Root Finding Problems

P. Sam Johnson

March 5, 2020

Overview

The problem of finding an approximation to the root of an equation can be traced back at least as far as 1700 BC.

Root-finding problem is of the most basic problems of numerical approximation. This process involves finding a **root** (or **zero**, or **solution**), of an equation of the form $f(x) = 0$, for a given function f. Often it will not be possible to solve such root-finding problems analytically.

When this occurs we turn to numerical methods to approximate the solution. The methods employed are usually **iterative**.

Generally speaking, algorithms for solving problems numerically can be divided into two main groups: **direct methods** and **iterative methods**.

Overview

Direct methods are those which can be completed in a predetermined finite number of steps.

Iterative methods are methods which converge to the solution over time.

These algorithms run until some convergence criterion is met. When choosing which method to use, one important consideration is how quickly the algorithm converges to the solution or the method's **convergence rate**.

We discuss some iterative methods with their convergence rates in three lectures.

Impressive Good Approximation to Square Root of 2 (in the period of Babylonians)

This is a Babylonian clay tablet from around 1700 BC. It's known as *YBC*7289, since it's one of many in the Yale Babylonian Collection. It's a diagram of a square with one side marked as having length 1/2. They took this length, multiplied it by the square root of 2, and got the length of the diagonal.

Since the Babylonians used base 60, they thought of 1/2 as 30/60. But since they hadn't invented anything like a "decimal point", they wrote it as 30. More precisely, they wrote it as this:

$$1 + \frac{24}{60} + \frac{51}{60^2} + \frac{10}{60^3} \approx 1.41421297....$$

This is an impressively good approximation to $\sqrt{2} \approx 1.41421356, \ldots,$ which is accurate to within 10^{-5}.

Introduction : Another Example

The growth of a population can be modeled over short periods of time by assuming that the population grows continuously with time at a rate proportional to the number present at that time.

If we let $N(t)$ denote the number at time t and λ denote the constant birth rate of the population, then the population satisfies the differential equation $\frac{dN(t)}{dt} = \lambda N(t)$. The solution to this equation is $N(t) = N_0 e^{\lambda t}$, where N_0 denotes the initial population. This exponential model is valid only when the population is isolated, with no immigration.

If immigration is permitted at a constant rate ν, then the differential equation becomes

$$\frac{dN(t)}{dt} = \lambda N(t) + \nu,$$

whose solution is

$$N(t) = N_0 e^{\lambda t} + \frac{\nu}{\lambda}(e^{\lambda t} - 1).$$

Approximate Solution

Suppose a certain population contains, $1,000,000$ individuals initially, that $435,000$ individuals immigrate into the community in the first year, and that $1,564,000$ individuals are present at the end of one year.

To determine the birth rate of this population, we must **solve for** λ in the equation

$$1,564,000 = 1,000,000e^{\lambda} + \frac{435,000}{\lambda}(e^{\lambda} - 1).$$

Numerical methods are used to approximate solutions of equations of this type, when the exact solutions cannot be obtained by algebraic methods.

Algebraic Equations

An **algebraic expression** is a mathematical phrase that can contain ordinary numbers, variables (like x or y) and operators (like addition, subtraction, multiplicaiton, division, etc).

An expression is **algebraic** if it involves a finite combination of numbers, variables and algebraic operations (addition, subtraction, multiplication, division, raising to a power).

Two important types of such equations are **linear equations**, in the form $y = ax + b$, and **quadratic equations**, in the form $y = ax^2 + bx + c$.

A **solution** is a numerical value that makes the equation a true statement when substituted for a variable.

For example, $x^5 - 3x + 1 = 0$ is an algebraic equation with integer coefficients and $y^4 + \frac{xy}{2} = \frac{x^3}{3} - xy^2 + y^2 - \frac{1}{7}$ is a multivariate polynomial equation over the rationals.

Polynomial and Transcendental Equations

A **polynomial** is an expression consisting of variables (or indeterminates) and coefficients, that involves only the operations of addition, subtraction, multiplication, and non-negative integer exponents.

An example of a polynomial of a single indeterminate (or variable) x, is $x^2 - 4x + 7$, which is a quadratic polynomial. Polynomials appear in a wide variety of areas of mathematics and science.

Transcendental equation is an equation containing transcendental functions (for example, exponential, logarithmic, trigonometric, or inverse trigonometric functions) of the unknowns. Examples of transcendental equations are

$$\sin x + \log x = x \quad \text{and} \quad 2x - \log x = \arccos x.$$

Iterative Method

The limitations of analytical methods for the solution of equations have necessitated the use of iterative methods.

An iterative method begins with an approximate value of the root which is generally obtained with the help of the intermediate value property (IVP). This initial approximation is then successively improved iteration by iteration and this process stops when the desired level of accuracy is achieved.

The various iterative methods begin their process with one or more initial approximations.

Based on a number of initial approximations used, these iterative methods are divided into two categories: **bracketing method** and **open-end method**.

Bracketing Method

Bracketing methods begin with 2 initial approximations which bracket the root. Then the width of this bracket is systematically reduced until the root is reached to desired accuracy.

The commonly used methods in the category are:

1. Graphical method
2. Bisection method
3. Method of false position.

Open-end Method

Open-end methods are used on formulae which require a single starting value or two starting values which do not necessarily bracket the root.

The following methods fall under the category:

1. Secant method
2. Iteration method
3. Newton-Raphson method.

Approximations in Numerical Analysis

Many algorithms in numerical analysis are iterative methods that produce a sequence $\{x_n\}$ of approximate solutions which, ideally, converges to a limit that is the exact solution as n approaches ∞.

Because we can only perform a finite number of iterations, we cannot obtain the exact solution, and we have **computational error**.

If our iterative method is properly designed, then this computational error will approach zero as n approaches ∞. However, it is important that we obtain a sufficiently accurate approximate solution using as few computations as possible.

