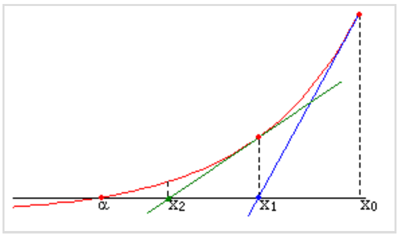
WIKIPEDIA
The Free Encyclopedia
WIKIPEDIA

Newton's method

In numerical analysis, the **Newton–Raphson method**, also known simply as **Newton's method**, named after Isaac Newton and Joseph Raphson, is a root-finding algorithm which produces successively better approximations to the roots (or zeroes) of a real-valued function. The most basic version starts with a real-valued function f, its derivative f', and an initial guess x_0 for a root of f. If f satisfies certain assumptions and the initial guess is close, then



An illustration of Newton's method.

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

is a better approximation of the root than x_0. Geometrically, $(x_1, 0)$ is the x-intercept of the tangent of the graph of f at $(x_0, f(x_0))$: that is, the improved guess, x_1, is the unique root of the linear approximation of f at the initial guess, x_0. The process is repeated as

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

until a sufficiently precise value is reached. The number of correct digits roughly doubles with each step. This algorithm is first in the class of Householder's methods, and was succeeded by Halley's method. The method can also be extended to complex functions and to systems of equations.

Description

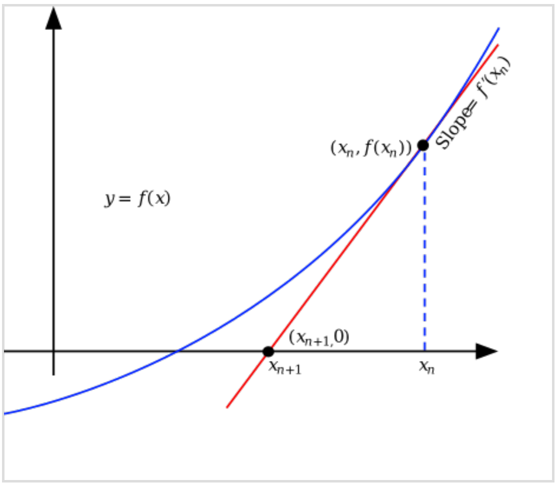
The idea is to start with an initial guess, then to approximate the function by its tangent line, and finally to compute the x-intercept of this tangent line. This x-intercept will typically be a better approximation to the original function's root than the first guess, and the method can be iterated.

If the tangent line to the curve $f(x)$ at $x = x_n$ intercepts the x-axis at x_{n+1} then the slope is

$$f'(x_n) = \frac{f(x_n) - 0}{x_n - x_{n+1}}.$$

Solving for x_{n+1} gives

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

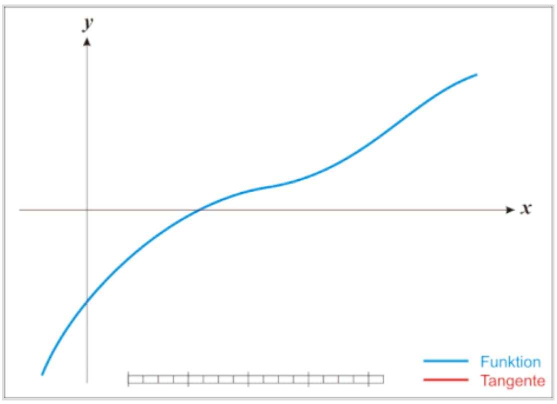


x_{n+1} is a better approximation than x_n for the root x of the function f (blue curve)

We start the process with some arbitrary initial value x_0. (The closer to the zero, the better. But, in the absence of any intuition about where the zero might lie, a "guess and check" method might narrow the possibilities to a reasonably small interval by appealing to the intermediate value theorem.)

The method will usually converge, provided this initial guess is close enough to the unknown zero, and that $f'(x_0) \neq 0$. Furthermore, for a zero of multiplicity 1, the convergence is at least quadratic (see *Rate of convergence*) in a neighbourhood of the zero, which intuitively means that the number of correct digits roughly doubles in every step. More details can be found in *§ Analysis* below.

Householder's methods are similar but have higher order for even faster convergence. However, the extra computations required for each step can slow down the overall performance relative to Newton's method, particularly if f or its derivatives are computationally expensive to evaluate.



Iteration typically improves the approximation

History

The name "Newton's method" is derived from Isaac Newton's description of a special case of the method in *De analysi per aequationes numero terminorum infinitas* (written in 1669, published in 1711 by William Jones) and in *De metodis fluxionum et serierum infinitarum* (written in 1671, translated and published as *Method of Fluxions* in 1736 by John Colson). However, his method differs substantially from the modern method given above. Newton applied the method only to polynomials, starting with an initial root estimate and extracting a sequence of error corrections. He used each correction to rewrite the polynomial in terms of the remaining error, and then solved for a new correction by neglecting higher-degree terms. He did not explicitly connect the method with derivatives or present a general formula. Newton applied this method to both numerical and algebraic problems, producing Taylor series in the latter case.

Newton may have derived his method from a similar, less precise method by Vieta. The essence of Vieta's method can be found in the work of the Persian mathematician Sharaf al-Din al-Tusi, while his successor Jamshīd al-Kāshī used a form of Newton's method to solve $x^P - N = 0$ to find roots of N (Ypma 1995). A special case of Newton's method for calculating square roots was known since ancient times and is often called the Babylonian method.

Newton's method was used by 17th-century Japanese mathematician Seki Kōwa to solve single-variable equations, though the connection with calculus was missing.[1]

Newton's method was first published in 1685 in *A Treatise of Algebra both Historical and Practical* by John Wallis.[2] In 1690, Joseph Raphson published a simplified description in *Analysis aequationum universalis*.[3] Raphson also applied the method only to polynomials, but he avoided Newton's tedious rewriting process by extracting each successive correction from the original polynomial. This allowed him to derive a reusable iterative expression for each problem. Finally, in 1740, Thomas Simpson described Newton's method as an iterative method for solving general nonlinear equations using calculus, essentially giving the description above. In the same publication, Simpson also gives the generalization to systems of two equations and notes that Newton's method can be used for solving optimization problems by setting the gradient to zero.

Arthur Cayley in 1879 in *The Newton–Fourier imaginary problem* was the first to notice the difficulties in generalizing Newton's method to complex roots of polynomials with degree greater than 2 and complex initial values. This opened the way to the study of the theory of iterations of rational functions.

Practical considerations

Newton's method is a powerful technique—in general the convergence is quadratic: as the method converges on the root, the difference between the root and the approximation is squared (the number of accurate digits roughly doubles) at each step. However, there are some difficulties with the method.

Difficulty in calculating the derivative of a function

Newton's method requires that the derivative can be calculated directly. An analytical expression for the derivative may not be easily obtainable or could be expensive to evaluate. In these situations, it may be appropriate to approximate the derivative by using the slope of a line through two nearby points on the function. Using this approximation would result in something like the secant method whose convergence is slower than that of Newton's method.

