5 Nonlinear Equations

Concepts of economic equilibrium are often expressed as systems of nonlinear equations. Such problems generally take two forms, zeros and fixed points. If $f : R^n \rightarrow R^n$, then a *zero of f* is any x such that $f(x) = 0$, and a *fixed point* of f is any x such that $f(x) = x$. These are essentially the same problem, since x is a fixed point of $f(x)$ if and only if it is a zero of $f(x) - x$. The existence of solutions is examined in Brouwer's theorem and its extensions.[1] In this chapter we examine numerical methods for solving nonlinear equations.

The Arrow-Debreu concept of general equilibrium reduces to finding a price vector at which excess demand is zero; it is the most famous nonlinear equation problem in economics. Other examples include Nash equilibria of games with continuous strategies and the transition paths of deterministic dynamic systems. More recently, economists have solved nonlinear dynamic problems in infinite-dimensional spaces by approximating them with nonlinear equations in finite-dimensional Euclidean spaces. Nonlinear equation solving is a central component in many numerical procedures in economics.

In this chapter we examine a wide range of methods for solving nonlinear equations. We first discuss techniques for one-dimensional problems. We then turn to several methods for solving general finite-dimensional problems, including Gauss-Jacobi, Gauss-Seidel, successive approximation, Newton's method, and homotopy methods. These methods offer a variety of approaches, each with its own strengths and weaknesses. We apply these methods to problems from static general equilibrium and oligopoly theory.

5.1 One-Dimensional Problems: Bisection

We first consider the case, $f : R \rightarrow R$, of a single variable and the problem $f(x) = 0$. This special case is an important one; it is also the basis of some multidimensional routines. We use this case to simply exposit ideas that generalize to n dimensions.

Bisection

The first method we examine is bisection, which is displayed in figure 5.1. Suppose f is continuous and $f(a) < 0 < f(b)$ for some a, b, $a < b$. Under these conditions, the intermediate value theorem (IVT) tells us that there is some zero of f in (a, b). The bisection method uses this result repeatedly to compute a zero.

Consider $c = \frac{1}{2}(a + b)$, the midpoint of $[a, b]$. If $f(c) = 0$, we are done. If $f(c) < 0$, the IVT says that there is a zero of f in (c, b). The bisection method then continues

[1] For a review of fixed-point theory, see Border (1985).

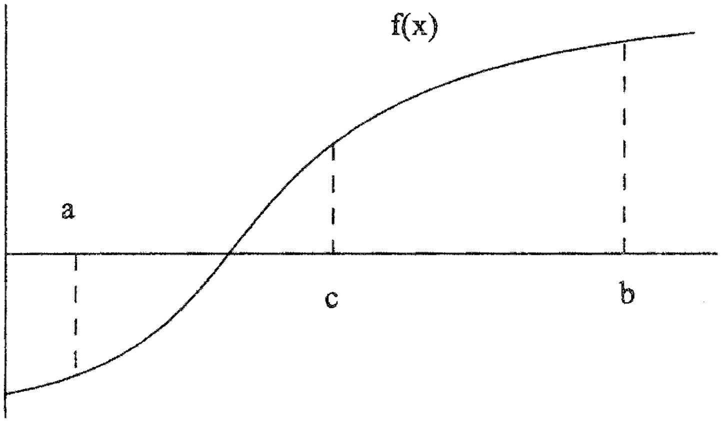

Figure 5.1
Bisection method

by focusing on the interval (c, b). If $f(c) > 0$, as in figure 5.1, there is a zero in (a, c), and the bisection method continues with (a, c). In either case, the bisection method continues with a smaller interval. There could be zeros in both (a, c) and (b, c), but our objective here is to find some zero, not all zeros. The bisection method applies this procedure repeatedly, constructing successively smaller intervals containing a zero.

Algorithm 5.1 Bisection Method

Objective: Find a zero of $f(x), f : R^1 \to R^1$.

Initialization. Initialize and bracket a zero: Find $x^L < x^R$ such that $f(x^L) \cdot f(x^R) < 0$, and choose stopping rule parameters $\varepsilon, \delta > 0$.

