Online Optimization with Barzilai-Borwein Quasi-Newton Method

Iyanuoluwa Emiola (Student Member, IEEE) and Chinwendu Enyioha

Abstract—This paper considers the online case for the Barzilai-Borwein quasi-Newton method and presents a regret analysis. To solve online convex optimization problems, sequential decisions are made at each time-step using some algorithm of choice. We use a greedy online gradient algorithm based on Barzilai-Borwein (BB) step sizes and show that the regret obtained from our algorithm is sublinear in time and that the average regret approaches zero. Analysis is presented for the two forms of the Barzilai-Borwein step sizes.

I. INTRODUCTION

This paper presents a gradient-based algorithm using the Barzilai-Borwein step sizes to solve an online optimization problem. In an online optimization problem, the objective of the online agent is to make a sequence of accurate decisions given knowledge of the optimal solution to previous decisions. The problem of online optimization has applications to a number of fields including game theory, the smart grid and classification in machine learning amongst others. Performance of online optimization algorithms is usually measured in terms of the aggregate regret suffered by the online agent compared with the known optimal solution of each problem across the sequence of problems.

Online optimization methods and algorithms have been studied using different methods including gradient-based methods [19, 22, 10]. Extensions have been considered on unconstrained problems [14] and online problems with long-term [13]. Problems in dynamic environments have also been analyzed [15]. As well-structured as gradient methods are, applying them to large-scale online problems face several challenges and become impractical due to their well-known slow convergence rates in the static settings [2]. To address the slow convergence rates of first order methods, second-order (popularly called Newton-type) methods have been proposed [17]. While Newton-type iterative methods have quadratic convergence, they also present a significant computational overhead from the need to invert and store the hessian of the objective function being optimized, which makes them impractical for large-scale online optimization problems.

To leverage the benefits of the computational simplicity of gradient methods and the convergence properties of second-order methods, the so-called quasi-Newton methods have been introduced; for example, the Broyden-Fletcher-Goldfarb-Shanno (BFGS) algorithm [12, 8] and the Barzilai-Borwein (BB) algorithm [11, 6]. Quasi-Newton methods exploit the second-order (curvature information) of the objective function being optimized into the first-order frame-work. For example, the BFGS method approximates the information in the curvature of the hessian between time steps to use in its update, though scaling is a known issue [11]. The Stochastic BFGS and its low-memory variant (the L-BFGS) quasi-Newton method has been studied in online settings [17, 3] with good performance relative to the standard gradient method. The BB method, on the other hand, computes a step size such that the computed step size and gradient contain information that approximates the hessian curvature. Convergence rate analyses have been obtained for these quasi-Newton methods [4, 5] and these methods are increasingly being used in large-scale, computation-intensive applications such as distributed learning.

In this paper, we present an online Barzilai-Borwein quasi-Newton algorithm and analyze its performance for the two variations of the BB step sizes using the regret. We show that the regret increases sublinearly in time. Following an introduction of the problem and brief summary of existing approaches (Section II), we introduce quasi-Newton methods that exploit known fast convergence of second-order methods (Section III) and present our main result (Section IV). Concluding remarks follow in Section V.

Notation: Vectors and matrices are represented by lower and upper case letters, respectively. We denote a vector or matrix transpose as $(\cdot)^T$, and the L2-norm of a vector by $\|\cdot\|$. The gradient of a function $f(\cdot)$ is denoted $\nabla f(\cdot)$, and we respectively denote the set of reals numbers as $\mathbb{R}$.

II. PROBLEM FORMULATION

Consider an online optimization problem

$$\min_{x(k) \in \mathcal{X}} f_k(x(k)), \tag{1}$$

in which the feasible decision set $\mathcal{X} \subset \mathbb{R}^n$ is known, assumed to be convex, non-empty, bounded, closed and fixed for all time $k = 1, \ldots, K$. We assume the number of iterations during which the online players make choices, $K$, is unknown to the player. By convexity of the cost function $f_k(\cdot)$ and $\mathcal{X}$, Problem (1) has an optimal solution $x^*$, which is the best possible choice or decision agents can make at each time $k$. A player (an online agent) at time $k$ uses some algorithm to choose a point $x(k) \in \mathcal{X}$, after which the player receives a loss function $f_k(\cdot)$. The loss incurred by the player is $f_k(x(k))$. These problems are common in contexts such as real time resource allocation, online classification [10]. The goal of the online agent is to minimize the aggregate loss by determining a sequence of feasible online solutions $x(k)$ at each time-step of the algorithm.

