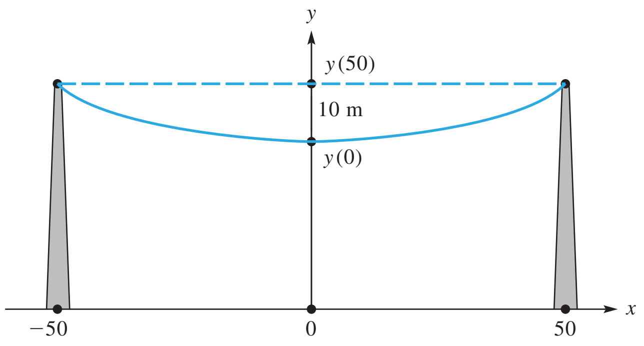
3 Locating Roots of Equations

An electric power cable is suspended (at points of equal height) from two towers that are 100 meters apart. The cable is allowed to dip 10 meters in the middle. How long is the cable?



It is known that the curve assumed by a suspended cable is a **catenary**. When the y-axis passes through the lowest point, we can assume an equation of the form $y = \lambda \cosh(x/\lambda)$. Here λ is a parameter to be determined. The conditions of the problem are that $y(50) = y(0) + 10$. Hence, we obtain

$$\lambda \cosh\left(\frac{50}{\lambda}\right) = \lambda + 10$$

By the methods of this chapter, the parameter is found to be $\lambda = 126.632$. After this value is substituted into the arc length formula of the catenary, the length is determined to be 102.619 meters. (See Computer Problem 5.1.4.)

3.1 Bisection Method

Introduction

Let f be a real- or complex-valued function of a real or complex variable. A number r, real or complex, for which $f(r) = 0$ is called a **root** of that equation or a **zero** of f. For example, the function

$$f(x) = 6x^2 - 7x + 2$$

has $\frac{1}{2}$ and $\frac{2}{3}$ as zeros, as can be verified by direct substitution or by writing f in its factored form:

$$f(x) = (2x - 1)(3x - 2)$$

For another example, the function

$$g(x) = \cos 3x - \cos 7x$$

has not only the obvious zero $x = 0$, but every integer multiple of $\pi/5$ and of $\pi/2$ as well, which we discover by applying the trigonometric identity

$$\cos A - \cos B = 2\sin\left[\frac{1}{2}(a+b)\right]\sin\left[\frac{1}{2}(b-a)\right]$$

Consequently, we find

$$g(x) = 2\sin(5x)\sin(2x)$$

Why is locating roots important? Frequently, the solution to a scientific problem is a number about which we have little information other than that it satisfies some equation. Since every equation can be written so that a function stands on one side and zero on the other, the desired number must be a zero of the function. Thus, if we possess an arsenal of methods for locating zeros of functions, we shall be able to solve such problems.

We illustrate this claim by use of a specific engineering problem whose solution is the root of an equation. In a certain electrical circuit, the voltage V and current I are related by two equations of the form

$$\begin{cases} I = a(e^{bV} - 1) \\ c = dI + V \end{cases}$$

in which a, b, c, and d are constants. For our purpose, these four numbers are assumed to be known. When these equations are combined by eliminating I between them, the result is a single equation:

$$c = ad(e^{bV} - 1) + V$$

In a concrete case, this might reduce to

$$12 = 14.3(e^{2V} - 1) + V$$

and its solution is required. (It turns out that $V \approx 0.299$ in this case.)

In some problems in which a root of an equation is sought, we can perform the required calculation with a hand calculator. But how can we locate zeros of complicated functions such as these?

$$\begin{aligned} f(x) &= 3.24x^8 - 2.42x^7 + 10.34x^6 + 11.01x^2 + 47.98 \\ g(x) &= 2^{x^2} - 10x + 1 \\ h(x) &= \cosh\left(\sqrt{x^2+1} - e^x\right) + \log|\sin x| \end{aligned}$$

What is needed is a general numerical method that does not depend on special properties of our functions. Of course, continuity and differentiability are special properties, but they are common attributes of functions that are usually encountered. The sort of special property that we probably *cannot* easily exploit in general-purpose codes is typified by the trigonometric identity mentioned previously.

Hundreds of methods are available for locating zeros of functions, and three of the most useful have been selected for study here: the bisection method, Newton's method, and the secant method.

Let f be a function that has values of opposite sign at the two ends of an interval. Suppose also that f is continuous on that interval. To fix the notation, let $a < b$ and $f(a)f(b) < 0$. It then follows that f has a root in the interval (a, b). In other words, there must exist a number r that satisfies the two conditions $a < r < b$ and $f(r) = 0$. How is this conclusion reached? One must recall the **Intermediate-Value Theorem**.* If x traverses an interval $[a, b]$, then the values of $f(x)$ completely fill out the interval between $f(a)$ and $f(b)$. No intermediate values can be skipped. Hence, a specific function f must take on the value zero somewhere in the interval (a, b) because $f(a)$ and $f(b)$ are of opposite signs.

Bisection Algorithm and Pseudocode

The **bisection method** exploits this property of continuous functions. At each step in this algorithm, we have an interval $[a, b]$ and the values $u = f(a)$ and $v = f(b)$. The numbers u and v satisfy $uv < 0$. Next, we construct the midpoint of the interval, $c = \frac{1}{2}(a + b)$, and compute $w = f(c)$. It can happen fortuitously that $f(c) = 0$. If so, the objective of the algorithm has been fulfilled. In the usual case, $w \neq 0$, and either $wu < 0$ or $wv < 0$. (Why?) If $wu < 0$, we can be sure that a root of f exists in the interval $[a, c]$. Consequently, we store the value of c in b and w in v. If $wu > 0$, then we cannot be sure that f has a root in $[a, c]$, but since $wv < 0$, f must have a root in $[c, b]$. In this case, we store the value of c in a and w in u. In either case, the situation at the end of this step is just like that at the beginning except that the final interval is half as large as the initial interval. This step can now be repeated until the interval is satisfactorily small, say $|b - a| < \frac{1}{2} \times 10^{-6}$. At the end, the best estimate of the root would be $(a + b)/2$, where $[a, b]$ is the last interval in the procedure.

Now let us construct pseudocode to carry out this procedure. We shall not try to create a piece of high-quality software with many "bells and whistles," but we will write the pseudocode in the form of a procedure for general use. This will afford the reader an opportunity to review how a main program and one or more procedures can be connected.

As a general rule, in programming routines to locate the roots of arbitrary functions, unnecessary evaluations of the function should be avoided because a given function may be costly to evaluate in terms of computer time. Thus, any value of the function that may be needed later should be stored rather than recomputed. A careless programming of the bisection method might violate this principle.

The procedure to be constructed will operate on an arbitrary function f. An interval $[a, b]$ is also specified, and the number of steps to be taken, *nmax*, is given. Pseudocode to

*A formal statement of the **Intermediate-Value Theorem** is as follows: If the function f is continuous on the closed interval $[a, b]$, and if $f(a) \leqq y \leqq f(b)$ or $f(b) \leqq y \leqq f(a)$, then there exists a point c such that $a \leqq c \leqq b$ and $f(c) = y$.

perform *nmax* steps in the bisection algorithm follows:

```
procedure Bisection($f, a, b, nmax, \varepsilon$)
integer $n, nmax$;    real $a, b, c, fa, fb, fc, error$
$fa \leftarrow f(a)$
$fb \leftarrow f(b)$
if sign($fa$) = sign($fb$) then
    output $a, b, fa, fb$
    output "function has same signs at $a$ and $b$"
    return
end if
$error \leftarrow b - a$
for $n = 0$ to $nmax$ do
    $error \leftarrow error/2$
    $c \leftarrow a + error$
    $fc \leftarrow f(c)$
    output $n, c, fc, error$
    if $|error| < \varepsilon$ then
        output "convergence"
        return
    end if
    if sign($fa$) $\neq$ sign($fc$) then
        $b \leftarrow c$
        $fb \leftarrow fc$
    else
        $a \leftarrow c$
        $fa \leftarrow fc$
    end if
end for
end procedure Bisection
```

Many modifications are incorporated to enhance the pseudocode. For example, we use *fa*, *fb*, *fc* as mnemonics for u, v, w, respectively. Also, we illustrate some techniques of structured programming and some other alternatives, such as a test for convergence. For example, if u, v, or w is close to zero, then uv or wu may underflow. Similarly, an overflow situation may arise. A test involving the intrinsic function *sign* could be used to avoid these difficulties, such as a test that determines whether $\text{sign}(u) \neq \text{sign}(v)$. Here, the iterations terminate if they exceed *nmax* or if the error bound (discussed later in this section) is less than ε. The reader should trace the steps in the routine to see that it does what is claimed.

Examples

Now we want to illustrate how the bisection pseudocode can be used. Suppose that we have two functions, and for each, we seek a zero in a specified interval:

$$\begin{aligned} f(x) &= x^3 - 3x + 1 && \text{on } [0, 1] \\ g(x) &= x^3 - 2\sin x && \text{on } [0.5, 2] \end{aligned}$$

First, we write two procedure functions to compute $f(x)$ and $g(x)$. Then we input the initial intervals and the number of steps to be performed in a main program. Since this is a rather simple example, this information could be assigned directly in the main program or by way of statements in the subprograms rather than being read into the program. Also, depending on the computer language being used, an external or interface statement is needed to tell the compiler that the parameter f in the bisection procedure is *not* an ordinary variable with numerical values but the name of a function procedure defined externally to the main program. In this example, there would be two of these function procedures and two calls to the bisection procedure.

A call program or main program that calls the second bisection routine might be written as follows:

```
program Test_Bisection
integer $n, nmax \leftarrow 20$
real $a, b, \varepsilon \leftarrow \frac{1}{2}10^{-6}$
external function $f, g$
$a \leftarrow 0.0$
$b \leftarrow 1.0$
call $Bisection(f, a, b, nmax, \varepsilon)$
$a \leftarrow 0.5$
$b \leftarrow 2.0$
call $Bisection(g, a, b, nmax, \varepsilon)$
end program Test_Bisection

real function $f(x)$
real $x$
$f \leftarrow x^3 - 3x + 1$
end function $f$

real function $g(x)$
real $x$
$g \leftarrow x^3 - 2\sin x$
end function $g$
```

The computer results for the iterative steps of the bisection method for $f(x)$:

n	c_n	$f(c_n)$	error
0	0.5	-0.375	0.5
1	0.25	0.266	0.25
2	0.375	-7.23×10^{-2}	0.125
3	0.3125	9.30×10^{-2}	6.25×10^{-2}
4	0.34375	9.37×10^{-3}	3.125×10^{-2}
$\vdots$			
19	0.34729 67	-9.54×10^{-7}	9.54×10^{-7}
20	0.34729 62	3.58×10^{-7}	4.77×10^{-7}

Also, the results for $g(x)$ are as follows:

n	c_n	$g(c_n)$	error
0	1.25	5.52×10^{-2}	0.75
1	0.875	-0.865	0.375
2	1.0625	-0.548	0.188
3	1.15625	-0.285	9.38×10^{-2}
4	1.203125	-0.125	4.69×10^{-2}
⋮			
19	1.2361827	-4.88×10^{-6}	1.43×10^{-6}
20	1.2361834	-2.15×10^{-6}	7.15×10^{-7}

To verify these results, we use built-in procedures in mathematical software such as Matlab, Mathematica, or Maple to find the desired roots of f and g to be 0.3472963553 and 1.236183928, respectively. Since f is a polynomial, we can use a routine for finding numerical approximations to all the zeros of a polynomial function. However, when more complicated nonpolynomial functions are involved, there is generally no systematic procedure for finding all zeros. In this case, a routine can be used to search for zeros (one at a time), but we have to specify a point at which to start the search, and different starting points may result in the same or different zeros. It may be particularly troublesome to find all the zeros of a function whose behavior is unknown.

Convergence Analysis

Now let us investigate the *accuracy* with which the bisection method determines a root of a function. Suppose that f is a continuous function that takes values of opposite sign at the ends of an interval $[a_0, b_0]$. Then there is a root r in $[a_0, b_0]$, and if we use the midpoint $c_0 = (a_0 + b_0)/2$ as our estimate of r, we have

$$|r - c_0| \leqq \frac{b_0 - a_0}{2}$$

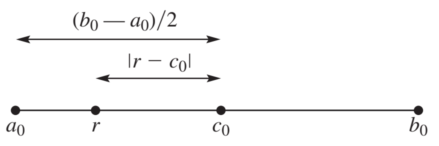
as illustrated in Figure 3.1. If the bisection algorithm is now applied and if the computed quantities are denoted by $a_0, b_0, c_0, a_1, b_1, c_1$ and so on, then by the same reasoning,

$$|r - c_n| \leqq \frac{b_n - a_n}{2} \qquad (n \geqq 0)$$

Since the widths of the intervals are divided by 2 in each step, we conclude that

$$|r - c_n| \leqq \frac{b_0 - a_0}{2^{n+1}} \tag{1}$$

FIGURE 3.1 Bisection method: Illustrating error upper bound



To summarize:

■ **THEOREM 1** **BISECTION METHOD THEOREM**

If the bisection algorithm is applied to a continuous function f on an interval $[a, b]$, where $f(a)f(b) < 0$, then, after n steps, an approximate root will have been computed with error at most $(b - a)/2^{n+1}$.

If an error tolerance has been prescribed in advance, it is possible to determine the number of steps required in the bisection method. Suppose that we want $|r - c_n| < \varepsilon$. Then it is necessary to solve the following inequality for n:

$$\frac{b-a}{2^{n+1}} < \varepsilon$$

By taking logarithms (with any convenient base), we obtain

$$n > \frac{\log(b-a) - \log(2\varepsilon)}{\log 2} \tag{2}$$

EXAMPLE 1 How many steps of the bisection algorithm are needed to compute a root of f to full machine single precision on a 32-bit word-length computer if $a = 16$ and $b = 17$?

Solution The root is between the two binary numbers $a = (10\,000.0)_2$ and $b = (10\,001.0)_2$. Thus, we already know five of the binary digits in the answer. Since we can use only 24 bits altogether, that leaves 19 bits to determine. We want the last one to be correct, so we want the error to be less than 2^{-19} or 2^{-20} (being conservative). Since a 32-bit word-length computer has a 24-bit mantissa, we can expect the answer to have an accuracy of only 2^{-20}. From the equation above, we want $(b - a)/2^{n+1} < \varepsilon$. Since $b - a = 1$ and $\varepsilon = 2^{-20}$, we have $1/2^{n+1} < 2^{-20}$. Taking reciprocals gives $2^{n+1} > 2^{20}$, or $n \geqq 20$. Alternatively, we can use Equation (2), which in this case is

$$n > \frac{\log 1 - \log 2^{-19}}{\log 2}$$

Using a basic property of logarithms ($\log x^y = y \log x$), we find that $n \geqq 20$. In this example, each step of the algorithm determines the root with one additional binary digit of precision. ■

A sequence $\{x_n\}$ exhibits **linear convergence** to a limit x if there is a constant C in the interval $[0, 1)$ such that

$$|x_{n+1} - x| \leqq C|x_n - x| \qquad (n \geqq 1) \tag{3}$$

If this inequality is true for all n, then

$$|x_{n+1} - x| \leqq C|x_n - x| \leqq C^2|x_{n-1} - x| \leqq \cdots \leqq C^n|x_1 - x|$$

Thus, it is a consequence of linear convergence that

$$|x_{n+1} - x| \leqq AC^n \qquad (0 \leqq C < 1) \tag{4}$$

The sequence produced by the bisection method obeys Inequality (4), as we see from Equation (1). However, the sequence need not obey Inequality (3).