Step 1. Compute midpoint: $x^M = (x^L + x^R)/2$.

Step 2. Refine the bounds: If $f(x^M) \cdot f(x^L) < 0$, $x^R = x^M$ and do not change x^L; else $x^L = x^M$ and leave x^R unchanged.

Step 3. Check stopping rule: If $x^R - x^L \leq \varepsilon(1 + |x^L| + |x^R|)$ or if $|f(x^M)| \leq \delta$, then STOP and report solution at x^M; else go to step 1.

While the bisection algorithm is simple, it displays the important components of any nonlinear equation solver: Make an initial guess, compute iterates, and check if the last iterate is acceptable as a solution. We next address the stopping criterion of the bisection method.

Stopping Rules

Iterative schemes seldom land on the true solution. Furthermore, continuing the iteration is eventually of no value since round-off error in computing $f(x)$ will eventually dominate the differences between successive iterates. Therefore a critical component of any nonlinear equation solver is the stopping rule. A stopping rule generally computes some estimate of the distance to a solution, and then stops the iteration when that estimate is small. In the case of the bisection method, it is clear that $x^R - x^L$ is an overestimate of the distance to a solution since the zero is bracketed. Step 3 says to stop in either of two cases. First, we can stop whenever the bracketing interval is so small that we do not care about any further precision. This is controlled by the choice of ε, which need not be small. For example, if the problem is finding the equilibrium price for a market, there is little point in being more precise than finding the solution to within a penny if the equilibrium price is on the order of \$1,000.

If we want high precision, we will choose small ε, but that choice must be reasonable. Choosing $\varepsilon = 0$ is nonsense, since it is unachievable; equally pointless is choosing $\varepsilon = 10^{-20}$ on a 12-digit machine where f can be calculated with at most 12 digits of accuracy. Note also that we use a relative step size condition in step 3 of the algorithm 5.1 for stopping because the computer can really detect only relatively small intervals. If x^L and x^R are of the order 10^{10}, demanding $x^R - x^L < 10^{-5}$ would be the same as choosing $\varepsilon = 0$ on a machine that has only 12-digit precision. This consideration alone would argue for a purely relative change test of the form $x^R - x^L < \varepsilon(|x^L| + |x^R|)$. Such a test would have problems in cases where the solution is close to $x = 0$ and x^L and x^R converges to 0. The "1" in the stopping criterion in step 3 avoids this problem.

Second, we must stop when round-off errors associated with computing f make it impossible to pin down the zero any further; that is, we stop when $f(x^M)$ is less than the expected error in calculating f. This is controlled in step 3 by δ; therefore δ should be at least the error expected in the computation of f. In some cases the computation of f may be rather complex and subject to a variety of numerical errors so that δ may be much larger than the machine precision. In other cases we may be satisfied with an x that makes $f(x)$ small but not zero. In those cases we should choose δ to be the maximal value for $f(x)$ that serves our purposes.

Convergence

If an iterative method is to be useful, it must converge to the solution and do so at a reasonable speed. Convergence properties for bisection are obvious. Bisection will always converge to a solution once we have found an initial pair of points, x^L and x^R, that bracket a zero.

Being essentially a linearly convergent iteration, bisection is a slow method. Suppose that you want to increase the accuracy by one significant decimal digit; that is, reduce the maximum possible error, $|x_k^L - x_k^R|$, by 90 percent. Since each iteration of bisection reduces the error by only 50 percent, it takes more than three iterations to add a decimal digit of accuracy.

5.2 One-Dimensional Problems: Newton's Method

While bisection is a reliable procedure, its slow speed makes it a poor method for solving one-dimensional equations. It is also a method that assumes only continuity of f. Newton's method uses smoothness properties of f to formulate a method that is fast when it works but may not always converge. Newton's method approximates $f(x)$ by a succession of linear functions, and it approximates a zero of f with the zeros of the linear approximations. If f is smooth, these approximations are increasingly accurate, and the successive iterates will converge rapidly to a zero of f. In essence Newton's method reduces a nonlinear problem to a sequence of linear problems, where the zeros are easy to compute.