This project was supported by a Southeastern Center for Electrical Engineering Education (SCEEE) Young Investigator Grant.

The authors are with the Electrical & Computer Engineering Department at the University of Central Florida, Orlando FL 32816, USA. Email: iemiola@knights.ucf.edu and cenyioha@ucf.edu
Let the aggregate loss incurred by the online algorithm that solves Problem (1) at time $K$ be given by:

$$f(K) = \sum_{k=1}^{K} f_k(x(k)).$$

To measure performance of the online player, we use the regret framework. The static regret is a measure of the difference between the loss of the online player and the loss from the static case

$$\min_{x \in \mathcal{X}} f_k(x),$$

where the single best decision $x^*$ is chosen with the benefit of hindsight. Let the aggregate loss up to time $K$ incurred by the single best decision be given by

$$f_x(K) = \sum_{k=1}^{K} f_k(x).$$

Then the static regret at time $K$ is defined as [10]:

$$R(K) = f(K) - \min_{x} f_x(K). \quad (2)$$

A. Algorithms for Online Optimization Problem

A commonly used algorithm for solving the static case of Problem (1) is the gradient descent method, which involves updating the variable $x(k)$ iteratively using the gradient of the cost function with the following equation:

$$x(k + 1) = x(k) - \alpha \nabla f(x(k)). \quad (3)$$

It is known that with an appropriate choice of the step size $\alpha$, the sequence $\{x(k)\}$ converges to $x^*$ in $O(1/k)$; that is, an $\varepsilon$-optimal solution is attained in about $O(1/\varepsilon^2)$ iterations [16]. In fact, when the $f(\cdot)$ is strongly convex, the iterative update in Equation (3) converge faster to the optimal solution. Even though the update scheme of gradient method are simple and easily implementable in a distributed architecture, convergence is slow [16]. Techniques to accelerate convergence exists though lag behind the Newton and quasi-Newton methods [21].

To improve convergence rates in static optimization problems, algorithms that use second order information (hessian of the cost function) have been introduced. These methods leverage curvature information of the cost function in addition to direction; and are known to speed up the convergence in the neighborhood of the optimal solution. The Newton-type methods have an update of the form:

$$x(k + 1) = x(k) - \nabla^2 f(x) \nabla f(x). \quad (4)$$

In fact, when $f$ is quadratic, the Newton algorithm is known to converge in one time-step. Though they have good convergence properties, there are computational costs associated with building and computing the inverse hessian. In addition, some modification are needed if the hessian is not positive definite [9]. To avoid the computation burden of second-order methods while maintaining the structure of first-order methods, quasi-Newton methods have been introduced.

B. Quasi-Newton Methods

A number of quasi-Newton methods have been proposed in the literature including the Broyden-Fletcher-Goldfarb-Shanno (BFGS) algorithm [8] and the Barzilai-Borwein (BB) algorithm [7], as well as the David-Fletcher-Powell (DFP) algorithm [20]. The central idea in the performance of these methods is to speed up convergence by exploiting the information from the inverse hessian without necessarily computing it explicitly; for example, Barzilai-Borwein computes step-sizes using the difference of successive iterates and the gradient evaluated at those iterates. In this paper, we use the gradient-based method using Barzilai-Borwein step sizes to solve Problem (1) and show that the regret increases sublinearly in time.

III. THE BARZILAI-BORWEIN QUASI-NEWTON METHOD

The Barzilai-Borwein quasi-Newton method is an iterative technique suitable for solving optimization problems that can yield superlinear convergence rates when the objective functions are strongly convex and quadratic [1][5]. It differs from other quasi-Newton methods because it only uses one step size for the iteration as opposed to other quasi-Newton methods that have more computation overhead. The Barzilai-Borwein method solves Problem (1) iteratively using the update in (3); however, the step-size $\alpha(k)$ is computed so that $\alpha(k)\nabla f(x(k))$ approximates the the inverse Hessian of the Newton update (4). We briefly introduce the two forms of the BB step-sizes used in Algorithm (1).