Therefore, it is not practical to simply perform enough iterations so that the computational error is determined to be sufficiently small, because it is possible that **another method** may yield comparable accuracy with less computational effort.

Big O Notation

The total computational effort of an iterative method depends on both the "effort per iteration" and the "number of iterations performed".

Therefore, in order to determine the amount of computation that is needed to attain a given accuracy, we must be able to measure the error in n as a function of n. The more rapidly this function approaches zero as n approaches ∞, the more rapidly the sequence of approximations $\{x_n\}$ converges to the exact solution, and as a result, **fewer iterations are needed to achieve a desired accuracy**.

We now introduce some terminology that will aid in the discussion of the convergence behavior of iterative methods. **Big O notation** characterizes functions according to their growth rates: different functions with the same growth rate may be represented using the same O notation.

Order of Function

The letter O is used because the **growth rate of a function** is also referred to as **order of the function**. A description of a function in terms of big O notation usually only provides an upper bound on the growth rate of the function.

Let f and g be two functions defined on some subset of the real numbers. One writes

$$f(x) = O(g(x)) \text{ as } x \to \infty$$

if and only if there is a positive constant M such that for all sufficiently large values of x, the absolute value of $f(x)$ is at most M multiplied by the absolute value of $g(x)$.

That is, $f(x) = O(g(x))$ if and only if there exists a positive real number M and a real number x_0 such that

$$|f(x)| \leq M|g(x)| \text{ for all } x \geq x_0.$$

In many contexts, the assumption that we are interested in the growth rate as the variable x goes to infinity, is left unstated, and one writes more simply that $f(x) = O(g(x))$.

The notation can also be used to describe the behavior of f near some real number a (often, $a = 0$): we say

$$f(x) = O(g(x)) \text{ as } x \to a$$

if and only if there exist positive numbers δ and M such that

$$|f(x)| \leq \ M|g(x)| \text{ for } |x - a| < \delta.$$

If $g(x)$ is non-zero for values of x sufficiently close to a, both of these definitions can be unified using the limit superior:

$$f(x) = O(g(x)) \text{ as } x \to a$$

if and only if

$$\limsup_{x \to a} \left| \frac{f(x)}{g(x)} \right| < \infty.$$

Rate of Convergence

As sequences are functions defined on $\mathbb{N}$, the domain of the natural numbers, we can apply big-O notation to sequences.

Therefore, this notation is useful to describe the rate at which a sequence of computations converges to a limit.

Definition 1.

Let $\{x_n\}$ and $\{y_n\}$ be sequences that satisfy

$$\lim_{n\to\infty} x_n = x, \lim_{n\to\infty} y_n = 0$$

where x is a real number. We say that $\{x_n\}$ converges to x with **rate of convergence** *$O(y_n)$ if there exists a positive real number M and a real number n_0 such that*

$$|x_n - x| \leq M|y_n| \text{ for all } n \geq n_0.$$

Which iterative method is better?

We say that an **iterative method converges rapidly**, in some sense, if it produces a sequence of approximate solutions whose rate of convergence is $O(y_n)$, where the terms of the sequence n approach zero rapidly as n approaches ∞.

Intuitively, if two iterative methods for solving the same problem perform a comparable amount of computation during each iteration, but one method exhibits a faster rate of convergence, then that method should be used because it will require less overall computational effort to obtain an approximate solution that is sufficiently accurate.

Example 2 (Rate of Convergence).

The sequences $\left\{\frac{n+1}{n+2}\right\}$ *and* $\left\{\frac{2n^2+4n}{n^2+2n+1}\right\}$ *converge to* 1 *and* 2 *with rates of convergence* $O(1/n)$ *and* $O(1/n^2)$ *respectively.*

Order of Convergence

A sequence of iterates $\{x_n : n \geq 0\}$ is said to **converge** with order $p \geq 1$ to a point α if

$$|\alpha - x_{n+1}| \leq c|\alpha - x_n|^p, \qquad n \geq 0 \tag{1}$$

for some $c > 0$, called **asymptotic error constant.**

If $p = 1$, the sequence is said to **converge linearly** to α. In that case, we require $c < 1$; the constant c is called the **rate of linear convergence** of x_n to α.

In general, a sequence with a high order of convergence converges more rapidly than a sequence with a lower order.

The asymptotic constant affects the speed of convergence but is not as important as the order.

Theorem 3.

Let $f(x)$ have $n+1$ continuous derivatives on $[a,b]$, and let $x_0, x = x_0 + h \in [a,b]$. Then

$$f(x_0 + h) = \sum_{k=0}^{n} \frac{f^{(k)}(x_0)}{k!} h^k + O(h^{n+1}).$$

Let us discuss now some methods to find root of an equation.

Graphical Solution of Equations

Let the equation be $f(x) = 0$.

1. Find the interval (a, b) in which a root of $f(x) = 0$ lies.
2. Write the equation $f(x) = 0$ as $g(x) = h(x)$, where $h(x)$ contains only terms in x and the constants.
3. Draw the graphs of $y = g(x)$ and $y = h(x)$ on the same scale and with respect to the same axes.
4. Read the abscissae of the points of intersection of the curves $y = g(x)$ and $y = h(x)$. These are the initial approximations to the roots of $f(x) = 0$.

Bisection Method (Bolzano Method)

The **bisection method** is based on the Intermediate Value Theorem.

Theorem 4 (Intermediate Value Theorem).

Suppose f is a continuous function defined on $[a, b]$, with $f(a)$ and $f(b)$ of having opposite signs. Then there exists a number α in (a, b) with $f(\alpha) = 0$.

Although the procedure will work when there is more than one root in the interval (a, b), we assume for simplicity that the root in this interval is unique. The method calls for a repeated halving of subintervals of $[a, b]$ and, at each step, locating the half containing α.

