A Quasi-Newton Approach to Nonsmooth Convex Optimization

Abstract

We extend the well-known BFGS quasi-Newton method and its limited-memory variant (LBFGS) to the optimization of nonsmooth convex objectives. This is done in a rigorous fashion by generalizing three components of BFGS to subdifferentials: The local quadratic model, the identification of a descent direction, and the Wolfe line search conditions. We apply the resulting sub(L)BFGS algorithm to $L_2$-regularized risk minimization with binary hinge loss. In both settings our generic algorithms perform comparable to or better than their counterparts in specialized state-of-the-art solvers.

1. Introduction

The (L)BFGS quasi-Newton method (Nocedal and Wright, 1999) is widely regarded as the workhorse of smooth nonlinear optimization due to its combination of computational efficiency with good asymptotic convergence. Given a smooth objective function $J : \mathbb{R}^d \to \mathbb{R}$ and a current iterate $w_t \in \mathbb{R}^d$, BFGS forms a local quadratic model of $J$:

$$Q_t(p) := J(w_t) + \frac{1}{2}p^\top B_t^{-1}p + \nabla J(w_t)^\top p, \quad (1)$$

where $B_t > 0$ is a positive-definite estimate of the inverse Hessian of $J$. Minimizing $Q_t(p)$ gives the quasi-Newton direction

$$p_t := -B_t \nabla J(w_t), \quad (2)$$

which is used for the parameter update:

$$w_{t+1} = w_t + \eta_t p_t. \quad (3)$$

The step size $\eta_t \in \mathbb{R}_+$ is normally determined by a line search obeying the Wolfe conditions:

$J(w_{t+1}) \leq J(w_t) + c_1 \eta_t \nabla J(w_t)^\top p_t$ and

$\nabla J(w_{t+1})^\top p_t \geq c_2 \nabla J(w_t)^\top p_t, \quad (4)$

with $0 < c_1 < c_2 < 1$. The matrix $B_t$ is then modified via the incremental rank-two update

$$B_{t+1} = (I - \varrho_t s_t y_t^\top)B_t(I - \varrho_t y_t s_t^\top) + \varrho_t s_t s_t^\top, \quad (5)$$

where $s_t := w_{t+1} - w_t$ and $y_t := \nabla J(w_{t+1}) - \nabla J(w_t)$ denote the most recent step along the optimization trajectory in parameter and gradient space, respectively, and $\varrho_t := (y_t^\top s_t)^{-1}$. Given a descent direction $p_t$, the Wolfe conditions ensure that $(\forall t) s_t^\top y_t > 0$ and hence $B_0 > 0 \implies (\forall t) B_t > 0$.

Limited-memory BFGS (LBFGS) is a variant of BFGS designed for solving large-scale optimization problems where the $O(d^2)$ cost of storing and updating $B_t$ would be prohibitively expensive. LBFGS approximates the quasi-Newton direction directly from the last $m$ pairs of $s_t$ and $y_t$ via a matrix-free approach. This reduces cost to $O(md)$ space and time per iteration, with $m$ freely chosen (Nocedal and Wright, 1999).

Smoothness of the objective function is essential for standard (L)BFGS because both the local quadratic model (1) and the Wolfe conditions (4) require the existence of the gradient $\nabla J$ at every point. Even though nonsmooth convex functions are differentiable everywhere except on a set of (Lebesgue) measure zero (Hiriart-Urruty and Lemaréchal, 1993), in practice (L)BFGS often fails to converge on such problems (Lukšan and Vlček, 1999; Haarala, 2004). Various subgradient-based approaches, such as subgradient descent (Nedich and Bertsekas, 2000) or bundle methods (Teo et al., 2007), are therefore preferred.

Although a convex function might not be differentiable everywhere, a subgradient always exists. Let $w$ be a point where a convex function $J$ is finite. Then a subgradient is the normal vector of any tangential supporting hyperplane of $J$ at $w$. Formally, $g$ is called a subgradient of $J$ at $w$ if and only if

$$J(w') \geq J(w) + (w' - w)^\top g, \quad \forall w'. \quad (6)$$

The set of all subgradients at a point is called the subdifferential, and is denoted $\partial J(w)$. If this set is not empty then $J$ is said to be subdifferentiable at $w$. If it contains exactly one element, i.e., $\partial J(w) = \{ \nabla J(w) \}$, then $J$ is said to be differentiable at $w$.

In this paper we systematically modify the standard (L)BFGS algorithm so as to make it amenable to sub-
gradients. This results in sub(L)BFGS, a new subgradient quasi-Newton method which is applicable to wide variety of nonsmooth convex optimization problems encountered in machine learning.

In the next section we describe our new algorithm generically, before we discuss its application to the task of $L_2$-regularized risk minimization with hinge loss in Section 3. Section 4 compares and contrasts our work with other recent efforts in this area. Encouraging experimental results are reported in Section 5. Finally, we conclude with an outlook and discussion in Section 6.

2. BFGS Subgradient Method

We modify the standard BFGS algorithm to derive our new algorithm (subBFGS, Algorithm 1) for nonsmooth convex optimization. These modifications can be grouped into three areas, which we shall elaborate on in turn: generalizing the local quadratic model, finding a descent direction, and finding a step size that obeys a subgradient reformulation of the Wolfe conditions.

Algorithm 1 subBFGS
1: Initialize $t := 0$, $w_0 = 0$, and $B_0 = I$;
2: Compute (sub)gradient $g_0 \in \partial J(w_0)$;
3: while not converged do
4: \hspace{1em} $p_t = \text{descentDirection}(g_t, \epsilon)$; (Algorithm 2)
5: \hspace{1em} if $p_t = 0$ then
6: \hspace{2em} Return $w_t$;
7: \hspace{1em} end if
8: \hspace{1em} Find $\eta_t$ that obeys (15); (e.g. Algorithm 3)
9: \hspace{1em} $s_t = \eta_t p_t$;
10: \hspace{1em} $w_{t+1} = w_t + s_t$;
11: \hspace{1em} Compute (sub)gradient $g_{t+1} \in \partial J(w_{t+1})$;
12: \hspace{1em} $y_t = g_{t+1} - g_t$;
13: \hspace{1em} Update $B_{t+1}$ via (5);
14: \hspace{1em} $t := t + 1$;
15: end while

