CHAPTER 6

Open Methods

For the bracketing methods in the previous chapter, the root is located within an interval prescribed by a lower and an upper bound. Repeated application of these methods always results in closer estimates of the true value of the root. Such methods are said to be *convergent* because they move closer to the truth as the computation progresses (Fig. 6.1*a*).

In contrast, the *open methods* described in this chapter are based on formulas that require only a single starting value of x or two starting values that do not necessarily bracket

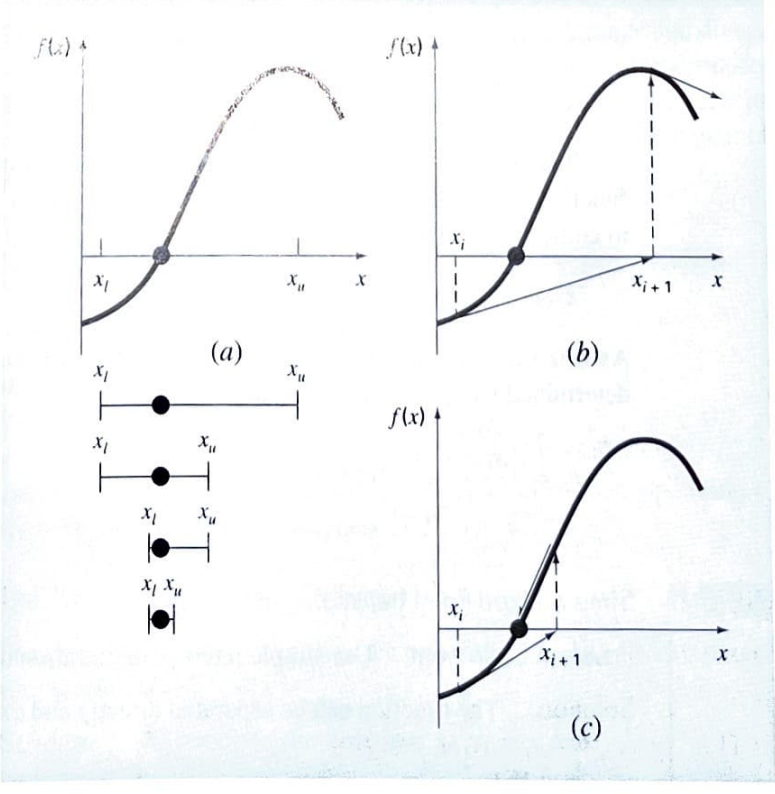


FIGURE 6.1
Graphical depiction of the fundamental difference between the (*a*) bracketing and (*b*) and (*c*) open methods for root location. In (*a*), which is the bisection method, the root is constrained within the interval prescribed by x_l and x_u. In contrast, for the open method depicted in (*b*) and (*c*), a formula is used to project from x_i to x_{i+1} in an iterative fashion. Thus, the method can either (*b*) diverge or (*c*) converge rapidly, depending on the value of the initial guess.

the root. As such, they sometimes *diverge* or move away from the true root as the computation progresses (Fig. 6.1*b*). However, when the open methods converge (Fig. 6.1*c*), they usually do so much more quickly than the bracketing methods. We will begin our discussion of open techniques with a simple version that is useful for illustrating their general form and also for demonstrating the concept of convergence.

6.1 SIMPLE FIXED-POINT ITERATION

As mentioned above, open methods employ a formula to predict the root. Such a formula can be developed for simple *fixed-point iteration* (or, as it is also called, one-point iteration or successive substitution) by rearranging the function $f(x) = 0$ so that x is on the left-hand side of the equation:

$$x = g(x) \tag{6.1}$$

This transformation can be accomplished either by algebraic manipulation or by simply adding x to both sides of the original equation. For example,

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

can be simply manipulated to yield

$$x = \frac{x^2 + 3}{2}$$

whereas $\sin x = 0$ could be put into the form of Eq. (6.1) by adding x to both sides to yield

$$x = \sin x + x$$

The utility of Eq. (6.1) is that it provides a formula to predict a new value of x as a function of an old value of x. Thus, given an initial guess at the root x_i, Eq. (6.1) can be used to compute a new estimate x_{i+1} as expressed by the iterative formula

$$x_{i+1} = g(x_i) \tag{6.2}$$

As with other iterative formulas in this book, the approximate error for this equation can be determined using the error estimator [Eq. (3.5)]:

$$\varepsilon_a = \left| \frac{x_{i+1} - x_i}{x_{i+1}} \right| 100\%$$

EXAMPLE 6.1 Simple Fixed-Point Iteration

Problem Statement. Use simple fixed-point iteration to locate the root of $f(x) = e^{-x} - x$.

Solution. The function can be separated directly and expressed in the form of Eq. (6.2) as

$$x_{i+1} = e^{-x_i}$$

Starting with an initial guess of $x_0 = 0$, this iterative equation can be applied to compute

i	x_i	ε_a (%)	ε_t (%)
0	0		100.0
1	1.000000	100.0	76.3
2	0.367879	171.8	35.1
3	0.692201	46.9	22.1
4	0.500473	38.3	11.8
5	0.606244	17.4	6.89
6	0.545396	11.2	3.83
7	0.579612	5.90	2.20
8	0.560115	3.48	1.24
9	0.571143	1.93	0.705
10	0.564879	1.11	0.399

Thus, each iteration brings the estimate closer to the true value of the root: 0.56714329.

6.1.1 Convergence

Notice that the true percent relative error for each iteration of Example 6.1 is roughly proportional (by a factor of about 0.5 to 0.6) to the error from the previous iteration. This property, called *linear convergence*, is characteristic of fixed-point iteration.

Aside from the "rate" of convergence, we must comment at this point about the "possibility" of convergence. The concepts of convergence and divergence can be depicted graphically. Recall that in Sec. 5.1, we graphed a function to visualize its structure and behavior (Example 5.1). Such an approach is employed in Fig. 6.2*a* for the function $f(x) = e^{-x} - x$. An alternative graphical approach is to separate the equation into two component parts, as in

$$f_1(x) = f_2(x)$$

Then the two equations

$$y_1 = f_1(x) \tag{6.3}$$

and

$$y_2 = f_2(x) \tag{6.4}$$

can be plotted separately (Fig. 6.2*b*). The x values corresponding to the intersections of these functions represent the roots of $f(x) = 0$.

EXAMPLE 6.2 The Two-Curve Graphical Method

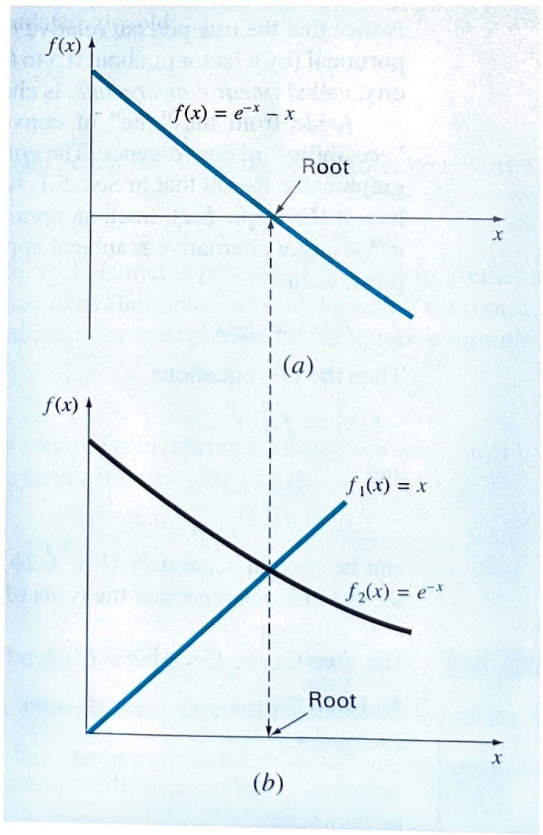
Problem Statement. Separate the equation $e^{-x} - x = 0$ into two parts and determine its root graphically.