Formally Newton's method is a simple iteration. Suppose that our current guess is x_k. At x_k we construct the linear approximation to f at x_k, yielding the function $g(x) \equiv f'(x_k)(x - x_k) + f(x_k)$. The functions $g(x)$ and $f(x)$ are tangent at x_k, and generally close in the neighborhood of x_k. Instead of solving for a zero of f, we solve for a zero of g, hoping that the two functions have similar zeros. Our new guess, x_{k+1}, will be the zero of $g(x)$, implying that

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}. \tag{5.2.1}$$

Figure 5.2 graphically displays the steps from x_1 to x_5. The new guess will likely not be a zero of f, but the hope is that the sequence x_k will converge to a zero of f. Theorem 2.1 provides a sufficient condition for convergence.

THEOREM 2.1 Suppose that f is C^2 and that $f(x^*) = 0$. If x_0 is sufficiently close to x^*, $f'(x^*) \neq 0$, and $|f''(x^*)/f'(x^*))| < \infty$, the Newton sequence x_k defined by (5.2.1) converges to x^*, and it is quadratically convergent, that is,

$$\limsup_{k\to\infty} \frac{|x_{k+1} - x^*|}{|x_k - x^*|^2} < \infty. \tag{5.2.2}$$

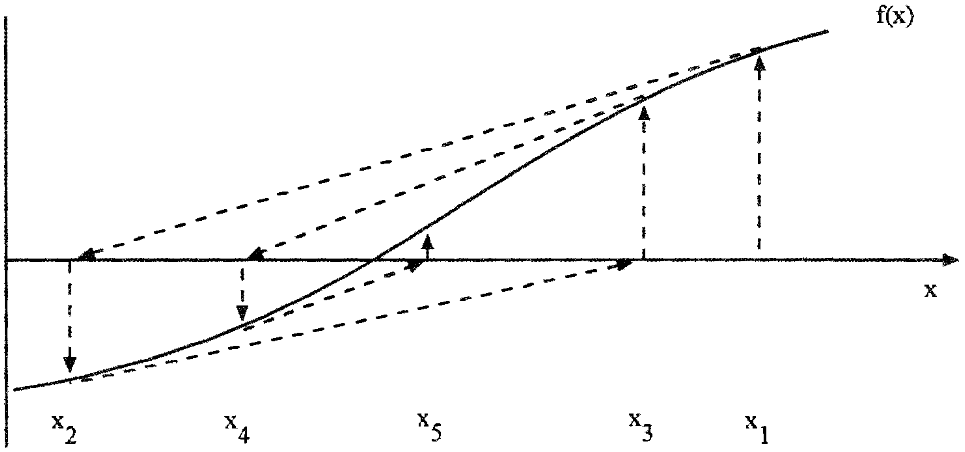

Figure 5.2
Newton method

Proof The proof of theorem 4.1.1 for Newton's method for optimization was really a proof that Newton's method for nonlinear equations converged when applied to the first-order conditions of an optimization problem. That proof applies here without change. In particular,

$$\lim_{k\to\infty}\frac{|x_{k+1}-x^*|}{|x_k-x^*|^2}=\frac{1}{2}\frac{|f''(x^*)|}{|f'(x^*)|} \tag{5.2.3}$$

which is finite if $f'(x^*)\neq 0$. ■

We see the usual trade-off when we compare Newton's method with bisection. Bisection is a safe method, always converging to a zero; unfortunately, it is slow. Newton's method sacrifices reliability for a substantial gain in speed if it converges. In choosing between the two methods, one must judge the relative importance of these features and the likelihood of Newton's method not converging.

Convergence Tests

Equation (5.2.1) is just the iterative portion of Newton's method. To complete the algorithm, we must append a convergence test, particularly since Newton's method does not bracket a zero. Typically an implementation of Newton's method applies a two-stage test in the spirit of the convergence tests discussed in chapter 2. First, we ask if the last few iterations have moved much. Formally we specify an ε and conclude that we have converged if $|x_k - x_{k-l}| < \varepsilon(1 + |x_k|)$ for $l = 1, 2, \ldots, L$; often we take $L = 1$.

