Foundations of Numerical Analysis

G. Donald Allen  
Department of Mathematics  
Texas A&M University  
College Station, TX 77843-3368
Contents

2 Solution of Nonlinear Equations  
1 Bisection Method .................................. 1
1 Exercises ........................................... 10
1 Computer exercises ............................... 11
2 Newton’s Method and the Secant Method ........ 12
2 Exercises ........................................... 22
2 Computer Exercises ............................... 24
3 Polynomial Methods and Müller’s Method ....... 24
3 Exercises .......................................... 32
3 Computer Exercises ............................... 34
4 Iterative Methods .................................. 36
4 Exercises .......................................... 47
4 Computer Exercises ............................... 49
Chapter 2

Solution of Nonlinear Equations

In the design of digital filters for electronic circuits a transfer function is determined which has the desired properties. For example the function

\[ Z(s) = \frac{1}{s^3 + 2s^2 + 2s + 1} \]

is the transfer function of a third order Butterworth filter. To know for what values of \( s \) the denominator is zero is important toward determining what electronic components must be used to construct the filter. Similarly more complicated transfer functions have for example the form

\[ R(s) = \frac{a_0 + a_1 s + a_2 s^2 + \ldots + a_{10} s^{10}}{b_0 + b_1 s + b_2 s^2 + \ldots + b_{10} s^{10}}. \]

Again it is required to know when the numerator and denominator are zero. These are just two examples of problems which require solving an equation of the form \( f(x) = 0 \). The determination of solutions of such equations is the subject of this chapter.

1 Bisection Method

Roots and Approximate Roots

Let a continuous function \( f(x) \) be given. It is desired to solve the equation

(1.1) \[ f(x) = 0 \]

for \( x \). Any solution \( x \) of this equation is called a root. A number \( x \) at which the function \( f(x) \) is zero is called a zero of the function \( f(x) \). (In short equations have roots, functions have zeros.) Thus \( x = 2 \) is a root of the equation

\[ x^2 - 4 = 0 \]
and a zero of the function \( f(x) = x^2 - 4 \).

Consider the function \( f(x) \) with zero \( x = a \) with graph shown in Figure 1.1.
Now for most functions the value of the zero \( (x = a) \) cannot be determined exactly, to every decimal point. Therefore a criteria is needed by which we can claim that a zero (or root) has been found. To be precise, we say that \( x = a \) is an approximate root of (1.1) if given some \( \varepsilon > 0 \),

\[
|f(a)| < \varepsilon.
\]

(1.2) The value \( \varepsilon \) is called the tolerance.

**Iteration and Convergence**

Most methods for solving (1.1) for a particular function are algorithms that, beginning with some starting value or values, generate a sequence of numbers called iterates. If, for instance, there is one starting value, denote it by \( x_0 \). Iterates are then denoted by \( x_1, x_2, x_3, \ldots \). If \( \lim_{n \to \infty} x_n = a \) we say that iterates converge to the root \( a \).

![Figure 1.1:

We agree to stop the iteration if for some prescribed tolerance \( \varepsilon_1 \)

\[
|x_{n+1} - a| < \varepsilon_1.
\]

(1.3) However, since we don’t know the root \( a \) this test must be made indirectly.

For many methods we do not know at the onset \( \lim_{n \to \infty} x_n = a \). To establish this for a particular method and given function what we do is to show that for certain starting values there is a sequence of positive numbers \( E_n \) satisfying

\[
\max_n E_n = M < 1
\]

for which

\[
|x_{n+1} - a| < E_n|x_n - a|.
\]

(1.4)
This means that each successive iterate is closer to the root than the previous one by the factor $M$. Thus for example

$$|x_1 - a| < M|x_0 - a|$$

$$|x_2 - a| < M|x_1 - a| < M^2|x_0 - a|$$

and so on. Generally,

$$|x_{n+1} - a| < M^{n+1}|x_0 - a|.$$  \hspace{1cm} (1.5)

Since $M < 1$, $M^n \to 0$ and convergence is established. Moreover, if $x_0$ is known to be within a certain distance to the root, say $d$, then (1.3) can be satisfied by making $M^{n+1}d < \varepsilon_1$.

**Bisection Method**

The simplest algorithm for finding roots requires two starting values $x_0$ and $x_1$, which satisfy the criteria

$$f(x_0)f(x_1) < 0$$  \hspace{1cm} (1.6)

which means that the function values have opposite signs. Since $f(x)$ is assumed to be continuous then the Intermediate Value Theorem guarantees that a root of (1.1) lies somewhere between $x_0$ and $x_1$.

Label two new variables $x_L = x_0$ and $x_R = x_1$. Define

$$x_2 = (x_L + x_R)/2.$$  

Unless $f(x_2) = 0$, it must follow that exactly one of

$$f(x_2)f(x_L) < 0$$ \hspace{1cm} (1.7)(a)  

$$f(x_2)f(x_R) < 0$$ \hspace{1cm} (1.7)(b)

is true. (See Figure 1.1.) If (1.7)(a) holds set $x_R = x_2$; if (1.7)(b) holds set $x_L = x_2$. Then the new interval defined by $x_L$ and $x_R$ contains the root of the function and is half the size. Now define

$$x_3 = \frac{x_L + x_R}{2},$$

and proceed as above for $x_2$. Thus, if $f(x_3)f(x_L) < 0$ then $x_R = x_3$, otherwise $x_L = x_3$. Continue in this manner generating the sequence $x_2, x_3, x_4, \ldots$. This procedure, or algorithm, is called the *Bisection Method*.

**Example 1.** Apply the Bisection Method to the function $f(x) = x^2 - 3x + 2$. Determine $x_2, x_3$, and $x_4$. Use the starting values $x_0 = 1.5$ and $x_1 = 3.$
Solution. Note that \( f(1.5) = -0.2500 \) \( f(3.0) = 2.0000 \), so
\[
f(x_0)f(x_1) = (-0.2500)(2.0000) = -0.5000 < 0.
\]
Criteria (1.6) is met. Set
\[
x_L = 1.5 \quad \text{and} \quad x_R = 3.0.
\]
Calculate \( x_2 \) and \( f(x_2) \)
\[
x_2 = (1.5 + 3.0)/2 = 2.25
\]
\[
f(x_2) = 0.3125.
\]
Since \( f(x_L)f(x_2) < 0 \) we set \( x_R = x_2 \) for the next iteration. So,
\[
x_3 = (1.5000 + 2.2500)/2 = 1.8750
\]
\[
f(x_3) = -0.1094.
\]
Since \( f(x_R)f(x_3) < 0 \) set \( x_L = x_3 \) for the next iteration. So,
\[
x_4 = (2.2500 + 1.8750)/2 = 2.0625
\]
\[
f(x_4) = 0.0664.
\]
Error
For the Bisection Method, we know that the actual root lies between successive intervals and moreover these intervals halve in size each iteration. From this knowledge we can develop a rather precise estimate for the error of approximation. Although we do not know that the actual root lies between \( x_n \) and \( x_{n+1} \) it is easy to see that
\[
|x_{n+1} - a| < |x_{n+1} - x_n|.
\]
Since the intervals halve in size each iteration it follows that
\[
|x_2 - x_1| = \frac{1}{2}|x_1 - x_0|
\]
\[
|x_3 - x_2| = \frac{1}{2}|x_2 - x_1|
\]
\[
|x_{n+1} - x_0| = \frac{1}{2}|x_n - x_{n-1}|.
\]
Thus for example
\[
|x_3 - x_2| = \frac{1}{2^2}|x_1 - x_0|
\]
\[
|x_4 - x_2| = \frac{1}{2^3}|x_1 - x_0|
1. BISECTION METHOD

and generally

\[ |x_{n+1} - x_n| = \frac{1}{2^n} |x_1 - x_0|. \]

This together with (1.8) implies

(1.9) \[ |x_{n+1} - a| \leq \frac{1}{2^n} |x_1 - x_0|. \]

Thus, we can control exactly how close we are to the actual root, by counting the number of iterations we compute. Suppose the desired proximity to \( a \) is

\[ |x_{n+1} - a| < \varepsilon_1 \]

where \( \varepsilon_1 \) is some prescribed value. Then, solve the following equation for \( n \).

(1.10) \[
\frac{1}{2^n} |x_1 - x_0| = \varepsilon_1 \\
2^n = \frac{|x_1 - x_0|}{\varepsilon_1} \\
n \ln 2 = \ln(\frac{|x_1 - x_0|}{\varepsilon_1}) \\
n = \frac{\ln(\frac{|x_1 - x_0|}{\varepsilon_1})}{\ln 2}.
\]

Since this value may not be an integer take instead the smallest integer greater than the above quantity.

(1.11) \[ n = \left\lfloor \frac{\ln(\frac{|x_1 - x_0|}{\varepsilon_1})}{\ln 2} \right\rfloor + 1 \]

where \( \lfloor \cdot \rfloor \) denotes the greatest integer function.\(^1\) As is evident from (1.11) the number of iterations \( n \) depends directly on magnitude \( |x_0 - x_1| \).

Example 2. Consider the function \( f(x) = x^2 - 3x + 2 \) of Example 1. With the starting values \( x_0 = 1.5 \) and \( x_1 = 3 \), determine how many iterations are required to locate the root within a tolerance \( \varepsilon_1 = 10^{-4} \).

Solution. Substituting the appropriate values into (1.11) we obtain

\[ n = \left\lfloor \frac{\ln(3 - 1.5/10^{-4})}{\ln 2} \right\rfloor + 1 = \left\lfloor 13.87 \right\rfloor + 1 = 14. \]

We summarize this discussion with the Bisection Method algorithm.

\(^1\) For example, \([1.6] = 1, [-3.6] = -4\), etc.
**Bisection Method**

Given:
- \( x_0 \) and \( x_1 \) satisfy \( f(x_0)f(x_1) < 0 \)
- \( \varepsilon, \varepsilon_1 \)

set: \( n = 1, x_L = x_0, x_R = x_1 \)
set: \( m = \frac{\ln(|x_0 - x_1|/\varepsilon_1)}{\ln 2} + 1 \)
while \(|f(x_n)| > \varepsilon \) and \( n \leq m \) do:
  - \( x_{n+1} = \frac{x_L + x_R}{2} \)
  - if \( f(x_{n+1})f(x_L) < 0 \) set \( x_R = x_{n+1} \)
  - else set \( x_L = x_{n+1} \)
  - \( n = n + 1 \)
end while.
print \( x_n \)
stop

**Starting Values**

To determine the starting values \( x_0 \) and \( x_1 \) a search algorithm of some type must be employed. Usually the procedure consists of some theoretical work to determine an interval \([a, b]\) in which the zero(s) must lie. This is followed by a point-by-point search for two values \( x_0 \) and \( x_1 \) for which \( f(x_0)f(x_1) < 0 \). That is, given some small number, say \( \delta \), successive pairs of values \( a, a + \delta, a + 2\delta, a + 3\delta, \ldots \) are tested until two are found having function values with opposite signs.

**Example 3.** Locate an interval \([a, b]\) which contains the zeros of the function \( f(x) = e^x - 3x \).

**Solution.** Graphing the functions \( e^x \) and \( 3x \) separately (Figure 1.2) we see that they intersect in the interval \([0, 2]\). We may then take \( a = 0 \) and \( b = 2 \), and begin the search.

**Caution:** If \( \delta \) is too large it is possible to skip over a zero of the function completely.

**Regula Falsi Method**

The Bisection Method uses the most straightforward idea to finding the successive iterates, namely that the bisection. Since the function values at iterates must be computed we could use this information as well to compute the next iterate. A natural idea, given the two points \((x_L, f(x_L))\) and \((x_R, f(x_R))\) for which \( f(x_L)f(x_R) < 0 \), is to take for the next iterate the \( x \)-intercept of the line passing through the two points. See Figure 1.3.

