Numer. Math. (2013) 123:177–188
DOI 10.1007/s00211-012-0480-x

Numerische Mathematik

A modified Brent's method for finding zeros of functions

Gautam Wilkins · Ming Gu

Received: 30 June 2011 / Revised: 14 March 2012 / Published online: 26 June 2012

Abstract Brent's method, also known as zeroin, has been the most popular method for finding zeros of functions since it was developed in 1972. This method usually converges very quickly to a zero; for the occasional difficult functions encountered in practice, it typically takes $O(n)$ iterations to converge, where n is the number of steps required for the bisection method to find the zero to approximately the same accuracy. While it has long been known that in theory Brent's method could require as many as $O(n^2)$ iterations to find a zero, such behavior had never been observed in practice. In this paper, we first show that Brent's method can indeed take $O(n^2)$ iterations to converge, by explicitly constructing such worst case functions. In particular, for double precision accuracy, Brent's method takes 2,914 iterations to find the zero of our function, compared to the 77 iterations required by bisection. Secondly, we present a modification of Brent's method that places a stricter complexity bound of $O(n)$ on the search for a zero. In our extensive testing, this modification appears to behave very similarly to Brent's method for all the common functions, yet it remains at worst five times slower than the bisection method for all difficult functions, in sharp contrast to Brent's method.

Mathematics Subject Classification (2000) 65Y20 · 65Y04

G. Wilkins (✉) · M. Gu
Department of Mathematics, University of California, Berkeley, USA
e-mail: gwilkins@berkeley.edu; gwilkins@math.ucsd.edu

Present Address:
G. Wilkins
Department of Mathematics, University of California, San Diego, USA

1 Introduction

Finding the zeros of single-variable, real-valued functions is a very common and basic task in scientific computing. Given a function $f(x)$ that is continuous on the interval $[a, b]$ with $f(a)f(b) < 0$, the bisection method is guaranteed to find a zero in $[a, b]$. The reliability of the bisection method is offset by its disappointing linear convergence. It typically requires $\log_2 \frac{b-a}{\delta}$ iterations to achieve a given accuracy tolerance δ.

On the other hand, methods such as the secant method (see Eq. (1)) can converge much more quickly, but could diverge without reliable initial guesses [2]. Brent's method [1] is a quite successful attempt at combining the reliability of the bisection method with the faster convergence of the secant method and the inverse quadratic interpolation method.

Brent's method is an improvement of an earlier algorithm originally proposed by Dekker [1,3]. Assume that a given function $f(x)$ is continuous on an interval $[a, b]$, such that $f(a)f(b) \leq 0$. It is well-known that a zero of f is guaranteed to exist somewhere in $[a, b]$. The secant method produces a better approximate zero c as

$$c = b - \frac{b-a}{f(b) - f(a)} f(b). \tag{1}$$

The secant method is a superlinearly convergent method if the zero is simple and f is twice differentiable [1]. In order to safeguard against the secant method leading in a spurious direction, Dekker employs a bisection step anytime the new point computed by the secant method is not between $\frac{a+b}{2}$ and the previous computed point. The algorithm will terminate either when an exact zero is found, or when the size of the interval $[a, b]$ shrinks below some prescribed numerical tolerance, $\delta > 0$.

Dekker's method, however, does not place a reasonable bound on the complexity of the search for a zero. For certain functions it may perform a very large number secant iterations yet make virtually no progress in shrinking the interval around the zero. Brent's method aims to avoid such stagnant iterations. The details of Brent's method are discussed in Sect. 2.

Since its development in 1972, Brent's method has been the most popular method for finding zeros of functions. This method usually converges very quickly to a zero; for the occasional difficult functions encountered in practice, it typically takes $O(n)$ iterations to converge, where n is the number of steps required for the bisection method to find the zero to approximately the same accuracy. Brent shows that this method requires as many as $O(n^2)$ iterations in the worst case, although in practice such behavior had never been observed.

Brent's method is the basis of the *fzero* function in `Matlab`, a commercial software package by The MathWorks Inc., because of its ability to use "rapidly convergent methods when they are available" and "a slower, but sure, method when it is necessary" (see [5]).