Second, we ask if $f(x_k)$ is "nearly" zero. More precisely we stop if $|f(x_k)| \leq \delta$ for some prespecified δ. The tightness of this criterion is governed by the choice of δ, and the considerations here are the same as the choice of δ in bisection method. The full Newton algorithm is presented in algorithm 5.2.

Algorithm 5.2 Newton's Method

Objective: Find a zero of $f(x)$, $f : R^1 \to R^1$, $f \in C^1$.

Initialization. Choose stopping criterion ε and δ, and starting point x_0. Set $k = 0$.

Step 1. Compute next iterate: $x_{k+1} = x_k - f(x_k)/f'(x_k)$.

Step 2. Check stopping criterion: If $|x_k - x_{k+1}| \leq \varepsilon(1 + |x_{k+1}|)$, go to step 3. Otherwise, go to step 1.

Step 3. Report results and STOP: If $|f(x_{k+1})| \leq \delta$, report success in finding a zero; otherwise, report failure.

Even if we have found a point that satisfies both the ε and δ tests, it may still not be a zero, or close to one. Consider the case of $f(x) = x^6$. Appying step 1 to x^6 implies that $x_{k+1} = \frac{5}{6}x_k$, which is a slow, linearly convergent iteration. The problem is that x^6 is flat at its zero. More generally, if a function is nearly flat at a zero, convergence can be quite slow, and loose stopping rules may stop far from the true zero.

These stopping rule issues are more important in the case of nonlinear equations than in optimization problems. In optimization problems one is often satisfied with some point x that is nearly as good as the true optimum x^* in optimizing the objective, $F(x)$. In such cases one does not care if the error $x - x^*$ is large. We are often more demanding when solving nonlinear equations, requiring the error $x - x^*$ to be small. Therefore stopping rule issues are more important.

Importance of Units

At this point we turn to an important consideration in constructing equations and implementing the stopping criterion in step 3. In step 3 we declare success as long as we have found some x such that $f(x)$ is less than δ in magnitude. This is actually an empty test, for we can always take $f(x)$, multiply it by 10^{-k} for some large k, and have a new function with the same zeros. We may now find many x such that $10^{-k}f(x) < \delta$. While we would not do this intentionally, we could easily do this accidentally.

The general point is that we need to keep track of the units we are implicitly using, and formulate the problem and stopping rules in unit-free ways. Step 3 must be carefully formulated for it to be meaningful. For example, in computing the equilibrium price, p, of a good, let $f(p) = D(p) - S(p)$ be excess demand and assume that D and S can be computed with double precision accuracy. In this case the stopping rule in step 3 could be reformulated as $D(p) - S(p) < \delta D(p)$ with $\delta = 10^{-8}$. This stopping rule avoids all unit problems; it stops when excess demand is small relative to total demand. Simple adjustments to step 3 like this are ideal because they avoid unit problems without affecting the algorithm and can often be implemented.

Pathological Examples

Newton's method works well when it works, but it can fail. Consider the case of $f(x) = x^{1/3}e^{-x^2}$. The unique zero of f is at $x = 0$. However, Newton's method produces (see Donovan et al. 1993 for details) the iteration

$$x_{n+1} = x_n\left(1 - \frac{3}{1 - 6x_n^2}\right) \tag{5.2.4}$$

which has two pathologies. First, for x_n small, (5.2.4) reduces to $x_{n+1} = -2x_n$, showing that (5.2.4) converges to 0 only if $x_0 = 0$ is the initial guess. For large x_n, (5.2.4) becomes $x_{n+1} = x_n(1 + 2/x_n^2)$, which diverges but slowly. In fact, for any ε, $\delta > 0$, there will ultimately be large x_n and x_{n+1} such that $|f(x_{n+1})| < \delta$ and $|x_{n+1} - x_n| < \varepsilon$; reducing δ and ε will cause Newton's method to stop at a value even farther from the true zero.