Failure of the method to converge to the root

It is important to review the proof of quadratic convergence of Newton's method before implementing it. Specifically, one should review the assumptions made in the proof. For situations where the method fails to converge, it is because the assumptions made in this proof are not met.

For example, in some cases, if the first derivative is not well behaved in the neighborhood of a particular root, then it is possible that Newton's method will fail to converge no matter where the initialization is set. In some cases, Newton's method can be stabilized by using successive over-relaxation, or the speed of convergence can be increased by using the same method.

In a robust implementation of Newton's method, it is common to place limits on the number of iterations, bound the solution to an interval known to contain the root, and combine the method with a more robust root finding method.

Slow convergence for roots of multiplicity greater than 1

If the root being sought has multiplicity greater than one, the convergence rate is merely linear (errors reduced by a constant factor at each step) unless special steps are taken. When there are two or more roots that are close together then it may take many iterations before the iterates get close enough to one of them for the quadratic convergence to be apparent. However, if the multiplicity m of the root is known, the following modified algorithm preserves the quadratic convergence rate:[4]

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

This is equivalent to using successive over-relaxation. On the other hand, if the multiplicity m of the root is not known, it is possible to estimate m after carrying out one or two iterations, and then use that value to increase the rate of convergence.

If the multiplicity m of the root is finite then $g(x) = \frac{f(x)}{f'(x)}$ will have a root at the same location with multiplicity 1. Applying Newton's method to find the root of $g(x)$ recovers quadratic convergence in many cases although it generally involves the second derivative of $f(x)$. In a particularly simple case, if $f(x) = x^m$ then $g(x) = \frac{x}{m}$ and Newton's method finds the root in a single iteration with

$$x_{n+1} = x_n - \frac{g(x_n)}{g'(x_n)} = x_n - \frac{\frac{x_n}{m}}{\frac{1}{m}} = 0\,.$$

Analysis

Suppose that the function f has a zero at α, i.e., $f(\alpha) = 0$, and f is differentiable in a neighborhood of α.

If f is continuously differentiable and its derivative is nonzero at α, then there exists a neighborhood of α such that for all starting values x_0 in that neighborhood, the sequence (x_n) will converge to α.[5]

If f is continuously differentiable, its derivative is nonzero at α, *and* it has a second derivative at α, then the convergence is quadratic or faster. If the second derivative is not 0 at α then the convergence is merely quadratic. If the third derivative exists and is bounded in a neighborhood of α, then:

$$\Delta x_{i+1} = \frac{f''(\alpha)}{2f'(\alpha)} (\Delta x_i)^2 + O(\Delta x_i)^3 ,$$

where

$$\Delta x_i \triangleq x_i - \alpha .$$

If the derivative is 0 at α, then the convergence is usually only linear. Specifically, if f is twice continuously differentiable, $f'(\alpha) = 0$ and $f''(\alpha) \neq 0$, then there exists a neighborhood of α such that, for all starting values x_0 in that neighborhood, the sequence of iterates converges linearly, with rate $\frac{1}{2}$.[6] Alternatively, if $f'(\alpha) = 0$ and $f'(x) \neq 0$ for $x \neq \alpha$, x in a neighborhood U of α, α being a zero of multiplicity r, and if $f \in C^r(U)$, then there exists a neighborhood of α such that, for all starting values x_0 in that neighborhood, the sequence of iterates converges linearly.

However, even linear convergence is not guaranteed in pathological situations.

In practice, these results are local, and the neighborhood of convergence is not known in advance. But there are also some results on global convergence: for instance, given a right neighborhood U_+ of α, if f is twice differentiable in U_+ and if $f' \neq 0, f \cdot f'' > 0$ in U_+, then, for each x_0 in U_+ the sequence x_k is monotonically decreasing to α.

Proof of quadratic convergence for Newton's iterative method

According to Taylor's theorem, any function $f(x)$ which has a continuous second derivative can be represented by an expansion about a point that is close to a root of $f(x)$. Suppose this root is α. Then the expansion of $f(\alpha)$ about x_n is:

$$f(\alpha) = f(x_n) + f'(x_n)(\alpha - x_n) + R_1 \qquad \textbf{(1)}$$

where the Lagrange form of the Taylor series expansion remainder is

$$R_1 = \frac{1}{2!} f''(\xi_n)(\alpha - x_n)^2 ,$$

where ξ_n is in between x_n and α.

Since α is the root, (**1**) becomes:

$$0 = f(\alpha) = f(x_n) + f'(x_n)(\alpha - x_n) + \tfrac{1}{2} f''(\xi_n)(\alpha - x_n)^2 \qquad \textbf{(2)}$$

Dividing equation (**2**) by $f'(x_n)$ and rearranging gives

$$\frac{f(x_n)}{f'(x_n)} + (\alpha - x_n) = \frac{-f''(\xi_n)}{2f'(x_n)} (\alpha - x_n)^2 \qquad \textbf{(3)}$$

Remembering that x_{n+1} is defined by

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} , \qquad \textbf{(4)}$$

one finds that

$$-f''(\xi_n) \quad {}^2$$

$$\underbrace{\alpha - x_{n+1}}_{\varepsilon_{n+1}} = \frac{}{2f'(x_n)}(\underbrace{\alpha - x_n}_{\varepsilon_n}) .$$

That is,

$$\varepsilon_{n+1} = \frac{-f''(\xi_n)}{2f'(x_n)} \cdot \varepsilon_n^2 . \qquad \textbf{(5)}$$

Taking the absolute value of both sides gives

$$|\varepsilon_{n+1}| = \frac{|f''(\xi_n)|}{2\,|f'(x_n)|} \cdot \varepsilon_n^2 . \qquad \textbf{(6)}$$

Equation (**6**) shows that the order of convergence is at least quadratic if the following conditions are satisfied:

1. $f'(x) \neq 0$; for all $x \in I$, where I is the interval $[\alpha - |\varepsilon_0|, \alpha + |\varepsilon_0|]$;
2. $f''(x)$ is continuous, for all $x \in I$;
3. $M\,|\varepsilon_0| < 1$

where M is given by

$$M = \frac{1}{2}\left(\sup_{x\in I} |f''(x)|\right)\left(\sup_{x\in I} \frac{1}{|f'(x)|}\right).$$

If these conditions hold,

$$|\varepsilon_{n+1}| \leq M \cdot \varepsilon_n^2 .$$

Fourier conditions

Suppose that $f(x)$ is a concave function on an interval, which is strictly increasing. If it is negative at the left endpoint and positive at the right endpoint, the intermediate value theorem guarantees that there is a zero ζ of f somewhere in the interval. From geometrical principles, it can be seen that the Newton iteration x_i starting at the left endpoint is monotonically increasing and convergent, necessarily to ζ.[7]

Joseph Fourier introduced a modification of Newton's method starting at the right endpoint:

$$y_{i+1} = y_i - \frac{f(y_i)}{f'(x_i)}.$$

This sequence is monotonically decreasing and convergent. By passing to the limit in this definition, it can be seen that the limit of y_i must also be the zero ζ.[7]

