Limited-memory BFGS Method

with Cautious Update and Lewis-Overton Line Search

Zirui Zhang

Cheng Kar-Shun Robotics Institute
The Hong Kong University of Science and Technology

September 22, 2025

Unconstrained Optimization

Consider a smooth and twice-differentiable unconstrained optimization problem:

$$\min_{\mathbf{x}} f(\mathbf{x})$$

Descent methods provide an iterative solution:

$$\mathbf{x}^{k+1} = \mathbf{x}^k + \alpha^k \cdot \mathbf{d}^k$$

where $\mathbf{d}^k$ is the direction, and α^k is the step size.

Newton's Method

By second-order Taylor expansion,

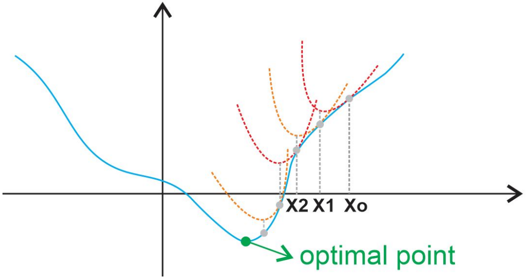
$$f(\mathbf{x}) \approx f(\mathbf{x}^k) + \nabla f(\mathbf{x}^k)^\top (\mathbf{x} - \mathbf{x}^k) + \frac{1}{2}(\mathbf{x} - \mathbf{x}^k)^\top \nabla^2 f(\mathbf{x}^k)(\mathbf{x} - \mathbf{x}^k)$$

Minimizing quadratic approximation,

$$\nabla^2 f(\mathbf{x}^k)(\mathbf{x} - \mathbf{x}^k) + \nabla f(\mathbf{x}^k) = 0$$

For $\nabla^2 f(\mathbf{x}^k) \succ 0$,

$$\mathbf{x}^{k+1} = \mathbf{x}^k - [\nabla^2 f(\mathbf{x}^k)]^{-1} \nabla f(\mathbf{x}^k)$$



Courtesy: Ardian Umam

Damped Newton Method

For $\nabla^2 f(\mathbf{x}^k) \not\succ 0$, the direction $\mathbf{d}^k$ cannot be directly solved from $\nabla^2 f(\mathbf{x}^k)\mathbf{d}^k = -\nabla f(\mathbf{x}^k)$. In such cases, a PD matrix $\mathbf{M}^k$ must be constructed to approximate the Hessian.

If the function is convex, $\nabla^2 f(\mathbf{x}^k)$ may be singular. Adding a regularization term ensures positive definiteness:

$$\mathbf{M}^k = \nabla^2 f(\mathbf{x}^k) + \lambda \mathbf{I}$$

$\lambda > 0$ starts small and grows until **Cholesky decomposition** works.

If the function is nonconvex, $\nabla^2 f(\mathbf{x}^k)$ may be indefinite. To handle this, the **Bunch-Kaufman decomposition** is applied to obtain $\mathbf{L}\tilde{\mathbf{D}}\mathbf{L}^\top$ and $\tilde{\mathbf{D}}$:

$$\mathbf{M}^k = \mathbf{L}\tilde{\mathbf{D}}\mathbf{L}^\top$$

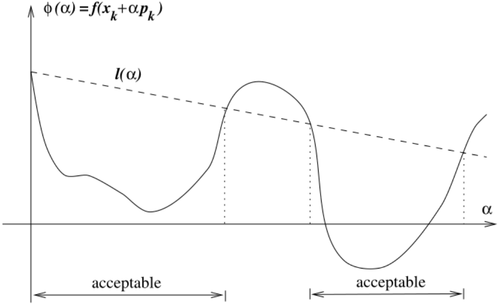
Finally, direction is solved from $\mathbf{M}^k\mathbf{d}^k = -\nabla f(\mathbf{x}^k)$.