Algorithm

To begin, set $a_1 = a$ and $b_1 = b$, and let x_1 be the midpoint of $[a, b]$. That is,

$$x_1 = a_1 + \frac{b_1 - a_1}{2} = \frac{a_1 + b_1}{2} \quad \textbf{(first approximation)}.$$

If $f(x_1) = 0$, then $\alpha = x_1$, and we are done. If $f(x_1) \neq 0$, then $f(x_1)$ has the same sign as either $f(a_1)$ or $f(b_1)$.

1. When $f(a_1)$ and $f(x_1)$ have the same sign, $\alpha \in (x_1, b_1)$, and we set $a_2 = x_1$ and $b_2 = b_1$.
2. When $f(a_1)$ and $f(x_1)$ have opposite signs, $\alpha \in (a_1, x_1)$, and we set $a_2 = a_1$ and $b_2 = x_1$.

We then reapply the process to the interval $[a_2, b_2]$ to get **second approximation** p_2.

Stopping Procedures

We can select a tolerance $\varepsilon > 0$ and generate $p_1, p_2, \ldots, p_N$ until one of the following conditions is met.

$$\begin{aligned} |p_N - p_{N-1}| &< \varepsilon, \quad (2) \\ |f(p_N)| &< \varepsilon, \quad (3) \\ \frac{|p_N - p_{N-1}|}{|p_N|} &< \varepsilon, \quad p_N \neq 0. \quad (4) \end{aligned}$$

Difficulties can arise when the first and second stopping criteria are used. For example,

- Sequences $(p_n)_{n=1}^{\infty}$ with the property that the differences $p_n - p_{n-1}$ can converge to zero while the sequence itself diverges.
- It is also possible for $f(p_n)$ to be close to zero while p_n differs significantly from α.

How to apply bisection algorithm?

- An interval $[a, b]$ must be found with $f(a).f(b) < 0$.
- As at each step the length of the interval known to contain a zero of f is reduced by a factor of 2, it is advantageous to choose the initial interval $[a, b]$ as small as possible. For, example, if $f(x) = 2x^3 - x^2 + x - 1$, we have both

$$f(-4).f(4) < 0 \text{ and } f(0).f(1) < 0,$$

so the bisection algorithm could be used on either on the intervals $[-4, 4]$ or $[0, 1]$. However, starting the bisection algorithm on $[0, 1]$ instead of $[-4, 4]$ will reduce by 3 the number of iterations required to achieve a specified accuracy.

Best Stopping Criterion

Without additional knowledge about f and α, inequality

$$\frac{|p_N - p_{N-1}|}{|p_N|} < \varepsilon, \;\; p_N \neq 0,$$

is the best stopping criterion to apply because it comes closest to testing relative error.

For example, the iteration process is terminated when the relative error is less then 0.0001; that is, when

$$\frac{|\alpha - p_n|}{|\alpha|} < 10^{-4}.$$

How to find the relative error bound?

How to find $\frac{|\alpha - p_n|}{|\alpha|}$, relative bound? The following theorem answers the question.

Theorem 5.

Suppose that $f \in C[a,b]$ and $f(a).f(b) < 0$. The bisection method generates a sequence $\{p_n\}_{n=1}^{\infty}$ approximating a zero α of f with

$$|p_n - \alpha| \leq \frac{b-a}{2^n}, \text{ when } n \geq 1.$$

- The method has the important property that **it always converges to a solution**, and for that reason it is often used as a starter for the more efficient methods. But it is slow to converge.
- It is important to realize that the above theorem gives only a bound for approximation error and that this bound may be quite conservative.

Number of Iterations Needed?

To determine the number of iterations necessary to solve

$$f(x) = x^3 + 4x^2 - 10 = 0$$

with accuracy 10^{-3} using $a_1 = 1$ and $b_1 = 2$ requires finding an integer N that satisfies

$$|p_N - \alpha| \leq 2^{-N}(b-a) = 2^{-N} < 10^{-3}.$$

A simple calculation shows that ten iterations will ensure an approximation accurate to within 10^{-3}.

Again, it is important to keep in mind that the error analysis gives only a bound for the number of iterations, and in many cases this bound is much larger than the actual number required.

How to find an approximation which is correct to at least some 's' significant digits?

Find the iteration number n_0 such that $\frac{|b_{n_0}-a_{n_0}|}{|a_{n_0}|} \leq \frac{1}{10^{s+1}}$.

Since

$$\frac{|\alpha - p_{n_0-1}|}{|\alpha|} \leq \frac{|b_{n_0} - a_{n_0}|}{|a_{n_0}|} \leq \frac{1}{10^{s+1}},$$

p_{n_0-1} is an approximation which is correct to at least some 's' significant digits.

As you have calculated p_{n_0}, so you can take p_{n_0}, which is a (good) approximation which is correct to at least some 's' significant digits.

Order of Convergence

Let p_n denote the nth approximation of α in the algorithm. Then it is easy to see that

$$\alpha = \lim_{n\to\infty} p_n$$

$$|\alpha - p_n| \leq \left[\frac{1}{2}\right]^n (b-a) \tag{5}$$

where $b-a$ denotes the length of the original interval. From the inequality $|\alpha - p_n| \leq \frac{1}{2}|\alpha - p_{n-1}| \quad n \geq 0$, we say that **the bisection method converges linearly with a rate of $\frac{1}{2}$.**

The actual error may not decrease by a factor of $\frac{1}{2}$ at each step, but the average rate of decrease is $\frac{1}{2}$.

Since e_{n+1}/e_n is almost $\frac{1}{2}$, the convergence in the bisection method is **linear**.

