3

Systems of Nonlinear Equations

Many systems can be modeled generically as

$$F(x) = 0 \tag{3.1}$$

where x is an n-vector and F represents a nonlinear mapping with both its domain and range in the n-dimensional real linear space R^n. The mapping F can also be interpreted as being an n-vector of functions

$$F(x) = \begin{bmatrix} f_1(x_1, x_2, \ldots, x_n) \\ f_2(x_1, x_2, \ldots, x_n) \\ \vdots \\ f_n(x_1, x_2, \ldots, x_n) \end{bmatrix} = 0 \tag{3.2}$$

where at least one of the functions is nonlinear. Each function may or may not involve all n states x_i, but it is assumed that every state appears at least once in the set of functions. The solution x^* of the nonlinear system cannot, in general, be expressed in closed form. Thus nonlinear systems are usually solved numerically. In many cases, it is possible to find an approximate solution $\hat{x}$ arbitrarily close to the actual solution x^*, by replacing each approximation with successively better (more accurate) approximations until

$$F(\hat{x}) \approx 0$$

Such methods are usually *iterative*. An iterative solution is one in which an initial guess (x^0) to the solution is used to create a sequence x^0, x^1, x^2, ... that (hopefully) converges arbitrarily close to the desired solution x^*.

Three principal issues arise with the use of iterative methods, namely,

1. Is the iterative process well defined? That is, can it be successively applied without numerical difficulties?

2. Do the iterates (i.e., the sequence of updates) converge to a solution of Equation (3.1)? Is the solution the desired solution?

3. How economical is the entire solution process?

The complete (or partial) answers to these issues are enough to fill several volumes, and as such cannot be discussed in complete detail in this chapter.

These issues, however, are central to the solution of nonlinear systems and cannot be fully ignored. Therefore, this chapter will endeavor to provide sufficient detail for the reader to be aware of the advantages (and disadvantages) of different types of iterative methods without providing exhaustive coverage.

3.1 Fixed-Point Iteration

Solving a system of nonlinear equations is a complex problem. To better understand the mechanisms involved in a large-scale system, it is instructive to first consider the one-dimensional, or scalar, nonlinear system

$$f(x) = 0 \tag{3.3}$$

One approach to solving any nonlinear equation is the tried-and-true "trial and error" method that most engineering and science students have used at one time or another in their careers.

Example 3.1
Find the solution to

$$f(x) = x^2 - 5x + 4 = 0 \tag{3.4}$$

Solution 3.1 This is a quadratic equation that has a closed form solution. The two solutions are

$$x_1^*, x_2^* = \frac{5 \pm \sqrt{(-5)^2 - (4)(4)}}{2} = 1, \;\; 4$$

If a closed form solution did not exist, however, one approach would be to use a trial and error approach. Since the solution occurs when $f(x) = 0$, the value of $f(x)$ can be monitored and used to refine the estimates to x^*.

k	x	$f(x)$
0	0	$0 - 0 + 4 = 4 > 0$
1	2	$4 - 10 + 4 = -2 < 0$
2	0.5	$0.25 - 2.5 + 4 = 1.75 > 0$
3	1.5	$2.25 - 7.5 + 4 = -1.25 < 0$

By noting the sign of the function and whether or not it changes sign, the interval in which the solution lies can be successively narrowed. If a function $f(x)$ is continuous and $f(a) \cdot f(b) < 0$, then the equation $f(x) = 0$ has at least one solution in the interval (a, b). Since $f(0.5) > 0$ and $f(1.5) < 0$, it can be concluded that one of the solutions lies in the interval $(0.5, 1.5)$. ■

This process, however, tends to be tedious, and there is no guidance to determine what the next guess should be other than the bounds established by the change in sign of $f(x)$. A better method would be to write the sequence of updates in terms of the previous guesses. Thus an iterative function can be defined as

$$I: \quad x^{k+1} = g\left(x^k\right), \quad k = 1, \ldots, \infty \tag{3.5}$$

This is known as a fixed-point iteration because at the solution

$$x^* = g\left(x^*\right) \tag{3.6}$$

Example 3.2
Find the solution to Equation (3.4) using a fixed-point iteration.

Solution 3.2 Equation (3.4) can be rewritten as

$$x = \frac{x^2 + 4}{5} \tag{3.7}$$

Adopting the notation of Equation (3.5), the iterative function becomes

$$x^{k+1} = g\left(x^k\right) = \frac{\left(x^k\right)^2 + 4}{5} \tag{3.8}$$

Using this iterative function, the estimates to x^* are

k	x^k	x^{k+1}
0	0	$\frac{0+4}{5} = 0.8$
1	0.8	$\frac{0.64+4}{5} = 0.928$
2	0.928	$\frac{0.856+4}{5} = 0.972$
3	0.971	$\frac{0.943+4}{5} = 0.989$

It is obvious that this sequence is converging to the solution $x^* = 1$.

Now consider the same example, except with a different initial guess:

k	x^k	x^{k+1}
0	5	$\frac{25+4}{5} = 5.8$
1	5.8	$\frac{33.64+4}{5} = 7.528$
2	7.528	$\frac{56.67+4}{5} = 12.134$

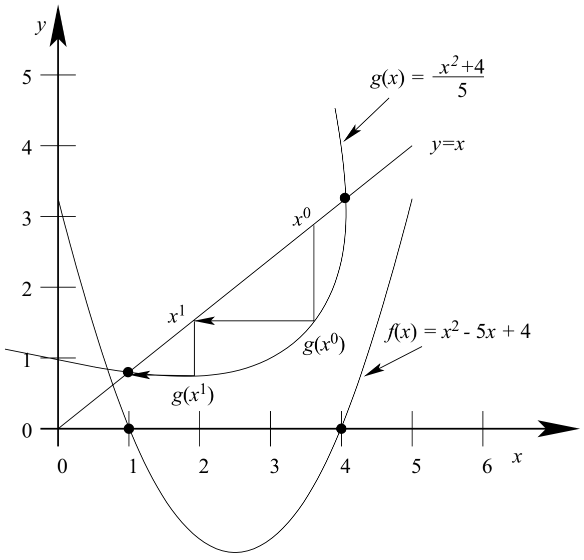


FIGURE 3.1
Graphical interpretation of the fixed-point iteration

In this case, the iterates are increasing rapidly and after a few more iterations would approach infinity. In this case, it is said that the iteration is *diverging.* ■

This example brings up two very important points: will a sequence of iterates converge, and, if so, to what solution will they converge? In order to address these questions, consider first a graphical interpretation of Example 3.2. Plotting both sides of the function in Equation (3.7) yields the two lines shown in Figure 3.1.

These two lines intersect at the same two points in which the original function $f(x) = 0$. The fixed-point iteration works by finding this intersection. Consider the initial guess x^0 shown in Figure 3.1. The function $g(x)$ evaluated at x^0 gives the updated iterate x^1. Thus a vertical line projected from x^0 points to $g(x^0)$ and a horizontal line projected from $g(x^0)$ gives x^1.

The projection of the function $g(x^1)$ yields x^2. Similar vertical and horizontal projections will eventually lead directly to the point at which the two lines intersect. In this way, the solution to the original function $f(x)$ can be obtained.

In this example, the solution $x^* = 1$ is the *point of attraction* of the fixed-point iteration. A point x^* is said to be a point of attraction of an iterative function I if there exists an open neighborhood S_0 of x^* such that, for all initial guesses x^0 in the subset S_0 of S, the iterates will remain in S and

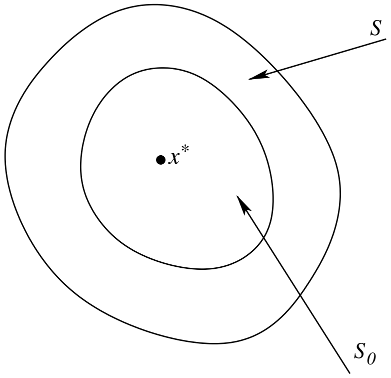


FIGURE 3.2
Domain of attraction of x^*

$$\lim_{k\to\infty} x^k = x^* \tag{3.9}$$

The neighborhood S_0 is called the *domain of attraction* of x^* [38]. This concept is illustrated in Figure 3.2 and implies that the iterates of I will converge to x^* whenever x^0 is sufficiently close to x^*. In Example 3.2, the fixed point $x^* = 1$ is a point of attraction of

$$I: \quad x^{k+1} = \frac{\left(x^k\right)^2 + 4}{5}$$

whereas $x^* = 4$ is not. The domain of attraction of $x^* = 1$ is all x in the domain $-\infty < x < 4$.

It is often difficult to determine a priori whether or not an iteration will converge. In some cases, a series of iterates will appear to be converging, but will not approach x^* even as $k \to \infty$. However, there are a number of theorems that provide insight as to whether an iteration of $x = g(x)$ will converge.

Mean Value Theorem: [52] *Suppose a function $g(x)$ and its derivative $g'(x)$ are both continuous in the interval $a \leq x \leq b$. Then there exists at least one ξ, $a < \xi < b$ such that*

$$g'(\xi) = \frac{g(b) - g(a)}{b - a} \tag{3.10}$$

The meaning of this theorem is shown in Figure 3.3. If a function $g(x)$ is defined in the region between $x = a$ and $x = b$ and is both differentiable (smooth) and continuous, then a secant line can be drawn between points A

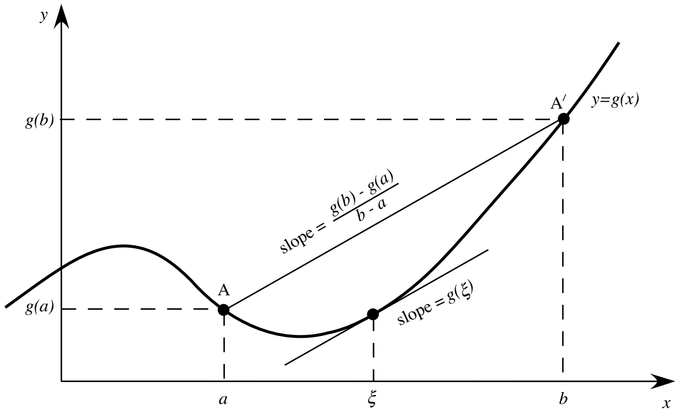


FIGURE 3.3
Meaning of the mean value theorem

and A'. The slope of this secant line is

$$\frac{g(b) - g(a)}{b - a}$$

The mean value theorem states that there is at least one point on the curve, at $x = \xi$, where the tangent to the curve has the same slope as the line AA'.

Rewriting the equation of the mean value theorem as

$$g(b) - g(a) = g'(\xi)(b - a)$$

then for successive iterates in which $x^{k+1} = b$ and $x^k = a$, then

$$g(x^{k+1}) - g(x^k) = g'(\xi^k)(x^{k+1} - x^k) \tag{3.11}$$

or taking the absolute values

$$\left|g(x^{k+1}) - g(x^k)\right| = \left|g'(\xi^k)\right| \left|(x^{k+1} - x^k)\right| \tag{3.12}$$

As long as the appropriate ξ^k is used, the mean value theorem can be successively applied to each iteration of the sequence. If the entire region includes x^* as well as all of the x^k, then the derivative $g'(x)$ is bounded. Therefore

$$\left|g'(\xi^k)\right| \leq M \tag{3.13}$$

for any k where M is the positive upper bound. Then, starting from the initial guess x^0,

$$\left|x^2 - x^1\right| \leq M \left|x^1 - x^0\right| \tag{3.14}$$

$$\left|x^3 - x^2\right| \leq M \left|x^2 - x^1\right| \tag{3.15}$$

$$\vdots \tag{3.16}$$

$$\left|x^{k+1} - x^k\right| \leq M \left|x^k - x^{k-1}\right| \tag{3.17}$$

and by combining yields

$$\left|x^{k+1} - x^k\right| \le M^k \left|x^1 - x^0\right| \tag{3.18}$$

Thus, for any initial guess x^0,

$$|g'(x)| \le M < 1 \tag{3.19}$$

the iterates will converge.

A similar, but slightly different method of determining whether or not an iterative process will converge is given by the **Ostrowski** theorem [38]. This theorem states that, if the iterative process

$$I: \quad x^{k+1} = g\left(x^k\right), \quad k = 1, \ldots, \infty$$

has a fixed-point x^* and is continuous and differentiable at x^*, and if $\left|\frac{\partial g(x^*)}{\partial x}\right| < 1$, then x^* is a point of attraction of I.

Example 3.3

Determine whether $x^* = 1$ and $x^* = 4$ are points of attraction of the iterative function of Equation (3.8).

Solution 3.3 The derivative of the iterative process I in Equation (3.8) is

$$\left|\frac{\partial g(x)}{\partial x}\right| = \left|\frac{2}{5}x\right|$$

Thus, for $x^* = 1$, $\left|\frac{2}{5}x^*\right| = \frac{2}{5} < 1$ and $x^* = 1$ is a point of attraction of I. For $x^* = 4$, $\left|\frac{2}{5}x^*\right| = \frac{2}{5}(4) = \frac{8}{5} > 1$; thus $x^* = 4$ is not a point of attraction of I. ■

There are four possible convergence types for fixed-point iterations. These are shown graphically in Figure 3.4. Figure 3.4(a) shows what happens if $g'(x)$ is between 0 and 1. Even if the initial guess x_0 is far from x^*, the successive values of x^k approach the solution from one side – this is defined as *monotonic convergence*. Figure 3.4(b) shows the situation when $g'(x)$ is between -1 and 0. Even if the initial guess x_0 is far from x^*, the successive values of x^k approach the solution from first one side and then the other oscillating around the root. This convergence is *oscillatory convergence*.

Figure 3.4(c) shows the case when $g'(x)$ is greater than 1, leading to *monotonic divergence*. Figure 3.4(d) illustrates the case when $g'(x) < -1$ and $|g'(x)| > 1$. This is *oscillatory divergence*.

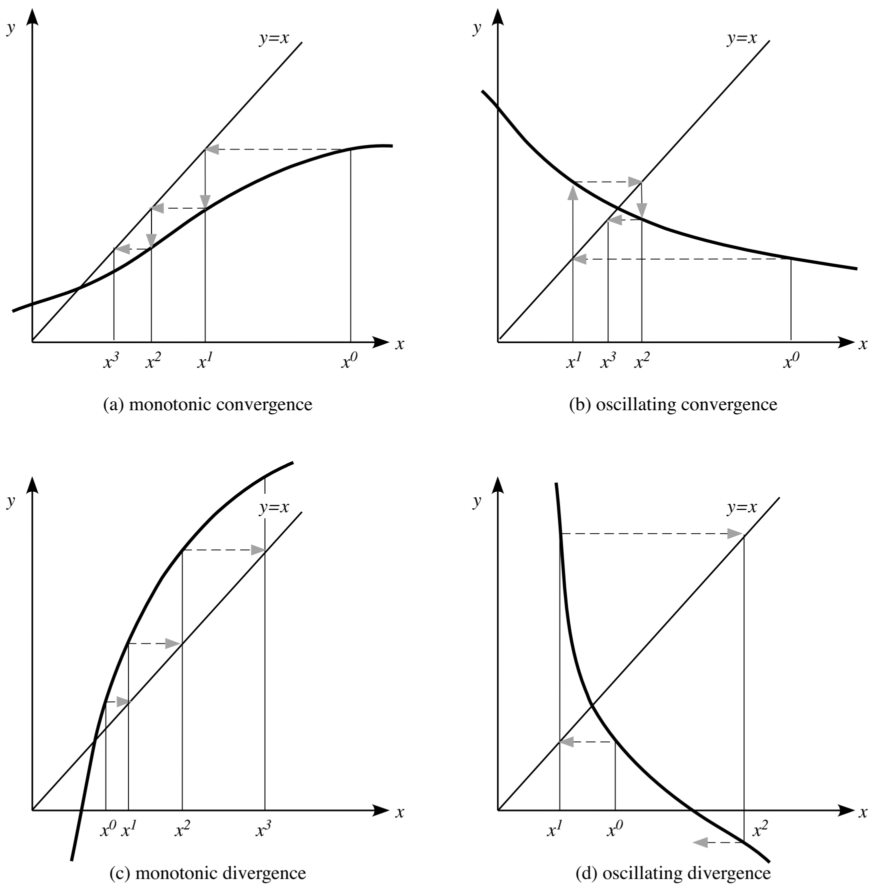


FIGURE 3.4
Four possible convergence types in the iteration $x = g(x)$

3.2 Newton–Raphson Iteration

Several iterative methods offer more robust convergence behavior than the simple fixed-point iteration described in the previous section. One of the most widely used iterative methods is the *Newton–Raphson* iterative method. This method can also be described by the iterative process

$$I: \quad x^{k+1} = g\left(x^k\right), \quad k = 1, \ldots, \infty$$

but frequently offers better convergence properties than the fixed-point iteration.

Consider again the scalar nonlinear function

$$f(x^*) = 0 \tag{3.20}$$

Expanding this function in a Taylor series expansion about the point x^k yields

$$f(x^*) = f\left(x^k\right) + \left.\frac{\partial f}{\partial x}\right|_{x^k} \left(x^* - x^k\right) + \frac{1}{2!}\left.\frac{\partial^2 f}{\partial x^2}\right|_{x^k} \left(x^* - x^k\right)^2 + \ldots = 0 \tag{3.21}$$

If it is assumed that the iterates will converge to x^* as $k \to \infty$, then the updated guess x^{k+1} can be substituted for x^*, yielding

$$f(x^{k+1}) = f\left(x^k\right) + \left.\frac{\partial f}{\partial x}\right|_{x^k} \left(x^{k+1} - x^k\right) + \frac{1}{2!}\left.\frac{\partial^2 f}{\partial x^2}\right|_{x^k} \left(x^{k+1} - x^k\right)^2 + \ldots = 0 \tag{3.22}$$

If the initial guess is "sufficiently close" to x^* and within the domain of attraction of x^*, then the higher-order terms of the expansion can be neglected, yielding

$$f(x^{k+1}) = f\left(x^k\right) + \left.\frac{\partial f}{\partial x}\right|_{x^k} \left(x^{k+1} - x^k\right) \approx 0 \tag{3.23}$$

Solving directly for x^{k+1} as a function of x^k yields the following iterative function:

$$I: \quad x^{k+1} = x^k - \left[\left.\frac{\partial f}{\partial x}\right|_{x^k}\right]^{-1} f\left(x^k\right) \tag{3.24}$$

which is the well-known Newton–Raphson iterative method.

