

Root-Finding Methods

♣ **Bisection Method.** The bisection method is a root-finding tool based on the *Intermediate Value Theorem*. The method is also called the **binary search method**.

CALCULUS. Suppose the function $F(x)$ is continuous on $[a_0, b_0]$ and $F(a_0)F(b_0) \leq 0$. Then there exists an $x^* \in [a_0, b_0]$ such that $F(x^*) = 0$.

Algorithm A

- Step 1. Define $0^* > 0$ as your zero, N as the maximum number of iterations and set $T := a_0$;
- Step 2. set $x_0 := b_0$ and $n := 1$;
- Step 3. find $x_n = \frac{T+x_{n-1}}{2}$;
- Step 4. if $|F(x_n)| := 0^*$ or $n > N$, then $x^* := x_n$ and Stop;
- Step 5. if $S := F(x_n)F(x_{n-1}) < 0$, then set $n := n + 1$, $T := x_{n-1}$ and Goto Step 3.;
- Step 6. if $S := F(x_n)F(x_{n-1}) > 0$, then set $n := n + 1$ and Goto Step 3.

we have:

$$x^* = \lim_{n \rightarrow \infty} x_n$$

$$|x_n - x^*| \leq \frac{b_0 - a_0}{2^n} = 2^{-n}(b_0 - a_0).$$

If $\epsilon = 10^{-r}$, then in order to accept x_N as x^* we need to choose an integer N which satisfies the condition $\log 2^{-N}(b_0 - a_0) \leq \log 10^{-r}$. This implies that

$$N \geq \frac{r + \log(b_0 - a_0)}{\log 2}.$$

Algorithm B

- Step 1. Define $0^* > 0$ as your zero, N as the maximum number of iterations and set n to zero;
- Step 2. set $x := \frac{a_n+b_n}{2}$;
- Step 3. if $|F(x)| := 0^*$ or $n > N$, then $x^* := x$ and Stop;
- Step 4. if $S := F(a_n)F(b_n) < 0$, set $a_{n+1} := a_n$, $b_{n+1} := x$ and Goto Step 2.;
- Step 5. if $S := F(a_n)F(b_n) > 0$, set $a_{n+1} := x$, $b_{n+1} := b_n$ and Goto Step 2.

We have:

$$x^* = \lim_{n \rightarrow \infty} a_n = \lim_{n \rightarrow \infty} b_n$$

$$N \geq r + \log(b_0 - a_0).$$

Advantage. Easy implementation, guaranteed convergence.

Disadvantage. It requires a_0 and b_0 to satisfy $F(a_0)F(b_0) < 0$. Also very slow convergence. In general, you can't expect to hit the solution exactly; you can only see if one of the *Stopping Criteria* such as

$$|F(x)| < \epsilon, \quad |x_n - x_{n-1}| < \epsilon \quad \text{or} \quad N \geq \frac{r + \log(b_0 - a_0)}{\log 2}.$$

are satisfied.

♣ **Newton-Raphson Method.** The Newton-Raphson (or simply Newton's) method is one of the most powerful numerical methods for solving a root-finding problem $F(x) = 0$.

CALCULUS. Suppose that $F(x)$ is twice continuously differentiable in some interval about a point x^* such that $F(x^*) = 0$, $F'(x^*) \neq 0$ and $F(x)$ is *nearly linear* near x^* (i.e., $|F''(x)| < \epsilon$ near x^*).

Let x_0 be a point near x^* . Consider the first degree Taylor polynomial for $F(x)$, expanded about x_0 ,

$$F(x) = F(x_0) + (x - x_0)F'(x_0) + \frac{(x - x_0)^2}{2}F''(\xi(x)),$$

where $\xi(x)$ lies between x and x_0 . Since $F(x^*) = 0$ and $F(x)$ is nearly linear about x^* , we have

$$0 \approx F(x_0) + (x^* - x_0)F'(x_0)$$

or

$$x^* \approx x_0 - \frac{F(x_0)}{F'(x_0)}.$$

Therefore

$$x_1 = x_0 - \frac{F(x_0)}{F'(x_0)}$$

should be a better approximation of x^* than x_0 .

Algorithm

Step 1. Define $0^* > 0$ as your zero, N as the maximum number of iterations, choose x_0 near x^* and set $n := 0$;

Step 2. if $|F(x_n)| := O^*$ or $n > N$, then $x^* := x_n$ and Stop;

Step 3. define $x_{n+1} := x_n - \frac{F(x_n)}{F'(x_n)}$, then set $n := n + 1$ and Goto Step 2.

