ORIE7391 Weak Wolfe Condition in BFGS Method for Nonsmooth Optimization Problems

Yingxi Li (yl877), Siyu Kong (sk3333)

Cornell University

May 3, 2022

Table of Contents

1 Problem Formulation and Motivation

2 Empirical Results

Overview on Algorithms for Nonsmooth Problems

Consider an unconstrained optimization problem of the form

$$\min_{x \in \mathbb{R}} f(x) \tag{1}$$

where $f(x)$ is not smooth over $\mathbb{R}^n$.
Particularly, steepest descent type of algorithm fails in these scenario due to the discontinuity of gradient around minimization points.
Some commonly applied alternative algorithms includes **bundle method**, **gradient sampling method**, **subgradient method** and also **BFGS method**.

Overview on Algorithms for Nonsmooth Problems

- ‘First order subgradient/gradient type’ algorithms: (i). bundle method; (ii). gradient sampling method [1].
  General philosophy: apply a set of subgradient/gradient information around a point to approximate function values.
  Good and Bad: Both of them have provable theoretical convergence result. However, the rate of convergence is slow.
- ‘Second order Hessian type’ algorithms: BFGS method.
  General philosophy: approximate second order Hessian matrix and apply line search to get iteration point.
  Good and Bad: No theoretical convergence guarantee for nonsmooth functions. However, in practice when the problem converges, the rate is much faster than the previous two methods.

Practical Problems in BFGS Packages

Theoretical Good Performance of BFGS in Smooth Case: Theoretical proof of good convergence result is only given in convex smooth case.
Practical Failure of Applying BFGS: In practice, despite substantial computational experience, the method is lack of theory in nonsmooth case.
Also, people find that BFGS diverges in some non-smooth problems when people apply standard library packages such as `SciPy`, `PyTorch` [2].
Usually, in this case, researchers will then turn to other subgradient method as alternatives.

Conjecture on BFGS Line Search Method

However, traditional BFGS packages including `SciPy` and `PyTorch` impose strong Wolfe conditions and polynomial interpolation in line search method.
In [3], A.Lewis and M.Overton proposed that in non-smooth case, the simpler weak Wolfe condition and bisection method may help speed up the convergence rate of BFGS method.

Recall on Wolfe Condition

In BFGS methods, to solve for

$$\min_{x\in\mathbb{R}^n} f(x) \tag{2}$$

Step One: Estimate inverse Hessian at k-th iteration by matrix B_k
Step Two: Set $x_{k+1} = x_k + \alpha_k p_k = x_k - \alpha_k B_k \nabla f(x_k)$, with step size α_k satisfying Armijo and strong Wolfe conditions:

$$\begin{aligned} f(x_k + \alpha_k p_k) &\le f(x_k) + c_1 \alpha_k p_k^T \nabla f(x_k) && \text{(Armijo condition)} \\ |p_k^T \nabla f(x_k + \alpha_k p_k)| &\le |c_2 p_k^T \nabla f(x_k)| && \text{(strong Wolfe condition)} \end{aligned} \tag{3}$$

Suggestions: Replace strong Wolfe condition by

$$p_k^T \nabla f(x_k + \alpha_k p_k) \ge c_2 p_k^T \nabla f(x_k) \quad \text{(weak Wolfe condition).} \tag{4}$$

Illustration on Failure of Strong Wolfe Conditions

Consider

$$\min_{x \in R^2} f(x) = |x_1| + |x_2|.$$

Given initial point $x_0 = c[1, 1]$ for an arbitrary c, and initial inverse Hessian matrix $H_0 = I_2$. The strong Wolfe condition is written as

$$|p_0^T \nabla f(x_0 + \alpha_0 p_0)| \leq 2c_2 \text{ for some } 0 < c_2 < 1 \text{ and } p_0 = \nabla f(x_0), \tag{5}$$

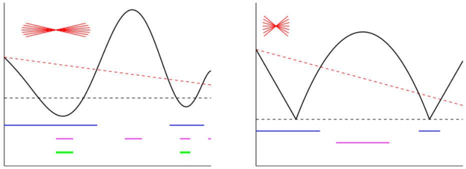
for any α_0, indeed the left hand always equals to 2, therefore we get inequality $2 < 2c_2$, which yields contradiction and will not be solvable by strong Wolfe condition.

