A limited memory BFGS algorithm for non-convex minimization with applications in matrix largest eigenvalue problem

Article *in* Mathematical Methods of Operational Research · December 2015
DOI: 10.1007/s00186-015-0527-8

Yunhai Xiao
Henan University

Math Meth Oper Res
DOI 10.1007/s00186-015-0527-8

A Limited Memory BFGS Algorithm for Non-Convex Minimization with Applications in Matrix Largest Eigenvalue Problem

Zhanwen Shi* Guanyu Yang* Yunhai Xiao*

Abstract

This study aims to present a limited memory BFGS algorithm to solve non-convex minimization problems, and then use it to find the largest eigenvalue of a real symmetric positive definite matrix. The proposed algorithm is based on the modified secant equation, which is used to the limited memory BFGS method without more storage or arithmetic operations. The proposed method uses an Armijo line search and converges to a critical point without convexity assumption on the objective function. More importantly, we do extensive experiments to compute the largest eigenvalue of the symmetric positive definite matrix of order up to $54,929$ from the UF sparse matrix collection, and do performance comparisons with EIGS (a Matlab implementation for computing the first finite number of eigenvalues with largest magnitude). Although the proposed algorithm converges to a critical point, not a global minimum theoretically, the compared results demonstrate that it works well, and usually finds the largest eigenvalue of medium accuracy.

Key words: unconstrained optimization, limited memory BFGS method, Armijo line search, global convergence, largest eigenvalue problem, critical point.

AMS subject classifications: 90C30, 65K05, 90C53

1. Introduction

In this paper, we consider the unconstrained minimization problem

$$\min_{x\in\mathbb{R}^n} \; f(x), \tag{1.1}$$

where $f:\mathbb{R}^n\to\mathbb{R}$ is a continuously differentiable function, whose gradient at point x_k is $g(x_k)$ (abbr. g_k); n is the number of variables, which is assumed to be large. A point x^* is said to be a critical point if the corresponding gradient is zero, i.e., $\nabla f(x^*)=0$. Quasi-Newton algorithm is very popular to solve (1.1) due to the fact that it formally requires relatively few function evaluations, and is very good at handling ill-conditioning.

Quasi-Newton method generates a sequence of iterates $\{x_k\}$ as

$$x_{k+1}=x_k+\alpha_k d_k,$$

*Institute of Applied Mathematics, College of Mathematics and Statistics, Henan University, Kaifeng 475000, China (Email: zwenshi@126.com, yangguanyuok@126.com, yhxiao@henu.edu.cn).

where α_k is the steplength and d_k is the search direction defined by the following linear system

$$B_k d_k + g_k = 0, \tag{1.2}$$

where B_k is an approximation of the Hessian matrix $\nabla^2 f(x_k)$ and traditionally updated at each step to satisfy the following secant equation

$$B_{k+1} s_k = y_k, \tag{1.3}$$

where $y_k = g_{k+1} - g_k$ and $s_k = x_{k+1} - x_k$. In particular, the standard BFGS update formula for B_k is given by

$$B_{k+1} = B_k - \frac{B_k s_k s_k^\top B_k}{s_k^\top B_k s_k} + \frac{y_k y_k^\top}{s_k^\top y_k}. \tag{1.4}$$

The convergence properties of the BFGS method for convex minimization have been well studied [5, 6, 8, 19]. A couple of earliest convergence results of BFGS method for non-convex minimization are owing to Li & Fukushima [16, 17]. Under some appropriate conditions, it was shown that both approaches in [16] and [17] converge globally and superlinearly for non-convex minimization problems.

The limited memory BFGS (L-BFGS) method is an adaptation of the BFGS method for large-scale problems [21]. The implementation is almost identical to that of the standard version but the inverse Hessian approximation is formed explicitly by a small number of BFGS updates. It is frequently shown that the L-BFGS method often provides a fast rate of linear convergence, and requires minimal storage. Nevertheless, the standard L-BFGS method is only suitable for minimizing convex problems. The modified variant of Xiao, Wei & Wang [28] brought many benefits to the performance, but it converged globally for convex minimization problems. Recently, basing on the secant condition of Li & Fukushima [16], Xiao, Li, & Wei [26] designed a modified L-BFGS algorithm, where the convex assumption on the objective function was not necessary. However, its global convergence is guaranteed by using Wolfe line search. As known to all, computing the gradient of each tentative point in the Wolfe line search rule may bring some computing burden especially in intricate problems.

Although the type of quasi-Newton method has been widely used in unconstrained minimization, it appears that its use for solving matrix eigenvalue problems is new. Therefore, to show the practical usefulness of the will proposed algorithm, we change our attention on reviewing the problems for finding the largest eigenvalue of symmetric positive definite matrix. Matrix eigenvalue problems appear in many scientific or engineering fields, such as, computing the frequency response of a circuit, building the earthquake response, and the energy levels of a molecule, etc. In the past few decades, various approaches have been proposed, analyzed, and implemented for computing extreme eigenvalue including the Power method, Lanczos method, QR method, Rayleigh quotient method [14] and others [22, 4]. For more approaches about the computation of eigenvalue of symmetric matrices, we refer the reader to [10, 11]. Given a symmetric positive definite matrix A, Auchmuty [3] showed that its eigenvalues can be computed via minimizing

$$\min_{x \in \mathbb{R}^n} \phi(\frac{1}{2}\|x\|^2) + \psi(\frac{1}{2}x^\top Ax), \tag{1.5}$$

where $\phi : [0, \infty) \to \mathbb{R}$ is continuous and twice continuously differentiable with $\phi'(y) \neq 0$ for $y \neq 0$; $\psi : \mathbb{R} \to \mathbb{R}$ is twice continuously differentiable with $\psi(0) = 0$. Some functions of the form (1.5) whose minima provide information on the largest eigenvalues and eigenvectors of A can refer to [3]. Subsequently, various optimization algorithms for computing the largest eigenvalue even their combined versions have been developed, analyzed, and implemented, which mainly includes Barzilai-Borwein gradient methods in [13, 15], and steepest descent algorithm, Newton methods, quasi-Newton methods in [20].

Even with higher efficiency in solving non-convex minimization problem, the method of Xiao, Li, & Wei [26], however, typically requires the calculation of gradient information at each tentative point that can become increasingly costly as the problem's dimension growing. The main purpose of the paper is to further improve the theoretical property of [26] without losing accuracy through a novel secant equation. Indeed, unlike the algorithm in [26], the remarkable feature of the proposed algorithm is that a more efficient Armijo line search is used. Under appropriate conditions, we prove that the proposed algorithm converges globally. Besides the theoretical results, we also demonstrate convincingly that with efficient implementations, the proposed method can compute the extreme eigenvalue of very large matrix, with moderate accuracy. The variety of large-scale matrix with dimension from $4,098$ and up to $54,929$ is from the UF sparse matrix collection [9]. Moreover, we test simultaneously the well-known EIGS (a matlab implementation for computing the first six eigenvalues with largest magnitude) for accurateness estimation. The numerical results indicate that the proposed method is practically effective.

The remaining parts of this paper are as follows. In Section 2., we give some preliminaries related to the eigenvalues and eigenvectors of a real symmetric positive definite matrix for our subsequent developments. In Section 3., we review the some modified secant equations, construct our algorithm, then analyze its global convergence, and finally show how to implement the algorithm to find the largest eigenvalues of a real symmetric positive definite matrix. In section 4., we report some numerical results and do some comparisons. Finally, we offer some conclusions in section 5.. Throughout this paper, A is assumed to be a real symmetric positive definite $n \times n$ matrix with real eigenvalues $0 < \lambda_n \leq \lambda_{n-1} \leq \cdots \leq \lambda_2 \leq \lambda_1$.

2. Preliminaries

Now, we quickly review some useful results from mathematical analysis and some unconstrained minimization problems for the eigenvalues and eigenvectors of a real symmetric matrix developed by Auchmuty [3].

A critical point $x^* \in \mathbb{R}^n$ of a differentiable function, $f(x)$, in the domain of f where its gradient is zero, i.e., $\nabla f(x^*) = 0$. A critical value of f is the value of $f(x^*)$ when x^* is a critical point. If the function f is smooth, or at least twice continuously differentiable, a critical point may be either a local maximum, a local minimum, or a saddle point. The different cases may be distinguished by considering the eigenvalues of the Hessian matrix $\nabla^2 f(x^*)$. A critical point x^* at which the Hessian matrix $\nabla^2 f(x^*)$ is nonsingular is said to be non-degenerate, and the Morse index $i(x^*)$ (the number of negative eigenvalues of $\nabla^2 f(x^*)$) determine the local behavior of f near a non-degenerate x^*. A non-degenerate critical point x^* is a local minimum of f if the Morse index $i(x^*)$ is zero, or, equivalently, if $\nabla^2 f(x^*)$ is positive definite.

