Dekker-Brent Algorithm

Julius Andrews

May 15, 2025

1 Abstract

The varying methods for fixed point iterations, particularly root finding algorithms, have varying strengths and weaknesses. Dekker's hypothesis and eventual algorithm, attempts to use the best method each iteration by setting particular tests. Brent expands on this idea by incorporating inverse quadratic interpolation. This paper will show the progression of ideas and the real world implementations of the Dekker-Brent algorithm.

2 The simple methods

The Dekker Algorithm uses both the bisection and secant methods in tandem. The bisection method uses the *theorem of zeros of continuous functions*. This corollary to the Intermediate Value Theorem states that any continuous function in $[a, b]$ which satisfies $f(a)f(b) < 0$, then f has at least one zero in (a, b). The bisection method starts with a valid interval and halves the interval every iteration. Mathematically: $m = \frac{a_k+b_k}{2}$ where k is the iteration count and m is the approximate root. Bisection method always converges if the starting condition holds. However, it converges linearly.

The secant method acts as a variant of Newton's method, by calculating $f'(x^{(k)})$ with an incremental ratio based on previously computed values of f. Mathematically, the approximate root at the kth iteration is: $x^{(k+1)} = x^{(k)} - (\frac{f(x^{(k)})-f(x^{(k-1)})}{x^{(k)}-x^{(k-1)}})^{-1} f(x^{(k)})$. For simple roots, the secant method converges super-linearly; for multiple roots, the convergence rate is only linear. However, the initial points should be sufficiently close to the root for the method to converge.

3 Dekker's Method

Dekker uses the strengths and weakness of each method to design his new method. His idea is to pick initial points that satisfy bisection method, then calculate the secant method. If the secant method is not sufficiently close to the zero, then the reliable but slow bisection method will be used instead.

The algorithm is:

$$s = \begin{cases} b_k - \frac{b_k - b_{k-1}}{f(b_k) - f(b_{k-1})} f(b_k), & \text{if } f(b_k) \neq f(b_{k-1}) \\ m & \text{if } f(b_k) = f(b_{k-1}) \end{cases}$$

$$m = \frac{a_k + b_k}{2}$$

if $|b_k| < |s| < |m|$ (if its "sufficiently close") then $b_{k+1} = s$, otherwise $b_{k+1} = m$

4 Dekker-Brent Method

Brent modifies Dekker's idea by replacing the secant method with inverse quadratic interpolation, which uses the three preceding iterates, x_{n-2}, x_{n-1}, and x_n, with their function values, $f(x_{n-2})$, $f(x_{n-1})$, and $f(x_n)$. Applying the Lagrange interpolation formula to do quadratic interpolation on the inverse of f yields:

$$x_{n+1} = \frac{f(x_{n-1})f(x_n)}{(f(x_{n-2}) - f(x_{n-1}))(f(x_{n-2}) - f(x_n))} x_{n-2}$$

$$+ \frac{f(x_{n-2})f(x_n)}{(f(x_{n-1}) - f(x_{n-2}))(f(x_{n-1}) - f(x_n))} x_{n-1}$$

$$+ \frac{f(x_{n-2})f(x_{n-1})}{(f(x_n) - f(x_{n-2}))(f(x_n) - f(x_{n-1}))} x_n$$

Using this root finding algorithm brings a different set of tests. Since the inverse quadratic interpolation uses the three preceding iterates, there are two cases for the test:

The following inequalities for a given δ:

If the previous step used bisection: $|\delta| < |b_k - b_{k-1}|$ and $|s - b_k| < \frac{1}{2}|b_k - b_{k-1}|$

If the previous step used interpolation: $|\delta| < |b_{k-1} - b_{k-2}|$ and $|s - b_k| < \frac{1}{2}|b_{k-1} - b_{k-2}|$

If the inequalities are true then the inverse quadratic interpolation is used. Otherwise, the bisection method is used. If the function f is well-behaved, then Brent's method will usually proceed by either inverse quadratic (thanks to the inverse quadratic interpolation), or superlinearly (at worst).

This modification ensures that at the k^{th} iteration, a bisection step will be performed in at most $2\log_2(|b_{k-1} - b_{k-2}|/\delta)$ additional iterations, because the above conditions force consecutive interpolation step sizes to halve every two iterations, and after at most $2\log_2(|b_{k-1} - b_{k-2}|/\delta)$ iterations, the step size will be smaller than δ, which invokes a bisection step.

5 Code

```
input a, b, and (a pointer to) a function for f
calculate f(a)
calculate f(b)
if f(a)f(b) ≥ 0 then
    exit function because the root is not bracketed.
end if
if |f(a)| < |f(b)| then
    swap (a,b)
end if
c := a
set mflag
repeat until f(b or s) = 0 or |b − a| is small enough (convergence)
    if f(a) ≠ f(c) and f(b) ≠ f(c) then
        s := af(b)f(c)/((f(a)−f(b))(f(a)−f(c))) + bf(a)f(c)/((f(b)−f(a))(f(b)−f(c))) + cf(a)f(b)/((f(c)−f(a))(f(c)−f(b))) (inverse quadratic interpolation)
    else
        s := b − f(b)(b−a)/(f(b)−f(a)) (secant method)
    end if
    if (condition 1) s is not between (3a + b)/4 and b or
       (condition 2) (mflag is set and |s−b| ≥ |b−c|/2) or
       (condition 3) (mflag is cleared and |s−b| ≥ |c−d|/2) or
       (condition 4) (mflag is set and |b−c| < |δ|) or
       (condition 5) (mflag is cleared and |c−d| < |δ|) then
        s := (a+b)/2 (bisection method)
        set mflag
    else
        clear mflag
    end if
    calculate f(s)
    d := c  (d is assigned for the first time here; it won't be used above on the first iteration
because mflag is set)
    c := b
    if f(a)f(s) < 0 then
        b := s
    else
        a := s
    end if
    if |f(a)| < |f(b)| then
        swap (a,b)
    end if
end repeat
output b or s (return the root)
```

Dekker-Brent Algorithm in MatLab

6 Works Cited

https://mathworld.wolfram.com/BrentsMethod.html

https://en.wikipedia.org/wiki/Brent%27s_method

https://gitlab.com/gnu-octave/octave/-
/blob/default/scripts/optimization/fzero.m?ref_type=heads