The bisection method is the simplest way to solve a nonlinear equation $f(x) = 0$. It arrives at the root by constraining the interval in which a root lies, and it eventually makes the interval quite small. Because the bisection method halves the width of the interval at each step, one can predict exactly how long it will take to find the root within any desired degree of accuracy. In the bisection method, not every guess is closer to the root than the previous guess because the bisection method does not use the nature of the function itself. Often the bisection method is used to get close to the root before switching to a faster method.

False Position (Regula Falsi) Method and Modifications

The **false position method** retains the main feature of the bisection method: that a root is trapped in a sequence of intervals of decreasing size. Rather than selecting the midpoint of each interval, this method uses the point where the secant lines intersect the x-axis.

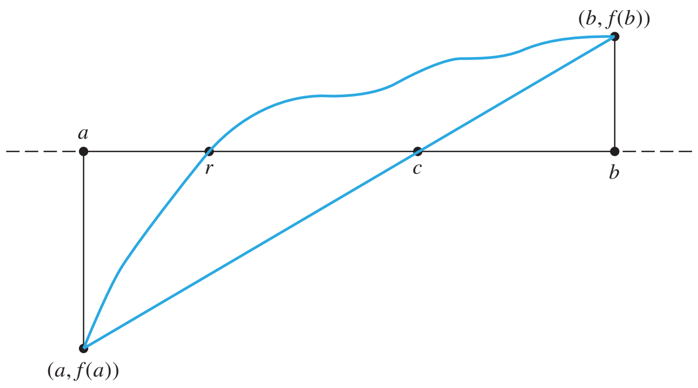


FIGURE 3.2 False position method

In Figure 3.2, the secant line over the interval $[a, b]$ is the chord between $(a, f(a))$ and $(b, f(b))$. The two right triangles in the figure are *similar*, which means that

$$\frac{b-c}{f(b)} = \frac{c-a}{-f(a)}$$

It is easy to show that

$$c = b - f(b)\left[\frac{a-b}{f(a)-f(b)}\right] = a - f(a)\left[\frac{b-a}{f(b)-f(a)}\right] = \frac{af(b)-bf(a)}{f(b)-f(a)}$$

We then compute $f(c)$ and proceed to the next step with the interval $[a, c]$ if $f(a)f(c) < 0$ or to the interval $[c, b]$ if $f(c)f(b) < 0$.

In the general case, the **false position method** starts with the interval $[a_0, b_0]$ containing a root: $f(a_0)$ and $f(b_0)$ are of opposite signs. The false position method uses intervals $[a_k, b_k]$ that contain roots in almost the same way that the bisection method does. However, instead of finding the midpoint of the interval, it finds where the secant line joining $(a_k, f(a_k))$ and $(b_k, f(b_k))$ crosses the x-axis and then selects it to be the new endpoint.

At the kth step, it computes

$$c_k = \frac{a_k f(b_k) - b_k f(a_k)}{f(b_k) - f(a_k)}$$

If $f(a_k)$ and $f(c_k)$ have the same sign, then set $a_{k+1} = c_k$ and $b_{k+1} = b_k$; otherwise, set $a_{k+1} = a_k$ and $b_{k+1} = c_k$. The process is repeated until the root is approximated sufficiently well.

For some functions, the false position method may repeatedly select the same endpoint, and the process may degrade to linear convergence. There are various approaches to rectify this. For example, when the same endpoint is to be retained twice, the **modified false position method** uses

$$c_k^{(m)} = \begin{cases} \dfrac{a_k f(b_k) - 2b_k f(a_k)}{f(b_k) - 2f(a_k)}, & \text{if } f(a_k)f(b_k) < 0 \\[2ex] \dfrac{2a_k f(b_k) - b_k f(a_k)}{2f(b_k) - f(a_k)}, & \text{if } f(a_k)f(b_k) > 0 \end{cases}$$

So rather than selecting points on the same side of the root as the regular false position method does, the modified false position method changes the slope of the straight line so that it is closer to the root. See Figure 3.3.

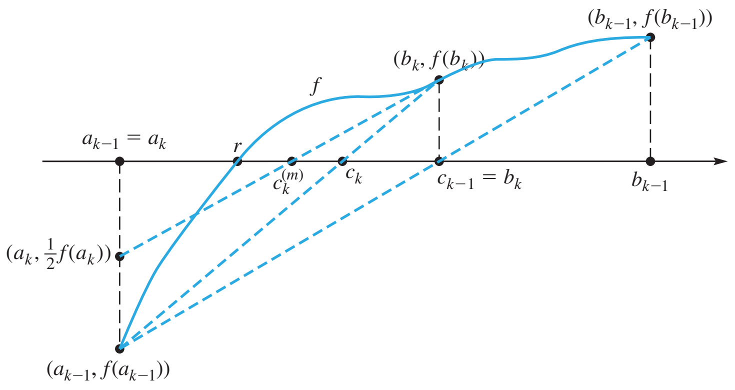


FIGURE 3.3 Modified false position method

The bisection method uses only the fact that $f(a)f(b) < 0$ for each new interval $[a, b]$, but the false position method uses the *values* of $f(a)$ and $f(b)$. This is an example showing how one can include additional information in an algorithm to build a better one. In the next section, Newton's method uses not only the function but also its first derivative.

Some variants of the modified false position procedure have superlinear convergence, which we discuss in Section 3.3. See, for example, Ford [1995]. Another modified false position method replaces the secant lines by straight lines with ever-smaller slope until the iterate falls to the opposite side of the root. (See Conte and de Boor [1980].) Early versions of the false position method date back to a Chinese mathematical text (200 B.C.E. to 100 C.E.) and an Indian mathematical text (3 B.C.E.).

Summary

(1) For finding a zero r of a given continuous function f in an interval $[a, b]$, n steps of the bisection method produce a sequence of intervals $[a, b] = [a_0, b_0]$, $[a_1, b_1]$, $[a_2, b_2]$, $\dots$, $[a_n, b_n]$ each containing the desired root of the function. The midpoints of these intervals $c_0, c_1, c_2, \dots, c_n$ form a sequence of approximations to the root, namely, $c_i = \frac{1}{2}(a_i + b_i)$. On each interval $[a_i, b_i]$, the error $e_i = r - c_i$ obeys the inequality

$$|e_i| \leqq \frac{1}{2}(b_i - a_i)$$

and after n steps we have

$$|e_n| \leqq \frac{1}{2^{n+1}}(b_0 - a_0)$$

(2) For an error tolerance ε such that $|e_n| < \varepsilon$, n steps are needed, where n satisfies the inequality

$$n > \frac{\log(b - a) - \log 2\varepsilon}{\log 2}$$

(3) For the kth step of the false position method over the interval $[a_k, b_k]$, let

$$c_k = \frac{a_k f(b_k) - b_k f(a_k)}{f(b_k) - f(a_k)}$$

If $f(a_k) f(c_k) > 0$, set $a_{k+1} = c_k$ and $b_{k+1} = b_k$; otherwise, set $a_{k+1} = a_k$ and $b_{k+1} = c_k$.

Problems 3.1

^{a}**1.** Find where the graphs of $y = 3x$ and $y = e^x$ intersect by finding roots of $e^x - 3x = 0$ correct to four decimal digits.

2. Give a graphical demonstration that the equation $\tan x = x$ has infinitely many roots. Determine one root precisely and another approximately by using a graph. *Hint:* Use the approach of the preceding problem.

3. Demonstrate graphically that the equation $50\pi + \sin x = 100 \arctan x$ has infinitely many solutions.

^{a}**4.** By graphical methods, locate approximations to all roots of the nonlinear equation $\ln(x + 1) + \tan(2x) = 0$.

5. Give an example of a function for which the bisection method does *not* converge linearly.

6. Draw a graph of a function that is discontinuous yet the bisection method converges. Repeat, getting a function for which it diverges.

7. Prove Inequality (1).

8. If $a = 0.1$ and $b = 1.0$, how many steps of the bisection method are needed to determine the root with an error of at most $\frac{1}{2} \times 10^{-8}$?

[a]**9.** Find all the roots of $f(x) = \cos x - \cos 3x$. Use two different methods.

[a]**10.** (Continuation) Find the root or roots of $\ln[(1 + x)/(1 - x^2)] = 0$.

11. If f has an inverse, then the equation $f(x) = 0$ can be solved by simply writing $x = f^{-1}(0)$. Does this remark eliminate the problem of finding roots of equations? Illustrate with $\sin x = 1/\pi$.

[a]**12.** How many binary digits of precision are gained in each step of the bisection method? How many steps are required for each decimal digit of precision?

13. Try to devise a stopping criterion for the bisection method to guarantee that the root is determined with *relative* error at most ε.

14. Denote the successive intervals that arise in the bisection method by $[a_0, b_0]$, $[a_1, b_1]$, $[a_2, b_2]$, and so on.

a. Show that $a_0 \leqq a_1 \leqq a_2 \leqq \cdots$ and that $b_0 \geqq b_1 \geqq b_2 \geqq \cdots$.

b. Show that $b_n - a_n = 2^{-n}(b_0 - a_0)$.

c. Show that, for all n, $a_n b_n + a_{n-1} b_{n-1} = a_{n-1} b_n + a_n b_{n-1}$.

15. (Continuation) Can it happen that $a_0 = a_1 = a_2 = \cdots$

16. (Continuation) Let $c_n = (a_n + b_n)/2$. Show that

$$\lim_{n\to\infty} c_n = \lim_{n\to\infty} a_n = \lim_{n\to\infty} b_n$$

[a]**17.** (Continuation) Consider the bisection method with the initial interval $[a_0, b_0]$. Show that after ten steps with this method,

$$\left| \frac{1}{2}(a_{10} + b_{10}) - \frac{1}{2}(a_9 + b_9) \right| = 2^{-11}(b_0 - a_0)$$

Also, determine how many steps are required to guarantee an approximation of a root to six decimal places (rounded).

18. (True–False) If the bisection method generates intervals $[a_0, b_0]$, $[a_1, b_1]$, and so on, which of these inequalities are true for the root r that is being calculated? Give proofs or counterexamples in each case.

a. $|r - a_n| \leqq 2|r - b_n|$ [a]**b.** $|r - a_n| \leqq 2^{-n-1}(b_0 - a_0)$

c. $|r - \frac{1}{2}(a_n + b_n)| \leqq 2^{-n-2}(b_0 - a_0)$

[a]**d.** $0 \leqq r - a_n \leqq 2^{-n}(b_0 - a_0)$ **e.** $|r - b_n| \leqq 2^{-n-1}(b_0 - a_0)$

19. (True–False) Using the notation of the text, determine which of these assertions are true and which are generally false:

[a]**a.** $|r - c_n| < |r - c_{n-1}|$ **b.** $a_n \leqq r \leqq c_n$ **c.** $c_n \leqq r \leqq b_n$

d. $|r - a_n| \leqq 2^{-n}$ [a]**e.** $|r - b_n| \leqq 2^{-n}(b_0 - a_0)$

20. Prove that $|c_n - c_{n+1}| = 2^{-n-2}(b_0 - a_0)$.

[a]**21.** If the bisection method is applied with starting interval $[a, a + 1]$ and $a \geqq 2^m$, where $m \geqq 0$, what is the correct number of steps to compute the root with full machine precision on a 32-bit word-length computer?

22. If the bisection method is applied with starting interval $[2^m, 2^{m+1}]$, where m is a positive or negative integer, how many steps should be taken to compute the root to full machine precision on a 32-bit word-length computer?

[a]**23.** Every polynomial of degree n has n zeros (counting multiplicities) in the complex plane. Does every real polynomial have n real zeros? Does every *polynomial of infinite degree* $f(x) = \sum_{n=0}^{\infty} a_n x^n$ have infinitely many zeros?

Computer Problems 3.1

1. Using the bisection method, determine the point of intersection of the curves given by $y = x^3 - 2x + 1$ and $y = x^2$.

2. Find a root of the following equation in the interval $[0, 1]$ by using the bisection method: $9x^4 + 18x^3 + 38x^2 - 57x + 14 = 0$.

3. Find a root of the equation $\tan x = x$ on the interval $[4, 5]$ by using the bisection method. What happens on the interval $[1, 2]$?

4. Find a root of the equation $6(e^x - x) = 6 + 3x^2 + 2x^3$ between -1 and $+1$ using the bisection method.

5. Use the bisection method to find a zero of the equation $\lambda \cosh(50/\lambda) = \lambda + 10$ that begins this chapter.

6. Program the bisection method as a recursive procedure and test it on one or two of the examples in the text.

7. Use the bisection method to determine roots of these functions on the intervals indicated. Process all three functions in *one* computer run.

$$\begin{aligned} f(x) &= x^3 + 3x - 1 && \text{on } [0, 1] \\ g(x) &= x^3 - 2\sin x && \text{on } [0.5, 2] \\ h(x) &= x + 10 - x\cosh(50/x) && \text{on } [120, 130] \end{aligned}$$

Find each root to full machine precision. Use the correct number of steps, at least approximately. Repeat using the false position method.

8. Test the three bisection routines on $f(x) = x^3 + 2x^2 + 10x - 20$, with $a = 1$ and $b = 2$. The zero is 1.36880 8108. In programming this polynomial function, use nested multiplication. Repeat using the modified false position method.

9. Write a program to find a zero of a function f in the following way: In each step, an interval $[a, b]$ is given and $f(a)f(b) < 0$. Then c is computed as the root of the linear function that agrees with f at a and b. We retain either $[a, c]$ or $[c, b]$, depending on whether $f(a)f(c) < 0$ or $f(c)f(b) < 0$. Test your program on several functions.

[a]**10.** Select a routine from your program library to solve polynomial equations and use it to find the roots of the equation

$$x^8 - 36x^7 + 546x^6 - 4536x^5 + 22449x^4 - 67284x^3 \\ +118124x^2 - 109584x + 40320 = 0$$

The correct roots are the integers $1, 2, \ldots, 8$. Next, solve the same equation when the coefficient of x^7 is changed to -37. Observe how a minor perturbation in the coefficients can cause massive changes in the roots. Thus, the roots are **unstable** functions of the coefficients. (Be sure to program the problem to allow for complex roots.) *Cultural Note:* This is a simplified version of **Wilkinson's polynomial**, which is found in Computer Problem 3.3.9.

[a]**11.** A **circular metal shaft** is being used to transmit power. It is known that at a certain critical angular velocity ω, any jarring of the shaft during rotation will cause the shaft to deform or buckle. This is a dangerous situation because the shaft might shatter under the increased centrifugal force. To find this critical velocity ω, we must first compute a number x that satisfies the equation

$$\tan x + \tanh x = 0$$

This number is then used in a formula to obtain ω. Solve for x $(x > 0)$.

12. Using built-in routines in mathematical software systems such as Matlab, Mathematica, or Maple, find the roots for $f(x) = x^3 - 3x + 1$ on $[0, 1]$ and $g(x) = x^3 - \sin x$ on $[0.5, 2]$ to more digits of accuracy than shown in the text.

13. (**Engineering problem**) Nonlinear equations occur in almost all fields of engineering. For example, suppose a given task is expressed in the form $f(x) = 0$ and the objective is to find values of x that satisfy this condition. It is often difficult to find an explicit solution and an approximate solution is sought with the aid of mathematical software. Find a solution of

$$f(x) = \frac{1}{\sqrt{2\pi}} e^{-(1/2)x^2} + \frac{1}{10}\sin(\pi x)$$

Plot the curve in the range $[-3.5, 3.5]$ for x values and $[-0.5, 0.5]$ for $y = f(x)$ values.

14. (**Circuit problem**) A simple circuit with resistance R, capacitance C in series with a battery of voltage V is given by $Q = CV[1 - e^{-T/(RC)}]$, where Q is the charge of the capacitor and T is the time needed to obtain the charge. We wish to solve for the unknown C. For example, solve this problem

$$f(x) = \left[10x\left(1 - e^{-0.004/(2000x)}\right) - 0.00001\right]$$

Plot the curve. *Hint:* You may wish to magnify the vertical scale by using $y = 10^5 f(x)$.