Solution. Reformulate the equation as $y_1 = x$ and $y_2 = e^{-x}$. The following values can be computed:

x	y_1	y_2
0.0	0.0	1.000
0.2	0.2	0.819
0.4	0.4	0.670
0.6	0.6	0.549
0.8	0.8	0.449
1.0	1.0	0.368

These points are plotted in Fig. 6.2*b*. The intersection of the two curves indicates a root estimate of approximately $x = 0.57$, which corresponds to the point where the single curve in Fig. 6.2*a* crosses the x axis.

FIGURE 6.2
Two alternative graphical methods for determining the root of $f(x) = e^{-x} - x$. (*a*) Root at the point where it crosses the x axis; (*b*) root at the intersection of the component functions.



The two-curve method can now be used to illustrate the convergence and divergence of fixed-point iteration. First, Eq. (6.1) can be re-expressed as a pair of equations $y_1 = x$ and $y_2 = g(x)$. These two equations can then be plotted separately. As was the case with Eqs. (6.3) and (6.4), the roots of $f(x) = 0$ correspond to the abscissa value at the intersection of the two curves. The function $y_1 = x$ and four different shapes for $y_2 = g(x)$ are plotted in Fig. 6.3.

For the first case (Fig. 6.3*a*), the initial guess of x_0 is used to determine the corresponding point on the y_2 curve $[x_0, g(x_0)]$. The point (x_1, x_1) is located by moving left horizontally to the y_1 curve. These movements are equivalent to the first iteration in the fixed-point method:

$$x_1 = g(x_0)$$

Thus, in both the equation and in the plot, a starting value of x_0 is used to obtain an estimate of x_1. The next iteration consists of moving to $[x_1, g(x_1)]$ and then to (x_2, x_2). This iteration

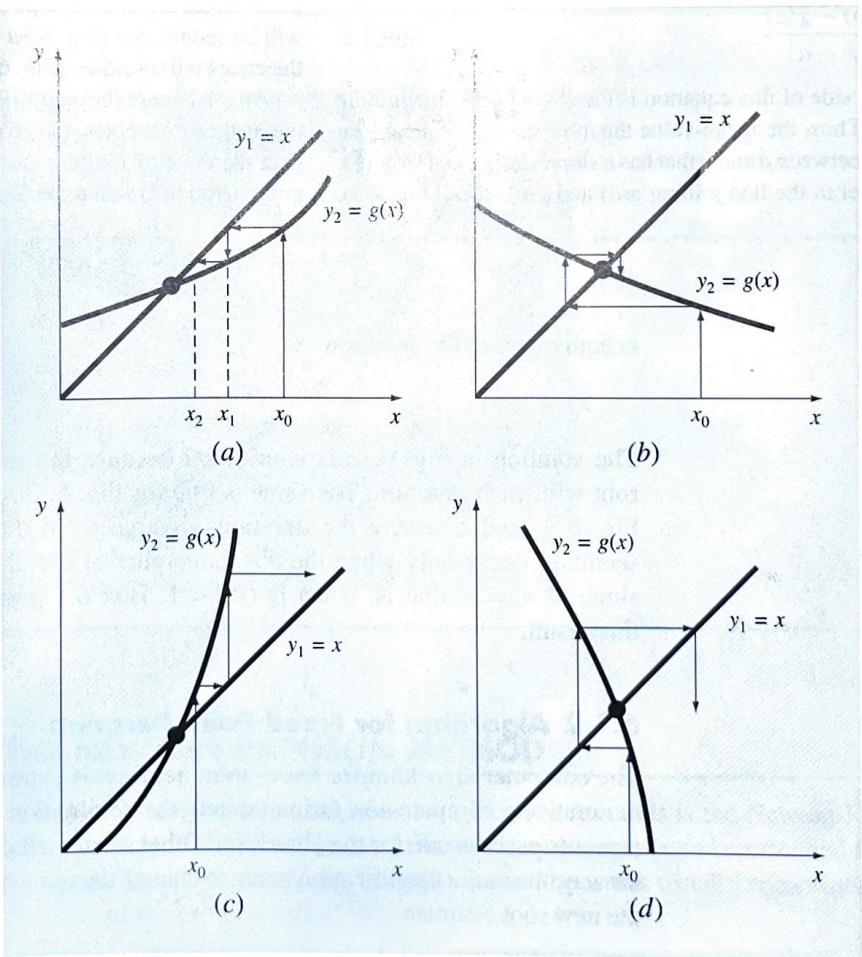


FIGURE 6.3
Graphical depiction of (*a*) and (*b*) convergence and (*c*) and (*d*) divergence of simple fixed-point iteration. Graphs (*a*) and (*c*) are called monotone patterns, whereas (*b*) and (*d*) are called oscillating or spiral patterns. Note that convergence occurs when $|g'(x)| < 1$.

Box 6.1 Convergence of Fixed-Point Iteration

From studying Fig. 6.3, it should be clear that fixed-point iteration converges if, in the region of interest, $|g'(x)| < 1$. In other words, convergence occurs if the magnitude of the slope of $g(x)$ is less than the slope of the line $f(x) = x$. This observation can be demonstrated theoretically. Recall that the iterative equation is

$$x_{i+1} = g(x_i)$$

Suppose that the true solution is

$$x_r = g(x_r)$$

Subtracting these equations yields

$$x_r - x_{i+1} = g(x_r) - g(x_i) \tag{B6.1.1}$$

The *derivative mean-value theorem* (recall Sec. 4.1.1) states that if a function $g(x)$ and its first derivative are continuous over an interval $a \le x \le b$, then there exists at least one value of $x = \xi$ within the interval such that

$$g'(\xi) = \frac{g(b) - g(a)}{b - a} \tag{B6.1.2}$$

The right-hand side of this equation is the slope of the line joining $g(a)$ and $g(b)$. Thus, the mean-value theorem states that there is at least one point between a and b that has a slope, designated by $g'(\xi)$, which is parallel to the line joining $g(a)$ and $g(b)$ (recall Fig. 4.3).

Now, if we let $a = x_i$ and $b = x_r$, the right-hand side of Eq. (B6.1.1) can be expressed as

$$g(x_r) - g(x_i) = (x_r - x_i)g'(\xi)$$

where ξ is somewhere between x_i and x_r. This result can then be substituted into Eq. (B6.1.1) to yield

$$x_r - x_{i+1} = (x_r - x_i)g'(\xi) \tag{B6.1.3}$$

If the true error for iteration i is defined as

$$E_{t,i} = x_r - x_i$$

then Eq. (B6.1.3) becomes

$$E_{t,i+1} = g'(\xi)E_{t,i}$$

Consequently, if $|g'(x)| < 1$, the errors decrease with each iteration. For $|g'(x)| > 1$, the errors grow. Notice also that if the derivative is positive, the errors will be positive, and hence the iterative solution will be monotonic (Fig. 6.3*a* and *c*). If the derivative is negative, the errors will oscillate (Fig. 6.3*b* and *d*).

An offshoot of the analysis is that it also demonstrates that when the method converges, the error is roughly proportional to and less than the error of the previous step. For this reason, simple fixed-point iteration is said to be *linearly convergent*.

is equivalent to the equation

$$x_2 = g(x_1)$$

The solution in Fig. 6.3*a* is *convergent* because the estimates of x move closer to the root with each iteration. The same is true for Fig. 6.3*b*. However, this is not the case for Fig. 6.3*c* and *d*, where the iterations diverge from the root. Notice that convergence seems to occur only when the absolute value of the slope of $y_2 = g(x)$ is less than the slope of $y_1 = x$, that is, when $|g'(x)| < 1$. Box 6.1 provides a theoretical derivation of this result.