To be more precise, relabel \( x_L = x_0 \) and \( x_R = x_1 \); suppose \( f(x_L)f(x_R) < 0 \); define

\[
(1.12) \quad x_2 = x_L - \frac{f(x_L)}{f(x_R) - f(x_L)}(x_R - x_L)
\]

Starting Values

To determine the starting values \( x_0 \) and \( x_1 \) a search algorithm of some type must be employed. Usually the procedure consists of some theoretical work to determine an interval \([a, b]\) in which the zero(s) must lie. This is followed by a point-by-point search for two values \( x_0 \) and \( x_1 \) for which \( f(x_0)f(x_1) < 0 \). That is, given some small number, say \( \delta \), successive pairs of values \( a, a + \delta, a + 2\delta, a + 3\delta, \ldots \) are tested until two are found having function values with opposite signs.

**Example 3.** Locate an interval \([a, b]\) which contains the zeros of the function \( f(x) = e^x - 3x \).

**Solution.** Graphing the functions \( e^x \) and \( 3x \) separately (Figure 1.2) we see that they intersect in the interval \([0, 2]\). We may then take \( a = 0 \) and \( b = 2 \), and begin the search.

**Caution:** If \( \delta \) is too large it is possible to skip over a zero of the function completely.

**Regula Falsi Method**

The Bisection Method uses the most straightforward idea to finding the successive iterates, namely that the bisection. Since the function values at iterates must be computed we could use this information as well to compute the next iterate. A natural idea, given the two points \((x_L, f(x_L))\) and \((x_R, f(x_R))\) for which \( f(x_L)f(x_R) < 0 \), is to take for the next iterate the \( x \)-intercept of the line passing through the two points. See Figure 1.3.

To be more precise, relabel \( x_L = x_0 \) and \( x_R = x_1 \); suppose \( f(x_L)f(x_R) < 0 \); define

\[
(1.12) \quad x_2 = x_L - \frac{f(x_L)}{f(x_R) - f(x_L)}(x_R - x_L)
\]
This is the $x$-intercept of the straight line joining $(x_L, f(x_L))$ and $(x_R, f(x_R))$. (See problem 21.) Again, as with the Bisection Method, either $f(x_L)f(x_2) < 0$ or $f(x_2)f(x_R) < 0$, unless of course $f(x_2) = 0$ which signals the end of the iteration. If $f(x_2)f(x_L) < 0$ then set $x_R = x_2$, otherwise set $x_L = x_2$. Now compute $x_3$ as the $x$-intercept of the line formed by $(x_L, f(x_L))$ and $(x_R, f(x_R))$, and continue as above, generating $x_2, x_3, x_4, \ldots$. This algorithm is called the

Figure 1.2:

Figure 1.3:
CHAPTER 2. SOLUTION OF NONLINEAR EQUATIONS

Regula Falsi method.

Example 4. Apply the Regula Falsi method to determine $x_2, x_3,$ and $x_4$ for the function $f(x) = x^2 - 3x + 2$ given $x_0 = 3/2$ and $x_1 = 3$.

Solution. $x_L = .5$ $x_R = 3$.

$$x_2 = 3/2 - f(1/2) \frac{3 - 3/2}{f(3) - f(1/2)} = 1.6667.$$ 

Continuing we generate the values tabulated below.

<table>
<thead>
<tr>
<th>$n$</th>
<th>$x_n$</th>
<th>$f(x_n)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1.5000</td>
<td>-2.500</td>
</tr>
<tr>
<td>1</td>
<td>3.0000</td>
<td>2.0000</td>
</tr>
<tr>
<td>2</td>
<td>1.6667</td>
<td>-0.2222</td>
</tr>
<tr>
<td>3</td>
<td>1.8000</td>
<td>-1.600</td>
</tr>
<tr>
<td>4</td>
<td>1.8899</td>
<td>-0.0988</td>
</tr>
<tr>
<td>5</td>
<td>1.9412</td>
<td>-0.0557</td>
</tr>
<tr>
<td>6</td>
<td>1.9697</td>
<td>-0.2094</td>
</tr>
</tbody>
</table>

See Fig. 1.4.

Error

Like the previous method, the *Regula Falsi* method converges to a root of $f(x) = 0$ which lies between $x_0$ and $x_1$. More importantly, it usually *converges faster*, but *not always*. However, it is easy to see from the calculation for $x_{n+1}$ that

$$|x_{n+1} - a| < |x_R - x_L|.$$ 

Thus the test

$$|f(x_{n+1})| < ?$$
1. **BISECTION METHOD**

together with
\[|x_R - x_L| < 1\]
are appropriate for this method.

**Regula Falsi Method.**

Given: \(x_0, x_1\) to satisfy \(f(x_0)f(x_1) < 0\)

\[\text{Set: } n = 1, x_L = x_0, x_R = x_1\]

for \(n = 1, 2, \ldots\) do
\[
x_{n+1} = x_L - f(x_L)(x_R - x_L)/(f(x_R) - f(x_L))
\]
if \(|f(x_{n+1})| < 0 \text{ and } |x_R - x_L| < 1\) then
print \(x_{n+1}\)
stop.
else
\[
\text{if } (f(x_{n+1})f(x_L)) < 0 \text{ set } x_R = x_{n+1}
\]
else set: \(x_L = x_{n+1}\)
end if
\[n = n + 1\]
end if

In summary, let us point out that the main advantage of the Bisection and Regula Falsi methods is that they are sure to locate an approximate root to \(f(x) = 0\). The disadvantages are that two starting values \(x_0\) and \(x_1\) must be found – this can be costly – and the convergence is rather slow.
CHAPTER 2. SOLUTION OF NONLINEAR EQUATIONS

1 Exercises

In problems 1–8 using the starting values $x_0$ and $x_1$ is given, compute $x_2, x_3,$ and $x_4$ using the bisection method.

1. $f(x) = x^2 - x$, 1.2  
2. $f(x) = x^3 - 6x^2 + 3$ 1,2  
3. $f(x) = e^{-x} - x$ 4. $f(x) = e^x - 4x$ 1,1  
5. $f(x) = x^2 - 2$ 6. $f(x) = x^2 + x - 1$ 0,1  
7. $f(x) = \cos x - x$ 8. $f(x) = \sin x - \cos x$ 0,1

In problems 9–16 determine graphically an interval that contains the zeros of the given function.

9. $f(x) = x^2 - 3x$ 10. $f(x) = x^2 - 3x - 7$  
11. $f(x) = e^x - x - 2$ 12. $f(x) = e^{-x} - 2x$  
13. $f(x) = \frac{x}{1+x^2}$ 14. $f(x) = \frac{x^2-1}{1+x^2}$  
15. $f(x) = \ln x + x$, $x > 0$ 16. $f(x) = \ln(x^2 - 4) + x$, $x > 2$

17. Show how the Bisection Method can be applied to determine the square root of any positive number $a$.

18. Show how the \textit{Regula Falsi} method can be applied to determine the cube root of any number $a$.

19. Suppose that $|x_0 - x_1| = 2$. Determine how many iterations are required to estimate a root with error less than $10^{-8}$.

20. Suppose the $|x_0 - x_1| < .1$. Determine how many iterations are required to estimate a root with error less than $10^{-6}$.

21. Verify formula (1.12) as the $x$-intercept of the line passing through $(x_L, f(x_L))$, $(x_R, f(x_R))$.

22. Consider the function $f(x) = x^{10} - 1/8$, which has a zero at $(1/8)^{1/10}$. Using a sketch of the graph of this function show graphically that Regula Falsi Method converges slower with starting values $x_0 = 0$ and $x_1 = 1$ than the Bisection Method.

In problems 23–28, using the starting values $x_0$ and $x_1$ as given, compute $x_2, x_3,$ and $x_4$, using Regula Falsi Method.

23. $f(x) = x^2 - 2x$ 1,3 24. $f(x) = x^2 - 5x + 4$ 3,5  
25. $f(x) = \sin x - 3 \cos x$ 0, $\pi/2$ 26. $f(x) = \tan x - 1$ 0, $\pi/3$  
27. $f(x) = e^x - x^2$ -1, 0 28. $f(x) = e^x - 6x$ 0,2
1. **Computer exercises**

I. Write a program to determine a zero of a given function using the Bisection Method. Use the given values \( x_0 \) and \( x_1 \) and tolerances \( \varepsilon \) and \( \varepsilon_1 \).

**Applications.** Use program I to find the zero of the functions given below together with \( x_0, x_1, \varepsilon, \) and \( \varepsilon_1 \) as given.

(a) \( f(x) = e^{-2x} - x, \) \( 1,0,10^{-4}10^{-4} \)
(b) \( f(x) = x^2 - 1 + 1nx \) \( 1/2, 2, 10^{-4}, 10^{-4} \)
(c) \( f(x) = \sin 3x - 2\cos x \) \( 0, \pi/4, 10^{-5}, 10^{-4} \)
(d) \( f(x) = x^3 + 4x - 1 \) \( 0, 1, 10^{-5}, 10^{-4} \)
(e) \( f(x) = x^4 + 2x^3 - 97x^2 - 266x + 360, \) \( 0, 2, 10^{-4}, 10^{-5} \)
(f) \( f(x) = x^5 - 27x^4 - 491x^3 + 15627x^2 - 49910x + 66000 \) \( 1, 10^{-4}, 10^{-4} \)

II. Write a search program to locate approximate values of the zeros of a given function \( f(x) \) by determining values \( x_0 \) and \( x_1 \) for which \( f(x_0)f(x_1) < 0 \). Search the given interval \([A, B]\) using a given increment size \( \delta \).

**Applications.** Use program II for the following functions with given interval \([A, B]\) and \( \delta \).

(a) \( f(x) = e^{-2x} - x, \) \([-1,1], .1 \)
(b) \( f(x) = \sin 2x, \) \([-5.1, 3.1], .5 \)
(c) \( f(x) = x - \tan x, \) \([-8, 8], .02 \)
(d) \( f(x) = x^2 \frac{1}{2} \cos x, \) \([-3, 3], .05 \)
(e) \( f(x) = x^3 - 2x^2 + 19x + 20, \) \([-5.6], .06 \)
(f) \( f(x) = x^3 - 1.5x^2 + .74x - .12, \) \([0,1], .05 \)

III. Write a program that combines program I and II as subroutines to determine the approximate roots of a given function in a given interval. Use tolerances \( \varepsilon, \varepsilon_1 \) and step size \( \delta \) as given.

**Applications.** Use program III to find the approximate zeros of the given functions. Search the given interval \([A, B]\) with given \( \delta \) and use given tolerances \( \varepsilon \) and \( \varepsilon_1 \).

(a) \( f(x) = x^2 - 7x + 10 \) \([10, 10], \delta = .1, \varepsilon = \varepsilon_1 = 10^{-5} \)
(b) \( f(x) = 2x - \tan x \) \([-\frac{2\pi}{7}, \frac{3\pi}{7}], \delta = .1, \varepsilon = \varepsilon_1 = 10^{-4} \)
(c) \( f(x) = e^{-x} - x/4 \) \([0, 2], \delta = .2, \varepsilon = \varepsilon_1 = 10^{-5} \)
(d) \( f(x) = x^3 - 2x^2 + 19x + 20 \) \([-5, 6], \delta = .2, \varepsilon = 10^{-4}, \varepsilon_1 = 10^{-5} \)
(e) \( f(x) = x^3 - 1.5x^2 + .74x - .12 \) \([0, 1], \delta = .02, \varepsilon = 10^{-5}, \varepsilon_1 = 10^{-4} \)
(f) \( f(x) = x^4 + 2x^3 - 97x^2 - 266x + 360 \) \([-12, 12], \delta = 1.0, \varepsilon = \varepsilon_1 = 10^{-5} \)

IV. Same as program III, using the **Regular Falsi** Method instead of the Bisection Method. Find the zeros of the same function as in program III.
2 Newton’s Method and the Secant Method

A prime criteria for selecting a root finding method is its speed of convergence. The Bisection Method is certainly reliable but there are methods that converge much faster. This section is devoted to two of them, the classical Newton’s Method and the relatively efficient Secant method.

Newton’s Method

Consider the function graphed in Figure 2.1, with zero at \( a \). Now select a single value \( x_0 \). Construct the straight line through \( (x_0, f(x_0)) \) with slope \( f'(x_0) \).

\[
y = f'(x_0)(x - x_0) + f(x_0).
\]

Define \( x_1 \) to be the \( x \)-intercept of this line; thus

\[
x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}. \tag{2.1}
\]

Determine \( x_2 \) from the point \( (x_1, f(x_1)) \) similarly, using the line with slope \( f'(x_1) \). Continue in this manner generating the sequence \( x_1, x_2, \ldots \). The general formula is

\[
x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \tag{2.2}
\]

This algorithm is called Newton’s Method. (See in Figure 2.2 the iterates \( x_1, x_2, x_3 \), for the function in Figure 2.1.)