The first contribution of this paper is to show that Brent's method can indeed take $O(n^2)$ iterations to converge. We do this by explicitly constructing such worst case functions. In particular, for double precision accuracy, Brent's method takes 2,914

iterations to find the zero of our function, whereas the bisection method takes only 77 iterations to achieve the same accuracy.

The second contribution is a simple modification of Brent's method, which we will call *modified zeroin*. We show that this modification requires at most $O(n)$ iterations to find a zero, where the constant hidden in the O notation does not exceed 5 in the worst case. In our extensive testing, this modification appears to behave very similarly to Brent's method for all the common functions, yet has the same order of convergence as the bisection method for all difficult functions, in sharp contrast to Brent's method.

Brent's method does not require any information about the derivatives of the function $f(x)$. This simplicity, plus the practical efficiency, has made Brent's method the method of choice for many practical zero finding computations. While derivative information can be used to develop more efficient zero finders, it will not be discussed in this paper as we are primarily interested in techniques that may be employed to safeguard superlinearly convergent methods so that they do not lead in spurious directions or take an unduly large number of iterations to converge to zeros of ill-behaved functions. The interested reader is referred to Kahan [4] for a broader discussion of zero finders, as well as recent papers [6–9] that present higher order zero finding methods. It is worth noting, however, that techniques discussed in this paper may be applied to any superlinearly convergent method to guarantee $O(n)$ convergence.

2 Brent's method

We begin this section by introducing the inverse quadratic interpolation method (IQI). Given three distinct points a, b, and c, with their corresponding function values $f(a)$, $f(b)$, and $f(c)$, the IQI method produces a new approximate zero as

$$d = \frac{f(b)f(c)}{(f(a)-f(b))(f(a)-f(c))}a + \frac{f(a)f(c)}{(f(b)-f(a))(f(b)-f(c))}b + \frac{f(a)f(b)}{(f(c)-f(a))(f(c)-f(b))}c, \tag{2}$$

if the right hand side of (2) is defined. The order of convergence of the IQI method is approximately 1.839, while the order of convergence of the secant method approximately 1.618.

Brent's method differs from Dekker's method in a number of ways. It uses inverse quadratic interpolation (see Eq. (2)) whenever possible, resulting in an increased speed of convergence. In addition, it changes the criteria for which a bisection step will be performed. Let b_j be the best approximation to the zero after the jth iteration of Brent's method. If interpolation was used to obtain b_j, then the following two inequalities must simultaneously be satisfied:

$$|b_{j+1} - b_j| < 0.5|b_{j-1} - b_{j-2}| \tag{3}$$

$$|b_{j+1} - b_j| > \delta \tag{4}$$

where δ is a numerical tolerance analogous to that in Dekker's method, and b_{j+1} is the new point computed by interpolation. If either of these inequalities is not satisfied, then b_{j+1} is discarded and a bisection step is performed instead. The first inequality thus places a bound on the distance between two successive points computed through interpolation that decreases by a factor of at least two, every two steps. Assuming that the first condition is never violated, then at the jth step the second condition will be violated after at most n additional steps, where:

$$\frac{|b_{j-1} - b_{j-2}|}{2^{n/2}} \in (\delta/2, \delta]$$
$$n = 2 \lg \left(\frac{|\mathrm{b_{j-1}} - \mathrm{b_{j-2}}|}{\delta} \right),$$

and we define $\lg(x) := \lceil \log_2(x) \rceil$. Thus, a bisection step will be performed at least every n steps following an interpolation step. If we assume that in the worst-case the interval is not shrunk at all by interpolation, and that bisection steps are performed as infrequently as possible, then the interval size decreases by a factor of two every n steps. Thus, given an initial interval $[a, b]$, Brent's method will terminate in no more than k steps, where:

$$\frac{|b - a|}{2^{k/n}} \in (\delta/2, \delta]$$
$$k = n \lg \left(\frac{|\mathrm{b} - \mathrm{a}|}{\delta} \right)$$
$$k = 2 \lg \left(\frac{|\mathrm{b} - \mathrm{a}|}{\delta} \right)^2.$$

So, Brent's method will terminate in $O(\log_2(\frac{|b-a|}{\delta})^2)$ time, where $O(\log_2(\frac{|b-a|}{\delta}))$ is the expected running time of the bisection method.