The Newton–Raphson method also lends itself to a graphical interpretation. Consider the same function as in Example 3.2 plotted in Figure 3.5. In this method, the slope of the function evaluated at the current iteration is used to produce the next guess. For any guess x^k, there corresponds a point on the function $f\left(x^k\right)$ with slope

$$\left.\frac{\partial f}{\partial x}\right|_{x=x^k}$$

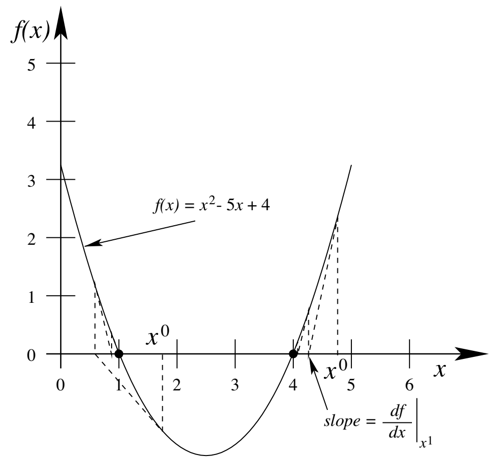


FIGURE 3.5
Graphical interpretation of the Newton–Raphson method

Therefore, the next guess x^{k+1} is simply the intersection of the slope and the x-axis. This process is repeated until the guesses are sufficiently close to the solution x^*. An iteration is said to have converged at x^k if

$$\left|f\left(x^k\right)\right| < \varepsilon$$

where ε is some predetermined tolerance.

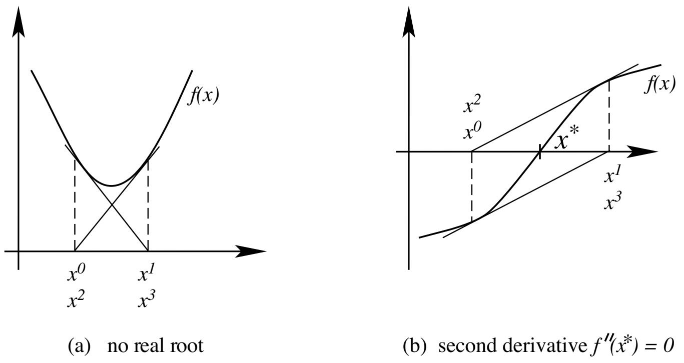
Example 3.4
Repeat Example 3.2 using a Newton–Raphson iteration.

Solution 3.4 Using the Newton–Raphson method of Equation (3.24), the iterative function is given by

$$I: \quad x^{k+1} = x^k - \frac{\left(x^k\right)^2 - 5x^k + 4}{2x^k - 5} \tag{3.25}$$

Using this iterative function, the estimates to x^* from an initial guess of $x^0 = 3$ are

k	x^k	x^{k+1}
0	3	$3 - \frac{9-15+4}{6-5} = 5$
1	5	$5 - \frac{25-25+4}{10-5} = 4.2$
2	4.2	$4.2 - \frac{17.64-21+4}{8.4-5} = 4.012$



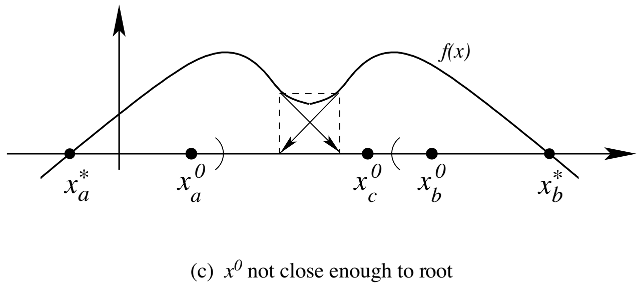


FIGURE 3.6
Newton–Raphson regions of convergence

Similarly, the estimates to x^* from an initial guess of $x^0 = 2$ are

k	x^k	x^{k+1}
0	2	$2 - \frac{4-10+4}{4-5} = 0$
1	0	$0 - \frac{0-0+4}{0-5} = 0.8$
2	0.8	$0.8 - \frac{0.64-4+4}{1.6-5} = 0.988$

In this case, both solutions are points of attraction of the Newton–Raphson iteration. ■

In some cases, however, the Newton–Raphson method will also fail to converge. Consider the functions shown in Figure 3.6. In Figure 3.6(a), the function has no real root. In Figure 3.6(b), the function is symmetric around x^* and the second derivative is zero. In Figure 3.6(c), an initial guess of x_a^0 will converge to the solution x_a^*. An initial guess of x_b^0 will converge to the solution x_b^*. However, an initial guess of x_c^0 will cause the iterates to get locked in and oscillate in the region denoted by the dashed box without ever converging to a solution. This figure supports the assertion that, if the initial guess is too far away from the actual solution, the iterates may not converge. Or conversely, the initial guess must be sufficiently close to the actual solution for the Newton–Raphson iteration to converge. This supports the initial assumption used to derive the Newton–Raphson algorithm in that, *if the iterates were sufficiently close to the actual solution*, the higher-order terms of the Taylor series expansion could be neglected. If the iterates are not sufficiently close to the actual solution, these higher-order terms are significant and the assumption upon which the Newton–Raphson algorithm is based is not valid.

3.2.1 Convergence Properties

Note that the rate of convergence to the solution in Example 3.4 is much faster than in Example 3.2. This is because the Newton–Raphson method exhibits *quadratic convergence*, whereas the fixed-point iteration exhibits only *linear convergence.* Linear convergence implies that, once the iterates x^k are sufficiently close to the actual solution x^*, then the error

$$\varepsilon^k = \left|x^k - x^*\right| \tag{3.26}$$

will approach zero in a linear fashion. The convergence of Examples 3.2 and 3.4 is shown in Figure 3.7. Plotted on a log-scale plot, the error for the fixed-point iteration is clearly linear, whereas the Newton–Raphson error exhibits quadratic convergence until it becomes too small to plot. Numerous methods have been proposed to predict the rate of convergence of iterative methods. Let the error of an iterative function be defined as in Equation (3.26). If there exists a number p and a constant $C \neq 0$ such that

$$\lim_{k\to\infty} \frac{\left|\varepsilon^{k+1}\right|}{\left|\varepsilon^k\right|^p} = C \tag{3.27}$$

then p is called the *order of convergence* of the iterative sequence and C is the *asympototic error constant.* If $p = 1$, the convergence is said to be *linear.* If $p = 2$, the convergence is *quadratic,* and if $p = 3$, the order of convergence is *cubic.* The Newton–Raphson method satisfies Equation (3.27) with $p = 2$ if

$$C = \frac{1}{2}\frac{\left|\frac{d^2f(x^*)}{dx^2}\right|}{\left|\frac{df(x^*)}{dx}\right|}$$

where $C \neq 0$ only if $\frac{d^2f(x^*)}{dx^2} \neq 0$. Thus, for most functions, the Newton–Raphson method exhibits quadratic convergence.

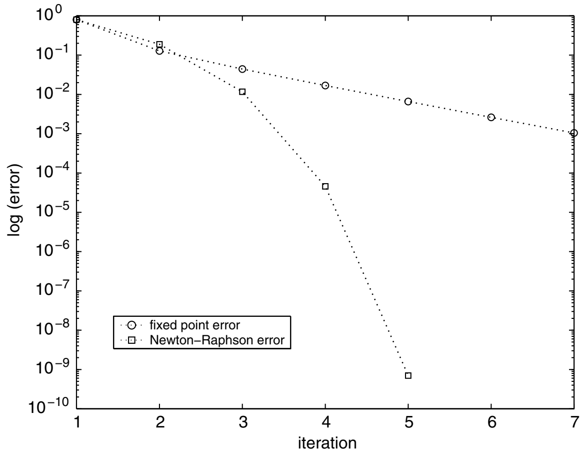


FIGURE 3.7
Nonconverging iteration (fixed-point vs. Newton–Raphson)

3.2.2 The Newton–Raphson for Systems of Nonlinear Equations

In science and engineering, many applications give rise to *systems* of equations such as those in Equation (3.2). With a few modifications, the Newton–Raphson method developed in the previous section can be extended to systems of nonlinear equations. Systems of equations can similarly be represented by Taylor series expansions. By making the assumption once again that the initial guess is sufficiently close to the exact solution, the multidimensional higher-order terms can be neglected, yielding the Newton–Raphson method for n-dimensional systems:

$$x^{k+1} = x^k - \left[J\left(x^k\right)\right]^{-1} F\left(x^k\right) \tag{3.28}$$

where

$$x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{bmatrix}$$

$$F\left(x^k\right) = \begin{bmatrix} f_1\left(x^k\right) \\ f_2\left(x^k\right) \\ f_3\left(x^k\right) \\ \vdots \\ f_n\left(x^k\right) \end{bmatrix}$$

and the Jacobian matrix $\left[J\left(x^k\right)\right]$ is given by

$$\left[J\left(x^k\right)\right] = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \frac{\partial f_1}{\partial x_3} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \frac{\partial f_2}{\partial x_3} & \cdots & \frac{\partial f_2}{\partial x_n} \\ \frac{\partial f_3}{\partial x_1} & \frac{\partial f_3}{\partial x_2} & \frac{\partial f_3}{\partial x_3} & \cdots & \frac{\partial f_3}{\partial x_n} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \frac{\partial f_n}{\partial x_1} & \frac{\partial f_n}{\partial x_2} & \frac{\partial f_n}{\partial x_3} & \cdots & \frac{\partial f_n}{\partial x_n} \end{bmatrix}$$

Typically, the inverse of the Jacobian $\left[J\left(x^k\right)\right]$ is not found directly, but rather through LU factorization by posing the Newton–Raphson method as

$$\left[J\left(x^k\right)\right]\left(x^{k+1} - x^k\right) = -F\left(x^k\right) \tag{3.29}$$

which is now in the form $Ax = b$ where the Jacobian is the matrix A, the function $-F\left(x^k\right)$ is the vector b, and the unknown x is the difference vector $\left(x^{k+1} - x^k\right)$. Convergence is typically evaluated by considering the norm of the function

$$\left\|F\left(x^k\right)\right\| < \varepsilon \tag{3.30}$$

Note that the Jacobian is a function of x^k and is therefore updated every iteration along with $F\left(x^k\right)$.

Example 3.5

Find the solution to

$$0 = x_1^2 + x_2^2 - 5x_1 + 1 \ = f_1\left(x_1, x_2\right) \tag{3.31}$$

$$0 = x_1^2 - x_2^2 - 3x_2 - 3 \ = f_2\left(x_1, x_2\right) \tag{3.32}$$

with an initial guess of

$$x^{(0)} = \begin{bmatrix} 3 \\ 3 \end{bmatrix}$$

Solution 3.5 The Jacobian of this system of equations is

$$J\left(x_1, x_2\right) = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} \end{bmatrix} = \begin{bmatrix} 2x_1 - 5 & 2x_2 \\ 2x_1 & -2x_2 - 3 \end{bmatrix}$$

Iteration 1

The Jacobian and the functions f_1 and f_2 are evaluated at the initial condition

$$\begin{bmatrix} 1 & 6 \\ 6 & -9 \end{bmatrix} \begin{bmatrix} x_1^{(1)} - 3 \\ x_2^{(1)} - 3 \end{bmatrix} = \begin{bmatrix} -4 \\ 12 \end{bmatrix} \tag{3.33}$$

Solving this linear system yields

$$\begin{bmatrix} x_1^{(1)} - 3 \\ x_2^{(1)} - 3 \end{bmatrix} = \begin{bmatrix} 0.8 \\ -0.8 \end{bmatrix} \tag{3.34}$$

Thus

$$x_1^{(1)} = 0.8 + x_1^{(0)} = 0.8 + 3 = 3.8 \tag{3.35}$$

$$x_2^{(1)} = -0.8 + x_2^{(0)} = -0.8 + 3 = 2.2 \tag{3.36}$$

The error at iteration 1 is

$$\left\| \begin{bmatrix} f_1\left(x_1^{(0)}, x_2^{(0)}\right) \\ f_2\left(x_1^{(0)}, x_2^{(0)}\right) \end{bmatrix} \right\|_\infty = 12$$

Iteration 2

The Jacobian and the functions f_1 and f_2 are evaluated at $x^{(1)}$

$$\begin{bmatrix} 2.6 & 4.4 \\ 7.6 & -7.4 \end{bmatrix} \begin{bmatrix} x_1^{(2)} - 3.8 \\ x_2^{(2)} - 2.2 \end{bmatrix} = \begin{bmatrix} -1.28 \\ 0.00 \end{bmatrix} \tag{3.37}$$

Solving this linear system yields

$$\begin{bmatrix} x_1^{(2)} - 3.8 \\ x_2^{(2)} - 2.2 \end{bmatrix} = \begin{bmatrix} -0.1798 \\ -0.1847 \end{bmatrix} \tag{3.38}$$

Thus

$$x_1^{(2)} = -0.1798 + x_1^{(1)} = -0.1798 + 3.8 = 3.6202 \tag{3.39}$$

$$x_2^{(2)} = -0.1847 + x_2^{(1)} = -0.1847 + 2.2 = 2.0153 \tag{3.40}$$

The error at iteration 2 is

$$\left\| \begin{bmatrix} f_1\left(x_1^{(1)}, x_2^{(1)}\right) \\ f_2\left(x_1^{(1)}, x_2^{(1)}\right) \end{bmatrix} \right\|_\infty = 1.28$$

Iteration 3

The Jacobian and the functions f_1 and f_2 are evaluated at $x^{(2)}$

$$\begin{bmatrix} 2.2404 & 4.0307 \\ 7.2404 & -7.0307 \end{bmatrix} \begin{bmatrix} x_1^{(3)} - 3.6202 \\ x_2^{(3)} - 2.0153 \end{bmatrix} = \begin{bmatrix} -0.0664 \\ 0.0018 \end{bmatrix} \tag{3.41}$$

Solving this linear system yields

$$\begin{bmatrix} x_1^{(3)} - 3.6202 \\ x_2^{(3)} - 2.0153 \end{bmatrix} = \begin{bmatrix} -0.0102 \\ -0.0108 \end{bmatrix} \tag{3.42}$$

Thus

$$x_1^{(3)} = -0.0102 + x_1^{(2)} = -0.0102 + 3.6202 = 3.6100 \tag{3.43}$$

$$x_2^{(3)} = -0.0108 + x_2^{(2)} = -0.0108 + 2.0153 = 2.0045 \tag{3.44}$$

The error at iteration 3 is

$$\left\| \begin{bmatrix} f_1\left(x_1^{(2)}, x_2^{(2)}\right) \\ f_2\left(x_1^{(2)}, x_2^{(2)}\right) \end{bmatrix} \right\|_\infty = 0.0664$$

At iteration 4, the functions f_1 and f_2 are evaluated at $x^{(3)}$ and yield the following:

$$\begin{bmatrix} f_1\left(x_1^{(3)}, x_2^{(3)}\right) \\ f_2\left(x_1^{(3)}, x_2^{(3)}\right) \end{bmatrix} = \begin{bmatrix} -0.221 \times 10^{-3} \\ 0.012 \times 10^{-3} \end{bmatrix}$$

Since the norm of this matrix is very small, it can be concluded that the iterates have converged and

$$\begin{bmatrix} x_1^{(3)} \\ x_2^{(3)} \end{bmatrix} = \begin{bmatrix} 3.6100 \\ 2.0045 \end{bmatrix}$$

are within an order of error of 10^{-3} of the actual solution. ■

In the solution to Example 3.5, the error at each iteration is

iteration	error
0	12.0000
1	1.2800
2	0.0664
3	0.0002

Note that, once the solution is sufficiently close to the actual solution, the error at each iteration decreases rapidly. If the iterations were carried far enough, the error at each iteration would become roughly the square of the previous iteration error. This convergence behavior is indicative of the quadratic convergence of the Newton–Raphson method.

3.3 Quasi-Newton Methods

Although the full Newton–Raphson method exhibits quadratic convergence and a minimum number of iterations, each iteration may require significant computation. For example, the computation of a full Jacobian matrix requires

n^2 calculations, and each iteration requires on the order of n^3 operations for the LU factorization if the Jacobian is a full matrix. Therefore, most modifications to the Newton–Raphson method propose to reduce either the calculation or the LU factorization of the Jacobian matrix.

Consider once again the iterative statement of the Newton–Raphson method:

$$I: \quad x^{k+1} = x^k - \left[J\left(x^k\right)\right]^{-1} f\left(x^k\right)$$

This iterative statement can be written in a more general form as

$$I: \quad x^{k+1} = x^k - \left[M\left(x^k\right)\right]^{-1} f\left(x^k\right) \tag{3.45}$$

where M is an $n \times n$ matrix that may or may not be a function of x^k. Note that, even if $M \neq J$, this iteration will still converge to a correct solution for x if the function $f(x)$ is driven to zero. So one approach to simplifying the Newton–Raphson method is to find a suitable substitute matrix M that is easier to compute than the system Jacobian. Methods that use a substitute matrix M in place of J are known as *quasi-Newton* methods.

