INT. J. MATH. EDUC. SCI. TECHNOL., 2002
VOL. 33, NO. 4, 521–527

Taylor & Francis
Taylor & Francis Group

Leap-frogging Newton's method

A. BATHI KASTURIARACHI
Department of Mathematics & Computer Science, Kent State University – Stark Campus, 6000 Frank Avenue, N.W. Canton, OH 44720, USA
email: bathi@stark.kent.edu

(*Received 30 May 2001*)

Using Newton's method as an intermediate step, we introduce an iterative method that approximates numerically the solution of $f(x) = 0$. The method is essentially a leap-frog Newton's method. The order of convergence of the proposed method at a simple root is cubic and the computational efficiency in general is less, but close to that of Newton's method. Like Newton's method, the new method requires only function and first derivative evaluations. The method can easily be implemented on computer algebra systems where high machine precision is available.

1. A higher order Newton-type method

Solutions to single non-linear equations that have no solutions in closed form are of much interest in physics. In these problems we seek methods that lead to approximate solutions. Newton's method is often the method of choice for approximating such solutions. Newton's method of iteration also appears in physics in the analysis of the chaotic behaviour of dynamical systems determined by rational maps [1]. The convergence of the iterates and the rate of convergence play an important role in the design of new iterative methods. We introduce a leap-frog Newton's method for solving the equation $f(x) = 0$, starting with a reasonable initial guess. The proposed method has third-order convergence at a simple root and the computational efficiency is comparable to that of Newton's method. It is well known that Newton's method (Newton–Raphson method) has second-order convergence at a simple root [2]. There are higher order methods that allow for faster convergence. For instance, Halley's method [3] has third-order convergence, but requires the second derivative. Similar to Newton's method, the proposed method will only require function and first derivative evaluations. In this respect, the method is easy to implement.

Suppose that the function $f(x)$ has a zero p in the interval $[a, b]$ and that $f \in C^2[a, b]$. Let x_0 be the initial guess. If the equation of the tangent line at $(x_0, f(x_0))$ passes through the intercept $(x_0, 0)$ we obtain the usual Newton's approximation,

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

International Journal of Mathematical Education in Science and Technology
ISSN 0020–739X print/ISSN 1464–5211 online © 2002 Taylor & Francis Ltd
http://www.tandf.co.uk/journals
DOI: 10.1080/00207390210131786

Here we have used $\overline{x_0}$, instead of x_1, since this is used only as an intermediate approximation. The equation of the secant line joining the points $(x_0, f(x_0))$ and $((\overline{x_0}, f(\overline{x_0}))$ is given by,

$$y - f(x_0) = \frac{[f(x_0) - f(\overline{x_0})]}{(x_0 - \overline{x_0})}(x - x_0)$$

Assume the secant line meets the x-axis at the point $(x_1, 0)$. This leads to the new estimate,

$$x_1 = x_0 - \frac{[f(x_0)]^2}{f'(x_0)[f(x_0) - f(\overline{x_0})]}$$

Repeating this process, we obtain a sequence of numbers $x_1, x_2, \ldots, x_n, \ldots$ that will approach the root p. In general, we have the iteration formula:

$$x_{n+1} = x_n - \frac{[f(x_n)]^2}{f'(x_n)[f(x_n) - f(\overline{x_n})]} \tag{1}$$

where

$$\overline{x_n} = x_n - \frac{f(x_n)}{f'(x_n)} \tag{2}$$

We will refer to equations (1) and (2) as the leap-frog Newton's method (figure 1). It should be noted that the denominator in equation (1) could become very small quickly, causing round-off problems. Using the machine precision n we can stop the iterations when $|f(x_n) - f(\overline{x_n})| < 10^{n-1}$. Such a manipulation can easily be performed on a computer algebra system such as Maple or Mathematica. The leap-frog Newton's method is really a combination of Newton's method followed by a pseudo-secant method. A method that uses a genuine secant method first, followed by Newton's method can be found in [4]. Such a method would lead to a bracketed interval that traps the root. However, a detailed analysis of convergence is not provided in [4].

The following two propositions will discuss the convergence of the iterates of the leap-frog Newton's method.

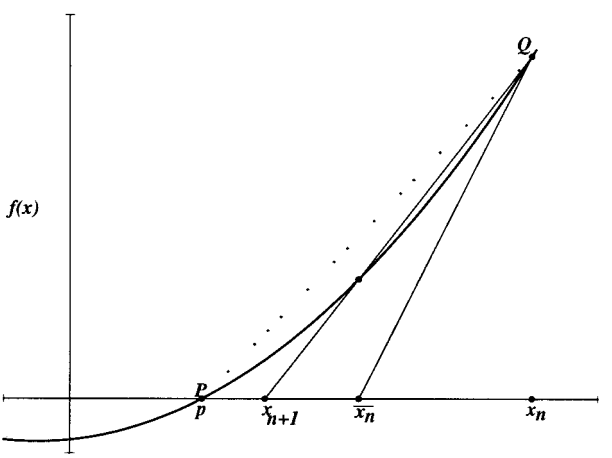

Figure 1. Illustration of the leap-frog Newton's method.

