This paper investigates the potential behavior, both good and bad, of the well-known BFGS algorithm for smooth minimization, when applied to nonsmooth functions. We consider three very particular examples. We first present a simple nonsmooth example, illustrating how BFGS (in this case with an exact line search) typically succeeds despite nonsmoothness. We then study, computationally, the behavior of the BFGS method with an inexact line search on the same example, and discuss the results. In further support of the heuristic effectiveness of BFGS despite nonsmoothness, we prove that, for the very simplest class of nonsmooth functions (maximums of two affine functions), the method cannot stall at a nonstationary point. On the other hand, we present a nonsmooth example where the inexact-line-search BFGS method converges to a point despite the presence of directions of linear descent. Finally, we briefly compare line-search and trust-region strategies for BFGS in the nonsmooth case.

Key words: BFGS, nonsmooth, line search, partial smoothness, stationary point, trust region

1 Introduction

We study the behavior of the standard BFGS variable metric method for smooth minimization, when applied to nonsmooth functions. The theory for BFGS applied to convex smooth functions is well established: Powell [10] showed that BFGS with a Wolfe inexact line search converges to a minimizer, when applied to a twice-differentiable convex function with bounded level sets. However, there is no corresponding convergence result for nonconvex smooth functions. Although various authors [9, 7, 6] made some progress by restricting the function class or modifying the method, the convergence theory for the BFGS algorithm on nonconvex functions remains poorly understood. There is even less experience with the BFGS method on nonsmooth functions. The success of variable metric methods on nonsmooth functions was observed many years ago [2], but it seems very challenging to give any rigorous convergence analysis.

Recent work by Lewis and Overton [5] gives a detailed analysis of the BFGS method with an exact line search on one very particular example: the Euclidean norm function in $\mathbb{R}^2$. While very special, the analysis illustrates how BFGS can work well on nonsmooth functions. A companion paper [4] investigates the behavior of BFGS with a suitable inexact line search on some nonsmooth examples: the authors observe that this inexact-line-search BFGS method typically converges to Clarke stationary points, and they pose the following challenge, to prove or disprove.

Consider any locally Lipschitz, semi-algebraic function $f$ with bounded level sets, and choose the initial point $x_0$ and and initial inverse Hessian estimate $H_0$ randomly. With probability one, the BFGS method generates an infinite sequence of iterates, for which any cluster point $\bar{x}$ is Clarke stationary, and furthermore the sequence of all function trial values converges to $f(\bar{x})$ $R$-linearly.

For more precise details on the terminology, see [5, Challenge 7.1].

This paper is largely motivated by these two papers. We highlight further the success of line search BFGS method on some nonsmooth examples, and analyze the potential reasons. By way of contrast, we illustrate the potential bad behavior of the line-search BFGS method by constructing a nonsmooth function on which the method converges to
a point at which there exist directions of linear descent. Our goal, throughout, is simply insight into the line-search BFGS method in the nonsmooth case.

As context, we also briefly discuss the behavior of a trust-region BFGS method when applied to nonsmooth functions. The line search and trust region philosophies for updating the current point of course differ considerably: trust region methods [1] approximate the original problem in a “trust region” by a quadratic subproblem, and take a corresponding step at each iteration. The trust-region BFGS method we discuss for illustration is a simple combination of the trust region method in [8] with the BFGS algorithm in [4]. Our purpose is to understand the fundamental difference between these two different strategies in nonsmooth optimization.

We organize this paper as follows. In Section 2 we demonstrate how the exact-line-search BFGS method succeeds on a representative convex nonsmooth function. We also provide numerical evidence for linear convergence of an inexact-line-search BFGS on the same example. In Section 3 we present an illustrative proof that the inexact-line-search BFGS method cannot stall at a spurious limit point when applied to a representative nonsmooth function without any stationary points. In Section 4 we give an example of how the inexact-line-search BFGS method can converge to a limit point with descent directions. (This example does not disprove the challenge question from [4], since the limit point is nonetheless Clarke stationary.) In Section 5 we discuss possible reasons why the line-search BFGS method seems so much more successful than the trust-region method when applied to nonsmooth functions.

In this paper, we study the BFGS and line search algorithms described in [5] and [4]. The line-search BFGS method applied to minimize a function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ iterates as follows. We use $x_k$, $H_k$, and $p_k$ to denote the current point, the approximate inverse Hessian matrix, and the line search direction at the $k$th iteration. We begin with an initial point $x_0$ and an initial positive semidefinite matrix $H_0$.

**Line-search BFGS method**

repeat

- **Search direction**: $p_k = -H_k \nabla f(x_k)$;

- **Step length**: $x_{k+1} = x_k + \alpha_k p_k$, where $\alpha_k$ satisfies the Wolfe conditions, for fixed $c_1 < c_2$ in $(0, 1)$:

  $$f(x_k + \alpha_k p_k) \leq f(x_k) + c_1 \alpha_k \nabla f(x_k)^T p_k$$  \hspace{1cm} (1)

  $$\nabla f(x_k + \alpha_k p_k)^T p_k \geq c_2 \nabla f(x_k)^T p_k;$$  \hspace{1cm} (2)

- **Gradient increment**: $y_k = \nabla f(x_{k+1}) - \nabla f(x_k)$;

- **Inverse Hessian factor**: $V_k = I - (p_k^T y_k)^{-1} p_k y_k^T$;

- **Inverse Hessian update**: $H_{k+1} = V_k H_k V_k^T + \alpha_k (p_k^T y_k)^{-1} p_k y_k^T$;

- **Iteration count**: $k = k + 1$;

end(repeat)

Notice that the line search employs a one-sided “weak” Wolfe condition, appropriate in the nonsmooth case, and not the more standard “strong” Wolfe condition: for more discussion, see [4, 5]. Well-known elementary properties of the BFGS method include the secant condition

$$s_k := x_{k+1} - x_k = H_{k+1} y_k,$$

and the fact that the matrix $H_k$ remains positive definite. For simplicity, we use the abbreviated notation $\nabla f_k := \nabla f(x_k)$. Notwithstanding the dependence of the standard smooth theory on the assumption $c_1 > 0$, for simplicity of exposition, we take $c_1 = 0$ and $c_2 = 0.9$ throughout this paper.
2 BFGS with exact line search

In this section, we will give a full analysis of the BFGS method with an exact line search, applied to one particular representative nonsmooth example. The exact line search step length is chosen by \( x_k \in \text{argmin}_x \{ f(x_0 + \alpha p_k) \} \). Since we set \( c_1 = 0 \) in this paper, the exact line search step length satisfies the Wolfe conditions. Unlike the algorithm described in [5], the BFGS method we consider here stops whenever it encounters a nonsmooth point.

We begin with a structural property of the exact-line-search BFGS method. For simplicity, we state the result for infinite sequences of iterates.

**Proposition 2.1.** Suppose the BFGS method with exact line search generates a sequence of points \( x_0, x_1, x_2 \ldots \) at which the function \( f \) is smooth and noncritical. Then the following properties hold for all \( k = 1, 2, 3, \ldots \):

1. \( \nabla f^T s_{k-1} = 0 \)
2. \( y^T_{k-1} s_k = 0 \)
3. \( \nabla f^T s_k < 0 \).

Conversely, suppose that \( f \) is a convex function on \( \mathbb{R}^2 \), that \( \nabla f^T s_0 < 0 \), and that properties 1 and 2 hold for all \( k = 1, 2, 3, \ldots \). Then \( x_0, x_1, x_2 \ldots \) is an exact-line-search BFGS sequence.

**Proof.** Property 1 follows immediately from the definition of the exact line search. To see property 2, note
\[
y^T_{k-1} s_k = \alpha y^T_{k-1} p_k = -\alpha y^T_{k-1} H_k \nabla f_k = -\alpha s^T_{k-1} \nabla f_k = 0,
\]
using the secant condition and property 1. Property 3 follows from the fact that \( H_k \) is positive definite.