2.1. Generalizing the Local Quadratic Model

Recall that BFGS assumes the objective function $J$ is differentiable everywhere, so that at the current iterate $w_t$ we can construct a local quadratic model (1) to $J(w_t)$. For a nonsmooth objective function, such a model becomes ambiguous at the non-differentiable points (Figure 1). To resolve the ambiguity, we could simply replace the gradient $\nabla J(w_t)$ in (1) with some subgradient, $g_t \in \partial J(w_t)$. However, as will be discussed later, the resulting quasi-Newton direction

\[ p_t := -B_t g_t \]

is not necessarily a descent direction. To address this fundamental modeling problem, we first generalize the local quadratic model as follows:

\[ Q_t(p) := J(w_t) + M_t(p), \text{ where} \]

\[ M_t(p) := \frac{1}{2} p^\top B_t^{-1} p + \sup_{g \in \partial J(w_t)} g^\top p. \]  

Note that where $J$ is differentiable, (7) reduces to the familiar BFGS quadratic model (1). At non-differentiable points, however, the model is no longer quadratic, as the sup may be attained at different elements of $\partial J(w_t)$ for different directions $p$. Instead it can be viewed as the tightest pseudo-quadratic fit to $J$ at $w_t$ (Figure 1).

Ideally, we would like to minimize $Q_t(p)$, or equivalently $M_t(p)$, in (7) to obtain the best search direction,

\[ p^* := \arg\inf_{p \in \mathbb{R}^d} M_t(p). \]  

This is generally intractable due to the presence of the supremum over the entire subdifferential $\partial J(w_t)$. In many machine learning problems, however, the set $\partial J(w_t)$ has some special structure that simplifies calculation of the supremum in (7). In what follows, we develop an iteration that is guaranteed to find a quasi-Newton descent direction, assuming an oracle that supplies $\arg\sup_{g \in \partial J(w_t)} g^\top p$ for a given direction $p \in \mathbb{R}^d$. In Section 3.1 we provide an efficient implementation of such an oracle for $L_2$-regularized risk minimization with the hinge loss.

2.2. Finding a Descent Direction

A direction $p_t$ is a descent direction if and only if

\[ g^\top p_t < 0 \ \forall g \in \partial J(w_t) \]  

(Belloni, 2005), or equivalently

\[ \sup_{g \in \partial J(w_t)} g^\top p_t < 0. \]  

Figure 1. Possible quadratic models (dashed) at a subdifferentiable point (solid red disk) versus tightest pseudo-quadratic approximation (7) (bold dashes) to the objective function (solid red line).
A Quasi-Newton Approach to Nonsmooth Convex Optimization

In particular, for a smooth convex function the quasi-Newton direction (2) is always a descent direction because \( \nabla J(w_t) \top p_t = -\nabla J(w_t) \top B_i \nabla J(w_t) < 0 \) holds due to the positivity of \( B_i \).

For nonsmooth functions, however, the quasi-Newton direction \( p_t := -B_i g_t \) for a given \( g_t \in \partial J(w_t) \) may not fulfill the descent condition (9), making it impossible to find a step that obeys (4), thus causing a failure of the line search. We now present an iterative approach to find a quasi-Newton descent direction.

Inspired by bundle methods (Teo et al., 2007), we iteratively compute the following convex lower bound to \( M_i(p) \):

\[
M^i_t(p) := \frac{1}{2} p \top B_i^{-1} p + \sup_{j \leq i} g^\top j p, \quad i, j \in \mathbb{N}. \tag{10}
\]

This bound is successively tightened by computing

\[
p^i := \arg\min_{p \in \mathbb{R}^d} M^i_t(p) \quad \text{and} \quad g^{i+1} := \arg\sup_{g \in \partial J(w_t)} g^\top p^i. \tag{11}
\]

Here we set \( g^1 \in \partial J(w_t) \), and assume that \( g^{i+1} \) is provided by an oracle. To solve \( \inf_{p \in \mathbb{R}^d} M^i_t(p) \), we rewrite it as a constrained optimization problem:

\[
\inf_{p, \xi} \frac{1}{2} p \top B_i^{-1} p + \xi, \quad \text{s.t.} \quad g^\top j p \leq \xi, \forall j \leq i. \tag{13}
\]

This problem can be solved exactly via quadratic programming (QP), but doing so may incur substantial computational expense. Instead we adopt an alternate approach (Algorithm 2) which does not solve \( \inf_{p \in \mathbb{R}^d} M^i_t(p) \) to optimality. The key idea is to write the proposed descent direction at iteration \( i + 1 \) as a convex combination of \( p^i \) and \( -B_i g^{i+1} \). The optimal combination coefficient \( \mu^* \) can be computed exactly (Step 5 of Algorithm 2) using an argument based on maximizing dual progress. We now define

\[
\epsilon^i := \min_{j \leq i} M^i_{j+1}(p^j) - M^i_j(p^j)
\]

\[
= \min_{j \leq i} p^j \top g^{j+1} - \frac{1}{2} (p^j \top \tilde{g}^j + p^i \top \tilde{g}^i), \tag{14}
\]

which is monotonically decreasing, and upper bounds the distance from optimality. This indicates that we can stop finding the search direction once \( \epsilon^i \) falls below a pre-specified tolerance \( \epsilon \). Details can be found in Appendix A, where we also prove the convergence of this procedure.\(^1\)

2.3. Subgradient Line Search

Given the current iterate \( w_t \) and a search direction \( p_t \), the task of a line search is to find a step size \( \eta \in \mathbb{R}_+ \) which decreases the objective function along the line \( w_t + \eta p_t, \ i.e., \ J(w_t + \eta p_t) := \Phi(\eta) \). The Wolfe conditions (4) are used in line search routines to enforce a sufficient decrease in the objective function, and to exclude unnecessarily small step sizes (Nocedal and Wright, 1999). However, the original Wolfe conditions require the objective function to be smooth (though not necessarily convex). To extend them to nonsmooth convex problems, we propose the following subgradient reformulation:

\(^1\)Appendices could not be submitted this year, though may be made available to reviewers if specifically requested.
\[ J(w_{t+1}) \leq J(w_t) + c_1 \eta_t \sup_{g \in \partial J(w_{t+1})} g^\top p_t, \]
and
\[ \sup_{g' \in \partial J(w_{t+1})} g'^\top p_t \geq c_2 \sup_{g \in \partial J(w_t)} g^\top p_t, \]  
(15)
where \(0 < c_1 < c_2 < 1\). Figure 2 illustrates how these conditions enforce non-trivial step sizes that decrease the objective. In Appendix B we formally show that for any given search direction we can always find a positive step size that satisfies (15). \(^1\)

2.4. Limited-Memory subBFGS (subLBFGS)

It is straightforward to implement an LBFGS variant of our subBFGS algorithm: We simply modify Algorithms 1 and 2 to compute all products of \(B_i\) with a vector by means of the standard LBFGS matrix-free scheme (Nocedal and Wright, 1999).