3.3.1 Secant Method

The Newton-Rapshon method is based on using the line tangent to the function $y = f(x)$. By using the slope of the tangent line, the update to the iteration can be calculated. The difficulty with this method is the computational complexity required to compute the function derivative, or $f'(x)$. An alternate approach to calculating the slope of the tangent is to take two points close to the desired root and interpolate between them to estimate the slope, as shown in Figure 3.8. This produces the linear function

$$q(x) = a_0 + a_1 x \tag{3.46}$$

where $q\left(x^0\right) = f\left(x^0\right)$ and $q\left(x^1\right) = f\left(x^1\right)$. This line is the *secant* line and is given by

$$q(x) = \frac{\left(x^1 - x\right) f\left(x^0\right) + \left(x - x^0\right) f\left(x^1\right)}{x^1 - x^0} \tag{3.47}$$

Setting $x_2 = x$ and solving yields

$$x^2 = x^1 - f\left(x^1\right)\left[\frac{f\left(x^1\right) - f\left(x^0\right)}{x^1 - x^0}\right]^{-1} \tag{3.48}$$

The process can now be repeated by using x^2 and x^1 to produce another secant line. By repeatedly updating the secant line, the generalized formula becomes

$$x^{k+1} = x^k - f\left(x^k\right)\left[\frac{f\left(x^k\right) - f\left(x^{k-1}\right)}{x^k - x^{k-1}}\right]^{-1} \tag{3.49}$$

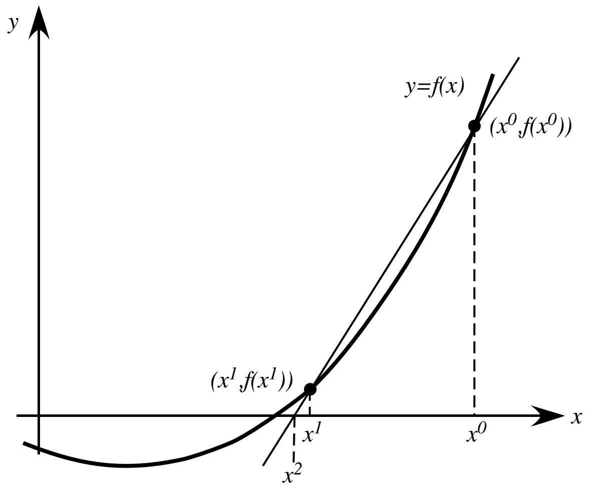


FIGURE 3.8
Illustration of secant method

Note that the secant method can be considered an approximation of the Newton–Raphson method

$$x^{k+1} = x^k - \frac{f\left(x^k\right)}{f'\left(x^k\right)} \tag{3.50}$$

by using the approximation

$$f'\left(x^k\right) = \frac{f\left(x^k\right) - f\left(x^{k-1}\right)}{x^k - x^{k-1}} \tag{3.51}$$

The secant method is often faster than the Newton–Raphson method even though it requires a greater number of iterations to converge to the same level of accuracy. This is because the Newton–Raphson method requires two function evaluations $\left(f\left(x^k\right) \text{ and } f'\left(x^k\right)\right)$, whereas the secant method only requires one function evaluation $\left(f\left(x^k\right)\right)$ since $f\left(x^{k-1}\right)$ can be saved from the previous iteration.

The secant method exhibits *super linear* convergence; its convergence is faster than linear convergence, but not as fast as quadratic convergence (of the Newton–Raphson method). Let the error at iteration k be given by

$$e^k = x^k - x^\star \tag{3.52}$$

where $x^\star$ is the exact solution. Using the Taylor series expansion

$$f\left(x^k\right) = f\left(x^\star + \left(x^k - x^\star\right)\right) \tag{3.53}$$

$$= f\left(x^\star + e^k\right) \tag{3.54}$$

$$= f\left(x^\star\right) + f'\left(x^\star\right) e^k + \frac{1}{2} f''\left(x^\star\right)\left(e^k\right)^2 + \dots \tag{3.55}$$

Similarly,

$$f\left(x^{k-1}\right) = f\left(x^\star\right) + f'\left(x^\star\right) e^{k-1} + \frac{1}{2} f''\left(x^\star\right)\left(e^{k-1}\right)^2 + \dots \tag{3.56}$$

Furthermore,

$$x^k - x^{k-1} = \left(x^k - x^\star\right) - \left(x^{k-1} - x^\star\right) = e^k - e^{k-1}$$

Subtracting $x^\star$ from both sides of Equation (3.49) and recalling that $f\left(x^\star\right) = 0$ yields

$$e^{k+1} = e^k - \frac{f'\left(x^\star\right) e^k + \frac{1}{2} f''\left(x^\star\right)\left(e^k\right)^2}{f'\left(x^\star\right) + \frac{1}{2} f''\left(x^\star\right)\left(e^k + e^{k-1}\right)} \tag{3.57}$$

or

$$e^{k+1} = \frac{1}{2}\frac{f''\left(x^\star\right)}{f'\left(x^\star\right)} e^k e^{k-1} + O\left(e^3\right) \tag{3.58}$$

Let

$$e^k = C_k\left(e^k\right)^r$$

where r is the convergence order. If $r > 1$, then the convergence rate is super linear. If the remainder term in Equation (3.58) is negligible, then Equation (3.58) can be rewritten

$$\frac{e^{k+1}}{e^k e^{k-1}} = \frac{1}{2}\frac{f''\left(x^\star\right)}{f'\left(x^\star\right)} \tag{3.59}$$

and in the limit

$$\begin{array}{c}\lim \\ k \to \infty\end{array} \frac{e^{k+1}}{e^k e^{k-1}} = C' \tag{3.60}$$

For large k,

$$e^k = C\left(e^{k-1}\right)^r$$

and

$$e^{k+1} = C\left(e^k\right)^r = C\left(C\left(e^{k-1}\right)^r\right)^r = C^{r+1}\left(e^{k-1}\right)^{r^2}$$

Substituting this into Equation (3.60) yields

$$\begin{array}{c}\lim \\ k \to \infty\end{array} C^r\left(e^{k+1}\right)^{r^2-r-1} = C' \tag{3.61}$$

Since $\lim_{k\to\infty} e^k = 0$, this relationship can only be satisfied if $r^2 - r - 1 = 0$, which has the solution

$$r = \frac{1+\sqrt{5}}{2} > 1 \tag{3.62}$$

and hence super linear convergence.

Example 3.6
Use the secant method to find a solution of

$$0 = e^{x^2-2} - 3\ln(x)$$

starting with $x^0 = 1.5$ and $x^1 = 1.4$.

Solution 3.6 Using Equation (3.49), the following set of values is obtained.

k	x^{k+1}	x^k	x^{k-1}	$f\left(x^k\right)$	$f\left(x^{k+1}\right)$
1	1.4418	1.4000	1.5000	−0.0486	0.0676
2	1.4617	1.4418	1.4000	−0.0157	−0.0486
3	1.4552	1.4617	1.4418	0.0076	−0.0157
4	1.4557	1.4552	1.4617	−0.0006	0.0076
5	1.4557	1.4557	1.4552	−0.0000	−0.0006

■

3.3.2 Broyden's Method

The generalization of the secant method to a system of equations is Broyden's method. Broyden's method is an example of using a secant update. Returning again to the equation

$$x^{k+1} = x^k - \left[M_k\left(x^k\right)\right]^{-1} f\left(x^k\right) \tag{3.63}$$

where the iteration matrix M_k is updated with each subsequent x^{k+1}. The updated matrix is found

$$M_{k+1} = M_k + \frac{\left(y - M_k s\right) s^T}{s^T s} \tag{3.64}$$

where

$$y = f\left(x^{k+1}\right) - f\left(x^k\right) \tag{3.65}$$

and

$$s = x^{k+1} - x^k \tag{3.66}$$

Substituting y from Equation (3.65) into Equation (3.64) yields

$$M_{k+1} = M_k + \frac{f\left(x^{k+1}\right) s^T}{s^T s} \tag{3.67}$$

Example 3.7
Repeat Example 3.5 using Broyden's method.

Solution 3.7 As before, the initial condition is

$$x^{(0)} = \begin{bmatrix} 3 \\ 3 \end{bmatrix}$$

This method also requires an initial guess for the iteration matrix M_0. One possible initial iteration matrix is the Jacobian matrix evaluated at x^0.

Initialization ($k = 0$)

$$f^0 = \begin{bmatrix} -4 \\ 12 \end{bmatrix}$$

$$M_0 = \begin{bmatrix} 1 & 6 \\ 6 & -9 \end{bmatrix} \text{ (both the same as in Example 3.5)}$$

Iteration 1: ($k = 1$)

$$\begin{aligned} x^1 &= x^0 - M_0^{-1} f^0 = \begin{bmatrix} 3.8000 \\ 2.2000 \end{bmatrix} \text{ (same as in Example 3.5)} \\ f^1 &= \begin{bmatrix} 1.2800 \\ 0.0000 \end{bmatrix} \\ M_1 &= M_0 + \frac{f^1 \left(x^1 - x^0\right)^T}{\left(x^1 - x^0\right)^T \left(x^1 - x^0\right)} \\ &= \begin{bmatrix} 1.8000 & 5.2000 \\ 6.0000 & -9.0000 \end{bmatrix} \\ \text{error} &= \|f^1\|_\infty = 1.2800 \end{aligned}$$

Iteration 2: ($k = 2$)

$$\begin{aligned} x^2 &= x^1 - M_1^{-1} f^1 = \begin{bmatrix} 3.5570 \\ 2.0380 \end{bmatrix} \text{ (same as in Example 3.5)} \\ f^2 &= \begin{bmatrix} 0.0205 \\ -0.6153 \end{bmatrix} \\ M_2 &= M_1 + \frac{f^2 \left(x^2 - x^1\right)^T}{\left(x^2 - x^1\right)^T \left(x^2 - x^1\right)} \\ &= \begin{bmatrix} 1.7416 & 5.1611 \\ 7.7527 & -7.8315 \end{bmatrix} \\ \text{error} &= \|f^2\|_\infty = 0.6153 \end{aligned}$$

This process continues until the error is driven sufficiently small. In the solution to Example 3.7, the error at each iteration is

iteration	error
0	12.0000
1	1.2800
2	0.6153
3	0.0506
4	0.0081
5	0.0001
6	0.0000

Note that it takes more iterations for this method to converge than the full Newton–Raphson. However, in most cases, the calculation of the iteration matrix M_k may require far less computation than the full Jacobian. The effort required is a trade-off between the computational effort required in calculating the Jacobian at every iteration and the number of iterations required for convergence. ■

3.3.3 Modifications to the Newton–Raphson Method

Another common simplification is to substitute each of the partial derivative entries $\frac{\partial f_i}{\partial x_j}$ by a difference approximation. For example, a simple approximation might be

$$\frac{\partial f_i}{\partial x_j} \approx \frac{1}{h_{ij}} \left[f_i \left(x + h_{ij} e^j \right) - f_i \left(x \right) \right] \tag{3.68}$$

where e^j is the jth unit vector:

$$e^j = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

where the 1 occurs in the jth row of the unit vector and all other entries are zero. The scalar h_{ij} can be chosen in numerous ways, but one common choice is to let $h_{ij}^k = x_j^k - x_j^{k-1}$. This choice for h_{ij} leads to a rate of convergence of 1.62, which lies between quadratic and linear convergence rates.

Another common modification to the Newton–Raphson method is to set M equal to the Jacobian matrix at occasional intervals. For example, the matrix M can be reevaluated whenever the convergence slows down, or at more regular intervals, such as every other or every third iteration. This modification is known as the *dishonest Newton* method. An extreme extension

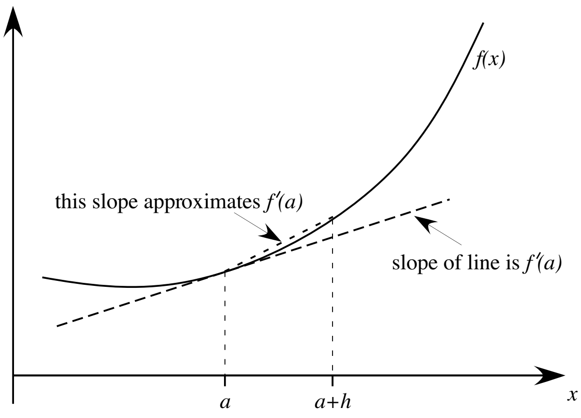


FIGURE 3.9
Graphical interpretation of the difference approximation of the slope of $f(a)$

of this method is to set M equal to the initial Jacobian matrix and then to hold it constant throughout the remainder of the iteration. This is commonly called the *very dishonest Newton* method. In addition to the reduction in computation associated with the calculation of the matrix, this method also has the advantage that the M matrix need only be factored into the LU matrices once since it is a constant. This can save considerable computation time in the LU factorization process. Similarly, the matrices of the dishonest Newton method need only be factored when the M matrix is reevaluated.

3.3.4 Numerical Differentiation

Using the Newton–Raphson method or any of its modifications requires the calculation of numerous partial derivatives. In many cases, the analytic computation of the partial derivative may be extremely complex or may be computationally expensive to compute. In these cases, it is desirable to compute the derivative numerically directly from the function $f(x)$ without explicit knowledge of $\frac{\partial f}{\partial x}$.

Consider the scalar function $f(x)$. The derivative of the function f' at the point $x = a$ is equivalent to the slope of the function $f(a)$. A reasonable approximation to the slope of a curve $f(a)$ is to use a nearby point $a + h$ to compute a *difference approximation*, as shown in Figure 3.9.

This has a mathematical basis that can be derived by application of the Taylor series expansion to $f(a + h)$:

$$f(a+h) = f(a) + h\frac{\partial f}{\partial x}(a) + \frac{h^2}{2!}\frac{\partial^2 f}{\partial x^2}(a) + \frac{h^3}{3!}\frac{\partial^3 f}{\partial x^3}(a) + \ldots \quad (3.69)$$

By rearranging

$$\frac{f(a+h)-f(a)}{h} = \frac{\partial f}{\partial x}(a) + \frac{h}{2!}\frac{\partial^2 f}{\partial x^2}(a) + \ldots \tag{3.70}$$

By neglecting the higher-order terms

$$\frac{\partial f}{\partial x}(a) \approx \frac{f(a+h)-f(a)}{h} \tag{3.71}$$

This approximation becomes increasingly more accurate as h becomes smaller (and is exact in the limit as $h \rightarrow 0$). This approximation is the one-sided difference approximation known as the *forward difference* approximation to the derivative of f. A similar approach can be taken in which the series is expanded about $a-h$ and

$$\frac{\partial f}{\partial x}(a) \approx \frac{f(a)-f(a-h)}{h} \tag{3.72}$$

which is the approximation known as the *backward difference* approximation. Consider now the combination of the two approaches:

$$\frac{\partial f}{\partial x}(a) \approx \frac{f(a+h)-f(a-h)}{2h} \tag{3.73}$$

This combination is often referred to as the *center difference* approximation, and is shown in Figure 3.10. The forward and backward difference approximations both have error on the order of $O(h)$, whereas the center approximation has an error on the order of $O\left(h^2\right)$ and will in general have better accuracy than either the forward or backward difference approximations.

Example 3.8

Consider the polynomial

$$f(x) = x^3 + x^2 - \frac{5}{4}x - \frac{3}{4}$$

Approximate the derivative of this polynomial in the range $[-2, 1.5]$ with $h = 0.2$ using the forward, backward, and center difference approximations.

Solution 3.8 The exact derivative of this function is given by

$$f'(x) = 3x^2 + 2x - \frac{5}{4}$$

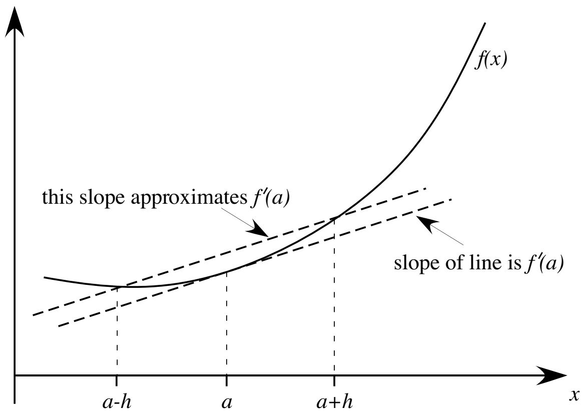


FIGURE 3.10
Graphical interpretation of the center difference approximation of the slope of $f(a)$

x	$f(x-h)$	$f(x)$	$f(x+h)$	$f'(x)$ backward	$f'(x)$ forward	$f'(x)$ center	$f'(x)$ exact
−2.0	−3.808	−2.250	−1.092	7.79	5.79	6.79	6.75
−1.8	−2.250	−1.092	−0.286	5.79	4.03	4.91	4.87
−1.6	−1.092	−0.286	0.216	4.03	2.51	3.27	3.23
−1.4	−0.286	0.216	0.462	2.51	1.23	1.87	1.83
−1.2	0.216	0.462	0.500	1.23	0.19	0.71	0.67
−1.0	0.462	0.500	0.378	0.19	−0.61	−0.21	−0.25
−0.8	0.500	0.378	0.144	−0.61	−1.17	−0.89	−0.93
−0.6	0.378	0.144	−0.154	−1.17	−1.49	−1.33	−1.37
−0.4	0.144	−0.154	−0.468	−1.49	−1.57	−1.53	−1.57
−0.2	−0.154	−0.468	−0.750	−1.57	−1.41	−1.49	−1.53
−0.0	−0.468	−0.750	−0.952	−1.41	−1.01	−1.21	−1.25
0.2	−0.750	−0.952	−1.026	−1.01	−0.37	−0.69	−0.73
0.4	−0.952	−1.026	−0.924	−0.37	0.51	0.07	0.03
0.6	−1.026	−0.924	−0.598	0.51	1.63	1.07	1.03
0.8	−0.924	−0.598	−0.000	1.63	2.99	2.31	2.27
1.0	−0.598	−0.000	0.918	2.99	4.59	3.79	3.75
1.2	−0.000	0.918	2.204	4.59	6.43	5.51	5.47
1.4	0.918	2.204	3.906	6.43	8.51	7.47	7.43

Figure 3.11 clearly shows the accuracy levels of the different derivative approximations. ■

By continuing in the same approach of using Taylor series expansions and including additional information, increasingly more accurate approximations can be achieved. One such approximation that is widely used is the *Richardson*

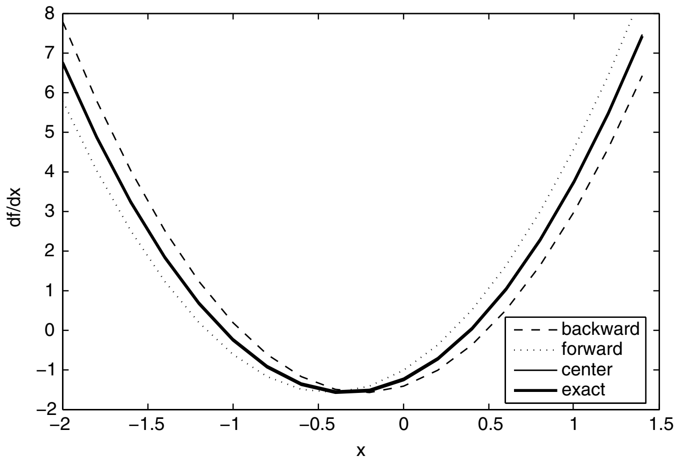


FIGURE 3.11
Exact versus approximate derivatives

approximation:

$$f'(x) \approx \frac{f(x-2h) - 8f(x-h) + 8f(x+h) - f(x+2h)}{12h} \tag{3.74}$$

This approximation has an error on the order $O\left(h^4\right)$.

