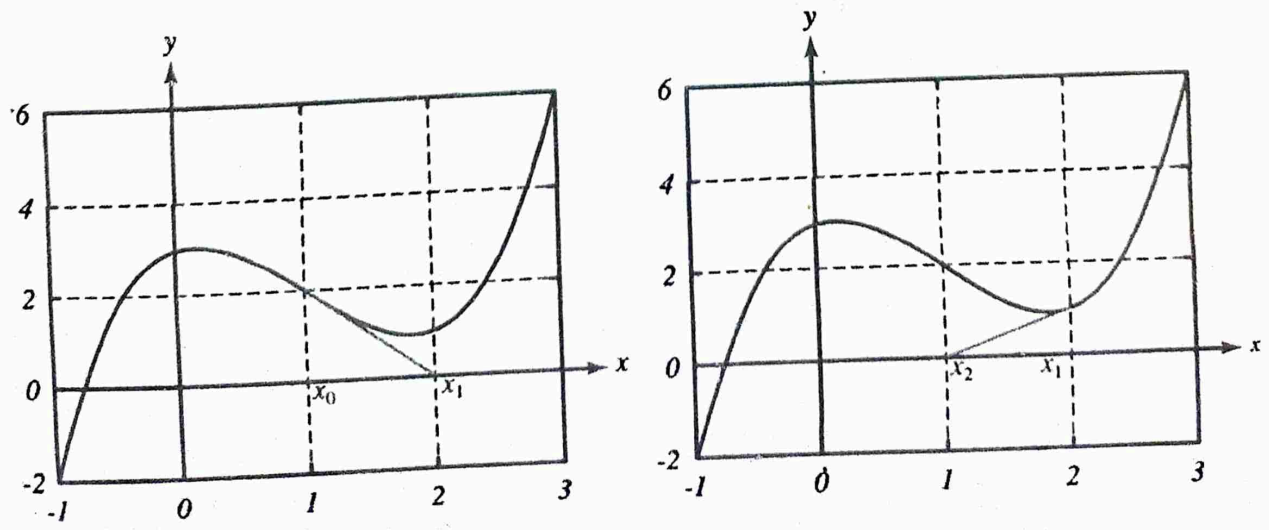


FIGURE 2.13: Oscillatory behavior of Newton's method.

Repeated Roots

Newton's method loses its quadratic convergence in the case of a repeated root. However, it is possible to modify the method to preserve the rapid convergence if the order of the repeated root is known (or can be estimated).

If we write Newton's method as $x = g(x) = x - \dfrac{f(x)}{f'(x)}$ we see that $g'(x) = \dfrac{f(x)f''(x)}{(f'(x))^2}$. Thus, at the zero x^*, (if $f'(x^*) \neq 0$), we have $g'(x^*) = 0$.

On the other hand, for a double root, $f(x) = (x - x^*)^2 h(x)$ and some simple calculations show that $g'(x^*) = 1 - 1/2 \neq 0$. We can modify the method to use $x = g(x) = x - 2\dfrac{f(x)}{f'(x)}$, or, in general for an m^{th}-order root, $x = g(x) = x - m\dfrac{f(x)}{f'(x)}$. A MATLAB function for a double root is given here. The functions f and df must be given as inline functions. The parameter C should be approximately constant for quadratic convergence.

Newton's Method for Double Root

```
function [x, y] = Newton_d(f,  df, x0,  tol, kmax)
x(1) = x0;   y(1) = f(x(1));   dy(1) = df(x(1));   C(1) = 0;
disp('     step     x(k)     y(k)    Dx    C')
for k = 1 : kmax
    x(k+1) = x(k) - 2*y(k) /dy(k);     y(k+1) = f(x(k+1));
    dy(k+1) = df(x(k+1));              Dx(k) = x(k+1) - x(k);
    if k > 1,  C(k) = Dx(k)/Dx(k-1)^2;   end
    if (abs(Dx(k)) < tol),   disp('method has converged');  break; end
    iter = k-1;   out = [ iter   x(k)   y(k)    Dx(k)    C(k) ];   disp(out)
    if (iter >= kmax+1),   disp('zero not found to desired tolerance'), end
end
```

EXAMPLE 2.11 Newton's Method for a Double Root

To illustrate the use of the function above, consider the root of

$$f(x) = (x^2 - 5)^2(x^2 - 3)$$

which clearly has a double root near $x_0 = 2$.

$$df(x) = 4x(x^2 - 5)(x^2 - 3) + 2x(x^2 - 5)^2$$

```
[x, y] = Newton_d(f,  df, 2,  0.0001, 10)
step     x(i)          y(i)           Dx              C
 0       2             1
 1       2.5           5.0781         0.5 0
 2       2.2903        0.13543        -0.20968        -0.83871
 3       2.2395        0.00047275     -0.050832       -1.1562
                                      -0.0034066      -1.3184
```

From the table we find that the equation $f(x) = 0$ has at least one root in the interval (0.5, 1). The exact root correct to ten decimal places is 0.5177573637.

2.2 BISECTION METHOD

This method is based on the repeated application of the intermediate value theorem. If we know that a root of $f(x) = 0$ lies in the interval $I_0 = (a_0, b_0)$, we bisect I_0 at the point $m_1 = (a_0 + b_0)/2$. Denote by I_1 the interval (a_0, m_1) if $f(a_0)\, f(m_1) < 0$ or the interval (m_1, b_0) if $f(m_1)\, f(b_0) < 0$. Therefore, the interval I_1 also contains the root. We bisect the interval I_1 and get a subinterval I_2 at whose end points $f(x)$ takes the values of opposite signs and therefore contains the root. Continuing this procedure, we obtain a sequence of nested sets of sub-intervals $I_0 \supset I_1 \supset I_2 \cdots$ such that each subinterval contains the root. After repeating the bisection process q times, we either find the root or find the interval I_q of length $(b_0 - a_0)/2^q$ which contains the root. We take the midpoint of the last subinterval as the desired approximation to the root. This root has error not greater than one-half of the length of the interval of which it is the midpoint. Thus, we have

$$m_{k+1} = a_k + \frac{1}{2}(b_k - a_k), \quad k = 0, 1, 2, \cdots$$

where

$$(a_{k+1}, b_{k+1}) = \begin{cases} (a_k, m_{k+1}), \text{ if } f(a_k)\, f(m_{k+1}) < 0 \\ (m_{k+1}, b_k), \text{ if } f(m_{k+1})\, f(b_k) < 0. \end{cases}$$

We notice that this method uses only the end points of the interval $[a_k, b_k]$ for which $f(a_k)\, f(b_k) < 0$ and not the values of $f(x)$ at these end points, to obtain the next approximation to the root. The method is simple to use and the sequence of approximations always converges to the root for any $f(x)$ which is continuous in the interval that contains the root. If the permissible error is ε, then the approximate number of iterations required may be determined from the relation

$$\frac{b_0 - a_0}{2^n} \leq \varepsilon \quad \text{or} \quad n \geq \frac{\log (b_0 - a_0) - \log \varepsilon}{\log 2}$$

Since n is an integer, we take n as the next nearest integer.

The minimum number of iterations required for converging to a root in the interval (0, 1) for a given ε are listed in Table 2.3.

Table 2.3 Number of Iterations

ε	10^{-2}	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}
n	7	10	14	17	20	24

Thus, the bisection method requires a large number of iterations to achieve a reasonable degree of accuracy for the root. It requires one function evaluation for each iteration.

Example 2.3 Perform five iterations of the bisection method to obtain the smallest positive root of the equation

$$f(x) = x^3 - 5x + 1 = 0.$$

Since $f(0) > 0$ and $f(1) < 0$, the smallest positive root lies in the interval (0, 1). Taking $a_0 = 0$, $b_0 = 1$, we get

$$m_1 = \frac{1}{2}(a_0 + b_0) = \frac{1}{2}(0 + 1) = 0.5$$

$$f(m_1) = -1.375 \text{ and } f(a_0)\, f(m_1) < 0.$$

Thus, the root lies in the interval (0, 0.5). Taking $a_1 = 0$, $b_1 = 0.5$, we get

$$m_2 = \frac{1}{2}(a_1 + b_1) = \frac{1}{2}(0 + 0.5) = 0.25$$

$$f(m_2) = f(0.25) = -0.234375 \text{ and } f(a_1)\, f(m_2) < 0.$$

Thus the root lies in the interval (0, 0.25). The sequence of intervals is given in Table 2.4.

Table 2.4 Sequence of Intervals for the Bisection Method

k	a_{k-1}	b_{k-1}	m_k	$f(m_k)\, f(a_{k-1})$
1	0	1	0.5	< 0
2	0	0.5	0.25	< 0
3	0	0.25	0.125	> 0
4	0.125	0.25	0.1875	> 0
5	0.1875	0.25	0.21875	< 0

Hence, the root lies in (0.1875, 0.21875). The approximate root is taken as the midpoint of this interval, that is 0.203125.

