Two novel gradient methods with optimal step sizes

Harry Oviedo[a], Oscar Dalmau[b] and Rafael Herrera[c]

[a,b,c]Mathematics Research Center, CIMAT A.C. Guanajuato, Mexico

ARTICLE HISTORY
Compiled May 16, 2020

ABSTRACT
In this work we introduce two new Barzilai and Borwein-like steps sizes for the classical gradient method for strictly convex quadratic optimization problems. The proposed step sizes employ second-order information in order to obtain faster gradient-type methods. Both step sizes are derived from two unconstrained optimization models that involve approximate information of the Hessian of the objective function. A convergence analysis of the proposed algorithm is provided. Some numerical experiments are performed in order to compare the efficiency and effectiveness of the proposed methods with similar methods in the literature. Experimentally, it is observed that our proposals accelerate the gradient method at nearly no extra computational cost, which makes our proposal a good alternative to solve large-scale problems.

KEYWORDS
Gradient methods, convex quadratic optimization, Hessian spectral properties, steplength selection.

1. Introduction.

In this paper, we consider the classical convex quadratic minimization problem,

$$\min_{x\in\mathbb{R}^n} f(x) = \frac{1}{2}x^\top Ax - x^\top b \tag{1}$$

where $b \in \mathbb{R}^n$ and $A \in \mathbb{R}^{n\times n}$ is a symmetric and positive definite matrix.

One of the most studied algorithms to address unconstrained optimization problems is the *steepest descent method*, which was first introduced by Cauchy in 1847 [1]. This method is part of the so-called first-order methods, which only use information about the objective function and its derivative. It is well known that the steepest descent method is an ineffective method to solve (1) numerically, due to its slow convergence rate and zigzag pattern along the iterations. However, this method has a well-established convergence analysis that has been of great help to study other algorithms. In addition, the steepest descent method requires little memory per iteration. In spite of its drawbacks, in the 80's, a rebirth of the method arose due to the works of Iudin-Nemirowsky [2] and Nesterov [3] which proposed a technique of acceleration

Emails: harry.oviedo@cimat.mx, dalmau@cimat.mx and rherrera@cimat.mx

of the gradient method by using an extra-momentum step that can be used to address large-scale convex minimization problems. Additionally, in 1988 Barzilai and Borwein [4] introduced two new alternatives to select the step size in the *gradient method* for solving problem (1) that greatly accelerate the convergence rate of the method. In [5], Raydan developed an ingenious convergence study of the Barzilai-Borwein method. In addition, Dai et al. [6] demonstrated that this method has R-linear convergence rate.

Initially, some step sizes were proposed for the quadratic case (1) [4, 10–12, 16] and then extended to the non-linear case. An extension for the case of non-linear optimization of the gradient method with Barzilai-Borwein steps (BB-steps) size has been provided by Raydan et al. in [7]. Other extensions of the method are found in [8, 9]. From the seminal paper [4], many researchers have taken up the steepest decent study in order to design novel effective gradient-type methods by introducing new step sizes to address large-scale optimization problems efficiently. To date, several formulations have been proposed to select step size, which attempt to incorporate second-order information without increasing significatively the computational cost of the classical steepest descent method. For example, in [16] Yuan introduced an ingenious step size that was built by imposing finite termination for the two dimensional quadratic problem. In addition, Dai and Yuan in [10], Dai [11] proposed two step sizes that alternate the BB-steps and the exact step size of the classic steepest descent, in the odd and even iterations. Additionally, a new step size has been proposed very recently in [12], which uses the BFGS update formula to build a new approximate optimal step size.

In this paper we study two step sizes based on optimization models for the gradient methods. Our main contribution is to introduce and analyze two new choices of the step size which are obtained as solutions of such optimization models. Using these two-point step size we design an efficient gradient algorithm. In addition, we provide a convergence analysis of the proposed algorithm and some numerical experiments where we compare our algorithm with similar algorithms from the state of the art.

This paper is organized as follows. In the next section we briefly present the gradient method and some of its variants. Section 3 is dedicated to summarizing the step sizes proposed by Barzilai and Borwein. Our contribution of two new step sizes is introduced in Section 4. A convergence analysis is provided in Section 5. In Section 6 we present some numerical results to show the efficiency and effectiveness of the proposed step sizes. Finally, in Section 7 we present the general conclusions of this work.

2. Steepest Descent Method and Its Variants.

The gradient method to solve (1) is an iterative method that uses the following recursive formula starting from a given point x_0:

$$x_{k+1} = x_k - \frac{1}{\alpha_k} g_k, \tag{2}$$

where $g_k = \nabla f(x_k) = Ax_k - b$, $\alpha_k > 0$ and $1/\alpha_k$ is known as the step-size. There are several alternatives for the step-size; the most popular one is given by choosing the step size of the k-th iteration (α_k) as the positive scalar that minimizes the objective

function along the direction of the negative gradient, i.e.

$$\alpha_k = \arg\min_{\alpha>0} f(x_k - \frac{1}{\alpha} g_k). \tag{3}$$

It is not difficult to prove that the solution to (3) is given by

$$\alpha_k^{SD} = \frac{g_k^\top A g_k}{g_k^\top g_k}. \tag{4}$$

When the gradient method uses the step size (4), it is known as *steepest descent method*. A detailed study of this method along with its convergence analysis is found in [15]. A standard variant of this method is to take α_k as that positive scalar that minimizes the gradient norm. More precisely,

$$\alpha_k = \arg\min_{\alpha>0} ||\nabla f(x_k - \frac{1}{\alpha} g_k)||_2. \tag{5}$$

The solution to the problem (5) is

$$\alpha_k^{MG} = \frac{g_k^\top A^2 g_k}{g_k^\top A g_k}, \tag{6}$$

which is known as *minimal gradient steplength* [13].

3. The Barzilai and Borwein Method.

In 1988, Barzilai and Borwein [4] proposed a gradient based method to solve problem (1) that uses a nonstandard strategy to compute α_k. The authors incorporated second–order information for the calculation of the step size and proposed to choose α_k by forcing a Quasi-Newton property. More precisely, they proposed to select α_k as one of the following two alternatives:

$$\alpha_k^{BB1} = \arg\min_{\alpha>0} ||B(\alpha)s_{k-1} - y_{k-1}||_2, \quad \text{or} \quad \alpha_k^{BB2} = \arg\min_{\alpha>0} ||s_{k-1} - B(\alpha)^{-1} y_{k-1}||_2, \tag{7}$$

where $s_{k-1} = x_k - x_{k-1}$, $y_k = g_k - g_{k-1}$, $B(\alpha) = \alpha I_n$ is considered as an approximation of the Hessian of f and I_n represents the identity matrix of order n. It is straightforward to prove that the solutions of the optimization problems (7) are given by:

$$\alpha_k^{BB1} = \frac{s_{k-1}^\top y_{k-1}}{s_{k-1}^\top s_{k-1}}, \quad \text{and} \quad \alpha_k^{BB2} = \frac{y_{k-1}^\top y_{k-1}}{s_{k-1}^\top y_{k-1}}, \tag{8}$$

respectively. Other equivalent expressions for step sizes α_k^{BB1} and α_k^{BB2} are,

$$\alpha_k^{BB1} = \frac{g_{k-1}^\top A g_{k-1}}{g_{k-1}^\top g_{k-1}}, \quad \text{and} \quad \alpha_k^{BB2} = \frac{g_{k-1}^\top A^2 g_{k-1}}{g_{k-1}^\top A g_{k-1}}. \tag{9}$$

It follows that the Barzilai and Borwein's method can be seen as the steepest descent method or as the gradient method with minimal gradient steplength with one-step delay. The delay is the most emblematic feature of the gradient method with BB-step, and this fact is what makes the method faster. Nowadays, there are different proposals for step sizes that employ more than one-step delay [14]. Barzilai and Borwein showed that their method enjoys superlinear convergence rate for the case $A \in \mathbb{R}^{2\times 2}$, see [4]. In addition, Dai and Liao [6] proved that this method converges R-linearly for $n > 2$.