Now, we review some functions whose global minima are actually the eigenvectors of A associate with the largest eigenvalue of A, and the fact that the critical points of these functions give the eigenvectors of A. These results to be reviewed are mainly chosen from [3, 20]. The nonzero critical points of the Rayleigh quotient defined by

$$R(x) = \frac{x^\top Ax}{\|x\|^2}$$

are actually the eigenvectors of A, and the critical points corresponding to the largest eigenvalue are global maxima of R. The constrained optimization problem

$$\max_{\|x\|=1} x^\top Ax$$

is also used to compute the largest eigenvalue of A. However, the nonlinearity of the constraint makes this model is undesirable.

The attractive fact about the unconstrained minimization (1.5) proposed by Auchmuty [3] is that its nonzero critical points are certain specific eigenvectors of A. Moreover, by [3, Section 2] we know that the eigenvectors of the Hessian of f at critical points are preciously the eigenvectors of A. The general results on function f with form (1.5) for a real symmetric matrix A are summarized as follows:

Theorem 2.1. *[3, Theorem 1] Let A be a real symmetric matrix with f defined by (1.5) where f is twice continuously differentiable. When x^* is a nonzero critical point of f, then*
(a) x^* is an eigenvector of A corresponding to the eigenvalue $\lambda = \phi'(\frac{1}{2}\|x^*\|^2)/\psi'(\frac{1}{2}{x^*}^\top Ax^*)$.
(b) The eigenvectors of $\nabla^2 f(x^*)$ are preciously the eigenvectors of A.

An immediate result follows from Theorem 2.1 is that the eigenvectors and the corresponding eigenvalues of a real symmetric matrix A can be obtained via finding the critical points of function f with form (1.5). Essentially, specifically designed function f has the ability to find the smallest, the largest, or some other special eigenvalues of A. While A is further assumed to be positive definite, Auchmuty [2] (see also in [3, Section 3]) introduced the following smooth function on $\mathbb{R}^n$ except for origin,

$$\min_{x\in\mathbb{R}^n\setminus\{0\}} \quad h(x) = \frac{1}{2}\|x\|^2 - \sqrt{x^\top Ax}. \tag{2.1}$$

The function has the form (1.5), with $\phi(y) = y$ and $\psi(y) = -\sqrt{y}$. The results on h with form (2.1) by Auchmuty [3] are outlined as follows:

Theorem 2.2. *[3, Theorem 3] Let A be a real symmetric positive definite matrix, and f be defined in (2.1). Then*
(a) f is coercive on $\mathbb{R}^n \setminus \{0\}$ with $\min_{x\in\mathbb{R}^n\setminus\{0\}} h(x) = -\lambda_1$, and λ_1 is the largest eigenvalue of A. The minimum is attained at any $\sqrt{\lambda_1}e_1$, where e_1 is a normalized eigenvector corresponding to the eigenvalue λ_1;
(b) The nonzero critical points of f are $\sqrt{\lambda_k}e_k$, where e_k is a normalized eigenvector corresponding to the eigenvalue λ_k. Moreover, if $\lambda_1 \neq \lambda_k$, $\sqrt{\lambda_k}e_k$ is a saddle point.

Alternatively, Auchmuty [3, Section 6] developed another smooth function

$$\min_{x\in\mathbb{R}^n} \quad f(x) := \frac{1}{4}\|x\|^4 - \frac{1}{2}x^\top Ax. \tag{2.2}$$

Comparing with (2.1), this function is smooth on $\mathbb{R}^n$, since it is a polynomial. The function has the form (1.5), with $\phi(y) = y^2$ and $\psi(y) = -y$. The function has the property that it only nonzero critical point arise at eigenvectors of A corresponding to positive eigenvalues of A. More preciously, the results on minimizing f are summarized as follows:

Theorem 2.3. *[3, Theorem 12] Let A be a real symmetric positive definite matrix, and f is defined by (2.2). Then*
(a) f is coercive and $\min_{x\in\mathbb{R}^n} f(x) = -\lambda_1^2/4$ and λ_1 is the largest eigenvalue of A;
(b) The nonzer critical points of f occur at $x^* = \sqrt{\lambda_j}e_j$, where λ_j is a positive eigenvalue of A and e_j is a normalized eigenvector corresponding to the eigenvalue λ_j. Moreover, if $\lambda_1 \neq \lambda_j$, $\sqrt{\lambda_j}e_j$ is a saddle point.

Finally, Mongeau and Torki [20] proposed a new variational principle for computing the largest eigenvalue of a positive definite matrix, which is defined by

$$\min_{x\in\mathbb{R}^n} \; l(x) := \|x\|^2 - \ln(x^\top Ax), \tag{2.3}$$

where "ln" is the natural logarithm. It is clearly to show that l is is not defined everywhere on $\mathbb{R}^n$ as the functions mentioned above, and l trends to $+\infty$ as x goes to zero. The results concerning (2.3) are summarized in the following theorem

Theorem 2.4. *[20, Theorem 2.3] Let A be a real symmetric positive definite matrix, and l is defined by (2.3). Then f is coercive and $\min_{x\in\mathbb{R}^n} f(x) = 1 - \ln\lambda_1$ and λ_1 is the largest eigenvalue of A. The minimum is attained at any x in the set of unitary eigenvectors associated with the eigenvalue λ_1.*

The feature of each minimization problems allow us to use the standard optimization techniques to minimize these formulations for computing the largest eigenpair, i.e, the largest eigenvalue and an associated eigenvector.

3. Algorithm and its convergence

This section is devoted to the design of a L-BFGS for solving non-convex unconstrained minimization problem with general form (1.1). In a first step, we review the iterative scheme of L-BFGS method. Next, we exploit the particular weakness of the exiting method this leads to our new algorithm, as well as the convergence properties which are analyzed immediately.

3.1. Review on quasi-Newton method

Quasi-Newton equations play a central role in quasi-Newton methods for optimization and various quasi-Newton equations are available. In recent literatures, a number of techniques have been proposed, analyzed, and implemented (see [29] for earlier developments). Among them, we focus on the new secant equation of Li & Fukushima [16]. That is

$$B_{k+1}s_k = \widehat{y}_k = y_k + t_k s_k, \tag{3.1}$$

where $t_k > 0$ with the general form

$$t_k = \bar{C}\|g_k\|^\mu + \max\left\{-\frac{s_k^\top y_k}{\|s_k\|^2}, 0\right\}, \tag{3.2}$$

in which $\bar{C} > 0$ and integer $\mu \geq 0$. Obviously, only the gradients information are exploited in the secant equation (3.1), while the function values available are neglected. Zhang, Deng, and Chen [30] developed a new secant equation which used both gradients and function values. That is

$$B_k s_{k-1} = \widetilde{y}_{k-1}, \tag{3.3}$$

in which $\widetilde{y}_{k-1} = y_{k-1} + \widetilde{\gamma} s_{k-1}$, where

$$\widetilde{\gamma} = \frac{3(g_k + g_{k-1})^T s_{k-1} + 6(f_{k-1} - f_k)}{\|s_{k-1}\|^2}.$$

The new secant equation is superior to the usual one (1.3) in the sense that $\widetilde{y}_{k-1}$ better approximates $\nabla^2 f(x_k)s_{k-1}$ than y_{k-1}. Another modified secant equation by using both function value and gradient information is due to Wei, Li, and Qi [24]:

$$B_k s_{k-1} = \bar{y}_{k-1}, \tag{3.4}$$

in which $\bar{y}_{k-1} = y_{k-1} + \bar{\gamma} s_{k-1}$ with the exact choice of $\bar{\gamma}$, that

$$\bar{\gamma} = \frac{(g_k + g_{k-1})^T s_{k-1} + 2(f_{k-1} - f_k)}{\|s_{k-1}\|^2}.$$

Additionally, comparing with the secant equation (3.3), it concludes that $\bar{\gamma} = \frac{1}{3}\widetilde{\gamma}$. It is a very interesting fact.