Exercises

1. Find a root of the equation $x^3 - 4x - 9 = 0$, using bisection method correct to 3 decimal places.
2. Using bisection method, find the negative root of the equation $x^2 + \cos x - 2 = 0$.
3. Using bisection method, find an approximate root of the equation $\sin x = 1/x$, that lies between $x = 1$ and $x = 1.5$ (measured in radians). Carry out computations upto 7th stage.
4. Find the root of the equation $\cos x = x\ e^x$ using bisection method correct to 4 decimal places.
5. Find a positive real root of $x \log_{10} x = 1.2$ using bisection method.

Fixed Point Iterative Method

The point p is a fixed point of the function g if $g(p) = p$.

We consider a problem of finding "fixed points of a function", called **fixed point problem.**

Example 6.

The function $f(x) = x^2$ *has fixed points* 0 *and* 1. *Whereas the function* $g(x) = x + 2$ *has no fixed point.*

Root-finding problems and fixed-point problems are equivalent classes in the following sense.

Theorem 7.

f *has a root at* α *iff* $g(x) = x - f(x)$ *has a fixed point at* α.

Several g may exist

There is more than one way to convert a function that has a root at α into a function that has a fixed point at α.

Example 8.

The function $f(x) = x^3 + 4x^2 - 10$ *has a root somewhere in the interval* $[1, 2]$. *Here are several functions that have a fixed point at that root.*

$$\begin{aligned}
g_1(x) &= x - f(x) = x - x^3 - 4x^2 + 10 && (6)\\
g_2(x) &= \sqrt{\frac{10}{x} - 4x} && (7)\\
g_3(x) &= \frac{1}{2}\sqrt{10 - x^3} && (8)\\
g_4(x) &= \sqrt{\frac{10}{4 - x}} && (9)\\
g_5(x) &= x - \frac{x^3 + 4x^2 - 10}{3x^2 + 8x} && (10)
\end{aligned}$$

Sufficient Conditions

Theorem 9 (Existence of a Fixed Point).

If $g \in C[a, b]$ and $g(x) \in [a, b]$ for all $x \in [a, b]$, then g has a fixed point.

Theorem 10 (Uniqueness of a Fixed Point).

If g has a fixed point and if $g'(x)$ exists on (a, b) and a positive constant $k < 1$ exists with

$$|g'(x)| \leq k \text{ for all } x \in (a, b),$$

then the fixed point in $[a, b]$ is unique.

The condition in the above theorem, is not necessary.

Example 11.

The function $g(x) = 3^{-x}$ on $[0, 1]$ has a unique fixed point. But $|g'(x)| \nleq 1$ on $(0, 1)$.

If sufficient conditions are satisfied, then how to find the fixed point?

To approximate the fixed point of a function g, we choose an initial approximation x_0 and generate the sequence $(x_n)_{n=0}^{\infty}$ by letting $x_n = g(x_{n-1})$, for each $n \geq 1$.

If the sequence converges to α and g is continuous, then

$$\alpha = \lim_{n\to\infty} x_n = \lim_{n\to\infty} g(x_{n-1}) = g(\lim_{n\to\infty} x_{n-1}) = g(\alpha),$$

and a solution to $x = g(x)$ is obtained.

This technique is called **fixed-point iteration, or functional iteration.**

Fixed-Point Theorem

Theorem 12.

Let $g \in C[a, b]$ be such that $g(x) \in [a, b]$ for all $x \in [a, b]$. Suppose, in addition, that g' exists on (a, b) and a positive constant $k < 1$ exists with

$$|g'(x)| \leq k \text{ for all } x \in (a, b).$$

Then, for any number x_0 in $[a, b]$, the sequence defined by

$$x_n = g(x_{n-1}), n \geq 1,$$

converges to the unique fixed point α in $[a, b]$.

Which g is better?

Using the Mean Value Theorem and the fact that $|g'(x)| \leq k$, we have, for each n,

$$|x_n - \alpha| \leq k|x_{n-1} - \alpha|.$$

Applying the above inequality inductively gives

$$|x_n - \alpha| \leq k^n|x_0 - \alpha|.$$

Since $0 < k < 1$, $(x_n)_{n=1}^{\infty}$ converges to α.

The rate of convergence depends on the factor k^n. The smaller the value of k, the faster the convergence, which may be very slow if k is close to 1.

Finally, we have got some clue (!) for g, which should be rejected.

When to stop the procedure if error bound is given?

If we are satisfied with an approximate solution which is in ε-neighbourhood of the exact value α (ε-distance away from the exact value α), then the following inequalities are helpful.

For all $n \geq 1$,

$$|x_n - \alpha| \leq k^n \max\{x_0 - a, b - x_0\} < \varepsilon$$

and

$$|x_n - \alpha| \leq \frac{k}{1-k}|x_n - x_{n-1}| < \varepsilon.$$

Find the difference between two consecutive approximations, $|x_n - x_{n-1}|$. If

$$|x_n - x_{n-1}| < \frac{1-k}{k}\varepsilon,$$

then we can say that x_n is ε-distance away from the exact value α.

Order of Convergence of Fixed Point Iterative Method

Consider the equation $f(x) = 0$. Suppose the equation can be expressed in the form $x = g(x)$.

Let $x_{n+1} = g(x_n)$ define an iteration method for solving the equation $f(x) = 0$.

Let α be the root of the above equation. Let $x_n = \alpha + \varepsilon_n$.

Then $\varepsilon_n = x_n - \alpha$ is the **error** in x_n.

If $g(x)$ is differentiable any number of times then the Taylor's formula for $g(x)$ is given by

$$g(x) = g(\alpha) + \frac{g'(\alpha)}{1!}(x-\alpha) + \frac{g''(\alpha)}{2!}(x-\alpha)^2 + \cdots .$$

Order of Convergence of Fixed Point Iterative Method

Hence

$$x_{n+1} = g(\alpha) + \frac{g'(\alpha)}{1!}\varepsilon_n + \frac{g''(\alpha)}{2!}\varepsilon_n^2 + \cdots .$$

The power of ε_n in the first non-vanishing term after $g(\alpha)$ is called the **order of the iteration** process.