What goes wrong? The divergence of Newton's method is due to the infinite value of $f''(0)/f'(0)$. The second problem arises because the e^{-x^2} factor squashes the function at large x, leading Newton's method, (5.2.4), to "think" that it is getting close to a zero; in some sense it is, since $f(\pm\infty) = 0$.

Newton's method can also converge to a cycle. Figure 5.3 illustrates how easy it is to construct such as case. To get Newton's method to cycle between, say, -1 and 1, we just need $f'(1) = 0.5 = f'(-1)$ and $f(1) = 1 = -f(-1)$. While these examples are contrived, their lessons should not be ignored. In particular one should be sure that $f'(x^*) \neq 0$ and $f''(x^*)$ are not too large. While this may seem trite, it is easy to write examples where some of the derivatives at the solution are very large, and the radius of convergence is small. In the less pathological case of figure 5.3, Newton's method will converge if we begin with $x \in [-0.5, 0.5]$, showing the importance of a good initial guess.

A General Equilibrium Example with the Limited Domain Problem

Our discussion has implicitly assumed that f is defined at each Newton iterate; this is not true in many economic examples. We have also not considered how multiple

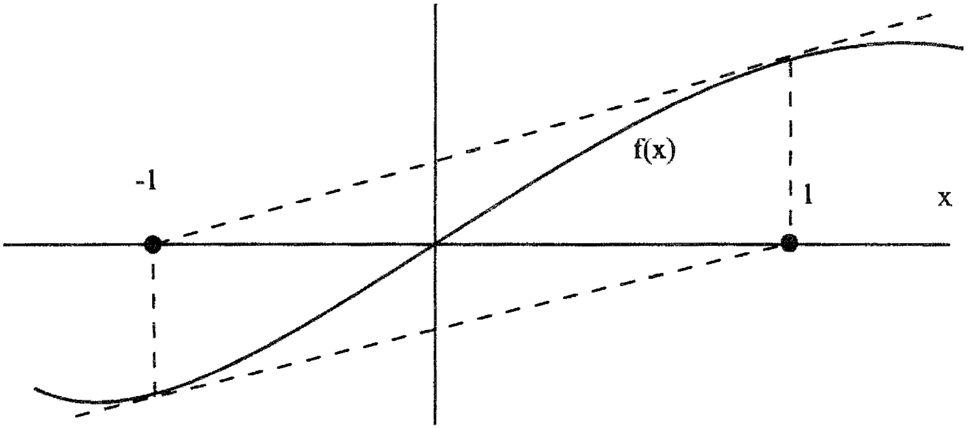

Figure 5.3
Newton method cycle

solutions affect the performance of Newton's method. We now examine both of these possibilities in the context of a simple equilibrium example.

Many economically interesting problems contain multiple solutions. This is illustrated in the following example of multiple equilibria taken from Kehoe (1991). Assume two goods and two consumers in an exchange economy. Agent i, $i = 1, 2$, has the utility function over consumption of goods 1 and 2:

$$u_i(x_1, x_2) = \frac{a_1^i x_1^{(\gamma_i+1)}}{\gamma_i + 1} + \frac{a_2^i x_2^{(\gamma_i+1)}}{\gamma_i + 1},$$

where $a_j^i \geq 0$, $\gamma_i < 0$. Let $\eta_i \equiv -1/\gamma_i$ be the constant elasticity of substitution in the utility function of agent i. If agent i has endowment $e^i \equiv (e_1^i, e_2^i)$ and the price of good j is p_j, then his demand function is $d_j^i(p) = \theta_j^i I^i p_j^{-\eta_i}$ where $I^i = p \cdot e^i$ is i's income and $\theta_j^i \equiv (a_j^i)^{\eta_i} / \sum_{l=1}^2 (a_l^i)^{\eta_i} p_l^{(1-\eta_i)}$. Assume that $a_1^1 = a_2^2 = 1024$, $a_2^1 = a_1^2 = 1$, $e_1^1 = e_2^2 = 12$, $e_2^1 = e_1^2 = 1$, $\gamma_1 = \gamma_2 = -5$, implying that $\eta_1 = \eta_2 = 0.2$. An equilibrium is a solution to the system

$$\sum_{i=1}^{2} d_1^i(p) = \sum_{i=1}^{2} e_1^i, \qquad p_1 + p_2 = 1, \tag{5.2.5}$$

where the first equation imposes supply equals demand for good 1 and the second equation normalizes the price vector to lie on the unit simplex. The pair of equations