It is worth stressing that the global convergence of corresponding BFGS based on (3.3) and (3.4) can not be guaranteed for non-convex unconstrained minimization. While for the modified secant equation in (3.1), it is easy to deduce that

$$\begin{aligned} \widehat{y}_k s_k \quad &= \quad \begin{cases} s_k^\top y_k + \bar{C}\|g_k\|^\mu \|s_k\|^2, & \text{if } s_k^\top y_k \geq 0 \\ s_k^\top y_k + \bar{C}\|g_k\|^\mu \|s_k\|^2 - s_k^\top y_k, & \text{if } s_k^\top y_k < 0 \end{cases} \\ &\geq \quad \bar{C}\|g_k\|^\mu \|s_k\|^2 > 0. \end{aligned} \tag{3.5}$$

Li & Fukushima [16] replaced all the y_k with $\widehat{y}_k$ in (1.4), and obtained a modified BFGS update formula

$$B_{k+1} = B_k - \frac{B_k s_k s_k^\top B_k}{s_k^\top B_k s_k} + \frac{\widehat{y}_k \widehat{y}_k^\top}{s_k^\top \widehat{y}_k}. \tag{3.6}$$

It is precisely for this attractive property of (3.5) that the related BFGS method converges globally only assuming that f has Lipschitz continuous gradient. Other developments on these modified secant conditions and related algorithms can refer to [23, 25, 32, 33, 31, 27] and the references therein.

3.2. Algorithm

At the beginning, we quickly review the earliest L-BFGS method [18, 21] for smooth unconstrained minimization problem (1.1). Let m be a positive integer and force $m := \min\{k+1, m\}$. Choose B_0 be an initial symmetric positive definite matrix. Update B_0 for m times via formula (1.4) with the pairs $\{s_i, y_i\}_{i=k-m+1}^{k}$, i.e., for $l = k-m+1, \ldots, k$ calculate

$$B_k^{(l+1)} = B_k^{(l)} - \frac{B_k^{(l)} s_l s_l^\top B_k^{(l)}}{s_l^\top B_k^{(l)} s_l} + \frac{y_l {y_l}^\top}{{y_l}^\top s_l}, \tag{3.7}$$

where $s_l = x_{l+1} - x_l$ and $B_k^{(k-m+1)} = B_0$ and then set the next B_{k+1} as $B_{k+1} = B_k^{(k+1)}$. Comparing with the standard BFGS updating version (1.4), it only needs to store a certain number of pairs $\{s_i, y_i\}$ instead of a matrix per-iteration. More specifically, Xiao, Li, & Wang [26] considered a L-BFGS method, in which an initial B_0 is updated for m times via the latest information of previous points

$$B_k^{(l+1)} = B_k^{(l)} - \frac{B_k^{(l)} s_l s_l^\top B_k^{(l)}}{s_l^\top B_k^{(l)} s_l} + \frac{\widehat{y}_l \widehat{y}_l^\top}{\widehat{y}_l^\top s_l}, \tag{3.8}$$

and then set $B_{k+1} = B_k^{(k+1)}$. The numerical experiment on some non-convex problems from CUTEr library [7] indicated that these modifications and extensions benefit algorithm's performance.

The Wolfe line search rule is used in the method of Xiao, Li, & Wang [26], which may bring some computing burden especially in intricate problems. For this reason, the aim of this paper is to reduce the computational cost without losing accuracy through adopting an Armijo line search, which only requires the functional values in the whole iterative process. Letting $\delta \in (0,1)$, $\rho \in (0,1)$, and $\tilde{\alpha} > 0$, we choose the smallest nonnegative integer j_k such that the stepsize $\alpha_k = \tilde{\alpha}\rho^{j_k}$ satisfies

$$f(x_k + \alpha_k d_k) \leq f(x_k) + \delta \alpha_k g_k^\top d_k. \tag{3.9}$$

Comparing Wolfe condition, a further advantage of the rule is that it is easily implemented especially when the gradients are time consuming.

Based on the above analysis, the scheme of L-BFGS algorithm with Armijo line search (Abbr. LBFGS_Ar) is described as follows:

Algorithm 3.1. *(LBFGS_Ar1)*

Step 0. Choose an initial point $x_0 \in \mathbb{R}^n$ and a symmetric positive definite matrix B_0. Let $0 < \delta < 1$, and $m > 0$ be given. Set $k := 0$.
Step 1. If $\|g_k\| = 0$, then stop.
Step 2. Determine d_k by $d_k = -B_k^{-1} g_k$.
Step 3. Find a steplength $\alpha_k > 0$ satisfying the Armijo condition (3.9).
Step 4. Set $x_{k+1} := x_k + \alpha_k d_k$.
Step 5. Let $m := \min\{k+1, m\}$. Update B_0 to get B_{k+1} by (3.8).
Step 6. Set $k := k+1$. Go to Step 1.

From Step 2, it seems that we need to save a matrix and compute its inverse explicitly, but practically d_k could directly be obtained from recursive iteration. More specifically, let H_k be the inverse Hessian approximation of f at x_k, i.e., $H_k = B_k^{-1}$. From (3.6), it is not difficult to deduce that the next H_{k+1} can be expressed as

$$H_{k+1} = \Big(I - \frac{s_k \widehat{y}_k^\top}{s_k^\top \widehat{y}_k}\Big) H_k \Big(I - \frac{\widehat{y}_k s_k^\top}{s_k^\top \widehat{y}_k}\Big) + \frac{s_k s_k^\top}{s_k^\top \widehat{y}_k}. \tag{3.10}$$

Denote

$$\widehat{\rho}_k = \frac{1}{s_k^\top \widehat{y}_k}, \quad \text{and} \quad \widehat{V}_k = I - \widehat{\rho}_k \widehat{y}_k s_k^\top,$$

where I is an identify matrix. If H_{k+1} is derived by updating an initial positive definite matrix H_0 $(= B_0^{-1})$ with m times by use of the latest m iterations, then it can be further reformulated as

$$\begin{aligned} H_{k+1} = & \Big[\widehat{V}_k^\top \cdots \widehat{V}_{k-m+1}^\top\Big] H_{k-m+1} \Big[\widehat{V}_{k-m+1} \cdots \widehat{V}_k\Big] \\ & + \widehat{\rho}_{k-m+1} \Big[\widehat{V}_{k-1}^\top \cdots \widehat{V}_{k-m+2}^\top\Big] s_{k-m+1} s_{k-m+1}^\top \Big[\widehat{V}_{k-m+2} \cdots \widehat{V}_{k-1}\Big] \\ & + \cdots \\ & + \widehat{\rho}_k s_k s_k^\top. \end{aligned} \tag{3.11}$$

To sum up, an alternative version of Algorithm 3.1 with Hessian inverse approximation H_k is formally outlined as follows:

Algorithm 3.2. *(LBFGS_Ar2)*

Step 0. Choose an initial point $x_0 \in \mathbb{R}^n$ and a symmetric positive definite matrix H_0. Let $0 < \delta < 1$, and $m > 0$ be given. Set $k := 0$.
Step 1. If $\|g_k\| = 0$, then stop.
Step 2. Determine d_k by $d_k = -H_k g_k$.
Step 3. Find a steplength $\alpha_k > 0$ satisfying the Armijo condition (3.9).
Step 4. Set $x_{k+1} := x_k + \alpha_k d_k$.
Step 5. Let $m := \min\{k + 1, m\}$. Update H_0 to get H_{k+1} by (3.11).
Step 6. Set $k := k + 1$. Go to Step 1.

Clearly, Algorithms 3.1 and 3.2 are equivalent theoretically. We would emphasize that the iterative scheme Algorithm 3.2 is implemented in practice, while Algorithm 3.1 is considered only for our upcoming convergence analysis.

3.3. Global convergence

This section is devoted entirely to proving the global convergence of Algorithm 3.1. Our proof is quite routine in some sense, and is closed based on the essential ideas developed by Li & Fukushima [16] for the convergence of BFGS for non-convex minimization. For clarity and ease of reference, we include the proof in this section. As usual, we assume throughout this paper that local minimizers satisfy the following standard assumptions, which guarantees that the proposed method converges globally to x^*. In the next section, we will show that these assumptions are not very restrictive, and they hold automatically for the functions mentioned in the previous section.

Assumption 3.1. *Function f is bounded below, and the level set $\mathcal{L} = \{x \mid f(x) \le f(x_0)\}$ is bounded.*

Assumption 3.2. *In some open neighborhood $\mathcal{N}$ of $\mathcal{L}$, the objective function f is Lipschitz continuous differentiable, i.e., there exist a positive constant L such that*

$$\|g(x) - g(y)\| \le L\|x - y\|, \quad \forall\ x, y \in \mathcal{N}. \tag{3.1}$$

The above assumptions show that $\nabla f(x)$ is bounded, namely, there is a constant $\gamma > 0$, such that

$$\|g(x)\| \le \gamma, \quad \forall x \in \Omega. \tag{3.2}$$

It also implies that there exists at least one critical point x^* even a local minimum such that $g(x^*) = 0$.

Indeed, to prove the global convergence of Algorithm 3.1, we mainly need to prove two conclusions which summarized in the following two lemmas respectively. The following lemma is concerned with the boundedness of $\|\widehat{y}_k\|$ and $\widehat{y}_k^\top s_k$.