3. sub(L)BFGS Implementation for \(L_2\)-Regularized Risk Minimization

Many machine learning algorithms can be viewed as minimizing a convex regularized risk:

\[ J(w) := \frac{c}{2} \|w\|^2 + \frac{1}{n} \sum_{i=1}^{n} l(w^\top x_i, z_i), \]  
(16)
where \(x_i \in \mathcal{X} \subseteq \mathbb{R}^d\) are the training instances, \(z_i \in \mathcal{Z} \subseteq \mathbb{R}\) the corresponding labels, and \(l\), the loss, is a non-negative convex function of \(w\) which measures the discrepancy between \(z_i\) and the predictions arising from \(w\) via \(w^\top x_i\). A loss function commonly used for binary classification is the hinge loss with \(z \in \{\pm 1\}\) and

\[ l(w^\top x, z) := \max(0, 1 - z w^\top x). \]  
(17)
It is easy to see that regularized risk minimization with the binary hinge loss is a convex but nonsmooth optimization problem. In this section we show how sub(L)BFGS (Algorithm 1) can be applied to this problem.

Differentiating (16) after plugging in (17) yields:

\[ \partial J(w) = c w - \frac{1}{n} \sum_{i=1}^{n} \beta_i z_i x_i, \]  
(18)
where \(\bar{w} := c w - \frac{1}{n} \sum_{i \in \mathcal{E}} z_i x_i\) with

\[ \beta_i := \begin{cases} 1 & \text{if } i \in \mathcal{E}, \\ 0,1 & \text{if } i \in \mathcal{M}, \\ 0 & \text{if } i \in \mathcal{W}. \end{cases} \]

\(\mathcal{E}\), \(\mathcal{M}\), and \(\mathcal{W}\) denote the set of points which are in error, on the margin, and well-classified, respectively.

3.1. Realizing the Direction Finding Method

Recall that in order to apply our sub(L)BFGS algorithm to a problem one needs an oracle that provides \(\text{argsup}_{g \in \partial J(w_t)} g^\top p\) for a given direction \(p\). For \(L_2\)-regularized risk minimization with binary hinge loss we can implement such an oracle at computational cost linear in the number \(|\mathcal{M}_t|\) of current marginal points. (Normally \(|\mathcal{M}_t| \ll n\).) Towards this end we use (18) to obtain

\[ \sup_{g \in \partial J(w_t)} g^\top p = \sup_{\beta \in \mathcal{M}_t} \left( \bar{w}_t - \frac{1}{n} \sum_{i \in \mathcal{M}_t} \beta_i z_i x_i \right)^\top p. \]  
(19)
Since for a given \(p\) the first term of the right-hand side of (19) is a constant, the supremum is attained when we set \(\beta_i \forall i \in \mathcal{M}_t\) via the following strategy:

\[ \beta_i := \begin{cases} 0 & \text{if } z_i x_i^\top p \geq 0, \\ 1 & \text{if } z_i x_i^\top p < 0. \end{cases} \]  
(20)

3.2. Implementing the Line Search

We first show that the one-dimensional convex function obtained by restricting (16) to a line can be evaluated efficiently. To see this rewrite the objective as

\[ J(w) := \frac{c}{2} \|w\|^2 + \frac{1}{n} \sum_{i=1}^{n} l(w^\top x_i, z_i), \]  
(21)
where \(0\) and \(1\) are column vectors of zeros and ones, respectively, \(\cdot\) denotes the Hadamard (componentwise) product, and \(z \in \mathbb{R}^n\) is a vector of correct labels corresponding to each row of data in \(X := [x_1, x_2, \cdots, x_n]^\top \in \mathbb{R}^{n \times d}\). This allows us to write

\[ \Phi(\eta) := J(w + \eta p) = \frac{c}{2} \|w\|^2 + c \eta \|w\| \|p\| + \frac{c \eta^2}{2} \|p\|^2 + \frac{1}{n} \sum_{i=1}^{n} \delta_{\eta}(1 - (f + \eta \Delta f)), \]  
(22)
where \(f := z(x^\top w), \Delta f := z(x^\top p)\) and for \(1 \leq i \leq n, \delta_{\eta}(i) := \begin{cases} 1 & \text{if } f(i) + \eta \Delta f(i) < 1, \\ 0,1 & \text{if } f(i) + \eta \Delta f(i) = 1, \\ 0 & \text{if } f(i) + \eta \Delta f(i) > 1. \end{cases} \]  
(23)
For a given \(w\) and \(p\) we cache \(f\) and \(\Delta f\), expending \(O(nd)\) computational effort. We also cache \(\frac{1}{2} \|w\|^2, c w^\top p, \) and \(\frac{1}{2} \|p\|^2\), each of which requires \(O(n)\) work. Computing \(\Phi(\eta)\) from this information requires the evaluation of \(\delta_{\eta}\) and \(\delta_{\eta}^t(1 - (f + \eta \Delta f))\), both of which can be calculated with \(O(n)\) effort. All other terms in (22) can be computed in constant time, thus reducing the overall complexity of evaluating \(\Phi(\eta)\) to \(O(n)\). We are now in a position to introduce an exact line search which takes advantage of this scheme.
3.2.1. Exact Line Search

Differentiating (22) with respect to \( \eta \) and setting the gradient to 0 shows that \( \eta^* := \arg \min_{\eta} \Phi(\eta) \) satisfies

\[
\eta^* = \frac{1}{n} \delta^\top \Delta f - c w^\top p_i}{c \| p_i \|^2}.
\]

(24)

It is easy to verify that \( \Phi(\eta) \) is piecewise quadratic (Figure 3). It is differentiable everywhere except at \( \eta = (1 - f(i)) / \Delta f(i) \), where it becomes subdifferential. At all these points an element of the indicator vector \( \delta_\eta \) changes from 0 to 1 or vice versa, implying that \( \delta_\eta \) remains constant in the interval between two subdifferential points, say \( (\eta_a, \eta_b) \).