The order is a measure for the speed of convergence.

For example in the case of **first order convergence**,

$$\varepsilon_{n+1} = g'(\alpha)\varepsilon_n.$$

Similarly, in the case of **second order convergence**,

$$\varepsilon_{n+1} = \frac{1}{2}g''(\alpha)\varepsilon_n^2.$$

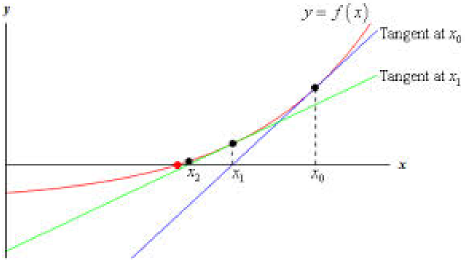
Exercises

6. Find a real root of the equation $\cos x = 3x - 1$ correct to 3 decimal places using iterative method.
7. Using iterative method, find a root of the equation $x^3 + x^2 - 1 = 0$ correct to 4 decimal places.
8. Apply iterative method to find the negative root of the equation $x^3 - 2x + 5 = 0$ correct to 4 decimal places.
9. Find a real root of $2x - \log_{10} x = 7$ correct to 4 decimal places using iterative method.
10. Find the smallest root of the equation

$$1 - x + \frac{x^2}{(2!)^2} - \frac{x^3}{(3!)^2} + \frac{x^4}{(4!)^2} - \frac{x^5}{(5!)^2} + \cdots = 0.$$

Newton-Raphson Method

Newton's (or the **Newton-Raphson**) **method** is one of the most powerful and well-known numerical methods for solving a root-finding problem. There are many ways of introducing Newton's method. First we explain the algorithm graphically. The figure illustrates how the approximations are obtained using successive tangents. The approximation x_2 is the x-intercept of the tangent line to the graph of f at $(x_1, f(x_1))$ and so on.



Starting with the initial approximation x_0, the approximation x_1 is the x-intercept of the tangent line to the graph of f at $(x_0, f(x_0))$.

Because of this geometrical interpretation Newton-Raphson method is also referred as the **method of tangents**.

Newton's Method by Taylor Polynomials

We introduce Newton's method by using Taylor polynomials.

Suppose $f \in C^2[a, b]$. Let $x_0 \in [a, b]$ be an approximation to the solution α of $f(x) = 0$ such that $f'(x_0) \neq 0$ and $|x - x_0|$ is "small".

Consider the first Taylor polynomial for $f(x)$ expanded about x_0, and evaluated at $x = \alpha$,

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

where $\xi(\alpha)$ lies between α and x_0.

Since $f(\alpha) = 0$, this equation gives

$$0 = f(x_0) + (\alpha - x_0)f'(x_0) + \frac{(\alpha - x_0)^2}{2} f''(\xi(\alpha)).$$

Newton's method is derived by assuming that since $|\alpha - x_0|$ is small, the term involving $(\alpha - x_0)^2$ is much smaller, so

$$0 \approx f(x_0) + (\alpha - x_0) f'(x_0).$$

Solving for x_0 gives

$$\alpha \approx x_1 \equiv x_0 - \frac{f(x_0)}{f'(x_0)}.$$

This sets the stage for Newton's method, which starts with an initial approximation x_0 and generates the sequence $(x_n)_{n=0}^{\infty}$ by

$$x_n = x_{n-1} - \frac{f(x_{n-1})}{f'(x_{n-1})}, \text{ for } n \geq 1.$$

Newton's method is a functional iteration technique of the form $x_n = g(x_{n-1})$, for which

$$g(x_{n-1}) = x_{n-1} - \frac{f(x_{n-1})}{f'(x_{n-1})}, \text{ for } n \geq 1.$$

- Newton's method cannot be continued if $f'(x_{n-1}) = 0$ for some n.
- The method is most effective when f' is bounded away from zero near α.
- **Importance of an accurate initial approximation.** The crucial assumption is that the term involving $(\alpha - x_0)^2$ is, by comparison with $|\alpha - x_0|$, so small that it can be deleted. **This will clearly be false** unless x_0 is a good approximation to α.
- **If x_0 is not sufficiently close to the actual root, there is little reason to suspect that Newton's method will converge to the root.**

Convergence of Newton's Method

The Newton-Raphson formula is $x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} = g(x_n)$.

The general form of above equation is $x = g(x)$. We know that the iteration method given by $x = g(x)$ converges if $|g'(x)| < 1$.

Note that $g(x) = x - \frac{f(x)}{f'(x)}$ and $g'(x) = \frac{f(x)f''(x)}{[f'(x)]^2}$.

Hence Newton's formula converges if

$$|f(x).f''(x)| < \{f'(x)\}^2$$

in the interval considered. Since $f(x), f'(x), f''(x)$ are all continuous, we can select a small interval in the neighbourhood of α, the actual root, in which the above condition is satisfied. Hence Newton's formula always converges, provided that the initial approximation x_0 is taken sufficiently close to α.

Theoretical Importance of the Choice of x_0

Theorem 13 (Convergence Theorem for Newton's Method).

Let $f \in C^2[a,b]$. If $\alpha \in [a,b]$ is such that $f(\alpha) = 0$ and $f'(\alpha) \neq 0$, then there exists a $\delta > 0$ such that Newton's method generates a sequence $(x_n)_{n=1}^{\infty}$ converges to α for any initial approximation $x_0 \in [\alpha - \delta, \alpha + \delta]$.

This result is important for the theory of Newton's method, but it is seldom applied in practice since it does not tell us how to determine the value of δ.

In a practical application, an initial approximation is selected, and successive approximations are generated by Newton's method. These will generally either converge quickly to the root, or it will be clear that convergence is unlikely.