4. Derivation of Two New Step Sizes.

In this section, we derive two new step sizes, and we present the respective gradient method associated with such step sizes. Our proposal is based on the step sizes introduced by Barzilai and Borwein in [4]. The general idea is to introduce a term that considers information of the Hessian. In this case we assume that $B(\alpha) \approx G_{k-1}$ where G_{k-1} contains information of the Hessian, for example, based on Quasi-Newton methods. Since any matrix admits a rank one decomposition, then G_{k-1} can be written as $G_{k-1} = \sum_{i=1}^{n} D_{k-1}^{i}$, where D_{k-1}^{i} is a rank-1 matrix for $i = 1, 2, \cdots, n$. Then, we can approximate the matrix G_{k-1} in a simpler form as follows $G_{k-1} \approx nD_{k-1}$ where D_{k-1} is a rank-1 matrix. Similarly, we can write $B(\alpha)^{-1} \approx nH_{k-1}$ where H_{k-1} is a rank-1 matrix. Based on the previous considerations, we propose to select the step size of the k-th iteration α_k as the solutions of the following two optimization models,

$$\alpha_k^{ODH1} = \arg\min_{\alpha>0} \frac{\theta}{n}||nD_{k-1} - B(\alpha)||_F^2 + ||B(\alpha)s_{k-1} - y_{k-1}||_2^2, \tag{10}$$

or,

$$\alpha_k^{ODH2} = \arg\min_{\alpha>0} \frac{\theta}{n}||nH_{k-1} - B(\alpha)^{-1}||_F^2 + ||s_{k-1} - B(\alpha)^{-1}y_{k-1}||_2^2, \tag{11}$$

where $\theta > 0$ is a regularization parameter that controls the relevance of each term, $B(\alpha) = \alpha I_n$ is considered an approximation of the Hessian of f in x_k. A simple calculation shows that the solutions to problems (10) and (11) are

$$\alpha_k^{ODH1} = \frac{\theta Tr[D_{k-1}] + s_{k-1}^\top y_{k-1}}{\theta + s_{k-1}^\top s_{k-1}} \quad \text{and} \quad \alpha_k^{ODH2} = \frac{\theta + y_{k-1}^\top y_{k-1}}{\theta Tr[H_{k-1}] + s_{k-1}^\top y_{k-1}}, \tag{12}$$

respectively. In order to ensure convergence of this algorithm, we consider $H_{k-1} = \frac{s_{k-1}s_{k-1}^\top}{s_{k-1}^\top y_{k-1}}$ and $D_{k-1} = \frac{y_{k-1}y_{k-1}^\top}{s_{k-1}^\top y_{k-1}}$. Note that these choices of H_{k-1} and D_{k-1} are proportional to the last terms that appear in the update formulas of the well-known BFGS [15] for the inverse of Hessian and for the Hessian respectively. It is straightforward to verify that the proposed step sizes correspond to a weighted sum of the Barzilai-Borwein step sizes with dynamic parameter. However if we take other matrices H_{k-1} and D_{k-1} we can obtain different step sizes. Our modified gradient method is summarized in Algorithm 1.

Algorithm 1 Modified gradient method with ODH-step

Require: $A \in \mathbb{R}^{n\times n}$, $b, x_0 \in \mathbb{R}^n$, $\theta > 0$, $g_0 = Ax_0 - b$, $k = 0$.
Ensure: X^* a stationary point.
1: **while** $||g_k||_2 > \epsilon$ **do**
2: Step size selection: take $\alpha_k = \alpha_k^{ODH1}$ or choose $\alpha_k = \alpha_k^{ODH2}$ given in (12)
3: $x_{k+1} = x_k - \frac{1}{\alpha_k} g_k$,
4: $g_{k+1} = g_k - \frac{1}{\alpha_k} A g_k$,
5: $k = k + 1$,
6: **end while**
7: $x^* = x_k$.

The following technical result shows that the step-size sequences $\{\alpha_k^{ODH1}\}$ and $\{\alpha_k^{ODH2}\}$ generated by Algorithm 1, are always bounded.

Proposition 4.1. *Consider the problem* (1). *Let* $0 < \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$ *be the eigenvalues of the matrix A. Then*

$$0 < \lambda_1 \leq \alpha_k^{ODH1}, \alpha_k^{ODH2} \leq \lambda_n. \tag{13}$$

Proof. It is not difficult to prove that α_k^{ODH1} and α_k^{ODH2} can be rewritten as

$$\alpha_k^{ODH1} = \frac{s_{k-1}^\top A M_{k-1} A s_{k-1}}{s_{k-1}^\top M_{k-1} A s_{k-1}}, \tag{14}$$

and

$$\alpha_k^{ODH2} = \frac{s_{k-1}^\top A N_{k-1} s_{k-1}}{s_{k-1}^\top N_{k-1} s_{k-1}}, \tag{15}$$

where $M_{k-1} = \theta I_n + s_{k-1} s_{k-1}^\top$ and $N_{k-1} = \theta I_n + A s_{k-1} s_{k-1}^\top A$.