This motivates the following line search strategy: For each candidate interval \( (\eta_a, \eta_b) \) compute the candidate step size

\[
\eta^*_{\text{trial}} = \frac{1}{n} \delta^\top \Delta f - c w^\top p_i}{c \| p_i \|^2},
\]

where \( \eta' \in (\eta_a, \eta_b) \), and accept it if \( \eta^*_{\text{trial}} \in [\eta_a, \eta_b] \), otherwise accept \( \eta_a \) if 0 \( \in \partial \Phi(\eta_a) \). The full implementation is detailed in Algorithm 3.

4. Related Work

Lukšan and Vlček (1999) propose an extension of BFGS to nonsmooth convex problems. Their algorithm samples gradients around non-differentiable points in order to obtain a descent direction. In many machine learning problems evaluating the objective function and its gradient is very expensive. Therefore, our direction finding algorithm (Algorithm 2) repeatedly samples subgradients from the set \( \partial J(w_i) \) via the oracle, which is computationally more efficient.

Recently, Andrew and Gao (2007) introduced a variant of nonsmooth BFGS, Orthant-Wise Limited-memory Quasi-Newton (OWL-QN) algorithm, suitable for optimizing \( L_1 \)-regularized log-linear models:

\[
J(w) := c \| w \|_1 + \frac{1}{n} \sum_{i=1}^n \log(1 + \exp(-z_i w^\top x_i)).
\]

(26)

In the \( L_1 \)-regularized log-linear models the regularizer is nonsmooth since it incorporates a hinge while the loss is smooth. From the viewpoint of an optimization algorithm this objective function is very similar to the \( L_2 \)-regularized hinge loss problem; the direction finding method and the line search that we discussed in Sections 3.1 and 3.2 respectively can be applied to this problem with slight modifications.

OWL-QN is based on the observation that the \( L_1 \) regularizer is linear within any given orthant. Therefore, it models the curvature of the loss and use an efficient scheme to select orthants for optimization. OWL-QN is specialized to exploit the sparsity induced by the \( L_1 \) regularizer. In fact, its success greatly depends on its direction finding subroutine, which demands a specially chosen subgradient to produce the quasi-Newton direction. As shown in Section 5, the direction finding subroutine of OWL-QN can be replaced by Algorithm 2, which in turn makes the algorithm more robust to the choice of subgradients.
Many optimization techniques use past gradients to build a model of the objective function. Bundle method solvers like BMRM (Teo et al., 2007) and SVMStruct (Joachims, 2006) use them to lower-bound the objective by a piecewise linear function which is minimized to obtain the next iterate. This fundamentally differs from the BFGS approach of using past gradients to approximate the Hessian, hence building a quadratic model of the objective function.

Vojtěch and Sönnenburg (2007) speed up the convergence of a bundle method solver for the $L_2$-regularized binary hinge loss. Their main idea is to perform a line search along the line connecting two successive iterates of a bundle method solver. Although developed independently, their method is very reminiscent of the line search we describe in Section 3.2.1, with one important difference: They only consider the end points of the intervals $(\eta_a, \eta_b)$ as candidate step sizes. Our experiments show that this sometimes results in oscillations which can slow down convergence.

5. Experiments

We now evaluate the performance of our algorithm and compare it to other state-of-the-art nonsmooth optimization methods on the $L_2$-regularized hinge loss. We also apply our direction-finding routine to the $L_1$-regularized logistic loss minimization problem and compare its performance with OWL-QN.

Four datasets were used in our experiments: Covertype dataset of Blackard, Jock & Dean; CCAT from the Reuters RCV1 collection; Astro-physics dataset of abstracts of scientific papers from the Physics ArXiv (Joachims, 2006) and MNIST dataset of handwritten digits with two classes: even and odd digits. We used subLBFGS with a buffer of size $m = 15$ throughout. Table 1 summarizes our parameter settings for all experiments. We followed the choices of Vojtěch and Sönnenburg (2007) for the $L_2$ regularization constants; for $L_1$ they were chosen from the set $10^{\{-6,-5,\ldots,-1\}}$ to achieve the highest prediction rate on the test dataset.

Note that on convex problems such as these every convergent optimizer will reach the same solution; comparing generalisation performance is hence pointless. We therefore combined training and test datasets to evaluate the convergence of each algorithm in terms of the objective function value vs. CPU seconds. All experiments were carried out on a Linux machine with dual 2.8 GHz Xeon processors with 2GB RAM.

### 5.1. $L_2$-Regularized Hinge Loss

For our first set of experiments, we applied subLBFGS together with our exact line search (Algorithm 3) to the task of $L_2$-regularized hinge loss minimization. Our competing algorithms are the bundle method solver BMRM (Teo et al., 2007) and an optimized cutting plane algorithm, OCAS version 0.4.4 (Vojtěch and Sönnenburg, 2007), both of which demonstrated strong results on the $L_2$-regularized hinge loss minimization in their corresponding papers.

As can be seen in Figure 4, subLBFGS (solid) converges noticeably (up to 8 times) faster toward the neighbourhood of the optimum (less than $10^{-3}$ away from the optimum) than BMRM (dashed). As BMRM’s approximation to the objective function improves over the course of optimization, it gradually catches up with subLBFGS, though it is still outperformed by subLBFGS on 3 out of 4 datasets in terms of final convergence speed. The performance of subLBFGS and OCAS (dash-dotted) are very similar. OCAS converges slightly faster than subLBFGS on the Astro-physics dataset, while it is clearly outperformed by subLBFGS on the MNIST dataset due to the oscillations induced by its inadequate line search (cf. end of Section 4).

### 5.2. $L_1$-Regularized Logistic Loss

To demonstrate the utility of our direction-finding routine (Algorithm 2) in its own right, we plugged it into the OWL-QN algorithm (Andrew and Gao, 2007) as an alternative direction-finding method, and compared this modified version (denoted OWL-QN*) with the original on the task of minimizing the $L_1$-regularized logistic loss.