6.1.2 Algorithm for Fixed-Point Iteration

The computer algorithm for fixed-point iteration is extremely simple. It consists of a loop to iteratively compute new estimates until the termination criterion has been met. Figure 6.4 presents pseudocode for the algorithm. Other open methods can be programmed in a similar way, the major modification being to change the iterative formula that is used to compute the new root estimate.

```
FUNCTION Fixpt(x0, es, imax, iter, ea)
  xr = x0
  iter = 0
  DO
    xrold = xr
    xr = g(xrold)
    iter = iter + 1
    IF xr ≠ 0 THEN
      ea = |(xr − xrold)/xr| · 100
    END IF
    IF ea < es OR iter ≥ imax EXIT
  END DO
  Fixpt = xr
END Fixpt
```

FIGURE 6.4
Pseudocode for fixed-point iteration. Note that other open methods can be cast in this general format.

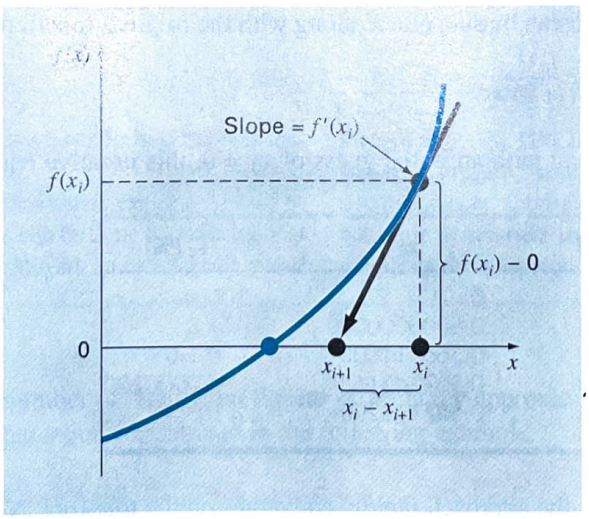


FIGURE 6.5
Graphical depiction of the Newton-Raphson method. A tangent to the function of x_i [that is, $f'(x_i)$] is extrapolated down to the x axis to provide an estimate of the root at x_{i+1}.

6.2 THE NEWTON-RAPHSON METHOD

Perhaps the most widely used of all root-locating formulas is the Newton-Raphson equation (Fig. 6.5). If the initial guess at the root is x_i, a tangent can be extended from the point $[x_i, f(x_i)]$. The point where this tangent crosses the x axis usually represents an improved estimate of the root.

The Newton-Raphson method can be derived on the basis of this geometrical interpretation (an alternative method based on the Taylor series is described in Box 6.2). As in Fig. 6.5, the first derivative at x is equivalent to the slope:

$$f'(x_i) = \frac{f(x_i) - 0}{x_i - x_{i+1}} \tag{6.5}$$

which can be rearranged to yield

$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)} \tag{6.6}$$

which is called the *Newton-Raphson formula.*

EXAMPLE 6.3 Newton-Raphson Method

Problem Statement. Use the Newton-Raphson method to estimate the root of $f(x) = e^{-x} - x$, employing an initial guess of $x_0 = 0$.

Solution. The first derivative of the function can be evaluated as

$$f'(x) = -e^{-x} - 1$$

which can be substituted along with the original function into Eq. (6.6) to give

$$x_{i+1} = x_i - \frac{e^{-x_i} - x_i}{-e^{-x_i} - 1}$$

Starting with an initial guess of $x_0 = 0$, this iterative equation can be applied to compute

i	x_i	ε_t (%)
0	0	100
1	0.500000000	11.8
2	0.566311003	0.147
3	0.567143165	0.0000220
4	0.567143290	$< 10^{-8}$

Thus, the approach rapidly converges on the true root. Notice that the true percent relative error at each iteration decreases much faster than it does in simple fixed-point iteration (compare with Example 6.1).

6.2.1 Termination Criteria and Error Estimates

As with other root-location methods, Eq. (3.5) can be used as a termination criterion. In addition, however, the Taylor series derivation of the method (Box 6.2) provides theoretical insight regarding the rate of convergence as expressed by $E_{i+1} = O(E_i^2)$. Thus the error should be roughly proportional to the square of the previous error. In other words, the

Box 6.2 Derivation and Error Analysis of the Newton-Raphson Method

Aside from the geometric derivation [Eqs. (6.5) and (6.6)], the Newton-Raphson method may also be developed from the Taylor series expansion. This alternative derivation is useful in that it also provides insight into the rate of convergence of the method.

Recall from Chap. 4 that the Taylor series expansion can be represented as

$$f(x_{i+1}) = f(x_i) + f'(x_i)(x_{i+1} - x_i) + \frac{f''(\xi)}{2!}(x_{i+1} - x_i)^2 \tag{B6.2.1}$$

where ξ lies somewhere in the interval from x_i to x_{i+1}. An approximate version is obtainable by truncating the series after the first derivative term:

$$f(x_{i+1}) \cong f(x_i) + f'(x_i)(x_{i+1} - x_i)$$

At the intersection with the x axis, $f(x_{i+1})$ would be equal to zero, or

$$0 = f(x_i) + f'(x_i)(x_{i+1} - x_i) \tag{B6.2.2}$$

which can be solved for

$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)}$$

which is identical to Eq. (6.6). Thus, we have derived the Newton-Raphson formula using a Taylor series.

Aside from the derivation, the Taylor series can also be used to estimate the error of the formula. This can be done by realizing that if the complete Taylor series were employed, an exact result would be obtained. For this situation $x_{i+1} = x_r$, where x is the true value of the root. Substituting this value along with $f(x_r) = 0$ into Eq. (B6.2.1) yields

$$0 = f(x_i) + f'(x_i)(x_r - x_i) + \frac{f''(\xi)}{2!}(x_r - x_i)^2 \tag{B6.2.3}$$

Equation (B6.2.2) can be subtracted from Eq. (B6.2.3) to give

$$0 = f'(x_i)(x_r - x_{i+1}) + \frac{f''(\xi)}{2!}(x_r - x_i)^2 \tag{B6.2.4}$$

Now, realize that the error is equal to the discrepancy between x_{i+1} and the true value x_r, as in

$$E_{t,i+1} = x_r - x_{i+1}$$

and Eq. (B6.2.4) can be expressed as

$$0 = f'(x_i)E_{t,i+1} + \frac{f''(\xi)}{2!}E_{t,i}^2 \tag{B6.2.5}$$

If we assume convergence, both x_i and ξ should eventually be approximated by the root x_r, and Eq. (B6.2.5) can be rearranged to yield

$$E_{t,i+1} \cong \frac{-f''(x_r)}{2f'(x_r)}E_{t,i}^2 \tag{B6.2.6}$$

According to Eq. (B6.2.6), the error is roughly proportional to the square of the previous error. This means that the number of correct decimal places approximately doubles with each iteration. Such behavior is referred to as *quadratic convergence*. Example 6.4 manifests this property.

number of significant figures of accuracy approximately doubles with each iteration. This behavior is examined in the following example.

EXAMPLE 6.4 Error Analysis of Newton-Raphson Method

Problem Statement. As derived in Box 6.2, the Newton-Raphson method is quadratically convergent. That is, the error is roughly proportional to the square of the previous error, as in

$$E_{t,i+1} \cong \frac{-f''(x_r)}{2f'(x_r)}E_{t,i}^2 \tag{E6.4.1}$$

Examine this formula and see if it applies to the results of Example 6.3.