Lemma 3.1. *Let $\{x_k\}$ be generated by Algorithm 3.1. If $\|g_k\| \ge \epsilon$ holds for all k with some sufficient small constant $\epsilon > 0$, then there exists a positive constant M such that*

$$\|\widehat{y}_k\| \le (L + M)\|s_k\|, \tag{3.3}$$

$$\bar{C}\epsilon^\mu \|s_k\|^2 \le \widehat{y}_k^\top s_k \le (L + M)\|s_k\|^2, \tag{3.4}$$

where L is the Lipschitz constant of g given by (3.1).

Proof. Assumption 3.2 shows that

$$\Big| -\frac{s_k^\top y_k}{\|s_k\|^2}\Big| = \frac{|s_k^\top y_k|}{\|s_k\|^2} \le \frac{\|y_k\|}{\|s_k\|} \le L.$$

Subsequently, the definition of t_k in (3.2) and the boundedness of $\|g_k\|$ in (3.2) indicate that

$$t_k \le \bar{C}\|g_k\|^\mu + L \le \bar{C}\gamma^\mu + L.$$

Denote $M \triangleq \bar{C}\gamma^\mu + L$. The definition of $\widehat{y}_k$ yields

$$\|\widehat{y}_k\| = \|y_k + t_k s_k\| \le \|y_k\| + M\|s_k\| \le (L+M)\|s_k\|.$$

The inequality (3.4) follows directly from (3.3), (3.5) and the fact $\|g_k\| \ge \epsilon$. □

Let θ_k denote the angle between s_k and $B_k s_k$, i.e.,

$$\cos\theta_k = \frac{s_k^\top B_k s_k}{\|s_k\|\|B_k s_k\|} = -\frac{g_k^\top s_k}{\|g_k\|\|s_k\|}. \tag{3.5}$$

From Lemma 3.1, it is easy to show that the angle θ_k is uniformly bounded away from 90^o. We summarize this assertion in the following lemma; its proof is omitted as it can be found similarly in [18, 26]

Lemma 3.2. *Let $\{x_k\}$ be generated by Algorithm 3.1, and assume that the matrix B_0 is chosen so that $\|B_0\|$ and $\|B_0^{-1}\|$ are bounded. If $\|g_k\| \ge \epsilon$ holds for all k with some constant $\epsilon > 0$, then there exists a positive constant $\xi > 0$, such that the inequality*

$$\cos\theta_k \ge \xi \tag{3.6}$$

holds for all k.

The global convergence of Algorithm 3.1 can be easily established based on the above results. We state it in the following theorem. A quite straightforward way of proving the following theorem is the combining of Lemma 3.2 and [1, Theorem 4.1]. For sake of completeness of the paper, we report the proof here.

Theorem 3.1. *Let $\{x_k\}$ be generated by Algorithm 3.1. Then we have*

$$\liminf_{k\to\infty} \|g_k\| = 0, \tag{3.7}$$

i.e., there exist a critical point x^* such that $\{x_k\} \to x^*$.

Proof. For contradiction, we suppose that (3.7) does not hold. Then there is a constant $\epsilon > 0$ such that

$$\|g_k\| \ge \epsilon \quad \forall k. \tag{3.8}$$

At the case of $\alpha_k \ne \tilde{\alpha}$, it follows from Step 3 of Algorithm 3.1 that $\rho^{-1}\alpha_k$ does not satisfy (3.9). This means

$$f(x_k + \rho^{-1}\alpha_k d_k) - f(x_k) > \delta\rho^{-1}\alpha_k g_k^\top d_k. \tag{3.9}$$

By the mean-value theorem, there is a $\theta_k \in (0,1)$ such that

$$\begin{aligned} f(x_k + \rho^{-1}\alpha_k d_k) - f(x_k) =& \rho^{-1}\alpha_k g(x_k + \theta_k\rho^{-1}\alpha_k d_k)^\top d_k \\ =& \rho^{-1}\alpha_k g_k^\top d_k + \rho^{-1}\alpha_k \big[g(x_k + \theta_k\rho^{-1}\alpha_k d_k) - g(x_k)\big]^\top d_k \\ \le& \rho^{-1}\alpha_k g_k^\top d_k + L\rho^{-2}\alpha_k^2\|d_k\|^2, \end{aligned}$$

that is,

$$(\delta - 1)g_k^\top d_k < L\rho^{-1}\alpha_k\|d_k\|^2,$$

i.e,

$$\alpha_k > -\frac{(1-\delta)\rho}{L}\frac{g_k^\top d_k}{\|d_k\|^2}. \tag{3.10}$$

Combining with (3.9) yields

$$f(x_k + \alpha_k d_k) - f(x_k) < -\delta\frac{(1-\delta)\rho}{L}\frac{(g_k^\top d_k)^2}{\|d_k\|^2},$$

or, equivalently

$$f(x_k) - f(x_k + \alpha_k d_k) > \delta\frac{(1-\delta)\rho}{L}\|g_k\|^2\cos^2\theta_k.$$

Then, for all $k > 0$, we have

$$f(x_0) - f(x_k) = \sum_{i=0}^{k-1}[f(x_i) - f(x_{i+1})] > \delta\frac{(1-\delta)\rho}{L}\sum_{i=0}^{k-1}\|g_i\|^2\cos^2\theta_i. \tag{3.11}$$

Taking limits on both sides of (3.11) and noting $f(x_k) \to f(x^*)$ yields

$$\liminf_{k\to\infty}\|g_k\|\cos\theta_k = 0.$$

Combining the low boundness of $\cos\theta_k$ in Lemma 3.2, we get $\liminf_{k\to\infty}\|g_k\| = 0$. This contradicts the assumption (3.8), and then shows the desirable result (3.7).

At the case of $\alpha_k = \tilde{\alpha}$, it follows from (3.9) that

$$f(x_k + \alpha_k d_k) = f(x_k) + \delta\tilde{\alpha}g_k^\top d_k = f(x_k) - \delta\tilde{\alpha}\|g_k\|\|d_k\|\cos\theta_k.$$

Similarly, adding both sides of the above equality and taking $k \to \infty$ yields

$$\liminf_{k\to\infty}\|g_k\|\|d_k\|\cos\theta_k = 0.$$

Keeping in mind that $\theta_k \geq \xi$ for all k, we get

$$\liminf_{k\to\infty}\|g_k\|\|d_k\| = 0.$$

From the fact that B_k is symmetric positive definite and the relation $d_k = -B_k^{-1}g_k$, it holds that

$$0 < g_k^\top B_k^{-1}g_k \leq \|g_k\|\|B_k^{-1}g_k\| = \|g_k\|\|d_k\| \to 0,$$

which also implies that $\liminf_{k\to\infty}\|g_k\| = 0$. This contradicts the assumption (3.8) again, and then the desirable result (3.7) holds. Taking both cases together, the assertion of this theorem follows directly. $\square$

3.4. Applications in computing eigenvalues

In this section, we restrict our attention to the Algorithm 3.2 of computing the largest eigenvalue of a symmetric positive definite matrix A. Particularly, we focus on the formulation (2.2) which has been frequently tested via different type of optimization techniques in [13, 15, 20].

It is easy to deduce that the gradient and the Hessian of the objective function in (2.2) are respectively given by

$$\nabla f(x) = \|x\|^2 x - Ax,$$

and

$$\nabla^2 f(x) = \|x\|^2 I_n + 2xx^\top - A,$$

where I_n is an identity matrix in $\mathbb{R}^{n\times n}$. On the one hand, it is not difficult to see that function f is bounded below, and the level set $\{x \mid f(x) \le f(x_0)\}$ is bounded for some arbitrary x_0. On the other hand, since $\nabla f(x)$ is twice continuously differentiable, for any x and y, there exists a positive scalar $L > 0$ such that $\|\nabla f(x) - \nabla f(y)\| \le L\|x - y\|$. Both cases imply that Assumptions 3.1 and 3.1 hold automatically, which means that the L-BFGS method proposed above can be used to compute the critical points of the minimization formulation (2.2). When applied to the model (2.2), Algorithm 3.2 yields the following iterative scheme:

Algorithm 3.3. *(LBFGS_Ar)*

Step 0. Input a symmetric positive definite matrix A. Choose an initial point $x_0 \in \mathbb{R}^n$ and a symmetric positive definite matrix H_0. Let $0 < \delta < 1$, and $m > 0$ be given. Compute $g_0 = \|x_0\|^2 x_0 - Ax_0$. Set $k := 0$.
Step 1. If $\|g_k\| = 0$, then stop.
Step 2. Comput d_k by $d_k = -H_k g_k$.
Step 3. Find a steplength $\alpha_k > 0$ satisfying the Armijo condition (3.9).
Step 4. Set $x_{k+1} := x_k + \alpha_k d_k$.
Step 5. Let $m := \min\{k + 1, m\}$. Compute $g_{k+1} = \|x_{k+1}\|^2 x_{k+1} - Ax_{k+1}$. Set $y_k = g_{k+1} - g_k$, and $s_k = x_{k+1} - x_k$. Update H_0 to get H_{k+1} by (3.11).
Step 6. Set $k := k + 1$. Go to Step 1.