Consider the update $x(k + 1) = x(k) - \alpha(k)\nabla f(x(k))$. To compute $\alpha(k)$, we introduce two variables $s(k)$ and $y(k)$ expressed as

$$s(k - 1) \triangleq x(k) - x(k - 1), \quad \text{and} \quad y(k - 1) = \nabla f(x(k)) - \nabla f(x(k - 1)).$$

Then the hessian of $f(\cdot)$ satisfies

$$\nabla^2 f(x(k))s(k - 1) = y(k - 1).$$

The objective is to pick $\alpha(k)$ such that

$$(\alpha(k)^{-1}I)s(k - 1) \approx y(k - 1), \quad (5)$$

where $I$ is the identity matrix of appropriate dimension. Two different step sizes $\alpha(k)$ that satisfy Equation (5), are typically derived from solving two formulations of a least squares problem obtained from (5). If we let $\gamma = \alpha^{-1}$, the first step size $\alpha_1(k)$ is obtained by solving the problem

$$\alpha(k)^{-1} = \arg \min_{\gamma} \frac{1}{2} \|s(k - 1)\gamma - y(k - 1)\|^2$$

for $\alpha(k)$, from which we obtain

$$\alpha_1(k) = \frac{s(k - 1)^T s(k - 1)}{s(k - 1)^T y(k - 1)}. \quad (6)$$

Similarly, the second step size, $\alpha_2(k)$, is obtained from solving the least square problem:

$$\alpha(k)^{-1} = \arg \min_{\alpha} \frac{1}{2} \|s(k - 1) - y(k - 1)\alpha\|^2,$$
from which we obtain the following expression
\[ \alpha_2(k) = \frac{s(k-1)^T y(k-1)}{y(k-1)^T y(k-1)}. \] (7)

In general, there is flexibility in the choice to use \( \alpha_1(k) \) or \( \alpha_2(k) \) \([1]\), and both step sizes can be alternated within the same algorithm after a considerable amount of iterations to facilitate convergence. The rest of this work will characterize performance of the online Algorithm \([1]\) using the step sizes in Equations \((6)\) and \((7)\), which as we will show has a regret that is sublinear in time with the average regret approaching zero.

Before stating the main result, we state some assumptions about Problem \([1]\) and Algorithm \([1]\).

**Assumption 1.** The decision set \( X \) is bounded. This implies that there exists some constant \( 0 \leq B < \infty \) such that \( |X| \leq B \).

**Assumption 2.** The decision set \( X \) is closed; that is, suppose all agents’ decisions follow an iterative sequence \( x(k) \in X \). If there exists some \( \hat{x} \in \mathbb{R}^n \) such that \( \lim_{k \to \infty} x(k) = \hat{x} \), then \( \hat{x} \in X \).

**Assumption 3.** For all decision iterates \( x(k) \), the cost function \( f(x(k)) \) is differentiable and its derivative is also bounded; that is, \( |\nabla f_k(x(k))| \leq B < \infty \) \( \forall k \).

**Algorithm 1** Online Barzilai-Borwein Quasi-Newton Alg.

- **Given:** Feasible set \( X \) and time horizon \( K \)
- **Initialize:** \( x(0) \) and \( \nabla f_0(x(0)) \) arbitrarily
- **for** \( k = 1 \) to \( K \) **do**
  - Agents predicts \( x(k) \) and observes \( f_k(\cdot) \)
  - Update \( x(k+1) = x(k) - \alpha(k) \nabla f_k(x(k)) \)
- **end for**

**IV. REGRET BOUNDS**

Before we present our main results (Theorems \([1]\) and \([2]\)), we first present two lemmas that will be used in its proof. The first is a result in \([22]\), and the other is the Sedrakyan’s inequality.

**Lemma 1.** (\([22]\)) Without loss of generality, for all iterates \( k \), there exists gradient \( g_k(\cdot) \in \mathbb{R}^n \) such that for all \( x \), \( g_k(x) = \nabla f_k(x(k)) \). Where \( g_k(\cdot) \)?