So, in the case of a concave increasing function with a zero, initialization is largely irrelevant. Newton iteration starting anywhere left of the zero will converge, as will Fourier's modified Newton iteration starting anywhere right of the zero. The accuracy at any step of the iteration can be determined directly from the difference between the location of the iteration from the left and the location of the iteration from the right. If f is twice continuously differentiable, it can be proved using Taylor's theorem that

$$\lim_{i\to\infty} \frac{y_{i+1} - x_{i+1}}{(y_i - x_i)^2} = -\frac{1}{2}\frac{f''(\zeta)}{f'(\zeta)},$$

showing that this difference in locations converges quadratically to zero.[7]

All of the above can be extended to systems of equations in multiple variables, although in that context the relevant concepts of monotonicity and concavity are more subtle to formulate.[8] In the case of single equations in a single variable, the above monotonic convergence of Newton's method can also be generalized to replace concavity by positivity or negativity conditions on an arbitrary higher-order derivative of f. However, in this generalization, Newton's iteration is modified so as to be based on Taylor polynomials rather than the tangent line. In the case of concavity, this modification coincides with the standard Newton method.[9]

Examples

Use of Newton's method to compute square roots

Newton's method is one of many known methods of computing square roots. Given a positive number a, the problem of finding a number x such that $x^2 = a$ is equivalent to finding a root of the function $f(x) = x^2 - a$. The Newton iteration defined by this function is given by

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} = x_n - \frac{x_n^2 - a}{2x_n} = \frac{1}{2}\left(x_n + \frac{a}{x_n}\right).$$

This happens to coincide with the "Babylonian" method of finding square roots, which consists of replacing an approximate root x_n by the arithmetic mean of x_n and a/x_n. By performing this iteration, it is possible to evaluate a square root to any desired accuracy by only using the basic arithmetic operations.

The following three tables show examples of the result of this computation for finding the square root of 612, with the iteration initialized at the values of 1, 10, and −20. Each row in a "x_n" column is obtained by applying the preceding formula to the entry above it, for instance

$$306.5 = \frac{1}{2}\left(1 + \frac{612}{1}\right).$$

x_n	$f(x_n)$
1	−611
306.5	9.3330×10^4
154.2483686786	2.3180×10^4
79.1079978644	5.6461×10^3
43.4221286822	1.2735×10^3
28.7581624288	215.03
25.0195385369	13.977
24.7402106712	7.8024×10^{-2}
24.7386338040	2.4865×10^{-6}
24.7386337537	2.5256×10^{-15}

x_n	$f(x_n)$
10	−512
35.6	655.36
26.3955056180	84.722
24.7906354925	2.5756
24.7386882941	2.6985×10^{-3}
24.7386337538	2.9746×10^{-9}

x_n	$f(x_n)$
−20	−212
−25.3	28.09
−24.7448616601	0.30818
−24.7386345374	3.8777×10^{-5}
−24.7386337537	6.1424×10^{-13}

The correct digits are underlined. It is seen that with only a few iterations one can obtain a solution accurate to many decimal places. The first table shows that this is true even if the Newton iteration were initialized by the very inaccurate guess of 1.

When computing any nonzero square root, the first derivative of f must be nonzero at the root, and that f is a smooth function. So, even before any computation, it is known that any convergent Newton iteration has a quadratic rate of convergence. This is reflected in the above tables by the fact that once a Newton iterate gets close to the root, the number of correct digits approximately doubles with each iteration.

Solution of $\cos(x) = x^3$ using Newton's method

Consider the problem of finding the positive number x with $\cos x = x^3$. We can rephrase that as finding the zero of $f(x) = \cos(x) - x^3$. We have $f'(x) = -\sin(x) - 3x^2$. Since $\cos(x) \leq 1$ for all x and $x^3 > 1$ for $x > 1$, we know that our solution lies between 0 and 1.

A starting value of 0 will lead to an undefined result which illustrates the importance of using a starting point close to the solution. For example, with an initial guess $x_0 = 0.5$, the sequence given by Newton's method is:

$$\begin{aligned}
x_1 &= x_0 - \frac{f(x_0)}{f'(x_0)} &&= 0.5 - \frac{\cos 0.5 - 0.5^3}{-\sin 0.5 - 3 \times 0.5^2} &&= 1.112\,141\,637\,097\ldots \\
x_2 &= x_1 - \frac{f(x_1)}{f'(x_1)} &&= \qquad \vdots &&= \underline{0.}909\,672\,693\,736\ldots \\
x_3 &= \qquad \vdots &&= \qquad \vdots &&= \underline{0.86}7\,263\,818\,209\ldots \\
x_4 &= \qquad \vdots &&= \qquad \vdots &&= \underline{0.865\,47}7\,135\,298\ldots \\
x_5 &= \qquad \vdots &&= \qquad \vdots &&= \underline{0.865\,474\,033\,1}11\ldots \\
x_6 &= \qquad \vdots &&= \qquad \vdots &&= \underline{0.865\,474\,033\,102}\ldots
\end{aligned}$$

The correct digits are underlined in the above example. In particular, x_6 is correct to 12 decimal places. We see that the number of correct digits after the decimal point increases from 2 (for x_3) to 5 and 10, illustrating the quadratic convergence.

Slow convergence

The function $f(x) = x^2$ has a root at 0.[10] Since f is continuously differentiable at its root, the theory guarantees that Newton's method as initialized sufficiently close to the root will converge. However, since the derivative $f\,'$ is zero at the root, quadratic convergence is not ensured by the theory. In this particular example, the Newton iteration is given by

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

It is visible from this that Newton's method could be initialized anywhere and converge to zero, but at only a linear rate. If initialized at 1, dozens of iterations would be required before ten digits of accuracy are achieved.

The function $f(x) = x + x^{4/3}$ also has a root at 0, where it is continuously differentiable. Although the first derivative $f\,'$ is nonzero at the root, the second derivative $f\,''$ is nonexistent there, so that quadratic convergence cannot be guaranteed. In fact the Newton iteration is given by

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} = \frac{x^{4/3}}{3 + 4x^{1/3}}.$$

From this, it can be seen that the rate of convergence is superlinear but subquadratic. This can be seen in the following tables, the left of which shows Newton's method applied to the above $f(x) = x + x^{4/3}$ and the right of which shows Newton's method applied to $f(x) = x + x^2$. The quadratic convergence in iteration shown on the right is illustrated by the orders of magnitude in the distance from the iterate to the true root (0,1,2,3,5,10,20,39,...) being approximately doubled from row to row. While the convergence on the left is superlinear, the order of magnitude is only multiplied by about 4/3 from row to row (0,1,2,4,5,7,10,13,...).

x_n	$x + x_n^{4/3}$
1	2
1.4286×10^{-1}	2.1754×10^{-1}
1.4669×10^{-2}	1.8260×10^{-2}
9.0241×10^{-4}	9.8961×10^{-4}
2.5750×10^{-5}	2.6511×10^{-5}
2.4386×10^{-7}	2.4539×10^{-7}
5.0366×10^{-10}	5.0406×10^{-10}
1.3344×10^{-13}	1.3344×10^{-13}

x_n	$x + x_n^{2}$
1	2
3.3333×10^{-1}	4.4444×10^{-1}
6.6666×10^{-2}	7.1111×10^{-2}
3.9216×10^{-3}	3.9369×10^{-3}
1.5259×10^{-5}	1.5259×10^{-5}
2.3283×10^{-10}	2.3283×10^{-10}
5.4210×10^{-20}	5.4210×10^{-20}
2.9387×10^{-39}	2.9387×10^{-39}