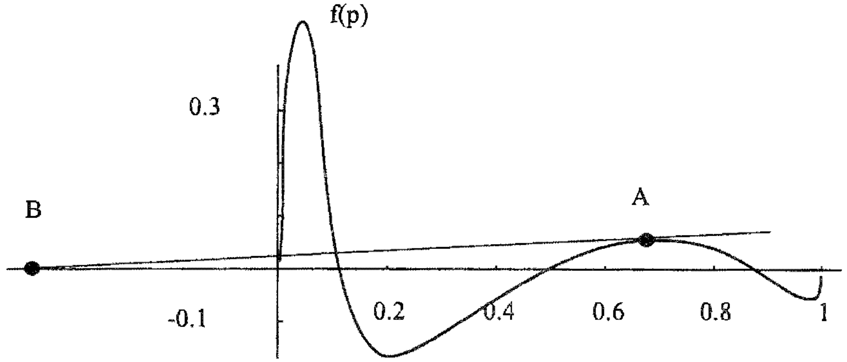

Figure 5.4
Excess demand function in (5.2.6)

in (5.2.5) suffice since Walras's law implies that supply will equal demand for good 2 at any solution to (5.2.5).

There are three equilibria to (5.2.5). They are $p^1 = (0.5, 0.5)$, the symmetric outcome, and two asymmetric equilibria at $p^2 = (0.1129, 0.8871)$ and $p^3 = (0.8871, 0.1129)$. We can reduce this problem to a one-variable problem when we make the substitution $p_2 = 1 - p_1$, yielding the problem

$$f(p_1) \equiv \sum_{i=1}^{2} d_1^i(p_1, 1 - p_1) - \sum_{i=1}^{2} e_1^i = 0. \tag{5.2.6}$$

The graph of f is displayed in figure 5.4.

We can use this example to examine the "dynamic" behavior of Newton's method. Each of the three equilibria are locally stable attraction points for Newton's method since the d_1^i have finite derivatives at each equilibrium.

However, between the equilibria Newton's method is not so well-behaved. A serious problem arises because the excess demand function f is defined only for positive p_1. Upon applying Newton's method to (5.2.6), one will often experience a severe problem; Newton's method will sometimes try to evaluate f at a negative price, where f is not defined. This *limited domain problem* is demonstrated in figure 5.4. If we began Newton's method at A, the next guess for p_1 would be B, outside the domain of f.

There are three approaches to deal with the limited domain problem. First, one could check at each iteration whether $f(x^{k+1})$ is defined, and if it is not, move x^{k+1} toward x^k until f is defined. This will prevent a crash due to undefined numerical operations. Unfortunately, this strategy requires access to the source code of the zero-finding routine, something that is often not possible. This new point may also generate an iterate outside the domain of f. With this strategy, one must be prepared to start over at a different initial guess.

Second, one could extend the definition of f so that it is defined at any price. Newton's method will now be globally defined. However, this is not generally easy. For Newton's method to have a good chance of getting back to the correct region, the extension should be a smooth extension of the existing function. In this case we also have the problem that demand is infinite at $p_1 = 0$, implying that any successful change must involve changing f for some region of positive p_1 values. One way to implement this *extension method* replaces f with $\tilde{f}(p_1)$, defined by

$$\tilde{f}(p_1) = \begin{cases} f(p_1), & p_1 > \varepsilon, \\ f(\varepsilon) + f'(\varepsilon)(p_1 - \varepsilon) + \dfrac{f''(\varepsilon)(p_1 - \varepsilon)^2}{2}, & p_1 \leq \varepsilon, \end{cases} \tag{5.2.7}$$

for some small $\varepsilon > 0$. This replaces f with a C^2 function that agrees with f at most positive prices and is defined for all prices. We may have introduced extra solutions, but if Newton's method converges to a solution with a negative price, we should just try again with a different initial guess. The trick here is to choose ε so that there are no solutions to $f(p_1) = 0$ in $(0, \varepsilon)$, the set of positive prices where $\tilde{f}(p_1) \neq f(p_1)$. Outside of this region, any positive zero of $\tilde{f}$ is also a zero of f. This approach is easy to apply to single variable problems but not so easy to implement in multivariate problems.