Example 2.4 Perform five iterations of the bisection method to obtain a root of the equation

$$f(x) = \cos x - xe^x = 0$$

Since $f(0) = 1 > 0$ and $f(1) = -2.1780 < 0$, the root lies in the interval (0, 1). Taking the initial approximations as $a_0 = 0$, $b_0 = 1$, we get

$$m_1 = \frac{1}{2}(a_0 + b_0) = \frac{1}{2}(0 + 1) = 0.5$$

$$f(m_1) = f(0.5) = 0.0532 \text{ and } f(a_0)\, f(m_1) > 0.$$

Therefore, the root lies in the interval (0.5, 1.0).

Taking $a_1 = 0.5$, $b_1 = 1.0$, we get

$$m_2 = \frac{1}{2}(a_1 + b_1) = \frac{1}{2}(0.5 + 1.0) = 0.75$$

$$f(m_2) = -0.8561 \text{ and } f(a_1)\, f(m_2) < 0.$$

Therefore, the root lies in the interval (0.5, 0.75). The sequence of intervals is given in Table 2.5.

Table 2.5 Sequence of Intervals for the Bisection Method

k	a_{k-1}	b_{k-1}	m_k	$f(m_k)\, f(a_{k-1})$
1	0	1	0.5	> 0
2	0.5	1	0.75	< 0
3	0.5	0.75	0.625	< 0
4	0.5	0.625	0.5625	< 0
5	0.5	0.5625	0.53125	< 0

Hence, the root lies in the interval (0.5, 0.53125). The approximate root is taken as the midpoint of this interval, that is, 0.515625.

2.3 ITERATION METHODS BASED ON FIRST DEGREE EQUATION

We have already seen that if $f(x) = 0$ is a first degree equation in x then it can be readily solved. We now study the iteration methods which will produce exact results whenever $f(x) = 0$ is a first degree equation. Thus, if we approximate $f(x)$ by a first degree equation in the neighbourhood of the root, then we may write

$$f(x) = a_0 x + a_1 = 0. \tag{2.8}$$

The solution of (2.8) is given by

$$x = -\frac{a_1}{a_0} \tag{2.9}$$

where $a_0 \neq 0$ and a_1 are arbitrary parameters to be determined by prescribing two appropriate conditions on $f(x)$ and/ or its derivatives.

Secant and Regula-Falsi Methods

If x_{k-1} and x_k are two approximations to the root, then we determine a_0 and a_1 in (2.8) by using the conditions

$$f_{k-1} = a_0 x_{k-1} + a_1$$
$$f_k = a_0 x_k + a_1$$

where $f_{k-1} = f(x_{k-1})$ and $f_k = f(x_k)$.

On solving, we obtain

$$a_0 = (f_k - f_{k-1})/(x_k - x_{k-1})$$
$$a_1 = (x_k f_{k-1} - x_{k-1} f_k)/(x_k - x_{k-1}). \tag{2.10}$$

From the equations (2.9) and (2.10), the next approximation x_{k+1} to the root is given by

$$x_{k+1} = \frac{x_{k-1} f_k - x_k f_{k-1}}{f_k - f_{k-1}} \tag{2.11}$$

which may also be written as

$$x_{k+1} = x_k - \frac{x_k - x_{k-1}}{f_k - f_{k-1}} f_k, \quad k = 1, 2, \ldots \tag{2.12}$$

This is called the **secant** or the **chord method**.

Geometrically, in this method we replace the function $f(x)$ by a straight line or a chord passing through the points (x_k, f_k) and (x_{k-1}, f_{k-1}) and take the point of intersection of the straight line with the x-axis as the next approximation to the root (Fig. 2.2a). If the approximations are such that $f_k f_{k-1} < 0$, then the method (2.11) or (2.12) is known as the **Regula-Falsi method**. The method is shown graphically in Fig. 2.2b. Since (x_{k-1}, f_{k-1}), (x_k, f_k) are known before the start of the iteration, the secant and the Regular-Falsi methods require one function evaluation per iteration.

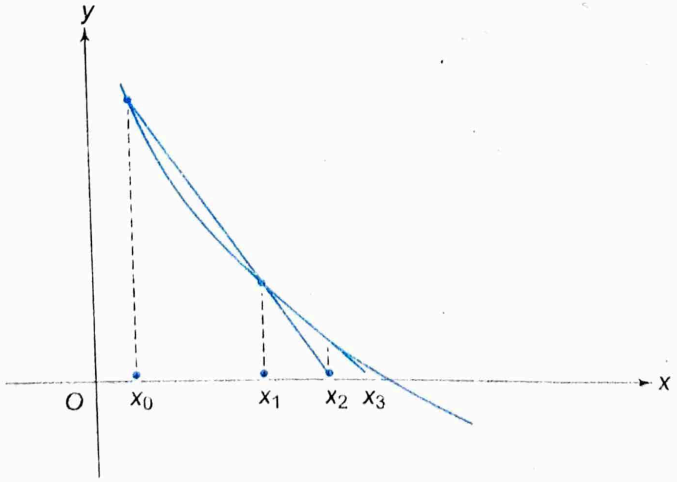


Fig. 2.2 (a). Secant method.

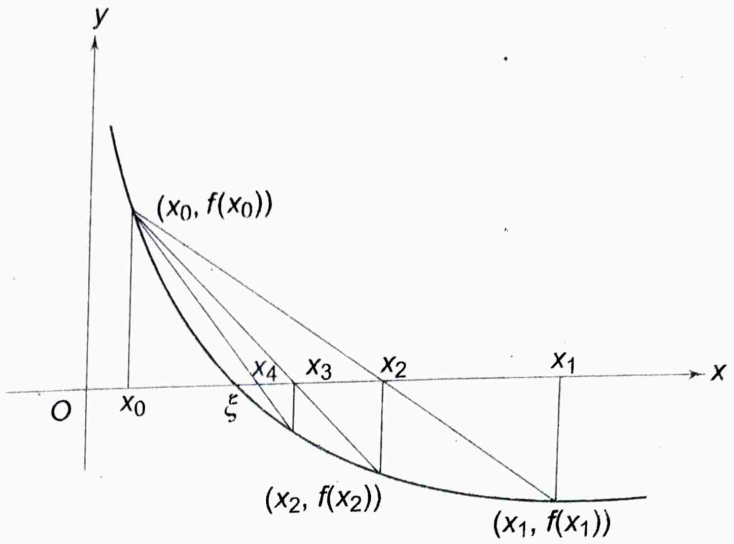


Fig. 2.2 (b). The Regula-Falsi method.

Example 2.5 A real root of the equation

$$f(x) = x^3 - 5x + 1 = 0$$

lies in the interval (0, 1). Perform four iterations of the secant method and the Regula-Falsi met to obtain this root.

We have $x_0 = 0,\ x_1 = 1,\ f_0 = f(x_0) = 1,\ f_1 = f(x_1) = -3.$

Secant method

$$x_2 = x_1 - \left[\frac{x_1 - x_0}{f_1 - f_0}\right] f_1 = 0.25,\ f_2 = f(x_2) = -0.234375.$$

$$x_3 = x_2 - \left[\frac{x_2 - x_1}{f_2 - f_1}\right] f_2 = 0.186441,\ f_3 = f(x_3) = 0.074276.$$

$$x_4 = x_3 - \left[\frac{x_3 - x_2}{f_3 - f_2}\right] f_3 = 0.201736,\ f_4 = f(x_4) = -0.000470.$$

$$x_5 = x_4 - \left[\frac{x_4 - x_3}{f_4 - f_3}\right] f_4 = 0.201640.$$

Regula-Falsi method

$$x_2 = x_1 - \left[\frac{x_1 - x_0}{f_1 - f_0}\right] f_1 = 0.25,\ f_2 = f(x_2) = -0.234375.$$

Since $f(x_0)\ f(x_2) < 0,\ \xi \in (x_0, x_2)$. Therefore,

$$x_3 = x_2 - \left[\frac{x_2 - x_0}{f_2 - f_0}\right] f_2 = 0.202532,\ f_3 = f(x_3) = -0.004352.$$

Since $f(x_0)\, f(x_3) < 0$, $\xi \in (x_0, x_3)$. Therefore,

$$x_4 = x_3 - \left[\frac{x_3 - x_0}{f_3 - f_0}\right] f_3 = 0.201654,\ f_4 = f(x_4) = -0.000070.$$

Since $f(x_0)\, f(x_4) < 0$, $\xi \in (x_0, x_4)$. Therefore,

$$x_5 = x_4 - \left[\frac{x_4 - x_0}{f_4 - f_0}\right] f_4 = 0.201640.$$

Example 2.6 Use the secant and Regula-Falsi methods to determine the root of the equation

$$\cos x - x\, e^x = 0.$$

Taking the initial approximations as $x_0 = 0$, $x_1 = 1$, we obtain for the secant method

$$f(0) = 1,$$

$$f(1) = \cos 1 - e = -2.177979523$$

$$x_2 = x_1 - \left[\frac{x_1 - x_0}{f_1 - f_0}\right] f_1 = 0.3146653378$$

$$f_2 = f(x_2) = 0.519871175$$

$$x_3 = x_2 - \left[\frac{x_2 - x_1}{f_2 - f_1}\right] f_2 = 0.4467281466$$

$$f_3 = f(x_3) = 0.203544710$$

$$x_4 = x_3 - \left[\frac{x_3 - x_2}{f_3 - f_2}\right] f_3 = 0.5317058606.$$

Now, for the Regula-Falsi method, we get

$$x_2 = x_1 - \left[\frac{x_1 - x_0}{f_1 - f_0}\right] f_1 = 0.3146653378,$$

$$f_2 = f(x_2) = 0.519871175.$$

Since $f(x_1)\, f(x_2) < 0$, $\xi \in (x_1, x_2)$. Therefore,

$$x_3 = x_2 - \left[\frac{x_2 - x_1}{f_2 - f_1}\right] f_2 = 0.4467281466$$

$$f_3 = f(x_3) = 0.203544710.$$

Since $f(x_1)\, f(x_3) < 0$, $\xi \in (x_1, x_3)$. Therefore,

$$x_4 = x_3 - \left[\frac{x_3 - x_1}{f_3 - f_1}\right] f_3 = 0.4940153366.$$

The computed results are tabulated in Table 2.6.