Solution. The first derivative of $f(x) = e^{-x} - x$ is

$$f'(x) = -e^{-x} - 1$$

which can be evaluated at $x_r = 0.56714329$ as $f'(0.56714329) = -1.56714329$. The second derivative is

$$f''(x) = e^{-x}$$

which can be evaluated as $f''(0.56714329) = 0.56714329$. These results can be substituted into Eq. (E6.4.1) to yield

$$E_{t,i+1} \cong -\frac{0.56714329}{2(-1.56714329)} E_{t,i}^2 = 0.18095 E_{t,i}^2$$

From Example 6.3, the initial error was $E_{t,0} = 0.56714329$, which can be substituted into the error equation to predict

$$E_{t,1} \cong 0.18095(0.56714329)^2 = 0.0582$$

which is close to the true error of 0.06714329. For the next iteration,

$$E_{t,2} \cong 0.18095(0.06714329)^2 = 0.0008158$$

which also compares favorably with the true error of 0.0008323. For the third iteration,

$$E_{t,3} \cong 0.18095(0.0008323)^2 = 0.000000125$$

which is the error obtained in Example 6.3. The error estimate improves in this manner because, as we come closer to the root, x and ξ are better approximated by x_r [recall our assumption in going from Eq. (B6.2.5) to Eq. (B6.2.6) in Box 6.2]. Finally,

$$E_{t,4} \cong 0.18095(0.000000125)^2 = 2.83 \times 10^{-15}$$

Thus, this example illustrates that the error of the Newton-Raphson method for this case is, in fact, roughly proportional (by a factor of 0.18095) to the square of the error of the previous iteration.

6.2.2 Pitfalls of the Newton-Raphson Method

Although the Newton-Raphson method is often very efficient, there are situations where it performs poorly. A special case—multiple roots—will be addressed later in this chapter. However, even when dealing with simple roots, difficulties can also arise, as in the following example.

EXAMPLE 6.5 Example of a Slowly Converging Function with Newton-Raphson

Problem Statement. Determine the positive root of $f(x) = x^{10} - 1$ using the Newton-Raphson method and an initial guess of $x = 0.5$.

Solution. The Newton-Raphson formula for this case is

$$x_{i+1} = x_i - \frac{x_i^{10} - 1}{10x_i^9}$$

which can be used to compute

Iteration	x
0	0.5
1	51.65
2	46.485
3	41.8365
4	37.65285
5	33.887565
.	
.	
.	
∞	1.0000000

Thus, after the first poor prediction, the technique is converging on the true root of 1, but at a very slow rate.

Aside from slow convergence due to the nature of the function, other difficulties can arise, as illustrated in Fig. 6.6. For example, Fig. 6.6*a* depicts the case where an inflection point [that is, $f''(x) = 0$] occurs in the vicinity of a root. Notice that iterations beginning at x_0 progressively diverge from the root. Figure 6.6*b* illustrates the tendency of the Newton-Raphson technique to oscillate around a local maximum or minimum. Such oscillations may persist, or as in Fig. 6.6*b*, a near-zero slope is reached, whereupon the solution is sent far from the area of interest. Figure 6.6*c* shows how an initial guess that is close to one root can jump to a location several roots away. This tendency to move away from the area of interest is because near-zero slopes are encountered. Obviously, a zero slope [$f'(x) = 0$] is truly a disaster because it causes division by zero in the Newton-Raphson formula [Eq. (6.6)]. Graphically (see Fig 6.6*d*), it means that the solution shoots off horizontally and never hits the x axis.

Thus, there is no general convergence criterion for Newton-Raphson. Its convergence depends on the nature of the function and on the accuracy of the initial guess. The only remedy is to have an initial guess that is "sufficiently" close to the root. And for some functions, no guess will work! Good guesses are usually predicated on knowledge of the physical problem setting or on devices such as graphs that provide insight into the behavior of the solution. The lack of a general convergence criterion also suggests that good computer software should be designed to recognize slow convergence or divergence. The next section addresses some of these issues.

6.2.3 Algorithm for Newton-Raphson

An algorithm for the Newton-Raphson method is readily obtained by substituting Eq. (6.6) for the predictive formula [Eq. (6.2)] in Fig. 6.4. Note, however, that the program must also be modified to compute the first derivative. This can be simply accomplished by the inclusion of a user-defined function.

Additionally, in light of the foregoing discussion of potential problems of the Newton-Raphson method, the program would be improved by incorporating several additional features:

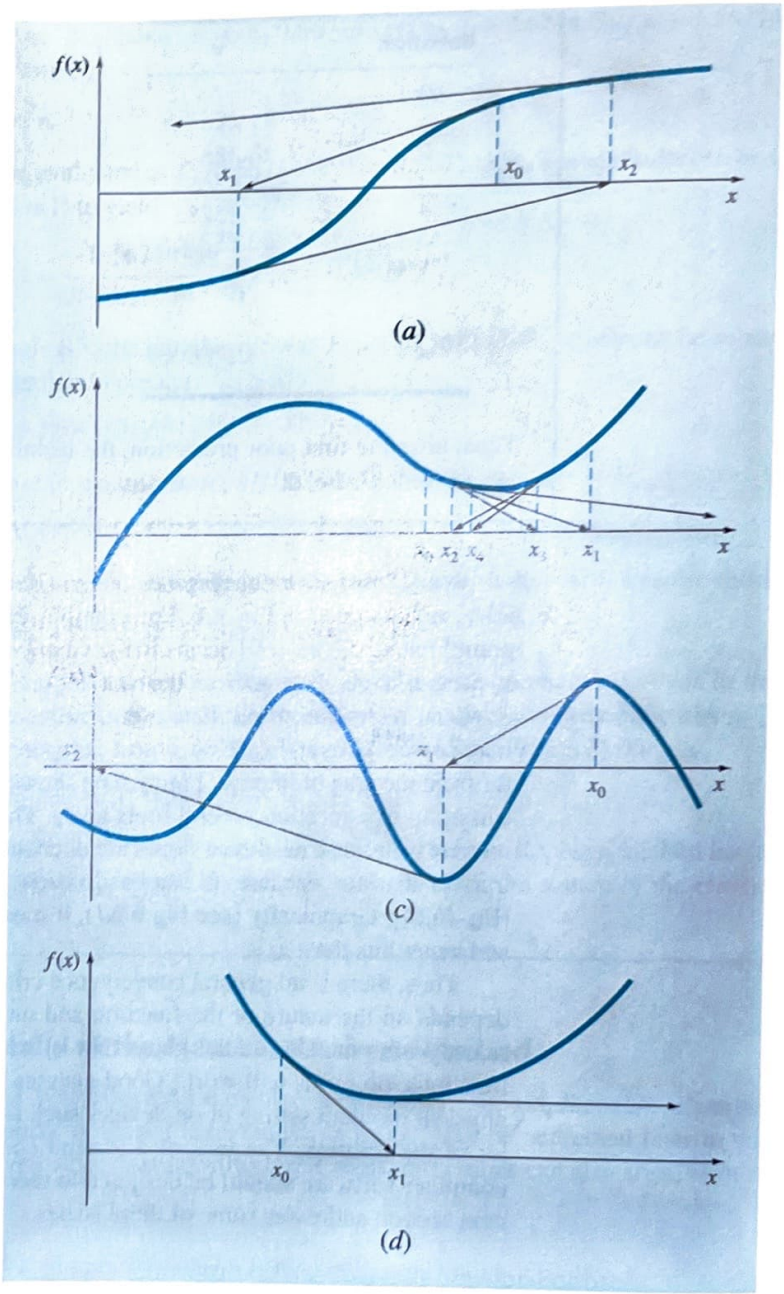


FIGURE 6.6
Four cases where the Newton-Raphson method exhibits poor convergence.