Practical Newton Method

Moreover, we can select α^k by backtracking line search to ensure sufficient decrease in the objective function, satisfying the **Armijo condition**:

$$f(\mathbf{x}^k + \alpha^k \mathbf{d}^k) \leq f(\mathbf{x}^k) + c_1 \cdot \alpha^k \nabla f(\mathbf{x}^k)^\top \mathbf{d}^k$$

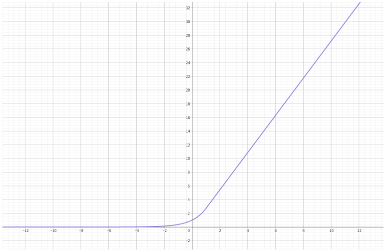
where $c_1 \in (0, 1)$ is a small constant.



Courtesy: Cornell University

Newton's Method: Limitations

- **High Cost**: Computing the Hessian and its inverse requires $\mathcal{O}(n^3)$ operations, impractical for large problems.
- **Indefinite Hessian**: In nonconvex cases, the Hessian may lead to steps toward saddle points.
- **Ill-Conditioning**: Poorly conditioned Hessians amplify errors and hinder convergence.
- **Inaccurate Model**: Local quadratic approximations may fail for complex functions, causing inefficiency or divergence.



Quasi-Newton Approximation

Newton Approximation:

$$f(\mathbf{x}) \approx f(\mathbf{x}^k) + (\mathbf{x} - \mathbf{x}^k)^\top \mathbf{g}^k + \frac{1}{2}(\mathbf{x} - \mathbf{x}^k)^\top \mathbf{H}^k (\mathbf{x} - \mathbf{x}^k)$$

$$\mathbf{H}^k \mathbf{d}^k = -\mathbf{g}^k$$

Quasi-Newton Approximation:

$$f(\mathbf{x}) \approx f(\mathbf{x}^k) + (\mathbf{x} - \mathbf{x}^k)^\top \mathbf{g}^k + \frac{1}{2}(\mathbf{x} - \mathbf{x}^k)^\top \mathbf{B}^k (\mathbf{x} - \mathbf{x}^k)$$

$$\mathbf{B}^k \mathbf{d}^k = -\mathbf{g}^k$$

The matrix $\mathbf{B}^k$ should:

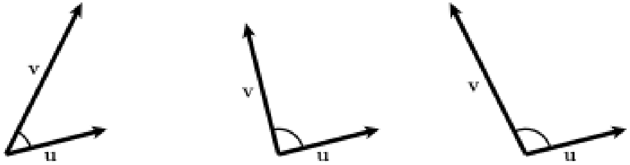
- Avoid full second-order derivatives.
- Have a closed-form solution for linear equations.
- Retain first-order curvature information.
- Preserve the descent direction.

Descent Direction:

Search direction $\mathbf{d}^k$ should make an acute angle with the negative gradient:

$$(\mathbf{g}^k)^\top \mathbf{d}^k = -(\mathbf{g}^k)^\top (\mathbf{B}^k)^{-1} \mathbf{g}^k < 0$$

$\mathbf{B}^k$ must be positive definite (PD) since for all non-negative $\mathbf{g}^k$, $(\mathbf{g}^k)^\top (\mathbf{B}^k)^{-1} \mathbf{g}^k > 0$.



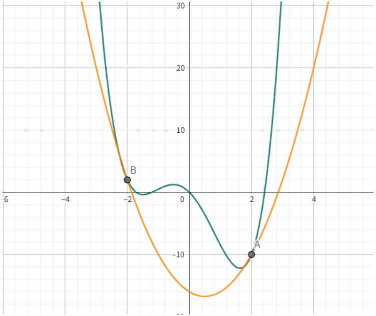
Courtesy: Active Calculus

Curvature Information

At the point $\mathbf{x}^{k+1}$, the gradient is $\mathbf{g}^{k+1}$. We want $\mathbf{B}^{k+1}$ to satisfy:

$$\mathbf{B}^{k+1}(\mathbf{x}^{k+1} - \mathbf{x}^{k}) \approx \mathbf{g}^{k+1} - \mathbf{g}^{k}$$

$$\mathbf{B}^{k+1}\mathbf{s}^{k} = \mathbf{y}^{k}$$



The Optimal $\mathbf{B}^{k+1}$?