Convergence Theorem for Newton's Method states that, under reasonable assumptions, Newton's method converges provided **a sufficiently accurate initial approximation is chosen.**

It also implies that the constant k that bounds the derivative of g, and, consequently, indicates the speed of convergence of the method, decreases to 0 as the procedure continues.

Major weakness is to know the value of the derivative of f at each approximation. Frequently, $f'(x)$ is far more difficult and needs more arithmetic operations to calculate than $f(x)$.

Rate of Convergence of Newton-Raphson Method

The Newton-Raphson iteration formula is given by

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} = g(x_n).$$

On comparing this with the relation $x_{n+1} = g(x_n)$, we get $g(x) = x - \frac{f(x)}{f'(x)}$.

Hence

$$g'(x) = 1 - \frac{[f'(x)]^2 - f(x)f''(x)}{[f'(x)]^2} = \frac{f(x)f''(x)}{[f'(x)]^2}.$$

If α is the desired root of the equation $f(x) = 0$, we have

$$g'(\alpha) = \frac{f(\alpha)f''(\alpha)}{[f'(\alpha)]^2} = 0.$$

Hence

$$\begin{aligned} x_{n+1} &= g(\alpha) + \frac{g'(\alpha)}{1!}\varepsilon_n + \frac{g''(\alpha)}{2!}\varepsilon_n^2 + \cdots \\ &= \alpha + \frac{g''(\alpha)}{2!}\varepsilon_n^2 + \cdots \qquad [\text{since } g(\alpha) = \alpha,\ g'(\alpha) = 0]. \end{aligned}$$

Hence

$$\varepsilon_{n+1} = \frac{g''(\alpha)}{2!}\varepsilon_n^2 + \cdots .$$

That is, the error at any state (i.e., the difference between the approximation and the actual value of the root) is proportional to the **square** of the error in the previous stage.

Thus Newton-Raphson method has a **quadratic convergence** or its order of convergence is 2.

Some Deductions from Newton-Raphson Method

We can derive the following useful results from the Newton's iterative formula.

Formula to find $1/N$	$x_{n+1} = x_n(2 - Nx_n)$
Formula to find $\sqrt{N}$	$x_{n+1} = \frac{1}{2}(x_n + \frac{N}{x_n})$
Formula to find $\frac{1}{\sqrt{N}}$	$x_{n+1} = \frac{1}{2}(x_n + \frac{1}{Nx_n})$
Formula to find $\sqrt[k]{N}$	$x_{n+1} = \frac{1}{k}\left[(k-1)x_n + \frac{N}{x_n^{k-1}}\right]$

11. Evaluate the following (correct to 4 decimal places) by Newton's iterative method.

(a) $1/31$
(b) $\sqrt{5}$
(c) $1/\sqrt{14}$
(d) $\sqrt[3]{24}$
(e) $30^{-\frac{1}{5}}$
(f) $22/7$.

Generalized Newton's Method

If α is a root of $f(x) = 0$ with multiplicity p, then the iteration formula corresponding to

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

is taken as

$$x_{n+1} = x_n - p\frac{f(x_n)}{f'(x_n)} \tag{11}$$

which means that $(1/p)f'(x_n)$ is the slope of the straight line passing through (x_n, y_n) and intersecting the x-axis at the point $(x_{n+1}, 0)$.

Equation (11) is called the **generalized Newton's formula** and reduces to Newton-Raphson's formula if $p = 1$.

Since α is a root of $f(x) = 0$ with multiplicity p, it follows that α is also a root of $f'(x) = 0$ with multiplicity $(p-1)$, and so on. Hence the expressions

$$x_0 - p\frac{f(x_0)}{f'(x_0)}, \quad x_0 - (p-1)\frac{f'(x_0)}{f''(x_0)}, \quad x_0 - (p-2)\frac{f''(x_0)}{f'''(x_0)}$$

must have the same value if there is a root with multiplicity p, provided that the initial approximation x_0 is chosen sufficiently close to the root.

Generalized Newton's formula has a **second order convergence** for determining a multiple root.

Exercises

12. Find a double root of the equation $f(x) = x^3 - x^2 - x + 1 = 0$.
13. Show that the generalized (modified) Newton-Raphson method $x_{n+1} = x_n - \frac{2f(x_n)}{f'(x_n)}$ gives a quadratic convergence when the equation $f(x) = 0$ has a pair of double roots in the neighbourhood of $x = x_n$.

Secant Method : Derivation by using Taylor Polynomial

To avoid the derivation in Newton's method, we introduct a slight variation.

By definition,

$$f'(x_{n-1}) = \lim_{x \to x_{n-1}} \frac{f(x) - f(x_{n-1})}{x - x_{n-1}}.$$

Letting $x = x_{n-2}$, we have

$$f'(x_{n-1}) \approx \frac{f(x_{n-2}) - f(x_{n-1})}{x_{n-2} - x_{n-1}} = \frac{f(x_{n-1}) - f(x_{n-2})}{x_{n-1} - x_{n-2}}.$$

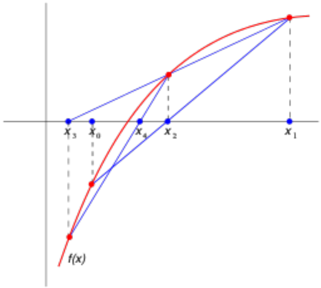
Using this approximation for $f'(x_{n-1})$ in Newton's formula gives

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

Derivation of Secant Method Graphically

Starting with the two initial approximations x_0 and x_1, the approximation x_2 is the x-intercept of the line joining $(x_0, f(x_0))$ and $(x_1, f(x_1))$.

The approximation x_3 is the line joining $(x_1, f(x_1))$ and $(x_2, f(x_2))$, and so on. **The convergence is superlinear and the order of convergence is 1.618.**



Root Bracketing

Each successive pair of approximations in the bisection method brackets a root α of the equation; that is, for each positive integer n, a root lies between a_n and b_n.