Thus, $\frac{1}{\alpha_k^{ODH1}}$ is the Rayleigh quotient of the matrix $W_{k-1} = M_{k-1}^{-1/2} A^{-1} M_{k-1}^{1/2}$ at the non-zero vector $p_{k-1} = M_{k-1}^{1/2} A s_{k-1}$. Similarly, α_k^{ODH2} is the Rayleigh quotient of the matrix $Z_{k-1} = N_{k-1}^{-1/2} A N_{k-1}^{1/2}$ at $d_{k-1} = N_{k-1}^{1/2} s_{k-1}$. Then we obtain

$$\lambda_{\min}^W \leq \frac{1}{\alpha_k^{ODH1}} \leq \lambda_{\max}^W, \tag{16}$$

and

$$\lambda_{\min}^Z \leq \alpha_k^{ODH2} \leq \lambda_{\max}^Z, \tag{17}$$

where $(\lambda_{\min}^W, \lambda_{\max}^W)$ are the smallest and the highest eigenvalue of the matrix W_{k-1} respectively. Analogously, $(\lambda_{\min}^Z, \lambda_{\max}^Z)$ are the smallest and the highest eigenvalue of the matrix Z_{k-1} respectively. This follows from the fact that W_{k-1} and A^{-1} are similar matrices, that

$$0 < \frac{1}{\lambda_n} = \lambda_{\min}^{A^{-1}} \leq \frac{1}{\alpha_k^{ODH1}} \leq \lambda_{\max}^{A^{-1}} = \frac{1}{\lambda_1}, \tag{18}$$

or equivalently,

$$0 < \lambda_1 \leq \alpha_k^{ODH1} \leq \lambda_n. \tag{19}$$

Analogously, since the matrices Z_{k-1} and A are similar matrices we have,

$$0 < \lambda_1 \leq \alpha_k^{ODH2} \leq \lambda_n, \tag{20}$$

which completes the proof. □ □

4.1. Two Variants of ODH Step Sizes.

In this subsection, we present two variants of the proposed ODH step sizes following the ideas in [17, 18]. Zhou et.al. [17] proposed the following adaptive Barizilai-Borwein step size,

$$\alpha_k^{ABB} = \begin{cases} \alpha_k^{BB2} & \text{if } \alpha_k^{BB2} \leq \kappa \alpha_k^{BB1} \\ \alpha_k^{BB1} & \text{otherwise} \end{cases} \tag{21}$$

where $\kappa \in (0, 1)$. On the other hand, Frassoldati et.al. in [18], proposed two different step sizes based on the BB-steps, one of them is given by

$$\alpha_k^{ABB_{min1}} = \begin{cases} \alpha_k = \min\{\alpha_j^{BB2} | j = M, \ldots, k\} & \text{if } \alpha_k^{BB2} \leq \kappa \alpha_k^{BB1} \\ \alpha_k^{BB1} & \text{otherwise} \end{cases} \tag{22}$$

where $M = \max\{1, k - m\}$ for some m fixed. The step sizes defined in (21)–(22) are one of the most efficient steps to date. In a recent numerical study [20] these step sizes obtained a good performance.

Taking into account the previous ideas (21)–(22) and the good performance of these step sizes on large scale and very ill–conditioned problems, in this section we include similar alternatives. The first one is based on (21) and is defined as follows:

$$\alpha_k^{AODH} = \begin{cases} \alpha_k^{ODH1} & \text{if } \alpha_k^{ODH1} \leq \kappa \alpha_k^{ODH2} \\ \alpha_k^{ODH2} & \text{otherwise} \end{cases} \tag{23}$$

The second strategy is based on (22):

$$\alpha_k^{AODH_{min1}} = \begin{cases} \alpha_k = \min\{\alpha_j^{ODH1} | j = M, \ldots, k\} & \text{if } \alpha_k^{ODH1} \leq \kappa \alpha_k^{ODH2} \\ \alpha_k^{ODH2} & \text{otherwise} \end{cases} \tag{24}$$

In Section 6, we present a numerical study of the previous variants (23)-(24). For this purpose, we use different test problems. According to the experimental results, the step size α_k^{AODH} tends to be more efficient than α_k^{ABB}, while the step sizes $\alpha_k^{ABB_{min1}}$ and $\alpha_k^{AODH_{min1}}$ obtain a similar performance.

5. Convergence Analysis.

In this section, we establish the convergence analysis of our algorithm applied to the strictly convex quadratic optimization problem (1). To demonstrate the global convergence of Algorithm 1, we adapted the Raydan's proof [5] to our step sizes. We consider the updating formula of the gradient method

$$x_{k+1} = x_k - \frac{1}{\alpha_k} g_k, \tag{25}$$

where α_k is given by our proposals (12). In the rest of this section, we only prove the convergence of our algorithm for the following matrix selections: $H_{k+1} = \frac{s_k s_k^\top}{s_k^\top y_k}$ and $D_{k+1} = \frac{y_k y_k^\top}{s_k^\top y_k}$.

Let $0 < \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$ be the eigenvalues of A associated with the eigenvectors $\mathbb{V} := \{v_1, v_2, \ldots, v_n\}$ and suppose, without loss of generality, that $\mathbb{V}$ is an orthonormal basis of $\mathbb{R}^n$.

Now, we denote by x^* to the unique minimizer of f, $\{x_k\}$ the sequence generated by the recursive formula (25) starting at a given initial point x_0. Let us define the k-th error by $e_k = x^* - x_k$ for all k. Then, using (25) and $g_k = Ax_k - b$, where $b = Ax^*$ we have

$$Ae_k = \alpha_k s_k, \; \forall k. \tag{26}$$

By substituting $s_k = e_k - e_{k+1}$ in (26) we obtain

$$e_{k+1} = \frac{1}{\alpha_k}(\alpha_k I_n - A)e_k, \tag{27}$$

for all k. Now, for any initial error e_0, there are scalars $d_1^0, d_2^0, \ldots, d_n^0$ such that

$$e_0 = \sum_{i=1}^{n} d_i^0 v_i, \tag{28}$$

where $\{v_1, v_2, \ldots, v_n\}$ are the orthonormal eigenvectors of A associated with the eigenvalues $0 < \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$. Using (27) we get

$$e_{k+1} = \sum_{i=1}^{n} d_i^{k+1} v_i, \tag{29}$$

for any positive integer k, where

$$d_i^{k+1} = \left(\frac{\alpha_k - \lambda_i}{\alpha_k}\right) d_i^k = \prod_{j=0}^{k} \left(\frac{\alpha_j - \lambda_i}{\alpha_j}\right) d_i^0. \tag{30}$$

Therefore, the convergence properties of the sequence $\{e_k\}$ will only depend on the behavior of each of the sequences $\{d_i^k\}$, with $1 \leq i \leq n$. The next lemma establishes

the Q-linear convergence of the sequence $\{d_1^k\}$.

Lemma 5.1. *The sequence $\{d_1^k\}$ converges to zero Q-linearly with convergence factor $\hat{c} = 1 - \frac{\lambda_1}{\lambda_n}$.*

Proof. The proof of this lemma appears in [5]. □ □

Additionally, the proposed step sizes satisfy the following result.

Proposition 5.2. *The step sizes defined in* (12) *can be rewritten as*

$$\alpha_{k+1}^{ODH1} = \frac{\theta\alpha_k^2 e_k^\top A^4 e_k + (e_k^\top A^3 e_k)^2}{\theta\alpha_k^2 e_k^\top A^3 e_k + (e_k^\top A^2 e_k)(e_k^\top A^3 e_k)}, \tag{31}$$

and

$$\alpha_{k+1}^{ODH2} = \frac{\theta\alpha_k^2 e_k^\top A^3 e_k + (e_k^\top A^4 e_k)(e_k^\top A^3 e_k)}{\theta\alpha_k^2 e_k^\top A^2 e_k + (e_k^\top A^3 e_k)^2}, \tag{32}$$

respectively.