Using the stopping criteria suggested by Andrew and
A Quasi-Newton Approach to Nonsmooth Convex Optimization

Figure 4. Objective function value vs. CPU seconds on $L_2$-regularized hinge loss minimization tasks.

6. Outlook and Discussion

We proposed an extension of BFGS suitable for handling nonsmooth problems often encountered in the machine learning context. As our experiments show, our algorithms are versatile and applicable to many problems. At the same time their performance is comparable to if not better than that of their counterparts in custom-built solvers.

In some experiments we observe that sub(L)BFGS initially makes rapid progress towards the solution but slows down closer to the optimum. We hypothesize that initially its quadratic model allows sub(L)BFGS to make rapid progress, but closer to the optimum it is no longer an accurate model of an objective function dominated by the nonsmooth hinges. We are therefore contemplating hybrid solvers which seamlessly switch between sub(L)BFGS and BMRM.

In this paper we applied sub(L)BFGS to the task of $L_2$-regularized risk minimization with binary hinge loss. It can also be extended to deal with generalizations of the hinge loss, e.g. multi-class, multi-category, and ordinal regression problems. This is part of our ongoing research.

Finally, to put our contributions in perspective, recall that we modified three aspects of the standard BFGS algorithm, namely the quadratic model (Section 2.1), the descent direction-finding (Section 2.2), and the line search (Section 2.3). Each of these modifications is...
A Quasi-Newton Approach to Nonsmooth Convex Optimization

Figure 5. Objective value vs. CPU seconds on $L_1$-regularized logistic loss minimization tasks.

versatile enough to be used as a component in other nonsmooth optimization algorithms. This not only offers the promise of improving existing algorithms, but may also help clarify connections between them. We hope that this will focus attention on those core subroutines that need to be made more efficient in order to handle larger and larger datasets.

References


A. Bundle Search for Descent Direction

Recall from the main paper that at a subdifferential iterate \( w \) our goal is to find a descent direction \( p^* \) which minimizes the pseudo-quadratic model:\footnote{For the ease of extrapolation, from now on we drop the iteration index \( t \).}

\[
M(p) := \frac{1}{2} p^T B^{-1} p + \max_{g \in \partial J(w)} g^T p. \tag{27}
\]

However, this is generally intractable due to the presence of the supremum over the entire subdifferential \( \partial J(w) \). Therefore, we propose a bundle-based descent direction finding procedure (Algorithm 2), which progressively approaches to \( M(p) \) from below by a series of convex functions: \( M^1(p), \ldots, M^i(p) \) (10), each taking the same form as \( M(p) \) except the supremum is defined over a countable subset of \( \partial J(w) \). For instance, at step \( i \) the convex lower bound \( M^i(p) \) takes the form

\[
M^i(p) := \frac{1}{2} p^T B^{-1} p + \max_{g \in V^i} g^T p, \text{ where } V^i := \{g^j : j \leq i, i, j \in \mathbb{N}\} \subseteq \partial J(w). \tag{28}
\]

Successively minimizing such convex lower bounds yields a series of trial search directions: \( p^1, \ldots, p^i \). It is worth noting that the definition (12) enforces that any subgradient \( g^j \in V^i \) (28) with \( 1 < j \leq i \) is provided by an oracle\footnote{\( g^1 \) is randomly chosen from \( \partial J(w) \).} such that

\[
g^j := \arg\max_{g \in \partial J(w)} g^T p^{i-1}. \tag{29}
\]

In this case, we not only make \( M^i(p) \) coincide with \( M(p) \) at \( p^{i-1} \), i.e.,

\[
M^i(p^{i-1}) = M(p^{i-1}), \tag{30}
\]

but also ensure the convex lower bounds on \( M(p) \) is tightened over steps, i.e., \( M^i(p) \geq M^j(p) \forall j \leq i \). In the following, we refer the subgradients given by (29) as \textit{violating subgradients}. The iterative direction finding procedure continuous until the stopping criteria is satisfied. Later, we will show our stopping criteria guarantees the best solution so far, \( \arg\min_{j \leq i} M_i(p^j) \), to be a descent direction (9), and hence, suitable for the parameter update (3).

We now show that at certain step \( i \) how we can construct a trial search direction \( p^i \) which minimizes \( M^i(p) \). Ideally, we would like to find the best solution \( \arg\max_{g \in \partial J(w)} M^i(p) \). To do so, we rewrite \( \inf_{p \in \mathbb{R}^d} M^i(p) \) as a constrained optimization problem (13), and work in the dual space. First of all, we construct the Lagrangian of (13):

\[
L^i(p, \alpha, \xi) := \frac{1}{2} p^T B^{-1} p + \xi - \alpha^T (\xi 1 - G^i p), \tag{31}
\]

where \( \alpha := [\alpha^1, \alpha^2, \ldots, \alpha^i] \) is the column vector of non-negative Lagrange multipliers and \( G^i := [g^1, g^2, \ldots, g^i] \in \mathbb{R}^{d \times i} \) collects past violating subgradients (29), which are involved in the constrains of (13). Setting the derivative of (31) w.r.t. \( \xi \) and \( p \) to 0, respectively, gives

\[
\alpha^T1 = 1 \quad \text{and} \quad p = -BG^i \alpha, \tag{32}
\]

where the primal and the dual solutions, \( p \) and \( \alpha \), respectively, are related via the dual connection (33). To eliminate primal variables \( \xi \) and \( p \), we plug (32) and (33) back into (31) to obtain the dual of \( M^i(p) \):

\[
D^i(\alpha) := -\frac{1}{2} (G^i \alpha)^T B(G^i \alpha), \tag{34}
\]

s.t. \( \alpha \in [0,1]^i, \|\alpha\|_1 = 1 \).

Note maximizing the dual objective \( D^i(\alpha) \) (resp. minimizing the primal objective \( M^i(p) \)) can be done exactly via quadratic programming. However, doing so may incur substantial computational expense. In the sequel we show that we can adopt an alternative approach which does not maximize \( D^i(\alpha) \), but always guarantees an improvement in the dual objective value.

