2

Nonlinear Problems in One Variable

We begin our study of the solution of nonlinear problems by discussing problems in just one variable: finding the solution of one nonlinear equation in one unknown, and finding the minimum of a function of one variable. The reason for studying one-variable problems separately is that they allow us to see those principles for constructing good local, global, and derivative-approximating algorithms that will also be the basis of our algorithms for multivariable problems, without requiring knowledge of linear algebra or multivariable calculus. The algorithms for multivariable problems will be more complex than those in this chapter, but an understanding of the basic approach here should help in the multivariable case.

Some references that consider the problems of this chapter in detail are Avriel (1976), Brent (1973), Conte and de Boor (1980), and Dahlquist, Björck, and Anderson (1974).

2.1 WHAT IS NOT POSSIBLE

Consider the problem of finding the real roots of each of the following three nonlinear equations in one unknown:

$$f_1(x) = x^4 - 12x^3 + 47x^2 - 60x,$$

$$f_2(x) = x^4 - 12x^3 + 47x^2 - 60x + 24,$$

$$f_3(x) = x^4 - 12x^3 + 47x^2 - 60x + 24.1.$$

(see Figure 2.1.1). It would be wonderful if we had a general-purpose computer routine that would tell us: "The roots of $f_1(x)$ are $x = 0$, 3, 4, and 5; the real roots of $f_2(x)$ are $x = 1$ and $x \cong 0.888$; $f_3(x)$ has no real roots."

It is unlikely that there will ever be such a routine. In general, the questions of existence and uniqueness—does a given problem have a solution, and is it unique?—are beyond the capabilities one can expect of algorithms that solve nonlinear problems. In fact, we must readily admit that for any computer algorithm there exist nonlinear functions (infinitely continuously differentiable, if you wish) perverse enough to defeat the algorithm. Therefore, all a user can be guaranteed from any algorithm applied to a nonlinear problem is the answer, "An approximate solution to the problem is _____," or, "No approximate solution to the problem was found in the alloted time." In many cases, however, the supplier of a nonlinear problem knows from practical considerations that it has a solution, and either that the solution is unique or that a solution in a particular region is desired. Thus the inability to determine the existence or uniqueness of solutions is usually not the primary concern in practice.

It is also apparent that one will be able to find only approximate solutions to most nonlinear problems. This is due not only to the finite precision of our computers, but also to the classical result of Galois that for some polynomials of degree $n \geq 5$, no closed-form solutions can be found using integers and the operations $+$, $-$, $\times$, $\div$, exponentiation, and second through nth roots. Therefore, we will develop methods that try to find one approximate solution of a nonlinear problem.

2.2 NEWTON'S METHOD FOR SOLVING ONE EQUATION IN ONE UNKNOWN

Our consideration of finding a root of one equation in one unknown begins with Newton's method, which is the prototype of the algorithms we will generate. Suppose we wish to calculate the square root of 3 to a reasonable number of places. This can be viewed as finding an approximate root x_* of the func-

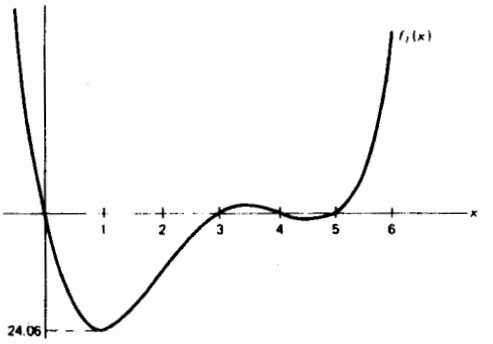


Figure 2.1.1 The equation $f_1(x) = x^4 - 12x^3 + 47x^2 - 60x$

tion $f(x) = x^2 - 3$ (see Figure 2.2.1). If our initial or current estimate of the answer is $x_c = 2$, we can get a better estimate x_+ by drawing the line that is tangent to $f(x)$ at $(2, f(2)) = (2, 1)$, and finding the point x_+ where this line crosses the x axis. Since

$$x_+ = x_c - \Delta x,$$

and

$$f'(x_c) = \frac{\Delta y}{\Delta x} = \frac{f(x_c)}{\Delta x},$$

we have that

$$f'(x_c)\Delta x = \Delta y = f(x_c)$$

or

$$x_+ = x_c - \frac{f(x_c)}{f'(x_c)} \tag{2.2.1}$$

which gives

$$x_+ = 2 - \tfrac{1}{4} = 1.75.$$

The logical thing to do next is to apply the same process from the new current estimate $x_c = 1.75$. Using (2.2.1) gives $x_+ = 1.75 - (0.0625/3.5) = 1.732\frac{1}{7}$, which already has four correct digits of $\sqrt{3}$. One more iteration gives $x_+ \cong 1.7320508$, which has eight correct digits.

The method we have just developed is called the *Newton-Raphson method* or *Newton's method*. It is important to our understanding to take a more abstract view of what we have done. At each iteration we have constructed a *local model* of our function $f(x)$ and solved for the root of the model. In

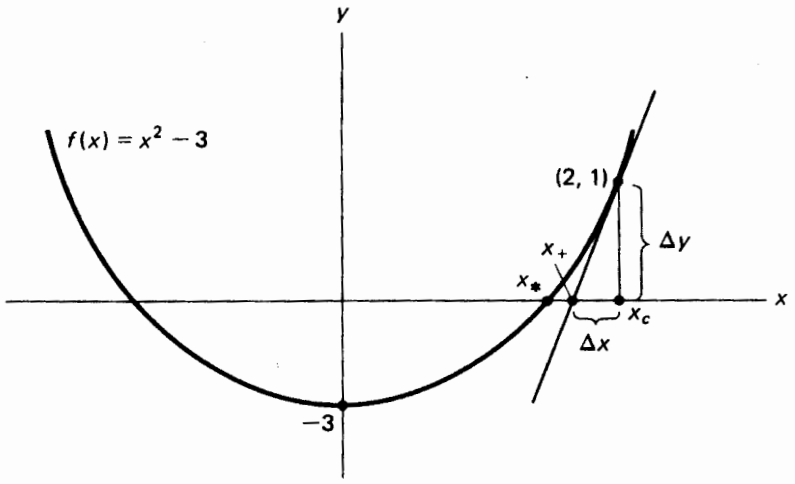


Figure 2.2.1 An iteration of Newton's method on $f(x) = x^2 - 3$ (not to scale)

the present case, our model

$$M_c(x) = f(x_c) + f'(x_c)(x - x_c) \tag{2.2.2}$$

is just the unique line with function value $f(x_c)$ and slope $f'(x_c)$ at the point x_c. [We use capital M to be consistent with the multidimensional case and to differentiate from minimization problems where our model is denoted by $m_c(x)$.] It is easy to verify that $M_c(x)$ crosses the x axis at the point x_+ defined by (2.2.1).

Pedagogical tradition calls for us to say that we have obtained Newton's method by writing $f(x)$ as its Taylor series approximation around the current estimate x_c,

$$f(x) = f(x_c) + f'(x_c)(x - x_c) + \frac{f''(x_c)(x - x_c)^2}{2!} + \cdots$$

$$= \sum_{i=0}^{\infty} \frac{f^i(x_c)(x - x_c)^i}{i!}, \tag{2.2.3}$$

and then approximating $f(x)$ by the affine* portion of this series, which naturally is given also by (2.2.2). Again the root is given by (2.2.1). There are several reasons why we prefer a different approach. It is unappealing and unnecessary to make assumptions about derivatives of any higher order than those actually used in the iteration. Furthermore, when we consider multivariable problems, higher-order derivatives become so complicated that they are harder to understand than any of the algorithms we will derive.

Instead, Newton's method comes simply and naturally from Newton's theorem,

$$f(x) = f(x_c) + \int_{x_c}^{x} f'(z)\, dz.$$

It seems reasonable to approximate the indefinite integral by

$$\int_{x_c}^{x} f'(z)\, dz \cong f'(x_c)(x - x_c)$$

and once more obtain the affine approximation to $f(x)$ given by (2.2.2). This type of derivation will be helpful to us in multivariable problems, where geometrical derivations become less manageable.

