Muller’s Method

Muller’s method is a generalization of the secant method, in the sense that it does not require the derivative of the function. It is an iterative method that requires three starting points \((p_0, f(p_0)), (p_1, f(p_1)),\) and \((p_2, f(p_2))\). A parabola is constructed that passes through the three points; then the quadratic formula is used to find a root of the quadratic for the next approximation. It has been proved that near a simple root Muller’s method converges faster than the secant method and almost as fast as Newton’s method. The method can be used to find real or complex zeros of a function and can be programmed to use complex arithmetic.

Without loss of generality, we assume that \(p_2\) is the best approximation to the root and consider the parabola through the three starting values, shown in Figure 2.17. Make the change of variable

\[
(9) \quad t = x - p_2,
\]

and use the differences

\[
(10) \quad h_0 = p_0 - p_2 \quad \text{and} \quad h_1 = p_1 - p_2.
\]

Consider the quadratic polynomial involving the variable \(t\):

\[
(11) \quad y = at^2 + bt + c.
\]
Each point is used to obtain an equation involving \(a\), \(b\), and \(c\):

\[
\begin{align*}
\text{At } t = h_0: & \quad a h_0^2 + b h_0 + c = f_0, \\
\text{At } t = h_1: & \quad a h_1^2 + b h_1 + c = f_1, \\
\text{At } t = 0: & \quad a 0^2 + b 0 + c = f_2.
\end{align*}
\]

(12)

From the third equation in (12), we see that

\[
c = f_2.
\]

(13)

Substituting (13) into the first two equations in (12) and using the definition \(e_0 = f_0 - c\) and \(e_1 = f_1 - c\) results in the linear system

\[
\begin{align*}
ah_0^2 + b h_0 &= f_0 - c = e_0, \\
ah_1^2 + b h_1 &= f_1 - c = e_1.
\end{align*}
\]

(14)

Solving the linear system for \(a\) and \(b\) results in

\[
\begin{align*}
a &= \frac{e_0 h_1 - e_1 h_0}{h_1 h_0^2 - h_0 h_1^2}, \\
b &= \frac{e_1 h_0^2 - e_0 h_1^2}{h_1 h_0^2 - h_0 h_1^2}.
\end{align*}
\]

(15)

The quadratic formula is used to find the roots \(t = z_1, z_2\) of (11):

\[
z = \frac{-2c}{b \pm \sqrt{b^2 - 4ac}}.
\]

Formula (16) is equivalent to the standard formula for the roots of a quadratic and is better in this case because we know that \(c = f_2\).

To ensure the stability of the method, we choose the root in (16) that has the smallest absolute value. If \(b > 0\), use the positive sign with the square root, and if \(b < 0\), use the negative sign. Then \(p_3\) is shown in Figure 2.17 and is given by

\[
p_3 = p_2 + z.
\]

(17)

To update the iterates, choose the new \(p_0\) and the new \(p_1\) to be the two values selected from among the old \(\{p_0, p_1, p_3\}\) that lie closest to \(p_3\) (i.e., throw out the one that is farthest away). Then take new \(p_2\) to be old \(p_3\). Although a lot of auxiliary calculations are done in Muller’s method, it only requires one function evaluation per iteration.

If Muller’s method is used to find the real roots of \(f(x) = 0\), it is possible that one may encounter complex approximations, because the roots of the quadratic in (16) might be complex (nonzero imaginary components). In these cases the imaginary components will have a small magnitude and can be set equal to zero so that the calculations proceed with real numbers.
Table 2.12  Comparison of Convergences near a Simple Root

\[
\begin{array}{|c|c|c|c|c|}
\hline
k & \text{Secant method} & \text{Muller’s method} & \text{Newton’s method} & \text{Steffensen with Newton} \\
\hline
0 & -2.600000000 & -2.600000000 & -2.400000000 & -2.400000000 \\
1 & -2.400000000 & -2.500000000 & -2.076190476 & -2.076190476 \\
2 & -2.106598985 & -2.400000000 & -2.003596011 & -2.003596011 \\
3 & -2.022641412 & -1.985275287 & -2.000008589 & -1.982618143 \\
5 & -2.000022537 & -2.00000218 & -2.000000000 & -2.000000000 \\
6 & -2.000000022 & -2.000000000 & -2.000000000 & -2.000000000 \\
\hline
\end{array}
\]