We prove the converse by induction. Since \( \nabla f(x_0)^T s_0 < 0 \), we can find a positive definite matrix \( H_0 \) such that
\[
p_0 = s_0 = -H_0 \nabla f_0.
\]
The exact line search then seeks \( a_0 \) minimizing \( f(x_0 + ap_0) \). Since \( f \) is convex and \( \nabla f^T p_0 = 0 \), it follows that \( a_0 = 1 \) is a minimizer, and hence \( x_1 \) is an exact-line-search BFGS iterate.

Now consider an exact-line-search BFGS sequence \( x_0, x_1, x_2 \ldots , x_k \). By the secant condition and property 1, we know
\[
p^T_k y_{k-1} = -\nabla f^T_k H_k y_{k-1} = -\nabla f^T s_{k-1} = 0.
\]
Since we are working on the space \( \mathbb{R}^2 \) and property 2 holds, the lines \( R p_k \) and \( R s_k \) must coincide, both being orthogonal to the nonzero vector \( y_{k-1} \). Now the conditions that \( f \) is convex and \( \nabla f^T s_k = 0 \) imply that \( \alpha = \alpha_k \) minimizes the function \( f(x_k + \alpha p_k) \) in the line search. Hence \( x_{k+1} \) is a valid next iterate for the method. The proposition follows.

**A parametrized example**

We next consider a simple but illustrative nonsmooth example on \( \mathbb{R}^2 \). This function has a global minimizer at zero and is nonsmooth at every point on one axis (and indeed is “partly smooth” [3] relative to that axis). If we initialize appropriately, the algorithm will generate a sequence of points alternating between two parabolas and converging linearly to the optimal solution. In all other cases, the algorithm will terminate at a nonsmooth point after finitely many iterations.

**Proposition 2.2.** Consider the exact-line-search BFGS method, applied to minimize the function
\[
f(u, v) = \max\{u^2 + v, u^2 - av\}
\]
for some fixed parameter \( a > 0 \), initialized with
\[
(u_0, v_0) = \left(1, \frac{2}{a^2 + 3a + 1}\right) \quad \text{and} \quad H_0 = \begin{pmatrix} a/(a+1)^2 & 0 \\ 0 & 2/(a+1)^2 \end{pmatrix}.
\]
The iterates converge linearly to the unique global minimizer zero, with rate \( \rho = \frac{a}{(a+1)^2} \), and oscillate between the two parabolas
\[
v = \frac{2}{a^2 + 3a + 1} u^2 \quad \text{and} \quad v = -\frac{2a}{a^2 + 3a + 1} u^2.
\]
Explicitly, the iterates are given by
\[
(u_{2k}, v_{2k}) = \left( \rho^k, \frac{2 \rho^{2k}}{a^2 + 3a + 1} \right), \quad (u_{2k+1}, v_{2k+1}) = \left( \frac{\rho^k}{a + 1}, -\frac{2 \rho^{2k+1}}{a^2 + 3a + 1} \right).
\]
Moreover, the corresponding inverse Hessian approximations are, for \( k > 0 \),
\[
\begin{align*}
H_1 &= \frac{1}{2(a^2 + a + 1)} \begin{pmatrix} 2a^2 + a & 2a(1-a)(1+a)^{-1} \\ 2a(1-a)(1+a)^{-1} & 4(a^3 + a + 1)(1+a)^{-3} \end{pmatrix}, \\
H_{2k} &= \frac{1}{2a^2(2a^2 + a + 1)} \begin{pmatrix} a^2(a^2 + 2a + 2) & 2a \rho^k \\ 2a \rho^k & 4(a^2 + 1) \rho^{2k} \end{pmatrix}, \\
H_{2k+1} &= \frac{1}{2(1+a)^2(a^2 + a + 1)} \begin{pmatrix} (1+a)^2(2a^2 + 2a + 1) & -2a^2(1+a) \rho^k \\ -2a^2(1+a) \rho^k & 4(a^2 + 1) \rho^{2k} \end{pmatrix}.
\end{align*}
\]
The step sizes are given, for \( k > 0 \), by
\[
\alpha_0 = 1, \quad \alpha_1 = \frac{1}{a(1+a)}, \quad \alpha_{2k} = a \rho, \quad \alpha_{2k+1} = \frac{\rho^k}{a}.
\]

**Proof.** A simple calculation verifies that the given sequence of iterates is indeed an exact-line-search BFGS sequence, by Proposition 2.1. Furthermore, since the function is strictly convex along each search direction, the given sequence is the unique exact-line-search BFGS sequence under the given initialization. The formulae for the inverse Hessian approximations are easy to verify directly by induction: see the Appendix. \( \square \)

Notice that the convergence rate \( \rho \) is unchanged under the transformation \( a \leftarrow \frac{1}{a} \). This is not surprising, given the invariance of the method under scaling of the objective, and a consequent simple symmetry property.

In the example above, for very specific initial values, BFGS generates a sequence of points oscillating between two parabolas and converging linearly to the optimal solution, zero. We also observe, at each iteration, that the method crosses the axis on which the function is nonsmooth. Seemingly this property allows BFGS to “learn” the nonsmooth structure of the problem, coded into the inverse Hessian approximations. By contrast, as we see next, under general initial conditions, unless all the iterates except for the initial point lie on the two parabolas, the exact line search causes the simple nonsmooth BFGS method we consider here to halt at a nonsmooth point that is not optimal.

**Proposition 2.3.** Consider the exact-line-search BFGS method applied to minimize the function
\[
f(u, v) = \max\{a^2 + v, u^2 - av\}
\]
for some fixed parameter \( a > 0 \). Unless the first two iterates \( (u_0, v_0) \) and \( (u_1, v_1) \) satisfy the conditions \( u_1 = (1+a)^{-1}u_0 \) and
\[
v_1 = -\frac{2a}{a^2 + 3a + 1} u_1^2 \text{ or } v_1 = \frac{2}{a^2 + 3a + 1} u_1^2,
\]
the algorithm will stop at a nonsmooth point after finitely many iterations.

**Proof.** For simplicity, we focus on the case \( a = 1 \). Assume the method generates an infinite sequence of smooth points \( x_k = (u_k, v_k)^T \) for \( k = 1, 2, 3, \ldots \). We first claim that the coordinate \( v_k \) must change sign at every iteration. If not, then without loss of generality there exists an iteration \( n \) such that \( v_{n-1} < 0 \) and \( v_n < 0 \). The previous result ensures
\[
(\nabla f_n - \nabla f_{n-1})^T (x_n - x_{n-1}) = 0,
\]
so the search direction \( p_n \) must be in the direction of the vector \( (0, 1)^T \). But the exact line search then causes termination at the nonsmooth point \( x_{n+1} = (u_n, 0)^T \), contradicting our assumption.

Without loss of generality, we can next assume \( v_{2k} > 0 \), \( v_{2k+1} < 0 \) for all \( k = 1, 2, 3, \ldots \). By applying the previous result, we easily arrive at the recursion
\[
\begin{align*}
\frac{\rho_{2k+1} - \rho_{2k-1}}{2} &= -\frac{u_{2k} - u_{2k-1}}{2} \\
v_{2k+1} &= v_{2k} + \frac{(u_{2k} - u_{2k-1})(3u_{2k} - u_{2k-1})}{2},
\end{align*}
\]
and similarly

\[ u_{2k} = \frac{-u_{2k-1} - u_{2k-2}}{2}, \]
\[ v_{2k} = v_{2k-1} + \frac{(u_{2k-1} - u_{2k-2})(-3u_{2k-1} + u_{2k-2})}{2}. \]

Hence we deduce

\[ u_n = -\frac{u_{n-1} - u_{n-2}}{2} \]

for all iterates \( n > 2 \), and consequently

\[ u_n + u_{n-1} = \frac{u_{n-1} + u_{n-2}}{2}. \]

Induction implies

\[ u_n + u_{n-1} = \frac{u_1 + u_0}{2n-1}. \]

We deduce, for \( k = 1, 2, 3, \ldots \),

\[ u_{2k} = \frac{2}{3}\left(1 - \frac{1}{2^k}\right)(u_1 + u_0) + u_0, \]
\[ u_{2k+1} = \frac{2}{3}\left(\frac{1}{2^{k+1}} + 1\right)(u_1 + u_0) - u_0. \]

In particular, \( u_{2k} \to \lambda := \frac{1}{2}u_0 - \frac{2}{7}u_1 \) and \( u_{2k+1} \to -\lambda \) as \( k \to \infty \).