The rate of convergence is distinguished from the number of iterations required to reach a given accuracy. For example, the function $f(x) = x^{20} - 1$ has a root at 1. Since $f'(1) \neq 0$ and f is smooth, it is known that any Newton iteration convergent to 1 will converge quadratically. However, if initialized at 0.5, the first few iterates of Newton's method are approximately 26214, 24904, 23658, 22476, decreasing slowly, with only the 200th iterate being 1.0371. The following iterates are 1.0103, 1.00093, 1.0000082, and 1.00000000065, illustrating quadratic convergence. This highlights that quadratic convergence of a Newton iteration does not mean that only few iterates are required; this only applies once the sequence of iterates is sufficiently close to the root.[11]

Convergence dependent on initialization

The function $f(x) = x(1 + x^2)^{-1/2}$ has a root at 0. The Newton iteration is given by

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} = -x_n^3.$$

From this, it can be seen that there are three possible phenomena for a Newton iteration. If initialized strictly between ± 1, the Newton iteration will converge (super-)quadratically to 0; if initialized exactly at 1 or -1, the Newton iteration will oscillate endlessly between ± 1; if initialized anywhere else, the Newton iteration will diverge.[12] This same trichotomy occurs for $f(x) = \arctan x$.[10]

In cases where the function in question has multiple roots, it can be difficult to control, via choice of initialization, which root (if any) is identified by Newton's method. For example, the function $f(x) = x(x^2 - 1)(x - 3)e^{-(x-1)^2/2}$ has roots at −1, 0, 1, and 3.[13] If initialized at −1.488, the Newton iteration converges to 0; if initialized at −1.487, it diverges to ∞; if initialized at −1.486, it converges to −1; if initialized at −1.485, it diverges to $-\infty$; if initialized at −1.4843, it converges to 3; if initialized at −1.484, it converges to 1. This kind of subtle dependence on initialization is not uncommon; it is frequently studied in the complex plane in the form of the Newton fractal.

Divergence even when initialization is close to the root

Consider the problem of finding a root of $f(x) = x^{1/3}$. The Newton iteration is

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} = x_n - \frac{x_n^{1/3}}{\frac{1}{3}x_n^{-2/3}} = -2x_n.$$

Unless Newton's method is initialized at the exact root 0, it is seen that the sequence of iterates will fail to converge. For example, even if initialized at the reasonably accurate guess of 0.001, the first several iterates are −0.002, 0.004, −0.008, 0.016, reaching 1048.58, −2097.15, ... by the 20th iterate. This failure of convergence is not contradicted by the analytic theory, since in this case f is not differentiable at its root.

In the above example, failure of convergence is reflected by the failure of $f(x_n)$ to get closer to zero as n increases, as well as by the fact that successive iterates are growing further and further apart. However, the function $f(x) = x^{1/3}e^{-x^2}$ also has a root at 0. The Newton iteration is given by

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} = x_n\left(1 - \frac{3}{1-6x_n^2}\right).$$

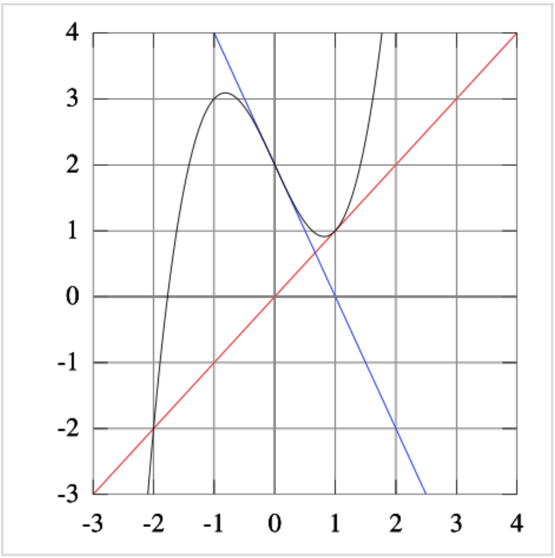
In this example, where again f is not differentiable at the root, any Newton iteration not starting exactly at the root will diverge, but with both $x_{n+1} - x_n$ and $f(x_n)$ converging to zero.[14] This is seen in the following table showing the iterates with initialization 1:

x_n	$f(x_n)$
1	0.36788
1.6	9.0416×10^{-2}
1.9342	2.9556×10^{-2}
2.2048	1.0076×10^{-2}
2.4396	3.5015×10^{-3}
2.6505	1.2307×10^{-3}
2.8437	4.3578×10^{-4}
3.0232	1.5513×10^{-4}

Although the convergence of $x_{n+1} - x_n$ in this case is not very rapid, it can be proved from the iteration formula. This example highlights the possibility that a stopping criterion for Newton's method based only on the smallness of $x_{n+1} - x_n$ and $f(x_n)$ might falsely identify a root.

Oscillatory behavior

It is easy to find situations for which Newton's method oscillates endlessly between two distinct values. For example, for Newton's method as applied to a function f to oscillate between 0 and 1, it is only necessary that the tangent line to f at 0 intersects the x-axis at 1 and that the tangent line to f at 1 intersects the x-axis at 0.[14] This is the case, for example, if $f(x) = x^3 - 2x + 2$. For this function, it is even the case that Newton's iteration as initialized sufficiently close to 0 or 1 will *asymptotically* oscillate between these values. For example, Newton's method as initialized at 0.99 yields iterates 0.99, −0.06317, 1.00628, 0.03651, 1.00196, 0.01162, 1.00020, 0.00120, 1.000002, and so on. This behavior is present despite the presence of a root of f approximately equal to −1.76929.



The tangent lines of $x^3 - 2x + 2$ at 0 and 1 intersect the x-axis at 1 and 0 respectively, illustrating why Newton's method oscillates between these values for some starting points.

Undefinedness of Newton's method

In some cases, it is not even possible to perform the Newton iteration. For example, if $f(x) = x^2 - 1$, then the Newton iteration is defined by

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

So Newton's method cannot be initialized at 0, since this would make x_1 undefined. Geometrically, this is because the tangent line to f at 0 is horizontal (i.e. $f'(0) = 0$), never intersecting the x-axis.

Even if the initialization is selected so that the Newton iteration can begin, the same phenomenon can block the iteration from being indefinitely continued.

If f has an incomplete domain, it is possible for Newton's method to send the iterates outside of the domain, so that it is impossible to continue the iteration.[14] For example, the natural logarithm function $f(x) = \ln x$ has a root at 1, and is defined only for positive x. Newton's iteration in this case is given by

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

So if the iteration is initialized at e, the next iterate is 0; if the iteration is initialized at a value larger than e, then the next iterate is negative. In either case, the method cannot be continued.