Table 2.6 Approximations to the Root by the Secant and the Regula-Falsi Methods

k	*Secant Method*		*Regula-Falsi Method*	
	x_{k+1}	$f(x_{k+1})$	x_{k+1}	$f(x_{k+1})$
1	0.3146653378	0.519871	0.3146653378	0.519871
2	0.4467281466	0.203545	0.4467281446	0.203545
3	0.5317058606	−0.429311(− 01)	0.4940153366	0.708023(− 01)
4	0.5169044676	0.259276(− 02)	0.5099461404	0.236077(− 01)
5	0.5177474653	0.301119(− 04)	0.5152010099	0.776011(− 02)
6	0.5177573708	−0.215132(− 07)	0.5169222100	0.253886(− 02)
7	0.5177573637	0.178663(− 12)	0.5174846768	0.829358(− 03)
8	0.5177573637	0.222045(− 15)	0.5176683450	0.270786(− 03)
10	—	—	0.5177478783	0.288554(− 04)
20	—	—	0.5177573636	0.396288(− 09)

The numbers within the parentheses denote exponentiation.

Newton-Raphson Method

We determine a_0 and a_1 in (2.8) using the conditions

$$f_k = a_0 x_k + a_1$$
$$f'_k = a_0 \tag{2.13}$$

where a prime denotes differentiation with respect to x.

On substituting a_0 and a_1 from (2.13) in (2.9) and representing the approximate value of x by x_{k+1}, we obtain

$$x_{k+1} = x_k - \frac{f_k}{f'_k}, \quad k = 0, 1, \cdots \tag{2.14}$$

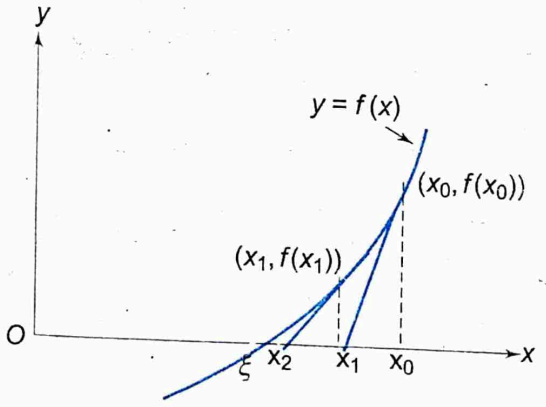


Fig. 2.3. The Newton-Raphson method.

This method is called the **Newton-Raphson** method. The method (2.14) may also be obtained directly from (2.12) by taking the limit $x_{k-1} \to x_k$. In the limit when $x_{k-1} \to x_k$, the chord passing through the

points (x_k, f_k) and (x_{k-1}, f_{k-1}) becomes the tangent at the point (x_k, f_k). Thus, in this case the problem of finding the root of the equation (2.1) is equivalent to finding the point of intersection of the tangent to the curve $y = f(x)$ at the point (x_k, f_k) with the x-axis. The method is shown graphically in Fig. 2.3. The Newton-Raphson method requires two evaluations f_k, f'_k for each iteration.

Alternative

Let x_k be an approximation to the root of the equation $f(x) = 0$. Let Δx be an increment in x such that $x_k + \Delta x$ is an exact root. Therefore,

$$f(x_k + \Delta x) \equiv 0.$$

Expanding in Taylor series about the point x_k, we get

$$f(x_k) + \Delta x\, f'(x_k) + \frac{1}{2!}(\Delta x)^2 f''(x_k) + \cdots = 0.$$

Neglecting the second and higher powers of Δx, we obtain

$$f(x_k) + \Delta x\, f'(x_k) \approx 0$$

or

$$\Delta x \approx -\frac{f(x_k)}{f'(x_k)}.$$

Hence, we obtain the iteration method

$$x_{k+1} = x_k + \Delta x = x_k - \frac{f(x_k)}{f'(x_k)}, \quad k = 0, 1, \cdots$$

which is same as (2.14).

Example 2.7 Perform four iterations of the Newton-Raphson method to find the smallest positive root of the equation

$$f(x) = x^3 - 5x + 1 = 0.$$

The smallest positive root lies in the interval (0, 1). Take the initial approximation as $x_0 = 0.5$. We have

$$f(x) = x^3 - 5x + 1, \; f'(x) = 3x^2 - 5.$$

Using the Newton-Raphson method

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}$$

we get

$$x_{k+1} = x_k - \frac{x_k^3 - 5x_k + 1}{3x_k^2 - 5} = \frac{2x_k^3 - 1}{3x_k^2 - 5}, \quad k = 0, 1, \cdots$$

Starting with $x_0 = 0.5$, we obtain

$$x_1 = 0.176471, \; x_2 = 0.201568.$$
$$x_3 = 0.201640, \; x_4 = 0.201640.$$

The exact value correct to six decimal places is 0.201640.

Example 2.8 Perform four iterations of the Newton-Raphson method to obtain the approximate value of $(17)^{1/3}$ starting with the initial approximation $x_0 = 2$.

Let $x = (17)^{1/3}$. We obtain $x^3 = 17$ and $f(x) = x^3 - 17 = 0$. Using the Newton-Raphson method

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}, \quad k = 0, 1, \cdots$$

we get

$$x_{k+1} = x_k - \frac{x_k^3 - 17}{3x_k^2} = \frac{2x_k^3 + 17}{3x_k^2}, \quad k = 0, 1, \cdots$$

Starting with $x_0 = 2$, we obtain

$$x_1 = \frac{2x_0^3 + 17}{3x_0^2} = 2.75, \qquad x_2 = \frac{2x_1^3 + 17}{3x_1^2} = 2.582645$$

$$x_3 = \frac{2x_2^3 + 17}{3x_2^2} = 2.571332, \quad x_4 = \frac{2x_3^3 + 17}{3x_3^2} = 2.571282.$$

The exact value correct to six decimal places is 2.571282.

Example 2.9 Apply Newton-Raphson's method to determine a root of the equation

$$f(x) = \cos x - xe^x = 0$$

such that $|f(x^*)| < 10^{-8}$, where x^* is the approximation to the root. Take the initial approximation as $x_0 = 1$.

We write (2.14) in the form

$$x_{k+1} = x_k - \Delta x_k, \quad k = 0, 1, 2, \cdots$$

where $$\Delta x_k = \frac{f(x_k)}{f'(x_k)} = \frac{(\cos x_k - x_k e^{x_k})}{(-\sin x_k - x_k e^{x_k} - e^{x_k})}$$

Starting with $x_0 = 1$, we get

$$\Delta x_0 = \frac{\cos x_0 - x_0 e^{x_0}}{-\sin x_0 - x_0 e^{x_0} - e^{x_0}} = \frac{-2.17797952}{-6.27803464} = 0.34692060$$

$$x_1 = x_0 - \Delta x_0 = 1 - 0.34692060 = 0.65307940$$

$$\Delta x_1 = \frac{\cos x_1 - x_1 e^{x_1}}{-\sin x_1 - x_1 e^{x_1} - e^{x_1}} = \frac{-0.46064211}{-3.78394215} = 0.12173603$$

$$x_2 = x_1 - \Delta x_1 = 0.53134337.$$

The results obtained are given in Table 2.7.

Table 2.7 Approximations to the Root by the Newton-Raphson Method

k	x_k	Δx_k	x_{k+1}	$f(x_{k+1})$
0	1.0	0.3469	0.65307940	− 0.4606
1	0.65307940	0.1217	0.53134337	− 0.4180(− 1)
2	0.53134337	0.1343(− 1)	0.51790991	− 0.4641(− 3)
3	0.51790991	0.1525(− 3)	0.51775738	− 0.5926(− 7)
4	0.51775738	0.1948(− 7)	0.51775736	− 0.2910(− 10)

Example 2.10 Show that the initial approximation x_0 for finding $1/N$, where N is a positive integer, by the Newton-Raphson method must satisfy $0 < x_0 < 2/N$, for convergence.