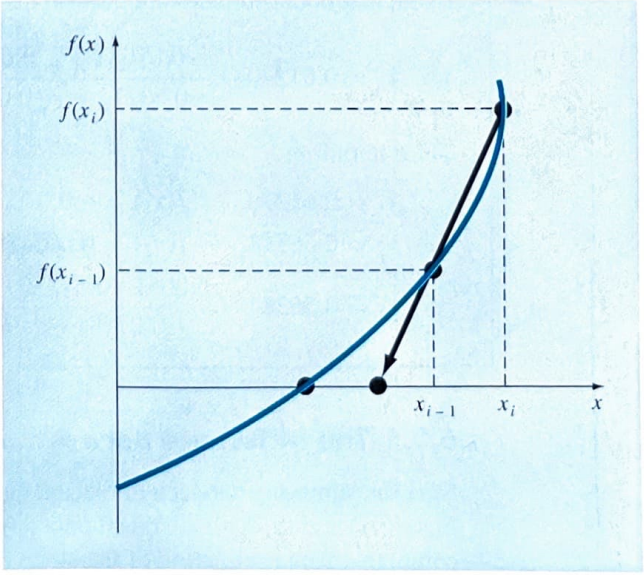
1. A plotting routine should be included in the program.
2. At the end of the computation, the final root estimate should always be substituted into the original function to compute whether the result is close to zero. This check partially guards against those cases where slow or oscillating convergence may lead to a small value of ε_a while the solution is still far from a root.
3. The program should always include an upper limit on the number of iterations to guard against oscillating, slowly convergent, or divergent solutions that could persist interminably.
4. The program should alert the user and take account of the possibility that $f'(x)$ might equal zero at any time during the computation.

6.3 THE SECANT METHOD

A potential problem in implementing the Newton-Raphson method is the evaluation of the derivative. Although this is not inconvenient for polynomials and many other functions, there are certain functions whose derivatives may be extremely difficult or inconvenient to evaluate. For these cases, the derivative can be approximated by a backward finite divided difference, as in (Fig. 6.7)

$$f'(x_i) \cong \frac{f(x_{i-1}) - f(x_i)}{x_{i-1} - x_i}$$

FIGURE 6.7
Graphical depiction of the secant method. This technique is similar to the Newton-Raphson technique (Fig. 6.5) in the sense that an estimate of the root is predicted by extrapolating a tangent of the function to the x axis. However, the secant method uses a difference rather than a derivative to estimate the slope.



This approximation can be substituted into Eq. (6.6) to yield the following iterative equation:

$$x_{i+1} = x_i - \frac{f(x_i)(x_{i-1} - x_i)}{f(x_{i-1}) - f(x_i)} \tag{6.7}$$

Equation (6.7) is the formula for the *secant method.* Notice that the approach requires two initial estimates of x. However, because $f(x)$ is not required to change signs between the estimates, it is not classified as a bracketing method.

EXAMPLE 6.6 The Secant Method

Problem Statement. Use the secant method to estimate the root of $f(x) = e^{-x} - x$. Start with initial estimates of $x_{-1} = 0$ and $x_0 = 1.0$.

Solution. Recall that the true root is 0.56714329. . . .

First iteration:

$$x_{-1} = 0 \qquad f(x_{-1}) = 1.00000$$

$$x_0 = 1 \qquad f(x_0) = -0.63212$$

$$x_1 = 1 - \frac{-0.63212(0 - 1)}{1 - (-0.63212)} = 0.61270 \qquad \varepsilon_t = 8.0\%$$

Second iteration:

$$x_0 = 1 \qquad f(x_0) = -0.63212$$

$$x_1 = 0.61270 \qquad f(x_1) = -0.07081$$

(Note that both estimates are now on the same side of the root.)

$$x_2 = 0.61270 - \frac{-0.07081(1 - 0.61270)}{-0.63212 - (-0.07081)} = 0.56384 \qquad \varepsilon_t = 0.58\%$$

Third iteration:

$$x_1 = 0.61270 \qquad f(x_1) = -0.07081$$

$$x_2 = 0.56384 \qquad f(x_2) = 0.00518$$

$$x_3 = 0.56384 - \frac{0.00518(0.61270 - 0.56384)}{-0.07081 - (-0.00518)} = 0.56717 \qquad \varepsilon_t = 0.0048\%$$

6.3.1 The Difference Between the Secant and False-Position Methods

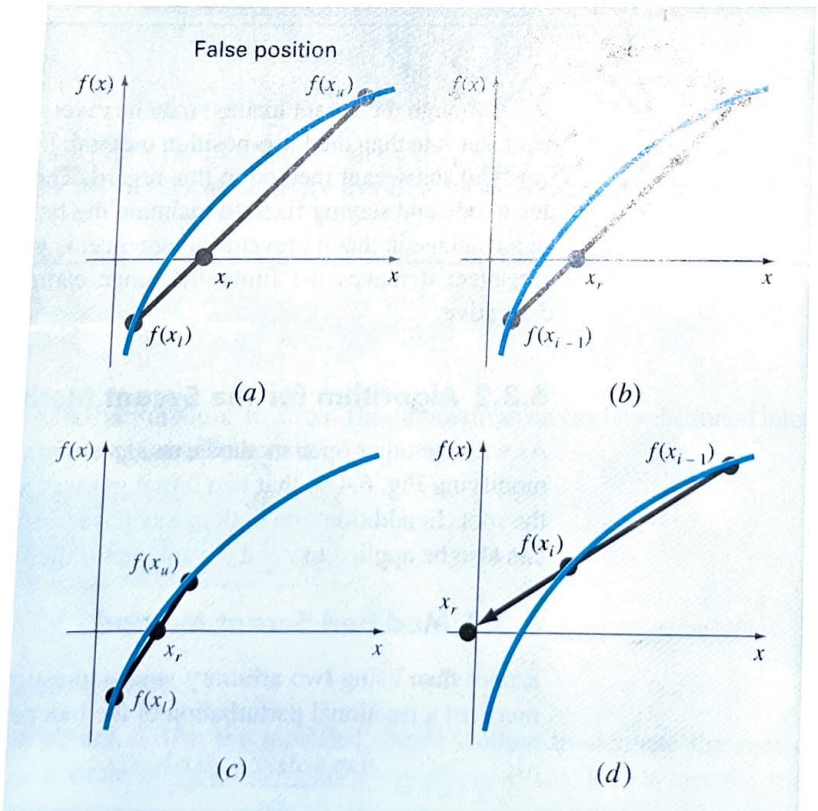
Note the similarity between the secant method and the false-position method. For example, Eqs. (6.7) and (5.7) are identical on a term-by-term basis. Both use two initial estimates to compute an approximation of the slope of the function that is used to project to the x axis for a new estimate of the root. However, a critical difference between the methods is how

one of the initial values is replaced by the new estimate. Recall that in the false-position method the latest estimate of the root replaces whichever of the original values yielded a function value with the same sign as $f(x_r)$. Consequently, the two estimates always bracket the root. Therefore, for all practical purposes, the method always converges because the root is kept within the bracket. In contrast, the secant method replaces the values in strict sequence, with the new value x_{i+1} replacing x_i and x_i replacing x_{i-1}. As a result, the two values can sometimes lie on the same side of the root. For certain cases, this can lead to divergence.

EXAMPLE 6.7 Comparison of Convergence of the Secant and False-Position Techniques

Problem Statement. Use the false-position and secant methods to estimate the root of $f(x) = \ln x$. Start the computation with values of $x_l = x_{i-1} = 0.5$ and $x_u = x_i = 5.0$.

FIGURE 6.8
Comparison of the false-position and the secant methods. The first iterations (*a*) and (*b*) for both techniques are identical. However, for the second iterations (*c*) and (*d*), the points used differ. As a consequence, the secant method can diverge, as indicated in (*d*).



Solution. For the false-position method, the use of Eq. (5.7) and the bracketing criterion for replacing estimates results in the following iterations:

Iteration	x_l	x_u	x_r
1	0.5	5.0	1.8546
2	0.5	1.8546	1.2163
3	0.5	1.2163	1.0585

As can be seen (Fig. 6.8*a* and *c*), the estimates are converging on the true root which is equal to 1.

For the secant method, using Eq. (6.7) and the sequential criterion for replacing estimates results in

Iteration	x_{i-1}	x_i	x_{i+1}
1	0.5	5.0	1.8546
2	5.0	1.8546	−0.10438

As in Fig. 6.8*d*, the approach is divergent.