The idea behind Newton’s Method is this: A function moves in the direction of its derivative (slope). So, if we follow the line tangent to the graph
2. NEWTON’S METHOD AND THE SECANT METHOD

Figure 2.2:

of the function at some point to the $x$-axis we should obtain a closer approximate to the root. Or more graphically put, we “shot” to the $x$-axis from some point on the graph using the derivative of the function as the direction in which we shoot. Successive applications of this idea generates the iterates.

**Example 1.** Let $f(x) = x^2 - 3x + 2$. Using $x_0 = 3$ apply Newton’s Method to calculate $x_1, x_2, x_3, x_4$.

**Solution.** We need $f'(x) = 2x - 3$. So, for example

$$x_1 = 3 - \frac{f(3)}{f'(3)} = 3 - \frac{20}{3} = 2.3333.$$ 

This and the other values found using (2.2) are tabulated below

<table>
<thead>
<tr>
<th>$n$</th>
<th>$x_n$</th>
<th>$f(x_n)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>3.0</td>
<td>20</td>
</tr>
<tr>
<td>1</td>
<td>2.33333</td>
<td>.444444</td>
</tr>
<tr>
<td>2</td>
<td>2.06667</td>
<td>.071111</td>
</tr>
<tr>
<td>3</td>
<td>2.00392</td>
<td>.003937</td>
</tr>
<tr>
<td>4</td>
<td>2.00002</td>
<td>.000016</td>
</tr>
</tbody>
</table>

□

The simplicity of this idea is both its strength and its weakness. That is to say, the iterates generated by Newton’s Method may not converge at all (to a root of $f(x) = 0$), but when they do converge they usually converge very, very fast. Some examples will clarify this. Consider the graph shown in Figure 2.3. Note that for the given starting value $x_0$ the first iterate $x_1$ is at relative minimum of the function and as such $f'(x_1) = 0$. Thus $x_2$ does not exist and the iteration stops. Even if $x_1$ is only very close to the relative minimum, the next iterate
may be very large as \( f'(x_1) \) is likely to be very small. Now consider the function graphed in Figure 2.4 which could for example be the graph of \( y = \arctan x \). Notice that at \( x = 0 \) the function is 0. However, with the starting value \( x_0 \) as shown, all the iterates \( x_1, x_2, \ldots \) exist but diverge to infinity. Even so, the reader may easily verify in both examples that if \( x_0 \) is selected close enough to the root the iterates do indeed converge to the root. This is a key point. Newton’s Method is more likely to converge when the starting value is sufficiently close to the root.

**Example 2.** Analyze graphically what can happen when applying Newton’s Method with various starting values to the function \( f(x) = x^2 + x^2 \).

**Solution.** The graph is shown in Figure 2.5. The relative (actually absolute) extrema are at \( x = 1 \), and the function is odd, i.e. \( f(-x) = -f(x) \). So whatever we conclude about convergence of the iterates for a positive starting value \( x_0 \) will apply to the negative starting value \(-x_0\). Note that if \( x_0 > 1 \) the sequence of iterates diverges to \( +\infty \). Hence, if \( x_0 < -1 \), the iterates diverge to \(-\infty\). If \( x_0 = 1 \), \( f'(x_0) = 0 \), and the iteration stops. Notice also that there is a value \( c \) such that if \( c < x_0 < 1 \) the iterates also diverge that is: \( x_1 > -c \), and \( x_2, x_3, \ldots \) diverge to \(-\infty\) as we have seen. If \( 0 \leq x_0 < c \) it is apparent that the iterates converge. The same applies to \((-1,-c)\) and \((-c,0)\). Thus, the iterates converge if \( |x_0| < c \) and diverge if \( |x_0| \geq c \). The value of \( c \) is determined as that starting value \( x_0 \) for which the first iterate (in Newton’s Method) is \( x_1 = -c \). Thus we solve

\[ -c = c - \frac{f(c)}{f'(c)}. \]

A simple calculation shows that \( c = 1/\sqrt{3} \). (See problem 3.)

**Simple and Higher Order Roots**

We say that the value \( x = a \) is a simple root of the equation \( f(x) = 0 \) if \( f(a) = 0 \) and \( f'(a) \neq 0 \). This means that the graph of \( f(x) \) is not tangent to the \( x \)-axis at \( x = a \). The roots \( x = a \) shown in Figures 2.1–2.5 are all simple. Simple roots are usually the easiest to find.

We say that the value \( x = a \) is a root of order \( m \) of the equation \( f(x) = 0 \) if \( f(a) = 0, f'(a) = 0, \ldots, f^{(m-1)}(a) = 0, \) and \( f^{(m)}(a) \neq 0 \). Thus simple roots are
first order roots. The equation \(x^2 = 0\) has a root of order 2 at \(x = 0\). Similarly if \(\kappa\) is a positive integer, \((x - 1)^\kappa = 0\) has a root of order \(\kappa\) at \(x = 1\). The function \(f(x) = e^x + e^{-x} - 2\) has a zero of order 2 at \(x = 0\) because \(f(0) = 0\), \(f'(x)|_{x=0} = e^x - e^{-x}|_{x=0} = 0\) but \(f''(x)|_{x=0} = e^x + e^{-x}|_{x=0} = 2\).

Suppose \(f(x)\) has a Taylor series at \(x = a\). Thus
\[
f(x) = f(a) + f'(a)(x - a) + \frac{f''(a)}{2!}(x - a)^2 + \ldots.
\]
If \(x = a\) is a root of order \(m\) then the first nonzero term of the Taylor series is \(f^{(m)}(a)/m!(x - a)^m\). For
\[
f(x) = f(a) + f'(a)(x - a) + \ldots + \frac{f^{(m-1)}(a)}{(m-1)!} + \frac{f^{(m)}(a)}{m!}(x - a)^m + \ldots
\]
\[
= \frac{f^{(m)}(a)}{m!}(x - a)^m + \frac{f^{(m+1)}(a)}{(m+1)!}(x - a)^{m+1} + \ldots.
\]
From this we conclude that if \(x = a\) is a root of order \(m\) then \(f(x)\) behaves very much like a multiple of the polynomial \((x - a)^m\) near \(x = a\). See Figure 2.6.

**Error**

We seek a formula of the form \(|x_{n+1} - a| \leq E_n|x_n - a|\), where \(\{E_n\}\) is some sequence. Suppose \(f(a) = 0\) and \(f'(a) \neq 0\); that is, \(x = a\) is a simple root.
Assume $x_0$ is given and that $f'(x_0) \neq 0$. Then

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

$$= \frac{[(x_0 - a)f'(x_0) - f(x_0)]}{f'(x_0)}$$

$$= \frac{[f(x_0) + f'(x_0)(a - x_0)]}{f'(x_0)}.$$  

From Taylor’s theorem with remainder, expanding at the point $x_0$

$$f(a) = f(x_0) + f'(x_0)(a - x_0) + \frac{1}{2} f''(c_0)(a - x_0)^2$$

where $c_0$ is between $x_0$ and $a$. Since $f(a) = 0$ it follows that

$$f(x_0) + f'(x_0)(a - x_0) = -\frac{1}{2} f''(c_0)(a - x_0)^2.$$  

So

(2.3) \[ x_1 - a = \frac{1}{2} \frac{f''(c_0)}{f'(x_0)} (x - a)^2. \]

By starting at the $n^{\text{th}}$ iterate $x_n$, a similar formula results

(2.4) \[ x_{n+1} - a = \frac{1}{2} \frac{f''(c_n)}{f'(x_n)} (x_n - a)^2 \]

where $c_n$ is between $x_n$ and $a$. Taking absolute values we obtain

$$|x_{n+1} - a| = \frac{1}{2} \left| \frac{f''(c_n)}{f'(x_n)} (x_n - a) \right|^2.$$  

This formula shows that the error of approximation of the $(n + 1)^{\text{st}}$ iterate depends on the square of the error of approximation of the $n^{\text{th}}$ iterate. For this reason Newton’s Method is said to converge quadratically. Correspondingly, the Bisection Method is said to converge linearly.

Because the denominator of the factor contains $f'(x_n)$ and we suppose $\lim_{n \to \infty} x_n = a$ it becomes apparent why the assumption $f'(a) \neq 0$ was made; namely, to bound the size of the factor $|f''(c_n)/f'(x_n)|$. Rewrite the formula (2.4) as

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

We obtain $|x_{n+1} - a| < |x_n - a|$ if

(2.5) \[ \frac{1}{2} \left| \frac{f''(c_n)}{f'(x_n)} (x_n - a) \right| < 1. \]

In general, (2.5) will be satisfied if $x_0$ is sufficiently close to the root $a$, confirming mathematically what was observed in Figures 2.3–2.5.
2. NEWTON'S METHOD AND THE SECANT METHOD

Figure 2.6:

(a) Double root

(b) Simple root

(c) Triple root
To illustrate the vast improvement of quadratic convergence over linear convergence we consider the following situations:

\[ |x_{n+1} - a| < \frac{1}{2} |x_n - a| \quad \text{Linear convergence} \]
\[ |\tilde{x}_{n+1} - a| < 2|\tilde{x}_n - a|^2 \quad \text{Quadratic convergence} \]

and assume the same starting value \( x_0 = \tilde{x}_0 \) and \( |x_0 - a| = .1 \). Bounds for \( |x_n - a| \) and \( |\tilde{x}_n - a| \) are tabulated below.

<table>
<thead>
<tr>
<th>( n )</th>
<th>( x_n - a )</th>
<th>( \tilde{x}_n - a )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>.1</td>
<td>.1</td>
</tr>
<tr>
<td>1</td>
<td>.050000</td>
<td>.020000</td>
</tr>
<tr>
<td>2</td>
<td>.025000</td>
<td>.000800</td>
</tr>
<tr>
<td>3</td>
<td>.012500</td>
<td>.000001</td>
</tr>
<tr>
<td>4</td>
<td>.006250</td>
<td>(&lt; 3.3 \times 10^{-12})</td>
</tr>
</tbody>
</table>

For Newton’s Method an estimate of the quantity \( |x_{n+1} - a| \) is difficult to achieve. However, it can be shown that the quantity \( |x_{n+1} - x_n| \) is a relatively good estimate \( |x_{n+1} - a| \). Therefore the convergence criteria for Newton’s Method are

(2.6) \[ |f(x_{n+1})| < \varepsilon \]
(2.7) \[ |x_{n+1} - x_n| < \varepsilon_1. \]

Because nonconvergence is a definite possibility, any programming implementation must account for this. A conservative estimate for the number of iterations required for convergence (for most functions) depends of course on \( \varepsilon \) and \( \varepsilon_1 \). Experimentation with a broad spectrum of functions and \( \varepsilon, \varepsilon_1 \) reveals that criteria (2.6) and (2.7) will be met in less than 25 iterations. So an iteration count limit \( N \) is specified. If the convergence criteria (2.6) and (2.7) are not met after \( N \) iterations the algorithm is stopped.

Newton’s Method

Given \( x_0, \varepsilon, \varepsilon_1 \) and \( N \) (the iteration counter limit)
set: \( n = 0 \)
while \( n \leq N \) do

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

if \(|f(x_{n+1})| < \varepsilon \) and \( |x_{n+1} - x_n| < \varepsilon_1 \) then

print \( x_{n+1} \)
stop.
else

\( n = n + 1 \)

end while.
Secant Method

The big advantage of Newton’s Method is its speed of convergence. Besides the possibility of nonconvergence the principal defect of the method is the requirement that both the function and its derivative be calculated at each step. This can be very costly, especially in multi-dimensional versions. A “cheap” alternative to Newton’s Method which avoids the calculation of the derivative is the **Secant Method** described as follows: Given two starting values \(x_0\) and \(x_1\) compute the difference quotient

\[
\frac{f(x_1) - f(x_0)}{x_1 - x_0}
\]

and use this in place of \(f'(x_1)\) in the Newton algorithm to compute \(x_2\). Thus

\[
x_2 = x_1 - \frac{f(x_1)}{\frac{f(x_1) - f(x_0)}{x_1 - x_0}}.
\]

(See Figure 2.7.)