We write
$$f(x) = \frac{1}{x} - N = 0.$$

The Newton-Raphson method becomes
$$x_{n+1} = 2x_n - N x_n^2.$$

Let us now draw the graphs of $y = x$ and $y = 2x - Nx^2$. The second curve is the parabola
$$\left(x - \frac{1}{N}\right)^2 = -\frac{1}{N}\left(y - \frac{1}{N}\right).$$

The graphs are given in Fig. 2.4. The point of intersection of these two curves is the required value $1/N$. From Fig. 2.4, we find that for any initial approximation outside the range $0 < x_0 < 2/N$, the method diverges. If $x_0 = 0$, the iterations do not converge to $1/N$ but remain zero always. This shows the importance of choosing a suitable initial approximation.

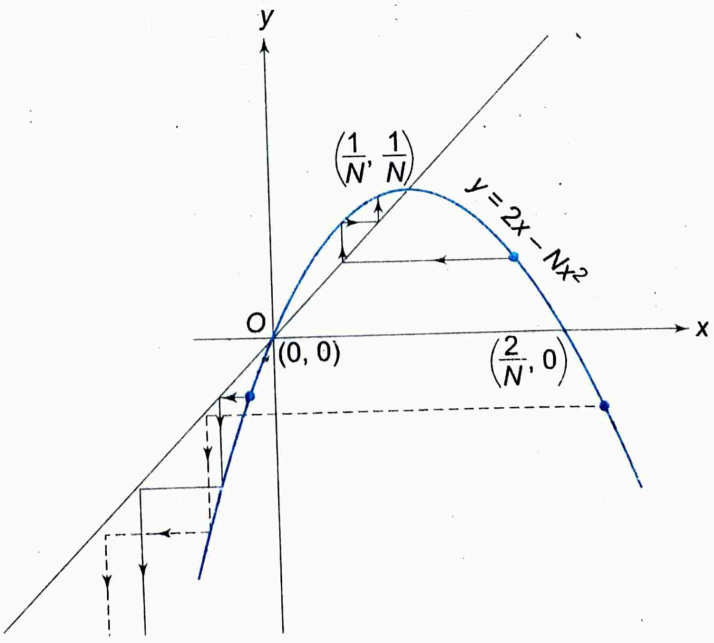


Fig. 2.4. Choice of the initial approximation.

EXERCISE 2.1

1. Find the interval in which the smallest positive root of the following equations lies:
   (a) $\tan x + \tanh x = 0$ (b) $x^3 - x - 4 = 0$.

   Determine the roots correct to two decimals using the bisection method.

2. The negative root of the smallest magnitude of the equation

$$f(x) = 3x^3 + 10x^2 + 10x + 7 = 0$$

   is to be obtained.
   (i) Find an interval of unit length which contains this root.
   (ii) Perform two iterations of the bisection method.
   (iii) Taking the end points of the last interval as initial approximations, perform three iterations of the (a) secant method, (b) Regula-Falsi method.

3. The smallest positive root of the equation

$$f(x) = x^4 - 3x^2 + x - 10 = 0$$

   is to be obtained.
   (i) Find an interval of unit length which contains this root.
   (ii) Perform two iterations of the bisection method.
   (iii) Taking the midpoint of the last interval as the initial approximation, perform three iterations of the Newton-Raphson method.

4. Given the following equations:
   (i) $x^4 - x - 10 = 0$
   (ii) $x - e^{-x} = 0$

   Determine the initial approximations to find the smallest positive root. Use these to find the root correct to three decimal places with the following methods:
   (a) the Regula-Falsi method, (b) the secant method,
   (c) the Newton-Raphson method.

5. Find the iterative methods based on the Newton-Raphson method for finding $\sqrt{N}$, $1/N$, $N^{1/}$ where N is a positive real number. Apply the methods to $N = 18$ to obtain the results correc to two decimals.

6. Perform two iterations with the Muller method for the following equations:
   (a) $x^3 - (1/2) = 0$, $x_0 = 0$, $x_1 = 1$, $x_2 = (1/2)$
   (b) $\ln x - x + 3 = 0$, $x_0 = 1/4$, $x_1 = 1/2$, $x_2 = 1$.

7. Use the Chebyshev third order method with $f(x) = x^2 - a$ and with $f(x) = 1 - a/x^2$ to obtai the iteration methods converging to $a^{1/2}$, in the form

$$x_{k+1} = \frac{1}{2}\left(x_k + \frac{a}{x_k}\right) - \frac{1}{8x_k}\left(x_k - \frac{a}{x_k}\right)^2$$

and $x_{k+1} = \frac{1}{2}x_k\left(3 - \frac{x_k^2}{a}\right) + \frac{3}{8}x_k\left(1 - \frac{x_k^2}{a}\right)^2$.

Perform two iterations with these methods to find the value of $\sqrt{6}$.

8. The equation

$$f(x) = x^4 - x - 10 = 0$$

has a root in the interval (1, 2). Taking the initial approximation $x_0 = 1.5$, perform three iterations of the

(i) Chebyshev iteration method (2.29),
(ii) multipoint iteration method (2.31),
(iii) multipoint iteration method (2.33).

9. The equation $x = f(x)$ is solved by the iteration method $x_{k+1} = f(x_k)$, and a solution is wanted with a maximum error not greater than 0.5×10^{-4}. The first and second iterates were computed; $x_1 = 0.50000$, $x_2 = 0.52661$. How many iterations must be performed further, if it is known that $|f'(x)| \leq 0.53$ for all values of x?

10. The root of the equation $x = g(x)$ is to be found by the iteration $x_{n+1} = g(x_n)$ with an error atmost equal to ε, $\varepsilon > 0$. It is known that $|g'| \leq 0.4$ in the neighbourhood of the root. The rounding errors on calculating $g(x)$ could be neglected in comparison with ε. The iterations proceed until the wanted accuracy is reached.

(a) How is this condition tested?
(b) How many iterations could be expected if the starting approximation differs with about $10^{-4}\varepsilon$ from a root.

(Royal Inst. Tech., Stockholm, Sweden, BIT 26(1986), 263)

11. A root of the equation $f(x) = x - F(x) = 0$ can often be determined by combining the iteration method with Regula-Falsi:

(i) With a given approximate value x_0 we compute

$$x_1 = F(x_0), \quad x_2 = F(x_1)$$

(ii) Observing that $f(x_0) = x_0 - x_1$ and $f(x_1) = x_1 - x_2$, we find a better approximation x' using Regula-Falsi on the points $(x_0, x_0 - x_1)$ and $(x_1, x_1 - x_2)$.
(iii) This last x' is taken as a new x_0 and we start from (i) all over again. Compute the smallest root of the equation $x - 5 \log_e x = 0$ with an error less than 0.5×10^{-4} starting with $x_0 = 1.3$.

(Inst. Tech., Stockholm, Sweden, BIT 6(1966), 176)

12. A program for the secant method was used to determine zeros of the function $f(x) = e^{-x}(2x^2 + 5x + 2) + 1$ with starting values $x_0 = -1$, $x_1 = 2.2$ and obtained $x_2 = 0.0511$, $x_3 = -18.4844$, $x_4 = 0.0511$, $x_5 = 0.0511$. For $n = 4$, the stopping criterion

$$|x_{n+1} - x_n| \leq 10^{-5}\,|x_{n+1}|$$

was satisfied and the program considered x_5 to be a good approximation to a zero.
(a) Estimate the error in x_5.
(b) Considering the answer to (a), suggest a suitable stopping criterion.

(Inst. Tech., Linköping, Sweden, BIT 28(1988), 128)

13. The root of the equation $x = (1/2) + \sin x$, by using the iteration method

$$x_{k+1} = (1/2) + \sin x_k,\ x_0 = 1$$

correct to six decimal places is $x = 1.497300$. Determine the number of iteration steps required to reach the root by the linear iteration. If the Aitken Δ^2 process is used after three approximations are available, how many iterations are required?

14. (a) Newton-Raphson's method for solving the equation $f(x) = c$, where c is a real valued constant, is applied to the function

$$f(x) = \begin{cases} \cos x & \text{when } |x| \le 1 \\ \cos x + \left(x^2 - 1\right)^2 & \text{when } |x| \ge 1. \end{cases}$$

For which c is $x_n = (-1)^n$, when $x_0 = 1$ and the calculations are carried out with no errors?
(b) Even in high precision arithmetic, say 10 decimals, the convergence is troublesome. Explain.

(Uppsala Univ., Sweden, BIT 24(1984), 129)

15. The equation $f(x) = 0$ where

$$f(x) = 0.1 - x + \frac{x^2}{(2!)^2} - \frac{x^3}{(3!)^2} + \frac{x^4}{(4!)^2} - \cdots$$

has one root in the interval $(0, 1)$. Calculate this root correct to 5 decimals.

(Inst. Tech., Linköping, Sweden, BIT 24(1984), 258)

16. Find all positive roots of the equation

$$10 \int_0^x e^{-x^2} dt - 1 = 0$$

with six correct decimals.