Although the secant method may be divergent, when it converges it usually does so at a quicker rate than the false-position method. For instance, Fig. 6.9 demonstrates the superiority of the secant method in this regard. The inferiority of the false-position method is due to one end staying fixed to maintain the bracketing of the root. This property, which is an advantage in that it prevents divergence, is a shortcoming with regard to the rate of convergence; it makes the finite-difference estimate a less-accurate approximation of the derivative.

6.3.2 Algorithm for the Secant Method

As with the other open methods, an algorithm for the secant method is obtained simply by modifying Fig. 6.4 so that two initial guesses are input and by using Eq. (6.7) to calculate the root. In addition, the options suggested in Sec. 6.2.3 for the Newton-Raphson method can also be applied to good advantage for the secant program.

6.3.3 Modified Secant Method

Rather than using two arbitrary values to estimate the derivative, an alternative approach involves a fractional perturbation of the independent variable to estimate $f'(x)$,

$$f'(x_i) \cong \frac{f(x_i + \delta x_i) - f(x_i)}{\delta x_i}$$

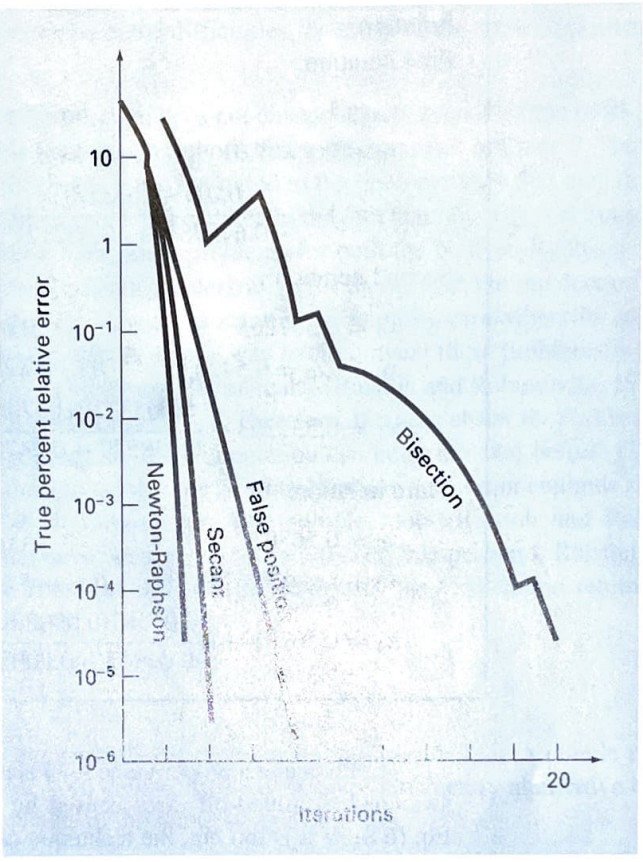


FIGURE 6.9
Comparison of the true percent relative errors ε_t for the methods to determine the roots of $f(x) = e^{-x} - x$.

where δ = a small perturbation fraction. This approximation can be substituted into Eq. (6.6) to yield the following iterative equation:

$$x_{i+1} = x_i - \frac{\delta x_i f(x_i)}{f(x_i + \delta x_i) - f(x_i)} \tag{6.8}$$

EXAMPLE 6.8 Modified Secant Method

Problem Statement. Use the modified secant method to estimate the root of $f(x) = e^{-x} - x$. Use a value of 0.01 for δ and start with $x_0 = 1.0$. Recall that the true root is 0.56714329. . . .

Solution.
First iteration:

$$x_0 = 1 \qquad f(x_0) = -0.63212$$
$$x_0 + \delta x_0 = 1.01 \qquad f(x_0 + \delta x_0) = -0.64578$$
$$x_1 = 1 - \frac{0.01(-0.63212)}{-0.64578 - (-0.63212)} = 0.537263 \qquad |\varepsilon_t| = 5.3\%$$

Second iteration:

$$x_0 = 0.537263 \qquad f(x_0) = 0.047083$$
$$x_0 + \delta x_0 = 0.542635 \qquad f(x_0 + \delta x_0) = 0.038579$$
$$x_1 = 0.537263 - \frac{0.005373(0.047083)}{0.038579 - 0.047083} = 0.56701 \qquad |\varepsilon_t| = 0.0236\%$$

Third iteration:

$$x_0 = 0.56701 \qquad f(x_0) = 0.000209$$
$$x_0 + \delta x_0 = 0.572680 \qquad f(x_0 + \delta x_0) = -0.00867$$
$$x_1 = 0.56701 - \frac{0.00567(0.000209)}{-0.00867 - 0.000209} = 0.567143 \qquad |\varepsilon_t| = 2.365 \times 10^{-5}\%$$

The choice of a proper value for δ is not automatic. If δ is too small, the method can be swamped by round-off error caused by subtractive cancellation in the denominator of Eq. (6.8). If it is too big, the technique can become inefficient and even divergent. However, if chosen correctly, it provides a nice alternative for cases where evaluating the derivative is difficult and developing two initial guesses is inconvenient.

6.4 MULTIPLE ROOTS

A *multiple root* corresponds to a point where a function is tangent to the x axis. For example, a double root results from

$$f(x) = (x - 3)(x - 1)(x - 1) \tag{6.9}$$

or, multiplying terms, $f(x) = x^3 - 5x^2 + 7x - 3$. The equation has a *double root* because one value of x makes two terms in Eq. (6.9) equal to zero. Graphically, this corresponds to the curve touching the x axis tangentially at the double root. Examine Fig. 6.10*a* at $x = 1$. Notice that the function touches the axis but does not cross it at the root.

A *triple root* corresponds to the case where one x value makes three terms in an equation equal to zero, as in

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

or, multiplying terms, $f(x) = x^4 - 6x^3 + 12x^2 - 10x + 3$. Notice that the graphical depiction (Fig. 6.10*b*) again indicates that the function is tangent to the axis at the root, but that for this case the axis is crossed. In general, odd multiple roots cross the axis, whereas even ones do not. For example, the quadruple root in Fig. 6.10*c* does not cross the axis.

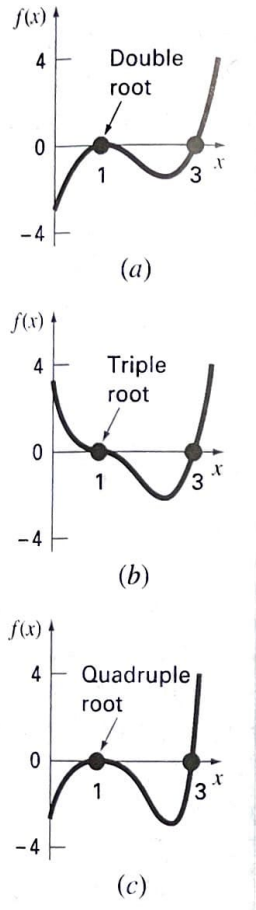


FIGURE 6.10
Examples of multiple roots that are tangential to the x axis. Notice that the function does not cross the axis on either side of even multiple roots (a) and (c), whereas it crosses the axis for odd cases (b).