3 Convergence issues with Brent's method

While a function that actually results in worst-case convergence of Brent's method is not likely to be encountered in practice, there are practical examples where Brent's method performs poorly (i.e. much worse than bisection). Consider the function $f(x) = (x + 2)(x + 1)^2(x)(x - 1)^3(x - 2)$. If Brent's method is used, with an initial interval of $[-1.5, 1.75]$, then the initial interval contains zeros at $x = -1, 0, 1$. However, no sign reversal occurs for the zero at $x = -1$, since it is a root with even multiplicity. The stopping criteria for Brent's method are either that it find a value of x for which $f(x)$ is identically zero, or that it finds two values, a and b, such that $f(a)f(b) < 0$ and $|a - b| < \delta$, for some numerical tolerance, $\delta > 0$. Thus, Brent's method will be unable to straddle the zero at $x = -1$, since there is no sign reversal.

For the interval $[-1.5, 1.75]$, the inverse quadratic interpolation employed by Brent's method will approach $x = -1$, taking increasingly small steps. As there

is in fact a zero at that value of x, this is not surprising. It does, however, create a problem, as Brent's method will be unable to shrink the interval around $x = -1$. What happens instead, is that the method creeps toward $x = -1$ from the left with ever shrinking steps. Thus, until a bisection step is forced, the inverse quadratic interpolation will continue to approach a zero that Brent's method cannot find unless an iterate hits $x = -1$ exactly by an unlikely accident.

4 Proof of $\Omega(n^2)$ time

Brent placed an upper bound of $O(n^2)$ on the complexity of his method, although did not present any functions that exhibited this worst-case performance. In this section we show that the complexity of Brent's method is indeed $\Omega(n^2)$. We present a method of constructing such a function.

We begin by defining the x points, a set of $\Theta(n^2)$ distinct values, for an interval $[a, b]$. These points will be chosen, together with a properly chosen function $f(x)$ to be constructed later, to force Brent's method to iterate through all of them, thereby exhibiting its worst-case performance.

4.1 The worst-case iterations

Our goal is to make Brent's method perform the maximum number of interpolation steps in between every pair of bisection steps. Given an initial interval of $[a, b]$ and numerical tolerance, δ, we know from Sect. 2 that the maximum number of interpolation steps that may be performed before a bisection step is forced is: $k = 2\log_2 \frac{|b-a|}{\delta}$. For our purposes, however, we may only perform $k = \log_2 \frac{|b-a|}{\delta}$ steps before forcing a bisection. The work in Sect. 2 assumed that interpolation did not shrink the interval at all, since its goal was to place an upper bound on the number of iterations. In actuality, the last interpolation step before a bisection is performed (the kth step) shrinks the interval by δ, violating inequality (3). Since we want interpolation to be performed for each of the k steps, they must also satisfy inequality (4). If the distance between every two successive steps shrinks by a factor less than 2^{-1}, then it is sufficient for the distance between every successive step to shrink by a factor less that $2^{(-1/2)}$. Thus, if we chose some factor $p > \sqrt{2}$, the first interpolation step must reduce the interval by $(p^{k-1}\delta)$, the second interpolation step must reduce the interval by $(p^{k-2}\delta)$, and the ith interpolation step must reduce the interval by $(p^{k-i}\delta)$. So the worst-case set of x values will be $[b, b - p^{k-1}\delta, b - p^{k-1}\delta - p^{k-2}\delta, \ldots, b - \sum_{j=1}^{k} p^{k-j}\delta]$. After the kth interpolation step, the interval will be bisected, giving us a new interval, $[a, b']$, of roughly half the length of the original interval. Now it is clear why we may not perform

$$k = 2\log_2\left(\frac{|b-a|}{\delta}\right)$$

steps before forcing a bisection, since

$$\sqrt{2}^{2\log_2\left(\frac{|b-a|}{\delta}\right)}\delta = |b-a|,$$

meaning the size of the first step would be equal to the size of the initial interval. We may now apply this process to the interval $[a, b']$, and continue until we reach an interval whose size is less than δ, at which point Brent's method will end. There are $\Theta(n^2)$ such points, and an analysis similar to that of Sect. 2 leads us to expect approximately $n^2/2$ points.