Advantage. Very fast convergence ultimately.

Disadvantage. It often requires a good initial guess for x_0 . It requires two function evaluations ($F(x_n)$ and $F'(x_n)$) at each iteration. If $F(x)$ is a polynomial, then the Horner's method can be used to obtain both $F(x_n)$ and $F'(x_n)$.

Note. If $F(x) = \sqrt{|x|}$, then $F(0) = 0$, so $x^* = 0$. Let $x_0 \neq 0$, then $x_1 = -x_0$ and $x_2 = x_0$; this way you obtain $x_{2n} = x_0$ and $x_{2n+1} = -x_0$. Therefore Newton's method never works for this particular function.

♣ **Secant Method.** The secant method is a variation of Newton's method. We need to make the same assumptions for $F(x)$ as we did for the Newton's method.

CALCULUS. Let x_0 and x_1 be two points near x^* . If $|x_1 - x_0| < \epsilon$, then

$$F'(x_1) = \lim_{x_1 \rightarrow x_0} \frac{F(x_1) - F(x_0)}{x_1 - x_0}.$$

Consider the Taylor polynomial for $F(x)$, expanded about x_1 ,

$$F(x) = F(x_1) + (x - x_1)F'(x) + \frac{(x - x_1)^2}{2}F''(\xi(x)),$$

where $\xi(x)$ lies between x and x_1 . By replacing $F'(x_1)$ with $\frac{F(x_1) - F(x_0)}{x_1 - x_0}$ in the above formula and using the fact that $F(x^*) = 0$ and $F(x)$ is nearly linear about x^* , we obtain

$$0 \approx F(x_1) + (x^* - x_1) \frac{F(x_1) - F(x_0)}{x_1 - x_0}$$

or

$$x^* \approx x_1 - \frac{x_1 - x_0}{F(x_1) - F(x_0)} F(x_1).$$

Therefore by choosing

$$x_2 = x_1 - \frac{x_1 - x_0}{F(x_1) - F(x_0)} F(x_1)$$

we should get a better approximation of x^* than x_1 .

Algorithm

Step 1. Define $0^* > 0$ as your zero, N as the maximum number of iterations, choose x_0 and x_1 near x^* and set $n := 1$;

Step 2. if $F(x_n) := O^*$ or $n > N$, then $x^* := x_n$ and Stop;

Step 3. define $x_{n+1} := x_n - \frac{x_n - x_{n-1}}{F(x_n) - F(x_{n-1})}F(x_n)$, then set $n := n + 1$ and Goto Step 2.

Advantage. Fast convergence; only one function evaluation.

Disadvantage. It requires some arithmetic per iteration; care must be taken to prevent divergence.

♣ Comparison of the Three Methods. In this section we explain some of the characteristics of the bisection, Newton and secant methods.

Speed. The speed of a root-finding method is measured by the number of iterations involved in order to reach a satisfactory solution.

| | |
|-------------------|-----------|
| Newton: | very fast |
| Secant: | fast |
| Bisection: | very slow |

Error Analysis. The absolute error at the n-th iteration is $|x_n - x^*|$ and is denoted by E_n .

Suppose $\{x_n\}_{n=0}^\infty$ is a sequence that converges to x^* . If there exist $\lambda > 0$ and $\alpha > 0$ such that

$$\lim_{n \rightarrow \infty} \frac{E_{n+1}}{E_n^\alpha} = \lambda,$$

then $\{x_n\}_{n=0}^\infty$ is said to converge to x^* of order α , with asymptotic error constant λ .

1. If $\alpha = 1$, the method is called *linear*.
2. If $\alpha = 2$, the method is called *quadratic*.

| | |
|-------------------|---|
| Newton: | quadratic convergence $\alpha = 2$ |
| Secant: | $E_{n+1} \approx \lambda E_n E_{n-1}$ $\alpha = (1 + \sqrt{5})/2$ |
| Bisection: | Max $\{ a_n - x^* , b_n - x^* \} < a_0 - b_0 2^{-n}$ |

Efficiency. The efficiency of a method is the cost per iteration.

| | |
|-------------------|--|
| Bisection: | 1 function evaluation, 1 multiplication and a little logic per iteration. |
| Secant: | 1 function evaluation and some arithmetic per iteration. |
| Newton: | 2 function evaluations (F and F') and some arithmetic per iteration. |

Reliability:

| | |
|-------------------|--|
| Bisection: | Convergence is assured once appropriate a_0 and b_0 are found. |
| Newton: | Needs a good initial guess for x_0 . |
| Secant: | Needs good choice of x_0 and x_1 . |

Summary. For general use, the bisection method is far too slow. The other two methods are fast enough in general, but care must be taken to prevent divergence. The fact that the secant method does not require evaluation of $F'(x)$ can be a crucial advantage. Secant method is regarded as best general purpose method.