Proposition 1. Let $f(a)f(b) < 0$ and $f \in C^2[a,b]$ and $f'(x)$, $f''(x)$ are non zero and preserving signs on $[a,b]$. Choose an initial approximation $x_0 \in [a,b]$ such that $f(x_0)f''(x_0) > 0$. Then the leap-frog Newton's method given by (1) and (2) can be used to compute the root p of $f(x) = 0$ to any degree of accuracy.

Proof. We consider the case when $f(a) < 0, f(b) > 0$, and $f'(x) > 0, f''(x) > 0$ on $[a,b]$. The other cases:

- $f(a) > 0, f(b) < 0$, and $f'(x) < 0, f''(x) > 0$ on $[a,b]$,
- $f(a) < 0, f(b) > 0$, and $f'(x) > 0, f''(x) < 0$ on $[a,b]$,
- $f(a) > 0, f(b) < 0$, and $f'(x) < 0, f''(x) < 0$ on $[a,b]$,
- $f(a) > 0, f(b) > 0$, $f^{(n)}(p) = 0$ for some $n \geq 1$, and $f''(x) \geq 0$ on $[a,b]$ (multiple root),
- $f(a) < 0, f(b) < 0$, $f^{(n)}(p) = 0$ for some $n \geq 1$, and $f''(x) \leq 0$ on $[a,b]$ (multiple root),

can be handled similarly.

For each n, equation (1) can be rewritten as,

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \cdot \frac{f(x_n)}{\left[\dfrac{f(x_n) - f(\overline{x_n})}{x_n - \overline{x_n}}\right](x_n - \overline{x_n})} \tag{3}$$

Using equation (2) and the Mean Value Theorem we can reduce equation (3) to

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(c_n)} \tag{4}$$

for some c_n satisfying $\overline{x_n} < c_n < x_n$. According to our assumptions (see figure 1),

$$\textit{slope of } PQ = \frac{f(x_n)}{x_n - p} < f'(c_n) < f'(x_n)$$

Therefore we obtain the following inequalities:

$$\begin{aligned} p = x_n - \frac{f(x_n)}{f(x_n)/(x_n - p)} &< x_{n+1} = x_n - \frac{f(x_n)}{f'(c_n)} \\ &< x_n - \frac{f(x_n)}{f'(x_n)} = \overline{x_n} < c_n < x_n \end{aligned} \tag{5}$$

It follows that the approximations x_n form a bounded monotonic sequence, therefore its limit, $\lim_{n\to\infty} x_n = p_0$, exists. The string of inequalities (5) also show that the sequence c_n is bounded and monotonic, so $\lim_{n\to\infty} c_n = c_0$ exists. Passing to the limit in equation (4) we obtain

$$p_0 = p_0 - \frac{f(p_0)}{f'(c_0)}$$

It follows that $f(p_0) = f(p) = 0$, so that $p = p_0$, which completes the proof. □

The next proposition describes the rate of convergence of the proposed scheme at a simple root.

Proposition 2. Let p be a solution of the equation $f(x) = 0$. Suppose that $f(x)$, $f'(x)$, and $f''(x)$ are all continuous for all x in some neighbourhood of p. Assume $f(p) = 0, f'(p) \neq 0$. If x_0 is chosen sufficiently close to p, the convergence of the iterates x_n, $n \geq 0$, of the leap-frog Newton's method given by (1) and (2), to the root p, is cubic.

Proof. We begin with equation (1) and write

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

We will choose x_0 such that $|f(\overline{x_n})/f(x_n)| < 1$. Expanding as a series,

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \left[1 + \frac{f(\overline{x_n})}{f(x_n)} + \left(\frac{f(\overline{x_n})}{f(x_n)} \right)^2 + \cdots \right] \tag{6}$$