Comparison of Methods

Steffensen’s method can be used together with the Newton-Raphson fixed-point function \(g(x) = x - f(x)/f'(x)\). In the next two examples we look at the roots of the polynomial \(f(x) = x^3 - 3x + 2\). The Newton-Raphson function is \(g(x) = (2x^3 - 2)/(3x^2 - 3)\). When this function is used in Program 2.7, we get the calculations under the heading Steffensen with Newton in Tables 2.12 and 2.13. For example, starting with \(p_0 = -2.4\), we would compute

\[
(18) \quad p_1 = g(p_0) = -2.076190476,
\]

and

\[
(19) \quad p_2 = g(p_1) = -2.003596011.
\]

Then Aitken’s improvement will give \(p_3 = -1.982618143\).

Example 2.19 (Convergence near a Simple Root).  This is a comparison of methods for the function \(f(x) = x^3 - 3x + 2\) near the simple root \(p = -2\).

Newton’s method and the secant method for this function were given in Examples 2.14 and 2.16, respectively. Table 2.12 provides a summary of calculations for the methods.

Example 2.20 (Convergence near a Double Root).  This is a comparison of the methods for the function \(f(x) = x^3 - 3x + 2\) near the double root \(p = 1\). Table 2.13 provides a summary of calculations.

Newton’s method is the best choice for finding a simple root (see Table 2.12). At a double root, either Muller’s method or Steffensen’s method with the Newton-Raphson formula is a good choice (see Table 2.13). Note in the Aitken’s acceleration formula (4) that division by zero can occur as the sequence \(\{p_k\}\) converges. In this case, the last calculated approximation to zero should be used as the approximation to the zero of \(f\).
### Table 2.13 Comparison of Convergence Near a Double Root

<table>
<thead>
<tr>
<th>$k$</th>
<th>Secant method</th>
<th>Muller’s method</th>
<th>Newton’s method</th>
<th>Steffensen with Newton</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1.40000000000</td>
<td>1.40000000000</td>
<td>1.20000000000</td>
<td>1.20000000000</td>
</tr>
<tr>
<td>1</td>
<td>1.20000000000</td>
<td>1.30000000000</td>
<td>1.10303030303</td>
<td>1.10303030303</td>
</tr>
<tr>
<td>2</td>
<td>1.138461538</td>
<td>1.20000000000</td>
<td>1.052356417</td>
<td>1.052356417</td>
</tr>
<tr>
<td>3</td>
<td>1.083873738</td>
<td>1.003076923</td>
<td>1.02640814</td>
<td>0.996890433</td>
</tr>
<tr>
<td>4</td>
<td>1.053093854</td>
<td>1.003838922</td>
<td>1.013257734</td>
<td>0.998446023</td>
</tr>
<tr>
<td>5</td>
<td>1.032853156</td>
<td>1.000027140</td>
<td>1.006643418</td>
<td>0.99923213</td>
</tr>
<tr>
<td>6</td>
<td>1.020429426</td>
<td>0.999997914</td>
<td>1.003325375</td>
<td>0.999999193</td>
</tr>
<tr>
<td>7</td>
<td>1.012648627</td>
<td>0.999999747</td>
<td>1.001663607</td>
<td>0.999999597</td>
</tr>
<tr>
<td>8</td>
<td>1.007382124</td>
<td>1.000000000</td>
<td>1.000832034</td>
<td>0.999999798</td>
</tr>
<tr>
<td>9</td>
<td>1.004844757</td>
<td>1.000416075</td>
<td>1.000416075</td>
<td>0.999999999</td>
</tr>
</tbody>
</table>

In the following program the sequence \( \{p_k\} \), generated by Steffensen’s method with the Newton-Raphson formula, is stored in a matrix \( Q \) that has max 1 rows and three columns. The first column of \( Q \) contains the initial approximation to the root, \( p_0 \), and the terms \( p_3, p_6, \ldots, p_{3k}, \ldots \) generated by Aitken’s acceleration method (4). The second and third columns of \( Q \) contain the terms generated by Newton’s method. The stopping criteria in the program are based on the difference between consecutive terms from the first column of \( Q \).