Multidimensional formulations

Systems of equations

k variables, *k* functions

One may also use Newton's method to solve systems of k equations, which amounts to finding the (simultaneous) zeroes of k continuously differentiable functions $f : \mathbb{R}^k \to \mathbb{R}$. This is equivalent to finding the zeroes of a single vector-valued function $F : \mathbb{R}^k \to \mathbb{R}^k$. In the formulation given above, the scalars x_n are replaced by vectors $\mathbf{x}_n$ and instead of dividing the function $f(x_n)$ by its derivative $f'(x_n)$ one instead has to left multiply the function $F(\mathbf{x}_n)$ by the inverse of its $k \times k$ Jacobian matrix $J_F(\mathbf{x}_n)$.[15][16][17] This results in the expression

$$\mathbf{x}_{n+1} = \mathbf{x}_n - J_F(\mathbf{x}_n)^{-1} F(\mathbf{x}_n).$$

or, by solving the system of linear equations

$$J_F(\mathbf{x}_n)(\mathbf{x}_{n+1} - \mathbf{x}_n) = -F(\mathbf{x}_n)$$

for the unknown $\mathbf{x}_{n+1} - \mathbf{x}_n$.[18]

k variables, *m* equations, with *m* > *k*

The k-dimensional variant of Newton's method can be used to solve systems of greater than k (nonlinear) equations as well if the algorithm uses the generalized inverse of the non-square Jacobian matrix $J^+ = (J^\mathrm{T}J)^{-1}J^\mathrm{T}$ instead of the inverse of J. If the nonlinear system has no solution, the method attempts to find a solution in the non-linear least squares sense. See Gauss–Newton algorithm for more information.

Example

For example, the following set of equations needs to be solved for vector of points $[\, x_1, x_2 \,]$, given the vector of known values $[\, 2, 3 \,]$.[19]

$$\begin{array}{lcr} 5\ x_1^2 + x_1\ x_2^2 + \sin^2(2\ x_2) & = & 2 \\ e^{2\ x_1 - x_2} + 4\ x_2 & = & 3 \end{array}$$

the function vector, $F(X_k)$, and Jacobian Matrix, $J(X_k)$ for iteration k, and the vector of known values,

Y , are defined below.

$$F(X_k) \;=\; \begin{bmatrix} f_1(X_k) \\ f_2(X_k) \end{bmatrix} \;=\; \begin{bmatrix} 5\,x_1^2 + x_1\,x_2^2 + \sin^2(2\,x_2) \\ e^{2\,x_1 - x_2} + 4\,x_2 \end{bmatrix}_k$$

$$J(X_k) = \begin{bmatrix} \frac{\partial f_1(X)}{\partial x_1}\,, & \frac{\partial f_1(X)}{\partial x_2} \\ \frac{\partial f_2(X)}{\partial x_1}\,, & \frac{\partial f_2(X)}{\partial x_2} \end{bmatrix}_k \;=\; \begin{bmatrix} 10\,x_1 + x_2^2\,, & 2\,x_1\,x_2 + 4\,\sin(2\,x_2)\,\cos(2\,x_2) \\ 2\,e^{2\,x_1 - x_2}\,, & -\,e^{2\,x_1 - x_2} + 4 \end{bmatrix}_k$$

$$Y = \begin{bmatrix} 2 \\ 3 \end{bmatrix}$$

Note that $F(X_k)$ could have been rewritten to absorb Y , and thus eliminate Y from the equations. The equation to solve for each iteration are

$$\begin{bmatrix} 10\,x_1 + x_2^2\,, & 2x_1x_2 + 4\,\sin(2\,x_2)\,\cos(2\,x_2) \\ 2\,e^{2\,x_1 - x_2}\,, & -\,e^{2\,x_1 - x_2} + 4 \end{bmatrix}_k \begin{bmatrix} c_1 \\ c_2 \end{bmatrix}_{k+1} = \begin{bmatrix} 5\,x_1^2 + x_1\,x_2^2 + \sin^2(2\,x_2) - 2 \\ e^{2\,x_1 - x_2} + 4\,x_2 - 3 \end{bmatrix}_k$$

and

$$X_{k+1} \;=\; X_k - C_{k+1}$$

The iterations should be repeated until $\left[\sum_{i=1}^{i=2} \left| f(x_i)_k - (y_i)_k \right|\right] < E$, where E is a value acceptably small enough to meet application requirements.

If vector X_0 is initially chosen to be $[\,1 \quad 1\,]$, that is, $x_1 = 1$, and $x_2 = 1$, and E , is chosen to be 1.10^{-3}, then the example converges after four iterations to a value of $X_4 = [\,0.567297, \; -0.309442\,]$.

Iterations

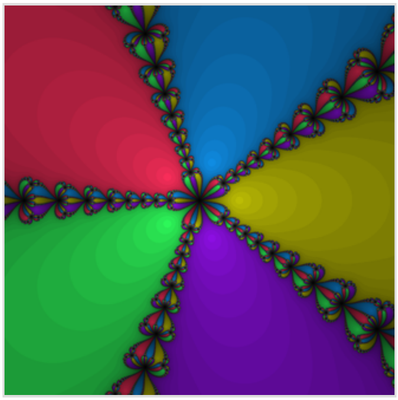
The following iterations were made during the course of the solution.

Converging iteration sequence

Step	Variable	Value
0	$x =$	$[\,1,\quad 1\,]$
	$f(x) =$	$[\,6.82682,\quad 6.71828\,]$
1	$J =$	$\begin{bmatrix} 11, & 0.486395 \\ 5.43656, & 1.28172 \end{bmatrix}$
	$c =$	$[\,0.382211,\quad 1.27982\,]$
	$x =$	$[\,0.617789,\quad -0.279818\,]$
	$f(x) =$	$[\,2.23852,\quad 3.43195\,]$
2	$J =$	$\begin{bmatrix} 6.25618, & -2.1453 \\ 9.10244, & -0.551218 \end{bmatrix}$
	$c =$	$[\,0.0494549,\quad 0.0330411\,]$
	$x =$	$[\,0.568334,\quad -0.312859\,]$
	$f(x) =$	$[\,2.01366,\quad 3.00966\,]$
3	$J =$	$\begin{bmatrix} 5.78122, & -2.25449 \\ 8.52219, & -0.261095 \end{bmatrix}$
	$c =$	$[\,0.00102862,\quad -0.00342339\,]$
	$x =$	$[\,0.567305,\quad -0.309435\,]$
	$f(x) =$	$[\,2.00003,\quad 3.00006\,]$
4	$J =$	$\begin{bmatrix} 5.7688, & -2.24118 \\ 8.47561, & -0.237805 \end{bmatrix}$
	$c =$	$[\,7.73132\times 10^{-6},\quad 6.93265\times 10^{-6}\,]$
	$x =$	$[\,0.567297,\quad -0.309442\,]$
	$f(x) =$	$[\,2,\quad 3\,]$

Complex functions

When dealing with complex functions, Newton's method can be directly applied to find their zeroes.[20] Each zero has a basin of attraction in the complex plane, the set of all starting values that cause the method to converge to that particular zero. These sets can be mapped as in the image shown. For many complex functions, the boundaries of the basins of attraction are fractals.