We will appeal to Taylor's theorem to replace the terms in the square brackets of equation (6):

$$f(\overline{x_n}) = f(x_n) + f'(x_n)(\overline{x_n} - x_n) + \frac{f''(x_n)}{2}(\overline{x_n} - x_n)^2 + \frac{f^{(3)}(x_n)}{6}(\overline{x_n} - x_n)^3 + \frac{f^{(4)}(c_n)}{24}(\overline{x_n} - x_n)^4$$

where $\overline{x_n} < c_n < x_n$. Substituting for $\overline{x_n} - x_n$ from equation (2) and simplifying we obtain

$$\frac{f(\overline{x_n})}{f(x_n)} = \frac{f''(x_n)}{2} \cdot \frac{f(x_n)}{f'(x_n)^2} - \frac{f^{(3)}(x_n)}{6} \cdot \frac{f(x_n)^2}{f'(x_n)^3} + \frac{f^{(4)}(c_n)}{24} \cdot \frac{f(x_n)^3}{f'(x_n)^4}$$

Substituting the above in equation (6) and collecting terms,

$$x_{n+1} - p = x_n - p - \frac{f(x_n)}{f'(x_n)} - \frac{f''(x_n)}{2} \cdot \frac{f(x_n)^2}{f'(x_n)^3} + \frac{f^{(3)}(x_n)}{6} \cdot \frac{f(x_n)^3}{f'(x_n)^4} + O\left(1/|f'(x_n)|^5\right) \tag{7}$$

Using Taylor's theorem again,

$$0 = f(p) = f(x_n) + f'(x_n)(p - x_n) + \frac{f''(x_n)}{2} \cdot (p - x_n)^2 + \frac{f^{(3)}(d_n)}{6} \cdot (p - x_n)^3 \tag{15}$$

where $p < d_n < x_n$. Reducing further,

$$\frac{-f(x_n)}{f'(x_n)} = p - x_n + \frac{f''(x_n)}{2} \frac{(p - x_n)^2}{f'(x_n)} + \frac{f^{(3)}(d_n)}{6} \frac{(p - x_n)^3}{f'(x_n)}$$

Finally, substituting the above in equation (7) and collecting terms we obtain

$$x_{n+1} - p = \left[\frac{f''(x_n)^2}{2f'(x_n)^2} - \frac{f^{(3)}(x_n)}{6f'(x_n)} - \frac{f^{(3)}(d_n)}{6f'(x_n)}\right](x_n - p)^3 + O\left(|p - x_n|^4\right) \tag{8}$$

In the limit $n \to \infty$, $x_n \to p$ and $d_n \to p$, so that equation (8) reduces to

$$\lim_{n\to\infty} \frac{|x_{n+1} - p|}{|x_n - p|^3} = \frac{|3(f''(p))^2 - 2f^{(3)}(p)f'(p)|}{6(f'(p))^2} \tag{9}$$

Equation (9) proves that the convergence of the sequence $\{x_n\}_{n=0}^{\infty}$ to p, is cubic. Notice the same equation can be used to prove the convergence of the leap-frog Newton's method given in Proposition 1 in the case of a simple root. □

The preceding proposition shows the functional iteration method we have introduced has a faster rate of convergence than Newton's method. However, it makes more sense to compare the *informational efficiency* and *computational efficiency* of the two schemes.

There are two types of indices that measure the *informational efficiency* [5]. They are,

$$EFF = \frac{\alpha}{d} \quad \text{and} \quad {}^*EFF = \alpha^{1/d}$$

where α is the order of the method and d is the informational usage, which is defined as the number of new pieces of information required per iteration. For Newton's method (with $d = 2$),

$$EFF = 1 \quad \text{and} \quad {}^*EFF = \sqrt{2} \approx 1.4142$$

For the leap-frog Newton's method (with $d = 3$),

$$EFF = 1 \quad \text{and} \quad {}^*EFF = \sqrt[3]{3} \approx 1.4422$$

This shows that the leap-frog Newton's method has slightly higher informational efficiency.

The *computational efficiency* measures the amount of computation needed to arrive at an approximation. The computational efficiency index [6], is defined by,

$$EI = \alpha^{1/\theta}$$

where α is the order of the method and θ is the cost per iteration. The computational efficiency index for Newton's method and leap-frog Newton's method are respectively given by,

$$EI = 2^{1/(1+\theta_1)}, \qquad EI = 3^{1/(2+\theta_1)}$$

where θ_1 is the cost of evaluating $f'(x)$. A straightforward calculation will show that if $\theta_1 > 0.7095$, the leap-frog Newton's method is more efficient. Since the cost of evaluating a derivative is often much less than that of evaluating a function, we can conclude that in general, Newton's method is slightly more computationally efficient.

2. Examples

The following examples will illustrate leap-frog Newton's method and compare it to Newton's method. The first example demonstrates cubic convergence.