Ideal Method. Combine global reliability with fast local convergence while achieving optimal efficiency at each iteration.

If, say one is constructing a computer code which should be fast, reliable and efficient, then there is no need to restrict oneself to the “pure” form of the above (or any other) methods.

Example. Find (to 9 decimal places) the square root of 2.

Let

$$F(x) = x^2 - 2 = [(1)x + 0]x - 2.$$

| | Bisection | | Newton | secant |
|-----|------------------|------------|---------------|---------------|
| n | a_n | b_n | x_n | x_n |
| 0 | 1.0 | 2.0 | 1.0 | 0.5 1.0 |
| 1 | 1.0 | 1.5 | 1.50 | 1.444444444 |
| 2 | 1.25 | 1.5 | 1.416666667 | 1.409090909 |
| 3 | 1.375 | 1.5 | 1.414215686 | 1.414159292 |
| 4 | 1.375 | 1.4375 | 1.414213562 | 1.414213661 |
| 5 | 1.40625 | 1.4375 | | 1.414213562 |
| ⋮ | ⋮ | ⋮ | | |
| ⋮ | ⋮ | ⋮ | | |
| 29 | 1.414213561 | 1.41421362 | | |

♣ **Other Iterative Root-Finding Methods.** All root-finding methods are basically based on two geometric ideas:

- (i) Bracketing the initial interval and reducing the size of the brackets at each iteration (bisection method).
- (ii) Using slope (Newton and secant methods).

It is important to note that every iterative method must be provided with at least an initial guess x_0 , and will have difficulty of finding x^* accurately. In general *Slope methods* are faster than *Bracketing* methods, but the latter guarantee the convergence.

♡ **Bracketing Methods.** What all bracketing methods have in common is that they can be used only if $F(x)$ has opposite signs on either side of x^* , What distinguish one method from another is the strategy used to get solution x^* . A solution p found by a bracketing method is called a *test point*.

♠ **Regula Falsi (or False Position) Method.** The bisection method does not use values of $f(x)$; only their sign. However, the values could be exploited. One way to use values of $f(x)$ is to bias the search according to value of $f(x)$; so instead of choosing the point c as the midpoint of a and b , choose it as the point of intersection of the x-axis and the secant line through $(a, F(a))$ and $(b, F(b))$.

Assume that $F(x)$ is continuous such that for some a and b , $F(a)F(b) < 0$. The formula for the secant line is

$$\frac{y - F(b)}{x - a} = \frac{F(b) - F(a)}{b - a}$$

pick $y = 0$, the intercept is: $x_1 = c = b - \frac{(b - a)}{F(b) - F(a)}F(b)$.

If $f(c) = 0$, then $x^* = c$.

If $f(a)f(c) < 0$, a zero lies in the interval $[a, c]$, so we set $b = c$.

If $f(b)f(c) < 0$, a zero lies in the interval $[c, b]$, so we set $a = c$.

Then we use the secant formula to find the new approximation for x^* :

$$x_2 = c = b - \frac{(b - a)}{F(b) - F(a)}F(b).$$

We repeat the process until we find an x_n with $|x_n - x_{n-1}| < Tol_x$ or $|f(x_n)| < Tol_y$.

Algorithm

Step 1. Define $0^* > 0$ as your zero, N as the maximum number of iterations and set n to zero;

Step 2. set $x := b_n - \frac{(b_n - a_n)}{F(b_n) - F(a_n)} F(b_n)$;

Step 3. if $|F(x)| := 0^*$ or $n > N$, then $x^* := x$ and Stop;

Step 4. if $S := F(a_n)F(b_n) < 0$, set $a_{n+1} := a_n$, $b_{n+1} := x$ and Goto Step 2;

Step 5. if $S := F(a_n)F(b_n) > 0$, set $a_{n+1} := x$, $b_{n+1} := b_n$ and Goto Step 2.

