Broyden–Fletcher–Goldfarb–Shanno algorithm

From Wikipedia, the free encyclopedia
(Redirected from BFGS method)

In numerical optimization, the **Broyden–Fletcher–Goldfarb–Shanno** (**BFGS**) **algorithm** is an iterative method for solving unconstrained nonlinear optimization problems.

The BFGS method approximates Newton's method, a class of hill-climbing optimization techniques that seeks a stationary point of a (preferably twice continuously differentiable) function. For such problems, a necessary condition for optimality is that the gradient be zero. Newton's method and the BFGS methods are not guaranteed to converge unless the function has a quadratic Taylor expansion near an optimum. These methods use both the first and second derivatives of the function. However, BFGS has proven to have good performance even for non-smooth optimizations.[1]

In quasi-Newton methods, the Hessian matrix of second derivatives doesn't need to be evaluated directly. Instead, the Hessian matrix is approximated using rank-one updates specified by gradient evaluations (or approximate gradient evaluations). Quasi-Newton methods are generalizations of the secant method to find the root of the first derivative for multidimensional problems. In multi-dimensional problems, the secant equation does not specify a unique solution, and quasi-Newton methods differ in how they constrain the solution. The BFGS method is one of the most popular members of this class.[2] Also in common use is L-BFGS, which is a limited-memory version of BFGS that is particularly suited to problems with very large numbers of variables (e.g., >1000). The BFGS-B[3] variant handles simple box constraints.

Contents [hide]

Rationale [edit]

The search direction $\mathbf{p}_k$ at stage k is given by the solution of the analogue of the Newton equation

$$B_k \mathbf{p}_k = -\nabla f(\mathbf{x}_k)$$

where B_k is an approximation to the Hessian matrix which is updated iteratively at each stage, and $\nabla f(\mathbf{x}_k)$ is the gradient of the function evaluated at $\mathbf{x}_k$. A line search in the direction $\mathbf{p}_k$ is then used to find the next point $\mathbf{x}_{k+1}$. Instead of requiring the full Hessian matrix at the point $\mathbf{x}_{k+1}$ to be computed as B_{k+1}, the approximate Hessian at stage k is updated by the addition of two matrices.

$$B_{k+1} = B_k + U_k + V_k$$

Both U_k and V_k are symmetric rank-one matrices but have different (matrix) bases. The symmetric rank one assumption here means that we may write

$$C = \mathbf{a}\mathbf{b}^{\mathrm{T}}$$

So equivalently, U_k and V_k construct a rank-two update matrix which is robust against the scale problem often suffered in the gradient descent searching (*e.g.*, in Broyden's method).

The quasi-Newton condition imposed on this update is

$$B_{k+1}(\mathbf{x}_{k+1} - \mathbf{x}_k) = \nabla f(\mathbf{x}_{k+1}) - \nabla f(\mathbf{x}_k).$$

Algorithm [edit]

From an initial guess $\mathbf{x}_0$ and an approximate Hessian matrix B_0 the following steps are repeated as $\mathbf{x}_k$ converges to the solution.

1. Obtain a direction $\mathbf{p}_k$ by solving: $B_k \mathbf{p}_k = -\nabla f(\mathbf{x}_k)$.
2. Perform a line search to find an acceptable stepsize α_k in the direction found in the first step, then update $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k$.
3. Set $\mathbf{s}_k = \alpha_k \mathbf{p}_k$.
4. $\mathbf{y}_k = \nabla f(\mathbf{x}_{k+1}) - \nabla f(\mathbf{x}_k)$.
5. $B_{k+1} = B_k + \frac{\mathbf{y}_k \mathbf{y}_k^{\mathrm{T}}}{\mathbf{y}_k^{\mathrm{T}} \mathbf{s}_k} - \frac{B_k \mathbf{s}_k \mathbf{s}_k^{\mathrm{T}} B_k}{\mathbf{s}_k^{\mathrm{T}} B_k \mathbf{s}_k}$.

$f(\mathbf{x})$ denotes the objective function to be minimized. Convergence can be checked by observing the norm of the gradient, $|\nabla f(\mathbf{x}_k)|$. Practically, B_0 can be initialized with $B_0 = I$, so that the first step will be equivalent to a gradient descent, but further steps are more and more refined by B_k, the approximation to the Hessian.

The first step of the algorithm is carried out using the inverse of the matrix B_k, which is usually obtained efficiently by applying the Sherman–Morrison formula to the fifth line of the algorithm, giving

$$B_{k+1}^{-1} = \left(I - \frac{s_k y_k^T}{y_k^T s_k}\right) B_k^{-1} \left(I - \frac{y_k s_k^T}{y_k^T s_k}\right) + \frac{s_k s_k^T}{y_k^T \, s_k}.$$