Now assume \( \frac{u_0}{u_1} \neq \frac{1}{2} \), so \( \lambda \neq 0 \). Suppose first that \( \lambda > 0 \). (The case \( \lambda < 0 \) is similar.) Then for all large \( k \) we must have \( u_{2k-1} < 0 \) and \( u_{2k} > 0 \). By the previous result we know

\[ (\nabla f_k - \nabla f_{k-1})^T(x_{2k+1} - x_{2k}) = 0, \]

so the search direction \( p_{2k} \) is in the direction \((-1, \mu)^T\) where \( \mu = u_{2k} - u_{2k-1} \). By definition of the exact line search, we know \( x_{2k+1} = x_{2k} + \beta(-1, \mu)^T \) where the scalar \( \beta \) minimizes

\[ (u_{2k} - \beta)^2 + |v_{2k} + \beta\mu|. \]

If \( v_{2k} + \beta\mu > 0 \), then either \( v_{2k+1} = 0 \) or \( v_{2k+1} > 0 \). In the case \( v_{2k+1} = 0 \), our method stops at this nonsmooth point. If, on the other hand, \( v_{2k+1} > 0 \), then the same argument shows \( v_{2k+2} = 0 \).

Suppose, on the other hand, \( v_{2k} + \beta\mu < 0 \). Then, by its definition, \( \beta \) minimizes

\[ (u_{2k} - \beta)^2 - (v_{2k} + \beta\mu), \]

so a quick calculation shows

\[ \beta = \frac{3u_{2k} - u_{2k-1}}{4}. \]

Since \( u_{2k} > 0 \) and \( u_{2k-1} < 0 \), we deduce \( \beta > 0 \). We also know \( v_{2k} > 0 \) and \( \mu > 0 \), so we deduce

\[ v_{2k+1} = v_{2k} + \beta\mu > 0, \]

which contradicts the property \( v_{2k+1} < 0 \).

Now consider the final case, where \( \frac{u_0}{u_1} = \frac{1}{2} \) but \( v_1 \neq -\frac{2}{3}u_1^2 \). Then the formula above for the component \( u_n \) reduces to

\[ u_n = \frac{u_0}{2n}, \]

and we can similarly deduce a formula for the component \( v_n \):

\[ v_{2k+1} = v_{2k} + \frac{(u_{2k} - u_{2k-1})(3u_{2k} - u_{2k-1})}{2} = v_{2k} - \frac{u_0^2}{24k+1}, \]

and

\[ v_{2k} = v_{2k-1} + \frac{(u_{2k-1} - u_{2k-2})(3u_{2k-1} - u_{2k-2})}{2} = v_{2k-1} + \frac{u_0^2}{24k-1}. \]

Hence \( v_n \) converges to \( v_1 + \frac{2}{3}u_1^2 \), which, by assumption, is nonzero. Thus \( v_n \) eventually does not change sign, which quickly gives a contradiction. \( \square \)
Corollary 2.4. Consider the exact-line-search BFGS method applied to minimize the function
\[ f(u, v) = \max \{u^2 + v, u^2 - av\} \]
for some fixed parameter \(a > 0\). Suppose the method generates an infinite sequence of smooth points \(x_0, x_1, x_2, \ldots\). Then that sequence must oscillate between the following two parabolas
\[ v = \frac{2}{a^2 + 3a + 1} u^2 \quad \text{and} \quad v = -\frac{2a}{a^2 + 3a + 1} u^2, \]
converging linearly to the global minimizer zero.

Proof. Again we concentrate on the case \(a = 1\) for simplicity. By the previous result, we can assume \(v_{2k} > 0, v_{2k+1} < 0, u_0 = 2u_1,\) and \(v_1 = -\frac{2}{5} u_1^2\). Then we deduce the formulae
\[ u_{2k} = \frac{u_0}{2^{2k}}, \quad v_{2k} = \frac{2u_0^2}{5 \cdot 4^{2k}}, \quad u_{2k+1} = \frac{u_0}{2^{2k+1}}, \quad v_{2k+1} = -\frac{2u_0^2}{5 \cdot 4^{2k+2}}, \]
so result follows. □

To summarize this very simple theoretical case study, we observe two possible cases. Either the exact-line-search BFGS method converges linearly to the global minimizer zero, oscillating between two parabolas, or the line search causes the method to terminate prematurely at a nonoptimal nonsmooth point.

3 BFGS with inexact line search

We turn next from the idealized version of BFGS of the previous section to a more realistic version with an inexact line search. Again we focus on very simple examples, seeking insight on the method in the nonsmooth case, rather than extensive practical experience.

Recall that, for minimizing the function \(f\), with the current iterate \(x_k\) and search direction \(p_k\), the line search seeks a step length \(\alpha_k\): that is, a scalar \(t\) satisfying the two Wolfe conditions:
\[
\begin{align*}
    f(x_k + tp_k) &\leq f(x_k) + c_1 t \nabla f(x_k)^T p_k \quad (3) \\
    \nabla f(x_k + tp_k)^T p_k &\geq c_2 \nabla f(x_k)^T p_k. \quad (4)
\end{align*}
\]

We here use the following algorithm [4, Alg. 2.6] to find a step length.

Algorithm 3.1. (Inexact line search)
\[
\begin{align*}
    \alpha &\leftarrow 0; \\
    \beta &\leftarrow +\infty; \\
    t &\leftarrow 1; \\
    \text{repeat} & \\
    \quad \text{if inequality (3) fails, } \beta \leftarrow t \\
    \quad \text{else if inequality (4) fails, } \alpha \leftarrow t \\
    \quad \text{else stop}; \\
    \quad \text{if } \beta < +\infty, \ t \leftarrow (\alpha + \beta)/2 \\
    \quad \text{else } t \leftarrow 2\alpha; \\
    \text{end(repeat)}
\end{align*}
\]

As we saw in the previous section, for nonsmooth examples, the behavior of BFGS with an exact line search can depend heavily on the initialization. By contrast, the behavior with an inexact line search in practice seems more robust. We consider here the question of to what extent we can gain insight on the convergence rate (with respect to the number of function evaluations) when using an inexact line search from the behavior with an exact line search.
The parametrized example: ill conditioning

For illustration, we return to the previous example

\[ f(u, v) = u^2 + \max\{u, -au\}. \]  

(5)

We are particularly interested in the broad dependence of the rate of convergence on the parameter \( a \), which gives a certain measure of the “conditioning” of the problem. Using the inexact line search above with \( c_1 = 0 \) and \( c_2 = 0.9 \), numerical experimentation shows that, with random initialization, inexact-line-search BFGS eventually crosses the \( v \)–axis at each iteration, and has a linear convergence rate, plotted in red on the figure. (This behavior is relatively insensitive to the choice of \( c_2 \).)

A reasonable fit to the observed linear convergence rate is given by the function \( r(a) \), defined for \( 0 < a < 1 \) by

\[ \log_2(r(a)) = \frac{\log_2(3a^2 + 3a + 1) - \log_2((a^2 + 3a + 3)(a + 1)^2)}{2 \log_2(1 + \frac{1}{a})}, \]  

and for \( a > 1 \) by the obvious symmetry \( r(a) = r(1/a) \). This function is plotted in blue on the figure. We arrive at this rough fit through the following loose intuition.