Illustration on Failure of Strong Wolfe Conditions

Define $\phi(\alpha) = f(x_k + \alpha p_k)$, then Armijo and strong Wolfe are written as

$$\phi(\alpha) \le \phi(0) + c_1 \alpha \phi'(0), \quad |\phi'(\alpha)| \le |c_2 \phi'(0)|.$$

We plot $\phi(\alpha)$ in the following graph:


(a) *Smooth* $\phi(\alpha)$. (b) *nonsmooth* $\phi(\alpha)$.

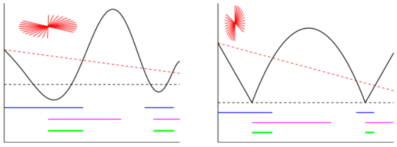
- Red dashed line: Upper bound of $\phi(\alpha)$ from Armijo;
- Red line in left corner: Accepted $\phi'(\alpha)$ from strong Wolfe;
- Blue: Accepted α by Armijo. Magenta: Accepted α by strong Wolfe.

Illustration on Adjustment from weak Wolfe Conditions

Define $\phi(\alpha) = f(x_k + \alpha p_k)$, then Armijo and weak Wolfe are written as

$$\phi(\alpha) \leq \phi(0) + c_1 \alpha \phi'(0), \quad \phi'(\alpha) \leq c_2 \phi'(0).$$

We plot $\phi(\alpha)$ in the following graph:



(a) *Smooth* $\phi(\alpha)$. (b) *nonsmooth* $\phi(\alpha)$.

- Red dashed line: Upper bound of $\phi(\alpha)$ from Armijo;
- Red line in left corner: Accepted $\phi'(\alpha)$ from weak Wolfe;
- Blue: Accepted α by Armijo. Magenta: Accepted α by weak Wolfe.

Inexact Line Search with Bisection Update

Algorithm Inexact line search with bisection update

```
Given iteration point $x$ and direction $p$, set $\alpha \leftarrow 0, \beta \leftarrow +\infty, t \leftarrow 1$, $0 < c_1 < c_2 < 1$
repeat
    if $f(x + \alpha p) > f(x) + c_1 \alpha p^T \nabla f(x)$ (Armijo condition fails):
        $\beta \leftarrow t$
    else if  $p^T \nabla f(x + \alpha p) > c_2 p^T \nabla f(x)$ (weak Wolfe fails)
or $h(t) = f(x + tp)$ is not differentiable at $t$ (*):
        $\alpha \leftarrow t$
    else STOP
    if $\beta < +\infty$, $t \leftarrow (\alpha + \beta)/2$
    else $t \leftarrow 2\alpha$
end
```

Convergence Analysis

Assumptions

$h(t)$ in Alg1 is absolutely continuous on every bounded interval, bounded below and $\limsup_{t\downarrow 0} \dfrac{h(t)-h(0)}{t} < 0$.

Remark: This assumptions is true for a broad class of functions including i.e. locally Lipschitz functions, semialgebraic functions.

Convergence Analysis

Theorem (Convergence of line search Alg1)

(i). When line search iteration terminates, the final trial step t is a weak Wolfe step. In particular, it terminates under the assumption

$$\lim_{t \uparrow \bar{t}} h'(t) \text{ exists in } [-\infty, \infty] \text{ for all } \bar{t} > 0 \tag{6}$$

(ii). If on the other hand the iteration doesn't terminate, then it eventually generates a nested sequence of finite intervals $[\alpha, \beta]$, halving in length at each iteration and each containing a set of nonzero measure of weak Wolfe steps. These intervals converge to a step $t_0 > 0$ such that

$$h(t_0) = c_1 s t_0 \text{ and } \limsup_{t \uparrow t_0} h'(t) \geq c_2 s. \tag{7}$$

Empirical Results: l1 Norm

We start by considering the considering the l_1-norm function

$$f(x) = \|x\|_1. \tag{8}$$

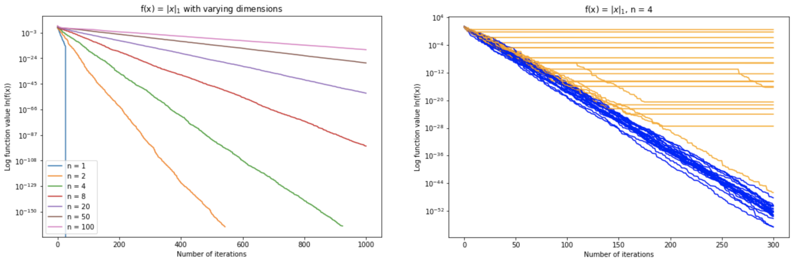


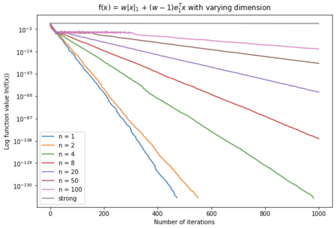
Figure: Plot on the left is BFGS with weak Wolfe on the l1-norm function with varying dimensions. Plot on the right compares BFGS with strong and weak Wolfe conditions in R^4. The convergence of BFGS with strong Wolfe is highly sensitive to initialization

Empirical Results: Tilted l1 Norm

Consider the tilted l_1-norm function

$$f(x) = w\|x\|_1 + (w-1)e_1^T x, \tag{9}$$

where e_1 is the first coordinate vector.