**Proof.** Suppose \( x^* \) is an optimal vector, we equivalently obtain:
\[ f_k(x^*) \geq \nabla f_k(x(k)) (x^* - x(k)) + f_k(x(k)). \] (8)

Because \( x^* \) is an optimal vector, we equivalently obtain:
\[ f_k(x^*) \geq \nabla f_k(x(k)) (x^* - x(k)) + f_k(x(k)). \] (9)

Thereore according to \((8)\) and \((9)\), we obtain:
\[ f_k(x) - f_k(x^*) \leq \nabla f_k(x(k)) (x(k) - x^*) \]

**Lemma 2.** (The Sedrakyan’s Inequality) For all positive reals \( a_1, a_2, \ldots, a_n \) and \( b_1, b_2, \ldots, b_n \), the following inequality holds:
\[ \sum_{i=1}^{n} \frac{a_i^2}{b_i} \geq \left( \sum_{i=1}^{n} \frac{a_i}{b_i} \right)^2. \]

**Proof.** We refer readers to \([18]\) for a proof. \( \Box \)

Another result we will use is the generalized bounds for online gradient descent \([10]\):
\[ R(K) \leq D^2 \frac{1}{K} \alpha(K) + ||\nabla f(x(k))||^2 \sum_{k=1}^{K} \alpha(k), \] (10)

where \( D = ||x(k) - x^*||^2 \).

We will now proceed to characterize the regret obtained from Algorithm \([1]\) for Problem \([1]\) with the two BB step sizes.

**Theorem 1.** Consider Problem \([1]\) and let:
\[ \alpha(k) = \frac{s(k-1)^T s(k-1)}{s(k-1)^T y(k-1)} \]
in Algorithm \([1]\). Then the average regret is bounded by:
\[ \frac{R(K)}{K} \leq D^2 \frac{1}{K} \alpha(K) + ||\nabla f(x(k))||^2 \Gamma, \]

where \( \Gamma = \frac{||x(1)||^2}{L} + \frac{\sum_{i=1}^{T} ||x(i)||^2}{L} + L \sum_{i=1}^{K} ||x(k)||^2 \), and \( L = \max \frac{\alpha}{\|x(k)\|^2}, \frac{\alpha}{\|x(k)\|^2} \) and \( L_k \) is the Lipschitz parameter of \( \nabla f_k(x), \nabla f_k(x) \) in Problem \([1]\) and \( \lim_{K \to \infty} \frac{R(K)}{K} \leq 0. \)

**Proof.** First, by using the results of Lemma \([1]\) the regret of Algorithm \([1]\) can be expressed as:
\[ R(K) = \sum_{k=1}^{K} (x(k) - x^*) g(k). \]

Then from Equation \((4)\), the regret
\[ R(K) = \sum_{k=1}^{K} (x(k) - \alpha(k) - 1 \nabla f(x(k)) - x^*) g(k), \]

where \( \alpha(k) \) is as expressed in \((6)\). To prove Theorem \([1]\) the approach will be to upper-bound the aggregate sum of the step size \( \alpha(k) \) and use the generalized bound for online gradient descent in Equation \((10)\). This approach is possible since the gradient of the cost function at each time in the sequence of problems is bounded (Assumption \([3]\)). Proceeding, the running sum of the step sizes \( \alpha(k) \) up to time \( K \) is expressed as
\[ \sum_{k=1}^{K} \alpha(k) = \sum_{k=1}^{K} \frac{s(k-1)^T s(k-1)}{s(k-1)^T y(k-1)} = \sum_{k=1}^{K} \frac{(x(k) - x(k-1))^T (x(k) - x(k-1))}{(x(k) - x(k-1))^T (\nabla f(x(k)) - \nabla f(x(k-1)))} \]
\[ = \sum_{k=1}^{K} \frac{||x(k)-x(k-1)||^2}{(x(k)-x(k-1))^T (\nabla f(x(k)) - \nabla f(x(k-1)))}. \]
Apply the result in Lemma 2 to the right hand side of the preceding inequality, we obtain that:

$$
\sum_{k=1}^{K} \alpha(k) \geq \frac{\| \sum_{k=1}^{K} (x(k) - x(k-1)) \|^2}{\sum_{k=1}^{K} (x(k) - x(k-1))^T (\nabla f(x(k)) - \nabla f(x(k-1)))}
$$