Consider once again the Newton–Raphson method, which requires the calculation of the Jacobian. The approximations can be utilized to calculate the derivatives in the Jacobian rather than a direct analytic calculation. For example, consider the system of nonlinear equations:

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

The Jacobian for this system consists of partial derivatives of the form $\frac{\partial f_i}{\partial x_j}$, which can now be approximated using one of the approximation methods introduced. For example, using the center difference

$$\frac{\partial f_i}{\partial x_j} = \frac{f_i(x_j + \Delta x_j) - f_i(x_j - \Delta x_j)}{2\Delta x_j}$$

where Δx_j is usually chosen to be a small incremental change (of about 1%).

3.3.5 Newton–GMRES

The discussions of the previous sections all assumed that the series of linear equations obtained from the Newton–Raphson series expansion were solved using the LU factorization direct method. There is no underlying principle that requires that direct methods be used. The linear equations could just as easily be solved with one of the iterative methods described in Chapter 2. Note that using an iterative method to solve linear system equations leads to a set of nested iteration loops: an outer loop for the nonlinear equations and an inner loop for the linear equations.

Consider the solution of the linear equations. At each iteration of the outer loop $x^{k+1} = x^k - M_k^{-1} f(x^k)$, the linear iterates will be driven to convergence to solve for x^k, even though, for low values of k, these updates can be significantly inaccurate. Therefore, it is plausible to reduce the number of linear system iterations in the inner loop without having an effect on the overall convergence of the outer iteration loop. These mixed linear–nonlinear iterative methods are named to reflect the linear method they use, such as Newton–SOR, Newton–CG, and Newton–GMRES. In this section, the approach to the Newton–GMRES method will be discussed, although the basic procedure is applicable to the Newton–SOR and Newton–CG methods as well.

If the linear iterative method is one of the Krylov subspace methods, then each inner loop iteration requires the evaluation of the action of the Jacobian $\frac{\partial f}{\partial x} = J\left(x^k\right)$ on a vector. In many implementations, the action of $\frac{\partial f}{\partial x}$ on a vector w can be approximated by a forward difference $D_h f(x : w)$ for some h. Note that this is entirely different from forming the Jacobian matrix using a forward difference approximation and then multiplying that matrix by w. This approach of directly finding the vector product is often referred to as a Jacobian-free Newton–GMRES method, since the Jacobian matrix is not explicitly calculated.

Jacobian-Free Newton–GMRES Algorithm for Solving $F(x) = 0$

1. Outer Loop (Newton):

   Initialize outer loop:

$$\begin{aligned} x &= x^0 \\ F^0 &= F\left(x^0\right) \\ q &= 0 \end{aligned}$$

   While $\left\|F(x^k)\right\| \geq \text{tol}$ and $q \leq q_{\max}$,

2. Inner Loop (GMRES):

$$\begin{aligned} k &= 1 \\ r_0 &= -F(x^q) \end{aligned}$$

$$\begin{aligned} v_1 &= r_0/\|r_0\| \\ y &= [0\ \ 0\ \ 0\ \ \ldots 0]^T \\ \text{error} &= \|r_0\| \end{aligned}$$

While error $\geq \varepsilon$

(a) Calculate $D_h f(x : v_k)$

(b) $H(j,k) = (D_h f(x : v_k))^T v_j, \quad j = 1, \ldots, k$

(c) $v_{k+1} = D_h f(x : v_k) - \sum_{j=1}^{k} H(j,k) v_j$

(d) $H(k+1,k) = \|v_{k+1}\|$

(e) $v_{k+1} = v_{k+1}/\|v_{k+1}\|$

(f) Givens rotation:

i.

$$\begin{bmatrix} H(j,k) \\ H(j+1,k) \end{bmatrix} = \begin{bmatrix} \text{cs}(j) & \text{sn}(j) \\ -\text{sn}(j) & \text{cs}(j) \end{bmatrix} \begin{bmatrix} H(j,k) \\ H(j+1,k) \end{bmatrix}$$
$$j = 1, \ldots, k-1$$

ii.

$$\text{cs}(k) = \frac{H(k,k)}{\sqrt{H(k+1,k)^2 + H(k,k)^2}}$$
$$\text{sn}(k) = \frac{H(k+1,k)}{\sqrt{H(k+1,k)^2 + H(k,k)^2}}$$

iii. Approximate residual norm

$$\begin{aligned} \alpha &= \text{cs}(k) s(k) \\ s(k+1) &= -\text{sn}(k) s(k) \\ s(k) &= \alpha \\ \text{error} &= |s(k+1)| \end{aligned}$$

iv. Set

$$\begin{aligned} H(k,k) &= \text{cs}(k) H(k,k) + \text{sn}(k) H(k+1,k) \\ H(k+1,k) &= 0 \end{aligned}$$

3. (outer loop)

$$\begin{aligned} &\text{Solve } Hz = s \text{ for } z \\ &x^q = x^q + v_k z \\ &q = q + 1 \end{aligned}$$

Example 3.9

Repeat Example 3.5 using the Jacobian-Free Newton–GMRES method using $h = 0.001$ for the directional derivative calculation.

Solution 3.9

1. Initialize the outer loop:

$$\begin{aligned} q &= 1 \\ x^{(0)} &= \begin{bmatrix} 3 \\ 3 \end{bmatrix} \end{aligned}$$

   and

$$F(x^0) = \begin{bmatrix} 4 \\ -12 \end{bmatrix}$$

2. Starting the inner loop (GMRES iteration):

$$\begin{aligned} k &= 1 \\ r_0 &= -F(x^0) = \begin{bmatrix} -4 \\ 12 \end{bmatrix} \\ v_1 &= r_0/\|r_0\| = \begin{bmatrix} -0.3162 \\ 0.9487 \end{bmatrix} \end{aligned}$$

   Note that the norm used in this example is the 2-norm.

   Inner loop $k = 1$:

   (a) Calculate $D_h f(x : v_1)$

$$\begin{aligned} D_h f(x : v_1) &= \frac{F\left(x^0 + hv_1\right) - F\left(x^0\right)}{h} \\ x^0 + hv_1 &= \begin{bmatrix} 2.9997 \\ 3.0009 \end{bmatrix} \\ F\left(x^0 + hv_1\right) &= \begin{bmatrix} 4.0054 \\ -12.0104 \end{bmatrix} \\ D_h f(x : v_1) &= \begin{bmatrix} 5.3769 \\ -10.4363 \end{bmatrix} \end{aligned}$$

   (b) $H(1,1) = (D_h f(x : v_1))^T v_1 = -11.6011$

   (c) $v_2 = D_h f(x : v_1) = \begin{bmatrix} 5.3769 \\ -10.4363 \end{bmatrix}$

   (d) $H(2,1) = \|v_2\| = 1.8007$

   (e) $v_2 = v_2/\|v_2\| = \begin{bmatrix} 0.9487 \\ 0.3162 \end{bmatrix}$

(f) Givens rotation:

$$\begin{aligned} H &= 11.7400 \\ \text{error } &= |s(2)| = 1.9401 > \text{ tol, therefore } k = 2, \text{ repeat} \end{aligned}$$

Inner loop $k = 2$:

(a) Calculate $D_h f(x : v_2)$

$$\begin{aligned} D_h f(x : v_2) &= \frac{F\left(x^0 + hv_2\right) - F\left(x^0\right)}{h} \\ x^0 + hv_2 &= \begin{bmatrix} 3.0009 \\ 3.0003 \end{bmatrix} \\ F\left(x^0 + hv_2\right) &= \begin{bmatrix} 4.0028 \\ -11.9972 \end{bmatrix} \\ D_h f(x : v_2) &= \begin{bmatrix} 2.8470 \\ 2.8468 \end{bmatrix} \end{aligned}$$

(b)

$$H = \begin{bmatrix} 11.7400 & 1.8004 \\ 0 & 3.6012 \end{bmatrix}$$

(c)

$$v_3 = D_h f(x : v_2) - \sum_{j=1}^{2} H(j, 2) v_j = \begin{bmatrix} 0.1104 \\ 0.9939 \end{bmatrix}$$

(d) $H(3, 2) = 0$

(e) $v_3 = v_3 / \|v_3\| = \begin{bmatrix} 0.1104 \\ 0.9939 \end{bmatrix}$

(f) Givens rotation:

$$\begin{aligned} H &= \begin{bmatrix} 11.7400 & -1.2268 \\ 0.0000 & 3.8347 \end{bmatrix} \\ \text{error } &= |s(3)| = 2.0346 \times 10^{-15} < \text{ tol, therefore proceed} \end{aligned}$$

3. Solve $Hz = s$ for z

$$\begin{aligned} \begin{bmatrix} 11.7400 & -1.2268 \\ 0 & 3.8347 \end{bmatrix} z &= \begin{bmatrix} -12.4994 \\ 1.9401 \end{bmatrix} \\ z &= \begin{bmatrix} -1.0118 \\ 0.5059 \end{bmatrix} \\ x^1 &= x^0 + v_3 z \end{aligned}$$

$$
\begin{aligned}
x^1 &= \begin{bmatrix} 3 \\ 3 \end{bmatrix} + \begin{bmatrix} -0.3163 & 0.9487 \\ 0.9487 & 0.3162 \end{bmatrix} \begin{bmatrix} -1.0118 \\ 0.5059 \end{bmatrix} = \begin{bmatrix} 3.7999 \\ 2.2001 \end{bmatrix} \\
F\left(x^1\right) &= \begin{bmatrix} 1.2803 \\ -0.0012 \end{bmatrix} \\
q &= 2
\end{aligned}
$$

Since $\|F\left(x^1\right)\| >$ tol, the outer loop continues and the inner loop is started again with $k = 1$. The process repeats until the convergence tolerance criterion is met, and x converges to

$$x = \begin{bmatrix} 3.6100 \\ 2.0045 \end{bmatrix}$$

which is the same solution as before. ■

In the solution to Example 3.9, the error at each iteration is

iteration	error
0	12.0000
1	1.2803
2	0.0661
3	0.0002

which are nearly identical to the errors obtained in the true Newton–Raphson solution in Example 3.5. The potential savings from using a Jacobian-free Newton–GMRES method arise from not having to calculate the system Jacobian at every step. Furthermore, it is possible that the GMRES iteration may not require a full number of iterations ($k < n$) as x^q approaches convergence.

3.4 Continuation Methods

Many of the iteration methods described so far will, in general, converge to a solution $x^\star$ of $f(x) = 0$ only if the initial condition is sufficiently close to $x^\star$. The continuation method approach may be considered to be an attempt to widen the region of convergence of a given method. In many physical systems, the problem defined by the mathematical equation $f(x) = 0$ may in some way depend in a natural way on a parameter λ of the system. When this parameter is set equal to 0, the system $f_0(x) = 0$ has a known solution x^0. However, for varying λ, an entire family of functions $H(x, \lambda)$ exists such that

$$H(x, 0) = f_0(x), \quad H(x, 1) = f(x) \tag{3.75}$$

where a solution x^0 of $H(x, 0) = 0$ is known, and the equation $H(x, 1) = 0$ is the desired problem to be solved.

Even if $f(x)$ does not depend naturally on a suitable parameter λ, a family of problems satisfying Equation (3.75) can be defined by

$$H(x, \lambda) = \lambda f(x) + (1-\lambda) f_0(x), \quad \lambda \in [0,\ 1] \tag{3.76}$$

when the solution x^0 of $f_0(x) = 0$ is known. As λ varies from 0 to 1, the family of mappings varies from $f_0(x) = 0$ to $f_1(x) = 0$ where the solution of $f_1(x) = f(x) = 0$ is the desired value $x^1 = x^\star$.

As a first approach to obtaining $x^\star = x^1$, the interval [0, 1] can be partitioned as

$$0 = \lambda_0 < \lambda_1 < \lambda_2 < \ldots < \lambda_N = 1$$

and consider solving the problems

$$H(x, \lambda_i) = 0, \quad i = 1, \ldots, N \tag{3.77}$$

Assuming that a Newton–Raphson iteration is used to solve each problem i in Equation (3.77), then the initial condition for the ith problem is the solution from $H(x, \lambda_{i-1}) = 0$. For small enough intervals between i and $i+1$, this solves the problem of identifying a good initial condition.

The relationship given in Equation (3.76) is an example of a *homotopy* in which two functions $f(x)$ and $f_0(x)$ are embedded in a single continuous function. Formally, a homotopy between any two functions is a continuous mapping $f, f_0 : X \to Y$:

$$H : [0,\ 1] \times X \to Y \tag{3.78}$$

such that Equation (3.75) holds. If such a mapping exists, then it is said that f is *homotopic* to f_0.

Homotopy functions are used to define a path from the known solution to a relatively simple problem ($f_0(x) = 0$) to the solution of a more complex problem to which the solution is desired ($f(x) = 0$) . This path comprises the solutions to a family of problems which represent the continuous deformation from the simple problem to the desired problem. Continuation methods are a numerical approach to following the deformation path.

Homotopy continuation methods can be constructed to be exhaustive and globally convergent, meaning that all solutions to a given system of nonlinear equations can be found and will converge regardless of choice of initial condition [63]. Since a homotopy problem is equal to zero at every point $\lambda \in [0,\ 1]$ along the path, it is therefore equal to zero at $\lambda = \lambda^k$ and $\lambda = \lambda^{k+1}$, which are two successive points along the path. This gives

$$0 = H\left(x^k, \lambda^k\right) = \lambda^k f(x^k) + \left(1 - \lambda^k\right) f_0(x^k) \tag{3.79}$$

$$0 = H\left(x^{k+1}, \lambda^{k+1}\right) = \lambda^{k+1} f(x^{k+1}) + \left(1 - \lambda^{k+1}\right) f_0(x^{k+1}) \tag{3.80}$$

For paths along the path, the homotopy parameter λ^{k+1} is associated with the parameter set

$$x^{k+1} = x^k + \Delta x$$

If the changes in the parameters are small, the functions $f_0(x^{k+1})$ and $f(x^{k+1})$ can be linearly approximated by using a Taylor series expansion about x^k and neglecting all terms higher than second order. Applying this technique to Equation (3.80) yields

$$\left(\lambda^k + \Delta\lambda\right)\left[F_x\left(x^k\right)\Delta x\right] + \left(1 - \lambda^k - \Delta\lambda\right)\left[f_0\left(x^k\right) + F_{0x}\left(x^k\right)\Delta x\right] = 0 \quad (3.81)$$

where F_x and F_{0x} are the Jacobians of $f(x)$ and $f_0(x)$ with respect to x, respectively. Subtracting Equation (3.79) from Equation (3.81) and canceling like terms yields

$$0 = \left[\lambda^{k+1}F_x\left(x^k\right) + \left(1 - \lambda^{k+1}\right)F_{0x}\left(x^k\right)\right]\Delta x + \left[f\left(x^k\right) - f_0\left(x^k\right)\right]\Delta\lambda \quad (3.82)$$

Using $x^{k+1} = x^k + \Delta x$, Equation (3.82) can be rewritten in terms of the homotopy function to yield the update equation

$$x^{k+1} = x^k - \Delta\lambda H_x\left(x^k, \lambda^{k+1}\right)^{-1}\frac{\partial}{\partial\lambda}H\left(x^k, \lambda^{k+1}\right) \quad (3.83)$$

where

$$\lambda^{k+1} = \lambda^k + \Delta\lambda$$

and $H_x(x, \lambda)$ is the Jacobian of $H(x, \lambda)$ with respect to x.

Example 3.10

Solve

$$0 = f_1(x_1, x_2) = x_1^2 - 3x_2^2 + 3 \quad (3.84)$$
$$0 = f_2(x_1, x_2) = x_1x_2 + 6 \quad (3.85)$$

using the homotopic mapping with

$$0 = f_{01}(x_1, x_2) = x_1^2 - 4 \quad (3.86)$$
$$0 = f_{02}(x_1, x_2) = x_2^2 - 9 \quad (3.87)$$

Solution 3.10 Set up a homotopy such that

$$H(x, \lambda) = \lambda f(x) + (1 - \lambda) f_0(x), \quad \lambda \in [0,\ 1] \quad (3.88)$$

$$0 = \lambda\left(x_1^2 - 3x_2^2 + 3\right) + (1 - \lambda)\left(x_1^2 - 4\right) \quad (3.89)$$
$$0 = \lambda\left(x_1x_2 + 6\right) + (1 - \lambda)\left(x_2^2 - 9\right) \quad (3.90)$$

The continuation method advances the solution via Equation (3.83):

$$\lambda^{k+1} = \lambda^k + \Delta\lambda \tag{3.91}$$

$$\begin{bmatrix} x_1^{k+1} \\ x_2^{k+1} \end{bmatrix} = \begin{bmatrix} x_1^k \\ x_2^k \end{bmatrix} - \Delta\lambda \left[\lambda^{k+1} \begin{bmatrix} 2x_1^k & -6x_2^k \\ x_2^k & x_1^k \end{bmatrix} + \left(1 - \lambda^{k+1}\right) \begin{bmatrix} 2x_1^k & 0 \\ 0 & 2x_2^k \end{bmatrix} \right]^{-1} \times \\ \begin{bmatrix} (x_1^k)^2 - 3(x_2^k)^2 + 3 - \left((x_1^k)^2 - 4\right) \\ x_1^k x_2^k + 6 - \left((x_2^k)^2 - 9\right) \end{bmatrix} \tag{3.92}$$

The solution is then refined through the Newton–Raphson solution for x_1^{k+1} and x_2^{k+1} from

$$0 = \lambda^{k+1} \left((x_1^{k+1})^2 - 3(x_2^{k+1})^2 + 3\right) + (1 - \lambda^{k+1}) \left((x_1^{k+1})^2 - 4\right) \tag{3.93}$$

$$0 = \lambda^{k+1} \left(x_1^{k+1} x_2^{k+1} + 6\right) + (1 - \lambda^{k+1}) \left((x_2^{k+1})^2 - 9\right) \tag{3.94}$$

Starting with $\lambda^0 = 0$ and $\Delta\lambda = 0.1$ yields the easily obtained initial solution of the system:

$$x_1^0 = 2$$
$$x_2^0 = 3$$

Predicting the values for $k = 1$ from Equations (3.91) and (3.92) yields

$$x_1^1 = 2.3941$$
$$x_2^1 = 2.7646$$

Refining the solution via the Newton–Raphson solution to Equations (3.93) and (3.94) yields

$$x_1^1 = 2.3628$$
$$x_2^1 = 2.7585$$

This process is repeated until $\lambda = 1$ and $x_1 = -3$ and $x_2 = 2$, which are the correct solutions to the desired problem.