15. (**Engineering polynomials**) Equations such as $A + Bx^2e^{Cx} = 0$ and $A + Bx + Cx^2 + Dx^3 + Ex^4 = 0$ occur in engineering problems. Using mathematical software, find one or more solutions to the following equations and plot their curves:

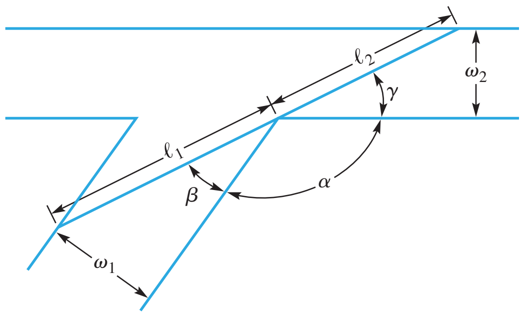
a. $2 - x^2e^{-0.385x} = 0$ **b.** $1 - 32x + 160x^2 - 256x^3 + 128x^4 = 0$

16. (**Reinforced concrete**) In the design of reinforced concrete with regard to stress, one needs to solve numerically a quadratic equation such as

$$2414707.2x[450 - 0.822x(225)] - 265{,}000{,}000 = 0$$

Find approximate values of the roots.

17. (**Board in hall problem**) In a building, two intersecting halls with widths $w_1 = 9$ feet and $w_2 = 7$ feet meet at an angle $\alpha = 125°$, as shown:



Assuming a two-dimensional situation, what is the longest board that can negotiate the turn? Ignore the thickness of the board. The relationship between the angles θ and the length of the board $\ell = \ell_1 + \ell_2$ is $\ell_1 = w_1 \csc(\beta)$, $\ell_2 = w_2 \csc(\gamma)$, $\beta = \pi - \alpha - \gamma$ and $\ell = w_1 \csc(\pi - \alpha - \gamma) + w_2 \csc(\gamma)$. The maximum length of the board that can make the turn is found by minimizing ℓ as a function of γ. Taking the derivative and setting $d\ell/d\gamma = 0$, we obtain

$$w_1 \cot(\pi - \alpha - \gamma) \csc(\pi - \alpha - \gamma) - w_2 \cot(\gamma) \csc(\gamma) = 0$$

Substitute in the known values and numerically solve the nonlinear equation. This problem is similar to an example in Gerald and Wheatley [1999].

18. Find the rectangle of maximum area if its vertices are at $(0, 0)$, $(x, 0)$, $(x, \cos x)$, $(0, \cos x)$. Assume that $0 \leqq x \leqq \pi/2$.

19. Program the false position algorithm and test it on some examples such as some of the nonlinear problems in the text or in the computer problems. Compare your results with those given for the bisection method.

20. Program the modified false position method, test it, and compare it to the false position method when using some sample functions.

3.2 Newton's Method

The procedure known as Newton's method is also called the **Newton-Raphson iteration**. It has a more general form than the one seen here, and the more general form can be used to find roots of systems of equations. Indeed, it is one of the more important procedures in numerical analysis, and its applicability extends to differential equations and integral equations. Here it is being applied to a single equation of the form $f(x) = 0$. As before, we seek one or more points at which the value of the function f is zero.

Interpretations of Newton's Method

In Newton's method, it is assumed at once that the function f is differentiable. This implies that the graph of f has a definite *slope* at each point and hence a unique tangent line. Now let us pursue the following simple idea. At a certain point $(x_0, f(x_0))$ on the graph of f, there is a tangent, which is a rather good approximation to the curve in the vicinity of that point. Analytically, it means that the linear function

$$l(x) = f'(x_0)(x - x_0) + f(x_0)$$

is close to the given function f near x_0. At x_0, the two functions l and f agree. We take the zero of l as an approximation to the zero of f. The zero of l is easily found:

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}$$

Thus, starting with point x_0 (which we may interpret as an approximation to the root sought), we pass to a new point x_1 obtained from the preceding formula. Naturally, the process can be repeated (iterated) to produce a sequence of points:

$$x_2 = x_1 - \frac{f(x_1)}{f'(x_1)}, \qquad x_3 = x_2 - \frac{f(x_2)}{f'(x_2)}, \qquad \text{etc.}$$

Under favorable conditions, the sequence of points will approach a zero of f.

The geometry of Newton's method is shown in Figure 3.4. The line $y = l(x)$ is tangent to the curve $y = f(x)$. It intersects the x-axis at a point x_1. The slope of $l(x)$ is $f'(x_0)$.

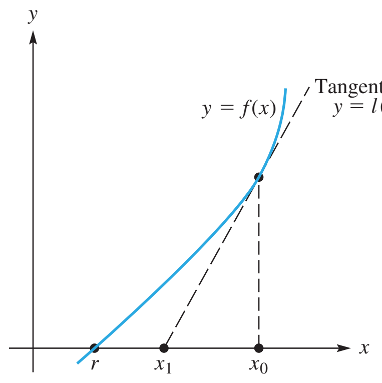


FIGURE 3.4 Newton's method

There are other ways of interpreting Newton's method. Suppose again that x_0 is an initial approximation to a root of f. We ask: *What* correction h *should be added to* x_0 *to obtain the root precisely*? Obviously, we want

$$f(x_0 + h) = 0$$

If f is a sufficiently well-behaved function, it will have a Taylor series at x_0 [see Equation (11) in Section 1.2]. Thus, we could write

$$f(x_0) + hf'(x_0) + \frac{h^2}{2}f''(x_0) + \cdots = 0$$

Determining h from this equation is, of course, not easy. Therefore, we give up the expectation of arriving at the true root in one step and seek only an approximation to h. This can be obtained by ignoring all but the first two terms in the series:

$$f(x_0) + hf'(x_0) = 0$$

The h that solves this is *not* the h that solves $f(x_0 + h) = 0$, but it is the easily computed number

$$h = -\frac{f(x_0)}{f'(x_0)}$$

Our new approximation is then

$$x_1 = x_0 + h = x_0 - \frac{f(x_0)}{f'(x_0)}$$

and the process can be repeated. In retrospect, we see that the Taylor series was not needed after all because we used only the first two terms. In the analysis to be given later, it is assumed that f'' is continuous in a neighborhood of the root. This assumption enables us to estimate the errors in the process.

If Newton's method is described in terms of a sequence $x_0, x_1, \ldots,$ then the following **recursive** or **inductive** definition applies:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

Naturally, the interesting question is whether

$$\lim_{n\to\infty} x_n = r$$

where r is the desired root.

EXAMPLE 1 If $f(x) = x^3 - x + 1$ and $x_0 = 1$, what are x_1 and x_2 in the Newton iteration?

Solution From the basic formula, $x_1 = x_0 - f(x_0)/f'(x_0)$. Now $f'(x) = 3x^2 - 1$, and so $f'(1) = 2$. Also, we find $f(1) = 1$. Hence, we have $x_1 = 1 - \frac{1}{2} = \frac{1}{2}$. Similarly, we obtain $f\left(\frac{1}{2}\right) = \frac{5}{8}$, $f'\left(\frac{1}{2}\right) = -\frac{1}{4}$, and $x_2 = 3$. ■

Pseudocode

A pseudocode for Newton's method can be written as follows:

```
procedure Newton($f$, $f'$, $x$, $nmax$, $\varepsilon$, $\delta$)
integer $n$, $nmax$;    real $x$, $fx$, $fp$, $\varepsilon$, $\delta$
external function $f$, $f'$
$fx \leftarrow f(x)$
output 0, $x$, $fx$
for $n = 1$ to $nmax$ do
    $fp \leftarrow f'(x)$
    if $|fp| < \delta$ then
        output "small derivative"
        return
    end if
    $d \leftarrow fx/fp$
    $x \leftarrow x - d$
    $fx \leftarrow f(x)$
    output $n$, $x$, $fx$
    if $|d| < \varepsilon$ then
        output "convergence"
        return
    end if
end for
end procedure Newton
```

Using the initial value of x as the starting point, we carry out a maximum of $nmax$ iterations of Newton's method. Procedures must be supplied for the external functions $f(x)$ and $f'(x)$. The parameters ε and δ are used to control the convergence and are related to the accuracy desired or to the machine precision available.

Illustration

Now we illustrate Newton's method by locating a root of $x^3 + x = 2x^2 + 3$. We apply the method to the function $f(x) = x^3 - 2x^2 + x - 3$, starting with $x_0 = 3$. Of course, $f'(x) = 3x^2 - 4x + 1$, and these two functions should be arranged in nested form for efficiency:

$$f(x) = ((x - 2)x + 1)x - 3$$
$$f'(x) = (3x - 4)x + 1$$

To see in greater detail the rapid convergence of Newton's method, we use arithmetic with double the normal precision in the program and obtain the following results:

n	x_n	$f(x_n)$
0	3.0	9.0
1	2.4375	2.04
2	2.21303 27224 73144 5	0.256
3	2.17555 49386 14368 4	6.46×10^{-3}
4	2.17456 01006 55071 4	4.48×10^{-6}
5	2.17455 94102 93284 1	1.97×10^{-12}

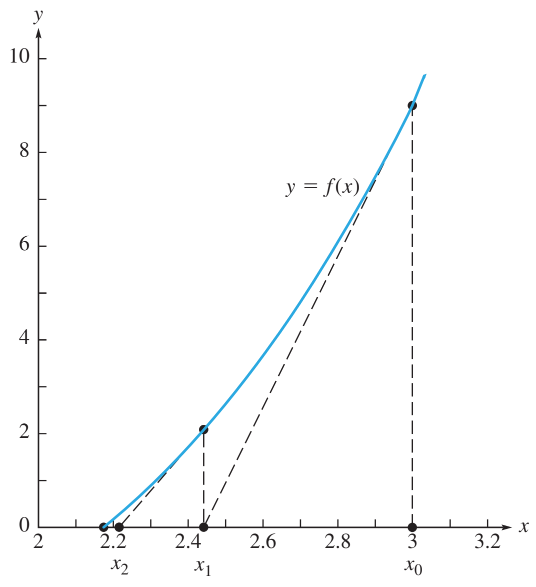


FIGURE 3.5 Three steps of Newton's method $f(x) = x^3 - 2x^2 + x - 3$

Notice the doubling of the accuracy in $f(x)$ (and also in x) until the maximum precision of the computer is encountered. Figure 3.5 shows a computer plot of three iterations of Newton's method for this sample problem.

Using mathematical software that allows for complex roots such as in Matlab, Maple, or Mathematica, we find that the polynomial has a single real root, 2.17456, and a pair of complex conjugate roots, $-0.0872797 \pm 1.17131i$.

Convergence Analysis

Anyone who has experimented with Newton's method—for instance, by working some of the problems in this section—will have observed the remarkable rapidity in the convergence of the sequence to the root. This phenomenon is also noticeable in the example just given. Indeed, the number of correct figures in the answer is nearly *doubled* at each successive step. Thus in the example above, we have first 0 and then 1, 2, 3, 6, 12, 24, ... accurate digits from each Newton iteration. Five or six steps of Newton's method often suffice to yield full machine precision in the determination of a root. There is a theoretical basis for this dramatic performance, as we shall now see.

Let the function f, whose zero we seek, possess two continuous derivatives f' and f'', and let r be a zero of f. Assume further that r is a **simple zero**; that is, $f'(r) \neq 0$. Then Newton's method, if started sufficiently close to r, **converges quadratically** to r. This means that the errors in successive steps obey an inequality of the form

$$|r - x_{n+1}| \leqq c|r - x_n|^2$$

We shall establish this fact presently, but first, an informal interpretation of the inequality may be helpful.

Suppose, for simplicity, that $c = 1$. Suppose also that x_n is an estimate of the root r that differs from it by at most one unit in the kth decimal place. This means that

$$|r - x_n| \leqq 10^{-k}$$

The two inequalities above imply that

$$|r - x_{n+1}| \leqq 10^{-2k}$$

In other words, x_{n+1} differs from r by at most one unit in the $(2k)$th decimal place. So x_{n+1} has approximately twice as many correct digits as x_n! This is the doubling of significant digits alluded to previously.

THEOREM 1 **NEWTON'S METHOD THEOREM**

If f, f', and f'' are continuous in a neighborhood of a root r of f and if $f'(r) \neq 0$, then there is a positive δ with the following property: If the initial point in Newton's method satisfies $|r - x_0| \leqq \delta$, then all subsequent points x_n satisfy the same inequality, converge to r, and do so quadratically; that is,

$$|r - x_{n+1}| \leqq c(\delta)|r - x_n|^2$$

where $c(\delta)$ is given by Equation (2) below.

Proof To establish the quadratic convergence of Newton's method, let $e_n = r - x_n$. The formula that defines the sequence $\{x_n\}$ then gives

$$e_{n+1} = r - x_{n+1} = r - x_n + \frac{f(x_n)}{f'(x_n)} = e_n + \frac{f(x_n)}{f'(x_n)} = \frac{e_n f'(x_n) + f(x_n)}{f'(x_n)}$$

By Taylor's Theorem (see Section 1.2), there exists a point ξ_n situated between x_n and r for which

$$0 = f(r) = f(x_n + e_n) = f(x_n) + e_n f'(x_n) + \frac{1}{2}e_n^2 f''(\xi_n)$$

(The subscript on ξ_n emphasizes the dependence on x_n.) This last equation can be rearranged to read

$$e_n f'(x_n) + f(x_n) = -\frac{1}{2}e_n^2 f''(\xi_n)$$

and if this is used in the previous equation for e_{n+1}, the result is

$$e_{n+1} = -\frac{1}{2}\left(\frac{f''(\xi_n)}{f'(x_n)}\right)e_n^2 \tag{1}$$

This is, at least qualitatively, the sort of equation we want. Continuing the analysis, we define a function

$$c(\delta) = \frac{1}{2}\,\frac{\max\limits_{|x-r|\leqq\delta} |f''(x)|}{\min\limits_{|x-r|\leqq\delta} |f'(x)|} \qquad (\delta > 0) \tag{2}$$

By virtue of this definition, we can assert that, for any two points x and ξ within distance δ of the root r, the inequality $\frac{1}{2}|f''(\xi)/f'(x)| \leqq c(\delta)$ is true. Now select δ so small that $\delta c(\delta) < 1$. This is possible because as δ approaches 0, $c(\delta)$ converges to $\frac{1}{2}|f''(r)/f'(r)|$, and so $\delta c(\delta)$ converges to 0. Recall that we assumed that $f'(r) \neq 0$. Let $\rho = \delta c(\delta)$. In the remainder of this argument, we hold δ, $c(\delta)$, and ρ fixed with $\rho < 1$.

Suppose now that some iterate x_n lies within distance δ from the root r. We have

$$|e_n| = |r - x_n| \leqq \delta \qquad \text{and} \qquad |\xi_n - r| \leqq \delta$$

By the definition of $c(\delta)$, it follows that $\frac{1}{2}|f''(\xi_n)|/|f'(x_n)| \leqq c(\delta)$. From Equation (1), we now have

$$|e_{n+1}| = \frac{1}{2}\left|\frac{f''(\xi_n)}{f'(x_n)}\right| e_n^2 \leqq c(\delta)e_n^2 \leqq \delta c(\delta)|e_n| = \rho|e_n|$$

Consequently, x_{n+1} is also within distance δ of r because

$$|r - x_{n+1}| = |e_{n+1}| \leqq \rho|e_n| \leqq |e_n| \leqq \delta$$

If the initial point x_0 is chosen within distance δ of r, then

$$|e_n| \leqq \rho|e_{n-1}| \leqq \rho^2|e_{n-1}| \leqq \cdots \leqq \rho^n|e_0|$$

Since $0 < \rho < 1$, $\lim_{n\to\infty} \rho^n = 0$ and $\lim_{n\to\infty} e_n = 0$. In other words, we obtain

$$\lim_{n\to\infty} x_n = r$$

In this process, we have $|e_{n+1}| \leqq c(\delta)e_n^2$. ■

In the use of Newton's method, consideration must be given to the proper choice of a starting point. Usually, one must have some insight into the shape of the graph of the function. Sometimes a coarse graph is adequate, but in other cases, a step-by-step evaluation of the function at various points may be necessary to find a point near the root. Often several steps of the bisection method is used initially to obtain a suitable starting point, and Newton's method is used to improve the precision.