4.2 The worst-case function

Now we explicitly construct a special function $f : \mathbb{R} \to \mathbb{R}$ such that when one applies Brent's method to $f(x)$, the resulting iterates are precisely the x points defined in Sect. 4.1.

We first consider the secant method (1). Given two points x_a, x_b and their corresponding function values $f_a = f(x_a)$, $f_b = f(x_b)$, the secant method (1) generates a new iterate

$$x_c = x_b - \frac{x_b - x_a}{f_b - f_a} f_b.$$

Solving this equation for f_b, we have

$$f_b = f_a \frac{x_b - x_c}{x_a - x_c}.$$

We define the function

$$s(x_a, x_c, x_b, f_a) = f_a \frac{x_b - x_c}{x_a - x_c}. \tag{5}$$

If the secant method is called with the points x_a, and x_b, for a function whose value at x_a is f_a, and whose value at x_b is $f_b = s(x_a, x_c, x_b, f_a)$, then the point returned by the secant method will be exactly x_c.

For the IQI method (2), assume that we are given three points x_a, x_b, and x_c, along with their function values $f_a = f(x_a)$, $f_b = f(x_b)$ and $f_c = f(x_c)$. Then the IQI method returns a new iterate

$$x_d = \frac{f_b f_c}{(f_a - f_b)(f_a - f_c)} x_a + \frac{f_a f_c}{(f_b - f_a)(f_b - f_c)} x_b + \frac{f_a f_b}{(f_c - f_a)(f_c - f_b)} x_c.$$

As in the case of the secant method, we want to turn this formula around and solve for f_c for a given x_d. Equivalently, we would like to define f_c to be the zero of the following equation

$$x_d = \frac{f_b y}{(f_a - f_b)(f_a - y)} x_a + \frac{f_a f_c}{(f_b - f_a)(f_b - y)} x_b + \frac{f_a f_b}{(y - f_a)(y - f_b)} x_c.$$

Multiplying both sides by $(y - f_a)(y - f_b)$, we obtain a quadratic equation in terms of y:

$$(y - f_a)(y - f_b)x_d = -\frac{f_b y}{(f_a - f_b)}x_a(y - f_b) - \frac{f_a f_c}{(f_b - f_a)}x_b(y - f_a) + f_a f_b x_c.$$

This equation simplifies to

$$g(y) = \alpha y^2 + \beta y + \gamma = 0, \tag{6}$$

where

$$\begin{aligned}\alpha &= (x_a - x_d) f_b + (x_d - x_b) f_a \\ \beta &= (x_b - x_d) f_a^2 + (x_d - x_a) f_b^2 \\ \gamma &= f_a f_b (f_b - f_a)(x_c - x_d).\end{aligned} \tag{7}$$

The two zeros of Eq. (6) are

$$y = \frac{-\beta \pm \sqrt{\beta^2 - 4\alpha\gamma}}{2\alpha}. \tag{8}$$

In the following, we will consider which sign to choose in Eq. (8). We assume that $x_a < x_b$ and $f_a < 0$ and $f_b > 0$, so the zero for the function $f(x)$ is in interval $[x_a, x_b]$. We further assume $x_a < x_d < x_c < x_b$. Under these conditions, it is easy to verify that

$$\begin{aligned}g(0) &= \gamma = f_a f_b (f_b - f_a)(x_c - x_d) < 0, \quad \text{and} \\ g(f_b) &= f_a f_b (f_b - f_a)(x_c - x_b) > 0.\end{aligned}$$