The same process will work if the initial solutions are chosen as $x_1^0 = -2$ and $x_2^0 = -3$. In this case, the values obtained are the alternate solution $x_1 = 3$ and $x_2 = -2$ to the desired problem. ■

3.5 Power System Applications

The solution and analysis procedures outlined in this chapter form the basis of a set of powerful tools that can be used for a myriad of power system applications. One of the most outstanding features of power systems is that they are modeled as an extremely large set of nonlinear equations. The North American transmission grid is one of the largest nonlinear engineering systems. Most types of power system analysis require the solution in one form or another of this system of nonlinear equations. The applications described below are a handful of the more common applications, but are certainly not a complete coverage of all possible nonlinear problems that arise in power system analysis.

3.5.1 Power Flow

Many power system problems give rise to systems of nonlinear equations that must be solved. Probably the most common nonlinear power system problem is the *power flow* or *power flow* problem. The underlying principle of a power flow problem is that, given the system loads, generation, and network configuration, the system bus voltages and line flows can be found by solving the nonlinear power flow equations. This is typically accomplished by applying Kirchoff's law at each power system bus throughout the system. In this context, Kirchoff's law can be interpreted as *the sum of the powers entering a bus must be zero,* or that the power at each bus must be conserved. Since power is comprised of two components, active power and reactive power, each bus gives rise to two equations – one for active power and one for reactive power. These equations are known as the *power flow equations*:

$$0 = \Delta P_i = P_i^{inj} - V_i \sum_{j=1}^{N_{bus}} V_j Y_{ij} \cos(\theta_i - \theta_j - \phi_{ij}) \tag{3.95}$$

$$0 = \Delta Q_i = Q_i^{inj} - V_i \sum_{j=1}^{N_{bus}} V_j Y_{ij} \sin(\theta_i - \theta_j - \phi_{ij}) \tag{3.96}$$

$$i = 1, \ldots, N_{bus}$$

where P_i^{inj}, Q_i^{inj} are the active and reactive power injected at the bus i, respectively. Loads are modeled by negative power injection. The values V_i and V_j are the voltage magnitudes at bus i and bus j, respectively. The values θ_i and θ_j are the corresponding phase angles. The value $Y_{ij}\angle\phi_{ij}$ is the (ij)th element of the network admittance matrix Y. The constant N_{bus} is the number of buses in the system. The updates ΔP_i^k and ΔQ_i^k of Equations (3.95) and (3.96) are called the *mismatch* equations because they give a measure of the power difference, or mismatch, between the calculated power values as functions of voltage and phase angle, and the actual injected powers. As the Newton–Raphson iteration continues, this mismatch is driven to zero until the power leaving a bus, calculated from the voltages and phase angles, equals the injected power. At this point, the converged values of voltages and phase angles are used to calculate line flows, slack bus powers, and the injected reactive powers at the generator buses.

The formulation in Equations (3.95) and (3.96) is called the *polar* formulation of the power flow equations. If $Y_{ij}\angle\phi_{ij}$ is instead given by the complex sum $g_{ij} + jb_{ij}$, then the power flow equations may be written in *rectangular form* as

$$0 = P_i^{inj} - V_i \sum_{j=1}^{N_{bus}} V_j \left(g_{ij}\cos(\theta_i - \theta_j) + b_{ij}\sin(\theta_i - \theta_j)\right) \tag{3.97}$$

$$0 = Q_i^{inj} - V_i \sum_{j=1}^{N_{bus}} V_j \left(g_{ij}\sin(\theta_i - \theta_j) - b_{ij}\cos(\theta_i - \theta_j)\right) \tag{3.98}$$

$$i = 1, \ldots, N_{bus}$$

In either case, the power flow equations are a system of nonlinear equations. They are nonlinear in both the voltage and phase angle. There are, at most, $2N_{bus}$ equations to solve. This number is then further reduced by removing one power flow equation for each known voltage (at voltage controlled buses) and the slack bus angle. This reduction is necessary since the number of equations must equal the number of unknowns in a fully determined system. Once the nonlinear power flow equations have been determined, the Newton–Raphson method may be directly applied.

The most common approach to solving the power flow equations by the Newton–Raphson method is to arrange the equations by phase angle followed by the voltage magnitudes as

$$\begin{bmatrix} J_1 & J_2 \\ J_3 & J_4 \end{bmatrix} \begin{bmatrix} \Delta\theta_1 \\ \Delta\theta_2 \\ \Delta\theta_3 \\ \vdots \\ \Delta\theta_{N_{bus}} \\ \Delta V_1 \\ \Delta V_2 \\ \Delta V_3 \\ \vdots \\ \Delta V_{N_{bus}} \end{bmatrix} = - \begin{bmatrix} \Delta P_1 \\ \Delta P_2 \\ \Delta P_3 \\ \vdots \\ \Delta P_{N_{bus}} \\ \Delta Q_1 \\ \Delta Q_2 \\ \Delta Q_3 \\ \vdots \\ \Delta Q_{N_{bus}} \end{bmatrix} \tag{3.99}$$

where

$$\Delta\theta_i = \theta_i^{k+1} - \theta_i^k$$
$$\Delta V_i = V_i^{k+1} - V_i^k$$

These equations are then solved using LU factorization with forward/backward substitution. The Jacobian is typically divided into four submatrices, where

$$\begin{bmatrix} J_1 & J_2 \\ J_3 & J_4 \end{bmatrix} = \begin{bmatrix} \frac{\partial\Delta P}{\partial\theta} & \frac{\partial\Delta P}{\partial V} \\ \frac{\partial\Delta Q}{\partial\theta} & \frac{\partial\Delta Q}{\partial V} \end{bmatrix} \tag{3.100}$$

Each submatrix represents the partial derivatives of each of the mismatch equations with respect to each of the unknowns. These partial derivatives yield eight types – two for each mismatch equation, where one is for the diagonal element and the other is for off-diagonal elements. The derivatives are summarized as

$$\frac{\partial \Delta P_i}{\partial \theta_i} = V_i \sum_{j=1}^{N_{bus}} V_j Y_{ij} \sin(\theta_i - \theta_j - \phi_{ij}) + V_i^2 Y_{ii} \sin \phi_{ii} \tag{3.101}$$

$$\frac{\partial \Delta P_i}{\partial \theta_j} = -V_i V_j Y_{ij} \sin(\theta_i - \theta_j - \phi_{ij}) \tag{3.102}$$

$$\frac{\partial \Delta P_i}{\partial V_i} = -\sum_{i=1}^{N_{bus}} V_j Y_{ij} \cos(\theta_i - \theta_j - \phi_{ij}) - V_i Y_{ii} \cos \phi_{ii} \tag{3.103}$$

$$\frac{\partial \Delta P_i}{\partial V_j} = -V_i Y_{ij} \cos(\theta_i - \theta_j - \phi_{ij}) \tag{3.104}$$

$$\frac{\partial \Delta Q_i}{\partial \theta_i} = -V_i \sum_{j=1}^{N_{bus}} V_j Y_{ij} \cos(\theta_i - \theta_j - \phi_{ij}) + V_i^2 Y_{ii} \cos \phi_{ii} \tag{3.105}$$

$$\frac{\partial \Delta Q_i}{\partial \theta_j} = V_i V_j Y_{ij} \cos(\theta_i - \theta_j - \phi_{ij}) \tag{3.106}$$

$$\frac{\partial \Delta Q_i}{\partial V_i} = -\sum_{j=1}^{N_{bus}} V_j Y_{ij} \sin(\theta_i - \theta_j - \phi_{ij}) + V_i Y_{ii} \sin \phi_{ii} \tag{3.107}$$

$$\frac{\partial \Delta Q_i}{\partial V_j} = -V_i Y_{ij} \sin(\theta_i - \theta_j - \phi_{ij}) \tag{3.108}$$

A common modification to the power flow solution is to replace the unknown update ΔV_i by the normalized value $\frac{\Delta V_i}{V_i}$. This formulation yields a more symmetric Jacobian, as the Jacobian submatrices J_2 and J_4 are now multiplied by V_i to compensate for the scaling of ΔV_i by V_i. All partial derivatives of each submatrix then become quadratic in voltage magnitude.

The Newton–Raphson method for the solution of the power flow equations is relatively straightforward to program, since both the function evaluations and the partial derivatives use the same expressions. Thus it takes little extra computational effort to compute the Jacobian once the mismatch equations have been calculated.

Example 3.11

Find the voltage magnitudes, phase angles, and line flows for the small power system shown in Figure 3.12 with the following per unit system parameters:

bus	type	V	P_{gen}	Q_{gen}	P_{load}	Q_{load}
1	slack	1.02	–	–	0.0	0.0
2	PV	1.00	0.5	–	0.0	0.0
3	PQ	–	0.0	0.0	1.2	0.5

i	j	R_{ij}	X_{ij}	B_{ij}
1	2	0.02	0.3	0.15
1	3	0.01	0.1	0.1
2	3	0.01	0.1	0.1

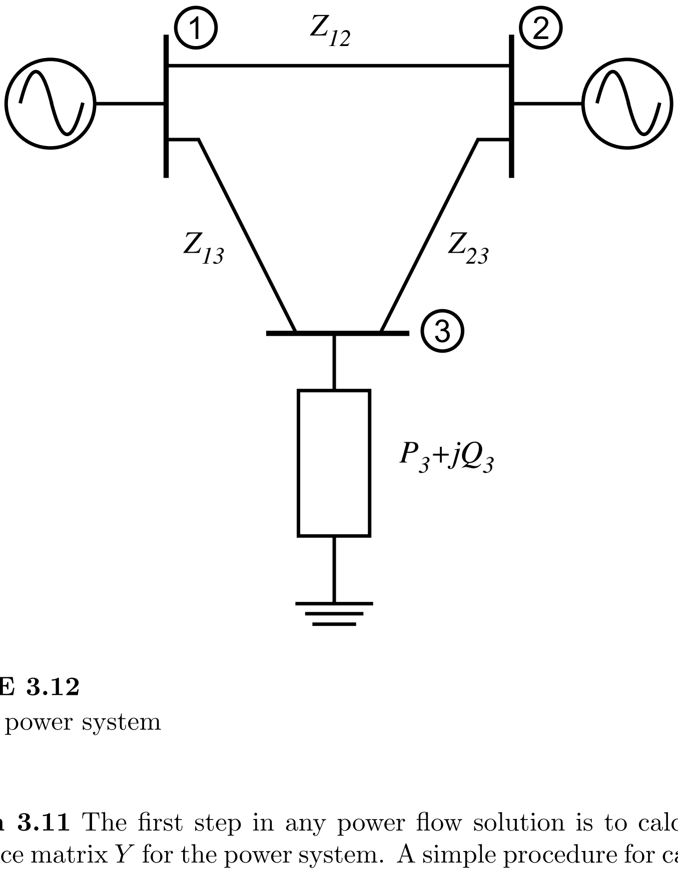


FIGURE 3.12
Example power system

Solution 3.11 The first step in any power flow solution is to calculate the admittance matrix Y for the power system. A simple procedure for calculating the elements of the admittance matrix is

$Y(i,j)$	negative of the admittance between buses i and j
$Y(i,i)$	sum of all admittances connected to bus i

Calculating the admittance matrix for this system yields

$$Y = \begin{bmatrix} 13.1505\angle -84.7148^\circ & 3.3260\angle 93.8141^\circ & 9.9504\angle 95.7106^\circ \\ 3.3260\angle 93.8141^\circ & 13.1505\angle -84.7148^\circ & 9.9504\angle 95.7106^\circ \\ 9.9504\angle 95.7106^\circ & 9.9504\angle 95.7106^\circ & 19.8012\angle -84.2606^\circ \end{bmatrix} \tag{3.109}$$

By inspection, this system has three unknowns: θ_2, θ_3, and V_3; thus three power flow equations are required. These power flow equations are

$$0 = \Delta P_2 = 0.5 - V_2 \sum_{j=1}^{3} V_j Y_{ij} \cos(\theta_2 - \theta_j - \theta_{ij}) \tag{3.110}$$

$$0 = \Delta P_3 = -1.2 - V_3 \sum_{j=1}^{3} V_j Y_{ij} \cos(\theta_3 - \theta_j - \theta_{ij}) \tag{3.111}$$

$$0 = \Delta Q_3 = -0.5 - V_3 \sum_{j=1}^{3} V_j Y_{ij} \sin(\theta_3 - \theta_j - \theta_{ij}) \tag{3.112}$$

Substituting in the known quantities for $V_1 = 1.02, V_2 = 1.00$, and $\theta_1 = 0$ and the admittance matrix quantities yields

$$\begin{aligned} \Delta P_2 = 0.5 &- (1.00)\left((1.02)(3.3260)\cos(\theta_2 - 0 - 93.8141^\circ)\right. \\ &+ (1.00)(13.1505)\cos(\theta_2 - \theta_2 + 84.7148^\circ) \\ &+ \left.(V_3)(9.9504)\cos(\theta_2 - \theta_3 - 95.7106^\circ)\right) \end{aligned} \tag{3.113}$$

$$\begin{aligned} \Delta P_3 = -1.2 &- (V_3)\left((1.02)(9.9504)\cos(\theta_3 - 0 - 95.7106^\circ)\right. \\ &+ (1.00)(9.9504)\cos(\theta_3 - \theta_2 - 95.7106^\circ) \\ &+ \left.(V_3)(19.8012)\cos(\theta_3 - \theta_3 + 84.2606^\circ)\right) \end{aligned} \tag{3.114}$$

$$\begin{aligned} \Delta Q_3 = -0.5 &- (V_3)\left((1.02)(9.9504)\sin(\theta_3 - 0 - 95.7106^\circ)\right. \\ &+ (1.00)(9.9504)\sin(\theta_3 - \theta_2 - 95.7106^\circ) \\ &+ \left.((V_3)(19.8012)\sin(\theta_3 - \theta_3 + 84.2606^\circ)\right) \end{aligned} \tag{3.115}$$

The Newton–Raphson iteration for this system is then given by

$$\begin{bmatrix} \frac{\partial \Delta P_2}{\partial \theta_2} & \frac{\partial \Delta P_2}{\partial \theta_3} & \frac{\partial \Delta P_2}{\partial V_3} \\ \frac{\partial \Delta P_3}{\partial \theta_2} & \frac{\partial \Delta P_3}{\partial \theta_3} & \frac{\partial \Delta P_3}{\partial V_3} \\ \frac{\partial \Delta Q_3}{\partial \theta_2} & \frac{\partial \Delta Q_3}{\partial \theta_3} & \frac{\partial \Delta Q_3}{\partial V_3} \end{bmatrix} \begin{bmatrix} \Delta\theta_2 \\ \Delta\theta_3 \\ \Delta V_3 \end{bmatrix} = - \begin{bmatrix} \Delta P_2 \\ \Delta P_3 \\ \Delta Q_3 \end{bmatrix} \tag{3.116}$$

where

$$\begin{aligned} \frac{\partial \Delta P_2}{\partial \theta_2} &= 3.3925 \sin(\theta_2 - 93.8141^\circ) \\ &\quad + 9.9504 V_3 \sin(\theta_2 - \theta_3 - 95.7106^\circ) \end{aligned}$$

$$\begin{aligned}
\frac{\partial \Delta P_2}{\partial \theta_3} &= -9.9504 V_3 \sin(\theta_2 - \theta_3 - 95.7106^\circ) \\
\frac{\partial \Delta P_2}{\partial V_3} &= -9.9504 \cos(\theta_2 - \theta_3 - 95.7106^\circ) \\
\frac{\partial \Delta P_3}{\partial \theta_2} &= -9.9504 V_3 \sin(\theta_3 - \theta_2 - 95.7106^\circ) \\
\frac{\partial \Delta P_3}{\partial \theta_3} &= 10.1494 V_3 \sin(\theta_3 - 95.7106^\circ) \\
&\quad +9.9504 V_3 \sin(\theta_3 - \theta_2 - 95.7106^\circ) \\
\frac{\partial \Delta P_3}{\partial V_3} &= -10.1494 \cos(\theta_3 - 95.7106^\circ) \\
&\quad -9.9504 \cos(\theta_3 - \theta_2 - 95.7106^\circ) \\
&\quad -39.6024 V_3 \cos(84.2606^\circ) \\
\frac{\partial \Delta Q_3}{\partial \theta_2} &= 9.9504 V_3 \cos(\theta_3 - \theta_2 - 95.7106^\circ) \\
\frac{\partial \Delta Q_3}{\partial \theta_3} &= -10.1494 V_3 \cos(\theta_3 - 95.7106^\circ) \\
&\quad -9.9504 V_3 \cos(\theta_3 - \theta_2 - 95.7106^\circ) \\
\frac{\partial \Delta Q_3}{\partial V_3} &= -10.1494 \sin(\theta_3 - 95.7106^\circ) \\
&\quad -9.9504 \sin(\theta_3 - \theta_2 - 95.7106^\circ) \\
&\quad -39.6024 V_3 \sin(84.2606^\circ)
\end{aligned}$$

Recall that one of the underlying assumptions of the Newton–Raphson iteration is that the higher-order terms of the Taylor series expansion are negligible only if the initial guess is sufficiently close to the actual solution to the nonlinear equations. Under most operating conditions, the voltages throughout the power system are within ±10% of the nominal voltage and therefore fall in the range $0.9 \leq V_i \leq 1.1$ per unit. Similarly, under most operating conditions, the phase angle differences between adjacent buses are typically small. Thus, if the slack bus angle is taken to be zero, then all phase angles throughout the system will also be close to zero. Therefore, in initializing a power flow, it is common to choose a "flat start" initial condition. That is, all voltage magnitudes are set to 1.0 per unit and all angles are set to zero.