Example 1. The function $f(x) = x^3 - 3x^2 - 5$ has a simple zero on the interval $[2, 5]$. Based on Proposition 1, we can make $x_0 = 5.0$ our initial guess. Since the root is solvable by radicals, we can obtain an approximation for the root to any desired accuracy. For instance, up to 64 decimals we may take the value of the root to be,

$$p = 3.425\,988\,757\,361\,622\,126\,076\,418\,043\,532\,96$$
$$0\,077\,351\,061\,466\,129\,852\,522\,913\,586\,974\,0$$

The exact value of the root is,

$$p = \frac{1}{2}\sqrt[3]{28 + 12\sqrt{5}} + \frac{2}{\sqrt[3]{28 + 12\sqrt{5}}} + 1$$

Table 1 illustrates cubic convergence of the leap-frog Newton's method by demonstrating the tripling of the accuracy with each step. In comparison, Newton's method doubles in accuracy with each step. Here we have recorded the error $|p - x_n|$ for each method. The table was generated by running both methods in Maple with number of digits set to 256 (`Digits := 256`).

Table 1 clearly demonstrates cubic convergence of the leap-frog Newton's method. It is not entirely fair to compare the values of leap-frog Newton's method to that of Newton's method. After all, a second Newton step can begin while the leap-frog method completes the calculation. One alternative is to compare a single leap-frog Newton's method to that of two Newton's methods. By doing so the disadvantage is shifted to the leap-frog Newton's method, which requires only one derivative calculation, while two Newton's method applications require two derivative calculations! Recall that the *informational usage*, d, for Newton's method and the leap-frog Newton's method are 2 and 3 respectively. Therefore, it is reasonable to compare the results of every *third* Newton's step to every *second* leap-frog Newton step. At this stage the number of pieces of information used is the same (six) for both methods. In Table 1, if we compare error values corresponding to $n = 3, 6, 9, \ldots$ of Newton's method to those of $n = 2, 4, 6, \ldots$ of the leap-frog Newton's method, we see that the proposed method clearly stands out.

The next example will illustrate the convergence of the leap-frog Newton's method when Newton's method fails.

n	Newton error	Leap-frog Newton error
1	5.740×10^{-1}	2.504×10^{-1}
2	1.156×10^{-1}	2.983×10^{-3}
3	6.134×10^{-3}	6.527×10^{-9}
4	1.860×10^{-5}	6.859×10^{-26}
5	1.718×10^{-10}	7.959×10^{-77}
6	1.467×10^{-20}	1.244×10^{-228}
7	1.069×10^{-40}	0
8	5.675×10^{-81}	0
9	1.599×10^{-161}	0

Table 1. Comparative rates of convergence: Newton's method versus leap-frog Newton's method.

n	Newton x_n	Leap-frog Newton x_n
1	−2.0	0.107 243 151 757 945 9
2	4.0	−0.035 119 987 560 042 18
3	−8.0	0.011 501 093 598 977 81
4	16.0	−0.003 766 378 155 638 774
5	−32.0	0.001 233 413 526 217 517
6	64.0	−0.000 403 918 264 122 981 3
7	−128.0	0.000 132 275 153 972 448 3
8	256.0	−0.000 043 317 467 697 146 94
9	−512.0	0.000 014 185 604 411 272 93
10	1024.0	−0.000 004 645 501 762 015 047

Table 2. A pathological example: Newton's method versus leap-frog Newton's method.

Example 2. The function $f(x) = x^{1/3}$ has a simple root at $p = 0$. This example is often used as a pathological example to illustrate a situation in which the Newton's method fails (table 2). However, the leap-frog Newton's method converges to the root slowly (≈ 0.5 order). Notice in this example the conditions of Proposition 1 are not met.

In conclusion, the leap-frog Newton's method introduced above has third-order convergence at a simple root with comparable computational efficiency. Generalization of the method, modifications at multiple roots, a higher order iteration method involving the curvature at a point, and other examples of pedagogical interest will be detailed in another article in preparation [7].

Acknowledgments

I am grateful to the Professional Activities Advisory Committee of Kent State University—Stark Campus for providing me with a load release to complete the writing of this paper.

References

[1] Barnsley, M., 2000, *Fractals Everywhere*, 2nd edn (Academic Press).
[2] Burden, R. L., and Faires, J. D., 1997, *Numerical Analysis*, 6th edn (Brooks/Cole).
[3] Ralston, A., 1965, *A First Course in Numerical Analysis* (McGraw-Hill).
[4] Demidovich, B. P., and Maron, I. A., 1981, *Computational Mathematics* (English translation) (Mir Publishers).
[5] Traub, J. F., 1982, *Iterative Methods for the Solution of Equations* (New York: Chelsea).
[6] Ralston, A., and Rabinowitz, P., 1978, *A First Course in Numerical Analysis* (McGraw-Hill).
[7] Kasturiarachi, A. B., 2002, *Higher Order Function Iteration Methods* (in preparation).