Continue in this manner calculating \(x_{n+1}\) from \((x_n, f(x_n))\) and the difference quotient \((f(x_n) - f(x_{n-1}))/ (x_n - x_{n-1})\). This is the **Secant Method**:

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

or more compactly

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

Secant Method.
Note, two starting values \( x_0 \) and \( x_1 \) are required. And as with Newton’s Method proximity of \( x_0 \) and \( x_1 \) to the actual root \( x = a \) is required to guarantee convergence. Also important is the requirement that the root be simple. If these criteria are satisfied the convergence of the Secant method is fast, not as fast as the (quadratic) Newton’s method but substantially faster than any linearly converging method. Indeed what can be shown is that the error satisfies a formula of the form

\[
|x_{n+1} - a| \leq M|x_n - a|^\lambda
\]

where \( \lambda = (1 + \sqrt{5})/2 \approx 1.62 \), as \( n \to \infty \). The constant \( M \) depends on the ratio \( f''(a)/f'(a) \). This type of convergence is properly between linear and quadratic.

**Example 3.** Using \( x_0 = 3 \) and \( x_1 = 2.5 \) use the Secant Method to compute \( x_2, x_3 \) and \( x_4 \) for the function \( f(x) = x^2 - 3x + 2 \).

**Solution.** The iterate \( x_2 \) satisfies

\[
x_2 = 2.5 - \frac{f(2.5)}{f(2.5) - f(3.0)} (2.5 - 3.0) = 2.20000.
\]

The other values are tabulated below

<table>
<thead>
<tr>
<th>( n )</th>
<th>( x_n )</th>
<th>( f(x_n) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>3.2</td>
<td>2.</td>
</tr>
<tr>
<td>1</td>
<td>2.2</td>
<td>.750000</td>
</tr>
<tr>
<td>2</td>
<td>2.20000</td>
<td>.240000</td>
</tr>
<tr>
<td>3</td>
<td>2.05882</td>
<td>.062284</td>
</tr>
<tr>
<td>4</td>
<td>2.00934</td>
<td>.009428</td>
</tr>
</tbody>
</table>

The algorithm for the Secant Method is nearly identical to that for Newton’s Method, the same convergence criteria and iteration count tests are used. The differences are the requirements for two starting values and the replacement of derivative by the difference quotient.

**Secant Method**

Given: \( x_0, x_1 \in \varepsilon, \varepsilon_1 \) and \( N \), the iteration count limit

set: \( n = 1 \)

while \( n \leq N \) do

\[
x_{n+1} = x_n - f(x_n)(x_n - x_{n-1})/(f(x_n) - f(x_{n-1}))
\]

if \( |(f(x_{n+1})| < \varepsilon \) and \( |x_{n+1} - x_n| < \varepsilon_1 \) then

print \( x_{n+1} \)

stop

else

\( n = n + 1 \)

end if

end while.
In summary, the main advantages of Newton’s Method are the high speed of convergence and the requirement of only one starting value. The main disadvantage is that it does not converge unless the starting value is sufficiently close to the actual root. Also, convergence is much slower (if at all) for multiple roots. (See problem 29 for a multiple root version of Newton’s method.) Newton’s Method also requires two function evaluations for each iteration. The advantages of the Secant Method are fast convergence and one function evaluation per iteration. The main disadvantages is nonconvergence unless the starting values are sufficiently close to the actual root.
2 Exercises

1. Suppose \( f(x) \) is a linear function with nonzero slope. Show that by using any starting value \( x_0 \) Newton’s Method finds the \( x \)-intercept of \( f(x) \) in just one step.

2. Same as problem 1, using any two starting values.

3. Show that the value \( c \) in Example 2 is \( 1\sqrt{3} \).

4. Derive equation (2.1).

In problems 5–8 analyze graphically what can happen when applying Newton’s method for various starting values.

5. \( f(x) = x^2 - 3x + 2 \)

6. \( f(x) = x^2 - 1 \)

7. \( f(x) = xe^{-x^2} \)

8. \( f(x) = \frac{-x}{1+x^2} \)

In problems 9–16 compute the iterates \( x_1, x_2, \) and \( x_3 \) using Newton’s Method for the given function and starting value \( x_0 \).

9. \( f(x) = x^2 + x + 2 \) \quad \( x_0 = 2 \)

10. \( f(x) = x^2 + x - 2 \) \quad \( x_0 = -3 \)

11. \( f(x) = \sqrt{x} \) \quad \( x_0 = 1 \)

12. \( f(x) = x - \sqrt{x} \) \quad \( x_0 = 2 \)

13. \( f(x) = \sin x - 2 \cos x \) \quad \( x_0 = 0 \)

14. \( f(x) = \arctan x \) \quad \( x_0 = \frac{\pi}{2} \)

15. \( f(x) = e^x - x \) \quad \( x_0 = 5 \)

16. \( f(x) = e^{-x} + 2x \) \quad \( x_0 = 1 \)

17. Derive and simplify the formula for Newton’s Method (2.2) for the determination of the square root of any positive number \( a \). (Hint. The function \( f(x) = x^2 - a \) may be useful.)

18. The unique root of the equation \( f(x) = 1 - \frac{1}{x^3} = 0 \) is \( x = 1/a \). Derive and simplify (2.2) so that the resulting formula has no divisions.

In problems 19–24 determine the order of the given zero \( a \) for the given function.

19. \( f(x) = x^2 \) \quad \( a = 3 \)

20. \( f(x) = x^3 - 3x^2 + 3x - 1 \) \quad \( a = 1 \)

21. \( f(x) = \cos x \) \quad \( a = 0 \)

22. \( f(x) = \cos x - \sin x \) \quad \( a = \pi/4 \)

23. \( f(x) = x \sin x \) \quad \( a = 0 \)

24. \( f(x) = (x - 1)(e^{x-1} - 1) \) \quad \( a = 1 \)

25. Show that for the Secant Method the iterate \( x_{n+1} \) is the \( x \)-intercept of the line joining \((x_n, f(x_n))\) and \((x_{n-1}, f(x_{n-1}))\).

26. Another method, similar to the Secant method, called Steffenson’s Method, is given by

\[ x_{n+1} = x_n - \frac{f(x_n)}{d(x_n)} \]

where \( d(x) = |f(x + f(x)) - f(x)|/f(x) \). Apply this method to compute \( x_1, x_2, \) and \( x_3 \) for \( f(x) = \ln x \) and \( x_0 = 2 \).
27. A method due to Olver is that:

\[ x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \quad n = 0, 1, 2, \ldots \]

This method is cubically convergent for simple roots. Given \( x_0 \), compute \( x_1, x_2 \) and \( x_3 \) for the polynomial \( p(x) = x^3 + 2x - 1 \) using this method. (The unique zero of \( p(x) \), correct to 8 decimal places, is .45339765.)

28. Suppose that \( a \) is a simple zero of \( f(x) \). Suppose that \( f(x) \) is convex at \( a \) (i.e. \( f''(a) > 0 \)). Show graphically that all the iterates of Newton’s Method eventually lie either to the left or the right of \( x = a \).

29. Newton’s Method has been adapted to determine multiple roots with quadratic convergence. In fact, if \( x = a \) is an \( m^{th} \) order zero of \( f(x) \) then if \( x_0 \) is close enough to \( a \) the iterates defined by

\[ x_{n+1} = x_n - m \frac{f(x_n)}{f'(x_n)} \quad n = 0, 1, 2, \ldots \]

converge quadratically to \( a \).

(i) Show that \( f(x) = 1 - \cos x \) has a zero of order 2 at \( x = 0 \).

(ii) Apply the above formula with \( m = 2 \) and \( x_0 = \pi/4 \) to determine \( x_2, x_3 \) and \( x_4 \).

(iii) Prove (2.10). (Hint: If \( f(x) \) has a zero of multiplicity \( m \) at \( x = a \) then \( f'(x) \) has a zero of multiplicity \( m - 1 \) at \( x = a \). Use Taylor’s theorem.)
2 Computer Exercises

I. Write a program that implements Newton’s Method for a given function with a given starting value \( x_0 \) and with tolerances \( \varepsilon \) and \( \varepsilon_1 \). Check for divergence by putting a limit on the number of iterations \( N \). The function is programmed as a subroutine, the values \( x_0, \varepsilon, \varepsilon_1 \) and \( N \) are read from standard input.

Applications. Apply program I to the following functions \( f(x) \) with given starting values \( x_0 \), tolerances \( \varepsilon \) and \( \varepsilon_1 \) and with iteration count limit \( N \):

(a) \( f(x) = e^{-x}(x^2 - \sin x) - 1 \) \( x_0 = 1 \) \( \varepsilon = 10^{-4} \) \( \varepsilon_1 = 10^{-4} \) \( N = 10 \)
    \( x_0 = -10 \) \( \varepsilon = 10^{-4} \) \( \varepsilon_1 = 10^{-4} \) \( N = 10 \)
(b) \( f(x) = \sin x - 3 \cos x \) \( x_0 = 0 \) \( \varepsilon = 10^{-5} \) \( \varepsilon_1 = 10^{-5} \) \( N = 10 \)
    \( x_0 = 2\pi \) \( \varepsilon = 10^{-6} \) \( \varepsilon_1 = 10^{-6} \) \( N = 15 \)
(c) \( f(x) = \frac{x}{1 + x^4} \) \( x_0 = 3 \) \( \varepsilon = 10^{-4} \) \( \varepsilon_1 = 10^{-4} \) \( N = 15 \)
    \( x_0 = .5 \) \( \varepsilon = 10^{-5} \) \( \varepsilon_1 = 10^{-5} \) \( N = 15 \)
(d) \( f(x) = x^2 - 5x + 4 \) \( x_0 = 5 \) \( \varepsilon = 10^{-5} \) \( \varepsilon_1 = 10^{-5} \) \( N = 6 \)
    \( x_0 = 0 \) \( \varepsilon = 10^{-5} \) \( \varepsilon_1 = 10^{-5} \) \( N = 7 \)

II. Same as program I except with the Secant Method and given two starting values.

Applications. Apply program II to the above functions \( f(x) \) with given starting values \( x_0 \) and \( x_1 \), tolerances \( \varepsilon \) and \( \varepsilon_1 \), and with iteration limit \( N \).

III. Combine the search program of the previous section (§1, Program II) to locate neighborhoods of the zeros of the given function in the specified interval and then use one of the pairs of values found as the starting value of Newton’s Method of Program I.

Applications. Determine the approximate roots of the following functions in the given interval using the given values of \( \delta, A, B \), tolerances \( \varepsilon, \varepsilon_1 \), and iteration count limit \( M \):

(a) \( f(x) = \sin x - 2 \cos x \) \( \delta = .1 \) \([-3, 3] \) \( \varepsilon = \varepsilon_1 = 10^{-6} \) \( N = 25 \)
(b) \( f(x) = e^{-x} - \sin 2x \) \( \delta = .5 \) \([0, 4]\) \( \varepsilon = \varepsilon_1 = 10^{-5} \) \( N = 25 \)
(c) \( f(x) = x^3 - x \) \( \delta = .2 \) \([-5, 5] \) \( \varepsilon = \varepsilon_1 = 10^{-5} \) \( N = 20 \)
(d) \( f(x) = x^4 + 4x^3 - 68x^2 - 144x + 1152 \) \( \delta = .21 \) \([-10, 10] \) \( \varepsilon = \varepsilon_1 = 10^{-5} \) \( N = 20 \)
(e) \( f(x) = x^5 - 1.4x^4 + .33x^3 - .092x^2 - .0196x - .0024 \) \( \delta = .11 \) \([-2, 2] \) \( \varepsilon = \varepsilon = 10^{-4} \) \( N = 20 \)
(f) \( f(x) = x^6 - 14x^4 + 49x^2 - 36 \) \( \delta = .22 \) \([-5, 5] \) \( \varepsilon = \varepsilon_1 = 10^{-5} \) \( N = 15 \)

3 Polynomial Methods and Müller’s Method

When polynomials are the functions under consideration there are many methods, some quite old, for determining their zeros. For example, any of the
methods discussed thus far will work to determine real zeros of polynomials. Newton’s Method, used with complex arithmetic, will even find simple complex roots. However, there are specialized methods which work just for polynomials, and other methods suitable for polynomials and other functions as well. In this section we streamline Newton’s Method to determine roots of polynomials. In addition another quite reliable technique called Müller’s Method is formulated.