Infinitely many $\mathbf{B}^{k+1}$ satisfy the secant condition. To choose the best one, we define the following weighted least square problem:

$$\min_{\mathbf{B}} \|\mathbf{B} - \mathbf{B}^k\|_{\mathbf{W}}^2 \quad \text{subject to} \quad \mathbf{B} = \mathbf{B}^\top, \mathbf{B}\mathbf{s}^k = \mathbf{y}^k$$

In BFGS, the weight matrix is selected as:

$$\mathbf{W} = \int_0^1 \nabla^2 f[(1-\tau)\mathbf{x}^k + \tau\mathbf{x}^{k+1}]\mathrm{d}\tau$$

Closed-form Solution for $\mathbf{B}^{k+1}$

To derive the optimal $\mathbf{B}^{k+1}$, we construct the Lagrangian function:

$$\mathscr{L}(\mathbf{B}, \mathbf{\Lambda}) = \frac{1}{2}\|\mathbf{B} - \mathbf{B}^k\|_{\mathbf{W}}^2 + \operatorname{tr}\left[\mathbf{\Lambda}^\top \left(\mathbf{B}\mathbf{s}^k - \mathbf{y}^k\right)\right]$$

Taking the derivative of the Lagrangian with respect to $\mathbf{B}$ and setting it to zero gives:

$$\frac{\partial \mathscr{L}}{\partial \mathbf{B}} = \mathbf{W}(\mathbf{B} - \mathbf{B}^k)\mathbf{W} + \mathbf{\Lambda}(\mathbf{s}^k)^\top = 0$$

Rearranging the terms, we express $\mathbf{B}$ as:

$$\mathbf{B} = \mathbf{B}^k - \mathbf{W}^{-1}\mathbf{\Lambda}(\mathbf{s}^k)^\top \mathbf{W}^{-1}$$

Closed-form Solution for $\mathbf{B}^{k+1}$ (cont.)

Substituting this expression into the secant condition $\mathbf{B}\mathbf{s}^k = \mathbf{y}^k$, we obtain:

$$\left(\mathbf{B}^k - \mathbf{W}^{-1}\boldsymbol{\Lambda}(\mathbf{s}^k)^\top\right)\mathbf{s}^k = \mathbf{y}^k$$

Solving for $\boldsymbol{\Lambda}$, we find:

$$\boldsymbol{\Lambda} = \mathbf{W}\left(\mathbf{y}^k - \mathbf{B}^k\mathbf{s}^k\right)\left((\mathbf{s}^k)^\top\mathbf{W}^{-1}\mathbf{s}^k\right)^{-1}$$

Finally, substituting $\boldsymbol{\Lambda}$ back, the closed-form solution for $\mathbf{B}^{k+1}$ is:

$$\mathbf{B}^{k+1} = \mathbf{B}^k + \frac{\mathbf{y}^k(\mathbf{y}^k)^\top}{(\mathbf{s}^k)^\top\mathbf{y}^k} - \frac{\mathbf{B}^k\mathbf{s}^k(\mathbf{B}^k\mathbf{s}^k)^\top}{(\mathbf{s}^k)^\top\mathbf{B}^k\mathbf{s}^k}$$

BFGS Update Rules

Given the initial value $\mathbf{B}^0 = \mathbf{I}$, the updates are performed iteratively:

$$\mathbf{B}^{k+1} = \mathbf{B}^k + \frac{\mathbf{y}^k(\mathbf{y}^k)^\top}{(\mathbf{s}^k)^\top\mathbf{y}^k} - \frac{\mathbf{B}^k\mathbf{s}^k(\mathbf{B}^k\mathbf{s}^k)^\top}{(\mathbf{s}^k)^\top\mathbf{B}^k\mathbf{s}^k}$$

where:

$$\mathbf{s}^k = \mathbf{x}^{k+1} - \mathbf{x}^k, \quad \mathbf{y}^k = \mathbf{g}^{k+1} - \mathbf{g}^k$$

For computational efficiency, we often work with the inverse of $\mathbf{B}^k$ directly:

$$\mathbf{C}^{k+1} = \left(I - \frac{\mathbf{s}^k(\mathbf{y}^k)^\top}{(\mathbf{s}^k)^\top\mathbf{y}^k}\right)\mathbf{C}^k\left(I - \frac{\mathbf{y}^k(\mathbf{s}^k)^\top}{(\mathbf{s}^k)^\top\mathbf{y}^k}\right) + \frac{\mathbf{s}^k(\mathbf{s}^k)^\top}{(\mathbf{s}^k)^\top\mathbf{y}^k}$$