(Uppsala Univ., Sweden, BIT 27(1987), 129)

17. The equation $x = 0.2 + 0.4 \sin (x/b)$, where b is a parameter, has one solution near $x = 0.3$. The parameter is known only with some uncertainty: $b = 1.2 \pm 0.05$. Calculate the root with an accuracy reasonable with respect to the uncertainty of b.

(Royal Inst. Tech., Stockholm, Sweden, BIT 26(1986), 386)

18. Show that the equation

$$f(x) = \cos\left(\frac{\pi(x+1)}{8}\right) + 0.148x - 0.9062 = 0$$

has one root in the interval $(-1, 0)$ and one in $(0, 1)$. Calculate the negative root correct to 4 decimals.

(Inst. Tech., Lyngby, Denmark, BIT 25(1985), 299)

19. Find all the roots of $\cos x - x^2 - x = 0$ to five decimal places.

(Lund. Univ., Sweden, BIT 27(1987), 285)

20. Find a Catenary $y = c \cosh((x - a)/c)$ passing through the points $(1, 1)$ and $(2, 3)$.

(Royal Inst. Tech., Stockholm, Sweden, BIT 29(1989), 375)

21. (a) Show that the equation $\log_e(x) = x^2 - 1$ has exactly two real roots, $\alpha_1 = 0.45$ and $\alpha_2 = 1$.

(b) Determine for which initial approximation x_0, the iteration

$$x_{n+1} = \sqrt{1 + \log_e(x_n)}$$

converges to α_1 or α_2.

(c) Determine for which initial approximation x_0, the iteration

$$x_{n+1} = x_n - \frac{x_n - \exp\left(x_n^2 - 1\right)}{1 - 2x_n \exp\left(x_n^2 - 1\right)}$$

converges to a root of the equation in (a) and in case of convergence also which root.

(Uppsala Univ., Sweden, BIT 10(1970), 115)

22. The equation $x^2 + ax + b = 0$ has two real roots α and β. Show that the iteration method

$$x_{k+1} = -(ax_k + b)/x_k$$

is convergent near $x = \alpha$ if $|\alpha| > |\beta|$ and that

$$x_{k+1} = -b/(x_k + a)$$

is convergent near $x = \alpha$ if $|\alpha| < |\beta|$.

Show also that the iteration method

$$x_{k+1} = -(x_k^2 + b)/a$$

is convergent near $x = \alpha$ if $2|\alpha| < |\alpha + \beta|$.

23. A root of the equation $f(x) = 0$ can be obtained by combining the Newton-Raphson method and the Regula-Falsi method. We start from $x_0 = \xi + \varepsilon$ where ξ is the true solution of $f(x) = 0$. Further $y_0 = f(x_0)$, $x_1 = x_0 - f_0/f_0'$ and $y_1 = f_1$ are computed. Lastly, a straight line is drawn through the points (x_1, y_1) and $((x_0 + x_1)/2, y_0/2)$. If ε is sufficiently small, the intersection of the line and the x-axis gives a good approximation to ξ. To what power of ε is the error term proportional? Use this method to compute the root of the equation $x^4 - x - 10 = 0$, correct to three decimal places.

24. Determine the order of convergence of the iterative method

$$x_{k+1} = (x_0 f(x_k) - x_k f(x_0))/(f(x_k) - f(x_0)).$$

for finding a simple root of the equation $f(x) = 0$.

25. Let $x = \xi$ be a simple root of the equation $f(x) = 0$. We try to find the root by means of the iteration formula

$$x_{i+1} = x_i - (f(x_i))^2/(f(x_i) - f(x_i - f(x_i))).$$

Find the order of convergence and compare the convergence properties with those of Newton-Raphson's method.

(Bergen Univ., Sweden, BIT 20(1980), 26

26. Determine p, q and r so that the order of the iterative method

$$x_{n+1} = px_n + qa/x_n^2 + ra^2/x_n^5$$

for $a^{1/3}$ becomes as high as possible. For this choice of p, q and r, indicate how the error in x_{n+1} depends on the error in x_n.

(Lund. Univ., Sweden, BIT 8(1968), 13

27. The equation $f(x) = 0$ has a simple root in the interval (1, 2). The function $f(x)$ is such that

$$|f'(x)| \geq 4 \text{ and } |f''(x)| \leq 3 \text{ for all } x \text{ in } (1, 2).$$

Assuming that the Newton-Raphson method converges for all initial approximations in (1, 2) find the maximum number of iterations required to obtain root correct to 6 decimal places after rounding.

28. Consider the iteration method

$$x_{k+1} = \phi(x_k), \quad k = 0, 1, 2, \cdots$$

for solving the equation $f(x) = 0$. We choose the iteration function in the form

$$\phi(x) = x - \gamma_1 f(x) - \gamma_2 f^2(x) - \gamma_3 f^3(x)$$

where γ_1, γ_2 and γ_3 are arbitrary parameters to be determined. Find the γ's such that the iteration method has the following orders: (i) third, (ii) fourth.

Apply these methods to determine a root of the equation

$$x = \frac{1}{5} e^x \text{ with } x_0 = 0.4$$

correct to three decimal places.

29. The equation

$$x^3 - 5x^2 + 4x - 3 = 0$$

has one root near $x = 4$, which is to be computed by the iteration

$$x_0 = 4$$

$$x_{n+1} = \frac{3+(k-4)x_n + 5x_n^2 - x_n^3}{k}, \quad k \text{ integer.}$$

(a) Determine which value of k will give the fastest convergence.
(b) Using this value of k, iterate three times and estimate the error in x_3.

(Royal Inst. Tech., Stockholm, Sweden, BIT 11(1971), 125)

30. The equation $f(x) = 0$ is given,
(i) Obtain an iteration method using the rational approximation

$$f(x) = \frac{x - a_0}{b_0 + b_1 x}$$

where the coefficients a_0, b_0 and b_1 are determined by evaluating $f(x)$ at x_k, x_{k-1} and x_{k-2}.
(ii) Find the order of convergence for this method.
(iii) Carry out two iterations using this method for the equation

$f(x) = 2x^3 - 3x^2 + 2x - 3 = 0$ with $x_0 = 0$, $x_1 = 1$, $x_2 = 2$.

31. Find the order of convergence of the **Steffensen** method

$$x_{k+1} = x_k - \frac{f_k}{g_k}, \quad k = 0, 1, 2, \cdots$$

$$g_k = \frac{f(x_k + f_k) - f_k}{f_k}$$

where $f_k = f(x_k)$. Use this method to determine the nonzero root of the equation

$$f(x) = x - 1 + e^{-2x} = 0, \text{ with } x_0 = 0.7$$

correct to three decimals.

32. A sequence $\{x_n\}_1^\infty$ is defined by

$$x_0 = 5$$

$$x_{n+1} = \frac{1}{16}x_n^4 - \frac{1}{2}x_n^3 + 8x_n - 12.$$

Show that it gives cubic convergence to $\xi = 4$.
Calculate the smallest integer n for which the inequality

$$|x_n - \xi| < 10^{-6}$$

is valid.

(Uppsala Univ., Sweden, BIT 13(1973), 493)

33. If an attempt is made to solve the equation $x = 1.4 \cos x$ by using the iteration formula

$$x_{n+1} = 1.4 \cos x_n$$

it is found that for large n, x_n alternates between two values A and B.

(i) Calculate A and B correct to 3 decimal places.
(ii) Calculate the correct solution of the equation to 4 decimal places.

(Lund. Univ., Sweden, BIT 17(1977), 115)

34. We wish to compute the root of the equation

$$e^{-x} = 3 \log_e(x)$$

using the formula

$$x_{n+1} = x_n - \frac{3 \log_e(x_n) - \exp(-x_n)}{p}.$$

Show that $p = 3$ gives rapid convergence.

(Stockholm Univ., Sweden, BIT 14(1974), 254)

35. The equation

$$2e^{-x} = \frac{1}{x+2} + \frac{1}{x+1}$$

has two roots greater than -1.
Calculate these roots correct to five decimal places.

(Inst. Tech., Lund., Sweden, BIT 21(1981), 136)

36. Find the positive root of the equation

$$e^x = 1 + x + \frac{x^2}{2} + \frac{x^3}{6}\, e^{0.3x}$$

correct to six decimal places.

(Royal Inst. Tech., Stockholm, Sweden, BIT 21(1981), 242)

37. Assuming that Δx in the Taylor expansion of $f(x_0 + \Delta x)$, can be approximated by

$$\Delta x = a_1 f(x_0) + a_2 f^2(x_0) + a_3 f^3(x_0) + \cdots,$$

where $a_0, a_1, a_2, \cdots$ are arbitrary parameters to be determined, derive the Chebyshev methods of third and fourth orders for finding a root of $f(x) = 0$.

38. We consider the multipoint iteration method

$$x_{k+1} = x_k - \alpha \frac{f(x_k)}{f'(x_k - \beta f(x_k)/f'(x_k))}$$

where α and β are arbitrary parameters, for solving the equation $f(x) = 0$. Determine α and β such that the multipoint method is of order as high as possible for finding ξ, a simple root of $f(x) = 0$.