As Proposition 2.2 indicates, when applying exact-line-search BFGS to this function with appropriate starting points, we generate the iterates

\[ x_{2k} = \left( \rho^k, \frac{2\rho^{2k}}{a^2 + 3a + 1} \right) \quad \text{and} \quad x_{2k+1} = \left( \frac{\rho^k}{a + 1}, -\frac{2\rho^{2k+1}}{a^2 + 3a + 1} \right), \]  

(7)

with step lengths

\[ \alpha_{2k} = \left( 1 + \frac{1}{a} \right)^2 \quad \text{and} \quad \alpha_{2k+1} = (1 + a)^{-2}. \]

The linear convergence rate per iteration is, in this case,

\[ \frac{f(x_{2k+1})}{f(x_{2k})} = \frac{1 + \frac{2a^2}{a^2 + 3a + 1}}{1 + \frac{2}{a^2 + 3a + 1}} \quad \text{and} \quad \frac{f(x_{2k+2})}{f(x_{2k+1})} = \frac{1 + \frac{2a^2}{a^2 + 3a + 1}}{1 + \frac{2a}{a^2 + 3a + 1}}. \]

Consider the case when \( a > 0 \) is small. In that case, the odd iterations generate a large decrease in function value with a step length close to one. By contrast, the even iterations generate only a small decrease, and to do so need to use a small step length \( (1 + 1/a)^{-2} \). We might expect a bisection-based line search to need roughly \( \log_2((1 + 1/a)^2) \) function evaluations to locate the step. Hence an estimate of the convergence rate during those iterations is

\[ r(a) = \left( \frac{f(x_{2k+1})}{f(x_{2k})} \right)^{\frac{1}{\log_2((1 + 1/a)^2)}}. \]  

(8)
Numerical experiments with inexact-line-search BFGS suggest that iterations analogous to the one above are in fact typical. Hence we arrive at the estimate (8), which does indeed give a reasonable fit to the experimental data. A similar argument applies to large $a > 0$.

We can explore this behavior in a more controlled fashion. Consider, for the moment, the behavior of our inexact-line search when started at the exact-line-search iterates $x_{2k}$ (or $x_{2k+1}$) described by (7) and searching in the corresponding directions $p_{2k}$ and $p_{2k+1}$. Numerical results (with $c_0 = 0, c_1 = 0.9$) suggest that the number of function evaluations needed by the line search only depends on $a$ and doesn’t depend on the iteration count $k$. The following lemma throws some light on that dependence.

**Lemma 3.2.** Consider the function (5) with $a = 2^m - 1$ (for $m = 1, 2, 3, \ldots$), and the exact-line-search BFGS iterates (7) and corresponding search directions $p_{2k}$ and $p_{2k+1}$. With those iterates and search directions, the inexact line search would generate the step lengths $\alpha_{2k} = 1$ and $\alpha_{2k+1} = 2^{-m}$. On the other hand, in the case $a = 1/(2^m - 1)$, we generate $\alpha_{2k} = 2^{-m}$ and $\alpha_{2k+1} = 1$.

**Proof.** We only prove the case when $a > 1$. The proof for $a < 1$ is similar.

For the even iterations, since $x_{2k} = (\rho^k, \frac{2\rho^k - 1}{a^2 + 3a + 1})$ and $p_{2k} = (-\frac{(a+1)a^2}{a}, -\frac{2\rho^k}{a^2 + 3a + 1})$, we deduce
\[
\begin{align*}
    f(x_{2k}) &= \left(1 + \frac{2}{a^2 + 3a + 1}\right)\rho^{2k}, \\
nabla f(x_{2k})^T p_{2k} &= a (a + 1) - 2a \rho^{2k},
\end{align*}
\]
then we obtain
\[
\begin{align*}
x_{2k+1} &= (\frac{1}{a + 1} - a)\rho^k, \\
nabla f(x_{2k+1})^T p_{2k+1} &= -\left(\frac{2}{a + 1} + 2a\right)\rho^{2k+1}.
\end{align*}
\]

Consider the case $a = 2^{-l}$ for some integer $l = 1, 2, 3, \ldots, m - 1$. We have
\[
\begin{align*}
    \nabla f(x_{2k+1} + \alpha p_{2k+1})^T p_{2k+1} &= -\left(\frac{2}{a + 1} - 2a\right)\rho^{2k} + 2\rho^{2k+1} > 0.
\end{align*}
\]
Notice $\alpha \geq \frac{2}{a + 1}$, so
\[
\begin{align*}
f(x_{2k+1} + \alpha p_{2k+1}) &= \left(\alpha - \frac{1}{a + 1}\right)\rho^{2k} + \left(2\alpha - \frac{2}{a^2 + 3a + 1}\right)\rho^{2k+1} \\
&\geq \left(\frac{2}{a + 1} - \frac{1}{a + 1}\right)\rho^{2k} + \left(2\alpha - \frac{2}{a^2 + 3a + 1}\right)\rho^{2k+1} \\
&> \frac{1}{(a + 1)^2}\rho^{2k} + \frac{2a}{a^2 + 3a + 1}\rho^{2k+1} = f(x_{2k+1}).
\end{align*}
\]
Therefore, the line search algorithm will successively try $\alpha = 1, \frac{1}{2}, \cdots, \frac{1}{2^{m-1}}$, and finally $\alpha = \frac{1}{2^m - 1}$. At this point we have
\[
\begin{align*}
    \nabla f(x_{2k+1} + \alpha p_{2k+1})^T p_{2k+1} &= -\left(\frac{2}{a + 1} - 2a\right)\rho^{2k} + 2\rho^{2k+1} > 0,
\end{align*}
\]
and
\[
\begin{align*}
f(x_{2k+1} + \alpha p_{2k+1}) &= \left(\alpha - \frac{1}{a + 1}\right)\rho^{2k} + \left(2\alpha - \frac{2}{a^2 + 3a + 1}\right)\rho^{2k+1} \\
&\leq \frac{2a}{(a + 1)^2}\rho^{2k+1} \leq f(x_{2k+1}),
\end{align*}
\]
so the Wolfe conditions are satisfied. The result follows. $\square$
The result above suggests that, if we were following the iterates generated by exact-line-search BFGS, then, for small $a > 0$, the “work” involved in each iteration, measured loosely by the number of function evaluations our inexact line search would take, is dominated by the even iterations (which achieve a reduction factor of $f(x_{2k+1})/f(x_{2k})$), for which it is approximately $\log_2 (1 + \frac{1}{a})$. Again we see the ingredients of the estimate (8).

**Example: a ridge**

Rigorous analysis of inexact-line-search BFGS in the nonsmooth case seems very challenging in general. Here, for reassurance, we prove one very modest result. In the simplest possible case — a maximum of two affine functions (a “ridge”) — we can at least be sure that the method will not converge to a spurious limit. More precisely, we have the following result.

**Proposition 3.3.** If the inexact-line-search BFGS method applied to the function $f(u, v) = |u| + v$ generates a sequence of iterates $x_k = (u_k, v_k)^T$ with $u_k \neq 0$ (for $k = 0, 1, 2, \ldots$), then $x_k$ does not converge.

**Proof.** The Wolfe conditions hold at each iteration. Hence, if the current point satisfies $u_k > 0$, then the search direction $p_k = (m_k, l_k)^T$ satisfies $m_k < 0$, and at the next iteration we must have $u_{k+1} < 0$. A similar argument holds if $u_k < 0$.

We first prove $|m_k| > |l_k|$ for all iterations $k$. Without loss of generality, suppose $u_k > 0$. Since $\nabla f(x_k) = (1, 1)^T$, then

$$m_k + l_k = -(1, 1)H_k \begin{pmatrix} 1 \\ 1 \end{pmatrix} < 0.$$ Notice that $y_k = \nabla f(x_{k+1}) - \nabla f(x_k) = (-2, 0)^T$ and

$$V_k = I - (p_k^T y_k)^{-1} p_k y_k = \begin{pmatrix} 0 & 0 \\ \frac{-l_k}{m_k} & 1 \end{pmatrix}.$$ Hence

$$H_k = \begin{pmatrix} a_k & b_k \\ b_k & c_k \end{pmatrix} \Rightarrow H_k + 1 = V_k H_k V_k^T + a_k (p_k^T y_k)^{-1} p_k y_k = \begin{pmatrix} \frac{-a_k m_k}{2} & -\frac{a_k l_k}{2} \\ -\frac{a_k l_k}{2} & \frac{1}{m_k} c_k \end{pmatrix}.$$ Hence, $m_{k+1} = \frac{a_k (l_k - m_k)}{2} > 0$, which implies $l_k - m_k > 0$. Combined with the fact that $m_k + l_k < 0$, we have $|m_k| > |l_k|$.