This implies that, for each n, the bisection method iterations satisfy

$$|x_n - \alpha| < \frac{1}{2}|a_n - b_n|,$$

which provides an easily calculated error bound for the approximation.

Root bracketing is not guaranteed for either Newton's method or the secant method.

The initial approximations x_0 and x_1 bracket the root, but the pair of approximations x_3 and x_4 fail to do so.

Exercises

13. Find the positive root of $x^4 - 4 = 0$ correct to 3 decimal places using Newton-Raphson method.
14. Find by Newton's method, the real root of the equation $3x = \cos x + 1$, correct to 4 decimal places.
15. Using Newton's iterative method, find the real root of $x \log_{10} x = 1.2$ correct to 5 decimal places.
16. Find a root of the equation $x^3 - 2x - 5 = 0$ using secant method correct to 3 decimal places.
17. Find the root of the equation $x\ e^x = \cos x$ using the secant method correct to 4 decimal places.

Method of False Position (also called Regula Falsi)

The method of false position generates approximations in the same manner as the secant method, but it includes a test to ensure that the root is always bracketed between successive iterations. It illustrates how bracketing can be incorporated.

First choose initial approximations x_0 and x_1 with $f(x_0).f(x_1) < 0$. Let $a_0 = x_0$ and $b_0 = x_1$. Note that a_0 and b_0 bracket a root.

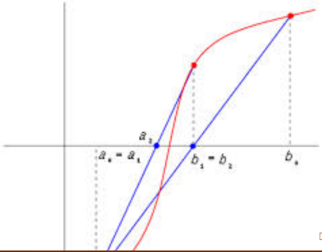
The approximation x_2 is chosen in the same manner as in the secant method, as the x-intercept of the line joining $(a_0, f(a_0))$ and $(b_0, f(b_0))$.

Method of False Position (also called Regula Falsi)

To decide which secant line to use to compute x_3, we check $f(a_0).f(x_3)$. If this value is negative, then a_0 and x_3 bracket a root. (If not, x_3 and b_0 bracket a root.)

A relabeling of a_0 and x_3 is performed as $a_1 = a_0$ and $b_1 = x_3$. (If not, a relabeling of x_3 and b_0 is performed as $a_1 = x_3$ and $b_1 = b_0$). The relabeling ensures that the root is bracketed between successive iterations.

The order of convergence of Regula-Falsi method is 1.618.



More Calculations Needed for the "Added Insurance" of the Method of False Position

The added insurance of the method of False Position commonly requires more calculation than the Secant method, just as the simplication that the Secant method provides over Newton's method usually comes at the expense of additional iterations.

Exercises

18. Find a real root of the equation $x^3 - 2x - 5 = 0$ by the method of false position correct to 3 decimal places.
19. Find the root of the equation $\cos x = x\ e^x$ using the regula-falsi method correct to 4 decimal places.
20. Find a real root of the equation $x \log_{10} x = 1.2$ by regula-falsi method correct to 4 decimal places.
21. Use the method of false position, to find the fourth root of 32 correct to 3 decimal places.

Newton-Raphson's Method for Simultaneous Equations

The real solutions of simultaneous algebraic and transcendental equations in several unknowns can be found by Newton-Raphson method.

We restrict ourselves to the case of **two unknowns** and explain Newton-Raphson method for solving them.

Consider the equations

$$f(x, y) = 0 \qquad g(x, y) = 0. \tag{12}$$

Let (x_0, y_0) be an initial approximate solution of (12).

Let $x_1 = x_0 + h$ and $y_1 = y_0 + k$ be the next approximation.

Expanding f and g by Taylor's theorem for a function of two variables around the point (x_1, y_1), we have

$$f(x_1, y_1) = f(x_0 + h, y_0 + k) = f(x_0, y_0) + h\left(\frac{\partial f}{\partial x}\right)_{(x_0,y_0)} + k\left(\frac{\partial f}{\partial y}\right)_{(x_0,y_0)}$$

and

$$g(x_1, y_1) = g(x_0 + h, y_0 + k) = g(x_0, y_0) + h\left(\frac{\partial g}{\partial x}\right)_{(x_0,y_0)} + k\left(\frac{\partial g}{\partial y}\right)_{(x_0,y_0)}$$

(omitting higher powers of h and k).

If (x_1, y_1) is a solution of (12), then $f(x_1, y_1) = 0$ and $g(x_1, y_1) = 0$.

Hence

$$f(x_0, y_0) + hf_x(x_0, y_0) + kf_y(x_0, y_0) = 0 \tag{13}$$

and

$$g(x_0, y_0) + hg_x(x_0, y_0) + kg_y(x_0, y_0) = 0. \tag{14}$$

If the Jacobian $J = \begin{vmatrix} f_x & f_y \\ g_x & g_y \end{vmatrix} \neq 0$, then the equations (13) and (14) provide a **unique solution** of h and k.

Now $x_1 = x_0 + h$ and $y_1 = y_0 + k$ give a new approximation to the solution.

By repeating this process we obtain the required solution to the desired accuracy. If the iteration converges, it does so **quadratically**.

We find the **initial approximation** by trail and error or by graphical method so that we can improve the accuracy and the convergence is guaranteed.

The above method can be extended to **simultaneous equations in any number of unknowns**.

Sufficient Condition for Convergence

The following theorem gives the conditions which are sufficient for convergence.

Theorem 14.

