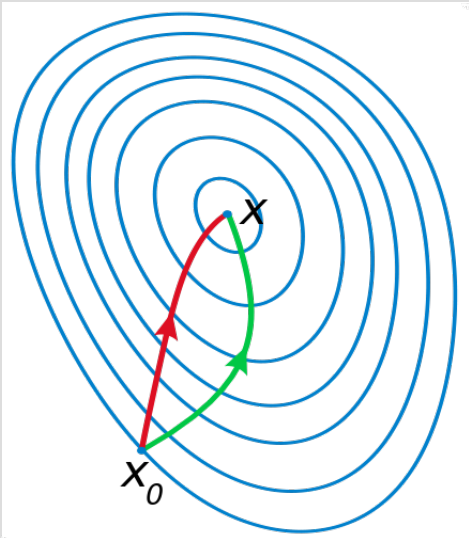
Newton's method in optimization

In calculus, **Newton's method** (also called **Newton–Raphson**) is an iterative method for finding the roots of a differentiable function F, which are solutions to the equation $F(x) = 0$. As such, Newton's method can be applied to the derivative f' of a twice-differentiable function f to find the roots of the derivative (solutions to $f'(x) = 0$), also known as the critical points of f. These solutions may be minima, maxima, or saddle points; see section "Several variables" in Critical point (mathematics) and also section "Geometric interpretation" in this article. This is relevant in optimization, which aims to find (global) minima of the function f.



A comparison of gradient descent (green) and Newton's method (red) for minimizing a function (with small step sizes). Newton's method uses curvature information (i.e. the second derivative) to take a more direct route.

Newton's method

The central problem of optimization is minimization of functions. Let us first consider the case of univariate functions, i.e., functions of a single real variable. We will later consider the more general and more practically useful multivariate case.

Given a twice differentiable function $f : \mathbb{R} \to \mathbb{R}$, we seek to solve the optimization problem

$$\min_{x \in \mathbb{R}} f(x).$$

Newton's method attempts to solve this problem by constructing a sequence $\{x_k\}$ from an initial guess (starting point) $x_0 \in \mathbb{R}$ that converges towards a minimizer x_* of f by using a sequence of second-order Taylor approximations of f around the iterates. The second-order Taylor expansion of f around x_k is

$$f(x_k + t) \approx f(x_k) + f'(x_k)t + \frac{1}{2}f''(x_k)t^2.$$

The next iterate x_{k+1} is defined so as to minimize this quadratic approximation in t, and setting $x_{k+1} = x_k + t$. If the second derivative is positive, the quadratic approximation is a convex function of t, and its minimum can be found by setting the derivative to zero. Since

$$0 = \frac{\mathrm{d}}{\mathrm{d}t}\left(f(x_k) + f'(x_k)t + \frac{1}{2}f''(x_k)t^2\right) = f'(x_k) + f''(x_k)t,$$

the minimum is achieved for

$$t = -\frac{f'(x_k)}{f''(x_k)}.$$

Putting everything together, Newton's method performs the iteration

$$x_{k+1} = x_k + t = x_k - \frac{f'(x_k)}{f''(x_k)}.$$

Geometric interpretation

The geometric interpretation of Newton's method is that at each iteration, it amounts to the fitting of a parabola to the graph of $f(x)$ at the trial value x_k, having the same slope and curvature as the graph at that point, and then proceeding to the maximum or minimum of that parabola (in higher dimensions, this may also be a saddle point), see below. Note that if f happens to *be* a quadratic function, then the exact extremum is found in one step.

Higher dimensions

The above iterative scheme can be generalized to $d > 1$ dimensions by replacing the derivative with the gradient (different authors use different notation for the gradient, including $f'(x) = \nabla f(x) = g_f(x) \in \mathbb{R}^d$), and the reciprocal of the second derivative with the inverse of the Hessian matrix (different authors use different notation for the Hessian, including $f''(x) = \nabla^2 f(x) = H_f(x) \in \mathbb{R}^{d\times d}$). One thus obtains the iterative scheme

$$x_{k+1} = x_k - [f''(x_k)]^{-1} f'(x_k), \qquad k \geq 0.$$

Often Newton's method is modified to include a small step size $0 < \gamma \leq 1$ instead of $\gamma = 1$:

$$x_{k+1} = x_k - \gamma[f''(x_k)]^{-1} f'(x_k).$$

This is often done to ensure that the Wolfe conditions, or much simpler and efficient Armijo's condition, are satisfied at each step of the method. For step sizes other than 1, the method is often referred to as the relaxed or damped Newton's method.

Convergence

If f is a strongly convex function with Lipschitz Hessian, then provided that x_0 is close enough to $x_* = \arg\min f(x)$, the sequence $x_0, x_1, x_2, \ldots$ generated by Newton's method will converge to the (necessarily unique) minimizer x_* of f quadratically fast.[1] That is,

$$\|x_{k+1} - x_*\| \leq \frac{1}{2}\|x_k - x_*\|^2, \qquad \forall k \geq 0.$$

Computing the Newton direction

Finding the inverse of the Hessian in high dimensions to compute the Newton direction $h = -(f''(x_k))^{-1} f'(x_k)$ can be an expensive operation. In such cases, instead of directly inverting the Hessian, it is better to calculate the vector h as the solution to the system of linear equations

$$[f''(x_k)]h = -f'(x_k)$$

which may be solved by various factorizations or approximately (but to great accuracy) using iterative methods. Many of these methods are only applicable to certain types of equations, for example the Cholesky factorization and conjugate gradient will only work if $f''(x_k)$ is a positive definite matrix. While this may seem like a limitation, it is often a useful indicator of something gone wrong; for example if a minimization problem is being approached and $f''(x_k)$ is not positive definite, then the iterations are converging to a saddle point and not a minimum.