We now prove the proposition by contradiction. Suppose the sequence $x_k$ converges. Then $\alpha_k m_k \to 0$ and $\alpha_k l_k \to 0$.

Note

$$p_{k+1} = -H_k \nabla f(x_{k+1}) = \begin{pmatrix} \frac{-a_k m_k}{2} & -\frac{a_k l_k}{2} \\ -\frac{a_k l_k}{2} & \frac{1}{m_k} c_k \end{pmatrix} \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \begin{pmatrix} \frac{a_k (l_k - m_k)}{2} \\ \frac{a_k (l_k - m_k)}{2} - c_{k+1} \end{pmatrix},$$

where $c_{k+1} = \frac{a_k (l_k - m_k)}{2} - 2b_k \frac{m_k}{m_k} + c_k - \frac{a_k (l_k - m_k)}{2 m_k}$. We deduce $m_{k+1} = \frac{a_k (l_k - m_k)}{2} \to 0$. We now show that the positive number $c_{k+1}$ stays bounded away from zero.

To this end, note by induction we have

$$H_{k+1} = V_k \cdots V_0 H_0 V_0^T \cdots V_k^T + \alpha_0 V_k \cdots V_1 (p_0^T y_0)^{-1} p_0 p_0^T V_1^T \cdots V_k^T.$$ Since

$$V_i V_j = \begin{pmatrix} 0 & 0 \\ \frac{l_i}{m_i} & 1 \end{pmatrix} \begin{pmatrix} 0 & 0 \\ \frac{l_j}{m_j} & 1 \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ \frac{l_i}{m_i} & 1 \end{pmatrix} = V_j,$$

we simplify to

$$H_{k+1} = V_0 H_0 V_0^T + \alpha_0 V_1 (p_0^T y_0)^{-1} p_0 p_0^T V_1^T + \cdots + \alpha_k (p_k^T y_k)^{-1} p_k p_k^T.$$ It is easy to see that $p_k^T y_k > 0$ for all $k$. Hence the $(2, 2)$-entry of the matrix $H_k$ is increasing in $k$, and hence at least as large as the corresponding entry in the matrix $V_0 H_0 V_0^T$, namely

$$a_0 (\frac{l_0}{m_0})^2 - 2b_0 \frac{l_0}{m_0} + c_0 > 0,$$ as required.

Finally, observe $l_{k+1} = \frac{-a_k l_k}{2} - c_{k+1}$ cannot converge to zero, which contradicts the fact $|m_k| > |l_k|$. The result follows. □
The idea of this proof extends to any maximum of two affine functions on $\mathbb{R}^n$. Note too how this example illustrates behavior that seems to drive the success of BFGS in the nonsmooth case: the inexact line search crosses the $u = 0$ axis (the manifold with respect to which the function is partly smooth) at each iteration, allowing the method to “learn” the nonsmooth structure.

4 A limit point with descent directions

In the above sections we illustrated good behavior of BFGS on some nonsmooth functions. In this section, we contrast with an illustration of some bad behavior.

The reference [4] conjectures that the inexact-line-search BFGS method converges to points that are Clarke stationary: for Lipschitz functions, this amounts to saying that we can find convex combinations of gradients at nearby points that are arbitrarily small. For a large class of functions (for example, those of the form $h(c(\cdot))$ with $h$ finite and convex and $c$ smooth), Clarke stationarity guarantees that there exist no directions of linear descent. However, in general Clarke stationarity does not rule out descent directions: the function $x \mapsto -|x|$ at $x = 0$ is a simple example.

Here we show how BFGS can converge to a point at which there exist directions of linear descent. We begin with some relevant definitions (see [11]).

Definition 4.1. Consider a function $f : \mathbb{R}^n \to \mathbb{R}$ and a point $\bar{x}$ with $f(\bar{x})$ finite. Consider a vector $v \in \mathbb{R}^n$.

1. We call $v$ a regular subgradient of $f$ at $\bar{x}$, written $v \in \hat{\partial} f(\bar{x})$, if
   
   $$f(x) \geq f(\bar{x}) + \langle v, x - \bar{x} \rangle + o(x - \bar{x}) \text{ as } x \to \bar{x};$$

2. We call $\bar{x}$ regular stationary if $0 \in \hat{\partial} f(\bar{x})$. (In other words, $\bar{x}$ is a local minimizer, up to first order.)

3. We call $v$ a limiting subgradient of $f$ at $\bar{x}$, written $v \in \partial f(\bar{x})$, if there are sequences $x^\nu \to \bar{x}$ with $f(x^\nu) \to f(\bar{x})$ and $v^\nu \in \partial f(x^\nu)$ with $v^\nu \to v$.

4. We call $\bar{x}$ limiting stationary if $0 \in \partial f(\bar{x})$.

5. If $f$ is Lipschitz around $\bar{x}$ and $0 \in \text{con}(\partial f(\bar{x}))$, then we call $\bar{x}$ Clarke stationary.

A direction $p \in \mathbb{R}^n$ satisfying

$$\limsup_{t \downarrow 0} \frac{f(\bar{x} + tp) - f(\bar{x})}{t} < 0,$$

is called a direction of linear descent. (In this case, $\bar{x}$ is clearly not regular stationary.)

Reference [5, Corollary 4.13] gives an example of exact-line-search BFGS applied to $f(x) = ||x||$ in $\mathbb{R}^2$. The complete statement is as follows.

Proposition 4.2. Consider the exact-line-search BFGS method applied to the Euclidean norm in $\mathbb{R}^2$, initialized by

$$x_0 = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \text{ and } H_0 = \begin{pmatrix} 3 & -\sqrt{3} \\ -\sqrt{3} & 3 \end{pmatrix}.$$  

The method generates a sequence of vectors $x_t$ that rotate clockwise through an angle of $\frac{\pi}{3}$ and shrink by a factor $\frac{1}{2}$ at each iteration.

In fact using our inexact line search (Algorithm 3.1) instead of the exact line search generates the same points, as the following calculation shows.

Proposition 4.3. Consider inexact-line-search BFGS applied to the Euclidean norm in $\mathbb{R}^2$. For any $0 < c_1 < \frac{2}{3}$ and $c_1 < c_2 < 1$, suppose the method is initialized by

$$x_0 = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \text{ and } H_0 = \begin{pmatrix} 3 & -\sqrt{3} \\ -\sqrt{3} & 3 \end{pmatrix}.$$
Then the method generates a sequence of vectors $x_k$ that rotate counterclockwise through an angle of $\frac{\pi}{3}$ and shrink by a factor $\frac{1}{2}$ at each iteration. Consider the matrix

$$R = \begin{pmatrix} \frac{1}{2} & \frac{\sqrt{3}}{2} \\ -\frac{\sqrt{3}}{2} & \frac{1}{2} \end{pmatrix}.$$  

Then in fact, at the $k$th iteration we have:

$$x_k = 2^{-k}R^{-k}x_0, \quad \alpha_k = \frac{1}{4}, \quad p_k = 2^{-k}R^{-k}\begin{pmatrix} -3 \\ \sqrt{3} \end{pmatrix} \quad \text{and} \quad H_k = 2^{-k}R^{-k}H_0R^k.$$  

**Proof.** A direct calculation (see [4, Thm. 4.2]) shows that exact-line-search BFGS applied to the function $||x||$ initialized with $x_0$ and $H_0$ generates the sequence $(x_k)$. It is also easy to check that the exact step size for each iteration is $\frac{1}{4}$.

In order to prove the result, it is sufficient to prove that the step size for inexact-line-search BFGS is also $\frac{1}{4}$ for each iteration.