Although Newton's method is truly a marvelous invention, its convergence depends upon hypotheses that are difficult to verify a priori. Some graphical examples will show what can happen. In Figure 3.6(a), the tangent to the graph of the function f at x_0 intersects the x-axis at a point remote from the root r, and successive points in Newton's iteration *recede*

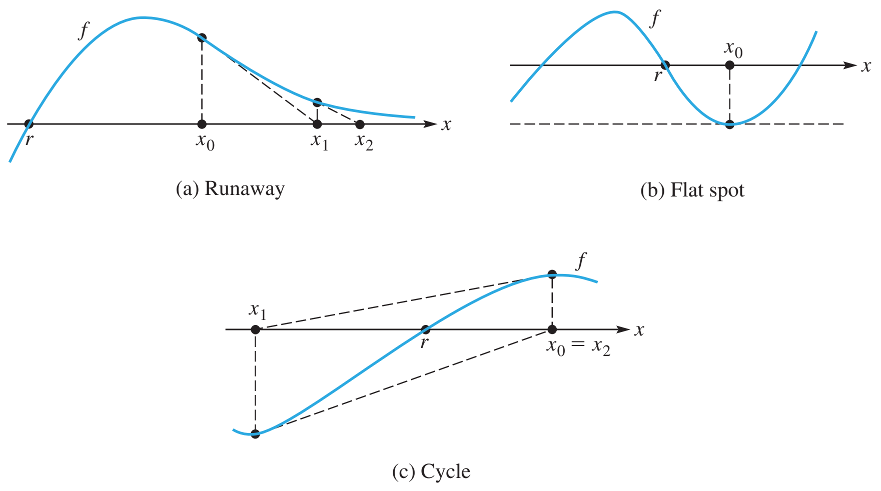


FIGURE 3.6 Failure of Newton's method due to bad starting points

from r instead of converging to r. The difficulty can be ascribed to a poor choice of the initial point x_0; it is *not* sufficiently close to r. In Figure 3.6(b), the tangent to the curve is parallel to the x-axis and $x_1 = \pm\infty$, or it is assigned the value of machine infinity in a computer. In Figure 3.6(c), the iteration values *cycle* because $x_2 = x_0$. In a computer, roundoff errors or limited precision may eventually cause this situation to become unbalanced such that the iterates either spiral inward and converge or spiral outward and diverge.

The analysis that establishes the quadratic convergence discloses another troublesome hypothesis; namely, $f'(r) \neq 0$. If $f'(r) = 0$, then r is a zero of f and f'. Such a zero is termed a **multiple zero** of f—in this case, at least a double zero. Newton's iteration for a multiple zero converges only linearly! Ordinarily, one would not know in advance that the zero sought was a multiple zero. If one knew that the multiplicity was m, however, Newton's method could be accelerated by modifying the equation to read

$$x_{n+1} = x_n - m\frac{f(x_n)}{f'(x_n)}$$

in which m is the *multiplicity* of the zero in question. The **multiplicity** of the zero r is the least m such that $f^{(k)}(r) = 0$ for $0 \leqq k < m$, but $f^{(m)}(r) \neq 0$. (See Problem 3.2.35.)

As is shown in Figure 3.7, the equation $p_2(x) = x^2 - 2x + 1 = 0$ has a root at 1 of multiplicity 2, and the equation $p_3(x) = x^3 - 3x^2 + 3x - 1 = 0$ has a root at 1 of multiplicity 3. It is instructive to plot these curves. Both curves are rather flat at the roots, which slows down the convergence of the regular Newton's method. Also, the figures illustrate the curves of two nonlinear functions with multiplicities as well as their regions of uncertainty about the curves. So the computed solutions could be anywhere within the indicated intervals on the x-axis. This is an indication of the difficulty in obtaining precise solutions of nonlinear functions with multiplicities.

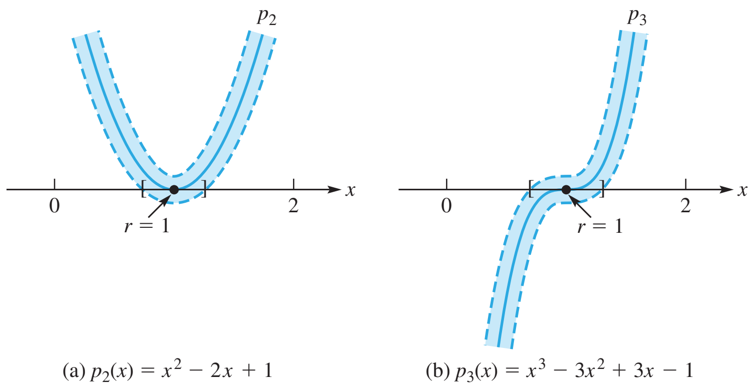


FIGURE 3.7 Curves p_2 and p_3 with multiplicity 2 and 3

Systems of Nonlinear Equations

Some physical problems involve the solution of systems of N nonlinear equations in N unknowns. One approach is to **linearize** and **solve**, repeatedly. This is the same strategy used by Newton's method in solving a single nonlinear equation. Not surprisingly, a natural extension of Newton's method for nonlinear systems can be found. The topic of systems of nonlinear equations requires some familiarity with matrices and their inverses. (See Appendix D.)

In the general case, a system of N nonlinear equations in N unknowns x_i can be displayed in the form

$$\begin{cases} f_1(x_1, x_2, \ldots, x_N) &= 0 \\ f_2(x_1, x_2, \ldots, x_N) &= 0 \\ & \vdots \\ f_N(x_1, x_2, \ldots, x_N) &= 0 \end{cases}$$

Using vector notation, we can write this system in a more elegant form:

$$\mathbf{F}(\mathbf{X}) = \mathbf{0}$$

by defining column vectors as

$$\mathbf{F} = [f_1, f_2, \ldots, f_N]^T$$
$$\mathbf{X} = [x_1, x_2, \ldots, x_N]^T$$

The extension of Newton's method for nonlinear systems is

$$\mathbf{X}^{(k+1)} = \mathbf{X}^{(k)} - \left[\mathbf{F}'\left(\mathbf{X}^{(k)}\right)\right]^{-1} \mathbf{F}\left(\mathbf{X}^{(k)}\right)$$

where $\mathbf{F}'\left(\mathbf{X}^{(k)}\right)$ is the **Jacobian matrix**, which will be defined presently. It comprises partial derivatives of $\mathbf{F}$ evaluated at $\mathbf{X}^{(k)} = \left[x_1^{(k)}, x_2^{(k)}, \ldots, x_N^{(k)}\right]^T$. This formula is similar to the previously seen version of Newton's method except that the derivative expression is not in the denominator but in the numerator as the inverse of a matrix. In the computational form of the formula, $\mathbf{X}^{(0)} = \left[x_1^{(0)}, x_2^{(0)}, \ldots, x_N^{(0)}\right]^T$ is an initial approximation vector, taken to be close to the solution of the nonlinear system, and the inverse of the Jacobian matrix is not computed but rather a related system of equations is solved.

We illustrate the development of this procedure using three nonlinear equations

$$\begin{cases} f_1(x_1, x_2, x_3) &= 0 \\ f_2(x_1, x_2, x_3) &= 0 \\ f_3(x_1, x_2, x_3) &= 0 \end{cases} \tag{3}$$

Recall the Taylor expansion in three variables for $i = 1, 2, 3$:

$$f_i(x_1 + h_1, x_2 + h_2, x_3 + h_3) = f_i(x_1, x_2, x_3) + h_1 \frac{\partial f_i}{\partial x_1} + h_2 \frac{\partial f_i}{\partial x_2} + h_3 \frac{\partial f_i}{\partial x_3} + \cdots \tag{4}$$

where the partial derivatives are evaluated at the point (x_1, x_2, x_3). Here only the linear terms in step sizes h_i are shown. Suppose that the vector $\mathbf{X}^{(0)} = \left(x_1^{(0)}, x_2^{(0)}, x_3^{(0)}\right)^T$ is an approximate solution to (3). Let $\mathbf{H} = \left[h_1, h_2, h_3\right]^T$ be a computed **correction** to the initial guess so that $\mathbf{X}^{(0)} + \mathbf{H} = \left[x_1^{(0)} + h_1, x_2^{(0)} + h_2, x_3^{(0)} + h_3\right]^T$ is a better approximate solution. Discarding the higher-order terms in the Taylor expansion (4), we have in vector notation

$$\mathbf{0} \approx \mathbf{F}\left(\mathbf{X}^{(0)} + \mathbf{H}\right) \approx \mathbf{F}\left(\mathbf{X}^{(0)}\right) + \mathbf{F}'\left(\mathbf{X}^{(0)}\right)\mathbf{H} \tag{5}$$

where the **Jacobian matrix** is defined by

$$\mathbf{F}'\left(\mathbf{X}^{(0)}\right) = \begin{bmatrix} \dfrac{\partial f_1}{\partial x_1} & \dfrac{\partial f_1}{\partial x_2} & \dfrac{\partial f_1}{\partial x_3} \\ \dfrac{\partial f_2}{\partial x_1} & \dfrac{\partial f_2}{\partial x_2} & \dfrac{\partial f_2}{\partial x_3} \\ \dfrac{\partial f_3}{\partial x_1} & \dfrac{\partial f_3}{\partial x_2} & \dfrac{\partial f_3}{\partial x_3} \end{bmatrix}$$

Here all of the partial derivatives are evaluated at $\mathbf{X}^{(0)}$; namely,

$$\frac{\partial f_i}{\partial x_j} = \frac{\partial f_i\left(\mathbf{X}^{(0)}\right)}{\partial x_j}$$

Also, we assume that the Jacobian matrix $\mathbf{F}'\left(\mathbf{X}^{(0)}\right)$ is nonsingular, so its inverse exists. Solving for $\mathbf{H}$ in (5), we have

$$\mathbf{H} \approx -\left[\mathbf{F}'\left(\mathbf{X}^{(0)}\right)\right]^{-1}\mathbf{F}\left(\mathbf{X}^{(0)}\right)$$

Let $\mathbf{X}^{(1)} = \mathbf{X}^{(0)} + \mathbf{H}$ be the better approximation after the correction; we then arrive at the first iteration of Newton's method for nonlinear systems

$$\mathbf{X}^{(1)} = \mathbf{X}^{(0)} - \left[\mathbf{F}'\left(\mathbf{X}^{(0)}\right)\right]^{-1}\mathbf{F}\left(\mathbf{X}^{(0)}\right)$$

In general, Newton's method uses this iteration:

$$\mathbf{X}^{(k+1)} = \mathbf{X}^{(k)} - \left[\mathbf{F}'\left(\mathbf{X}^{(k)}\right)\right]^{-1}\mathbf{F}\left(\mathbf{X}^{(k)}\right)$$

In practice, the computational form of Newton's method does not involve inverting the Jacobian matrix but rather solves the **Jacobian linear systems**

$$\left[\mathbf{F}'\left(\mathbf{X}^{(k)}\right)\right]\mathbf{H}^{(k)} = -\mathbf{F}\left(\mathbf{X}^{(k)}\right) \tag{6}$$

The next iteration of Newton's method is then

$$\mathbf{X}^{(k+1)} = \mathbf{X}^{(k)} + \mathbf{H}^{(k)} \tag{7}$$

This is **Newton's method for nonlinear systems**. The linear system (6) can be solved by procedures *Gauss* and *Solve* as discussed in Chapter 7. Small systems of order 2 can be solved easily. (See Problem 3.2.39.)

EXAMPLE 2 As an illustration, we can write a pseudocode to solve the following nonlinear system of equations using a variant of Newton's method given by (6) and (7):

$$\begin{cases} x + y + z = 3 \\ x^2 + y^2 + z^2 = 5 \\ e^x + xy - xz = 1 \end{cases} \tag{8}$$

Solution With a sharp eye, the reader immediately sees that the solution of this system is $x = 0$, $y = 1$, $z = 2$. But in most realistic problems, the solution is not so obvious. We wish to develop a numerical procedure for finding such a solution. Here is a pseudocode:

```
$\mathbf{X} = [0.1, \quad 1.2, \quad 2.5]^T$
for $k = 1$ to $10$ do
    $\mathbf{F} = \begin{bmatrix} x_1 + x_2 + x_3 - 3 \\ x_1^2 + x_2^2 + x_3^2 - 5 \\ e^{x_1} + x_1x_2 - x_1x_3 - 1 \end{bmatrix}$
    $\mathbf{J} = \begin{bmatrix} 1 & 1 & 1 \\ 2x_1 & 2x_2 & 2x_3 \\ e^{x_1} + x_2 - x_3 & x_1 & -x_1 \end{bmatrix}$
    solve $\mathbf{JH} = \mathbf{F}$
    $\mathbf{X} = \mathbf{X} - \mathbf{H}$
end for
```

When programmed and executed on a computer, we found that it converges to $\boldsymbol{x} = (0, 1, 2)$, but when we change to a different starting vector, $(1, 0, 1)$, it converges to another root, $(1.2244, -0.0931, 1.8687)$. (Why?) ■

We can use mathematical software such as in Matlab, Maple, or Mathematica and their built-in procedures for solving the system of nonlinear equations (8). The important application area of solving systems of nonlinear equations is used in Chapter 16 on minimization of functions.

Fractal Basins of Attraction

The applicability of Newton's method for finding complex roots is one of its outstanding strengths. One need only program Newton's method using complex arithmetic.

The frontiers of numerical analysis and nonlinear dynamics overlap in some intriguing ways. Computer-generated displays with fractal patterns, such as in Figure 3.8, can easily be created with the help of the Newton iteration. The resulting pictures show intricately

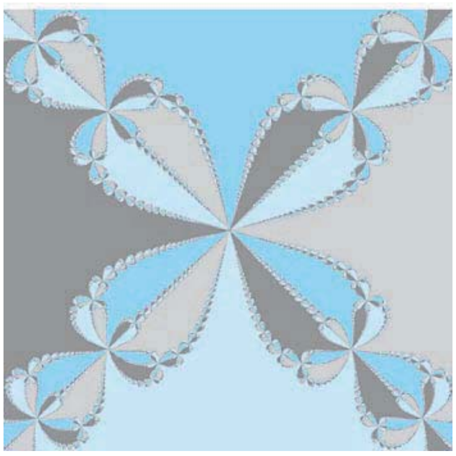


FIGURE 3.8 Basins of attraction

interwoven sets in the plane that are quite beautiful if displayed on a color computer monitor. One begins with a polynomial in the complex variable z. For example, $p(z) = z^4 - 1$ is suitable. This polynomial has four zeros, which are the fourth roots of unity. Each of these zeros has a **basin of attraction**, that is, the set of all points z_0 such that Newton's iteration, started at z_0, will converge to that zero. These four basins of attraction are disjoint from each other, because if the Newton iteration starting at z_0 converges to one zero, then it cannot also converge to another zero. One would naturally expect each basin to be a simple set surrounding the zero in the complex plane. But they turn out to be far from simple. To see what they are, we can systematically determine, for a large number of points, which zero of p the Newton iteration converges to if started at z_0. Points in each basin can be assigned different colors. The (rare) points for which the Newton iteration does not converge can be left uncolored. Computer Problem 3.2.27 suggests how to do this.

Summary

(1) For finding a zero of a continuous and differentiable function f, **Newton's method** is given by

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \qquad (n \geqq 0)$$

It requires a given initial value x_0 and two function evaluations (for f and f') per step.