It follows that Eq. (6) always has a positive zero in $(0, f_b)$, and we will define f_c to be this zero. To find the location of the other zero in Eq. (6), we observe that when $\alpha < 0$, function $g(y) < 0$ for sufficiently large y, which means that Eq. (6) has another zero in (f_b, ∞). Similarly, Eq. (6) has another zero in $(-\infty, 0)$ when $\alpha > 0$. This discussion implies that the positive zero in $(0, f_b)$ corresponds to the plus sign in Eq. (8):

$$f_c = \frac{-\beta + \sqrt{\beta^2 - 4\alpha\gamma}}{2\alpha}.$$

Additionally, it is easy to see from (7) that β must always be positive under our restrictions. Thus, we rewrite the above formula for f_c to avoid numerical cancellation as

$$f_c = -\frac{2\gamma}{\beta + \sqrt{\beta^2 - 4\alpha\gamma}}.$$

In summary, we define

$$q(x_a, x_d, x_c, x_b, f_a, f_b) = -\frac{2\gamma}{\beta + \sqrt{\beta^2 - 4\alpha\gamma}}, \tag{9}$$

where α, β and γ are defined in Eq. (7). For $x_a < x_d < x_c < x_b$ and $f_a < 0 < f_b$, the function value $f_c = q(x_a, x_d, x_c, x_b, f_a, f_b)$ is always in $(0, f_b)$. As in secant method, if the inverse quadratic interpolation method is called with the points x_a, x_c, x_b, for a function whose value at x_a is f_a, whose value at x_b is f_b, and whose value at x_c is f_c, then the point returned by the method will be exactly x_d.

We are now ready to construct the worst-case function, $f : X \to \mathbb{R}$. We use X to denote the worst-case x values generated in Sect. 4.1 in increasing order. Let $m = |X|$, $a = X_1$, and $b = X_m$. Then $[a, b]$ is the initial interval. For the initial interval to be valid, we choose $f(a) = \epsilon$ for some $\epsilon < 0$. We define $f(b) = s(a, X_{m-1}, b, f(a))$. Given that $f(a) < 0$ and $a < X_{m-1} < b$, it is easy to verify through Eq. (5) that $f(b) > 0$. Brent's method first will evaluate the function at the points a and b. It will then choose a third point c via the secant method. By construction, we must have $c = X_{m-1}$.

At this point, Brent's method will make a number of inverse quadratic interpolation steps. Define

$$f(X_{m-1}) = q(a, X_{m-2}, X_{m-1}, X_m, f(a), f(X_m)).$$

It is clear that by construction, the next iterate from Brent's method is precisely X_{m-2}. We may continue this process until we reach X_k ($k = \log_2(\frac{b-a}{\delta})$), which is where a bisection step will be performed. Since we are bisecting the function regardless of its value at X_k, we may choose an arbitrary positive value for $f(X_k)$, such as 100. After the bisection step is performed, we simply repeat the above process on the bisected interval.

This gives us a sequence X, and a function $f : X \to \mathbb{R}$, which causes Brent's method to evaluate f for every value in X, and thus perform $\Theta(\log_2(\frac{b-a}{\delta})^2)$ iterations before it converges. The blue dots in Fig. 1 show a short X sequence along with the corresponding (positive) function values. It is clear that Brent's method is very inefficient with our function $f(x)$ because it repeatedly zooms into local minima of $f(x)$ which are not even close to any zero.

5 Proposed modifications

We propose that instead of the criteria used in Brent's method, a bisection step should be forced under the following circumstances:

1. If five successive interpolation steps fail to reduce the size of the original interval or the last interval generated by a bisection step by a factor of two.
2. If an interpolation step produces a point, b, such that $|f(b)|$ is not at least a factor of two smaller than the previous best point.

The first condition reduces the worst-case complexity of the search to $O(n)$, where n is again the number of steps required for the bisection method to terminate.