Proof. Using the equation (14) we have,

$$\alpha_{k+1}^{ODH1} = \frac{s_k^\top A M_k A s_k}{s_k^\top M_k A s_k},$$

or equivalently,

$$\alpha_{k+1}^{ODH1} = \frac{\theta s_k^\top A^2 s_k + (s_k^\top A s_k)^2}{\theta s_k^\top A s_k + (s_k^\top s_k)(s_k^\top A s_k)},$$

replacing the equation (26) in the previous expression we obtain,

$$\alpha_{k+1}^{ODH1} = \frac{\theta(\frac{1}{\alpha_k}Ae_k)^\top A^2(\frac{1}{\alpha_k}Ae_k) + ((\frac{1}{\alpha_k}Ae_k)^\top A(\frac{1}{\alpha_k}Ae_k))^2}{\theta(\frac{1}{\alpha_k}Ae_k)^\top A(\frac{1}{\alpha_k}Ae_k) + ((\frac{1}{\alpha_k}Ae_k)^\top(\frac{1}{\alpha_k}Ae_k))((\frac{1}{\alpha_k}Ae_k)^\top A(\frac{1}{\alpha_k}Ae_k))},$$

or equivalently,

$$\alpha_{k+1}^{ODH1} = \frac{\theta\alpha_k^2 e_k^\top A^4 e_k + (e_k^\top A^3 e_k)^2}{\theta\alpha_k^2 e_k^\top A^3 e_k + (e_k^\top A^2 e_k)(e_k^\top A^3 e_k)}.$$

On the other hand, from equation (15) we have

$$\alpha_{k+1}^{ODH2} = \frac{s_k^\top A N_k s_k}{s_k^\top N_k s_k},$$

which implies that

$$\alpha_{k+1}^{ODH2} = \frac{\theta s_k^\top A s_k + (s_k^\top A^2 s_k)(s_k^\top A s_k)}{\theta s_k^\top s_k + (s_k^\top A s_k)^2}.$$

Substituting the equation (26) in the above expression we get

$$\alpha_{k+1}^{ODH2} = \frac{\theta(\frac{1}{\alpha_k}Ae_k)^\top A(\frac{1}{\alpha_k}Ae_k) + ((\frac{1}{\alpha_k}Ae_k)^\top A^2(\frac{1}{\alpha_k}Ae_k))((\frac{1}{\alpha_k}Ae_k)^\top A(\frac{1}{\alpha_k}Ae_k))}{\theta(\frac{1}{\alpha_k}Ae_k)^\top(\frac{1}{\alpha_k}Ae_k) + ((\frac{1}{\alpha_k}Ae_k)^\top A(\frac{1}{\alpha_k}Ae_k))^2},$$

that is

$$\alpha_{k+1}^{ODH2} = \frac{\theta\alpha_k^2 e_k^\top A^3 e_k + (e_k^\top A^4 e_k)(e_k^\top A^3 e_k)}{\theta\alpha_k^2 e_k^\top A^2 e_k + (e_k^\top A^3 e_k)^2},$$

which completes the proof. □ □

The following lemma is valid for our Algorithm 1 using the update formula (25) with step size $\alpha_k := \alpha_k^{ODH1}$ given by Eq. (12) in all iterations.

Lemma 5.3. *If $\theta > 0$ and all the sequences $\{d_1^k\}, \{d_2^k\}, \ldots, \{d_l^k\}$ converge to zero for a fixed positive integer l, $1 \leq l \leq n$. Then*

$$\lim_{k\to\infty} \inf |d_{l+1}^k| = 0.$$

Proof. Suppose by contradiction, that there exists $\epsilon > 0$ such that

$$(d_{l+1}^k)^2\lambda_{l+1}^2 \geq \epsilon, \quad \forall k. \tag{33}$$

Substituting (29) in the expression (31) given by Proposition 5.2 and using the fact that $\mathbb{V}$ is an orthonormal set we obtain,

$$\alpha_{k+1} = \frac{\theta\alpha_k^2\Sigma_1^4 + (\Sigma_1^3)^2}{\theta\alpha_k^2\Sigma_1^3 + \Sigma_1^2\Sigma_1^3}, \tag{34}$$

where $\Sigma_q^p := \sum_{i=q}^n (d_i^k)^2\lambda_i^p$.

Now, since all the sequences $\{d_1^k\}, \{d_2^k\}, \ldots, \{d_l^k\}$ converge to zero, then there must exist $\hat{k}_1 \in \mathbb{N}$ sufficiently large such that,

$$\sum_{i=1}^{l} (d_i^k)^2\lambda_i^2 \leq \frac{1}{4}\epsilon, \quad \forall k \geq \hat{k}_1, \tag{35}$$

and, similarly there exists $\hat{k}_2 \in \mathbb{N}$ sufficiently large such that

$$\sum_{i=1}^{l} (d_i^k)^2\lambda_i^3 \leq \frac{1}{5}\epsilon\lambda_{l+1}, \quad \forall k \geq \hat{k}_2. \tag{36}$$

Let us take $\hat{k} = \max(\hat{k}_1, \hat{k}_2)$. Since $\lambda_i \leq \lambda_j$ for $i \leq j$, and by (34)-(35), we have for each $k \geq \hat{k}$

$$\begin{aligned}
\alpha_{k+1} &= \frac{\theta\alpha_k^2\Sigma_1^4 + (\Sigma_1^3)^2}{\theta\alpha_k^2\Sigma_1^3 + \Sigma_1^2\Sigma_1^3} \\
&\geq \frac{\theta\alpha_k^2\lambda_{l+1}C + \Sigma_1^3}{\theta\alpha_k^2 + \Sigma_1^2} \\
&\geq \frac{\theta\alpha_k^2 + \Sigma_{l+1}^2}{\theta\alpha_k^2 + \Sigma_1^2}C\lambda_{l+1} \\
&\geq \frac{\theta\alpha_k^2 + \Sigma_{l+1}^2}{\theta\alpha_k^2 + \frac{1}{4}\epsilon + \Sigma_{l+1}^2}C\lambda_{l+1}, \qquad (37)
\end{aligned}$$

where $C = \frac{\Sigma_{l+1}^3}{\Sigma_1^3}$. Note that $0 < C \leq 1$.