Newton's method is typical of methods for solving nonlinear problems; it is an iterative process that generates a sequence of points that we hope come increasingly close to a solution. The obvious question is, "Will it work?" The

* We will refer to (2.2.2) as an *affine* model, although colloquially it is often called a *linear* model. The reason is that an affine model corresponds to an affine subspace through $(x, f(x))$, a line that does not necessarily pass through the origin, whereas a linear subspace must pass through the origin.

answer is a qualified "Yes." Notice that if $f(x)$ were linear, Newton's method would find its root in one iteration. Now let us see what it will do for the general square-root problem;

$$\text{given } \alpha > 0, \qquad \text{find } x \text{ such that } f(x) = x^2 - \alpha = 0,$$

starting from a current guess $x_c \neq 0$. Since

$$x_+ = x_c - \frac{f(x_c)}{f'(x_c)} = x_c - \frac{x_c^2 - \alpha}{2x_c} = \frac{x_c}{2} + \frac{\alpha}{2x_c},$$

one has

$$x_+ - \sqrt{\alpha} = \frac{x_c}{2} + \frac{\alpha}{2x_c} - \sqrt{\alpha} = \frac{(x_c - \sqrt{\alpha})^2}{2x_c}, \tag{2.2.4a}$$

or, using relative error, one has

$$\frac{x_+ - \sqrt{\alpha}}{\sqrt{\alpha}} = \left(\frac{x_c - \sqrt{\alpha}}{\sqrt{\alpha}}\right)^2 \cdot \left(\frac{\sqrt{\alpha}}{2x_c}\right). \tag{2.2.4b}$$

Thus as long as the initial error $|x_c - \sqrt{\alpha}|$ is less than $|2x_c|$, the new error $|x_+ - \sqrt{\alpha}|$ will be smaller than the old error $|x_c - \sqrt{\alpha}|$, and eventually each new error will be much smaller than the previous error. This agrees with our experience for finding the square root of 3 in the example that began this section.

The pattern of decrease in error given by (2.2.4) is typical of Newton's method. The error at each iteration will be approximately the square of the previous error, so that, if the initial guess is good enough, the error will decrease and eventually decrease rapidly. This pattern is known as local q-quadratic convergence. Before deriving the general convergence theorem for Newton's method, we need to discuss rates of convergence.

2.3 CONVERGENCE OF SEQUENCES OF REAL NUMBERS

Given an iterative method that produces a sequence of points $x_1, x_2, \ldots$, from a starting guess x_0, we will want to know if the iterates converge to a solution x_*, and if so, how quickly. If we assume that we know what it means to write

$$\lim_{k \to \infty} a_k = 0$$

for a real sequence $\{a_k\}$, then the following definition characterizes the properties we will need.

Definition 2.3.1 Let $x_* \in \mathbb{R}$, $x_k \in \mathbb{R}$; $k = 0, 1, 2, \ldots$. Then the sequence

$\{x_k\} = \{x_0, x_1, x_2, \ldots\}$ is said to *converge* to x_* if

$$\lim_{k \to \infty} |x_k - x_*| = 0.$$

If in addition, there exists a constant $c \in [0, 1)$ and an integer $\hat{k} \geq 0$ such that for all $k \geq \hat{k}$,

$$|x_{k+1} - x_*| \leq c|x_k - x_*| \tag{2.3.1}$$

then $\{x_k\}$ is said to be *q-linearly convergent* to x_*. If for some sequence $\{c_k\}$ that converges to 0,

$$|x_{k+1} - x_*| \leq c_k|x_k - x_*|, \tag{2.3.2}$$

then $\{x_k\}$ is said to *converge q-superlinearly* to x_*. If there exist constants $p > 1$, $c \geq 0$, and $\hat{k} \geq 0$ such that $\{x_k\}$ converges to x_* and for all $k \geq \hat{k}$,

$$|x_{k+1} - x_*| \leq c|x_k - x_*|^p, \tag{2.3.3}$$

then $\{x_k\}$ is said to converge to x_* with *q-order at least p*. If $p = 2$ or 3, the convergence is said to be *q-quadratic* or *q-cubic*, respectively.

If $\{x_k\}$ converges to x_* and, in place of (2.3.2),

$$|x_{k+j} - x_*| \leq c_k|x_k - x_*|$$

for some fixed integer j, then $\{x_k\}$ is said to be *j-step q-superlinearly convergent* to x_*. If $\{x_k\}$ converges to x_* and, in place of (2.3.3), for $k > \hat{k}$,

$$|x_{k+j} - x_*| \leq c|x_k - x_*|^p$$

for some fixed integer j, then $\{x_k\}$ is said to have *j-step q-order convergence* of order *at least p*.

An example of a q-linearly convergent sequence is

$$x_0 = 2,\ x_1 = \tfrac{3}{2},\ x_2 = \tfrac{5}{4},\ x_3 = \tfrac{9}{8},\ \ldots,\ x_i = 1 + 2^{-i},\ \ldots.$$

This sequence converges to $x_* = 1$ with $c = \frac{1}{2}$; on a CDC machine it will take 48 iterations until $\mathrm{fl}(x_k) = 1$. An example of a q-quadratically convergent sequence is

$$x_0 = \tfrac{3}{2},\ x_1 = \tfrac{5}{4},\ x_2 = \tfrac{17}{16},\ x_3 = \tfrac{257}{256},\ \ldots,\ x_k = 1 + 2^{-2^k},\ \ldots,$$

which converges to $x_* = 1$ with $c = 1$; on a CDC machine, $\mathrm{fl}(x_6)$ will equal 1. In practice, q-linear convergence can be fairly slow, whereas q-quadratic or q-superlinear convergence is eventually quite fast. However, actual behavior also depends upon the constants c in (2.3.1–2.3.3); for example, q-linear convergence with $c = 0.001$ is probably quite satisfactory, but with $c = 0.9$ it is not. (For further examples see Exercises 2 and 3). It is worth emphasizing that the utility of q-superlinear convergence is directly related to how many iterations are needed for c_k to become small.

The prefix "q" stands for quotient and is used to differentiate from "r" (root) orders of convergence. R-order* is a weaker type of convergence rate; all that is said of the errors $|x_k - x_*|$, of a sequence with r-order p, is that they are bounded above by another sequence of q-order p. A definitive reference is Ortega and Rheinboldt [1970]. An iterative method that will converge to the correct answer at a certain rate, provided it is started close enough to the correct answer, is said to be *locally convergent* at that rate. In this book we will be interested mainly in methods that are locally q-superlinearly or q-quadratically convergent and for which this behavior is apparent in practice.

2.4 CONVERGENCE OF NEWTON'S METHOD

We now show that, for most problems, Newton's method will converge q-quadratically to the root of one nonlinear equation in one unknown, provided it is given a good enough starting guess. However, it may not converge at all from a poor start, so that we need to incorporate the global methods of Section 2.5. The local convergence proof for Newton's method hinges on an estimate of the errors in the sequence of affine models $M_c(x)$ as approximations to $f(x)$. Since we obtained the approximations by using $f'(x_c)(x - x_c)$ to approximate

$$\int_{x_c}^{x} f'(z)\, dz$$

we are going to need to make some smoothness assumptions on f' in order to estimate the error in the approximation, which is

$$f(x) - M_c(x) = \int_{x_c}^{x} [f'(z) - f'(x_c)]\, dz.$$

First we define the notion of Lipschitz continuity.

Definition 2.4.1 A function g is Lipschitz continuous with constant γ in a set X, written $g \in \mathrm{Lip}_\gamma(X)$, if for every $x, y \in X$,

$$|g(x) - g(y)| \leq \gamma |x - y|.$$

In order to prove the convergence of Newton's method, we first prove a simple lemma showing that if $f'(x)$ is Lipschitz continuous, then we can obtain a bound on how close the affine approximation $f(x) + f'(x)(y - x)$ is to $f(y)$.