Third, one can change the variable so that Newton's method stays in the appropriate region. This typically involves a nonlinear change of variables. In this case we want p_1 to stay within $[0, 1]$. If $P : R \to (0, 1)$ is C^2, monotonically increasing, and onto, then restating the problem in terms of $z \equiv P^{-1}(p_1)$ produces a new system, $f(P(z)) = 0$, which is defined for all $z \in R$ and whose zeros match one-to-one with the zeros of $f(p_1)$. In particular, one could use $p_1 = P(z) \equiv e^z/(e^z + e^{-z})$ with the inverse map $z = (1/2)\ \ln(p_1/(1 - p_1))$. Newton's method applied to

$$g(z) \equiv f(P(z)) = 0 \tag{5.2.8}$$

results in the iteration

$$z_{k+1} = z_k - \frac{g(z_k)}{g'(z_k)}. \tag{5.2.9}$$

Equation (5.2.9) implies the p_1 iteration

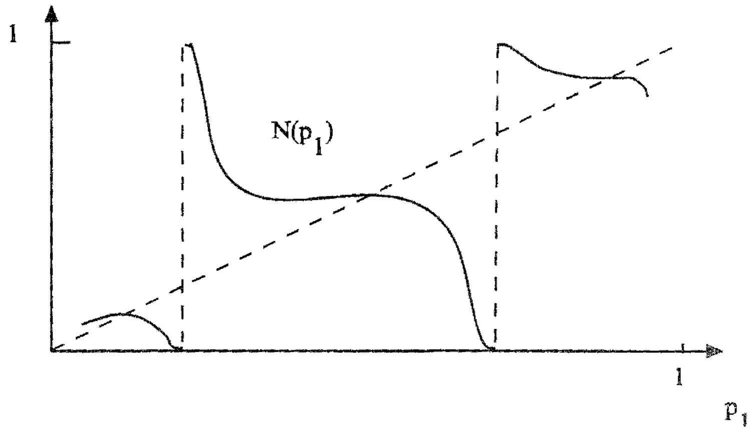

Figure 5.5
First iterate of Newton method applied to (5.2.6)

$$p_{1,k+1} = P\left(P^{-1}(p_{1,k}) - \frac{f(p_{1,k})}{f'(p_{1,k})P'(P^{-1}(p_{1,k}))}\right) \equiv N(p_{1,k}), \tag{5.2.10}$$

where each iterate is positive.

The limited domain problem may arise in many numerical contexts. We touched on it briefly in the optimization chapter. The extension approach and nonlinear transformations will also work for domain problems that arise in optimization problems. Domain problems can arise in almost any kind of numerical problem. In general, one of these tricks will be necessary to construct a reliable algorithm when some function used by the algorithm is not globally defined.

Even after transforming the variable, Newton's method can behave strangely. We plot the transformed Newton's iteration (5.2.10) as $N(p_1)$ in figure 5.5. Note that it is discontinuous near $p_1 = 0.21$ and $p_1 = 0.7$. In this case let us consider the domains of attraction for each of the solutions. Figure 5.6 displays the fourth iterate of N. Figure 5.6 shows that (5.2.10) converges to some equilibrium after four iterates for most initial guesses but that the domains of attraction are not connected. In particular, we find that (5.2.10) converges to $p_1 = 0.1129$ from $p_1 \in [0, 0.2]$ and $p_1 \in [0.67, 0.70]$ but not from most intervening initial guesses. In particular, there is no reason to trust that Newton's method will converge to the equilibrium nearest the initial guess.