Since $\Sigma_{l+1}^2 = \sum_{i=l+1}^n (d_i^k)^2\lambda_i^2 \geq (d_{l+1}^k)^2\lambda_{l+1}^2 \geq \epsilon$ and $\theta \geq 0$ we get $\theta\alpha_k^2 + \Sigma_{l+1}^2 \geq \Sigma_{l+1}^2 \geq \epsilon$, or equivalently, $5(\theta\alpha_k^2 + \Sigma_{l+1}^2) \geq 4(\theta\alpha_k^2 + \frac{1}{4}\epsilon + \Sigma_{l+1}^2)$, which leads to

$$\frac{\theta\alpha_k^2 + \Sigma_{l+1}^2}{\theta\alpha_k^2 + \frac{1}{4}\epsilon + \Sigma_{l+1}^2} \geq \frac{4}{5}. \qquad (38)$$

Using (37) and (38) we obtain,

$$\lambda_n \geq \alpha_{k+1} \geq \frac{\theta\alpha_k^2 + \Sigma_{l+1}^2}{\theta\alpha_k^2 + \frac{1}{4}\epsilon + \Sigma_{l+1}^2}C\lambda_{l+1} \geq \frac{4}{5}C\lambda_{l+1}. \qquad (39)$$

On the other hand, using (36) we get

$$1 \geq C = \frac{\Sigma_{l+1}^3}{\Sigma_1^3} \geq \frac{\Sigma_{l+1}^3}{\frac{1}{5}\lambda_{l+1}\epsilon + \Sigma_{l+1}^3}. \qquad (40)$$

Now, it is clear that, $\Sigma_{l+1}^3 \geq (d_{l+1}^k)^2\lambda_{l+1}^3 = [(d_{l+1}^k)^2\lambda_{l+1}^2]\lambda_{l+1} \geq \epsilon\lambda_{l+1}$, which implies that, $(6\Sigma_{l+1}^3 - 5\Sigma_{l+1}^3) \geq \epsilon\lambda_{l+1}$, or equivalently, $6\Sigma_{l+1}^3 \geq 5\left(\frac{1}{5}\lambda_{l+1}\epsilon + \Sigma_{l+1}^3\right)$, i.e.

$$\frac{\Sigma_{l+1}^3}{\frac{1}{5}\lambda_{l+1}\epsilon + \Sigma_{l+1}^3} \geq \frac{5}{6}. \qquad (41)$$

Using, (40) and (41) we obtain

$$1 \geq C \geq \frac{\Sigma_{l+1}^3}{\frac{1}{5}\lambda_{l+1}\epsilon + \Sigma_{l+1}^3} \geq \frac{5}{6}. \qquad (42)$$

From (39) and (42) we get the following inequality,

$$\alpha_k \geq \frac{4}{5}C\lambda_{l+1} \geq \frac{4}{5}\frac{5}{6}\lambda_{l+1} = \frac{2}{3}\lambda_{l+1}, \quad \forall k \geq \hat{k}, \qquad (43)$$

which implies the bound

$$\left|1-\frac{\lambda_{l+1}}{\alpha_k}\right| \leq \max\left(\frac{1}{2}, 1-\frac{\lambda_1}{\lambda_n}\right), \quad \forall k \geq \hat{k}+1. \tag{44}$$

Finally, using (44) and the first part of (30), we obtain

$$|d_{l+1}^{k+1}| = \left|1-\frac{\lambda_{l+1}}{\alpha_k}\right||d_{l+1}^{k}| \leq \hat{c}|d_{l+1}^{k}|,$$

for all $k \geq \hat{k}+1$, where

$$\hat{c} = \max\left(\frac{1}{2}, 1-\frac{\lambda_1}{\lambda_n}\right) < 1.$$

This contradicts the hypothesis (33), and therefore the lemma is proved. □ □

Proposition 5.4. *Let $k \in \mathbb{N}$ and a positive integer $1 \leq l \leq n$. The step size α_{k+1}^{ODH2} given in* (12) *satisfies,*

$$\alpha_{k+1}^{OD2} \geq \frac{U_k\theta\alpha_k^2+\Sigma_{l+1}^3}{U_k\theta\alpha_k^2+\Sigma_1^3}C\lambda_{l+1}, \tag{45}$$

where Σ_q^p is defined as in the proof of Lemma 5.3, $U_k = \frac{\Sigma_1^2}{\Sigma_1^3}$ and $0 < C = \frac{\Sigma_{l+1}^2}{\Sigma_1^2} \leq 1$.

Proof. Replacing (29) in the expression (32) of Proposition 5.2 and using the orthonormality of $\mathbb{V}$, we obtain

$$\alpha_{k+1} = \frac{\theta\alpha_k^2\Sigma_1^3+\Sigma_1^4\Sigma_1^3}{\theta\alpha_k^2\Sigma_1^2+(\Sigma_1^3)^2}. \tag{46}$$

So, using $\lambda_i \leq \lambda_j$ whenever $i \leq j$ and using (46) we have,

$$\begin{aligned}
\alpha_{k+1} &= \frac{\theta\alpha_k^2\Sigma_1^3+\Sigma_1^4\Sigma_1^3}{\theta\alpha_k^2\Sigma_1^2+(\Sigma_1^3)^2} \\
&\geq \frac{\theta\alpha_k^2C+\frac{\Sigma_{l+1}^3}{\Sigma_1^2}\Sigma_1^3}{\theta\alpha_k^2+\frac{\Sigma_1^3}{\Sigma_1^2}\Sigma_1^3}\lambda_{l+1} \\
&\geq \frac{\theta\alpha_k^2\frac{\Sigma_1^2}{\Sigma_1^3}+\Sigma_{l+1}^3}{\theta\alpha_k^2\frac{\Sigma_1^2}{\Sigma_1^3}+\Sigma_1^3}C\lambda_{l+1} \\
&= \frac{\theta\alpha_k^2U_k+\Sigma_{l+1}^3}{\theta\alpha_k^2U_k+\Sigma_1^3}C\lambda_{l+1}. \quad \square
\end{aligned}$$

□

Remark 1. Using a similar reasoning to the one we used in the proof of Lemma 5.3 in combination with the result of Proposition 5.4, it can proved that $\alpha_{k+1} := \alpha_{k+1}^{ODH2} \geq \frac{2}{3}$. Thus, Lemma 5.3 is also valid for the step size α_{k+1}^{ODH2} defined in (12).

Theorem 5.5. *Let $f(x)$ be a strictly convex quadratic function. Let $\{x_k\}$ the sequence generated by Algorithm 1 using any of the two options for the selection of the step size proposed in* (12)*, and let x^* the unique minimizer of f. Then, either $x_j = x^*$ for some finite positive integer j, or the sequence $\{x_k\}$ converges to x^*.*

Proof. The proof follows as a direct consequence of the two previous lemma's using mathematical induction. For more details see the proof of Theorem 1 in [5]. □ □

6. Numerical results.

In this section we carry out three different experiments in order to compare our proposals with other state–of–the–art gradient methods. We compare our step sizes α_k^{ODH1}, α_k^{ODH2}, α_k^{AODH} given by (23) with $\kappa = 0.5$, and $\alpha_k^{AODHmin1}$ as in (24) (all our methods with $\theta = n$, and using $(m, \tau) = (9, 0.65)$ for our AODHmin1) with the conjugate gradient method (CG) [19], the *BB1* method [4], the *BB2* method [4], the *AM* method [10], the *ABB* method [17] with $\kappa = 0.5$, the *Yuan* method [16], the *LDGM* method [12], the *AS* method [11], the ABB_{min1} method with $(m, \tau) = (9, 0.8)$ [18]. All the experiments were performed using Matlab 7.10 on an intel(R) CORE(TM) i7-4770, CPU 3.40 GHz with 500GB HD and 16GB RAM. For all test problems, we use $\epsilon = 1e-8$ as the tolerance for the gradient norm stopping criterion and $N = 10000$ as the maximum number of iterations. The initial step size $\alpha_0 = 1$ is adopted at the first iteration for all methods except for the CG method.