Basins of attraction for $x^5 - 1 = 0$; darker means more iterations to converge.

In some cases there are regions in the complex plane which are not in any of these basins of attraction, meaning the iterates do not converge. For example,[21] if one uses a real initial condition to seek a root of $x^2 + 1$, all subsequent iterates will be real numbers and so the iterations cannot converge to either root, since both roots are non-real. In this case almost all real initial conditions lead to chaotic behavior, while some initial conditions iterate either to infinity or to repeating cycles of any finite length.

Curt McMullen has shown that for any possible purely iterative algorithm similar to Newton's method, the algorithm will diverge on some open regions of the complex plane when applied to some polynomial of degree 4 or higher. However, McMullen gave a generally convergent algorithm for polynomials of degree 3.[22] Also, for any polynomial, Hubbard, Schleicher, and Sutherland gave a method for selecting a set of initial points such that Newton's method will certainly converge at one of them at least.[23]

In a Banach space

Another generalization is Newton's method to find a root of a functional F defined in a Banach space. In this case the formulation is

$$X_{n+1} = X_n - \left(F'(X_n)\right)^{-1} F(X_n),$$

where $F'(X_n)$ is the Fréchet derivative computed at X_n. One needs the Fréchet derivative to be boundedly invertible at each X_n in order for the method to be applicable. A condition for existence of and convergence to a root is given by the Newton–Kantorovich theorem.[24]

Nash–Moser iteration

In the 1950s, John Nash developed a version of the Newton's method to apply to the problem of constructing isometric embeddings of general Riemannian manifolds in Euclidean space. The *loss of derivatives* problem, present in this context, made the standard Newton iteration inapplicable, since it could not be continued indefinitely (much less converge). Nash's solution involved the introduction of smoothing operators into the iteration. He was able to prove the convergence of his smoothed Newton method, for the purpose of proving an implicit function theorem for isometric embeddings. In the 1960s, Jürgen Moser showed that Nash's methods were flexible enough to apply to problems beyond isometric embedding, particularly in celestial mechanics. Since then, a number of mathematicians, including Mikhael Gromov and Richard Hamilton, have found generalized abstract versions of the Nash–Moser theory.[25][26] In Hamilton's formulation, the Nash–Moser theorem forms a generalization of the Banach space Newton method which takes place in certain Fréchet spaces.

Modifications

Quasi-Newton methods

When the Jacobian is unavailable or too expensive to compute at every iteration, a quasi-Newton method can be used.

Chebyshev's third-order method

Over p-adic numbers

In p-adic analysis, the standard method to show a polynomial equation in one variable has a p-adic root is Hensel's lemma, which uses the recursion from Newton's method on the p-adic numbers. Because of the more stable behavior of addition and multiplication in the p-adic numbers compared to the real numbers (specifically, the unit ball in the p-adics is a ring), convergence in Hensel's lemma can be guaranteed under much simpler hypotheses than in the classical Newton's method on the real line.

q-analog

Newton's method can be generalized with the q-analog of the usual derivative.[27]

Modified Newton methods

Maehly's procedure

A nonlinear equation has multiple solutions in general. But if the initial value is not appropriate, Newton's method may not converge to the desired solution or may converge to the same solution found earlier. When we have already found N solutions of $f(x)=0$, then the next root can be found by applying Newton's method to the next equation:[28][29]

$$F(x)=\frac{f(x)}{\prod_{i=1}^{N}(x-x_i)}=0.$$

This method is applied to obtain zeros of the Bessel function of the second kind.[30]

Hirano's modified Newton method

Hirano's modified Newton method is a modification conserving the convergence of Newton method and avoiding unstableness.[31] It is developed to solve complex polynomials.

Interval Newton's method

Combining Newton's method with interval arithmetic is very useful in some contexts. This provides a stopping criterion that is more reliable than the usual ones (which are a small value of the function or a small variation of the variable between consecutive iterations). Also, this may detect cases where Newton's method converges theoretically but diverges numerically because of an insufficient floating-point precision (this is typically the case for polynomials of large degree, where a very small change of the variable may change dramatically the value of the function; see Wilkinson's polynomial).[32][33]

Consider $f \to \mathcal{C}^1(X)$, where X is a real interval, and suppose that we have an interval extension F' of f', meaning that F' takes as input an interval $Y \subseteq X$ and outputs an interval $F'(Y)$ such that:

$$\begin{aligned}F'([y,y]) &= \{f'(y)\}\\ F'(Y) &\supseteq \{f'(y) \mid y \in Y\}.\end{aligned}$$

We also assume that $0 \notin F'(X)$, so in particular f has at most one root in X. We then define the interval Newton operator by:

$$N(Y)=m-\frac{f(m)}{F'(Y)}=\left\{m-\frac{f(m)}{z}\;\middle|\; z\in F'(Y)\right\}$$

where $m \in Y$. Note that the hypothesis on F' implies that $N(Y)$ is well defined and is an interval (see interval arithmetic for further details on interval operations). This naturally leads to the following sequence:

$$\begin{aligned}X_0 &= X\\ X_{k+1} &= N(X_k)\cap X_k.\end{aligned}$$

The mean value theorem ensures that if there is a root of f in X_k, then it is also in X_{k+1}. Moreover, the hypothesis on F' ensures that X_{k+1} is at most half the size of X_k when m is the midpoint of Y, so this sequence converges towards $[x^*, x^*]$, where x^* is the root of f in X.

If $F'(X)$ strictly contains 0, the use of extended interval division produces a union of two intervals for $N(X)$; multiple roots are therefore automatically separated and bounded.

Applications

Minimization and maximization problems

Newton's method can be used to find a minimum or maximum of a function $f(x)$. The derivative is zero at a minimum or maximum, so local minima and maxima can be found by applying Newton's method to the derivative.[34] The iteration becomes:

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

Multiplicative inverses of numbers and power series

An important application is Newton–Raphson division, which can be used to quickly find the reciprocal of a number a, using only multiplication and subtraction, that is to say the number x such that $\frac{1}{x} = a$. We can rephrase that as finding the zero of $f(x) = \frac{1}{x} - a$. We have $f'(x) = -\frac{1}{x^2}$.

Newton's iteration is

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} = x_n + \frac{\frac{1}{x_n} - a}{\frac{1}{x_n^2}} = x_n(2 - ax_n).$$

Therefore, Newton's iteration needs only two multiplications and one subtraction.

This method is also very efficient to compute the multiplicative inverse of a power series.

Solving transcendental equations

Many transcendental equations can be solved up to an arbitrary precision by using Newton's method. For example, finding the cumulative probability density function, such as a Normal distribution to fit a known probability generally involves integral functions with no known means to solve in closed form. However, computing the derivatives needed to solve them numerically with Newton's method is generally known, making numerical solutions possible. For an example, see the numerical solution to the inverse Normal cumulative distribution.

Numerical verification for solutions of nonlinear equations

A numerical verification for solutions of nonlinear equations has been established by using Newton's method multiple times and forming a set of solution candidates.