Multiple roots pose some difficulties for many of the numerical methods described in Part Two:

1. The fact that the function does not change sign at even multiple roots precludes the use of the reliable bracketing methods that were discussed in Chap. 5. Thus, of the methods covered in this book, you are limited to the open methods that may diverge.
2. Another possible problem is related to the fact that not only $f(x)$ but also $f'(x)$ goes to zero at the root. This poses problems for both the Newton-Raphson and secant methods, which both contain the derivative (or its estimate) in the denominator of their respective formulas. This could result in division by zero when the solution converges very close to the root. A simple way to circumvent these problems is based on the fact that it can be demonstrated theoretically (Ralston and Rabinowitz, 1978) that $f(x)$ will always reach zero before $f'(x)$. Therefore, if a zero check for $f(x)$ is incorporated into the computer program, the computation can be terminated before $f'(x)$ reaches zero.
3. It can be demonstrated that the Newton-Raphson and secant methods are linearly, rather than quadratically, convergent for multiple roots (Ralston and Rabinowitz, 1978). Modifications have been proposed to alleviate this problem. Ralston and Rabinowitz (1978) have indicated that a slight change in the formulation returns it to quadratic convergence, as in

$$x_{i+1} = x_i - m\frac{f(x_i)}{f'(x_i)} \tag{6.9a}$$

where m is the multiplicity of the root (that is, $m = 2$ for a double root, $m = 3$ for a triple root, etc.). Of course, this may be an unsatisfactory alternative because it hinges on foreknowledge of the multiplicity of the root.

Another alternative, also suggested by Ralston and Rabinowitz (1978), is to define a new function $u(x)$, that is, the ratio of the function to its derivative, as in

$$u(x) = \frac{f(x)}{f'(x)} \tag{6.10}$$

It can be shown that this function has roots at all the same locations as the original function. Therefore, Eq. (6.10) can be substituted into Eq. (6.6) to develop an alternative form of the Newton-Raphson method:

$$x_{i+1} = x_i - \frac{u(x_i)}{u'(x_i)} \tag{6.11}$$

Equation (6.10) can be differentiated to give

$$u'(x) = \frac{f'(x)f'(x) - f(x)f''(x)}{[f'(x)]^2} \tag{6.12}$$

Equations (6.10) and (6.12) can be substituted into Eq. (6.11) and the result simplified to yield

$$x_{i+1} = x_i - \frac{f(x_i)f'(x_i)}{[f'(x_i)]^2 - f(x_i)f''(x_i)} \tag{6.13}$$

EXAMPLE 6.9 Modified Newton-Raphson Method for Multiple Roots

Problem Statement. Use both the standard and modified Newton-Raphson methods to evaluate the multiple root of Eq. (6.9), with an initial guess of $x_0 = 0$.

Solution. The first derivative of Eq. (6.9) is $f'(x) = 3x^2 - 10x + 7$, and therefore, the standard Newton-Raphson method for this problem is [Eq. (6.6)]

$$x_{i+1} = x_i - \frac{x_i^3 - 5x_i^2 + 7x_i - 3}{3x_i^2 - 10x_i + 7}$$

which can be solved iteratively for

i	x_i	ε_t (%)
0	0	100
1	0.4285714	57
2	0.6857143	31
3	0.8328654	17
4	0.9133290	8.7
5	0.9557833	4.4
6	0.9776551	2.2

As anticipated, the method is linearly convergent toward the true value of 1.0.

For the modified method, the second derivative is $f''(x) = 6x - 10$, and the iterative relationship is [Eq. (6.13)]

$$x_{i+1} = x_i - \frac{(x_i^3 - 5x_i^2 + 7x_i - 3)(3x_i^2 - 10x_i + 7)}{(3x_i^2 - 10x_i + 7)^2 - (x_i^3 - 5x_i^2 + 7x_i - 3)(6x_i - 10)}$$

which can be solved for

i	x_i	ε_t (%)
0	0	100
1	1.105263	11
2	1.003082	0.31
3	1.000002	0.00024

Thus, the modified formula is quadratically convergent. We can also use both methods to search for the single root at $x = 3$. Using an initial guess of $x_0 = 4$ gives the following results:

i	Standard	ε_t (%)	Modified	ε_t (%)
0	4	33	4	33
1	3.4	13	2.636364	12
2	3.1	3.3	2.820225	6.0
3	3.008696	0.29	2.961728	1.3
4	3.000075	0.0025	2.998479	0.051
5	3.000000	2×10^{-7}	2.999998	7.7×10^{-5}

Thus, both methods converge quickly, with the standard method being somewhat more efficient.

The above example illustrates the trade-offs involved in opting for the modified Newton-Raphson method. Although it is preferable for multiple roots, it is somewhat less efficient and requires more computational effort than the standard method for simple roots.

It should be noted that a modified version of the secant method suited for multiple roots can also be developed by substituting Eq. (6.10) into Eq. (6.7). The resulting formula is (Ralston and Rabinowitz, 1978)

$$x_{i+1} = x_i - \frac{u(x_i)(x_{i-1} - x_i)}{u(x_{i-1}) - u(x_i)}$$

6.5 SYSTEMS OF NONLINEAR EQUATIONS

To this point, we have focused on the determination of the roots of a single equation. A related problem is to locate the roots of a set of simultaneous equations,

$$\begin{aligned} f_1(x_1, x_2, \ldots, x_n) &= 0 \\ f_2(x_1, x_2, \ldots, x_n) &= 0 \\ &\cdot \quad\quad \cdot \\ &\cdot \quad\quad \cdot \\ &\cdot \quad\quad \cdot \\ f_n(x_1, x_2, \ldots, x_n) &= 0 \end{aligned} \tag{6.14}$$

The solution of this system consists of a set of x values that simultaneously result in all the equations equaling zero.

In Part Three, we will present methods for the case where the simultaneous equations are linear—that is, they can be expressed in the general form

$$f(x) = a_1x_1 + a_2x_2 + \cdots + a_nx_n - b = 0 \tag{6.15}$$

where the b and the a's are constants. Algebraic and transcendental equations that do not fit this format are called *nonlinear equations*. For example,

$$x^2 + xy = 10$$

and

$$y + 3xy^2 = 57$$

are two simultaneous nonlinear equations with two unknowns, x and y. They can be expressed in the form of Eq. (6.14) as

$$u(x, y) = x^2 + xy - 10 = 0 \tag{6.16a}$$

$$v(x, y) = y + 3xy^2 - 57 = 0 \tag{6.16b}$$

Thus, the solution would be the values of x and y that make the functions $u(x, y)$ and $v(x, y)$ equal to zero. Most approaches for determining such solutions are extensions of the open

methods for solving single equations. In this section, we will investigate two of these: fixed-point iteration and Newton-Raphson.

6.5.1 Fixed-Point Iteration

The fixed-point-iteration approach (Sec. 6.1) can be modified to solve two simultaneous, nonlinear equations. This approach will be illustrated in the following example.

EXAMPLE 6.10 Fixed-Point Iteration for a Nonlinear System

Problem Statement. Use fixed-point iteration to determine the roots of Eq. (6.16). Note that a correct pair of roots is $x = 2$ and $y = 3$. Initiate the computation with guesses of $x = 1.5$ and $y = 3.5$.

Solution. Equation (6.16*a*) can be solved for

$$x_{i+1} = \frac{10 - x_i^2}{y_i} \tag{E6.10.1}$$

and Eq. (6.16*b*) can be solved for

$$y_{i+1} = 57 - 3x_i y_i^2 \tag{E6.10.2}$$

Note that we will drop the subscripts for the remainder of the example.

On the basis of the initial guesses, Eq. (E6.10.1) can be used to determine a new value of x:

$$x = \frac{10 - (1.5)^2}{3.5} = 2.21429$$

This result and the initial value of $y = 3.5$ can be substituted into Eq. (E6.10.2) to determine a new value of y:

$$y = 57 - 3(2.21429)(3.5)^2 = -24.37516$$

Thus, the approach seems to be diverging. This behavior is even more pronounced on the second iteration:

$$x = \frac{10 - (2.21429)^2}{-24.37516} = -0.20910$$

$$y = 57 - 3(-0.20910)(-24.37516)^2 = 429.709$$

Obviously, the approach is deteriorating.