We can bound the numerator of Equation (11) to become:

$$
\sum_{k=1}^{K} (x(k) - x(k-1))^2 \geq \| x(2) - x(1) + \ldots \|^2,
$$

and since it is a telescoping series, we obtain:

$$
\sum_{k=1}^{K} (x(k) - x(k-1))^2 \leq \| x(1) \|^2.
$$

To bound the denominator of Equation (11), we use the Lipschitz continuity of the gradients of \( f(\cdot) \) with parameter \( L > 0 \). Therefore,

$$
\sum_{k=1}^{K} (x(k) - x(k-1))^2 \leq \sum_{k=1}^{K} L \| x(k) - x(k-1) \|^2.
$$

By using the bounds on both the numerator and denominator of (11), we obtain:

$$
\sum_{k=1}^{K} \frac{\| x(k) - x(k-1) \|^2}{L \sum_{k=1}^{K} \| x(k) \|^2 + \sum_{k=1}^{K} \| x(k-1) \|^2} \leq \frac{\| x(1) \|^2}{L \sum_{k=1}^{K} \| x(k-1) \|^2}\]

By the triangle inequality, we obtain the bound for using the first BB step size as:

$$
\sum_{k=1}^{K} \alpha(k) \leq \frac{\| x(1) \|^2}{L \sum_{k=1}^{K} \| x(k) \|^2 + \sum_{k=1}^{K} \| x(k-1) \|^2}.
$$

By using the regret bound equation in (10), we obtain:

$$
R(K) \leq D^2 \frac{1}{\alpha(K)} + \| \nabla f(x(K)) \|^2 \Psi,
$$

where \( \Psi = \frac{\| x(1) \|^2}{L \sum_{k=1}^{K} \| x(k) \|^2 + \sum_{k=1}^{K} \| x(k-1) \|^2} \).

The average regret over \( K \) time steps can then be expressed as

$$
\frac{R(K)}{K} \leq D^2 \frac{1}{\alpha(K)} + \frac{\| \nabla f(x(K)) \|^2}{K} \Psi.
$$

Since \( D \) is constant based on its value in (10), and the derivatives \( \| \nabla f(x(k)) \|^2 \) for all time-steps \( k \) are also bounded, we conclude that that the average regret satisfies

$$
\lim_{K \to \infty} \frac{R(K)}{K} \leq 0.
$$

Next, we consider the performance of Algorithm 1 using the second BB step size in Equation (7).

**Theorem 2.** Consider Problem 1 and let Algorithm 1 be used to solve Problem 1 where \( \alpha(k) = \frac{s(k-1)^T y(k-1)}{y(k-1)^T y(k-1)} \), and \( L \) is the maximum of all Lipschitz continuity parameters of all gradients of the cost function in Problem 1. Then, the regret is upper bounded by

$$
R(K) \leq D^2 \frac{1}{\alpha(K)} + \| \nabla f(x(K)) \|^2 \zeta,
$$

where

$$
\zeta = \frac{(-x(1))^T \sum_{k=1}^{K} \| x(k) - x(k-1) \|^2}{L \sum_{k=1}^{K} \| x^2(k) + x^2(k-1) \|^2}.
$$

and the average regret

$$
\lim_{K \to \infty} \frac{R(K)}{K} \leq 0.
$$

**Proof.** The approach to proving Theorem 2 will be similar to that of Theorem 1 where we will obtain bounds for the aggregate sum of the step sizes in \( R(K) \) and use the generalized bound for online gradient descent algorithm. In this case, the sum of the aggregate step sizes is expressed as

$$
\sum_{k=1}^{K} \alpha(k) = \sum_{k=1}^{K} \frac{s(k-1)^T y(k-1)}{y(k-1)^T y(k-1)}
$$

By using the relationship

$$
s(k-1) \triangleq x(k) - x(k-1), \text{ and } y(k-1) = \nabla f(x(k)) - \nabla f(x(k-1)).
$$

and by noting that \( y(k-1)^T y(k-1) = \| y(k-1) \|^2 \), and also expressing as a product of three different functions, we obtain:

$$
\sum_{k=1}^{K} \alpha(k) = \sum_{k=1}^{K} \frac{(x(k) - x(k-1))^T}{\sum_{k=1}^{K} \| x(k) \|^2 + \sum_{k=1}^{K} \| x(k-1) \|^2} \| \nabla f(x(k)) - \nabla f(x(k-1)) \|^2
$$