Let \( \alpha^i \in [0,1]^i \) be our solution for \( D^i(\alpha) \).\footnote{It is obvious we can set \( \alpha^i = [1] \) such that it is feasible for \( D^i(\alpha) \) given any initial subgradient matrix \( G^i = [g^i] \) with \( g^i \in \partial J(w) \).} Then, given \( \alpha^i \) we can construct a solution \( p^i := -BG^i \alpha^i \) for \( M^i(p) \) through the dual connection (33). In turn, a new violating subgradient:

\[
g^{i+1} := \arg\max_{g \in \partial J(w)} g^T p^i \tag{35}
\]

can be found for the construction of a tighter convex lower bound \( M^{i+1}(p) \) on \( M(p) \) (27) in the primal. Correspondingly, a new objective function \( D^{i+1}(\alpha) \) is then constructed as the dual of \( M^{i+1}(p) \):

\[
D^{i+1}(\alpha) := -\frac{1}{2} (G^{i+1} \alpha)^T B(G^{i+1} \alpha), \tag{36}
\]

s.t. \( \alpha \in [0,1]^{i+1}, \|\alpha\|_1 = 1 \),

where the subgradient matrix is extended with the new violating subgradient:

\[
G^{i+1} = [G^i, g^{i+1}], \tag{37}
\]
To construct a feasible solution \( \alpha \in [0,1]^{i+1} \) for \( D^{i+1}(\alpha) \), we constrain it to take the following form:

\[
\alpha := [(1 - \mu)\bar{\alpha}^i; \mu], \quad \mu \in [0,1]. \tag{38}
\]

Under the constrain (38), the task of maximizing \( D^{i+1}(\alpha) \) (36) reduces to maximizing a one-dimensional function \( D^{i+1} : [0,1] \to \mathbb{R} : \)

\[
\begin{aligned}
\bar{D}^{i+1}(\mu) &= -\frac{1}{2} (G^{i+1}\alpha)^\top B (G^{i+1}\alpha) \\
&= -\frac{1}{2} ((1-\mu)\bar{g}^i + \mu g^{i+1})^\top B ((1-\mu)\bar{g}^i + \mu g^{i+1}),
\end{aligned} \tag{39}
\]

where

\[
\bar{g}^i := G^i \alpha^i \in \partial J(w) \tag{40}
\]

lies in the convex hull of \( g^j \in \partial J(w) \ \forall j \leq i \) (and hence lies in the convex set \( \partial J(w) \)) due to \( \alpha^i \in [0,1]^i \) and \( \|\alpha^i\|_1 = 1 \). Moreover, \( \mu \in [0,1] \) ensures the feasibility of the dual solution. Noting that \( \bar{D}^{i+1}(\mu) \) is a concave quadratic function, we set

\[
\partial \bar{D}^{i+1}(\mu) = (\bar{g}^i - g^{i+1})^\top B ((1-\eta)\bar{g}^i + \eta g^{i+1}) = 0 \tag{41}
\]

to obtain the optimum

\[
\mu^* := \text{argmax}_{\mu \in [0,1]} \bar{D}^{i+1}(\mu) = \min \left( 1, \max \left( 0, \frac{(\bar{g}^i - g^{i+1})^\top B \bar{g}^i}{(\bar{g}^i - g^{i+1})^\top B (\bar{g}^i - g^{i+1})} \right) \right). \tag{42}
\]

Our dual solution is, then, construct via

\[
\alpha^{i+1} := [(1 - \mu^*)\bar{\alpha}^i; \mu^*]. \tag{43}
\]

Furthermore, from (37) and (38) it follows that \( \bar{g}^i \) can be maintained via an incremental update (Step 6 of Algorithm 2):

\[
\bar{g}^{i+1} := G^{i+1} \alpha^{i+1} = (1 - \mu^*)\bar{g}^i + \mu^* g^{i+1}, \tag{44}
\]

which combined with the dual connection (33) gives an update for the primal solution (Step 7 of Algorithm 2):

\[
P^{i+1} := -B\bar{g}^{i+1} = -(1 - \mu^*)B\bar{g}^i - \mu^* Bg^{i+1} = (1 - \mu^*)P^i - \mu^* Bg^{i+1}. \tag{45}
\]

This indicates that to derive a primal solution (Step 5–7 of Algorithm 2), it costs a total of \( O(d^2) \) time (resp. \( O(mld) \) time for the memory-limited variant with memory buffer size \( m \)), \( d \) being the dimension of the optimization problem. Note that, in general, maximizing \( D^{i+1}(\alpha) \) directly results in a larger progress than that obtained by our approach, i.e.,

\[
0 \leq D^{i+1}(\alpha^{i+1}) - D^{i+1}(\alpha^i) \leq \max_{\alpha \in [0,1]^{i+1}} D^{i+1}(\alpha) - D^{i+1}(\alpha^i). \tag{46}
\]

In the following we define a measure that indicates the quality of the dual solution at step \( i \):

\[
\epsilon^i := \min_{j \leq i} M^{j+1}(P^j) - D^j(\alpha^j) = \min_{j \leq i} M(P^j) - D^j(\alpha^j), \tag{47}
\]

where the second equality follows directly from (30). Let \( D(\alpha) \) be the corresponding dual problem of \( M(p) \) (27) with the property \( D(\alpha^i; 0; \cdots; 0) = D^i(\alpha^i); \) and denote by \( \alpha^* \) the optimal solution to max\(\alpha \in \mathcal{A} D(\alpha) \), where \( \mathcal{A} \) is some domain of interest. Then, the weak duality theorem suggests that \( \min_{p \in \mathbb{R}^d} M(p) \geq D(\alpha^*) \). By the definition of \( \epsilon^i \), we get

\[
\epsilon^i \geq \min_{p \in \mathbb{R}^d} M(p) - D^i(\alpha^i) \geq D(\alpha^*) - D^i(\alpha^i) = D(\alpha^*) - D(\alpha^i) - D(\alpha^i) \geq 0. \tag{48}
\]

This means \( \epsilon^i \) upper bounds the distance from the optimal dual objective value, \( D(\alpha^*) \). In fact, Theorem A.3 shows that \( \epsilon^i - \epsilon^{i+1} \) is lower bounded from 0, i.e., \( \epsilon^i \) is monotonically decreasing. Hence, we can stop the direction finding procedure at step \( i \) once \( \epsilon^i \) falls below a pre-specified tolerance \( \epsilon \). Note that calculating \( \epsilon^i \) naively requires the inversion of a matrix, i.e., \( B^{-1} \). Instead, we employ the dual connection (33), and calculate it via

\[
\epsilon^i = \min_{j \leq i} P^j \top g^{j+1} - \frac{1}{2} (P^j \top \bar{g}^j + P^j \top \bar{g}^j). \tag{49}
\]

Furthermore, since continuous progress in the dual objective does not necessarily prevent an increase in the primal objective: \( M(p^{i+1}) \geq M(p^i) \), we choose the best so far primal solution, \( p := \text{argmin}_{j \leq i} M(p^j) \), as the search direction (Step 15 of Algorithm 2) for the parameter update (3). It is easy to check that \( p \) is a descent direction as long as \( p^i \) is one.