♠ **Brent's Method.** This method is a modified version of the Regula Falsi method which involves $\frac{1}{2}(x_n - x_{n-1})$, secant method and bracketing techniques.

Brent has taken great care to avoid underflow and overflow difficulties with this method; but the method is somewhat complicated to explain here.

♡ **Slope Methods.**

♠ **Modified Newton's Method.** A Proposed generalization of the Newton's method is

$$x_{n+1} = x_n - \omega \frac{F(x_n)}{F'(x_n)},$$

where the constant ω is an *acceleration factor* chosen to increase the rate of convergence. It is known that if x^* is a double root of $F(x)$, then x_n can go faster to x^* if we choose $\omega = 2$. When the multiplicity of x^* is m , then we should choose $\omega = m$.

♠ **Steffensen's Method.**

$$x_{n+1} = x_n - \frac{F(x_n)}{S(x_n)}, \quad \text{where} \quad S(x) = \frac{F(x + F(x)) - F(x)}{F(x)}.$$

This method is as fast as Newton's method and does not require the evaluation of $F'(x)$.

♠ **Olver's Method.**

$$x_{n+1} = x_n - \frac{F(x_n)}{F'(x_n)} - \frac{1}{2} \left(\frac{[F(x_n)]^2 F''(x_n)}{[F'(x)]^3} \right).$$

This method is cubically convergent, therefore it is faster than Newton's method which is only quadratically convergent.

♠ **Halley's Method.**

$$x_{n+1} = x_n - \frac{1}{a_n}, \quad \text{where} \quad a_n = \frac{F'(x_n)}{F(x)} - \frac{1}{2} \left(\frac{F''(x_n)}{F'(x_n)} \right).$$

This method is slower than the previous method but involves less arithmetic, and is considered very good for finding real root of polynomials along with the Horner's method.

♡ **Modified Newton's Method.** To avoid computing the derivative at each step, it has been proposed to replace $F'(x_n)$ with $F'(x_0)$ in all steps. The convergence of Newton's method is quadratic, but then the convergence of the proposed method is only linear. To obtain more rapid convergence of Newton's method, it has also been proposed that the derivative be computed every other step. This method is given by

$$x_{2n+1} = x_{2n} - \frac{F(x_{2n})}{F'(x_{2n})} \quad \text{and} \quad x_{2n+2} = x_{2n+1} - \frac{F(x_{2n+1})}{F'(x_{2n+1})}.$$

♡ **Modified Secant Methods.** It has been proposed that the slope of the secant line be replaced by

$$\frac{F(x_n) - F(x_0)}{x_n - x_0}.$$

Under certain conditions, x_n converges to x^* .

♠ **Müller's Method.** This is a useful method for finding *real* and *complex* roots of a Polynomial. Müller's method is a generalization of the approach that led to the secant method.

Given three points x_0 , x_1 and x_2 , a quadratic polynomial is constructed which passes through these three points $\{(x_i, F(x_i))\}$ ($i = 0, 1, 2$). One of the roots of this polynomial is used as an improved estimate for x^* . The method uses the following formula:

$$x_{n+1} = x_n - \frac{F(x_n)}{[(F'(x_n))^2 - F(x_n)F''(x_n)]^{1/2}}.$$

The method converges cubically for a simple root but the convergence becomes linear at a multiple root.

With the secant method, real choices of x_0 and x_1 will lead to a real value of x_2 . But with Müller's method, real choices of x_0 , x_1 and x_2 may lead to complex roots of $F(x)$. This is an important aspect of Müller's method. For this reason it is widely used for finding the roots of polynomials.

♠ **Bairstow's Method.** The objective is to find quadratic polynomials

$$Q(x) = x^2 - rx - s$$

which are factors of the polynomial $P(x)$. Bairstow observed that the *Synthetic division* can be performed with $Q(x)$ to obtain the *real* and *complex* roots of the polynomial $P(x)$.

♣ **Fixed-Point Iterations.** Finding the real zero of a function $F(x)$ is the same as finding the intersection of $F(x)$ and the x-axis. Consider the function $G(x) = F(x) + x$, the point of intersection of $G(x)$ and $y = x$ (i.e., $G(x) = x$) is exactly the same as the real zero of $F(x)$. Any solution x^* to the equation $G(x) = x$ is called a fixed point of the function $G(x)$.