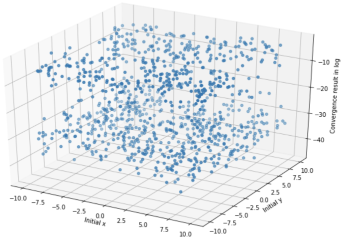


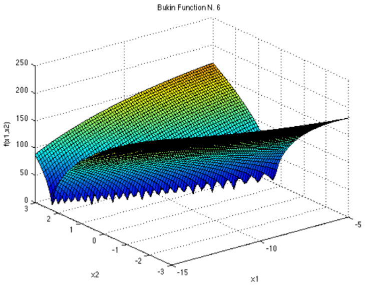
Figure: Plot on the left is BFGS with weak Wolfe on tilted norm function with varying dimensions. Plot on the right explores the relationship between initialization and convergence result on strong Wolfe.

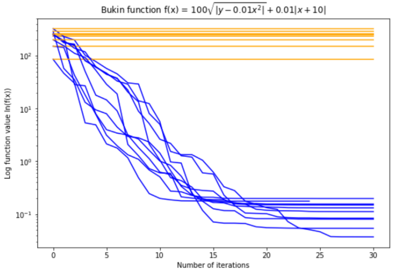
Empirical Results: Bukin Function

Bukin function is of the form

$$f(x_1, x_2) = 100\sqrt{|x_2 - 0.01x_1^2|} + 0.01|x_1 + 10|. \qquad (10)$$

The plot is given as follow, which contains a lot of local minima lying in a ridge.





Empirical Results: Large max iteration of line search

$$f(x) = \|x\|_1 + \|y\|_2^2. \tag{11}$$

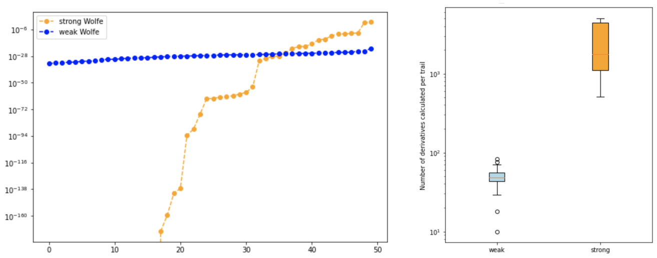


Figure: Plot on the left shows the final convergence result of 50 trails of both strong and weak Wolfe condition when the max iteration of line search is large, and plot on the right displays the number of derivative calculations incurred by line search per trail.

Empirical Results: Tilted l1-norm, n = 500

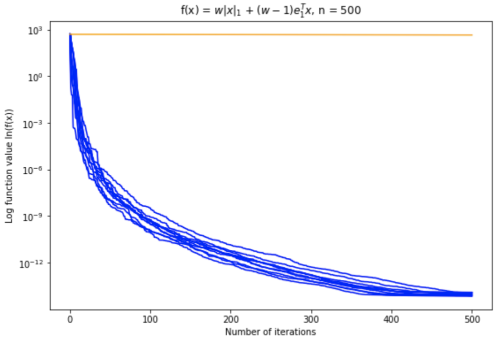


Figure: BFGS with strong and weak Wolfe conditions on the tilted norm function. Superiority of the weak Wolfe condition is demonstrated. When dimension is larger, BFGS has on l1-norm a sub-linear convergence rate.

Empirical Results: Sum of l1-norm and l2-norm squared

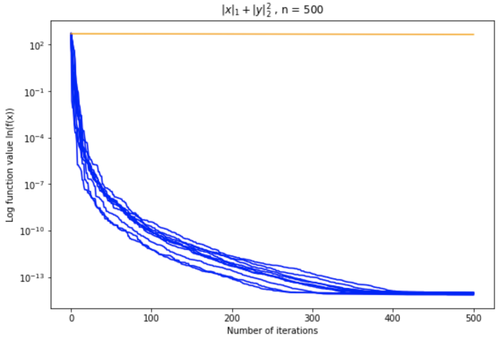


Figure: BFGS with strong and weak Wolfe conditions on function f(x) = $\|x\|_1 + \|y\|_2^2$, with dimension 500.