In what follows, we prove our main theorems via several technical intermediate steps.

**Lemma A.1** Denote by \( \bar{D}^{i+1}(\mu) \) the one-dimensional concave quadratic function as defined in (39), \( \alpha^i \) a feasible solution for the dual objective \( D^i(\alpha) \), and \( p^i \) the primal solution deduced from \( \alpha^i \) via the dual connection (33). Then,

\[
\epsilon^i \leq \partial \bar{D}^{i+1}(0). \tag{50}
\]
Proof Since $p^i$ is related to $\alpha^i$ via the dual connection (33), we have
\[ p^i = -B\bar{g}^i, \quad \text{where} \quad \bar{g}^i = G^i \alpha^i. \] (51)
It follows from the definition of $M(p)$ (27) that
\[ M(p^i) = \frac{1}{2} p^i \top B^{-1} p^i + p^i \top g^{i+1}, \] (52)
where
\[ g^{i+1} := \arg\max \ g^\top p^i. \] (53)
Using (51), we have $B^{-1} p^i = -B\bar{g}^i = -g^i$, which implies
\[ M(p^i) = p^i \top g^{i+1} - \frac{1}{2} p^i \top g^i. \] (54)
Similarly, we have
\[ D^i(\alpha^i) = -\frac{1}{2}(G^i \alpha^i)^\top B(G^i \alpha^i) = \frac{1}{2} p^i \top g^i. \] (55)
From (41) and (51) it follows that
\[ \partial D^{i+1}(0) = (\bar{g}^i - g^{i+1}) \top B\bar{g}^i = (g^{i+1} - g^i) \top p^i. \] (56)
where $g^{i+1}$ is a violating subgradient chosen via (35), and hence coincides with (53). Using (54), (55) and (56), we get
\[ M(p^i) - D^i(\alpha^i) = (g^{i+1} - g^i) \top p^i = \bar{D}^{i+1}(0). \] (57)
Finally, by the definition of $\epsilon^i$ we have
\[ \bar{D}^{i+1}(0) = M(p^i) - D^i(\alpha^i) \geq \min_{j \leq i} M(p^j) - D^j(\alpha^j) = \epsilon^i. \] (58)
\[ \square \]

Lemma A.2 Denote by $f : [0, 1] \to \mathbb{R}$ a concave quadratic function with $f(0) = 0$, $\partial f(0) \in [0, h]$, and $\partial f^2(x) \geq -h$ for some $h \geq 0$. Then for all $x \in [0, 1]$ we have $\max_{x \in [0, 1]} f(x) \geq \frac{(\partial f(0))^2}{2h}$. 

Proof Using a second-order Taylor expansion around 0, we have $f(x) \geq \partial f(0)x - \frac{1}{2} x^2$. $x^* = \partial f(0)/h$ is the unconstrained maximum of the lower bound. Since $\partial f(0) \in [0, h]$, we have $x^* \in [0, 1]$. Plugging $x^*$ into the lower bound yields $(\partial f(0))^2/(2h)$. \[ \square \]

Theorem A.3 Assume that at iterate $w$, the convex objective function $J : \mathbb{R}^d \to \mathbb{R}$ has bounded subgradient, i.e., $\|\partial J(w)\| \leq G$. Also assume that $B$ is bounded, i.e., $\|B\| \leq H$. Then,
\[ \epsilon^i - \epsilon^{i+1} \geq \frac{(\epsilon^i)^2}{\sigma^2 H}. \] (59)
Proof Recall that we constrain the form of feasible dual solutions for $D^{i+1}(\alpha)$ (34) to $\{ (1 - \mu) \alpha^i : \mu \}$. Thus, instead of $D^{i+1}(\alpha)$, we work with the one-dimensional concave quadratic function of $\mu$: $D^{i+1}_\mu : [0, 1] \to \mathbb{R}$ (39) with $D^{i+1}_\mu(\mu) = D^{i+1}(\mu \alpha^i, \mu)$. It is obvious that $[\alpha^i ; 0]$ is a feasible solution for $D^{i+1}(\alpha)$. In this case, $D^{i+1}(0) = D^{i}(\alpha^i)$. It follows from (43) that $D^{i+1}_\mu(\mu^*) = D^{i+1}(\alpha^i)$. Therefore, using the definition of $\epsilon^i$ (47), we have
\[ \epsilon^i - \epsilon^{i+1} \geq D^{i+1}(\alpha^i) - D^{i}(\alpha^i) = \bar{D}^{i+1}(\mu^*) - \bar{D}^{i+1}(0). \] (60)
It is easy to see that $(\epsilon^i - \epsilon^{i+1})$ upper bounds the maximal value of the concave quadratic function: $f(\mu) := D^{i+1}(\mu) - D^{i+1}(0) \mu \in [0, 1]$ and $f(0) = 0$. Furthermore, from the definitions of $\bar{D}^{i+1}(\mu)$ and $f(\mu)$ it follows that
\[ \partial f(0) = \partial \bar{D}^{i+1}(0) = (\bar{g}^i - g^{i+1}) \top B\bar{g}^i \] (61)
\[ \partial^2 f(\mu) = \partial^2 \bar{D}^{i+1}(\mu) = -(\bar{g}^i - g^{i+1}) \top B(\bar{g}^i - g^{i+1}). \]
Since $\|\partial J(w)\| \leq G$ and $\bar{g}^i \in \partial J(w)$ (40), we have $\|\bar{g}^i - g^{i+1}\| \leq 2G$. Our assumption on $\|B\|$, then, gives $|\partial f(0)| \leq 2G^2 H$ and $|\partial^2 f(\mu)| \leq 4G^2 H$. Additionally, inequality (58) and the fact that $B > 0$ imply
\[ \partial f(0) = \partial \bar{D}^{i+1}(0) \geq 0 \] and
\[ \partial^2 f(\mu) = \partial^2 \bar{D}^{i+1}(\mu) < 0, \] (62)
which means
\[ \partial f(0) \in [0, 2G^2 H] \subset [0, 4G^2 H] \quad \text{and} \quad \partial^2 f(\mu) > -4G^2 H. \] (63)
Invoking Lemma A.2, we immediately get
\[ \epsilon^i - \epsilon^{i+1} \geq \frac{(\partial f(0))^2}{\sigma^2 H} = \frac{(\partial \bar{D}^{i+1}(0))^2}{\sigma^2 H}. \] (64)
Since $\epsilon^i \leq \bar{D}^{i+1}(0)$ by Lemma A.1, the inequality in (64) still holds when $\partial \bar{D}^{i+1}(0)$ is substituted with $\epsilon^i$. \[ \square \]