This can be computed efficiently without temporary matrices, recognizing that B_k^{-1} is symmetric, and that $\mathbf{y}_k^{\mathrm{T}} B_k^{-1} \mathbf{y}_k$ and $\mathbf{s}_k^{\mathrm{T}} \mathbf{y}_k$ are scalar, using an expansion such as

$$B_{k+1}^{-1} = B_k^{-1} + \frac{(\mathbf{s}_k^{\mathrm{T}} \mathbf{y}_k + \mathbf{y}_k^{\mathrm{T}} B_k^{-1} \mathbf{y}_k)(\mathbf{s}_k \mathbf{s}_k^{\mathrm{T}})}{(\mathbf{s}_k^{\mathrm{T}} \mathbf{y}_k)^2} - \frac{B_k^{-1} \mathbf{y}_k \mathbf{s}_k^{\mathrm{T}} + \mathbf{s}_k \mathbf{y}_k^{\mathrm{T}} B_k^{-1}}{\mathbf{s}_k^{\mathrm{T}} \mathbf{y}_k}.$$

In statistical estimation problems (such as maximum likelihood or Bayesian inference), credible intervals or confidence intervals for the solution can be estimated from the inverse of the final Hessian matrix. However, these quantities are technically defined by the true Hessian matrix, and the BFGS approximation may not converge to the true Hessian matrix.

Implementations [edit]

The GSL implements BFGS as gsl_multimin_fdfminimizer_vector_bfgs2. Ceres Solver implements both BFGS and L-BFGS. In SciPy, the scipy.optimize.fmin_bfgs function implements BFGS. It is also possible to run BFGS using any of the L-BFGS algorithms by setting the parameter L to a very large number.

Octave uses BFGS with a double-dogleg approximation to the cubic line search.

In the MATLAB Optimization Toolbox, the fminunc function uses BFGS with cubic line search when the problem size is set to "medium scale."

A high-precision arithmetic version of BFGS (pBFGS), implemented in C++ and integrated with the high-precision arithmetic package ARPREC is robust against numerical instability (e.g. round-off errors).

Another C++ implementation of BFGS, along with L-BFGS, L-BFGS-B, CG, and Newton's method) using Eigen (C++ library) are available on github under the MIT License here.

BFGS and L-BFGS are also implemented in C as part of the open-source Gnu Regression, Econometrics and Time-series Library (gretl).

See also [edit]

- Quasi-Newton methods
- Davidon–Fletcher–Powell formula
- L-BFGS
- Gradient descent
- Nelder–Mead method
- Pattern search (optimization)
- BHHH algorithm

Notes [edit]

1. ^ Lewis, Adrian S.; Overton, Michael (2009), "Nonsmooth optimization via BFGS" (PDF), *SIAM J. Optimiz*
2. ^ Nocedal & Wright (2006), page 24
3. ^ Byrd, Richard H.; Lu, Peihuang; Nocedal, Jorge; Zhu, Ciyou (1995), "A Limited Memory Algorithm for Bound Constrained Optimization", *SIAM Journal on Scientific Computing* **16** (5): 1190–1208, doi:10.1137/0916069

Bibliography [edit]

- 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, pp. xiv+490, doi:10.1007/978-3-540-35447-5, ISBN 3-540-35445-X, MR 2265882
- Broyden, C. G. (1970), "The convergence of a class of double-rank minimization algorithms", *Journal of the Institute of Mathematics and Its Applications* **6**: 76–90, doi:10.1093/imamat/6.1.76
- Fletcher, R. (1970), "A New Approach to Variable Metric Algorithms", *Computer Journal* **13** (3): 317–322, doi:10.1093/comjnl/13.3.317
- Fletcher, Roger (1987), *Practical methods of optimization* (2nd ed.), New York: John Wiley & Sons, ISBN 978-0-471-91547-8
- Goldfarb, D. (1970), "A Family of Variable Metric Updates Derived by Variational Means", *Mathematics of Computation* **24** (109): 23–26, doi:10.1090/S0025-5718-1970-0258249-6
- Luenberger, David G.; Ye, Yinyu (2008), *Linear and nonlinear programming*, International Series in Operations Research & Management Science **116** (Third ed.), New York: Springer, pp. xiv+546, ISBN 978-0-387-74502-2, MR 2423726
- Nocedal, Jorge; Wright, Stephen J. (2006), *Numerical Optimization* (2nd ed.), Berlin, New York: Springer-Verlag, ISBN 978-0-387-30303-1
- Shanno, David F. (July 1970), "Conditioning of quasi-Newton methods for function minimization", *Math. Comput.* **24** (111): 647–656, doi:10.1090/S0025-5718-1970-0274029-X, MR 42:8905
- Shanno, David F.; Kettler, Paul C. (July 1970), "Optimal conditioning of quasi-Newton methods", *Math. Comput.* **24** (111): 657–664, doi:10.1090/S0025-5718-1970-0274030-6, MR 42:8906

External links [edit]

- Source code of high-precision BFGS A C++ source code of BFGS with high-precision arithmetic

v · t · e **Optimization: Algorithms, methods, and heuristics** [hide]

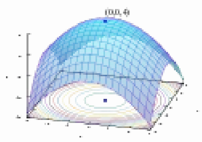
Unconstrained nonlinear: Methods calling ... [show]

Constrained nonlinear [show]

Convex optimization [show]

Combinatorial [show]

Metaheuristics [show]



Categories (Algorithms and methods · Heuristics) · **Software**

Categories: Optimization algorithms and methods