Now we will repeat the computation but with the original equations set up in a different format. For example, an alternative formulation of Eq. (6.16*a*) is

$$x = \sqrt{10 - xy}$$

and of Eq. (6.16*b*) is

$$y = \sqrt{\frac{57 - y}{3x}}$$

Now the results are more satisfactory:

$$x = \sqrt{10 - 1.5(3.5)} = 2.17945$$

$$y = \sqrt{\frac{57 - 3.5}{3(2.17945)}} = 2.86051$$

$$x = \sqrt{10 - 2.17945(2.86051)} = 1.94053$$

$$y = \sqrt{\frac{57 - 2.86051}{3(1.94053)}} = 3.04955$$

Thus, the approach is converging on the true values of $x = 2$ and $y = 3$.

The previous example illustrates the most serious shortcoming of simple fixed-point iteration—that is, convergence often depends on the manner in which the equations are formulated. Additionally, even in those instances where convergence is possible, divergence can occur if the initial guesses are insufficiently close to the true solution. Using reasoning similar to that in Box 6.1, it can be demonstrated that sufficient conditions for convergence for the two-equation case are

$$\left|\frac{\partial u}{\partial x}\right| + \left|\frac{\partial u}{\partial y}\right| < 1$$

and

$$\left|\frac{\partial v}{\partial x}\right| + \left|\frac{\partial v}{\partial y}\right| < 1$$

These criteria are so restrictive that fixed-point iteration has limited utility for solving nonlinear systems. However, as we will describe later in the book, it can be very useful for solving linear systems.

6.5.2 Newton-Raphson

Recall that the Newton-Raphson method was predicated on employing the derivative (that is, the slope) of a function to estimate its intercept with the axis of the independent variable—that is, the root (Fig. 6.5). This estimate was based on a first-order Taylor series expansion (recall Box 6.2),

$$f(x_{i+1}) = f(x_i) + (x_{i+1} - x_i) f'(x_i) \tag{6.17}$$

where x_i is the initial guess at the root and x_{i+1} is the point at which the slope intercepts the x axis. At this intercept, $f(x_{i+1})$ by definition equals zero and Eq. (6.17) can be rearranged to yield

$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)} \tag{6.18}$$

which is the single-equation form of the Newton-Raphson method.

The multiequation form is derived in an identical fashion. However, a multivariable Taylor series must be used to account for the fact that more than one independent variable contributes to the determination of the root. For the two-variable case, a first-order Taylor series can be written [recall Eq. (4.26)] for each nonlinear equation as

$$u_{i+1} = u_i + (x_{i+1} - x_i)\frac{\partial u_i}{\partial x} + (y_{i+1} - y_i)\frac{\partial u_i}{\partial y} \tag{6.19a}$$

and

$$v_{i+1} = v_i + (x_{i+1} - x_i)\frac{\partial v_i}{\partial x} + (y_{i+1} - y_i)\frac{\partial v_i}{\partial y} \tag{6.19b}$$

Just as for the single-equation version, the root estimate corresponds to the values of x and y, where u_{i+1} and v_{i+1} equal zero. For this situation, Eq. (6.19) can be rearranged to give

$$\frac{\partial u_i}{\partial x}x_{i+1} + \frac{\partial u_i}{\partial y}y_{i+1} = -u_i + x_i\frac{\partial u_i}{\partial x} + y_i\frac{\partial u_i}{\partial y} \tag{6.20a}$$

$$\frac{\partial v_i}{\partial x}x_{i+1} + \frac{\partial v_i}{\partial y}y_{i+1} = -v_i + x_i\frac{\partial v_i}{\partial x} + y_i\frac{\partial v_i}{\partial y} \tag{6.20b}$$

Because all values subscripted with i's are known (they correspond to the latest guess or approximation), the only unknowns are x_{i+1} and y_{i+1}. Thus, Eq. (6.20) is a set of two linear equations with two unknowns [compare with Eq. (6.15)]. Consequently, algebraic manipulations (for example, Cramer's rule) can be employed to solve for

$$x_{i+1} = x_i - \frac{u_i\dfrac{\partial v_i}{\partial y} - v_i\dfrac{\partial u_i}{\partial y}}{\dfrac{\partial u_i}{\partial x}\dfrac{\partial v_i}{\partial y} - \dfrac{\partial u_i}{\partial y}\dfrac{\partial v_i}{\partial x}} \tag{6.21a}$$

$$y_{i+1} = y_i - \frac{v_i\dfrac{\partial u_i}{\partial x} - u_i\dfrac{\partial v_i}{\partial x}}{\dfrac{\partial u_i}{\partial x}\dfrac{\partial v_i}{\partial y} - \dfrac{\partial u_i}{\partial y}\dfrac{\partial v_i}{\partial x}} \tag{6.21b}$$

The denominator of each of these equations is formally referred to as the determinant of the *Jacobian* of the system.

Equation (6.21) is the two-equation version of the Newton-Raphson method. As in the following example, it can be employed iteratively to home in on the roots of two simultaneous equations.

EXAMPLE 6.11 Newton-Raphson for a Nonlinear System

Problem Statement. Use the multiple-equation Newton-Raphson method to determine roots of Eq. (6.16). Note that a correct pair of roots is $x = 2$ and $y = 3$. Initiate the computation with guesses of $x = 1.5$ and $y = 3.5$.

Solution. First compute the partial derivatives and evaluate them at the initial guesses of x and y:

$$\frac{\partial u_0}{\partial x} = 2x + y = 2(1.5) + 3.5 = 6.5 \qquad \frac{\partial u_0}{\partial y} = x = 1.5$$

$$\frac{\partial v_0}{\partial x} = 3y^2 = 3(3.5)^2 = 36.75 \qquad \frac{\partial v_0}{\partial y} = 1 + 6xy = 1 + 6(1.5)(3.5) = 32.5$$

Thus, the determinant of the Jacobian for the first iteration is

$$6.5(32.5) - 1.5(36.75) = 156.125$$

The values of the functions can be evaluated at the initial guesses as

$$u_0 = (1.5)^2 + 1.5(3.5) - 10 = -2.5$$

$$v_0 = 3.5 + 3(1.5)(3.5)^2 - 57 = 1.625$$

These values can be substituted into Eq. (6.21) to give

$$x = 1.5 - \frac{-2.5(32.5) - 1.625(1.5)}{156.125} = 2.03603$$

$$y = 3.5 - \frac{1.625(6.5) - (-2.5)(36.75)}{156.125} = 2.84388$$

Thus, the results are converging to the true values of $x = 2$ and $y = 3$. The computation can be repeated until an acceptable accuracy is obtained.

Just as with fixed-point iteration, the Newton-Raphson approach will often diverge if the initial guesses are not sufficiently close to the true roots. Whereas graphical methods could be employed to derive good guesses for the single-equation case, no such simple procedure is available for the multiequation version. Although there are some advanced approaches for obtaining acceptable first estimates, often the initial guesses must be obtained on the basis of trial and error and knowledge of the physical system being modeled.

The two-equation Newton-Raphson approach can be generalized to solve n simultaneous equations. Because the most efficient way to do this involves matrix algebra and the solution of simultaneous linear equations, we will defer discussion of the general approach to Part Three.

PROBLEMS

6.1 Use simple fixed-point iteration to locate the root of

$$f(x) = 2\sin(\sqrt{x}) - x$$

Use an initial guess of $x_0 = 0.5$ and iterate until $\varepsilon_a \le 0.001\%$. Verify that the process is linearly convergent as described in Box 6.1.

6.2 Determine the highest real root of

$$f(x) = 2x^3 - 11.7x^2 + 17.7x - 5$$

(a) Graphically.

(b) Fixed-point iteration method (three iterations, $x_0 = 3$). Note: Make certain that you develop a solution that converges on the root.

(c) Newton-Raphson method (three iterations, $x_0 = 3$).

(d) Secant method (three iterations, $x_{-1} = 3, x_0 = 4$).

(e) Modified secant method (three iterations, $x_0 = 3, \delta = 0.01$).

Compute the approximate percent relative errors for your solutions.