For the purpose of clarity, let

$$
A(k) = (x(k) - x(k-1))
$$

$$
B(k) = (\nabla f(x(k)) - \nabla f(x(k-1))) \text{ and } C(k) = \| \nabla f(x(k)) - \nabla f(x(k-1)) \|^{-2}
$$

Applying the Cauchy-Schwarz inequality to the right hand side of Equation (12), we obtain that:

$$
\sum_{k=1}^{K} \| A(k)^T B(k) \| \| C(k) \| \leq \sum_{k=1}^{K} \| A(k)^T \| \sum_{k=1}^{K} \| B(k) \| \sum_{k=1}^{K} \| C(k) \|
$$

By telescoping series principle on the the term \( \sum_{k=1}^{K} \| A(k) \| \) term above, and using the fact that \( \| x(k) - x(k-1) \| = \| (x(k) - x(k-1))^T \| \), we obtain:
Therefore, we can conclude that
\[ \sum_{k=1}^{K} \|A(k)\| = \sum_{k=1}^{K} \|x(k) - x(k-1)\| \leq \|x(1)\|. \]

Using the Lipschitz continuity condition on the gradient of the cost function, a bound for the aggregate norm of the gradient difference term \(B(k)\) can be obtained as follows:
\[ \sum_{k=1}^{K} \|B(k)\| = \sum_{k=1}^{K} \|\nabla f(x(k)) - \nabla f(x(k-1))\| \leq L \sum_{k=1}^{K} \|x(k) - x(k-1)\| \]

Therefore, it follows that:
\[ \sum_{k=1}^{K} \|\nabla f(x(k)) - \nabla f(x(k-1))\|^2 \leq L^2 \sum_{k=1}^{K} \|x(k) - x(k-1)\|^2. \]

For all \(x(k) > 0, x(k-1) > 0\) this then implies that:
\[ \sum_{k=1}^{K} \|\nabla f(x(k)) - \nabla f(x(k-1))\|^2 \leq L^2 \sum_{k=1}^{K} \|x(k) - x(k-1)\|^2. \]

Therefore, we can conclude that
\[ \sum_{k=1}^{K} \alpha(k) \leq \frac{L(-x(1)) \sum_{k=1}^{K} \|x(k) - x(k-1)\|}{L^2 \sum_{k=1}^{K} \|x(k) - x(k-1)\|^2} \]
\[ \leq \frac{(-x(1)) \sum_{k=1}^{K} \|x(k) - x(k-1)\|}{L \sum_{k=1}^{K} \|x^2(k) + x^2(k-1)\|}. \]

Applying the generalized regret bound to Equation (13), we obtain the regret \(R(K)\) as:
\[ R(K) \leq D^2 \frac{1}{\alpha(K)} + \|\nabla f(x(k))\|^2 \zeta, \]
where
\[ \zeta = \frac{(-x(1)) \sum_{k=1}^{K} \|x(k) - x(k-1)\|}{L \sum_{k=1}^{K} \|x^2(k) + x^2(k-1)\|}. \]

Therefore the average regret is
\[ \frac{R(K)}{K} \leq D^2 \frac{1}{K \alpha(K)} + \frac{\|\nabla f(x(k))\|^2}{K} \zeta. \]

Furthermore, since \(D\) is constant based on its value in [10], and the terms \(\|x(k) - x(k-1)\|\) and \(\|x^2(k) + x^2(k-1)\|\) are also bounded, we conclude that the average regret satisfies
\[ \lim_{K \to \infty} \frac{R(K)}{K} \leq 0. \]

The Barzilai-Borwein step size in the gradient-based Algorithm [1] results in a regret that grows sublinearly in time and yields an average regret of zero as time \(K\) goes to infinity.

V. CONCLUSIONS

In this work, we presented the online Barzilai-Borwein quasi-Newton algorithm and analyzed the regret. The analysis for both Barzilai-Borwein step sizes showed that the regret of the algorithm grows sublinearly in time and that the average regret approaches zero. The use of the generalized regret bounds for online gradient descent introduced in [10] simplified the analyses.