Code

The following is an example of a possible implementation of Newton's method in the Python (version 3.x) programming language for finding a root of a function `f` which has derivative `f_prime`.

The initial guess will be $x_0 = 1$ and the function will be $f(x) = x^2 - 2$ so that $f'(x) = 2x$.

Each new iteration of Newton's method will be denoted by `x1`. We will check during the computation whether the denominator (`yprime`) becomes too small (smaller than `epsilon`), which would be the case if $f'(x_n) \approx 0$, since otherwise a large amount of error could be introduced.

```python
def f(x):
    return x**2 - 2   # f(x) = x^2 - 2

def f_prime(x):
    return 2*x        # f'(x) = 2x

def newtons_method(x0, f, f_prime, tolerance, epsilon, max_iterations):
    """Newton's method

    Args:
      x0:              The initial guess
      f:               The function whose root we are trying to find
      f_prime:         The derivative of the function
      tolerance:       Stop when iterations change by less than this
      epsilon:         Do not divide by a number smaller than this
      max_iterations:  The maximum number of iterations to compute
    """
    for _ in range(max_iterations):
        y = f(x0)
        yprime = f_prime(x0)

        if abs(yprime) < epsilon:       # Give up if the denominator is too small
            break

        x1 = x0 - y / yprime            # Do Newton's computation

        if abs(x1 - x0) <= tolerance:   # Stop when the result is within the desired tolerance
            return x1                   # x1 is a solution within tolerance and maximum number of iterations

        x0 = x1                         # Update x0 to start the process again

    return None                         # Newton's method did not converge
```

See also

- Aitken's delta-squared process
- Bisection method
- Euler method
- Fast inverse square root
- Fisher scoring
- Gradient descent
- Integer square root
- Kantorovich theorem
- Laguerre's method
- Methods of computing square roots
- Newton's method in optimization
- Richardson extrapolation
- Root-finding algorithm
- Secant method
- Steffensen's method
- Subgradient method

Notes

1. "Chapter 2. Seki Takakazu" (http://www.ndl.go.jp/math/e/s1/2.html). *Japanese Mathematics in the Edo Period*. National Diet Library. Retrieved 24 February 2019.
2. Wallis, John (1685). *A Treatise of Algebra, both Historical and Practical* (http://www.e-rara.ch/zut/content/titleinfo/2507537). Oxford: Richard Davis. doi:10.3931/e-rara-8842 (https://doi.org/10.3931%2Fe-rara-8842).
3. Raphson, Joseph (1697). *Analysis Æequationum Universalis* (https://archive.org/details/bub_gb_4nlbAAAAQAAJ) (in Latin) (2nd ed.). London: Thomas Bradyll. doi:10.3931/e-rara-13516 (https://doi.org/10.3931%2Fe-rara-13516).
4. "Accelerated and Modified Newton Methods" (https://web.archive.org/web/20190524083302/http://mathfaculty.fullerton.edu/mathews/n2003/NewtonAccelerateMod.html). Archived from the original (http://mathfaculty.fullerton.edu/mathews/n2003/newtonacceleratemod.html) on 24 May 2019. Retrieved 4 March 2016.
5. Ryaben'kii, Victor S.; Tsynkov, Semyon V. (2006), *A Theoretical Introduction to Numerical Analysis* (https://books.google.com/books?id=V8gWP031-hEC&pg=PA243), CRC Press, p. 243, ISBN 9781584886075.
6. Süli & Mayers 2003, Exercise 1.6

7. Ostrowski, A. M. (1973). *Solution of equations in Euclidean and Banach spaces*. Pure and Applied Mathematics. Vol. 9 (Third edition of 1960 original ed.). New York–London: Academic Press. MR 0359306 (https://mathscinet.ams.org/mathscinet-getitem?mr=0359306). Zbl 0304.65002 (https://zbmath.org/?format=complete&q=an:0304.65002).
8. Ortega and Rheinboldt, Section 13.3
9. Traub, J. F. (1964). *Iterative methods for the solution of equations*. Prentice-Hall Series in Automatic Computation. Englewood Cliffs, NJ: Prentice-Hall, Inc. MR 0169356 (https://mathscinet.ams.org/mathscinet-getitem?mr=0169356). Zbl 0121.11204 (https://zbmath.org/?format=complete&q=an:0121.11204).
10. J. E. Dennis, Jr. and Robert B. Schnabel. Numerical methods for unconstrained optimization and nonlinear equations. SIAM
11. Anthony Ralston and Philip Rabinowitz. A first course in numerical analysis, second edition
12. Yuri Nesterov. Lectures on convex optimization, second edition. Springer Optimization and its Applications, Volume 137.
13. Süli & Mayers 2003.
14. Kenneth L. Judd. Numerical methods in economics. MIT Press
15. Burden, Burton; Fairs, J. Douglas; Reunolds, Albert C (July 1981). *Numerical Analysis* (https://archive.org/details/numericalanaly00burd/) (2nd ed.). Boston, MA, United States: Prindle, Weber & Schmidt. pp. 448–452. ISBN 0-87150-314-X. OCLC 1036752194 (https://search.worldcat.org/oclc/1036752194).
16. Evans, Gwynne A. (1995). *Practical Numerical Analysis* (https://archive.org/details/practicalnumeric0000evan/). Chichester: John Wiley & Sons. pp. 30–33. ISBN 0471955353. OCLC 1319419671 (https://search.worldcat.org/oclc/1319419671).
17. Demidovich, Boris Pavlovich; Maron, Isaak Abramovich (1981). *Computational Mathematics* (https://archive.org/details/computational-mathematics/mode/2up) (Third ed.). Moscow: MIR Publishers. pp. 460–478. ISBN 9780828507042.
18. Kiusalaas, Jaan (March 2013). *Numerical Methods in Engineering with Python 3* (https://www.cambridge.org/9781107033856) (3rd ed.). New York: Cambridge University Press. pp. 175–176. ISBN 978-1-107-03385-6.
19. This example is similar to one in reference,[15] pages 451 and 452, but simplified to two equations instead of three.
20. Henrici, Peter (1974). *Applied and Computational Complex Analysis*. Vol. 1. Wiley. ISBN 9780471598923.
21. Strang, Gilbert (January 1991). "A chaotic search for i". *The College Mathematics Journal*. **22** (1): 3–12. doi:10.2307/2686733 (https://doi.org/10.2307%2F2686733). JSTOR 2686733 (https://www.jstor.org/stable/2686733).
22. McMullen, Curt (1987). "Families of rational maps and iterative root-finding algorithms" (https://dash.harvard.edu/bitstream/handle/1/9876064/McMullen_FamiliesRationalMap.pdf?sequence=1) (PDF). *Annals of Mathematics*. Second Series. **125** (3): 467–493. doi:10.2307/1971408 (https://doi.org/10.2307%2F1971408). JSTOR 1971408 (https://www.jstor.org/stable/1971408).
23. Hubbard, John; Schleicher, Dierk; Sutherland, Scott (October 2001). "How to find all roots of complex polynomials by Newton's method" (https://dx.doi.org/10.1007/s002220100149). *Inventiones Mathematicae*. **146** (1): 1–33. Bibcode:2001InMat.146....1H (https://ui.adsabs.harvard.edu/abs/2001InMat.146....1H). doi:10.1007/s002220100149 (https://doi.org/10.1007%2Fs002220100149). ISSN 0020-9910 (https://search.worldcat.org/issn/0020-9910). S2CID 12603806 (https://api.semanticscholar.org/CorpusID:12603806).
24. Yamamoto, Tetsuro (2001). "Historical Developments in Convergence Analysis for Newton's and Newton-like Methods". In Brezinski, C.; Wuytack, L. (eds.). *Numerical Analysis: Historical Developments in the 20th Century*. North-Holland. pp. 241–263. ISBN 0-444-50617-9.
25. Hamilton, Richard S. (1982). "The inverse function theorem of Nash and Moser" (https://doi.org/10.1090%2Fs0273-0979-1982-15004-2). *Bulletin of the American Mathematical Society*. New Series. **7** (1): 65–222. doi:10.1090/s0273-0979-1982-15004-2 (https://doi.org/10.1090%2Fs0273-0979-1982-15004-2). MR 0656198 (https://mathscinet.ams.org/mathscinet-getitem?mr=0656198). Zbl 0499.58003 (https://zbmath.org/?format=complete&q=an:0499.58003).
26. Gromov, Mikhael (1986). *Partial differential relations*. Ergebnisse der Mathematik und ihrer Grenzgebiete (3). Vol. 9. Berlin: Springer-Verlag. doi:10.1007/978-3-662-02267-2 (https://doi.org/10.1007%2F978-3-662-02267-2). ISBN 3-540-12177-3. MR 0864505 (https://mathscinet.ams.org/mathscinet-getitem?mr=0864505).