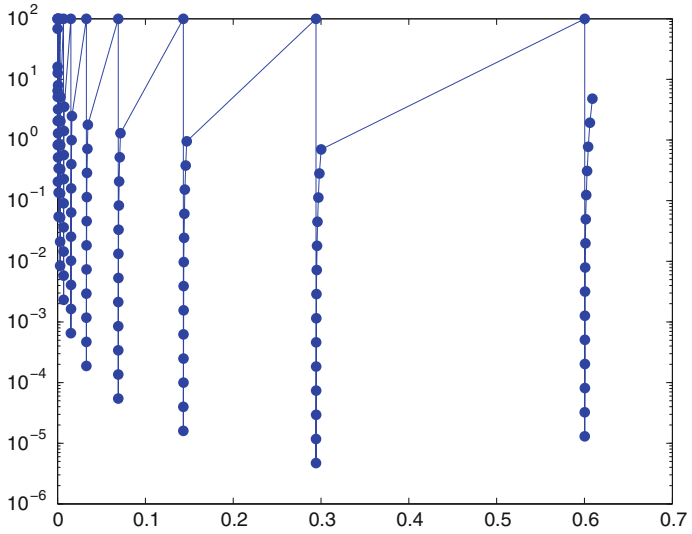

Fig. 1 124 Brent's method iterates for a root near zero, tolerance = 1e−5

Since the primary criterion for termination of the search is reduction of the interval size below some tolerance, it stands to reason that if the interval is not shrinking quickly then something has gone wrong with the search. The second condition addresses an issue with very flat functions. When presented with such functions Brent's method will shrink the interval quickly enough to satisfy the first condition, but will still be outperformed by the bisection method. The second condition effectively means that if the interpolation steps fail to quickly reduce the function value, then a bisection step will be performed. We call our method *modified zeroin*.

6 Results

Interestingly, the reduction in worst-case complexity provided by our modification to Brent's method seems to come at almost no cost to performance in general. For most of the functions and intervals tested, the difference between the number of function evaluations required for Brent's method and modified zeroin is within 2 either way (see Table 1). The functions and intervals for the comparison came from [2]. We also include a plot (Fig. 2) that compares the performance of Brent's method, modified zeroin, and the Bisection method for the function defined in Sect. 4. From the plot it is clear that Brent's method exhibits $\Theta(n^2)$ behavior, while modified zeroin exhibits $\Theta(n)$ behavior. In particular, Brent's method took 2,914 iterations for double precision accuracy, whereas bisection and modified zeroin took 77 and 85 iterations, respectively.

7 Further improvements

In addition to the proposed modifications to the search for a zero in Brent's method, we also suggest that different criteria be used for termination of the search for a zero.