Guaranteeing PD of $\mathbf{B}^{k+1}$

To ensure that $\mathbf{B}^{k+1}$ remains positive definite (PD), the following curvature condition must hold:

$$(\mathbf{y}^k)^\top \mathbf{s}^k > 0$$

For any nonzero vector $\mathbf{z}$, using the Cauchy-Schwarz inequality:

$$\begin{aligned}
\mathbf{z}^\top \mathbf{B}^{k+1}\mathbf{z} &= \mathbf{z}^\top \mathbf{B}^k\mathbf{z} + \frac{(\mathbf{z}^\top \mathbf{y}^k)^2}{(\mathbf{y}^k)^\top \mathbf{s}^k} - \frac{(\mathbf{z}^\top \mathbf{B}^k\mathbf{s}^k)^2}{(\mathbf{s}^k)^\top \mathbf{B}^k\mathbf{s}^k} \\
&\geq \frac{\mathbf{z}^\top \mathbf{B}^k\mathbf{z}(\mathbf{s}^k)^\top \mathbf{B}^k\mathbf{s}^k - (\mathbf{z}^\top \mathbf{B}^k\mathbf{s}^k)^2}{(\mathbf{s}^k)^\top \mathbf{B}^k\mathbf{s}^k} \geq 0
\end{aligned}$$

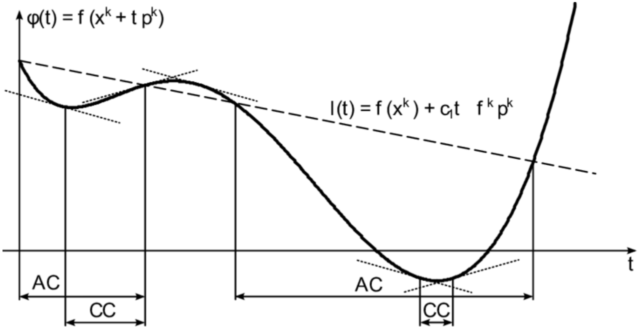
Equalities hold only when $\mathbf{z}^\top \mathbf{y}^k = 0$ and $\mathbf{z} \parallel \mathbf{s}^k$. Given that $(\mathbf{y}^k)^\top \mathbf{s}^k > 0$, these conditions cannot hold simultaneously. Therefore, if $\mathbf{B}^k \succ 0$, it follows that $\mathbf{B}^{k+1} \succ 0$.

Guranteeing $(\mathbf{y}^k)^\top \mathbf{s}^k > 0$

Armijo Condition (AC) cannot gurantee curvature, we need curvature condition (CC):

$$(\mathbf{d}^k)^\top \nabla f(\mathbf{x}^k + \alpha^k \mathbf{d}^k) \geq c_2 \cdot (\mathbf{d}^k)^\top \nabla f(\mathbf{x}^k)$$

Typically, $c_1 = 10^{-4}$, $c_2 = 0.9$.



Courtesy: Ján Kopačka

Lewis-Overton Line Search

The Lewis-Overton line search is a sophisticated backtracking line search designed specifically for quasi-Newton methods:

1. Given search direction $\mathbf{d}^k$, current point $\mathbf{x}^k$ and gradient $\mathbf{g}^k$
2. Initialize trial step $\alpha := 1$, $\alpha_l := 0$, $\alpha_r := +\infty$
3. Repeat
    1. Update bounds:
        - If AC(α) fails, set $\alpha_r := \alpha$
        - Else if CC(α) fails, set $\alpha_l := \alpha$
        - Else, accept α and break
    2. Update α:
        - If $\alpha_r < +\infty$, set $\alpha := (\alpha_l + \alpha_r)/2$
        - Else, set $\alpha := 2 \cdot \alpha_l$
    3. Ensure $\alpha \in [\alpha_{\min}, \alpha_{\max}]$

Cautious Update

Sometimes, when line search is inexact or the function is poorly conditioned, $(\mathbf{y}^k)^\top \mathbf{s}^k > 0$ cannot gurantee. To ensure numerical stability and maintain the PD Hessian approximation, L-BFGS employs a cautious update strategy:

- **Skip Update**: If the curvature condition $(\mathbf{y}^k)^\top \mathbf{s}^k > \epsilon |\mathbf{s}^k|^2$ is not satisfied, where ϵ is a small positive constant (e.g., 10^{-6}), skip the update for this iteration: $\mathbf{B}^{k+1} = \mathbf{B}^k$.
- **Powell's Damping**: If the curvature condition $(\mathbf{y}^k)^\top \mathbf{s}^k \geq \eta (\mathbf{s}^k)^\top \mathbf{B}^k \mathbf{s}^k$ is not satisfied, where η is a small positive constant (e.g., 0.2 or 0.25),

$$\tilde{\mathbf{y}}^k = \theta \mathbf{y}^k + (1-\theta)\mathbf{B}^k \mathbf{s}^k, \qquad \theta = \frac{(1-\eta)\cdot(\mathbf{s}^k)^\top \mathbf{B}^k \mathbf{s}^k}{(\mathbf{s}^k)^\top \mathbf{B}^k \mathbf{s}^k - (\mathbf{y}^k)^\top \mathbf{s}^k}$$

Cautious updates guaranteed to have its iterates converge to a **critical point** if the function has bounded sublevel sets and a Lipschitz continuous gradient.

Two-Loop Recursion

L-BFGS uses a two-loop recursion to compute the search direction without explicitly forming the Hessian approximation. The algorithm maintains a history of the most recent m pairs $(\mathbf{s}^i, \mathbf{y}^i)_{i=k-m}^{k-1}$, where typically m is between 5 and 20.

1. Initialize an empty array $\mathscr{A}$ of length m, $\mathbf{d} = \mathbf{g}^k$
2. For $i = k-1, k-2, \ldots, k-m$:
   1. $\mathscr{A}^{i+m-k} := \langle \mathbf{s}^i, \mathbf{d} \rangle / \langle \mathbf{s}^i, \mathbf{y}^i \rangle$
   2. $\mathbf{d} := \mathbf{d} - \mathscr{A}^{i+m-k} \mathbf{y}^i$
3. $\mathbf{d} := \mathbf{d} \cdot \langle \mathbf{s}^{k-1}, \mathbf{y}^{k-1} \rangle / \langle \mathbf{y}^{k-1}, \mathbf{y}^{k-1} \rangle$
4. For $i = k-m, k-m+1, \ldots, k-1$:
   1. $a := \langle \mathbf{y}^i, \mathbf{d} \rangle / \langle \mathbf{s}^i, \mathbf{y}^i \rangle$
   2. $\mathbf{d} := \mathbf{d} + \mathbf{s}^i (\mathscr{A}^{i+m-k} - a)$
5. Return $\mathbf{d}$

This approach reduces the storage requirement from $\mathscr{O}(n^2)$ to $\mathscr{O}(mn)$ and the computational cost per iteration from $\mathscr{O}(n^2)$ to $\mathscr{O}(mn)$.

Algorithm Summary

The complete L-BFGS algorithm with cautious update and Lewis-Overton line search:

1. Initialize $\mathbf{x}^0$, $\mathbf{g}^0 := \nabla f(\mathbf{x}^0)$, choose m
2. For $k = 0, 1, 2, \ldots$ until convergence:
   1. Compute search direction: $\mathbf{d}^k$ using **L-BFGS two-loop recursion**
   2. Find step size α^k using **Lewis-Overton line search**
   3. Update: $\mathbf{x}^{k+1} = \mathbf{x}^k + \alpha^k \mathbf{d}^k$
   4. Compute $\mathbf{s}^k = \mathbf{x}^{k+1} - \mathbf{x}^k$, $\mathbf{y}^k = \nabla f(\mathbf{x}^{k+1}) - \nabla f(\mathbf{x}^k)$
   5. Apply **cautious update** to $(\{\mathbf{s}^k\}, \{\mathbf{y}^k\})$

Open Source Implementation

- https://github.com/chokkan/liblbfgs
- https://github.com/ZJU-FAST-Lab/LBFGS-Lite
- https://github.com/yixuan/LBFGSpp
- https://github.com/hjmshi/PyTorch-LBFGS

Thank you for listening !

Zirui Zhang