Begin with a polynomial \( p(x) \) of degree \( n \).

(3.1) \[ p(x) = a_n x^n + a_{n-1} x^{n-1} + \ldots + a_1 x + a_0. \]

Recall this fundamental result about polynomials:

\[ \text{Every polynomial of degree } n \text{ has at most } n \text{ (possibly complex) zeros.} \]

Our goal is to find all the zeros of a polynomial.

First of all, any iterative method to determine the roots of \( p(x) = 0 \) will require multiple evaluations of \( p(x) \). An easily programmable, highly efficient way to accomplish this uses \textit{nested multiplication} (also called Horner’s method). It is

(3.2) \[ p(x) = (((a_n x + a_{n-1}) x + a_{n-2}) \ldots + a_1) x + a_0. \]

For example, the nested form of \( p(x) = 4x^3 - 3x^2 + 7x - 2 \) is

\[ p(x) = ((4x - 3)x + 7)x - 2 \]

The key advantage of nested arithmetic for polynomial evaluation is that (3.2) requires only \( n \) multiplications whereas the evaluations of (3.1) directly, assuming \( x^j \) requires \( j - 1 \) multiplications, requires more than \( n^2/2 \) multiplications (see problem 1). The derivative \( p'(x) \),

\[ p'(x) = na_n x^{n-1} + (n-1)a_{n-1} x^{n-2} + \ldots + 2a_2 x + a_1 \]

can also be calculated using nested multiplication

(3.3) \[ p'(x) = (((na_n x + (n-1)a_{n-1}) x + \ldots + 2a_2) x + a_1. \]

\textbf{Deflation}

Next, assuming some starting value \( x_0 \) is given, Newton’s Method can be applied to find a root; call it \( a_1 \). Using the search method discussed in §1 it is possible to first determine crude approximate zeros of \( p(x) \) and then apply Newton’s Method directly using the approximates as starting values. However, the search procedure is very costly and locates only real roots and therefore should be avoided when possible. If, instead, a different starting value is prescribed then what can happen – and often does – is that the first root is found again! To
overcome this difficulty we factor the polynomial – numerically. That is, we consider for each value \( x \) a new polynomial

\[
p_1(x) = \frac{p(x)}{x - r_1}
\]

where \( r_1 \) is the first root determined. This process is known as deflation. The derivative of \( p_1(x) \) is

\[
p'_1(x) = \frac{(x - r_1)p'(x) - p(x)}{(x - r_1)^2}
\]

With nested arithmetic to compute \( p(x) \) and \( p'(x) \) Newton’s Method can be applied to find a root of \( p_1(x) = 0 \).

The process of deflation can be applied to each new approximate root successively. Suppose that the approximate roots \( r_1, r_2, \ldots, r_k \) have been determined, using the deflated polynomials \( p(x), p_1(x), p_{k-1}(x) \). Then

\[
p_k(x) = \frac{p(x)}{\prod_{j=1}^{k} (x - r_j)}
\]

Let \( r(x) = \prod_{j=1}^{k} (x - r_j) \). The derivative of \( p_k(x) \) is

\[
p'_k(x) = \frac{r(x)p'(x) - p(x)r'(x)}{(r(x))^2} = \frac{r(x)}{p(x)} \left[ p'(x) - p(x) \frac{r'(x)}{r(x)} \right].
\]

Repeated use of the product rule of differentiation and cancelling factors gives

\[
r'(x) \quad r(x) = \frac{1}{x - r_1} + \frac{1}{x - r_2} + \ldots + \frac{1}{x - r_k}.
\]

See problem 20. So,

\[
p'_k(x) = \frac{1}{\prod_{j=1}^{k} (x - r_j)} \left[ p'(x) - p(x) \sum_{j=1}^{k} \frac{1}{x - r_j} \right].
\]

This method of deflation, due to Maehly, is somewhat superior to the synthetic division version of deflation (problem 12) and can even be used for nonpolynomials.

---

2 The notation \( \prod_{j=1}^{k} b_j \) means the product \( b_1b_2\ldots b_k \). So \( \prod_{j=1}^{k} (x - r_j) = (x-r_1)(x-r_2)\ldots(x-r_k) \). See Appendix A2.
Newton’s Method with deflation for a polynomial

Given: $x_0, \varepsilon, \varepsilon_1,$ and $N$ (the iteration count limit)

Input: $n$, the degree of the polynomial

$a_n, a_{n-1}, \ldots, a_0$ the coefficients

For $k = 1, 2, \ldots, n$ do

set: $m = 0$

while $n \leq N$ do

$s = \sum_{j=1}^{k} \frac{1}{x_m - \text{roots}(j)}$

$p = \prod_{j=1}^{k} (x - \text{roots}(j))$

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

if ($|f(x_{n+1})| < \varepsilon$ and $|x_{n+1} - x_n| < \varepsilon_1$) then

roots($k + 1) = x_{n+1}$

else

$x_n = x_{n+1}$

end if

end while

print roots($j$), $j = 1, \ldots, k$

stop.

### Selecting a starting value

There are numerous theoretical tools for locating regions of the complex plane which contain roots of $p(x) = 0$. Some of them are easy to apply and can be used to select reasonable starting values $x_0$.

I. For the polynomial (3.1) define\(^3\)

$$r_1 = n |a_0/a_1|$$
$$r_n = |a_0/a_n|^{1/n}.$$  

Then at least one zero of $p(x)$ lies inside the circle, in the complex plane, of radius

$$r_0 = \min(r_1, r_n).$$

Thus a good starting value $x_0 = r_0/2$.

---

\(^3\)If $a_0 = 0$ then $x = 0$ is a root of $p(x)$. If $a_n = 0$ then, properly, $p(x)$ has degree $n - 1$.  

II. For the polynomial (3.1) define
\[ r = 1 + \max_{0 \leq k \leq n-1} |a_k/a_n|. \]

Then every zero lies inside the circle, in the complex plane of radius \( r \). An alternate starting value is \( x_0 = r/2 \).

**Example 1.** The roots of the polynomial \( p(x) = x^3 - 3x^2 - 94x + 360 \) are \(-10, 4\) and 9. Determine \( r_0 \) and \( r \).

**Solution.** \( a_3 = 1, a_2 = -3, a_1 = -94, \) and \( a_0 = 360 \). So \( r_1 = 3(360)/94 = 11.49 \) and \( r_3 = (360)^{1/3} = 7.11 \). Thus \( r_0 = 7.11 \). Also \( |a_0/a_3| = 360, |a_1/a_3| = 94, |a_2/a_3| = 3 \). So \( r = 361 \). As is evident the values \( r_0 \) and \( r \) are not necessarily sharp estimates for all polynomials. However, they are difficult to improve. For example the linear polynomial \( x - 360 \) gives \( r_0 = 360 \), the root exactly.

**Müller’s Method**

The idea of Newton’s Method was to use the \( x \)-intercept of a certain tangent line as the next iterate. The Secant Method used two points on the curve and the \( x \)-intercept of the line joining them because the next iterate. Generalizing this idea we could construct the parabola through any three points on the curve, determine its roots, and choose one of them as the next iterate. This is precisely the idea of Müller’s Method. Figure 3.1 shows the location of the next iterate (\( x_3 \)) as one of the zeros of the parabola through \((x_0, f(x_0)), (x_1, f(x_1))\), and \((x_2, f(x_2))\). To construct the algorithm we require some preliminary facts.

First of all, through any three points in the plane (with different \( x \)-values) there is a unique parabola. Consider the three points \((x_n, f(x_n)), (x_{n-1}, f(x_{n-1}))\), and \((x_{n-2}, f(x_{n-2}))\). Define the two quantities

\[
(3.8) \quad f[x_{n-2}, x_{n-1}, x_n] = \frac{f(x_n) - f(x_{n-1})}{x_n - x_{n-1}} - \frac{f(x_{n-1}) - f(x_{n-2})}{x_{n-1} - x_{n-2}},
\]

\[
(3.9) \quad f[x_{n-1}, x_n] = \frac{f(x_n) - f(x_{n-1})}{x_n - x_{n-1}}
\]

and let

\[ d_n = f[x_{n-1}, x_n] + f[x_{n-2}, x_{n-1}, x_n] \frac{(x_n - x_{n-1})}{(x_n - x_{n-1})}. \]

Now define the quadratic polynomial

\[
(3.10) \quad q(x) = f(x_n) + d_n(x - x_n) + f[x_{n-2}, x_{n-1}, x_n] (x - x_n)^2.
\]

It can be verified that \( q(x) \) passes through the above three points. (See problem 21.) The roots of \( q(x) = 0 \) are given by the quadratic formula

\[
(r - x_n) = \frac{-d_n \pm \sqrt{d_n^2 - 4f(x_n)f[x_{n-2}, x_{n-1}, x_n]}}{2f[x_{n-2}, x_{n-1}, x_n]}.\]
As this formula is inherently inaccurate we rationalize it to obtain the alternate form

\[(r - x_n) = \frac{-2f(x_n)}{d_n \pm \sqrt{d_n^2 - 4f(x_n)f[x_{n-2}, x_{n-1}, x_n]}}.\]

Now select the sign + or − so that the denominator is the largest and label the right hand side as \(\tilde{x}_{n+1}\). The next iterate \(x_{n+1}\) is defined to be

\[\tilde{x}_{n+1} = x_n + \tilde{x}_{n+1}.\]

Since complex numbers may result this algorithm must be programmed in variables defined as complex.

There are definite advantages to using this method. Foremost among them, it converges fast – almost quadratically – for simple roots. Indeed, in the limit as \(n \to \infty\)

\[|x_{n+1} - a| \leq M|x_n - a|^p \quad p \approx 1.84\]

where \(M\) depends on a power of the quantity \(\frac{1}{6}|f''(a)/f'(a)|\). (Of course, \(f(x)\) must be three times differentiable.) It does not require the calculation of a derivative but does require a square root. Most important, it does not require an initial approximation to the root, any three points can be used to start it. Another advantage of Müller’s Method is that it converges to multiple roots, however at a slower rate.

**Example 2.** Apply Müller’s Method to determine the iterates \(x_2, \ldots, x_6\) for the cubic \(p(x) = x^3 - 5x^2 + 9x - 5\) using the starting values \(x_0 = 2, x_1 = 3, \text{ and } x_2 = 4.\) (The actual zeros are \(2 \pm i\) and 1.)

**Solution.** The iterates are complex numbers and are tabulated below
CHAPTER 2. SOLUTION OF NONLINEAR EQUATIONS

To approximate all the roots, deflation can be applied for each successive approximate root. If the original or deflated polynomial is quadratic, (3.6) gives the root(s) in one step. The same standard tests used for Newton’s method are employed to stop the iteration for Müller’s Method. Finally, even if we know that there are only real roots, Müller’s Method may produce complex roots. However, the complex component is then quite small and can be neglected.

\[
\begin{array}{c|c|c}
 n & x_n & f(x_n) \\
0 & 2. & 1. \\
1 & 3. & 4. \\
2 & 4. & 15. \\
3 & 2.12500 + 0.048412i & 0.820312 + 0.514381i \\
4 & 2.06352 + 0.729224i & 0.434706 + 0.442927i \\
5 & 1.96730 + 0.920159i & 0.204698 + 0.083847i \\
6 & 1.99627 + 1.002794i & 0.01933 - 0.013042i \\
\end{array}
\]

To approximate all the roots, deflation can be applied for each successive approximate root. If the original or deflated polynomial is quadratic, (3.6) gives the root(s) in one step. The same standard tests used for Newton’s method are employed to stop the iteration for Müller’s Method. Finally, even if we know that there are only real roots, Müller’s Method may produce complex roots. However, the complex component is then quite small and can be neglected.