---

* We will capitalize the prefix letters R and Q when they begin a sentence, but not otherwise.

LEMMA 2.4.2 For an open interval D, let $f: D \to \mathbb{R}$ and let $f' \in \text{Lip}_\gamma(D)$. Then for any $x, y \in D$,

$$|f(y) - f(x) - f'(x)(y - x)| \leq \frac{\gamma(y - x)^2}{2}. \tag{2.4.1}$$

Proof. From basic calculus, $f(y) - f(x) = \int_x^y f'(z)\,dz$, or equivalently,

$$f(y) - f(x) - f'(x)(y - x) = \int_x^y [f'(z) - f'(x)]\,dz. \tag{2.4.2}$$

Making the change of variables

$$z = x + t(y - x), \qquad dz = dt(y - x),$$

(2.4.2) becomes

$$f(y) - f(x) - f'(x)(y - x) = \int_0^1 [f'(x + t(y - x)) - f'(x)](y - x)\,dt,$$

and so by the triangle inequality applied to the integral and the Lipschitz continuity of f',

$$|f(y) - f(x) - f'(x)(y - x)| \leq |y - x| \int_0^1 \gamma |t(y - x)|$$

$$= \gamma |y - x|^2/2. \qquad \square$$

Note that (2.4.1) closely resembles the error bound given by the Taylor series with remainder, with the Lipschitz constant γ taking the place of a bound on $|f''(\xi)|$ for $\xi \in D$. The main advantage of using Lipschitz continuity is that we do not need to discuss this next higher derivative. This is especially convenient in multiple dimensions.

We are now ready to state and prove a fundamental theorem of numerical mathematics. We will prove the most useful form of the result and leave the more general ones as exercises (see Exercises 13-14.)

THEOREM 2.4.3 Let $f: D \to \mathbb{R}$, for an open interval D, and let $f' \in \text{Lip}_\gamma(D)$. Assume that for some $\rho > 0$, $|f'(x)| \geq \rho$ for every $x \in D$. If $f(x) = 0$ has a solution $x_* \in D$, then there is some $\eta > 0$ such that: if $|x_0 - x_*| < \eta$, then the sequence $\{x_k\}$ generated by

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}, \qquad k = 0, 1, 2, \ldots$$

exists and converges to x_*. Furthermore, for $k = 0, 1, \ldots,$

$$|x_{k+1} - x_*| \leq \frac{\gamma}{2\rho} |x_k - x_*|^2. \tag{2.4.3}$$

Proof. Let $\tau \in (0, 1)$, let $\hat{\eta}$ be the radius of the largest open interval around x_* that is contained in D, and define $\eta = \min\{\hat{\eta}, \tau(2\rho/\gamma)\}$. We will show by induction that for $k = 0, 1, 2, \ldots$, (2.4.3) holds, and

$$|x_{k+1} - x_*| \leq \tau |x_k - x_*| < \eta.$$

The proof simply shows at each iteration that the new error $|x_{k+1} - x_*|$ is bounded by a constant times the error the affine model makes in approximating f at x_*, which from Lemma 2.4.2 is $O(|x_k - x_*|^2)$. For $k = 0$,

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

$$= \frac{1}{f'(x_0)} [f(x_*) - f(x_0) - f'(x_0)(x_* - x_0)].$$

The term in brackets is $f(x_*) - M_0(x_*)$, the error at x_* in the local affine model at $x_c = x_0$. Thus from Lemma 2.4.2,

$$|x_1 - x_*| \leq \frac{\gamma}{2|f'(x_0)|} |x_0 - x_*|^2,$$

and by the assumptions on $f'(x)$

$$|x_1 - x_*| \leq \frac{\gamma}{2\rho} |x_0 - x_*|^2.$$

Since $|x_0 - x_*| \leq \eta \leq \tau \cdot 2\rho/\gamma$, we have $|x_1 - x_*| \leq \tau |x_0 - x_*| < \eta$. The proof of the induction step then proceeds identically. □

The condition in Theorem 2.4.3 that $f'(x)$ have a nonzero lower bound in D simply means that $f'(x_*)$ must be nonzero for Newton's method to converge quadratically. Indeed, if $f'(x_*) = 0$, then x_* is a multiple root, and Newton's method converges only linearly (see Exercise 12). To appreciate the difference, we give below sample iterations of Newton's method applied to $f_1(x) = x^2 - 1$ and $f_2(x) = x^2 - 2x + 1$, both starting from $x_0 = 2$. Notice how much more slowly Newton's method converges on $f_2(x)$ because $f_2'(x_*) = 0$.

EXAMPLE 2.4.4 Newton's Method Applied to Two Quadratics (CDC, Single Precision)

$f_1(x) = x^2 - 1$		$f_2(x) = x^2 - 2x + 1$
2	x_0	2
1.25	x_1	1.5
1.025	x_2	1.25
1.0003048780488	x_3	1.125
1.0000000464611	x_4	1.0625
1.0	x_5	1.03125

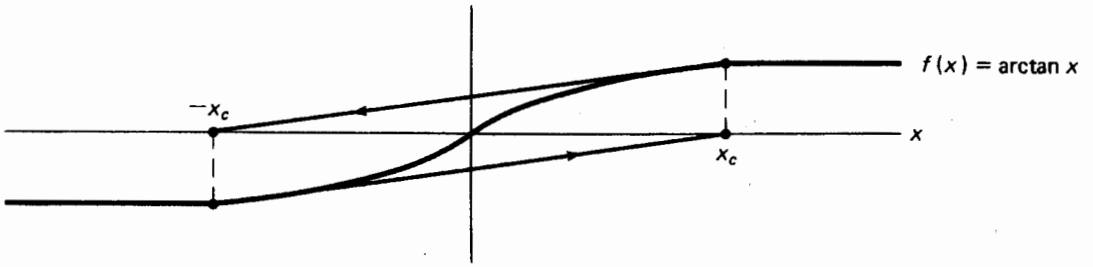


Figure 2.4.1 Newton's method applied to $f(x) = \arctan(x)$

It is also informative to examine the constant $\gamma/2\rho$ involved in the q-quadratic convergence relation (2.4.3). The numerator γ, a Lipschitz constant for f' on D, can be considered a measure of the nonlinearity of f. However, γ is a scale-dependent measure; multiplying f or changing the units of x by a constant will scale f' by that constant without making the function more or less nonlinear. A partially scale-free measure of nonlinearity is the relative rate of change in $f'(x)$, which is obtained by dividing γ by $f'(x)$. Thus, since ρ is a lower bound on $f'(x)$ for $x \in D$, γ/ρ is an upper bound on the relative nonlinearity of $f(x)$, and Theorem 2.4.3 says that the smaller this measure of relative nonlinearity, the faster Newton's method will converge. If f is linear, then $\gamma = 0$ and $x_1 = x_*$.

Theorem 2.4.3 guarantees the convergence of Newton's method only from a good starting point x_0, and indeed it is easy to see that Newton's method may not converge at all if $|x_0 - x_*|$ is large. For example, consider the function $f(x) = \arctan x$ (see Figure 2.4.1). For some $x_c \in [1.39, 1.40]$, if $x_0 = x_c$, then Newton's method will produce the cycle $x_1 = -x_c$, $x_2 = x_c$, $x_3 = -x_c, \ldots$. If $|x_0| < x_c$, Newton's method will converge to $x_* = 0$, but if $|x_0| > x_c$, Newton's method will diverge; i.e., the error $|x_k - x_*|$ will increase at each iteration. Thus Newton's method is useful to us for its fast local convergence, but we need to incorporate it into a more robust method that will be successful from farther starting points.