Table 1 Comparison of methods

	Function	Interval	Brent's method	Modified zeroin	Bisection
1	$\sqrt{x} - \cos(x)$	[0.0,1.0]	7	8	52
2	$3(x+1)(x-5)(x-1)$	$[-2.0, 1.5]$	2	2	53
3	$3(x+1)(x-5)(x-1)$	$[-1.2, 2.5]$	8	8	54
4	$x^3 - 7x^2 + 14x - 6$	[0.0, 1.0]	8	8	51
5	$x^3 - 7x^2 + 14x - 6$	[3.2, 4.0]	13	14	47
6	$x^4 - 2x^3 - 4x^2 + 4x + 4$	$[-2.0, -1.0]$	9	9	51
7	$x^4 - 2x^3 - 4x^2 + 4x + 4$	[0.0, 2.0]	8	8	52
8	$x - 2(-x)$	[0.0, 1.0]	5	6	52
9	$e^x - x^2 + 3x - 2$	[0.0, 1.0]	7	7	52
10	$2x\cos(2x) - (x+1)^2$	$[-3.0, -2.0]$	8	8	50
11	$2x\cos(2x) - (x+1)^2$	$[-1.0, 0.0]$	9	9	52
12	$3x - e^x$	[1.0, 2.0]	9	10	50
13	$x + 3\cos(x) - e^x$	[0.0, 1.0]	7	6	52
14	$x^2 - 4x + 4 - \log(x)$	[1.0, 2.0]	9	8	49
15	$x^2 - 4x + 4 - \log(x)$	[2.0, 4.0]	10	10	51
16	$x + 1 - 2\sin(\pi x)$	[0.0, 0.5]	9	8	51
17	$x + 1 - 2\sin(\pi x)$	[0.5, 1.0]	10	10	51
18	$e^x - 2 - \cos(e^x - 2)$	$[-1.0, 2.0]$	11	11	53
19	$(x+2)(x+1)^2x(x-1)^3(x-2)$	$[-0.5, 2.4]$	13	13	53
20	$(x+2)(x+1)^2x(x-1)^3(x-2)$	$[-0.5, 3.0]$	15	15	52
21	$(x+2)(x+1)^2x(x-1)^3(x-2)$	$[-3.0, -0.5]$	13	13	52
	$(x+2)(x+1)x(x-1)^3(x-2)$	$[-1.5, 1.8]$	15	15	53
	$x^4 - 3x^2 - 3$	[1.0, 2.0]	7	8	51
	$x^3 - x - 1$	[1.0, 2.0]	9	10	51
25	$\pi + 5\sin(x/2) - x$	[0.0, 6.3]	7	7	52
	$2^{-x} - x$	[0.3, 1.0]	6	6	51
	$(2 - e^{-x} + x^2)/3 - x$	$[-5.0, 5.0]$	11	15	53
	$5x^{-2} + 2 - x$	[1.0, 5.0]	8	8	52
29	$\sqrt{\frac{e^x}{3} - x}$	[2.0, 4.0]	9	9	51
30	$5^{-x} - x$	$[-2.0, 5.0]$	8	9	54
31	$5(\sin(x) + \cos(x)) - x$	$[-2.0, 1.0]$	7	7	53
	$-x^3 - \cos(x)$	$[-3.0, 3.0]$	13	13	54
	$x^3 - 2x^2 - 5$	[1.0, 4.0]	10	9	52
	$x^3 + 3x^2 - 1$	$[-3.0, -2.0]$	8	8	50
	$x - \cos(x)$	[0.0, 1.6]	7	7	52
	$x - 8 - 2\sin(x)$	[0.0, 1.6]	7	7	52
	$e^x + 2^{-x} + 2\cos(x) - 6$	[1.0, 2.0]	8	8	51
38	$\log(x-1) + \cos(x-1)$	[1.3, 2.0]	9	8	50

Table 1 continued

Function	Interval	Brent's method	Modified zeroin	Bisection
$2x\cos(2x) - (x-2)^2$	[2.0, 3.0]	8	8	50
$e^x - 3x^2$	[3.0, 5.0]	11	12	51
$\sin(x) - e^{-x}$	[0.0, 1.0]	7	7	52
$3x - e^x$	[1.0, 2.0]	9	10	50
$x + 3\cos(x) - e^x$	[0.0, 1.0]	7	6	52
$x^2 - 4x + 4 - \log(x)$	[1.0, 2.0]	9	8	49
$x + 1 - 2\sin(\pi x)$	[0.0, 0.5]	9	8	51
$x + 1 - 2\sin(\pi x)$	[0.5, 1.0]	10	10	51

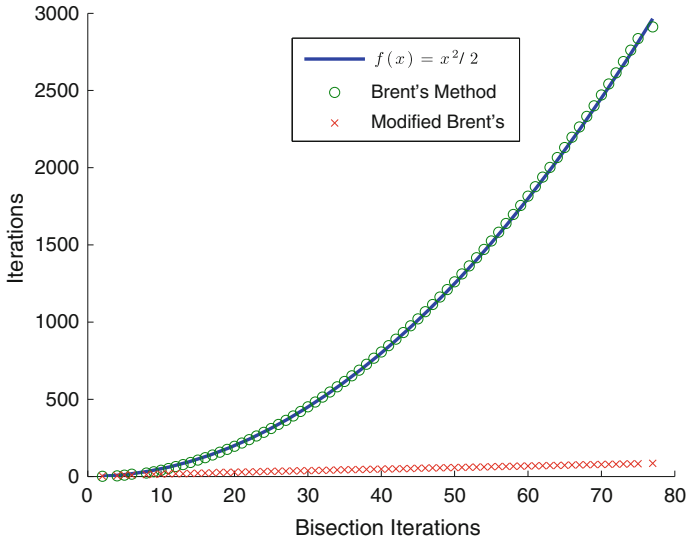

Fig. 2 Brent's method comparison

For the purposes of comparison, however, our algorithm presented above uses the same termination conditions as Brent's method.