**Müller’s Method**

Given \(x_0, x_1, x_2, \varepsilon, \varepsilon_1, \text{ and } N\) (the iteration count limit)

set: \(n = 3\)

while \(n \leq N\)

calculate \(d_n = f[x_{n-1}, x_n] + f[x_{n-2}, x_{n-1}, x_n] (x_n - x_{n-1})\)

set: \(\tilde{x}_{n+1} = -2f(x_n)/(d_n \pm \sqrt{d_n^2 - 4f(x_n)f[x_{n-2}, x_{n-1}, x_n]})\)

where the sign is chosen to maximize the modulus of the denominator

set: \(x_{n+1} = x_n + \tilde{x}_{n+1}\)

if (\(|f(x_{n+1})| < \varepsilon\) and \(|x_{n+1} - x_n| < \varepsilon_1\)) then

print \(x_{n+1}\)

stop.

else

\(n = n + 1\)

end while.

stop.

**Additional Remarks**

1. The process of deflation can and does lead to errors in the approximation of roots, not so much for lower degree polynomials (degree \(\leq 10\)) but for those of higher degree. To locate roots with even finger precision, often times the purification algorithm is included in a root finding program. By this we mean that the set of approximate roots determined using deflation is used as starting values for Newton’s Method with the original polynomial.
2. For polynomials only, synthetic division can be used for deflation. That is, if \( a_1 \) is the first approximation root found, the coefficients of the polynomial \( p_1(x) = p(x)/(x - a_1) \) are determined by synthetic division. Newton’s Method is then applied to \( p_1(x) \). Continue this way, determining an approximate root and synthetically dividing, until all are found or no more can be found. The inherent problem with this method is that rather large errors may occur, because the remainders are continually dropped. Purification should definitely be used in this case.

3. There are numerous special methods for finding roots of polynomials. Included among them are Graffe’s root squaring technique [ ], Bairstow’s method [ ], and Laguerre’s method [ ]. The interested reader may pursue these classical methods in the indicated references.
CHAPTER 2. SOLUTION OF NONLINEAR EQUATIONS

3 Exercises

1. Assume that the evaluation of \( x^n \) requires \( n-1 \) multiplications. Determine the number of multiplications required to directly evaluate the polynomial
\[
p(x) = a_n x^n + a_{n-1} x^{n-1} + \ldots + a_1 x + a_0.
\]
(Hint. You may need the formula
\[
\sum_{j=1}^{n} j = n(n+1)/2.
\]

2. Show that the quadratic \( q(x) \) in (3.10) satisfies
\[
qu(x_n) = f(x_n), \quad q(x_{n-1}) = f(x_{n-1}) \quad \text{and} \quad q(x_{n-2}) = f(x_{n-2}).
\]

For the polynomials given in problems 3–8 determine \( r_0 \) and \( r_1 \).

3. \( p(x) = 17x^3 + 2x^2 - 30x + 4 \).

4. \( p(x) = 31x^4 - 29x + 2 \).

5. \( p(x) = x^6 + x^5 + x^4 - x^3 - x^2 + 1 \).

6. \( p(x) = 16x^4 + x^3 + x^2 - x + 12 \).

7. \( p(x) = x^6 + 100x + 1 \).

8. \( p(x) = x^7 - 1 \).

9. Consider the quadratic \( q(x) = ax^2 + bx + c \). Show that it has roots given by the formula
\[
\frac{-b \pm \sqrt{b^2 - 4ac}}{2c}
\]

10. Show that the reciprocals of the roots of \( ax^2 + bx + c \) are the roots of \( cx^2 + bx + a \). (Assume \( ac \neq 0 \).)

11. The polynomial \( p(x) = x^3 - 5x^2 + 6x \) has the root \( a = 2 \). Using long division to determine \( p_1(x) = p(x)/(x - 1) \) and find the other two roots of \( p(x) \).

12. Suppose that \( x = a \) is a root of the polynomial \( p(x) = a_n x^n + a_{n-1} x^{n-1} + \ldots + a_1 x + a_0 \), and that \( q(x) = p(x)/(x - a) \) has the form
\[
q(x) = b_{n-1} x^{n-1} + b_{n-2} x^{n-2} + \ldots + b_1 x + b_0.
\]
Show that the \( b_j \), \( n-1, \ldots, 1 \) can be determined by the recursion relation
\[
b_j = a_{j+1} - a b_{j+1}, \quad j = n-1, n-2, \ldots, 1,
\]
where we assume \( b_n = 0 \). (This is the formula for synthetic division.)

In problems 13–18 determine using Müller’s method the iterate \( x_3 \) for the given function \( f(x) \) and given starting values. Use four decimal places accuracy.
3. EXERCISES

13. $p(x) = x^4 - 10x^3 + 35x^2 - 50$ \quad -1, 0, 1
14. $p(x) = x^3 - 6x + 11x - 6$ \quad 1, 2, 3
15. $p(x) = x^2 + 2x + 17$ \quad 0, 1, 2
16. $p(x) = x^2 + x + 1$ \quad -1, 0, 1
17. $p(x) = x^3 - 7x + 17x - 15$ \quad -1, 0, 1
18. $p(x) = x^5 - 3x^4 - 23x^3 + 51x^2 + 94$ \quad -0, 1, 1

19. Show that for quadratics, Müller’s method is exact. That is, $x_3$ is an exact root.

20. Verify formula (3.6).
3 Computer Exercises

I. Write a program to calculate \( p(x) \) and \( p'(x) \) using nested multiplication. Input the variables: \( n \), the degree of the polynomial, \( a_0, a_1, \ldots, a_n \), the coefficients and the value(s) of \( x \).

**Applications.** Evaluate polynomials given below and their derivatives at the integer values \(-3, -2, -1, 0, 1, \) and 2.

(a) \( p(x) = x^3 + x^2 + x + 1 \)
(b) \( p(x) = 4x^4 - 6x^3 + 7x \)
(c) \( p(x) = x^6 - 17x^3 + x^2 - 1 \)
(d) \( p(x) = 25x^4 - 16x^3 + 32x^2 - 1 \)
(e) \( p(x) = 10x^{10} + 9x^9 + 8x^8 + \ldots + x \)

II. Using Newton’s Method together with nested arithmetic write a program to find approximate real roots of the polynomial \( x^6 - 14x^4 + 49x^2 - 36 \).

Use tolerances \( \varepsilon = \varepsilon_1 = 10^{-4} \) and the iteration count limit \( N = 25 \). (Remember to reset the iteration count after finding each root.) Use the same starting value \( x_0 = r_0/2 \), each time.

III. Write a program to find the approximate real roots of a given polynomial using Newton’s Method together with nested arithmetic and deflation. Input the variable: \( n \), the degree of the polynomial; \( a_0, a_1, \ldots, a_n \), the coefficients; \( \varepsilon \) and \( \varepsilon_1 \) to tolerances, and \( N \) the iteration count limit.

**Applications.** Find approximations to the real roots of the following polynomials using the given tolerances \( \varepsilon, \varepsilon_1 \) and iteration count limit \( N \). Use the same starting value \( x_0 = r_0/2 \) each time.

(a) \( x^3 + 4x^2 + x - 6 \) \( \varepsilon = \varepsilon_1 = 10^{-4} \) \( N = 15 \)
(b) \( x^3 - 3.6x^2 + 4.31x - 1.716 \) \( \varepsilon = \varepsilon_1 = 10^{-3} \) \( N = 15 \)
(c) \( x^4 - 2x^3 - 91x^2 + 272x - 180 \) \( \varepsilon - \varepsilon_1 = 10^{-4} \) \( N = 20 \)
(d) \( x^4 - 16x^3 + 89x^2 - 226x + 208 \) \( \varepsilon - \varepsilon_1 = 10^{-4} \) \( N = 20 \)
(e) \( x^6 - 1.3x^5 - 105.86x^4 + 131.78x^3 + 567.88x^2 - 178x + 12 \) \( \varepsilon - \varepsilon_1 = 10^{-5} \) \( N = 30 \)
(f) \( x^7 - 4x^6 + 8x^5 - 12x^4 + 13x^3 - 12x^2 + 6x - 4 \) \( \varepsilon - \varepsilon_1 = 10^{-5} \) \( N = 30 \)

IV. Write a program to find one approximate root of a given polynomial (or function) using Müller’s Method. Let \( \varepsilon = \varepsilon_1 = 10^{-5}, N = 15 \), and \( x_0 = 1, x_2 = 2 \) and \( x_3 = 3 \). Remember to use complex arithmetic.

**Applications.** Determine one approximate root of the following functions.
(The answer may be complex.)
3. COMPUTER EXERCISES

(a) \( p(x) = x^3 - 8x^2 + 22x - 20 \)
(b) \( p(x) = x^4 + x^2 - 102x^2 - 100x + 200 \)
(c) \( f(x) = 2 \cos x - \sin x \)
(d) \( f(x) = 2x^2 - \sin 3x \)

V. Write a program to find all of the roots of a given polynomial using Müller’s Method together with nested multiplication and (complex) deflation. Input the variables: \( n \), the degree of the polynomial; \( a_0, a_1, \ldots, a_n \), the coefficients; \( \varepsilon \) and \( \varepsilon_1 \), the tolerances; \( N \), the iteration count limit; \( x_0, x_1, \) and \( x_2 \) the starting values.

Applications. Determine all the roots of the following polynomials using the given input values \( \varepsilon, \varepsilon_1, N, x_0, x_1, x_2 \).

(a) \( x^3 - 12x^2 + 47x - 60 \) \( \varepsilon = \varepsilon_1 = 10^{-5} \), \( N = 10, 1, 2, 3 \)
(b) \( x^4 - 9.1x^3 + 26.7x^2 - 5.2x - 42 \) \( \varepsilon = \varepsilon_1 = 10^{-5} \), \( N = 15, -1, 0, 1 \)
(c) \( x^4 - 10.6x^3 + 16.11x^2 - 1.106x - .06 \) \( \varepsilon = \varepsilon_1 = 10^{-4} \), \( N = 20, 0, 1, 2 \)
(d) \( x^7 + x^6 + x^5 + 4x^3 + 4x^2 + 4x + 4 \) \( \varepsilon = \varepsilon_1 = 10^{-5} \), \( N = 20, 0, 1, -1 \)
(e) \( x^6 - x^5 - 34x^4 + 34x^3 + 225x^2 - 225x \) \( \varepsilon = \varepsilon_1 = 10^{-5} \), \( N = 25, 6, 7, 8 \)
4 Iterative Methods

Suppose that the equation \( f(x) = 0 \) can be written in the form

\[ x = g(x) \]  

(4.1)

in such a way that any root of (4.1) is also a root of \( f(x) = 0 \). For example, the by now familiar quadratic equation \( x^2 - 3x + 2 = 0 \) can be written in the forms

\[
\begin{align*}
    x &= \frac{1}{3}x^2 + \frac{2}{3} \\
    x &= 3 - \frac{2}{x} \\
    x &= \sqrt{3x - 2} \\
    x &= x - \frac{1}{4}(x^2 - 3x + 2).
\end{align*}
\]

Equation (4.1) is a convenient form for defining an iterative scheme. Begin with a starting value \( x_0 \). Define

\[
\begin{align*}
x_1 &= g(x_0) \\
x_2 &= g(x_1)
\end{align*}
\]

and so on. The iterative method is

\[ x_{n+1} = g(x_n) \quad n = 0, 1, \ldots, \]

(4.2)

If the sequence \( \{x_n\} \) converges to some value \( a \) then \( a = g(a) \), and so \( a \) is a root of (4.1) and hence \( f(a) = 0 \). This method can be very effective for finding roots of \( f(x) = 0 \), provided the function \( g(x) \) behaves properly. Consider the following example.