39. Show that the equation

$$f(x) = 1 - x\, e^{1-x} = 0$$

has a double root at $x = 1$. The root is obtained by using the

(i) Newton-Raphson method (2.14),
(ii) modified Newton-Raphson method (2.62) with $m = 2$, starting with $x_0 = 0$.

Verify that the rate of convergence is linear in (i) and quadratic in (ii).

40. Apply the Newton-Raphson method, with $x_0 = 0.8$, the secant method with $x_0 = 0.8$, $x_1 = 1.2$, and the Muller method with $x_0 = 0.6$, $x_1 = 0.8$, $x_2 = 1.2$ to the equation

$$f(x) = x^3 - x^2 - x + 1 = 0$$

and verify that the convergence is only of first order in each case.

Then, apply the Newton-Raphson method (2.62), with $m = 2$ and verify that the convergence is of second order.

41. The multiple root ξ of multiplicity two of the equation $f(x) = 0$ is to be determined. We consider the multipoint method

$$x_{k+1} = x_k - \frac{1}{2}\frac{f\left(x_k + 2f(x_k)/f'(x_k)\right)}{f'(x_k)}$$

Show that the iteration method has third order rate of convergence. Hence, solve the equation

$$9x^4 + 30x^3 + 34x^2 + 30x + 25 = 0, \text{ with } x_0 = -1.4$$

correct to three decimals.

42. Determine the constants α and β in the iteration method (modified Chebyshev method)

$$x_{k+1} = x_k - \alpha\frac{f(x_k)}{f'(x_k)} - \beta\left[\frac{f(x_k)}{f'(x_k)}\right]^2\frac{f''(x_k)}{f'(x_k)}, \quad k = 0, 1, \cdots$$

so that the method has cubic rate of convergence for obtaining multiple roots of multiplicity m.

43. The system of equations

$$x^2y + y^3 = 10$$
$$xy^2 - x^2 = 3$$

has a solution near $x = 0.8$, $y = 2.2$. Perform two iterations of the Newton's method to obtain this root.

44. Calculate all solutions of the system

$$x^2 + y^2 = 1.12$$
$$xy = 0.23$$

correct to three decimal places.

(Lund. Univ., Sweden, BIT 20(1980), 389)

45. The system of equations

$$y\cos(xy) + 1 = 0$$
$$\sin(xy) + x - y = 0$$

Chapter 2

has one solution close to $x = 1$, $y = 2$. Calculate this solution correct to 2 decimal places.

(Umea Univ., Sweden, BIT 19(1979), 552)

46. The system of equations

$$\log_e(x^2 + y) - 1 + y = 0$$
$$\sqrt{x} + xy = 0$$

has one approximate solution $(x_0, y_0) = (2.4, -0.6)$. Improve this solution and estimate the accuracy of the result.

(Lund. Univ., Sweden, BIT 18(1978), 366)

47. Describe how, in general, suitable values of a, b, c and d may be estimated so that the sequence of values of x and y determined from the recurrence relations

$$x_{n+1} = x_n + af(x_n, y_n) + bg(x_n, y_n)$$
$$y_{n+1} = y_n + cf(x_n, y_n) + dg(x_n, y_n)$$

will converge to a solution of

$$f(x, y) = 0$$
$$g(x, y) = 0.$$

Illustrate the method by finding a suitable initial point and a recurrence relation to find the real solution of

$$y = \sin(x + y)$$
$$x = \cos(y - x).$$

48. Calculate the solution of the system of equations

$$x^3 + y^3 = 53$$
$$2y^3 + z^4 = 69$$
$$3x^5 + 10z^2 = 770$$

which is close to $x = 3$, $y = 3$, $z = 2$.

(Stockholm Univ., Sweden, BIT 19(1979), 285)

49. Consider the system of equations $f(x, y) = 0$, $g(x, y) = 0$.

Let $x = x_0 + \Delta x$, $y = y_0 + \Delta y$, where (x_0, y_0) is an initial approximation to the solution. Assume that

$$\Delta x = A_1(x_0, y_0) + A_2(x_0, y_0) + A_3(x_0, y_0) + \cdots$$
$$\Delta y = B_1(x_0, y_0) + B_2(x_0, y_0) + B_3(x_0, y_0) + \cdots$$

where $A_1(x_0, y_0)$, $B_1(x_0, y_0)$ are linear in f_0, g_0; $A_2(x_0, y_0)$, $B_2(x_0, y_0)$ are quadratic in f_0, g_0 and so on. Use Taylor's series method to derive iterative methods of second and third orders.

50. Perform three iterations of the Newton-Raphson method to determine the complex roots of the following equations

(i) $1 + z^2 = 0$, $z_0 = (1 + i)/2$.

(ii) $z^3 - 4iz^2 - 3e^z = 0$, $z_0 = -0.53 - 0.36i$.

EXERCISE 2.1

1. (a) (2.3, 2.4). Root lies in the interval (2.3625, 2.36875). Root correct to two decimals is 2.37.
   (b) (1.0, 2.0). Root lies in the interval (1.7959, 1.7969). Root correct to two decimals is 1.80.
2. (i) (– 3, – 2), (ii) root lies in the interval (– 2.5, – 2.25).
   (iii) (a) $x_0 = -2.5$, $x_1 = -2.25$, $x_2 = -2.321596$, $x_3 = -2.334260$, $x_4 = -2.333324$.
   (b) $x_0 = -2.5$, $x_1 = -2.25$, $\xi \in (x_0, x_1)$, $x_2 = -2.321596$; $\xi \in (x_0, x_2)$, $x_3 = -2.331744$; $\xi \in (x_0, x_3)$, $x_4 = -2.333119$.
3. (i) (2, 3), (ii) root lies in the interval (2, 2.25),
   (iii) $x_0 = 2.125$, $x_1 = 2.163712$, $x_2 = 2.162429$, $x_3 = 2.162428$.
4. (i) Root lies in the interval (1.0, 2.0), 1.856.
   (ii) Root lies in the interval (0.0, 1.0), 0.567.
5. (i) $x_{n+1} = (1/2)\,[x_n + (N/x_n)]$, $n = 0, 1, \cdots$
   (ii) $x_{n+1} = x_n\,(2 - N x_n)$, $n = 0, 1, \cdots$
   (iii) $x_{n+1} = (2x_n^3 + N)/(3x_n^2)$, $n = 0, 1, \cdots$
   For $N = 18$, (i) 4.24, (ii) 0.06, (iii) 2.62.
6. (a) $x_3 = 0.7676$, $x_4 = 0.7929$. (b) $x_3 = 2.1572$, $x_4 = 3.2006$.
7. (i) $x_0 = 2$, $x_1 = 2.4375$, $x_2 = 2.4495$.
   (ii) $x_0 = 2$, $x_1 = 2.4167$, $x_2 = 2.4495$.

8. (i) $x_0 = 1.5,\ x_1 = 1.728557,\ x_2 = 1.852522,\ x_3 = 1.855584.$

(ii) $x_0 = 1.5,\ x_1^* = 1.7575,\ x_1 = 1.810776;\ x_2^* = 1.834063, x_2 = 1.855524;\ x_3^* = 1.855554,$
$x_3 = 1.855585.$

(iii) $x_0 = 1.5,\ x_1^* = 2.015,\ x_1 = 1.657366;\ x_2^* = 1.896300,$
$x_2 = 1.836186;\ x_3^* = 1.855907,\ x_3 = 1.855574.$

9. $|x_{n+1} - \xi| \leq \dfrac{c}{1-c}|x_{n+1} - x_n|,\ |f'(\xi)| \leq c$

$$|x_2 - \xi| \leq \frac{c}{1-c}|x_2 - x_1| \leq 0.03001$$

$$|x_{n+1} - \xi| \leq c^n |x_2 - \xi| < 5 \times 10^{-5}$$

which gives $n \geq 11$.

10. (a) $|x_{n+1} - x_n| < 1.5\ \varepsilon$. (b) about 10 iterations.

11. 1.2959.

12. (a) $|f''(x_5)| / |f'(x_5)| \leq 1.2.$

(b) include a test $|f(x_{n+1})| \leqq$ tolerance.

13. 6 by linear iteration; 3 iterations of Δ^2 process.

14. (a) Applying the Newton-Raphson method to the equation $f(x) - c = 0$, we get

$$x_{n+1} = x_n - [f(x_n) - c]/f'(x_n).$$

For $n = 0$, we get $x_1 = x_0 + (1/\sin 1)(\cos 1 - c) = -1$. Hence, $c = \cos 1 + 2 \sin 1$. With this value of c, we get $x_2 = 1,\ x_3 = -1,\ \cdots,\ x_n = (-1)^n$.

(b) Since $f' = 0$ between x_0 and the root(s) and also at $x = 0$, the convergence will be poor inspite of high precision arithmetic.

15. $0.10260 \pm 2 \times 10^{-6}$.