In particular we disagree with the use of the numerical tolerance parameter, δ. The same parameter appears in Dekker's Algorithm as well and is ubiquitous in zero finding algorithms. Algorithms that employ δ to find a zero of a function f establish an interval $[a, b]$ such that $f(a)f(b) < 0$, terminate when $|b-a|/\max\{|b|, 1\} \leq \delta$, and also set δ as the minimum size by which the interval must shrink every iteration. This requires the user to either specify δ or trust that the default value of δ is appropriate. Choosing an appropriate δ is not a trivial task, nor is there a choice that is universally acceptable. For example, Matlab's default choice of δ in its *fzero* routine is the machine precision ($\approx 10^{-16}$ for double precision). However, if a user wishes to find distinct zeros that both lie very close to zero, then the default value of δ is no longer appropriate; it must be made smaller.

Ultimately δ is a confusing value to ask the user to supply, and is open to misinterpretation. Users may think that δ specifies a cutoff function value, when it does nothing of the sort. In fact the value of the function at the point that is returned may be arbitrarily large, as long as a sign change occurs across the interval and the interval is sufficiently small (something that would occur if the method converged to a pole).

Instead of δ we propose two modifications, one to the algorithm and the other to the way that users supply functions to the algorithm. First, the algorithm should terminate either when it finds an exact zero of the function, or when the ends of the interval straddle adjacent floating point numbers. Second, the user should determine an error associated with computing the function, and when the value of the function is sufficiently small so that the error makes it indistinguishable from a zero, set the value of the function identically to zero. In general, this should be a much easier task for the user than determining a good choice for δ, and even if the user does not do so the worst case is that the algorithm takes a few more iterations to find adjacent floating point numbers before stopping.

Acknowledgments The authors thank Prof. William Kahan and Dr. Hanyou Chu for a number of enlightening discussions while the ideas in this paper were in development.

References

1. Brent, R.: Algorithms for Minimization Without Derivatives. Dover Publications, Mineola (2002)
2. Burden, R.L., Faires, J.D.: Numerical Analysis. Brooks Cole, Pacific Grove (2005)
3. Dekker, T.J.: Finding a zero by means of successive linear interpolation. In: Dejon, B., Henrici, P. (eds.) Constructive Aspects of the Fundamental Theorem of Algebra. Wiley, London (1969)
4. Kahan, W.: Lecture notes on real root-finding (2009)
5. Moler, C.: Numerical Computing with MATLAB. SIAM, Auckland (2008)
6. Shengguo, L., Xiangke, L., Lizhi, C.: A new fourth-order iterative method for finding multiple roots of nonlinear equations. Appl. Math. Comput. **215**(3), 1288–1292 (2009). doi:10.1016/j.amc.2009.06.065. http://www.sciencedirect.com/science/article/pii/S0096300309006444
7. Wang, X., Liu, L.: New eighth-order iterative methods for solving nonlinear equations. J. Comput. Appl. Math. **234**(5), 1611–1620 (2010). doi:10.1016/j.cam.2010.03.002. http://www.sciencedirect.com/science/article/pii/S0377042710001433
8. Yun, B.I.: Transformation methods for finding multiple roots of nonlinear equations. Appl. Math. Comput. **217**(2), 599–606 (2010). doi:10.1016/j.amc.2010.05.094. http://www.sciencedirect.com/science/article/pii/S0096300310006673
9. Yun, B.I.: Solving nonlinear equations by a new derivative free iterative method. Appl. Math. Comput. **217**(12), 5768–5773 (2011). doi:10.1016/j.amc.2010.12.055. http://www.sciencedirect.com/science/article/pii/S0096300310012579