Consider the $k$th iteration. The line search algorithm will try $t = 1$ first. Since $\nabla f(x_k)^T p_k = -3 \times 2^{-k}$ and $f(x_k + p_k) = \sqrt{7} \times 2^{-k}$, then

$$f(x_k + p_k) = 2^{-k} \sqrt{7} > 2^{-k}(1 - 3c_1) = f(x_k) + c_1\nabla f(x_k)^T p_k$$

and

$$\nabla f(x_k + p_k)^T p_k = \frac{9}{\sqrt{7}} \times 2^{-k} \geq -3 \times 2^{-k}c_2 = -c_2\nabla f(x_k)^T p_k.$$  

Hence the algorithm will try $t = \frac{1}{2}$. This time we note

$$f\left(x_k + \frac{1}{2}p_k\right) = 2^{-k} > 2^{-k}(1 - \frac{3}{2}c_1) = f(x_k) + \frac{c_1}{2}\nabla f(x_k)^T p_k,$$

and

$$c_2\nabla f(x_k)^T p_k = -3 \times 2^{-k}c_2 < 3 \times 2^{-k} = \nabla f(x_k + \frac{1}{2}p_k)^T p_k.$$  

Now the algorithm will try $t = \frac{1}{4}$. We observe

$$f(x_k + \frac{1}{4}p_k) = 2^{-(k+1)} > 2^{-k}(1 - \frac{3}{4}c_1) (c_1 < \frac{2}{3}) = f(x_k) + \frac{c_1}{4}\nabla f(x_k)^T p_k,$$

since $c_1 < 2/3$, and

$$c_2\nabla f(x_k)^T p_k = -3 \times 2^{-k}c_2 < 0 = \nabla f(x_k + \frac{1}{4}p_k)^T p_k.$$  

We deduce $\alpha_k = 1/4$. The result follows. \hfill \Box

The above example indicates that inexact-line-search BFGS only visits points on the half-lines $R_+(\cos \frac{\pi n}{3}, \sin \frac{\pi n}{3})$ (for integers $n$). To construct an example where the algorithm converges to a point at which there exist descent directions, the idea is to ensure that BFGS still only visits those points, but to change the function values elsewhere.

**Proposition 4.4.** Consider inexact-line-search BFGS applied to the function

$$g(u, v) = \begin{cases} \sqrt{u^2 + v^2} \cdot \cos(18 \arctan \frac{u}{v}) & (u, v) \neq (0, 0) \\ 0 & (u, v) = (0, 0), \end{cases}$$

or equivalently in polar coordinates

$$g(r, \theta) = r \cos(18\theta).$$

For any $0 < c_1 < \frac{2}{3}$ and $c_1 < c_2 < 1$, if we initialize with

$$x_0 = \begin{pmatrix} u_0 \\ v_0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \quad \text{and} \quad H_0 = \begin{pmatrix} 3 & \sqrt{3} \\ -\sqrt{3} & 3 \end{pmatrix},$$

the method generates the same sequence as in Proposition 4.3. Moreover, the method converges to the point zero, at which there exist directions of linear descent.
Proof. The existence of directions of linear descent at zero is clear, so we simply need to prove that the BFGS method generates the same sequence \((x_k)\) as in Proposition 4.3 by induction.

Since
\[
R = \begin{pmatrix} \frac{2}{3} & \frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{3} & \frac{1}{2} \end{pmatrix} = \begin{pmatrix} \cos \frac{\pi}{3} & \sin \frac{\pi}{3} \\ -\sin \frac{\pi}{3} & \cos \frac{\pi}{3} \end{pmatrix},
\]
we compute
\[
R^{-1} = \begin{pmatrix} \frac{\sqrt{2}}{3} & \frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{3} & \frac{1}{2} \end{pmatrix} \quad R^k = \begin{pmatrix} \cos \frac{2\pi}{3} & \sin \frac{2\pi}{3} \\ -\sin \frac{2\pi}{3} & \cos \frac{2\pi}{3} \end{pmatrix} \quad R^{-k} = \begin{pmatrix} \cos \frac{2\pi}{3} & -\sin \frac{2\pi}{3} \\ \sin \frac{2\pi}{3} & \cos \frac{2\pi}{3} \end{pmatrix}.
\]
Then \(g(x_k) = g(2^{-k}R^{-k}x_0) = 2^{-k} = f(x_k)\). Furthermore we have
\[
\frac{\partial g(x)}{\partial u} \bigg|_{x=x_k} = \frac{u_k}{\|x_k\|} \cos \left(18 \arctan \frac{v_k}{u_k}\right) = \frac{u_k}{\|x_k\|} \quad \text{and} \quad \frac{\partial g(x)}{\partial v} \bigg|_{x=x_k} = \frac{v_k}{\|x_k\|} \cos \left(18 \arctan \frac{v_k}{u_k}\right) = \frac{v_k}{\|x_k\|},
\]
so
\[
\nabla g(x_k) = \frac{x_k}{\|x_k\|} = \nabla f(x_k).
\]

If we can show that, at each iteration \(k\), the step size is always \(1\), then the iterates indeed coincide, as required. The idea of the proof is to compare the functions \(f = \| \cdot \|\) and \(g\) along the search directions at each iteration, and observe that the calculations during the inexact line search are identical. The figure illustrates.

![Figure 2: A comparison of \(f\) and \(g\) along search direction \(p_0\).](image)

When \(k = 0\), we have
\[
p_0 = -H_0 \nabla g(x_0) = \begin{pmatrix} -3 \\ \sqrt{3} \end{pmatrix}, \quad \nabla g(x_0) = \nabla f(x_0) = \begin{pmatrix} 1 \\ 0 \end{pmatrix}.
\]
Consider the line search algorithm applied to \(g\). We first try \(t = 1\). Since
\[
g(x_0 + p_0) = g(-2, \sqrt{3}) = \sqrt{7} \cos(18 \arctan \frac{-\sqrt{3}}{2}) > 1 = g(x_0) > g(x_0) + c_1 \nabla g(x_0)^T p_0,
\]
so \(t = 1\) doesn’t satisfy the first Wolfe condition. Moreover, as Figure 2 illustrates,
\[
\nabla g(x_0 + p_0)^T p_0 \geq \nabla f(x_0 + p_0)^T p_0.
\]
Since
\[
\nabla f(x_0 + p_0)^T p_0 \geq c_2 \nabla f(x_0)^T p_0 = c_2 \nabla g(x_0)^T p_0,
\]
the value \(t = 1\) satisfies the second Wolfe condition for \(g\). Therefore the line search algorithm will try \(t = \frac{1}{2}\). As Figure 2 indicates,
\[
g(x_0 + \frac{1}{2} p_0) = f(x_0 + \frac{1}{2} p_0) \quad \text{and} \quad \nabla g(x_0 + \frac{1}{2} p_0) = \nabla f(x_0 + \frac{1}{2} p_0).
\]
Hence $t = \frac{1}{2}$ doesn’t satisfy the first Wolfe condition but does satisfy the second, following Proposition 4.3. Hence the line search will next try $t = \frac{1}{2}$. Since
\[ g(x_0) = f(x_0), \quad \nabla g(x_0) = \nabla f(x_0), \quad g(x_0 + \frac{1}{4}p_0) = f(x_0 + \frac{1}{4}p_0), \quad \nabla g(x_0 + \frac{1}{4}p_0) = \nabla f(x_0 + \frac{1}{4}p_0), \]
it follows that $t = \frac{1}{2}$ satisfies the Wolfe conditions. Hence the iterates coincide for $k = 1$.