16. The equation $f(x) = 10x\ e^{-x^2} - 1 = 0$ has two positive roots in the intervals (0, 1) and (1, 2). Using the Newton-Raphson method with $x_0 = 0.1$, we get the first root as 0.100336 and with $x_0 = 1.6$, we obtain the second root as 1.679631.

17. Take $b = 1.2$ and use the Newton-Raphson method with $x_0 = 0.3$. The root correct to three decimal places is 0.299.

18. Use the Newton-Raphson method with $x_0 = -0.5$. We get $x_1 = -0.504397,\ x_2 = -0.508143,$ $x_3 = -0.508146$. Root is -0.5081.

19. The equation $f(x) = 0$ has a real root in the interval $(-2, -1)$ and another root in the interval (0, 1). Roots are -1.25115 and 0.55001.

20. Since the Catenary passes through the points (1, 1) and (2, 3), we have

$c \cosh [(1 - a)/c] = 1$ and $c \cosh [(2 - a)/c] = 3$.

On eliminating a from these equations, we get

$$c = \frac{1 + c\cosh^{-1}(1/c)}{\cosh^{-1}(3/c)} = g(c).$$

Using the iteration method $c_{k+1} = g(c_k)$, $c_0 = 0.5$, we obtain $c_1 = 0.669311$, $c_2 = 0.752368$, $c_3 = 0.774114$, $c_4 = 0.777000$, $c_5 = 0.777277$, $c_6 = 0.777303$, $c_7 = 0.777305$ and $a = 0.424822$. Newton-Raphson method can also be used.

21. (a) The roots are points of intersection of the curves $y = \log_e x$ and $y = x^2 - 1$. These curves intersect at $\alpha_1 = 0.45$ and $\alpha_2 = 1$.

(b) $x_0 < \alpha_1$, no convergence; $x_0 > \alpha_2$, convergence to α_2.

(c) $x_0 < \gamma$, convergence to α_1; $x_0 > \gamma$, convergence to α_2, where γ is the zero of $1 - 2xe^{x^2-1}$.

23. Error term is proportional to ε_0^4. Root correct to three decimals is 1.856.

24. Linear. $\varepsilon_{k+1} = (1/2)\ \varepsilon_k\ \varepsilon_0\ f''(\xi)/f'(\xi) + O(\varepsilon_k^2\ \varepsilon_0 + \varepsilon_k\ \varepsilon_0^2)$.

25. Order of convergence = 2 when $f'(\xi) \neq 1$ or $f'(\xi) \neq 0$. The asymptotic error constant is $[(1 - f'(\xi)]\ f''(\xi)/(2f'(\xi))$. In Newton-Raphson method, asymptotic error constant is $f''(\xi)/(2f'(\xi))$.

26. $p = 5/9$, $q = 5/9$, $r = -1/9$. Third order $\varepsilon_{n+1} = 5\varepsilon_n^3/(3\xi^2) + O(\varepsilon_n^4)$.

27. The error equation for the Newton-Raphson method is given by

$\varepsilon_{k+1} = c\varepsilon_k^2$, $k = 0, 1, \cdots$ and $c = f''(\xi)/(2f'(\xi))$.

Replacing k by $k-1$ successively, we get $\varepsilon_k = c^{2^k-1}\varepsilon_0^{2^k}$. Since $\xi \in (1, 2)$, we get $c \leq 0.375$. Also, $|\varepsilon_0| \leq 1$ in the interval $(1, 2)$. Therefore, find k such that $|\varepsilon_k| \leq (0.375)^{2^k-1} \leq 5 \times 10^{-7}$. We obtain $k \geq 4$.

28. Third order: $\gamma_1 = \dfrac{1}{f_k'}$, $\gamma_2 = \dfrac{1}{2}\dfrac{f_k''}{(f_k')^3}$, $\gamma_3 = 0$,

Fourth order: $\gamma_1 = \dfrac{1}{f_k'}$, $\gamma_2 = \dfrac{1}{2}\dfrac{(f_k'')^2}{(f_k')^3}$, $\gamma_3 = \left[\dfrac{1}{2}\dfrac{(f_k'')^2}{(f_k')^2} - \dfrac{1}{6}\dfrac{f_k'''}{f_k'}\right]\dfrac{1}{(f_k')^3}$

Root: 0.259.

29. (a) Let ξ be the exact root. We have $\xi^3 - 5\xi^2 + 4\xi - 3 = 0$. Using $x_n = \xi + \varepsilon_n$ and $\xi = 4 + \delta$, we obtain the error equation

$$k\ \varepsilon_{n+1} = (k - 12)\ \varepsilon_n + O(\delta\varepsilon_n)$$

For fastest convergence $k = 12$.

(b) $x_1 = 4.25$, $x_2 = 4.2122$, $x_3 = 4.2230$, $x_4 = 4.2201$. Since the root is correct to 2 decimal places, maximum error is 0.005.

30. (i) $x_{k+1} = x_k + \dfrac{(x_k - x_{k-1})(x_k - x_{k-2})(f_{k-1} - f_{k-2}) f_k}{(x_k - x_{k-1})(f_{k-2} - f_k) f_{k-1} + (x_k - x_{k-2})(f_k - f_{k-1}) f_{k-2}}$

(ii) $\varepsilon_{k+1} = \left(\dfrac{c_2^2}{c_1^2} - \dfrac{c_3}{c_1} \right) \varepsilon_k \, \varepsilon_{k-1} \, \varepsilon_{k-2}$, $c_r = f^{(r)}(\xi)/r!$. Order: 1.84.

(iii) $x_3 = 1.615$, $x_4 = 1.485$.

31. $\varepsilon_{k+1} = (1/2)\,[1 + f'(\xi)]\,[f''(\xi)/f'(\xi)]\,\varepsilon_k^2 + O(\varepsilon_k^3)$. Order 2. Root: 0.797.

32. For $\xi = 4$, $\varepsilon_{n+1} = (1/2)\,\varepsilon_n^3 + O(\varepsilon_n^4)$, $n \geq 4$.

33. (i) In the iteration method $x_{n+1} = g(x_n)$, where $g(x) = 1.4 \cos x$, the condition of convergence $|g'(x)| < 1$ is violated when $x_n > 0.79$. Starting with $x_0 = 0$, we get $x_1 = 1.4$, $x_2 = 0.238$, $x_3 = 1.3606$, $x_4 = 0.2921$, $x_5 = 1.3407$, $\cdots$, $x_{28} = 0.3615$, $x_{29} = 1.3095$, $x_{30} = 0.3616$. Hence, the root lies between two values $A = 0.362$ and $B = 1.309$.

(ii) We write $f(x) = x - 1.4 \cos x = 0$ and apply Newton-Raphson method. $x_0 = 0.5$, $x_1 = 0.935985$, $x_2 = 0.886273$, $x_3 = 0.885091$, $x_4 = 0.885770$, $x_5 = 0.885771$. Root is 0.8858.

34. The error equation is obtained as

$$\varepsilon_{n+1} = \left[1 - \left\{ (3/\xi) + e^{-\xi} \right\} / p \right] \varepsilon_n + O(\varepsilon_n^2)$$

The method will have rapid convergence if $p = e^{-\xi} + (3/\xi)$, where ξ is the root of the equation $e^{-x} - 3 \log_e x = 0$. The root of this equation is $\xi \approx 1.1154$. Hence, $p \approx 3$.

35. Define $f(x) = 2e^{-x} - \left[\dfrac{1}{x+2} + \dfrac{1}{x+1} \right]$. Since $f(-0.8) < 0$, $f(0) > 0$ and $f(1) < 0$, the roots of $f(x) = 0$ which are greater than -1 lie in the intervals $(-0.8, 0)$ and $(0, 1)$. Use Newton-Raphson method. Roots are -0.689752, 0.770091.

36. Use Newton-Raphson method. $2.363376 \pm 0.5 \times 10^{-6}$.

37. $x_{k+1} = x_k - \dfrac{f_k}{f_k'} + \dfrac{1}{2} \left[\dfrac{f_k}{f_k'} \right]^2 \dfrac{f_k''}{f_k'}$ (third order)

$$x_{k+1} = x_k - \frac{f_k}{f_k'} + \frac{1}{2} \left[\frac{f_k}{f_k'} \right]^2 \frac{f_k''}{f_k'} - \frac{1}{2} \left[\left\{ \frac{f_k''}{f_k'} \right\}^2 - \frac{1}{6} \frac{f_k'''}{f_k'} \right] \left[\frac{f_k}{f_k'} \right]^3 \quad \text{(fourth order)}$$

38. $\alpha = 1$, $\beta \neq 1/2$, second order. $\alpha = 1$, $\beta = 1/2$, third order. In this case,

$$\varepsilon_{n+1} = \left(\frac{11}{144} c_3^2 - \frac{1}{12} c_4 \right) \varepsilon_n^3 + O(\varepsilon_n^4); \; c_i = f^{(i)}(\xi)/f''(\xi).$$

39. Since $f(1) = f'(1) = 0$ and $f''(1) \neq 0$, the root $x = 1$ is a double root.

(i) $x_0 = 0$, $x_1 = 0.367879$, $x_2 = 0.626666$, $x_3 = 0.792119$, $x_4 = 0.889218$, $x_5 = 0.942620$. Since $\varepsilon_{k+1}/\varepsilon_k$, $k = 0, 1, 2, \cdots$ tends to a constant, convergence is linear.