CALCULUS. Let $G(x)$ be a continuous function on $[a, b]$ where $G'(x) \leq k < 1$ for all $x \in (a, b)$. If $x_{n+1} = G(x_n)$, then

$$|x_{n+1} - x_n| = |G(x_n) - G(x_{n-1})| \approx |G'(x)| |x_n - x_{n-1}| \leq k |x_n - x_{n-1}| \cdots < k^n |x_1 - x_0| < |x_1 - x_0|.$$

Thus $\lim_{n \rightarrow \infty} |x_{n+1} - x_n| = 0$. Hence $G(x)$ has a unique fixed point x^* on (a, b) .

Algorithm

- Step 1. Define $\epsilon > 0$ as your zero, N as the maximum number of iterations, set $n := 0$ and choose x_n near x^* ;
- Step 2. find $x_{n+1} = G(x_n)$;
- Step 3. if $|x_{n+1} - x_n| := \epsilon$ or $n > N$, then $x^* := x_n$ and Stop;
- Step 4. set $n := n + 1$ and Goto Step 2.

Advantage. It requires one function evaluation. When $G'(x) \leq k < 1$ on (a, b) , the convergence to the unique solution is guaranteed. The convergence is quadratic, when $G''(x)$ is continuous on (a, b) .

Disadvantage. When It is used to find the root of the function $F(x)$, it is not always clear how to construct $G(x)$ for a fast convergence. If the range of $G'(x)$ on (a, b) is not known, then the convergence is not guaranteed.

Example. The equation $x^3 + 4x^2 - 10 = 0$ has a unique root in $[1, 2]$. There are many ways to change the equation to the form $G(x) = x$ by simple algebraic manipulation. For example

$$\begin{aligned} (a) \quad G_1(x) &= x - x^3 - 4x^2 + 10 = x, & (b) \quad G_2(x) &= \left(\frac{10}{x} - 4x \right)^{1/2}, \\ (c) \quad G_3(x) &= \frac{1}{2}(10 - x^3)^{1/2} = x, & (d) \quad G_4(x) &= \left(\frac{10}{4 + x} \right)^{1/2}, \\ (e) \quad G_5(x) &= x - \frac{x^3 + 4x^2 - 10}{3x^2 + 8x} = x. \end{aligned}$$

Note that $G_5(x)$ is obtain from the Newton's formula by replacing x_n with x and x_{n+1} with $G_5(x)$ With $x_0 = 1.5$, the following table lists the results of the fixed-point iteration method for all five choices of $G(x)$.

| n/x_n | (a) | (b) | (c) | (d) | (e) |
|---------|----------|-----------------|-------------|-------------|-------------|
| 0 | 1.5 | 1.5 | 1.5 | 1.5 | 1.5 |
| 1 | -0.875 | 0.8165 | 1.286953768 | 1.348399725 | 1.373333333 |
| 2 | 6.732 | 2.9969 | 1.402540804 | 1.367376372 | 1.365262015 |
| 3 | -469.7 | $(-8.65)^{1/2}$ | 1.345458374 | 1.364957015 | 1.365230014 |
| 4 | diverges | diverges | 1.375170253 | 1.365264748 | 1.365230013 |
| 5 | | | 1.360094193 | 1.365225594 | |
| ⋮ | | | ⋮ | ⋮ | |
| ⋮ | | | ⋮ | ⋮ | |
| 15 | | | 1.365223680 | 1.365230013 | |
| ⋮ | | | ⋮ | | |
| 30 | | | 1.365230013 | | |

Trivia Questions:

1. Name a method which is often used in the “ built-in” root-finding routine of a programmable calculator.
2. Can you use Newton's method to find complex roots?
3. Bairstow's method is not recommended for hand calculation unless ...?
4. What is a test point and why some point are called test points?

Answers To The Trivia Questions.

1. Dohtem noitcesib.
2.)emosrebmuc yrev(margorp nartroF a ni)stnemetats tuptuo/tupni eht gniyfidom ylbatus dna(xelpmoc ot selbairav laer eht gnignahc yb.
3. Nwonk era sesseug laitini doog.
4. Tniop tset a dellac si tniop gnitluser eht os ;yltcaxe noituolos eht tih ot tcepxe lareneg ni t'nac eno ,dohtem gniteckarb a ni.