**Example 1.** Let \( f(x) = x^2 - 3x + 2 \). Write \( x^2 - 3x + 2 = 0 \) as

\[
\begin{align*}
x^2 &= 3x - 2 \\
x &= 3 - \frac{2}{x}.
\end{align*}
\]

Letting \( x_0 = 1.25 \) we tabulate the iterates below

<table>
<thead>
<tr>
<th>( n )</th>
<th>( x_n )</th>
<th>( g(x_n) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1.2500</td>
<td>1.4000</td>
</tr>
<tr>
<td>1</td>
<td>1.4000</td>
<td>1.5714</td>
</tr>
<tr>
<td>2</td>
<td>1.5714</td>
<td>1.7273</td>
</tr>
<tr>
<td>3</td>
<td>1.7273</td>
<td>1.8421</td>
</tr>
<tr>
<td>4</td>
<td>1.8421</td>
<td>1.9431</td>
</tr>
<tr>
<td>\vdots</td>
<td>\vdots</td>
<td>\vdots</td>
</tr>
<tr>
<td>10</td>
<td>1.9971</td>
<td>1.9985</td>
</tr>
</tbody>
</table>
These iterates seem to be converging to 2, a root of the equation \( f(x) = 0 \). Now manipulate \( x^2 - 3x + 2 = 0 \) as

\[
3x = x^2 + 2 \\
x = \frac{1}{3}x^2 + \frac{2}{3}.
\]

Again let \( x_0 = 1.2500 \). The iterates are tabulated below

<table>
<thead>
<tr>
<th>( n )</th>
<th>( x_n )</th>
<th>( g(x_n) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1.2500</td>
<td>1.1875</td>
</tr>
<tr>
<td>1</td>
<td>1.1875</td>
<td>1.1367</td>
</tr>
<tr>
<td>2</td>
<td>1.1367</td>
<td>1.0974</td>
</tr>
<tr>
<td>3</td>
<td>1.0974</td>
<td>1.0681</td>
</tr>
<tr>
<td>4</td>
<td>1.0681</td>
<td>1.0469</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>10</td>
<td>1.0066</td>
<td>1.0044</td>
</tr>
</tbody>
</table>

These iterates seem to be converging to 1, the other root of \( f(x) = 0 \). However, if we select another starting point, say \( x_0 = 3 \) the iterates diverge, as shown below.

<table>
<thead>
<tr>
<th>( n )</th>
<th>( x_n )</th>
<th>( g(x_n) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>3.0000</td>
<td>3.6667</td>
</tr>
<tr>
<td>1</td>
<td>3.6667</td>
<td>5.1481</td>
</tr>
<tr>
<td>2</td>
<td>5.1481</td>
<td>9.5011</td>
</tr>
<tr>
<td>3</td>
<td>9.5011</td>
<td>30.7572</td>
</tr>
<tr>
<td>4</td>
<td>30.7572</td>
<td>316.0025</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
The best way to understand why and when the method (4.2) converges is to use graphical analysis. First note that a solution of \( x = g(x) \) is a point where the graphs of the functions \( y = x \) and \( y = g(x) \) intersect. See Figure 4.1. For a given starting value \( x_0 \), the iterates can be determined graphically as shown in Figure 4.2, where they appear to be converging to \( x = a \). Figure 4.3 illustrates three other examples in two of which the iterates do not converge (a) and (c). The reason for nonconvergence in (a) and (c) is not because \( x_0 \) was not chosen close enough to \( a \), but rather because the function \( g(x) \) does not behave properly: namely, the magnitude of the derivative \( g'(x) \) is too large.

To be more precise, suppose that \( g(a) = a \), and that \( g'(x) \) is continuous in an interval centered at \( a \), \( |x - a| \leq r \). Suppose that there is a number \( M < 1 \) so that \( |g'(x)| < M < 1 \) in this interval. Select \( x_0 \) in the interval. Then we have

\[
x_1 = g(x_0)
\]
\[ x_1 - a = g(x_0) - a \]
\[ x_1 - a = g(x_0) - g(a). \]

By the mean value theorem there is a constant \( c_0 \) between \( x_0 \) and \( a \) so that
\[ g(x_0) - g(a) = g'(c_0)(x_0 - a). \]

Therefore
\[ |x_1 - a| = |g'(c_0)||x_0 - a|. \]

Since \( |g'(c_0)| < M < 1 \) (by assumption) we conclude that \( x_1 \) is closer to \( a \) than \( x_0 \). If, on the other hand, \( |g'(x)| > 1 \) in the interval the iterate \( x_1 \) is farther away from \( a \) than \( x_0 \). This same sort of analysis can be applied to each iterate. Thus
\[ x_{n+1} = g(x_n) \]
\[ x_{n+1} - a = g(x_n) - g(a) \]
\[ x_{n+1} - a = g'(c_n)(x_n - 1) \quad (4.3) \]

by the mean value theorem. Thus
\[ |x_{n+1} - a| = |g'(c_n)||x_n - a|. \quad (4.4) \]

Now define
\[ M = \max_{|x-a| \leq r} |g'(x)|. \]

Then
\[ |x_{n+1} - a| < M|x_n - a|. \quad (4.5) \]

Applying this formula repeatedly we obtain
\[ |x_n - a| < M^n|x_0 - a|. \]

Since \( M < 1 \), the sequence \( M^n \) converges to zero and so we conclude that the iterates converge to \( a \); i.e. \( \lim_{n \to \infty} x_n = a \). We summarize all this as follows:

If \( a \) is a root of the equation \( x = g(x) \), if
\[ |g'(x)| < M < 1 \] in some interval \( |x-a| < r \), and if \( x_0 \) is chosen in this interval, then the iterates \( x_n \) defined by \( x_{n+1} = g(x_n), n = 0, 1, \ldots \) converge to \( a \).

A root \( a \) of (4.1) for which \( |g'(a)| < 1 \) is often called a point of attraction.

**Example 2.** Let \( g(x) = 3 - 2/x \), as in Example 1. Show that if \( x_0 \) is near enough to \( a = 2 \) the iterates converge to 2, and that no matter how close \( x_0 \) is to \( a = 1 \) the iterates do not converge to 1.
4. ITERATIVE METHODS

Solution. To apply the analysis above we require the derivative $g'(x) = \frac{2}{3x}$.

Now $g'(2) = 3/4$. Thus by the continuity of $g'(x)$ there is an interval centered at 2, $|x - 2| < r$, for which $|g'(x)| < M < 1$, given any $3/4 < M < 1$. If $x_0$ is in this interval the iterates $\{x_n\}$ converge to 2. On the other hand, since $g'(1) = 2$ we see that no matter how close $x_0$ is to 1 the next iterate $x_1$ will be farther away. Thus convergence to 1 is impossible. See Figure 4.4.

Consider again the equation (4.4). That the method exhibits at least linear convergence when $|g'(x)| < M < 1$ near $a$ follows because as $n \to \infty$

\[
\text{Figure 4.4:}
\]

then $x_n \to a$ and hence $c_n \to a$ (by the Sandwich theorem). Thus we have

\[
x_{n+1} - a = g'(a)(x_n - a).
\]

Notice that the smaller $|g'(a)|$ is the faster the iterates converge. In fact, if $g'(a) = 0$ we can argue that

\[
g(x_n) - g(a) = \frac{g''(a)}{2}(x_n - a)^2
\]

by Taylor’s theorem, where $d_n$ is between $x_n$ and $a$. Hence when $g'(a) = 0$

\[
x_{n+1} - a = \frac{g''(a)}{2}(x_n - a)^2
\]

yielding quadratic convergence.

As with most of the previous methods there is no practical way to determine how close $x_n$ is to the actual root $a$. So the test

\[
|x_{n+1} - x_n| < \varepsilon
\]

is commonly used. The second test $|f(x_n)| < \varepsilon_1$ cannot be used here unless $f(x)$ is known. The algorithm is quite simple.
Algorithm: Iterative Method
Given: \( x_0, \varepsilon, N \), the iteration count limit
Set: \( n = 0 \)
while \( n < N \)
\[ x_{n+1} = g(x_n) \]
if \( |x_{n+1} - x_n| < \varepsilon \) then stop
else
\[ n = n + 1 \]
end if.

Accelerating Convergence
Like the Bisection Method, iterative methods usually converge very slowly. Fortunately, a method is available to “speed up” convergence. Called the Aitken’s Acceleration, it can be applied to any iteratively generated sequence and has a very simple geometric description. Refer to Figure 4.5. Let \( x_0 \) be given. Compute \( \hat{x}_1 = g(x_0) \). Construct the line through the points \((x_0, \hat{x}_1)\) and \((\hat{x}_1, g(\hat{x}_1))\). Define the next iterate \( x_1 \) to be the \( x \)-value where this line intersects the line \( y = x \). The formula for \( x_1 \) is given by

\[
(4.8) \quad x_1 = x_0 - \frac{\hat{x}_1 - x_0}{g(\hat{x}_1) - 2\hat{x}_1 + x_0}
\]

(See problem 14.)

Generally, starting at the \( n \)th iterate \( x_n \), compute the intermediate values \( \hat{x}_{n+1} = g(x_n) \) and \( g(\hat{x}_{n+1}) \). Define the next iterate \( x_{n+1} \) to be the \( x \)-value of the intersection of the line \( y = x \) and the line passing through the two points \((x_n, \hat{x}_{n+1})\) and \((\hat{x}_{n+1}, g(\hat{x}_{n+1}))\). This gives

\[
(4.6) \quad x_{n+1} = x_n - \frac{\hat{x}_{n+1} - x_n}{g(\hat{x}_{n+1}) - 2\hat{x}_{n+1} + x_n}, \hat{x}_{n+1} = g(x_n)
\]

This sequence of iterates converges faster than the iterates constructed by (4.2). In particular for simple roots (of \( f(x) = 0 \)), the convergence is at least quadratic. Note, however, that for each step two function evaluations are required in (4.9) compared with just one function evaluation in (4.2).

**Example 3.** Let \( g(x) = 3 - 2/x \) and \( x_0 = 1.25 \). Applying the Aitken’s acceleration method we tabulate the first five iterates below
Table 4.5:

<table>
<thead>
<tr>
<th>$n$</th>
<th>$x_n$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1.2500</td>
</tr>
<tr>
<td>1</td>
<td>0.2000</td>
</tr>
<tr>
<td>2</td>
<td>-2.7647</td>
</tr>
<tr>
<td>3</td>
<td>2.6679</td>
</tr>
<tr>
<td>4</td>
<td>2.0418</td>
</tr>
<tr>
<td>5</td>
<td>2.0004</td>
</tr>
</tbody>
</table>

**Aitken’s Acceleration.**

Given: $x_0$, $\varepsilon$, and $N$ the iteration and count limit

Set: $n = 1$

while $n \leq N$ do

\[ \hat{x}_{n+1} = g(x_n) \]

\[ x_{n+1} = x_n - \frac{(\hat{x}_{n+1} - x_n)^2}{g(\hat{x}_{n+1}) - 2\hat{x}_{n+1} + x_n} \]

if $(|x_{n+1} - x_n| < \varepsilon)$ then

\[ x_{n+1} \]

print $x_{n+1}$

stop.

else

\[ n = n + 1 \]

end if.

end while

stop.

Figure 4.5:
CHAPTER 2. SOLUTION OF NONLINEAR EQUATIONS

Figure 4.6:

Quasi Newton methods

Recall that for Newton’s Method \( x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \) the main disadvantage was the requirement of two function evaluations per step. This can be quite costly, especially in higher dimensional formulations. The Secant Method is an attempt to avoid the derivative evaluation by substituting for it a difference quotient. As such it is called a \textit{quasi-Newton method}.

Consider the following function

\begin{equation}
    g_m(x) = x - \frac{f(x)}{m}
\end{equation}

where \( m \) is a constant. The induced iterative method is

\begin{equation}
    x_{n+1} = x_n - \frac{f(x_n)}{m}.
\end{equation}

It is easy to show that the line constructed through the point \((x_n, f(x_n))\) with slope \( m \) has \( x_{n+1} \) as its \( x \)-intercept (Problem 31). See Figure 4.6. This method is also called a quasi-Newton method. Recall equation (4.6). For \( g_m(x) = x - f(x)/m \) we have the approximate rate of convergence

\[ g_m'(a) = 1 - f'(a)/m. \]

Selecting \( m \) so that \( |g_m'(a)| < 1 \) will guarantee that the method (4.8) converges. And if \( m = f'(a) \) is chosen the convergence is at least quadratic. Of course, \( a \) is not known and in practice this is impossible. However, if \( x_0 \) is known to be an approximate root and \( f'(x) \) does not vary too much, then a good choice for \( m \) is \( f'(x_0) \).