Note (61) and (62) imply that the optimal combination coefficient $\mu^*$ (42) has the following property:
\[ \mu^* = \min \left( 1, \frac{-\partial \bar{D}^{i+1}(0)}{-\partial^2 \bar{D}^{i+1}(\mu)} \right). \] (65)
Moreover, using (33), we can set $B\bar{g}^i$ in (42) to be $p^i$ to reduce the computational cost.

To prove the convergence of Algorithm 2, we use the following lemma, which is given as Sublemma 5.4, and proven by induction in "Polynomial Learnability of Stochastic Rules with Respect to the KL-Divergence and Quadratic Distance" by Abe et al. (IEICE Transactions on Information and Systems, 2001).

**Lemma A.4** Let $\{e^1, e^2, \ldots, e^i\} \in \mathbb{N}$ be a sequence of non-negative numbers satisfying the following recurrence:

$$e^i - e^{i+1} \geq c (e^i)^2,$$

where $c \in \mathbb{R}_+$ is a positive constant. Then, for all $i \geq 1$, we have

$$e^i \leq \frac{1}{c (t + \frac{1}{c^2})}.$$  \hspace{1cm} (67)

We are now ready to show in Theorem A.5 that Algorithm 2 decreases $e^i$ to a pre-defined tolerance $\epsilon$ in $O(1/\epsilon)$ steps.

**Theorem A.5** Under the assumptions of Theorem A.3 Algorithm 2 converges to the desired precision $\epsilon$ after

$$n \leq \frac{8G^2H}{\epsilon} - 4$$

steps for any $\epsilon < 2G^2H$.

**Proof** It follows from Theorem A.3 that

$$\epsilon^i - \epsilon^{i+1} \geq \frac{(\epsilon^i)^2}{8G^2H},$$

where $\epsilon^i \forall i \in \mathbb{N}$ is non-negative by (48). Thus, applying Lemma A.4, we get

$$\epsilon^i \leq \frac{1}{c (t + \frac{1}{c^2})},$$

where $c := \frac{1}{8G^2H}$.

Our assumptions on $\|\partial J(w)\|$ and $\|B\|$ imply $D^{i+1}(0) = (g^i - g^{i+1})^\top B\bar{g}^i \leq 2G^2H$ (63). Hence, $\epsilon^i \leq 2G^2H$ by Lemma A.1. This means the inequality (70) holds with $\epsilon^1 = 2G^2H$. Therefore, we can solve the following equation

$$\epsilon = \frac{1}{c (t + \frac{1}{c^2})},$$

with $c := \frac{1}{8G^2H}$ and $\epsilon^1 := 2G^2H$,

$$t = \frac{8G^2H}{\epsilon} - 4.$$ \hspace{1cm} (71)

to obtain an upper bound $t$ on the number of steps $n$ such that $\epsilon^n < \epsilon < 2G^2H$. The solution to (71) is $t \approx \frac{8G^2H}{\epsilon} - 4$. Hence, we prove our claim.

**B. Positive Step Size**

To formally show that there exists a positive step size satisfying the subgradient Wolfe conditions (15) we need Theorem 2.3.3 of Hiriart-Urruty and Lemaréchal (1993), which we restate in a slightly modified form:

**Lemma B.1** Given two points $w \neq w'$ in $\mathbb{R}^d$ define $w_\eta = \eta w' + (1 - \eta)w$. Let $J : \mathbb{R}^d \rightarrow \mathbb{R}$ be convex. There exists $\eta \in (0, 1)$ such that

$$J(w') - J(w) \leq \sup_{g \in \partial J(w_\eta)} g^\top (w' - w).$$

**Theorem B.2** Suppose $\Phi(\eta) := J(w_t + \eta p_t)$ which we restate in a slightly modified form:

$$J(w_t + \eta p_t) \leq \eta \sup_{g \in \partial J(w_t + \eta p_t)} g^\top p_t.$$  \hspace{1cm} (74)

**Proof** Since $p_t$ is a descent direction, the line $J(w_t) + c_1 \eta \sup_{g \in \partial J(w_t)} g^\top p_t$ with $c_1 \in (0, 1)$ must intersect $\Phi(\eta)$ at least once at some $\eta > 0$ (see Figure 3 for geometric intuition). Let $\eta'$ be the smallest such intersection point; then

$$J(w_t + \eta' p_t) = J(w_t) + c_1 \eta' \sup_{g \in \partial J(w_t)} g^\top p_t.$$  \hspace{1cm} (73)

Since $\Phi(\eta)$ is lower bounded, the first condition in (15) holds for all $\eta'' \in [0, \eta']$.

Setting $w = w_t$ and $w' = w_t + \eta' p_t$ in lemma B.1, it follows that there exists $\eta'' \in (0, \eta')$ such that

$$J(w_t + \eta' p_t) - J(w_t) \leq \eta' \sup_{g \in \partial J(w_t + \eta'' p_t)} g^\top p_t.$$  \hspace{1cm} (74)

Plugging in (73) and simplifying yields

$$c_1 \sup_{g \in \partial J(w_t)} g^\top p_t \leq \sup_{g \in \partial J(w_t + \eta'' p_t)} g^\top p_t.$$ \hspace{1cm} (75)

Since $p_t$ is a descent direction it follows that $\sup_{g \in \partial J(w_t)} g^\top p_t < 0$, and hence the above inequality also holds when $c_1$ is replaced by $c_2 \in (c_1, 1)$.