Experiment 1: In order to illustrate the numerical behavior of our gradient methods with the step sizes α_k^{ODH1} and α_k^{ODH2}, we present a numerical example taken from [20]. The matrix $A \in \mathbb{R}^{n\times n}$ is a diagonal matrix $A = diag(\lambda_1, \lambda_2, \ldots, \lambda_n)$ defined by the following two structures:

- **Example 1**: $\lambda_1 = 1e^4$, $\lambda_n = 1$ and the rest of $\lambda_j's$ are determined in such a way that λ_i/λ_{i-1} is constant.
- **Example 2**: $\lambda_{\min} = 1$, $\lambda_{\max} = 1e^3$, $\lambda_i = \lambda_{\min} + (\lambda_{\max} - \lambda_{\min})u_i$, where u_i is a random number from a uniform distribution in (0,0.2) for $i = 1, 2, \ldots, n/2$ and in (0.8,1) for $i = n/2 + 1, \ldots, n$.

In this example we choose $n = 1000$, as optimal solution x^* a random unit vector from a uniform distribution and $b = Ax^*$. The starting point was the zero vector. Figures 1 and 2 show the results of our ODH step sizes defined in equation (12). Just for illustration purposes, we also include the results of the Barzilai–Borwein methods. Note that the BB1 gradient method has a similar performance and behavior to our ODH2 method, and the same happens between step sizes BB2 and ODH1. Furthermore, we can see in figures 1 and 2 that the methods BB1 and ODH2 tend to perform more oscillations than the BB2 and ODH1 methods.

Experiment 2: In this experiment we consider $A = diag(1, 2, \ldots, n)$, where n is the condition number of the Hessian of the function $f(x)$. In order to evaluate the performance of the algorithms when increasing the condition number, we consider

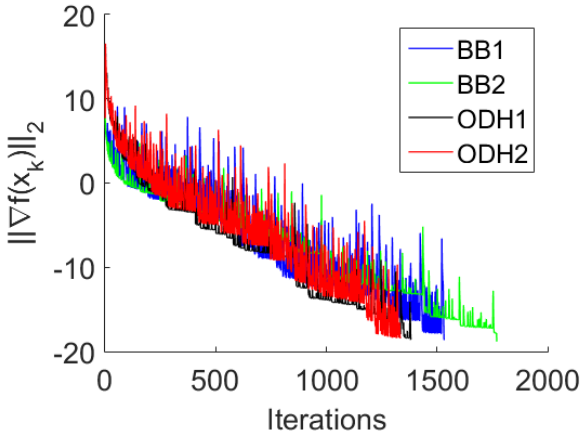

Figure 1. Example 1: Behavior of the residual norm vs iterations with $n = 1000$.

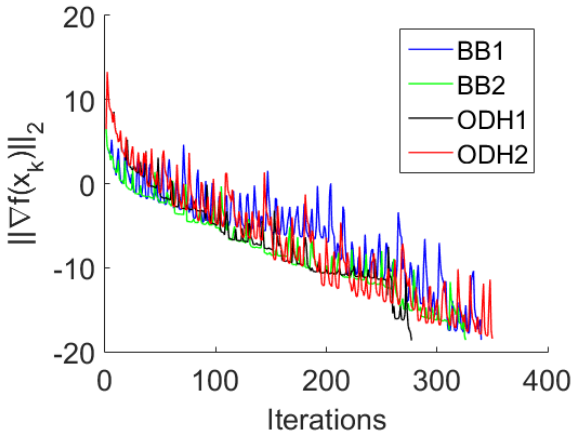

Figure 2. Example 2: Behavior of the residual norm vs iterations with $n = 1000$.

Table 1. Number of iterations required by the algorithms in Experiment 2.

n	BB1	BB2	AM	Yuan	LDGM	ABB	ABBmin1	ODH1	ODH2	AODH	AODHmin1	CG
100	146	151	104	191	121	135	130	115	93	129	105	64
1000	486	563	434	848	492	448	342	366	324	425	370	212
2500	721	772	570	982	646	652	704	606	557	689	658	338
5000	1255	1080	917	1704	924	1089	894	932	1052	1144	902	480
10000	1501	2165	1450	2597	1394	1345	1281	1014	1516	1135	1232	681
20000	2189	2185	2630	3386	2058	2242	1535	1831	1869	1815	1856	968
30000	3602	3210	2540	4166	2945	2509	2074	1880	1959	2393	1954	1188
40000	3693	3299	3495	4476	2714	2552	2266	2921	3071	2813	2206	1374
50000	4557	3415	4286	5682	3045	2978	3003	2753	2753	2733	2648	1538

the cases $n = 100k$ with $k = 1, 10, 25, 50, 100, 200, 300, 400, 500$. For all problems the solution vector is $x^* = (1, 1, \ldots, 1)^\top$ and the initial point is the zero vector $(0, \ldots, 0)^\top$. Table 1 shows the numerical results in terms of number of iterations performed by each algorithm. As expected, the CG method presents the best numerical performance compared with the rest of the methods. On the other hand, we can see that our gradient method ODH2 converges in a smaller number of iterations than the rest of the gradient methods in 33.3% of the test problems, while the ODH1, AODH and ABBmin1 methods outperform in 22.2% of the problems in terms of iterations. In addition, we note that our AODH and AODHmin1 variants show competitive results and slightly higher than the ABB and ABBmin1 methods, respectively.

Experiment 3: In this experiment, we take a problem from [12]. We generate the matrix $A \in \mathbb{R}^{n\times n}$ as $A = H^\top DH$, where H is the product of three Householder matrices, that is

$$H = (I - 2v_1v_1^\top)(I - 2v_2v_2^\top)(I - 2v_3v_3^\top),$$

where v_1, v_2 and v_3 are randomly generated unit vectors, and $D = diag(1, \lambda_2, \lambda_3, \ldots, \lambda_{n-1}, cond(A))$, is a diagonal matrix, where $\lambda_i = 1 + (cond(A) - 1) * \texttt{rand}$, for $i = 2, 3, \ldots, n-1$, *cond(A)* is the condition number of A and `rand` denotes the corresponding built-in Matlab function. The vector b is generated by using the Matlab command $b = -10 + 20 * \texttt{rand}(n, 1)$ and as initial point, we take the null vector $x_0 = (0, 0, \ldots, 0)^\top$.

In this experiment we solve ten independent problems for each pair n, $cond(A)$ and compute the average number of iterations. Table 2 shows the average number of iterations of all studied algorithms. CG method requires fewer number of iterations, since CG algorithm is the best method to solve the problem (1). The remainder algorithms show a competitive performance. However, the alternate minimization gradient method (AM) and the Yuan's method need a greater number of iterations to achieve the required tolerance than the rest of the gradient methods. This happens with more frequency while the condition number of A increases. In addition, our algorithm with ODH1 and ODH2 together with the ABBmin1 are the gradient methods that obtained the best results. In addition, we note that our step-size ODH1 obtained better performance than the ODH2. We can also observe that the AODH adaptive strategy achieved better results than the ABB step size, although both showed very similar results. On the other hand, the AODHmin1 variant obtained results very similar to the gradient method with step size ABBmin1.