It follows from Theorem 3.1, we know that the sequence $\{x_k\}$ generated by Algorithm 3.3 converges to the critical point of problem (2.2). Combining with Theorem 2.4 or [20, Theorem 2.3], it further indicates that if x^* is a global minimum, i.e., $x^* = \arg\min_{x\in\mathbb{R}^n} f(x)$, and the critical value is $f(x^*) = -\lambda_1^2/4$, then λ_1 is the largest eigenvalue of A.

4. Numerical experiments

The experiments of this section aim to show the practical usefulness of the proposed algorithm in finding the extreme eigenvalues of symmetric positive definite matrices. Firstly, in order to determine an appropriate m for this application, we carefully implement the algorithm with different choices. Secondly, as a further comparison, we also test against with EIGS — a matlab implementation for computing the first six eigenvalues with largest magnitude. The method is implemented in Matlab R2013a with double precision and the experiments are performed on a personal computer equipped with an Intel Core CPU at 3.30GHz and 4 GB of memory.

Because the computer memory is limited, the considered algorithm is tested on 42 large-scale symmetric positive definite matrices with size at least $4,000$ and up to $54,929$ from the University of Florida sparse matrix collection (UF) [9]. The collection contains a large and actively growing set of sparse matrices that arise in real applications, and is widely used by the numerical linear algebra community for the development and performance evaluation of sparse matrix algorithms. For a tutorial on the details of each matrix "A", one may consult the website `http://www.cise.ufl.edu/research/sparse/matrices`. For each one in the set, we seek its largest eigenvalue via two different approaches.

First of all, for each matrix to be tested, we artificially estimate its largest eigenvalue via the Matlab command EIGS in advance (denote V_eigs), and measure the accuracy of an optimal solution by using the following relative error:

$$\text{Relerr} = \frac{|\text{V\_eigs} - \lambda_1|}{|\lambda_1|},$$

where λ_1 is the largest eigenvalue derived by LBFGS_Ar. For each test problem, our algorithm starts randomly at the following point:

$$x_0 = \frac{x'}{\|x'\|} \quad \text{where} \quad x' = \text{randn}(n, 1),$$

where "randn" is a Matlab command to generate values from a normal distribution. For each test problem, the termination condition is

$$\text{Relerr} \leq 10^{-6}.$$

In addition, the iterative process is also stopped if the number of iterations exceeds 1000. Meanwhile, the input parameters' values for command EIGS are set as

$$k = 1, \quad \text{tol} = 10^{-6}, \quad v_0 = x_0, \quad \text{sigma} =' la', \quad \text{maxit} = 6000, \quad \text{and} \quad p = 20,$$

where p is the number of basis vectors, "maxit" denotes the maximum number of allowed iterations.

In the first part of the experiment, we test the influence of the positive integer m on the algorithm's performance. The numerical results obtained by LBFGS_Ar with different m values are reported in Tables 1-2, which contain the name of the problems in UF (Name), order of each matrix (Order), number of iterations (Iter), CPU time required in seconds (Time), the produced largest eigenvalues (Largest eig), and the relative errors (Relerr). The main conclusion that can be drawn from these tables is that the proposed algorithm, used as Armijo line search, is able to substantially produce medium accuracy largest eigenvalues.

Table 1. Numerical Results LBFGS_Ar with m = 2, 3, and 4

		m=2				m=3				m=4			
Name	Order	Time	Iter	Largest eig	Relerr	Time	Iter	Largest eig	Relerr	Time	Iter	Largest eig	Relerr
sts4098	4098	0.25	24	3.07102340e+08	5.3131e-07	0.18	18	3.07102309e+08	6.3118e-07	0.32	29	3.07102311e+08	6.2645e-07
bcsstk28	4410	0.35	20	7.69620657e+08	9.6828e-07	0.23	13	7.69620972e+08	5.5984e-07	0.54	29	7.69620714e+08	8.9428e-07
mhd4800b	4800	0.19	20	2.19626686e+00	8.0600e-07	0.16	18	2.19626681e+00	8.2796e-07	0.29	29	2.19626731e+00	6.0185e-07
bcsstk16	4884	0.93	30	4.94316082e+09	9.7454e-07	1.06	34	4.94316126e+09	8.8430e-07	2.58	77	4.94316230e+09	6.7393e-07
c-29	5033	0.07	8	2.22804966e+06	1.0455e-07	0.06	7	2.22804988e+06	4.2281e-09	0.08	9	2.22804954e+06	1.5670e-07
c-30	5321	0.05	5	4.68199012e+06	1.8655e-07	0.06	6	4.68199037e+06	1.3354e-07	0.12	11	4.68199067e+06	6.9187e-08
c-33	6317	0.10	11	2.04540826e+05	2.0651e-08	0.07	7	2.04540710e+05	5.8444e-07	0.13	13	2.04540787e+05	2.0940e-07
rajat01	6833	1.48	82	4.21268065e+01	8.9803e-07	1.28	70	4.21268082e+01	8.5839e-07	2.15	113	4.21268089e+01	8.4193e-07
c-36	7479	0.37	26	8.74111004e+03	9.1346e-07	0.16	11	8.74111647e+03	1.7815e-07	0.31	21	8.74111282e+03	5.9626e-07
aft01	8205	0.09	6	1.00000000e+15	7.2955e-12	0.07	5	9.99999996e+14	4.0915e-09	0.09	6	9.99999996e+14	4.0915e-09
bloweybq	10001	0.11	10	4.99974980e+03	5.0653e-08	0.11	10	4.99974636e+03	7.3898e-07	0.13	11	4.99974905e+03	1.9941e-07
bcsstk17	10974	3.09	62	1.29606086e+10	5.5552e-07	2.64	54	1.29606050e+10	8.3157e-07	6.43	121	1.29606048e+10	8.5137e-07
bcsstk25	15439	1.53	36	1.06001959e+15	8.6540e-07	0.86	21	1.06001945e+15	9.9397e-07	2.71	61	1.06001968e+15	7.8216e-07
olafu	16146	2.79	30	9.47869446e+11	9.4223e-07	0.66	7	9.47869950e+11	4.1015e-07	9.51	93	9.47869511e+11	8.7377e-07
gyro_k	17361	6.09	46	3.65694980e+09	6.9155e-07	4.43	34	3.65694955e+09	7.5972e-07	14.46	97	3.65694879e+09	9.6743e-07
gyro	17361	6.14	46	3.65694980e+09	6.9155e-07	4.44	34	3.65694955e+09	7.5972e-07	15.41	97	3.65694879e+09	9.6743e-07
rajat27	20640	0.61	30	7.69114407e+05	7.5314e-07	0.41	21	7.69114424e+05	7.3174e-07	1.37	61	7.69114362e+05	8.1200e-07
c-50	22401	0.29	10	2.65859462e+05	7.7526e-07	0.23	8	2.65859469e+05	7.5052e-07	0.39	13	2.65859642e+05	9.9420e-08
as-22july06	22963	1.53	36	7.16129608e+01	5.5232e-07	1.43	35	7.16129365e+01	8.9170e-07	4.34	85	7.16129292e+01	9.9266e-07
ca-CondMat	23133	2.91	48	3.79540840e+01	7.6200e-07	1.92	34	3.79540890e+01	6.2849e-07	3.85	57	3.79540899e+01	6.0656e-07
c-52	23948	0.26	8	1.94346228e+15	1.1168e-07	0.20	6	1.94346106e+15	7.3524e-07	0.45	13	1.94346230e+15	9.6956e-08
mult_dcop_02	25187	0.59	14	1.25613744e+03	2.1411e-07	0.39	10	1.25613760e+03	8.5984e-08	0.77	17	1.25613744e+03	2.1936e-07
net100	29920	9.45	44	1.22667791e+02	4.6336e-07	7.48	37	1.22667736e+02	9.1557e-07	23.92	98	1.22667743e+02	8.5424e-07
bloweybl	30003	0.33	14	1.00007495e+02	6.0378e-08	0.23	10	1.00007448e+02	5.3359e-07	0.54	21	1.00007500e+02	5.3059e-09
c-53	30235	1.44	24	5.36188474e+03	4.6472e-07	0.94	16	5.36188437e+03	5.3364e-07	1.79	29	5.36188467e+03	4.7774e-07
c-54	31793	0.54	10	1.82553355e+08	1.3488e-07	0.23	4	1.82553344e+08	1.9372e-07	0.23	4	1.82553207e+08	9.4650e-07
c-56	35910	0.57	10	1.20694938e+05	8.9080e-07	0.36	6	1.20694997e+05	4.0154e-07	1.03	17	1.20695043e+05	2.2158e-08
mark3juc080sc	36609	0.85	22	1.04857597e+06	2.9881e-08	0.61	16	1.04857589e+06	1.0646e-07	1.07	26	1.04857546e+06	5.1922e-07
rajat15	37261	2.20	32	3.26846386e+05	6.8038e-07	1.18	18	3.26846337e+05	8.2986e-07	3.95	53	3.26846394e+05	6.5567e-07
c-57	37833	1.30	22	7.27009599e+04	3.6908e-07	0.95	16	7.27009678e+04	2.6074e-07	2.08	33	7.27009493e+04	5.1534e-07
c-58	37595	0.76	10	5.41547064e+04	1.3915e-07	0.48	6	5.41546740e+04	7.3683e-07	0.71	9	5.41546740e+04	7.3826e-07
rajat22	39899	1.51	32	9.98382440e+05	4.1202e-07	1.25	27	9.98381916e+05	9.3669e-07	2.33	46	9.98382337e+05	5.1464e-07
c-59	41282	4.23	56	8.38555820e+03	6.6516e-07	1.62	23	8.38555995e+03	4.5611e-07	7.33	89	8.38555786e+03	7.0585e-07
net150	43520	13.55	40	1.45592976e+02	6.0882e-07	13.69	44	1.45592924e+02	9.6460e-07	46.77	117	1.45592932e+02	9.1039e-07
mark3jac100sc	45769	1.08	20	1.04857567e+06	3.1205e-07	0.72	13	1.04857528e+06	6.8813e-07	1.68	29	1.04857581e+06	1.8088e-07
c-65	48066	1.59	22	1.31412930e+05	9.2458e-07	1.61	22	1.31413049e+05	2.1779e-08	2.80	37	1.31412931e+05	9.1840e-07
c-66	49989	1.41	16	1.70373493e+04	7.9667e-07	1.28	15	1.70373539e+04	5.2388e-07	1.84	21	1.70373466e+04	9.5351e-07
rajat26	51032	2.32	34	8.26005894e+05	5.7106e-07	1.13	17	8.26006120e+05	2.9695e-07	3.63	49	8.26005835e+05	6.4202e-07
c-64b	51035	1.33	12	2.00080770e+05	4.0142e-07	0.98	9	2.00080782e+05	3.4412e-07	1.46	13	2.00080704e+05	7.3323e-07
ecl32	51993	7.78	49	9.61854529e+03	2.6354e-07	4.47	29	9.61854112e+03	6.9667e-07	13.50	82	9.61853979e+03	8.3447e-07
dictionary28	52652	5.94	39	2.50615848e+01	8.7789e-07	7.30	49	2.50615880e+01	7.5024e-07	11.18	69	2.50616053e+01	6.0262e-08
mark3jac120sc	54929	1.75	24	1.04857521e+06	7.5451e-07	1.16	16	1.04857600e+06	4.0445e-09	2.54	33	1.04857523e+06	7.3748e-07