Iteration 1

Evaluating the Jacobian and the mismatch equations at the flat start initial conditions yields

$$\left[J^0\right] = \begin{bmatrix} -13.2859 & 9.9010 & 0.9901 \\ 9.9010 & -20.0000 & -1.9604 \\ -0.9901 & 2.0000 & -19.4040 \end{bmatrix}$$

$$\begin{bmatrix} \Delta P_2^0 \\ \Delta P_3^0 \\ \Delta Q_3^0 \end{bmatrix} = \begin{bmatrix} 0.5044 \\ -1.1802 \\ -0.2020 \end{bmatrix}$$

Solving

$$[J^0] \begin{bmatrix} \Delta\theta_2^1 \\ \Delta\theta_3^1 \\ \Delta V_3^1 \end{bmatrix} = - \begin{bmatrix} \Delta P_2^0 \\ \Delta P_3^0 \\ \Delta Q_3^0 \end{bmatrix}$$

by LU factorization yields

$$\begin{bmatrix} \Delta\theta_2^1 \\ \Delta\theta_3^1 \\ \Delta V_3^1 \end{bmatrix} = \begin{bmatrix} -0.0096 \\ -0.0621 \\ -0.0163 \end{bmatrix}$$

Therefore,

$$\begin{aligned} \theta_2^1 &= \theta_2^0 + \Delta\theta_2^1 = 0 - 0.0096 = -0.0096 \\ \theta_3^1 &= \theta_3^0 + \Delta\theta_3^1 = 0 - 0.0621 = -0.0621 \\ V_3^1 &= V_3^0 + \Delta V_3^1 = 1 - 0.0163 = 0.9837 \end{aligned}$$

Note that the angles are given in *radians* and not degrees. The error at the first iteration is the largest absolute value of the mismatch equations, which is

$$\varepsilon^1 = 1.1802$$

One quick check of this process is to note that the voltage update V_3^1 is slightly less than 1.0 per unit, which would be expected given the system configuration. Note also that the diagonals of the Jacobian are all equal or greater in magnitude than the off-diagonal elements. This is because the diagonals are summations of terms, whereas the off-diagonal elements are single terms.

Iteration 2

Evaluating the Jacobian and the mismatch equations at the updated values θ_2^1, θ_3^1, and V_3^1 yields

$$[J^1] = \begin{bmatrix} -13.1597 & 9.7771 & 0.4684 \\ 9.6747 & -19.5280 & -0.7515 \\ -1.4845 & 3.0929 & -18.9086 \end{bmatrix}$$

$$\begin{bmatrix} \Delta P_2^1 \\ \Delta P_3^1 \\ \Delta Q_3^1 \end{bmatrix} = \begin{bmatrix} 0.0074 \\ -0.0232 \\ -0.0359 \end{bmatrix}$$

Solving for the update yields

$$\begin{bmatrix} \Delta\theta_2^2 \\ \Delta\theta_3^2 \\ \Delta V_3^2 \end{bmatrix} = \begin{bmatrix} -0.0005 \\ -0.0014 \\ -0.0021 \end{bmatrix}$$

and

$$\begin{bmatrix} \theta_2^2 \\ \theta_3^2 \\ V_3^2 \end{bmatrix} = \begin{bmatrix} -0.0101 \\ -0.0635 \\ 0.9816 \end{bmatrix}$$

where

$$\varepsilon^2 = 0.0359$$

Iteration 3

Evaluating the Jacobian and the mismatch equations at the updated values θ_2^2, θ_3^2, and V_3^2 yields

$$\left[J^2\right] = \begin{bmatrix} -13.1392 & 9.7567 & 0.4600 \\ 9.6530 & -19.4831 & -0.7213 \\ -1.4894 & 3.1079 & -18.8300 \end{bmatrix}$$

$$\begin{bmatrix} \Delta P_2^0 \\ \Delta P_3^0 \\ \Delta Q_3^0 \end{bmatrix} = \begin{bmatrix} 0.1717 \\ -0.5639 \\ -0.9084 \end{bmatrix} \times 10^{-4}$$

Solving for the update yields

$$\begin{bmatrix} \Delta\theta_2^2 \\ \Delta\theta_3^2 \\ \Delta V_3^2 \end{bmatrix} = \begin{bmatrix} -0.1396 \\ -0.3390 \\ -0.5273 \end{bmatrix} \times 10^{-5}$$

and

$$\begin{bmatrix} \theta_2^3 \\ \theta_3^3 \\ V_3^3 \end{bmatrix} = \begin{bmatrix} -0.0101 \\ -0.0635 \\ 0.9816 \end{bmatrix}$$

where

$$\varepsilon^3 = 0.9084 \times 10^{-4}$$

At this point, the iterations have converged, since the mismatch is sufficiently small and the values are no longer changing significantly.

The last task in power flow is to calculate the generated reactive powers, the slack bus active power output, and the line flows. The generated powers can be calculated directly from the power flow equations

$$P_i^{inj} = V_i \sum_{j=1}^{N_{bus}} V_j Y_{ij} \cos\left(\theta_i - \theta_j - \phi_{ij}\right)$$

$$Q_i^{inj} = V_i \sum_{j=1}^{N_{bus}} V_j Y_{ij} \sin\left(\theta_i - \theta_j - \phi_{ij}\right)$$

Therefore,

$$P_{gen,1} = P_1^{inj} = 0.7087$$
$$Q_{gen,1} = Q_1^{inj} = 0.2806$$
$$Q_{gen,2} = Q_2^{inj} = -0.0446$$

The active power losses in the system are the difference between the sum of the generation and the sum of the loads; in this case

$$P_{loss} = \sum P_{gen} - \sum P_{load} = 0.7087 + 0.5 - 1.2 = 0.0087\, pu \tag{3.117}$$

The line losses for line $i-j$ are calculated at both the sending and receiving ends of the line. Therefore, the power sent from bus i to bus j is

$$S_{ij} = V_i \angle \theta_i I_{ij}^* \tag{3.118}$$

and the power received at bus j from bus i is

$$S_{ji} = V_j \angle \theta_j I_{ji}^* \tag{3.119}$$

Thus

$$P_{ij} = V_i V_j Y_{ij} \cos(\theta_i - \theta_j - \phi_{ij}) - V_i^2 Y_{ij} \cos(\phi_{ij}) \tag{3.120}$$

$$Q_{ij} = V_i V_j Y_{ij} \sin(\theta_i - \theta_j - \phi_{ij}) + V_i^2 Y_{ij} \sin(\phi_{ij}) - V_i^2 B_{ij}/2 \tag{3.121}$$

Similarly, the powers P_{ji} and Q_{ji} can be calculated. The active power loss on any given line is the difference between the active power sent from bus i and the active power received at bus j. Calculating the reactive power losses is more complex, since the reactive power generated by the line-charging (shunt capacitances) must also be included. ■

3.5.2 Regulating Transformers

One of the most common controllers found in the power system network is the *regulating transformer*. This is a transformer that is able to change the winding ratios (tap settings) in response to changes in load-side voltage. If the voltage on the secondary side (or load side) is lower than a desired voltage (such as during heavy loading), the tap will change so as to increase the secondary voltage while maintaining the primary side voltage. A regulating transformer is also frequently referred to as an *under-load-tap-changing* or ULTC transformer. The tap setting t may be real or complex, and per unit, the tap ratio is defined as $1 : t$ where t is typically within 10% of 1.0. A phase-shifting transformer is achieved by allowing the tap t to be complex with both magnitude and angle.

The effect of the regulating transformer is incorporated into the power flow algorithm through the admittance matrix. To incorporate a regulating transformer into the admittance matrix, consider the regulating transformer as a two-port network relating the input currents I_i and I_j to the input voltages V_i and V_j, as shown in Figure 3.13. The receiving end current is given by

$$I_j = (V_j - tV_i)\, Y \tag{3.122}$$

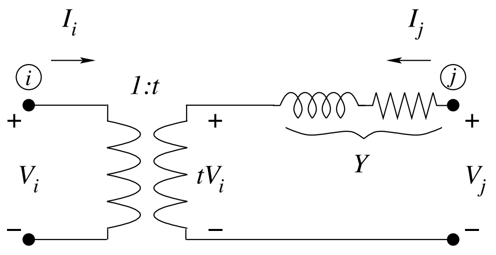


FIGURE 3.13
A regulating transformer

Note that the currents can be found from the power transfer equation

$$S_i = V_i I_i^* = -tV_i I_j^* \tag{3.123}$$

Therefore,

$$\begin{aligned} I_i &= -t^* I_j & (3.124)\\ &= -t^* \left(V_j - tV_i\right) Y & (3.125)\\ &= tt^* Y V_i - t^* Y V_j & (3.126)\\ &= \left|t\right|^2 Y V_i - t^* Y V_j & (3.127)\end{aligned}$$

Therefore, the off-diagonal entries in the admittance matrix become

$$\begin{aligned} Y(i,j) &= -t^* Y\\ Y(j,i) &= -tY \end{aligned}$$

and $\left|t\right|^2 Y$ is added to $Y(i,i)$ and Y is added to $Y(j,j)$.

Since regulating transformers are used as voltage control devices, a common computational exercise is to find the tap setting t that will hold the secondary bus voltage magnitude V_j at a specified voltage $\hat{V}$. This may be interpreted as adding one additional variable to the system (t) and one additional constraint $\left(V_j = \hat{V}\right)$. Since the additional constraint is counterbalanced by the additional degree of freedom, the dimension of the problem remains the same. There are two primary approaches for finding the tap setting t that results in $V_j = \hat{V}$. One approach is an iterative approach, while the second approach calculates t directly from the power flow equations.

The iterative approach may be summarized as

1. Set $t = t_0$
2. Run a power flow to calculate V_j

3. Is $V_j > \hat{V}$? If yes, then $t = t - \Delta t$, and go to step 2.
4. Is $V_j < \hat{V}$? If yes, then $t = t + \Delta t$, and go to step 2.
5. Done

This approach is conceptually simple and requires no changes to the power flow algorithm. However, it may require numerous runs of a power flow program if t_0 is far from the required tap setting.

The direct approach applies the Newton–Raphson method directly to the updated power flow equations as functions of the tap setting t.

1. Set $V_j = \hat{V}$ and let t be an unknown state
2. Modify the Newton–Raphson Jacobian such that the row of partial derivatives with respect to V_j is replaced by the row of partial derivatives with respect to t
3. Modify the state vector x such that

$$x = \begin{bmatrix} \theta_2 \\ \theta_3 \\ \vdots \\ \theta_n \\ V_2 \\ V_3 \\ \vdots \\ V_{j-1} \\ t \\ V_{j+1} \\ \vdots \\ V_n \end{bmatrix}$$

   Note that the state V_j is replaced by t.

4. Perform the Newton–Raphson

In this case, the set of power flow equations is solved only once, but since the system Jacobian is modified, a standard power flow program cannot be used.

Since the tap cannot move continuously along the transformer windings, but must move vertically from one winding to the adjacent winding, the real tap setting is not a continuous state. Therefore, in both cases, the calculated tap setting must be rounded to the nearest possible physical tap setting.

Example 3.12

For the system shown in Figure 3.12, place a transformer with reactance X and real tap t between bus 3 and the load (introduce a new bus 4). Find the new admittance matrix and the corresponding Jacobian entries.

Solution 3.12 Let the admittance matrix of the subsystem containing buses 1 through 3 be given by

$$Y_{bus} = \begin{bmatrix} Y_{11}\angle\phi_{11} & Y_{12}\angle\phi_{12} & Y_{13}\angle\phi_{13} \\ Y_{21}\angle\phi_{21} & Y_{22}\angle\phi_{22} & Y_{23}\angle\phi_{23} \\ Y_{31}\angle\phi_{31} & Y_{32}\angle\phi_{32} & Y_{33}\angle\phi_{33} \end{bmatrix} \tag{3.128}$$

Adding the transformer between buses 3 and 4 yields the new admittance matrix

$$Y_{bus} = \begin{bmatrix} Y_{11}\angle\phi_{11} & Y_{12}\angle\phi_{12} & Y_{13}\angle\phi_{13} & 0 \\ Y_{21}\angle\phi_{21} & Y_{22}\angle\phi_{22} & Y_{23}\angle\phi_{23} & 0 \\ Y_{31}\angle\phi_{31} & Y_{32}\angle\phi_{32} & Y_{33}\angle\phi_{33} + \frac{t^2}{jX} & \frac{-t}{jX} \\ 0 & 0 & \frac{-t}{jX} & \frac{1}{jX} \end{bmatrix} \tag{3.129}$$

The power flow equations at bus 3 become

$$\begin{aligned} 0 &= P_3 - V_3V_1Y_{31}\cos(\theta_3-\theta_1-\phi_{31}) - V_3V_2Y_{32}\cos(\theta_3-\theta_2-\phi_{32}) \\ &\quad -V_3V_4\left(\frac{t}{X}\right)\cos(\theta_3-\theta_4-90^\circ) - V_3^2Y_{33}\cos(-\phi_{33}) - V_3^2\left(\frac{t^2}{X}\right)\cos(90^\circ) \\ 0 &= Q_3 - V_3V_1Y_{31}\sin(\theta_3-\theta_1-\phi_{31}) - V_3V_2Y_{32}\sin(\theta_3-\theta_2-\phi_{32}) \\ &\quad -V_3V_4\left(\frac{t}{X}\right)\sin(\theta_3-\theta_4-90^\circ) - V_3^2Y_{33}\sin(-\phi_{33}) - V_3^2\left(\frac{t^2}{X}\right)\sin(90^\circ) \end{aligned}$$

Since V_4 is specified, there is no partial derivative $\frac{\partial \Delta P_3}{\partial V_4}$; instead there is a partial derivative with respect to t:

$$\frac{\partial \Delta P_3}{\partial t} = -\frac{V_3V_4}{X}\cos(\theta_3-\theta_4-90^\circ) \tag{3.130}$$

Similarly, the partial derivative of $\frac{\partial \Delta Q_3}{\partial t}$ becomes

$$\frac{\partial \Delta Q_3}{\partial t} = -\frac{V_3V_4}{X}\sin(\theta_3-\theta_4-90^\circ) + 2V_3^2\frac{t}{X} \tag{3.131}$$

The partial derivatives with respect to θ_1, θ_2, V_1, and V_2 do not change, but the partial derivatives with respect to θ_3, θ_4, and V_3 become

$$\begin{aligned} \frac{\partial \Delta P_3}{\partial \theta_3} &= V_3V_1Y_{31}\sin(\theta_3-\theta_1-\phi_{31}) + V_3V_2Y_{32}\sin(\theta_3-\theta_2-\phi_{32}) \\ &\quad +V_3V_4\frac{t}{X}\sin(\theta_3-\theta_4-90^\circ) \\ \frac{\partial \Delta P_3}{\theta_4} &= -V_3V_4\frac{t}{X}\sin(\theta_3-\theta_4-90^\circ) \\ \frac{\partial \Delta P_3}{\partial V_3} &= -V_1Y_{31}\cos(\theta_3-\theta_1-\phi_{31}) - V_2Y_{32}\cos(\theta_3-\theta_2-\phi_{32}) \\ &\quad V_4\frac{t}{X}\cos(\theta_3-\theta_4-90^\circ) - 2V_3Y_{33}\cos(-\phi_{33}) \end{aligned}$$

$$
\begin{aligned}
\frac{\partial \Delta Q_3}{\partial \theta_3} &= -V_3V_1Y_{31}\cos(\theta_3-\theta_1-\phi_{31}) - V_3V_2Y_{32}\cos(\theta_3-\theta_2-\phi_{32}) \\
&\quad -V_3V_4\frac{t}{X}\cos(\theta_3-\theta_4-90^\circ) \\
\frac{\partial \Delta Q_3}{\partial \theta_4} &= V_3V_4\frac{t}{X}\cos(\theta_3-\theta_4-90^\circ) \\
\frac{\partial \Delta Q_3}{\partial V_3} &= -V_1Y_{31}\sin(\theta_3-\theta_1-\phi_{31}) - V_2Y_{32}\sin(\theta_3-\theta_2-\phi_{32}) \\
&\quad -V_4\frac{t}{X}\sin(\theta_3-\theta_4-90^\circ) - 2V_3Y_{33}\sin(-\phi_{33}) - 2V_3\frac{t^2}{X}
\end{aligned}
$$

These partial derivatives are used in developing the Newton–Raphson Jacobian for the iterative power flow method. ■

3.5.3 Decoupled Power Flow

The power flow is one of the most widely used computational tools in power systems analysis. It can be successfully applied to problems ranging from a single machine system to a power system containing tens of thousands of buses. For very large systems, the full power flow may require significant computational resources to calculate, store, and factorize the Jacobian matrix. As discussed previously, however, it is possible to replace the Jacobian matrix with a matrix M that is easier to calculate and factor and still retain good convergence properties. The power flow equations naturally lend themselves to several alternate matrices for the power flow solution that can be derived from the formulation of the system Jacobian. Recall that the system Jacobian has the form

$$\begin{bmatrix} J_1 & J_2 \\ J_3 & J_4 \end{bmatrix} = \begin{bmatrix} \frac{\partial \Delta P}{\partial \theta} & \frac{\partial \Delta P}{\partial V} \\ \frac{\partial \Delta Q}{\partial \theta} & \frac{\partial \Delta Q}{\partial V} \end{bmatrix} \tag{3.132}$$

The general form of the P submatrices is

$$\frac{\partial \Delta P_i}{\partial \theta_j} = -V_iV_jY_{ij}\sin(\theta_i-\theta_j-\phi_{ij}) \tag{3.133}$$

$$\frac{\partial \Delta P_i}{\partial V_j} = V_iY_{ij}\cos(\theta_i-\theta_j-\phi_{ij}) \tag{3.134}$$

For most transmission lines, the line resistance contributes only nominally to the overall line impedance; thus, the phase angles ϕ_{ij} of the admittance matrix entries are near $\pm 90^\circ$. Additionally, under normal operating conditions the phase angle difference between adjacent buses is typically small; therefore,

$$\cos(\theta_i-\theta_j-\phi_{ij}) \approx 0 \tag{3.135}$$

leading to

$$\frac{\partial \Delta P_i}{\partial V_j} \approx 0 \tag{3.136}$$

Similar arguments can be made such that

$$\frac{\partial \Delta Q_i}{\partial \theta_j} \approx 0 \tag{3.137}$$

Using the approximations of Equations (3.136) and (3.137), a possible substitution for the Jacobian matrix is the matrix

$$M = \begin{bmatrix} \frac{\partial \Delta P}{\partial \theta} & 0 \\ 0 & \frac{\partial \Delta Q}{\partial V} \end{bmatrix} \tag{3.138}$$

Using this matrix M as a replacement for the system Jacobian leads to a set of *decoupled* iterates for the power flow solution:

$$\theta^{k+1} = \theta^k - \left[\frac{\partial \Delta P}{\partial \theta}\right]^{-1} \Delta P \tag{3.139}$$

$$V^{k+1} = V^k - \left[\frac{\partial \Delta Q}{\partial V}\right]^{-1} \Delta Q \tag{3.140}$$

where the ΔP and ΔQ iterations can be carried out independently. The primary advantage of this decoupled power flow is that the LU factorization computation is significantly reduced. The LU factorization of the full Jacobian requires $(2n)^3 = 8n^3$ floating point operations per iteration, whereas the decoupled power flow requires only $2n^3$ floating point operations per iteration.