We now proceed inductively, in similar fashion. We suppose that up to $k$th iteration the iterates coincide, and furthermore $p_k = 2^{-k}R^{-k}p_0$ and $H_k = R^{-k}H_0R^k$. We want to prove coincidence at the $k + 1$th iteration. First notice that $x_k + tp_k = 2^{-k}R^{-k}(x_0 + tp_0)$ can be obtained by rotating $2^{-k}(x_0 + tp_0)$ counterclockwise through an angle of $\frac{k\pi}{2}$. Then we have $f(x_k + tp_k) = 2^{-k}f(x_0 + tp_0)$ and $g(x_k + tp_k) = 2^{-k}g(x_0 + t_0p_0)$. Therefore, by the above argument, the line search step size should be $\alpha_k = \frac{1}{2}$. Hence, the result follows. \hfill $\square$

5 Line-search BFGS versus trust-region BFGS

Given the apparent success of line-search BFGS methods on nonsmooth functions, it is natural to compare with trust-region versions. We consider here a trust-region BFGS algorithm from [8].

Algorithm 5.1. (Trust-region BFGS algorithm)
Given a starting point $x_0$, initial Hessian approximation $B_0$, trust-region radius $\Delta_0$, maximum number of iteration $N$, parameters $\eta \in (0, 10^{-3})$ and $r \in (0, 1)$;
$k \leftarrow 0$;
while $k < N$;

Exactly solve the subproblem
\[ s_k \leftarrow \arg\min_{s} \{ \nabla f_k^T s + \frac{1}{2}s^T B_k s : ||s|| \leq \Delta_k \} ; \]

Compute
\[
\begin{align*}
 y_k & \leftarrow \nabla f(x_k + s_k) - \nabla f_k \\
 \text{ared} & \leftarrow f_k - f(x_k + s_k) \\
 \text{pred} & \leftarrow -\langle \nabla f_k^T s_k + \frac{1}{2}s_k^T B_k s_k \rangle ;
\end{align*}
\]

if $\frac{\text{ared}}{\text{pred}} > \eta$
\[ x_{k+1} \leftarrow x_k + s_k ; \]
else $x_{k+1} = x_{k+1}$;
end(if)
if $\frac{\text{ared}}{\text{pred}} > 0.75$
\[ \text{if } ||s_k|| \leq 0.8\Delta_k, \Delta_{k+1} \leftarrow \Delta_k ; \]
\[ \text{else } \Delta_{k+1} \leftarrow 2\Delta_k ; \]
end(if)
elseif $0.1 \leq \frac{\text{ared}}{\text{pred}} \leq 0.75, \Delta_{k+1} \leftarrow \Delta_k ;$
else $\Delta_{k+1} = 0.5\Delta_k ;$
end(if)
if $||s_k^T (y_k - B_k s_k)|| \geq r ||s_k|| \cdot ||y_k - B_k s_k||$,
\[ B_{k+1} \leftarrow B_k - \frac{B_k s_k y_k^T B_k}{s_k^T B_k s_k} + \frac{y_k y_k^T}{y_k^T s_k} ; \]
else $B_{k+1} \leftarrow B_k ;$
end(if)
k \leftarrow k + 1 ;
end(while)
Numerical experiments show that line-search BFGS methods work well for broad classes of nonsmooth functions, while trust-region versions fail even on simple examples. In this section, we use the simple nonsmooth function $f(u, v) = u^2 + |v|$ to explore some intuitive reasons for the success of line-search BFGS methods over their trust-region counterparts.

We present some simple numerical experiments. The following graph on the left is an example where trust-region BFGS fails to converge to the optimal solution. In contrast, the right graph shows the success of line-search BFGS on the same example.

![Graphs showing convergence](image)

(a) trust region BFGS with starting point (9,9)  
(b) line search BFGS with starting point (9,9)

Figure 3: numerical results on $f(u, v) = u^2 + |v|$.

Points on the axis $v = 0$ are nonsmooth. Numerical results show that the line-search BFGS method generates a sequence of points that eventually cross that axis at every iteration (see the lower right figure above). Indeed, this property can be proved analytically for exact-line search BFGS, as we saw above. However, trust-region BFGS method seems to satisfy no analogous property. The trust region seems overly restrictive on the updated point and approximate Hessian. Somehow, the line-search BFGS method seems to detect the nonsmooth structure of the function better than the trust-region BFGS method.

Secondly, line-search BFGS method updates the approximate Hessian when it finds a point satisfying the Wolfe conditions along the current search direction, and the Wolfe conditions seem to ensure that the updated point is satisfactory for this update. However, trust-region BFGS updates the approximated Hessian matrix at each iteration, even when the current subproblem is not a good approximation of the original problem around the current point.

Thirdly, numerical results show that the radius of the trust region converges to zero quickly (see the lower left figure above). When the trust region is small, the method cannot take a big step even though the subproblem is a good approximation of the original problem. This causes the method to converge very slowly. In addition, for the same reason, the method fails to take advantage of a well approximated subproblem to better update the approximate Hessian.

6 Conclusion

This paper presents some initial explorations of the line-search BFGS method on some very simple nonsmooth examples. The examples provide some interesting illustrations of how the method seems gradually to identify the nonsmooth structure. In particular, the inexact line-search algorithm seems important in responding to nonsmoothness in the objective function. However, the theory underlying this phenomenon is poorly understood.
7 Appendix: Supplemental proofs

Proof of Proposition 2.2. Let $\rho = \frac{a}{(a+1)^{2}}$. We have

$$p_0 = -H_0 \nabla f_0 = \begin{pmatrix} -\frac{a}{(a+1)^{2}} \\ -\frac{a+1}{(a+1)^{2}} \end{pmatrix} \quad \gamma_0 = \nabla f_1 - \nabla f_0 = \begin{pmatrix} \frac{2a}{a+1} \\ -(a+1) \end{pmatrix}$$

$$V_0 = I - (p_0^T \gamma_0) p_0 \gamma_0^T$$

$$= \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} - \frac{(a+1)^2}{2a^2 + a + 1} \begin{pmatrix} \frac{2a}{(a+1)^2} & a \\ \frac{1}{a+1} & 2 \end{pmatrix}$$

$$= \begin{pmatrix} \frac{a+1}{a^2 + a + 1} & \frac{a}{2(a^2 + a + 1)} \\ \frac{a}{a^2 + a + 1} & \frac{2a^2 + a + 1}{a^2 + a + 1} \end{pmatrix}$$

$$\alpha_0 = \text{argmin} \{ (1 - \frac{a}{a + 1} \alpha)^2 - a(\frac{2}{a^2 + 3a + 1} - \frac{2}{(a + 1)^2} \alpha) \} = 1.$$  

$$H_1 = V_0 H_0 V_0^T + \alpha_0 (p_0^T \gamma_0) p_0$$

$$= \begin{pmatrix} \frac{a+1}{a^2 + a + 1} & \frac{a}{2(a^2 + a + 1)} \\ \frac{a}{a^2 + a + 1} & \frac{2a^2 + a + 1}{a^2 + a + 1} \end{pmatrix}$$

$$+ \frac{(a+1)^2}{2a^2 + 2a + 2} \begin{pmatrix} \frac{a}{(a^2 + a + 1)} \\ \frac{1}{(a+1)(a^2 + a + 1)} \end{pmatrix}$$

$$= \begin{pmatrix} \frac{a^2 + a + 1}{2(a^2 + a + 1)} & \frac{a}{2(a^2 + a + 1)} \\ \frac{1}{2(a^2 + a + 1)} & \frac{a^2 + a + 1}{a^2 + a + 1} \end{pmatrix}$$

$$p_1 = -H_1 \nabla f_1 = -\frac{2a^2 + a + 1}{(a^2 + a + 1)(a^2 + a + 1)} \begin{pmatrix} \frac{a}{a^2 + a + 1} \\ \frac{1}{2(a^2 + a + 1)} \end{pmatrix}$$

$$\alpha_1 = \text{argmin} \{ (1 - \frac{a}{a + 1} \alpha)^2 + \max \left\{ \frac{2a}{a^2 + 3a + 1}, \frac{1}{(a+1)^2}, \frac{2a^2}{(a+1)^3}, a - \frac{2a}{a^2 + 3a + 1} \right\} \}$$

$$= \text{argmin} \{ (1 - \frac{a}{a + 1} \alpha)^2 + \frac{2a}{a^2 + 3a + 1} \}$$

$$= \frac{1}{a(a+1)}$$

$$p_{2k} = -H_{2k} \nabla f_{2k}$$

$$= -\begin{pmatrix} \frac{a^2 + 2a + 2}{2(a^2 + a + 1)} & \rho^k \\ \frac{1}{a(a+1)} & 1 \end{pmatrix} \begin{pmatrix} 2 \rho^k \\ 1 \end{pmatrix}$$

$$= \begin{pmatrix} -\frac{(a+1)\rho^k}{a^2 + a + 1} \\ \frac{\rho^k}{a(a+1)} \end{pmatrix}$$