(2) The errors are related by

$$e_{n+1} = -\frac{1}{2}\left(\frac{f''(\xi_n)}{f'(x_n)}\right) e_n^2$$

which leads to the inequality

$$|e_{n+1}| \leqq c|e_n|^2$$

This means that Newton's method has **quadratic convergence** behavior for x_0 sufficiently close to the root r.

(3) For an $N \times N$ system of nonlinear equations $\mathbf{F}(\mathbf{X}) = \mathbf{0}$, **Newton's method** is written as

$$\mathbf{X}^{(k+1)} = \mathbf{X}^{(k)} - \left[\mathbf{F}'\left(\mathbf{X}^{(k)}\right)\right]^{-1}\mathbf{F}\left(\mathbf{X}^{(k)}\right) \qquad (k \geqq 0)$$

which involves the Jacobian matrix $\mathbf{F}'\left(\mathbf{X}^{(k)}\right) = \mathbf{J} = \left[\left(\partial f_i\left(\mathbf{X}^{(k)}\right)/\partial x_j\right)\right]_{N\times N}$. In practice, one solves the **Jacobian linear system**

$$\left[\mathbf{F}'(\mathbf{X}^{(k)})\right]\mathbf{H}^{(k)} = -\mathbf{F}\left(\mathbf{X}^{(k)}\right)$$

using Gaussian elimination and then finds the next iterate from the equation

$$\mathbf{X}^{(k+1)} = \mathbf{X}^{(k)} + \mathbf{H}^{(k)}$$

Additional References

For additional details and sample plots, see Kincaid and Cheney [2002] or Epureanu and Greenside [1998]. For other references on fractals, see Crilly, Earnshall, and Jones [1991], Feder [1998], Hastings and Sugihara [1993], and Novak [1998].

Moreover, an expository paper by Ypma [1995] traces the historical development of Newton's method through notes, letters, and publications by Isaac Newton, Joseph Raphson, and Thomas Simpson.

Problems 3.2

1. Verify that when Newton's method is used to compute $\sqrt{R}$ (by solving the equation $x^2 = R$), the sequence of iterates is defined by

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

2. (Continuation) Show that if the sequence $\{x_n\}$ is defined as in the preceding problem, then

$$x_{n+1}^2 - R = \left[\frac{x_n^2 - R}{2x_n}\right]^2$$

Interpret this equation in terms of quadratic convergence.

[a]3. Write Newton's method in simplified form for determining the reciprocal of the square root of a positive number. Perform two iterations to approximate $1/\pm\sqrt{5}$, starting with $x_0 = 1$ and $x_0 = -1$.

[a]4. Two of the four zeros of $x^4 + 2x^3 - 7x^2 + 3$ are positive. Find them by Newton's method, correct to two significant figures.

5. The equation $x - Rx^{-1} = 0$ has $x = \pm R^{1/2}$ for its solution. Establish Newton's iterative scheme, in simplified form, for this situation. Carry out five steps for $R = 25$ and $x_0 = 1$.

6. Using a calculator, observe the sluggishness with which Newton's method converges in the case of $f(x) = (x - 1)^m$ with $m = 8$ or 12. Reconcile this with the theory. Use $x_0 = 1.1$.

[a]7. What linear function $y = ax + b$ approximates $f(x) = \sin x$ best in the vicinity of $x = \pi/4$? How does this problem relate to Newton's method?

8. In Problems 1.2.11 and 1.2.12, several methods are suggested for computing $\ln 2$. Compare them with the use of Newton's method applied to the equation $e^x = 2$.

[a]9. Define a sequence $x_{n+1} = x_n - \tan x_n$ with $x_0 = 3$. What is $\lim_{n\to\infty} x_n$?

10. The iteration formula $x_{n+1} = x_n - (\cos x_n)(\sin x_n) + R\cos^2 x_n$, where R is a positive constant, was obtained by applying Newton's method to some function $f(x)$. What was $f(x)$? What can this formula be used for?

[a]11. Establish Newton's iterative scheme in simplified form, not involving the reciprocal of x, for the function $f(x) = xR - x^{-1}$. Carry out three steps of this procedure using $R = 4$ and $x_0 = -1$.

12. Consider the following procedures:

[a]**a.** $x_{n+1} = \frac{1}{3}\left(2x_n - \frac{r}{x_n^2}\right)$ **b.** $x_{n+1} = \frac{1}{2}x_n + \frac{1}{x_n}$

Do they converge for any nonzero initial point? If so, to what values?

13. Each of the following functions has $\sqrt[3]{R}$ as a zero for any positive real number R. Determine the formulas for Newton's method for each and any necessary restrictions on the choice for x_0.

[a]**a.** $a(x) = x^3 - R$ **b.** $b(x) = 1/x^3 - 1/R$ [a]**c.** $c(x) = x^2 - R/x$

d. $d(x) = x - R/x^2$ [a]**e.** $e(x) = 1 - R/x^3$ **f.** $f(x) = 1/x - x^2/R$

[a]**g.** $g(x) = 1/x^2 - x/R$ **h.** $h(x) = 1 - x^3/R$

14. Determine the formulas for Newton's method for finding a root of the function $f(x) = x - e/x$. What is the behavior of the iterates?

[a]**15.** If Newton's method is used on $f(x) = x^3 - x + 1$ starting with $x_0 = 1$, what will x_1 be?

16. Locate the root of $f(x) = e^{-x} - \cos x$ that is nearest $\pi/2$.

[a]**17.** If Newton's method is used on $f(x) = x^5 - x^3 + 3$ and if $x_n = 1$, what is x_{n+1}?

18. Determine Newton's iteration formula for computing the cube root of N/M for nonzero integers N and M.

[a]**19.** For what starting values will Newton's method converge if the function f is $f(x) = x^2/(1 + x^2)$?

20. Starting at $x = 3$, $x < 3$, or $x > 3$, analyze what happens when Newton's method is applied to the function $f(x) = 2x^3 - 9x^2 + 12x + 15$.

[a]**21.** (Continuation) Repeat for $f(x) = \sqrt{|x|}$, starting with $x < 0$ or $x > 0$.

[a]**22.** To determine $x = \sqrt[3]{R}$, we can solve the equation $x^3 = R$ by Newton's method. Write the loop that carries out this process, starting from the initial approximation $x_0 = R$.

23. The **reciprocal** of a number R can be computed without division by the iterative formula

$$x_{n+1} = x_n(2 - x_n R)$$

Establish this relation by applying Newton's method to some $f(x)$. Beginning with $x_0 = 0.2$, compute the reciprocal of 4 correct to six decimal digits or more by this rule. Tabulate the error at each step and observe the quadratic convergence.

24. On a certain modern computer, floating-point numbers have a 48-bit mantissa. Moreover, floating-point hardware can perform addition, subtraction, multiplication, and reciprocation, but not division. Unfortunately, the reciprocation hardware produces a result accurate to less than full precision, whereas the other operations produce results accurate to full floating-point precision.

a. Show that Newton's method can be used to find a zero of the function $f(x) = 1 - 1/(ax)$. This will provide an approximation to $1/a$ that is accurate to full floating-point precision. How many iterations are required?

b. Show how to obtain an approximation to b/a that is accurate to full floating-point precision.

25. Newton's method for finding $\sqrt{R}$ is

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

Perform three iterations of this scheme for computing $\sqrt{2}$, starting with $x_0 = 1$, and of the bisection method for $\sqrt{2}$, starting with interval $[1, 2]$. How many iterations are needed for each method in order to obtain 10^{-6} accuracy?

26. (Continuation) Newton's method for finding $\sqrt{R}$, where $R = AB$, gives this approximation:

$$\sqrt{AB} \approx \frac{A + B}{4} + \frac{AB}{A + B}$$

Show that if $x_0 = A$ or B, then two iterations of Newton's method are needed to obtain this approximation, whereas if $x_0 = \frac{1}{2}(A + B)$, then only one iteration is needed.

[a]**27.** Show that Newton's method applied to $x^m - R$ and to $1 - (R/x^m)$ for determining $\sqrt[m]{R}$ results in two similar yet different iterative formulas. Here $R > 0$, $m \geqq 2$. Which formula is better and why?

28. Using a handheld calculator, carry out three iterations of Newton's method using $x_0 = 1$ and $f(x) = 3x^3 + x^2 - 15x + 3$.

[a]**29.** What happens if the Newton iteration is applied to $f(x) = \arctan x$ with $x_0 = 2$? For what starting values will Newton's method converge? (See Computer Problem 3.2.7.)

30. Newton's method can be interpreted as follows: Suppose that $f(x + h) = 0$. Then $f'(x) \approx [f(x + h) - f(x)]/h = -f(x)/h$. Continue this argument.

[a]**31.** Derive a formula for Newton's method for the function $F(x) = f(x)/f'(x)$, where $f(x)$ is a function with simple zeros that is three times continuously differentiable. Show that the convergence of the resulting method to any zero r of $f(x)$ is at least quadratic. *Hint:* Apply the result in the text to F, making sure that F has the required properties.

[a]**32.** The Taylor series for a function f looks like this:

$$f(x + h) = f(x) + hf'(x) + \frac{h^2}{2}f''(x) + \frac{h^3}{6}f'''(x) + \cdots$$

Suppose that $f(x)$, $f'(x)$, and $f''(x)$ are easily computed. Derive an algorithm like Newton's method that uses three terms in the Taylor series. The algorithm should take as input an approximation to the root and produce as output a better approximation to the root. Show that the method is cubically convergent.

33. To avoid computing the derivative at each step in Newton's method, it has been proposed to replace $f'(x_n)$ by $f'(x_0)$. Derive the rate of convergence for this method.

34. Refer to the discussion of Newton's method and establish that

$$\lim_{n\to\infty}\left(e_{n+1}e_n^{-2}\right) = -\frac{1}{2}\left[\frac{f''(r)}{f'(r)}\right]$$

How can this be used in a practical case to test whether the convergence is quadratic? Devise an example in which r, $f'(r)$, and $f''(r)$ are all known, and test numerically the convergence of $e_{n+1}e_n^{-2}$.

[a]**35.** Show that in the case of a zero of multiplicity m, the **modified Newton's method**

$$x_{n+1} = x_n - m\frac{f(x_n)}{f'(x_n)}$$

is quadratically convergent. *Hint:* Use Taylor series for each of $f(r+e_n)$ and $f'(r+e_n)$.

[a]**36.** The **Steffensen method** for solving the equation $f(x) = 0$ uses the formula

$$x_{n+1} = x_n - \frac{f(x_n)}{g(x_n)}$$

in which $g(x) = \{f[x + f(x)] - f(x)\}/f(x)$. It is quadratically convergent, like Newton's method. How many function evaluations are necessary per step? Using Taylor series, show that $g(x) \approx f'(x)$ if $f(x)$ is small and thus relate Steffensen's iteration to Newton's. What advantage does Steffensen's have? Establish the quadratic convergence.

[a]**37.** A proposed **Generalization of Newton's method** is

$$x_{n+1} = x_n - \omega\frac{f(x_n)}{f'(x_n)}$$

where the constant ω is an acceleration factor chosen to increase the rate of convergence. For what range of values of ω is a simple root r of $f(x)$ a **point of attraction**; that is, $|g'(r)| < 1$, where $g(x) = x - \omega f(x)/f'(x)$? This method is quadratically convergent *only* if $\omega = 1$ because $g'(r) \neq 0$ when $\omega \neq 1$.

38. Suppose that r is a double root of $f(x) = 0$; that is, $f(r) = f'(r) = 0$ but $f''(r) \neq 0$, and suppose that f and all derivatives up to and including the second are continuous in some neighborhood of r. Show that $e_{n+1} \approx \frac{1}{2}e_n$ for Newton's method and thereby conclude that the rate of convergence is *linear* near a double root. (If the root has multiplicity m, then $e_{n+1} \approx [(m-1)/m]e_n$.)

39. (**Simultaneous nonlinear equations**) Using the Taylor series in two variables (x, y) of the form

$$f(x+h, y+k) = f(x, y) + hf_x(x, y) + kf_y(x, y) + \cdots$$

where $f_x = \partial f/\partial x$ and $f_y = \partial f/\partial y$, establish that Newton's method for solving the two simultaneous nonlinear equations

$$\begin{cases} f(x, y) = 0 \\ g(x, y) = 0 \end{cases}$$

can be described with the formulas

$$x_{n+1} = x_n - \frac{fg_y - gf_y}{f_xg_y - g_xf_y}, \qquad y_{n+1} = y_n - \frac{f_xg - g_xf}{f_xg_y - g_xf_y}$$

Here the functions f, f_x, and so on are evaluated at (x_n, y_n).

40. Newton's method can be defined for the equation $f(z) = g(x, y) + ih(x, y)$, where $f(z)$ is an analytic function of the complex variable $z = x + iy$ (x and y real) and $g(x, y)$ and $h(x, y)$ are real functions for all x and y. The derivative $f'(z)$ is given by $f'(z) = g_x + ih_x = h_y - ig_y$ because the **Cauchy-Riemann equations** $g_x = h_y$ and $h_x = -g_y$ hold. Here the partial derivatives are defined as $g_x = \partial g/\partial x$, $g_y = \partial g/\partial y$, and so on. Show that Newton's method

$$z_{n+1} = z_n - \frac{f(z_n)}{f'(z_n)}$$

can be written in the form

$$x_{n+1} = x_n - \frac{gh_y - hg_y}{g_x h_y - g_y h_x}, \qquad y_{n+1} = y_n - \frac{hg_x - gh_x}{g_x h_y - g_y h_x}$$

Here all functions are evaluated at $z_n = x_n + iy_n$.

[a]**41.** Consider the algorithm of which *one* step consists of two steps of Newton's method. What is its order of convergence?

42. (Continuation) Using the idea of the preceding Problem, show how we can easily create methods of arbitrarily high order for solving $f(x) = 0$. Why is the order of a method not the only criterion that should be considered in assessing its merits?

43. If we want to solve the equation $2 - x = e^x$ using Newton's iteration, what are the equations and functions that must be coded? Give a pseudocode for doing this problem. Include a suitable starting point and a suitable stopping criterion.

44. Suppose that we want to compute $\sqrt{2}$ by using Newton's Method on the equation $x^2 = 2$ (in the obvious, straightforward way). If the starting point is $x_0 = \frac{7}{5}$, what is the numerical value of the correction that must be added to x_0 to get x_1? *Hint:* The arithmetic is quite easy if you do it using ratios of integers.

45. Apply Newton's method to the equation $f(x) = 0$ with $f(x)$ as given below. Find out what happens and why.

a. $f(x) = e^x$ **b.** $f(x) = e^x + x^2$

46. Consider Newton's method $x_{n+1} = x_n - f(x_n)/f'(x_n)$. If the sequence converges then the limit point is a solution. Explain why or why not.

Computer Problems 3.2

1. Using the procedure *Newton* and a single computer run, test your code on these examples: $f(t) = \tan t - t$ with $x_0 = 7$ and $g(t) = e^t - \sqrt{t + 9}$ with $x_0 = 2$. Print each iterate and its accompanying function value.

2. Write a simple, self-contained program to apply Newton's method to the equation $x^3 + 2x^2 + 10x = 20$, starting with $x_0 = 2$. Evaluate the appropriate $f(x)$ and $f'(x)$, using nested multiplication. Stop the computation when two successive points differ by $\frac{1}{2} \times 10^{-5}$ or some other convenient tolerance close to your machine's capability. Print all intermediate points and function values. Put an upper limit of ten on the number of steps.

3. (Continuation) Repeat using double precision and more steps.

[a]**4.** Find the root of the equation

$$2x(1 - x^2 + x)\ln x = x^2 - 1$$

in the interval [0, 1] by Newton's method using double precision. Make a table that shows the number of correct digits in each step.