Empirical Results: LASSO

We explore the LASSO function with a ill-conditioned A (that has spectral decaying eigenvalues).

$$f(x) = ||Ax + b||_2^2 + c||x||_1 \tag{12}$$

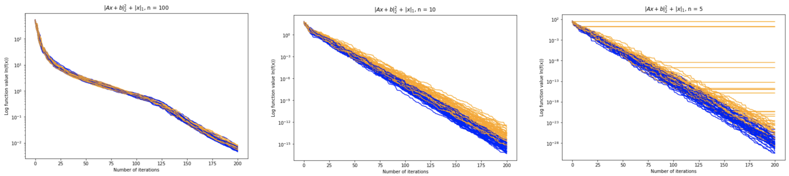
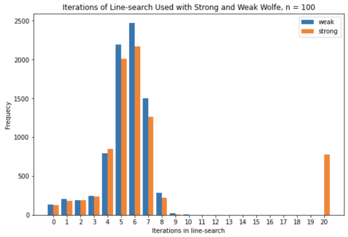


Figure: BFGS on LASSO function with varying dimensions. To our surprise, strong Wolfe performs much stabler in higher dimensions.

Empirical Results: LASSO (continuing)

Other than convergence performances, we also compared the computational cost under weak and strong Wolfe conditions.



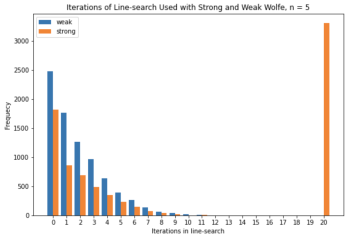


Figure: The number of iterations in line search used to find the step-sizes. Despite the similar convergence results, weak Wolfe condition allows for much lower number of iterations per line search and therefore much cheaper computational cost.

Empirical Results: $f(x) = \sqrt{x^T Ax} + x^T Bx$

We consider another convex partly smooth function.

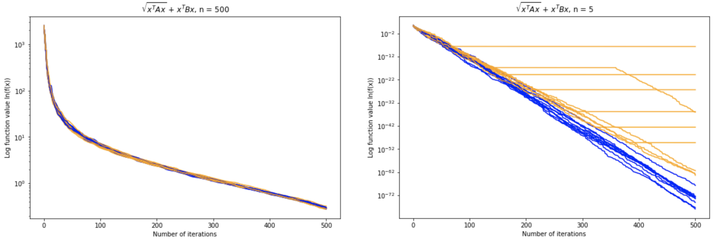
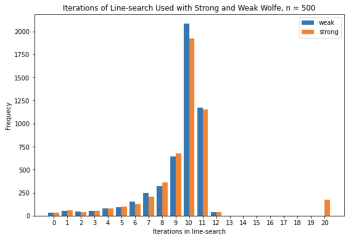


Figure: BFGS on this function with varying dimensions. Again, line search with strong Wolfe performs much stabler in higher dimensions while line search with weak Wolfe retains a consistent good performance.

Empirical Results: $f(x) = \sqrt{x^T A x} + x^T B x$

We consider the cost of line search under this setting:



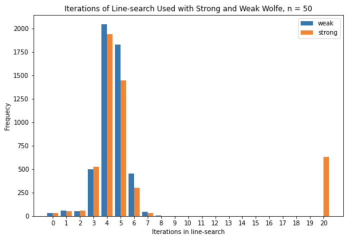


Figure: The number of iterations in line search used to find the step-sizes. Again, weak Wolfe condition allows for lower number of iterations per line search and a cheaper computational cost.

Thank You!

J. V. Burke, A. S. Lewis, and M. L. Overton, “A robust gradient sampling algorithm for nonsmooth, nonconvex optimization,” *SIAM Journal on Optimization*, vol. 15, no. 3, pp. 751–779, 2005. [Online]. Available: https://doi.org/10.1137/030601296

A. Nelson, “Github repository.” [Online]. Available: https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.minimize.html

A. Lewis and M. Overton, “Nonsmooth optimization via quasi-newton methods,” *Mathematical Programming*, vol. 141, no. 1-2, pp. 135–163, Oct. 2013, copyright: Copyright 2013 Elsevier B.V., All rights reserved.