2.5 GLOBALLY CONVERGENT METHODS* FOR SOLVING ONE EQUATION IN ONE UNKNOWN

We will use a simple philosophy to incorporate Newton's method into a globally convergent algorithm: use Newton's method whenever it seems to be working well, otherwise fall back on a slower but sure global method. This strategy produces globally convergent algorithms with the fast local conver-

* For our definition of "global method," see the last paragraph of Section 1.1.

gence of Newton's method. In this section we discuss two global methods and then show how to combine a global method with Newton's method into a hybrid algorithm. We also discuss the stopping tests and other computer-dependent criteria necessary to successful computational algorithms.

The simplest global method is the method of bisection. It makes the somewhat reasonable assumption that one starts with an interval $[x_0, z_0]$ that contains a root. It sets x_1 to the midpoint of this interval, chooses the new interval to be the one of $[x_0, x_1]$ or $[x_1, z_0]$ that contains a root, and continues to halve the interval until a root is found (see Figure 2.5.1). This is expressed algebraically as:

given x_0, z_0 such that $f(x_0) \cdot f(z_0) < 0$,

for $k = 0, 1, 2, \ldots$, do

$$x_{k+1} = \frac{x_k + z_k}{2}$$

$$z_{k+1} = \begin{cases} x_k, & \text{if } f(x_k) \cdot f(x_{k+1}) < 0, \\ z_k, & \text{otherwise.} \end{cases}$$

The method of bisection always works in theory, but it is guaranteed only to reduce the error bound by $\frac{1}{2}$ for each iteration. This makes the method very marginal for practical use. Programs that use bisection generally do so only until an x_k is obtained from which some variant of Newton's method will converge. The method of bisection also does not extend naturally to multiple dimensions.

A method more indicative of how we will proceed in n-space is the following. Think of Newton's method as having suggested not only the step $x_N = x_c - f(x_c)/f'(x_c)$, but also the direction in which that step points. [Assume $f'(x_c) \neq 0$.] Although the Newton step may actually cause an increase in the absolute value of the function, its direction always will be one in which the absolute function value decreases initially (see Figure 2.5.2). This should be obvious geometrically; for the simple proof, see Exercise 16. Thus, if the Newton point x_N doesn't produce a decrease in $|f(x)|$, a reasonable strategy is to backtrack from x_N toward x_c until one finds a point x_+ for

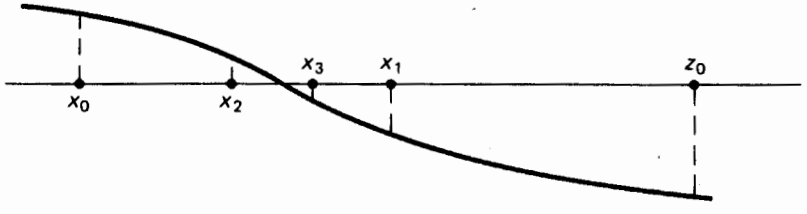


Figure 2.5.1 The method of bisection

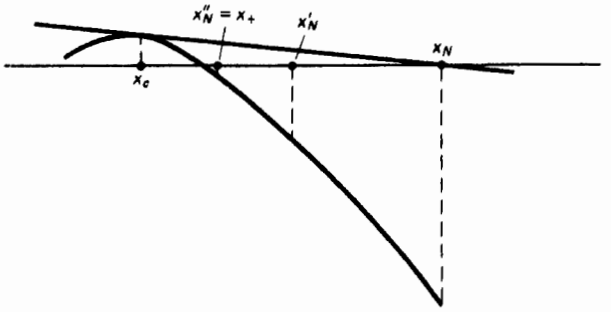


Figure 2.5.2 Backtracking from the Newton step

which $|f(x_+)| < |f(x_c)|$. A possible iteration is

$$x_+ = x_c - \frac{f(x_c)}{f'(x_c)}$$

$$\text{while } |f(x_+)| \geq |f(x_c)| \text{ do}$$

$$x_+ \leftarrow \frac{x_+ + x_c}{2}. \tag{2.5.1}$$

Note that this strategy does not require an initial interval bracketing a root.

Iteration (2.5.1) is an example of a hybrid algorithm, one that attempts to combine global convergence and fast local convergence by first trying the Newton step at each iteration, but always insisting that the iteration decreases some measure of the closeness to a solution. Constructing such hybrid algorithms is the key to practical success in solving multivariable nonlinear problems. Below is the general form of a class of hybrid algorithms for finding a root of one nonlinear equation; it is meant to introduce and emphasize those basic techniques for constructing globally and fast locally convergent algorithms that will be the foundation of all the algorithms in this book.

ALGORITHM 2.5.1 General hybrid quasi-Newton algorithm for solving one nonlinear equation in one unknown:

given $f: \mathbb{R} \to \mathbb{R}$, x_0,

for $k = 0, 1, 2, \ldots,$ do

1. *decide* whether to stop; if not:

2. *make a local model* of f around x_k, and find the point x_N that solves (or comes closest to solving) the model problem.

3. (a) *decide* whether to take $x_{k+1} = x_N$, if not,

   (b) choose x_{k+1} using a *global strategy* (make more conservative use of the solution to the model problem).

Step 1 is discusse
and problem-depende
Newton step, or a vari
is an example of Step
Step 3(a) has to be ch
convergence in most c

Deciding when t
for every problem, ye
computer-representabl
"close enough." This
mately solved the pro
virtually the same pla
"Is $|f(x_+)| < \tau_1$?" wh
being close enough to
$(\text{macheps})^{1/2}$. Naturall
important that a rout
x_+ that satisfies $|f(x$
Partly to guard aga
question is included
$|x_+|) < \tau_2$?" A reaso
to stopping whenever
τ_2 greater than mach
second test is usually
$|x_c|\} < \tau_2$?" A better
typical size of x in the
that the stopping con

if $|f(x_+)|$:

In practice, $f(x_+)$ usu
which local converge
only linear, the step
the choice of stoppi
scaled problems. We

2.6 METHODS W
ARE UNAVA

In many practical ap
output from some c
ually is not availabl
$f(x) = 0$ must be mod

Step 1 is discussed below; it requires our first use of computer-dependent and problem-dependent tolerances. Step 2 usually involves calculating the Newton step, or a variant without derivatives (see Section 2.6). Equation (2.5.1) is an example of Step 3(a)–(b). We will see in Chapter 6 that the criterion in Step 3(a) has to be chosen with only a little bit of care to assure the global convergence in most cases of the hybrid algorithm to a solution.

Deciding when to stop is a somewhat ad hoc process that can't be perfect for every problem, yet it calls for considerable care. Since there may be no computer-representable x_* such that $f(x_*) = 0$, one must decide when one is "close enough." This decision usually takes two parts: "Have you approximately solved the problem?" or "Have the last two (or few) iterates stayed in virtually the same place?" The first question is represented by a test such as, "Is $|f(x_+)| < \tau_1$?" where the tolerance τ_1 is chosen to reflect the user's idea of being close enough to zero for this problem. For example, τ_1 might be set to $(\text{macheps})^{1/2}$. Naturally this test is very sensitive to the scale of $f(x)$, and so it is important that a routine instruct the user to choose τ_1, or scale f, so that an x_+ that satisfies $|f(x_+)| < \tau_1$ will be a satisfactory solution to the problem. Partly to guard against this condition's being too restrictive, the second question is included and it is tested by a relation such as, "Is $(|x_+ - x_c|/|x_+|) < \tau_2$?" A reasonable tolerance is $\tau_2 = (\text{macheps})^{1/2}$, which corresponds to stopping whenever the left half of the digits of x_c and x_+ agree, though any τ_2 greater than macheps can be selected. Since x_+ might be close to zero, the second test is usually modified to something like, "Is $(|x_+ - x_c|/\max\{|x_+|, |x_c|\} < \tau_2$?" A better test uses a user-supplied variable typx containing the typical size of x in the place of the $|x_c|$ in the denominator (see Exercise 17), so that the stopping condition on a CDC machine might be,

$$\text{if} \quad |f(x_+)| \le 10^{-5} \quad \text{or} \quad \frac{|x_+ - x_c|}{\max\{\text{typx}, |x_+|\}} \le 10^{-7}, \quad \text{stop.}$$

In practice, $f(x_+)$ usually gets small before the step does in any problem for which local convergence is fast, but for a problem on which convergence is only linear, the step may become small first. The reader can already see that the choice of stopping rules is quite a can of worms, especially for poorly scaled problems. We will treat it more completely in Chapter 7.