Table 2. Numerical Results LBFGS_Ar with $m = 5$, 6, and 7

		m=5				m=6				m=7			
Name	Order	Time	Iter	Largest eig	Relerr	Time	Iter	Largest eig	Relerr	Time	Iter	Largest eig	Relerr
sts4098	4098	0.39	36	3.07102311e+08	6.2645e-07	0.49	43	3.07102311e+08	6.2645e-07	0.58	50	3.07102203e+08	9.7830e-07
bcsstk28	4410	0.68	36	7.69620714e+08	8.9428e-07	0.82	43	7.69620714e+08	8.9428e-07	0.92	50	7.69620803e+08	7.7939e-07
mhd4800b	4800	0.36	36	2.19626811e+00	2.3526e-07	0.63	61	2.19626749e+00	5.2006e-07	0.59	57	2.19626752e+00	5.0325e-07
bcsstk16	4884	3.17	96	4.94316093e+09	9.5098e-07	3.86	115	4.94316094e+09	9.5049e-07	3.54	106	4.94316271e+09	5.9240e-07
c-29	5033	0.09	11	2.22804953e+06	1.6251e-07	0.11	13	2.22804953e+06	1.6193e-07	0.10	10	2.22804939e+06	2.2384e-07
c-30	5321	0.13	13	4.68199067e+06	6.9187e-08	0.16	15	4.68199067e+06	6.9187e-08	0.05	5	4.68199068e+06	6.6694e-08
c-33	6317	0.17	16	2.04540787e+05	2.1187e-07	0.19	19	2.04540787e+05	2.1187e-07	0.30	29	2.04540809e+05	1.0479e-07
rajat01	6833	1.94	106	4.21268214e+01	5.4479e-07	3.02	169	4.21268266e+01	4.2229e-07	5.22	295	4.21268327e+01	2.7796e-07
c-36	7479	0.38	26	8.74111275e+03	6.0434e-07	0.44	31	8.74111275e+03	6.0434e-07	1.82	120	8.74111190e+03	7.0081e-07
aft01	8205	0.10	7	9.99999996e+14	4.0915e-09	0.12	8	9.99999996e+14	4.0915e-09	0.05	3	9.99999768e+14	2.3164e-07
bloweybq	10001	0.15	13	4.99974905e+03	1.9941e-07	0.18	15	4.99974905e+03	1.9941e-07	0.17	15	4.99974859e+03	2.9246e-07
bcsstk17	10974	7.98	151	1.29606047e+10	8.5368e-07	9.74	181	1.29606047e+10	8.5369e-07	4.97	92	1.29606075e+10	6.3814e-07
bcsstk25	15439	3.60	81	1.06002016e+15	3.2542e-07	4.13	97	1.06002016e+15	3.2944e-07	4.66	106	1.06001968e+15	7.7707e-07
olafu	16146	11.56	116	9.47869511e+11	8.7377e-07	14.07	139	9.47869511e+11	8.7377e-07	1.49	15	9.47870222e+11	1.2329e-07
gyro_k	17361	17.45	121	3.65694879e+09	9.6745e-07	21.09	145	3.65694879e+09	9.6745e-07	14.44	99	3.65694914e+09	8.7105e-07
gyro	17361	17.52	121	3.65694879e+09	9.6745e-07	21.14	145	3.65694879e+09	9.6745e-07	26.07	176	3.65694884e+09	9.5323e-07
rajat27	20640	2.24	96	7.69114227e+05	9.8785e-07	2.74	115	7.69114227e+05	9.8785e-07	3.51	148	7.69114532e+05	5.9143e-07
c-50	22401	0.47	16	2.65859642e+05	9.9420e-08	0.57	19	2.65859642e+05	9.9420e-08	0.67	22	2.65859462e+05	7.7825e-07
as-22july06	22963	4.99	106	7.16129292e+01	9.9267e-07	6.07	127	7.16129292e+01	9.9266e-07	6.55	134	7.16129578e+01	5.9372e-07
ca-CondMat	23133	4.72	71	3.79540903e+01	5.9398e-07	5.81	85	3.79540905e+01	5.9082e-07	7.35	106	3.79540833e+01	7.7897e-07
c-52	23948	0.37	11	1.94346166e+15	4.2716e-07	0.44	13	1.94346166e+15	4.2716e-07	0.98	29	1.94346150e+15	5.1094e-07
mult_dcop_02	25187	0.95	21	1.25613744e+03	2.1937e-07	1.14	25	1.25613744e+03	2.1937e-07	1.30	29	1.25613725e+03	3.6594e-07
net100	29920	30.62	122	1.22667743e+02	8.5543e-07	37.63	146	1.22667743e+02	8.5543e-07	33.87	134	1.22667780e+02	5.5711e-07
bloweybl	30003	0.68	26	1.00007500e+02	5.3058e-09	0.83	31	1.00007500e+02	5.3058e-09	0.98	36	1.00007492e+02	9.1653e-08
c-53	30235	2.68	41	5.36188540e+03	3.4140e-07	3.20	49	5.36188540e+03	3.4140e-07	4.69	71	5.36188249e+03	8.8356e-07
c-54	31793	0.24	4	1.82553207e+08	9.4650e-07	0.24	4	1.82553207e+08	9.4650e-07	0.56	10	1.82553249e+08	7.1368e-07
c-56	35910	1.28	21	1.20695043e+05	2.2350e-08	1.54	25	1.20695043e+05	2.2350e-08	1.36	22	1.20695024e+05	1.7744e-07
mark3juc080sc	36609	1.34	32	1.04857505e+06	9.0663e-07	1.59	38	1.04857504e+06	9.1708e-07	2.47	57	1.04857554e+06	4.3501e-07
rajat15	37261	5.04	66	3.26846394e+05	6.5567e-07	6.15	79	3.26846394e+05	6.5567e-07	4.31	57	3.26846572e+05	1.1190e-07
c-57	37833	2.60	41	7.27009591e+04	3.7971e-07	3.17	49	7.27009493e+04	5.1554e-07	3.27	50	7.27009656e+04	2.9016e-07
c-58	37595	0.87	11	5.41546745e+04	7.2895e-07	1.06	13	5.41546745e+04	7.2895e-07	1.83	22	5.41547138e+04	2.4010e-09
rajat22	39899	3.18	61	9.98382573e+05	2.7838e-07	4.24	80	9.98382585e+05	2.6678e-07	5.69	106	9.98382352e+05	4.9993e-07
c-59	41282	9.40	111	8.38555786e+03	7.0585e-07	11.41	133	8.38555786e+03	7.0585e-07	9.67	113	8.38555603e+03	9.2350e-07
net150	43520	60.20	146	1.45592935e+02	8.9058e-07	76.14	175	1.45592935e+02	8.9058e-07	31.95	78	1.45592955e+02	7.5142e-07
mark3jac100sc	45769	2.12	36	1.04857581e+06	1.8088e-07	2.85	43	1.04857581e+06	1.8088e-07	2.99	50	1.04857572e+06	2.6358e-07
c-65	48066	4.77	61	1.31412970e+05	6.2151e-07	5.94	73	1.31412945e+05	8.1606e-07	5.26	64	1.31413001e+05	3.8396e-07
c-66	49989	2.35	26	1.70373466e+04	9.5351e-07	2.84	31	1.70373466e+04	9.5351e-07	4.67	50	1.70373599e+04	1.7263e-07
rajat26	51032	5.34	71	8.26005771e+05	7.1990e-07	6.48	85	8.26005753e+05	7.4159e-07	7.13	92	8.26005551e+05	9.8632e-07
c-64b	51035	1.86	16	2.00080704e+05	7.3323e-07	2.23	19	2.00080704e+05	7.3323e-07	3.51	30	2.00080706e+05	7.2070e-07
ecl32	51993	20.24	121	9.61853936e+03	8.8008e-07	22.59	133	9.61854608e+03	1.8076e-07	29.85	176	9.61854617e+03	1.7130e-07
dictionary28	52652	20.90	126	2.50615873e+01	7.7663e-07	25.44	151	2.50615873e+01	7.7663e-07	20.21	120	2.50615965e+01	4.0805e-07
mark3jac120sc	54929	3.21	41	1.04857519e+06	7.7698e-07	3.88	49	1.04857519e+06	7.7698e-07	4.46	57	1.04857563e+06	3.5535e-07