27. Rajković, Predrag M.; Stanković, Miomir S.; Marinković, Slađana D. (2002). "Mean value theorems in q-calculus" (https://www.emis.de/journals/MV/0234/16.html). *Matematicki Vesnik*. **54** (3–4): 171–178.
28. Press et al. 2007
29. Stoer, Josef; Bulirsch, Roland (1980). *Introduction to numerical analysis* (https://archive.org/details/introductiontonu0000stoe/mode/2up). p. 279. OCLC 1244842246 (https://search.worldcat.org/oclc/1244842246).
30. Zhang, Shanjie; Jin, Jianming (1996). *Computation of Special Functions*. Wiley. ISBN 9780471119630.
31. Murota, Kazuo (1982). "Global Convergence of a Modified Newton Iteration for Algebraic Equations". *SIAM Journal on Numerical Analysis*. **19** (4): 793–799. Bibcode:1982SJNA...19..793M (https://ui.adsabs.harvard.edu/abs/1982SJNA...19..793M). doi:10.1137/0719055 (https://doi.org/10.1137%2F0719055).
32. Moore, R. E. (1979). *Methods and applications of interval analysis* (Vol. 2). Siam.
33. Hansen, E. (1978). Interval forms of Newtons method. *Computing*, 20(2), 153–163.
34. Boyd, Stephen; Vandenberghe, Lieven (2004). *Convex optimization*. Cambridge: Cambridge University Press. doi:10.1017/CBO9780511804441 (https://doi.org/10.1017%2FCBO9780511804441). ISBN 0-521-83378-7. MR 2061575 (https://mathscinet.ams.org/mathscinet-getitem?mr=2061575). Zbl 1058.90049 (https://zbmath.org/?format=complete&q=an:1058.90049).

References

- Gil, A.; Segura, J.; Temme, N. M. (2007). *Numerical methods for special functions* (https://www.researchgate.net/publication/220693008). Society for Industrial and Applied Mathematics. ISBN 978-0-89871-634-4.
- Süli, Endre; Mayers, David (2003). *An Introduction to Numerical Analysis*. Cambridge University Press. ISBN 0-521-00794-1.

Further reading

- Kendall E. Atkinson, *An Introduction to Numerical Analysis*, (1989) John Wiley & Sons, Inc, ISBN 0-471-62489-6
- Tjalling J. Ypma, Historical development of the Newton–Raphson method, *SIAM Review* **37** (4), 531–551, 1995. doi:10.1137/1037125 (https://doi.org/10.1137%2F1037125).
- Bonnans, J. Frédéric; Gilbert, J. Charles; Lemaréchal, Claude; Sagastizábal, Claudia A. (2006). *Numerical optimization: Theoretical and practical aspects* (https://www.springer.com/mathematics/applications/book/978-3-540-35445-1). Universitext (Second revised ed. of translation of 1997 French ed.). Berlin: Springer-Verlag. pp. xiv+490. doi:10.1007/978-3-540-35447-5 (https://doi.org/10.1007%2F978-3-540-35447-5). ISBN 3-540-35445-X. MR 2265882 (https://mathscinet.ams.org/mathscinet-getitem?mr=2265882).
- P. Deuflhard, *Newton Methods for Nonlinear Problems. Affine Invariance and Adaptive Algorithms.* Springer Series in Computational Mathematics, Vol. 35. Springer, Berlin, 2004. ISBN 3-540-21099-7.
- C. T. Kelley, *Solving Nonlinear Equations with Newton's Method*, no 1 in Fundamentals of Algorithms, SIAM, 2003. ISBN 0-89871-546-6.
- J. M. Ortega, W. C. Rheinboldt, *Iterative Solution of Nonlinear Equations in Several Variables.* Classics in Applied Mathematics, SIAM, 2000. ISBN 0-89871-461-3.
- Press, W. H.; Teukolsky, S. A.; Vetterling, W. T.; Flannery, B. P. (2007). "Chapter 9. Root Finding and Nonlinear Sets of Equations Importance Sampling" (http://apps.nrbook.com/empanel/index.html#pg=442) . *Numerical Recipes: The Art of Scientific Computing* (3rd ed.). New York: Cambridge University Press. ISBN 978-0-521-88068-8.. See especially Sections 9.4 (http://apps.nrbook.com/empanel/index.html#pg=456), 9.6 (http://apps.nrbook.com/empanel/index.html#pg=473), and 9.7 (http://apps.nrbook.com/empanel/index.html#pg=477).
- Avriel, Mordecai (1976). *Nonlinear Programming: Analysis and Methods*. Prentice Hall. pp. 216–221. ISBN 0-13-623603-0.

External links

- "Newton method" (https://www.encyclopediaofmath.org/index.php?title=Newton_method), *Encyclopedia of Mathematics*, EMS Press, 2001 [1994]
- Weisstein, Eric W. "Newton's Method" (https://mathworld.wolfram.com/NewtonsMethod.html). *MathWorld*.
- Newton's method, Citizendium. (http://en.citizendium.org/wiki/Newton%27s_method)
- Mathews, J., The Accelerated and Modified Newton Methods, Course notes. (https://web.archive.org/web/20190524083302/http://mathfaculty.fullerton.edu/mathews/n2003/NewtonAccelerateMod.html)
- Wu, X., Roots of Equations, Course notes. (http://www.ece.mcmaster.ca/~xwu/part2.pdf)