(ii) $x_0 = 0$, $x_1 = 0.735759$, $x_2 = 0.978188$, $x_3 = 0.999809$, $x_4 = 1.000018$. Since $\varepsilon_{k+1}/\varepsilon_k^2$ tends to a constant, convergence is quadratic.

41. Error equation is

$$\varepsilon_{n+1} = \left(\frac{11}{144}c_3^2 - \frac{1}{12}c_4\right)\varepsilon_n^3 + O(\varepsilon_n^4);\ c_i = f^{(i)}(\xi)/f''(\xi),\ i = 3, 4, \ldots$$

$x_0 = 1.4$, $x_1 = -1.66056$, $x_2 = -1.66667$, $x_3 = -1.66667$. Root is -1.667.

42. $$\frac{f_k}{f_k'} = \frac{1}{m}\varepsilon_n - \frac{\varepsilon_n^2}{m^2(m+1)}c_{m+1} + \frac{\varepsilon_n^3}{m^2(m+1)}\left[\frac{1}{m}c_{m+1}^2 - \frac{2}{m+2}c_{m+2}\right] + \cdots$$

where $c_{m+i} = f^{(m+i)}(\xi)/f^{(m)}(\xi)$, $i = 1, 2, \cdots$

$$\frac{f_k''}{f_k'} = \frac{m-1}{\varepsilon} + \frac{1}{m}c_{m+1} + \frac{\varepsilon_n}{m}\left[\frac{2}{m+1}c_{m+2} - \frac{1}{m}c_{m+1}^2\right] + \cdots$$

Substituting in the given method and equating the coefficients of ε_n and ε_n^2 to zero, we get $\alpha = m(3-m)/2$ and $\beta = m^2/2$. The error equation becomes

$$\varepsilon_{n+1} = \frac{1}{m(m+1)}\left[\frac{m+3}{2m(m+1)}c_{m+1}^2 - \frac{1}{m+2}c_{m+2}\right]\varepsilon_n^3 + O(\varepsilon_n^4).$$

43. $$\begin{pmatrix} x_1 \\ y_1 \end{pmatrix} = \begin{pmatrix} 0.9013 \\ 2.0409 \end{pmatrix},\ \begin{pmatrix} x_2 \\ y_2 \end{pmatrix} = \begin{pmatrix} 0.9771 \\ 2.0079 \end{pmatrix}$$

44. $x = 1.035$, $y = 0.222$. Hence, solutions of the equation are $(1.035, 0.222)$ and $(-1.035, -0.222)$.

45. $x = 1.09$, $y = 1.94$.

46. $x = 2.412249$, $y = -0.643856$.

47. $a = -g_y/D$, $b = f_y/D$, $c = g_x/D$, $d = -f_x/D$, $D = f_x g_y - g_x f_y$; partial derivatives to be evaluated at (x_n, y_n); $x_0 = 0$, $y_0 = \sqrt{2}$.

48. $x_1 = 2.999805$, $y_1 = 2.963158$, $z_1 = 2.030921$; $x_2 = 2.999820$, $y_2 = 2.962681$, $z_2 = 2.030252$.

49. (i) $x_{k+1} = x_k + A_1,\ y_{k+1} = y_k + B_1$, (Second order),

$A_1 = -(fg_y - gf_y)/D,\ B_1 = -(gf_x - fg_x)/D,\ D = f_x g_y - g_x f_y$

(ii) $x_{k+1} = x_k + A_1 + A_2;\ y_{k+1} = y_k + B_1 + B_2$, (Third order),

$A_2 = -(f_2 g_y - g_2 f_y)/D;\ B_2 = -(g_2 f_x - f_2 g_x)/D,$

$f_2 = (A_1^2 f_{xx} + 2A_1 B_1 f_{xy} + B_1^2 f_{yy})/2,$

$g_2 = (A_1^2 g_{xx} + 2A_1 B_1 g_{xy} + B_1^2 g_{yy})/2.$

50. (i) $(-0.25, 0.75)$, $(0.075, 0.975)$, $(-0.0017, 0.9973)$.

(ii) $(-0.5080, -0.3864)$, $(-0.5088, -0.3866)$, $(-0.5088, -0.3867)$.

EXERCISE 2.2

Practice the Techniques

Find the positive real zero of the given functions; start with consecutive integers a and b that bracket the root for bisection, regula falsi, or secant methods. Use $(a+b)/2$ as the starting value for Newton's method and as the third starting value for Muller's method.

(a) *Use the bisection method.*
(b) *Use regula falsi.*
(c) *Use the secant method.*
(d) *Use Newton's method.*
(e) *Use Muller's method.*

P2.1.
$$f(x) = x^2 - 2$$

P2.2.
$$f(x) = x^2 - 5$$

P2.3.
$$f(x) = x^2 - 7$$

P2.4.
$$f(x) = x^3 - 3$$

P2.5.
$$f(x) = x^3 - 4$$

P2.6.
$$f(x) = x^3 - 6$$

P2.7.
$$f(x) = x^4 - 0.45$$

P2.8.
$$f(x) = x^4 - 0.65$$

P2.9.
$$f(x) = x^4 - 0.06$$

P2.10.
$$f(x) = x^4 - 0.25$$

Find all real zeros of the functions that follow; find starting values as above.

(a) *Use the bisection method.*
(b) *Use regula falsi.*
(c) *Use the secant method.*
(d) *Use Newton's method.*
(e) *Use Muller's method.*

P2.11.
$$f(x) = x^3 - 9x + 2$$

P2.12.
$$f(x) = x^3 - 2x^2 - 5$$

P2.13.
$$f(x) = x^3 + 3x^2 - 1$$

P2.14.
$$f(x) = x^3 - 4x + 1$$

P2.15.
$$f(x) = x^3 - x^2 - 4x - 3$$

P2.16.
$$f(x) = x^3 - 6x^2 + 11x - 5$$

P2.17.
$$f(x) = 6x^3 - 23x^2 + 20x$$

P2.18.
$$f(x) = 3x^3 - x^2 - 18x + 6$$

P2.19.
$$f(x) = x^3 - x^2 - 24x - 32$$

P2.20.
$$f(x) = x^3 - 7x^2 + 14x - 7$$

P2.21. For each of the following equations, use Newton's method with the specified starting value to find a root; discuss the source of the difficulty if Newton's method fails.

(a) $f(x) = -5x^4 + 11x^2 - 2; x_0 = 1.$
(b) $f(x) = x^3 - 4x + 1; x_0 = 0.$
(c) $f(x) = 5x^4 - 11x^2 + 2;$
$x_0 = 1; x_0 = 1/2; x_0 = 0.$
(d) $f(x) = x^5 - 0.5; x_0 = 1.$

P2.22. Find the zeros of the following Legendre polynomials:

(a) $P_2(x) = (3x^2 - 1)/2.$
(b) $P_3(x) = (5x^3 - 3x)/2.$
(c) $P_4(x) = (35x^4 - 30x^2 + 3)/8.$
(d) $P_5(x) = (63x^5 - 70x^3 + 15x)/8.$

P2.23. Find the first three positive zeros of $y = x\cos(x) + \sin(x)$.

P2.24. Find the intersection(s) of $y = e^x$ and $y = x^3$; i.e., find the zeros of $f(x) = e^x - x^3$.

P2.25. Find the intersection(s) of $y = e^x$ and $y = x^2$.

P2.26. Find all intersections of $y = 2^x$ and $y = x^2$.

P2.27. Find the point(s) of intersection of x^c and c^x for $c = 3$ and $c = 2.7$.

(It is interesting to note that for $c = e$, $x^c \leq c^x$ for all x.)

P2.28. Find the intersection(s) of $y = -a + e^x$ and $y = b + \log(x)$.

(a) $a = 5, b = 1.$
(b) $a = 3, b = 2.$
(c) $a = 1, b = 5.$

P2.29. Find the zeros of $y = f(x) = \log(x + 0.1) + 1.5$.

Chapter 2

P2.1 x = 1.4142

P2.3 x = 2.6458

P2.5 x = 1.5874

P2.7 x = 0.8190

P2.9 x = 0.4949

P2.11 x = -3.1055, 0.2235, 2.8820

P2.13 x = -2.8794, -0.6527, 0.5321

P2.15 x = 2.8063

P2.17 x = 0, 1.3333, 2.5

P2.19 x = -3.3240, -1.6197, 5.9437

P2.21 (a) x alternates

(b) x = 0.2541;

(c) x0=1, x alternates

x0=0.5, x = 0.4472;

x0=0, f'(x0)=0; method fails.

(d) x = 0.8706

P2.23 x = 2.0288, 4.9132, 7.9787

P2.25 x = -0.70347

P2.27 (a) x = 2.4781, 3

(b) x = 2.7368, 2.7

P2.29 x = 0.12313