Let (x_0, y_0) be an approximation to a root (α, β) of the system

$$f(x,y) = 0 \quad \text{and} \quad g(x,y) = 0 \tag{15}$$

in the closed neighbourhood D containing (α, β). If f, g and all their first and second derivatives are continuous and bounded in D, and, the Jacobian $J(f,g) \neq 0$ in D, then the sequence of approximations given by

$$x_{n+1} = x_n - \frac{1}{J(f,g)} \begin{vmatrix} f & g \\ \frac{\partial f}{\partial y} & \frac{\partial g}{\partial y} \end{vmatrix} \quad \text{and} \quad y_{n+1} = y_n - \frac{1}{J(f,g)} \begin{vmatrix} f & g \\ \frac{\partial g}{\partial x} & \frac{\partial f}{\partial x} \end{vmatrix}$$

converges to the root (α, β) of the system (15).

Exercises

22. Solve the equations $x = x^2 + y^2,\ y = x^2 - y^2$ using Newton-Raphson method with the approximation $(0.8, 0.4)$.
23. Find a root of the system of nonlinear equations by Newton-Raphson method, $x^2 + y = 11,\ y^2 + x = 7$ with $x_0 = 3.5$ and $y_0 = -1.8$.
24. Solve the system of equations

$$\begin{aligned} \sin xy + x - y &= 0 \\ y \cos xy + 1 &= 0 \end{aligned}$$

with $x_0 = 1$ and $y = 2$, by Newton-Raphson's method.

Muller's Method

Muller's method is a root-finding algorithm, a numerical method for solving equations of the form

$$f(x) = 0.$$

It is first presented by David E. Muller in 1956.

David E. Muller

Muller's Method

Muller's method is a generalization of the secant method.

Instead of starting with two initial values and then joining them with a straight line in secant method, Mullers method starts with **three initial approximations** to the root and then join them with a second degree polynomial (a parabola).

Then the quadratic formula is used to find a root of the quadratic for the next approximation.

That is if x_0, x_1 and x_2 are the initial approximations then x_3 is obtained by solving the quadratic which is obtained by means of x_0, x_1 and x_2.

Then two values among x_0, x_1 and x_2 which are close to x_3 are chosen for the next iteration.

Muller's Method

Muller's method is a recursive method which generates an approximation of the root ξ of f at each iteration.

Starting with the three initial values x_0, x_{-1} and x_{-2}, the first iteration calculates the first approximation x_1, the second iteration calculates the second approximation x_2, the third iteration calculates the third approximation x_3, etc.

Hence the kth iteration generates approximation x_k.

Each iteration takes as input the last three generated approximations and the value of f at these approximations.

Hence the kth iteration takes as input the values x_{k-1}, x_{k-2} and x_{k-3} and the function values $f(x_{k-1})$, $f(x_{k-2})$ and $f(x_{k-3})$.

Muller's Method

The approximation x_k is calculated as follows.

A parabola $y_k(x)$ is constructed which goes through the three points $(x_{k-1}, f(x_{k-1})), (x_{k-2}, f(x_{k-2}))$ and $(x_{k-3}, f(x_{k-3}))$.

When written in the Newton form, $y_k(x)$ is

$$y_k(x) = f(x_{k-1}) + (x - x_{k-1}) f[x_{k-1}, x_{k-2}] + (x - x_{k-1})(x - x_{k-2}) f[x_{k-1}, x_{k-2}, x_{k-3}]$$

where $f[x_{k-1}, x_{k-2}]$ and $f[x_{k-1}, x_{k-2}, x_{k-3}]$ denote divided differences.

Muller's Method

This can be rewritten as

$$y_k(x) = f(x_{k-1}) + w(x - x_{k-1}) + f[x_{k-1}, x_{k-2}, x_{k-3}](x - x_{k-1})^2$$

where

$$w = f[x_{k-1}, x_{k-2}] + f[x_{k-1}, x_{k-3}] - f[x_{k-2}, x_{k-3}].$$

The next iterate x_k is now given as the solution closest to x_{k-1} of the quadratic equation $y_k(x) = 0$.

This yields the recurrence relation

$$x_k = x_{k-1} - \frac{2f(x_{k-1})}{w \pm \sqrt{w^2 - 4f(x_{k-1})f[x_{k-1}, x_{k-2}, x_{k-3}]}}.$$

Muller's Method

In this formula, the sign should be chosen such that the denominator is as large as possible in magnitude.

We do not use the standard formula for solving quadratic equations because that may lead to loss of significance.

Note that x_k can be complex, even if the previous iterates were all real. This is in contrast with other root-finding algorithms like the secant method, Sidi's generalized secant method or Newton's method, whose iterates will remain real if one starts with real numbers.

Having complex iterates can be an advantage (if one is looking for complex roots) or a disadvantage (if it is known that all roots are real), depending on the problem.

Speed of convergence

The order of convergence of Muller's method is approximately 1.84. This can be compared with 1.62 for the secant method and 2 for Newton's method. So, the secant method makes less progress per iteration than Muller's method and Newton's method makes more progress.

More precisely, if ξ denotes a single root of f (so $f(\xi) = 0$ and $f'(\xi) \neq 0$), f is three times continuously differentiable, and the initial guesses x_0, x_1, and x_2 are taken sufficiently close to ξ, then the iterates satisfy

$$\lim_{k \to \infty} \frac{|x_k - \xi|}{|x_{k-1} - \xi|^{\mu}} = \left| \frac{f'''(\xi)}{6f'(\xi)} \right|^{(\mu-1)/2}$$

where $\mu \approx 1.84$ is the positive solution of $x^3 - x^2 - x - 1 = 0$.

References

1. **Richard L. Burden and J. Douglas Faires**, *Numerical Analysis - Theory and Applications*, Cengage Learning, Singapore.
2. **Kendall E. Atkinson**, *An Introduction to Numerical Analysis*, Wiley India.
3. **David Kincaid and Ward Cheney**, *Numerical Analysis - Mathematics of Scientific Computing*, American Mathematical Society, Providence, Rhode Island.
4. **S.S. Sastry**, *Introductory Methods of Numerical Analysis*, Fourth Edition, Prentice-Hall, India.