REFERENCES

[1] Barzilai, J. and J. M. Borwein (1988). Two-point step size gradient methods. IMA journal of numerical analysis 8(1), 141–148.
[2] Boyd, S. and L. Vandenberghe (2004). Convex optimization. Cambridge university press.
[3] Byrd, R. H., S. L. Hansen, J. Nocedal, and Y. Singer (2016). A stochastic quasi-newton method for large-scale optimization. SIAM Journal on Optimization 26(2), 1008–1031.
[4] Byrd, R. H., J. Nocedal, and Y.-X. Yuan (1987). Global convergence of a class of quasi-newton methods on convex problems. SIAM Journal on Numerical Analysis 24(5), 1171–1190.
[5] Dai, Y.-H. (2013). A new analysis on the barzilai-borwein gradient method. Journal of the operations Research Society of China 1(2), 187–198.
[6] Dai, Y.-H. and R. Fletcher (2005). Projected barzilai-borwein methods for large-scale box-constrained quadratic programming. Numerische Mathematik 100(1), 21–47.
[7] Dai, Y.-H. and L.-Z. Liao (2002). R-linear convergence of the barzilai and borwein gradient method. IMA Journal of Numerical Analysis 22(1), 1–10.
[8] Eisen, M., A. Mokhtari, and A. Ribeiro (2017). Decentralized quasi-newton methods. IEEE Transactions on Signal Processing 65(10), 2613–2628.
[9] Gill, P. E. and W. Murray (1972). Quasi-newton methods for unconstrained optimization. IMA Journal of Applied Mathematics 9(1), 91–108.
[10] Hazan, E. et al. (2016). Introduction to online convex optimization. Foundations and Trends® in Optimization 2(3-4), 157–325.
[11] Ibrahimi, M. A. H. B., M. Mamat, L. W. June, A. Zaidi, and M. Sohi. The scaling of bfgs-sd in solving unconstrained optimization problem.
[12] Li, D.-H. and M. Fukushima (2001). On the global convergence of the bfgs method for nonconvex unconstrained optimization problems. SIAM Journal on Optimization 11(4), 1054–1064.
[13] Mahdavi, M., R. Jin, and T. Yang (2012). Trading regret for efficiency: online convex optimization with long term constraints. Journal of Machine Learning Research 13(9), 2503–2528.
[14] McMahan, B. and M. Streeter (2012). No-regret algorithms for unconstrained online convex optimization. In Advances in neural information processing systems, pp. 2402–2410.
[15] Mokhtari, A., S. Shahrampour, A. Jadbabaie, and A. Ribeiro (2016). Online optimization in dynamic environments: Improved regret rates for strongly convex problems. In 2016 IEEE 55th Conference on Decision and Control (CDC), pp. 7195–7201. IEEE.
[16] Nesterov, Y. (1998). Introductory lectures on convex programming volume i: Basic course. Lecture notes 3(4), 5.
[17] Schraudolph, N. N., J. Yu, and S. Günter (2007). A stochastic quasi-newton method for online convex optimization. In Artificial intelligence and statistics, pp. 436–443.
[18] Sedrakyan, H. and N. Sedrakyan (2018). Algebraic inequalities. Springer.
[19] Shalev-Shwartz, S. and S. M. Kakade (2009). Mind the duality gap: Logarithmic regret algorithms for online optimiza-
tion. In Advances in Neural Information Processing Systems, pp. 1457–1464.

[20] Sofi, A., M. Mamat, and M. Ibrahim (2013). Reducing computation time in dfp (davidon, fletcher & powell) update method for solving unconstrained optimization problems. In AIP Conference Proceedings, Volume 1522, pp. 1337–1345. AIP.

[21] Su, W., S. Boyd, and E. Candes (2014). A differential equation for modeling nesterovs accelerated gradient method: Theory and insights. In Advances in Neural Information Processing Systems, pp. 2510–2518.

[22] Zinkevich, M. (2003). Online convex programming and generalized infinitesimal gradient ascent. In Proceedings of the 20th International Conference on Machine Learning (ICML-03), pp. 928–936.