Table 2. Average number of iterations required by the all methods in Experiment 3.

n	cond(A)	BB1	BB2	AM	Yuan	LDGM	ABB	ABBmin1	ODH1	ODH2	AODH	AODHmin1	CG
500	$1e^1$	44	42	45	53	43	47	42	42	44	42	42	37
	$1e^2$	149	147	142	201	147	135	136	133	129	130	134	102
	$1e^3$	473	507	487	722	470	328	330	298	319	377	342	145
	$5e^3$	1045	1150	1216	1715	984	616	472	379	385	600	483	156
	$1e^4$	1677	1796	2283	2501	1174	781	644	558	556	853	629	174
	$1e^5$	5397	5528	9760	8082	3521	2639	982	934	774	2464	954	180
1000	$1e^1$	43	44	45	54	43	44	43	43	44	43	42	38
	$1e^2$	152	153	143	209	150	137	140	142	141	137	135	110
	$1e^3$	474	497	502	758	450	406	363	353	352	384	356	187
	$5e^3$	1247	1178	1250	1750	981	755	603	525	634	747	612	217
	$1e^4$	1520	1629	2149	2498	1342	940	800	792	794	819	740	235
	$1e^5$	6110	6405	> 10000	8065	3665	2172	949	687	722	1984	1044	241
2500	$1e^1$	46	45	45	55	45	48	45	45	46	44	43	38
	$1e^2$	151	149	148	212	154	138	140	142	149	139	139	119
	$1e^3$	496	481	492	763	478	413	399	413	410	401	405	257
	$5e^3$	1241	1151	1265	1751	1022	876	737	869	843	849	734	334
	$1e^4$	1533	1679	2130	2537	1363	1108	973	1085	1003	1075	966	367
	$1e^5$	5446	6400	> 10000	8233	3886	2619	1550	1403	1299	2611	1457	372
5000	$1e^1$	46	45	47	56	44	48	44	46	45	44	45	39
	$1e^2$	157	147	147	216	157	144	142	148	150	142	145	121
	$1e^3$	570	520	505	751	475	444	428	440	441	450	421	318
	$5e^3$	1241	1246	1299	1757	991	926	836	870	905	838	859	453
	$1e^4$	1624	1793	2019	2609	1420	1180	1057	1114	1118	1155	1025	466
	$1e^5$	6372	6623	> 10000	8560	3480	2927	2028	1501	1593	2340	1902	504
10000	$1e^1$	47	46	44	57	44	51	47	46	47	45	44	39
	$1e^2$	149	158	154	221	156	141	144	150	153	140	148	123
	$1e^3$	533	518	503	766	491	455	442	477	500	458	441	361
	$5e^3$	1243	1268	1300	1825	1081	978	897	1110	1072	1017	895	553
	$1e^4$	1809	1772	2161	2532	1559	1398	1249	1492	1408	1425	1321	681
	$1e^5$	6207	6584	> 10000	8532	3928	3617	2416	2853	2749	3049	2385	774

Experiment 4: In this case A is a tridiagonal matrix whose entries are defined by

$$a_{i,i} = \frac{2}{h^2}; \;\; a_{i,i-1} = -\frac{1}{h^2} \;\; \text{if } i \neq 1; \;\; a_{i,i+1} = -\frac{1}{h^2} \;\; \text{if } i \neq n, \tag{47}$$

for all $i \in \{1, \ldots, n\}$, where $h = 11/n$ and n varies in $n \in \{500, 1000, 1500, 2000\}$, this example was taken from [12]. The optimal solution x^* was first generated by the following Matlab command $x^* = -10 + 20 * \texttt{rand}(n, 1)$ and then we set $b = Ax^*$. These kinds of problems often arise in the numerical solution of two-point boundary value problems, and can yield very ill-conditioned matrices.

In the experiment, we randomly generate ten independent test problems and solve each problem using twelve optimization methods. The maximum number of iterations was 35000 for all methods and the starting point x_0 of the algorithms was the zero vector. Table 3 shows a summary of the numerical results associated with this experiment. In this table we report the average of: number of iterations (Nitr), gradient norm (NrmG) and the CPU time in seconds (Time) achieved by the algorithms. We can observe that the CG method is the most efficient procedure among the compared methods and it takes exactly n iterations to find the solution in all the test problems. In addition, we note that the gradient methods ODH1 and ODH2 obtain a very similar performance compared to those proposed by Barzilai and Borwein. The most efficient gradient methods were ABBmin1 and AODHmin1, both of which showed highly competitive results. The variant AODH worked slightly better than the gradient method with step size ABB.

Experiment 5: Finally we include an experiment take from [12] in order to compare the numerical performance of some methods in terms of the average number of iterations solving randomly generated sparse systems of equations built using the following Matlab commands, $n = 1000$, $A = \texttt{sprandsym}(n, 0.8, 1/condA, 1)$,

Table 3. Performance of algorithms in Experiment 4.

Method	Nitr	Time	NrmG	Nitr	Time	NrmG
	n = 500			n = 1000		
BB1	5540	0.25	8.16e-9	14427	2.68	8.91e-9
BB2	6551	0.30	7.40e-9	13488	2.52	8.91e-9
AM	11251	0.49	9.33e-9	34208	6.38	1.45e-7
Yuan	7802	0.39	9.42e-9	14799	2.95	9.47e-9
LDGM	4091	0.34	8.24e-9	8005	2.03	9.02e-9
ABB	3457	0.17	6.89e-9	7842	1.53	8.13e-9
ABBmin1	2514	0.14	7.18e-9	5326	1.09	6.98e-9
ODH1	5851	0.32	8.67e-9	15060	3.08	8.07e-9
ODH2	5637	0.28	7.90e-9	13322	2.81	8.46e-9
AODHmin1	2674	0.16	7.61e-9	5288	1.21	8.13e-9
AODH	3228	0.15	7.20e-9	7562	1.54	6.38e-9
CG	500	0.02	7.86e-13	1000	0.19	7.15e-12
	n = 1500			n = 2000		
BB1	22706	23.48	8.07e-9	29538	52.83	8.82e-8
BB2	20613	21.20	8.82e-9	31726	56.69	1.46e-7
AM	35000	35.99	7.68e-4	35000	62.01	3.75e-2
Yuan	23016	24.22	9.05e-9	32530	58.19	1.85e-8
LDGM	11767	13.28	9.56e-9	14558	27.07	8.91e-9
ABB	11805	12.40	5.81e-9	19411	34.49	7.57e-9
ABBmin1	7843	8.26	6.49e-9	10931	19.59	6.39e-9
ODH1	20930	22.13	8.01e-9	31022	57.37	1.27e-7
ODH2	19992	21.13	8.32e-9	29292	52.56	7.30e-8
AODHmin1	7826	8.31	8.62e-9	10859	19.64	9.03e-9
AODH	12150	12.99	6.33e-9	18087	32.52	6.95e-9
CG	1500	1.57	6.63e-12	2000	3.56	1.04e-11

Table 4. Average number of iterations required by the six methods with ten different starting points for the experiment 5.