Suppose up to iteration 2k the statement is true. Let us compute the cases for iterations 2k + 1 and 2k + 2.

$$\alpha_{2k} = \text{argmin} \{ (\rho^k - \frac{(a + 1)\rho^k}{a^2 - a})^2 - a(\frac{2\rho^{2k}}{a^2 + 3a + 1} - \frac{2\rho^{2k}}{a^2 - a}) \} = a \rho.$$
\[ y_{2k} = \nabla f_{2k+1} - \nabla f_{2k} = (2 - \frac{\rho^k}{a + 1} - 2\rho^k, -(a + 1))^T = \left(-\frac{2\rho^k}{a + 1}, -(a + 1)\right)^T. \]

\[ V_{2k} = I - (p_{2k}^T y_{2k})^{-1} p_{2k} y_{2k}^T \]
\[ = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} - \frac{2\rho^{2k}}{a^2 + (a + 1)} \frac{\rho^{-1}}{a^2 + (a + 1)} \begin{pmatrix} 2\rho^k & 0 \\ 0 & 2\rho^k \end{pmatrix} \]
\[ = \begin{pmatrix} \alpha^{-1} & 0 \\ 0 & \alpha^{-1} \end{pmatrix} - \frac{a\alpha^{-1}}{2a^2 + (a + 1)} \begin{pmatrix} \alpha^{-1} & 0 \\ 0 & \alpha^{-1} \end{pmatrix} \]

\[ H_{2k+1} = V_{2k} H_{2k} V_{2k}^T + \alpha_{2k} (p_{2k}^T y_{2k})^{-1} (p_{2k} y_{2k}^T) \]
\[ = \begin{pmatrix} \alpha^{-1} & 0 \\ 0 & \alpha^{-1} \end{pmatrix} - \frac{a\alpha^{-1}}{2a^2 + (a + 1)} \begin{pmatrix} \alpha^{-1} & 0 \\ 0 & \alpha^{-1} \end{pmatrix} \]
\[ = \begin{pmatrix} \alpha^{-1} & 0 \\ 0 & \alpha^{-1} \end{pmatrix} - \frac{a\alpha^{-1}}{2a^2 + (a + 1)} \begin{pmatrix} \alpha^{-1} & 0 \\ 0 & \alpha^{-1} \end{pmatrix} \]
\[ = \begin{pmatrix} \alpha^{-1} & 0 \\ 0 & \alpha^{-1} \end{pmatrix} - \frac{a\alpha^{-1}}{2a^2 + (a + 1)} \begin{pmatrix} \alpha^{-1} & 0 \\ 0 & \alpha^{-1} \end{pmatrix} \]

\[ p_{2k+1} = -H_{2k+1} \nabla f_{2k+1} = \begin{pmatrix} \alpha^{-1} & 0 \\ 0 & \alpha^{-1} \end{pmatrix} - \frac{a\alpha^{-1}}{2a^2 + (a + 1)} \begin{pmatrix} \alpha^{-1} & 0 \\ 0 & \alpha^{-1} \end{pmatrix} \]

\[ \alpha_{2k+1} = \argmin \left\{ \left( \frac{\rho^k}{a + 1} - \rho^k \right)^2 - \frac{2\rho^{2k+1}}{a^2 + 3a + 1} + 2\rho^{2k+1} \right\} \]
\[ = \argmin \left\{ \rho^{2k} a^2 - \frac{2\rho^{2k}}{a + 1} + 2\rho^{2k+1} - \frac{2\rho^{2k+1}}{a^2 + 3a + 1} + \rho^{2k} \right\} \]
\[ = \frac{\rho}{a} \]

\[ y_{2k+1} = \nabla f_{2k+2} - \nabla f_{2k+1} = \left(-\frac{2\rho^k}{a^2 + (a + 1)} \right) \]

\[ V_{2k+1} = I - (p_{2k+1}^T y_{2k+1})^{-1} p_{2k+1} y_{2k+1}^T \]
\[ = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} - \frac{(a + 1)^2}{2\rho^{2k}(a^2 + a + 1)} \begin{pmatrix} \rho^k & 0 \\ 0 & \rho^k \end{pmatrix} \]
\[ = \begin{pmatrix} \alpha^{-1} & 0 \\ 0 & \alpha^{-1} \end{pmatrix} - \frac{a\alpha^{-1}}{2a^2 + (a + 1)} \begin{pmatrix} \alpha^{-1} & 0 \\ 0 & \alpha^{-1} \end{pmatrix} \]

16
\begin{align*}
H_{2k+2} &= V_{2k+1}H_{2k+1}V_{2k+1}^T + \alpha_{2k+1}(p_{2k+1}^T y_{2k+1})^{-1} p_{2k+1} p_{2k+1}^T \\
&= \left( \begin{array}{c}
\frac{a^2+2a+1}{2(a^2+a+1)} \\
\frac{(a+1)^3}{(a+1)(a^2+a+1)} \\
\frac{2a^2+2a+1}{2(a^2+4a+1)} \\
\frac{a^2+2a+1}{2(a^2+4a+1)} \\
\frac{a^2a+1}{4a^2} \\
\frac{a^2}{2a^2+4a+1} \\
\frac{a^2a+1}{4a^2} \\
\frac{a^2}{2a^2+4a+1} \\
\frac{a^2a+1}{4a^2} \\
\frac{a^2}{2a^2+4a+1}
\end{array} \right) \left( \begin{array}{c}
\frac{2a^2-2a+1}{2(a^2+4a+1)} \\
\frac{a^2+2a+1}{2(a^2+4a+1)} \\
\frac{a^2}{2a^2+4a+1} \\
\frac{a^2}{2a^2+4a+1} \\
\frac{a^2}{2a^2+4a+1} \\
\frac{a^2}{2a^2+4a+1} \\
\frac{a^2}{2a^2+4a+1} \\
\frac{a^2}{2a^2+4a+1} \\
\frac{a^2}{2a^2+4a+1} \\
\frac{a^2}{2a^2+4a+1}
\end{array} \right) \\
&= \frac{1}{2a^2(a^2+a+1)} \left( \begin{array}{c}
\frac{a^2+2a+1}{2(a^2+a+1)} \\
\frac{(a+1)^3}{(a+1)(a^2+a+1)} \\
\frac{2a^2+2a+1}{2(a^2+4a+1)} \\
\frac{a^2+2a+1}{2(a^2+4a+1)} \\
\frac{a^2a+1}{4a^2} \\
\frac{a^2}{2a^2+4a+1} \\
\frac{a^2a+1}{4a^2} \\
\frac{a^2}{2a^2+4a+1} \\
\frac{a^2a+1}{4a^2} \\
\frac{a^2}{2a^2+4a+1}
\end{array} \right) \left( \begin{array}{c}
\frac{2a^2-2a+1}{2(a^2+4a+1)} \\
\frac{a^2+2a+1}{2(a^2+4a+1)} \\
\frac{a^2}{2a^2+4a+1} \\
\frac{a^2}{2a^2+4a+1} \\
\frac{a^2}{2a^2+4a+1} \\
\frac{a^2}{2a^2+4a+1} \\
\frac{a^2}{2a^2+4a+1} \\
\frac{a^2}{2a^2+4a+1} \\
\frac{a^2}{2a^2+4a+1} \\
\frac{a^2}{2a^2+4a+1}
\end{array} \right)
\end{align*}

The proposition follows. \(\square\)

References