2.6 METHODS WHEN DERIVATIVES ARE UNAVAILABLE

In many practical applications, $f(x)$ is not given by a formula; rather it is the output from some computational or experimental procedure. Since $f'(x)$ usually is not available then, our methods that use values of $f'(x)$ to solve $f(x) = 0$ must be modified to require only values of $f(x)$.

We have been using $f'(x)$ in modeling f near the current solution estimate x_c by the line tangent to f at x_c. When $f'(x)$ is unavailable, we replace this model by the secant line that goes through f at x_c and at some nearby point $x_c + h_c$ (see Figure 2.6.1). It is easy to calculate that the slope of this line is

$$a_c = \frac{f(x_c + h_c) - f(x_c)}{h_c} \tag{2.6.1}$$

and so the model we obtain is the line

$$\hat{M}_c(x) = f(x_c) + a_c(x - x_c).$$

Therefore, what we have done is equivalent to replacing the derivative $f'(x_c)$ in our original model $M_c(x) = f(x_c) + f'(x_c)(x - x_c)$ by the approximation a_c. The quasi-Newton step to the zero of $\hat{M}_c(x)$ then becomes

$$\hat{x}_N = x_c - \frac{f(x_c)}{a_c}.$$

Two questions immediately arise: "Will this method work?" and, "How should h_c be chosen?" We know from calculus that as $h_c \to 0$, a_c will converge to $f'(x_c)$. If h_c is chosen to be a small number, a_c is called a *finite-difference approximation* to $f'(x_c)$. It seems reasonable that the finite-difference Newton method, the quasi-Newton method based on this approximation, should work almost as well as Newton's method, and we will see later that it does. However, this technique requires an additional evaluation of f at each iteration, and if $f(x)$ is expensive, the extra function evaluation is undesirable. In this case, h_c is set to $(x_- - x_c)$, where x_- is the previous iterate, so that $a_c = (f(x_-) - f(x_c))/(x_- - x_c)$ and no additional function evaluations are used. The resultant quasi-Newton algorithm is called a *secant method*. While it may seem totally ad hoc, it also turns out to work well; the convergence of the secant method is slightly slower than a properly chosen finite-difference method, but it is usually more efficient in terms of total function evaluations required to achieve a specified accuracy.

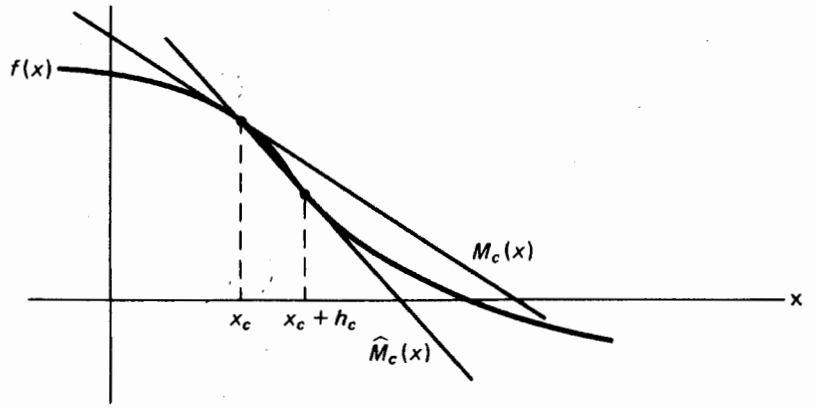


Figure 2.6.1 A secant approximation to $f(x)$

Example 2.6.1
and the finite-differ
$f(x) = x^2 - 1$, $x_0 =$
Notice how simila
method are; the sec

EXAMPLE 2.6.1
applied to $f(x) = x^2$

Finite- (h_k
2
1.2500
1.0250
1.0003
1.0000
1.0

There is a gr
of the difference
difference Newton

Then

$$x_+ - x_* = x_c -$$

$$= a_c^{-1}$$

$$= a_c^{-1}$$

$$= a_c^{-1}$$

$$= a_c^{-1}$$

If we define $e_c =$ |

under the same
2.4.2.

Example 2.6.1 contains samples of the convergence of the secant method and the finite-difference Newton's method (with $h_c = 10^{-7}x_c$) on the problem $f(x) = x^2 - 1$, $x_0 = 2$, that was solved by Newton's method in Example 2.4.4. Notice how similar Newton's method and the finite-difference Newton's method are; the secant method is a bit slower.

EXAMPLE 2.6.1 Finite-difference Newton's method and the secant method applied to $f(x) = x^2 - 1$ *(CDC, single precision).*

Finite-Difference N.M. ($h_k = 10^{-7} \cdot x_k$)		Secant Method (x_1 chosen by f.d.N.M.)
2	x_0	2
1.2500000266453	x_1	1.2500000266453
1.0250000179057	x_2	1.0769230844910
1.0003048001120	x_3	1.0082644643823
1.0000000464701	x_4	1.0003048781354
1.0	x_5	1.0000012544523
	x_6	1.0000000001912
	x_7	1.0

There is a great deal of insight to be gained from an analysis of the effect of the difference step h_c on the convergence rate of the resultant finite-difference Newton method. Let us take

$$x_+ = \hat{x}_N = x_c - \frac{f(x_c)}{a_c}.$$

Then

$$\begin{aligned} x_+ - x_* &= x_c - x_* - \frac{f(x_c)}{a_c} \\ &= a_c^{-1}[f(x_*) - f(x_c) - a_c(x_* - x_c)] \\ &= a_c^{-1}[f(x_*) - \hat{M}_c(x_*)] \\ &= a_c^{-1}\{f(x_*) - f(x_c) - f'(x_c)(x_* - x_c) + [f'(x_c) - a_c](x_* - x_c)\} \\ &= a_c^{-1}\left\{\int_{x_c}^{x_*} [f'(z) - f'(x_c)]\, dz + [f'(x_c) - a_c](x_* - x_c)\right\}. \end{aligned}$$

If we define $e_c = |x_c - x_*|$ and $e_+ = |x_+ - x_*|$, then we have

$$e_+ \le |a_c^{-1}|\left(\frac{\gamma}{2} e_c^2 + |f'(x_c) - a_c|\, e_c\right), \tag{2.6.2}$$

under the same Lipschitz continuity assumption $f' \in \mathrm{Lip}_\gamma(D)$ as in Lemma 2.4.2.

The reader can see that (2.6.2) is similar to our error bound for Newton's method in Section 2.4, except that the right-hand side of (2.6.2) has $|a_c^{-1}|$ in the place of $|f'(x_c)^{-1}|$, and an additional term involving the difference between $f'(x_c)$ and its approximation a_c. Notice also that the above analysis so far is independent of the value of $a_c \neq 0$. Now let us define a_c by (2.6.1) and again bring in the assumption of Lipschitz continuity of f' from Section 2.4. Then we have an easy corollary which tells us how close, as a function of h_c, the finite-difference approximation (2.6.1) is to $f'(x_c)$.