Example 3.13
Repeat Example 3.11 using the decoupled power flow algorithm.

Solution 3.13 The Jacobian of Example 3.11 evaluated at the initial condition is

$$\left[J^0\right] = \begin{bmatrix} -13.2859 & 9.9010 & 0.9901 \\ 9.9010 & -20.0000 & -1.9604 \\ -0.9901 & 2.0000 & -19.4040 \end{bmatrix} \tag{3.141}$$

Note that the off-diagonal submatrices are much smaller in magnitude than the diagonal submatrices. For example,

$$\|[J_2]\| = \left\| \begin{bmatrix} 0.9901 \\ -1.9604 \end{bmatrix} \right\| << \|[J_1]\| = \left\| \begin{bmatrix} -13.2859 & 9.9010 \\ 9.9010 & -20.000 \end{bmatrix} \right\|$$

and

$$\|[J_3]\| = \left\| \begin{bmatrix} -0.9901 & 2.0000 \end{bmatrix} \right\| << \|[J_4]\| = \|[-19.4040]\|$$

Thus it is reasonable to neglect the off-diagonal matrices J_2 and J_3. Therefore, the first iteration of the decoupled power flow becomes

$$\begin{bmatrix} \Delta\theta_2^1 \\ \Delta\theta_3^1 \end{bmatrix} = [J_1]^{-1} \begin{bmatrix} \Delta P_2 \\ \Delta P_3 \end{bmatrix} \tag{3.142}$$

$$= \begin{bmatrix} -13.2859 & 9.9010 \\ 9.9010 & -20.000 \end{bmatrix}^{-1} \begin{bmatrix} 0.5044 \\ -1.1802 \end{bmatrix} \tag{3.143}$$

$$\left[\Delta V_3^1\right] = [J_4]^{-1} \Delta Q_3 \tag{3.144}$$

$$= -19.4040^{-1} (-0.2020) \tag{3.145}$$

leading to the updates

$$\begin{bmatrix} \theta_2^1 \\ \theta_3^1 \\ V_3^1 \end{bmatrix} = \begin{bmatrix} -0.0095 \\ -0.0637 \\ 0.9896 \end{bmatrix}$$

The iterative process continues similar to the full Newton–Raphson method by continually updating the J_1 and J_4 Jacobian submatrices and the mismatch equations. The iteration converges when both the ΔP mismatch equations and the ΔQ mismatch equations are both less than the convergence tolerance. Note that it is possible for one set of mismatch equations to meet the convergence criteria before the other; thus the number of "P" iterations required for convergence may differ from the number of "Q" iterations required for convergence. ■

3.5.4 Fast Decoupled Power Flow

In Example 3.13, each of the decoupled Jacobian submatrices is updated at every iteration. As discussed previously, however, it is often desirable to have constant matrices to minimize the number of function evaluations and LU factorizations. This is often referred to as the *fast decoupled power flow* and can be represented as

$$\left[\frac{\Delta P^k}{V}\right] = [B'] \left[\Delta\theta^{k+1}\right] \tag{3.146}$$

$$\left[\frac{\Delta Q^k}{V}\right] = [B''] \left[\Delta V^{k+1}\right] \tag{3.147}$$

where the B' and B'' are constant [53]. There are a number of variations that are typically denoted as the BB, XB, BX, and XX methods [1].

To derive these matrices from the power flow Jacobian, consider the decoupled power flow relationships for the Newton–Raphson method:

$$[\Delta P] = -[J_1][\Delta\theta] \tag{3.148}$$

$$\left[\frac{\Delta Q}{V}\right] = -[J_4][\Delta V] \tag{3.149}$$

where the Jacobian submatrices in rectangular form are

$$J_1(i,i) = V_i \sum_{j\neq i} V_j \left(g_{ij}\sin\theta_{ij} - b_{ij}\cos\theta_{ij}\right) \tag{3.150}$$

$$J_1(i,j) = -V_iV_j \left(g_{ij}\sin\theta_{ij} - b_{ij}\cos\theta_{ij}\right) \tag{3.151}$$

$$J_4(i,i) = 2V_ib_{ii} - \sum_{j\neq i} V_j \left(g_{ij}\sin\theta_{ij} - b_{ij}\cos\theta_{ij}\right) \tag{3.152}$$

$$J_4(i,j) = -V_i \left(g_{ij}\sin\theta_{ij} - b_{ij}\cos\theta_{ij}\right) \tag{3.153}$$

where $b_{ij} = |Y_{ij}\sin\phi_{ij}|$ are the imaginary elements of the admittance matrix and $g_{ij} = |Y_{ij}\cos\phi_{ij}|$ are the real elements of the admittance matrix.

By noting that $\phi_{ij} \approx 90°$, $\cos\phi_{ij} \approx 0$, which implies that $g_{ij} \approx 0$. By further approximating all voltage magnitudes as 1.0 per unit,

$$J_1(i,i) = -\sum_{j\neq i} b_{ij} \tag{3.154}$$

$$J_1(i,j) = b_{ij} \tag{3.155}$$

$$J_4(i,i) = 2b_{ii} - \sum_{j\neq i} b_{ij} \tag{3.156}$$

$$J_4(i,j) = b_{ij} \tag{3.157}$$

Since the J_1 submatrix relates the changes in active power to changes in angle, elements that affect mainly reactive power flow can be omitted from this matrix with negligible impact on the convergence properties. Thus shunt capacitors (including line-charging) and external reactances as well as the shunts formed due to representation of off-nominal non-phase-shifting transformers (i.e., taps are set to 1.0) are neglected. Hence, the admittance matrix diagonal elements are devoid of these shunts. Similarly, the J_4 submatrix relates the changes in reactive power to changes in voltage magnitude; therefore, elements that primarily affect active power flow are omitted. Thus all phase-shifting transformers are neglected.

Note that these approximations do not require that the line resistances R_{ij} be neglected, since

$$b_{ij} = \frac{-X_{ij}}{R_{ij}^2 + X_{ij}^2} \tag{3.158}$$

The different variations of the fast decoupled power flow arise primarily from how the line resistances are incorporated into the Jacobian approximations.

When all of the resistances are included, the approximation is known as the BB method. The approximation is seldom used since it usually suffers from poor convergence properties. Similarly, the XX method completely ignores the impact of the resistances. It has slightly better convergence than the BB method.

The most common method is the XB version of the fast decoupled power flow. In this method, the following B' and B'' matrices are defined:

$$B'_{ij} = \frac{1}{X_{ij}} \tag{3.159}$$

$$B'_{ii} = -\sum_{j \neq i} B'_{ij} \tag{3.160}$$

where B' is an approximation to the J_1 matrix.

B'' is an approximation to the J_4 matrix, where

$$B''_{ij} = b_{ij} \tag{3.161}$$

$$B''_{ii} = 2b_i - \sum_{j \neq i} B''_{ij} \tag{3.162}$$

and b_i is the shunt susceptance at bus i (i.e., the sum of susceptances of all the shunt branches connected to bus i).

This method results in a set of constant matrices that can be used to approximate the power flow Jacobian in the Newton–Raphson iteration. Both B' and B'' are real, sparse, and contain only network or admittance matrix elements. In the Newton–Raphson method, these matrices are only factorized once for the LU factorization, and are then stored and held constant throughout the iterative solution process. These matrices were derived based on the application of certain assumptions. If these assumptions do not hold (i.e., the voltage magnitudes deviate substantially from 1.0 per unit, the network has high R/X ratios, or the angle differences between adjacent buses are not small), then convergence problems with the fast decoupled power flow iterations can arise. Work still continues on developing modifications to the XB method to improve convergence [36], [37], [43]. The BX method is similar to the XB method, except that the resistances are ignored in the B'' matrix and not in the B' matrix.

Example 3.14

Repeat Example 3.11 using the fast decoupled power flow algorithm.

Solution 3.14 The line data for the example system are repeated below for convenience:

i	j	R_{ij}	X_{ij}	B_{ij}
1	2	0.02	0.3	0.15
1	3	0.01	0.1	0.1
2	3	0.01	0.1	0.1

and lead to the following admittance matrix:

$$Y_{bus} = \begin{bmatrix} 13.1505\angle-84.7148^\circ & 3.3260\angle 93.8141^\circ & 9.9504\angle 95.7106^\circ \\ 3.3260\angle 93.8141^\circ & 13.1505\angle-84.7148^\circ & 9.9504\angle 95.7106^\circ \\ 9.9504\angle 95.7106^\circ & 9.9504\angle 95.7106^\circ & 19.8012\angle-84.2606^\circ \end{bmatrix} \tag{3.163}$$

Taking the imaginary part of this matrix yields the following B matrix:

$$B = \begin{bmatrix} -13.0946 & 3.3186 & 9.9010 \\ 3.3186 & -13.0946 & 9.9010 \\ 9.9010 & 9.9010 & -19.7020 \end{bmatrix} \tag{3.164}$$

From the line data and the associated B matrix, the following B' and B'' matrices result:

$$B' = \begin{bmatrix} -\frac{1}{x_{21}} - \frac{1}{x_{23}} & \frac{1}{x_{23}} \\ \frac{1}{x_{23}} & -\frac{1}{x_{31}} - \frac{1}{x_{32}} \end{bmatrix} = \begin{bmatrix} -13.3333 & 10 \\ 10 & -20 \end{bmatrix} \tag{3.165}$$

$$\begin{aligned} B'' &= \left[2b_3 - (B_{31} + B_{32})\right] \\ &= \left[2\,(0.05 + 0.05) - (9.9010 + 9.9010)\right] = -19.6020 \end{aligned} \tag{3.166}$$

Compare these matrices to the J_1 and J_4 submatrices of Example 3.11 evaluated at the initial condition

$$J_1 = \begin{bmatrix} -13.2859 & 9.9010 \\ 9.9010 & -20.000 \end{bmatrix}$$
$$J_4 = [-19.4040]$$

The similarity between the matrices is to be expected as a result of the defining assumptions of the fast decoupled power flow method.

Iteration 1

The updates can be found by solving the linear set of equations

$$\begin{bmatrix} \Delta P_2^0 \\ \Delta P_3^0 \end{bmatrix} = \begin{bmatrix} 0.5044 \\ -1.1802 \end{bmatrix} = -\begin{bmatrix} -13.3333 & 10 \\ 10 & -20 \end{bmatrix} \begin{bmatrix} \Delta\theta_2^1 \\ \Delta\theta_3^1 \end{bmatrix}$$
$$\left[\Delta Q_3^0\right] = [-0.2020] = -19.6020 \Delta V_3^1$$

where $\Delta\theta_2^1 = \theta_2^{(1)} - \theta_2^{(0)}$, $\Delta\theta_3^1 = \theta_3^{(1)} - \theta_3^{(0)}$, and $\Delta V_3^1 = V_3^{(1)} - V_3^{(0)}$ and the initial conditions are a "flat start." Solving for the updates yields

$$\begin{bmatrix} \theta_2^1 \\ \theta_3^1 \\ V_3^1 \end{bmatrix} = \begin{bmatrix} -0.0103 \\ -0.0642 \\ 0.9897 \end{bmatrix}$$

where the phase angles are in radians. This process is continued until convergence in both the "P" and "Q" iterations is achieved. ■

Note that, in both the decoupled power flow cases, the objective of the iterations is the same as for the full Newton–Raphson power flow algorithm.

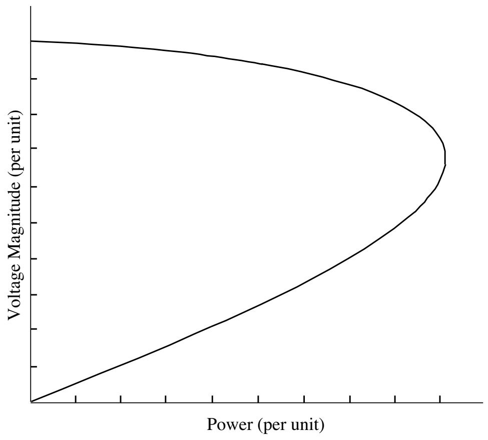


FIGURE 3.14
A PV curve

The objective is to drive the mismatch equations ΔP and ΔQ to within some tolerance. Therefore, regardless of the number of iterations required to achieve convergence, the accuracy of the answer is the same as for the full Newton–Raphson method. In other words, the voltages and angles of the decoupled power flow methods will be the same as with the full Newton–Raphson method as long as the iterates converge.

3.5.5 PV Curves and Continuation Power Flow

The power flow is a useful tool for monitoring system voltages as a function of load change. One common application is to plot the voltage at a particular bus as the load is varied from the base case to a loadability limit (often known as the point of maximum loadability). If the load is increased to the loadability limit and then decreased back to the original loading, it is possible to trace the entire power-voltage or "PV" curve. This curve, shown in Figure 3.14, is sometimes called the *nose curve* for its shape.

At the loadability limit, or tip of the nose curve, the system Jacobian of the power flow equations will become singular as the slope of the nose curve becomes infinite. Thus the traditional Newton–Raphson method of obtaining the power flow solution will break down. In this case, a modification of the Newton–Raphson method known as the *continuation method* is employed. The continuation method introduces an additional equation and unknown into the basic power flow equations. The additional equation is chosen specifically to ensure that the augmented Jacobian is no longer singular at the loadability limit. The additional unknown is often called the continuation parameter.

Continuation methods usually depend on a predictor-corrector scheme and the means to change the continuation parameter as necessary. The basic approach to tracing the PV curve is to choose a new value for the continuation parameter (either in power or voltage) and then predict the power flow solution for this value. This is frequently accomplished using a tangential (or linear) approximation. Using the predicted value as the initial condition for the nonlinear iteration, the augmented power flow equations are then solved (or corrected) to achieve the solution. So the solution is first predicted and then corrected. This prediction/correction step is shown in Figure 3.15.

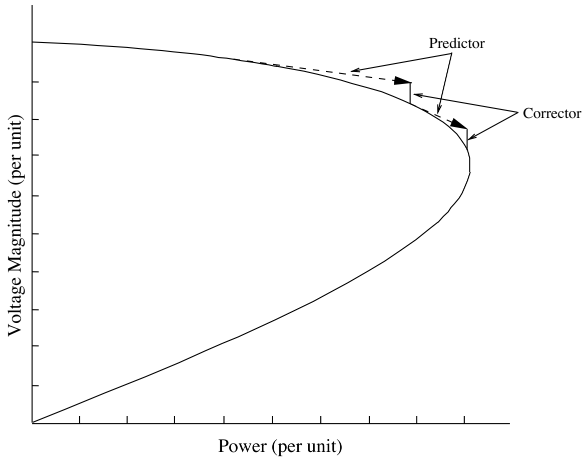


FIGURE 3.15
The predictor/corrector step

Let the set of power flow equations be given as

$$\lambda K - f(\theta, V) = 0 \tag{3.167}$$

or

$$F(\theta, V, \lambda) = 0 \tag{3.168}$$

where K is the loading profile (i.e., the base case relationship between P and Q) and λ is the loading parameter which will vary from unity (at the base case) to the point of maximum loadability. Equation (3.168) may be linearized to yield

$$\frac{\partial F}{\partial \theta} d\theta + \frac{\partial F}{\partial V} dV + \frac{\partial F}{\partial \lambda} d\lambda = 0 \tag{3.169}$$

Due to λ, the number of unknowns in Equation (3.169) is one larger than the number of equations, so one more equation is required:

$$e_k \begin{bmatrix} d\theta \\ dV \\ d\lambda \end{bmatrix} = 1 \tag{3.170}$$

where e_k is a row vector of zeros with a single ± 1 at the position of the unknown that is chosen to be the continuation parameter. The sign of the one in e_k is chosen based on whether the continuation parameter is increasing or decreasing. When the continuation parameter is λ (power), the sign is positive, indicating that the load is increasing. When voltage is the continuation parameter, the sign is negative, indicating that the voltage magnitude is expected to decrease toward the tip of the nose curve.

The unknowns are predicted such that

$$\begin{bmatrix} \theta \\ V \\ \lambda \end{bmatrix}^{\text{predicted}} = \begin{bmatrix} \theta_0 \\ V_0 \\ \lambda_0 \end{bmatrix} + \sigma \begin{bmatrix} d\theta \\ dV \\ d\lambda \end{bmatrix} \tag{3.171}$$

where

$$\begin{bmatrix} d\theta \\ \\ dV \\ \cdots \\ d\lambda \end{bmatrix} = \begin{bmatrix} & \vdots & \\ J_{LF} & \vdots & K \\ & \vdots & \\ \cdots\ \cdots & \cdots & \cdots \\ & [e_k] & \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix}$$

and σ is the step size (or length) for the next prediction. Note that the continuation state $dx_k = 1$; thus

$$x_k^{\text{predicted}} = x_{k0} + \sigma$$

so σ should be chosen to represent a reasonable step size in terms of what the continuation parameter is (usually voltage or power).

The corrector step involves the solution of the set of equations

$$F(\theta, V, \lambda) = 0 \tag{3.172}$$

$$x_k - x_k^{\text{predicted}} = 0 \tag{3.173}$$

where x_k is the chosen continuation parameter. Typically, the continuation parameter is chosen as the state that exhibits the greatest rate of change.

Example 3.15

Plot the PV curve (P versus V) of the system shown in Figure 3.16 using the continuation power flow method as the load varies from zero to the point of maximum loadability.