[a]**5.** In 1685, John Wallis published a book called *Algebra*, in which he described a method devised by Newton for solving equations. In slightly modified form, this method was also published by Joseph Raphson in 1690. This form is the one now commonly called Newton's method or the Newton-Raphson method. Newton himself discussed the method in 1669 and illustrated it with the equation $x^3 - 2x - 5 = 0$. Wallis used the same example. Find a root of this equation in double precision, thus continuing the tradition that every numerical analysis student should solve this venerable equation.

6. In celestial mechanics, **Kepler's equation** is important. It reads $x = y - \varepsilon \sin y$, in which x is a planet's mean anomaly, y its eccentric anomaly, and ε the eccentricity of its orbit. Taking $\varepsilon = 0.9$, construct a table of y for 30 equally spaced values of x in the interval $0 \leqq x \leqq \pi$. Use Newton's method to obtain each value of y. The y corresponding to an x can be used as the starting point for the iteration when x is changed slightly.

7. In Newton's method, we progress in each step from a given point x to a new point $x - h$, where $h = f(x)/f'(x)$. A refinement that is easily programmed is this: If $|f(x - h)|$ is not smaller than $|f(x)|$, then reject this value of h and use $h/2$ instead. Test this refinement.

[a]**8.** Write a brief program to compute a root of the equation $x^3 = x^2 + x + 1$, using Newton's method. Be careful to select a suitable starting value.

[a]**9.** Find the root of the equation $5(3x^4 - 6x^2 + 1) = 2(3x^5 - 5x^3)$ that lies in the interval [0, 1] by using Newton's method and a short program.

10. For each equation, write a brief program to compute and print eight steps of Newton's method for finding a positive root.

[a]**a.** $x = 2\sin x$ [a]**b.** $x^3 = \sin x + 7$ [a]**c.** $\sin x = 1 - x$

[a]**d.** $x^5 + x^2 = 1 + 7x^3$ for $x \geqq 2$

11. Write and test a recursive procedure for Newton's method.

12. Rewrite and test the *Newton* procedure so that it is a character function and returns key words such as `iterating`, `success`, `near-zero`, `max-iteration`. Then a case statement can be used to print the results.

13. Would you like to see the number 0.55887 766 come out of a calculation? Take three steps in Newton's method on $10 + x^3 - 12\cos x = 0$ starting with $x_0 = 1$.

[a]**14.** Write a short program to solve for a root of the equation $e^{-x^2} = \cos x + 1$ on [0, 4]. What happens in Newton's method if we start with $x_0 = 0$ or with $x_0 = 1$?

15. Find the root of the equation $\frac{1}{2}x^2 + x + 1 - e^x = 0$ by Newton's method, starting with $x_0 = 1$, and account for the slow convergence.

16. Using $f(x) = x^5 - 9x^4 - x^3 + 17x^2 - 8x - 8$ and $x_0 = 0$, study and explain the behavior of Newton's method. *Hint:* The iterates are initially cyclic.

17. Find the zero of the function $f(x) = x - \tan x$ that is closest to 99 (radians) by both the bisection method and Newton's method. *Hint:* Extremely accurate starting values are needed for this function. Use the computer to construct a table of values of $f(x)$ around 99 to determine the nature of this function.

18. Using the bisection method, find the positive root of $2x(1 + x^2)^{-1} = \arctan x$. Using the root as x_0, apply Newton's method to the function $\arctan x$. Interpret the results.

19. If the root of $f(x) = 0$ is a double root, then Newton's method can be accelerated by using

$$x_{n+1} = x_n - 2\frac{f(x_n)}{f'(x_n)}$$

Numerically compare the convergence of this scheme with Newton's method on a function with a known double root.

20. Program and test Steffensen's method, as described in Problem 3.2.36.

21. Consider the nonlinear system

$$\begin{cases} f(x, y) = x^2 + y^2 - 25 = 0 \\ g(x, y) = x^2 - y - 2 = 0 \end{cases}$$

Using a software package that has 2D plotting capabilities, illustrate what is going on in solving such a system by plotting $f(x, y)$, $g(x, y)$, and show their intersection with the (x, y)-plane. Determine approximate roots of these equations from the graphical results.

22. Solve this pair of simultaneous nonlinear equations by first eliminating y and then solving the resulting equation in x by Newton's method. Start with the initial value $x_0 = 1.0$.

$$\begin{cases} x^3 - 2xy + y^7 - 4x^3y = 5 \\ y\sin x + 3x^2y + \tan x = 4 \end{cases}$$

23. Using Equations (7) and (8), code Newton's methods for nonlinear systems. Test your program by solving one or more of the following systems:

a. System in Computer Problem 3.2.21.

b. System in Computer Problem 3.2.22.

c. System (3) using starting values (0, 0, 0).

d. Using starting values $\left(\frac{3}{4}, \frac{1}{2}, -\frac{1}{2}\right)$, solve

$$\begin{cases} x + y + z = 0 \\ x^2 + y^2 + z^2 = 2 \\ x(y + z) = -1 \end{cases}$$

e. Using starting values $(-0.01, -0.01)$, solve

$$\begin{cases} 4y^2 + 4y + 52x - 19 = 0 \\ 169x^2 + 3y^2 + 111x - 10y - 10 = 0 \end{cases}$$

f. Select starting values, and solve

$$\begin{cases} \sin(x+y) = e^{x-y} \\ \cos(x+6) = x^2y^2 \end{cases}$$

24. Investigate the behavior of Newton's method for finding complex roots of polynomials with real coefficients. For example, the polynomial $p(x) = x^2 + 1$ has the complex conjugate pair of roots $\pm i$ and Newton's method is $x_{n+1} = \frac{1}{2}(x_n - 1/x_n)$. First, program this method using real arithmetic and real numbers as starting values. Second, modify the program using complex arithmetic but still using only real starting values. Finally, use complex numbers as starting values. Observe the behavior of the iterates in each case.

25. Using Problem 3.2.40, find a complex root of each of the following:

a. $z^3 - z - 1 = 0$

b. $z^4 - 2z^3 - 2iz^2 + 4iz = 0$

c. $2z^3 - 6(1+i)z^2 - 6(1-i) = 0$

d. $z = e^z$

Hint: For the last part, use Euler's relation $e^{iy} = \cos y + i \sin y$.

26. In the Newton method for finding a root r of $f(x) = 0$, we start with x_0 and compute the sequence $x_1, x_2, \ldots$ using the formula $x_{n+1} = x_n - f(x_n)/f'(x_n)$. To avoid computing the derivative at each step, it has been proposed to replace $f'(x_n)$ with $f'(x_0)$ in all steps. It has also been suggested that the derivative in Newton's formula be computed only every other step. This method is given by

$$\begin{cases} x_{2n+1} = x_{2n} - \dfrac{f(x_{2n})}{f'(x_{2n})} \\ x_{2n+2} = x_{2n+1} - \dfrac{f(x_{2n+1})}{f'(x_{2n})} \end{cases}$$

Numerically compare both proposed methods to Newton's method for several simple functions that have known roots. Print the error of each method on every iteration to monitor the convergence. How well do the proposed methods work?

27. **(Basin of attraction)** Consider the complex polynomial $z^3 - 1$, whose zeros are the three cube roots of unity. Generate a picture showing three basins of attraction in the complex plane in the square region defined by $-1 \leqq \text{Real}(z) \leqq 1$ and $-1 \leqq \text{Imaginary}(z) \leqq 1$. To do this, use a mesh of 1000×1000 pixels inside the square. The center point of each pixel is used to start the iteration of Newton's method. Assign a particular basin color to each pixel if convergence to a root is obtained with $nmax = 10$ iterations. The large number of iterations suggested can be avoided by doing some analysis with the aid of Theorem 1, since the iterates get within a certain neighborhood of the root and the iteration can be stopped. The criterion for convergence is to check both $|z_{n+1} - z_n| < \varepsilon$ and $|z_{n+1}^3 - 1| < \varepsilon$ with a small value such as $\varepsilon = 10^{-4}$ as well as a maximum number of iterations. *Hint:* It is best to debug your program and get a crude picture with only a small number of pixels such as 10×10.

28. (Continuation) Repeat for the polynomial $z^4 - 1 = 0$.

29. Write **real function** $Sqrt(x)$ to compute the square root of a real argument x by the following algorithm: First, reduce the range of x by finding a real number r and an integer m such that $x = 2^{2m}r$ with $\frac{1}{4} \leqq r < 1$. Next, compute x_2 by using three iterations of Newton's method given by

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

with the special initial approximation

$$x_0 = 1.27235\,367 + 0.24269\,3281r - \frac{1.02966\,039}{1+r}$$

Then set $\sqrt{x} \approx 2^m x_2$. Test this algorithm on various values of x. Obtain a listing of the code for the square-root function on your computer system. By reading the comments, try to determine what algorithm it uses.

30. The following method has third-order convergence for computing $\sqrt{R}$:

$$x_{n+1} = \frac{x_n\left(x_n^2 + 3R\right)}{3x_n^2 + R}$$

Carry out some numerical experiments using this method and the method of the preceding problem to see whether you observe a difference in the rate of convergence. Use the same starting procedures of range reduction and initial approximation.

31. Write **real function** *CubeRoot*(x) to compute the cube root of a real argument x by the following procedure: First, determine a real number r and an integer m such that $x = r2^{3m}$ with $\frac{1}{8} \leqq r < 1$. Compute x_4 using four iterations of Newton's method:

$$x_{n+1} = \frac{2}{3}\left(x_n + \frac{r}{2x_n^2}\right)$$

with the special starting value

$$x_0 = 2.50292\,6 - \frac{8.04512\,5(r + 0.38775\,52)}{(r + 4.61224\,4)(r + 0.38775\,52) - 0.35984\,96}$$

Then set $\sqrt[3]{x} \approx 2^m x_4$. Test this algorithm on a variety of x values.

32. Use mathematical software such as in Maple or Mathematica to compute ten iterates of Newton's method starting with $x_0 = 0$ for $f(x) = x^3 - 2x^2 + x - 3$. With 100 decimal places of accuracy and after nine iterations, show that the value of x is

2.17455 94102 92980 07420 23189 88695 65392 56759 48725 33708
24983 36733 92030 23647 64792 75760 66115 28969 38832 0640

Show that the values of the function at each iteration are 9.0, 2.0, 0.26, 0.0065, 0.45×10^{-5}, 0.22×10^{-11}, 0.50×10^{-24}, 0.27×10^{-49}, 0.1×10^{-98}, and 0.1×10^{-98}. Again notice that the number of digits of accuracy in Newton's method doubles (approximately) with each iteration once they are sufficiently close to the root. (Also, see Bornemann, Wagon, and Waldvogel [2004] for a 100-Digit Challenge, which is a study in high-accuracy numerical computing.)

33. (Continuation) Use Maple or Mathematica to discover that this root is exactly

$$\sqrt[3]{\frac{79}{54}+\frac{1}{6}\sqrt{77}}+\frac{1}{9\sqrt[3]{\frac{79}{54}+\frac{1}{6}\sqrt{77}}}+\frac{2}{3}$$

Clearly, the decimal results are of more interest to us in our study of numerical methods.

34. (Continuation) Find all the roots including complex roots.

35. Numerically, find all the roots of the following systems of nonlinear equations. Then plot the curves to verify your results:

a. $y = 2x^2 + 3x - 4$, $y = x^2 + 2x + 3$

b. $y + x + 3 = 0$, $x^2 + y^2 = 17$

c. $y = \frac{1}{2}x - 5$, $y = x^2 + 2x - 15$

d. $xy = 1$, $x + y = 2$

e. $y = x^2$, $x^2 + (y - 2)^2 = 4$

f. $3x^2 + 2y^2 = 35$, $4x^2 - 3y^2 = 24$

g. $x^2 - xy + y^2 = 21$, $x^2 + 2xy - 8y^2 = 0$

36. Apply Newton's method on these test problems:

a. $f(x) = x^2$. *Hint:* The first derivative is zero at the root and convergence may not be quadratic.

b. $f(x) = x + x^{4/3}$. *Hint:* There is no second derivative at the root and convergence may fail to be quadratic.

c. $f(x) = x + x^2 \sin(2/x)$ for $x \neq 0$ and $f(0) = 0$ and $f'(x) = 1 + 2x \sin(2/x) - 2\cos(2/x)$ for $x \neq 0$ and $f'(0) = 1$. *Hint:* The derivative of this function is not continuous at the root and convergence may fail.

37. Let $\mathbf{F}(\mathbf{X}) = \begin{bmatrix} x_1^2 - x_2 + c \\ x_2^2 - x_1 + c \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$. Each component equation $f_1(x) = 0$ and $f_2(x) = 0$ describes a parabola. Any point (x^*, y^*) where these two parabolas intersect is a solution to the nonlinear system of equations. Using Newton's method for systems of nonlinear equations, find the solutions for each of these values of the parameter $c = \frac{1}{2}, \frac{1}{4}, -\frac{1}{2}, -1$. Give the Jacobian matrix for each. Also for each of these values, plot the resulting curves showing the points of intersection. (Heath 2000, p. 218)

38. Let $\mathbf{F}(\mathbf{X}) = \begin{bmatrix} x_1^2 + 2x_2 - 2 \\ x_1 + 4x_2^2 - 4 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$. Solve this nonlinear system starting with $\mathbf{X}^{(0)} = (1, 2)$. Give the Jacobian matrix. Also plot the resulting curves showing the point(s) of intersection.

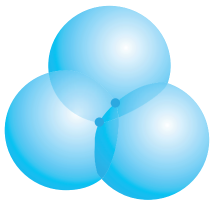
39. Using Newton's method, find the zeros of $f(z) = z^3 - z$ with these starting values $z^{(0)} = 1 + 1.5i, 1 + 1.1i, 1 + 1.2i, 1 + 1.3i$.

40. Use Halley's method to produce a plot of the basins of attraction for $p(z) = z^6 - 1$. Compare to Figure 3.8.

41. (**Global positioning system project**) Each time a GPS is used, a system of nonlinear equations of the form

$$
\begin{aligned}
(x-a_1)^2 + (y-b_1)^2 + (z-c_i)^2 &= [(C(t_1 - D)]^2\\
(x-a_2)^2 + (y-b_2)^2 + (z-c_i)^2 &= [(C(t_2 - D)]^2\\
(x-a_3)^2 + (y-b_3)^2 + (z-c_i)^2 &= [(C(t_3 - D)]^2\\
(x-a_4)^2 + (y-b_4)^2 + (z-c_i)^2 &= [(C(t_4 - D)]^2
\end{aligned}
$$

is solved for the (x, y, z) coordinates of the receiver. For each satellite i, the locations are (a_i, b_i, c_i), and t_i is the synchronized transmission time from the satellite. Further, C is the speed of light, and D is the difference between the synchronized time of the satellite clocks and the earth-bound receiver clock. While there are only two points on the intersection of three spheres (one of which can be determined to be the desired location), a fourth sphere (satellite) must be used to resolve the inaccuracy in the clock contained in the low-cost receiver on earth. Explore various ways for solving such a nonlinear system. See Hofmann-Wellenhof, Lichtenegger, and Collins [2001], Sauer [2006], and Strang and Borre [1997].



42. Use mathematical software such as in Matlab, Maple, or Mathematica and their built-in procedures to solve the system of nonlinear equations (8) in Example 2. Also, plot the given surfaces and the solution obtained. *Hint:* You may need to use a slightly perturbed starting point (0.5, 1.5, 0.5) to avoid a singularity in the Jacobian matrix.

3.3 Secant Method

We now consider a general-purpose procedure that converges almost as fast as Newton's method. This method mimics Newton's method but avoids the calculation of derivatives. Recall that Newton's iteration defines x_{n+1} in terms of x_n via the formula

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \tag{1}$$

In the secant method, we replace $f'(x_n)$ in Formula (1) by an approximation that is easily computed. Since the derivative is *defined* by

$$f'(x) = \lim_{h\to 0} \frac{f(x+h) - f(x)}{h}$$

we can say that for small h,

$$f'(x) \approx \frac{f(x+h) - f(x)}{h}$$

(In Section 4.3, we revisit this subject and learn that this is a finite difference approximation to the first derivative.) In particular, if $x = x_n$ and $h = x_{n-1} - x_n$, we have

$$f'(x_n) \approx \frac{f(x_{n-1}) - f(x_n)}{x_{n-1} - x_n} \tag{2}$$

When this is used in Equation (1), the result defines the **secant method**:

$$x_{n+1} = x_n - \left(\frac{x_n - x_{n-1}}{f(x_n) - f(x_{n-1})} \right) f(x_n) \tag{3}$$

The secant method (like Newton's) can be used to solve systems of equations as well.

The name of the method is taken from the fact that the right member of Equation (2) is the slope of a secant line to the graph of f (see Figure 3.9). Of course, the left member is the slope of a *tangent* line to the graph of f. (Similarly, Newton's method could be called the "tangent method.")

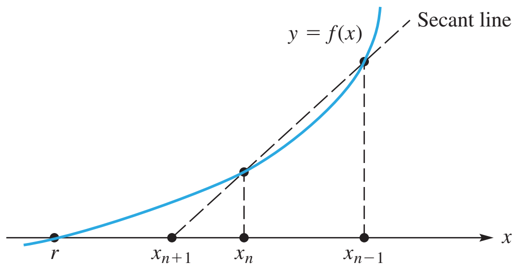


FIGURE 3.9
Secant method

A few remarks about Equation (3) are in order. Clearly, x_{n+1} depends on *two* previous elements of the sequence. So to start, two points (x_0 and x_1) must be provided. Equation (3) can then generate $x_2, x_3, \ldots$. In programming the secant method, we could calculate and test the quantity $f(x_n) - f(x_{n-1})$. If it is nearly zero, an overflow can occur in Equation (3). Of course, if the method is succeeding, the points x_n will be approaching a zero of f, so $f(x_n)$ will be converging to zero. (We are assuming that f is continuous.) Also, $f(x_{n-1})$ will be converging to zero, and, a fortiori, $f(x_n) - f(x_{n-1})$ will approach zero. If the terms $f(x_n)$ and $f(x_{n-1})$ have the same sign, additional significant digits are canceled in the subtraction. So we could perhaps halt the iteration when $|f(x_n) - f(x_{n-1})| \leqq \delta |f(x_n)|$ for some specified tolerance δ, such as $\frac{1}{2} \times 10^{-6}$. (See Computer Problem 3.3.18.)

Secant Algorithm

A pseudocode for *nmax* steps of the secant method applied to the function f starting with the interval $[a, b] = [x_0, x_1]$ can be written as follows:

```
procedure Secant($f, a, b, nmax, \varepsilon$)
integer $n, nmax$;   real $a, b, fa, fb, \varepsilon, d$
external function $f$
$fa \leftarrow f(a)$
$fb \leftarrow f(b)$
```

```
if $|fa| > |fb|$ then
    $a \longleftrightarrow b$
    $fa \longleftrightarrow fb$
end if
output $0, a, fa$
output $1, b, fb$
for $n = 2$ to nmax do
    if $|fa| > |fb|$ then
        $a \longleftrightarrow b$
        $fa \longleftrightarrow fb$
    end if
    $d \leftarrow (b - a)/(fb - fa)$
    $b \leftarrow a$
    $fb \leftarrow fa$
    $d \leftarrow d \cdot fa$
    if $|d| < \varepsilon$ then
        output "convergence"
        return
    end if
    $a \leftarrow a - d$
    $fa \leftarrow f(a)$
    output $n, a, fa$
end for
end procedure Secant
```

Here $\longleftrightarrow$ means interchange values. The endpoints $[a, b]$ are interchanged, if necessary, to keep $|f(a)| \leqq |f(b)|$. Consequently, the absolute values of the function are nonincreasing; thus, we have $|f(x_n)| \geqq |f(x_{n+1})|$ for $n \geqq 1$.

EXAMPLE 1 If the secant method is used on $p(x) = x^5 + x^3 + 3$ with $x_0 = -1$ and $x_1 = 1$, what is x_8?

Solution The output from the computer program corresponding to the pseudocode for the secant method is as follows. (We used a 32-bit word-length computer.)

n	x_n	$p(x_n)$
0	-1.0	1.0
1	1.0	5.0
2	-1.5	-7.97
3	-1.05575	0.512
4	-1.11416	-9.991×10^{-2}
5	-1.10462	7.593×10^{-3}
6	-1.10529	1.011×10^{-4}
7	-1.10530	2.990×10^{-7}
8	-1.10530	2.990×10^{-7}

We can use mathematical software to find the single real root, -1.1053, and the two pairs of complex roots, $-0.319201 \pm 1.35008i$ and $0.871851 \pm 0.806311i$. ■

Convergence Analysis

The advantages of the secant method are that (after the first step) only one function evaluation is required per step (in contrast to Newton's iteration, which requires two) and that it is almost as rapidly convergent. It can be shown that the basic secant method defined by Equation (3) obeys an equation of the form

$$e_{n+1} = -\frac{1}{2}\left(\frac{f''(\xi_n)}{f'(\zeta_n)}\right)e_n e_{n-1} \approx -\frac{1}{2}\left(\frac{f''(r)}{f'(r)}\right)e_n e_{n-1} \tag{4}$$

where ξ_n and ζ_n are in the smallest interval that contains r, x_n, and x_{n-1}. Thus, the ratio $e_{n+1}(e_n e_{n-1})^{-1}$ converges to $-\frac{1}{2}f''(r)/f'(r)$. The rapidity of convergence of this method is, in general, between those for bisection and for Newton's method.

To prove the second part of Equation (4), we begin with the definition of the secant method in Equation (3) and the error

$$\begin{aligned} e_{n+1} &= r - x_{n+1} \\ &= r - \frac{f(x_n)x_{n-1} - f(x_{n-1})x_n}{f(x_n) - f(x_{n-1})} \\ &= \frac{f(x_n)e_{n-1} - f(x_{n-1})e_n}{f(x_n) - f(x_{n-1})} \\ &= \left[\frac{x_n - x_{n-1}}{f(x_n) - f(x_{n-1})}\right]\left[\frac{\dfrac{f(x_n)}{e_n} - \dfrac{f(x_{n-1})}{e_{n-1}}}{x_n - x_{n-1}}\right]e_n e_{n-1} \end{aligned} \tag{5}$$

By Taylor's Theorem, we establish

$$f(x_n) = f(r - e_n) = f(r) - e_n f'(r) + \frac{1}{2}e_n^2 f''(r) + \mathcal{O}\left(e_n^3\right)$$

Since $f(r) = 0$, this gives us

$$\frac{f(x_n)}{e_n} = -f'(r) + \frac{1}{2}e_n f''(r) + \mathcal{O}\left(e_n^2\right)$$

Changing the index to $n - 1$ yields

$$\frac{f(x_{n-1})}{e_{n-1}} = -f'(r) + \frac{1}{2}e_{n-1} f''(r) + \mathcal{O}\left(e_{n-1}^2\right)$$

By subtraction between these equations, we arrive at

$$\frac{f(x_n)}{e_n} - \frac{f(x_{n-1})}{e_{n-1}} = \frac{1}{2}(e_n - e_{n-1})f''(r) + \mathcal{O}\left(e_{n-1}^2\right)$$

Since $x_n - x_{n-1} = e_{n-1} - e_n$, we reach the equation

$$\frac{\dfrac{f(x_n)}{e_n} - \dfrac{f(x_{n-1})}{e_{n-1}}}{x_n - x_{n-1}} \approx -\frac{1}{2}f''(r)$$

The first bracketed expression in Equation (5) can be written as

$$\frac{x_n - x_{n-1}}{f(x_n) - f(x_{n-1})} \approx \frac{1}{f'(r)}$$

Hence, we have shown the second part of Equation (4).

We leave the establishment of the first part of Equation (4) as a problem because it depends on some material to be covered in Chapter 4. (See Problem 3.3.18.)

From Equation (4), the order of convergence for the secant method can be expressed in terms of the inequality

$$|e_{n+1}| \leqq C|e_n|^{\alpha} \tag{6}$$

where $\alpha = \frac{1}{2}\left(1+\sqrt{5}\,\right) \approx 1.62$ is the **golden ratio**. Since $\alpha > 1$, we say that the convergence is **superlinear**. Assuming that Inequality (6) is true, we can show that the secant method converges under certain conditions.

Let $c = c(\delta)$ be defined as in Equation (2) of Section 3.2. If $|r - x_n| \leqq \delta$ and $|r - x_{n-1}| \leqq \delta$, for some root r, then Equation (4) yields

$$|e_{n+1}| \leqq c|e_n||e_{n-1}| \tag{7}$$

Suppose that the initial points x_0 and x_1 are sufficiently close to r that $c|e_0| \leqq D$ and $c|e_1| \leqq D$ for some $D < 1$. Then

$$\begin{aligned}
&c|e_1| \leqq D, \quad c|e_0| \leqq D\\
&c|e_2| \leqq c|e_1|\,c|e_0| \leqq D^2\\
&c|e_3| \leqq c|e_2|\,c|e_1| \leqq D^3\\
&c|e_4| \leqq c|e_3|\,c|e_2| \leqq D^5\\
&c|e_5| \leqq c|e_4|\,c|e_3| \leqq D^8\\
&\text{etc.}
\end{aligned}$$

In general, we have

$$|e_n| \leqq c^{-1} D^{\lambda_{n+1}} \tag{8}$$

where inductively,

$$\begin{cases} \lambda_1 = 1, \quad \lambda_2 = 1 \\ \lambda_n = \lambda_{n-1} + \lambda_{n-2} \qquad (n \geqq 3) \end{cases} \tag{9}$$

This is the recurrence relation for generating the famous **Fibonacci sequence**, 1, 1, 2, 3, 5, 8, ... It can be shown to have the surprising explicit form

$$\lambda_n = \frac{1}{\sqrt{5}}(\alpha^n - \beta^n) \tag{10}$$

where $\alpha = \frac{1}{2}\left(1+\sqrt{5}\,\right)$ and $\beta = \frac{1}{2}\left(1-\sqrt{5}\,\right)$. Since $D < 1$ and $\lambda_n \to \infty$, we conclude from Inequality (8) that $e_n \to 0$. Hence, $x_n \to r$ as $n \to \infty$, and the secant method converges to the root r if x_0 and x_1 are sufficiently close to it.

Next, we show that Inequality (6) is in fact *reasonable*—not a proof. From Equations (7), we now have

$$\begin{aligned}
|e_{n+1}| &\leqq c|e_n||e_{n-1}| \\
&= c|e_n|^{\alpha}|e_n|^{1-\alpha}|e_{n-1}| \\
&\approx c|e_n|^{\alpha}\left(c^{-1}D^{\lambda_{n+1}}\right)^{1-\alpha}\left(c^{-1}D^{\lambda_n}\right) \\
&= |e_n|^{\alpha}c^{\alpha-1}D^{\lambda_{n+1}(1-\alpha)+\lambda_n} \\
&= |e_n|^{\alpha}c^{\alpha-1}D^{\lambda_{n+2}-\alpha\lambda_{n+1}}
\end{aligned}$$

by using an approximation to Inequality (8). In the last line, we used the recurrence relation (9). Now $\lambda_{n+2} - \alpha\lambda_{n+1}$ converges to zero. (See Problem 3.3.6.). Hence, $c^{\alpha-1}D^{\lambda_{n+2}-\alpha\lambda_{n+1}}$ is bounded, say, by C, as a function of n. Thus, we have

$$|e_{n+1}| \approx C|e_n|^{\alpha}$$

which is a reasonable approximation to Inequality (6).

Another derivation (with a bit of *hand waving*) for the order of convergence of the secant method can be given by using a general recurrence relation. Equation (4) gives us

$$e_{n+1} \approx K e_n e_{n-1}$$

where $K = -\frac{1}{2}f''(r)/f'(r)$. We can write this as

$$|K e_{n+1}| \approx |K e_n|\,|K e_{n-1}|$$

Let $z_i = \log|K e_i|$. Then we want to solve the recurrence equation

$$z_{n+1} = z_n + z_{n-1}$$

where z_0 and z_1 are arbitrary. This is a linear recurrence relation with constant coefficients similar to the one for the Fibonacci numbers (9) except that the first two values z_0 and z_1 are unknown. The solution is of the form

$$z_n = A\alpha^n + B\beta^n \tag{11}$$

where $\alpha = \frac{1}{2}\left(1+\sqrt{5}\right)$ and $\beta = \frac{1}{2}\left(1-\sqrt{5}\right)$. These are the roots of the quadratic equation $\lambda^2 - \lambda - 1 = 0$. Since $|\alpha| > |\beta|$, the term $A\alpha^n$ dominates, and we can say that

$$z_n \approx A\alpha^n$$

for large n and for some constant A. Consequently, we have

$$|K e_n| \approx 10^{A\alpha^n}$$

Then it follows that

$$|K e_{n+1}| \approx 10^{A\alpha^{n+1}} = \left(10^{A\alpha^n}\right)^{\alpha} = |K e_n|^{\alpha}$$

Hence, we have

$$|e_{n+1}| \approx C|e_n|^{\alpha} \tag{12}$$

for large n and for some constant C. Again, Inequality (6) is *essentially* established! A rigorous proof of Inequality (6) is tedious and quite long.

Comparison of Methods

In this chapter, three primary methods for solving an equation $f(x) = 0$ have been presented. The bisection method is reliable but slow. Newton's method is fast but often only near the root and requires f'. The secant method is nearly as fast as Newton's method and does not require knowledge of the derivative f', which may not be available or may be too expensive to compute. The user of the bisection method must provide two points at which the signs of $f(x)$ differ, and the function f need only be continuous. In using Newton's method, one must specify a starting point near the root, and f must be differentiable. The secant method requires two good starting points. Newton's procedure can be interpreted as the repetition of a two-step procedure summarized by the prescription *linearize and solve*. This strategy is applicable in many other numerical problems, and its importance cannot be overemphasized. Both Newton's method and the secant method fail to bracket a root. The modified false position method can retain the advantages of both methods.

The secant method is often faster at approximating roots of nonlinear functions in comparison to bisection and false position. Unlike these two methods, the intervals $[a_k, b_k]$ do not have to be on opposite sides of the root and have a change of sign. Moreover, the slope of the secant line can become quite small, and a step can move far from the current point. The secant method can fail to find a root of a nonlinear function that has a small slope near the root because the secant line can jump a large amount.

For nice functions and guesses relatively close to the root, most of these methods require relatively few iterations before coming close to the root. However, there are pathological functions that can cause troubles for any of those methods. When selecting a method for solving a given nonlinear problem, one must consider many issues such as what you know about the behavior of the function, an interval $[a, b]$ satisfying $f(a)f(b) < 0$, the first derivative of the function, a good initial guess to the desired root, and so on.

Hybrid Schemes

In an effort to find the *best* algorithm for finding a zero of a given function, various hybrid methods have been developed. Some of these procedures combine the bisection method (used during the early iterations) with either the secant method or the Newton method. Also, adaptive schemes are used for monitoring the iterations and for carrying out stopping rules. More information on some hybrid secant-bisection methods and hybrid Newton-bisection methods with adaptive stopping rules can be found in Bus and Dekker [1975], Dekker [1969], Kahaner, Moler, and Nash [1989], and Novak, Ritter, and Woźniakowski [1995].

Fixed-Point Iteration

For a nonlinear equation $f(x) = 0$, we seek a point where the curve f intersects the x-axis ($y = 0$). An alternative approach is to recast the problem as a fixed-point problem $x = g(x)$ for a related nonlinear function g. For the fixed point problem, we seek a point where the curve g intersects the diagonal line $y = x$. A value of x such that $x = g(x)$ is a **fixed point** of g because x is unchanged when g is applied to it. Many iterative algorithms for solving a nonlinear equation $f(x) = 0$ are based on a fixed-point iterative method $x^{(n+1)} = g\left(x^{(n)}\right)$ where g has fixed points that are solutions of $f(x) = 0$. An initial starting value $x^{(0)}$

is selected, and the iterative method is applied repeatedly until it converges sufficiently well.

EXAMPLE 2 Apply the fixed-point procedure, where $g(x) = 1+2/x$, starting with $x^{(0)} = 1$, to compute a zero of the nonlinear function $f(x) = x^2 - x - 2$. Graphically, trace the convergence process.

Solution The fixed-point method is

$$x^{(n+1)} = 1 + \frac{2}{x^{(n)}}$$

Eight steps of the iterative algorithm are $x^{(0)} = 1$, $x^{(1)} = 3$, $x^{(2)} = 5/3$, $x^{(3)} = 11/5$, $x^{(4)} = 21/11$, $x^{(5)} = 43/21$, $x^{(6)} = 85/43$, $x^{(7)} = 171/85$, and $x^{(8)} = 341/171 \approx 1.99415$. In Figure 3.10, we see that these steps spiral into the fixed point 2.

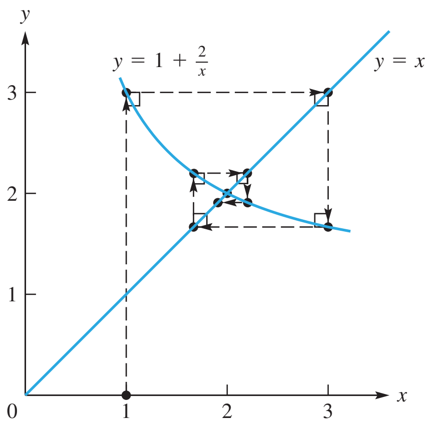


FIGURE 3.10 Fixed point iterations for $f(x) = x^2 - x - 2$

For a given nonlinear equation $f(x) = 0$, there may be many equivalent fixed-point problems $x = g(x)$ with different functions g, some better than others. A simple way to characterize the behavior of an iterative method $x^{(n+1)} = g\left(x^{(n)}\right)$ is *locally convergent* for x^* if $x^* = g(x^*)$ and $|g'(x^*)| < 1$. By *locally convergent*, we mean that there is an interval containing $x^{(0)}$ such that the fixed-point method converges for any starting value $x^{(0)}$ within that interval. If $|g'(x^*)| > 1$, then the fixed-point method diverges for any starting point $x^{(0)}$ other than x^*. Fixed-point iterative methods are used in standard practice for solving many science and engineering problems. In fact, the fixed-point theory can simplify the proof of the convergence of Newton's method.

Summary

(1) The **secant method** for finding a zero r of a function $f(x)$ is written as

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

for $n \geqq 1$, which requires two initial values x_0 and x_1. After the first step, only one new function evaluation per step is needed.

(2) After $n+1$ steps of the secant method, the error iterates $e_i = r - x_i$ obey the equation

$$e_{n+1} = -\frac{1}{2}\left(\frac{f''(\xi_n)}{f'(\zeta_n)}\right) e_n e_{n-1}$$

which leads to the approximation

$$|e_{n+1}| \approx C|e_n|^{1/2(1+\sqrt{5})} \approx C|e_n|^{1.62}$$

Therefore, the secant method has **superlinear convergence** behavior.

Additional References

For supplemental reading and study, see Barnsley [2006], Bus and Dekker [1975], Dekker [1969], Dennis and Schnabel [1983], Epureanu and Greenside [1998], Fauvel, Flood, Shortland, and Wilson [1988], Feder [1988], Ford [1995], Householder [1970], Kelley [1995], Lozier and Olver [1994], Nerinckx and Haegemans [1976], Novak, Ritter, and Woźniakowski [1995], Ortega and Rheinboldt [1970], Ostrowski [1966], Rabinowitz [1970], Traub [1964], Westfall [1995], and Ypma [1995].

Problems 3.3

[a]**1.** Calculate an approximate value for $4^{3/4}$ using one step of the secant method with $x_0 = 3$ and $x_1 = 2$.

2. If we use the secant method on $f(x) = x^3 - 2x + 2$ starting with $x_0 = 0$ and $x_1 = 1$, what is x_2?

[a]**3.** If the secant method is used on $f(x) = x^5 + x^3 + 3$ and if $x_{n-2} = 0$ and $x_{n-1} = 1$, what is x_n?

[a]**4.** If $x_{n+1} = x_n + (2 - e^{x_n})(x_n - x_{n-1})/(e^{x_n} - e^{x_{n-1}})$ with $x_0 = 0$ and $x_1 = 1$, what is $\lim_{n\to\infty} x_n$?

5. Using the bisection method, Newton's method, and the secant method, find the largest positive root correct to three decimal places of $x^3 - 5x + 3 = 0$. (All roots are in $[-3, +3]$.)

6. Prove that in the first analysis of the secant method, $\lambda_{n+1} - \alpha\lambda_n$ converges to zero as $n \to \infty$.

7. Establish Equation (10).

8. Write out the derivation of the order of convergence of the secant method that uses recurrence relations; that is, find the constants A and B in Equation (11), and fill in the details in arriving at Equation (12).

[a]**9.** What is the appropriate formula for finding square roots using the secant method? (Refer to Problem 3.2.1.)

10. The formula for the secant method can also be written as

$$x_{n+1} = \frac{x_{n-1} f(x_n) - x_n f(x_{n-1})}{f(x_n) - f(x_{n-1})}$$

Establish this, and explain why it is inferior to Equation (3) in a computer program.

11. Show that if the iterates in Newton's method converge to a point r for which $f'(r) \neq 0$, then $f(r) = 0$. Establish the same assertion for the secant method. *Hint:* In the latter, the Mean-Value Theorem of Differential Calculus is useful. This is the case $n = 0$ in Taylor's Theorem.

[a]**12.** A method of finding a zero of a given function f proceeds as follows. Two initial approximations x_0 and x_1 to the zero are chosen, the value of x_0 is fixed, and successive iterations are given by

$$x_{n+1} = x_n - \left(\frac{x_n - x_0}{f(x_n) - f(x_0)} \right) f(x_n)$$

This process will converge to a zero of f under certain conditions. Show that the rate of convergence to a simple zero is *linear* under some conditions.

13. Test the following sequences for different types of convergence (i.e., linear, superlinear, or quadratic), where $n = 1, 2, 3 \ldots.$

[a]**a.** $x_n = n^{-2}$ **b.** $x_n = 2^{-n}$ [a]**c.** $x_n = 2^{-2^n}$

d. $x_n = 2^{-a_n}$ with $a_0 = a_1 = 1$ and $a_{n+1} = a_n + a_{n-1}$ for $n \geqq 2$

14. This problem and the next three deal with the method of **functional iteration**. The method of functional iteration is as follows: Starting with any x_0, we define $x_{n+1} = f(x_n)$, where $n = 0, 1, 2, \ldots.$ Show that if f is continuous and if the sequence $\{x_n\}$ converges, then its limit is a fixed point of f.

[a]**15.** (Continuation) Show that if f is a function defined on the whole real line whose derivative satisfies $|f'(x)| \leqq c$ with a constant c less than 1, then the method of functional iteration produces a fixed point of f. *Hint:* In establishing this, the Mean-Value Theorem from Section 1.2 is helpful.

[a]**16.** (Continuation) With a calculator, try the method of functional iteration with $f(x) = x/2 + 1/x$, taking $x_0 = 1$. What is the limit of the resulting sequence?

[a]**17.** (Continuation) Using functional iteration, show that the equation $10 - 2x + \sin x = 0$ has a root. Locate the root approximately by drawing a graph. Starting with your approximate root, use functional iteration to obtain the root accurately by using a calculator. *Hint:* Write the equation in the form $x = 5 + \frac{1}{2}\sin x$.

18. Establish the first part of Equation (4) using Equation (5). *Hint:* Use the relationship between divided differences and derivatives from Section 4.2.

Computer Problems 3.3

[a]**1.** Use the secant method to find the zero near -0.5 of $f(x) = e^x - 3x^2$. This function also has a zero near 4. Find this positive zero by Newton's method.

2. Write

```
procedure Secant(f, x1, x2, epsi, delta, maxf, x, ierr)
```

which uses the secant method to solve $f(x) = 0$. The input parameters are as follows: f is the name of the given function; $x1$ and $x2$ are the initial estimates of the solution; *epsi* is a positive tolerance such that the iteration stops if the difference between two consecutive iterates is smaller than this value; *delta* is a positive tolerance such that the iteration stops if a function value is smaller in magnitude than this value; and *maxf* is a positive integer bounding the number of evaluations of the function allowed. The output parameters are as follows: x is the final estimate of the solution, and *ierr* is an integer error flag that indicates whether a tolerance test was violated. Test this routine using the function of Computer Problem 3.3.1. Print the final estimate of the solution and the value of the function at this point.

3. Find a zero of one of the functions given in the introduction of this chapter using one of the methods introduced in this chapter.

4. Write and test a recursive procedure for the secant method.

5. Rerun the example in this section with $x_0 = 0$ and $x_1 = 1$. Explain any unusual results.

6. Write a simple program to compare the secant method with Newton's method for finding a root of each function.

[a]**a.** $x^3 - 3x + 1$ with $x_0 = 2$ **b.** $x^3 - 2\sin x$ with $x_0 = \frac{1}{2}$

Use the x_1 value from Newton's method as the second starting point for the secant method. Print out each iteration for both methods.

[a]**7.** Write a simple program to find the root of $f(x) = x^3 + 2x^2 + 10x - 20$ using the secant method with starting values $x_0 = 2$ and $x_1 = 1$. Let it run at most 20 steps, and include a stopping test as well. Compare the number of steps needed here to the number needed in Newton's method. Is the convergence quadratic?

8. Test the secant method on the set of functions $f_k(x) = 2e^{-k}x + 1 - 3e^{-kx}$ for $k = 1, 2, 3, \ldots, 10$. Use the starting points 0 and 1 in each case.

[a]**9.** An example by Wilkinson [1963] shows that minute alterations in the coefficients of a polynomial may have massive effects on the roots. Let

$$f(x) = (x - 1)(x - 2)\cdots(x - 20)$$

which has become known as the **Wilkinson polynomial**. The zeros of f are, of course, the integers $1, 2, \ldots, 20$. Try to determine what happens to the zero $r = 20$ when the function is altered to $f(x) - 10^{-8}x^{19}$. *Hint:* The secant method in double precision will locate a zero in the interval [20, 21].

10. Test the secant method on an example in which r, $f'(r)$, and $f''(r)$ are known in advance. Monitor the ratios $e_{n+1}/(e_n e_{n-1})$ to see whether they converge to $-\frac{1}{2}f''(r)/f'(r)$. The function $f(x) = \arctan x$ is suitable for this experiment.

11. Using a function of your choice, verify numerically that the iterative method

$$x_{n+1} = x_n - \frac{f(x_n)}{\sqrt{[f'(x_n)]^2 - f(x_n)f''(x_n)}}$$

is cubically convergent at a simple root but only linearly convergent at a multiple root.

12. Test numerically whether **Olver's method**, given by

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} - \frac{1}{2}\frac{f''(x_n)}{f'(x_n)}\left[\frac{f(x_n)}{f'(x_n)}\right]^2$$

is cubically convergent to a root of f. Try to establish that it is.

13. (Continuation) Repeat for **Halley's method**

$$x_{n+1} = x_n - \frac{1}{a_n} \quad \text{with} \quad a_n = \frac{f'(x_n)}{f(x_n)} - \frac{1}{2}\left[\frac{f''(x_n)}{f'(x_n)}\right]$$

14. (**Moler-Morrison algorithm**) Computing an approximation for $\sqrt{x^2+y^2}$ does not require square roots. It can be done as follows:

```
real function $f(x, y)$
integer $n$;   real $a, b, c, x, y$
$f \leftarrow \max\{|x|, |y|\}$
$a \leftarrow \min\{|x|, |y|\}$
for $n = 1$ to 3 do
    $b \leftarrow (a/f)^2$
    $c \leftarrow b/(4+b)$
    $f \leftarrow f + 2cf$
    $a \leftarrow ca$
end for
end function $f$
```

Test the algorithm on some simple cases such as $(x, y) = (3, 4), (-5, 12)$, and $(7, -24)$. Then write a routine that uses the function $f(x, y)$ for approximating the **Euclidean norm** of a vector $x = (x_1, x_2, \dots, x_n)$; that is, the nonnegative number $\|x\| = \left(x_1^2 + x_2^2 + \cdots + x_n^2\right)^{1/2}$.

15. Study the following functions by starting with any initial value of x_0 in the domain $[0, 2]$ and iterating $x_{n+1} = F(x_n)$. First use a calculator and then a computer. Explain the results.

a. Use the **tent function**

$$F(x) = \begin{cases} 2x & \text{if } 2x < 1 \\ 2x - 1 & \text{if } 2x \geqq 1 \end{cases}$$

b. Repeat using the function

$$F(x) = 10x \text{ (modulo 1)}$$

Hint: Don't be surprised by chaotic behavior. The interested reader can learn more about the dynamics of one-dimensional maps by reading papers such as the one by Bassien [1998].

16. Show how the secant method can be used to solve systems of equations such as those in Computer Problems 3.2.21–3.2.23.

17. (**Student research project**) **Muller's method** is an algorithm for computing solutions of an equation $f(x) = 0$. It is similar to the secant method in that it replaces f locally by a simple function, and finds a root of it. Naturally, this step is repeated. The simple function chosen in Muller's method is a quadratic polynomial, p, that interpolates f at the three most recent points. After p has been determined, its roots are computed, and one of them is chosen as the next point in the sequence. Since this quadratic function may have complex roots, the algorithm should be programmed with this in mind. Suppose that points x_{n-2}, x_{n-1}, and x_n have been computed. Set

$$p(x) = a(x - x_n)(x - x_{n-1}) + b(x - x_n) + c$$

where a, b, and c are determined so that p interpolates f at the three points mentioned previously. Then find the roots of p and take x_{n+1} to be the root of p closest to x_n. At the beginning, three points must be furnished by the user. Program the method, allowing for complex numbers throughout. Test your program on the example

$$p(x) = x^3 + x^2 - 10x - 10$$

If the first three points are 1, 2, 3, then you should find that the polynomial is $p(x) = 7(x - 3)(x - 2) + 14(x - 3) - 4$ and $x_4 = 3.17971\,086$. Next, test your code on a polynomial having real coefficients but some complex roots.

18. Program and test the code for the secant algorithm after incorporating the stopping criterion described in the text.

19. Using mathematical software such as Matlab, Mathematica, and Maple, find the real zero of the polynomial $p(x) = x^5 + x^3 + 3$. Attain more digits of accuracy than shown in the solution to Example 1 in the text.

20. (Continuation) Using mathematical software that allows for complex roots, find all zeros of the polynomial.

21. Program a hybrid method for solving several of the nonlinear problems given as examples in the text, and compare your results with those given.

22. Find the fixed points for each of the following functions:

a. $e^x + 1$ **b.** $e^{-x} - x$ **c.** $x^2 - 4\sin x$ **d.** $x^3 + 6x^2 + 11x - 6$ **e.** $\sin x$

23. For the nonlinear equation $f(x) = x^2 - x - 2 = 0$ with roots 1 and 2, write four fixed-point problems $x = g(x)$ that are equivalent. Plot all of these, and show that they all intersect the line $x = y$. Also, plot the convergence steps of each of these fixed-point iterations for different starting values $x^{(0)}$. Show that the behavior of these fixed-point schemes can vary wildly: slow convergence, fast convergence, and divergence.