COROLLARY 2.6.2 Let $f: D \to \mathbb{R}$ for an open interval D and let $f' \in \text{Lip}_\gamma(D)$. Let $x_c, x_c + h_c \in D$, and define a_c by (2.6.1). Then

$$|a_c - f'(x_c)| \leq \frac{\gamma |h_c|}{2}. \tag{2.6.3}$$

Proof. From Lemma 2.4.2,

$$|f(x_c + h_c) - f(x_c) - h_c f'(x_c)| \leq \frac{\gamma |h_c|^2}{2}$$

Dividing both sides by $|h_c|$ gives the desired result. □

Substituting (2.6.3) into (2.6.2) gives

$$e_+ \leq \frac{\gamma}{2|a_c|}(e_c + |h_c|)e_c.$$

Now, if we bring in from Theorem 2.4.3 the assumption that $|f'(x)| \geq \rho > 0$ in a neighborhood of x, it is easy to show from (2.6.3), for $|h_c|$ sufficiently small, and $x_c \in D$, that $|a_c^{-1}| \leq 2\rho^{-1}$. This gives

$$e_+ \leq \frac{\gamma}{\rho}(e_c + |h_c|)e_c. \tag{2.6.4}$$

At this point it is really rather easy to finish a proof of the following theorem.

THEOREM 2.6.3 Let $f: D \to \mathbb{R}$ for an open interval D and let $f' \in \text{Lip}_\gamma(D)$. Assume that $|f'(x)| \geq \rho$ for some $\rho > 0$ and for every $x \in D$. If $f(x) = 0$ has a solution $x_* \in D$, then there exist positive constants η, η' such that if $\{h_k\}$ is a real sequence with $0 < |h_k| \leq \eta'$, and if $|x_0 - x_*| < \eta$, then the sequence $\{x_k\}$ defined by

$$x_{k+1} = x_k - \frac{f(x_k)}{a_k}, \qquad a_k = \frac{f(x_k + h_k) - f(x_k)}{h_k}, \qquad k = 0, 1, \ldots,$$

is defined and converges q-linearly to x_*. If $\lim_{k\to\infty} h_k = 0$, then the convergence is q-superlinear. If there exists some constant c_1 such that

$$|h_k| \le c_1 |x_k - x_*|,$$

or equivalently, a constant c_2 such that

$$|h_k| \le c_2 |f(x_k)|, \tag{2.6.5}$$

then the convergence is q-quadratic. If there exists some constant c_3 such that

$$|h_k| \le c_3 |x_k - x_{k-1}|, \tag{2.6.6}$$

then the convergence is at least two-step q-quadratic; this includes the secant method $h_k = x_{k-1} - x_k$ as a special case.

We began this discussion with a claim that insight is contained in the existence and simplicity of the analysis. In particular, the finite-difference idea seemed somewhat ad hoc when we introduced it. However, not only can we observe its excellent computational behavior in practice, but with analysis hardly more complicated than for Newton's method we can characterize exactly how it will converge as a function of the stepsizes $\{h_k\}$. This is just one of many instances in numerical analysis where theoretical analysis is an ally to algorithmic development.

If order of convergence were the only consideration in choosing $\{h_k\}$ in the finite-difference Newton's method, then we would just set $h_c = c_2 |f(x_c)|$ for some constant c_2, and our analysis would be complete, since it is easy to show for x_c sufficiently close to x_* that $|f(x_c)| \le \hat{c} |x_c - x_*|$. (See Exercise 18.) However, finite-precision arithmetic has a very important effect that we want to look at now. Obviously in practice h_c cannot be too small in relation to x_c. For example, if $\text{fl}(x_c) \neq 0$ and $|h_c| < |x_c| \cdot$ macheps, then $\text{fl}(x_c + h_c) = \text{fl}(x_c)$ and so the numerator of (2.6.1) will be computed to be zero, since the subroutine that evaluates f will have been called with the same floating-point argument both times.

Even if $|h_c|$ is large enough that $\text{fl}(x_c + h_c) \neq \text{fl}(x_c)$, there remains the loss of accuracy in the floating-point subtraction necessary to form the numerator of (2.6.1). After all, since f is continuously differentiable, if f is evaluated at two nearby points, then the resulting values of $f(x_c + h_c)$ and $f(x_c)$ should also be nearly equal. We might even have $\text{fl}(f(x_c + h_c)) = \text{fl}(f(x_c))$. If $|h_c|$ is small, we certainly should expect to have several of the leftmost digits of the mantissa the same. For example, suppose on a base-10 machine with 5-digit mantissa we have $f(x_c) = 1.0001$ and $f(x_c + h_c) = 1.0010$ returned by some hypothetical function subroutine with $h_c = 10^{-4}$. In this case, $f(x_c + h_c) - f(x_c)$ would be computed as 9×10^{-4} and a_c as 9. Thus we have lost most of the significant digits in taking the difference of the function values.

In practice, furthermore, one often has progressively less confidence in the digits of the function values as one scans to the right. This is due not only to finite-precision arithmetic, but also to the fact that the function values are sometimes themselves just the approximate results returned by a numerical routine. Thus in the above example the difference $f(x_c + h_c) - f(x_c) = 9 \times 10^{-4}$ may reflect only random fluctuation in the rightmost digits of f and have no significance at all. The result is that the slope of the model of f at x_c may not even have the same sign as $f'(x_c)$.

The obvious way to compute a_c more accurately is to take $|h_c|$ large enough so that not all the most trustworthy leading digits are the same for $f(x_c + h_c)$ and $f(x_c)$. There is a limit to how large $|h_c|$ can be, since our whole object in computing a_c is to use it as an approximation to $f'(x_c)$, and (2.6.3) indicates that this approximation deteriorates as $|h_c|$ increases. A good compromise is to try to balance the nonlinearity error caused by taking $|h_c|$ too large with the finite-precision and function evaluation errors from allowing $|h_c|$ too small. We develop this as an exercise (Exercise 19) and discuss it further in Section 5.4.

A simple rule for the case when f is assumed to be computed accurately to machine precision is to perturb roughly half the digits of x_c:

$$|h_c| = \sqrt{\text{macheps}} \cdot \max \{\text{typ}x, |x_c|\},$$

where typ$x$ is the typical size of x used in Section 2.5. This often-used rule is generally satisfactory in practice. If the accuracy of the f-subroutine is suspect, then $|h_c|$ should be large enough so that only half the reliable digits of $f(x_c + h_c)$ and $f(x_c)$ will be the same. If this $|h_c|$ is large enough to cast doubt on the utility of (2.6.1) as an approximation to $f'(x_c)$, then one remedy is to use the central difference,

$$a_c = \frac{f(x_c + h_c) - f(x_c - h_c)}{2h_c}. \tag{2.6.7}$$

This gives a more accurate derivative approximation for a given h_c (see Exercise 20) than the forward difference approximation (2.6.1), but it has the disadvantage of doubling the expense in function evaluations.

2.7 MINIMIZATION OF A FUNCTION OF ONE VARIABLE

We conclude our study of one-dimensional problems by discussing minimization of a function of one variable. It turns out that this problem is so closely related to solving one nonlinear equation in one unknown that we virtually know already how to compute solutions.

First of all, one must again admit that one cannot practically answer questions of existence and uniqueness. Consider, for example, the fourth-

degree polynomial of Figure 2.1.1. Its *global minimizer*, where the function takes its absolute lowest value, is at $x \cong 0.943$, but it also has a *local minimizer*, a minimizing point in an open region, at $x \cong 4.60$. If we divide the function by x, it becomes a cubic with a local minimum at $x \cong 4.58$, but no finite global minimizer since

$$\lim_{x \to -\infty} f(x) = -\infty.$$

In general, it is practically impossible to know if you are at the global minimum of a function. So, just as our algorithms for nonlinear equations can find only one root, our optimization algorithms at best can locate one local minimum; usually this is good enough in practice. Once again, a closed-form solution is out of the question: if we could find a closed form for a global minimizer of $f(x)$, we could solve $\hat{f}(x) = 0$ by setting $f(x) = \hat{f}(x)^2$.

The reason we said earlier that the reader already knows how to solve the minimization problem is that a local minimum of a continuously differentiable function must come at a point where $f'(x) = 0$. Graphically, this just says that the function can't initially decrease in either direction from such a point. A proof of this fact suggests an algorithm, so we give it below. It will be useful to denote by $C^1(D)$ and $C^2(D)$, respectively, the sets of once and twice continuously differentiable functions from D into $\mathbb{R}$.