To more visually show the behavior of algorithm LBFGS_Ar at each case, we draw performance profiles of Dolan and Moré [12] as evaluation tool. That is, for a subset of the methods being analyzed, we plot the fraction P of problems for which any given method is within a factor τ of the best. Meanwhile, we use the iteration and CPU time consuming as performance measure, since they reflect the computational cost and the efficiency for each method. The performance profiles of algorithm LBFGS_Ar with different m are plotted in Figure 1. Most importantly, these tables indicate that the value $m = 3$ may be the best choice to achieve better performance.

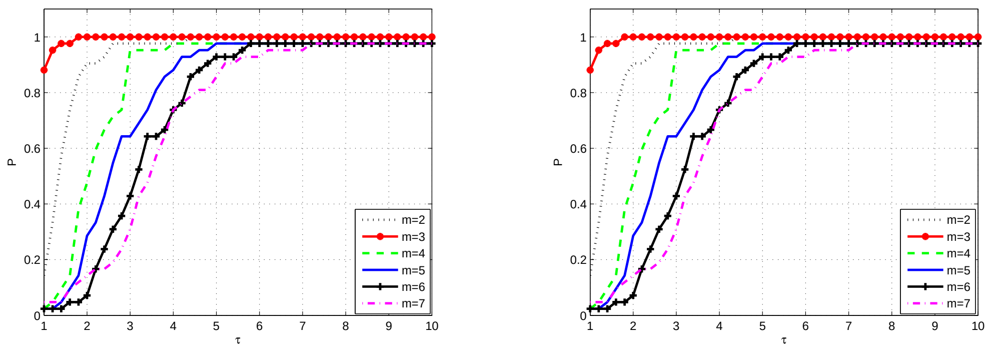

Figure 1: Performance profile based on iterations (left) and CPU time (right) for $m = 2, 3, 4, 5, 6, 7$.

To further investigate the performance or the accuracy of the proposed algorithm, we test against the Matlab command EIGS. The parameter values in EIGS are set as stated previously. We report the final largest eigenvalue in Table 3 together with the relative errors. It is clear from the table to see that LBFGS_Ar derives competitive solutions with EIGS in sense of lower relative errors. Taking everything together, we see that LBFGS_Ar provides an alternative approach to find the largest eigenvalues of symmetric positive definite matrixes and its performance is competitive with the compared one.

5. Conclusions

The first contribution of the paper is the designing of a L-BFGS algorithm to solve non-convex unconstrained minimization problems. In contrast to [26], the attractive character of the proposed method is the using of the Armijo line search instead of the Wolfe line search. We theoretically show that the proposed algorithm converges globally to a critical point of the non-convex function. The second contribution of the paper is application the algorithm in finding the largest eigenvalue of high dimensional symmetric positive definite matrix. The matrix largest eigenpair problem appears in many scientific or engineering fields, and receives many research activities especially in numerical algebra literature. Our algorithm belongs to the optimization category and aims to produce a critical point, which may be either a local maximum, a local minimum, or a saddle point. Nevertheless, the extensive numerical experiments illustrate that the proposed algorithm works well, and usually produce medium quality solutions from the point of view of the final relative error. To this end, although the proposed method does not obtain significant development as we have expected, we think that, the use of L-BFGS method for finding matrix largest eigenvalue is new, and its enhancement is still noticeable.

Table 3. Numerical Results of EIGS and LBFGS_Ar with $m = 3$

		EIGS	LBFGS_Ar(m=3)			
Name	Order	Largest eig	Time	Iter	Largest eig	Relerr
sts4098	4098	3.07102503e+08	0.18	18	3.07102309e+08	6.3118e-07
bcsstk28	4410	7.69621402e+08	0.23	13	7.69620972e+08	5.5984e-07
mhd4800b	4800	2.19626863e+00	0.16	18	2.19626681e+00	8.2796e-07
bcsstk16	4884	4.94316563e+09	1.06	34	4.94316126e+09	8.8430e-07
c-29	5033	2.22804989e+06	0.06	7	2.22804988e+06	4.2281e-09
c-30	5321	4.68199099e+06	0.06	6	4.68199037e+06	1.3354e-07
c-33	6317	2.04540830e+05	0.07	7	2.04540710e+05	5.8444e-07
rajat01	6833	4.21268444e+01	1.28	70	4.21268082e+01	8.5839e-07
c-36	7479	8.74111803e+03	0.16	11	8.74111647e+03	1.7815e-07
aft01	8205	1.00000000e+15	0.07	5	9.99999996e+14	4.0915e-09
bloweybq	10001	4.99975005e+03	0.11	10	4.99974636e+03	7.3898e-07
bcsstk17	10974	1.29606158e+10	2.64	54	1.29606050e+10	8.3157e-07
bcsstk25	15439	1.06002050e+15	0.86	21	1.06001945e+15	9.9397e-07
olafu	16146	9.47870339e+11	0.66	7	9.47869950e+11	4.1015e-07
gyro_k	17361	3.65695233e+09	4.43	34	3.65694955e+09	7.5972e-07
gyro	17361	3.65695233e+09	4.44	34	3.65694955e+09	7.5972e-07
rajat27	20640	7.69114987e+05	0.41	21	7.69114424e+05	7.3174e-07
c-50	22401	2.65859669e+05	0.23	8	2.65859469e+05	7.5052e-07
as-22july06	22963	7.16130003e+01	1.43	35	7.16129365e+01	8.9170e-07
ca-CondMat	23133	3.79541129e+01	1.92	34	3.79540890e+01	6.2849e-07
c-52	23948	1.94346249e+15	0.20	6	1.94346106e+15	7.3524e-07
mult_dcop_02	25187	1.25613771e+03	0.39	10	1.25613760e+03	8.5984e-08
net100	29920	1.22667848e+02	7.48	37	1.22667736e+02	9.1557e-07
bloweybl	30003	1.00007501e+02	0.23	10	1.00007448e+02	5.3359e-07
c-53	30235	5.36188723e+03	0.94	16	5.36188437e+03	5.3364e-07
c-54	31793	1.82553380e+08	0.23	4	1.82553344e+08	1.9372e-07
c-56	35910	1.20695045e+05	0.36	6	1.20694997e+05	4.0154e-07
mark3juc080sc	36609	1.04857600e+06	0.61	16	1.04857589e+06	1.0646e-07
rajat15	37261	3.26846609e+05	1.18	18	3.26846337e+05	8.2986e-07
c-57	37833	7.27009867e+04	0.95	16	7.27009678e+04	2.6074e-07
c-58	37595	5.41547139e+04	0.48	6	5.41546740e+04	7.3683e-07
rajat22	39899	9.98382851e+05	1.25	27	9.98381916e+05	9.3669e-07
c-59	41282	8.38556378e+03	1.62	23	8.38555995e+03	4.5611e-07
net150	43520	1.45593064e+02	13.69	44	1.45592924e+02	9.6460e-07
mark3jac100sc	45769	1.04857600e+06	0.72	13	1.04857528e+06	6.8813e-07
c-65	48066	1.31413052e+05	1.61	22	1.31413049e+05	2.1779e-08
c-66	49989	1.70373629e+04	1.28	15	1.70373539e+04	5.2388e-07
rajat26	51032	8.26006365e+05	1.13	17	8.26006120e+05	2.9695e-07
c-64b	51035	2.00080850e+05	0.98	9	2.00080782e+05	3.4412e-07
ecl32	51993	9.61854782e+03	4.47	29	9.61854112e+03	6.9667e-07
dictionary28	52652	2.50616068e+01	7.30	49	2.50615880e+01	7.5024e-07
mark3jac120sc	54929	1.04857600e+06	1.16	16	1.04857600e+06	4.0445e-09