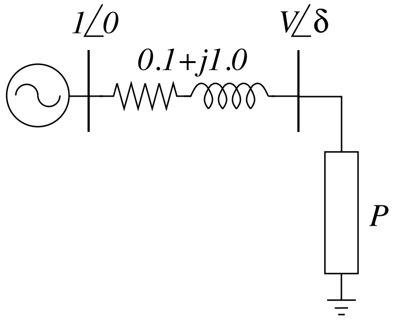


FIGURE 3.16
System for Example 3.15

Solution 3.15 The power flow equations for the system shown in Figure 3.16 are

$$0 = -P - 0.995V\cos(\theta - 95.7°) - 0.995V^2\cos(84.3°) \tag{3.174}$$
$$0 = -0.995V\sin(\theta - 95.7°) - 0.995V^2\sin(84.3°) \tag{3.175}$$

During the continuation power flow, the vector of injected active and reactive powers will be replaced by the vector λK. The loading vector λK is

$$\lambda K = \lambda \begin{bmatrix} -1 \\ 0 \end{bmatrix}$$

where λ will vary from zero to the maximum loading value. Typically, the vector K will contain the base case values for all injected active and reactive powers in the system. In this case, the entry for the load P is negative, indicating that the injected power is negative (i.e., a load).

The power flow Jacobian for this set of power flow equations is

$$J_{LF} = \begin{bmatrix} 0.995V\sin(\theta - 95.7°) & -0.995\cos(\theta - 95.7°) - 1.99V\cos(84.3°) \\ -0.995V\cos(\theta - 95.7°) & -0.995\sin(\theta - 95.7°) - 1.99V\sin(84.3°) \end{bmatrix}$$

Iteration 1

Initially, the continuation parameter is chosen to be λ since the load will change more rapidly than the voltage at points far from the tip of the nose curve. At $\lambda = 0$, the circuit is under no-load and the initial voltage magnitude and angle are $1\angle 0°$. With $\sigma = 0.1$ pu, the predictor step yields

$$\begin{bmatrix} \theta \\ V \\ \lambda \end{bmatrix}^{\text{predicted}} = \begin{bmatrix} \theta \\ V \\ \lambda \end{bmatrix}^{i} + \sigma \begin{bmatrix} & \vdots & \\ J_{LF} & \vdots & K \\ & \vdots & \\ \cdots \; \cdots & \cdots & \cdots \; \cdots \\ & [e_k] & \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix} \tag{3.176}$$

$$= \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} + \sigma \begin{bmatrix} -0.9901 & -0.0988 & -1 \\ 0.0988 & -0.9901 & 0 \\ 0 & 0 & 1 \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \tag{3.177}$$

$$= \begin{bmatrix} -0.1000 \\ 0.9900 \\ 0.1000 \end{bmatrix} \tag{3.178}$$

where θ is in radians. Note that the predicted value for λ is 0.1 pu.

The corrector step solves the system of equations

$$0 = -\lambda - 0.995V\cos(\theta - 95.7°) - 0.995V^2\cos(84.3°) \tag{3.179}$$

$$0 = -0.995V\sin(\theta - 95.7°) - 0.995V^2\sin(84.3°) \tag{3.180}$$

with the load parameter λ set to 0.1 pu. Note that this is a regular power flow problem and can be solved without program modification.

The first corrector step yields

$$\begin{bmatrix} \theta \\ V \\ \lambda \end{bmatrix} = \begin{bmatrix} -0.1017 \\ 0.9847 \\ 0.1000 \end{bmatrix}$$

Note that this procedure is consistent with the illustration in Figure 3.15. The prediction step is of length σ taken tangentially to the PV at the current point. The corrector step will then occur along a vertical path because the power (λK) is held constant during the correction.

Iteration 2

The second iteration proceeds as the first. The predictor step yields the following guess:

$$\begin{bmatrix} \theta \\ V \\ \lambda \end{bmatrix} = \begin{bmatrix} -0.2060 \\ 0.9637 \\ 0.2000 \end{bmatrix}$$

where λ is increased by the step size $\sigma = 0.1$ pu.

Correcting the values yields the second update:

$$\begin{bmatrix} \theta \\ V \\ \lambda \end{bmatrix} = \begin{bmatrix} -0.2105 \\ 0.9570 \\ 0.2000 \end{bmatrix}$$

Iterations 3 and 4
The third and fourth iterations progress similarly. The values to this point are summarized:

λ	V	θ	σ
0.1000	0.9847	−0.1017	0.1000
0.2000	0.9570	−0.2105	0.1000
0.3000	0.9113	−0.3354	0.1000
0.4000	0.8268	−0.5050	0.1000

Beyond this point, the power flow fails to converge for a step size of $\sigma = 0.1$. The method is nearing the point of maximum power flow (the tip of the nose curve), as indicated by the rapid decline in voltage for relatively small changes in λ. At this point, the continuation parameter is switched from λ to V to ensure that the corrector step will converge. The predictor step is modified such that

$$\begin{bmatrix} d\theta \\ dV \\ d\lambda \end{bmatrix} = \begin{bmatrix} d\theta_0 \\ dV_0 \\ d\lambda_0 \end{bmatrix} + \sigma \begin{bmatrix} \multicolumn{2}{c}{[J_{LF}]} & K \\ & & 0 \\ 0 & -1 & 0 \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

where the -1 in the last row (the e_k vector) now corresponds to V rather than λ. The step size σ is changed to 0.025 pu because changes in voltage are smaller than power (λ). Note that the minus sign will ensure that the voltage magnitude will *decrease* by 0.025 pu at every step.

The corrector step is also modified when the continuation parameter switches to voltage magnitude. The new augmented equations become:

$$0 = f_1(\theta, V, \lambda) = -\lambda - 0.995V\left(\cos\left(\theta - 95.7^\circ\right) + V\cos(84.3^\circ)\right) \quad (3.181)$$
$$0 = f_2(\theta, V, \lambda) = -0.995V\sin\left(\theta - 95.7^\circ\right) - 0.995V^2\sin(84.3^\circ) \quad (3.182)$$
$$0 = f_3(\theta, V, \lambda) = V^{\text{predicted}} - V \quad (3.183)$$

which cannot be solved with a traditional power flow program due to the last equation. This equation is necessary to keep the Newton–Raphson iteration nonsingular. Fortunately, the Newton–Raphson iteration uses the same iteration matrix as the predictor matrix:

$$\begin{bmatrix} \multicolumn{2}{c}{[J_{LF}]} & -\lambda \\ & & 0 \\ 0 & -1 & 0 \end{bmatrix}^{-1} \left(\begin{bmatrix} \theta \\ V \\ \lambda \end{bmatrix}^{(k+1)} - \begin{bmatrix} \theta \\ V \\ \lambda \end{bmatrix}^{(k)} \right) = - \begin{bmatrix} f_1 \\ f_2 \\ f_3 \end{bmatrix} \quad (3.184)$$

thus minimizing the computational requirement.

Note that the corrector step is now a horizontal correction in voltage. The voltage magnitude is held constant while λ and θ are corrected. These iterates proceed as

Predicted				Corrected		
λ	V	θ	σ	λ	V	θ
0.4196	0.8019	−0.5487	0.0250	0.4174	0.8019	−0.5474
0.4326	0.7769	−0.5887	0.0250	0.4307	0.7769	−0.5876
0.4422	0.7519	−0.6268	0.0250	0.4405	0.7519	−0.6260
0.4487	0.7269	−0.6635	0.0250	0.4472	0.7269	−0.6627
0.4525	0.7019	−0.6987	0.0250	0.4511	0.7019	−0.6981
0.4538	0.6769	−0.7328	0.0250	0.4525	0.6769	−0.7323
0.4528	0.6519	−0.7659	0.0250	0.4517	0.6519	−0.7654

Note that, at the last updates, the load parameter λ has started to decrease with decreasing voltage. This indicates that the continuation power flow is now starting to map out the lower half of the nose curve. However, while the iterations are still close to the tip of the nose curve, the Jacobian will still be ill conditioned, so it is a good idea to take several more steps before switching the continuation parameter from voltage magnitude back to λ.

Predicted				Corrected		
λ	V	θ	σ	λ	V	θ
0.4497	0.6269	−0.7981	0.0250	0.4487	0.6269	−0.7977
0.4447	0.6019	−0.8295	0.0250	0.4438	0.6019	−0.8291
0.4380	0.5769	−0.8602	0.0250	0.4371	0.5769	−0.8598
0.4296	0.5519	−0.8902	0.0250	0.4288	0.5519	−0.8899
0.4197	0.5269	−0.9197	0.0250	0.4190	0.5269	−0.9194
0.4084	0.5018	−0.9486	0.0250	0.4077	0.5018	−0.9483
0.3958	0.4768	−0.9770	0.0250	0.3951	0.4768	−0.9768
0.3820	0.4518	−1.0051	0.0250	0.3814	0.4518	−1.0049
0.3670	0.4268	−1.0327	0.0250	0.3665	0.4268	−1.0325
0.3510	0.4018	−1.0600	0.0250	0.3505	0.4018	−1.0598
0.3340	0.3768	−1.0870	0.0250	0.3335	0.3768	−1.0868

After switching the continuation parameter back to λ, the e_k vector becomes

$$e_k = [0 \quad 0 \quad -1]$$

and σ is set to 0.1 pu which indicates that λ will decrease by 0.1 pu at every step (i.e., the power is decreasing back to the base case). The predictor/corrector steps proceed as above, yielding

Predicted				Corrected		
λ	V	θ	σ	λ	V	θ
0.2335	0.2333	−1.2408	0.1000	0.2335	0.2486	−1.2212
0.1335	0.1312	−1.3417	0.1000	0.1335	0.1374	−1.3340
0.0335	0.0309	−1.4409	0.1000	0.0335	0.0338	−1.4375

These values are combined in the PV curve shown in Figure 3.17. Note the change in step size when the continuation parameter switches from λ to voltage near the tip of the PV curve. The appropriate choice of step size is problem dependent and can be adaptively changed to improve computational efficiency. ■

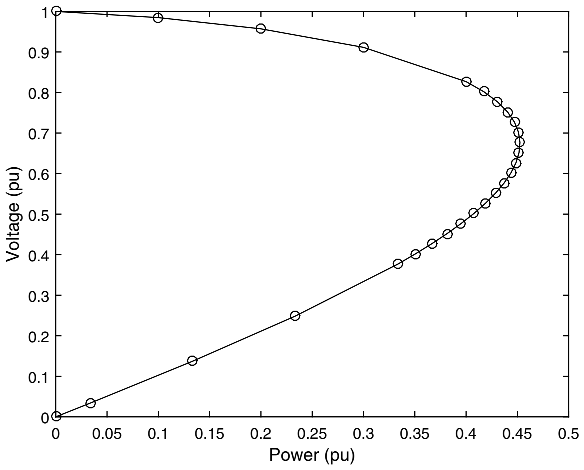


FIGURE 3.17
PV curve for system of Example 3.15

3.5.6 Three-Phase Power Flow

Another special-purpose power flow application is for three-phase power flow analysis. Although much of the power system analysis is performed on balanced three-phase systems using one line equivalent diagrams, certain situations call for a three-phase analysis. In particular, in the situation where the transmission system is not balanced due to nontransposed lines or when the loads are considerably unbalanced, it may be desirable to perform a full three-phase power flow to ascertain the effect on the individual line flows and bus voltages. The development of the three-phase power flow is similar to that of the single-phase equivalent, except that the mutual coupling between lines must be incorporated into the admittance matrix, yielding a $3n \times 3n$ matrix with elements Y_{ij}^{pq} where the subscript ij indicates bus number $(1 \leq (i,j) \leq n)$ and the superscript pq indicates phase $(p, q \in [a,b,c])$.

The incorporation of each phase individually leads to the similar, but slightly more complex, three-phase power flow equations

$$0 = \Delta P_i^p = P_i^{inj,p} - V_i^p \sum_{q \in (a,b,c)} \sum_{j=1}^{N_{bus}} V_j^q Y_{ij}^{pq} \cos\left(\theta_i^p - \theta_j^q - \phi_{ij}^{pq}\right) \quad (3.185)$$

$$0 = \Delta Q_i^p = Q_i^{inj,p} - V_i^p \sum_{q\in(a,b,c)} \sum_{j=1}^{N_{bus}} V_j^q Y_{ij}^{pq} \sin\left(\theta_i^p - \theta_j^q - \phi_{ij}^{pq}\right) \quad (3.186)$$
$$i = 1,\ldots,N_{bus} \text{ and } p \in (a,b,c)$$

There are three times as many power flow equations as in the single-phase equivalent power flow equations. Generator (PV) buses are handled similarly with the following exceptions:

1. $\theta^a = 0°, \theta^b = -120°, \theta^c = 120°$ for the slack bus
2. All generator voltage magnitudes and active powers in each phase must be equal since generators are designed to produce balanced output

A three-phase power flow "flat start" is to set each bus voltage magnitude to

$$V_i^a = 1.0\angle 0°$$
$$V_i^b = 1.0\angle -120°$$
$$V_i^c = 1.0\angle 120°$$

The system Jacobian used in the Newton–Raphson solution of the power flow equations will have a possible $(3(2n)\times 3(2n))$ or $36n^2$ entries. The Jacobian partial derivatives are found in the same manner as with the single-phase equivalent system except that the derivatives must also be taken with respect to phase differences. For example,

$$\frac{\partial \Delta P_i^a}{\partial \theta_j^b} = V_i^a V_j^b Y_{ij}^{ab} \sin\left(\theta_i^a - \theta_j^b - \phi_{ij}^{ab}\right) \quad (3.187)$$

which is similar to the single-phase equivalent system. Similarly,

$$\frac{\partial \Delta P_i^a}{\partial \theta_i^a} = -V_i^a \sum_{q\in(a,b,c)} \sum_{j=1}^{N_{bus}} V_j^q Y_{ij}^{pq} \sin\left(\theta_i^p - \theta_j^q - \phi_{ij}^{pq}\right) + \left(V_i^a\right)^2 Y_{ii}^{pp} \cos\left(\phi_{ii}^{pp}\right) \quad (3.188)$$

The remaining partial derivatives can be calculated in a similar manner and the solution process of the three-phase power flow follows the method outlined in Section 3.5.1.

3.6 Problems

1. Prove that the Newton–Raphson iteration will diverge for the following functions regardless of choice of initial condition

   (a) $f(x) = x^2 + 1$

(b) $f(x) = 7x^4 + 3x^2 + \pi$

2. Devise an iteration formula for computing the fifth root of any positive real number.

3. Using the Newton–Raphson method, solve

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

with $[x^0\ y^0]^T = [1\ 1]^T$.

4. Using the Newton–Raphson method, solve

$$\begin{aligned} 0 &= x - 2y + y^2 + y^3 - 4 \\ 0 &= -xy + 2y^2 - 1 \end{aligned}$$

with $[x^0\ y^0]^T = [1\ 1]^T$.

5. Repeat Problems 3 and 4 using numerical differentiation to compute the Jacobian. Use a perturbation of 1% in each variable.

6. Repeat Problems 3 and 4 using Broyden's method.

7. Repeat Problems 3 and 4 using the Jacobian-Free Newton–GMRES method.

8. Repeat Problems 3 and 4 using homotopic mapping with $0 = f_{01} = x_1 - 2$ and $0 = f_{02} = x_2 - 4$. Use $(2, 4)$ as the initial guess and a $\Delta\lambda$ of 0.05.

9. Write a *generalized* (for any system) power flow program. Your program should

   (a) Read in load, voltage, and generation data. You may assume that bus #1 corresponds to the slack bus.

   (b) Read in line and transformer data and create the Y_{bus} matrix.

   (c) Solve the power flow equations using the Newton–Raphson algorithm, for a stopping criterion of $\left\| f\left(x^k\right) \right\| \leq \epsilon = 0.0005$.

   (d) Calculate all dependent unknowns, line flows, and line losses.

   The Newton–Raphson portion of the program should call the lufact and permute subroutines. Your program should give you the option of using either a "flat start" or "previous values" as an initial guess. The easiest way to accomplish this is to read and write to the same data file. Note that the first run must be a "flat start" case.

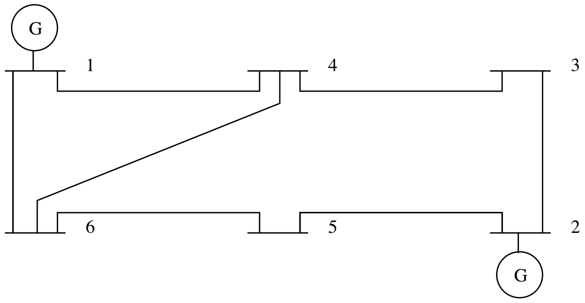


FIGURE 3.18
Ward–Hale six-bus system

10. The data for the system shown in Figure 3.18 are given below:

No.	Type	$\|V\|$	θ	P_{gen}	Q_{gen}	P_{load}	Q_{load}
1	0	1.05	0	0	0	0.25	0.1
2	1	1.05	0	0.5	0	0.15	0.05
3	2	1.00	0	0	0	0.275	0.11
4	2	1.00	0	0	0	0	0
5	2	1.00	0	0	0	0.15	0.09
6	2	1.00	0	0	0	0.25	0.15

No.	To	From	R	X	B
1	1	4	0.020	0.185	0.009
2	1	6	0.031	0.259	0.010
3	2	3	0.006	0.025	0.000
4	2	5	0.071	0.320	0.015
5	4	6	0.024	0.204	0.010
6	3	4	0.075	0.067	0.000
7	5	6	0.025	0.150	0.017

Calculate the power flow solution for the system data given above. Remember to calculate all dependent unknowns, line flows, and line losses.

11. Modify your power flow program so that you are using a *decoupled power flow* (i.e., assume $\left[\frac{\partial \Delta P}{\partial V}\right] = 0$ and $\left[\frac{\partial \Delta Q}{\partial \theta}\right] = 0$). Repeat Problem 10. Discuss the difference in convergence between the decoupled and the full Newton–Raphson power flows.

12. Increase the line resistances by 75% (i.e., multiply all resistances by 1.75) and repeat Problem 10 and Problem 11. Discuss your findings.

13. Using a continuation power flow, map out the "PV" curve for the original system data by increasing/decreasing the load on bus 6, holding a constant P/Q ratio from $P = 0$ to the point of maximum power transfer.

14. Making the following assumptions, find a **constant, decoupled** Jacobian that could be used in a fast, decoupled three-phase power flow.

    - $V_i^p \approx 1.0$ pu for all i and p
    - $\theta_{ij}^{pp} \approx 0$
    - $\theta_{ij}^{pm} \approx \pm 120^\circ \;\; p \neq m$
    - $g_{ij}^{pm} << b_{ij}^{pm}$