THEOREM 2.7.1 Let $f \in C^1(D)$ for an open interval D, and let $z \in D$. If $f'(z) \neq 0$, then for any s with $f'(z) \cdot s < 0$, there is a constant $t > 0$ for which $f(z + \lambda s) < f(z)$, for every $\lambda \in (0, t)$.

Proof. We need only choose t, using the continuity of f', so that $f'(z + \lambda s)s < 0$ and $z + \lambda s \in D$ for every $\lambda \in (0, t)$. The rest is immediate from calculus, since, for any such λ,

$$f(z + \lambda s) - f(z) = \int_0^\lambda f'(z + \alpha s)s \, d\alpha < 0. \qquad \square$$

Theorem 2.7.1 suggests that we find a minimizing point x_* of a function f by solving $f'(x) = 0$. On the other hand, by applying the theorem to $g(x) = -f(x)$, we see that this condition also identifies possible maximizing points. That is, solving $f'(x) = 0$ is necessary for finding a minimizing point for f, but not sufficient. Theorem 2.7.2 shows that a sufficient additional condition is $f''(x_*) > 0$.

THEOREM 2.7.2 Let $f \in C^2(D)$ for an open interval D and let $x_* \in D$ for which $f'(x_*) = 0$ and $f''(x_*) > 0$. Then there is some open subinterval $D' \subset D$ for which $x_* \in D'$ and $f(x) > f(x_*)$ for any other $x \in D'$.

Proof. Let D' be chosen about x_* so that $f'' > 0$ on D'. The use of a Taylor series with remainder does not lead to later problems in the multidimensional instance of the minimization problem, so we note that for any $x \in D'$ there is an $\bar{x} \in (x_*, x) \subset D'$ for which

$$f(x) - f(x_*) = f'(x_*)(x - x_*) + \tfrac{1}{2} f''(\bar{x})(x - x_*)^2.$$

Thus we are finished, since $f'(x_*) = 0$ and $\bar{x} \in D'$ imply

$$f(x) - f(x_*) = \tfrac{1}{2}(x - x_*)^2 f''(\bar{x}) > 0. \qquad \square$$

Now that we can recognize a solution, let us decide how to compute one. The easiest way to the class of algorithms we will use is to think of solving $f'(x) = 0$ by applying the hybrid Newton's method strategy we studied in Section 2.5, only making sure that we find a minimizer and not a maximizer by incorporating into the globalizing strategy the requirement that $f(x_k)$ decreases as k increases.

An iteration of the hybrid method starts by applying Newton's method, or a modification discussed in Section 2.6, to $f'(x) = 0$ from the current point x_c. The Newton step is

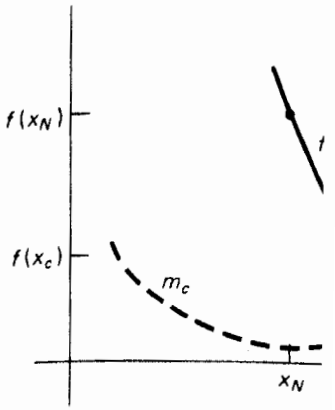
$$x_+ = x_c - \frac{f'(x_c)}{f''(x_c)}. \tag{2.7.1}$$

It is important to note the meaning of this step in terms of model problems. Since (2.7.1) is derived by making an affine model of $f'(x)$ around x_c, it is equivalent to having made a quadratic model of $f(x)$ around x_c,

$$m_c(x) = f(x_c) + f'(x_c)(x - x_c) + \tfrac{1}{2} f''(x_c)(x - x_c)^2,$$

and setting x_+ to the critical point of this model. A quadratic model is more appropriate than an affine model of $f(x)$ for either maximization or minimization because it has at most one extreme point. Thus the step (2.7.1) will find the extreme point of a quadratic function in one iteration; also, by exactly the same proof as in Theorem 2.4.3, it will converge locally and q-quadratically to an extreme point x_* of $f(x)$ if $f''(x_*) \neq 0$ and f'' is Lipschitz continuous near x_*.

Our global strategy for minimization will differ from that in Section 2.5 in that, rather than deciding to use the Newton point x_N by the condition $|f'(x_N)| < |f'(x_c)|$, which measures progress toward a zero of $f'(x)$, we will want $f(x_N) < f(x_c)$, which indicates progress toward a minimum. If $f(x_N) \geq f(x_c)$ but $f'(x_c)(x_N - x_c) < 0$, then Theorem 2.7.1 shows that $f(x)$ must initially decrease in the direction from x_c toward x_N, and so we can find an acceptable next point x_+ by backtracking from x_N toward x_c. From (2.7.1), $f'(x_c)(x_N - x_c)$ will be negative if and only if $f''(x_c)$ (or its approximation) is positive. That is, if the local model used to derive the Newton step has a minimum and not a



Local quadratic model's h
minimum implies $f(x)$ de
initially from x_c toward x

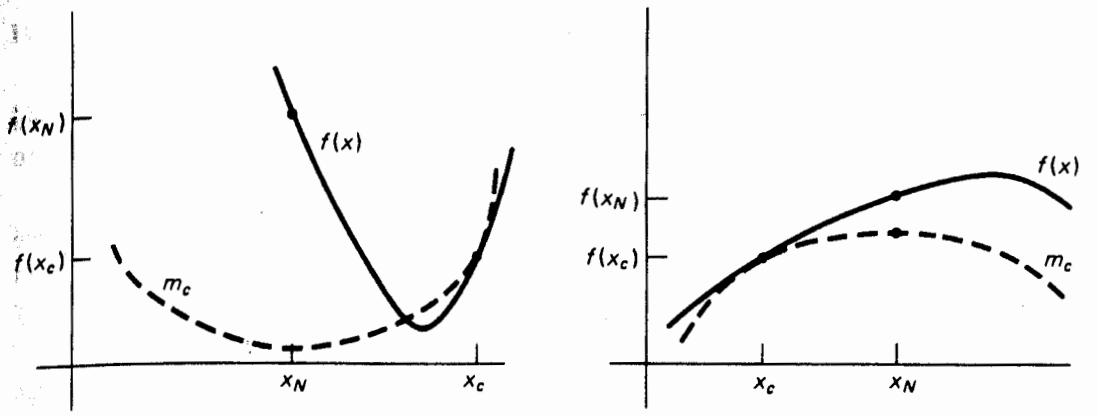
Figure 2.7.1 Corre
slope from x_c towar

maximum, then it is gu
Figure 2.7.1). On the ot
initially increases going
opposite direction. One
backtrack, if necessary,
for minimization are dis

The stopping crit
solving nonlinear equat
the problem?" or, "Ha
second question is teste
in Section 2.5, "Is $|x_+$
"Is $|f'(x_+)| < \tau_2$?" but
now also on the scale c
it: one can ask for the
value less than τ_2, or e
change in x,

to be less than τ_2 in abs
account for scaling in b

Finally, there is tl
available. The solution
defer consideration of
Exercise 21).



Local quadratic model's having minimum implies $f(x)$ decreases initially from x_c toward x_N

Local quadratic model's having maximum implies $f(x)$ increases initially from x_c toward x_N

Figure 2.7.1 Correspondence of the local quadratic model to the initial slope from x_c toward x_N

maximum, then it is guaranteed to provide an appropriate step *direction* (see Figure 2.7.1). On the other hand, if $f''(x_c) < 0$ and $f'(x_c)(x_N - x_c) > 0$, then $f(x)$ initially increases going from x_c toward x_N, and we should take a step in the opposite direction. One strategy is to try a step of length $|x_N - x_c|$ and then backtrack, if necessary, until $f(x_+) < f(x_c)$. More advanced global strategies for minimization are discussed in Chapter 6.