On the other hand, if a constrained optimization is done (for example, with Lagrange multipliers), the problem may become one of saddle point finding, in which case the Hessian will be symmetric indefinite and the solution of x_{k+1} will need to be done with a method that will work for such, such as the $LDL^\top$ variant of Cholesky factorization or the conjugate residual method.

There also exist various quasi-Newton methods, where an approximation for the Hessian (or its inverse directly) is built up from changes in the gradient.

If the Hessian is close to a non-invertible matrix, the inverted Hessian can be numerically unstable and the solution may diverge. In this case, certain workarounds have been tried in the past, which have varied success with certain problems. One can, for example, modify the Hessian by adding a correction matrix B_k so as to make $f''(x_k) + B_k$ positive definite. One approach is to diagonalize the Hessian and choose B_k so that $f''(x_k) + B_k$ has the same eigenvectors as the Hessian, but with each negative eigenvalue replaced by $\epsilon > 0$.

An approach exploited in the Levenberg–Marquardt algorithm (which uses an approximate Hessian) is to add a scaled identity matrix to the Hessian, μI, with the scale adjusted at every iteration as needed. For large μ and small Hessian, the iterations will behave like gradient descent

with step size $1/\mu$. This results in slower but more reliable convergence where the Hessian doesn't provide useful information.

Some caveats

Newton's method, in its original version, has several caveats:

1. It does not work if the Hessian is not invertible. This is clear from the very definition of Newton's method, which requires taking the inverse of the Hessian.
2. It may not converge at all, but can enter a cycle having more than 1 point. See the Newton's method § Failure analysis.
3. It can converge to a saddle point instead of to a local minimum, see the section "Geometric interpretation" in this article.

The popular modifications of Newton's method, such as quasi-Newton methods or Levenberg-Marquardt algorithm mentioned above, also have caveats:

For example, it is usually required that the cost function is (strongly) convex and the Hessian is globally bounded or Lipschitz continuous, for example this is mentioned in the section "Convergence" in this article. If one looks at the papers by Levenberg and Marquardt in the reference for Levenberg–Marquardt algorithm, which are the original sources for the mentioned method, one can see that there is basically no theoretical analysis in the paper by Levenberg, while the paper by Marquardt only analyses a local situation and does not prove a global convergence result. One can compare with Backtracking line search method for Gradient descent, which has good theoretical guarantee under more general assumptions, and can be implemented and works well in practical large scale problems such as Deep Neural Networks.

See also

- Quasi-Newton method
- Gradient descent
- Gauss–Newton algorithm
- Levenberg–Marquardt algorithm
- Trust region
- Optimization
- Nelder–Mead method
- Self-concordant function - a function for which Newton's method has very good global convergence rate.[2]:Sec.6.2

Notes

1. Nocedal, Jorge; Wright, Stephen J. (2006). *Numerical optimization* (2nd ed.). New York:

Springer. p. 44. ISBN 0387303030.
2. Nemirovsky and Ben-Tal (2023). "Optimization III: Convex Optimization" (http://www2.isye.gatech.edu/~nemirovs/OPTIIILN2023Spring.pdf) (PDF).

References

- Avriel, Mordecai (2003). *Nonlinear Programming: Analysis and Methods*. Dover Publishing. ISBN 0-486-43227-0.
- Bonnans, J. Frédéric; Gilbert, J. Charles; Lemaréchal, Claude; Sagastizábal, Claudia A. (2006). *Numerical optimization: Theoretical and practical aspects*. Universitext (Second revised ed. of translation of 1997 French ed.). Berlin: Springer-Verlag. 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).
- Fletcher, Roger (1987). *Practical Methods of Optimization* (https://archive.org/details/practicalmethods0000flet) (2nd ed.). New York: John Wiley & Sons. ISBN 978-0-471-91547-8.
- Givens, Geof H.; Hoeting, Jennifer A. (2013). *Computational Statistics*. Hoboken, New Jersey: John Wiley & Sons. pp. 24–58. ISBN 978-0-470-53331-4.
- Nocedal, Jorge; Wright, Stephen J. (1999). *Numerical Optimization*. Springer-Verlag. ISBN 0-387-98793-2.
- Kovalev, Dmitry; Mishchenko, Konstantin; Richtárik, Peter (2019). "Stochastic Newton and cubic Newton methods with simple local linear-quadratic rates". arXiv:1912.01597 (https://arxiv.org/abs/1912.01597) [cs.LG (https://arxiv.org/archive/cs.LG)].

External links

- Korenblum, Daniel (Aug 29, 2015). "Newton-Raphson visualization (1D)" (http://bl.ocks.org/dannyko/ffe9653768cb80dfc0da/). *Bl.ocks*. ffe9653768cb80dfc0da.