**Example 4.** Consider the function \( f(x) = x^3 + 2x - 1 \). Determine a starting value \( x_0 \) near a root and apply the quasi-Newton method (4.8) using \( m = f'(x_0) \). Compute \( x_1, x_2, \ldots, x_6 \).
4. ITERATIVE METHODS

Solution. Graphing \( f(x) \) we see that a root lies between 0 and 1 (Figure 4.7). So take \( x_0 = 1 \). From \( f'(x) = 3x^2 + 2 \), compute \( f'(1) = 5 \). With \( m = 5 \) the iterates are computed from

\[
x_{n+1} = x_n - \frac{1}{5} (x_n^3 + 2x_n - 1)
\]

and are tabulate below

<table>
<thead>
<tr>
<th>( n )</th>
<th>( x_n )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1.0000</td>
</tr>
<tr>
<td>1</td>
<td>.6000</td>
</tr>
<tr>
<td>2</td>
<td>.5168</td>
</tr>
<tr>
<td>3</td>
<td>.4825</td>
</tr>
<tr>
<td>4</td>
<td>.4670</td>
</tr>
<tr>
<td>5</td>
<td>.4598</td>
</tr>
<tr>
<td>6</td>
<td>.4565</td>
</tr>
</tbody>
</table>

To obtain the actual root correct to 8 decimal place (.45339765) requires 25 iterations. Using the closer starting value \( x_0 = .5 \) we obtain the same value after only 6 iterations.

A compromise between Newton’s Method and the quasi-Newton method (4.8) is to “update” \( m \) every few steps with a new \( m \), computed from the derivative \( f'(x) \). For example, updating every second step gives the two step method with iterates

\[
x_1 = x_0 - f(x_0)/f'(x_0)
\]
CHAPTER 2. SOLUTION OF NONLINEAR EQUATIONS

Figure 4.8:

\[ y = f(x) \]

\[ \text{slope} = f'(x_2) \]

\[ \text{slope} = f'(x_0) \]

\[ x_2 = x_1 - \frac{f(x_1)}{f'(x_0)} \]

\[ x_3 = x_2 - \frac{f(x_2)}{f'(x_2)} \]

\[ x_4 = x_3 - \frac{f(x_3)}{f'(x_2)}. \]

See Figure 4.8.

Generally, for the two step update the formulas are:

\[
\begin{align*}
x_{2n+1} &= x_{2n} - \frac{f(x_{2n})}{f'(x_{2n})} \\
x_{2n+2} &= x_{2n+1} - \frac{f(x_{2n+1})}{f'(x_{2n})}
\end{align*}
\]

Using \( x_0 = 1 \) for \( f(x) = x^3 \times 3x + 1 \) as in Example 4 we have the following iterates

<table>
<thead>
<tr>
<th>( n )</th>
<th>( x_n )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1.000000</td>
</tr>
<tr>
<td>1</td>
<td>0.600000</td>
</tr>
<tr>
<td>2</td>
<td>0.516800</td>
</tr>
<tr>
<td>3</td>
<td>0.455531</td>
</tr>
<tr>
<td>4</td>
<td>0.453551</td>
</tr>
<tr>
<td>5</td>
<td>0.453398</td>
</tr>
<tr>
<td>6</td>
<td>0.453398</td>
</tr>
</tbody>
</table>

In fact, \( x_6 \) is exact to 8 decimals places.

Of course, any number of steps between updates can be used. Naturally, the more often the update the faster the convergence but the more costly in terms of function evaluations.
4. Exercises

In problems 1–8 using the given function \( g(x) \) and starting value \( x_0 \), compute the iterates \( x_1, x_2, x_3, \) and \( x_4 \) using (4.2).

1. \( g(x) = \cos 2x, \quad x_0 = 0 \)
2. \( g(x) = \sqrt[3]{x}, \quad x_0 = 2 \)
3. \( g(x) = x - \ln x, \quad x_0 = 3 \)
4. \( g(x) = x^3 - 2x, \quad x_0 = 2 \)
5. \( g(x) = x^2 - 4, \quad x_0 = 2 \)
6. \( g(x) = x^2 - 3, \quad x_0 = 3 \)
7. \( g(x) = x^3 - x + 1, \quad x_0 = .5 \)
8. \( g(x) = e^x - 3, \quad x_0 = 1 \)

In problems 9–12 determine whether the iterates (4.2) will converge to the given root \( a \) if \( x_0 \) is close enough to \( a \).

9. \( g(x) = \frac{1}{2} \sin \pi x, \quad a = 0 \)
10. \( g(x) = 5 - \frac{4}{x}, \quad a = 4 \)
11. \( g(x) = x - 3 \log x, \quad a = 1 \)
12. \( g(x) = \sqrt{5x - 6}, \quad a = 3 \).

13. Let \( g(x) = x - f(x)/f'(x) \). (i) Show that the iterative method (4.2) for this function is in fact Newton’s Method. (ii) Suppose \( a \) is a simple root of \( f(x) = 0 \). Show that \( g'(a) = 0 \), thus giving an alternate proof of the quadratic convergence of Newton’s Method.

14. Show that (4.8) gives the \( x \)-value of the intersection of the lines \( y = x \) and the line

\[
y - \hat{x}_1 = \frac{g(\hat{x}_1) - \hat{x}_1}{\hat{x}_1 - x_0}(x - x_0).
\]

In problems 15–22 use the Aitken’s acceleration to compute \( x_1, x_2 \) and \( x_3 \) for the given functions \( g(x) \) and starting value \( x_0 \).

15. \( g(x) = e^{-x}, \quad x_0 = .5 \)
16. \( g(x) = 1/2x, \quad x_0 = 1 \)
17. \( g(x) = 1/2x + 1, \quad x_0 = -1 \)
18. \( g(x) = -1.2x + 3, \quad x_0 = 6 \)
19. \( g(x) = x^3 - x + 1, \quad x_0 = 2 \)
20. \( g(x) = x^3 - \frac{1}{2}x - 2x + 1, \quad x_0 = 2 \)
21. \( g(x) = \sqrt{x}, \quad x_0 = 2 \)
22. \( g(x) = \ln x + 2, \quad x_0 = 1 \).

23. The cubic \( p(x) = x^3 - 2x + 1 \) has a zero near \( a = .61803399 \). Show that if \( m = -1 \) the quasi-Newton method (4.8) will converge to \( a \) for any starting value \( x_0 \) in \([1/4, 3/4]\).

24. The polynomial \( p(x) = x^4 - 1 \) has a zero \( a = 1.0 \). Suppose \( m = 5 \). Determine how close \( x_0 \) must be to 1 in order that the quasi-Newton method (4.8) converges.
In problems 25–28 determine $x_1$, $x_2$ and $x_3$ using the quasi-Newton method (4.8) for the given functions and values $m$ and $x_0$.

25. $f(x) = x^2 - 3x + 2$, $x_0 = 3$, $m = 4$

26. $f(x) = x - \tan x$, $x_0 = \pi/4$, $m = -1$

27. $f(x) = x \ln x$, $x_0 = 2$, $m = -1$

28. $f(x) = x^3 - x + 1$, $x_0 = -1$, $m = 10$

29. Write the general formulas for the three step quasi-Newton method, analogous to (4.9).

30. Apply the quasi-Newton two step formula (4.9) to determine $x_1$, $x_2$ and $x_3$ to problems 25 and 27.

31. Show that (4.8) gives the $x$-intercept of the straight line passing through $(x_n, f(x_n))$ with slope $m$. 

4 Computer Exercises

I. Write a program that computes iterates (4.2) for a given function and stops when $|x_{n+1} - x_n| < \varepsilon$ or if the iteration count limit is exceeded.

**Applications.** Use the program to find a root of (4.1) for the given function $g(x)$ starting point $x_0$, tolerance $\varepsilon$, and iteration count limit $N$.

(a) $g(x) = -1 + \frac{6}{x}$, \hspace{1cm} $x_0 = 4$ \hspace{1cm} $\varepsilon = 10^{-4}$ \hspace{1cm} $N = 20$
(b) $g(x) = \frac{1}{10}(x^3 - 2x + 1)$, \hspace{1cm} $x_0 = 1$ \hspace{1cm} $\varepsilon = 10^{-5}$ \hspace{1cm} $N = 20$
(c) $g(x) = e^{-x}$, \hspace{1cm} $x_0 = 2$ \hspace{1cm} $\varepsilon = 10^{-4}$ \hspace{1cm} $N = 25$
(d) $g(x) = \arctan x$, \hspace{1cm} $x_0 = \pi/4$ \hspace{1cm} $\varepsilon = 10^{-5}$ \hspace{1cm} $N = 25$.

II. Write a program that determines an approximate root of (4.1) using Aitken’s Acceleration (4.9) for $g(x) = x^3 - 3x^2 + 3.1x$ using $\varepsilon = 10^{-5}$, $x_0 = 1.5$ and $N = 20$. Try also $x_0 = 2.5$. If $x_0 = 4$ how many iterations are required for convergence?

III. Write a program that determines an approximate root of (4.1) using Aitken’s acceleration (4.9) for a given function $g(x)$ with given values $x_0$, $\varepsilon$ and $N$.

**Applications.** Use program III to determine an approximate root to (4.1) using the given functions $g(x)$ and values $x_0$, $\varepsilon$, and $N$,

(a) $g(x) = e^{-x} + 2$, \hspace{1cm} $x_0 = 1.0$ \hspace{1cm} $\varepsilon = 10^{-4}$ \hspace{1cm} $N = 10$
(b) $g(x) = x^4 - 4x^3 + 5.9x^3 - 4.1x + 2$ \hspace{1cm} $x_0 = 1.0$ \hspace{1cm} $\varepsilon = 10^{-5}$ \hspace{1cm} $N = 10$
(c) $g(x) = -0.42 \cos x + 0.91 \sin x + 1.0019$, \hspace{1cm} $x_0 = 1.0$ \hspace{1cm} $\varepsilon = 10^{-6}$ \hspace{1cm} $N = 15$
(d) $g(x) = e^{(x - 3)^2} + 1$ \hspace{1cm} $x_0 = 1$ \hspace{1cm} $\varepsilon = 10^{-5}$ \hspace{1cm} $N = 10$
(e) $g(x) = x^3 - 2.9x^2 + 3x$ \hspace{1cm} $x_0 = 0.5$ \hspace{1cm} $\varepsilon = 10^{-6}$ \hspace{1cm} $N = 10$
(f) $g(x) = \sqrt{6x + x^2}$ \hspace{1cm} $x_0 = 5$ \hspace{1cm} $\varepsilon = 10^{-6}$ \hspace{1cm} $N = 10$.

IV. Write a program which determines a zero of the function $f(x) = x^4 - 4x^3 - 7x^2 + 22x + 24$ using the quasi-Newton two step method (4.9). Use $\varepsilon = \varepsilon_1 = 10^{-6}$, $x_0 = -3, -1.75, 2,$ and $6$. Take $N = 20$.

V. Combine the quasi-Newton two step method with deflation discussed in §3 to determine all the real roots of a given function. Use given tolerances $\varepsilon$ and $\varepsilon_1$ and given $x_0$ and $N$.

**Applications.** Apply program V to determine all the real roots of the given functions using the given values $\varepsilon, \varepsilon_1, x_0$ and $N$.

(a) $f(x) = e^x - x^4$ \hspace{1cm} $x_0 = 0$ \hspace{1cm} $\varepsilon = \varepsilon_1 = 10^{-6}$ \hspace{1cm} $N = 20$
(b) $f(x) = x^3 - \sin x$ \hspace{1cm} $x_0 = 3$ \hspace{1cm} $\varepsilon = \varepsilon_1 = 10^{-6}$ \hspace{1cm} $N = 10$
(c) $f(x) = x^4 + 4x^3 - 7x^2 - 22x + 24$ \hspace{1cm} $x_0 = 0$ \hspace{1cm} $\varepsilon = \varepsilon_1 = 10^{-5}$ \hspace{1cm} $N = 15$
(d) $f(x) = x^5 - 22.1x^4 + 169.7x^3 - 554.7x^2 + 784.8x - 378$, \hspace{1cm} $x_0 = 3$ \hspace{1cm} $\varepsilon = \varepsilon_1 = 10^{-4}$ \hspace{1cm} $N = 15$.