The stopping criteria for optimization are a bit different than those for solving nonlinear equations. Again one asks, "Have we approximately solved the problem?" or, "Have the iterates stayed in virtually the same place?" The second question is tested by the same relative/absolute step-size condition as in Section 2.5, "Is $|x_+ - x_c|/\max\{\text{typ}x, |x_+|\} < \tau_1$?" The first can simply be "Is $|f'(x_+)| < \tau_2$?" but once again this is dependent upon the scale of f, and now also on the scale of x. This time, however, something can be done about it: one can ask for the relative rate of change in f, $f'(x)/f(x)$, to have absolute value less than τ_2, or even for the relative change in f divided by the relative change in x,

$$\frac{f'(x_+)}{f(x_+)}\, x_+ ,$$

to be less than τ_2 in absolute value. The last test is appealing because it tries to account for scaling in both f and x.

Finally, there is the question of what to do if the derivatives $f'(x)$ are not available. The solution is so similar to the discussion in Section 2.6 that we defer consideration of this problem to the multidimensional case (see also Exercise 21).

2.8 EXERCISES

1. Carry out one iteration of Newton's method from $x = 2$ for the function $f_1(x)$ given in Section 2.1. Which root are you converging to? What happens if you start from $x = 1$?
2. Approximately how many iterations will the q-linearly convergent sequence $x_k = 1 + (0.9)^k$, $k = 0, 1, \ldots$, take to converge to 1 on a CDC machine? (Do this on paper, not on the computer!) Is q-linear convergence with constant 0.9 satisfactory for general computational algorithms?
3. Consider the sequence $x_k = 1 + 1/k!$, $k = 0, 1, \ldots$. Does this sequence converge q-linearly to 1? Does it converge q-linearly with any constant $c > 0$? What type of convergence is this? Does $\{x_k\}$ converge to 1 with q-order l for any constant $l > 1$?
4. Prove that the r-order of a sequence is always at least as large as its q-order. Give a counterexample to the converse.
5. Prove that $\{x_k\}$ has q-order at least p if and only if it has a sequence of error bounds $\{b_k\}$ with $b_k \geq e_k = |x_k - x_*|$ for which $\{b_k\}$ converges to zero with q-order at least p and
$$\overline{\lim} \frac{b_k}{e_k} < \infty.$$
6. (Harder) There is an interesting relationship between l-step q-order and "1-step" r-order. See if you can discover and prove this result. To find the result and an application see Gay (1979).
7. On the IBM 360-370 series, extended-precision multiplication is hardwired, but extended-precision division is done by doing double-precision division, and then using a Newton's method iteration that requires only multiplication and addition. To understand this, (a) derive a suitable Newton's method iteration for calculating $x_* = 1/a$ for a given real number a, and (b) carry out three iterations for $a = 9$ given $x_0 = 0.1$. What sort of convergence are you getting?
8. Analyze the convergence rate of the bisection method.
9. Write, debug, and test a program for solving one nonlinear equation in one unknown. It should follow the hybrid strategy explained in Section 2.5. For the local step, use either Newton's method, finite-difference Newton's method, or the secant method. For the global strategy, use bisection, backtracking, a strategy that interpolates between x_N and x_k, or another strategy of your own choosing. Run your program on:
   (a) $f(x) = \sin x - \cos 2x$, $x_0 = 1$
   (b) $f(x) = x^3 - 7x^2 + 11x - 5$, $x_0 = 2, 7$
   (c) $f(x) = \sin x - \cos x$, $x_0 = 1$
   (d) $f_2(x)$ given in Section2.1, $x_0 = 0, 2$.
   (If you use bisection, pick an appropriate initial interval for each problem, and constrain all your iterates to remain in that interval.) What rate of convergence do you observe in each case? It won't always be quadratic. Why?
10. Use your program from Exercise 9 to find the point $c > 0$ from which Newton's method for arc tan $x = 0$ will cycle (see Figure 2.4.1).

11. Modify your program from Exercise 9 to find the minimum of a function of one variable. (The modifications are explained in Section 2.7.)

12. What rate of convergence (to $x_* = 0$) does Newton's method have when solving $x^2 = 0$? $x^3 = 0$? How about $x + x^3 = 0$? $x + x^4 = 0$? Which assumptions of Theorem 2.4.3, if any, are violated in each case?

13. Prove that if $f(x_*) = f'(x_*) = 0, f''(x_*) \neq 0$, Newton's method converges q-linearly to x_* with $\lim_{k\to\infty} e_{k+1}/e_k = \frac{1}{2}$. [If you wish, generalize this result to $f(x_*) = f'(x_*) = \cdots = f^n(x_*) = 0, f^{n+1}(x_*) \neq 0$.]

14. Prove that if $f(x_*) = 0, f'(x_*) \neq 0, f''(x_*) = 0, f'''(x_*) \neq 0$, Newton's method converges q-cubicly to x_* [If you wish, generalize this result to $f'(x_*) \neq 0$, $f(x_*) = f^2(x_*) = \cdots = f^n(x_*) = 0, f^{n+1}(x_*) \neq 0$.]

15. Derive a cubicly convergent method for solving $f(x) = 0$ by modeling f by a quadratic Taylor series approximation at each step. What are the major problems in using this method? Do you think a quadratic model is appropriate for this problem? (See also "Müller's method" in most introductory numerical analysis texts.)

16. Given $f \in C^1(D)$, $x_c \in D$ with $f'(x_c) \neq 0$, and $d = -f(x_c)/f'(x_c)$, show that there exists some $t > 0$ such that $|f(x_c + \lambda d)| < |f(x_c)|$ for all $\lambda \in (0, t)$. [*Hint*: Choose t so that $f'(x_c + \lambda d) \cdot f'(x_c) > 0$ for all $\lambda \in (0, t)$, and then use the techniques of proof of Theorem 2.7.1.] (Assume $f(x_c)$ is not 0.)

17. Consider solving $x^2 = 0$ by Newton's method starting from $x_0 = 1$. Will any two successive iterates satisfy $|x_{k+1} - x_k|/\max\{|x_k|, |x_{k+1}|\} < 10^{-7}$? What about the test involving "typ$x$" suggested at the end of Section 2.6?

18. Given $f \in C^1(D)$ and $x_* \in D$ with $f(x_*) = 0$, show that there exists a constant $\alpha > 0$ and an interval $\hat{D} \subset D$ containing x_* such that for all $x \in \hat{D}$, $|f(x)| \leq \alpha |x - x_*|$. [*Hint*: Choose $\hat{D}$ so that $|f'(x)| \leq \alpha$ for all $x \in \hat{D}$.]

19. Suppose $f: D \to \mathbb{R}$ for an open interval D and assume $f' \in \text{Lip}_\gamma(D)$. Suppose that the computer routine for evaluating $f(x)$ has a total error bounded by $\eta |f(x)|$, where η is a nonnegative constant. Develop an upper bound on the total error in $[f(x_c + h_c) - f(x_c)]/h_c$ as an approximation to $f'(x_c)$, as a function of h_c [and γ, η, $f(x_c)$, and $f(x_c + h_c)$]. Find the value of h_c that minimizes this bound for fixed γ, η, and $f(x_c)$. [Assume here that $f(x_c + h_c) \cong f(x_c)$.] Also comment on the computational utility of finite differences if $f'(x)$ is extremely small in comparison to $f(x)$.

20. Suppose D is an open interval containing $x_c - h_c$, $x_c + h_c \in \mathbb{R}$, and let $f'': D \to \mathbb{R}$ satisfy $f'' \in \text{Lip}_\gamma(D)$. Prove that the error in the central difference formula (2.6.7) as an approximation to $f'(x_c)$ is bounded by $\gamma |h_c|^2/6$. [*Hint*: Expand the techniques of Lemma 2.4.2 to show that

$$f(z) - \left[f(x) + f'(x)(z - x) + \frac{f''(x)(z - x)^2}{2} \right] \leq \frac{\gamma |z - x|^3}{6},$$

or derive a similar bound from the Taylor series with remainder.]

21. Suppose you are solving a minimization problem in one variable in which $f(x)$ and $f'(x)$ are analytically available but expensive to evaluate, and $f''(x)$ is not analytically available. Suggest a local method for solving this problem.