Acknowledgements

We are grateful to the associate editor and two anonymous referees for their valuable comments and suggestions which have helped us improve the presentation of this paper greatly. The work was supported by the Major State Basic Research Development Program of China (973 Program) (Grant No. 2015CB856003), the National Natural Science Foundation of China (Grant No. 11471101), and the Program for Science and Technology Innovation Talents in Universities of Henan Province (Grant No. 13HASTIT050).

References

[1] P.-A. Absil, R. Mahony, and B. Andrews, *Convergence of the iterates of descent methods for analytic cost functions*, SIAM J. Optim., 16 (2005), 531-547.

[2] G. Auchmuty, *Duality for non-convex variational principles*, J. Differential Equations, 50 (1983), 80-145.

[3] G. Auchmuty, *Unconstrained variational principles for eigenvalues of real symmetric matrices*, SIAM J. Math. Anal., 20 (1989), 1186-1207.

[4] Z. Bai, J. Dongarra, A. Ruhe, and H. van der Vorst, *Templates for the solution of algebraic eigenvalue problems: A practical guide*, SIAM, Philadelphia, 2000.

[5] R. H. Byrd and J. Nocedal, *A tool for the analysis of quasi-Newton methods with application to unconstrained minimization*, SIAM J. Numer. Anal., 26 (1989), 727-739.

[6] R. H. Byrd, J. Nocedal and Y. Yuan, *Global convergence of a class of quasi-Newton methods on convex problems*, SIAM J. Numer. Anal., 24 (1987), 1171-1189.

[7] A. R. Conn, N. I. M. Gould, and Ph. L. Toint, *CUTE: constrained and unconstrained testing environment*, ACM Trans. Math. Softw. 21 (1995) 123-160.

[8] Y. H. Dai, *Convergence properties of the BFGS algorithm*, SIAM J. Optim., 13 (2002), 693-701.

[9] T. A. Davis and Y. H. Hu, *The university of Florida sparse matrix collection*, ACM Trans. Math. Softw., 2010.

[10] J. Demmel and K. Veselić, *Jacobis Method is More Accurate than QR*, SIAM. J. Matrix Anal. Appl., 13 (1992), 1204C1245.

[11] J. Demmel, O. A. Marques, B. N. Parlett, and C. Vömel, *Performance and accuracy of LAPACK's symmetric tridiagonal eigensolvers*, SIAM J. Sci. Comput., 30 (2008), 1508-1526.

[12] E. D. Dolan and J. J. Moré. *Benchmarking optimization software with performance profiles*, Math. Program., 91 (2002), 201-213.

[13] H. Gao, Y. H. Dai, and X. Tong, *Barzilai-Borwein-like methods for the extreme eigenvalue problem*, J. Ind. Manag. Optim., 11 (2015), 999-1019.

[14] G. H. Golub and C. F. Van Loan, *Matrix computation[M]*, 3rd ed. john Hopkins University Press, Baltimore, MD, 1996.

[15] B. Jiang and Y. H. Dai, *Feasible Barzilai-Borwein-like methods for extreme symmetric eigenvalue problems*, Optim. Methods Softw., 28 (2013), 756-784.

[16] D. H. Li and M. Fukushima, *A modified BFGS method and its global convergence in nonconvex minimization*, J. Comput. Appl. Math., 129 (2001), 15-35.

[17] D. H. Li and M. Fukushima, *On the global convergence of the BFGS method for nonconvex unconstrained optimization problems*, SIAM J. Optim., 11 (2001), 1054-1064.

[18] D. Liu and J. Nocedal, *On the limited memory BFGS method for large-scale optimization*, Math. Program., 45 (1989), 503-528.

[19] W. F. Mascarenhas, *The BFGS method with exact line searchs fails for non-convex objective functions*, Math. Program., 99 (2004), 49-61.

[20] M. Mongeau and M. Torki, *Computing eigenelements of real symmetric Matrices via optimization*, Comput. Optim. Appl., 29 (2004), 263-287.

[21] J. Nocedal, *Updating quasi-Newton matrices with limited storage*, Math. Comput., 35 (1980), 773-782.

[22] A. H. Sameh and J.A. Wisniewski. *A trace minimization algorithm for the generalized eigenvalue problem compuations*, SIAM J. Numer. Anal., 19 (1982), 1243-1259.

[23] Z. Wei, G. Yu, G. Yuan, and Z. Lian, *The superlinear convergence of a modified BFGS-type method for uconstrained optimization*, Comput. Optim. Appl., 29 (2004), 315-332.

[24] Z. Wei, G. Li, and L.Q. Qi, *New quasi-Newton methods for unconstrained optimization problems*, Appl. Math. Comput., 175 (2006), 1156-1188.

[25] Z. Wei, G. Li, and L.Q. Qi, *Global convergence of the Polak-Ribiere-Polyak conjugate gradient method with inexact line searches for nonconvex unconstrained optimization problems*, Math. Comput., 77 (2008), 2173-2193

[26] Y. Xiao, T. Li, and Z. Wei, *Global convergence of a modified limited memory BFGS method for nonconvex minimization*, Acta Math. Appl. Sin.-E, 29 (2013), 555-566.

[27] Y. Xiao, Q. Wang, and D. Wang, *Notes on the Dai_Yuan_Yuan modified spectral gradient method*, J. Comput. Appl. Math., 234 (2010) 2986-2992.

[28] Y. Xiao, Z. Wei, and Z. Wang, *A limited memory BFGS-type method for large-scale unconstrained optimization*, Comput. Math. Appl., 56 (2008), 1001-1009.

[29] C.X. Xu and J.Z. Zhang, *A survey of quasi-Newton equations and quasi-Newton methods for optimization. Optimization and numerical algebra*, Ann. Oper. Res., 103 (2001), 213-234.

[30] J. Z. Zhang, N.Y. Deng, and L.H. Chen, *New quasi-Newton equation and related methods for unconstrained optimization*, J. Optim. Theory Appl., 102 (1999), 147-167.

[31] J. Zhang, Y. Xiao, and Z. Wei, *Nonlinear conjugate gradient methods with sufficient descent condition for large-scale unconstrained optimization*, Math. Prob. Eng. 2009 (2009), Article ID 243290, 16 pages.

[32] L. Zhang, *A globally convergent BFGS method for nonconvex minimization without line searches*, Optim. Method. Softw., 20 (2005), 737-747.

[33] W. Zhou and L. Zhang, *Global convergence of the nonmonotone MBFGS method for nonconvex unconstrained minimization*, J. Comput. Appl. Math., 223 (2009), 40-47.