These examples illustrate both the power of and problems with Newton's method. Newton's method is very useful for solving nonlinear equations, but it must be

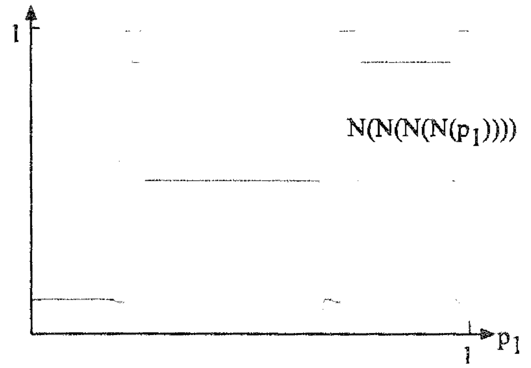

Figure 5.6
Fourth iterate of Newton method applied to (5.2.6)

applied carefully, with the user alert for the potential problems. These lessons will apply even more forcefully to the multivariate case studied below.

5.3 Special Methods for One-Dimensional Problems

We next consider two special one-dimensional methods. These do not directly generalize to multivariate problems but are quite useful in the single-variable case.

Secant Method

A key step in Newton's method is the computation of $f'(x)$, which may be costly. The *secant method* employs the idea of linear approximations but never evaluates f'. Instead, the secant method approximates $f'(x_k)$ with the slope of the secant of f between x_k and x_{k-1}, resulting in the iteration

$$x_{k+1} = x_k - \frac{f(x_k)\,(x_k - x_{k-1})}{f(x_k) - f(x_{k-1})}. \tag{5.3.1}$$

The secant method is the same as Newton's method except that step 1 uses equation (5.3.1) to compute the next iterate. The secant method suffers the same convergence problems as Newton's method, and when it converges, convergence is slower in terms of the number of required evaluations of f because of the secant approximation to the derivative. However, the running time can be much less because the secant method never evaluates f'. The convergence rate is between linear and quadratic.

THEOREM 3.1 If $f(x^*) = 0$, $f'(x^*) \neq 0$, and $f'(x)$ and $f''(x)$ are continuous near x^*, then the secant method converges at the rate $(1+\sqrt{5})/2$, that is

$$\limsup_{k\to\infty} \frac{|x_{k+1} - x^*|}{|x_k - x^*|^{(1+\sqrt{5})/2}} < \infty\,. \tag{5.3.2}$$

Proof See Young and Gregory (1988, vol. 1, pp. 150ff). ■

Fixed-Point Iteration

As with linear equations, we can often rewrite nonlinear problems in ways that suggest a computational approach. In general, any fixed-point problem $x = f(x)$ suggests the iteration $x_{k+1} = f(x_k)$. Consider the problem

$$x^3 - x - 1 = 0. \tag{5.3.3}$$

Equation (5.3.3) can be rewritten in the fixed-point form $x = (x+1)^{1/3}$, which suggests the iteration

$$x_{k+1} = (x_k + 1)^{1/3}. \tag{5.3.4}$$

The iteration (5.3.4) does converge to a solution of (5.3.3) if $x_0 = 1$. However, if we rewrite (5.3.3) as $x = x^3 - 1$, the suggested scheme

$$x_{k+1} = x_k^3 - 1 \tag{5.3.5}$$

diverges to $-\infty$ if $x_0 = 1$.

The fixed-point iteration approach to solving nonlinear equations is often useful but not generally reliable. We have focused on the more reliable methods, but that does not mean that one should ignore fixed-point iteration schemes. Also we should be flexible, looking for transformations that can turn unstable schemes into stable schemes, just as we transformed (5.3.3) into (5.3.4). In economic analysis the objective is to solve the problem $f(x) = 0$; how one finds the solution is of secondary importance.

5.4 Elementary Methods for Multivariate Nonlinear Equations

Most problems have several unknowns, requiring the use of multidimensional methods. Suppose that $f : R^n \to R^n$ and that we want to solve $f(x) = 0$, a list of n equations in n unknowns: