An Improved Newton’s Method
by John H. Mathews
California State University, Fullerton, CA 92634

John earned his Ph.D. in mathematics from Michigan State University under the direction of Peter Lappan. He has been a member of the mathematics faculty at California State University Fullerton for nineteen years, and keeps active in the areas of complex analysis and numerical analysis.

Introduction

Newton’s method is used to locate roots of the equation \( f(x) = 0 \). The Newton-Raphson iteration formula is:

\[
g(x) = x - \frac{f(x)}{f'(x)}. \tag{1}
\]

Given a starting value \( p_0 \), the sequence \( \{p_k\} \) is computed using:

\[
p_{k+1} = g(p_k) \quad \text{for } k = 0, 1, \ldots \text{ (provided that } f'(p_k) \neq 0). \tag{2}
\]

If the value \( p_0 \) is chosen close enough to the root \( p \), then the sequence generated in (2) will converge to the root \( p \). Sometimes the speed at which \( \{p_k\} \) converges is fast (quadratic) and at other times it is slow (linear). To distinguish these two cases we make the following definitions.

DEFINITION 1. Let \( \{p_k\} \) be a sequence that converges to \( p \), and set \( e_k = p - p_k \) for \( k = 0, 1, \ldots \). If there exists a constant \( A \neq 0 \) such that

\[
\lim_{n \to \infty} \frac{|e_{k+1}|}{|e_k|^2} = A, \tag{3}
\]

then \( \{p_k\} \) is said to converge quadratically to \( p \). If there exists a constant \( A \neq 0 \) such that

\[
\lim_{n \to \infty} \frac{|e_{k+1}|}{|e_k|} = A, \tag{4}
\]

then \( \{p_k\} \) is said to converge linearly to \( p \).

The mathematical characteristic for determining which case occurs is the “multiplicity” of the root \( p \).

DEFINITION 2. If \( f(x) \) can be factored as

\[
f(x) = (x-p)^M h(x), \tag{5}
\]

and \( h(x) \) is continuous at \( x = p \) and \( h(p) \neq 0 \), then we say that \( f(x) \) has a root of order \( M \) at \( x = p \).

A root of order \( M = 1 \) is called a simple root, and if \( M > 1 \) it is called a multiple root. The next result is well known and can be found in the references.
THEOREM 1. [Order of convergence for Newton-Raphson iteration] Let the sequence \{p_k\} generated by (2) converge to the root p. If p is a simple root, the convergence is quadratic and

\[ |e_{k+1}| \approx \frac{|f'(p)|}{2|f(p)|} |e_k|^2 \text{ for } k \text{ sufficiently large.} \tag{6} \]

If p is a root of order M, convergence is linear and

\[ |e_{k+1}| \approx \frac{M-1}{M} |p_k| \text{ for } k \text{ sufficiently large.} \tag{7} \]

There are two common ways to use Theorem 1 and gain quadratic convergence at multiple roots. We shall call these methods A and B (see Mathews, 1987, p. 72 and Ralston & Rabinowitz, 1978, pp. 353-356).

\textbf{Method A}

Suppose that p is a root of order \(M > 1\). Then the accelerated Newton-Raphson formula is:

\[ g(x) = x - \frac{M f(x)}{f'(x)}. \tag{8} \]

Let the starting value \(p_0\) be close to p, and compute the sequence \{p_k\} iteratively;

\[ p_{k+1} = p_k - \frac{M f(p_k)}{f'(p_k)} \text{ for } k = 0, 1, \ldots. \tag{9} \]

Then the sequence generated by (9) will converge quadratically to p.

On the other hand, if \(f(x) = (x-p)^M h(x)\) then one can show that the function \(u(x) = f(x)/f'(x)\) has a simple root at \(x = p\). Using \(u(x)\) in place of \(f(x)\) in formula (1) yields Method B.

\textbf{Method B}

Suppose that p is a root of order \(M > 1\). Then the modified Newton-Raphson formula is:

\[ g(x) = x - \frac{f(x)f'(x)}{[f'(x)]^2 - f(x)f''(x)}. \tag{10} \]

Let the starting value \(p_0\) be close to p, and compute the sequence \{p_k\} iteratively;

\[ p_{k+1} = p_k - \frac{f(p_k)f'(p_k)}{[f'(p_k)]^2 - f(p_k)f''(p_k)}. \tag{11} \]

Then the sequence generated by (11) converges quadratically to p.

\textbf{Limitations of Methods A and B}

Method A has the disadvantage that the order \(M\) of the root must be known a priori. Determining \(M\) is often laborious because some type of mathematical analysis must be used. It is usually found by looking at the values of the higher derivatives of \(f(x)\). That is, \(f(x) = 0\) has a root of order \(M\) at \(x = p\) if and only if

\[ f(p) = 0, f'(p) = 0, \ldots, f^{(M-1)}(p) = 0 \text{ and } f^{(M)}(p) \neq 0. \tag{12} \]
Dodes (1978, pp. 81-82) has observed that in practical problems it is unlikely that we will know the multiplicity. However, a constant $M$ should be used in (8) to speed up convergence, and it should be chosen small enough so that $p_{k+1}$ does not shoot to the wrong side of $p$. Rice (1983, pp. 232-233) has suggested a way to empirically find $M$. If $p^*$ is a good approximation to $p$ and $p_1$ and $p_2$ somewhat distant from $p^*$ then $M$ can be determined by the calculation:

$$M \approx \frac{\ln(f(p_1)/f(p_2))}{\ln((p_1-p^*)/(p_2-p^*))}.$$  

Method B has a disadvantage, it involves three functions $f(x)$, $f'(x)$ and $f''(x)$. Again, the laborious task of finding the formula for $f''(x)$ could detract from using Method B. Furthermore, Ralston and Rabinowitz (1978, pp. 353-356) have observed that $u(x)$ will have poles at points where the zeros of $f'(x)$ are not roots of $f(x)$. Hence, $u(x)$ may not be a continuous function.

**The New Method C**

The adaptive Newton-Raphson method incorporates a linear search method with formula (8). The following values are computed:

$$p_j = p_0 - jf(p_0)/f(p_0) \text{ for } j = 1, 2, ..., M.$$  

Our task is to determine the value $M$ to use in formula (13), because it is not known a priori. First, we take the derivative of $f(x)$ in formula (5), and obtain:

$$f'(x) = (x-p)^{M}h'(x) + M(x-p)^{M-1}h(x).$$  

When (5) and (14) are substituted into formula (1) we get:

$$g(x) = x - \frac{1}{M}(x-p) - \frac{1}{1 + (x-p)h'(x)/[Mh(x)]}.$$  

This enables us to rewrite (13) as:

$$p_j = p_0 - \frac{j}{M}(p_0 - p) - \frac{1}{1 + (p_0-p)h'(p_0)/[Mh(p_0)]}.$$  

We shall assume that the starting value $p_0$ is close enough to $p$ so that

$$\frac{1}{1 + (p_0-p)h'(p_0)/[Mh(p_0)]} = 1 + \epsilon,$$

where $\epsilon \approx 0$. The iterates $p_j$ in (15) satisfy the following:

$$p_j = p_0 - \frac{j}{M}(p_0 - p)(1 + \epsilon) \text{ for } j = 1, 2, ... .$$  

If we subtract $p$ from both sides of (17) then the result after simplification is:

$$p_j - p = \left[ (1 - j/M) - je/M \right] (p_0 - p).$$  

Since $je/M \approx 0$, the iterates $p_j$ get closer to $p$ as $j$ goes from 1 to $M$, which is manifest by the inequalities:

$$|p_0 - p| \geq |p_1 - p| \geq \cdots \geq |p_{M-1} - p| \geq |p_M - p|.$$  

The values $p_j$ are shown in Figure 1. Notice that if the iteration (15) was continued for $M+1$ and $M+2$ then $|f(p_{M+1})|$ and $|f(p_{M+2})|$ should be larger than $|f(p_M)|$. This is proven by
using the derivatives in (12) and the Taylor polynomial approximation of degree \( M \) for \( f(x) \) expanded about \( x = p \):

\[
f(x) \approx \frac{f^{(M)}(p)}{M!} (x - p)^M.
\]

The Adaptive Newton-Raphson Algorithm

Start with \( p_0 \), then we determine the next approximation \( p_i \) as follows:

1. \( D_x = \frac{f(p_0)}{f'(p_0)} \).
2. \( p_1 = p_0 - D_x \) and \( y_1 = f(p_1) \).
3. \( p_2 = p_0 - 2D_x \) and \( y_2 = f(p_2) \).
4. \( j = 3 \)
5. **WHILE** \( |y_2| \leq |y_1| \) **DO**
   1. \( p_1 := p_2 \)
   2. \( y_1 := y_2 \)
   3. \( p_2 := p_0 - j* D_x \)
   4. \( y_2 := f(p_2) \)
   5. \( j := j + 1 \)
6. **ENDWHILE**

If \( p_j \) is closer to \( p \) than \( p_i \), then (19) and (20) imply that \( |f(p_j)| < |f(p_i)| \), hence we have:

\[
|f(p_0)| > |f(p_1)| > \ldots > |f(p_j)| > \ldots > |f(p_M)|.
\]  

Therefore, the way to computationally determine \( M \) is to successfully compute the values \( p_j \) using formula (13) for \( j = 1,2,\ldots,M+1 \) until we arrive at \( |f(p_M)| < |f(p_{M+1})| \).

Figure 1. The values \( p_1, p_2, p_3 \) and \( p_4 \) obtained by using formula (15) near a root \( p \) of order \( M = 2 \). Notice that \( |f(p_3)| > |f(p_2)| \).
Observe that (22) involves a linear search in either the interval \((-\infty, p_0)\) when \(p_1 < p_0\) or in the interval \((p_0, \infty)\) when \(p_0 < p_1\). In the algorithm, the value \(Dx = f(p_0)/f'(p_0)\) is stored so that unnecessary computations are avoided. After the point \((p_1, f(p_1))\) has been found, it should replace \((p_0, f(p_0))\) and the process is repeated.

**EXAMPLE.** Use the value \(p_0 = 1.3\) and compare Methods A, B and C for finding the double root \(p = 1\) of the equation \(x^3 - 3x + 2 = 0\). Solution: The function is \(f(x) = x^3 - 3x + 2\) and \(f'(x) = 3x^2 - 3\) and \(f''(x) = 6x\). The table gives the iterates. For method C, all the values in the linear search are included.

<table>
<thead>
<tr>
<th>(k)</th>
<th>(P_k)</th>
<th>(P_k)</th>
<th>(P_k)</th>
<th>(P_k)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1.3000000000</td>
<td>1.3000000000</td>
<td>1.3000000000</td>
<td>1.3000000000</td>
</tr>
<tr>
<td>1</td>
<td>1.1565217391</td>
<td>1.0130434783</td>
<td>0.99997423872</td>
<td>1.0130434783</td>
</tr>
<tr>
<td>2</td>
<td>1.0801542777</td>
<td>1.0000258171</td>
<td>1.000000031</td>
<td>1.0000281717</td>
</tr>
<tr>
<td>3</td>
<td>1.0405919010</td>
<td>1.0000000205</td>
<td>1.0000000205</td>
<td>1.0000000205</td>
</tr>
<tr>
<td>4</td>
<td>1.0204305277</td>
<td>1.0000000205</td>
<td>1.0000000205</td>
<td>1.0000000205</td>
</tr>
<tr>
<td>5</td>
<td>1.0102496960</td>
<td>1.0000000205</td>
<td>1.0000000205</td>
<td>1.0000000205</td>
</tr>
<tr>
<td>6</td>
<td>1.0051335581</td>
<td>1.0000000205</td>
<td>1.0000000205</td>
<td>1.0000000205</td>
</tr>
<tr>
<td>7</td>
<td>1.0025689696</td>
<td>1.0000000205</td>
<td>1.0000000205</td>
<td>1.0000000205</td>
</tr>
<tr>
<td>8</td>
<td>1.0012850341</td>
<td>1.0000000205</td>
<td>1.0000000205</td>
<td>1.0000000205</td>
</tr>
<tr>
<td>9</td>
<td>1.0006426547</td>
<td>1.0000000205</td>
<td>1.0000000205</td>
<td>1.0000000205</td>
</tr>
<tr>
<td>10</td>
<td>1.0003213626</td>
<td>1.0000000205</td>
<td>1.0000000205</td>
<td>1.0000000205</td>
</tr>
<tr>
<td>11</td>
<td>1.0001606905</td>
<td>1.0000000205</td>
<td>1.0000000205</td>
<td>1.0000000205</td>
</tr>
<tr>
<td>12</td>
<td>1.0000803488</td>
<td>1.0000000205</td>
<td>1.0000000205</td>
<td>1.0000000205</td>
</tr>
<tr>
<td>13</td>
<td>1.0000401743</td>
<td>1.0000000205</td>
<td>1.0000000205</td>
<td>1.0000000205</td>
</tr>
<tr>
<td>14</td>
<td>1.0000200866</td>
<td>1.0000000205</td>
<td>1.0000000205</td>
<td>1.0000000205</td>
</tr>
<tr>
<td>15</td>
<td>1.0000100650</td>
<td>1.0000000205</td>
<td>1.0000000205</td>
<td>1.0000000205</td>
</tr>
</tbody>
</table>

Observe in the example that the standard Newton-Raphson method converges linearly and that methods A and B converge quadratically. The reader can use formulas (12) to verify that \(M=2\) is the order of the root \(p=1\). The formula for \(g(x)\) in Method B is:

\[
g(x) = x - \frac{x^3 - 3x + 2}{3x^2 - 3} - \frac{x^3 - 3x + 2}{6x}
\]

Notice that all the iterates in Method A occur in Method C, hence the subsequence \([p_0, p_2]\) in Method C converges quadratically. Only a few extra function evaluations were required by Method C while searching for the best value \(M\). However, this time is well spent when Method C is compared to Method A.

**A Computer Program for Method C**

PROGRAM Adaptive.Newton.Raphson;

CONST Delta=0.000000001; Epsilon=0.000000001; Max=200;
VAR Cond,J,K: INTEGER; Df,Dp,Y0,Error,P0,P1,P2,Y1,Y2: REAL;

FUNCTION F( X:REAL):REAL;
BEGIN F:=X*X*X - 3*X + 2; END;

FUNCTION F1(X :REAL):REAL;
BEGIN F1:=3*X*X - 3; END;
BEGIN
WRITE ('ENTER the initial approximation P0 = '); READLN(P0);
K:=0;
Cond:=0;
Y0:=F(P0);
P1:=P0+1;
WRITELN('P(',K:2,') = ',P0,' F(P(',K:2,')) = ',Y0);
WHILE (K < Max) AND (Cond =0) DO
BEGIN
Df:=F1(P0);
IF Df = 0 THEN
BEGIN
Dp:=P1-P0; P1:=P0; Cond:=1;
END
ELSE
BEGIN
Dp:=Y0/Df; P1:=P0-Dp;
END;
Y1:=F(P1);
WRITELN('P(',K+1:2,') = ',P1,' F(P(',K+1:2,')) = ',Y1);
P2:=P0-2*Dp;
Y2:=F(P2);
J:=3;
WHILE (ABS(Y2) < ABS(Y1)) AND (J < 11) DO
BEGIN
P1:=P2; Y1:=Y2;
P2:=P0-J*Dp; Y2:=F(P2);
J:=J+1; K:=K+1;
WRITELN('P(',K+1:2,') = ',P1,' F(P(',K+1:2,')) = ',Y1);
END
Error:=ABS(Dp);
IF (Error<Delta) OR (ABS(Y1)<Epsilon) THEN Cond:=1;
P0:=P1;
Y0:=Y1; K:=K+1;
END;
END.

References