Tol	CondA	LDGM	ABB	ABBmin1	AODH	AODHmin1	CG
1e-3	1e+1	21	21	22	21	20	19
	1e+2	57	60	57	54	53	49
	1e+3	136	133	130	140	135	121
	1e+4	269	300	279	282	268	273
	1e+5	400	441	387	436	395	536
	1e+6	398	472	420	427	390	770
1e-6	1e+1	32	32	34	33	31	29
	1e+2	95	91	98	93	94	83
	1e+3	284	265	262	249	240	224
	1e+4	690	748	721	694	681	583
	1e+5	1825	1813	1801	1831	1713	1459
	1e+6	4212	4313	4543	4638	4115	3521
1e-9	1e+1	42	43	43	44	43	40
	1e+2	130	130	132	133	126	116
	1e+3	378	375	380	375	371	326
	1e+4	1108	1186	1110	1075	1054	889
	1e+5	3026	3257	3099	3307	2889	2368
	1e+6	9496	10900	8822	10560	8501	6148

$x^* = -10 + 20 * \texttt{rand}(n, 1)$ and $b = Ax$, where *condA* denotes the condition number of the matrix A. The initial point was randomly generated by $x_0 = -10 + 20 * \texttt{rand}(n, 1)$. For this specific experiment we consider $\epsilon \in \{1e-1, 1e-3, 1e-6\}$ as the tolerance for the gradient norm, and $N = 20000$ is the maximum number of iterations allowed for all algorithms. In this experiment we only compare the performance of the followings methods: ABB, ABBmin1, LDGM, AODH, AODHmin1 and CG, which were typically the most efficient methods in the experiments presented previously.

Table 4 summarized the average number of iterations required by the six methods with ten different starting points for the experiment 5. As shown in Table 4, the conjugate gradient method was the more efficient procedure when high precision is required. Additionally, we can see that our AODHmin1 was the second more efficient algorithm only surpassed by CG method. In addition, we observe that our variant AODHmin1 outperforms the ABBmin1 method in fact, comparing AODHmin1 against ABBmin1 we note that our variant converge more faster than ABBmin1 in fifteen of the eighteen test with a tie twice. While our variant AODH method failed to overcome the ABB method, in fact, both methods tie in two tests and both wins at eight times, so both procedures show similar performance.

7. Conclusions.

In this paper, we propose two efficient spectral gradient methods for convex quadratic minimization. These methods use the negative gradient direction in combination with two different optimal step-sizes based on the Barzilai and Borwein approach. To design these step-sizes, two matrices of rank one are considered, which correspond to the last term of the quasi-Newton formula of BFGS and DFP, respectively. Different choices

of such matrices lead to different optimal step-sizes, which produces different gradient methods. In addition, by adapting Raydan's proof [5], we establish the global convergence of our algorithms. Finally, our numerical experiments seem to indicate that the new gradient methods are competitive and sometimes preferable to other accelerated versions of the gradient method for large scale problems.

References

[1] Cauchy, Augustin. Méthode générale pour la résolution des systemes déquations simultanées, *Comp. Rend. Sci. Paris* **25**:1847 (1847), pp. 536–538.

[2] Iudin, DB and Nemirovskii, Arkadi S. Informational complexity and efficient methods for solving complex extremal problems, *Matekon* **13**:3 (1983), pp. 25–45.

[3] Nesterov, Yuri E. One class of methods of unconditional minimization of a convex function, having a high rate of convergence, *USSR Computational Mathematics and Mathematical Physics* **24**:4 (1984), pp. 80–82.

[4] Barzilai Jonathan and Borwein Jonathan M. Two-point step size gradient methods, *IMA journal of numerical analysis* **8**:1 (1988), pp. 141–148.

[5] Raydan, Marcos. On the Barzilai and Borwein choice of steplength for the gradient method, *IMA Journal of Numerical Analysis* **13**:3 (1993), pp. 321–326.

[6] Dai, Yu-Hong and Liao, Li-Zhi. R-linear convergence of the Barzilai and Borwein gradient method, *IMA Journal of Numerical Analysis* **22**:1 (2002), pp. 1–10.

[7] Raydan, Marcos. The Barzilai and Borwein gradient method for the large scale unconstrained minimization problem, *SIAM Journal on Optimization* **7**:1 (1997), pp. 26–33.

[8] Molina, Brigida and Raydan, Marcos. Preconditioned Barzilai-Borwein method for the numerical solution of partial differential equations, *Numerical Algorithms* **13**:1 (1996), pp. 45–60.

[9] Dai, Yu-Hong and Fletcher, Roger. Projected Barzilai-Borwein methods for large-scale box-constrained quadratic programming, *Numerische Mathematik* **100**:1 (2005), pp. 21–47.

[10] Dai, Yu-Hong and Yuan, Ya-Xiang. Alternate minimization gradient method, *IMA Journal of Numerical Analysis* **23**:3 (2003), pp. 377–393.

[11] Dai, Yu-Hong. Alternate step gradient method, *Optimization* **52**:4-5 (2003), pp. 395–415.

[12] Liu, Zexian and Liu, Hongwei and Dong, Xiaoliang. An efficient gradient method with approximate optimal stepsize for the strictly convex quadratic minimization problem, *Optimization* **67**:3 (2018), pp. 427–440.

[13] De Asmundis, Roberta and Di Serafino, Daniela and Hager, William W and Toraldo, Gerardo and Zhang, Hongchao. An efficient gradient method using the Yuan steplength, *Computational Optimization and Applications* **59**:3 (2014), pp. 541–563.

[14] Friedlander, Ana and Martínez, José Mario and Molina, Brigida and Raydan, Marcos. Gradient method with retards and generalizations, *SIAM Journal on Numerical Analysis* **36**:1 (1998), pp. 275–289.

[15] Luenberger, David G and Ye, Yinyu. *Linear and nonlinear programming*, Springer (1984).

[16] Yuan, Ya-xiang. A new stepsize for the steepest descent method, *Journal of Computational Mathematics* (2006), pp. 149–156.

[17] Zhou, Bin and Gao, Li and Dai, Yu-Hong. Gradient methods with adaptive step-sizes, *Computational Optimization and Applications* **35**:1 (2006), pp. 69–86.

[18] Frassoldati, Giacomo and Zanni, Luca and Zanghirati, Gaetano. New adaptive stepsize selections in gradient methods, *Journal of industrial and management optimization* **4**:2 (2008), pp. 299–312.

[19] Hestenes, Magnus Rudolph and Stiefel, Eduard. Methods of conjugate gradients for solving linear systems, **49**:1 (1952), NBS Washington, DC. 299–312.

[20] Di Serafino, Daniela and Ruggiero, Valeria and Toraldo, Gerardo and Zanni, Luca. On the

steplength selection in gradient methods for unconstrained optimization, *Applied Mathematics and Computation* **318** (20018), pp. 176–195.